log thema 9

Margriet van der Molen

2023-09-13

```r
knitr::opts_chunk$set(echo = TRUE)
library(pander)
library(ggplot2)
library(cowplot)
library(GGally)
```

Week 1 / 2

In the first week of this project, there will be two key things that need to be done. One, find an appropriate data set and formulate a research question based on the found set. Two, start with an exploratory data analysis.

The data set

The data that will be used is about penguins, and was originally used in a paper about ecological sexual dimorphism where they examined if environmental variability is associated with differences in the foraging niches of the male and female penguins:

Gorman KB, Williams TD, Fraser WR (2014) Ecological Sexual Dimorphism and Environmental Variability within a Community of Antarctic Penguins (Genus Pygoscelis). PLoS ONE 9(3): e90081. doi:10.1371/journal.pone.0090081.

It contains information about the location, physical characteristics, sample data, and data about the egg, of 345 penguins on Antartica. While all penguins were from the Anvers region, they were not all from the same island. All the penguins were nesting and blood samples, measurements, and molecular sexing, was all done when they were at the one-egg stage.

The delta C-13 and N-15 values were found using an elemental analyzer interfaced with an isotope ratio mass spectrometer at the Stable Isotope Facility, University of California, and calculated using the following equation:

$$\delta N15orC13 = (\frac{\delta R_{sample}}{\delta R_{standard}} - 1) * 1000$$

Here the R_sample is the ratio of found C-13 to C - 12 or N-15 to N-14 in the sample. The R_standard is the standard ratio for international standards (Vienna PeeDee Belemnite for carbon, and atmospheric N2 (Air) for nitrogen).

The research question

During this project, we'll focus on answering the question:

"How accurately can a machine learning model predict the sex of a penguin when given the measurements of some physical attributes"

Data processing

First, we read the .csv and create a data frame called ‘penguin.data’ from said file. Let’s take a look at the first few instances of the data.

```r
penguin.data <- read.csv("./data/penguins_lter.csv", header = T, na.strings = "NA")

pander(head(penguin.data))
```

Table 1: Table continues below

studyName	Sample.Number	Species	Region	Island
PAL0708	1	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen
PAL0708	2	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen
PAL0708	3	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen
PAL0708	4	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen
PAL0708	5	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen
PAL0708	6	Adelie Penguin (Pygoscelis adeliae)	Anvers	Torgersen

Table 2: Table continues below

Stage	Individual.ID	Clutch.Completion	Date.Egg
Adult, 1 Egg Stage	N1A1	Yes	11/11/07
Adult, 1 Egg Stage	N1A2	Yes	11/11/07
Adult, 1 Egg Stage	N2A1	Yes	11/16/07
Adult, 1 Egg Stage	N2A2	Yes	11/16/07
Adult, 1 Egg Stage	N3A1	Yes	11/16/07
Adult, 1 Egg Stage	N3A2	Yes	11/16/07

Table 3: Table continues below

Culmen.Length..mm.	Culmen.Depth..mm.	Flipper.Length..mm.	Body.Mass..g.
39.1	18.7	181	3750
39.5	17.4	186	3800
40.3	18	195	3250
NA	NA	NA	NA
36.7	19.3	193	3450
39.3	20.6	190	3650

Sex	Delta.15.N..o.oo.	Delta.13.C..o.oo.	Comments
MALE	NA	NA	Not enough blood for isotopes.
FEMALE	8.95	-24.69	
FEMALE	8.368	-25.33	
	NA	NA	Adult not sampled.
FEMALE	8.767	-25.32	
MALE	8.665	-25.3	

We see that we have some NA’s and some blank spaces, so not all missing values were properly labelled.

To clarify what is in the file we’ve just processed, we add a table with the name of the attribute and a description of it.

```r
info.data <- read.csv("info_data.csv")

pander(info.data)
```

Attribute	Description
studyName	Abbreviation of the studyname
Sample Number	ID for the blood sample that was taken
Species	Species of penguin (Common name followed by latin name)
Region	Which region the penguin was found
Island	Which island the penguin was found
Stage	Stage in which the penguin lives
Individual ID	ID for the penguin pairs
Clutch Completion	Whether or not the nest was completed
Date Egg	Date at which the egg is observed
Culmen Length (mm)	Length of the culmen in millimeters
Culmen Depth (mm)	Depth of the culmen in millimeters
Flipper Length (mm)	Length of the flipper in millimeters
Body Mass (g)	Weight of the penguin in grams
Sex	Sex of the penguin
Delta 15 N (0/00)	isotopic nitrogen found in blood in mille
Delta 13 C (0/00)	isotopic carbon found in blood in mille
Comments	additional comments on the penguins

Here, we can note that the comments section of the data won't be of great use to our further exploration of the data therefore we remove this column. We previously noted that sometimes there's a blank space, to prevent errors we'll be turning those into NA's. The stage attribute is the same for every penguin, it will give us little relevant information, and thus it is removed.

```r
penguin.data[penguin.data == "" | penguin.data == "."] <- NA # Change empty spaces and dots into NA's

penguin.data <- penguin.data[,-17] # Remove comments from data table
penguin.data <- penguin.data[,-6] # Remove stage attribute from data table
```

The column speaking of the sex of a penguin is of the character data type currently because this is actually a categorical variable, we change it into a factor. We also remove the NA's here because this is the label we'll be training our model on. Since the model will be supervised, it will need this label for training. The species column is also of the character data type, it contains the common name followed by the latin name this create quite a long label. We change species into a factor and just use the common name for the sake of clarity.

```r
# Change sex into a factor
penguin.data$Sex <- as.factor(penguin.data$Sex)
#Remove missing values from the column which describes the sex of the penguins
penguin.data <- penguin.data[!is.na(penguin.data$Sex),]

# Change species into a factor
penguin.data$Species <- as.factor(penguin.data$Species)
# Changes the names of the factor from full latin name with common name to just the common name
levels(penguin.data$Species) <- c("Adelie","Chinstrap","Gentoo")
```

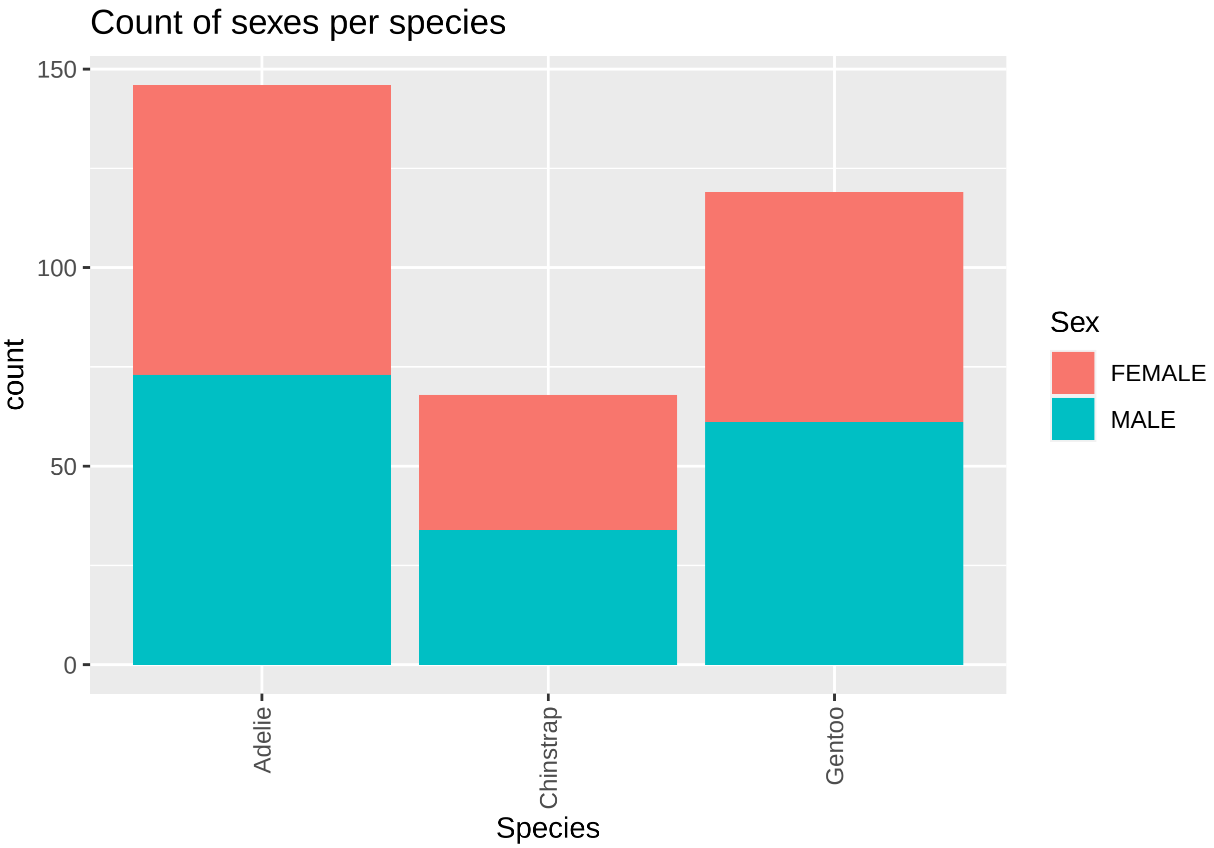
Due to the fact we'll want to do some pairwise t tests later on, we'll create a factor with species and sex in one table. We will remove this column later.

```r
penguin.data$Species_Sex <- paste(penguin.data$Species, penguin.data$Sex, sep = "_")
```

Data exploration univariate

With the data sufficiently processed for now, we continue with exploring the data to see if we can find further issues or even start to see groups start to form. Let us start with the species, and what number we have per species, and see if the sexes are equally divided.

```
ggplot(penguin.data, aes(Species, fill = Sex)) + geom_bar() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + ggtitle("Count of sexes per species")
```

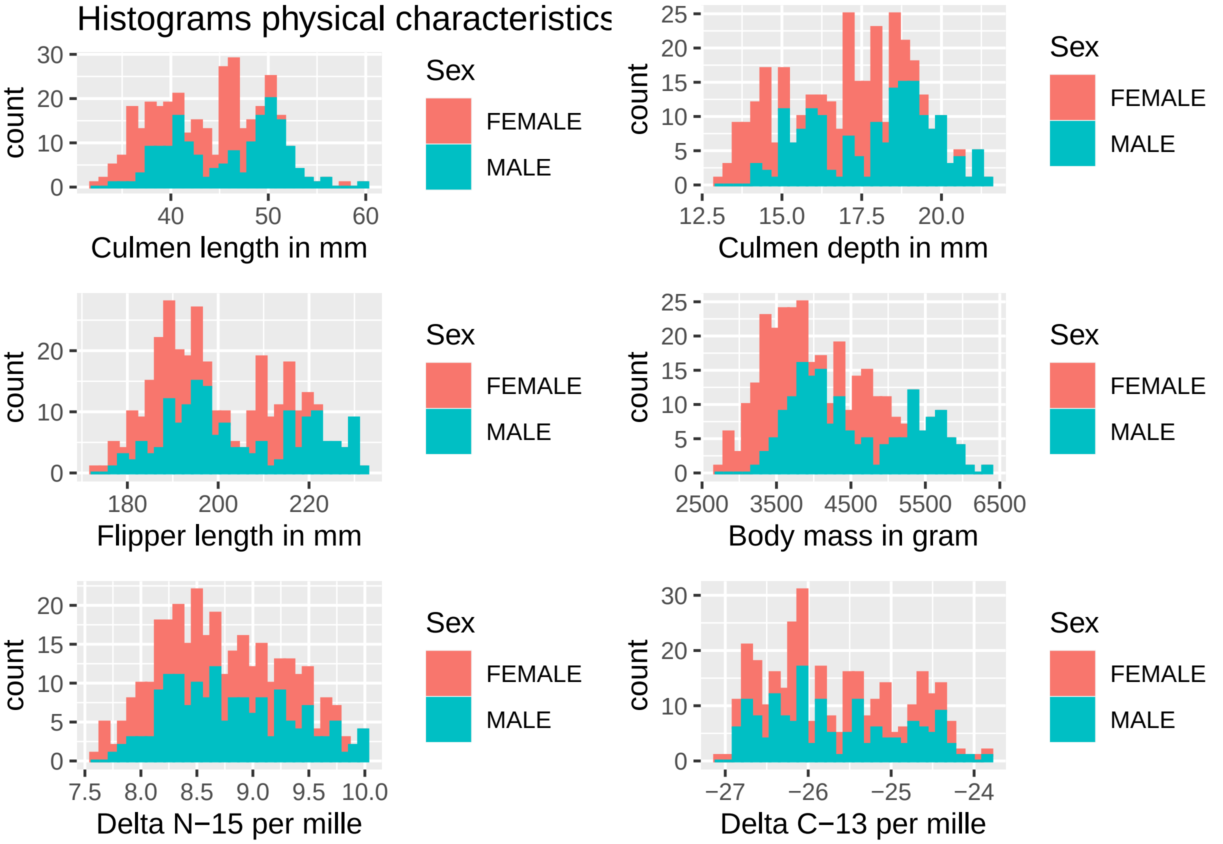


In the barplot above we can clearly see that we don't have that many Chinstrap Penguins. The study from which the data was gathered has an explanation for this, there were simply not enough chinstap penguins that were breeding in the rookeries on the island where they studied. The amount of female and male penguins observed is relatively equal to each other, no skewing there.

To look into how our data is distributed, we'll look into some histograms.

```
hist.depth <- ggplot(penguin.data, aes(x=Culmen.Length..mm., color=Sex, fill = Sex))+ geom_histogram() +

hist.length <- ggplot(penguin.data, aes(x=Culmen.Depth..mm., color=Sex, fill = Sex)) + geom_histogram()
  xlab("Culmen depth in mm")
hist.flip <- ggplot(penguin.data, aes(x=Flipper.Length..mm., color=Sex, fill = Sex)) + geom_histogram()
  xlab("Flipper length in mm")
hist.mass <- ggplot(penguin.data, aes(x=Body.Mass..g., color=Sex, fill = Sex)) + geom_histogram() +
  xlab("Body mass in gram")
hist.n15 <- ggplot(penguin.data, aes(x=penguin.data$Delta.15.N..o.oo., color=Sex, fill = Sex))+
  geom_histogram() +   xlab("Delta N-15 per mille")

hist.c13 <- ggplot(penguin.data, aes(x=penguin.data$Delta.13.C..o.oo., color=Sex, fill = Sex))+
  geom_histogram() +   xlab("Delta C-13 per mille")
```

```
plot_grid(hist.depth, hist.length, hist.flip, hist.mass, hist.n15, hist.c13, label_size = 12, ncol = 2)
```

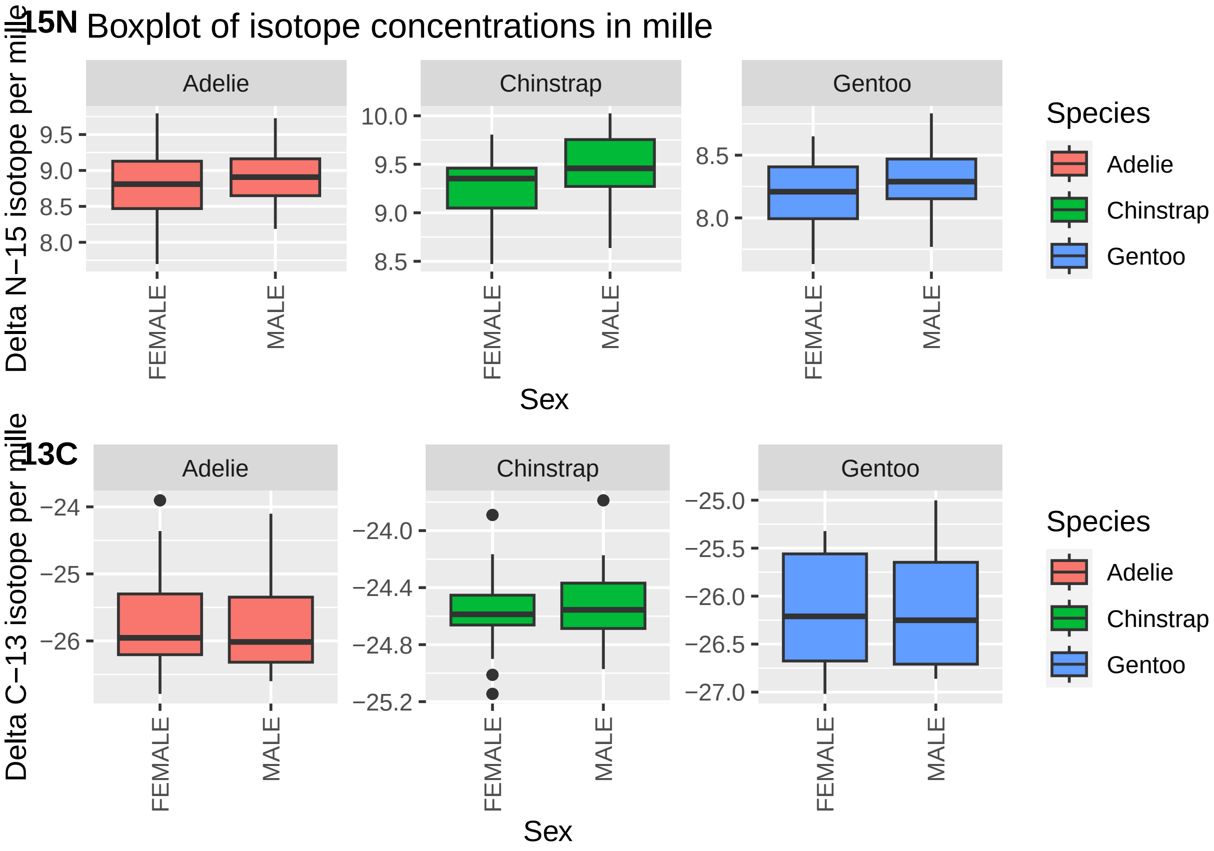


In the plot 'histograms of physical characteristics' we can see that the data has 3 peaks in the culmen data and two clear peaks in flipper length and body mass. This can be explained by the fact that we have three species. These species may have wildly varying culmens. To get a clearer look at what may be going on, we can make some boxplots and divide the data into species first and than the sex. This would allow us to see where the averages are, and whether or not this would cause these peaks.

Looking at the delta of the isotopes, the delta N-15 seems to have a vague bell curve but it does not suffer from the several peaks we see in other characteristics. Meanwhile the delta C-13 seems to have a slight skew to the left.

To get a better view on the distribution when taking species into account and to look over potential outliers, we create a few boxplots with the characteristics of our interests.

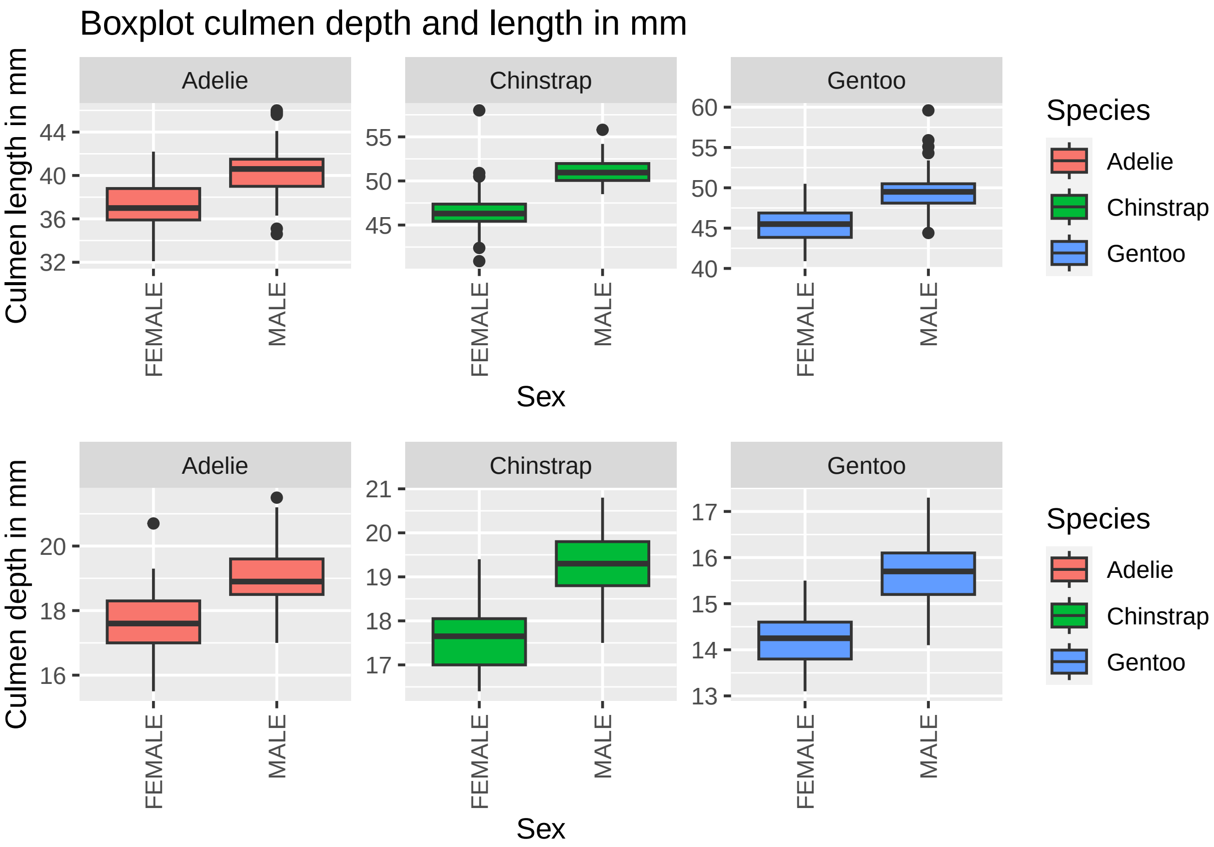
```
plot.15N <- ggplot(penguin.data, aes(x=Sex, y=Delta.15.N..o.oo., fill=Species)) +
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Delta N-15 isotope per mille")  +
  ggtitle("Boxplot of isotope concentrations in mille")

plot.13C <- ggplot(penguin.data, aes(x=Sex, y=Delta.13.C..o.oo., fill=Species)) +
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Delta C-13 isotope per mille")

plot_grid(plot.15N, plot.13C, labels = c('15N', '13C'), label_size = 12, ncol = 1)
```



In the plot 'Boxplot of isotope concentrations in mille' we can see that the different isotopes values don't look all that different between the sexes of the difference species. Of course, looks can be deceiving and we will look into this further later. There are a few outliers but nothing too extreme.

Pressing on, we move on to the physical attributes of the penguin.

```
plot.cl.length <- ggplot(penguin.data, aes(x=Sex, y=Culmen.Length..mm., fill=Species)) +
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Culmen length in mm") +
  ggtitle("Boxplot culmen depth and length in mm")

plot.cl.depth <- ggplot(penguin.data, aes(x=Sex, y=penguin.data$Culmen.Depth..mm., fill=Species)) +
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Culmen depth in mm")

plot_grid(plot.cl.length, plot.cl.depth, ncol = 1)
```



Looking at the 'Boxplot Culmen Depth and Length in mm' we can also see that for the Chinstrap and Gentoo penguins culmen depth and length has quite the difference between female and male. The Adelie penguins seem to have less of a difference. There's, again, a few outliers. The culmen length of male Gentoo penguins in particular is quite a diverse range. To see if these differences are significant, we'll perform a pairwise t test. Take these results with a grain of salt, as the distribution might is not entirely normal.

```
pairwise.t.test(penguin.data$Culmen.Length..mm.,
                penguin.data$Species_Sex,p.adjust.method = "bon", pool.sd = F)
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  penguin.data$Culmen.Length..mm. and penguin.data$Species_Sex
##
##                  Adelie_FEMALE Adelie_MALE Chinstrap_FEMALE Chinstrap_MALE
## Adelie_MALE      7.2e-14       -           -                -
## Chinstrap_FEMALE < 2e-16       6.7e-13     -                -
## Chinstrap_MALE   < 2e-16       < 2e-16     1.3e-08          -
## Gentoo_FEMALE    < 2e-16       < 2e-16     1.00000          < 2e-16
## Gentoo_MALE      < 2e-16       < 2e-16     0.00039          0.00576
##                  Gentoo_FEMALE
## Adelie_MALE      -
## Chinstrap_FEMALE -
## Chinstrap_MALE   -
## Gentoo_FEMALE    -
## Gentoo_MALE      2.0e-13
##
## P value adjustment method: bonferroni
```

Now we can see that in the species the female and male penguins all differ significantly in terms of culmen length. Between species, Gentoo females and Chinstrap females do not differ significantly. We'll run the same test for culmen depth.

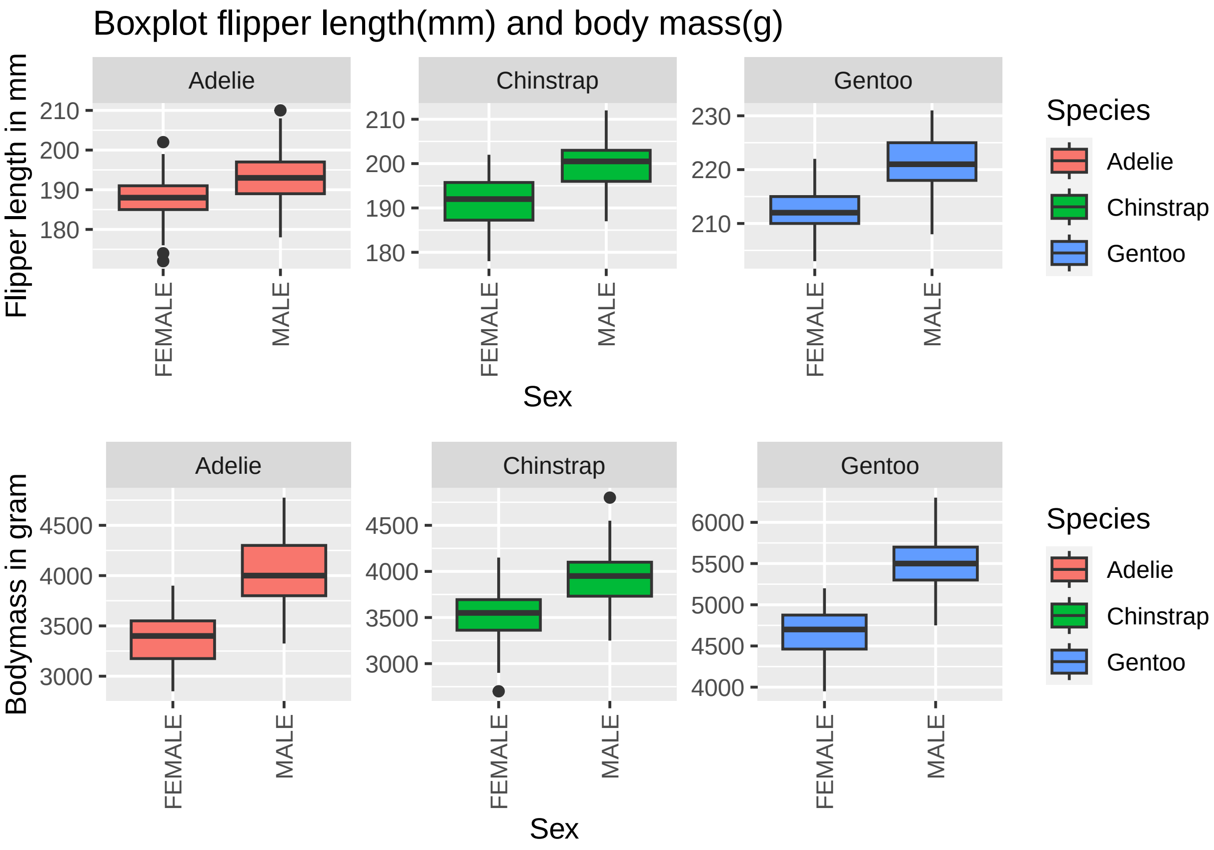
```
pairwise.t.test(penguin.data$Culmen.Depth..mm.,
                penguin.data$Species_Sex, p.adjust.method = "bon", pool.sd = F)
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  penguin.data$Culmen.Depth..mm. and penguin.data$Species_Sex
##
##                  Adelie_FEMALE Adelie_MALE Chinstrap_FEMALE Chinstrap_MALE
## Adelie_MALE      2.9e-14       -           -                -
## Chinstrap_FEMALE 1             2.9e-11     -                -
## Chinstrap_MALE   1.3e-13       1           9.9e-12          -
## Gentoo_FEMALE    < 2e-16       < 2e-16     < 2e-16          < 2e-16
## Gentoo_MALE      < 2e-16       < 2e-16     5.4e-16          < 2e-16
##                  Gentoo_FEMALE
## Adelie_MALE      -
## Chinstrap_FEMALE -
## Chinstrap_MALE   -
## Gentoo_FEMALE    -
## Gentoo_MALE      < 2e-16
##
## P value adjustment method: bonferroni
```

Here,again, when looking at the same species, the differences between male and female penguins are all significant. Adelie females are not significantly different than Chinstrap females and Adelie males are not significantly different than Chinstrap males.

We'll go through this procedure again but with flipper length and body mass. First we create a boxplot.

```
plot.flip.length <- ggplot(penguin.data, aes(x=Sex, y=penguin.data$Flipper.Length..mm., fill=Species))
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Flipper length in mm") +
  ggtitle("Boxplot flipper length(mm) and body mass(g)")

plot.body.mass <- ggplot(penguin.data, aes(x=Sex, y=penguin.data$Body.Mass..g., fill=Species)) +
    geom_boxplot() + facet_wrap(~Species, scale="free") +
  scale_x_discrete(guide = guide_axis(angle = 90))+ ylab("Bodymass in gram")

plot_grid(plot.flip.length, plot.body.mass, ncol = 1)
```



When looking at 'Boxplot flipper length(mm) and body mass(g)' we can see that Chinstrap and Gentoo penguins are more prone to sexual dimorphism than the Adelie penguins, though even the Adelie penguins appear to have a difference in body mass. To further test this, let's run another pairwise t test with flipper length first followed by body mass.

```r
pairwise.t.test(penguin.data$Flipper.Length..mm.,
                penguin.data$Species_Sex, p.adjust.method = "bon", pool.sd = F)
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  penguin.data$Flipper.Length..mm. and penguin.data$Species_Sex
##
##                  Adelie_FEMALE Adelie_MALE Chinstrap_FEMALE Chinstrap_MALE
## Adelie_MALE      0.00017       -           -                -
## Chinstrap_FEMALE 0.02202       1.00000     -                -
## Chinstrap_MALE   3.2e-13       2.2e-06     3.8e-06          -
## Gentoo_FEMALE    < 2e-16       < 2e-16     < 2e-16          5.6e-14
## Gentoo_MALE      < 2e-16       < 2e-16     < 2e-16          < 2e-16
##                  Gentoo_FEMALE
## Adelie_MALE      -
## Chinstrap_FEMALE -
## Chinstrap_MALE   -
## Gentoo_FEMALE    -
## Gentoo_MALE      1.1e-15
##
## P value adjustment method: bonferroni
```

In terms of flipper length, within the species, the sexes all significantly differ. Adelie males are not significantly different when compared to chinstrap females. Moving on to body mass.

```
pairwise.t.test(penguin.data$Body.Mass..g.,
                penguin.data$Species_Sex, p.adjust.method = "bon", pool.sd = F)
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  penguin.data$Body.Mass..g. and penguin.data$Species_Sex
##
##                  Adelie_FEMALE Adelie_MALE Chinstrap_FEMALE Chinstrap_MALE
## Adelie_MALE      < 2e-16       -           -                -
## Chinstrap_FEMALE 0.13          8.7e-11     -                -
## Chinstrap_MALE   1.2e-09       1.00        3.4e-05          -
## Gentoo_FEMALE    < 2e-16       < 2e-16     < 2e-16          2.6e-13
## Gentoo_MALE      < 2e-16       < 2e-16     < 2e-16          < 2e-16
##                  Gentoo_FEMALE
## Adelie_MALE      -
## Chinstrap_FEMALE -
## Chinstrap_MALE   -
## Gentoo_FEMALE    -
## Gentoo_MALE      < 2e-16
##
## P value adjustment method: bonferroni
```

Again, within species, the body mass differs significantly from male to female penguins. Adelie females do not differ significantly from Chinstrap females, and Adelie males do not differ from their Chinstrap counterparts.

Later on a decision was made to use a two-way Anova rather than the pairwise t-test. This one was in the following manner.

```
# Apply a custom lambda function on the columns flipper length and body mass. This function returns
# the p-values of the anova.
aov.results <- apply(penguin.data[11:12], 2, FUN = function(x) summary(aov(x~penguin.data$Species*pengu

rownames(aov.results) <- c("P value two-paired Anova Sex",
                           "P value two-paired Anova Species", "P value Species * Sex")
pander(aov.results)
```

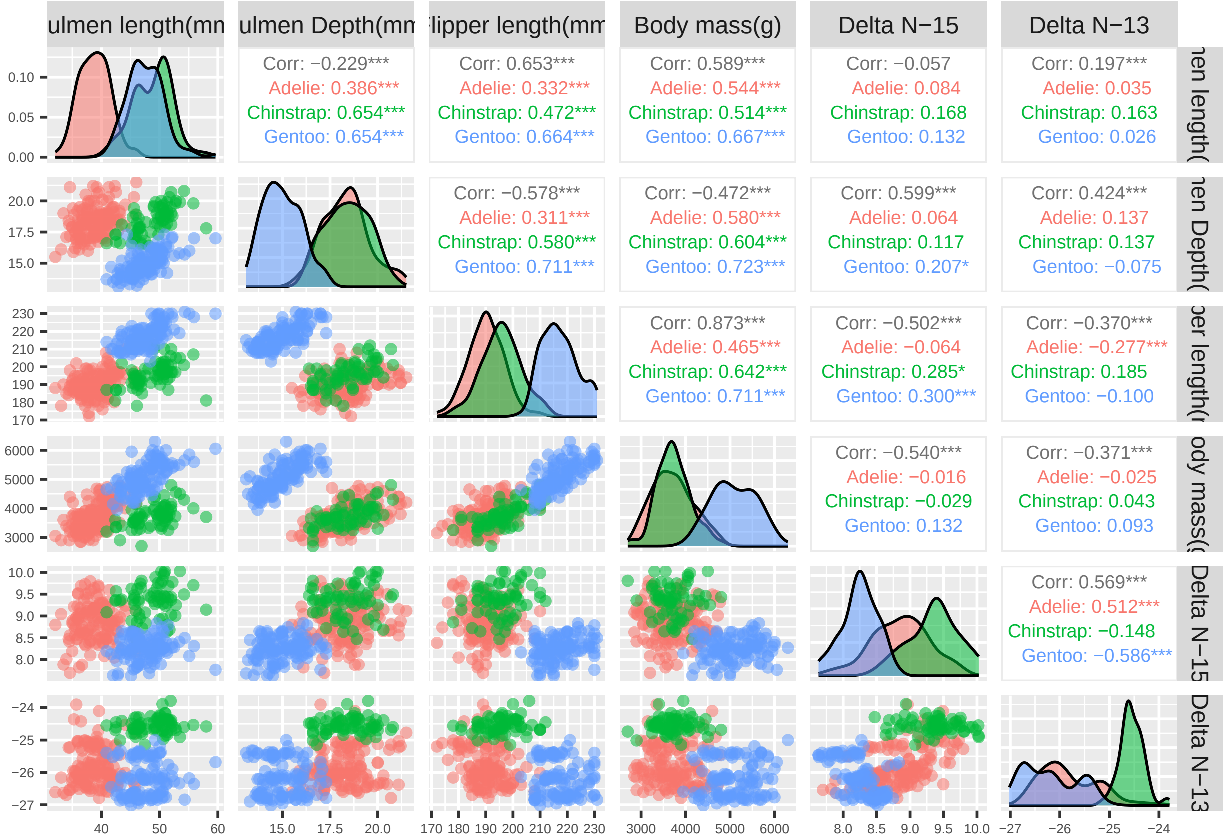
	Flipper.Length..mm.	Body.Mass..g.
P value two-paired Anova Sex	6.278e-126	1.54e-123
P value two-paired Anova Species	2.461e-24	1.902e-57
P value Species * Sex	0.006314	0.0001973

From the test, it is shown that both flipper length and body mass, the sexes all significantly differ as do the species. In the bottom row, where the p value from the interaction between species and sex is, indicates this interaction has a significant effect on both flipper length and body mass.

Data exploration bivariate

Delving deeper into correlation, we create several scatter plots using the different attributes. Earlier we stated that some species are more prone to sexual dimorphism, this means if we can find attributes that help us differentiate species and sexes, we may have quite a fitting attribute on our hands. So we plot both the scatterplot for the species and for the sexes. By plotting both we also hope to clear up any confusion regarding the peaks we saw in our histograms.

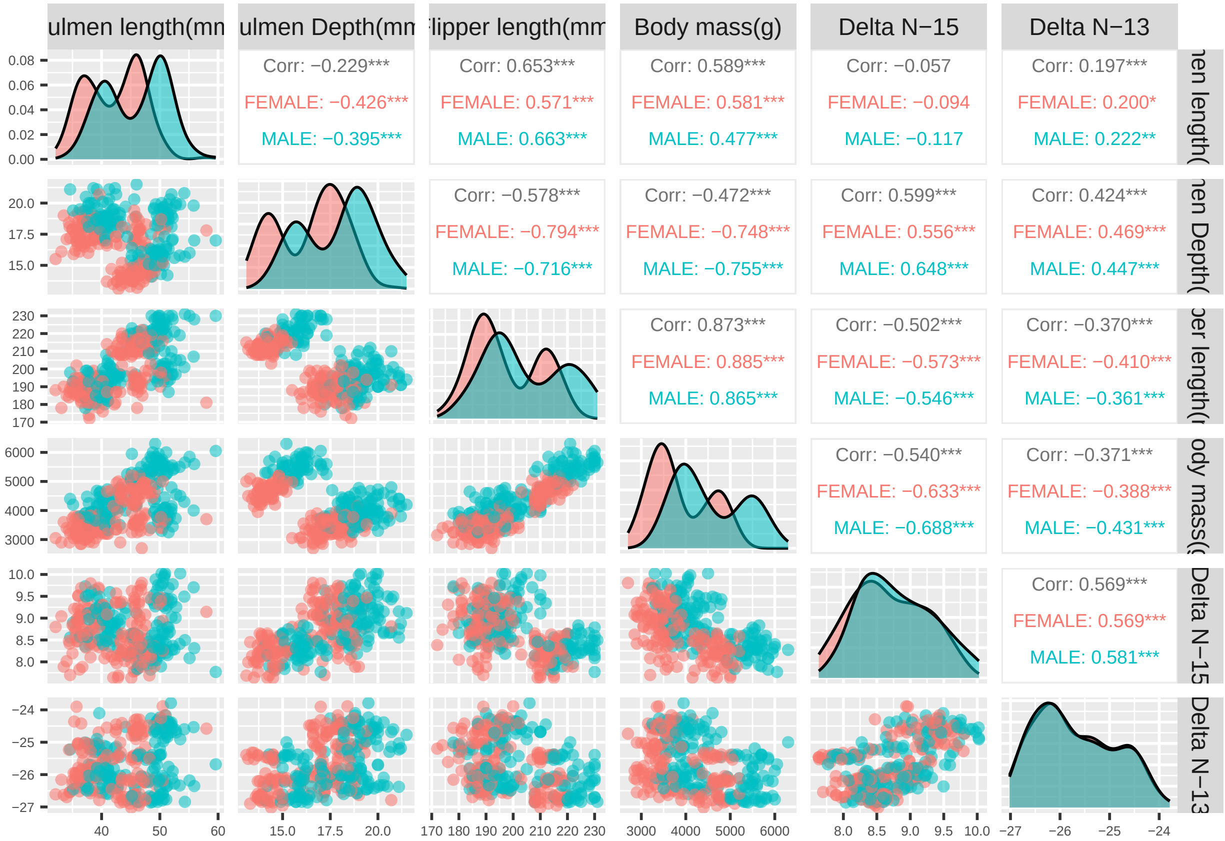
```r
ggpairs(penguin.data[c(9:12, 14, 15)],  columnLabels = c("Culmen length(mm)",
                                                     "Culmen Depth(mm)",
                                                     "Flipper length(mm)",
                                                     "Body mass(g)", "Delta N-15",
                                                     "Delta N-13"),
        ggplot2::aes(colour=penguin.data$Species, alpha = 0.7),
        upper = list(continuous = wrap("cor", size = 2.5)), progress = FALSE) +
  theme(axis.text = element_text(size = 5))
```



A scatterplot is used to get a rough idea of the correlation between the different attributes. While we created a lot of plots, some noteworthy findings here is the fact that culmen length or flipper length can aid us in differentiating the species. These attributes also have a notable correlation with on another as does body mass and culmen length. There's still quite a bit of overlap with our three species, the flipper length of Chinstraps and Adelies for example. However, our focus is on differentiating sexes so we continue.

We'll create the same kind of plot but instead of using species, we'll use the sex of the penguin to label.

```r
ggpairs(penguin.data[c(9:12, 14, 15)],  columnLabels = c("Culmen length(mm)",
                                                     "Culmen Depth(mm)",
                                                     "Flipper length(mm)",
                                                     "Body mass(g)", "Delta N-15",
                                                    "Delta N-13"),
      ggplot2::aes(colour=penguin.data$Sex, alpha = 0.7),
      upper = list(continuous = wrap("cor", size = 2.5)),
      progress = FALSE ) +
  theme(axis.text = element_text(size = 5))
```



Looking further into our data, when we take into account sex, we see that finding fitting attributes might be harder here as we have a lot of overlap going on. Culmen depth, culmen length, flipper length, or body mass seem to be the best candidates for an attribute that will give us the necessary information to be able to differentiate between the sexes.

In terms of correlation, body mass and flipper length share a strong positive correlation. No other attributes seem to be as correlated as these two.

Week 3

In week 3 we focus on cleaning the data so that we can use it in Weka, and getting started on the Results & Discussion section of the report.

So to get started, let's clean our data. Earlier we added an unique ID cause we thought we might need it, we do not and thus we remove it along with the study name, sample number, individual ID Since clutch completion and region is always the same answer, we also remove that. The date of our egg is not of interest to our research question and thus it is removed.

```
penguin.data <- penguin.data[,!names(penguin.data) %in% c("studyName", "Sample.Number",
                                                    "Region", "Individual.ID", "uniqueID",
                                                    "Clutch.Completion", "Date.Egg", "Species_Sex
```

```r
# Put the column sex at the end of the file
penguin.data <- penguin.data[,c(1:6, 8, 9 ,7)]
```

We see that we also still have some NA's in the delta N-15 and C-13 columns. In total, those are 17 NA's, so 8 penguins. The reasons for these NA is that the blood sample taken from these birds was simply insufficient, there wasn't enough to collect data from. To not skew the data, we've chosen to remove these penguins from our final file.

```r
sum(is.na(penguin.data[,8:9]))
```

```
## [1] 8
```

```r
penguin.data <- na.omit(penguin.data)
```

Now we create our final file.

```r
write.csv(penguin.data, "./data/cleaned_penguin_data.csv", row.names = F)
```

Week 4

In week 4 we start with machine learning, first we'll have to decide what is important to our dataset, and after such we'll start exploring what algorithms will be best fitted for the data.

Determining quality metrics

Accuracy will be the most important quality metric of this algorithm, and we can use it because our data is not skewed. We won't pay much mind to sensitivity and specificity, as we are dealing with male and female penguins. It is as bad to guess a female penguin when it is actually a male penguin as it is to guess a male penguin while it is actually a female penguin. They will be equally measured.

To avoid overfitting, we'll be using 10 fold cross validation.

Exploring ML algorithms

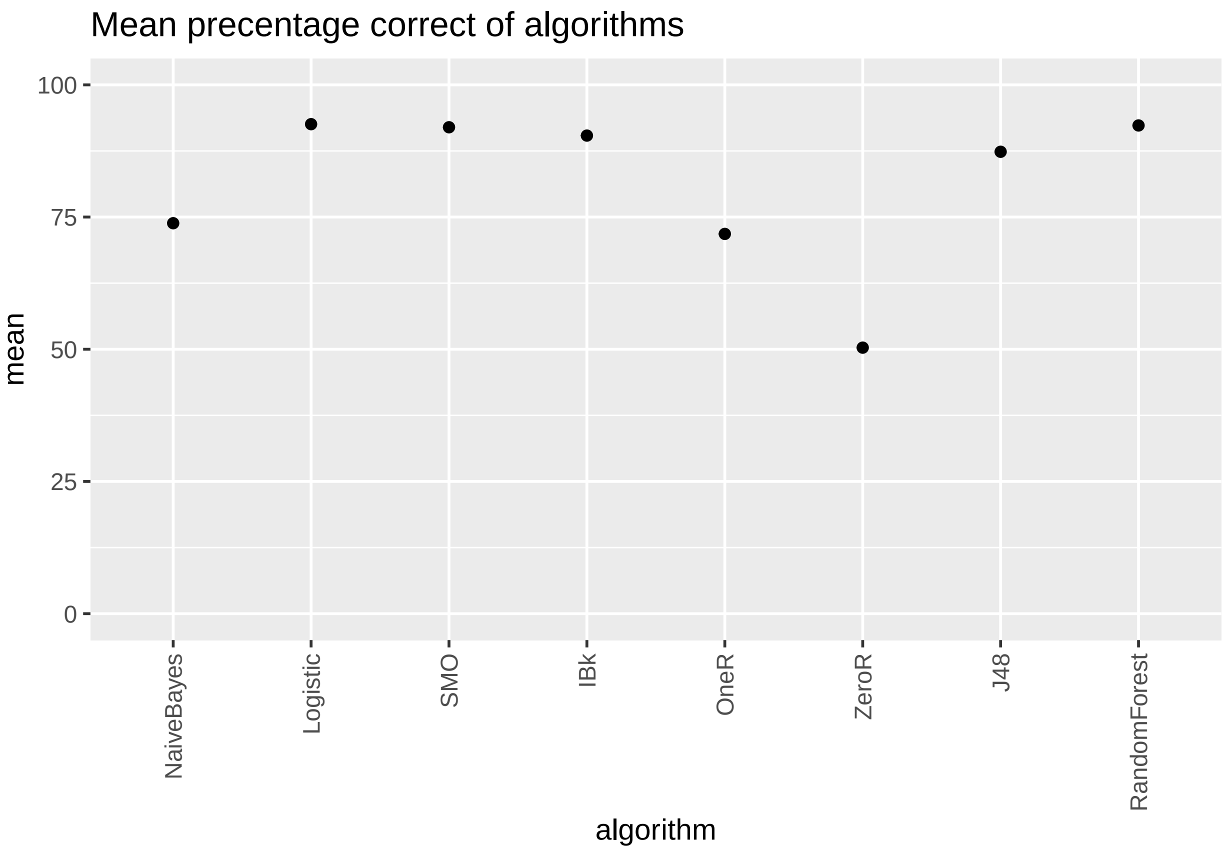
Keeping the quality metrics in mind, we'll run the Weka Experimenter with the following algorithms: - ZeroR - OneR - J48 - NaiveBayes - IBk - SMO - Logistics - RandomForest

All algorithms will be ran with their default parameters. This is just to get an idea of how well these algorithms perform. Later in this log, we'll try to optimize them.

```r
result.explorer <- read.csv("data/test.csv", header = T)

#Turn Key_Scheme into factor for easy plotting
result.explorer$Key_Scheme <- as.factor(result.explorer$Key_Scheme)
levels(result.explorer$Key_Scheme) <- c("NaiveBayes","Logistic",
                                        "SMO","IBk", "OneR","ZeroR","J48","RandomForest")
```

Because we are interested in what the most accurate algorithm is, we focus on percent_correct column.

```r
# Create dataframe with the means of the groups
percentage.correct <- setNames(aggregate(result.explorer$Percent_correct,
                                         list(result.explorer$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))

ggplot(percentage.correct, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 100)) +
  ggtitle("Mean precentage correct of algorithms")
```

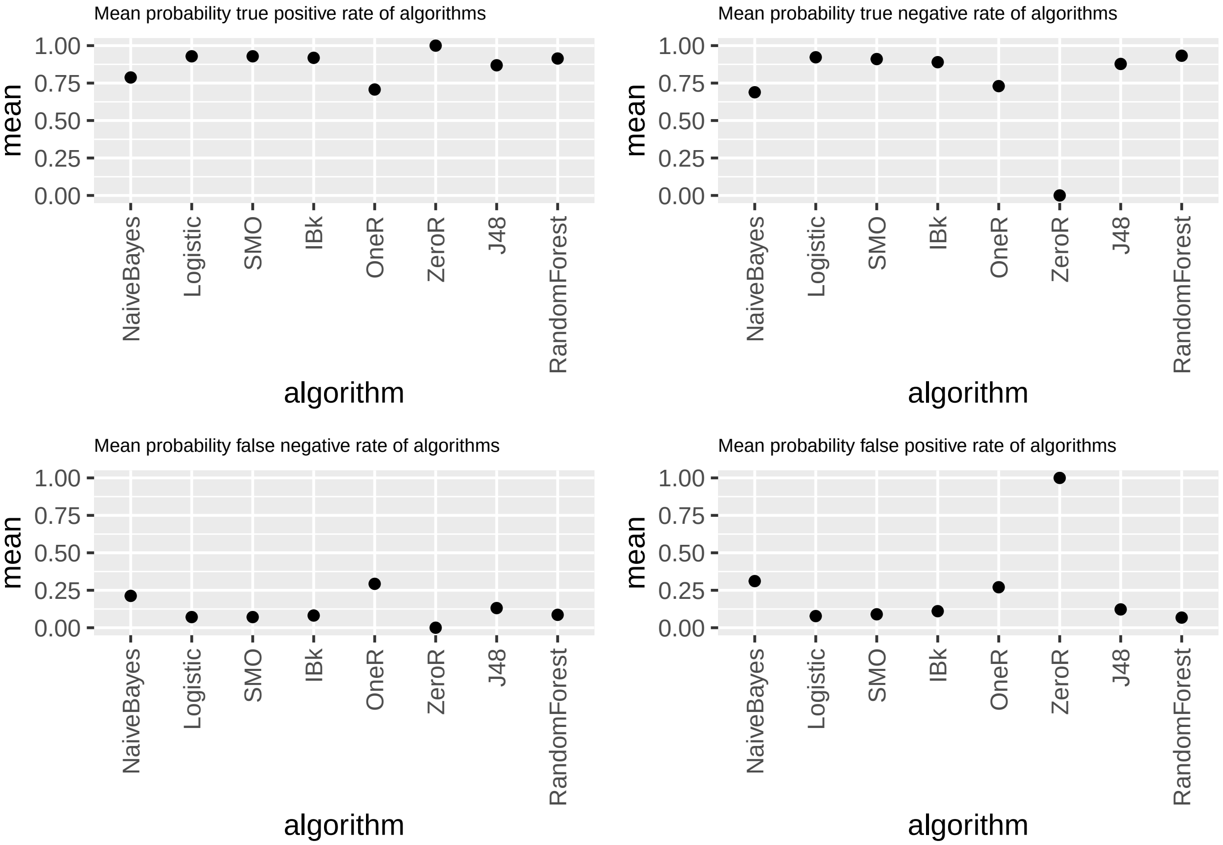


In the plot above we can see that the algorithms Logistic, SMO, IBk, and RandomForest all score rather well if we're just looking at.

Let's continue to look at some of the details of this accuracy. We'll look at the true positive rate, the true negative rate, false positive, and false negative rate for all the algorithms as well.

```r
#Again, creating the datasets necessary for plotting
prob.true.pos <- setNames(aggregate(result.explorer$True_positive_rate,
                                         list(result.explorer$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))

prob.true.neg <- setNames(aggregate(result.explorer$True_negative_rate,
                                         list(result.explorer$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))

prob.false.neg <- setNames(aggregate(result.explorer$False_negative_rate,
                                         list(result.explorer$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))

prob.false.pos <- setNames(aggregate(result.explorer$False_positive_rate,
                                         list(result.explorer$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))
# Creating the plot objects
true.pos <- ggplot(prob.true.pos, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 1)) +
  ggtitle("Mean probability true positive rate of algorithms")+
  theme(plot.title = element_text(size=7))
```

```r
true.neg <- ggplot(prob.true.neg, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 1)) +
  ggtitle("Mean probability true negative rate of algorithms")+
  theme(plot.title = element_text(size=7))

false.neg <- ggplot(prob.false.neg, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 1)) +
  ggtitle("Mean probability false negative rate of algorithms")+
  theme(plot.title = element_text(size=7))

false.pos <- ggplot(prob.false.pos, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 1)) +
  ggtitle("Mean probability false positive rate of algorithms")+
  theme(plot.title = element_text(size=7))
# Returning the grid with the made plots
plot_grid(true.pos, true.neg, false.neg, false.pos)
```


Mean probability true positive rate of algorithms

Mean probability true negative rate of algorithms

Mean probability false negative rate of algorithms

Mean probability false positive rate of algorithms

While there's no additional weighing going on, it is still nice to get a look at what the algorithms may find more difficult. For example, NaiveBayes seems to be quite adapt at getting the true positives but it struggles with the true negatives which results in a higher false positive rate. RandomForest, an algorithm that was quite accurate, struggles more with getting the true positives right which results in a -slightly- higher false negative rate.

Looking at the accuracy, we'll continue exploring 2 algorithms. RandomForest and the logistics algorithms. Starting with optimizing the parameters. We leave the amount of splitting going on alone but looking at the iteration depth might be useful from a performance standpoint. For that reason, we limit it between 1 and 100 and run the CVParameterSelection tool which tells us that 45 iterations is optimal here.

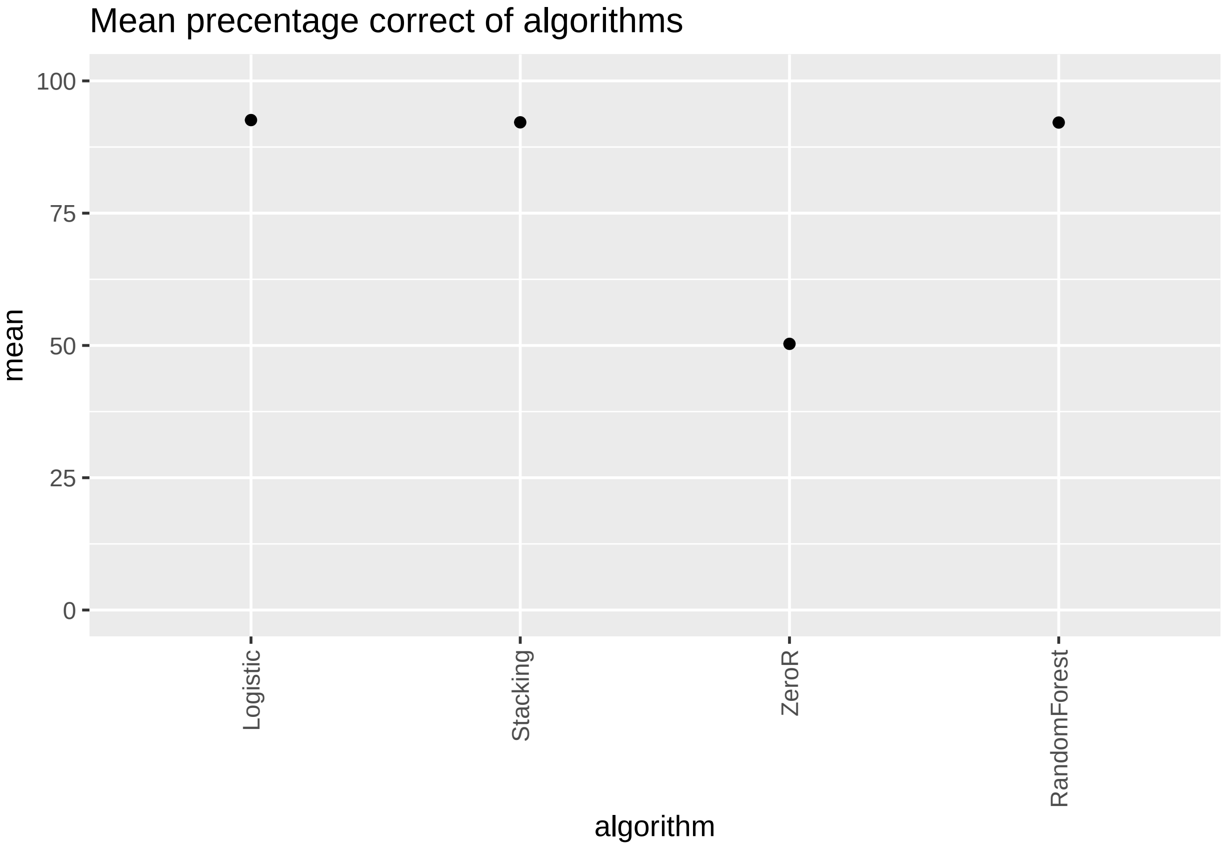
We do the same for the amount of iterations of the Logistic's algorithm which results in, again, 12 iterations being optimal.

While RandomForest is an ensemble technique, let's also attempt stacking with the 'weaker' algorithms. We'll take IbK, OneR, ZeroR, and a J48 tree and use another J48 tree as our meta learner. In terms of optimizing our parameters, the ibK tree runs with 11 neighbors and to prevent overfitting the first J48 tree runs with 10 in the minObj variable.

```
adjusted.results <- read.csv("data/adjusted_para_results.csv", header = T)

#Turn Key_Scheme into factor for easy plotting
adjusted.results$Key_Scheme <- as.factor(adjusted.results$Key_Scheme)
#Turn Key_Scheme into factor for easy plotting
levels(adjusted.results$Key_Scheme) <- c("Logistic","Stacking",
                                         "ZeroR","RandomForest")
```

Just like we did with the non-optimized settings, let's now do such with our 'optimized' models.

```
# Create dataframe with the means of the groups
percentage.correct.adjusted <- setNames(aggregate(adjusted.results$Percent_correct,
                                          list(adjusted.results$Key_Scheme),
                  FUN=mean), c("algorithm", "mean"))

ggplot(percentage.correct.adjusted, aes(algorithm, mean)) + geom_point() +
  scale_x_discrete(guide = guide_axis(angle = 90)) + coord_cartesian(ylim = c(0, 100)) +
  ggtitle("Mean precentage correct of algorithms")
```



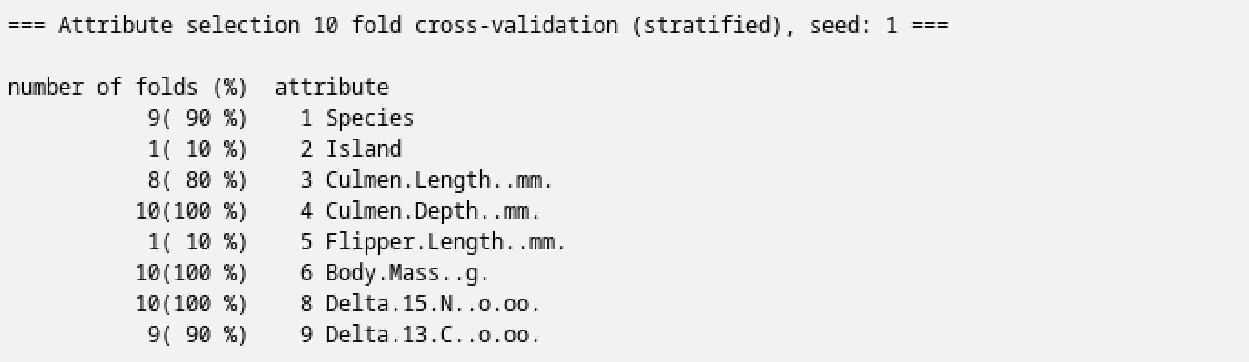
Stacking the simple algorithms together appears to have boosted their performance quite a bit as they are now catching up to Logistic and RandomForest. They are all still performing better than ZeroR, our point of comparison.

```r
#Create matrix with relevant values
data.comparison <- matrix(c(percentage.correct$mean[percentage.correct$algorithm == "RandomForest"], pe
#Add names to the table
colnames(data.comparison) <- c("RandomForest", "Logistic")
rownames(data.comparison) <- c("Non adjusted", "Adjusted")

#Show the table with relevant values
as.table(data.comparison)
```

```
##              RandomForest Logistic
## Non adjusted     92.32102 92.56534
## Adjusted         92.13163 92.59470
```

In the table above we determine that the Logistic algorithm does perform better with the iterations set at 12 but the RandomForest setting doesn't perform that much better with iterations set at 45, though the difference is quite small and the performance gained by this setting may make up for it.

We'll say that the Logistic algorithm with the iterations set at 45 is looking promising and we'll continue with this algorithm to use the WrapperSubsetEval on. There's one more thing to look at, the selection of attributes. For this, we use the WrapperSubsetEval tool and we select the BestSearch for the search method, and we use 10-fold cross validation. This results in the following table:



```
=== Attribute selection 10 fold cross-validation (stratified), seed: 1 ===

number of folds (%)  attribute
           9( 90 %)    1 Species
           1( 10 %)    2 Island
           8( 80 %)    3 Culmen.Length..mm.
          10(100 %)    4 Culmen.Depth..mm.
           1( 10 %)    5 Flipper.Length..mm.
          10(100 %)    6 Body.Mass..g.
          10(100 %)    8 Delta.15.N..o.oo.
           9( 90 %)    9 Delta.13.C..o.oo.
```

Figure 1: "A picture of the WrapperSubsetEval results"

Here we can see that culmen length, culmen depth, body mass, and delta 15-N were chosen in all of the subsets. Meanwhile island and flipper length were only chosen in 10% of the subsets. Species and delta 13 were both in 90% of the subsets. Let's run our experiment again, but we remove flipper length and island as attributes.

```r
#Remove Island and Flipper Length attributes
adjusted.penguin.data <- penguin.data[,!names(penguin.data) %in% c("Island","Flipper.Length..mm.")]
#Save the new csv file
write.csv(adjusted.penguin.data, "./data/adjusted_penguin_data.csv", row.names = F)
```

Now, we run the same experiment above with the adjusted dataset.

```r
final.results <- read.csv("data/final_results.csv", header = T)

#Turn Key_Scheme into factor for easy plotting
final.results$Key_Scheme <- as.factor(final.results$Key_Scheme)
#Adjust factor level names
levels(final.results$Key_Scheme) <- c("Logistic","Stacking",
                                      "ZeroR","RandomForest")
```

```r
#Add new column to the data frame that combines Key Scheme and Key Dataset
final.results$Key_Data_Scheme <- as.factor(paste(final.results$Key_Dataset, final.results$Key_Scheme, s
```

And lastly, we show the scores. Not in plot form this time, this is due to the fact that the improvement is so small that it isn't really visible in a plot

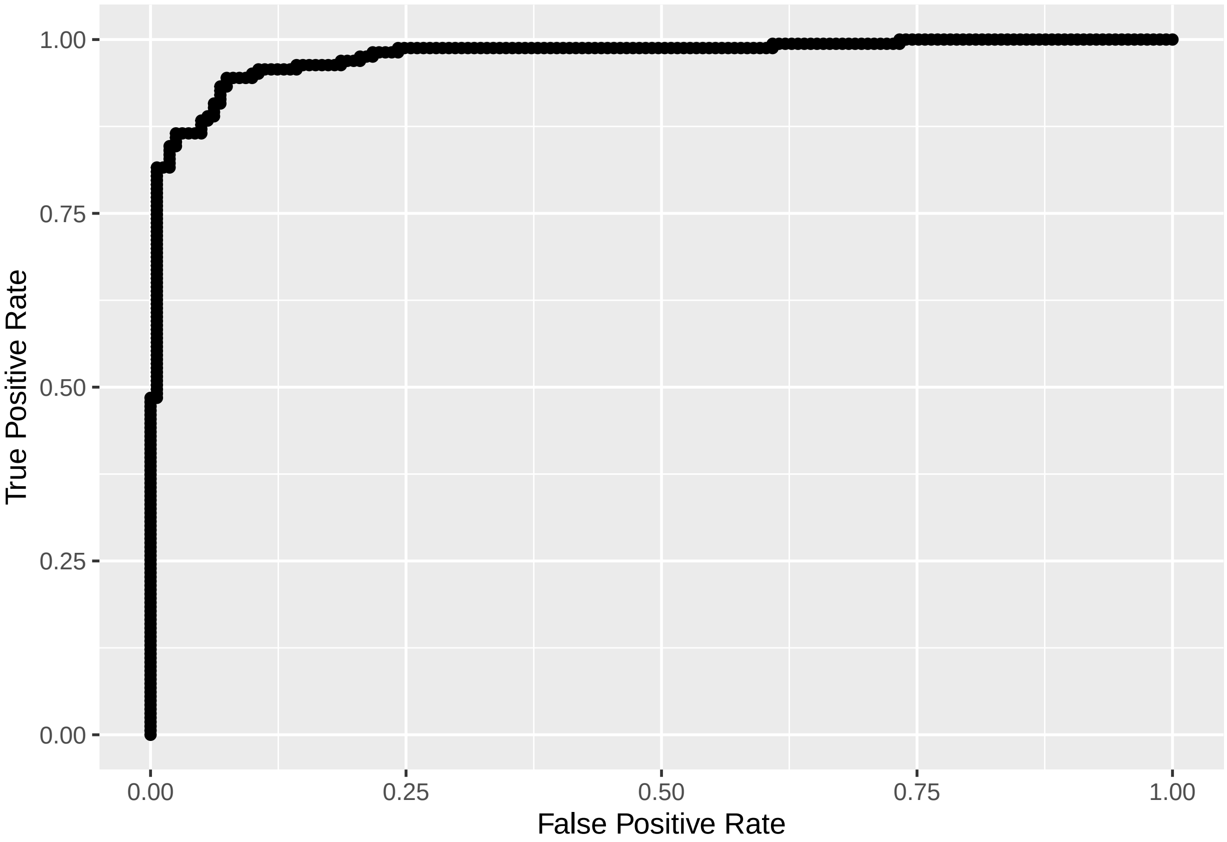
```r
# Create dataframe with the means of the groups
percentage.correct.final <- setNames(aggregate(final.results$Percent_correct,
                                        list(final.results$Key_Data_Scheme),
                  FUN=mean), c("algorithm", "mean"))

pander(percentage.correct.final)
```

algorithm	mean
adjusted_penguin_data_Logistic	92.84
adjusted_penguin_data_RandomForest	91.86
adjusted_penguin_data_Stacking	92.33
adjusted_penguin_data_ZeroR	50.3
cleaned_penguin_data_Logistic	92.59
cleaned_penguin_data_RandomForest	92.13
cleaned_penguin_data_Stacking	92.17
cleaned_penguin_data_ZeroR	50.3

Logistic, and surprisingly Stacking, both show an improvement. Due to the fact the tool was ran on just the Logistic it makes sense that Logistic is showing the most improvement. While the slight improvement isn't all that significant, what is important is that we can remove two attributes from the data to run our algorithm.

The Logistic algorithm remains in first place for choice algorithm due to its accuracy and relatively simple workings. Let's take a lot at the ROC of this algorithm. First, we run the explorer with the Logistic algorithm and its mentioned parameters with the adjusted data set, the female penguin are set to the positive value, and after we load it into R.

```r
ROC <- read.csv("./data/ROC_result.arff", header = F, sep = ",", comment.char = "@")

colnames(ROC) <- c("Instance_number","True Positives","False Negatives",
                   "False Positives","True Negatives",
                   "False Positive Rate","True Positive Rate",
                   "Precision numeric","Recall","Fallout",
                   "FMeasure","Sample Size","Lift","Threshold numeric")

ggplot(data = ROC, aes(y =`True Positive Rate`, x=`False Positive Rate`)) +
  geom_point()
```



The ROC plot above shows us the performance of our chosen algorithm. The area under the curve looks quite good, and while it is not a perfect score of 1, it is certainly no less than 0.5. In fact, if we quickly take a look at the actual score.

```
tapply(final.results$Area_under_ROC, final.results$Key_Data_Scheme, mean)[1]
```

```
## adjusted_penguin_data_Logistic
##                      0.9790786
```

We see that the area under the curve should 0.979 which looks to be about correct. With this we conclude that the Logistic algorithm will be the chosen algorithm for our research.