PREDICTION AND ANALYSIS OF LIVER PATIENT DATA(ML)

Team ID-(SWTID1720175375)

Project Overview

The coexistence of liver diseases poses significant clinical challenges, requiring effective predictive models for early detection and intervention. In this study, we employed decision tree and logistic regression algorithms to predict the likelihood of liver disease in individuals diagnosed . Distinct datasets were utilized, for liver disease prediction, containing relevant clinical attributes. Through rigorous experimentation and evaluation, our models demonstrated promising performance in identifying the presence of liver disease in individuals.

In the ever-evolving field of healthcare, predicting and preventing liver diseases have become paramount to ensuring the well-being of individuals and communities. Today, we will delve into two powerful machine learning techniques, Logistic Regression and Decision Tree, which have shown significant potential in predicting the likelihood of these diseases. Logistic Regression is a statistical method that allows us to model the relationship between predictor variables and a binary outcome, such as the presence or absence of liver diseases. This technique is particularly useful when we want to understand the effect of various factors on the probability of a specific disease. Decision Trees, on the other hand, are a non-parametric method used for both classification and regression tasks. They work by recursively splitting the data into subsets based on the most significant predictor variables, thus creating a tree- like model that can be easily interpreted and understood. In the context of predicting liver diseases, decision trees can help identify the most important risk factors and provide a visual representation of the decision-making process. Combining these two techniques can lead to more accurate and robust predictions, as well as a deeper understanding of the complex interplay between various risk factors and the likelihood of developing liver diseases. By employing these machine learning algorithms, researchers and healthcare professionals can develop personalized preventive measures, early detection strategies, and more effective treatments to improve overall patient outcomes. In conclusion, the integration of Logistic Regression and Decision Tree in predicting liver diseases holds great promise for advancing healthcare and saving lives. As we explore these techniques further, we can expect to gain valuable insights into disease risk factors and contribute to the development of more effective, personalized healthcare strategies.

OBJECTIVES:

The overall objective of an ML project involving liver patient data can be broken down into two main parts: prediction and analysis. Here's a closer look at each:

Prediction:

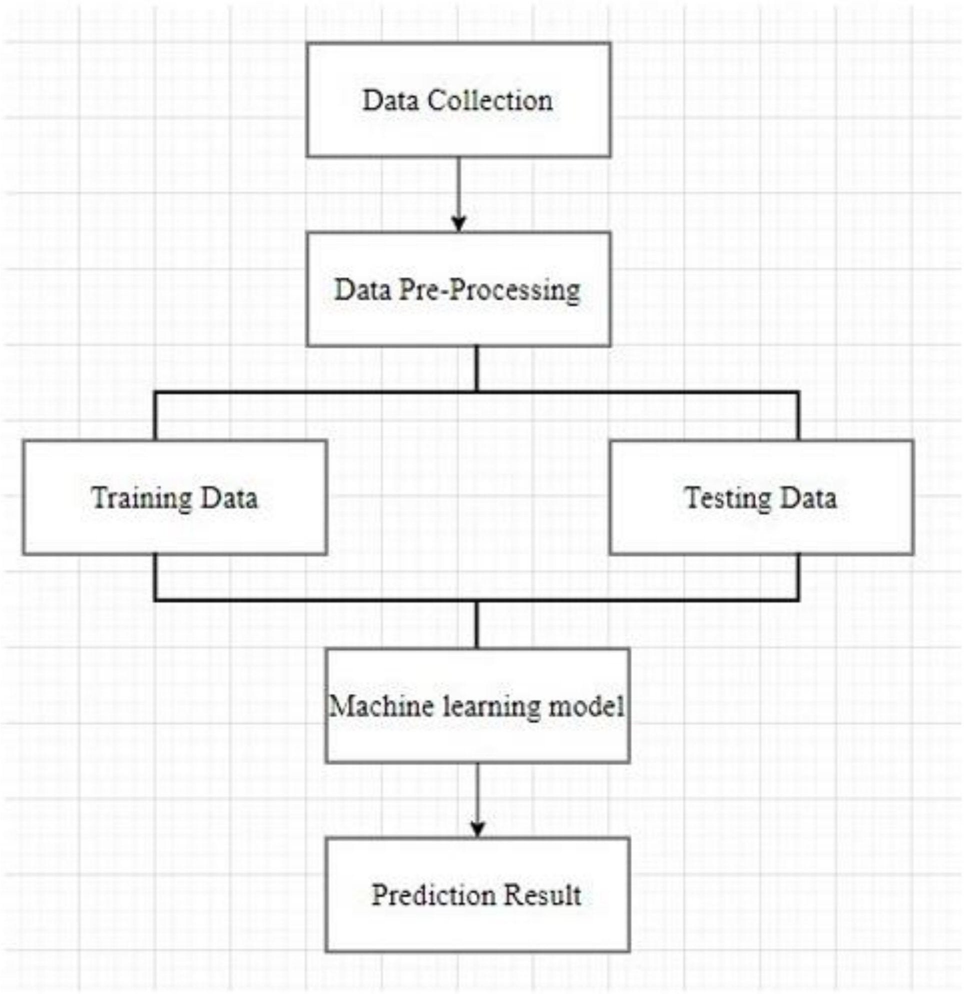
- **Identify patients at risk:** The project can aim to develop a model that predicts which patients are most likely to develop liver disease. This allows for early intervention and potentially prevents the progression of the disease.

- **Monitor disease progression:** The model can be used to track how a patient's liver disease is progressing over time. This allows healthcare providers to adjust treatment strategies as needed. (can be do in future)
- **Predict treatment response:** Machine learning can analyze patient data to predict how they might respond to different treatment options. This allows for more targeted and effective treatment plans. (can be done in future)

Problem statement:

Develop a model to predict the likelihood of a patient developing liver disease or classify the specific type of liver disease they might have.

Initial Planning:



Project Overview	
Objective	to determine whether a person has liver illness or not based on their health info
Scope	Can be developed into a full-fledged software and use it worldwide
Problem Statement	

Description	Develop a model to predict the likelihood of a patient developing liver disease or classify the specific type of liver disease they might have.
Impact	Large number of people will be able to is he/she has a chance of having liver disease and they don't have to spend a huge amounts of money for health care facilities
Proposed Solution	
Approach	we obtain information from the client about their age, protein levels, etc and try to predict if he/she has liver disease or not by training a model with the given data
Key Features	The customer will be informed right away whether he is at risk for liver illness.

Data Collection:

Data Collection The liver dataset was collected from the northeast of Andhra Pradesh, India. This dataset consists of 583 liver patient's data whereas 75.64% male patients and 24.36% are female patients. This dataset has contained 12 parameters where we choose 11 parameters for our further analysis and 1 parameter as a target class. Such as,

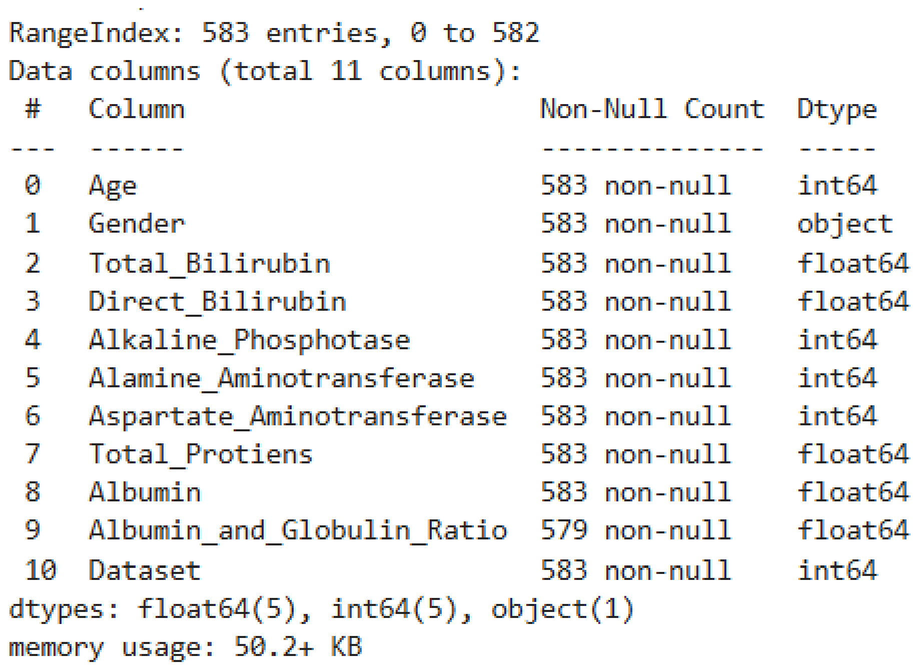
1. Age of the patient
2. Gender of the patient
3. Total Bilirubin
4. Direct Bilirubin
5. Alkaline Phosphatase
6. Alanine Aminotransferase
7. Aspartate Aminotransferase
8. Total Proteins
9. Albumin
10. Albumin and Globulin Ratio
11. Cholesterol
12. Dataset: field used to split the data into two sets (patient with liver disease, or no disease)

Data quality report:

Data Source	Data quality issue	Severity	**Resolution Plan**

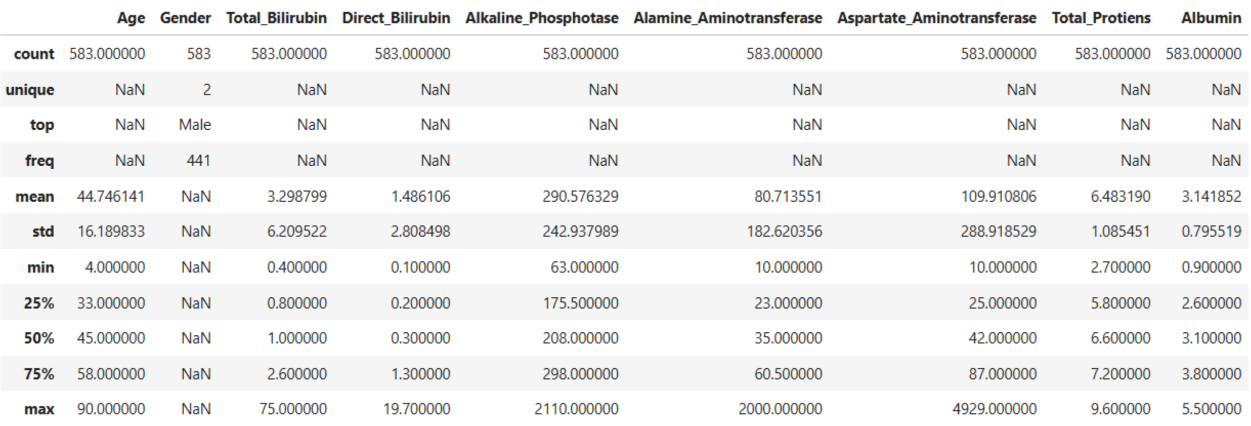
indian-liver-patient	Outliers: The dataset contains some features with a wide range of values, indicating the potential presence of outliers that should be identified and handled appropriately.	Moderate	Some of the feature values, such as Alamine_Aminotransferase, Aspartate_Aminotransferase, and Total_Bilirubin, appear to have a wide range, indicating the potential presence of outliers. Outliers can significantly impact model performance and should be identified and handled appropriately.
indian-liver-patient	Data Quality: The dataset contains some duplicate rows, which should be identified and removed to ensure data integrity	Moderate	The dataset has a large number of rows (over 500) and a wide range of values for the different features, indicating it may be a comprehensive dataset. However, there are a few rows with duplicate values for some patients, which could indicate data quality issues that need to be addressed
indian-liver-patient	Imbalanced Classes: The target variable (liver disease diagnosis) is imbalanced, with more instances of liver disease (class 1) than no liver disease (class 2). This may require techniques like oversampling or undersampling to balance the classes	Low	The "Dataset" column indicates that the target variable (liver disease diagnosis) is imbalanced, with 1 representing liver disease and 2 representing no liver disease. Imbalanced target variables can pose challenges for machine learning models and may require techniques like oversampling or undersampling to address

Data Overview



```
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Age                         583 non-null    int64
 1   Gender                      583 non-null    object
 2   Total_Bilirubin             583 non-null    float64
 3   Direct_Bilirubin            583 non-null    float64
 4   Alkaline_Phosphotase        583 non-null    int64
 5   Alamine_Aminotransferase    583 non-null    int64
 6   Aspartate_Aminotransferase  583 non-null    int64
 7   Total_Protiens              583 non-null    float64
 8   Albumin                     583 non-null    float64
 9   Albumin_and_Globulin_Ratio  579 non-null    float64
 10  Dataset                     583 non-null    int64
dtypes: float64(5), int64(5), object(1)
memory usage: 50.2+ KB
```

Univariate Analysis:

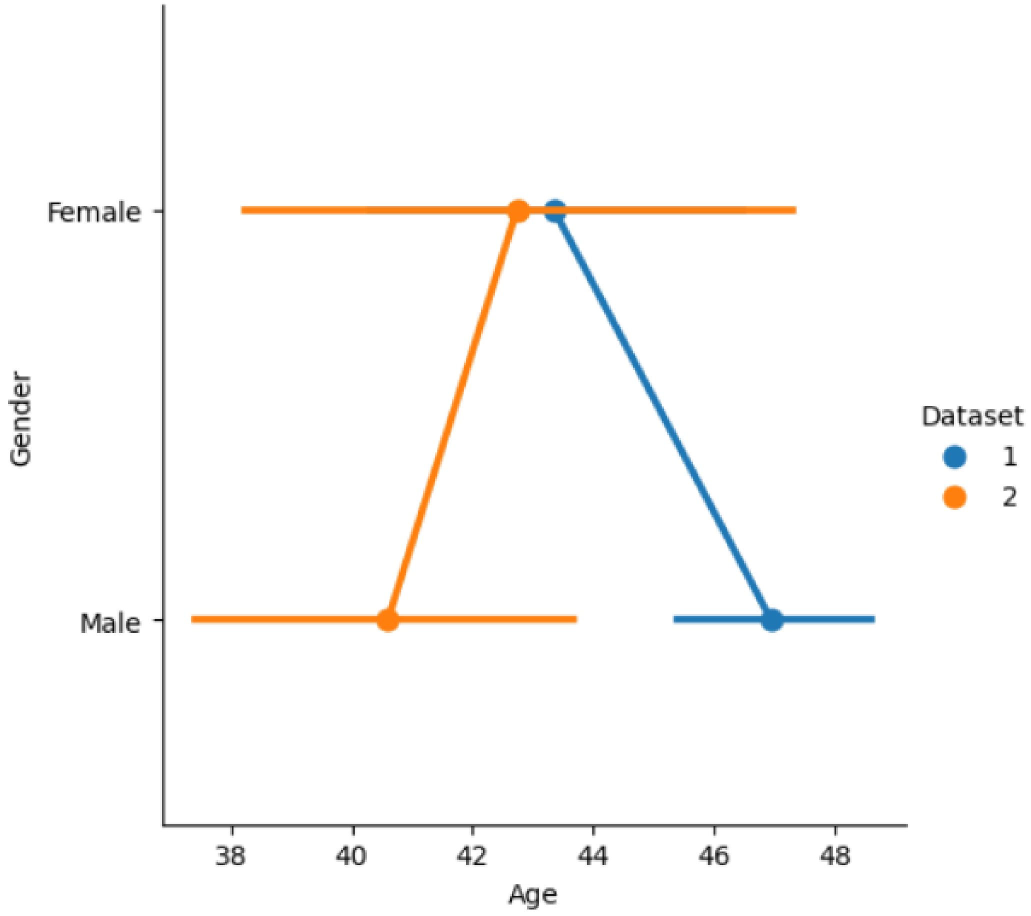


	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin
count	583.000000	583	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000
unique	NaN	2	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	Male	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	441	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	44.746141	NaN	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852
std	16.189833	NaN	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519
min	4.000000	NaN	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000
25%	33.000000	NaN	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000
50%	45.000000	NaN	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000
75%	58.000000	NaN	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000
max	90.000000	NaN	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000

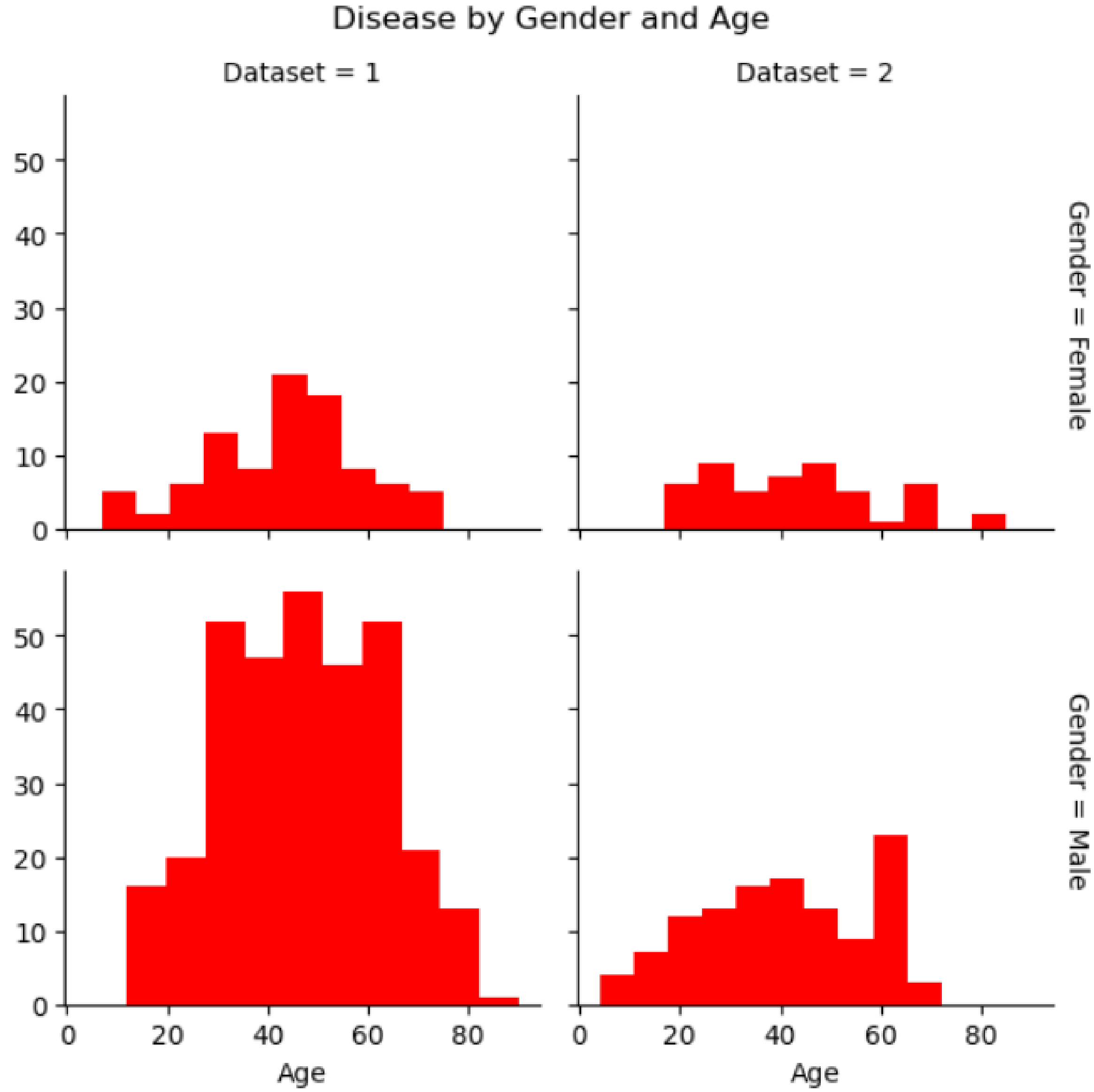
Bivariate Analysis:

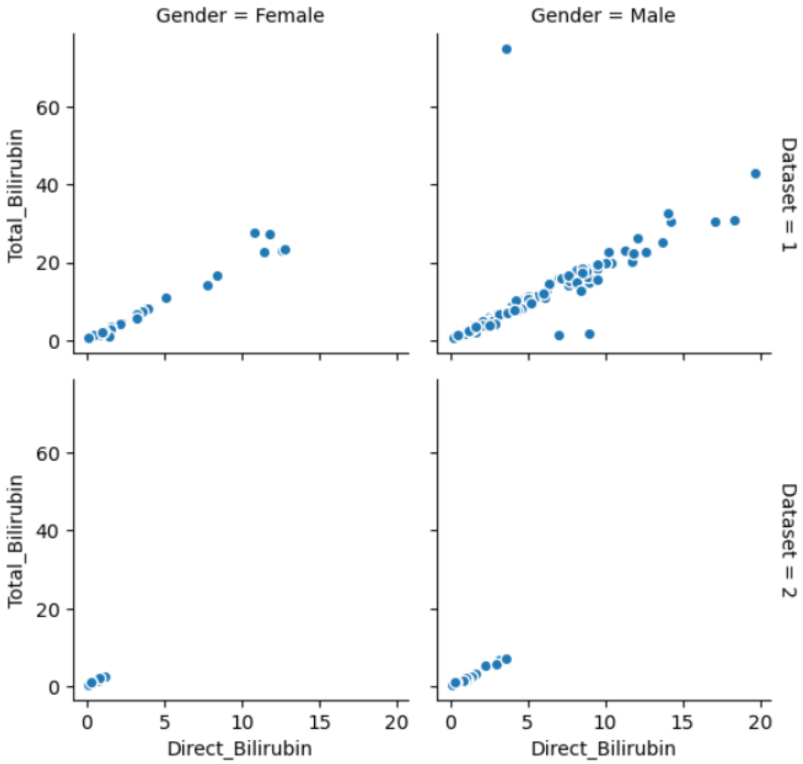
```
sns.catplot(x="Age", y="Gender", hue="Dataset", data=liver_df, kind="point")
```

```
<seaborn.axisgrid.FacetGrid at 0x1b6818204d0>
```

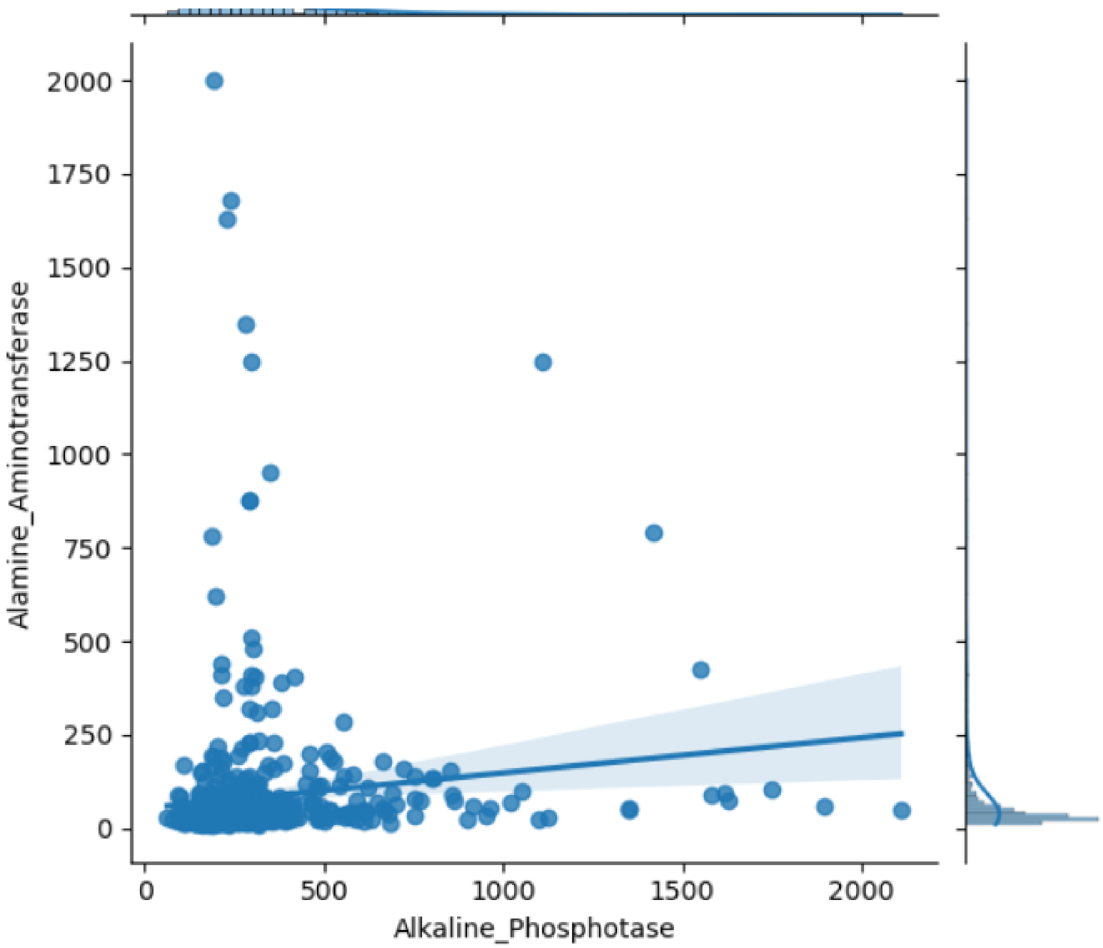


- Age seems to be a factor for liver disease for both male and female genders

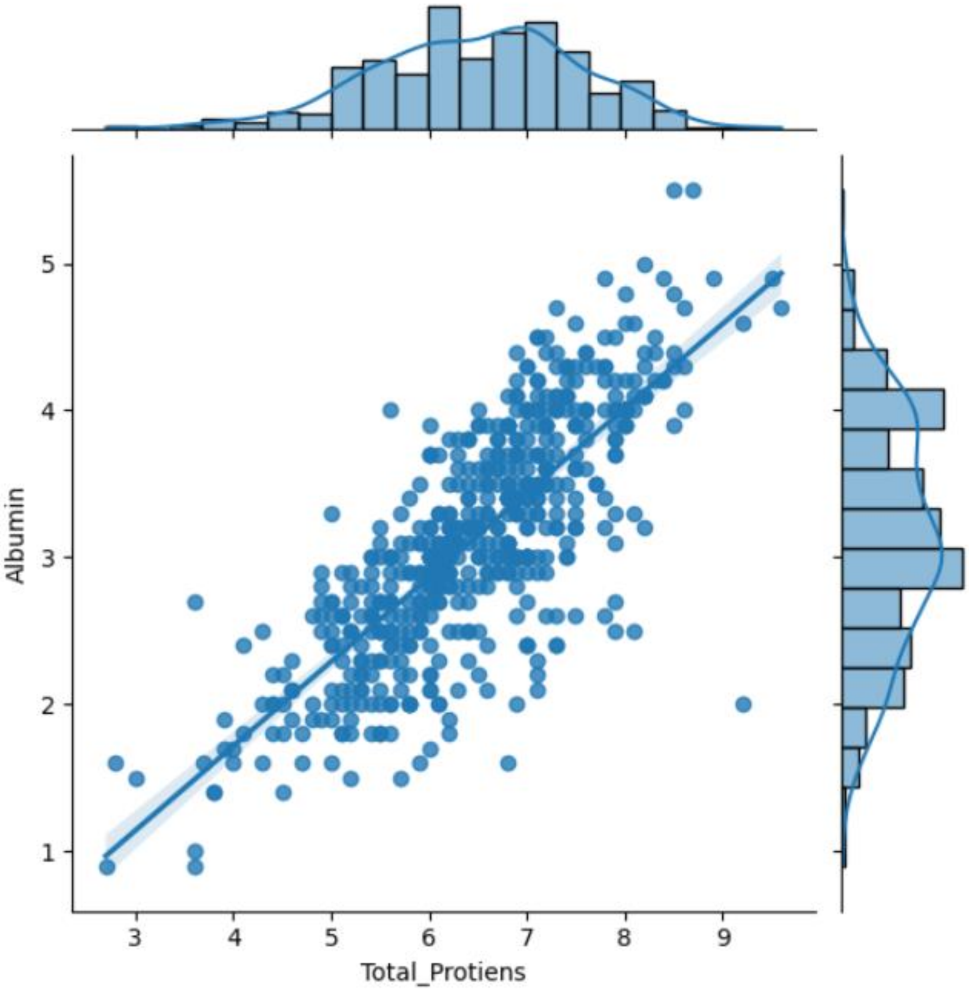

Disease by Gender and Age
Dataset = 1
Dataset = 2
Gender = Female
Gender = Male
0
10
20
30
40
50
0
20
40
60
80
Age
Age



- There seems to be direct relationship between Total_Bilirubin and Direct_Bilirubin. We have the possibility of removing one of this feature.



- No linear correlation between Alkaline_Phosphotase and Alamine_Aminotransferase


5
4
3
2
1
Albumin
3
4
5
6
7
8
9
Total_Protiens

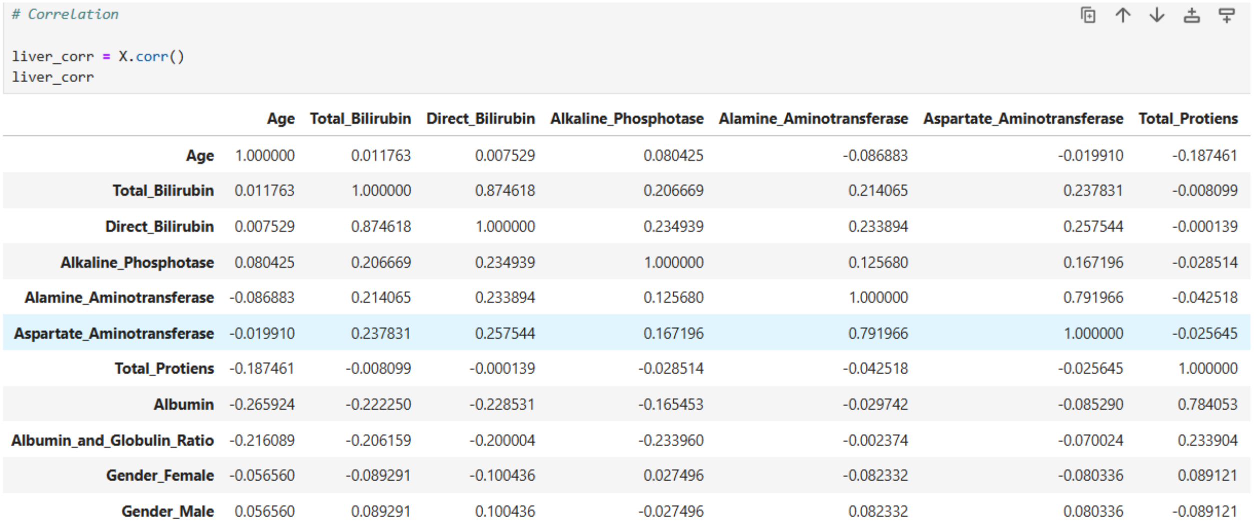
Observation:

From the above jointplots and scatterplots, we find direct relationship between the following features:

- Direct_Bilirubin & Total_Bilirubin
- Aspartate_Aminotransferase & Alamine_Aminotransferase
- Total_Protiens & Albumin
- Albumin_and_Globulin_Ratio & Albumin

Hence, we can very well find that we can omit one of the features. I'm going to keep the follwing features:

- Total_Bilirubin
- Alamine_Aminotransferase
- Total_Protiens
- Albumin_and_Globulin_Ratio
- Albumin



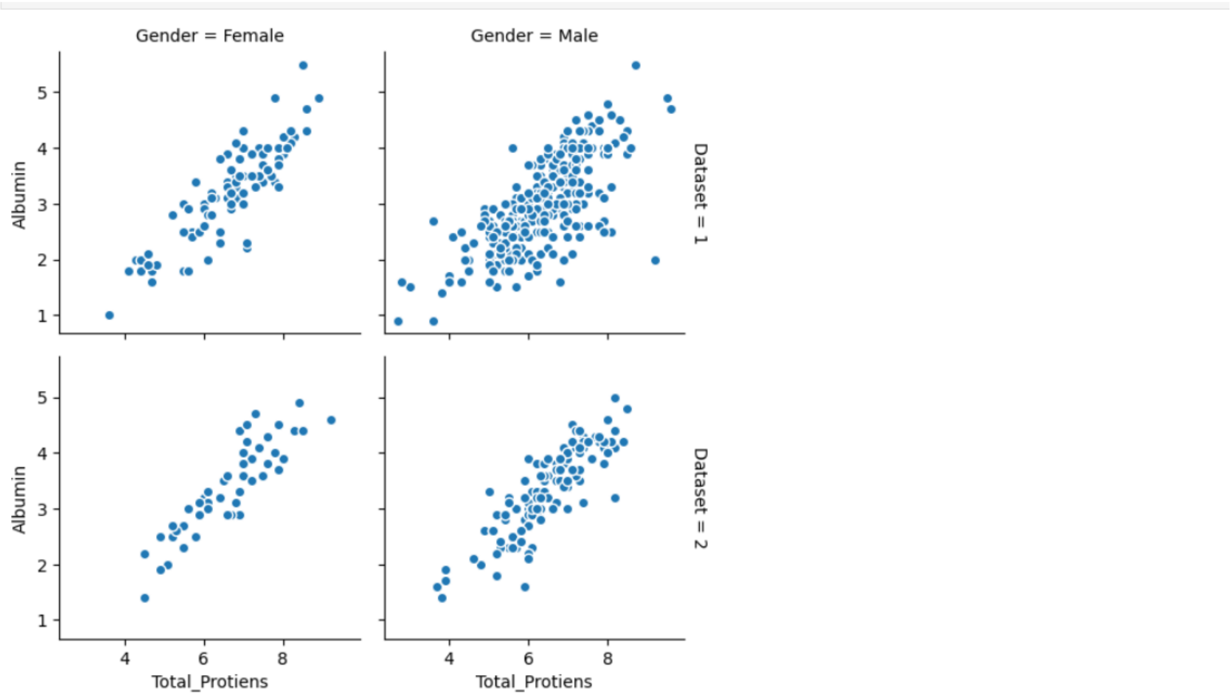
```
# Correlation

liver_corr = X.corr()
liver_corr
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens
Age	1.000000	0.011763	0.007529	0.080425	-0.086883	-0.019910	-0.187461
Total_Bilirubin	0.011763	1.000000	0.874618	0.206669	0.214065	0.237831	-0.008099
Direct_Bilirubin	0.007529	0.874618	1.000000	0.234939	0.233894	0.257544	-0.000139
Alkaline_Phosphotase	0.080425	0.206669	0.234939	1.000000	0.125680	0.167196	-0.028514
Alamine_Aminotransferase	-0.086883	0.214065	0.233894	0.125680	1.000000	0.791966	-0.042518
Aspartate_Aminotransferase	-0.019910	0.237831	0.257544	0.167196	0.791966	1.000000	-0.025645
Total_Protiens	-0.187461	-0.008099	-0.000139	-0.028514	-0.042518	-0.025645	1.000000
Albumin	-0.265924	-0.222250	-0.228531	-0.165453	-0.029742	-0.085290	0.784053
Albumin_and_Globulin_Ratio	-0.216089	-0.206159	-0.200004	-0.233960	-0.002374	-0.070024	0.233904
Gender_Female	-0.056560	-0.089291	-0.100436	0.027496	-0.082332	-0.080336	0.089121
Gender_Male	0.056560	0.089291	0.100436	-0.027496	0.082332	0.080336	-0.089121

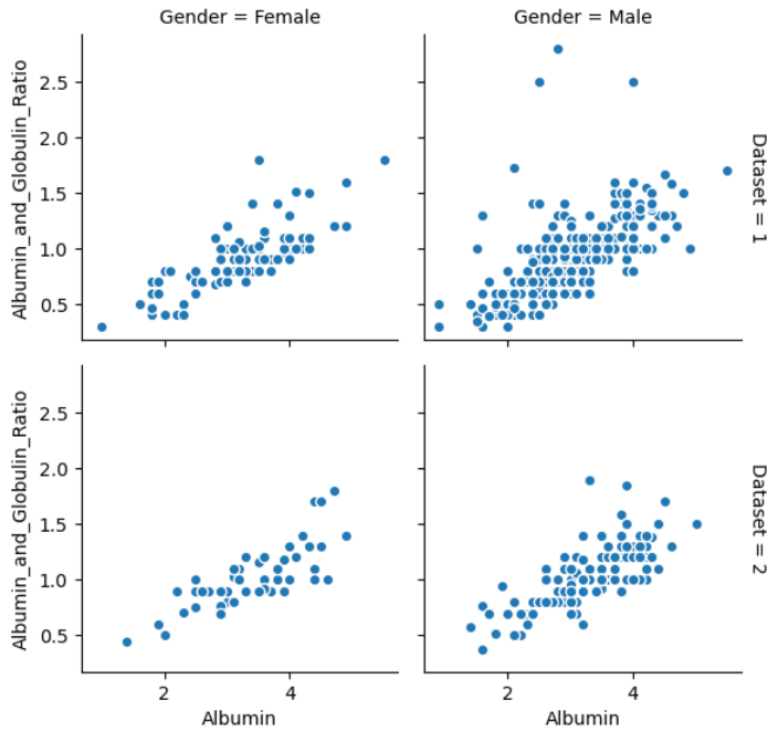
High Correlation between:

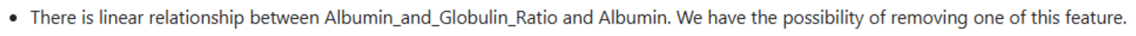
- Total_Protiens & Albumin
- Alamine_Aminotransferase & Aspartate_Aminotransferase
- Direct_Bilirubin & Total_Bilirubin
- There is some correlation between Albumin_and_Globulin_Ratio and Albumin. But its not as high as Total_Protiens & Albumin

Multivariate Analysis:



- There is linear relationship between Total_Protiens and Albumin and the gender. We have the possibility of removing one of this feature.



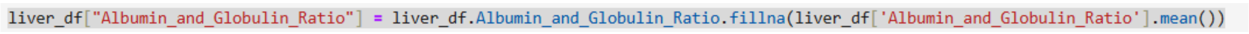


- There is linear relationship between Albumin_and_Globulin_Ratio and Albumin. We have the possibility of removing one of this feature.

Anomalies:

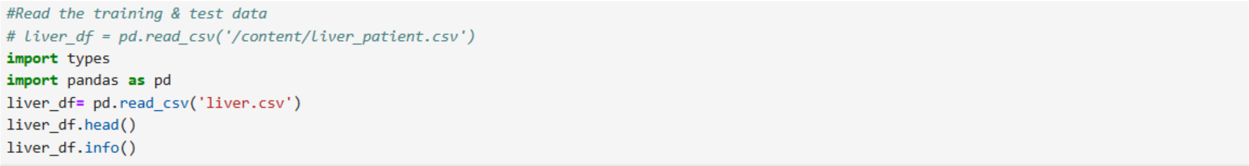
Replaced Null values with mean value.

Removed duplicate values.



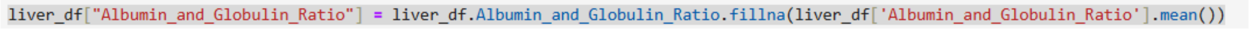
```
liver_df["Albumin_and_Globulin_Ratio"] = liver_df.Albumin_and_Globulin_Ratio.fillna(liver_df['Albumin_and_Globulin_Ratio'].mean())
```

Loading Data:



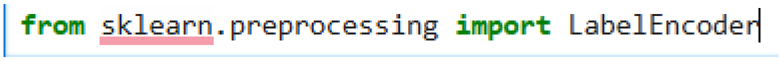
```
#Read the training & test data
# liver_df = pd.read_csv('/content/liver_patient.csv')
import types
import pandas as pd
liver_df= pd.read_csv('liver.csv')
liver_df.head()
liver_df.info()
```

Handling Missing Data:

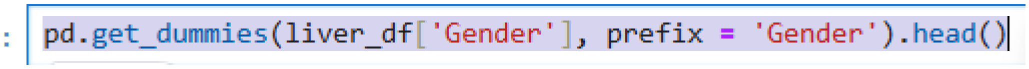


```
liver_df["Albumin_and_Globulin_Ratio"] = liver_df.Albumin_and_Globulin_Ratio.fillna(liver_df['Albumin_and_Globulin_Ratio'].mean())
```

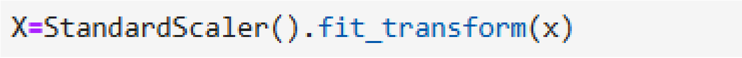
Data Transformation:



```
from sklearn.preprocessing import LabelEncoder
```

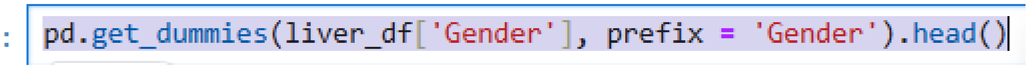


```
: pd.get_dummies(liver_df['Gender'], prefix = 'Gender').head()
```

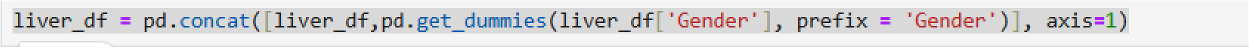


```
X=StandardScaler().fit_transform(x)
```

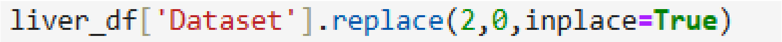
Feature Engineering:



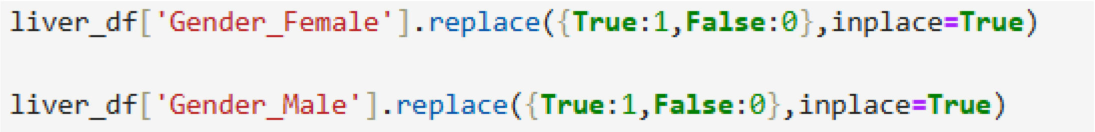
```
: pd.get_dummies(liver_df['Gender'], prefix = 'Gender').head()
```



```
liver_df = pd.concat([liver_df,pd.get_dummies(liver_df['Gender'], prefix = 'Gender')], axis=1)
```



```
liver_df['Dataset'].replace(2,0,inplace=True)
```



```
liver_df['Gender_Female'].replace({True:1,False:0},inplace=True)

liver_df['Gender_Male'].replace({True:1,False:0},inplace=True)
```

Feature Selection report:

Feature	Description	Selected (Yes/No)	Reasoning
F-score	The harmonic mean of precision and recall. It provides a balanced measure of a model's performance.	Yes	It helps optimize the decision threshold to strike the best balance between correctly identifying diseased individuals and minimizing false positives and negatives.
Total_Protiens & Albumin, gender **Alkaline_Phosphotase, Total_Bilirubin,**	From the jointplots and scatterplots, we find direct relationship with the feature	Yes	There is some correlation between Albumin_and_Globulin_Ratio and Albumin. But its not as high as Total_Protiens & Albumin. Normally distributed data
Direct_billirubin, Aspartate_aminotransferase	The bilirubin that your liver has previously digested is this one. It passes through the small intestine and liver before being eliminated in your urine. The characteristic yellow color of urine is caused by direct bilirubin.	NO	There is high correlation between direct and total billirubin. So dropped the direct billirubin for better model training

	Aspartate transferase (AST) is an enzyme that’s found in your liver, heart, pancreas, muscles and other tissues in your body. An AST blood test is often included in a liver panel and comprehensive metabolic panel, and healthcare providers most often use it to help assess your liver health		
AUC-ROC	The curve plots the true positive rate against the false positive rate at various classification thresholds	Yes	Less sensitive to class imbalance compared to accuracy. Medical applications where the trade-off between sensitivity and specificity must be carefully balanced.

Model Selection Report:

Model-1(Logistic regression)

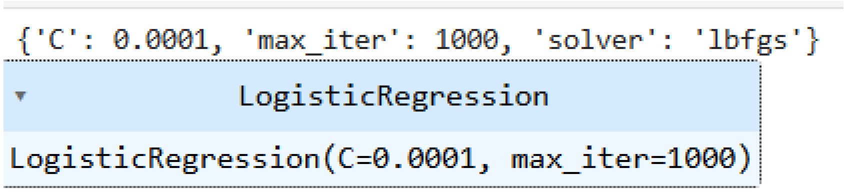
Description:

One of the simplest and best ML classification algorithms is logistic regression. LR is a supervised ML binary classification algorithm widely used in most applications. It operates on a categorical dependent variable, the result can be a discrete or binary categorical variable 0 or 1.

Logistic sigmoid function:

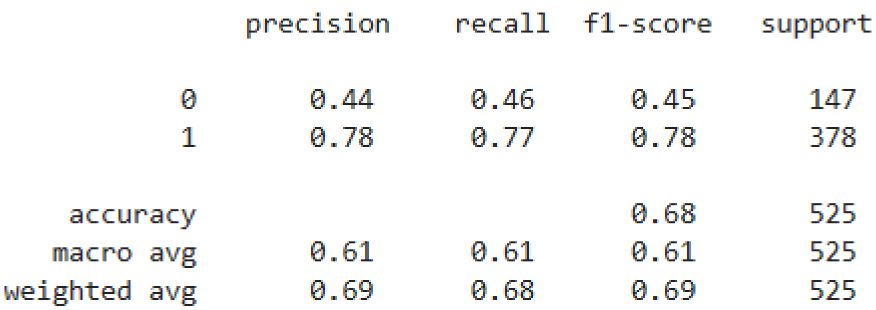
$$prob(Y=1)=\frac{e^{z}}{1+e^{z}}$$

Hyperparameters:

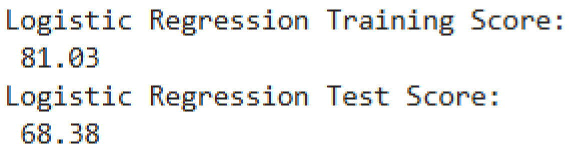


```
{'C': 0.0001, 'max_iter': 1000, 'solver': 'lbfgs'}
▾            LogisticRegression
LogisticRegression(C=0.0001, max_iter=1000)
```

Accuracy, Precision and Recall:



```
              precision    recall  f1-score   support

           0       0.44      0.46      0.45       147
           1       0.78      0.77      0.78       378

    accuracy                           0.68       525
   macro avg       0.61      0.61      0.61       525
weighted avg       0.69      0.68      0.69       525
```



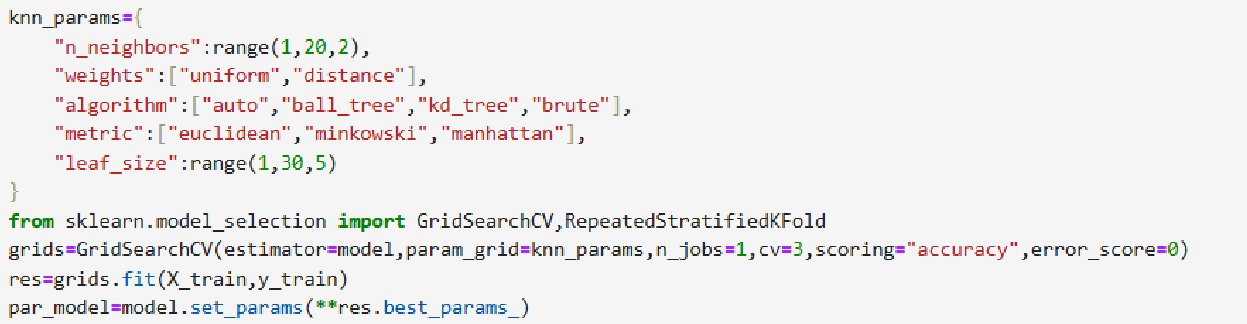
```
 Logistic Regression Training Score:
  81.03
 Logistic Regression Test Score:
  68.38
```

Model-2(KNN):

Description:

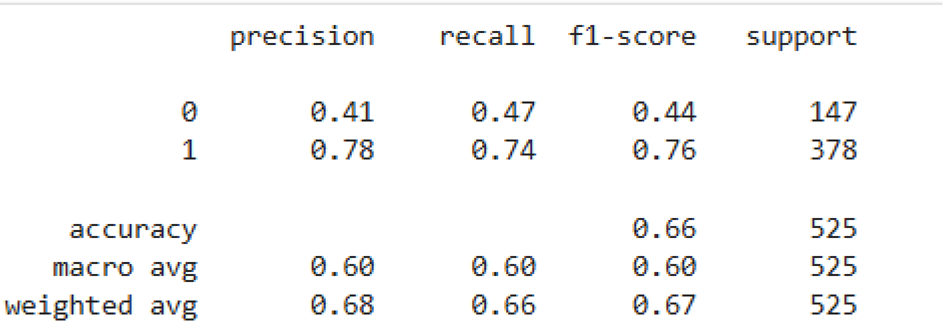
Using supervised machine learning, the K-Nearest Neighbors (KNN) technique is used to solve regression and classification issues.

Hyperparameters:



```
knn_params={
    "n_neighbors":range(1,20,2),
    "weights":["uniform","distance"],
    "algorithm":["auto","ball_tree","kd_tree","brute"],
    "metric":["euclidean","minkowski","manhattan"],
    "leaf_size":range(1,30,5)
}
from sklearn.model_selection import GridSearchCV,RepeatedStratifiedKFold
grids=GridSearchCV(estimator=model,param_grid=knn_params,n_jobs=1,cv=3,scoring="accuracy",error_score=0)
res=grids.fit(X_train,y_train)
par_model=model.set_params(**res.best_params_)
```

Accuracy, Precision and Recall:



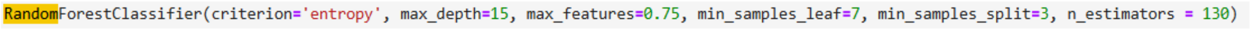
```
              precision    recall  f1-score   support

           0       0.41      0.47      0.44       147
           1       0.78      0.74      0.76       378

    accuracy                           0.66       525
   macro avg       0.60      0.60      0.60       525
weighted avg       0.68      0.66      0.67       525
```

Model-3(Random Forest):

Description:

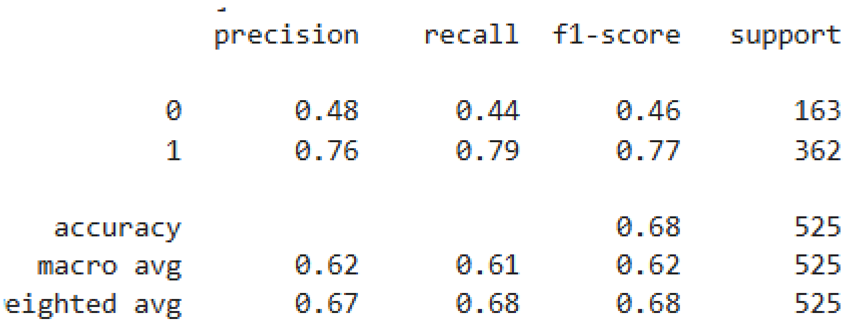
A random forest is a meta estimator that employs averaging to increase prediction accuracy and manage over-fitting after fitting several decision tree classifiers on different subsamples of the dataset. The best split strategy, or passing splitter="best" to the underlying DecisionTreeRegressor, is employed by the trees in the forest. If bootstrap=True (the default), the sub-sample size is managed using the max_samples argument; if not, each tree is constructed using the entire dataset.

Hyperparameters:



```
RandomForestClassifier(criterion='entropy', max_depth=15, max_features=0.75, min_samples_leaf=7, min_samples_split=3, n_estimators = 130)
```

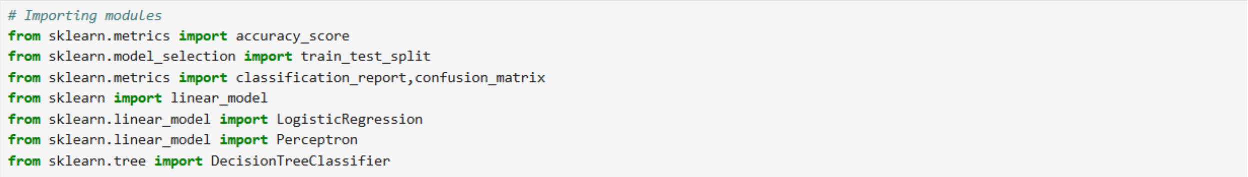
Accuracy, Precision and Recall:



```
              precision    recall  f1-score   support

           0       0.48      0.44      0.46       163
           1       0.76      0.79      0.77       362

    accuracy                           0.68       525
   macro avg       0.62      0.61      0.62       525
eighted avg       0.67      0.68      0.68       525
```

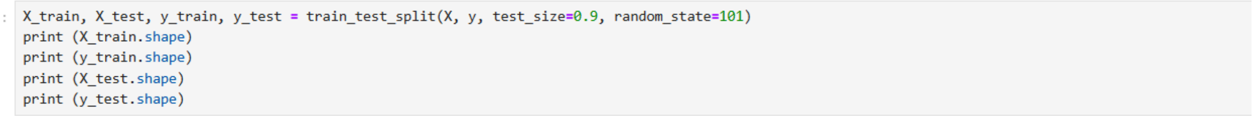
Initial Model Training Code, Model Validation and Evaluation Report

Initial Model Training Code:



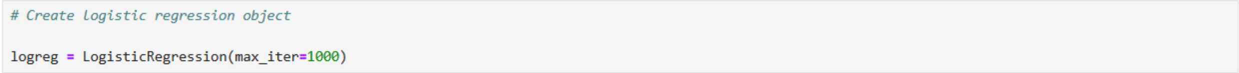
```
# Importing modules
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
from sklearn import linear_model
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import Perceptron
from sklearn.tree import DecisionTreeClassifier
```

Train Test Split:

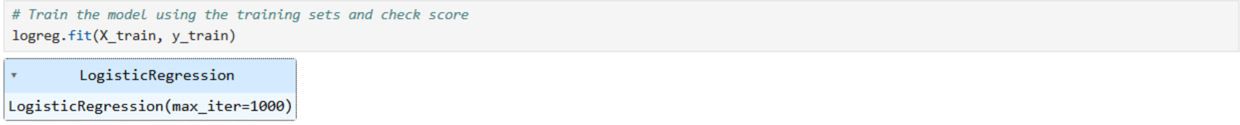


```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.9, random_state=101)
print (X_train.shape)
print (y_train.shape)
print (X_test.shape)
print (y_test.shape)
```

Logistic Regression:

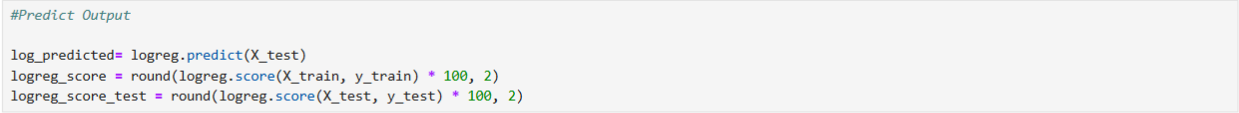


```
# Create logistic regression object

logreg = LogisticRegression(max_iter=1000)
```



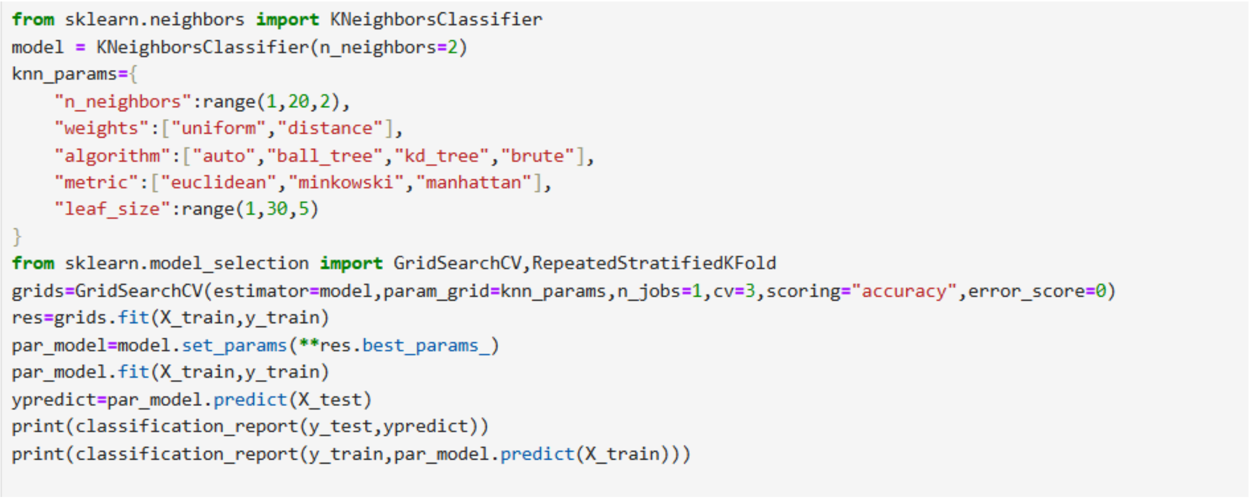
```
# Train the model using the training sets and check score
logreg.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression(max_iter=1000)
```



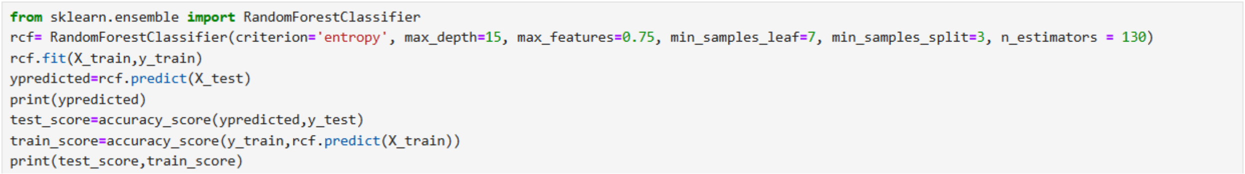
```
#Predict Output

log_predicted= logreg.predict(X_test)
logreg_score = round(logreg.score(X_train, y_train) * 100, 2)
logreg_score_test = round(logreg.score(X_test, y_test) * 100, 2)
```

KNN:



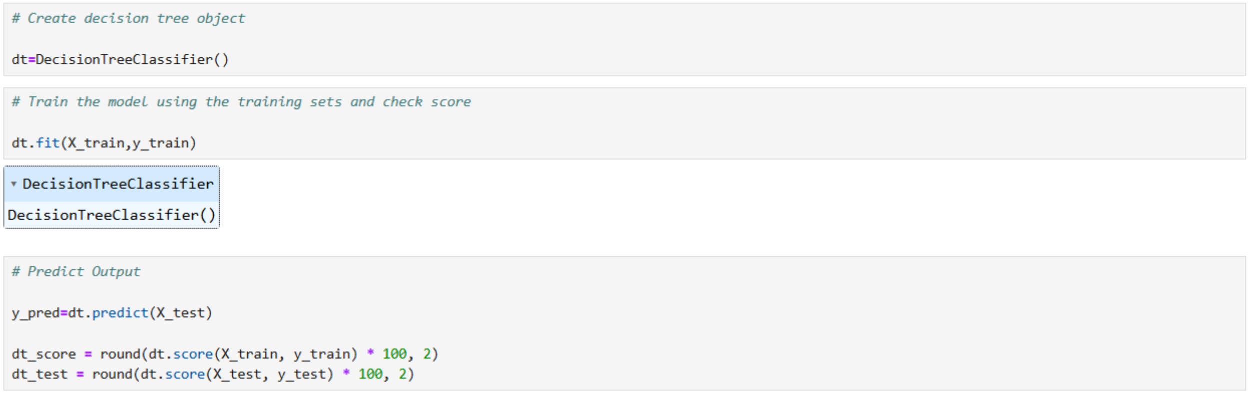
```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=2)
knn_params={
    "n_neighbors":range(1,20,2),
    "weights":["uniform","distance"],
    "algorithm":["auto","ball_tree","kd_tree","brute"],
    "metric":["euclidean","minkowski","manhattan"],
    "leaf_size":range(1,30,5)
}
from sklearn.model_selection import GridSearchCV,RepeatedStratifiedKFold
grids=GridSearchCV(estimator=model,param_grid=knn_params,n_jobs=1,cv=3,scoring="accuracy",error_score=0)
res=grids.fit(X_train,y_train)
par_model=model.set_params(**res.best_params_)
par_model.fit(X_train,y_train)
ypredict=par_model.predict(X_test)
print(classification_report(y_test,ypredict))
print(classification_report(y_train,par_model.predict(X_train)))
```

Random Forest classifier:



```
from sklearn.ensemble import RandomForestClassifier
rcf= RandomForestClassifier(criterion='entropy', max_depth=15, max_features=0.75, min_samples_leaf=7, min_samples_split=3, n_estimators = 130)
rcf.fit(X_train,y_train)
ypredicted=rcf.predict(X_test)
print(ypredicted)
test_score=accuracy_score(ypredicted,y_test)
train_score=accuracy_score(y_train,rcf.predict(X_train))
print(test_score,train_score)
```

Decision tree classifier:



```
# Create decision tree object

dt=DecisionTreeClassifier()
```

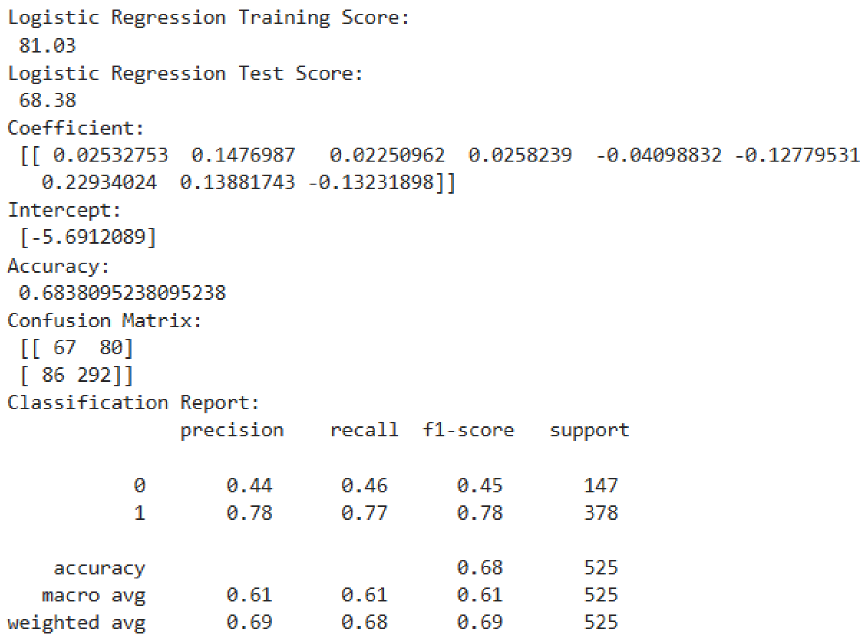
```
# Train the model using the training sets and check score

dt.fit(X_train,y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
# Predict Output

y_pred=dt.predict(X_test)

dt_score = round(dt.score(X_train, y_train) * 100, 2)
dt_test = round(dt.score(X_test, y_test) * 100, 2)
```

Model Validation and Evaluation Report:

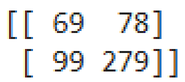
Model 1(Logistic Regression):



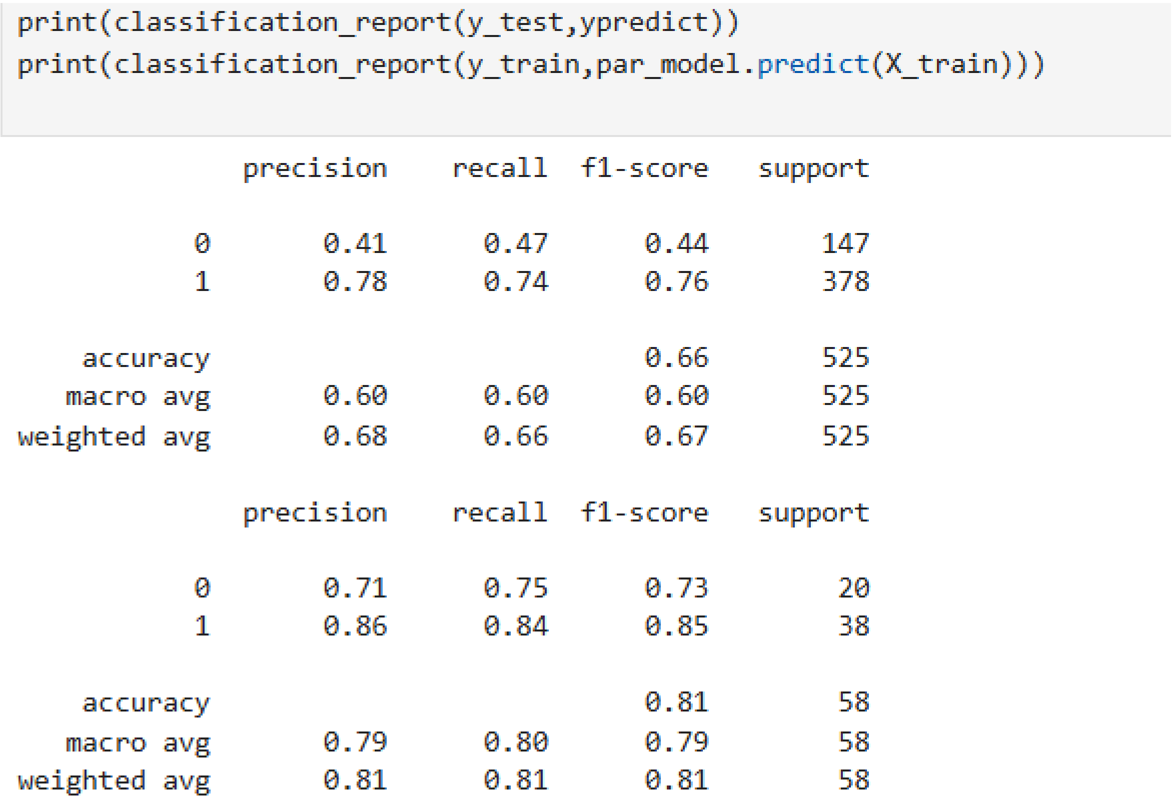
```
Logistic Regression Training Score:
 81.03
Logistic Regression Test Score:
 68.38
Coefficient:
 [[ 0.02532753  0.1476987   0.02250962  0.0258239  -0.04098832 -0.12779531
   0.22934024  0.13881743 -0.13231898]]
Intercept:
 [-5.6912089]
Accuracy:
 0.6838095238095238
Confusion Matrix:
 [[ 67  80]
 [ 86 292]]
Classification Report:
               precision    recall  f1-score   support

           0       0.44      0.46      0.45       147
           1       0.78      0.77      0.78       378

    accuracy                           0.68       525
   macro avg       0.61      0.61      0.61       525
weighted avg       0.69      0.68      0.69       525
```

Model-2(KNN):

Confusion matrix:



```
[[ 69  78]
 [ 99 279]]
```

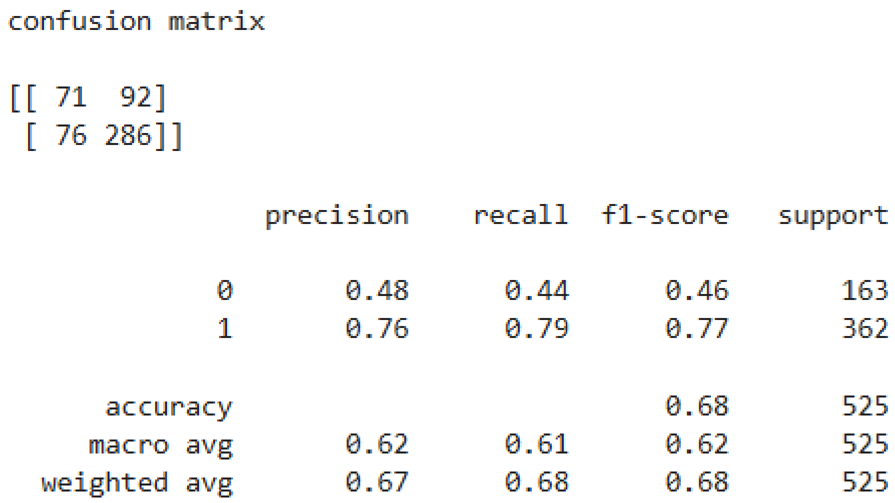


```
print(classification_report(y_test,ypredict))
print(classification_report(y_train,par_model.predict(X_train)))
```

```
              precision    recall  f1-score   support

           0       0.41      0.47      0.44       147
           1       0.78      0.74      0.76       378

    accuracy                           0.66       525
   macro avg       0.60      0.60      0.60       525
weighted avg       0.68      0.66      0.67       525

              precision    recall  f1-score   support

           0       0.71      0.75      0.73        20
           1       0.86      0.84      0.85        38

    accuracy                           0.81        58
   macro avg       0.79      0.80      0.79        58
weighted avg       0.81      0.81      0.81        58
```

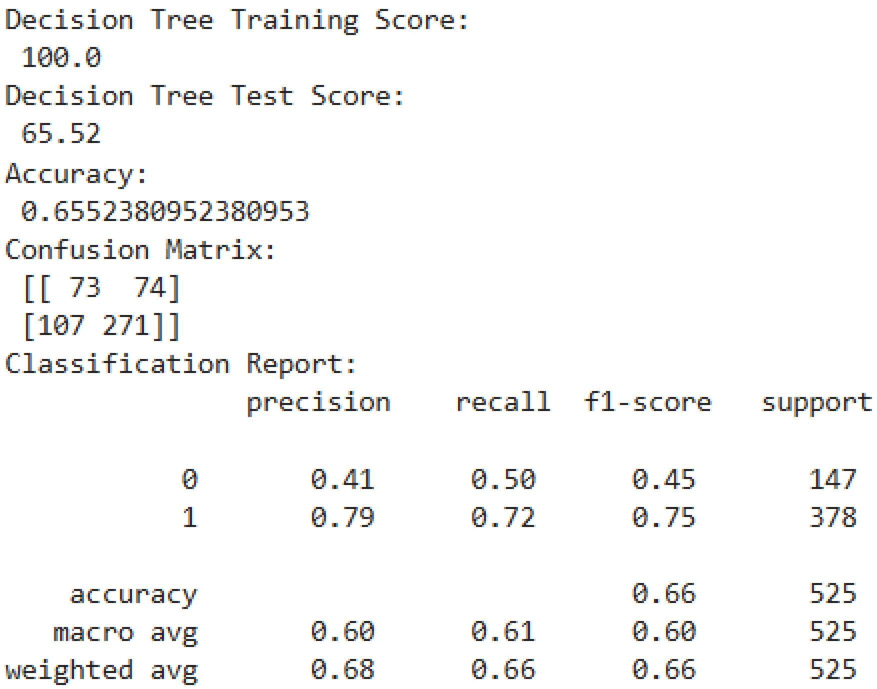
Model-3(Random forest Classifier):

Test Data:



```
confusion matrix

[[ 71  92]
 [ 76 286]]
              precision    recall  f1-score   support

           0       0.48      0.44      0.46       163
           1       0.76      0.79      0.77       362

    accuracy                           0.68       525
   macro avg       0.62      0.61      0.62       525
weighted avg       0.67      0.68      0.68       525
```

Model-4(Decision Tree Classifier):



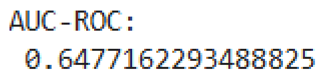
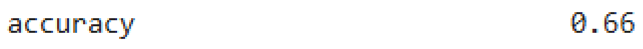
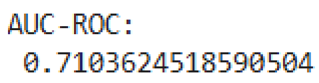
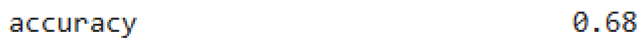
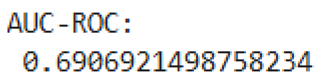
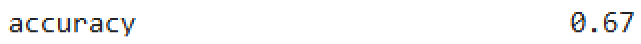
```
Decision Tree Training Score:
 100.0
Decision Tree Test Score:
 65.52
Accuracy:
 0.6552380952380953
Confusion Matrix:
 [[ 73  74]
 [107 271]]
Classification Report:
               precision    recall  f1-score   support

           0       0.41      0.50      0.45       147
           1       0.79      0.72      0.75       378

    accuracy                           0.66       525
   macro avg       0.60      0.61      0.60       525
weighted avg       0.68      0.66      0.66       525
```

Hyperparameter tuning:

Model	**Tuned Hyperparameters**	**Optimal Values**
KNN	"n_neighbors", "weights”, "algorithm”, "metric”, "leaf_size”	3,uniform,auto,euclidean,1
Random forest classifier	Criterion, max_depth, max_features, min_samples_leaf, min_samples_split, n_estimators	Entropy,15,0,75,7,3,130
Logistic regression	‘C’, ‘max-iter’, ‘solver’	0.0001, 1000,lbfgs

Performance Metrics Comparison Report

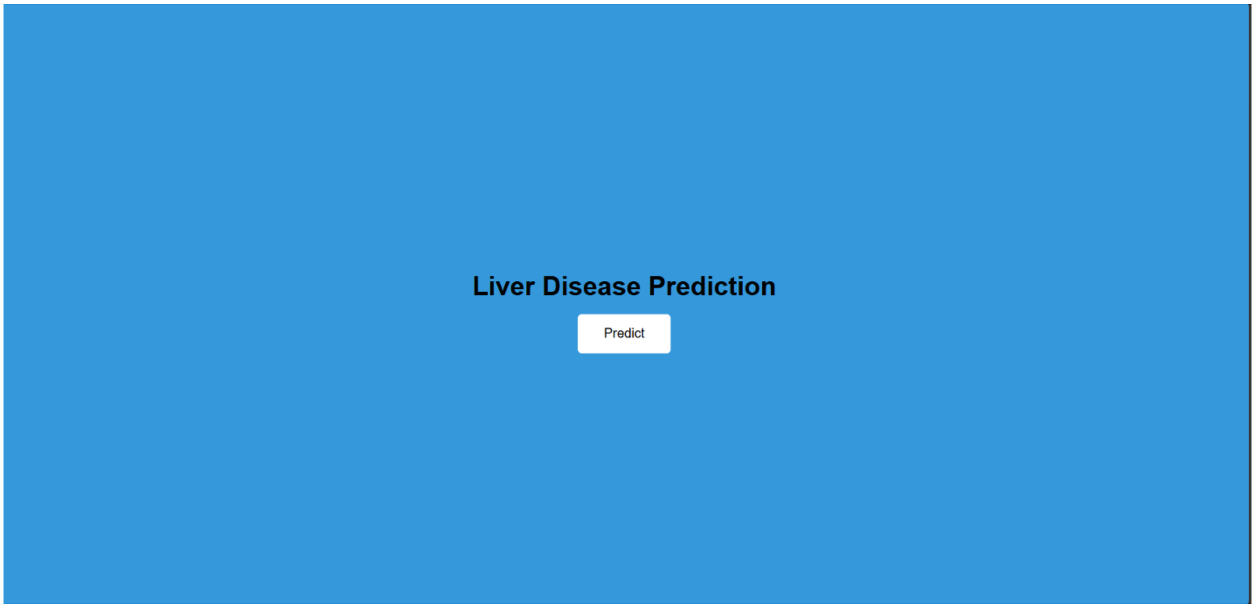
Model	**Baseline Metric**	**Optimized Metric**

KNN	 AUC-ROC: 0.6477162293488825	 accuracy 0.66
Random forest classifier	 AUC-ROC: 0.7103624518590504	 accuracy 0.68
Logistic regression	 AUC-ROC: 0.6906921498758234	 accuracy 0.67

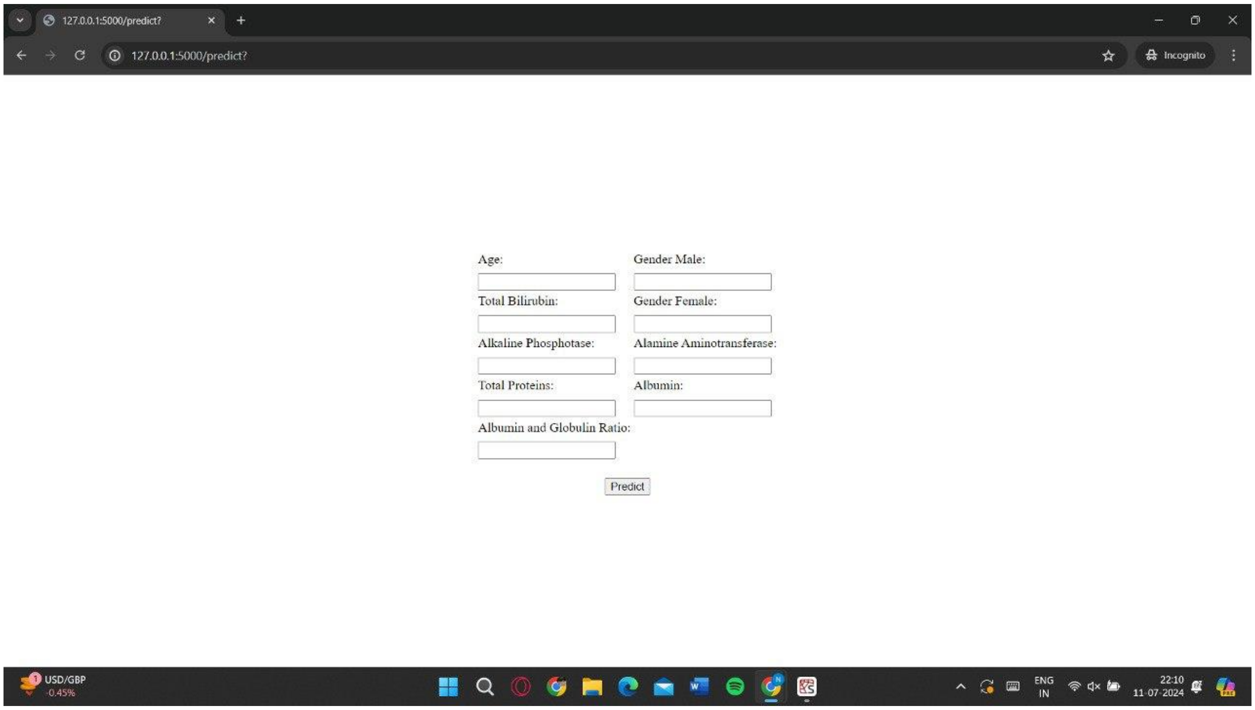
Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Logistic regression	Has high accuracy when compared with other two models High recall score and high precision score The model strikes a balance between interpretability and good results

Output Screenshots



Liver Disease Prediction



Age:

Gender Male:

Total Bilirubin:

Gender Female:

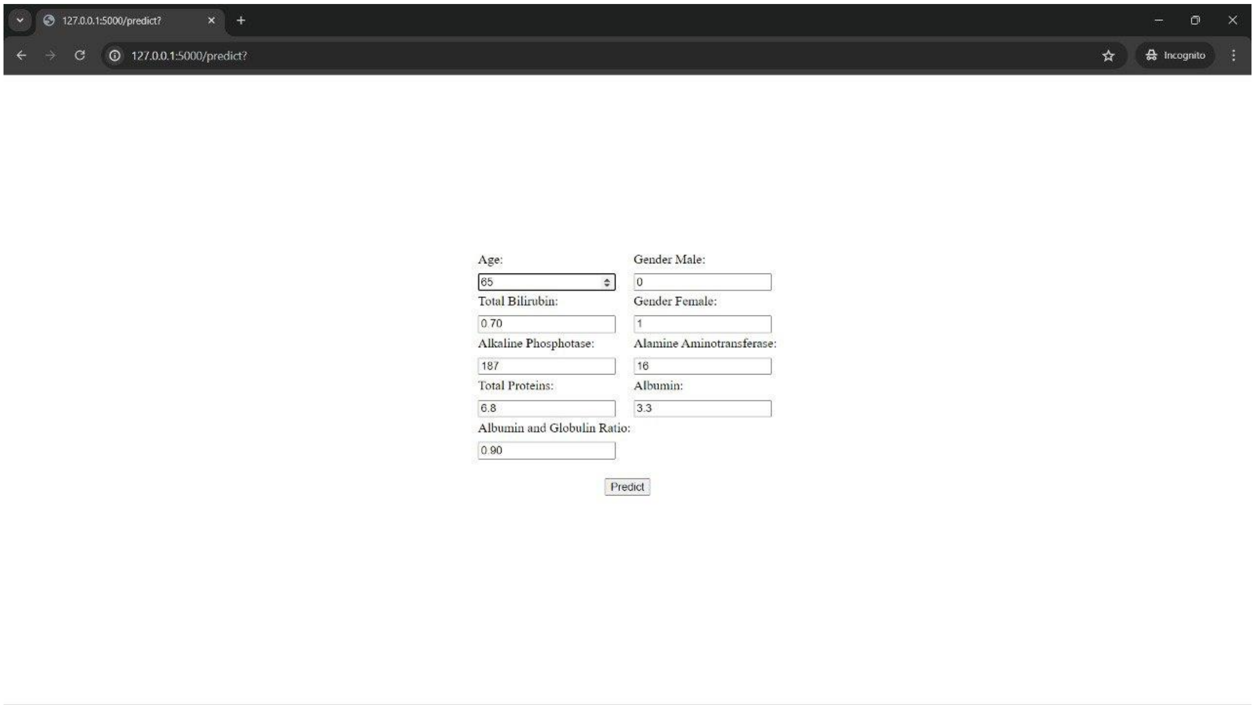
Alkaline Phosphotase:

Alamine Aminotransferase:

Total Proteins:

Albumin:

Albumin and Globulin Ratio:



Age: 65

Gender Male: 0

Total Bilirubin: 0.70

Gender Female: 1

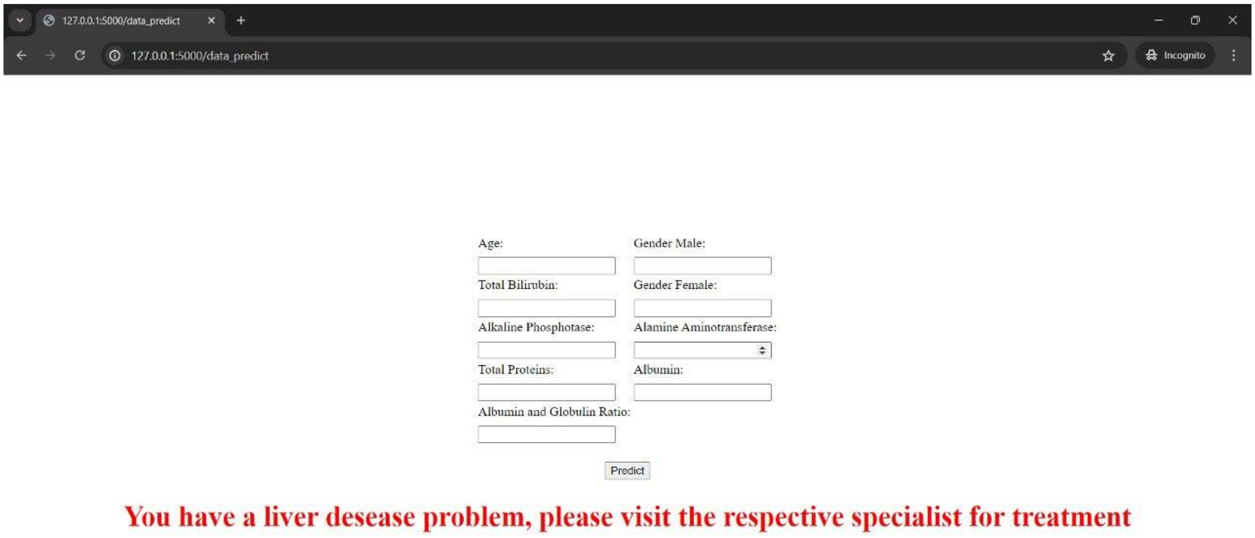
Alkaline Phosphotase: 187

Alamine Aminotransferase: 16

Total Proteins: 6.8

Albumin: 3.3

Albumin and Globulin Ratio: 0.90



Age:

Gender Male:

Total Bilirubin:

Gender Female:

Alkaline Phosphotase:

Alamine Aminotransferase:

Total Proteins:

Albumin:

Albumin and Globulin Ratio:

You have a liver desease problem, please visit the respective specialist for treatment

---

Advantages and Disadvantages:

1. Accuracy and Efficiency:

- Machine learning classifiers can analyze large volumes of patient data efficiently, potentially identifying patterns and relationships that are not immediately apparent through traditional statistical methods.

- This accuracy can lead to earlier detection of liver disease or risk factors, allowing for timely intervention and improved patient outcomes.

2. Personalized Medicine:

- By leveraging patient records which include demographic information, medical history, and biomarkers, machine learning models can tailor predictions and treatments to individual patients.

- This personalized approach enhances healthcare delivery by optimizing treatment plans and resource allocation based on patient-specific risks.

3. Integration with Electronic Health Records (EHRs):

- Many healthcare systems now utilize electronic health records (EHRs) that contain a wealth of patient information.

- Machine learning classifiers can seamlessly integrate with EHR systems, allowing for real-time analysis and decision support in clinical settings.

4. Feature Selection and Interpretability:

- Machine learning models can automatically select relevant features from patient records, identifying key predictors of liver disease.

- Techniques such as feature importance ranking in Random Forests or coefficient interpretation in Logistic Regression provide insights into which patient variables are most influential.

5. Scalability and Automation:

- Once trained, machine learning models can automate the prediction process, reducing the burden on healthcare professionals and enabling scalable deployment across different healthcare facilities.

Disadvantages:

1. Data Quality and Preprocessing:

- Patient records can be incomplete, inconsistent, or contain errors, which can affect the performance of machine learning models.

- Extensive data preprocessing steps, such as handling missing values and standardizing data formats, are often required to ensure data quality.

2. Interpretability and Trust:

- Complex machine learning models (e.g., deep learning) may provide high accuracy but lack interpretability, making it challenging to understand how predictions are made.

- Clinicians may be reluctant to trust predictions from black-box models without clear explanations or validation against clinical guidelines.

3. Overfitting and Generalization:

- Machine learning models, especially when trained on large and diverse patient datasets, may overfit to noise or specific characteristics of the training data.

- Ensuring models generalize well to new patient data from different demographics or geographic regions is crucial for their clinical applicability.

4. Ethical and Legal Considerations:

- Predictive models based on patient records raise ethical concerns related to patient privacy, informed consent, and potential biases in the data.

- Adhering to regulations such as GDPR (General Data Protection Regulation) or HIPAA (Health Insurance Portability and Accountability Act) is essential to protect patient confidentiality and rights.

5. Clinical Integration and Validation:

- Successfully integrating machine learning predictions into clinical workflows requires collaboration between data scientists, clinicians, and healthcare administrators.

- Validating model predictions against real-world outcomes and ensuring they align with clinical guidelines are critical steps to gaining acceptance and adoption in healthcare settings.

Conclusion and Future scope:

The conclusion and future work for predicting liver diseases using logistic regression and decision tree classifier can be summarized as follows: Using logistic regression and decision tree classifier models, it is possible to predict the likelihood of liver diseases based on various risk factors and patient data. These machine learning algorithms can help identify high-risk individuals and assist healthcare professionals in making informed decisions for early diagnosis and intervention.

From the study it is observed that as the training set ratio increases, the model's performance on the training data generally improves, as expected. This is indicated by the

increasing trend in training accuracy. However, the performance on the testing data may not necessarily follow the same pattern. It may peak at a certain point and then start to decrease due to overfitting.
A higher value of ccp_alpha increases the regularization strength, leading to simpler trees. In this case, it is set to 0.04, indicating a moderate level of regularization.
Managing the tree's depth aids in avoiding overfitting. While a deeper tree may be able to identify more complex patterns in the training set, overfitting could result from the tree learning to remember noise. The tree more effectively generalizes to unknown data by restricting the depth.

Feature Selection and Model Optimization: Future work should focus on selecting the most relevant features that contribute significantly to the prediction of
liver diseases. This can be achieved through feature selection techniques and model optimization. By reducing the number of input features, the models can be made more efficient and accurate.

The future work includes:
Ensemble Methods and Hybrid Models:
Incorporating ensemble methods and hybrid models can further improve the predictive power of the logistic regression and decision tree classifier models. By combining multiple models or algorithms, it is possible to achieve better accuracy and robustness in disease prediction.

Incorporating Advanced Techniques:

Exploring advanced machine learning techniques such as deep learning, random forests, and gradient boosting can lead to better disease prediction models. These techniques can capture complex patterns and relationships in the data, which may not be evident in logistic regression and decision tree classifier models.

Handling Imbalanced Datasets:

As liver diseases are relatively rare compared to other health conditions, datasets may be imbalanced.

Handling Imbalanced Datasets:

Future research should address techniques to handle imbalanced datasets,such as oversampling, undersampling, and synthetic minority oversampling technique (SMOTE), to improve the models' performance in predicting rare disease cases.

Multi-class Prediction and Interpretability:

Extending the current binary classification models to multi-class prediction can help identify various liver disease types. Additionally, ensuring the models' interpretability will allow healthcare professionals to understand the factors contributing to the disease prediction, leading to better decision-making and patient care.

In conclusion, the prediction of liver diseases using logistic regression and decision tree classifier models has shown promising results. Further research and development in feature selection, model optimization, advanced techniques, real-world implementation, handling imbalanced datasets, and multi-class prediction will contribute to more accurate and reliable disease prediction models in the future.