Machine Learning, Spring 2021
Homework 4

Daniil Bulat, Jonas Josef Huwyler, Haochen Li,Giovanni Magagnin

3/25/2021

Exercise 1

As usual we assume that our separating line is given by the equation

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

or equivalently

$$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2} x_1$$

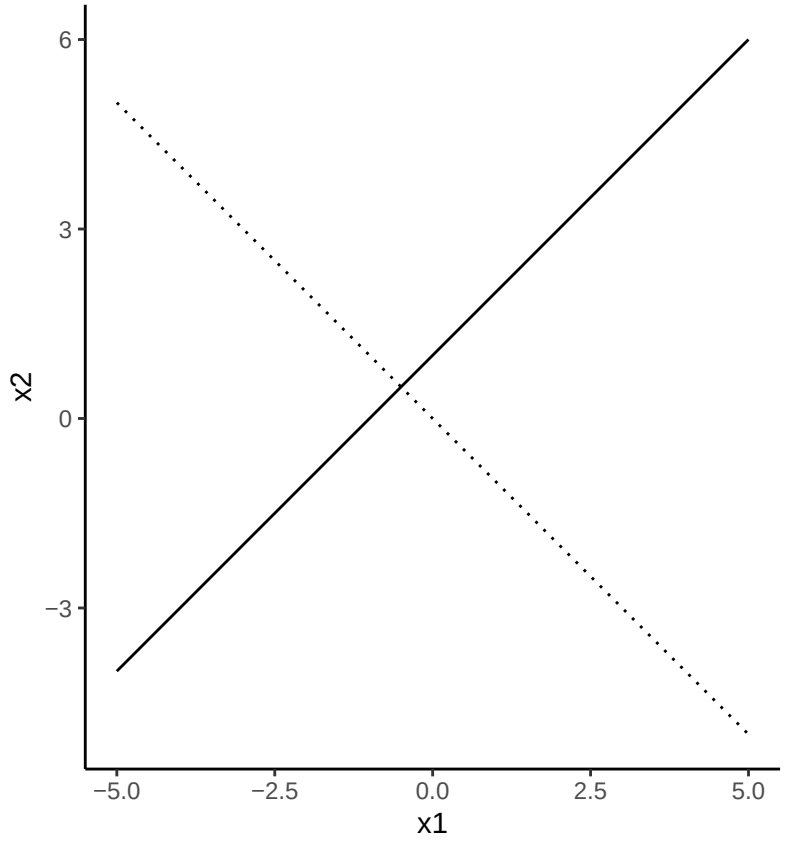
Assume that $w_0, w_1 > 0$ and $w_2 < 0$. The vector (w_1,w_2) is always perpendicular to the separating line, it is called the normal vector of the line.

```
# Initial Setting ------------------------------------------------------------

## clear the workplace
rm(list = ls())

## load the necessary package
library(ggplot2)

## Choose some suitable values
w0 <- 1
w1 <- 1
w2 <- -1

## Prepare data
x1 <- seq(-5,5,0.1)
x2 <- rep(NA,101)
x2 <- -w0/w2 - w1/w2*x1
nv1 <- x1  # stands for normal vector
nv2 <- w2/w1*nv1
data <- data.frame(x1,x2,nv1,nv2)
```

Question 1

Draw an exemplary graph for this setting (by choosing some suitable values for w_0, w_1, w_2). In the same plot, draw the vector (w_1, w_2). If it does not intersect the separating line, extend the vector to a line.

```
# Question 1 ------------------------------------------------------------

ggplot(data,aes(x=x1)) +
  geom_line(aes(y=x2)) +
  geom_line(aes(y=nv2),lty ="dotted")+
  coord_fixed() +
  theme(legend.position="none")+
  theme_classic()
```



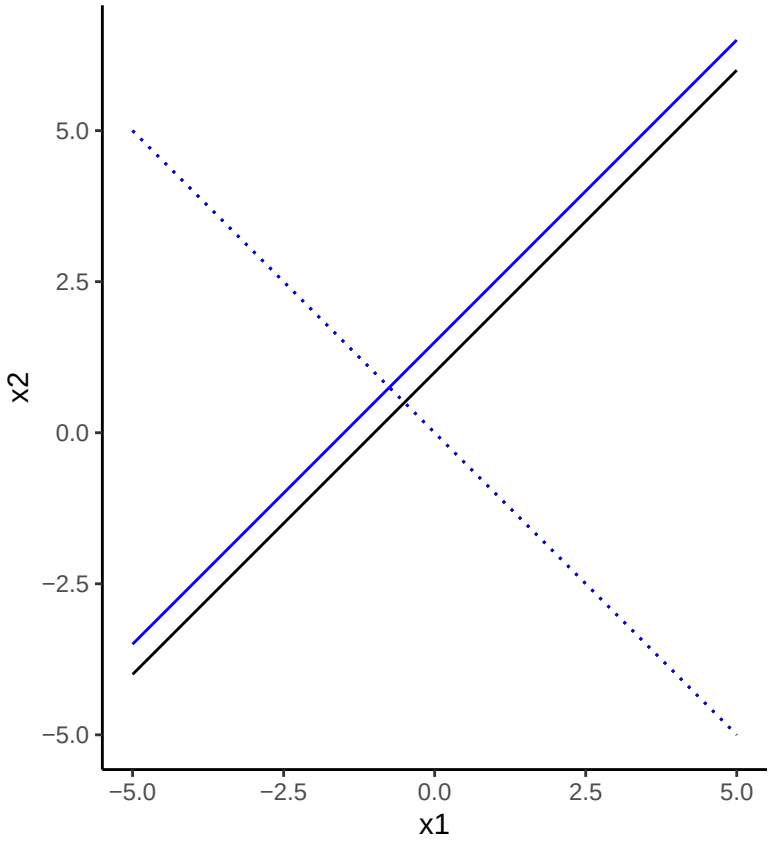
The exemplary graph is shown above by choosing $w_0 = 1, w_1 = 1, w_2 = -1$. Though these paramaters chosen make the graph look good, the codes can generate similar results under other setting.

In the graph, the solid line is separating line generated by the function $x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2}x_1$. The dotted line is the extended line with regard to the corresponding normal vector. Notice the line is always perpendicular to the separating line and so it has the slope as $\frac{w_2}{w_1}$.

Question 2

What happens to the line if we increase w_0 and let w_1, w_2 unchanged?

```
# question 2 ------------------------------------------------------------

w0.ex2 <- w0 + 0.5
data$x2.ex2 <- -w0.ex2/w2 - w1/w2*data$x1
data$nv2.ex2 <- w2/w1*data$nv1

ggplot(data,aes(x=x1)) +
  geom_line(aes(y=x2)) +
  geom_line(aes(y=nv2),lty ="dotted")+
  geom_line(aes(y=x2.ex2),color = "blue") +
  geom_line(aes(y=nv2.ex2),color = "blue",lty ="dotted")+
  coord_fixed()+
  theme(legend.position="none")+
  theme_classic()
```

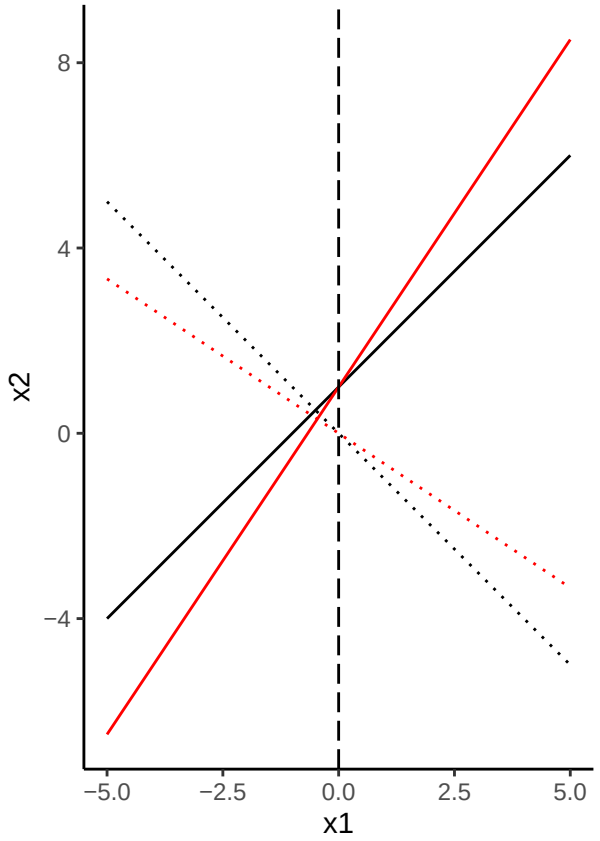


The black lines are under the setting before changing and the blue lines are under setting after changing. If we increase w_0 and hold other parameters unchanged, the separating line moves toward left and the normal vector is unchanged. When w_0 increases, only the intercept of the separating line decreases and the normal vector is still $(w_1, w_2) = (1, -1)$.

Question 3

What happens to the line if we increase w_1 and let w_0, w_2 unchanged?

```
# question 3 ------------------------------------------------------------------

w1.ex3 <- w1 + 0.5
data$x2.ex3 <- -w0/w2 - w1.ex3/w2*data$x1
data$nv2.ex3 <- w2/w1.ex3*data$nv1

ggplot(data,aes(x=x1)) +
  geom_line(aes(y=x2)) +
  geom_line(aes(y=nv2),lty ="dotted")+
  geom_line(aes(y=x2.ex3),color = "red") +
  geom_line(aes(y=nv2.ex3),color = "red",lty ="dotted")+
  coord_fixed()+
  theme(legend.position="none")+
  theme_classic()+
  geom_vline(xintercept = 0,lty = "longdash")
```

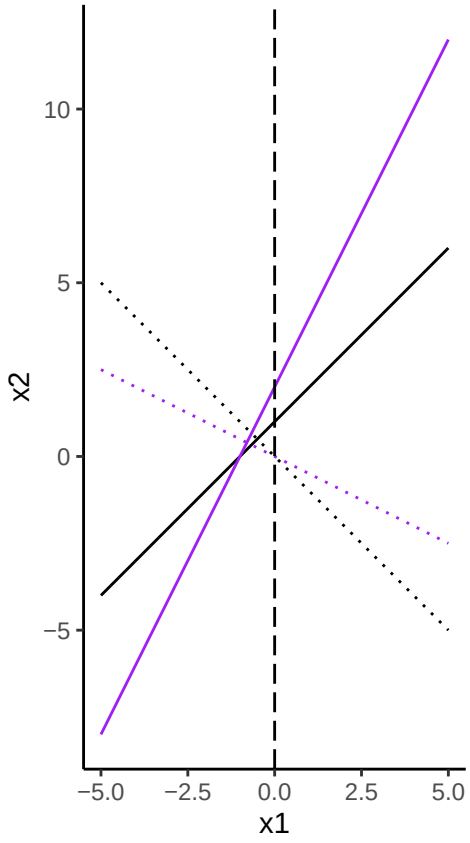


The black lines are under the setting before changing and the red lines are under setting after changing. There is a reference line of $x_1 = 0$ in the graph. If we increase w_1 and hold other parameters unchanged, the separating line rotates counter-clockwise and the normal vector also rotates to be perpendicular to the separating line. When w_1 increases, the intercept of the separating line doesn't change but the slope increases. The normal vector changes to $(w_1, w_2) = (1.5, -1)$.

Question 4

What happens to the line if we increase w_2 and let w_1, w_0 unchanged?

```
# question 4 ------------------------------------------------------------

w2.ex4 <- w2 + 0.5
data$x2.ex4 <- -w0/w2.ex4 - w1/w2.ex4*data$x1
data$nv2.ex4 <- w2.ex4/w1*data$nv1

ggplot(data,aes(x=x1)) +
  geom_line(aes(y=x2)) +
  geom_line(aes(y=nv2),lty ="dotted")+
  geom_line(aes(y=x2.ex4),color = "purple") +
  geom_line(aes(y=nv2.ex4),color = "purple",lty ="dotted")+
  coord_fixed()+
  theme(legend.position="none")+
  theme_classic() +
  geom_vline(xintercept = 0,lty = "longdash")
```



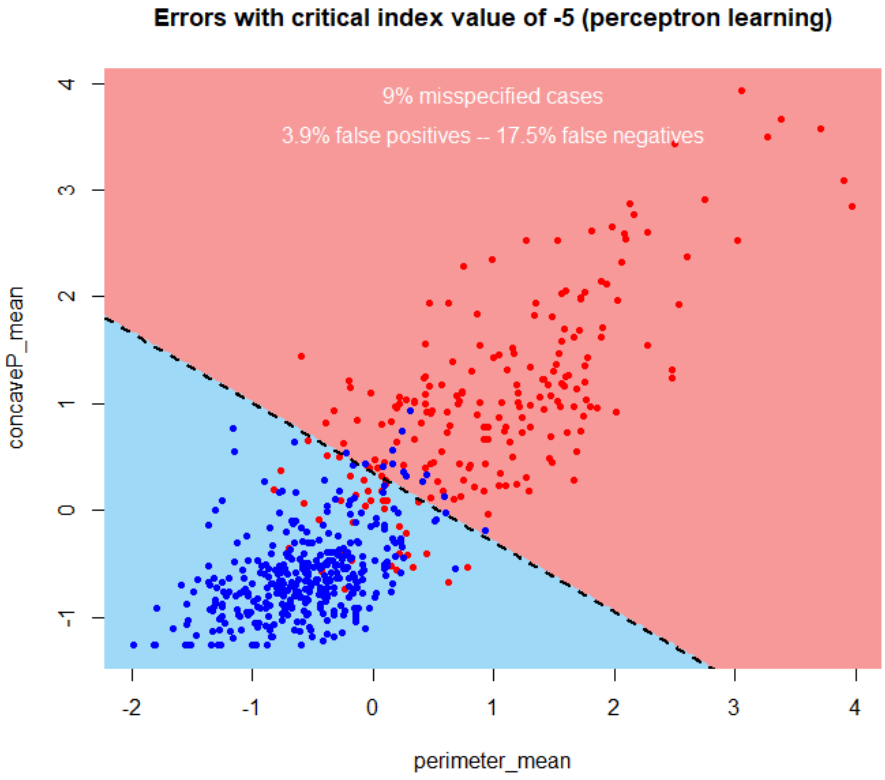
The black lines are under the setting before changing and the purple lines are under setting after changing. There is a reference line of $x_1 = 0$ in the graph. If we increase w_2 and hold other parameters unchanged, the separating line rotates counter-clockwise and moves toward left. When w_2 increases, both intercept and slope increases, since the absolute value of w_2 decreases.The normal vector also rotates to be perpendicular to the separating line and changes to $(w_1, w_2) = (1, -0.5)$.

Exercise 2

Look at the algorithm *PerceptronError.R* from StudyNet. Try to understand what each section in the script before the #Plotting section does. In particular, make sure you understand the definition and calculation of false positive and false negatives.

Question 1

What is a false positive error in this setting (content-wise)? Which numerical value does it take on in the algorithm.



A false positive in this setting is a patient to whom it gets diagnosed a malign breast cancer tumor, although it is benign instead. In this algorithm,with critical index value of -5,the percentage of false positives in terms of negatives is 3.9%.

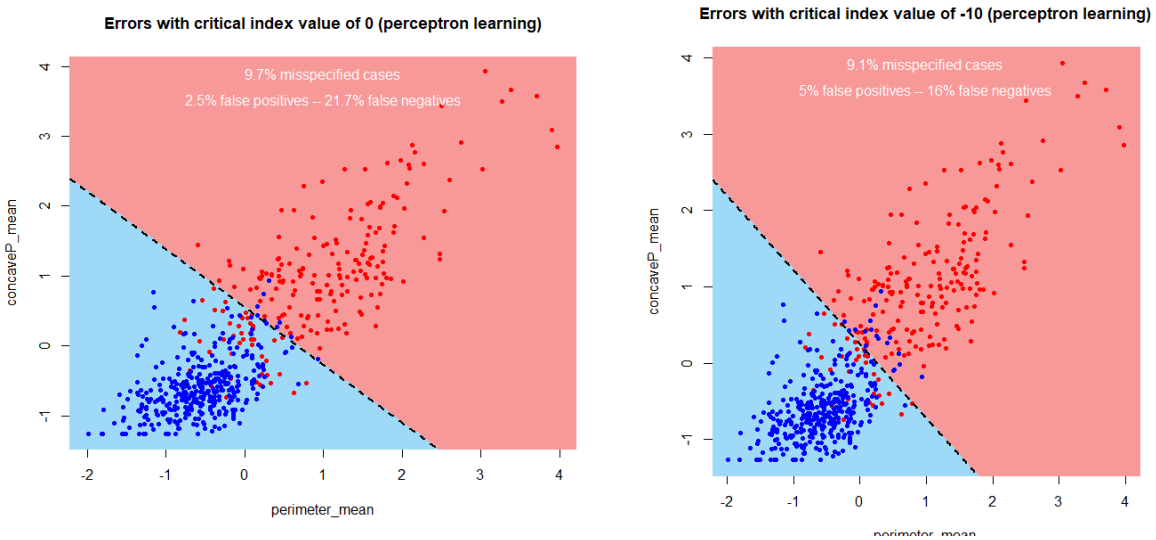
Question 2

Say you are responsible for the project of automated cancer diagnosis based on the WDBC data. Which type of error (false negative or false positive) would you minimize, and why?

If we are responsible for the project of automated cancer diagnosis, we would minimize the false negative errors. We believe it is more important to be sure that the malign is diagnosed precisely and we would trade off some accuracy in the false positives. The utility we get from diagnosing precisely the malign cases is larger. By decreasing the number of false negatives, we increase the possibility of saving more lives since malign cancer is considerably more dangerous than benign. The downsides of having more patients doing wrong treatments in the case they are benign are not as bad as having more people risking their lives.

Question 3

You can fin-tune the trade-off between false positives and false negatives with the parameter *decCrit*? What happens (to the decision boundary and the number of false positives/negatives) if you increase or decrease this value?

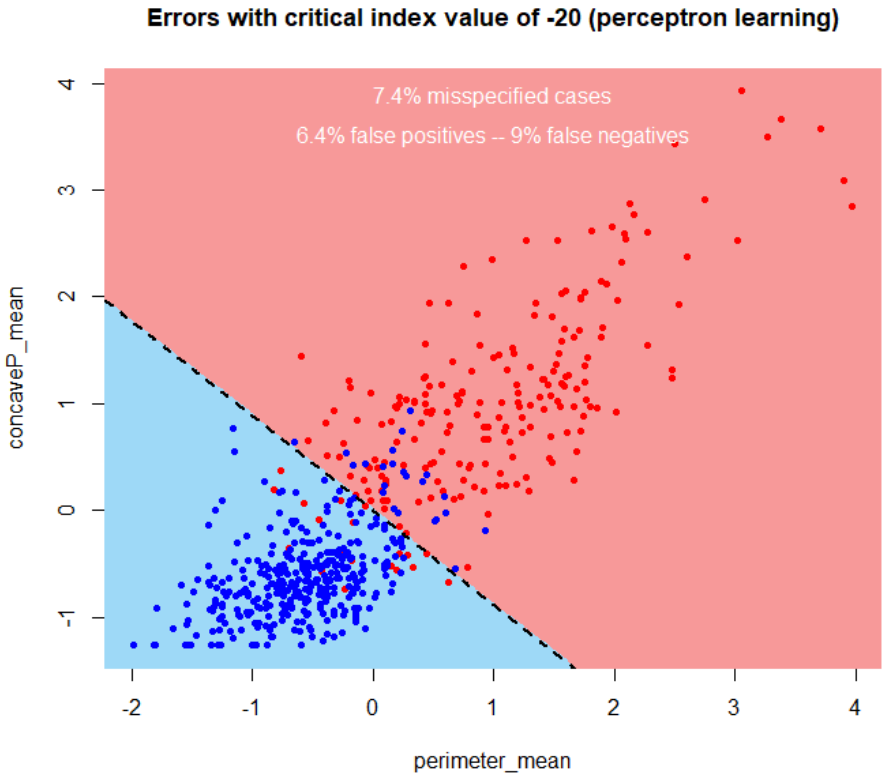


Yes, you can fine tune the trade-off. Instead of critical index value of -5, we show two graph with critical index values of 0 and -10. According to the value of the critical index, both decision boundaries tilt counter-clockwise. Although there is not a formal test of the relationship between critical index value and false negative/positive, we find a possible pattern:

- The smaller the critical index, the smaller the percentage of false negatives;
- The larger the critical index, the smaller the percentage of false positives.

Question 4

Based on your answer to the previous question, run different version of the script. Which value of *decCrit* would you advise?



The optimal value for decCrit in our opinion is -20. As we explained earlier in the exercise, the priority is to avoid false negatives because we don't want people with malign breast cancer to be diagnosed as benign cancer mistakenly. However, we do not want to sacrifice the integrity of the experiment by only focusing on false negatives. Instead, We also want to minimize the misspecified cases to improve the accuracy of the automated cancer diagnosis.

The value of -20 gives a good overall performances with 9% of false negative, 6.4% of false positives and 7.4% of misspecified cases. Compared to other solutions, this values offers very good results given the objectives of minimizing both false negatives and misspecified cases.

Exercise 3

** Compute the mean and the sample standard deviation of x_1, x_2, x_3, repectively, and normalize the folling data (you can do this by hand or by using R):

	D_1	D_2	D_3	D_4
x_1	13	-21	7	11
x_2	-2	-1.2	2	0.5
x_3	-1.3	2.4	2	-1

Note that, as in class, the row labels are the features and the column are the four data points. **

```
# Exercise 3 -----------------------------------------------------------

# set up
# for the ease of future computation I will invert the columns and rows of the table presented in the h

x1 = c(13, -21, 7, 11)
x2 = c(-2, -1.2, 2, 0.5)
x3 = c(-1.3, 2.4, 2, -1)

table_inverted = data.frame(x1, x2, x3)
rownames(table_inverted) = c('D1', 'D2', 'D3', 'D4')

### computing the means and the sample standard deviations of x1, x2, x3
# means
mean_x1 = mean(x1)
mean_x2 = mean(x2)
mean_x3 = mean(x3)
means = c(mean_x1, mean_x2, mean_x3)

# standard deviations
std_x1 = sd(x1)
std_x2 = sd(x2)
std_x3 = sd(x3)
standard_deviations = c(std_x1, std_x2, std_x3)

# summing them up in a table
statistics = data.frame(means, standard_deviations)
rownames(statistics) = c('x1', 'x2', 'x3')
colnames(statistics) = c('mean', 'standard deviation')

# standardizing the data
table_standard = data.frame(scale(table_inverted))
```

The means and the sample standard deviations are computed following the procedure above. The results are shown below:

```
print(statistics)
```

```
##      mean standard deviation
## x1  2.500          15.864005
## x2 -0.175           1.785824
## x3  0.525           1.944865
```

To normalize the following data, we use the formula

$$x_k \rightarrow \frac{x_k - E[x_k]}{\sigma(x_k)}$$

where $E[x_k]$ denotes the mean (expected value) and $\sigma(x_k)$ denotes the sample standard deviation of x_k. Note that the sample standard deviation is calculated from the formula $\sigma(x_k) = \sqrt{\frac{n}{n-1}(E[x_k^2] - E[x_2]^2)}$.

In R, the command *scale()* uses the same formula, so we use the formula to avoid multiple calculations. The results are shown below.

```
print(table_standard)
```

```
##            x1         x2         x3
## D1  0.6618757 -1.0219373 -0.9383685
## D2 -1.4813409 -0.5739648  0.9640772
## D3  0.2836610  1.2179253  0.7584074
## D4  0.5358042  0.3779768 -0.7841161
```