University of the Aegean

Department of Mathematics, Division of Statistics &
Actuarial-Financial Mathematics

Master Thesis

Combining Multiple Diagnostic Tests for Classification

Author:
Alexia Artemis Baika

Supervisor:
John Tsimikas

October 7, 2019

Committee

Leonidas E. Bantis
Dept. of Biostatistics and Data Science, University of Kansas Medical Center

John Tsimikas, *Thesis Supervisor*
Dept. of Mathematics, University of the Aegean

Spyridon J. Hatjispyros
Dept. of Mathematics, University of the Aegean

Acknowledgments

I would like to thank my thesis advisor Professor Tsimikas of the Department of Mathematics at the University of the Aegean for his patient guidance, support and understanding over the past months.

I am very grateful to Professor Bantis for providing the Liver Cancer dataset enabling me to work with this dataset in this thesis.

I also wish to thank all of the department faculty members for their help and support.

I am particularly gratitude to Michalis for his continued support and encouragement. He experienced all of the ups and downs of my master's studies and I was continually amazed by his patience.

Alice, Achilleas, Dimitra, Dimitris, Georgia, Maria, Nota, Kostas D. Kostas K. and Xristos who provided a much needed form of escape from my studies also deserve thanks for helping me keep things in perspective.

A big thank you goes to Searis for helping me make my dream come true by working in a field I really enjoy and with people who I not only see as colleagues but also as friends.

Finally, I thank my parents, Helga and Nikos, for their love and encouragement, without them I would never have enjoyed so many opportunities. I am especially impressed by my mother who throughout my years of studies has read for her largely meaningless mathematics and listened to countless of methods and algorithms as I was studying.

Combining Multiple Diagnostic Tests for Classification

Alexia Artemis Baika

Department of Mathematics

Division of Statistics & Actuarial-Financial Mathematics

University of the Aegean

2019

ABSTRACT

The data that will be analyzed in this thesis is a Liver Cancer data set which was collected at Shanghai Cang-zheng Hospital in China. The data set includes 145 subjects, of which 54 patients have hepatoma, 39 patients have hepatitis and hepatocirrhosis - chronic liver disease and 52 individuals are healthy. To each subject corespondents 236 Markers. For simplicity, we create two types of groups. The first group consists of healthy individuals and the second group consists of diseased patients.

The goal is to correctly classify a subject in one of the two classes, diseased-non-diseased. For this classification task we use two supervised classification methods: Logistic Regression and Support Vector Machines. Logistic Regression was developed by David Cox in 1958 and is one of the most traditional parametric classification methods. Support Vector Machines was created by Vladimir Vapnik (1995). It can efficiently perform a non-linear classification using what is called the kernel trick, by mapping the inputs into a high-dimensional feature spaces.

In many cases the goal is to develop a model which can explain the relationship between the features and the dependent variable. A severe mathematical problem is when the dimension of the data is greater than the number of the available data points. For this purpose we will describe methods for feature selection and regularization, including subset

selection and lasso.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Liver Cancer Data

The liver is located in the upper right of the abdominal cavity and lies beneath the diaphragm. It is the largest glandular organ. The liver can be divided into two parts: a right lobe and a left lobe. Blood vessels and bile ducts also define the boundaries of eight separate segments. The liver is the only internal organ capable of natural regeneration of lost tissue. As little as 25% of a liver can regenerate into a whole liver. The liver stores a multitude of substances and also produces coagulation factors that are critical to cessation of blood loss from damaged vessels. Due to its wide range of functions the liver is absolutely essential for sustaining life.

Primary liver cancer originates either in the gallbladder [1] or liver. The majority of cancer instances are a result of spreading from other organs. Cancer cells can separate from the tumor and be transported via the bloodstream to the liver where they can attach and establish secondary sites [2]. If the tumor is not too large and remaining liver tissue is healthy, surgical removal is possible. After the excision the liver can regenerate back to full size within a few months. Hepatoma, also known as hepatocellular carcinoma, is a type of primary liver cancer. Common causes of hepatoma include hepatitis. Hepatitis - and hepatocirrhosis - is the inflammation of the liver. Hepatitis and hepatocirrhosis are chronic liver diseases. This means that the scar tissue replaces healthy liver tissue and stops the liver from working normally.

The data that will be analyzed in this thesis are liver cancer data and were collected at Shanghai Cang-zheng Hospital in China. The data includes 145 subjects, of which 54 patients have hepatoma (H) , 39 patients have hepatitis and hepatocirrhosis - chronic liver disease (LD) and 52 individuals are healthy/normal (No). To each subject corresponds a serum. Each serum sample is measured by the Surface-enhanced laser desorption/ionization (SELDI), which is a soft ionization [3] method in mass spectrometry [4] (MS) for its protein

[1] The gallbladder is a small organ that lies beneath the liver.

[2] Secondary site is the place where the cancer has transported and grows.

[3] Ionisation is the process by which an atom acquires a negative or positive charge by gaining or losing electrons to form atoms that have a non-zero net electrical charge.

[4] Mass spectrometry is an analytical technique.

and peptide abundance. The peak detection and the alignment procedures are performed with Ciphergen ProteinChip Software 3.0. The mass range is from 1.000 to 4.000 Da [5]. For all the samples, there are 236 peaks identified. Each peak is a Marker, that means we have 236 Markers for each subject.

For simplicity, we create two types of groups. The first group consists of normal individuals (No) and the second group consists of diseased patients, so the LD and H patients are both in a group denoted as LDH. We demonstrate a scatter plot (see appendix), which at the x axes shows the diseased group (as 1) and the non-diseased group (as 0) and at the y axes shows one of the 236 Markers.

1.2 Training and test set

Our original data set is a sample that represents the population. In order to evaluate the performance of the statistical methods we divide the data into a training set and a test set. The training set is used to fit the model and the test set is used to test the performance of our model.

1.3 Sensitivity and Specificity

In this section we will determine how we can measure the accuracy of a diagnostic method. A test in medicine is considered as diagnostic when it can detect a disease. Thus, it should correctly classify individuals in one of the following categories: "healthy" or "diseased". The outcome of that kind of test can be summarised in a 2×2 table, known as Classification Table. There are two basic concepts related to the accuracy of a test: sensitivity and specificity. These two concepts become more clear through the classification table which can be used for all types of dichotomous tests.

We denote true status as D=1 for disease and D=0 for non-disease. The variable Y is the outcome of the test. We denote Y=1 as the positive for disease outcome of the diagnostic test and Y=0 is the negative outcome.

A classificaton table is depicted in table (1.1).

Note: TP represents the number of the true positive (diseased patients with positive test), FP the false positive (healthy patients with positive test), TN the true negative (healthy patients with negative test) and FN the false negative (diseased patients with negative test)

[5] Dalton (Da) is an unified atomic mass. It is a standard unit of mass that quantifies mass on an atomic scale.

Table 1.1: Classifiation table that summarizes the results of a dichotomous diagnostic test

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	TN	FP	n_0
`Disease, D=1`	FN	TP	n_1
`Total`	m_0	m_1	N

- **True Positive Rate (TPR) or Sensitivity** is the conditional probability of correctly classifying diseased patients.

$$TPR = Sensitivity = P(Y = 1|D = 1) = \frac{TP}{TP + FN} = \frac{TP}{n_1}$$

- **True Negative Rate (TNR), or Specificity** is the conditional probability of correctly classifying healthy patients.

$$TNR = Specificity = P(Y = 0|D = 0) = \frac{TN}{TN + FP} = \frac{TN}{n_0}$$

- **False Negative Rate (FNR), or 1-sensitivity** is the conditional probability of falsely classifying diseased patients.

$$FNR = 1 - Sensitivity = P(Y = 0|D = 1) = \frac{FN}{FN + TP} = \frac{FN}{n_1}$$

- **False Positive Rate (FPR), or 1-specificity** is the conditional probability of falsely classifying healthy patients.

$$FPR = 1 - Specificity = P(Y = 1|D = 0) = \frac{FP}{FP + TN} = \frac{FP}{n_0}$$

The result Y can either be binary or quantitative. In the case of a quantitative outcome we have to define a cut-off value c to separate patients into two groups. The one group is classified with the condition $Y \geq c$ (diseased group) and the second if $Y < c$ (healthy group). Note that different cut-off values have different classification tables and sensitivity-specificity strongly depend on the chosen cut-off value. As c increase, sensitivity decrease but specificity increases and vice versa when c decrease.

Example

In the next example we will work with a quantitative outcome Y. Let's say that we have two populations, healthy and disease and each follows a normal distribution. Suppose that the

healthy one follows the standard normal distribution and the disease population follows the normal distribution with mean 2 and variance 1. In the figure (1.1) the blue curve is the healthy population distribution while with red is the disease population distribution. The chosen cut-off line is set to point 0. From the graph we can observe how sensitivity and specificity depend from the cut-off line. Sensitivity or the True Positive fraction is the area under the red curve, from 0 to 6. 1-Specificity or False Negative fraction is the area under the blue curve from -4 to 0.

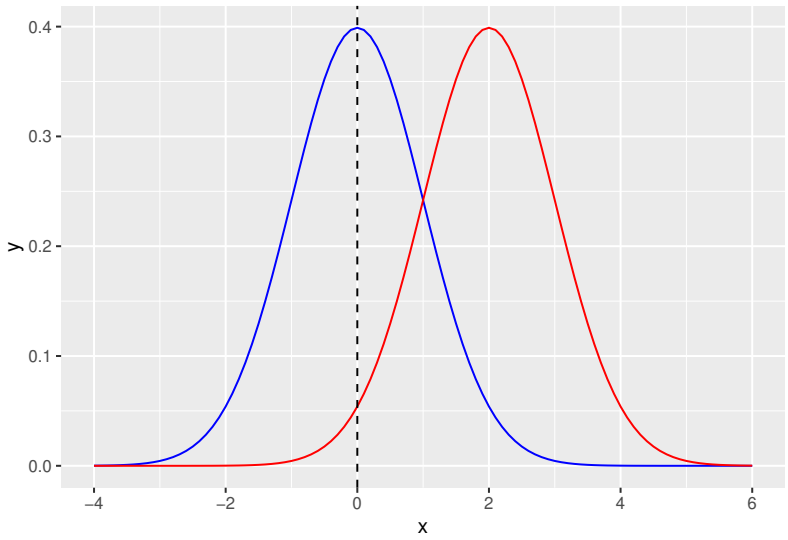


Fig. 1.1: Relation between cut off line and Sensitivity-Specificity

1.4 ROC curve

The Receiver Operating Characteristic Curve (ROC) was developed in 1950 for signal detection. 20 years later it was used to evaluate the performance of any kind of diagnostic tests, consequently the accuracy of statistical models that divide populations into two classes.

A ROC curve is a graphical plot that demonstrates the capability of a binary classifier ,while its threshold is varied from 0 to 1. By taking into account all possible values of the cut-off point c, which is in fact a decision point, the ROC curve can be constructed as a plot of sensitivity (TP) versus 1-specificity (FP). As we saw in the previous example for any cut off point c, we define:

$$TPF(c) = P(Y \geqslant c|D = 1)$$
$$FPF(c) = P(Y \geqslant c|D = 0)$$

Thus the ROC curve is

$$ROC(\cdot) = \{FPF(c), TPF(c), c \in (0,1)\}$$

The Area under the curve (AUC) can take values from 0 to 1. The AUC is equal to 1 if the ROC curve consists of the line (0,0)-(0,1) and the line (0,1)-(1,1). This ROC curve represents a perfect diagnostic method. Contrary, if the AUC is equal to 0, we have a perfect inaccurate diagnostic method. In the event that AUC is equal with the line (0,0)-(1,1), (resulting AUC $= 0.5$) the diagnostic method has no diagnostic value, or equivalently, the decision if a person is healthy or not is taken by flipping a coin.

1.5 Example

From our liver data set, we select Marker 197 and Marker 154. For simplicity we use only one explanatory variable, X. We fit a univariate logistic regression model [6] for each Marker to obtain the probability of each subject to be diseased and, based on this probability we classify each subject as healthy or diseased.

Marker 197
In the scatter plot bellow (figure 1.2) we denote the diseased subjects as 1 and the healthy subjects as 0. We can see that higher values of Marker 197 correspond to the diseased group. The mean of the diseased group is 4.58 and the mean of the healthy group is 1.11.

The logistic regression model is the following:

$$\hat{p}(X) = \frac{exp\{-4.0955 + 2.5996X\}}{1 + exp\{-4.0955 + 2.5996X\},} \tag{1.1}$$

which gives the probability of a subject to be diseased, given the known value of X.
Equivalently, equation (1.1) can be written as:

$$\frac{\hat{p}(X)}{1 - \hat{p}(X)} = exp\{-4.0955 + 2.5996X\} \tag{1.2}$$

$$log\left(\frac{\hat{p}(X)}{1 - p(X)}\right) = -4.0955 + 2.5996X \tag{1.3}$$

The left-hand side of the equation (1.3) is called logit and it is linear in X. This logit model gives values from $-\inf$ to $\inf$, where values which are less than 0 are related with probabilities less than 0.5 (diseased).

For each subject we give the true disease Status (1 denotes disease, 0 denotes healthy), the Marker value (for Marker 197), together with the probabilities from the equation (1.1) and

[6] for more details, see chapter 2

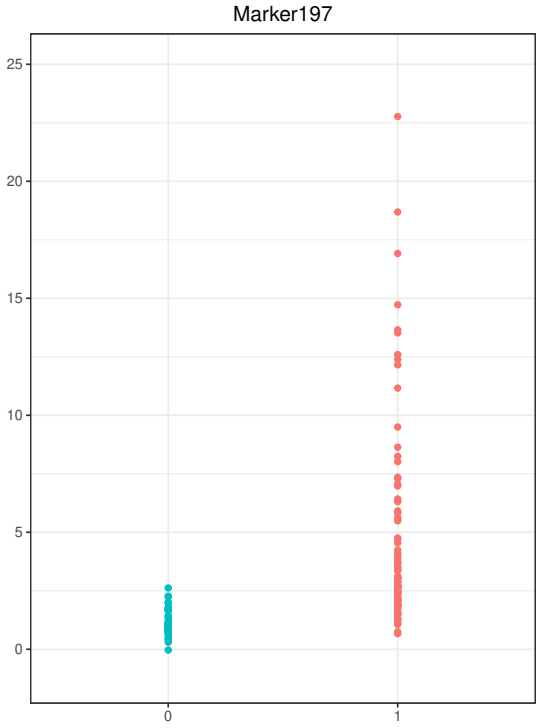


Fig. 1.2: Scatter Plot for Marker 197

the score values from the equation (1.3), in table 5.1 (see appendix).

We can plot each of the probabilities of disease given the Marker 197 (see Figure 1.3). We denote 1 (red points) for disease and 0 (black points) for healthy.

We set a cut off point at 0.5 on probability scale. Every subject that has a higher probability of 0.5 is classified as diseased and every subject that has a lower probability of 0.5 is classified as healthy. Taking into account their true diseased status and the probability of disease we construct the classification table.

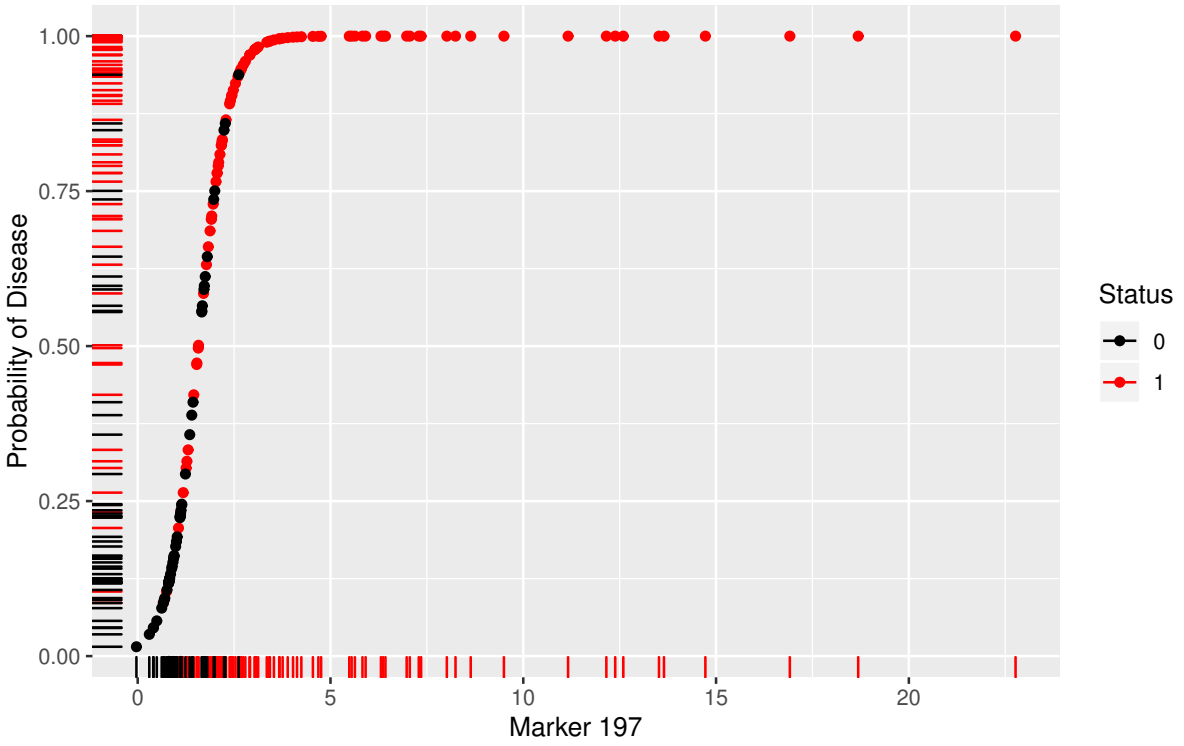


Fig. 1.3: Predicted probabilities of disease using logistic regression

Table 1.2: Classification Table using logistic regression for Marker 197

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	40	12	52
`Disease, D=1`	12	81	93
`Total`	52	93	145

Here we see that 81 subjects of diseased people are correctly classified as diseased, 12 subjects are incorrectly classified as non-diseased, 40 subjects are correctly classified as non-diseased and 12 are incorrect classified as diseased.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{81}{93} = 0.87$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{12}{52} = 0.23$
- The True Negative Rate is equal with: $TNR = \frac{40}{52} = 0.76$

- The False Negative Rate is equal with: $FNR = \frac{12}{93} = 0.12$

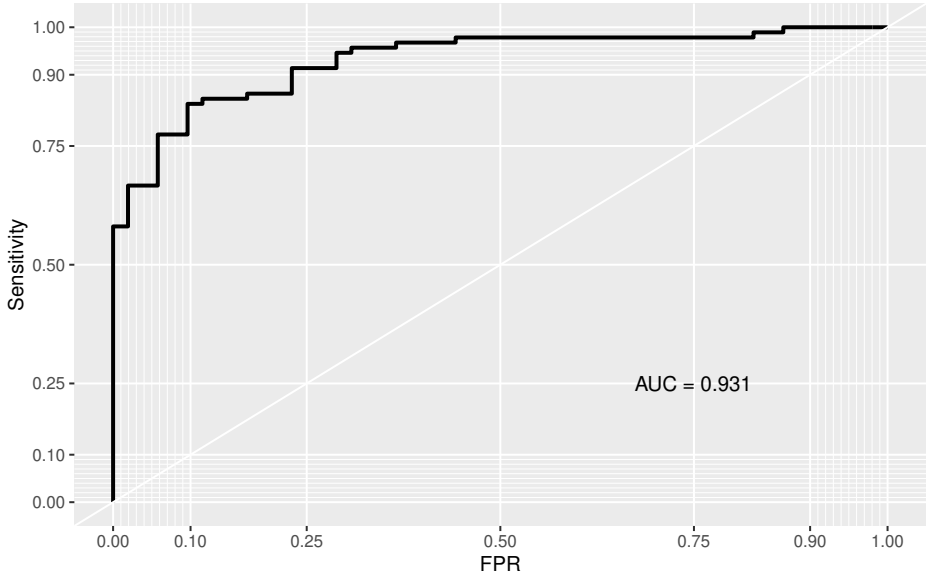


Fig. 1.4: ROC Curve for Marker 197

In this example we set a cut of value at 0.5. From the ROC curve, we see for all cut off values the relationship between sensitivity and FPR. For False Positive Rate (or 1-specificity) 23% we have 87% sensitivity. This means that for FPR=0.23, the ability to detect a diseased subject is 87%, or in other words 87% of the diseased subjects are classified correctly by using this model. The area under the ROC curve (AUC) is equal with 0.931.

Marker 154

In the scatter plot bellow we denote the subjects who are diseased as 1 and the subjects who are healthy as 0. We can see that lower values of Marker 154 are more likely to correspond to the diseased group. The mean of the diseased group is 14.48 and the mean of the healthy group is 23.18.

The logistic regression model is:

$$\hat{p}(X) = \frac{exp\{2.939 - 0.126X\}}{1 + exp\{2.9393 - 0.1267X\}} \tag{1.4}$$

and the logit transformation is:

$$log\left(\frac{\hat{p}(X)}{1 - \hat{p}(X)}\right) = 2.939 - 0.126X \tag{1.5}$$

where X is a vector with the values of Marker 152.
From equation (1.4) and (1.5) we obtain the probabilities of each subject to be diseased and

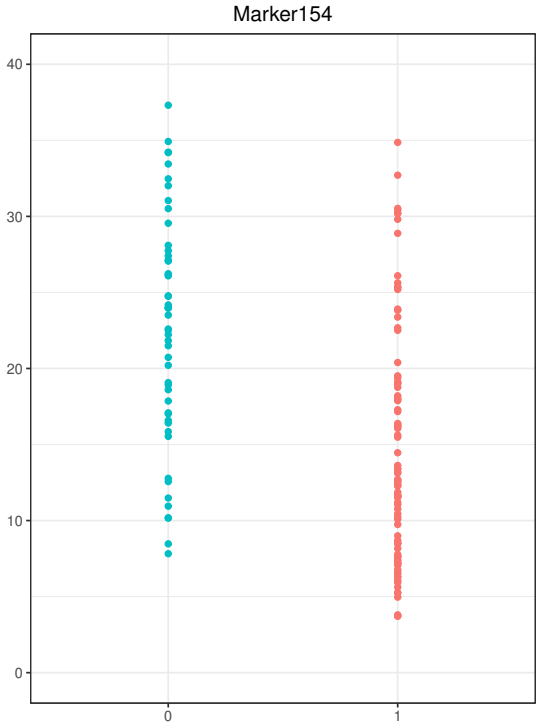


Fig. 1.5: Scatter Plot for Marker 154

the score values (see appendix).

We see that $\hat{b}_1 = -0.126$. This indicates that an increase in the value of Marker 154 is associated with a dicrease in the probability of disease. To be precise, a one-unit increase in Marker 154 is associated with a decrease in the log odd of disease by -0.126 units.

We can plot each of the probabilities of diseased given the Marker 154 (see Figure 1.6). We denote 1 for disease and 0 for healthy.

From the plot we can see that higher values of Marker 154 have in average a lower probability of disease. In figure 1.6 there are many overlaps of the probabilities from the diseased and healthy subjects. We will use two different cut off values and discus their classification performance.

Cut-off value at 0.6 on probability scale.

Taking into account the true diseased status and the probability of disease given Marker 154 we construct the classification table based on the classification rule on probability scale 0.6.

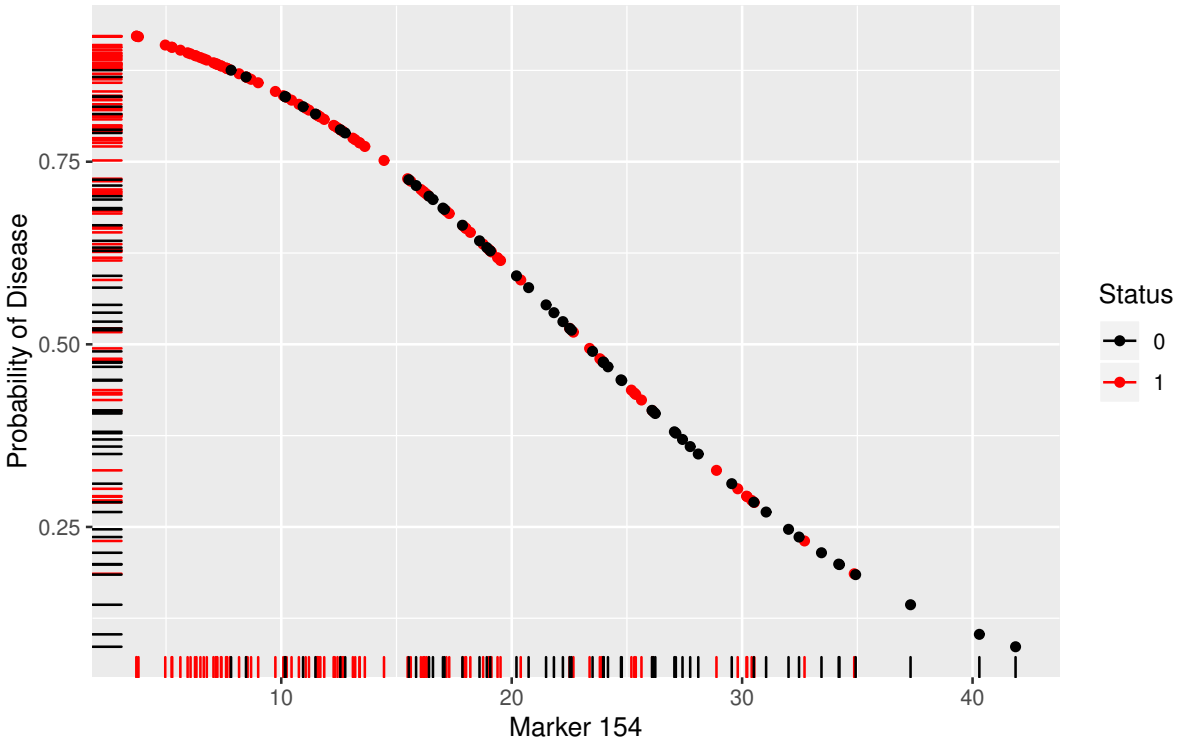


Fig. 1.6: Predicted probabilities of disease using logistic regression

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	34	18	52
`Disease, D=1`	19	74	93
`Total`	53	92	145

Table 1.3: Classification Table using logistic regression for Marker 154 with cut off value at 0.6 on probability scale

Here we see that 74 subjects of diseased people are correctly classified as diseased, 19 subjects are incorrectly classified as non-diseased, 34 subjects are correctly classified as non-diseased and 18 are incorrect classified as diseased.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{74}{93} = 0.79$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{18}{52} = 0.34$

Cut-off value at 0.5 on probability scale
We set the cut off point at 0.5. The classification table is:

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	27	25	52
`Disease, D=1`	16	77	93
`Total`	43	102	145

Table 1.4: Classification Table using logistic regression for Marker 154 with cut off value at 0.5 on probability scale

with corresponding Sensitivity and FPR:

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{77}{93} = 0.82$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{25}{52} = 0.48$

A good model has high sensitivity and high specificity (or low FPR).

- For a cut off value on probability scale at 0.6 we have
  $(sensitivity, specificity) = (sensitivity, 1-FPR) = (0.79, 1-0.34) = (0.79, 0.66)$
- For a cut off value on probability scale at 0.5 we have
  $(sensitivity, specificity) = (sensitivity, 1-FPR) = (0.82, 1-0.48) = (0.82, 0.52)$

For the cut off value 0.6 the model is more specific and less sensitive than the model with cut off value 0.5. Thus the first model has a better ability to correctly detect subjects which are diseased making less false positives and, the second model has a higher probability to detect diseased subjects but by making more false positives. Depending on how specific and sensitive we want our model to be we choose the cut-off value. The choice of the cut-off value is also associated with how expensive the test is and how a false positive result can harm the subject we choose the cut-of value.

As we saw before an increase of the Marker 154 value is associated with a decreased probability of disease. This type of relationship affects the ROC curve which is below the line (0,0)-(1,1).

On every Marker in our liver data set we use a logistic regression model, in which we try to predict if a subject is diseased or not diseased, considering only one Marker. For each Marker we created a ROC curve (see Appendix for ROC curves).

From each ROC graph we observe the Area Under the ROC Curve and choose the 10 highest AUC values were:

$$AUC_i = max\{AUC_i, 1 - AUC_i\}, \quad i \in 1, ..., 236$$

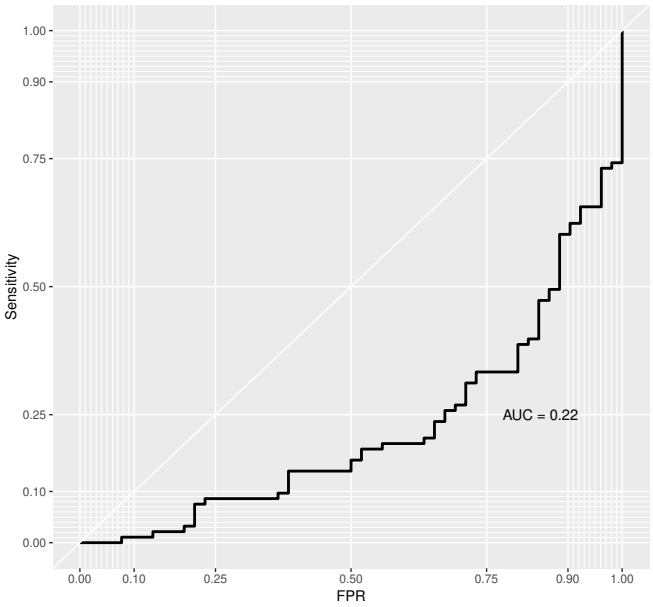


Fig. 1.7: ROC Curve for Marker 154 with AUC=0.22

The 10 highest values of the AUC_i corresponds to the Markers:

197, 188, 189, 196, 193, 187, 192, 190, 205, 206

with corresponding AUC:

$AUC_i = (0.931, 0.897, 0.892, 0.891, 0.882, 0.881, 0.869, 0.864, 0.864, 0.860)$.

The correlation matrix between the 10 Markers is given below. We observe that many markers are highly correlated. For example markers 205 and 206 have correlation of 0.9 and Markers 190, 187 have correlation of 0.96.

	Marker 206	Marker 205	Marker 190	Marker 192	Marker 187	Marker 193	Marker 196	Marker 189	Marker 188	Marker 197
Marker 206	1	0.903	-0.227	-0.175	-0.214	-0.199	0.474	-0.229	-0.218	-0.365
Marker 205		1	-0.207	-0.167	-0.199	-0.183	-0.382	-0.205	-0.202	-0.331
Marker 190			1	0.848	0.962	0.873	0.378	0.917	0.909	0.868
Marker 192				1	0.880	0.941	0.233	0.833	0.918	0.897
Marker 187					1	0.839	0.275	0.856	0.900	0.841
Marker 193						1	0.352	0.940	0.942	0.903
Marker 196							1	0.411	0.336	0.521
Marker 189								1	0.950	0.855
Marker 188									1	0.881
Marker 197										1

Table 1.5: Correlation matrix for 10 Markers

We choose the first two Markers, Marker 197 and Marker 188. From the correlation matrix we see that the correlation between them is 0.88.

Before we analyze the multiple logistic regression model for two variables Marker 197 and Marker 188 we can see how well performs each one separately.

Marker 197

We analyzed Marker 197 in a previous example. We saw that the using a logistic model the AUC was equal to 0.931.

Marker 188

In the scatter plot bellow we denote the subjects who are diseased as 1 and the subjects who are healthy as 0. We can see that higher values of Marker 188 correspond to the diseased group. The mean of the diseased group is 8.48 and the mean of the healthy group is 0.63.

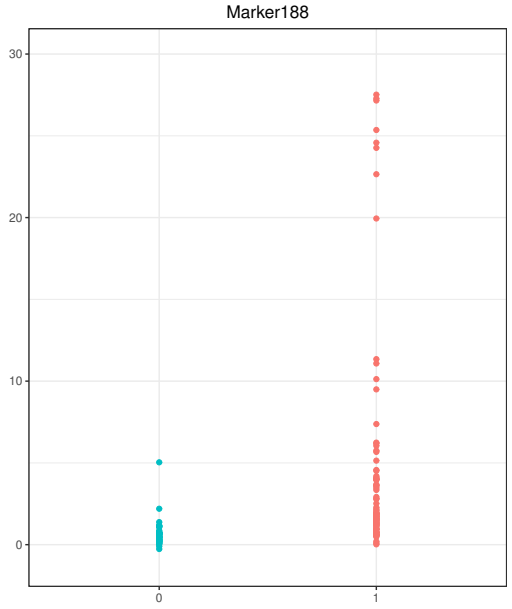


Fig. 1.8: Scatter Plot for Marker 188

The logistic regression model is:

$$\hat{p}(X) = \frac{exp\{-1.59 + 1.77X\}}{1 + exp\{-1.59 + 1.77X\}} \tag{1.6}$$

where X is a vector with the values of Marker 188.
From this model we can obtain the probability for each subject to be diseased.
Equivalently, equation (1.6) can be written as:

$$\frac{\hat{p}(X)}{1 - \hat{p}(X)} = exp\{-1.59 + 1.77X\} \tag{1.7}$$

$$log\left(\frac{\hat{p}(X)}{1 - \hat{p}(X)}\right) = -1.59 + 1.77X \tag{1.8}$$

We plot each of the probabilities of disease given the Marker 188 (see Figure 1.9). We

denote 1 for disease and 0 for healthy.

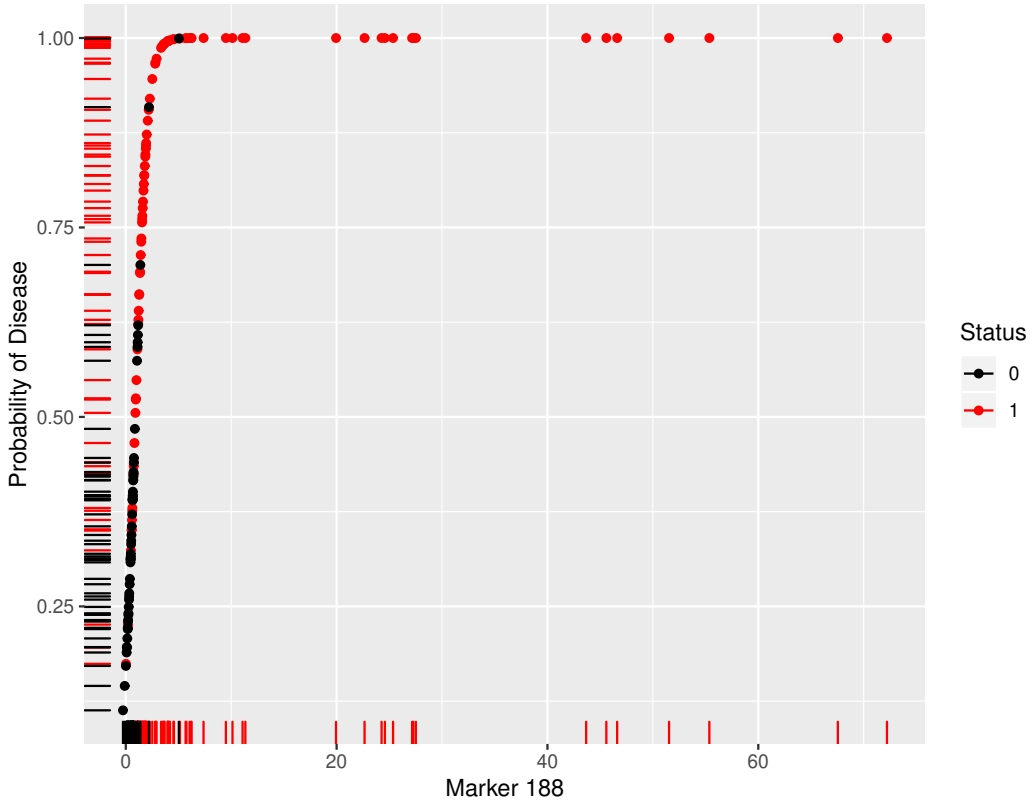


Fig. 1.9: Predicted probabilities of disease using logistic regression

Taking into account their true diseased status and the probability of disease given Marker 188 we create the classification table based on the classification rule on probability scale 0.5.

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	44	8	52
`Disease, D=1`	16	77	93
`Total`	53	92	145

Table 1.6: Classification Table using logistic regression for Marker 188 with cut off value at 0.5 on probability scale

In Table 1.6 the 77 subjects of diseased people are correctly classified as diseased, 16 subjects are incorrectly classified as non-diseased, 44 subjects are correctly classified as non-diseased and 8 are incorrect classified as diseased.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{77}{93} = 0.82$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{8}{52} = 0.15$

The ROC curve is given in figure (1.10)and the AUC is equal with 0.897.

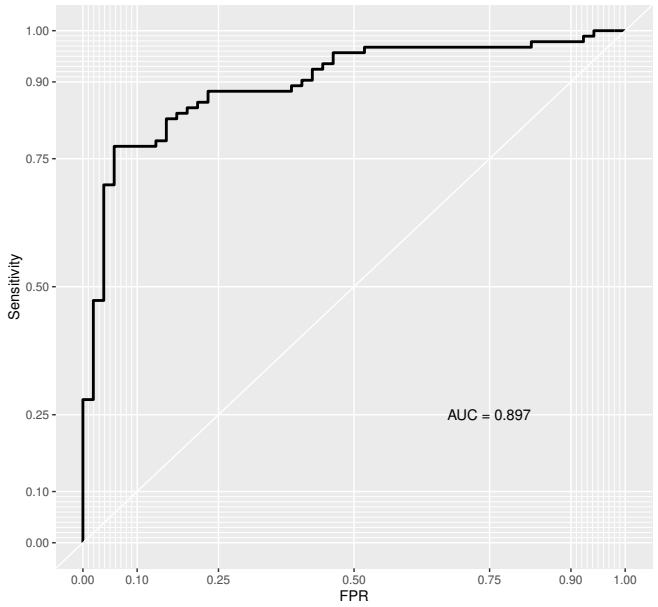


Fig. 1.10: ROC Curve for Marker 188

Here we see that using for example a cut off value with FPR equal with 0.15 (or equivalently a test with specificity 0.75) has a sensitivity up to 0.82. This means that for a false positive rate 0.15, 82% of the subjects are correctly classified as diseased.

Multiple Logistic model for Marker 197 and Marker 188

The logistic regression model that using Marker 197 and Marker 188 as variables is:

$$\hat{p}(X) = \frac{\exp(-4.175 + 2.113X_1 + 0.784X_2)}{1 + \exp(-4.175 + 2.113X_1 + 0.784X_2)} \tag{1.9}$$

Where $X = (X_1, X_2) = (Marker197, Marker188)$.
Both Markers are statistical significant in the presence of the other, with p-values < 0.05. This indicates that using both markers improves the classification. The ROC curve is given and the area under the curve is equal with 0.946. Here we see that using for example a

cut off value with FPR equal with 0.08 (or equivalently a test with specificity 0.92) has a sensitivity up to 0.75. This means that for a false positive rate 0.08, 75% of the subjects are correctly classified as diseased.

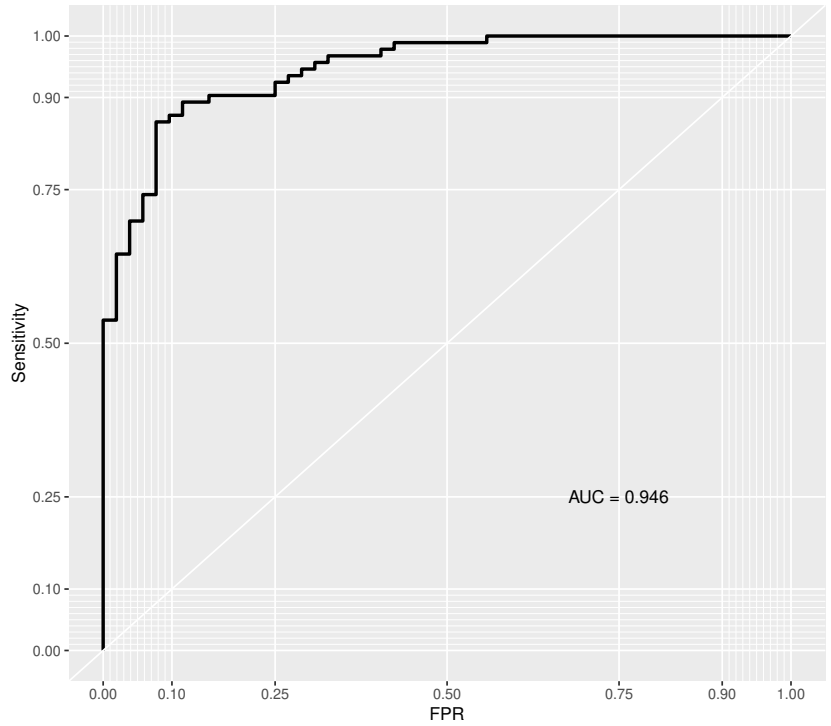


Fig. 1.11: ROC Curve using Marker 197 and Marker 188

The corresponding classification table on the classification rule on probability scale at 0.5 is given below.

Table 1.7: Classification Table on the classification rule on probability scale at 0.5

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	44	8	52
`Disease, D=1`	9	84	93
`Total`	53	92	145

Here we see that 84 patients are correctly classified as diseased, 44 healthy individuals are correctly classified as healthy, 8 healthy individual are incorrectly classified as diseased and 9 patients are incorrectly classified as healthy.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{84}{93} = 0.90$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{8}{52} = 0.15$

Conclusions:

	AUC	Sensitivity	FPR	Specificity
`Marker 197`	0.931	0.87	0.23	0.77
`Marker 188`	0.897	0.82	0.15	0.85
`Marker 197 and 188`	0.946	0.90	0.15	0.85

Table 1.8: Results, based on the classification rule on probability scale at 0.5

In table 1.8 we see the AUC values for marker 197, marker 188 and the AUC value using both markers. Sensitivity, FPR and specificity is given for the same classification rule on probability scale at 0.5. Using both markers the AUC has a higher value and also both Sensitivity and Specificity have equal or higher values, than using each marker separately. Using marker 197 we have a higher value in sensitivity but a lower value in specificity, than using marker 188 which has a lower value in sensitivity and a higher value in specificity.

Logistic Regression for all 10 Markers

We apply a multiple logistic regression model on the liver data set with 10 variables (10 markers) and we want to predict if a subject is diseased or not. The multiple logistic regression model is of the form:

$$p(X) = \frac{\exp(b_0 + b_1X_1 + b_2X_2 + ... + b_{10}X_{10})}{1 + \exp(b_0 + b_1X_1 + b_2X_2 + ... + b_{10}X_{10})} \tag{1.10}$$

Where $X = (X_1, X_2, , ..., X_{10}) = (Marker206, Marker205, ..., Marker197)$.

The ROC curve is given in figure (1.12) and the area under the curve is equal with 0.969.

Here we see that using for example a cut off value with FPR equal with 0.11 (or equivalently a test with specificity 0.89) has a sensitivity up to 0.92. This means that for a false positive rate 0.11, 92% of the subjects are correctly classified as diseased.

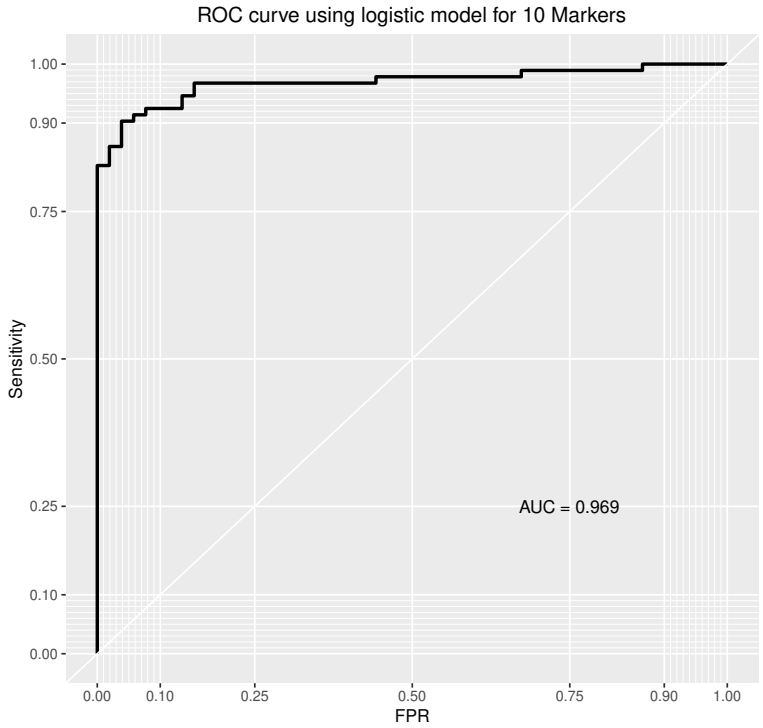


Fig. 1.12: ROC Curve using training set for 10 Markers

The corresponding classification table on the classification rule on probability scale at 0.5 is given below.

Table 1.9: Classification Table on the classification rule on probability scale at 0.5

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	46	6	52
`Disease, D=1`	7	86	93
`Total`	53	92	145

Here we see that 86 patients are correctly classified as diseased, 46 healthy individuals are correctly classified as healthy, 6 healthy individual are incorrectly classified as diseased and 7 patients are incorrectly classified as healthy.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{86}{93} = 0.92$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{6}{52} = 0.11$

Validation

We use the validation method to estimate the test error. We divide the data set into two groups. In the first group we randomly choose 50% of the diseased subjects and 50% of the non-diseased subjects. This is our training set. The remaining 50% and 50% consists the test set. More specifically, the diseased group consists in total of 93 patients and the healthy group of 52 individuals. That means in the training set we have 46 diseased patients and 26 healthy individuals. In the test set we have the remaining 47 diseased patients and the 26 remaining healthy individuals.

We apply a multivariate logistic regression model on the training set with 10 variables and we want to predict if a subject is diseased or not. The multivariate logistic regression model is of the form:

$$p(X) = \frac{\exp(b_0 + b_1X_1 + b_2X_2 + ... + b_{10}X_{10})}{1 + \exp(b_0 + b_1X_1 + b_2X_2 + ... + b_{10}X_{10})} \tag{1.11}$$

Where $X = (X_1, X_2, , ..., X_{10}) = (Marker206, Marker205, ..., Marker197)$ from the training set.

Training set

The ROC curve is given and the area under the curve is equal with 0.987 by using the logistic model and apply it to the training set itself. Here we see that using for example a cut off value with FPR equal with 0.07 (or equivalently a test with specificity 0.93) has a sensitivity up to 0.93. This means that for a false positive rate 0.07, 93% of the subjects are correctly classified as diseased.

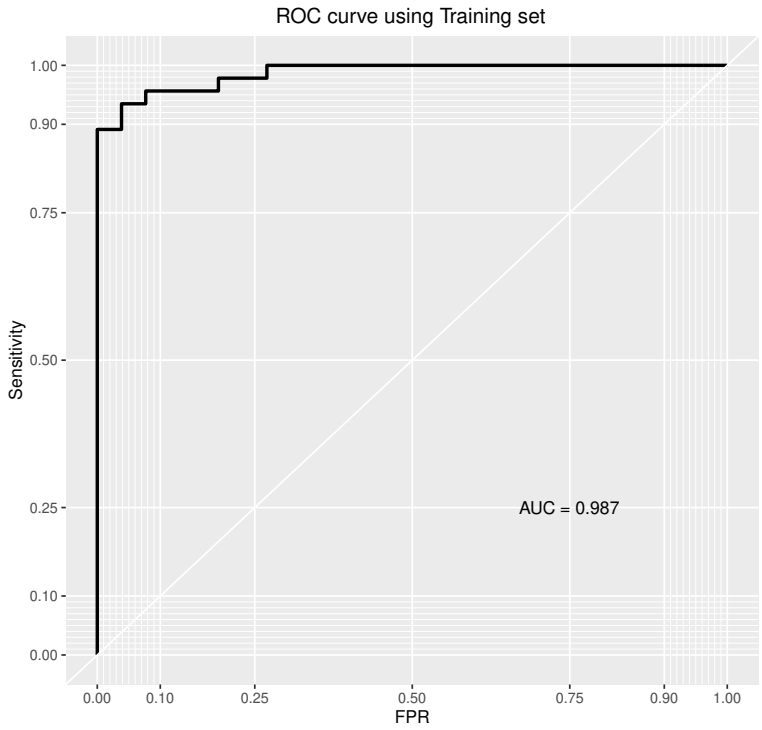


Fig. 1.13: ROC Curve using training set for 10 Markers

The corresponding classification table on the classification rule on probability scale at 0.5 is given below.

Table 1.10: Classification Table on the classification rule on probability scale at 0.5

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	24	2	26
`Disease, D=1`	3	43	46
`Total`	27	45	72

Here we see that 43 patients are correctly classified as diseased, 24 healthy individuals are correctly classified as healthy, 2 healthy individual are incorrectly classified as diseased and 3 patients are incorrectly classified as healthy.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{43}{46} = 0.93$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{2}{26} = 0.07$

The estimated training error is $\frac{3+2}{72} = 0.06$. That is a 6.94% training error rate.

Test set

The ROC curve has AUC = 0.934.

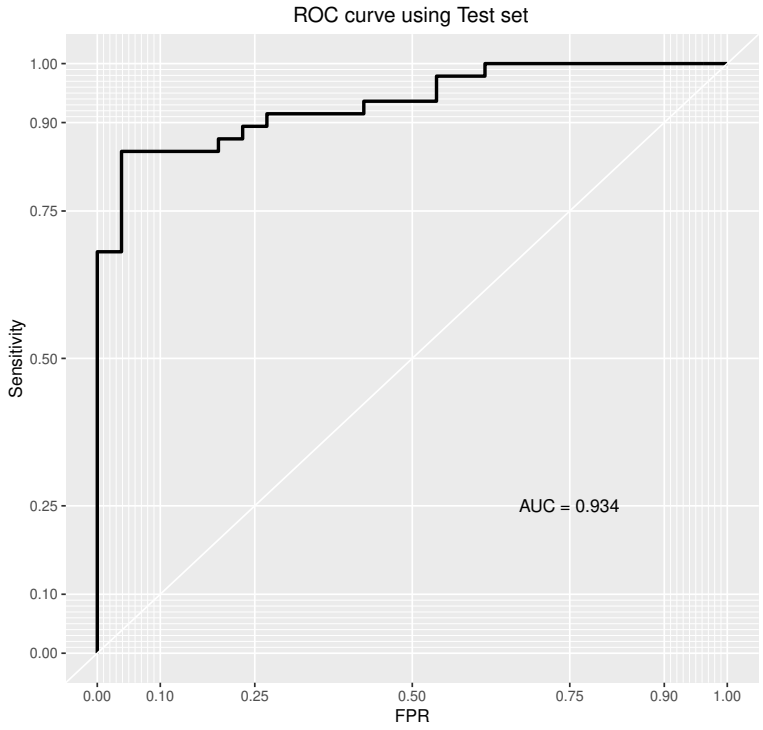


Fig. 1.14: ROC Curve using test set for 10 Markers

The classification table that we observe using a classification rule on probability scale at 0.5 is the following.

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	19	7	26
`Disease, D=1`	5	42	47
`Total`	24	49	73

Table 1.11: Classification Table using logistic regression for 10 Markers with cut off value at 0.5 on probability scale

Here we see that 42 out of 47 diseased patients are correctly classified as diseased, 19 out of the 26 healthy individuals are correctly classified as healthy, 7 healthy individual is incorrectly classified as diseased and 5 patients are incorrectly classified as healthy.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{42}{47} = 0.89$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{7}{26} = 0.26$

The estimated test error is $\frac{5+7}{73} = 0.16$. That is a 16.43% test error rate.

Note: The way we created this validation set is by randomly selecting 50% and 50% from the diseased and non-diseased group. If we try to do this again we will see that we will create another test set and classification table. It can be shown that the test error that we will observe each time has a significant variance. As we have already mention, this problem can be solved by cross-validation. We must also consider the fact that we use only half of the information that is available to us, because we use only 50% of the data set for fitting the model.

Conclusions:

	AUC	Sensitivity	FPR	Specificity
`liver data set`	0.969	0.92	0.11	0.89
`training set`	0.987	0.93	0.07	0.93
`test set`	0.934	0.89	0.26	0.74

Table 1.12: Results based on the classifiaction rule on probability scale at 0.5

From table 2.8 we see that all AUC values are very high. The highers AUC value corresponds to the training set with AUC equal with 0.987. The values from sensitivity and specificity are also the highest for the training set. Using half of the data that are available to us and fitting a model, creates a more flexible model than fitting a model to all the data set. The better the model fit the training set the worst will the model fit the test set. The training error (0.06) is lower than the test error (0.16). Also to the test set corresponds the lowest AUC value and the lowest sensitivity and specificity value.

1.6 Supervised and Unsupervised Learning

Most of the statistical learning problems fall into one of the following two categories: supervised or unsupervised. The two categories have one main difference, in a supervised problem the dependent variable y is known whereas in an unsupervised problem the dependent variable is unknown.

Supervised Learning

In a supervised setting for each observation x_i, $i = 1, ..., n$ there is an associated response measurement y_i. The goal is to fit a model that accurately predicts the dependent variable y and to understand the relationship between the independent variables x the the dependent variable y (if it is possible). Many statistical learning methods operate in this setting, for example linear regression, logistic regression and support vector machines.

Unsupervised Learning

In an unsupervised setting we don't have a dependent variable but only the observations x_i. It is for example not possible to to fit a linear regression model since there is no response measurement for each x_i. The goal here is to understand the relationships between the variables or between the observations.

1.7 Resampling Methods

Resampling methods help us obtain additional information about the fitted model by repeatedly drawing samples from a training set and refitting our model on each sample. In this way we have more information that would not be available by using only once the original training set. In this chapter we will discuss two common resampling methods, validation and cross-validation.

Before we describe what validation and cross-Validation exactly is and how it works we will first briefly mention what a a training error and a test error is.

Training error

In general we fit the model to the training set. If we apply the same training set to our model the error of the model is called training error.

The training error is given by:

$$\text{Training Error } = \frac{\text{Total number of misclassification in the training set}}{\text{Total number of training set observations}}$$

As we increase the complexity of our model, the model usually adjust to the training set but in many cases it doesn't fit well to the unseen data (test set). This is called over fitting which is the inability of the model to generalise well the data. For example a polynomial of

degree $n+1$ can perfectly fit (training error $= 0$) to n data points, but what about a new unseen point?

Test error

In order to evaluate the performance of our model we use the test set. The average error that results from applying these new observations to our model is called test error.

The test error is given by:

$$\text{Test Error} = \frac{\text{Total number of misclassification in the test set}}{\text{Total number of test set observations}}$$

Ideally we want to find the the model complexity which has the smallest test error. Some methods use only the training set to estimate the test error, for instance, Cp statistic, AIC and BIC. These methods try to increase the training error, taking into account the amount of fitting and the variance.

1.7.1 Validation

Validation is a method used for estimating the test error, also known as prediction error. It divides randomly the data set into two parts. We name the one part training set and the second part test set. We fit the model to the training set and estimate the test error using the test set. One drawback is that if we divide the data again we will end up with different training and test set thus we will have a different test error. The variance from the test error is quite high.

Fig. 1.15: Source: An Introduction to Statistical Learning with Applications in R. A schematic display of the validation set approach. A set of n observations are randomly split into a training set (shown in blue, containing observations 7, 22, and 13, among others) and a validation set (shown in beige, and containing observation 91, among others). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.

1.7.2 k-Fold Cross-Validation

A more common way to estimate the test error from a given model, is to use k-fold cross-validation, in order to evaluate the models performance (known as model assessment) or to select the appropriate level of flexibility (known as model selection).

In k-fold cross-validation we divide the data into k approximately equal parts. The first part is treated as the test set, and we fit the model on the remaining $k-1$ parts. We then compute the test error $Test.Error_1$. This procedure is repeated k times, each time, a different k part is treated as a test set. We end up with k estimates of the test error: $Test.Error_1$, $Test.Error_2$, $Test.Error_k$. The k-fold CV estimate is computed by averaging these values:

$$CV_k = \frac{1}{k}\sum_{i=1}^{k} Test.Error_i$$

Common values of k are 5 and 10 fold cross validation. Using k-fold cross validation often gives more accurate estimates of the test error.

1.8 Bias-Variance Trade-off

Assessing Model Accuracy

Suppose that we have a model $h(x)$ and we fit it to the training set. We wish to see how well the model performs. One way of assessing the models accuracy is by computing the mean squared training error. But this may be biased towards more overfit models. As we saw in the previous section we can instead use the test set and compute the mean squared test error.

Example

In the figure (1.16) on the left side we see a black curve which is simulated. This is our true function that we want to estimate. The points were generated by this curve with error.

We also have three different models fit to this data (the orange, the blue and the green model) and they are ordered in complexity. The orange model is a linear model, the blue model is a more flexible model and the green model is an even more flexible model than the blue one (we can see that is gets closer to the data points).

Since this is a simulated example we can compute the mean squared error on a very large test data set. On the right figure we plot the mean squared error for the test set (red curve). It starts of high for the linear model, then it drops down for the in-between model and then for the more flexible model it's starts increasing again.

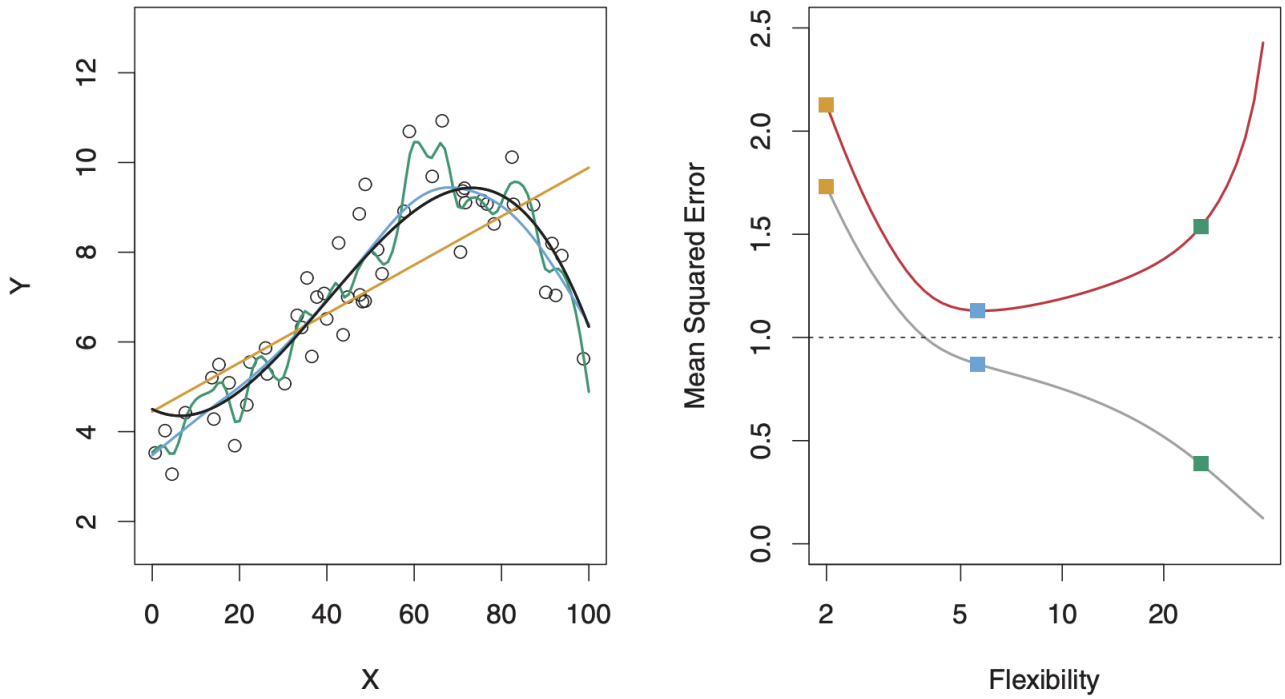


Fig. 1.16: Source: An Introduction to Statistical Learning with Applications in R. Left: Data simulated from h shown in black. Three estimates of h are shown: the linear regression line (orange curve), and two non linear lines (blue and green curves). Right: Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.

The mean squared error for the training set (grey curve) keeps on decreasing. The more flexible the models is, the more closer it gets to the data points.

For the MSE for the test set, we see that there is an optimal point which minimises the MSE. In this example this optimal point is the blue point on the red curve which corespondents to the medium flexible (blue) model. We also see that the blue curve gets very close to the black curve. Because this is a simulation model, the dotted line in the right figure is the MSE that the true function makes and this is the irreducible error $Var(\epsilon)$.

If we want to have a model that has good prediction error (this is measured here in terms of MSE on the test set. We would like to estimate the red curve from the right side figure and we can do that by having a test data set which we can use to evaluate the performance of the different models.

Bias-Variance Trade-off

Suppose the we have fit a model $\hat{h}(x)$ to a training set and let (x_0, y_0) be a test observation drawn from the population. If the true model is $Y = h(x) + \epsilon$ with $h(x) = E(Y|X = x)$, then

$$E(y_0 - \hat{h}(x_0))^2 = Var(\hat{h}(x_0)) + [Bias(\hat{h}((x_0))]^2 + Var(\epsilon)$$

- $Var(\epsilon)$: irreducible error that comes from the random variation in the new test point y_0.
- $Var(\hat{h})$: is the variance that comes from having different training sets. If we got a new training set and fit our model again, we would get a different function h. But if we have many different training sets, there would be variability in the prediction at x_0.
- $Bias(\hat{h}(x_0))$: is the difference between the average prediction at x_0 over all these different training sets E[(x_0)] and the true $h(x_0)$.

Typically as the flexibility of $\hat{h}$ increases, its variance increases because it's going after the individual training set, but its bias decreases. So choosing the flexibility bases on average test error amounts to a bias-variance trade-off.

Continuing previous example

Continuing the previous example, we have that as the flexibility of the model increases, the bias decreases and the variance increases. By adding these two components (bias which is shown in blue in figure (1.17) and variance which is shown in orange), we get the MSE (red curve) of the test set.

Choosing the amount of flexibility of a model amounts to a bias-variance trade off. Depending on the problem we might want to make the trade off in a different place. We can use the test set to help us make that choice.

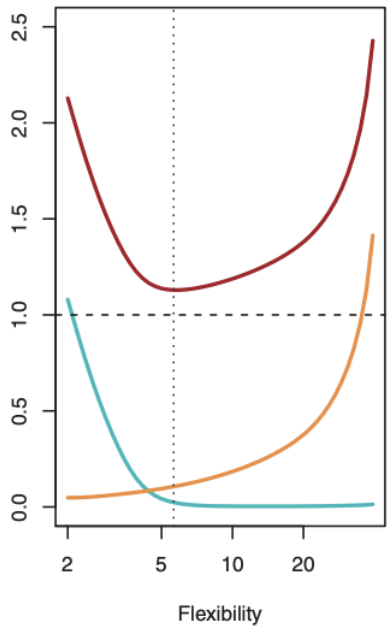


Fig. 1.17: Source: An Introduction to Statistical Learning with Applications in R. The red curve is the MSE of the test set. Then we have the two important components of the MSE: the bias and the variance. The bias is shown with blue and the variance with orange.

1.9 Curse of Dimensionality

Suppose that we want to estimate a model h where $\hat{h}(x) = E(Y|X = x)$. Suppose also that we want to estimate the h for $x = 4$ and we have few if any data points with $x = 4$. We will relax the definition and let $\hat{h}(x) = E(Y|X \in N(x))$ where $N(x)$ is some neighborhood of x. This is called nearest neighbor averaging and it can be good for small number of dimensions p and large number of training set. Nearest neighbor methods can be lousy when p is large and the reason is the curse of dimensionality. Nearest neighbors tend to be far away in high dimensions.

We need to get a reasonable fraction of the N values of y_i to average to bring the variance down - e.g. 10%. A 10% neighborhood in high dimensions need no longer be local, so we lose the spirit of estimating $E(Y|X = x)$ by local averaging.

Example

Suppose that we have two variables x_1 and x_2. They are uniformly distributed from -1 to 1 in the left plot of figure (1. 18). We form two 10% neighborhoods. The first neighborhood is just involving the variable x_1 ignoring x_2. This is indicated by the vertical dotted lines. The target point is at 0 and we spread out a neighborhood to the left and the right until we capture 1-% of the data points with respect to the variable x_1. The dotted line indicates the width of the neighborhood.

Alternatively, if we want to find a neighborhood in two dimensions, we spread out a circle centered at the target point (the red dot) until we have captured 10% of the points. Note that the radius of the circle in two dimensions is much bigger than the radius of the circle in one dimension which is the width between the two dotted lines.

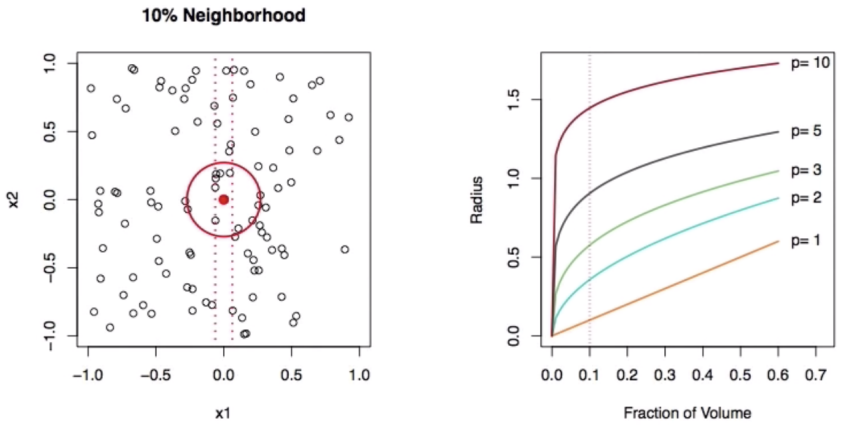


Fig. 1.18: Source: An Introduction to Statistical Learning with Applications in R.

To capture 10% of the points in two dimensions, we have to go out further and so we are less local than we are in one dimension.

On the right hand plot we can see how far we have to go out in $p = 1, 2, 3, 5, 10$ dimensions in order to capture a certain fraction of the volume.

If we take for example 10% or 0.1 fraction of the volume, for $p = 1$, if the data are uniform, we have to go out 10% of the distance. In two dimensions, as we saw before we have to go out more. On 5 dimensions we have to go out about 0.9 on each coordinate axes to get 10% of the data which is about the whole radius of the sphere of the left plot. In 10 dimensions we actually have to go break out of this sphere in order to get points in the corner to capture the 10%.

So is it very hard to find neighborhoods in high dimensions and stay local. In chapter 3 we will describe methods that avoid the curse of dimensionally.

Chapter 2

Classification

In a classification problem the goal is to classify correctly a subject, depending on some characteristics (features). An example of a classification problem is to predict if a person is diseased or not diseased. In this (binary) classification problem we try to predict if a person y is either 0 (for non diseased for instance) or 1 (for diseased). That is $y \in \{0, 1\}$. In this chapter we will see why the linear model is not appropriate for a classification problem and how a transformation of the linear predictor–logistic regression can produce more sensible results. An alternative to this transformation is to use kernel methods and specifically the method support vector machines.

2.1 Linear Models

The most common linear model is the linear regression model. It is seldom the case that a linear regression model fits well to our data, though linear model is a first order Taylor approximation and it gives an easy interpretation to our problem.

2.1.1 Cost function

In linear regression the cost function helps us to fit the best possible straight line to our data. The model has the following form, $h_\theta(x) = \theta_0 + \theta_1 x$ and θ_0 and θ_1 are the parameter. One problem is how to choose the values of the parameters. With different combinations of the parameters we get different h functions.

One way to estimate θ_0 and θ_1 is to select those values that fit $h_\theta(x)$ as close as possible to y –least squares method. In other words, we have a minimisation problem where we want to minimise over θ_0 and θ_1:

$$\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \tag{2.1}$$

where m is the size of the training sample. For convenience we will minimize

$$\frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \tag{2.2}$$

which gives the same values of θ_0 and θ_1 like (2.1).

We define a cost function $J(\theta_0, \theta_1)$ so that:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \tag{2.3}$$

Goal: We want to find θ_0 and θ_1 so that they minimize $J(\theta_0, \theta_1)$. This cost function is also called the squared error cost function. In the case where we have only two parameters θ_0 and θ_1 the form of the cost function is as shown in the graph below.

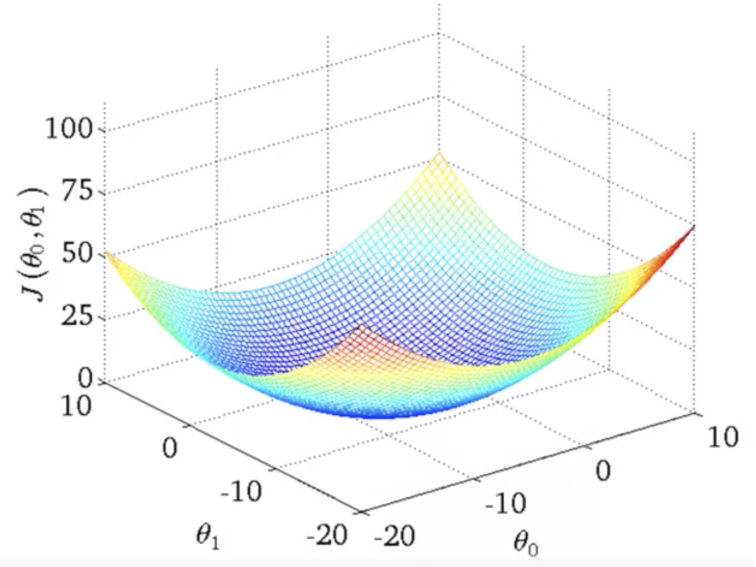


Fig. 2.1: Source: Coursera, Machine Learning by Stanford University with Andrew Ng. Linear regression cost function for θ_0 and θ_1

Note that for different values of θ_0 and θ_1 we get a different value of $J(\theta_0, \theta_1)$ which is given by the hight of the surface. So $J(\theta_0, \theta_1)$ is a function of the parameters θ_0 and θ_1, and $h_\theta(x)$ for fixed θ_0 and θ_1 is a function of x.

2.1.2 Gradient Descent

A numerical solution to the optimisation problem can be given by algorithms like Gradient Descent.

The idea of Gradient descent is to:

- Start with some θ_0 and θ_1.
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum.

Algorithm 1 Gradient descent Algorithm

1: Initialize θ_0 and θ_1
2: Repeat until convergence $\theta_j := \theta_j - a\frac{\partial}{\partial\theta_j}J(\theta_0, \theta_1)$ for $j = 0$ and $j = 1$

Note that by $:=$ we denote assignment. The a is the learning rate and it controls how big is the step we take with gradient descent. So if a is very large, then that corresponds to a very aggressive gradient descent procedure where we take huge steps. If a is very small then we are taking small steps. We update simultaneously θ_0 and θ_1.

Suppose that we have only one parameter θ_1 and we want to minimize $J(\theta_1)$, $\theta_1 \in R$. $\frac{\partial}{\partial\theta_j}J(\theta_1)$ gives us either a positive or a negative number indicating the direction of where the next step should be to get closer to the minimum. Gradient descent can converge to a local minimum even with the learning rate a fixed because as we approach a local minimum, gradient descent will automatically take smaller steps.

2.1.3 Gradient Descent for Linear regression

The linear regression model is

$$h_\theta(x) = \theta_0 + \theta_1 x$$

and the cost function is

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

The goal is to minimize $J(\theta_0, \theta_1)$.

Let's find the gradient descent for linear regression. We have that

$$\frac{\partial}{\partial\theta_j}J(\theta_0, \theta_1) = \frac{\partial}{\partial\theta_j}\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 = \frac{\partial}{\partial\theta_j}\frac{1}{2m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

For θ_0 , $j = 0$:

$$\frac{\partial}{\partial\theta_0}J(\theta_0, \theta_1) = \frac{1}{m}\sum(h_\theta(x(i) - y_i)$$

For θ_1 , $j = 1$:

$$\frac{\partial}{\partial\theta_1}J(\theta_0, \theta_1) = \frac{1}{m}\sum(h_\theta(x(i) - y_i)x^{(i)}$$

Note that the cost function for the linear regression function is a convex function, therefore it has one global optimum.

Algorithm 2 Gradient descent Algorithm for Linear regression

1: Initialize θ_0 and θ_1
2: Repeat until convergence

$$\theta_0 := \theta_0 - a\frac{1}{m}\sum(h_\theta(x(i) - y_i)x^{(i)}$$
$$\theta_1 := \theta_1 - a\frac{1}{m}\sum(h_\theta(x(i) - y_i)x^{(i)}$$

Why not linear regression

To attempt classification, one method is to use linear regression and map all predictions greater than 0.5 as a 1 and all less than 0.5 as a 0. However, this method doesn't work well because classification is not actually a linear function. The classification problem is like the linear regression problem, except that the values we want to predict are between 0 and 1. We will focus on the binary classification problem in which y can take only two values, 0 and 1.

2.2 Classification of liver data

The liver data set, which we described in Chapter 1, comprise diseased and non-diseased individuals (two categories). In such cases we want to classify the subjects s_i, in these categories. In general the categories can be more than two. In the liver data set, the response variable Y is the true diseases status and the observations are the 236 Markers. To each subject s_i belongs 236 measurements and to each measurement corresponds a marker. That is:

$$S = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_{145} \end{bmatrix}, Y = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_{145} \end{bmatrix}, X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1,236} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2,236} \\ x_{31} & x_{32} & x_{3,3} & \dots & x_{3,236} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{1,451} & x_{1,452} & x_{1,453} & \dots & x_{145,236} \end{bmatrix} = \begin{bmatrix} X_1' \\ X_2' \\ X_3' \\ \vdots \\ X_{451}' \end{bmatrix}$$

Where

- The matrix S consists of the 145 subjects
- Y comprise the disease status of each subject
- Y_i is a binary variable $Y_i = \begin{cases} 0, & \text{if the subject is healthy} \\ 1, & \text{if the subject is diseased} \end{cases}$
- X is a 145×236 matrix. Each row corresponds to a subject and consists of 236 measurements and each column consists of the values of a marker.

- The subject s_i has 236 measurements: $X_i = \begin{bmatrix} x_{i1} & x_{i2} & x_{i3} & \dots & x_{i236} \end{bmatrix}$ where $i \in 1, 2, ..., 145$
- Each marker has 145 values: $Marker_i = \begin{bmatrix} x_{1i} & x_{2i} & x_{3i} & \dots & x_{145i} \end{bmatrix}$ where $i \in 1, 2, ..., 236$

In the remaining part of this Chapter we will cover two classification methods, Logistic Regression which is one of the most traditional parametric classification methods and Support Vector Machines which is a kernel method and closely related to logistic regression.

2.3 Logistic Regression

The Logistic regression model was developed by David Cox in 1958 and is used in various fields. Logistic regression is a supervised classification method.

In the Logistic regression model we want $0 \leq h_\theta(x) \leq 1$ so that our predictions (which are probabilities) are between 0 and 1. In Linear regression the form of h is $h_\theta(x) = \theta^T x$. For Logistic regression we modify this function as

$$h_\theta(x) = g(\theta^T x) \tag{2.4}$$

with

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.5}$$

so that $g(z) \in (0, 1)$.

$g(z)$ is called the sigmoid function or logistic function. If we put the equations (2.5) and (2.6) together we have:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \tag{2.6}$$

The form of the sigmoid function is shown in figure (2.2):

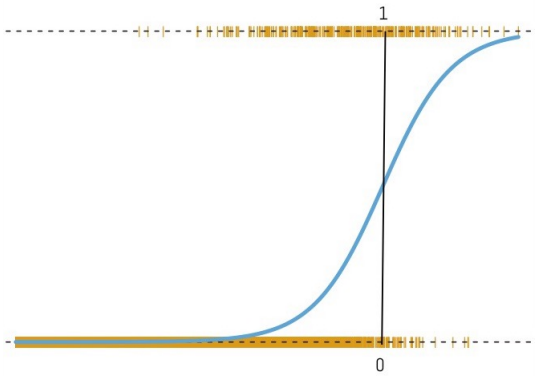


Fig. 2.2: Source: An Introduction to Statistical Learning with Applications in R . Here we see the predicted probabilities using logistic regression. All probabilities (the blue curve) lie between 0 and 1. The orange points represent the data set with is used to fit the model.

Notice that the sigmoid function, while z goes to minus infinity, $g(z)$ approaches 0, and as z goes to infinity, $g(z)$ approaches 1.

The next step is to fit the parameters θ. Given a training set we need to pick a value for the parameters θ and then $h_\theta(x)$ will help us to make predictions.

Interpretation of the $h_\theta(x)$ output:

$h_\theta(x)$ gives us the estimated probability that $y = 1$ on input x. That is:

$$h_\theta(x) = P(y = 1|x, \theta)$$

which is the probability that $y = 1$, given x, parameterized by θ. Because y is either 0 or 1, we have that:

$$P(y = 1|x, \theta) + P(y = 0|x, \theta) = 1$$
$$P(y = 0|x, \theta) = 1 - P(y = 1|x, \theta)$$

Notice that:

$$h_\theta(x) = P(y = 1|x, \theta) = \frac{1}{1 + e^{-\theta^T x}} = \frac{e^{\theta^T x}}{1 + e^{\theta^T x}}$$

- For $\theta^T x \to +\infty$, $e^{\theta^T x} \to +\infty$ and $\frac{e^{\theta^T x}}{1+e^{\theta^T x}} \to 1$.
- For $\theta^T x \to -\infty$, $e^{\theta^T x} \to 0$ and $\frac{e^{\theta^T x}}{1+e^{\theta^T x}} \to 0$.
- So $P(y = 1|x, \theta) \in [0, 1]$ for any values of $\theta^T x$.

Decision Boundary

The function $h_\theta(x)$ estimates the probability that y is equal to 1, given x parameterized by θ. So if we want to predict if y is equal to 1 or if y is equal to 0, here's something we might do: whenever $h_\theta(x)$ outputs that the probability of y is greater or equal to 0.5, then we predict that y equals 1. And otherwise, if the estimated probability of y being 1 is less than 0.5, then we predict that y equals 0.

The way the logistic function g behaves is that when its input is greater or equal to zero, it's output is greater or equal to 0.5:

$$g(z) \geq 0.5 \text{ when } z \geq 0$$

So if our input to g is $\theta^T X$, then that means:

$$h_\theta(x) = g(\theta^T x) \geq 0.5 \text{ when } \theta^T x \geq 0$$

From these statements we can say:

$$\theta^T x \geq 0 : y = 1$$

$$\theta^T x < 0 : y = 0$$

The decision boundary is the line that separates the area where $y = 0$ and where $y = 1$ and it is created by the $h_\theta(x)$ function.

2.3.1 Cost function of the logistic regression

We can not use the same cost function that we use for linear regression because the logistic function will cause the output to be wavy, causing many local optima. In other words, it will not be a convex function. In logistic regression a reasonable assumption is that y_i follows Bernoulli distribution.

We have that:

$$P(y_i = 1|x, \theta) = h_\theta(x)$$

thus

$$P(y_i = 0|x, \theta) = 1 - h_\theta(x)$$

Because y_i follows Bernoulli distribution we have:

$$P(y_i|x, \theta) = (h_\theta(x))^{y_i} (1 - h_\theta(x))^{(1-y_i)}$$

Then the likelihood is:

$$P(y_i|x,\theta) = \prod_i^m (h_\theta(x))^{y_i} \, (1 - h_\theta(x))^{(1-y_i)}$$

and the negative log likelihood is:

$$-\sum_i^m y_i \log h_\theta(x) + (1 - y_i) \log(1 - h_\theta(x))$$

or

$$-ylog(h_\theta(x)) - (1 - y)log(1 - h_\theta(x))$$

which is the cost function.

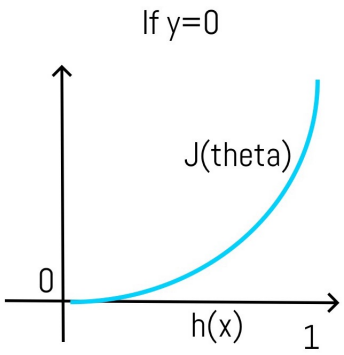


Fig. 2.3: Plot of the cost function $J(\theta)$. If the correct answer for y is 0, then the cost function will be 0 if $h_\theta(x)$ outputs 0. If $h_\theta(x)$ approaches 1, then the cost function will approach infinity.

We can fully write out our entire cost function as follows:

$$J(\theta) = \frac{1}{m} Cost(h_\theta(x), y)$$

where:

$$Cost(h_\theta(x), y) = -\log(h_\theta(x)) \text{ if } y = 1$$
$$Cost(h_\theta(x), y) = -\log(1 - h_\theta(x)) \text{ if } y = 0$$

Also:

$$Cost(h_\theta(x), y) = 0 \text{ if } h_\theta(x) = y$$

and

$$Cost(h_\theta(x), y) \to +\infty \text{ if } y = 0 \text{ and } h_\theta(x) \to 1$$

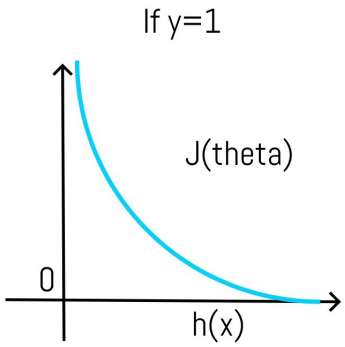


Fig. 2.4: Plot of the cost function $J(\theta)$. If the correct answer for y is 1, then the cost function will be 0 if the $h_\theta(x)$ function outputs 1. If $h_\theta(x)$ approaches 0 , then the cost function will approach infinity.

$$Cost(h_\theta(x), y) \to +\infty \text{ if } y = 1 \text{ and } h_\theta(x) \to 0$$

If the correct answer for y is 0, then the cost function will be 0 if $h_\theta(x)$ outputs 0. If $h_\theta(x)$ approaches 1, then the cost function will approach infinity.

If the correct answer for y is 1, then the cost function will be 0 if the $h_\theta(x)$ function outputs 1. If $h_\theta(x)$ approaches 0 , then the cost function will approach infinity.

$$J(\theta) = -\frac{1}{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1 - y^{(i)})log(1 - h_\theta(x^{(i)}))]$$

The cost function $J(\theta)$ is convex for logistic regression.

A vectorized implementation is:

$$h = g(X\theta)$$

$$J(\theta) = \frac{1}{m}(-y^T log(h) - (1 - y)^T log(1 - h))$$

For this optimisation problem we don't have a close form solution, but since this is a convex optimisation problem and we can use numerical solutions.

2.3.2 Gradient Descent of the logistic regression

As we saw before, the general form of the gradient descent is:

Algorithm 3 General form of Gradient descent Algorithm

1: Initialize θ_0 and θ_1
2: Repeat until convergence $\theta_j := \theta_j - a\frac{\partial}{\partial\theta_j}J(\theta)$

By using the cost function $J(\theta)$ and calculating the derivative part, we have:

Algorithm 4 Gradient descent for Logistic regression

1: Initialize θ_0 and θ_1
2: Repeat until convergence $\theta_j := \theta_j - \frac{a}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$

As before, we have to simultaneously update all values in θ.

Advantages of Logistic regression

- Logistic regression is interpretable
- Logistic regression can be extended for more categories.
- It forms the foundation for some more complex methods like neural networks.

Disadvantages of Logistic regression

- The performance is not necessarily as good as some other methods but it always depends on the problem.

2.4 Support Vector Machines

Support Vector Machines (SVM) is a supervised classification method, created by Vladimir Vapnik (1995). It approaches the classification problem directly, making no use of densities or a probability model but it tries to separate the feature space by finding the largest margin between the decision boundary and the features. We start this section by describing the relationship between logistic regression and support vector machine.

2.4.1 An alternative view of logistic regression

As we saw in Chapter 2, the logistic regression model is

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

For $z = \theta^T x$ we have $h_\theta(x) = g(z)$, where $g(x)$ is the sigmoid function.

If we have an example where $y = 1$ (in the training or in the test set), then we hope that the logistic model will output a value close to 1, $h_\theta(x) \approx 1$ or in other words $\theta^T x \gg 0$.

Conversely, if we have an example where $y = 0$ (in the training or in the test set), then we want $h_\theta(x) \approx 0$ or in other words $\theta^T x \ll 0$.

The cost function of logistic regression for one example (x, y) is:

$$-(y \log h_\theta(x)) + (1 - y) log(1 - h_\theta(x))) =$$
$$-y \log \frac{1}{1 + e^{-\theta^T x}} - (1 - y) \log(1 - \frac{1}{1 + e^{-\theta^T x}})$$

In the first case suppose that $y = 1$ (we want $\theta^T x \gg 0$):

In this case only the first term in the cost function matters (the second term is equal with 0).

So when $y = 1$ we have:

$$-\log \frac{1}{1 + e^{-\theta^T x}}$$

If we plot this function as a function of z (where $z = \theta^T x$) we get the blue curve in figure (2.5). Thus we see that when z is equal to some large number, $\theta^T x$ is large and that corresponds to a very small contribution of $\theta^T x$ to the cost function (-$\log \frac{1}{1+e^{-\theta^T x}}$ is small).

In support vector machine we modify $-\log \frac{1}{1+e^{-z}}$. The new cost function is shown in red in figure (2.5). It is flat for $z > 1$ and linear for $z < 1$. We call this function $cost_1(z)$ (for $y = 1$).

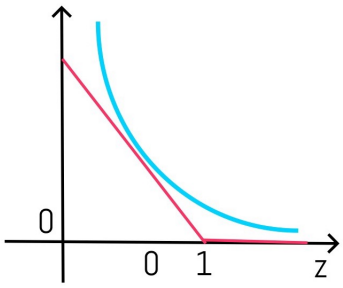


Fig. 2.5: Plot of the cost function of the logistic and SVM for $y = 1$. The blue curve corresponds to the cost function of the logistic regression for $y = 1$ and the red line corresponds to the cost function of the support vector machine for $y = 1$

In the second case suppose that $y = 0$ (we want $\theta^T x \ll 0$):

In this case only the second term in the cost function matters (the first term is equal with 0).

So when $y = 0$ we have:

$$-\log(1 - \frac{1}{1 + e^{-\theta^T x}})$$

If we plot this function as a function of z (where $z = \theta^T x$) we get the blue curve in figure (2.6). Thus we see that when z is equal to some small number, $\theta^T x$ is small and that corresponds to a very small contribution of z to the cost function $-\log(1 - \frac{1}{1+e^{-z}})$ is small).

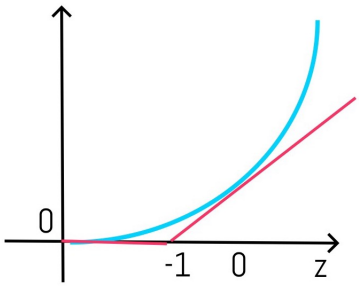


Fig. 2.6: Plot of the cost function of the logistic and SVM for $y = 0$. The blue curve corresponds to the cost function of the logistic regression for $y = 0$ and the red line corresponds to the cost function of the support vector machine for $y = 0$

In support vector machine as before we modify $-\log(1 - \frac{1}{1+e^{-z}})$. The new cost function is shown in red in figure (2.6). It is flat for $z < -1$ and linear for $z > -1$. We call this function $cost_0(z)$ (for $y = 0$).

The cost function for logistic regression is:

$$min_\theta \frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \left(-\log(h_\theta(x^{(i)})) \right) + (1 - y^{(i)})(-log(1 - h_\theta(x^{(i)}))) \right]$$

Thus the cost function for support vector machine is:

$$min_\theta \ \frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right]$$

By minimizing this as a function of the parameters θ the cost function we get the parameters for support vector machine. Unlike logistic regression, support vector machine does not output a probability. The support vector machine makes a prediction of y being equal to 1 or 0, directly.

2.4.2 Large Margin Classifier

In this section we will include the regularization term (see Chapter 3) in the cost function in order to describe how the large margin classifier works.

The cost function of support vector machine with the regularization term (see Chapter 3) is:

$$min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \sum_{j=1}^{p} |\theta_j|$$

When we take a positive example, then $cost_1(z) = 0$ only when $z \geq 1$, or in other words $\theta^T x \geq 1$.

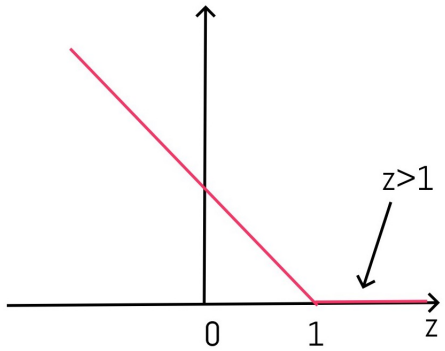


Fig. 2.7: Plot of the cost function of the SVM for $y = 1$. For a positive example the $cost_1(z) = 0$ only when $z \geq 1$

Conversely, if $y = 0$ then $cost_1(z) = 0$ only when $z \leq -1$, or in other words $\theta^T x \leq -1$. Notice that by putting $z \geq 1$, $z \leq -1$ and not 0, in the support vector machine we don't want to get barely the example right. Instead we want this to be a lot bigger or smaller than 0 (≥ 1 or ≤ -1). This builds an extra safety factor into the support vector machine.

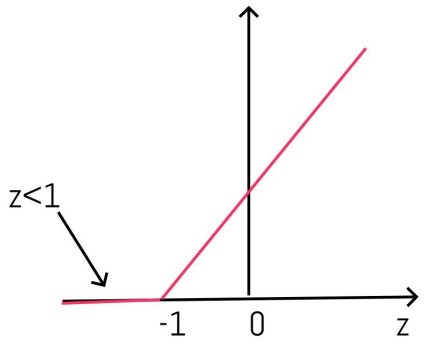


Fig. 2.8: Plot of the cost function of the SVM for $y = 0$. For a negative example the $cost_1(z) = 0$ only when $z \leq -1$.

The cost function of support vector machine with the regularization term is:

$$min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \sum_{j=1}^{p} |\theta_j|$$

Suppose that C is some very large value. By minimizing the cost function as a function of θ we choose the parameters θ so that:

$$\sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] = 0$$

So we have the following optimaization problem:

$$min_\theta C * 0 + \sum_{j=1}^{p} |\theta_j| =$$

$$min_\theta \sum_{j=1}^{p} |\theta_j|$$

So when $y^{(i)} = 1 : \theta^T x^{(i)} \geq 1$ and when $y^{(i)} = 0 : \theta^T x^{(i)} \leq -1$. The decision boundary we get by minimizing this as function of the parameters θ is shown in the following example.

Example 1

Suppose that the red points are the positive samples and the blue crosses are the negative samples in figure (2.9). We can see that we can draw a straight line that can separate the positive and the negative samples perfectly.

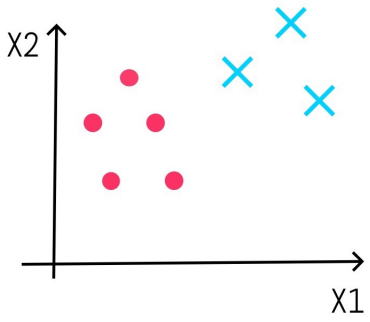


Fig. 2.9: Scatter plot of the data for example 1. With red we have the positive examples and with blue the negative examples.

We can for example plot these three lines shown in figure (2.10) with green, red and black.

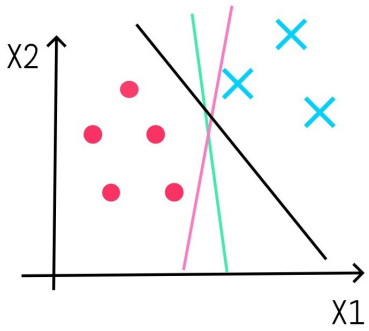


Fig. 2.10: There are many decision boundaries which can separate the data, here we see three of them in green, pink and black.

The black line looks to be a better decision boundary than the other two because it seems to be more robust. The black decision boundary has the largest distance from the samples. This distance is called the margin of the support vector machine (see figure (2.11), the dotted lines). The margins give the support vector machine a robustness because it tries to separate the data with a margin as large as possible. For this reason the support vector machine is often called large margin classifier. This is also a consequence of the optimization problem which will be discussed later.

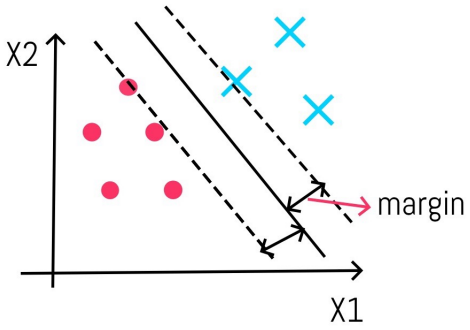


Fig. 2.11: The optimal decision boundary seems to be the black one because it has the largest distance with the data, thus the largest margin.

Example 2
Given a data set shown in figure (2.12) in the case when C (the regularisation parameter) is very large, the large margin classifier is the black line in figure (2.12).

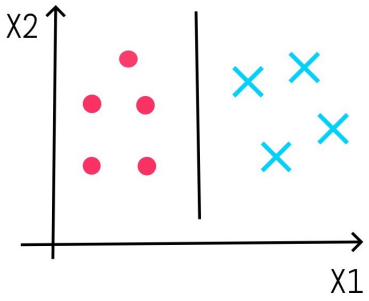


Fig. 2.12: Scatter plot of the data for example 2. For C some large number the decision boundary is the black line.

But by using only the large margin classifier the SVM can be sensitive to outliers. If we add an extra blue cross we can get a very different decision boundary (see figure (2.13)).

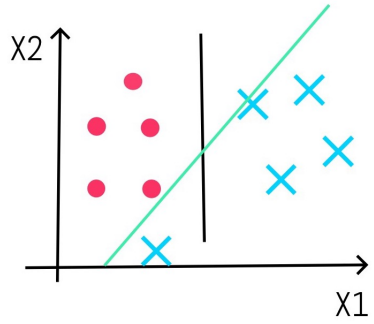


Fig. 2.13: By adding one only outlier and keeping the value of C large, the decision boundary changes from the black line to the green line.

So based on a single outlier the decision boundary has changed from the black one, over to the green one. So if the regularisation parameter C is very large then the SVM will have as decision boundary the green line. But if C is small (or if the data are not linearly separable) the SVM will have the black line as the decisions boundary (see figure (2.14)). All the above statements will became more clear later in this Chapter.

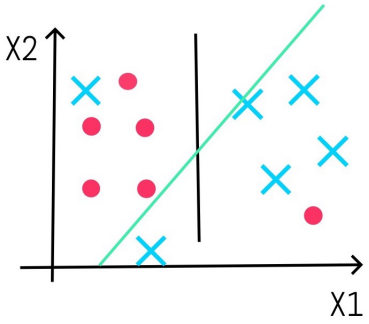


Fig. 2.14: Adding more outliers to the data, choosing the optimal value of C can be difficult. For C large the decision boundary will fit the (training) data as good as possible being also more sensitive to outliers. For C small the decision boundary will remain the black line and being also more robust to outliers.

2.4.3 Some mathematic background

Vector Inner Products

Suppose that we have 2 two dimensional vectors u and v. The inner product is $u^T v$ and the euclidian length of u (or the norm) is $||u|| = \sqrt{u_1^2 + u_2^2} \in R$. For computing the inner product of u and v we project the vector v onto the vector u as shown in figure (2.15). We measure the length of p (see figure) which is the length of the projection of the vector v onto the vector u.

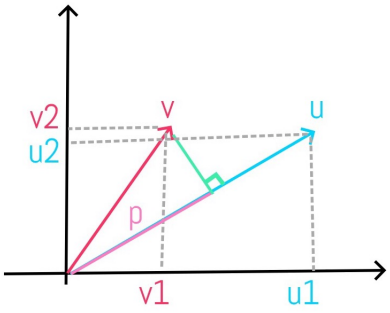


Fig. 2.15: By projecting u onto v (or v onto u) we get the length of p. The inner product of $u^T v$ is $p||u|$ where p is the length shown in pink and $||u||$ is the norm of u.

The inner product is equal to:

$$u^T v = p||u|| = u_1 v_1 + u_2 v_2 = v^T u$$

where $p \in R$ and $||u|| \in R$. Note also that p can be positive or negative. For example, in the case where the angle between the two vectors is greater than 90^o then if we project v onto u , p is negative (see figure 2.16).

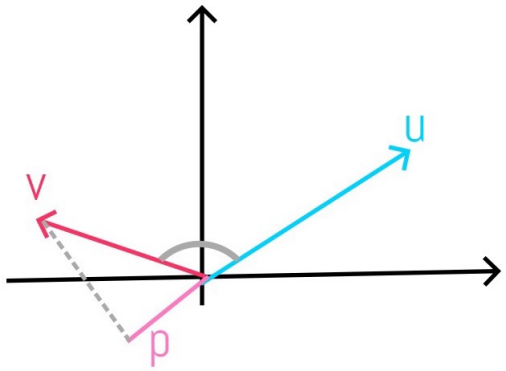


Fig. 2.16: The length of p can also be negative if the angle between the two vectors is greater than 90^o.

2.4.4 Minimizing the cost function

In section "large margin classifier" we saw that for C some large number the cost function is:

$$min_\theta \sum_{j=1}^{p} |\theta_j|$$

Suppose that $\theta_0 = 0$ we we have only two features x_1 and x_2. Then:

$$min_\theta \sum_{j=1}^{p} |\theta_j| = min_\theta |\theta_1 + \theta_2| = min_\theta \sqrt{\theta_1^2 + \theta_2^2} = min_\theta ||\theta||$$

Given a parameter θ and an example x (only one!) the inner product of the vector θ and x is equal with the length p times the norm of the vector θ.

$$\theta^T x = p||\theta|| = \theta_1 x_1 + \theta_2 x_2$$

So we have for an example x_i that :

$$\theta^T x^{(i)} = p^{(i)}||\theta||$$

Therefore:

$$\theta^T x^{(i)} \geq 1 \text{ if } y^{(i)} = 1$$
$$\theta^T x^{(i)} \leq -1 \text{ if } y^{(i)} = 0$$

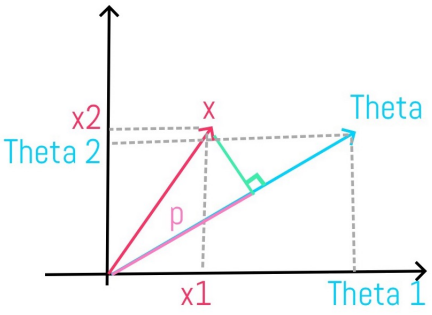


Fig. 2.17: Projection from the vector x onto the vector θ. Regarding where the vector x is the value of p can be positive or negative.

and:

$$p^{(i)}||\theta|| \geq 1 \text{ if } y^{(i)} = 1$$
$$p^{(i)}||\theta|| \leq -1 \text{ if } y^{(i)} = 0$$

where $p^{(i)}$ is the projection of the i training example $x^{(i)}$ onto the parameter θ.

Example

Suppose the data set shown in figure (2.18) and $\theta_0 = 0$. Suppose also that the decision boundary is the green line in the same figure.

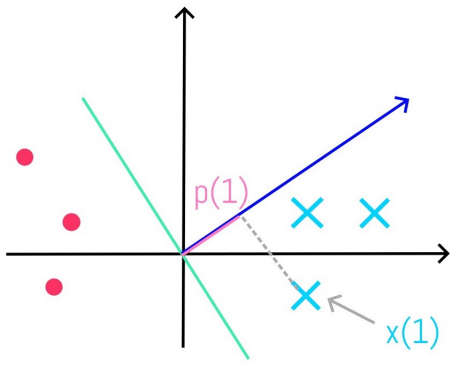


Fig. 2.18: Suppose that the red points and the blue crosses is our data set. The blue cross x_1 is our first example, we project it onto the vector θ - which is the blue vector, and we get the distance $p^{(1)}$.

The vector θ is shown in the figure (2.19) with blue. Because $\theta_0 = 0$ the θ vector passes through the origin $(0, 0)$.

Suppose that the blue cross x_1 is the first example. The projection of this example to the parameter θ is $p^{(1)}$ which is a small number.

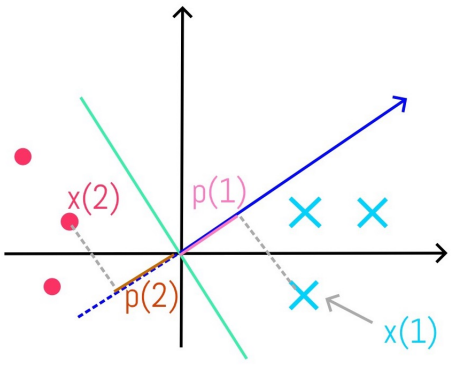


Fig. 2.19: For the red point x_2 the projection onto the parameter θ is $p^{(2)}$, where $p^{(2)} < 0$.

For the red point x_2 the projection of this example onto the parameter θ is $p^{(2)}$ (where $p^{(2)} < 0$) as shown in the figure.

For this decision boundary the $p^{(i)}$ are small numbers. For positives examples we need $p^{(i)}||\theta|| \geq 1$, but if $p^{(i)}$ is small we need the $||\theta||$ to be large. Similarly, for a negative example, we need $p^{(i)}||\theta|| \leq -1$, but if $p^{(i)}$ is small we need the $||\theta||$ to be large.

This comes in contrast with what we want to do to the cost function $\theta^T x^{(i)} = p^{(i)}||\theta||$, where we try to find a setting of parameters where the norm $||\theta||$ is small.

Suppose the green line in figure (2.20) is another decision boundary. The vector θ is shown in blue. The projection of $x^{(1)}$ and $x^{(2)}$ onto θ is $p^{(1)}$ and $p^{(2)}$ respectively. So $p^{(1)}$ and $p^{(2)}$ are bigger now and θ can be smaller than by using the previous decision boundary.

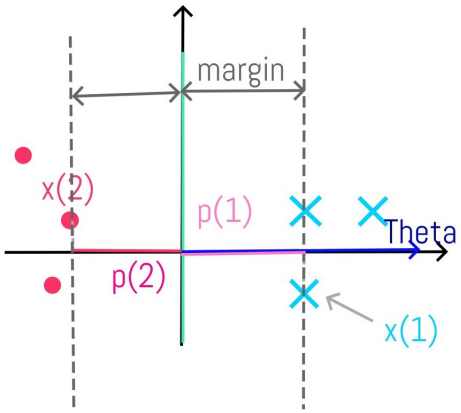


Fig. 2.20: By choosing this green line for our decision boundary the $p^{(1)}$ and $p^{(2)}$ values are larger than for the previous decision boundary. The margin is also larger, therefore this green decision boundary is a better choice.

By choosing this decision boundary instead of the previous one, the SVM can make the norm $||\theta||$ of the parameter smaller, therefore the cost function smaller. This is also the reason of why the SVM will choose this decision boundary. By making the $p^{(i)}$ large, we make the margin large, so that the SVM can have smaller values of θ which is our goal.

In this example we supposed that $\theta_0 = 0$ therefore the decision boundaries will cross through the origin. So when $\theta_0 \neq 0$ the decision boundary doesn't have to cross the origin $(0, 0)$.

2.4.5 Non-linearities and Kernels

Support vector machine can also be used to develop more complex nonlinear classifiers. The technique for doing that is by using kernels. We will start with the normal kernel which is the most popular.

Normal Kernel

To explaining the normal kernel we start with an example.

Example

Suppose that we have the training set shown in figure (2.21) and we would like to distinguish the red (positive) and the blue (negative) examples.

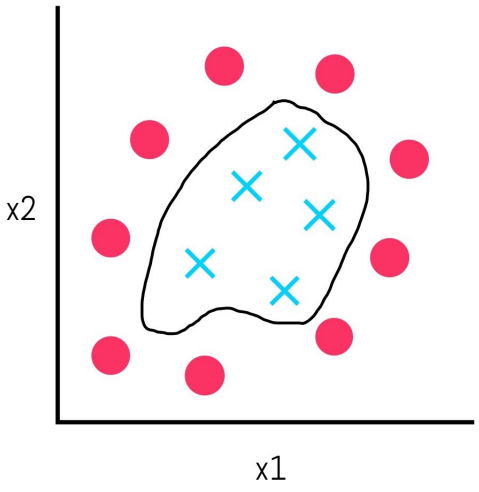


Fig. 2.21: Training set containing red circles and blue crosses. The decision boundary is a more complex function (the black circle)

Ideally the decision boundary would look like the black circle. One way to do this is to come up with a set of complex polynomial features:

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + +\theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + ... \tag{2.7}$$

So we have:

$$h_\theta(x) = \begin{cases} 1, & \text{if } \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + ... \geq 0 \\ 0, & \text{if otherwise} \end{cases}$$

The function (2.7) can also be written as:

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4 + ...$$

where with f_i we denote the features ($f_1 = x_1, f_2 = x_2, f_3 = x_1 x_2, ...$)

Is there a better choice of the features $f_1, f_2, f_3, ...$ that we can use?

We will answer this question in the following example.

Example

In this example we define three red points l^1, l^2, l^3. The goal is to find new features $f_1, f_2, f_3, ...$ bases on the similarity of the training set with the l^1, l^2, l^3.

Suppose that we have two features x_1 and x_2. We will define the new features as follows:

Given x:
The first feature f_1 is some measure of the similarity between the training example x and l^1. One form to measure the similarity is:

$$\exp\left(-\frac{||x-l^1||^2}{2\sigma^2}\right)$$

where $||x-l^1||^2$ is the euclidean distance between the point x and l_1.

Similarly the next feature f_2 is the similarity between the training example x and l^2:

$$\exp\left(-\frac{||x-l^2||^2}{2\sigma^2}\right)$$

And f_3 is the similarity between the training example x and l^3:

$$\exp\left(-\frac{||x-l^3||^2}{2\sigma^2}\right)$$

The different similarity functions are called kernels. This specific choice of similarity measure is called the Gaussian kernel denoted as:

$$k(x, l^{(i)})$$

So for l^1, the similarity between x and l^1 is

$$f_1 = similarity(x, l^{(1)}) = \exp\left(-\frac{||x-l^1||^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^{m}(x_j - l_j^{(1)})^2}{2\sigma^2}\right)$$

Suppose that x is close to $l^{(1)}$: Then the euclidean distance $||x-l^1||^2$ will be close to 0. So

$$f_1 \approx \exp\left(-\frac{0}{2\sigma^2}\right) \approx 1$$

Conversely, if x is far from l^1:

$$f_1 \approx \exp\left(-\frac{(\text{ large number })^2}{2\sigma^2}\right) \approx 0$$

So what these $f_1, f_2, f_3, ...$ features do is to measure how similar x's are with one of the l. The feature f is close to 1 when x is close to l and f is close to 0 when x is far from to l. Each l^1, l^2, l^3 defines a new feature f_1, f_2, f_3. So given a new example x we can compute three new featuresf_1, f_2, f_3 given l^1, l^2, l^3.

How do we choose the l^i and how many should we choose?

Suppose that we have the training set shown in figure (2.22). The idea is to take the m examples (red or blue) and for every training example that we have we put l_i where

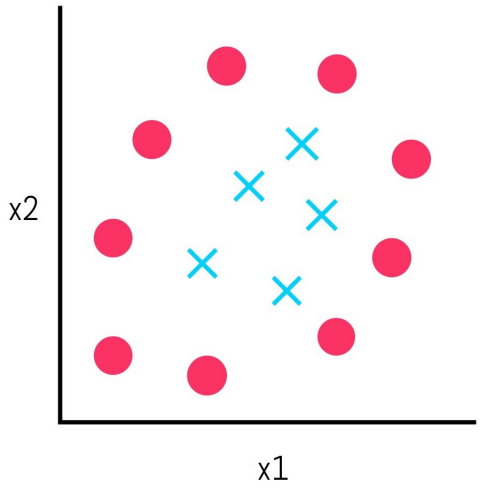


Fig. 2.22: Training set containing red circles and blue crosses. For every training point $x^{(i)}$ we compute an $l^{(i)}$.

$i = 1, 2, ..., m$ as exactly the same location as the training examples. So we have m l's: $l^1, l^2, ..., l^m$ for m training examples.

So given:

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})$$

we choose

$$l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, ..., l^{(m)} = x^{(m)}$$

Given an example x (in the training or in the test set):

$$f_1 = \text{similarity } (x, l^{(1)})$$

$$f_2 = \text{similarity } (x, l^{(2)})$$

$$...$$

$$f_m = \text{similarity } (x, l^{(m)})$$

we get a feature vector:

$$f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$$

Suppose we have a training example $(x^{(i)}, y^{(i)})$, the features that we will compute are:

$$f_1^{(i)} = \text{similarity } (x^{(i)}, l^{(1)})$$

$$f_2^{(i)} = \text{ similarity } (x^{(i)}, l^{(2)})$$

...

$$f_m^{(i)} = \text{ similarity } (x^{(i)}, l^{(m)})$$

Somewhere in the middle we will have:

$$f_i^{(i)} = \text{ similarity } (x^{(i)}, l^{(i)})$$

where $l^{(i)} = x^{(i)}$. So $f_i^{(i)}$ will be the similarity between x and itself. By using the Gaussian kernel we have:

$$f_i^{(i)} = \text{ similarity } (x^{(i)}, l^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1$$

So at least one of the features for this training example is 1.

We can also take all of these m features and group then into a feature vector. So instead of $x^{(i)}$ we have

$$f^{(i)} = \begin{bmatrix} f_1^{(i)} \\ f_2^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix}$$

where $f^{(i)}$ the the new feature vector. These new feature vectors are also called support vectors.

2.4.6 Support vector machines with Kernels

Given a set of parameters θ then for a value x we compute the feature $f \in R^{m+1}$ and we predict $y = 1$ if $\theta^T f \geq 0$. Note that the new feature vector is called also support vector because it can give us which $f^{(i)}$ plays a bigger role in classifying x. The $f^{(i)}$ which are near the decision boundary have a greater value ($f^{(i)} \approx 1$) than those how are not close ($f^{(i)} \approx 0$).

The parameters θ can be computed from the support vector machine cost function

$$min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \sum_{j=1}^{p} |b_j|$$

But instead of using the $x^{(i)}$ we use the new features $f^{(i)}$ so we have:

$$min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T f^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T f^{(i)}) \right] + \sum_{j=1}^{m} |b_j|$$

By solving this minimization problem we get the parameters of θ for the support vector machine. Note also what the number of new features is m (the amount of the training example).

2.4.7 Other Kernels

Until now we have seen the Gaussian kernel and the linear kernel (when we basically don't use any kernel).

For the **linear kernel** we predict $y = 1$ if $\theta^T x \geq 0$. This type of kernel can be used if we have a large number of features p and a small training set m. We can fit a linear decision boundary and not try to fit a very complicated non-linear function because we might not have enough data and we might risk overfitting.

The **Gaussian kernel** is useful when the feature space p is small and we have available a large training set m. This way we can fit a more complex decision boundary and separate the data better. Note also that if the features are in a different scale we need to do feature scaling before using the Gaussian kernel. This is because if we have:

$$||x - l||^2 = (x_1 - l_1)^2 + (x_2 - l_2)^2 + ... + (x_n - l_n)^2$$

If the range of the x_i is different then some $(x_i - l_i)^2$ are going to be very large and some $(x_i - l_i)^2$ very small.

Another option of kernel is the **Polynomial kernel** and here the similarity between x and l is defined as (one version of similarity measure):

$$k(x, l) = (x^T l)^2$$

or

$$k(x, l) = (x^T l)^3$$

or

$$k(x, l) = (x^T l + 1)^3$$

The general form is:

$$k(x, l) = (x^T l + \text{ constant })^d$$

where the constant and the d parameters has to be specified. If x and l are very similar to each other then the inner product between then is large.

2.4.8 Bias variance trade off for SVM

How to choose the parameter C

If C some large number, this corespondents to not using much regularization and tend to have lower bias and high variance (see chapter 1). If we use a smaller value for C we have higher bias and low variance.

How to choose the parameter σ^2 in the Gaussian kernel

If the σ^2 is large then the features f_i vary more smoothly. We have higher bias and lower variance.

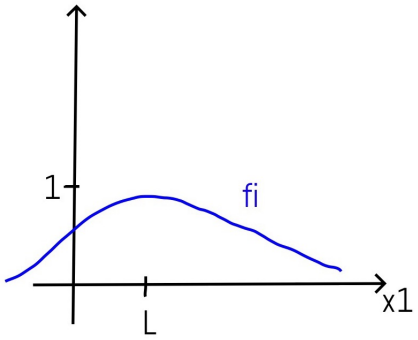


Fig. 2.23: Plot of $f^{(i)}$. If the σ^2 is large then the feature $f^{(i)}$ vary more smoothly.

If σ^2 is small the features f_i vary less smoothly. We have lower bias and higher variance.

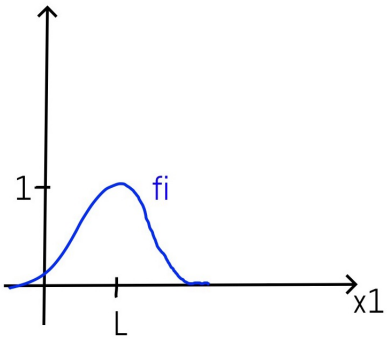


Fig. 2.24: Plot of $f^{(i)}$. If σ^2 is small the feature $f^{(i)}$ vary less smoothly.

2.4.9 Logistic regression vs SVM

When should we use logistic regression and when the support vector machine?

When the number of the feature space p is large relative to the number of training samples then the best option is to use the logistic regression or the SVM without a kernel (linear kernel) because a liner function would probably perform good and we don't have enough data to fit a more complicated nonlinear function.

If the feature space p is small and the number of training samples m is large (but not very large) then a good choice is the SVM with Gaussian kernel.

If the feature space p is small and the number of training samples m is very very large then a good choice is to create more features and then use logistic regression or SVM without a kernel. The SVM with Gaussian kernel can also be used but because of the massive training size will be slow to run.

Logistic regression and SVM without a kernel are similar algorithms and give similar performance, but depending on the implementation details, one may be more efficient than the other. The power of SVM can be seen then we use different kernels to learn complex nonlinear functions.

Chapter 3

Model Selection and Regularization methods

In Chapter 1 we described the liver cancer data set which consist of $n = 145$ subjects and a large number of independent variables (features), X_j, $j = 1, ..., 236$. Not all these features have the same contribution in a statistical model (or in the problem). In many cases the goal is to develop a model which can explain the relationship between the features and the dependent variable. For example it is more preferable to use a linear model in comparison with a complex model like support vector machines, but even in a linear model when the number of features is huge is very difficult to interpret. A severe mathematical problem is when the dimension of the data is greater than the number of the available data points. For example in linear regression when the number of the features is larger than the number of the observations there is no longer a unique least squares coefficient estimate, the variance is infinite and the method cannot be used.

There are two main options to address the issue of reducing the number of features.

- Manually select which features to keep.
- Use a model selection method or to use a regularization method.

In this chapter we will describe methods for feature selection and regularization, including subset selection and lasso.

3.1 Model Selection for Linear Regression

3.1.1 Subset Selection

Subset Selection includes Best Subset Selection and Stepwise Model Selection. We will see that best Subset selection is only useful if the number of features is relatively small. Stepwise Model Selection consists of two methods: Forward Stepwise Selection and Backward Stepwise Selection.

Best Subset Selection

Suppose that we have p features but we want to have a simpler model that contains only a subset of these p features. The first step in best subset selection is to find the best model with 0, then 1, then 2 and so on until p features. Best here is defined as having the largest R^2. Note that the value R^2 indicates how well the model fits the data. Big R^2 denotes a model that fits the data well. We end up with p models, each one of them with different size of features. The second step is to find the test error in each model and the model with the smallest test error is the final best model.

Algorithm 5 Best Subset Selection

1: Let M_0 denote the null model, which contains no features.
2: For $k = 1, 2, ..., p$:

- Fit all $\begin{pmatrix} p \\ k \end{pmatrix}$ models that contain exactly k features.
- Pick the best model among these $\begin{pmatrix} p \\ k \end{pmatrix}$ models and call it M_k. Here best is defined as having the highest R^2.

3: Select a single best model from $M_0, M_1, ..., M_p$ using the test error.

This method requires the computation from 2^p models. For a large number of p this is a problem because we are looking at an enormous number of models, and the chance of finding a model that fits our training data well, even it might not have any predictive power, is high. This can lead us to overfitting the data and high variance of the coefficient estimates. In the next two methods we will see that this problem can be reduced (but not entirely solved).

Stepwise Selection

Stepwise selection methods explore less subsets of the models in comparison with best subset selection. It includes two methods: Forward stepwise selection and backward stepwise selection.

Forward Stepwise Selection

Forward stepwise selection begins with the model M_0 which has no features. At each step it adds the feature that improves the model. At the final step, like in best subset selection we end up with $M_0, M_1, ..., M_p$ models and choose the best among them which is the model with the smallest test error.

Algorithm 6 Forward Stepwise Selection

1: Let M_0 denote the null model, which contains no features.
2: For $k = 0, 1, ..., p-1$:

- Consider all p-k models that augment the features in M_k with one additional feature.
- Choose the best among these p-k models and call it M_{k+1}. Here best is defined as having the highest R^2.

3: Select a single best model from $M_0, ..., M_p$ using the test error.

Backward Stepwise Selection

Backward selection goes the opposite way from forward selection. We begin with the model M_p that contains all the features. At each step we remove the feature which arise a model with higher R^2 value and name it M_{p-1}. At the last step we choose among the the M_p, M_{p-1}, M_0 models the best one which is the model with the smallest test error.

Algorithm 7 Backward Stepwise Selection

1: Let M_p denote the full model that has all p the features.
2: For $k = p, p-1, ..., 1$:

- Consider all k models that contain all but one of the features in M_k, for a total of $k-1$ features.
- Choose the best among these k models and call it M_{k-1}. Here best is defined as having the highest R^2.

3: Select a single best model from $M_0, ..., M_p$ using the test error.

3.2 Model Selection for Logistic Regression

In Logistic regression we cannot use the R^2 method because we have a binary response variable. Instead we use two other methods called score test which test for inclusion of a term, and the Wald test which can be used to test for exclusion of a term. Neither of these require iterative fitting and are based on the maximum-likelihood fit of the current model. Note also that for finding the best model between models with the same feature number k-fold cross validation can also be used.

3.2.1 Wald test

The Wald test is used to find out if the independent variables in a model are statistical significant, or in other words if they add information to the model. If they add nothing, the

independent variables can be deleted without affecting the model in any meaningful way. For the Wald test the hypothesis is:

$$H_0 : \theta = \theta_0$$

$$vs$$

$$H_1 : \theta \neq \theta_0$$

If the null hypothesis is rejected, it suggests that the variable with coefficient θ can be removed without much harm to the model fit.

The Wald Test statistic formula is:

$$W = (\hat{\theta} - \theta_0)[Var(\hat{\theta})]^{-1}(\hat{\theta} - \theta_0)$$

Where: $\hat{\theta}$ is the maximum likelihood estimator.

Under the null hypothesis W follows a χ^2 distribution. The number of degrees of freedom depends from the number of parameters we test. If we test p parameters we have p degrees of freedom.

3.2.2 Score test

The Lagrange multiplier or score test, as with the Wald test, requires estimating only a single model. The difference is that with the score test the model that is estimated does not include the parameter(s) of interest. The test statistic is calculated based on the slope (or score) of the likelihood function at the observed values of the independent variables in the model. The scores are then used to estimate the improvement in the model fit if additional independent variables were included in the model. The test statistic is the expected change in the χ^2 statistic for the model if independent variable(s) is/are added to the model.

Suppose l is the log-likelihood. We build the score vector S by evaluating the derivative of the likelihood with respect to each of the parameters of the model:

$$S = \left[\frac{\partial l}{\partial \theta_1}, \frac{\partial l}{\partial \theta_2}, ...\right]^T$$

The Score test formula is:

$$LM = S(\theta_0)^T[Var(\theta_0)]^{-1}S(\theta_0)$$

Under the null hypothesis LM follows a χ^2_p distribution, where p is the number of parameters. Note that none of these terms involve the unconstrained maximum likelihood estimate of the parameters. $S(\theta_0)$ is just the values of the parameters under the null hypothesis. This has the benefit of not needing to estimated the maximum likelihood for all parameters, which can cause problems when we have many parameters (thous high dimensional data)

3.2.3 Subset Selection

Best subset selection and Stepwise selection for logistic regression is the same with linear regression except that we use the Wald and the Score test inserted of the R^2. As we mentioned before, for including an independent variable (feature) we use the score test and for excluding an independent variable we use the Wald test.

3.3 Lasso

The Least Absolute Shrinkage and Selection Operator, or Lasso is a shrinkage method introduced by Tibshirani (1996). This method adds a penalty term in existing methods (linear regression, logistic regression, SVM), in order to shrink the coefficients towards zero and shrink some of those to exactly zero. They are very useful when we have a huge number of features.

The shrinkage methods (or in other words, the regularization methods) improve the performance of the model of the unseen data (test set) by not picking up noise in the training set. These methods put a penalty against complexity, the more features we have in our model, the higher the complexity is. Increasing the regularization term, we have a higher penalty at large coefficients.

The goal is to find a model which performs good to the test set and not memorising the training set.

The Lasso uses an l_1 penalty:

$$||\theta||_1 = \sum |\theta_j|$$

The l_1 penalty has the effect of forcing some coefficient estimates to be exactly equal to zero when the regularisation parameter λ is sufficiently large. Hence, like subset selection, the lasso performs variable selection.

3.3.1 Lasso for logistic regression

As we saw in chapter 2, for logistic regression the goal is to minimize the cost function. If we want to regularize the cost function with Lasso, we add the additional term $\lambda \sum_{j=1}^{p} |\theta_j|$ that increases as the value of the parameter weights increase. The λ term controls the regularization strength. When λ is sufficiently large some of the coefficient can be exactly 0. This means that by applying Lasso the model uses only a subset of the variables. For selecting a good value for λ cross-validation can be used.

The goal is to minimize the cost function with the additional regularisation term and to choose the optimal λ value.

The cost function of Logistic Regression with the Lasso regularization term is:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))] + \lambda\sum_{j=1}^{p}|\theta_j|$$

3.3.2 Lasso for support vector machines

As we saw in Chapter 2 the cost function for support vector machines is

$$\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right]$$

If we add the regularisation term the cost function becomes

$$\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \lambda\sum_{j=1}^{p}|\theta_j|$$

Note that $\frac{1}{m}$ is a constant so we can ignore it in the cost function because it will not change the result of the minimum number of θ.

Going back in logistic regression, if we set

$$A = \frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))]$$

which is the cost that comes from the training set and

$$B = \sum_{j=1}^{p}|\theta_j|$$

which is the regularization term. Then we can control the trade of of these two terms. We want to minimize $A + \lambda B$. By setting different values of λ we can trade off the weight of how much we want to fit the training set well (minimizing A) versus how much we want to keep the variables in the model (minimizing B).

For support vector machine we use (just by convention) instead of λ a parameter C where:

$$CA + B$$

and so the cost function becomes:

$$C\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \sum_{j=1}^{p}|\theta_j|$$

So if we set to C a small value then that corresponds to giving the second term a much larger rate than the first one. Note that in Chapter 4 we will use the notation C in the first part of the cost function for both in Logistic regression and Support Vector Machines.

Chapter 4

Results

In this chapter we apply the methods described in the previous chapters to our liver data set (see Chapter 1). Our goal is to correctly predict if a person is diseased or not. We apply the logistic regression model and the support vector machines in order fit a model to out data for this classification task.

We start with the Logistic regression model using subset selection. From the best subset selection, forward stepwise selection and backward stepwise selection we use only the forward stepwise selection algorithm because using the other two methods requires to fit a logistic regression model using all 236 features in a data set with 145 subjects. This is not possible as we need at least p points to determine p free parameters. Then we continue with the Logistic regression model using Lasso and finally we fit a support vector machines algorithm with the regularisation term (Lasso). We start the support vector machines (SVM) subsection by using only 2 features (SVM with a gaussian kernel) and then all features (SVM with a linear kernel).

Before fitting any model we scale all the independent variables. We split the data into test and training set and we keep the percentage of diseased and healthy subjects the same as in the original data set. At the end of each method we fit the same test set in order to evaluate the final performance of the model.

4.1 Logistic regression

In our data set the dependent variable is binary (diseased-healthy) therefore we will build a binary logistic regression model. One assumption that is required is that the independent variables (the features) have no high dependence between them and only the meaningful features (a subset of them) should be included in the model.

In the figure (4.1) we can see all the correlations between the marker by colour. The yellow colour means that there is a negative correlation starting from -0.6. As we approach the light-blue colour the correction is near 0. As the colour is getting dark blue the correlation tends to 1.

Markers that are close to the diagonal have in general a higher correlation in contrast with the other markers. On the diagonal the correlation is 1 as we have $corr(x_i, x_i) = 1$ for each marker.

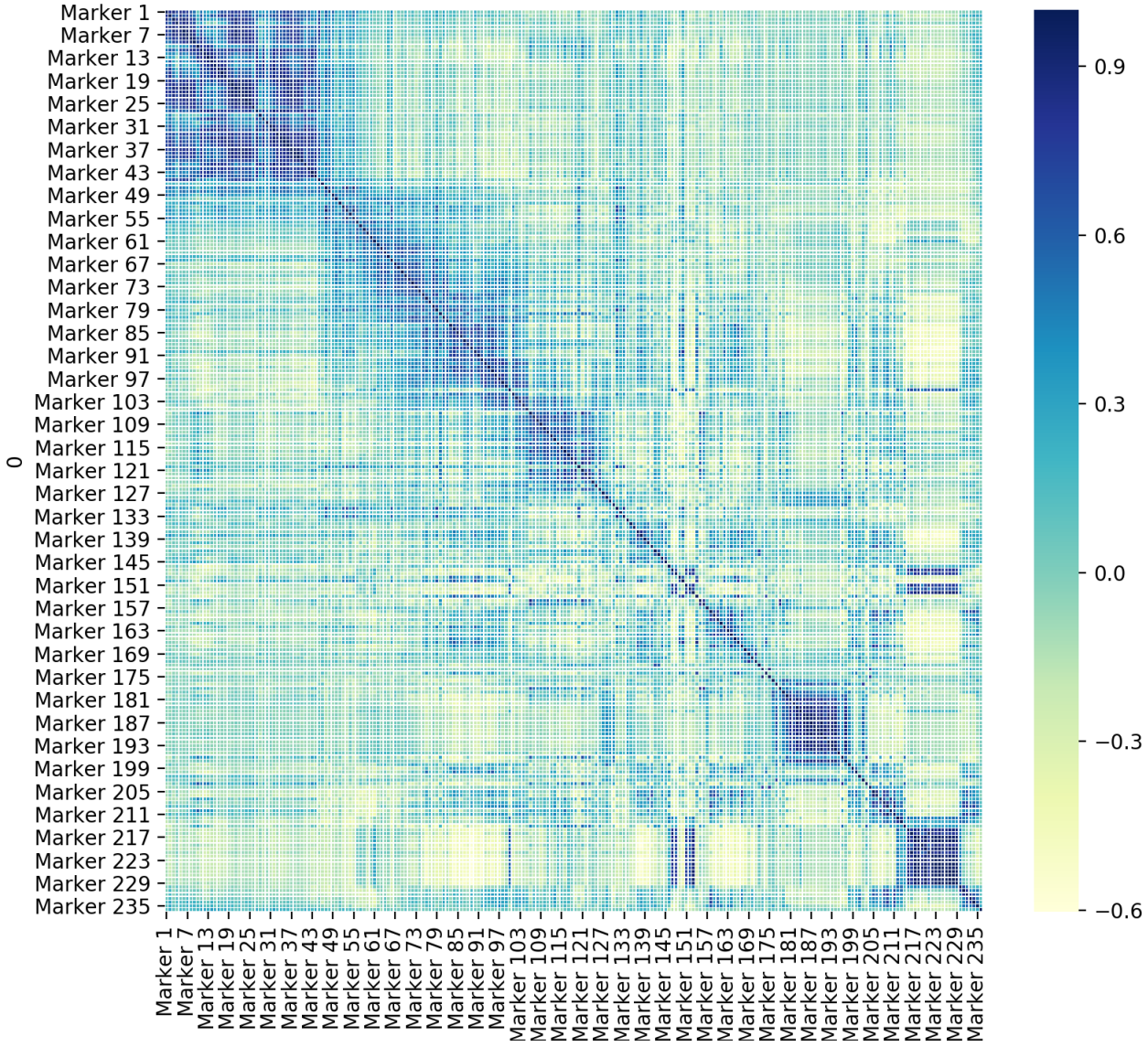


Fig. 4.1: Correlation between variables scaled by colour. The yellow colour means that there is a negative correlation -0.6 and as we approach the light-blue colour the correction is near 0. As the colour is getting dark blue the correlation tends to 1.

As we have discussed in earlier chapters, by using all markers several problems arise. The features space is very large and we can easily overfit the data. We use forward selection and lasso to solve this problem.

4.1.1 Forward selection

By using only the training set, we apply the forward subset selection algorithm with 3-fold cross validation. By using cross validation we can estimate more accurate the accuracy of the models with different number of features.

We use the forward selection algorithm and we select the best model for each number of features. From these 145 models we obtain the Average Score (the percentage of accurate predictions from the validation sets) by using 3-fold cross validation. In figure (4.2) we see all the average scores for each number of features. Note that the best model for each number of features is selected based on the average score using 3-fold cross validation.

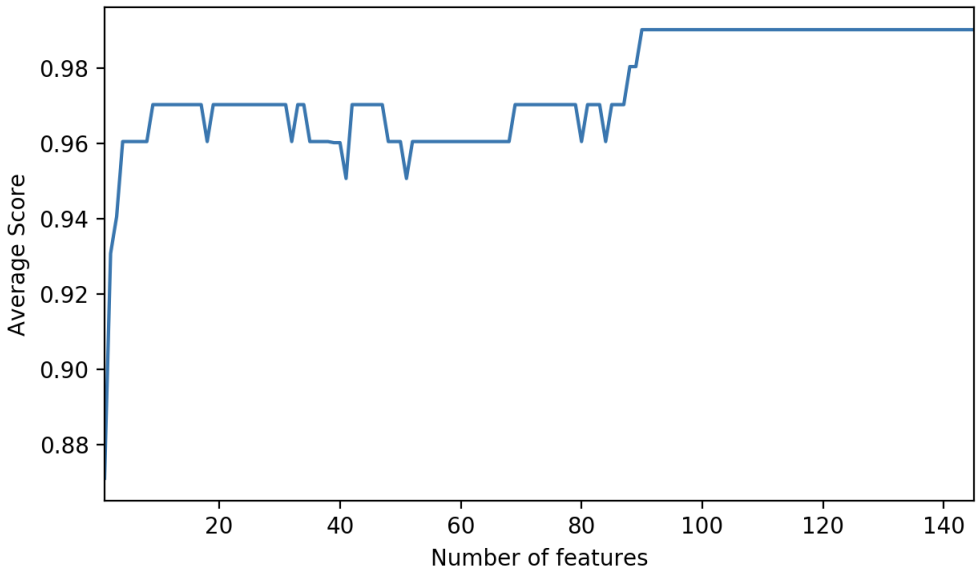


Fig. 4.2: Forward feature selection with 3-fold cross validation. We select the best model for each number of features. From the 145 models we plot the average score from the validation sets.

In figure (4.2) we see that a good number of features to choose is 9 with average score 0.97 and for this model the best combination of Markers are: marker 18, 26, 47, 68, 71, 134, 150, 184, 197.

The correlation between these markers is shown in figure (4.3)

We use these markers in the logistic regression model. From the test set we also use only these 9 markers.

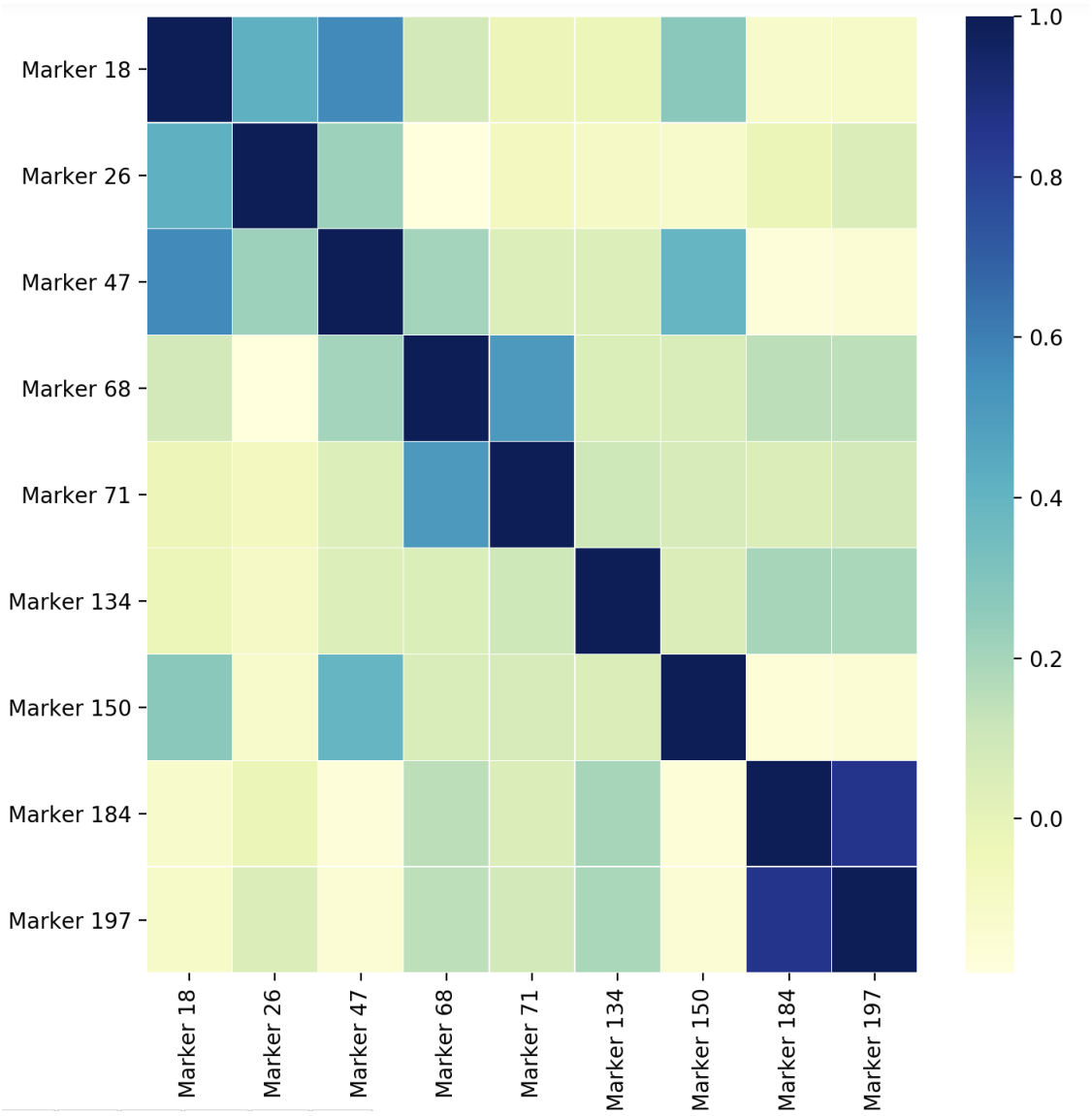


Fig. 4.3: Correlation between variables scaled by colour. The light yellow colour means that there is a -1 negative correlation and as we approach the yellow colour the correlation is near 0. As the colour is getting dark blue the correlation tends to 1.

Taking into account the true diseased status and the probability of disease given the 9 markers we create the classification table with 0.5 as the cut-off value.

In Table 4.1 we see that 13 healthy people out of 16 are correctly classified as healthy, 5 subjects are incorrectly classified as non-diseased, 23 subjects out of 28 are correctly classified as diseased and 3 are incorrectly classified as diseased.

Table 4.1: Classification Table

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	13	3	16
`Disease, D=1`	5	23	28
`Total`	18	26	44

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{23}{28} = 0.82$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{3}{16} = 0.18$

The ROC curve is given in figure (4.4) and the AUC score is equal with 0.84.

The accuracy score is 81%

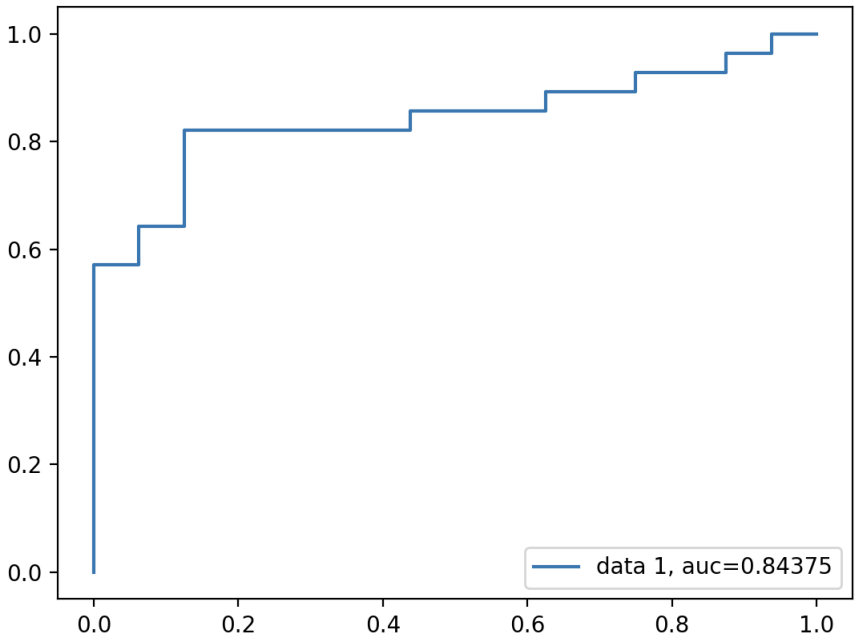


Fig. 4.4: ROC curve using a logistic regression model with 9 markers which were selected using forward feature selection.

4.1.2 Lasso

The cost function of the Logistic regression with the regularisation term (Lasso) as we saw in previous Chapters is:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))] + \lambda\sum_{j=1}^{p}|\theta_j|$$

In this example we suppose that the regularisation term is at the left side of the cost function and denoted as C:

$$J(\theta) = -C\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))] + \sum_{j=1}^{p}|\theta_j|$$

Thus if C is sufficiently small some of the coefficient can be exactly zero and vice versa. In the case where $p > m$ the lasso selects at most m variables before it regularizers the parameters. The goal is to minimize the cost function as a function of θ and to choose the optimal value C. As the value of the regularization term changes the values of the coefficients changes also. In figure (4.5) the models are ordered from strongest regularized to least regularized for $C \in (0, 20)$. The 145 coefficients are plotted against the C value. On the left-hand side of the figure we have a strong regularization and all the coefficients are exactly 0. When regularization gets progressively looser, coefficients can get non-zero values one after the other.

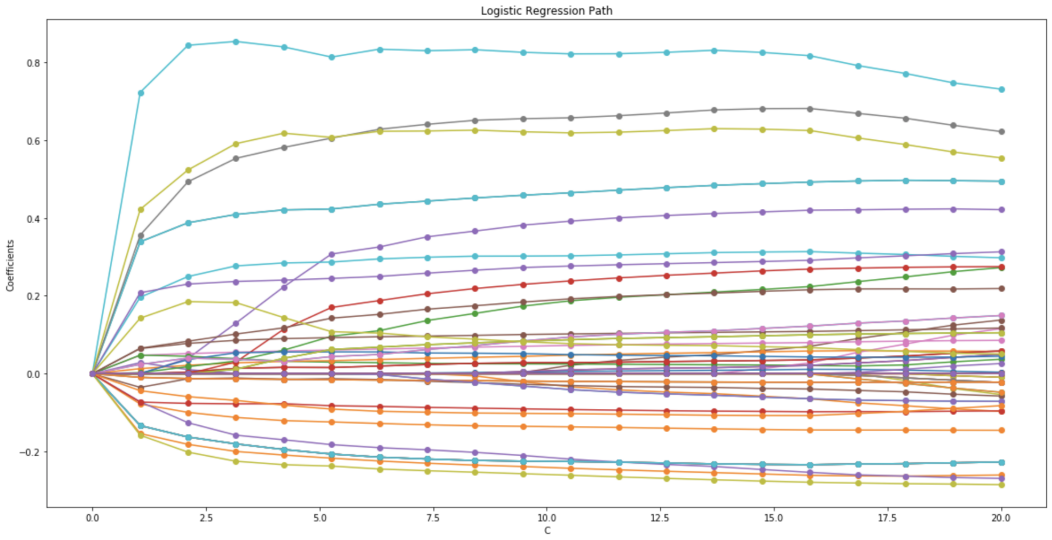


Fig. 4.5: The 145 coefficients are plotted for $C \in (0, 20)$. On the left-hand side of the figure we have a strong regularization and all the coefficients are exactly 0. When regularization gets progressively looser, coefficients can get non-zero values one after the other.

In figure (4.6) we can visualise better the results for $C \in (0, 1)$. On the left side of the plot some coefficients are already zero because the maximum value of C here is 1, and so we only see 23 non-zero values of the coefficients.

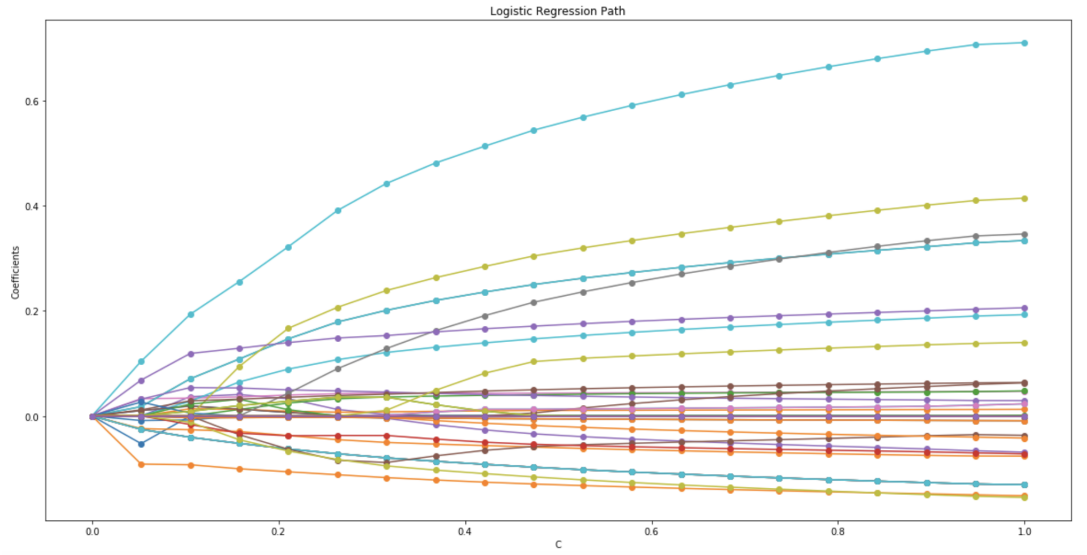


Fig. 4.6: On the left side of the plot some coefficients are already zero because the maximum value of C here is 1, and so we only see 23 non-zero values of the coefficients.

We use a set of numbers for the value of C, $C = 0.01, 0.04, 0.07, 0.1, 0.4, 0.6, 0.8, 1$ and use 3-fold cross-validation to find the optimal value of C, which is $C = 0.04$. The non-zero coefficients corresponds to the 19 markers: 8, 84, 91, 92, 121, 122, 139, 145, 150, 151, 156, 163, 194, 196, 197, 206, 207, 217, 218.

The correlation between these markers is shown in figure (4.7)

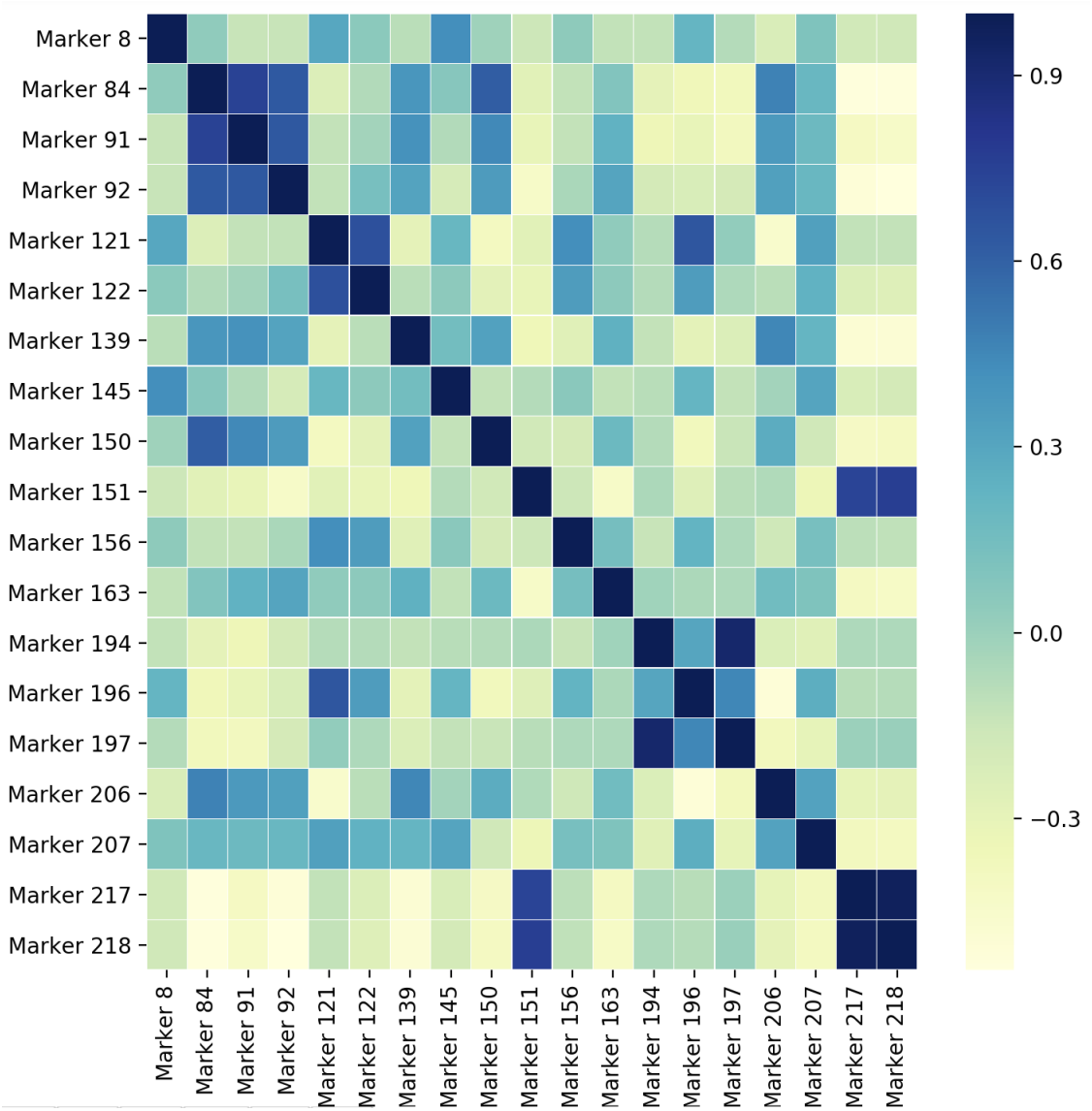


Fig. 4.7: Correlation between variables scaled by colour. The light yellow colour means that there is a -4 negative correlation and as we approach the light blue colour the correlation is near 0. As the colour is getting dark blue the correlation tends to 1.

From the test set we choose these 19 markers. In Table 4.2 we see that 13 healthy people from 16 are correctly classified as non-diseased, 3 subjects are incorrectly classified as non-diseased, 25 subjects from the 28 are correctly classified as diseased and 3 are incorrectly classified as diseased.

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{25}{28} = 0.88$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{3}{16} = 0.18$

Table 4.2: Classification Table

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	13	3	16
`Disease, D=1`	3	25	28
`Total`	16	28	44

The accuracy score is 86%.

The ROC curve is shown in figure (4.8)

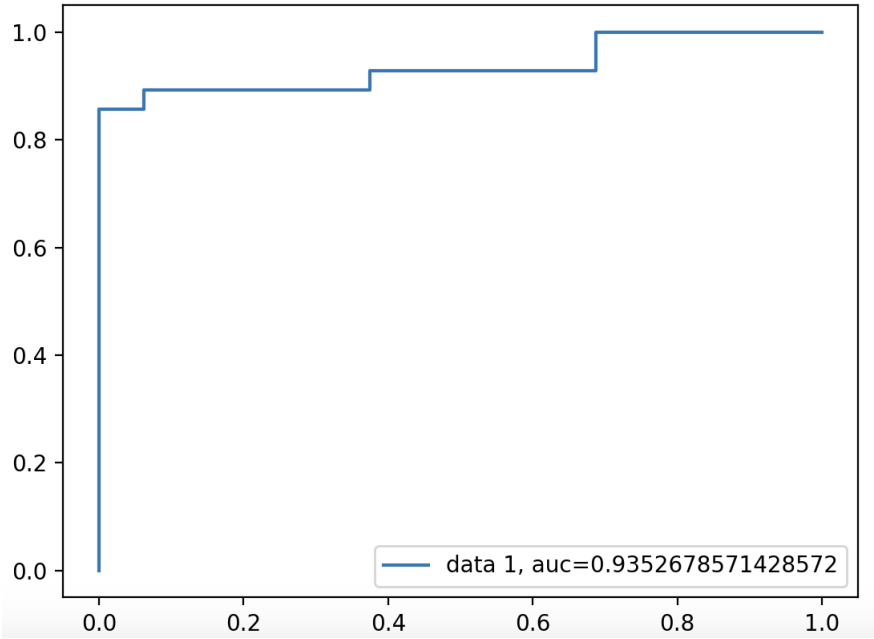


Fig. 4.8: ROC curve using a logistic regression model with 19 markers which were selected using Lasso.

AUC score is 0.93. AUC score 1 represents perfect classifier, and 0.5 represents a worthless classifier.

4.2 Support Vector Machines

In this section we start by selecting two markers, marker 40 and marker 100. We separate the data into training set and test set which is the 20% of the data.

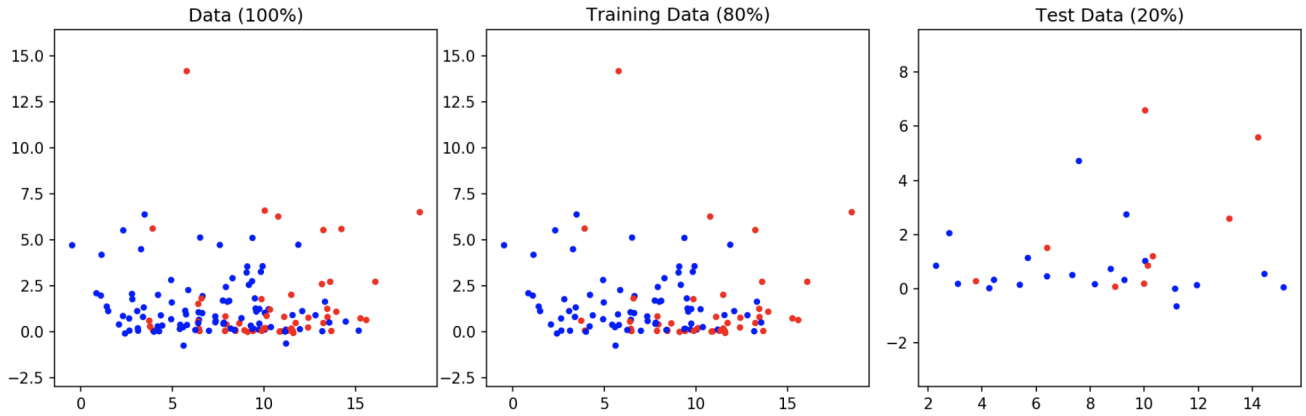


Fig. 4.9: Scatter plot of marker 40 and marker 100. In the first plot are is the hole data set, in the second plot is the training set which consists of the 80% of the data points, in the third plot is the test set which consists of the remaining 20% of the data points. Notice that the data points are not linearly separable.

The data are not linearly separable. We use two features and we have 145 data points, therefore we will use a Gaussian kernel. For different values of the regularisation parameter C and the variance σ^2 we get different decision boundaries. For example if we use $C = 1000$ and $\sigma^2 = 0.01$ we get the decision boundary shown by the solid line and the margins shown with the dotted lines in figure (4.10).

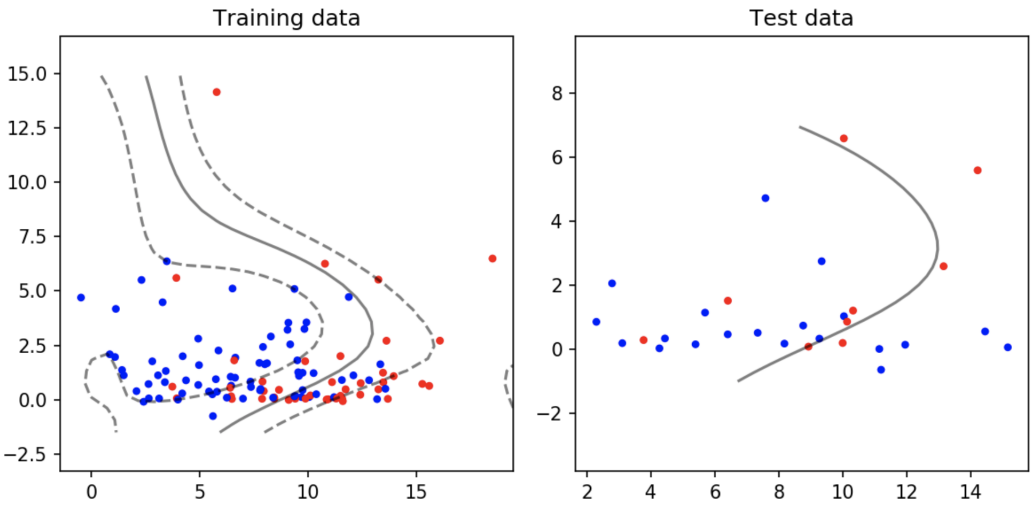


Fig. 4.10: By using a Gaussian kernel with $C = 100$ and $\sigma^2 = 0.01$ we get the decision boundary shown by the solid line and the margins shown with the dotted lines.

The support vectors are denoted with black points in figure (4.11)

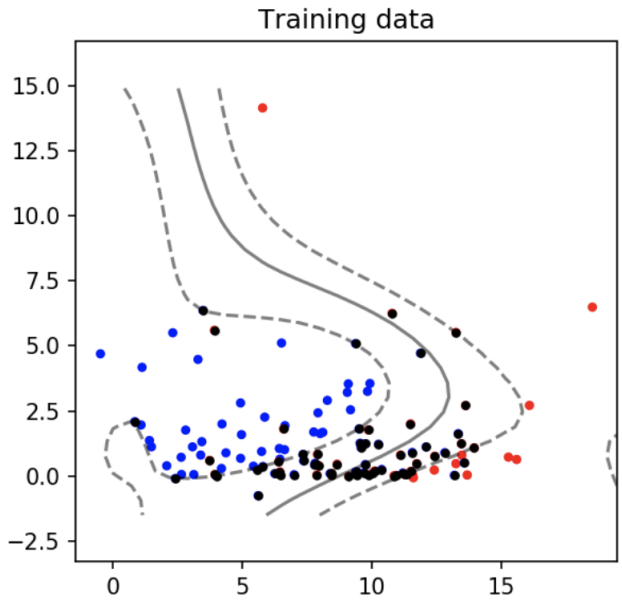


Fig. 4.11: The support vectors are shown in black.

But is this the best combination of the C and the σ^2 value? To answer this question we use 3-fold cross validation and select the best combination of values for $C = 0.1, 0.5, 10, 100$ and $\sigma^2 = 1, 0.1, 0.01, 0.001$.

The best combination that arises is for $C = 10$ and $\sigma^2 = 0.001$. The decision boundary together with the margin are shown in figure (4.8) on the left side. On the right side we see how the test set gets separate from the optimal SVM decision boundary.

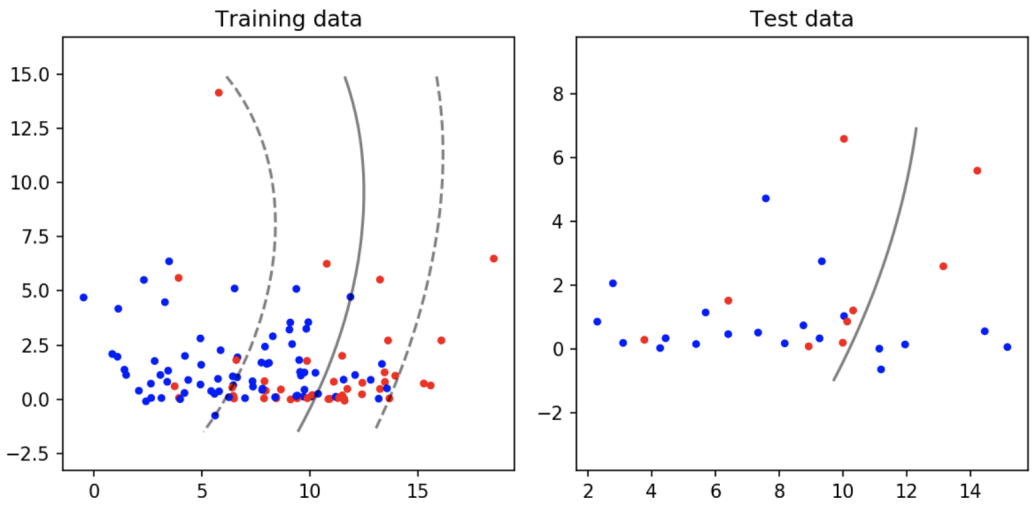


Fig. 4.12: On the left side we see the decision boundary together with the margin. On the right side we see how the test set gets separate from the optimal SVM decision boundary.

SVM using all the Markers

The cost function of the support vector machines with the lasso regularisation term is

$$C\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \sum_{j=1}^{p}|\theta_j|$$

Our liver data set has 145 subjects and 236 markers, which means that by using a more complex kernel there is a high probability of overfitting therefore we will use a linear kernel. We separate the data into training and test set (the test set is the same as in logistic regression). Again by using different values of C (now we don't have to choose a value for σ^2 because we use a linear kernel) we get a different decision boundaries.

As before we use 3-fold cross validation and for $C = 0.01, 0.1, 0.3, 0.5, 0.8, 1$ the best choice is $C = 0.01$. To evaluate our model we use the classification table on the test set. We use the predicted $y's$ for the $X's$ from the test set and the true status y.

In Table 4.3 we see that 15 healthy people out of 16 are correctly classified as non-diseased, 2 subjects are incorrectly classified as non-diseased, 26 subjects from the 28 are correctly classified as diseased and 1 is incorrectly classified as diseased.

Table 4.3: Classification Table

	Test result		
True disease status	Negative, Y=0	Positive, Y=1	Total
`Healthy, D=0`	15	1	16
`Disease, D=1`	2	26	28
`Total`	17	27	44

- The True Positive Rate (sensitivity) is equal with: $TPR = \frac{26}{28} = 0.92$
- The False Positive Rate (1-specificity) is equal with: $FPR = \frac{1}{16} = 0.06$

And the accuracy score is $93,18\%$.

The number of support vectors for each class is 16 for the diseased and 13 for the non diseased.

4.3 Conclusions

In order to classify the healthy and the diseased subjects from the liver data set Logistic Regression and Support Vector Machines were applied. Using forward feature selection and Lasso we reduce the number of features.

First we split the liver data set into test and training set and we keep the percentage of diseased and healthy subjects the same as in the original data set (stratification). At each method we use the test set in order to evaluate the final performance of the model using the accuracy score.

Using the Logistic regression algorithm with forward feature selection we use 3-fold cross validation for evaluating the models with different number of features. We choose the model with average score 0.97 which corresponds to a model with 9 features. The 9 markers are: 18, 26, 47, 68, 71, 134, 150, 184, 197. The accuracy score using the test set is 81%.

In Logistic regression with Lasso using 3-fold cross validation we found that the optimal value of the regularisation parameter C is 0.04. The 19 Markers with non-zero coefficients are marker: 8, 84, 91, 92, 121, 122, 139, 145, 150, 151, 156, 163, 194, 196, 197, 206, 207, 217, 218 (with only marker 150 and 197 be common from the previous method). The accuracy score using the test set is 86%.

Continuing with Support Vector Machines first we used two random markers and applied Support Vector Machines with a Gaussian Kernel. By choosing different values of C and σ^2 we saw that the decision boundary varies. Using all 236 markers we choose to use the linear kernel because for fitting a more complicated nonlinear function usually requires more subjects than features. The accuracy score using the test set is 93.18%.

Bibliography

[1] Friedman, J., Hastie, T., Tibshirani, R. (2001) 'The elements of statistical learning' (Vol. 1, No. 10). *New York: Springer series in statistics.*

[2] James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). 'An introduction to statistical learning' (Vol. 112, p. 18). *New York: springer.*

[3] Zou, H., Hastie, T. (2005). 'Regularization and variable selection via the elastic net'. *Journal of the royal statistical society: series B (statistical methodology),* 67(2), 301-320.

[4] Andrew Ng, (2005). 'Machine Learning by Stanford University' https://www.coursera.org/learn/machine-learning?

[5] Pepe, M. S. (2003). 'The statistical evaluation of medical tests for classification and prediction'. *Medicine.*

[6] Wang, Z., Chang, Y. C. I. (2010). 'Marker selection via maximizing the partial area under the ROC curve of linear risk scores'. *Biostatistics,* 12(2), 369-385.

[7] Tibshirani, R. (1996). 'Regression shrinkage and selection via the lasso'. *Journal of the Royal Statistical Society: Series B (Methodological),* 58(1), 267-288.

[8] Cortes, C., Vapnik, V. (1995). 'Support-vector networks'. *Machine learning,* 20(3), 273-297.

[9] Fahrmeir, L., Kneib, T., Lang, S., Marx, B. (2013). 'Regression: models, methods and applications'. *Springer Science Business Media.*

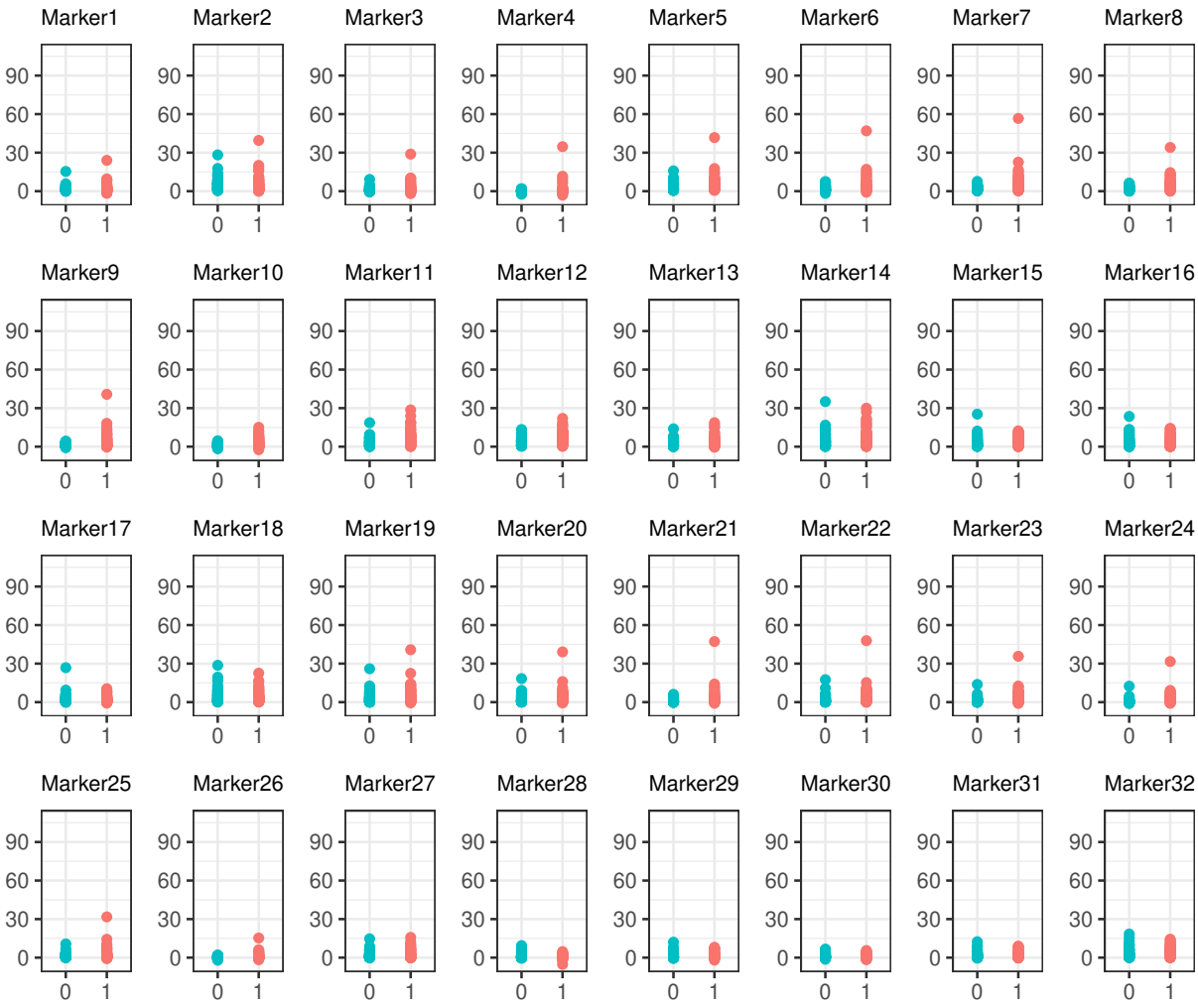
[10] Hastie, T., Tibshirani, R., Wainwright, M. (2015). 'Statistical learning with sparsity: the lasso and generalizations'. *Chapman and Hall/CRC.*

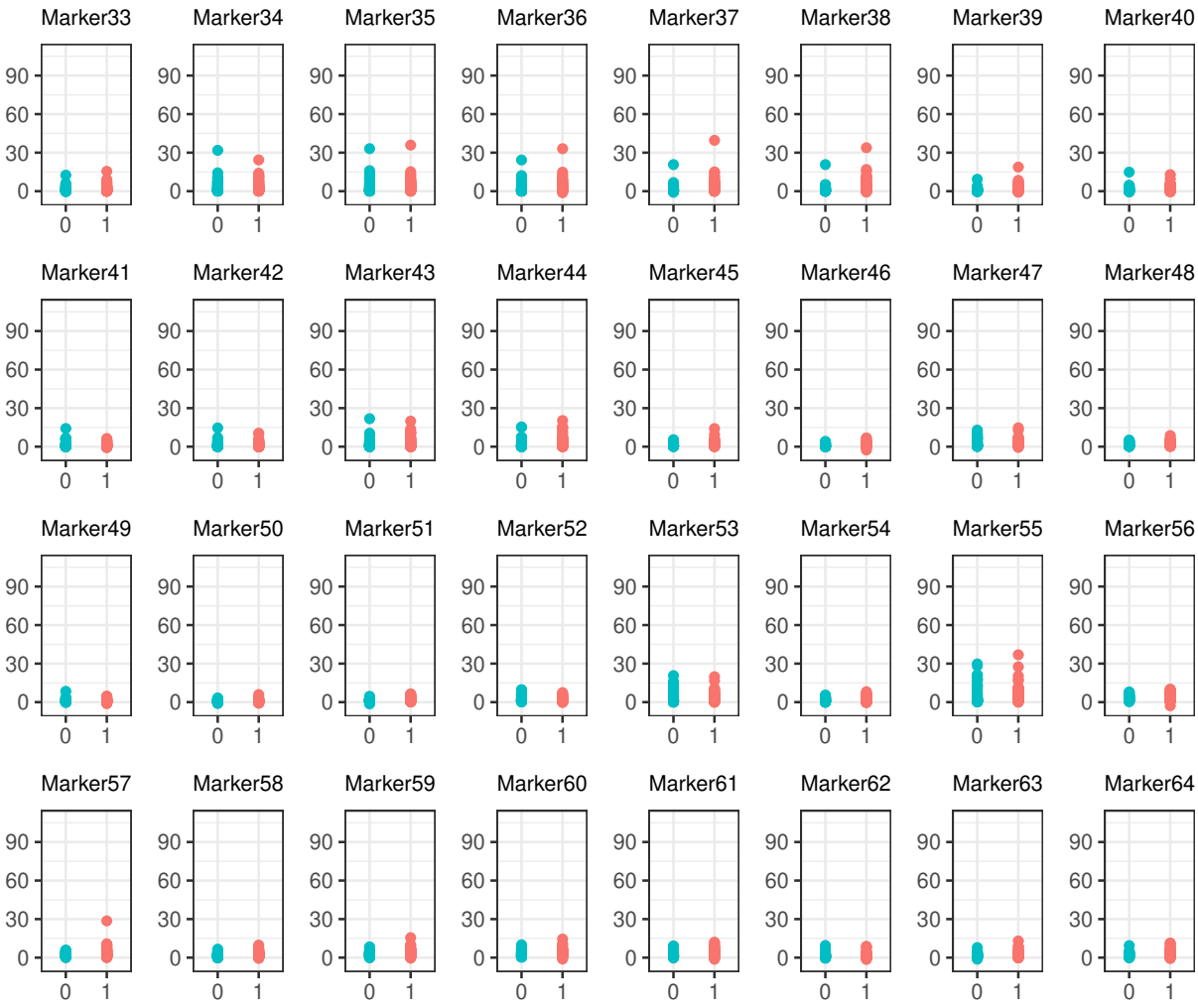
Chapter 5

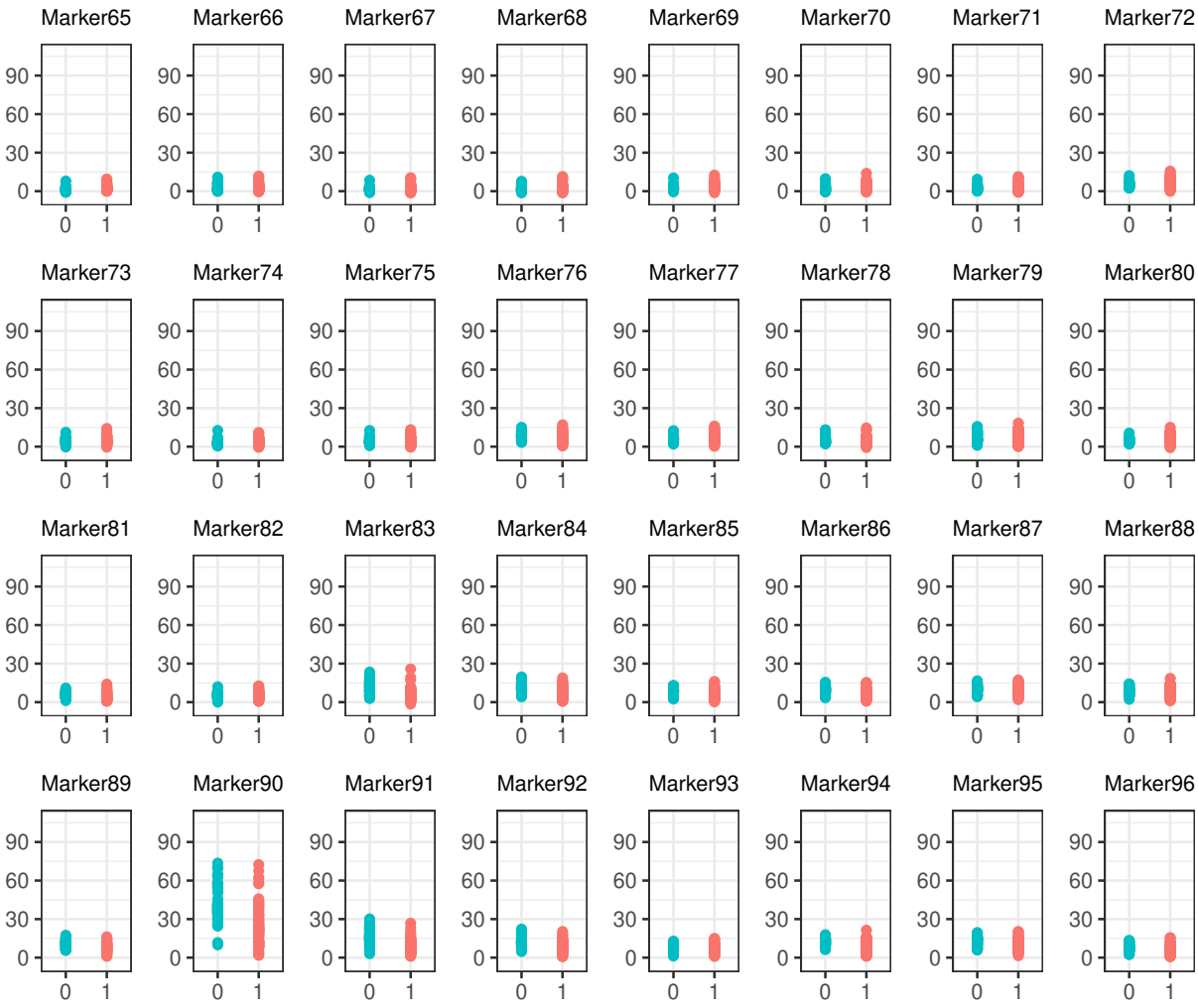
Appendix

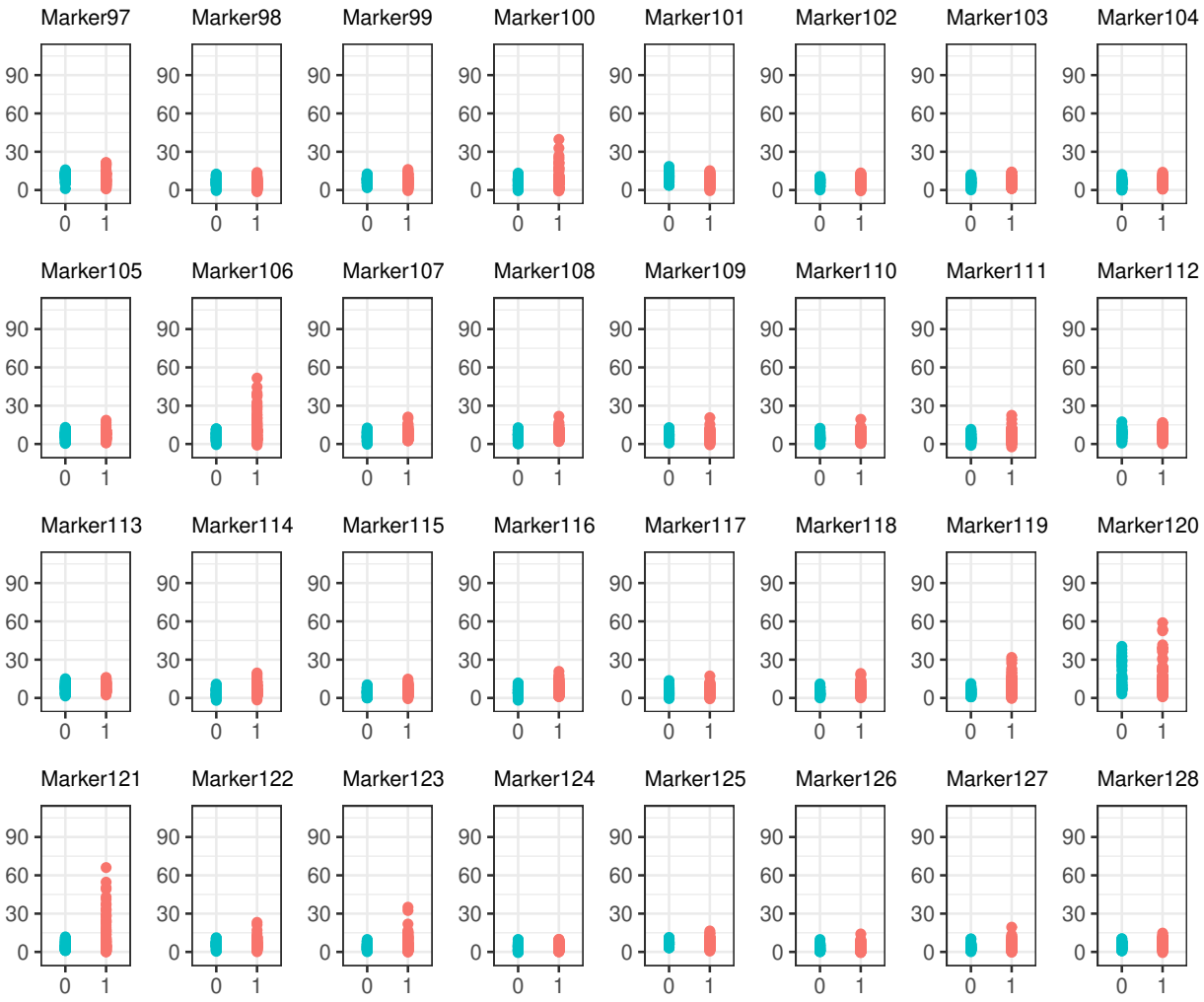
Scatter Plot

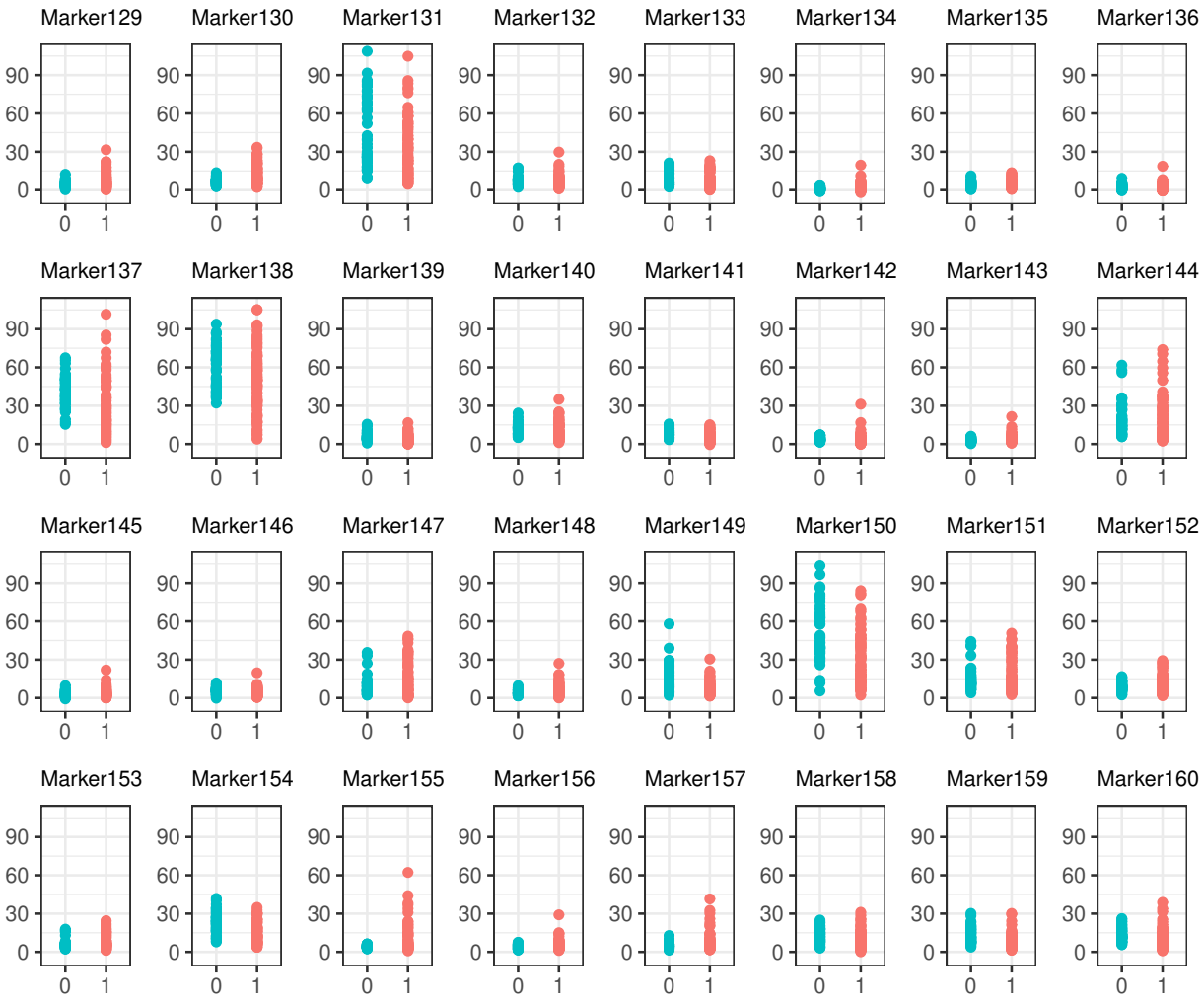
In the graphs below, the diseased group is denoted as 1 and the non-diseased group as 0. We demonstrate a scatter plot, which at the x aces has the 1-0 group and at the y aces has one of the 236 Markers.

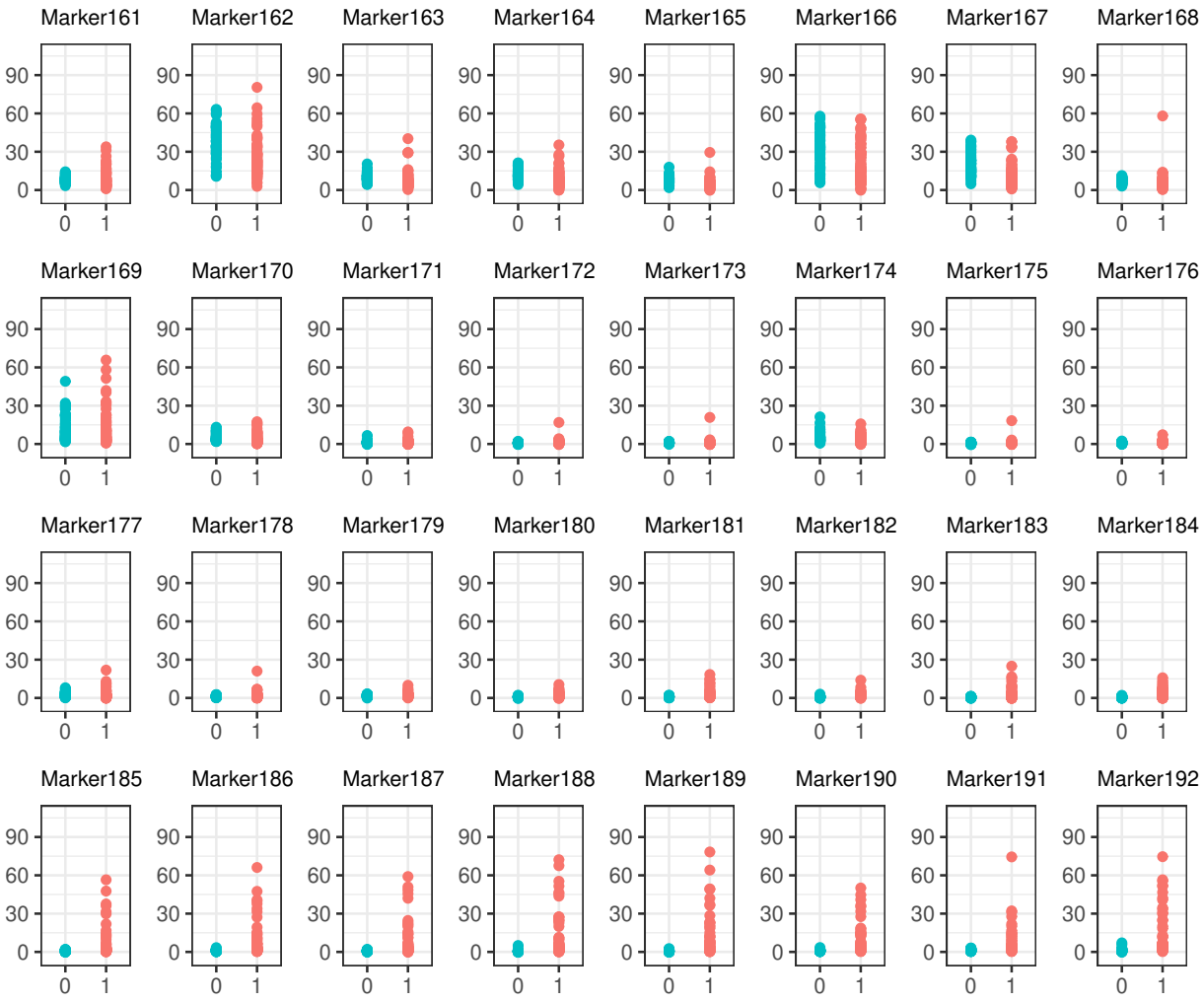

Marker1
Marker2
Marker3
Marker4
Marker5
Marker6
Marker7
Marker8
Marker9
Marker10
Marker11
Marker12
Marker13
Marker14
Marker15
Marker16
Marker17
Marker18
Marker19
Marker20
Marker21
Marker22
Marker23
Marker24
Marker25
Marker26
Marker27
Marker28
Marker29
Marker30
Marker31
Marker32
90
60
30
0
0
1


Marker33
Marker34
Marker35
Marker36
Marker37
Marker38
Marker39
Marker40
Marker41
Marker42
Marker43
Marker44
Marker45
Marker46
Marker47
Marker48
Marker49
Marker50
Marker51
Marker52
Marker53
Marker54
Marker55
Marker56
Marker57
Marker58
Marker59
Marker60
Marker61
Marker62
Marker63
Marker64
90
60
30
0
0
1


Marker65
Marker66
Marker67
Marker68
Marker69
Marker70
Marker71
Marker72
Marker73
Marker74
Marker75
Marker76
Marker77
Marker78
Marker79
Marker80
Marker81
Marker82
Marker83
Marker84
Marker85
Marker86
Marker87
Marker88
Marker89
Marker90
Marker91
Marker92
Marker93
Marker94
Marker95
Marker96
90
60
30
0
0
1


Marker97
Marker98
Marker99
Marker100
Marker101
Marker102
Marker103
Marker104
Marker105
Marker106
Marker107
Marker108
Marker109
Marker110
Marker111
Marker112
Marker113
Marker114
Marker115
Marker116
Marker117
Marker118
Marker119
Marker120
Marker121
Marker122
Marker123
Marker124
Marker125
Marker126
Marker127
Marker128
90
60
30
0
0
1


Marker129
Marker130
Marker131
Marker132
Marker133
Marker134
Marker135
Marker136
Marker137
Marker138
Marker139
Marker140
Marker141
Marker142
Marker143
Marker144
Marker145
Marker146
Marker147
Marker148
Marker149
Marker150
Marker151
Marker152
Marker153
Marker154
Marker155
Marker156
Marker157
Marker158
Marker159
Marker160
90
60
30
0
0
1


Marker161
Marker162
Marker163
Marker164
Marker165
Marker166
Marker167
Marker168
Marker169
Marker170
Marker171
Marker172
Marker173
Marker174
Marker175
Marker176
Marker177
Marker178
Marker179
Marker180
Marker181
Marker182
Marker183
Marker184
Marker185
Marker186
Marker187
Marker188
Marker189
Marker190
Marker191
Marker192
90
60
30
0
0
1

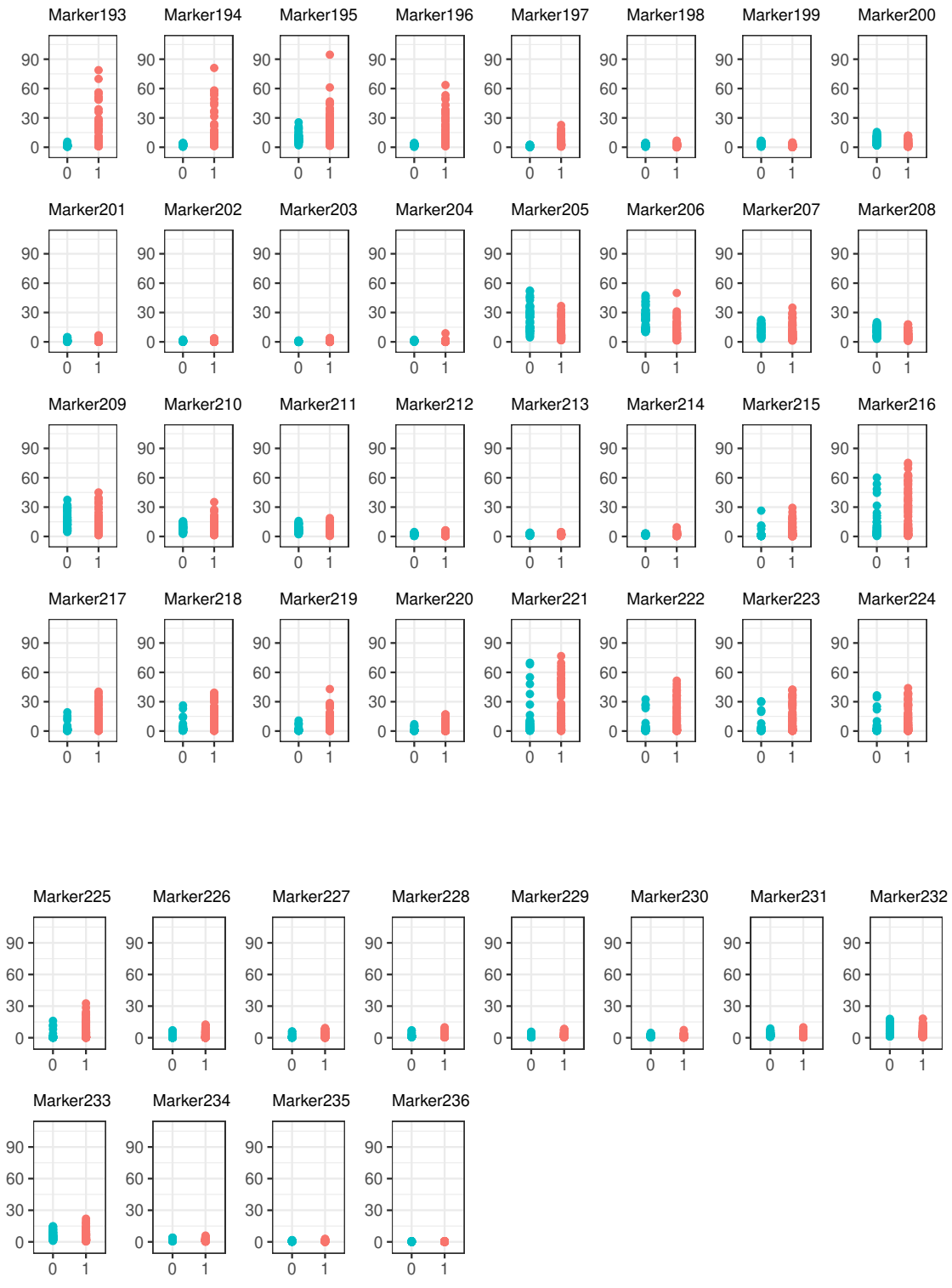


Fig. 5.1: Scatter Plot

Outcome values

In table 3.1 and 3.2 the true disease status is given, the score value and the probability of disease, from the logistic regression model, for each subject for Marker 197 and Marker 154.

Table 5.1: Marker 197

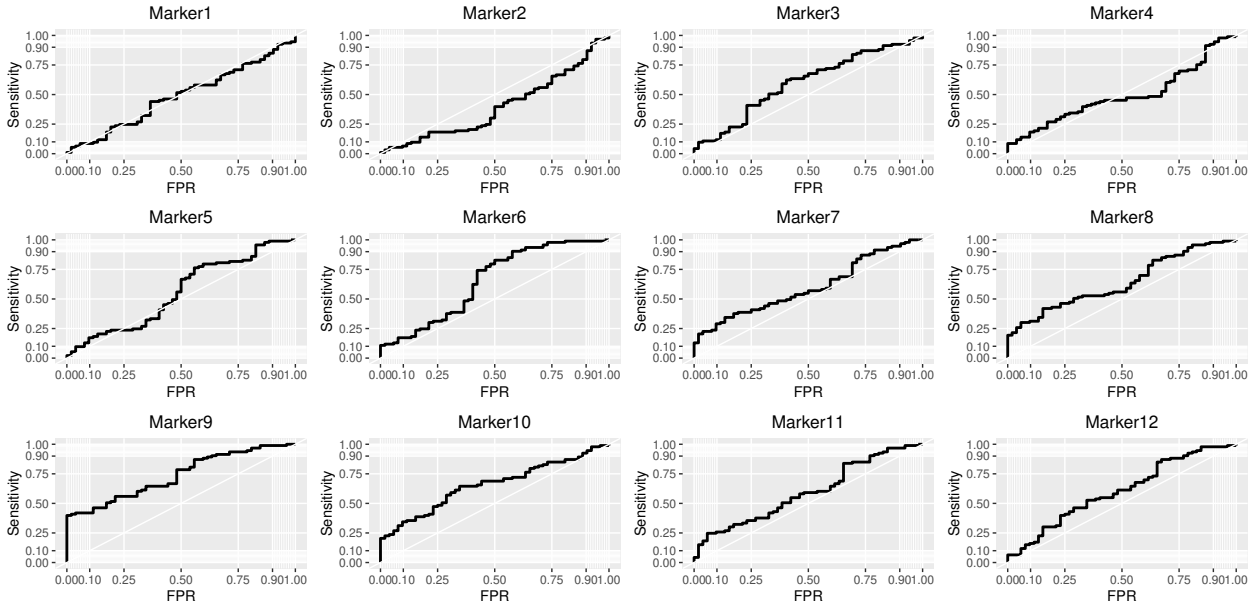
Subject	Status	Marker value	Score value	Probability	Subject	Status	Marker value	Score value	Probability
1	1	4.544	7.717	0.999	74	1	3.758	5.675	0.996
2	1	2.403	2.152	0.895	75	1	2.904	3.455	0.969
3	1	2.791	3.160	0.959	76	1	5.910	11.270	0.999
4	1	16.913	39.873	1.00	77	1	3.123	4.025	0.982
5	1	1.529	-0.119	0.470	78	1	2.442	2.253	0.904
6	1	2.060	1.260	0.779	79	1	1.956	0.989	0.729
7	1	0.665	-2.364	0.085	80	1	1.909	0.869	0.704
8	1	1.255	-0.831	0.303	81	1	2.169	1.543	0.823
9	1	4.131	6.645	0.998	82	1	2.060	1.261	0.779
10	1	4.685	8.085	0.999	83	1	2.689	2.895	0.947
11	1	2.086	1.32	20.790	84	1	1.919	0.894	0.709
12	1	18.686	44.483	1.00	85	1	3.348	4.608	0.990
13	1	2.598	2.658	0.934	86	1	3.890	6.018	0.997
14	1	9.497	20.595	1.00	87	1	2.435	2.236	0.903
15	1	1.057	-1.345	0.206	88	1	6.365	12.452	0.999
16	1	1.180	-1.027	0.263	89	1	5.632	10.546	0.999
17	1	1.707	0.343	0.584	90	1	7.289	14.854	0.999
18	1	2.382	2.098	0.890	91	1	6.425	12.608	0.999
19	1	2.910	3.471	0.969	92	1	3.073	3.895	0.980
20	1	7.349	15.011	0.999	93	1	3.678	5.466	0.995
21	1	1.782	0.537	0.631	94	0	1.095	-1.246	0.223
22	1	1.577	0.005	0.501	95	0	0.799	-2.016	0.117
23	1	7.056	14.249	0.999	96	0	0.880	-1.806	0.141
24	1	1.830	0.664	0.6660	97	0	1.349	-0.5877	0.357
25	1	2.100	1.364	0.796	98	0	0.799	-2.016	0.117
26	1	0.747	-2.15	0.104	99	0	0.890	-1.779	0.144
27	1	1.533	-0.109	0.472	100	0	0.937	-1.657	0.160
28	1	1.570	-0.012	0.496	101	0	1.121	-1.179	0.235
29	1	5.829	11.059	0.999	102	0	0.824	-1.953	0.124
30	1	11.161	24.921	1.000	103	0	0.701	-2.270	0.093
31	1	13.649	31.388	1.000	104	0	1.099	-1.236	0.225
32	1	5.547	10.326	0.999	105	0	1.717	0.369	0.591
33	1	2.184	1.582	0.829	106	0	0.805	-2.001	0.119
34	1	2.478	2.348	0.912	107	0	0.758	-2.124	0.106
35	1	1.875	0.780	0.685	108	0	1.676	0.261	0.565
36	1	2.169	1.543	0.823	109	0	0.403	-3.047	0.045
37	1	14.722	34.178	1.000	110	0	1.660	0.220	0.554
38	1	12.384	28.099	1.000	111	0	1.142	-1.125	0.245
39	1	22.781	55.093	1.000	112	0	0.983	-1.538	0.176
40	1	1.307	-0.696	0.332	113	0	0.300	-3.315	0.035
41	1	2.131	1.444	0.809	114	0	0.687	-2.307	0.090
42	1	13.519	31.050	1.00	115	0	0.943	-1.642	0.162
43	1	1.114	-1.198	0.231	116	0	-0.033	-4.181	0.015
44	1	1.453	-0.316	0.421	117	0	0.825	-1.949	0.124
45	1	3.410	4.770	0.991	118	0	0.828	-1.941	0.125
46	1	12.154	27.501	1.00	119	0	1.139	-1.132	0.243
47	1	8.242	17.331	0.999	120	0	1.110	-1.209	0.229
48	1	8.017	16.747	0.999	121	0	0.664	-2.369	0.085
49	1	5.487	10.170	0.999	122	0	0.621	-2.478	0.077
50	1	3.428	4.817	0.991	123	0	0.494	-2.803	0.056
51	1	12.594	28.645	1.000	124	0	1.726	0.393	0.597
52	1	4.756	8.270	0.999	125	0	0.810	-1.989	0.120
53	1	4.028	6.377	0.998	126	0	1.101	-1.230	0.226
54	1	8.636	18.357	0.9999	127	0	0.928	-1.680	0.157
55	1	3.031	3.785	0.977	128	0	2.237	1.721	0.848
56	1	2.534	2.494	0.923	129	0	0.805	-2.000	0.119
57	1	1.275	-0.780	0.314	130	0	1.401	-0.452	0.388
58	1	3.532	5.088	0.993	131	0	1.751	0.456	0.612
59	1	3.055	3.848	0.979	132	0	1.434	-0.365	0.409
60	1	1.911	0.873	0.705	133	0	2.618	2.712	0.937
61	1	2.738	3.023	0.953	134	0	2.271	1.808	0.859
62	1	6.975	14.038	0.999	135	0	0.911	-1.726	0.151
63	1	2.664	2.831	0.944	136	0	1.663	0.227	0.556
64	1	2.634	2.753	0.940	137	0	1.998	1.100	0.750
65	1	3.670	5.446	0.995	138	0	0.816	-1.974	0.121
66	1	2.289	1.856	0.864	139	0	1.237	-0.877	0.293
67	1	4.244	6.937	0.999	140	0	1.004	-1.483	0.184
68	1	2.682	2.878	0.946	141	0	1.023	-1.434	0.192
69	1	2.907	3.463	0.969	142	0	1.970	1.028	0.736
70	1	3.043	3.815	0.978	143	0	0.410	-3.027	0.046
71	1	2 .030	1.181	0.765	144	0	0.851	-1.881	0.132
72	1	2.193	1.607	0.833	145	0	1.803	0.594	0.644
73	1	6.308	12.303	0.999	.	.	.	.	.

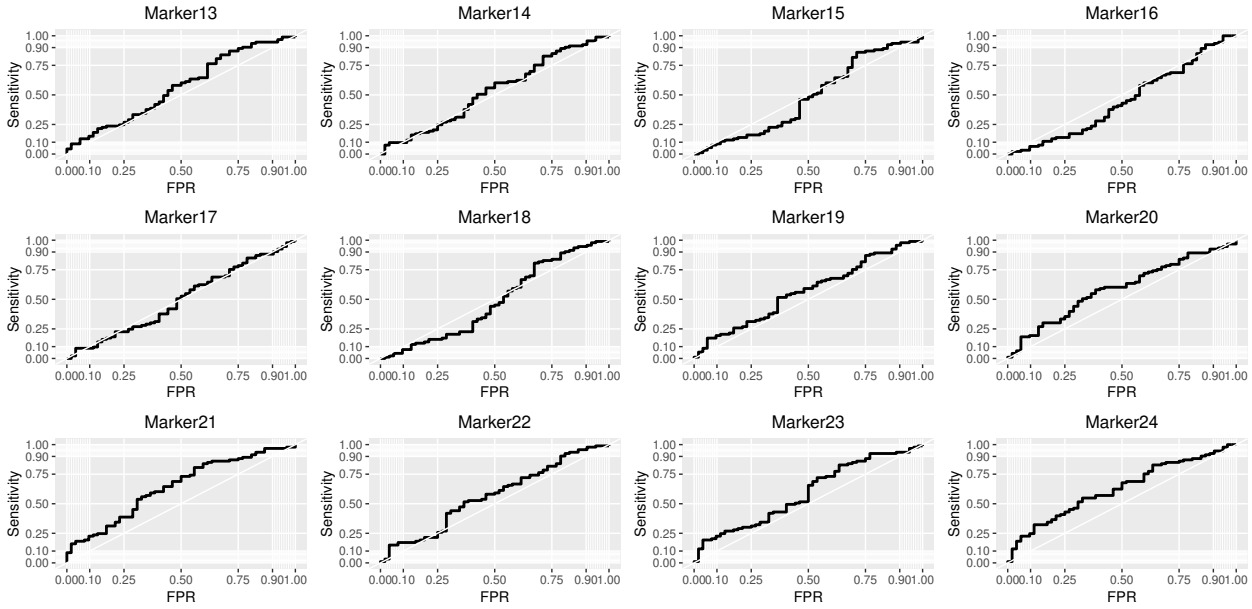
Table 5.2: Marker 154

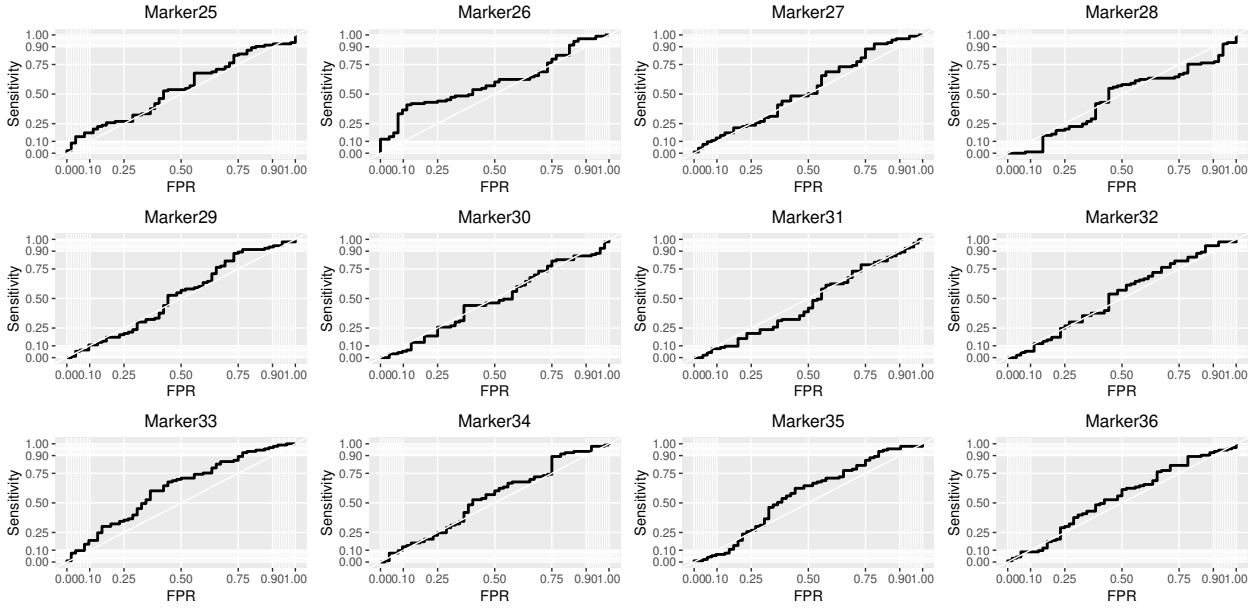
Subject	Status	Marker value	Score value	Probability	Subject	Status	Marker value	Score value	Probability
1	1	7.154	2.033	0.884	74	1	30.218	-0.888	0.291
2	1	7.211	2.025	0.883	75	1	3.801	2.457	0.921
3	1	13.404	1.241	0.775	76	1	11.637	1.465	0.812
4	1	17.882	0.674	0.662	77	1	13.108	1.278	0.782
5	1	13.207	1.266	0.780	78	1	19.046	0.526	0.628
6	1	16.158	0.892	0.709	79	1	12.701	1.330	0.790
7	1	11.058	1.538	0.823	80	1	9.744	1.704	0.846
8	1	5.266	2.272	0.906	81	1	11.694	1.457	0.811
9	1	22.677	0.066	0.516	82	1	18.018	0.656	0.658
10	1	28.885	-0.719	0.327	83	1	15.483	0.977	0.726
11	1	17.985	0.660	0.659	84	1	13.626	1.213	0.770
12	1	25.383	-0.276	0.431	85	1	32.707	-1.203	0.230
13	1	19.385	0.483	0.618	86	1	7.057	2.045	0.885
14	1	19.121	0.517	0.626	87	1	3.724	2.467	0.921
15	1	18.759	0.562	0.637	88	1	5.621	2.227	0.902
16	1	18.204	0.633	0.653	89	1	8.488	1.863	0.865
17	1	6.768	2.081	0.889	90	1	4.965	2.310	0.909
18	1	30.189	-0.884	0.292	91	1	6.636	2.098	0.890
19	1	25.318	-0.267	0.433	92	1	17.288	0.749	0.679
20	1	22.509	0.087	0.521	93	1	5.240	2.275	0.906
21	1	11.567	1.473	0.813	94	0	17.081	0.775	0.684
22	1	12.322	1.378	0.798	95	0	15.536	0.971	0.725
23	1	15.618	0.960	0.723	96	0	22.222	0.124	0.531
24	1	7.220	2.024	0.883	97	0	21.491	0.216	0.553
25	1	30.415	-0.913	0.286	98	0	23.987	-0.099	0.475
26	1	25.194	-0.252	0.437	99	0	16.582	0.838	0.698
27	1	16.384	0.863	0.703	100	0	10.159	1.652	0.839
28	1	12.560	1.348	0.793	101	0	24.743	-0.195	0.451
29	1	26.097	-0.366	0.409	102	0	10.946	1.552	0.825
30	1	19.514	0.467	0.614	103	0	12.563	1.347	0.793
31	1	16.291	0.875	0.705	104	0	17.861	0.676	0.663
32	1	10.090	1.661	0.840	105	0	15.850	0.931	0.717
33	1	13.392	1.242	0.776	106	0	24.775	-0.199	0.450
34	1	12.272	1.384	0.799	107	0	19.071	0.523	0.627
35	1	23.907	-0.089	0.477	108	0	23.508	-0.038	0.490
36	1	10.233	1.643	0.837	109	0	32.469	-1.173	0.236
37	1	11.515	1.480	0.814	110	0	32.013	-1.116	0.246
38	1	7.774	1.954	0.875	111	0	29.549	-0.803	0.309
39	1	7.605	1.975	0.878	112	0	27.059	-0.488	0.380
40	1	8.539	1.857	0.865	113	0	23.969	-0.097	0.475
41	1	18.006	0.532	0.630	114	0	27.122	-0.496	0.378
42	1	23.380	-0.022	0.494	115	0	30.508	-0.925	0.283
43	1	11.581	1.472	0.813	116	0	7.821	1.948	0.875
44	1	12.636	1.338	0.792	117	0	33.442	-1.297	0.214
45	1	29.808	-0.836	0.302	118	0	34.190	-1.391	0.199
46	1	10.765	1.575	0.828	119	0	16.411	0.860	0.702
47	1	5.936	2.187	0.899	120	0	31.039	-0.992	0.270
48	1	6.062	2.171	0.897	121	0	27.745	-0.575	0.360
49	1	16.059	0.904	0.711	122	0	8.469	1.866	0.866
50	1	16.209	0.886	0.708	123	0	26.156	-0.374	0.407
51	1	6.491	2.116	0.892	124	0	40.294	-2.164	0.102
52	1	7.649	1.970	0.877	125	0	34.222	-1.395	0.198
53	1	30.525	-0.927	0.283	126	0	34.925	-1.484	0.184
54	1	7.384	2.003	0.881	127	0	24.177	-0.123	0.469
55	1	23.826	-0.078	0.480	128	0	20.207	0.379	0.593
56	1	20.392	0.356	0.588	129	0	26.089	-0.365	0.409
57	1	14.459	1.107	0.751	130	0	18.918	0.542	0.632
58	1	34.863	-1.477	0.185	131	0	20.733	0.312	0.577
59	1	6.256	2.146	0.895	132	0	21.832	0.173	0.543
60	1	12.440	1.363	0.796	133	0	23.985	-0.099	0.475
61	1	17.166	0.764	0.682	134	0	41.866	-2.364	0.085
62	1	6.318	2.138	0.894	135	0	27.412	-0.533	0.369
63	1	3.714	2.468	0.921	136	0	12.776	1.320	0.789
64	1	7.404	2.001	0.880	137	0	26.230	-0.383	0.405
65	1	11.864	1.436	0.807	138	0	10.182	1.649	0.838
66	1	8.997	1.799	0.858	139	0	22.593	0.077	0.519
67	1	10.446	1.615	0.834	140	0	37.308	-1.786	0.143
68	1	8.689	1.838	0.862	141	0	28.096	-0.618	0.349
69	1	11.201	1.520	0.820	142	0	22.518	0.086	0.521
70	1	8.486	1.864	0.865	143	0	18.603	0.582	0.641
71	1	25.631	-0.307	0.423	144	0	17.016	0.783	0.686
72	1	7.628	1.972	0.877	145	0	11.482	1.484	0.815
73	1	8.168	1.904	0.870	.	.	.	.	.

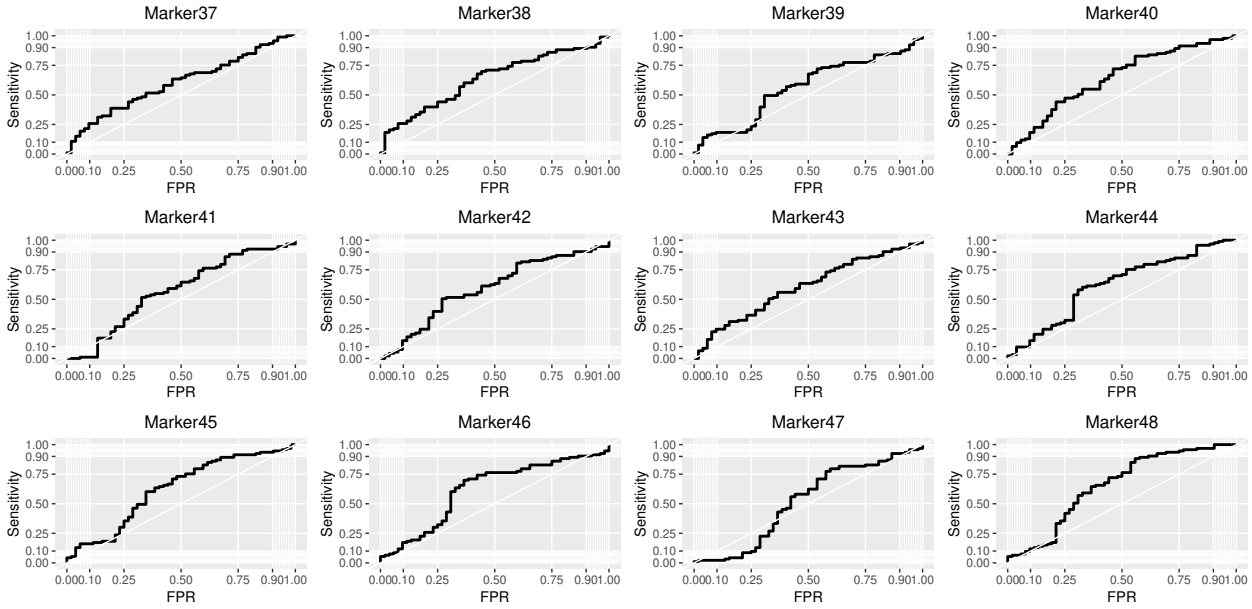
ROC curves for each Marker

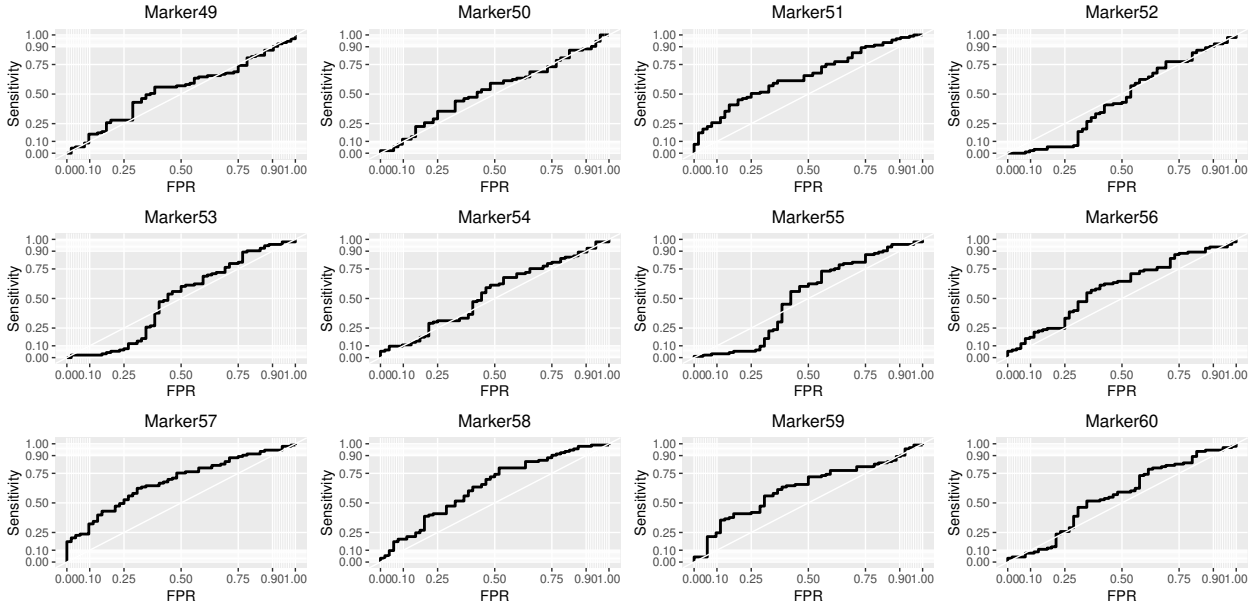
On the following graphs, we used on every Marker a logistic regression model, in which we tried to predict if a subject is diseased or not diseased, considering only one Marker.

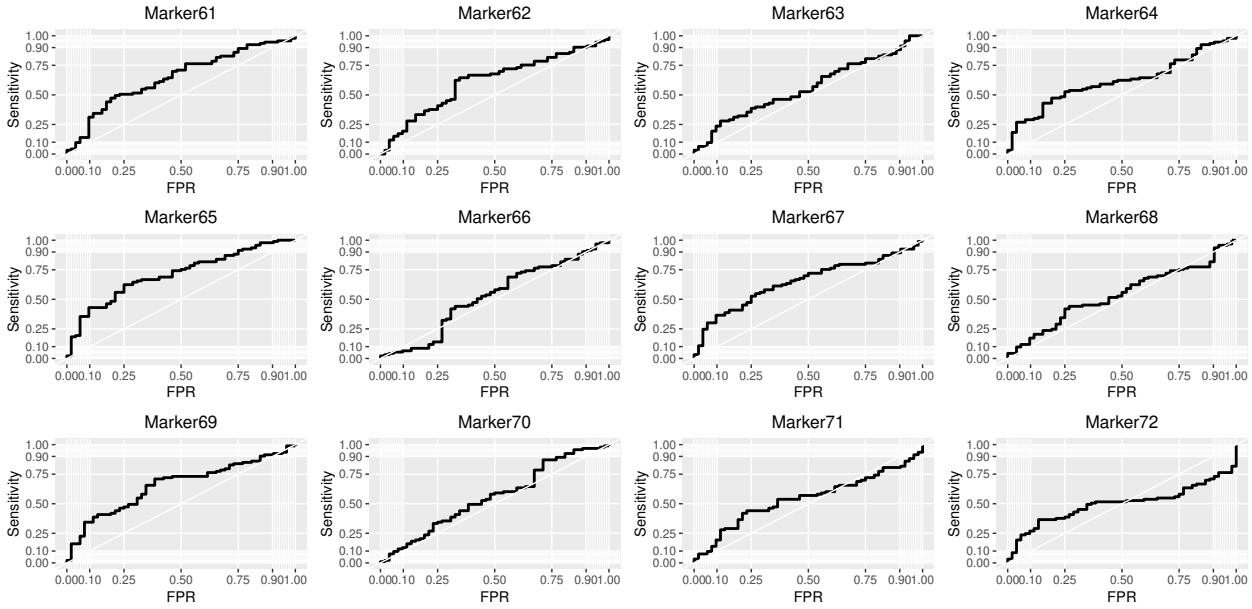


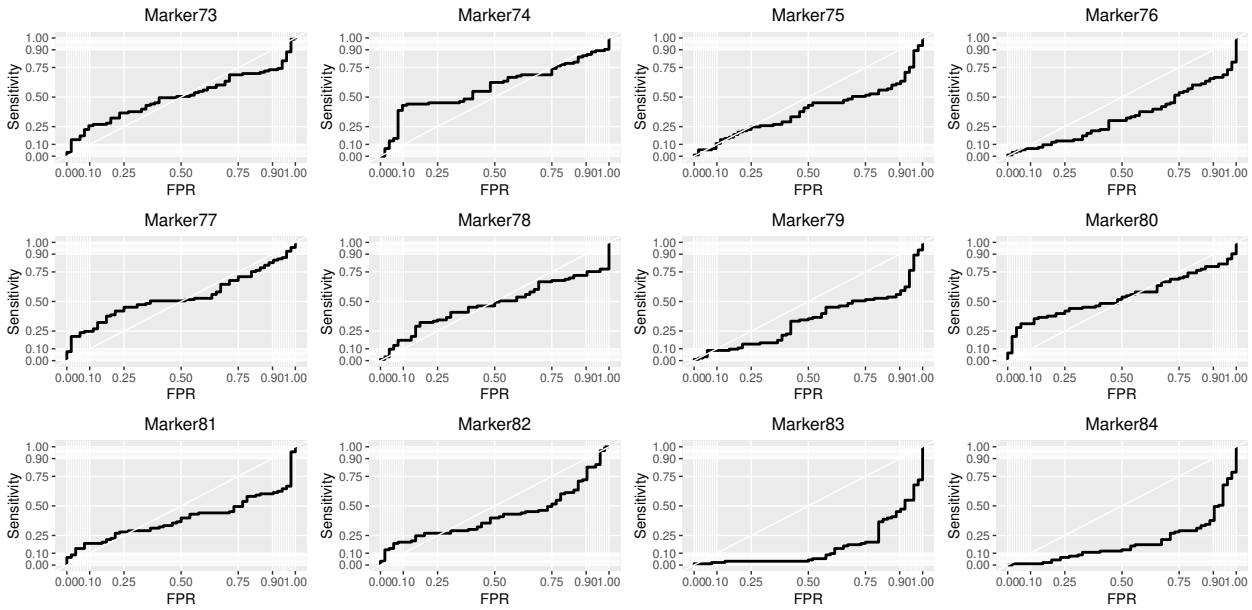

Marker13
Marker14
Marker15
Marker16
Marker17
Marker18
Marker19
Marker20
Marker21
Marker22
Marker23
Marker24
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

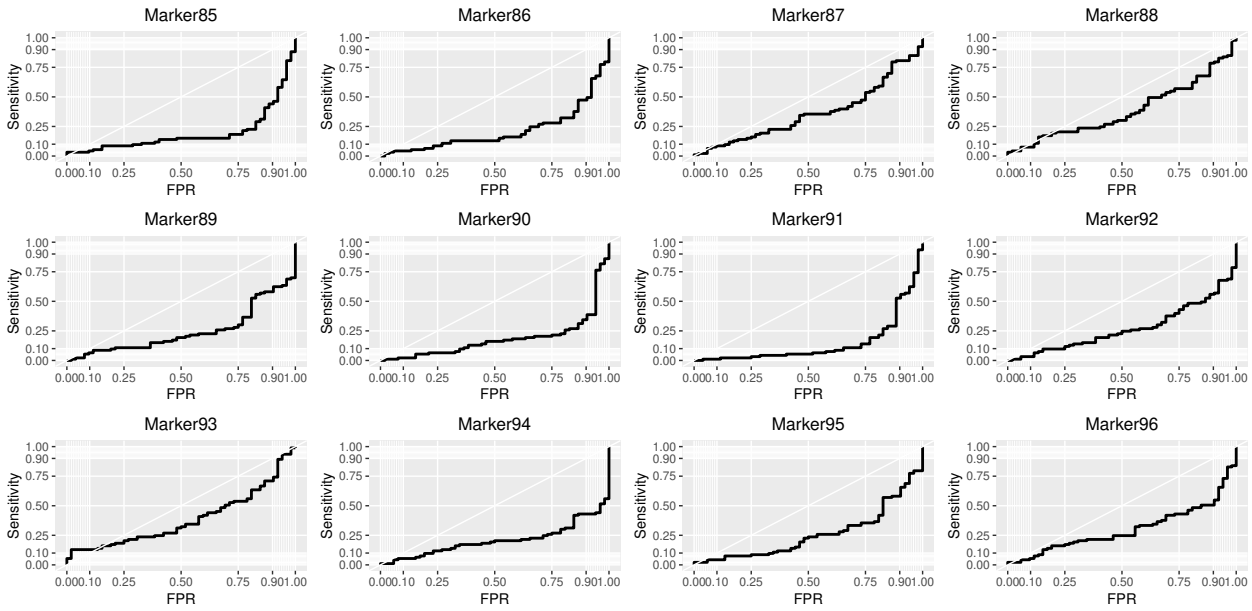

Marker25
Marker26
Marker27
Marker28
Marker29
Marker30
Marker31
Marker32
Marker33
Marker34
Marker35
Marker36
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

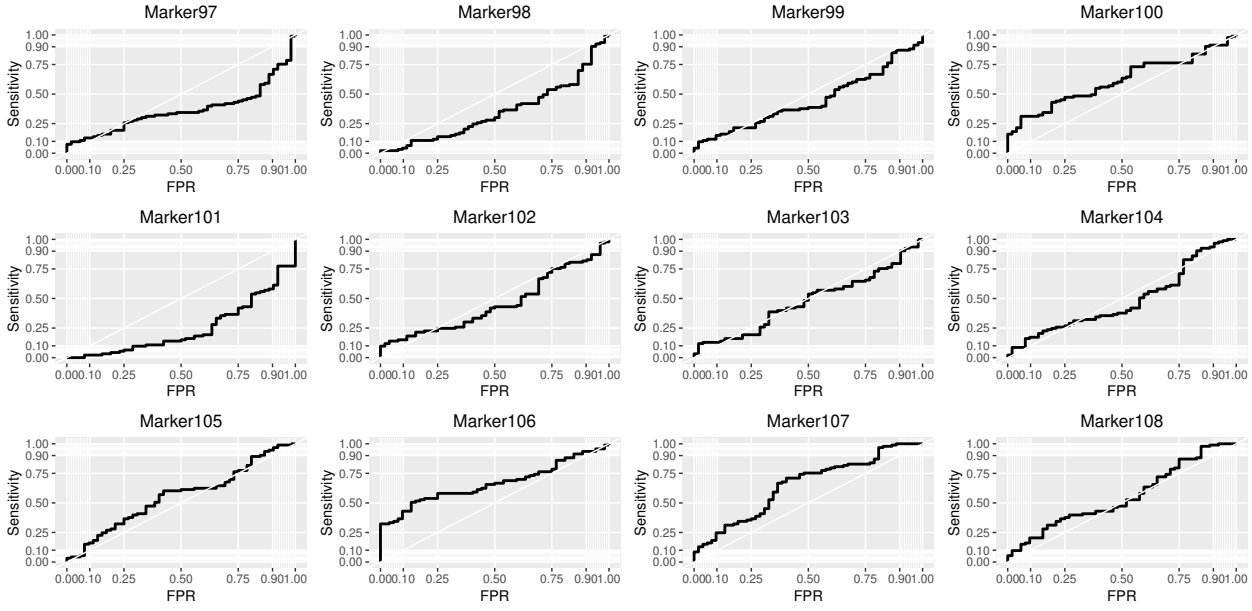

Marker37
Marker38
Marker39
Marker40
Marker41
Marker42
Marker43
Marker44
Marker45
Marker46
Marker47
Marker48
Sensitivity
FPR

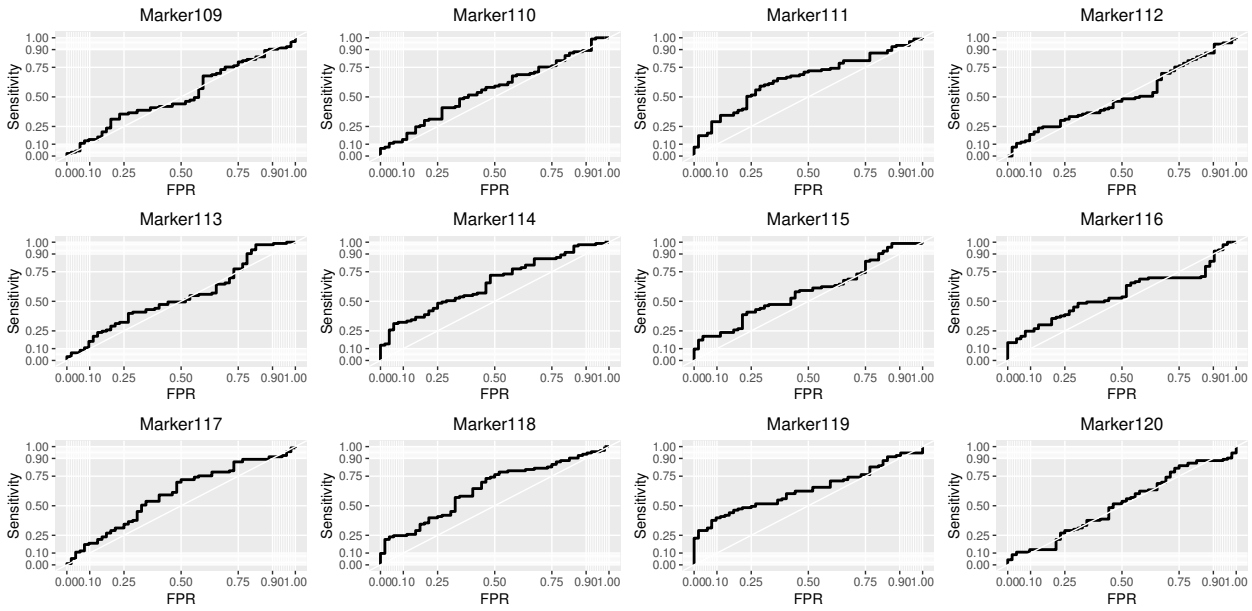

Marker49
Marker50
Marker51
Marker52
Marker53
Marker54
Marker55
Marker56
Marker57
Marker58
Marker59
Marker60
Sensitivity
FPR

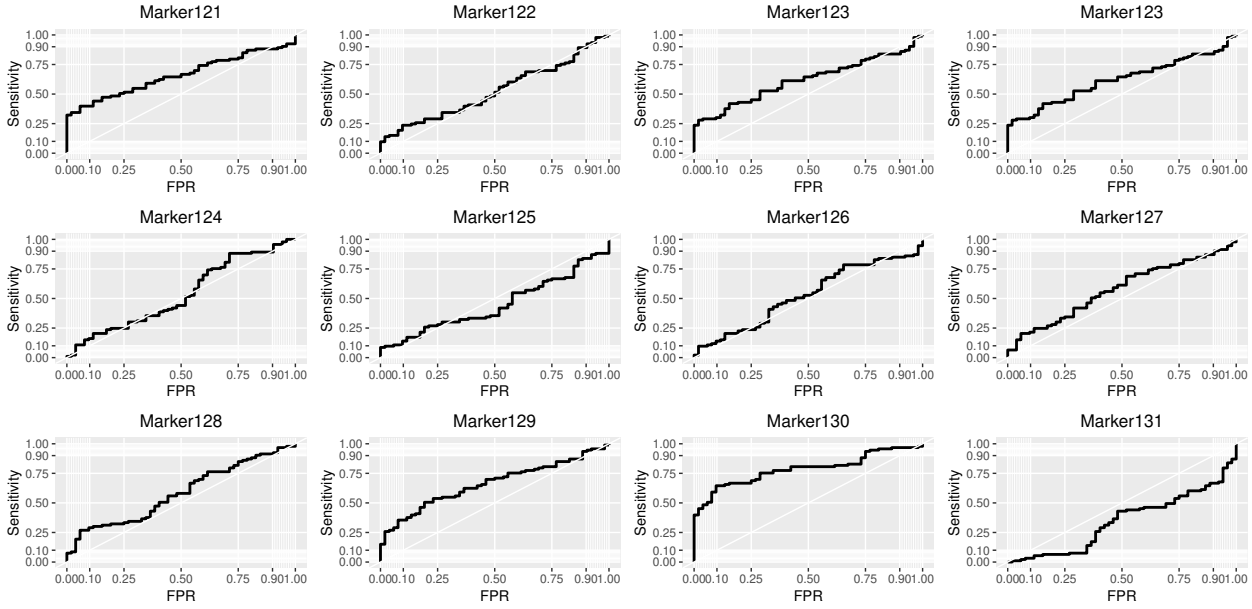

Marker61
Marker62
Marker63
Marker64
Marker65
Marker66
Marker67
Marker68
Marker69
Marker70
Marker71
Marker72
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

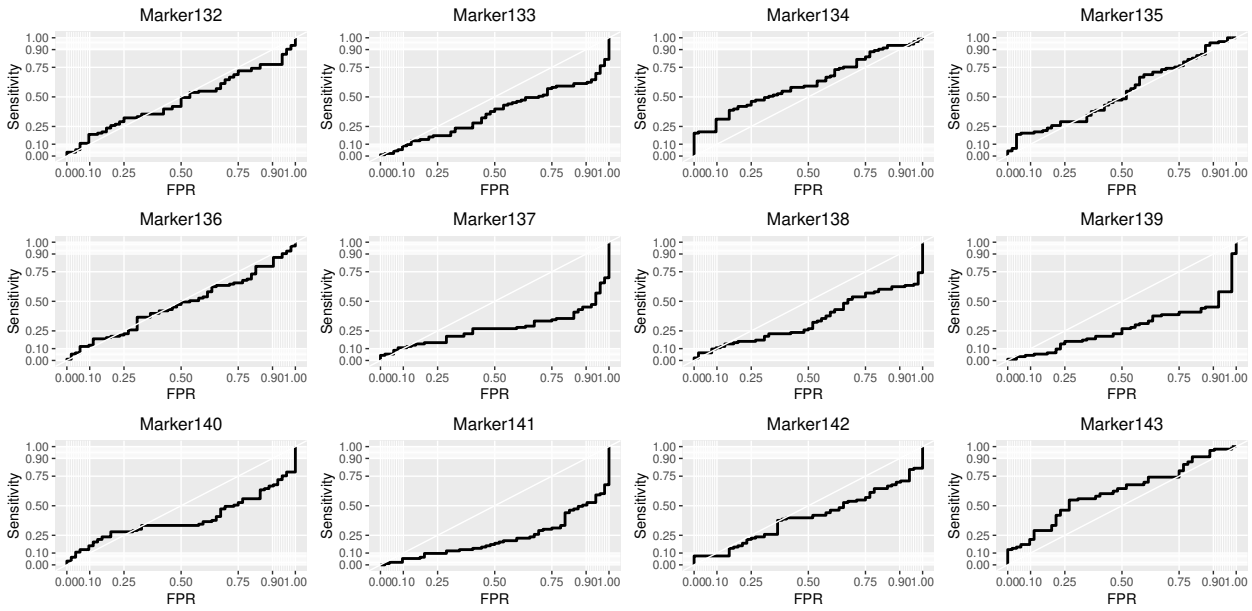

Marker73
Marker74
Marker75
Marker76
Marker77
Marker78
Marker79
Marker80
Marker81
Marker82
Marker83
Marker84
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

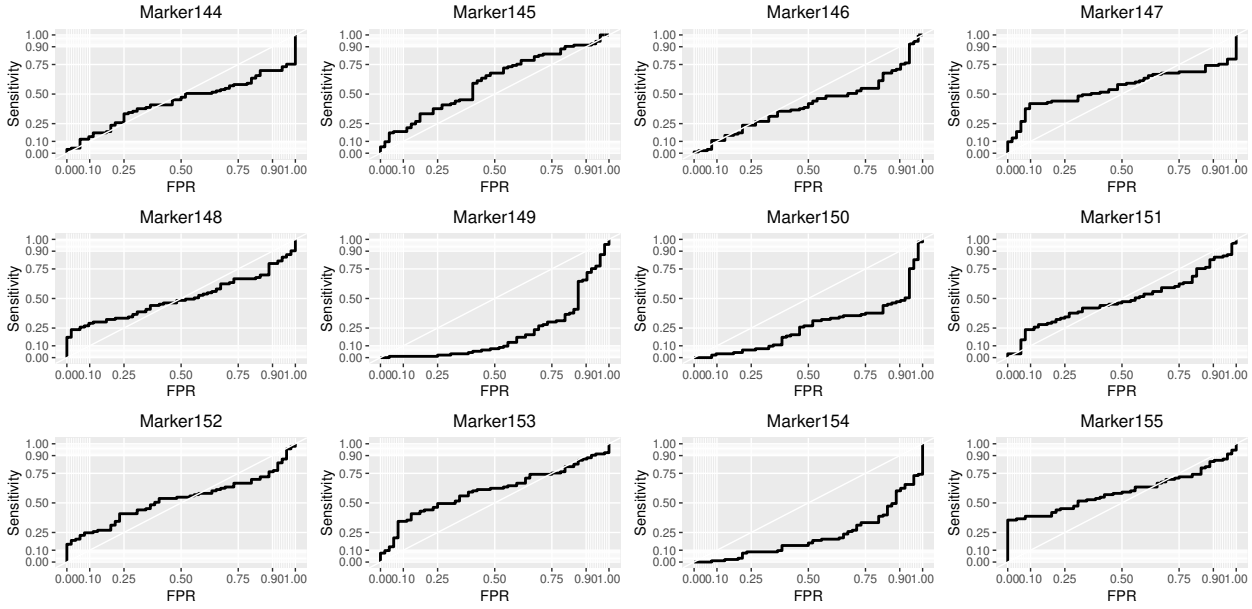

Marker85
Marker86
Marker87
Marker88
Marker89
Marker90
Marker91
Marker92
Marker93
Marker94
Marker95
Marker96
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

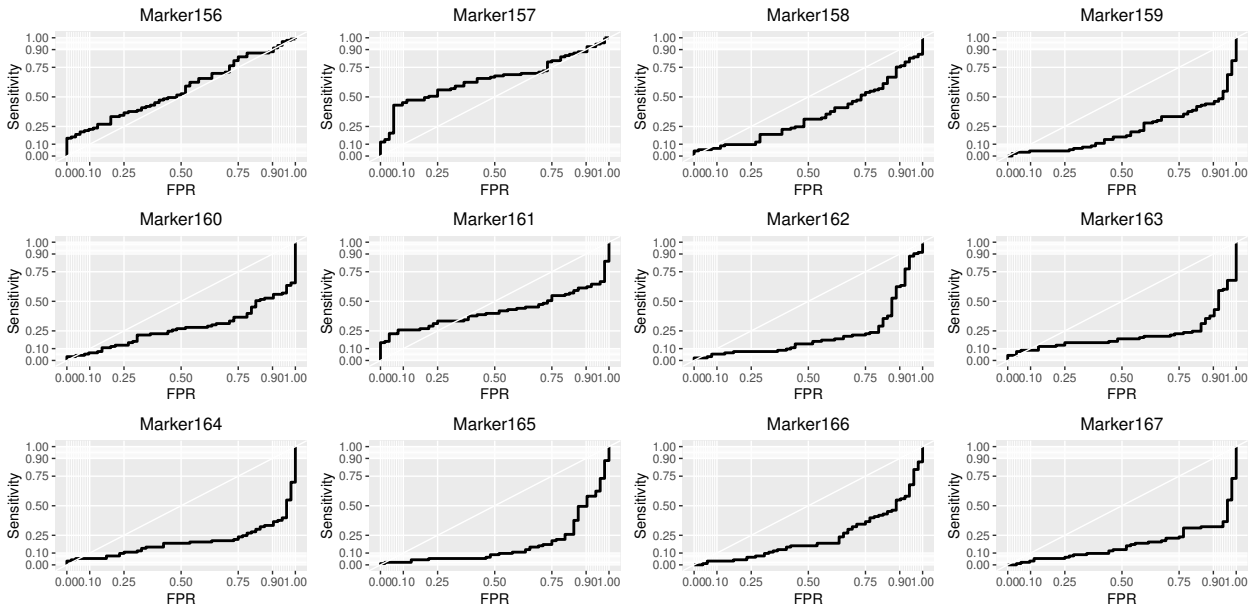

Marker97
Marker98
Marker99
Marker100
Marker101
Marker102
Marker103
Marker104
Marker105
Marker106
Marker107
Marker108
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

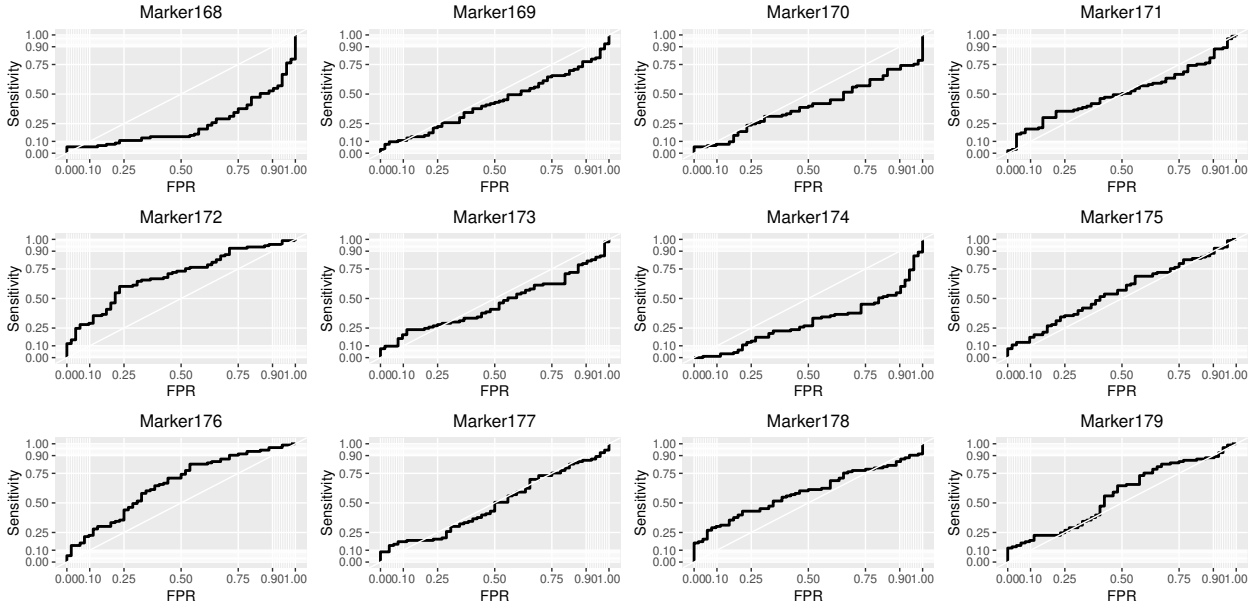

Marker109
Marker110
Marker111
Marker112
Marker113
Marker114
Marker115
Marker116
Marker117
Marker118
Marker119
Marker120
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

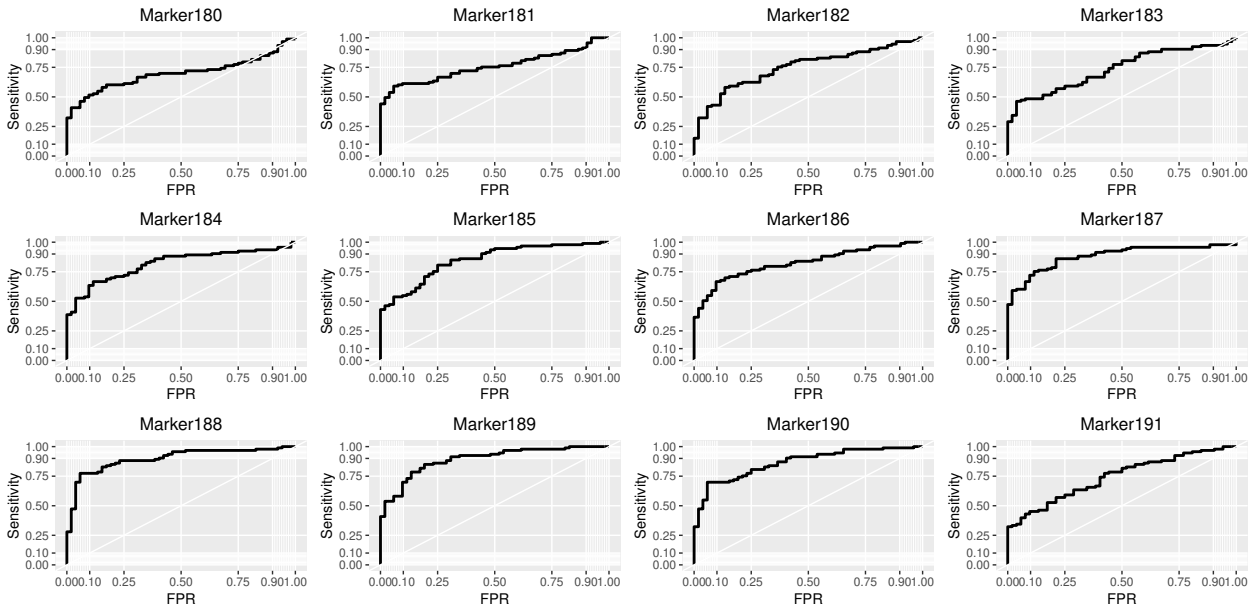

Marker121
Marker122
Marker123
Marker123
Marker124
Marker125
Marker126
Marker127
Marker128
Marker129
Marker130
Marker131
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

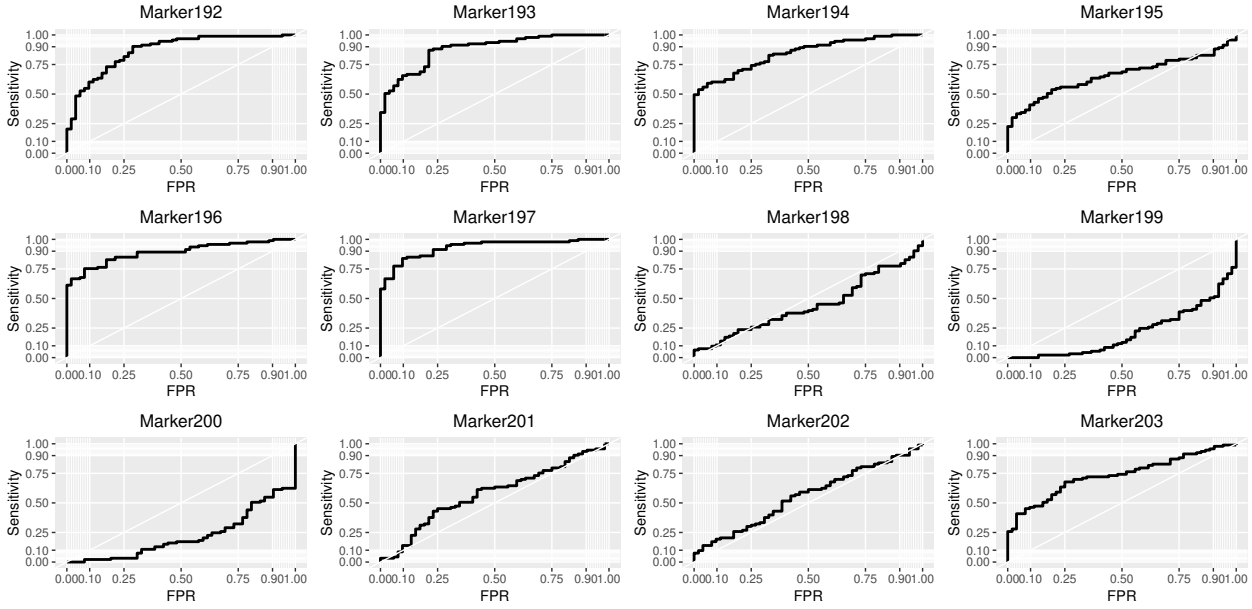

Marker132
Marker133
Marker134
Marker135
Marker136
Marker137
Marker138
Marker139
Marker140
Marker141
Marker142
Marker143
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

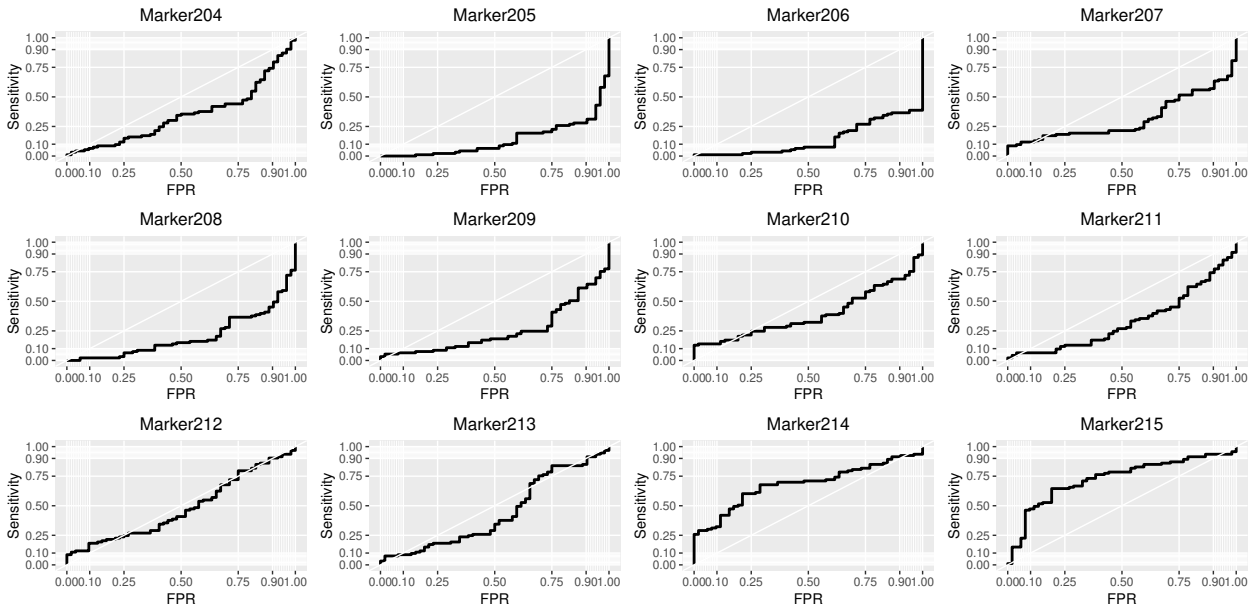

Marker144
Marker145
Marker146
Marker147
Marker148
Marker149
Marker150
Marker151
Marker152
Marker153
Marker154
Marker155
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

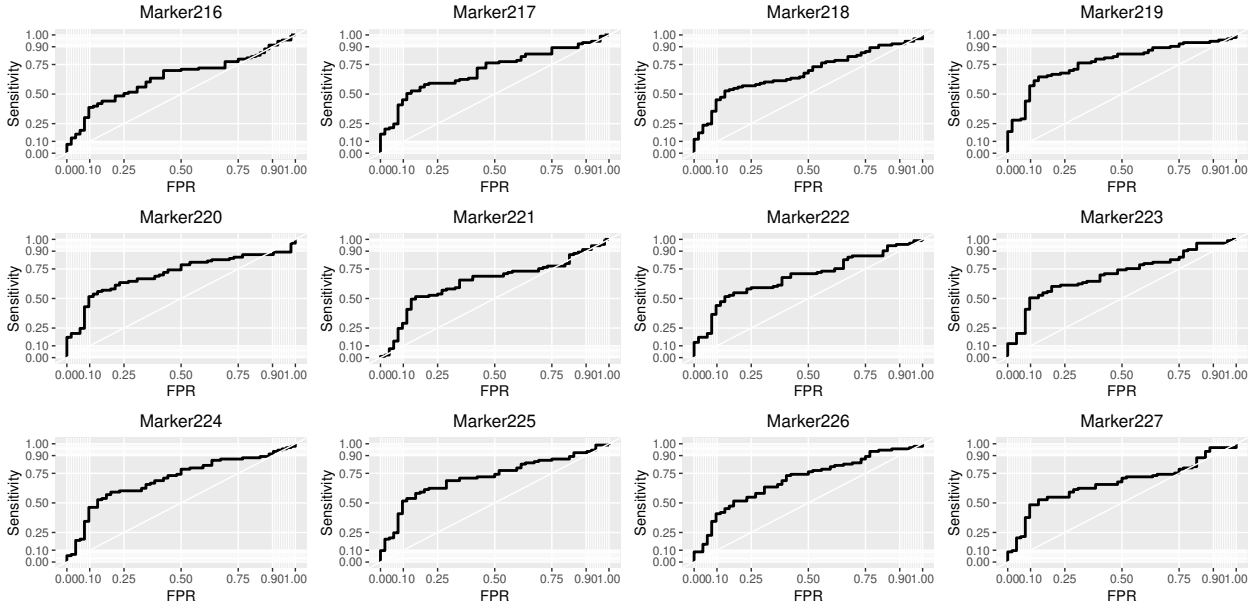

Marker156
Marker157
Marker158
Marker159
Marker160
Marker161
Marker162
Marker163
Marker164
Marker165
Marker166
Marker167
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00

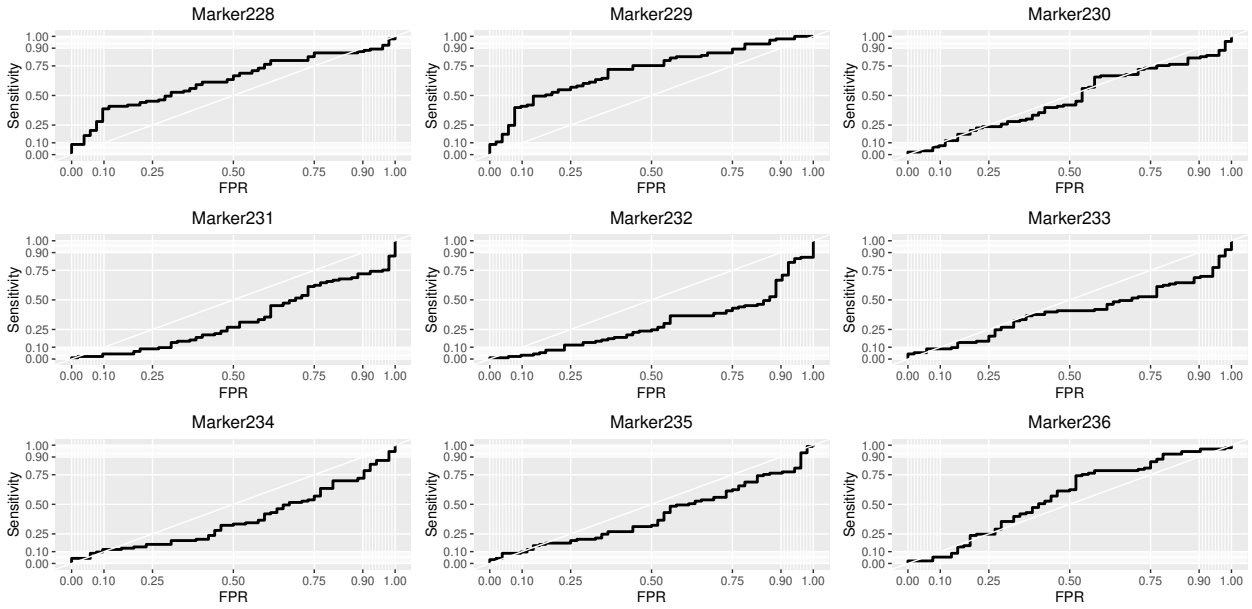

Marker168
Marker169
Marker170
Marker171
Marker172
Marker173
Marker174
Marker175
Marker176
Marker177
Marker178
Marker179
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00


Marker180
Marker181
Marker182
Marker183
Marker184
Marker185
Marker186
Marker187
Marker188
Marker189
Marker190
Marker191
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00


Marker192
Marker193
Marker194
Marker195
Marker196
Marker197
Marker198
Marker199
Marker200
Marker201
Marker202
Marker203
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00


Marker204
Marker205
Marker206
Marker207
Marker208
Marker209
Marker210
Marker211
Marker212
Marker213
Marker214
Marker215
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00


Marker216
Marker217
Marker218
Marker219
Marker220
Marker221
Marker222
Marker223
Marker224
Marker225
Marker226
Marker227
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00


Marker228
Marker229
Marker230
Marker231
Marker232
Marker233
Marker234
Marker235
Marker236
Sensitivity
FPR
1.00
0.90
0.75
0.50
0.25
0.10
0.00
0.00 0.10 0.25 0.50 0.75 0.90 1.00