Untitled

Neelima

2023-04-23

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```
## Warning: package 'purrr' was built under R version 4.3.2
```

```
## Warning: package 'stringr' was built under R version 4.3.2
```

```
## Warning: package 'forcats' was built under R version 4.3.2
```

```
## Warning: package 'lubridate' was built under R version 4.3.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.3.2
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## Loaded glmnet 4.1-8
```

```r
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.3.2
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.2
```

```
## corrplot 0.92 loaded
```

```r
library(ggplot2)
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.3.2
```

```
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
##
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.2
```

```
## 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
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.3.2
```

```
## Loaded gbm 2.1.8.1
```

```
library(nnet)
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.3.2
```

```
library(kknn)
```

```
## Warning: package 'kknn' was built under R version 4.3.2
```

```
##
## Attaching package: 'kknn'
##
## The following object is masked from 'package:caret':
##
##     contr.dummy
```

```
library(cluster)
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.3.2
```

```
##
## Attaching package: 'mice'
##
## The following object is masked from 'package:stats':
```

```
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(readr)
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.3.2
```

```
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:Hmisc':
##
##     impute
```

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 4.3.2
```

```
library(caretEnsemble)
```

```
## Warning: package 'caretEnsemble' was built under R version 4.3.2
```

```
##
## Attaching package: 'caretEnsemble'
##
## The following object is masked from 'package:ggplot2':
##
##     autoplot
```

```
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 4.3.2
```

```
library(plotly)
```

```
## Warning: package 'plotly' was built under R version 4.3.2
```

```
##
## Attaching package: 'plotly'
##
## The following object is masked from 'package:Hmisc':
##
##     subplot
##
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(table1)
```

```
## Warning: package 'table1' was built under R version 4.3.2
```

```
##
## Attaching package: 'table1'
##
## The following objects are masked from 'package:Hmisc':
##
##     label, label<-, units
##
## The following objects are masked from 'package:base':
##
##     units, units<-
```

```
library(mboost)
```

```
## Warning: package 'mboost' was built under R version 4.3.2
```

```
## Loading required package: parallel
## Loading required package: stabs
```

```
## Warning: package 'stabs' was built under R version 4.3.2
```

```
##
## Attaching package: 'mboost'
##
## The following object is masked from 'package:glmnet':
##
##     Cindex
##
## The following object is masked from 'package:tidyr':
##
##     extract
##
## The following object is masked from 'package:ggplot2':
##
##     %+%
```

```
library(MLmetrics)
```

```
## Warning: package 'MLmetrics' was built under R version 4.3.2
```

```
##
## Attaching package: 'MLmetrics'
##
## The following object is masked from 'package:mboost':
##
##     AUC
##
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
##
## The following object is masked from 'package:base':
##
##     Recall
```

```
library(parallel)
library(iterators)
```

```
## Warning: package 'iterators' was built under R version 4.3.2
```

```
library(DT)
```

```
## Warning: package 'DT' was built under R version 4.3.2
```

```
library(foreach)
```

```
## Warning: package 'foreach' was built under R version 4.3.2
```

```
##
## Attaching package: 'foreach'
##
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
library(gganimate)
```

```
## Warning: package 'gganimate' was built under R version 4.3.2
```

```
library(gifski)
```

```
## Warning: package 'gifski' was built under R version 4.3.2
```

```
library(formatR)
```

```
## Warning: package 'formatR' was built under R version 4.3.2
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.3.2
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:randomForest':
##
##     combine
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(grid)
library(vcd)
```

```
## Warning: package 'vcd' was built under R version 4.3.2
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.3.2
```

```
library(corrplot)
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 4.3.2
```

```
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.3.2
```

```
library(PRROC)
```

```
## Warning: package 'PRROC' was built under R version 4.3.2
```

Introduction

Breast cancer is one of the most common cancers among women worldwide, affecting millions of women each year. This project aims to analyze a dataset containing information about breast cancer tumors to build predictive models that can classify tumors as benign or malignant.

#Data Loading

```
url <- "https://drive.google.com/uc?id=1fgt_sIS2V6COS_E-I6n-wx7UTG5taCsz"
breast_cancer <- read.csv(url)
```

#Data Exploration

```
# Summary and class type for each column
summary(breast_cancer)
```

```
##        id              diagnosis          radius_mean      texture_mean
##  Min.   :     8670   Length:569         Min.   : 6.981   Min.   : 9.71
##  1st Qu.:   869218   Class :character   1st Qu.:11.700   1st Qu.:16.17
##  Median :   906024   Mode  :character   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
```

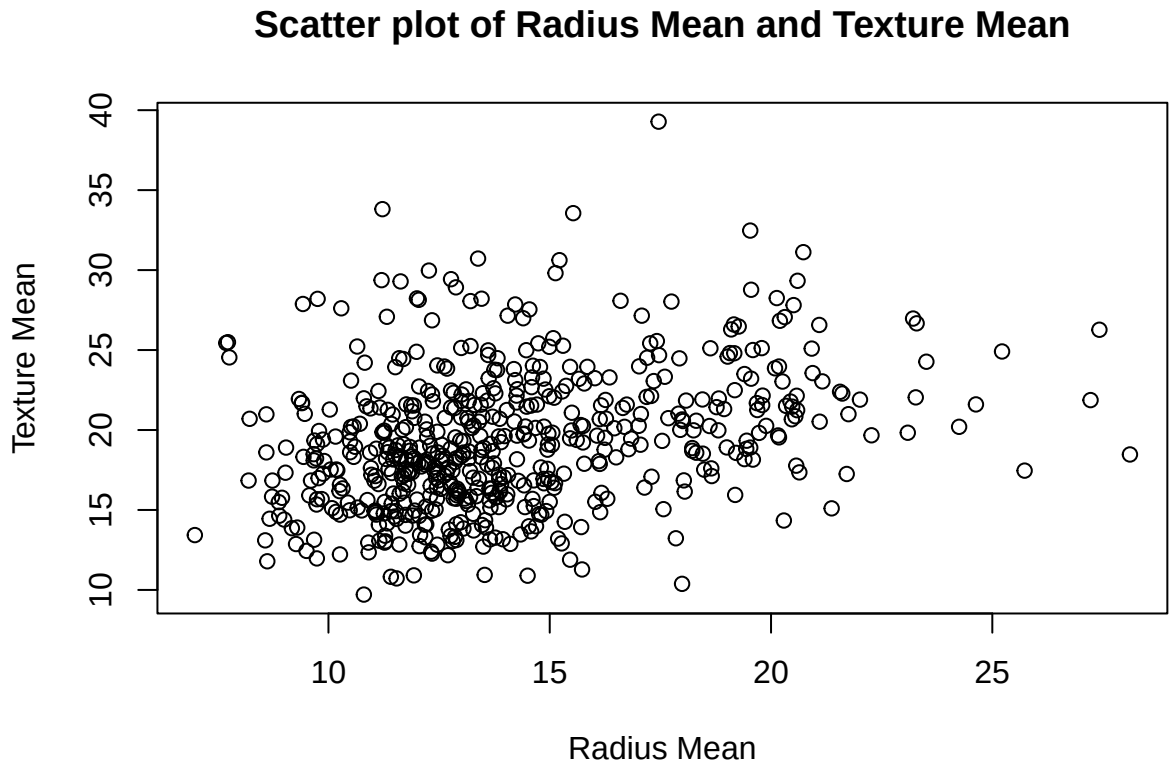
```
##  3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190
##  Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200
##  smoothness_se      compactness_se      concavity_se     concave.points_se
##  Min.   :0.001713   Min.   :0.002252   Min.   :0.00000   Min.   :0.000000
##  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
```

```
sapply(breast_cancer, class)
```

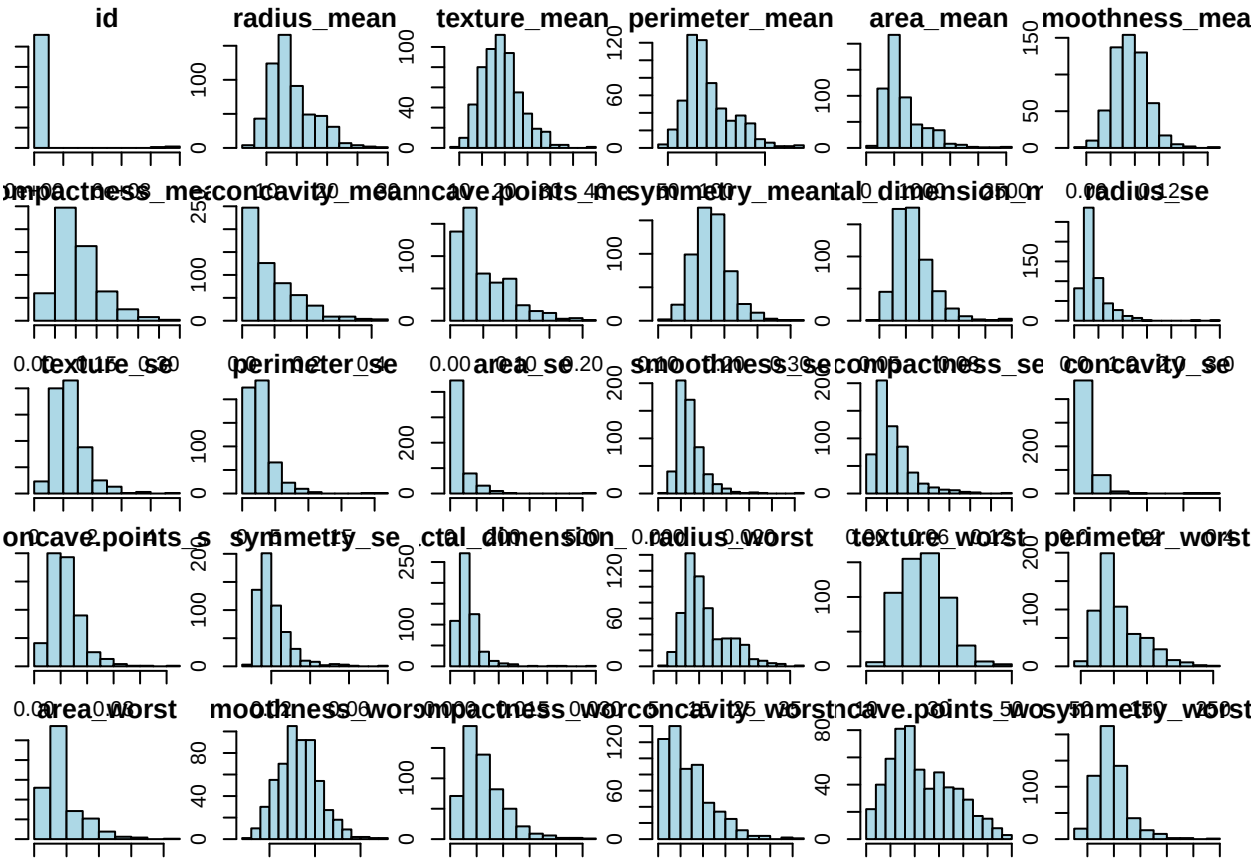
```
##                     id              diagnosis            radius_mean
##              "integer"            "character"              "numeric"
##           texture_mean         perimeter_mean              area_mean
##              "numeric"              "numeric"              "numeric"
##      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

```
##              "numeric"              "numeric"
```

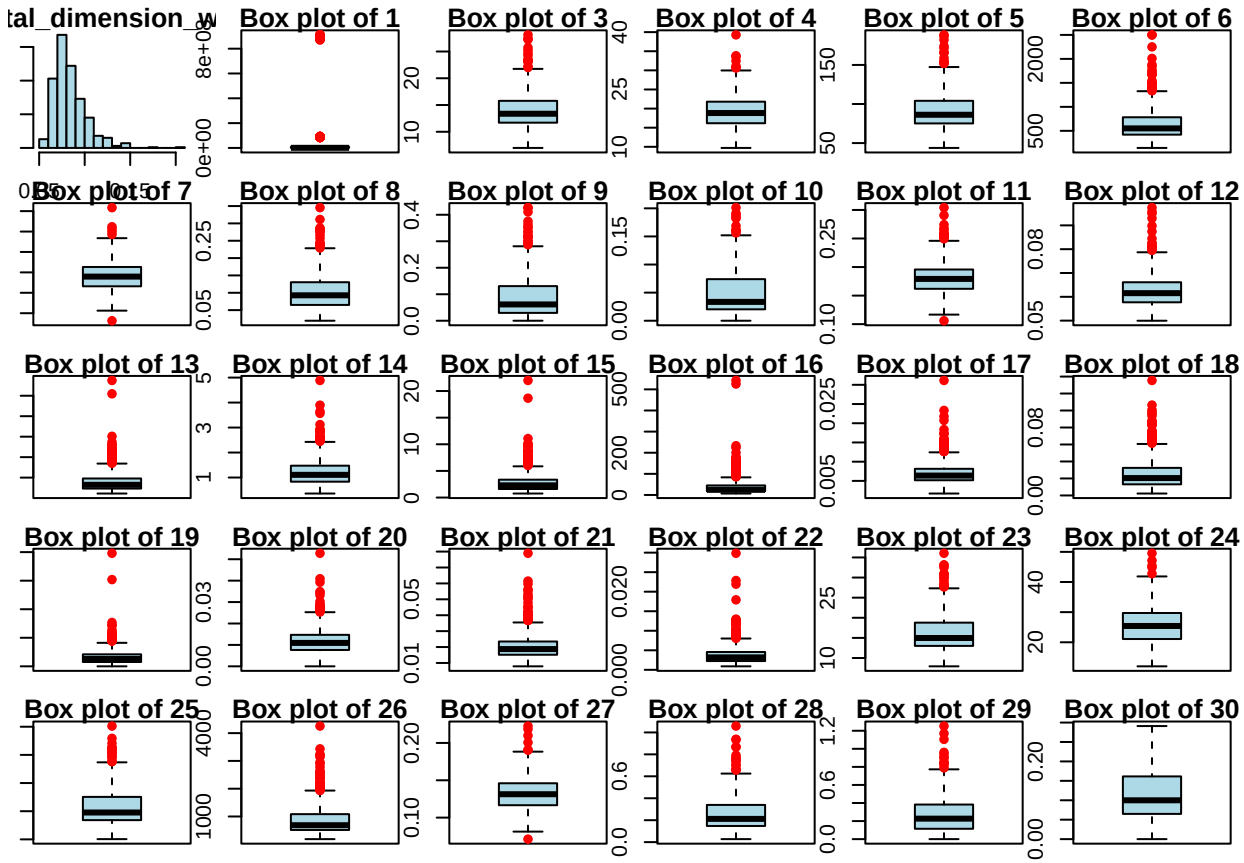
```r
# Scatter plot for radius mean and texture mean
plot(breast_cancer$radius_mean, breast_cancer$texture_mean,
     main = "Scatter plot of Radius Mean and Texture Mean",
     xlab = "Radius Mean", ylab = "Texture Mean")
```




```r
# Histograms for each continuous variable
par(mfrow=c(5,6), mar=c(1,1,1,1))
for (i in which(sapply(breast_cancer, is.numeric))) {
  hist(breast_cancer[,i], main=colnames(breast_cancer)[i], col="lightblue")
}
```

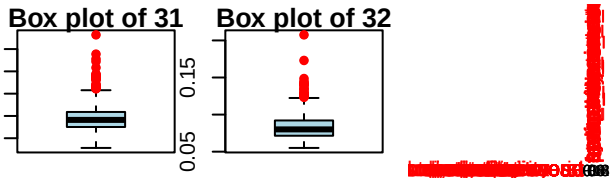



```r
# Box plot for each numeric variable
for (col in which(sapply(breast_cancer, is.numeric))) {
  boxplot(breast_cancer[[col]], main = paste("Box plot of", col), ylab = col, col = "lightblue")
  outliers <- boxplot.stats(breast_cancer[[col]])$out
  if (length(outliers) > 0) {
    points(rep(1, length(outliers)), outliers, col = "red", pch = 16)
  }
}
```




```
# Correlation matrix
correlation_matrix <- cor(breast_cancer[, sapply(breast_cancer, is.numeric)], use="pairwise.complete.obs
corrplot(correlation_matrix, method="circle")
```

```
## Warning in corrplot(correlation_matrix, method = "circle"): Not been able to
## calculate text margin, please try again with a clean new empty window using
## {plot.new(); dev.off()} or reduce tl.cex
```



Data Preparation

```
data <- breast_cancer %>%
  select(-id) %>%
  mutate(diagnosis = factor(ifelse(diagnosis == "B", "Benign", "Malignant")))

sum(is.na(data)) # Checking for missing values
```

```
## [1] 0
```

```
# Normalize the data excluding the target variable
data_normalized <- as.data.frame(scale(data %>% select(-diagnosis))) # Scale only numeric predictors
data_normalized$diagnosis <- data$diagnosis # Add the diagnosis factor column back in after scaling
```

Data Modeling

```
# Data Modeling - Train models

# Set the seed for reproducibility
set.seed(123)

# Create a partition to split the data into training and testing sets
train_index <- createDataPartition(data_normalized$diagnosis, p = 0.75, list = FALSE)
```

```r
train_data <- data_normalized[train_index, ]
test_data <- data_normalized[-train_index, ]

# Train a Random Forest model
library(randomForest)
model_rf <- randomForest(diagnosis ~ ., data = train_data, ntree=500, mtry=2, importance=TRUE)

# Generalized Linear Model via glmnet
# Prepare matrix for glmnet with the response variable 'diagnosis'
x_train <- model.matrix(diagnosis ~ . - 1, data = train_data) # Removing the intercept term
y_train <- ifelse(train_data$diagnosis == "Malignant", 1, 0)  # Convert to binary outcomes
x_test <- model.matrix(diagnosis ~ . - 1, data = test_data) # Test data for prediction phase

# Fit the model using glmnet with cross-validation to select lambda
library(glmnet)
cv_fit <- cv.glmnet(x_train, y_train, family = "binomial", alpha = 1) # Lasso penalty

# Fit the final model using the lambda that minimized cross-validation error
final_model_glmnet <- glmnet(x_train, y_train, family = "binomial", alpha = 1, lambda = cv_fit$lambda.m

# Gradient Boosting Machine model with caret for parameter tuning and cross-validation
library(gbm)
set.seed(123) # Resetting seed for reproducibility with GBM
train_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search = "grid")
model_gbm <- train(diagnosis ~ ., data = train_data, method = "gbm", trControl = train_control,
                   verbose = FALSE, tuneLength = 5)
```

#Model Evaluation

```r
# Predict and evaluate the Random Forest model
predictions_rf <- predict(model_rf, newdata = test_data)
confusion_rf <- confusionMatrix(predictions_rf, test_data$diagnosis)

# Predict and evaluate the glmnet model (need to make predictions on the test set)
predictions_glmnet_prob <- predict(final_model_glmnet, newx = x_test, type = "response")
predictions_glmnet <- ifelse(predictions_glmnet_prob > 0.5, "Malignant", "Benign")
predictions_glmnet_factor <- factor(predictions_glmnet, levels = levels(train_data$diagnosis))
confusion_glmnet <- confusionMatrix(predictions_glmnet_factor, test_data$diagnosis)

# Predict and evaluate the GBM model
predictions_gbm <- predict(model_gbm, newdata = test_data, type = "raw")
confusion_gbm <- confusionMatrix(predictions_gbm, test_data$diagnosis)

# Performance summaries
rf_summary <- summary(confusion_rf)
glmnet_summary <- summary(confusion_glmnet)
gbm_summary <- summary(confusion_gbm)

# Calculate ROC for Random Forest
rf_roc <- roc(response = test_data$diagnosis, predictor = as.numeric(predictions_rf == "Malignant"))
```

```
## Setting levels: control = Benign, case = Malignant
```

```
## Setting direction: controls < cases
```

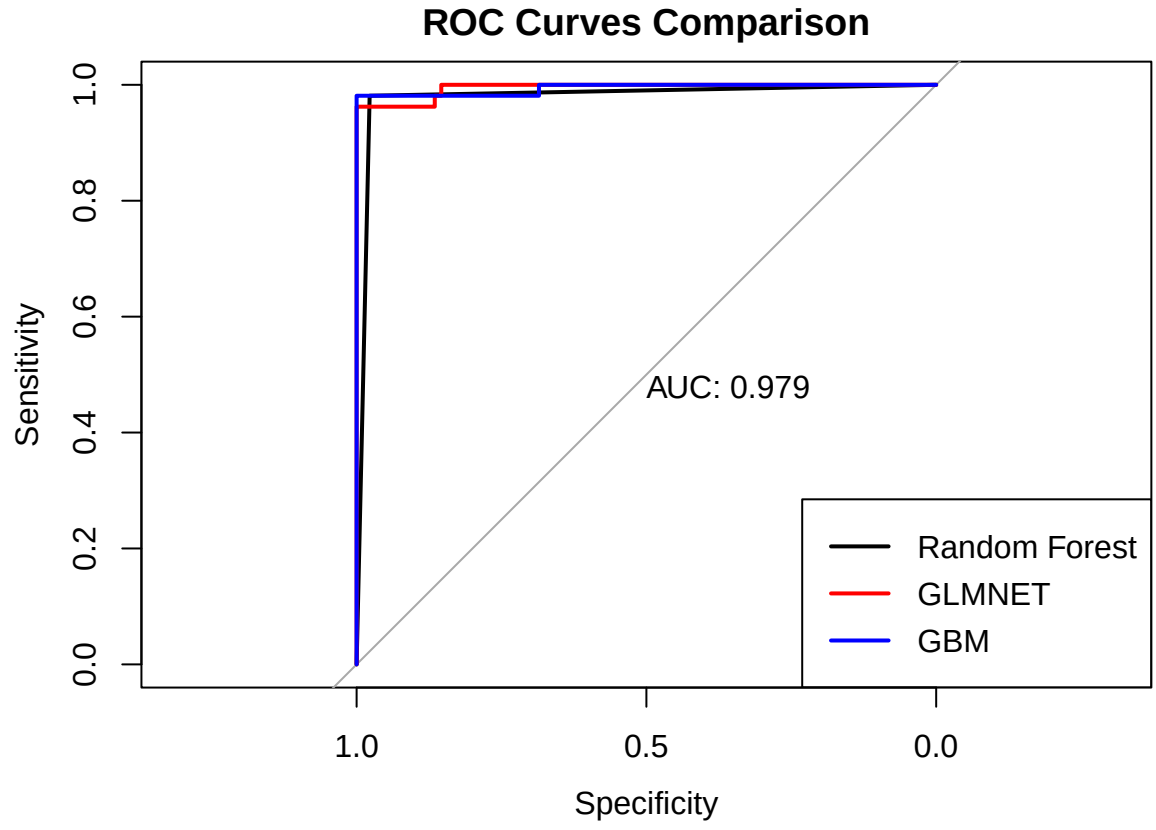
```
# Calculate ROC for GLMNET
glmnet_roc <- roc(response = test_data$diagnosis, predictor = predictions_glmnet_prob)
```

```
## Setting levels: control = Benign, case = Malignant

## Warning in roc.default(response = test_data$diagnosis, predictor =
## predictions_glmnet_prob): Deprecated use a matrix as predictor. Unexpected
## results may be produced, please pass a numeric vector.

## Setting direction: controls < cases
```

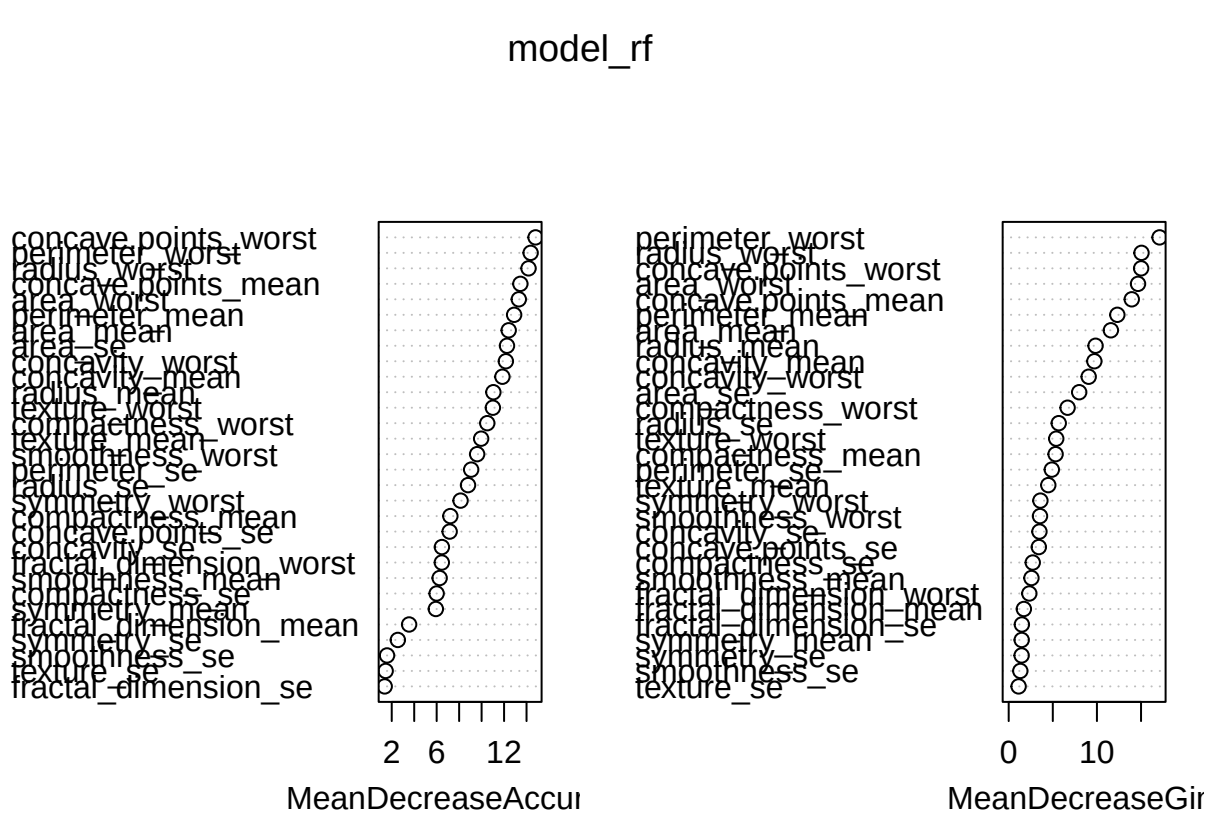
```
# Calculate ROC for GBM
gbm_probs <- predict(model_gbm, newdata = test_data, type = "prob")
gbm_roc <- roc(response = test_data$diagnosis, predictor = gbm_probs[, "Malignant"])
```

```
## Setting levels: control = Benign, case = Malignant
## Setting direction: controls < cases
```

```
# Plot ROC Curves
plot(rf_roc, print.auc = TRUE, main="ROC Curves Comparison")
lines(glmnet_roc, col = "red", print.auc = TRUE)
lines(gbm_roc, col = "blue", print.auc = TRUE)
legend("bottomright", legend = c("Random Forest", "GLMNET", "GBM"),
       col = c("black", "red", "blue"), lwd = 2)
```



```r
# Variable Importance for Random Forest (if deemed necessary)
varImpPlot(model_rf)
```




```r
# Extract accuracy value from confusion matrix
rf_accuracy <- confusion_rf$overall["Accuracy"]
glmnet_accuracy <- confusion_glmnet$overall["Accuracy"]
gbm_accuracy <- confusion_gbm$overall["Accuracy"]

# Create a data frame to compare accuracies
accuracy_df <- data.frame(
  Model = c("Random Forest", "GLMNET", "GBM"),
  Accuracy = c(rf_accuracy, glmnet_accuracy, gbm_accuracy)
)

# Print the accuracy values
print(accuracy_df)
```

```
##           Model  Accuracy
## 1 Random Forest 0.9788732
## 2        GLMNET 0.9788732
## 3           GBM 0.9859155
```

```r
# Find the model with the highest accuracy
best_model <- accuracy_df[which.max(accuracy_df$Accuracy), ]
```

```r
# Print the best model
cat("The model with the highest accuracy is:", best_model$Model)
```

```
## The model with the highest accuracy is: GBM
```

```r
cat("Accuracy:", best_model$Accuracy)
```

```
## Accuracy: 0.9859155
```

Conclusion

This report provided an analysis of a breast cancer dataset with the aim of predicting cancer malignancy. The GBM model demonstrated good performance in classification tasks, and the evaluation metrics support its reliability as a predictive model.