Final Assignment (Τελική Εργασία)

Παπαντζίκος Δημήτριος

Ολόκληρη η εργασία έγινε στην Αγγλική γλώσσα με σκοπό να με βοηθήσει να εξοικειωθώ με τη χρήση της στην επικοινωνία των αποτελεσμάτων από την μελέτη μου. Ελπίζω να μην αποτελέσει πρόβλημα. Καλή ανάγνωση!

This is a complete assignment for the course of *Statistical Learning and Knowledge Processing* for the winter semester *20-21*. In this assignment what we first have to do, is to choose a dataset from R libraries or from a small subset of Kaggle. Personally, I chose the **FirstYearGPA** (from R libraries) dataset with *219* rows each of which has some information for a unique student. After choosing the dataset we have to **describe** it and then make some first calculations to have a general look (*Descriptive Statistics*). Secondly, we will make some **hypotheses testing** in order to make statistically strong assumptions for the population. Lastly, we are ready to move on with **predicting** our valuable variable, the students’ grade (factor variable of classes A, B, C based on GPA). Enjoy the process!

# Part 1 : Describing the dataset

### First, we will take a look in the structure of the dataset to get a grasp of what we have :

## 'data.frame': 219 obs. of 10 variables:  
## $ GPA : num 3.06 4.15 3.41 3.21 3.48 2.95 3.6 2.87 3.67 3.49 ...  
## $ HSGPA : num 3.83 4 3.7 3.51 3.83 3.25 3.79 3.6 3.36 3.7 ...  
## $ SATV : int 680 740 640 740 610 600 710 390 630 680 ...  
## $ SATM : int 770 720 570 700 610 570 630 570 560 670 ...  
## $ Male : int 1 0 0 0 0 0 0 0 0 0 ...  
## $ HU : num 3 9 16 22 30.5 18 5 10 8.5 16 ...  
## $ SS : num 9 3 13 0 1.5 3 19 0 15.5 12 ...  
## $ FirstGen : int 1 0 0 0 0 0 0 0 0 0 ...  
## $ White : int 1 1 0 1 1 1 1 0 1 1 ...  
## $ CollegeBound: int 1 1 1 1 1 1 1 0 1 1 ...

### We will make some changes in order to make our dataset more “meaningful”

So our modified dataset looks like this :

## 'data.frame': 219 obs. of 10 variables:  
## $ GPA : num 3.06 4.15 3.41 3.21 3.48 2.95 3.6 2.87 3.67 3.49 ...  
## $ HSGPA : num 3.83 4 3.7 3.51 3.83 3.25 3.79 3.6 3.36 3.7 ...  
## $ SATV : int 680 740 640 740 610 600 710 390 630 680 ...  
## $ SATM : int 770 720 570 700 610 570 630 570 560 670 ...  
## $ Male : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...  
## $ HU : num 3 9 16 22 30.5 18 5 10 8.5 16 ...  
## $ SS : num 9 3 13 0 1.5 3 19 0 15.5 12 ...  
## $ FirstGen : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...  
## $ White : Factor w/ 2 levels "0","1": 2 2 1 2 2 2 2 1 2 2 ...  
## $ CollegeBound: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 1 2 2 ...

### Variable description :

* **GPA :** First-year college GPA on a 0.0 to 4.0 scale
* **HSGPA :** High school GPA on a 0.0 to 4.0 scale
* **SATV :** Verbal/critical reading SAT score
* **SATM :** Math SAT score
* **Male :** Male(1) or Female(0)
* **HU :** Number of credit hours earned in humanities courses in high school
* **SS :** Number of credit hours earned in social science courses in high school
* **FirstGen :** 1 = student is the first in her/his family to attend college, 0 = otherwise
* **White :** 1 = white students, 0 = minority students
* **CollegeBound :** 1 = attended a high school where >=50% students intended to go on to college, 0 = otherwise

### We are now able to move on to Descriptive Statistics on our dataset

#### Taking a general look :

## GPA HSGPA SATV SATM Male   
## Min. :1.930 Min. :2.340 Min. :260.0 Min. :430.0 0:117   
## 1st Qu.:2.745 1st Qu.:3.170 1st Qu.:565.0 1st Qu.:580.0 1:102   
## Median :3.150 Median :3.500 Median :610.0 Median :640.0   
## Mean :3.096 Mean :3.453 Mean :605.1 Mean :634.3   
## 3rd Qu.:3.480 3rd Qu.:3.760 3rd Qu.:670.0 3rd Qu.:690.0   
## Max. :4.150 Max. :4.000 Max. :740.0 Max. :800.0   
## HU SS FirstGen White CollegeBound  
## Min. : 0.00 Min. : 0.000 0:194 0: 46 0: 17   
## 1st Qu.: 8.00 1st Qu.: 3.000 1: 25 1:173 1:202   
## Median :13.00 Median : 6.000   
## Mean :13.11 Mean : 7.249   
## 3rd Qu.:17.00 3rd Qu.:11.000   
## Max. :40.00 Max. :21.000

#### Descriptive Statistics for the categorical variables :

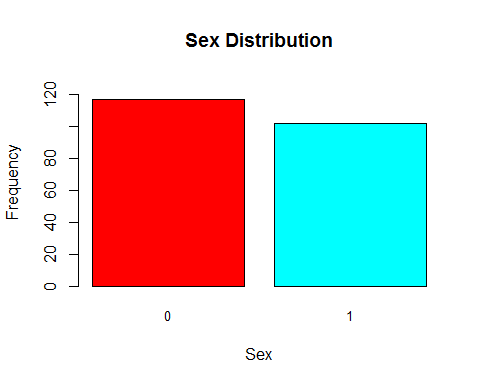
##### Male (factor) :

###### Frequency tables :

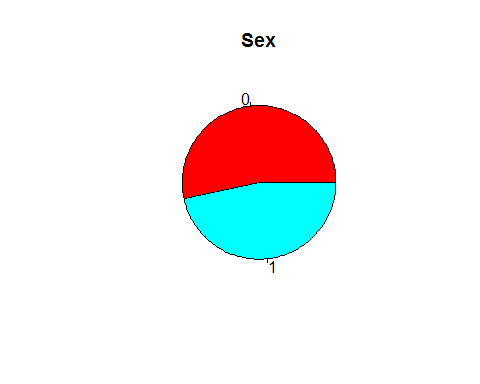
## Sex Frequency Relative Frequency  
## Female 0 117 0.5342466  
## Male 1 102 0.4657534

We can see that there are slightly more females in the sample.

###### Barplot :



###### Pie chart :



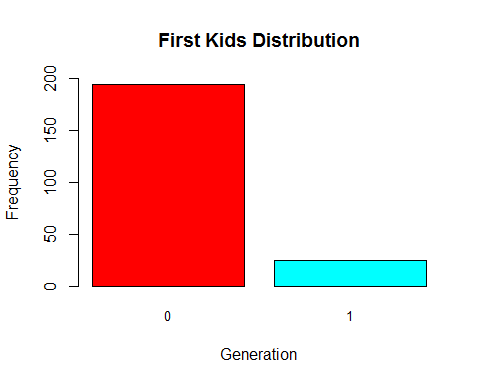
##### FirstGen (factor) :

###### Frequency tables :

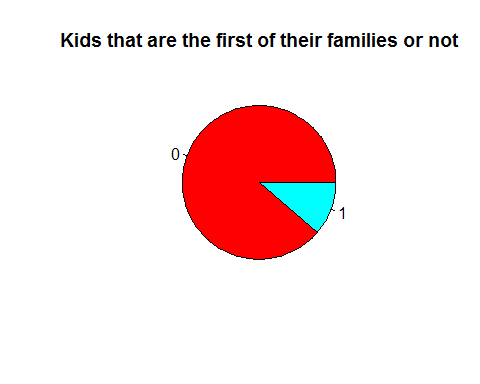
## First Kid of the family Frequency Relative Frequency  
## No 0 194 0.8858447  
## Yes 1 25 0.1141553

We can clearly see that there are much more kids that are not the first kids of their families to go to college.

###### Barplot :



###### Pie chart :



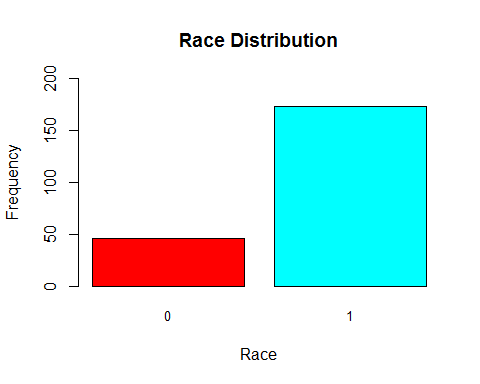
##### White (factor) :

###### Frequency tables :

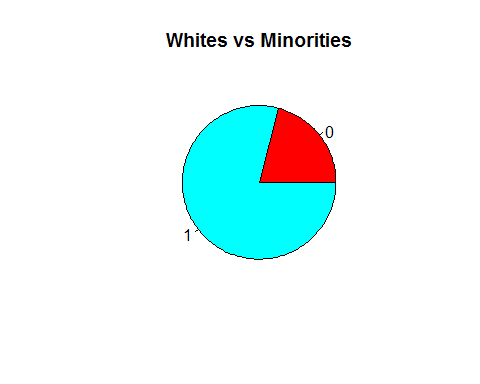
## Race of Students Frequency Relative Frequency  
## Minority 0 46 0.2100457  
## White 1 173 0.7899543

We can clearly see that there are much more white students than minority students.

###### Barplot :



###### Pie chart :



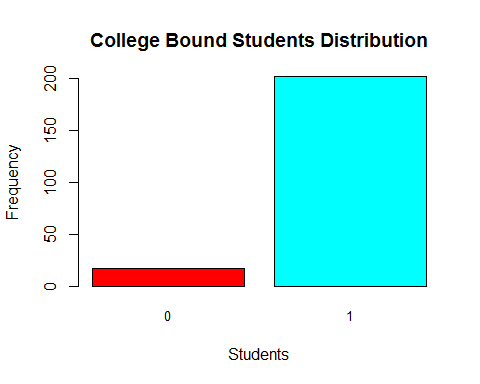
##### CollegeBound (factor) :

###### Frequency tables :

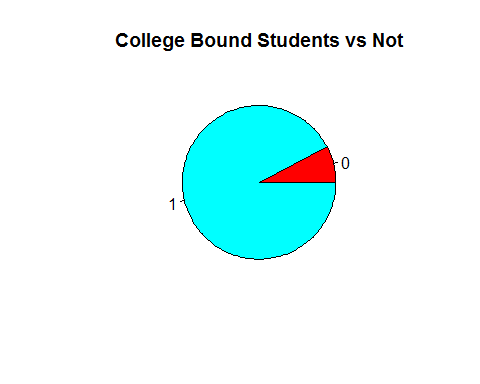
## College Bound Students Frequency Relative Frequency  
## No 0 17 0.07762557  
## Yes 1 202 0.92237443

We can clearly see that more than 9/10 students were in a high school where >50% of students aimed for college

###### Barplot :



###### Pie chart :



##### Crosstabulation Matrix for Race and H.S. boundness to College :

## CollegeBound  
## White No Yes  
## No 3 43  
## Yes 14 159

We cannot observe a significant difference by means of race and high school’s students’ boundness to college so we make a chi-squared test to be sure

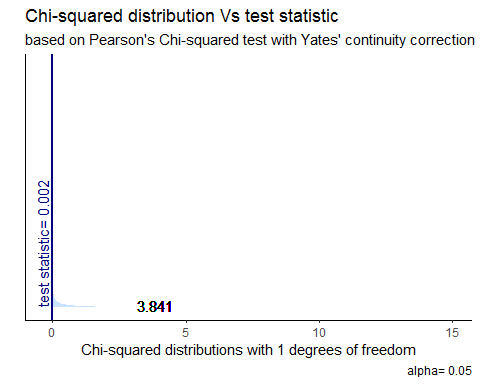
##### Chi-Squared Test :

## Warning in chisq.test(White, CollegeBound): Chi-squared approximation may be  
## incorrect

##   
## Pearson's Chi-squared test with Yates' continuity correction  
##   
## data: White and CollegeBound  
## X-squared = 0.0019253, df = 1, p-value = 0.965

The p-value is really high (0.965) so we cannot reject the null hypothesis that the race and the high school boundness is not correlated.

###### Visualising the result :



##### Crosstabulation Matrix for either the kid being the first in the family that goes to college or not and its H.S. boundness to College :

## CollegeBound  
## FirstGen No Yes  
## No 13 181  
## Yes 4 21

We cannot observe a significant difference by means of order of child and high school boundness to college so we make a chi-squared test to be sure

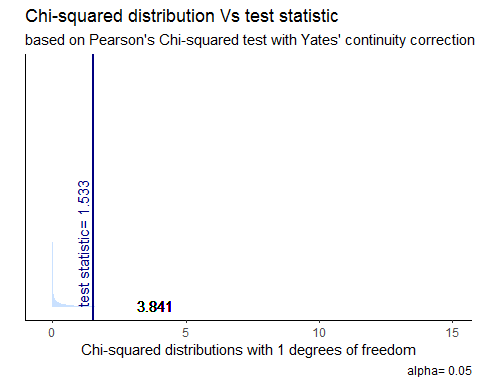
##### Chi-Squared Test :

## Warning in chisq.test(FirstGen, CollegeBound): Chi-squared approximation may be  
## incorrect

##   
## Pearson's Chi-squared test with Yates' continuity correction  
##   
## data: FirstGen and CollegeBound  
## X-squared = 1.5335, df = 1, p-value = 0.2156

The p-value is higher than 0.05 (0.2156) so we cannot reject the null hypothesis that the order of child and the high school boundness is not correlated.

###### Visualising the result :



There is no point in analysing whether there is or not a correlation between the student being or not the first kid of the family to attend college and his race or sex so we move on.

#### Descriptive Statistics for the numerical variables :

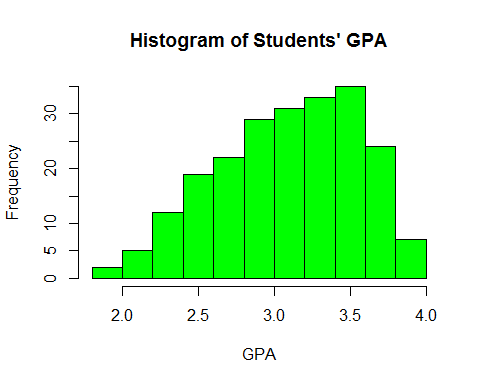
describe(dataset, IQR = T)[-c(5,8,9,10),-c(1,2,6,7,13)]

## mean sd median min max range skew kurtosis IQR  
## GPA 3.10 0.46 3.15 1.93 4 2.07 -0.33 -0.66 0.73  
## HSGPA 3.45 0.37 3.50 2.34 4 1.66 -0.45 -0.57 0.59  
## SATV 605.07 83.39 610.00 260.00 740 480.00 -0.99 1.48 105.00  
## SATM 634.29 75.24 640.00 430.00 800 370.00 -0.40 -0.14 110.00  
## HU 13.11 7.22 13.00 0.00 40 40.00 0.73 0.48 9.00  
## SS 7.25 5.00 6.00 0.00 21 21.00 0.45 -0.46 8.00

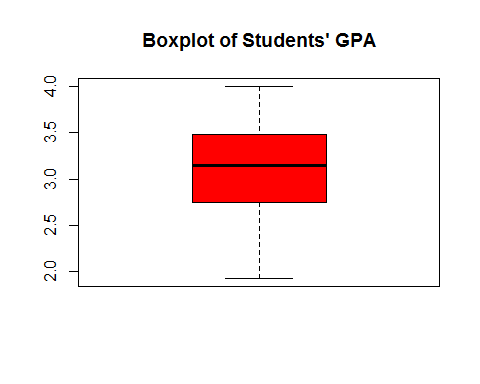
With this line of code we have every information that we want about the numerical variables in a well-organized matrix

##### GPA (numeric) :

###### Histogram :



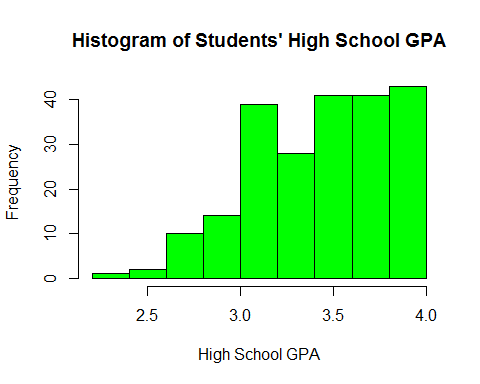
###### Boxplot :



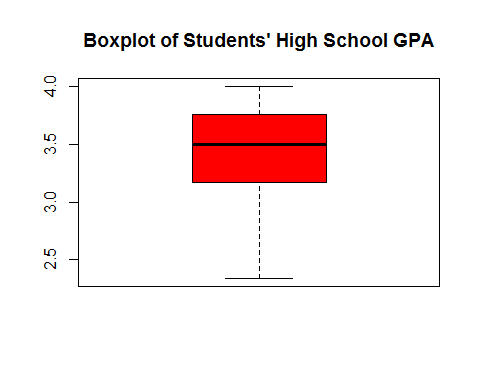
We may think that GPA’s tend to follow a normal distribution, but we will test that assumption later on.

##### HSGPA (numeric) :

###### Histogram :

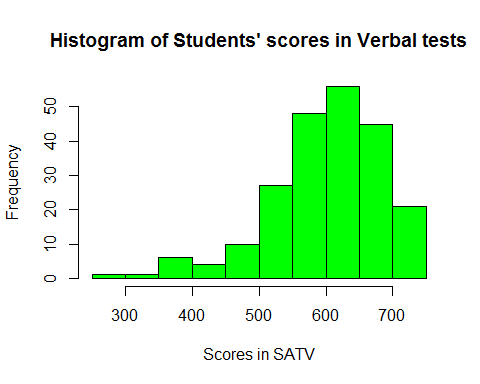


###### Boxplot :

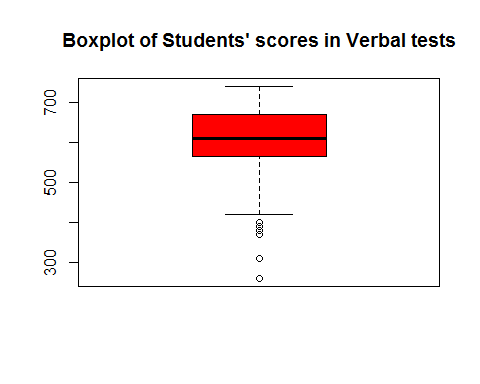


##### SATV (numeric) :

###### Histogram :

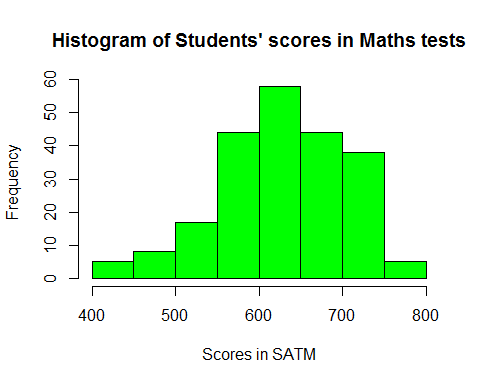


###### Boxplot:

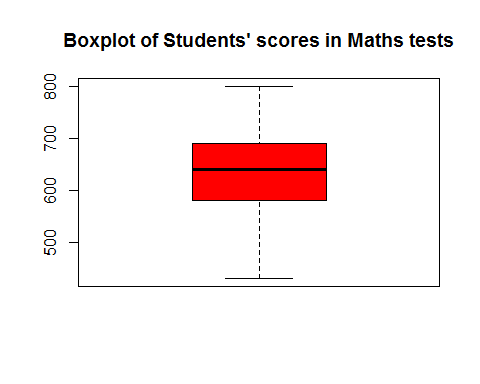


##### SATM (numeric) :

###### Histogram :

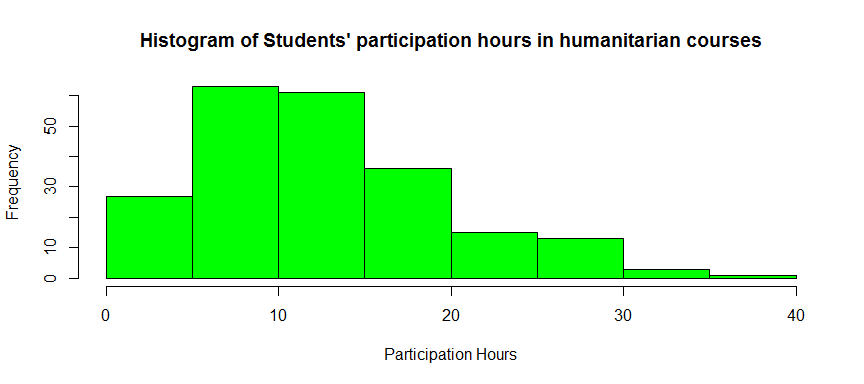


###### Boxplot :

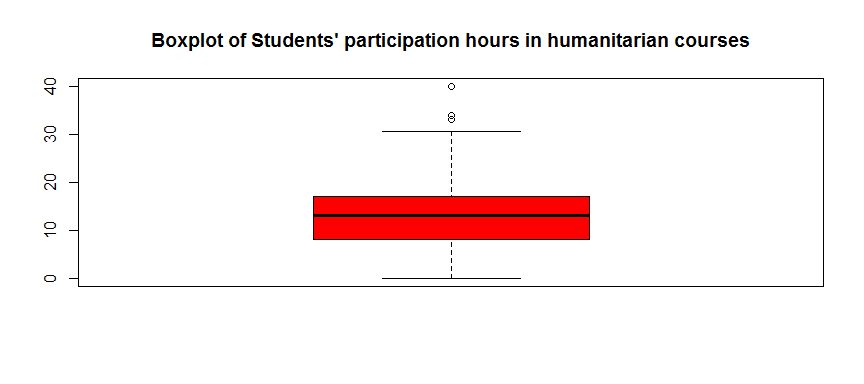


##### HU (numeric) :

###### Histogram :

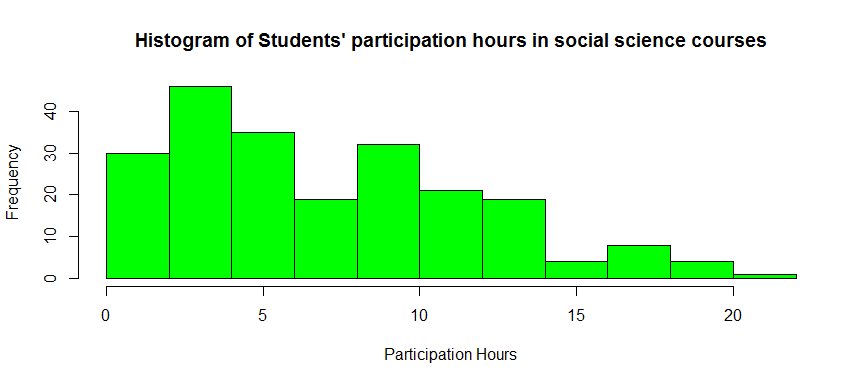


###### Boxplot :

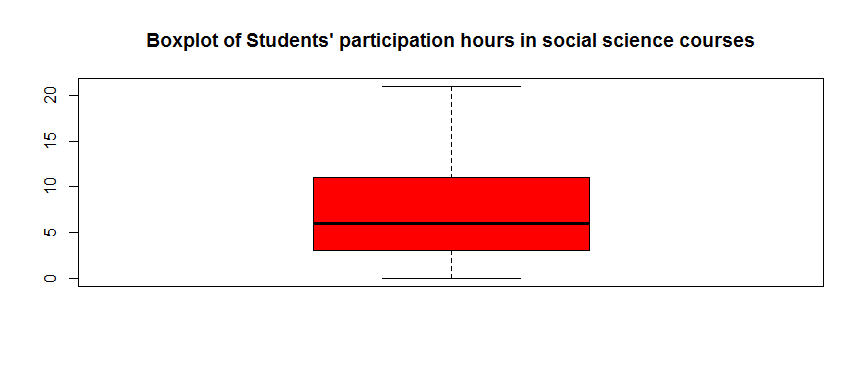


##### SS (numeric) :

###### Histogram :

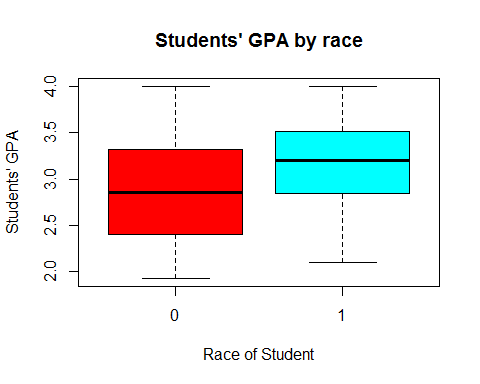


###### Boxplot :



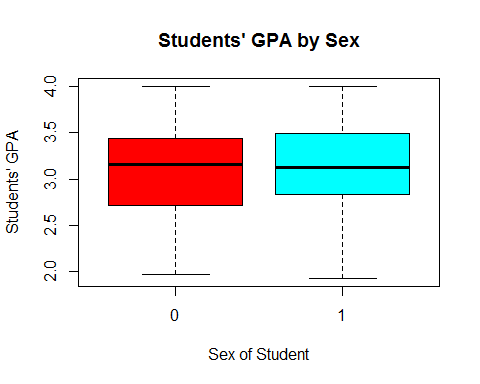
#### Descriptive Statistics for Factor and Numeric Variables

##### GPA by Race :



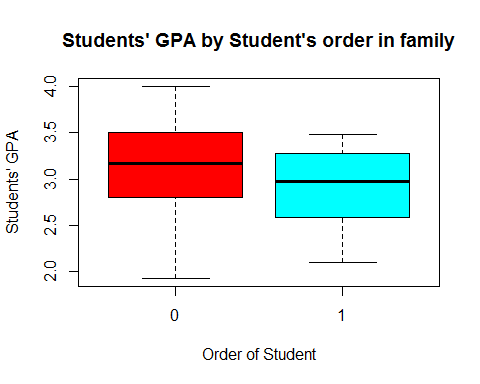
We can see that White Students’ GPA’s are higher in general than minority students

##### GPA by Sex :



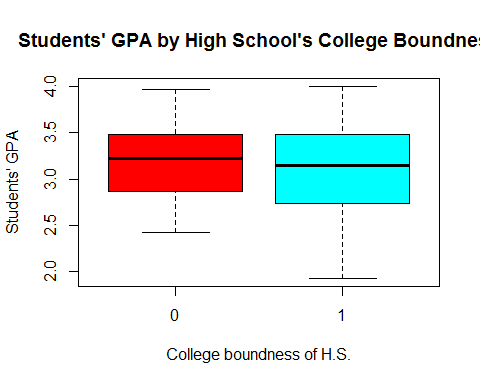
We can see that Male and Female Students’ GPA’s are not so different in general

##### GPA by FirstGen :



We can see that GPA’s of not first kids of families to attend college are higher than first kids of families to attend college.

##### GPA by CollegeBound :



We can see a slight difference in GPA’s with students from non-college-bound high schools being first.

#### We will now move on to check if some of our numeric variables obey the normal distribution

##### GPA :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: GPA  
## D = 0.059656, p-value = 0.05627

Even if our p-value is relatively small (0.06421) it is still higher than 0.05 so we cannot be confident to reject the null hypothesis. Therefore, we will make another test (the Shapiro-Wilk).

##   
## Shapiro-Wilk normality test  
##   
## data: GPA  
## W = 0.97891, p-value = 0.00231

Now, the p-value is less than 0.05 (0.005792) so we confidently reject the null hypothesis that the GPA’s obey a normal distribution.

##### HSGPA :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: HSGPA  
## D = 0.078618, p-value = 0.002229

The p-value is smaller than 0.05 (0.002229) so we reject the null hypothesis that the Students’ High School GPA’s obey a normal distribution.

##### SATV :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: SATV  
## D = 0.08962, p-value = 0.0002052

Again, we reject the null hypothesis that the scores of students in a verbal sat test follow a normal distribution (p-value = 0.0002052).

##### SATM :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: SATM  
## D = 0.066294, p-value = 0.02064

Like before, we reject the null hypothesis (p-value = 0.02064).

##### HU :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: HU  
## D = 0.086216, p-value = 0.0004468

We reject the null hypothesis (p-value = 0.0004468).

##### SS :

##   
## Lilliefors (Kolmogorov-Smirnov) normality test  
##   
## data: SS  
## D = 0.10889, p-value = 1.271e-06

We reject the null hypothesis (p-value = 0.000001271).

So, none of our numeric variables obeys a normal distribution. Therefore, we move on to non-parametrical tests.

#### Non-Parametrical Tests :

##### GPA :

##   
## Runs Test  
##   
## data: GPA  
## statistic = -0.40731, runs = 107, n1 = 109, n2 = 109, n = 218, p-value  
## = 0.6838  
## alternative hypothesis: nonrandomness

p-value is 0.6838 > 0.05 so we confidently accept the null hypothesis that the sample is random.

##### HSGPA :

##   
## Runs Test  
##   
## data: HSGPA  
## statistic = -0.61249, runs = 104, n1 = 109, n2 = 106, n = 215, p-value  
## = 0.5402  
## alternative hypothesis: nonrandomness

p-value is 0.5402 > 0.05 so we confidently accept the null hypothesis that the sample is random.

##### SATV :

##   
## Runs Test  
##   
## data: SATV  
## statistic = -0.52837, runs = 99, n1 = 107, n2 = 97, n = 204, p-value =  
## 0.5972  
## alternative hypothesis: nonrandomness

p-value is 0.5972 > 0.05 so we confidently accept the null hypothesis that the sample is random.

##### SATM :

##   
## Runs Test  
##   
## data: SATM  
## statistic = -0.4138, runs = 101, n1 = 101, n2 = 105, n = 206, p-value =  
## 0.679  
## alternative hypothesis: nonrandomness

p-value is 0.679 > 0.05 so we confidently accept the null hypothesis that the sample is random.

##### HU :

##   
## Runs Test  
##   
## data: HU  
## statistic = -0.24518, runs = 103, n1 = 99, n2 = 109, n = 208, p-value =  
## 0.8063  
## alternative hypothesis: nonrandomness

p-value is 0.8063 > 0.05 so we confidently accept the null hypothesis that the sample is random.

##### SS :

##   
## Runs Test  
##   
## data: SS  
## statistic = 1.553, runs = 104, n1 = 108, n2 = 81, n = 189, p-value =  
## 0.1204  
## alternative hypothesis: nonrandomness

p-value is 0.1204 > 0.05 so we confidently accept the null hypothesis that the sample is random.

*We are now ready to move on to non-parametrical Hypothesis testing*

# Part 2 : Non-Parametrical Hypothesis Testing

##### GPA ~ Sex :

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: dataset[Male == "0", "GPA"] and dataset[Male == "1", "GPA"]  
## W = 5618.5, p-value = 0.4569  
## alternative hypothesis: true location shift is not equal to 0

We can accept the null hypothesis that the GPA of Males is the same as Females

##### GPA ~ Race :

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: GPA by White  
## W = 2546, p-value = 0.0001764  
## alternative hypothesis: true location shift is not equal to 0

We can confidently reject the null hypothesis that the GPA of White students is the same as the Minority students. By taking into account a boxplot that we saw earlier, we can summarize that White students have higher GPA’s in general.

##### GPA ~ FirstGen :

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: GPA by FirstGen  
## W = 3105, p-value = 0.02267  
## alternative hypothesis: true location shift is not equal to 0

We can reject the null hypothesis that students’ GPA’s who are or are not the first in their families to attend college, are the same. By taking into account a boxplot that we saw earlier, we can summarize that students who are not the first kids in their families to attend college, tend to have higher GPA’s than students who are the first in their families to do so.

##### GPA ~ CollegeBound :

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: GPA by CollegeBound  
## W = 1906, p-value = 0.4525  
## alternative hypothesis: true location shift is not equal to 0

We cannot reject the null hypothesis that there is no significant difference between the GPA’s of students who went to college bound high schools and students that did not.

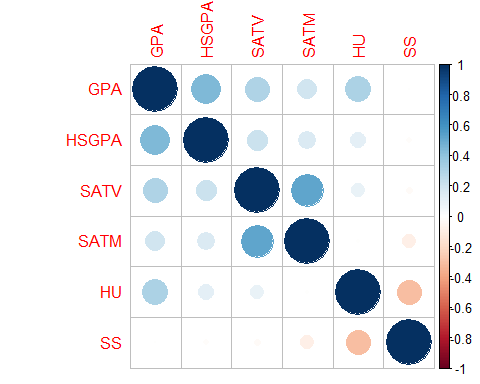
#### Now, we move on to examine the correlation of all of our numerical variables

cor(dataset[,-c(5,8,9,10)])

## GPA HSGPA SATV SATM HU  
## GPA 1.000000000 0.44611832 0.30286285 0.193261781 0.316481987  
## HSGPA 0.446118318 1.00000000 0.21032124 0.152839634 0.116031169  
## SATV 0.302862851 0.21032124 1.00000000 0.526943819 0.098748556  
## SATM 0.193261781 0.15283963 0.52694382 1.000000000 -0.009601549  
## HU 0.316481987 0.11603117 0.09874856 -0.009601549 1.000000000  
## SS -0.002320251 -0.01725443 -0.02646987 -0.087839974 -0.306607866  
## SS  
## GPA -0.002320251  
## HSGPA -0.017254428  
## SATV -0.026469871  
## SATM -0.087839974  
## HU -0.306607866  
## SS 1.000000000

With this simple line of code we are able to see that there is not even one strong correlation between our numerical variables (the highest is ~0.52)

##### Visualising the results :



##### We concluded that there is no important correlation between any of the numerical variables, so we will skip the part of plotting each pair of numerical variables.

#### We are finally ready to proceed with the part of machine learning in order to predict the students’ first year grade based on the other features.

# Part 3 : Machine - Learning and Model Selection

##### First, I will make a new categorical variable “grade1” replacing the numerical GPA in order to make predictions for a class (code shown below).

grade1 = as.factor(ifelse(GPA>=3.3,"High Pass","Pass"))  
dataset2 = data.frame(dataset[-c(1)],grade1)   
attach(dataset2)

Ideally, I would make the grade variable consisting of 3 levels, but since we only learned how to make predictions for categorical variables of 2 levels I will follow that. Therefore, I will only make 2 levels called “High Pass” and “Pass” to keep it in my reach.

#### I have decided to use two models for my predictions, a Decision Tree Classification and an SVM.

So, firstly we will divide our dataset into a train set and a test set as shown in the code below :

s\_size <- floor(0.8 \* nrow(dataset2))  
set.seed(1)  
train\_index <- sample(1:nrow(dataset2), size = s\_size)  
train <- dataset2[train\_index,]  
test <- dataset2[-train\_index,]

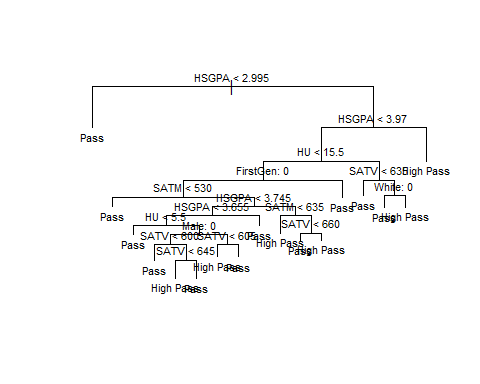
### Decision Tree Classification Models :

And now we are ready to train our classifier which is the tree in this case :

library(tree)  
tree.grade <- tree(grade1 ~ ., train)

We can see some first results :

##   
## Classification tree:  
## tree(formula = grade1 ~ ., data = train)  
## Variables actually used in tree construction:  
## [1] "HSGPA" "HU" "FirstGen" "SATM" "Male" "SATV" "White"   
## Number of terminal nodes: 17   
## Residual mean deviance: 0.7349 = 116.1 / 158   
## Misclassification error rate: 0.1657 = 29 / 175

Visualising the tree : 

Now, we will make some ***predictions*** on our test set, based on our tree model and we will make the ***Confusion Matrix***:

## grade.test  
## tree.pred1 High Pass Pass  
## High Pass 5 5  
## Pass 12 22

And now, we are ready to **evaluate** our model :

## Confusion Matrix and Statistics  
##   
## grade.test  
## tree.pred1 High Pass Pass  
## High Pass 5 5  
## Pass 12 22  
##   
## Accuracy : 0.6136   
## 95% CI : (0.455, 0.7564)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.5659   
##   
## Kappa : 0.1179   
##   
## Mcnemar's Test P-Value : 0.1456   
##   
## Sensitivity : 0.2941   
## Specificity : 0.8148   
## Pos Pred Value : 0.5000   
## Neg Pred Value : 0.6471   
## Prevalence : 0.3864   
## Detection Rate : 0.1136   
## Detection Prevalence : 0.2273   
## Balanced Accuracy : 0.5545   
##   
## 'Positive' Class : High Pass   
##

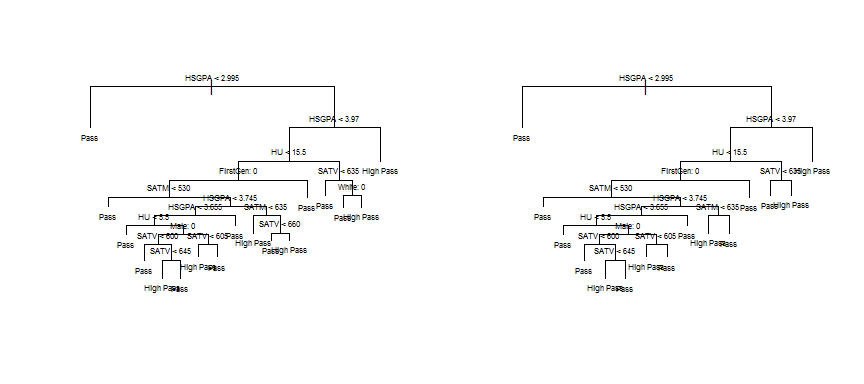
We get an accuracy of 0.6136

Although, even if we already have a good accuracy we will try to *optimize* the results by **pruning the tree** :

## $size  
## [1] 17 15 9 5 4 1  
##   
## $dev  
## [1] 60 56 58 57 75 78  
##   
## $k  
## [1] -Inf 1.000000 1.333333 1.750000 5.000000 5.666667  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

We can see that the lowest **dev** which represents error percentage, is at size 15.

Therefore, we will make a tree with size 7 to see if it has indeed better results and we will plot the two trees next to each other.



Now we will make **predictions** with our pruned tree and make the Confusion Matrix :

## grade.test  
## tree.pred2 High Pass Pass  
## High Pass 6 5  
## Pass 11 22

And again, **evaluate** our model :

## Confusion Matrix and Statistics  
##   
## grade.test  
## tree.pred2 High Pass Pass  
## High Pass 6 5  
## Pass 11 22  
##   
## Accuracy : 0.6364   
## 95% CI : (0.4777, 0.7759)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.4432   
##   
## Kappa : 0.1795   
##   
## Mcnemar's Test P-Value : 0.2113   
##   
## Sensitivity : 0.3529   
## Specificity : 0.8148   
## Pos Pred Value : 0.5455   
## Neg Pred Value : 0.6667   
## Prevalence : 0.3864   
## Detection Rate : 0.1364   
## Detection Prevalence : 0.2500   
## Balanced Accuracy : 0.5839   
##   
## 'Positive' Class : High Pass   
##

We can see that the accuracy slightly increased (0.6364)

### SVM Classification Models :

#### Linear :

To begin with, I will make a linear SVM model with *linear* kernel and cost 10 and see its summary :

##   
## Call:  
## svm(formula = grade1 ~ ., data = train, kernel = "linear", cost = 10)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 10   
##   
## Number of Support Vectors: 123  
##   
## ( 64 59 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## High Pass Pass

Now, I will use the SVM model to make **predictions** for the test set and make its Confusion Matrix :

## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26

We can see there are some misclassifications so we will quantify it by…

**Evaluating** the SVM model :

## Confusion Matrix and Statistics  
##   
## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26  
##   
## Accuracy : 0.7273   
## 95% CI : (0.5721, 0.8504)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.079595   
##   
## Kappa : 0.3545   
##   
## Mcnemar's Test P-Value : 0.009375   
##   
## Sensitivity : 0.3529   
## Specificity : 0.9630   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.7027   
## Prevalence : 0.3864   
## Detection Rate : 0.1364   
## Detection Prevalence : 0.1591   
## Balanced Accuracy : 0.6580   
##   
## 'Positive' Class : High Pass   
##

The accuracy is pretty good (0.7273), but we will try to optimize the SVM model.

I will now use the *tune function* in order to **optimize** my model by trying different cost values.

set.seed(123)  
tune.out=tune(svm , grade1~., data=train, kernel ="linear",  
 ranges =list(cost=c(0.01,0.1, 1, 10, 100, 1000)))  
summary(tune.out)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 1  
##   
## - best performance: 0.3147059   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 1e-02 0.3892157 0.1015289  
## 2 1e-01 0.3316993 0.1063111  
## 3 1e+00 0.3147059 0.1144781  
## 4 1e+01 0.3205882 0.1244546  
## 5 1e+02 0.3205882 0.1244546  
## 6 1e+03 0.3261438 0.1342108

Lowest error is for cost of 1.

Making the ***best*** linear model for *cost = 1* and seeing its summary :

##   
## Call:  
## svm(formula = grade1 ~ ., data = train, kernel = "linear", cost = 1)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 1   
##   
## Number of Support Vectors: 124  
##   
## ( 63 61 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## High Pass Pass

Now, I will use the best linear model to make predictions for the test set and make its Confusion Matrix :

**Evaluating** the best linear SVM model :

## Confusion Matrix and Statistics  
##   
## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26  
##   
## Accuracy : 0.7273   
## 95% CI : (0.5721, 0.8504)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.079595   
##   
## Kappa : 0.3545   
##   
## Mcnemar's Test P-Value : 0.009375   
##   
## Sensitivity : 0.3529   
## Specificity : 0.9630   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.7027   
## Prevalence : 0.3864   
## Detection Rate : 0.1364   
## Detection Prevalence : 0.1591   
## Balanced Accuracy : 0.6580   
##   
## 'Positive' Class : High Pass   
##

Accuracy : 0.7273

Since the linear kernel SVM model with *cost = 10* has the same accuracy (0.7727) with the other linear kernel SVM model with *cost = 1*, I will keep as the *best*, the one with the **lower cost**, which is the **“bestmodel”**.

#### Radial :

I will use the tune function again in order to find the best parameters (cost and gamma) for the SVM model with *radial* kernel :

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 0.1  
##   
## - best performance: 0.3091503   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-01 0.01 0.3892157 0.10152891  
## 2 1e+00 0.01 0.3558824 0.13226617  
## 3 1e+01 0.01 0.3147059 0.10510769  
## 4 1e+02 0.01 0.3771242 0.12227833  
## 5 1e+03 0.01 0.3490196 0.07614060  
## 6 1e-01 0.10 0.3892157 0.10152891  
## 7 1e+00 0.10 0.3091503 0.09792053  
## 8 1e+01 0.10 0.3614379 0.07938180  
## 9 1e+02 0.10 0.4415033 0.12183354  
## 10 1e+03 0.10 0.4411765 0.12883538  
## 11 1e-01 1.00 0.3892157 0.10152891  
## 12 1e+00 1.00 0.4006536 0.07378838  
## 13 1e+01 1.00 0.4284314 0.03597906  
## 14 1e+02 1.00 0.4284314 0.03597906  
## 15 1e+03 1.00 0.4284314 0.03597906  
## 16 1e-01 10.00 0.3892157 0.10152891  
## 17 1e+00 10.00 0.3892157 0.10152891  
## 18 1e+01 10.00 0.3892157 0.10152891  
## 19 1e+02 10.00 0.3892157 0.10152891  
## 20 1e+03 10.00 0.3892157 0.10152891

Best parameters : cost = 1, gamma = 0.1

Making the best SVM model with radial kernel and seeing its summary :

bestmodel1 <- svm(grade1 ~ .,data=train,kernel="radial",cost=1,gamma=0.1)  
summary(bestmodel1)

##   
## Call:  
## svm(formula = grade1 ~ ., data = train, kernel = "radial", cost = 1,   
## gamma = 0.1)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 133  
##   
## ( 70 63 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## High Pass Pass

I will now use the model for **predictions** and make the Confusion Matrix :

Again, evaluating the model :

## Confusion Matrix and Statistics  
##   
## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26  
##   
## Accuracy : 0.7273   
## 95% CI : (0.5721, 0.8504)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.079595   
##   
## Kappa : 0.3545   
##   
## Mcnemar's Test P-Value : 0.009375   
##   
## Sensitivity : 0.3529   
## Specificity : 0.9630   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.7027   
## Prevalence : 0.3864   
## Detection Rate : 0.1364   
## Detection Prevalence : 0.1591   
## Balanced Accuracy : 0.6580   
##   
## 'Positive' Class : High Pass   
##

Accuracy : 0.7273

#### Sigmoid :

I will use again the *tune* function in order to find the *best parameters* for the SVM model with *sigmoid* kernel :

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 100 0.01  
##   
## - best performance: 0.3147059   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-01 0.01 0.3892157 0.10152891  
## 2 1e+00 0.01 0.3892157 0.10152891  
## 3 1e+01 0.01 0.3316993 0.10631105  
## 4 1e+02 0.01 0.3147059 0.11447812  
## 5 1e+03 0.01 0.3313725 0.09856070  
## 6 1e-01 0.10 0.3892157 0.10152891  
## 7 1e+00 0.10 0.3369281 0.11309530  
## 8 1e+01 0.10 0.3839869 0.11997663  
## 9 1e+02 0.10 0.3728758 0.10347380  
## 10 1e+03 0.10 0.3784314 0.09203085  
## 11 1e-01 1.00 0.4352941 0.11824868  
## 12 1e+00 1.00 0.4000000 0.09287027  
## 13 1e+01 1.00 0.4401961 0.11523232  
## 14 1e+02 1.00 0.4513072 0.12090473  
## 15 1e+03 1.00 0.4513072 0.11509219  
## 16 1e-01 10.00 0.3954248 0.13212749  
## 17 1e+00 10.00 0.3934641 0.13997885  
## 18 1e+01 10.00 0.4450980 0.09856793  
## 19 1e+02 10.00 0.4277778 0.11019368  
## 20 1e+03 10.00 0.4277778 0.11019368

Best parameters : cost = 100, gamma = 0.01

Making the optimized sigmoid kernel SVM :

bestmodel2 <- svm(grade1 ~ .,data=train,kernel="sigmoid",cost=100,gamma=0.01)  
bestmodel2

##   
## Call:  
## svm(formula = grade1 ~ ., data = train, kernel = "sigmoid", cost = 100,   
## gamma = 0.01)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: sigmoid   
## cost: 100   
## coef.0: 0   
##   
## Number of Support Vectors: 125

Using the model to make **predictions** on the test set and making its Confusion Matrix :

## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26

**Evaluating** the model :

## Confusion Matrix and Statistics  
##   
## truth  
## predict High Pass Pass  
## High Pass 6 1  
## Pass 11 26  
##   
## Accuracy : 0.7273   
## 95% CI : (0.5721, 0.8504)  
## No Information Rate : 0.6136   
## P-Value [Acc > NIR] : 0.079595   
##   
## Kappa : 0.3545   
##   
## Mcnemar's Test P-Value : 0.009375   
##   
## Sensitivity : 0.3529   
## Specificity : 0.9630   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.7027   
## Prevalence : 0.3864   
## Detection Rate : 0.1364   
## Detection Prevalence : 0.1591   
## Balanced Accuracy : 0.6580   
##   
## 'Positive' Class : High Pass   
##

Accuracy : 0.7273 (same as the other 2 SVM’s)

From our results we can see that we have *equal accuracy in the optimized SVM models with linear, radial and sigmoid kernels*. If I had to choose one, I would choose the simplest one with the **lower cost**, which is the **linear** (cost = 1 vs cost = 100).

Eventually, the SVM was better than our tree classifier.

Therefore, my **conclusion** is that I choose the **Linear SVM classifier** in order to predict the grade of a student by their other features.

Για τη συγγραφή της συγκεκριμένης εργασίας σε Word χρησιμοποιήθηκε η πλατφόρμα R Markdown.