Liver cirrhosis Classification

Project work

PONDICHERRY UNIVERSITY

MASTER OF SCIENCE
IN
STATISTICS

By
Mahesh Madan

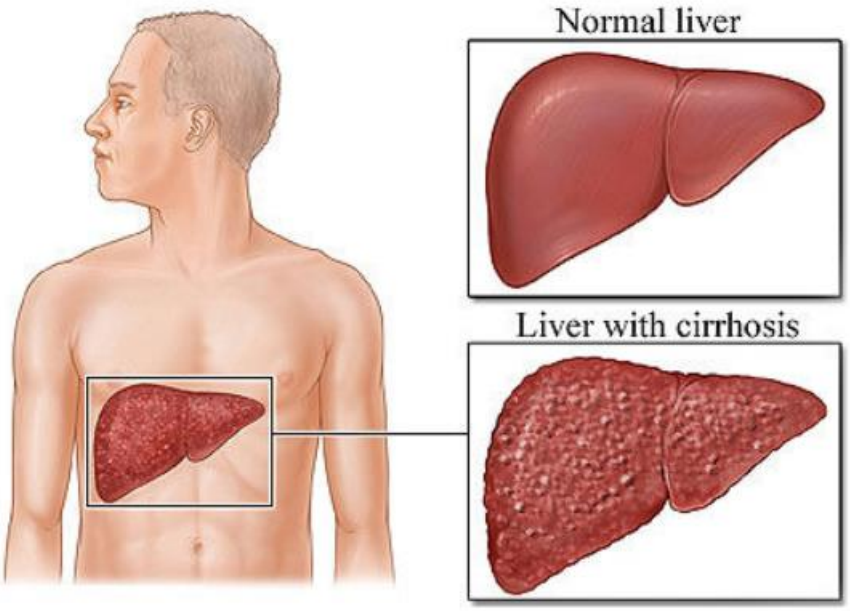
RAMANUJAN SCHOOL OF MATHEMATICS SCIENCES
DEPARTMENT OF STATISTICS
PONDICHERRY UNIVERSITY
R.V NAGAR, KALAPET
PONDICHERRY -605014

Introduction

Liver cirrhosis is a serious health issue globally, with increasing rates attributed to factors like alcohol consumption, chronic hepatitis infections, and obesity-related liver disease. Despite its high mortality, liver diseases affect different sub-populations unequally. Early detection is crucial for better patient outcomes, yet female patients seem to face challenges in early diagnosis.

This project focuses on predicting liver disease in the North East region of Andhra Pradesh, India, where liver cirrhosis rates are rising. The dataset comprises 584 patient records, including information on biochemical markers such as albumin and other metabolism-related enzymes. By analyzing this data, we aim to improve early detection and treatment outcomes for liver diseases in this region.

The primary objective of this project is to develop a predictive model that can accurately identify patients suffering from liver disease based on their biochemical markers. This model can assist healthcare professionals in diagnosing liver diseases early, especially among populations that may be marginalized or underserved. This model can potentially improve early diagnosis and treatment outcomes for liver diseases in the North East region of Andhra Pradesh, India



Attribute information -:

FEATURE NAME	DESCRIPTION
AGE	Age of the patient
GENDER	Gender of the patient
TOT BILIRUBIN	Total bilirubin is a yellowish pigment produced during the normal breakdown of red blood cells. It is formed when the heme group of hemoglobin is metabolized. Elevated levels of total bilirubin in the blood can indicate liver disease or other medical conditions.
DIRECT BILIRUBIN	Direct Bilirubin is the form of bilirubin that has been processed by the liver and is conjugated with glucuronic acid, making it water-soluble and able to be excreted in bile. Elevated levels of direct bilirubin in the blood can indicate liver or bile duct issues.
ALKPHOS	Alkphos, or alkaline phosphatase, is an enzyme found in various tissues throughout the body, with particularly high concentrations in the liver, bones, kidneys, and digestive system. Elevated levels of alkaline phosphatase can indicate liver disease, bone disorders, or bile duct obstructions.
SGPT	SGPT, or serum glutamic pyruvic transaminase, is an enzyme found primarily in the liver. It is released into the bloodstream when liver cells are damaged. Elevated levels of SGPT in the blood can indicate liver damage
SGOT	SGOT, or serum glutamic oxaloacetic transaminase, is an enzyme found in various tissues, with particularly high concentrations in the liver, heart, and skeletal muscle. Elevated levels of SGOT in the blood can indicate liver damage or disease.
TOT PROTEINS	Total proteins refer to the total amount of proteins in the blood, including albumin and globulin. These proteins play various roles in the body, such as transporting substances, maintaining fluid balance, and supporting immune function. Changes in total protein levels can be seen in conditions such as liver or kidney disease.
ALBUMIN	Albumin is a protein produced by the liver that helps maintain the oncotic pressure of blood, which is essential for maintaining proper fluid balance between the blood vessels and tissues. Normal albumin levels typically range from 3.5 to 5.0 grams per deciliter (g/dL) of blood. Abnormal albumin levels can indicate various health conditions, such as liver disease
AG_RATIO	The A/G ratio, or albumin/globulin ratio, is a calculation based on the levels of albumin and globulin in the blood. A normal A/G ratio is usually around 1.0 to 2.2, but the specific range can vary depending on the laboratory and the individual's age and gender. Changes in the A/G ratio can indicate certain health conditions, such as liver disease
SELECTOR	Selector variable indicates whether a patient has been diagnosed liver disease or not.

Procedure: -

1. Data Collection: Data was collected from Kaggle.
2. Data Preprocessing: The dataset underwent preprocessing steps, including handling missing values, and treating the unbalanced classes.
3. Exploratory Data Analysis: A thorough analysis of the dataset was conducted to understand the distribution of variables and identify any patterns or trends.
4. Model Development: Various machine learning models, such as logistic regression, SVM, decision trees, and random forest were trained on the dataset to predict liver disease status.
5. Model Evaluation: The performance of the models was evaluated using metrics like accuracy, precision, recall and F1-score.
6. Expected Outcome: By the end of this project, we expect to have a predictive model that can accurately identify patients with liver disease based on their biochemical markers.

Statistical tools used: -

- Logistic regression
- SVM
- Decision tree
- Random Forest
- Hyperparameter Tuning

Statistical software used: -

- Advance Excel
- R
- Python

Data preprocessing -:

The dataset comprises records from 583 patients, including 416 diagnosed with liver disease and 167 without. Each patient's information includes age, gender, total bilirubin, direct bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT, and Alkphos levels. There are 441 male patients and 142 female patients. Initially, the dataset suffered from class imbalance, which was addressed using the Random Over-Sampling Examples (ROSE) method. This method helped balance the classes for more effective binary classification.
Upon checking for missing values, only the A/G ratio variable showed 4 missing values, which were imputed with the median since the distribution was skewed. Subsequent visualization and analysis revealed the presence of outliers in the data. However, due to the nature of medical data, removing these outliers was deemed inappropriate as they could represent critical and valid observations.
The decision not to remove outliers was based on the following considerations:
Clinical relevance: Outliers in medical data might signify extreme physiological responses or rare conditions, which are crucial for diagnosis and treatment.
Diagnostic value: Outliers could indicate underlying health issues or anomalies, providing valuable insights for diagnosis. Removing them might obscure these patterns and lead to inaccurate conclusions.

After preprocessing, various machine learning algorithms were applied to the data, and the best-performing algorithm was selected based on the analysis of results.

Data Visualization

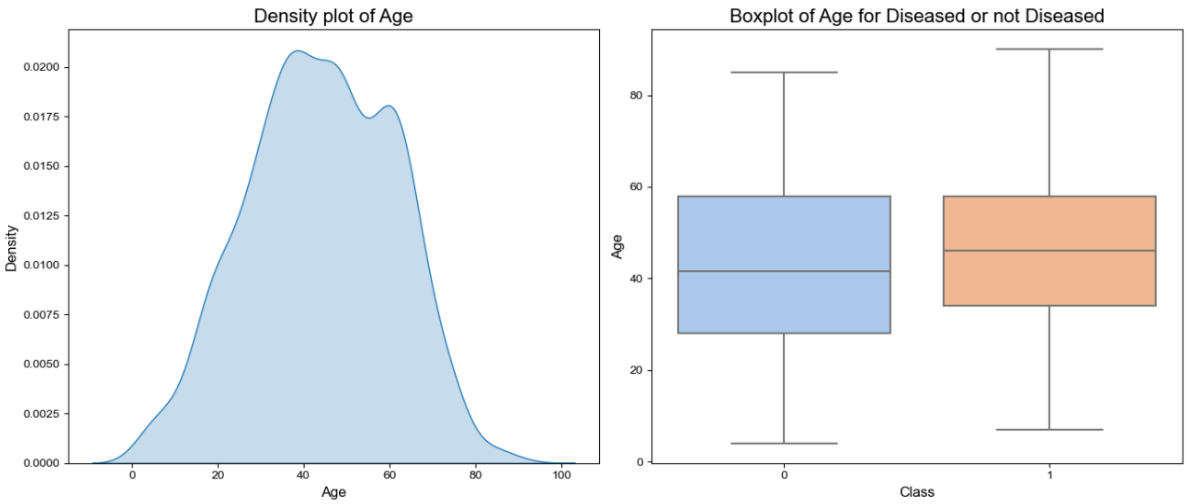
This plot reveals a graphical representation of the relationships between pairs of variables in a dataset. It provides a comprehensive view of how variables relate to each other, revealing patterns such as linear or non-linear relationships, clustering of points, and outliers. The diagonal of the pair plot shows histograms (or kernel density estimates) of each variable, allowing us to understand the distribution of each variable individually. The scatterplots provide a visual indication of the correlation between variables. Highly correlated variables will show a clear pattern in the scatterplots, while uncorrelated variables will show a more scattered plot

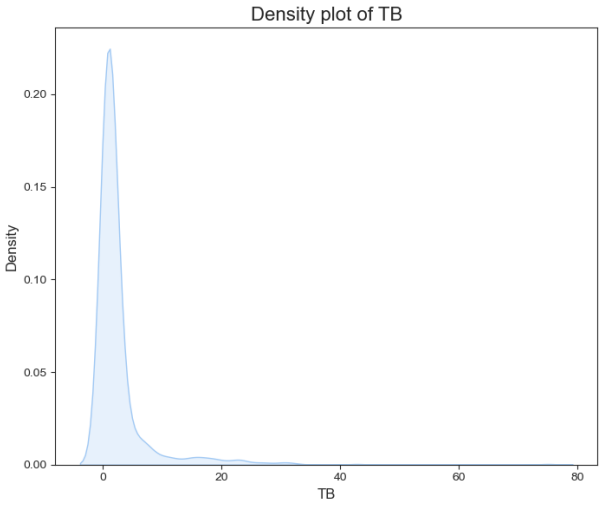
Box plot and Density plot: -

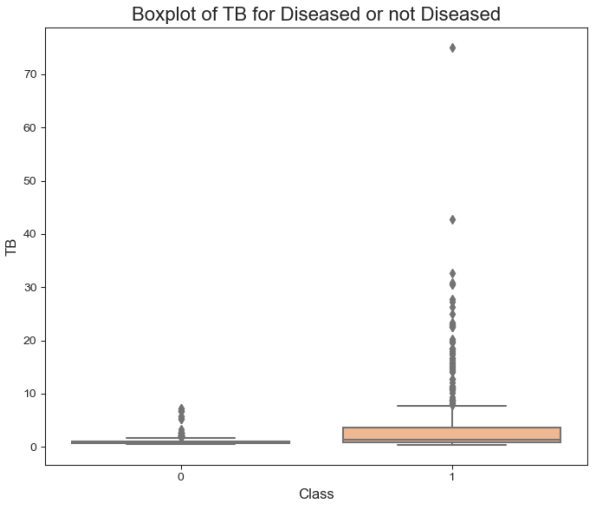
Box plots show the median and the spread of the data (interquartile range, IQR, represented by the box's height). The whiskers extend to the smallest and largest observations within 1.5 times the IQR from the first and third quartiles, respectively, it also reveals outliers, which are data points that fall outside the whiskers. These points are displayed as individual points beyond the whiskers.

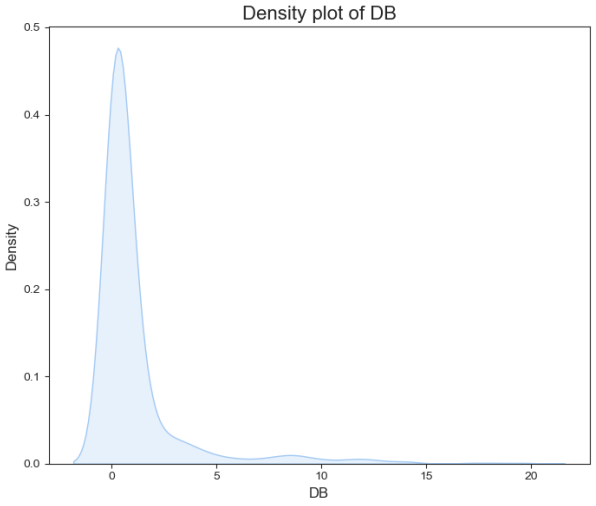
Density plots show the distribution of the data along the range of values. They are useful for visualizing the shape of the distribution, including whether it is symmetric, skewed, or multimodal. Areas under the curve represent probabilities, with the total area under the curve being. Box plots are more focused on summarizing key statistics and identifying outliers, while density plots provide a more detailed view of the data distribution.

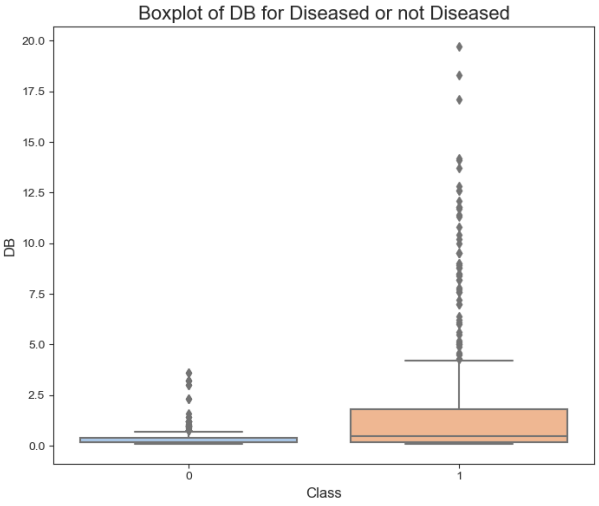
The below plots give a detailed understanding of how different variables are distributed among different classes.

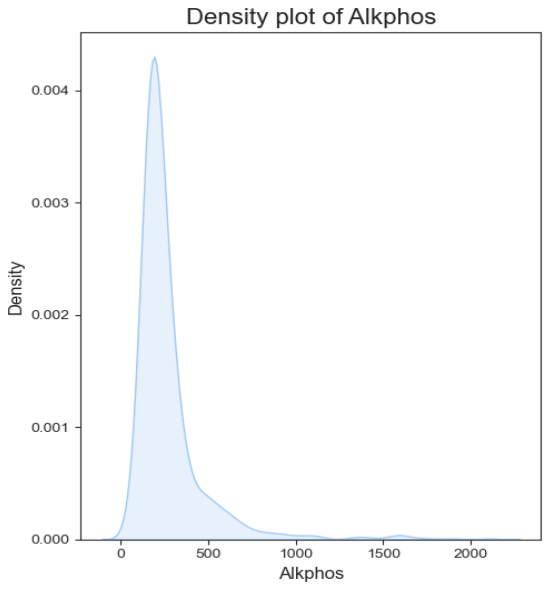


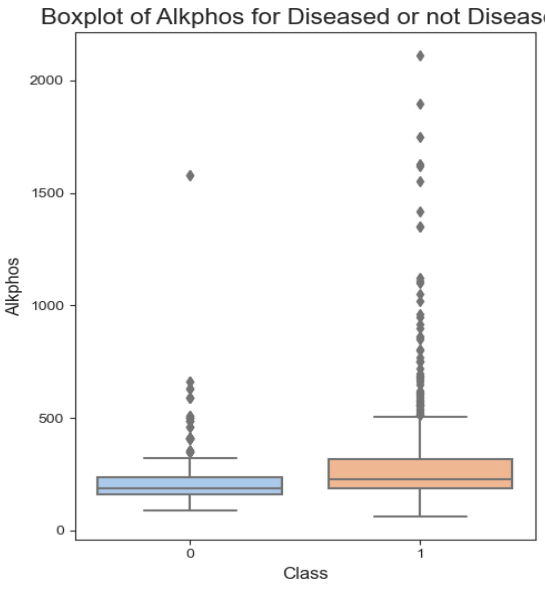

Density plot of TB
Density
TB
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80

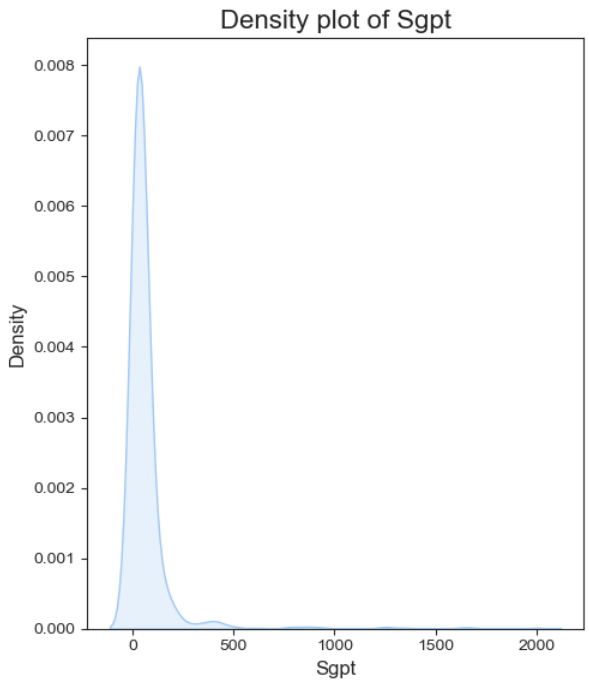

Boxplot of TB for Diseased or not Diseased
TB
Class
0
10
20
30
40
50
60
70
0
1

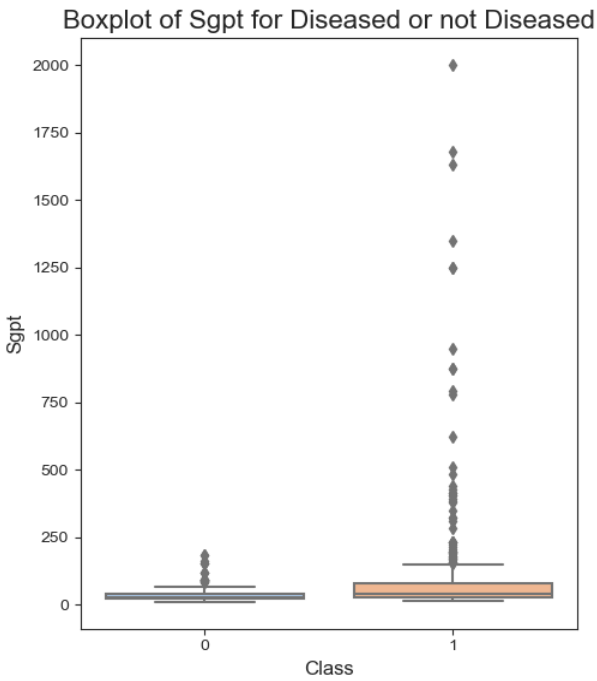

Density plot of DB
Density
DB
0.0
0.1
0.2
0.3
0.4
0.5
0
5
10
15
20

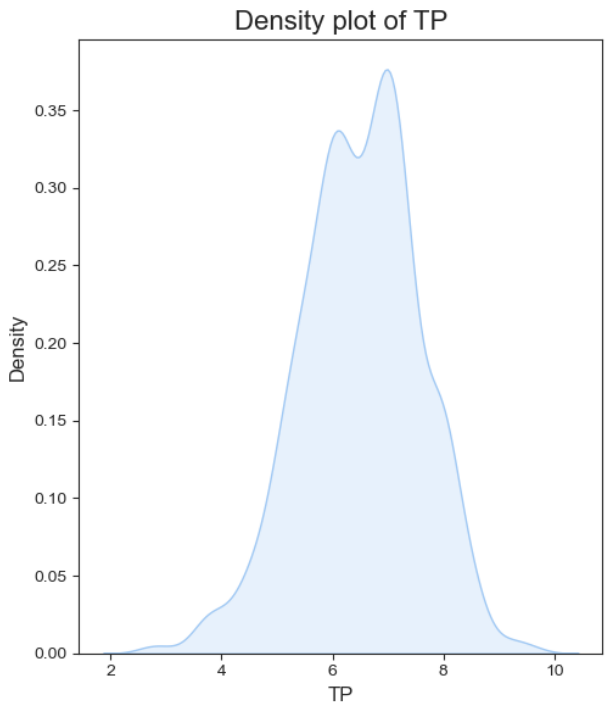

Boxplot of DB for Diseased or not Diseased
DB
Class
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0
1

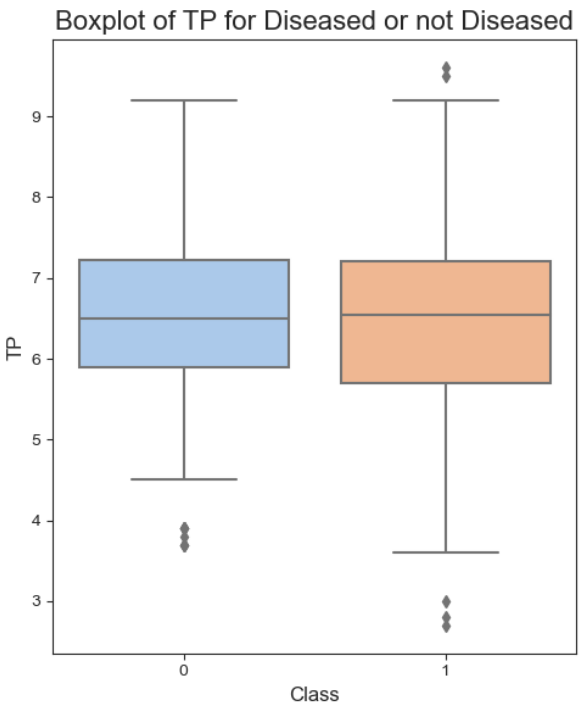

Density plot of Alkphos
Density
Alkphos
0.000
0.001
0.002
0.003
0.004
0
500
1000
1500
2000

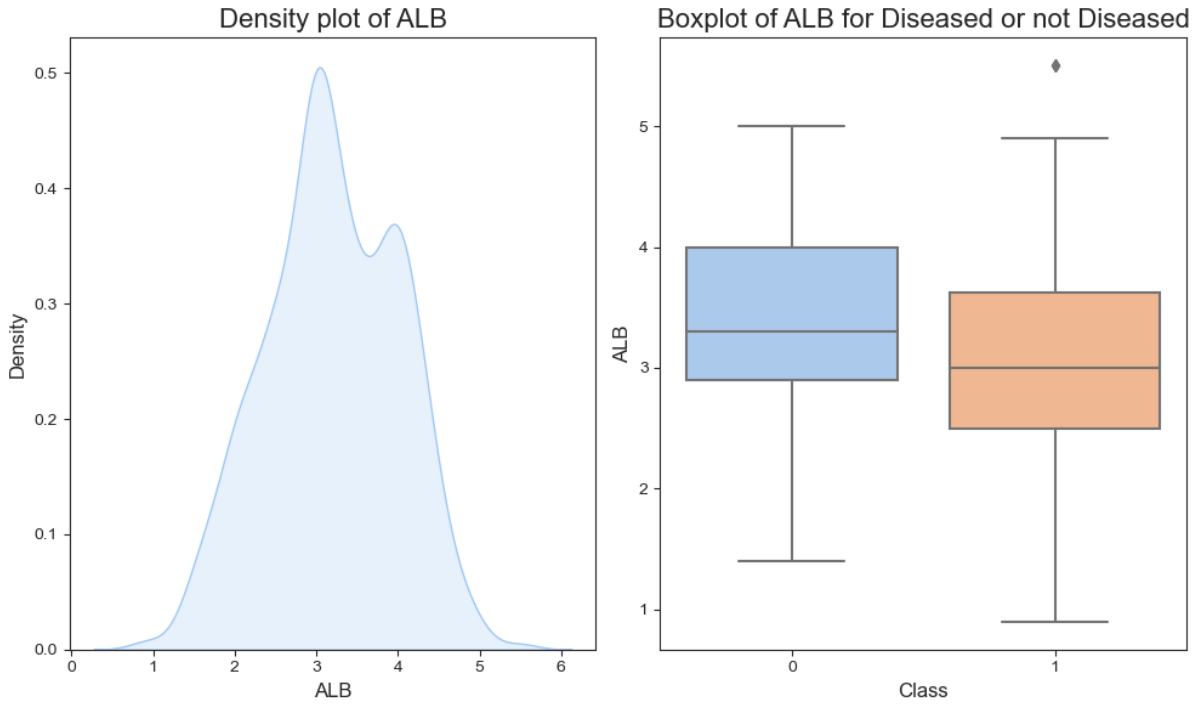

Boxplot of Alkphos for Diseased or not Diseased
Alkphos
Class
0
500
1000
1500
2000
0
1


Density plot of Sgpt
Density
Sgpt
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0
500
1000
1500
2000

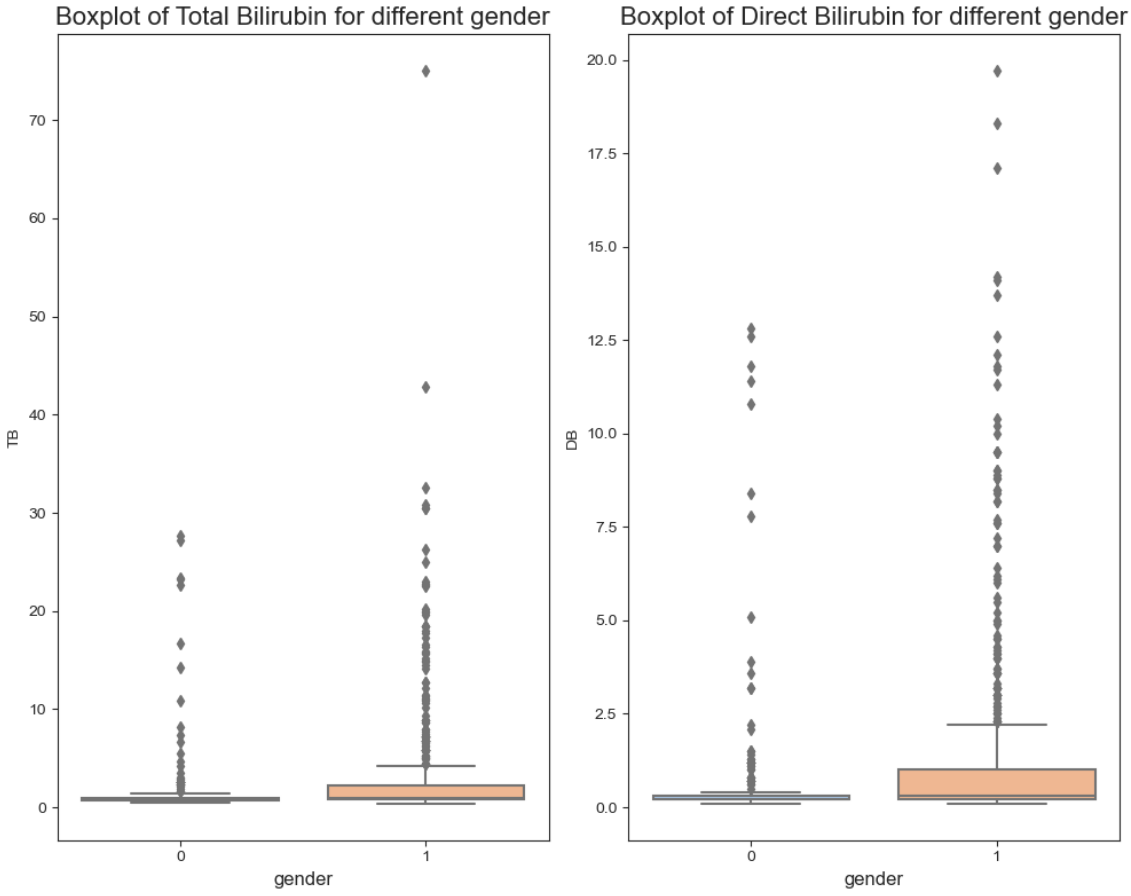

Boxplot of Sgpt for Diseased or not Diseased
Sgpt
Class
0
250
500
750
1000
1250
1500
1750
2000
0
1

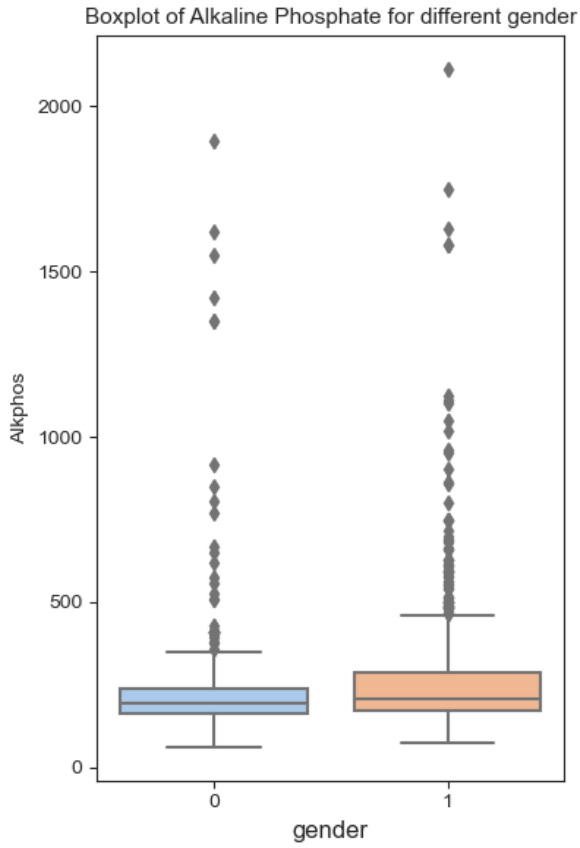

Density plot of TP
Density
TP
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
2
4
6
8
10

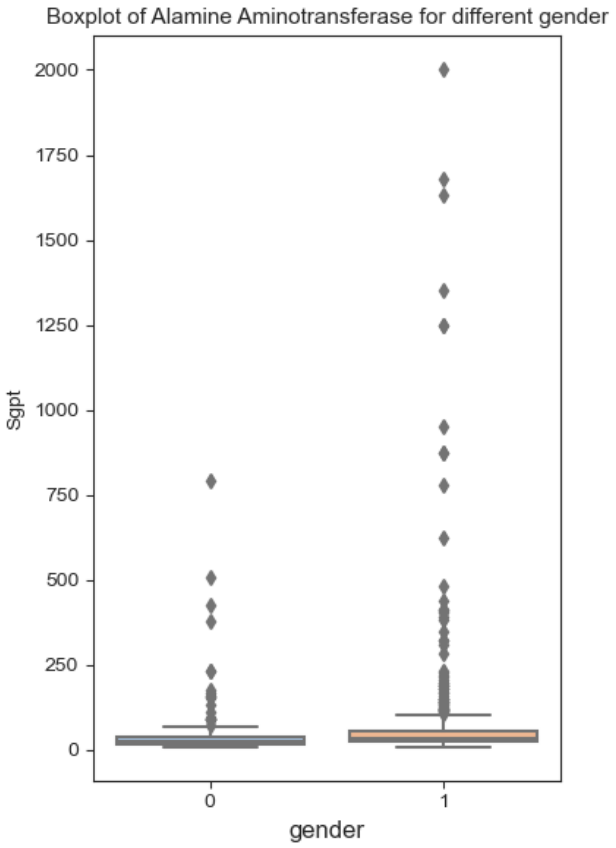

Boxplot of TP for Diseased or not Diseased
TP
Class
3
4
5
6
7
8
9
0
1

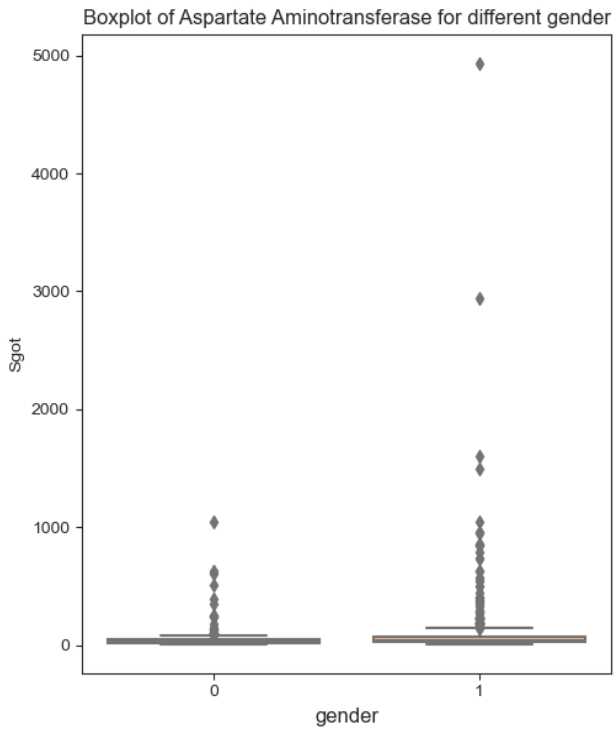


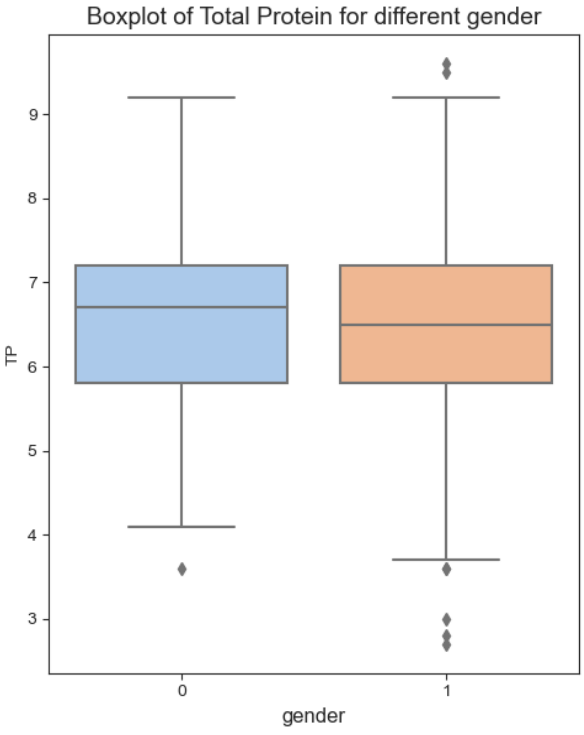
The visualization below tells us how different variables are distributed among genders, also we can compare the difference in means among the variables.

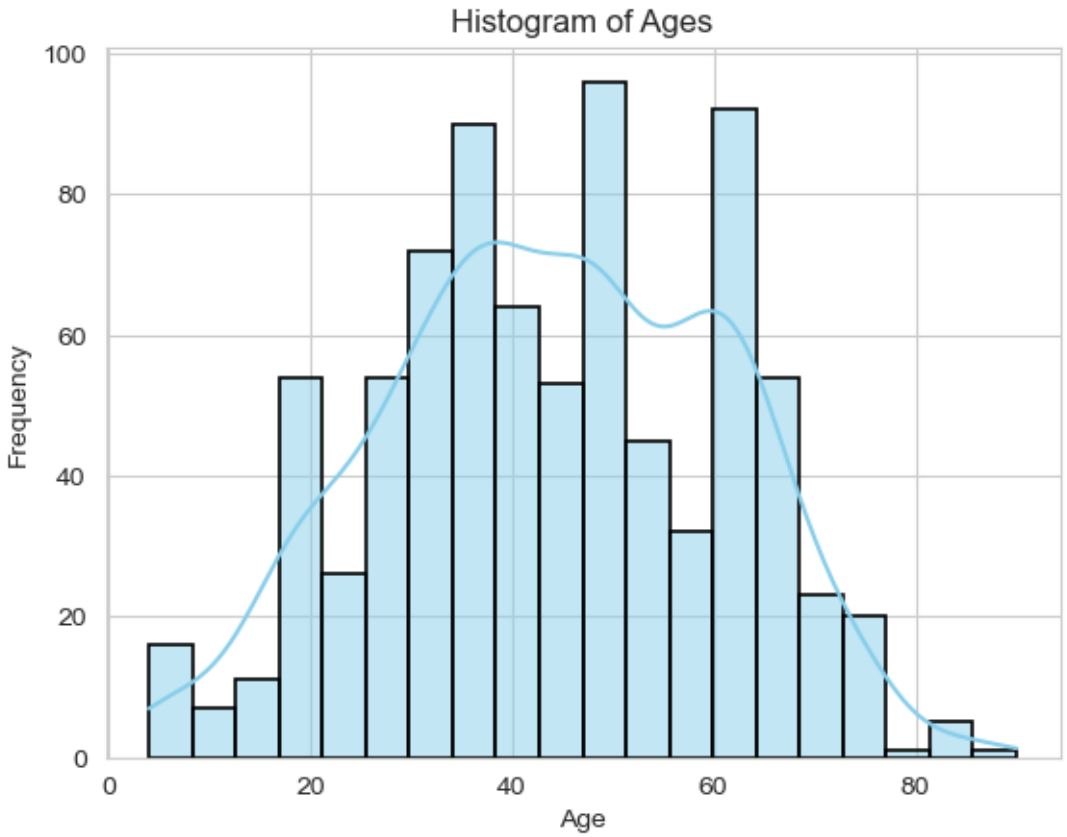



Boxplot of Alkaline Phosphate for different gender
Alkphos
2000
1500
1000
500
0
0
1
gender


Boxplot of Alamine Aminotransferase for different gender
Sgpt
2000
1750
1500
1250
1000
750
500
250
0
0
1
gender


Boxplot of Aspartate Aminotransferase for different gender
Sgot
5000
4000
3000
2000
1000
0
0
1
gender


Boxplot of Total Protein for different gender
TP
9
8
7
6
5
4
3
0
1
gender



Plot gives us insights about the distribution of ages in data. we can see that the maximum number of people are from groups 25-60

.

Descriptive Statistics of data

	Age	TB	DB	Alkphos	Sgpt	Sgot	TP	ALB	AG_Ratio
count	816.000000	816.000000	816.000000	816.000000	816.000000	816.000000	816.000000	816.000000	816.000000
mean	43.976716	2.670588	1.169853	273.531863	67.112745	89.265931	6.492157	3.191789	0.967855
std	16.760550	5.361553	2.438183	219.843684	156.308911	246.907547	1.064099	0.806488	0.315595
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000
25%	32.000000	0.700000	0.200000	168.000000	22.000000	24.000000	5.800000	2.600000	0.800000
50%	45.000000	0.900000	0.300000	202.000000	31.000000	36.000000	6.500000	3.100000	1.000000
75%	58.000000	1.900000	0.900000	289.000000	53.250000	66.000000	7.200000	3.900000	1.100000
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000

Checking the Normality of the variables

We used the Shapiro-Wilk test to assess the normality of variables. This test is a statistical tool used to evaluate whether a dataset is likely to originate from a normally distributed population. The null hypothesis is rejected if the p-value is below 0.05.

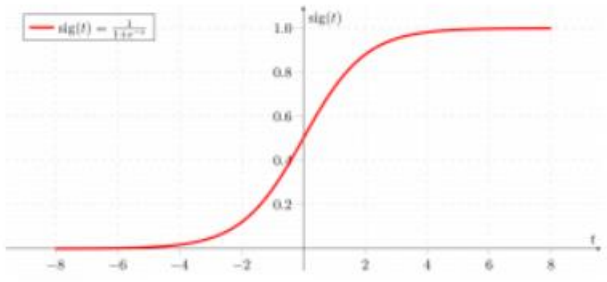
Variable	Class	Statistics	p-value	Normality
0 Age	0	0.976510	4.464110e-06	Not Gaussian (reject H0)
1 TB	0	0.502213	5.828308e-32	Not Gaussian (reject H0)
2 DB	0	0.508093	8.131433e-32	Not Gaussian (reject H0)
3 Alkphos	0	0.555298	1.330897e-30	Not Gaussian (reject H0)
4 Sgpt	0	0.706599	7.348484e-26	Not Gaussian (reject H0)
5 Sgot	0	0.601529	2.611075e-29	Not Gaussian (reject H0)
6 TP	0	0.990020	8.090091e-03	Not Gaussian (reject H0)
7 ALB	0	0.977756	8.090991e-06	Not Gaussian (reject H0)
8 AG_Ratio	0	0.977093	5.882406e-06	Not Gaussian (reject H0)
9 Age	1	0.990166	6.987524e-03	Not Gaussian (reject H0)
10 TB	1	0.528195	7.621885e-32	Not Gaussian (reject H0)
11 DB	1	0.606037	1.094295e-29	Not Gaussian (reject H0)
12 Alkphos	1	0.622385	3.424623e-29	Not Gaussian (reject H0)
13 Sgpt	1	0.373496	2.129111e-35	Not Gaussian (reject H0)
14 Sgot	1	0.319101	1.767582e-36	Not Gaussian (reject H0)
15 TP	1	0.991071	1.294942e-02	Not Gaussian (reject H0)
16 ALB	1	0.992797	4.311235e-02	Not Gaussian (reject H0)
17 AG_Ratio	1	0.927944	2.894293e-13	Not Gaussian (reject H0

We can see that normality is not followed in any of the variables. We can't use Parametric models, such as Linear Discriminant Analysis (LDA), which assume that the data are normally distributed.

Logistic regression

Logistic regression estimates the probability of an event occurring, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied to the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds or the natural logarithm of odds, and this logistic function is represented by the following formulas:

$$\sigma(z) = \frac{1}{1-e^{-z}}$$



Sigmoid function

Procedure:-

We split the data into train and test, 80 percent of data in the training set and 20 percent data in the testing set. Then we fitted a logistic regression model on the training data,
We have got the log-odds function as -:

$$\ln\frac{p}{1-p} = -4.02508 + 0.01603 * Age + 0.55214 * DB + 0.01685 * Sgpt + 0.87964 * TP - 1.59294 * ALB + 1.58747 * AG\_Ratio$$

$$\text{probability} = e^{-4.02508+0.01603*\text{Age}+0.55214*\text{DB}+0.01685*\text{Sgpt}+0.87964*\text{TP}-1.59294*\text{ALB}+1.58747*\text{AG\_Ratio}}$$

```
Call:  glm(formula = Classes ~ Age + DB + Sgpt + TP + ALB + AG_Ratio,
    family = binomial, data = train_dataset)

Coefficients:
(Intercept)          Age           DB         Sgpt           TP          ALB
   -4.02508      0.01603      0.55214      0.01685      0.87964     -1.59294
   AG_Ratio
    1.58747

Degrees of Freedom: 651 Total (i.e. Null);  645 Residual
Null Deviance:      903.8
Residual Deviance: 721.4        AIC: 735.4
```

Checking the significance of regression coefficients: -

```
Call:
glm(formula = Classes ~ ., family = binomial, data = train_dataset)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.9314214  1.0867255  -3.618 0.000297 ***
Age          0.0166705  0.0054996   3.031 0.002435 **
gender      -0.0994140  0.2042348  -0.487 0.626426
TB          -0.3264871  0.3671742  -0.889 0.373902
DB           1.1103995  0.7070390   1.570 0.116301
Alkphos      0.0007896  0.0006317   1.250 0.211350
Sgpt         0.0133933  0.0040989   3.268 0.001085 **
Sgot         0.0027258  0.0026529   1.028 0.304184
TP           0.8171828  0.2933244   2.786 0.005337 **
ALB         -1.4789983  0.5607226  -2.638 0.008348 **
AG_Ratio     1.5378880  0.8735795   1.760 0.078332 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 903.81  on 651  degrees of freedom
Residual deviance: 717.48  on 641  degrees of freedom
AIC: 739.48

Number of Fisher Scoring iterations: 7
```

The variables Age, Sgpt, TP, ALB contribute more towards the prediction of the diseased.
we checked the multicollinearity of the data using the VIF (Variance Inflation Factor)

```
      Age        DB      Sgpt        TP       ALB  AG_Ratio
 1.078603  1.111736  1.105360 11.521978 23.914501  8.017980
```

From the above values we can see that there is some multicollinearity in the data.

By using feature selection, we mitigate multicollinearity by selecting a subset of features that are less correlated with each other. It will help us in speed up the training process and make model deployment more efficient.

Feature selection using stepwise AIC method -:

```
Step:  AIC=735.41
Classes ~ Age + DB + Sgpt + TP + ALB + AG_Ratio

           Df Deviance    AIC
<none>          721.41 735.41
+ Alkphos   1   719.65 735.65
+ Sgot      1   720.23 736.23
+ TB        1   720.65 736.65
- AG_Ratio  1   724.97 736.97
+ gender    1   721.20 737.20
- Age       1   730.16 742.16
- ALB       1   730.58 742.58
- TP        1   731.45 743.45
- DB        1   756.59 768.59
- Sgpt      1   769.51 781.51
```

From the feature selection, we got 4 variables and we fitted the model using this variables.

```
Call:  glm(formula = Classes ~ Age + DB + Sgpt + TP + ALB + AG_Ratio,
    family = binomial, data = train_dataset)

Coefficients:
(Intercept)          Age           DB         Sgpt           TP          ALB
   -4.02508      0.01603      0.55214      0.01685      0.87964     -1.59294
   AG_Ratio
    1.58747

Degrees of Freedom: 651 Total (i.e. Null);  645 Residual
Null Deviance:      903.8
Residual Deviance: 721.4        AIC: 735.4
> -
```

Checking the significance of the coefficients:-

```
Call:
glm(formula = Classes ~ Age + DB + Sgpt + TP + ALB + AG_Ratio,
    family = binomial, data = train_dataset)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.025079   1.061882  -3.791 0.000150 ***
Age          0.016034   0.005466   2.933 0.003355 **
DB           0.552139   0.142527   3.874 0.000107 ***
Sgpt         0.016846   0.003167   5.319 1.04e-07 ***
TP           0.879641   0.291640   3.016 0.002560 **
ALB         -1.592936   0.556739  -2.861 0.004221 **
AG_Ratio     1.587467   0.872241   1.820 0.068761 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 903.81  on 651  degrees of freedom
Residual deviance: 721.41  on 645  degrees of freedom
AIC: 735.41

Number of Fisher Scoring iterations: 7
```

Now we fitted the model on the data and We got train test accuracy of the model as: -

```
Training_Accuracy Test_Accuracy
        0.7009202     0.6646341
```

Confusion Matrix for test data

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 60 44
         1 11 49

               Accuracy : 0.6646
                 95% CI : (0.5868, 0.7364)
    No Information Rate : 0.5671
    P-Value [Acc > NIR] : 0.006806

                  Kappa : 0.3526

 Mcnemar's Test P-Value : 1.597e-05

            Sensitivity : 0.8451
            Specificity : 0.5269
         Pos Pred Value : 0.5769
         Neg Pred Value : 0.8167
             Prevalence : 0.4329
         Detection Rate : 0.3659
   Detection Prevalence : 0.6341
      Balanced Accuracy : 0.6860

       'Positive' Class : 0
```

Precision recall and f1 score: -

	precision	recall	f1_score
1	0.8450704	0.5769231	0.6857143

Interpretation

The logistic regression model achieved an accuracy of 70% on the training data and 66% on the test data. This means that the model correctly predicted the outcome for 70% of the observations in the training set and 66% of the observations in the test set.

The precision of the model is 0.84, which means that when the model predicts a positive outcome, it is correct 84% of the time. The recall of the model is 0.57, which means that the model correctly identifies 57% of all positive cases. The F1 score, which is the harmonic mean of precision and recall, is 0.68. This indicates the balance between precision and recall, with higher values indicating a better balance.

Overall, the model's performance is decent, with a relatively high precision indicating that it is good at avoiding false positives. However, the lower recall suggests that the model may miss some positive cases. The F1 score provides a balanced assessment of the model's performance, considering both precision and recall

Support vector machine

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. The main objective of the SVM algorithm is to find the optimal hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

Hyperparameter tuning is the process of selecting the optimal values for a machine learning model's hyperparameters. Hyperparameters are settings that control the learning process of the model, such as the learning rate, the number of neurons in a neural network, or the kernel size in a support vector machine. The goal of hyperparameter tuning is to find the values that lead to the best performance on a given task. Hyperparameters are configuration variables that are set before the training process of a model begins. They control the learning process itself, rather than being learned from the data. Hyperparameters are often used to tune the performance of a model, and they can have a significant impact on the model's accuracy, generalization, and other metrics.

Procedure -:

Scaling is important in SVMs (Support Vector Machines) primarily because SVMs are sensitive to the scale of the input features. SVMs try to find the hyperplane that best separates the classes in feature space. If the features are not on a similar scale, the SVM might give more weight to features with larger scales, leading to a suboptimal hyperplane.

Scaling the features ensures that all features contribute equally to the distance computations that SVMs rely on. This can result in a better-performing model with a more appropriate decision boundary. In SVMs with kernels, the importance of scaling becomes even more significant, as the kernel functions compute distances between data points. After scaling the features, We fitted the SVM algorithm to the liver data to predict the diseased and not diseased. Used hyperparameter tuning to increase the accuracy of the model by changing the values of cost and gamma. Also calculated accuracy and other metrics to increase the accuracy of our model.

```
Call:
svm(formula = Classes ~ ., data = train_dataset, kernel = "radial", cost = 0.01, gamma = 0.1, type = "C-classification", scale = TRUE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  0.01

Number of Support Vectors:  646

 ( 323 323 )
```

Train test accuracy

```
  Training_Accuracy Test_Accuracy
1         0.5046012     0.4329268
```

Hyperparameter tunning to get best parameters for the model

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
    1   100

- best performance: 0.07827506

- Detailed performance results:
    cost gamma      error dispersion
1    0.1   0.1 0.30214452 0.05225315
2    1.0   0.1 0.30209790 0.04602469
3   10.0   0.1 0.28668998 0.04181484
4  100.0   0.1 0.23307692 0.05034659
5    0.1   1.0 0.36165501 0.09991597
6    1.0   1.0 0.18095571 0.01709474
7   10.0   1.0 0.13655012 0.02775857
8  100.0   1.0 0.14272727 0.03659534
9    0.1  10.0 0.52755245 0.03783169
10   1.0  10.0 0.08752914 0.02642699
11  10.0  10.0 0.08293706 0.03277227
12 100.0  10.0 0.08293706 0.03277227
13   0.1 100.0 0.52755245 0.03783169
14   1.0 100.0 0.07827506 0.02661334
15  10.0 100.0 0.07827506 0.02661334
16 100.0 100.0 0.07827506 0.02661334
```

Fitting model with best parameters

```
Call:
best.tune(METHOD = svm, train.x = Classes ~ ., data = train_dataset, ranges = list(cost = c(0.1, 1, 10, 100), gamma = c(0.1, 1, 10, 100)),
    kernel = "radial", scale = TRUE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  557

 ( 237 320 )
```

Accuracy after hyperparameter tunning

```
 Training_Accuracy Test_Accuracy
                 1     0.9634146
```

Confusion matrix for test data

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 65  0
         1  6 93

               Accuracy : 0.9634
                 95% CI : (0.9221, 0.9865)
    No Information Rate : 0.5671
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.9247

 Mcnemar's Test P-Value : 0.04123

            Sensitivity : 0.9155
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.9394
             Prevalence : 0.4329
         Detection Rate : 0.3963
   Detection Prevalence : 0.3963
      Balanced Accuracy : 0.9577

       'Positive' Class : 0
```

Precision, Recall and F1 Score for SVM

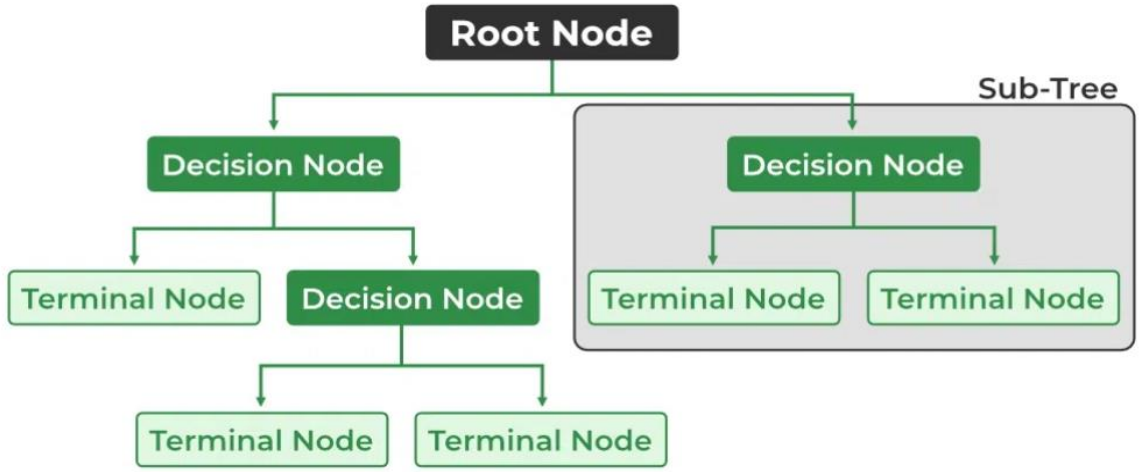
```
  precision recall  f1_score
1  0.915493      1 0.9558824
```

Interpretation: -

The model has a high overall accuracy, correctly classifying 96% of the instances in the test dataset. This indicates that the model is effective at distinguishing between the classes in the dataset. The model's precision of 0.91 indicates that when it predicts a positive result, it is correct about 91% of the time. This suggests a relatively low false positive rate .The recall of 1 indicates that the model can correctly identify all actual positive instances in the dataset. This suggests that the model has a high sensitivity and is capturing all positive cases. The F1 score is high at 0.95, indicating a good balance between precision and recall. This suggests that the model is performing well in terms of both identifying positive instances and avoiding false positives. The SVM model seems to be performing very well on the test dataset, with high accuracy, precision, recall, and F1-score.

Decision tree

A decision tree is a flowchart-like tree structure where each internal node denotes the feature, branches denote the rules and the leaf nodes denote the result of the algorithm. It is a versatile supervised machine-learning algorithm, which is used for both classification and regression problems. It is one of the very powerful algorithms. It is also used in Random Forest to train on different subsets of training data, which makes Random Forest one of the most powerful algorithms in machine learning.



Decision Tree

---

1. We fit the decision tree algorithm on the data and the data gave us a tree-:
Feature Selection before Pruning:

•Advantages: Performing feature selection before pruning can reduce the complexity of the tree and potentially improve its interpretability. Removing irrelevant or redundant features early in the process can lead to a more concise and understandable tree.

•Disadvantages: Pruning a decision tree before feature selection may lead to overfitting, as the tree may capture noise or irrelevant details in the data. Pruning relies on the tree being initially grown to a sufficiently large size to capture the underlying patterns in the data.

2. Pruning before Feature Selection:

•Advantages: Pruning a decision tree before feature selection can help reduce the risk of overfitting by simplifying the tree's structure early in the modeling process. This can lead to a more generalizable model.

•Disadvantages: Pruning before feature selection may result in a less interpretable model, as the tree may retain unnecessary complexity that could have been eliminated by feature selection.

In practice, the choice between pruning before or after feature selection depends on the specific characteristics of the dataset, the modelling goals, and the trade-offs between model complexity and interpretability. It may be beneficial to experiment with both approaches and compare the performance of the resulting models to determine the most effective strategy for a given problem. We decided to go with 2nd method

Procedure-:

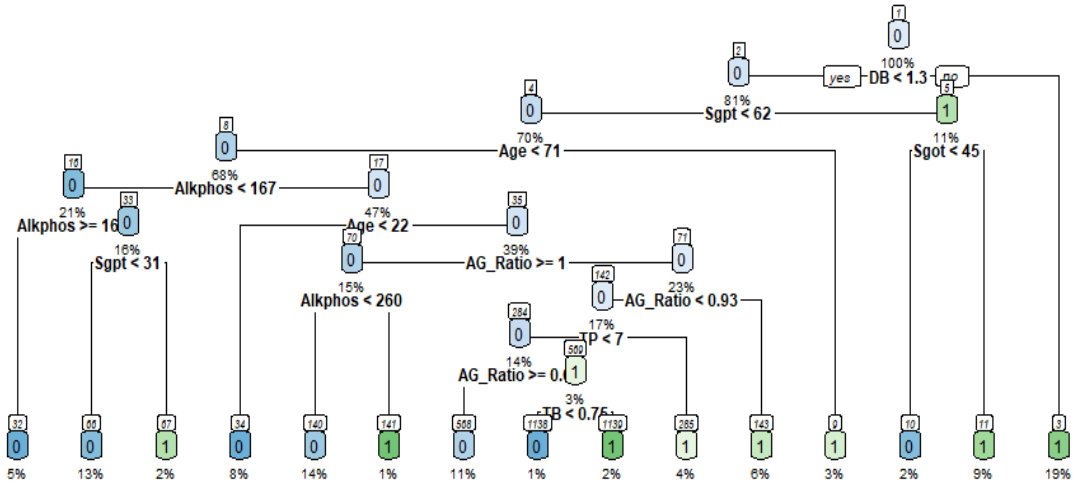
We fitted our decision tree on data to predict the diseased and not diseased. Then to get better results we pruning of tree to increase the efficiency of model .Then fitted the model with the best variables to increase accuracy and calculated metrics.

```
Classification tree:
tree(formula = Classes ~ ., data = train_dataset)
Variables actually used in tree construction:
[1] "DB"      "Sgpt"    "Alkphos" "Age"     "Sgot"    "TP"
Number of terminal nodes:  22
Residual mean deviance:  0.7648 = 481.8 / 630
Misclassification error rate: 0.1825 = 119 / 652
```

Accuracy for the Model :-

```
  Training_Accuracy Test_Accuracy
1         0.8174847     0.7378049
```

From accuracy we can see that model is suffering from overfitting so we decided that we will do pruning of the model. Pruning is often used to reduce overfitting, where the model learns noise from the training data. While this is generally beneficial, if the model was initially overfitting significantly, pruning could lead to a decrease in accuracy on the training data.



After pruning accuracy of the model is coming out:-

```
 Training_Accuracy Test_Accuracy
         0.8128834     0.7621951
```

Confusion Matrix for the data

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 58 26
         1 13 67

               Accuracy : 0.7622
                 95% CI : (0.6896, 0.8251)
    No Information Rate : 0.5671
    P-Value [Acc > NIR] : 1.504e-07

                  Kappa : 0.5259

 Mcnemar's Test P-Value : 0.05466

            Sensitivity : 0.8169
            Specificity : 0.7204
         Pos Pred Value : 0.6905
         Neg Pred Value : 0.8375
             Prevalence : 0.4329
         Detection Rate : 0.3537
   Detection Prevalence : 0.5122
      Balanced Accuracy : 0.7687

       'Positive' Class : 0
```

Precision recall and f1_score for the decision tree

```
 precision    recall  f1_score
 0.8169014 0.6904762 0.7483871
```

Interpretation

The decision tree model achieved an accuracy of 76% on the test data and 81% on the training data after pruning. This means that after simplifying the decision tree to reduce overfitting, the model's performance improved. Outliers can significantly affect the structure and performance of the tree. Outliers may be treated as unique cases and lead to the creation of branches that are specific to these outliers, potentially leading to overfitting. Additionally, outliers can affect the impurity measures used in decision tree algorithms, such as Gini impurity or entropy, leading to suboptimal splits.

Random Forest

Random Forest is a powerful ensemble learning method that combines the predictions of multiple decision trees to improve the overall accuracy and robustness of the model. It works by constructing many decision trees during training, each trained on a bootstrapped sample of the data and considering only a subset of the features at each node. This randomness helps to reduce overfitting and decorrelate the individual trees, resulting in a more accurate and stable prediction. For classification tasks, Random Forest combines the predictions of the individual trees using voting, while for regression tasks, it averages the predictions. Additionally, Random Forest provides insights into feature importance, making it a popular choice for both classification and regression problems, especially in situations with high-dimensional data and complex relationships.

Procedure -:
We fitted our model with the data to predict the diseased and not diseased. Initially we decided to go with ntree=10 trees and number of variables at each split as 3 for random forest then we did hyperparameter tuning and got ntree=500 and number of variables at each split as 4 and thereafter calculated accuracy and other important metrics.

```
Call:
 randomForest(formula = train_dataset$Classes ~ ., data = train_dataset,      ntree = 10,
method = "classification")
               Type of random forest: classification
                     Number of trees: 10
No. of variables tried at each split: 3

        OOB estimate of  error rate: 21.22%
Confusion matrix:
    0   1 class.error
0 296  28  0.08641975
1 108 209  0.34069401
```

Train and test accuracy for our model-:

```
 Training_Accuracy Test_Accuracy
         0.9846626     0.8414634
```

To increase the accuracy of model we have done hyperparameter tunning

```
Call:
 randomForest(x = x, y = y, mtry = param$mtry)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 15.49%
Confusion matrix:
    0   1 class.error
0 304  25  0.07598784
1  76 247  0.23529412
```

Training accuracy and test accuracy after tunning are coming out to be

```
Training_Accuracy Test_Accuracy
                1      0.902439
```

Confusion matrix for fitted data: -

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 68 13
         1  3 80

               Accuracy : 0.9024
                 95% CI : (0.8464, 0.9432)
    No Information Rate : 0.5671
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.8046

 Mcnemar's Test P-Value : 0.02445

            Sensitivity : 0.9577
            Specificity : 0.8602
         Pos Pred Value : 0.8395
         Neg Pred Value : 0.9639
             Prevalence : 0.4329
         Detection Rate : 0.4146
   Detection Prevalence : 0.4939
      Balanced Accuracy : 0.9090

       'Positive' Class : 0
```

Precision recall and f1_score for Random Forest

precision, recall, and F1-score-:

	precision	recall	f1_score
1	0.9577465	0.8395062	0.8947368

Interpretation: -

Having a training accuracy of 1 (100%) and a test accuracy of 0.90 (90%) suggests that your random forest model is performing quite well. Here are some conclusions you could draw from these results: The model is likely overfitting to some extent, as indicated by the perfect training accuracy. This means the model has learned the training data too well and may not generalize as effectively to unseen data. Despite some overfitting, the model is still able to generalize reasonably well to the test data, as evidenced by the 90% test accuracy. This indicates that the model is capturing the underlying patterns in the data.

The model might be relatively complex, considering its ability to fit the training data perfectly. This complexity could potentially be reduced to improve generalization without sacrificing too much performance. Precision (0.95) indicates that when the model predicts a positive result, it is correct about 95% of the time. In other words, among all the instances the model classified as positive, 95% of them are positive.

Recall suggests that the model correctly captures 83% of all positive cases in the dataset.

The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics. An F1 score of 0.89 indicates a good balance between precision and recall.

Comparative study of models:-

Models	Training Accuracy	Test Accuracy
Logistic Regression	0.7009202	0.664631
SVM (Support Vector Machine)	1	0.9634146
Decision Tree	0.8128834	0.761951
Random Forest	1	0.902439

Conclusion:-

After evaluating various machine learning models for predicting liver cirrhosis in the North East of Andhra Pradesh, India, based on biochemical markers, including albumin and other enzymes related to metabolism various conclusions can be drawn:

The SVM (Support Vector Machine) model outperformed other models, achieving a training accuracy of 100% and a test accuracy of 96.34%. This indicates that the SVM model was able to learn the patterns in the training data effectively and generalize well to unseen data, making it a strong candidate for predicting liver disease in this region.

The Decision Tree model exhibited some overfitting, with a training accuracy of 81.29% and a test accuracy of 76.20%. This suggests that the model may have learned noise in the training data, leading to reduced performance on unseen data.

The Random Forest model showed good generalization, with a training accuracy of 100% and a test accuracy of 90.24%. Although it performed well, it had a slightly lower test accuracy compared to the SVM model, indicating that the SVM model might generalize better to unseen data in this context.

The Logistic Regression model had the lowest performance, with a training accuracy of 70.09% and a test accuracy of 66.46%. This suggests that the model may not capture the complex relationships present in the data as effectively as the other models.

Based on the evaluation metrics, the SVM model appears to be the most suitable for predicting liver disease in patients in the northeast of Andhra Pradesh, India, using biochemical markers. It offers a balance between high accuracy and generalization, making it a reliable choice for this task

SVM model shows great promise for predicting liver cirrhosis in this region, offering a valuable tool for healthcare professionals to identify at-risk patients early and potentially reduce the mortality rates associated with liver disease.

Reference-:

- Introduction to statistical learning
  by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani
- Introduction to machine learning (NPTEL course) by Prof. Balaraman Ravindran , IIT Madras .

Code link :- https://github.com/Maheshmadan/Projects.git