i. **Introduction:**

The goal of this project is to explore various classification techniques. In this example, I am working with automotive data to determine whether the mean miles per gallon (mpg) is correlated with the features available in the dataset. Specifically, the aim is to classify vehicles based on their mpg and assess how well the dataset's features can predict this outcome.

a. **I**n this project, I will demonstrate the application of several classification techniques, including Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Naïve Bayes, Logistic Regression, and K-Nearest Neighbor (KNN) with the optimal value for K.

b. For this project, the miles per gallon (mpg) variable will be treated as categorical. The median mpg will be used as a threshold to create a binary classification (0 or 1). This binary classification will serve as the response variable, representing the outcome we aim to predict.

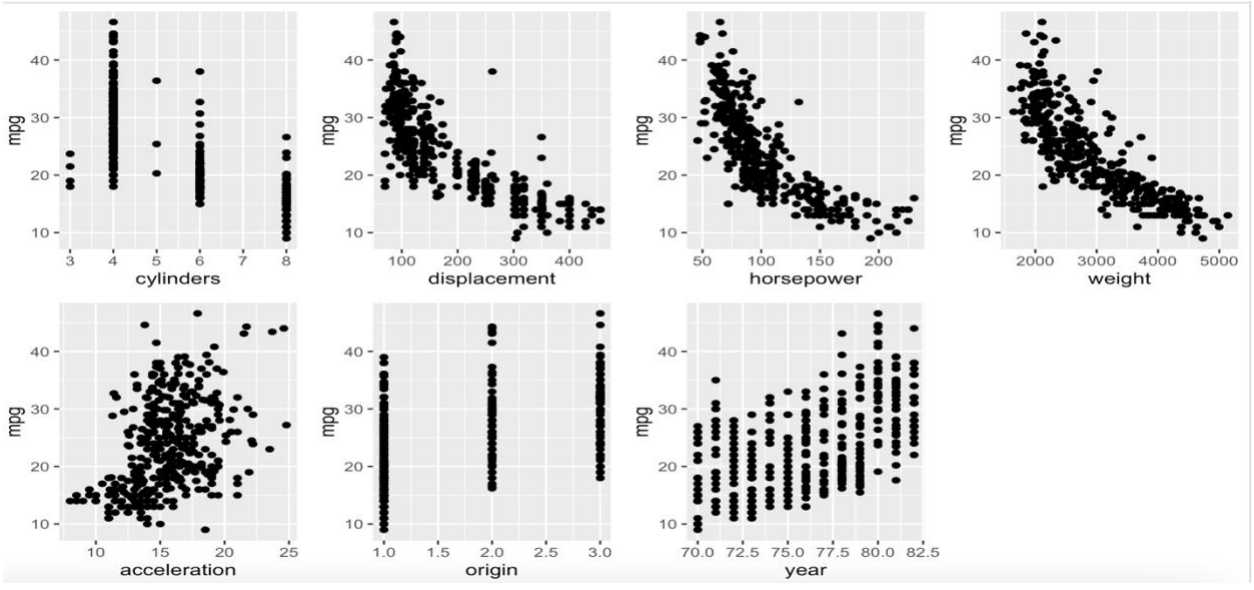
c. The data is split every 5th row, with those rows assigned to the training set and the rest to the testing set, resulting in approximately an 80-20 split. However, given the small dataset size, a single split may not be sufficient to validate the model. For more robust validation, future experiments should consider using cross-validation (CV).

ii. Exploratory Data Analysis:

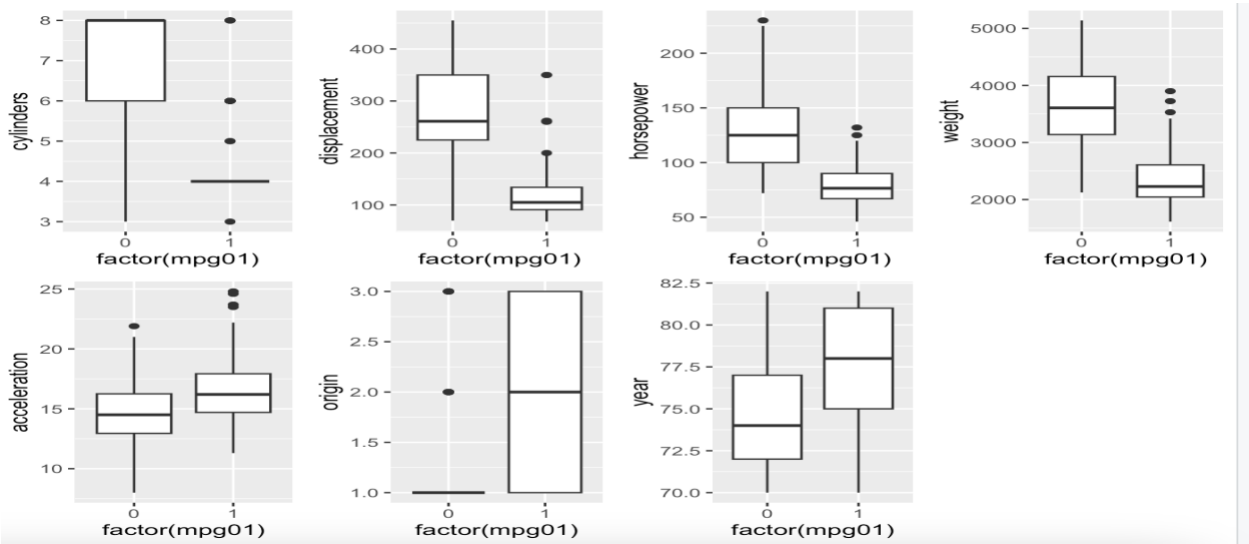
a. First, I created a set of scatter plots to visualize the relationship between our response variable and the available features. This helps identify any potential correlations or patterns that could inform the classification models.

b. Next, I created boxplots to display the distribution of the response variable against the other features. This aims to understand further how the features might influence or vary with the response variable, providing insights into potential predictors for the models.

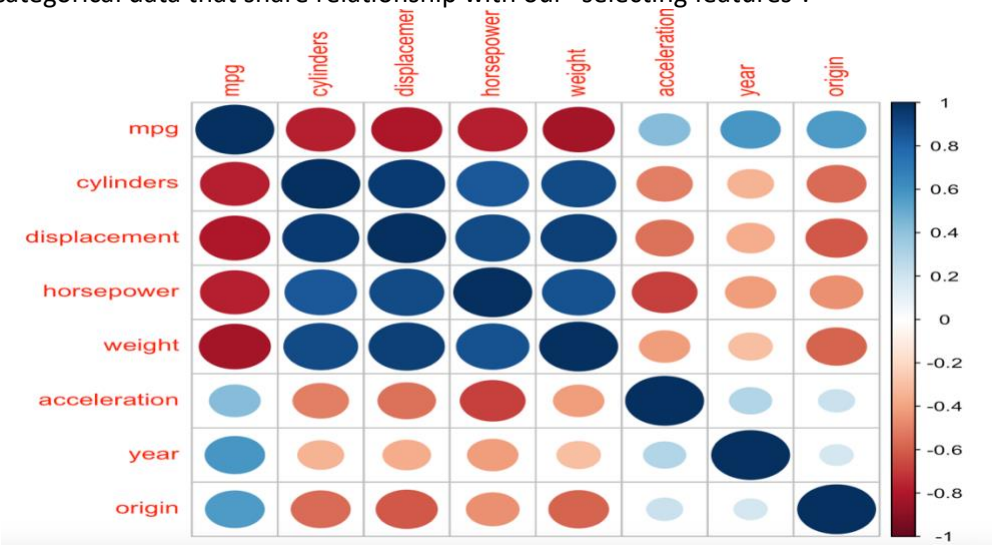
c. The scatter plot highlights a strong relationship between MPG and displacement, horsepower, weight, and acceleration. Additionally, the year variable exhibits a positive linear relationship with MPG. Since cylinder and origin are categorical features, they may also play a role in the model. Based on this analysis, I would include displacement, horsepower, weight, and acceleration as key predictors while retaining cylinder and origin as categorical features.



d. The boxplot demonstrates a balanced distribution of values above and below the median. However, for certain features, the relationship is not clearly defined. Based on this analysis, I would include all features in the model.



e. To further validate the features shown in the graphs, a correlation chart would provide clearer insight by indicating which features are strongly correlated within the range of [-1,1]. Based on this chart, the most correlated features are cylinder, displacement, horsepower, and weight. Additionally, origin should be retained as it is part of the categorical data that share relationship with our "selecting features".

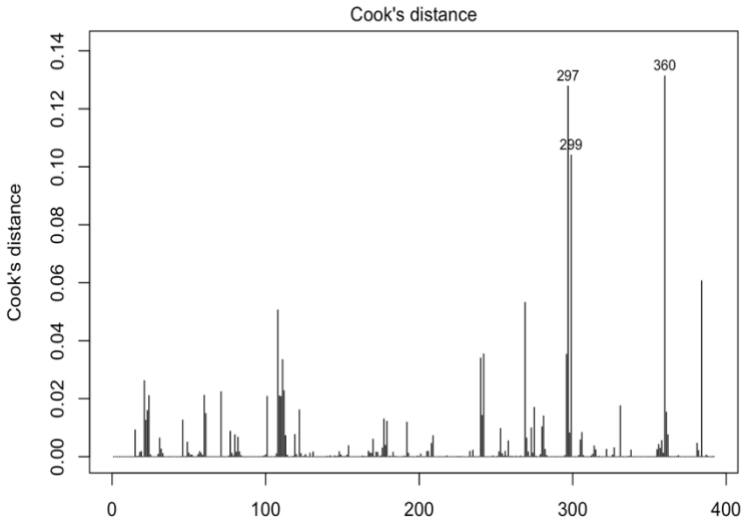


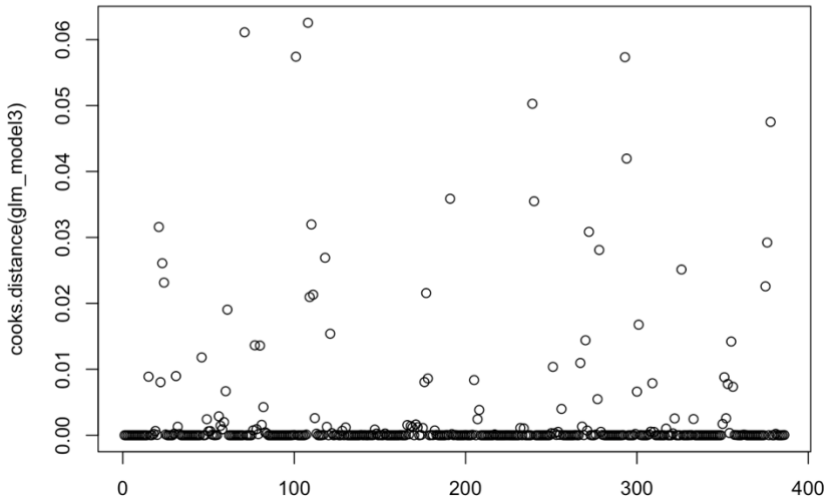
f. At this stage, I am fairly confident in selecting features, but I am not entirely convinced that all features in the dataset should be excluded. For example, acceleration and horsepower, as well as origin and cylinder, displacement, and weight, may still be relevant.

i. To further validate feature selection, I built a simple logistic regression model using all available features and performed stepwise regression (direction="both") based on AIC scores. The selected features were horsepower, weight, year, and origin. Since this did not fully align with my initial findings, I proceeded to check model assumptions. I compared both models and found that the new model distribution had a slight right skew. Based on qqnorm plot of residual, it's high stretch on the tail. Additionally, I checked for variance inflation factors (VIF) and found that displacement had a high VIF, indicating multicollinearity.

cylinders	displacement	horsepower	weight	acceleration	year	origin
4.900837	8.852836	2.691385	4.865771	2.651544	1.686770	1.901268

ii. I then conducted Cook's distance analysis to identify potential outliers in the data, as removing outliers could improve model performance. The initial Cook's distance plot highlighted three influential points (297, 299, 360). I iteratively removed a total of six rows until the plot appeared normalized. I concluded with 6 features besides displacement.





iii. **Methods:**

a. **LDA:** lda(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)
   i. **Testing Error:** .0909
   ii. **Training Error:** .0809

b. **QDA:** qda(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)
   i. **Testing Error:** .104
   ii. **Training Error:** .0809

c. **Naitve Bayes:** naiveBayes(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)

i. **Testing Error:** .104
ii. **Training Error:** .077

d. Logistic Regression: multinom(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)

i. **Testing Error:** .078
ii. **Training Error:** .061

e. KNN with optimize K=3:

i. **Testing Error:** .104
ii. **Training Error:** .058

f. KNN with optimize K=9: I ran a loop of 100 iterations to find the best k to use for the future model. The best k value fluctuates between 6, 8, and 9 as the best k selected based on 100 run. I chose 9 because 3 is my favorite number, and 3^2 is 9.

	best_k	Freq
1	6	18
2	8	45
3	9	37

i. **Testing Error:** .091
ii. **Training Error:** .107

iv. **Result/Finding:** To support my findings, I ran 100 iterations and calculated the average training and testing errors. Based on the dataset and the selected features, logistic regression emerged as the best-performing model. This is likely due to the small size of the dataset and the relatively linear relationships between the features and the target variable, making logistic regression a strong fit. As for LDA and QDA, both models assume that each feature follows a normal distribution, which does not entirely hold for this dataset. Naïve Bayes assumes that all predictors are independent of one another, but in this case, some features share relationships, violating this assumption. For KNN, the model's performance is highly dependent on selecting the optimal K value. Since K can vary significantly, it can either improve or worsen the model's performance. Based on common sense, factors such as cylinder, weight, year, acceleration, and horsepower are crucial in determining a car's MPG. Additionally, the car's origin might also influence the prediction. I believe that all features in this dataset are valuable, and with slight transformations, they should fit the model well.

Model	Mean_Train	Mean_Test
LDA	0.081	0.091
QDA	0.081	0.104
Naive Bayes	0.078	0.104
Logistic Regression	0.061	0.078
KNN (Optimal K)	0.107	0.091

CODE

```
k = 9
knn_er_train = mean(knn(train=train[,model_list],test=train[,model_list],cl=train$mpg01,k=k) !=
train$mpg01)
knn_er_test = mean(knn(train=train[,model_list],test=test[,model_list],cl=train$mpg01,k=k) !=
test$mpg01)


knn_er_trainK = as.numeric()
knn_er_testK = as.numeric()
kval = 1:10
run = 100
best_k = as.numeric(run)
for (r in 1:run){
  for (i in kval){
  knn_er_trainK[i] = round(mean(knn(train=train[,model_list],test=train[,model_list],cl=train$mpg01,k=i)
!= train$mpg01),3)
  knn_er_testK[i] = round(mean(knn(train=train[,model_list],test=test[,model_list],cl=train$mpg01,k=i) !=
test$mpg01),3)
  }
  knn = data.frame(k = kval, TrainingEr = knn_er_trainK, TestingEr = knn_er_testK)
  best_k[r] = knn[which.min(knn$TestingEr), "k"]
}
kval_count = as.data.frame(table(best_k))
kval_count
sum(kval_count$Freq)

print(knn) # k=6


final_train = NULL
final_test = NULL
B = 100
for (i in 1:B){
  splitting = seq(5,nrow(new_auto), by = 5)
  train = new_auto[-splitting,]
  test = new_auto[splitting,]

  model_list = c("cylinders","horsepower", "weight", "acceleration", "year", "origin")
  ####### LDA #######
  lda_model = lda(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)
  lda_er_train = mean(predict(lda_model, newdata =train[,model_list])$class != train$mpg01)
  lda_er_test = mean(predict(lda_model, newdata = test[,model_list])$class != test$mpg01)

  ####### QDA #######
  qda_model = qda(mpg01~cylinders + horsepower + weight + acceleration + year + origin, data=train)
  qda_er_train = mean(predict(qda_model, newdata = train[,model_list])$class != train$mpg01)
```

```
  qda_er_test = mean(predict(qda_model, newdata = test[,model_list])$class != test$mpg01)


  ####### Naïve Bayes #######
  nbayes_model = naiveBayes(mpg01~cylinders + horsepower + weight + acceleration + year + origin,
data=train)
  nbayes_er_train = mean(predict(nbayes_model, train[,model_list]) != train$mpg01)
  nbayes_er_test = mean(predict(nbayes_model, test[,model_list]) != test$mpg01)

  ####### Logistic Regression #######
  log_model = multinom(mpg01~cylinders + horsepower + weight + acceleration + year + origin,
data=train)
  log_er_train = mean(predict(log_model, train[,model_list])!= train$mpg01)
  log_er_test = mean(predict(log_model, test[,model_list])!= test$mpg01)

  ##### KNN K = ? #####
  knn_er_trainK = round(mean(knn(train=train[,model_list],test=train[,model_list],cl=train$mpg01,k=9)
!= train$mpg01),3)
  knn_er_testK = round(mean(knn(train=train[,model_list],test=test[,model_list],cl=train$mpg01,k=9) !=
test$mpg01),3)

  final_train = rbind(final_train, cbind(lda_er_train, qda_er_train, nbayes_er_train, log_er_train,
knn_er_trainK))
  final_test = rbind(final_test, cbind(lda_er_test, qda_er_test, nbayes_er_test, log_er_test,
knn_er_testK))
}


dim(final_train)
apply(final_train, 2, mean)
apply(final_train, 2, var)

dim(final_test)
apply(final_test, 2, mean)
apply(final_test, 2, var)

pretty_tab = data.frame(Model = c("LDA", "QDA", "Naive Bayes", "Logistic Regression", "KNN (Optimal
K)"),
                        Mean_Train = round(apply(final_train, 2, mean),3),
                        Mean_Test = round(apply(final_test, 2, mean),3)
                          )
```