


1-2-2013

# Towards Personalized Medicine Using Systems Biology And Machine Learning

Calin Voichita  
*Wayne State University,*

Follow this and additional works at: [http://digitalcommons.wayne.edu/oa\\_dissertations](http://digitalcommons.wayne.edu/oa_dissertations)

 Part of the [Artificial Intelligence and Robotics Commons](#), and the [Bioinformatics Commons](#)

---

## Recommended Citation

Voichita, Calin, "Towards Personalized Medicine Using Systems Biology And Machine Learning" (2013). *Wayne State University Dissertations*. Paper 805.

This Open Access Dissertation is brought to you for free and open access by DigitalCommons@WayneState. It has been accepted for inclusion in Wayne State University Dissertations by an authorized administrator of DigitalCommons@WayneState.

# TOWARDS PERSONALIZED MEDICINE USING SYSTEMS BIOLOGY AND MACHINE LEARNING

by

CĂLIN VOICHIȚA

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

**DOCTOR OF PHILOSOPHY**

2013

MAJOR: COMPUTER SCIENCE  
(Bioinformatics)

Approved by:

\_\_\_\_\_  
Advisor

\_\_\_\_\_  
Date

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# DEDICATION

*To*

*My parents, Lucia and Doru Voichita*

*For their endless love and support that made my entire journey possible*

## ACKNOWLEDGMENTS

First, and foremost I would like to thank my advisor, Dr. Sorin Draghici, who taught me how to find and ask the important questions that are worth investigating. I am thankful for his patience to my mistakes and invaluable feedback he provided.

Many thanks to my committee members: Dr. Robert G. Reynolds, Dr. Daniel Grosu and Dr. Michael Tainsky for their invaluable reviews and suggestions. In addition, I would like to thank Dr. Tainsky for the challenges he threw at me regarding cancer classification and biomarker discovery.

I would like to thank all the past and current members of the Intelligent Systems and Bioinformatics laboratory (ISBL) for their encouragement and support both in and out of the lab. Many thanks to Dr. Purvesh Khatri who helped me tremendously in my first years as a graduate student. I am thankful to Michele Donato for our great collaboration and friendship through the ups and down of our graduate endeavor. Zhonghui Xu thought me a thing or two about determination and hard work through our collaboration on the machine learning techniques.

Without the continuous support and encouragement from my family, I could not have completed the doctoral studies. My parents taught me the most important lessons in life, determination and optimism, without which my journey would have been less exciting and rewarding.

I am grateful for both for the new friendships I developed through my graduate studies and the old ones that stood the test of time and space.



# TABLE OF CONTENTS

<b>Dedication</b> . . . . .	<b>ii</b>
<b>Acknowledgments</b> . . . . .	<b>iii</b>
<b>List of Tables</b> . . . . .	<b>vii</b>
<b>List of Figures</b> . . . . .	<b>viii</b>
<b>Chapter 1: Introduction</b> . . . . .	<b>1</b>
<b>Part I Patient Subtyping Using Machine Learning</b> . . . . .	<b>4</b>
<b>Chapter 2: Support Vector Machines and classifier ensembles</b> . . . . .	<b>4</b>
2.1 Support Vector Machines . . . . .	5
2.2 Calibrating Probability Scores . . . . .	9
2.3 Classification ensembles . . . . .	11
2.4 Limitations . . . . .	12
<b>Chapter 3: Posterior probabilities for classifier ensembles</b> . . . . .	<b>14</b>
3.1 Posterior probabilities based on $z$ -score . . . . .	14
3.1.1 Standard Gaussian CDF fitting . . . . .	17
3.1.2 Logistic fitting . . . . .	17
3.1.3 Isotonic regression . . . . .	18
3.2 Experiment design . . . . .	20
3.3 Performance on artificial data . . . . .	22
3.4 Accuracy of the predictions on benchmark data sets from the UCI machine learning repository . . . . .	26
3.5 Quality of the posterior probability estimates on benchmark data sets from the UCI machine learning repository . . . . .	28
3.6 $z$ -score drawbacks, alternative statistics and limitations . . . . .	34
3.7 Summary . . . . .	35
<b>Chapter 4: Patient subtyping using classification uncertainty</b> . . . . .	<b>38</b>
4.1 Related work on classification with rejection . . . . .	39
4.2 Uncertainty based on USVM margin . . . . .	39

4.2.1	Iris data set example . . . . .	41
4.3	Uncertainty based on posterior probability . . . . .	43
4.4	Automatic detection of uncertainty . . . . .	44
4.5	Performance on artificial data . . . . .	46
4.6	Performance on benchmark data sets from the UCI machine learning repository . . . . .	52
4.7	Results on clinical samples . . . . .	53
4.8	Summary . . . . .	57
<b>Part II Understanding the Disease Mechanism Using Systems Biology . . . . .</b>		<b>59</b>
<b>Chapter 5: Biological networks and available analysis methods . . . . .</b>		<b>59</b>
5.1	Biological pathways and available resources . . . . .	60
5.2	Gene set analysis methods . . . . .	64
5.2.1	Over representation analysis . . . . .	65
5.2.2	Functional class scoring . . . . .	66
5.2.3	Non-independent gene sets . . . . .	67
5.3	Topology based analysis methods . . . . .	68
<b>Chapter 6: The impact analysis . . . . .</b>		<b>71</b>
6.1	Impact factor . . . . .	71
6.2	Significance of the impact factor . . . . .	75
6.3	Performance analysis of the impact factor . . . . .	80
6.4	System of equations and alternative significance computation . . . . .	89
6.5	Impact analysis that uses significance of individual genes . . . . .	91
6.5.1	Colorectal cancer case study . . . . .	95
6.5.2	Rank comparison over a pool of 24 datasets . . . . .	96
6.6	Summary . . . . .	96
<b>Chapter 7: Cut-off free impact analysis . . . . .</b>		<b>99</b>

7.1	Method	101
7.2	Results and discussion	102
7.3	Multiple sclerosis case study	104
7.4	Summary	106
<b>Chapter 8: ROntoTools User Guide</b>		<b>108</b>
8.1	Pathway database	108
8.2	Experiment data	113
8.3	Setting the node weights	115
8.4	Pathway analysis and results summary	115
8.5	Graphical representation of results	117
8.6	Future work	122
<b>Chapter 9: Estimating Gene Contributions in Signaling Pathways</b>		<b>123</b>
9.1	Genetic algorithms	123
9.2	Determining gene contributions	125
9.3	Training	126
9.4	Evaluation	128
9.5	Summary	131
<b>Part III Conclusions</b>		<b>133</b>
<b>Chapter 10: Conclusions and future work</b>		<b>133</b>
<b>Appendix</b>		<b>137</b>
<b>Bibliography</b>		<b>173</b>
<b>Abstract</b>		<b>195</b>
<b>Autobiographical Statement</b>		<b>197</b>

## LIST OF TABLES

Table 3.1: SVM model parameters and data sets summary . . . . .	23
Table 3.2: Z-bag error rate comparison . . . . .	29
Table 3.3: Z-bag posterior probability estimates comparison . . . . .	32
Table 4.1: Performance of uncertainty SVM on pima data set . . . . .	53
Table 4.2: Performance of uncertainty SVM on mnist data set . . . . .	54
Table 6.1: Comparison of two models that incorporate gene significance with the model without on a colorectal cancer data set . . . . .	95
Table 7.1: Comparison of two cut-off free models with the original model (SPIA) . . . . .	103
Table 7.2: Significant pathways in common among the four datasets . . . . .	105
Table 9.1: The pool of data sets used for the parameter estimation . . . . .	127

## LIST OF FIGURES

Figure 2.1:	Binary classification example . . . . .	12
Figure 3.1:	$z$ score as confidence in the sign of $F$ . . . . .	16
Figure 3.2:	Distribution of $F$ values for individual samples . . . . .	18
Figure 3.3:	Comparison of fitted posterior probability models . . . . .	19
Figure 3.4:	Procedure to asses the performance of the proposed bagging SVM ensemble . . . . .	21
Figure 3.5:	Comparison of estimated posterior with true posterior on ringnorm data with sample size of . . . . .	24
Figure 3.6:	Comparison of estimated posterior with true posterior on ringnorm data with sample size of 300 . . . . .	25
Figure 3.7:	Performance on ringnorm artificial data . . . . .	27
Figure 3.8:	Pair-wise comparison of the probability to predict the correct class . . . . .	33
Figure 3.9:	$z$ score pattern stabilization . . . . .	36
Figure 4.1:	Example of how uncertainty region is determined in the feature space . . . . .	40
Figure 4.2:	Classifying with uncertainty on the linearly separable setosa and versicolor classes of the <i>iris</i> dataset . . . . .	42
Figure 4.3:	Classifying with uncertainty on the non-linearly separable versicolor and virginca classes of the <i>iris</i> dataset . . . . .	44
Figure 4.4:	Contingency table for classifying with uncertainty . . . . .	45
Figure 4.5:	Class dependent distributions of each feature in the artificial data sets . . . . .	47
Figure 4.6:	Uncertainty region comparison on the twonorm artificial data set . . . . .	49
Figure 4.7:	Uncertainty region comparison on the ringnorm artificial data set . . . . .	50
Figure 4.8:	Variance of the uncertainty threshold . . . . .	51
Figure 4.9:	Evolution of the accuracy and uncertainty percentage with regards to geometric margin . . . . .	54
Figure 4.10:	Sample of errors declared as uncertain by USVM on the mnist data set . . . . .	55
Figure 4.11:	Comparison of SVM and USVM prediction on the MLL data set . . . . .	57
Figure 5.1:	Example of KEGG signaling pathway: Apoptosis . . . . .	61
Figure 5.2:	The types of interactions in a KEGG signaling pathway . . . . .	61

Figure 5.3:	Example of Reactome signaling pathway: Apoptosis execution phase	62
Figure 5.4:	Example of KEGG metabolic pathway: Fatty acid metabolism . . .	63
Figure 5.5:	Yeast PPI network . . . . .	63
Figure 5.6:	Gene set enrichment analysis . . . . .	67
Figure 5.7:	KEGG Insulin Signaling Pathway . . . . .	69
Figure 6.1:	Example of perturbation propagation . . . . .	75
Figure 6.2:	Regulation of actin cytoskeleton as impacted in breast cancer . . . .	76
Figure 6.3:	A performance comparison between classical approaches (hypergeometric, GSEA) and the impact analysis on lung adenocarcinoma . .	81
Figure 6.4:	Focal adhesion as impacted in lung adenocarcinoma . . . . .	83
Figure 6.5:	A performance comparison between classical approaches (hypergeometric, GSEA) and the impact analysis on breast cancer . . . . .	86
Figure 6.6:	The complement and coagulation cascade as affected by treatment with palmitate in a hepatic cell line . . . . .	87
Figure 6.7:	A performance comparison between classical approaches (hypergeometric, GSEA) and the impact analysis on hepatic cell line treated with palmitate . . . . .	88
Figure 6.8:	Shortcomings of analyses that use only fold change . . . . .	92
Figure 6.9:	Comparison of the two weighting schemas for the gene significance. .	94
Figure 6.10:	Performance analysis of impact analysis that incorporates significance over a pool of 24 data sets . . . . .	97
Figure 7.1:	The selection of DE genes is a severe truncation of the information available . . . . .	100
Figure 7.2:	Comparison of the two weighting schemas for the gene significance. .	102
Figure 7.3:	Performance analysis of cut-off free impact analysis over a pool of 24 data sets . . . . .	104
Figure 7.4:	Overlap of significant pathways over the four MS data sets . . . . .	106
Figure 7.5:	Overlap of significant pathways over four MS data sets after module extraction . . . . .	107
Figure 8.1:	ROntoTools download stats . . . . .	109
Figure 8.2:	Two-way plot of Pathway-Express result . . . . .	118
Figure 8.3:	Pathway level statistics . . . . .	119

Figure 8.4:	Perturbation propagation on the Thyroid cancer signaling pathway .	120
Figure 8.5:	Perturbation propagation on the T cell receptor signaling pathway .	121
Figure 9.1:	Evolution over all generations of the best and mean evaluations . . .	129
Figure 9.2:	Null distributions using random gene contributions . . . . .	130
Figure 9.3:	Comparison between the estimated and default gene contributions. .	131

# Chapter 1 Introduction

Once heralded as the holy grail, the capability of obtaining a comprehensive list of genes, proteins or metabolites that are different between a disease and normal phenotypes is routine today. And yet, the holy grail of high-throughput has not delivered so far. Even though such high-throughput comparisons have become relatively easy to perform, understanding the phenomena that determine the measured changes is as challenging as ever, if not more so. At the same time, we have started to understand that evolution of many diseases such as cancer, are the results of the interplay between the disease itself and the immune system of the host. It is now well accepted that cancer is not a single disease, but a “complex collection of distinct genetic diseases united by common hallmarks” [123]. The heterogeneity of diseases such as breast cancer is well recognized [147] and gene expression profiling has been used to identify at least four major subtypes: luminal A, luminal B, HER2+ and basal-like [148, 169]. In the past decade, important clinical advances in cancer treatments are attributed to molecularly targeted treatments aiming at specific genes such as estrogen receptor alpha ( $ER-\alpha$ ), HER2, EGFR, MET, BRAF, etc. The targeted treatments result in greater efficacy and lesser debilitating or dose limiting side effects [163]. This clearly proves that it is important to identify and appropriately treat each particular individual or disease subtype. However, our current understanding of disease subtypes appears to be very limited. Despite targeted treatment advances, targeted therapies often fail for some patients. For breast cancer, while 20% of tumors overexpress the HER2 oncogene, one-third of these fail to show response to HER2-targeted therapies right from the outset. Research and clinical studies present a similar story for anti-estrogen treatment of  $ER-\alpha$ -positive breast cancer and androgen ablation of androgen receptor positive (AR+) prostate cancer [34, 88]. Not all patients demonstrate an initial response, and for those who do, a significant number will develop resistance. The fact that a substantial fraction of patients with a given subtype of



cancer respond very differently to the same treatment, either immediately or later on, means that either the mechanism of action is not homogeneous in the same subtype or that patients have different defense mechanism against the same tumor subtype. For example, the second largest pharmaceutical company in the world has recently announced the discontinuation of a Phase III multi-million dollar clinical trial because it did not meet the primary objective of improving overall survival.<sup>1</sup> According to Dr. Mace Rothenberg, senior vice president of Clinical Development and Medical Affairs for Pfizer's Oncology Business Unit, Pfizer Inc. will continue to investigate the respective drug "in specific patient subsets within the heterogeneous patient population enrolled in this trial", which leads to the conclusion that the failure was most likely due to a previously unknown division in the study population.

Another aspect is related to the choice of optimal treatments. For example, cytotoxic chemotherapy remains the standard adjuvant therapy for lung cancer and it is not routinely recommended as part of the initial course of treatment for individuals with early stage disease [5, 203]. However, the high recurrence rates for stage I non-small cell lung cancer (NSCLC) raises the consideration that a subset of patients may benefit from adjuvant therapy. Indeed, recent multinational clinical trials show that adjuvant chemotherapy can significantly improve the survival of patients with advanced early-stage (Stage IB-II) disease [17]. It follows that the capability to prognosticate outcomes – e.g., which tumors are likely to recur after surgical resection – would allow for better disease management where only patients who will benefit are treated and others who will not do not receive unnecessary over-treatments.

The ultimate goal of personalized medicine is the ability to identify the specific mechanism of disease in each patient independently and provide specialized treatment accordingly. Currently, most of the treatments available are based on the type of disease. An intermediate step in achieving the ultimate goal is to be able to identify homogeneous disease subtypes and patient sub-groups. To this extent, one angle of this thesis is focused on increasing the

<sup>1</sup><http://press.pfizer.com/press-release/pfizer-discontinues-phase-3-study-inotuzumab-ozogamicin-relapsed-or-refractory-aggress>

prediction accuracy and prediction confidence of existing machine learning techniques. In particular, it is very important to distinguish between samples for which the prediction is trustworthy because they fall in a region of the input space that is rich in samples from one class, and samples for which the prediction is not very trustworthy either because they fall very near the decision boundary or because they fall in an area of input space very far from any input sample used in the training. These latter samples can in fact represent previously unknown disease subtypes or patient sub-groups. This problem is addressed in Part I by first describing the techniques already available in Chapter 2 and then proposing a novel method for computing classification confidence in Chapter 3. The problem of sub-grouping the patient population is addressed in Chapter 4.

Once these sub-groups have been identified, the outstanding problem is the adequate characterization of the disease mechanism (see Part II). The particular avenue we have chosen to follow in order to achieve this goal is to eliminate the gene selection stage currently the norm in the pathway analysis of high throughput data. The goal of pathway analysis, in the context of this thesis, is to take as input the experimental data (i.e., the expression changes between two phenotypes) and the pathway database (i.e., the set of pathways that describe various biological processes) and identify the pathways that are significantly impacted between the two phenotypes (see Chapters 5 and 6). It has been shown that the specific choices made in the selection of differentially expressed (DE) genes have a crucial impact on the specific set of genes selected [140]. Therefore, the results of the pathway analysis step which uses the set of DE genes as its input are also going to be highly dependent on these choices. By eliminating this step, one can eliminate the possibility of missing important genes. This will improve our ability to understand the disease mechanisms of different disease subtypes. Our cut-off free (i.e., no DE gene selection) approach is presented in Chapter 7 and Chapter 8 presents the implementation of these methods as released in Bioconductor [73].

# Part I

## Patient Subtyping Using Machine Learning

### Chapter 2 Support Vector Machines and classifier ensembles

In this part of the thesis we use machine learning to focus on patient classification, diagnosis and subtyping. The ability to identify the correct disease subtype or stage of a patient is crucial to the follow-up investigations and treatment. Moreover, identifying previously unknown disease subtypes can lead to better design of clinical trials and can reduce the drug development time [69]. To this extent we focused on improving the performance of one of most commonly used algorithms for classification, the Support Vector Machines (SVMs) [18, 36, 191].

In this chapter we introduce necessary background information on SVMs that will be used in the later chapters. We present the theory behind the maximal marginal classifier, as well as different techniques to calibrate posterior probabilities to the output of the SVM classifiers. In addition, techniques to use SVMs as part of ensemble classifiers are presented. The chapter concludes by outlining the current SVM limitations.

## 2.1 Support Vector Machines

When building an SVM classifier, the input  $(\mathbf{x}^{(i)}, Y^{(i)})$  is a set of labeled points in  $n$ -dimensional input space such that  $\mathbf{x}^{(i)} \in \mathbb{R}^n$ . For binary classification, the label  $Y^{(i)}$  belongs to  $\{-1, 1\}$ , while for an  $m$ -class classification  $Y^{(i)} \in \{1, 2, \dots, m\}$ . Without loss of generality the following discussion will refer to the binary classification problem. In the multi-class scenario different techniques can be used to divide the problem in a set of binary classification problems, such as one-vs-all or one-vs-one. The predictions of the binary classification problems are then combined to a final prediction using a voting schema for example.

Using the labeled input points, SVM finds a function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  such that for a given input  $\mathbf{x}$ ,  $f(\mathbf{x}) \geq 0$ , if  $\mathbf{x}$  belongs to the class denoted by 1, otherwise  $f(\mathbf{x}) < 0$ . The equation  $f(\mathbf{x}) = 0$  defines a hyperplane that is used for classification of unknown samples. When the input consists of linearly separable classes, it is easy to find such a hyperplane using:

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b = \sum_{k=1}^n w_k x_k + b \quad (2.1)$$

where,  $\mathbf{w}$  is the normal vector of the hyperplane defined by  $f(\mathbf{x}) = 0$  and  $b$  is the offset from the origin. If  $b = 0$ , the hyperplane passes through the origin.

However, finding such a function is more complex in a non-linearly separable data set. The complexity of the target function  $f(\mathbf{x})$  depends on the way input data is represented. A common pre-processing practice in machine learning is to map the input data into another space where the two classes will be linearly separable:

$$\mathbf{x} = (x_1, \dots, x_n) \mapsto \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \dots, \phi_n(\mathbf{x})) \quad (2.2)$$

The new space is often referred to as a *feature space* in the literature. In the new feature

space,  $f(\mathbf{x})$  is represented as:

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \phi(\mathbf{x}) \rangle + b = \sum_{k=1}^n w_k \phi_k(\mathbf{x}) + b \quad (2.3)$$

It can be shown that  $f(\mathbf{x})$  can be expressed as a linear combination of the input data points [38]:

$$f(\mathbf{x}) = \sum_{i=1}^l \alpha_i Y^{(i)} \langle \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}) \rangle + b \quad (2.4)$$

where  $l$  is the number of samples in the training set. The parameter  $\alpha_i$  is referred to as the *embedding strength* of the pattern  $\mathbf{x}^{(i)}$ , which is proportional to the number of times the pattern  $\mathbf{x}^{(i)}$  is misclassified during the training.

However, mapping the input space into a new feature space can be a time-consuming operation since the feature space is very likely higher dimensional. In addition, due to higher dimensionality, it is more difficult to find the hyperplane for classification. Therefore, SVMs employ a shortcut through the use of a *kernel* function that allows the inner product in Eq. 2.4 to be computed without explicitly mapping the input points into the feature space. A kernel function is defined as:

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle \quad (2.5)$$

Examples of such kernels include the *linear* (L), *radial* (R) and *polynomial* (P) kernels:

$$L : K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \mathbf{x}^{(i)} * \mathbf{x}^{(j)} \quad (2.6)$$

$$R : K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = e^{-r * |\mathbf{x}^{(i)} - \mathbf{x}^{(j)}|^2} \quad (2.7)$$

$$P : K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = (r * \mathbf{x}^{(i)} * \mathbf{x}^{(j)} + coef0)^{degree} \quad (2.8)$$

Using Eq. 2.5, we can rewrite Eq. 2.4 as:

$$f(\mathbf{x}) = \sum_{i=1}^l \alpha_i Y^{(i)} K(\mathbf{x}^i, \mathbf{x}) + b \quad (2.9)$$

The definition of the hyperplane,  $f(\mathbf{x}) = 0$ , determines the type of the classifier. The simplest classifier is a *maximal margin* classifier, which is only suitable for input data that is linear separable in the feature space. The geometric margin  $\gamma_i$  of a pattern  $\mathbf{x}^{(i)}$  is defined as the Euclidean distance from the given pattern to the separating hyperplane. The geometric margin of a training set  $S$  with respect to a given hyperplane is the minimum geometric margin of all patterns in the training set  $S$ . The margin  $\gamma$  of a training set  $S$  is the maximum geometric margin over all possible hyperplanes. The maximal margin classifier, as the name implies, finds the hyperplane which maximizes the geometric margin of the training set. It can be shown that finding the maximal margin hyperplane is equivalent to the following optimization problem [38]:

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \langle \mathbf{w} \cdot \mathbf{w} \rangle \\ \text{s.t.} \quad & Y^{(i)} (\mathbf{w}^t \mathbf{x}^{(i)} + b) \geq 1, \quad i = 1, \dots, l \end{aligned} \quad (2.10)$$

The hyperplane obtained by solving this optimization problem realizes the margin  $\gamma = \frac{1}{\|\mathbf{w}\|_2}$ . It is often easier to solve an optimization problem in its dual form than its primal form because of the inequality constraints. The optimization problem above can be expressed in its dual form as:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l Y^{(i)} Y^{(j)} \alpha_i \alpha_j \langle \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} \rangle \\ \text{s.t.} \quad & \sum_{i=1}^l Y^{(i)} \alpha_i = 0, \quad \alpha_i \geq 0, \quad i = 1, \dots, l \end{aligned}$$

Because the dual formulation only consists of an inner product between patterns of the input data, we can use a kernel function as defined in Eq. 2.5 to find the optimal hyperplane in

the feature space. Hence, the optimization problem can be re-written as [38]:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l Y^{(i)} Y^{(j)} \alpha_i \alpha_j K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \\ \text{s.t.} \quad & \sum_{i=1}^l Y^{(i)} \alpha_i = 0, \alpha_i \geq 0, i = 1, \dots, l \end{aligned}$$

The optimal hyperplane obtained using the dual optimization problem realizes the margin:

$$\gamma = \frac{1}{\|\mathbf{w}\|_2} = \left( \sum_{i \in SV} \alpha_i^* \right)^{-\frac{1}{2}} \quad (2.11)$$

where,  $\alpha^*$  is the solution of the dual optimization problem and  $SV$  represents the set of support vectors (i.e., subset of the input data for which  $\alpha_i^* \neq 0$ ).

The maximal margin classifiers have limited applications since they can only be applied to data sets that are linearly separable in the feature space. This limitation can be avoided by using a *soft margin optimization* technique. The classifier that uses the soft margin optimization allows misclassification of some of the samples during training. This training error is controlled by slack variables. In other words, we need to find a hyperplane such that the constraint of the primal optimization problem above is modified as

$$\begin{aligned} \text{s.t.} \quad & Y^{(i)}(\langle \mathbf{w} \cdot \mathbf{x}^{(i)} \rangle + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, \dots, l \end{aligned}$$

The generalization error is bounded by either the 2-norm or the 1-norm of the slack vector [38]. The 1-norm soft margin optimization problem is described as

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_{i=1}^l \xi_i \\ \text{s.t.} \quad & Y^{(i)}(\langle \mathbf{w} \cdot \mathbf{x}^{(i)} \rangle + b) \geq 1 - \xi_i, \xi_i \geq 0 \end{aligned} \quad (2.12)$$

and its dual as

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l Y^{(i)}Y^{(j)}\alpha_i\alpha_j K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \\ \text{s.t.} \quad & \sum_{i=1}^l Y^{(i)}\alpha_i = 0, C \geq \alpha_i \geq 0, i = 1, \dots, l \end{aligned} \quad (2.13)$$

Using the solution of the optimization problem ( $\alpha_i^*$  and  $b^*$ ) the optimal decision function  $h^*(X)$  is defined as:

$$h^*(\mathbf{x}) = \text{sign}(f^*(\mathbf{x})) = \text{sign}\left(\sum_{i \in SV} \alpha_i^* Y^{(i)} K(\mathbf{x}^{(i)}, \mathbf{x}) + b^*\right) \quad (2.14)$$

where  $SV$  is the set of support vectors (i.e., the set of input samples for which  $\alpha_i^* \neq 0$ ),  $X$  is a new testing sample, and  $f^*$  is the optimal decision function. This hyperplane realizes the margin:

$$\gamma = \left( \sum_{i,j \in SV} Y^{(i)}Y^{(j)}\alpha_i^*\alpha_j^* K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \right)^{-\frac{1}{2}} \quad (2.15)$$

## 2.2 Calibrating Probability Scores

The decision function,  $h^*(\mathbf{x})$  (see Eq. 2.14), is not a probabilistic output. In some applications, having a high accuracy or a large area under the receiver operating characteristic (ROC) is not enough, and it is important to obtain accurate probability estimates. Efforts have been made to assess how well calibrated the output scores are for a given algorithm [40], and to create transformations that re-scale the scores back into calibrated probability estimates [211]. Two popular parametric and non-parametric approaches are Platt's scaling [145] and isotonic regression [154] respectively. A common feature of both methods is that the resulting transformations are monotonically increasing functions, which is what would be expected from such transformations.

Predicting good probabilities is not an easy task, since in practice the true posterior probability  $P(Y = 1|X)$  is not known. However, one sanity check is the requirement that



the posterior probabilities are well calibrated. This means that for any interval of probabilities  $[p_1, p_2]$ , the probability of drawing a positive example given the classifier predicts  $\hat{P}(Y = 1|X) \in [p_1, p_2]$  should also be in  $[p_1, p_2]$ . Previous studies have suggested that many classifiers, including naive Bayes and maximum margin methods, do not predict well calibrated probabilities [211, 145]. It has been shown that naive Bayes models that make simplistic assumptions about the probability structure push the posterior towards 0 and 1, while maximum margin methods such as SVM and boosted trees push away from the extreme probabilities [135]. In addition, it has been shown that Platt's scaling is effective for maximum margin methods, while it is less suited for naive Bayes [135]. Isotonic regression is also effective, but is inferior for smaller datasets. Since the calibration process requires internal cross validation, which makes the effective training data even smaller, Platt's scaling is preferred in SVMs [135].

Platt's model fits the posterior probability to  $f^*(\mathbf{x}^{(i)})$  (i.e.,  $p(Y = 1|f^*)$ ) [145]. Based on empirical data, he observed that the class-conditional densities of  $f^*$  are exponential and calculated the posterior probability as:

$$p(Y = 1|f^*) = \frac{1}{1 + e^{Af^*+B}} \quad (2.16)$$

The parameters  $A$  and  $B$  can be computed based on the input data using maximum likelihood estimation (MLE). Platt used the following regularized maximum likelihood optimization problem:

$$\min_{A,B} - \sum_i \left( t_i \log(p_i) + (1 - t_i) \log(1 - p_i) \right) \quad (2.17)$$

where  $p_i = p(Y^{(i)} = 1|f^*(\mathbf{x}^{(i)}))$  and  $t_i = \begin{cases} \frac{N_++1}{N_++2} & : Y^{(i)} = +1 \\ \frac{1}{N_-+2} & : Y^{(i)} = -1 \end{cases}$  with  $N_+$  and  $N_-$  the number of positive labels and the number of negative labels respectively.

## 2.3 Classification ensembles

By definition, an ensemble of classifiers is a collection of base classifiers, together with an appropriate aggregation technique that will combine the output of each individual base classifier into an ultimate prediction. Among the existing ensemble building methods, bagging and boosting received a lot of attention. In bagging, or bootstrap aggregating [19], the base classifiers are trained independently on bootstrapping samples drawn with replacement from the entire training set. The aggregation is typically based on majority voting, i.e. each base classifier accounts for one vote in predicting the class label; the one receiving most votes will be the ultimate predicted label. In boosting (most commonly AdaBoost [66]) the base classifiers are trained sequentially with weighted data. The weight of each training sample is associated with the performance of the preceding base classifier on that sample. A previously misclassified sample will be given more weight in training the subsequent base classifier. The final aggregated prediction is a weighted average calculated from the base classifiers, with base classifiers of higher accuracy having more weight. Independently of how the outputs of the base classifiers are aggregated, ensembles of classifiers have been proven to yield better accuracy than single classifiers.

Constructing ensemble classifiers using SVM as base classifiers is not a novel idea. Kim et. al. [116] employed bagging and boosting SVM with three aggregation methods, i.e. majority voting, LSE-based weighting, and double-layer hierarchical combining. All the proposed ensemble methods outperformed the single SVM. Valentini et. al. [185] proposed a variation of bagging, *Lobag*, that selects low-bias classifiers before aggregation is applied. Recently, Wang et. al. [200] analyzed and compared 4 different SVM ensemble techniques (bagging, AdaBoost, Arc-X4 and a modified AdaBoost) with 20 UCI data sets. In their experiments the bagged SVM ensemble performed as well or better than any other method.

In addition to classification accuracy, the confidence (i.e., the posterior probability) of the prediction carries an important role. The existing methods for generating posterior prob-

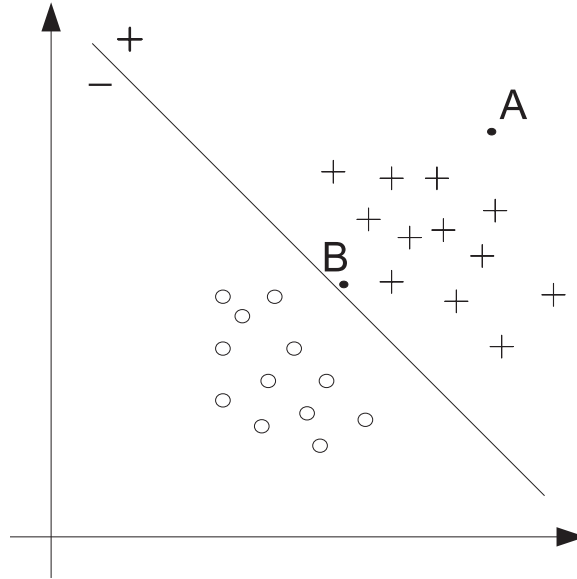


Figure 2.1: A simple example of a two dimensional binary classification problem. Once a hyperplane that separates the crosses (+) from the circles (o) is found, any new test example falling on the positive side of the hyperplane will be classified as a cross, with no distinction made between points such as A and B. However, this is undesirable since the prediction confidence of A should be greater than that of B. A small change in the training protocol could alter slightly the position of the hyperplane and B could in fact be classified as a circle. We propose for the prediction of this type of examples to be marked as uncertain. *Source:* [198].

abilities for classification ensembles are design to either combine the base classifier probability outputs (e.g., distribution summation [33, 3]) or to calibrate the aggregated decision value (e.g, naive Bayes [182]; Platt scaling, Isotonic regression, logistic correction for boosting [134]). The first category are likely to be computational intensive due to the requirement of fitting a posterior probability model for each base classifier. The second category were developed from the need to recover some of the information lost through the aggregation process.

## 2.4 Limitations

Many classification methods including SVMs have limited classification powers when a test instance lies either: i) very close to the decision boundary or ii) it is very different from the training samples. As described in the previous sections, once the SVM optimization problem is solved, the prediction of a test sample is given by  $sign(f^*(\mathbf{x}))$  (see Eq. 2.14).

Hence, considering the example in Fig. 2.1, once the decision boundary is set between the two classes (crosses and circles), the two points A and B are going to be predicted as crosses. However, one should be more confident that A is a cross in comparison to B. The prediction of the latter can be highly influenced by noise, or choice of training parameters. Currently, the output of classical SVM does not include any information allowing the user to make the distinction between A and B. It is also possible that a test instance, which is very different from the training sample, may belong to a completely different class that the SVM classifier was not designed to identify. In such a case, the SVM is unable to detect the fact that the given instance is far from anything used during the training and will always classify the test sample to one of classes that it is trained for. Even though, different methods have been proposed to calibrate posterior probabilities, the selection of a confidence threshold is still a difficult task. Moreover, in various real-world classification applications, there is significant amount of uncertainty due to noise, insufficient data, and specific training and testing protocols. In many applications, like clinical diagnostic, it is beneficial to explicitly recognize this uncertainty instead of trusting the classifier output and ignoring the problem altogether. The applications that benefit from identifying uncertainty are the applications where the cost of rejecting a sample due to uncertainty is less than the cost of an incorrect classification.

In this thesis, we aim to address these limitations by proposing a method to drastically improve the performance and quality of the posterior probabilities for ensemble classifiers (see Chapter 3) and a method to automatically detect an uncertainty region around the decision hyperplane of the SVM (see Chapter 4).

## Chapter 3      Posterior probabilities for classifier ensembles

It has been shown that the performance of any classifier can be improved by using an ensemble of the same classifiers [85, 204, 66, 120, 19, 21, 116, 118, 16, 146, 92, 72, 157, 117]. Ensemble systems improve the generalization of single classifiers by aggregating the prediction of a set of base classifier. Assessing classification reliability (posterior probability) is crucial when identifying the cancer stage and therefore the appropriate treatment. In this scenario, the cost of a misclassified sample is unacceptable high (i.e., a patient will undergo invasive procedures or chemotherapy when it is not needed). Existing methods are limited to either calibrate the posterior probability on an aggregated decision value or obtain a posterior probability for each base classifier and aggregate the result which is sub-optimal both from quality and time complexity. In this chapter, we propose a method that takes advantage of the distribution of the decision values from the base classifiers to summarize a statistic which is subsequently used to generate the posterior probability. Three approaches are considered to fit the probabilistic output to the statistic: the standard *Gaussian CDF*, *isotonic regression* and *linear logistic*. Even though the results presented here use a bagged SVM ensemble (*Z*-bag), our approach is not limited by the aggregation method selected, the choice of base classifiers, nor the statistic used. These methods were proposed in [206].

### 3.1 Posterior probabilities based on *z*-score

To construct an SVM ensemble with posterior probability output we can simply apply the existing naive Bayes [182, 118] approach to produce probabilistic output from a set of class labels. Another natural attempt is to first build a collection of probabilistic single SVMs using the existing methods, and then aggregate them to a final posterior probability, similar to

distribution summation [33, 3]. To illustrate this we considered the model proposed by Platt and an isotonic regression model to estimate the posterior probability for each base classifier in the bagging ensemble and then aggregate these probabilities using average. We define these models as *average Platt* and *average Isotonic*, respectively. This is, however, computationally expensive and redundant especially in the case of average Platt, since it requires parameter fitting for each base classifier. In addition, fitting a sigmoid to a bootstrapped sample may in theory pose a problem, since it is very likely the class conditional decision value distribution is abnormal due to duplicate instances in the bootstrap sampling with replacement. In contrast, what we propose here is a two-stage posterior probabilistic bagging SVM ensemble system. The first stage is to aggregate the distribution of decision values from base SVM classifiers to a summary statistic ( $z$  score), while the second stage is to model the posterior probability given the statistic instead of the original input vector. As opposed to the conventional approach where either the decision values are averaged heuristically before modeling or each base classifier is modeled independently before averaging, we are proposing a general approach from a distributional perspective (i.e., exploring the distribution of the decision values to extract useful information for posterior probability modeling). We exemplify this approach with an SVM bagging ensemble by studying and exploring the empirical distribution of decision values for subsequent posterior calibration.

As described in Section 2.3, an ensemble classifier is a collection of base classifiers. For each base classifier  $k$  we consider the signed distance from the decision hyperplane:

$$F_k = F_k(\mathbf{x}) = \frac{f_k^*(\mathbf{x})}{\|\mathbf{w}_k^*\|} \quad (3.1)$$

where  $f_k^*(\mathbf{x})$  is the decision value (Eq. 2.14) of the  $k$ -th base classifier and  $\mathbf{w}_k^*$  is the coefficient vector. Further we define a  $z$  score of the decision boundary (i.e.,  $f^*(\mathbf{x}) = 0$ ) with respect

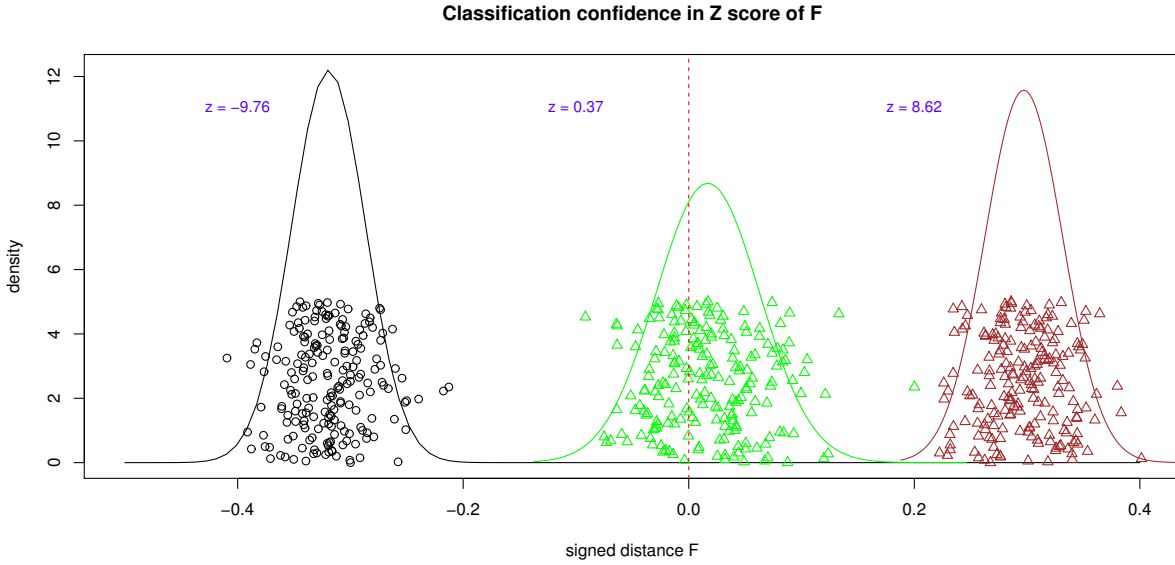


Figure 3.1:  **$z$  score as confidence in the sign of  $F$** : Distribution of decision values from three different samples from the wdbc data set. The curves are estimated using the mean and standard deviation for the signed  $F$  distances. The red line represents the decision boundary and the  $z$  score represents how likely is for a sample to be on one side or the other of the decision boundary. For example, both green sample and brown sample belong to the same class, but the confidence of the ensemble predictions are very different. All ensemble base classifiers agree on the class membership of the brown sample while a lot of them disagree on the green sample. This different level of agreement is reflected in the  $z$  score. *Source*: [206].

to the distribution of the signed distances from the collection of base classifiers:

$$z = z(F) = \frac{0 - \mu_F}{\sigma_F} = -\frac{\mu_F}{\sigma_F} \quad (3.2)$$

where  $\mu_F$  and  $\sigma_F$  are the mean and standard deviation of the signed distance for the same sample over all the individual classifiers. This score represents how confident the sign of  $F$  is (see Fig. 3.1). Note that we use  $F$  instead of  $f^*$  to calculate the  $z$  score, since a slight change in the position of the decision boundary might result in a large leap in the decision value  $f^*$ , which will be over-weighted in computing  $z$  score. Since  $z$  score reflects the confidence in the sign of  $F$ , thereby in the class label  $Y$  of a sample, we based our posterior probabilistic ensemble on the approximation:  $p(Y = 1|X) \simeq p(Y = 1|z)$ . We consider three different approaches to estimate this probability: *Gaussian fitting*, *logistic fitting* and *Isotonic regression*.

### 3.1.1 Standard Gaussian CDF fitting

Our first approach directly estimates the posterior probability based on a voting schema. The higher the number of votes a sample receives from each base classifier, the higher the probability of that prediction. Therefore, our posterior probability estimate for a class given the  $z$  score (i.e.,  $p(Y = 1|z)$ ) can be approximated by the voting percentage given the  $z$  score (i.e.,  $v(Y = 1|z)$ ). Further, we assume that the set of  $F$  values for each instance derived from the base classifiers follows a Gaussian distribution. This assumption is justified by the empirical distributions obtained from multiple data sets (see Fig. 3.2). In addition, a significant number of samples from various data sets from the UCI machine learning repository passed the Shapiro test for normality [162]. Using a Bonferroni corrected Shapiro p-value, 80% of the wdbc test data, 78% of the australian test data and 90% the of spam test data were not significantly different from the normal distribution at 5% significance. Under this Gaussian approximation, the voting percentage can be calculated as the cumulative distribution function (CDF) of the standard Gaussian distribution:

$$p(Y = 1|z) \simeq v(Y = 1|z) \simeq \frac{1}{\sqrt{2\pi}} \int_{-\infty}^z e^{-x^2/2} dx, \quad (3.3)$$

In fact, direct plotting of  $v(y = 1|z)$  shows a very good fitting to the standard Gaussian CDF. In Fig. 3.3, the distribution of the voting percentages represented by the black points follows closely the Gaussian CDF probability estimation represented by the green curve.

### 3.1.2 Logistic fitting

As the second approach, we propose an extension of Platt's [145] method similar to boosting with Platt scaling [134] for bagging ensembles. The main difference between the two methods is the base for the posterior probability modeling. For boosting, the model is based on the weighted aggregated decision value, while for our bagging ensemble the model is based on the summary statistic of the decision values ( $z$  score). The sigmoidal shape of



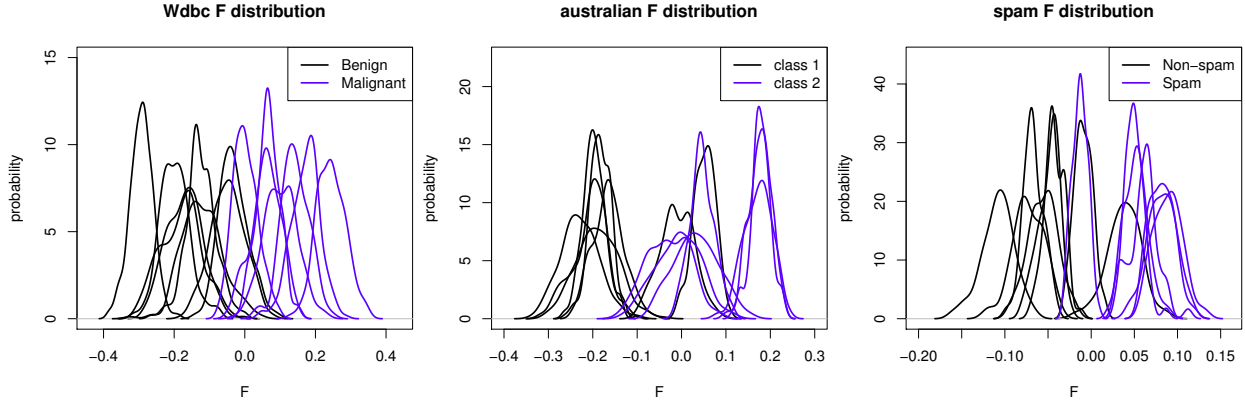


Figure 3.2: **Distribution of  $F$  values for individual samples:** Each curve represents the  $F$  value distribution for an individual instance from the testing data from one of the 3 data sets used here: *wdbc*, *australian* and *spam*. Only a small portion of instances for each data set are shown here. Most of the curves would be well approximated by a Gaussian: using a Bonferroni corrected Shapiro p-value, 80% of the *wdbc* test data, 78% of the *australian* test data and 90% of the *spam* test data were not significantly different from the normal distribution at 5% significance. *Source:* [206].

the estimated posterior probability given the  $z$  score (see the blue curve in Fig. 3.3) points to a similar posterior probability model:

$$p(Y = 1|z) = \frac{1}{1 + e^{Az+B}} \quad (3.4)$$

The parameters  $A$  and  $B$  are determined using the maximum likelihood estimation (Eq. 2.17).

### 3.1.3 Isotonic regression

As the third method for obtaining the posterior probabilities, we apply the isotonic regression [154, 210, 211, 134] on the  $z$  scores. The isotonic regression assumes that:

$$Y^{(i)} = c(z_i) + \epsilon \quad (3.5)$$

where  $c$  is an isotonic (monotonically increasing) function. The solution is going to be a piecewise constant monotonically increasing function:

$$\hat{c} = \operatorname{argmin}_c \sum (Y^{(i)} - c(z_i))^2 \quad (3.6)$$

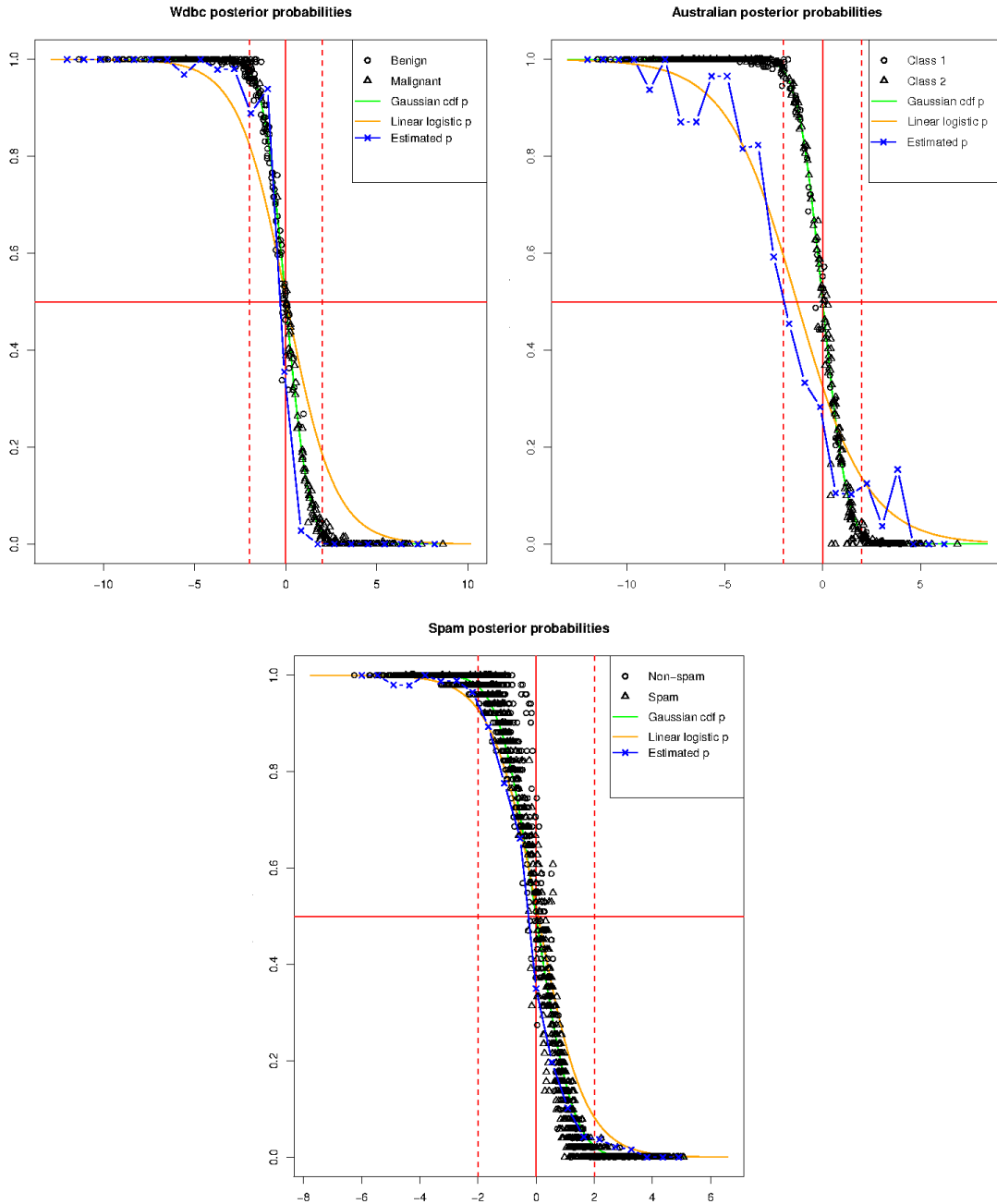


Figure 3.3: **Comparison of fitted posterior probability models:** Two fitted posterior probability models (*Gaussian CDF Z-bag* in green and *logistic linear Z-bag* in orange) are compared with the real estimated posterior probability (blue) and the voting percentage (represented as points in black). The comparison is shown with regards to the  $z$  score (horizontal axis) on three different data sets (wdbc, australian and spam). First, note that the distribution of the voting percentages (black points) follows quite accurately the Gaussian CDF model (green). Second, in the case of australian data set, the Gaussian CDF model (green) overestimates the posterior probabilities in regards to the real estimated probabilities (blue). The real estimated posterior probability is computed using the real class labels through binning over the  $z$  score. *Source:* [206].

This function maps the summarized bagging statistic to calibrated posterior probabilities. In comparison with the previous two methods, this approach is likely to yield 0 or 1 posterior probabilities, which in fact are not well calibrated. Although additional corrections can be applied to avoid such extreme values (e.g., round them to the closest non-extreme value), we considered the raw output in our comparison.

## 3.2 Experiment design

To evaluate the SVM ensembles, we split each data set into two class-balanced sub-sets, one for fitting the posterior probability model (training data) and one for evaluating the model (testing data). The proposed testing procedure is described in Fig. 3.4. The non-probabilistic models (*single SVM* and *majority voting*) are build directly on the training data without the need for any other parameter tuning. To fit the proposed posterior probability models we use 10-fold cross validation in order to compute an unbiased set of signed distances for each sample from the training data. Based on this set we estimate the parameters of the proposed probabilistic models. The performance of each model is evaluated on the testing data using: i) the number of prediction errors, ii) the negative logarithm likelihood ( $-\log LH$ ) and by iii) direct comparison of the posterior probability output vectors from each model. We used Friedman test followed by Nemenyi test for comparisons of multiple models on multiple data sets, both for the model error rates and likelihoods, as recommended in [41]. We also applied the binomial sign test for a pairwise comparison [164].

We evaluated our *Z-bag* approach on the ringnorm artificial data [20] and twelve data sets from the UCI Machine Learning repository [7]: the MAGIC Gamma Telescope Data Set (*magic04*), the *ionosphere* data set, the *thyroid* disease data set, the Contraceptive Method Choice Data Set (*cmc*), the Letter Recognition Data Set (*letters*), the Wisconsin diagnostic breast cancer data set (*wdbc*), Australian credit data set (*australian*), the *spam* email data set, the *adult* income census data set, the *pima* Indians diabetes data set, the Wisconsin

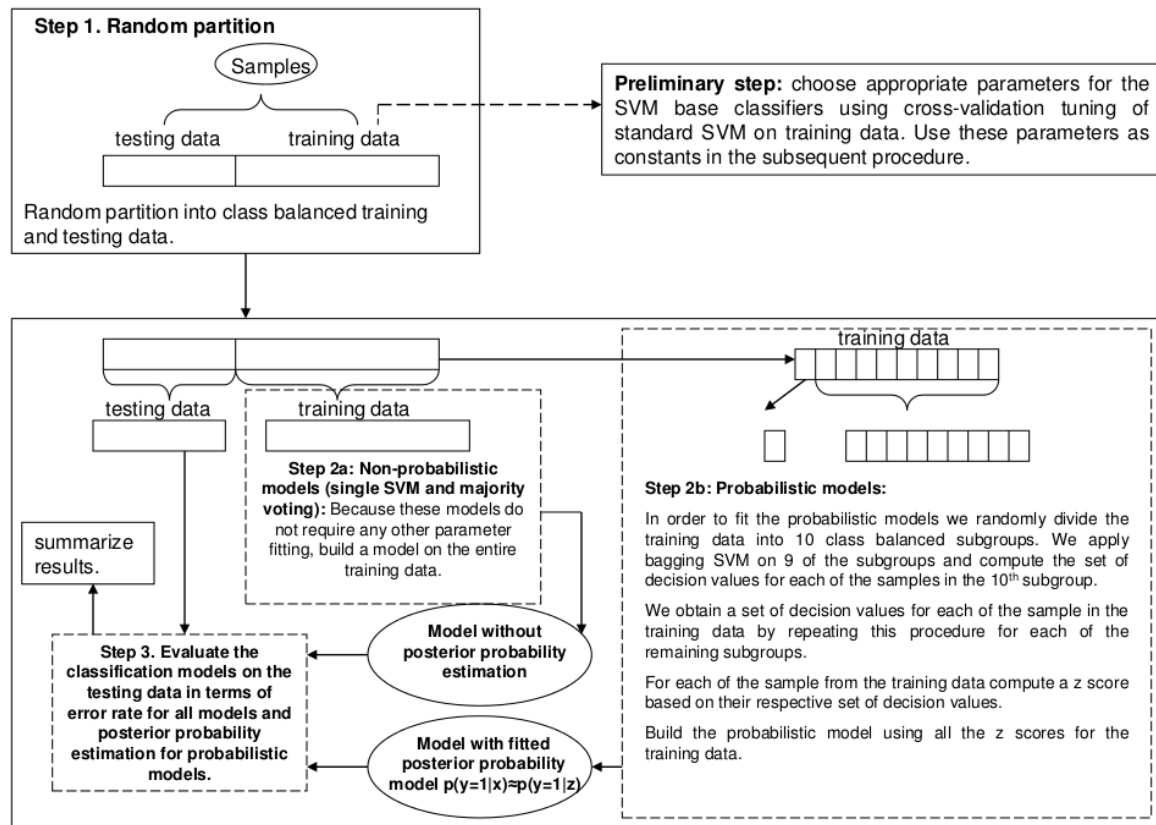


Figure 3.4: Procedure to assess the performance of the proposed bagging SVM ensemble Source: [206].

breast cancer data set (*wbc*), and the German credit data set (*german*). For the ringnorm artificial data, we set the number of features to 5, and 2000 random samples for testing on models with varying training size from 50 to 300. We converted the *letters* data set into a binary classification problem by treating A-M as one class and N-Z as another. For the *thyroid* data, we group class 1 and 2 as one class of size 284 and randomly sampled 284 samples from class 3 for training. And we randomly sampled 500 samples from each classes as the total data for analysis for *magic04*. For all data sets, each feature is scaled to zero mean and unit variance before evaluation. This allows to set the SVM kernel parameters  $r$  and  $coef\gamma$  to  $1/p$ , as required by LIBSVM [27]. Table 3.1 shows the choice of training parameters for each the data sets. On each data set the proposed SVM ensembles, *Gaussian CDF*, *logistic linear* and *Z isotonic*, are compared with the *single SVM*, *majority voting*, *naive Bayes*, *average Platt* and *average Isotonic* models.

### 3.3 Performance on artificial data

We use the artificial data to asses the performance of these models to estimate the real posterior probabilities. For each probabilistic model we map (Fig. 3.5 and Fig. 3.6 for training size of 50 and 300, respectively) the estimated posterior probabilities to the real ones. An ideal estimation of the posterior probability will display a perfect correlation of estimated and real probabilities (i.e., the points in Fig. 3.5 will follow the red diagonal). Independent of the training sample size, the *naive Bayes* model not only achieves poor estimation of posterior probabilities (the samples are far away from the diagonal), but also displays extreme outputs very close to 0 or 1. Similarly, the *Gaussian CDF* model may not be appropriate for this type of data, since the low accuracy of bagging on this data may lead to a poor approximation in Eq. 3.3. The *Z-score isotonic* model displays its piecewise constant nature and therefore obtains a poor correlation to the real probabilities. Taking the mean of the bagged isotonic probabilities, the *average Isotonic* does not display the same

Data set	Training size	Testing size	Ensemble size	type	Kernel parameters			
					C	r	degree	coeff0
wdbc	56	513	201	L	10			
				P	10	1/30	2	1/30
				R	10	1/30		
australian	137	553	201	L	1			
				P	1	1/14	3	1/14
				R	1	1/14		
spam	919	3682	51	L	10			
				P	10	1/57	2	1/57
				R	10	1/57		
adult	1605	15060	101	L	5			
				P	5	1/14	2	1/14
				R	5	1/14		
pima	76	692	501	L	1			
				P	1	1/8	2	1/8
				R	1	1/8		
wbc	67	616	501	L	1			
				P	1	1/10	2	1/10
				R	1	1/10		
german	100	900	501	L	1			
				P	1	1/24	2	1/24
				R	1	1/24		
magic04	100	900	51	L	5			
				P	5	1/10	2	1/10
				R	5	1/10		
ionosphere	70	281	101	L	4			
				P	4	1/33	2	1/33
				R	4	1/33		
thyroid	568	3428	51	L	0.00001			
				P	0.00001	1/21	2	1/21
				R	0.00001	1/21		
cmc	287	675	101	L	1			
				P	1	1/9	2	1/9
				R	1	1/9		
letters	199	19801	51	L	10			
				P	10	1/16	3	1/16
				R	10	1/16		

Table 3.1: **SVM model parameters and data sets summary:** linear (L), polynomial (P), and radial basis (R) represent the SVM kernels. The cost  $C$  and *degree* are tuned through cross validation, while  $r$  and *coeff0* are set to  $1/p$ , where  $p$  is the number of features of the data set.

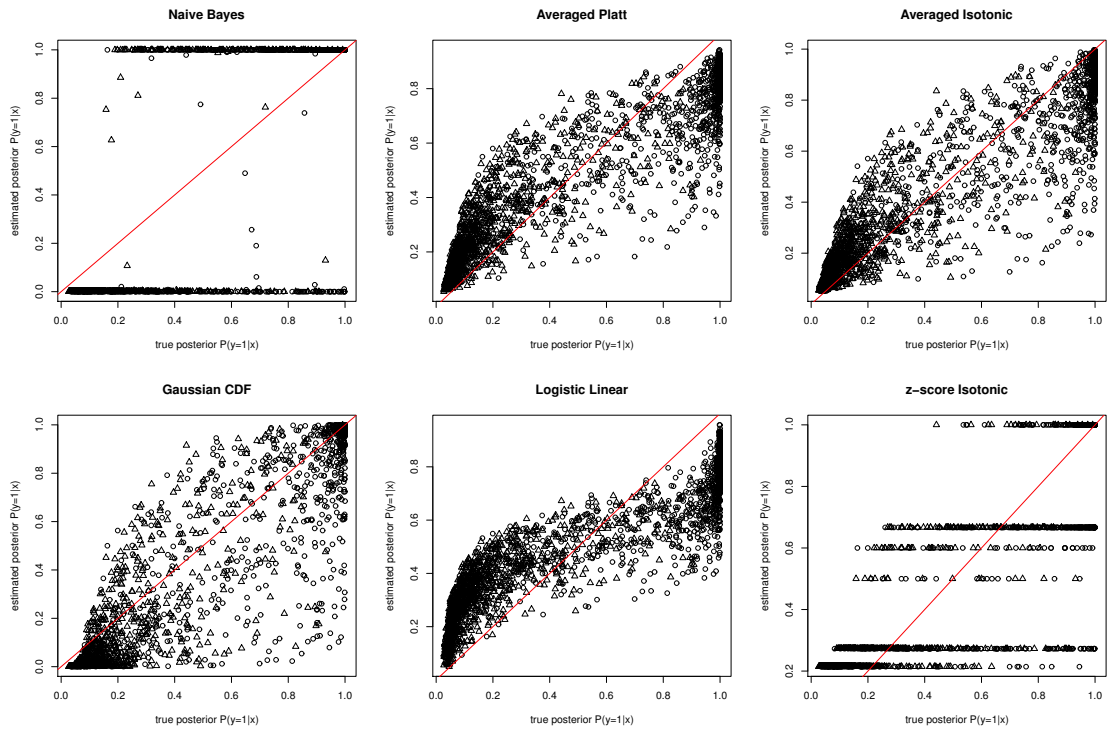


Figure 3.5: **Comparison of estimated posterior with true posterior on ringnorm data with sample size of 50.** A perfect estimation will follow the red diagonal. The *Naive Bayes* outputs extreme probabilities and achieve very poor correlation to the true probabilities. The *isotonic* model exhibits its piece-wise nature, while the average isotonic by taking the average over the bagged probabilities does not display the same nature and achieves a better correlation. This model, together with *average Platt* and *logistic linear*, tend to underestimate the posterior probability at extreme values. For hard to classify samples, with true posterior of 0.5, the *logistic linear* model performs best. *Source:* [206].

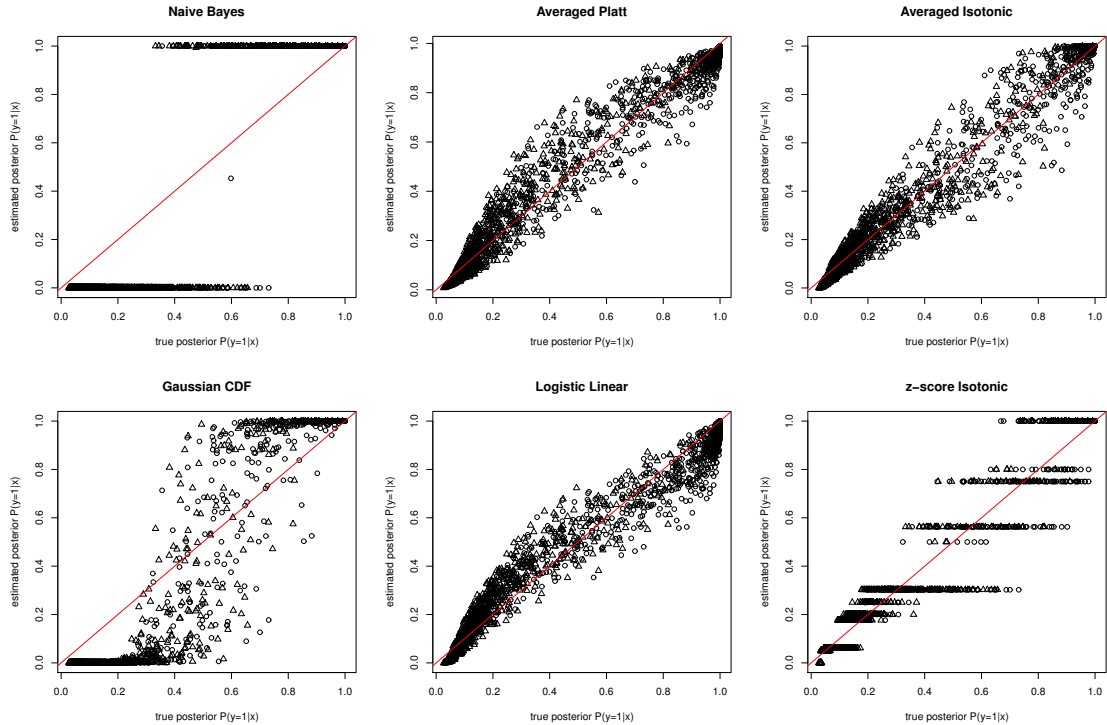


Figure 3.6: **Comparison of estimated posterior with true posterior on ringnorm data with sample size of 300.** Similar to the comparison with fewer samples, the *Naive Bayes* achieve very poor correlation to the true probabilities by outputting extreme probabilities. The *isotonic* model is of piece-wise nature. With increased sample size, the other three models perform the best again with an even better correlation to the true probabilities. *Source:* [206].

nature and achieves better correlation. This model, together with *average Platt* and *logistic linear*, tend to underestimate the posterior probability at extreme values which is due to the nature of bagging that requires most of the base classifiers to agree for these predictions. The variance introduced by bagging reduces the chances for such a unanimous decision. In contrast, when considering samples that are hard to classify (true posterior probability around 0.5) the *logistic linear* performs better than the other two methods. The distribution of the samples closer to the diagonal supports this observation (Fig. 3.5). All these three models perform better when the sample size is increased to 300 samples (Fig.3.6) exhibiting the same trend of underestimation of probabilities at extremes and a better performance for the *logistic linear* model for samples with a true probability around 0.5. In addition both *average Platt* and *logistic linear* tend to yield lower estimates in comparison to the *average Isotonic* model for samples with extreme posterior probabilities.



Besides the direct comparison against the true posterior probability we recorded the mean square error (MSE) as a function of the training sample sizes (left side of Fig. 3.7). *Naive Bayes* performs consistently worse than the other models, since it not only bears unrealistic assumption on the independence of base classifiers but also needs a large training data set to estimate a reliable prior class probabilities and class conditional probabilities. In contrast, the *average Isotonic* model is consistently ranked the best against all the other models which goes against the previous observation that the *logistic linear* model performs better for samples prone to misclassification (see the scatter plot comparison in previous paragraph). To further investigate this phenomenon we consider a subset of the testing data that has a true posterior probability between 0.2 and 0.8 (right side of Fig. 3.7). On this subset we note that *logistic linear* model is indeed ranked as the best model and therefore is confirms the observation that it performs better on samples that are harder to classify. The *average isotonic* model performs better than the *logistic linear* due to the nature of the ringnorm data set used as base for this experiment which tends to have samples with extreme posterior probabilities (out of 2000 samples, 54% have a posterior above 0.9 or below 0.1, and 74% above 0.8 or below 0.2). Furthermore, such comparison is also dependent on the measure used for comparison. We use MSE because it can be applied to any model which is not true for Kullback-Leibler divergence which is not suitable for models that produce 0 or 1 probabilities. For example, with a training sample size of 50, the *logistic linear* model achieves the best Kullback-Leibler divergence measure (0.46) in comparison to *averaged Platt* (0.66) and *average Isotonic* (0.49).

### 3.4 Accuracy of the predictions on benchmark data sets from the UCI machine learning repository

In terms of prediction accuracy, we used the error rate to compare the performance of *Z-bag* with the existing methods. For each data set and method combination we compute

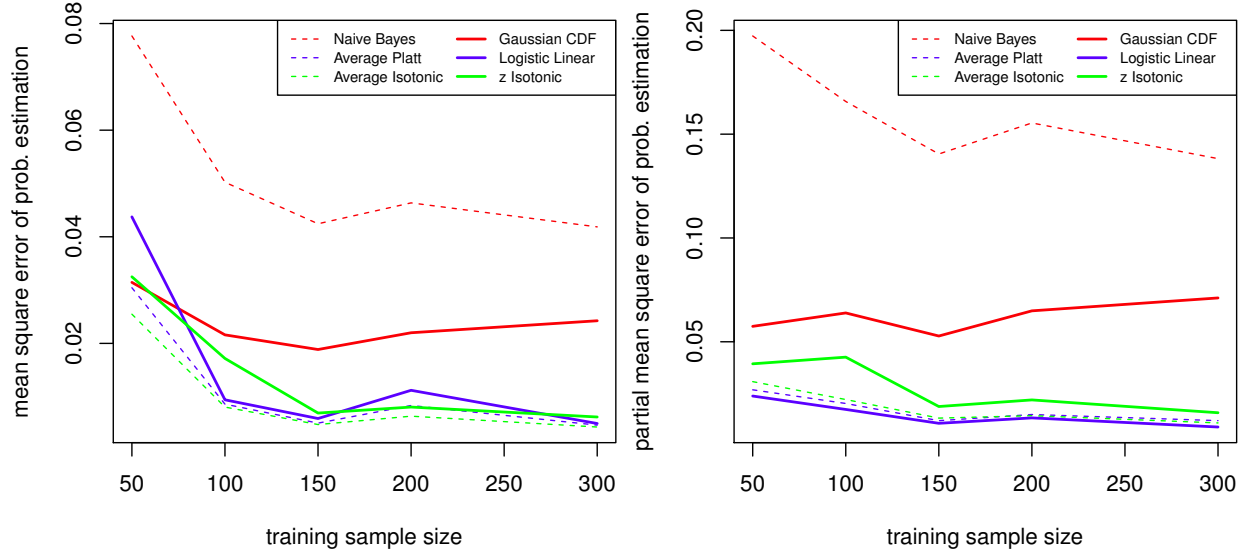


Figure 3.7: **Performance on ringnorm artificial data:** Left panel gives mean squared error (MSE) for each models; the right panel shows MSE when testing samples are restricted to having a true posterior within 0.2 to 0.8. *Source:* [206].

the error rate for three different kernel types (Table 3.2) and select the kernel with best performance as representative for the subsequent model comparison. Evidently, the model that yields the smallest number of errors is the better model. However, to quantify the significance of *Z-bag*'s performance increase we ranked the performance of the models on each data set and computed the average rank over all data sets for each model. This averaged rank gives an ordering of all models in terms of prediction accuracy. To further assess the statistical significance of differences between the models, we used Friedman test for multiple classifier comparison, followed by the post-hoc Nemenyi test on all pairwise classifiers, as recommended in [41].

*Logistic linear* ranks as the best model overall (average rank 2.5) followed by *Gaussian CDF* (3.67) and *average Isotonic* (3.88). The Friedman test over all the models suggest a significant difference among them ( $p = 7.5e^{-4}$ ). To locate the difference we used the Nemenyi test which in our case (number of data sets and models) has a critical difference among average ranks of 3.03 at 5% significance. Using this critical threshold we note that the *single SVM* is significant different than *averaged Platt*, *averaged Isotonic* and *logistic linear*. This result is expected as classifier ensembles are expected to generally perform better than single

classifiers. With a less stringent threshold of 10% the difference between *naive Bayes* and *logistic linear* becomes significant. Even though, there is no significant difference between the proposed models and the existing ones, a simple win-tie-lose quantifies the performance increase. The *logistic model* achieves a score of 9-1-2 against *naive Bayes*, 10-0-2 against *averaged Platt* and 8-1-3 against *averaged Isotonic*. The fact that *logistic linear* ranks the best is in accordance with our expectation from the artificial data. In Section 3.3 we showed that *logistic linear* works best on samples with true posterior probability around 0.5 which is generally encountered in real world data sets.

### 3.5 Quality of the posterior probability estimates on benchmark data sets from the UCI machine learning repository

The quality of the posterior probability estimates was assessed using the negative log likelihood for the test data and pair-wise comparison of the probability of predicting the correct class. The probability of predicting the correct class, based on a given posterior probability, is equal to the posterior probability for a correctly classified sample and one minus the posterior probability for a misclassified sample.

A common measure used to evaluate the performance of a classifier that provides posterior probability estimates is the negative log likelihood of predicting the correct class over the entire test data set. This is calculated as:

$$-\log LH = -\log \prod_{i=1}^n P(y = Y^{(i)} | \mathbf{x}^{(i)}) = -\sum_{i=1}^n \log P(y = Y^{(i)} | \mathbf{x}^{(i)}) \quad (3.7)$$

Since the product above is a product of probabilities, all the factors are less than or equal to 1. This means the product will be less than 1 and the negative log of the entire product will be non-negative. Ideally all the input vectors will be assigned to their correct class with

Data	K	single SVM	Ensemble						
			majority voting	Probabilistic					
				Existing			Z-bag		
				Naive Bayes	averaged Platt	averaged Iso- tonic	Gaussian CDF	logistic linear	Z-score iso- tonic
wdbc	L	40 <sup>8</sup>	39	36	37	37	35	33	33 <sup>5</sup>
	P	66	69	74	82	66	73	82	84
	R	42	27 <sup>2.5</sup>	33 <sup>5</sup>	33 <sup>5</sup>	36 <sup>7</sup>	23 <sup>1</sup>	27 <sup>2.5</sup>	35
australian	L	80	70 <sup>1</sup>	74 <sup>5.5</sup>	72 <sup>2.5</sup>	72 <sup>2.5</sup>	73 <sup>4</sup>	74 <sup>5.5</sup>	75 <sup>7.5</sup>
	P	113	105	95	87	85	104	88	92
	R	75 <sup>7.5</sup>	79	76	77	79	80	80	80
spam	L	320	292	291	301	287	288	292	330
	P	348	328	323	324	289	323	327	324
	R	293 <sup>8</sup>	285 <sup>6</sup>	278 <sup>4</sup>	271 <sup>1</sup>	273 <sup>2</sup>	286 <sup>7</sup>	274 <sup>3</sup>	283 <sup>5</sup>
adult	L	2401 <sup>7</sup>	2384	2566	2400	2382	2381	2370	2380
	P	2414	2339 <sup>3</sup>	2464 <sup>8</sup>	2384	2360	2335	2327 <sup>1.5</sup>	2344 <sup>4</sup>
	R	2427	2340	2474	2363 <sup>6</sup>	2357 <sup>5</sup>	2327 <sup>1.5</sup>	2344	2405
pima	L	204 <sup>8</sup>	198 <sup>7</sup>	198	197 <sup>5.5</sup>	195 <sup>4</sup>	193 <sup>2.5</sup>	197	193 <sup>2.5</sup>
	P	216	207	197 <sup>5.5</sup>	203	197	204	190 <sup>1</sup>	218
	R	217	209	203	209	208	207	206	200
wbc	L	27	20	19 <sup>3.5</sup>	22	19 <sup>3.5</sup>	19 <sup>3.5</sup>	19 <sup>3.5</sup>	18 <sup>1</sup>
	P	32	30	27	28	19	28	24	25
	R	20 <sup>7</sup>	20 <sup>7</sup>	20	20 <sup>7</sup>	21	20	21	24
german	L	278	239 <sup>3</sup>	259	244 <sup>5</sup>	238 <sup>2</sup>	239	242 <sup>4</sup>	260
	P	268	241	258 <sup>7</sup>	255	242	230 <sup>1</sup>	268	248 <sup>6</sup>
	R	259 <sup>8</sup>	270	266	268	269	270	270	270
magic04	L	232	229	231	235	234	240	230	246
	P	231	235	224	221 <sup>4</sup>	219 <sup>3</sup>	235	216 <sup>1</sup>	218 <sup>2</sup>
	R	229 <sup>8</sup>	223 <sup>6</sup>	223 <sup>6</sup>	225	224	223 <sup>6</sup>	228	221
ionosphere	L	54	41	30	41	38	46	27	32
	P	54	61	53	45	25 <sup>2</sup>	61	28	30
	R	29 <sup>5</sup>	31 <sup>8</sup>	30 <sup>6.5</sup>	28 <sup>4</sup>	29	30 <sup>6.5</sup>	18 <sup>1</sup>	27 <sup>3</sup>
thyroid	L	131 <sup>6.5</sup>	131 <sup>6.5</sup>	124 <sup>4</sup>	111 <sup>2</sup>	113 <sup>3</sup>	129 <sup>5</sup>	94 <sup>1</sup>	135 <sup>8</sup>
	P	194	163	173	181	209	167	181	235
	R	191	155	144	115	125	147	102	145
cmc	L	204	202	207	201	202	198	198	203
	P	194	185	204	186	188 <sup>6.5</sup>	183	185	194
	R	185 <sup>4</sup>	180 <sup>1</sup>	188 <sup>6.5</sup>	186 <sup>5</sup>	189	182 <sup>3</sup>	181 <sup>2</sup>	191 <sup>8</sup>
letters	L	6180	5938	5951	6054	6028	6025	6039	6229
	P	4737	4637	4624	4694	4711	4609	4630	4653
	R	4347 <sup>7</sup>	4204 <sup>2</sup>	4194 <sup>1</sup>	4314 <sup>5</sup>	4332 <sup>6</sup>	4280 <sup>3</sup>	4295 <sup>4</sup>	4572 <sup>8</sup>
<b>Average rank</b>		<b>7.000</b>	<b>4.417</b>	<b>5.208</b>	<b>4.333</b>	<b>3.875</b>	<b>3.667</b>	<b>2.500</b>	<b>5.000</b>

Table 3.2: **Error rate comparison:** We record the number of errors (lower is better) on of the twelve UCI data sets for each kernel type: linear (L), polynomial (P), and radial basis (R). The models shaded in gray, are the best model for each data set and method combination. This is the model used subsequently to compare the methods. For each data set the models are ranked based on the number of errors (the supper script marks the rank). In case of ties, all tied methods receive the same rank obtained by averaging their respective order in the ranking. Both *Gaussian CDF Z-bag* and *logistic linear Z-bag* achieve a better average rank over all data sets compared to any of the existing methods.

a posterior probability close to 1, which will yield a product close to 1 and a  $\log$  close to 0. Therefore, the lower the negative log likelihood the better the classifier. The results based on the likelihood from the six probabilistic models are summarized in the Table 3.3. As mentioned in Section 3.1, models employing isotonic regression are likely to have probability outputs of exactly 0, that yields an undefined logarithm (these vales are marked ND in Table 3.3). In addition to *isotonic regression*, *naive Bayes* and occasionally *Gaussian CDF* can produce such extreme probabilities. Note that such cases of undefined logarithm do not imply a weaker performance.

Our first comparison includes the data sets and models that have a defined logarithm likelihood for all three kernels used. Four models (*average platt*, *average Isotonic*, *Gaussian CDF* and *logistic linear*) out of six models on seven out of twelve data sets satisfy this condition. We followed a similar approach to the error rate comparison. The *average Isotonic* model ranks as the best model, followed by *logistic linear* (the win-tie-lose is 4-0-3). The overall Friedman p-value is 0.011, and the critical threshold of Nemenyi test is 1.77 (5%), suggesting a significant difference between *average Isotonic* and *Gaussian CDF*. In fact the *Gaussian CDF* model has the worst ranking among the four since it requires stringent assumptions that need to be checked for each specific data set. Therefore, *Gaussian CDF* is not a general purpose model in terms of estimating posterior probability. The good performance of the *averaged Isotonic* model on the total likelihood is consistent with our analysis of the total MSE on artificial data. If we exclude the non-general purpose *Gaussian CDF* model, the Friedman test shows no significant difference among the remaining three models ( $p = 0.16$ ). To take advantage of all data sets we compare *averaged Platt* and *logistic linear* on all data sets. These two models were selected because they do not produce any undefined likelihood measure. To avoid assuming commensurability of log likelihood scores on different data sets, we apply a nonparametric binomial sign test for the comparison. The *logistic linear* model outperforms *average Platt* on 10 out of 12 data sets, representing a significance p-value of 0.0386.

Although *naive Bayes* and *z-score isotonic* have undefined likelihood measure for most of the data sets, the *naive Bayes* defines it for the ionosphere data set, and *z-score isotonic* for the german data set (see Table 3.3). The poor likelihood measure of *naive Bayes* on ionosphere data is consistent with our observation on the artificial data that it tends to predict extreme posterior probability even for misclassified samples. The likelihood of *z-score isotonic* on german data set is on par with the other models and we expect it to improve with larger training data sets (as observed from artificial data).

In order to further investigate the relationship between these models on estimating posterior probabilities, we performed a direct comparison of the probability to predict the correct class (shown in Fig. 3.8). This figure contains a direct comparison of *logistic linear* against *Gaussian CDF*, *average Platt* and *average Isotonic* respectively. The left panel shows that the posterior probabilities yielded by the *Gaussian CDF Z-bag* method are consistently higher than the ones provided by *logistic linear*. The middle panel reveals good agreement between *logistic linear* and *average Platt*. All three comparisons indicate that the *logistic linear* model gives better posterior estimation for samples prone to misclassification (e.g. samples in the lower-left quadrant with low posterior of correct class for both models) which agrees with our evidence from artificial data.

Although the experiments indicate that the *Gaussian CDF* model would generate higher posterior probabilities than any of the other methods, the theory does not support it. One can imagine a case where parameter  $B$  is set to 0 and parameter  $A$  is smaller than  $-\sqrt{8/\pi}$  in Eq. 3.4. This yields the *logistic linear* derivative in the origin larger the *Gaussian CDF* one, which means that the former will assign higher posterior probabilities than the latter. Same argument stands in the comparison of *Gaussian CDF* against *average Platt*, as the latter is essentially a summation over a set of logistic linear models.

To summarize, *logistic linear* model performs very well in terms of both error rate (ranked best) and likelihood (ranked second best). It significantly outperforms *average Platt* model on likelihood measure. The *Gaussian CDF* model is generally applicable for classification

Data	K	Ensemble					
		Existing			Z-bag		
		<i>Naive Bayes</i>	<i>averaged Platt</i>	<i>averaged Isotonic</i>	<i>Gaussian CDF</i>	<i>logistic linear</i>	<i>Z-score isotonic</i>
spam	L	ND	901.32	783.38	865.83 <sup>4</sup>	828.28 <sup>3</sup>	ND
	P	ND	1001.92	833.82	3824.02	1175.03	ND
	R	ND	779.75 <sup>2</sup>	775.42 <sup>1</sup>	1661.41	879.65	ND
adult	L	ND	5300.65	5198.07	5913.53 <sup>4</sup>	5388.71	ND
	P	ND	5295.32 <sup>3</sup>	5186.89 <sup>1</sup>	6859.94	5423.72	ND
	R	ND	5316.44	5251.16	6505.27	5280.66 <sup>2</sup>	ND
pima	L	ND	390.16	386.169	485.85 <sup>4</sup>	381.782	ND
	P	ND	386.78	380.499	645.31	374.32 <sup>1</sup>	ND
	R	ND	380.43 <sup>3</sup>	379.63 <sup>2</sup>	607.23	377.23	ND
german	L	ND	485.39 <sup>4</sup>	473.14 <sup>1</sup>	482.226	483.88 <sup>3</sup>	ND
	P	ND	499.431	479.632	481.67 <sup>2</sup>	508.189	511.17
	R	ND	526.237	525.182	1599.06	551.469	539.92
magic04	L	ND	493.49	491.49	627.81	492.363	ND
	P	ND	464.42	467.148	583.84	459.074	ND
	R	ND	450.10 <sup>3</sup>	448.79 <sup>2</sup>	554.33 <sup>4</sup>	435.00 <sup>1</sup>	ND
cmc	L	ND	379.64	385.684	821.098	382.544	ND
	P	ND	391.448	394.487	763.98 <sup>4</sup>	386.521	ND
	R	ND	371.95 <sup>3</sup>	370.41 <sup>2</sup>	925.052	367.94 <sup>1</sup>	ND
letters	L	ND	11280.3	11303.7	16006.3	11305	ND
	P	ND	9808.55	9657.19	9855.27	9584.38	ND
	R	ND	8917.09 <sup>1</sup>	8962.61 <sup>2</sup>	9791.87 <sup>4</sup>	9015.15 <sup>3</sup>	ND
<b>Average rank</b>			<b>2.714</b>	<b>1.571</b>	<b>3.714</b>	<b>2.000</b>	
wdbc	L	ND	89.46	ND	81.068	84.35	ND
	P	ND	193.10	175.22	190.6	195.75	ND
	R	ND	115.69	98.19	92.61	112.57	ND
australian	L	ND	201.21	192.96	269.9	199.29	ND
	P	ND	217.95	ND	591.77	219.45	ND
	R	ND	200.21	207.71	492.78	203.32	ND
wbc	L	ND	73.0148	52.6096	67.7347	68.3488	ND
	P	ND	92.344	54.042	204.427	71.783	ND
	R	ND	79.253	ND	124.393	84.0036	ND
ionosphere	L	ND	114.31	96.27	104.82	87.28	ND
	P	6181	97.676	70.555	488.55	83.328	ND
	R	5547	68.597	ND	225.804	66.126	ND
thyroid	L	ND	773.897	ND	3273.57	533.688	ND
	P	ND	960.615	ND	2346.45	759.295	ND
	R	ND	889.77	ND	ND	621	ND

Table 3.3: **Posterior probability estimates comparison:** The table contains the results obtained using the negative log likelihood measure. Models marked with gray represent the best model for that respective combination of data set and method. The average rank is computed for the four models in the middle on the top where data sets, where the superscript number gives the particular ranking. "ND" represents "not defined" for the negative log likelihood.

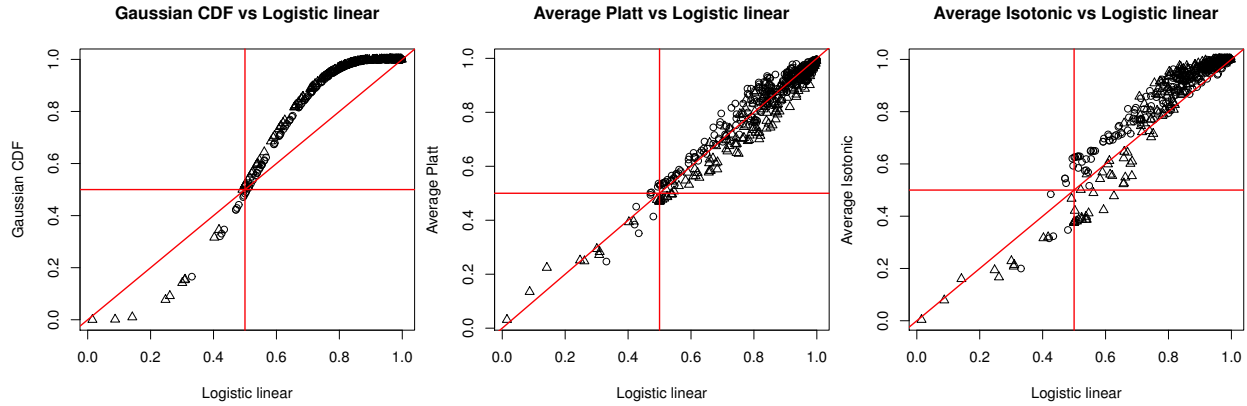


Figure 3.8: **Pair-wise comparison of the probability to predict the correct class:** The probability to predict the correct class is the posterior probability for a correctly classified sample and one minus the posterior probability for a misclassified sample. The upper right quadrant in each panel is where both models make a correct classification, the lower left quadrant is where both misclassify, and they do not agree for the others. The middle panel shows that there is a good agreement between the posterior probabilities given by *averaged Platt* and *logistic linear Z-bag* models. The left panel shows that the posterior probabilities given by the *Gaussian CDF Z-bag* are consistently higher than those generated from *logistic linear Z-bag* (i.e., above the diagonal in the upper right quadrant and below the diagonal in the lower left quadrant). All panels suggest that *logistic linear Z-bag* is better at estimating posterior for samples prone to misclassification. *Source:* [206].

purpose (ranked second best), but in the estimation of posterior it needs stringent assumptions that might not be satisfied by all data sets. The *Z-score Isotonic* model does not perform well in general, but can be improved with a larger sample size. *Average Isotonic* performs good (ranked third best on error rate and best on likelihood), despite its limitation with small size training sets. This maybe attributed to the combining effect of bagging averaging (reducing variance) and isotonic regression (reducing bias toward a specific logistic sigmoid, as opposed to the Platt model). The *average Platt* model achieves similar but worse performance than the *logistic linear* model. The *naive Bayes* model may be used for classification but yields poor posterior probability estimation. Another advantage that these measures do not capture is the reduced computational complexity of the *logistic linear Z-bag* model. That is, in the case of the *average Platt*, one logistic model needs to be fitted to each of the bootstrap samples. If we consider the time complexity of fitting one logistic model to be  $O(\Phi(n))$ , then the overall complexity of the *average Platt* method will be  $O(k * \Phi(n))$  where  $k$  is the number of base classifiers. This process is time consuming and redundant.



The *logistic linear Z-bag* only fits one logistic model on the statistic generated from the outputs of all the base classifiers and hence the complexity of the *logistic linear model* will be  $O(\Phi(n))$ . Therefore the complexity is reduced by a factor equal to the size of the ensemble. In addition, fitting a sigmoid to a bootstrapped sample may in theory be invalid due to duplicate instances. Furthermore, bagging is a variance reduction technique, and in the case of the *averaged Platt* model, the variance reduction happens at the final step of averaging the posterior probabilities obtained from the base classifiers. This is another key distinction from our *logistic linear Z-bag* method where the  $z$  score is used to absorb the variance from the decision values before we fit a probabilistic model. The variance of  $z$  statistic is reduced by the number of bootstrapped samples compared to the one of the decision values.

### 3.6 $z$ -score drawbacks, alternative statistics and limitations

Since we base our posterior probability models on the  $z$  score, it becomes vital to the performance of the models. The way a set of SVM classifiers is constructed influences the  $z$  score value. Factors such as the size of the bootstrap sample and the number of bootstrap samples have a significant influence. In our analysis, we keep the value of these factors constant in all models for comparison purposes. However, we did explore some of these concerns. We analyzed the change of the  $z$  score with regards to the number of bootstrap samples. The *wdbc* test data set was considered as testbed (see Fig. 3.9). The  $z$  score pattern achieves relative stability after the number of bootstrap samples exceeds 50. The correlation coefficient between the  $z$  score patterned obtained with 50 bootstrapped samples and 500 bootstrapped samples is 0.99. Further experiments are needed to assess this and other factors affecting the  $z$  score and thereby the performance of proposed models. In addition,  $z$  score may not be a sufficient summary statistic to capture all the information carried by the decision values of a set of SVM classifiers and the way they were obtained. For instance, one

parameter that could be taken into consideration is the number of bootstrapped samples. A simple way to incorporate this is to replace the  $z$  score with the  $t$  statistic and consider the number of bootstrap samples as the degrees of freedom. This is, however, not necessary in our analysis since the smallest number of bootstrap samples we used was 51, and it is generally accepted that the  $t$ -distribution closely approximates standard normal distribution when the sample size exceeds 30. Nevertheless, incorporating other variables or replacing the statistic is currently under investigation.

Even though we used bagging as example for our framework, the same approach can be applied to a variety of ensemble systems. The main limitation is the ability to identify the distribution that best characterizes the decision values of the base classifiers. In addition, it is also important to select an appropriate summary statistic that is able to extract adequate information for the posterior probability modeling.

### 3.7 Summary

Ensemble classifier methods were proven to achieve better classification accuracy than single classifiers. Estimating confidence measures for the prediction are essential in a number of applications such as biomedical and diagnostics applications. The existing methods for generating posterior probabilities for classification ensembles are designed to either combine the base classifier probability outputs or to calibrate the aggregated decision value. The first category are likely to be computational intensive due to the requirement of fitting a posterior probability model for each base classifier while the second category were developed from the need to recover some of the information lost through the aggregation process. In this chapter, we proposed a method that extracts additional information from the decision values of the base classifiers to fit a single posterior probability model. The method uses bagging to construct an ensemble with SVM as base classifiers and the  $z$  statistic to summarize the decision values of the base classifiers, hence *Z-bag*. To fit the probabilistic outputs to the

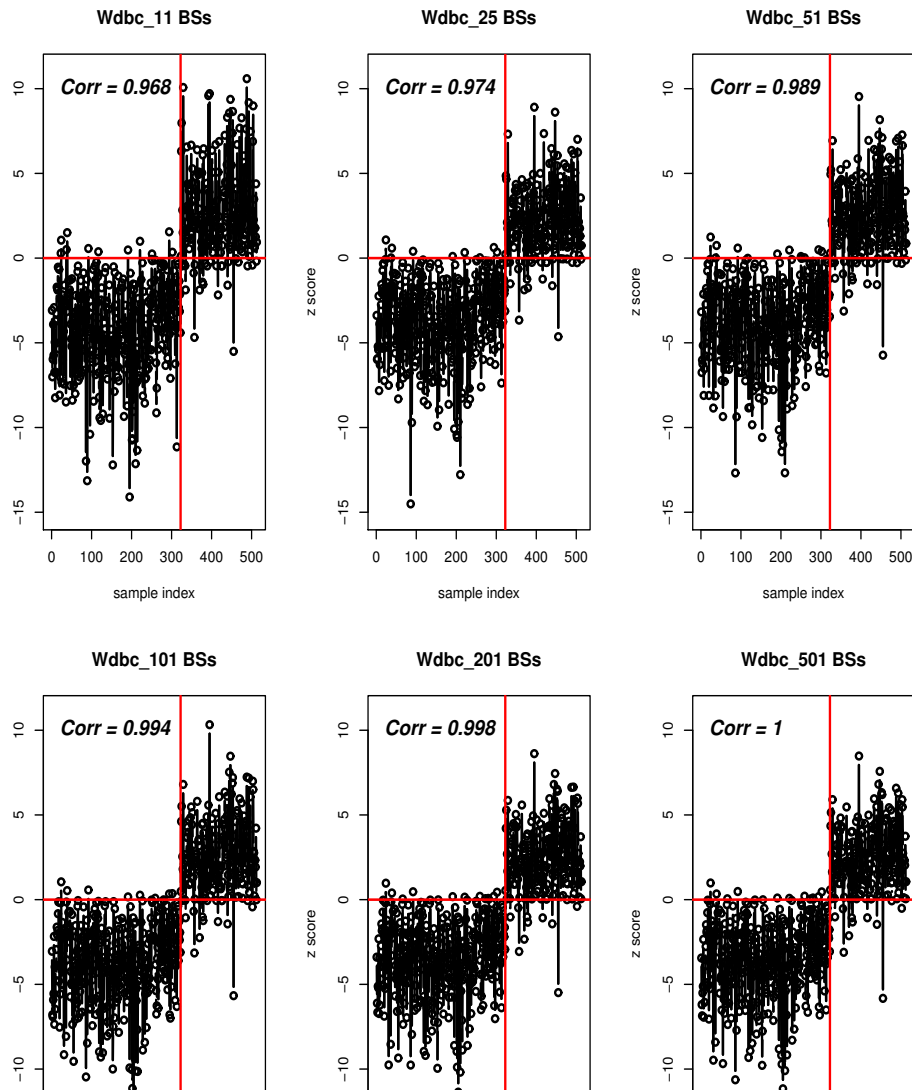


Figure 3.9: **z score pattern stabilization:** We present the variation of the  $z$  score with respect to the number of bootstrap samples ( $BS$ ). Each circle is a specific instance from the *wdbc* testing data and the vertical red line separates the two classes. The Pearson correlation (*corr*) is computed between each of the first 5  $z$  score patterns and the last one. Notice that while there are visual differences between the top three graphics ( $BS < 50$ ), the differences are not present between any of the graphics with  $BS \geq 50$ . Moreover, the correlation coefficient between any model with  $BS \geq 50$  and  $BS = 501$  is 0.99 or better. *Source:* [206].

statistic, we considered three variations: *Gaussian CDF*, *logistic linear* and *Z-score Isotonic*. The *logistic linear* model is generally preferred because of the stable good performance, better performance at estimating posterior probability for samples prone to misclassification, and improved computational cost. Alternatively, the *Gaussian CDF Z-bag* model is easier to be implemented without further parameter training and it ranks as second best in terms of accuracy while estimation of posterior can be appropriate if some conditions met. In conclusion, these approaches achieve comparable or better prediction accuracy and posterior probability estimation in comparison with the existing ensemble calibration methods while reducing computational cost.

## Chapter 4 Patient subtyping using classification uncertainty

Even though, the Support Vector Machines (SVMs) [18, 36, 191] have been shown to work well in a large number of applications, they are unable to identify previously unknown conditions. In a recent paper in New England Journal of Medicine [15], and a follow-up in New York Times,<sup>1</sup> Bleyer and Welch point out that after the introduction of early screening, an additional 1.5 million women received a diagnoses of early-stage breast cancer. As the authors point out, this would be very good news if the number of women diagnosed with late-stage cancer also dropped by 1.5 million. However, the late-stage diagnosis dropped by only 0.1 million, suggesting that 1.4 million women received treatments – most of which included surgery, chemotherapy or radiation – for a “cancer” that was never going to make them sick. The ability to correctly identify and profile disease subtypes and patient subgroups is a pre-condition to the ability to distinguish between patients who are sick and need the most aggressive treatments available and those who will never progress, recur, or develop resistance.

This chapter describes the methods we proposed to address this problem. As mention above, one of the most common classification technique, the SVM, is unable to recognize cases that were unknown during training. Here we propose the identification of a region where the SVM prediction should not be trusted, an uncertainty region that will identify possible unknown sub-groups of the population. Even though there has been done work in this direction, all existing methods require the setting of specific costs that are not trivial to compute. The key contribution here is a fully automatic method to reject these uncertain samples.

<sup>1</sup><http://www.nytimes.com/2012/11/22/opinion/cancer-survivor-or-victim-of-overdiagnosis.html>

## 4.1 Related work on classification with rejection

The first seminal work that considers a rejection option, and given the true conditional probability  $\eta(\mathbf{x})$ , formulates the optimal rejection rule, is the work of Chow [31]. Since then, rejection option has been considered for many probabilistic classifiers [86]. One of the early works for increasing the reliability of SVMs by thresholding can be found in [132], which computes an empirical distribution of the margin based on the training data and removes the top  $\alpha\%$  of the margin closest to zero regardless of the label  $Y$ . Later applications of thresholding have confirmed the effectiveness of this approach for including a rejection option ([184, 197]). However, how to choose an optimal threshold has not been clear. Since choosing any threshold  $\tau > 0$  from the output of the hinge loss is not *infinite sample consistent* (Classification-Caliberated) [9], the natural evolution was to develop surrogate convex loss functions that replace the standard hinge loss used in SVM to support a rejection option [9, 209, 81, 201]. However all existing methods require an *extended* cost matrix, or equivalently, require a threshold of the true conditional probability  $\eta(\mathbf{x})$ , to be explicitly defined *a priori*. Furthermore, these surrogate convex losses create twice as many constraints compared to the standard hinge loss, and as such require more samples to reach the same level of accuracy. Given that in most practical applications, deriving with an appropriate extended cost matrix is not trivial, it is worthwhile to forego infinite sample consistency for an adaptive threshold adjustment based on the empirical error.

## 4.2 Uncertainty based on USVM margin

The geometric margin  $\gamma$  of SVM decision hyperplane with regards to the training data set is given by Eq. 2.15. The a geometric, by definition, is the minimal distance between any point in the training set and the hyperplane. We will consider the subspace of the feature

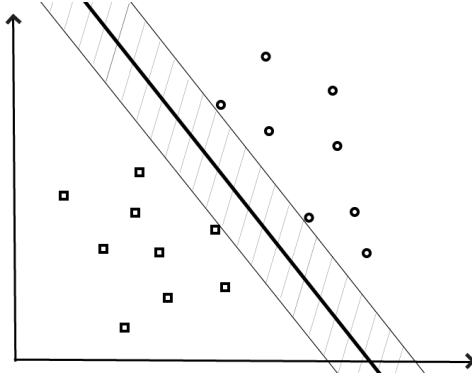


Figure 4.1: An example showing how the uncertainty region  $U$  (the hashed region) is determined in the feature space using  $\gamma$ . The thick line stands for the hyperplane constructed by the SVM classifier. Note that, the distance between the hyperplane and the closest point of each class is constant. This constant is in fact  $\gamma$ . The band-line uncertainty area in the feature space will be warped to a generally non-linear area in the input space by the inverse kernel mapping much as the linear boundary in the feature space will be warped to a non-linear one in the input space. *Source:* [197].

space with the property:

$$-\gamma \leq f(\mathbf{x}) \leq \gamma \quad (4.1)$$

This subspace will include the set of points that are situated at a distance from the class boundary which is less than or equal to the distance from the boundary to the closest point in the training set. Clearly, the training set contains no evidence that the points in this area belong to a class rather than the other one. This area is defined as the uncertainty region  $U$ :

$$U \equiv \{\mathbf{x} \in \mathbb{R}^n \mid -\gamma \leq f(\mathbf{x}) \leq \gamma\} \quad (4.2)$$

and is graphically illustrated in Fig. 4.1. This however imposes a hard threshold which does not permit to incorporate specific costs for uncertainty or misclassification. This can be achieved by the addition of the threshold parameter  $\tau$  that can be used to input the different costs:

$$U \equiv \{\mathbf{x} \in \mathbb{R}^n \mid |f(\mathbf{x})| \leq \tau \cdot \gamma\}$$

The value of the threshold  $\tau$  can be automatically computed using the method in 4.4.

### 4.2.1 Iris data set example

We illustrate the concept of uncertainty region using the Iris data set [64] from the UCI repository [7] using the threshold  $\tau = 1$ . The data set consists of 150 samples, where each sample is described using four attributes: i) petal length, ii) petal width, iii) sepal length and iv) sepal width. The samples are divided into three classes: i) setosa, ii) virginica and iii) versicolor. Among the three classes, setosa and versicolor are linearly separable using petal length and sepal length, whereas virginica and versicolor are non-linearly separable using the same attributes.

We used an SVM classifier with both linear and Gaussian kernels when training on setosa vs. versicolor (linearly separable classes), whereas only the Gaussian kernel was used for classifying versicolor and virginica (non-linearly separable classes). Fig. 4.2 shows the classification results with and without identifying the uncertain regions using  $\gamma$ . When uncertainty areas are not identified, the classifier always assigns a test point to one of the two classes (either red or black regions in Fig. 4.2(a) and Fig. 4.2(c)). For example, the training points do not really provide enough information to reliably determine the class membership of point *A*. Nevertheless, the classical SVM assigns it to one of the classes without providing any feedback to the user that the label of this point is somewhat less trustworthy than the label of a test point in the middle of one of the classes (like point *B* in Fig. 4.2(a)). Fig. 4.2(c) shows the classical SVM with a Gaussian kernel. Point *C* is classified as blue although the data would suggest it is more likely to belong to the green class. Point *B* and similar points in the same direction all the way to infinity will also be classified as blue although they are very different from all training points in either class. Finally, the class label for point *A* will depend on the exact location of the decision hyperplane, determined by tuning parameters, rather than by the distribution of the patterns in the training set. Once the uncertainty areas have been calculated (shown in white in Fig. 4.2(b) and Fig. 4.2(d)), a test sample from those regions will not be assigned to any of the two classes. This will be the way in which the



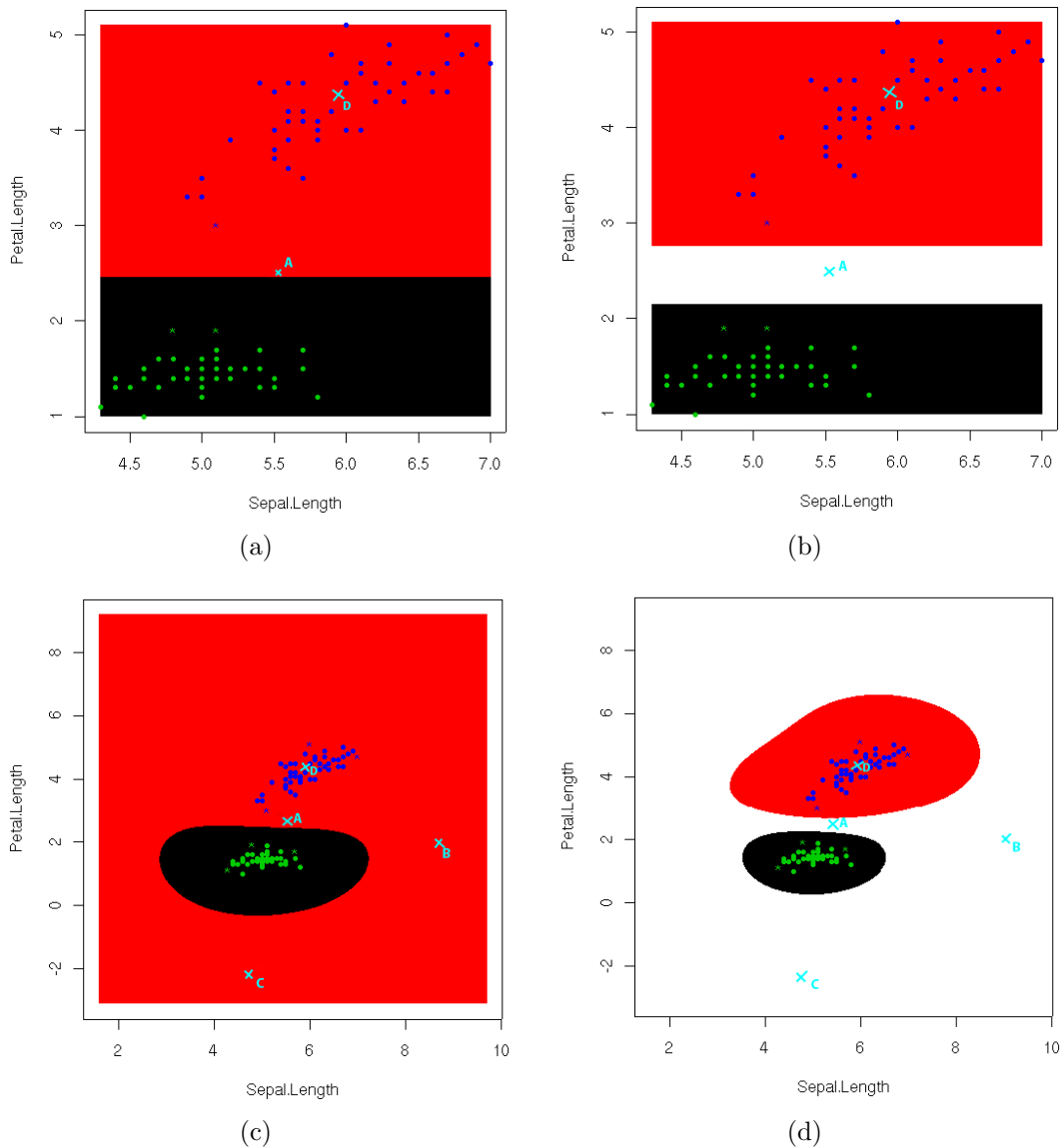


Figure 4.2: Application of soft margin SVM to classify linearly separable setosa and versicolor classes from the Iris data set with and without uncertainty region using linear and Gaussian kernel. The attributes sepal length and petal length were used for classification. (a): The classical SVM using a linear kernel with no uncertainty zone. The training points do not really provide enough information to reliably determine the class membership of point A. Nevertheless, the trained SVM assigns it to one class without providing feedback to the user that the label of this point is somewhat less trustworthy than the label of a test point in the middle of one of the classes like point D. (b): SVM with a linear kernel and with the uncertainty region. The trained SVM will decline to classify point A and all other points in the uncertainty region. (c): The classical SVM with a Gaussian kernel. Point C is classified as blue although the data would suggest it is more likely to belong to the green class. Point B and similar points in the same direction will also be classified as blue although they are very different from all training points in either class. The classification of point A will depend on the exact location of the boundary and tuning parameters rather than on the distribution of the patterns in the training set. (d) The SVM with a Gaussian kernel and uncertainty areas. Points A, B and C are all classified as uncertain conveying to the user, the self-detected limitations of the classifier. *Source:* [197].

classifier will convey to the user that the point may be classified in either class (in-between classes) or that the point is very unlike any of the instances in the training set (towards infinity in any direction, far from any training point). Note that when a linear kernel is used, the problem is not completely solved since a test sample very far from the hyperplane can still be classified into one of the classes. In other words, even with the uncertainty region in the proximity of the hyperplane, the region corresponding to each class still extends to infinity. However, when a Gaussian kernel is used, the region corresponding to each class is bounded as shown in Fig. 4.2(d).

When training a SVM to classify virginica and versicolor classes from the Iris data set, we only used a Gaussian kernel because we knew *a priori* that the data is not linearly separable. Figure 4.3 shows the results with and without the uncertainty regions. Notice that when the uncertainty region is not determined, SVM misclassifies 5 out of 100 training samples. However, when uncertainty region is determined, SVM misclassifies only 2 training samples and does not classify 7 samples to any classes. In addition, when the uncertainty region is not identified, the region corresponding to one of the classes is bounded, while the region corresponding to the other class occupies the rest of the space. However, when the uncertainty region is identified, the region corresponding to each class is bounded.

### 4.3 Uncertainty based on posterior probability

Since the output of SVM,  $f^*(\mathbf{x})$ , is not a probabilistic measure, one of the calibration methods discussed in Section 2.2 can be used to obtain probabilistic scores. Subsequently, the uncertainty areas can be defined using these probabilities. Here, we have opted for Platt's scaling since it is more efficient than the counterpart isotonic regression method [135]. As describe in Section 2.2, Platt's scaling is a parametric calibration method which approximates the posterior probabilities by fitting the output of the SVM to a sigmoidal function (Eq. 2.16). In our experiments, we used the Platt scaling [122] implementation available in the LIBSVM

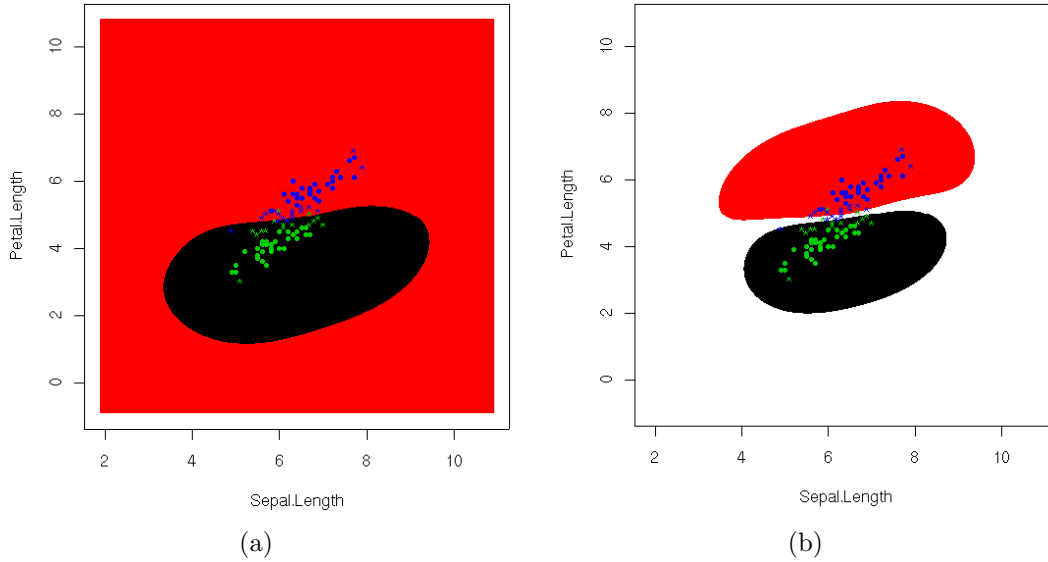


Figure 4.3: Application of soft margin SVM to classify non-linearly separable versicolor and virginica classes from the Iris data set without uncertainty region (a), and with uncertainty region (b), using a Gaussian kernel. The attributes sepal length and petal length were used for classification. Any test sample that is in the uncertainty region (white) is not assigned to either of the classes. *Source:* [197].

package [28].

We define our uncertainty regions based on the class-dependent posterior probabilities. Any sample with posterior probability below the threshold  $\tau$  is uncertain:

$$\mathcal{U} = \{\mathbf{x} \in \mathbb{R}^n, Y_{\mathbf{x}} \in \{-1, +1\} \mid P(Y = Y_{\mathbf{x}} \mid \mathbf{x}) < \tau\}$$

The value of the threshold  $\tau$  will be computed by the method described in the next section.

## 4.4 Automatic detection of uncertainty

Both methods for defining uncertainty regions are dependent on the adequate selection of the threshold  $\tau$ . This threshold represents the trade-off between obtaining uncertain or incorrectly classified samples, and is controlled by their respective costs. The problem we address in this section is the automatic detection of the data-dependent threshold.

	prediction	
	incorrect	correct
uncertain	a	b
certain	c	d

Figure 4.4: To evaluate the effectiveness of using uncertainty we consider the contingency table between correct/incorrect prediction and uncertain/certain label. For us, the true positives will be  $a$  which represents the incorrect predictions that fall in the uncertainty regions. Hence, the precision, recall, and enrichment are defined in terms of  $a$ .

In the process of selecting a threshold, our goal is to eliminate as many incorrect examples as possible (by declaring them as uncertain) without eliminating any correct samples. Hence, we define true positives as the incorrect samples that are in the uncertainty region (Fig. 4.4). The precision, in this case, is defined as the percentage of examples in the uncertain region that are incorrectly classified ( $Prec = \frac{a}{a+b}$ ). The recall is the percentage of incorrectly classified examples in the uncertain regions as a fraction of the total incorrectly classified ( $Rec = \frac{a}{a+c}$ ). Our goal is to simultaneously maximize the precision and recall. This can be achieved by maximizing the  $F_1$  measure [189], which is the weighted harmonic mean of precision and recall ( $F_1 = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec}$ ). We automatically choose the optimal threshold  $\tau$ , as the threshold that achieves the best  $F_1$  measure over all possible threshold values determinable by the training set. In addition, the generalized  $F$  measure,  $F_\beta = \frac{(1+\beta^2)Prec \cdot Rec}{(\beta^2 Prec) + Rec}$ , offers the possibility of selecting a bias towards the precision or recall. Hence, it allows the user to input domain specific knowledge in the automatic selection of the threshold.

Another method for automatic selection of the threshold is to compute the enrichment of the incorrect examples in the uncertainty region. This can be achieved using the Fisher's exact test [62]. The probability of obtaining the observed number of incorrect samples in the uncertainty region just by chance follows a hypergeometric distribution:

$$P(X = a) = \frac{\binom{a+b}{a} \binom{c+d}{c}}{\binom{n}{a+c}} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{a!b!c!d!n!}$$

where  $n = a + b + c + d$ . The probability of observing more than  $a$  incorrect examples in the uncertainty region is  $P(X > a) = 1 - \sum_{i=0}^a P(X = i)$ . By minimizing this measure,

we provide the uncertainty region that incorporates the most incorrect examples while also preserving the largest number of correctly classified examples, and therefore the best choice of threshold  $\tau$ .

Independent of the method for automatic selection of the threshold, our method determines uncertainty regions in the SVM prediction. We will refer to it as SVM with uncertainty or USVM.

## 4.5 Performance on artificial data

We used the artificial data to analyze the behavior of the uncertainty threshold under different degrees of overlap between the classes. We used two types of artificial data sets. The first type consists of two gaussians with the same standard deviation ( $\sigma$ ) and various distances between their two means ranging from 50% to 300% of  $\sigma$ . Two examples of feature generation are presented in Fig. 4.5(a) and Fig. 4.5(b). This data set is called the *twonorm* data set. The second type of artificial data sets is also made up of two gaussians with one mode completely overlapping the other. This data set is called the *ringnorm* data set (Fig. 4.5(c)). For each artificial data set we generate 500 samples equally divided in the two classes and trained an  $L_2$  regularized hinge loss SVM model with an Gaussian kernel. The best parameters ( $\gamma^*, C^*$ ) were chosen using a grid search and five fold cross validation.

In addition to analyzing the performance, we also used the artificial data to show how the uncertainty threshold is obtained both based on the  $F_1$  measure and Fisher's score, and to show the differences between computing the threshold on the geometric margin versus the posterior probability. One example of computing the threshold on the *twonorm* data set is presented in Fig. 4.6 and for the *ringnorm* data set in Fig. 4.7. For each measure we compute all possible values across the entire training set, 4.6(c)-4.6(f) and 4.7(c)-4.7(f), and select the maximum value as the uncertainty threshold. For the Fisher exact test we used the negative logarithm of the p-value and hence the maximum is the most significant

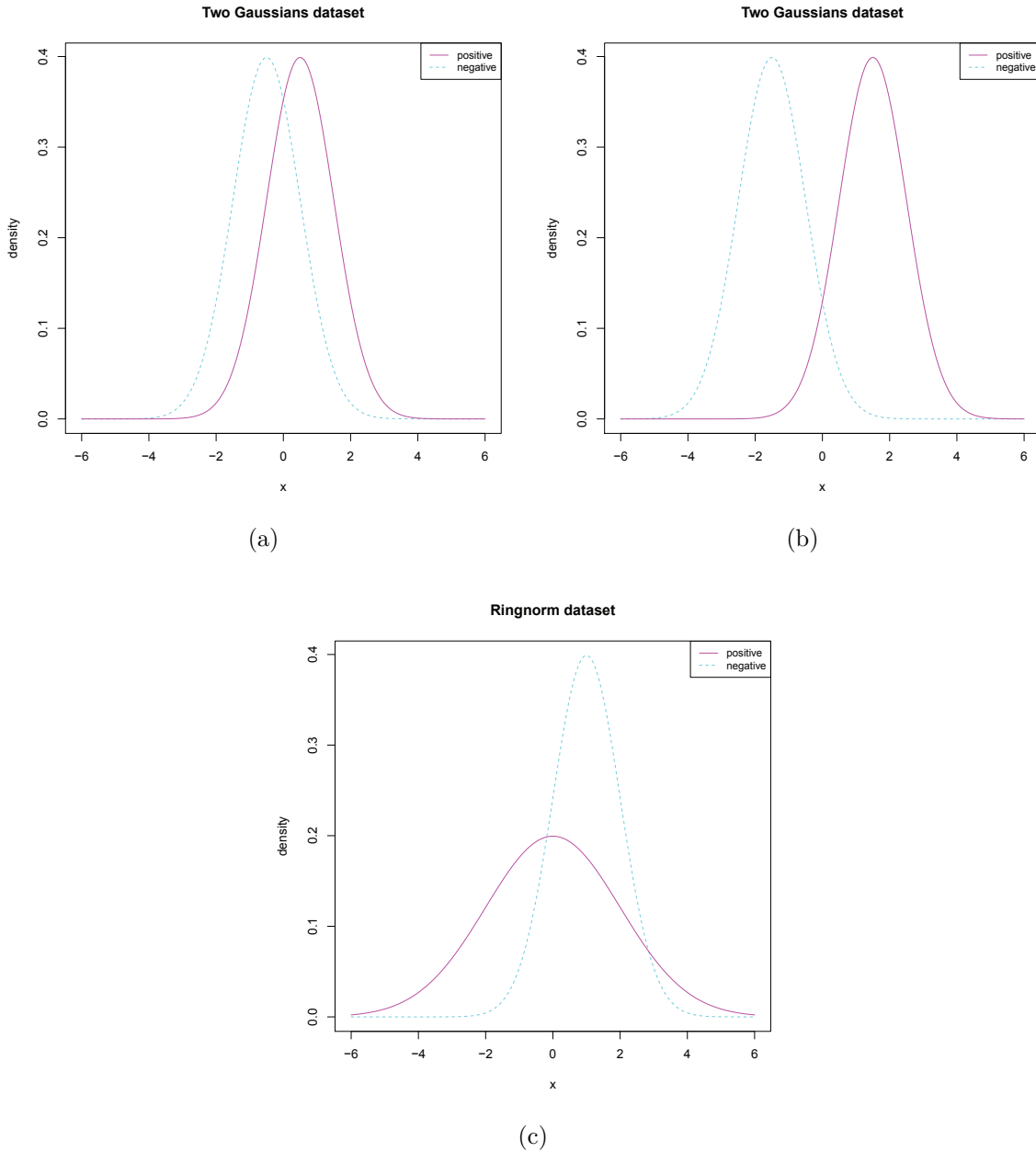


Figure 4.5: Class dependent distributions of each feature in the artificial data sets. Each feature is generated from two normal distributions with the same standard deviation ( $\sigma$ ) ((a) and (b)). We considered eleven different distances between the means of the two distributions ranging from  $0.5\sigma$  to  $3\sigma$ . Two examples are presented in (a) and (b) for the distance equal to  $\sigma$  and  $3\sigma$  respectively. In (c) a different type of artificial data set is considered, the ringnorm data set. In this case the standard deviations of the two normal distributions are no longer identical, they are 1 and 2 respectively, and the distance between the means is  $1/\sqrt{d}$ , where  $d$  is the dimensionality of the data set (in our case equal to 2).

enrichment. These maximums are marked with vertical blue lines. Based on this example, there is no clear difference between using geometric margin (4.6(a) and 4.7(a)) and posterior probabilities (4.6(b) and 4.7(b)) to select the uncertainty threshold. However, in both cases the  $F_1$  score (dotted lines) selects a more stringent threshold in comparison to the Fisher's exact test (dashed lines). Another difference between using the  $F_1$  score and the Fisher's score for selecting the uncertainty threshold is the ability to set a significance threshold. This significance threshold (horizontal brown line in 4.7(c), 4.7(e), 4.6(c) and 4.6(f)) can be used to decide if uncertainty is needed or not. If the maximum Fisher's score is below this significance threshold using uncertainty is not warranted.

Using the *twonorm* data set we estimated the variance of the uncertainty threshold by generating 50 random data sets for each distance between the means. For each iteration we trained a model, computed the threshold and recorded the amount of data reported as uncertain during training. We used this data to determine the confidence intervals of the uncertainty threshold when using the *Fisher's score* in Fig. 4.8(a) and the  $F_1$  score in Fig. 4.8(b). Both  $F_1$  and *Fisher's score* capture the general monotonically decreasing uncertainty in the data sets with the increase in the distance between the means, with the  $F_1$  measure being more conservative and rejecting fewer samples. In addition, there is no clear difference between selecting the uncertainty threshold on the geometric margin or the posterior probability. Since exact misclassification and rejection costs can rarely be defined explicitly, going through the extra step of calibrating posterior probabilities is not justified for selecting the uncertainty threshold. We applied the same procedure for the *ringnorm* data set (Fig. 4.8) and the same conclusions can be drawn: the  $F_1$  score is more conservative and there is no difference in selecting uncertainty threshold between the geometric margin and posterior probability.

The goal of the SVM classification is to predict the class label as accurately as possible. This implies the modeling of decision boundaries and not conditional probabilities. SVMs achieve this by passing the conditional probabilities using the *hinge loss*, thus focusing on

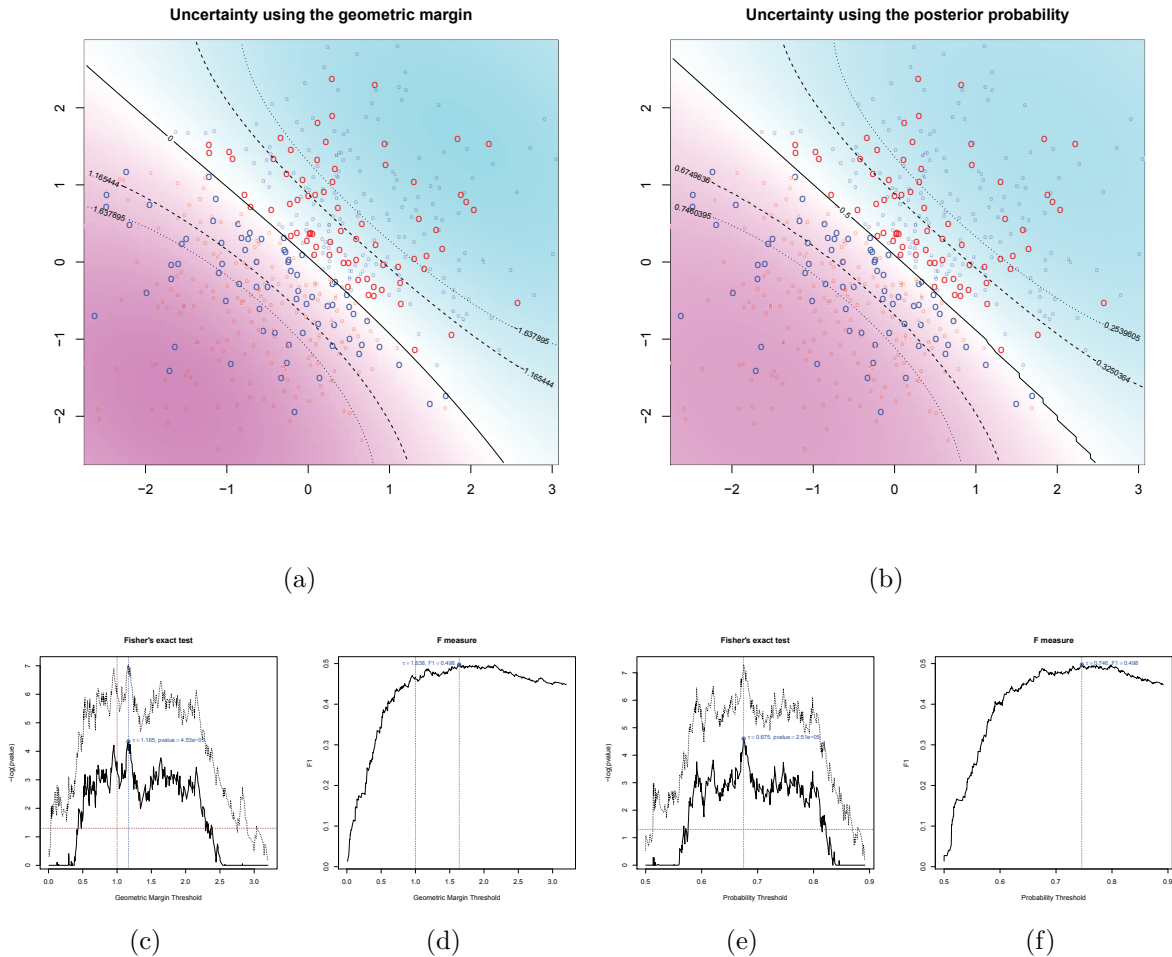


Figure 4.6: Side-by-side comparison of uncertainty region detection using geometric margin (a)(c)(d) and posterior probability (b)(e)(f) on the *twonorm* artificial data set with the distance between the means equal to  $0.5\sigma$ , where  $\sigma$  is the standard deviation. The solid lines are the decision boundaries based on the geometric margin (a) and the posterior probability (b) respectively, while the shades represent the confidence of the discriminant function. The circles are the examples from the training set, with the bold circles representing the ones misclassified during training. The uncertainty thresholds are chosen as the maximum values in the trade-off graphs (c) and (e) for Fisher, and (d) and (f) for  $F_1$ . These optimal thresholds ( $\tau$ ) are marked with vertical blue lines in the trade-off graphs and with dashed (Fisher) and dotted ( $F_1$ ) lines in (a) and (b). The trade-off graphs represent the range of all possible values of Fisher and  $F_1$  respectively over the entire training set. By choosing the maximum on these graphs we optimize the uncertainty region to contain the most misclassified samples and the fewest correctly classified samples at the same time.



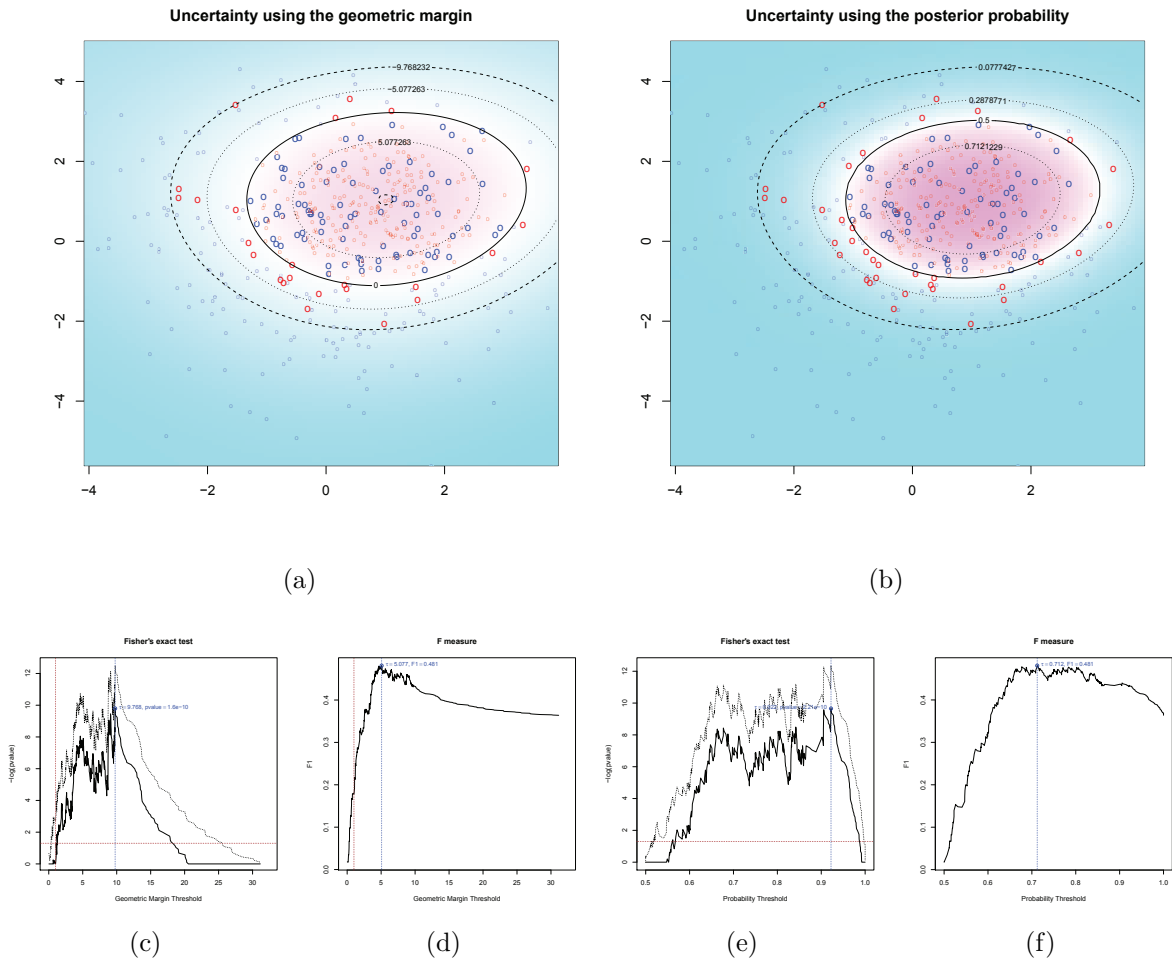


Figure 4.7: Side-by-side comparison of uncertainty region detection using geometric margin (a) and posterior probability (b) on the ringnorm artificial data set which contains two gaussians with one mode completely overlapping the other. The circles represent examples from the training set, while the bold circles represent the misclassified samples during training. The solid lines are the decision boundaries based on geometric margin (a) and posterior probability (b) respectively, and the shades represent the confidence of the discriminant function. The dashed (Fisher) and dotted ( $F_1$ ) lines are the automatically chosen thresholds ( $\tau$ ) for the uncertainty region. The thresholds are chosen as the maximum values in the trade-off graphs (c) and (e) for Fisher, and in the graphs (d) and (f) for  $F_1$ . These maximum values are marked with vertical blue lines in the trade-off graphs. The trade-off graphs represent the range of all possible values of Fisher and  $F_1$  respectively over the entire training set. By choosing the maximum on these graphs we optimize the uncertainty region to contain the most misclassified samples and the fewest correctly classified samples at the same time.

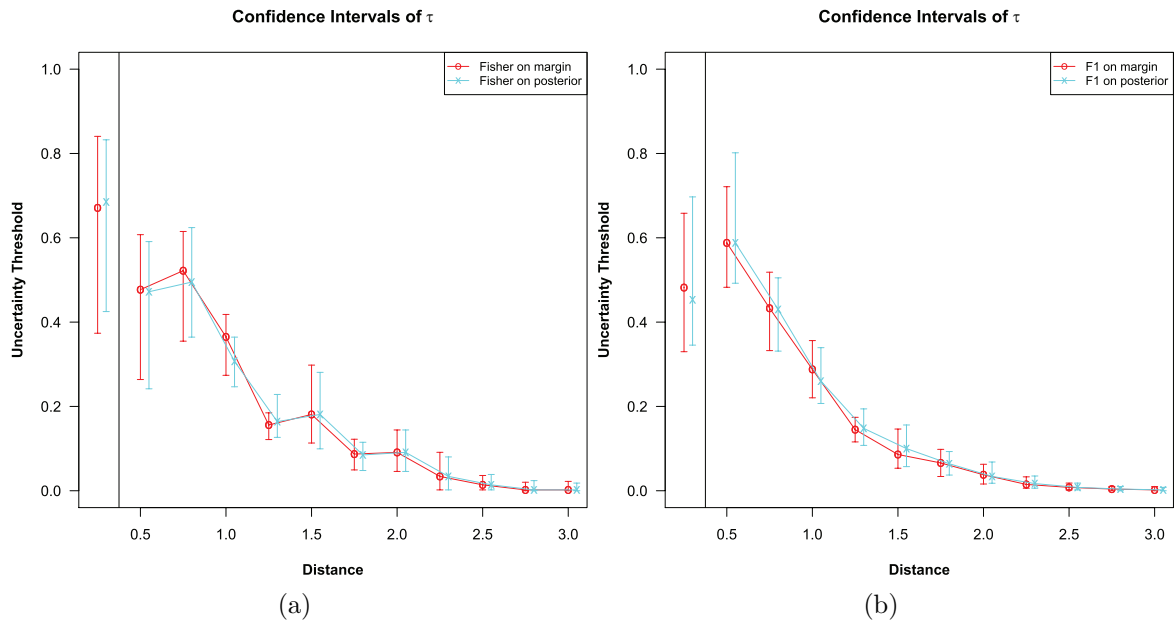


Figure 4.8: Variance of the uncertainty threshold chosen based on Fisher's exact test (a) and  $F_1$  test (b). The variance is assessed on the *ringnorm* data set (left graph of (a) and (b)) and on the *twonorm* data set (right side (a) and (b)) with distance between the means ranging from  $0.5\sigma$  to  $3\sigma$ . Both  $F_1$  and Fisher's exact test capture the monotonically decreasing amount of uncertainty with the increase of the distance between the means. The  $F_1$  measure is slight more conservative and rejects fewer samples. In addition, there is no difference in selecting the threshold on the geometric margin (red curve) or the posterior probability (blue curve).

predicting as accurately as possible near the decision boundary, while extreme points receive less attention. Although calibration procedures try to correct this problem, it is at the expense of further unwarranted computation. Since this region is where the most uncertainty lies, the goal of USVM to quantify the confidence is more in line with the goal of classification and the requirements of low complexity and sparsity than the goal of probabilistic prediction. Essentially predicting accurate probabilities is a much harder problem than classification itself, and sometimes it might not be worthwhile to predict the probabilities accurately in the entire feature space. Our experiments suggest that calibration of SVM outputs leads to comparable results near the decision boundary with further computational cost, which is undesirable for a rejection option when costs are unknown.

## 4.6 Performance on benchmark data sets from the UCI machine learning repository

We compared the performance of USVM with SVM on two data sets from the UCI machine learning repository [7]. The first one is the Pima Indian diabetes data set (referred to as *pima*), which is a binary classification task with eight features and 768 samples. The second data set is the large scale high-dimensional *mnist* digit dataset [121]. It consists of 60,000 training and 10,000 testing samples of hand written digits. There are ten classes with approximately equal number of instances in each class. Each instance is a  $28 \times 28$  pixel image, thus 784 features in total. All images are centered, and their scale and rotation normalized. For all data sets we used the Gaussian kernel and choose the bandwidth  $\gamma$ , regularizer  $C$  and the uncertainty threshold  $\tau$  using cross-validation on the training set. The model for *pima* is tested during cross-validation, while the model for *mnist* is tested on the independent test set provided in the UCI machine learning repository.

On the *pima* data set, the uncertainty threshold is chosen such that the region of uncertainty is highly error prone (40.37%) in comparison with the acceptance region (13.05%).

	SVM	USVM	Uncertainty region	
			error rate	percentage discarded
D vs. H	22.66%	13.05%	40.37%	35.16%

Table 4.1: The error rate of the SVM prediction is reduced by approximately 44% when using the  $F_1$  automatic method to compute the uncertainty threshold on the pima data set. This is achieved by discarding the points in the uncertain region which is characterized by a higher error rate.

In terms of overall performance, USVM reduces the error rate by approximately 44% over the standard SVM. The cost of this performance increase is the percentage of samples that were declared uncertain - 35% of the test set. These results are summarized in Table 4.1. In addition, in Fig. 4.9, we present the evolution of the USVM accuracy (solid curve) and uncertainty percentage (dashed line) for all the possible thresholds on the geometric margin. Based on this figure, it can be noticed that the selection of the threshold is subjective and based on prior knowledge of the data set it can be chosen more or less stringent. However, USVM automatically selects the threshold that achieves the best odds ratio of errors in the uncertainty region.

For the *mnist* data set, we decomposed it into binary classification problems. We performed the comparison for each digit against the rest. For each one of the ten classifiers, USVM reduces the error rate by approximately half by only declaring on average 2% of the test data to be uncertain (Table 4.2). This is achieved by choosing the uncertainty region such that it is highly error prone. For example, for the digit 8, 75% of the samples in the uncertainty region are errors. We present in Fig. 4.10 a set of samples that were declared uncertain by USVM. Each row represents one of the ten digit classifiers with the left side containing false negatives and the right side false positives.

## 4.7 Results on clinical samples

A first such data set we analyzed involved gene expression profiles of leukemia patients.

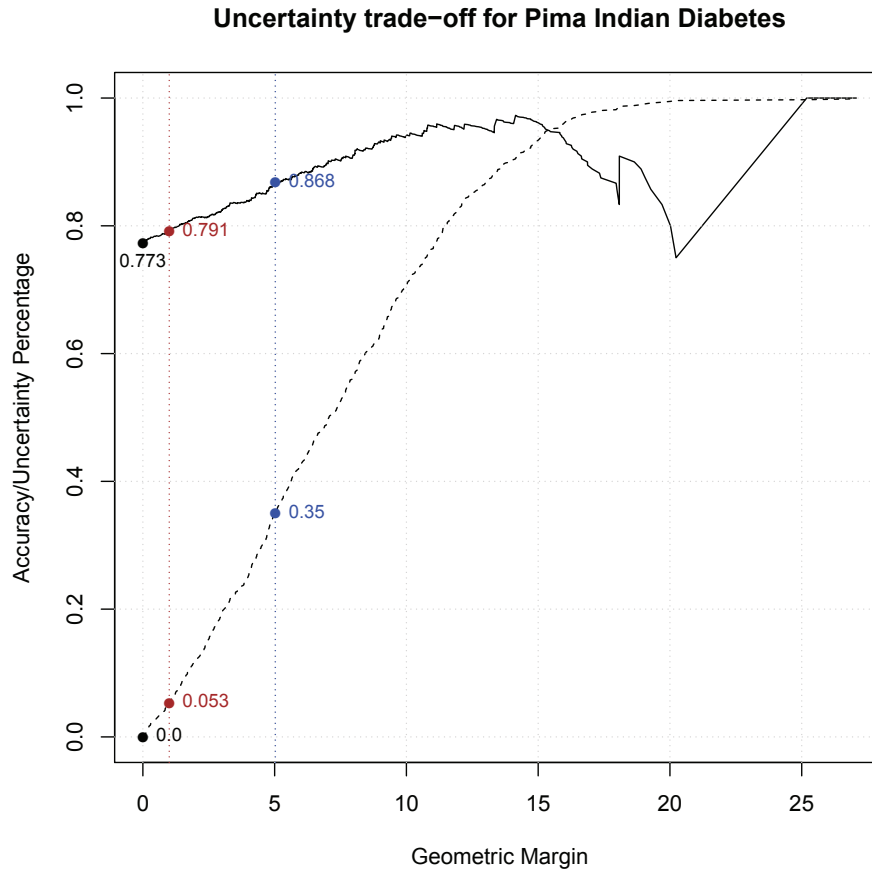


Figure 4.9: Evolution of the accuracy (solid curve) and uncertainty percentage (dashed curve) with the increase in geometric margin. The vertical lines represent the standard geometric margin  $\gamma$  (brown) and the optimal threshold chosen using the  $F_1$  score (blue). The standard SVM accuracy and uncertainty are represented by the black points (77.30% and 0% respectively), while USVM accuracy and uncertainty by the blue points (86.8% and 35% respectively).

	SVM	USVM	Uncertainty region	
			error rate	percentage discarded
0 vs. ALL	0.81%	0.33%	59.26%	0.82%
1 vs. ALL	0.45%	0.12%	35.48%	0.93%
2 vs. ALL	1.66%	0.84%	57.75%	1.44%
3 vs. ALL	1.93%	0.82%	47.01%	2.39%
4 vs. ALL	1.69%	1.01%	55.28%	1.25%
5 vs. ALL	2.01%	0.93%	34.82%	3.19%
6 vs. ALL	1.01%	0.56%	60.81%	0.75%
7 vs. ALL	1.81%	0.78%	47.47%	2.21%
8 vs. ALL	3.18%	1.71%	75.65%	1.99%
9 vs. ALL	3.11%	1.71%	44.51%	3.29%

Table 4.2: For all the class specific models on the mnist data set the error rates of the SVM predictions are approximately halved when using  $F_1$  automatic method to compute the uncertainty threshold. Note the high percentage of errors discarded by using the uncertainty region.

0	0	0		8	5	2	3
1	1	1	1	4	7	9	8
2	2	2	2	8	3	4	7
3	3	3	3	5	8	1	2
4	4			9	8	6	5
5	3	5	5	3	6	8	0
6	6	6	6	4	8	2	5
7	7	1	7	9	2	8	3
8	8	8	8	9	5	2	3
9	9	9	9	4	7	5	6

Figure 4.10: Sample of errors declared as uncertain by USVM on the mnist data set. Each row is the result of the discrimination between digits 0 through 9 against all the remaining digits. The left side represent false negatives (target digit samples that would have been missclassified without uncertainty) and the right side are false positives (other digit samples that would have been missclassified without uncertainty). The empty spaces are due to lack of more false negatives for classes 0 and 4.

Armstrong et al. [4] showed that Acute Lymphoblastic Leukemias carrying a chromosomal translocation involving the mixed-lineage leukemia gene (MLL, ALL1, HRX) have different expression profiles from conventional acute lymphoblastic (ALL) and acute myelogenous leukemias (AML). We use this data set to show that USVM can detect the novel MLL subtype from the conventional ALL and AML.

The MLL data set contains gene expression profiles of three leukemia subtypes (ALL, AML and MLL), each class containing (24, 28 and 20) samples respectively. The data from all classes were normalized together using Robust Multiarray Average (RMA) [98] and gene expression was obtained by averaging all the probesets associated to the same gene for all 8,655 genes covered by the microarray. Subsequently, the MLL class was set aside for testing and both SVM and USVM were trained on samples contained in the ALL and AML groups. The training was performed on 11 genes selected using the nearest shrunken centroid [181] method only on the training data. After training the model using our USVM method we evaluated it on the MLL test samples. Out of the 20 MLL test samples, 13 were found to be dissimilar to the samples used for training and were classified as not belonging to either ALL, nor AML. As expected the classical SVM misclassified all the MLL samples as either ALL or AML. We show a graphical comparison between SVM and USVM using the first three primary components obtained on the training set (Fig. 4.11). The red samples (ALL) and blue samples (AML) are the training set, while the rest are all MLL testing samples that were either misclassified as ALL (orange), or misclassified as AML (cyan), or rejected in the case of USVM (gray). USVM is able to correctly classify 65% of the samples from the new class as a never-seen-before subtype.

The second dataset is a cardiocography dataset (CTG) that contains 2,126 fetal cardiocograms with 39 features [6, 7]. This dataset is particularly interesting for detecting uncertain samples, since the fetal state, which is to be classified, is labeled as Normal, Pathologic and Suspect. In essence, the Suspect class includes those samples of which even the human experts were uncertain of. Our premise was that by training only on the Normal

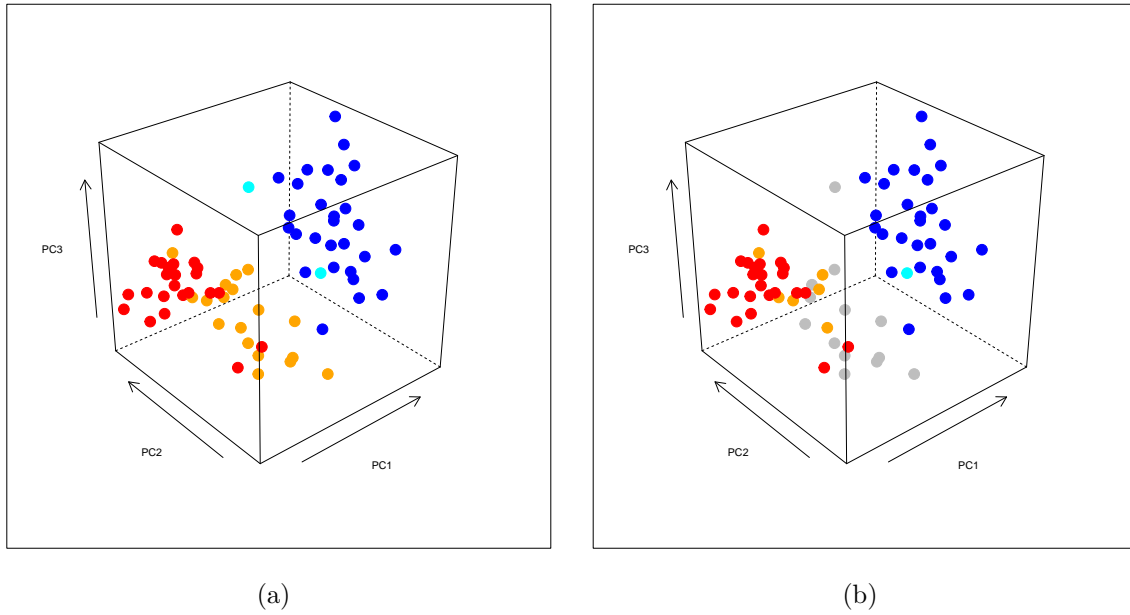


Figure 4.11: Plot of the MLL data set using the first three primary components. The models for (a) SVM and (b) USVM were trained on the AML (red) vs. ALL (blue) samples and tested against the MLL samples. SVM fails to identify any of the MLL samples and misclassifies all of them as either AML (orange) or ALL (cyan). USVM identifies 65% of the MLL samples (gray) as being different from both AML or ALL.

against Pathologic samples our method is able to identify during the prediction the Suspect samples. In the cross-validation stage, the error rate was reduced by approximately 77%, from 1.3% for the classical SVM to 0.3% for USVM. This happened because the uncertainty region of the USVM captured a lot of the SVM errors (41.67% of the SVM prediction in this area are errors). In turn, this led to a significant decrease of errors in the prediction areas. Even though only a small percentage of samples are declared uncertain by USVM (1.91%), this margin of uncertainty identifies 21.36% of the Suspect cases while SVM is unable to identify any.

## 4.8 Summary

The widely used SVM classifiers essentially make guesses for those points of the input space situated very close to the decision boundary (i.e., samples that are hard to classify). In spite of the fact that these guesses are well-informed and shown to minimize the error in lack of any additional information [192], these samples may in fact be an unforeseen division



in the study population. A better characterization of the patients selected for a clinical trial is one of the key ingredients to ensure its success [69]. To this extent we proposed a method that automatically identifies samples that hard to classify between the two groups under study and may result in the discovery of a new subgroup in the study population.

# Part II

## Understanding the Disease Mechanism Using Systems Biology

### Chapter 5 Biological networks and available analysis methods

Together with the ability of generating a large amount of data per experiment, high throughput technologies also brought the challenge of translating such data into a better understanding of the underlying biological phenomena. Independent of the platform and the analysis methods used, the result of a high-throughput experiment is, in many cases, a list of differentially expressed (DE) genes. The common challenge faced by all researchers is to translate such lists of DE genes into a better understanding of the underlying biological phenomena and in particular, to put this in the context of the whole organism as a complex system. More precisely, we focus on the task of identifying the pathways, as defined in Section 5.1, that are significantly impacted in a given experimental condition. This chapter focuses on presenting alternative approaches currently available to tackle this problem. In a recent review [114] the existing pathway analysis methods were grouped in three generations based on the order they appeared and the new features they introduced. In the context of this thesis a simpler division is more appropriate: i) gene set analysis methods (first and second generation [114]) - Section 5.2 - and ii) methods that include topology information (third generation [114]) - Section 5.3.

## 5.1 Biological pathways and available resources

The purpose of this section is to introduce the biological concepts involved and to describe the available resources for computational analysis. The genome (i.e., all the genes in an organism) is considered to be the "source-code" for all the processes that take place in a cell and in the organism as a whole. Various technologies have been developed to measure the level of gene expression. The process of gene expression represents the "execution" phase through which the code stored in the genes is used to produce proteins and enzymes that will perform particular functions in the cell. It was later understood that the genes products do not act independently, but come together to perform more complicated functions. How these processes take place and what the interactions represent are described in pathways.

A **pathway** can be defined as a part of a larger system that has a clear set of properties: i) it has a set of components (i.e., genes, proteins, metabolites, etc.); ii) it has a set of relations between the components (i.e., gene signals, chemical reactions, etc.); iii) there are more shared properties between the components of the pathway than the rest of the system; iv) the components work together to accomplish a clearly defined task. Even though a generally accepted definition of a pathway does not exist, the definition outlined here is general enough to describe most available resources. Currently, there are three general classes of pathways based on the components and the interactions between them: signaling pathways, metabolic pathways and protein-protein interaction networks.

**Signaling pathways** use nodes to represent genes or gene products and edges to represent the signals that are passed from one gene to another (e.g., activation, repression, etc.). As an example, Fig. 5.1 shows the apoptosis signaling pathway, as constructed by KEGG, which describes the genetically controlled mechanisms responsible for cell death. The different types of interactions (signals) are represented by different edges (e.g., activation between *FADD* and *CASP10*, inhibition between *IAP* and *CASP6*, etc.). A full diagram of all possible interactions is displayed in Fig. 5.2. Even though the definition of a signaling pathway seems

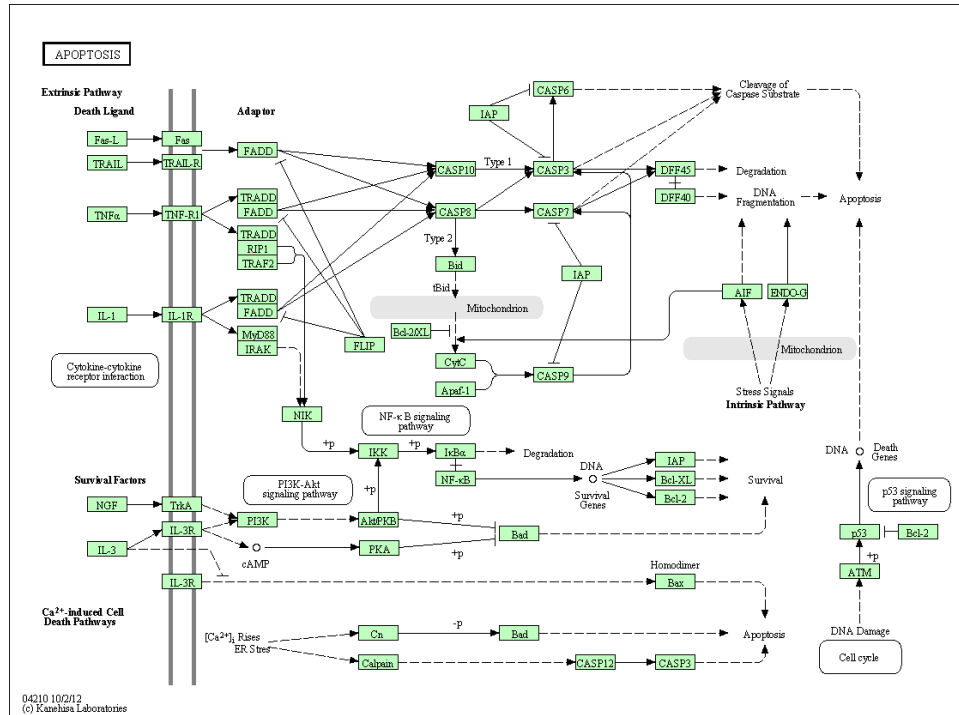
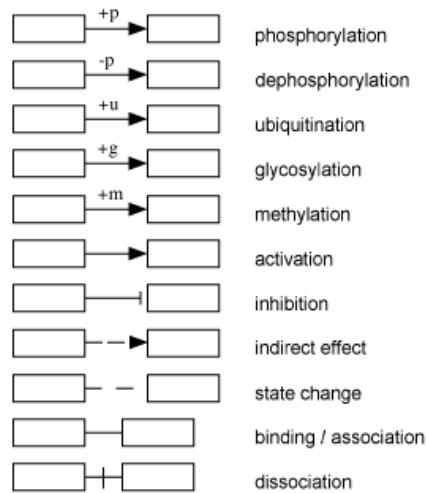
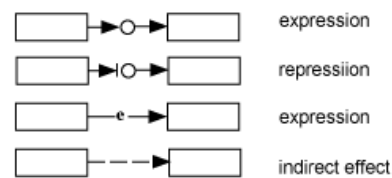


Figure 5.1: KEGG Apoptosis signaling pathway. *Source:* KEGG - <http://www.genome.jp/kegg/pathway/hsa/hsa04210.html>.

#### Protein-protein interactions



#### Gene expression relations



#### Enzyme-enzyme relations



Figure 5.2: The types of interactions in a KEGG signaling pathway. *Source:* KEGG - [http://www.genome.jp/kegg/document/help\\_pathway.html](http://www.genome.jp/kegg/document/help_pathway.html).

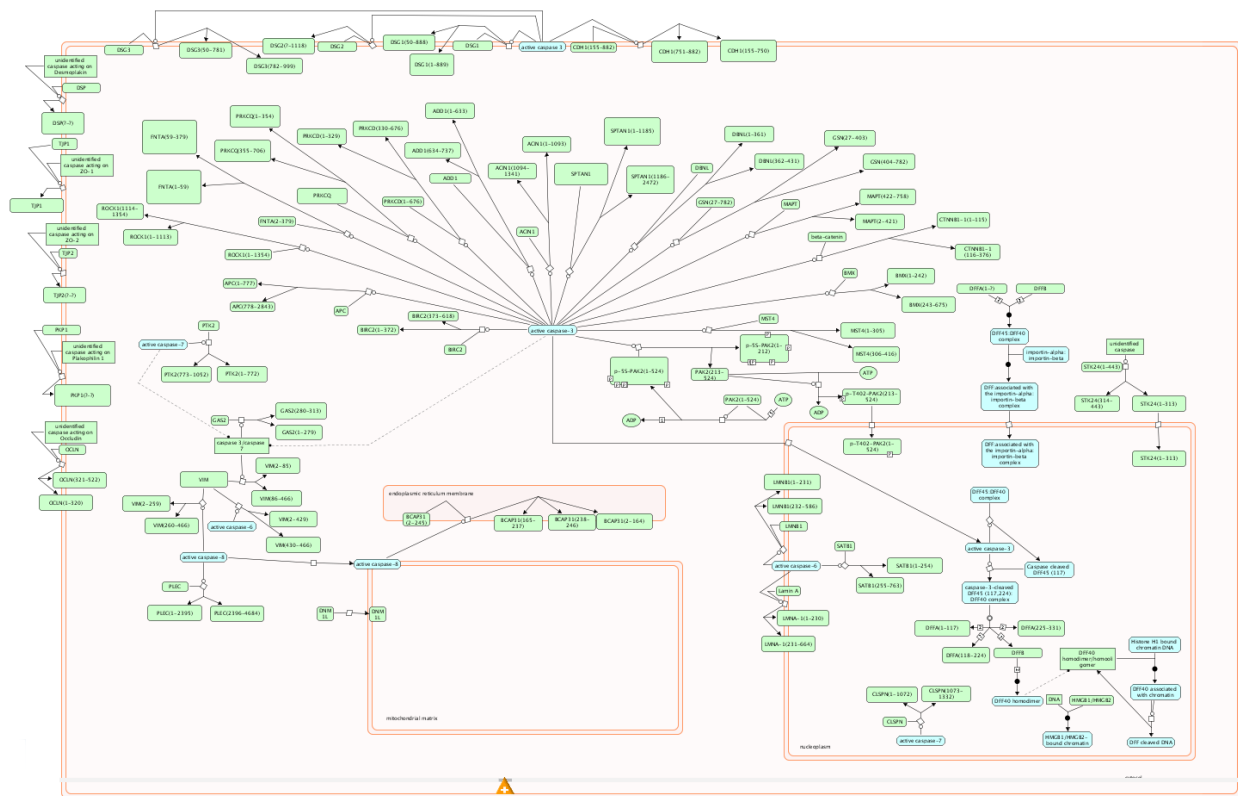


Figure 5.3: Reactome apoptosis execution phase pathway. *Source:* Reactome - <http://www.reactome.org/>.

clear, there is still a large variation from one data provider to another. See Fig. 5.3 for the same pathway as described by Reactome [102, 39].

**Metabolic pathways** use nodes to represent biochemical compounds as nodes and edges to represent chemical reactions that produce, combine or transform these compounds. In some cases these reactions are facilitated by catalysts, a role carried out by enzymes. As an example, see Fig. 5.4 describes the metabolism of fatty acids. Notice the enzyme commission (EC) numbers on the edges which represent the gene products in the form of enzyme that catalyze the reactions.

In a **protein-protein interaction (PPI)** network the nodes represent proteins, while the edges represent physical interactions, such as binding, between two proteins. The main problems with these networks, when it comes to pathway analysis as focused in this thesis, is that they do not imply causality and they only represent a possible interaction between two proteins. If the two proteins are not in the same cellular location at the same time, the

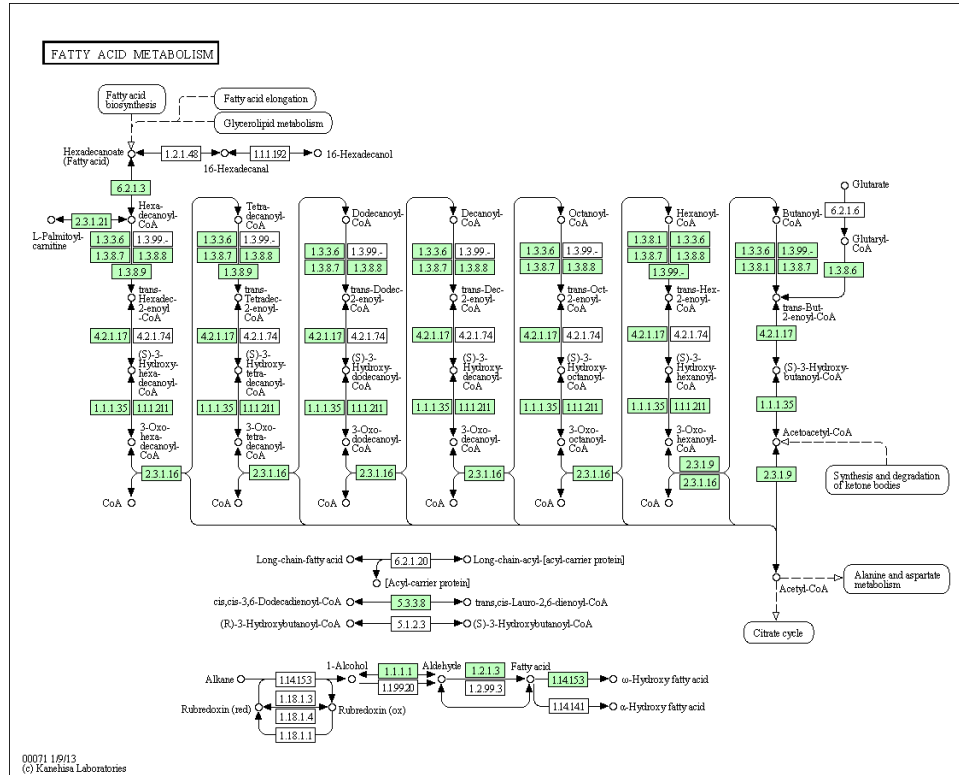


Figure 5.4: KEGG Fatty acid metabolism pathway. Source: KEGG - <http://www.genome.jp/kegg/pathway/hsa/hsa00071.html>.

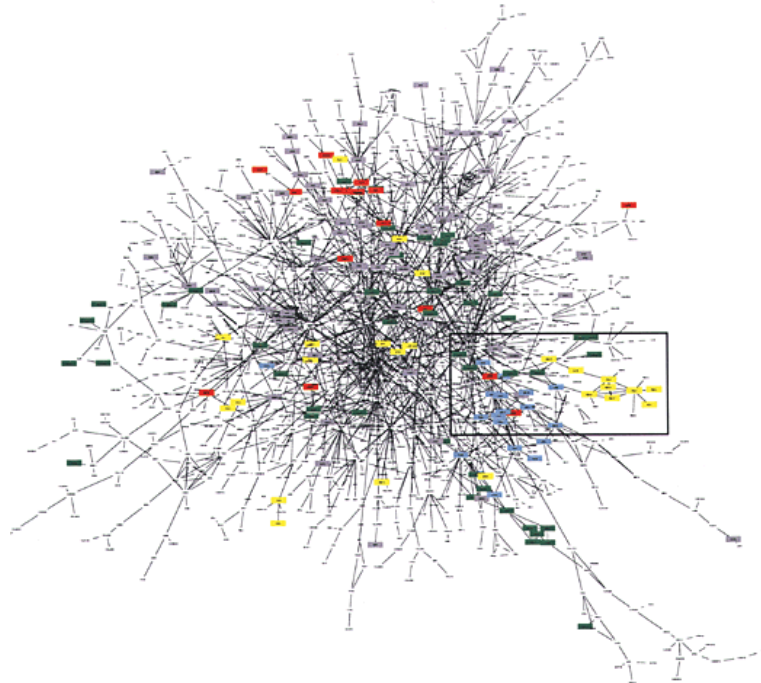


Figure 5.5: Yeast protein-protein PPI network. Source: [160].

binding never happens. Fig. 5.5 shows an yeast PPI network. Examples of databases that provide such PPI networks include: BioGRID [170, 29], STRING [65], etc.

Several pathway databases such as KEGG [137, 103, 104, 105], BioCarta [14], WikiPathways [106], Pathway Commons [26], PANTHER Pathway [93], SMPDB [67] and Reactome [102, 39], currently describe such signaling and metabolic pathways. Various methods, which will be described in the next sections, were designed to take advantage of the rich information available in these databases. In general, the term pathway analysis in this thesis is used for analysis methods that compare two phenotypes and identify the pathways that are significantly impacted in the given condition. These methods receive as input the experimental data and pathway database information and return a ranked list of impacted pathways.

## 5.2 Gene set analysis methods

The general characteristic of the methods in this group is that they consider the pathways as simple set of genes and ignore the interactions between them. Most of the techniques that are part of this group focus on the gene enrichment analysis in the pathway (viewed as a gene set). These methods were first designed to deal with the Gene Ontology (GO) and identify the enrichment of the genes annotated to specific GO terms. We reviewed over 20 tools available for computing enrichment using the gene ontology [110]. A more recent review focused on the tools available for enrichment analysis reported over 68 tools available [94] that were divided into three classes: i) singular enrichment analysis (SEA); ii) gene set enrichment analysis (GSEA) and iii) modular enrichment analysis (MEA). This division is a specific case of the generation division in [114] from the perspective of enrichment tools: i) first generation or over representation analysis (ORA) or SEA; ii) second generation or functional class scoring (FCS) or GSEA; iii) topology based methods or MEA.

### 5.2.1 Over representation analysis

One of the first over representation analysis using the Gene Ontology (GO) was proposed in 2002 [111, 50]. This functional profiling takes a list of DE genes and uses a statistical analysis to identify the GO categories (e.g. biological processes, etc.) that are over- or under-represented in the condition under study. Given a set of DE genes, this approach compares the number of DE genes found in each category of interest with the number of genes expected to be found in the given category just by chance. If the observed number is substantially different from the one expected by chance, the category is reported as significant. A statistical model (e.g. hypergeometric) can be used to calculate the probability of observing the actual number of genes just by chance:

$$P(X = K_p | N, N_p, K) = \frac{\binom{N_p}{K_p} \cdot \binom{N - N_p}{K - K_p}}{\binom{N}{K}} \quad (5.1)$$

where:  $K_p$  and  $K$  are the number of number of DE genes that fall in the pathway and in total, and  $N_p$  and  $N$  are the number of reference genes that fall in the pathway and in total respectively. The probability of having a value less than or equal to the observed value is then:

$$\sum_{i=0}^{K_p} \frac{\binom{N_p}{i} \cdot \binom{N - N_p}{K - i}}{\binom{N}{K}} \quad (5.2)$$

Therefore, the probability of having a value greater than or equal to the observed value (the probability of observing  $K_p$  or more DE genes on this pathway) is the over representation



p-value, which can be expressed as:

$$p = 1 - \sum_{i=0}^{K_p} \frac{\binom{N_p}{i} \cdot \binom{N - N_p}{K - i}}{\binom{N}{K}} \quad (5.3)$$

By computing this p-value for each of the functional categories available and applying a family wise error correction, the set of functional categories or pathways that are over represented by the set of DE genes can be selected.

Currently, there are over 40 tools using this over-representation approach (ORA) [42, 212, 2, 11, 25, 12, 161, 125, 214, 91, 150, 208] using one or several statistical models like: hypergeometric, binomial, chi-square, etc. [50, 94]. In spite of its wide adoption, this approach has a number of limitations related to the type, quality, and structure of the annotations available [110].

## 5.2.2 Functional class scoring

An alternative approach considers the distribution of the pathway genes in the entire list of genes and performs a functional class scoring (FCS) which also allows adjustments for gene correlations [77, 143]. One of the most important differences between ORA and FCS approaches is that ORA relies on a previous selection of a subset of differentially expressed genes while FCS considers the entire list of genes (also known as *cut-off* free analysis).

The most popular method in the FCS category, the Gene Set Enrichment Analysis (GSEA) [131, 172, 180], ranks all genes based on the correlation between their expression and the given phenotypes (Fig. 5.6). A running score is computed by walking down the list of genes ordered by expression change. The score is increased for every gene that belongs to the gene set and decreased for every gene that does not. The enrichment score (ES) is the maximum deviation from zero (Fig. 5.6) and corresponds to the a weighted Kolmogorov-Smirnov statistic [172]. The enrichment score reflects the degree to which a given pathway

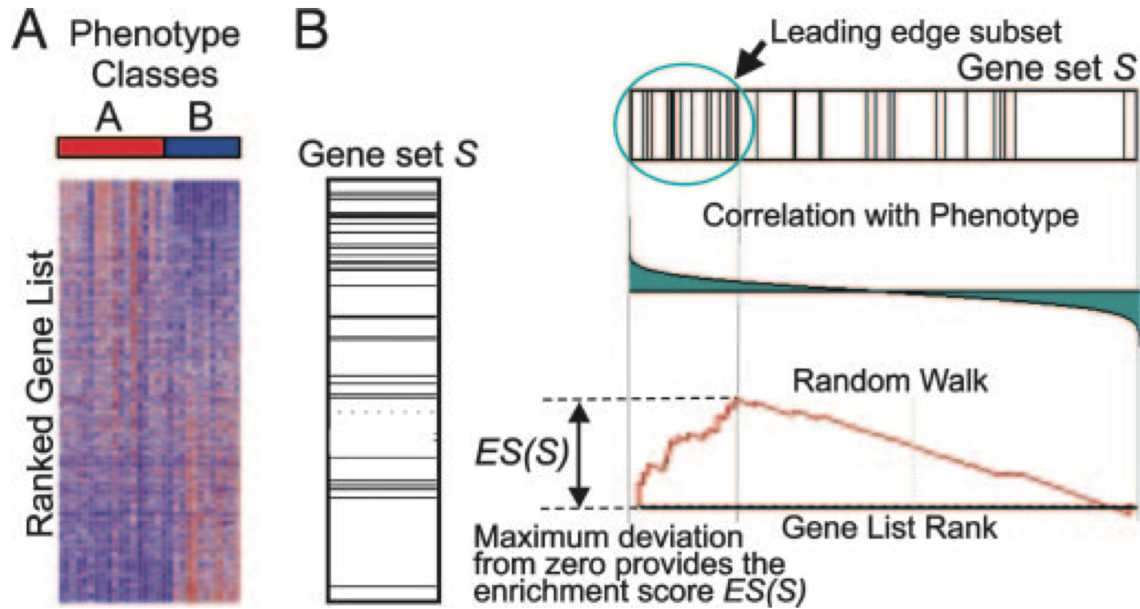


Figure 5.6: Gene set enrichment analysis: i) rank the genes based on their correlation with the phenotype (A); ii) calculate the enrichment score of the gene set by walking down the gene list and increasing the score when the gene belongs to the set and decrease otherwise. *Source:* [172].

is represented at the extremes of the ranked list. Statistical significance for each gene set is established with respect to a null distribution constructed by permutations. Another FCS method is the Gene Set Analysis (GSA) method [58] that was proposed to improve on GSEA by using the maxmean as the summary statistic. In addition, GSA re-standardizes the gene set scores in order to take in consideration the distribution of all possible gene set scores. Several methods using similar approaches were proposed [1, 99, 119, 96].

### 5.2.3 Non-independent gene sets

All the methods presented so far compute a score with a significance for each one of the available pathways (as gene sets) and therefore considers them to be independent. Given that the processes on all pathways take place at the same time, in the same cell, this assumption is flawed. A few approaches presented recently try to address this problem. The general characteristic of these methods is that they consider pathways to be non-independent and incorporate this information in assessing the significance of each pathway.

One approach is to apply a correction after computing the significance of each gene set. A general method for correcting the cross-talk effects between pathways was exemplified using an ORA approach [44, 45]. This method is analyzing the overlap between sets of genes in each pathway and assesses based on the given phenotype the likelihood for a particular gene to be involved in one pathway or another.

Another approach is the PADOG [175] method that considers the frequency of each gene in the set of all pathways. The score for each gene set is a weighted mean of absolute moderated t-scores of the genes belonging to the set. The weight for each gene is computed in such way that the genes that appear in fewer pathways have a higher weight. In addition, it performs a re-standardization as presented in [58]. This method belongs to the FCS category and it was shown to perform better than GSEA or GSA [175].

### 5.3 Topology based analysis methods

The gene set analysis methods used for pathway analysis are limited by the fact that each functional category is analyzed independently without a unifying analysis at a pathway or system level [180]. These approaches are not well suited for a systems biology approach that aims to account for system level dependencies and interactions, as well as identify perturbations and modifications at the pathway or organism level [171].

Firstly, *these approaches consider only the set of genes on a pathway and ignore their positions in that pathway.* This may be unsatisfactory from a biological point of view. If a pathway is triggered by a single gene product or activated through a single receptor and if that particular protein is not produced, the pathway will be greatly impacted. A good example is the insulin pathway (Fig. 5.7). If the insulin receptor (*INSR*) is not present, the entire pathway is shut off. Conversely, if several genes are involved in a pathway but they only appear somewhere downstream, changes in their levels may not affect the given pathway as much.

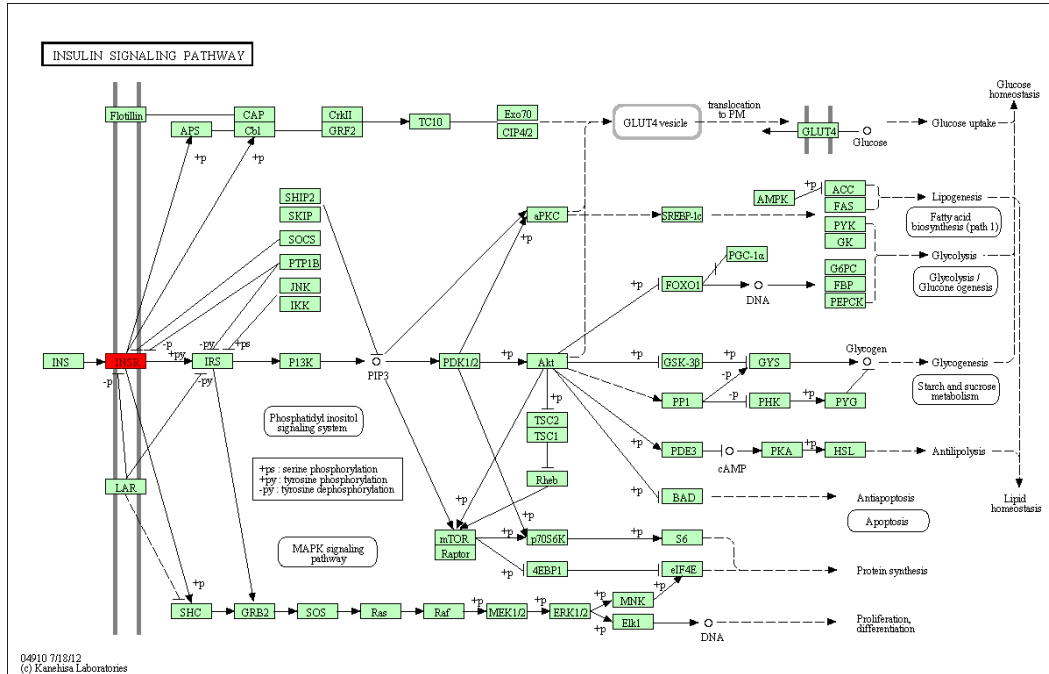


Figure 5.7: KEGG Insulin Signaling Pathway. The insulin receptor (*INSR* in red) is the only entry point in this pathway and big change in its expression will have a high effect on the entire pathway. *Source:* KEGG - <http://www.genome.jp/kegg/pathway/hsa/hsa04910.html>.

Secondly, some genes have multiple functions and are involved in several pathways but with different roles. For instance, the above *INSR* is also involved in the adherens junction pathway as one of many tyrosine kinase receptors. However, if the expression of *INSR* changes, this pathway is not likely to be heavily perturbed because *INSR* is just one of many receptors on this pathway.

We addressed the limitations of classical approaches by proposing the very first method to incorporate interaction knowledge into the analysis of signaling pathways [52]. The impact analysis extended the classical analysis by incorporating important biological factors like (i) the magnitude of expression change for each gene, (ii) their position on the pathway, as well as (iii) the type of gene interactions on the pathway (the method is described in more detail in Chapter 6). This was soon followed by a plethora of other methods [97, 52, 54, 55, 59, 60, 61, 71, 75, 76, 82, 95, 100, 127, 130, 150, 165, 166, 176, 193, 101, 205, 213]. The majority of them use variations of centrality measures (e.g., node degree, node betweenness, etc.) to score genes according to their position in the pathway and the number of neighboring genes. In addition,

methods like ScorePAGE [150] and PWEA [95] also use gene expression similarity measures (e.g., correlation coefficients) between genes on the same pathway to identify tight clusters of highly correlated genes. Alternatively, methods like PARADIGM [193], PathOlogist [59], TAPPA [71], BPA [100] consider the expression of genes (i.e., nodes) in the pathway as random variables and use the interactions to define conditional dependency. Independent of the model used to incorporate pathway topology, all these methods focus on identifying the significantly impacted pathways in a given experimental condition.

## Chapter 6 The impact analysis

This chapter describes the first method that incorporates topology in the pathway analysis, the impact analysis. This method is able to include in the analysis important biological factors such as: i) type and position of each of the differentially expressed genes in the pathway; ii) the magnitude of their expression change; and iii) the type of interaction between all genes in the pathway. This method was proposed by our group [52] and my original contribution are the algorithms to propagate the gene perturbation through the pathway. My contribution also consisted in the development of a novel type of impact analysis able to capture the individual gene significance. This is the first method ever proposed that is able to take into consideration this type of information in establishing the significance of a given pathway [195].

### 6.1 Impact factor

The goal of the impact analysis was to push the pathway analysis of high throughput data beyond the currently ubiquitous approach that looks at a pathway as a mere set of genes. We achieved this by adding new dimensions, able to capture the phenomena related to the complex interactions and signaling described by the pathway topology. Here, we present an analysis model that would require both a statistically significant number of differentially expressed (DE) genes *as well as* biologically meaningful changes on a given pathway. In this model, the *impact factor* (IF) of a pathway  $P_i$  is calculated as the sum of two terms:

$$IF(P_i) = \log\left(\frac{1}{p_i}\right) + \frac{\sum_{g \in P_i} |PF(g)|}{|\Delta E| \cdot N_{de}(P_i)} \quad (6.1)$$

The first term is a probabilistic term that captures the significance of the given pathway  $P_i$  from the perspective of *the set of genes* contained in it. This term captures the infor-

mation provided by the currently used classical statistical approaches and can be calculated using either an ORA (e.g., z-test [46], contingency tables [139, 142], etc.), a FCS approach (e.g., GSEA [131, 172]) or other approaches [22, 155, 180]. The  $p_i$  value corresponds to the probability of obtaining a value of the statistic used at least as extreme as the one observed, when the null hypothesis is true. The results presented here were obtained using the hypergeometric model [177, 50] in which  $p_i$  is the probability of obtaining at least the observed number of DE genes,  $N_{de}$ , just by chance.

The second term in Eq. 6.1 is a functional term that depends on *the identities of the specific genes* that are differentially expressed as well as on the interactions described by the pathway (i.e., its topology). In essence, this term sums up the absolute values of the *perturbation factors* (PF) for all genes  $g$  on the given pathway  $P_i$ . The perturbation factor of a gene  $g$  is calculated as follows:

$$PF(g) = \Delta E(g) + \sum_{u \in US_g} \beta_{ug} \cdot \frac{PF(u)}{N_{ds}(u)} \quad (6.2)$$

In this expression, the first term captures the quantitative information measured in the gene expression experiment. The factor  $\Delta E(g)$  represents the signed normalized measured expression change of the gene  $g$  determined using one of the available methods [32, 47, 149, 207]. The second term is a sum of all perturbation factors of the genes  $u$  directly upstream of the target gene  $g$ , normalized by the number of downstream genes of each such gene  $N_{ds}(u)$ , and weighted by a factor  $\beta_{ug}$ , which reflects the type of interaction:  $\beta_{ug} = 1$  for induction,  $\beta_{ug} = -1$  for repression.<sup>1</sup>  $US_g$  is the set of all such genes upstream of  $g$ . The second term here is similar to the PageRank used by Google [138] only that we weight the downstream instead of the upstream connections (a web page is important if other pages point to it whereas a gene is important if it influences other genes). If there are no measured differences in the expression values of any of the genes upstream of  $g$ ,  $PF(u) = 0$  for all genes in  $US_g$ ,

<sup>1</sup> In KEGG, which is the source of the pathways used here, this information about the type of interaction is available for every link between two genes in the description of the pathway topology.

and the second term becomes zero. In this case the perturbation factor reduces to:

$$PF(g) = \Delta E \quad (6.3)$$

This is exactly the classical approach, in which the amount of perturbation of an individual gene in a given condition is measured through its expression change  $\Delta E$ .

Under the null hypothesis, which assumes that the list of DE genes only contains random genes, the likelihood that a pathway has a large impact factor is proportional to the number of such “differentially expressed” genes that fall on the pathway, which in turn is proportional to the pathway size. Thus, the second term in Eq. 6.1 is normalized with respect to the pathway size by dividing the total perturbation by the number of DE genes on the given pathway,  $N_{de}(P_i)$ . Furthermore, various technologies can yield systematically different estimates of the fold changes. For instance, the fold changes reported by microarrays tend to be compressed with respect to those reported by RT-PCR [23, 49]. In order to make the impact factors as independent as possible from the technology, and also comparable between problems, we also divide the second term in Eq. 6.1 by the mean absolute fold change  $|\overline{\Delta E}|$ , calculated across all DE genes. Assuming that there are at least some DE genes anywhere in the data set<sup>2</sup>, both  $|\overline{\Delta E}|$  and  $N_{de}(P_i)$  are different from zero.

Our original implementation, as part of Pathway Express from the Onto-Tools suite <sup>3</sup>, used an iterative algorithm to determine the perturbation factor of each gene in the give pathway. Fig. 6.1 illustrates the computation and propagation of the perturbations over two steps in a small area of the actin cytoskeleton pathway as impacted in breast cancer (shown in its entirety in Fig. 6.2). As already mentioned, in all data shown here the regulatory efficiency is  $\beta = 1$  for all genes. In this case, the gene *DIAPH3* is the input gene with an observed fold change  $\Delta E = 1.4841$ . Since there are no genes upstream of *DIAPH3*, its second term in Eq. 6.2 is zero. Using Eq. 6.2, the PF of gene *DIAPH3* is simply its measured

<sup>2</sup>If there are no DE genes anywhere, the problem of finding the impact on various pathways is meaningless.

<sup>3</sup>The Onto-Tools suite is available at <http://vortex.cs.wayne.edu>



expression change:

$$PF(DIAPH3) = 1.4841 + 0 = 1.4841 \quad (6.4)$$

The next step involves the computation of the perturbation for *BAIAP2*. This gene receives signals from *DIAPH3* but also from *RAC1*, *RAC1P4*, *RAC2* and *RAC3*. Using Eq. 6.2, the PF for the gene *BAIAP2* can be calculated as:

$$\begin{aligned} PF(BAIAP2) = & \Delta E(BAIAP2) + \frac{PF(DIAPH3)}{N_{ds}(DIAPH3)} + \frac{PF(RAC1)}{N_{ds}(RAC1)} + \frac{PF(RAC1P4)}{N_{ds}(RAC1P4)} \\ & + \frac{PF(RAC2)}{N_{ds}(RAC2)} + \frac{PF(RAC3)}{N_{ds}(RAC3)} \end{aligned}$$

The previously calculated perturbations for *RAC1*, *RAC1P4*, *RAC2* and *RAC3* are:

$$PF(RAC1) = PF(RAC1P4) = PF(RAC2) = PF(RAC3) = -0.251 \quad (6.5)$$

Each of these genes signals only to *BAIAP2* so for each of them the number of downstream genes,  $N_{ds}$  will be equal to 1. Hence, the PF for the gene *BAIAP2* can be calculated as:

$$PF(BAIAP2) = 1.4841 + \frac{-0.251}{1} + \frac{-0.251}{1} + \frac{-0.251}{1} + \frac{-0.251}{1} = 0.4801 \quad (6.6)$$

Similarly, using Eq. 6.2, the PF for the gene *DIAPH1* is

$$PF(DIAPH1) = \Delta E(DIAPH1) + \frac{PF(BAIAP2)}{N_{ds}(BAIAP2)} = 0 + \frac{0.4801}{3} = 0.16 \quad (6.7)$$

The perturbation of the other two genes, *LOC286404* and *WASF2* is analogous and yields the same numerical value.

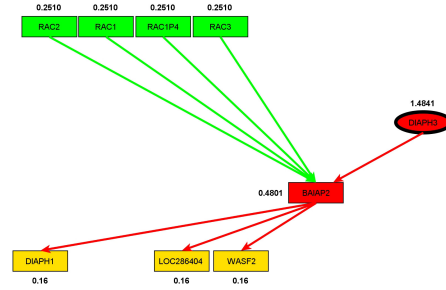


Figure 6.1: The computation of the PF for a gene and the subsequent propagation of the perturbation according to Eq. 6.2. The genes are part of regulation of actin cytoskeleton pathway shown in Fig. 6.2. Some of the interactions between the genes have been removed in order to simplify the figure. The labels next to each gene indicate the PF. *Source:* Supplementary materials of [52].

## 6.2 Significance of the impact factor

The impact factor, as computed in Eq. 6.1, is not a probability and is not suitable for comparison between pathways. This section shows that the impact factor corresponds to the negative log of the global probability of having both a statistically significant number of differentially expressed genes and a large perturbation in the given pathway [52].

The first type of evidence that is used in the impact analysis is the probability that the number of differentially expressed genes,  $X$ , is larger than or equal to the observed number of differentially expressed genes,  $N_{de}$  just by chance:

$$p_i = P(X \geq N_{de} | H_0) \quad (6.8)$$

The second type of evidence is the impact of the topology, the gene interactions, and gene fold changes that are captured in the second term of Eq. 6.1 through the pathway perturbation factor:

$$PF = \frac{\sum_{g \in P_i} |PF(g)|}{|\Delta E| \cdot N_{de}(P_i)} \quad (6.9)$$

Let  $PF$  denote the perturbation factor as a random variable and  $pf$  be the observed value for a particular pathway. The score  $pf$  is always positive, and the higher its value, the less likely the null hypothesis (that the pathway is not significant). Moreover this likelihood de-

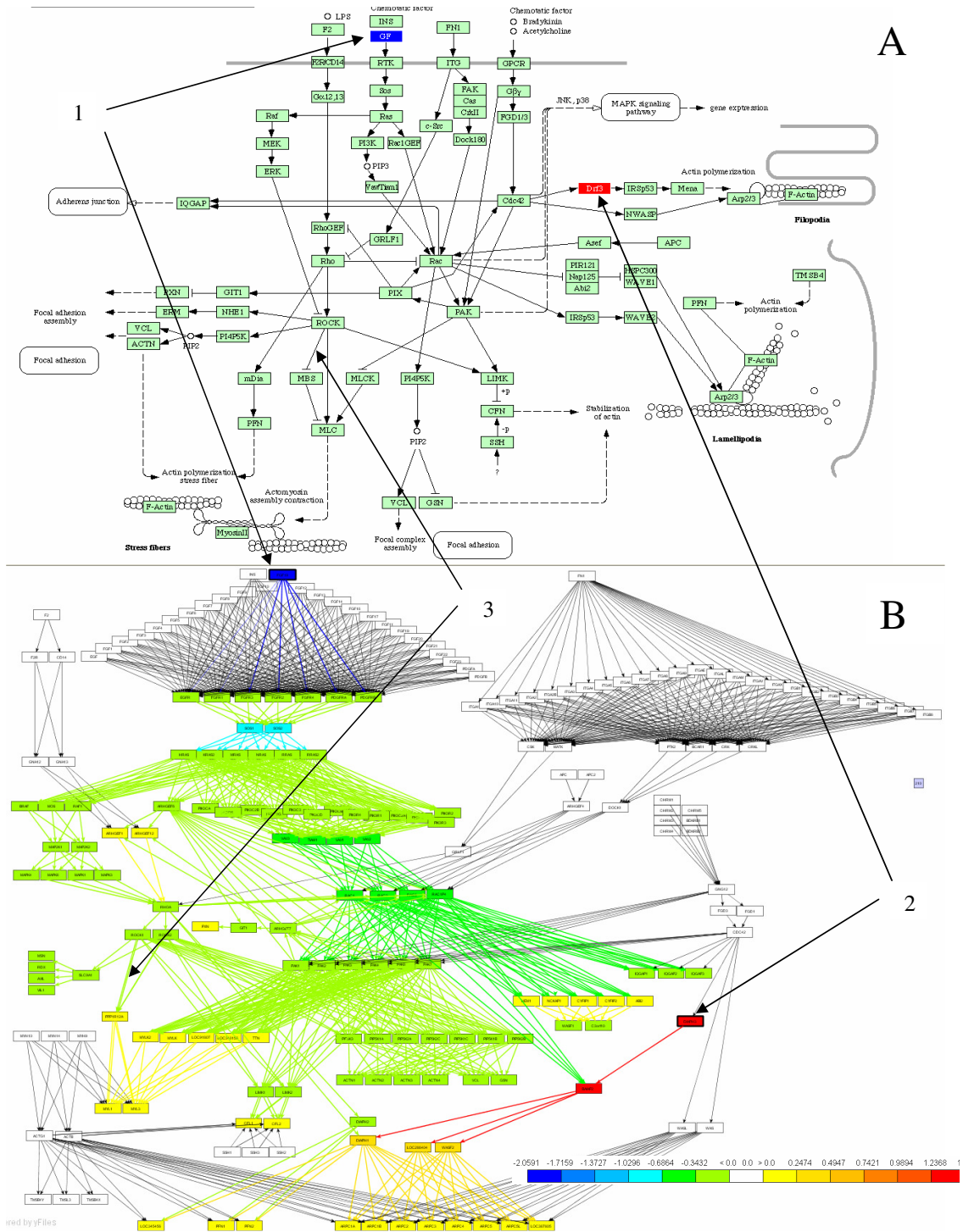


Figure 6.2: Regulation of actin cytoskeleton as impacted in breast cancer [190]: the KEGG pathway diagram (A) and its internal graph representation (B). Note that the unique symbol GF (blue) in the KEGG diagram A, actually stands for 25 FGF genes in the internal graph B, only one of which is differentially expressed (1). The colors show the propagation of the gene perturbations throughout the pathway. The differentially expressed genes are FGF18 (1) and DIAPH3 (2). Changes from blue/green to yellow/red and viceversa correspond to inhibitory interactions. For instance, since ROCK inhibits MBS, the negative perturbation of ROCK propagates as a positive perturbation of MBS (3). *Source:* Supplementary materials of [52].

cays very fast as  $pf$  gets away from zero. These features point to the exponential distribution as an appropriate model for the random variable  $PF$ .

Under the null hypothesis, differentially expressed genes would fall on the pathway randomly, and would not interact with each other in any concerted way. In other words, in the second term in Eq. 6.2 (which captures the influence of the genes upstream) roughly half of those influences will be positives, and half negative, canceling each other out. In such circumstances, the perturbation of each gene would be limited to its own measured fold change (due to random unrelated causes):

$$PF(g) = \Delta E(g) + \sum_{u \in US_g} \beta_{ug} \cdot \frac{PF(u)}{N_{ds}(u)} = \Delta E(g) + 0 = \Delta E(g) \quad (6.10)$$

Consequently, under the same null hypothesis, the expected value for the perturbation of a pathway (from Eq. 6.9) will be:

$$E(PF) = E\left(\frac{\sum_{g \in P_i} |PF(g)|}{|\Delta E| \cdot N_{de}(P_i)}\right) = E\left(\frac{1}{|\Delta E|} \frac{\sum_{k=1}^{N_{de}(P_i)} |\Delta E(g)|}{N_{de}(P_i)}\right) = E\left(\frac{|\overline{\Delta E}_{P_i}|}{|\Delta E|}\right) = 1 \quad (6.11)$$

The last fraction above is the ratio between the mean fold change on the given pathway,  $P_i$ , and the mean fold change in the entire data set. Under the null hypothesis, the genes are distributed randomly across pathways and the two means should be equal. Since this expected value is 1, the distribution of the random variable  $PF$  can be modeled by the exponential of mean 1,  $exp(1)$ .

If we use the  $PF$  score as a test statistics and assume its null distribution is exponential with mean 1, then the p-value  $p_{pf}$  resulting from the perturbation analysis will have the form:

$$p_{pf} = P(PF \geq pf | H_0) = e^{-pf} \quad (6.12)$$

This is the probability of observing a perturbation factor,  $PF$ , greater or equal to the one observed,  $pf$ , when the null hypothesis is true.

Let us now consider that for a given pathway we observe a perturbation factor equal to  $pf$  and a number of differentially expressed genes equal to  $N_{de}$ . A ‘global’ probability  $p_{global}$ , of having just by chance both a higher than expected number of differentially expressed genes AND a significant biological perturbation (large  $PF$  in the second term), can be defined as the joint probability:

$$p_{global} = P(X \geq N_{de}, PF \geq pf | H_0) \quad (6.13)$$

Since the pathway perturbation factor in Eq. (6.9) is calculated by dividing the total pathway perturbation by the number of differentially expressed genes on the given pathway, the  $PF$  will be independent of the number of differentially expressed genes  $X$ , and the joint probability above becomes a product of two single probabilities:

$$p_{global} = P(X \geq N_{de} | H_0) \cdot P(PF \geq pf | H_0) \quad (6.14)$$

This  $p_{global}$  provides a global significance measure that requires both a statistically significant number of differentially expressed genes on the pathway,  $N_{de}$ , and at the same time, large perturbations on the same pathway as described by  $pf$ . Using equations (6.8) and (6.12), the formula (6.14) becomes:

$$p_{global} = p_i \cdot e^{-pf} \quad (6.15)$$

We take a natural log of both sides and obtain:

$$\log(p_{global}) = \log(p_i) - pf \quad (6.16)$$

which can be re-written as:

$$-\log(p_{global}) = -\log(p_i) + pf \quad (6.17)$$

in which we can substitute the definition of  $pf$  from (6.9) above to yield:

$$-\log(p_{global}) = -\log(p_i) + PF \quad (6.18)$$

The right hand side of this expression is exactly our definition of the impact factor:

$$IF = -\log(p_i) + \frac{\sum_{g \in P_i} |PF(g)|}{|\Delta E| \cdot N_{de}(P_i)} \quad (6.19)$$

This shows that the proposed impact factor, IF, is in fact the negative log of the global probability of having both a statistically significant number of differentially expressed genes and a large perturbation in the given pathway.

Ignoring the discrete character of the hypergeometric distribution, under the null hypothesis  $p_i = P(X \geq N_{RP}|H_0)$  has a uniform distribution. By taking negative log, the distribution changes into exponential with parameter 1, similar to the distribution we assumed for PF, the second term in IF formula.

$$-\log(p_i) \sim \exp(1); \quad PF \sim \exp(1); \quad \exp(1) = \Gamma(1, 1) \quad (6.20)$$

Then, as the sum of two independent exponential random terms, the IF will follow a Gamma distribution  $\Gamma(2, 1)$  [87]. The pdf of this distribution is:

$$f(x) = xe^{-x}, \quad x \geq 0 \quad (6.21)$$

Finally, the p-value corresponding to the observed value  $if$  of the statistic  $IF$  can be easily computed by integrating the density (6.21):

$$p = P(IF \geq if|H_0) = \int_{if}^{\infty} f(x)dx = \int_{if}^{\infty} xe^{-x}dx = (if + 1) * e^{-if} \quad (6.22)$$

### 6.3 Performance analysis of the impact factor

We will illustrate this novel pathway analysis on several datasets. The first set includes genes associated with better survival in lung adenocarcinoma [10]. These genes have the potential to represent an important tool for the therapeutical decision and, if the correct regulatory mechanisms are identified, they could also be potential drug targets. The expression values of the 97 genes associated with better survival identified by Beer et. al. were compared between the cancer and healthy groups. These data were then analyzed using a classical ORA approach (hypergeometric), a classical FCS approach (GSEA), and our impact analysis. Fig. 6.3 shows a comparison between the results of these methods.

From a statistical perspective, the power of both classical techniques appears to be very limited. The corrected p-values do not yield any pathways independently of the type of correction. If the significance levels were to be ignored and the techniques used only to rank the pathways, the results would still be unsatisfactory. According to ORA, the most significantly affected pathways in this data set are *prion disease*, *focal adhesion* and *Parkinson's disease*. In reality, both prion and Parkinson's diseases are pathways specifically associated to diseases of the central nervous system and are unlikely to be related to lung adenocarcinomas. In this case, *prion disease* ranks at the top only due to one gene, *LAMB1*. Since this pathway is rather small (14 genes), every time any one gene is differentially expressed, the hypergeometric analysis will rank it highly. A similar phenomenon happens on *Parkinson's disease*, indicating that this is a problem associated with the method rather than with a specific pathway. At the same time, pathways highly relevant to cancer such as *cell cycle* and *Wnt signaling* are ranked in the lower half of the list. The most significant pathways reported as enriched in cancer by GSEA [172] are: *cell cycle*, *Huntington disease*, *DRPLA*, *Alzheimer's* and *Parkinson's* (see Fig. 6.3). Among these, only *cell cycle* is relevant, while *Huntington's*, *Alzheimer's* and *Parkinson's* are clearly incorrect. However, although ranked first, *cell cycle* is not significant in GSEA, even at the most lenient 10% significance and

Pathway name	ORA (hypergeometric)		
	p-value	FDR	Bonferroni
Prion disease	0.149649	0.627132	1
Focal adhesion	0.155424	0.627132	1
Parkinson's disease	0.164842	0.627132	1
Dentatorubropallidoluysian atrophy	0.179767	0.627132	1
Calcium signaling pathway	0.262884	0.627132	1
Alzheimer's disease	0.277100	0.627132	1
Apoptosis	0.283744	0.627132	1
TGF-beta signaling pathway	0.303663	0.627132	1
Huntington's disease	0.327491	0.627132	1
Toll-like receptor signaling pathway	0.330069	0.627132	1
Wnt signaling pathway	0.369145	0.637613	1
Regulation of actin cytoskeleton	0.439390	0.695701	1
MAPK signaling pathway	0.560814	0.762988	1
Phosphatidylinositol signaling system	0.572396	0.762988	1
Adherens junction	0.602359	0.762988	1
Complement and coagulation cascades	0.680333	0.766820	1
Cell cycle	0.686102	0.766820	1
Cytokine-cytokine receptor interaction	0.820650	0.866242	1
Neuroactive ligand-receptor interaction	0.972996	0.972996	1

Enriched in cancer			
Pathway Name	NOM p-val	FDR q-val	FWER p-val
Cell cycle	0.038	0.118	0.140
Huntington's disease	0.074	0.217	0.546
Dentatorubropallidoluysian atrophy (DRPLA)	0.149	0.291	0.751
Alzheimer's disease	0.189	0.344	0.877
Parkinson's disease	0.373	0.485	0.984
Adherens junction	0.583	0.651	0.998
Wnt signaling pathway	0.861	0.785	1

Enriched in normal			
Pathway Name	NOM p-val	FDR q-val	FWER p-val
MAPK signaling pathway	0.007	0.170	0.361
Apoptosis	0.019	0.175	0.304
Complement and coagulation cascades	0.037	0.255	0.298
Phosphatidylinositol signaling system	0.189	0.343	0.823
Regulation of actin cytoskeleton	0.010	0.356	0.223
Focal adhesion	0.160	0.384	0.817
Cytokine-cytokine receptor interaction	0.241	0.420	0.910
Toll-like receptor signaling pathway	0.330	0.451	0.963
Calcium signaling pathway	0.308	0.489	0.960
Prion disease	0.474	0.563	0.986
TGF-beta signaling pathway	0.631	0.699	0.998
Neuroactive ligand-receptor interaction	0.947	0.957	1

Pathway name	Impact Factor			
	IF	p-value	FDR	Bonferroni
Cell cycle	19.26	8.76E-08	1.66E-06	1.66E-006
Focal adhesion	7.414	0.005072	0.048180	0.0956831
Wnt signaling pathway	6.780	0.008840	0.055988	0.1679642
Dentatorubropallidoluysian atrophy	5.535	0.025788	0.122495	0.4899810
Huntington's disease	4.543	0.058985	0.203925	1
Apoptosis	4.407	0.065921	0.203925	1
Regulation of actin cytoskeleton	4.246	0.075130	0.203925	1
TGF-beta signaling pathway	3.511	0.134730	0.319984	1
Complement and coagulation cascades	3.161	0.176357	0.354145	1
Adherens junction	2.953	0.206279	0.354145	1
Alzheimer's disease	2.752	0.239378	0.354145	1
Parkinson's disease	2.631	0.261455	0.354145	1
Toll-like receptor signaling pathway	2.576	0.272054	0.354145	1
Prion disease	2.572	0.272839	0.354145	1
Calcium signaling pathway	2.538	0.279588	0.354145	1
Cytokine-cytokine receptor interaction	2.353	0.318815	0.366952	1
Phosphatidylinositol signaling system	2.311	0.328326	0.366952	1
MAPK signaling pathway	2.205	0.353353	0.372984	1
Neuroactive ligand-receptor interaction	0.576	0.885936	0.885936	1

Figure 6.3: A comparison between the results of the classical probabilistic approaches (A - hypergeometric, B - GSEA) and the results of the pathway impact analysis (C) for a set of genes differentially expressed in lung adenocarcinoma. The pathways marked with green are considered most likely to be linked to this condition in this experiment. The ones in red are unlikely to be related. The ranking of the pathways produced by the classical approaches is very misleading. According to the hypergeometric model, the most significant pathways in this condition are: *prion disease*, *focal adhesion*, and *Parkinson's disease*. Two out of these 3 are likely to be incorrect. GSEA yields *cell cycle* as the most enriched pathway in cancer but 3 out of the 4 subsequent pathways are clearly incorrect. In contrast, all 3 top pathways identified by the impact analysis are relevant to the given condition. The impact analysis is also superior from a statistical perspective. According to both hypergeometric and GSEA, no pathway is significant at the usual 1% or 5% levels on corrected p-values. In contrast, according to the impact analysis the *cell cycle* is significant at 1%, and *focal adhesion* and *Wnt signaling* are significant at 5% and 10%, respectively. Source: [52].



with the least conservative correction.

In contrast, the impact analysis reports *cell cycle* as the most perturbed pathway in this condition and also as highly significant from a statistical perspective ( $p = 1.6 \cdot 10^{-6}$ ). Since early papers on lung cancers [168, 133], until the most recent ones [141, 35], there is a consensus that the cell cycle is highly deranged in lung cancers. Moreover, cell cycle genes have started to be considered both as potential prognostic factors and therapeutic targets [194]. The second most significant pathway as reported by the impact analysis is *focal adhesion*. An inspection of this pathway (Fig. 6.4) shows that in these data, both *ITG* and *RTK* receptors are perturbed, as well as the *EGF/GF* ligand. Because these 3 genes appear at the very beginning and affect both entry points controlling this pathway, their perturbations are widely propagated throughout the pathway. Furthermore, the *CRK* oncogene was also found to be up-regulated. Increased levels of *CRK* proteins have been observed in several human cancers and over-expression of *CRK* in epithelial cell cultures promotes enhanced cell dispersal and invasion [156]. For this pathway, the impact analysis yields a raw p-value of 0.005, which remains significant even after the FDR correction ( $p = 0.048$ ), at the 5% level. In contrast, the hypergeometric model yields a raw p-value of 0.155 (FDR corrected to 0.627) while GSEA yields a raw p-value of 0.16 (FDR corrected to 0.384). For both techniques, not even the raw p-values are significant at the usual levels of 5% or 10%. This is not a mere accident but an illustration of the intrinsic limitations of the classical approaches. These approaches completely ignore the position of the genes on the given pathways, and therefore they are not able to identify this pathway as being highly impacted in this condition. Note that any ORA approach will yield the same results for this pathway for any set of 4 differentially expressed genes from the set of genes on this pathways. Similarly, GSEA will yield the same results for any other set of 5 genes with similar expression values (yielding similar correlations with the phenotype). Both techniques are unable to distinguish between a situation in which these genes are upstream, potentially commandeering the entire pathway as in this example, or randomly distributed throughout

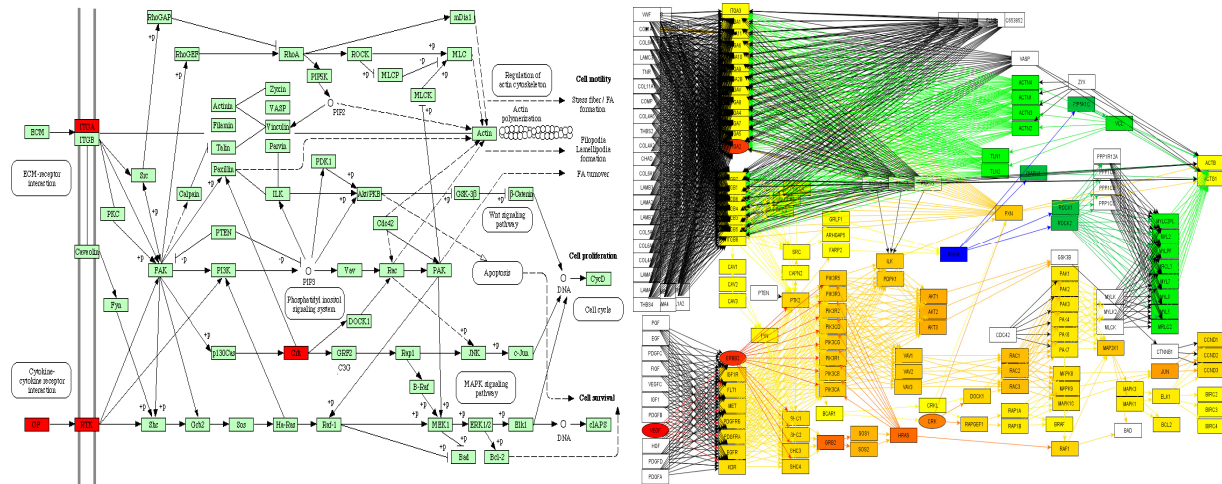


Figure 6.4: The focal adhesion in lung adenocarcinoma (left-KEGG diagram, right-internal representation). Both *ITG* and *RTK* receptors are perturbed, as well as the *VEGF/GF* ligand. Since these 3 genes appear at the very beginning and affect both entry points to this pathway, their perturbations are widely propagated throughout (right), and this pathway appears as highly impacted. All classical approaches ignore the positions of the genes, and fail to identify this pathway as significant.

the pathway.

The third pathway as ranked by the impact analysis is *Wnt signaling* (FDR corrected  $p=0.055$ , significant at 10%). In fact, Mazieres et al. [128] recently identified 3 different mechanisms for the activation of the Wnt signaling in lung cancers and argued that the blockade of Wnt pathway may lead to new treatment strategies. In the same data set, *Huntington's*, *Parkinson's*, *prion* and *Alzheimer's* diseases have low impact factors (corrected  $p$ -values of above 0.20), correctly indicating that they are unlikely to be relevant in lung adenocarcinomas.

A second data set includes genes identified as being associated with poor prognosis in breast cancer [190]. Fig. 6.5 shows a comparison between the classical hypergeometric approach, GSEA, and the pathway impact analysis. On this data, GSEA finds no significantly impacted pathways at any of the usual 5% or 10% levels. In fact, the only FDR-corrected value below 0.25, in the entire data set is 0.11, corresponding to the *ubiquitin mediated proteolysis*. Furthermore, GSEA's ranking does not appear to be useful for this data, with none of the cancer-related pathways being ranked towards the top. The most significant signalling pathway according to the hypergeometric analysis, *cell cycle* is also the most significant in

the impact analysis. However, the agreement between the two approaches stops here. In terms of statistical power, according to the classical hypergeometric model, there are no other significant pathways at either 5% or 10% significance on the corrected p-values. If we were to ignore the usual significance thresholds and only consider the ranking, the third highest pathway according to the hypergeometric model is *Parkinson's disease*. In fact, based on current knowledge, Parkinson's disease is unlikely to be related to rapid metastasis in breast cancer. At the same time, the impact analysis finds several other pathways as significant. For instance, *focal adhesion* is significant with an FDR-corrected p-value of 0.03. In fact, a link between focal adhesion and breast cancer has been previously established [79, 188]. In particular, *FAK*, a central gene on the focal adhesion pathway, has been found to contribute to cellular adhesion and survival pathways in breast cancer cells which are not required for survival in non-malignant breast epithelial cell [13]. Recently, it has also been shown that Doxorubicin, an anti-cancer drug, caused the formation of well defined focal adhesions and stress fibers in mammary adenocarcinoma MTLn3 cells early after treatment [187]. Consequently, the *FAK/PI-3 kinase/PKB* signaling route within the focal adhesion pathway has been recently proposed as the mechanism through which Doxorubicin triggers the onset of apoptosis [187].

*TGF-beta signaling* ( $p=0.032$ ) and *MAPK* ( $p=0.064$ ) are also significant. Both fit well with previous research results. *TGF-beta1*, the main ligand for the *TGF-beta signaling* pathway, is known as a marker of invasiveness and metastatic capacity of breast cancer cells [183]. In fact, it has been suggested as the missing link in the interplay between estrogen receptors and *HER-2* (human epidermal growth factor receptor 2) [183]. Furthermore, plasma levels of *TGF-beta1* have been used to identify low-risk postmenopausal metastatic breast cancer patients [136]. Finally, *MAPK* has been shown to be connected not only to cancer in general, but to this particular type of cancer. For instance the proliferative response to progestin and estrogen was shown to be inhibited in mammary cells microinjected with inhibitors of MAP kinase pathway [30]. Also, it is worth noting the gap between the p-values for *regulation*

of *actin cytoskeleton* ( $p=0.111$ ), which may be relevant in cancer, and the next pathway, *Parkinson's disease* ( $p=0.239$ ), which is irrelevant in this condition.

A third data set involves a set of differentially expressed genes obtained by studying the response of a hepatic cell line when treated with palmitate [174]. Fig. 6.7 shows the comparison between the classical statistical analysis (ORA) and the pathway impact analysis<sup>4</sup>. The classical statistical analysis yields 3 pathways significant at the 5% level: *complement and coagulation cascades*, *focal adhesion* and *MAPK*. The impact analysis agrees on all these, but also identifies several additional pathways. The top 4 pathways identified by the impact analysis are well supported by the existing literature. There are several studies that support the existence of a relationship between different coagulation factors, present in the *complement and coagulation cascades* pathway, and palmitate. Sanders et al., for instance, demonstrated that a high palmitate intake affects factor VII coagulant (*FVIIc*) activity [159]. Interestingly, Fig. 6.6 shows not only that this pathway has a higher than expected proportion of differentially expressed genes, but also that 6 out of 7 such genes are involved in the same region of the pathway, suggesting a coherently propagated perturbation. The focal adhesion and tight junction pathways involve cytoskeletal genes. Swagell et al. [174] considered the presence of the cytoskeletal genes among the differentially expressed genes as very interesting and hypothesized that the down-regulation of these cytoskeletal genes indicates that palmitate decreases cell growth. Finally, the link between *MAPK* and the palmitate was established by Susztak et al. who showed that *p38* MAP kinase is a key player in the palmitate-induced apoptosis [173].

---

<sup>4</sup>The GSEA analysis requires expression values for all genes. Since this experiment was performed with a custom array and not all values are publicly available, GSEA could not be applied here.

**A**

Pathway name	ORA (hypergeometric)		
	p-value	FDR	bonferroni
Cell cycle	3.1E-07	2.8E-06	2.765E-06
MAPK signaling pathway	0.02513	0.11309	0.2261834
Parkinson's disease	0.10752	0.32255	0.9676532
Cytokine-cytokine receptor interaction	0.24736	0.47992	1
Focal adhesion	0.29628	0.47992	1
Calcium signaling pathway	0.37158	0.47992	1
Regulation of actin cytoskeleton	0.40691	0.47992	1
TGF-beta signaling pathway	0.42660	0.47992	1
Neuroactive ligand-receptor interaction	0.58749	0.58749	1

**B**

**Enriched in poor prognosis**

Pathway Name	NOM p-val	FDR q-val	FWER p-val
Ubiquitin mediated proteolysis	0.031	0.113	0.111
Prion disease	0.352	0.570	0.802
Alzheimer's disease	0.279	0.625	0.683
Tight junction	0.848	0.749	0.974
Parkinson's disease	0.638	0.795	0.958

**Enriched in good prognosis**

Pathway Name	NOM p-val	FDR q-val	FWER p-val
Notch signaling pathway	0.082	0.277	0.636
Neuroactive ligand-receptor interaction	0.050	0.280	0.542
Adherens junction	0.136	0.400	0.829
Wnt signaling pathway	0.058	0.410	0.534
Circadian rhythm	0.078	0.582	0.960
Complement and coagulation cascades	0.232	0.638	0.997
Apoptosis	0.212	0.691	0.996
MAPK signaling pathway	0.046	0.693	0.479
Amyotrophic lateral sclerosis	0.244	0.738	0.993
Jak-STAT signaling pathway	0.913	0.952	1
Dentatorubropallidoluysian atrophy	0.792	0.987	1
Cytokine-cytokine receptor interaction	0.913	0.987	1
Calcium signaling pathway	0.522	1	1
Focal adhesion	0.556	1	1
Regulation of actin cytoskeleton	0.575	1	1
Phosphatidylinositol signaling system	0.735	1	1
TGF-beta signaling pathway	0.815	1	1
Cell cycle	0.859	1	1
Huntington's disease	0.885	1	1

**C**

Pathway name	Impact Factor			
	IF	p-value	FDR	Bonferroni
Cell cycle	18.8	1.3E-07	1.2E-06	1.19E-006
Focal adhesion	7.06	0.00692	0.03112	0.0622412
TGF-beta signaling pathway	6.56	0.01075	0.03225	0.0967557
MAPK signaling pathway	5.40	0.02886	0.06493	0.2597164
Regulation of actin cytoskeleton	4.49	0.06180	0.11125	0.5562285
Parkinson's disease	3.12	0.18207	0.23946	1
Cytokine-cytokine receptor interaction	3.09	0.18624	0.23946	1
Neuroactive ligand-receptor interaction	2.87	0.21942	0.24685	1
Calcium signaling pathway	2.44	0.30047	0.30047	1

Figure 6.5: A comparison between the results of the classical (ORA) probabilistic approach (A), GSEA (B) and the results of the pathway impact analysis (C) for a set of genes associated with poor prognosis in breast cancer. The pathways marked with green are well supported by the existing literature. The ones in red are unlikely to be related. After correcting for multiple comparisons, GSEA fails to identify any pathway as significantly impacted in this condition at any of the usual significance levels (1%, 5% or 10%). The hypergeometric model pinpoints *cell cycle* as the only significant pathway. Relevant pathways such as *focal adhesion*, *TGF-beta signaling*, and *MAPK* do not appear as significant from a hypergeometric point of view. While agreeing on the *cell cycle*, the impact analysis also identifies the 3 other relevant pathways as significant at the 5% level. Source: [52].

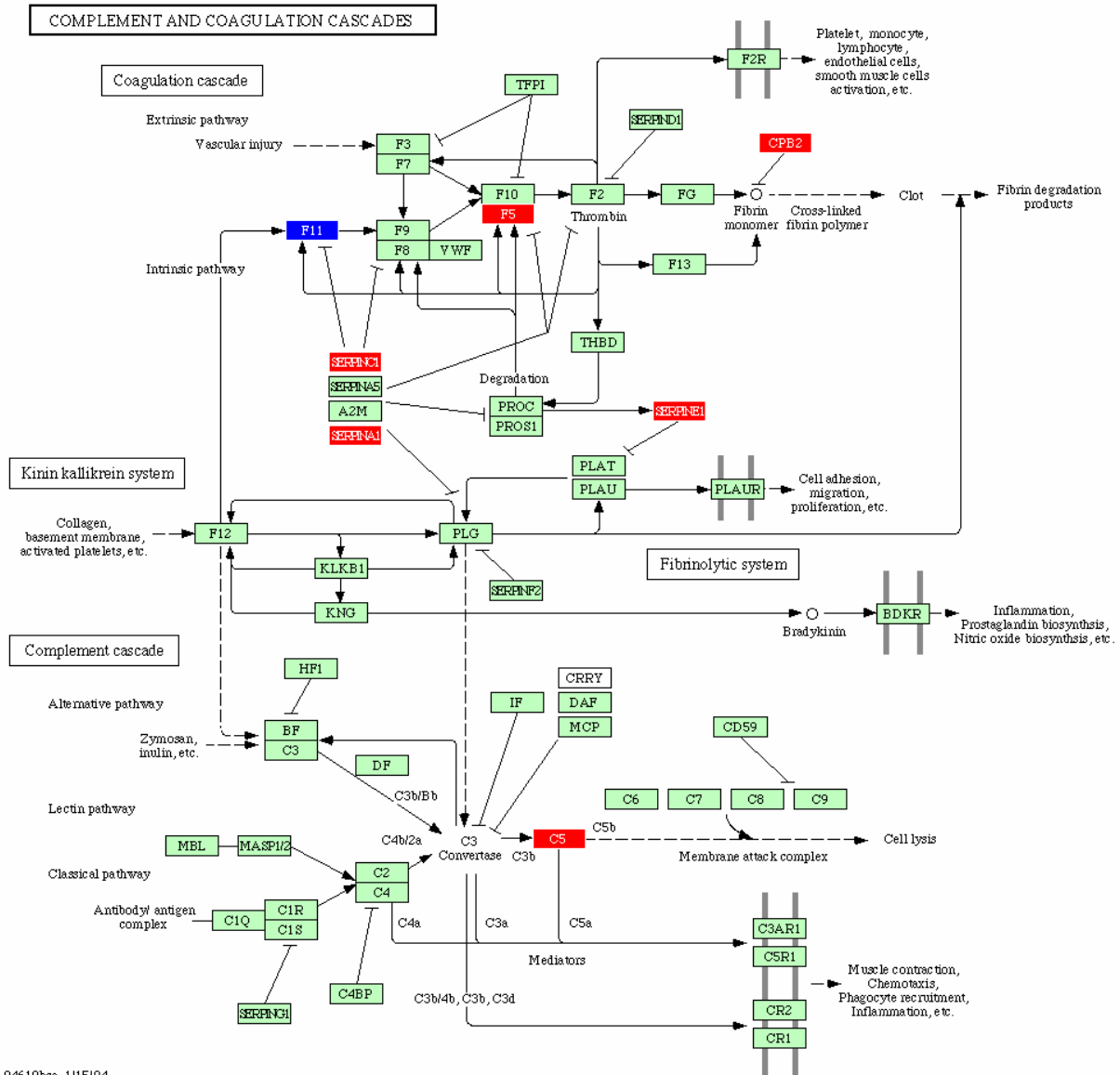


Figure 6.6: The complement and coagulation cascade as affected by treatment with palmitate in a hepatic cell line. There are 7 differentially expressed genes (up-regulated in red, down-regulated in blue) out of 69 total genes. All classical ORA models would give any other pathway with the same proportion of genes a similar p-value, disregarding the fact that 6 out of these 7 genes are involved in the same region of the pathway, closely interacting with each other. Both ORA and GSEA would yield exactly the same significance value to this pathway even if the diagram were to be completely re-designed by future discoveries. In contrast, the impact factor can distinguish between this pathway and any other pathway with the same proportion of differentially expressed gene, as well as take into account any future changes to the topology of the pathway. *Source:* [52].

**A**

Pathway name	ORA (hypergeometric)		
	p-value	FDR	Bonferroni
Complement and coagulation cascades	1.26958E-07	2.28525E-06	2.28525E-06
Focal adhesion	4.03691E-05	0.000363322	0.000726643
MAPK signaling pathway	0.000523961	0.003143765	0.009431295
TGF-beta signaling pathway	0.011698758	0.052644412	0.210577648
Toll-like receptor signaling pathway	0.018714569	0.067372448	0.336862241
Calcium signaling pathway	0.024575814	0.072600598	0.442364654
Tight junction	0.028233566	0.072600598	0.508204185
Wnt signaling pathway	0.050174237	0.100857467	0.903136270
Phosphatidylinositol signaling system	0.058285692	0.100857467	1
Prion disease	0.060516063	0.100857467	1
Jak-STAT signaling pathway	0.061635119	0.100857467	1
Apoptosis	0.106427143	0.146873866	1
Cell cycle	0.106427143	0.146873866	1
Regulation of actin cytoskeleton	0.115415266	0.146873866	1
Alzheimer's disease	0.122394888	0.146873866	1
Huntington's disease	0.146968097	0.165339109	1
Neuroactive ligand-receptor interaction	0.233787848	0.247540075	1
Cytokine-cytokine receptor interaction	0.429908167	0.429908167	1

**B**

Pathway name	Impact Factor			
	IF	p-value	FDR	Bonferroni
Complement and coagulation cascades	19.374	7.85335E-08	1.41360E-06	1.44761E-06
Focal adhesion	13.791	1.51580E-05	1.36422E-04	3.01180E-04
MAPK signaling pathway	9.475	8.03922E-04	0.004823531	0.014470593
Tight junction	7.128	0.006521277	0.029345745	0.117382981
TGF-beta signaling pathway	6.868	0.008187095	0.029473543	0.147367717
Toll-like receptor signaling pathway	6.391	0.012391594	0.037174781	0.223048688
Calcium signaling pathway	5.774	0.021048873	0.052496861	0.378879719
Apoptosis	5.653	0.023331938	0.052496861	0.419974887
Regulation of actin cytoskeleton	5.225	0.033492741	0.066985482	0.602869334
Jak-STAT signaling pathway	4.983	0.041004319	0.073807774	0.738077735
Wnt signaling pathway	4.313	0.071158653	0.116441431	1
Phosphatidylinositol signaling system	3.975	0.093427025	0.133344438	1
Prion disease	3.937	0.096304316	0.133344438	1
Huntington's disease	3.839	0.104111596	0.133857767	1
Alzheimer's disease	3.387	0.148324272	0.171694058	1
Cell cycle	3.350	0.152616940	0.171694058	1
Neuroactive ligand-receptor interaction	2.414	0.305405348	0.323370368	1
Cytokine-cytokine receptor interaction	2.208	0.352624224	0.352624224	1

Figure 6.7: A comparison between the results of the classical probabilistic approach (A) and the results of the impact analysis (B) for a set of genes found to be differentially expressed in a hepatic cell line treated with palmitate. Green pathways are well supported by literature evidence while red pathways are unlikely to be relevant. The classical statistical analysis yields 3 pathways significant at the 5% level: *complement and coagulation cascades*, *focal adhesion* and *MAPK*. The impact analysis agrees on these 3 pathways but also identifies several additional pathways. Among these, *tight junction* is well supported by the literature. Source: [52].



## 6.4 System of equations and alternative significance computation

As an additional effort in our group, it has been shown that the the set of all perturbations defined by Eq. 6.2 form a system of linear equations [112]:

$$\begin{pmatrix} 1 - \frac{\beta_{11}}{N_{ds}(g_1)} & -\frac{\beta_{21}}{N_{ds}(g_2)} & \cdots & -\frac{\beta_{n1}}{N_{ds}(g_n)} \\ -\frac{\beta_{12}}{N_{ds}(g_1)} & 1 - \frac{\beta_{22}}{N_{ds}(g_2)} & \cdots & -\frac{\beta_{n2}}{N_{ds}(g_n)} \\ \cdots & \cdots & \cdots & \cdots \\ -\frac{\beta_{1n}}{N_{ds}(g_1)} & -\frac{\beta_{2n}}{N_{ds}(g_2)} & \cdots & 1 - \frac{\beta_{nn}}{N_{ds}(g_n)} \end{pmatrix} \begin{pmatrix} PF(g_1) \\ PF(g_2) \\ \cdots \\ PF(g_n) \end{pmatrix} = \begin{pmatrix} \Delta E(g_1) \\ \Delta E(g_2) \\ \cdots \\ \Delta E(g_n) \end{pmatrix}$$

This can also be written as:

$$(\mathbf{I} - \mathbf{B}) \cdot \mathbf{PF} = \Delta \mathbf{E} \quad (6.23)$$

where  $\mathbf{B}$  represents the normalized weighted directed adjacency matrix of the graph describing the gene signaling network:

$$\mathbf{B} = \begin{pmatrix} \frac{\beta_{11}}{N_{ds}(g_1)} & \frac{\beta_{12}}{N_{ds}(g_2)} & \cdots & \frac{\beta_{1n}}{N_{ds}(g_n)} \\ \frac{\beta_{21}}{N_{ds}(g_1)} & \frac{\beta_{22}}{N_{ds}(g_2)} & \cdots & \frac{\beta_{2n}}{N_{ds}(g_n)} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\beta_{n1}}{N_{ds}(g_1)} & \frac{\beta_{n2}}{N_{ds}(g_2)} & \cdots & \frac{\beta_{nn}}{N_{ds}(g_n)} \end{pmatrix} \quad (6.24)$$

$\mathbf{I}$  is the identity matrix, and

$$\Delta \mathbf{E} = \begin{pmatrix} \Delta E(g_1) \\ \Delta E(g_2) \\ \cdots \\ \Delta E(g_n) \end{pmatrix} \quad (6.25)$$



is the vector of measured expression changes for genes  $g_1, g_2, \dots, g_n$ . With this notation, the gene perturbation factors can be calculated as:

$$\mathbf{PF} = (\mathbf{I} - \mathbf{B})^{-1} \cdot \Delta \mathbf{E} \quad (6.26)$$

This system of equations can be solved exactly for most pathways [112].

This method was later implemented in the signaling pathway impact analysis (SPIA) R package [176]. In addition, the method implemented in SPIA takes a different approach at computing the significance of a given pathway. It computes a probability for each one of the two types of evidence: i) the over-representation of DE genes in a given pathway and ii) the abnormal perturbation of that pathway. The first probability,  $P_{NDE} = P(X \geq N_{DE}|H_0)$ , is usually computed using the hypergeometric distribution based on the assumption that the number of differentially expressed genes that fall in the pathway follows such distribution. The second probability captures the perturbation propagation and is computed based on the perturbation accumulation. The accumulation at the gene level is obtained from Eq. 6.2:

$$Acc(g_i) = PF(g_i) - \Delta E(g_i) \quad (6.27)$$

This term represents the perturbation accumulation at each one of the gene and therefore captures the information related to the topology of the pathway. The accumulation at the pathway level is computed as:

$$t_A = \sum_i Acc(g_i) \quad (6.28)$$

The significance of this term is computed through a bootstrap technique and estimates the probability  $P_{PERT} = P(T_A \geq t_A|H_0)$ . The two type of evidence,  $P_{NDE}$  and  $P_{PERT}$ , are then combined into a global probability. Multiple methods to combine the two independent evidences can be employed. In the original study the Fisher method [63] for combining evidences was used:

$$P_G = c_i - c_i \cdot \ln(c_i) \quad (6.29)$$

where  $c_i = P_{NDE}(i) \cdot P_{PERT}(i)$ .

## 6.5 Impact analysis that uses significance of individual genes

The impact analysis model presented in the previous sections takes in consideration the magnitude of expression change of each DE gene. Even though this has been proven to work well, this approach is not able to consider the reliability of each individual gene measurement. In all available analysis methods, including the classical impact analysis, once the set of DE gene is selected, genes that are marginally significant are given the same importance as highly significant genes. For instance, a gene that has a p-value of  $2.84 \times 10^{-10}$ , such as gene 2353 in Fig. 6.8(a), it is extremely clear that there is a huge expression change that is expected to happen by chance only once in ten billion cases. In contrast, a gene with a p-value of 0.049, such as gene 25966 in Fig. 6.8(a), is barely below the arbitrary threshold of 5%. To put things in perspective, one can get the latter change about once in every 20 cases compared to once in ten billion cases, as in the case of the other gene. All existing pathway analysis methods will ignore this difference and treat these two genes in the same way, although clearly the amount of trust that we can be put in these two measurements are very different.

In this particular case, the gene with the higher significance had a much higher fold change, as well, so one could argue that gene 2353 would therefore still weight more in the analysis. However, genes with the same measured fold change can still have widely different significance values depending on the shape of their distributions (see Fig. 6.8(b)). Another example are the genes 3725 and 84612 in Fig. 6.8(a). Even though the two genes have similar absolute fold changes, 1.53 and  $-1.21$  respectively, their significance is widely different  $1.15 \times 10^{-6}$  versus 0.0499. It is clear there is a need to take in consideration the significance values alongside the measured expression changes for each gene.

We proposed a new method by adding a term  $\alpha_g$  in the Eq. 6.2 that is able to capture the

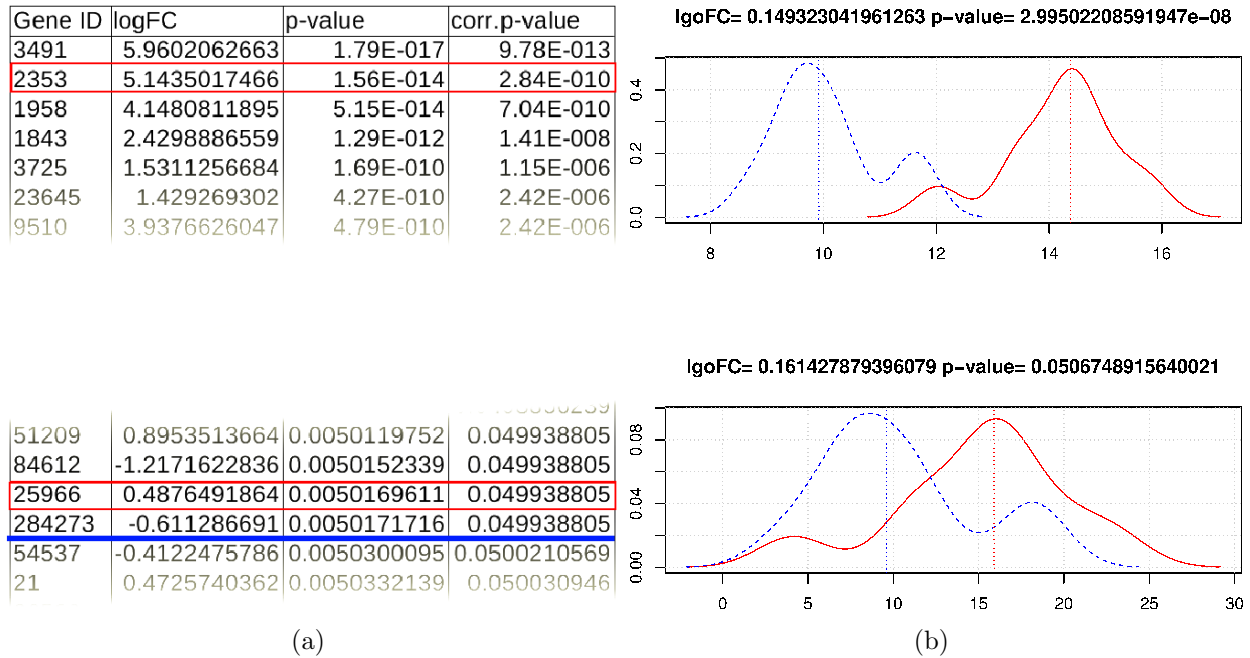


Figure 6.8: Shortcomings of analyses that ignore the p-value: (a) after the selection of differentially expressed genes (the genes above the 5% significance threshold marked by the blue line) all selected genes are considered to have the same importance. For instance, gene ID 2353 (marked with a red box) is given the same importance as gene ID 25966 (marked with a red box) in the colorectal cancer dataset ID: GSE4701 from GEO [56, 8]. (b) An example showing how two genes with the same fold change might have completely different significance. *Source:* [195].

information about the reliability of the measured expression change. This factor will weight the expression change  $\Delta E$  accordingly to its significance:

$$PF(g) = \alpha_g \cdot \Delta E(g) + \sum_{u \in US_g} \beta_{ug} \cdot \frac{PF(u)}{N_{ds}(u)} \quad (6.30)$$

The preliminary results presented here use the implementation of the impact analysis available as the SPIA R package [176]. Therefore, the p-value  $P_{PERT}$  is computed on the perturbation accumulation defined as:

$$Acc(g_i) = PF(g_i) - \alpha_g \cdot \Delta E(g_i) \quad (6.31)$$

We proposed two alternative expressions of the factor  $\alpha_g$ . The first alternative is the minus log (MLG) ratio between the p-value and the significance threshold:

$$\alpha_g = -\log \frac{p_g}{\alpha_t} \quad (6.32)$$

where,  $p_g$  is the significance value for the measured expression change of the gene  $g$ , and  $\alpha_t$  is the significance threshold used for the selection of differentially regulated genes. This modified formulation of the perturbation accumulation incorporates gene significance information into the model and allows the genes detected with a very small p-value (highly significant change) to contribute to the perturbation factor more than genes barely passing the threshold. For instance, if the usual significance level  $\alpha_t = 0.01$  is used and a gene is found to be differentially expressed with a p-value  $p_g = 0.0001$ , the corresponding perturbation will be:

$$PF(g) = -\log_{10} \frac{0.0001}{0.01} \cdot \Delta E(g) + \dots = 2 \cdot \Delta E(g) + \dots \quad (6.33)$$

Conversely, changes detected very close to the detection threshold ( $p_g \approx \alpha_t$ ) will be less

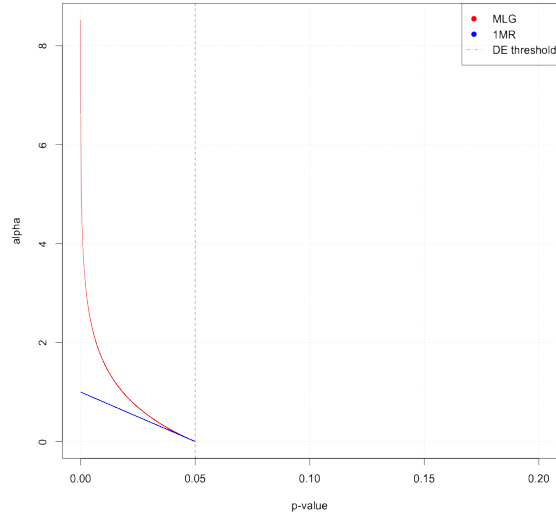


Figure 6.9: Comparison of the two weighting schemas for the gene significance. Notice how the small p-values are emphasized by the MLG model (red), while the 1MR model (blue) has a bounded range of values.

important:

$$PF(g) = -\log_{10} \frac{0.00999}{0.01} \cdot \Delta E(g) + \dots \approx 0.0004 \cdot \Delta E(g) + \dots \quad (6.34)$$

Because this formula will actually emphasize very small p-values with the danger of going to infinity when p-values are zero (some analysis packages can in fact produce p-values equal to zero), we considered an alternative expression:

$$\alpha_g = 1 - \frac{p_g}{\alpha_t} \quad (6.35)$$

We will refer to this alternative model as one minus ratio or 1MR. A graphical comparison between the two expressions is shown in Fig. 6.9.

For those genes with a  $p_g$  value close to zero (highly significant gene expression changes), the second term  $p_g/\alpha_t$ , will be close to zero,  $\alpha$  will be close to 1, and the entire fold change  $\Delta E$  is taken into consideration, as before. For genes with a p-value very close to the significance threshold used to select the genes, the term  $p_g/\alpha_t$  will be close to 1 and  $\alpha$  will be close to zero, making their fold changes contribute less to the perturbation factor.

We evaluated the impact analysis with gene significance by comparing the two alternative

SPIA			SPIA_MLG			SPIA_1MR		
Name	pv	adj.pv	Name	pv	adj.pv	Name	pv	adj.pv
ECM-receptor interaction	0.000005	0.00034	<b>Colorectal cancer</b>	0.004	0.181	ECM-receptor interaction	0.000005	0.00034
Focal adhesion	0.000005	0.00034	Dilated cardiomyopathy	0.004	0.181	Focal adhesion	0.000005	0.00034
Small cell lung cancer	0.002000	0.09067	Serotonergic synapse	0.004	0.181	Small cell lung cancer	0.001000	0.04533
Glutamatergic synapse	0.005000	0.17000	Bile secretion	0.006	0.181	Glutamatergic synapse	0.009000	0.22667
VEGF signaling pathway	0.010000	0.21371	Amphetamine addiction	0.007	0.181	Pathways in cancer	0.010000	0.22667
Pathways in cancer	0.011000	0.21371	Prion diseases	0.008	0.181	VEGF signaling pathway	0.010000	0.22667
Systemic lupus erythematosus	0.011000	0.21371	<b>Toll-like receptor signaling pathway</b>	0.013	0.253	Pathogenic Escherichia coli infection	0.022000	0.42500
Pathogenic Escherichia coli infection	0.023000	0.31733	Protein processing in endoplasmic reticulum	0.015	0.255	Systemic lupus erythematosus	0.025000	0.42500
Chemokine signaling pathway	0.023000	0.31733	Focal adhesion	0.023	0.348	<b>Colorectal cancer</b>	0.033000	0.48960
Cytokine-cytokine receptor interaction	0.025000	0.31733	Cocaine addiction	0.033	0.420	Dilated cardiomyopathy	0.036000	0.48960
<b>Colorectal cancer</b>	0.027000	0.31733	Pathways in cancer	0.034	0.420	Type II diabetes mellitus	0.042000	0.51927
African trypanosomiasis	0.028000	0.31733	Systemic lupus erythematosus	0.046	0.435	<b>PPAR signaling pathway</b>	0.063000	0.54400
Hepatitis C	0.043000	0.43714	VEGF signaling pathway	0.046	0.435	Serotonergic synapse	0.063000	0.54400
Staphylococcus aureus infection	0.045000	0.43714	ECM-receptor interaction	0.047	0.435	Staphylococcus aureus infection	0.068000	0.54400
...	...	...	...	...	...	...	...	...

Table 6.1: Comparison of two models that incorporate gene significance (SPIA\_MLG and SPIA\_1MR) with the model without (SPIA). The *Colorectal cancer pathway* and *Toll-like receptor signaling pathway* are both ranked better by SPIA\_MLG. More over, the *PPAR signaling pathway* is rank better by SPIA\_1MR.

methods (SPIA\_MLG, using Eq. 6.32, and SPIA\_1MR, using Eq. 6.35) with SPIA [176] on a publicly available colorectal cancer dataset (GSE4701 [89]). In addition, we use the framework proposed by Tarca *et. al.* [175] to compare the ranking of the signaling pathways over a pool of 24 datasets.

### 6.5.1 Colorectal cancer case study

The dataset chosen for this case study consists of gene expression profiling, using the Affymetrix HG-U133 Plus 2.0 microarray platform, of 12 early onset of colorectal cancer samples versus 10 normal samples [89]. The data is available as part of the Gene Expression Omnibus [56, 8] (ID: GSE4701). After normalizing the data set, we performed a moderated t-test for each probe to compute the significance of the change between the two conditions and selected for each gene the probe with the most significant change. The false discovery rate (FDR) is used to control for multiple comparison error. For the models that require selection of DE genes we use a 1% threshold on the FDR-adjusted p-value and a log fold change threshold of 1.5. The comparison of the top ranked pathways is presented in Table 6.1.

Besides the *Colorectal cancer pathway* itself, several other pathways are know to be related to colorectal cancer including: *PPAR signaling pathway* [167] and *Toll-like receptor signaling pathway* [199, 68]. The SPIA\_MLG model ranks the both *Colorectal cancer* and the *Toll-like*

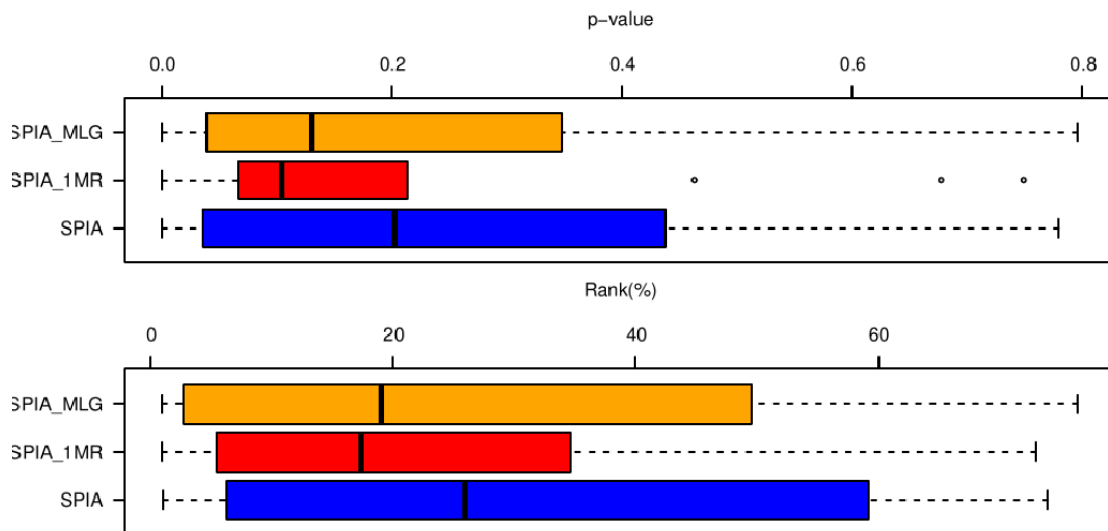
*receptor signaling* pathways better than the original SPIA method.

### 6.5.2 Rank comparison over a pool of 24 datasets

The comparison framework proposed by Tarca *et.al.* [175] consists of 24 datasets from multiple biological conditions including: Alzheimer's disease, Parkinson's disease, Huntington's disease, colorectal cancer, renal cancer, pancreatic cancer, etc. For each dataset a target pathway is defined that is most likely to be affected in the respective condition (e.g., colorectal cancer pathway is relevant to colorectal cancer). Given that all KEGG pathways were constructed from literature search [103] and to maintain this method objective we made sure that none of the 24 datasets were used in order to construct the KEGG pathways. For each of the methods compared, the rank and p-value of the target pathway is recorded for all datasets. We compare the methods based on the distribution of the ranks of the target pathways. We only consider the datasets where at least 3 DE genes are found on the target pathway. Given our threshold choice, only 14 out of the 24 datasets satisfy this requirement. The comparison of the rank distributions is presented in Fig. 6.10. Both in terms of ranks and p-values the two models that incorporate gene significance (SPIA\_MLG and SPIA\_1MR) perform better than the method that does not incorporate it (SPIA), yielding lower p-values for the target pathway, hence offering a more accurate insight on the biological phenomenon.

## 6.6 Summary

This chapter contains the description and evolution of a pathway impact analysis that includes information related to the topology of the pathway. This allows it to capture important biological factors like: i) type and position of each of the differentially expressed genes in the pathway; ii) the magnitude of their expression change; and iii) the type of interaction between all genes in the pathway. In addition, a method to incorporate the



Method	p geomean	p med	% p.value<0.05	% q.value<0.05	rank mean	rank med
1 SPIA_MLG	0.0379	0.1310	28.57%	14.29%	27.3518	19.0368
2 SPIA_1MR	0.0300	0.1046	21.43%	14.29%	24.2791	17.3387
3 SPIA	0.0424	0.2021	28.57%	14.29%	32.7214	25.9420

Figure 6.10: Comparison using the list of DE genes: distribution of the rank and p-value of the target pathway over 14 data sets. Both methods that incorporate gene significance rank the target pathways better than SPIA and also assign them a more significant p-value. There is no clear difference in performance between the two methods we proposed. *Source:* [195].



significance of the expression change (p-value) of individual genes was presented.

## Chapter 7      Cut-off free impact analysis

With the recent advances in high-throughput technologies, such as RNA-Seq, we are able to measure and quantify gene expression at unprecedented levels of quality and coverage. The typical next-generation sequencing pipeline starts by sequencing the entire transcriptome, in a true genome-wide effort. Many millions of short DNA sequences are obtained from the entire transcriptome of a sample. This transcriptome is obtained from the entire genome consisting of approximately 3 billion base pairs. These hundreds of millions of reads are assembled and mapped on the entire genome in an immensely challenging computational task. Eventually, an expression level is computed for each of the approximately 100,000 transcripts. Expression levels are then calculated for each gene and then a small subset of differentially expressed (DE) genes are selected. Usually, the set of DE genes consists of a few hundred genes. A typical size of 300 DE genes would represent only 1% of the 30,000 known genes. Most current pathway analysis techniques use only the DE genes and throw away the remaining 99% of the gene expression values in spite of the huge laboratory and computational effort that was used to quantify their values. Furthermore, the current approach uses only 1% of the genes to allegedly infer “system-wide” conclusions about the given phenotype. This extremely severe truncation of the information available is illustrated in Fig. 7.1. There are literally many tens of thousand of genes whose expression changes are completely ignored just because they do not meet an arbitrary threshold. It has been already shown that the choice of threshold used to select the DE genes to be considered further, can affect dramatically both the set of genes selected, as well as the final results of the analysis [140]. The challenge here is to allow the pathway analysis stage to really use “genome-wide” data, and while at the same time prevent the noise present in the entire data set from overwhelming the analysis.

This chapter explores a new impact analysis method that is able to include all the mea-

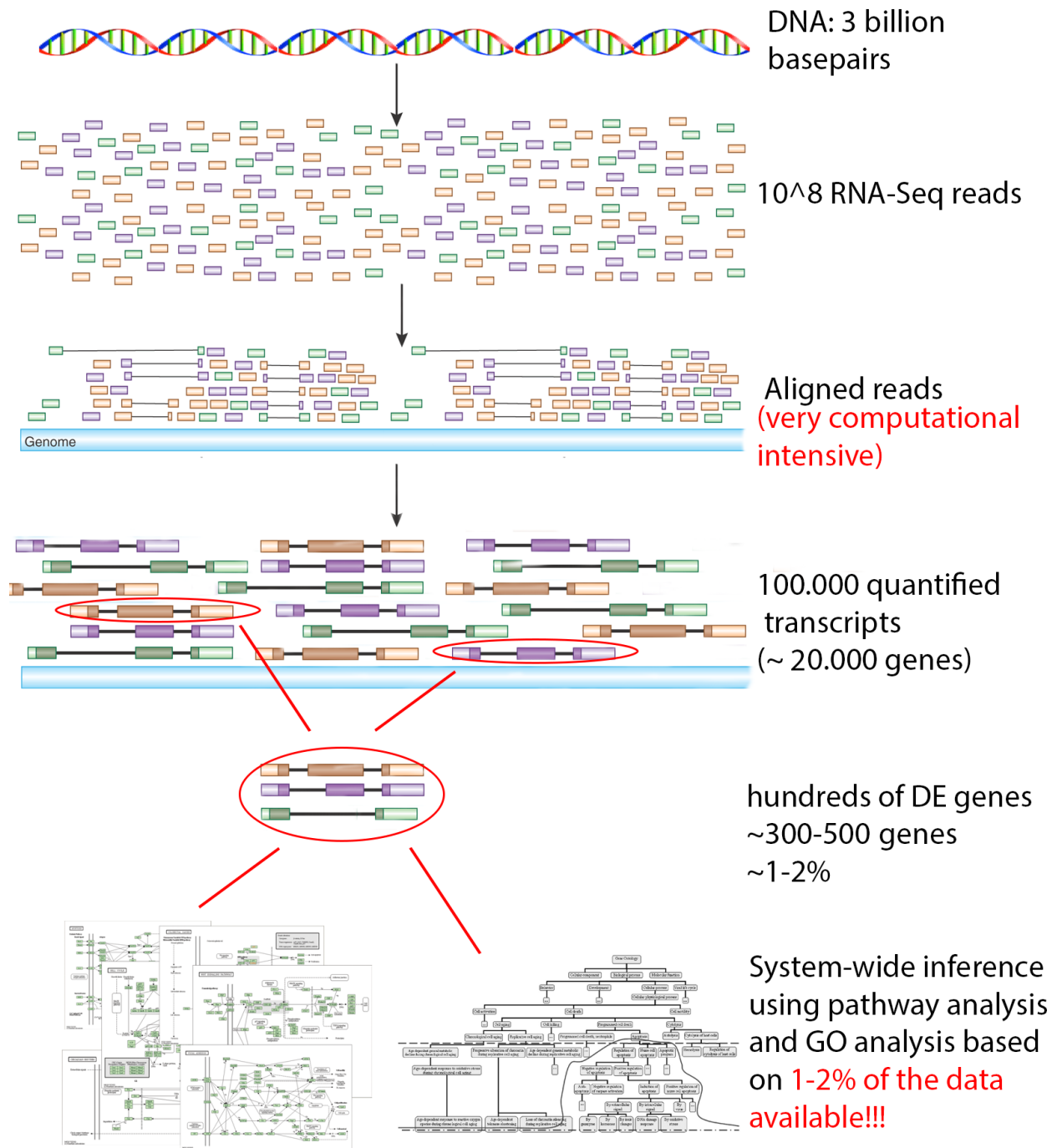


Figure 7.1: The selection of DE genes is a severe truncation of the information available. *Source:* Based on figures from [84, 110].

sured expression changes available in the experiment [195]. This method does allow the pathway analysis to effectively use all measured gene expression changes, thus having the potential to incorporate wide-spread or important changes that are below the usual significance thresholds. Another important benefit of this approach is that the results of the pathway analysis stage become much more reproducible since the huge variability introduced by the selection of DE genes is eliminated. This method was deployed as part of the ROntoTools Bioconductor package (See Chapter 8).

## 7.1 Method

One characteristic of the impact analysis is that is based on an *a priori* selection of differentially expressed (DE) genes. However, analyzing only the list of DE genes represents an artificial truncation of the information available, as well as an unnecessary reliance on an upstream gene selection method, which may be far from optimal [140]. Here we present a method able to perform impact analysis using the entire set of gene expression values and their significance without the need for a preliminary gene selection. We now define the weight of the gene as follows:

$$ALL\_MLG : \alpha_g = -\log \frac{p_g}{p_{max}} \text{ and } ALL\_1MR : \alpha_g = 1 - \frac{p_g}{p_{max}} \quad (7.1)$$

In this new model  $p_g$  is the p-value of a given gene  $g$ , while the  $p_{max}$  is the maximum p-value (least significant) calculated across the entire set of genes. In many cases, this will be 1 or close to it yielding:

$$ALL\_MLG : \alpha_g = -\log p_g \text{ and } ALL\_1MR : \alpha_g = 1 - p_g \quad (7.2)$$

A graphical representation between the two models is presented in Fig. 7.2.

The entire set of expression values can now be considered, with the added benefit that

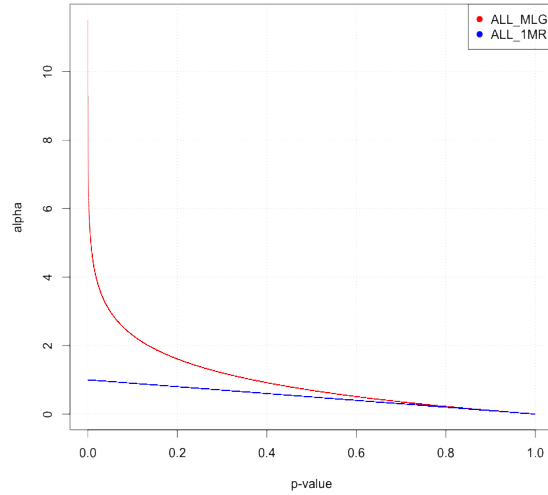


Figure 7.2: Comparison of the two weighting schemas for the gene significance. Notice how the small p-values are emphasized by the ALL\_MLG model (red), while the ALL\_1MR model (blue) has a bounded range of values.

highly significant genes will be automatically weighted more while less significant genes will be weighted less.

The impact analysis with differentially expressed genes relies on two types of evidence, one of which is the probability to observe a higher number of DE genes on the given pathway just by chance. Since all the genes are now considered and therefore all the genes in the pathway are expected to be measured, the over-representation p-value  $P_{NDE}$  is not defined. For this reason, the significance of a pathway is represented only by the probability of obtain a higher perturbation than the one observed  $P_{PERT}$ . This probability is computed using a bootstrap approach.

## 7.2 Results and discussion

Similar to the evaluation with DE genes, we evaluated the utility of using a cut-off free approach by comparing the two alternative methods (ALL\_MLG and SPIA\_1MR, using Eq. 7.1) with SPIA [176] on a publicly available colorectal cancer dataset (GSE4701 [89]). Moreover, we use the framework proposed by Tarca *et. al.* [175] to compare the ranking of

SPIA				ALL_MLG				ALL_1MR			
Name	pv	adj.pv		Name	pv	adj.pv		Name	pv	adj.pv	
ECM-receptor interaction	0.000005	0.00034		Focal adhesion	0.002	0.174		Cytokine-cytokine receptor interaction	5.0e-06	0.000171	
Focal adhesion	0.000005	0.00034		Serotonergic synapse	0.004	0.174		Chemokine signaling pathway	5.0e-06	0.000171	
Small cell lung cancer	0.002000	0.09067		<b>Colorectal cancer</b>	0.005	0.174		Focal adhesion	5.0e-06	0.000171	
Glutamatergic synapse	0.005000	0.17000		ECM-receptor interaction	0.006	0.174		ECM-receptor interaction	5.0e-06	0.000171	
VEGF signaling pathway	0.010000	0.21371		Dilated cardiomyopathy	0.007	0.174		Staphylococcus aureus infection	1.0e-03	0.022833	
Pathways in cancer	0.011000	0.21371		Prion diseases	0.010	0.174		Systemic lupus erythematosus	1.0e-03	0.022833	
Systemic lupus erythematosus	0.011000	0.21371		Parkinson's disease	0.011	0.174		Pathways in cancer	2.0e-03	0.034250	
Pathogenic Escherichia coli infection	0.023000	0.31733		Cocaine addiction	0.013	0.174		Small cell lung cancer	2.0e-03	0.034250	
Chemokine signaling pathway	0.023000	0.31733		<b>PPAR signaling pathway</b>	0.014	0.174		Pathogenic Escherichia coli infection	5.0e-03	0.076111	
Cytokine-cytokine receptor interaction	0.025000	0.31733		Bile secretion	0.014	0.174		<b>PPAR signaling pathway</b>	9.0e-03	0.112091	
<b>Colorectal cancer</b>	0.027000	0.31733		Pathways in cancer	0.014	0.174		<b>Colorectal cancer</b>	9.0e-03	0.112091	
African trypanosomiasis	0.028000	0.31733		Systemic lupus erythematosus	0.016	0.183		Hepatitis C	1.0e-02	0.114167	
Hepatitis C	0.043000	0.43714		Renal cell carcinoma	0.018	0.190		Glutamatergic synapse	1.3e-02	0.137000	
Staphylococcus aureus infection	0.045000	0.43714		Protein processing in endoplasmic reticulum	0.023	0.209		Sulfur relay system	1.6e-02	0.146133	
...	...	...		...	...	...		...	...	...	

Table 7.1: Comparison of two cut-off free models (ALL\_MLG and ALL\_1MR) with the model original model (SPIA). The *Colorectal cancer pathway* and *PPAR signaling pathway* are both ranked better by ALL\_MLG.

the signaling pathways over a pool of 24 datasets.

## Colorectal cancer case study

The dataset chosen for this case study consists of gene expression profiling, using the Affymetrix HG-U133 Plus 2.0 microarray platform, of 12 early onset of colorectal cancer samples versus 10 normal samples [89]. The data is available as part of the Gene Expression Omnibus [56, 8] (ID: GSE4701). After normalizing the data set, we performed a moderated t-test for each probe to compute the significance of the change between the two conditions and selected for each gene the probe with the most significant change. The false discovery rate (FDR) is used to control for multiple comparison error. The comparison of the top ranked pathways when using a cut-off free analysis is presented in Table 7.1.

Using the cut-off free analysis, both the *Colorectal cancer pathway* and *PPAR signaling pathway* which are known to be related to colorectal cancer, are ranked better than the analysis with DE genes.

## Rank comparison over a pool of 24 datasets

The comparison framework proposed by Tarca *et.al.* [175] consists of 24 datasets from multiple biological conditions including: Alzheimer's disease, Parkinson's disease, Huntig-

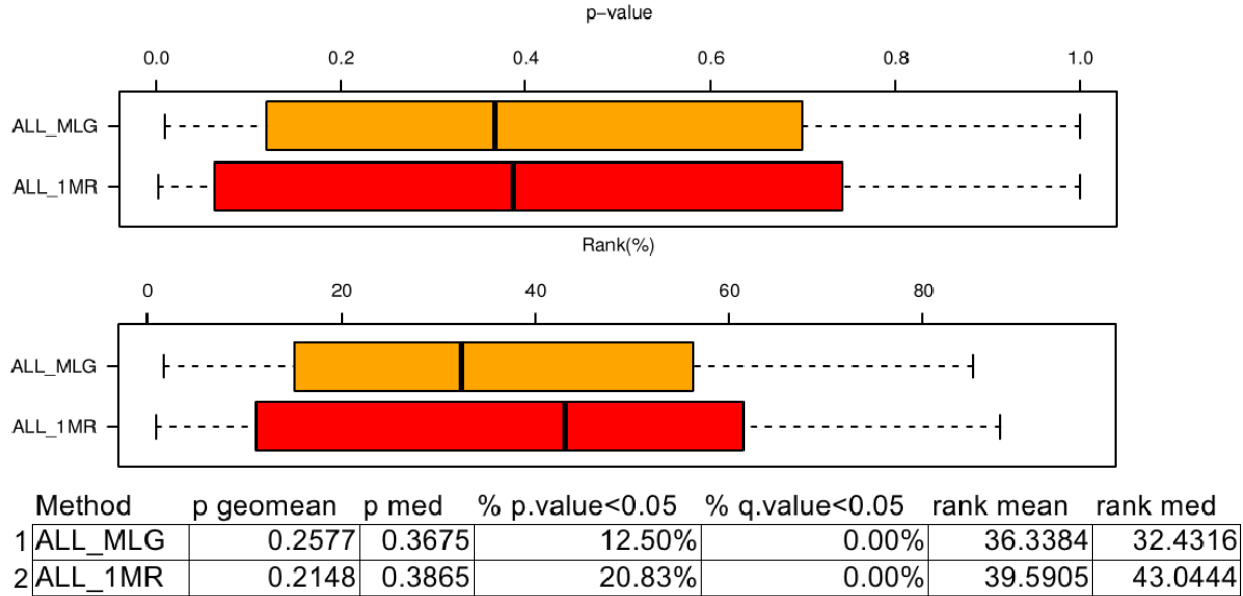


Figure 7.3: **Comparison of cut-off free analysis:** distribution of the rank and p-value of the target pathway over 24 data sets. Both methods perform similar in terms of rank and p-value with the ALL\_MLG model performing slightly better. *Source:* [195].

ton's disease, colorectal cancer, renal cancer, pancreatic cancer, etc. For each dataset a target pathway is defined that is most likely to be affected in the respective condition (e.g., colorectal cancer pathway is relevant to colorectal cancer). For each of the methods compared, the rank and p-value of the target pathway is recorded for all datasets. We compare the methods based on the distribution of the ranks of the target pathways. Fig. 7.3 shows that ALL\_MLG model performs slightly better than ALL\_1MR.

### 7.3 Multiple sclerosis case study

The goal of this section is to assess the ability of the cut-off free impact analysis to return reproducible results independent of the experiment design and platform. To this extent we searched the available public databases for experiments (data sets) that investigate the same phenomenon or disease. The databases used were the Gene Expression Omnibus [56, 8] and ArrayExpress [158]. For this analysis we search for experiments comparing normal versus multiple sclerosis (MS) blood samples. The data sets we found are:

KEGG id	Pathway title
<a href="#">hsa05140</a>	Leishmaniasis - Homo sapiens (human)
<a href="#">hsa04940</a>	Type I diabetes mellitus - Homo sapiens (human)
<a href="#">hsa05330</a>	Allograft rejection - Homo sapiens (human)
<a href="#">hsa05332</a>	Graft-versus-host disease - Homo sapiens (human)
<a href="#">hsa05323</a>	Rheumatoid arthritis - Homo sapiens (human)
<a href="#">hsa05320</a>	Autoimmune thyroid disease - Homo sapiens (human)
<a href="#">hsa05310</a>	Asthma - Homo sapiens (human)

Table 7.2: Significant pathways in common among the four datasets

- E-MTAB-358 [126]: Expression profiling of peripheral blood mononuclear cells (PBMCs) in 19 MS patients and 14 controls.
- E-GEOD-21942 [107]: The experiment was conducted to perform a genome-wide expression study in peripheral blood mononuclear cells (PBMC) from 12 MS patients and 15 controls in order to identify differentially expressed genes and pathways in MS.
- E-GEOD-17449 [74]: Two comparison were performed for women before pregnancy (healthy vs MS) and at 9 months (healthy vs MS).
- E-MTAB-380 [129]: Expression profiling of peripheral blood mononuclear cells (PBMCs) in 23 Relapsing-Remitting Multiple Sclerosis (RRMS) and 22 controls.
- E-GEOD-17048 [70] : Expression analysis of whole blood cells in 99 MS (43 Primary progressive MS, 36 RRMS, 20 Secondary Progressive MS), and 45 healthy controls.

In order to remove the variability we excluded the experiment that only analyzed data coming from pregnant women (E-GEOD-17449). We independently performed the cut-off free impact analysis on all the remaining data sets. Using a 10% significance threshold, we identified seven pathways that are significant in **all** data sets (see Fig. 7.4). These pathways are presented in Table 7.2. All these pathways are associated in one way or another to the immune response, which is to be expected as is well known that multiple sclerosis is an autoimmune disease that affects the brain and spinal cord (central nervous system). Based on this observation we tested if there is any common module among these pathways that would be associated to multiple sclerosis.



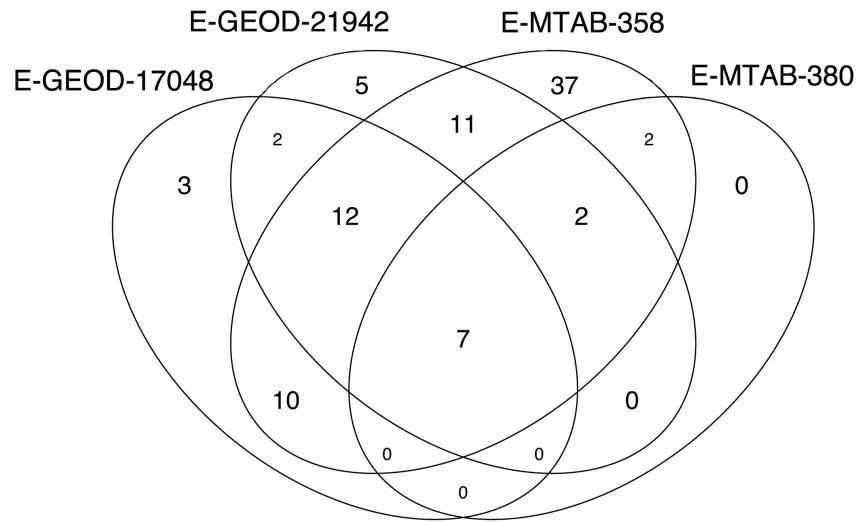


Figure 7.4: Overlap of significant pathways over the four MS data sets. A group of seven pathways are identified as significant over all four data sets. These pathways are all associated with the immune response which is a good confirmation as multiple sclerosis is an autoimmune disease.

Independent of any dataset, we selected the set of genes that are in common among all the seven pathways and considered it as a separate pathway. In addition, we removed this module from any pathway that contains all the genes in the module. Using the updated set of pathways we then re-analyzed all the datasets. The only pathway that was found significant in all datasets was the new module (see Fig. 7.5). Also, the module was found to be significant even in the dataset that was originally discarded (E-GEOD-17049). This module is composed by genes belonging to the Major Histocompatibility Complex II (MHCII), found to be strongly associated with autoimmune diseases [202].

## 7.4 Summary

This chapter described an impact analysis method that is not influenced by the threshold used to select differentially expressed genes. This approach is able to consider all measured gene expressing changes, thus eliminating the possibility that small but important gene changes (e.g. transcription factors) are omitted from the analysis. Furthermore, this approach provides consistent results that do not depend on the choice of an arbitrary thresh-

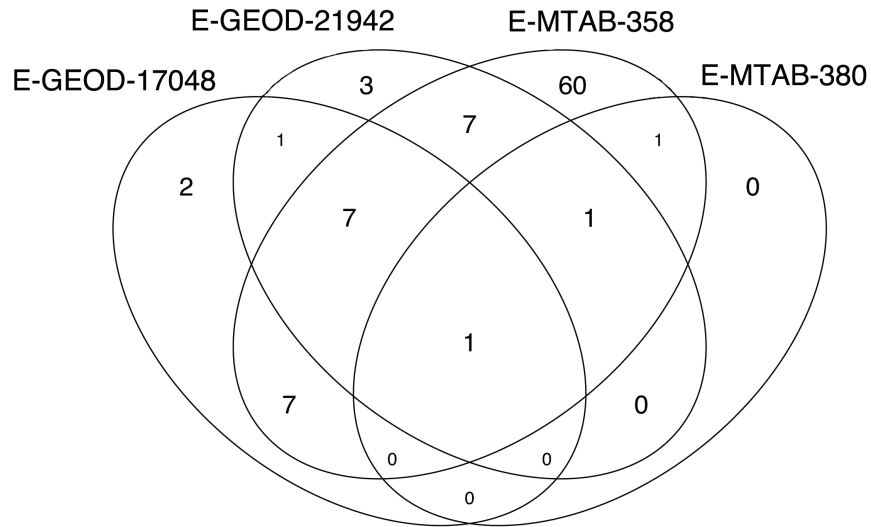


Figure 7.5: Overlap of significant pathways over four MS data sets after module extraction. The module found to be in common among the seven significant pathways in all four data sets has been extracted from all pathways and considered as a different module. After performing the cut-off free analysis over the updated set of pathways, only the new module was found to be significant in all four data sets.

old for the differential regulation. It is also shown that this approach is able to obtain consistent results across multiple experiments performed by different groups on different technologies.

## Chapter 8 ROntoTools User Guide

This chapter contains the user guide for the Pathway-Express (`pe`) tool for the analysis of signaling pathways. This tool was released in the *ROntoTools* package<sup>1</sup> as part of Bioconductor 2.12 [73] and it was well received with downloads from more than 100 unique IP's per month (Fig. 8.1). The analyses implemented in this package include the impact analysis with differentially expressed genes [52, 176], the incorporation of gene significance [195] and the elimination of the need to select differentially expressed genes [195]. The impact analysis requires two sources of data, one is the pathway database and the other is the experimental data that we wish to analyze. The following sections describe how to prepare the set of pathways, how to set the gene significance, the format of the experimental data and how to visualize the results.

### 8.1 Pathway database

Pathway-Express is a general tool that accepts any set of signaling pathways defined using the standard implementation provided in the *graph* package. The only requirement is that each pathway, defined as an object of type *graph*, has a weight defined for each edge, representing the efficiency of the propagation between the two genes, and a weight for each node, that will capture the type of gene or the significance of the measured expression change. This package provides tools to access the KEGG database for signaling pathways and also tools to set these weights.

For example, to download and parse the signaling pathways available in KEGG use:

```
> require(graph)
> require(ROntoTools)
> kpg <- keggPathwayGraphs("hsa", verbose = FALSE)
```

<sup>1</sup><http://bioconductor.org/packages/2.12/bioc/html/ROntoTools.html>

### Download stats for Software package ROntoTools

This page was generated on 2013-06-24 08:03:31 -0700 (Mon, 24 Jun 2013).

ROntoTools home page: [release version](#), [devel version](#).

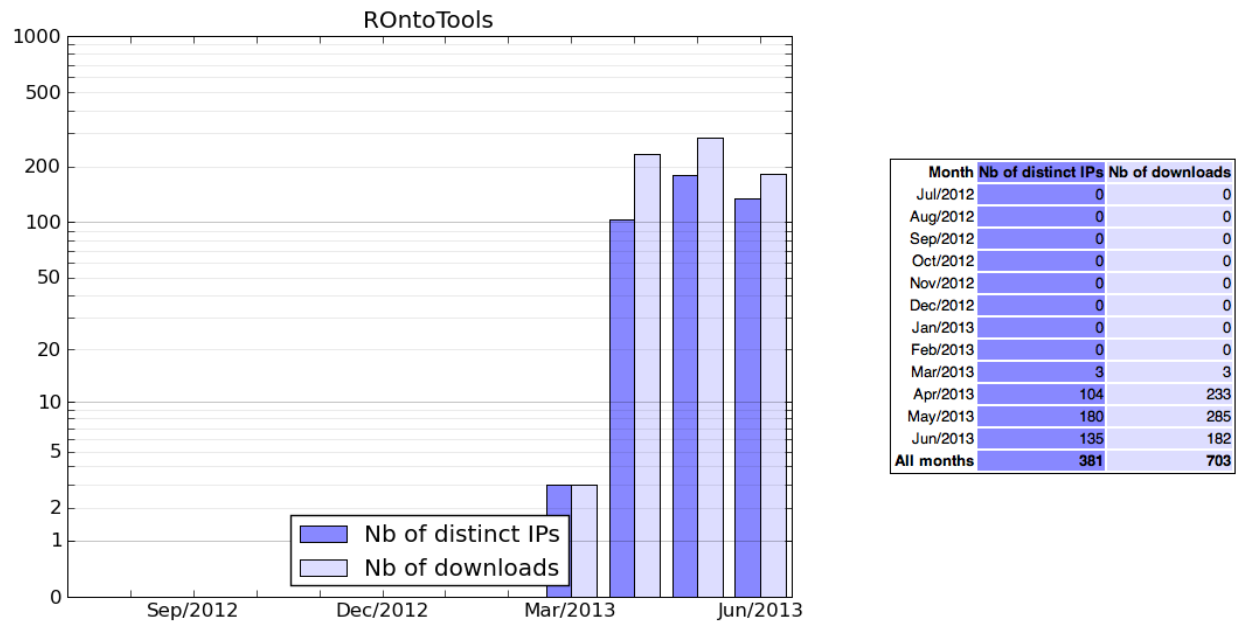


Figure 8.1: ROntoTools download stats as obtained from the official Bioconductor website: <http://bioconductor.org/packages/stats/bioc/ROntoTools.html>.

The above code will load the available cached data for human (i.e., KEGG id *hsa*). To update the cache and download the latest KEGG pathways available use the `updateCache` parameter:

```
> kpg <- keggPathwayGraphs("hsa", updateCache = TRUE, verbose = TRUE)
```

This command is time consuming and depends on the available bandwidth.

The `kpg` is a list of *graph* objectes:

```
> head(names(kpg))
```

```
[1] "path:hsa03008" "path:hsa03013" "path:hsa03015" "path:hsa03018"
```

```
[5] "path:hsa03320" "path:hsa03460"
```

To inspect one of the pathway graphs, only the ID is required. Here is an example for the Cell Cycle:

```
> kpg[["path:hsa04110"]]
```

A graphNEL graph with directed edges

Number of Nodes = 124

Number of Edges = 630

```
> head(nodes(kpg[["path:hsa04110"]]))
```

```
[1] "hsa:1029" "hsa:51343" "hsa:4171" "hsa:4172" "hsa:4173" "hsa:4174"
```

```
> head(edges(kpg[["path:hsa04110"]]))
```

```
$`hsa:1029`
```

```
[1] "hsa:4193" "hsa:1019" "hsa:1021" "hsa:595" "hsa:894" "hsa:896"
```

```
$`hsa:51343`
```

```
[1] "hsa:983" "hsa:85417" "hsa:891" "hsa:9133"
```

```
$`hsa:4171`
```

```
character(0)
```

```
$`hsa:4172`  
character(0)
```

```
$`hsa:4173`  
character(0)
```

```
$`hsa:4174`  
character(0)
```

In addition the parser extracted the type of interaction for each gene-gene interaction in an attribute called `subtype`:

```
> head(edgeData(kpg[["path:hsa04110"]], attr = "subtype"))
```

```
$`hsa:1029|hsa:4193`  
[1] "inhibition"
```

```
$`hsa:1029|hsa:1019`  
[1] "inhibition"
```

```
$`hsa:1029|hsa:1021`  
[1] "inhibition"
```

```
$`hsa:1029|hsa:595`  
[1] "inhibition"
```

```
$`hsa:1029|hsa:894`  
[1] "inhibition"
```

```
$`hsa:1029|hsa:896`
```

```
[1] "inhibition"
```

Using this attribute the function `setEdgeWeights` sets the same weight for all the interactions of the same type:

```
> kpg <- setEdgeWeights(kpg, edgeTypeAttr = "subtype",
+   edgeWeightByType = list(activation = 1, inhibition = -1,
+   expression = 1, repression = -1),
+   defaultWeight = 0)
```

At this point, `kpg` contains a list of graphs with weighted edges:

```
> head(edgeData(kpg[["path:hsa04110"]], attr = "weight"))
```

```
$`hsa:1029|hsa:4193`
```

```
[1] -1
```

```
$`hsa:1029|hsa:1019`
```

```
[1] -1
```

```
$`hsa:1029|hsa:1021`
```

```
[1] -1
```

```
$`hsa:1029|hsa:595`
```

```
[1] -1
```

```
$`hsa:1029|hsa:894`
```

```
[1] -1
```

```
$`hsa:1029|hsa:896`
```

```
[1] -1
```

To retrieve the title of the pathways and not just their ids the function `keggPathwayNames`

can be used:

```
> kpn <- keggPathwayNames("hsa")
> head(kpn)
          path:hsa03008          path:hsa03013
"Ribosome biogenesis in eukaryotes"  "RNA transport"
          path:hsa03015          path:hsa03018
"mRNA surveillance pathway"         "RNA degradation"
          path:hsa03320          path:hsa03460
"PPAR signaling pathway"           "Fanconi anemia pathway"
```

## 8.2 Experiment data

As an example, the package includes a pre-processed data set from ArrayExpress ([E-GEOD-21942](#)) that studies the expression change in peripheral blood mononuclear cells (PBMC) between 12 MS patients and 15 controls. The data was preprocessed using the *limma* package [24]. Only probe sets with a gene associated to them have been kept and for each gene only the most significant probe set has been selected (the table is already ordered by p-value):

```
> load(system.file("extdata/E-GEOD-21942.topTable.RData",
+               package = "ROntoTools"))
> head(top)
          logFC      P.Value  adj.P.Val  entrez
200946_x_at -1.0175141 5.833411e-13 4.172652e-09  hsa:2746
228697_at   -3.6479368 7.985427e-13 4.172652e-09  hsa:135114
210254_at    3.2807123 3.086572e-12 9.677020e-09  hsa:932
234726_s_at -0.9792301 7.368175e-12 1.760593e-08  hsa:64418
215905_s_at -1.7733135 7.861797e-12 1.760593e-08  hsa:9410
```



```
235542_at    -0.9447467 1.617944e-11 2.536288e-08 hsa:200424
```

Select differentially expressed genes at 1% and save their fold change in a vector *fc* and their p-values in a vector *pv*:

```
> fc <- top$logFC[top$adj.P.Val <= .01]
> names(fc) <- top$entrez[top$adj.P.Val <= .01]
> pv <- top$P.Value[top$adj.P.Val <= .01]
> names(pv) <- top$entrez[top$adj.P.Val <= .01]
> head(fc)
 hsa:2746 hsa:135114    hsa:932  hsa:64418    hsa:9410 hsa:200424
-1.0175141 -3.6479368  3.2807123 -0.9792301 -1.7733135 -0.9447467
> head(pv)
 hsa:2746  hsa:135114    hsa:932  hsa:64418    hsa:9410  hsa:200424
5.833411e-13 7.985427e-13 3.086572e-12 7.368175e-12 7.861797e-12 1.617944e-11
```

Alternatively, an analysis with all genes can be performed:

```
> fcAll <- top$logFC
> names(fcAll) <- top$entrez
> pvAll <- top$P.Value
> names(pvAll) <- top$entrez
```

The reference contains all the genes measured in the analysis:

```
> ref <- top$entrez
> head(ref)
[1] "hsa:2746"    "hsa:135114" "hsa:932"     "hsa:64418"  "hsa:9410"
[6] "hsa:200424"
```

### 8.3 Setting the node weights

The node weights are used to encode for the significance of each gene, the term described as  $\alpha$  in [195]. The two alternative formulas to incorporate the gene significance:

$$\alpha = 1 - p/p_{thr} \text{ and } \alpha = -\log(p/p_{thr}) \quad (8.1)$$

are implemented as two function `alpha1MR` and `alphaMLG`.

To set the node weights the function `setNodeWeights` is used:

```
> kpg <- setNodeWeights(kpg, weights = alphaMLG(pv), defaultWeight = 1)
> head(nodeWeights(kpg[["path:hsa04110"]]))
hsa:1029 hsa:51343 hsa:4171 hsa:4172 hsa:4173 hsa:4174
1.0000000 1.0000000 0.8120949 1.0000000 1.0000000 1.0000000
```

### 8.4 Pathway analysis and results summary

Up to this point all the pieces need for the analysis have been assembled:

- the pathway database with the experiment specific gene significance - `kpg`
- the experiment data - `fc` and `ref`

To perform the analysis the function `pe` is used (increase the parameter `nboot` to obtain more accurate results):

```
> peRes <- pe(x = fc, graphs = kpg, ref = ref, nboot = 200,
+           verbose = FALSE)
```

The result object can be summarized in a table format with the desired columns using the function `Summary`:

```
> head(Summary(peRes))
              totalAcc totalPert totalAccNorm totalPertNorm      pPert
path:hsa05010 21.5034630 128.32957    0.6904808    2.616553 0.029850746
```

```

path:hsa05110 22.8375919 87.30055 4.7628726 5.958016 0.004975124
path:hsa04142 0.2232419 90.73185 0.1145369 5.086767 0.004975124
path:hsa04145 0.0000000 102.93799 NA 6.198880 0.004975124
path:hsa05152 140.0243475 237.47577 5.5160343 6.589622 0.004975124
path:hsa04722 56.2219227 114.29999 2.0457742 3.099216 0.004975124

```

```

          pAcc          pORA          pComb pPert.fdr  pAcc.fdr
path:hsa05010 0.422885572 8.544061e-06 4.127118e-06 0.04231271 0.58883799
path:hsa05110 0.004975124 1.085083e-04 8.330837e-06 0.01452736 0.04067896
path:hsa04142 0.915422886 2.047498e-04 1.507308e-05 0.01452736 0.96916875
path:hsa04145          NA 2.424942e-04 1.764759e-05 0.01452736          NA
path:hsa05152 0.009950249 5.666823e-04 3.884739e-05 0.01452736 0.05762852
path:hsa04722 0.039800995 9.960632e-04 6.548746e-05 0.01452736 0.10245071

```

```

          pORA.fdr  pComb.fdr
path:hsa05010 0.001264521 0.0006025592
path:hsa05110 0.008029611 0.0006081511
path:hsa04142 0.008972285 0.0006441370
path:hsa04145 0.008972285 0.0006441370
path:hsa05152 0.012677874 0.0011343437
path:hsa04722 0.016379706 0.0015935283

```

```

> head(Summary(peRes, pathNames = kpn, totalAcc = FALSE, totalPert = FALSE,
+          pAcc = FALSE, pORA = FALSE, comb.pv = NULL, order.by = "pPert"))

```

```

          pathNames          pPert  pPert.fdr
path:hsa03013          RNA transport 0.004975124 0.01452736
path:hsa04020          Calcium signaling pathway 0.004975124 0.01452736
path:hsa04060 Cytokine-cytokine receptor interaction 0.004975124 0.01452736
path:hsa04062          Chemokine signaling pathway 0.004975124 0.01452736
path:hsa04066          HIF-1 signaling pathway 0.004975124 0.01452736

```

path:hsa04110

Cell cycle 0.004975124 0.01452736

## 8.5 Graphical representation of results

To visualize the summary of the Pathway-Express results use the function `plot` (see Fig. 8.2):

```
> plot(peRes)
> plot(peRes, c("pAcc", "pORA"), comb.pv.func = compute.normalInv,
+       threshold = .01)
```

Pathway level statistics can also be displayed one at a time using the function `plot` (see Fig. 8.3):

```
> plot(peRes@pathways[["path:hsa05216"]], type = "two.way")
> plot(peRes@pathways[["path:hsa05216"]], type = "boot")
```

To visualize the propagation across the pathway, two functions - `peNodeRenderInfo` and `peEdgeRenderInfo` - are provided to extract the required information from a `pePathway` object:

```
> p <- peRes@pathways[["path:hsa05216"]]
> g <- layoutGraph(p@map, layoutType = "dot")
> graphRenderInfo(g) <- list(fixedsize = FALSE)
> edgeRenderInfo(g) <- peEdgeRenderInfo(p)
> nodeRenderInfo(g) <- peNodeRenderInfo(p)
> renderGraph(g)
```

This is the *Thyroid cancer* signaling pathway and is shown in Fig. 8.4. Another example is the *T cell receptor signaling pathway* and is presented in Fig. 8.5. These pathways can also be visualized with other tool, like the *pathview* [124].

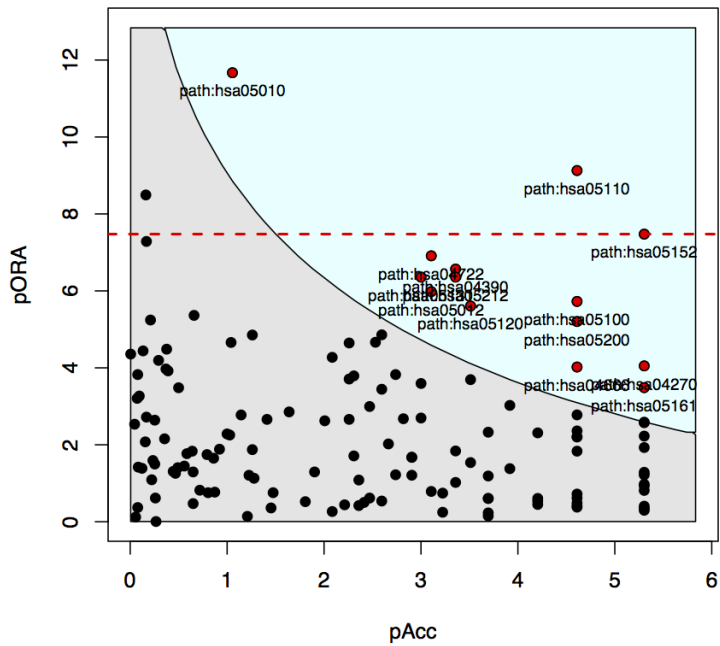
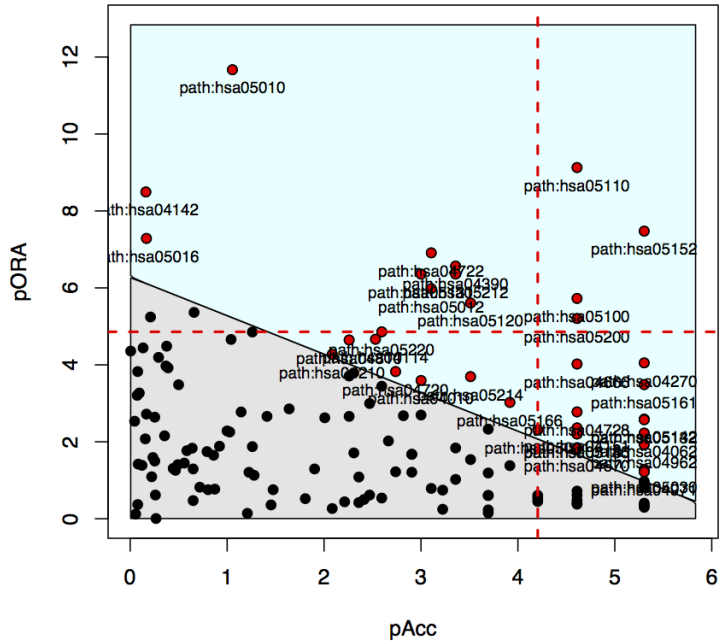


Figure 8.2: Two-way plot of Pathway-Express result

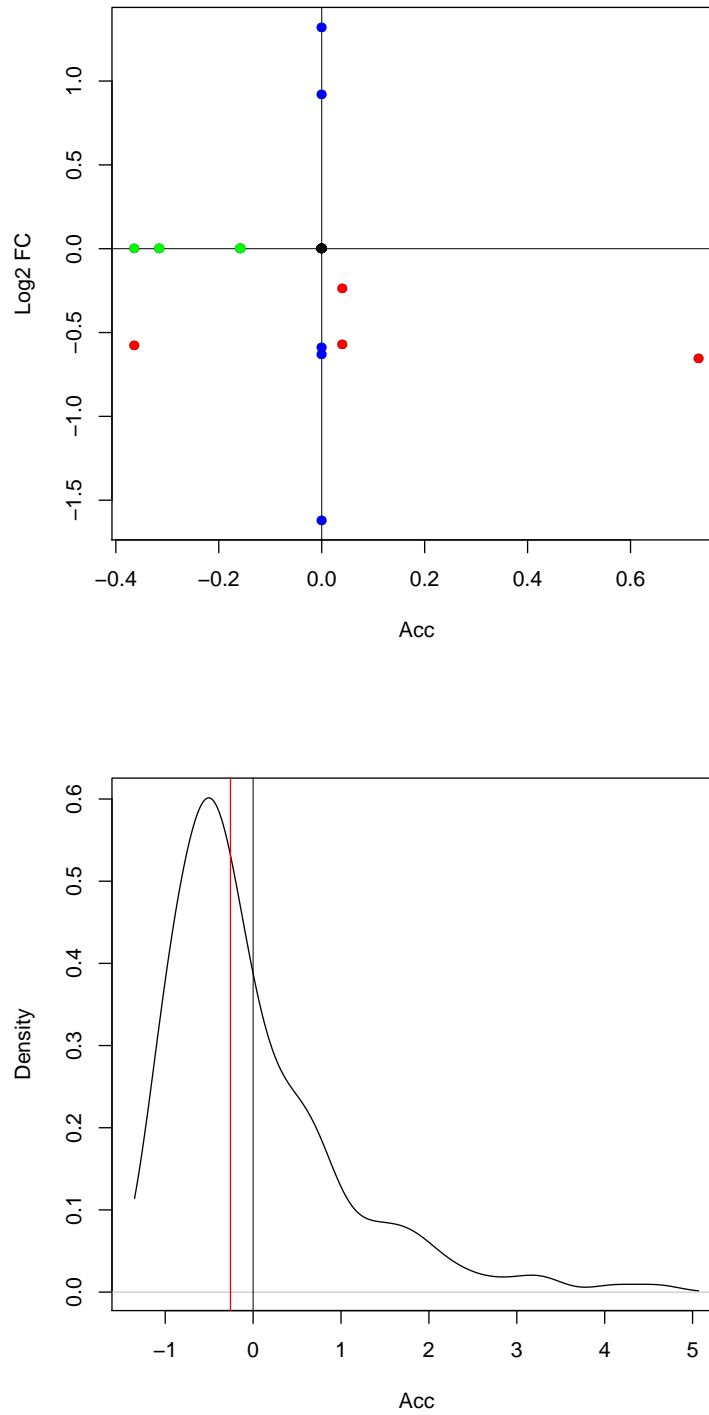


Figure 8.3: Pathway level statistics: perturbation accumulation versus the measured expression change (above) and the bootstrap simulations of the perturbation accumulation (below).

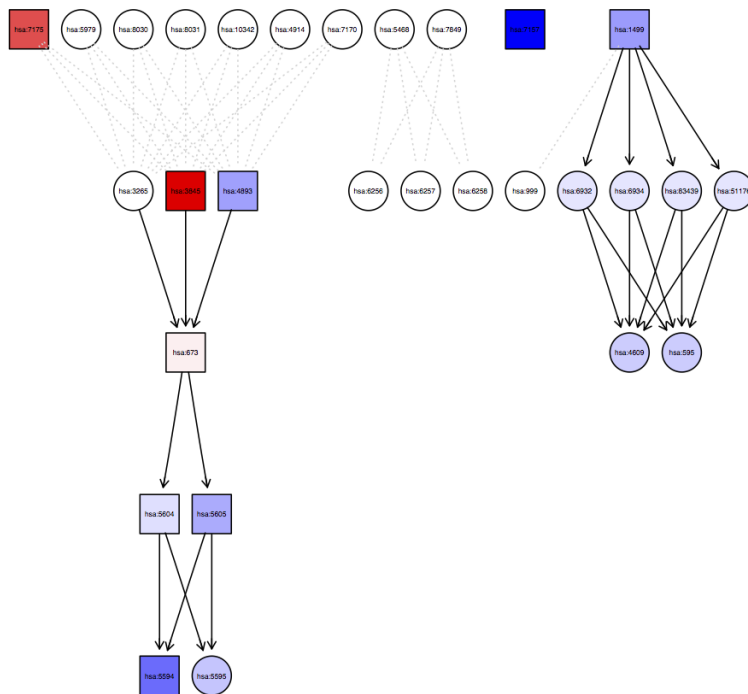


Figure 8.4: Perturbation propagation on the *Thyroid cancer signaling pathway*. The input genes are marked with squares, while all the other genes are circles. The color of the genes represent either a positive (red) or negative (blue) perturbation, while the intensity represents the magnitude of the respective perturbation. The interactions marked with arrows represent activation-like signals and have a positive weight. The dotted interactions have zero weight and no perturbation will be propagated through these edges. Notice how the propagation flows throughout the pathway from the input genes through the non-zero weight edges.

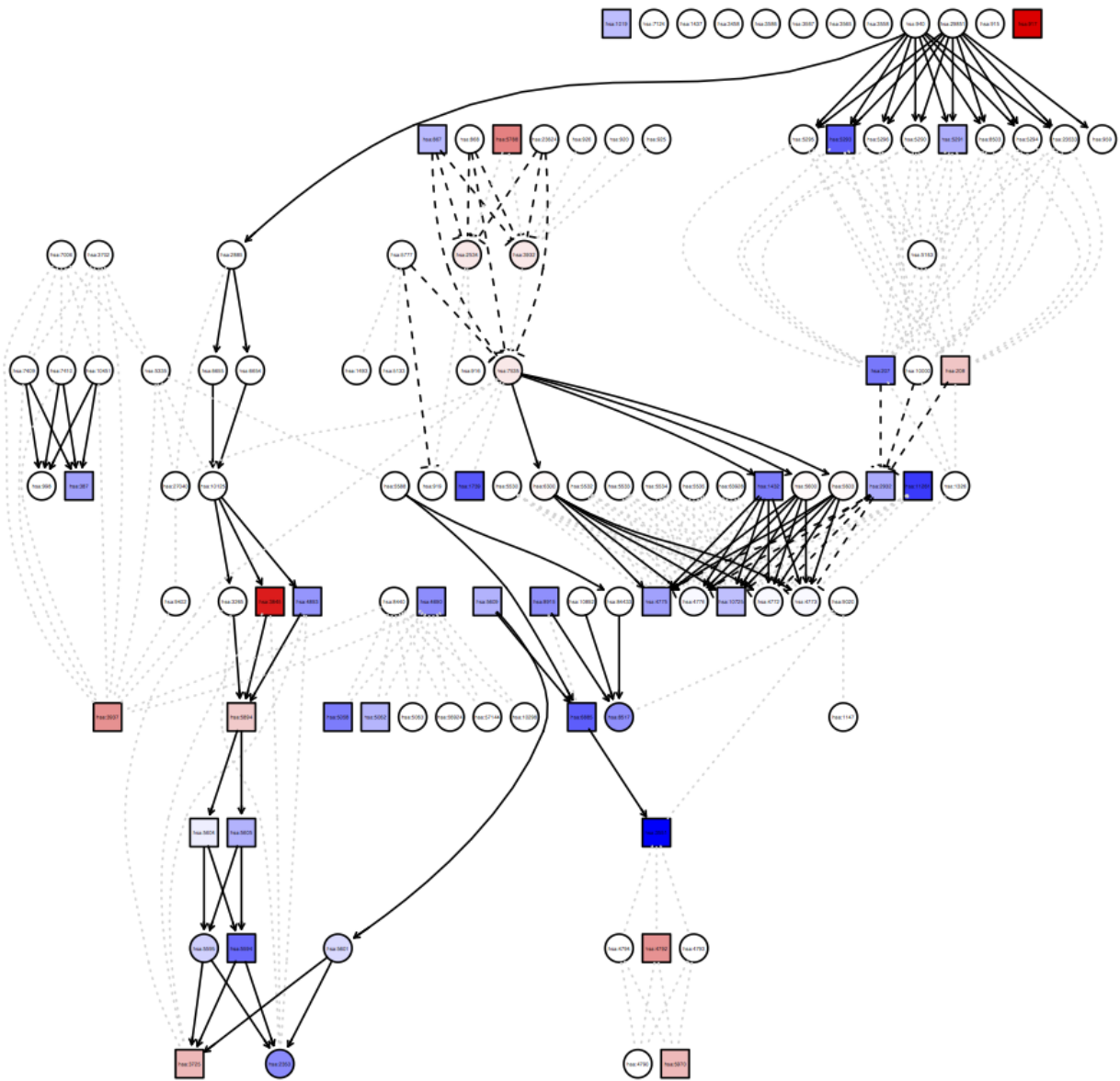


Figure 8.5: Perturbation propagation on the *T cell receptor signaling pathway*. The input genes are marked with squares, while all the other genes are circles. The color of the genes represent either a positive (red) or negative (blue) perturbation, while the intensity represents the magnitude of the respective perturbation. The interactions marked with arrows represent activation-like signals and have a positive weight, while the dashed arrows represent inhibition-like events and have a negative weight. The dotted interactions have zero weight and no perturbation will be propagated through these edges. Notice how the propagation flows throughout the pathway from the input genes through the non-zero weight edges.



## 8.6 Future work

This package is intended to be the R implementation of the web-based data mining and analysis suite of tools called Onto-Tools [111, 50, 48, 51, 108, 48, 113, 109, 53, 108, 109, 43, 109, 115, 52, 112]. Among these, Onto-Express (OE) was the first publicly available tool for the GO profiling of high throughput data and Pathway-Express (PE) the first tool to perform analysis of signaling pathways using important biological factors like all the interactions between the genes, the type of interaction between them and the position and magnitude of expression change for all the differentially expressed genes. The online tools have over 10,000 registered users from 53 countries. Approximately, 5,000 of these are regular users (more than 10 data sets processed). This R package will provide these users with access to the direct functionalities of the online version and to new analysis methods.

# Chapter 9 Estimating Gene Contributions in Signaling Pathways

In this chapter we are addressing a problem that is affecting all available pathway analysis methods. All methods rely on the quality of the available pathways. These pathways were designed to describe the general mechanism of a particular disease or biological process. The known pathways encompass the results of many biological experiments and even though they represent our current understanding of those particular biological processes, they are still generally considered sketchy and incomplete. One piece of information that is generally missing regards the role or importance of a gene in a given pathway which we refer to as the gene contribution. We describe a method, based on genetic algorithms, to objectively quantify the contribution of each gene. This method was proposed in [196].

## 9.1 Genetic algorithms

Genetic algorithms (GA) are search methods based on natural concepts such as natural selection, evolution and genetics. In its basic form, a GA involves the evolution of a fixed size population across generations, where each individual of the population represents a possible solution in the search space. Each individual is represented by a set of *genes*, and each gene is represented in a way that depends on the implementation of the GA (e.g. binary string or floating point). The evolution of the population results in one or more of the individuals satisfying a certain criterion of the search, for example a local maximum or minimum. The evolution is led by a few key events: selection, crossover, and mutation. Selection is the process of elimination of the individuals that do not pass a fit-test. Several methods exist for selection, but the basic idea is to give preference to the better individuals. The key element in the selection is the way of determining which individual is better. For this purpose,

an evaluation can come from an objective function that gives each individual a score that can serve, for example, for ranking the population. Crossover is the event in which two individuals A and B are chosen to be mated. The two sets of genes belonging to the two individuals are parted in the same way, and then two new individuals are constructed taking one of the partitions of A and one the partitions of B. Mutation, like crossover, is a way to explore different structures. This event represents a single, usually low-probability, random change in a gene. Selection, crossover, and mutation are applied across generations, and the average evaluation of the population increases. A stopping criterion is then applied (e.g. limit to the number of generations, threshold on the result of the evaluation function for the best individuals), and the best individuals are chosen as solutions. The use of GAs in bioinformatics is widespread, from applications in sequence alignment [80] to RNA structure prediction [186]. This technique, however, found little use in the context of regulatory pathway analysis. Most of the approaches apply genetic algorithms while trying to model the underlying, unknown, network [215], or to simulate the network dynamics [90]. However, to date there are no applications, to the best of our knowledge, of GAs to the analysis of regulatory pathways in the context of phenotype change.

The capabilities of the GA paradigm have been greatly expanding by adding an additional level on which information is transmitted from one generation to another, much like human beings do through their trans-generational culture. These **cultural algorithms** [151] use a belief space to incorporate *a priori* domain knowledge that will be updated throughout the evolutionary process. Even though cultural algorithms have been shown to work well in a large number of applications [37, 144, 153, 152], we elected genetic algorithms as a baseline of what can be achieved through evolutionary computation. Cultural algorithms are an extension of genetic algorithms and therefore we expect to achieve even better performance when we will extend to this approach.

## 9.2 Determining gene contributions

Using the cut-off impact analysis presented in the Chapter 7, where the gene contribution can be captured by the factor  $\alpha_g$  in Eq. 7.1, and genetic algorithms (GA) we propose a framework to estimate the individual gene contributions in signaling pathways. Given a predefined set of pathways  $S_P = \{P_1, P_2, \dots, P_k\}$ , we obtain the set of unique genes  $U$  contained in these pathways:

$$U = \bigcup_{P_i \in S_P} \{g \in P_i\} \quad (9.1)$$

We design our individual as a vector of size equal to the size of the set  $U$ . Each gene, in the context of genetic algorithms, will be a floating point number between 0 and 1, representing the contribution of each gene  $g \in U$ :

$\alpha_{g_1}$	$\alpha_{g_2}$	$\alpha_{g_3}$	$\dots$	$\alpha_{g_n}$
----------------	----------------	----------------	---------	----------------

where  $\alpha_{g_i} \in [0, 1]$ .

The genetic algorithm has been implemented in the R framework [179], adapting the *genalg* package for parallel execution of the evaluation of the individuals. Single point uniform mutation chance has been set to 10%, while the selection from one generation to another was elitism of the top 20% of the population ranked by fitness. The size of the population was selected equal to and double the size of the individual. The type of crossover was single point. All these parameters have been chosen according to the indications in [78].

The goal of our evaluation function is to capture the ability of the gene weights to model biological knowledge encoded in the given pathways and not any specific condition (i.e., disease). Based to the objective method [175] to compare pathway analysis methods that was used in Section 6.5 we designed our evaluation function. This comparison technique is based on the concept of *target pathway*. In this framework, one selects a number of public datasets that are obtained from samples associated with a given condition (e.g. colorectal cancer, pancreatic cancer, Parkinson's disease, etc.). These conditions are selected based on

the fact that a known pathway exists for each of them. These pathways become the “target pathways” for their respective data sets. The idea is that if we did not know what condition a patient was suffering from, we would like a good pathway analysis method to be able to correctly identify the associated condition-related pathway as significant in that particular condition. In other words, the colorectal cancer pathway should be reported as significant in a colorectal cancer dataset, etc. Therefore, given an *a priori* defined set of data sets  $DS$  with their associated target pathways, the evaluation function scores each individual by applying the impact analysis on each data set independently and recording the normalized rank of the target pathway associated with each data set. The return value of the evaluation function will be the average normalized rank of the target pathway over all data sets in  $DS$ . Hence, the lower the result of the evaluation function, the better the individual.

### 9.3 Training

Our starting pool of data sets consists of 24 data sets that represent 12 different conditions and therefore involving 12 different target pathways. This pool of data sets is available in the R package *KEGGdzPathwaysGEO* and is summarized in Table 9.1. We divide this pool into *training* and *testing* groups to emulate two scenarios. These two scenarios were chosen in a way that captures an ideal environment and the real environment. In the ideal environment, each one pathway of the 140 pathways available in the KEGG database would be associated with a dataset. Since this is not the case, in the real environment we have both pathways that are associated to a dataset and pathways that are not.

For both scenarios we selected the training and testing datasets in a similar fashion. First, we performed cut-off free impact analysis with the default set of  $\alpha = 1$ , on each data set. We next ordered the data sets based on the normalized rank of the target pathway and selected datasets starting at the top of the list. This approach allowed us to avoid data sets in which the target pathway was badly ranked, possibly indicating that those particular data sets

Data set	Disease / Condition	Pathway	Scen. 1	Scen. 2
GSE1297	Alzheimer's disease	hsa05010	training	training
GSE5281_EC	Alzheimer's disease	hsa05010	testing	testing
GSE5281_HIP	Alzheimer's disease	hsa05010	testing	testing
GSE5281_VCX	Alzheimer's disease	hsa05010	testing	testing
GSE20153	Parkinson's disease	hsa05012	testing	testing
GSE20291	Parkinson's disease	hsa05012	training	training
GSE8762	Huntington's disease	hsa05016	-	testing
GSE4107	Colorectal Cancer	hsa05210	training	training
GSE8671	Colorectal Cancer	hsa05210	testing	testing
GSE9348	Colorectal Cancer	hsa05210	testing	testing
GSE14762	Renal Cancer	hsa05211	-	-
GSE781	Renal Cancer	hsa05211	-	-
GSE15471	Pancreatic Cancer	hsa05212	training	training
GSE16515	Pancreatic Cancer	hsa05212	testing	testing
GSE19728	Glioma	hsa05214	-	testing
GSE21354	Glioma	hsa05214	-	training
GSE6956C	Prostate Cancer	hsa05215	-	training
GSE6956AA	Prostate Cancer	hsa05215	-	testing
GSE3467	Thyroid Cancer	hsa05216	-	training
GSE3678	Thyroid Cancer	hsa05216	-	testing
GSE9476	Acute myeloid leukemia	hsa05221	training	-
GSE18842	Non-Small Cell Lung Cancer	hsa05223	-	testing
GSE19188	Non-Small Cell Lung Cancer	hsa05223	-	training
GSE3585	Dilated cardiomyopathy	hsa05414	-	-

Table 9.1: The pool of data sets used for the analysis with their association to scenario 1 / scenario 2 and training / testing groups. For a more detailed description see [175]

contained bad data or they were not representative for that particular condition.

## The real environment scenario

Based on the ordered list of data sets, in the first scenario we selected as *training* data sets the top five data sets that represent different conditions. The *testing* set was chosen based on the conditions of five data sets selected as *training*. All the remaining data sets that study one of the five conditions was selected as the *testing* group. The set of pathways for which the gene contributions will be estimated was chosen based on the *training* data sets. We selected five pathways representing the target pathway for the respective conditions. We then selected the top three pathways (based on the impact analysis results) on each of the five *training* data sets to be part of the *training* set. This provided an additional 11 pathways, generating a training set that included a total of 16 pathways. The set of genes used in this scenario was the set of genes that appear at least once in any of the pathway selected for training. We obtained a set of 1,355 genes. As described in Section 9.2, each gene

is associated with a parameter  $\alpha$ . Therefore, each individual had a total of 1,355 genes to be used in the genetic algorithm search. We choose the size of the population to be equal to the number of genes of an individual (1,355 individuals), and we performed 100 generations with a mutation rate of 1%. We applied an elitist selection to the population, after each evaluation, where the top 20% individuals in the list ranked by the evaluation function were passed to the next generation with no crossover.

### The ideal environment scenario

In the second scenario, based on the same ordered list of data sets we selected the top ranked data set for each condition as the *training* group. Given that some of the conditions did not have at least two data sets associated to them and therefore no *testing* data set could be selected, these conditions are removed from further analysis. Two other datasets relative to *renal cancer* had to be removed due to the excessive variability of the rank of the target pathway when the analysis was performed with default parameters. Hence, the *training* group would contain eight data sets and the *testing* group eleven data sets (see Table 9.1). As this scenario would represent the ideal environment, we only selected for analysis the eight pathways associated as target pathways with the conditions in the *training* group. The total number of unique genes in these pathways was 372. We choose the size of the population to be 600, and we performed 100 generations with a mutation rate of 1%. We applied an elitist selection to the population, after each evaluation, where the top 20% individuals in the list ranked by the evaluation function were passed to the next generation with no crossover.

## 9.4 Evaluation

For both scenarios, at the end of the evolution of the population we extracted a random individual among the individuals with the best ranks (smallest value from the evaluation function) and we evaluated its performance in the testing set. This yielded an average

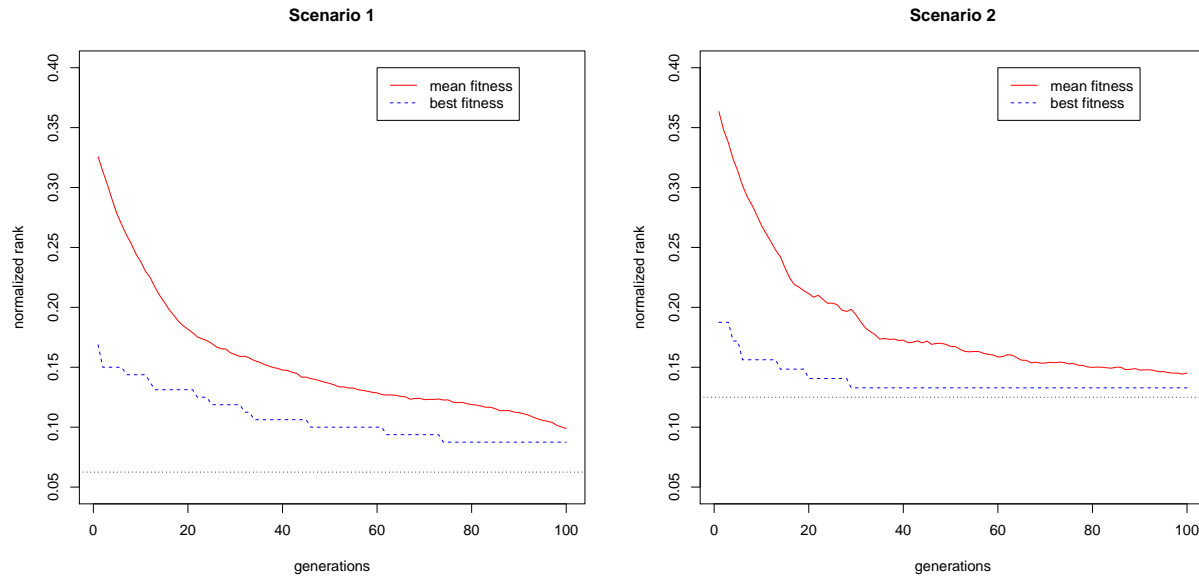


Figure 9.1: The evolution of the best and mean evaluations over the entire population at each generation. The evaluation function for one individual is the mean normalized rank over the training data sets. Because the evaluation function uses the normalized rank, the minimal value of the evaluation function is dependent on the total number of pathways evaluated (16 for Scenario 1 and 8 for Scenario 2). This minimal value achievable is show with a horizontal dotted line and is equal to  $1/16$  for Scenario 1 and  $1/8$  for Scenario 2. This value represents the case where the target pathways are ranked as first in all training data sets. According to the objective function chosen, the minimal value could not have been achieved due to ties. Other evaluation functions could do better. *Source:* [196].

normalized rank of the individual on the testing set of datasets. The evolution of the fitness function over all generation for each scenario is presented in Fig. 9.1.

For both scenarios, the rank of the best individuals was better than the result obtained with default parameters. By default parameters, we refer to the case where the contribution of each gene is considered to be maximum  $\alpha_g = 1$  for all genes (see Eq. 6.2). In order to assess if the performance of the individual was significantly better than a random choice, we used a bootstrap approach, generating the null distribution of the average normalized ranks, as described in [57]. We obtained this by creating 1,000 individuals with values of the gene weights randomly drawn from an uniform distribution with range  $[0, 1]$ . Each individual was then evaluated on the testing set. This procedure yields a p-value, computed as the number of random individuals that obtain a score lower than the score of the best individual. This p-value represents the probability of getting a score lower than the best



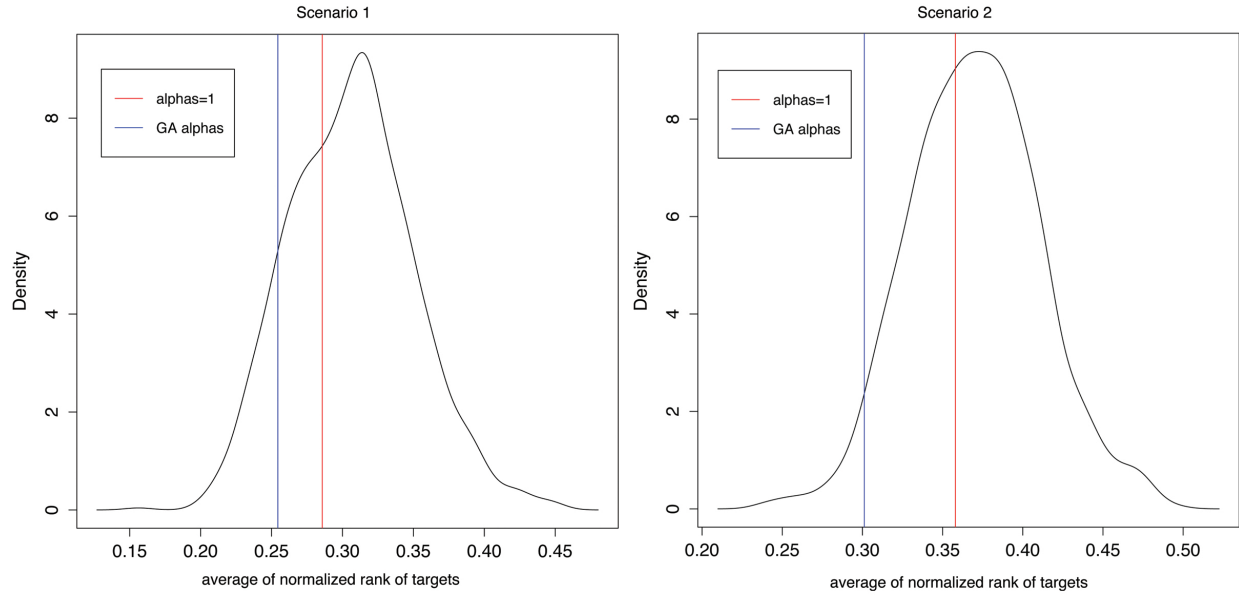


Figure 9.2: Null distributions of the average mean ranks of random individuals on the testing sets. The left panel shows the distribution of the evaluation of random individuals on the *testing* set associated with the selection of pathways in the first scenario, while the right panel shows the distribution of the evaluation of random individuals on the *testing* set associated with the selection of pathways in the second scenario. The blue lines represent the value of the average normalized rank of the best individual in the two populations, while the red lines represent the average mean rank of the *default* individuals (all the  $\alpha$ 's equal to 1). The results show that the default values are reasonable but only slightly better than those provided by a random choice. In both cases, the values obtained after the GA search are significantly better than the mean of the random chance values. *Source:* [196].

individual just by chance. We performed this procedure for the populations obtained with both scenarios described in Section 9.3. The best individual of the population obtained from the first scenario achieved a p-value of 10.4% on the testing set, while the best individual of the population obtained with the second scenario achieved a p-value of 3.3% on the testing set. The distributions relative to the two bootstraps are shown in Figure 9.2.

These results show that in both cases the optimization reaches significantly better results than the results obtained with the default set of parameters. In the testing set of the *real environment* scenario described in Section 9.3 a random choice would perform worse than the optimized parameters 89.6% of the times. In the testing set of the *ideal environment* scenario described in Section 9.3 a random choice would perform worse than the optimized parameters in more than 96% of the times. These values are also considerably better than those obtained with the default values.

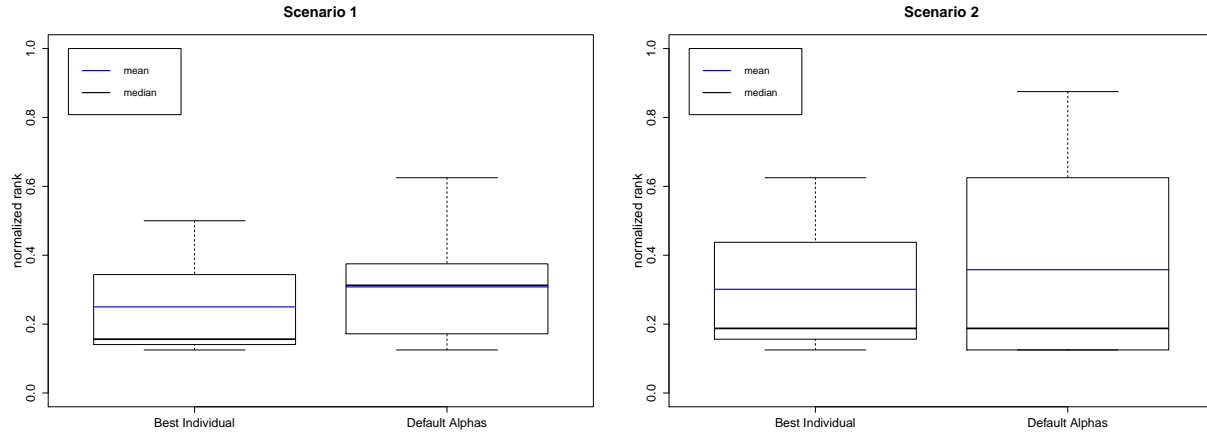


Figure 9.3: Normalized ranks of target pathways using parameters from best individuals (left side of each panel) and default parameters (right side of each panel). The left panel shows the comparison between the best individual of scenario 1 and default parameters in the testing set from scenario 1, while the right panel shows the comparison between the best individual of scenario 2 and the default parameters in the testing set from scenario 2. The blue line represents the mean of ranks, while the black line represents the median. In the left panel (scenario 1, real environment) the optimization procedure results in lower mean rank and lower median. In the right panel (scenario 2, ideal environment) the optimization procedure results in the lower mean rank, reduced variance, and the same median. *Source:* [196].

Figure 9.3 shows the comparisons between the ranks obtained with best individuals and the ranks obtained with the default parameters for both scenarios when we perform the analysis in the respective testing sets. The left panel shows the comparison performed in the testing set of the first scenario. In this scenario the best individual outperforms the default parameters obtaining a lower mean (0.25 against 0.335), lower median (0.156 against 0.25), while there is no improvement in terms of variance (0.296 against 0.207). The right panel shows the comparison performed in the testing set of the second scenario. In this scenario the best individual obtains a lower mean (0.301 against 0.357), the same median (0.187), and a decreased variance (0.038 against 0.102).

## 9.5 Summary

This chapter presented an evolutionary computation framework able to assess the individual gene contribution in the impact analysis of signaling pathways. This framework was exemplified using two experimental scenarios. The evaluation (i.e., fitness) of each individual

was performed with the method described in [175] and the data sets available in the R package *KEGGdzPathwaysGEO*. We assessed the statistical significance of the results of these optimization processes with a bootstrap technique, and we confirmed that the results are significantly better than the results obtained using default parameters. This optimization framework shows that evolutionary computation approaches such as genetic algorithms can be successfully applied in the optimization of the parameters of the impact analysis.

The evaluation framework is one of the limitations of this approach, due to the limited number of data sets included, as well as the fact that two of them, related to *renal cancer* had to be excluded due to the excessive variability of the rank of the target pathway when the analysis was performed with default parameters. Also, some biological aspects of the conditions described in the various data sets could be neglected by the *target pathway* approach. Probably the most important limitation associated with this framework is still related to the evaluation of the pathway analysis results. Our evaluation only considered the rank of the target pathways. An individual was fitter than another one if its average rank of the target pathways was lower. In reality, the more important distinction is between those individuals that rank the target pathways as significant and those that do not. An improvement in rank that still has the target pathways as not-significant is not really an increase in accuracy, and therefore it should not be represented as an increased fitness. Conversely, a decrease in ranking within the significance range may not be a decrease in accuracy, and therefore should not always be penalized as a decrease in fitness. However, using a step-like evaluation function based on the significance would have introduced abrupt changes that we think would have increased the difficulty of the genetic algorithm search.

Despite the limitations, the results obtained with this framework show the effectiveness of evolutionary computation techniques in the optimization of parameters in bioinformatic applications. This framework is general enough to be applied to a multitude of methods for the analysis of biological pathways, where parameters are often chosen arbitrarily.

## Part III

### Chapter 10 Conclusions and future work

Even though the advances in high-throughput technologies had made it easier than ever to perform comparison between two phenomena, like disease and healthy, our ability to understand the mechanisms behind these measurements are as challenging as ever. Part of the challenge lies in the intricate interactions between the disease and the immune system of the host. This points to the need of developing drugs tailored specific to individual patients (i.e., personalized medicine). This thesis aims to be a step towards this ultimate goal by first providing a methodology able to identify patients that are likely to require these specialized treatments and then provide methodologies to better understand the mechanism of the disease.

Part I was focused on improving our ability to classify and dichotomize the patient population. The classification method used throughout is the Support Vector Machines (SVMs), one of the most used classification techniques. First, we proposed an approach to increase the performance and the quality of posterior probability outputs for classifier ensembles. Even though classifier ensembles were shown to achieve better classification accuracy than single classifier, the existing methods to generate posterior probabilities for ensembles are rudimentary and either combine the base classifier probability outputs or calibrate the aggregated value. These approaches are computationally intensive because a posterior probability model needs to be fitted for each one of the base classifiers. We proposed a method, using bagging and  $z$ -score, that extracts additional information from the decision values of the base classi-

fiers to fit a single posterior probability model. To fit the probabilistic outputs to the  $z$ -score statistic, we considered three variations: *Gaussian CDF*, *logistic linear* and *Z-score Isotonic*. Based on our assessments over 12 datasets from the UCI machine learning repository, the *logistic linear* model is generally preferred because of the stable good performance, better performance at estimating posterior probability for samples prone to misclassification, and improved computational cost. Alternatively, the *Gaussian CDF Z-bag* model is easier to be implemented without further parameter training and it ranks as second best in terms of accuracy while estimation of posterior can be appropriate if some conditions met. The three different approaches proposed achieve comparable or better prediction accuracy and posterior probability estimation in comparison with the existing ensemble calibration methods while reducing computational cost.

Besides improving the performance of existing classifiers, we aimed to augment the ability of existing classifiers to identify potential disease subtypes in the study population. Existing classifiers, like the widely used SVMs, are unable to detect patient samples that come from a completely different distribution than the training set. We proposed a machine learning technique based on SVMs to automatically detect samples for which the prediction is unreliable because they are either too dissimilar from the two groups (e.g., disease types or sub-types) under study or they are too close to the decision hyperplane where a small change in the training parameters would switch their prediction. We should both on data from the UCI machine learning repository and gene expression data that our method is able to identify such subgroups. One of our experiments was to classifying leukemia patients based on gene expression data. During training, only patients with conventional acute lymphoblastic leukemia (ALL) and acute myelogenous leukemia (AML) were used to train both the classical SVM and our method - SVM with uncertainty (USVM). We tested the two models based on mixed-lineage leukemia (MLL) and the USVM model was able to identify 65% of these patients as being dissimilar from anything it was trained on, while, as expected, the classical SVM was unable to identify any.

With the methods presented in Part I, we are now able to classify better and refine our understanding of disease subtypes. A parallel step towards personalized medicine is to improve our ability to understand and characterize disease mechanism. Part II contains the methods we proposed towards this goal.

The first method was aimed at taking full advantage of the rich pathway databases and the gene interaction they describe. We proposed a technique able to include the reliability measurement for each gene in addition to other important factors like the type and position of each of the differentially expressed gene in the pathway, the magnitude of their expression change, and the type of interaction between all genes in the pathway. We showed that this methods achieves better performance than the classical methods that ignore this information on two gene expression data sets.

The second method is aimed at taking full advantage of the recent improvements in gene quantification technologies and expand the pathway analysis to a true genome-wide scale by incorporating all measured expression changes. The cut-off free impact analysis is the first method able to use all gene expression changes, as well as fully incorporate and exploit the topology of each pathway. We show in a multiple sclerosis study how this approach is able to obtain consistent results across multiple experiments performed by different groups on different technologies.

Moreover, we proposed an evolutionary computational technique that is able to set specific parameters in the cut-off free impact analysis. These parameters, like the contribution of a gene in a pathway, are hard to derive in practice. Using an objective evaluation function over 24 datasets we showed that running these parameters achieves better performance than both random and default parameters.

The pathway analysis methods have been made available as part of Bioconductor 2.12, in the package ROntoTools.<sup>1</sup>

---

<sup>1</sup><http://bioconductor.org/packages/2.12/bioc/html/ROntoTools.html>

The quest to personalized medicine is not over and additional steps and developments need to be taken. The machine learning technique proposed here is able to identify a subgroup that is dissimilar from what it was trained on. However, this is not sensitive enough to be able to identify the correct treatment for each individual patients. Additional divisions in the rejected samples and maybe even in the classes used for training need to be sought-after. The pathway analysis methods proposed here rely on gene expression data that is unable to capture post-translational modification like phosphorylation and methylation. Incorporating additional types of data [83, 178] is a key component in refining our understanding of disease mechanism for the individual patients.

## APPENDIX

### ROntoTools implementation as released as part of Bioconductor

#### 2.12:

```

> #' Pathway-Express result class
> #'
> #' This class is used to encode the results of the pathway analysis
> #' performed by the function \link{pe}.
> #'
> #' @details
> #'
> #' The slots input and ref record global information
> #' related to the whole analysis, while the pathways slot
> #' records the specific results as \link{pePathway} for each
> #' one of the pathways used in the analysis.
> #'
> #' @section Slots:
> #'
> #' \describe{
> #'   \item{pathways:}{A list of \link{pePathway}}
> #' objects.}
> #'   \item{input:}{named vector of fold changes used
> #' for the analysis. The names of the vector are the IDs
> #' originally used.}
> #'   \item{ref:}{character vector containing the IDs
> #' used as reference in the analysis.}
> #'   \item{cutOffFree:}{boolean value indicating if a
> #' cut-of-free analysis has been performed.}
> #' }
> #'
> #' @seealso \link{pe}, \link{pePathway}
> #'
> #' @aliases peRes-class
> #' @exportClass peRes
> setClass("peRes",
+         representation(pathways = "list",
+                         input = "numeric",
+                         ref = "character",
+                         cutOffFree = "logical"),
+         prototype(pathways = list(), cutOffFree = FALSE)
+ )
> #' Class that encodes the result of Pathway-Express for a single pathway
> #'

```



```

> #' @section Slots:
> #'
> #' \describe{
> #'   \item{\code{map}:}{an object of type graph (e.g.,
> #' \code{\link{graphNEL}}).}
> #'   \item{\code{input}:}{named vector of fold changes
> #' for genes on this pathway. The names of the genes are
> #' the original IDS used in the analysis}
> #'   \item{\code{ref}:}{vector of reference IDs on this
> #' pathway}
> #'   \item{\code{boot}:}{an object of class \code{boot}
> #' encoding the bootstrap information.}
> #'   \item{\code{Pert}:}{the gene perturbation factors for
> #' all genes on the pathway, as computed by Pathway-Express.}
> #'   \item{\code{Acc}:}{the gene accumulations for all
> #' genes on the pathway, as computed by Pathway-Express.}
> #' }
> #'
> #'
> #' @seealso \code{\link{pe}}, \code{\link{peRes}}
> #'
> #' @aliases pePathway-class
> #' @import graph
> #' @exportClass pePathway
> setClass("pePathway",
+         representation(map = "graph",
+                         input = "numeric",
+                         ref = "character",
+                         boot = "ANY",
+                         Pert = "numeric",
+                         Acc = "numeric"
+         ),
+         prototype(map = new("graphNEL")
+         )
+ )
> #' Retrieve the node weights of a graph
> #'
> #' @description
> #'
> #' A generic function that returns the node weights of a graph.
> #' If \code{index} is specified, only the weights of the specified
> #' nodes are returned. The user can control which node attribute
> #' is interpreted as the weight.
> #'
> #' @param object A graph, any object that inherits the \code{graph}

```

```

> #' class.
> #' @param index If supplied, a character or numeric vector of node
> #' names or indices.
> #' @param ... Unused.
> #' @param attr The name of the node attribute to use as a weight.
> #' You can view the list of defined node attributes and their default
> #' values using nodeDataDefaults.
> #' @param default The value to use if {object} has no node
> #' attribute named by the value of {attr}. The default is the
> #' value 1.
> #'
> #' @details
> #'
> #' The weights of all nodes identified by the {index} are returned.
> #' If {index} is not supplied, the weights of all nodes are returned.
> #'
> #' By default, {nodeWeights} looks for an node attribute with name
> #' "weight" and, if found, uses these values to construct the node weight
> #' vector.
> #' You can make use of attributes stored under a different name by
> #' providing a value for the {attr} argument.
> #' For example, if {object} is a graph instance with an node
> #' attribute named "WTS", then the call {nodeWeights(object,
> #' attr="WTS")} will attempt to use those values.
> #'
> #' If the graph instance does not have an node attribute with name given
> #' by the value of the {attr} argument, {default} will be used
> #' as the weight for all nodes.
> #' Note that if there is an attribute named by {attr}, then its
> #' default value will be used for nodes not specifically customized.
> #' See nodeData and nodeDataDefaults for more information.
> #'
> #' @return
> #'
> #' A named vector with the node weights. The names of the vector are
> #' the names of the specified {index}, or all nodes if {index}
> #' was not provided.
> #'
> #' @seealso
> #'
> #' {link{nodes}}, {link{nodeData}}
> #'
> #' @examples
> #'
> #' library(graph)

```

```

> #' V <- LETTERS[1:4]
> #' g <- graphNEL(nodes = V, edgemode = "directed")
> #' nodeWeights(g)
> #' nodeWeights(g, "B")
> #' nodeWeights(g, attr = "WT", default = 3)
> #'
> #' @rdname nodeWeights
> #'
> #' @export
> setGeneric("nodeWeights",
+           function(object, index, ..., attr="weight", default=1)
+           standardGeneric("nodeWeights")
+ )
> #' Summarize the results of a Pathway-Express analysis
> #'
> #' @usage Summary(x, pathNames = NULL, totalAcc = TRUE,
> #' totalPert = TRUE, normalize = TRUE,
> #' pPert = TRUE, pAcc = TRUE, pORA = TRUE,
> #' comb.pv = c("pPert", "pORA"), comb.pv.func = compute.fischer,
> #' order.by = "pComb", adjust.method = "fdr")
> #'
> #' @param x Pathways-Express result object obtained using
> #' \link{pe}
> #' @param pathNames named vector of pathway names;
> #' the names of the vector are the IDs of the pathways
> #' @param totalAcc boolean value indicating if the total
> #' accumulation should be computed
> #' @param totalPert boolean value indicating if the
> #' total perturbation should be computed
> #' @param normalize boolean value indicating if normalization
> #' with regards to the bootstrap simulations should be performed
> #' on totalAcc and totalPert
> #' @param pPert boolean value indicating if the significance of
> #' the total perturbation in regards to the bootstrap permutations
> #' should be computed
> #' @param pAcc boolean value indicating if the significance of
> #' the total accumulation in regards to the bootstrap permutations
> #' should be computed
> #' @param pORA boolean value indicating if the over-representation
> #' p-value should be computed
> #' @param comb.pv vector of the p-value names to be combine
> #' (any of the above p-values)
> #' @param comb.pv.func the function to combine the p-values;
> #' takes as input a vector of p-values and returns the combined p-value
> #' @param order.by the name of the p-value that is used to

```

```

> #' order the results
> #' @param adjust.method the name of the method to adjust
> #' the p-value (see \link{p.adjust})
> #'
> #' @seealso \code{\link{pe}}
> #'
> #' @examples
> #'
> #' # load experiment
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #' ref <- top$entrez
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #' kpg <- setEdgeWeights(kpg)
> #' kpg <- setNodeWeights(kpg, defaultWeight = 1)
> #'
> #' # perform the pathway analysis
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' # obtain summary of results
> #' head(Summary(peRes))
> #'
> #' kpn <- keggPathwayNames("hsa")
> #'
> #' head(Summary(peRes))
> #'
> #' head(Summary(peRes, pathNames = kpn, totalAcc = FALSE,
> #'             totalPert = FALSE, pAcc = FALSE, pORA = FALSE,
> #'             comb.pv = NULL, order.by = "pPert"))
> #'
> #' @rdname Summary-methods
> #'
> #' @aliases Summary.peRes
> #' @aliases Summary,peRes-method
> #' @export
> setMethod("Summary", c("x" = "peRes"),
+           function(x, pathNames = NULL, totalAcc = TRUE,
+                   totalPert = TRUE, normalize = TRUE,
+                   pPert = TRUE, pAcc = TRUE, pORA = TRUE,
+                   comb.pv = c("pPert", "pORA"),
+                   comb.pv.func = compute.fischer,

```

```

+         order.by = "pComb", adjust.method = "fdr")
+     {
+         ifelse <- function(test, trueCase, falseCase){
+             if(test) return(trueCase)
+             else return(falseCase)
+         }
+
+         pathStats <- function(pePath)
+         {
+             pStats <- NULL
+
+             pStats$totalAcc <- ifelse(totalAcc, get.totalAcc(pePath),
+                                     NULL)
+             pStats$totalPert <- ifelse(totalPert, get.totalPert(pePath),
+                                       NULL)
+
+             pStats$totalAccNorm <- ifelse(totalAcc & normalize,
+                                           get.totalAccNorm(pePath), NULL)
+             pStats$totalPertNorm <- ifelse(totalPert & normalize,
+                                           get.totalPertNorm(pePath), NULL)
+
+             pStats$pPert <- ifelse(pPert, compute.pPert(pePath), NULL)
+             pStats$pAcc <- ifelse(pAcc, compute.pAcc(pePath), NULL)
+
+             pStats$pORA <- ifelse(pORA & !x@cutOffFree,
+                                   compute.pORA(pePath, length(x@input),
+                                               length(x@ref)), NULL)
+
+             pStats$pComb <- ifelse(!is.null(comb.pv) &
+                                   !any(is.null(pStats[comb.pv])),
+                                   as.numeric(comb.pv.func(unlist(pStats[comb.pv]))), NULL)
+
+             return(unlist(pStats))
+         }
+
+         if (pORA & x@cutOffFree)
+         {
+             pORA <- FALSE
+             if ("pORA" %in% comb.pv)
+             {
+                 order.by <- setdiff(comb.pv, "pORA")[1]
+                 comb.pv <- NULL
+             }
+             message("The over-representaion p-value is not defined
+                    for cut-off free analysis and will not

```

```

+         be computed!")
+     }
+
+     if(!is.null(comb.pv))
+     {
+         if(!all(comb.pv %in% c("pPert", "pAcc", "pORA")))
+         {
+             warning("The p-value to be combined are not specified
+                 correctly. No combination p-value
+                 will be calculated!")
+             comb.pv <- NULL
+             if(order.by == "pComb")
+                 order.by <- NULL
+             }else{
+                 for(i in 1:length(comb.pv))
+                     assign(comb.pv[i], TRUE)
+             }
+         }
+
+     topStats <- data.frame(do.call(rbind, lapply(x@pathways,
+         pathStats)))
+
+     if(!is.null(pathNames))
+     {
+         pathNames <- pathNames[rownames(topStats)]
+         topStats <- cbind(pathNames, topStats)
+     }
+
+     if(order.by %in% colnames(topStats))
+     {
+         topStats <- topStats[order(topStats[,order.by]),]
+     }
+
+     allPVs <- c("pPert", "pAcc", "pORA", "pComb")
+
+     lapply(allPVs[allPVs %in% colnames(topStats)],
+         function(pv)
+             topStats[[paste(pv, ".", adjust.method, sep = "")]] <<-
+                 p.adjust(topStats[[pv]], adjust.method)
+         )
+     return(topStats)
+ }
+ )
> #' Set node weights
> #'

```

```

> #' @param graphList a list of \code{graph} (e.g., \code{\link{graphNEL}})
> #' objects
> #' @param weights named vector or matrix; if vector, the node is
> #' going to have the same weight in all graphs it appears;
> #' if matrix, the rows represent nodes and columns represent graphs
> #' and the node will have different weights in each pathway
> #' @param defaultWeight the default weight for all nodes not set
> #' by the parameter \code{weights}
> #'
> #' @return
> #'
> #' The \code{graphList} with the node weights set.
> #'
> #' @examples
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #'
> #' kpg <- setNodeWeights(kpg)
> #'
> #' nodeWeights(kpg[["path:hsa04110"]])
> #'
> #' @export
> setNodeWeights <- function(graphList, weights = NULL, defaultWeight = 1)
+ {
+   if(is.null(weights))
+   {
+     graphList <- lapply(graphList,
+       function(g)
+       {
+         nodeDataDefaults(g, "weight") <- defaultWeight
+         return(g)
+       })
+     return(graphList)
+   }
+   if(is.vector(weights))
+   {
+     graphList <- lapply(graphList,
+       function(g)
+       {
+         i <- nodes(g)[nodes(g) %in% names(weights)]
+
+         nodeDataDefaults(g, "weight") <- defaultWeight
+
+         if(length(i) != 0)

```

```

+             nodeData(g, i, "weight") <- weights[i]
+
+             return(g)
+         })
+     return(graphList)
+ }
+ if(is.matrix(weights))
+ {
+     graphList <- Recall(graphList, numeric(0), defaultWeight)
+
+
+     gi <- names(graphList)[names(graphList) %in% colnames(weights)]
+
+     if(length(gi) == 0)
+         return(graphList)
+
+     for(i in 1:length(gi))
+     {
+
+         w <- weights[,gi[i]]
+         names(w) <- rownames(weights)
+
+
+         g <- graphList[[gi[i]]]
+
+         j <- nodes(g)[nodes(g) %in% names(w)]
+
+         nodeDataDefaults(g, "weight") <- defaultWeight
+
+         if(length(j) != 0)
+             nodeData(g, j, "weight") <- w[j]
+
+         graphList[gi[i]] <- g
+     }
+
+     return(graphList)
+ }
+ return(NULL)
+ }
> #' Set gene weights based on edge type
> #'
> #' \code{setEdgeWeights}
> #'
> #' @param graphList a list of \code{\link{graphNEL}} objects

```



```

> #' @param edgeTypeAttr edge attribute to be considered
> #' as the edge type. If the edge has multiple types,
> #' the edge type attribute is considered as a comma separated
> #' list of types
> #' @param edgeWeightByType named list of weights, where
> #' the names of the list are the
> #' edge type (values of the attribute defined by \code{edgeTypeAttr})
> #' @param defaultWeight default value for an edge with
> #' a type not defined in \code{edgeWeightByType}
> #' @param combineWeights for the edges with multiple types,
> #' the function to be applied on the vector of weights
> #' @param nodeOnlyGraphs boolean value marking if graphs
> #' with no edges should be returned or not; note that graphs with
> #' all edge weights equal to 0 are considered node only graphs
> #'
> #' @return
> #'
> #' The \code{graphList} with the edge weights set.
> #'
> #' @examples
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #'
> #' kpg <- setEdgeWeights(kpg)
> #'
> #' edgeWeights(kpg[["path:hsa04110"]])
> #'
> #' @export
setEdgeWeights <- function(graphList,
+                           edgeTypeAttr = "subtype",
+                           edgeWeightByType = list(
+                             activation = 1, inhibition = -1,
+                             expression = 1, repression = -1),
+                           defaultWeight = 0,
+                           combineWeights = sum,
+                           nodeOnlyGraphs = FALSE)
+ {
+   graphList <- lapply(graphList,
+                       function(g)
+                         {
+                           gftM <- graph2ftM(g)
+
+                           if(nrow(gftM) == 0)
+                             return(g)

```

```

+
+     weights <- sapply(strsplit(unlist(edgeData(g,
+     gftM[,1], gftM[,2], edgeTypeAttr)), ","),
+     function(x){
+         weightNames <- names(edgeWeightByType)[
+             match(x, names(edgeWeightByType))]
+         weightNames <- weightNames[!is.na(weightNames)]
+
+         if(length(weightNames) == 0)
+             return(defaultWeight)
+
+         return(combineWeights(unlist(edgeWeightByType[
+             weightNames])))
+     })
+
+     suppressWarnings(g <- addEdge(gftM[,1], gftM[,2], g, weights))
+
+     return(g)
+ })
+
+ if (!nodeOnlyGraphs)
+     return(graphList[sapply(graphList, function(x)
+         length(unlist(edges(x)))) != 0])
+
+ return(graphList)
+ }
> #' @import KEGGREST
> #' @import KEGGgraph
> loadKEGGpathwayDataREST <- function(organism = "hsa",
+     updateCache = FALSE,
+     verbose = TRUE)
+ {
+     dfUnparsed <- paste(system.file("extdata",package="ROntoTools"),
+         "/KEGGRESTunparsed_",organism, ".RData", sep = "")
+
+     if (!file.exists(dfUnparsed) || updateCache)
+     {
+         pathList <- keggList("pathway", organism)
+
+         if (verbose)
+         {
+             message("Downloading pathway data:")
+             pb <- txtProgressBar(min = 0, max = length(pathList)-1, style = 3)
+         }
+     }
+ }
+

```

```

+   tmpDir <- tempfile("ROntoTools", tempdir())
+   dir.create(tmpDir)
+
+
+   allPathwayInfo <- lapply(names(pathList),
+                             function(pathID) {
+
+                                 p <- keggGet(pathID, "kgml")
+
+                                 pKgml <- file.path(tmpDir,
+                                                     paste(strsplit(pathID, ":")[[1]][2], ".kgml",
+                                                       sep = ""))
+
+                                 write(p, pKgml)
+
+                                 pathData <- parseKGML(pKgml)
+
+                                 file.remove(pKgml)
+
+                                 if(verbose)
+                                   setTxtProgressBar(pb, getTxtProgressBar(pb) + 1)
+                                 return(pathData)
+                             })
+   dbInfo <- keggInfo("pathway")
+
+   file.remove(tmpDir)
+
+   save(allPathwayInfo, dbInfo, file = dfUnparsed)
+ }else{
+   load(dfUnparsed)
+   message(paste("Using cached pathway data.
+                 Database info:\n", dbInfo, sep = ""))
+ }
+
+ return(allPathwayInfo)
+ }
> #' Download and parse KEGG pathway data
> #'
> #' @param organism organism code as defined by KEGG
> #' @param targRelTypes target relation types
> #' @param relPercthresh percentage of the number of relation
> #' types over all possible relations in the pathway
> #' @param nodeOnlyGraphs allow graphs with no edges
> #' @param updateCache re-download KEGG data
> #' @param verbose show progress of downloading and parsing

```

```

> #'
> #' @return
> #'
> #' A list of \link{graphNEL} objects encoding the pathway
> #' information.
> #'
> #' @seealso \link{keggPathwayNames}
> #'
> #' @examples
> #'
> #' # The pathway cache provided as part of the pathway contains only the
> #' # pathways that passed the default filtering. We recommend,
> #' # re-downloading the pathways using the updateCache parameter
> #' kpg <- keggPathwayGraphs("hsa")
> #'
> #' # to update the pathway cache for human run:
> #' # kpg <- keggPathwayGraphs("hsa", updateCache = TRUE)
> #' # this is time consuming and depends on the available bandwidth.
> #'
> #' head(names(kpg))
> #'
> #' kpg[["path:hsa04110"]]
> #' head(nodes(kpg[["path:hsa04110"]]))
> #' head(edges(kpg[["path:hsa04110"]]))
> #'
> #' @importMethodsFrom KEGGgraph
> #' @export
> keggPathwayGraphs <- function(organism = "hsa",
+                               targRelTypes = c("GRel", "PRel", "PPrel"),
+                               relPercThresh = 0.9,
+                               nodeOnlyGraphs = FALSE,
+                               updateCache = FALSE,
+                               verbose = TRUE)
+ {
+   defaultParameters <- FALSE
+   if ((organism == "hsa") & all(targRelTypes == c("GRel", "PRel",
+           "PPrel"))) & (relPercThresh == 0.9) & (nodeOnlyGraphs == FALSE))
+     defaultParameters <- TRUE
+
+   allPathwayInfo <- loadKEGGpathwayDataREST(organism, updateCache,
+       verbose)
+
+   if (defaultParameters & !updateCache)
+   {
+     message("Default parameters detected. Using pre-parsed data.")

```

```

+   load(paste(system.file("extdata",package="ROntoTools"),
+     "/kpgDefault.RData", sep = ""))
+   return(pathwayGraphs)
+ }
+
+ l <- lapply(allPathwayInfo, function(path) {
+   l <- sapply(edges(path), getType)
+   if(length(l) == 0)
+     return(0)
+   t <- table(l)
+   return(t)
+ })
+
+ allRelTypes <- unique(unlist(lapply(l, names)))
+
+ counts <- do.call(rbind,lapply(l, function(x)
+   as.vector(x[allRelTypes])))
+ colnames(counts) <- allRelTypes
+
+ accIndex <- rowSums(counts[,targRelTypes], na.rm=T) /
+   rowSums(counts, na.rm=T) >= relPercThresh
+ accIndex[is.na(accIndex)] <- FALSE
+ allPathwayInfo <- allPathwayInfo[accIndex]
+
+ names(allPathwayInfo) <- sapply(allPathwayInfo, getName)
+
+ if (verbose)
+ {
+   message("Parsing pathway data:")
+   pb <- txtProgressBar(min = 0, max = length(allPathwayInfo)-1,
+     style = 3)
+ }
+
+ pathwayGraphs <- lapply(allPathwayInfo, function(g)
+ {
+   g <- KEGGgraph::KEGGpathway2Graph(g)
+   kg <- new("graphNEL", nodes(g), edges(g), edgemode = "directed")
+
+   if (length(getKEGGedgeData(g)) == 0)
+   {
+     if (verbose)
+       setTxtProgressBar(pb, getTxtProgressBar(pb) + 1)
+     return(NULL)
+   }
+ }
+
+ edgeDataDefaults(kg, "subtype") <- NA

```

```

+
+   relGeneTable <- data.frame(cbind(
+     do.call(rbind, strsplit(names(getKEGGedgeData(g)), '~')),
+     sapply(getKEGGedgeData(g), function(e)
+       paste(lapply(getSubtype(e), getName), collapse=","))
+   ), stringsAsFactors = FALSE)
+   names(relGeneTable) <- c("from","to","subtype")
+
+   edgeData(kg, relGeneTable$from, relGeneTable$to, "subtype") <-
+     relGeneTable$subtype
+
+   if (verbose)
+     setTxtProgressBar(pb, getTxtProgressBar(pb) + 1)
+
+   return(kg)
+ })
+
+ pathwayGraphs <- pathwayGraphs[!sapply(pathwayGraphs, is.null)]
+
+ if (defaultParameters)
+   save(pathwayGraphs, paste(system.file("extdata",package="ROntoTools"),
+     "/kpgDefault.RData", sep = ""))
+
+ return(pathwayGraphs)
+ }
> #' Obtain KEGG pathway titles
> #'
> #' @param organism organism code as defined by KEGG
> #' @param updateCache re-download KEGG data
> #' @param verbose show progress of downloading and parsing
> #'
> #' @return
> #'
> #' A named vector of pathway titles. The names of the vector
> #' are the pathway KEGG IDs.
> #'
> #' @seealso {\link{keggPathwayGraphs}}
> #'
> #' @examples
> #'
> #' kpn <- keggPathwayNames("hsa")
> #'
> #' # to update the pathway cache for human run:
> #' # kpn <- keggPathwayNames("hsa", updateCache = TRUE)
> #' # this is time consuming and depends on the available bandwidth.

```

```

> #'
> #' head(kpn)
> #'
> #' @import KEGGgraph
> #' @export
> keggPathwayNames <- function(organism = "hsa",
+                               updateCache = FALSE,
+                               verbose = TRUE)
+ {
+   allPathwayInfo <- loadKEGGpathwayDataREST(organism, updateCache,
+       verbose)
+
+   allNames <- sapply(allPathwayInfo, getTitle)
+
+   names(allNames) <- sapply(allPathwayInfo, getName)
+
+   return(allNames)
+ }
> #' @rdname nodeWeights
> #' @aliases nodeWeights,graph,character-method
> #' @import graph
> #' @export
> setMethod("nodeWeights", signature(object="graph", index="character"),
+           function (object, index, attr, default)
+           {
+             if (!is.character(attr) || length(attr) != 1)
+               stop("'attr' must be character(1)")
+             if (!attr %in% names(nodeDataDefaults(object))) {
+               nodeDataDefaults(object, attr) <- default
+             }
+             nw <- nodeData(object, index, attr = attr)
+             if (length(nw) != 0)
+               return(unlist(nw))
+             else
+               return(numeric(0))
+           }
+ )
> #' @rdname nodeWeights
> #' @aliases nodeWeights,graph,numeric-method
> #' @import graph
> #' @export
> setMethod("nodeWeights", signature(object="graph", index="numeric"),
+           function (object, index, attr, default)
+           {
+             index <- nodes(object)[index]

```

```

+         nodeWeights(object, index, attr = attr, default = default)
+     })
> #' @rdname nodeWeights
> #' @aliases nodeWeights,graph,missing-method
> #' @import graph
> #' @export
> setMethod("nodeWeights", signature(object="graph", index="missing"),
+         function (object, index, attr, default)
+         {
+             index <- nodes(object)
+             nodeWeights(object, index, attr = attr, default = default)
+         })#' Pathway-Express: Pathway analysis of signaling pathways
> #'
> #' @param x named vector of log fold changes for the
> #' differentially expressed genes; names(x) must use
> #' the same id's as ref and the nodes of the graphs
> #' @param graphs list of pathway graphs as objects of type
> #' graph (e.g., link{graphNEL}); the graphs must
> #' be weighted graphs (i.e., have an attribute weight
> #' for both nodes and edges)
> #' @param ref the reference vector for all genes in the analysis;
> #' if the reference is not provided or it is identical to names(x)
> #' a cut-off free analysis is performed
> #' @param nboot number of bootstrap iterations
> #' @param verbose print progress output
> #' @param cluster a cluster object created by makeCluster for
> #' parallel computations
> #' @param seed an integer value passed to set.seed() during
> #' the bootstrap permutations
> #'
> #' @details
> #'
> #' See details in the cited articles.
> #'
> #' @return
> #'
> #' An object of class link{peRes}.
> #'
> #' @references
> #'
> #' Voichita C., Donato M., Draghici S.: "Incorporating gene
> #' significance in the impact analysis of signaling pathways",
> #' IEEE Machine Learning and Applications (ICMLA), 2012 11th
> #' International Conference on, Vol. 1, p.126-131, 2012
> #'

```



```

> #' Tarca AL., Draghici S., Khatri P., Hassan SS., Kim J., Kim CJ.,
> #' Kusanovic JP., Romero R.: "A Signaling Pathway Impact Analysis
> #' for Microarray Experiments", 2008, Bioinformatics, 2009, 25(1):75-82.
> #'
> #' Khatri P., Draghici S., Tarca AL., Hassan SS., Romero R.:
> #' "A system biology approach for the steady-state analysis of
> #' gene signaling networks". Progress in Pattern Recognition,
> #' Image Analysis and Applications, Lecture Notes in Computer
> #' Science. 4756:32-41, November 2007.
> #'
> #' Draghici S., Khatri P., Tarca A.L., Amin K., Done A., Voichita C.,
> #' Georgescu C., Romero R.: "A systems biology approach for
> #' pathway level analysis". Genome Research, 17, 2007.
> #'
> #' @seealso \code{\link{Summary}}, \code{\link{plot.pRes}},
> #' \code{\link{keggPathwayGraphs}}, \code{\link{setNodeWeights}},
> #' \code{\link{setEdgeWeights}}
> #'
> #' @examples
> #'
> #' # load a multiple sclerosis study (public data available in
> #' # Array Express ID: E-GEOD-21942)
> #' # This file contains the top table, produced by the limma package with
> #' # added gene information. All the probe sets with no gene associate to
> #' # them, have been removed. Only the most significant probe set for
> #' # each gene has been kept (the table is already ordered by p-value)
> #' # The table contains the expression fold change and significance of
> #' # each probe set in peripheral blood mononuclear cells (PBMC) from
> #' # 12 MS patients and 15 controls.
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #'                 package = "ROntoTools"))
> #' head(top)
> #'
> #' # select differentially expressed genes at 1% and save their fold
> #' # change in a vector fc and their p-values in a vector pv
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #'
> #' pv <- top$P.Value[top$adj.P.Val <= .01]
> #' names(pv) <- top$entrez[top$adj.P.Val <= .01]
> #'
> #' # alternativly use all the genes for the analysis
> #' # NOT RUN:
> #' # fc <- top$logFC
> #' # names(fc) <- top$entrez

```

```

> #'
> #' # pv <- top$P.Value
> #' # names(pv) <- top$entrez
> #'
> #' # get the reference
> #' ref <- top$entrez
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #'
> #' # set the beta information (see the cited documents for
> #' # meaning of beta)
> #' kpg <- setEdgeWeights(kpg)
> #'
> #' # include the significance information in the analysis (see
> #' # Voichita:2012 for more information)
> #' # set the alpha information based on the pv with one of the
> #' # predefined methods
> #' kpg <- setNodeWeights(kpg, weights = alphaMLG(pv), defaultWeight = 1)
> #'
> #' # perform the pathway analysis
> #' # in order to obtain accurate results the number of bootstraps, nboot,
> #' # should be increase to a number like 2000
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' # obtain summary of results
> #' head(Summary(peRes))
> #'
> #' @export
> pe <- function(x, graphs, ref = NULL, nboot = 2000, verbose = TRUE,
+   cluster = NULL, seed = NULL)
+ {
+   cutOffFree <- FALSE
+   if (is.null(ref))
+   {
+     ref <- names(x)
+     cutOffFree <- TRUE
+   }
+   else
+   {
+     if (!any(is.na(match(ref,names(x)))))
+     {
+       warning("All the reference IDs are part of the input.
+         Cut-off free analysis is performed.")
+       ref <- names(x)

```

```

+     cutOffFree <- TRUE
+   }
+ }
+
+ preservedSeed <- NULL
+ if (exists(".Random.seed"))
+   preservedSeed <- .Random.seed
+
+ if(!is.null(seed))
+   set.seed(seed)
+
+ if (any(is.na(match(names(x), ref))))
+ {
+   warning("There are input IDs not available in the reference.
+           These will be excluded from analysis.")
+   x <- x[!is.na(match(names(x), ref))]
+ }
+
+ peRes <- pf.helper(x = x, ref = ref, graphs = graphs, nboot = nboot,
+                   verbose = verbose, cluster = cluster, seed = seed)
+ peRes@cutOffFree <- cutOffFree
+
+ if(is.null(preservedSeed))
+ {
+   if(!is.null(seed))
+     rm(.Random.seed)
+ }
+ else
+   .Random.seed <- preservedSeed
+
+ return(peRes)
+ }
> #' @import parallel
> pf.helper <- function(x, graphs, ref = NULL, nboot = 2000,
+                       verbose = TRUE, cluster = NULL, seed = NULL)
+ {
+   if(verbose)
+   {
+     message("Performing pathway analysis...")
+     if(is.null(cluster))
+     {
+       pb <- txtProgressBar(min = 0, max = length(graphs), style = 3)
+     }
+   }
+ }
+
+

```

```

+ t1 <- Sys.time()
+ if (is.null(cluster))
+ {
+   allBoot <- lapply(graphs, function(g, seed) {
+     if(!is.null(seed))
+       set.seed(seed)
+     ret <- pe.boot(g, x = x, ref = ref, nboot = nboot)
+     if(verbose)
+       setTxtProgressBar(pb, getTxtProgressBar(pb) + 1)
+     return(ret)
+   }, seed = seed)
+ }
+ else
+ {
+   clusterExport(cluster, c("pe.boot", "compute.inverse", "compute.B"))
+   clusterEvalQ(cluster, library(ROntoTools))
+
+   allBoot <- parLapply(cluster, graphs, function(g, seed) {
+     if(!is.null(seed))
+       set.seed(seed)
+     ret <- pe.boot(g, x = x, ref = ref, nboot = nboot)
+     return(ret)
+   }, seed = seed)
+
+ }
+ t2 <- Sys.time()
+
+ if(verbose)
+ {
+   message("Analysis completed in ", format(t2-t1), ".")
+ }
+
+ allBoot <- allBoot[!sapply(allBoot, is.null)]
+
+ peRes <- new("peRes", pathways = allBoot, input = x, ref = ref)
+
+ return(peRes)
+ }
> #' @import boot
> #' @keywords internal
> pe.boot <- function(g, x, ref, nboot, all.genes = F)
+ {
+   inv <- compute.inverse(compute.B(g))
+
+   pePath <- new("pePathway",

```

```

+           map = g,
+           input = x[names(x) %in% nodes(g)],
+           ref = ref[ref %in% nodes(g)]
+
+   if (is.null(inv) | (length(pePath@input) == 0))
+     return(NULL)
+
+   # same number of DE genes at any position in the pathway
+   # (given by the gene from the pathway in the reference)
+   ran.gen.de <- function(x, l) {
+     y <- sample(l$fc, length(x))
+     names(y) <- sample(l$ref, length(x))
+     return(y)
+   }
+
+   pePath@boot <- boot(pePath@input,
+     function(x, inv) {
+       xx <- rep(0, nrow(inv)); names(xx) <- rownames(inv);
+       xx[names(x)] <- x
+       xx <- xx * nodeWeights(pePath@map, names(xx))
+       tt = inv %*% xx;
+       ret <- c(sum(abs(tt-xx)), sum(abs(tt)))
+       names(ret) <- c("tAcc", "tPert")
+       return(ret)
+     },
+     nboot,
+     "parametric", ran.gen = ran.gen.de, mle =
+     list(ref = pePath@ref, fc = as.numeric(x)),
+     inv = inv
+   )
+   colnames(pePath@boot$t) <- names(pePath@boot$t0)
+
+   xx <- rep(0, nrow(inv))
+   names(xx) <- rownames(inv)
+   xx[names(pePath@input)] <- pePath@input
+   pePath@Pert = (inv %*% xx)[,1];
+   pePath@Acc = pePath@Pert - xx
+
+   return(pePath)
+ }
> compute.inverse <- function(M, eps = 1e-5)
+ {
+   if ( abs(det(M)) >= eps )
+   {
+     s = svd(M);

```

```

+   inv = s$v %*% diag(1/s$d) %*% t(s$u)
+   rownames(inv) <- colnames(inv) <- rownames(M)
+   return(inv)
+ }else{
+   return(NULL)
+ }
+ }
> compute.B <- function(g, non.zero = TRUE)
+ {
+   if (non.zero)
+     # number of downstream genes (like in SPIA)
+     nds <- sapply(edgeWeights(g), function(x) sum(x != 0 ))
+   else
+     nds <- sapply(edges(g), length)
+
+   # add 1 for all genes that do not have downstream genes to avoid
+   # division by 0 this does not affect the computation
+   nds[nds == 0] <- 1
+
+   # compute B = (I - beta/nds)
+   #B <- t(diag(length(nodes(g)))) - as(g, "matrix") / nds)
+   B <- diag(length(nodes(g))) - t(as(g, "matrix")) /
+     matrix(nds, byrow=TRUE, nrow = length(nds), ncol = length(nds))
+
+   return(B)
+ }
> compute.pAcc <- function(pePath)
+   ifelse( !all(pePath@boot$t[, "tAcc"] == 0),
+     compute.bootPV(pePath@boot$t0["tAcc"], pePath@boot$t[, "tAcc"]),
+     NA)
> compute.pPert <- function(pePath)
+   ifelse( !all(pePath@boot$t[, "tPert"] == 0),
+     compute.bootPV(pePath@boot$t0["tPert"],
+       pePath@boot$t[, "tPert"]), NA)
> compute.pORA <- function(pePath, inputSize, refSize)
+   phyper(q = length(pePath@input)-1,
+     m = length(pePath@ref),
+     n = refSize-length(pePath@ref),
+     k = inputSize,
+     lower.tail = FALSE)
> get.totalAcc <- function(pePath)
+   as.numeric(pePath@boot$t0["tAcc"])
> get.totalAccNorm <- function(pePath)
+   ifelse( !all(pePath@boot$t[, "tAcc"] == 0),
+     as.numeric((pePath@boot$t0["tAcc"] -

```

```

+             mean(pePath@boot$t[, "tAcc"])) /
+             sd(pePath@boot$t[, "tAcc"))),
+         NA)
> get.totalPert <- function(pePath)
+   as.numeric(pePath@boot$t0["tPert"])
> get.totalPertNorm <- function(pePath)
+   ifelse( !all(pePath@boot$t[, "tPert"] == 0),
+         as.numeric((pePath@boot$t0["tPert"] -
+                     mean(pePath@boot$t[, "tPert"])) /
+                     sd(pePath@boot$t[, "tPert"])),
+         NA)
> #' Plot pathway level statistics
> #'
> #' @description Display graphical representation of pathway level
> #' statistic like:
> #' i) two way comparison between the measured expression change and
> #' one of the factors computed by Pathway-Express (\link{pe}) or
> #' ii) the bootstrap statistics of the same factors.
> #'
> #'
> #' @param x an object of type \link{pePathway}
> #' @param y if provided, the factor to be plotted (either \code{Acc}
> #' (default) or \code{Pert}; see \link{pePathway})
> #' @param ... Arguments to be passed to methods,
> #' such as \link{par}
> #' @param type type of plot (either \code{two.way} (default) or
> #' \code{boot})
> #' @param eps any value smaller than this will be plotted as 0
> #'
> #' @seealso \link{pe}, \link{plot.peRes},
> #' \link{peNodeRenderInfo}, \link{peEdgeRenderInfo}
> #'
> #' @examples
> #'
> #' # load experiment
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #' ref <- top$entrez
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #' kpg <- setEdgeWeights(kpg)
> #' kpg <- setNodeWeights(kpg, defaultWeight = 1)

```

```

> #'
> #' perform the pathway analysis (for more accurate results use
> #' nboot = 2000)
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' plot(peRes@@pathways[[50]])
> #'
> #' plot(peRes@@pathways[[50]], "Pert", main = "Perturbation factor")
> #'
> #' plot(peRes@@pathways[[50]], type = "boot")
> #'
> #' plot(peRes@@pathways[[50]], "Pert", type = "boot",
> #' main = "Perturbation factor")
> #'
> #' @rdname plot.pePathway-methods
> #' @name plot.pePathway
> #'
> #' @aliases plot.pePathway
> #' @aliases plot,pePathway,missing-method
> #' @export
> setMethod("plot", signature(x="pePathway", y="missing"),
+           function(x, y, ..., type = "two.way", eps = 1e-6)
+           {
+             plot(x, y = "Acc", ..., type = type, eps = eps)
+           }
+ )
> #' @rdname plot.pePathway-methods
> #'
> #' @aliases plot.pePathway
> #' @aliases plot,pePathway,character-method
> #' @export
> setMethod("plot", signature(x="pePathway", y="character"),
+           function(x, y, main = "", ... , type = "two.way", eps = 1e-6)
+           {
+             if (!(y %in% c("Acc", "Pert")))
+               stop("Undefined slot selected: ", y, ".")
+
+             switch(type,
+                   two.way={
+                     iy <- y
+
+                     extInput <- rep(0, length(slot(x, iy)))
+                     names(extInput) <- names(slot(x, iy))
+                     extInput[names(x@input)] <- x@input
+

```



```

+         cl <- rep("black", length(slot(x, iy)))
+         cl[abs(slot(x, iy)) >= eps] <- "green"
+         cl[abs(extInput) >= eps] <- "blue"
+         cl[abs(slot(x, iy)) >= eps & abs(extInput) >=
+             eps] <- "red"
+
+         plot(slot(x, iy), extInput, pch = 16, xlab = y,
+             ylab = "Log2 FC", main = main, ...)
+         abline(v=0,h=0, lwd = .5)
+         points(slot(x, iy), extInput, pch = 16, col = cl)
+
+         return(invisible())
+     },
+     boot={
+         iy <- paste("t", y, sep = "")
+
+         tB <- x@boot$t0[iy]
+         allB <- x@boot$t[,iy]
+
+         tB <- (tB - mean(allB)) / sd(allB)
+         allB <- (allB - mean(allB)) / sd(allB)
+
+         plot(density(allB), xlab = y, main = main, ...)
+         abline(v=0, lwd = .5)
+         abline(v=tB, lwd = 1, col = "red")
+
+         return(invisible())
+     }
+ )
+     stop(type, " is not a valid plot type.")
+ }
+ )
> #' Plot Pathway-Express result
> #'
> #' @description Display a two-way plot using two of the p-values
> #' from the Pathway-Express analysis.
> #'
> #' @param x an object of type \link{peRes}
> #' @param y vector of two p-values names to be combined using
> #' \code{comb.pv.func} (default: \code{c("pAcc", "pORA")}).
> #' @param ... Arguments to be passed to methods, such as
> #' \code{\link{par}}.
> #' @param comb.pv.func the function to combine the p-values -
> #' takes as input a vector of p-values

```

```

> #' and returns the combined p-value (default:
> #' \link{compute.fischer}).
> #' @param adjust.method the name of the method to adjust the
> #' p-value (see \link{p.adjust})
> #' @param threshold corrected p-value threshold
> #' @param eps any value smaller than this will be considered as
> #' {eps} (default: {1e-6}).
> #'
> #' @seealso \link{pe}, \link{Summary.peRes},
> #' \link{plot.pePathway}
> #'
> #' @examples
> #'
> #' # load experiment
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #' ref <- top$entrez
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #' kpg <- setEdgeWeights(kpg)
> #' kpg <- setNodeWeights(kpg, defaultWeight = 1)
> #'
> #' perform the pathway analysis (for more accurate results use
> #' nboot = 2000)
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' plot(peRes)
> #'
> #' plot(peRes, c("pPert","pORA"), comb.pv.func = compute.normalInv,
> #' threshold = .01)
> #'
> #' @rdname plot.peRes-methods
> #' @name plot.peRes
> #'
> #' @aliases plot.peRes
> #' @aliases plot,peRes,missing-method
> #' @export
> setMethod("plot", signature(x="peRes", y="missing"),
+         function(x, y, ... , comb.pv.func = compute.fischer,
+                 adjust.method = "fdr",
+                 threshold = .05, eps = 1e-6)
+         {

```

```

+         plot(x, y = c("pAcc", "pORA"), ... , comb.pv.func = comb.pv.func,
+             adjust.method = adjust.method,
+             threshold = threshold, eps = eps)
+     }
+ )
> #' @rdname plot.peRes-methods
> #'
> #' @aliases plot.peRes
> #' @aliases plot,peRes,character-method
> #' @export
> setMethod("plot", signature(x="peRes", y="character"),
+         function(x, y, ... , comb.pv.func = compute.fischer,
+             adjust.method = "fdr", threshold = .05, eps = 1e-6)
+         {
+
+             st <- Summary(x, comb.pv = y,
+                 comb.pv.func = comb.pv.func,
+                 adjust.method = adjust.method)
+             st <- st[!is.na(st[, paste("pComb", adjust.method,
+                 sep = ".")]),]
+
+             st[,y[1]][st[,y[1]] <= eps] <- eps
+             st[,y[2]][st[,y[2]] <= eps] <- eps
+
+             i <- st[, paste("pComb", adjust.method, sep = ".")] <=
+                 threshold
+             thr.comb <- mean(min(st[!i,"pComb"]),max(st[i,"pComb"]))
+
+             xrange <- c(min(-log(st[,y[1]])), max(-log(st[,y[1]])))
+             xrange[2] <- xrange[2] + (xrange[2]-xrange[1]) * .1
+             yrange <- c(min(-log(st[,y[2]])), max(-log(st[,y[2]])))
+             yrange[2] <- yrange[2] + (yrange[2]-yrange[1]) * .1
+
+             i <- seq(xrange[1], xrange[2], length.out=200)
+             j <- seq(yrange[1], yrange[2], length.out=200)
+             expGrid <- expand.grid(i,j)
+             z <- apply(1/exp(expGrid), 1, comb.pv.func) <= thr.comb
+
+             plot(c(min(-log(st[,y[1]])),min(-log(st[,y[2]]))),
+                 xlab = y[1], ylab = y[2],
+                 xlim = xrange,
+                 ylim = yrange,
+                 col = "white", ...)
+             nonSig <- expGrid[!z,][chull(expGrid[!z,]),]
+             sig <- expGrid[z,][chull(expGrid[z,]),]

```

```

+
+     polygon(nonSig, col="gray90")
+     polygon(sig, col="lightcyan")
+
+     i <- st[, paste("pComb", adjust.method, sep = ".")] <=
+         threshold
+
+     points(-log(st[,y[1]]), -log(st[,y[2]]), xlab = y[1],
+           ylab = y[2], pch = 19)
+     if (any(i))
+     {
+         points(-log(st[,y[1]])[i], -log(st[,y[2]])[i], pch = 21,
+               bg = "red")
+         text(-log(st[,y[1]])[i], -log(st[,y[2]])[i] - .5,
+              labels = rownames(st)[i],
+              cex = .75)
+     }
+
+     i <- st[, paste(y[1], adjust.method, sep = ".")] <= threshold
+     if (any(i))
+     {
+         thr1 <- mean(min(st[!i,y[1]]),max(st[i,y[1]]))
+         abline(v = -log(thr1), col = "red", lwd = 2, lty = 2)
+     }
+
+     i <- st[, paste(y[2], adjust.method, sep = ".")] <= threshold
+     if (any(i))
+     {
+         thr2 <- mean(min(st[!i,y[2]]),max(st[i,y[2]]))
+         abline(h = -log(thr2), col = "red", lwd = 2, lty = 2)
+     }
+ }
+ )
> #' Extract edge render information from a \code{pePathway} object
> #'
> #' @param x an object of class \code{\link{pePathway}}
> #' @param pos.col color of the edges with possitive weight
> #' @param pos.lty line type of the edges with possitive weight
> #' @param pos.ah arrow head of the edges with possitive weight
> #' @param neg.col color of the edges with negative weight
> #' @param neg.lty line type of the edges with negative weight
> #' @param neg.ah arrow head of the edges with negative weight
> #' @param zero.col color of the edges with zero weight
> #' @param zero.lty color of the edges with zero weight
> #' @param zero.ah color of the edges with zero weight

```

```

> #'
> #' @value a named list as expected by \code{\link{edgeRenderInfo}}
> #'
> #' @seealso \code{\link{edgeRenderInfo}}, \code{\link{par}}
> #'
> #' @examples
> #'
> #' # load experiment
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #' ref <- top$entrez
> #'
> #' # load the set of pathways
> #' kpg <- keggPathwayGraphs("hsa")
> #' kpg <- setEdgeWeights(kpg)
> #' kpg <- setNodeWeights(kpg, defaultWeight = 1)
> #'
> #' # perform the pathway analysis
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' p <- peRes@@pathways[[50]]
> #' g <- layoutGraph(p@@map, layoutType = "dot")
> #' graphRenderInfo(g) <- list(fixedsize = FALSE)
> #' edgeRenderInfo(g) <- peEdgeRenderInfo(p)
> #' nodeRenderInfo(g) <- peNodeRenderInfo(p)
> #' # notice the different type of edges in the graph (solid/dashed
> #' # /dotted)
> #' renderGraph(g)
> #'
> #' @export
> peEdgeRenderInfo <- function(x,
+   pos.col = "black", pos.lty = "solid", pos.ah = "vee",
+   neg.col = "black", neg.lty = "dashed", neg.ah = "tee",
+   zero.col = "lightgray", zero.lty = "dotted", zero.ah = "none")
+ {
+   stopifnot(class(x) == "pePathway")
+
+   ew <- unlist(edgeWeights(x@map))
+
+   aHead <- rep(zero.ah, length(edgeNames(x@map)))
+   names(aHead) <- edgeNames(x@map)
+   aHead[ew > 0] <- pos.ah
+   aHead[ew < 0] <- neg.ah

```

```

+   aHead <- aHead[setdiff(seq(along=ew), removedEdges(x@map))]
+
+   eCol <- rep(zero.col, length(edgeNames(x@map)))
+   names(eCol) <- edgeNames(x@map)
+   eCol[ew > 0] <- pos.col
+   eCol[ew < 0] <- neg.col
+   eCol <- eCol[setdiff(seq(along=ew), removedEdges(x@map))]
+
+   eStyle <- rep(zero.lty, length(edgeNames(x@map)))
+   names(eStyle) <- edgeNames(x@map)
+   eStyle[ew > 0] <- pos.lty
+   eStyle[ew < 0] <- neg.lty
+   eStyle <- eStyle[setdiff(seq(along=ew), removedEdges(x@map))]
+
+   return(list(
+     arrowhead = aHead,
+     col = eCol,
+     lty = eStyle
+   ))
+ }
> #' Extract node render information from a {pePathway} object
> #'
> #' @param x an object of class {\link{pePathway}}
> #' @param y a string representing the factor to be represented
> #' ({Pert, Acc} or {input}; see {\link{pePathway}})
> #' @param input.shape shape of nodes that have measured expression change
> #' @param default.shape shape of all other nodes
> #' @param pos.col color of nodes with a positive {y} factor
> #' @param neg.col color of nodes with a negative {y} factor
> #' @param zero.col color of nodes with the {y} factor equal to zero
> #'
> #' @value a named list as expected by {\link{nodeRenderInfo}}
> #'
> #' @seealso {\link{nodeRenderInfo}}, {\link{par}}
> #'
> #' @examples
> #'
> #' # load experiment
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #' fc <- top$logFC[top$adj.P.Val <= .01]
> #' names(fc) <- top$entrez[top$adj.P.Val <= .01]
> #' ref <- top$entrez
> #'
> #' # load the set of pathways

```

```

> #' kpg <- keggPathwayGraphs("hsa")
> #' kpg <- setEdgeWeights(kpg)
> #' kpg <- setNodeWeights(kpg, defaultWeight = 1)
> #'
> #' # perform the pathway analysis
> #' peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
> #'
> #' p <- peRes@@pathways[[50]]
> #' g <- layoutGraph(p@@map, layoutType = "dot")
> #' graphRenderInfo(g) <- list(fixedsize = FALSE)
> #' edgeRenderInfo(g) <- peEdgeRenderInfo(p)
> #' nodeRenderInfo(g) <- peNodeRenderInfo(p)
> #' # notice the different type of nodes in the graph (box/circle)
> #' # the color of each node represents the perturbation
> #' (red = positive, blue = negative)
> #' # the shade represents the strength of the perturbation
> #' renderGraph(g)
> #'
> #' nodeRenderInfo(g) <- peNodeRenderInfo(p, "Acc")
> #' # now, the color of each node represents the accumulation
> #' (red = positive, blue = negative)
> #' # notice that square nodes with no parents have no accumulation
> #' renderGraph(g)
> #'
> #' @export
> peNodeRenderInfo <- function(x, y = "Pert",
+                               input.shape = "box",
+                               default.shape = "ellipse",
+                               pos.col = "red",
+                               neg.col = "blue",
+                               zero.col = "white")
+ {
+   stopifnot(class(x) == "pePathway")
+   stopifnot(y %in% c("input", "Pert", "Acc"))
+
+   nShape <- rep(default.shape, length(nodes(x@map)))
+   names(nShape) <- nodes(x@map)
+   nShape[names(x@input)] <- input.shape
+
+   nFillColor <- rep(zero.col, length(nodes(x@map)))
+   names(nFillColor) <- nodes(x@map)
+   pf <- slot(x, y)
+   nFillColor[pf <= 0] <- colorRampPalette(c(zero.col, neg.col))(256)[
+     as.numeric(cut(abs(pf[pf<=0]), 256))]
+   nFillColor[pf >= 0] <- colorRampPalette(c(zero.col, pos.col))(256)[

```

```

+         as.numeric(cut(abs(pf[pf>=0]), 256))
+
+   return(list(
+     shape = nShape,
+     fill = nFillColor
+   ))
+ }#' @keywords internal
> addNames <- function(x, nms)
+ {
+   if (length(x) == length(nms))
+     names(x) <- nms
+
+   x <- rep(x, length(nms))
+   names(x) <- nms
+
+   return(x)
+ }
> #' @keywords internal
> compute.bootPV <- function(real, dist)
+   ( sum(abs(dist - mean(dist)) >
+     abs(real - mean(dist))) + 1 ) / (1 + length(dist))
> #' Combine independent p-values using the Fischer method
> #'
> #' @param p a vector of independent p-values
> #' @param eps the minimal p-value considered
> #' (all p-values smaller will be set to this value)
> #'
> #' @value the combined p-value
> #'
> #' @seealso \link{pe}, \link{compute.normalInv}
> #'
> #' @examples
> #'
> #' p <- c(.1, .01)
> #' compute.fischer(p)
> #'
> #' @export
> compute.fischer <- function(p, eps = 1e-6)
+ {
+   stopifnot(any(p >= 0 & p<=1))
+   p[p < eps] <- eps
+
+   k <- prod(p);
+   return(k-k*log(k))
+ }

```



```

> #' Combine independent p-values using the normal inversion method
> #'
> #' @param p a vector of independent p-values
> #' @param eps the minimal p-value considered
> #' (all p-values smaller will be set to this value)
> #'
> #' @value the combined p-value
> #'
> #' @seealso \link{pe}, \link{compute.fischer}
> #'
> #' @examples
> #'
> #' p <- c(.1, .01)
> #' compute.normalInv(p)
> #'
> #' @export
> compute.normalInv <- function(p, eps = 1e-6)
+ {
+   stopifnot(any(p >= 0 & p<=1))
+   p[p < eps] <- eps
+   return(pnorm(sum(sapply(p, qnorm)) / sqrt(length(p))))
+ }
> #' @keywords internal
> graph2ftM <- function(g)
+ {
+   ind <- which(as.vector(as(g, "matrix"))!=0)
+
+   from <- (ind-1) %% length(nodes(g)) + 1
+   to <- (ind-1) %% length(nodes(g)) + 1
+
+   return(cbind(nodes(g)[from], nodes(g)[to]))
+ }
> #' Compute alpha weights
> #'
> #' Transform a vector of p-values into weights.
> #'
> #' @details
> #'
> #' Computes a set of weights from p-values using the formula
> #' \code{1-pv/threshold}.
> #'
> #' @param pv vector of p-values
> #' @param threshold the threshold value that was used to select DE genes
> #'
> #' @seealso \link{pe}

```

```

> #'
> #' @examples
> #'
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #'
> #' head(alpha1MR(top$adj.P.Val))
> #'
> #' @export
> alpha1MR <- function(pv, threshold = max(pv))
+ {
+   if (!is.numeric(pv))
+     stop("pv is not numeric")
+   if (threshold == 0)
+     stop("threshold cannot be 0")
+
+   return(1-pv/threshold)
+ }
> #' Compute alpha weights
> #'
> #' Transform a vector of p-values into weights.
> #'
> #' @details
> #'
> #' Computes a set of weights from p-values using the formula
> #'  $\{-\log_{10}(pv/threshold)\}$ .
> #'
> #' @param pv vector of p-values
> #' @param threshold the threshold value that was used to select DE genes
> #'
> #' @seealso \link{pe}
> #'
> #' @examples
> #'
> #' load(system.file("extdata/E-GEOD-21942.topTable.RData",
> #' package = "ROntoTools"))
> #'
> #' head(alphaMLG(top$adj.P.Val))
> #'
> #' @export
> alphaMLG <- function(pv, threshold = max(pv))
+ {
+   if (!is.numeric(pv))
+     stop("pv is not numeric")
+   if (threshold == 0)

```

```
+     stop("threshold cannot be 0")  
+  
+     return(-log10(pv/threshold))  
+ }
```

## BIBLIOGRAPHY

- [1] M. Ackermann and K. Strimmer. A general modular framework for gene set enrichment analysis. *BMC Bioinformatics*, 10(1):47, 2009.
- [2] F. Al-Shahrour, R. Diaz-Uriarte, and J. Dopazo. FatiGO: a web tool for finding significant associations of Gene Ontology terms with groups of genes. *Bioinformatics*, 20(4):578–580, 2004.
- [3] K. Ali and M. Pazzani. Error reduction through learning multiple descriptions. *Machine Learning*, 24(3):173–202, 1996.
- [4] S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer. MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics*, 30(1):41–47, 2001.
- [5] R. Arriagada, B. Bergman, T. L. Chevalier, J.-P. Pignon, and J. Vansteenkiste. Cisplatin-based adjuvant chemotherapy in patients with completely resected non-small-cell lung cancer. *New England Journal of Medicine*, 350(4):351–360, 2004.
- [6] D. Ayres-de Campos, J. Bernardes, A. Garrido, J. Marques-de Sa, and L. Pereira-Leite. Sisporto 2.0: a program for automated analysis of cardiocograms. *Journal of Maternal-Fetal and Neonatal Medicine*, 9(5):311–318, 2000.
- [7] K. Bache and M. Lichman. UCI machine learning repository. <http://archive.ics.uci.edu/ml>, 2013.
- [8] T. Barrett, D. Troup, S. Wilhite, P. Ledoux, C. Evangelista, I. Kim, M. Tomashevsky, K. Marshall, K. Phillippy, P. Sherman, et al. NCBI GEO: archive for functional genomics data sets - 10 years on. *Nucleic Acids Research*, 39 (suppl 1):D1005–D1010, 2011.
- [9] P. L. Bartlett and M. H. Wegkamp. Classification with a reject option using a hinge

- loss. *Journal of Machine Learning Research*, 9:1823–1840, 2008.
- [10] D. G. Beer, S. L. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas, M. L. Lizyness, R. Kuick, S. Hayasaka, J. M. Taylor, M. D. Iannettoni, M. B. Orringer, and S. Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8(8):816–824, 2002.
- [11] T. Beissbarth and T. Speed. GOstat: find statistically overrepresented gene ontologies within a group of genes. *Bioinformatics*, 20(9):1464–1465, 2004.
- [12] G. F. Berriz, O. D. King, B. Bryant, C. Sander, and F. P. Roth. Characterizing gene sets with FuncAssociate. *Bioinformatics*, 19(18):2502–2504, 2003.
- [13] L. Beviglia, V. Golubovskaya, L. Xu, X. Yang, R. J. Craven, and W. G. Cance. Focal adhesion kinase N-terminus in breast carcinoma cells induces rounding, detachment and apoptosis. *Biochem J*, 373(1):201–210, 2003.
- [14] BioCarta. Biocarta - charting pathways of life. <http://www.biocarta.com>.
- [15] A. Bleyer and H. G. Welch. Effect of three decades of screening mammography on breast-cancer incidence. *New England Journal of Medicine*, 367(21):1998–2005, 2012.
- [16] P. Boinee, A. De Angelis, and G. L. Foresti. Ensembling classifiers-an application to image data classification from cherenkov telescope experiment. *World Academy of Science, Engineering and Technology*, 12:66–70, 2005.
- [17] C. Booth and F. Shepherd. Adjuvant chemotherapy for resected non-small cell lung cancer. *Journal of Thoracic Oncology*, 1(2):180–187, 2006.
- [18] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory, Proceedings of the 5th annual workshop on*, 144–152. ACM, 1992.
- [19] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
- [20] L. Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California, Berkeley, CA, USA, 1996.

- [21] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- [22] T. Breslin, M. Krogh, C. Peterson, and C. Troein. Signal transduction pathway profiling of individual tumor samples. *BMC Bioinformatics*, 6:163, 2005.
- [23] R. D. Canales, Y. Luo, J. C. Willey, B. Austermler, C. C. Barbacioru, C. Boysen, K. Hunkapiller, R. V. Jensen, C. R. Knight, K. Y. Lee, Y. Ma, B. Maqsodi, A. Papallo, E. H. Peters, K. Poulter, P. L. Ruppel, R. R. Samaha, L. Shi, W. Yang, L. Zhang, and F. M. Goodsaid. Evaluation of DNA microarray results with quantitative gene expression platforms. *Nature Biotechnology*, 24(9):1115–1122, 2006.
- [24] V. Carey. Machine learning concepts and tools for statistical genomics. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, 273–292. Springer, 2005.
- [25] C. I. Castillo-Davis and D. L. Hartl. GeneMerge-post-genomic analysis, data mining, and hypothesis testing. *Bioinformatics*, 19(7):891–892, 2003.
- [26] E. G. Cerami, B. E. Gross, E. Demir, I. Rodchenkov, Ö. Babur, N. Anwar, N. Schultz, G. D. Bader, and C. Sander. Pathway commons, a web resource for biological pathway data. *Nucleic Acids Research*, 39(suppl 1):D685–D690, 2011.
- [27] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.
- [28] G.-J. J. Chang, B. S. Davis, A. R. Hunt, D. A. Holmes, and G. Kuno. Flavivirus DNA vaccines. *Annals of the New York Academy of Sciences*, 951(1):272–285, 2001.
- [29] A. Chatr-Aryamontri, B.-J. Breitkreutz, S. Heinicke, L. Boucher, A. Winter, C. Stark, J. Nixon, L. Ramage, N. Kolas, and L. O’Donnell. The biogrid interaction database: 2013 update. *Nucleic Acids Research*, 41(D1):D816–D823, 2013.
- [30] Z. Chen, T. B. Gibson, F. Robinson, L. Silvestro, G. Pearson, B. e Xu, A. Wright, C. Vanderbilt, and M. H. Cobb. MAP kinases. *Chemical Reviews*, 101(8):2449–2476, 2001.
- [31] C. Chow. On optimum recognition error and reject tradeoff. *Information Theory*,

- IEEE Transactions on*, 16(1):41–46, 1970.
- [32] G. A. Churchill. Fundamentals of experimental design for cDNA microarrays. *Nature Genetics*, 32(Suppl):490–495, 2002.
- [33] P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Machine learning (EWSL), Proceedings of the European Working Session on*, 151–163. Springer, 1991.
- [34] R. Clarke, M. Liu, K. Bouker, Z. Gu, R. Lee, Y. Zhu, T. Skaar, B. Gomez, K. O’Brien, and Y. Wang. Antiestrogen resistance in breast cancer and the role of estrogen receptor signaling. *Oncogene*, 22(47):7316–7339, 2003.
- [35] B. P. Coe, W. W. Lockwood, L. Girard, R. Chari, C. Macaulay, S. Lam, A. F. Gazdar, J. D. Minna, and W. L. Lam. Differential disruption of cell cycle pathways in small cell and non-small cell lung cancer. *Br J Cancer*, 94(12):1927–1935, 2006.
- [36] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
- [37] D. Cortés Rivera, R. Landa Becerra, and C. A. Coello Coello. Cultural algorithms, an alternative heuristic to solve the job shop scheduling problem. *Engineering Optimization*, 39(1):69–85, 2007.
- [38] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
- [39] D. Croft, G. OKelly, G. Wu, R. Haw, M. Gillespie, L. Matthews, M. Caudy, P. Garapati, G. Gopinath, B. Jassal, et al. Reactome: a database of reactions, pathways and biological processes. *Nucleic Acids Research*, 39(suppl 1):D691–D697, 2011.
- [40] M. DeGroot and S. Fienberg. The comparison and evaluation of forecasters. *The Statistician*, 32(1-2):12–22, 1983.
- [41] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.
- [42] G. Dennis Jr., B. T. Sherman, D. A. Hosack, J. Yang, W. Gao, H. C. Lane, and R. A.

- Lempicki. DAVID: Database for annotation, visualization, and integrated discovery. *Genome Biology*, 4(5):P3, 2003.
- [43] V. Desai, P. Khatri, A. Done, A. Friedman, M. Tainsky, and S. Draghici. A novel bioinformatics technique for predicting condition-specific transcription factor binding sites. In *Computational Intelligence in Bioinformatics and Computational Biology, Proceedings of IEEE Symposium on*, 2005.
- [44] M. Donato and S. Draghici. Signaling pathways coupling phenomena. In *Neural Networks (IJCNN), The International Joint Conference on*, 2010.
- [45] M. Donato, Z. Zhu, A. Tomoiaga, P. Westfall, R. Romero, and S. Draghici. A method for analysis and correction of cross-talk effects in pathway analysis. In *Neural Networks (IJCNN), The International Joint Conference on*, 2012.
- [46] S. W. Doniger, N. Salomonis, K. D. Dahlquist, K. Vranizan, S. C. Lawlor, and B. R. Conklin. MAPPFinder: using gene ontology and genmapp to create a global gene expression profile from microarray data. *Genome Biology*, 4(1):R7, 2003.
- [47] S. Draghici. Statistical intelligence: effective analysis of high-density microarray data. *Drug Discovery Today*, 7(11):S55–S63, 2002.
- [48] S. Draghici, P. Khatri, P. Bhavsar, A. Shah, S. A. Krawetz, and M. A. Tainsky. Onto-tools, the toolkit of the modern biologist: Onto-express, onto-compare, onto-design and onto-translate. *Nucleic Acids Research*, 31(13):3775–81, 2003.
- [49] S. Draghici, P. Khatri, C. A. Eklund, and Z. Szallasi. Reliability and reproducibility issues in DNA microarray measurements. *Trends Genet*, 22(2):101–109, 2006.
- [50] S. Draghici, P. Khatri, R. P. Martins, G. C. Ostermeier, and S. A. Krawetz. Global functional profiling of gene expression. *Genomics*, 81(2):98–104, 2003.
- [51] S. Draghici, P. Khatri, A. Shah, and M. Tainsky. Assessing the functional bias of commercial microarrays using the Onto-Compare database. *BioTechniques, Microarrays and Cancer: Research and Applications*:55–61, 2003.
- [52] S. Draghici, P. Khatri, A. L. Tarca, K. Amin, A. Done, C. Voichita, C. Georgescu, and



- R. Romero. A systems biology approach for pathway level analysis. *Genome Research*, 17(10):1537–1545, 2007.
- [53] S. Draghici, S. Sellamuthu, and P. Khatri. Babel’s tower revisited: a universal resource for cross-referencing across annotation databases. *Bioinformatics*, 22(23):2934–2939, 2006.
- [54] I. Drozdov, C. Ouzounis, A. Shah, and S. Tsoka. Functional genomics assistant (FUGA): a toolbox for the analysis of complex biological networks. *BMC Research Notes*, 4(1):462, 2011.
- [55] B. Dutta, A. Wallqvist, and J. Reifman. Pathnet: A tool for pathway analysis using topological information. *Source Code for Biology and Medicine*, 7(1):10, 2012.
- [56] R. Edgar, M. Domrachev, and A. Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210, 2002.
- [57] B. Efron. Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.
- [58] B. Efron and R. Tibshirani. On testing the significance of sets of genes. *The Annals of Applied Statistics*, 1(1):107–129, 2007.
- [59] S. Efroni, C. Schaefer, and K. Buetow. Identification of key processes underlying cancer phenotypes using biologic pathway analysis. *PLoS ONE*, 2(5):e425, 2007.
- [60] Z. Fang, W. Tian, and H. Ji. A network-based gene-weighting approach for pathway analysis. *Cell Research*, 22:565–580, 2011.
- [61] F. Farfán, J. Ma, M. Sartor, G. Michailidis, and H. Jagadish. THINK Back: KNowledge-based Interpretation of High Throughput data. *BMC Bioinformatics*, 13(Suppl 2):S4, 2012.
- [62] R. Fisher. On the interpretation of  $\chi^2$  from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94, 1922.
- [63] R. Fisher. *Statistical methods for research workers*. Genesis Publishing Pvt Ltd, 1925.
- [64] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals of*

- Eugenics*, 7(2):179–188, 1936.
- [65] A. Franceschini, D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonovic, A. Roth, J. Lin, P. Minguez, P. Bork, C. von Mering, et al. String v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Research*, 41(D1):D808–D815, 2013.
- [66] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.
- [67] A. Frolkis, C. Knox, E. Lim, T. Jewison, V. Law, D. D. Hau, P. Liu, B. Gautam, S. Ly, A. C. Guo, et al. Smpdb: the small molecule pathway database. *Nucleic Acids Research*, 38(suppl 1):D480–D487, 2010.
- [68] M. Fukata and M. Abreu. Role of toll-like receptors in gastrointestinal malignancies. *Oncogene*, 27(2):234–243, 2008.
- [69] H. K. Gan, B. You, G. R. Pond, and E. X. Chen. Assumptions of expected benefits in randomized phase iii trials evaluating systemic treatments for cancer. *Journal of the National Cancer Institute*, 104(8):590–598, 2012.
- [70] K. Gandhi, F. McKay, M. Cox, C. Riveros, N. Armstrong, R. Heard, S. Vucic, D. Williams, J. Stankovich, M. Brown, et al. The multiple sclerosis whole blood mrna transcriptome and genetic associations indicate dysregulation of specific t cell pathways in pathogenesis. *Human Molecular Genetics*, 19(11):2134–2143, 2010.
- [71] S. Gao and X. Wang. TAPPA: topological analysis of pathway phenotype association. *Bioinformatics*, 23(22):3100–3102, 2007.
- [72] G. Ge and W. G. Wong. Classification of premalignant pancreatic cancer mass-spectrometry data using decision tree ensembles. *BMC Bioinformatics*, 9:275, 2008.
- [73] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. Yang, and J. Zhang. Bioconductor: open software de-

- velopment for computational biology and bioinformatics. *Genome Biology*, 5(10):R80, 2004.
- [74] F. Gilli, R. Lindberg, P. Valentino, F. Marnetto, S. Malucchi, A. Sala, M. Capobianco, A. Di Sapia, F. Sperli, L. Kappos, et al. Learning from nature: pregnancy changes the expression of inflammation-related genes in patients with multiple sclerosis. *PLoS ONE*, 5(1):e8962, 2010.
- [75] E. Glaab, A. Baudot, N. Krasnogor, R. Schneider, and A. Valencia. Enrichnet: network-based gene set enrichment analysis. *Bioinformatics*, 28(18):i451–i457, 2012.
- [76] E. Glaab, A. Baudot, N. Krasnogor, and A. Valencia. TopoGSA: network topological gene set analysis. *Bioinformatics*, 26(9):1271–1272, 2010.
- [77] J. J. Goeman, S. A. van de Geer, F. de Kort, and H. C. van Houwelingen. A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, 20(1):93–99, 2004.
- [78] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6:333–362, 1992.
- [79] V. Golubovskaya, L. Beviglia, L. H. Xu, r. Earp, H. S., R. Craven, and W. Cance. Dual inhibition of focal adhesion kinase and epidermal growth factor receptor pathways cooperatively induces death receptor-mediated apoptosis in human breast cancer cells. *J Biol Chem*, 277(41):38978–38987, 2002.
- [80] C. Gondro and B. Kinghorn. A simple genetic algorithm for multiple sequence alignment. *Genet. Mol. Res*, 6(4):964–982, 2007.
- [81] Y. Grandvalet, A. Rakotomamonjy, J. Keshet, and S. Canu. Support vector machines with a reject option. In *Neural Information Processing Systems (NIPS)*, 537–544, 2008.
- [82] Z. Gu, J. Liu, K. Cao, J. Zhang, and J. Wang. Centrality-based pathway enrichment: a systematic approach for finding significant pathways dominated by key genes. *BMC Systems Biology*, 6(1):56, 2012.
- [83] C. Guichard, G. Amaddeo, S. Imbeaud, Y. Ladeiro, L. Pelletier, I. B. Maad, J. Calder-

- aro, P. Bioulac-Sage, M. Letexier, F. Degos, et al. Integrated analysis of somatic mutations and focal copy-number changes identifies key genes and pathways in hepatocellular carcinoma. *Nature Genetics*, 44(6):694–698, 2012.
- [84] B. J. Haas, M. C. Zody, et al. Advancing RNA-seq analysis. *Nature Biotechnology*, 28(5):421, 2010.
- [85] L. Hansen and P. Salamon. Neural network ensembles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(10):993–1001, 1990.
- [86] R. Herbei and M. Wegkamp. Classification with reject option. *Canadian Journal of Statistics*, 34(4):709–721, 2006.
- [87] R. V. Hogg and A. T. Craig. *Introduction to Mathematical Statistics*. New York: Macmillan, 1978.
- [88] C. Hoimes and W. Kelly. Redefining hormone resistance in prostate cancer. *Therapeutic Advances in Medical Oncology*, 2(2):107–123, 2010.
- [89] Y. Hong, K. S. Ho, K. W. Eu, and P. Y. Cheah. A susceptibility gene set for early onset colorectal cancer that integrates diverse signaling pathways: implication for tumorigenesis. *Clin Cancer Res*, 13(4):1107–1114, 2007.
- [90] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer. COPASI—a complex pathway simulator. *Bioinformatics*, 22(24):3067–3074, 2006.
- [91] D. A. Hosack, G. Dennis Jr., B. T. Sherman, H. C. Lane, and R. A. Lempicki. Identifying biological themes within lists of genes with EASE. *Genome Biology*, 4(6):P4, 2003.
- [92] Q. Hu, Z. He, Z. Zhang, and Y. Zi. Fault diagnosis of rotating machinery based on improved wavelet package transform and SVMs ensemble. *Mechanical Systems and Signal Processing*, 21:688–705, 2007.
- [93] M. Huaiyu and P. Thomas. Panther pathway: An ontology-based pathway database coupled with data analysis tools. protein networks and pathway analysis. *Methods in*

- Molecular Biology*, 563:123–140, 2009.
- [94] D. W. Huang, B. T. Sherman, and R. A. Lempicki. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic Acids Research*, 37(1):1–13, 2009.
- [95] J. Hung, T. Whitfield, T. Yang, Z. Hu, Z. Weng, C. DeLisi, et al. Identification of functional modules that correlate with phenotypic difference: the influence of network topology. *Genome Biology*, 11(2):R23, 2010.
- [96] J.-H. Hung, T.-H. Yang, Z. Hu, Z. Weng, and C. DeLisi. Gene set enrichment analysis: performance evaluation and usage guidelines. *Briefings in Bioinformatics*, 13(3):281–291, 2012.
- [97] M. A.-H. Ibrahim, S. Jassim, M. A. Cawthorne, and K. Langlands. A topology-based score for pathway enrichment. *Journal of Computational Biology*, 19(5):563–573, 2012.
- [98] R. Irizarry, B. Hobbs, F. Collin, Y. Beazer-Barclay, K. Antonellis, U. Scherf, and T. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2003.
- [99] R. Irizarry, C. Wang, Y. Zhou, and T. Speed. Gene set enrichment analysis made simple. *Johns Hopkins University, Dept. of Biostatistics Working Papers*, 2009.
- [100] S. Isci, C. Ozturk, J. Jones, and H. Otu. Pathway analysis of high-throughput biological data within a bayesian network framework. *Bioinformatics*, 27(12):1667–1674, 2011.
- [101] L. Jacob, P. Neuvial, and S. Dudoit. Gains in power from structured two-sample tests of means on graphs. *U.C. Berkeley Division of Biostatistics Working Paper Series*, 2010.
- [102] G. Joshi-Tope, M. Gillespie, I. Vasrik, P. D’Eustachio, E. Schmidt, B. de Bone, B. Jassal, G. R. Gopinath, G. R. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Research*, 33(Database issue):D428–D432, 2005.
- [103] M. Kanehisa and S. Goto. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic*

- Acids Research*, 28(1):27–30, 2000.
- [104] M. Kanehisa, S. Goto, S. Kawashima, and A. Nakaya. The KEGG databases at GenomeNet. *Nucleic Acids Research*, 30(1):42–46, 2002.
- [105] M. Kanehisa, S. Goto, S. Kawashima, Y. Okunom, and M. Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Research*, 32(Database issue):277–280, 2004.
- [106] T. Kelder, M. P. van Iersel, K. Hanspers, M. Kutmon, B. R. Conklin, C. T. Evelo, and A. R. Pico. Wikipathways: building research communities on biological pathways. *Nucleic Acids Research*, 40(D1):D1301–D1307, 2012.
- [107] A. Kemppinen, J. Kaprio, A. Palotie, and J. Saarela. Systematic review of genome-wide expression studies in multiple sclerosis. *BMJ Open*, 1(1):e000053, 2011.
- [108] P. Khatri, P. Bhavsar, G. Bawa, and S. Draghici. Onto-tools: an ensemble of web-accessible, ontology-based tools for the functional design and interpretation of high-throughput gene expression experiments. *Nucleic Acids Research*, 32:W449–W456, 2004.
- [109] P. Khatri, V. Desai, A. L. Tarca, S. Sellamuthu, D. E. Wildman, R. Romero, and S. Draghici. New Onto-Tools: Promoter-Express, nsSNPCounter, and Onto-Translate. *Nucleic Acids Research*, 34:W626–W631, 2006.
- [110] P. Khatri and S. Draghici. Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 21(18):3587–3595, 2005.
- [111] P. Khatri, S. Draghici, G. C. Ostermeier, and S. A. Krawetz. Profiling gene expression using Onto-Express. *Genomics*, 79(2):266–270, 2002.
- [112] P. Khatri, S. Draghici, A. L. Tarca, S. S. Hassan, and R. Romero. A systems biology approach for the steady-state analysis of gene signaling networks. In *Pattern Recognition, Proceedings of the 12th Iberoamerican Congress on*, 32–41, 2007.
- [113] P. Khatri, S. Sellamuthu, P. Malhotra, K. Amin, A. Done, and S. Draghici. Recent additions and improvements to the Onto-Tools. *Nucleic Acids Research*, 33(Web server

- issue):W762–W765, 2005.
- [114] P. Khatri, M. Sirota, and A. J. Butte. Ten Years of Pathway Analysis: Current Approaches and Outstanding Challenges. *PLoS Computational Biology*, 8(2):e1002375, 2012.
- [115] P. Khatri, C. Voichita, K. Kattan, N. Ansari, A. Khatri, C. Georgescu, A. L. Tarca, and S. Draghici. Onto-Tools: New additions and improvements in 2006. *Nucleic Acids Research*, 37(Web Server issue):W206–W211, 2007.
- [116] H. Kim, S. Pang, H. Je, D. Kim, and S. Bang. Constructing support vector machine ensemble. *Pattern Recognition*, 36:2757–2767, 2003.
- [117] M. J. Kim and D. K. Kang. Ensemble with neural networks for bankruptcy prediction. *Expert Syst. Appl.*, 37:3373–3379, 2010.
- [118] L. Kuncheva. *Combining pattern classifiers: methods and algorithms*. Wiley-Interscience, 2004.
- [119] M. Laakso and S. Hautaniemi. Integrative platform to translate gene sets to networks. *Bioinformatics*, 26(14):1802–1803, 2010.
- [120] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In *Research and development in information retrieval, Proceedings of the 19th annual international ACM SIGIR conference on*, SIGIR '96, 289–297. ACM, 1996.
- [121] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [122] H.-S. Lin, H. S. Talwar, A. L. Tarca, A. Ionan, M. Chatterjee, B. Ye, J. Wojciechowski, S. Mohapatra, M. D. Basson, G. H. Yoo, B. Peshek, F. Lonardo, C.-J. G. Pan, A. J. Folbe, S. Draghici, J. Abrams, and M. A. Tainsky. Autoantibody approach for serum-based detection of head and neck cancer. *Cancer Epidemiol Biomarkers Prev*, 16(11):2396–2405, 2007.
- [123] J. Luo, N. L. Solimini, and S. J. Elledge. Principles of cancer therapy: oncogene and non-oncogene addiction. *Cell*, 136(5):823–837, 2009.

- [124] W. Luo and C. Brouwer. Pathview: an r/bioconductor package for pathway based data integration and visualization. *Bioinformatics*, 29(14):1830–1831, 2013.
- [125] D. Martin, C. Brun, E. Remy, P. Mouren, D. Thieffry, and B. Jacq. GOToolBox: functional analysis of gene datasets based on gene ontology. *Genome Biology*, 5(12):R101, 2004.
- [126] F. Martinelli-Boneschi, C. Fenoglio, P. Brambilla, M. Sorosina, G. Giacalone, F. Esposito, M. Serpente, C. Cantoni, E. Ridolfi, M. Rodegher, et al. MicroRNA and mRNA expression profile screening in multiple sclerosis patients to unravel novel pathogenic steps and identify potential biomarkers. *Neuroscience letters*, 508(1):4–8, 2012.
- [127] M. Massa, M. Chiogna, and C. Romualdi. Gene set analysis exploiting the topology of a pathway. *BMC Systems Biology*, 4(1):121, 2010.
- [128] J. Mazieres, B. He, L. You, Z. Xu, and D. M. Jablons. Wnt signaling in lung cancer. *Cancer Lett*, 222(1):1–10, 2005.
- [129] R. Menon, M. Di Dario, C. Cordiglieri, S. Musio, L. La Mantia, C. Milanese, A. Di Stefano, M. Crabbio, D. Franciotta, R. Bergamaschi, et al. Gender-based blood transcriptomes and interactomes in multiple sclerosis: Involvement of sp1 dependent gene transcription. *Journal of Autoimmunity*, 38(2-3):J144–J155, 2011.
- [130] J. Mieczkowski, K. Swiatek-Machado, and B. Kaminska. Identification of pathway deregulation—gene expression based analysis of consistent signal transduction. *PLoS ONE*, 7(7):e41541, 2012.
- [131] V. K. Mootha, C. M. Lindgren, K.-F. Eriksson, A. Subramanian, S. Sihag, J. Lehar, P. Puigserver, E. Carlsson, M. Ridderstråle, E. Laurila, N. Houstis, M. J. Daly, N. Patterson, J. P. Mesirov, T. R. Golub, P. Tamayo, B. Spiegelman, E. S. Lander, J. N. Hirschhorn, D. Altshuler, and L. C. Groop. PGC-1 alpha-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nature Genetics*, 34(3):267–273, 2003.
- [132] S. Mukherjee, P. Tamayo, D. Slonim, A. Verri, T. Golub, J. Mesirov, and T. Poggio.



- Support vector machine classification of microarray data. *CBCL Paper*, 182, 1999.
- [133] M. M. Nau, B. J. Brooks, J. Battay, E. Sausville, A. F. Gazdar, I. R. Kirsch, O. W. McBride, V. Bertness, G. F. Hollis, and J. D. Minna. L-myc, a new myc-related gene amplified and expressed in human small cell lung cancer. *Nature*, 318(6041):69–73, 1985.
- [134] A. Niculescu-Mizil and R. Caruana. Obtaining calibrated probabilities from boosting. In *Uncertainty in Artificial Intelligence (UAI), Proceedings of the 12th Annual Conference on*, 413–420. AUAI Press, 2005.
- [135] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *Machine learning, Proceedings of the 22nd International Conference on*, 625–632. ACM, 2005.
- [136] D. Nikolic-Vukosavljevic, N. Todorovic-Rakovic, M. Demajo, V. Ivanovic, B. Neskovic, M. Markicevic, and Z. Neskovic-Konstantinovic. Plasma tgf-beta1-related survival of postmenopausal metastatic breast cancer patients. *Clin Exp Metastasis*, 21(7):581–585, 2004.
- [137] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, et al. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 27(1):29–34, 1999.
- [138] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.
- [139] D. Pan, N. Sun, K. H. Cheung, Z. Guan, L. Ma, M. Holford, X. Deng, and H. Zhao. PathMAPA: a tool for displaying gene expression and performing statistical tests on metabolic pathways at multiple levels for Arbidopsis. *BMC Bioinformatics*, 4(1):56, 2003.
- [140] K. H. Pan, C. J. Lih, and S. N. Cohen. Effects of threshold choice on biological conclusions reached during analysis of gene expression by DNA microarrays. *Proceedings of the National Academy of Sciences of the USA*, 102(25):8961–8965, 2005.

- [141] A. D. Panani and C. Roussos. Cytogenetic and molecular aspects of lung cancer. *Cancer Lett*, 239(1):1–9, 2006.
- [142] R. Pandey, R. K. Guru, and D. W. Mount. Pathway Miner: extracting gene association networks from molecular pathways for predicting the biological significance of gene expression microarray data. *Bioinformatics*, 20(13):2156–2158, 2004.
- [143] P. Pavlidis, J. Qin, V. Arango, J. J. Mann, and E. Sibille. Using the gene ontology for microarray data mining: A comparison of methods and application to age effects in human prefrontal cortex. *Neurochemical Research*, 29(6):1213–1222, 2004.
- [144] B. Peng and R. G. Reynolds. Cultural algorithms: Knowledge learning in dynamic environments. In *Evolutionary Computation (CEC), IEEE Congress on*, 1751–1758. IEEE, 2004.
- [145] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers*, 61–74. MIT Press Cambridge, 2000.
- [146] R. Polikar. Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45, 2006.
- [147] K. Polyak. Heterogeneity in breast cancer. *The Journal of Clinical Investigation*, 121(10):3786, 2011.
- [148] K. Polyak and P. K. Vogt. Progress in breast cancer research. *Proceedings of the National Academy of Sciences of the USA*, 109(8):2715–2717, 2012.
- [149] J. Quackenbush. Computational analysis of microarray data. *Nature Reviews Genetics*, 2(6):418–427, 2001.
- [150] J. Rahnenführer, F. S. Domingues, J. Maydt, and T. Lengauer. Calculating the statistical significance of changes in pathway activity from gene expression data. *Statistical Applications in Genetics and Molecular Biology*, 3(1), 2004.
- [151] R. G. Reynolds. *An adaptive computer model of the evolution of agriculture for hunter-gatherers in the Valley of Oaxaca, Mexico*. University of Michigan, 1979.

- [152] R. G. Reynolds. An introduction to cultural algorithms. In *Evolutionary Programming, Proceedings of the 3rd Annual Conference on*, 131–139. World Scientific, 1994.
- [153] R. G. Reynolds and B. Peng. Cultural algorithms: modeling of how cultures learn to solve problems. In *Tools with Artificial Intelligence (ICTAI). Proceeding of the 16th IEEE International Conference on*, 166–172. IEEE, 2004.
- [154] T. Robertson, F. Wright, and R. Dykstra. *Order restricted statistical inference*. Wiley New York, 1988.
- [155] P. N. Robinson, A. Wollstein, U. Bohme, and B. Beattie. Ontologizing gene-expression microarray data: characterizing clusters with Gene Ontology. *Bioinformatics*, 20(6):979–981, 2004.
- [156] S. Rodrigues, K. Fathers, G. Chan, D. Zuo, F. Halwani, S. Meterissian, and P. M. CrkI and CrkII function as key signaling integrators for migration and invasion of cancer cells. *Molecular Cancer Research*, 3(4):183–194, 2005.
- [157] L. Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1):1–39, 2010.
- [158] G. Rustici, N. Kolesnikov, M. Brandizi, T. Burdett, M. Dylag, I. Emam, A. Farne, E. Hastings, J. Ison, M. Keays, et al. ArrayExpress update trends in database growth and links to data analysis tools. *Nucleic Acids Research*, 41(D1):D987–D990, 2013.
- [159] T. A. Sanders, T. de Grassi, G. J. Miller, and S. E. Humphries. Dietary oleic and palmitic acids and postprandial factor VII in middle-aged men heterozygous and homozygous for factor VII R353Q polymorphism. *The American Journal of Clinical Nutrition*, 69(2):220–225, 1999.
- [160] B. Schwikowski, P. Uetz, and S. Fields. A network of protein–protein interactions in yeast. *Nature Biotechnology*, 18(12):1257–1261, 2000.
- [161] N. Shah and N. Fedoroff. CLENCH: a program for calculating cluster enrichment using the gene ontology. *Bioinformatics*, 20(7):1196–1197, 2004.
- [162] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611, 1965.

- [163] S. Sharma and J. Settleman. Oncogene addiction: setting the stage for molecularly targeted cancer therapy. *Genes and Development*, 21(24):3214–3231, 2007.
- [164] D. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman Hall CRC, 2000.
- [165] A. Shojaie and G. Michailidis. Aanalysis of gene sets based on the underlying regulatory network. *Journal of Computational Biology*, 16(3):407–426, 2009.
- [166] A. Shojaie and G. Michailidis. Network enrichment analysis in complex experiments. *Statistical Applications in Genetics and Molecular Biology*, 9(1), 2010.
- [167] I. Shureiqi, W. Jiang, X. Zuo, Y. Wu, J. Stimmel, L. Leesnitzer, J. Morris, H. Fan, S. Fischer, and S. Lippman. The 15-lipoxygenase-1 product 13-S-hydroxyoctadecadienoic acid down-regulates PPAR- $\delta$  to induce apoptosis in colorectal cancer cells. *Proceedings of the National Academy of Sciences of the USA*, 100(17):9968–9973, 2003.
- [168] R. J. Slebos and S. Rodenhuis. The molecular genetics of human lung cancer. *Eur Respir J*, 2(5):461–469, 1989.
- [169] T. S rlie, C. M. Perou, R. Tibshirani, T. Aasf, S. Geislerg, H. Johnsenb, T. Hastiee, M. B. Eisen, M. van de Rijni, et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences of the USA*, 98(19):10869–10874, 2001.
- [170] C. Stark, B.-J. Breitkreutz, A. Chatr-Aryamontri, L. Boucher, R. Oughtred, M. S. Livstone, J. Nixon, K. Van Auken, X. Wang, and X. Shi. The BioGRID interaction database: 2011 update. *Nucleic Acids Research*, 39(suppl 1):D698–D704, 2011.
- [171] J. Stelling. Mathematical models in microbial systems biology. *Current Opinion in Microbiology*, 7(5):513–518, 2004.
- [172] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide

- expression profiles. *Proceedings of the National Academy of Sciences of the USA*, 102(43):15545–15550, 2005.
- [173] K. Susztak, E. Ciccone, P. McCue, K. Sharma, and E. P. Böttinger. Multiple metabolic hits converge on CD36 as novel mediator of tubular epithelial apoptosis in diabetic nephropathy. *PLoS Medicine*, 2(2):e45, 2005.
- [174] C. Swagell, D. Henly, and C. P. Morris. Expression analysis of a human hepatic cell line in response to palmitate. *Biochemical and Biophysical Research Communications*, 328(2):432–441, 2005.
- [175] A. L. Tarca, S. Draghici, G. Bhatti, and R. Romero. Down-weighting overlapping genes improves gene set analysis. *BMC Bioinformatics*, 13(1):136, 2012.
- [176] A. L. Tarca, S. Draghici, P. Khatri, S. S. Hassan, P. Mittal, J. sun Kim, C. J. Kim, J. P. Kusanovic, and R. Romero. A novel signaling pathway impact analysis (SPIA). *Bioinformatics*, 25(1):75–82, 2009.
- [177] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.
- [178] B. S. Taylor, N. Schultz, H. Hieronymus, A. Gopalan, Y. Xiao, B. S. Carver, V. K. Arora, P. Kaushik, E. Cerami, B. Reva, et al. Integrative genomic profiling of human prostate cancer. *Cancer Cell*, 18(1):11–22, 2010.
- [179] R. D. C. Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005.
- [180] L. Tian, S. A. Greenberg, S. W. Kong, J. Altschuler, I. S. Kohane, and P. J. Park. Discovering statistically significant pathways in expression profiling studies. *Proceedings of the National Academy of Sciences of the USA*, 102(38):13544–13549, 2005.
- [181] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, 99(10):6567–6572, 2002.
- [182] D. Titterington, G. Murray, L. Murray, D. Spiegelhalter, A. Skene, J. Habbema, and

- G. Gelpke. Comparison of discrimination techniques applied to a complex data set of head injured patients. *Journal of the Royal Statistical Society. Series A (General)*, 144(2):145–175, 1981.
- [183] N. Todorovic-Rakovic. Tgf-beta 1 could be a missing link in the interplay between er and her-2 in breast cancer. *Med Hypotheses*, 65(3):546–551, 2005.
- [184] F. Tortorella. Reducing the classification cost of support vector classifiers through an ROC-based reject rule. *Pattern Analysis and Applications*, 7(2):128–143, 2004.
- [185] G. Valentini and T. G. Dietterich. Low bias bagged support vector machines. In *Machine Learning, Proceedings of the 20th International Conference*, 752–759, 2003.
- [186] F. Van Batenburg, A. P. Gulyaev, and C. W. Pleij. An apl-programmed genetic algorithm for the prediction of rna secondary structure. *Journal of Theoretical Biology*, 174(3):269–280, 1995.
- [187] M. J. van Nimwegen, M. Huigsloot, A. Camier, I. B. Tijdens, and B. van de Water. Focal adhesion kinase and protein kinase b cooperate to suppress doxorubicin-induced apoptosis of breast tumor cells. *Mol Pharmacol*, 70(4):1330–1339, 2006.
- [188] M. J. van Nimwegen and B. van de Water. Focal adhesion kinase: A potential target in cancer therapy. *Biochem Pharmacol*, 73(5):597–609, 2007.
- [189] C. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.
- [190] L. J. van't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveenothers, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.
- [191] V. Vapnik. *The nature of statistical learning theory*. Springer, 1995.
- [192] V. N. Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing NIPS 4*, 831–838. Morgan Kaufmann, 1992.
- [193] C. J. Vaske, S. C. Benz, J. Z. Sanborn, D. Earl, C. Szeto, J. Zhu, D. Haussler, and J. M. Stuart. Inference of patient-specific pathway activities from multi-dimensional

- cancer genomics data using PARADIGM. *Bioinformatics*, 26(12):i237–i245, 2010.
- [194] B. Vincenzi, G. Schiavon, M. Silletta, D. Santini, G. Perrone, M. Di Marino, S. Angeletti, A. Baldi, and G. Tonini. Cell cycle alterations and lung cancer. *Histol Histopathol*, 21(4):423–435, 2006.
- [195] C. Voichita, M. Donato, and S. Draghici. Incorporating gene significance in the impact analysis of signaling pathways. In *Machine Learning Applications (ICMLA), International Conference on*, 126–131, 2012.
- [196] C. Voichita, M. Donato, and S. Draghici. A genetic algorithms framework for estimating individual gene contributions in signaling pathways. In *Evolutionary Computation (CEC), IEEE Congress on*, 650–657, 2013.
- [197] C. Voichita, P. Khatri, and S. Draghici. Identifying uncertainty regions in support vector machines using geometric margin and convex hulls. In *Neural Networks (IJCNN), International Joint Conference on*, 3319–3324, 2008.
- [198] C. Voichita, Z. Xu, S. Haidarian, R. Romero, and S. Draghici. Identifying patient sub-groups using classification uncertainty. *under review*, 2013.
- [199] E. Wang, Z. Qian, M. Nakasono, T. Tanahashi, K. Yoshimoto, Y. Bando, E. Kudo, M. Shimada, and T. Sano. High expression of toll-like receptor 4/myeloid differentiation factor 88 signals correlates with poor prognosis in colorectal cancer. *British Journal of Cancer*, 102(5):908–915, 2010.
- [200] Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009.
- [201] M. Wegkamp and M. Yuan. Support vector machines with a reject option. *Bernoulli*, 17(4):1368–1385, 2011.
- [202] B. Weinshenker, P. Santrach, A. Bissonet, S. McDonnell, D. Schaid, S. Moore, and M. Rodriguez. Major histocompatibility complex class ii alleles and the course and outcome of ms a population-based study. *Neurology*, 51(3):742–747, 1998.
- [203] T. Winton, R. Livingston, D. Johnson, J. Rigas, M. Johnston, C. Butts, Y. Cormier,



- G. Goss, R. Inculet, E. Vallieres, et al. Vinorelbine plus cisplatin vs. observation in resected non-small-cell lung cancer. *New England Journal of Medicine*, 352(25):2589–2597, 2005.
- [204] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.
- [205] J. Xia and D. Wishart. Metpa: a web-based metabolomics tool for pathway analysis and visualization. *Bioinformatics*, 26(18):2342–2344, 2010.
- [206] Z. Xu, C. Voichita, S. Draghici, and R. Romero. Z-bag: A classification ensemble system with posterior probabilistic outputs. *Computational Intelligence*, 29(2):310–330, 2013.
- [207] Y. H. Yang and T. Speed. Design issues for cDNA microarray experiments. *Nature Reviews Genetics*, 3(8):579–588, 2002.
- [208] A. Young, N. Whitehouse, J. Cho, and C. Shaw. Ontologytraverser: an R package for GO analysis. *Bioinformatics*, 21(2):275–276, 2005.
- [209] M. Yuan and M. Wegkamp. Classification methods with reject option based on convex risk minimization. *The Journal of Machine Learning Research*, 11:111–130, 2010.
- [210] B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Machine learning, International Workshop and Conference on*, 609–616, 2001.
- [211] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Knowledge Discovery and Data mining (KDD), Proceedings of the 8th ACM SIGKDD international conference on*, 694–699. ACM, 2002.
- [212] B. R. Zeeberg, W. M. Feng, G. Wang, M. D. Wang, A. T. Fojo, M. Sunshine, S. Narasimhan, D. W. Kane, W. C. Reinhold, S. Lababidi, K. J. Bussey, J. Riss, J. C. Barrett, and J. N. Weinstein. Gominer: a resource for biological interpretation of genomic and proteomic data. *Genome Biology*, 4(4):R28, 2003.
- [213] Y. Zhao, M. Chen, B. Pei, D. Rowe, D. Shin, W. Xie, F. Yu, and L. Kuo. A bayesian approach to pathway analysis by integrating gene–gene functional directions and mi-



- croarray data. *Statistics in Biosciences*, 4(1):105–131, 2012.
- [214] S. Zhong, L. Tian, C. Li, K.-F. Storch, and W. H. Wong. Comparative analysis of gene sets in the gene ontology space under the multiple hypothesis testing framework. In *Computational Systems Bioinformatics (CSB), Proceedings of the IEEE conference on*, 425–435, 2004.
- [215] W. Zou and V. V. Tolstikov. Pattern recognition and pathway analysis with genetic algorithms in mass spectrometry based metabolomics. *Algorithms*, 2(2):638–666, 2009.

**ABSTRACT****TOWARDS PERSONALIZED MEDICINE USING  
SYSTEMS BIOLOGY AND MACHINE LEARNING**

by

**CĂLIN VOICHIȚA**

August 2013

**Advisor:** Dr. Sorin Draghici**Major:** Computer Science (Bioinformatics)**Degree:** Doctor of Philosophy

The rate of acquiring biological data has greatly surpassed our ability to interpret it. At the same time, we have started to understand that evolution of many diseases such as cancer, are the results of the interplay between the disease itself and the immune system of the host. It is now well accepted that cancer is not a single disease, but a “complex collection of distinct genetic diseases united by common hallmarks”. Understanding the differences between such disease subtypes is key not only in providing adequate treatments for known subtypes but also identifying new ones. These unforeseen disease subtypes are one of the main reasons high-profile clinical trials fail. To identify such cases, we proposed a classification technique, based on Support Vector Machines, that is able to automatically identify samples that are dissimilar from the classes used for training. We assessed the performance of this approach both with artificial data and data from the UCI machine learning repository. Moreover, we showed in a leukemia experiment that our method is able to identify 65% of the MLL patients when it was trained only on AML vs. ALL. In addition, to augment our ability to understand the disease mechanism in each subgroup, we proposed a systems biology approach able to consider all measured gene expressing changes, thus eliminating the possibility that small but important gene changes (e.g. transcription factors) are omitted from the analysis. We showed that this approach provides consistent results that do not depend on the choice of

an arbitrary threshold for the differential regulation. We also showed in a multiple sclerosis study that this approach is able to obtain consistent results across multiple experiments performed by different groups on different technologies, that could not be achieved based solely using differential expression. The cut-off free impact analysis was released as part of the ROntoTools Bioconductor package.

# AUTOBIOGRAPHICAL STATEMENT

CALIN VOICHITA

## Education

- Ph.D. Computer Science, Wayne State University, Detroit MI, USA, August 2013.
- M.S. Computer Science, Wayne State University, Detroit MI, USA, May 2010.
- B.S. Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania, June 2006.

## Peer review publications

1. **C. Voichita\***, Z. Xu\*, S. Haidarian, R. Romero and S. Draghici. *Identifying patient sub-groups using classification uncertainty*. (under review).
2. C. Mitrea\*, Z. Taghavi\*, B. Bokanizad, S. Hanoudi, R. Tagett, M. Donato, **C. Voichita** and S. Draghici. *Methods and approaches for topology-based analysis of biological pathways*. (under review).
3. **C. Voichita\***, M. Donato\*, and S. Draghici. *A Genetic Algorithms Framework for Estimating Individual Gene Contributions in Signaling Pathways*. In Proceedings of the IEEE Congress on Evolutionary Computation, IEEE CEC 2013, Cancun, Mexico, 20-23 Jun. 2013.
4. **C. Voichita**, M. Donato, and S. Draghici. *Incorporating gene significance in the impact analysis of signaling pathways*. In Proceedings of the International Conference on Machine Learning and Applications, ICMLA 2012, Boca Raton, FL, USA, 12-15 Dec. 2012.
5. Z. Xu\*, **C. Voichita\***, S. Draghici, and R. Romero. *Z-bag: A Classification Ensemble System With Posterior Probabilistic Outputs*. Computational Intelligence, 2012 – DOI: 10.1111/j.1467-8640.2012.00432.x.
6. N. Ansari, R. Bau, **C. Voichita**, S. Draghici. *Detecting phenotype-specific interactions between biological processes from microarray data and annotations*. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 9, no. 5 (2012): 1399-1409.
7. **C. Voichita**, P. Khatri, and S. Draghici. *Identifying uncertainty regions in support vector machines using geometric margin and convex hulls*. In Proceedings of the International Joint Conference on Neural Networks, JCNN 2008. IEEE World Congress on Computational Intelligence, Hong Kong, 1-8 Jun. 2008.
8. S. Draghici, P. Khatri, A.L. Tarca, K. Amin, A. Done, **C. Voichita**, C. Georgescu, and R. Romero. *A systems biology approach for pathway level analysis*. Genome Research, 17(10):1537–1545, 2007.
9. P. Khatri, **C. Voichita**, K. Kattan, N. Ansari, A. Khatri, C. Georgescu, A.L. Tarca, and S. Draghici. *Onto-Tools: New additions and improvements in 2006*. Nucleic Acids Research, 37(Web Server issue), Jul. 2007.

## Software packages

- **C. Voichita** and S. Draghici. *ROntoTools: R Onto-Tools suite (Bioconductor 2.12)*. Apr. 2013. <http://bioconductor.org/packages/release/bioc/html/ROntoTools.html>

\* joint first authors.