Logo, company name

Description automatically generated

**Data, Statistics and Probabilities**

**Programme Name: MSc Artificial Intelligence**

**Module Name and Code: Data, Statistics and Probabilities**

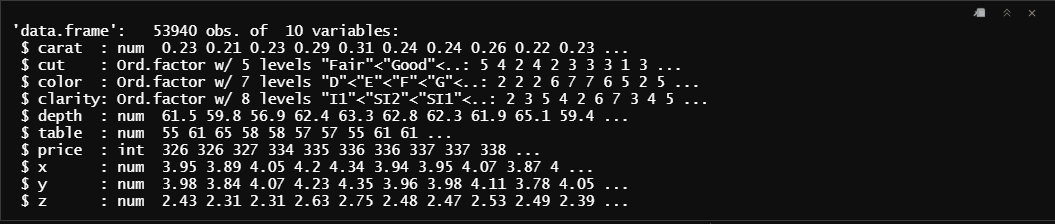
**Student ID Number: Pardalis Orestis**

**Lecturer’s Name: Skoularikis Kyriakos**

**Word Count: 5956**

1. VISUALIZE THE DATA

Το σύνολο δεδομένων που έχουμε να επεξεργαστούμε ονομάζονται diamonds, περιέχει 53940 παρατηρήσεις (διαμάντια) και μας παρέχει πληροφορίες για 10 μεταβλητές τους, δηλαδή χαρακτηριστικά, από τα οποία τα τρία αποτελούν ποιοτικές μεταβλητές και τα υπόλοιπα 7 αριθμητικές. Βασικές πληροφορίες διαφαίνονται εδώ:



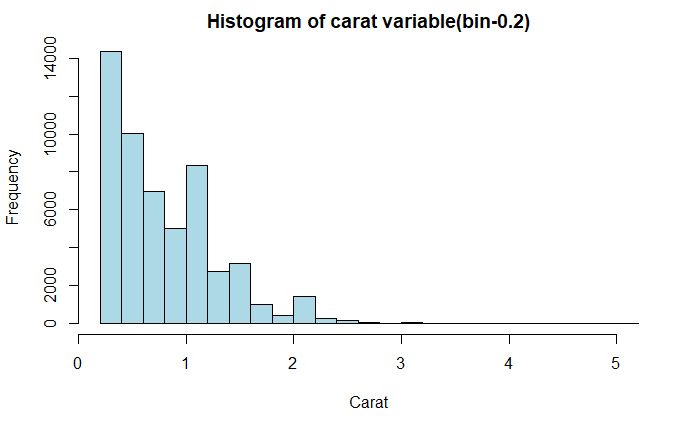
Αρχικά, όσον αφορά τον έλεγχο της κατανομής των αριθμητικών μεταβλητών του συνόλου δεδομένων μας, διαπιστώσαμε το εξής:

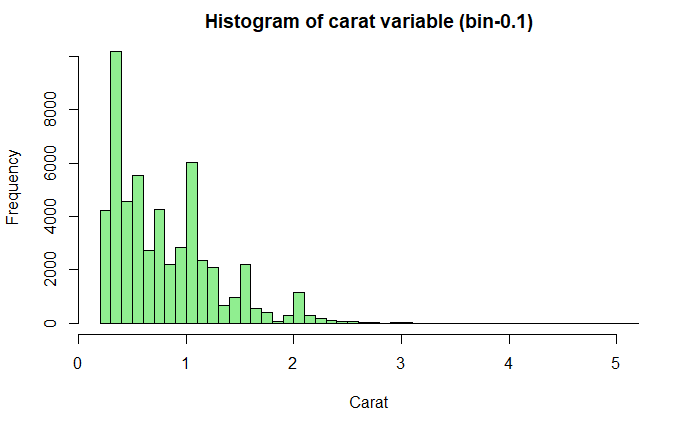
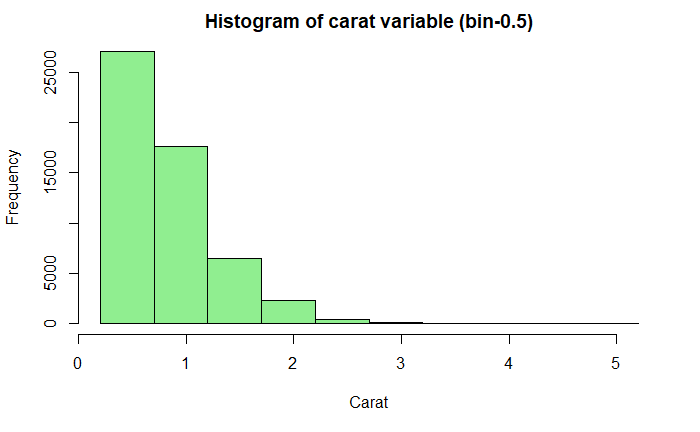
Καμία από τις μεταβλητές αυτές δεν ακολουθεί κανονική κατανομή.

Όλες οι μεταβλητές ακολουθούν πλατύκυρτη κατανομή με θετική, μάλιστα, ασσυμετρία. Αυτό διαπιστώθηκε από:

1. Έλεγχο υπόθεσης κανονικότητας (Anderson-Darling normality test)
2. Συντελεστές (στατιστικοί δείκτες) κύρτωσης και λοξότητας
3. Τις κατανομές των συχνοτήτων των μεταβλητών τόσο γραφικά με ιστογράμματα, όσο και μαθηματικά. Να σημειωθεί πως οι κατανομές συχνοτήτων υπολογίσθηκαν για τρία διαφορετικά εύρη bins.

Παρακάτω, ενδεικτικά παρουσιάζουμε κάποια γραφήματα κατανομών συχνοτήτων για διαφορετικά bins:

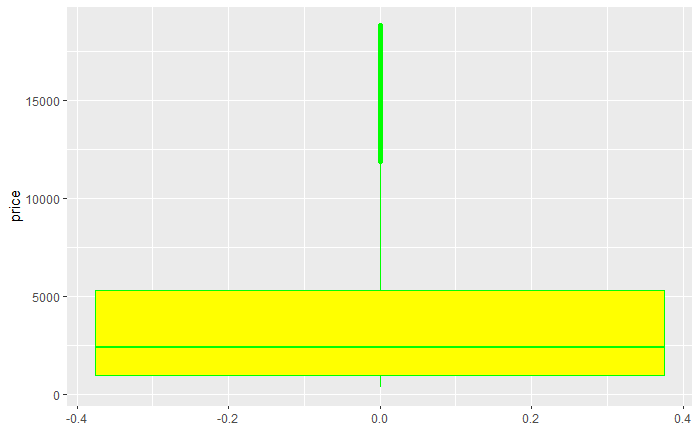


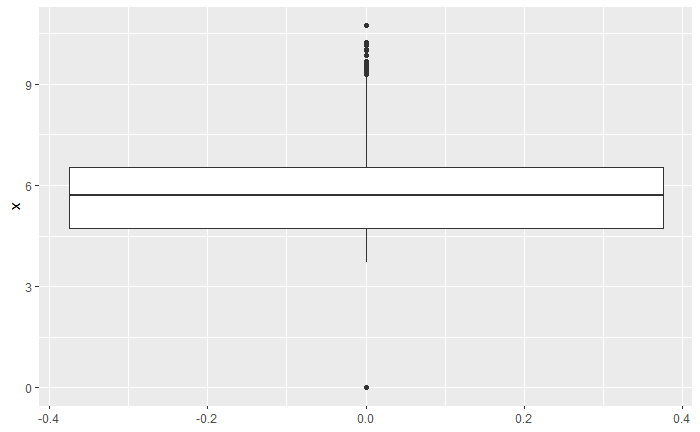


Όσον αφορά τα outliers, μελετώντας και αναπαριστώντας, αυτή τη φορά με θηκογράμματα (boxplots) και μαθηματικά, τις αριθμητικές μεταβλητές μας συμπεραίνουμε πως:

όλες περιέχουν τέτοιες ακραίες τιμές, άλλες λιγότερες σε ποσότητα, και άλλες περισσότερες.

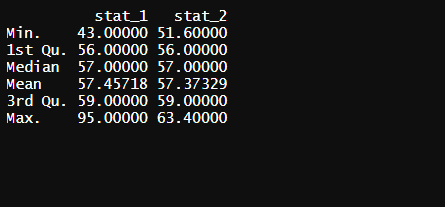
Ενδεικτικά, δύο τέτοια γραφήματα που παρουσιάζουν στατιστικά δύο μεταβλητές:



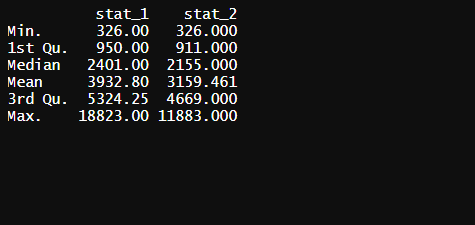


Εν συνεχεία, συγκρίναμε τα βασικά μέτρα κεντρικής τάσης και διασποράς στις δύο φάσεις – πριν και μετά την αφαίρεση των ακραίων τιμών (outliers). Στους δύο παρακάτω πίνακες παρατηρούμε τις διαφορές στους στατιστικούς δείκτες όσον αφορά τις μεταβλητές Price και Table.

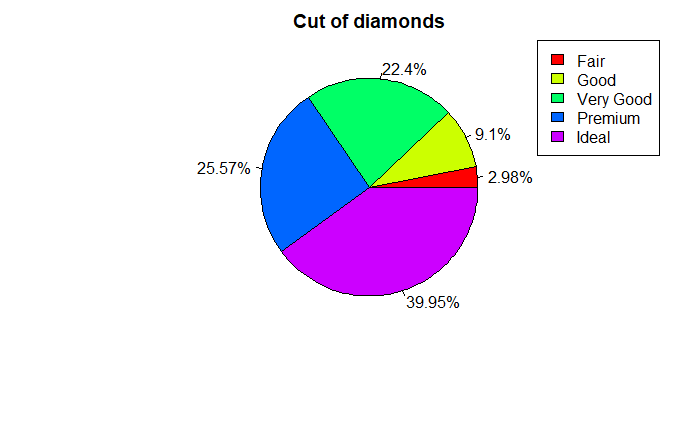
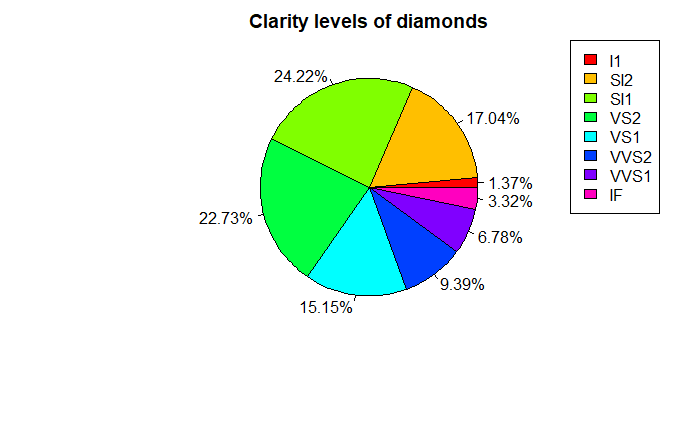
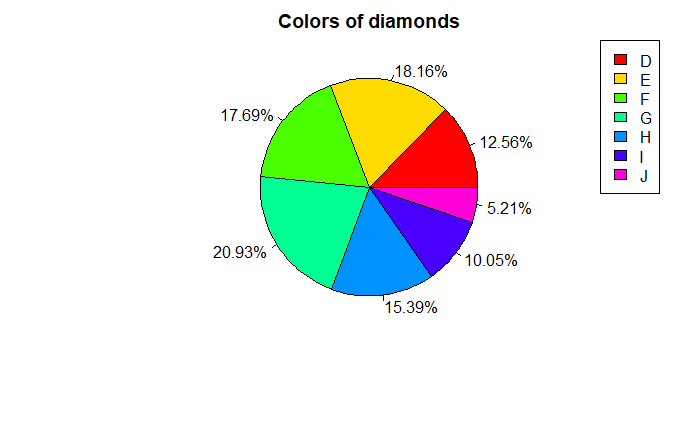
Table:



Price:



Γραφήματα πίτας (Piecharts) κατηγορικών μεταβλητών (αναγραφόμενα τα ποσοστά της κάθε κατηγορίας – λέβελ της μεταβλητής):

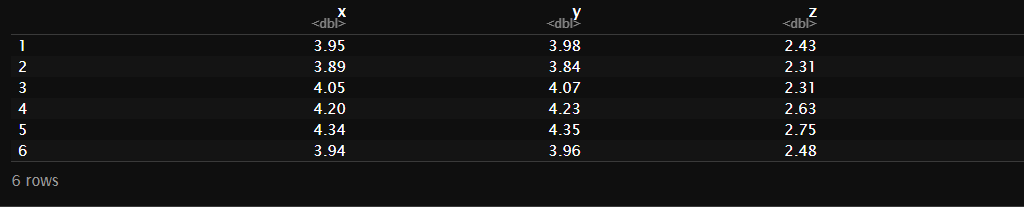


Όσον αφορά τις συσχετίσεις μεταξύ των μεταβλητών πραγματοποιήσαμε έλεγχο υπόθεσης με το συντελεστή συσχέτισης Pearson για τις αριθμητικές μεταβλητές και X2 έλεγχο για τις ποιοτικές.

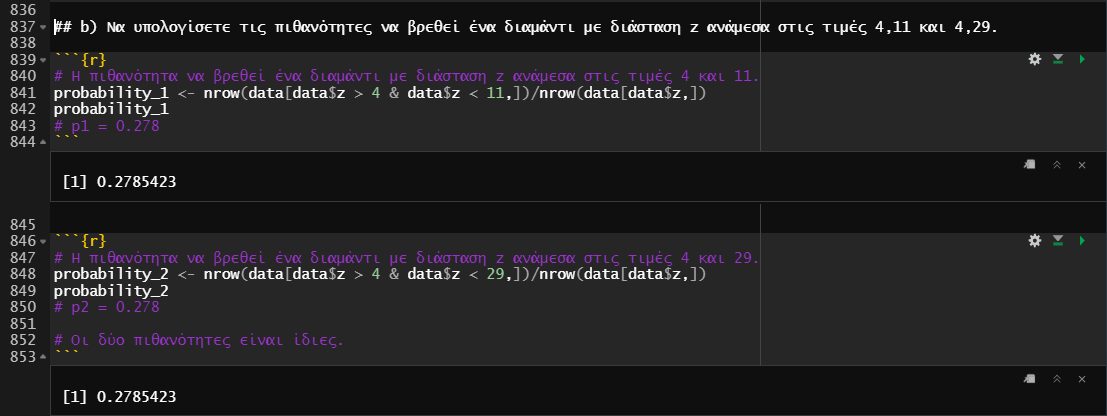
Το αξιοσημείωτο, εδώ, είναι πως μόνο οι μεταβλητές carat – price είναι πλήρως εξαρτημένες μεταξύ τους από τις αριθμητικές και οι υπόλοιπες έχουν χαμηλή έως καθόλου εξάρτηση, ενώ οι ποιοτικές μεταβλητές είναι κυρίως εξαρτημένες όλες μεταξύ τους.

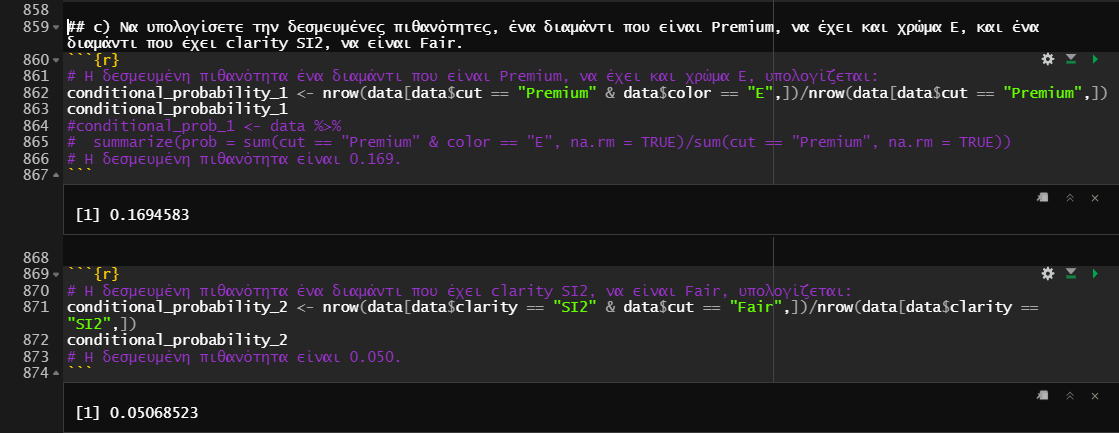
1. FREQUENT TABLE ΚΑΙ ΠΙΘΑΝΟΤΗΤΕΣ

Ο Frequent Table που κατασκευάσαμε για τις διαστάσεις των διαμαντιών (x,y,z) και είναι ανάλογος του ερωτήματος 1 παρουσιάζεται παρακάτω (για 6 διαστήματα):



Όσον αφορά τα ερωτήματα των πιθανοτήτων απλά μεταφέρουμε τους μαθηματικούς τύπους των ορισμών στα μέτρα του προγράμματος που δουλεύουμε R. Και έτσι, πήραμε:

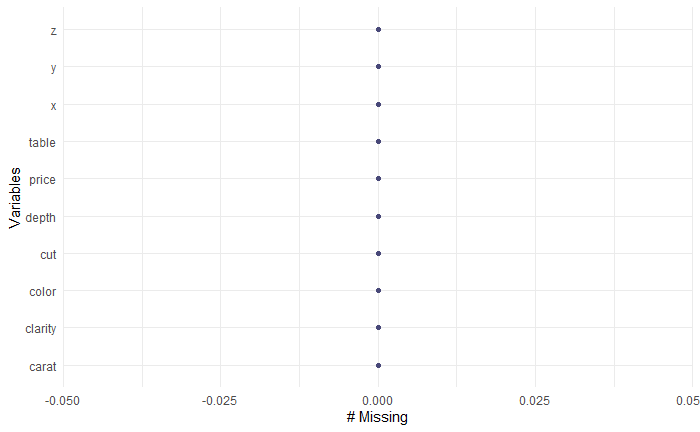
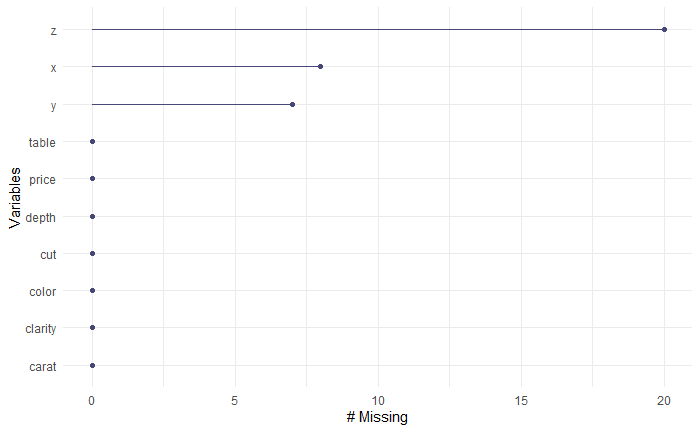




1. MISSING VALUES

Εδώ εφαρμόσαμε τις δύο μεθοδολογίες που ζητήθηκαν (mean & rf), ωστόσο η τεχνική του Random Forest Imputator απέτυχε από άποψη υπολογιστικού χρόνου, καθώς δοκιμάστηκαν και δύο διαφορετικές συναρτήσεις που δεν κατάφεραν να τρέξουν τόσο μεγάλο όγκο δεδομένων. Τα ΝΑ values υπήρχαν στις μεταβλητές που παρουσιάζουν τις διαστάσεις των διαμαντιών (x,y,z).

Και έτσι τα δεδομένα μας περιεχόμενα και μη NA τιμές διαφαίνονται όπως κάτωθι:



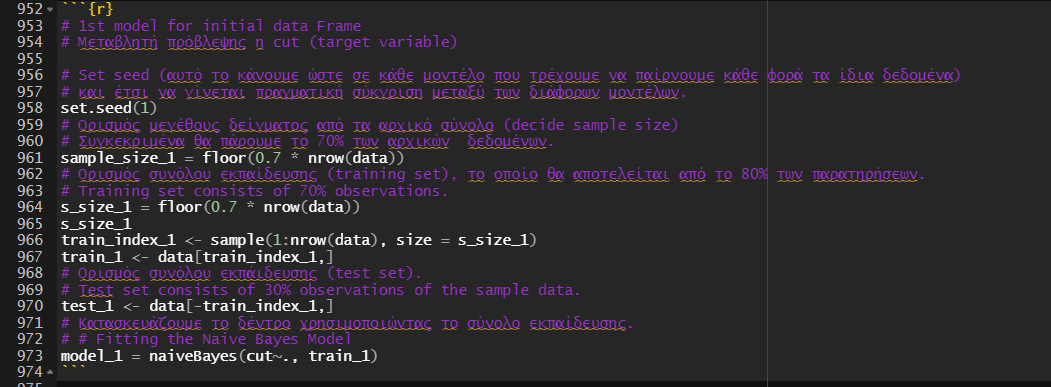
1. TRAIN AND TEST NAÏVE BAYES CLASSIFICATION

&

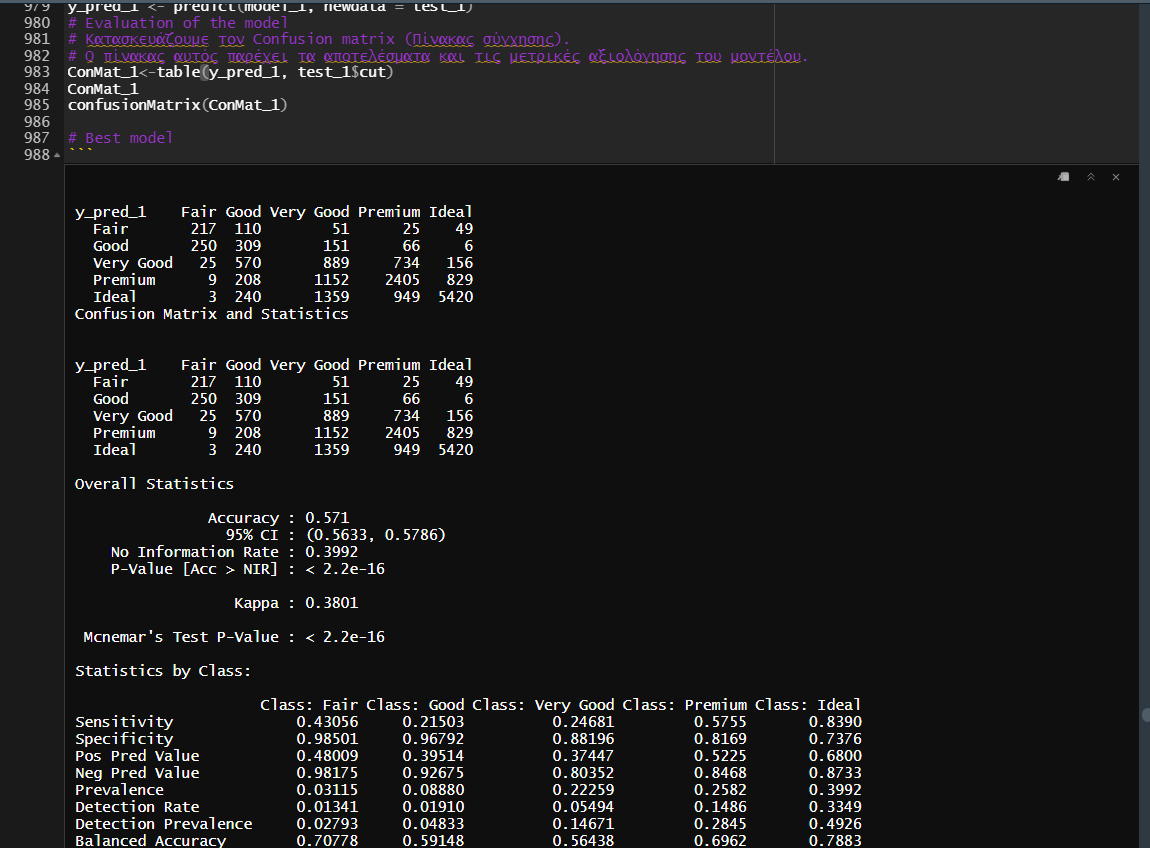
1. FUTURE SELECTION AND ADJUSTMENTS

Τέλος, για το 4 και το 5 ερώτημα εφαρμόσαμε διαδοχικά τον αλγόριθμο Naïve Bayes με κάποιες παραλλαγές κάθε φορά με σκοπό την δημιουργία του καλύτερου δυνατού μοντέλου που θα κάνει την βέλτιστη ταξινόμηση (classification) των διαμαντιών βάσει της κοπής του (target group: cut).

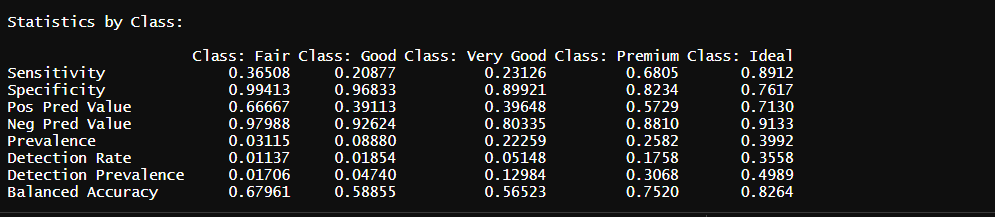
Ενδεικτικά, παρουσιάζουμε παρακάτω, ένα από τα έξι μοντέλα που κατασκευάστηκαν και τα αποτελέσματα του:



Και τα αποτελέσματα τον προβλέψεων (αξιολόγηση – πίνακας σύγχυσης):



Τέλος, το καλύτερο performance (μεγαλύτερη ακρίβεια και καλύτεροι δείκτες γενικά), το είχε το 4ο μοντέλο Naïve Bayes με τα εξής στατιστικά:



και

* Accuracy: 0.6129
* 95% CI: (0.6053, 0.6204)
* No information Rate: 0.3992

title: "Data, Statistics and Probabilities"

author: "Orestis Pardalis"

date: "13/5/2022"

output: word\_document

---

.

# Import Libraries

library(dplyr)

library(ggplot2)

library(nortest)

library(ggpubr)

library(DescriptiveStats.OBeu)

library(stats)

library(naniar)

library(missForest)

library(Hmisc)

View(diamonds)

# Import Data from ggplot library as DataFrame and see the basic information about it.

* **data = as.data.frame(diamonds)**
* **str(data)**

# 1. VISUALIZE THE DATA

## a) Να απαντήσετε αν το data set ακολουθεί Κανονική Κατανομή για τις μεταβλητές carat, price, cut, x,y,z Να δοκιμάσετε και Hypothesis Testing, αλλά και με Frequent Distributions. Στη 2η περίπτωση να δοκιμάσετε διαφορετικά μεγέθη για το bin. Όσες μεταβλητές δεν ακολουθούν Κανονική Κατανομή, να ονοματίσετε την Κατανομή τους.

# a) Check if the varibales: carat, price, x, y, z follow the normal distribution.

# 1st Check: wih Hypothesis Testing (Anderson-Darling normality test)

# Hypothesis Testing:

# Μηδενική Υπόθεση Ηο: The variable follows the normal distribution.

# Εναλλακτική Υπόθεση: The variable does not follow the normal distribution.

# 2nd Check: with Frequent Distribution Plots

# Feature carat:

* **normtest\_1 <- ad.test(data$carat)**
* **normtest\_1**
* **str(normtest\_1)**

# Οπτικός Έλεγχος (QQ-plot) #

* **qqplot\_1 <- ggqqplot(data = data$carat, conf.int = TRUE,**

**conf.int.level = 0.95, title = "Q-Q plot for Carat")**

* **qqplot\_1**

#Συντελεστής λοξότητας #

* **ds.skewness(data$carat)**
* **ds.kurtosis(data$carat)**

# συντελεστής κύρτωσης <3, συντελεστής λοξότητας θετικός και p-value<0.05, άρα

# H μεταβλητή ακολουθεί πλατύκυρτη κατανομή με θετική ασσυμετρία, δηλαδή μη - κανονική κατανομή. #

# Frequency distribution

# 1st (bin-0.2)

* **range(data$carat)**
* **breaks\_1 <- seq(0.2, 5.2, by=0.2)**
* **carat\_cut = cut(data$carat, breaks\_1, right=FALSE)**
* **carat\_freq\_1 = table(carat\_cut)**
* **cbind(carat\_freq\_1)**

# Histogram #

* **carat\_hist <- hist(data$carat, breaks = breaks\_1, main = "Histogram of carat variable(bin-0.2)", xlab = "Carat", col = "lightblue", freq = TRUE)**

# 2nd (bin-0.1)

* **breaks\_2 <- seq(0.2, 5.2, by=0.1)**
* **carat\_cut\_2 = cut(data$carat, breaks\_2, right=FALSE)**
* **carat\_freq\_2 = table(carat\_cut\_2)**
* **cbind(carat\_freq\_2)**

# Histogram #

* **carat\_hist <- hist(data$carat, breaks = breaks\_2, main = "Histogram of carat variable (bin-0.1)", xlab = "Carat", col = "lightgreen", freq = TRUE)**

# 3rd (bin-0.5)

* **breaks\_3 <- seq (0.2, 5.2, by=0.5)**
* **carat\_cut\_3 = cut (data$carat, breaks\_3, right=FALSE)**
* **carat\_freq\_3 = table(carat\_cut\_3)**
* **cbind(carat\_freq\_3)**

# Histogram #

* **carat\_hist <- hist(data$carat, breaks = breaks\_3, main = "Histogram of carat variable (bin-0.5)", xlab = "Carat", col = "lightgreen", freq = TRUE)**

# Feature price:

* **normtest\_2 <- ad.test(data$price)**
* **normtest\_2**
* **str(normtest\_2)**

# Οπτικός Έλεγχος (QQ-plot) #

* **qqplot\_2 <- ggqqplot (data = data$price, conf.int = TRUE,**

**conf.int.level = 0.95, title = "Q-Q plot for Price")**

* **qqplot\_2**

#Συντελεστής λοξότητας #

* **ds.skewness(data$price)**
* **ds.kurtosis(data$price)**

# συντελεστής κύρτωσης <3, συντελεστής λοξότητας θετικός και p-value<0.05, άρα

# H μεταβλητή ακολουθεί πλατύκυρτη κατανομή με θετική ασσυμετρία, δηλαδή μη - κανονική κατανομή. #

# Frequency distribution

# 1st (bin-10)

* **range(data$price)**
* **breaks\_1 <- seq(325, 18830, by=10)**
* **price\_cut\_1 = cut(data$price, breaks\_1, right=FALSE)**
* **price\_freq\_1 = table(price\_cut\_1)**
* **cbind(price\_freq\_1)**

# Histogram #

* **price\_hist <- hist(data$price, breaks = breaks\_1, main = "Histogram of Price variable(bin-10)", xlab = "Carat", col = "lightblue", freq = TRUE)**

# 2nd (bin-100)

* **breaks\_2 <- seq(325, 18830, by=100)**
* **price\_cut\_2 = cut(data$price, breaks\_2, right=FALSE)**
* **price\_freq\_2 = table(price\_cut\_2)**
* **cbind(price\_freq\_2)**

# Histogram #

**price\_hist <- hist(data$price, breaks = breaks\_1, main = "Histogram of Price variable(bin-100)",xlab = "Carat", col = "lightgreen", freq = TRUE)**

# 3rd (bin-500)

* **breaks\_3 <- seq(325, 18830, by=500)**
* **price\_cut\_3 = cut(data$price, breaks\_3, right=FALSE)**
* **price\_freq\_3 = table(price\_cut\_3)**
* **cbind(price\_freq\_3)**

# Histogram #

* **price\_hist <- hist(data$price, breaks = breaks\_3, main = "Histogram of Price variable(bin-500)", xlab = "Carat", col = "orange", freq = TRUE)**

# Feature x:

* **normtest\_3 <- ad.test(data$x)**
* **normtest\_3**
* **str(normtest\_3)**

# Οπτικός Έλεγχος (QQ-plot) #

* **qqplot\_3 <- ggqqplot(data = data$x, conf.int = TRUE,**

**conf.int.level = 0.95, title = "Q-Q plot for variable x")**

* **qqplot\_3**

#Συντελεστής λοξότητας #

* **ds.skewness(data$x)**
* **ds.kurtosis(data$x)**

# συντελεστής κύρτωσης <3, συντελεστής λοξότητας θετικός και p-value<0.05, άρα

# H μεταβλητή ακολουθεί πλατύκυρτη κατανομή με θετική ασσυμετρία, δηλαδή μη - κανονική κατανομή. #

# Frequency distribution

# 1st (bin-0.1)

* **range(data$x)**
* **breaks\_1 <- seq(0, 11, by=0.1)**
* **x\_cut\_1 = cut(data$x, breaks\_1, right=FALSE)**
* **x\_freq\_1 = table(x\_cut\_1)**
* **cbind(x\_freq\_1)**

# Histogram #

* **x\_hist <- hist(data$x, breaks = breaks\_1, main = "Histogram of x variable(bin-0.1)",**

**xlab = "X", col = "lightblue", freq = TRUE**)

# 2nd (bin-1)

* **breaks\_2 <- seq(0, 11, by=1)**
* **x\_cut\_2 = cut(data$x, breaks\_2, right=FALSE)**
* **x\_freq\_2 = table(x\_cut\_2)**
* **cbind(x\_freq\_2)**

# Histogram #

* **x\_hist <- hist(data$x, breaks = breaks\_2, main = "Histogram of x variable(bin-1)",**

**xlab = "X", col = "lightgreen", freq = TRUE)**

# 3rd (bin-0.5)

* **breaks\_3 <- seq(0, 11, by=0.5)**
* **x\_cut\_3 = cut(data$x, breaks\_3, right=FALSE)**
* **x\_freq\_3 = table(x\_cut\_3)**
* **cbind(x\_freq\_3)**

# Histogram #

* **x\_hist <- hist(data$x, breaks = breaks\_3, main = "Histogram of x variable(bin-0.5)",**

**xlab = "X", col = "orange", freq = TRUE)**

# Feature y:

* **normtest\_4 <- ad.test(data$y)**
* **normtest\_4**
* **str(normtest\_4)**

# Οπτικός Έλεγχος (QQ-plot) #

* **qqplot\_4 <- ggqqplot(data = data$y, conf.int = TRUE,**

**conf.int.level = 0.95, title = "Q-Q plot for variable y")**

* **qqplot\_4**

#Συντελεστής λοξότητας #

* **ds.skewness(data$y)**
* **ds.kurtosis(data$y)**

# συντελεστής κύρτωσης <3, συντελεστής λοξότητας θετικός και p-value<0.05, άρα

# H μεταβλητή ακολουθεί¨πλατύκυρτη κατανομή με θετική ασσυμετρία, δηλαδή μη - κανονική κατανομή. #

# Frequency distribution

# 1st (bin-1)

* **range(data$y)**
* **breaks\_1 <- seq(0, 60, by=1)**
* **y\_cut\_1 = cut(data$y, breaks\_1, right=FALSE)**
* **y\_freq\_1 = table(y\_cut\_1)**
* **cbind(y\_freq\_1)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_1, main = "Histogram of y variable(bin-1)",**

**xlab = "Y", col = "lightblue", freq = TRUE)**

# 2nd (bin-2)

* **breaks\_2 <- seq(0, 60, by=2)**
* **y\_cut\_2 = cut(data$y, breaks\_2, right=FALSE)**
* **y\_freq\_2 = table(y\_cut\_2)**
* **cbind(y\_freq\_2)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_2, main = "Histogram of y variable(bin-2)", xlab = "Y", col = "lightgreen", freq = TRUE)**

# 3rd (bin-5)

* **breaks\_3 <- seq(0, 60, by=5)**
* **y\_cut\_3 = cut(data$y, breaks\_3, right=FALSE)**
* **y\_freq\_3 = table(y\_cut\_3)**
* **cbind(y\_freq\_3)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_3, main = "Histogram of y variable(bin-5)", xlab = "Y", col = "orange", freq = TRUE)**

# Feature z:

* **normtest\_5 <- ad.test(data$z)**
* **normtest\_5**
* **str(normtest\_5)**

# Οπτικός Έλεγχος (QQ-plot) #

* **qqplot\_5 <- ggqqplot(data = data$z, conf.int = TRUE,**

**conf.int.level = 0.95, title = "Q-Q plot for variable z")**

* **qqplot\_5**

#Συντελεστής λοξότητας #

* **ds.skewness(data$z)**
* **ds.kurtosis(data$z)**

# συντελεστής κύρτωσης <3, συντελεστής λοξότητας θετικός και p-value<0.05, άρα

# H μεταβλητή ακολουθεί¨πλατύκυρτη κατανομή με θετική ασσυμετρία, δηλαδή μη - κανονική κατανομή. #

# Frequency distribution

# 1st (bin-1)

* **range(data$y)**
* **breaks\_1 <- seq(0, 60, by=1)**
* **y\_cut\_1 = cut(data$y, breaks\_1, right=FALSE)**
* **y\_freq\_1 = table(y\_cut\_1)**
* **cbind(y\_freq\_1)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_1, main = "Histogram of y variable(bin-1)", xlab = "Y", col = "lightblue", freq = TRUE)**

# 2nd (bin-2)

* **breaks\_2 <- seq(0, 60, by=2)**
* **y\_cut\_2 = cut(data$y, breaks\_2, right=FALSE)**
* **y\_freq\_2 = table(y\_cut\_2)**
* **cbind(y\_freq\_2)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_2, main = "Histogram of y variable(bin-2)",**

**xlab = "Y", col = "lightgreen", freq = TRUE)**

# 3rd (bin-5)

* **breaks\_3 <- seq(0, 60, by=5)**
* **y\_cut\_3 = cut(data$y, breaks\_3, right=FALSE)**
* **y\_freq\_3 = table(y\_cut\_3)**
* **cbind(y\_freq\_3)**

# Histogram #

* **y\_hist <- hist(data$y, breaks = breaks\_3, main = "Histogram of y variable(bin-5)",**

**xlab = "Y", col = "orange", freq = TRUE)**

# Frequency distribution

# 1st (bin-0.5)

* **range(data$z)**
* **breaks\_1 <- seq(0, 32, by=0.5)**
* **z\_cut\_1 = cut(data$z, breaks\_1, right=FALSE)**
* **z\_freq\_1 = table(z\_cut\_1)**

cbind(z\_freq\_1)

# Histogram #

* **z\_hist <- hist(data$z, breaks = breaks\_1, main = "Histogram of z variable(bin-0.5)", xlab = "Z", col = "lightblue", freq = TRUE)**

# 2nd (bin-1)

* **breaks\_2 <- seq(0, 60, by=1)**
* **z\_cut\_2 = cut(data$z, breaks\_2, right=FALSE)**
* **z\_freq\_2 = table(z\_cut\_2)**
* **cbind(z\_freq\_2)**

# Histogram #

* **z\_hist <- hist(data$z, breaks = breaks\_2, main = "Histogram of z variable(bin-1)", xlab = "Z", col = "lightgreen", freq = TRUE)**

# 3rd (bin-2)

* **breaks\_3 <- seq(0, 60, by=2)**
* **z\_cut\_3 = cut(data$z, breaks\_3, right=FALSE)**
* **z\_freq\_3 = table(z\_cut\_3)**
* **cbind(z\_freq\_3)**

# Histogram #

* **z\_hist <- hist(data$z, breaks = breaks\_3, main = "Histogram of z variable(bin-2)", xlab = "Z", col = "orange", freq = TRUE)**

## b) Να απαντήσετε αν υπάρχουν outliers ή extreme values στο data set, στις ποσοτικές μεταβλητές και αν ναι να συγκρίνετε πόσο επηρεάζουν τα βασικά μέτρα θέσης και διασποράς για τα αντίστοιχα ποσοτικά features.

* str(data)

# Numerical variables: carat, depth, table, price, x, y, z (6).

# For outliers exploration we are going to make a boxplot and a math-numeric analysis.

# Carat

# Boxplot

* **ggplot(data = data, aes(y = carat)) +**

**geom\_boxplot(color = "orange", fill = "green")**

# Compare basic stats with and without ouliers

* **stat\_1 = summary(data$carat)**
* **Q1 <- quantile(data$carat, .25)**
* **Q3 <- quantile(data$carat, .75)**
* **IQR <- IQR(data$carat)**
* **no\_outliers <- subset(data, data$carat > (Q1 - 1.5\*IQR) & data$carat < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$carat)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# Depth

# Boxplot

* **ggplot(data = data, aes(y = depth)) +**

**geom\_boxplot(color = "blue", fill = "white")**

# Compare basic stats with and without outliers

* **stat\_1 = summary(data$depth)**
* **Q1 <- quantile(data$depth, .25)**
* **Q3 <- quantile(data$depth, .75)**
* **IQR <- IQR(data$depth)**
* **no\_outliers <- subset(data, data$depth > (Q1 - 1.5\*IQR) & data$depth < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$depth)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# Table

# Boxplot

* **ggplot(data = data, aes(y = table)) +**

**geom\_boxplot(color = "grey", fill = "red")**

# Compare basic stats with and without outliers

* **stat\_1 = summary(data$table)**
* **Q1 <- quantile(data$table, .25)**
* **Q3 <- quantile(data$table, .75)**
* **IQR <- IQR(data$table)**
* **no\_outliers <- subset(data, data$table > (Q1 - 1.5\*IQR) & data$table < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$table)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# Price

# Boxplot

* **ggplot(data = data, aes(y = price)) +**

**geom\_boxplot(color = "green", fill = "yellow")**

# Compare basic stats with and without outliers

* **stat\_1 = summary(data$price)**
* **Q1 <- quantile(data$price, .25)**
* **Q3 <- quantile(data$price, .75)**
* **IQR <- IQR(data$price)**
* **no\_outliers <- subset(data, data$price > (Q1 - 1.5\*IQR) & data$price < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$price)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# X variable

# Boxplot

* **ggplot(data = data, aes(y = x)) +**

**geom\_boxplot()**

# Compare basic stats with and without outliers

* **stat\_1 = summary(data$x)**
* **Q1 <- quantile(data$x, .25)**
* **Q3 <- quantile(data$x, .75)**
* **IQR <- IQR(data$x)**
* **no\_outliers <- subset(data, data$x > (Q1 - 1.5\*IQR) & data$x < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$x)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# Y variable

# Boxplot

* **ggplot(data = data, aes(y = y)) +**

**geom\_boxplot()**

# Compare basic stats with and without ouliers

* **stat\_1 = summary(data$y)**
* **Q1 <- quantile(data$y, .25)**
* **Q3 <- quantile(data$y, .75)**
* **IQR <- IQR(data$y)**
* **no\_outliers <- subset(data, data$y > (Q1 - 1.5\*IQR) & data$y < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$y)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

# Z variable

# Boxplot

* **ggplot(data = data, aes(y = z)) +**

**geom\_boxplot()**

# Compare basic stats with and without ouliers

* **stat\_1 = summary(data$z)**
* **Q1 <- quantile(data$z, .25)**
* **Q3 <- quantile(data$z, .75)**
* **IQR <- IQR(data$z)**
* **no\_outliers <- subset(data, data$z > (Q1 - 1.5\*IQR) & data$z < (Q3 + 1.5\*IQR))**
* **stat\_2 = summary(no\_outliers$z)**

# Combine them to a matrix.

* **cbind(stat\_1, stat\_2)**

## c)Να αναπαραστήσετε σε Pie chart, το ποσοστό των κλάσεων στα features που έχουν κατηγορηματικά δεδομένα.

# We have 3 categorical features to examine: cut, color and clarity.

# First, we construct the frequency tables.

# Then, we use them to represent three Piecharts, one for each categorical feature.

# Cut

# Πίνακας συχνοτήτων.

* **freq\_cut <- table(data$cut)**
* **freq\_cut\_df <- as.data.frame(freq\_cut)**
* **colnames(freq\_cut\_df) <- c("Levels","Frequency")**
* **rel\_freq\_cut <- prop.table(freq\_cut)**
* **rel\_freq\_cut\_df <- as.data.frame(rel\_freq\_cut)**
* **colnames(rel\_freq\_cut\_df ) = c("Levels"," Relative Frequency")**
* **cut\_df = cbind(freq\_cut\_df,rel\_freq\_cut\_df)**
* **cut\_df**

# Ραβδόγραμμα

* **freq\_cut\_barplot <- barplot(freq\_cut, main="Cut", xlab="Cut levels of diamonds",ylab = "Frequency", horiz = FALSE, cex.names = 0.8, col = "green")**

# Γράφημα Πίτας (piechart)

* **piepercent <- paste(round(100\*rel\_freq\_cut,2),"%", sep="")**
* **freq\_cut\_pie <- pie(freq\_cut, labels=piepercent, col=rainbow(length(freq\_cut)),**

**main="Cut of diamonds")**

* **legend(x="topright",legend = names(freq\_cut),fill = rainbow(length(freq\_cut)))**

# Color

# Πίνακας συχνοτήτων.

* **freq\_color <- table(data$color)**
* **freq\_color\_df <- as.data.frame(freq\_color)**
* **colnames(freq\_color\_df) <- c("Colors","Frequency")**
* **rel\_freq\_color <- prop.table(freq\_color)**
* **rel\_freq\_color\_df <- as.data.frame(rel\_freq\_color)**
* **colnames(rel\_freq\_color\_df ) = c("Colors"," Relative Frequency")**
* **color\_df = cbind(freq\_color\_df,rel\_freq\_color\_df)**
* **color\_df**

# Ραβδόγραμμα

* **freq\_color\_barplot <- barplot(freq\_color, main="Color", xlab="Colors of diamonds",**  **ylab = "Frequency", horiz = FALSE, cex.names = 0.8, col = "lightblue")**

# Γράφημα Πίτας (piechart)

* **piepercent <- paste(round(100\*rel\_freq\_color,2),"%", sep="")**
* **freq\_color\_pie <- pie(freq\_color, labels=piepercent, col=rainbow(length(freq\_color)),main="Colors of diamonds")**
* **legend(x="topright",legend = names(freq\_color),fill = rainbow(length(freq\_color)))**

# Clarity

# Πίνακας συχνοτήτων.

* **freq\_clarity <- table(data$clarity)**
* **freq\_clarity\_df <- as.data.frame(freq\_clarity)**
* **colnames(freq\_clarity\_df) <- c("Levels","Frequency")**
* **rel\_freq\_clarity <- prop.table(freq\_clarity)**
* **rel\_freq\_clarity\_df <- as.data.frame(rel\_freq\_clarity)**
* **colnames(rel\_freq\_clarity\_df ) = c("Levels"," Relative Frequency")**
* **clarity\_df = cbind(freq\_clarity\_df,rel\_freq\_clarity\_df)**
* **clarity\_df**

# Ραβδόγραμμα

* **freq\_clarity\_barplot <- barplot(freq\_clarity, main="Color", xlab="Clarity levels of diamonds", ylab = "Frequency", horiz = FALSE, cex.names = 0.8, col = "orange")**

# Γράφημα Πίτας (piechart)

* **piepercent <- paste(round(100\*rel\_freq\_clarity,2),"%", sep="")**
* **freq\_clarity\_pie <- pie(freq\_clarity, labels=piepercent, col=rainbow(length(freq\_clarity)),**

**main="Clarity levels of diamonds")**

* **legend(x="topright",legend = names(freq\_clarity),fill = rainbow(length(freq\_clarity)))**

## d) Να παρατηρήσετε συσχετίσεις μεταξύ των features και να τις ονοματίσετε σε γραφικές παραστάσεις.

# We check for correlations between variables with Pearson for numerical data

# and X - squared test for categorical variables.

# Numerical data

# carat - depth

* **ggscatter(data, x = 'carat', y = 'depth',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Carat values", ylab = "Depth")**

# Συντελεστής Pearson R = 0.028, οι μεταβλητές είναι (σχεδόν) ανεξάρτητες.

# carat - table

* **ggscatter(data, x = 'carat', y = 'table',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Carat values", ylab = "Table")**

# Συντελεστής Pearson R = 0.18, οι μεταβλητές είναι εξαιρετικά χαμηλή εξάρτηση.

# carat - price

* **ggscatter(data, x = 'carat', y = 'price',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Carat values", ylab = "Price")**

# Συντελεστής Pearson R = 0.92, οι μεταβλητές είναι σχεδόν πλήρως εξαρτημένα.

# depth - table

* **ggscatter(data, x = 'depth', y = 'table',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Depth", ylab = "Table")**

# Συντελεστής Pearson R = -0.3, οι μεταβλητές έχουν χαμηλή αρνητική εξάρτηση.

# depth - price

* **ggscatter(data, x = 'depth', y = 'price',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Depth", ylab = "Price")**

# Συντελεστής Pearson R = -0.011, οι μεταβλητές είναι (σχεδόν) ανεξάρτητες.

# table - price

* **ggscatter(data, x = 'table', y = 'price',**

**add = "reg.line", conf.int = TRUE,**

**cor.coef = TRUE, cor.method = "pearson",**

**xlab = "Table", ylab = "Price")**

# Συντελεστής Pearson R = 0.13, οι μεταβλητές έχουν πολύ χαμηλή εξάρτηση.

# Categorical Features (X^2 Test)

# Μηδενική υπόθεση Η\_ο: τα 2 χαρακτηριστικά που μελετώνται είναι ανεξάρτητα.

# Εναλλακτική υπόθεση Η\_1: τα 2 χαρακτηριστικά είναι εξαρτημένα.

# cut - color #

* **cut\_color\_chitest <- chisq.test(table(data$cut, data$color))**
* **cut\_color\_chitest**
* **str(cut\_color\_chitest)**

# p-value < 0.05, επομένως απορρίπτουμε την μηδενική υπόθεση και είναι εξαρτημένες μεταβλητές.

# cut - clarity #

* **cut\_clarity\_chitest <- chisq.test(table(data$cut, data$clarity))**
* **cut\_clarity\_chitest**
* **str(cut\_clarity\_chitest)**

# p-value < 0.05, επομένως απορρίπτουμε την μηδενική υπόθεση και είναι εξαρτημένες μεταβλητές.

# color - clarity #

* **color\_clarity\_chitest <- chisq.test(table(data$color, data$clarity))**
* **color\_clarity\_chitest**
* **str(color\_clarity\_chitest)**

# p-value < 0.05, επομένως απορρίπτουμε την μηδενική υπόθεση και είναι εξαρτημένες μεταβλητές.

# Correlations between numerical and categorical data.

# Ανάλυση διασποράς (ANOVA)

# Θα εξετάσουμε αν η διασπορά μιας ποσοτικής μεταβλητής για τις διάφορες κατηγορίες

# (περισσότερες από 2) μιας ποιοτικής είναι ίδια ή όχι

# (μόνο για την cut που μας ενδιαφέρει στα μοντέλα μετέπειτα).

# Η\_ο: διασπορές ίδιες.

# Η\_1: διασπορές άνισες.

# cut – carat

* **cut\_carat\_var.test<-var.test(x=data[data$cut=="Fair","carat"],**

**y=data[data$cut=="Good","carat"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_carat\_var.test**
* **cut\_carat\_var.test2<-var.test(x=data[data$cut=="Good","carat"],**

**y=data[data$cut=="Very Good","carat"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_carat\_var.test2**
* **cut\_carat\_var.test3<-var.test(x=data[data$cut=="Very Good","carat"],**

**y=data[data$cut=="Premium","carat"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_carat\_var.test3**
* **cut\_carat\_var.test4<-var.test(x=data[data$cut=="Premium","carat"],**

**y=data[data$cut=="Ideal","carat"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_carat\_var.test4**

# carat - cut

# Οι διασπορές είναι ίσες (p-value>0.05) μόνο για τις κατηγορίες good & very good (υπάρχει μερική εξάρτηση)

# , στα υπόλοιπα είναι διάφορες.

# cut – depth

* **cut\_depth\_var.test<-var.test(x=data[data$cut=="Fair","depth"],**

**y=data[data$cut=="Good","depth"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_depth\_var.test**
* **cut\_depth\_var.test2<-var.test(x=data[data$cut=="Good","depth"],**

**y=data[data$cut=="Very Good","depth"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_depth\_var.test2**
* **cut\_depth\_var.test3<-var.test(x=data[data$cut=="Very Good","depth"],**

**y=data[data$cut=="Premium","depth"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_depth\_var.test3**
* **cut\_depth\_var.test4<-var.test(x=data[data$cut=="Premium","depth"],**

**y=data[data$cut=="Ideal","depth"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_depth\_var.test4**

# Οι διασπορές είναι σε όλες τις περιπτώσεις άνισες (p-value<0.05), δεν έχουμε κάποια εξάρτηση.

# cut – table

* **cut\_table\_var.test<-var.test(x=data[data$cut=="Fair","table"],**

**y=data[data$cut=="Good","table"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_table\_var.test**
* **cut\_table\_var.test2<-var.test(x=data[data$cut=="Good","table"],**

**y=data[data$cut=="Very Good","table"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_table\_var.test2**
* **cut\_table\_var.test3<-var.test(x=data[data$cut=="Very Good","table"],**

**y=data[data$cut=="Premium","table"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_table\_var.test3**
* **cut\_table\_var.test4<-var.test(x=data[data$cut=="Premium","table"],**

**y=data[data$cut=="Ideal","table"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_table\_var.test4**

# Οι διασπορές είναι σε όλες τις περιπτώσεις άνισες (p-value<0.05), δεν έχουμε κάποια εξάρτηση.

# cut – price

* **cut\_price\_var.test<-var.test(x=data[data$cut=="Fair","price"],**

**y=data[data$cut=="Good","price"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_price\_var.test**
* **cut\_price\_var.test2<-var.test(x=data[data$cut=="Good","price"],**

**y=data[data$cut=="Very Good","price"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_price\_var.test2**
* **cut\_price\_var.test3<-var.test(x=data[data$cut=="Very Good","price"],**

**y=data[data$cut=="Premium","price"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_price\_var.test3**
* **cut\_price\_var.test4<-var.test(x=data[data$cut=="Premium","price"],**

**y=data[data$cut=="Ideal","price"],**

**alternative = "two.sided",conf.level = 0.95)**

* **cut\_price\_var.test4**

# Οι διασπορές είναι σε όλες τις περιπτώσεις άνισες (p-value<0.05),

# εκτός από τις κατηγορίες Fair με Price που υπάρχει εξάρτηση (διασπορές ίσες).

# 2. FREQUENT TABLE ΚΑΙ ΠΙΘΑΝΟΤΗΤΕΣ

## a) Να κατασκευάσετε Frequent Table για τις διαστάσεις των διαμαντιών (x,y,z).

# Frequent table for diamonds dimensions.

# First create a new dataset only for the dimensions of the diamonds.

* df\_dimensions = cbind(data$x, data$y, data$z)
* df\_dimensions = as.data.frame(df\_dimensions)
* df\_dimns <- df\_dimensions %>% rename( x = V1, y = V2, z = V3)
* head(df\_dimns)
* str(df\_dimns)

# Frequency table for x

* **range(data$x)**
* **breaks\_1 <- seq(0, 12, by=2)**
* **x\_cut\_1 = cut(data$x, breaks\_1, right=FALSE)**
* **x\_freq\_1 = table(x\_cut\_1)**
* **cbind(x\_freq\_1)**

# Frequency table for y

* **range(data$y)**
* **breaks\_1 <- seq(0, 60, by=10)**
* **y\_cut\_1 = cut(data$y, breaks\_1, right=FALSE)**
* **y\_freq\_1 = table(y\_cut\_1)**
* **cbind(y\_freq\_1)**

# Frequency table for z

* **range(data$z)**
* **breaks\_1 <- seq(0, 30, by=5)**
* **z\_cut\_1 = cut(data$z, breaks\_1, right=FALSE)**
* **z\_freq\_1 = table(z\_cut\_1)**
* **cbind(z\_freq\_1)**

# To combine all 3 dimensions in one frequency table:

* **cbind(x\_freq\_1, y\_freq\_1, z\_freq\_1)**

# The best thing to do is to remain to our best results for feature frequency from exercise 1a) - (xyz). This is not optimal.

## b) Να υπολογίσετε τις πιθανότητες να βρεθεί ένα διαμάντι με διάσταση z ανάμεσα στις τιμές 4,11 και 4,29.

# Η πιθανότητα να βρεθεί ένα διαμάντι με διάσταση z ανάμεσα στις τιμές 4 και 11.

* **probability\_1 <- nrow(data[data$z > 4 & data$z < 11,])/nrow(data[data$z,])**
* **probability\_1**

# p1 = 0.278

# Η πιθανότητα να βρεθεί ένα διαμάντι με διάσταση z ανάμεσα στις τιμές 4 και 29.

* **probability\_2 <- nrow(data[data$z > 4 & data$z < 29,])/nrow(data[data$z,])**
* **probability\_2**

# p2 = 0.278

# Οι δύο πιθανότητες είναι ίδιες.

## c) Να υπολογίσετε την δεσμευμένες πιθανότητες, ένα διαμάντι που είναι Premium, να έχει και χρώμα Ε, και ένα διαμάντι που έχει clarity SI2, να είναι Fair.

# Η δεσμευμένη πιθανότητα ένα διαμάντι που είναι Premium, να έχει και χρώμα Ε, υπολογίζεται:

* **conditional\_probability\_1 <- nrow(data[data$cut == “Premium” & data$color == “E”,])/nrow(data[data$cut == “Premium”,])**
* **conditional\_probability\_1**
* **conditional\_prob\_1 <- data %>%**
* **summarize(prob = sum(cut == "Premium" & color == "E", na.rm = TRUE)/sum(cut == "Premium", na.rm = TRUE))**

# Η δεσμευμένη πιθανότητα είναι 0.169.

# Η δεσμευμένη πιθανότητα ένα διαμάντι που έχει clarity SI2, να είναι Fair, υπολογίζεται:

* **conditional\_probability\_2 <- nrow(data[data$clarity == "SI2" & data$cut == "Fair",])/nrow(data[data$clarity == "SI2",])**
* **conditional\_probability\_2**

# Η δεσμευμένη πιθανότητα είναι 0.050.

# 3. MISSING VALUES

## Να βρείτε εάν υπάρχουν missing values ή μηδενικές τιμές (προφανώς σε features που δεν μπορούν να έχουν τιμή=0) και να το αντιμετωπίσετε. Να χρησιμοποιήσετε και RF, αλλά και την τεχνική με mean/επικρατούσα τιμή. Να αποθηκεύσετε το καινούργιο data set που θα προκύψει από κάθε διαφορετική επεξεργασία, σε νέα μεταβλητή, ώστε να είναι διαθέσιμα τρία dataset κατά τη διαδικασία του classification.

# Check for missing values with a plot (gg\_miss\_var function from library naniar)

# Check the ranges of the values for numerical variables.

# We can see that the variables for the dimensions of the dataset have zero values witch is FALSE, as they are diamonds.

# So, x,y and z variables will be replaced for our purpose.

* **range(data$carat)**
* **range(data$depth)**
* **range(data$table)**
* **range(data$price)**
* **range(data$x)**
* **range(data$y)**
* **range(data$z)**

# Check for other NA values graphically.

* **gg\_miss\_var(data)**

# Convert, first zero values to NA from columns x,y and z.

* **data2 = data**
* **data2$x[data2$x == 0] = NA**
* **data2$y[data2$y == 0] = NA**
* **data2$z[data2$z == 0] = NA**
* **gg\_miss\_var(data2)**

# Now, we are ready to implement the two methods for filling the NA's values.

# 1st method: Missing Value with Mean of the variable

* **data\_mean <- data2**
* **data\_mean$x <- with(data\_mean, impute(x, mean))**
* **data\_mean$y <- with(data\_mean, impute(y, mean))**
* **data\_mean$z <- with(data\_mean, impute(z, mean))**
* **gg\_miss\_var(data\_mean)**

# 2nd method: Missing Value with Meanb Imputations by randomForest

# We will use library missForest and its imputation as Random Forest's library imputation does not work for very large data as diamonds dataset.

# missForest is an implementation of random forest algorithm

* **data\_rf <- missForest(data2)**
* **gg\_miss\_var(data\_rf)**

# 4. TRAIN AND TEST NAÏVE BAYES CLASSIFICATION

## a) Να κάνετε random sample τα δεδομένα, ώστε να πάρετε ένα επαρκές δείγμα, κατά την κρίση σας, από τις περίπου 50000 παρατηρήσεις.

## b) Στη συνέχεια να κάνετε split το sample σας σε training και test set.

## c) Να εκπαιδεύσετε το μοντέλο σας και να το αξιολογήσετε με το test set, με Naïve Bayes classification και ως μεταβλητή στόχος την cut. Δηλαδή να μπορεί το μοντέλο μας να κατηγοριοποιεί, ένα διαμάντι με βάση κάποια χαρακτηριστικά που εσείς θα επιλέξετε.

## d) H διαδικασία να γίνει και για τα τρία data set.

## e) Να παρουσιάσετε τις μετρήσεις και για τα τρία data set.

```{r}

# 1st model for initial data Frame

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

set.seed(1)

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_1 = floor(0.7 \* nrow(data))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_1 = floor(0.7 \* nrow(data))**
* **s\_size\_1**
* **train\_index\_1 <- sample(1:nrow(data), size = s\_size\_1)**
* **train\_1 <- data[train\_index\_1,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_1 <- data[-train\_index\_1,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_1 = naiveBayes(cut~. , train)**

# Προβλέψεις

# Make predictions on test dataset.

* **y\_pred\_1 <- predict(model\_1, newdata = test\_1)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_1<-table(y\_pred\_1, test\_1$cut)**
* **ConMat\_1**
* **confusionMatrix(ConMat\_1)**

# Best model

# 2nd model for 2nd data Frame (where NAs are filled with the mean)

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_2 = floor(0.7 \* nrow(data\_mean))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_2 = floor(0.7 \* nrow(data\_mean))**
* **s\_size\_2**
* **train\_index\_2 <- sample(1:nrow(data\_mean), size = s\_size\_2)**
* **train\_2 <- data\_mean[train\_index\_2,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_2 <- data\_mean[-train\_index\_2,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_2 = naiveBayes(cut~. , train\_2)**

# Προβλέψεις

# Make predictions on test dataset.

* **y\_pred\_2 <- predict(model\_2, newdata = test\_2)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_2<-table(y\_pred\_2, test\_2$cut)**
* **ConMat\_2**
* **confusionMatrix(ConMat\_2)**

# 3rd model for 3rd data Frame (where NA's are filled with the Random Forest imputation)

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_3 = floor(0.7 \* nrow(data\_rf))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_3 = floor(0.7 \* nrow(data\_rf))**
* **s\_size\_3**
* **train\_index\_3 <- sample(1:nrow(data\_rf), size = s\_size\_3)**
* **train\_3 <- data\_rf[train\_index\_3,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_3 <- data\_rf[-train\_index\_3,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_3 = naiveBayes(cut~. , train\_3)**

# Προβλέψεις

# Make predicitons on test dataset.

* **y\_pred\_3 <- predict(model\_3, newdata = test\_3)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_3<-table(y\_pred\_3, test\_3$cut)**
* **ConMat\_3**
* **confusionMatrix(ConMat\_3)**

# 5. FUTURE SELECTION AND ADJUSTMENTS

Στο μοντέλο που θα σας δώσει την καλύτερη ακρίβεια (accuracy) από τα τρία, να δείτε πως μπορείτε να την βελτιώσετε.

## a) Μέσα από τις συσχετίσεις που θα βρείτε στο 1ο ερώτημα, να επιλέξετε κατά την κρίση σας τα features που είναι απαραίτητα για την διαδικασία του classification, και να συγκρίνετε τις επιδόσεις για διαφορετικά sets από features.

# Κατασκυεάζουμε δύο καινούρια σύνολα δεδομένων πάνω στα οπποία θα κατασκευαστούν αντίστοιχα μοντέλα,

# και τα οποία προκύπτουν από τις συσχετίσεις που υπολογίστηκαν στην 1η άσκηση.

# 1o - 4th model

* **data\_f1 <- diamonds[,c("carat", "cut", "color", "clarity", "depth", "table", "price")]**

# 4th model for 4th data Frame

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_4 = floor(0.7 \* nrow(data\_f1))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_4 = floor(0.7 \* nrow(data\_f1))**
* **s\_size\_4**
* **train\_index\_4 <- sample(1:nrow(data\_f1), size = s\_size\_4)**
* **train\_4 <- data\_f1[train\_index\_4,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_4 <- data\_f1[-train\_index\_4,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_4 = naiveBayes(cut~. , train\_4)**

# Προβλέψεις

# Make predicitons on test dataset.

* **y\_pred\_4 <- predict(model\_4, newdata = test\_4)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_4<-table(y\_pred\_4, test\_4$cut)**
* **ConMat\_4**
* **confusionMatrix(ConMat\_4)**

# Κατασκυεάζουμε δύο καινούρια σύνολα δεδομένων πάνω στα οπποία θα κατασκευαστούν αντίστοιχα μοντέλα,

# και τα οποία προκύπτουν από τις συσχετίσεις που υπολογίστηκαν στην 1η άσκηση.

# 2o - 4th model

* **data\_f2 <- diamonds[,c( "cut", "x", "y", "z")]**

# 4th model for 4th data Frame

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_5 = floor(0.7 \* nrow(data\_f2))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_5 = floor(0.7 \* nrow(data\_f2))**
* **s\_size\_5**
* **train\_index\_5 <- sample(1:nrow(data\_f2), size = s\_size\_5)**
* **train\_5 <- data\_f2[train\_index\_5,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_5 <- data\_f2[-train\_index\_5,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_5 = naiveBayes(cut~. , train\_5)**

# Προβλέψεις

# Make predicitons on test dataset.

* **y\_pred\_5 <- predict(model\_5, newdata = test\_5)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_5<-table(y\_pred\_5, test\_5$cut)**
* **ConMat\_5**
* **confusionMatrix(ConMat\_5)**

## b) Να δοκιμάσετε να κάνετε scale τα δεδομένα και να συγκρίνετε τις επιδόσεις.

# 7th model for 3rd data Frame (where NA's are filled with the Random Forest imputation)

# But with rescaled data.

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_7 = floor(0.7 \* nrow(data\_rf))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_7 = floor(0.7 \* nrow(data\_rf))**
* **s\_size\_7**
* **train\_index\_7 <- sample(1:nrow(data\_rf), size = s\_size\_7)**
* **train\_7 <- data\_rf[train\_index\_7,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_7 <- data\_rf[-train\_index\_7,]**

# Feature Scaling

* **train\_scale <- scale(train\_7[, 1:10])**
* **test\_scale <- scale(test\_7[, 1:10])**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_7 = naiveBayes(cut~. , train\_scale)**

# Προβλέψεις

# Make predicitons on test dataset.

* **y\_pred\_7 <- predict(model\_7, newdata = test\_scale)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_7<-table(y\_pred\_7, test\_7$cut)**
* **ConMat\_7**
* **confusionMatrix(ConMat\_7)**

## c) Να μεγαλώσετε το δείγμα για το training set, και γενικά το δείγμα σας σε σχέση με τον πληθυσμό των 50000 παρατηρήσεων.

# Τέλος, εδώ, Θα αυξήσουμε τόσο το αρχικό δείγμα που πήραμε όσο και το ποσοστό των δεδομένων που

# θα ανήκουν στο σύνολο εκπαίδευσης.

# Συγκεκριμένα, θα έχουμε 80% των αρχικών παρατηρήσεων στο δείγμα μας.

# Και επίσης, το σύνολο εκπαίδευσης θα περιέχει 80% του δείγματος, ενώ το σύνολο ελέγχου το 20%.

# 7th model for 3rd data Frame (where NAs are filled with the Random Forest imputation)

# But with rescaled data.

# Μεταβλητή πρόβλεψης η cut (target variable)

# Set seed (αυτό το κάνουμε ώστε σε κάθε μοντέλο που τρέχουμε να παίρνουμε κάθε φορά τα ίδια δεδομένα)

# και έτσι να γίνεται πραγματική σύκγριση μεταξύ των διάφορων μοντέλων.

* **set.seed(1)**

# Ορισμός μεγέθους δείγματος από τα αρχικό σύνολο (decide sample size)

# Συγκεκριμένα θα πάρουμε το 70% των αρχικών δεδομένων.

* **sample\_size\_8 = floor(0.8 \* nrow(data\_rf))**

# Ορισμός συνόλου εκπαίδευσης (training set), το οποίο θα αποτελείται από το 80% των παρατηρήσεων.

# Training set consists of 70% observations.

* **s\_size\_8 = floor(0.8 \* nrow(data\_rf))**
* **s\_size\_8**
* **train\_index\_8 <- sample(1:nrow(data\_rf), size = s\_size\_8)**
* **train\_8 <- data\_rf[train\_index\_8,]**

# Ορισμός συνόλου εκπάιδευσης (test set).

# Τest set consists of 30% observations of the sample data.

* **test\_8 <- data\_rf[-train\_index\_8,]**

# Κατασκευάζουμε το δέντρο χρησιμοποιώντας το σύνολο εκπαίδευσης.

# # Fitting the Naive Bayes Model

* **model\_8 = naiveBayes(cut~. , train\_8)**

# Προβλέψεις

# Make predicitons on test dataset.

* **y\_pred\_8 <- predict(model\_8, newdata = test\_8)**

# Evaluation of the model

# Κατασκευάζουμε τον Confusion matrix (Πίνακας σύγχησης).

# Ο πίνακας αυτός παρέχει τα αποτελέσματα και τις μετρικές αξιολόγησης του μοντέλου.

* **ConMat\_8<-table(y\_pred\_8, test\_8$cut)**
* **ConMat\_8**
* **confusionMatrix(ConMat\_8)**