Foundations of Probability and Statistics, Project

Emiliano Capasso, Antonello Scarcella, Simone Bellavia

2023-01-30

Introduction to Analysis

Breast cancer is one of the most prevalent forms of cancer in women worldwide. According to the World Health Organization, more than 1.7 million new cases of breast cancer are diagnosed each year, making it the most common form of cancer among women. Early detection and proper classification of the cancer are critical to ensure a positive prognosis and appropriate treatment.

The **Breast Cancer Wisconsin (Diagnostic) Data Set** provides information on the characteristics of cancer cells found in breast tissue and the final diagnosis (malignant or benign). This dataset has been used as a benchmark for many classification algorithms and continues to be a benchmark for researchers and developers of artificial intelligence systems in the field of medicine.

This dataset will be used in this project for the analysis of breast cancer. To this end, the project consists of several sections: data exploration, descriptive statistical analysis, feature selection with related testing part, and application of the linear model.

This dataset will be used in this project for the analysis of breast cancer. To this end, the report consists of several sections:

- the first part of the project will be based on **Data Preparation and Cleaning.** We will check the correctness of the type of data available, the presence of missing values and outliers;
- the second part will consist of **Descriptive Statistical Analysis.** Covariances and correlations between features will be checked to give us a better understanding of the nature of the data and its distribution;
- the third part will be based on **Inferential Statistics.** Tests and hypothesis testing will be carried out in order to be able to make considerations about the diagnosis of benign or malignant tumor;
- the fourth part will see the application of the **Linear Model.** The outputs will give more information about the data at hand.

Data Preparation and Cleaning

Importing data

The dataset is imported from a CSV file provided by the UCI Repository.

The only feature that identifies the type of diagnosis is represented by, precisely, *diagnosis*. Therefore, being a string, it is converted already as a factor from the import.

```r
# import data
data <- read.csv("data.csv",
                    header = TRUE,
                    sep=",",
                    stringsAsFactors = TRUE)
```

Several considerations can be made about the dataset. It consists of 33 features and 569 observations. Thanks to UCI, it is known that these features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. A fine needle aspiration (FNA) is a type of biopsy. It uses a very thin needle and syringe to remove a sample of cells, tissue or fluid from an abnormal area or lump in the body. The sample is then examined under a microscope. FNA is also called fine needle aspiration biopsy, or fine needle biopsy. [1] In this case, the features describe characteristics of the cell nuclei present in the image. A few of the images can be found at Web Link.

Some information about the features:

1) id: id number;
2) diagnosis (response): the diagnosis of breast tissues (M = malignant, B = benign);

From 3 to 32 ten real-valued features are computed for each cell nucleus:

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 (perimeterˆ2 / 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 more in-depth insight, the summary of all attributes and the head of the dataset are presented.

```
# get summary of variables
summary(data)
```

```
##        id             diagnosis  radius_mean      texture_mean  
##  Min.   :     8670   B:357     Min.   : 6.981   Min.   : 9.71  
##  1st Qu.:   869218   M:212     1st Qu.:11.700   1st Qu.:16.17  
##  Median :   906024             Median :13.370   Median :18.84  
##  Mean   : 30371831             Mean   :14.127   Mean   :19.29  
##  3rd Qu.:  8813129             3rd Qu.:15.780   3rd Qu.:21.80  
##  Max.   :911320502             Max.   :28.110   Max.   :39.28  
##  perimeter_mean     area_mean      smoothness_mean   compactness_mean 
##  Min.   : 43.79   Min.   : 143.5   Min.   :0.05263   Min.   :0.01938  
##  1st Qu.: 75.17   1st Qu.: 420.3   1st Qu.:0.08637   1st Qu.:0.06492  
##  Median : 86.24   Median : 551.1   Median :0.09587   Median :0.09263  
##  Mean   : 91.97   Mean   : 654.9   Mean   :0.09636   Mean   :0.10434  
##  3rd Qu.:104.10   3rd Qu.: 782.7   3rd Qu.:0.10530   3rd Qu.:0.13040  
##  Max.   :188.50   Max.   :2501.0   Max.   :0.16340   Max.   :0.34540  
##  concavity_mean    concave.points_mean symmetry_mean    fractal_dimension_mean
##  Min.   :0.00000   Min.   :0.00000     Min.   :0.1060   Min.   :0.04996       
##  1st Qu.:0.02956   1st Qu.:0.02031     1st Qu.:0.1619   1st Qu.:0.05770       
##  Median :0.06154   Median :0.03350     Median :0.1792   Median :0.06154       
##  Mean   :0.08880   Mean   :0.04892     Mean   :0.1812   Mean   :0.06280       
##  3rd Qu.:0.13070   3rd Qu.:0.07400     3rd Qu.:0.1957   3rd Qu.:0.06612       
##  Max.   :0.42680   Max.   :0.20120     Max.   :0.3040   Max.   :0.09744       
##    radius_se        texture_se      perimeter_se       area_se       
##  Min.   :0.1115   Min.   :0.3602   Min.   : 0.757   Min.   :  6.802  
##  1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850  
##  Median :0.3242   Median :1.1080   Median : 2.287   Median : 24.530  
##  Mean   :0.4052   Mean   :1.2169   Mean   : 2.866   Mean   : 40.337  
##  3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190  
##  Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200  
##  smoothness_se      compactness_se      concavity_se     concave.points_se 
##  Min.   :0.001713   Min.   :0.002252   Min.   :0.00000   Min.   :0.000000  
##  1st Qu.:0.005169   1st Qu.:0.013080   1st Qu.:0.01509   1st Qu.:0.007638  
##  Median :0.006380   Median :0.020450   Median :0.02589   Median :0.010930  
##  Mean   :0.007041   Mean   :0.025478   Mean   :0.03189   Mean   :0.011796  
##  3rd Qu.:0.008146   3rd Qu.:0.032450   3rd Qu.:0.04205   3rd Qu.:0.014710  
##  Max.   :0.031130   Max.   :0.135400   Max.   :0.39600   Max.   :0.052790  
##   symmetry_se       fractal_dimension_se  radius_worst   texture_worst  
##  Min.   :0.007882   Min.   :0.0008948    Min.   : 7.93   Min.   :12.02  
##  1st Qu.:0.015160   1st Qu.:0.0022480    1st Qu.:13.01   1st Qu.:21.08  
##  Median :0.018730   Median :0.0031870    Median :14.97   Median :25.41  
##  Mean   :0.020542   Mean   :0.0037949    Mean   :16.27   Mean   :25.68  
##  3rd Qu.:0.023480   3rd Qu.:0.0045580    3rd Qu.:18.79   3rd Qu.:29.72  
##  Max.   :0.078950   Max.   :0.0298400    Max.   :36.04   Max.   :49.54  
##  perimeter_worst    area_worst     smoothness_worst  compactness_worst
##  Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729  
##  1st Qu.: 84.11   1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720  
##  Median : 97.66   Median : 686.5   Median :0.13130   Median :0.21190  
##  Mean   :107.26   Mean   : 880.6   Mean   :0.13237   Mean   :0.25427  
##  3rd Qu.:125.40   3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910  
##  Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800  
##  concavity_worst  concave.points_worst symmetry_worst   fractal_dimension_worst
##  Min.   :0.0000   Min.   :0.00000      Min.   :0.1565   Min.   :0.05504        
##  1st Qu.:0.1145   1st Qu.:0.06493      1st Qu.:0.2504   1st Qu.:0.07146        
##  Median :0.2267   Median :0.09993      Median :0.2822   Median :0.08004        
##  Mean   :0.2722   Mean   :0.11461      Mean   :0.2901   Mean   :0.08395        
##  3rd Qu.:0.3829   3rd Qu.:0.16140      3rd Qu.:0.3179   3rd Qu.:0.09208        
##  Max.   :1.2520   Max.   :0.29100      Max.   :0.6638   Max.   :0.20750        
##     X          
##  Mode:logical  
##  NA's:569      
##                
##                
##                
##                
```

```
# getting the head of dataset
head(data)
```

```
##         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
```

Missing Values

It is important to check that the available dataset does not contain missing or null values. For this reason, a spot check is performed.

```
# check for missing values
colSums(is.na(data))
```

```
##                      id               diagnosis             radius_mean
##                       0                       0                       0
##            texture_mean          perimeter_mean               area_mean
##                       0                       0                       0
##         smoothness_mean        compactness_mean          concavity_mean
##                       0                       0                       0
##     concave.points_mean           symmetry_mean  fractal_dimension_mean
##                       0                       0                       0
##               radius_se              texture_se            perimeter_se
##                       0                       0                       0
##                 area_se           smoothness_se          compactness_se
##                       0                       0                       0
##            concavity_se       concave.points_se             symmetry_se
##                       0                       0                       0
##    fractal_dimension_se            radius_worst           texture_worst
##                       0                       0                       0
##         perimeter_worst              area_worst        smoothness_worst
##                       0                       0                       0
##       compactness_worst         concavity_worst    concave.points_worst
##                       0                       0                       0
##          symmetry_worst fractal_dimension_worst                       X
##                       0                       0                     569
```

There aren't missing values in the considered dataset, except for 32th feature 'X' that is full of NA. For this reason, we remove the attribute completely, as having no relevant information is not useful for the analysis.

```
data <- data %>% select(-X)
```

For the same reason, although it does not contain null values, the 'id' attribute is also removed.

```
data <- data %>% select(-id)
```

A check is made on the effective removal of these attributes.

```
colnames(data)
```

```
##  [1] "diagnosis"               "radius_mean"
##  [3] "texture_mean"            "perimeter_mean"
##  [5] "area_mean"               "smoothness_mean"
##  [7] "compactness_mean"        "concavity_mean"
##  [9] "concave.points_mean"     "symmetry_mean"
## [11] "fractal_dimension_mean"  "radius_se"
## [13] "texture_se"              "perimeter_se"
## [15] "area_se"                 "smoothness_se"
## [17] "compactness_se"          "concavity_se"
## [19] "concave.points_se"       "symmetry_se"
## [21] "fractal_dimension_se"    "radius_worst"
## [23] "texture_worst"           "perimeter_worst"
## [25] "area_worst"              "smoothness_worst"
## [27] "compactness_worst"       "concavity_worst"
## [29] "concave.points_worst"    "symmetry_worst"
## [31] "fractal_dimension_worst"
```

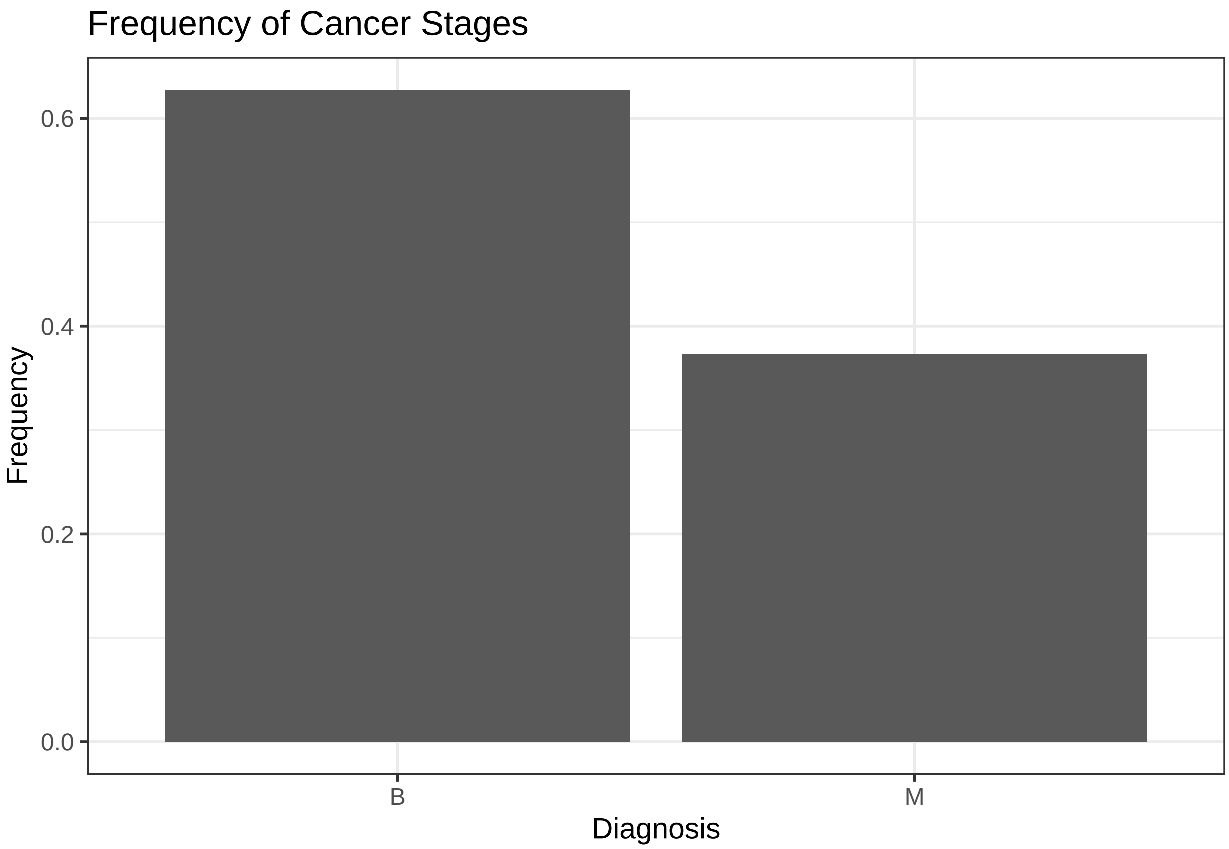
Descriptive Statistical Analysis

The descriptive statistical analysis section aims to explore the properties and relationships among the different variables in the dataset. This section will include an analysis of the frequency of cancer diagnoses (malignant or benign), as well as an analysis of the relationship between diagnoses and cancer cell attributes. The distribution of attributes and the relationships between them will also be screened, providing an overview of the fundamental properties of the dataset. This section will form the basis for the subsequent analysis of the relationships between the variables and their importance in breast cancer classification.

Benignant or Malignant diagnosis

A check is made on the frequency of the two types of breast cancer diagnosis, benign or malignant.

```
ggplot(data, aes(x = diagnosis)) +
  geom_bar(aes(y = (after_stat(count))/sum(after_stat(count)))) +
  scale_fill_manual(values = c("#0468BF","#D9A23D")) +
theme_bw() +
  labs(x = "Diagnosis", y = "Frequency", title = "Frequency of Cancer Stages")
```



It is possible to verify how the frequency of benign tumors is much higher than malignant ones.

Contingecy Tables & Chi-sq Test

Due to the fact that the response variable “diagnosis” it’s a categorial one, we can’t use correlation value to analyze the dipendency over the explanatory variables.

It is needed to create contingency tables and test the indipendence of the variable using the Chi-squared test:

H0: The two variables are independent. H1: The two variables relate to each other.

We will only keep the variables which are dependent to the response. Furthermore as we need to find which variable is more dependent than the other we create a list containing all the normalised chi-squared value.

```
# function for plotting a dataframe containing variables dependencies with chi-squared values

dependency_list <- function(df) {
    features_mean <- names(df)[2:11]
    features_se <- names(df)[12:21]
    features_worst <- names(df)[22:31]

    chivaluesN <- c(1)
    indipendentV <- c(FALSE)

    for (x in features_mean) {
        con <- table(cut(df[,x],breaks = 7),df$diagnosis)
        indipendent <- chisq.test(con)$p.value > 0.05
        chivalueN <- round(chisq.test(con)$statistic / length(df$diagnosis),digits = 4)
        indipendentV <- append(indipendentV, indipendent)
        chivaluesN <- append(chivaluesN, chivalueN)
    }

    for (x in features_se) {
        con <- table(cut(df[,x],breaks = 3),df$diagnosis)
        indipendent <- chisq.test(con)$p.value > 0.05
        chivalueN <- round(chisq.test(con)$statistic / length(df$diagnosis),digits = 4)
        indipendentV <- append(indipendentV, indipendent)
        chivaluesN <- append(chivaluesN, chivalueN)
    }

    for (x in features_worst) {
        con <- table(cut(df[,x],breaks = 7),df$diagnosis)
        indipendent <- chisq.test(con)$p.value > 0.05
        chivalueN <- round(chisq.test(con)$statistic / length(df$diagnosis),digits = 4)
        indipendentV <- append(indipendentV, indipendent)
        chivaluesN <- append(chivaluesN, chivalueN)
    }

    features <- names(df)[1:31]
    dv <- data.frame(features,chivaluesN,indipendentV)

    return(dv)
}

dependency_v <- dependency_list(data)
dependency_v <- dependency_v[dependency_v$features != "diagnosis",]
```

We discard all the values which are indipendente so all the TRUE, which correspond with a p-value > 0.05.

```
dependency_v <- dependency_v[dependency_v$indipendentV == "FALSE",]
```

On the remaining ones, we select those with a chi-squared normalised values > 0.25.

```
dependency_v <- dependency_v[dependency_v$chivaluesN > 0.25,]
dependency_v
```

```
##                features chivaluesN indipendentV
## 2           radius_mean     0.5635        FALSE
## 4        perimeter_mean     0.5964        FALSE
## 5             area_mean     0.5261        FALSE
## 7      compactness_mean     0.3666        FALSE
## 8        concavity_mean     0.5640         FALSE
## 9   concave.points_mean     0.6695         FALSE
## 22          radius_worst     0.6699         FALSE
## 24       perimeter_worst     0.6991         FALSE
## 25            area_worst     0.6543         FALSE
## 27     compactness_worst     0.3658         FALSE
## 28       concavity_worst     0.5181         FALSE
## 29  concave.points_worst     0.6833         FALSE
```

It is possible to discard:

- all the variables "*_se";
- texture_*;
- smoothness_*;
- symmerty_*;
- fractal_dimension_*.

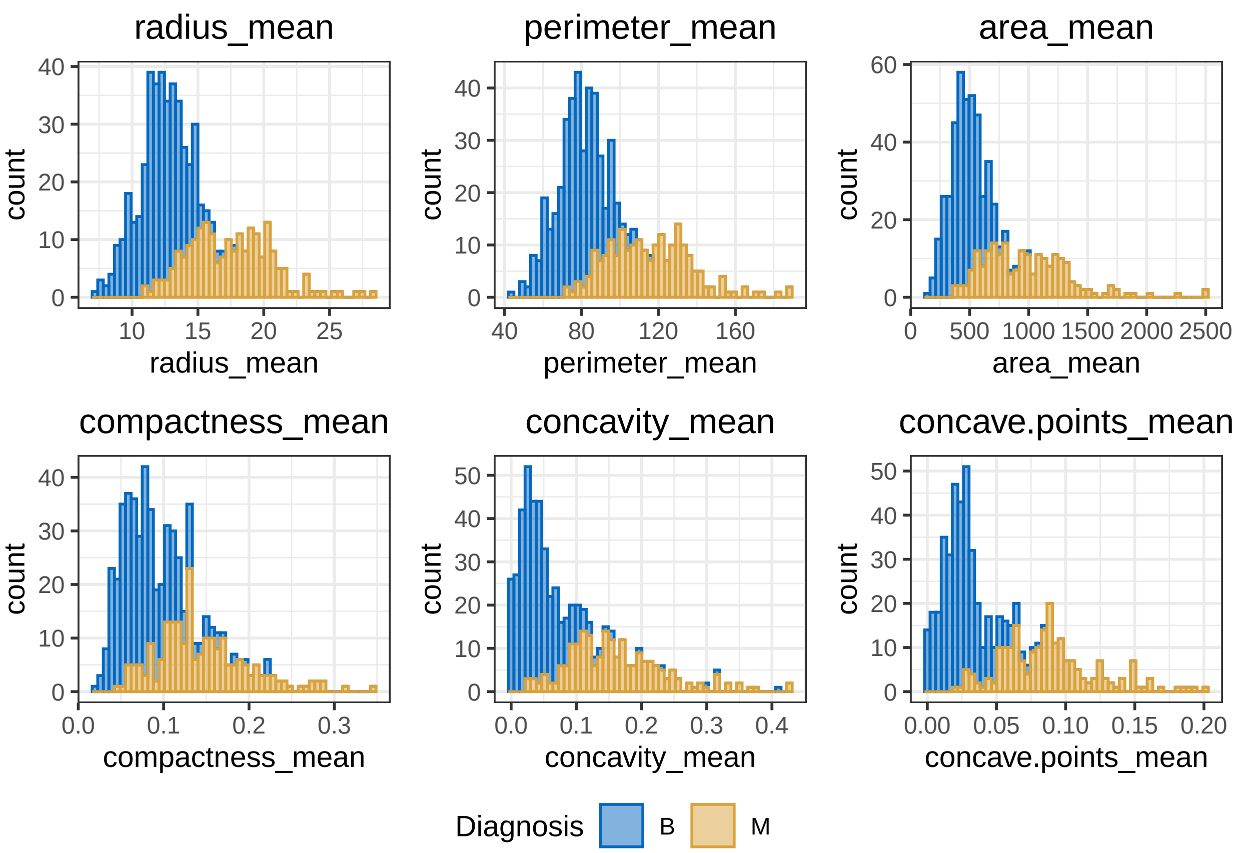
On the remaining features, a more in-depth analysis can be conducted.

A graphical way to see the features related to diagnosis

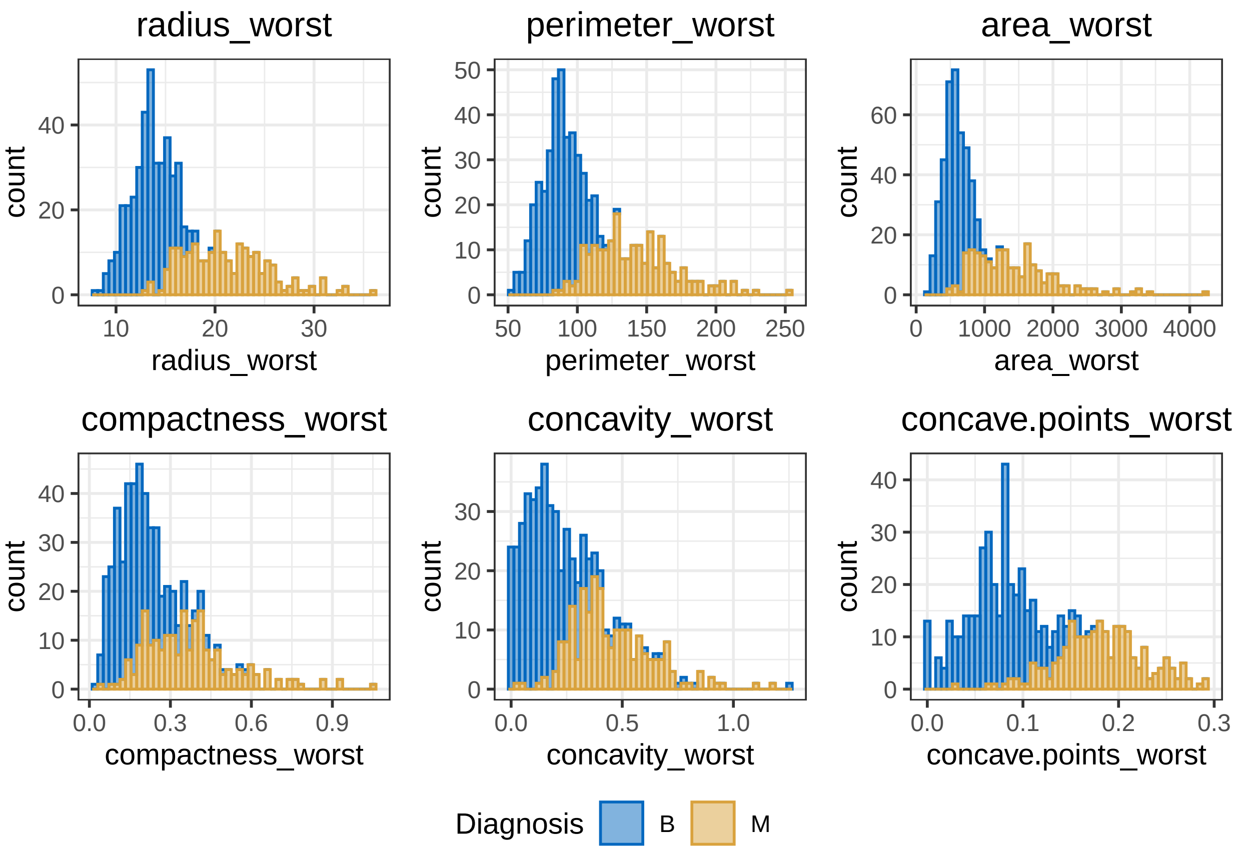
In the previous paragraph we saw which features are related to the target variable diagnosis. In this paragraph we attempt to explain it by a graphical way: comparison histograms between features and the distribution of malignant or benign tumor diagnosis are generated. These can be conveyed in order to make assertions about their distributions and significance.

```
#features_mean <- names(data)[2:11]

features_mean <- dependency_v$features

plots <- lapply(1:length(features_mean), function(x) {
  g <- ggplot(data, aes_string(x = features_mean[x],
                              fill = as.factor(data$diagnosis))) +
    geom_histogram(binwidth = (max(data[,features_mean[x]]) - min(data[,features_mean[x]]))/50,
                  alpha = 0.5, aes(color = as.factor(data$diagnosis))) +
    scale_fill_manual(values = c("#0468BF", "#D9A23D")) +
    scale_color_manual(values = c("#0468BF", "#D9A23D")) +
    ggtitle(features_mean[x]) +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5)) +
    labs(fill = "Diagnosis", color = "Diagnosis")
  return(g)
})

ggarrange(plotlist = plots,
          ncol = 3 , nrow = 2,
          common.legend = T,
          legend = "bottom")
```

```
## $`1`
```



```
##
## $`2`
```

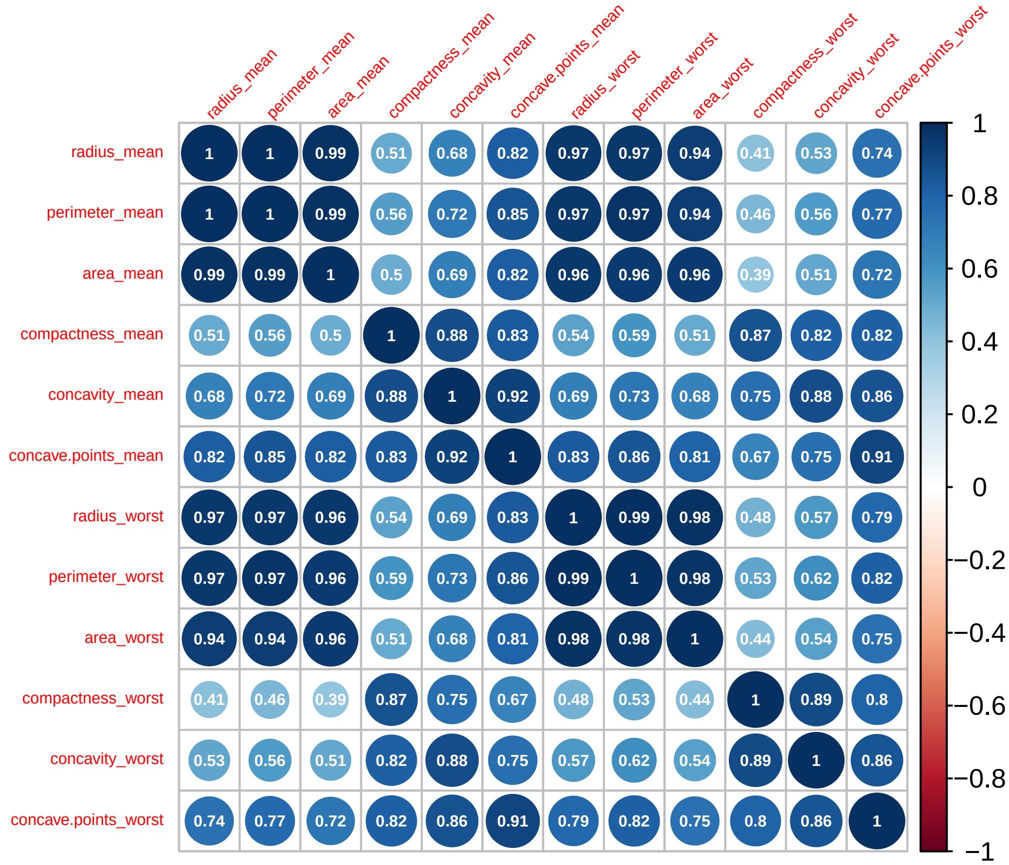


```
##
## attr(,"class")
## [1] "list"      "ggarrange"
```

Correlation map

A correlation map with a heatmap is generated between the selected variables.

```
feature_data_matrix <- subset(data, select = dependency_v$features) #%>% select(-diagnosis)
# Calculate the correlation matrix among features
corr_matrix <- cor(feature_data_matrix)

testRes = cor.mtest(feature_data_matrix, conf.level = 0.95)

corrplot(corr_matrix, p.mat = testRes$p, addCoef.col ='white',
tl.cex = 0.5, tl.srt = 45, number.cex = 0.5)
```



It is possible to verify the correlations among features to reduce their number, encreasing the explanability of the multilinear regression model we will face soon.

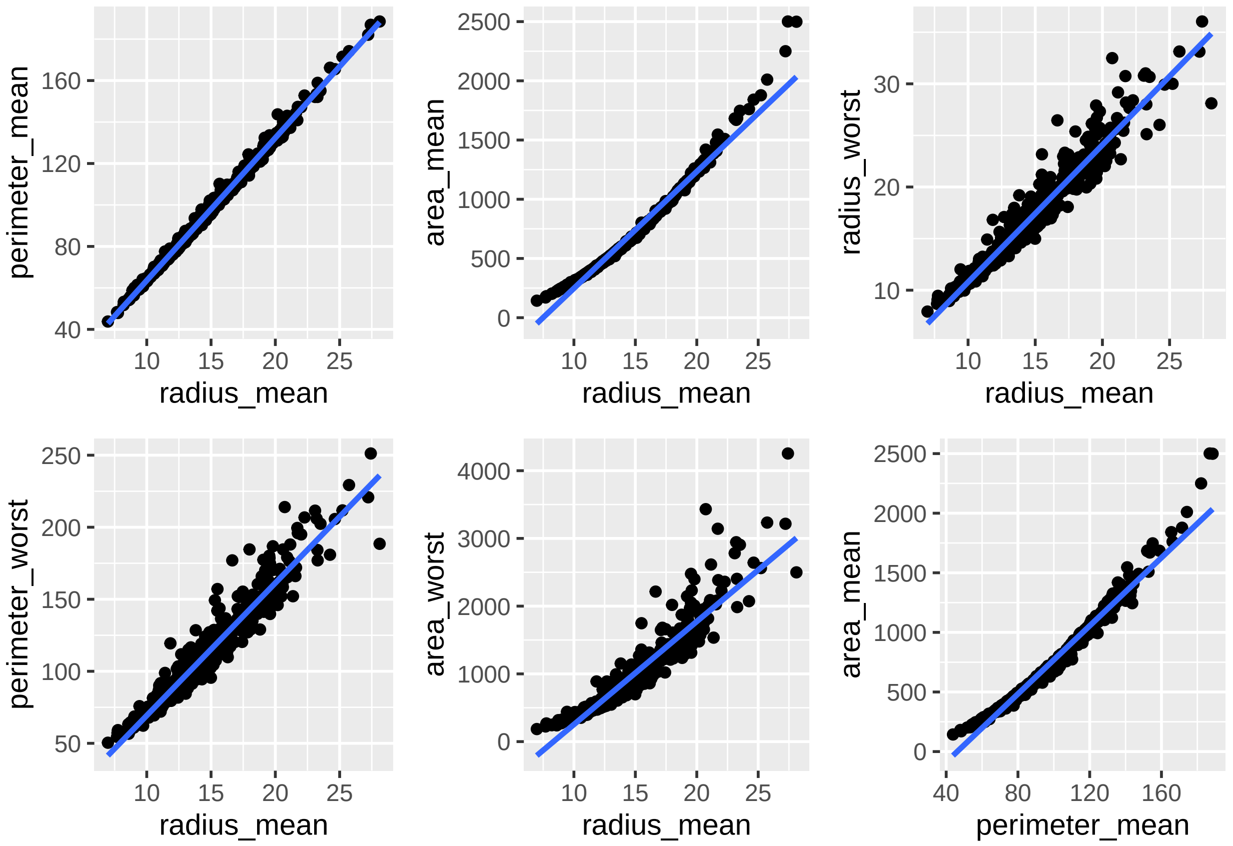
Covariance and Correlation

Following the scatter plot and the analysis above we can explore more the other variables.

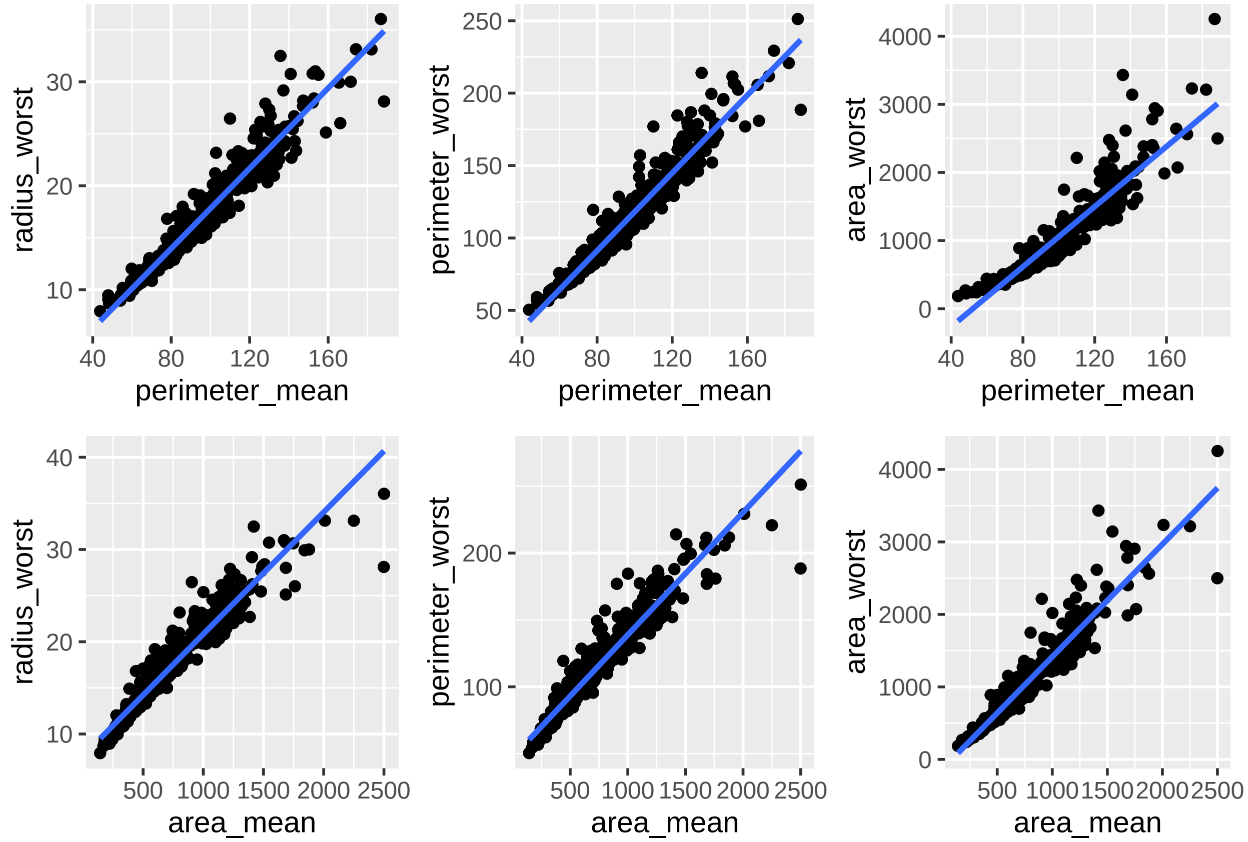
```
first_features <- data[c("radius_mean","perimeter_mean","area_mean","radius_worst",
"perimeter_worst","area_worst")]

cols <- colnames(first_features)
cols_combinations <- combn(cols, 2, FUN = list)

plot_first_list <- lapply(cols_combinations, function(cols) {
  x <- first_features[, cols[1]]
  y <- first_features[, cols[2]]
  ggplot(first_features, aes_string(x = cols[1], y = cols[2])) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE)
    # +  ggtitle(paste(cols[1], "vs", cols[2]))
})
```

```r
ggarrange(plotlist = plot_first_list, ncol = 3, nrow = 2)
```

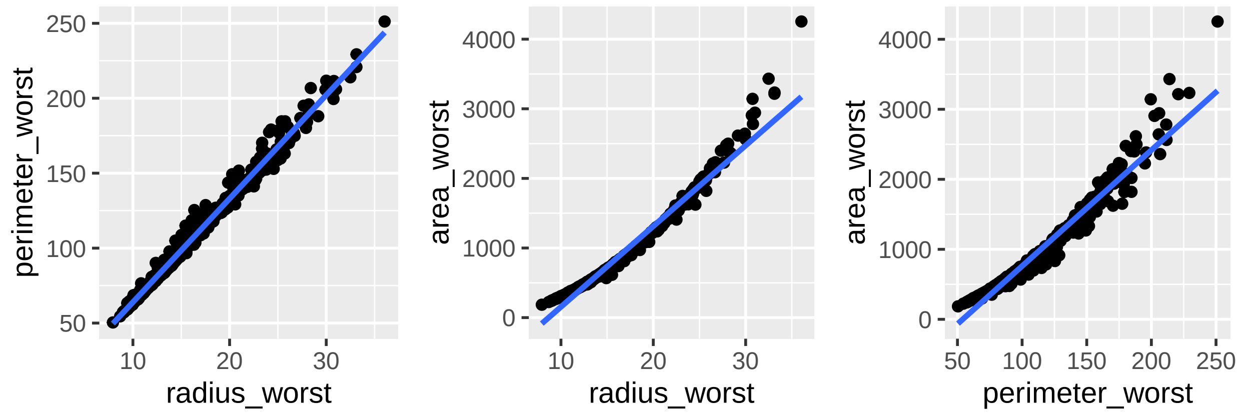
```
## $`1`
```



```
##
## $`2`
```



```
##
## $`3`
```



```
##
## attr(,"class")
## [1] "list"      "ggarrange"
```

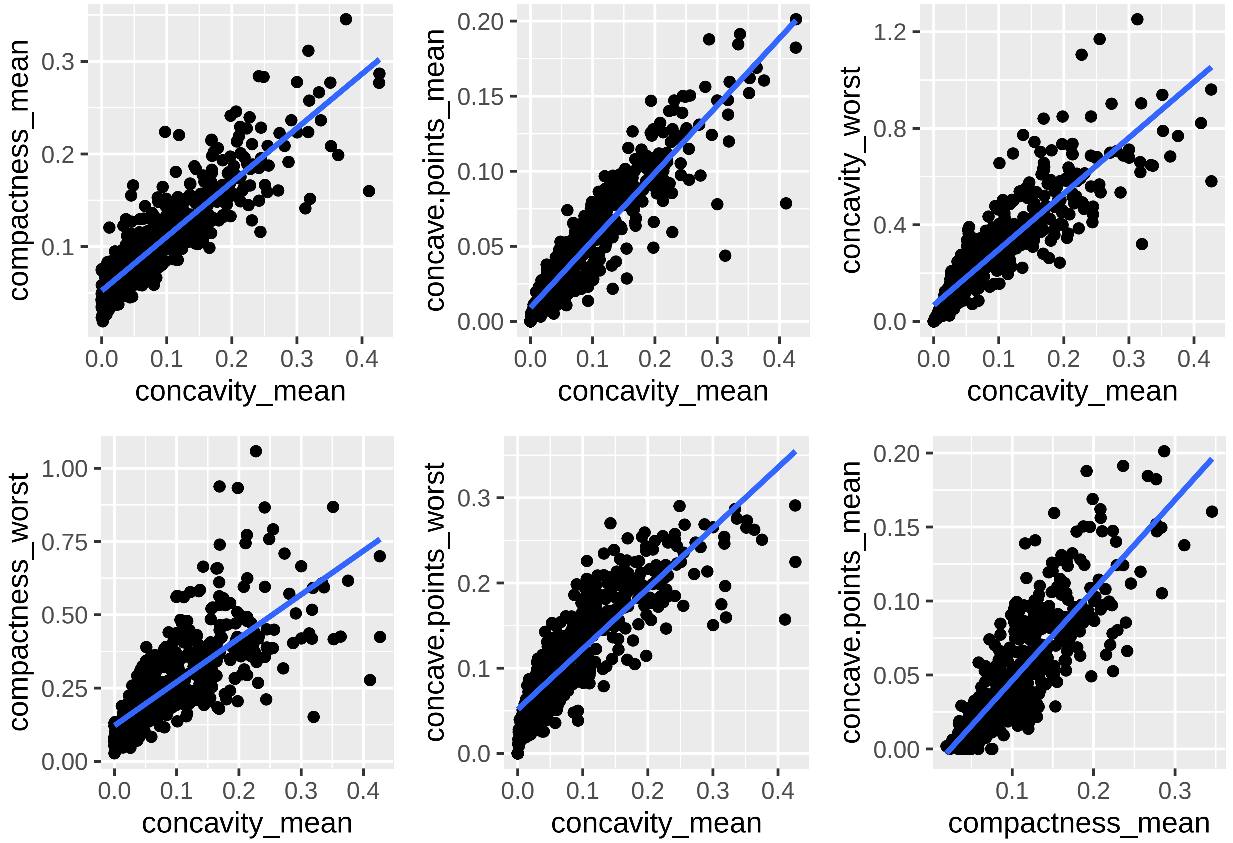
```
cols <- colnames(first_features)
cols_combinations <- combn(cols, 2, FUN = list)

first_corr_list <- lapply(cols_combinations, function(cols){
                    x <- first_features[, cols[1]]
                    y <- first_features[, cols[2]]
                    corr <- cor(x,y)
                    return(c(cols[1], cols[2], corr))
                  }

)

corr_features_df <- as.data.frame(do.call(rbind, first_corr_list))
colnames(corr_features_df) <- c("V1", "V2", "correlation")

corr_features_df <- corr_features_df %>% arrange(desc(correlation))
corr_features_df
```

```
##                  V1              V2       correlation
## 1      radius_mean  perimeter_mean 0.997855281493811
## 2     radius_worst perimeter_worst 0.993707916102951
## 3      radius_mean       area_mean 0.987357170056612
## 4   perimeter_mean       area_mean  0.98650680399139
## 5     radius_worst      area_worst 0.984014564459074
## 6  perimeter_worst      area_worst 0.977578091406388
## 7   perimeter_mean perimeter_worst 0.970386887042639
## 8      radius_mean    radius_worst 0.969538972611206
## 9   perimeter_mean    radius_worst 0.969476363466314
## 10     radius_mean perimeter_worst 0.965136513955988
## 11       area_mean    radius_worst 0.962746086047083
## 12       area_mean      area_worst   0.9592133256499
## 13       area_mean perimeter_worst 0.959119574355265
## 14  perimeter_mean      area_worst 0.941549808002307
## 15     radius_mean      area_worst 0.941082459586047
```

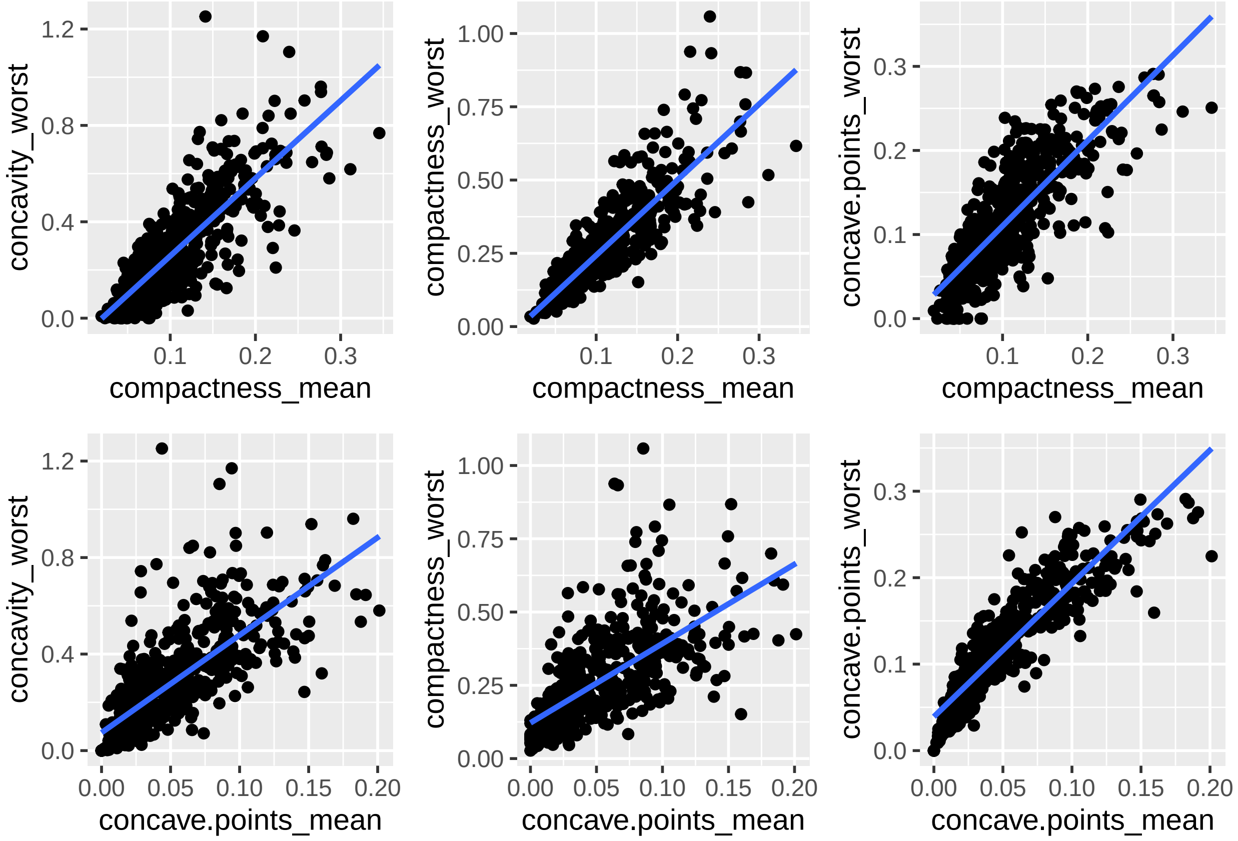
From the plot and the correlation values we can see a very strong correlation between all the features, so we can drop them all except for one. We select the feature which has the higher association with the response variable diagnosis so we select the perimeter_worst with a value of 0.6991. We now test the last remaining variables.

```
remained_features <- data[c("concavity_mean","compactness_mean","concave.points_mean",
"concavity_worst","compactness_worst","concave.points_worst")]

cols <- colnames(remained_features)
cols_combinations <- combn(cols, 2, FUN = list)

plot_remained_list <- lapply(cols_combinations, function(cols) {
  x <- remained_features[, cols[1]]
  y <- remained_features[, cols[2]]
  ggplot(remained_features, aes_string(x = cols[1], y = cols[2])) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE)
    # +     ggtitle(paste(cols[1], "vs", cols[2]))
})


ggarrange(plotlist = plot_remained_list, ncol = 3, nrow = 2)
```

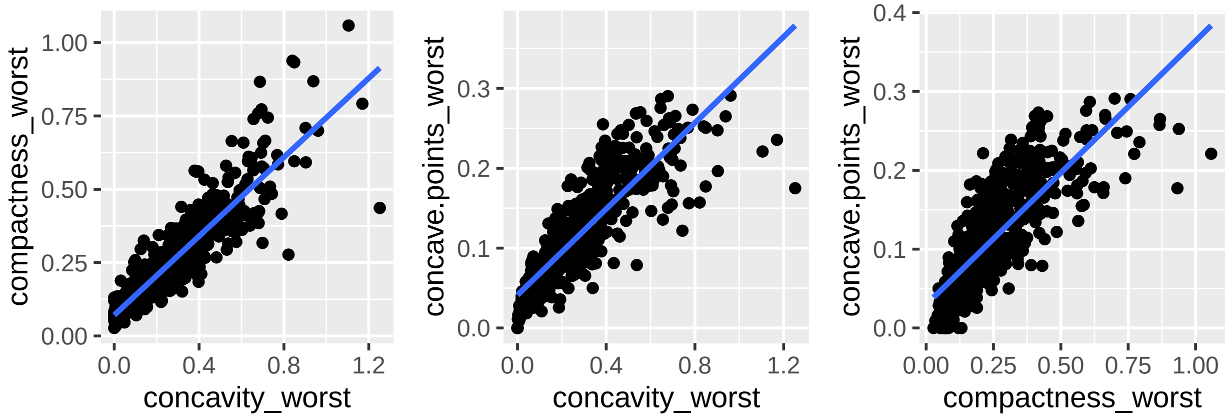
```
## $`1`
```


compactness_mean
0.3
0.2
0.1
0.0 0.1 0.2 0.3 0.4
concavity_mean
concave.points_mean
0.20
0.15
0.10
0.05
0.00
0.0 0.1 0.2 0.3 0.4
concavity_mean
concavity_worst
1.2
0.8
0.4
0.0
0.0 0.1 0.2 0.3 0.4
concavity_mean
compactness_worst
1.00
0.75
0.50
0.25
0.00
0.0 0.1 0.2 0.3 0.4
concavity_mean
concave.points_worst
0.3
0.2
0.1
0.0
0.0 0.1 0.2 0.3 0.4
concavity_mean
concave.points_mean
0.20
0.15
0.10
0.05
0.00
0.1 0.2 0.3
compactness_mean

```
##
## $`2`
```


concavity_worst
1.2
0.8
0.4
0.0
0.1 0.2 0.3
compactness_mean
compactness_worst
1.00
0.75
0.50
0.25
0.00
0.1 0.2 0.3
compactness_mean
concave.points_worst
0.3
0.2
0.1
0.0
0.1 0.2 0.3
compactness_mean
concavity_worst
1.2
0.8
0.4
0.0
0.00 0.05 0.10 0.15 0.20
concave.points_mean
compactness_worst
1.00
0.75
0.50
0.25
0.00
0.00 0.05 0.10 0.15 0.20
concave.points_mean
concave.points_worst
0.3
0.2
0.1
0.0
0.00 0.05 0.10 0.15 0.20
concave.points_mean

```
##
## $`3`
```



```
##
## attr(,"class")
## [1] "list"      "ggarrange"
```

```
cols <- colnames(remained_features)
cols_combinations <- combn(cols, 2, FUN = list)

remained_corr_list <- lapply(cols_combinations, function(cols){
                    x <- remained_features[, cols[1]]
                    y <- remained_features[, cols[2]]
                    corr <- cor(x,y)
                    return(c(cols[1], cols[2], corr))
                  }

)

corr_features_df <- as.data.frame(do.call(rbind, remained_corr_list))
colnames(corr_features_df) <- c("V1", "V2", "correlation")

corr_features_df <- corr_features_df %>% arrange(desc(correlation))
corr_features_df
```

```
##                     V1                   V2       correlation
## 1       concavity_mean  concave.points_mean 0.921391026378859
## 2  concave.points_mean concave.points_worst 0.910155314298593
## 3      concavity_worst    compactness_worst 0.892260898776469
## 4       concavity_mean      concavity_worst 0.884102639094382
## 5       concavity_mean     compactness_mean 0.883120670177251
## 6     compactness_mean    compactness_worst 0.865809039802263
## 7       concavity_mean concave.points_worst 0.861323033637951
## 8      concavity_worst concave.points_worst 0.855433860343999
## 9     compactness_mean  concave.points_mean 0.831135043133699
## 10    compactness_mean      concavity_worst 0.816275249800029
## 11    compactness_mean concave.points_worst 0.815573223569065
## 12   compactness_worst concave.points_worst 0.801080364635253
## 13      concavity_mean    compactness_worst 0.754968015906397
## 14 concave.points_mean      concavity_worst 0.752399497574964
## 15 concave.points_mean    compactness_worst 0.667453676825712
```

From the analysis and the plot we can see that concave.point_worst and concave.points_mean are strongly correlated so we keep only concave.point_worst which has the higher association with diagnosis (0.6833).

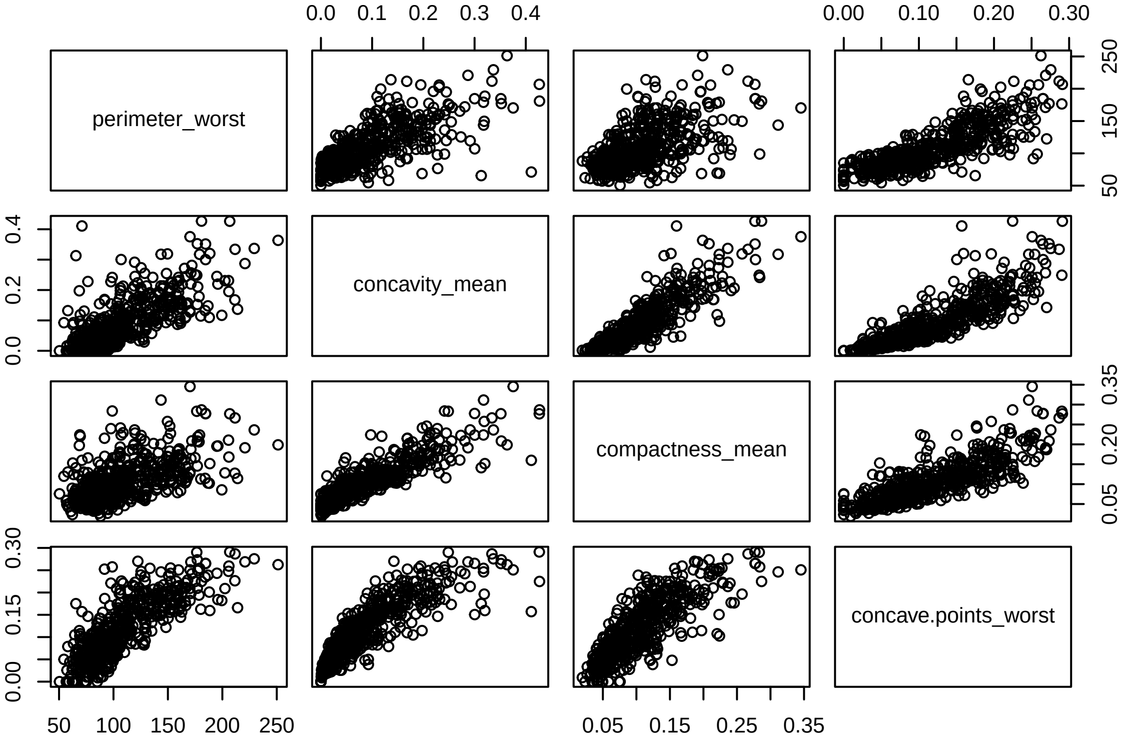
Same goes for concavity and compacteness mean with they respective worst have a correlation value less than 0.6 but still strongly correlated. We keep concavity_mean and compactness_mean whose have the higher association with the diagnosis (0.5640,0.3666).

As a summary, we plot the correlation matrix of selected features.

```
cor(data[c("perimeter_worst","concavity_mean","compactness_mean","concave.points_worst")])
```

```
##                      perimeter_worst concavity_mean compactness_mean
## perimeter_worst            1.0000000      0.7295649        0.5902104
## concavity_mean             0.7295649      1.0000000        0.8831207
## compactness_mean           0.5902104      0.8831207        1.0000000
## concave.points_worst       0.8163221      0.8613230        0.8155732
##                      concave.points_worst
## perimeter_worst                 0.8163221
## concavity_mean                  0.8613230
## compactness_mean                0.8155732
## concave.points_worst            1.0000000
```

```
pairs(data[c("perimeter_worst","concavity_mean","compactness_mean","concave.points_worst")])
```



```
data_fs <- data[c("perimeter_worst","concavity_mean","compactness_mean","concave.points_worst","diagnosis"
```

Inferential Statistics

The inferential statistical analysis section focuses on using statistical methods to make inferences about the properties of populations based on the data in the dataset. This section aims to identify relationships between variables and determine the importance of individual variables in breast cancer classification. Hypothesis testing will be used to confirm or reject relationships between variables. This section will provide a deeper understanding of the properties of the dataset and their relationship to breast cancer diagnosis. Finally, regression techniques will be used to determine the relationship between attributes and diagnoses and to identify the most important attributes for tumor classification.

Test

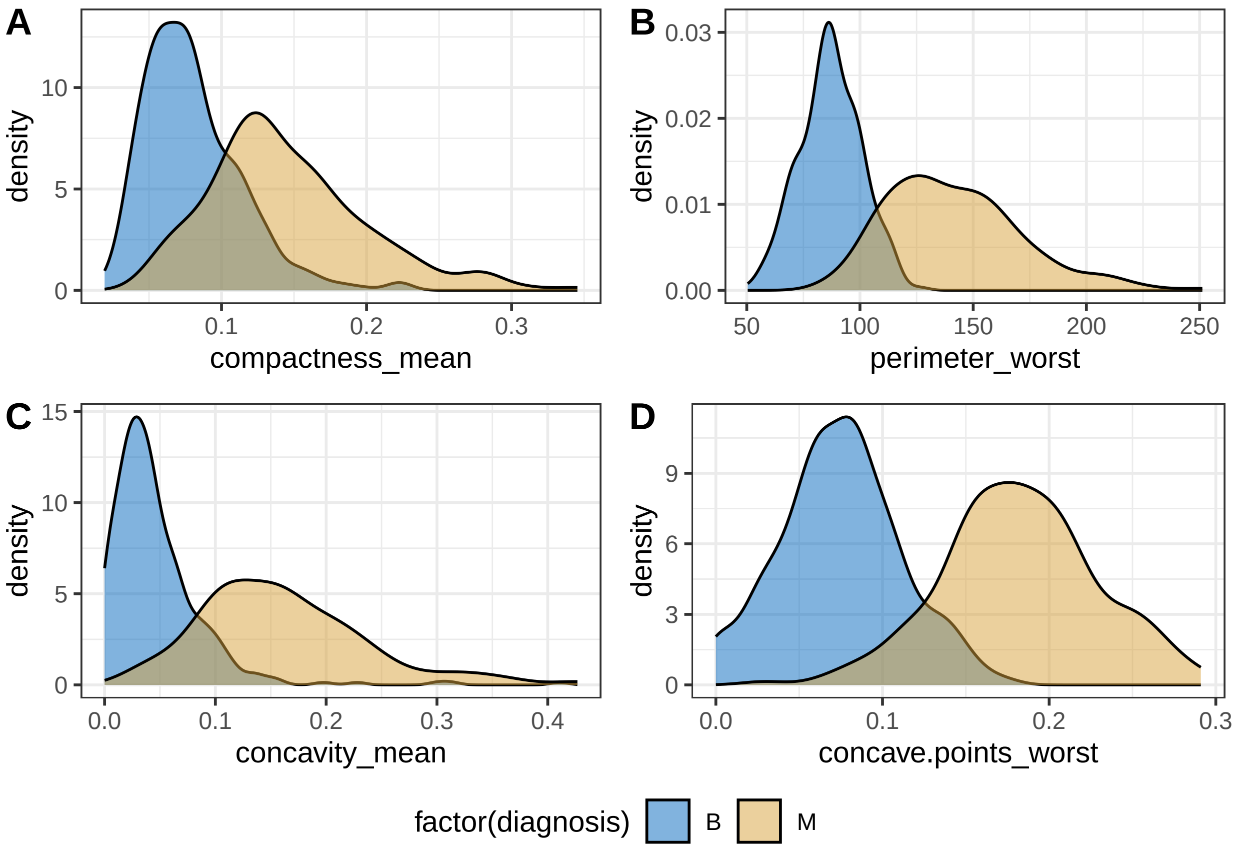
We want to determine whether the features selected are significantly different between healthy (benign) and diseased patients (malignant).

A t-test assigns a “t” test statistic value to each feature. A good feature, represented by little to no overlap of the distributions and a large difference in means, would have a high “t” value.

Firstly, we divide the dataset.

```r
data$diagnosis <- ifelse(data$diagnosis=="M",1,0)

mdf <- data[data$diagnosis == 1, ] # group of Malignant tumor
bdf <- data[data$diagnosis == 0, ] # group of Benign tumor
```

~~~
```r
cm <- ggplot(data_fs, aes(x=compactness_mean, group=diagnosis,fill=factor(diagnosis))) +
geom_density(alpha=0.5) +
scale_fill_manual(values = c("#0468BF","#D9A23D")) +
theme_bw()

pw <- ggplot(data_fs, aes(x=perimeter_worst, group=diagnosis,fill=factor(diagnosis))) +
geom_density(alpha=0.5) +
scale_fill_manual(values = c("#0468BF","#D9A23D")) +
theme_bw()

cw <- ggplot(data_fs, aes(x=concavity_mean, group=diagnosis,fill=factor(diagnosis))) +
geom_density(alpha=0.5) +
scale_fill_manual(values = c("#0468BF","#D9A23D")) +
theme_bw()

cp <- ggplot(data_fs, aes(x=concave.points_worst, group=diagnosis,fill=factor(diagnosis))) +
geom_density(alpha=0.5) +
scale_fill_manual(values = c("#0468BF","#D9A23D")) +
theme_bw()

ggarrange(cm, pw, cw, cp,
          labels = c("A", "B", "C", "D"),
          ncol = 2 , nrow = 2,
          common.legend = T,
          legend = "bottom")
~~~



```
t.test(mdf$perimeter_worst,bdf$perimeter_worst, alternative="two.sided", var.equal=FALSE,conf.level=0.95)
```

```
##
##  Welch Two Sample t-test
##
## data:  mdf$perimeter_worst and bdf$perimeter_worst
## t = 25.332, df = 264.69, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  50.13888 58.58991
## sample estimates:
## mean of x mean of y
## 141.37033  87.00594
```

```
t.test(mdf$concavity_mean,bdf$concavity_mean, alternative="two.sided", var.equal=FALSE,conf.level=0.95)
```

```
##
##  Welch Two Sample t-test
##
## data:  mdf$concavity_mean and bdf$concavity_mean
## t = 20.332, df = 296.43, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1036135 0.1258207
## sample estimates:
##  mean of x  mean of y
## 0.16077472 0.04605762
```

```
t.test(mdf$compactness_mean,bdf$compactness_mean, alternative="two.sided", var.equal=FALSE,conf.level=0.95
```

```
##
##  Welch Two Sample t-test
##
## data:  mdf$compactness_mean and bdf$compactness_mean
## t = 15.818, df = 310.39, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.05700496 0.07320136
## sample estimates:
##  mean of x  mean of y
## 0.14518778 0.08008462
```

```
t.test(mdf$concave.points_worst,bdf$concave.points_worst, alternative="two.sided", var.equal=FALSE,conf.le
```

```
##
##  Welch Two Sample t-test
##
## data:  mdf$concave.points_worst and bdf$concave.points_worst
## t = 29.118, df = 360.42, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1005128 0.1150732
## sample estimates:
##  mean of x  mean of y
## 0.18223731 0.07444434
```

From the t value we can say that the better feature which helps us to distinguish malignant and benign is the **concave.point__worst** with a t value of 29.

Multiple Linear Regression Model

We use the four selected features to apply the multiple linear regression model.

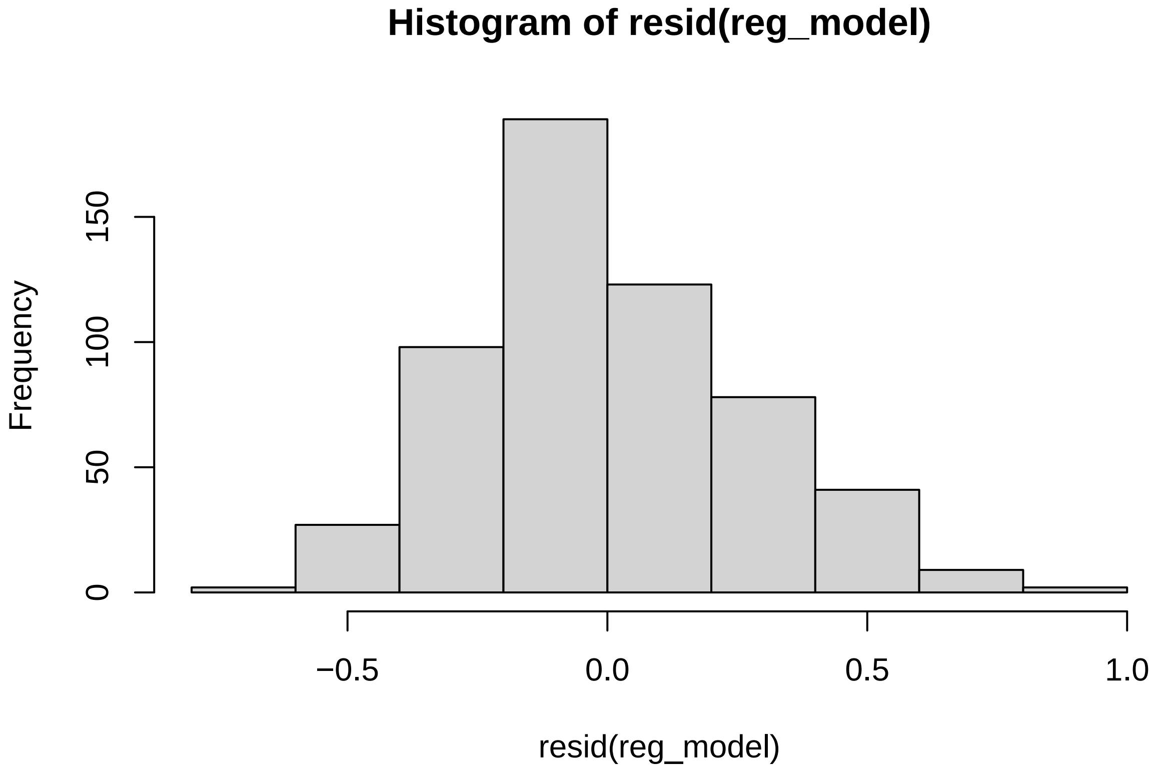
```
reg_model <- lm(data$diagnosis ~ perimeter_worst
                                + concavity_mean
                                + compactness_mean
                                + concave.points_worst
                                , data=data )

summary(reg_model)
```

```
##
## Call:
## lm(formula = data$diagnosis ~ perimeter_worst + concavity_mean +
##     compactness_mean + concave.points_worst, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74737 -0.17695 -0.03746  0.16821  0.98866
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -0.5643106  0.0562812 -10.027  < 2e-16 ***
## perimeter_worst       0.0053352  0.0006316   8.447 2.55e-16 ***
## concavity_mean        0.5774504  0.3737160   1.545   0.1229
## compactness_mean     -1.0910288  0.5063888  -2.155   0.0316 *
## concave.points_worst  3.7274789  0.4402767   8.466  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2717 on 564 degrees of freedom
## Multiple R-squared:  0.6871, Adjusted R-squared:  0.6848
## F-statistic: 309.6 on 4 and 564 DF,  p-value: < 2.2e-16
```

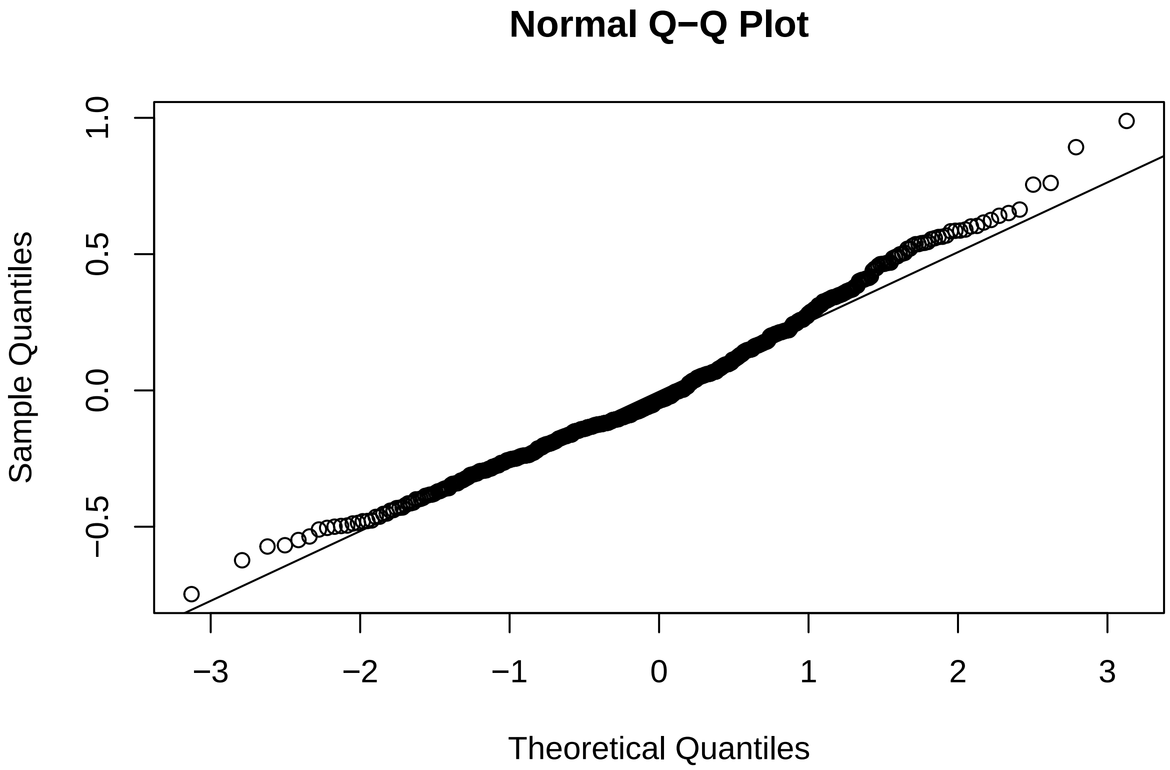
Regression Diagnostics

We check if the residuals of our linear regression are normally distributed.

```
hist(resid(reg_model))
```



```
qqnorm(resid(reg_model))
qqline(resid(reg_model))
```



As we can see from the histogram and the qqplot, the distribution of the residuals seems almost normal.

To confirm that, a check with the Shapiro–Wilk test is conducted.

- H0: there is no difference between the residuals distribution and a normal distribution;
- H1: the two distribution are not equal.

```
shapiro.test(resid(reg_model))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(reg_model)
## W = 0.98517, p-value = 1.538e-05
```

Although the test returns a very high coefficient, having a p-value < 0.05 we can't accept the null Hypothesis and have to conclude that the result is not statistically relevant.

Bibliography

[1] https://cancer.ca/en/treatments/tests-and-procedures/fine-needle-aspiration-fna