EDX-choose your own

Hussin Almustafa

2023-10-01

Introduction:

Breast cancer is the most frequently occurring cancer in women. If the cancer is diagnosed and treated at an early stage, the patient has a survival rate of 99% after 5 years, but it significantly drops to 29% when it reaches a distant stage. Thus, it is very important to detect 'positive cancer cells' in the early stage, so I analyzed the 569 breast cancer cell data provided by the University of Wisconsin using R , the goal of this Project is to created a predictive model that can detect positive cancer cells based on different Algorithms.

the data set is Available on

this is a link to my github repo

load data :

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

library(tidyverse)
library(caret)
library(rpart)
```

Diagnostic Wisconsin Breast Cancer Database :

```
dl <- "Breast Cancer Wisconsin (Diagnostic) Data Set"
if(!file.exists(dl))
  download.file("https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data", dl)

#   https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data?resource=download


data□breast <- read.csv("/Users/hussinalmustafa/Downloads/data.csv")
head(data□breast)
```

```
##         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
```

remove columns (1,33)

```
data_breast <- data_breast[,-c(1,33)]


# Changing the postiion of dependent variable .
```

```r
data_breast <- data_breast%>% relocate(diagnosis )

str (data_breast)
```

```
## 'data.frame':    569 obs. of  31 variables:
##  $ diagnosis              : chr  "M" "M" "M" "M" ...
##  $ 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 ...
```

clean the data :

```r
# check   missing data :

sapply(data_breast, function(.x) sum(is.na(.x)))
```

```
##              diagnosis            radius_mean           texture_mean
##                      0                      0                      0
##         perimeter_mean              area_mean        smoothness_mean
##                      0                      0                      0
##       compactness_mean         concavity_mean    concave.points_mean
##                      0                      0                      0
##          symmetry_mean fractal_dimension_mean              radius_se
```

```
##                         0                         0                         0
##              texture_se            perimeter_se                 area_se
##                         0                         0                         0
##           smoothness_se          compactness_se            concavity_se
##                         0                         0                         0
##       concave.points_se             symmetry_se    fractal_dimension_se
##                         0                         0                         0
##            radius_worst           texture_worst         perimeter_worst
##                         0                         0                         0
##              area_worst        smoothness_worst       compactness_worst
##                         0                         0                         0
##         concavity_worst    concave.points_worst          symmetry_worst
##                         0                         0                         0
## fractal_dimension_worst
##                         0
```

there is no missing data .

```
data_breast %>% select_if(is.character) %>%
                  map_if(is.character, unique)
```

check the level of character in data set :

```
## $diagnosis
## [1] "M" "B"
```

```
table(data_breast$diagnosis)
```

check how balanced is our response variable :

```
##
##   B   M
## 357 212
```

```
data_breast<- data_breast %>% mutate_if(is_character, as.factor)
```

change the variables into factors :

remove highly correlated predictors :

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
# correlation for all numeric  variables


data_corr <-   cor(data_breast[,2:31])


#  remove the highly correlated ones using the caret package.

data_b <-  data_breast %>% select(-findCorrelation(data_corr, cutoff = 0.9))

str(data_b)
```

```
## 'data.frame':    569 obs. of  21 variables:
##  $ 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ 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 ...
```

our new data is 21 variables :

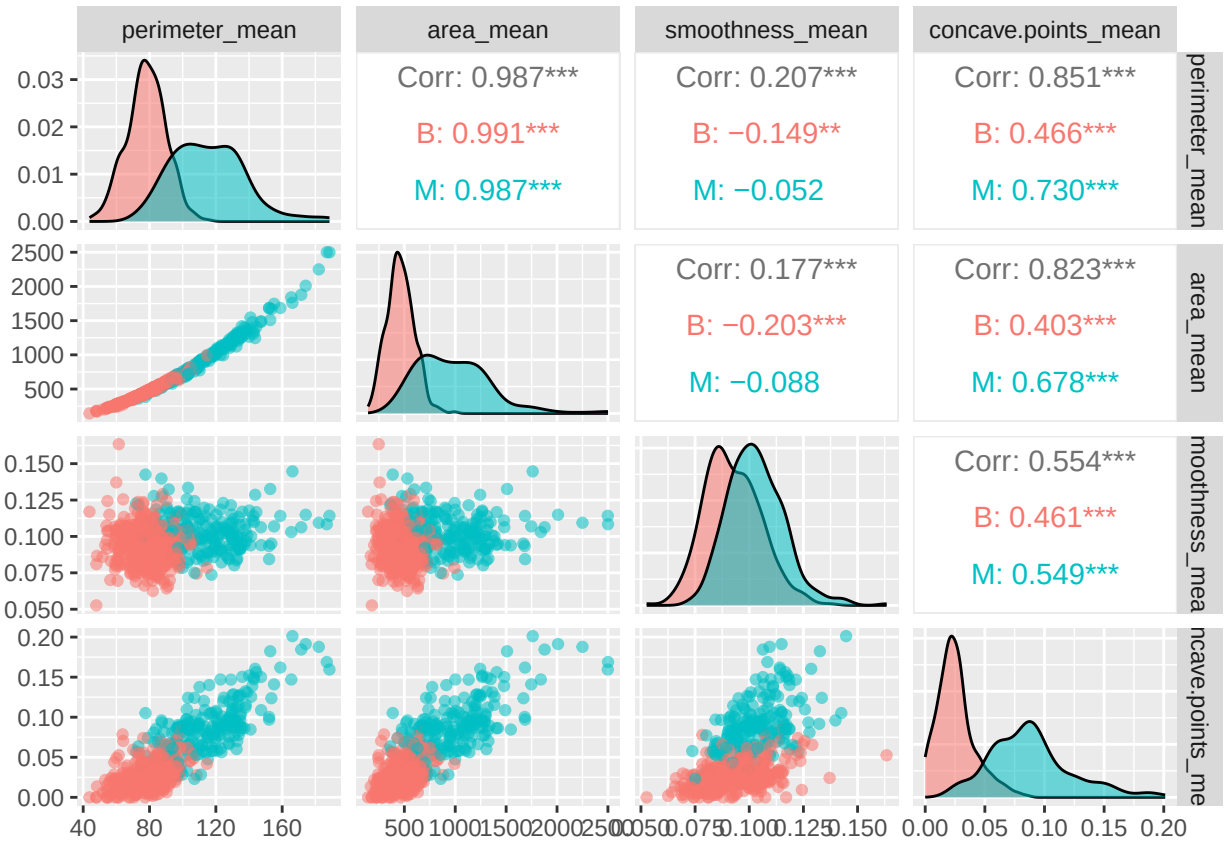
data visualization :

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# We have many variables in this dataset.  we will focus only on the first four .

ggpairs( data_b[1:4], aes(color=data_breast$diagnosis , alpha=0.4))
```




```
dat <- cbind(diagnosis = data_breast$diagnosis, data_b)
str(dat)
```

```
## 'data.frame':    569 obs. of  22 variables:
##  $ diagnosis               : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ radius_worst            : num  25.4 25 23.6 14.9 22.5 ...
##  $ 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 ...
```

Split dat into test and training sets:

```
set.seed(42, sample.kind = "Rounding")


test_index <- createDataPartition(dat$diagnosis, times = 1, p = 0.2, list = FALSE) # create a 20% test
test_set <- dat[test_index,]
train_set <- dat[-test_index,]

nrow(train_set)
```

```
## [1] 454
```

1- Logistic regression models :

```
set.seed(1, sample.kind = "Rounding")
train_glm <- train(diagnosis ~ ., method = "glm", data = train_set)

# The accuracy can be calculated using the following code:
glm_preds <- predict(train_glm, test_set)
mean(glm_preds == test_set$diagnosis)
```

```
## [1] 0.9826087
```

2- kNN model :

```
set.seed(6, sample.kind = "Rounding")

train_knn <- train(diagnosis ~ .,
                   method = "knn",
                   data = train_set,
                   tuneGrid = data.frame(k = seq(3, 51, 2)),
                   trControl = trainControl(method = "cv", number = 10, p = 0.9))
train_knn$bestTune
```

```
##   k
## 3 7
```

```
# The accuracy can be calculated using the following code:
knn_preds <- predict(train_knn, test_set)
mean(knn_preds == test_set$diagnosis)
```

```
## [1] 0.9391304
```

3- Classification tree model :

```
set.seed(10, sample.kind = "Rounding")
train_rpart <- train(diagnosis ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.05, 0.002)),
                     data = train_set)
train_rpart$bestTune
```

```
##       cp
## 13 0.024
```

```
# the accuracy of the decision tree model on the test set :

rpart_preds <- predict(train_rpart, test_set)
mean(rpart_preds == test_set$diagnosis)
```

```
## [1] 0.9391304
```

4- Random forest model :

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(100, sample.kind = "Rounding")
train_rf <- train(diagnosis ~ .,
                  data = train_set,
                  method = "rf",
                  ntree = 100,
                  tuneGrid = data.frame(mtry = seq(1:7)))
train_rf$bestTune
```

```
##   mtry
## 1    1
```

```r
# The accuracy can be calculated using the following code:
  rf_preds <- predict(train_rf, test_set)
mean(rf_preds == test_set$diagnosis)
```

```
## [1] 0.9565217
```

*determine the importance of various predictors to the random
 forest model*

The most important variable can be found using the following code:

```r
varImp(train_rf)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 21)
##
##                          Overall
## concave.points_worst    100.000
## radius_worst             67.350
## area_worst               60.903
## area_mean                57.374
## concave.points_mean      53.647
## radius_se                53.310
## perimeter_mean           42.376
## area_se                  38.116
## compactness_worst        30.499
## smoothness_worst         23.433
## concavity_se             17.375
## concavity_worst          16.904
## concave.points_se        15.769
## symmetry_worst           13.385
## compactness_se           12.498
## smoothness_mean           8.854
## fractal_dimension_worst   5.245
## fractal_dimension_mean    3.685
## symmetry_mean             1.959
## smoothness_se             1.270
```

we see that the model with best accuracy is Logistic regression model.

Citations:

Rafael A. Irizarry - Introduction to Data Science

Max Kuhn -The caret Package

François de Ryckel - Machine Learning with R