Capstone Project

Classifying Liver Disease dataset

Machine Learning Engineer Nanodegree

Ragala Sreekala

February 5th, 2019

Project Overview :

Problems with liver patients are not easily discovered in an early stage as it will be functioning normally even when it is partially damaged. An early diagnosis of liver problems will increase patient's survival rate. Liver failures are at high rate of risk among Indians. It is expected that by 2025 India may become the World Capital for Liver Diseases. The widespread occurrence of liver infection in India is contributed due to deskbound lifestyle, increased alcohol consumption and smoking. There are about 100 types of liver infections.

1. A patient going to a doctor with certain symptoms.
2. The doctor recommending certain tests like blood test, urine test etc depending on the symptoms.
3. The patient taking the aforementioned tests in an analysis lab.
4. The patient taking the reports back to the reports back to the hospital, where they are examined the disease is identified

Reference Link: https://www.irjet.net/archives/V5/i4/IRJET-V5I4896.pdf

Problem Statement:

Given a dataset containing various attributes of 583 Indian patients, define classification algorithms. To apply different classification algorithms on the Indian patient liver disease dataset than choose the best algorithms based on the accuracy which can identify whether a person is suffering from liver disease or not.

The dataset that I am working is downloaded from

https://www.kaggle.com/sharadhiv/indian-liver-patient-dataset-ilpd

The number of instances are 583. It is a multivariate data set, contain 10 variables that are age, gender, total Bilirubin, direct Bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkphos. All values are real integers. This data set contains 416 liver patient records and 167 non liver patient records. The data set was collected from north east of Andhra Pradesh, India.

Selector is a class label used to divide into groups(liver patient or not). This data set contains 441 male patient records and 142 female patient records. ‘is_ patient ‘ label '1' representing presence of disease and '2' representing absence of disease.

Strategy:

This seems to be a classic example of supervised learning. We have been provided with a fixed number of features for each data point, and our aim will be to train a variety of Supervised Learning algorithms on this data, so that when a new data point arises, our best performing classifier can be used to categorize the data point as a positive example or negative.

Metrics :

I want to use accuracy score as evaluation metric for prediction of liver disease.The performance of a model cannot be assessed by considering only the accuracy, because there is a possibility for misleading. Therefore this experiment considers the F1 score along with the accuracy for evaluation.. This is because depending on the context like severity of disease, sometimes it is more important that an algorithm does not wrongly predict a disease as a non-disease.

Thus, here we will use F-beta score as a performance metric, which is basically the weighted harmonic mean of precision and recall. Precision and Recall are defined as:

Precision=TP/ (TP+FP), Recall=TP/ (TP+FN), where

TP=True Positive

FP=False Positive

FN=False Negative

In the same vein, F-beta score is:

F-beta score = (1+β^2)*precision*recall/((β^ 2*precision)+recall)

We can use **F-beta score** as a metric that considers both precision and recall

Additionally, one more metric called as Receiver Operating Characteristics (ROC) curve will be used. It plots the curve of True Positive Rate and the False Positive Rate for a given algorithm, with a greater area under the curve indicating a better True Positive Rate for the same False Positive Rate, indicating the usefulness of the classifier.

Analysis

Exploring the Data :

The Indian liver patient dataset contains ten features as listed below:

1. Age

2. Gender

3. Total bilirubin

4. Direct bilirubin

5. Total proteins

6. Albumin

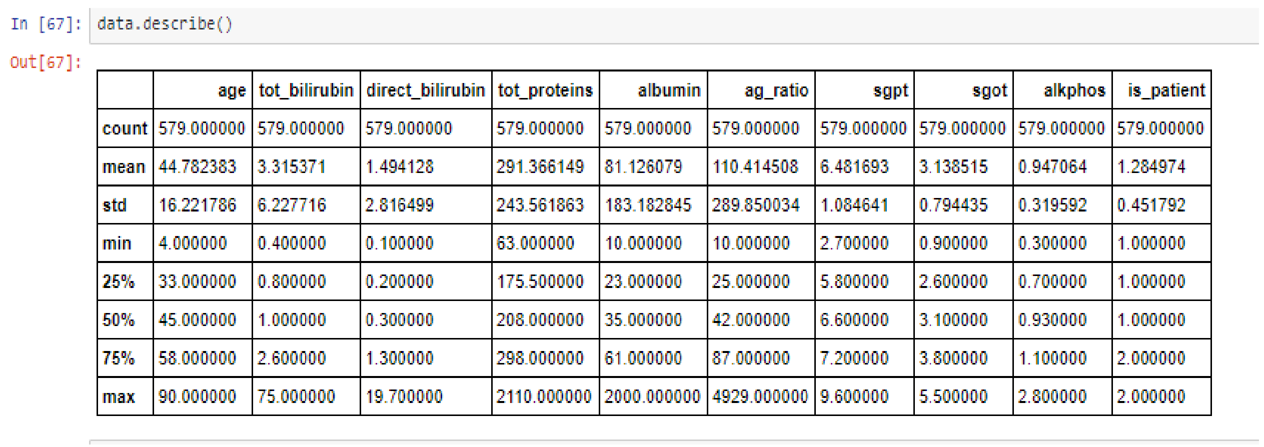
7. A/G ratio

8. SGPT

9. SGOT

10. Alkphos

All features, except Gender are real valued integers. The last column, is_patient, is the label with '1' representing presence of disease and '2' representing absence of disease. Total number of data points is 583, with 416 liver patient records and 167 non liver patient records. A brief description of dataset, including parameters like mean, min, max for each column is given below:

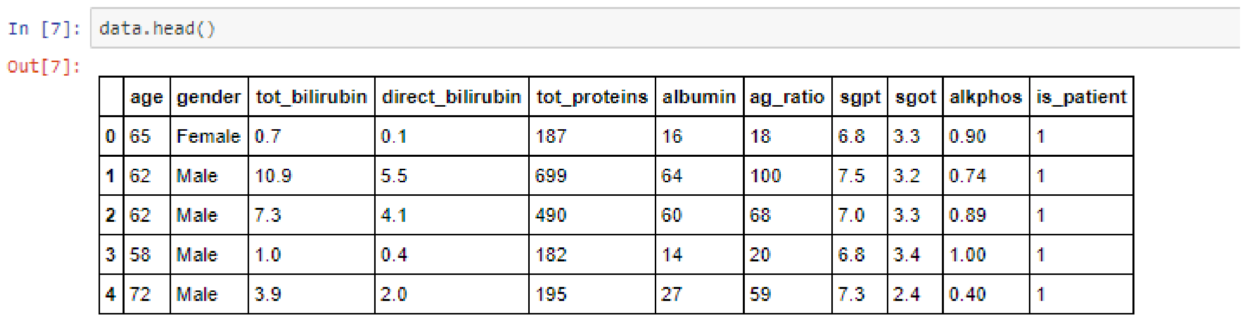


```
In [67]: data.describe()
```

Out[67]:

	age	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
count	579.000000	579.000000	579.000000	579.000000	579.000000	579.000000	579.000000	579.000000	579.000000	579.000000
mean	44.782383	3.315371	1.494128	291.366149	81.126079	110.414508	6.481693	3.138515	0.947064	1.284974
std	16.221786	6.227716	2.816499	243.561863	183.182845	289.850034	1.084641	0.794435	0.319592	0.451792
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000	1.000000
25%	33.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000	1.000000
50%	45.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.930000	1.000000
75%	58.000000	2.600000	1.300000	298.000000	61.000000	87.000000	7.200000	3.800000	1.100000	2.000000
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000	2.000000

A snapshot of the dataset containing first 5 rows is as follow:



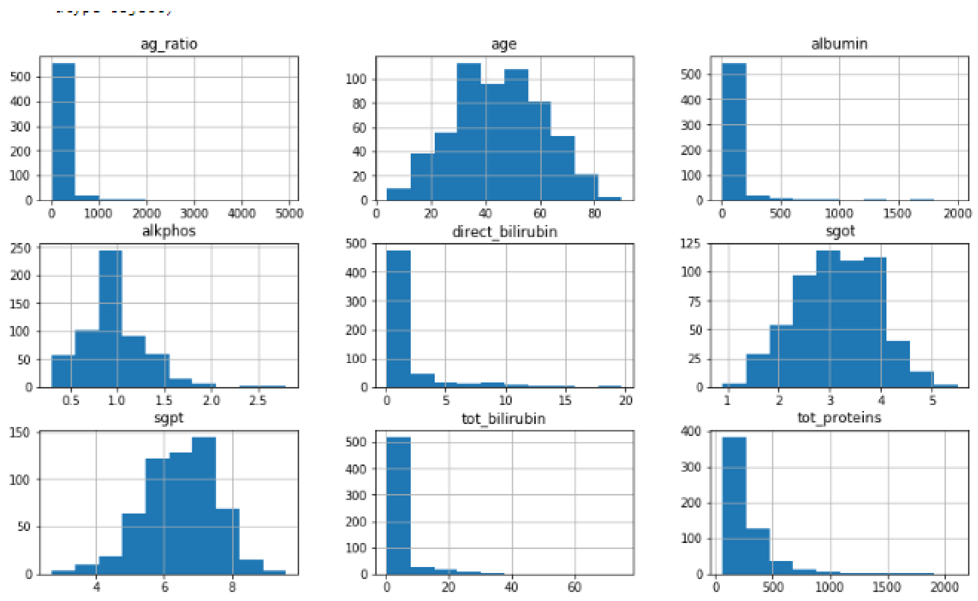
```
In [7]: data.head()
Out[7]:
```

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
0	65	Female	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	62	Male	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	58	Male	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	72	Male	3.9	2.0	195	27	59	7.3	2.4	0.40	1

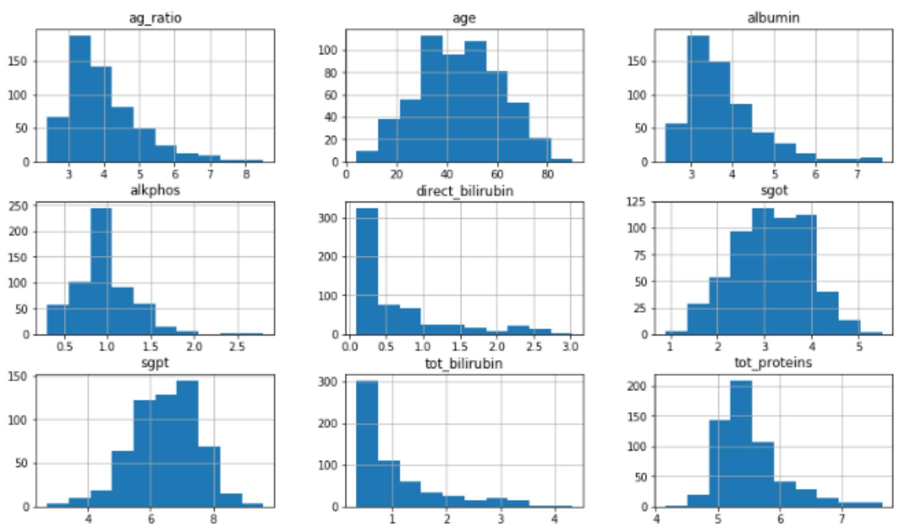
In the description of this dataset, it is observed that some values are ‘Null’ for the ‘Álkphos’ column. Accordingly, 4 rows containing those values are removed. Remaining number of rows in the dataset is 579.

Exploratory Visualization

After removing the column’ is_patient ‘from the dataset as it is the label, we display all features in a histogram format to check if any feature is skewed (contains a small number of outlier values).



Skewed features found are Albumin, Direct Bilirubin, A/G ratio, Total Bilirubin, Total Protein. On these, a log transformation is applied to reduce their range. Again, all the transformed features are shown in a histogram format:



Algorithms and Techniques:

Three supervised learning approaches are selected for this problem. Care is taken that all these approaches are fundamentally different from each other, so that we can cover as wide an umbrella as possible in term of possible approaches. For example- We will not select Random Forest and Ada Boost together as they come from the same family of 'ensemble' approaches. The choice of algorithms was influenced from these source:

https://stackoverflow.com/questions/2595176/which-machine-learning-classifier-to-choose-in-general

For each algorithm, we will try out different values of a few hyper parameters to arrive at the best possible classifier. This will be carried out with the help of grid search cross validation technique. For these dataset we apply the different supervised learning algorithms there are:

AdaBoostClassifier:

Adaboost is a boosting type ensemble learner. This method works by combining multiple individual "weak" learning hypotheses to create one strong model. Each weak hypothesis used is better at classifying the data than random chance. However, it's the combination of all of these independent weak learning hypotheses what makes the model more capable of predicting accurately on unseen data than each of the individual hypothesis would.

n_estimators: The maximum number of estimators at which boosting is terminated. In case of perfect fit, the learning procedure is stopped early.

learning_rate: Learning rate shrinks the contribution of each classifier by learning_rate. There is a trade-off between learning_rate and n_estimators.

Advantages:

- AdaBoost is the of the ensemble method. The AdaBoost is more robust than single estimators, have improved generalizability.
- Simple models can be combined to build a complex model, which is computationally fast

Disadvantages:

- If we have a biased underlying classifier, it will lead to a biased boosted model.
- AdaBoost can be sensitive to noisy data and outliers. In some problems, however, it can be less susceptible to the overfitting problem than most learning algorithms.

Support Vector Machine:

SVM aims to find an optimal hyperplane that separates the data into different classes, using a method called as kernel to project data points belonging to a particular class into different dimensions, so that a hyperplane can easily pass through and maintain the largest possible distance between itself and these data points.

Advantages:

- Performs well with high dimensional data. SVM's are very good when we have no idea on the data.
- Works well with even unstructured and semi structure data like text, Images.
- The kernel trick is strength of SVM. By using the kernel function to solve the complex problem.
- The SVM model have generalization in practice, the risk of overfitting is less in SVM.

Disadvantages:

- Choosing the good kernel function is not easy and it take long training time for large datasets.
- Difficult to understand and interpret the final model, variable weights and individual impact.

Decision Tree Classifier:

The decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions. Though a commonly used tool in data mining for deriving a strategy to reach a particular goal, its also widely used in machine learning.

Advantages:

- Able to handle categorical and numerical data.
- Doesn’t require much data pre-processing, and can handle data which hasn’t been normalized, or encoded for Machine Learning Suitability.
- Simple to understand ,visualize and interpret.

Disadvantages:

- Complex Decision Trees do not generalize well to the data and can result in overfitting.
- Unstable, as small variations in the data can result in a different decision tree.

Benchmark:

However the problem lies in finding a dataset where the results are given in such a fashion which is easily comparable with our classification values. In datasets it is intrinsically difficult to compare the scores given with our outputs. Therefore, we will use a simple algorithm like Logistic Regression as our benchmark model and try to improve upon its performance by using other algorithms like SVM, ensemble methods etc. If i classifies the data applying on different algorithms we got the accuracy_score with minimum 60% accuracy.

Data Preprocessing :

Some datasets contain irrelevant information, noise, missing values, and so on. These datasets should be handled properly to get a better result for the data mining process. Data preprocessing includes data cleaning, preparation, transformation, and dimensionality reduction, which convert the raw data into a form that is suitable for further processing.

As explained in the section 'Exploring the data', rows having 'Null' values were removed from the dataset. Thereafter, log transformation was applied to features which were showing a skewed pattern (Albumin, Direct Bilirubin, A/G ratio, Total Bilirubin, Total Protein).

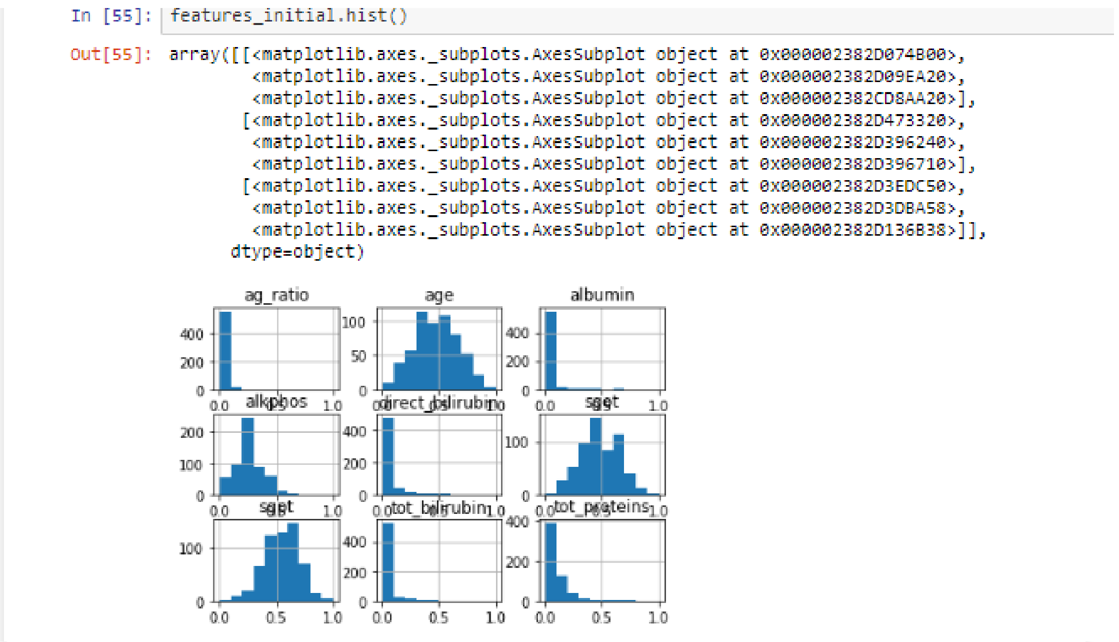
Thereafter, all columns in the dataset except ‘Gender’ are normalized. We use MinMaxScaler here as StandardScaler gives very low values here, with some in the order of 10^-16, which might be difficult to relate to and visualize.

Then we use pd.get_dummies() method to one-hot encode the feature ‘gender’ as well as the label ‘is_patient’ with the integer ‘1’ representing the presence of disease and ‘2’ representing the not presence of disease.

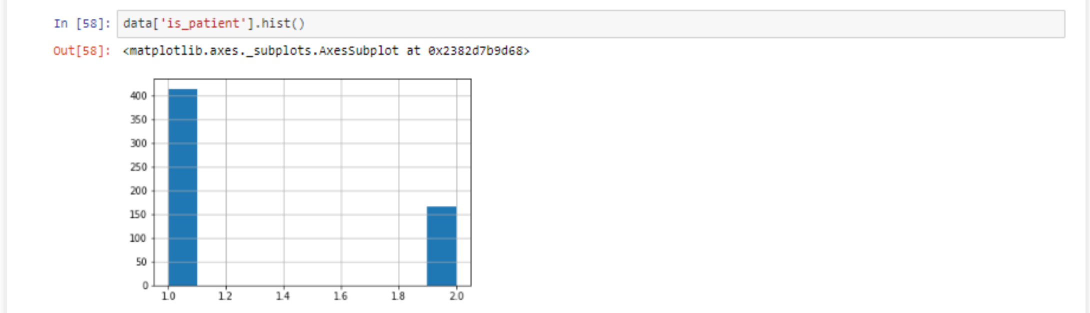
Data Visualization:

The visualization of the data, identifying the outliers, and skewed predictors. These tasks help to inspect the data and thereby spot the missing values and irrelevant information in the dataset. A data cleanup process is performed to handle these issues and to ensure data quality. Gaining a better understanding of the dataset helps to identify useful information and supports decision making.

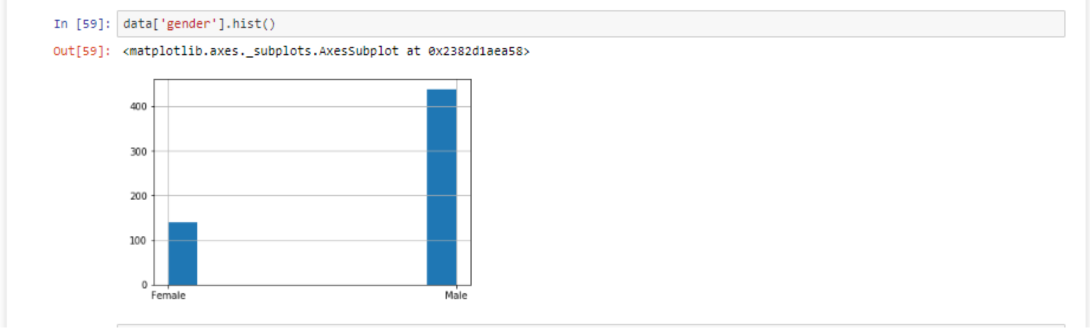
The Indian Liver Patient dataset consists of 579 records in which 416 are records of people with liver disease, and the remaining are records of people without any liver disease. The dataset has 10 features in which there is only one categorical data . The endmost column of the dataset represent the class in which each sample falls liver patient or not. A value of 1 indicates the person has liver disease and a 2 indicates the person does not have the disease. There is no missing value in the dataset.



```
In [55]: features_initial.hist()

Out[55]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000002382D074B00>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D09EA20>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382CD8AA20>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x000002382D473320>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D396240>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D396710>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x000002382D3EDC50>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D3DBA58>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D136B38>]],
               dtype=object)
```

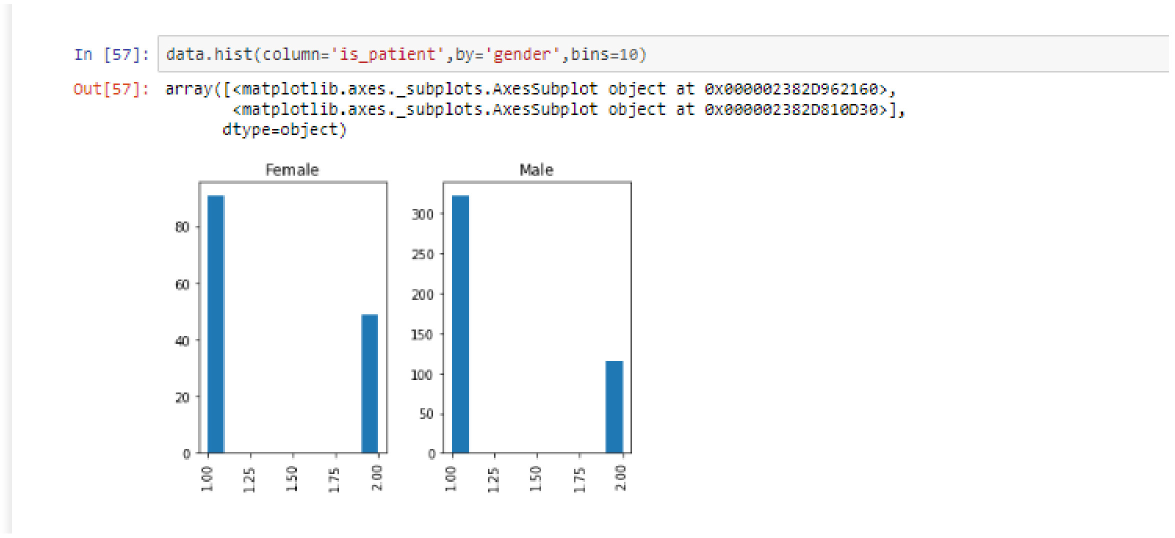


```
In [58]: data['is_patient'].hist()

Out[58]: <matplotlib.axes._subplots.AxesSubplot at 0x2382d7b9d68>
```



```
In [59]: data['gender'].hist()

Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x2382d1aea58>
```

visualization of the number of patients with liver dissease and patients with no liver disease and represents a visualization of the male and female population in the dataset.



```
In [57]: data.hist(column='is_patient',by='gender',bins=10)

Out[57]: array([<matplotlib.axes._subplots.AxesSubplot object at 0x000002382D962160>,
                <matplotlib.axes._subplots.AxesSubplot object at 0x000002382D810D30>],
               dtype=object)
```

Implementation :

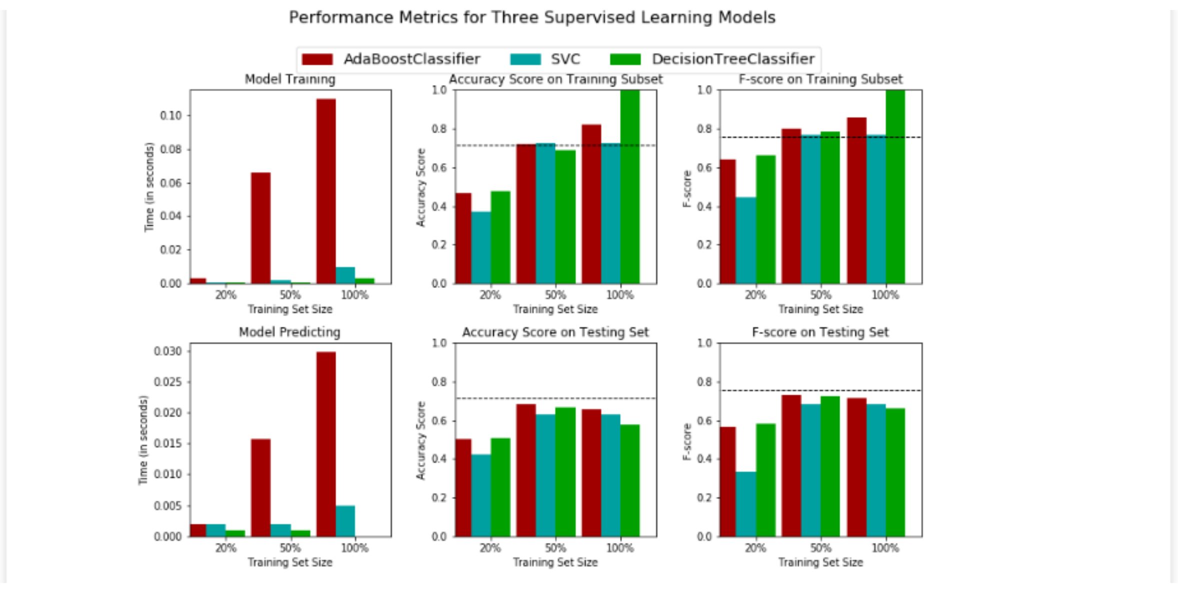
The dataset will be split into training and testing set as a 80% of training data and 20% of testing data can be split using train_test_split method from sklearn. Random state will be specified as a particular number .

Before applying any supervised learning technique, we will implement a naïve predictor, that will simply return that every data point has ‘Disease’= True. We will check our accuracy on that predictor. Note that in this case naive predictor will perform artificially well unlike in real world, a large proportion of patients has around 70% do have the disease.

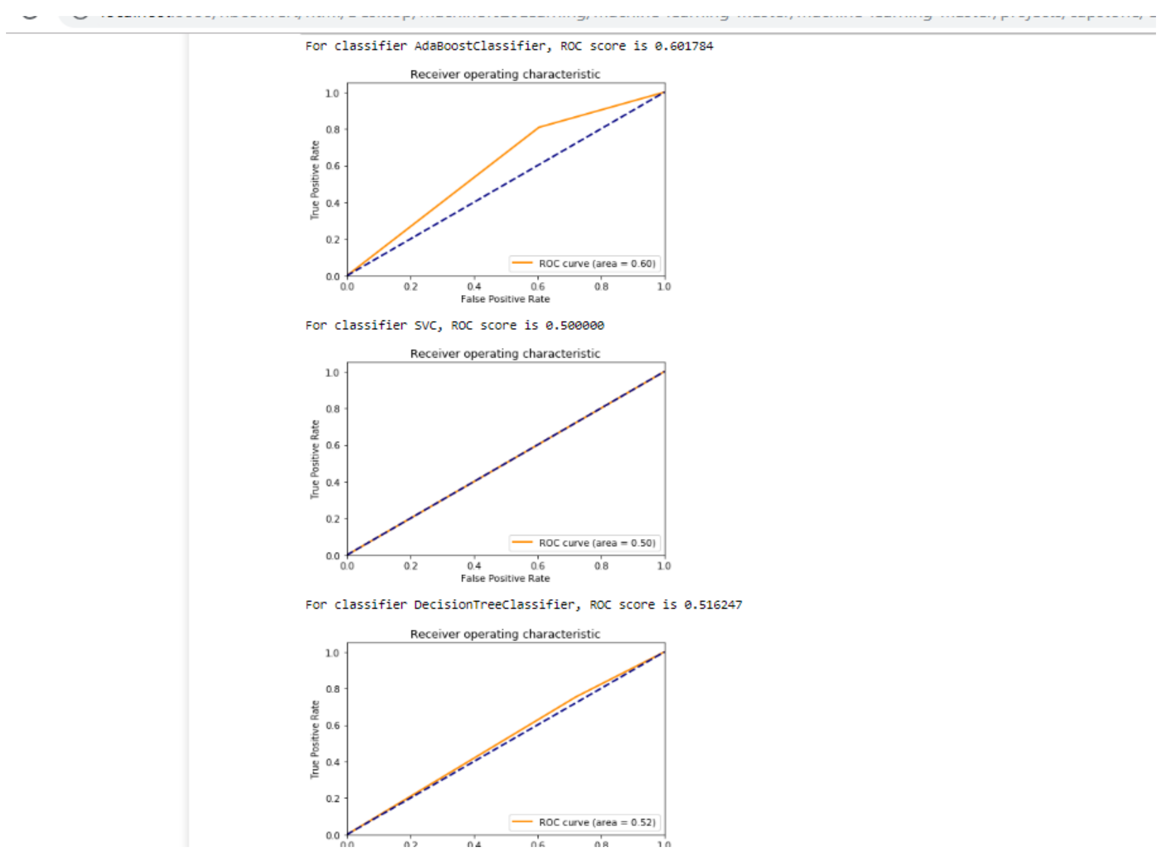
Then, a method called as 'train_predict' is defined that takes as input the following: learner, sample_size, X_train, y_train, X_test, y_test. It returns the accuracy and F-beta score on training and testing set respectively.

The three classifiers are sent to the 'train_predict' method, with 1%, 10% and 100% of training data respectively so that it can be seen how performance varies with sample size.

Initially, some difficulty was observed in outputting the results in the desired manner. The remedy was found in the form of a python file called as 'visuals.py' , that returns all the output variable in a bar graph format. This file has been derived from the 'finding_donors' project with some changes implemented. The output from this file is as follows:



An additional metric called as Receiver Operator Characteristics(ROC) curve will be used. It plots the curve of True Positive Rate and False positive Rate, with a greater area under the curve indicating a better True Positive Rate for the same False Positive Rate. This can be helpful in this case as simply knowing the number of correct predictions may not suffice.



```
For classifier AdaBoostClassifier, ROC score is 0.601784
```

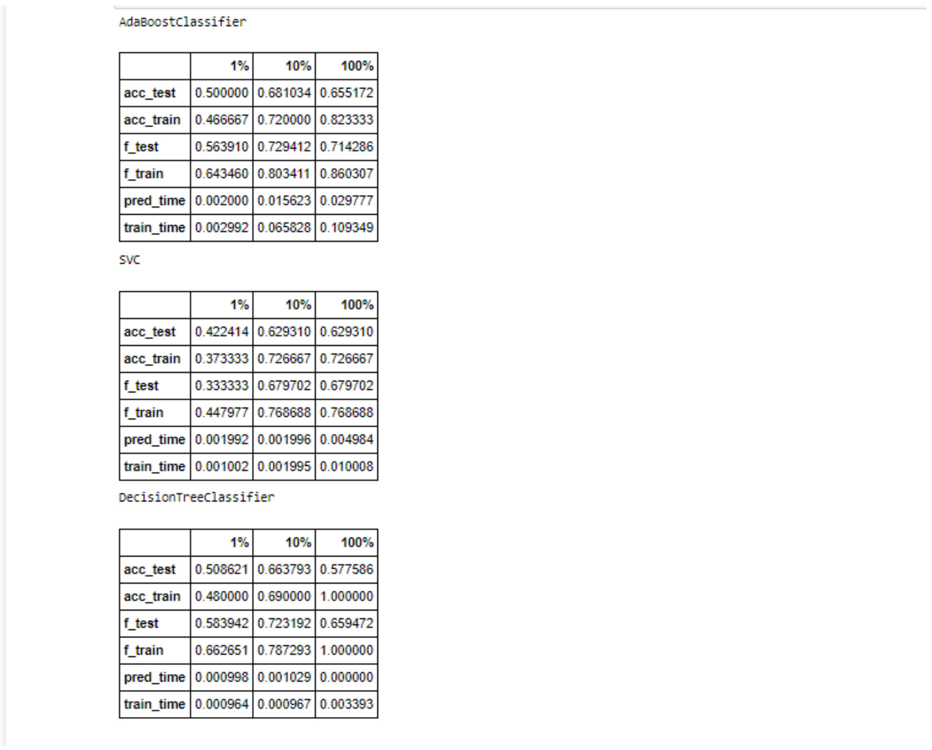
```
For classifier SVC, ROC score is 0.500000
```

```
For classifier DecisionTreeClassifier, ROC score is 0.516247
```

Results

Model Evaluation and Validation :

Since the size of dataset is small at present , there is not much difference between training and testing times of different algorithms. However, for the sake of comparison, these times have been displayed in the 'Implementation' sub-heading of 'Analysis' section. Adaboost Classifier consumes maximum time during training and good accuracy score ,F_score. From this dataset use three models based on the accuracy_score,F_score we decide Adaboost is best suitable for this dataset.



AdaBoostClassifier

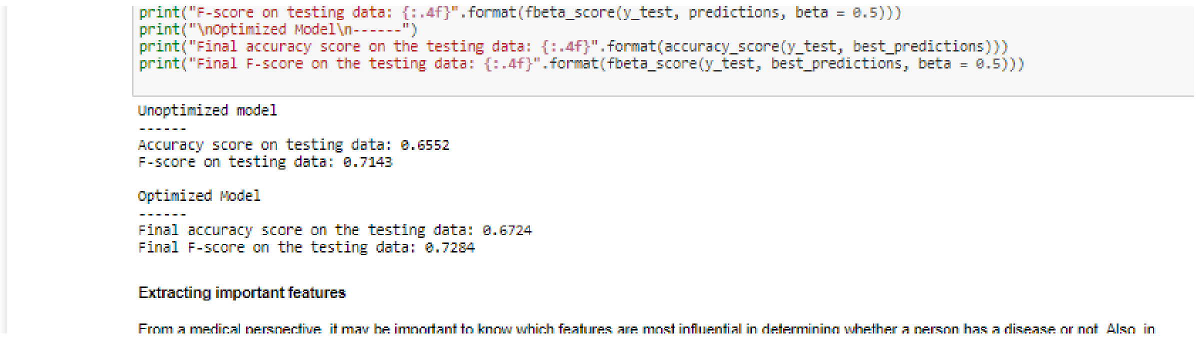
	1%	10%	100%
acc_test	0.500000	0.681034	0.655172
acc_train	0.466667	0.720000	0.823333
f_test	0.563910	0.729412	0.714286
f_train	0.643460	0.803411	0.860307
pred_time	0.002000	0.015623	0.029777
train_time	0.002992	0.065828	0.109349

SVC

	1%	10%	100%
acc_test	0.422414	0.629310	0.629310
acc_train	0.373333	0.726667	0.726667
f_test	0.333333	0.679702	0.679702
f_train	0.447977	0.768688	0.768688
pred_time	0.001992	0.001996	0.004984
train_time	0.001002	0.001995	0.010008

DecisionTreeClassifier

	1%	10%	100%
acc_test	0.508621	0.663793	0.577586
acc_train	0.480000	0.690000	1.000000
f_test	0.583942	0.723192	0.659472
f_train	0.662651	0.787293	1.000000
pred_time	0.000998	0.001029	0.000000
train_time	0.000964	0.000967	0.003393



```
print("F-score on testing data: {:.4f}".format(fbeta_score(y_test, predictions, beta = 0.5)))
print("\nOptimized Model\n------")
print("Final accuracy score on the testing data: {:.4f}".format(accuracy_score(y_test, best_predictions)))
print("Final F-score on the testing data: {:.4f}".format(fbeta_score(y_test, best_predictions, beta = 0.5)))
```

```
Unoptimized model
------
Accuracy score on testing data: 0.6552
F-score on testing data: 0.7143

Optimized Model
------
Final accuracy score on the testing data: 0.6724
Final F-score on the testing data: 0.7284
```

Extracting important features

From a medical perspective, it may be important to know which features are most influential in determining whether a person has a disease or not. Also, in

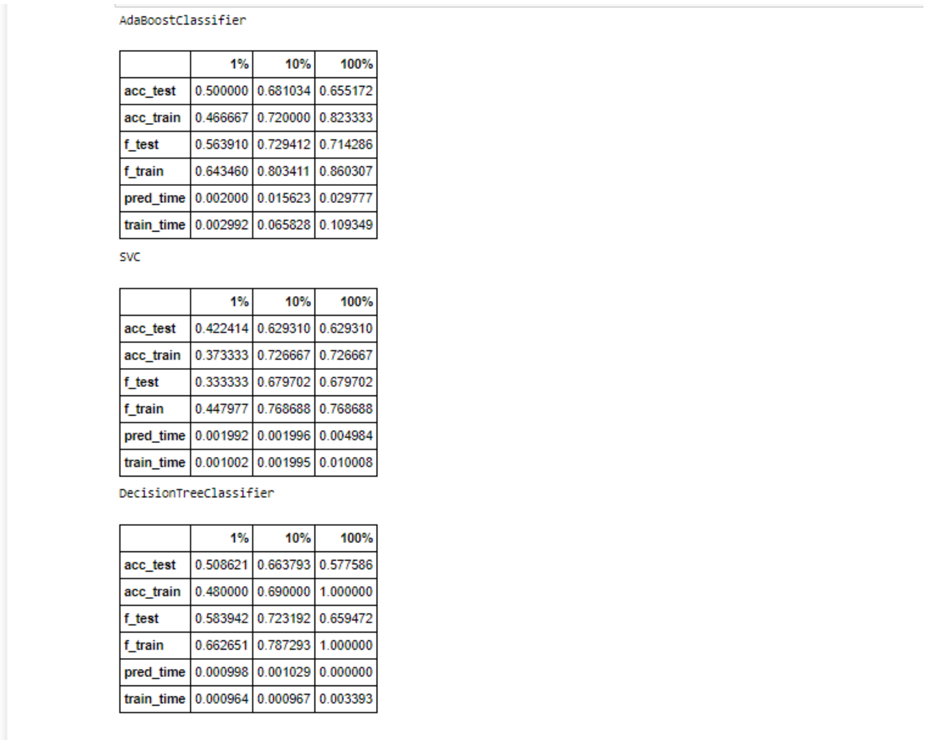
Justification :

I used accuracy score and f_score as evaluation metric for prediction of liver disease. Here I am predicting the accuracy score ,F_score and tesing, traing time for the selected models. Despite the ambiguous results, it was decided to select Adaboost Classifier as the final classifier and Final accuracy score on the testing data: 0.6724

Final F-score on the testing data: 0.7284

When we compare Decision tree classifier and SVM the adaboost classifier has good accuracy_score and f_score but it will take more training and testing time compare to other models.

When we see the area under the ROC curve 0.60 is greater than that of SVM(0.50) and Decision tree classifier(0.51) . As discussed before, we are considering ROC score as an important metric.



AdaBoostClassifier

	1%	10%	100%
acc_test	0.500000	0.681034	0.655172
acc_train	0.466667	0.720000	0.823333
f_test	0.563910	0.729412	0.714286
f_train	0.643460	0.803411	0.860307
pred_time	0.002000	0.015623	0.029777
train_time	0.002992	0.065828	0.109349

SVC

	1%	10%	100%
acc_test	0.422414	0.629310	0.629310
acc_train	0.373333	0.726667	0.726667
f_test	0.333333	0.679702	0.679702
f_train	0.447977	0.768688	0.768688
pred_time	0.001992	0.001996	0.004984
train_time	0.001002	0.001995	0.010008

DecisionTreeClassifier

	1%	10%	100%
acc_test	0.508621	0.663793	0.577586
acc_train	0.480000	0.690000	1.000000
f_test	0.583942	0.723192	0.659472
f_train	0.662651	0.787293	1.000000
pred_time	0.000998	0.001029	0.000000
train_time	0.000964	0.000967	0.003393

Conclusion

Free Form Visualization :

ROC curves that show the performance of the respective algorithms before and after applying GridSearchCV have been displayed in the preceding sections.

Reflection :

Initially, the dataset was explored and made ready to be fed into the classifiers. This was achieved by removing some rows containing null values, transforming some columns which were showing skewness and using appropriate methods(one-hot encoding) to convert the labels so that they can be useful for classification purposes.

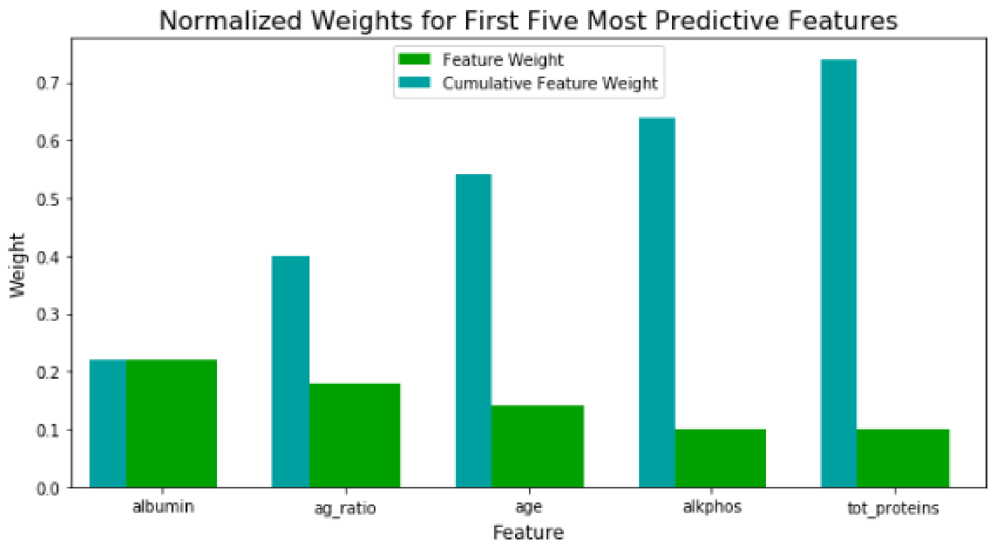
Performance metrics on which the models would be evaluated were decided. The dataset was then split into a training and testing set. Firstly , a naive predictor and a benchmark model were run on the dataset to determine the benchmark . Then , three classifiers were trained on the dataset and tested on the testing set.

Finally, all three models were refined to a certain extent by choosing different values of their hyperparameters using GridSearchCV. The models were compared on their F-beta as well as their ROC scores. The results turned out to be ambiguous, with SVM,Decision tree classifier and Adaboost running for best performance. The performance with respect to F-beta score was also marginally better SVM, Decision tree classifier and slightly worse for Adaboost. However, it is believed that these algorithms will improve once they are presented with a much larger dataset. I am learned the intreseted topic is visualization of the distribution function for every feature.

Secondly, fine tuning models using GridSearch was also tough, as it was computationally too expensive to include all the parameters for Adaboost. For this, I went through the definitions of hyperparameters in scikit-learn documentation, and then selected the best performing parameters and their values on the basis of a limited number of trials. This exercise made me realize that parameter tuning is not only a very interesting but also a very important part of machine learning. I think this area can warrant further improvement, if we are willing to invest a greater amount of time as well as computing power.

Improvement:

One way for improving is to conduct a more exhaustive search for a combination of parameters which might give better results using Grid Search, but that is computationally expensive. Also, can save some training time at the cost of accuracy by extracting the most influential features using features_importance_ attribute of the AdaBoost and training the model only on those features. This is implemented in the final portion of the project, and an F-beta score of 0.72 was obtained on predicting with 5 most influential features.


Normalized Weights for First Five Most Predictive Features
Feature Weight
Cumulative Feature Weight
Weight
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
albumin
ag_ratio
age
alkphos
tot_proteins
Feature