Final_raisin

Thu Tran

2024-04-17

1. Load data

```
library(readxl)
library(car)
```

```
## Loading required package: carData
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::recode() masks car::recode()
## x purrr::some()   masks car::some()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
raisin<-read_excel("Raisin_Dataset.xlsx")
raisin$Class<-factor(raisin$Class)
str(raisin)
```

```
## tibble [900 x 8] (S3: tbl_df/tbl/data.frame)
##  $ Area           : num [1:900] 87524 75166 90856 45928 79408 ...
##  $ MajorAxisLength: num [1:900] 442 407 442 287 352 ...
##  $ MinorAxisLength: num [1:900] 253 243 266 209 291 ...
##  $ Eccentricity   : num [1:900] 0.82 0.802 0.798 0.685 0.564 ...
##  $ ConvexArea     : num [1:900] 90546 78789 93717 47336 81463 ...
##  $ Extent         : num [1:900] 0.759 0.684 0.638 0.7 0.793 ...
##  $ Perimeter      : num [1:900] 1184 1122 1209 844 1073 ...
##  $ Class          : Factor w/ 2 levels "Besni","Kecimen": 2 2 2 2 2 2 2 2 2 2 ...
```

2. EDA (Exploratory Data Analysis)

```r
options(scipen = 999)
#Summary table
library(skimr)
library(gt)
skim_tb<-skim(raisin)


raisin_summary<-data.frame(
  Type = skim_tb$skim_type,
  Variables = skim_tb$skim_variable,
  Missing = skim_tb$n_missing,
  Min = skim_tb$numeric.p0,
  Mean = skim_tb$numeric.mean,
  Median = skim_tb$numeric.p50,
  Max = skim_tb$numeric.p100,
  SD = skim_tb$numeric.sd
)
raisin_summary[2:8,4:8]<-round(raisin_summary[2:8,4:8],2)
gt(raisin_summary)%>%
  tab_header(
    title = "STATISTICAL SUMMARY TABLE"
  )
```

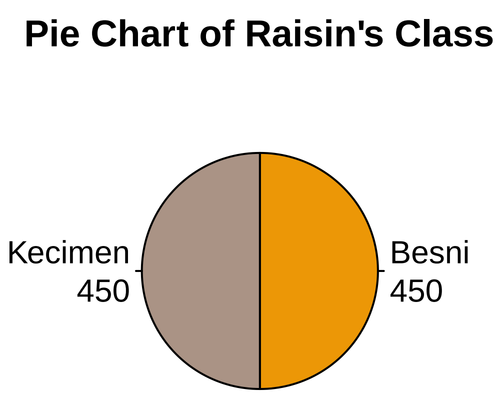
STATISTICAL SUMMARY TABLE

Type	Variables	Missing	Min	Mean	Median	Max	SD
factor	Class	0	NA	NA	NA	NA	NA
numeric	Area	0	25387.00	87804.13	78902.00	235047.00	39002.11
numeric	MajorAxisLength	0	225.63	430.93	407.80	997.29	116.04
numeric	MinorAxisLength	0	143.71	254.49	247.85	492.28	49.99
numeric	Eccentricity	0	0.35	0.78	0.80	0.96	0.09
numeric	ConvexArea	0	26139.00	91186.09	81651.00	278217.00	40769.29
numeric	Extent	0	0.38	0.70	0.71	0.84	0.05
numeric	Perimeter	0	619.07	1165.91	1119.51	2697.75	273.76

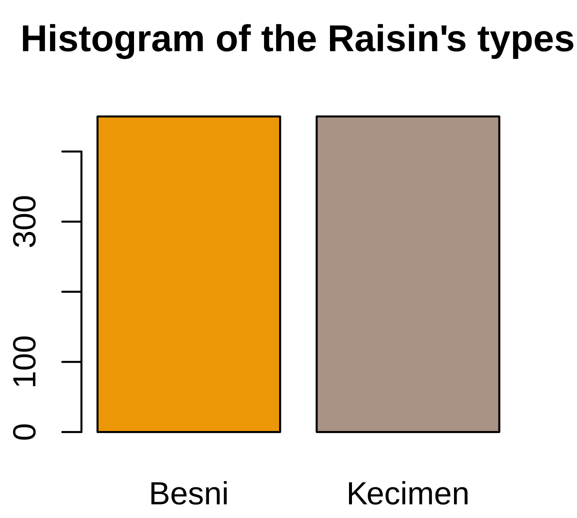
```r
raisin[c(86,291),]
```

```
## # A tibble: 2 x 8
##     Area MajorAxisLength MinorAxisLength Eccentricity ConvexArea Extent
##    <dbl>           <dbl>           <dbl>        <dbl>      <dbl>  <dbl>
## 1 180898            844.            323.        0.924     221396  0.454
## 2 136340            723.            311.        0.902     176818  0.530
## # i 2 more variables: Perimeter <dbl>, Class <fct>
```

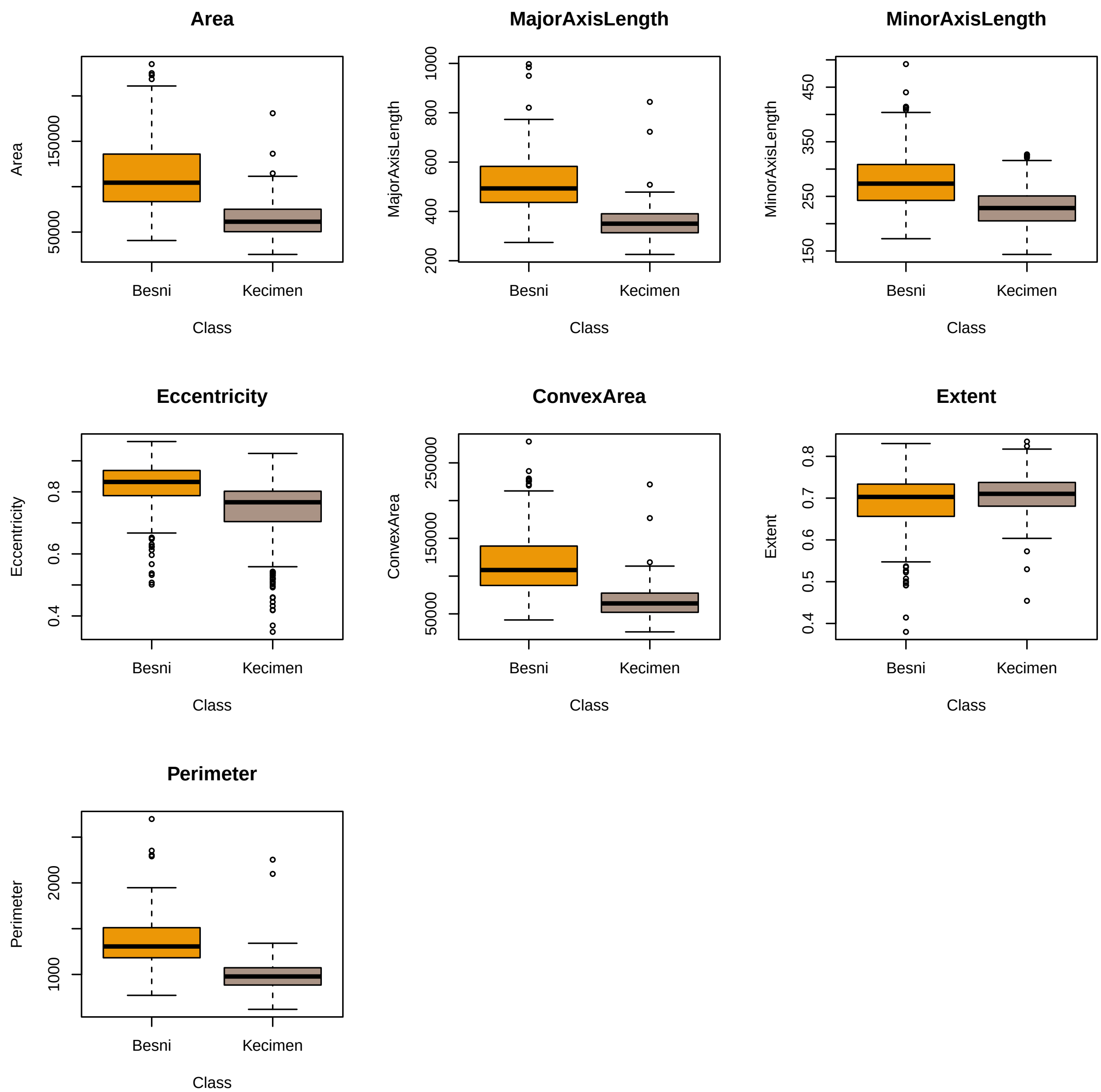
```r
# Pie Chart from data frame with Appended Sample Sizes
mytable <- table(raisin$Class)
lbls <- paste(names(mytable), "\n", mytable, sep="")
pie(mytable, labels = lbls, clockwise = T,col=c("#EC9706","#AA9385"),
   main="Pie Chart of Raisin's Class")
```




```r
# Histogram of raisin
plot(raisin$Class,col=c("#EC9706","#AA9385"), main= "Histogram of the Raisin's types")
```



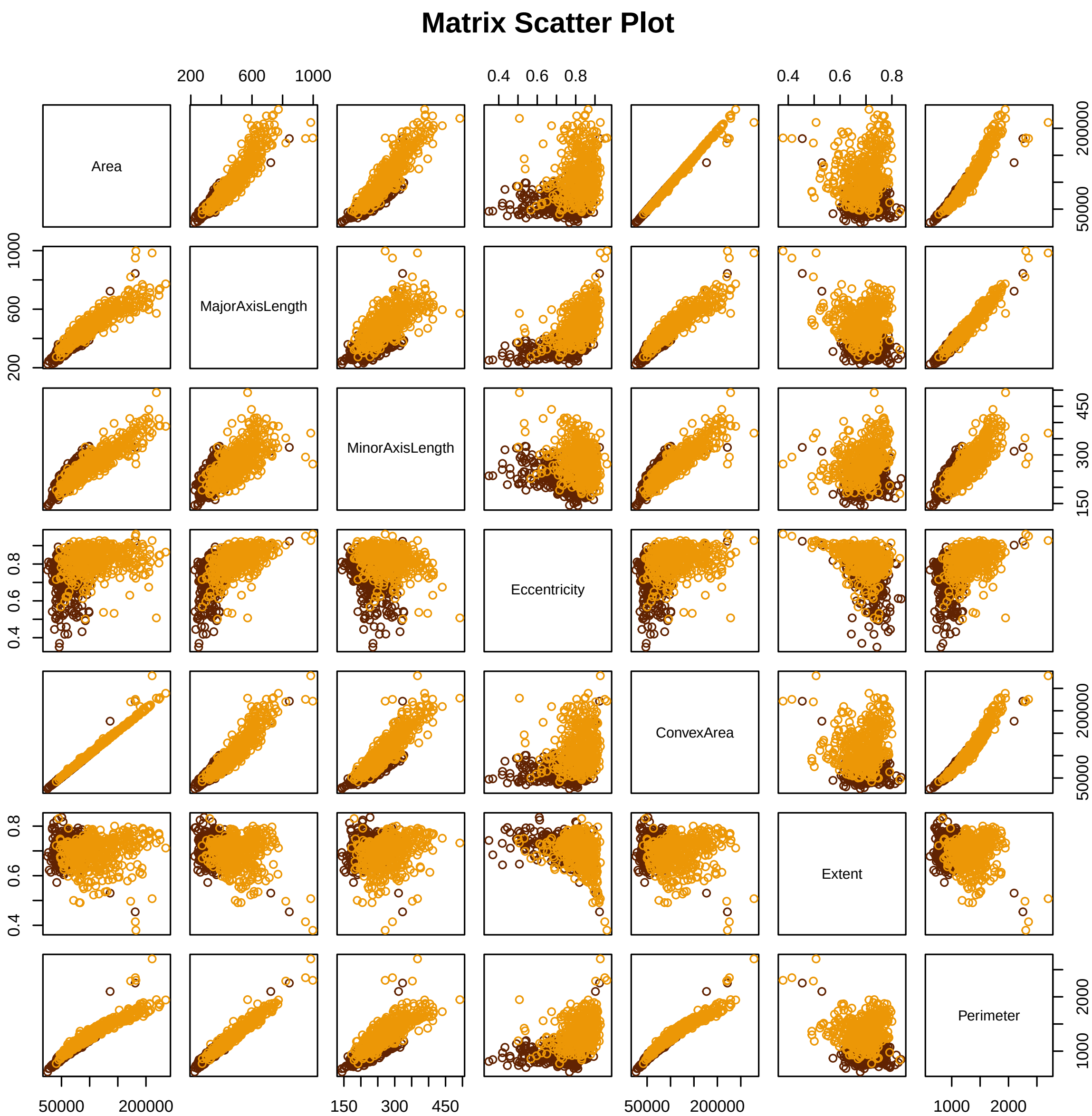
```
# Multiple side-by side boxplot
par(mfrow=c(3,3))  # Set up the layout

for (col_name in names(raisin)[1:7]) {
  boxplot(raisin[[col_name]] ~ raisin$Class,
          data = raisin,
          main = col_name,
          xlab = "Class",
          ylab = col_name,
          col = c("#EC9706","#AA9385"))
}
```



According to the multiple boxplot, the Besni raisin seems to have higher median in each measurement than the Kecimen raisin.

c. Matrix Scatter plot

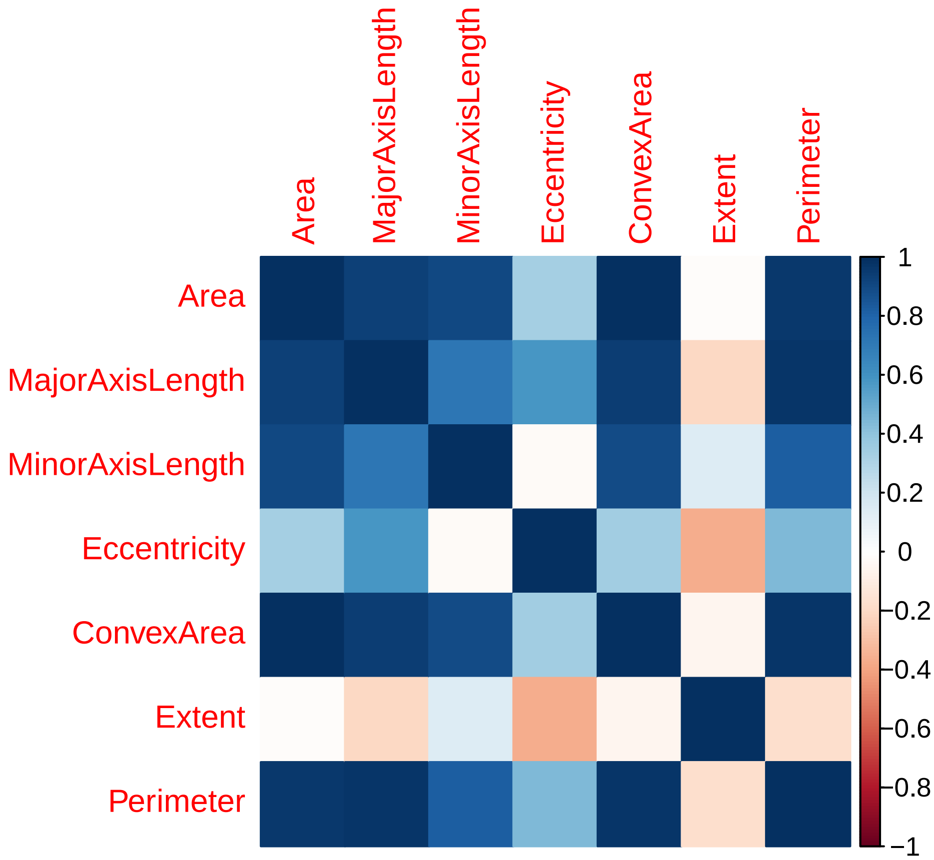
```r
class_col<-ifelse(raisin$Class=="Besni","#EC9706","#612302")
pairs(raisin[1:7],
      pch = 21,
      col = class_col,
      main = "Matrix Scatter Plot")
```



```r
# Correlation Heatmap
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```r
corr_matrix<-cor(raisin[,1:7])
corrplot(corr_matrix, method="color")
```



3. Statistic Analysis

a. Full model

```
fullmodel<-glm(Class ~ . ,data = raisin, family = binomial)
summary(fullmodel)
```

```
##
## Call:
## glm(formula = Class ~ ., family = binomial, data = raisin)
##
## Coefficients:
##                    Estimate Std. Error z value     Pr(>|z|)
## (Intercept)       2.2319880  7.0449015   0.317     0.751378
## Area             -0.0005010  0.0001244  -4.027 0.0000566058 ***
## MajorAxisLength   0.0445871  0.0159710   2.792     0.005242 **
## MinorAxisLength   0.0910874  0.0269403   3.381     0.000722 ***
## Eccentricity      3.8908609  4.9124257   0.792     0.428335
## ConvexArea        0.0004089  0.0001191   3.434     0.000594 ***
## Extent            0.6828128  2.7159964   0.251     0.801502
## Perimeter        -0.0361306  0.0066136  -5.463 0.0000000468 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1247.66  on 899  degrees of freedom
## Residual deviance:  608.98  on 892  degrees of freedom
## AIC: 624.98
##
## Number of Fisher Scoring iterations: 7
```

```
nullmodel<-glm(Class~1,data=raisin,family=binomial)
summary(nullmodel)
```

```
##
## Call:
## glm(formula = Class ~ 1, family = binomial, data = raisin)
##
## Coefficients:
##                             Estimate             Std. Error z value Pr(>|z|)
## (Intercept) 0.0000000000000003901 0.0666666666666666574       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1247.7  on 899  degrees of freedom
## Residual deviance: 1247.7  on 899  degrees of freedom
## AIC: 1249.7
##
## Number of Fisher Scoring iterations: 2
```

b. Variable selection:

AIC backward:

```
model1<-step(fullmodel,trace=0)
summary(model1)
```

```
##
## Call:
## glm(formula = Class ~ Area + MajorAxisLength + MinorAxisLength +
##     ConvexArea + Perimeter, family = binomial, data = raisin)
##
## Coefficients:
##                   Estimate Std. Error z value     Pr(>|z|)
## (Intercept)      6.7317525  4.4215966   1.522     0.127891
## Area            -0.0004777  0.0001159  -4.120 0.0000378603 ***
## MajorAxisLength  0.0467310  0.0156343   2.989     0.002799 **
## MinorAxisLength  0.0788838  0.0212508   3.712     0.000206 ***
## ConvexArea       0.0003990  0.0001127   3.540     0.000401 ***
## Perimeter       -0.0360055  0.0063523  -5.668 0.0000000144 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1247.66  on 899  degrees of freedom
## Residual deviance:  609.64  on 894  degrees of freedom
## AIC: 621.64
##
## Number of Fisher Scoring iterations: 7
```

```
# Multicolinearity
vif(model1)
```

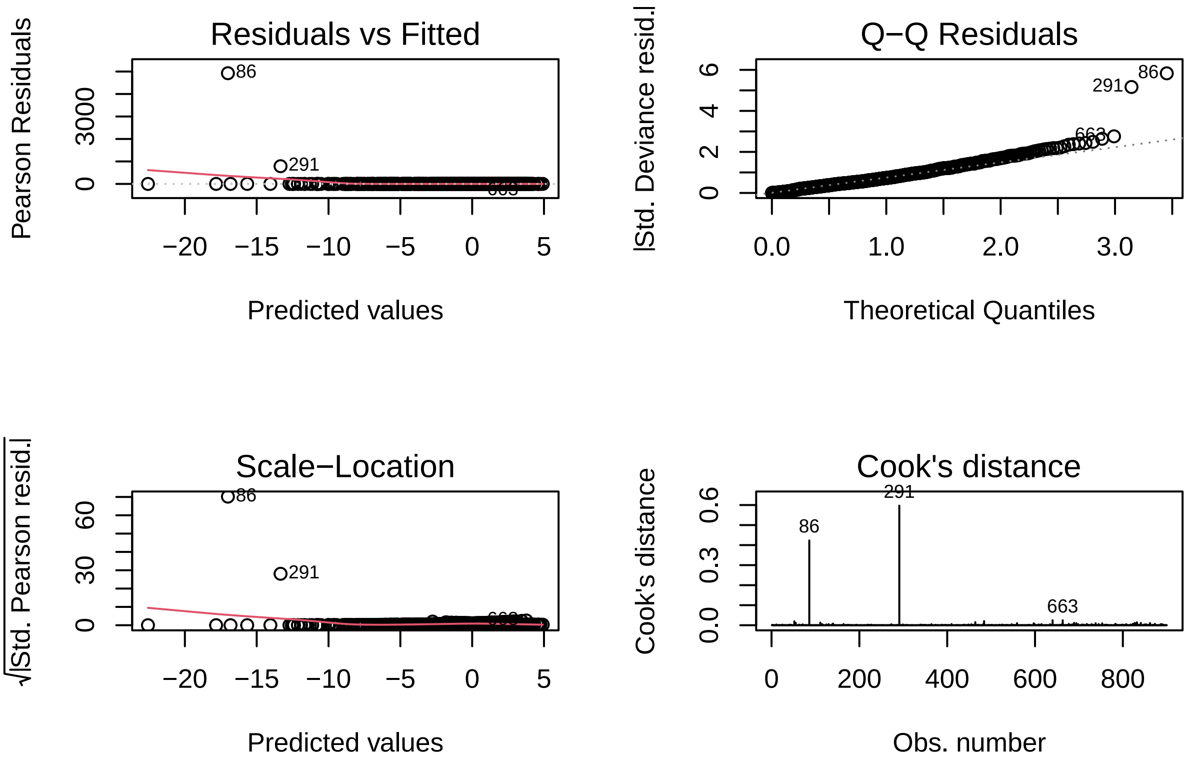
```
##            Area MajorAxisLength MinorAxisLength      ConvexArea       Perimeter
##       429.59691        68.26062        51.69712       431.27441        65.34105
```

Since the Area and CovexArea has a high multi colinearity. We decide to drop ConvexArea in the model1

```r
model2<-glm(formula = Class ~Area+ MajorAxisLength + MinorAxisLength + Perimeter,
            family = binomial, data = raisin)
summary(model2)
```

```
##
## Call:
## glm(formula = Class ~ Area + MajorAxisLength + MinorAxisLength +
##     Perimeter, family = binomial, data = raisin)
##
## Coefficients:
##                   Estimate Std. Error z value    Pr(>|z|)
## (Intercept)      3.3473160  4.2360894   0.790      0.4294
## Area            -0.0001113  0.0000562  -1.981      0.0476 *
## MajorAxisLength  0.0321112  0.0148931   2.156      0.0311 *
## MinorAxisLength  0.0682675  0.0209138   3.264      0.0011 **
## Perimeter       -0.0219106  0.0044520  -4.921 0.000000859 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1247.7  on 899  degrees of freedom
## Residual deviance:  618.6  on 895  degrees of freedom
## AIC: 628.6
##
## Number of Fisher Scoring iterations: 7
```

```r
par(mfrow=c(2,2))
plot(model2,1:4)
```


Residuals vs Fitted
Pearson Residuals
Predicted values
Q−Q Residuals
|Std. Deviance resid.|
Theoretical Quantiles
Scale−Location
√|Std. Pearson resid.|
Predicted values
Cook's distance
Cook's distance
Obs. number

c. Evaluation:

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
# Prediction
logistic_models <- list(fullmodel,model1,model2, nullmodel)
Accuracy_score <- c()
confusion_matrix <-list()
AUC_score<-c()
AIC_score<-c()

for (i in seq_along(logistic_models)) {
  probabilities <- predict(logistic_models[[i]], newdata = raisin, type="response")
  predictions <- ifelse(probabilities>0.5,"Kecimen","Besni")
  cm<-table(prediction =predictions, actual=raisin$Class )
  confusion_matrix[[i]]<-cm
  acc<-sum(diag(cm))/900
  Accuracy_score[[i]]<-acc
  roc_obj<-roc(raisin$Class,probabilities)
  auc_score<-auc(roc_obj)
  AUC_score[[i]]<-auc_score
  aic_score<-AIC(logistic_models[[i]])
  AIC_score[[i]]<-aic_score
}
```

```
## Setting levels: control = Besni, case = Kecimen

## Setting direction: controls < cases

## Setting levels: control = Besni, case = Kecimen

## Setting direction: controls < cases

## Setting levels: control = Besni, case = Kecimen

## Setting direction: controls < cases

## Setting levels: control = Besni, case = Kecimen

## Setting direction: controls < cases
```

```r
models=c('fullmodel','model1','model2', 'nullmodel')
cbind(models,Accuracy_score,AUC_score,AIC_score)
```

```
##      models      Accuracy_score AUC_score AIC_score
## [1,] "fullmodel" 0.8577778      0.9279111 624.9814
## [2,] "model1"    0.8555556      0.9278864 621.6371
## [3,] "model2"    0.8611111      0.9333531 628.6019
## [4,] "nullmodel" 0.5            0.5       1249.665
```

```r
logic_tb<-data.frame(Models=models,
                     Num.Predictors= c(7,5,4,0),
                     Accuracy=array(unlist(Accuracy_score), dim = c(length(Accuracy_score))),
                     AUC=array(unlist(AUC_score), dim = c(length(AUC_score))),
                     AIC=array(unlist(AIC_score), dim = c(length(AIC_score)))
                     )
```

```r
gt(logic_tb)%>%
  tab_header(
    title = "MULTIPLE LOGISTICS REGRESSION MODELS"
  )
```

MULTIPLE LOGISTICS REGRESSION MODELS

Models	Num.Predictors	Accuracy	AUC	AIC
fullmodel	7	0.8577778	0.9279111	624.9814
model1	5	0.8555556	0.9278864	621.6371
model2	4	0.8611111	0.9333531	628.6019
nullmodel	0	0.5000000	0.5000000	1249.6649

=> Conclusion: 'transform1' has the good value in most of metrics, however the p_value in the summary table shows that the coefficient are not significant. The full model also did a great job in clasification of the raisin model, however the evaluate metric is lower than 'model2'. In conclusion, we choose 'model2' which has 4 predictors (Area, MajorAxisLength, MinorAxisLength, Perimeter) as the final model.

4. Cross validation:

a. Split data:

```r
set.seed(666)
n<-nrow(raisin)
train_index<-sample(1:n,round(0.7*n))
trainset<-raisin[train_index,]
testset<-raisin[-train_index,]
```

b. Our model:

```r
# Fit model on train set
glm.train<-glm(Class ~  Area + MajorAxisLength + MinorAxisLength +
    Perimeter, data = trainset, family = binomial)
summary(glm.train)
```

```
##
## Call:
## glm(formula = Class ~ Area + MajorAxisLength + MinorAxisLength +
##     Perimeter, family = binomial, data = trainset)
##
## Coefficients:
##                    Estimate  Std. Error z value Pr(>|z|)
## (Intercept)     -0.21208366  4.88890062  -0.043 0.965398
## Area            -0.00015278  0.00006573  -2.325 0.020099 *
## MajorAxisLength  0.03336397  0.01723271   1.936 0.052857 .
## MinorAxisLength  0.07633034  0.02431499   3.139 0.001694 **
## Perimeter       -0.01798013  0.00504396  -3.565 0.000364 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 873.37  on 629  degrees of freedom
## Residual deviance: 449.78  on 625  degrees of freedom
## AIC: 459.78
##
## Number of Fisher Scoring iterations: 7
```

```r
# Evaluate
prob.test <- predict (glm.train, newdata=testset, type= "response")
preds.test<- ifelse(prob.test >0.5,"Kecimen","Besni")
# Confusion matrix
cm1<- table (prediction = preds.test,
            actual= testset$Class)
addmargins(cm1)
```

```
##            actual
## prediction Besni Kecimen Sum
##    Besni     109       9 118
##    Kecimen    26     126 152
##    Sum       135     135 270
```

```r
# Accuracy
Accuracy1<-sum(diag(cm1))/270
#Sensitivity (TP) identify Kecimen  type of raisin
Sensitivity1 <-cm1[2,2]/135
# Specificity (TF)
Specificity1<-cm1[1,1]/135
# AUC
roc.test <- roc(testset$Class,prob.test)
```

```
## Setting levels: control = Besni, case = Kecimen
```

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

```r
auc_glm<-auc(roc.test)
```

c. Decision Tree:

```r
# Fit trainset with decision tree
library(rpart)
tree.train<-rpart(Class~.,data=trainset, method= "class")
```

```r
# Evaluate
tree_prob <- predict (tree.train, newdata=testset)[,2]
tree_pred <- predict (tree.train, newdata=testset, type="class")
# Confusion matrix
cm2<- table (prediction = tree_pred,
            actual= testset$Class)
addmargins(cm2)
```

```
##           actual
## prediction Besni Kecimen Sum
##    Besni     108       9 117
##    Kecimen    27     126 153
##    Sum       135     135 270
```

```r
# Accuracy
Accuracy2<-sum(diag(cm2))/270
#Sensitivity (TP) identify Kecimen  type of raisin
Sensitivity2 <-cm2[2,2]/135
# Specificity (TF)
Specificity2<-cm2[1,1]/135
# AUC
tree.roc <- roc(testset$Class,tree_prob)
```

```
## Setting levels: control = Besni, case = Kecimen
```

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

```r
auc_tree<-auc(tree.roc)
```

d. Random Forest:

```r
# Fit trainset with random forest
# library(randomForest)
set.seed(123)
rf.train<-randomForest(Class~.,data=trainset,type="classification")
```

```r
# Evaluate
rf_prob <- predict (rf.train, newdata=testset,type="prob")[,2]
rf_pred <- predict (rf.train, newdata=testset, type="class")
# Confusion matrix
cm3<- table (prediction = rf_pred,
            actual= testset$Class)
addmargins(cm3)
```

```
##           actual
## prediction Besni Kecimen Sum
##    Besni     108      11 119
##    Kecimen    27     124 151
##    Sum       135     135 270
```

```r
# Accuracy
Accuracy3<-sum(diag(cm3))/270
#Sensitivity (TP) identify Kecimen  type of raisin
Sensitivity3 <-cm3[2,2]/135
# Specificity (TF)
Specificity3<-cm3[1,1]/135
# AUC
rf.roc <- roc(testset$Class,rf_prob)
```

```
## Setting levels: control = Besni, case = Kecimen
```

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

```r
auc_rf<-auc(rf.roc)
```

e. Comparative table:

```
tb1<-data.frame(Models= c("Logistic Regression","Decision Tree","Random Forest"),
           Accuracy= c(round(Accuracy1,3),round(Accuracy2,3),round(Accuracy3,3)),
           Sensitivity=c(round(Sensitivity1,3),round(Sensitivity2,3),round(Sensitivity3,3)),
           Specificity=c(round(Specificity1,3),round(Specificity2,3),round(Specificity3,3)),
           AUC= c(round(auc_glm,3),round(auc_tree,3),round(auc_rf,3)))
gt(tb1)%>%
  tab_header(
    title = "MODELS COMPARATIVE TABLE"
  )
```

MODELS COMPARATIVE TABLE

Models	Accuracy	Sensitivity	Specificity	AUC
Logistic Regression	0.870	0.933	0.807	0.934
Decision Tree	0.867	0.933	0.800	0.867
Random Forest	0.859	0.919	0.800	0.932

Conclusion: Out final model did a great job in predicting the model, most of the metrics are higher than other machine learning method (decision tree, random forest)