Final Group Project Report

Thyroid Disease Prediction and Data Analysis

Contributing Partners:

Rahul Bharadwaj Machiraju(A20502085)
Sai Pavan Kunda(A20496516)
Rahul Maddula(A20488730)

1. ABSTRACT

One of the most prevalent disorders among women is thyroid disease. Thyroid illness frequently manifests as hypothyroidism. It is obvious that people with hypothyroidism are typically female. Because most people are unaware of that illness, it is quickly developing into a severe illness. It is crucial to catch it so that doctors can provide patients with better treatment. Machine learning illness prediction is a challenging task. In forecasting diseases, machine learning is crucial. Once more, unique feature selection methods have aided in the process of disease assumption and prediction. There are two different thyroid conditions: Hyperthyroid and Hypothyroid. For this, we used 4 Machine Learning models such as K-NN model, Random Forest, Multinomial Regression model, and Naive Bayes. By looking at the results, provide us with around 98% accuracy for all four classification algorithms. According to the findings, machine learning models are a better option for detecting thyroid disease in terms of accuracy and computational complexity.

2. OVERVIEW

2.1. Problem Statement

The thyroid is an endocrine gland located in the anterior region of the neck: its main task is to produce thyroid hormones, which are functional to our entire body. Its possible dysfunction can lead to the production of an insufficient or excessive amount of thyroid hormone. We have looked at various machine learning models, compared their accuracies, and made feature selections in an effort to identify the best model to predict hyperthyroid and hypothyroid. It is crucial to catch it early so that doctors can give patients better treatment to prevent it from becoming a significant problem.

2.2. Literature Review

1. Razia, S.; SwathiPrathyusha, P.; Krishna, N.V.; Sumana, N.S. A Comparative study of machine learning algorithms on thyroid disease prediction. Int. J. Eng. Technol. 2018, 7, 315

   In this paper, they used a sample size of 7200 samples with 21 attributes, they built machine learning models with accuracies of SVM(96%), MLR(91%), NB(6.31%) and DT(98%) that had 2 classes.

2. Salman, K.; Sonuç, E. Thyroid Disease Classification Using Machine Learning Algorithms. J. Phys. Conf. Ser. IOP Publ. 2021, 1963, 012140.

In this paper, they used a sample size of 1250 samples with 17 attributes, they build machine learning models with accuracies are SVM(92%), RF(91%), LDA(83.2%), and LR(91%) that has 3 classes.

3. Hosseinzadeh, M.; Ahmed, O.H.; Ghafour, M.Y.; Safara, F.; Hama, H.; Ali, S.; Vo, B.; Chiang, H.S. A multiple multilayer perceptron neural network with an adaptive learning algorithm for thyroid disease diagnosis in the internet of medical things. J. Supercomput. 2021, 77, 3616–3637

   In this paper, they used a sample size of 7200 sample with 21 attributes and they Build machine learning models with accuracies are multiple MLP(97%) with 3 classes.

4. Alyas, T.; Hamid, M.; Alissa, K.; Faiz, T.; Tabassum, N.; Ahmad, A. Empirical Method for Thyroid Disease Classification Using a Machine Learning Approach. BioMed Res. Int. 2022, 2022, 9809932.

   In this paper, they used a sample size of 3163 sample size with 21 attributes and they built machine learning models with accuracies DT, RF, KNN, and ANN, the best performance accuracy for RF is 94.8%. With 2 classes.

2.3 - Proposed Methodology

1.)Firstly, we need to identify the major target classes and prepare the data accordingly.
2.)Clean the data based on the target classes and arrange the data accordingly.
3.)Then perform exploratory data analysis on the data to identify the key features.
4.)After identifying the key features, split the data into train and test. Use the training data for training the designed models.
5.)After training, test the trained model with the testing data and see how well it performs.
6.)Finally, find the Accuracy, precision, and f-1 score of the models and conclude which among the two used classifiers perform well on the data and state the reasons why.

3. DATA PROCESSING

3.1. Pipeline Details

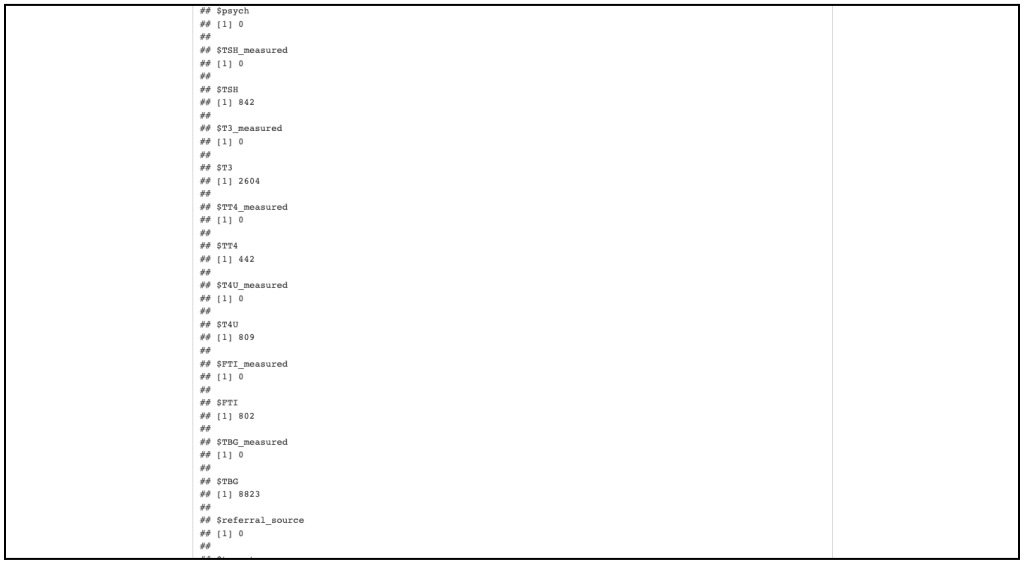
The dataset was found on Kaggle https://www.kaggle.com/datasets/emmanuelfwerr/thyroid-disease-data and describes Thyroid data for 9172 data with 31 features.

3.2 . Data Issues

Pre-processing is used to fix a variety of issues, such as noisy data, redundant data, missing data values, and more . High-quality data will produce high-quality outcomes and lower data mining costs. Pre-processing should be done on missing data.

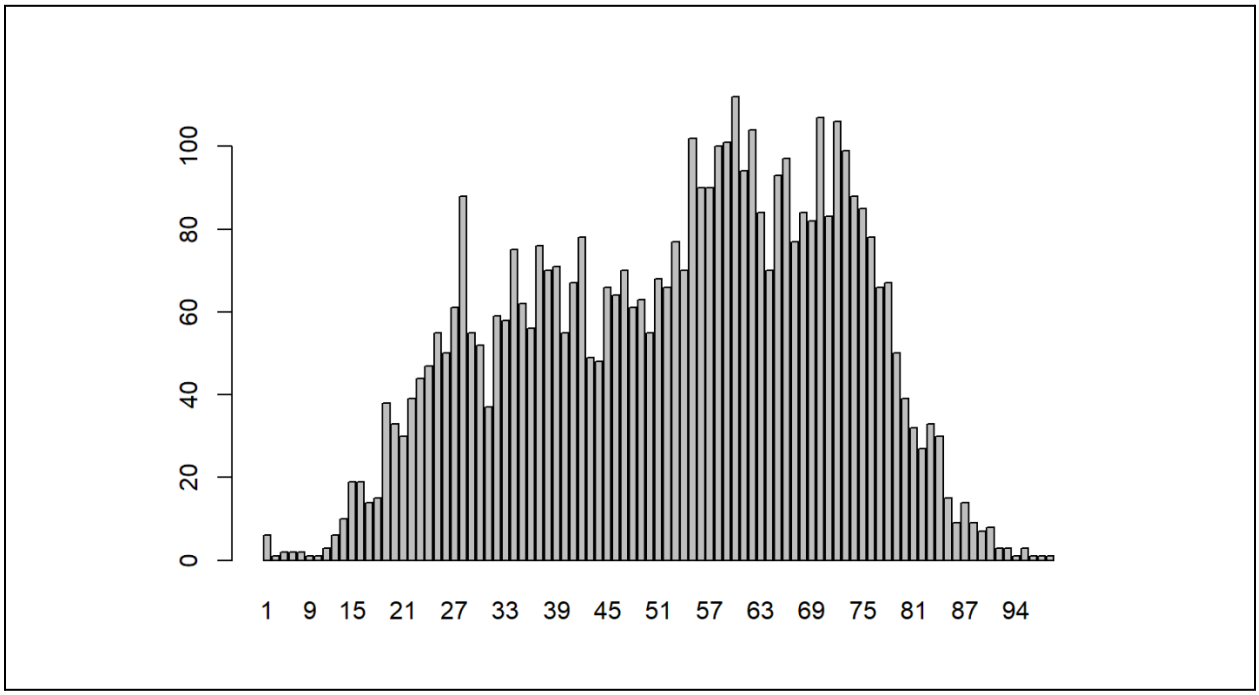
After looking at the data, we can see that there are many unwanted features, which we can clean by dropping the features, and Also we verified Null values in the dataset which can be dropped from the dataset. The class counts make it very evident that the dataset is highly imbalanced For example, the majority of the dataset's samples don't fit into any particular class. In order to obtain the standard dataset for our performance evaluation, data preparation is done. so we remapped the target variables into 3 labels ‘negative’, ‘hyperthyroid’, and ‘hypothyroid’.

From the below image, we can see Null values in some attributes, so we had to clean them, by removing the unnecessary rows and attributes.

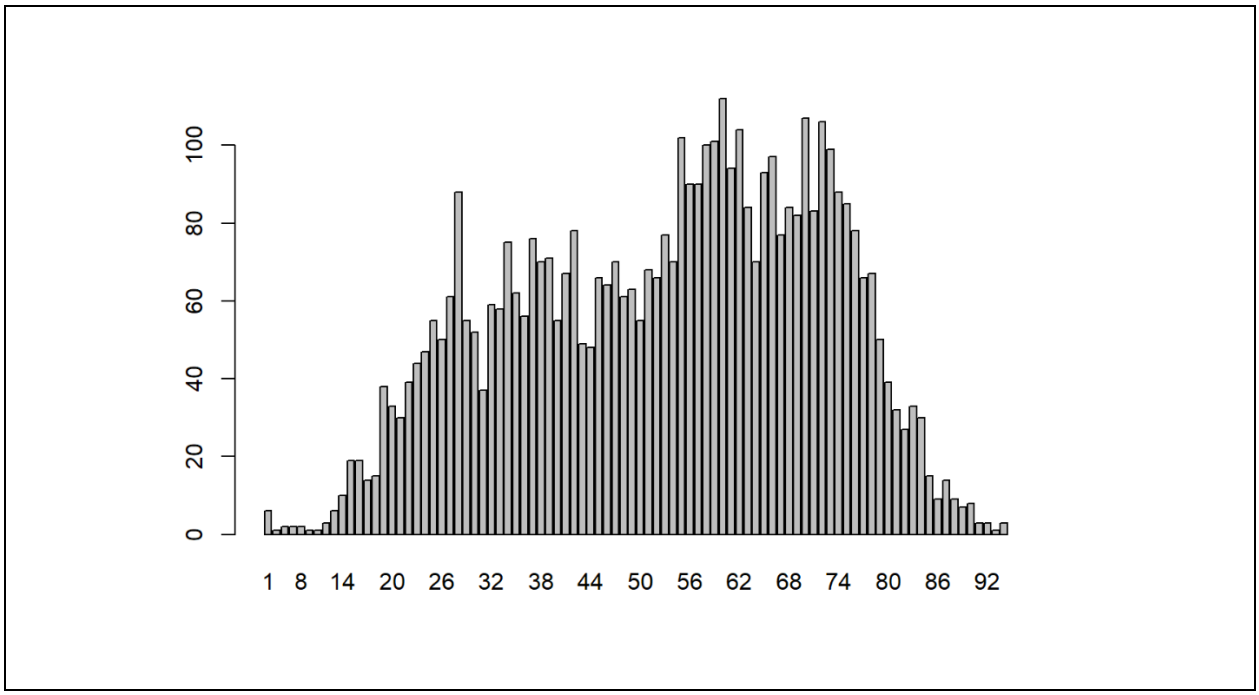


4. EXPLORATORY DATA ANALYSIS

We will visually represent a few different properties of the dataset we so meticulously cleaned and raised from its inception before training and assessing the success of our models.

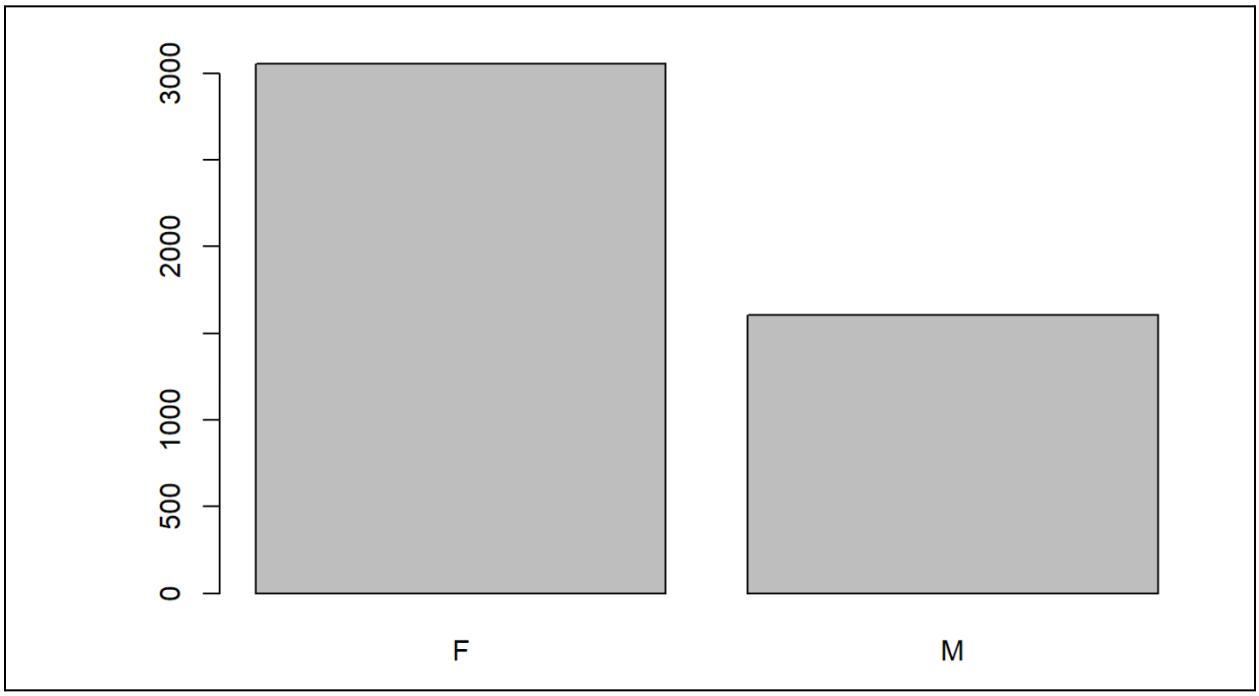


We examined the several age categories that the Thyroid falls under from the illustration from the above image.



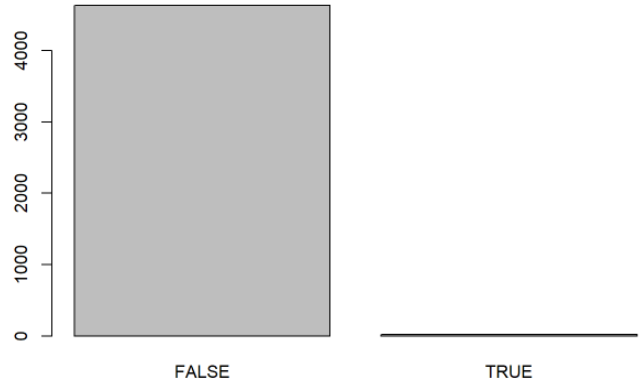
Now, we had taken people age less than 100, then examined the categories that the Thyroid falls under from the illustration from the above image.

We can understand from the images above that Thyroid affects most people between the ages of 50 and 75.



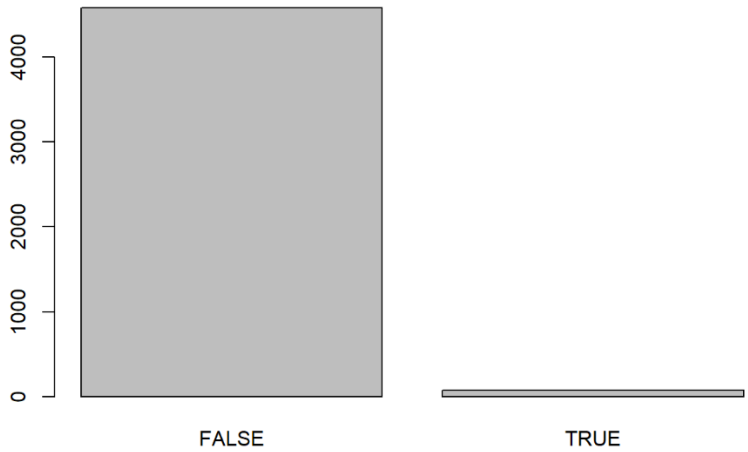
As can be seen from the figure above, females are more severely harmed by thyroid than males.

```
counts = table(df4$pregnant)
barplot(counts)
```



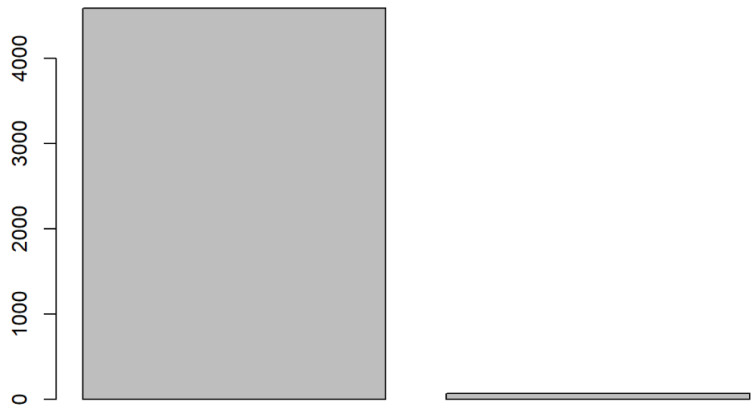
We can understand from the above image that, in the dataset, we had less pregnant women.

```
counts = table(df4$on_antithyroid_meds)
barplot(counts)
```



Now, we can look at people who are using the medicine for thyroid. We understand that most people do not use any medicine for thyroid.

```
counts = table(df4$thyroid_surgery)
barplot(counts)
```

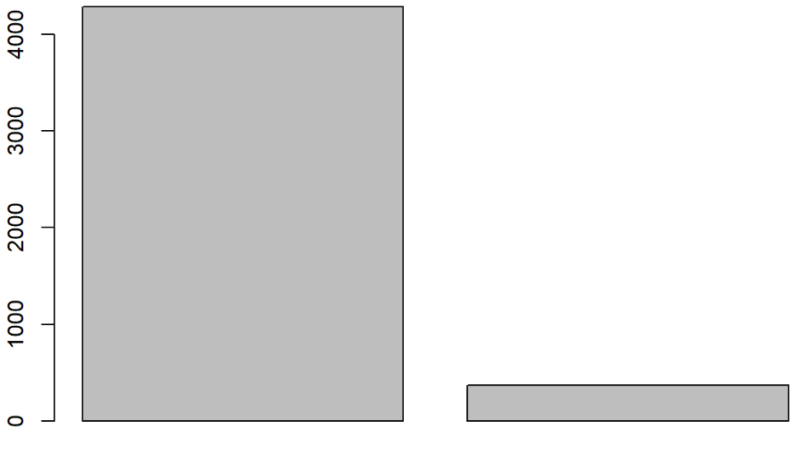


Now, we can look at people who went Thyriod surgery. We understand that most people does not went for any surgery.

```
counts = table(df4$query_hypothyroid)
barplot(counts)
```

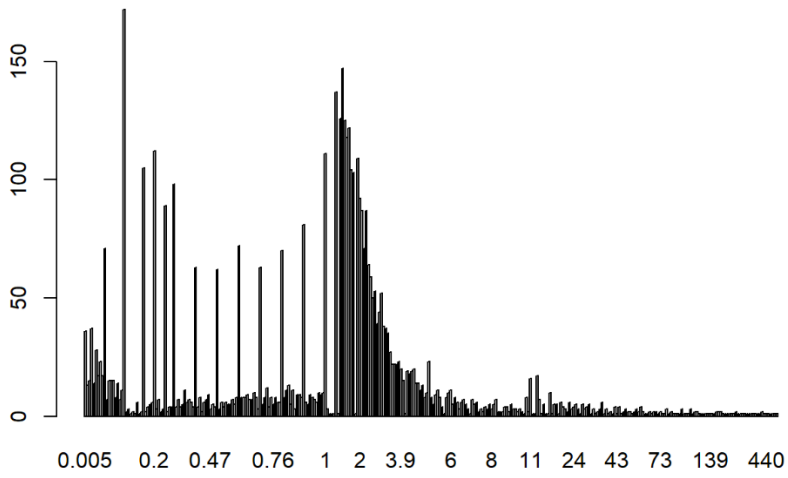
From the above image, we can say that, whether the patient believes they have hypothyroid or not.

```
counts = table(df4$query_hyperthyroid)
barplot(counts)
```

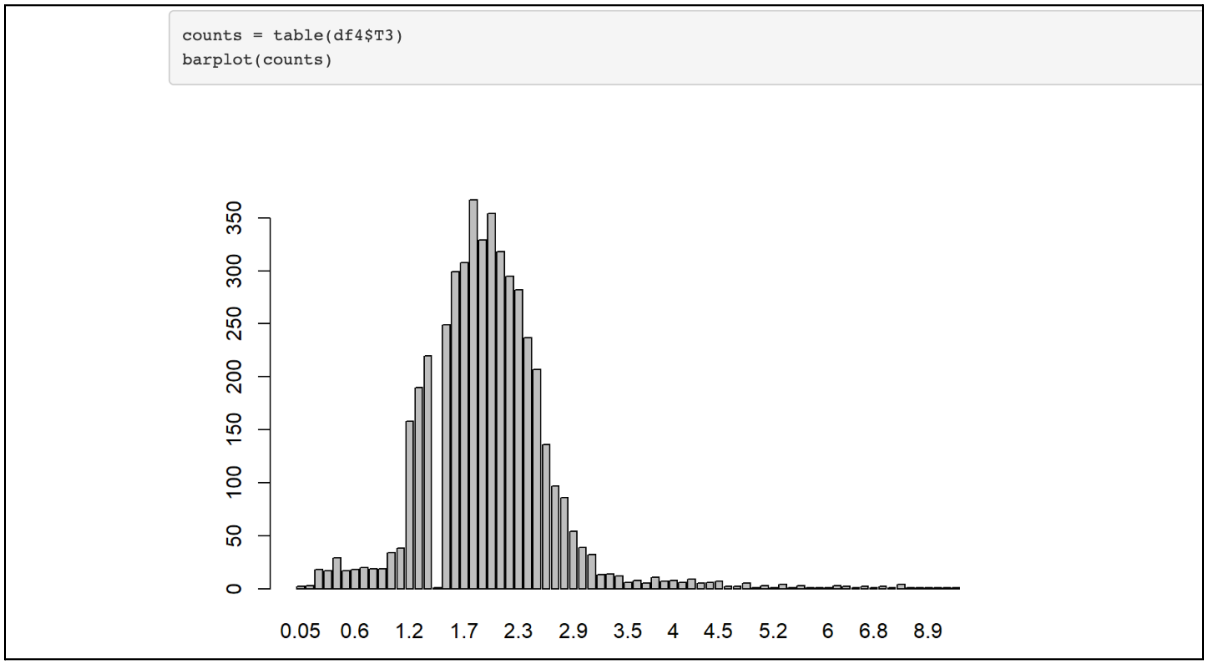


Same thing for hyperthyroid.

```
counts = table(df4$TSH)
barplot(counts)
```

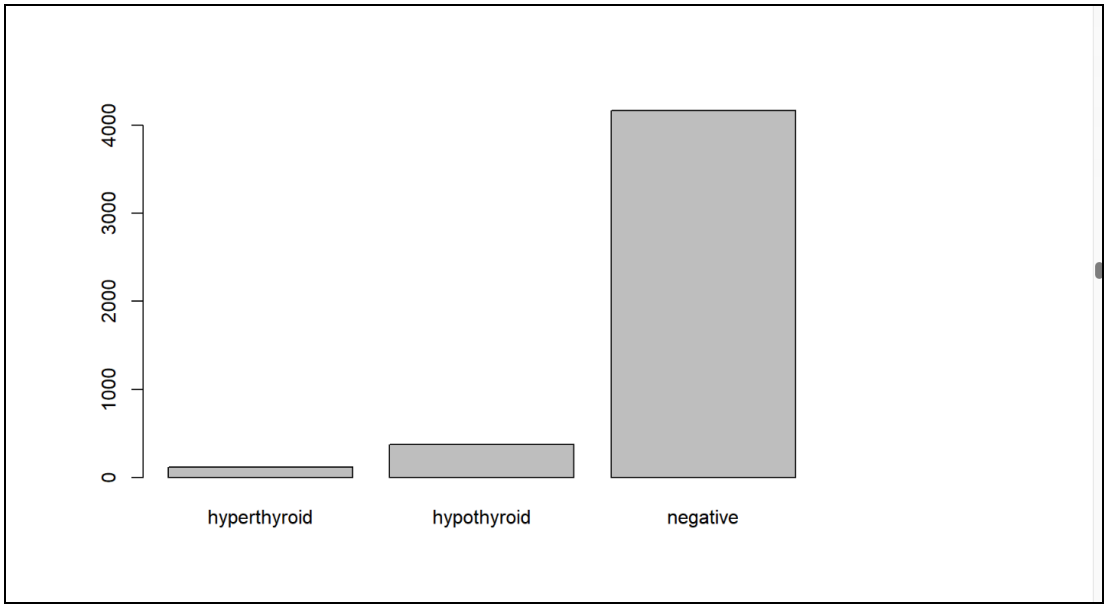


From image, we understand the thyroid stimulating hormone (TSH) level in data.

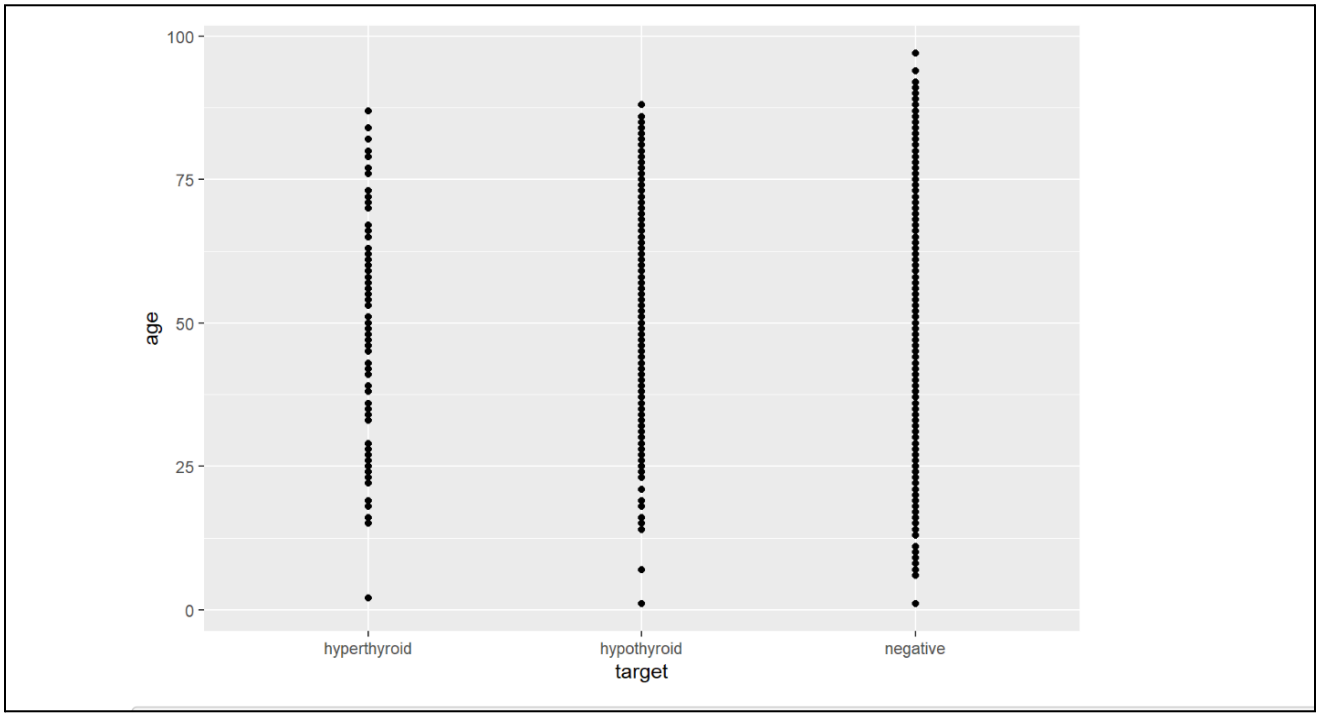


A high TSH level implies insufficient thyroid hormone production by the thyroid gland (hypothyroid). Conversely, a low TSH level typically signifies that the thyroid is overproducing thyroid hormone (hyperthyroid).

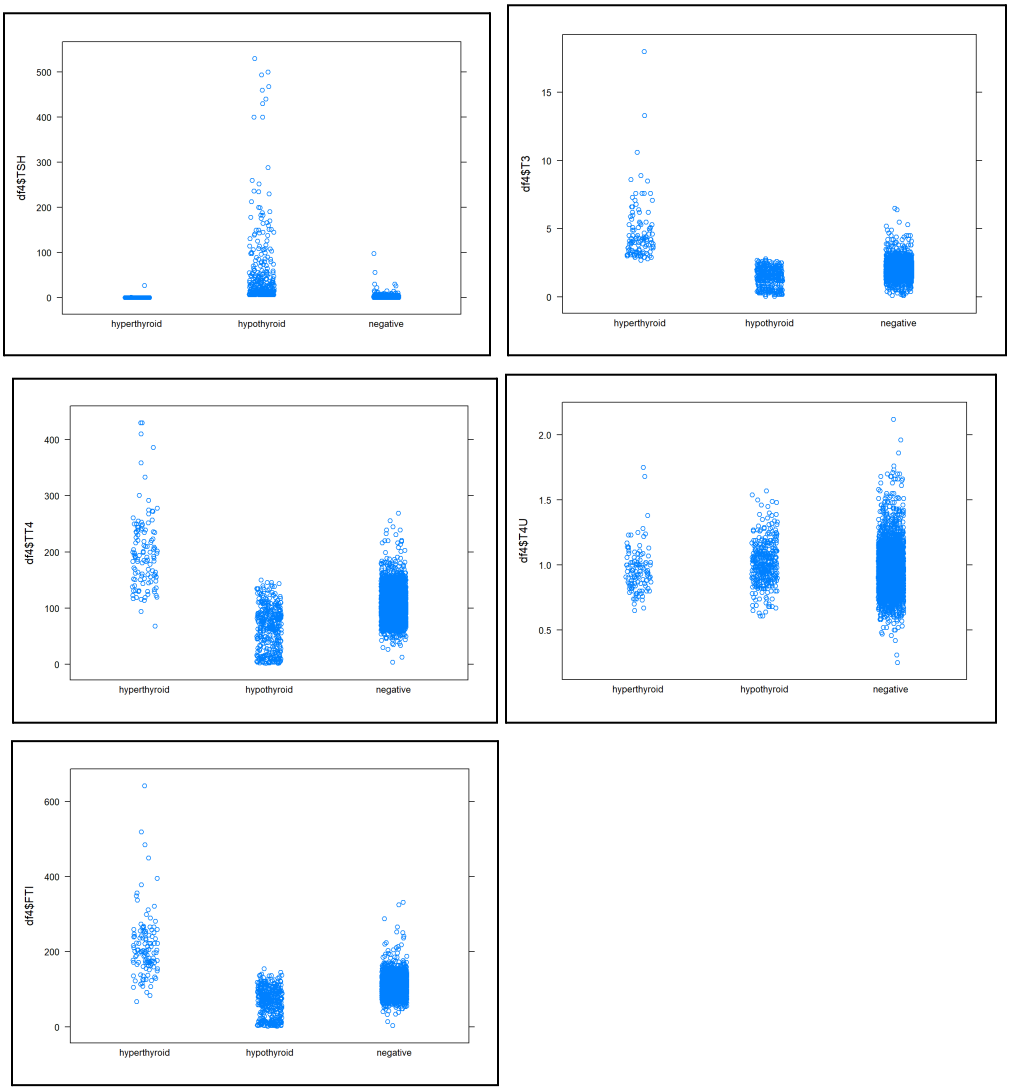
From the above image, It measures the blood level of the hormone T3 (triiodothyronine), some of which are produced directly by the thyroid gland. We can understand that the level is between 1.7 to 2.6. And also from the graph we can also understand that data is distributed normally.



From the above image we see the count of hyperthyroid, hypothyroid, and negatives in the dataset. We can see that most people come under the negative category.

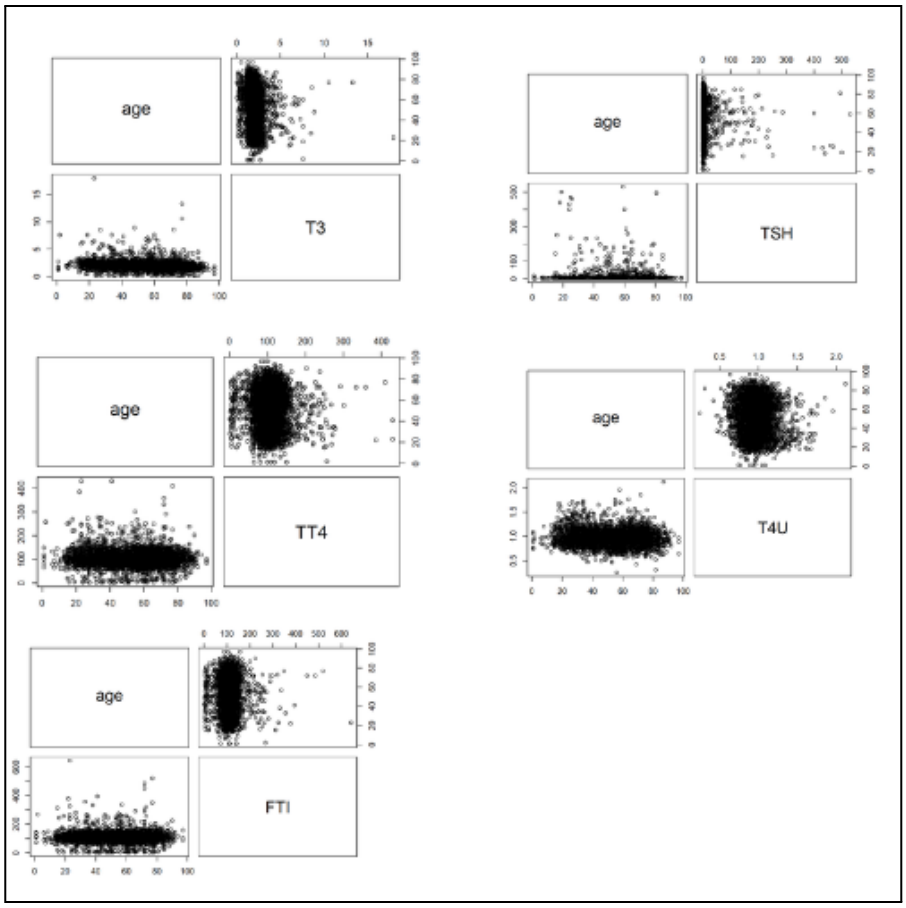


From the above plot, we may deduce that hyperthyroid is impacted around the ages of 25, 50, and 60 and that hypothyroid is affected between the ages of 24 and 80.



As you can see from the photos above, strip plots can help you see how the data is distributed. We must examine each data point that demonstrates the distribution throughout the range of values because we have fewer data.

From the above plots, we can understand the TSH, T3, TT4, T4U, FTI level, and data distributions for the target labels.



The Pair Plot aids in the visualization of both the relationships between two variables as well as the distribution of single variables. They are an excellent way to find patterns between variables for further investigation. These plots are used to determine the most distinct clusters or the best combination of features to describe a correlation between two features.

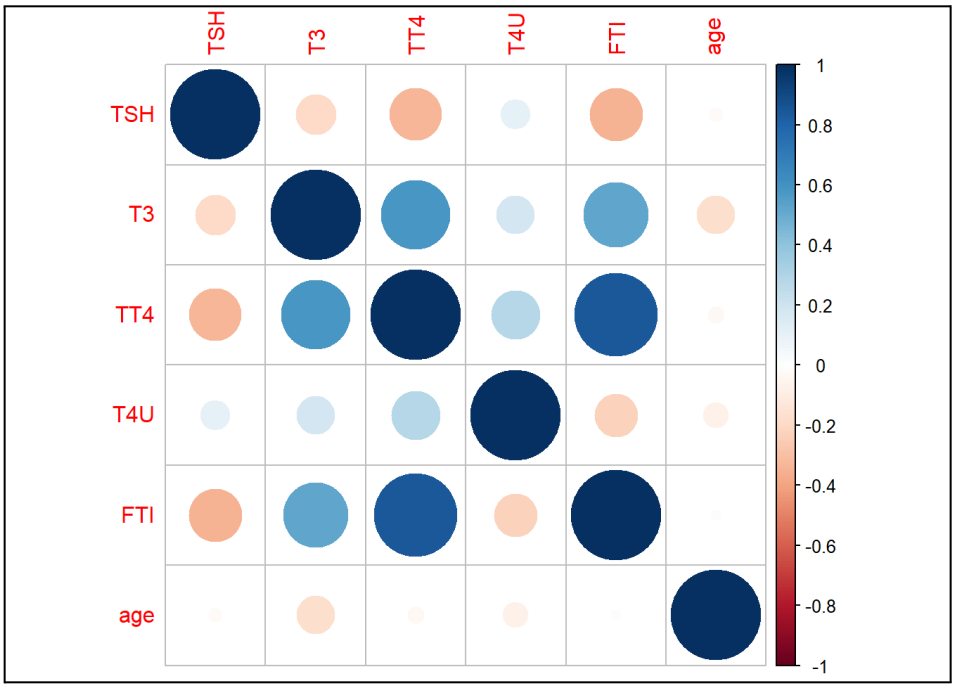
We can see the data distribution with age and T3, TSH, TT4, T4U, and FTI levels in blood samples. From this we understand that age is correlated with T3, TSH, TT4, T4U, and FTI levels in blood.

Analyzing Correlations

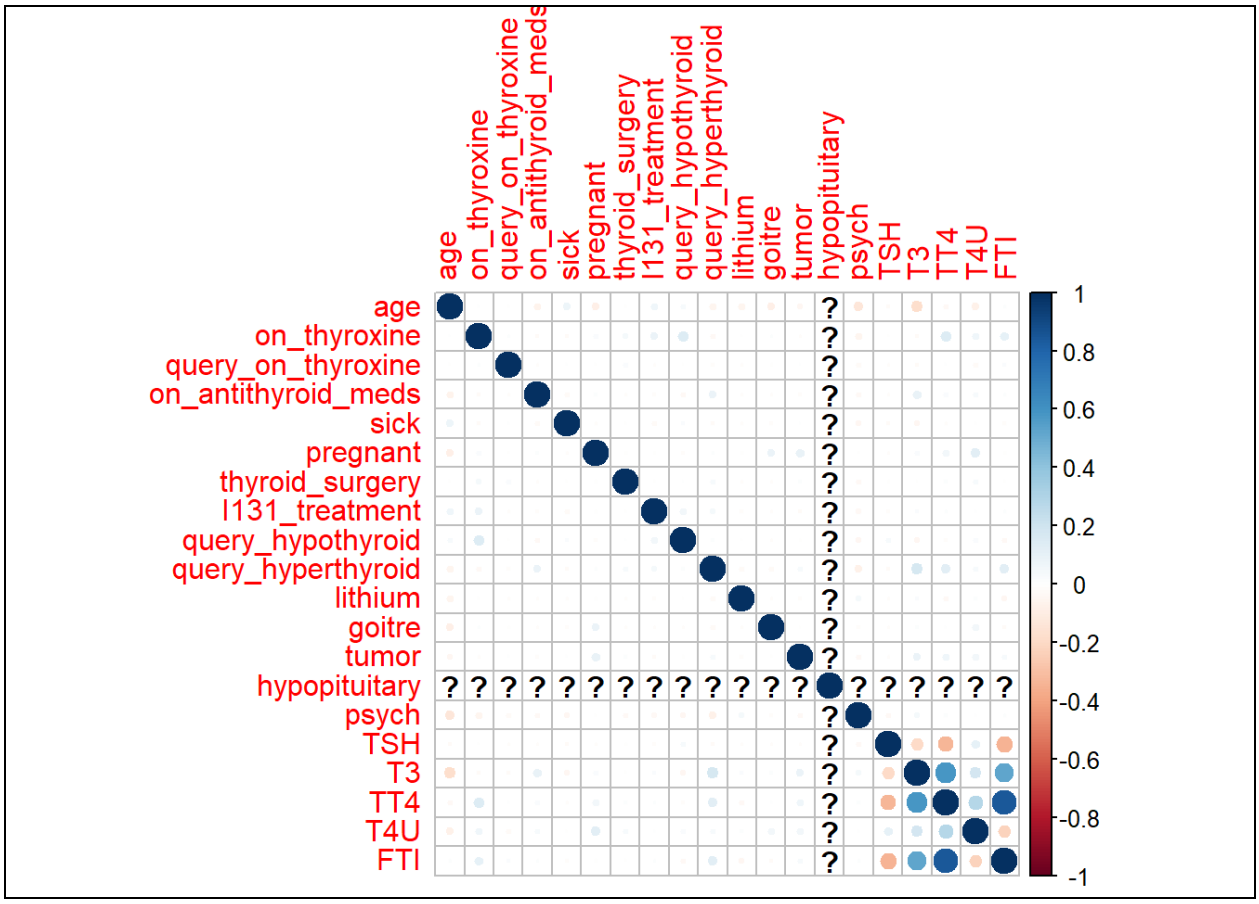
The correlation matrix of the numerical variables is shown below. There are few highly correlated variables as highlighted in the figure.

Then, we can correlate features of age, T3, TSH, TT4, T4U, and FTI levels in blood with the data samples.

From the below plot, we can say that the above feature is strongly correlated so we can use these features for building the model.



And also, From the graph below, we can understand that none of the features are strongly correlated except the above features.



As a result, even after analyzing every single point in the dataset, it is still very obvious that only a subset of the attributes is substantially associated.

5. MODEL TRAINING & MODEL VALIDATION

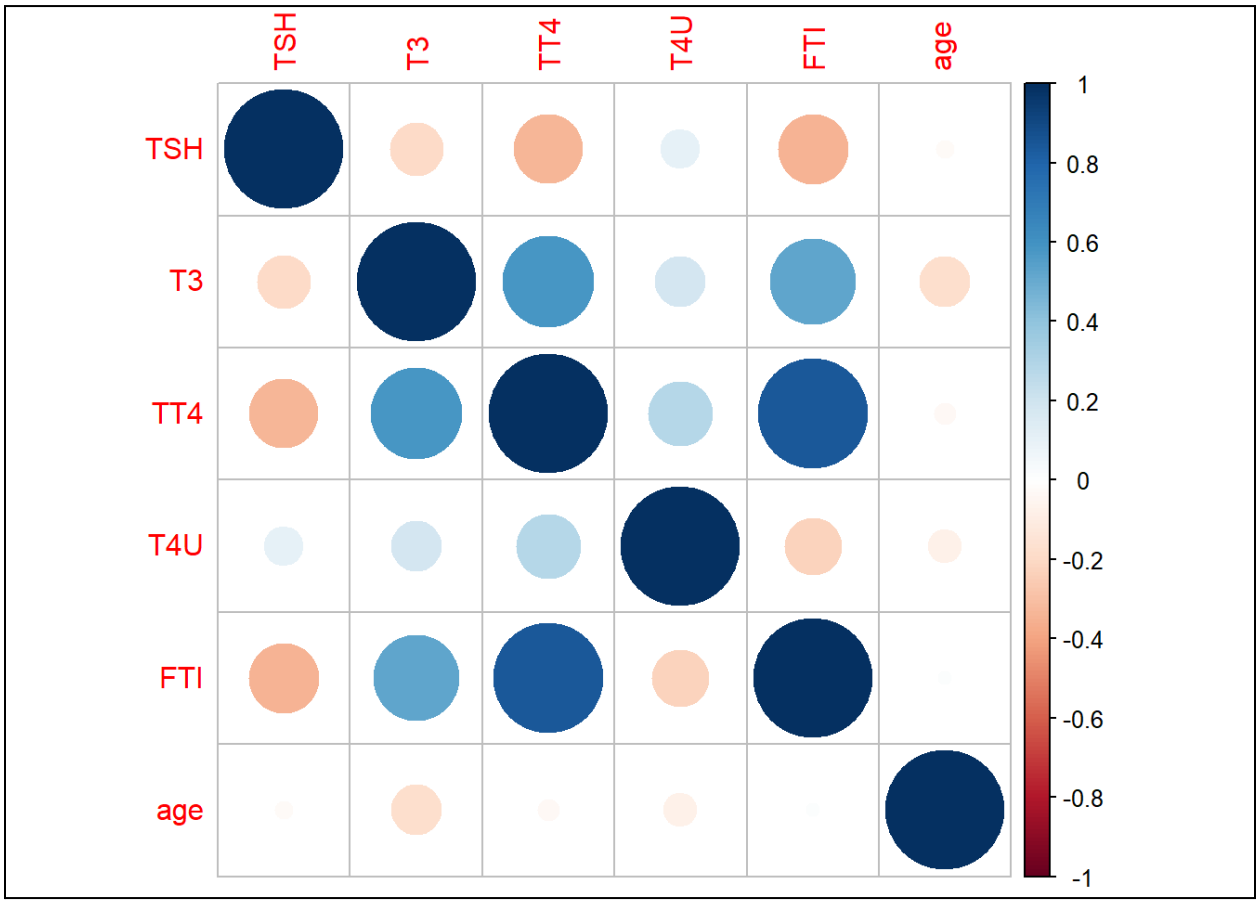
5.1 Correlations :

Features that are correlated:

- TSH
- T3
- TT4
- T4U
- FTI
- Age

Target variables:

- Negative
- Hyperthyroid
- Hypothyroid



Splitting data into train and test

The next step is data splitting and applying algorithms using data that is divided into 80-20 split.Here we have used three libraries to process further,they were e1071,caTools,class where in each and every library have their special features,i.e

E1071:The mentioned package offers functions for probabilistic and statistical techniques like support vector machines, fuzzy classifiers, naive Bayes classifiers, bagged clustering, and short-time Fourier transform, among others. provides SVM implementation that is rapid and simple. provides the majority of common kernels, including sigmoid, linear, and polynomial,

processing capacity for decision-making, probability values for predictions, as well as cross-validation and class weighting in the classification mode.

caTools:Contains several basic utility functions including: moving (rolling, running) window statistic functions, read/write for GIF and ENVI binary files, fast calculation of AUC, LogitBoost classifier, base64 encoder/decoder, round-off-error-free sum.

Class:Class is the blueprint that **helps to create an object and contains its member variable along with the attributes**.There are two classes of R, S3, and S4.

After the data is split into 80-20 i.e 80% for the training set and the rest 20% for testing,3 algorithms such as have been used in total to predict accuracy,such as K-NN classifier, Random forest, Naive Bayes.

KNN classifier

A non-parametric supervised learning technique called the k-nearest neighbor algorithm (k-NN) is utilized for classification and regression. The input in both situations consists of a data set's k closest training samples. Whether k-NN is used for classification or regression will affect the outcome. For k-NN classification, the output is a class membership. By a majority vote of its neighbors, an object is classed, and the object is then given the class that is most popular among its k closest neighbors (k is a positive integer, typically small). The object is simply put into the class of its one nearest neighbor if k = 1. The output of k-NN regression is the object's property value.

Working:KNN finds the distances between a query and each example in the data, chooses the K examples closest to the query, and then, in the case of classification, votes for the label with the highest frequency or averages the labels.

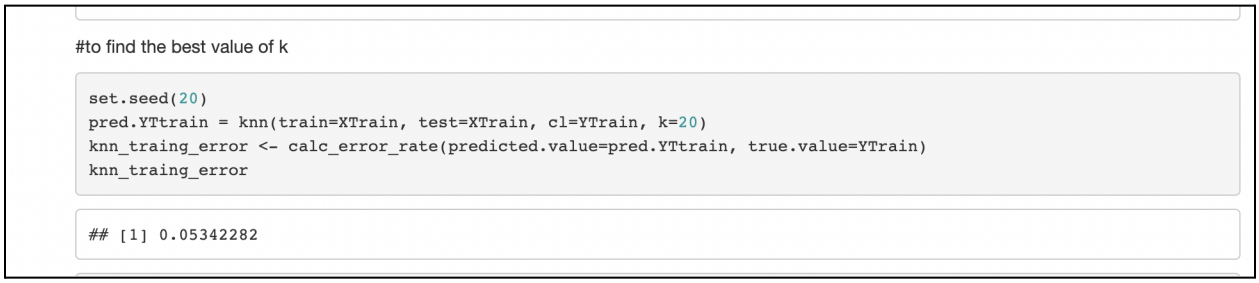
KNN before removal of outliers and scaling features

Firstly an error function has been calculated using predicted value and true value.

set.seed(20) & k=20

Seed function is used for developing code that involves creating variables that take on random values, the set.seed() function is used to produce results that are repeatable. We may ensure that the same random numbers are generated every time the code is run by utilizing the set.seed()

function,and by putting k=20 the knn model is trained with 80% of data and attained 94.7% accuracy.

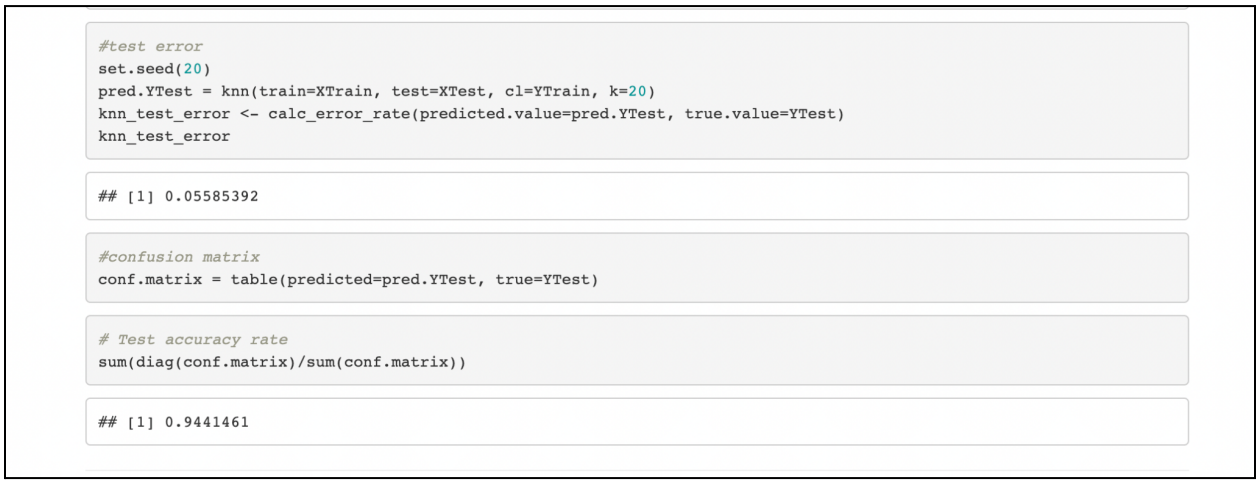


#to find the best value of k

```
set.seed(20)
pred.YTtrain = knn(train=XTrain, test=XTrain, cl=YTrain, k=20)
knn_traing_error <- calc_error_rate(predicted.value=pred.YTtrain, true.value=YTrain)
knn_traing_error
```

```
## [1] 0.05342282
```

Next step we concentrated more on testing data where the k value is 20 and using confusion matrix and 20% of testing data obtained accuracy of 94.4%



```
#test error
set.seed(20)
pred.YTest = knn(train=XTrain, test=XTest, cl=YTrain, k=20)
knn_test_error <- calc_error_rate(predicted.value=pred.YTest, true.value=YTest)
knn_test_error
```

```
## [1] 0.05585392
```

```
#confusion matrix
conf.matrix = table(predicted=pred.YTest, true=YTest)
```

```
# Test accuracy rate
sum(diag(conf.matrix)/sum(conf.matrix))
```

```
## [1] 0.9441461
```

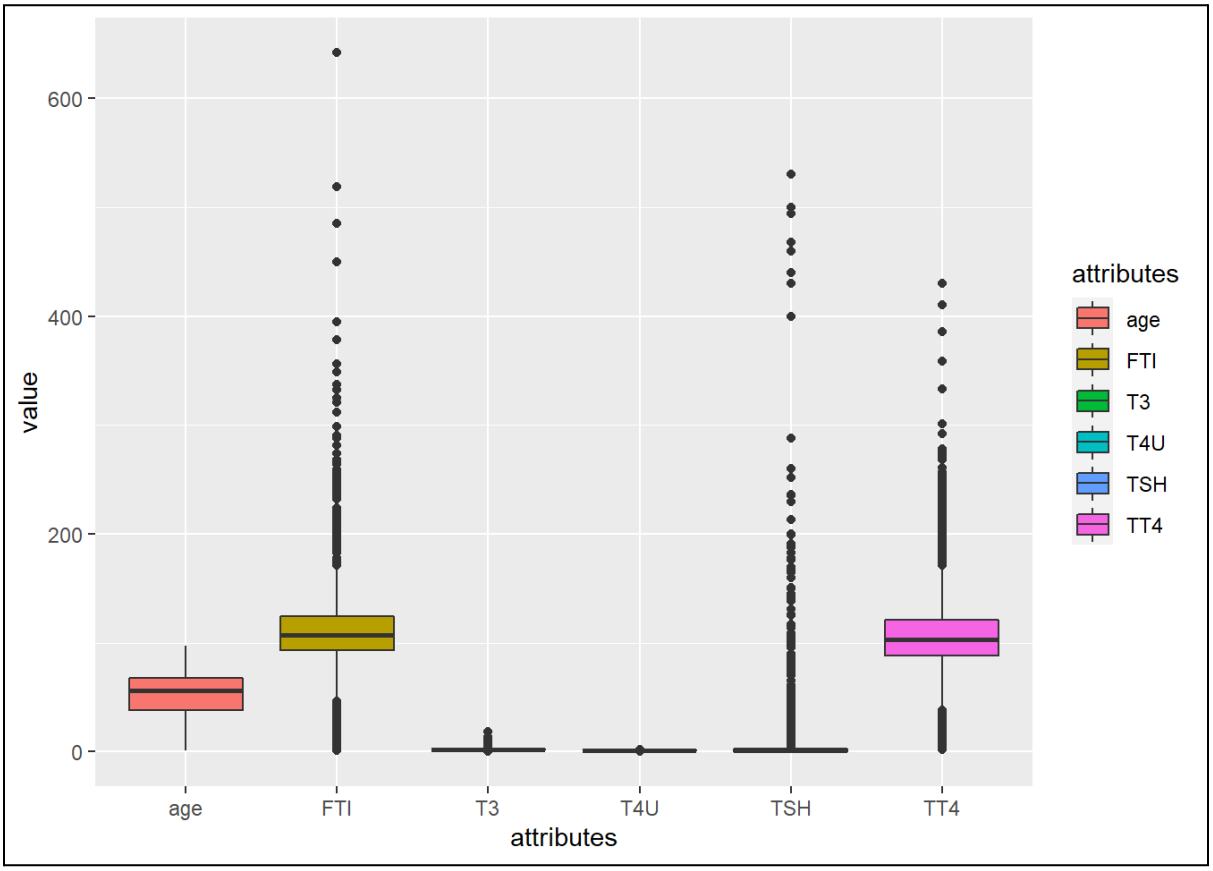
Re-Data preparation using Box plot

Box plots are used to display the distributions of numerical data values, particularly when comparing them across various groups. They are designed to give high-level information at a glance and provide details like the symmetry, skew, variance, and outliers of a set of data.

Working of boxplot is when the distribution is said to be symmetrical when the box's median is in the center and its whiskers are roughly the same on both sides. The distribution is positively

skewed when the median is closer to the bottom of the box and the whisker is shorter on the lower end of the box (skewed right).

An observation that differs abnormally from other values in a population-based random sample is referred to as an outlier. In a way, this definition defers to the analyst's judgment as to what constitutes abnormal behavior is called outliers.Finding and dealing with outliers is one of the most crucial processes in data preparation since they can have a negative impact on statistical analysis and the training of a machine learning algorithm, resulting in decreased accuracy.

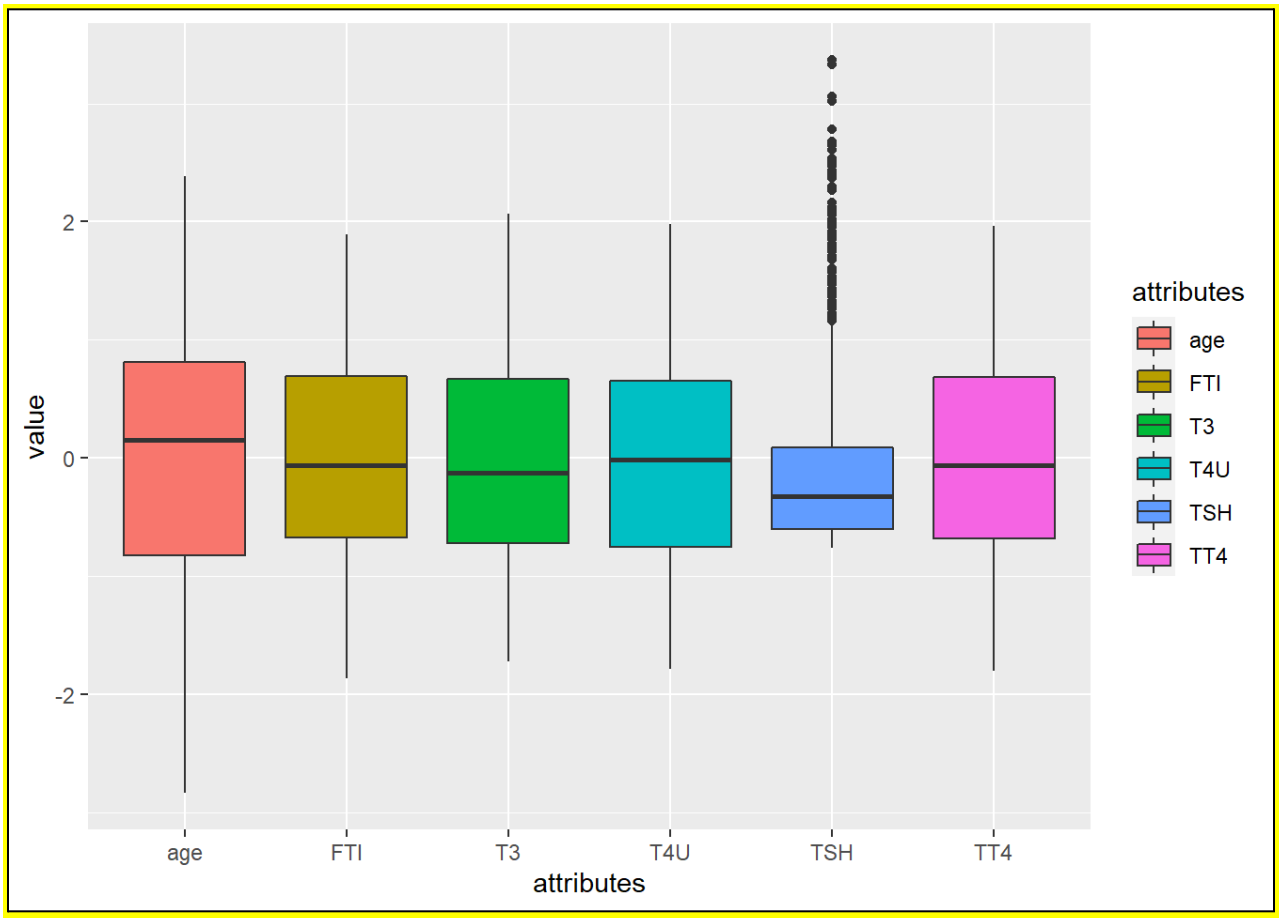


After removing outliers and scaling the features

The above image shows that all the features are not accurate i.e many have outliers and are not accurate. For T3,T4U,TSH the plots were not visible due to outliers to overcome this a squish function is used and scaling is done upon the features to attain proper boxplots and proper accuracy.

Scaling : A method for comparing data that isn't measured the same way is scaling. Scaling is the process of transforming a dataset into a normal form using the mean and standard deviation. Scaling is typically used when working with vectors or columns in a data frame.

Below image shows the data after removing outliers and applying scaling



Re-Applying KNN classifier on scaled data

Taking K value as 20 and redoing the process by on the scaled data i.e using 80% of data for training and 20% for testing and attained an **accuracy of 97%**.Before scaling the **accuracy attained is 94%** and after scaling all the features poses equal level and have highest accuracy which says that the trained model is capable of estimating the hypothyroid and hyper thyroid with **97% accuracy** irrespectively.

#to find the best value of k

```
set.seed(20)
pred.YTtrain = knn(train=XTrain, test=XTrain, cl=YTrain, k=20)
knn_traing_error <- calc_error_rate(predicted.value=pred.YTtrain, true.value=YTrain)
knn_traing_error
```

```
## [1] 0.0295302
```

```
#test error
set.seed(20)
pred.YTest = knn(train=XTrain, test=XTest, cl=YTrain, k=20)
knn_test_error <- calc_error_rate(predicted.value=pred.YTest, true.value=YTest)
knn_test_error
```

```
## [1] 0.02685285
```

```
#confusion matrix
conf.matrix = table(predicted=pred.YTest, true=YTest)
```

```
# Test accuracy rate
sum(diag(conf.matrix)/sum(conf.matrix))
```

```
## [1] 0.9731472
```

Training and Testing model using Random forest

A classification system made up of several decision trees is called the random forest. It attempts to produce an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree by using bagging and feature randomness when generating each individual tree.For a more accurate prediction, Random Forest produces multiple decision trees that are then combined. The Random Forest model is based on the idea that several uncorrelated models (the various decision trees) work significantly better together than they do individually.

Libraries used:

Random forest: In R, a random forest that can be used for both classification and regression was created by an aggregating tree. Its ability to prevent overfitting is among its many advantages. The random forest can handle a lot of features and aids in identifying the crucial characteristics.

Caret: The caret library, which follows a standard syntax for data preparation, model development, and model evaluation, is one of the most effective and well-liked packages. It makes it simple for data science practitioners. Classification and regression training, or Caret, is likely the largest R work.

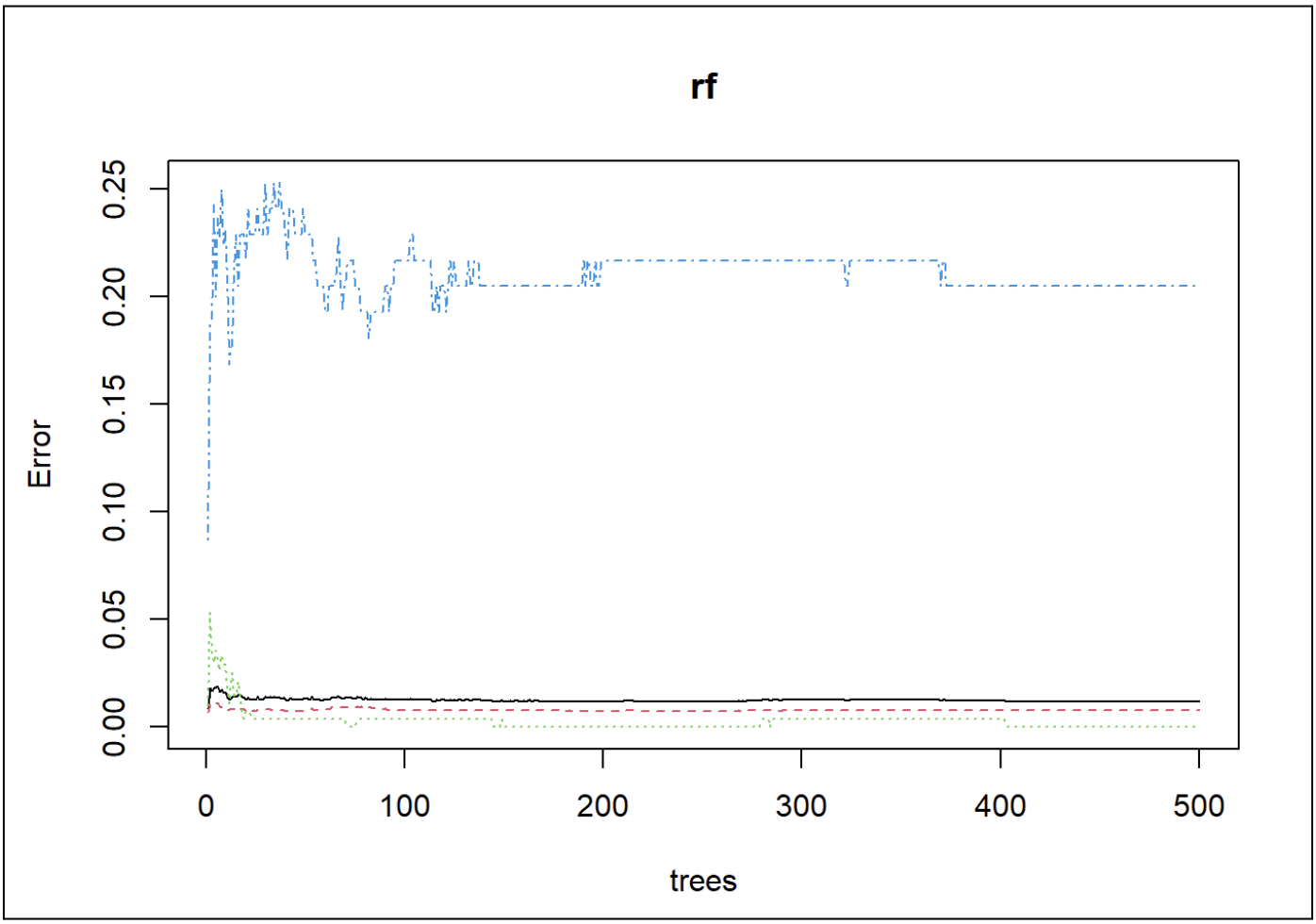
Datasets: In R, a DataSet is defined. A dataset in R is a central area in the package in RStudio where data from different sources are saved, maintained, and made available for usage.

We have used libraries like random forest,caret,datasets to create the model.we imported data and splitted into 70% train and 30% of testing data after that we created the random forest model and trained the raw model using training dataset and plotted the error graph and calculated the training accuracy and finally we attained error as 1.23% and accuracy of training data as approximately 98%.

```
##
## Call:
##  randomForest(formula = target ~ ., data = train, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 1.2%
## Confusion matrix:
##              negative hypothyroid hyperthyroid class.error
## negative         2868           5           17 0.007612457
## hypothyroid         0         280            0 0.000000000
## hyperthyroid       17           0           66 0.204819277
```

Out of the bag error rate is 1.23% so the accuracy of the training data is approximately 98%.

As per above the model contains 500 trees and 2 splits,so the below graph indicates the error of each tree in the model.



Graph showing Error Vs Trees

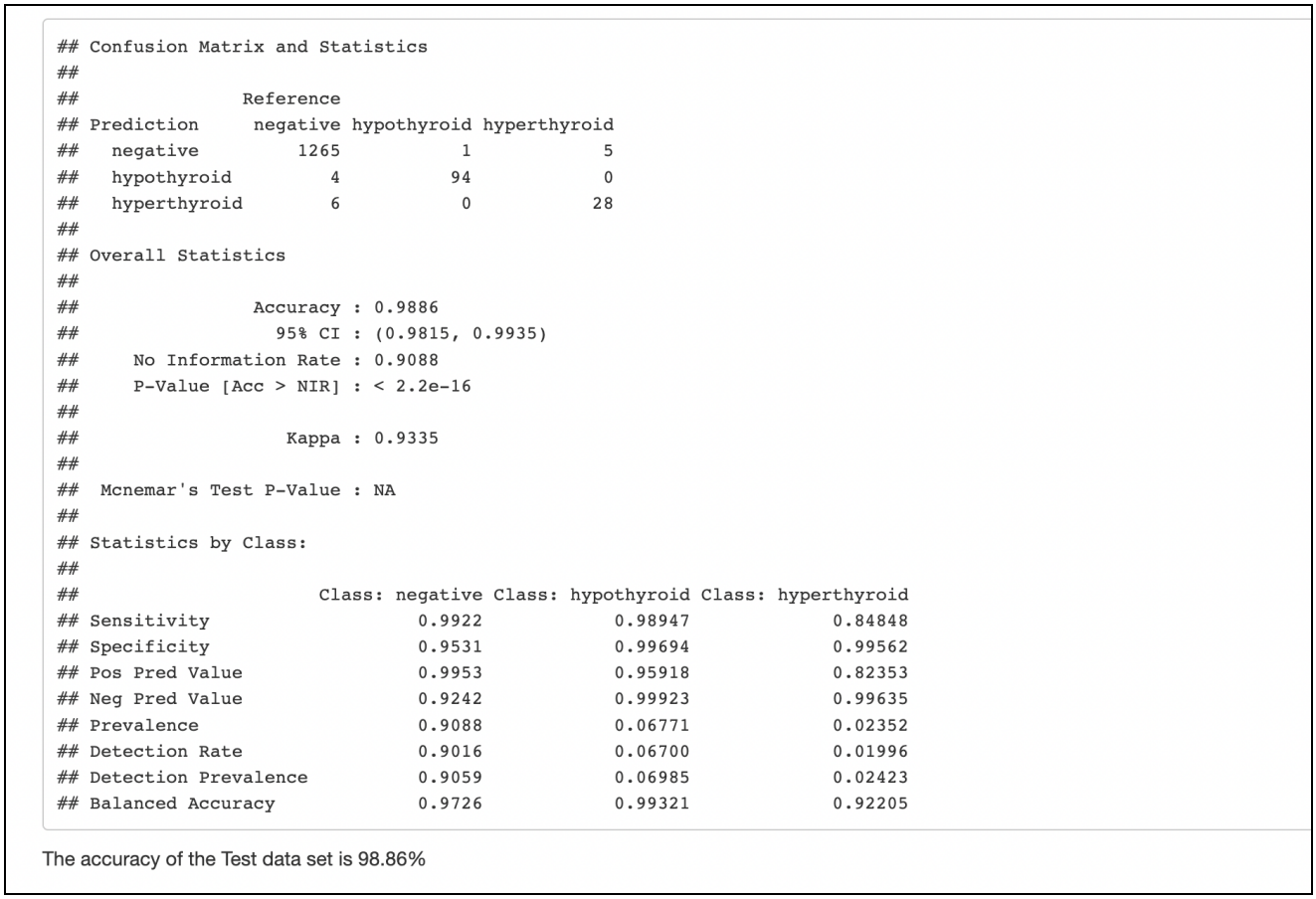
Model Evaluation:

Firstly the trained model is tested with the trained data and achieved an **accuracy of 99.78%** with the trained data and observed result using a confusion matrix.

```
## Confusion Matrix and Statistics
##
##               Reference
## Prediction     negative hypothyroid hyperthyroid
##   negative         2883           0            0
##   hypothyroid         0         280            0
##   hyperthyroid        7           0           83
##
## Overall Statistics
##
##                Accuracy : 0.9978
##                  95% CI : (0.9956, 0.9991)
##     No Information Rate : 0.8884
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9895
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: negative Class: hypothyroid Class: hyperthyroid
## Sensitivity                   0.9976            1.00000             1.00000
## Specificity                   1.0000            1.00000             0.99779
## Pos Pred Value                1.0000            1.00000             0.92222
## Neg Pred Value                0.9811            1.00000             1.00000
## Prevalence                    0.8884            0.08607             0.02551
## Detection Rate                0.8863            0.08607             0.02551
## Detection Prevalence          0.8863            0.08607             0.02767
## Balanced Accuracy             0.9988            1.00000             0.99890
```

Image displaying the accuracy and confusion matrix

Firstly the trained model is tested with the testing data and achieved an **accuracy of 98.86%** with the tested data and observed result using a confusion matrix.



```
## Confusion Matrix and Statistics
##
##               Reference
## Prediction     negative hypothyroid hyperthyroid
##   negative         1265           1             5
##   hypothyroid         4          94             0
##   hyperthyroid        6           0            28
##
## Overall Statistics
##
##                Accuracy : 0.9886
##                  95% CI : (0.9815, 0.9935)
##     No Information Rate : 0.9088
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9335
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: negative Class: hypothyroid Class: hyperthyroid
## Sensitivity                   0.9922            0.98947             0.84848
## Specificity                   0.9531            0.99694             0.99562
## Pos Pred Value                0.9953            0.95918             0.82353
## Neg Pred Value                0.9242            0.99923             0.99635
## Prevalence                    0.9088            0.06771             0.02352
## Detection Rate                0.9016            0.06700             0.01996
## Detection Prevalence          0.9059            0.06985             0.02423
## Balanced Accuracy             0.9726            0.99321             0.92205
```


Image displaying the accuracy and confusion matrix

Training and Testing model using Naive Bayes

The Naive Bayes Classifier is one of the most straightforward and efficient classification algorithms widely available. It aids in the development of quick machine learning models capable of making accurate predictions. It is a probabilistic classifier, which means that it makes predictions based on the probability that an object will occur.The Naive Bayes classifier operates according to the Bayes theorem's definition of conditional probability. Probability is typically represented by the letter P when it comes to probability calculations. Following are some scenarios where this would be probable: The likelihood of receiving two heads is 1/4.

Libraries used:

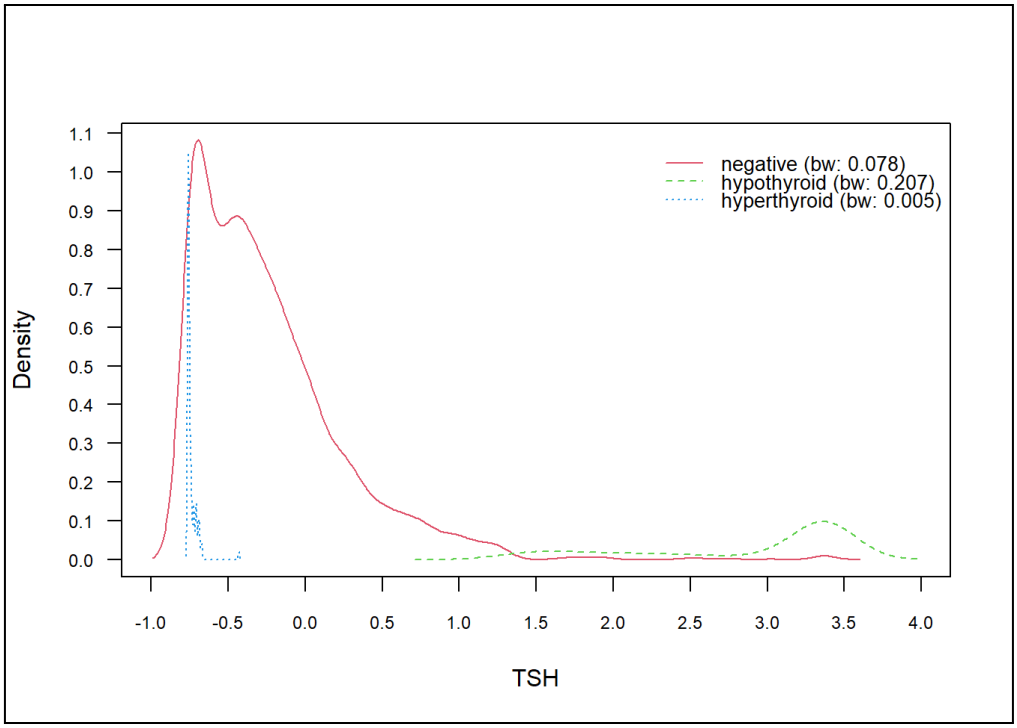
Dplyr: Dplyr is a grammar for data manipulation that offers a consistent collection of verbs to assist you in resolving the most typical problems with data manipulation: Variables that are functions of existing variables are added with the method modify(). Variables are chosen by choose() depending on their names. Based on their values, filter() selects cases.

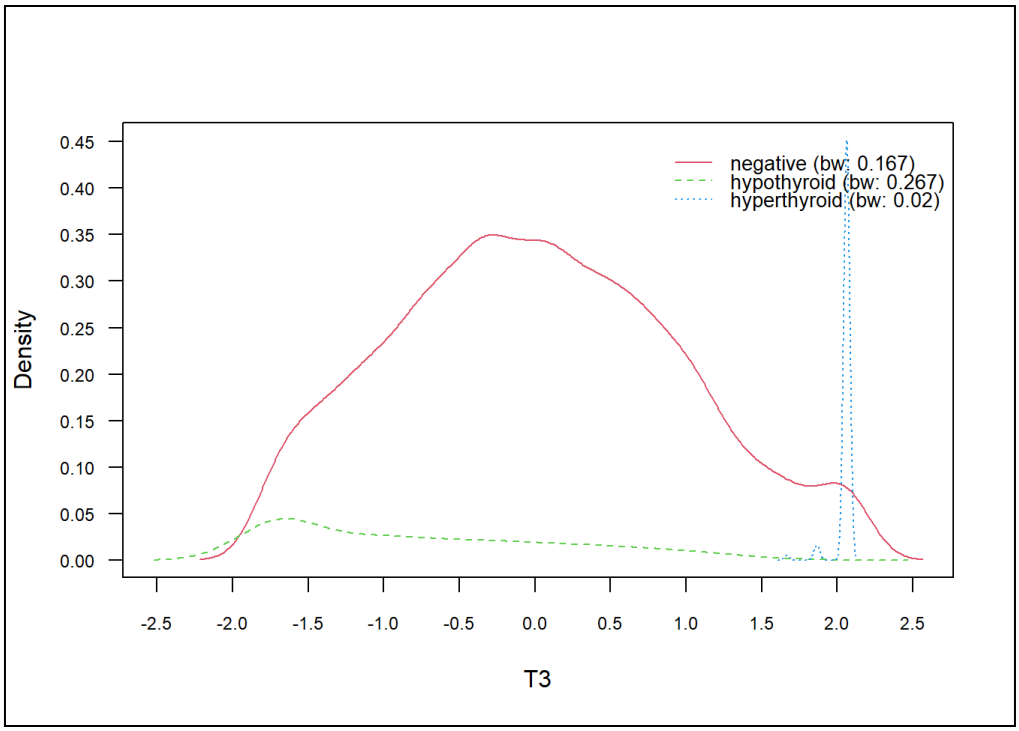
Ggplot2:The tidyverse collection contains a package called ggplot2 whose main purpose is to produce visualizations. It is a well-known R library built around the idea of layered graphics grammar.

Psych: A set of all-purpose tools for experimental psychology, psychometric theory, and personality. Although several of the functions also offer basic descriptive statistics, they are mostly used for multivariate analysis and scale design using factor analysis, principal component analysis, cluster analysis, and reliability analysis.

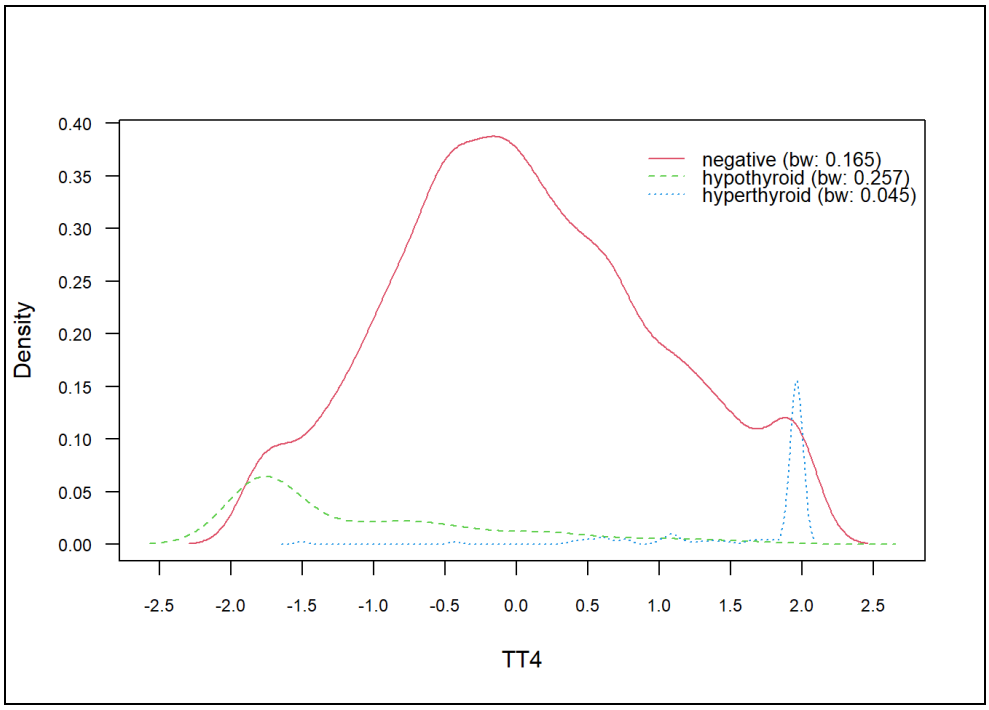
We have used dplyr, ggplot2, psych libraries to create Naive Bayes model.The data imported is splitted into 80% training and 20% testing and now we created the raw model and trained this model using the training data and predicted the density of each data point,after the model trained we calculated the accuracy of training and testing data.

Graphs of data points with respect to density

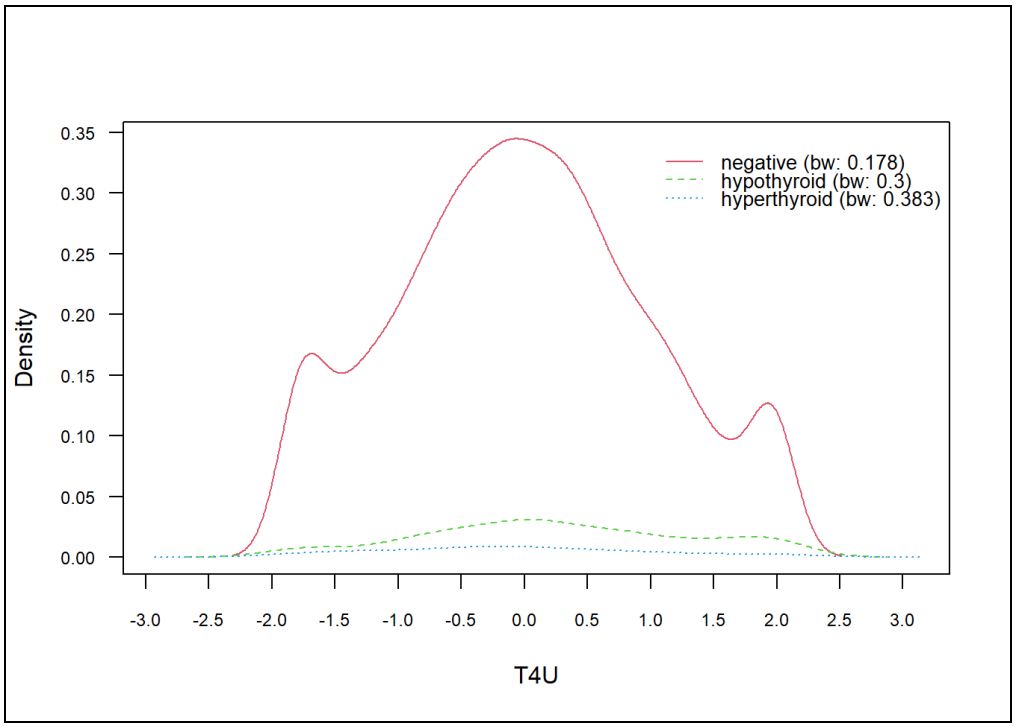




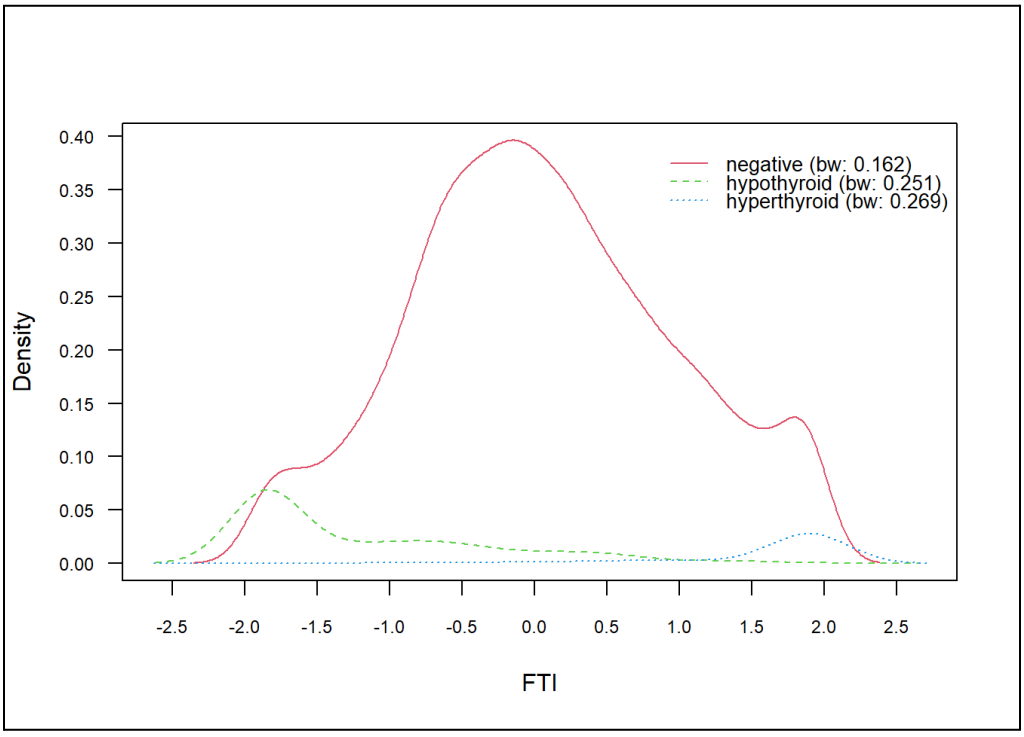
T3 vs Density



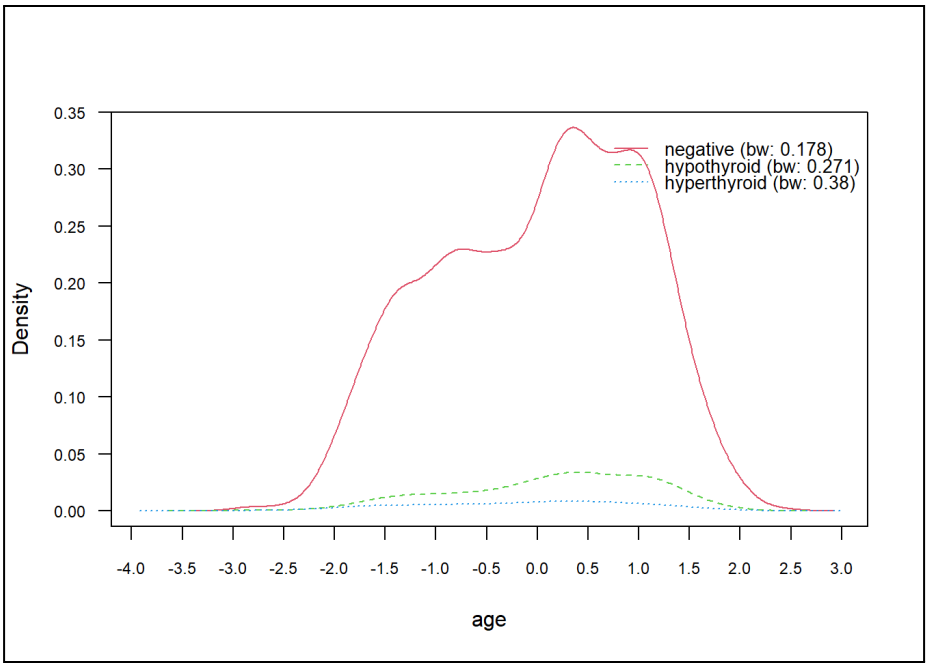
TT4 vs Density



T4U vs Density



FTI vs Density



Age vs Density

Accuracy attained for training and testing models

1. Trained data

```
p1 <- predict(model, train)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
(tab1 <- table(p1, train$target))
```

```
##
## p1             negative hypothyroid hyperthyroid
##   negative         3242          38            1
##   hypothyroid        18         263            0
##   hyperthyroid       62           0           88
```

```
1 - sum(diag(tab1)) / sum(tab1)
```

```
## [1] 0.03205819
```

The accuracy rate of training data is **96.8%**

2. Testing data

```
p2 <- predict(model, test)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```
(tab2 <- table(p2, test$target))
```

```
##
## p2             negative hypothyroid hyperthyroid
##   negative          825          10            0
##   hypothyroid         3          64            0
##   hyperthyroid       15           0           27
```

```
1 - sum(diag(tab2)) / sum(tab2)
```

```
## [1] 0.02966102
```

The accuracy rate of the testing data is **97.1%**

Training and Testing model using Multinomial Regression model

Regression is a method for determining how independent features or variables relate to a dependent feature or result. It is a technique for machine learning predictive modeling, where an algorithm is used to forecast continuous outcomes.

Plotting a line of best fit through the data points is a common step in machine learning regression. To find the line that fits each point the best, the distance between each point and the line is minimized. Alongside classification, regression is one of the main applications of the supervised type of machine learning

Library used:

Caret: The caret library, which uses a consistent syntax for data preparation, model development, and model evaluation, is one of the most potent and well-liked tools.

nnet(package):Feed-Forward Neural Networks and Multinomial Log-Linear Models. Software for feed-forward neural networks with a single hidden layer, and for multinomial log-linear models.

We have used caret library & nnet package, to create a regression model.The data imported is splitted into 80% training and 20% testing and now we created the raw model and trained this model using data provided,after the model trained we calculated the accuracy of training and testing data.

Accuracy attained for training and testing models

1. Trained data

```
# Predicting the values for train dataset
train$ClassPredicted <- predict(multinom_model, newdata = train)

# Building classification table
tab <- table(train$target, train$ClassPredicted)

# Calculating accuracy - sum of diagonal elements divided by total obs
round((sum(diag(tab))/sum(tab))*100,2)
```

```
## [1] 97.13
```

2. Testing data

```
# Predicting the class for test dataset
test$ClassPredicted <- predict(multinom_model, newdata = test)

# Building classification table
tab <- table(test$target, test$ClassPredicted)
tab
```

```
##
##                negative hypothyroid hyperthyroid
##   negative          823           3            7
##   hypothyroid         9          66            0
##   hyperthyroid        3           0           20
```

```
round((sum(diag(tab))/sum(tab))*100,2)
```

```
## [1] 97.64
```

Comparing Between the Model's Training Accuracy and Testing Accuracies

Model	Training Accuracy	Testing Accuracy
K-NN before scaling	95.66%	94.44%
K-NN after Scaling	98.46%	97.31
Random Forest	99.78%.	98.86%
Naive Bayes	96.8%	97.1%
Multinomial Regression Model	97.13%	97.64%.

Conclusion And Future Work:

We can basically conclude that the preparation of the data is a very important factor as it helps in improving the overall accuracy of the model. This can be pretty much explained based on the K-NN accuracy before box-plot preparation and K-NN accuracy after the Box-plot preparation. It is clear that the change in overall accuracy of the model is significant and definitely needs a great attention to work on during any project. Such difference before preparation and post preparation would be common for any classifier and not only KNN and hence we can state that proper choosing of the features and processing them well leads to greater accuracies of the models.

Based on our project, we can say that the overall accuracy of the random forest is the highest in both the training and the testing phase and since the target classes is only 3 and not many so such sort of algorithms work well and also While the trees are developing, the random forest adds more randomness to the model. When splitting a node, it looks for the best feature from a random subset of features rather than the most crucial feature.

Finally, the future scope of this project would be to test the data on a neural network classifier to achieve much better accuracy as it works differently and also a trending one.

Also, we can try and remap the target classes to binary classes (thyroid yes or no) rather than 3 classes so that we can use the AIC,BIC and other techniques which help in attaining better

characteristics of the models and also we could try and implement the different approaches to identify the features (like best subset selection or stepwise feature selection) rather than using the correlation plots or heat maps to understand all the features and choose among them.

Data Sources:

The link below will give the access to dataset we had used in this project

https://www.kaggle.com/datasets/emmanuelfwerr/thyroid-disease-data

Source Code:

The link below will provide access to the github, where all the code files and data files have been uploaded :

https://github.com/Rahul-Machiraju/thyroid-disease-prediction-CSP571--Final-Project

References Citations:

Ken Tang and Ponnuthurai N. Suganthan and Xi Yao and A. Kai Qin. Linear dimensionality reduction using relevance weighted LDA. School of Electrical and Electronic Engineering Nanyang Technological University. 2005.

Zhi-Hua Zhou and Yuan Jiang. NeC4.5: Neural Ensemble Based C4.5. IEEE Trans. Knowl. Data Eng, 16. 2004.

Xiaoyong Chai and Li Deng and Qiang Yang and Charles X. Ling. Test-Cost Sensitive Naive Bayes Classification. ICDM. 2004.

Vassilis Athitsos and Stan Sclaroff. Boosting Nearest Neighbor Classifiers for Multiclass Recognition. Boston University Computer Science Tech. Report No, 2004-006. 2004.

Michael L. Raymer and Travis E. Doom and Leslie A. Kuhn and William F. Punch. Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 33. 2003.

Lukasz A. Kurgan and Waldemar Swiercz and Krzysztof J. Cios. Semantic Mapping of XML Tags Using Inductive Machine Learning. ICMLA. 2002.

Qiang Yang and Jing Wu. Enhancing the Effectiveness of Interactive Case-Based Reasoning with Clustering and Decision Forests. Appl. Intell, 14. 2001.

Erin L. Allwein and Robert E. Schapire and Yoram Singer. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. ICML. 2000.