Interpretable Methods in Cancer Diagnostics

Adrin Jalali

June 26, 2019

Contents

Abstract

Cancer is a hard problem. It is hard for the patients, for the doctors and nurses, and for the researchers working on understanding the disease and finding better treatments for it. The challenges faced by a pathologist diagnosing the disease for a patient is not necessarily the same as the ones faced by cell biologists working on experimental treatments and understanding the fundamentals of cancer. In this thesis we work on different challenges faced by both of the above teams.

This thesis first presents methods to improve the analysis of the flow cytometry data used frequently in the diagnosis process, specifically for the two subtypes of non-Hodgkin Lymphoma which are our focus: Follicular Lymphoma and Diffuse Large B Cell Lymphoma. With a combination of concepts from graph theory, dynamic programming, and machine learning, we present methods to improve the diagnosis process and the analysis of the abovementioned data. The interpretability of the method helps a pathologist to better understand a patient's disease, which itself improves their choices for a treatment.

In the second part, we focus on the analysis of DNA-methylation and gene expression data, both of which presenting the challenge of being very high dimensional yet with a few number of samples comparatively. We present an ensemble model which adapts to different patterns seen in each given data, in order to adapt to noise and batch effects. At the same time, the interpretability of our model helps a pathologist to better find and tune the treatment for the patient: a step further towards personalized medicine.

"Growth for the sake of growth is the ideology of the cancer cell."

- Edward Abbey

1 Introduction

Cancer has been with the human species throughout our 4000 years of history. Although our understanding of cancer has changed drastically over time, its treatment remains a challenge and for some cancer subtypes we have not been able to find an effective treatment yet. The immense frustration of dealing with cancer not only affects the patients, but also the doctors, pathologists, and oncologists treating those patients. Siddhartha Mukherjee in his book *"The Emperor of All Maladies"* explains the feeling with these words [93, prologue]:

> ...
>
> There were seven such cancer fellows at this hospital. On paper, we seemed like a formidable force: graduates of five medical schools and four teaching hospitals, sixty-six years of medical and scientific training, and twelve postgraduate degrees among us. But none of those years or degrees could possibly have prepared us for this training program. Medical school, internship, and residency had been physically and emotionally grueling, but the first months of the fellowship flicked away those memories as if all of that had been child's play, the kindergarten of medical training.
>
> ...
>
> The stories of my patients consumed me, and the decisions that I made haunted me. *Was it worthwhile continuing yet another round of chemotherapy on a sixty-six-year-old pharmacist with lung cancer who had failed all other drugs? Was it better to try a tested and potent combination of drugs on a twenty-six-year-old woman with Hodgkin's disease and risk losing her fertility, or to choose a more experimental*

combination that might spare it? Should a Spanish-speaking mother of three with colon cancer be enrolled in a new clinical trial when she can barely read the formal and inscrutable language of the consent forms?

I can better put this thesis in context by giving my personal perspective on cancer research and diagnosis. I spent over a year as a researcher at British Columbia's Cancer Research Center (BCCRC) in Vancouver, Canada, which also admitted patients for diagnosis and treatment. As a result, I worked closely with oncologists who were diagnosing patients as well as cell biologists researching the fundamentals of cancer and new treatments for it. Their experiences and the challenges they were facing had a great impact on me and to a large extent shaped my research over the next few years. Then, once I moved to the Bioinformatics Lab at the Max Planck Institute for Informatics in Saarbrücken, Germany, the Lab and my supervisors had a very strong background in machine learning and as a result I was better equipped with the required statistics and machine learning skills to tackle the computational problems explained here. Here is a list of some of the challenges I saw people facing at BCCRC:

- Some patients enter the clinic carrying cancer type A, which is mild and does not require an aggressive treatment. Therefore they are given the appropriate treatment while their condition is monitored over time. However, the disease in some of these patients develops into another type, let's say type B, which is more aggressive and sometimes requires a stronger or a different treatment. Considering the fact that like many other diseases, cancer can be defeated best while in its earliest stages, the prognosis for these patients would be better, if we knew their disease would develop into type B earlier.

- Similar to the above issue, out of the many patients who go in remission, *i.e* they seem free of cancer after the course of the treatment, some relapse with a cancer which is significantly more resistant to usual treatments compared to when they were originally diagnosed. This sometimes happens when a very small number of cells from the original cancer are or become resistant to the drugs and survive the treatment, but go undetected for a while in the tests and scans. It may take months for those cell populations to grow large enough to be detected again. Now the question is, looking back at the data of these patients, could we have detected those cells, or something about the original cancer cells, predicting the relapse earlier during the treatment or even at the time of the original diagnosis?

- At BCCRC, cancer cell biologists also research cancer by looking at the effects of different drugs and drug combinations targeting different genes. Some of my cell biologist friends, often taking recommendations from their supervisors, would choose a few genes and spend years investigating the role of those genes in the development of a particular type of cancer. Of course they would do their best to choose the most relevant set of genes, but the

task of choosing a few genes out of over 22k genes on the human genome is rather challenging and does not always lead to successful treatments and positive results. There is also a bias towards the genes which have been discovered earlier and have been studied more in depth. If the cancer happens to be related to one of the less studied genes, it will usually stay under the radar for while.

All of the abovementioned issues involve decisions that are hard to make, and it does not help that we lack a deep understanding of cancer. The battle against cancer has many fronts, including prevention, diagnosis, and treatment, all of which benefit from advancements in understanding the disease. As part of the process, cancer researchers try to understand the disease in the lab, and once their findings are confirmed, accepted by the community, and pass the legal requirements, they are used by pathologists and oncologists in clinics. However, the diagnosis itself is also complex, challenging, and in many cases not a definitive one. This is why sometimes doctors do not agree on the exact diagnosis, and a counsel of experts is required for a better and more reliable diagnosis and a treatment which hopefully results in a better prognosis.

To better understand the challenge, we need to realize that cancer is a collection of extremely smart and complicated diseases. Although they share many common characteristics, the same treatment does not result in a similar prognosis in different patients. For example, two patients may come in with two physically very similar malignant tumors in their breasts. However, one of the patient's tumor grows in response to estrogen, while the other one shows no reaction to estrogen. In this case, a treatment which blocks estrogen receptors is very effective for the first patient (ER+), while being completely ineffective in the second patient who has an ER- subtype of breast cancer.

At the core of it, it comes down to the fact that in normal cells there are processes and checks put in place which define when and if the cell should divide or die at a certain time or under certain conditions. Some of those mechanisms act like an automated self destruct switch which is triggered if something goes wrong in the cell. However, our cells are under constant stress from the external factors which damage them, UV being one example, and sometimes the damage to the cell affects those mentioned mechanisms and disables them. This may lead the cell to divide uncontrollably and become cancerous. Another difference between normal cells and cancerous ones, is that normal cells are capable of repairing most of the mutations happening on their DNA as a result of either external stress or during cell division, whereas those processes themselves are often damaged in a cancerous cell. As a result, the rate of mutation in cancerous cells is higher by a few orders of magnitude compared to a normal healthy cell. The high rate of cell division in combination with hypermutation, makes cancerous cells very adaptive to their environment, as well as against the drugs attacking them.

Cancer treatments are methods and substances which ideally target only the cancerous cells and kill them, or stop their growth and cell division. In a sense, they are poison, but ideally only to cancer. However, cancer cells are derived from

our own cells, and therefore it is not always easy to distinguish them from normal cells. The bigger the difference between the cancer cells and our normal healthy cells, the easier it is to target them; but unfortunately not all cancer subtypes are easily distinguishable from healthy cells for the purpose of treatment.

The efforts in this thesis are divided into two sections. In the first part, *i.e.* Chapter 3, we tackle some of the issues faced by clinicians while analyzing flow cytometry data. A flow cytometer, *i.e.* the instrument producing the data, is capable of measuring a dozen or so different characteristics of individual cells in a given sample, which in our case are all from biopsies or blood samples taken from patients. This type of single cell measurement data enables us to detect and sort different cell populations and cell types within a single given sample. Chapter 3 explains the methods we designed and implemented to analyze flow cytometry to accomplish the following goals:

- *Automated analysis of the data:* manual analysis of flow cytometry data involves looking at 2D projections of the data along two selected features at a time, and potentially filtering a part of the data before moving on to a different projection. This way at each step a sub-population, *i.e.* a subset of the cells, is selected for further analysis. Presence or absence of a cell population, or its predominance compared to other cell populations, can be indicative of a certain type of cancer. The methods we developed can automate this process and help clinicians to quickly find a certain cell population within a given sample.

- *Novel cell population discovery*: our method analyzes the relationship between all the cell populations it finds with the target cancer subtypes, and reports the cell populations which seem to be informative or predictive in differentiating two subtypes. Discovering new predictive cell populations can help the diagnosis process, as well as potentially a better choice of treatment for each patient.

- *Quality assurance*: in clinics, oncologists sometimes go through a QA process in which they randomly look at a limited number of past patients and retrospectively check the quality of the original diagnosis. However, due to limited available resources, they can only go through a small number of cases. We develop a method to report cases where there is a higher probability that the diagnosis could be improved, hoping that retrospective investigation of those cases could generally improve the diagnostic process.

- *Visualization*: one important aspect of analyzing cell populations is that they are related to one another, in the sense that some cell populations are a subset of others. When a tool reports some cell populations as important or relevant, it is also essential to report and visualize which characteristics of those cell groups are more important in defining and detecting them, so that they can also be best detected with the manual process. Another reason to visualize the data the way we do, is that it helps a cancer researcher to better understand why or how those cell populations are relevant.

- *Enabling similar tests using cheaper machines*: one important benefit of visualizing the cell populations the way we do, is that it shows alternative ways that a cell population can be isolated from the other cells using fewer measurements per cell. As a result, the same cell population can then be filtered using a smaller number or a cheaper variation of chemical reagents used in flow cytometry. Consequently, a pathologist in a place where there is harder or no access to more expensive cytometers and reagents, can benefit from the findings of a research institute with significantly more budget at their disposal.

The second part of the thesis, described in Chapter 4, focuses on some types of data that are not yet available as a routine test in clinics, but are essential to our deeper understanding of cancer. These data are measurements from the whole human genome, and our focus is on gene expression profiles and a modification on the DNA, called methylation, *i.e.* a methyl component is attached a Cytosine(C) or an Adenine(A) base on the DNA. A gene expression profile measures the activity of all 22k+ genes in a given sample, and a usual DNA methylation profile measures the methylation level of about 450k sites on human DNA. DNA methylation levels can change due to environmental factors and during cell differentiation and aging; there is also evidence that they are sometimes heritable. These data, as well as others such as DNA, RNA, and protein sequence data have been increasingly used by biologists and computational biologists to better understand cell biology in many fields, including cancer research. These data and methods have been so essential to our understanding of cancer that the classification of some cancer types now depend on them. In some cases, molecular and chemical analysis of cancer has shown us that two different classes of cancer are indeed the same disease, only in different stages. Lymphoma and leukemia are two good examples which used to be considered two different cancers, and now together they form the *lymphoid neoplasms* group, since on the molecular level they have a lot in common.

Although there have been magnificent advancements in the field from the computational perspective, the computational problems are still considered very hard problems. The curse of dimensionality on top of the low number of samples compared to the number of dimensions in the data all result in a hard computational problem as explained in detail in Chapter 2. Also, the fact that the data is often affected by noise and batch effects doesn't help the case either, covered in more detail in Chapter 4. Another challenging factor is that cancer, and even a single cancer tumor, is heterogeneous and different cases show very different genetic profiles. Historically this has lead to further classification of cancer and, at times changing the classification and merging some classes altogether as mentioned above. As a result, ideally when a patient is prescribed a treatment, that treatment is fine tuned and adapted as much as possible to tackle the patient's specific disease. This fine tuning, which covers the spectrum from choosing the best combination of drugs to designing and manufacturing a drug specific to that patient and that patient alone, is called personalized medicine. With a focus on adaptive and interpretable models, Chapter 4 addresses the following challenges:

- *Per-patient significant gene/genome region discovery*: the models we have developed are adaptive, *i.e.* for a given data from a single patient, the model can give some information about the potential underlying cause of the disease for that specific patient. The provided information would give clues to practitioners to better find a treatment for the patient.

- *Per-disease significant gene/genome region discovery*: in order to help cancer researchers studying cancer and cancer treatments, we design computational methods capable of reporting a list of promising genes that are influential in determining the cancer subtypes. The reported set of genes can then be a starting point for cell biologists to increase the probability of finding a new treatment.

- *Interpretability, per-patient, per-disease*: our models investigate genes in networks and take their relationships between one another which can be seen from the data into account. Interpreting and visualizing the models to show the significance of each gene as well as their relationships helps pathologists and cancer cell biologists to better trust the methods and understand how the model works.

In the following chapters, some of the basics required to follow the later sections, including some related concepts in machine learning, graph theory, and biology are covered in Chapter 2. In Chapter 3 we focus on flow cytometry data and explain the design and implementation of a few pieces which together make an end to end pipeline to analyze such data, and apply that to some specific lymphoma subtypes. Then we continue in Chapter 4 with the analysis of mostly DNA methylation data and design some adaptable and interpretable models with an eye towards personalized medicine.

2
Background

This chapter covers the very basics of some of the concepts required to follow the work in later chapters. These concepts are divided into three sections: Section 2.1 machine learning, Section 2.2 graph theory, and Section 2.3 cell biology. This chapter by far does not exhaustively covers explained concepts, and the reader is strongly recommended to study the subjects using the references and text books cited throughout the chapter. In particular, The Elements of Statistical Learning [62], Pattern Recognition and Machine Learning [17], and Introduction to Graph Theory [154] cover machine learning and graph theory respectively; and the biology and cancer background is best covered by Molecular Biology of the Cell [7] and Postgraduate Haematology [64].

2.1 Machine Learning

Machine learning techniques are used to extract information from data, or make some predictions about the data. We can recognize two groups of learning problems: supervised and unsupervised learning. Unsupervised learning, deals with data sets which are in the form of a set of data points, and no desired output is given. Clustering is the most studied unsupervised learning problem which is the task of grouping similar data together in certain clusters [62, Ch. 14] [96, Ch. 1].

Supervised learning deals with data sets that are in the form of a set of input and outputs, and the task at hand is to predict the output using the input [62, Ch. 2], [96, Ch. 1]. Classification and regression are supervised learning problems.

Classification is the problem of putting data into different classes [62, Ch. 1]. During the training phase, the matrix $X_{samples \times features}$ is given as the input and

$y_{samples}$ as the desired output. The vector y has values from a discrete set. If the set has only two distinct values, the problem is called a binary classification. On the other hand, if the output is a continues variable, then it is called a regression problem [62, Ch. 1].

Logistic regression [151, 34], Support Vector Machines (SVM) [147, 20], and decision trees [62, Ch. 9] are examples of classification methods, and Linear regression [62, Ch. 3], Gaussian processes [118], and kernel based regression [127, Ch. 9] are some well established regression methods.

2.1.1 Empirical Risk Minimization

As mentioned above, supervised learning deals with predicting an output $y_i \in Y$ given an input $x_i \in X$. The task is to find a function $f(.)$ which best predicts the output, given any possible input. Because in practice we have access to a limited given data set, the best we can do is to find a function that best predicts the output given any input in the data set.

We can formulate finding the best function $f(.)$, as finding a function that has the minimum loss over the data. Therefore we need a loss function defined as $L(x, y, f(x))$, with x being the input, y the desired output, and $f(x)$ the predicted output. The loss value has to be in $[0, \infty)$, and $L(x, y, y) = 0$ [127, p. 62]. The empirical risk function is then defined as [127, p. 67]:

$$R_{emp}[f] := \frac{1}{m} \sum_{i=1}^{m} L(x_i, y_i, f(x_i)) \tag{2.1}$$

Now let $\mathcal{F}$ be the function space available to choose $f(.)$ from it. The best function is one which minimizes the risk function [127, p. 67]:

$$\arg\min_{f \in \mathcal{F}} R_{emp}[f] = \arg\min_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} L(x_i, y_i, f(x_i)) \tag{2.2}$$

But there is a problem with the above formulation if the function space $\mathcal{F}$ is rich enough to fit to the given data too well. Imagine a function that returns y_i for each x_i in the training set, and 0 otherwise. This function clearly has a minimum loss of 0, but does not generalize on unseen data. This is called overfitting in machine learning. One way to fix this issue is to regularize the loss function in some way, and give preference to functions $f(.)$ with lower complexity, or smoother functions. This is referred to as regularized empirical risk minimization [127, Ch. 4.1], or structural risk minimization [146, Ch. 4.1]. Assume $\Omega(f)$ is a penalty assigned to function $f(.)$; then the regularized empirical risk is formulated as:

$$R_{emp}[f] := \frac{1}{m} \sum_{i=1}^{m} L(x_i, y_i, f(x_i)) + \lambda \Omega(f) \tag{2.3}$$

Parameter λ is the regularization term and we estimate it, among other model parameters, using cross validation.

For instance, in the case of a linear model, assume a model minimizes a loss function $E(X, Y)$, where X is the input matrix and Y is the output vector or matrix, defined as:

$$E(X, Y) = \| Y - X\beta \|_2 \tag{2.4}$$

The optimization algorithm finds a β that minimizes the loss function in Formula 2.4. As explained above, having enough number of features, the optimization algorithm might find a β that gives a perfect loss, *i.e.* 0. But in noisy environments the resulting β is probably not the real β of the underlying model producing the data. The vector β might also have some extreme values that are likely not desired. Penalizing the *size* of β as shown in Formula 2.6 will address the abovementioned concern. The size of a vector in this context is represented by its l_1 or l_2 norm as defined in Formula 2.5. The l_1-regularization is an appropriate tool when the intention is to reduce the number of features a model takes into account for prediction as well as its complexity [104], since it favors more absolute zeros in the β vector, and hence is also used as a feature selection tool.

$$\| \beta \|_p := \left(\sum_{i=1}^{n} | \beta_i |^p \right)^{1/p} \tag{2.5}$$

$$E(X, Y) + \alpha \| \beta \|_2 = \| Y - X\beta \|_2 + \alpha \| \beta \|_2$$
or
$$E(X, Y) + \alpha \| \beta \|_1 = \| Y - X\beta \|_2 + \alpha \| \beta \|_1 \tag{2.6}$$

2.1.2 Cross Validation

Cross validation is a technique used in method selection and performance estimation. In cross validation we divide the given training data into k folds, set aside one of those k folds, train the model on $k - 1$ remaining sections, and test the performance of the model on the set aside part of the data. Then repeat this process for all k folds to assess the overall performance of the method. A special case of k-fold cross validation is leave-one-out in which $k = n$, the number of samples. Leave-one-out cross validation is computationally intensive for relatively large number of samples. A popular k is 10, which is shown to have lower variance than leave-one-out method, and it has a low bias [62, Ch. 7].

There are also some variations to the simple k-fold cross validation scheme. One way is to repeat the k-fold system multiple times with a random shuffle of the data before each k-fold test, and calculate the estimated error using the repeated test. Another variation is to randomly partition the data into train

and test partitions several times and use these sets to estimate the performance of the method.

The latter two variations are shown to give better estimates of the true error of the method compared to leave one out and a single 10-fold scheme [74, 39]. Because repeating a k-fold scheme can be computationally intensive depending on the method being tested, we sometimes use a repeated random partitioning of the data in our work. In some even more computationally intensive cases, we have limited our analysis to a single k-fold scheme to select methods.

2.1.3 Feature Selection

Feature selection is the task of selecting features most relevant and predictive to the problem at hand from the given set of features. It becomes particularly a hard task when the number of features in the data is of a higher magnitude compared to the number of given samples. Table 2.1 shows an example number of samples vs. number of features in a typical data in this study. One of the challenges when dealing with such a large number of features is that if there are enough number of features, even if they have a probability distribution independent of the outcome, some of them might falsely seem correlated with the outcome due to the relatively small sample size. Another obstacle comes from the fact that our features are not independent and they function in complex networks. As a result, features should be considered in groups, which is a combinatorial and intractable problem.

Sample Data		
Sample Count	Gene Expression Data Feature Count	450K Methylation Chip Data Feature Count
500	$\approx 20,000$	$\approx 450,000$

Table 2.1: An example number of samples and features in our usual data

We have used correlation [107], mutual information [129], and l_1-regularized methods [104] as techniques to select features.

2.1.4 Support Vector Machines

Support vector machines (SVM) can be used both for regression and classification tasks [20, 133]. As a binary classifier, SVM finds an optimal hyperplane to separate data points in the feature space by maximizing the hyperplane's margin to the nearest points on both sides of it. Therefore given a test data point, its side with regard to the hyperplane determines its class. As a regressor however, SVM finds an optimal hyperplane to interpolate given data points by minimizing the hyperplane's distance from data points. In this work we use SVMs as a binary classifier.

Formally speaking, given a data-set $\mathcal{D}$ of n data points:

$$\mathcal{D} = (\mathbf{x}_i, y_i)|\mathbf{x}_i \in \mathbb{R}^p, y_i \in -1, 1_{i=1}^n \tag{2.7}$$

where $\mathbf{x}_i$ is a real vector of length p, and y_i is either 1 or -1. A p-dimensional hyperplane, characterized by its normal vector $\mathbf{w}$ and its intercept $\mathbf{b}$, is the set of points $\mathbf{x}$ that fit in Formula 2.8.

$$\mathbf{w}.\mathbf{x} - b = 0 \tag{2.8}$$

Now consider two hyperplanes on both sides of the abovementioned hyperplane as formulated bellow:

$$\begin{aligned} &\mathbf{w}.\mathbf{x} - b = 1 \\ &\mathbf{w}.\mathbf{x} - b = -1 \end{aligned} \tag{2.9}$$

The distance between each of these hyperplanes and the one in the middle is $\frac{1}{\|\mathbf{w}\|}$. Therefore the distance between the two of them is $\frac{2}{\|\mathbf{w}\|}$. For now we assume the data is linearly separable in its feature space, *i.e.* there exists a hyperplane that perfectly separates the data into two classes without error. Such a hyperplane satisfies the following constraint:

$$y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1 \text{ for all } 1 \leq i \leq n \tag{2.10}$$

An optimal hyperplane is one such that it maximizes the margin; hence formulated as Formula 2.11. An illustration of the optimal solution is presented in Fig. 2.1.

$$\begin{aligned} &\arg\max_{(\mathbf{w},b)} \frac{1}{\|\mathbf{w}\|_2} \\ &\text{s.t.} \\ &y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1 \text{ for all } 1 \leq i \leq n \end{aligned} \tag{2.11}$$

However, for an easier optimization and mathematical convenience, the above optimization problem is usually formulated as Formula 2.12 which has the same solution as $\mathbf{w}$ and b [146, Ch. 5], [127, Ch. 7].

$$\begin{aligned} &\arg\min_{(\mathbf{w},b)} \frac{1}{2}\|\mathbf{w}\|_2^2 \\ &\text{s.t.} \\ &y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1 \text{ for all } 1 \leq i \leq n \end{aligned} \tag{2.12}$$

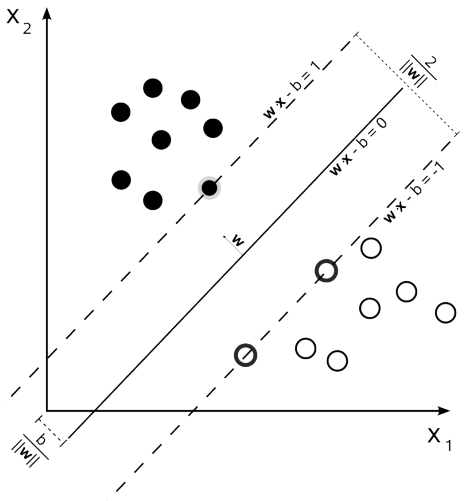


Figure 2.1: Illustration of the optimal hyperplane in a support vector machine model, for a 2-dimensional data.

which can be written as Formula 2.13 after introducing Karush-Kuhn-Tucker (KKT) multipliers [77] [146, Ch. 5].

$$\begin{aligned} &\arg\min_{\mathbf{w},b}\max_{\boldsymbol{\alpha}}\left\{\frac{1}{2}\|\mathbf{w}\|_2^2-\sum_{i=1}^{n}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x_i}-b)-1]\right\}\\ &\text{s.t.}\\ &\alpha_i\geq 0 \text{ for } 1\leq i\leq n \end{aligned} \tag{2.13}$$

Multipliers α_i will be 0 for each $\mathbf{x}_i$ that does not lie on either of the marginal hyperplanes. For example in Figure 2.1, α_i is non-zero for only three of the data points; the ones that are exactly on either of the marginal lines. The corresponding $\mathbf{x}_i$ for which α_i is non-zero are called *support vectors*.

It can be shown that Formula 2.14 is a dual of the optimization problem defined in Formula 2.13 [127, p. 14] [146, Ch. 5].

$$\begin{aligned} &\arg\max_{\boldsymbol{\alpha}}\left\{\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j\right\}\\ &\text{s.t.}\\ &\alpha_i\geq 0 \text{ for } 1\leq i\leq n\\ &\sum_{i=1}^{n}\alpha_i y_i=0 \end{aligned} \tag{2.14}$$

Now assume the following notations and definitions:

$$\begin{aligned} \phi(\mathbf{x}) &:= \mathbf{x} \\ \langle \mathbf{x}_i, \mathbf{x}_j \rangle &:= \mathbf{x}_i^T \mathbf{x}_j \\ k(\mathbf{x}_i, \mathbf{x}_j) &:= \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \end{aligned} \tag{2.15}$$

Putting function k in Formula 2.14, the SVM's optimization problem can be written as:

$$\begin{aligned} &\arg\max_{\boldsymbol{\alpha}} \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \right\} \\ &\text{s.t.} \\ &\alpha_i \geq 0 \text{ for } 1 \leq i \leq n \\ &\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{2.16}$$

The identity function used in Formula 2.15 is not the only option. We can transform the data into another feature space using a different $\phi(.)$, and then use dot-product in that space. This is useful for cases that the data is not linearly separable in its original feature space, but linearly separable using a non-linear transformation.

Using Mercer's theorem [90] and its corollary Mercer's condition, it can be shown that any function k satisfying the following condition can be used as a *kernel* in Formula 2.16 [127, Ch. 2.2].

$$\forall \mathcal{D}, \forall c_i, c_j \in \mathbb{R} : \sum_{i=1}^{n} \sum_{j=1}^{n} k(\mathbf{x}_i, \mathbf{x}_j) c_i c_j \geq 0 \tag{2.17}$$

This means to solve the SVM optimization, we only need the kernel matrix $\mathbf{K}$, which has $k(\mathbf{x}_i, \mathbf{x}_j)$ as its values. For many kernels, this matrix can be calculated directly without transforming the data into the alternate representing feature space. This also means that we can use kernels which have an infinite dimensional representing feature space. This technique is sometimes called the *kernel trick*. Arguably, other than dot-product, the most famous kernel function k satisfying the above condition is the *Gaussian kernel*, also known as the *radial basis function (RBF) kernel* [127, Ch. 2]:

$$\begin{aligned} &k(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2\sigma^2} \right) \\ &\sigma \in \mathbb{R} \end{aligned} \tag{2.18}$$

This is an example of a kernel with an infinite dimensional representing kernel space [134]. Many implementations use a different formulation which uses a different parametrization, using $\gamma = \frac{1}{2\sigma^2}$:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma||\mathbf{x_i} - \mathbf{x_j}||^2\right)$$
$$\gamma \in \mathbb{R}^+ \tag{2.19}$$

Regularization of Support Vector Machines

In real-world applications data-sets are often not linearly separable, *i.e.* no hyperplane can perfectly separate the two classes of the data-set. To handle such cases, Formula 2.12 can be modified as Formula 2.20 with the introduction of ξ_i called slack variables [32],[127, Ch. 7.5]. This allows some of the data points to be within the margin area or to be on the wrong side of the hyperplane. This formulation is also referred to as a soft margin hyperplane.

$$\arg\min_{(\mathbf{w},b)} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i$$
s.t.
$$y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1 - \xi_i \text{ for all } 1 \leq i \leq n \tag{2.20}$$

A formula similar to Formula 2.3 can be derived from Formula 2.20 as shown in Formula 2.21 [62, Ch. 12] [61].

$$\arg\min_{(\mathbf{w},b)} \sum_{i=1}^{n}[1 - y_i(\mathbf{w}x_i - b)] + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \tag{2.21}$$

Note that parameter C in Formula 2.20 corresponds to $\frac{1}{\lambda}$ in Formula 2.21. The corresponding penalty function of Formula 2.3 in Formula 2.21 is the l^2-norm of the vector $\mathbf{w}$. Similar to methods such as *lasso* [62, Ch. 3] this penalty function can be replaced with the l^1-norm of the vector $\mathbf{w}$ shown in Formula 2.22 [163].

$$\arg\min_{(\mathbf{w},b)} \sum_{i=1}^{n}[1 - y_i(\mathbf{w}x_i - b)] + \frac{\lambda}{2}\|\mathbf{w}\|_1^2 \tag{2.22}$$

It is important to note that the algorithm to solve the above optimization problem does not involve the kernel trick [163], which means we cannot use similarity measures that require transforming data into spaces we cannot compute, such as the RBF kernel. In order to use the l^1-norm regularized SVM with such kernels, an approximation of the feature space can be used to transform the data first, and then apply the above optimization problem on the transformed data [116].

2.1.5 Gaussian Processes

Given a regression or a classification problem, one approach is to find the most likely function among the functions we consider reasonable for our problem. For instance, a linear regression assumes the underlying function explaining the data to be linear, and then tries to find one which is most probable to be the real underlying function for the given data. Another example are support vector machines for a classification problem, which try to find the best separating hyperplane, *i.e.* a linear function, and assume that function explains the data the best.

An alternative approach is to consider all available functions at the same time, and assign a probability to each function according to how well they explain the data. To illustrate the idea better, assume we have a family of functions as our prior (Figure 2.2(a)), and then we observe a few values from the underlying function. If the data is noiseless, only those functions passing all of our observations can be considered (Figure 2.2(b)). Considering all those functions, we can calculate a posterior mean and variance for each unobserved value as shown in Figure 2.2(c) [118].

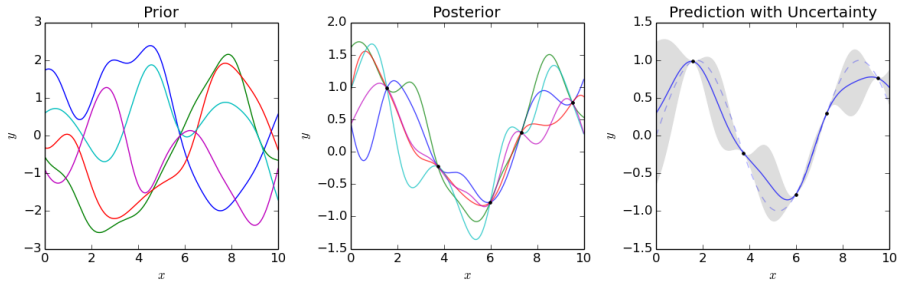


Figure 2.2: (a) Samples from prior family of functions, (b) samples from posterior family of functions, and (c) predicted mean and variance of the posterior[2].

Gaussian processes are particularly useful if not only the mean of the prediction is of interest, but also an estimate of the variance of the posterior probability distribution is important, which is the case for us as explained in section 4.2.

To formulate the above intuition, consider a regression problem, with some observed inputs x_i and corresponding outputs y_i. The goal is usually to find the best function f from a given family of functions such as linear functions, for which $y_i = f(x_i)$. Alternatively, we could infer a distribution over functions given the data, *i.e.* $p(f|\mathbf{X}, \mathbf{y})$, and give predictions for a new input $\mathbf{x}_*$ as shown in formula 2.23 [96].

[2]Image by Cdipaolo96 (`https://en.wikipedia.org/wiki/Gaussian_process`) licensed under CC BY-SA 4.0.

$$p(y_*|x_*, \mathbf{X}, \mathbf{y}) = \int p(y_*|f, \mathbf{x}_*)p(f|\mathbf{X}, \mathbf{y})df \tag{2.23}$$

Fortunately, it turns out given a finite input dataset, predicting the mean and variance of the output given a new input can be done without having to compute the above integral. To explain, here we follow the path in *Pattern Recognition and Machine Learning, Bishop* [17]. Similar to SVMs, consider $\phi(\mathbf{x})$ to be the transformation function for input $\mathbf{x}$, and the following linear model:

$$y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) \tag{2.24}$$

Now assume a Gaussian distribution over the weight vector $\mathbf{w}$ as Formula 2.25, in which α is the inverse variance of the distribution.

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) \tag{2.25}$$

Any sample taken from the above distribution represents a function in Formula 2.24, hence the above distribution defines a distribution over linear functions. Now we are interested in evaluating the function on training data $\mathbf{x}_{\{1...N\}}$, *i.e.* function values $y(\mathbf{x}_{\{1...N\}})$ denoted by the vector $\mathbf{y}$, written as:

$$\begin{aligned} \mathbf{y} =& \Phi\mathbf{w} \\ \Phi_{nk} =& \phi_k(\mathbf{x}_n) \end{aligned} \tag{2.26}$$

The probability distribution of $\mathbf{y}$ is Gaussian since it is a linear combination of elements of $\mathbf{w}$, which are Gaussian distributed. Hence we have:

$$\begin{aligned} \mathbb{E}[\mathbf{y}] &= \Phi\mathbb{E}[\mathbf{w}] = 0 \\ cov[\mathbf{y}] &= \mathbb{E}[\mathbf{y}\mathbf{y}^T] = \Phi\mathbb{E}[\mathbf{w}\mathbf{w}^T]\Phi^T = \alpha^{-1}\Phi\Phi^T = \mathbf{K} \\ \mathbf{K}_{mn} &= k(\mathbf{x}_n, \mathbf{x}_m) = \alpha^{-1}\phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m) \end{aligned} \tag{2.27}$$

Similar to SVMs, $k(\mathbf{x}, \mathbf{x}')$ is called the kernel function. In general, a Gaussian process is a distribution over functions $y(\mathbf{x})$ such that the join distribution of $y(\mathbf{x}_{1...N})$ is Gaussian for any input set $\mathbf{x}_{1...N}$. Since the joint distribution can be specified using the second order statistics, *i.e.* the mean and the covariance of the distribution, a Gaussian process is completely specified given the two statistics. In many applications we do not have a prior knowledge about the mean of $y(\mathbf{x})$ and by symmetry we assume it to be 0. Therefore specification of the covariance function would be the only requirement, which itself is given by the kernel function:

$$\mathbb{E}[y(\mathbf{x}_n)y(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m) \tag{2.28}$$

The kernel function defined in Formula 2.27 specifies a Gaussian process defined by a linear regression. Two other commonly used kernel functions are Gaussian kernel defined in Formula 2.18 and exponential kernels defined as:

$$k(\mathbf{x}_n, \mathbf{x}_m) = exp(-\theta|x - x'|) \tag{2.29}$$

Now given a new test data $\mathbf{x}_{N+1}$, we calculate the kernel matrix and partition it as shown in Formula 2.30.

$$\mathbf{K}_{N+1} = \begin{pmatrix} \mathbf{K}_N & \mathbf{k} \\ \mathbf{k}^T & c \end{pmatrix} \tag{2.30}$$

Then, as shown in *Bishop, 2006* [17, Ch. 6.4], the predicted mean and variance for the input would be:

$$\begin{aligned} m(\mathbf{x}_{N+1}) &= \mathbf{k}^T \mathbf{K}_N^{-1} \mathbf{y} \\ \sigma^2(\mathbf{x}_{N+1}) &= c - \mathbf{k}^T \mathbf{K}_N^{-1} \mathbf{k} \end{aligned} \tag{2.31}$$

2.1.6 Boosting and Ensemble Methods

For a given prediction problem the idea of boosting is to find an optimal combination of classifiers, also called "weak learners", in such a way that a combination of their outputs improves the prediction accuracy. This combination can be majority voting, average, or a weighted average of the outputs of the weak learners [36]. There are many methods of finding the optimal combination of such weak learners, two of which are stochastic gradient boosting [52] and AdaBoost [51]. Stochastic gradient boosting tries to estimate the gradients of the loss function and train each individual weak learner in a way that best improves the loss function. AdaBoost tries to identify samples among given data samples that are harder to classify, and gives them more weight in the process of training individual weak learners. One way of improving AdaBoost is to take into account the confidences of predictions given by weak learners if possible and use estimated confidences in the voting process [126].

2.2 Shortest Path Algortihms for Graphs

A graph G is a set of vertices (also called nodes) V, and a set of edges $E = \{(v_i, v_j)|v_i, v_j \in V\}$ that connect vertices in V. A graph can be directed or undirected. In directed graphs, edges have direction, *i.e.* edge (s, t) is different than the edge (t, s). In other words, the following list shows the possible sets of edges regarding vertices s and t in a directed graph:

$$
\begin{aligned}
E &= \{\} \\
E &= \{(s,t)\} \\
E &= \{(t,s)\} \\
E &= \{(s,t),(t,s)\}
\end{aligned} \tag{2.32}
$$

In an undirected graph however, edges (s,t) and (t,s) are identical, and can in fact be represented as a set $\{s,t\}$ instead of an ordered pair.

Graphs can also be weighted or not. If a graph G is weighted, then there is a weight assigned to each edge of the graph. We use $w_{s,t}$ to note the weight of the edge (s,t). In undirected graphs, $w_{s,t}$ is always the same as $w_{t,s}$. Sometimes the weight of an edge is referred to as the length of an edge and noted as $l_{s,t}$ depending on the context in the literature. A sequence of n nodes $(v_1, v_2, \ldots, v_n) \in V^n$ defines a path p of length n if for every consecutive nodes v_i and v_{i+1}, (v_i, v_{i+1}) is an edge in the graph. The weight or the length of a given path is the sum over the corresponding weights/lengths of its edges. An undirected graph G is connected if there is at least one path between every given two vertices on the graph.

2.2.1 The Shortest Path Problem

The shortest path problem is to find a path between two vertices s and t such that the total weight of the path is the minimum among all possible paths between the two nodes. In unweighted graphs, the weight of each edge is considered to be 1. Figure 2.3 highlights the shortest path between vertices A and F on the given weighted directed graph.

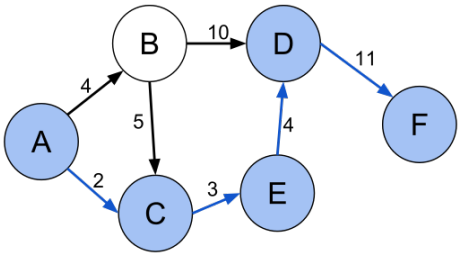


Figure 2.3: A given weighted directed graph and the highlighted shortest path between vertices A and F.

Some prominent algorithms to solve the shortest path problem are Dijkstra's [37], Bellman-Ford [11], and Floyd-Warshall [47] algorithms. Dijkstra's algorithm applies to the single-source shortest path problem on graphs with non-negative weight values with time complexity $O(|E| + |V| \log |V|)$ [50], whereas Bellman-Ford algorithm works on graphs with also negative weights having time complexity $O(|V||E|)$. Floyd-Warshall algorithm, on the other hand, solves the

all pairs shortest path problem, *i.e.* it finds shortest paths between all pairs of vertices, on graphs with negative and non-negative weight values and its time complexity is $O(|V|^3)$. The big O notation defines an asymptotically upper bound of a function up to a constant factor. Formula 2.33 formally defines the notation [31].

$$f(x) = O(g(x)) \iff \exists k > 0 \exists n_0 \forall n > n_0 |f(n)| \leq k|g(n)| \tag{2.33}$$

2.2.2 The k Shortest Paths Problem

The k shortest paths problem is to find the k paths from s to t with minimum weight among all distinct possible paths from s to t. Whether or not loops in the paths are allowed results in two different definitions of this problem.For the case when the goal is to find k best shortest paths from a single source to all other nodes, Jin Y. Yen published an algorithm of the time complexity $O(k|V|(|E| + |V|\log|V|))$ in 1971 for the loopless setting which still has the best available time complexity [159]. It is possible to achieve better worst case time complexity if we allow loops in the paths. In 1998 Eppstein came up with an algorithm with $O(|E| + |V|\log|V| + |V|k)$ time complexity, and $O(|E| + |V|\log|V| + k)$ if the problem is reduced to the single source single destination case [43]. There has been improvements to Eppstein's algorithm, but the worst case time complexity has not been improved.

In our case, the graph is a directed acyclic graph (DAG), *i.e.* there are no directed loops in the graph. Therefore despite we require paths to be loopless, Eppstein's algorithm is sufficient and gives desirable paths. Intuitively, the algorithm starts with the shortest path between s and t, and in each iteration it finds the next shortest path by modifying a part of the previous path. This is achieved by storing a tree of all shortest paths to the destination t, then calculating the cost of jumping from one shortest path to another one using edges that are not a part of that tree (called sidetracks), and at the end picking the sidetrack edges with the least costs. Here we give an overview of the algorithm and postpone our use case in detail to Chapter 3.

First we need to introduce some concepts and notations, and for the sake of easier reference to the original work, we keep the notation as the work done by Eppstein. Assume the problem is to find the k shortest paths from s to t on a connected directed graph G. Then consider the following:

- T: a single destination shortest path tree with destination t, *i.e.* T includes all vertices of G and a shortest path from each node to t.
- $d(v_i, v_j)$: the weight of a shortest path from v_i to v_j, or in other words the distance between the two vertices.
- $head(e), tail(e)$: if e is (v_i, v_j), *head* and *tail* of e are v_i and v_j respectively.
- $l(e)$: weight or length of edge e.

- $\delta(e)$: intuitively the cost of including e in a shortest path to t, defined as:

$$\delta(e) = l(e) + d(head(e), t) - d(tail(e), t) \tag{2.34}$$

If the edge e is not a part of T, it is a *sidetrack* and the cost of including it in a path to t is non-negative [43, Lemma 1].

A key point to understanding the algorithm is the way paths are represented. A path p from s to t can be represented by the list of *sidetrack* edges it includes. If the path p includes only one *sidetrack* edge (v_i, v_j), it means the path is the shortest path from s to v_i, then the edge (v_i, v_j), and then the shortest path from v_j to t. The set $sidetracks(p)$ includes all edges in p that are not in the shortest path tree T, *i.e.* they are in $G - T$. The graph $G - T$ is defined as the graph G excluding edges that are present in the graph T.

To calculate the length of the path p we have [43, Lemma 2]:

$$l(p) = d(s, t) + \sum_{e \in sidetracks(p)} \delta(e) \tag{2.35}$$

Given a path p, let $S = sidetracks(p)$ be the sequence of edges of p that are in $G - T$. We also define $path(S)$ as the function calculating the path p from a given S. Next, we define $prefix(S)$ to be the sequence of edges in S except the last one. Therefore $prefix(S)$ can define a path as $prefpath(p) := path(prefix(S))$.

Next we have: if the path p is from s to t in G and has a nonempty $sidetracks(p)$, then $l(p) \geq l(prefpath(p))$ [43, Lemma 3]. Please note that $sidetracks(p)$ has to be nonempty or else $prefix(S)$ and hence $prefpath(p)$ is undefined. As a corollary of Lemma 2 and 3 we can construct a natural tree of paths which is also a heap style tree. It is a tree in a way that each node is a path p, and it has all possible paths p' for which $prefpath(p') = path(p)$. It is also a heap style tree in a way that the length of a parent node is less than or equal to all its children.

To overcome this challenge each path p, roughly speaking, is replaced by a heap of the edges that have tails on the path from $head(lastedge(p))$ to t and ordered by $\delta(e)$. Then using two intermediate directed acyclic graphs $D(G)$ [43, Lemma 4] and $P(G)$ [43, Lemma 5], a heap $H(G)$ [43, Lemma 6] is constructed with the following properties:

- $H(G)$ is a 4-heap;
- There is a bijection mapping between nodes in $H(G)$ and $s - t$ paths in G;
- The length of an $s - t$ path in G is $d(s, t)$ plus the weight of the corresponding node in $H(G)$.

Finding k smallest nodes in a min-heap costs $O(k \log k)$, which can be further improved by Frederickson's technique [49], and hence the time complexity of the Eppstein's algorithm [43, Lemma 7].

Although Eppstein's algorithm has the best know worst-case time complexity, it can be shown that in practice we can achieve faster running times by constructing some parts of the algorithm's intermediate structures as they're needed.

In 2003 Víctor M. Jiménez and Andrés Marzal published the modified version of the algorithm and a more detailed explanation of Eppstein's algorithm [70].

2.3 Cell Biology

In order to understand cancer, we need some basics of cellular molecular biology, most importantly the central dogma of molecular biology which shows how information is transferred and transformed inside cells. The central dogma deals with three types of molecules: Deoxyribonucleic acid (DNA), Ribonucleic acid (RNA), and protein. The flow of information between these three types of molecules is depicted in Figure 2.4.

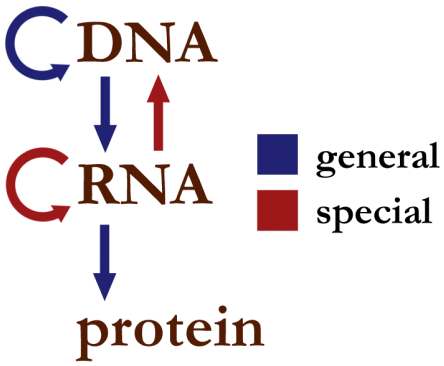


Figure 2.4: Flow of information in biological cells. Blue arrows show the usual flow, and the red arrows show the flow in some special cases.

Although most of the explanations in this section apply to all cellular organisms, for the sake of simplicity we focus on multicellular eukariotic organisms, *i.e.* we assume cells have a nucleus and organisms have organs. In this section we cover a minimal background required to explain and understand the basics of the biology of cancer. For an extensive explanation of these topics refer to *"Molecular Biology of the Cell, alberts, et al."*[7].

2.3.1 Deoxyribonucleic acid (DNA)

DNA molecules are polymers, mostly made of four different unit types called nucleotides: pyrimidines (thymine (T), cytosine (C)) and purines (adenine (A) and guanine (G)). DNA is usually in the form of a double stranded helix, and the two strands complement each other, *i.e.* T complements A and C complements G (Figure 2.5).

DNA is the genetic code that is carried on from cell to cell, and generation to generation. All the cells in an organism have the same genetic material. In

[3]Image by Zephyris (`https://en.wikipedia.org/wiki/DNA`) licensed under CC BY-SA 3.0

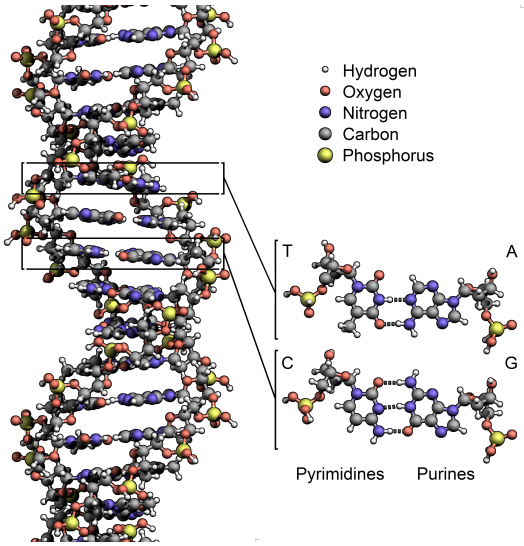


Figure 2.5: DNA double helix and base pairs[3].

computational biology we usually think of DNAs as long strings with *T, C, G, A* as characters. But it is important to remember that for each given string, there is a complement attached to it.

2.3.2 Ribonucleic acid (RNA)

RNA molecules are polymers like DNA, but they carry uracil (U) instead of thymine (T). They are much shorted polymers compared to usual DNAs. There are different types of RNAs with different functions and messenger RNAs (mRNA) are the ones we are interested in, in the context of the central dogma. We can think of RNAs as strings of *U, C, G, A* [7, Ch. 6].

mRNAs are constructed in a process called *transcription* by reading a part of a strand of the DNA and constructing its complement nucleotide by nucleotide, except whenever whenever a thymine (T) is required, instead a uracil (U) is used. This process is shown by a dark blue arrow from DNA to RNA in Figure 2.4 [7, Ch. 6]. The term *gene expression* refers to the rate at which genes are transcribed and mRNAs are synthesized from them. In other terms, a more active gene has a higher gene expression level.

2.3.3 Protein

Proteins are polymers made of amino acids. There are 20 different amino acids in humans. Proteins form and perform most of structures and functions in cells. Enzymes and cell membrane, a.k.a. cytoskeleton [7, Ch. 16], are two examples of molecules and structures mostly made of proteins.

A process called translation, translates mRNA strands into protein strands. In this process, a ribosome complex gets attached to the start codon near the beginning of the mRNA (usually AUG), and then the mRNA strand is decoded codon by codon by tRNAs. In the decoding process, the mRNA molecule is processed three nucleotides at a time, each 3 encoding and representing a specific amino acid. A tRNA is a small RNA molecule which can attach to one codon on one side, and has an amino acid attached to its the other side. This process is continued until a stop codon (UAA, UAG, or UGA) is reached. Figure 2.6 illustrates a simplified version of this process. Translation is shown as a dark blue arrow from RNA to protein in Figure 2.4.

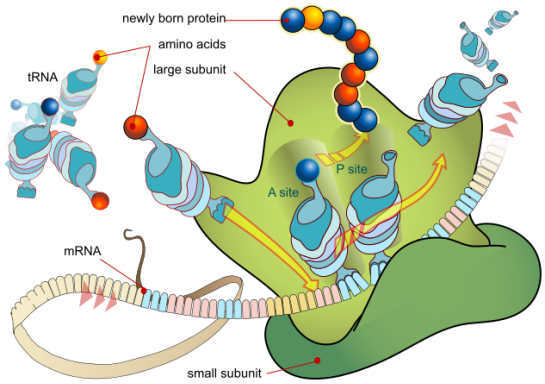


Figure 2.6: A ribosome translating an mRNA with the help of tRNAs.

2.3.4 Pathways

Cellular processes involve collaboration of several molecules (RNAs and proteins included) in the form of a long chain of reactions. A chain of reactions with a specific goal is called a pathway. Among other things, pathways can result in production of a molecule, a change in the cell, activating or deactivating a gene, or to make the cell move. A graphical representation of a pathway has molecules as nodes and reactions and dependencies as edges. Figure 2.7 shows a graphical representation of the apoptosis pathway which results in programmed cell death, whose significance is explained shortly [19, 92, 79, 157, 105]. As you can see, the same proteins and RNAs are used in different reactions, and reactions are interdependent, *i.e.* the product of a reaction is a prerequisite of another reaction. It is also important to note that pathways are not mutually exclusive. The same proteins and other molecules may be used in several pathways.

[4] `http://www.wikipathways.org/index.php/Pathway:WP254`

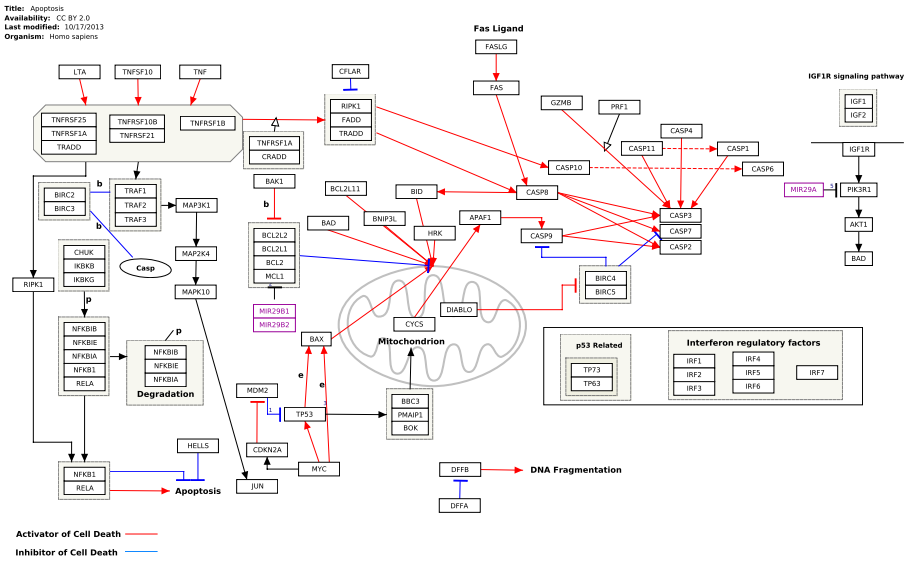


Figure 2.7: Apoptosis (programmed cell death) pathway in homo sapiens.[4]

2.3.5 Cell Reproduction

A new cell is created only when another cell duplicates, which itself is the result of a delicately ordered set of events and stages. The set of events ending with a cell division is called the cell cycle [7, Ch. 17]. Figure 2.8 illustrates the cell cycle, giving each phase approximately the time it takes the phase to complete in proportion to the whole cycle.

During the S phase (S for DNA Synthesis) a complete copy of the cell's DNA is produced. During the M phase, first the nucleus is devided into two, *i.e.* mitosis, then the whole cell divides into two cells, *i.e.* cytokinesis). For a cell which has an approximately a 24 hour cycle, the M phase takes only one hour. G_1 and G_2 are gaps between the S and the M phase. During the G_2 phase, some processes make sure that the DNA is replicated properly and completely, and the cell is prepared to enter the M phase. During G_1 phase, the cell grows, and it only enters the S phase if the environment and conditions are favourable. G_1 may take a long time, and it can also enter the G_0 state, or resting state. A cell may stay in G_0 for years or even indefinitely until cell death. Human nerve cells, for instance, enter this state early in the body's development and they never duplicate. The switch between different phases are controlled by the cell's cell cycle control system which takes into account the cell's environment using the signals received from the surroundings, and the internal cell conditions using some feedbacks received from the cell's development in different stages.

[6]Image by Zephyris (`https://en.wikipedia.org/wiki/DNA`) licensed under CC BY-SA 3.0

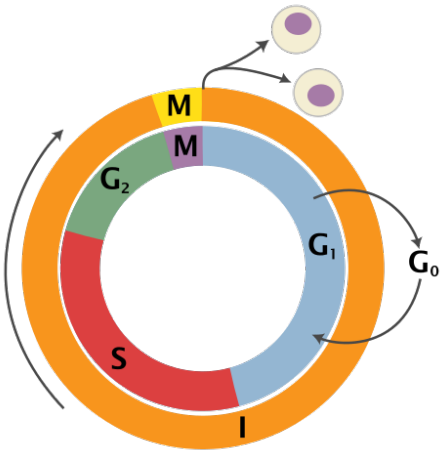


Figure 2.8: Cell cycle: I: interphase, M: mitosis, G_0: resting, G_1: gap 1, S: DNA Synthesis, G_2: gap 2.[6]

2.3.6 Cell Death

Programmed cell death is as important as cell division and reproduction to a healthy tissue. An organism or a tissue can only maintain its size if cells die with the same rate as they divide, otherwise the tissue will keep growing uncontrollably. It is also critical for the cells to die in an orchestrated way during fetus development for limbs and tissues to take their desired form. A third case for programmed cell death is when cells are damaged or infected, to make sure they are removed before threatening the organism's health [7, Ch. 18].

In most cases this programmed cell death occurs from within the cell via apoptosis, *i.e.* through a set of processes and pathways which result in the cell to shrink and die and then to be eaten by other responsible cells. If the cell is large, it will be dismantled into membrane enclosed pieces. The membrane is also altered to give other cells the signal to eat them quickly. Apoptosis ensures that the contents of the cell are not spilled over other cells and that the remains are digested quickly.

In contrast to apoptosis, cells which die as a result of a physical trauma or lack of blood supply usually go through cell necrosis resulting in the swell and burst of the cell triggering an inflammatory response. As we will discuss in Section 2.3.9, damages to pathways related to apoptosis play a role in cancer.

2.3.7 Epigenetics

As stated in *An operational definition of epigenetics* [15]: "An epigenetic trait is a stably heritable phenotype resulting from changes in a chromosome without alterations in the DNA sequence". Two of the most studied such alterations are histone modification and methylation/demethylation of the cytosine in a CG

sequence, which is usually referred to as DNA-methylation. Figure 2.9 illustrates epigenetic alterations on the DNA sequence.

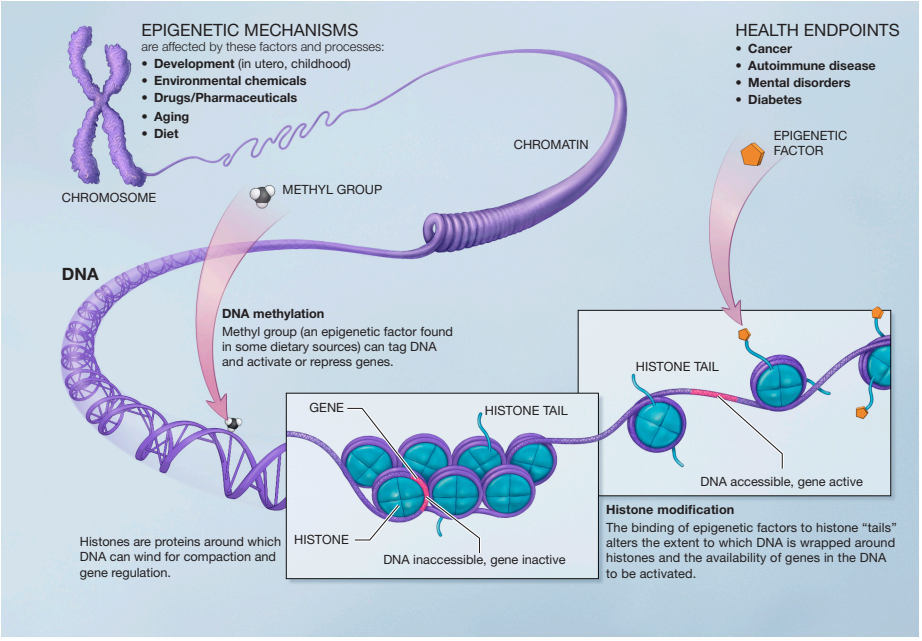


Figure 2.9: Epigenetic alteration mechanisms on the DNA sequence[7].

If a methyl component is attached to a DNA nucleotide, that nucleotide is called *methylated*. In vertebrate cells, except stem cells, methylation happens mostly on a cytosine in a CG sequence, on both sides of the DNA strand. During cell division this property is usually kept and inherited by the offspring cells. A methylated upstream of a gene can suppress that gene's expression. In human and mice cells, it is shown that DNA methylation can inhibit the activity of an entire chromosome [7, Ch. 7].

Histones are proteins acting as cylinders around which a DNA molecule winds and therefore is packed. As a result of this condensation, a DNA fiber of the length $180mm$ and $30nm$ is compressed into a $12\mu m$ long and $700nm$ thick molecule [119]. Histones have a tail which can be altered, *e.g.* methylation or acetylation of some of the amino acids, and some of these alterations are passed to the progenies of the cell, and hence are epigenetic markers [144, 15, 91]. Presence, absence, or alterations of histones near a gene can activate, deactivate, or regulate the expression of that gene [144].

[7]Image available at `https://commonfund.nih.gov/epigenomics/figure`

2.3.8 Innate and Adaptive Immune System

Multicellular organisms use a mechanism called innate immune response to defend themselves against pathogens, through some physical barriers such as the sweat on the skin or saliva in the mouth, as well as biochemically responding to common patterns presented by pathogens called *pathogen-associated molecular patterns (PAMPs)* using receptors called *pattern recognition receptors (PRRs)* [7, Ch. 24]. These PRRs detect different categories of PAMPs, triggering an inflammatory response, which in turn in animals, proceeds with the engulfment of the pathogen by a phagocytic cell. Phagocytes, macrophages, natural killer cells (NK cells), and dendritic cells are among cell types comprising the innate immune system.

Dendritic cells, present in vertebrates, are a crucial component connecting the innate immune system to the adaptive immune system. They detect a large variety of PAMPs by expressing an exhaustive variety of PRRs on their surface. Once a dendritic cell detects a pathogen or one of its products, it engulfs it through phagocytosis, through which it also becomes an activated dendritic cell. These activated cells then travel to a nearby lymphoid organ such as a lymph node to activate T-cells (discussed bellow) and present the invading pathogen to them.

At some point during the evolution, vertebrates developed an adaptive immune system, in theory capable of detecting and remembering any foreign pathogen. The core of this system are the T and B lymphocytes, commonly referred to as T and B cells. They are called T and B lymphocytes because they mature in the *B*one marrow and the *T*hymus respectively. Through their development, these cells undergo an inheritable somatic hyper-mutation process resulting in the expression of virtually all possible receptors and antibodies. This process, however, sometimes produces some lymphocytes detecting the organism's own cells. Therefore a separate process called immunological self-tolerance ensures lymphocytes detecting one's own cells are either destroyed or deactivated. A failure to do so causes an allergy or an autoimmune disease [7, Ch. 24].

Both B and T cells differentiate from hematopoietic stem cells residing in bone marrow. A product of this differentiation are lymphoid progenitor cells, some of which stay in the bone marrow and develop to become B cells, and some others move to the thymus and become thymocytes and then T cells [7, Ch. 24]. The differentiation does not end at this stage, and these two types of cells will further develop into more specific cells such as effector B/T cells, naïve B/T cells, and memory B/T cells.

In simple terms, the adaptive immune response can be seen as two main processes. A foreign pathogen may trigger either of these two or both processes: antibody response and T cell mediated immune response. The antibody response involves B cells producing specific antibodies which bind to the targeted pathogen, disabling them from binding to the organism's own cells and also marking them for destruction. These antibodies circulate in blood and other body fluids reaching and detecting pathogens in the whole organism. T cell mediated immune response, on the other hand, works by T cells detecting some cell surface proteins

called MHC proteins. These proteins are encoded in *major histocompatibility complex* and are expressed in most cells of vertebrates. These MHC proteins can carry some fragments of the products of a pathogen from inside the cell to its surface. When dendritic cells present these pieces to T cells in a lymph gland, the T cells detecting those proteins are activated. These activated cells divide and differentiate into memory T cells and effector T cells. The effector T cells travel to the site of the infection and locally detect the infected cells, marking them for destruction [7, Ch. 24].

2.3.9 Cancer

Thinking of an organism as a society of cells helps to better understand and explain cancer. In a healthy and functional organism, cells collaborate, do not invade each other's space, and sacrifice themselves through programmed cell death to control their own population. A group of cells reproducing abnormally and faster than the usual rate leads to a neoplasm. If these cells are not invasive to the neighboring cells ,the formed tumor is called benign. On the other hand, if the cells start reproducing as wells as invading the surrounding tissue, it then is called a malignant tumor, or cancer [7, Ch. 20].

A malfunction in a pathway related to the cell cycle can cause cells to reproduce uncontrollably and/or not die according to the plan. These malfunctions can be caused by over or under expression of a gene resulting in excess or lack of a protein or an RNA in a pathway. A mutation in the upstream of a gene, e.g. the promoter region of the gene, or certain epigenetic changes such as methylation of the upstream region of a gene can lead to under expression or completely disable the expression of that gene. On the other hand, demethylation of the promoter region might enhance the expression of the gene and disrupt the related pathways. It can also be the case that a mutation on a specific gene renders a protein dysfunctional, hence a malfunction in the pathways to which it belongs [44].

Even without the presence of mutation inducing agents, mutations naturally occur on the DNA during cell division, most of which are corrected due to DNA repair mechanisms present in the cell. Out of the mutations which remain on the DNA, only a few can lead to a malignant cell. There is also strong evidence that some cancer causing mutations need to be present on both chromosomes for it to cause cancer, unless that mutation already exists on one of the chromosomes, inherited from one of the parents; in which case, only one mutation on the right position is enough to activate/deactivate that corresponding gene [93].

Cancer cells also go though some morphological changes which are visible under a microscope. These cells have a larger nucleus and a smaller cytoplasm, therefore a larger nucleus/cytoplasm ratio, and both nucleus and the cell have irregular shapes. Investigation of these characteristics play a role in cancer diagnosis [8]. Figure 2.10 depicts some of the differences between normal and cancer cells.

Since cancer cells are evolved from the cells of the organism itself, they are not detected by the immune system as enemies. These cancerous cells also know all the internal protocols of the organism. For instance, as soon as a cancerous

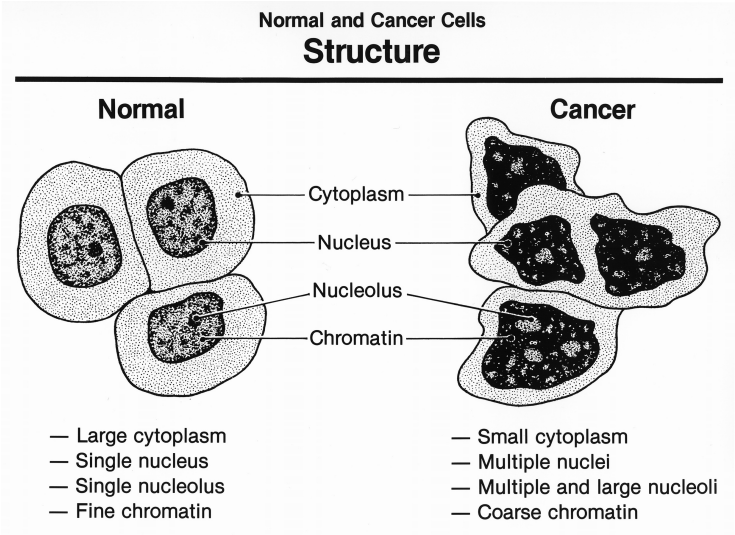


Figure 2.10: Some morphological differences between normal and cancer cells.

cell grows into a tumor-like mass, it needs more blood supply. Therefore it sends signals triggering new veins to develop to support the tumor with the food and oxygen it needs. The cancerous cells may then enter the blood stream of lymphatic vessels, land on another part of the body, and form a secondary tumor, *i.e.* metastasis [7, Ch. 20].

2.3.10 Lymphoid Neoplasms and Lymphoma

Like other diseases, cancer had traditionally been categorized according to the symptoms and the organ in which it appears. With the advancements in medicine, we could detect metastasis and therefore classify the disease according to the organ from which it originates. This classification, however, is far from perfect. Depending on the underlying genetic cause of the cancer, it can practically be considered different diseases although happening on the same organ. For instance, breast cancer was considered one disease for thousands of years, until it was discovered that tumor cells with estrogen receptors on their surface as inhibitors, *i.e.* ER+ breast cancer tumors, are a different disease than the ones not having estrogen as an inhibitor. This discovery resulted in subclassification of cancer into ER+ and ER- tumors improving diagnosis, treatment, and prognosis specially for patients with ER+ tumors [93].

When it comes to the cancers of the immune system, a few factors make the classification of the disease even more complicated. One is the location of the cancer, which can be any place along the development path of our immune system, bone marrow, thymus, blood stream, and lymph nodes included. For this reason, cancers of the immune system used to be put in two main classes: (1) if

the disease appears only in blood stream and does not form a mass: leukemia (2) if it forms a mass mainly where lymph nodes are located: lymphoma [139, 160]. However, our better understanding of the immune system and these diseases proved traditional classification of these diseases inefficient at best, and as a result, WHO classification of these malignancies now puts them all together under the category of *lymphoid neoplasms* [100, 139]. For example, Chronic lymphocytic leukemia and small lymphocytic lymphoma are now considered two different stages of the same disease affecting mature B cells. Generally speaking, lymphoma are neoplasms of the immune system resulting in a solid tumor [160, Ch. 1].

As mentioned in Section 2.3.8, B and T cells both undergo somatic hypermutation. B cells, for example, once activated by an antigen, undergo a somatic mutation phase with a rate at least 100,000 times higher than the normal rate of mutation across the genome [89]. Although the hypermutation phase is highly regulated, sometimes errors happen which can lead to a possible malignancy.

The first level of lymphoma classification creates two main categories: Hodgkin and non-Hodgkin lymphoma (NHL). The disease is considered Hodgkin lymphoma if Reed-Sternberg cells are present in the biopsy. These cells are large lymphocytes usually derived from B cells and can have more than one nucleus, as shown in Figure 2.11 [135].

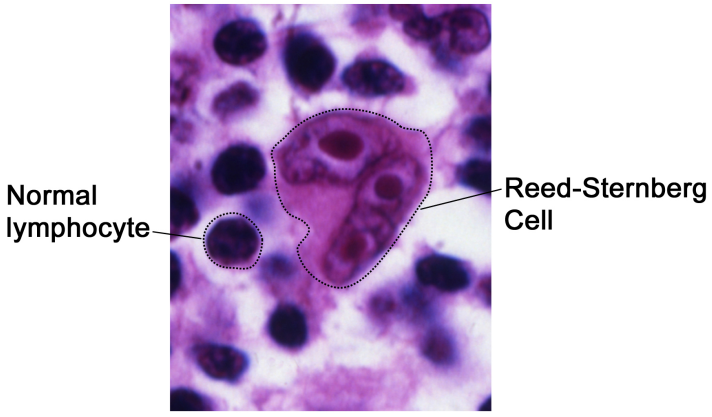


Figure 2.11: Reed-Sternberg cells in a sea of normal lymphocytes, indicative of Hodgkin lymphoma.

Like Hodgkin lymphoma, the classification of non-Hodgkin cases also started by focusing on morphological features of the cancer cells. However, through improved understanding of the disease, immunologic, genetic, and clinical criteria are now also considered to better distinguish between cancer subtypes [64, Ch. 33]. For instance, in the case of anaplastic large T cell lymphoma (ALCL), there are treatments if an anaplastic lymphoma kinase (ALK) translocation (t[2,5])

is present in the neoplasm, leading to a much better prognosis. As a result, a classification of ALCL now includes ALK^+ and ALK^- subtypes [89, Ch. 29]. Our focus in this work is on two major types of mature B cell neoplasms, namely Follicular Lymphoma (FL) and Diffuse Large B Cell Lymphomas (DLBCL).

DLBCL is a high grade aggressive NHL and the most common among NHL cases. FL on the other hand, is the most common low grade NHL and is only second to DLBCL in terms of the number of NHL cases. Both FL and DLBCL are heterogeneous diseases and have several subtypes of their own. FL is defined as a neoplasm of B cells in germinal centers, comprising different proportions of small centrocytes and large centroblasts with a follicular growth pattern, as shown in Figure 2.12 [64, Ch. 31]. Its cases are divided into three main grades, and the third grade divided into 3a and 3b. Except FL-3b, the other cases progress slowly and all have a similar prognosis outcome. Grade 3b, on the other hand, is more aggressive and is clinically closer to DLBCL [64, Ch. 33].

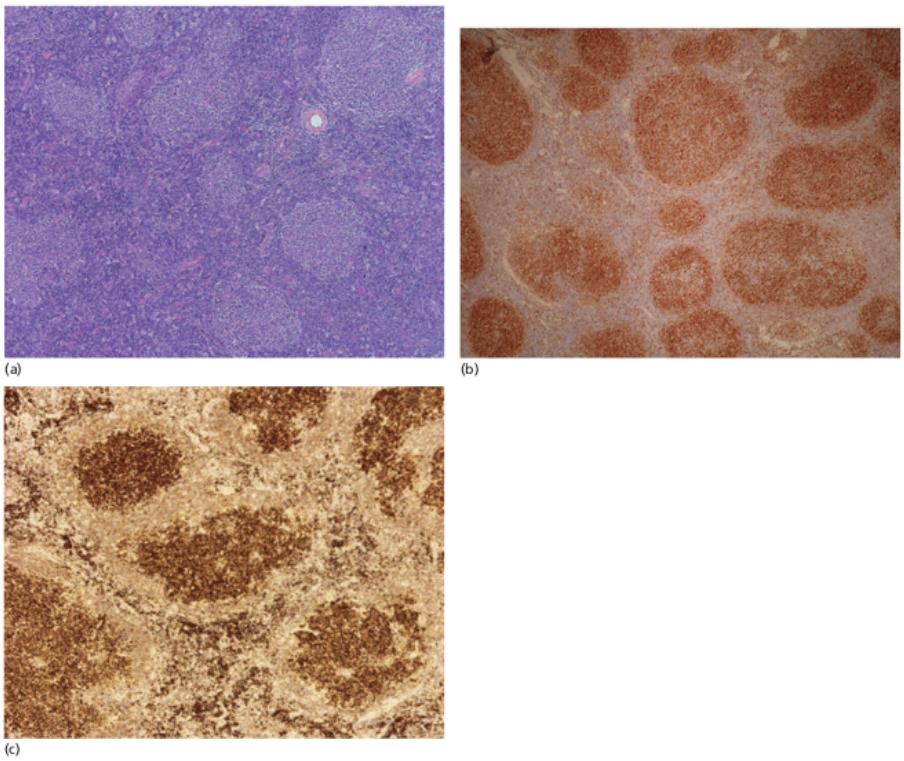


Figure 2.12: Follicular lymphoma. (a) The tumour grows in a follicular pattern with expanded germinal centres without macrophages. The tumour cells are positive for CD10 (b) and BCL-2 (c).[9]

[9]Image and caption taken from *Postgraduate Haematology*, p. 582 [64, Ch. 31].

DLBCL is a neoplasm of B cells with the commonality of having a diffuse growth pattern. It includes a very diverse set of subtypes. There are also classes of neoplasms categorized between DLBCL and other types of lymphoid neoplasms such as Hodgkin lymphoma or Burkitt lymphoma. Although in many cases it is not clear what the background of the occurring DLBCL is, in some rather rare cases it can be a transformation from other diseases such as FL [64, Ch. 31, 33].

Since many lymphoid neoplasms are heterogeneous diseases each with a different underlying genetic cause, accurate classification of each case using gene expression profiling (GEP) or next generation sequencing (NGS) enables oncologists and pathologists to target specific genes, inhibiting or activating them, in order to activate apoptosis or inhibit cell growth as a method of treatment. However, since it is not feasible to use GEP or NGS for all patients due to their complexity and costs, immunophenotyping is often used as a proxy to diagnose each patient's subtype. In this work, Chapter 3 deals with immunophenotype data, and then Chapter 4 explores analysis of genetic and epigenetics data retrieved from biopsies of cancer patients.

3
Flow Cytometry Analysis

As mentioned in the introduction (Chapter 1), as a part of this work we touch on the data and problems available and related to clinics. For this purpose, we tackle some problems using flow cytometry data which is available on a daily basis for immunophenotyping and diagnosis purposes. This chapter first introduces flow cytometry, and then our contributions related to flow cytometry data. It is also worth mentioning that Section 3.2.2 and 3.2.3 are previously published articles both well received and cited by the community. Since both these articles are the result of equal contribution of 2 and 3 authors (including me), it is important to note that I do not claim the credit for the whole work and I will outline my personal contribution to these works at the end of this chapter. However, due to space journal limitations not all required background were included in them and therefore some of the required background are included in Chapter 2 and some here in this chapter.

3.1 Flow Cytometry

Flow cytometry is a technology that allows measurement of biomarkers inside and outside cells on a single cell basis [68]. The technology can also sort and separate certain cells according to a given criterion [53, 68].

Cell preparation in flow cytometry involves suspension of the cells in a liquid containing biomarker reagents. These biomarkers are antibodies which usually attach to proteins on the surface of cells' surface. Some markers can penetrate the cell's membrane and attach to a protein inside the cell, and hence are called intracellular markers which tend to be more expensive than surface markers and often result in cell death, therefore they are not as commonly used as surface

markers. Reagents are marked antibodies that can be detected by the laser beams in the flow cytometer machine (FCM) [131]. The antibodies are usually marked with a fluorescent label. Each fluorescent marker has a corresponding peak excitation and emission wavelength which can be detected using lasers or lamps available on the flow cytometer machine. The combination of markers has to be chosen such that their corresponding wavelengths have minimal overlap; otherwise they cannot be distinguished from one another due to interference between them.

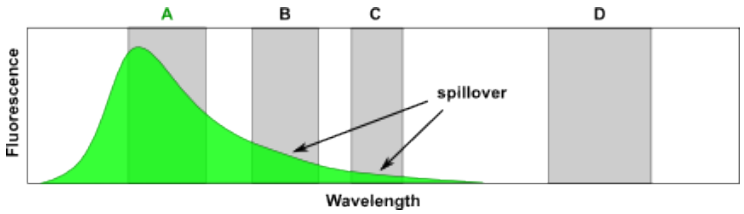


Figure 3.1: Fluorescence of a green flurochrome (e.g. FITC) is primarily detected by detector A. However as the emission spectrum is relatively broad some of the fluorescence is detected also by detectors B and C. This is called fluorescence spillover and needs to be corrected for otherwise it could compromise detection of other fluorochromes by their appropriate detectors[1].

In a flow cytometer cells flow in a liquid stream one by one, where lamps or laser beams in conjunction with sensors measure the intensity of reflected light from the cells. These measured values can be in linear or logarithmic space [131]. The measured values depend on the light intensity projected onto cells which can be tuned by changing the voltage of the lasers or lamps. Different wavelengths correspond to different markers, but they might overlap. When the tail of the emission spectrum of a marker overlaps with the main part of the emission spectrum of another marker, it is called *spillover* as shown in Fig. 3.1 [122].

Compensating for spillover requires a spillover matrix (SM). $SP_{i,j}$ shows the percentage that marker i spills over marker j. The compensation matrix (CM) is then the calculated as the inverse of the spill over matrix. Let S be the true signal value, and O be the observed value. Then we have[1]:

$$\begin{aligned} CM &= SM^{-1} \\ S &= O \times CM \end{aligned} \tag{3.1}$$

This operation is often called *compensation*. It is clear from Formula 3.1 that this is a linear operation, which means we need to transform the data onto a linear space if required. The measured fluorescent intensities almost exponentially correspond to the number of existing fluorescent markers on or inside the cell. Therefor a proper transformation of the raw data is essential in order to have the

[1] `http://bioinformin.net/cytometry/compensation.php`

Marker	Cell types
CD2	T cell, natural killer (NK) cell
CD3	T cell
CD4	T helper cell, monocyte, macrophage, dendritic cell
CD5	T cell
CD7	Thymocite, mature T cell
CD8	Thymocite, cytotoxic T cell, natural killer cell, dendritic cell
CD19	B cell
CD20	B cell
CD27	T cell
CD28	naive T cell
CD33	myeloid
CCR5	T cell
CCR7	several lymphoid tissues

Table 3.1: Some common markers and cells expressing those proteins. CD: *cluster of differentiation*, CCR: *C-C chemokine receptor*.

data in a linear space. Logarithmic, log-linear hybrid transformation Logicle [106], and hyperbolic arcsine are some commonly used transformations [101]. Some studies have compared different transformation techniques and reported their advantages and disadvantages [46, 113].

In practice data are produced through time and also maybe in different labs. This means different reagent batches are used, and flow cytometry machines are not necessarily calibrated alike, which also affects compensation matrices. Therefore normalization is a crucial step to make samples comparable [60].

Different cell types express different proteins on their surface and inside them. As a result, we can use flow cytometry to detect and sort different cell types. Table 3.1 lists some of the usual markers used to identify different immune cells, which are relevant to the rest of this chapter. Neither the list of proteins nor the list of cells expressing them is complete. For a more exhaustive list the reader is referred to *Janeways's Immunobiology* book [95].

There are two other measurements taken by a flow cytometer machine, namely Forward Scatter (FS) and Side Scatter (SS), indicating volume and morphological complexity of cells respectively. And the end, if we assume $20,000$ cells are passed through the FCM, stained with 8 markers, including FS and SS, the resulting data will be a $20,000 \times 10$ matrix, each of those 10 columns being a *dimension* of the data.

3.2 High Dimensional Analysis and Visualization

Manual analysis of flow cytometry data involves *gating*. Researchers use density or scatter plots of one or two selected dimensions of flow cytometry data in order

to visualize and also select some areas on those plots to further investigate cells within the selected area. Visualization and further gating of those selected cells is commonly a next step to the analysis, *i.e.* plotting only those selected cells using two other dimensions for further gating.

Manual gating of cells across several samples is a labor intensive and time consuming process. Not being able to analyze the data in its original higher dimensional space is another disadvantage of manual flow cytometry data analysis.

The rest of this chapter first explains how we extract features from flow cytometry data using flowType [2]. Then we show how RchyOtimyx uses the outcome of flowType to summarize and visualize gating strategies [4]. Then an improvement to both functionality and performance of both packages, as well as a pipeline using both methods are presented.

3.2.1 Cell Population Identification: flowType

Assume there is a threshold corresponding to each marker/dimension for a given flow cytometry data. For a given marker $\mathbf{M}$ and the threshold t, cells are divided into two groups regarding the threshold as shown in Formula 3.2.

$$\begin{aligned} \mathbf{M}^- &:= \{c_i | \mathbf{M}(c_i) < t\} \\ \mathbf{M}^+ &:= \{c_i | \mathbf{M}(c_i) \geq t\} \end{aligned} \tag{3.2}$$

$\mathbf{M}(c_i)$ is the observed value of marker $\mathbf{M}$ for cell c_i. Identifying a cell population, some markers might be irrelevant. If a marker does not play a role defining a population, we call it neutral. Therefore regarding each marker, there are 3 possible populations, *i.e.* positive, negative, neutral. Figure 3.2 shows all possible populations considering only two markers[2]. Note that a cell population identified by more than one marker is the intersection of the two corresponding sets of cells. As a result, given k markers, there are 3^k possible cell populations to identify, given the above assumptions.

The flowType method[3] iterates through all possible combinations of cell populations and reports cell count and/or Mean Fluorescence Intensity (MFI) of each dimension for each cell population. Given m markers, there are 3^m possible combinations, and flowType reports all of them. Some possible further analysis using these reported cell counts and/or MFIs are explored in the original publication [2].

3.2.2 Hierarchical Analysis of Cell Populations: RchyOptimyx

Recent advances in FCM instrumentation and reagents have enabled high-dimensional analyses to identify large numbers of cell populations with potentially significant correlations to an external outcome. However, studies often fail

[2]Credit: Figure 1-A of [2]

[3]`https://www.bioconductor.org/packages/release/bioc/html/flowType.html`

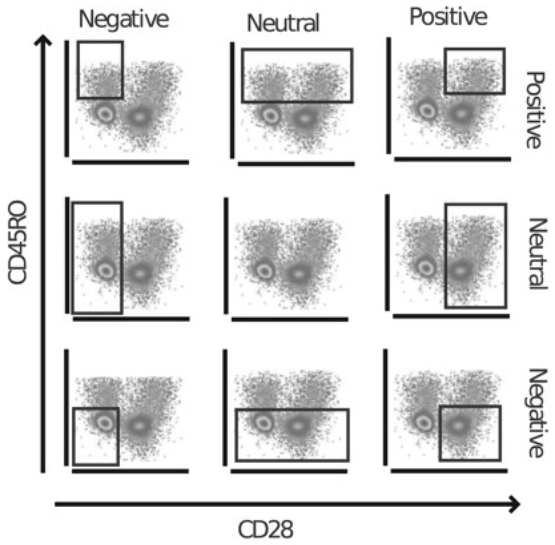


Figure 3.2: Population identification

to characterize the complex relationships between the markers involved in the identification of these cell populations. Revealing this information can provide additional insight into the biological characteristics of the populations identified. The choice of markers for new panels has been a source of ongoing debate, including efforts such as the Human ImmunoPhenotyping Consortium (HIPC), the Federation of Clinical Immunology Societies Federation of Clinical Immunology Societies (FOCiS) sponsored Flow Immunophenotyping Technical Meetings (FITMaN), and the Optimized Multicolor Immunophenotyping Panels (OMIPs) articles [85, 124, 86, 28, 152, 16, 48, 94, 42, 165, 80, 110]. Understanding the relationships between the markers involved in identification of the target cell population and the characteristics of that cell population (*e.g.,* its correlation with a clinical outcome) is fundamental to the design of effective marker panels. For example, one could use a high-dimensional flow or mass cytometry assay to measure a large list of candidate markers. However, this can result in parsing the cells into (*e.g.,* clinically) redundant subsets [12]. Excluding these redundancies (*e.g.*, markers less important for prediction of a clinical outcome) will result in a panel of the most clinically relevant markers.

High dimensional FCM data is usually analyzed using a laborious sequential manual analysis procedure in which a series of thresholds or 2-dimensional polygons (or gates) are applied to histograms or scatter plots of markers (*e.g.,* [108, 55]). However, manual gates provide little insight into the relative importance of each gate to the final results. For example, consider a six color assay with markers named 1 to 6. If the expression of each marker is considered to be on, off, or does not matter (*e.g.,* markers named 1, 2, and 3 in phenotype 1^+2^-, respectively), a total of $3^6 = 729$ cell populations can be distinguished based on

these markers. A given immunophenotype involving all six of these markers (*e.g.,* $1^+2^-3^+4^-5^+6^-$) can have $2^6 = 64$ parent populations (*e.g.,* 1^+, 1^+2^-). Quantifying the relationship between the cell population of interest and these parent populations is fundamental to our understanding of the importance of the markers for different gating strategies. The order in which the gates are applied to the data is not important, as long as all of the gates are used (*i.e.,* sequential gating is commutative). However, to decrease the size of the marker panel, the relative importance of the gates should be determined. For example, the measurement of the phenotype mentioned above using only five colors requires the determination of the importance of each marker to identify and remove the least important one (*i.e.,* the identification of the parent population with five markers that is most similar to the original phenotype). This is further complicated by the fact that some cell populations can be identified using more than one combination of markers and gating strategy; therefore, each marker can be used in different positions in the gating hierarchy and can have different priorities, depending on the choice of the gating strategy. For example, the 3^+ gate is involved in both $1^+2^-3^+$ and $3^+4^-5^+$, both parents of the $1^+2^-3^+4^-5^+6^-$ phenotype described above. However, depending on the amount of redundancy between marker 3 and others, this marker can have different levels of importance for these two parent populations.

Another use–case for measuring the importance of the markers is the investigation of a large number of closely related phenotypes (*e.g.*, those identified by bioinformatics pipelines) by identifying their common parent populations. Several computational tools have been developed for automated identification of cell populations (*e.g.,* [83, 45, 112, 24, 99, 162, 114, 137, 5, 13, 115]) and recent studies have used these tools to identify novel cell populations that correlate with clinical outcomes (*e.g.,* [2, 161, 33, 123, 10]). In addition, the results of the FlowCAP-II project[4] have shown that several algorithms can accurately and reproducibly identify cell populations correlated with external outcomes. However, these algorithms provide limited information regarding the importance of the markers involved in defining the cell populations [2, 25]. This situation is even more complicated than sequential manual gating, since most of these bioinformatics pipelines work based on multivariate classifiers, and as a result, more than one cell population can be responsible for the final predictions. Therefore, markers can have different relative importance in defining the multiple cell populations within the multivariate model. Quantifying the markers for each phenotype involved in the multivariate model can provide additional insight into the differences between closely related cell populations. For example, if two phenotypes $1^+2^-3^+4^-5^+$ and $1^+2^-3^+4^-6^+$ are identified as correlates of a disease, and if markers 5 and 6 (which are the only differences between them) are the least important markers for the former and latter phenotypes respectively, then these two phenotypes are likely to correspond to the same cell population (as far as the correlation with the disease is concerned). However, if markers 5 and 6 are the most important for the phenotypes, these can correspond to two biologically

[4] http://flowcap.flowsite.org/summit2011.html

different cell populations.

To address these problems, we developed RchyOptimyx, a computational tool that uses dynamic programming and optimization techniques from graph theory to construct a cellular hierarchy, providing the best gating strategies to identify target populations to a desired level of purity or correlation with a clinical outcome, using the simplest possible combination of markers.

Materials and Methods

Our methodology builds on the flowType pipeline[2]. flowType comprehensively identifies cell populations defined by all possible gating strategies (hierarchies) in the data set using a partitioning strategy (*e.g.*, clustering algorithm like flowMeans [2]) and scores them by a statistical test (*e.g.*, the log rank test for difference in survival distributions). Given the list of all cell populations and their scores, RchyOptimyx uses a dynamic programming approach to find the best cellular hierarchy within a reasonable time for interactive data analysis (*e.g.* less than 2 minutes for 30 color data), as well as a number of best suboptimal hierarchies, to enable mining of the space of best gating strategies and purities for a given target cell population.

Terms and Definitions

Let $\mathcal{M}$ be the set of m markers of interest (*e.g.*, $\mathcal{M} = \{KI\text{–}67, CD28, CD45RO\}$), a single marker phenotype be a phenotype having only one marker (*e.g.*, $CD28^+$), a phenotype P be a set of single marker phenotypes (*e.g.*, $P = KI\text{–}67^+CD28^-$), and M (not to be mistaken with $\mathcal{M}$) be a phenotype of size m that involves all of the markers (*e.g.* $M = KI\text{–}67^+CD28^-CD45RO^-$). The power set of M, $\mathcal{P}(M)$, is of size 2^m and contains every possible subset of M. The scoring function $S(.)$ assigns a score to each member of $\mathcal{P}(M)$, such that higher values are assigned to more important phenotypes (*e.g.*, those with a stronger correlation with a clinical outcome).

Given an arbitrary M, the directed acyclic graph (DAG) G_M has $m+1$ levels from 0 to m, each level i including every member of $\mathcal{P}(M)$ of size i. Node s is connected to node t with a directed edge (s, t) if and only if $|t| = |s| + 1$ and the two associated sets of s and t differ only in one single phenotype marker (*i.e.,* t is an immediate parent of s). Let the weight of edge (s, t) be $-S(t)$ (so that paths with maximum score can be found by searching for paths with minimum total weight).

The node with 0 markers is the root (or source) node, and the node with the complete set of markers is the sink node. A path from source to sink is called a hierarchy path, or simply a hierarchy. An example of graph G_M for $M = KI\text{–}67^+CD4^-CCR5^+CD127^-$ is illustrated in Supplementary Figure 3.3.

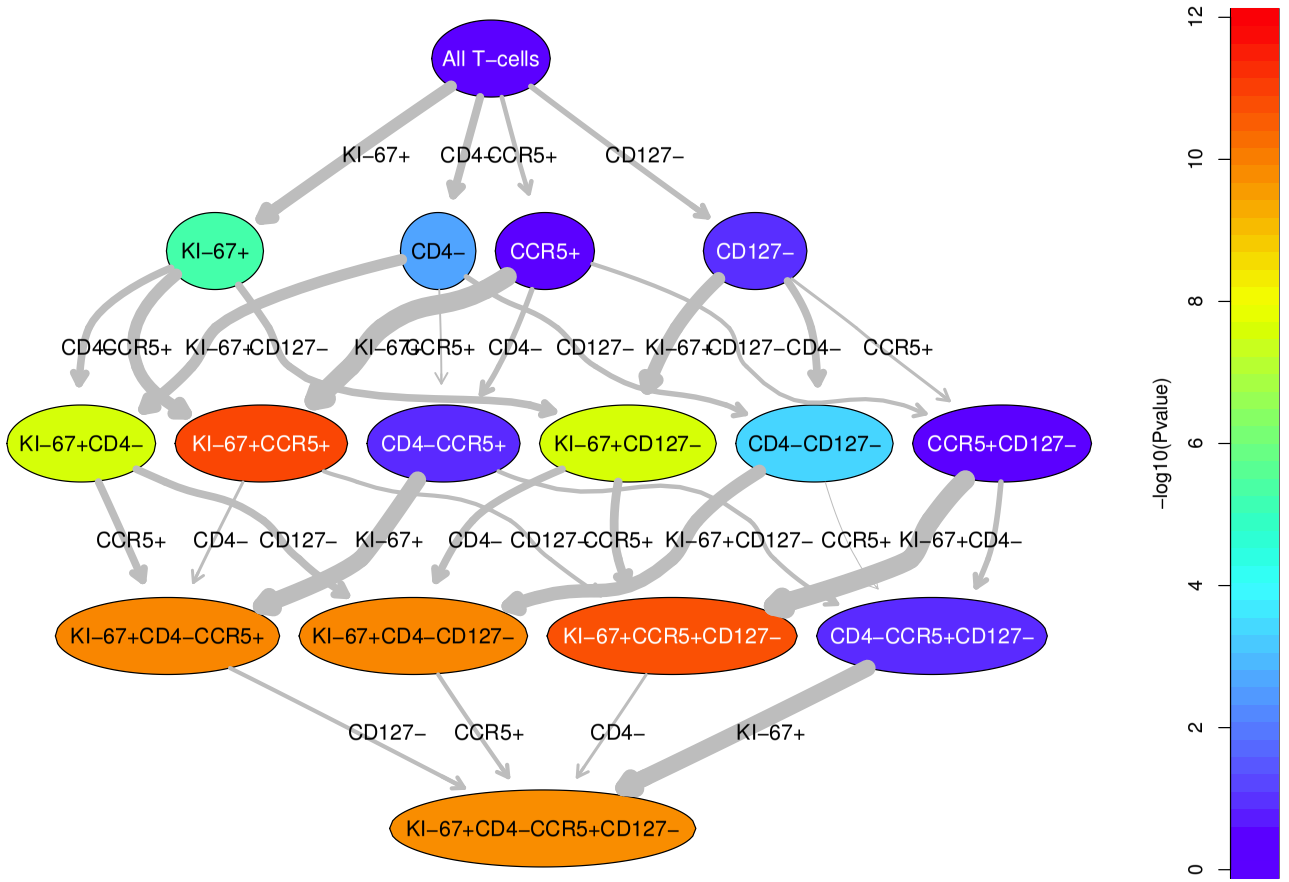


Figure 3.3: A complete cellular hierarchy for prediction of HIV's clinical outcome using $KI67^{+}CD4^{-}CCR5^{+}CD127^{-}$ T-cells. The color of the nodes indicates the significance of the correlation with clinical outcome (p-value of the logrank test for the Cox proportional hazards model) and the width of each edge (arrow) shows the amount of change in this variable between the respective nodes. The positive and negative correlation of each immunophenotype with outcome can be seen from the arrow type leading to the node; however, as all correlations are negative in this hierarchy, only one arrow type is shown.

The graph G_M has $|\mathcal{P}(M)| = 2^m$ nodes, one node for each parent phenotype of the phenotype of interest. The number of edges is equal to the number of markers (m), times the number of edges that have the specified marker. Each marker appears in 2^{m-1} nodes, therefore the number of edges is $m \times 2^{m-1}$.

A scoring function is needed to find the best hierarchy. This function should give a higher rank to hierarchies that go through more important parent populations earlier (*i.e.*, those that achieve a higher clinical significance with fewer markers). Because each node of the hierarchy is a phenotype, and each phenotype has a given score value $S(.)$, we use the *total score* function $T(.)$ - the sum of all negated phenotype scores in the hierarchy - as the scoring function:

$$\begin{aligned} T(\mathcal{H}) &= \sum_{(s,t)\in E_{\mathcal{H}}} W(s,t) \\ &= \sum_{(s,t)\in E_{\mathcal{H}}} -S(t) \\ &= \sum_{t\in V_{\mathcal{H}}\setminus v_0} -S(t) \end{aligned} \tag{3.3}$$

where $\mathcal{H}$ is the given hierarchy, $E_{\mathcal{H}}$ is the set of edges of hierarchy $\mathcal{H}$, $V_{\mathcal{H}}$ is the set of vertices of same hierarchy, and v_0 is the first node in the hierarchy. Applying this function to G_M, the best hierarchy is the minimum weighted path in G_M. We note that, in principle, more complex functions can be used to compute the total score of a given hierarchy; for example, in applications in which phenotypes with fewer markers are more important than the other phenotypes, an exponential function can be used to increase the weight of the earlier phenotypes in the hierarchy.

Dynamic Programming to Identify the Best Hierarchy

For cell populations characterized by m markers, finding the best hierarchy by searching through all possible hierarchies would require time $O(m!)$, which is impractical for even moderately large m. To make this problem tractable using dynamic programming, we define *best total score* function $T^*(.)$, which computes the score of the best hierarchy leading to the given phenotype. $T^*(.)$ is defined recursively as follows:

$$T^*(P^k) = \begin{cases} -S(P^k) & \text{if } k = 1 \\ \min\{T^*(P^k \setminus P_i^k) - S(P^k) | i = 1, \ldots, k\} & \text{otherwise} \end{cases}, \tag{3.4}$$

where P^k is a cell population defined by k single marker phenotypes, and $P^k \setminus P_i^k$ is P^k with the i^{th} single marker phenotype removed. For example, if $P^3 = KI\text{–}67^+CD28^-CD45RO^+$, then $P^3 \setminus P_1^3 = CD28^-CD45RO^+$. In other words, there is an edge from $P^k \setminus P_i^k$ to P^k in G_M where, P^k is a subset of M. Also note that $-S(P^k)$ is the weight of the edge $(P^k, P^k \setminus P_i^k)$ in G_M.

Using dynamic programming, we calculate the value of $T^*(.)$, iterating from level 0 to m on G_M. Calculating each node's score requires a number of constant

time operations equal to the number of edges entering the node. Therefore, the total number of operations is proportional to total number of edges ($m \times 2^{m-1}$), and the overall time complexity of our programming procedure for determining $T^*(.)$ values for all phenotypes in the graph is $O(m \times 2^{m-1})$. An illustration of the dynamic programming space for three dimensional space, *i.e.* having three markers, as well as two paths in that space is shown in Figure 3.4.

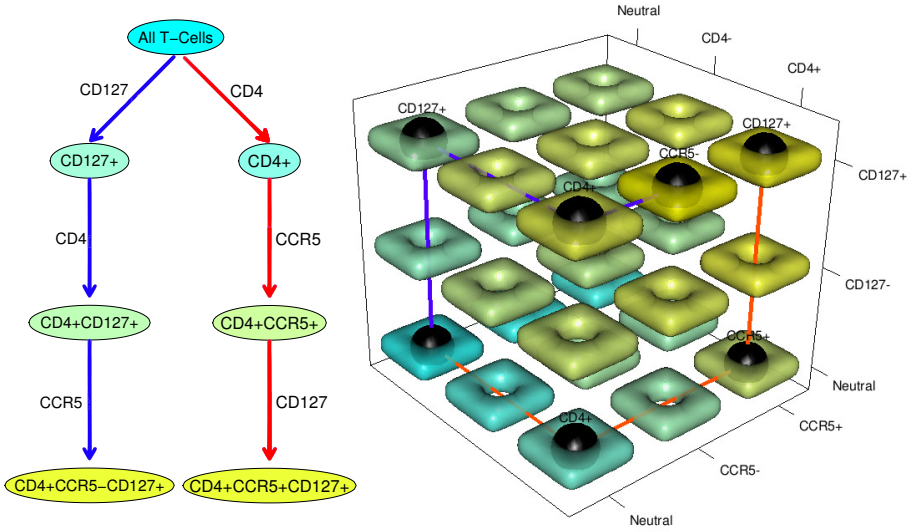


Figure 3.4: Dynamic programming algorithm for two cell populations defined by 3 markers. The best path for each of the cell population is shown in red and blue respectively. As an example, the red path ends at $CD4^+CCR5^+CD127^+$. Three markers are available to be added. First, CD4 is added (changes from does not matter to positive). Then two options will be available for the next step (CD127 and CCR5). After selection of CCR5, only one option will be left for the final step (CD127). Therefore for three markers, $\frac{3\cdot(3-1)}{2} = 6$ comparisons were required. **Left:** A hierarchy for the two paths. The label of an edge is the name of the single marker phenotype that is the difference between its head set (s) and its tail set (t). **Right:** the dynamic programming space for the 3 markers. Black spheres mark the nodes in the dynamic programing space used by the two paths. The colors of the nodes on the left match that of the square tori on the right and correspond to the relative score of each cell population.

Search for Near-Optimal Hierarchies

The hierarchy selected by the dynamic programming algorithm is the best gating strategy for a given cell population. However, we would also like to identify

alternate gating strategies with slightly less desirable scores. To find these near-optimal paths, we reformulate the problem as identification of a desired number of minimum weight paths: In G_M, the minimum weight path from source to sink is the best hierarchy (identical to the one generated by dynamic programming). To generate additional, sub-optimal hierarchies, a list of the next minimum weight paths must also be generated. These paths can be identified using the method by Eppstein [43]. As noted in the original article, elaborating the details of this algorithm is complicated and requires substantial background in algorithm design, which is well beyond the scope of this work. Briefly, this method uses the minimum spanning tree of G_M and computes a heap structure for each node; it then merges the heaps in an efficient way to construct a 4-heap data structure. Using this 4-heap and a given arbitrary number l (the number of desired paths), it generates l-minimum weight paths in time $O(e+v+l)$ for a DAG with e edges and v nodes (see Theorem 4 of [43] for details).

Hence, the time complexity of our algorithm can be calculated based on the number of edges and nodes using the time complexity of the l-minimum weight paths method:

$$\begin{aligned} O(e+v+l) &= O(m \times 2^{m-1} + 2^m + l) \\ &= O(m \times 2^{m-1} + 2 \times 2^{m-1} + l) \\ &= O((m+2) \times 2^{m-1} + l). \end{aligned} \tag{3.5}$$

For example, the number of operations with our approach on a dataset with $m = 10$ markers would be $\approx 10^4$ compared to $\approx 3 \times 10^6$ for the exhaustive search approach. Our method therefore takes ≈ 0.23 CPU seconds vs ≈ 69 CPU seconds for exhaustive search, run under 64 bit Linux (version 3.3) on $2.93GHz$ Intel Xeon CPU with sufficient memory (proportional to 2^M). For a phenotype involving $m = 20$ markers, these numbers increase to ≈ 1.2 CPU seconds vs $\approx 10^{11}$ CPU seconds (more than 4000 years), respectively. Even for a phenotype involving $m = 30$ markers measured by a CyTOF assay (mass spectrometry-flow cytometry hybrid device [102, 13, 27]), RchyOptimyx remains feasible, with a runtime of ≈ 102 CPU seconds, while the brute-force method would take $\approx 10^{22}$ CPU seconds. The final output of RchyOptimyx is the corresponding subgraph of G_M that includes all calculated paths (*i.e.*, the optimized hierarchy, *e.g.*, Fig. 3.5).

Datasets

We validated RchyOptimyx on two high-dimensional datasets, produced by mass and polychromatic flow cytometry.

Mass cytometry analysis of bone marrow cells from normal donors In this dataset, 31 parameters were measured for mononuclear cells from a healthy human bone marrow (see [13] for details). We used the results of three assays on samples subject to *ex vivo* stimulation by IL7 (measured by pSTAT5), BCR (measured by pBLNK), and LPS (measured by p-p38) as well as an unstimulated control. 13 surface markers were included in the analysis: CD3, CD45, CD45RA,

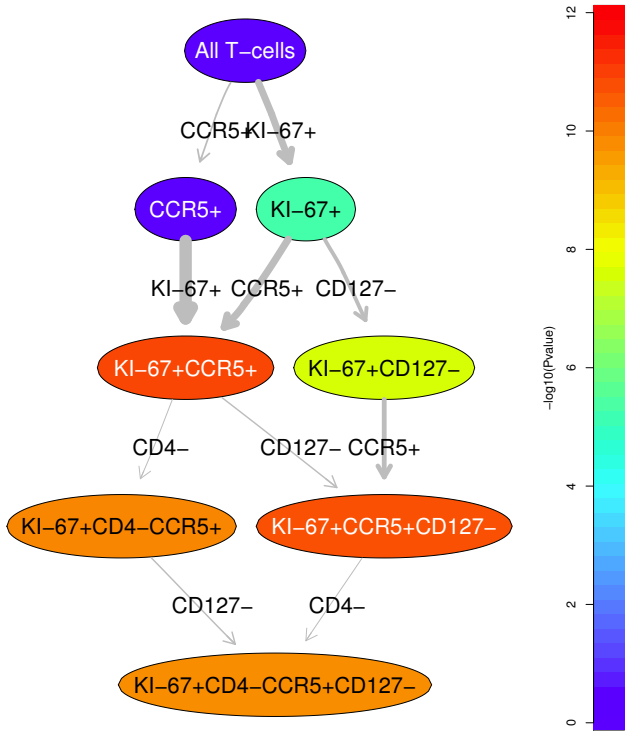


Figure 3.5: An optimized cellular hierarchy for prediction of HIV's clinical outcome using $KI67^{+}CD4^{-}CCR5^{+}CD127^{-}$ T-cells. The color of the nodes shows the significance of the correlation with clinical outcome (p-value of the logrank test for the Cox proportional hazards model) and the width of each edge (arrow) shows the amount of change in this variable between the respective nodes.

CD19, CD11b, CD4, CD8, CD20, CD34, CD33, CD123, CD38, and CD90. Singlets were gated manually, as described in the original publication.

Polychromatic flow cytometry analysis of HIV^{+} patients This dataset consists of 13 color PFC assays of 466 HIV^{+} subjects enrolled in the Infectious Disease Clinical Research Program's HIV Natural History Study. Basic demographic characteristics of this dataset are described elsewhere [153]. Cryopreserved peripheral blood mononuclear cells stored within 18 months of the date of seroconversion were analyzed using PFC as described by Ganesan *et al.* [54]. The cohort included 135 death/AIDS events, as defined by 1993 guidelines [23]. The date of the last follow-up or initiation of highly active anti-retroviral therapy (HAART) was considered a censoring event. CD14 and V-amine dye were used to exclude monocytes and dead cells, respectively, CD3 was used to gate T-cells. Using the staining panel and flowType, we enumerated various subsets of naive and

memory T-cells, defined by CD4, CD8, CD45RO, CD27, CD28, CD57, CCR5, CCR7, CD127, and KI-67. Using a log rank test with Bonferroni's multiple test correction, we scored each subset (cell population) in terms of its correlation with HIV progression [2].

Results

Designing a Panel to Detect a Population Expressing an Intracellular Marker using Surface Markers

In this use–case, our goal was to identify cell populations that are affected by different stimulations in the mass cytometry dataset. We used flowType to identify a list of populations that had a high overlap with either the $IL3^+$, BCR^+, or LPS^+ populations (determined manually - see Fig. 3.6). For each cell population, this value was calculated as the difference in its intersection with the $IL3^+$, BCR^+, or LPS^+ compartments between the stimulated and unstimulated sample. For example, for a given cell population CP, the overlap with $IL3^+$ was defined as:

$$Overlap^{IL3^+}(CP) = \left(\frac{\#\ \mathrm{IL3}^+ cells\ in\ CP}{\#\ cells\ in\ CP}\right)_{stim} - \left(\frac{\#\ \mathrm{IL3}^+ cells\ in\ CP}{\#\ cells\ in\ CP}\right)_{unstim} \tag{3.6}$$

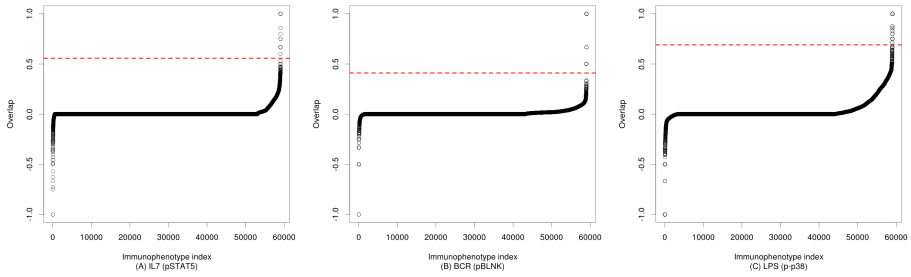
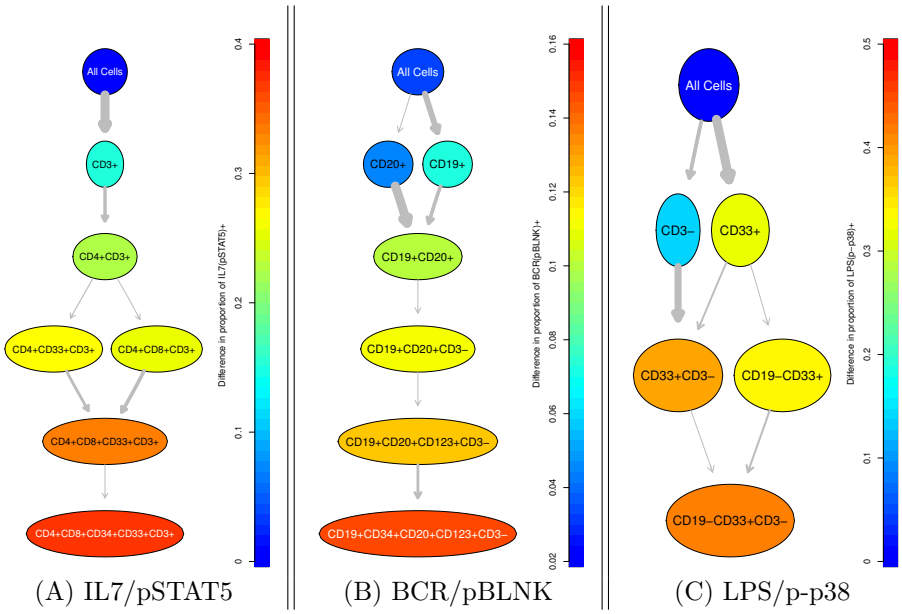


Figure 3.6: All immunophenotypes ordered by their overlap with the cell population of interest. The red dashed lines indicate the cutoffs used for selecting the immunophenotypes with "high overlap".

The immunophenotypes with a high overlap, as identified by flowType, are listed in Tables A.1, A.2, and A.3. These immunophenotypes were analyzed using RchyOptimyx (*e.g.*, Fig. 3.7 for BCR) and then merged into a single graph, shown in Fig. 3.7. This graph suggests that T-cells ($CD3^+$) followed by cytotoxic T-cells ($CD3^+CD4^+$) are the main parent populations that are affected by IL7 stimulation (panel A). As expected, BCR stimulation affected B-cells ($CD19^+CD20^+CD3^-$), and LPS stimulation increased the proportion of $CD19^-CD33^+CD3^-$ cells (Panels B and C, respectively). These results are

generally consistent with those reported in the original study (Figure 2 and panel C of Figure 3 of [13]).



(A) IL7/pSTAT5 (B) BCR/pBLNK (C) LPS/p-p38

Figure 3.7: Three optimized hierarchies for identification of cell populations with maximum response to IL7, BCR, and LPS measured by pSTAT5, pBLNK, and p-p38, respectively. The colour of the nodes and the thickness of the edges shows the proportion and change in proportion of cells expressing the intracellular marker of interest, respectively.

Simplifying Gating Strategies

Here we use RchyOptimyx to demonstrate an example of the use case of establishing a simpler combination of markers that can be used to identify a target population at a desired level of purity. For analysis of the PFC dataset, Ganesan *et al.* used a strict, but potentially redundant definition for naive T-cells, of $CD28^+CD45RO^-CD57^-CCR5^-CD27^+CCR7^+$, within the $CD3^+CD14^-$ compartment [54]. The purity of a given parent cell population (CP) of this target was defined as its mean purity for the strictly-defined naive T-cells:

$$Purity(CP) = \frac{\sum_{All\ Samples} \frac{\#\ CD28^+CD45RO^-CD57^-CCR5^-CD27^+CCR7^+\ cells}{\#\ cells\ in\ CP}}{\#\ Samples} \tag{3.7}$$

Figure 3.8 shows the results of analysis with RchyOptimyx where a combination of only three markers ($CD45RO^-CCR5^-CCR7^+$) identified the strict naive T cell population to 95% purity (within the $CD3^+CD14^-$ compartment). The

range of available purities, and determination of an appropriate cutoff is experiment dependent (*e.g.*, on the range of available markers or biological question being researched) and this result is only provided as an example of the utility.

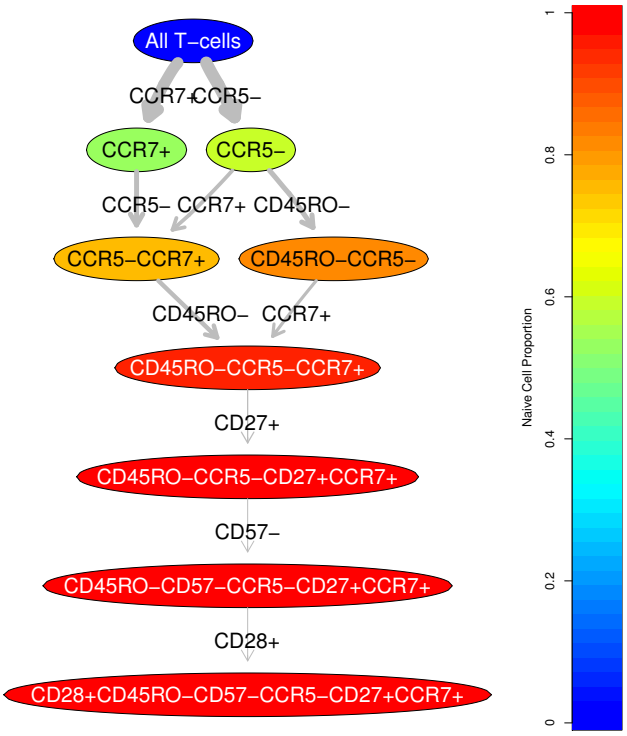


Figure 3.8: An optimized cellular hierarchy for identifying naive T-cells. The color of the nodes and the thickness of the edges shows the purity and change in purity of the original naive phenotype within the given cell population, respectively.

Characterization of a Large Number of Immunophenotypes

Here we use RchyOptimyx to demonstrate an example of the use-case of summarizing a large list of immunophenotypes of interest (as identified by a bioinformatics pipeline) into a single hierarchy using their most important common parent populations.

In a previous study of the PFC dataset, we identified 101 immunophenotypes (Table A.4) in HIV$^+$ patients that had a statistically significant correlation with HIV's progression [2]. The score of each population was calculated as $-log_{10}(p)$ where p was the p-value of the logrank test before adjustment for multiple testing (higher values represent a stronger correlation with the clinical outcome). The 101 immunophenotypes were analyzed using RchyOptimyx and

the resulting hierarchies were merged into a single graph (Figure 3.9). This graph indicated three groups of immunophenotypes that were significantly correlated with HIV's outcome (left, center, and right branches). The left branch consisted of KI-$67^{+}$$CD4^{-}$$CCR5^{+}$$CD127^{-}$ T-cells. These cells were thought to be statistical significant mainly because they are long-lived ($CD127^{-}$) T-cells with high proliferation (KI-67^{+}). RchyOptimyx showed that the significance of this population is related to the KI-$67^{+}$$CCR5^{+}$ compartment and not $CD127^{-}$ (Figure 3.9, the left branch) as the CD127 marker is not needed to achieve the approximately the same score. This is in agreement with the results of two recent studies [58, 69]. The terminal node of the center branch consisted of seven markers ($CD45RO^{-}$$CD8^{+}$$CD57^{+}$$CCR5^{-}$$CD27^{+}$$CCR7^{-}$$CD127^{-}$). RchyOptimyx revealed that its most important parent population is $CD8^{+}$$CCR7^{-}$$CD127^{-}$, with a weaker correlation with the clinical outcome. Finally, the right branch ($CD28^{-}$$CD45RO^{+}$$CD4^{-}$$CD57^{-}$ $CD27^{-}$$CD127^{-}$) suggests several parent populations with minimal overlap and strong correlation with the clinical outcome (*e.g.,* $CD28^{-}$$CD4^{-}$$CD57^{-}$$CD127^{-}$ and $CD45RO^{+}$ $CD4^{-}$$CD127^{-}$).

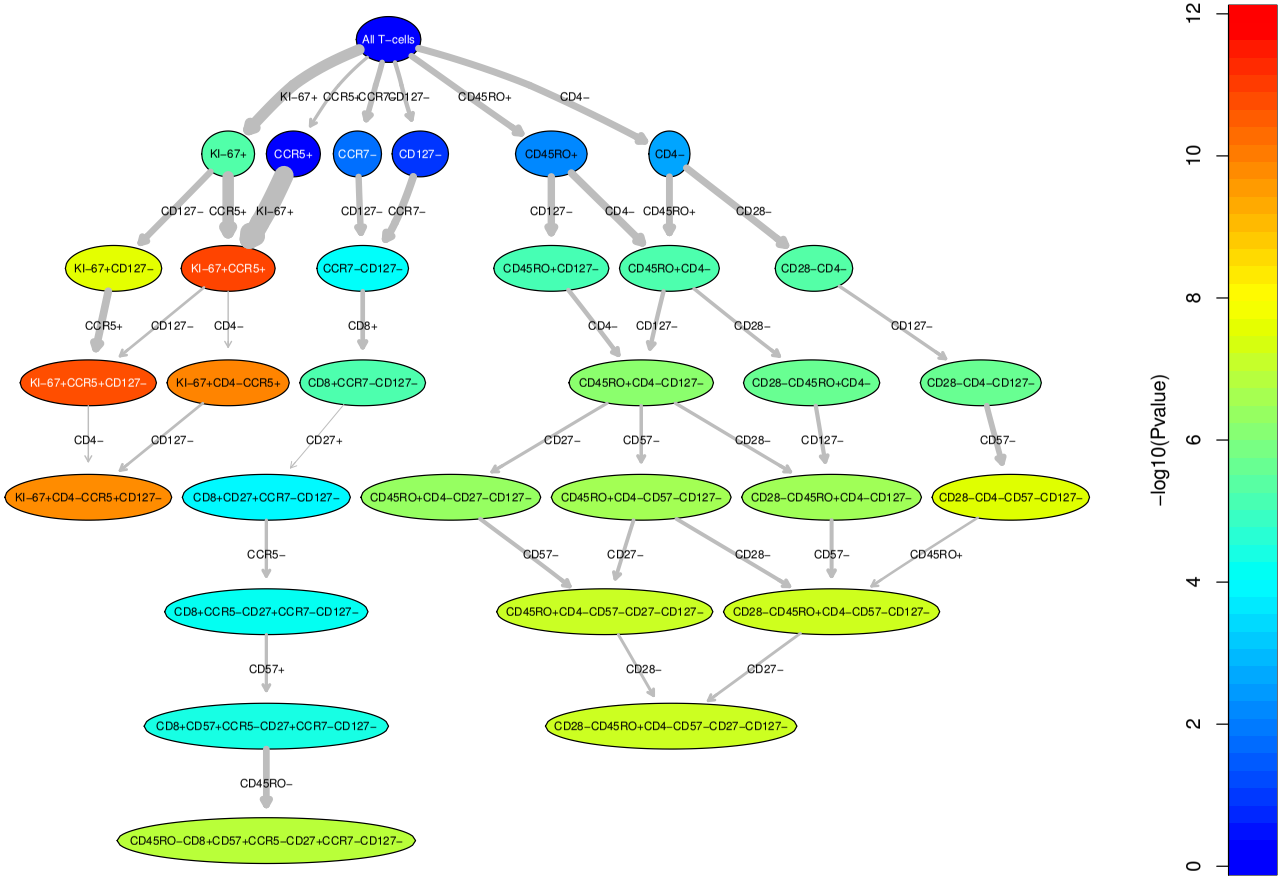


Figure 3.9: An optimized hierarchy for all three populations correlated with protection against HIV. The color of the nodes shows the significance of the correlation with the clinical outcome (p-value of the logrank test for the Cox proportional hazards model) and the width of each edge (arrow) shows the amount of change in this variable between the respective nodes. The positive and negative correlation of each immunophenotype with outcome can be seen from the arrow type leading to the node; however as all correlations are negative in this hierarchy, only one arrow type is shown.

Discussion

Sequential analysis of the markers involved in manual or automated identification of cell populations is fundamental to our understanding of the characteristics of the cell population. In sequential gating, the order in which the gates have been applied does not affect the final results. However, ordering the gates by their relative importance has two use-cases: 1) identifying a cell population of interest, using the smallest possible panel of markers; 2) summarizing a long list of closely related (and perhaps overlapping) immunophenotypes by identifying their most important common parent populations. However, increasing the number of markers quickly renders this approach infeasible (*e.g.*, Fig. 3.10 for only six markers).

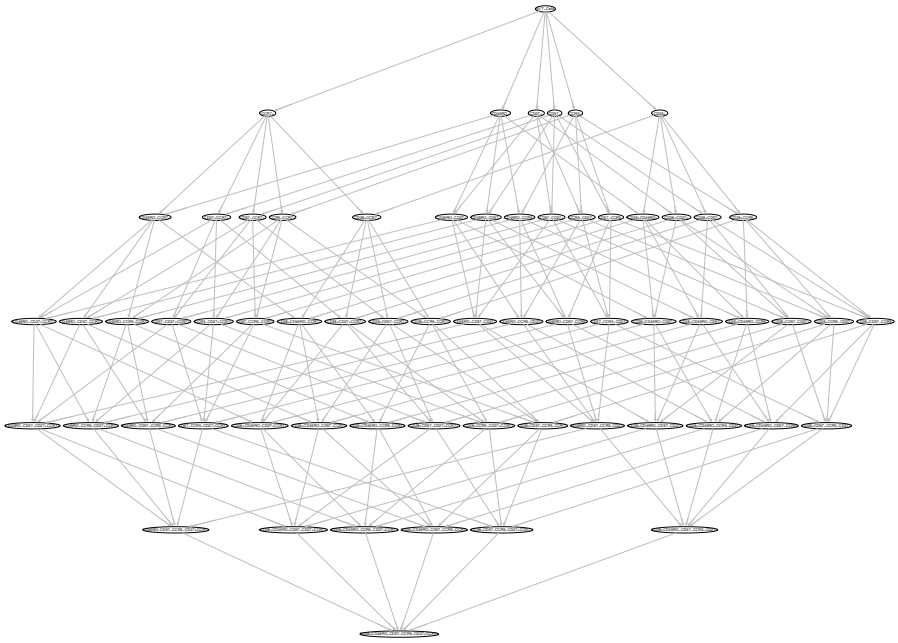


Figure 3.10: A complete cellular hierarchy for identifying naive T-cells. The colour of the nodes and the thickness of the edges have been removed to facilitate visualization of the complex graph.

To address this challenge, we developed RchyOptimyx, a computational tool that automatically characterizes the complex findings of high-dimensional exploratory FCM studies. RchyOptimyx sorts all parent populations of an immunophenotype of interest into hierarchies, and selects those hierarchies that

are better able to maintain the characteristics of the immunophenotype of interest (*e.g.,* correlation with a clinical outcome). This reveals the best order in which markers can be excluded from an immunophenotype. RchyOptimyx uses dynamic programming and efficient tools from graph theory to make the problem tractable using the computing resources readily available in most laboratories.

Since most cells can be described using more than one combination of markers, there usually are several alternative cellular hierarchies associated with every population. RchyOptimyx finds all these "paths" and merges them into a single hierarchy, starting from "all cells", or any arbitrary point in a hierarchy, and finishing at the terminal population of interest. This reveals the relationships between different gating strategies and how they differ, and also facilitates the reproduction of high–dimensional exploratory studies using low–color instruments. The ability to suggest multiple panels is particularly important when designing new panels, because the choice of markers depends on a large number of external parameters including, but not limited to, reagents available through vendors, potential spectral overlaps, the instruments available, and budget limitations.

Another important use-case for RchyOptimyx is in the interpretation of the findings of bioinformatics pipelines. While these pipelines have recently been very successful in identifying cell populations correlated with clinical outcomes, their findings cannot be easily understood for two reasons: 1) they usually rely on high-dimensional clustering of the data and therefore cannot propose gating strategies for reproduction of their results; 2) their predictive power often relies on a large list of immunophenotypes. Some of these immunophenotypes are closely related (*e.g.*, refer to close or overlapping cell populations) while others are not. RchyOptimyx addresses the first problem by suggesting optimized gating hierarchies for identification of these cell populations to a desired level of purity or correlation with clinical outcome. The latter problem is addressed by summarizing closely related immunophenotypes using their most important common parents.

In evaluating RchyOptimyx, we combined its functionality with the automated gating functionality provided by flowMeans and flowType. However, RchyOptimyx can be built upon the results of any cell population identification method, including manual analysis, provided all intermediate cell populations (*i.e.,* each layer, removing one marker at a time) from the cell population of interest up to the desired start of the hierarchy are provided to the method.

We evaluated RchyOptimyx for three use-cases, using a small but high-dimensional mass cytometry dataset and a clinical dataset of high-dimensional conventional FCM assays of 466 patients, previously analyzed by both manual and automated analysis. First, we constructed cellular hierarchies for identification of cells that were produced in response to different stimulations. This use-case represents the problem of designing panels of surface markers (primarily for sorting) for cells that can only be defined using their intra-cellular signature (possibly after proper stimulation). For example, plasmacytoid dendritic cell (PDC)s are known to express the toll-like receptor 9 (TLR9) in response to stimulation using CpG [76]. A large number of surface candidates were recently proposed for PDCs [87, 140, 128, 22]. An interesting direction to extend this

work would be to measure all these markers in a single panel, subject to CpG stimulation (using appropriate controls) to design a panel of surface markers for PDCs. In this case, TLR9 could be used as the external variable for optimization.

Second, we demonstrated that RchyOptimyx can be used to simplify existing gating strategies, using as an example the identification of naive T-cells previously defined using a complex panel of six markers to a 95% purity using only three. This proof-of-concept use-case is relevant when a subset of markers needs to be selected for reproduction of the results using fewer colors. For certain biological use–cases, purity of higher than 95% can be required. For such use–cases, a larger number of markers for exclusion of non-naive T-cells should be included in the panel.

Third, we showed that RchyOptimyx, together with a complex bioinformatics pipeline, can analyze a large high-dimensional clinical dataset, to reveal correlates of a clinical outcome, hidden from previous manual and automated analysis of the same dataset. In addition, RchyOptimyx suggests the best gating strategies and marker panels for reproduction of these results in low-color settings. By identifying the best cellular hierarchies, RchyOptimyx allows the user to make an informed decision about the trade-off between the number of markers and the significance of the correlation with the clinical outcome. This feature is particularly important in hypothesis generating studies that need to be further validated using large clinical studies.

For the third example, it is important to note that the correct measure for the amount of correlation with a clinical outcome is an effect size (such as the root squared error of the estimated proportional hazard). However, such effect size does not provide any information about the significance of the correlation. As RchyOptimyx is intended to be a decision support tool, and in this case the decision is the degree to which a cell population can be generalized while maintaining the statistical significance of the correlation, we decided that the p-values of the log-rank tests were more appropriate for optimization of the hierarchies. To support this decision, we empirically investigated the differences between the p-values and effect sizes of the Cox proportional hazard models (Fig. 3.11) and concluded that these values are highly correlated (which is not surprising considering the large size of our cohort). It should be noted that as RchyOptimyx allows the user to choose which measure to provide, they can make this decision as appropriate for their specific data.

The concept of computationally extracting cellular hierarchies from FCM data has previously been introduced by the SPADE algorithm [13, 115]. SPADE generates a large number of multidimensional clusters and then connects them to each other using the distance between their mean/median fluorescence intensities. These are then manually annotated by biologists with domain knowledge. This makes SPADE useful for identification and visualization of a large number of clusters, particularly when expression of markers change gradually (*e.g.,* cell-cycle analysis and some intracellular studies). However, the hierarchies generated by SPADE are logically and conceptually different from those generated by RchyOptimyx and have different use-cases. For example, the results of the mass cytometry dataset presented here are very close to results previously obtained

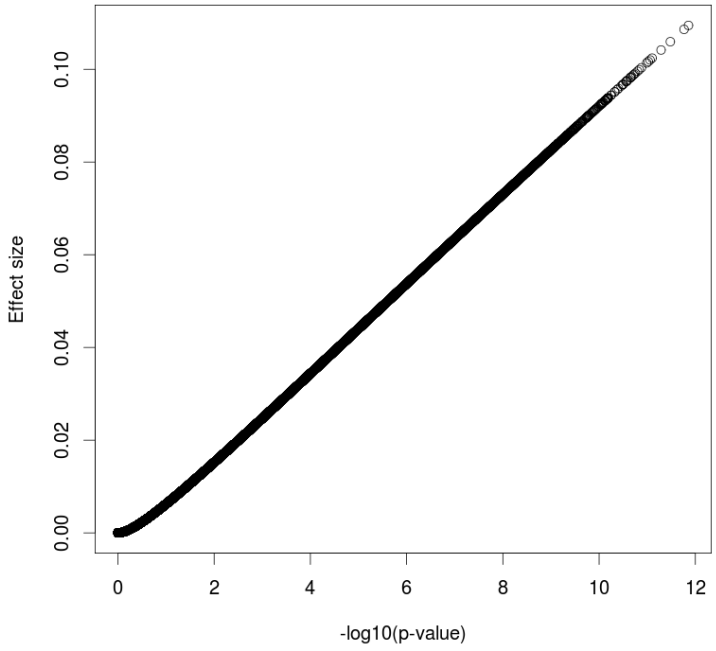


Figure 3.11: The correlation between the effect sizes and p-values of the log rank tests for the Cox proportional hazards models for each immunophenotype. The Pearson correlation coefficient was determined as 0.997, indicating a highly significant correlation with a p-value $< 2.2 \times 10^{-16}$.

from SPADE analysis. However, SPADE required manual annotation of the results by a human expert, using different plots demonstrating the expression of different surface markers and the intra–cellular marker of interest (Figure 2 and panel C of Figure 3 of [13]). More complicated relationships that involve several markers cannot be easily identified by these manual annotations. In addition, SPADE is limited in that the relationships between cell populations is exclusively defined using the multidimensional distances between them. However, two cell populations that are close to each other in the multidimensional space can be far in terms of specific markers (which can be the most important ones). The cellular hierarchies generated by RchyOptimyx are based on parent-child relationships, guided by an external variable (cell populations that have common parents with similar patterns of correlation with a clinical outcome or intracellular response to stimulation are grouped together). This enables RchyOptimyx to automatically annotate a large number of cell populations identified by other methods (*e.g.*,

manual gating or SPADE) in terms of the importance of the markers involved and summarize them in a single hierarchy.

There are several directions in which this work can be extended. RchyOptimyx provides no information about the robustness of the hierarchies. Bootstrapping strategies could be used to produce confidence intervals for the tree structure and increase generalizability to previously unseen data [138]. Also, our current implementation of RchyOptimyx assumes that every marker can be partitioned into a positive and negative population. While the underlying theory does support additional (*e.g.*, dim, bright, or low) populations, parts of the software package would need to be modified to accommodate these cases.

Availability

The RchyOptimyx R package (including source code, documentation, and examples) is freely available under an open source license (*Artistic 2.0*) and can be obtained from Bioconductor. The raw data and meta-data used in this study is publicly available through FlowRepository.org (under experiment ID *FR-FCM-ZZZK*) and through Cytobank.org (under experiment ID *6033*) for the PFC and CyTOF datasets, respectively.

3.2.3 flowType/RchyOptimyx pipeline

Flow cytometry has undergone a "chromatic explosion" over the past decade and can now measure 17 markers at once for each of hundreds of thousands of individual cells [29]. Since then, mass cytometry has enabled measurement of 30–45 markers per cell [14], while single-cell multiplexed RT-qPCR can measure 50–96 mRNAs per cell [156]. The growth in high-throughput single-cell data continues to outpace development of corresponding bioinformatics techniques [29]. To answer this challenge, we previously developed flowType [2] and RchyOptimyx [4]. flowType uses partitioning of cells, either manually or by clustering, into positive or negative for each marker to enumerate all cell types in a sample, e.g. [3]. RchyOptimyx measures the importance of these cell types by correlating their abundance to external outcomes, such as disease state or patient survival, and distills the identified phenotypes to their simplest possible form. These packages have been used to identify several novel cell populations correlated with HIV outcome [2]. More recently, this pipeline has been used to evaluate standardised immunological panels [150], to optimise lymphoma diagnosis [35], and to analyse a range of other clinical data (unpublished).

However, the higher dimensionality of data produced by mass cytometry generates up to $3^{45} \approx 10^{21}$ possible cell types, with an even greater number (up to $3^{96} \approx 10^{45}$) for single-cell qPCR; these magnitudes are beyond the capabilities of flowType and RchyOptimyx. Furthermore, flowType and RchyOptimyx have thus far only treated cells as being either positive or negative for a marker. In practice, many biomarkers can have a range of expression levels such as "dim" and "bright". In this application note, we detail architectural improvements to flowType and RchyOptimyx to overcome these limitations.

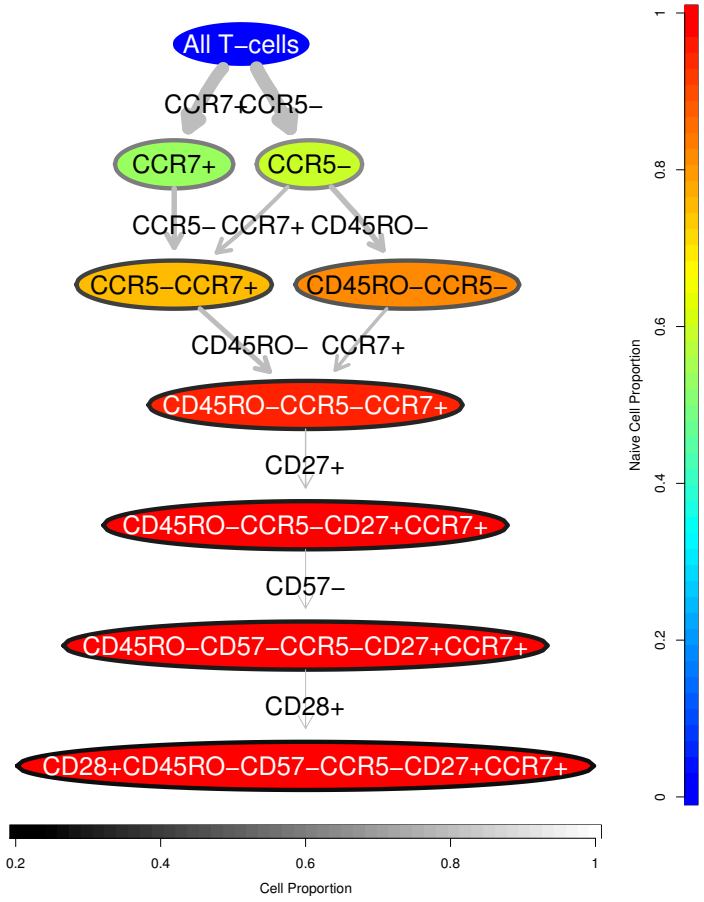


Figure 3.12: An optimized cellular hierarchy for identifying naive T-cells. The colour of the nodes and the thickness of the edges shows the purity and change in purity of the original naive phenotype within the given cell population, respectively. This is similar to Figure 6 in the main text except the color of the border of the nodes shows the cell proportion of the cell population.

Approach

Our primary challenge was to enable flowType to generate a number of cell types tractable on most common workstations (e.g. those with 4–12GB of RAM). We hereafter denote the original flowType implementation as flowType-BF (brute force), and the new version as flowType-DP (dynamic programming). Whereas flowType-BF completely enumerates all cell types over all $[1, ..., m]$ markers, we opted in flowType-DP to use a breadth-first strategy of enumerating all cell types defined over a subset of $k \leq m$ markers. We provide a memory use

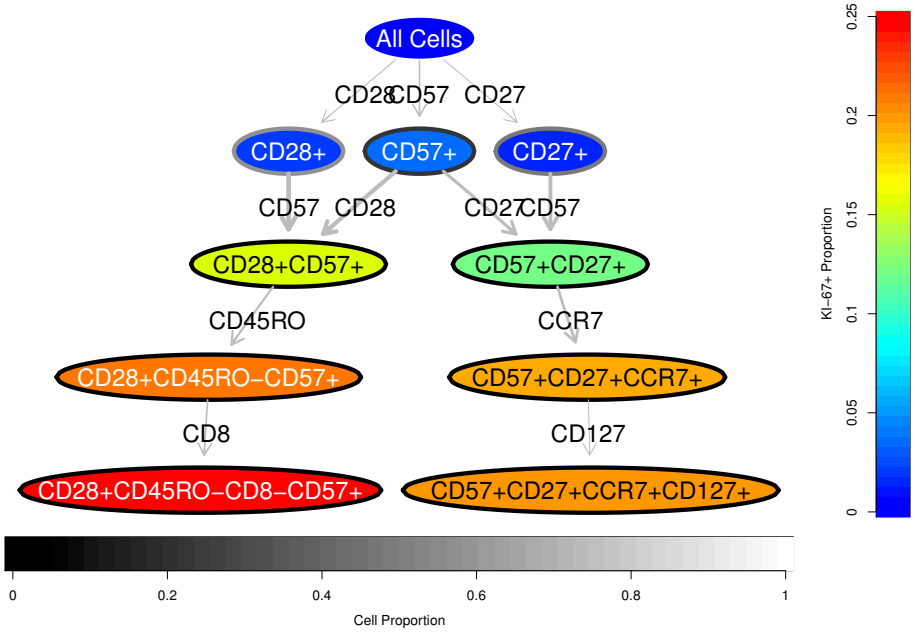


Figure 3.13: A cellular hierarchy for identifying KI-67$^+$ T-cells using surface markers. The colour of the nodes and the thickness of the edges shows the proportion and change in proportion of KI-67$^+$ T-cells, respectively. This is similar to Figure 7 in the main text except the color of the border of the nodes shows the cell proportion of the cell population.

estimation function, to assist users in finding a k that fits within the limits of their hardware. To improve computation time, in flowType-DP we implemented a dynamic programming approach, which exploits the fact that cell types can be arranged into a hierarchy, and membership of any given cell type over n markers is equal to the intersection of one of its parent types (over $n-1$ markers) with a single-marker cell type. flowType-DP first enumerates all cell types involving only 1 marker by simple partitioning and then iterates over 2, ..., k markers, computing all cell types for each level n by set intersections between corresponding cell types in levels $n-1$ and 1.

For example, membership of the cell type CD45^{++}CD117$^+$CD34$^-$ is computed as follows:

$$\begin{aligned} &\{\text{CD45}^{++}\text{CD117}^{+}\text{CD34}^{-}\} \\ =&\{\text{CD45}^{++}\text{CD117}^{+}\} \cap \{\text{CD34}^{-}\} \\ =&\{\text{CD45}^{++}\} \cap \{\text{CD117}^{+}\} \cap \{\text{CD34}^{-}\} \end{aligned} \tag{3.8}$$

To allow partitioning into levels other than positive and negative, we used

a string representation for cell types. The string has one integer character for every marker, denoting the partition, or zero if the marker is not used. Values $1, ..., n$ denote partitions 1 to n. For example, if the set of markers were {CD3, CD45, CD13, CD117, CD34} the cell type $CD45^{++}CD117^{+}CD34^{-}$ would be represented by `03021`. RchyOptimyx uses a dynamic programing algorithm for efficiently constructing k-shortest paths [43]. We modified RchyOptimyx' graph construction component to be able to handle more than one partition per marker.

Results and Discussion

We evaluated flowType-DP against flowType-BF on a 10-marker dataset available from Flow Repository (ID FR-FCM-ZZZK) [2]. flowType-DP showed a substantial speedup over flowType-BF, which increases exponentially with the number of cells and markers. For example, at 10^6 cells and 10 markers, flowType-DP is 14 times faster (see Fig. 3.14a and b). Comparison on larger datasets was not possible, due to the limitations of flowType-BF.

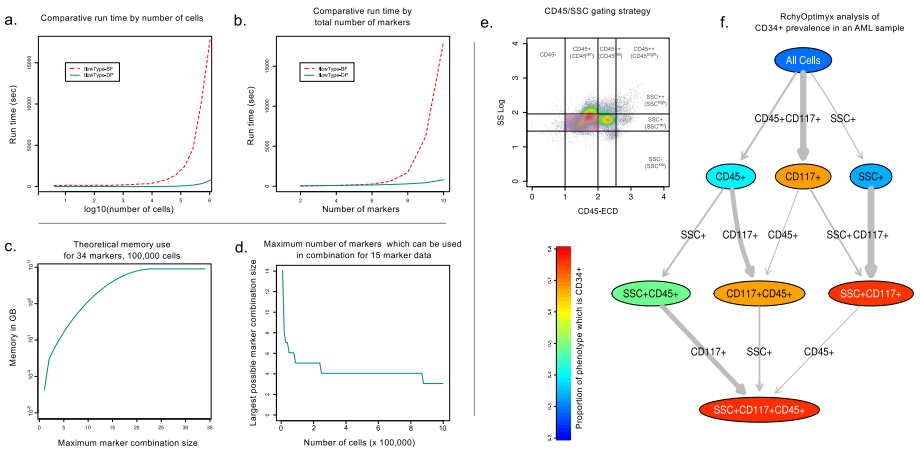


Figure 3.14: **a-b. Run time comparison of flowType-DP to flowType-BF** in terms of number of cells (a) and number of markers (b). **c-d. Possible thresholds for marker combinations using flowType-DP** for typical mass cytometry data (c) and polychromatic flow cytometry data (d). **e-f. Three/four-partition flowType-generated, RchyOptimyx-visualized cell type hierarchy** on a bone marrow sample from a patient with AML. Cell population identification strategy used for SSC and CD45, with the CD34-enriched subset highlighted (e). RchyOptimyx analysis showing CD34 enrichment (f).

We also computed the limits for k on a hypothetical machine with 12GB of RAM for samples representative of mass cytometry (Fig. 3.14c) and polychromatic flow cytometry (Fig. 3.14d), both of which would be intractable for flowType-BF. flowType and RchyOptimyx are now able, within the memory of a common workstation (12GB), to analyze 34-marker data.

Finally, to demonstrate the importance of several partitions per marker, we applied flowType and RchyOptimyx to an acute myeloid leukemia sample from Flow Repository (ID FR-FCM-ZZYA) (Fig. 3.14e-f). CD34 is a stem-cell marker typically expressed on AML blast cells. These blasts are also known to have dimly positive CD45 expression and low SSC [149]. By partitioning CD45 and SSC into four and three partitions, and naively running flowType and RchyOptimyx to search for CD34-enriched cell types, we were able to find that the $SSC^{low}CD45^{dim}$ cell type had a high proportion of $CD34^{+}$ cells, as expected. This would not have been possible with only two partitions for each of CD45 and SSC.

3.3 Lymphoma Diagnosis Quality Checking

Pathologists follow certain processes to diagnose cancer cases which can be imperfect and subjective. Also, since cancer is a heterogeneous disease and the cells from different parts of the same tumor might show different characteristics, the diagnosis may be inaccurate or unrepresentative of the whole tumor. As a result, pathologists do not always agree on the diagnosis, and in some cases a consortium of doctors is required to get a more reliable diagnosis. In order to improve this process, hospitals go through routine quality checking on the past diagnoses looking for potential mistakes in the hindsight. However, due to limited amount or resources, this process can be done only for a small fraction of all the patients. Here we develop a pipeline to assert candidate cases to be checked in this QC process. We also interpret our models to propose potential reasons for these cases being tagged.

3.3.1 Introduction

We have designed and developed two methods flowType and RchyOptimyx in this chapter. The flowType method extracts cell populations as features from flow cytometry data, and then, a method such as ROC-AUC scores those features. Finally RchyOptimyx summarizes and visualizes important cell populations of the data.

Here our goal is to extend the method and create a framework for diagnosis quality checking purposes, *i.e.* to report samples that might be misdiagnosed or require further investigation. For this, we first classify the given samples into two groups according to their diagnosed label; then for each sample, we train our model using all samples except the selected one, and check how much our model agrees with the given diagnosis label. This scheme is also known as leave one out cross validation (Section 2.1.2). At the end, taking into account the confidence of our model for each prediction, we report the difference between the given labels and our predicted labels, and samples with the most difference between their predicted label and given label are candidates for further investigation by the pathologist/oncologist.

3.3.2 Materials and Methods

Flow Cytometry Data

Our data includes samples from Diffuse large B-cell lymphoma (DLBCL) and Follicular Lymphoma (FL) patients. Table 3.2 shows a summary of the cohort.

Diagnosis Label	Count
FL	49
DLBCL	22
Sum	71

Table 3.2: Number of samples belonging to each diagnosis label

The samples the cohort were analyzed using three sets of 8 markers in three different tubes using an 8-color capable flow cytometry machine. We include forward and side scatter values to the analysis, and treat them as biomarkers. Therefore each tube gives us a 10 dimensional vector for each patient. Table 3.3 presents markers on each tube.

Tube	Markers
Tube 1	FS, SS, polyKappa, polyLambda, CD5, CD10, CD11c, CD20, CD3, CD19
Tube 2	FS, SS, FMC-7, CD103, CD5, CD38, CD23, CD25, CD19, CD3
Tube 3	FS, SS, CD57, CD7, CD5, CD2, CD56, CD8, CD3, CD4

Table 3.3: Combination of markers in three tubes. Markers essential to differentiate main cell types (*i.e.* CD3 and CD 5) are present in all tubes.

Cell Population Identification and Preprocessing

In this phase, we first identify cell populations on a single marker level using k-means clustering method. Then we use flowType to identify cell populations using all marker combinations, and after filtering out very rare cell populations, we construct a matrix in which each row corresponds to a sample and each column corresponds to a cell population. At the end we normalize each column of the constructed data matrix.

We assume each marker divides cells into two populations, positive and negative. For each marker, a k-means clustering method with $k = 2$ automatically clusters cells into two groups. The group with a higher value of the corresponding marker is our positive, and the other group is our negative population with regard to the marker. A threshold right in the middle of the two groups with the same euclidean distance from the two cluster centers, determines the two corresponding populations. For example, if $CD4$ is the marker, then $CD4^-$ and $CD4^+$ represent cells bellow and above the threshold respectively.

Then flowType counts the number of cells for each possible population considering determined thresholds. As the next step, we normalize these cell counts dividing them by total cell count to derive proportion of cells in each area. Having M markers, we detect 3^M cell populations (section 3.2.1), and put them in a matrix, having one row per sample and 3^M columns corresponding to the cell populations. At the end we normalize each column of the resulting matrix by estimating each column's mean (μ_i) and standard deviation (σ_i), and then transform its values (x) according to Formula 3.9:

$$x^t := \frac{x - \mu_i}{\sigma_i} \tag{3.9}$$

For the purpose of diagnosis, we are interested mostly in cell populations that on average have more than only a few cells per sample. Therefore we discard cell populations which have a median value less than 0.05% of total cell count. We performed this preprocessing for each tube leading to three matrices.

Sample Classification

We compare classification performance of several methods, which include support vector machines in three variants (section 2.1.4), gradient boosting classifier [52], and a method which performed well in a previous benchmark competition (FlowCAP), hereafter referred to as *team21*[5] [3].

In order to assess the performance of the above methods, we perform a leave one out cross validation on the data. This strategy leaves one sample out at a time, trains the model on the rest of the data, and then records the output of the trained model on the sample which was left out. Having N samples, this process is repeated N times. These recorded values are used to measure the overall performance of methods and to find samples which are valued for further investigations, as explained later.

Except for method *team21*, we need to find the appropriate hyperparameters for the other models. These are the parameters which we have to define before starting the training process. For each method, we find the best set of hyperparameters using a 10-fold cross validation; hence a nested cross validation scheme. Table 3.4 lists the search space of these parameters for their corresponding method.

As shown in Table 3.5, Gradient Boosting Classifier, $l1$-SVM-linear, and $l2$-SVM-linear all have comparable performances. Among these three methods, we choose $l2$-SVM-linear for the next steps for two reasons: proper handling of imbalanced data, and interpretability of its coefficient vector as features' importance. We choose $l2$-SVM-linear over $l1$-SVM-linear because $l1$ penalized SVM tends to be very sparse, meaning that it chooses only a few features as predictors. This results in usually choosing features that correspond to fewer cells and ignoring more abondant cell populations.

[5] `http://www.ehu.eus/biologiacomputacional/team21_vilar`

Method	Parameter	Range
Gradient Boosting Classifier	max used features	$5, 10, 15$
	max tree depth	$1, 2, 3$
	estimator count	$5, 20, 50, 100, 200$
$l2$-SVM	C	$2^{-10:10}$
	kernels	rbf, linear
	gamma (rbf kernel only)	$2^{-10:10}$
$l1$-SVM linear kernel	C	$2^{-10:10}$
$l2$-SVM linear kernel	C	$2^{-10:10}$

Table 3.4: Hyperparameter variables of each method and their corresponding tested value range.

Method	ROC AUC
team21	0.5
Gradient Boosting Classifier	0.84
$l2$-SVM	0.76
$l1$-SVM linear kernel	0.84
$l2$-SVM linear kernel	0.83

Table 3.5: Method Performances - SVM: Support Vector Machine, $\{l1, l2\}$-SVM-linear: SVMs with a linear kernel which are penalized using an $l1$ or an $l2$ term respectively.

Now we compute the difference between the output of the $l2$-SVM-linear method, and the desired output, *i.e.* -1 and 1. Note that the models' outputs are real values, and the sign of the output determines the predicted class. Since the SVM's output shows the distance of the input from its decision boundaries, and the further away the input sample from the boundary, the more confident the model is in its decision, the output can be interpreted as the model's confidence. But there is a catch: these values are computed using different models, and different scaling factors in these models means that the calculated real values cannot be directly compared between models. To fix this issue, we normalize decision values using the mean and variance of the outputs of the models on their corresponding training data. Now we can use the real values not only to find samples that are simply misclassified, but also to sort them according to the models' confidence. As a result, a good candidate sample to be investigated is one which is misclassified and that the model is confident in its classification.

Handling Tubes

Oncologists usually design the combination of markers in each tube such that each tube focuses on a different category of cell types or disease cubclasses. Therefore we treat tubes separately and train our models on each tube. Then we take the average output of the models on each tube and take the resulting

value as the final prediction for the given sample.

Table 3.6 shows a few entries of the resulting table. The first three entries show an example of samples our method has misclassified, and the last two entries are two samples for which our method agrees with their corresponding reported diagnosis. Figure 3.15 shows the distribution of the average normalized prediction values for the two classes.

Sample ID	Diagnosis	Target Value	Average Normalized Predicted Value
F09-0939	FL3	1	-0.81
F09-1578	FL12	1	-0.72
F09-0628	DLBCL	-1	0.50
F09-0578	FL12	1	1.65
F09-1471	DLBCL	-1	1.73

Table 3.6: A sample of prediction values compared to target values. The target value is -1 for DLBCL samples and 1 for FL samples. Average prediction value shows the average output of $l2$-SVM-linear method over three tubes.

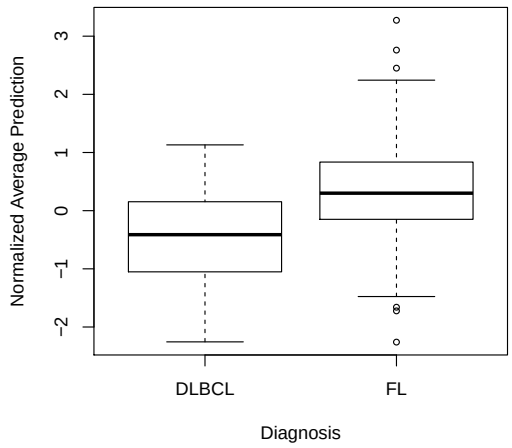
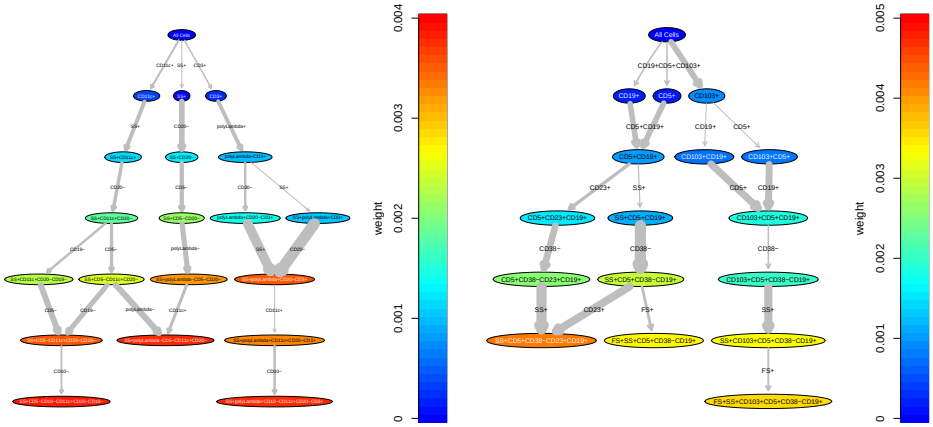


Figure 3.15: Distribution of average normalized prediction values for the two classes (DLBCL and FL).

Interpretation and Visualization

A support vector machine with a linear kernel, uses a linear combination of input features to classify samples. Because we transformed and normalized input features before training our models, it is possible to directly interpret feature weights assigned by the model as importance of features.

Using a leave-one-out cross-validation scheme over n samples, means we have n different trained models for each tube, given a specific method. Therefore for every tube, we have n different feature weights. We take the absolute value of the average feature weights over these n models as our final feature score. RchyOptimyx then uses these feature scores to visualize features with higher scores. Figure 3.16 shows the result of this analysis on the cohort.



(a) Tube 1

(b) Tube 2

(c) Tube 3

Figure 3.16: RchyOptimyx analysis shows important features classifying the two subtypes according to this cohort, as well as contribution of each marker to the estimated importance.

Now we present a post-analysis of our method for individual samples as mentioned before. Our goal is to give insight about why our models have misclassified an individual sample. An oncologist/pathologist can use this part of the analysis as a hint for further investigation on the patient's data.

As already mentioned, we follow a leave one out cross validation scheme. This means we train a model for each sample x_i, using all samples but x_i. Let this model be referred to as M_i. Then we use the weight vector of M_i as an indicator of feature significance. Now let the set $\mathcal{F}_{i,k}$ be the set of k features with largest absolute values in the weight vector of M_i. Then for each feature f_j in $\mathcal{F}_{i,k}$:

- Let $\mathcal{P}_{DLBCL,f_j} := \{P_{f_j,s} | s \in DLBCL\}$
- Let $\mathcal{P}_{FL,f_j} := \{P_{f_j,s} | s \in FL\}$

where $P_{f_j,s}$ is the cell count of the corresponding feature f_j for sample s, and $DLBCL$ and FL represent DLBCL diagnosed and FL diagnosed samples respectively. Then we draw the density plots corresponding to the two sets $\mathcal{P}_{DLBCL,f_j}$ and $\mathcal{P}_{DLBCL,f_j}$ over each other, and a vertical line indicating f_j of M_i. Figure 3.17 shows an example result of this analysis.

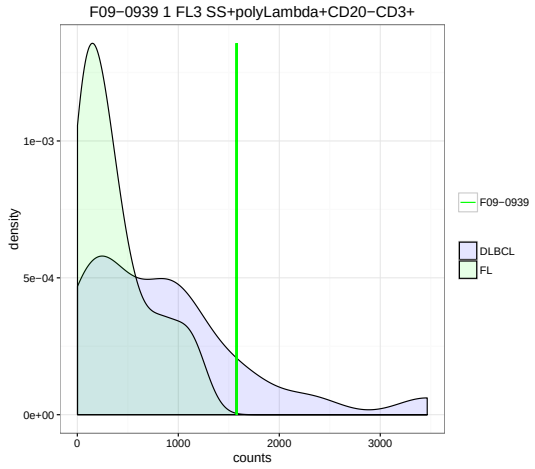


Figure 3.17: Sample density analysis: the X axis shows the cell count, and Y axis shows the density of samples with the corresponding cell count. Yellow and blue density plots represent FL and DLBCL samples respectively. The vertical line shows the cell count of the sample under study, and its color represents its diagnosed class, *i.e.* FL in this case.

Whenever there is a misclassification, it means there are features for which, the given sample lays where the opposite class has higher density. For a given misclassified sample x_i, and its corresponding set $F_{i,k}$, we draw the explained plots, and also their corresponding scatter plot of the cell population, which is similar to the result of a *manual gating*. Manual gating is a technique in flow-cytometry analysis to identify certain cell populations. It is an iterative process, and at each step the data is plotted using two chosen markers (a 2-D projection of the data), and the cells in a manually chosen area are selected for the next step. This process is often used by oncologists to identify certain cell populations

and hence a specific diagnosis. To give the oncologists a familiar representation of our detected cell populations, we provide a scatter plot of them as well.

Such a process results in plots shown in Figures 3.18 and 3.19. Figure 3.18 shows the analysis for three cell populations of an FL diagnosed sample. The three cell populations include:

- $SS^{+}polyLambda^{+}CD20^{-}CD3^{+}$
- $CD103^{+}CD5^{+}CD38^{-}CD23^{+}CD19^{+}$
- $FS^{+}CD7^{-}CD5^{+}CD2^{+}CD3^{+}CD4^{+}$

As shown on the left side of these plots, for all the three cell populations, the cell counts of the corresponding cell populations is more evidence for the sample being a DLBCL sample rather than an FL sample based on our models. The corresponding cells are shown on the right side of the density plots. Similarly, Figure 3.19 shows the analysis for a DLBCL diagnosed sample, and three cell populations contributing to our model labeling it as FL. These three cell populations are:

- $SS^{+}polyLambda^{-}CD5^{-}CD11c^{+}CD20^{-}$
- $SS^{+}FMC-7^{+}CD103^{+}CD38^{-}CD25^{+}CD19^{-}$
- $SS^{-}CD57^{+}CD7^{-}CD5^{+}CD8^{+}$

It is important to note that these cell populations are not the only factor in misclassification, but rather important contributors to it.

3.3.3 Summary

In this work we developed an interpretable method to classify flow cytometry samples, which can be used to enhance the internal quality assurance process of cancer hospitals and clinics. Although we further analyzed only misclassified samples, if the resources available to a QA team allow, they can continue to samples which are classified correctly, but with a lower confidence score. One can of course investigate the effectiveness of methods other than an $l2$-SVM with a linear kernel, as long as they can interpret the method and get values resembling importance of features from it. Also, as mentioned before, we divide cells into two groups according to each biomarker. We also showed in this chapter that this is not a constraint and cells can be divided into more than two groups if necessary. This might be important in some special cases where a certain biomarker usually divides into more than two groups.

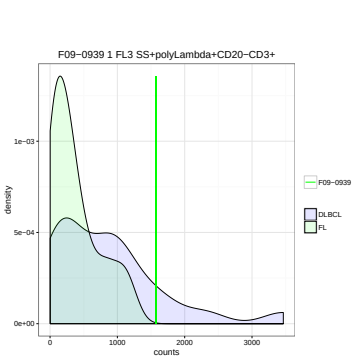
3.4 Contribution

Since I co-first authored the articles presented in sections 3.2.2 [4] and 3.2.3 [103], they are included in this chapter with minor modifications from the original

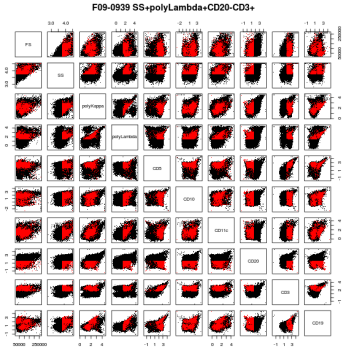
article, as a whole. However, for the purpose of this thesis, it is only fair for me to take credit only for the parts I was directly and heavily involved with.

In Section 3.2.2 my main contribution was to model the problem on graphs and solve it using a multiple k-shortest paths algorithm. I also took an existing implementation of the algorithm and wrote an R wrapper for it after fixing some pre-existing issues of that implementation. This resulted in two packages, one in CRAN and one Bioconductor.

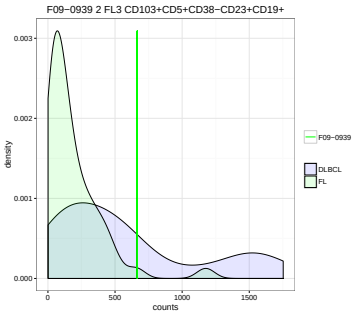
My contribution in Section 3.2.3 was to solve the problem using a dynamic algorithm, and its initial implementation in C++. The implementation then replaced the old implementation in the flowType Bioconductor repository.



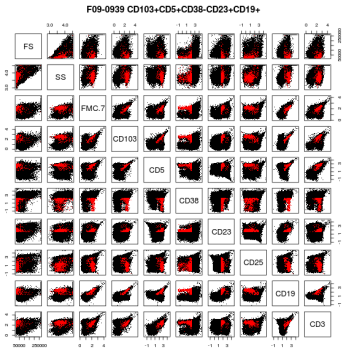
(a) Tube 1 - Density



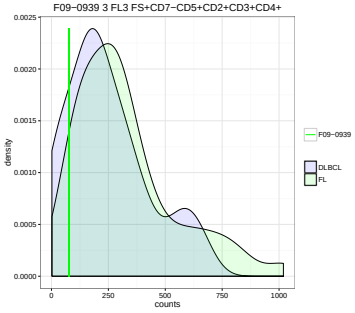
(b) Tube 1 - Cell Population



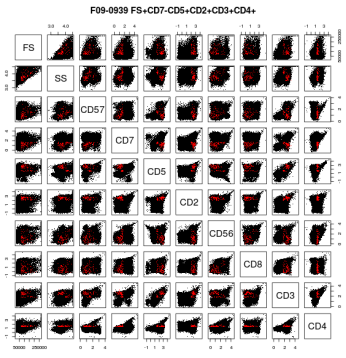
(c) Tube 2 - Density



(d) Tube 2 - Cell Population

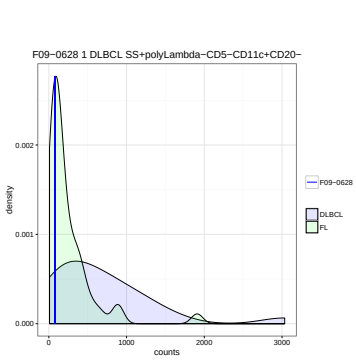


(e) Tube 3 - Density

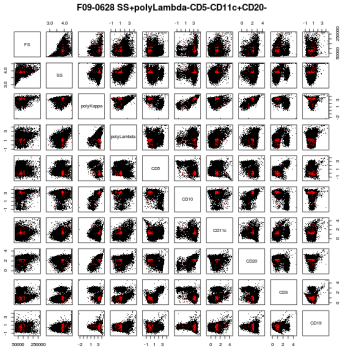


(f) Tube 3 - Cell Population

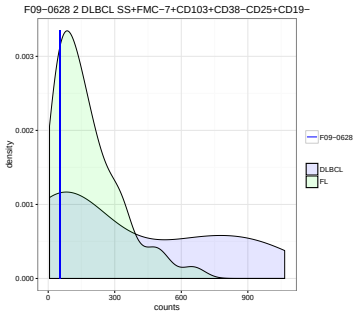
Figure 3.18: Density and scatter plots of the selected immunophenotypes for the sample F09-0939. The left column shows the kernel densities of observed number of cells of the corresponding immunophenotype in FL vs. DLBCL, and the number of cells of that same immunophenotype for the given sample. The right column shows the scatter plots of those selected cell populations.



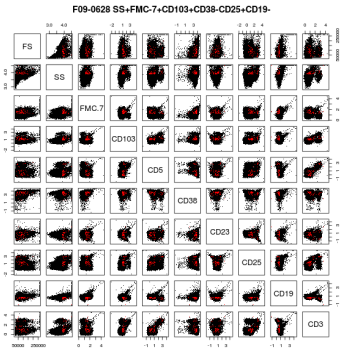
(a) Tube 1 - Density



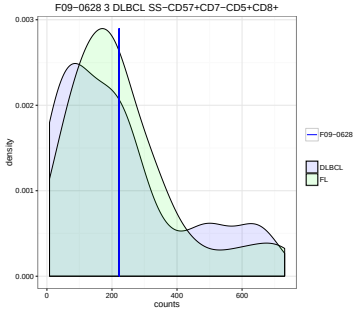
(b) Tube 1 - Cell Population



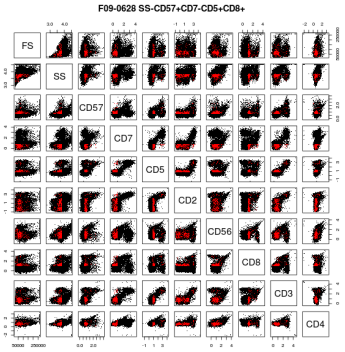
(c) Tube 2 - Density



(d) Tube 2 - Cell Population



(e) Tube 3 - Density



(f) Tube 3 - Cell Population

Figure 3.19: Density and scatter plots of the selected immunophenotypes for the sample F09-0628. The left column shows the kernel densities of observed number of cells of the corresponding immunophenotype in FL vs. DLBCL, and the number of cells of that same immunophenotype for the given sample. The right column shows the scatter plots of those selected cell populations.

"All models are wrong; some models are useful."

- George Box

4 Adaptive Learning

In contrast to Chapter 3 where we focused on data readily available in clinics, in this chapter we focus on more experimental data and hypotheses relevant to them. Specifically, in terms of data, we mostly use DNA methylation data (Section 2.3.7) and we partly explore the idea that not only DNA mutations are a factor in cancer, but epigenetic markers such as methyl components are also predictive of cancer, suggesting a potential causal relationship between epigenetic markers and cancer. Another main hypothesis we follow in this chapter is whether and how to design methods which adapt to every given sample. We call such methods adaptive, in the sense that they adapt to each given new input.

Like the methods introduced in the previous chapter, we do our best to interpret our methods to give biological insight to the biologists and pathologists. In this chapter we first discuss some common issues with regard to cancer and cancer data, showing the need for adaptive solutions in diagnostics. Then we present an extended version of a previously published article in Section 4.2 [66]. Finally we present an idea initially designed and developed with my collaboration, and then was continued more concretely by other members of the team after me (Section 4.3).

4.1 Challenges in Cancer Data

In this chapter we try to tackle some challenges which tend to occur while analyzing cancer data. Before we start with the methods, here we explain those challenges, which include noise, batch effects, and cancer (tumor) heterogeneity. The first two (noise and batch effects) are not specific to cancer and are challenges related to the nature of the data and the way the data is collected. In this context,

by cancer data we mean the data generated in labs using experiments on biopsies from cancer patients. These experiments include but are not limited to gene expression profiling such as microarrays [155], DNA and RNA sequencing data, epigenomics data such as methylation profiles, etc.

Noise in the data is due to two main different sources. One is the fact that the data is generated through a biochemical process and factors such as impurity or room temperature may affect the results [117]. Another important factor is that biochemical processes are stochastic in nature which leads to potentially getting different results running the same experiment on the same sample twice [72, 71].

Batch effects, on the other hand, are the result of the experiment being run under different setups [111]. This includes the chemical reagents used in experiments. A lab, for instance, may order a binding agent with a higher sensitivity once the old batch is running out. Or different labs may be using markers manufactured by different providers. Machine calibration is another common factor for slightly different measured values. Although these effects are consistent within a batch, a dataset usually includes data from different batches, affecting the accuracy and at times the validity of the analysis. For instance, two different labs may be providing the data for two different subclasses of a disease, each providing the data for one subclass; and the method perfectly classifying the two subclasses while in reality only detecting an effect in the data which is due to batch effects.

Another major challenge stems from the heterogeneity of cancer and cancer tumors [88]. Cancer cells are prone to a much higher rate of mutation than healthy cells and as a result, the cells in a single tumor may have varying gene expression profiles. This means taking two separate biopsies from the same tumor may result in different gene expression profiles which then may affect the analysis. However, since cancer can start from one single cell, the cells of a tumor tend to share the same genetic background driving the cancer and many methods, including ours, try to detect those factors.

Taking the above challenges into account, our goal is to come up with methods detecting these issues based on the observed information in a single given sample data. We propose two approaches, both of which try to adapt themselves differently to each given data point. We exploit the fact that our input features have genetic background and that genes and proteins work in complex networks, hence are dependent and correlated random variables. We extract patterns which occur between the input features by looking at the training dataset, and use those expected patterns to identify the features of a given new sample data whose values are less affected by the abovementioned factors, and rely mostly on those features to give an output prediction.

In the remaining of this chapter, we first investigate a pre-existing kernel based method which takes into account a protein-protein interaction network as an auxiliary input. Unfortunately our conclusion is that the method cannot achieve its promise on real data. Then we continue with proposing two other methods in Sections 4.2 and 4.3.

4.2 RatBoost

Over the past few decades, biology has transformed into a high throughput research field, both in terms of the number of different measurement techniques as well as the amount of variables measured by each technique (e.g., from Sanger sequencing to deep sequencing), and is more and more targeted to individual cells [130]. This has led to an unprecedented growth of biological information. Consequently, techniques that can help researchers find important insights into the data are becoming increasingly important. Predicting survival of cancer patients based on measurements from microarray experiments has been a field of great interest, but there is often very little overlap between the important genes or biomarkers identified by different studies [41]. Several reasons have been suggested to explain these findings (e.g., heterogeneity of cancer samples or insufficient sample size). Attempts have been made to incorporate additional information from other sources, such as protein-protein interaction (PPI) networks, to make the predictions more robust [30]. One of the latest approaches integrates network and expression data by introducing a network-induced classification kernel (NICK) [81]. Although this method exhibits state-of-the-art performance, the way it penalizes genes that are connected to not-predictive genes can result in selection of isolated features as important features for prediction. We observed this bias of the method towards isolated nodes on additional experiments on synthesized data as shown in Section 4.2. Another issue is that in PPI networks, genes or proteins, which have been known to researchers longer and are well-known, are studied more and therefore have more edges connected to them; whereas less well-known genes and proteins are in sparser areas of the network. This bias might further affect the judgment of methods like NICK that use a PPI network as an input. Consequently, we rely on the fact that such networks exist between genes and proteins, but we do not take them as an input. If there is a dependency between input features, which is the case in many biological settings, our method can benefit from this effect. Otherwise, it is reduced to a standard ensemble method. Furthermore, a central assumption underlying many methods is that all data are drawn from the same unknown underlying distribution. This may not be the case, especially for heterogeneous cancer samples, and in particular not for all measured genes.

In this work, we introduce a method that is aware of this potential bias and utilizes an estimate of the differences during the generation of the final prediction method. For this, we introduce a set of sparse classifiers based on $l1$-SVMs [21], where each set of features used by one classifier is disjoint from the selected feature set of any other classifier. Furthermore, for each feature chosen by one of the classifiers, we introduce a regression model that uses additional features and is based on Gaussian process regression. These regression models are then used to estimate how predictable the features of each classifier are for each test sample. This information can then be used to find a confidence weighting of the classifiers, *i.e.* up-weighting classifiers with high confidence and down-weighting classifiers with lower confidence, for each test sample. Schapire and Singer show that incorporating confidences of classifiers can improve the performance of an

ensemble method [126]. However, in their setting, confidences of classifiers are estimated using the training data and are thus fixed for all test samples, whereas in our setting, we estimate confidences of individual classifiers per given test sample. Another related work includes mixture of experts, in which the model trains a set of neural networks and uses a gating network to set the weights of the networks [65]. One issue with their method is that neural networks with lower performance will not be optimized as much as networks with better performance on training data since the gate module down-weights the error propagated back to them. Also training of the gating network is interconnected with the neural network experts and afftects training of those modules. Our method, in contrast, trains each module independently using all training samples, and their reliability does not affect how they are trained. Bayesian hierarchical mixtures of experts takes a more similar approach, but the method is complex, and it has a high time complexity to train the architecture of the hierarchy [18].

We show that this method exhibits state-of-the-art performance for different cancer types, with gene expression or methylation datasets as the input. Since the weighting of the classifiers is customized for each test sample, the estimated confidences can offer insights into the specific characteristics of each individual's cancer. To facilitate interpretation of the model, we then create a visualization of the important genes found through this analysis for each test sample. Additionally, we show how the important genes of the training set can be found using our learning method and cross validation.

Our idea might resemble ensemble feature selection, which involves aggregating multiple feature scores from several scoring mechanisms. These scoring mechanisms vary from being several different methods, to being the same method applied to different parts of the data such as a random cross validation scheme [125]. This idea has been studied further by other researchers and they introduced two different methods to aggregate scores from different models. They use an ensemble of support vector machines which on its own has been used to select features in a given dataset in other works [59]. Although we use an ensemble of support vector machines, our goal is not to give a ranking to features of the dataset, rather to find multiple parsimonious gene sets that are predictive of the outcome on their own, and use all of them in parallel to predict the outcome.

Similar to this approach, in another work, iRDA uses a different approach and can report multiple parsimonious gene sets [78]. One significant difference between iRDA and our work is that we have an embedded prediction approach using these sets, which iRDA lacks. Furthermore, gene sets are somehow ordered in iRDA according to their "strength", and within each set, redundant genes are removed. In our model redundant genes can be included in two different ways. One is within different individual learners. For example, if genes g_1 and g_2 are both strong but redundant, individual learner 1 might include g_1, and individual learner 2 might include g_2. Also, if there are more redundant or related genes in the gene pool, they will be used to estimate how reliable g_1 and g_2 are. Therefore instead of dismissing them, we exploit the fact that they exist.

Related to sorting genes and testing for significance of a reported gene set,

Gene Set Enrichment Analysis (GSEA) and its modifications are a commonly used tool [132, 136]. GSEA based methods rank genes depending on how much they relate to the outcome. The choice of relationship is rather free and can vary from Pearson correlation to mutual information. Then for a given gene set, a p-value is calculated by estimating how often a random gene set appears before the given set on the list. There have been several modifications and improvements to the method [97, 38]. Although it is true that GSEA is used to assess the relevance or importance of a given set to the outcome, we need to remember that a particular gene set might consist of genes that are not necessarily important on their own, but are predictive once considered together. Our method does not consider genes individually whereas GSEA does to sort the genes in the first place. Therefore we believe GSEA based methods are not suitable to assess how well our method performs.

Analysis of NICK

Lavi, et al. modified the standard SVM formulation (Formula 2.12) as shown in Formula 4.1 [81]. This formulation adds a penalty function to penalize weight differences if their corresponding features are connected in the given graph. The intuition behind the idea is that if features are connected in the network, their weight should be somehow similar.

$$\min_{\mathbf{w},w_0} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}\beta \sum_{(j,k)\in E} (w_j - w_k)^2 \right\}$$

s.t.:

$$\forall i \in \{1, \cdots, n\} : (\mathbf{w}\mathbf{x}_i + w_0)y_i \geq 1 \tag{4.1}$$

In the above formulation, E is the set of edges of the given network. They also show how to derive the dual of the above optimization problem as shown in Formula 4.2:

$$\max_{\alpha} \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{L})(\mathbf{L}^T \mathbf{x}_j) \right\}$$

$$\mathbf{L}\mathbf{L}^T = (\mathbf{I} + \beta\mathbf{B})^{-1}$$

s.t.:

$$\forall i \in \{1, \cdots, n\} : \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\forall i \in \{1, \cdots, n\} : \alpha_i \geq 0$$

Laplacian matrix:

$$\mathbf{B} = \mathbf{D} - \mathbf{A} \tag{4.2}$$

In Formula 4.2, $\mathbf{D}$ is a diagonal matrix having degrees of nodes on its main diagonal, $\mathbf{A}$ is the adjacency matrix, and $\mathbf{B}$ is called the graph Laplacian matrix.

The benefit of the above formulation is that the input vectors can be transformed using the matrix $\mathbf{L}$, which itself comes from the Cholesky decomposition of the matrix $(\mathbf{I} + \beta\mathbf{B})^{-1}$. The parameter β in both Formulea 4.1 and 4.2 sets how much we penalize weight differences for connected vertices.

After training the model on the data using the above formulation, we can calculate back the vector $\mathbf{w}$ using Formula 4.3. We use the vector $\mathbf{w}$ to investigate which nodes and pairs of nodes are given a relatively high value compared to other features.

$$\mathbf{w} = (\mathbf{I} + \beta\mathbf{B})^{-1} \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{4.3}$$

We then use calculated feature weights in the vector $\mathbf{w}$ of both normal and modified SVM (NICK) to show which feature pairs are selected as important in the model, as shown step by step bellow. Please note that NICK transforms the data using the matrix $\mathbf{L}$, and then solves a normal SVM on the transformed data, and therefore in the following whenever we refer to *transformed data*, it refers to the NICK method.

1. Solve SVM problem for original and transformed data.
2. Calculate $\mathbf{w}$ for both models.
3. Compute for each pair of nodes, for each model:

$$Score(i,j) = \frac{|w_i| + |w_j|}{2} \times e^{-max(d_G(i,j),1)} \tag{4.4}$$

4. Report pairs with highest scores for both trained models.

In order to evaluate the method, we need to synthesize some data because in the real data it is not clear which features are the true discriminating features. For this purpose, we randomly generate a graph, and assign nodes to three different classes. Nodes in this graph represent genes/features in the dataset. Each feature is a random variable sampled from a Gaussian distribution. If the node is independent of the target class, it gets its value from a Gaussian distribution regardless of the target class. If the feature is selected as a *signal* node, then it takes its value from two different Gaussian distributions that differ in their mean, depending on the target class of the sample. Some of these signal features are connected only to random features, and some are connected to other signal nodes. We call connected signal features a *pathway*. The generated graph and an example of selected feature nodes are shown in Figure 4.1.

To generate our synthesized dataset, for each data point, and each feature of that data point, we first assign a class to that data point, and according to the assigned class, we sample from the corresponding distribution, according to the following functions:

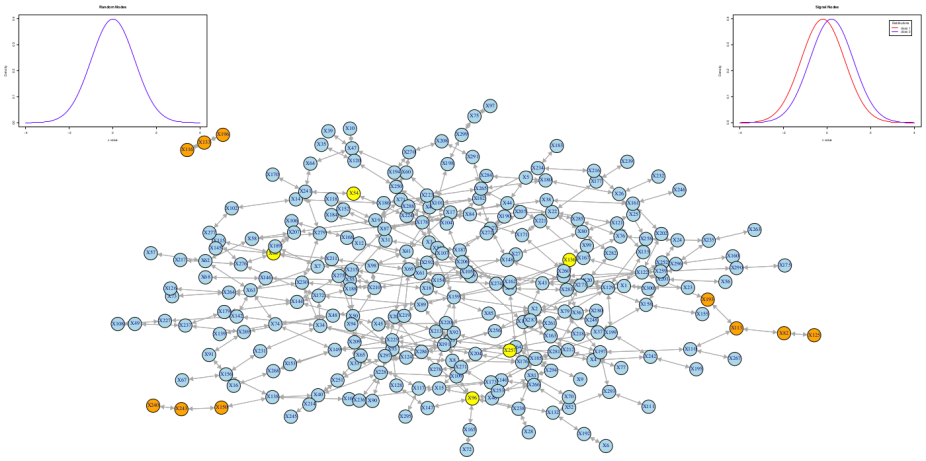


Figure 4.1: Blue: random gene, Orange: Signal node being a member of a pathway of signal nodes, Yellow: A lonely signal node

- Signal nodes (genes): $f(n) = \begin{cases} N(-\mu, 1) & \text{if } n \text{ is in class 1} \\ N(\mu, 1) & \text{if } n \text{ is in class 2} \end{cases}$
- Random nodes (non-signal genes): $f(n) = N(0, 1)$

We perform three experiments. Each experiment uses the same graph structure, but has a different set of signal nodes. First we put *pathway* nodes on the boundaries of the graph, then we move one *pathway* deep into the graph structure, and in the last experiment all *pathway* nodes are inside the graph structure. These three scenarios are shown in Figures 4.2, 4.3, and 4.4 respectively.

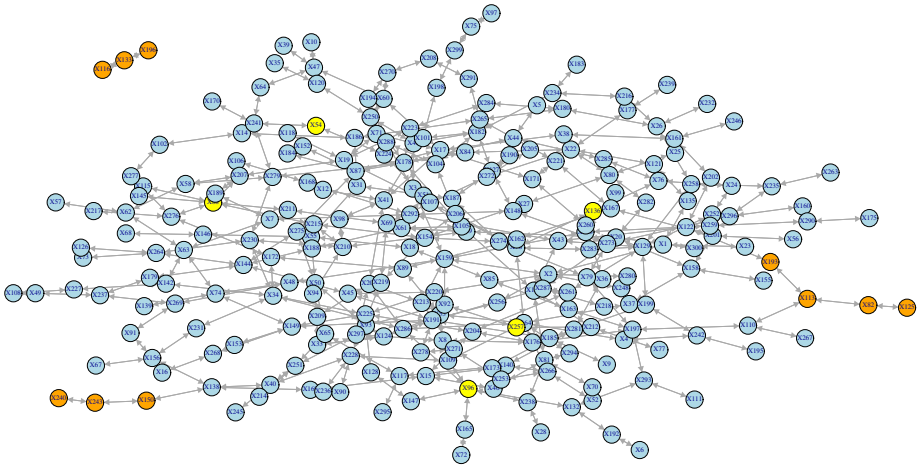
Figure 4.2(b, c) presents node pairs with highest assigned scores as calculated in Formula 4.4, comparing normal SVM and NICK. Orange and yellow colored cells are pathway and lonely signal nodes in the graph accordingly. This experiment shows that NICK discovers signal gene pairs more effectively than a normal SVM, using the graph structure.

Figure 4.3 shows the experiment, in which one of the pathways is located inside the network. As illustrated in Figure 4.3(b, c), NICK mostly chooses the pathway nodes located on the boundaries of the network.

Figure 4.4 illustrates the fact that non-signal features down-weight and penalize signal features when connected to them. In this example, a normal SVM detects more signal nodes than NICK does.

We used the network provided in [9] for NICK to classify Van 't Veer data [145]. As expected, we realize that NICK prefers nodes outside the network to the nodes that are deep into the given network. This is shown in Figure 4.5, comparing preferred nodes in a normal SVM and NICK. The first column is the gene ID, and the second is its corresponding degree in the given graph.

These experiments all together, show how such a modification in SVM optimization problem gives a bias towards genes that are not *hubs*. This is problem-



(a) Corresponding network

Original			
X196	X196	X53	X53
X233	X233	X39	X39
X88	X88	X196	X133
X116	X116	X127	X127
X197	X197	X127	X148
X148	X148	X150	X150
X148	X273	X116	X133
X160	X160	X96	X96
X95	X95	X273	X273
X88	X115	X40	X40
X53	X8	X53	X164
X195	X195	X56	X56

(b) Discovered nodes (no NICK)

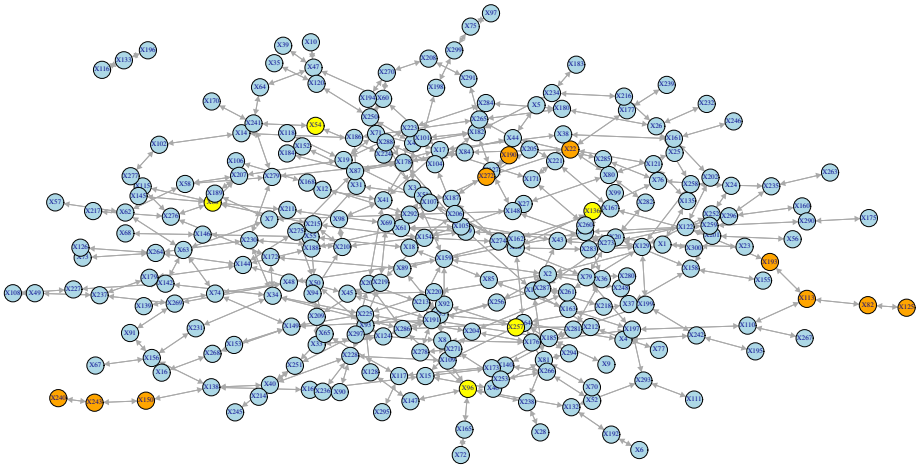
Transformed			
X196	X196	X233	X233
X196	X133	X133	X133
X133	X116	X116	X116
X95	X95	X240	X240
X39	X39	X240	X243
X59	X59	X106	X106
X243	X243	X106	X168
X114	X114	X168	X168
X243	X150	X56	X56
X39	X47	X298	X298
X150	X150	X247	X247
X125	X125	X83	X83

(c) Discovered nodes (NICK)

AUC (Original):	60.6
AUC (Transformed):	62.4
wc p-value (paired):	5.669e-09

(d) Performance measures

Figure 4.2: An easy example: here all signal pathways are on the border of the network.



(a) Corresponding network

Original			
X190	X190	X104	X104
X233	X233	X190	X272
X277	X277	X88	X88
X190	X127	X165	X165
X272	X272	X272	X22
X106	X106	X165	X96
X150	X150	X250	X250
X88	X215	X22	X22
X51	X51	X28	X28
X73	X73	X35	X35
X162	X162	X113	X113
X112	X112	X277	X102

(b) Discovered nodes (no NICK)

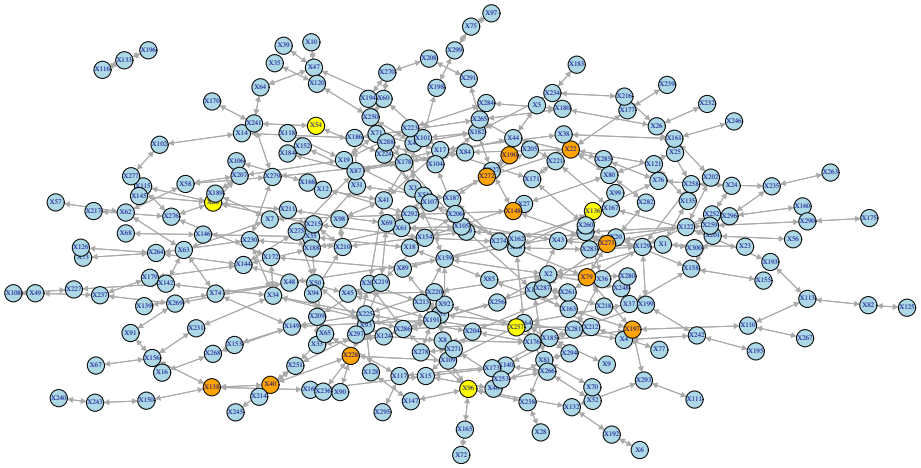
Transformed			
X233	X233	X190	X190
X112	X112	X240	X240
X190	X272	X240	X243
X86	X86	X243	X243
X243	X150	X190	X127
X150	X150	X272	X272
X246	X246	X298	X298
X106	X106	X125	X125
X35	X35	X125	X82
X247	X247	X272	X69
X272	X22	X82	X82
X100	X100	X257	X257

(c) Discovered nodes (NICK)

AUC (Original):	60.1
AUC (Transformed):	61.5
wc p-value (paired):	1.383e-06

(d) Performance measures

Figure 4.3: A medium example: here some signal pathways are on the border of the network.



(a) Corresponding network

Original			
X190	X190	X101	X101
X233	X233	X190	X272
X88	X88	X297	X297
X190	X127	X93	X93
X26	X26	X138	X138
X272	X272	X272	X22
X101	X41	X123	X123
X22	X22	X101	X198
X146	X146	X228	X228
X278	X278	X72	X72
X88	X115	X96	X96
X148	X148	X112	X112

(b) Discovered nodes (no NICK)

Transformed			
X233	X233	X190	X190
X112	X112	X190	X272
X86	X86	X190	X127
X272	X272	X272	X205
X205	X205	X146	X146
X146	X68	X68	X68
X298	X298	X272	X22
X90	X90	X127	X127
X100	X100	X272	X69
X297	X297	X72	X72
X127	X148	X155	X155
X247	X247	X196	X196

(c) Discovered nodes (NICK)

AUC (Original):	60.2
AUC (Transformed):	62.5
wc p-value (paired):	8.151e-13

(d) Performance measures

Figure 4.4: A hard example: here none of the signal pathways are on the border of the network.

Original	
Node	Degree
85453	12
6605	98
56886	26
10640	16
8817	152
56894	28
5733	150
57758	8
7532	86
51	172
7566	16
3267	56
89953	4
5713	126
5193	32
5365	70
10874	132
5982	172
92140	20
332	328

(a) Discovered nodes (no NICK)

NICK	
Node	Degree
9917	0
84279	0
197370	0
51143	0
58475	0
55585	0
25949	0
54892	0
126695	0
57168	0
10456	0
148223	0
9742	0
253558	0
342527	0
10175	0
83930	0
57035	0
145482	0
57465	0

(b) Discovered nodes (NICK)

Figure 4.5: Comparison of selected nodes on Van 't Veer data [145] using NICK and a normal SVM.

atic considering many of those hubs in the network are partially, if not mostly, hubs due to the fact that they were discovered earlier and have been investigated the most. Therefore those are the most well-known genes, which in many cases happen to be biologically most relevant genes. A method such as NICK tends to penalized them because they are connected to many genes that are irrelevant to the disease in study. This experiment is our motivation to use the fact that such a biological network exists, but not to use it directly in our method.

4.2.1 Methods

Materials

Data Sources: In this article, our method is applied to two different data types: gene expression data and DNA methylation data, which we retrieved from The Cancer Genome Atlas (TCGA) [141]. TCGA is a joint effort of the National Cancer Institute and the National Human Genome Research Institute to advance the understanding of the molecular basis of cancer. They provide access to the

different measurements from cancer samples that have been analyzed to external researchers. Samples are categorized according to diagnosed cancer from which we use the following groups:

- *Acute Myeloid Leukemia (LAML)* [143]: At the time of writing, the dataset includes 200 samples. 194 samples contain methylation data and we use the part of the data measured by JHU-USC HumanMethylation450 arrays. 173 samples contain mRNA data measured by HG-U133 arrays. In this article the methylation data is referred to as TCGA-LAML. Among available characteristics of samples, "risk group" and "vital status" are chosen as target classes. These labels show the aggressiveness of the disease. In our analysis, regarding risk group, {favorable} and {intermediate/normal, poor} samples form our two group, and in the analysis of vital status, {alive} and {dead} samples form our two groups of samples.

- *Breast invasive carcinoma (BRCA)* [142]: This dataset includes 993 samples with clinical data, and we use the methylation data component measured by JHU-USC HumanMethylation450 arrays. Only very few samples in this dataset are indicated as having metastasized (8 samples). Hence the data are analyzed according to "tumor size", "affected nearby lymph nodes", "stage", and "estrogen receptor". Estrogen receptor was shown to be an important factor in prognosis [75], and along with other factors directly affects the decision for therapy [57, 98]. For tumor size {T1, T2} samples are one category and {T3, T4} the other category; in order to analyze affected nearby lymph nodes, {N0} is compared to {N1, N2, N3}; stage is analyzed as having {stage I, stage II} vs. {stage III} samples. Estrogen receptor status of samples is either positive or negative, and they form our two classes.

Data Preprocessing: To prepare gene expression data for analysis, microarray probes are mapped to their respective gene. If there are multiple probes for a gene, the median reported gene expression value of those probes is adopted as the gene expression for that gene.

Preparing the methylation data, we use the nearby gene for each methylation site available for each sample and each methylation site. The median beta value of methylation sites mapped to each gene is taken as the methylation value of the corresponding gene. In this process only methylation sites located on the promoter region of a gene are considered and others are discarded.

Learning a Mixture of Disjoint Classifiers

When dealing with cancer, we need to consider the fact that tumors of the same type of cancer can be very different in nature and they are usually classified as different cancer subtypes. In fact, even one single tumor can be very heterogeneous [63]. This means that the malignancies causing the cancer to happen are genetically different between subtypes, or even within subtypes, and it is possible to have multiple underlying cellular processes causing a particular cancer. Also

it is important to note that the nature of our given data is such that the input features are properties measured from genes, e.g. gene expression or methylation values, and these variables are correlated and statistically dependent on each other. Our method tries to exploit these properties of the problem to infer an interpretable model with state-of-the-art performance.
Our method can be characterized by the following key parts:

Training phase:

- Fit several individual classifiers to the data, in such a way that the features of the data they use are disjoint sets.

Prediction phase:

- Calculate the prediction confidence of each individual classifier by:
  - Estimating the reliability of input features of the classifier;
  - Estimating the confidence of the output based on the decision values.
- Calculate a weighted prediction label based on the individual classifier confidences.

Properties of the Individual Classifiers: A wide variety of classifiers can be used within our framework. One requirement is that the classifier is regularized (*i.e.*, the stronger the regularization, the less complex the model gets and consequently the less features are used). The classifier is also required to report the probability of its calculated output, or to give a decision value according to which it chooses the predicted class. We use an $L1$ regularized SVM for this purpose with a linear kernel [21]. The $L1$ regularization makes the SVM sparse, *i.e.* using only a few input features, and the linear kernel allows us to infer which features are used in the decision function of the SVM after it is fit to the data.

Training the Individual Classifiers: The model starts with no individual classifier and an empty set of excluded features. In each step, the excluded set of features is removed from the data, then a classifier is fit to the data. Next the features used by the most recent trained classifier are added to the excluded set. In the case of a linear kernel SVM, this is achieved by finding features with a non-zero coefficient in the model. This way the features being used by classifiers are disjoint and might represent different underlying causes of groups into which samples are to be classified.

Combining Classifiers by Estimating Confidences of Individual Predictors: Given a set of classifiers, the question is how to combine them to come up with a joint prediction value for each test sample for which we want to predict the output label. The intuition behind combining the classifiers is to put more weight on classifiers that use features whose behavior is similar to the training data. This is motivated by the fact that some parts of the test data might behave very differently to the training data, meaning that a classifier using these features should have lower performance than a classifier using features that are

distributed similarly to the training data. Therefore we need to evaluate the reliability of the input features of each individual classifier. In scenarios like gene expression or methylation analysis, we usually have many input features. Furthermore, many features are correlated and statistically dependent. The idea of our new method is to build separate prediction models for each feature of each classifier. These prediction models can then be used to obtain a confidence for the feature in a given test sample. These confidences can then be combined for each classifier to give a weighting of the classifiers for the given test sample. To evaluate an observed feature f, we try to choose a few statistically dependent features, and fit a model to predict f. To find these features, first the estimated maximal information coefficient (MIC) of all other features with feature f is calculated [120]. Then, features having MIC value within the top 5% or the 5 features with highest MIC with f (if the top 5% features consist of less than 5 features), are selected as predictors of f. Given a test sample, the closer the predicted value of f is to the observed value, the more reliable it is. To quantify this, we need to not only know the predicted value of the feature, but also a confidence interval for that prediction. This can be achieved using Gaussian processes, which give the mean and variance of the posterior probability under the condition of observed values for selected features. A weighted average of these values gives us the overall reliability of the features of an individual classifier. A schematic view of the trained classifier is shown in Fig. 4.6.

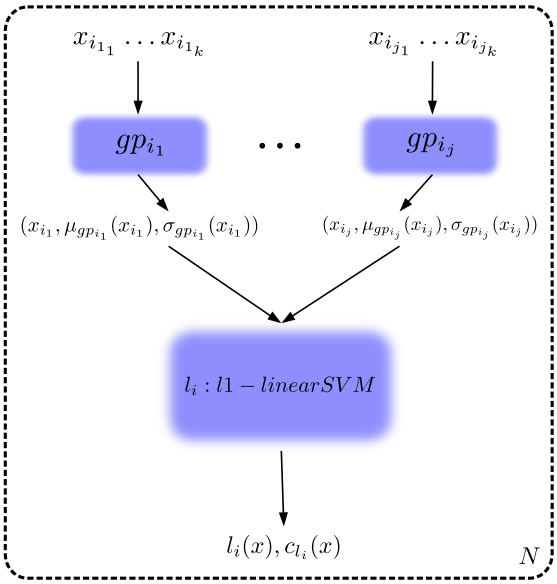


Figure 4.6: Schematic view of the method

In addition to the confidence in the classifier estimated by looking at the confidences of its individual features, we also account for the confidence that the classifier has in the prediction label of the test sample. If the method supplies such

a confidence value (e.g., Gaussian processes), we can directly use it. Otherwise, we estimate it using the decision value. In our setting, the linear SVM gives a decision value whose sign defines the predicted class. Using these values we estimate a confidence for each individual classifier. Several approaches exist for deriving a confidence from the decision values [82]. Whether these or other additional methods could lead to further improvements of our method, will be topic of further study.

More formally speaking, define X to be the set of input samples, X_s to be the input vector of sample s, y_s and $\hat{y}_s$ to be respectively the original label and predicted output of sample s, Δ to be the set of individual classifiers, l_i to be an individual classifier, Φ_{l_i} the set of input features of classifier l_i, $l_i(X_s)$ to be the label predicted by classifier l_i for sample X_s, and f to be a feature, $X_{s,f}$ to be the observed value of feature f in sample X_s, $|w_{l_i}(f)|$ to be the absolute value of the weight of feature f in the decision function of classifier l_i, and g_f to be the Gaussian process predicting feature f using feature set Φ_f. Also $\mu_{g_f(X_s)}$ and $\sigma_{g_f(X_s)}$ are the mean and standard deviation of the posterior probability given by Gaussian process g_f under the condition of observing values of features in Φ_f, and μ_{l_i} and σ_{l_i} are respectively the expected mean and standard deviation of the decision value of classifier l_i. Here F is the cumulative distribution function of a standard normal distribution.

The training phase of the model is shown in Fig. 4.7, in which, N is the number of individual learners to be included in the model, Φ_l is the union over all Φ_{l_i} and $X_{-\Phi_l}$ is the input X after discarding all features of the set Φ_l. TOP is the function which selects the maximum of the top 5 and top 5% features f' of all features ordered by MIC with feature f.
Now given a test sample X_s, the estimated confidence of a feature f is:

$$c_f(X_s) := 2 \cdot F\left(-\left|\frac{X_{s,f} - \mu_{g_f(X_s)}}{\sigma_{g_f(X_s)}}\right|\right) \tag{4.5}$$

Then the overall feature reliability or confidence of a classifier l_i is estimated as:

$$c^1_{l_i}(X_s) := \frac{\sum_{f \in \Phi_{l_i}} c_f(X_s) \cdot |w_{l_i}(f)|}{\sum_{f \in \Phi_{l_i}} |w_{l_i}(f)|} \tag{4.6}$$

Also the estimated output confidence of the classifier l_i is:

$$c^2_{l_i}(X_s) := 1 - 2 \cdot F\left(-\left|\frac{l_i(X_s) - \mu_{l_i}}{\sigma_{l_i}}\right|\right) \tag{4.7}$$

and the final confidence of the classifier l_i is then:

$$c_{l_i}(X_s) := c^1_{l_i}(X_s) \cdot c^2_{l_i}(X_s) \tag{4.8}$$

Finally, the predicted class $\hat{y}_s$ is calculated as the sign of a weighted vote among individual classifiers:

$$\hat{y}_s := \text{sign}\left(\frac{\sum_{l_i \in \Delta} c_{l_i}(X_s) \cdot l_i(X_s)}{\sum_{l_i \in \Delta} c_{l_i}(X_s)}\right) \tag{4.9}$$

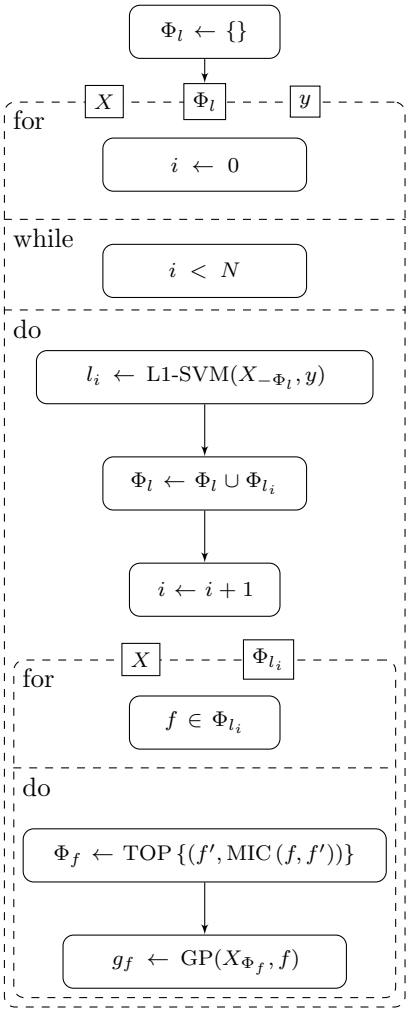


Figure 4.7: UML activity diagram of the training process

Visualization of Model Predictions

The interpretation of the model can be understood on two different ways. First we assume for a given training dataset, the model is trained and a new test sample is given. For the given test sample it is possible to visualize the reliability of each used feature in individual classifiers, as well as the overall confidence of each individual classifier. Used features can be superimposed onto a PPI network as well as their reliability and the confidence of their respective individual classifier. Gene expression and methylation level measurements from cancer samples are usually very noisy. Furthermore, cancers are usually very heterogeneous. Additionally, there might be different subgroups for each interesting group (e.g., cancer stage), for which the importance of the features also differs. To get a global picture of the important features, we therefore evaluate how often certain features are selected by the classifiers using 100 random train test partitionings with 80% of the data for training and 20% of the data for testing. To visualize high confidence relationships between features, we create a graph which has a node for every chosen feature in any of the 100 train partitions in any of the individual classifiers. The weight of an edge (s, t) is defined as the number of times the respective features have occurred together in an individual classifier. Then, all edges with low weights are discarded. In order to find a threshold to prune edges according to their weights, a Gaussian kernel density estimate is fit to the weights of the edges, and the threshold is chosen at the 90th percentile. Nodes that have an appearance frequency higher than the threshold are labeled by their gene names and edges having a higher weight than the threshold are kept in the graph.

For illustration purposes, choosing the regularization parameter is done in a way to maximize the number of genes selected with high confidence, as well as minimizing the number of genes pruned out in the process. It is important to remember that considering the results of the method under different regularization parameters is essential to make sure the selected genes possess a high confidence and are also stable regardless of sampling of the training dataset.

Implementation Details

To compare the performance of our method with other methods, the implementations present in Python *scikit-learn(0.14)* package are taken. In the case of stochastic gradient boosting, the representing class is *GradientBoostingClassifier*, the number of classifiers is set to 100, and to make it sparse and prevent over-fitting, the maximum number of features for splits in trees is set to 5, and the maximum number of layers is set to 2. For AdaBoost, *AdaBoostClassifier* is used, which is an implementation of AdaBoost-SAMME [164], with weak learner set to *DecisionTreeClassifier* with maximum depth set to 2, and the number of weak classifiers set to 100. Parameters of the two boosting algorithms are chosen by a grid search on their parameter space over all the datasets and selecting the parameter sets which give a robust and stable result over all experiments.

As an SVM, ν-SVM with $\nu = 0.25$ is used, once with a linear kernel, and once

with an RBF kernel; γ parameter of the RBF kernel is set to (num of features)$^{-1}$. The ν parameter is set to the maximum value for which the optimization function is solvable with *libsvm* for all analyzed datasets [26]. Smaller values cause the SVM to overfit to the data and not generalize well. The Gaussian process's correlation function is a squared-exponential, and MIC is estimated using *minepy* package [6].

The PPI network used in our analysis is from the Human Protein Reference Database (HPRD) [109]. Almost all edges and relationships between proteins that are added to this database are manually extracted from literature by biologists, hence it has a lower rate of edges included in the database for which there is no evidence in the literature.

4.2.2 Results and discussion

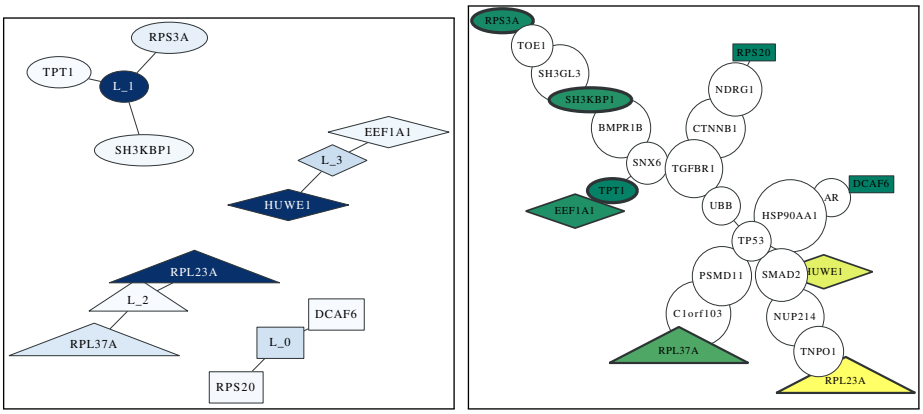
Interpretability of Predictions

Here we present the results of running the method on the TCGA-LAML gene expression dataset.

Visualization of Features Important for a Particular Test Sample: Having a model trained on the data, and given a test sample, it is possible to infer and visualize which individual classifier(s) is (are) influencing the prediction most. To this end, individual learners as well as the features they use are visualized as in Fig. 4.8(a). In this figure, nodes with labels starting with "$L\_$" represent individual classifiers, and other nodes are labeled with their respective gene name. The color of the node shows its confidence compared to other nodes; the darker the node, the higher the confidence. In the case of a gene, it is the confidence or reliability of the feature (c_f), and in the case of an individual classifier, it is the overall estimated confidence (c_{l_i}). Edges show which classifier is using which genes in its decision function. The shape of a node represents the individual classifier they belong to.

To get a better overview of the individual features that were chosen by the classifiers for the particular test sample, we visualized the corresponding genes on a graph containing information about the PPI network in Fig. 4.8(b). We extracted the PPI information from HPRD as explained before. This way, it is possible to find over- or under-regulated pathways that might be responsible for the label (e.g., cancer stage) of the test sample. Since PPI networks can be quite dense, we removed parts of the induced network. For this purpose we computed each shortest path between all pairs of selected features. Then, the minimum spanning tree of that section was plotted, after removing branches with no selected feature.

Most of the features chosen by any of the classifiers (colored nodes) are not connected to any other chosen feature. It is known that there is in many cases a correlation between expression value of the genes whose corresponding proteins interact [67]. Therefore, a regularized model will only choose a subset of the correlated features. This explains the observation that features selected by a single model can be distant from each other on a PPI network; but if multiple



(a) (b)

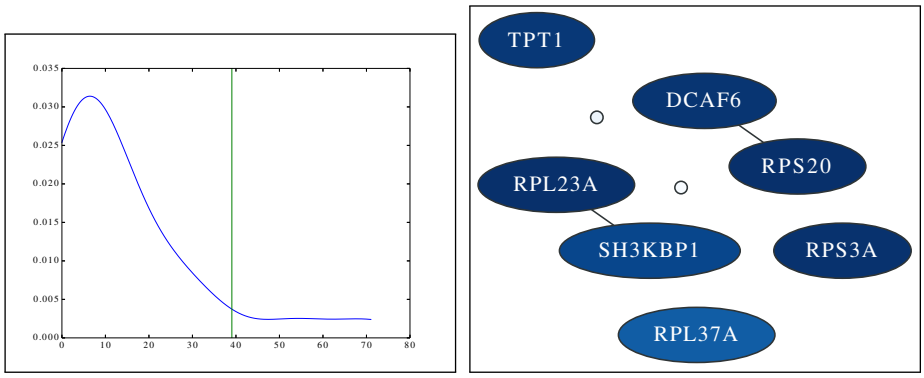
Figure 4.8: **Visualization of one model** A sample model for TCGA-LAML gene expression data **(a)** individual classifiers and their selected features; higher confidence of a node is shown by a darker color, **(b)** selected genes plotted over the PPI network; green and yellow show low and high confidence respectively, and the thickness of the border of the node shows the respective confidence of the individual classifier to which it belongs.

disjoint sparse models are fit to the data, their selected features might happen to be close to each other on the PPI network (e.g., node TPT1 and node EEF1A1 in Fig. 4.8(b)).

It is worth noting that these plots are the result of analyzing one single given test sample. Therefore in practice, these interpretations can be used for each patient and if useful, influence the treatment that the oncologist prescribe for the patient.

Visualization of Important Global Features: As explained in Section 4.2.1, a graph is created from model structures of all 100 random training partitions, and then it is pruned to keep only high confidence nodes and edges. The density estimation of the graph edge weights and the pruned graph are plotted in Fig. 4.9 where the nodes with labels are the ones that are not pruned. The nodes in this figure that do not have any label, are the ones with frequency lower than the corresponding threshold. Among the features considered to be important were features that had previously been linked to leukemia such as SH3KBP1 [1].

What was more intriguing to see was that four out of the seven important features of the TCGA-LAML gene expression dataset contained ribosomal proteins when using the risk group label, *i.e.* RPL37A, RPS20, RPS3A, and RPL23A. For a long time ribosomes were just considered machines that perform an unbiased translation of genes from mRNA to amino acid sequences, but this view has recently been challenged [158]. One new hypothesis is that the ribosome introduces an additional regulatory layer. Therefore, it could very well be that mutations in ribosomal proteins can lead to a misregulation of expression levels of impor-



(a) (b)

Figure 4.9: **(a) Determine pruning threshold** Threshold is determined by finding the point after which, 90% of the area under the curve is observed from left to right. The horizontal axis shows the observed frequency or weight of the edges. **(b) Important Global Features** High confidence nodes and edges of the graph generated from the model on TCGA-LAML gene expression data. Darker color represents higher rate of being selected by a classifier.

tant genes and ultimately to the development of cancer (in this case leukemia). One of the ribosomal proteins we found was RPL23A. It has been shown that loss of RPL23A can impede growth and lead to morphological abnormalities in Arabidopsis Thaliana [158]. Therefore, a mutation in RPL23A might also have severe effects in humans. A missense mutation in RPL23A was recently found in patients having Diamond-Blackfan anemia, which is an inherited form of pure red cell aplasia (related to leukemia) [56]. Note that the model for LAML has low performance for the regularization value chosen. Nevertheless, the features shown here are also the ones with the highest confidence for models learnt with less regularization (with several other additional features). The models with less regularization show similar performance to the other methods shown in Fig. 4.10

Performance comparison

The performance of the method was compared with that of two ensemble methods, AdaBoost and stochastic gradient boosting, as well as an SVM with linear kernel, and an SVM with an RBF kernel. We also included our implementation of the NICK method [81]. We randomly partitioned the data into training and test sets with 80% of the data for training and 20% of the data for testing. To compare the performance of the different methods, Area Under the receiver operating characteristic Curve (AUC) [40] was calculated on the test set over the decision values returned by the methods on the individual samples. The process was repeated 100 times to reduce random effects. As seen in Fig. 4.10, overall performances of all methods are comparable. In some cases a single SVM

works better, in some other cases ensemble algorithms give a better performance. However, in most cases an improvement in performance is observed by adding individual learners to the model, with the greatest gains due to the first few individual learners added to the model. In two cases, TCGA-LAML/Vital status and TCGA-LAML/Risk Group, our reported performance measures are significantly lower than other methods. This, however, comes from the fact that we have enforced extreme sparsity measures. The performance of the method increases and reaches the other methods' performance levels if this constraint is relaxed, as reported in supplementary 1. We enforced those sparsity measures for all models to avoid over-fitting. Optimizing the sparsity constraint via cross-validation would have been computationally expensive, which is why we preferred to be conservative. Had we optimized the sparsity constraint, we would have still been able to find the significant features while having similar performance as the other methods.

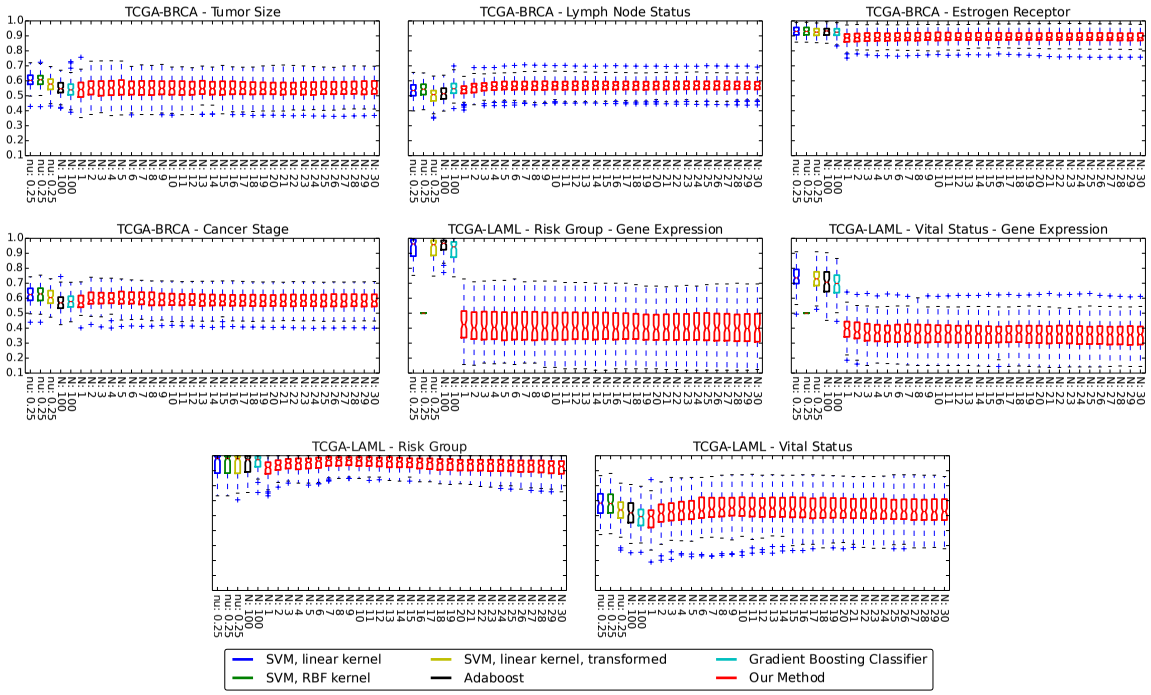


Figure 4.10: **Performance Summary (AUC)** Each box shows a 25–75% interval, as well as the median, which is shown as a horizontal line in each box.

4.2.3 Conclusions

Machine learning has become more and more popular in many real world scenarios for making sense of large collections of facts. Differences between the data used for training the method and new data for which the label should be predicted can limit the performance of prediction methods on those data. In this work we introduced a method that estimates these potential partial biases and incorporates them into the prediction function. We applied it to gene expression and DNA methylation measurements from cancer patients. Our method has state-of-the-art performance on many different prediction tasks. Furthermore, we show how to make sense of the predictions. Visualizing the important genes can lead to new biological insights, as shown for the TCGA-LAML dataset with the risk group label. Instead of mapping the genes to PPI networks, one could also think of mapping them to signaling pathways [73].

Recently, a study showed that most published signatures are not significantly more associated with cancer outcome than random signatures [148]. One of the reasons for this finding is that the data comes from slightly different underlying hidden data distributions. Since our new method estimates this bias and corrects for it by up-weighting the classifiers that have higher confidence, we expect that it should be less susceptible to such differences in the data.

In this work we designed and developed a method that besides being a predictive model, it can be used for two different purposes. It can be used as an exploratory method to reveal potential features used in future studies; and it can be used to different underlying causes of the same disease and with its interpretability help oncologists to choose the treatment accordingly.

We would like to point out that the applicability of our method is not limited to cancer outcome prediction, and it can apply to many more scenarios. The method assumes that the data has enough features to select from, and that there are related features to those selected ones that can be used to estimate their reliability. These are conditions that almost all biological data satisfy, hence the method can be applied to them.

The method also works as a skeleton whose components can be easily substituted. For example, by changing the classifier used in individual learners to a multi-class classifier, the method would work on multi-class problems. For the sake of simplicity and without loss of generality we performed the evaluations only on binary classification problems. Also, due to the structure of our model, one possible approach would be to use a method such as iRDA and use those gene sets as features of individual learners. Whether this approach leads to better results or not requires further research. Also, the combination of maximal information coefficient and Gaussian processes is not the only feasible option, and they can be replaced with other faster methods if the time complexity of the method is of any concern. Some of these alternatives are already available on the *github* repository of the method.

4.2.4 Enhancements and Parameter Selection

In order to withdraw the hassle of parameter setting from our method, we utilize a nested cross-validation scheme to automatically search for and find best parameters for a given dataset. A nested cross-validation scheme tests the method with different parameters several times, and therefore the method must be fast enough for the process to be feasible. For this purpose, we modify and enhance the method.

Before we fix the computational bottleneck, we need to review our method's modules and their parameters. On the highest level, we have the number of weak learners. We use nested cross-validation at this level, to select the best number of weak learners for the given dataset. The next parameter on a lower level, is the complexity of each weak learner and correspondingly their number of selected features. At this stage, we choose the best parameter again using another nested cross-validation. The main parameter on the lowest level is the complexity of our feature confidence estimators. This is where we have the most computationally intensive task, *i.e.* computing MICs.

The reason we compute all these MICs in the original RatBoost approach (see Fig. 4.7) is that Gaussian Processes in their normal setting, having as many features as we have, overfit to the data, and therefore a pre-selected small subset of features is what we feed to each Gaussian Process to prevent this overfitting. Although another way of reducing the complexity of a GP is to use a covariance function such as squared exponential with the automatic relevance determination (ARD) covariance function [118], the increased running time caused by the covariance function made it impractical for it to be used in our setting.

Now consider the following as a module: given a feature f_i, compute all relevant MICs, select features according to them, and feed them to a GP. We can replace it with the following: using a nested cross-validation scheme, find the best parameters of a fast method to predict f_i using other features. We use a C-SVM regression model with a linear kernel and the parameter C is set according to a nested cross-validation scheme. Please note that C-SVM is not a Bayesian method and therefore does not give a posterior distribution, which is indicated with a mean and a variance as the output. Therefore we need to also change the way the confidence is calculated for each feature from the one shown in Formula 4.5, to the following Formula:

$$c_f(X_s) := 2 \cdot F\left(-\left|\frac{(m_f(X_s) - X_{s,f}) - \mu_{err_{m_f}}}{\sigma_{err_{m_f}}}\right|\right) \tag{4.10}$$

where m_f is the C-SVM trained to predict f, and $\mu_{err_{m_f}}$ and $\sigma_{err_{m_f}}$ are the expected mean and standard deviation of the residual error of model m_f predicting f. The latter two values are estimated using a cross-validation on the training set.

We tested the above modified method on a Sequence-based Gene Expression (EXP-S) Lymphoma dataset[1] [121], separating Diffuse Large B-cell Lymphoma

[1] `https://icgc.org/icgc/cgp/64/345/53049`

(DLBCL) samples from Follicular Lymphoma (FL) samples. This dataset is hereafter referred to as ICGC-Lymphoma. Table 4.1 shows dataset's sample counts and Table 4.2 shows the resulting performance measures in terms of average precision recall score for the same dataset.

Diffuse large B-cell lymphoma	39
Follicular lymphoma	40
Total Count	79

Table 4.1: ICGC-Lymphoma in numbers.

Method Name	APRS (mean $\pm 2\times$ std)
RatBoost	0.953 ± 0.087
Gradient Boosting Classifier	0.944 ± 0.152
Adaboost	0.886 ± 0.21
SVM	0.867 ± 0.212

Table 4.2: Average Precision Recall Scores (APRS) for ICGC-Lymphoma dataset. Estimated confidence intervals come from a 50 stratified shuffle split with 80% and 20% of the dataset as train and test sets respectively.

The result of the above changes is available in the same repository on *github*[2].

As shown in Table 4.2, our method not only outperforms the other tested methods, but also shows a smaller variance in performance, indicating that it performs well on such a problem with a higher confidence. We also showed that our proposed changes improve the computational complexity of the method enough that it is feasible to run a nested cross-validation on all critical hyperparameters of our method. It is also worth noting that the changes do not interfere with the interpretability of the method and one can interpret the model the same way as explained in Section 4.2.

4.3 Raccoon

In Section 4.2 we developed the idea of an ensemble method, taking into account properties of a new given sample to assign confidences to each individual learner while calculating the final output of the model. In this section we take the idea further and train a model for each given new input. This means that unlike usual supervised methods, a part of training happens for every given new sample. In order to explain the method, we divide the steps into *prepare* and *fit-predict*.

During *prepare* phase, we only learn the relationships between the input features. This information is then used to identify features of a given sample which are more reliable than others. Then, given a new sample during the *fit-predict* phase, a model is trained on the training data, using only the identified

[2] `https://github.com/adrinjalali/Network-Classifier`

"reliable" features. At the end, the putput of that model is considered as the output of this method for the given input. We use code name *Raccoon* for this method.

4.3.1 Prepare

In this step we only learn the relationships between the features of the training data. Since it would be computationally infeasible to learn the relationships between all the features, we first select a pool of candidate features which have the potential of being used later in the *fit-predict* step. Once members of this candidate features pool are selected, we try to find a pattern between other features of the training data and each selected feature the same way we did in Section 4.2, *i.e.* we try to predict each chosen feature using all other features available in the data. Algorithm 1 depicts a pseudocode of this step.

Algorithm 1 Prepare

function PREPARE(X, y)
  $\Phi \leftarrow$ FEATURES_OF(X)
  $\Phi_l \leftarrow$ SELECT_CANDIDATES$_{k_1}(X, y)$
  $\mathcal{M} \leftarrow \{\}$
  for all $f \in \Phi_l$ **do**
    $\Phi_f \leftarrow$ TOP$_{k_2}(\{(f', \text{MIC}(f, f') | f' \in \Phi \setminus f\})$
    $g_f \leftarrow$ GP(X_{Φ_f}, f)
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{g_f\}$
  end for
  return $\mathcal{M}, \Phi_l$
end function

In the above pseudocode, the "SELECT_CANDIDATES" function returns a subset of all given features which are at least remotely related to the outcome y. It is basically a loose feature selection step which is free to choose also less than perfect features. We include a larger than usual feature set for two reasons:

1. The data is noisy and therefore some of the selected features are selected because they show a relationship with the outcome, whereas in reality they are not related.

2. Only a subset of these features are used for the prediction given a new sample, hence it is not critical to select a minimal set of "best" features.

Our implementation uses maximal information coefficient (MIC) of each feature with y to sort them, and take the best k_1 (hyperparameter) features forming the set Φ_l. Then for each feature f in Φ_l, a Gaussian Process model is trained to predict f using other features, similar to what we did in Section 4.2. Note that not all features are used to predict each selected feature. Similar to RatBoost, MIC is used here to select only k_2 (hyperparameter) best features as the input to each Gaussian Process. This to prevent the GPs from overfitting to the data.

4.3.2 Fit-Predict

This step happens for each given new test sample. For each given sample x, a confidence level is calculated for each $f \in \Phi_l$ according to Formula 4.5, and then a few features with the highest confidence are selected for the next step. The number of selected features is another hyperparameter. Φ_x being the candidate features selected for this particular test sample x, a model is trained on the original training data (X, y), using only features in Φ_x. The output of this model given the input x is our prediction for x. The type of model used here can be set by the user, and the user controls the hyperparameters of the given model. This model can of course run a grid search on its hyperparameters given X_{Φ_x}, looking for the best parameter set for a given set of features. Algorithm 2 shows a pseudocode of this step.

Algorithm 2 Fit-Predict

```
1: function FIT_PREDICT(X, y, x, Φ_l, M, M)
2:     Φ_x ← TOP_{k_3}({(f, CONFIDENCE(M_f, x, f) )|f ∈ Φ_l})
3:     M_x ← FIT(M, X_{Φ_x}, y)
4:     return PREDICT(M_x, x_{Φ_x})
5: end function
```

Line 2 in Algorithm 2 uses the Formula 4.5 to calculate feature confidences using the models which were prepared in Algorithm 1 and select the best k_3 (hyperparameter) features. Line 3 trains the model M on the given data $(X_{\Phi_x}, y$, using only the selected features Φ_x. As an example, M can be a C-SVM classifier which automatically searches for the best C using a grid search on (X_{Φ_x}, y). And at last, the final prediction is calculated in line 4 using only the selected features of the given sample x.

4.3.3 Discussion and Future Work

This method puts a heavy focus on selecting features specific to a given input x, and trains a separate model for each input. This means training these models should be preferably fast, otherwise it takes a very long time for the method to give an output for all samples in a dataset. Further analysis and experiments are required to understand the conditions under which, this method performs better than traditional models.

There are some potential paths which can be taken for further experiments. As explained above, the purpose of the *prepare* phase is to learn the relationships between the input features. Amid high computational costs, we choose a limited number of features for this step using a given dataset. However, once the candidate features are selected, the patterns predictive of those features using the rest of the input could potentially be learned using other datasets. For instance, if our input features are average methylation levels on and around each gene, we could try predicting these values using other datasets, including healthy tissue

datasets, as long as the datasets are conceptually compatible, *e.g.* they are taken from the same tissue type.

A minor modification worth exploring is to replace the $MIC \rightarrow GP$ pipeline (lines 6-7 Algorithm 1) with a C-SVM the same way it was done in Section 4.2.4, reducing the computational complexity of the method and enabling nested cross-validation schemes for easier parameter search.

Another computationally intensive operation in our method is MIC, which is used in lines 3 and 6 of Algorithm 1 for the purpose of selecting features. This operation can be replaced by faster schemes such as an $l_1 - SVM$ with a linear kernel, a LASSO model, or the randomized dependence coefficient (RDC) [84].

Interpreting *Raccoon* for each given sample x can be done in a similar fashion to *RatBoost*, and in two different levels. First we need to check which features are selected in Φ_x, and then see how those features are used by the model M_x. A combination of the confidences calculated for each $f \in \Phi_x$, and the importance of those features reported by M_x would give an idea of why the given sample is classified to the class it has. Please note that the interpretability of the method depends to some extend on the interpretability of the model M, which is a factor to be considered when choosing M.

We hope the fact that the above method is very flexible, adaptive, and interpretable proves it a step further towards personalized medicine while at the same time helping cell biologists study cancer with better clues derived from the data. An implementation of the method is available on *github*[3].

4.4 Conclusion

In this chapter we identified some of the challenges related to analyzing cancer data such as noise, batch effects, and cancer (tumor) heterogeneity. We then showed how we exploit the fact that input features are dependent random variables to gain insight about each given new sample.

The methods we introduced are adaptive and interpretable methods which not only have the potential of counteracting batch effects and noise, but are also a step further towards truly personalized medicine. The latter is a result of interpretability of our methods for each given sample, *i.e.* a different interpretation of the model per patient.

[3] `https://github.com/adrinjalali/Network-Classifier/tree/master/Raccoon`

5 Conclusion

In chapter 3 we covered a flow cytometry data analysis pipeline and its application on two types of non-Hodgkin lymphoma, *i.e.* Follicular Lymphoma and Diffuse Large B Cell Lymphoma. We showed how the pipeline can extract novel insights from the data as well as effectively automate a laborious workflow. We showed how flowType can extract features which can be used to train a model to classify subtypes. However, we did not consider the relationships between those features, *i.e.* cell populations, while training the model. One approach would be to design a kernel which takes into account those relationships and therefore implicitly reduces the dimensionality of the input data.

In chapter 4 we focused mostly on the analysis of DNA methylation data using adaptive and interpretable models. Although we put a heavy focus on the models, our experiments showed that the preprocessing step can also play a crucial role in the stability and performance of those models. In the case of DNA methylation data, for instance, a step to aggregate methylation levels over genes made the models more stable, faster, and better performing. Our observations support the need to put more focus on the preprocessing steps, and to document them in a more informative way. This would also greatly help towards improved reproducibility of publications in the field.

5.1 Future Work

This thesis is an effort towards a better understanding of cancer, as well as an improved diagnostic process. However, there are a few aspects which need to be done before these methods can best be employed in clinics:

- *Incorporating multiple data sources*: The approaches we took in this thesis

all incorporate a single input type at a time. Of course as mentioned in previous chapters, not all data sources are available for all patients or at the time of diagnosis. However, the same way that a pathologist would use different test results as accumulating evidence for a potential diagnosis or to support a treatment, a model should also be able to do the same and gain or loose confidence in a specific diagnosis as more data comes in for a single patient.

- *Adaptive to missing data sources*: Incorporating increasing evidence also means that models should be able to cope with missing data and missing data sources. Our models in Chapter 4 are efforts towards handling noisy or missing data in a single data source, but similar approaches can be taken to have an ensemble of models each working on a specific data source.

- *Models confidences*: From a pure mathematical and machine learning perspective, we sometimes tend to focus too much on a classifier's performance and forget about the value of reporting and estimating a model's confidence in the output for a specific input. In a sense, we put our confidence into a model based on its overall performance/confidence on a dataset, whereas in practice, it is crucial to know how confident a model is on its output for a given input. Bayesian models are useful for this purpose in the sense that the posterior probability's variance (for a Gaussian posterior for instance) can easily be interpreted as how confident the model is. However, it is also possible to calculate a proxy for model's confidence for non-Bayesian models as shown in Chapter 4. The analogy from the real world is that when you go to a doctor, independent of how "good" an individual doctor is, you'd always appreciate if they would tell you that they are not confident in their diagnosis, and that you should probably get a second opinion. In exactly the same way, if a model's confidence for a specific sample is not high enough, we can always rely more on a human doctor, or another model, or a model based on a different type of data.

- *Interpretability, reasoning*: Similar to people preferring a doctor who can tell them the reason behind a diagnosis, a pathologist can put more trust in a model which can report why a certain output is asserted by the model. Especially with all the noise, batch effects, small sample issues, etc., it is important that a doctor can validate a model's reasoning, and decide whether it is a valid conclusion or an artifact of one the abovementioned issues.

- *Real world issues such as batches, noise, etc.*: One of the reasons that a model's confidence and interpretability are important, is that the data in the real world is noisy and has batch effects, in contrast to the data cleaned and trimmed to include only nice and clean data, and the interpretability is a way to detect if a model may be malfunctioning due to those effects. However, those effects are real and happen too frequently for a bioinformatician to ignore. A model would ideally be able to automatically handle those effects seen in the data.

- *Reproduce, deploy, test*: Instead of looking at each publication which tries to deliver a better or different classifier for cancer diagnosis as an isolated effort, we can look at the communal work as an effort to improve and diversify the set of pieces of software we have, all of which can differentiate between a given set of cancer types. With this perspective, we can apply some of the concepts which are usually applied to software, in particular continuous integration, *i.e.* to continuously build, deploy, and test new models. This would also make the works more "reproducible", which is essential in an era where we have a reproducibility crisis in medical science. Having a place where these models are deployed would also help a larger community to test them in real life and see how they perform and for the developers of these models to get feedback and continuously improve them. It would also be easier to observe and work on some of the real life issues such as batch effects or differences between labs when the models are exposed and tested by a wider community.
- *Counsel of doctors* $\rightarrow$ *counsel of models*: One main benefit of having models developed by different groups is that we can also use a set of them for a given problem/data, the same way that we would sometimes desire a group of doctors to work on a patient's case instead of an individual doctor. One immediate use-case for such a system is to trigger/request a second opinion if the wisdom of the set disagrees with the diagnosis made by an individual doctor.

5.2 Closing Remarks

Finally, I would take the fact that many companies these days offer machine learning powered cancer diagnostics services as an evidence for how far we have moved forward in our understanding of cancer and our abilities to automatically diagnose patients given certain data. However neither adaptive, nor interpretable models are prevalent in the community and both doctors as well as the patients would better take advantage of and trust the models if they could see how they work and why they do what they do. This thesis is a contribution towards this goal and I hope it helps the community to build on top of or use some of the work done here.

A

RchyOptimyx Appendix

Table A.1: The phenotypes with a high overlap with the BCR(pBLNK)$^+$ compartment as identified by flowType. The table includes the cell proportion of these immunophenotypes (second column) and the differences in the cell proportion of BCR(pBLNK)$^+$ cells in the stimulated and unstimulated assays (third column).

Phenotype Name	Cell Proportion	$BCR^+_{(stim-unstim)}$
CD19+CD4-CD8-CD34+CD20+CD123+CD38-CD3-	0.001	0.160
CD19+CD4-CD34+CD20+CD123+CD38-CD3-	0.001	0.160
CD19+CD4-CD34+CD20+CD123+CD3-	0.001	0.155

Table A.2: The phenotypes with a high overlap with the IL7(pSTAT5)$^+$ compartment as identified by flowType. The table includes the cell proportion of these immunophenotypes (second column) and differences in the cell proportion of IL7(pSTAT5)$^+$ cells in the stimulated and unstimulated assays (third column).

Phenotype Name	Cell Proportion	$IL7^+_{(stim-unstim)}$
CD19-CD4+CD8+CD20+CD33+CD38-CD3+	0.008	0.364
CD19-CD4+CD8+CD20+CD33+CD3+	0.008	0.366
CD19-CD4+CD8+CD34+CD33+CD38-CD3+	0.008	0.366
CD19-CD4+CD8+CD34+CD33+CD3+	0.008	0.368
CD19-CD4+CD8+CD34+CD20+CD33+CD38-CD3+	0.006	0.399
CD19-CD4+CD8+CD34+CD20+CD33+CD3+	0.006	0.402
CD4+CD8+CD20+CD33+CD38-CD3+	0.011	0.365
CD4+CD8+CD20+CD33+CD3+	0.011	0.371
CD4+CD8+CD34+CD33+CD38-CD3+	0.011	0.366
CD4+CD8+CD34+CD33+CD3+	0.011	0.371
CD4+CD8+CD34+CD20+CD33+CD38-CD3+	0.008	0.399
CD4+CD8+CD34+CD20+CD33+CD3+	0.009	0.405
CD19+CD4+CD8+CD20+CD33+CD38-CD3+	0.003	0.364
CD19+CD4+CD8+CD20+CD33+CD3+	0.003	0.378
CD19+CD4+CD8+CD34+CD33+CD38-CD3+	0.003	0.359
CD19+CD4+CD8+CD34+CD33+CD3+	0.003	0.372
CD19+CD4+CD8+CD34+CD20+CD33+CD38-CD3+	0.002	0.397
CD19+CD4+CD8+CD34+CD20+CD33+CD3+	0.002	0.409

Table A.3: The phenotypes with a high overlap with the LPS(p-p38)$^+$ compartment as identified by flowType. The table includes the cell proportion of these immunophenotypes (second column) and differences in the cell proportion of LPS(p-p38)$^+$ cells in the stimulated and unstimulated assays (third column).

Phenotype Name	Cell Proportion	$LPS^+_{(stim-unstim)}$
CD19-CD4-CD8-CD34-CD20-CD33+CD123-CD38-CD3-	0.008	0.474
CD19-CD4-CD8-CD34-CD20-CD33+CD123-CD3-	0.008	0.473
CD19-CD4-CD8-CD34-CD20-CD33+CD38-CD3-	0.009	0.466
CD19-CD4-CD8-CD34-CD20-CD33+CD3-	0.009	0.465
CD19-CD4-CD8-CD34-CD33+CD123-CD38-CD3-	0.022	0.460
CD19-CD4-CD8-CD34-CD33+CD123-CD3-	0.022	0.459
CD19-CD4-CD8-CD34-CD33+CD38-CD3-	0.022	0.452
CD19-CD4-CD8-CD34-CD33+CD3-	0.022	0.451
CD19-CD4-CD8-CD34-CD20+CD33+CD123-CD38-CD3-	0.013	0.450
CD19-CD4-CD8-CD34-CD20+CD33+CD123-CD3-	0.013	0.449
CD19-CD4-CD8-CD20-CD33+CD123-CD38-CD3-	0.023	0.453
CD19-CD4-CD8-CD20-CD33+CD123-CD3-	0.023	0.452
CD19-CD4-CD34-CD20-CD33+CD123-CD38-CD3-	0.011	0.456
CD19-CD4-CD34-CD20-CD33+CD123-CD3-	0.011	0.455
CD19-CD8-CD34-CD20-CD33+CD123-CD38-CD3-	0.012	0.462
CD19-CD8-CD34-CD20-CD33+CD123-CD3-	0.012	0.461
CD19-CD8-CD34-CD20-CD33+CD38-CD3-	0.012	0.454
CD19-CD8-CD34-CD20-CD33+CD3-	0.012	0.454
CD4-CD8-CD34-CD20-CD33+CD123-CD38-CD3-	0.011	0.462
CD4-CD8-CD34-CD20-CD33+CD123-CD3-	0.011	0.461
CD4-CD8-CD34-CD20-CD33+CD38-CD3-	0.011	0.454
CD4-CD8-CD34-CD20-CD33+CD3-	0.011	0.454
CD8-CD34-CD20-CD33+CD123-CD38-CD3-	0.015	0.450
CD8-CD34-CD20-CD33+CD123-CD3-	0.015	0.449

Table A.4: Statistically significant immunophenotypic correlates of survival of HIV$^+$ subjects are predicted by flowType. The p-values of the log rank tests, 95% confidence intervals calculated using bootstrapping, adjusted p-values using Bonferroni's method, coefficients and R^2 values of the Cox proportional hazards regression models, and the frequency of the cells are provided as columns of the table.

#	Phenotype	p-value	p-value, CI	adjusted p-value	CPHR Coefficient	R^2	Cell Frequency
1	CD28-CD45RO+CD57-CCR5+	5.3e-07	(4.3e-14, 1.3e-02)	2e-02	20.5	0.056	0.03048
2	CD28-CD8+CD57-CD127-	2.5e-07	(2.3e-14, 3.8e-04)	1e-02	12.3	0.060	0.05975
3	CD28-CD45RO+CD57-CCR7-	5.1e-07	(2.3e-14, 6.1e-04)	2e-02	15.7	0.057	0.03829
4	CD28-CD45RO+CD4-CD57-	3.5e-07	(2.3e-14, 1.1e-03)	1e-02	13.2	0.058	0.04357
5	CD45RO+CD4-CD57-CD127-	2.7e-07	(1.2e-13, 7.1e-03)	1e-02	12.8	0.059	0.05062
6	CD28-CD45RO+CD57-CD127-	4.7e-08	(1.7e-14, 6.8e-04)	2e-03	16.0	0.067	0.03732
7	CD45RO+CD4-CD27-CD127-	4.4e-07	(5.8e-14, 1.1e-03)	2e-02	14.3	0.057	0.04830
8	CD28-CD45RO+CD57-	5.6e-07	(4.4e-14, 4.1e-04)	2e-02	12.4	0.056	0.05015
9	CD45RO+CD4-CD127-	6.5e-07	(4.7e-15, 2.9e-03)	2e-02	9.6	0.056	0.07176
10	CD28-CD45RO+CD4-CD127-	3.1e-07	(0.0e+00, 5.7e-03)	1e-02	11.7	0.059	0.05300
11	CD28-CD45RO+CD57-CCR5+CD27-CCR7+CD127-	4.7e-07	(5.7e-14, 7.7e-03)	2e-02	171.4	0.057	0.00315
12	CD28-CD45RO+CD4-CD57-CCR5+CD27-CCR7+CD127-	4.5e-07	(1.8e-13, 3.9e-04)	2e-02	176.2	0.057	0.00294
13	CD28-CD57-CD127-	3.3e-07	(3.4e-15, 8.0e-03)	1e-02	8.0	0.058	0.12341
14	CD28-CD4-CD57-	8.8e-07	(2.2e-15, 2.9e-03)	3e-02	7.2	0.054	0.15525
15	CD57-CD27-CD127-	6.2e-08	(2.4e-14, 4.7e-03)	2e-03	9.5	0.065	0.12173
16	CD4-CD57-CD27-CD127-	4.7e-08	(4.2e-14, 3.3e-03)	2e-03	9.7	0.067	0.09721
17	CD28-CD57-CCR7-CD127-	2.8e-07	(9.7e-15, 1.0e-02)	1e-02	9.8	0.059	0.08417
18	CD28-CD4-CD57-CD127-	3.3e-08	(2.0e-12, 5.7e-04)	1e-03	9.1	0.068	0.10852
19	CD4-CD57-CCR7-CD127-	6.5e-07	(3.8e-15, 2.3e-03)	2e-02	8.8	0.056	0.09501
20	CD45RO-CD4-CD57+CCR5-CD27+CCR7-CD127-	6.1e-07	(1.2e-12, 2.6e-03)	2e-02	498.4	0.056	0.00097
21	CD28-CD45RO-CD4-CD57+CCR5-CD27+CCR7-CD127-	2.5e-07	(0.0e+00, 7.7e-03)	1e-02	561.2	0.060	0.00074
22	CD45RO-CD8+CD57+CCR5-CD27+CCR7-CD127-	1.2e-07	(4.6e-14, 3.3e-04)	5e-03	638.6	0.063	0.00068
23	CD45RO-CD8+CD4-CD57+CCR5-CD27+CCR7-CD127-	1.2e-07	(5.1e-14, 2.0e-03)	5e-03	638.6	0.063	0.00068
24	CD28-CD45RO-CD4-CD57+CCR5-CD27+CD127-	5.7e-07	(1.1e-13, 2.3e-03)	2e-02	298.3	0.056	0.00099
25	KI-67+CD28-CCR5+	1.0e-11	(2.9e-13, 2.8e-03)	4e-07	96.1	0.101	0.00547
26	KI-67+CD28-CCR5+CD27-	8.7e-12	(1.5e-14, 8.9e-04)	3e-07	115.3	0.102	0.00453
27	KI-67+CCR5+	1.3e-11	(2.4e-14, 7.0e-03)	5e-07	53.4	0.100	0.01192
28	KI-67+CD28+CD45RO+CD57-CCR7-CD127-	4.2e-09	(5.6e-16, 3.0e-03)	2e-04	241.3	0.077	0.00209
29	KI-67+CD45RO-CD4-CD27-CCR7-CD127-	1.2e-09	(2.0e-14, 4.4e-03)	4e-05	161.9	0.082	0.00297
30	KI-67+CD28-CD45RO-CD8-CD4-	5.0e-09	(2.9e-12, 1.7e-03)	2e-04	176.0	0.076	0.00225
31	KI-67+CD8-CD4-	8.1e-09	(6.1e-13, 4.5e-02)	3e-04	58.1	0.074	0.00738
32	KI-67+CCR5+CD27-CCR7-	2.0e-11	(3.8e-14, 6.0e-04)	8e-07	109.8	0.099	0.00532
33	KI-67+CD8-CCR5+CCR7-	1.3e-10	(3.1e-13, 2.0e-03)	5e-06	147.3	0.091	0.00392
34	KI-67+CD28-CD8-CCR5+CCR7+CD127-	2.6e-09	(1.6e-14, 1.1e-02)	1e-04	625.8	0.079	0.00061
35	KI-67+CD28+CD45RO+CD8+CD57-CD27+CCR7+	6.7e-07	(3.8e-13, 1.5e-03)	3e-02	585.4	0.055	0.00051
36	KI-67+CD28+CD45RO+CD8+CD4-CD57-CD27+CCR7+	6.7e-07	(1.1e-16, 4.7e-03)	3e-02	585.4	0.055	0.00051
37	KI-67+CD8+CD27-CCR7-CD127-	4.7e-11	(1.3e-13, 1.4e-03)	2e-06	141.3	0.095	0.00292
38	KI-67+CD8+CD4-CD27-CCR7-CD127-	4.7e-11	(1.3e-13, 1.3e-03)	2e-06	141.3	0.095	0.00292
39	KI-67+CD28-CD8+CD27-CCR7-CD127-	2.7e-11	(1.0e-13, 7.6e-04)	1e-06	164.5	0.097	0.00241
40	KI-67+CD28-CD8+CD4-CD27-CCR7-CD127-	2.7e-11	(2.7e-13, 1.4e-03)	1e-06	164.5	0.097	0.00241
41	KI-67+CD28-CD8+CCR7-CD127-	6.6e-11	(5.6e-14, 1.5e-02)	3e-06	132.9	0.094	0.00293
42	KI-67+CD28-CD8+CD4-CCR7-CD127-	6.6e-11	(1.2e-14, 8.4e-04)	3e-06	132.9	0.094	0.00293

Table A.4: Statistically significant immunophenotypic correlates of survival of HIV$^+$ subjects are predicted by flowType. The p-values of the log rank tests, 95% confidence intervals calculated using bootstrapping, adjusted p-values using Bonferroni's method, coefficients and R^2 values of the Cox proportional hazards regression models, and the frequency of the cells are provided as columns of the table.

#	Phenotype	p-value	p-value, CI	adjusted p-value	CPHR Coefficient	R^2	Cell Frequency
43	KI-67+CD45RO+CD8+CD27-CCR7-	1.2e-09	(4.0e-12, 2.8e-03)	5e-05	143.6	0.082	0.00216
44	KI-67+CD45RO+CD8+CD4-CD27-CCR7-	1.2e-09	(1.0e-12, 1.2e-02)	5e-05	143.6	0.082	0.00216
45	KI-67+CD28-CD45RO+CD8+CD27-CCR7-	1.0e-09	(1.9e-15, 7.3e-04)	4e-05	188.5	0.082	0.00155
46	KI-67+CD28-CD45RO+CD8+CD4-CD27-CCR7-	1.0e-09	(1.7e-13, 2.0e-03)	4e-05	188.5	0.082	0.00155
47	KI-67+CD45RO+CD8+CD27-CD127-	7.1e-10	(1.2e-14, 6.8e-03)	3e-05	152.4	0.084	0.00221
48	KI-67+CD45RO+CD8+CD4-CD27-CD127-	7.1e-10	(3.4e-14, 1.5e-03)	3e-05	152.4	0.084	0.00221
49	KI-67+CD28-CD45RO+CD8+CD27-CD127-	5.0e-10	(6.0e-13, 3.1e-03)	2e-05	201.3	0.085	0.00163
50	KI-67+CD28-CD45RO+CD8+CD4-CD27-CD127-	5.0e-10	(4.6e-14, 2.7e-03)	2e-05	201.3	0.085	0.00163
51	KI-67+CD28-CD45RO+CD8+CD127-	1.0e-09	(1.2e-15, 3.2e-03)	4e-05	150.5	0.083	0.00222
52	KI-67+CD28-CD45RO+CD8+CD4-CD127-	1.0e-09	(1.5e-11, 3.6e-03)	4e-05	150.5	0.083	0.00222
53	KI-67+CD45RO+CD8+CD4-CD127-	2.2e-09	(2.8e-13, 2.1e-03)	9e-05	99.8	0.079	0.00362
54	KI-67+CD28-CD45RO+CD8+CD4-CCR7-	8.0e-09	(2.7e-12, 7.2e-04)	3e-04	133.6	0.074	0.00209
55	KI-67+CD28-CD45RO+CD57-CCR7+CD127-	5.9e-08	(4.0e-15, 4.5e-03)	2e-03	376.6	0.066	0.00075
56	KI-67+CD28-CD45RO+CD4-CD57-CCR7+CD127-	5.0e-08	(4.8e-13, 3.9e-03)	2e-03	409.6	0.066	0.00070
57	KI-67+CD57-CD27-CD127-	5.9e-10	(3.2e-14, 2.7e-03)	2e-05	44.9	0.085	0.00806
58	KI-67+CD28-CD27-CD127-	4.8e-10	(7.3e-15, 2.5e-03)	2e-05	50.6	0.086	0.00711
59	KI-67+CD4-CD127-	1.3e-10	(4.4e-16, 9.7e-03)	5e-06	37.1	0.091	0.01159
60	KI-67+CD28-CD127-	4.9e-10	(1.1e-12, 1.4e-03)	2e-05	41.4	0.086	0.00823
61	KI-67+CD4-CD27-	5.6e-09	(2.1e-14, 2.6e-03)	2e-04	28.6	0.075	0.01122
62	KI-67+CD28-CD4-CD27-	1.8e-09	(3.6e-13, 5.3e-03)	7e-05	40.2	0.080	0.00785
63	KI-67+CD27-CD127-	1.3e-09	(9.8e-15, 1.1e-03)	5e-05	33.0	0.082	0.01052
64	KI-67+CCR7-CD127-	6.5e-11	(1.4e-15, 9.6e-04)	2e-06	47.3	0.094	0.00947
65	KI-67+CD4-CD27-CCR7-	9.6e-11	(1.1e-16, 1.5e-03)	4e-06	52.1	0.092	0.00764
66	KI-67+CD4-CCR7-	1.7e-10	(3.0e-14, 1.0e-02)	7e-06	41.4	0.090	0.00987
67	KI-67+CD45RO+CD57-CCR7-	1.4e-09	(6.6e-13, 1.2e-03)	5e-05	49.6	0.081	0.00695
68	KI-67+CD45RO+CD57-CD27-CCR7-	9.1e-10	(8.6e-12, 2.5e-03)	3e-05	66.4	0.083	0.00505
69	KI-67+CD45RO+CD4-	2.0e-09	(8.0e-13, 2.5e-03)	8e-05	45.3	0.080	0.00851
70	KI-67+CD28-CD45RO+	1.3e-08	(1.2e-12, 2.4e-03)	5e-04	54.9	0.072	0.00525
71	KI-67+CD45RO+CD127-	1.1e-09	(4.4e-16, 1.5e-02)	4e-05	42.5	0.082	0.00834
72	KI-67+CD45RO+CD57-CD127-	2.9e-10	(1.5e-14, 6.4e-04)	1e-05	55.0	0.088	0.00719
73	KI-67+CD28-CD45RO+CD8+CD27-	9.2e-09	(2.6e-15, 2.3e-03)	4e-04	138.0	0.073	0.00201
74	KI-67+CD28-CD45RO+CD8+CD4-CD27-	9.2e-09	(1.0e-15, 4.6e-03)	4e-04	138.0	0.073	0.00201
75	KI-67+CD8+CD4-CD57-CD27-CD127-	1.9e-09	(5.9e-14, 7.0e-03)	7e-05	113.8	0.080	0.00274
76	KI-67+CD28-CD45RO+CD8+	9.3e-09	(5.9e-13, 1.4e-03)	4e-04	102.7	0.073	0.00279
77	KI-67+CD28-CD45RO+CD8+CD4-	9.3e-09	(0.0e+00, 1.6e-03)	4e-04	102.7	0.073	0.00279
78	KI-67+CD45RO+CD8+	2.1e-08	(6.9e-15, 6.8e-04)	8e-04	59.1	0.070	0.00512
79	KI-67+CD8+CCR7-	3.0e-08	(7.7e-13, 2.8e-03)	1e-03	49.5	0.068	0.00530
80	KI-67+CD8+CD27-CCR7-	8.3e-09	(1.0e-13, 3.6e-03)	3e-04	70.7	0.074	0.00377
81	KI-67+CD4-	2.8e-08	(1.0e-13, 2.3e-03)	1e-03	17.1	0.069	0.01627
82	KI-67+CD28-CD4-	1.1e-08	(5.9e-14, 4.0e-03)	4e-04	26.7	0.073	0.00950
83	KI-67+CD127-	2.7e-08	(1.2e-12, 2.1e-03)	1e-03	19.1	0.069	0.01460
84	KI-67+CCR7-	8.4e-08	(3.4e-15, 2.3e-03)	3e-03	18.3	0.064	0.01311

Table A.4: Statistically significant immunophenotypic correlates of survival of HIV$^+$ subjects are predicted by flowType. The p-values of the log rank tests, 95% confidence intervals calculated using bootstrapping, adjusted p-values using Bonferroni's method, coefficients and R^2 values of the Cox proportional hazards regression models, and the frequency of the cells are provided as columns of the table.

#	Phenotype	p-value	p-value, CI	adjusted p-value	CPHR Coefficient	R^2	Cell Frequency
85	KI-67+CD27-CCR7-	3.5e-08	(1.7e-13, 1.2e-03)	1e-03	25.2	0.068	0.00998
86	KI-67+CD45RO+CD27-	7.5e-07	(5.4e-13, 1.8e-03)	3e-02	24.0	0.055	0.00862
87	KI-67+CD45RO+CD57-	1.2e-07	(2.1e-13, 3.1e-03)	5e-03	22.9	0.062	0.01123
88	KI-67+CD4-CD57-	1.3e-08	(3.8e-15, 2.1e-03)	5e-04	25.3	0.072	0.01209
89	KI-67+CD28-CD4-CD57-	9.7e-09	(5.5e-12, 1.2e-03)	4e-04	37.7	0.073	0.00698
90	KI-67+CD57-CD127-	3.3e-09	(1.3e-13, 3.3e-03)	1e-04	28.1	0.078	0.01128
91	KI-67+CD45RO+CCR7-	4.2e-09	(7.8e-15, 2.5e-03)	2e-04	37.5	0.077	0.00819
92	KI-67+CD57-CCR7-	2.7e-08	(2.8e-13, 2.8e-03)	1e-03	26.6	0.069	0.01008
93	KI-67+CD57-CD27-CCR7-	1.2e-08	(4.9e-13, 2.6e-03)	5e-04	36.8	0.072	0.00762
94	KI-67+CD28-CCR7-	3.3e-09	(4.6e-14, 5.7e-03)	1e-04	37.7	0.078	0.00739
95	KI-67+CD28-CD27-CCR7-	3.3e-09	(2.6e-14, 6.5e-04)	1e-04	43.0	0.078	0.00647
96	KI-67+CD28-	1.9e-07	(4.0e-15, 2.7e-03)	7e-03	18.3	0.061	0.01053
97	KI-67+CD28-CD27-	7.1e-08	(1.5e-12, 8.6e-04)	3e-03	26.3	0.065	0.00874
98	KI-67+CD28-CD8-	8.3e-08	(5.5e-14, 2.5e-03)	3e-03	44.2	0.064	0.00523
99	KI-67+CD45RO+	8.9e-07	(1.9e-13, 2.5e-03)	3e-02	15.4	0.054	0.01343
100	KI-67+CD8+CD57-	1.1e-06	(4.4e-14, 3.1e-03)	4e-02	28.3	0.053	0.00648
101	KI-67+CD8+CD27-	6.4e-07	(2.3e-14, 1.1e-02)	2e-02	35.2	0.056	0.00560

List of Tables

List of Figures

Bibliography

[1] J. Adélaïde, V. Gelsi-Boyer, J. Rocquain, N. Carbuccia, D. J. Birnbaum, P. Finetti, F. Bertucci, M. J. Mozziconacci, N. Vey, D. Birnbaum, and M. Chaffanet. Gain of CBL-interacting protein, a possible alternative to CBL mutations in myeloid malignancies. *Leukemia*, 24(8):1539–41, Aug. 2010.

[2] N. Aghaeepour, P. K. Chattopadhyay, A. Ganesan, K. O'Neill, H. Zare, A. Jalali, H. H. Hoos, M. Roederer, and R. R. Brinkman. Early immunologic correlates of HIV protection can be identified from computational analysis of complex multivariate T-cell flow cytometry assays. *Bioinformatics*, 28(7):1009–1016, 2012.

[3] N. Aghaeepour, G. Finak, H. Hoos, T. R. Mosmann, R. Brinkman, R. Gottardo, R. H. Scheuermann, et al. Critical assessment of automated flow cytometry data analysis techniques. *Nature Methods*, 10(3):228–238, 2013.

[4] N. Aghaeepour, A. Jalali, K. O'Neill, P. K. Chattopadhyay, M. Roederer, H. H. Hoos, and R. R. Brinkman. RchyOptimyx: Cellular hierarchy optimization for flow cytometry. *Cytometry Part A*, 81(12):1022–30, 2012.

[5] N. Aghaeepour, R. Nikolic, H. Hoos, and R. Brinkman. Rapid cell population identification in flow cytometry data. *Cytometry Part A*, 79(1):6–13, 2011.

[6] D. Albanese, M. Filosi, R. Visintainer, S. Riccadonna, G. Jurman, and C. Furlanello. minerva and minepy: a c engine for the mine suite and its r, python and matlab wrappers. *Bioinformatics*, 29(3):407–408, 2013.

[7] B. Alberts, A. Johnson, J. Lewis, D. Morgan, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland Publishing, 2014.

[8] A. I. Baba and C. Câtoi. Tumor cell morphology. 2007.

[9] A.-L. Barabasi and Z. N. Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101–113, 2004.

[10] A. Bashashati, N. Johnson, A. Khodabakhshi, M. Whiteside, H. Zare, D. Scott, K. Lo, R. Gottardo, F. Brinkman, J. Connors, et al. B cells with high side scatter parameter by flow cytometry correlate with inferior survival in diffuse large b-cell lymphoma. *American Journal of Clinical Pathology*, 137(5):805–814, 2012.

[11] R. Bellman. On a routing problem. Technical report, DTIC Document, 1956.

[12] S. Bendall, G. Nolan, M. Roederer, and P. Chattopadhyay. A deep profiler's guide to cytometry. *Trends in Immunology*, 33(7):323–332, 2012.

[13] S. Bendall, E. Simonds, P. Qiu, E. Amir, P. Krutzik, R. Finck, R. Bruggner, R. Melamed, A. Trejo, O. Ornatsky, et al. Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum. *Science*, 332(6030):687–696, 2011.

[14] S. C. Bendall, G. P. Nolan, M. Roederer, and P. K. Chattopadhyay. A deep profiler's guide to cytometry. *Trends in Immunology*, 33(7):323–332, July 2012.

[15] S. L. Berger, T. Kouzarides, R. Shiekhattar, and A. Shilatifard. An operational definition of epigenetics. *Genes & development*, 23(7):781–783, 2009.

[16] A. Biancotto, P. Dagur, J. Chris Fuchs, M. Langweiler, and J. Philip McCoy Jr. OMIP-004: In-depth characterization of human T regulatory cells. *Cytometry Part A*, 81(1):360–361, 2011.

[17] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[18] C. M. Bishop and M. Svenskn. Bayesian hierarchical mixtures of experts. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 57–64. Morgan Kaufmann Publishers Inc., 2002.

[19] I. Bose and B. Ghosh. The p53-mdm2 network: from oscillations to apoptosis. *Journal of biosciences*, 32:991–997, 2007.

[20] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[21] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *ICML*, volume 98, pages 82–90, 1998.

[22] W. Cao. Molecular characterization of human plasmacytoid dendritic cells. *Journal of Clinical Immunology*, 29(3):257–264, 2009.

[23] K. Castro, J. Ward, L. Slutsker, J. Buehler, H. Jaffe, R. Berkelman, and J. Curran. Revised classification system for HIV infection and expanded surveillance case definition for AIDS among adolescents and adults. *MMWR Recomm Rep*, 41:1–19, 1992.

[24] C. Chan, F. Feng, J. Ottinger, D. Foster, M. West, and T. Kepler. Statistical mixture modeling for cell subtype identification in flow cytometry. *Cytometry Part A*, 73(8):693–701, 2008.

[25] C. Chan, L. Lin, J. Frelinger, V. Hérbert, D. Gagnon, C. Landry, R. Sékaly, J. Enzor, J. Staats, K. Weinhold, et al. Optimization of a highly standardized carboxyfluorescein succinimidyl ester flow cytometry panel and gating strategy design using discriminative information measure evaluation. *Cytometry Part A*, 77(12):1126–1136, 2010.

[26] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[27] P. Chattopadhyay and M. Roederer. Cytometry: Today's technology and tomorrow's horizons. *Methods*, 57(3):251–258, Feb 2012.

[28] P. Chattopadhyay, M. Roederer, and D. Price. OMIP-002: Phenotypic analysis of specific human CD8+ T-cells using peptide-MHC class I multimers for any of four epitopes. *Cytometry Part A*, 77(9):821–822, 2010.

[29] P. K. Chattopadhyay, C. M. Hogerkorp, and M. Roederer. A chromatic explosion: the development and future of multiparameter flow cytometry. *Immunology*, 125(4):441, 2008.

[30] H.-Y. Chuang, E. Lee, Y.-T. Liu, D. Lee, and T. Ideker. Network-based classification of breast cancer metastasis. *Molecular systems biology*, 3:140, Jan. 2007.

[31] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*, volume 6. MIT press Cambridge, 2001.

[32] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[33] E. Costa, C. Pedreira, S. Barrena, Q. Lecrevisse, J. Flores, S. Quijano, J. Almeida, M. del Carmen García-Macias, S. Bottcher, J. Van Dongen, et al. Automated pattern-guided principal component analysis vs expert-based immunophenotypic classification of b-cell chronic lymphoproliferative disorders: a step forward in the standardization of clinical immunophenotyping. *Leukemia*, 24(11):1927–1933, 2010.

[34] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

[35] F. Craig, R. Brinkman, E. Ten, and N. Aghaeepour. Computational analysis optimizes the flow cytometric evaluation for lymphoma. *Cytometry Part B - Clinical Cytometry*, Digital preprint, 2013.

[36] T. G. Dietterich. Ensemble learning. *The handbook of brain theory and neural networks*, pages 405–408, 2002.

[37] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[38] J. Dopazo. Functional interpretation of microarray experiments. *Omics: a journal of integrative biology*, 10(3):398–410, 2006.

[39] B. Efron and R. J. Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

[40] J. P. Egan. Signal detection theory and ROC analysis. *Academic Press, New York*, 1975.

[41] L. Ein-Dor, I. Kela, G. Getz, D. Givol, and E. Domany. Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics*, 21(2):171–8, Jan. 2005.

[42] M. Eller and J. Currier. OMIP-007: Phenotypic analysis of human natural killer cells. *Cytometry Part A*, 81(6):447–449, 2012.

[43] D. Eppstein. Finding the k shortest paths. *SIAM Journal on computing*, 28(2):652–673, 1998.

[44] M. Esteller. Epigenetics in cancer. *n Engl j Med*, 2008(358):1148–1159, 2008.

[45] G. Finak, A. Bashashati, R. Brinkman, and R. Gottardo. Merging mixture components for cell population identification in flow cytometry. *Advances in Bioinformatics*, v09, 2009.

[46] G. Finak, J.-M. Perez, A. Weng, and R. Gottardo. Optimizing transformations for automated, high throughput analysis of flow cytometry data. *BMC bioinformatics*, 11(1):546, 2010.

[47] R. W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

[48] K. Foulds, M. Donaldson, and M. Roederer. OMIP-005: Quality and phenotype of antigen-responsive rhesus macaque T cells. *Cytometry Part A*, 81(6):360–361, 2012.

[49] G. N. Frederickson. An optimal algorithm for selection in a min-heap. *Information and Computation*, 104(2):197–214, 1993.

[50] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[51] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. System Sci.*, 55(1):119–139, 1997.

[52] J. H. Friedman. Stochastic gradient boosting. *Comput. Stat. Data Anal.*, 38(4):367–378, 2002.

[53] M. J. Fulwyler. Electronic separation of biological cells by volume. *Science*, 150(3698):910–911, 1965.

[54] A. Ganesan, P. K. Chattopadhyay, T. M. Brodie, J. Qin, W. Gu, J. R. Mascola, N. L. Michael, D. A. Follmann, M. Roederer, C. Decker, T. Whitman, S. Tasker, A. Weintrob, G. Wortmann, M. Zapor, M. Landrum, V. Marconi, J. Okulicz, N. Crum-Cianflone, M. Bavaro, H. Chun, R. V. Barthel, A. Johnson, B. Agan, N. Aronson, W. Bradley, G. Gandits, L. Jagodzinski, R. O'Connell, C. Eggleston, and J. Powers. Immunologic and virologic events in early HIV infection predict subsequent rate of progression. *Journal of Infectious Diseases*, 201:272–284, Jan 2010.

[55] L. Gattinoni, E. Lugli, Y. Ji, Z. Pos, C. Paulos, M. Quigley, J. Almeida, E. Gostick, Z. Yu, C. Carpenito, et al. A human memory t cell subset with stem cell-like properties. *Nature Medicine*, pages 1290–1297, 2011.

[56] H. T. Gazda, M. Preti, M. R. Sheen, M.-F. O'Donohue, A. Vlachos, and S. M. Davies et al. Frameshift mutation in p53 regulator RPL26 is associated with multiple physical abnormalities and a specific pre-ribosomal RNA processing defect in diamond-blackfan anemia. *Human mutation*, 33(7):1037–44, July 2012.

[57] A. Goldhirsch, J. H. Glick, R. D. Gelber, A. S. Coates, and H.-J. Senn. Meeting highlights: international consensus panel on the treatment of primary breast cancer. *Journal of Clinical Oncology*, 19(18):3817–3827, 2001.

[58] S. Gordon, B. Cervasi, P. Odorizzi, R. Silverman, F. Aberra, G. Ginsberg, J. Estes, M. Paiardini, I. Frank, and G. Silvestri. Disruption of intestinal CD4+ T cell homeostasis is a key marker of systemic CD4+ T cell activation in HIV-infected individuals. *The Journal of Immunology*, 185(9):5169–5179, 2010.

[59] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

[60] F. Hahne, A. H. Khodabakhshi, A. Bashashati, C.-J. Wong, R. D. Gascoyne, A. P. Weng, V. Seyfert-Margolis, K. Bourcier, A. Asare, T. Lumley, et al. Per-channel basis normalization methods for flow cytometry data. *Cytometry Part A*, 77(2):121–131, 2010.

[61] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5(Oct):1391–1415, 2004.

[62] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[63] G. H. Heppner. Tumor heterogeneity. *Cancer research*, 44(6):2259–2265, 1984.

[64] A. V. Hoffbrand. *Postgraduate haematology*. John Wiley & Sons, 2016.

[65] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[66] A. Jalali and N. Pfeifer. Interpretable per case weighted ensemble method for cancer associations. *BMC genomics*, 17(1):501, 2016.

[67] R. Jansen, D. Greenbaum, and M. Gerstein. Relating whole-genome expression data with protein-protein interactions. *Genome research*, 12(1):37–46, 2002.

[68] M. J. Jaroszeski and G. Radcliff. Fundamentals of flow cytometry. *Molecular biotechnology*, 11(1):37–53, 1999.

[69] H. Jaspan, L. Liebenberg, W. Hanekom, W. Burgers, D. Coetzee, A. Williamson, F. Little, L. Myer, R. Coombs, D. Sodora, et al. Immune activation in the female genital tract during hiv infection predicts mucosal cd4 depletion and hiv shedding. *Journal of Infectious Diseases*, 204(10):1550–1556, 2011.

[70] V. M. Jiménez and A. Marzal. A lazy version of eppstein's k shortest paths algorithm. In *Experimental and Efficient Algorithms*, pages 179–191. Springer, 2003.

[71] I. Johnston. The chaos within: exploring noise in cellular biology. *Significance*, 9(4):17–21, 2012.

[72] M. Kærn, T. C. Elston, W. J. Blake, and J. J. Collins. Stochasticity in gene expression: from theories to phenotypes. *Nature Reviews Genetics*, 6(6):451, 2005.

[73] M. Kanehisa, S. Goto, Y. Sato, M. Kawashima, M. Furumichi, and M. Tanabe. Data, information, knowledge and principle: back to metabolism in KEGG. *Nucleic acids research*, 42(Database issue):D199–205, Jan. 2014.

[74] J.-H. Kim. Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. *Computational Statistics & Data Analysis*, 53(11):3735–3745, 2009.

[75] W. A. Knight, R. B. Livingston, E. J. Gregory, and W. L. McGuire. Estrogen receptor as an independent prognostic factor for early recurrence in breast cancer. *Cancer research*, 37(12):4669–4671, 1977.

[76] A. Krug, A. Towarowski, S. Britsch, S. Rothenfusser, V. Hornung, R. Bals, T. Giese, H. Engelmann, S. Endres, A. Krieg, et al. Toll-like receptor expression reveals CpG DNA as a unique microbial stimulus for plasmacytoid dendritic cells which synergizes with CD40 ligand to induce high

amounts of IL-12. *European Journal of Immunology*, 31(10):3026–3037, 2001.

[77] H.-T. KUHN. Aw (1951) nonlinear programming. In *2nd Berkeley Symposium. Berkeley, University of California Press.*

[78] H.-M. Lai, A. A. Albrecht, and K. K. Steinhöfel. irda: a new filter towards predictive, stable, and enriched candidate genes. *BMC genomics*, 16(1):1, 2015.

[79] N. D. Lakin and S. P. Jackson. Regulation of p53 in response to dna damage. *Oncogene*, 18(53), 1999.

[80] L. Lamoreaux, R. Koup, and M. Roederer. OMIP-009: Characterization of antigen-specific human T-cells. *Cytometry Part A*, 81(5):362–363, 2012.

[81] O. Lavi, G. Dror, and R. Shamir. Network-induced classification kernels for gene expression profile analysis. *J Comput Biol.*, 19(6):694–709, June 2012.

[82] H.-T. Lin, C.-J. Lin, and R. C. Weng. A note on Platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276, Aug. 2007.

[83] K. Lo, R. Brinkman, and R. Gottardo. Automated gating of flow cytometry data via robust model-based clustering. *Cytometry Part A*, 73(4):321–332, 2008.

[84] D. Lopez-Paz, P. Hennig, and B. Schölkopf. The randomized dependence coefficient. In *Advances in neural information processing systems*, pages 1–9, 2013.

[85] H. T. Maecker, J. P. McCoy, and R. Nussenblatt. Standardizing immunophenotyping for the Human Immunology Project. *Nature Reviews Immunology*, 12:191–200, 2012.

[86] Y. Mahnke and M. Roederer. OMIP-001: Quality and phenotype of Ag-responsive human T-cells. *Cytometry Part A*, 77(9):819–820, 2010.

[87] T. Marafioti, J. Paterson, E. Ballabio, K. Reichard, S. Tedoldi, K. Hollowood, M. Dictor, M. Hansmann, S. Pileri, M. Dyer, et al. Novel markers of normal and neoplastic human plasmacytoid dendritic cells. *Blood*, 111(7):3778–3792, 2008.

[88] A. Marusyk and K. Polyak. Tumor heterogeneity: causes and consequences. *Biochimica et Biophysica Acta (BBA)-Reviews on Cancer*, 1805(1):105–117, 2010.

[89] J. Mendelsohn, P. M. Howley, M. A. Israel, J. W. Gray, and C. B. Thompson. *The Molecular Basis of Cancer*. Elsevier Health Sciences, 2014.

[90] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, pages 415–446, 1909.

[91] D. Moazed. Mechanisms for the inheritance of chromatin states. *Cell*, 146(4):510–518, 2011.

[92] J. L. Mott, S. Kobayashi, S. F. Bronk, and G. J. Gores. mir-29 regulates mcl-1 protein expression and apoptosis. *Oncogene*, 26(42):6133–6140, 2007.

[93] S. Mukherjee. *The emperor of all maladies: a biography of cancer*. Simon and Schuster, 2011.

[94] D. Murdoch, J. Staats, and K. Weinhold. OMIP-006: Phenotypic subset analysis of human T regulatory cells via polychromatic flow cytometry. *Cytometry Part A*, 81(4):281–283, 2012.

[95] K. Murphy and C. Weaver. *Janeway's immunobiology*. Garland Science, 2016.

[96] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[97] D. Nam and S.-Y. Kim. Gene-set approach for expression pattern analysis. *Briefings in bioinformatics*, 9(3):189–197, 2008.

[98] National Institutes of Health Consensus Development Panel and others. National institutes of health consensus development conference statement: adjuvant therapy for breast cancer, november 1—3, 2000. *J Natl Cancer Inst.*, 93(13):979–989, 2001.

[99] U. Naumann, G. Luta, and M. Wand. The curvHDR method for gating flow cytometry samples. *BMC Bioinformatics*, 11(1):11–44, 2010.

[100] D. Norris and J. Stone. Who classification of tumours of haematopoietic and lymphoid tissues. 2008.

[101] K. O'Neill, N. Aghaeepour, J. Špidlen, and R. Brinkman. Flow cytometry bioinformatics. *PLoS computational biology*, 9(12):e1003365, 2013.

[102] O. Ornatsky, D. Bandura, V. Baranov, M. Nitz, M. Winnik, and S. Tanner. Highly multiparametric analysis by mass cytometry. *Journal of Immunological Methods*, 361(6030):1–20, 2010.

[103] K. O'Neill, A. Jalali, N. Aghaeepour, H. Hoos, and R. R. Brinkman. Enhanced flowtype/rchyoptimyx: a bioconductor pipeline for discovery in high-dimensional cytometry data. *Bioinformatics*, 30(9):1329–1330, 2014.

[104] M. Y. Park and T. Hastie. L1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):659–677, 2007.

[105] S.-Y. Park, J. H. Lee, M. Ha, J.-W. Nam, and V. N. Kim. mir-29 mirnas activate p53 by targeting p85α and cdc42. *Nature structural & molecular biology*, 16(1):23–29, 2009.

[106] D. R. Parks, M. Roederer, and W. A. Moore. A new "logicle" display method avoids deceptive effects of logarithmic scaling for low signals and compensated data. *Cytometry Part A*, 69(6):541–551, 2006.

[107] K. Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, pages 240–242, 1895.

[108] S. Perfetto, P. Chattopadhyay, and M. Roederer. Seventeen-colour flow cytometry: unravelling the immune system. *Nature Reviews Immunology*, 4(8):648–655, 2004.

[109] S. Peri, J. D. Navarro, R. Amanchy, T. Z. Kristiansen, C. K. Jonnalagadda, V. Surendranath, V. Niranjan, B. Muthusamy, T. Gandhi, M. Gronborg, et al. Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome research*, 13(10):2363–2371, 2003.

[110] F. Preijers, E. Huys, and B. Moshaver. OMIP-010: A new 10-color monoclonal antibody panel for polychromatic immunophenotyping of small hematopoietic cell samples. *Cytometry Part A*, 81(6):453–455, 2012.

[111] E. M. Price and W. P. Robinson. Adjusting for batch effects in dna methylation microarray data, a lesson learned. *Frontiers in genetics*, 9:83, 2018.

[112] S. Pyne, X. Hu, K. Wang, E. Rossin, T. Lin, L. Maier, C. Baecher-Allan, G. McLachlan, P. Tamayo, D. Hafler, et al. Automated high-dimensional flow cytometric data analysis. *Proceedings of the National Academy of Sciences*, 106(21):8519–8524, 2009.

[113] Y. Qian, Y. Liu, J. Campbell, E. Thomson, Y. M. Kong, and R. H. Scheuermann. Fcstrans: an open source software system for fcs file conversion and data transformation. *Cytometry Part A*, 81(5):353–356, 2012.

[114] Y. Qian, C. Wei, F. Eun-Hyung Lee, J. Campbell, J. Halliley, J. Lee, J. Cai, Y. Kong, E. Sadat, E. Thomson, et al. Elucidation of seventeen human peripheral blood B-cell subsets and quantification of the tetanus response using a density-based method for the automated identification of cell populations in multidimensional flow cytometry data. *Cytometry Part B: Clinical Cytometry*, 78(S1):S69–S82, 2010.

[115] P. Qiu, E. Simonds, S. Bendall, K. Gibbs Jr, R. Bruggner, M. Linderman, K. Sachs, G. Nolan, and S. Plevritis. Extracting a cellular hierarchy from high-dimensional cytometry data with spade. *Nature Biotechnology*, 29:886–891, 2011.

[116] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

[117] J. M. Raser and E. K. O'shea. Noise in gene expression: origins, consequences, and control. *Science*, 309(5743):2010–2013, 2005.

[118] C. E. Rasmussen and C. K. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[119] C. Redon, D. Pilch, E. Rogakou, O. Sedelnikova, K. Newrock, and W. Bonner. Histone h2a variants h2ax and h2az. *Current opinion in genetics & development*, 12(2):162–169, 2002.

[120] D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti. Detecting novel associations in large data sets. *Science*, 334(6062):1518–1524, 2011.

[121] J. Richter, M. Schlesner, S. Hoffmann, M. Kreuz, E. Leich, B. Burkhardt, M. Rosolowski, O. Ammerpohl, R. Wagener, S. H. Bernhart, et al. Recurrent mutation of the id3 gene in burkitt lymphoma identified by integrated genome, exome and transcriptome sequencing. *Nature genetics*, 44(12), 2012.

[122] M. Roederer. Compensation in flow cytometry. *Current Protocols in Cytometry*, pages 1–14, 2002.

[123] M. Roederer, J. Nozzi, and M. Nason. Spice: Exploration and analysis of post-cytometric complex multivariate datasets. *Cytometry Part A*, 79(2):167–174, 2011.

[124] M. Roederer and A. Tárnok. OMIPs—Orchestrating multiplexity in polychromatic science. *Cytometry Part A*, 77(9):811–812, 2010.

[125] Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In *Machine learning and knowledge discovery in databases*, pages 313–325. Springer, 2008.

[126] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[127] B. Scholkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[128] P. Schuster, N. Donhauser, K. Pritschet, M. Ries, S. Haupt, N. Kittan, K. Korn, and B. Schmidt. Co-ordinated regulation of plasmacytoid dendritic cell surface receptors upon stimulation with herpes simplex virus type 1. *Immunology*, 129(2):234–247, 2010.

[129] C. E. Shannon. The mathematical theory of communication. 1963. *MD computing: computers in medical practice*, 14(4):306–317, 1996.

[130] E. Shapiro, T. Biezuner, and S. Linnarsson. Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nat Rev Genet.*, 14(9):618–30, Sept. 2013.

[131] H. M. Shapiro. *Practical flow cytometry*. John Wiley & Sons, 2005.

[132] J. Shi and M. G. Walker. Gene set enrichment analysis (gsea) for interpreting gene expression profiles. *Current Bioinformatics*, 2(2):133–137, 2007.

[133] A. Smola and V. Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9:155–161, 1997.

[134] I. Steinwart, D. Hush, and C. Scovel. An explicit description of the reproducing kernel hilbert spaces of gaussian rbf kernels. *IEEE Transactions on Information Theory*, 52(10):4635–4643, 2006.

[135] C. Sternberg. Über eine eigenartige unter dem bilde der pseudoleukamie verlaufende tuberculose des lymphatischen apparates. *Ztschr Heilk*, 19:21–90, 1898.

[136] 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, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[137] I. Sugár and S. Sealfon. Misty Mountain clustering: application to fast unsupervised flow cytometry gating. *BMC Bioinformatics*, 11(1):502–508, 2010.

[138] R. Suzuki and H. Shimodaira. Pvclust: an r package for assessing the uncertainty in hierarchical clustering. *Bioinformatics*, 22(12):1540–1542, 2006.

[139] S. H. Swerdlow, E. Campo, S. A. Pileri, N. L. Harris, H. Stein, R. Siebert, R. Advani, M. Ghielmini, G. A. Salles, A. D. Zelenetz, et al. The 2016 revision of the world health organization classification of lymphoid neoplasms. *Blood*, 127(20):2375–2390, 2016.

[140] M. Swiecki and M. Colonna. Unraveling the functions of plasmacytoid dendritic cells during viral infections, autoimmunity, and tolerance. *Immunological Reviews*, 234(1):142–162, 2010.

[141] The Cancer Genome Atlas Network. The Cancer Genome Atlas (TCGA). `https://tcga-data.nci.nih.gov/tcga/`, 2006.

[142] The Cancer Genome Atlas Network. Comprehensive molecular portraits of human breast tumours. *Nature*, 490(7418):61–70, 2012.

[143] The Cancer Genome Atlas Network. Genomic and epigenomic landscapes of adult *de novo* acute myeloid leukemia. *N Engl J Med.*, 368(22):2059–2074, 2013.

[144] B. M. Turner. Histone acetylation and an epigenetic code. *Bioessays*, 22(9):836–845, 2000.

[145] L. J. van 't Veer, H. Dai, M. J. Van De Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[146] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 1995.

[147] V. Vapnik and A. Chervonenkis. A note on one class of perceptrons. *Automation and remote control*, 25(1), 1964.

[148] D. Venet, J. E. Dumont, and V. Detours. Most random gene expression signatures are significantly associated with breast cancer outcome. *PLoS computational biology*, 7(10):e1002240, Oct. 2011.

[149] J. P. Vial and F. Lacombe. Immunophenotyping of acute leukemia: Utility of CD45 for blast cell identification. *Methods in Cell Biology*, 64:343–358, 2001.

[150] F. Villanova, P. D. Meglio, M. Inokumad, N. Aghaeepour, E. Perucha, J. Mollon, L. Nomura, M. Hernandez-Fuentes, A. Copeh, T. Prevosti, S. Heck, V. Maino, G. Lord, R. R. Brinkman, , and F. O. Nestle. Integration of lyoplate based flow cytometry and computational analysis for standardized immunological biomarker discovery. *PLoS ONE*, 8(7), 2013.

[151] S. H. Walker and D. B. Duncan. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 54(1-2):167–179, 1967.

[152] C. Wei, J. Jung, and I. Sanz. OMIP-003: Phenotypic analysis of human memory B cells. *Cytometry Part A*, 79(11):894–896, 2011.

[153] A. Weintrob, A. Fieberg, B. Agan, A. Ganesan, N. Crum-Cianflone, V. Marconi, M. Roediger, S. Fraser, S. Wegner, and G. Wortmann. Increasing age at HIV seroconversion from 18 to 40 years is associated with favorable virologic and immunologic responses to HAART. *JAIDS Journal of Acquired Immune Deficiency Syndromes*, 49(1):40–47, 2008.

[154] D. B. West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[155] A. K. Whitchurch. Gene expression microarrays. *IEEE Potentials*, 21(1):30–34, 2002.

[156] A. K. White, M. VanInsberghe, O. I. Petriv, M. Hamidi, D. Sikorski, M. A. Marra, J. Piret, S. Aparicio, and C. L. Hansen. High-throughput microfluidic single-cell RT-qPCR. *Proceedings of the National Academy of Sciences*, 108(34), Aug. 2011.

[157] Y. Xiong, J.-H. Fang, J.-P. Yun, J. Yang, Y. Zhang, W.-H. Jia, and S.-M. Zhuang. Effects of microrna-29 on apoptosis, tumorigenicity, and prognosis of hepatocellular carcinoma. *Hepatology*, 51(3):836–845, 2010.

[158] S. Xue and M. Barna. Specialized ribosomes: a new frontier in gene regulation and organismal biology. *Nat Rev Mol Cell Biol.*, 13(6):355–69, June 2012.

[159] J. Y. Yen. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716, 1971.

[160] A. Younes. *Handbook of Lymphoma*. Springer, 2016.

[161] H. Zare, A. Bashashati, R. Kridel, N. Aghaeepour, G. Haffari, J. Connors, R. Gascoyne, A. Gupta, R. Brinkman, and A. Weng. Automated analysis of multidimensional flow cytometry data improves diagnostic accuracy between mantle cell lymphoma and small lymphocytic lymphoma. *American Journal of Clinical Pathology*, 137(1):75–85, 2012.

[162] H. Zare, P. Shooshtari, A. Gupta, and R. Brinkman. Data reduction for spectral clustering to analyze high throughput flow cytometry data. *BMC Bioinformatics*, 11(1):403–413, 2010.

[163] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. *Advances in neural information processing systems*, 16(1):49–56, 2004.

[164] J. Zhu, H. Zou, S. Rosset, and T. Hastie. Multi-class AdaBoost. *Statistics and its Interface*, 2(3), 2009.

[165] C. Zuleger and M. Albertini. OMIP-008: Measurement of Th1 and Th2 cytokine polyfunctionality of human T cells. *Cytometry Part A*, 81(6):450–452, 2012.