Compulsory exercise 1: Group 13

TMA4268 Statistical Learning V2019

Erik Andrè Klepp Vik and Vemund Tjessem

16 februar, 2020

Problem 1

For this problem you will need to include some LaTex code. Please install latex on your computer and then consult Compulsor1.Rmd for hints how to write formulas in LaTex.

a)

The expression for the test mean squared error is

$$\text{MSE}_{\text{test}} = \frac{1}{n_0}\sum_{j=1}^{n_0}(y_{0j} - \hat{f}(x_{0j}))^2 \tag{1}$$

The expected test mean squared error (MSE) at x_0 is defined as:

$$\text{E}[y_0 - \hat{f}(x_0)]^2 \tag{2}$$

b)

Using the fact that $y_0 = f(x_0) + \varepsilon$ gives

$$\text{E}[y_0 - \hat{f}(x_0)]^2 = \underbrace{\text{Var}(\varepsilon)}_{\text{Irreducible error}} + \underbrace{\text{Var}(\hat{f}(x_0))}_{\text{Variance of prediction}} + \underbrace{\Big(f(x_0) - \text{E}[\hat{f}(x_0)]\Big)^2}_{\text{Squared bias}} \tag{3}$$

c)

d)

e)

f)

g)

Problem 2

Here is a code chunk:

```
id <- "1nLen1ckdnX4P9n8ShZeU7zbXpLc7qiwt"  # google file ID
d.worm <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
head(d.worm)
```

```
##   Gattung Nummer GEWICHT FANGDATUM MAGENUMF
## 1      Oc     32    0.19  23.09.97     1.56
## 2      Oc     34    0.59  23.09.97     1.63
## 3      Oc     48    0.09  23.09.97     1.69
## 4      Oc     55    0.23  23.09.97     1.69
## 5      Oc     41    0.24  23.09.97     1.75
## 6      Oc     24    0.19  23.09.97     1.81
```

a)

b)

Below you have to complete the code and then replace `eval=FALSE` by `eval=TRUE` in the chunk options:

```
ggplot(d.worm, aes(x = ..., y = ..., colour = ...)) + geom_point() + theme_bw()
```

Note that the default figure width and height have been set globally as `fig.width=4, fig.height=3`, but if you would like to change that (e.g., due to space constraints), you can include a different width and height directly into the chunk options, again using `fig.width=..., fig.height=...`.

c)

d)

e)

f)

g)

Problem 3

Loading the files:

```
id <- "1GNbIhjdhuwPOBr0Qz82JMkdjUVBuSoZd"
tennis <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id),
    header = T)
head(tennis)
```

```
##         Player1      Player2 Result ACE.1 UFE.1 ACE.2 UFE.2
## 1     M.Koehler   V.Azarenka      0     2    18     3    14
## 2    E.Baltacha   F.Pennetta      0     0    10     4    14
## 3     S-W.Hsieh      T.Maria      1     1    13     2    29
## 4      A.Cornet       V.King      1     4    30     0    45
## 5  Y.Putintseva   K.Flipkens      0     2    28     6    19
## 6 A.Tomljanovic B.Jovanovski      0     6    42    11    40
```

a)

We have

$$p_i = \frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}} \tag{4}$$

which gives

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \log\left(\frac{\frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}}{1-\frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}}\right) \tag{5}$$

Multiplying both the numerator and denominator in Equation 5 by $1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}$ gives

$$\text{logit}(p_i) = \log\left(\frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}-\left(e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}\right)}\right) \tag{6}$$

which further results in

$$\text{logit}(p_i) = \log(e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}) = \beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4} \tag{7}$$

It can be seen from Equation 7 that $\text{logit}(p_i)$ is a linear function of the covariates x_{i1}, x_{i2}, x_{i3} and x_{i4}.

b)

```
##
## Call:
## glm(formula = Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, family = "binomial",
##     data = tennis)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0517  -0.8454   0.3725   0.8773   2.0959
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.02438    0.59302  -0.041 0.967211
## ACE.1        0.36338    0.10136   3.585 0.000337 ***
## ACE.2       -0.22388    0.07369  -3.038 0.002381 **
## UFE.1       -0.09847    0.02840  -3.467 0.000527 ***
## UFE.2        0.09010    0.02479   3.635 0.000278 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.04  on 117  degrees of freedom
## Residual deviance: 124.96  on 113  degrees of freedom
## AIC: 134.96
##
## Number of Fisher Scoring iterations: 4
```

Since the β_1 is positive one more ace for player one would increase the probability of player 1 winning. By incerasing x_i by 1 the odds ratio for $Y_i = 1$ increases by $\exp(\beta_1)$.

c)

```
# make variables for difference
tennis$ACEdiff = tennis$ACE.1 - tennis$ACE.2
tennis$UFEdiff = tennis$UFE.1 - tennis$UFE.2

# divide into test and train set
n = dim(tennis)[1]
n2 = n/2
set.seed(1234)  # to reproduce the same test and train sets each time you run the code
train = sample(c(1:n), replace = F)[1:n2]
tennisTest = tennis[-train, ]
tennisTrain = tennis[train, ]
```

The code for fitting a logistic regression model is given below.

```
# Fitting a logistic regression model on the form Result ~ ACEdiff + UFEdiff on
# the training set
fit.3c = glm(Result ~ ACEdiff + UFEdiff, data = tennisTrain, family = "binomial")
summary(fit.3c)
```

```
##
## Call:
## glm(formula = Result ~ ACEdiff + UFEdiff, family = "binomial",
##     data = tennisTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8546  -0.8968   0.4204   0.8247   1.9382
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.28272    0.31175   0.907  0.36447
## ACEdiff      0.22355    0.07959   2.809  0.00497 **
## UFEdiff     -0.08607    0.02832  -3.039  0.00237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 80.959  on 58  degrees of freedom
## Residual deviance: 63.476  on 56  degrees of freedom
## AIC: 69.476
##
## Number of Fisher Scoring iterations: 4
```

The class boundary will be where $\hat{P}(Y = 1|\boldsymbol{x}) = 0.5$. When the probability is 0.5 we have $\text{logit}(p_i) = \log(1) = 0$

$$0 = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{8}$$

This gives the class boundary

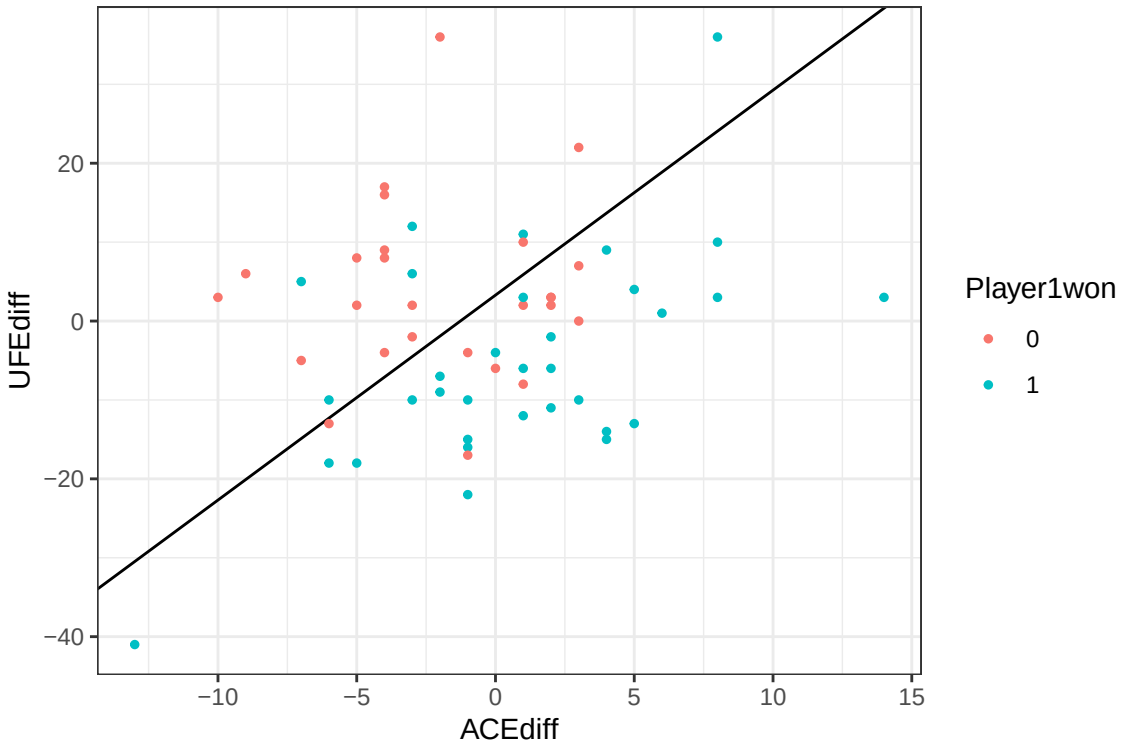
$$x_2 = -\frac{\beta_0}{\beta 2} - \frac{\beta_1}{\beta_2} x_1 \tag{9}$$

```
cof = coef(fit.3c)
a = -cof[1]/cof[3]
b = -cof[2]/cof[3]
```

The class boundary will be $x_2 = 3.285 + 2.597 x_1$

Making a plot of the training observations and the class boundary.

```
df = data.frame(tennisTrain, Player1won = as.factor(tennisTrain$Result))
ggplot(df, aes(x = ACEdiff, y = UFEdiff, color = Player1won)) + geom_abline(intercept = a,
    slope = b) + geom_point(size = 1) + theme_bw()
```



Making a confusion matrix

```
prd = predict(fit.3c, tennisTest, type = "response")
confMat = table(tennisTest$Result, prd > 0.5)
confMat
```

```
##
##      FALSE TRUE
##   0     22    7
##   1      6   24
```

The sensitivity is 0.8 and the specificity is 0.759

d)

- π_k is the prior class probabilities $\pi_k = \Pr(Y = k)$. In this case it will be the probability for player 1 winning and for player 1 losing.
- $\boldsymbol{\mu}_k$ is the mean of class k. In this case it will be a vector with the mean of difference in aces and difference in unforced errors.
- $\boldsymbol{\Sigma}$ is the covariance matrix, and in this case it is assumed equal for both classes. The diagonal elements will be the variance of the difference in aces the variance of the difference in unforced errors and the off-diagonal elements will be the covariance.
- $f_k(\boldsymbol{x})$ is the probability distribution of class k, and is assumed to be multivariate normally distributed with mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}$.

e)

Part 1

$$P(Y = 0|\mathbf{X} = \boldsymbol{x}) = P(Y = 1|\mathbf{X} = \boldsymbol{x}) \tag{10}$$

The first step is to insert for the probability and the probability distribution.

$$\frac{\pi_0}{\sum_{l=1}^{K} \pi_l f_l(\boldsymbol{x}) 2\pi|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_0})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_0})} = \frac{\pi_1}{\sum_{l=1}^{K} \pi_l f_l(\boldsymbol{x}) 2\pi|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_1})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1})} \tag{11}$$

Next is taking the logarithm on both sides, which gives

$$\log(\pi_0) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 + \boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 - \log\left(\sum_{l=1}^{K} \pi_l f_l(\boldsymbol{x}) 2\pi|\boldsymbol{\Sigma}|^{1/2}\right) = \tag{12}$$

$$\log(\pi_1) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 - \log\left(\sum_{l=1}^{K} \pi_l f_l(\boldsymbol{x}) 2\pi|\boldsymbol{\Sigma}|^{1/2}\right) \tag{13}$$

After removing the terms not depending on k, which are equal on both sides

$$\log(\pi_0) + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 + \frac{1}{2}\boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 = \log(\pi_1) + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 \tag{14}$$

which equals

$$\delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x}) \tag{15}$$

Part 2

The class boundary will be the values for $\boldsymbol{x}$ where $\delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x})$, hence Equation 14 can be used to find the class boundary.

$$\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\mu_0 - \mu_1) = \log\left(\frac{\pi_1}{\pi_0}\right) + \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 - \frac{1}{2}\boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 \tag{16}$$

Doing the matrix multiplications here will result in a equation on the form $ax_1 + bx_2 = c$, which can be used to find the class boundary on the form $x_2 = \frac{c}{b} - \frac{a}{b}x_1$.

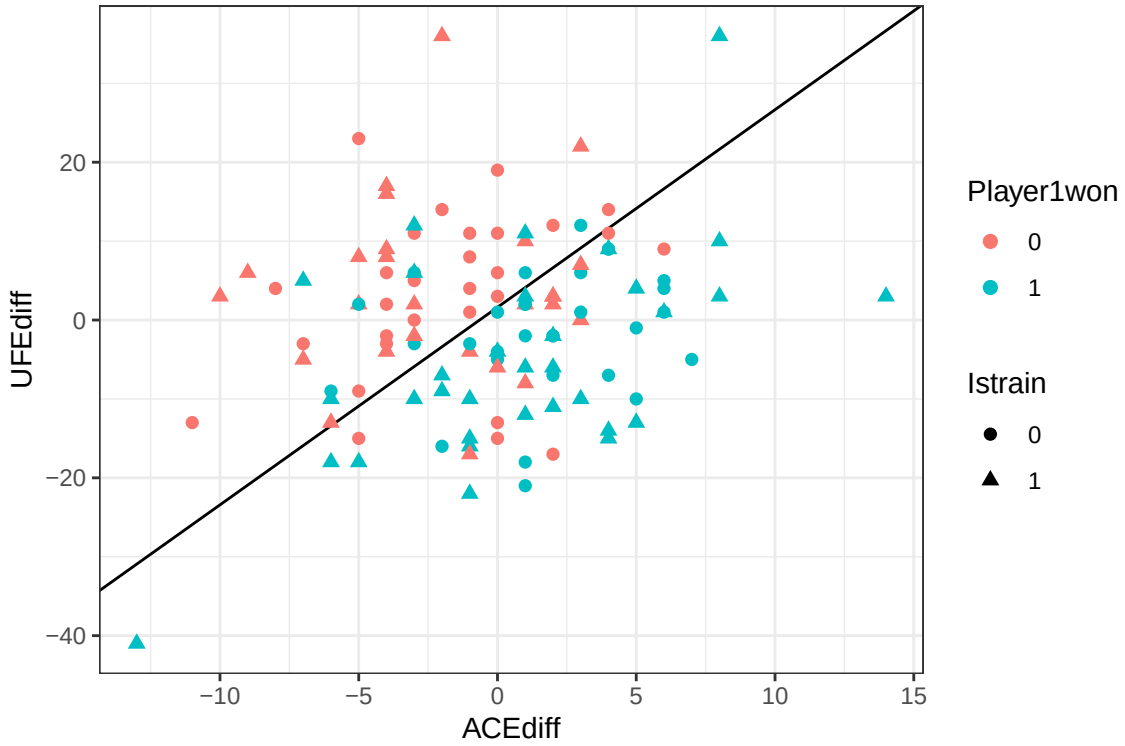
```
k0s = subset(tennisTrain, tennisTrain$Result < 0.5)
k1s = subset(tennisTrain, tennisTrain$Result > 0.5)
n0 = length(k0s$UFEdiff)
n1 = length(k1s$UFEdiff)
pi0 = n0/(n0 + n1)
pi1 = n1/(n0 + n1)
mu0 = matrix(c(mean(k0s$ACEdiff), mean(k0s$UFEdiff)), nrow = 2)
mu1 = matrix(c(mean(k1s$ACEdiff), mean(k1s$UFEdiff)), nrow = 2)
covmat0 = cov(cbind(k0s$ACEdiff, k0s$UFEdiff))
covmat1 = cov(cbind(k1s$ACEdiff, k1s$UFEdiff))
covK = 1/(n0 + n1) * ((n0 - 1) * covmat0 + (n1 - 1) * covmat1)
c = log(pi1/pi0) + 0.5 * t(mu1) %*% solve(covK) %*% mu1 - 0.5 * t(mu0) %*% solve(covK) %*%
```

```
    mu0
lhs = solve(covK) %*% (mu0 - mu1)
a = lhs[1]
b = lhs[2]
```

Find that $a = -0.228$, $b = 0.091$ and $c = 0.147$, which gives the class boundary $x_2 = 1.616 + 2.504x_1$

Part 3

```
tennis$istrain = 1
tennis[-train, ]$istrain = 0
df2 = data.frame(tennis, Player1won = as.factor(tennis$Result), Istrain = as.factor(tennis$istrain))
ggplot(df2, aes(x = ACEdiff, y = UFEdiff, color = Player1won, shape = Istrain)) +
    geom_abline(intercept = c/b, slope = -a/b) + geom_point(size = 2) + theme_bw()
```



f)

```
lda.fit = lda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)
lda.fit.p = predict(lda.fit, tennisTest)$class
confMat = table(lda.fit.p, tennisTest$Result)
confMat
```

```
##
## lda.fit.p  0  1
```

```
##         0 20  5
##         1  9 25
```

The sensitivity is 0.735 and the specificity is 0.8

g)

```
qda.fit = qda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)
qda.fit.p = predict(qda.fit, tennisTest)$class
confMat = table(qda.fit.p, tennisTest$Result)
confMat
```

```
##
## qda.fit.p  0  1
##         0 20  6
##         1  9 24
```

The sensitivity is 0.727 and the specificity is 0.769

h)

Problem 4

a)

b)

c)

```
id <- "1I6dk1fA4ujBjZPo3Xj8pIfnzIa94WKcy"  # google file ID
d.chd <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
fit.4c = glm(chd ~ sbp + sex, data = d.chd, family = "binomial")
summary(fit.4c)
```

```
##
## Call:
## glm(formula = chd ~ sbp + sex, family = "binomial", data = d.chd)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0647  -0.8697  -0.7749   1.4191   1.7794
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.386252   0.790657  -3.018  0.00254 **
## sbp          0.011337   0.006273   1.807  0.07075 .
## sex          0.322764   0.235786   1.369  0.17103
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 427.61  on 349  degrees of freedom
## Residual deviance: 422.63  on 347  degrees of freedom
## AIC: 428.63
##
## Number of Fisher Scoring iterations: 4
```

d)