Lab 3 - Group report

Araya, Caroline, Gustav Sternelöv, Thomas

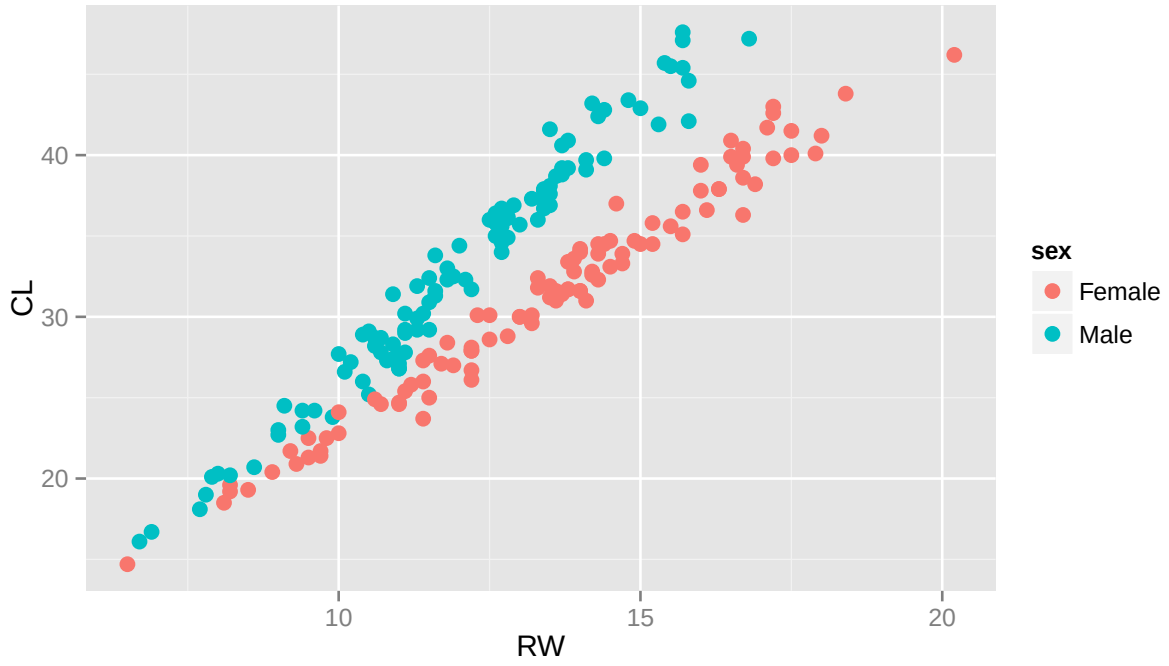
Monday, November 09, 2015

Assignment 1

In the first assignment a data material named australian crabs is analysed. It contanis information about measurements of the frontal lobe, rear witdh etc. for 200 crabs.

1.1

A scatterplot visualizing the carapace length versus rear width where observations are colored by sex.



We can clearly distinguish two separate linear trends in the scatterplot. One trend for the male crabs and one trend for the female crabs. I think it will be easy to classify the sex for the larger crabs, since at larger values of CL and RW they are well separated from each other. At smaller values for the two variables it will be harder to classify with linear discriminant analysis, since the two trends are not well separated there.

1.2

The discriminant functions can be written as:

$$\delta_k(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + log(\pi_k)$$

where

$$\hat{\mu}_c = \frac{1}{N_c}\Sigma_{i:y_i=c}x_i$$

$$\hat{\Sigma}_c = \frac{1}{N_c}\Sigma(x_i - \hat{\mu}_c)(x_i - \hat{\mu}_c)^T$$

$$\hat{\Sigma} = \frac{1}{N}\Sigma_{c=1}^{k} N_c \hat{\Sigma}_c$$

$$\hat{\pi}_c = \frac{N_c}{N}$$

After fitting the model, the discriminant function for the male crabs can then be written:

$$\delta_{male}(x) = x^T 0.029410201 \begin{pmatrix} 11.990 \\ 32.851 \end{pmatrix} - \frac{1}{2}\begin{pmatrix} 11.990 & 32.851 \end{pmatrix} 0.029410201 \begin{pmatrix} 11.990 \\ 32.851 \end{pmatrix} + log(0.50)$$

and the female function:

$$\delta_{female}(x) = x^T 0.029410201 \begin{pmatrix} 13.487 \\ 31.360 \end{pmatrix} - \frac{1}{2}\begin{pmatrix} 13.487 & 31.360 \end{pmatrix} 0.029410201 \begin{pmatrix} 13.487 \\ 31.360 \end{pmatrix} + log(0.50)$$

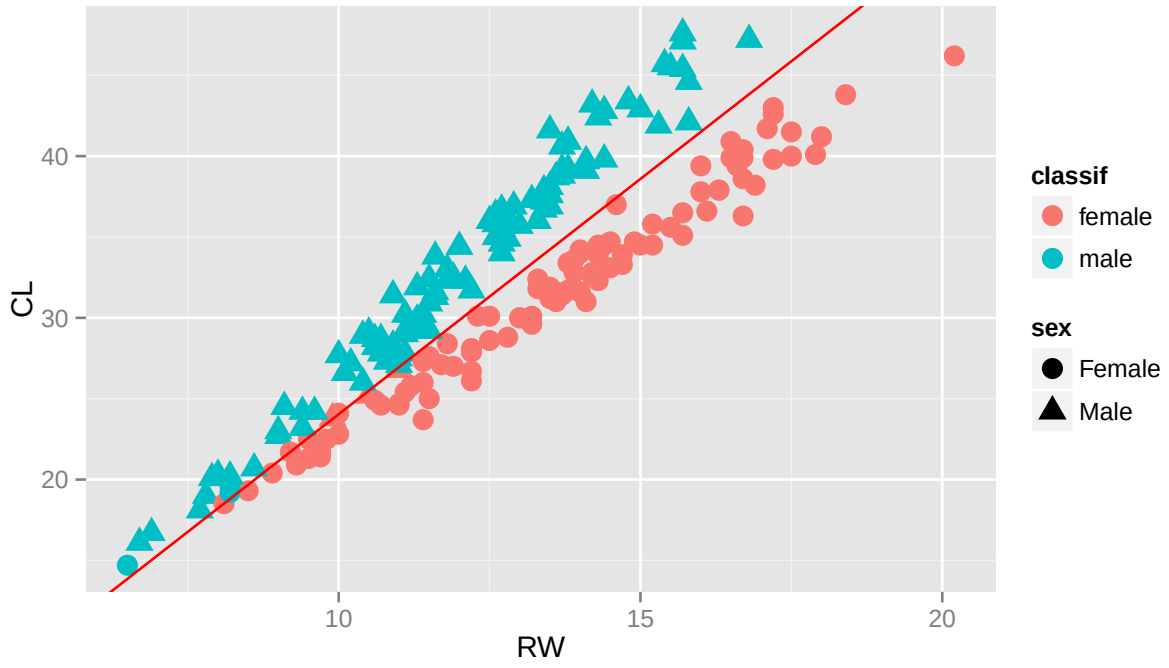
An observation is then classified to which ever discriminant function is greater.
Furthermore, the decision boundary is located where

$$\delta_{female}(x) = \delta_{male}(x)$$

And the estimated decision boundary is : CL = 2.91RW - 5.06

1.3

The original data of autralian-crabs.csv is plotted and coloured by the classification labels which are obtained from the function.

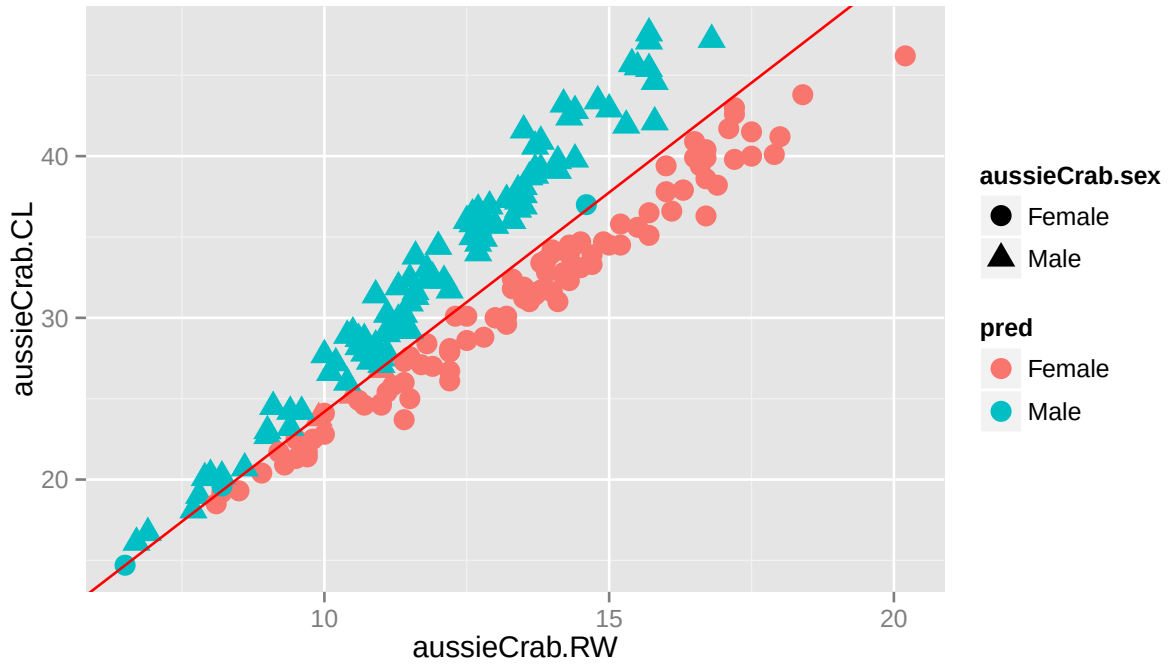



```
##         predict
## actual   female male
##   Female     97    3
##   Male        4   96
```

The red line indicates the decision boundary between female and male group. From looking at this plot, there are some misclassified data happened. The blue triangle indictes group as male while the orange circle represents female. Other symbols than these two (the blue triangle or the orange circle) suggest that it is misclassified. For example, the first point located at the left of the plot is in orange circle. The original is in the female group but when using LDA it is classified into male group. The quality of fit for using LDA is quite good. According to the confusion matrix which is given above, there are 7 observations which go into the wrong class.

1.4

Another method to classify the observations are by a logistic regression model. In the graph below the classifications are plotted in the same way as in 1.3.



The decision boundary for the logistic regression model: CL = 2.713RW - 2.94

The obtained result with logistic regression is very similar to the result given with the *lda* model. We see that the slope is a little bit less steep in the logistic regression decision boundary as compared to the LDA decision boundary.

Assignment 2

2.1 - 2.2

We use two measures of node impurity. For the first tree we use split = "deviance" which essentialy minimizes tree information entropy to pick the tree splits, and for second tree we use split = "gini", which minimizes tree Gini impurity. We find misclassification rates for the test data for the two trees.

Deviance: Misclassification rate for the training and test data, respectively.

```
## [1] 0.212
```

```
## [1] 0.268
```

Gini index: Misclassification rate for the training and test data, respectively.

```
## [1] 0.236
```

```
## [1] 0.364
```

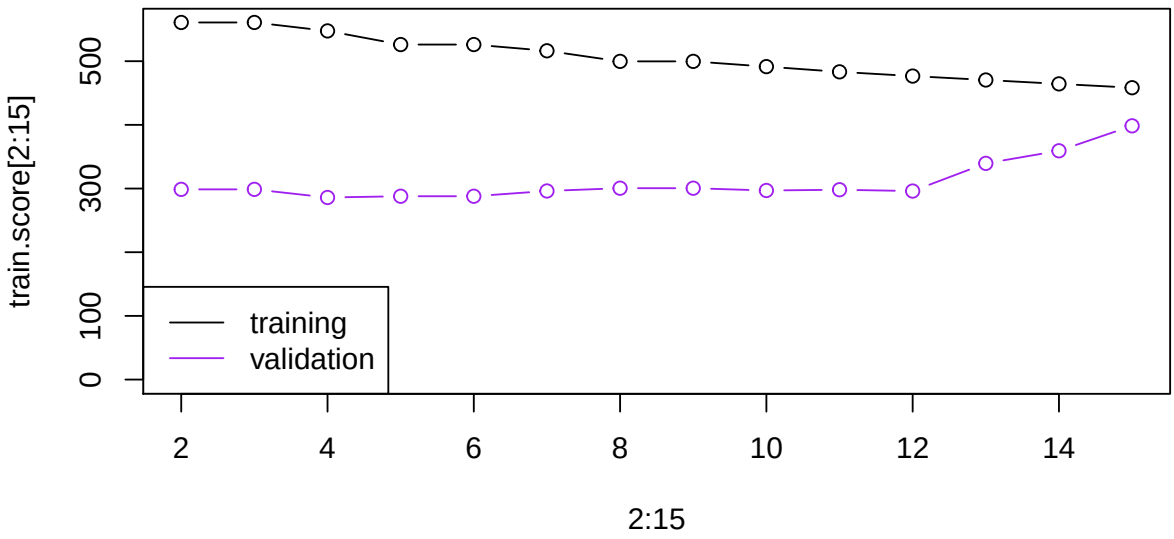
We see that the impurity measure of information entropy is better, so we will continue to evaluate the first decision tree.

2.3

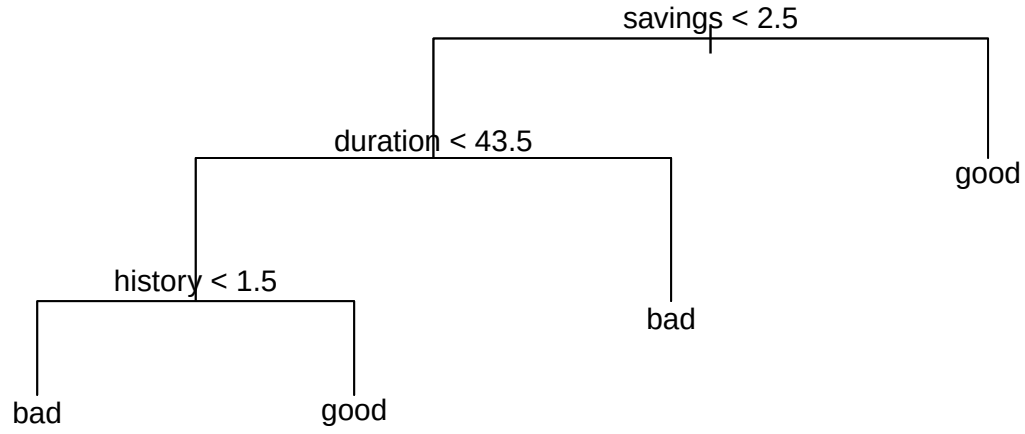
We are going to prune the chosen tree by minimizing

$$\text{Deviance of tree} + \lambda|T|$$

where $|T|$ is the number of terminal nodes and λ is an algorithmically determined cost parameter. Below you will find the plot of deviance of the tree for training data and for validation data against number of terminal nodes in the greedily and recursively pruned subtree of the original tree.



The purple line for validation data have the lower score than the training data. When the number if leaves is equal to four the lowest score is obtained. Therefore this plot suggests that the optimal tree for this dataset should have four leaves.



The tree presented above has a depth of 3 and uses 3 variables.
We see from the tree that the most important factor is that a borrower should have a high amount of savings. Further a loan of shorter duration are better credit risks than loans of longer duration and a good history is preferable.

The misclassification rate for test data is 0.256.

2.4

In 2.4 the classification of the costumers is performed with the Naive Bayes classifier. The first of the tables below gives the confusion matrix for training data and the second for test data.

```
##
## YfitBayesTrain bad good
##           bad   95   98
##           good  52  255
```

```
##
## YfitBayesTest bad good
##          bad   46   49
##          good  30  125
```

For training data the misclassification rate is 0.3 and for test data it is 0.316. The misclassification rates appear to be higher than for the unpruned decision trees.

2.5

We now use a loss matrix L,

$$L = \begin{matrix} \\ good \\ bad \end{matrix} \begin{matrix} \begin{matrix} good & bad \end{matrix} \\ \begin{pmatrix} 0 & 1 \\ 10 & 0 \end{pmatrix} \end{matrix}$$

where the rows are truth values and the columns are classified values in the naive bayes classifier, train a model based on training data and find the confusion matrices generated for the training data and test data.

Training data: Confusion matrix and misclassification rate

```
##       preds
##        bad good
##   bad  137   10
##   good 263   90
```

Test data: Confusion matrix and misclassification rate

```
##       preds
##        bad good
##   bad   71    5
##   good 122   52
```

The misclassification rate for training and test data are higher compared to the naive bayes classifier with a neutral loss function. When looking at the confusion matrix, it can be seen that the value of false "good" and true "good" for this result is much lower while false "bad" and true "bad" increase. This is due to the fact that loss matrix elements are changed to the given values above - a loss of 10 for the false "good". In most situations, we want to minimize as much as possible the number of falsely predicted "good" (customers are bad in mananging their loan but were predicted as having a good management) as the consequences are ten times more costly than the falsely predicted "bad". It would be better to make fewer mistakes of this kind, even if this was at the expense of predicting more falsely predicted "bad". Thus, the elevation in misclassification rate is the result of trying to minimize the falsely predicted "good".

Contribution from group members

All group members has contributed to the assignments with either code, text, plots or participation in discussions.
In 2.3 for the tree we got slightly different results. The tree presented in the report is Gustav's.

Appendix

```
aussieCrab <- read.csv("C:/Users/Gustav/Documents/Machine-Learning/Lab 3/australian-crabs.csv", sep=","
library(ggplot2)
ggplot(aussieCrab, aes(y=CL, x=RW)) + geom_point(aes(color=sex), size=3)

# Creates separte data sets for males and females
male_data <- subset(aussieCrab, sex=="Male")[5:6]
female_data <- subset(aussieCrab, sex=="Female")[5:6]

# Calculate the means and covariances
male_vec <- c(mean(male_data[,1]), mean(male_data[,2]))
female_vec <- c(mean(female_data[,1]), mean(female_data[,2]))
```

```r
male_cov <- cov(male_data)
female_cov <- cov(female_data)
covHat <- matrix(100/200 * male_cov + 100/200 * female_cov, ncol=2)

# prop priors
prior_male <- 100/200
prior_female <- 100/200

# The discriminant function

W_male <- (as.matrix(aussieCrab[, 5:6]))%*% solve(covHat) %*% as.matrix(male_vec)
Wo_male <-  (0.5 *  t(as.matrix(male_vec)) %*% solve(covHat) %*% as.matrix(male_vec) +
  log(prior_male))
disc_male <- W_male - as.vector(Wo_male)

W_female <- (as.matrix(aussieCrab[, 5:6]))%*% solve(covHat)%*% as.matrix(female_vec)
Wo_female <- (0.5 *  t(as.matrix(female_vec)) %*% solve(covHat) %*% as.matrix(female_vec) +
   log(prior_female))
disc_female <-W_female - as.vector(Wo_female)

# For loop that uses the obtained decision boundary
classif <- 0
for(i in 1:200){
  if(disc_male[i] > disc_female[i]){
    classif[i] = "male"
  }else{
    classif[i] = "female"
  }
}

lda_class <- data.frame(aussieCrab, classif)

ggplot(lda_class, aes(y=CL, x=RW)) +
  geom_point(aes(color=classif,shape=sex), size=4) +
  geom_abline(intercept = -5.06, slope=2.91, colour="red")

table(actual=aussieCrab$sex, predict=classif)


logi_class <- glm(sex~RW+CL,data=aussieCrab, family=binomial())
pred_logi <- data.frame(aussieCrab$RW,aussieCrab$CL,aussieCrab$sex , predict(logi_class, type="response

class <- rep(0,200)
pred <- cbind(round(pred_logi[,4]), class)
for(i in 1:200){
    if(pred[i] == 1){
        pred[i,2] <- "Male"
    }else{
        pred[i,2] <- "Female"
    }
}
pred_logi$pred <- pred[,2]
```

```
ggplot(pred_logi, aes(y=aussieCrab.CL, x=aussieCrab.RW)) +
  geom_point(aes(color=pred,shape=aussieCrab.sex), size=4) +
  geom_abline(intercept = -2.94, slope=2.713, colour="red")
data <- read.csv("C:/Users/Gustav/Documents/Machine-Learning/Lab 3/creditscoring.csv", sep=";", header =

n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=data[id,]
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
valid=data[id2,]
id3=setdiff(id1,id2)
test=data[id3,]

library(tree)
tree.dev <- tree(good_bad ~., train, split="deviance")
misclass <- function(data, measure, ...){
    set.seed(12345)
    x <- table(data$good_bad, predict(measure, newdata=data,...))
    n <- sum(x)
    return(1 - sum(diag(x))/n)
}

misclass(train, tree.dev, type="class")
misclass(test, tree.dev, type="class")


tree.gini <- tree(good_bad ~., train, split="gini")
misclass(train, tree.gini, type="class")
misclass(test, tree.gini, type="class")
train.score <- NULL
valid.score <- NULL

for(i in 2:15){
    pruned = prune.tree(tree.dev, best=i)
    pred = predict(pruned, newdata=valid, type="tree")
    train.score[i] = deviance(pruned)
    valid.score[i] = deviance(pred)
}
plot(2:15, train.score[2:15], type="b", ylim=c(0,560))
points(2:15, valid.score[2:15], type="b", col="purple")
legend("bottomleft",c("training","validation"),col=c("black","purple"), lwd=c(1,1))

finalTree <- prune.tree(tree.dev, best=4)
# The variables used
# Model tested on test data
NewFit <- predict(finalTree, newdata=test, type="class")
plot(finalTree)
text(finalTree, pretty=0)
NewFitTable <- table(test$good_bad, NewFit)
library(e1071)
```

```r
fitBayes<-naiveBayes(good_bad~., data=train)

YfitBayesTrain<-predict(fitBayes, newdata=train)
YfitBayesTrain <- table(YfitBayesTrain,train$good_bad)
YfitBayesTrain
YfitBayesTest<-predict(fitBayes, newdata=test)
YfitBayesTest <- table(YfitBayesTest,test$good_bad)
YfitBayesTest
# Repeating 2.4 but with a defined loss matrix.
raws <- data.frame(predict(fitBayes, newdata=train, type="raw"))

preds <- 0
for (i in 1:500){
  if((raws[i,1]/raws[i,2]) > 0.1){
    preds[i] = "bad"
  }else{
    preds[i] ="good"
  }
}

table(train$good_bad, preds)


rawTest <- predict(fitBayes, newdata=test, type="raw")
preds <- 0
for (i in 1:250){
  if((rawTest[i,1]/rawTest[i,2]) > 0.1){
    preds[i] = "bad"
  }else{
    preds[i] ="good"
  }
}

table(test$good_bad, preds)

##
```