La Vuelta

Laura Cebollero Ruiz, Alexandre Rodríguez Garau

4th January, 2019

Introduction

In this project we intend to predict the duration of the different stages of the cycling race *La Vuelta* using the information provided in the file `Vuelta0.mtp`.

```
library("readxl")
library("knitr")
library("ggplot2")
library("reshape2")
library("MASS")
```

We have exported the data to an extension that R can read: `.csv`.

```
data<- read.csv("Vuelta01.csv", sep = ';', header = TRUE, dec=",")
```

Data exploration

To predict the length of the stages we will use a set that contains 14 explanatory variables and a response variable called **`ForecastedTime`**. Let's take a look at the summary of the variables:

```
kable(summary(data[,1:6]))
```

Time	Distance	HeightIncr	AccumIncr	portsE	ports1
Min. :171.6	Min. :111.0	Min. :-940.0	Min. : 190	Min. :0.0000	Min. :0.000
1st Qu.:258.1	1st Qu.:167.2	1st Qu.:-180.0	1st Qu.: 510	1st Qu.:0.0000	1st Qu.:0.000
Median :301.4	Median :196.0	Median : 70.0	Median :1350	Median :0.0000	Median :0.000
Mean :302.6	Mean :193.0	Mean : 227.9	Mean :1477	Mean :0.1524	Mean :0.581
3rd Qu.:345.8	3rd Qu.:219.5	3rd Qu.: 540.0	3rd Qu.:2300	3rd Qu.:0.0000	3rd Qu.:1.000
Max. :439.7	Max. :264.0	Max. :2310.0	Max. :4216	Max. :2.0000	Max. :3.000

```
kable(summary(data[,6:11]))
```

ports1	ports2	ports3	year	week	bef_mount
Min. :0.000	Min. :0.0000	Min. :0.0000	Min. :1.000	Min. :1.000	Min. :0.0000
1st Qu.:0.000	1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:2.000	1st Qu.:1.000	1st Qu.:0.0000
Median :0.000	Median :0.0000	Median :1.0000	Median :4.000	Median :2.000	Median :0.0000
Mean :0.581	Mean :0.4857	Mean :0.8476	Mean :3.619	Mean :2.048	Mean :0.2952
3rd Qu.:1.000	3rd Qu.:1.0000	3rd Qu.:2.0000	3rd Qu.:5.000	3rd Qu.:3.000	3rd Qu.:1.0000
Max. :3.000	Max. :3.0000	Max. :4.0000	Max. :6.000	Max. :3.000	Max. :1.0000

```
kable(summary(data[,12:16]))
```

aft_mount	bef_tt	aft_tt	last	ForecastedTime
Min. :0.0000	Min. :0.0000	Min. :0.0000	Min. :0.00000	Min. :180.0
1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:0.00000	1st Qu.:263.4
Median :0.0000	Median :0.0000	Median :0.0000	Median :0.00000	Median :307.8
Mean :0.2952	Mean :0.1333	Mean :0.1905	Mean :0.04762	Mean :306.7
3rd Qu.:1.0000	3rd Qu.:0.0000	3rd Qu.:0.0000	3rd Qu.:0.00000	3rd Qu.:348.2
Max. :1.0000	Max. :1.0000	Max. :1.0000	Max. :1.00000	Max. :437.5

In the previous tables we can see a short summary of the variables. **ports1**, **ports2** and **ports3** indicate the number of mountain sections in a given stage and their category (1,2 or 3). **year** corresponds to the year in which this data was recorded (1 to 6). **Week** is the week of the race in wich the stage takes place (1 to 3). The variables **bef_mount** and **aft_mount** tell us whether a stage took place before or after a mountain stage, respectively. Similarly, the variables **bef_tt** and **aft_tt** indicate if a stage took place before or after a time trial stage. Finally, the variable **last** tells us if that stage was the last of the whole race.

By the looks of the summary we can't seem to find any outliers or abnormal values. Most of the explanatory variables range from 0 to very low values and are natural numbers. The only continuous variables are **ForecastedTime**, **Distance**, **HeightInc** and **AccumIncr**. This makes sense because the first variable indicates time and time is a continuous variable and the rest indicate distance which can also be continous.

The variable **Distance** seems to be quite balanced with a mean of 193 and min and max values of 111 and 264 respectively. **HeightIncr**, however, is the only variable that presents negative values and has very high variance. Its minimum value is -940 and the maximum value is 2310.

By taking a closer look at the data we detect some abnormal values in the **last** variable: Since this variable indicates if a stage was the last of the whole race, then there should as many stages with this value equal to 1 as years of data have been recorded. Since the recorded stages are from 6 different years then there should be 6 rows, but instead there are only 5, meaning that there is at least 1 missing stage.

```
data[data$last == 1,]$year
```

```
## [1] 6 5 4 2 1
```

If we look at the data we can easily see that the missing value belongs to the year 3. However, this will probably not greatly affect our predicting power. There is something to be said about the variables **year**, **week**, **bef_mount**, **aft_mount**, **bef_tt**, **aft_tt** and **last**: all of these variables are categorical and indicate the group or category a row belongs to. For this, we should transform these variables into factors.

```
data$year      <- as.factor(data$year)
data$week      <- as.factor(data$week)
data$bef_mount <- as.factor(data$bef_mount)
data$aft_mount <- as.factor(data$aft_mount)
data$aft_tt    <- as.factor(data$aft_tt)
data$bef_tt    <- as.factor(data$bef_tt)
data$last      <- as.factor(data$last)
```

Now that we have transformed the categorical columns to factors we should take another look at the summary of these variables:

```
kable(summary(data[,9:15]))
```

year	week	bef_mount	aft_mount	bef_tt	aft_tt	last
1:15	1:33	0:74	0:74	0:91	0:85	0:100
2:18	2:34	1:31	1:31	1:14	1:20	1: 5
3:17	3:38	NA	NA	NA	NA	NA
4:17	NA	NA	NA	NA	NA	NA
5:18	NA	NA	NA	NA	NA	NA
6:20	NA	NA	NA	NA	NA	NA

We can se that for the variables **year** and **week** the variables are quite balanced, each group contains a very similar amount of observations. The rest, however, are more unbalanced. `bef_mount` has more than twice 0s than 1s, which means there are twice the stages that do not precede a mountain stage. The same happens with `aft_mount`, which makes sense for it means there are only 31 stages after a mountain stage. Since they are the same numbers as the `bef_mount`, this tells us that the mountain stages are always in the middle and are not the first or last stages.

In `bef_tt` there are 9 times more stages **not** preceding a time trial stage than does preceding it. There are 4 times less after trial stages. And finally, as mentioned before, there are only 5 last stages and 100 that are normal ones.

Let's finally take a closer look to the numerical attributes:

```
num <- 6:8; num <- append(num, 16)
kable(summary(data[,1:5]))
```

Time	Distance	HeightIncr	AccumIncr	portsE
Min. :171.6	Min. :111.0	Min. :-940.0	Min. : 190	Min. :0.0000
1st Qu.:258.1	1st Qu.:167.2	1st Qu.:-180.0	1st Qu.: 510	1st Qu.:0.0000
Median :301.4	Median :196.0	Median : 70.0	Median :1350	Median :0.0000
Mean :302.6	Mean :193.0	Mean : 227.9	Mean :1477	Mean :0.1524
3rd Qu.:345.8	3rd Qu.:219.5	3rd Qu.: 540.0	3rd Qu.:2300	3rd Qu.:0.0000
Max. :439.7	Max. :264.0	Max. :2310.0	Max. :4216	Max. :2.0000

```
kable(summary(data[,num]))
```

ports1	ports2	ports3	ForecastedTime
Min. :0.000	Min. :0.0000	Min. :0.0000	Min. :180.0
1st Qu.:0.000	1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:263.4
Median :0.000	Median :0.0000	Median :1.0000	Median :307.8
Mean :0.581	Mean :0.4857	Mean :0.8476	Mean :306.7
3rd Qu.:1.000	3rd Qu.:1.0000	3rd Qu.:2.0000	3rd Qu.:348.2
Max. :3.000	Max. :3.0000	Max. :4.0000	Max. :437.5

We can see that the mean and median in Time are pretty close, which means that there is not an outlier that drastically decreases or increases the mean. The same happens with Distance, AccumIncr and ForacastedTime.

However, HeightIncr has a low Median but a high mean, which means that in general there is not a stable increase of height but some stages have a high increase of it. And we can see that there is at least one stage

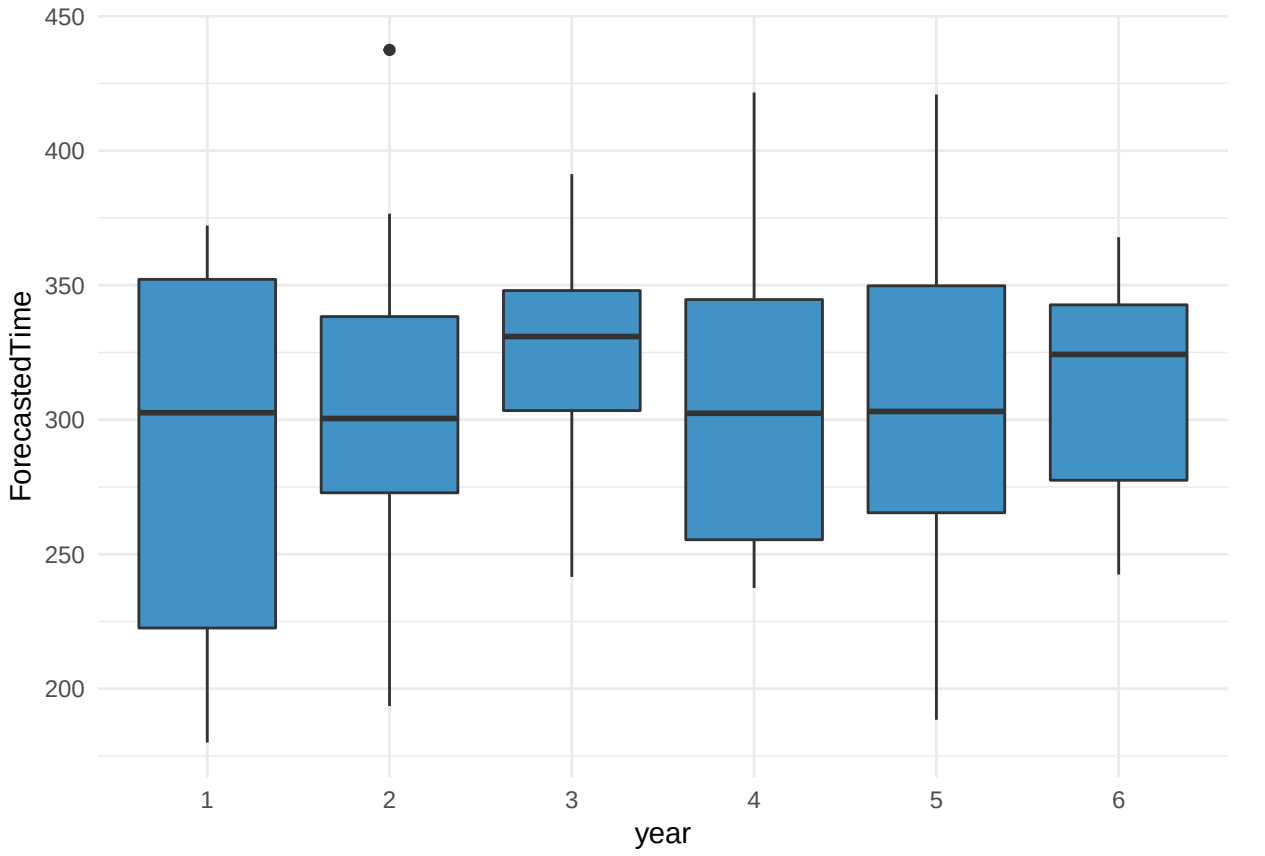
that consists of a decrease of almost 1km, whereas there is at least one stage of an increase of over 2km.

TODO: Comentar algo dels ports?

Data exploration visualization

Now, in order to find outliers and to see how the data behaves we will plot some boxplots. First of all we would like to see if the forecasted times are different depending on the year in which the race took place.

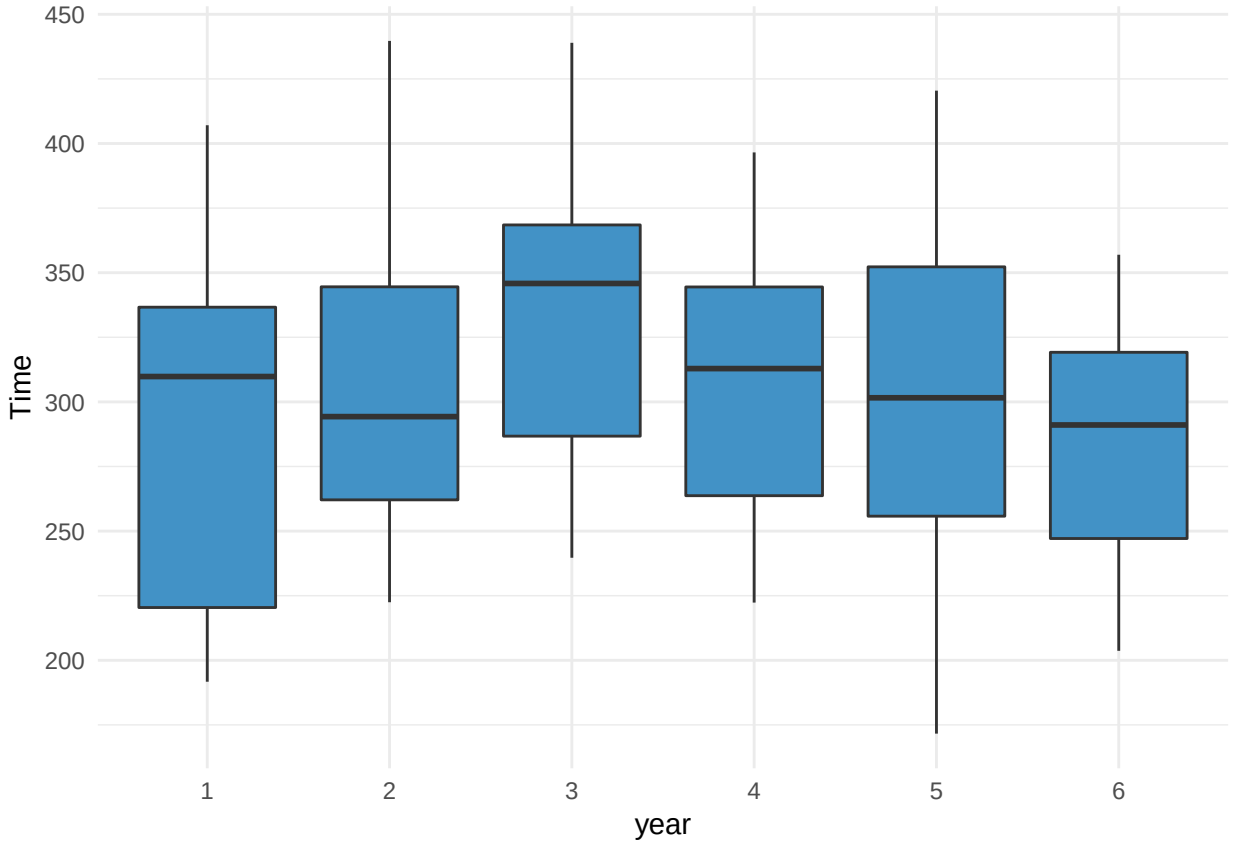
```
ggplot(data = data) +
  aes(x = year, y = ForecastedTime) +
  geom_boxplot(fill = "#4292c6") +
  theme_minimal()
```



At first glance we see that the medians are really similar among all years, excepting year 3 and year 6, which are about 25 units higher. The interquartile ranges for every year are different. Even though the Q3 percentile is very similar for all years, the Q1 percentile varies from 225 to 300. It is important to say that there appears to be an outlier in the second year.

Let's take a look at real Times:

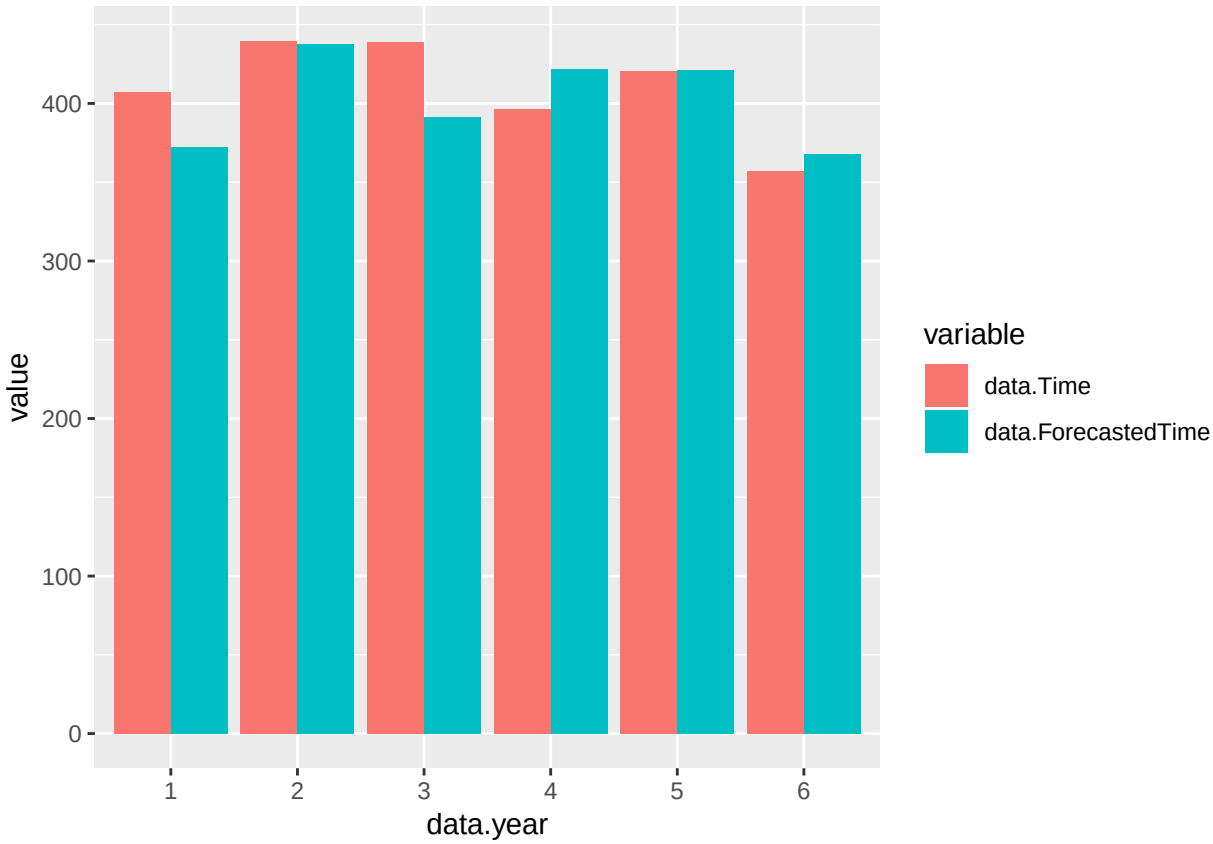
```
ggplot(data = data) +
  aes(x = year, y = Time) +
  geom_boxplot(fill = "#4292c6") +
  theme_minimal()
```



We can see how the highest Time is on the 3rd year, and there's a lot of variance on the 1st and 5th years' races. The second year seems to have some stages with a very high time, so there's a lot of variance on the upper bound, while on the 5th year there does is a lot of variance on the lower bound, meaning there are some stages with less Time than the rest. The 5th year is the only one where this phenomena is so visible.

And let's compare the forecasted Times to the real Times:

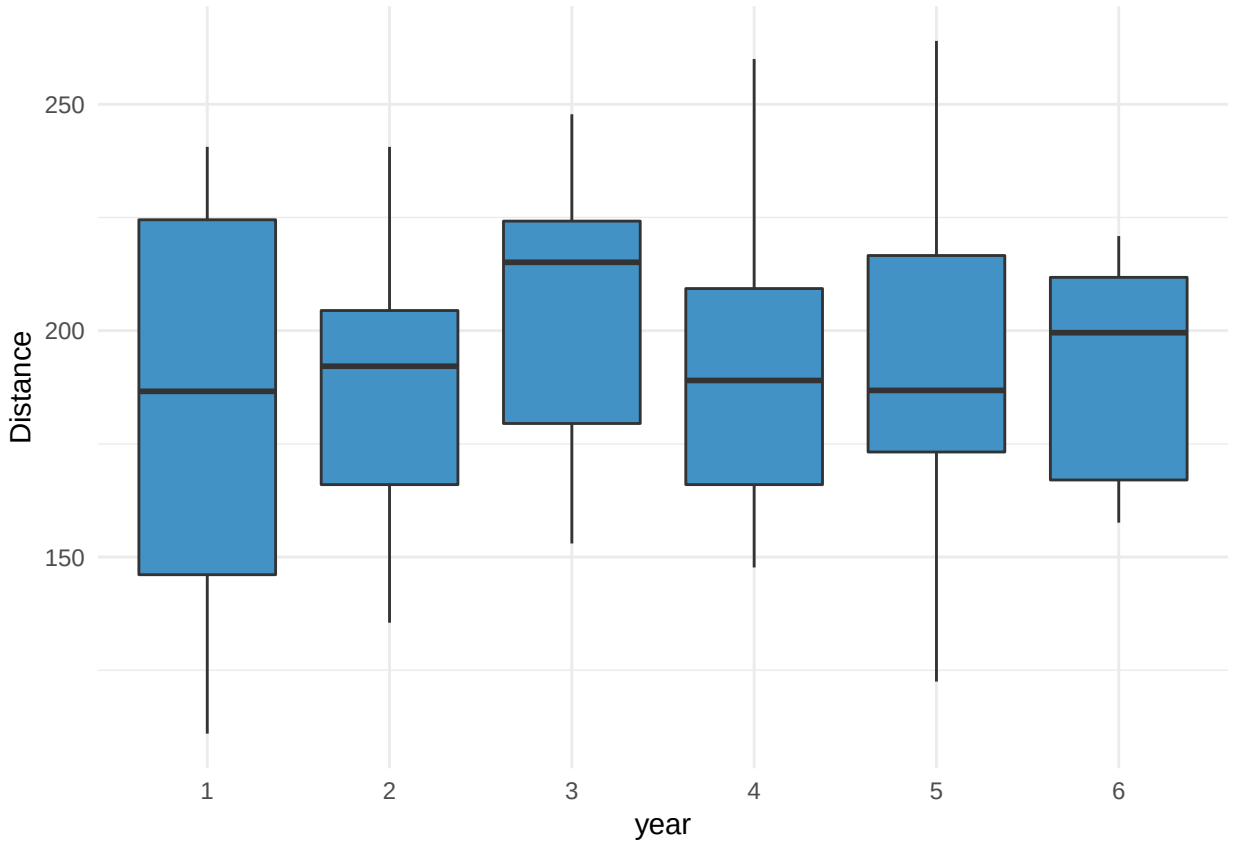
```
df1 <- data.frame(data$Time, data$ForecastedTime, data$year)
df2 <- melt(df1, id.vars='data.year')

ggplot(df2, aes(x=data.year, y=value, fill=variable)) +
    geom_bar(stat='identity', position='dodge')
```



We can see how the Forecasted time is pretty on par with the real time, so the prediction is very good and varies very little in small units.

The worst predictions have been obtained on the 1st and 3rd year. And it has nailed the predictions on the 2nd, 5th and 6th year. An overestimation has been done on the 4th year.

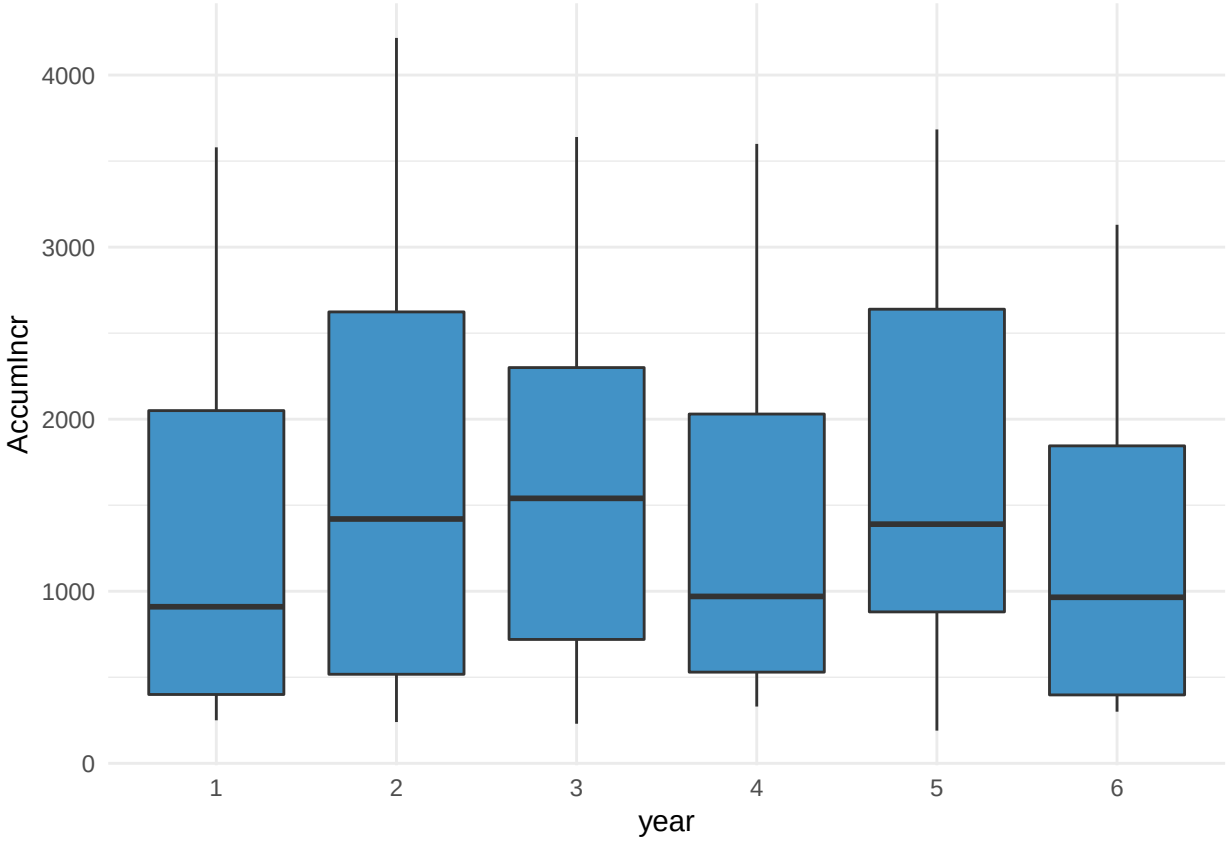
```
ggplot(data = data) +
  aes(x = year, y = Distance) +
  geom_boxplot(fill = "#4292c6") +
  theme_minimal()
```



In the plot above we can see how for each year, the distance does not vary a lot, except on the 3rd year, where there seems to be a higher overall distance but with small variance. The 1st year seems to be the one with most variance on distances between stages and the 5th one does not seem to vary so much, except on some stages qhere the distance differs more (between 100 and past 250). The 6th year seems to be the year with less variance in distance between stages.

Since the 4th year has very little distance, it may have affected the forecasted Time, which was above the real time.

```
ggplot(data = data) +
  aes(x = year, y = AccumIncr) +
  geom_boxplot(fill = "#4292c6") +
  theme_minimal()
```



Now, taking a look at the accumulated number of meters climbed on the stages for each year, we can see that the 2nd year is the one with most variance and, in general, most of them vary a lot on the upper bound but not so much on the lower one.

The 3rd year seems to be the one with a highest median and the 4th one seems to be the one with the lowest one.

TODO: Fer més boxplots?? TODO: Corr plot? Pero no ho hem fet a classe..

Model selection

Let's first try creating a linear model. NO usage of Year.

```
lm.model <- lm(Time ~ Distance + HeightIncr + AccumIncr  + ports1 + ports2 + ports3 + bef_mount + aft_m
summary(lm.model)
```

```
##
## Call:
## lm(formula = Time ~ Distance + HeightIncr + AccumIncr + ports1 +
##     ports2 + ports3 + bef_mount + aft_mount + bef_tt + aft_tt +
##     last, data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -50.82 -13.90  -3.30  11.57  58.80
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -39.019196  18.102582  -2.155  0.03371 *
## Distance      1.661638   0.083100  19.996  < 2e-16 ***
## HeightIncr    0.001041   0.005041   0.206  0.83690
## AccumIncr     0.018024   0.005654   3.188  0.00195 **
## ports1       -8.017926   5.098681  -1.573  0.11922
## ports2       -6.696263   3.984025  -1.681  0.09616 .
## ports3       -2.581659   2.475275  -1.043  0.29966
## bef_mount1    5.650151   5.821637   0.971  0.33429
## aft_mount1    0.582433   6.145878   0.095  0.92470
## bef_tt1       5.790429   7.512843   0.771  0.44282
## aft_tt1       5.169633   6.754871   0.765  0.44602
## last1        11.195790  12.024858   0.931  0.35424
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.91 on 93 degrees of freedom
## Multiple R-squared:  0.854,  Adjusted R-squared:  0.8367
## F-statistic: 49.44 on 11 and 93 DF,  p-value: < 2.2e-16
```

We can see we have a high adjusted R squared. Distance, heightIncr AccumIncr. Ports3 maybe. We prune those irrelevant.

```
lm.model2 <- lm(Time ~ Distance  + AccumIncr + ports2, data)
summary(lm.model2)
```

```
##
## Call:
## lm(formula = Time ~ Distance + AccumIncr + ports2, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.769 -13.308  -2.277  14.607  62.103
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -36.015106  14.800328  -2.433   0.0167 *
## Distance      1.665626   0.073784  22.574  < 2e-16 ***
## AccumIncr     0.012950   0.002472   5.239 8.85e-07 ***
## ports2       -4.142799   3.231176  -1.282   0.2027
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.73 on 101 degrees of freedom
## Multiple R-squared:  0.8438, Adjusted R-squared:  0.8391
## F-statistic: 181.8 on 3 and 101 DF,  p-value: < 2.2e-16
```

R squared adj increases slightly. POrts2 irrelevanty.

```
lm.model3 <- lm(Time ~ Distance  + AccumIncr, data)
summary(lm.model3)
```

```
##
## Call:
## lm(formula = Time ~ Distance + AccumIncr, data = data)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -55.718 -12.772  -3.763  14.480  62.436
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.392677  14.807793  -2.525   0.0131 *
## Distance      1.673312   0.073771  22.682  < 2e-16 ***
## AccumIncr     0.011516   0.002211   5.208 9.96e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.81 on 102 degrees of freedom
## Multiple R-squared:  0.8412, Adjusted R-squared:  0.8381
## F-statistic: 270.2 on 2 and 102 DF,  p-value: < 2.2e-16
```

R adj squared decreased slightly. We keep this model because it has less param.

Residuals check.

```
ncol(data)
```

```
## [1] 16
```

```
nrow(data)
```

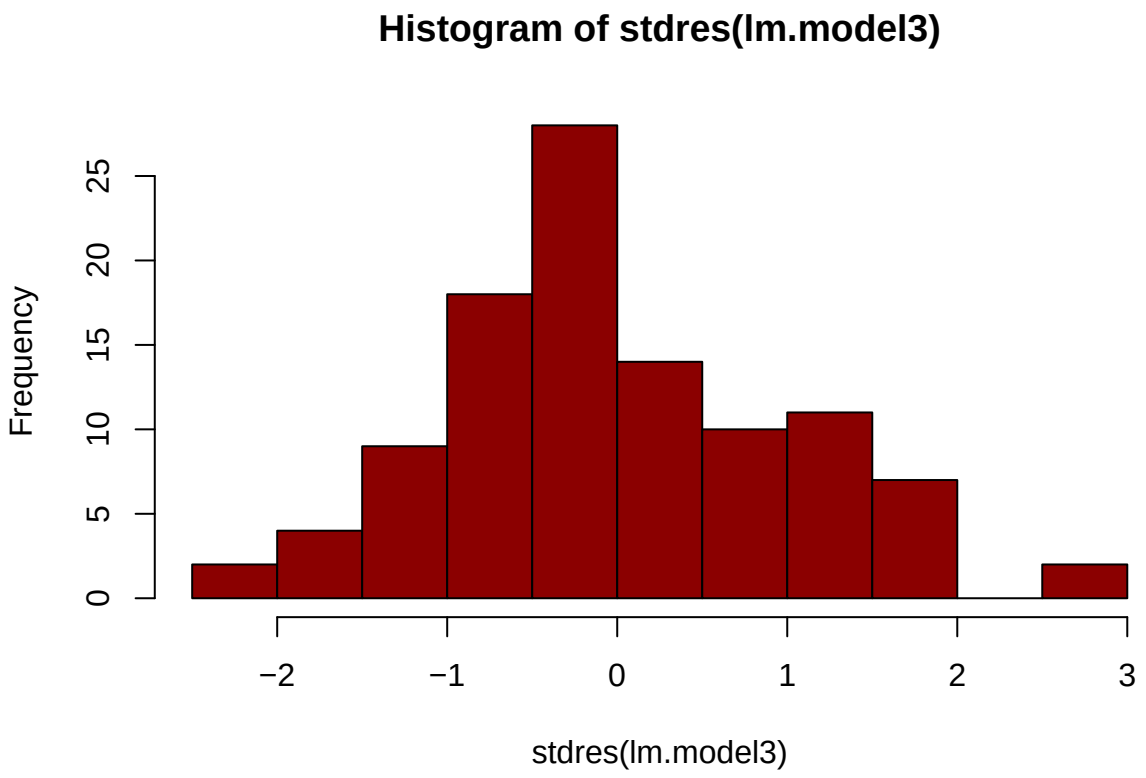
```
## [1] 105
```

Since we have 105 observations and 16 parameters, thus

$$N_{obs} >> p$$

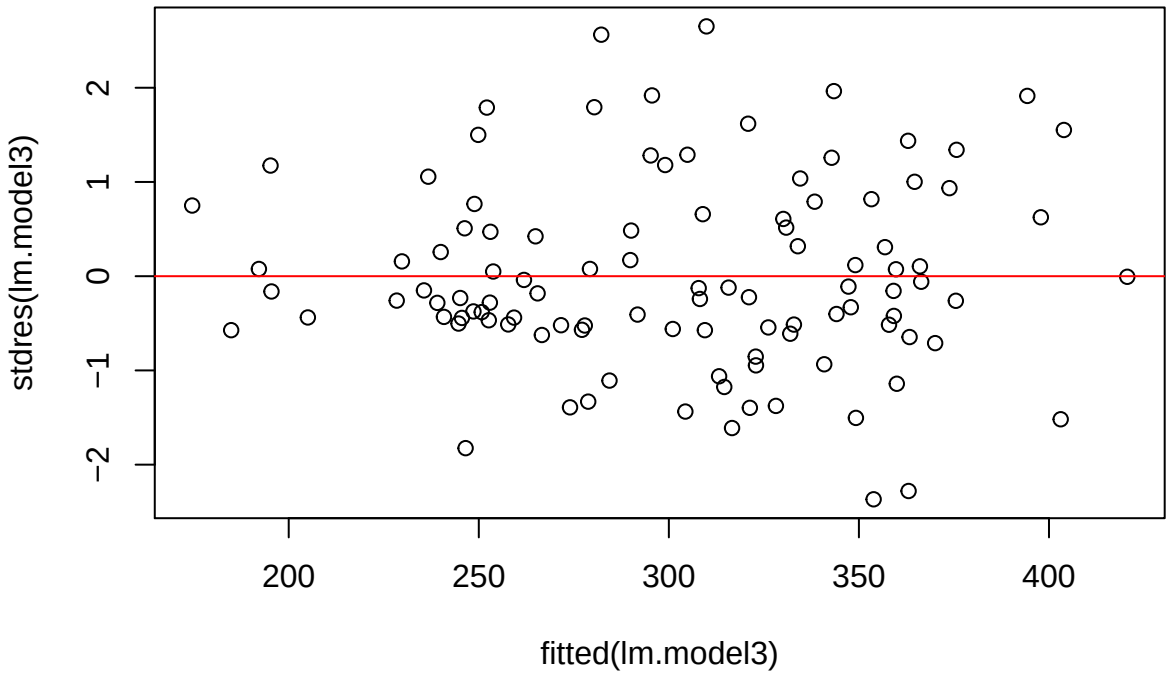
and we can check the histogram of standard residuals.

```
hist(stdres(lm.model3), col="darkred")
```



The histogram shows a normal distrib.

```
plot(fitted(lm.model3), stdres(lm.model3))
abline(h=0, col="red")
```

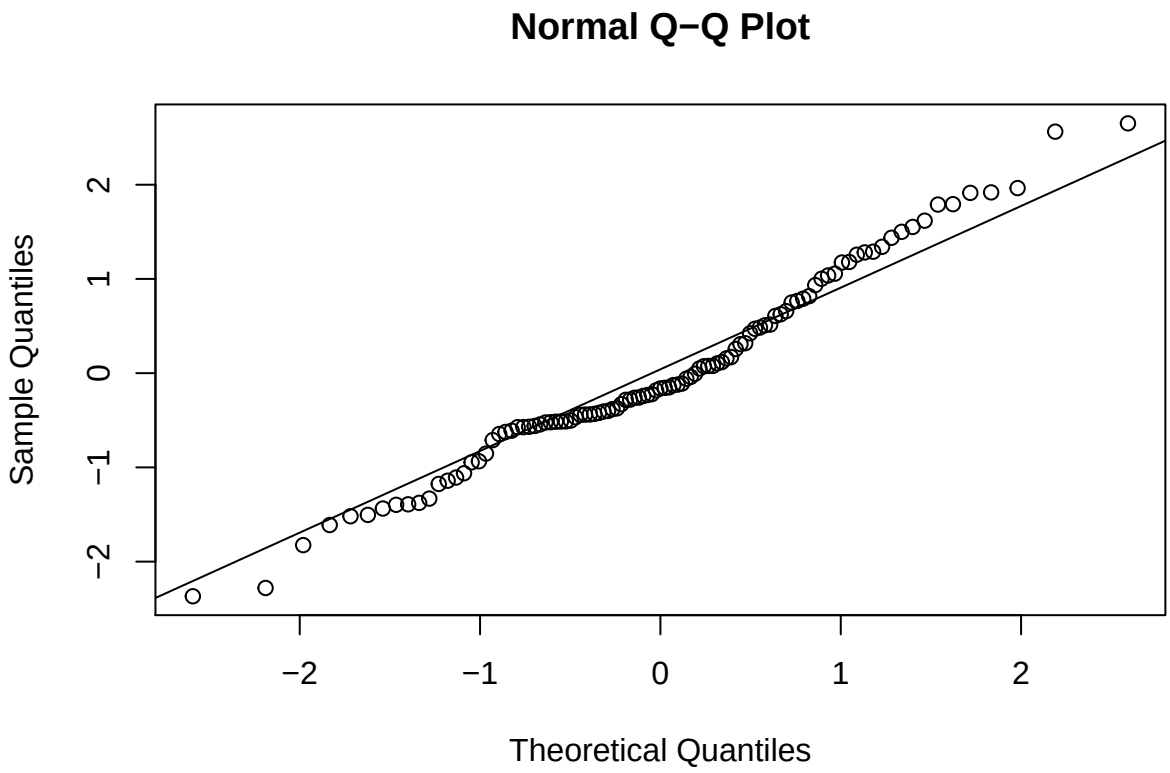


We can see that there is no shape and the points are just scattered randomly, thus there is no pattern.

It seems that 95% or more of the points range between -2 and 2, and thus we can see that Homocedasticity is complied because there is no shape.

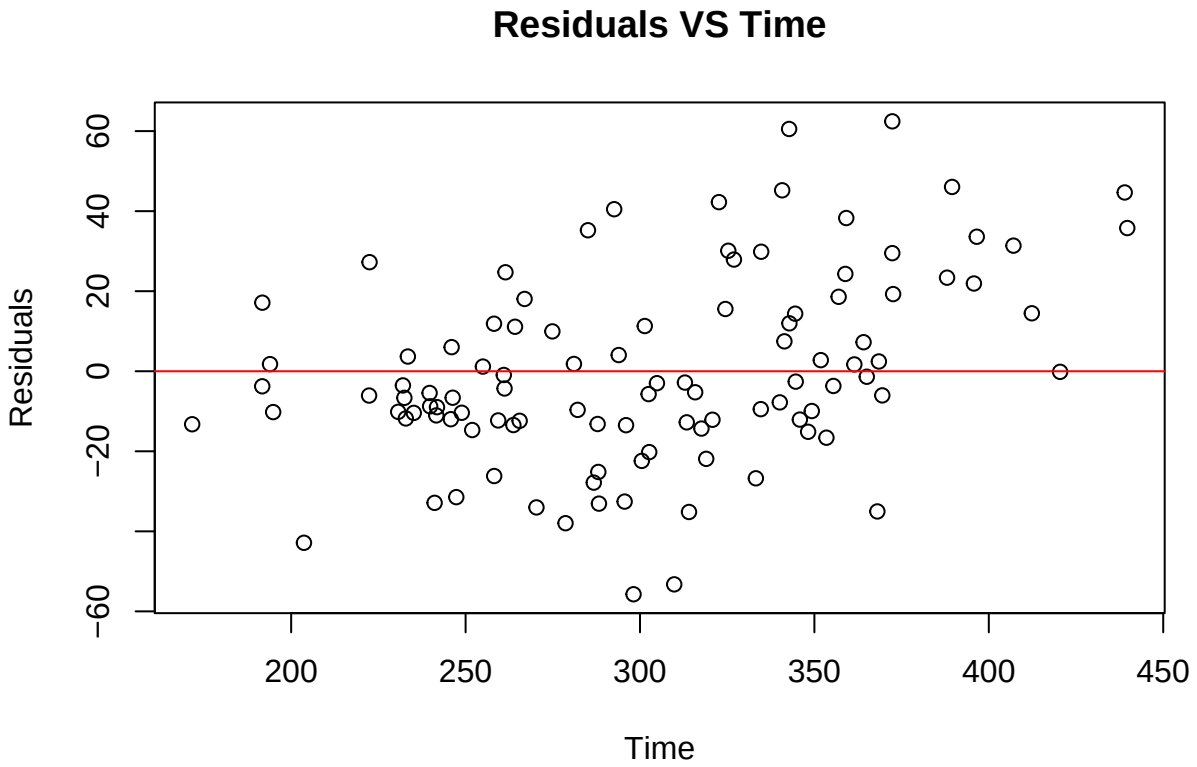
If we qqplot below the residuals we can see that the standardised residuals follow a normal distirbution and thus the normality assumption is satisfied.

```
res = stdres(lm.model3)
qqnorm(res); qqline(res)
```



Finally we are going to check the independence of resiudals against the predicted value:

```
res <- residuals(lm.model3)
plot(res ~ data$Time, ylab = 'Residuals', xlab = 'Time', main = 'Residuals VS Time')
abline(h = 0, col='red')
```



We can see that there is indeed no pattern in the residuals vs time so we can assume independence of variables.

Generalized Linear Model

Now we are going to try to model a Generalized Linear Model for our data in order to predict the Time.

We assume Time follows a general probability distribution (not necessarily a Normal distribution), and we are going to assume there is some linearity between the transformation of Time μ where

$$\mu = E(Time)$$

and the covariates.

Prediction

In this section we are going to compare our predictions with those on the Forecasted by an expert.