Universiteit Leiden

Computer Science

Binary classification and t-SNE for cancer datasets

Name:	Adam van Adrichem
Date:	12 Sept. 2019

1st supervisor: Michael Emmerich

2nd supervisor: Erwin Bakker

3rd supervisor: Andrew Stubbs

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)

Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

Abstract

A large part of cancer research is focused on detecting the cause of the development of cancer. Current approaches of identifying genes that effect cancer development takes several days. The equipment to prepare sequencing input and performing the sequencing itself is complex and requires good skilled IT specialists. This method is therefor less accessible for everyone and a faster method would be appreciated.

Research found that genes that influence cancer development are highly active in the cell. Therefore gene expression data gives a clearer image of the genes involved in cancer. Nowadays, gene expression data is relatively easy to obtain. Hence more cost-effective and timely than gene finding methods. This makes it an effective method to analyze active cancer genes and classify new patients.

In this master thesis, research is done on supervised clustering of gene expression data. In particular, gene expression profiles of cancer related datasets are analyzed. With state of the art classification algorithms were evaluated on several datasets. Because the size and high dimensionality of the datasets, the original dataset had to be reduced in lower dimensional subsets.

For visualization of the datasets we used t-SNE, a popular dimension reduction tool that clusters the data using a t-student distribution and a Gaussian distribution.

It was found that the state of the art algorithms were able to cluster and classify new patients in cancer related gene expression profiles in several case-studies.. The quality of the classification techniques differs per analyzed dataset, were the variance of the data made it challenging to correctly cluster the data using t-SNE.

Keywords: Gene expression, Cancer data analyses, Algorithms, Classification, Clustering, t-SNE.

1 Introduction

Over the year's cancer is found to be the most common cause of death by disease in the Netherlands. In 2017 31% of the people who died, died because of cancer (CBS, 2018). A lot of research is dedicated to better understand the causes of cancer and to finding possible cures and treatments. Here genes are known to play an important role in the development of cancer. Therefore identifying which genes have which role in the different forms of cancer is key to the understanding the diseases, and finding effective drugs.

In disease studies determining which genes are active, i.e. expressed, at different moments in time can be used to compare healthy versus diseased gene profiles to perceive dissimilarity.

Gene-expression is the process where information of a gene is expressed what results into a gene product. Genes contain genetic information which is read and transcribed from gene (DNA) to mRNA. mRNA is subsequently translated to a protein, enzyme or another gene product. Every cell in the body contains the same genes. Depending on epigenetics and other environmental factors (i.e. location in the body) a gene can be active, having a high gene expression level, or can have no or low activity, meaning to have a low or no gene expression level at all.

Searching for different gene expressions and by targeting these gene products in pathway analyses help understanding the cause of the disease and help finding a cure or inhibitor.

A special kind of gene to be found active in cancerous tissues are the so called fusion genes. Fusion genes are the product of two separated normal genes. The hybrid product is formed by fusion of parts of the two normal proteins. This fusion protein is extremely active (Yu, 2010) and causes to disorder important biological processes. Already many fusion genes are found in various cancer development studies (Edwards, 2010).

Beside fusion genes there are many other factors and processes that can mutate a gene and cause cancer development (GM., 2000). Most of the processes that are causing cancer have to do with over expressed genes influencing cell growth.

Cancer can be diagnosed in many ways. The diagnoses and treatments are depending on the symptoms of the patient. Based on the symptoms and possible type of cancer, different scans such as CT scans, MRI scans, Bone scans and others can be performed (Services, 2019). In most cases beside the scans, it is necessary to take a biopsy.

In cancer-research and treatment-development there is a lot of focus on what is causing the cancer and how to target the cause. This led to the following cancer diagnostics method as depicted in Figure 1.

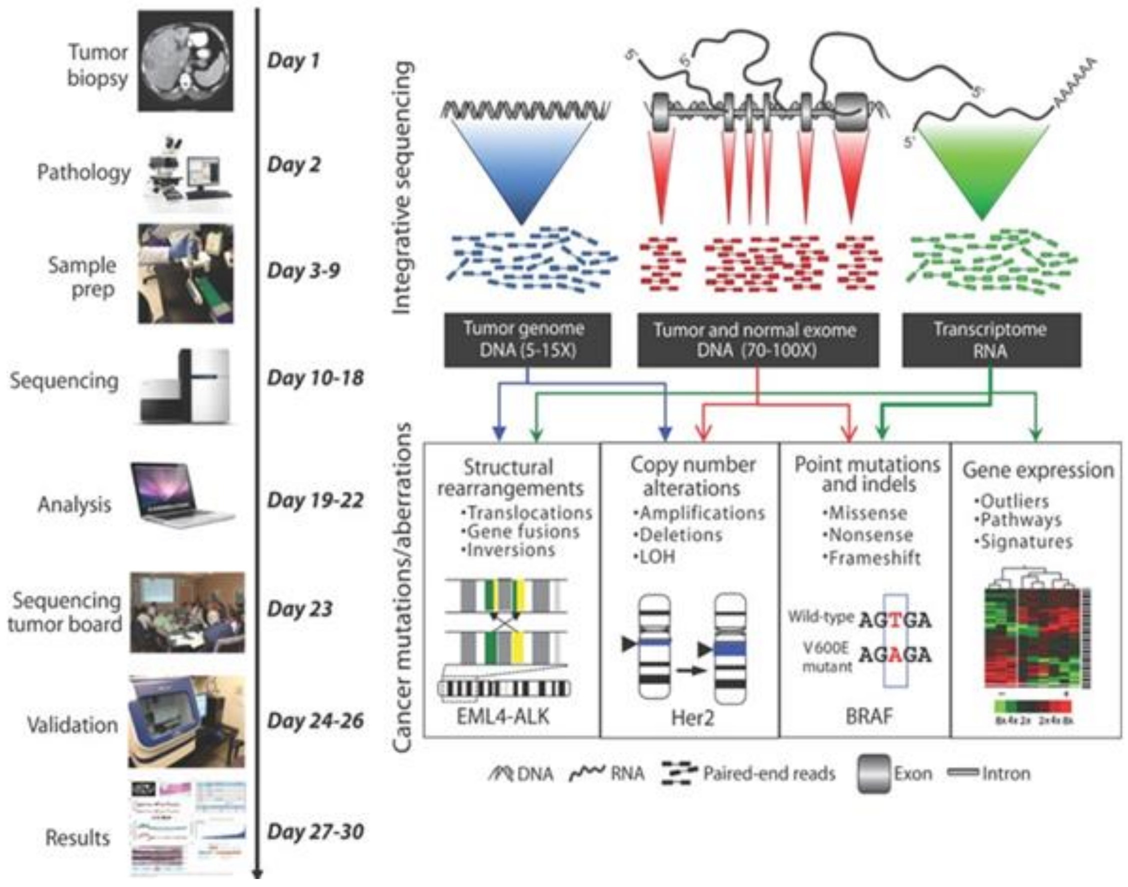


Figure 1: Brief overview of clinical approach of cancer treatment (Roychowdhury, 2011)

It consists of the following steps/actions:

1. (Day 1) When a patient is diagnosed with a tumor a biopsy is taken.
2. (Day 2) Pathologists investigate this biopsy to find the type of tumor and the current state of the tumor.
3. (Day 3 – 9th) A sample will be prepared, and the tumor cells will be cultured for further investigation.
4. (Day 10 – 18th) The prepared samples are sequenced, to measure the nucleotide sequence of the patient's DNA or RNA.
5. (Day 19 – 22nd) During sequencing small fragments of the DNA or RNA of a patient will be measured. With computational analysis the original RNA or DNA sequence can be put together. For the detection of fusion genes more advanced algorithms are necessary.
6. (Day 23) The sequencing tumor board will take place.
7. (Day 24 – 26th) Validation of fusion genes by polymerase chain reaction (PCR)
8. (Day 27 - 30) Finally, with the results the correct treatment can be applied.

In this thesis the following research questions will be studied.

- What is the state of the art of the current approach of finding fusion genes?
    - What are the costs and time span of current approach?
    - What makes it interesting to categorize on gene expression?

- Is it possible to categorize on cancer with gene expression datasets using state of the art machine learning algorithms?

- How does t-SNE compare to classical binary classification techniques?

Clustering is a data mining technique where objects get categorized into groups based on their similarity. Measuring the distance between objects tells something about their equality. The nearer an object the more related the objects are. The formed groups are called clusters.

There are several learning techniques within machine learning. Within this thesis we focus on supervised machine learning techniques. Supervised learning is based on training data where classes are known. An algorithm tries to distinguish the data based on their features/characteristics. There are many different algorithms which can do this and depending on the data one might be better than the other. There are algorithms good at dense areas others on small distance between cluster members, etc.

It is interesting to categorize on gene expression because it reduces the time span to find a target gene. This because obtaining gene expression is easier and cheaper than measuring reads for next generation sequencing. Less computational effort is needed (Skotheim, 2009).

Clustering gene expression data is different compared to other gene expression research where there is more focus on detecting specific genes which were found in earlier research to be involved in cancer development.

For better and easier understanding of the approach I will use the well-known ‘Iris dataset’ (Anderson, 1936) as an example dataset, to describe the methods and approaches in more detail, and to exemplify the machine learning principles and workflow on this running example.

In addition, we provide a cross-comparison of modern machine learning technologies, including the dimensional reduction and clustering technique t-SNE which is recently growing in popularity due to its potential to produce additional insight for the expert next to a binary classification and due to its ability to handle non-linearities. In this thesis we study the use of t-SNE as a clustering technique for (binary) classification (Maaten & Hinton, 2008).

The rest of the thesis is organized as follows; In section 2 the datasets that are used to evaluate are described. In section 3 the results are given, and finally in section 4 the conclusion and discussion are given.

2 Datasets

To evaluate the different analysis methods on gene expression data, several datasets are used. An example dataset (Maaten & Hinton, 2008) Iris flower dataset is used to explain the data preparation. The cancer related datasets are prepared in the same way as the iris dataset. Multiple datasets were analyzed but not all datasets where well suited to perform the classifiers on. This resulted into two of the four datasets being excluded. We still mention them in this thesis for completeness.

The Iris flower dataset
The Iris Dataset is a widely known dataset on the Iris flower that is used in many tutorials and studies. The iris dataset is introduced in 1936 by Ronald Fisher (Fisher, 1936). The dataset consists of 150 Iris flowers from 3 different plant species (Iris setosa, Iris versicolor and Iris virginica). For each specie 50 instances were given, where each instance consists of many measured features.

The Iris dataset has a multinomial classification. For each sample four features were measured. Each feature is a phenotypical observation. The length and the width of the sepals and petals of the plants were measured in centimeters.

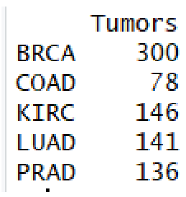
With this dataset, in his classical work on statistical classification, Fisher developed a linear discriminant model to distinguish the 3 different species. Since this seminal work the Iris data set has been very frequently used as a benchmark problem for binary classification. Therefore in this thesis we used this dataset as an example dataset being easy and understandable to explain the data preparation.

Prostate cancer dataset
The *prostate cancer dataset*, as we will call it in the thesis, is a dataset obtained in fusion gene research. More specifically it is about the role of the fusion gene TMPRS2-ERG in prostate cancer. In October 2009 it was published by Benjamin G. Barwick and his team (Barwick BG, 2010). They did analyses on the higher risk of recurrence when a patient has a TMPRS2-ERG fusion. From 139 patients with prostate cancer 502 molecular markers are measured using micro arrays. The micro arrays are analyzed with illumina assay cDNA Annealing, Selection, extension, and Ligation (DSAL). The data has a binary classification where 69 samples were positive for TMPRS2-ERG and 70 samples were not. The 502 molecular markers are

cancer-related genes and identify biomarkers of biochemical recurrence. The dataset‘s technical name is E-GEOD-18655.

Gene expression cancer RNA-seq dataset

The in this thesis called gene expression cancer dataset is as well obtained from the UCI machine learning repository (Dua, 2017). It was donated on June 2016 by Samuele Fiorini from the Genoa university. The original data set (hosted at [Web Link]#!Synapse:syn4301332) is maintained by the cancer genome atlas pan-cancer analysis project. The data set is a random extraction of gene expression data of patients having different tumors. The data contains 20531 attributes obtained from 801 instances. Every attribute is a gene. The 801 patients are labelled with a multinomial classification containing the five tumors BRCA, KIRC, COAD, LUAD and PRAD.



```
     Tumors
BRCA    300
COAD     78
KIRC    146
LUAD    141
PRAD    136
```

Table.2: Showing the number of patients for each tumor of the gene expression dataset.

The RNA-Seq gene expression levels were measured by illumina HiSeq platform. The Cancer Genomics Atlas (TCGA) has and still is analyzing large numbers of human tumors to discover molecular aberrations.

Also the following datasets were considered but in the end these were not used because of various reasons.

Leukemia cancer dataset
The *Leukemia dataset*, as we will call it in the thesis, is a dataset obtained in an Acute Myeloid Leukemia (AML) study in 2004 (Claudia D. Baldus, 2004). The dataset was published by Herbert Auer the same year. The study focused on Human BAC to study Chromosome 21 abnormalities. It contains 14 samples of 354 patients. Every sample is a gene and measured with micro arrays. 13 of the 14 genes are AML related, 1 gene is normal. The dataset technical name is GSE1065.
The labeling of the 354 patients was unclear which is the reason we could not use supervised learning algorithms. Testing the quality of the unsupervised clustering methods is not possible as input is unclear. This dataset is therefore excluded from this thesis.

Breast cancer dataset
The in this thesis called Breast cancer dataset is a dataset provided by the Oncology Institute to the UCI machine learning repository (Dua, 2017). M. Zwitter and M. Soklic provided the data in 1988 and is used in many machine learning studies. The data consists of 9 attributes (both linear and nominal). It has a binary classification for no-recurrence-events and recurrence-events. 201 instances of no-recurrence-events and 85 of recurrent events.

However, the attributes are not related to gene expression we could still preform supervised learning algorithms and compare results. This is excluded from the thesis.

2.1 Data preparation

To perform the classification methods on the datasets, some preparations have to be done. By having this preparation, all datasets can be approached and analyzed in the same way.

For this thesis we focused on binary classification. Both the Iris dataset and the Gene expression dataset have a multinomial classification. The Iris dataset contains tree different classes and the gene expression dataset contains five classes. The Prostrate cancer dataset is already binary classified which is the reason we did not have to change it. I separated the iris dataset into 4 different subsets where every subset has a binary classification. Each Iris class is once labeled as diseased. The other dataset is labeled as healthy. The Gene expression dataset is separated the same as the Iris dataset but into five subsets.

After creating the different subsets with binary labeling, I separated the datasets into a training set and a test set for supervised learning. The training set containing class labels is 2/3 of the data and the test set containing no labels is the remaining 1/3 of the data. Selecting the correct amount of data is important to prevent the model over/under fit the whole dataset. As well in other articles we found that 60% of the data is common to use. The instances of both the training set and test set are selected randomly.

The gene expression dataset contains more than 2000 attributes. Only the last 898 features are used to train and test the classification methods. The attributes of this dataset have many varieties. To be able to perform classification the dataset is normalized. This to reduce the variety and have better results. Normalization is done per column using the following normalization formula:

$$z = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Where z is the normalized value and x is the input value list you want to normalize.

Several well known classification algorithms, Linear Discriminant Analysis, Naïve Bayes, Random Forest and Support Vector Machine were applied to the datasets described in section 2. The training data is used to train the clustering model on. The test dataset is used to test the prediction quality of the algorithm.

The results were evaluated using precision, recall and the F-measure. To gain further insights also confusion matrices will be provided.

Within a classified group the precision is the number of instances correctly classified. The precision gives the true positives compared to the classified group. The instances within this classified group that are wrongly classified are called false positive.

$$\text{Precision} = \frac{tp}{tp + fp}$$

With the above formula you calculate the precision where tp stands for True positive and fp stands for false positive.

Recall gives the number of true positives compared to the correct
'real' labelling. Precision says something about the efficiency of predicting data where recall says something about the certainty of your classification.

$$\text{Recall} = \frac{tp}{tp + fn}$$

With the above formula you calculate the recall where tp stands for true positive and fn stands for false negative.

The f-measure is the harmonic mean between precision and recall and can be calculated using the following formula.

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

In summary, all datasets are prepared having binary classification and as less as possible variety reduced by normalization. The prostate cancer dataset is not reduced in size. The Iris flower dataset is used as an example dataset and is prepared to be an example for other multinomial classified datasets.

2.2 Technical explanation of algorithms

After preparing the data, classification algorithms can cluster the data. Using a prediction-model you are able to classify new data points. Several state-of-the-art classification algorithms are analyzed. The described algorithms are as well selected based on their availability in R packages and their ability to classify on gene expression data. Not all described algorithms are used for further analyses.

2.2.1. Algorithms used in further analyses

Linear Discriminant Analysis (LDA) (Fisher, 1936)
Linear Discriminant Analysis is a technique that reduces dimensionality within a dataset. The technique is a generalization of the Fisher's linear discriminant (Fisher, 1936). It is a supervised method where the input data should be labelled to train the classification model. The following steps are executed for LDA:

1. Calculate distance between the mean of different classes. This gives us the between class variance.
2. Calculate distance between the mean and samples of each class. This gives us the within class variance.
3. By maximizing the between class variance and minimizing the within class variance a lower dimensional space is created.

LDA's known advantages is that the method is relatively easy to preform, it's fast and effective.

In R LDA can be executed using the following command:

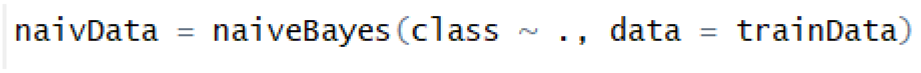
```
trainModel = lda( class ~ ., data = trainData)
```

Where 'class' describes the column name containing the labels for each instance. The '~ .' indicate there are no dependencies taken into account and where 'data' refers to the dataset.

Naïve Bayes
Naïve Bayes is as well a supervised algorithm named after the statistician and philosopher Thomas Bayes. The algorithm is based on the Bayes theorem where every feature is independent of other features. In real life this is not expected which is why it is called Naïve.

For every class and every feature, a probability score is calculated. When having a new data point its features will be measured. Per feature the new point will be categorized to a group it is most related to. Depending which class, the new point is categorized most the new point gets classified to.

In R Naïve bayes can be executed using the following command:

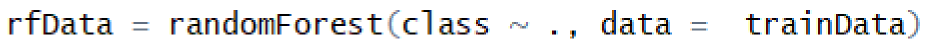


```
naivData = naiveBayes(class ~ ., data = trainData)
```

Where ‘class’ describes the column name containing the labels for each instance. The ‘~ .’ indicate there are no dependencies taken into account and where ‘data’ refers to the dataframe containing the training data.

Random Forest (Ho, 1995).
The random Forest algorithm is published by Tin Kam Ho in 1995 (Ho, 1995). The underlaying idea is based on decision tree logic. Sub selections with several features of the training data will be made. Of every subset a condition will be questioned based on the characteristics of the data. For every condition a decision will be made which will separate the data into categories. The sub selections of the data leads to more robust decisions as the more subsets with the same decision a higher probability the decision made is correct.

In R Random Forest can be executed using the following command:

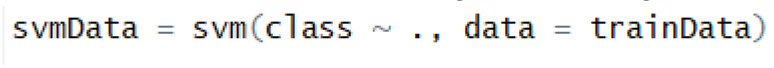


```
rfData = randomForest(class ~ ., data =  trainData)
```

Where ‘class’ describes the column name containing the labels for each instance. The ‘~ .’ indicate there are no dependencies taken into account and where ‘data’ refers to the dataframe containing the training data.

Support Vector Machine (SVM)
The support Vector Machine algorithm is based on the Generalized Portrait algorithm of Vapnik and Chervonenkis introduced in 1963. SVM is a non-probabilistic, binary linear classifier and as the other classifiers a supervised algorithm. With the SVM algorithms features are assigned either to one group or the other. The data should be a numeric model that needs to be designed in a vector space. SVM search for the best possible line or hyperplane to separate both classes. The points closest to the hyperplane are called the support vectors. The distance between the hyperplane and the support vectors is called the margin. The goal of SVM is to maximize the margin.

In R SVM can be executed using the following command:



```
svmData = svm(class ~ ., data = trainData)
```

Where ‘class’ describes the column name containing the labels for each instance. The ‘~ .’ indicate there are no dependencies taken into account and where ‘data’ refers to the dataframe containing the training data.

2.2.2. Algorithms not used in further analyses

K-Nearest Neighbor (k-NN) (E. Fix, 1989)
K-Nearest Neighbor is a supervised classification method known for its non-parametric approach. The classification does not make assumptions about the distribution. This classification method is good to use when there is little to no prior knowledge of the dataset. The method is developed because of the need of distinguishing datasets where there is no prior to little knowledge of the underlaying distribution and probability density. In 1951 E. Fix and J.L. Hodges introduced the k-nearest neighbor method in an unpublished US Air Force School of Aviation Medicine report (E. Fix, 1989).

The K-NN algorithm calculates for every test value the Euclidian distance between the test value and the training values. The K in the method stands for the number of neighbors the algorithm should look at. If k=3 the algorithm will calculate the Euclidian distance and select the 3 closest training points to further analyze. Are these 3 training points of the same class the test point will be classified in the same class. Depending on the k neighbor's classes a probability score depends in which the test point will be classified. K-NN is analyzed but not used in further analyses of the thesis.

Neural networks
There are several neural networks. The first mathematical model of a neuron was developed by Warren McCulloh and Walter Pitts in 1943. One of the first artificial neural networks was introduced by psychologist Frank Rosenblatt in 1958 called Perceptron. The purpose of this method was to model how the human brain processed visual data and learn recognizing objects. Neural networks mimic the human brain. Neural networks are a set of algorithms that cluster and classify real-world data such as images, sounds, text etc. It is self-learning using backpropagation. A neural network consists of input, multiple hidden layers of neurons and output. All input nodes neurons in the hidden layers and output nodes are connected and given a weight describing the effect of the node on the up-following node.

In summary, several state-of-the-art classification algorithms were selected. Beside their brought use they were as well selected based on their availability in R packages and their ability to classify on gene expression data.

2.3 Data visualization (t-SNE)

Gene expression data is high dimensional data. Every measured gene is a feature and has its own dimensionality. If you want to plot the data, you will plot a graph for each gene. To compare gene profiles of different patients a dimensionality reduction method is needed that can plot all genes for each patient at once in one graph. We looked as well at other visualization functionalities such as ggplot. In this thesis ggplot is used to visualize LDA predicted test data (including the predicted labelling).

To visualize (clustered) gene expression data several tools were considered, lending to the selection of the (student) t-Distributed Stochastic Neighbor Embedding (t-SNE) method.

The t-SNE method was introduced by Laurens van der Maaten and Geoffrey Hinton in 2008 (Maaten & Hinton, 2008). The technique is based on Stochastic Neighbor Embedding (Roweis, 2002). It is a non-linear technique. The main idea of the method is to plot high dimensional data into a two (max tree) dimensional map so you can visualize it. To do this, first a probability distribution is constructed where similar points get a high probability score and the less similar points get a lower probability score for being picked up. For this probability distribution the gaussian distribution is used. A low-dimensional map is created using the local relationships between points.

Gaussian distribution has a so called 'short tail' causing the nearby neighbor points overlap. The second step of t-SNE is performing the student t-distribution having a 'larger tail' and spreading the neighbors better.

t-SNE is performed on the complete dataset without labels. In this thesis we used the training and test dataset of the gene expression cancer dataset as input for t-SNE.

We considered multiple visualization methods, but t-SNE was selected for the visualization of gene expression data because it was easily accessible, easy in use and well documented. These reasons made it possible to integrate the method into a tool that on a standardized approach classify gene expression data and visualize it using t-SNE.

2.4 Binary Clustering Analysis and t-SNE (BCAT)

The created tool Binary Clustering Analysis and t-SNE BCAT is created in R.
All analyses are done in R version 3.5.2 (R Core Team, 2017), the packages Mass (Ripley, 2002), e1071 (Leisch, 2019) and Random Forest (Wiener, 2002) were used to run the classifiers. The packages ggplot2 (Wickham, 2016) and Rtsne (Krijthe, 2015) are used to produce the figures.

3 Results

In this chapter we show the classification results of multiple algorithms on the prostate cancer dataset. Here as well we show the results of LDA on the cancer gene expression dataset. For visualization we show some graphs constructed with t-SNE and ggplot. In addition, a standardized approach is created to cluster and classify gene expression profiles using state of the art algorithms.

3.1 Algorithms on datasets

As explained in the methods we performed several algorithms on the different datasets. The larger the dataset the more CPU is needed. In this thesis we did not analyze all datasets with all classification algorithms. Only a part of the data is analyzed and not all algorithms are performed on all the data which will be presented in this paragraph.

3.1.1 Prostate cancer dataset

The prostrate dataset contains 69 samples being positive for TMPRS2-ERG and 70 samples were not positive for the fusion TMPRS2-ERG. Sub selections of the data were made where training data and test data are defined. The test data consists of 21 patients being positive for TMPRS2-ERG and 26 patients being negative for TMPRS2-ERG.

In the table below is shown how LDA predicted the unlabeled test data. Over the rows the ‘real’ data is shown. Over the columns the predicted data is shown.

		Actual class	
		Fusion	**No fusion**
Predicted classes	**Fusion**	17	8
	No fusion	4	18

Table 1: Confusion matrix of LDA applied to the prostate cancer dataset.

From the 21 patients being positive for the fusion genes TMPRS2-ERG, 80,95% is clustered correctly. From the 26 patients being negative for TMPRS2-ERG, 69,23% is clustered correctly. The precision for classifying fusion genes is 17/17+8 => 17/25 = 0,68. The recall is 17/17+4 => 17/21 = 0,809. The f-measure is 2 * (0,68 * 0,809) / (0,68 + 0,809) => 2*(0,55 / 1,489) = 0,7388.

In the table below is shown how Naïve Bayes predicted the unlabeled test data. Over the rows the 'real' data is shown. Over the columns the predicted data is shown.

		Actual class	
		Fusion	**No fusion**
Predicted classes	**Fusion**	16	14
	No fusion	5	12

Table 2: Confusion matrix of Naïve Bayes applied to the prostate cancer dataset.

From the 21 patients being positive for the fusion genes TMPRS2-ERG, 76,19% is clustered correctly. From the 26 patients being negative for TMPRS2-ERG, 46,15% is clustered correctly. The precision for classifying fusion genes is 16/16+14 => 16/30 = 0,53. The recall is 16/16+5 => 16/21 = 0,76. The f-measure is 2 * (0,53 * 0,67) / (0,53 + 0,67) => 2*(0,355 / 1,2) = 0,592.

In the table below is shown how Random Forest predicted the unlabeled test data. Over the rows the 'real' data is shown. Over the columns the predicted data is shown.

		Actual class	
		Fusion	**No fusion**
Predicted classes	**Fusion**	18	13
	No fusion	3	13

Table 3: Confusion matrix of Random Forest applied to the prostate cancer dataset.

From the 21 patients being positive for the fusion genes TMPRS2-ERG, 85,71% is clustered correctly. From the 26 patients being negative for TMPRS2-ERG, 50% is clustered correctly. The precision for classifying fusion genes is 18/18+13 => 18/31 = 0,58. The recall is 18/18+3 => 18/21 = 0,857. The f-measure is 2 * (0,58* 0,857) / (0,58+ 0,857) => 2*(0,497 / 1,437) = 0,6917.

In the table below is shown how SVM predicted the unlabeled test data. Over the rows the 'real' data is shown. Over the columns the predicted data is shown.

		Actual class	
		Fusion	**No fusion**
Predicted classes	**Fusion**	15	10
	No fusion	6	16

Table 4: Confusion matrix of SVM applied to the prostate cancer dataset.

From the 21 patients being positive for the fusion genes TMPRS2-ERG, 71,43% is clustered correctly. From the 26 patients being negative for TMPRS2-ERG, 61,54% is clustered correctly. The precision for classifying fusion genes is 15/15+10 => 15/25 = 0,6. The recall is 15/15+6 => 15/21 = 0,714. The f-measure is 2 * (0,6 * 0,714) / (0,6 + 0,714) => 2*(0,4284 / 1,314) = 0,652.

3.2.4 Gene expression cancer dataset

The gene expression dataset consists of five different tumors and is separated into 5 different subsets. The subsets contain over 20 000 attributes. In particular, we looked at the last 859 attributes that were selected for further analyses. We performed the LDA algorithm on all the reduced subsets of the gene expression cancer dataset.

The first dataset analyzed is the "Kirc - Brca" dataset. The training dataset contain patients from the Kirc dataset labelled with 'cancer' and patients from the Brca dataset labeled with 'other'.

In the table below is shown how LDA predicted the unlabeled test data consisting of 103 patients labeled with cancer and 46 patients labeled with other.

		Actual class	
		Cancer	**Other**
Predicted classes	**Cancer**	103	0
	Other	0	46

Table 5: Confusion matrix of LDA applied to the Kirc-Brca dataset.

From the 103 patients labeled with cancer 100% is clustered correctly and from the 46 patients labeled with other is 100% clustered correctly. The precision for classifying the cancer class is 103/103+0 = 1. The recall is 103/103+0=1. The f-measure is 2 * (1 * 1) / (1 + 1) => 2*(1 / 2) = 1.

The next analyzed dataset is "Brca - Prad" where Brca is labeled as ‘cancer’ and Prad is labeled as ‘other’. For the training dataset 41 patients are labelled with ‘cancer’ and 105 patients are labeled with ‘other’. In the table below is shown how LDA predicted the unlabeled test data.

		Actual class	
		Cancer	**Other**
Predicted classes	**Cancer**	41	0
	Other	0	105

Table 6: Confusion matrix of LDA applied to the Brca-Prad dataset.

LDA clustered the data with a 100% accuracy. The precision for classifying the cancer class is 41/41+0 = 1. The recall is 41/41+0 = 1. The f-measure is 2 * (1 * 1) / (1 + 1) => 2*(1 / 2) = 1.

For the "Prad - Luad" dataset Prad is labeled as ‘cancer’ and Luad is labeled as ‘other’. The training dataset contain 51 patients labelled with ‘cancer’ and 42 patients are labeled with ‘other’. Here as well LDA clustered the data with a 100% accuracy. The precision for classifying the cancer class is 51/51+0 = 1. The recall is 51/51+0=1. The f-measure is 2 * (1 * 1) / (1 + 1) => 2*(1 / 2) = 1.

		Actual class	
		Cancer	**Other**
Predicted classes	**Cancer**	51	0
	Other	0	42

Table 7: Confusion matrix of LDA applied to the Prad-Luad dataset.

It was not possible to run LDA on the dataset "Luad - Coad" as the arrays were non-conformable. Too many fields were missing.

In the "Coad - Kirc" dataset Coad was labeled as ‘cancer’. There are 50 patients labelled with ‘cancer’. The Kirc dataset was labeled with ‘other’ were 24 patients are correctly classified with ‘other’ and 1 patient was classified false positive. Here LDA clustered the data with a 98,67% accuracy. The precision for classifying the cancer class is 50/50+0 = 1. The recall is 50/50+0=1. The f-measure is 2 * (1 * 1) / (1 + 1) => 2*(1 / 2) = 1.

		Actual class	
		Cancer	Other
Predicted classes	Cancer	50	1
	Other	0	24

Table 8: Confusion matrix of LDA applied to the Kirc-Brca dataset.

3.2 Visualization

For visualization we used two visualization functionalities to plot the gene expression cancer dataset. We made use of t-SNE including an unsupervised clustering method and we made use of the ggplot showing the predicted test data by the LDA classifier.

In the picture below (figure 2) output of the t-SNE method on dataset Kirc-Brca is shown. Both x-axis and y-axis represents units for the 2- dimensional space created by t-SNE preserving similarity information from the higher dimensionality.

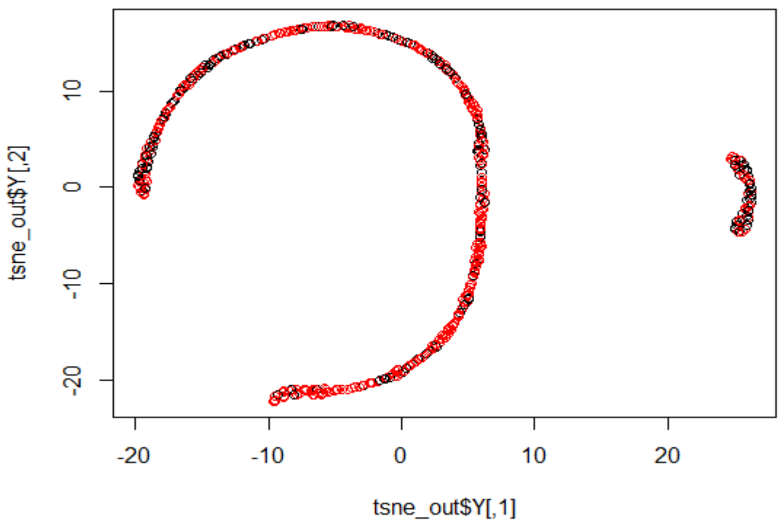


Figure 2: t-SNE graph as an result of performing the method on the dataset Kirc-Brca. On the x-axis and y-axis the 2-dimensional space created by t-SNE are presented.

In the picture below (figure 3) a ggplot of the LDA classification method on the Kirc-Brca dataset is shown. On the x-axis the predicted species are shown and, on the y-axis, the posterior probabilities.

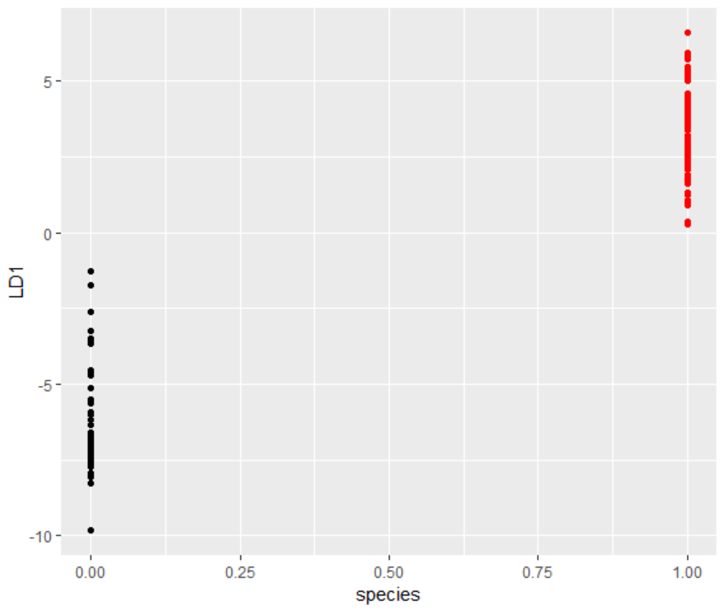


Figure 3: ggplot graph plotting the LDA predicted testdata points. On the x-axis the species and on the y-axis the LDA dimensionality space created by LDA.

The used input for the ggplots is the predicted testdata classified by LDA.

In the picture below (figure 4) a ggplot of the LDA classification method on the Brca-Prad dataset is shown. On the x-axis the predicted species are shown and, on the y-axis, the posterior probabilities.

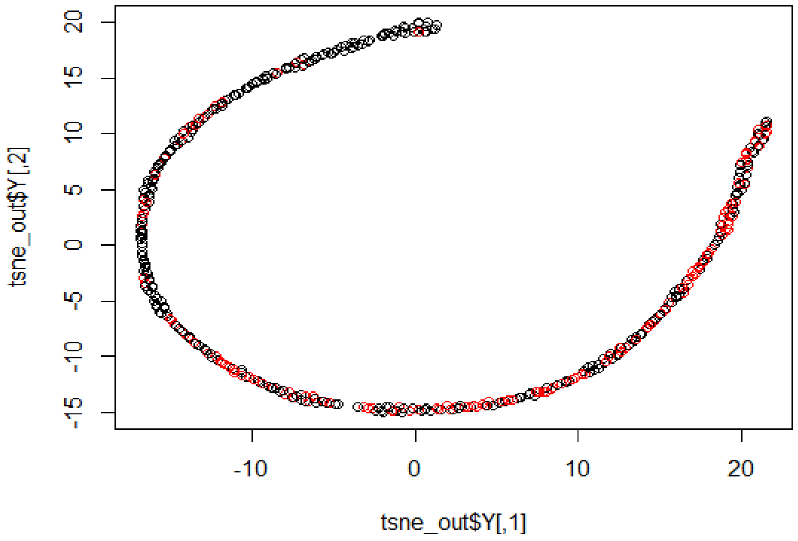


Figure 4: t-SNE graph as an result of performing the method on the dataset Brca-Prad. On the x-axis and y-axis the 2-dimensional space created by t-SNE are presented.

In the picture below (figure 5) a ggplot of the LDA classification method on the Brca-Prad dataset is shown. On the x-axis the predicted species are shown and on the y-axis the posterior probabilities.

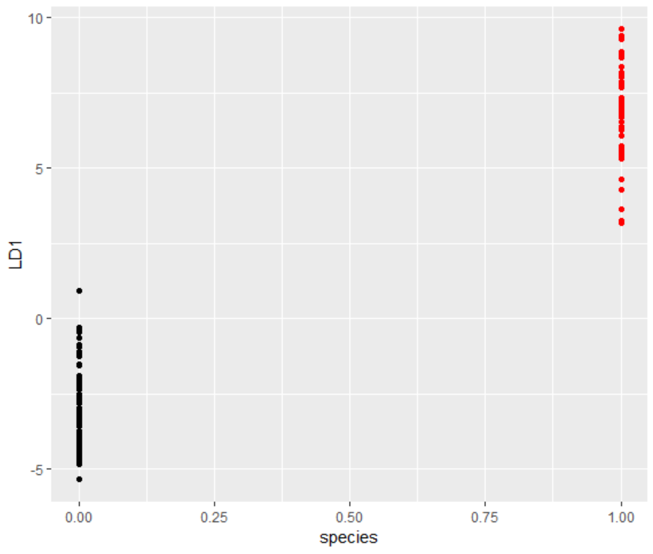


Figure 5: ggplot graph plotting the LDA predicted testdata points. On the x-axis the species and on the y-axis the LDA dimensionality space created by LDA.

In the picture below (figure 6) a ggplot of the LDA classification method on the Prad-Luad dataset is shown. On the x-axis the predicted species are shown and, on the y-axis, the posterior probabilities.

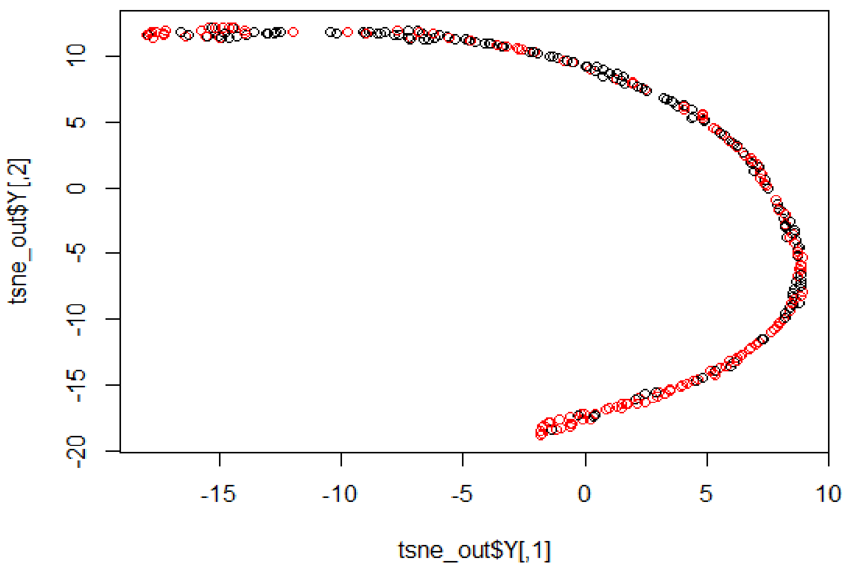


Figure 6: t-SNE graph as an result of performing the method on the dataset Prad-Luad. On the x-axis and y-axis the 2-dimensional space created by t-SNE are presented.

In the picture below (figure 7) a ggplot of the LDA classification method on the Kirc-Brca dataset is shown. On the x-axis the predicted species are shown and on the y-axis the posterior probabilities.

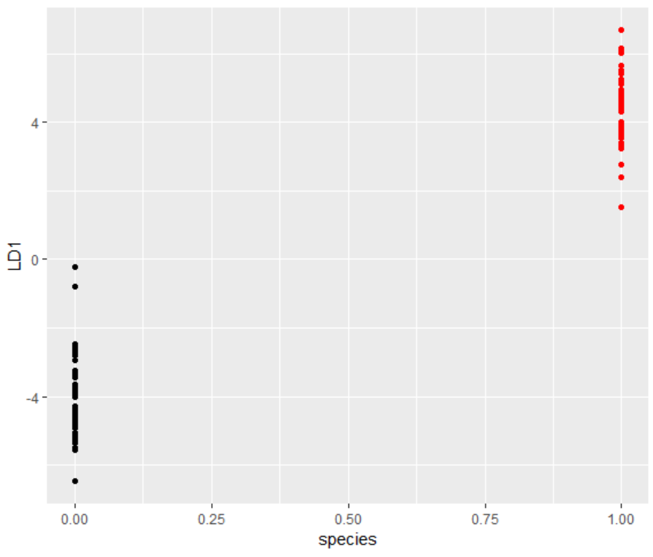


Figure 7: ggplot graph plotting the LDA predicted testdata points. On the x-axis the species and on the y-axis the LDA dimensionality space created by LDA.

As in the method explained the dataset Luad – Coad had missing values which is the reason we could not perform LDA and neither could perform t-SNE on this dataset.

In the picture below (figure 8) a ggplot of the LDA classification method on the Coad-Kirc dataset is shown. On the x-axis the predicted species are shown and, on the y-axis, the posterior probabilities.

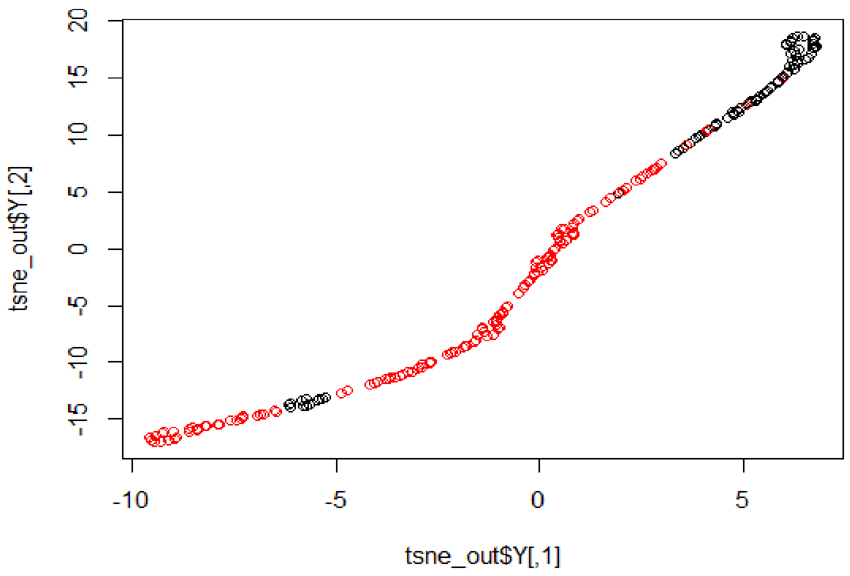


Figure 8: t-SNE graph as an result of performing the method on the dataset Coad-Kirc. On the x-axis and y-axis the 2-dimensional space created by t-SNE are presented.

In the picture below (figure 9) a ggplot of the LDA classification method on the Coad-Kirc dataset is shown. On the x-axis the predicted species are shown and on the y-axis the posterior probabilities.

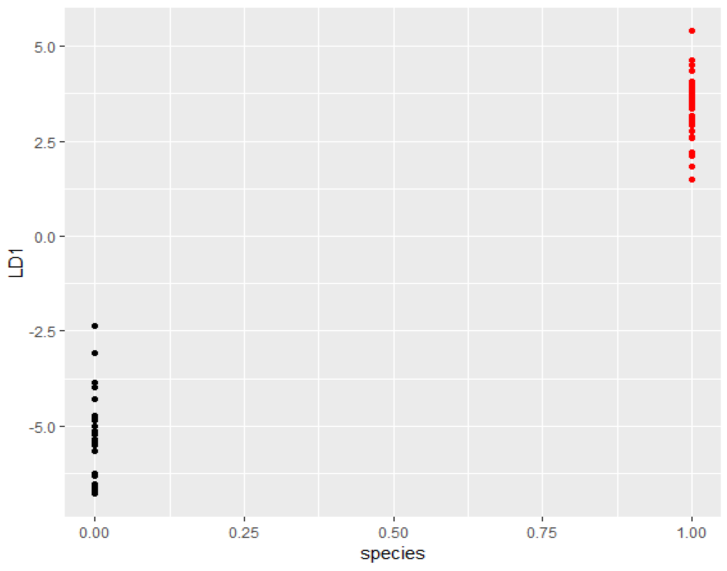


Figure 9: ggplot graph plotting the LDA predicted testdata points. On the x-axis the species and on the y-axis the LDA dimensionality space created by LDA.

4 Conclusion and discussion

The state of the art of the current approach of finding fusion genes is explained in detail in the introduction. As stated in the introduction, detecting fusion genes takes several days. Measuring DNA and RNA, before sequencing, is taking already 9 days. In this thesis it was tried to find an easier and faster approach of finding fusion genes by clustering gene expression data and use the classifiers to predict new patients. More analysis and fine tuning of the algorithms is needed to obtain more robust reliable results.

The classification method LDA scored with a 100% accuracy really well in predicting new patients for the gene expression cancer dataset. For the prostate cancer dataset consisting of gene profiles of biomarkers related to cancer, LDA was not as good in predicting new patients.

Gene expression profiles have many dimensions. Within a gene there is a lot of variety which makes clustering difficult for the classifiers. The classifiers are not able to cluster data with high variety.

It is possible to categorize on cancer with gene expression datasets using state of the art machine learning algorithms. However, it is not predicting always with 100% accuracy. There are still many false positives and false negatives found in the prostate cancer dataset which is too risky and not acceptable when doing cancer diagnosis.

Blindly follow clustering methods to identify cancer markers is discouraged because of the number of false negatives and false positives found in this research. In addition, the used algorithms were based on supervised learning which should predict better than unsupervised.

The visualization method t-SNE including an unsupervised clustering method is an interesting tool but seems not well suited for gene expression data when using default parameters. The graphs created by the tool with default parameters where not able to separate the data completely. Visualizing clustered gene expression profiles have a too large variety. Further analyses is needed to see if it is possible to tweak parameters leading to better results.

It can be concluded a standard approach is created to analyze state of the art clustering and classification techniques on gene expression profiles. A manual for how to use this product is given in appendix 6.1.

5 Future work

Not all algorithms on all data was analyzed. Further work would be to separate the gene expression cancer dataset more and perform and compare all algorithms described in this thesis.

We didn’t use Neural networks because of the variation of different networks. Because of time we were not able to investigate which neural network suits best for this type of data.

We thought about investigating other deep learning networks. These techniques require huge multilayered datasets. Not many gene expression datasets related to cancer were available. However, it is worthy to investigate in further work the different deep learning networks.

Besides t-SNE other visualization techniques might be interesting to use.

Further work might be more focus on fusion genes and its characteristics. You would expect for fusion genes a strong relation between 2 features. Fine tuning a classification algorithm would be an idea.

References

Anderson, E. (1936). The species problem in Iris. *Annals of the Missouri Botanical Garden., 23*(3), 457–509. doi:10.2307/2394164

Barwick BG, A. M.-J. (2010). Prostate cancer genes associated with TMPRSS2-ERG gene fusion and prognostic of biochemical recurrence in multiple cohorts. *British Journal of Cancer*, 570 – 576. doi:10.1038/sj.bjc.6605519

CBS. (2018, July 10). *Cancer is the most common cause of death by disease*. Retrieved from Cancer is the most common cause of death by disease: https://www.cbs.nl/nl-nl/nieuws/2018/28/kanker-oorzaak-bij-31-procent-van-de-sterfgevallen

Claudia D. Baldus, *. S. (2004). Acute Myeloid Leukemia with complex karyotypes and abnormal chromosome 21: Amplification discloses overexpression of APP, ETS2, and ERG genes. *PNAS, 101*(11), 3915–3920.

Dua, D. a. (2017, 12 1). *UCI Machine Learning Repository*. (D. a. Dua, Editor) Retrieved from UCI Machine Learning Repository: http://archive.ics.uci.edu/ml

E. Fix, &. H. (1989, December). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *International Statistical Review / Revue Internationale de Statistique, 57*(3), pp. 238-247. doi:10.2307/1403797

Edwards, P. A. (2010). Fusion genes and chromosome translocationsin the common epithelial cancers. *Journal of Pathology*, 244 – 254. doi:10.1002/path.2632

Fisher, R. A. (1936). THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS. *Annals of Eugenics banner, 7*(2), 179-188. doi:10.1111/j.1469-1809.1936.tb02137.x

GM., C. (2000). *The Cell: A Molecular Approach.* Sunderland (MA): Sinauer Associates.

Ho, T. K. (1995). Random decision forests. *ICDAR '95 Proceedings of the Third International Conference on Document Analysis and Recognition*, 278.

Krijthe, J. H. (2015). *Rtsne: T-Distributed Stochastic Neighbor Embedding using a Barnes-Hut Implementation.* Retrieved from https://github.com/jkrijthe/Rtsne

Leisch, D. M. (2019). *e1071: Misc Functions of the Department of Statistics, Probability.* Retrieved from https://CRAN.R-project.org/package=e1071

Maaten, L. v., & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research 9*, 27.

R Core Team. (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing,. Retrieved from https://www.R-project.org/

Ripley, W. N. (2002). *Modern Applied Statistics with S* (Fourth ed.). New York: Springer. Retrieved from http://www.stats.ox.ac.uk/pub/MASS4

Roweis, G. H. (2002). *Stochastic Neighbor Embedding* (Vol. 15). Cambridge: The MIT Press.

Roychowdhury, S. (2011, November 30). Personalized oncology through integrative high-throughput sequencing: a pilot study. *Sci Transl Med., 3*(111), 20. doi:10.1126/scitranslmed.3003161

Services, U. D. (2019, July 17). *National Cancer Institute*. Retrieved from National Cancer Institute: https://www.cancer.gov/

Skotheim, R. I. (2009). A universal assay for detection of oncogenic fusion transcripts by oligo microarray analysis. *Mol Cancer.*, 8: 5.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York. Retrieved from http://ggplot2.org

Wiener, A. L. (2002). Classification and Regression by randomForest. *R News, 2*(3), 18-22. Retrieved from https://CRAN.R-project.org/doc/Rnews/

Yu, J. (2010). An Integrated Network of Androgen Receptor, Polycomb, and TMPRSS2-ERG Gene Fusions in Prostate Cancer Progression. *Cancer Cell, 17*(5), 443–454. doi:10.1016/j.ccr.2010.03.018

6 Appendix

6.1 Source code Binary Clustering Analysis and t-SNE (BCAT) (including manual)

A standard approach is realized by creating the tool BCAT. This tool is written in R and contain functionalities that make it possible to standardize prepare your binary labelled dataset and run several classification algorithms on. It is as well possible by following this manual to perform a t-SNE classification and plot the t-SNE graphics.

```
##########################################
## Adam van Adrichem
## Master thesis 2019
## Binary classification and t-SNE for cancer datasets

## Used Clustering techniques
# LDA
# naive bayes
# Random forest â decision tree
# SVM
# logistic regression
# Multilayer percepton
```

1. The following libraries are needed to perform all the functions.

```
###########################
## Load libraries
loadLibraries <- function(){
  library(MASS)          #Used for LDA & SVM
  library(e1071)         #Used for Naive Bayes
  library(randomForest)  #Used for Random Forest
  library(ggplot2)       #Used for ggplot
  library(Rtsne)         #Used for tsne
}
```

2. Data preparation is needed and differ per dataset. The data should have binary classification. The column containing the labels should be named as ‘class’. The following code is used to prepare the datasets: Prostate cancer dataset, Leukemia dataset, breast cancer dataset and the gene expression cancer dataset.

```
##########################
readInData <- function(fileName){
  if(fileName == "prostate"){
    preData = read.delim("allPatients_class_weka(2).txt", header=T)
    preData = preData[,-1] # Delete row.names from first column
  }
  else if(fileName == "leukemia"){
    preData = read.delim("GSE1065_series_matrix_prepaired.txt", header=T)
  }
  else if(fileName == "breastcancer"){
    preData = read.table("breast-cancer.data", sep=',', header=FALSE)
    names(preData) = c("class", "age","menopause","tumor-size","inv-
nodes","node-caps","deg-malig","breast","breast-quad","irradiat")
    preData = preData[,c(2:10,1)]
  }
  else if(fileName == "geneExpression"){
    labels = read.table("labels.csv", sep=',')
    preData = read.table("data.csv", sep=',', header=FALSE)
    preData = data.frame(preData[,-1],labels[,2])
    row.names(preData) = labels[,1]
    names(preData)[length(names(preData))] = c("class")
  }
  preData=preData[sample(nrow(preData)),]
  return(preData)
}
```

3. The gene expression cancer dataset needs to be normalized because of the large amount of variety. This is done by using the following functions.

```
##########################
##Normalize column
normalizeCol <- function(x){
#  print(paste("Trying to process: ", min(x), max(x)))
  normalizedData = (x-as.numeric(min(x)))/as.numeric((max(x))-
as.numeric(min(x)))
  return(normalizedData)
}

## Normalize complete table
normalizeDataSet <- function(dataset,targetCol){
  x=as.numeric(dataset[,targetCol])
  for(i in c(c(targetCol+1):length(dataset[1,]))){
    x=data.frame(x,normalizeCol(as.numeric(dataset[,i])))
  }
  names(x)[length(x[1,])] = "class"
  return(x)
}
```

4. separating training data and test data is done with the following functionality. The output of this function is a list containing training data and test data and a variable containing the reference ID’s for both subsets.

```
##########################
createTestTrainData <- function(dataset){

  trainSetId = sample(1:length(dataset[,1]), 2*(length(dataset[,1])/3))
  trainData = dataset[trainSetId,]

  testSetId = rownames(dataset[-trainSetId,])
  testData = dataset[testSetId,-which(names(dataset)=="class")]

  testTrainData = list()
  testTrainData$trainSetId = trainSetId
  testTrainData$trainData = trainData
  testTrainData$testSetId = testSetId
  testTrainData$testData = testData

  return(testTrainData)
}
```

5. The next step is running the classifiers and prepare output tables. With below code the LDA classifier is ran. The input parameters are:
Dataset: complete dataset
testTrainData: Output vector from function 'createTestTrainData'

```
##########################
runLDA <- function(dataset, testTrainData){
  out = list()
  out$trainedModel = lda(class ~ ., dataset, subset = testTrainData$trainSetId)
  out$plda = predict(object = out$trainedModel, # predictions
                newdata = testTrainData$testData)

  #From the complet dataset select column class from all rows from the testset
  testDataClass =
dataset[testTrainData$testSetId,which(names(dataset)=="class")]
  #Create table with correct labelling
  realDataTable = table(testDataClass)
  names(realDataTable) = c("Other", "Cancer")
  #Create table with correct labelling vs predicted labelling
  predictedTable = table(testDataClass, out$plda$class)
  colnames(predictedTable) = c("Other", "Cancer")
  rownames(predictedTable) = c("Other", "Cancer")

  print("Test data contain the following classification: ")
  print(realDataTable)
  # print(predictedTable)
  out$result = predictedTable
  return(out)
}
```

6. below code the NaiveBayes classifier is ran. The input parameters are:
Dataset: complete dataset
testTrainData: Output vector from function ‘createTestTrainData’

```
##########################
runNaiveBayes <- function(dataset, testTrainData){
  out = list()
  out$trainedModel = naiveBayes(as.factor(class) ~ ., dataset, subset =
testTrainData$trainSetId)
  out$plda = predict(object = out$trainedModel, # predictions
                     newdata = testTrainData$testData)
  #From the complet dataset select column class from all rows from the testset
  testDataClass =
dataset[testTrainData$testSetId,which(names(dataset)=="class")]
  #Create table with correct labelling
  realDataTable = table(testDataClass)
  names(realDataTable) = c("Other", "Cancer")
  #Create table with correct labelling vs predicted labelling
  predictedTable = table(testDataClass, out$plda)
  colnames(predictedTable) = c("Other", "Cancer")
  rownames(predictedTable) = c("Other", "Cancer")

  print("Test data contain the following classification: ")
  print(realDataTable)
  # print(predictedTable)
  out$result = predictedTable
  return(out)
}
```

7. With below code the Random Forest classifier is ran. The input parameters are:
Dataset: complete dataset
testTrainData: Output vector from function ‘createTestTrainData’

```
##########################
runRandomForest <- function(dataset, testTrainData){
  out = list()
  out$trainedModel = randomForest(as.factor(class) ~ ., dataset, subset =
testTrainData$trainSetId)
  out$plda = predict(object = out$trainedModel, # predictions
                   newdata = testTrainData$testData)
  #From the complet dataset select column class from all rows from the testset
  testDataClass =
dataset[testTrainData$testSetId,which(names(dataset)=="class")]
  #Create table with correct labelling
  realDataTable = table(testDataClass)
  names(realDataTable) = c("Other", "Cancer")
  #Create table with correct labelling vs predicted labelling
  predictedTable = table(testDataClass, out$plda)
  colnames(predictedTable) = c("Other", "Cancer")
  rownames(predictedTable) = c("Other", "Cancer")

  print("Test data contain the following classification: ")
  print(realDataTable)
  # print(predictedTable)
  out$result = predictedTable
  return(out)
}
```

8. With below code the SVM classifier is ran. The input parameters are:
Dataset: complete dataset
testTrainData: Output vector from function ‘createTestTrainData’

```
##########################
runSVM <- function(dataset, testTrainData){
  out = list()
  out$trainedModel = svm(as.factor(class) ~ ., dataset, subset =
testTrainData$trainSetId)
  out$plda = predict(object = out$trainedModel, # predictions
                    newdata = testTrainData$testData)
  #From the complet dataset select column class from all rows from the testset
  testDataClass =
dataset[testTrainData$testSetId,which(names(dataset)=="class")]
  #Create table with correct labelling
  realDataTable = table(testDataClass)
  names(realDataTable) = c("Other", "Cancer")
  #Create table with correct labelling vs predicted labelling
  predictedTable = table(testDataClass, out$plda)
  colnames(predictedTable) = c("Other", "Cancer")
  rownames(predictedTable) = c("Other", "Cancer")

  print("Test data contain the following classification: ")
  print(realDataTable)
  # print(predictedTable)
  out$result = predictedTable
  return(out)
}
```

9. With below code a t-SNE classification will be performed with a t-SNE graph as result. The input parameters are:
Mat_data: complete untrained dataset without labels
K are the amount of kernels, with binary classification only 2. Are other fields are set default.

```
##########################
# preform tsne
tsne(mat_data, initial_config = NULL, k = 2, initial_dims = 30, perplexity = 30,
    max_iter = 1000, min_cost = 0, epoch_callback = NULL, whiten = TRUE,
    epoch=100)
```