Name of dataset - Breast cancer analysis

About Dataset

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. n the 3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server: ftp ftp.cs.wisc.edu cd math-prog/cpo-dataset/machine-learn/WDBC/

Also can be found on UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

Attribute Information:

1) ID number 2) Diagnosis (M = malignant, B = benign) 3-32)

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 (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)

The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.

All feature values are recoded with four significant digits.

Missing attribute values: none

Class distribution: 357 benign, 212 malignant

import liberties

```
library(party)
library(psych)
library(dplyr)
library(data.table)
library(ggplot2)
library(plotly)
library(expss)
library(pander)
library(forcats)
library(stringr)
library(caTools)
library(VIM)
library(caret)
require(reshape2)
library(GGally)
library(corrplot)
library(factoextra)
library(gridExtra)
library(C50)
library(highcharter)
library(rpart)
library(e1071)
library(ranger)
library(epiR)
library(randomForest)
library(party)
library(class)
library(kknn)
library(gbm)
library(ada)
library(c3)
```

See The Data and Data Type :

```
> head(data)
# A tibble: 6 × 33
        id diagn…¹ radiu…² textu…³ perim…⁴ area_…⁵ smoot…⁶ compa…⁷ conca…⁸ conca…⁹ symme…ˣ fract…ˣ
     <dbl> <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1  8.42e5 M          18.0    10.4   123.    1001    0.118   0.278   0.300   0.147   0.242   0.0787
2  8.43e5 M          20.6    17.8   133.    1326    0.0847  0.0786  0.0869  0.0702  0.181   0.0567
3  8.43e7 M          19.7    21.2   130     1203    0.110   0.160   0.197   0.128   0.207   0.0600
4  8.43e7 M          11.4    20.4    77.6    386.   0.142   0.284   0.241   0.105   0.260   0.0974
5  8.44e7 M          20.3    14.3   135.    1297    0.100   0.133   0.198   0.104   0.181   0.0588
6  8.44e5 M          12.4    15.7    82.6    477.   0.128   0.17    0.158   0.0809  0.209   0.0761
```

```
> str(data)
spec_tbl_df [568 × 33] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ id                     : num [1:568] 842302 842517 84300903 84348301 84358402 ...
 $ diagnosis              : chr [1:568] "M" "M" "M" "M" ...
 $ radius_mean            : num [1:568] 18 20.6 19.7 11.4 20.3 ...
 $ texture_mean           : num [1:568] 10.4 17.8 21.2 20.4 14.3 ...
 $ perimeter_mean         : num [1:568] 122.8 132.9 130 77.6 135.1 ...
 $ area_mean              : num [1:568] 1001 1326 1203 386 1297 ...
 $ smoothness_mean        : num [1:568] 0.1184 0.0847 0.1096 0.1425 0.1003 ...
 $ compactness_mean       : num [1:568] 0.2776 0.0786 0.1599 0.2839 0.1328 ...
 $ concavity_mean         : num [1:568] 0.3001 0.0869 0.1974 0.2414 0.198 ...
 $ concave points_mean    : num [1:568] 0.1471 0.0702 0.1279 0.1052 0.1043 ...
 $ symmetry_mean          : num [1:568] 0.242 0.181 0.207 0.26 0.181 ...
 $ fractal_dimension_mean : num [1:568] 0.0787 0.0567 0.06 0.0974 0.0588 ...
 $ radius_se              : num [1:568] 1.095 0.543 0.746 0.496 0.757 ...
 $ texture_se             : num [1:568] 0.905 0.734 0.787 1.156 0.781 ...
 $ perimeter_se           : num [1:568] 8.59 3.4 4.58 3.44 5.44 ...
 $ area_se                : num [1:568] 153.4 74.1 94 27.2 94.4 ...
 $ smoothness_se          : num [1:568] 0.0064 0.00522 0.00615 0.00911 0.01149 ...
 $ compactness_se         : num [1:568] 0.049 0.0131 0.0401 0.0746 0.0246 ...
 $ concavity_se           : num [1:568] 0.0537 0.0186 0.0383 0.0566 0.0569 ...
 $ concave points_se      : num [1:568] 0.0159 0.0134 0.0206 0.0187 0.0188 ...
 $ symmetry_se            : num [1:568] 0.03 0.0139 0.0225 0.0596 0.0176 ...
 $ fractal_dimension_se   : num [1:568] 0.00619 0.00353 0.00457 0.00921 0.00511 ...
 $ radius_worst           : num [1:568] 25.4 25 23.6 14.9 22.5 ...
 $ texture_worst          : num [1:568] 17.3 23.4 25.5 26.5 16.7 ...
 $ perimeter_worst        : num [1:568] 184.6 158.8 152.5 98.9 152.2 ...
 $ area_worst             : num [1:568] 2019 1956 1709 568 1575 ...
 $ smoothness_worst       : num [1:568] 0.162 0.124 0.144 0.21 0.137 ...
 $ compactness_worst      : num [1:568] 0.666 0.187 0.424 0.866 0.205 ...
 $ concavity_worst        : num [1:568] 0.712 0.242 0.45 0.687 0.4 ...
 $ concave points_worst   : num [1:568] 0.265 0.186 0.243 0.258 0.163 ...
 $ symmetry_worst         : num [1:568] 0.46 0.275 0.361 0.664 0.236 ...
 $ fractal_dimension_worst: num [1:568] 0.1189 0.089 0.0876 0.173 0.0768 ...
 $ ...33                  : logi [1:568] NA NA NA NA NA NA ...
```

> The dataset has 33 columns, but one is completely empty, so I removed it...

```
> data=data[,-33]
> str(data)
tibble [568 × 32] (S3: tbl_df/tbl/data.frame)
 $ id                     : num [1:568] 842302 842517 84300903 84348301 84358402 ...
 $ diagnosis              : chr [1:568] "M" "M" "M" "M" ...
 $ radius_mean            : num [1:568] 18 20.6 19.7 11.4 20.3 ...
 $ texture_mean           : num [1:568] 10.4 17.8 21.2 20.4 14.3 ...
 $ perimeter_mean         : num [1:568] 122.8 132.9 130 77.6 135.1 ...
 $ area_mean              : num [1:568] 1001 1326 1203 386 1297 ...
 $ smoothness_mean        : num [1:568] 0.1184 0.0847 0.1096 0.1425 0.1003 ...
 $ compactness_mean       : num [1:568] 0.2776 0.0786 0.1599 0.2839 0.1328 ...
 $ concavity_mean         : num [1:568] 0.3001 0.0869 0.1974 0.2414 0.198 ...
 $ concave points_mean    : num [1:568] 0.1471 0.0702 0.1279 0.1052 0.1043 ...
 $ symmetry_mean          : num [1:568] 0.242 0.181 0.207 0.26 0.181 ...
 $ fractal_dimension_mean : num [1:568] 0.0787 0.0567 0.06 0.0974 0.0588 ...
 $ radius_se              : num [1:568] 1.095 0.543 0.746 0.496 0.757 ...
 $ texture_se             : num [1:568] 0.905 0.734 0.787 1.156 0.781 ...
 $ perimeter_se           : num [1:568] 8.59 3.4 4.58 3.44 5.44 ...
 $ area_se                : num [1:568] 153.4 74.1 94 27.2 94.4 ...
 $ smoothness_se          : num [1:568] 0.0064 0.00522 0.00615 0.00911 0.01149 ...
 $ compactness_se         : num [1:568] 0.049 0.0131 0.0401 0.0746 0.0246 ...
 $ concavity_se           : num [1:568] 0.0537 0.0186 0.0383 0.0566 0.0569 ...
 $ concave points_se      : num [1:568] 0.0159 0.0134 0.0206 0.0187 0.0188 ...
 $ symmetry_se            : num [1:568] 0.03 0.0139 0.0225 0.0596 0.0176 ...
 $ fractal_dimension_se   : num [1:568] 0.00619 0.00353 0.00457 0.00921 0.00511 ...
 $ radius_worst           : num [1:568] 25.4 25 23.6 14.9 22.5 ...
 $ texture_worst          : num [1:568] 17.3 23.4 25.5 26.5 16.7 ...
 $ perimeter_worst        : num [1:568] 184.6 158.8 152.5 98.9 152.2 ...
 $ area_worst             : num [1:568] 2019 1956 1709 568 1575 ...
 $ smoothness_worst       : num [1:568] 0.162 0.124 0.144 0.21 0.137 ...
 $ compactness_worst      : num [1:568] 0.666 0.187 0.424 0.866 0.205 ...
 $ concavity_worst        : num [1:568] 0.712 0.242 0.45 0.687 0.4 ...
 $ concave points_worst   : num [1:568] 0.265 0.186 0.243 0.258 0.163 ...
 $ symmetry_worst         : num [1:568] 0.46 0.275 0.361 0.664 0.236 ...
 $ fractal_dimension_worst: num [1:568] 0.1189 0.089 0.0876 0.173 0.0768 ...
```

Missing Data

```
> missing_values = data %>% summarize_all(funs(sum(is.na(.))/n()))
> missing_values
# A tibble: 1 × 32
     id diagnosis radiu…¹ textu…² perim…³ area_…⁴ smoot…⁵ compa…⁶ conca…⁷ conca…⁸ symme…⁹ fract…ˣ
  <dbl>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1     0         0       0       0       0       0       0       0       0       0       0       0
# … with 20 more variables: radius_se <dbl>, texture_se <dbl>, perimeter_se <dbl>,
#   area_se <dbl>, smoothness_se <dbl>, compactness_se <dbl>, concavity_se <dbl>,
#   `concave points_se` <dbl>, symmetry_se <dbl>, fractal_dimension_se <dbl>,
#   radius_worst <dbl>, texture_worst <dbl>, perimeter_worst <dbl>, area_worst <dbl>,
#   smoothness_worst <dbl>, compactness_worst <dbl>, concavity_worst <dbl>,
#   `concave points_worst` <dbl>, symmetry_worst <dbl>, fractal_dimension_worst <dbl>, and
#   abbreviated variable names ¹radius_mean, ²texture_mean, ³perimeter_mean, ⁴area_mean, …
# ℹ Use `colnames()` to see all variable names
```

> There are no Missing Data

To design a machine learning algorithm that is able to correctly classify whether the tumor is benign or malignant.

Target variable

```
> table(data$diagnosis)

  B   M
356 212
> prop.table(table(data$diagnosis))*100

       B        M
62.67606 37.32394
>
```

```
> data$diagnosis=factor(data$diagnosis, labels=c('B','M'))
> prop.table(table(data$diagnosis))*100

       B        M
62.67606 37.32394
```

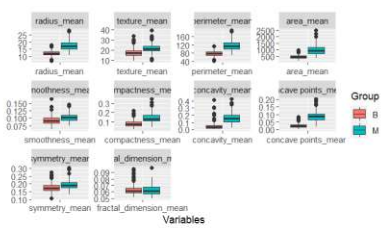
> Target variable is a character-type variable, so I convert it into a factor…

Descriptive analysis

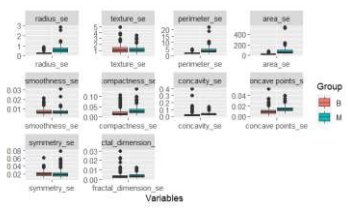
```
> psych::describeBy(data[3:32], group=data$diagnosis)

 Descriptive statistics by group
group: B
                         vars   n   mean     sd median trimmed    mad    min    max   range
radius_mean                 1 356  12.16   1.77  12.20   12.18   1.69   6.98  17.85   10.87
texture_mean                2 356  17.90   3.99  17.38   17.50   3.42   9.71  33.81   24.10
perimeter_mean              3 356  78.16  11.71  78.22   78.22  11.20  43.79 114.60   70.81
area_mean                   4 356 463.58 133.64 458.55  460.00 127.28 143.50 992.10  848.60
smoothness_mean             5 356   0.09   0.01   0.09    0.09   0.01   0.06   0.16    0.10
compactness_mean            6 356   0.08   0.03   0.08    0.08   0.03   0.02   0.22    0.20
concavity_mean              7 356   0.05   0.04   0.04    0.04   0.03   0.00   0.41    0.41
concave points_mean         8 356   0.03   0.02   0.02    0.02   0.01   0.00   0.09    0.09
symmetry_mean               9 356   0.17   0.02   0.17    0.17   0.02   0.11   0.27    0.17
fractal_dimension_mean     10 356   0.06   0.01   0.06    0.06   0.01   0.05   0.10    0.04
radius_se                  11 356   0.28   0.11   0.26    0.27   0.10   0.11   0.88    0.77
texture_se                 12 356   1.22   0.59   1.11    1.15   0.51   0.36   4.88    4.52
perimeter_se               13 356   2.00   0.77   1.85    1.91   0.68   0.76   5.12    4.36
area_se                    14 356  21.14   8.86  19.66   20.15   7.16   6.80  77.11   70.31
smoothness_se              15 356   0.01   0.00   0.01    0.01   0.00   0.00   0.02    0.02
compactness_se             16 356   0.02   0.02   0.02    0.02   0.01   0.00   0.11    0.10
concavity_se               17 356   0.03   0.03   0.02    0.02   0.01   0.00   0.40    0.40
concave points_se          18 356   0.01   0.01   0.01    0.01   0.00   0.00   0.05    0.05
symmetry_se                19 356   0.02   0.01   0.02    0.02   0.01   0.01   0.06    0.05
fractal_dimension_se       20 356   0.00   0.00   0.00    0.00   0.00   0.00   0.03    0.03
radius_worst               21 356  13.39   1.97  13.35   13.40   2.00   7.93  19.82   11.89
texture_worst              22 356  23.50   5.49  22.81   23.07   5.17  12.02  41.78   29.76
perimeter_worst            23 356  87.08  13.47  86.94   87.03  13.55  50.41 127.10   76.69
area_worst                 24 356 559.71 163.10 547.60  554.34 160.94 185.20 1210.00 1024.80
smoothness_worst           25 356   0.13   0.02   0.13    0.12   0.02   0.07   0.20    0.13
compactness_worst          26 356   0.18   0.09   0.17    0.17   0.09   0.03   0.58    0.56
concavity_worst            27 356   0.17   0.14   0.14    0.15   0.10   0.00   1.25    1.25
concave points_worst       28 356   0.07   0.04   0.07    0.07   0.03   0.00   0.17    0.17
symmetry_worst             29 356   0.27   0.04   0.27    0.27   0.04   0.16   0.42    0.27
fractal_dimension_worst    30 356   0.08   0.01   0.08    0.08   0.01   0.06   0.15    0.09
                         skew kurtosis   se
radius_mean             -0.06    -0.06 0.09
texture_mean             0.98     1.21 0.21
perimeter_mean          -0.04    -0.07 0.62
area_mean                0.35     0.28 7.08
smoothness_mean          0.73     1.77 0.00
compactness_mean         1.20     2.21 0.00
concavity_mean           3.45    20.43 0.00
concave points_mean      0.92     1.00 0.00
symmetry_mean            0.65     1.24 0.00
fractal_dimension_mean   1.63     4.35 0.00
radius_se                1.50     3.93 0.01
texture_se               1.64     5.29 0.03
perimeter_se             1.18     1.69 0.04
area_se                  1.63     5.18 0.47
smoothness_se            1.49     2.95 0.00
compactness_se           2.19     5.85 0.00
concavity_se             6.28    57.85 0.00
concave points_se        2.15    10.52 0.00
symmetry_se              1.37     3.26 0.00
fractal_dimension_se     4.29    27.09 0.00
radius_worst            -0.03    -0.18 0.10
texture_worst            0.72     0.43 0.29
perimeter_worst          0.02    -0.25 0.71
area_worst               0.37     0.16 8.64
smoothness_worst         0.34     0.38 0.00
compactness_worst        1.09     1.68 0.00
concavity_worst          2.54    12.37 0.01
concave points_worst     0.12    -0.19 0.00
symmetry_worst           0.24     0.25 0.00
fractal_dimension_worst  1.40     3.07 0.00

group: M
                         vars   n    mean     sd median trimmed    mad    min    max   range
radius_mean                 1 212   17.46   3.20  17.33   17.32   3.36  10.95  28.11   17.16
texture_mean                2 212   21.60   3.78  21.46   21.43   3.25  10.38  39.28   28.90
perimeter_mean              3 212  115.37  21.85 114.20  114.19  23.17  71.90 188.50  116.60
area_mean                   4 212  978.38 367.94 932.00  945.98 366.57 361.60 2501.00 2139.40
smoothness_mean             5 212    0.10   0.01   0.10    0.10   0.01   0.07   0.14    0.07
compactness_mean            6 212    0.15   0.05   0.13    0.14   0.04   0.05   0.35    0.30
concavity_mean              7 212    0.16   0.08   0.15    0.15   0.07   0.02   0.43    0.40
concave points_mean         8 212    0.09   0.03   0.09    0.09   0.03   0.02   0.20    0.18
symmetry_mean               9 212    0.19   0.03   0.19    0.19   0.03   0.13   0.30    0.17
fractal_dimension_mean     10 212    0.06   0.01   0.06    0.06   0.01   0.05   0.10    0.05
radius_se                  11 212    0.61   0.35   0.55    0.57   0.27   0.19   2.87    2.68
texture_se                 12 212    1.21   0.48   1.10    1.16   0.39   0.36   3.57    3.21
perimeter_se               13 212    4.32   2.57   3.68    3.96   1.76   1.33  21.98   20.65
area_se                    14 212   72.67  61.36  58.45   63.78  38.70  13.99 542.20  528.21
smoothness_se              15 212    0.01   0.00   0.01    0.01   0.00   0.00   0.03    0.03
compactness_se             16 212    0.03   0.02   0.03    0.03   0.01   0.01   0.14    0.13
concavity_se               17 212    0.04   0.02   0.04    0.04   0.02   0.01   0.14    0.13
concave points_se          18 212    0.02   0.01   0.01    0.01   0.00   0.01   0.04    0.04
symmetry_se                19 212    0.02   0.01   0.02    0.02   0.01   0.01   0.08    0.07
fractal_dimension_se       20 212    0.00   0.00   0.00    0.00   0.00   0.00   0.01    0.01
radius_worst               21 212   21.13   4.28  20.59   20.84   4.57  12.84  36.04   23.20
texture_worst              22 212   29.32   5.43  28.94   29.12   5.00  16.67  49.54   32.87
perimeter_worst            23 212  141.37  29.46 138.00  139.09  30.10  85.10 251.20  166.10
area_worst                 24 212 1422.29 597.97 1303.00 1352.81 548.27 508.10 4254.00 3745.90
smoothness_worst           25 212    0.14   0.02   0.14    0.14   0.02   0.09   0.22    0.13
compactness_worst          26 212    0.37   0.17   0.36    0.36   0.15   0.05   1.06    1.01
concavity_worst            27 212    0.45   0.18   0.40    0.44   0.17   0.02   1.17    1.15
concave points_worst       28 212    0.18   0.05   0.18    0.18   0.04   0.03   0.29    0.26
symmetry_worst             29 212    0.32   0.07   0.31    0.32   0.06   0.16   0.66    0.51
fractal_dimension_worst    30 212    0.09   0.02   0.09    0.09   0.02   0.06   0.21    0.15
                         skew kurtosis   se
radius_mean              0.49     0.31 0.22
texture_mean             0.69     2.25 0.26
perimeter_mean           0.60     0.52 1.50
area_mean                1.10     2.17 25.27
smoothness_mean          0.47     0.36 0.00
compactness_mean         0.82     0.77 0.00
concavity_mean           0.89     1.06 0.01
concave points_mean      0.73     0.65 0.00
symmetry_mean            0.80     1.21 0.00
fractal_dimension_mean   0.88     1.23 0.00
radius_se                2.48    11.66 0.02
texture_se               1.51     3.70 0.03
perimeter_se             2.77    13.42 0.18
area_se                  4.28    28.04 4.21
smoothness_se            3.90    26.68 0.00
compactness_se           1.83     5.40 0.00
concavity_se             1.60     3.55 0.00
concave points_se        1.42     3.56 0.00
symmetry_se              2.48     7.76 0.00
fractal_dimension_se     1.65     3.80 0.00
radius_worst             0.62     0.26 0.29
texture_worst            0.53     1.06 0.37
perimeter_worst          0.74     0.54 2.02
area_worst               1.29     2.54 41.07
smoothness_worst         0.43     0.73 0.00
compactness_worst        1.09     1.60 0.01
concavity_worst          0.90     1.31 0.01
concave points_worst    -0.09     0.08 0.00
symmetry_worst           1.13     2.36 0.01
fractal_dimension_worst  1.34     3.77 0.00
```

> In General, Malignant Diagnoses Have Higher Scores in All Variables

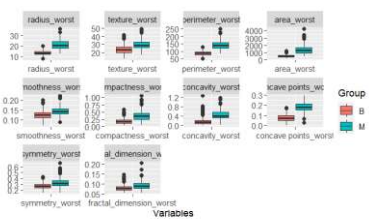
```
#Mean
df.m <- melt(data[,-c(1,13:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable, scales="free")+ xlab("Variables") + ylab("")+ guides(fill=guide_legend(title="Group"))
p

#Se
df.m <- melt(data[,-c(1,3:12,23:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable, scales="free")+ xlab("Variables") + ylab("")+ guides(fill=guide_legend(title="Group"))
p

#Worst
df.m <- melt(data[,c(2,23:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable, scales="free")+ xlab("Variables") + ylab("")+ guides(fill=guide_legend(title="Group"))
p
```



Mean



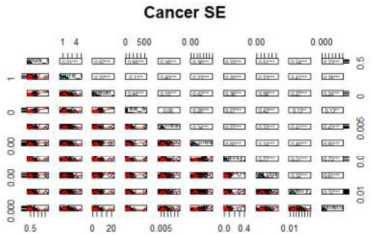
Se

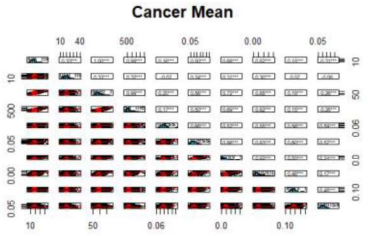


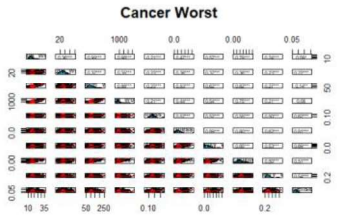
Worst

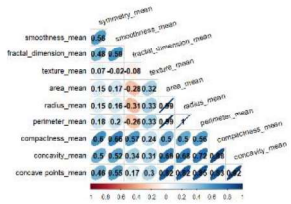
Correlations

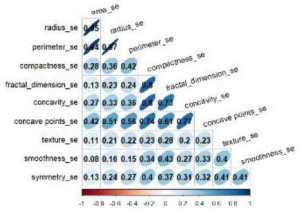
```
pairs.panels(data[,c(3:12)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points = TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer Mean")

pairs.panels(data[,c(13:22)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points = TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer SE")

pairs.panels(data[,c(23:32)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points = TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer Worst")


w.corr<-cor(data[,c(3:12)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col = 'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col = 'black',tl.srt=15)

w.corr<-cor(data[,c(13:22)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col = 'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col = 'black',tl.srt=15)

w.corr<-cor(data[,c(23:32)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col = 'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col = 'black',tl.srt=15)
```

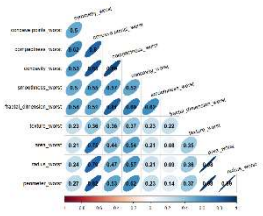


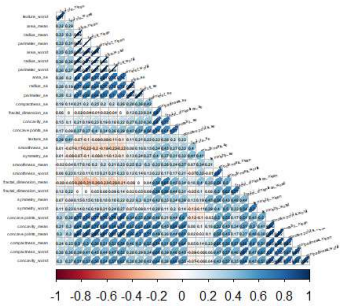






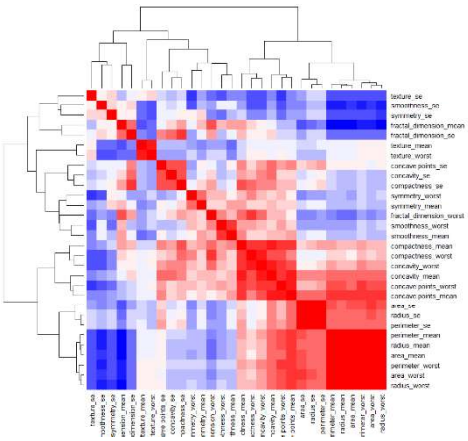






> We see that there are extremely high correlations between some of the variables…

Heatmap & clustering



Training and testing datasets

```
dataset=data
head(dataset)
```

Dataset is divided into two datasets: training (70%) and testing (30%)

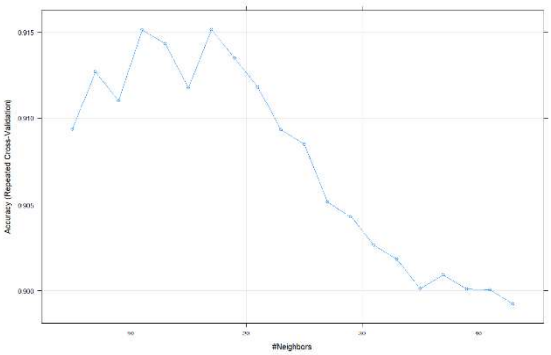
```
set.seed(123)
smp_size <- floor(0.70 * nrow(dataset))
train_ind <- sample(seq_len(nrow(dataset)), size = smp_size)
train <- dataset[train_ind, ]
test <- dataset[-train_ind, ]
```

Let's check the target variable.

```
> prop.table(table(train$diagnosis))*100

        B         M
64.98741 35.01259
> prop.table(table(test$diagnosis))*100

        B         M
57.30994 42.69006
```

K-NN

```
control <- trainControl(method='repeatedcv',
                        number=10,
                        repeats=3)
knnFit <- train(diagnosis ~ ., data = train[,-1], method = "knn", trControl = control,tuneLength = 20)
plot(knnFit)
```



Testing the model

```
> knnPredict <- predict(knnFit,newdata = test )
> cm_knn<-confusionMatrix(knnPredict, test$diagnosis )
> cm_knn
Confusion Matrix and Statistics

          Reference
Prediction  B  M
         B 97  7
         M  1 66

               Accuracy : 0.9532
                 95% CI : (0.9099, 0.9796)
    No Information Rate : 0.5731
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9034

 Mcnemar's Test P-Value : 0.0771

            Sensitivity : 0.9898
            Specificity : 0.9041
         Pos Pred Value : 0.9327
         Neg Pred Value : 0.9851
             Prevalence : 0.5731
         Detection Rate : 0.5673
   Detection Prevalence : 0.6082
      Balanced Accuracy : 0.9470

       'Positive' Class : B
```

Naive Bayes

```
> #Training the model
> learn_nb <- naiveBayes(train[,-c(1,2)], train$diagnosis)
> #Testing the model
> pre_nb <- predict(learn_nb, test[,-c(1,2)])
> cm_nb <- confusionMatrix(pre_nb, test$diagnosis)
> cm_nb
Confusion Matrix and Statistics

          Reference
Prediction  B  M
         B 96  6
         M  2 67

               Accuracy : 0.9532
                 95% CI : (0.9099, 0.9796)
    No Information Rate : 0.5731
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9037

 Mcnemar's Test P-Value : 0.2888

            Sensitivity : 0.9796
            Specificity : 0.9178
         Pos Pred Value : 0.9412
         Neg Pred Value : 0.9710
             Prevalence : 0.5731
         Detection Rate : 0.5614
   Detection Prevalence : 0.5965
      Balanced Accuracy : 0.9487

       'Positive' Class : B
```

Decision Tree

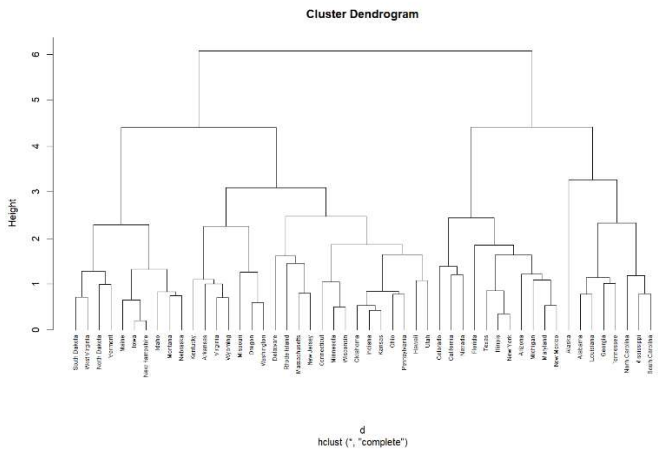
```
> #Training the model
> learn_ct <- ctree(diagnosis~., data=train[,-1], controls=ctree_control(maxdepth=2))
> #Testing the model
> pre_ct   <- predict(learn_ct, test[,-c(1,2)])
> cm_ct    <- confusionMatrix(pre_ct, test$diagnosis)
> cm_ct
Confusion Matrix and Statistics

          Reference
Prediction  B  M
         B 95 10
         M  3 63

               Accuracy : 0.924
                 95% CI : (0.8735, 0.9589)
    No Information Rate : 0.5731
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.8427

 Mcnemar's Test P-Value : 0.09609

            Sensitivity : 0.9694
            Specificity : 0.8630
         Pos Pred Value : 0.9048
         Neg Pred Value : 0.9545
             Prevalence : 0.5731
         Detection Rate : 0.5556
   Detection Prevalence : 0.6140
      Balanced Accuracy : 0.9162

       'Positive' Class : B
```

K-means

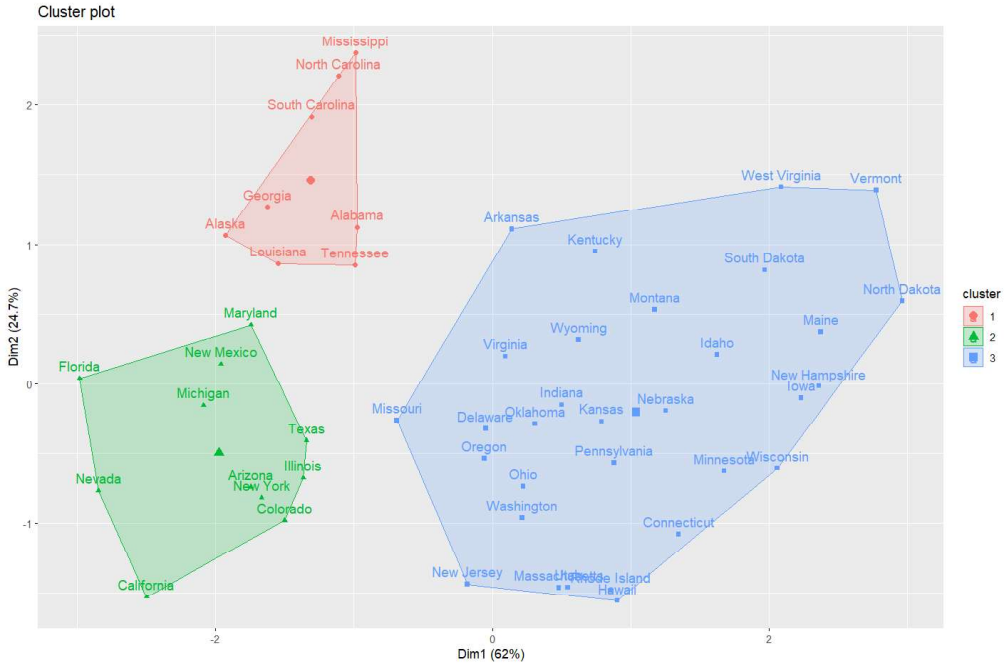
```
> #Training the model
> predict.kmeans <- function(newdata, object){
+   centers <- object$centers
+   n_centers <- nrow(centers)
+   dist_mat <- as.matrix(dist(rbind(centers, newdata)))
+   dist_mat <- dist_mat[-seq(n_centers), seq(n_centers)]
+   max.col(-dist_mat)
+ }
> learn_kmeans <- kmeans(train[,-c(1,2)], centers=2)
> pre_kmeans <- predict.kmeans(test[,-c(1,2)],learn_kmeans)
> pre_kmeans <- factor(ifelse(pre_kmeans == 1,"B","M"))
> cm_kmeans <- confusionMatrix(pre_kmeans, test$diagnosis)
> cm_kmeans
Confusion Matrix and Statistics

          Reference
Prediction  B  M
         B 98 21
         M  0 52

               Accuracy : 0.8772
                 95% CI : (0.8184, 0.9223)
    No Information Rate : 0.5731
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7395

 Mcnemar's Test P-Value : 1.275e-05

            Sensitivity : 1.0000
            Specificity : 0.7123
         Pos Pred Value : 0.8235
         Neg Pred Value : 1.0000
             Prevalence : 0.5731
         Detection Rate : 0.5731
   Detection Prevalence : 0.6959
      Balanced Accuracy : 0.8562

       'Positive' Class : B
```

Hierarchical

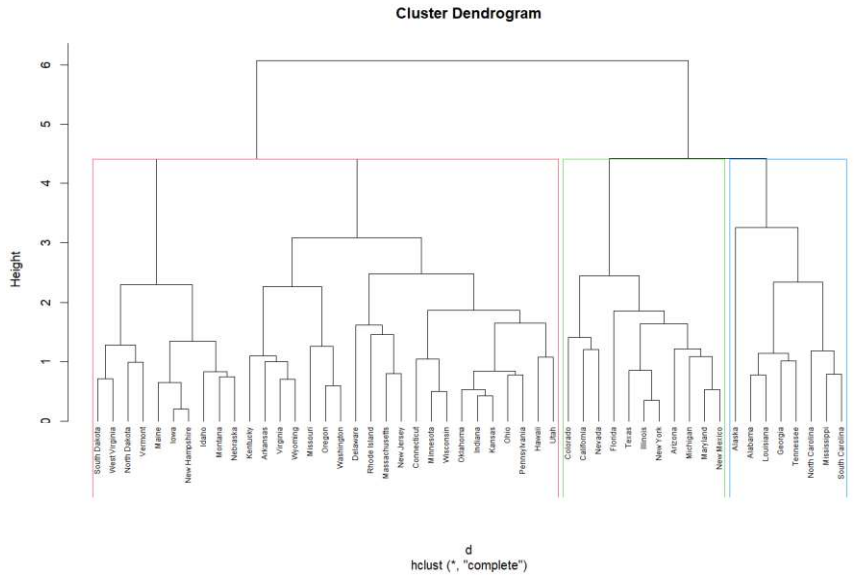
```
# Dissimilarity matrix
d <- dist(df, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc1 <- hclust(d, method = "complete" )

# Plot the obtained dendrogram
plot(hc1, cex = 0.6, hang = -1)
```



```r
# Plot the obtained dendrogram with
# rectangle borders for k clusters
plot(hc1, cex = 0.6, hang = -1)
rect.hclust(hc1, k = 3, border = 2:4)
```



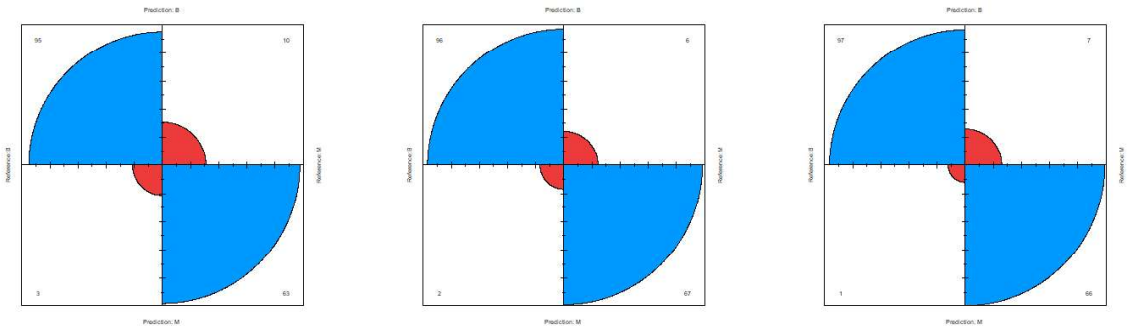

```r
# Cut tree into 3 groups
sub_grps <- cutree(hc1, k = 3)

# Visualize the result in a scatter plot
fviz_cluster(list(data = df, cluster = sub_grps))
```



Best ML Model Out Of Them :

```
col <- c("#ed3b3b", "#0099ff")
par(mfrow=c(2,3))
fourfoldplot(cm_ct$table, color = col, conf.level = 0, margin = 1, main=paste("CTree (",round(cm_ct$overall[1]*100),"%)",sep=""))
fourfoldplot(cm_nb$table, color = col, conf.level = 0, margin = 1, main=paste("NaiveBayes (",round(cm_nb$overall[1]*100),"%)",sep=""))
fourfoldplot(cm_knn$table, color = col, conf.level = 0, margin = 1, main=paste("K-nn (",round(cm_ranger$overall[1]*100),"%)",sep=""))
```

CTree (92%) NaiveBayes (95%) K-nn (98%)



```
> cm_knn$table
          Reference
Prediction  B  M
         B 97  7
         M  1 66
```

> out of them, k-nn is best predictor for this dataset… it has 98% accuracy….

```
#CODING PAGE

#import liberies
library(party)
library(psych)
library(dplyr)
library(data.table)
library(ggplot2)
library(plotly)
library(expss)
library(pander)
library(forcats)
library(stringr)
library(caTools)
library(VIM)
library(caret)
require(reshape2)
library(GGally)
library(corrplot)
library(factoextra)
library(gridExtra)
library(C50)
library(highcharter)
library(rpart)
library(e1071)
library(ranger)
library(epiR)
library(randomForest)
library(party)
library(class)
library(kknn)
library(gbm)
library(ada)
library(c3)

#The dataset has 33 columns, but one is completely empty, so I removed
it...
head(data)
str(data)
data=data[,-33]
str(data)
head(data)

#To design a machine learning algorithm that is able to correctly classify
whether the tumor is benign or malignant.

#missing data
missing_values = data %>% summarize_all(funs(sum(is.na(.))/n()))
missing_values
aggr(data, prop = FALSE, combined = TRUE, numbers = TRUE, sortVars = TRUE,
sortCombs = TRUE)

#Target variable
table(data$diagnosis)
prop.table(table(data$diagnosis))*100

#Target variable is a character-type variable, so I convert it into a
factor.
```

```
data$diagnosis=factor(data$diagnosis, labels=c('B','M'))
prop.table(table(data$diagnosis))*100

#Descriptive analysis
psych::describeBy(data[3:32], group=data$diagnosis)
'in general, malignant diagnoses have higher scores in all variables.'

#Mean
df.m <- melt(data[,-c(1,13:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable,
scales="free")+ xlab("Variables") + ylab("")+
guides(fill=guide_legend(title="Group"))
p

#Se
df.m <- melt(data[,-c(1,3:12,23:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable,
scales="free")+ xlab("Variables") + ylab("")+
guides(fill=guide_legend(title="Group"))
p

#Worst
df.m <- melt(data[,c(2,23:32)], id.var = "diagnosis")
p <- ggplot(data = df.m, aes(x=variable, y=value)) +
  geom_boxplot(aes(fill=diagnosis)) + facet_wrap( ~ variable,
scales="free")+ xlab("Variables") + ylab("")+
guides(fill=guide_legend(title="Group"))
p


#Correlations
pairs.panels(data[,c(3:12)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points
= TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer
Mean")

pairs.panels(data[,c(13:22)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points
= TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer
SE")

pairs.panels(data[,c(23:32)], method="pearson",
             hist.col = "#1fbbfa", density=TRUE, ellipses=TRUE, show.points
= TRUE,
             pch=1, lm=TRUE, cex.cor=1, smoother=F, stars = T, main="Cancer
Worst")


w.corr<-cor(data[,c(3:12)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col =
'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col =
'black',tl.srt=15)

w.corr<-cor(data[,c(13:22)],method="pearson")
```

```
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col =
'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col =
'black',tl.srt=15)

w.corr<-cor(data[,c(23:32)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col =
'black',type='lower', number.cex = 1,tl.cex = 1, diag=F,tl.col =
'black',tl.srt=15)

#We see that there are extremely high correlations between some of the
variables

w.corr<-cor(data[,c(3:32)],method="pearson")
corrplot(w.corr, order='hclust', method='ellipse',addCoef.col =
'black',type='lower', number.cex = 0.25,tl.cex = 0.25, diag=F,tl.col =
'black',tl.srt=15)

col<-colorRampPalette(c('blue','white','red'))(20)
heatmap(x=w.corr, col=col,symm=T)

#Training and testing datasets
dataset=data
head(dataset)

#Dataset is divided into two datasets: training (70%) and testing (30%)
set.seed(123)
smp_size <- floor(0.70 * nrow(dataset))
train_ind <- sample(seq_len(nrow(dataset)), size = smp_size)
train <- dataset[train_ind, ]
test <- dataset[-train_ind, ]

#Let's check the target variable.
prop.table(table(train$diagnosis))*100
prop.table(table(test$diagnosis))*100

#K-nn
control <- trainControl(method='repeatedcv',
                        number=10,
                        repeats=3)
knnFit <- train(diagnosis ~ ., data = train[,-1], method = "knn", trControl
= control,tuneLength = 20)
plot(knnFit)

#Testing the model
knnPredict <- predict(knnFit,newdata = test )
cm_knn<-confusionMatrix(knnPredict, test$diagnosis )
cm_knn

#Naive Bayes
#Training the model
learn_nb <- naiveBayes(train[,-c(1,2)], train$diagnosis)
#Testing the model
pre_nb <- predict(learn_nb, test[,-c(1,2)])
cm_nb <- confusionMatrix(pre_nb, test$diagnosis)
cm_nb

#Classification tree
#Training the model
```

```
learn_ct <- ctree(diagnosis~., data=train[,-1],
controls=ctree_control(maxdepth=2))
#Testing the model
pre_ct   <- predict(learn_ct, test[,-c(1,2)])
cm_ct    <- confusionMatrix(pre_ct, test$diagnosis)
cm_ct

# K-means
#Training the model.

predict.kmeans <- function(newdata, object){
  centers <- object$centers
  n_centers <- nrow(centers)
  dist_mat <- as.matrix(dist(rbind(centers, newdata)))
  dist_mat <- dist_mat[-seq(n_centers), seq(n_centers)]
  max.col(-dist_mat)
}
learn_kmeans <- kmeans(train[,-c(1,2)], centers=2)
#Testing the model.

pre_kmeans <- predict.kmeans(test[,-c(1,2)],learn_kmeans)
pre_kmeans <- factor(ifelse(pre_kmeans == 1,"B","M"))
cm_kmeans <- confusionMatrix(pre_kmeans, test$diagnosis)
cm_kmeans

#Hierarchical
# Dissimilarity matrix
d <- dist(df, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc1 <- hclust(d, method = "complete" )

# Plot the obtained dendrogram
plot(hc1, cex = 0.6, hang = -1)

# Cut tree into 3 groups
sub_grps <- cutree(hc1, k = 3)

# Visualize the result in a scatter plot
fviz_cluster(list(data = df, cluster = sub_grps))

# Plot the obtained dendrogram with
# rectangle borders for k clusters
plot(hc1, cex = 0.6, hang = -1)
rect.hclust(hc1, k = 3, border = 2:4)


col <- c("#ed3b3b", "#0099ff")
par(mfrow=c(2,3))
fourfoldplot(cm_ct$table, color = col, conf.level = 0, margin = 1,
main=paste("CTree (",round(cm_ct$overall[1]*100),"%)",sep=""))
fourfoldplot(cm_nb$table, color = col, conf.level = 0, margin = 1,
main=paste("NaiveBayes (",round(cm_nb$overall[1]*100),"%)",sep=""))
fourfoldplot(cm_knn$table, color = col, conf.level = 0, margin = 1,
main=paste("K-nn (",round(cm_ranger$overall[1]*100),"%)",sep=""))


cm_knn$table
```