Compulsory exercise 1: Group 3

TMA4268 Statistical Learning V2019

Elsie Backen Tandberg, Magnus Wølneberg

20 February, 2020

Problem 1

a)

This is the definiton of the expected test mean squared error.

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

b)

$$\begin{gathered}
\mathrm{E}[f(x_0) + \varepsilon - \hat{f}(x_0)]^2 \\
\mathrm{E}[f(x_0)^2] + \mathrm{E}[\varepsilon^2] + \mathrm{E}[\hat{f}(x_0)^2] - 2\mathrm{E}[f(x_0)\hat{f}(x_0)] \\
f(x_0)^2 + \mathrm{Var}(\varepsilon) + \mathrm{Var}(\hat{f}(x_0)) + \mathrm{E}[\hat{f}(x_0)]^2 \\
-2f(x_0)\mathrm{E}[\hat{f}(x_0)] \\
\mathrm{Var}(\varepsilon) + \mathrm{Var}(\hat{f}(x_0)) + \Big(f(x_0) - \mathrm{E}[\hat{f}(x_0)]\Big)^2
\end{gathered}$$

c)

The term $\mathrm{Var}(\varepsilon)$ is the irreducible error. It can not be reduced even though we get at better model. The term $\mathrm{Var}(\hat{f}(x_0))$ is the variance of prediction, if this is large the model does not generlize the data and tend to overfit. The term $\Big(f(x_0) - \mathrm{E}[\hat{f}(x_0)]\Big)^2$ is the squared bias, if this is large the model pay little attention to the training data and over simplifies.

d)

i) True
ii) False
iii) False
iv) True

e)

i) True
ii) False
iii) True
iv) False

f)

$$Corr = \frac{Cov(X_1X_2)}{\sqrt{VarX_1VarX_2}} = \frac{0.6}{\sqrt{3\cdot 4}} = 0.17$$

ii) The correlation between the two elements of X is 0.17.

g)

```r
m<-(cbind(c(1,0.2), c(0.2, 4)))
eigen(m)
```

```
## eigen() decomposition
## $values
## [1] 4.0132746 0.9867254
##
## $vectors
##             [,1]        [,2]
## [1,] 0.06622726 -0.99780457
## [2,] 0.99780457  0.06622726
```

Plot C corresponds to the following covariance matrix. We know that the length of the half axes are $\sqrt{b * \lambda}$, where λ is the eigenvalues. We also know that the eigenvectors determine the rotation of the ellipse.

Problem 2

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

dim(d.worm)
```

```
## [1] 143   5
```

a)

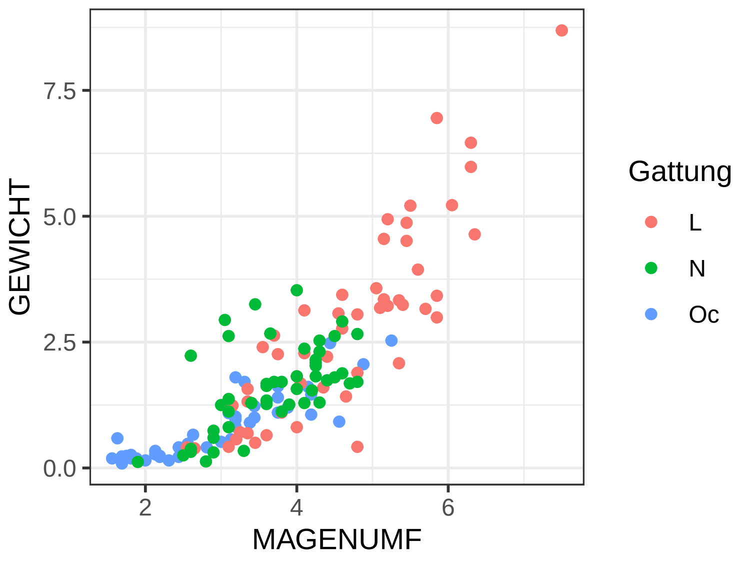
We find the dimensiones of the dataset using the function dim(), It is 143 rows and 5 coloumns.

Qualitative variables: Gattung, Nummer and Fangdatum

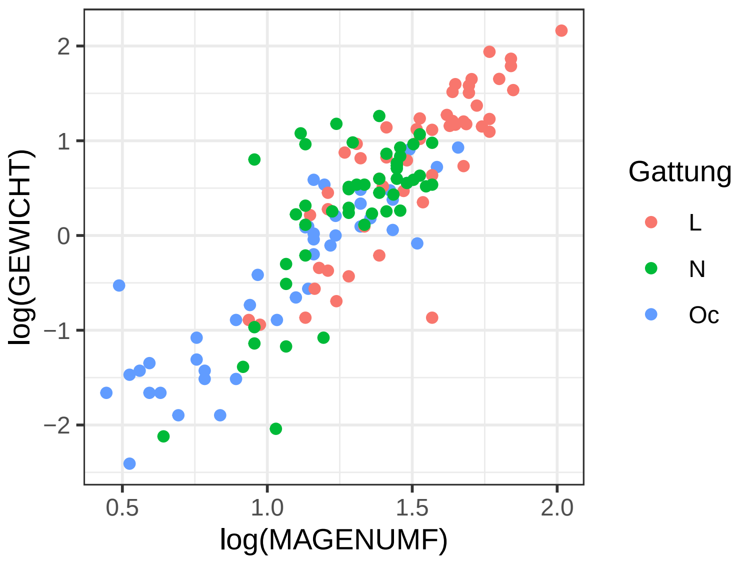
Quantitative variables: Gewicht and Magenumf

b)

```r
#Original plot
ggplot(d.worm,aes(x= MAGENUMF ,y= GEWICHT, color = Gattung)) + geom_point() + theme_bw()
```



```r
#Plot using log transformation
ggplot(d.worm,aes(x= log(MAGENUMF) ,y= log(GEWICHT), color = Gattung)) + geom_point() + theme_bw()
```



The first plot did not look linear, we decided to try to transform by using log. This is shown in the second plot and looks a lot better.

c)

We know fit the transformed data with a regression model.

```r
#Fitted regression model
L.lm <- lm(log(GEWICHT) ~ log(MAGENUMF) + Gattung , data=d.worm)
#Not fitted regression model
L2.lm <- lm(GEWICHT ~ MAGENUMF + Gattung, data=d.worm)
summary(L.lm)
```

```
##
## Call:
## lm(formula = log(GEWICHT) ~ log(MAGENUMF) + Gattung, data = d.worm)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.79642 -0.25996  0.02864  0.25159  1.40557
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -3.13865    0.24106 -13.020   <2e-16 ***
## log(MAGENUMF)  2.59310    0.15378  16.862   <2e-16 ***
## GattungN       0.07327    0.10252   0.715    0.476
## GattungOc     -0.06149    0.12219  -0.503    0.616
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4832 on 139 degrees of freedom
## Multiple R-squared:  0.7598, Adjusted R-squared:  0.7547
## F-statistic: 146.6 on 3 and 139 DF,  p-value: < 2.2e-16
```

```r
anova(L.lm)
```

```
## Analysis of Variance Table
##
## Response: log(GEWICHT)
##                Df  Sum Sq Mean Sq  F value Pr(>F)
## log(MAGENUMF)   1 102.264 102.264 438.0809 <2e-16 ***
## Gattung         2   0.394   0.197   0.8441 0.4321
## Residuals     139  32.448   0.233
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

L : $log(GEWICHT) = -3.13865 + 2.5931 * log(MAGENUMF)$

N : $log(GEWICHT) = -3.06538 + 2.5931 * log(MAGENUMF)$

Oc : $log(GEWICHT) = -3.20014 + 2.5931 * log(MAGENUMF)$

Gattung does not seem to be a relevant predictor because the p-value Pr(>F)=0.4321 is large compared to the other p-values.

d)

Here we look at the interaction between the variates.

```r
#Modell med interaction
N.lm = lm(log(GEWICHT) ~ log(MAGENUMF) + Gattung + log(MAGENUMF)*Gattung, data=d.worm)

summary(N.lm)
```

```
##
## Call:
## lm(formula = log(GEWICHT) ~ log(MAGENUMF) + Gattung + log(MAGENUMF) *
##     Gattung, data = d.worm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8114 -0.2830  0.0185  0.2457  1.4483
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -3.49061    0.42376  -8.237 1.25e-13 ***
## log(MAGENUMF)             2.82702    0.27804  10.168  < 2e-16 ***
## GattungN                  0.19491    0.61075   0.319    0.750
## GattungOc                 0.52368    0.48810   1.073    0.285
## log(MAGENUMF):GattungN   -0.05431    0.43824  -0.124    0.902
## log(MAGENUMF):GattungOc -0.45640    0.35462  -1.287    0.200
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4831 on 137 degrees of freedom
## Multiple R-squared:  0.7633, Adjusted R-squared:  0.7547
## F-statistic: 88.36 on 5 and 137 DF,  p-value: < 2.2e-16
```

```r
anova(N.lm)
```

```
## Analysis of Variance Table
##
## Response: log(GEWICHT)
##                          Df  Sum Sq Mean Sq  F value Pr(>F)
## log(MAGENUMF)             1 102.264 102.264 438.1138 <2e-16 ***
## Gattung                   2   0.394   0.197   0.8442 0.4321
## log(MAGENUMF):Gattung     2   0.469   0.235   1.0052 0.3686
## Residuals               137  31.978   0.233
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

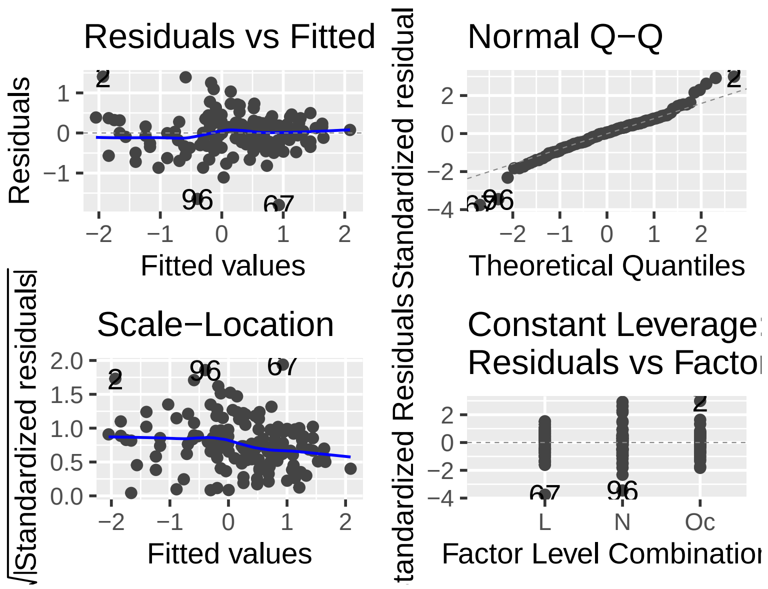
By looking at the summary you can see that the Gattung does not affect the Gewicht. Both Gattung and the interaction between Gattung and MAGENUMF seems to be irrelevant for the model, since both have large p-values.

e)

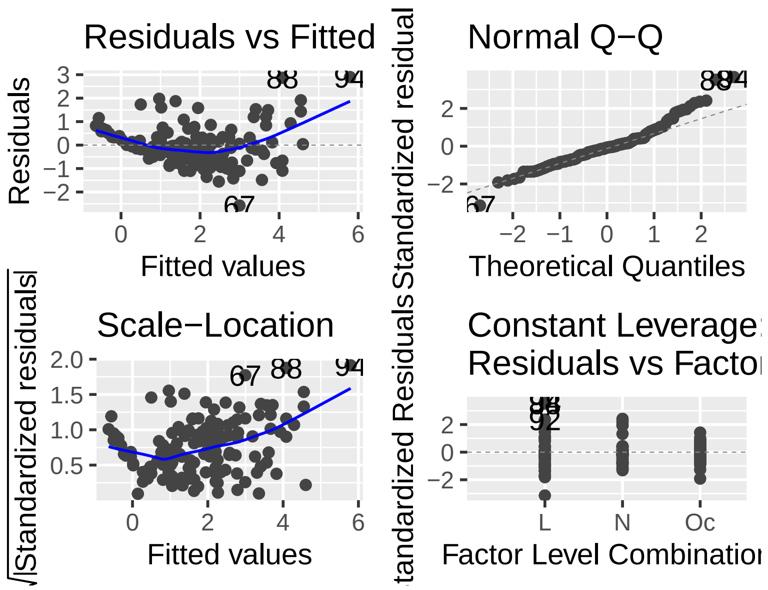
The assumptions for a linear model is: The expected value of ϵ_i is 0: $E(\epsilon_i) = 0$. All ϵ_i have the same variance: $Var(\epsilon_i) = \sigma$. The ϵ_i are normally distributed. The ϵ_i are independent of each other.

To check if the assumptions are fullfilled you can use residual plots.

```r
autoplot(L.lm)
```



```r
autoplot(L2.lm)
```



Here we have plotted both for the transformed covariates and the original. Comparing the different plots, it looks better using the log transformation. Especially the Tukey-Anscombe plot(residuals vs fitted) is better, for the not tranfsormed model it looks like there is non-linearity. Even though the plots are way better when they are transformed it looks like there is a little bit of a trend in both the Residual vs Fitted and the Scale_location plot. This indicates that the model may not be that good and you could try to transform with something else.

f)

It is important to carry out residual analysis, because it looks at the appropriateness of a linear regression model. If the assumptions are not fullfilled the results will be wrong. When assumptions are not fullfilled you can try to transform them using for example log.

g)

i) False
ii) False
iii) False
iv) True

Problem 3

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

a)

We are looking at a logistic regression model with p_i as the probability of player 1 to win a tennis match. First we will show that $logit(p_i)$ is a linear function of the covariates.

$$logit(p_i) = log(\frac{p_i}{1-p_i}) = log(p_i) - log(1-p_i)$$

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

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

$$= log(\frac{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}} - e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}})$$

$$= log(1) - log(1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}})$$

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

$$= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4}$$

The result is a linear function of the covariates.

b)

```r
r.tennis = glm(Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, data = tennis, family = "binomial")
summary(r.tennis)
```

```
##
## 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
```

If player one get one more ace the result will change, the odds of player one winning will get $e^{0.36338} = 1.43$ times bigger. As shown by the oddsratio below.

$$\frac{e^{\beta_0+\beta_1 * (ACE.1+1)+\beta_2 * ACE.2+\beta_3 * UFE.1+\beta_4 * UFE.2}}{e^{\beta_0+\beta_1 * ACE.1+\beta_2 * ACE.2+\beta_3 * UFE.1+\beta_4 * UFE.2}} = e^{\beta_1}$$

c)

We know divide the data into a training set and a test set. Then we fit a logistic regression model on our training set.

```r
# 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, ]

#Fitting a logistic regression model
glm_tennis = glm(Result ~ ACEdiff + UFEdiff, data = tennisTrain, family = 'binomial')
summary(glm_tennis)
```

```
##
## 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
```

We then derive a decision rule using a 0.5 cutoff.

$$\frac{e^{\beta_0+\beta_1*ACE.diff+\beta_2*UFE.diff}}{1+e^{\beta_0+\beta_1*ACE.diff+\beta_2*UFE.diff}} = 0.5$$

$$e^{\beta_0+\beta_1*ACE.diff+\beta_2*UFE.diff} = 0.5(1+e^{\beta_0+\beta_1*ACE.diff+\beta_2*UFE.diff})$$

$$0.5e^{\beta_0+\beta_1*ACE.diff+\beta_2*UFE.diff} = 0.5$$

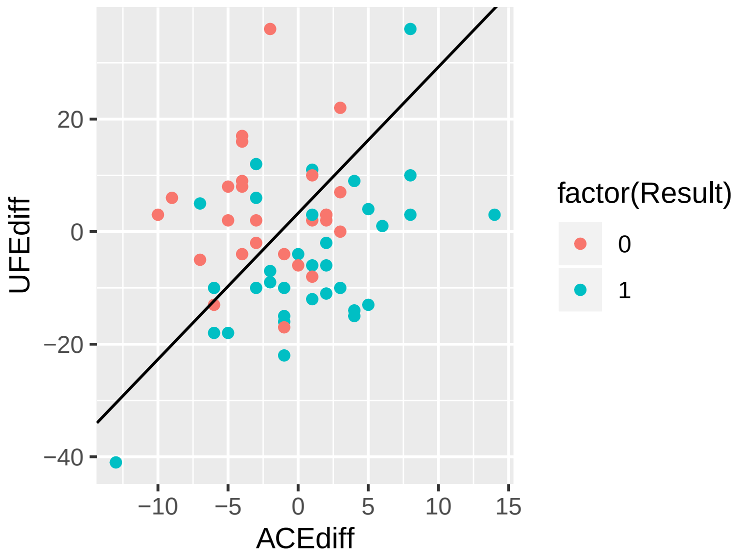
$$\beta_0 + \beta_1 * ACE.diff + \beta_2 * UFE.diff = 0$$

$$UFE.diff = -\frac{\beta_0}{\beta_2} - \frac{\beta_1}{\beta_2} ACE.diff$$

$$-\frac{\beta_0}{\beta_2} = 3.2847$$

$$-\frac{\beta_1}{\beta_2} = 2.5973$$

```
#Plotting the results and the line as the class boundary
ggplot(tennisTrain, aes(x= ACEdiff ,y= UFEdiff, color = factor(Result))) + geom_point() + geom_abline(s
```



We observe that the boundary seperate the variables well on the training set.

```
#Predicting the results of the test set
glm_probs_tennis <- predict(glm_tennis, tennisTest, type = "response")

GLMpred = ifelse(glm_probs_tennis > 0.5, "1", "0")

#Creating a confusion table with the actual result and the predicted.
t <- table(tennisTest$Result, GLMpred)
colnames(t) <- c('Predicted loss','Predicted win')
rownames(t) <- c('Actual loss', 'Actual win')
t
```

```
##                GLMpred
##                 Predicted loss Predicted win
##    Actual loss              22             7
##    Actual win                6            24
```

The confusion table shows how well the prediction is. The diagonal is the sum of the correct predictions. Only by observing the confusion table, we can conclude that the model did a quite good job. To get an even better understanding we can calculate the sensitivity and specificity. Since they are both high they confirm our suspicion about the model being good.

$$\text{Sens} = \frac{\text{True Positive}}{\text{Actual Positive}} = \frac{24}{24+6} = 0.8$$

$$\text{Spes} = \frac{\text{True Negative}}{\text{Actual Negative}} = \frac{22}{22+7} = 0.7586$$

d)

π_k is the prior probabilitiesnumber of observed k-events and can be interprented as the number of zeros or ones, divided by the total number of observations. mu_k is the mean of UFEdiff and ACEdiff. Σ is the common covariance matrix, which is a pooled version of the two covariance matrices belonging to each covariate. $f_k(x)$ is the multivariate normal distribution function for each k that explains the probability of a player winning or loosing.

e)

Knoiw we look at the class boundary for an LDA.

Given that

$$P(Y=0|X=x).$$

We have that

$$P(Y=k|X=x) = \frac{\pi_k f_k(x)}{\Sigma_{l=1}^K \pi_l f_l(x)}$$

and

$$f_k(x) = \frac{1}{2\pi^{p/2}\Sigma^{-1}} e^{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)}.$$

This gives

$$\frac{\pi_0 f_0(x)}{\pi_0 f_0(x) + \pi_1 f_1(x)} = \frac{\pi_1 f_1(x)}{\pi_0 f_0(x) + \pi_1 f_1(x)}$$
$$\pi_0 f_0(x) = \pi_1 f_1(x)$$

Inserting the equation $f_k(x)$

$$\pi_0 e^{-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)} = \pi_1 e^{-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)}$$

$$-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0) + log(\pi_0) = -\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) + log(\pi_1)$$

$$-\frac{1}{2}(x^T\Sigma^{-1}x - \underbrace{x^T\Sigma^{-1}\mu_0 - \mu_0^T\Sigma^{-1}x}_{-2x^T\Sigma^{-1}\mu_0} + \mu_0^T\Sigma^{-1}\mu_0) + log(\pi_0) = -\frac{1}{2}(x^T\Sigma^{-1}x - \underbrace{x^T\Sigma^{-1}\mu_1 - \mu_1^T\Sigma^{-1}x}_{-2x^T\Sigma^{-1}\mu_1} + \mu_1^T\Sigma^{-1}\mu_1) + log(\pi_1)$$

$$x^T\Sigma^{-1}\mu_0 - \frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0 + log(\pi_0) = x^T\Sigma^{-1}\mu_1 - \frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1 + log(\pi_1)$$

Which is what we wanted to prove.

```r
#Split tennistrain into matches won and lost
tennis0 <- tennisTrain[tennisTrain$Result == 0, 8:9]
tennis1 <- tennisTrain[tennisTrain$Result == 1, 8:9]

# find means
mu_0 <- c(mean(tennis0$ACEdiff), mean(tennis0$UFEdiff))
mu_1 <- c(mean(tennis1$ACEdiff), mean(tennis1$UFEdiff))
```

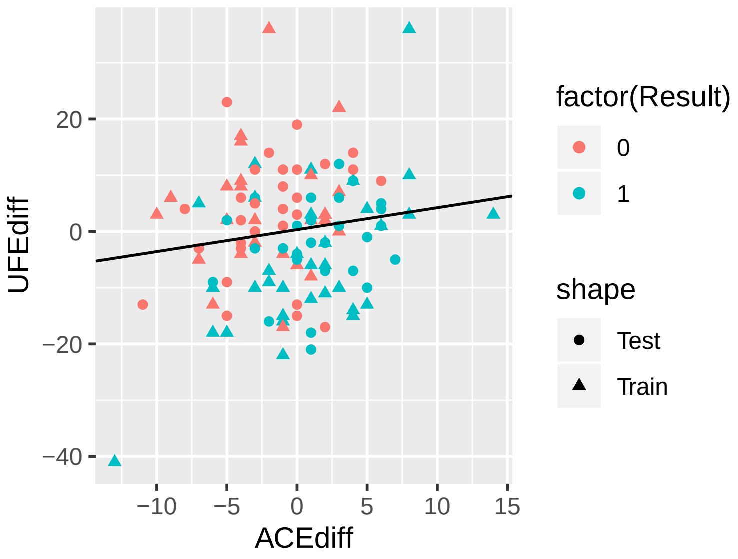
```r
#Find pi
pi_0 <- nrow(tennis0)/nrow(tennisTrain)
pi_1 <- nrow(tennis1)/nrow(tennisTrain)

## Create the covariance matrices
sigma_0 <- cov(tennis0[,c("ACEdiff", "UFEdiff")])
sigma_0
```

```
##           ACEdiff    UFEdiff
## ACEdiff 13.993846   2.098462
## UFEdiff  2.098462 120.764615
```

```r
sigma_1 <- cov(tennis1[,c("ACEdiff", "UFEdiff")])
sigma_1
```

```
##          ACEdiff   UFEdiff
## ACEdiff 27.56439  37.36742
## UFEdiff 37.36742 180.71780
```

```r
## Pool the covariance matrices
sigma <- (((nrow(tennis0) - 1)) * sigma_0 + (nrow(tennis1) - 1) * sigma_1) / (nrow(tennis) - 2)

x <- c(mean(tennisTrain$ACEdiff), mean(tennisTrain$UFEdiff))

#Calculate a and b

a = x %*% solve(sigma) %*% mu_0 - 0.5 * t(mu_0) %*% solve(sigma) %*% mu_0 - log(pi_0)

b = x %*% solve(sigma) %*% mu_1 - 0.5 * t(mu_1) %*% solve(sigma) %*% mu_1 - log(pi_1)

#Plot the training, the test set and the class boundary
ggplot(tennisTrain, aes(x= ACEdiff ,y= UFEdiff, color = factor(Result)) + geom_point(data = tennisTrai
```



f)

Performing LDA on the training data and classifying the results from the test set.

```r
#Creating an LDA
tennis_lda <- lda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)

#Gather the predictions
LDApred <- predict(tennis_lda, tennisTest)$class

#Creating a confusion matrix
t2<- table(tennisTest$Result, LDApred)
colnames(t2)=c("Predicted Loss","Predicted Win")
rownames(t2)=c("Actual Loss","Actual Win")
t2
```

```
##              LDApred
##               Predicted Loss Predicted Win
##   Actual Loss             20             9
##   Actual Win               5            25
```

$$\text{Sens} = \frac{\text{True Positive}}{\text{Actual Positive}} = \frac{25}{25+5} = 0.833$$
$$\text{Spes} = \frac{\text{True Negative}}{\text{Actual Negative}} = \frac{20}{20+9} = 0.6896$$

Looking at the confusion matrix, sensitivity and specificity we conclude that LDA does reasonable predictions.

g)

Perforing QDA on the training set and classifying the results from the test set.

```r
#Creating an QDA
tennis_qda <- qda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)

#Gather the predictions
QDApred <- predict(tennis_qda, tennisTest)$class

#Creating a confusion matrix
t3 <- table(tennisTest$Result, QDApred)
colnames(t3)=c("Predicted Loss","Predicted Win")
rownames(t3)=c("Actual Loss","Actual Win")
t3
```

```
##              QDApred
##               Predicted Loss Predicted Win
##   Actual Loss             20             9
##   Actual Win               6            24
```

$$\text{Sens} = \frac{\text{True Positive}}{\text{Actual Positive}} = \frac{24}{24+6} = 0.8$$
$$\text{Spes} = \frac{\text{True Negative}}{\text{Actual Negative}} = \frac{20}{20+9} = 0.6896$$

QDA has just a little bit lower sensitivty, so we think that LDA is a better fit.

h)

We observe that the decisioun boundary for QDA is not linear, but rather quadratic. Comparing to the plots for logistic regression and LDA, which both have linear boundaries, we observe that all tree plots work quite well. LDA assumes $\Sigma_k = \Sigma$ for all classes, where QDA is more flexible as it allows group-specific covariance matrices.

From our confusion matrices and the plots, we see that QDA misses one more point than LDA and the logistic regression. This is probably because the QDA is overfitting. This means that we would choose either LDA or logistic regression.

Problem 4

a)

The 10-fold cross validation is preformed by splitting your dataset into ten groups. Choose one group as a test set, and the remaining nine as training sets. Fit a model to the training sets and validate it on the test set. Then you do the same for each of the uniqe gropus as a test set.

To predict the error we use the mean squared error for each iteration.

$$MSE_k = \frac{1}{n_k} \sum_{i \in C_k} (y_i - \hat{y})^2$$

The cross validation is given by this formula:

$$CV_k = \frac{1}{n} \sum_{j=1}^{k} n_j MSE_j$$

b)

i) True
ii) True
iii) False
iv) False

c)

We perform a logistic regression and try to predict the probability of a male with $sbp = 140$ getting chd.

```r
id <- "1I6dk1fA4ujBjZPo3Xj8pIfnzIa94WKcy"  # google file ID
d.chd <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))

set.seed(1)

#Creates a logistic regression
GLMchd <- glm(chd ~ sbp + sex, data = d.chd, family = "binomial")
summary(GLMchd)
```

```
##
## 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
```

```
#Finding the probability of chd for a male with sbp = 140
new_data = data.frame(sbp = 140,sex=1)
p = predict(GLMchd,new_data,type="response")
p
```

```
##         1
## 0.3831131
```

The probability of a male with $sbp = 140$ to get chd is estimated to be 0.3831131.

d)

Running $B = 1000$ bootsprap samples to estimate the uncertainty of the probability derived in 4c.

```
n = nrow(d.chd)
new_data = data.frame(sbp = 140, sex=1)
estimated_prob = c() ##list of probabilities from each bootstrap sample
beta_0 = c()
B = 1000
for (i in 1:B) { #Running a loop 1000 times
index<- sample(n, n, replace = T) ##make a bootstrap sample of the data consisting of n observations ea
model<-glm(chd ~ sbp+sex, data = d.chd, subset = index,family = "binomial") ##fit a model
estimated_prob[i] = predict(model,new_data,type = "response") ##save probability for newdata
}
```

Calculating the standard error from the set of estimated probabilities. Then find the confidence interval with the following formula,

$$p \pm 1.96 \cdot SD(p)$$

.

```
standard_error<-sd(estimated_prob)
print(c("standard error:",standard_error))
```

```
## [1] "standard error:"    "0.0461336290260921"
```

```
#calculate confidence interval
c("confidence interval:",mean( estimated_prob) - 1.96 * sd(estimated_prob),mean(estimated_prob) + 1.96 *
```

```
## [1] "confidence interval:" "0.292723364570146"    "0.473567190352428"
```

The confidence interval tells us that there is a 95% probability of the probability of a male with $sbp = 140$ getting a chd to be between those values. We observe that the interval is quite large and that the uncertainty therefore also is quite large.