Classification

1. *Load and study the dataset and comment on the dataset's characteristics / structure.*
    - There are 215 observations in total of 6 variables in integer and number form.
    - There is no NA/ null/ duplicate value in the dataset
    - Class distribution:

1 (Healthy)	2 (Hyperthyroidism)	3 (Hypothyroidism)
150	35	30

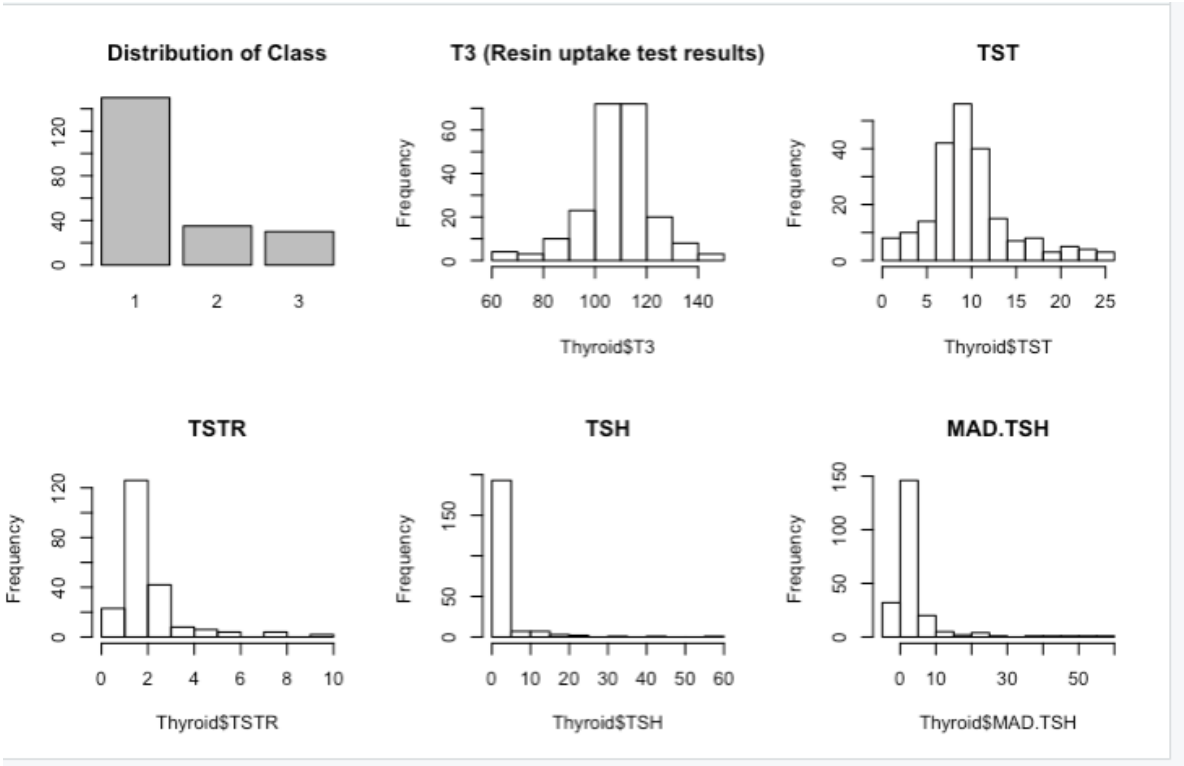
2. *Conduct exploratory data analysis for all variables to get a better understanding about each of them (numerically and visually). Comment on the results and findings.*

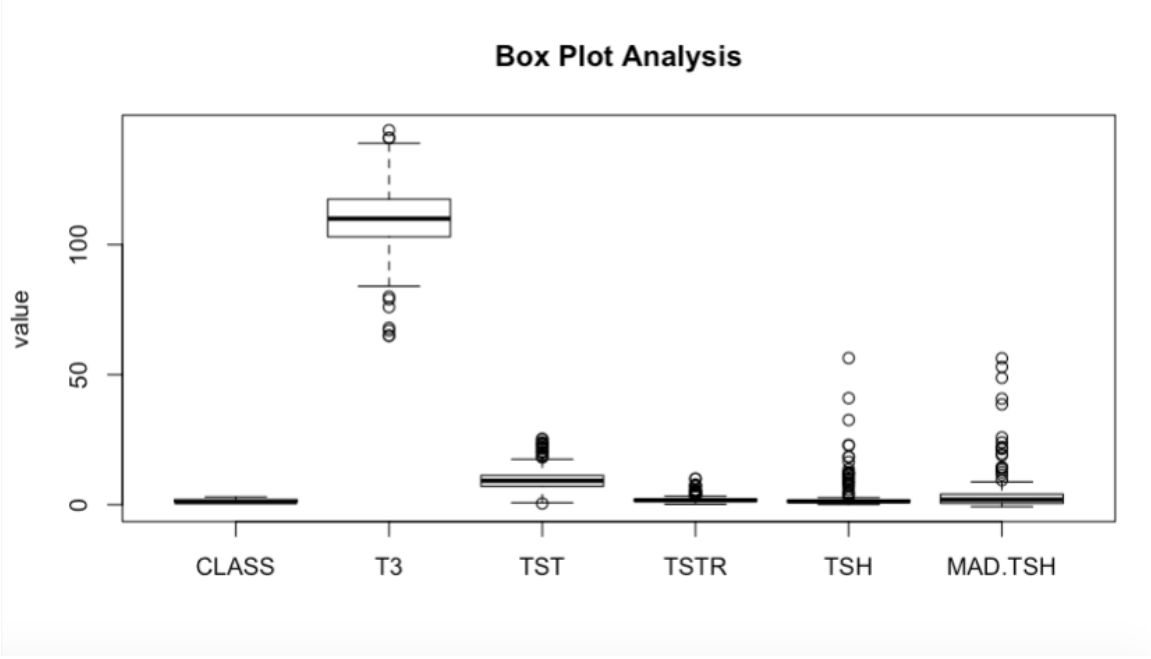
```
> summary(Thyroid)
     CLASS             T3             TST              TSTR            TSH          MAD.TSH
 Min.   :1.000   Min.   : 65.0   Min.   : 0.500   Min.   : 0.20   Min.   : 0.10   Min.   :-0.700
 1st Qu.:1.000   1st Qu.:103.0   1st Qu.: 7.100   1st Qu.: 1.35   1st Qu.: 1.00   1st Qu.: 0.550
 Median :1.000   Median :110.0   Median : 9.200   Median : 1.70   Median : 1.30   Median : 2.000
 Mean   :1.442   Mean   :109.6   Mean   : 9.805   Mean   : 2.05   Mean   : 2.88   Mean   : 4.199
 3rd Qu.:2.000   3rd Qu.:117.5   3rd Qu.:11.300   3rd Qu.: 2.20   3rd Qu.: 1.70   3rd Qu.: 4.100
 Max.   :3.000   Max.   :144.0   Max.   :25.300   Max.   :10.00   Max.   :56.40   Max.   :56.300
```

- CLASS: class attribute (healthy (1) or hyperthyroidism (2) or hypothyroidism (3)). Most cases are in class 1 (70%).
- T3: T3-resin uptake test results, range from 65 to 144, mean is 109.6.
- TST: Total Serum thyroxin as measured by the isotopic displacement method, range from 0.5 to 25.3, mean is 9.8
- TSTR: Total serum triiodothyronine as measured by radioimmunoassay, range from 0.2 to 10, mean is 2.05
- TSH: Basal thyroid-stimulating hormone (TSH) as measured by radioimmunoassay, range from 0.1 to 56.4, mean is 2.9. Majority of cases are from 0-5.

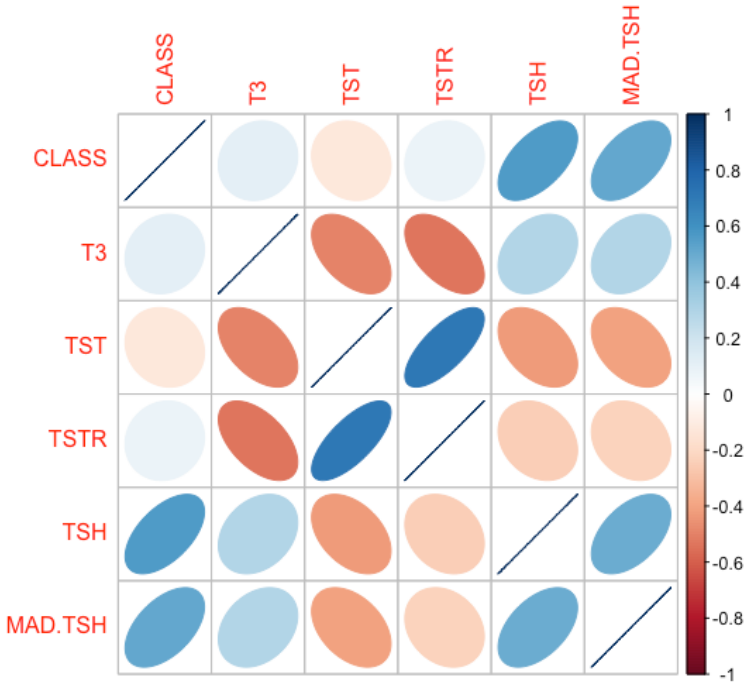
- MAD-TSH: Maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value, range from -0.7 to 56.3, mean is 4.1. Majority of cases are from 0-10.

Basic plotting of all variables are constructed. There are 5 variables have the outliers which are T3, TST, TSTR, TSH, MAD.TSH





- TST, TSTR have strong negative relationship with T3.



3. *Prepare the data for further analysis. Normalize the data using the min-max method. Using stratified random sampling, divide the data for training and testing as 70% of the data for the training set and 30% of the data for the test set.*

- Normalized data using min-max method

min-max normalization $x_{new} = \frac{(x - x_{min})}{(x_{max} - x_{min})}$

- Command:

Create a min-max function then create Thyroid_2 using the function. As the function will normalize all variables so we need to add back the original CLASS to Thyroid_2 using mutate function.

```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
Thyroid_2<-normalize(Thyroid)
str(Thyroid_2)

Thyroid_2<-Thyroid_2 %>%
  mutate(CLASS=Thyroid$CLASS)
```

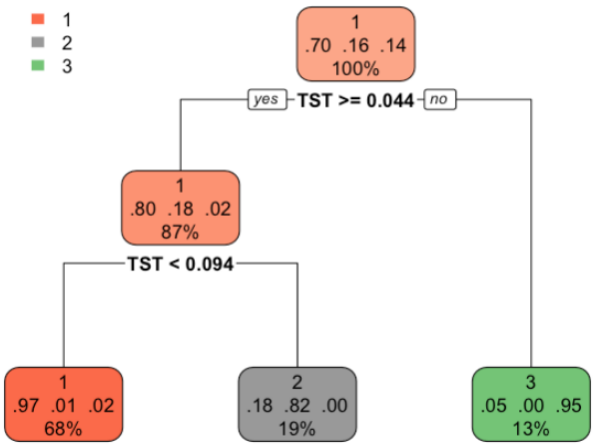
- Divide the data for training and testing as 70% of the data for the training set and 30% of the data for the testing set. There are 150 observations in the training set and 65 observation in the testing set.
- Visualize training and testing data using the plot of two variables T3 and TST:

4. *Conduct classification with two classification models (decision trees and k-nearest neighbor) to predict the class labels. Construct for each classifier a model that neither overfits nor underfits the data. Provide the corresponding parameter k (Knn) and minimum leave size (CART). Comment and explain your reasoning and findings in detail.*
5. *Discuss the classification performance of both models in detail. [Do not use "eval_class" function and you do not need to present "Recall" and "Precision" scores]*

(*)The below explanation is intended for both question 4 and 5

- **<u>Decision tree:</u>**

```
tree_model<-rpart(CLASS~., data=train, method='class',minbucket=20)
rpart.plot(tree_model)
```



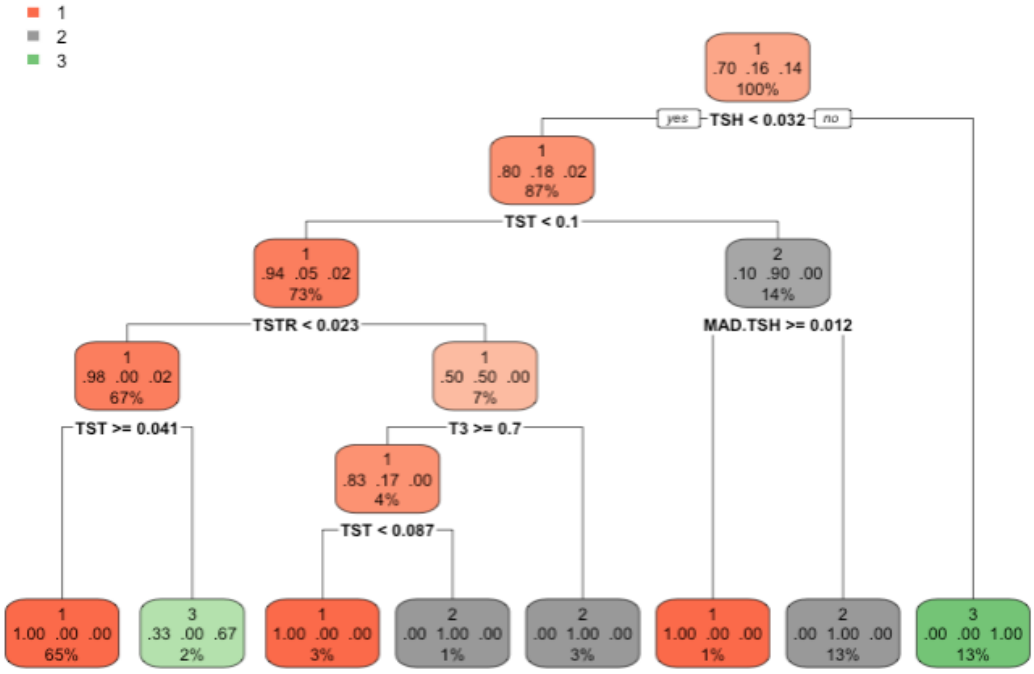

```
tabTrain
 treeTrainpred
   1  2  3
1 99  5  1
2  1 23  0
3  2  0 19
tabTest
 treeTestpred
   1  2  3
1 43  1  1
2  3  8  0
3  1  0  8
```

- The model is classified based on the TST variable, minbucket = 20.
- From Train table, there are 99 people are correctly classified positively in case 1 (TP). There are also 23 and 19 people are correctly classified negatively in case 2 and 3 (TN). There are 3 people are falsely classified as positive (FP) and 6 people are falsely classified as negative (FN).
- From Test table, there are 43 people are correctly classified positively in case 1 (TP). There are also 8 and 8 people are correctly classified negatively in case 2 and 3 (TN). There are 4 people are falsely classified as positive (FP) and 2 people are falsely classified as negative (FN).
- Accuracy rate of Train is **94%** and Test is **91%**, variance between Train and Test is 3%, which is quite low and suitable for the model.

○ <u>Construct an overfit model</u>

```
tree_model_2<-rpart(CLASS~., data=train, method='class', minbucket=1)
rpart.plot(tree_model_2)
```

- The model is classified based on TSH, TST, TSTR, T3 and MAD.TSH variables, minbucket = 1, which is very complex and intensive model.
- Accuracy rate of Train is **99%** and Test is **89%,** variance between Train and Test is 11%, which is quite high.



- Choosing the optimal leaves for the tree model: from 20-25 leaves where low bias and low variance between Train and Test data.

- **KNN Model**

```
KnnTrain_pred=knn(train[,2:6],train[,2:6],as.factor(train$CLASS),10)
```

```
KnnTrain_pred
   1  2  3
1 105  0  0
2   8 16  0
3   7  0 14
```

- TP: 105 patients are healthy (1)
- TN: 30 patients in total (16 people in case 2 and 14 people in case 3)
- FP: 15 patients
- FN: 0 patients

- **Accuracy KNNTrain: 90%**

```
KnnTest_pred=knn(train[,2:6],test[,2:6],as.factor(train$CLASS),10)
```

```
KnnTest_pred
   1  2  3
1 45  0  0
2  5  6  0
3  2  0  7
```

- TP: 45 patients are healthy (1)
- TN: 13 patients in total (6 people in case 2 and 7 people in case 3)
- FP: 15 patients
- FN: 0 patients

- **Accuracy KNNTest: 89%**
- Construct an overfit model

```
KnnTrain_pred_Un=knn(train[,2:6],train[,2:6],as.factor(train$CLASS),2)
KnnTest_pred_Un=knn(train[,2:6],test[,2:6],as.factor(train$CLASS),2)
```

```
KnnTrain_pred_Un
   1  2  3
1 103  0  2
2   0 24  0
3   2  0 19
```

- TP: 103 patients are healthy (1)
- TN: 43 patients in total (24 people in case 2 and 19 people in case 3)
- FP: 2 patients
- FN: 2 patients

- **Accuracy KNNTrain: 97.3%**

```
KnnTest_pred_Un
   1  2  3
1 44  0  1
2  3  8  0
3  3  0  6
```

- TP: 44 patients are healthy (1)
- TN: 14 patients in total (8 people in case 2 and 6 people in case 3)
- FP: 6 patients
- FN: 1 patient

- **Accuracy KNNTest: 89.2%**
- Variance between KKNTrain and KNNTest is high about 8%. This model pays a lot of attention to training data and does not generalize on the whole data set.

6. *Imagine you use a training and test set and choose the best parameter based on the test set performance. Is the performance of this method with the best performing parameter a good indicator of the performance on new observations that are not included in any of the two data sets [Generalization]. Explain your reasoning.*

   The method would be a good indicator for the performance on new observation that are not included in any of the two data sets. It could be understood as K-nearest neighbor classification algorithm when it will first compute the distance between a dataset and each new observation. As user has chosen the best parameter (k observations that are nearest to the new observation), we assign the most frequent class to k neighbors.

7. *What would be the optimal values for k based on KNN classification? What is the reason for your selection?*

- The optimal values for k based on KNN classification would be from 9-15. At this value, we could see low bias at 6%-12% error and also low variance between train and test data (1%-2%).

8. *Redo the KNN classification for the original data (without scaling). What do you observe? Compare the classification performances in both cases. Comment on the comparison and findings in detail.*

- Calculate the sample, training data and testing data based on the original data

```
sample_N=sample.split(Thyroid$CLASS,SplitRatio=0.7)
train_N=subset(Thyroid,sample==TRUE)
test_N=subset(Thyroid,sample==FALSE)
```

- Construct KNN model

```
KnnTrain_pred_N=knn(train_N[,2:6],train_N[,2:6],factor(train_N$CLASS),10)
KnnTest_pred_N=knn(train_N[,2:6],test_N[,2:6],factor(train_N$CLASS),10)
```

- Result: Accuracy for KNNTrain is 88.7% and KNNTest is 90.1%

```
KnnTrain_pred_N
    1   2   3
1 105   0   0
2   9  15   0
3   8   0  13
```

```
KnnTest_pred_N
   1  2  3
1 45  0  0
2  4  7  0
3  2  0  7
```

- There are high fluctuation of the variance of Train and Test data as regards to number of k neighbors. When k is smaller than 30, the accuracy of Test data is larger than Training data which is also unusual. It then very challenging to choose the optimal k as the larger k goes, the higher error for the model.
- Comparing to the scaling version, the original model also has much lower accuracy and instable data.

9. *Give a detailed conclusion on your findings and results. Consider and explain how these findings can help people in the medical field to treat the disease(s) as early as possible.*

   In conclusion, when the dataset has different scale range between variables, it is very necessary to normalize the data before moving to the classification. The model was built using the Decision Tree model with optimal leaves around 20-25 leaves and KNN model with optimal k neighbors from 9-15. It can be observed that Decision Tree model has higher accuracy with 94% (Train) and 91% (Test) while KNN model has lower accuracy rate with 90% (Train) and 89%(Test). The results also indicate that the Healthy outcome has been predicted quite accurate compare to Hypothyroidism and Hyperthyroidism classes.