Final_project_02_07_22_RMD

nicola manfrini

2/7/2022

INTRODUCTION/OVERVIEW/EXECUTIVE SUMMARY

-THE DATASET- Working in the cancer research field I selected from a database from Kaggle that had to do more or less with the my research interests. I decided to work on the "Breast Cancer Wisconsin" dataset (https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data). It is a dataset with various features that are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. Specifically the fetures describe characteristics of the cell nuclei present in the images of breast cancer cells analyzed. Data were published along with the paper of K. P. Bennett and O. L. Mangasarian "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", on the "Optimization Methods and Software" journal in 1992 (Volume 1, pages 23-34).

For each cell nucleus they give: 1) ID number 2) Diagnosis for the specific cell/nucleus sample (malignant, M, or, benign, B).

For each analyzed nucleus they also give the following values/infos:

a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

For all of these features authors computed the mean, standard error and "worst" (mean of the three largest values), giving rise to a total of 30 features.

-GOAL- My idea was, of course, to try to predict DIAGNOSIS (malignant, M, or benign, B) using all the other variables as possible predictors.

-APPROACH- I started by cleaning a bit the data (removing NAs from the database) Then I looked for variables which could be good predictors of DIAGNOSIS. Next, I looked for correlation between variables. I did this because variables which are correlated with one another are not to good for generating prediction models. I then performed logistic regression using all variables, setting this as a starting point from which I could only do better. Then I performed logistic regression with 2 selected variables getting a much better result. Next, I changed approach and performed LDA and random forest. Random forest was the best model of all. Lastly I performed KNN, but obtained results wich were not as good.

METHODS/ANALYSIS

First thing I installed and loaded the libraries I thought were important

```r
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse

## Warning: package 'tidyverse' was built under R version 4.0.5

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.0     v forcats 0.5.1

## Warning: package 'tibble' was built under R version 4.0.5

## Warning: package 'tidyr' was built under R version 4.0.5

## Warning: package 'readr' was built under R version 4.0.5

## Warning: package 'dplyr' was built under R version 4.0.5

## Warning: package 'stringr' was built under R version 4.0.2

## Warning: package 'forcats' was built under R version 4.0.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret

## Warning: package 'caret' was built under R version 4.0.5

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")

library(caret)
library(ggplot2)
library(tidyverse)
```

Next thing I did, I explored briefly the data and did a little bit of cleaning

```r
#Let's upload the dataset
breast_cancer  <- read.csv(file = "wisconsin_breast_c_data.csv")
# let's check all variables
head(breast_cancer)
```

```
##         id diagnosis radius_mean texture_mean perimeter_mean area_mean
## 1   842302         M       17.99        10.38         122.80    1001.0
## 2   842517         M       20.57        17.77         132.90    1326.0
## 3 84300903         M       19.69        21.25         130.00    1203.0
## 4 84348301         M       11.42        20.38          77.58     386.1
## 5 84358402         M       20.29        14.34         135.10    1297.0
## 6   843786         M       12.45        15.70          82.57     477.1
##   smoothness_mean compactness_mean concavity_mean concave.points_mean
## 1         0.11840          0.27760         0.3001             0.14710
## 2         0.08474          0.07864         0.0869             0.07017
## 3         0.10960          0.15990         0.1974             0.12790
## 4         0.14250          0.28390         0.2414             0.10520
## 5         0.10030          0.13280         0.1980             0.10430
## 6         0.12780          0.17000         0.1578             0.08089
##   symmetry_mean fractal_dimension_mean radius_se texture_se perimeter_se
## 1        0.2419                0.07871    1.0950     0.9053        8.589
## 2        0.1812                0.05667    0.5435     0.7339        3.398
## 3        0.2069                0.05999    0.7456     0.7869        4.585
## 4        0.2597                0.09744    0.4956     1.1560        3.445
## 5        0.1809                0.05883    0.7572     0.7813        5.438
## 6        0.2087                0.07613    0.3345     0.8902        2.217
##   area_se smoothness_se compactness_se concavity_se concave.points_se
## 1  153.40      0.006399        0.04904      0.05373           0.01587
## 2   74.08      0.005225        0.01308      0.01860           0.01340
## 3   94.03      0.006150        0.04006      0.03832           0.02058
## 4   27.23      0.009110        0.07458      0.05661           0.01867
## 5   94.44      0.011490        0.02461      0.05688           0.01885
## 6   27.19      0.007510        0.03345      0.03672           0.01137
##   symmetry_se fractal_dimension_se radius_worst texture_worst perimeter_worst
## 1     0.03003             0.006193        25.38         17.33          184.60
## 2     0.01389             0.003532        24.99         23.41          158.80
## 3     0.02250             0.004571        23.57         25.53          152.50
## 4     0.05963             0.009208        14.91         26.50           98.87
## 5     0.01756             0.005115        22.54         16.67          152.20
## 6     0.02165             0.005082        15.47         23.75          103.40
##   area_worst smoothness_worst compactness_worst concavity_worst
## 1     2019.0           0.1622            0.6656          0.7119
## 2     1956.0           0.1238            0.1866          0.2416
## 3     1709.0           0.1444            0.4245          0.4504
## 4      567.7           0.2098            0.8663          0.6869
## 5     1575.0           0.1374            0.2050          0.4000
## 6      741.6           0.1791            0.5249          0.5355
##   concave.points_worst symmetry_worst fractal_dimension_worst  X
## 1               0.2654         0.4601                 0.11890 NA
## 2               0.1860         0.2750                 0.08902 NA
## 3               0.2430         0.3613                 0.08758 NA
## 4               0.2575         0.6638                 0.17300 NA
## 5               0.1625         0.2364                 0.07678 NA
## 6               0.1741         0.3985                 0.12440 NA
```

```
#we will be interested in the diagnosis variable so will convert it in to factor to ease all of the ana
# let's convert in factors
breast_cancer$diagnosis <- as.factor(breast_cancer$diagnosis)
# Let's check it
breast_cancer$diagnosis
```

```
##   [1] M M M M M M M M M M M M M M M M M M M B B B M M M M M M M M M M M M M M M
##  [38] B M M M M M M M M B M B B B B B M M B M M B B B B M B M M B B B B M B M M
##  [75] B M B M M B B B M M B M M M B B B M B B M M B B B M M B B B B M B B M B B
## [112] B B B B B B M M M B M M B B B M M B M B M M B M M B B M B B M B B B B M B
## [149] B B B B B B B B M B B B B M M B M B B M M B B M M B B B B M B B M M M B M
## [186] B M B B B M B B M M B M M M M B M M M B M B M B B M B M M M M B B M M B B
## [223] B M B B B B B M M B B M B B M M B M B B B B M B B B B B M B M M M M M M M
## [260] M M M M M M M B B B B B B M B M B B M B B M B M M B B B B B B B B B B B B
## [297] B M B B M B M B B B B B B B B B B B B B B M B B B M B M B B B B M M M B B
## [334] B B M B M B M B B B M B B B B B B B M M M B B B B B B B B B B B M M B M M
## [371] M B M M B B B B B M B B B B B M B B B M B B M M B B B B B B M B B B B B B
## [408] B M B B B B B M B B M B B B B B B B B B B B B M B M M B M B B B B B M B B
## [445] M B M B B M B M B B B B B B B B M M B B B B B B M B B B B B B B B B B M B
## [482] B B B B B B M B M B B M B B B B B M M B M B M B B B B B M B B M B M B M M
## [519] B B B M B B B B B B B B B B B M B M M B B B B B B B B B B B B B B B B B B
## [556] B B B B B B B M M M M M M B
## Levels: B M
```

```
# let's check the structure of the database to see if it is tidy or if we have to tidy it up a little
str(breast_cancer)
```

```
## 'data.frame':    569 obs. of  33 variables:
##  $ id                     : int  842302 842517 84300903 84348301 84358402 843786 844359 84458202 8449
##  $ diagnosis              : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean              : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
##  $ X                      : logi  NA NA NA NA NA NA ...
```

```r
# we can see that the last column is full of NA
#let's check how many NAs we have
sum(is.na(breast_cancer))
```

```
## [1] 569
```

```r
#is seems to be only the one column. Let's make sure the length of the column in the dataset is the sam
#of NAs
nrow(breast_cancer)
```

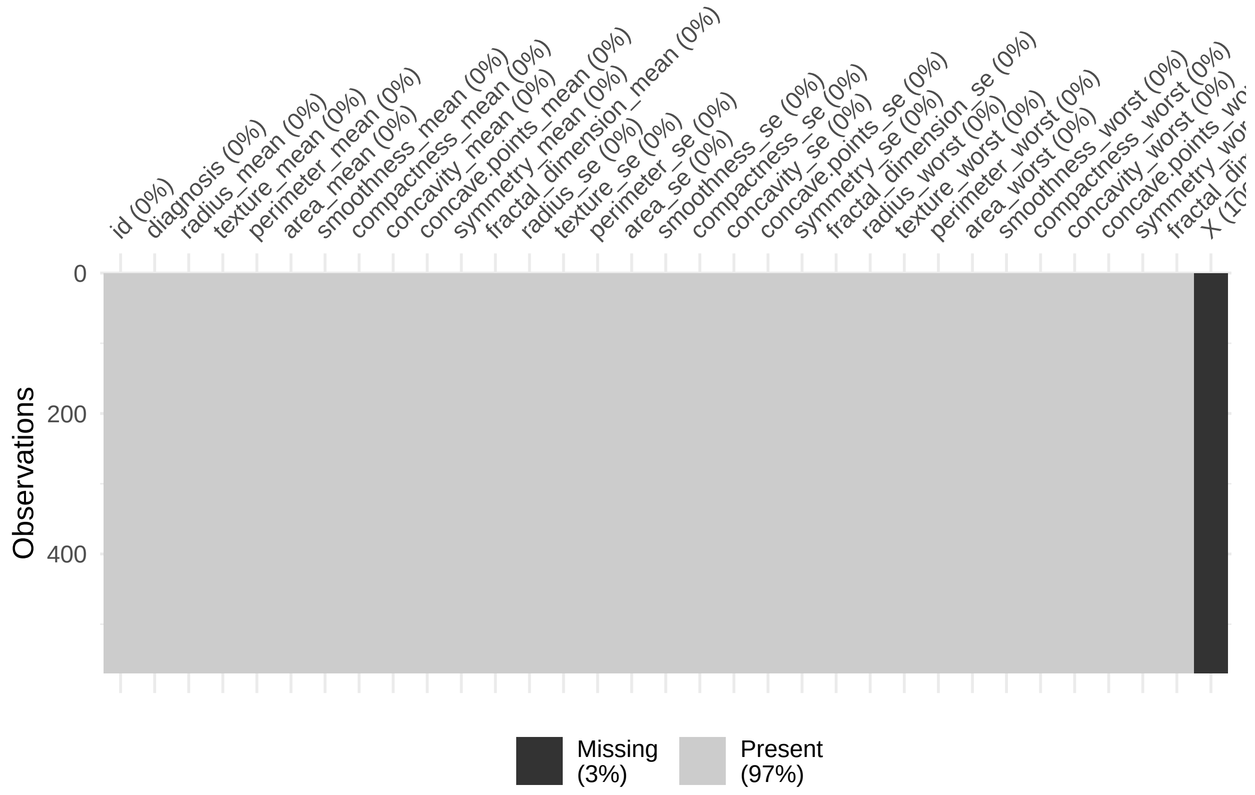
```
## [1] 569
```

```r
# yes, it is the same.  It seems as if only one full column should be filled with NAs
# but let's check better by looking at missing elements
#to do so we need the naniar package
if(!require(naniar)) install.packages("naniar", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: naniar
```

```
## Warning: package 'naniar' was built under R version 4.0.5
```

```r
library(naniar)
vis_miss(breast_cancer)
```



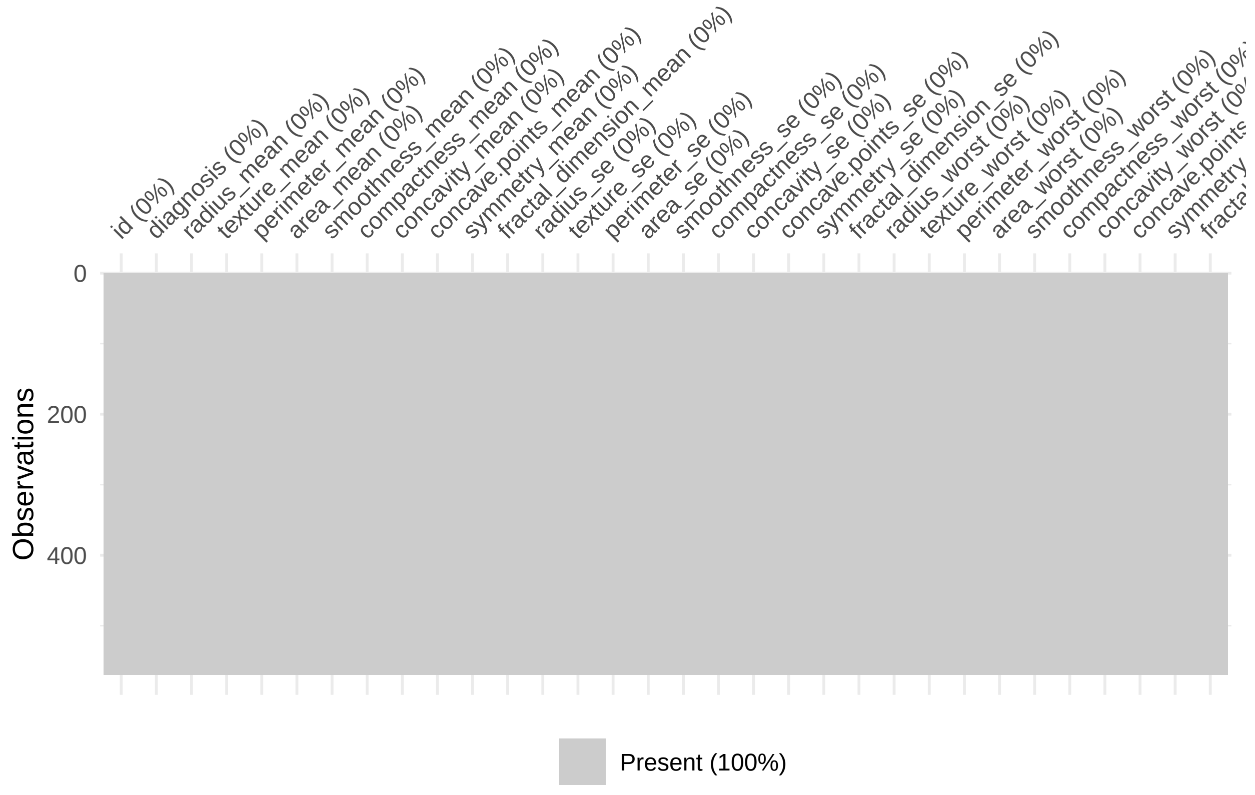
```r
# yes, the last column is made by missing values#
# how many columns do we have? This way we will remove the last one with all the NAs
ncol(breast_cancer)
```

```
## [1] 33
```

```r
# let's remove the last one with NAs and check our new database #
breast_cancer <- subset(breast_cancer[,-33])
# now, how many columns do we have? (should be 32)
ncol(breast_cancer)
```

```
## [1] 32
```

```r
# 32, perfect. Let's check If we have any missing values left
vis_miss(breast_cancer)
```



```
# no missing values, perfect!
#data is now clean! We are good to go!
```

Next, I performed a deeper exploration of data

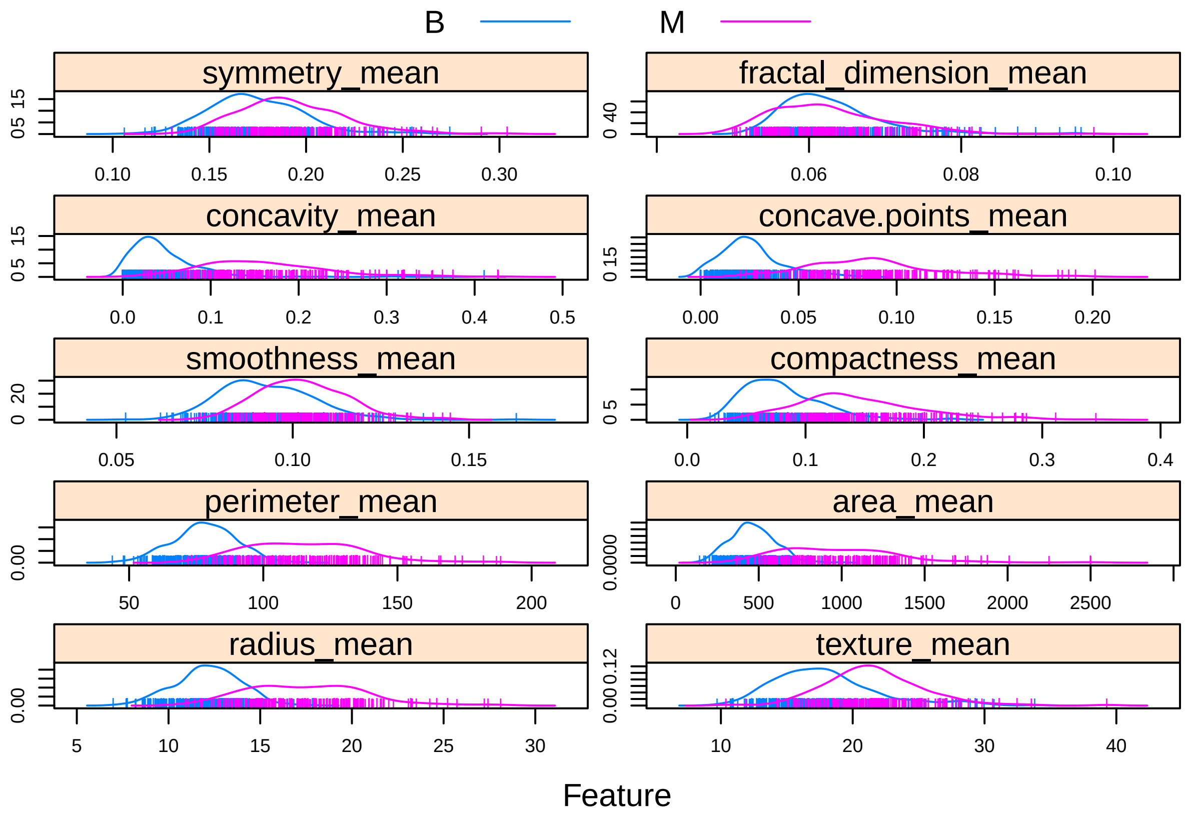
```
#By reading the description of the dataset in Kaggle we found that the most interesting variable is dia
#($diagnosis) which is B for benign and M for malign
#let's have a look
breast_cancer$diagnosis <- as.factor(breast_cancer$diagnosis)
breast_cancer$diagnosis
```

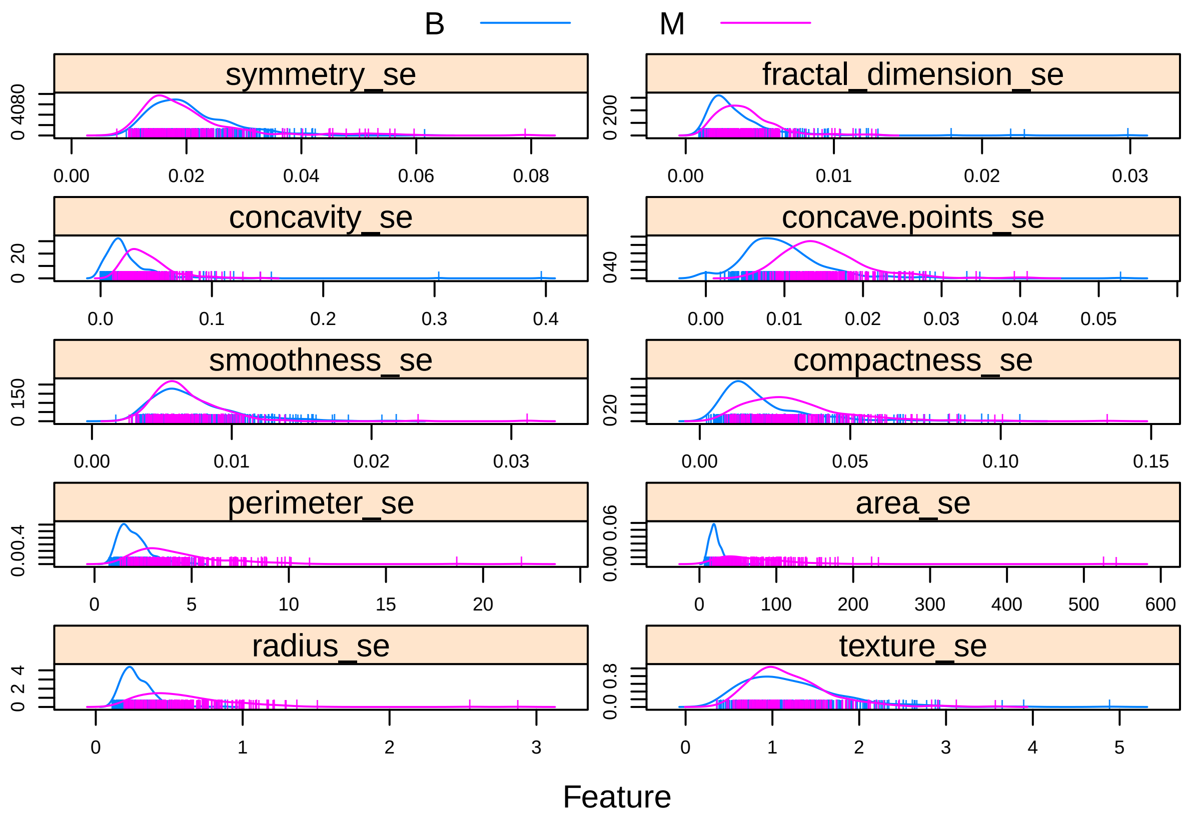
```
##   [1] M M M M M M M M M M M M M M M M M M M B B B M M M M M M M M M M M M M M M
##  [38] B M M M M M M M M B M B B B B B M M B M M B B B B M B M M B B B B M B M M
##  [75] B M B M M B B B M M B M M M B B B M B B M M B B B M M B B B B M B B M B B
## [112] B B B B B B M M M B M M B B B M M B M B M M B M M B B M B B M B B B B M B
## [149] B B B B B B B B M B B B B M M B M B B M M B B M M B B B B M B B M M M B M
## [186] B M B B B M B B M M B M M M M B M M M B M B M B B M B M M M M B B M M B B
## [223] B M B B B B B M M B B M B B M M B M B B B B M B B B B B M B M M M M M M M
## [260] M M M M M M M B B B B B B M B M B B M B B M B M M B B B B B B B B B B B B
## [297] B M B B M B M B B B B B B B B B B B B B B M B B B M B M B B B B M M M B B
## [334] B B M B M B M B B B M B B B B B B B M M M B B B B B B B B B B B M M B M M
## [371] M B M M B B B B B M B B B B B M B B B M B B M M B B B B B B M B B B B B B
## [408] B M B B B B B M B B M B B B B B B B B B B B B M B M M B M B B B B B M B B
## [445] M B M B B M B M B B B B B B B B M M B B B B B B M B B B B B B B B B B M B
## [482] B B B B B B M B M B B M B B B B B M M B M B M B B B B B M B B M B M B M M
## [519] B B B M B B B B B B B B B B B M B M M B B B B B B B B B B B B B B B B B B
## [556] B B B B B B B M M M M M M B
## Levels: B M
```

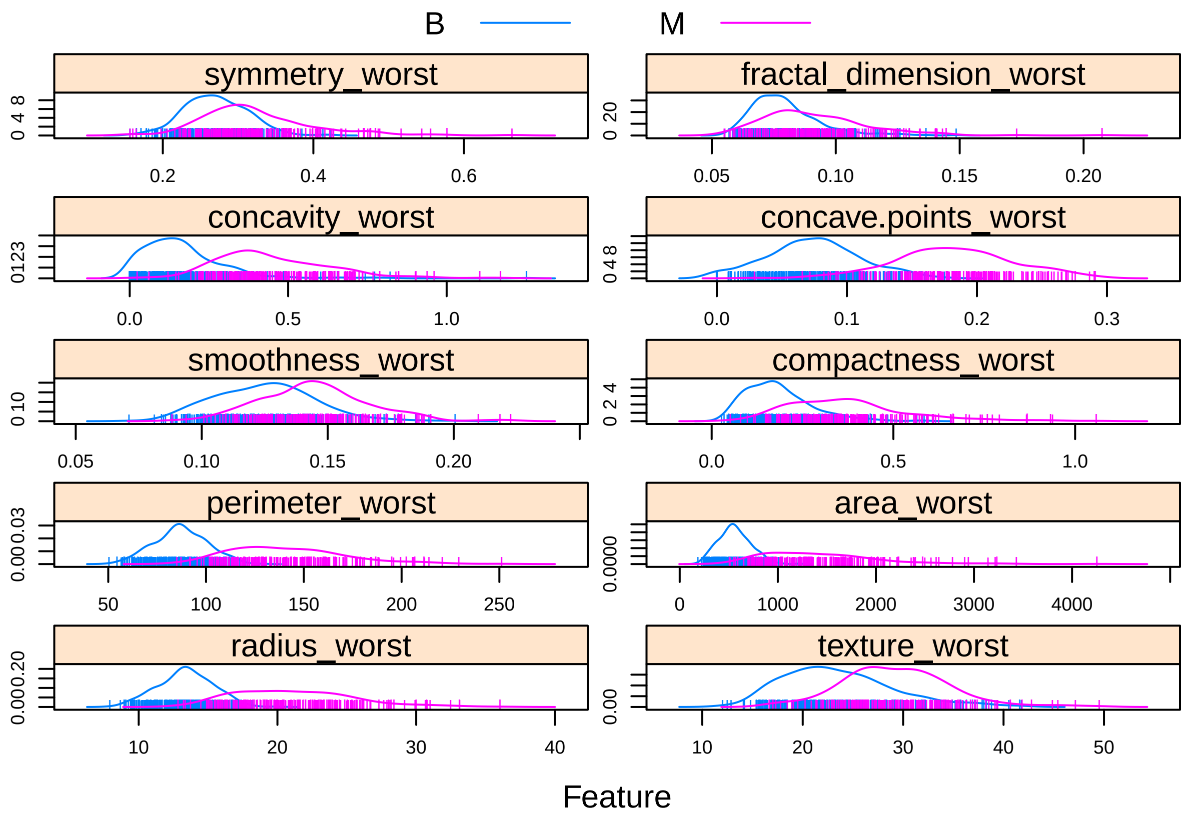
```
# Let's check how many malign and benign tumors we have overall (in percentage)
prop.table(table(breast_cancer$diagnosis))
```

```
##
##         B         M
## 0.6274165 0.3725835
```

```
# let's check a little bit more in the characteristics of the data and see if some predictors correlate
#specific diagnosis
# lets create breast_cancer_plot without first 2 columns which are Id and diagnosis (it is what we want
# diagnosis)
breast_cancer_plot <- breast_cancer[, -c(1,2)]
# Let's create breast_cancer_diag which is the diagnosis value we want to plot
breast_cancer_diag <- breast_cancer[,2]
# let's plot everything now and see if we have good predictors for diagnosis
scales <- list(x=list(relation="free"),y=list(relation="free"), cex=0.6)
featurePlot(x=breast_cancer_plot, y=breast_cancer_diag, plot="density", scales = scales,
            layout = c(2,5), auto.key = list(columns = 2), pch = "|", )
```




B
M
symmetry_se
fractal_dimension_se
concavity_se
concave.points_se
smoothness_se
compactness_se
perimeter_se
area_se
radius_se
texture_se
Feature



```r
# wow!! We clearly see some predictors that have more impact on diagnosis than others
# these are for example:
# concave points worst / perimeter worst / area worst /radius worst / concavity mean / concave points m
# compactness mean / radius mean / perimeter mean
```

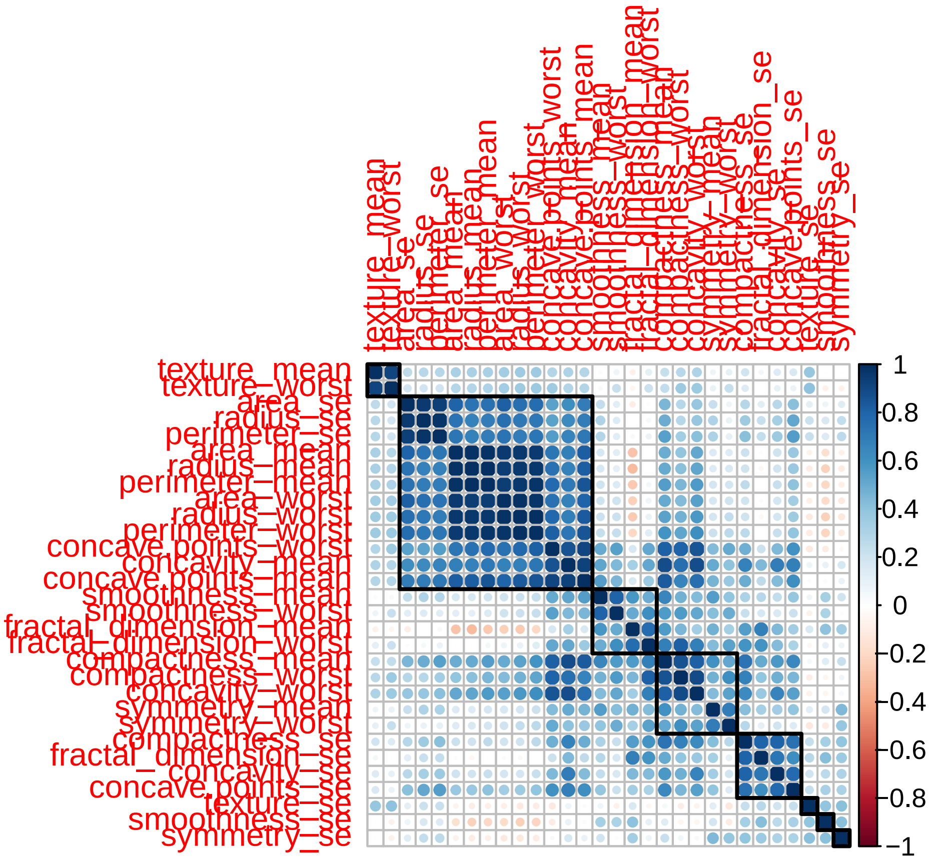
Next, I checked if variables had any correlation with one another as this is not really good for machine learning and model generation

```r
#Clearly some of the variables in the dataset are similar/should be correlated.
# So, let's check if we have any correlation between variables
# to do so we will, once again, remove the id and diagnosis columns
breast_cancer_corr <- cor(breast_cancer %>% select(-id, -diagnosis))
#now let's plot correlations, to do so we will need the corrplot package#
if(!require(corrplot)) install.packages("corrplot", repos = "http://cran.us.r-project.org")
```

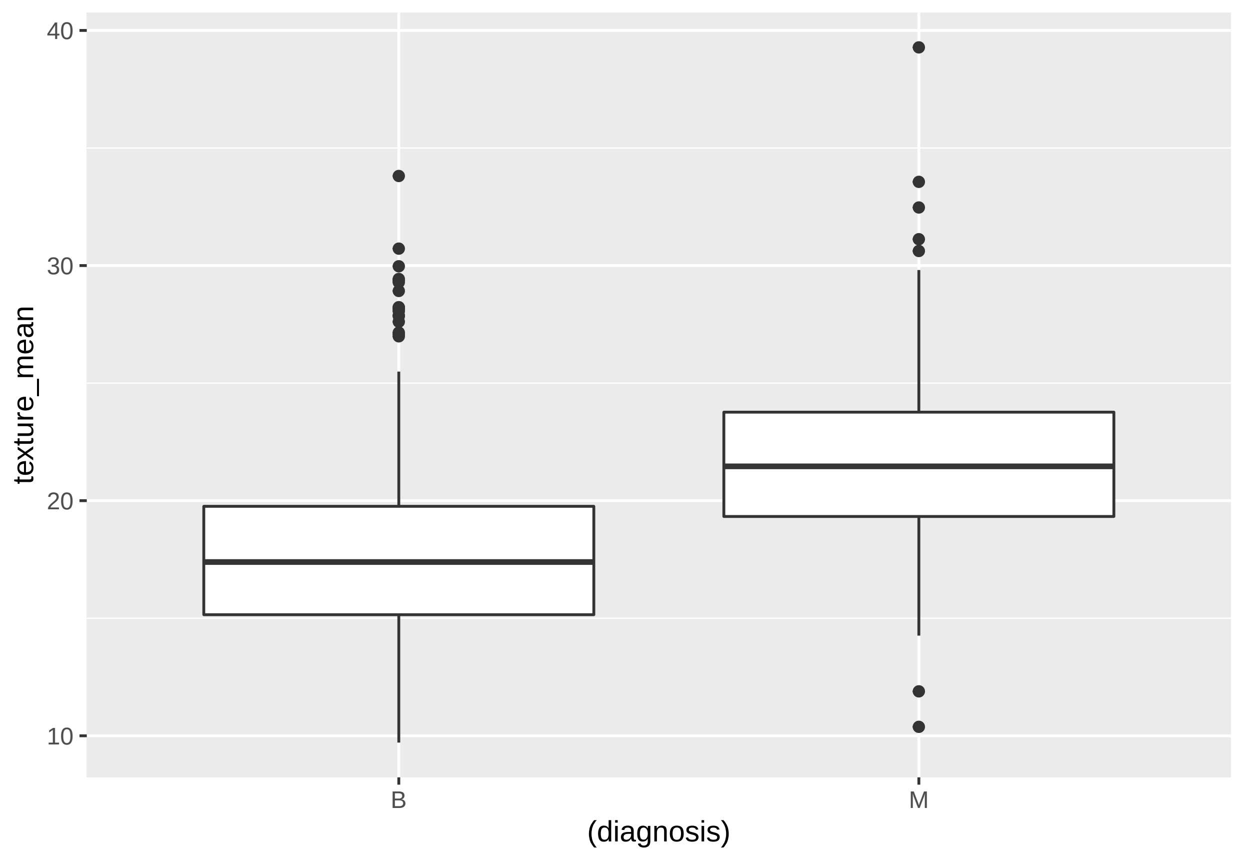
```
## Loading required package: corrplot

## corrplot 0.92 loaded
```

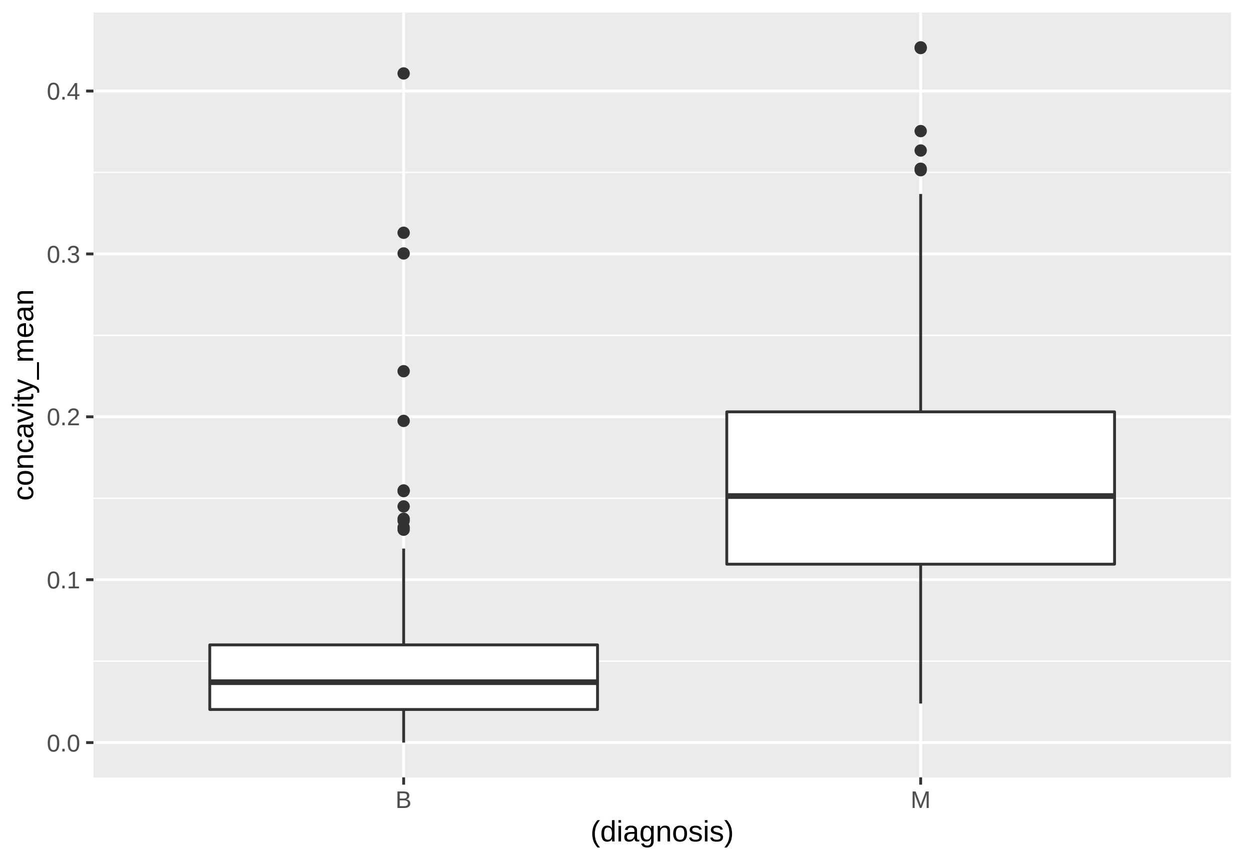
```r
library(corrplot)
corrplot::corrplot(breast_cancer_corr, order = "hclust", tl.cex = 1, addrect = 8)
```



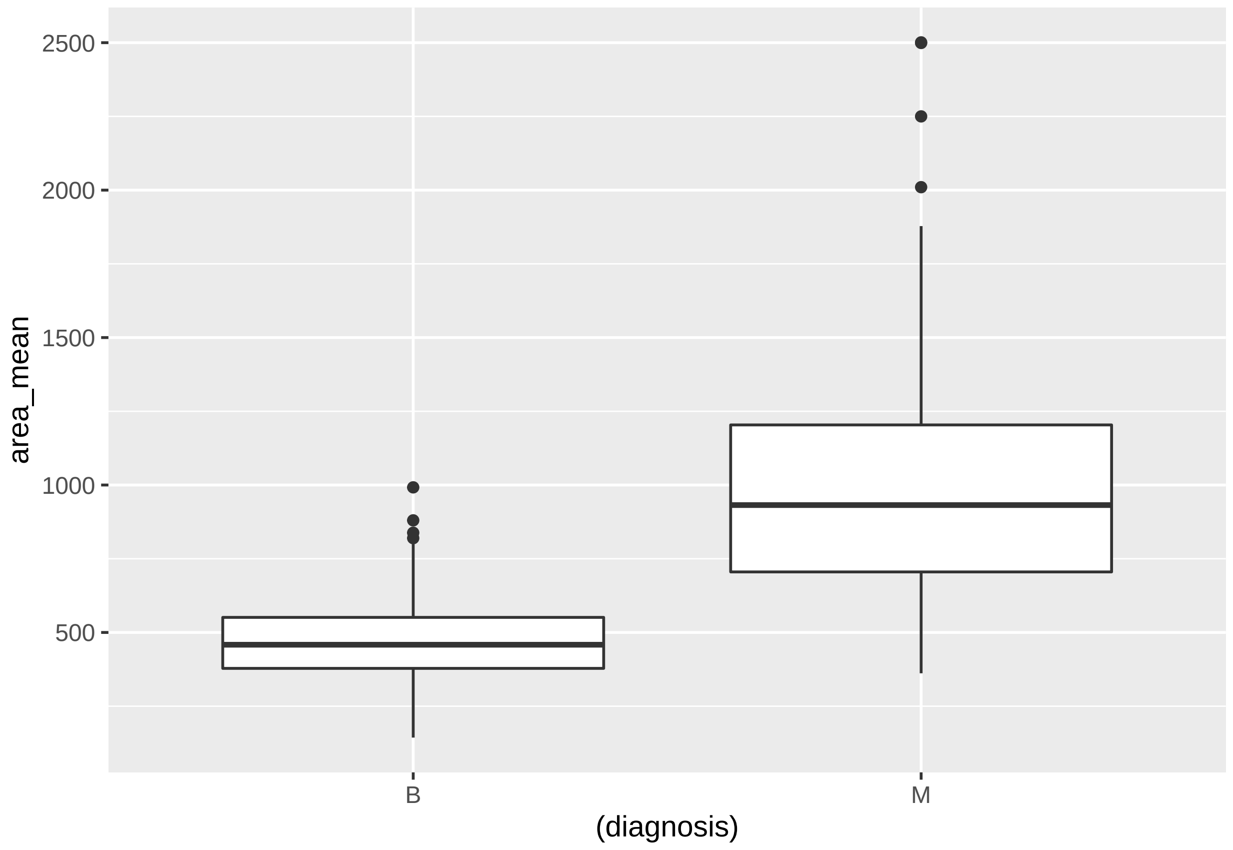
```r
# wow we have lots of correlated parameters, aka some variables are highly correlated with one another
# while others are not correlated at all we will focus only on some of them.
# using highly correlated variables in general is not good for machine learning
# let's blot some of the variables better in order to better see their impact on diagnosis
# let's check texture_mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = texture_mean)) +
  geom_boxplot()
```



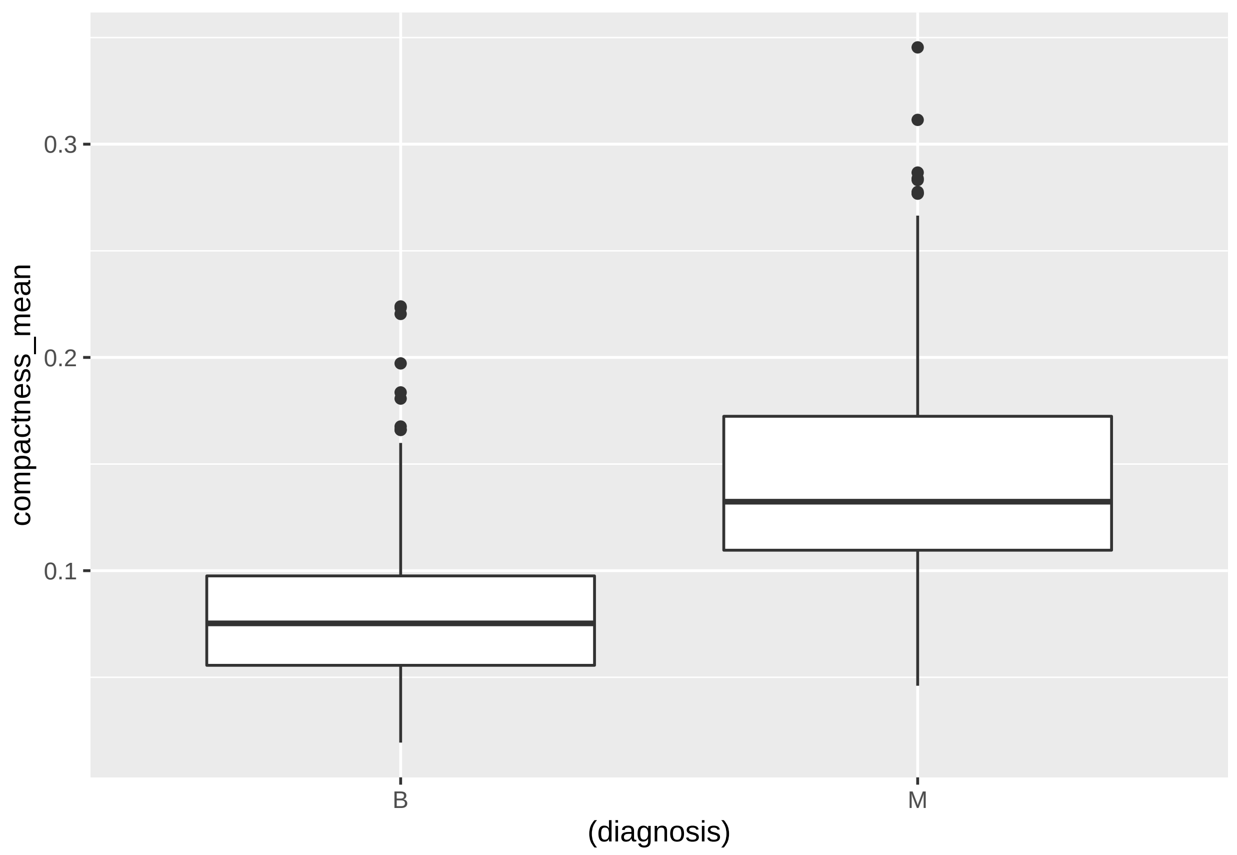
```
# let's check concavity_mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = concavity_mean)) +
  geom_boxplot()
```



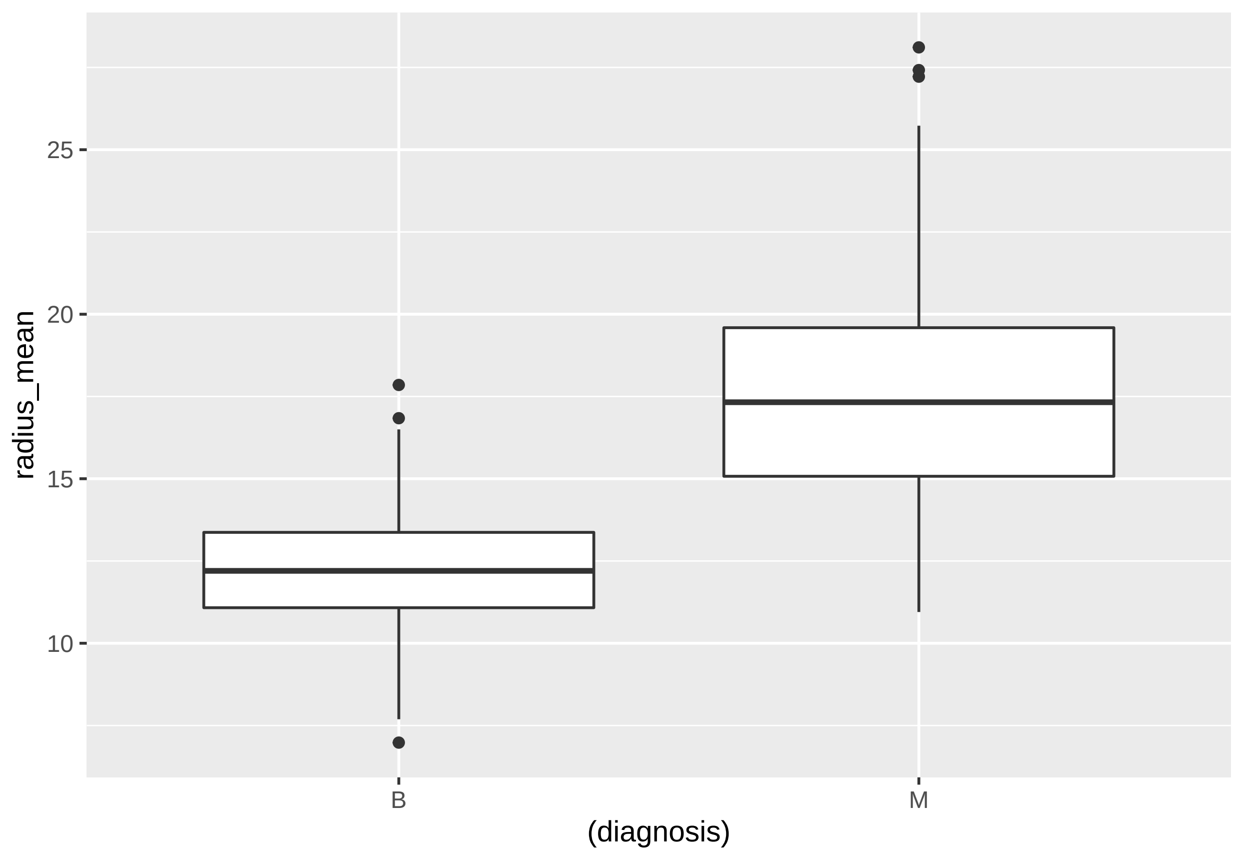
```
# let's check are_mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = area_mean)) +
  geom_boxplot()
```



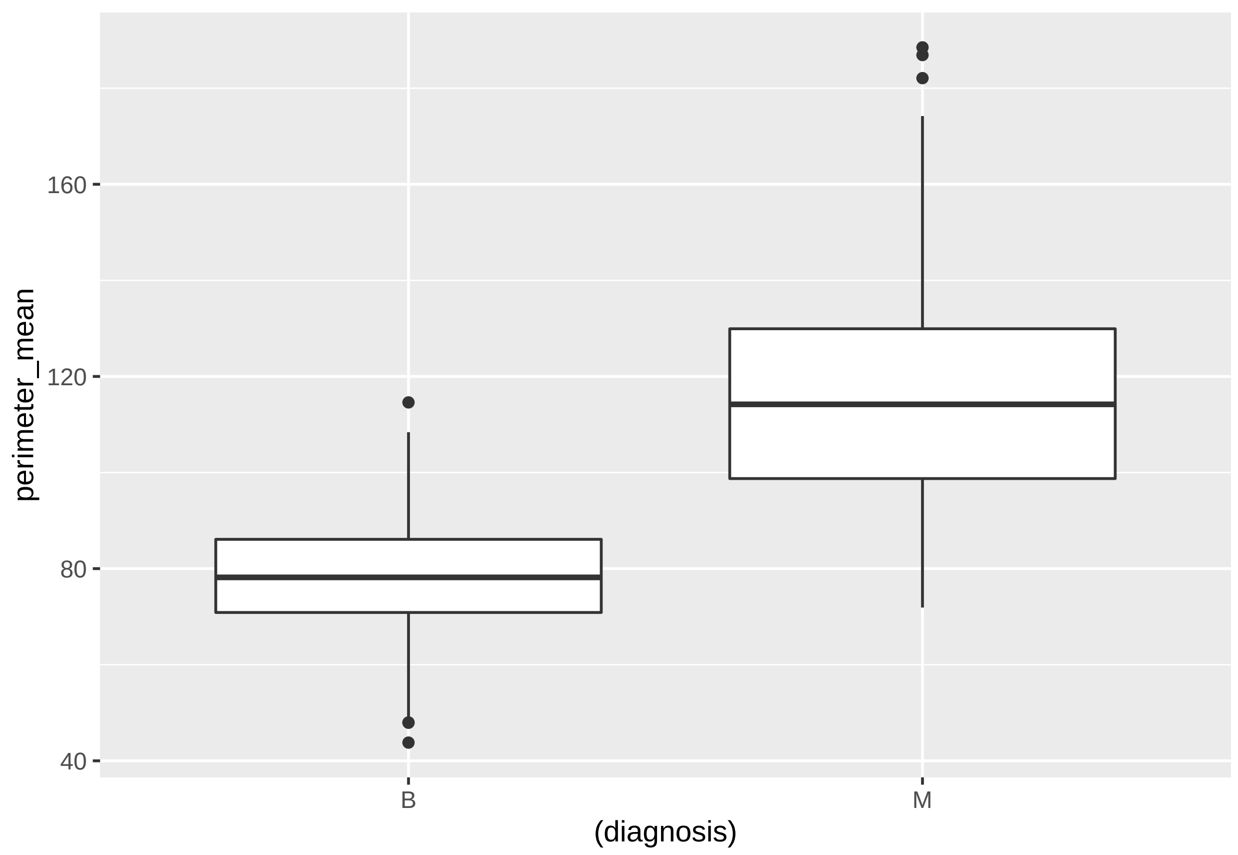
```r
# let's check compactness mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = compactness_mean)) +
  geom_boxplot()
```



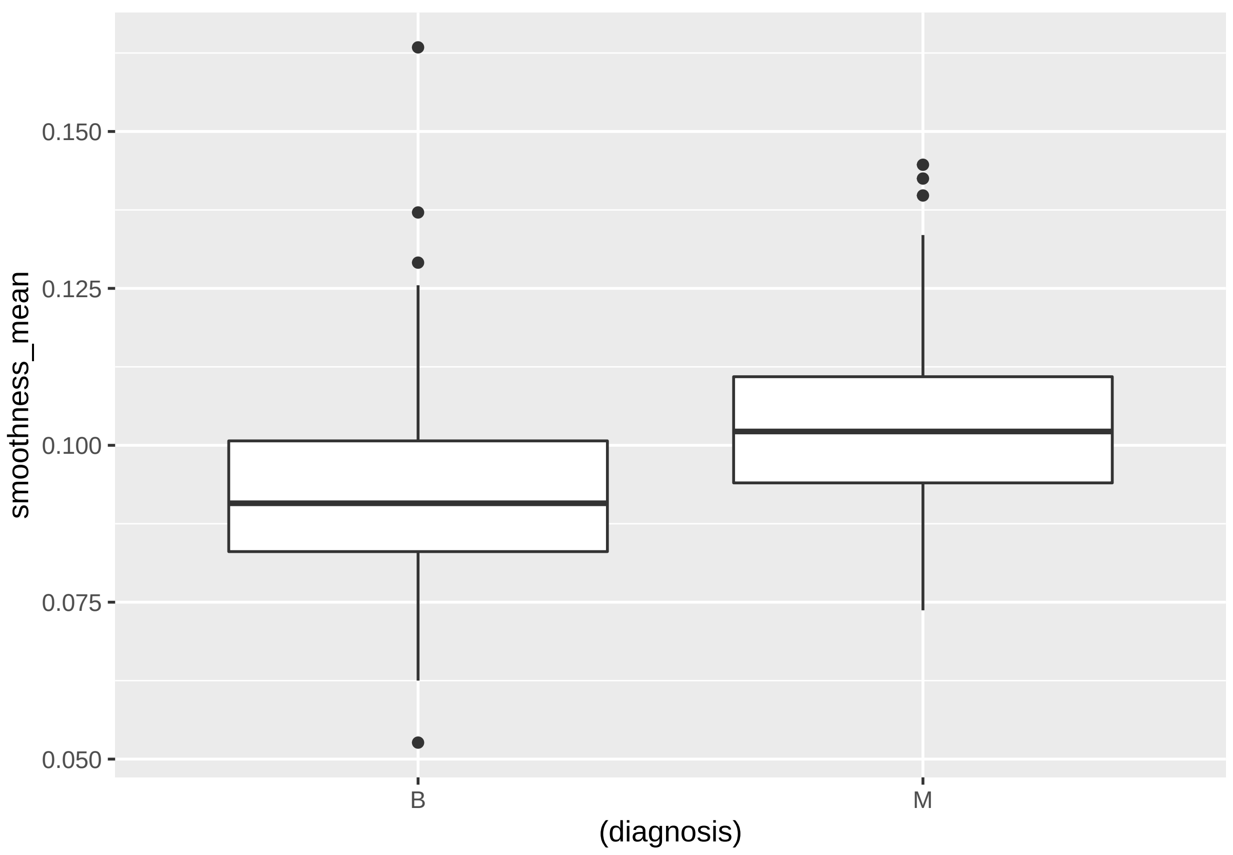
```
#let's check radius mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = radius_mean)) +
  geom_boxplot()
```



```r
# let's check perimeter mean
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = perimeter_mean)) +
  geom_boxplot()
```



```r
# let's check smoothness
ggplot(data= breast_cancer, aes(x = (diagnosis) , y = smoothness_mean)) +
  geom_boxplot()
```



```
# all of these variables seem to positively correlate with a malignant diagnosis
```

Next I started generating models, following these steps: 1) train and test set generation 2) logistic regression using alla variables 3) logistic regression using two of the most predictive variables 4) LDA 5) RANDOM FOREST 6) KNN with K= 5 7) KNN with K= 3

```
#Let's start a little machine learning
#First thing we need a training and a test set:

set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <- createDataPartition(y = breast_cancer$diagnosis, times = 1, p = 0.1, list = FALSE)
breast_train <- breast_cancer[-test_index,]
breast_test <- breast_cancer[test_index,]

# Let's check the target variable (diagnosis) and see how it is represented in the test and training se

prop.table(table(breast_train$diagnosis))*100
```

```
##
##      B      M
## 62.818 37.182
```

```r
prop.table(table(breast_test$diagnosis))*100
```

```
##
##        B        M
## 62.06897 37.93103
```

```r
# The selected variable is represented similarly in train and test sets. It's a good thing
#Let's start applying machine learning models
# Let's start with logistic regression using all predictors and 5-fold cross validation
# We know that using all predictors some of them will be highly correlated to one another and this is n
# for machine learning, so from what we obtain using all predictors we will then try to make it better
# Let's set the fit control with a  5-fold cross validation.
fitControl <- trainControl(method="cv",
                           number = 5,
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary)

# logistic regression using all predictors

logreg_all<-train(diagnosis~.,data=breast_train[,-1],method="glm",family=binomial(),
            trControl=fitControl)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
## in the result set. ROC will be used instead.

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
# let's see variable importance
varImp(logreg_all)
```

```
## glm variable importance
##
##   only 20 most important variables shown (out of 30)
##
##                         Overall
## concavity_se             100.00
## area_mean                 79.64
## perimeter_se              62.01
## area_se                   48.89
## concave.points_se         45.17
## fractal_dimension_se      41.84
## fractal_dimension_worst   32.93
## compactness_mean          32.06
## radius_se                 30.69
## texture_worst             30.30
## concavity_mean            27.53
## area_worst                23.00
## compactness_se            22.52
## smoothness_mean           22.22
## concavity_worst           17.69
## radius_worst              14.52
## symmetry_se               13.40
## fractal_dimension_mean    13.07
## symmetry_worst            12.42
## perimeter_worst           11.57
```

```
#Perfect, now let's test the "logistic regression with all predictors" model on the test set.

logreg_all_pred<-predict(logreg_all,breast_test[,-c(1,2)])
cm_logreg_all<-confusionMatrix(logreg_all_pred,breast_test$diagnosis)
cm_logreg_all
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 35  4
##          M  1 18
##
##                Accuracy : 0.9138
##                  95% CI : (0.8102, 0.9714)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 4.436e-07
##
##                   Kappa : 0.8119
##
##  Mcnemar's Test P-Value : 0.3711
##
##             Sensitivity : 0.9722
##             Specificity : 0.8182
##          Pos Pred Value : 0.8974
##          Neg Pred Value : 0.9474
##              Prevalence : 0.6207
##          Detection Rate : 0.6034
##    Detection Prevalence : 0.6724
##       Balanced Accuracy : 0.8952
##
##        'Positive' Class : B
##
```

```
#great, good accuracy!!!! 0.9138!!!
# but we can do better we know that some variables are more predictive of diagnosis than others and
#we know that some variables are higly correlated: not a good thing to train models, let's try to imple
# Let's try using only 2 variables with high importance but that are not correlated
logreg_area_compactness<-train(diagnosis~area_mean+compactness_mean, data=breast_train[,-1],method="glm
                  trControl=fitControl)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
## in the result set. ROC will be used instead.
```

```
varImp(logreg_area_compactness)
```

```
## glm variable importance
##
##                  Overall
## area_mean            100
## compactness_mean       0
```

```
#Now let's test the "logistic regression with only two good predictors" model on the test set.

logreg_area_compactness_pred<-predict(logreg_area_compactness,breast_test)
cm_logreg_area_compactness<-confusionMatrix(logreg_area_compactness_pred,breast_test$diagnosis)
cm_logreg_area_compactness
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 34  1
##          M  2 21
##
##                Accuracy : 0.9483
##                  95% CI : (0.8562, 0.9892)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 7.464e-09
##
##                   Kappa : 0.8911
##
##  Mcnemar's Test P-Value : 1
```

```
##
##             Sensitivity : 0.9444
##             Specificity : 0.9545
##          Pos Pred Value : 0.9714
##          Neg Pred Value : 0.9130
##              Prevalence : 0.6207
##          Detection Rate : 0.5862
##    Detection Prevalence : 0.6034
##       Balanced Accuracy : 0.9495
##
##        'Positive' Class : B
##
```

```
#Cool!! it is much better now, around 0.95 of accuracy, using less predictors is actually better
# Since using less predictors is actually better, let's try using a different approach: LDA
# LDA (Linear Discriminant Analysis) is a relatively simple solution to the problem of having too many f
# as it to assumes that the correlation structure is the same for all classes.
# This approach reduces the number of parameters to be estimated.
# LDA
model_breast_lda <- train(diagnosis~.,
                   breast_train,
                   method="lda2",
                   #tuneLength = 10,
                   metric="ROC",
                   preProc = c("center", "scale"),
                   trControl=fitControl)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
pred_breast_lda <- predict(model_breast_lda, breast_test)
cm_lda <- confusionMatrix(pred_breast_lda, breast_test$diagnosis, positive = "M")
cm_lda
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 36  4
##          M  0 18
##
##                Accuracy : 0.931
##                  95% CI : (0.8327, 0.9809)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 6.486e-08
##
##                   Kappa : 0.8482
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.8182
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9000
##              Prevalence : 0.3793
##          Detection Rate : 0.3103
##    Detection Prevalence : 0.3103
##       Balanced Accuracy : 0.9091
##
##        'Positive' Class : M
##
```

```
#good we get accuracy of 0.931, better than regression with alla variables but not as good as our previ
#(using only 2 good predictors)
#Let's try random forest
# random forest
if(!require(ranger)) install.packages("ranger", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: ranger

## Warning: package 'ranger' was built under R version 4.0.5
```

```
library("ranger")
model_breast_ranger  <- train(diagnosis ~ ., data = breast_train[,-1], method = "ranger")
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
# Let's check the model a bit
model_breast_ranger
```

```
## Random Forest
##
## 511 samples
##  30 predictor
##   2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 511, 511, 511, 511, 511, 511, ...
## Resampling results across tuning parameters:
##
##   mtry  splitrule   Accuracy   Kappa
##    2    gini        0.9591809  0.9113611
##    2    extratrees  0.9620148  0.9174271
##   16    gini        0.9556211  0.9039691
##   16    extratrees  0.9651432  0.9243654
##   30    gini        0.9506485  0.8931861
##   30    extratrees  0.9656045  0.9252841
##
## Tuning parameter 'min.node.size' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were mtry = 30, splitrule = extratrees
##  and min.node.size = 1.
```

```r
# Let's test it
pred_breast_ranger <- predict(model_breast_ranger, breast_test[,-c(1,2)])
cm_breast_ranger <- confusionMatrix(pred_breast_ranger, breast_test$diagnosis)
cm_breast_ranger
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   B  M
##            B 36  1
##            M  0 21
##
##                  Accuracy : 0.9828
##                    95% CI : (0.9076, 0.9996)
##       No Information Rate : 0.6207
##       P-Value [Acc > NIR] : 3.535e-11
##
##                     Kappa : 0.9631
##
##    Mcnemar's Test P-Value : 1
##
##               Sensitivity : 1.0000
##               Specificity : 0.9545
##            Pos Pred Value : 0.9730
##            Neg Pred Value : 1.0000
##                Prevalence : 0.6207
##            Detection Rate : 0.6207
##      Detection Prevalence : 0.6379
##         Balanced Accuracy : 0.9773
##
##          'Positive' Class : B
##
```

```r
# Wow! AWESOME!!!!! This is the best so far, we got an accuracy of 0.9828
# and really high sensitivity and specificity!!! perfect!!!! this is our final model!!!
# We probably can't do better than this, but
# For curiosity, let's try one last method: KNN -> K nearest neighbours using k= 5

knn5_breast_fit <- knn3(diagnosis ~ ., data = breast_train, k=5)
pred_breast_knn5 <- predict (knn5_breast_fit, breast_test, type= "class")
cm_breast_knn5 <- confusionMatrix(pred_breast_knn5, breast_test$diagnosis)
cm_breast_knn5
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   B  M
##            B 34 14
##            M  2  8
##
##                  Accuracy : 0.7241
##                    95% CI : (0.591, 0.8334)
##       No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 0.06612
##
##                   Kappa : 0.3446
##
##  Mcnemar's Test P-Value : 0.00596
##
##             Sensitivity : 0.9444
##             Specificity : 0.3636
##          Pos Pred Value : 0.7083
##          Neg Pred Value : 0.8000
##              Prevalence : 0.6207
##          Detection Rate : 0.5862
##    Detection Prevalence : 0.8276
##       Balanced Accuracy : 0.6540
##
##        'Positive' Class : B
##
```

```
# pretty bad this way only 0.7241 accuracy
# Let's try modifying the number of K, let's do 3
knn3_breast_fit <- knn3(diagnosis ~ ., data = breast_train, k=3)
pred_breast_knn3 <- predict (knn3_breast_fit, breast_test, type= "class")
cm_breast_knn3 <- confusionMatrix(pred_breast_knn3, breast_test$diagnosis)
cm_breast_knn3
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 34 13
##          M  2  9
##
##                Accuracy : 0.7414
##                  95% CI : (0.6096, 0.8474)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 0.036773
##
##                   Kappa : 0.3916
##
##  Mcnemar's Test P-Value : 0.009823
##
##             Sensitivity : 0.9444
##             Specificity : 0.4091
##          Pos Pred Value : 0.7234
##          Neg Pred Value : 0.8182
##              Prevalence : 0.6207
##          Detection Rate : 0.5862
##    Detection Prevalence : 0.8103
##       Balanced Accuracy : 0.6768
##
##        'Positive' Class : B
##
```

```
# A little better but still bad, let's stick with previous models
```

RESULTS

At this point I plotted the accuracies of all the models generated

```
model_results <- bind_rows(data_frame(method= "cm_logreg_all", accuracy = cm_logreg_all$overall["Accurac
                           data_frame(method= "cm_logreg_area_compactness", accuracy = cm_logreg_area_co
                           data_frame(method= "cm_lda", accuracy = cm_lda$overall["Accuracy"]),
                           data_frame(method= "cm_breast_ranger", accuracy = cm_breast_ranger$overall["A
                           data_frame(method= "cm_breast_knn5", accuracy = cm_breast_knn5$overall["Accur
                           data_frame(method= "cm_breast_knn3", accuracy = cm_breast_knn3$overall["Accur
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
```

```
model_results
```

```
## # A tibble: 6 x 2
##   method                     accuracy
##   <chr>                         <dbl>
## 1 cm_logreg_all                 0.914
## 2 cm_logreg_area_compactness    0.948
## 3 cm_lda                        0.931
## 4 cm_breast_ranger              0.983
## 5 cm_breast_knn5                0.724
## 6 cm_breast_knn3                0.741
```

```
# The best models are: RANDOM FOREST and logistic regression using area and compactness as predictors.
# we did great!!!!!
```

Clearly the best model is the one obtained with RANDOM FOREST, followed by the model generated using logistic regression with area and compactness and LDA. All three models have a quite high Accuracy, all above 0.9. But accuracy is not the only way to check how good a model is so I checked in detail sensitivity and specificity of these models

```
cm_logreg_area_compactness
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 34  1
##          M  2 21
##
##                Accuracy : 0.9483
##                  95% CI : (0.8562, 0.9892)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 7.464e-09
##
##                   Kappa : 0.8911
```

```
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9444
##             Specificity : 0.9545
##          Pos Pred Value : 0.9714
##          Neg Pred Value : 0.9130
##              Prevalence : 0.6207
##          Detection Rate : 0.5862
##    Detection Prevalence : 0.6034
##       Balanced Accuracy : 0.9495
##
##        'Positive' Class : B
##
```

```
cm_lda
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 36  4
##          M  0 18
##
##                Accuracy : 0.931
##                  95% CI : (0.8327, 0.9809)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 6.486e-08
##
##                   Kappa : 0.8482
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.8182
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9000
##              Prevalence : 0.3793
##          Detection Rate : 0.3103
##    Detection Prevalence : 0.3103
##       Balanced Accuracy : 0.9091
##
##        'Positive' Class : M
##
```

```
cm_breast_ranger
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 36  1
##          M  0 21
```

```
##
##                Accuracy : 0.9828
##                  95% CI : (0.9076, 0.9996)
##     No Information Rate : 0.6207
##     P-Value [Acc > NIR] : 3.535e-11
##
##                   Kappa : 0.9631
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 1.0000
##             Specificity : 0.9545
##          Pos Pred Value : 0.9730
##          Neg Pred Value : 1.0000
##              Prevalence : 0.6207
##          Detection Rate : 0.6207
##    Detection Prevalence : 0.6379
##       Balanced Accuracy : 0.9773
##
##        'Positive' Class : B
##
```

```
# they are really high in all cases, but once again RANDOM FOREST IS BETTER
# BEST MODEL: RANDOM FOREST!!
```

Looking at the two values for each model once again RANDOM FOREST wins, it is the best overall. This is great. I generated a really good model to predict if a breast cancer cell is malignant or benign based on info regarding the nucleus.

CONCLUSIONS

Applying different machine learning algorithms I obtained pretty good results. Worst models came out with KNN, that gave an accuracy of around 0.7. With logistic regression accuracy was always above 0.9. Best approach was RANDOM FOREST, a classification approach based on decision trees. This is not surprising as this method is good and indicated to be used when we are dealing with several variables.