ANALYSIS OF GENE EXPRESSION DATA OF TUMOR VS. NORMAL CELLS IN CLEAR CELL RENAL CELL CARCINOMA

ANTONIOS KOUTOUNIDIS

rijksuniversiteit
groningen

Master of Science
Computing Science
Faculty of Science and Engineering
University of Groningen

August 2018

SUPERVISORS:
First Supervisor: Michael Biehl, University of Groningen
Second Supervisor: Kerstin Bunte, University of Groningen

LOCATION:
Copenhagen

TIME FRAME:
August 2018

ABSTRACT

The last years as more and more reliable cancer data sets become available and more researchers work on them, the way that a drug is produced is changing. Knowledge about the mechanisms of a disease can be acquired from those data sets. Based on a resent analysis of gene expression data which addressed the prediction of recurrence risk in patients with clear cell Renal Cell Carci-noma, we study in more detail the classification problem, whether a sample is healthy or unhealthy. Using a GMLVQ classifier we observe that even a simple classifier trained by a significant small number of random genes can achieve great results in respect of performance. At the end we show that, even the information to classify a sample as healthy or unhealthy is spread on many genes, still there is a level of significance between the genes.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth [3]

ACKNOWLEDGMENTS

I would like to express my appreciation to Michael Biehl who gave me the opportunity to work on such an interesting project. He was always keen on giving me guidance answering every question, giving me the necessary amount of time to complete the research internship.
.

CONTENTS

ACRONYMS

GMLVQ General Matrix Learning Vector Quantization

TCGA The Cancer Genome Atlas

ccRCC clear cell Renal Cell Carcinoma

LVQ Learning Vector Quantization

NPC Nearest Prototype Classification

GLVQ Generalized Learning Vector Quantization

AUC Area Under the Curve

lasso least absolute shrinkage and selection operator

1 INTRODUCTION

The recent years large cancer data sets became available to the public. These data sets started changing the clinical care, as the sufficient data mining tools, used by scientists across the globe were able to extract valuable information.

Usually, gene expression and other significant data were available only on noisy unreliable platforms and only for small number of samples. Hence, results noticed from these platforms were not reliable, usually incomparable, and often not reproducible. As a result of this, those results are not taken into account to change clinical practices.

However, public data sets such as data sets included in The Cancer Genome Atlas (TCGA) repository [2], became available. This data sets include accurate sequence, expression and clinical data on a variety of cancers and this is transforming the way the drugs are discovered. Using these data sets, researchers will be first able to find patterns which can be applied to create a sufficient hypotheses about disease mechanisms. This way knowledge can be generated before the involvement of the laboratory, while at the same time laboratory tests will be more purposeful, since there will be priory knowledge.

A recent analysis of gene expression data addressed the prediction of recurrence risk in patients with clear cell Renal Cell Carcinoma (ccRCC) [5]. This study focused on the data available for tumor samples. An additional, preliminary analysis of tumor samples vs. matched healthy control samples showed the surprising result that a relatively simple classifier achieves nearly perfect separation of the two classes when applied to a randomly selected subset of, e.g., 80 genes (out of the 20532 genes in the dataset). While this result seems favorable in view of reliable diagnosis of ccRCC, the finding that random subsets of genes are discriminative complicate the search for genes that are relevant for disease mechanism (and not just correlated with its presence).

In this project, a more detailed study of the classification problem "normal cells vs. tumor samples" is to be performed. Generalized Matrix Relevance Learning Vector Quantization (GMLVQ) [7] as a classification will serve as an example classifier in order to address the following research questions:

How does the classification accuracy (error rates, AUC of ROC characteristics etc.) depend on the size of the randomly selected subset of genes? Can a characteristic size of the subset be determined below which the accuracy deteriorates?

b) In each of the randomly generated subsets, relevance learning can be used to identify the genes in the subset which are highly predictive for the disease status. By performing the training process on a large number of randomized gene panels, can we identify a reasonably small panel of most relevant genes?

2 DATA AND METHODS

2.1 DATA

We used a part of clear cell Renal Cell Carcinoma (ccRCC) data set from the TCGA [2] repository which contains accurate sequence and expression of genes. We used this data set to develop and test our methods that were implemented. In this data set an expression for 20532 genes for 130 patients is included, where half of the patients are healthy and half of them are not.

As we mentioned before, we tried to find a small subset of the 20532 genes that are significant on classifying whether a patient is sick or not. For classification tool we used a General Matrix Learning Vector Quantization (GMLVQ) [7] classifier that we trained it each time with a subset of genes. At the next section we firstly introduce the simple Learning Vector Quantization (LVQ) and then we introduce the GMLVQ.

2.2 LEARNING VECTOR QUANTIZATION

Kohonen in [4] introduces the supervised classification method LVQ. Until today the method is widely used and a variety of modifications of Kohonen's algorithm have been proposed. The resulted classifier of this method consists of labeled prototypes which represent the set of classes, and a distance measure. The prototypes lie at the same space as the input data and the distance metric can vary between many different such as: Euclidean, city block etc, according to the needs of the designer. To classify a new sample that its label is unknown, the classifier uses a Nearest Prototype Classification (NPC), where the sample is labeled with the same label as the closest prototype has (winner-takes-all decision).

LVQ algorithms are used to determine the points where each prototype lies. To decide on these points a training process takes place and is based on a set of known samples $X = \{(\xi_i, y_i)|\mathbb{R}^n \times \{1, ..., C\}\}$, called the training set, where $\mathbb{R}^n$ is the input data space and $\{1, ..., C\}$ is the set of classes.

In every iteration a random sample (ξ, y) (where y is the class of the sample) of the training set is chosen and the prototype that has the minimum distance from this sample is updated. If the winning prototype has the same label with the sample then it will move closer to the sample, if not it will drift away. The update is done according to:

$$w_L = w_L + \alpha(\xi - w_L), \qquad if\ (c(w_L) = y), \tag{2.1}$$

$$w_L = w_L - \alpha(\xi - w_L), \qquad if\ (c(w_L) \neq y), \tag{2.2}$$

where α is the well-known learning rate.

2.2.1 *Generalized LVQ*

Generalized Learning Vector Quantization (GLVQ) was proposed by Sato and Yamada [6]. This algorithm is a modification of LVQ which is based on an heuristic cost function. Assume that w_k and w_J are the prototypes with the minimum distance from the sample (ξ, y), and $c(w_J) = y$ and $c(w_K) \neq y$. Then the GLVQ cost function has the form:

$$E_{GLVQ} = \sum_{i=1}^{P} H(\mu_i), \qquad where \quad \mu_i = \frac{d_J(\xi_i) - d_K(\xi_i)}{d_J(\xi_i) + d_K(\xi_i)}, \tag{2.3}$$

where $d_J(\xi_i)$ and $d_K(\xi_i)$ are the distances between the sample and the correct and incorrect prototype respectively. and factor μ is the relative difference distance. H is a monotonically increasing function and the goal of the training procedure is to minimize E_{GLVQ} according to the model's parameters. In each time step the closest prototype that has the same label with the training sample moves towards the sample and the closest prototype that does not have the same label moves away from the sample.

2.2.2 *Matrix Learning LVQ*

The most common LVQ methods are preferred due to their robustness. These methods, although suffer from the "curse" of dimensionality. At high dimensional space distance becomes more and more meaningless and the isotropic clusters that we assume exist in Euclidean space lose their meaning and they become more and more vague [7]. Because our goal is to use a classifier on high dimensional data of 20532 genes a more general metric tool would be more useful.

2.2.2.1 *Advanced distance measure*

A more generalized distance measure has the form:

$$d^{\Lambda}(\xi, w) = (\xi - w)^T \Lambda (\xi - w), \tag{2.4}$$

where Λ is square matrix that can carry information about correlations between features. The above mentioned similarity measure defines a squared Euclidean distance if and only if is symmetric and positive definite. If this is the case then Equation 2.4 can be written as:

$$d^{\Lambda}(\xi, w) = [(\xi - w)^T \Omega^T][\Omega(\xi - w)] = [\Omega(\xi - w)]^2, \quad (2.5)$$

Finally the method we used is the GMLVQ method that uses this metric measure.

2.2.2.2 *Generalized Matrix Learning Vector Quantization*

In order to extend the GLVQ to the GMLVQ we simple replace the distance in Equation 2.3 by the distance mentioned in Equation 2.4, hence,

$$E_{GMLVQ} = \sum_{i=1}^{P} H(\mu_i^{\Lambda}), \qquad where \quad \mu_i^{\Lambda} = \frac{d_J^{\Lambda}(\xi_i) - d_K^{\Lambda}(\xi_i)}{d_J^{\Lambda}(\xi_i) + d_K^{\Lambda}(\xi_i)}, \quad (2.6)$$

where $d_J^{\Lambda}(\xi_i)$ and $d_K^{\Lambda}(\xi_i)$ are the distances between the sample and the correct and incorrect prototype respectively. To form the updates of the GMLVQ we calculate the derivatives of Equation 2.6 in respect with w_K, w_J, and Ω_{lm}. We can observe the derivatives at the following equations:

$$\Delta w_J = \alpha_1 \cdot H'(\mu^{\Lambda}(\xi)) \cdot \mu^{\Lambda}(\xi) \cdot \Lambda \cdot (\xi - w_J), \quad (2.7)$$

$$\Delta w_K = \alpha_1 \cdot H'(\mu^{\Lambda}(\xi)) \cdot \mu^{\Lambda}(\xi) \cdot \Lambda \cdot (\xi - w_K), \quad (2.8)$$

$$\begin{aligned} \Delta \Omega_{lm} = \alpha_2 \cdot H'(\mu^{\Lambda}(\xi)) \cdot [\mu_J^{\Lambda}(\xi) \cdot ((\xi_m - w_{Jm})(\Omega(\xi - w_J))_l \\ - \mu_K^{\Lambda}(\xi) \cdot ((\xi_m - w_{Jm})(\Omega(\xi - w_K))_l]. \end{aligned} \quad (2.9)$$

These updates are the standard Herb terms, where the closest prototype with the same label as the sample is pulled closer to the sample and the closest prototype that does not have the same label is pushed away from the sample.

3

APPLICATIONS AND CASE STUDIES

In order to test how the classification accuracy varies on different set magnitudes we used a GMLVQ classifier. And we tested its performance on different set sizes.

Because there are 20532 genes there is a very big number of unique sets of each magnitude we wanted to test, so we run 200 times 10-fold cross validation each time on a random set (subset of 20532 genes) of the specific magnitude that we test each time. So in order to check the performance of sets that are consisted of 20 genes we run 200 times 10-fold cross validation each time on a random subset of magnitude 20 and at the end we calculate the average performance using the Area Under the Curve (AUC).

3.1 AUC PERFORMANCES

We experimented on 4 different descending set sizes, $80, 20, 12, 5$, to check when the performance of the classification becomes low and unstable. The performance for each different set size is described below.

3.1.1 *Random sets of size 80*

We run the 10-fold cross validation for 200 random sets of magnitude 80. The average AUC of this run is calculated 0.9926, while the standard deviation of the AUCs is 0.0046 and the interquartile range equals 0.0055. In Figure 3.1 we can observe the histogram of the run.

One can notice that the performance is pretty high for every random set in fact none of the sets performed lower than 98%. This is a very stable performance too.

3.1.2 *Random sets of size 20*

At next we run the 10-fold cross validation for 200 random sets of magnitude 20. The average AUC of this run is 0.9696, while the standard deviation of the performance calculated 0.0165 and the interquartile range equals to 0.0262. In Figure 3.2 we can observe the histogram of the run.

In this experiment the performance is still significantly high and it is stable too, since none of the sets performances dropped down 92%.

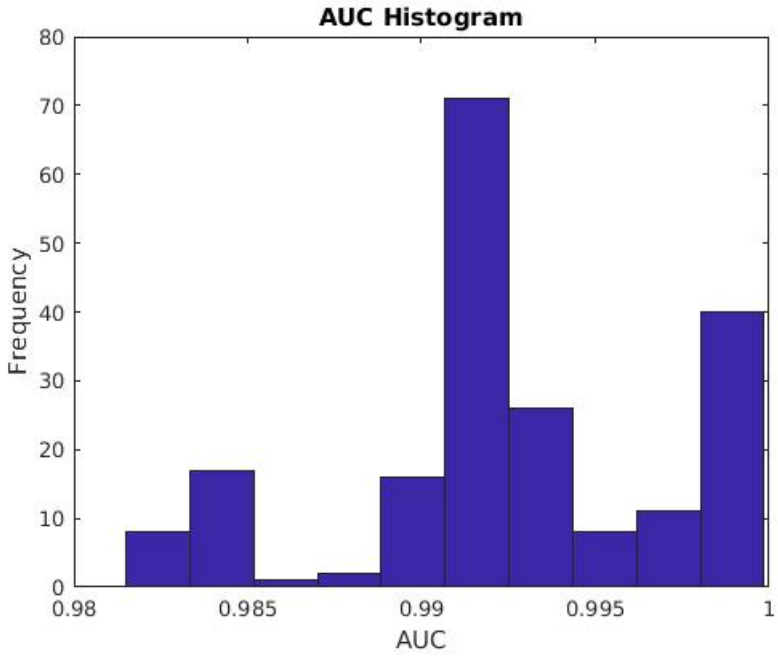

Figure 3.1: Histogram of AUC for 200 hundred random sets of size 80

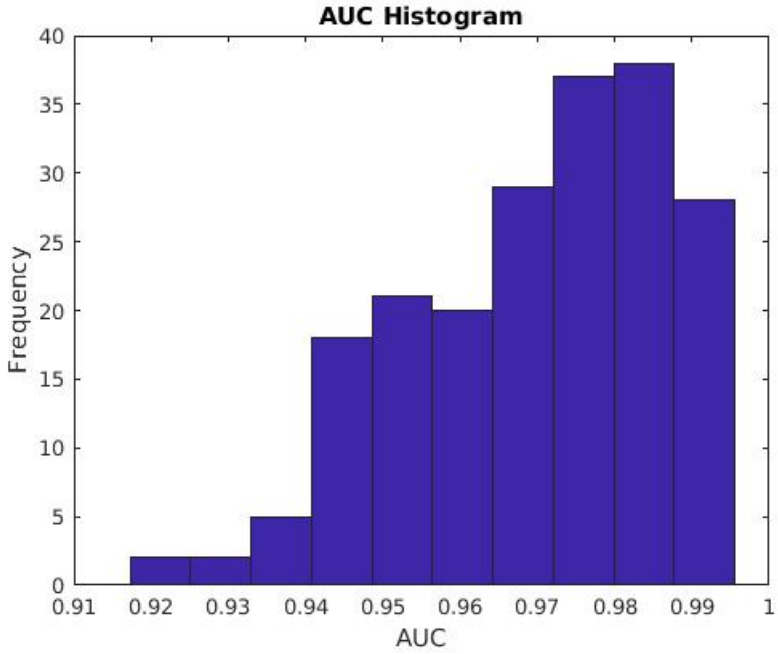

Figure 3.2: Histogram of AUC for 200 hundred random sets of size 20

3.1.3 *Random sets of size 12*

To continue with the search of a break point in performances and stability we experiment on 200 random sets of size 12. The average performance of this run is 0.9691, while the standard deviation of the

performance is 0.207 and the interquartile range calculated 0.0286. In Figure 3.3 we can observe the histogram of the run.

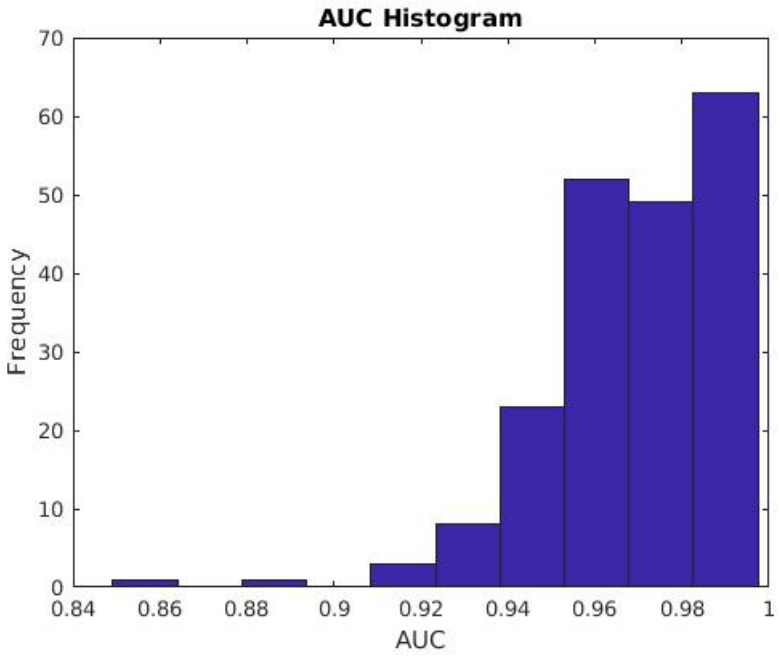

Figure 3.3: Histogram of AUC for 200 hundred random sets of size 12

Again the average performance remains very high and at the same time stable but at this point there is a decrease in stability. For the first time we observe that some performances dropped down from 90% (85%).

3.1.4 *Random sets of size 5*

Again we run a cross validation for 200 random sets but this time of magnitude 5. The performance of this run is estimated to 0.9321, while the standard deviation of the individual performances is 0.0634 and the interquartile range equals to 0,0639. In Figure 3.4 we can observe the histogram of the run.

The average performance of this case study is still very high as the previous ones. We can observe though that there are some performances were the AUC dropped down 60%. In fact a random set of magnitude 5 can not be marked as stable and reliable.

3.2 SEEKING FOR SIGNIFICANT GENES

In the previous section we observed that a rather simple GMLVQ classifier trained on random sets of genes size 80, 20, 12, 5 is able to solve our classification problem with high percentage of succession. This means that the information whether a person is sick or not is spread

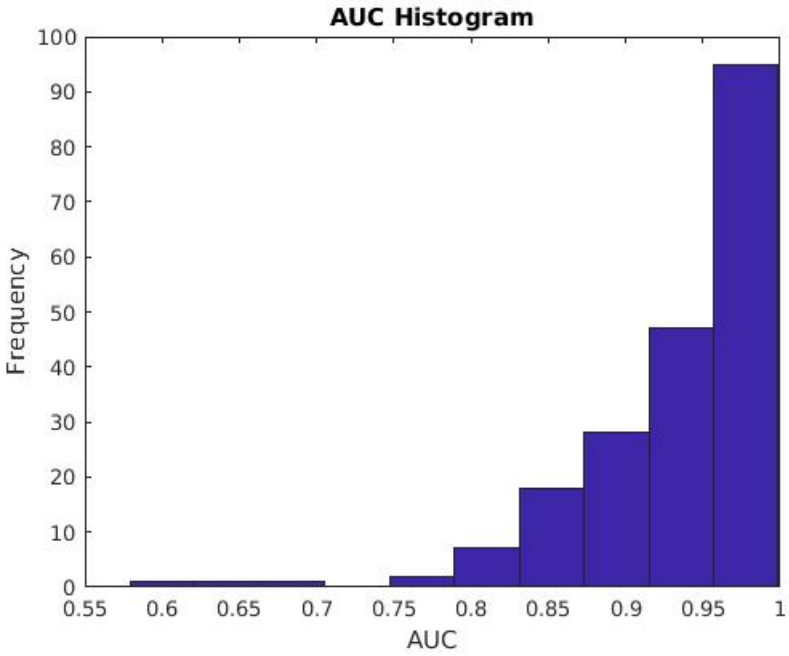

Figure 3.4: Histogram of AUC for 200 hundred random sets of size 5

across the genes. However some subsets of magnitude 5 performed really worse than others and this probably means that even if the information is spread, still is not equally spread.

3.2.1 *T-test*

In order to find significant differences between the healthy and unhealthy samples, so our classifier can be trained better, we apply a t-test. Then we train the classifier with the "best" and "worst" genes as classified by the t-test.

3.2.2 *Performances of set size 5*

At first the t-test is applied to find the 5 "best" and 5 "worst" genes.

3.2.2.1 *The best 5*

We train the classifier on the 5 best genes and the AUC is observed in Figure 3.5. As we can see the AUC calculated equal to 0.99868 which represents a perfect performance.

In addition in Figure 3.6 we can observe useful information about the prototypes of the two different classes and information about the relevance matrix of the GMLVQ.

We can see that the features of the prototypes of the two classes are almost identical negative to each other. This seems the reason that the classification is nearly perfect.

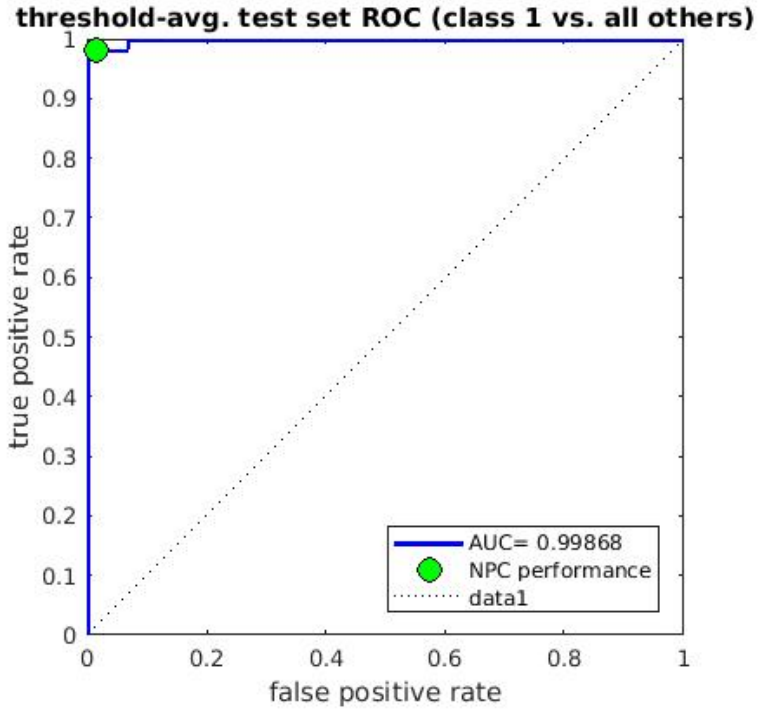

Figure 3.5: AUC when the classifier trained over the best 5 genes

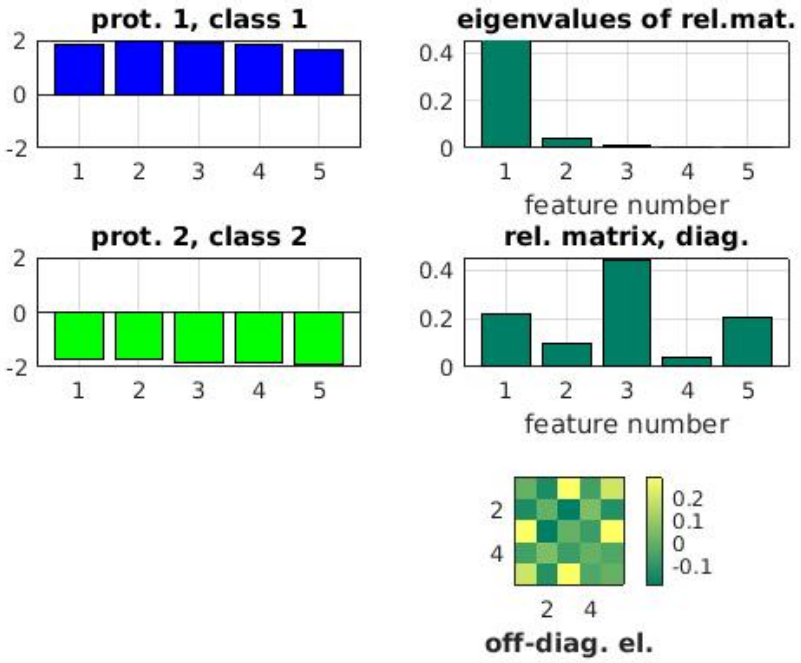

Figure 3.6: Information about prototypes and relevance matrix, when the classifier trained over the best 5 genes

3.2.2.2 *The worst 5*

We train the classifier on the 5 best genes and the AUC is observed in Figure 3.5. As we can see the AUC calculated equal to 0.51886 which represents a perfect performance.

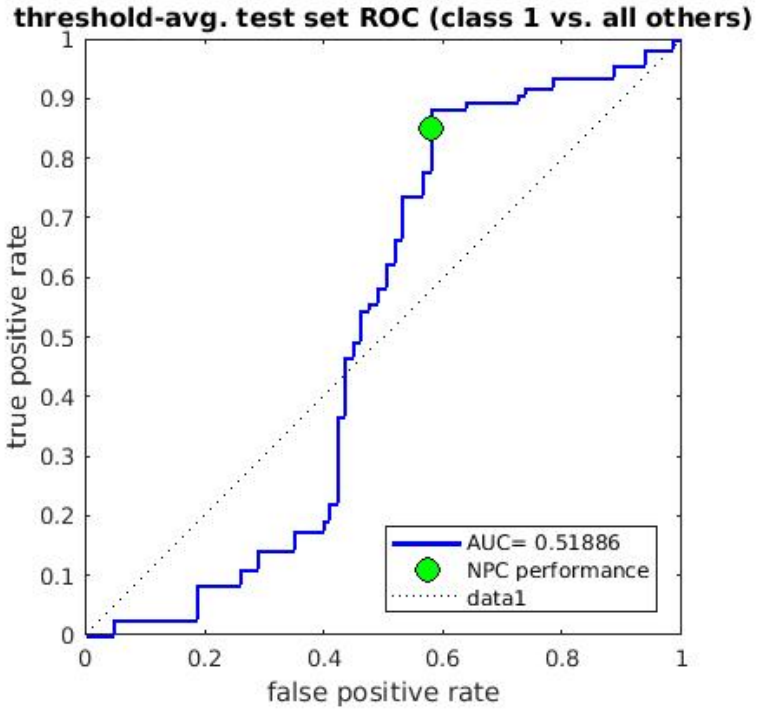

Figure 3.7: AUC when the classifier trained over the worst 5 genes

In addition in Figure 3.8 we can observe useful information about the prototypes of the two different classes and information about the relevance matrix of the GMLVQ.

We can see these time that there are no trivia differences between the features of the prototypes of the two classes.

3.2.3 *Performances of set size 80*

At next we train the classifier with the best 80 and then with the worst 80 genes.

3.2.3.1 *The best 80*

We train the classifier on the 80 best genes and the AUC is observed in Figure 3.9. As we can see the AUC calculated equal to 0.99193 which represents a perfect performance.

In addition in Figure 3.10 we can observe useful information about the prototypes of the two different classes and information about the relevance matrix of the GMLVQ.

We can see again here that the features of the prototypes of the two classes are almost identical negative to each other. This explains the perfect classification.

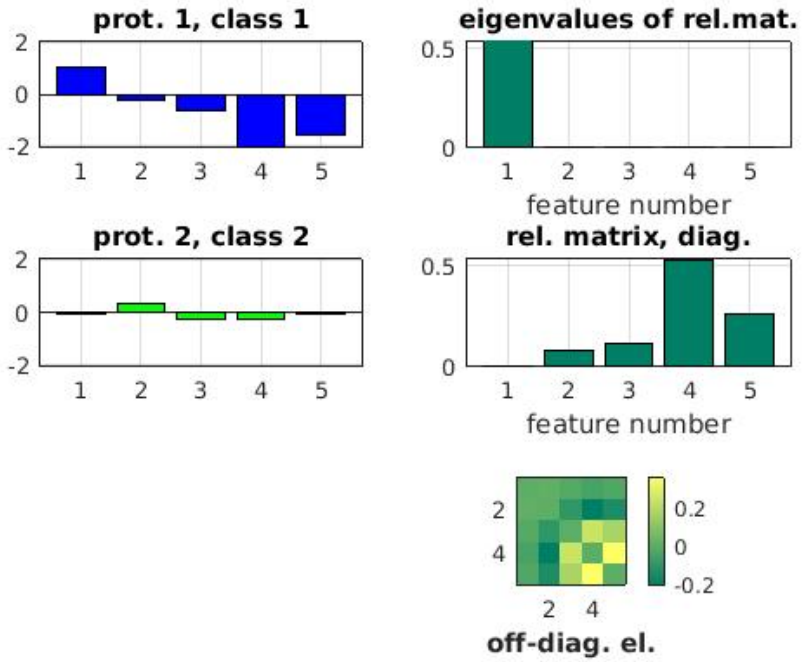

Figure 3.8: Information about prototypes and relevance matrix, when the classifier trained over the worst 5 genes

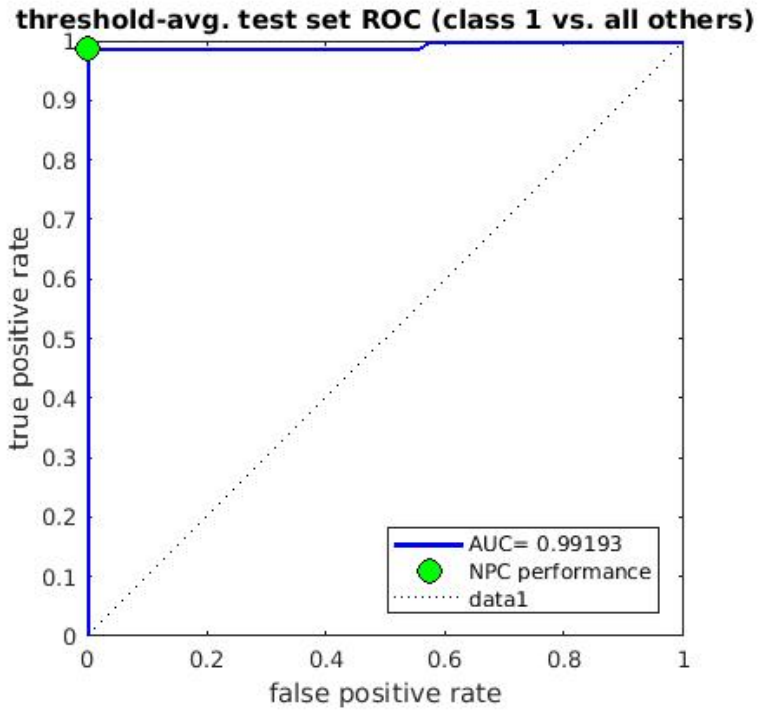

Figure 3.9: AUC when the classifier trained over the best 80 genes

3.2.3.2 *The worst 80*

We train the classifier on the 80 best genes and the AUC is observed in Figure 3.11. As we can see the AUC calculated equal to 0.53265 which represents a perfect performance.

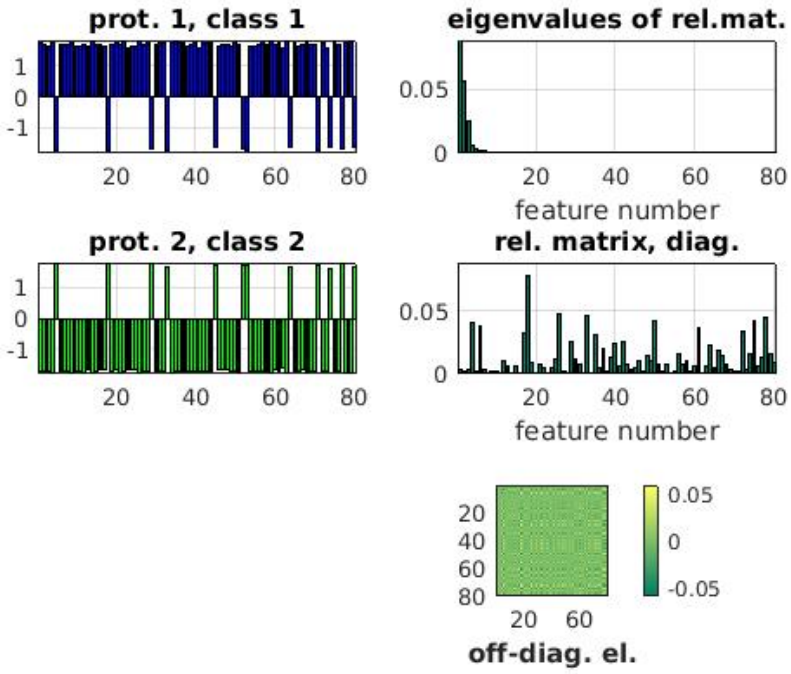

Figure 3.10: Information about prototypes and relevance matrix, when the classifier trained over the best 80 genes

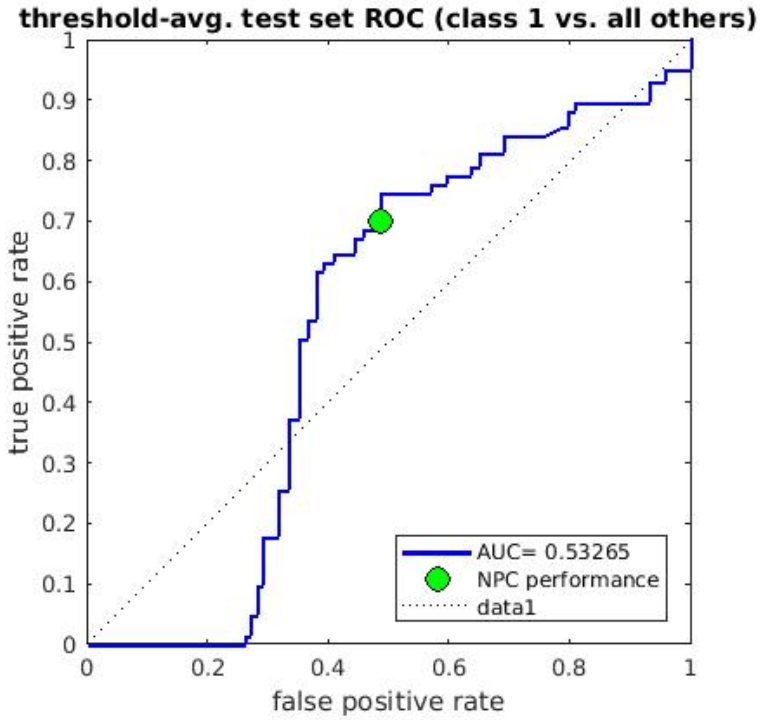

Figure 3.11: AUC when the classifier trained over the worst 80 genes

In addition in Figure 3.12 we can observe useful information about the prototypes of the two different classes and information about the relevance matrix of the GMLVQ.

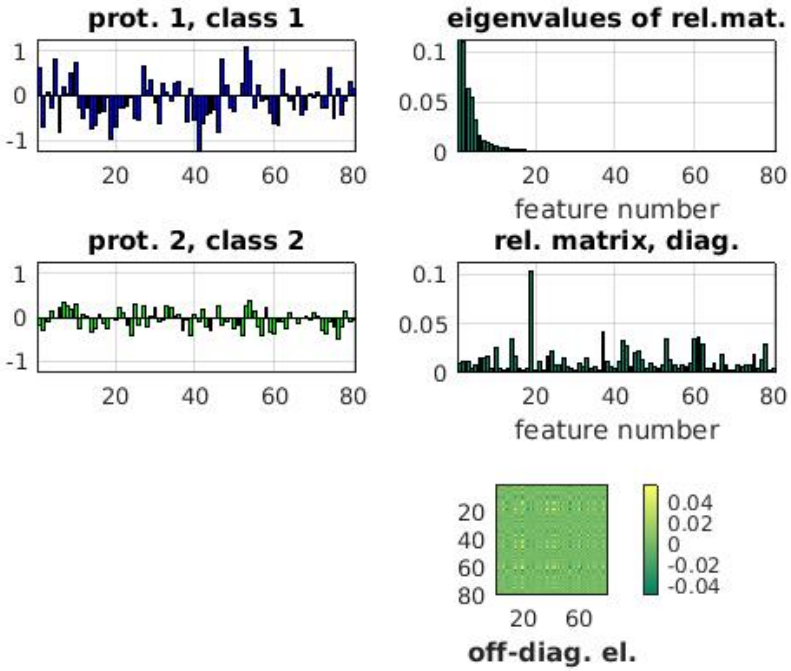

Figure 3.12: Information about prototypes and relevance matrix, when the classifier trained over the worst 80 genes

We can see these time that there are no trivia differences between the features of the prototypes of the two classes.

4 SUMMARY AND CONCLUSION

For this project we tried to study more detailed on the classification problem "normal cells vs. tumor" using a data set of TCGA repository. A GMLVQ classifier used to perform our experiments. We observed that even when the classifier trained on random sets of 5 genes was able to distinguish with high performance a healthy and an unhealthy person.

For this reason we conclude that the information whether someone is healthy or unhealthy is spread among the genes. However, when the classifier trained on some random 5 genes its performance was close to random. Hence, we thought that probably the information maybe is spread among a big number of genes but not all.

Thus, we applied a t-test on the dataset and trained the classifier with the best 5 genes that the test resulted and the worst 5. The performance of the best 5 genes was perfect while the performance of the worst was the random performance. The exact same happened when we tested the best 80 and worst 80 respectively.

Therefore, even if the information is spread among many genes there is still a level of significance.

4.1 FURTHER RESEARCH

Further research could be done on this level of significance. Other methods of feature selection can be used to determine significant genes and maybe will result to a different selection. Methods such as least absolute shrinkage and selection operator (lasso) [8] and boosting [1].

BIBLIOGRAPHY

[1] Jerome H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine." In: *Annals of Statistics* 29 (2000), pp. 1189–1232.

[2] National Cancer Institute and National Human Genome Research Institute. *The Cancer Genome Atlas (TCGA)*. 2015.

[3] Donald E. Knuth. "Computer Programming as an Art." In: *Communications of the ACM* 17.12 (1974), pp. 667–673.

[4] Teuvo Kohonen. "The Handbook of Brain Theory and Neural Networks." In: ed. by Michael A. Arbib. Cambridge, MA, USA: MIT Press, 1998. Chap. Learning Vector Quantization, pp. 537–540. ISBN: 0-262-51102-9. URL: http://dl.acm.org/citation.cfm?id=303568.303833.

[5] Gargi Mukherjee, Gyan Bhanot, Kevin Raines, Srikanth Sastry, Sebastian Doniach, and Michael Biehl. "Predicting recurrence in clear cell Renal Cell Carcinoma: Analysis of TCGA data using outlier analysis and generalized matrix LVQ." In: *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016. DOI: 10.1109/cec.2016.7743855. URL: https://doi.org/10.1109/cec.2016.7743855.

[6] Atsushi Sato and Keiji Yamada. "Generalized Learning Vector Quantization." In: *Proceedings of the 8th International Conference on Neural Information Processing Systems*. NIPS'95. Denver, Colorado: MIT Press, 1995, pp. 423–429. URL: http://dl.acm.org/citation.cfm?id=2998828.2998888.

[7] Petra Schneider. "Advanced methods for prototype-based classification." English. Relation: https://www.rug.nl/ Rights: University of Groningen. PhD thesis. 2010. ISBN: 9789036744058.

[8] Robert Tibshirani. "Regression Shrinkage and Selection Via the Lasso." In: *Journal of the Royal Statistical Society, Series B* 58 (1994), pp. 267–288.

DECLARATION

Copenhagen, August 2018

Antonios Koutounidis

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography *"The Elements of Typographic Style"*.