**LIVER ANALYSIS PATIENT**

Smart Bridge - Remote Summer Internship Program

Category - Machine Learning

****

By

Vuyyuru Sai Nithish Reddy

**Introduction:**

**Overview:**

Liver Disease is a major medical problem in India and can be cured if treated in the early stages. People are not aware that medical tests we take for different purposes could contain valuable information concerning liver diseases.

Attributes of various medical tests are investigated to distinguish which attributes may contain helpful information about the disease. The information says that it helps us to measure the severity of the problem, the predicted survival of the patient after the illness, the pattern of the disease and work for curing the disease.

**Purpose:**

Technological development, including machine learning, has a huge impact on health through an effective analysis of various diseases for more accurate diagnosis and successful treatment.

Our aim from the project is to make use of pandas, matplotlib, numpy, libraries from python to extract the libraries for machine learning for the liver disease prediction and the results are compared across a number of classification algorithms and regression techniques. The dataset is pre-processed by completing and normalizing missing data. The most relevant features are selected from the dataset for improved accuracy and reduced training time.

**Literature Survey**

**Existing Problem:**

Liver Disease is linked to several complications and in severe cases may also lead to liver transplantation and these complications contribute to high morbidity and mortality and poor quality of life.

**Proposed Solution:**

With the use of Machine Learning Model, there will be no limitation of the complexity increasing number of variables.This Model train and test the given factors which cause Liver Disease and with the best performing machine learning model it can effortlessly predict the result of the patient with much higher accuracy than traditional methods.

**Problem Statement:**

The problem statement is formally defined as:

‘Given a dataset containing various attributes of 584 Indian patients, use the features available in the dataset and define a supervised classification algorithm which can identify whether a person is suffering from liver disease or not. 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. This data set contains 441 male patient records and 142 female patient records. Any patient whose age exceeded 89 is listed as being of age "90".

**Metrics:**

In problems of disease classification like this one, simply comparing the accuracy, that is, the ratio of correct predictions to total predictions is not enough. 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, while predicting a healthy person as diseased will attract a comparatively less severe penalty.

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:

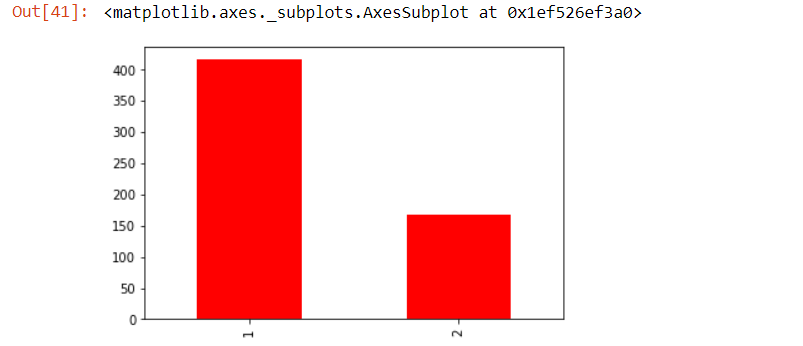
F-beta score = (1+β2*)\**precision\*recall/(( β 2\*precision)+recall)

β = A number that decides relative weightage of precision and recall. In this case, a disease being classified as a non-disease will incur a high penalty. So, more emphasis is placed on recall.

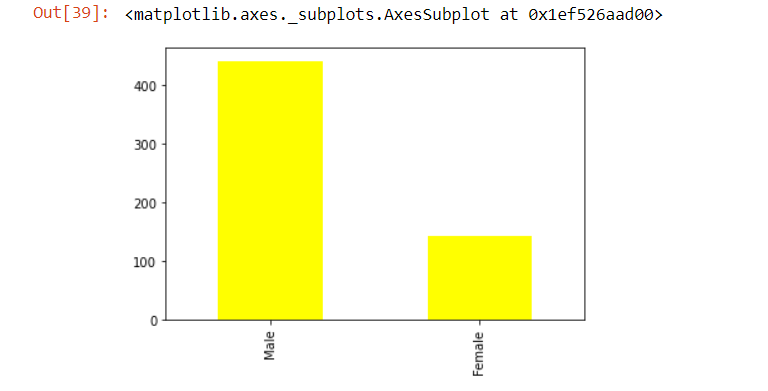
Additionally, one more metric called as Receiver Operating Characteristics (ROC) curve will be used. It plots the curve of True Positive Rate vs the False Positive Rate for a given algorithm, with a greater area under the curve indicating a better True Positive Rate for the same False Positive Rate,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

**Exploratory Data Analysis:**

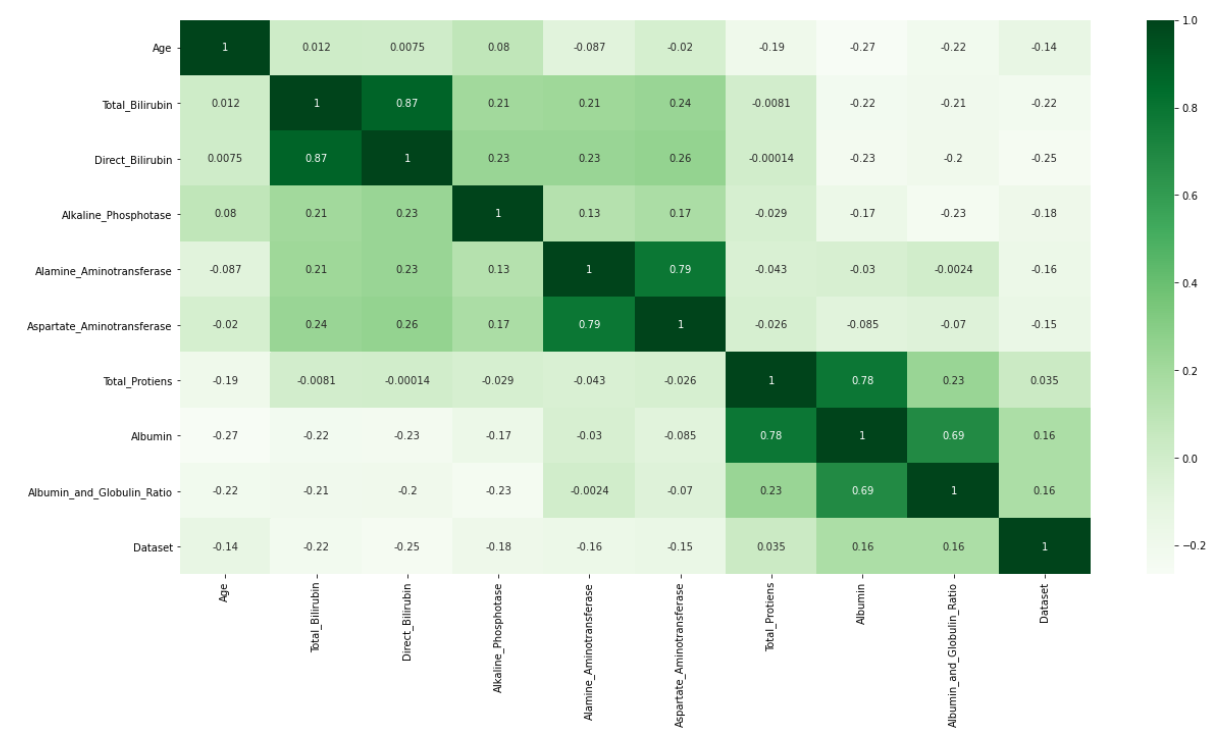
COUNT PLOT ON DIAGNOSED WITH LIVER DISEASE AND NOT DIAGNOSED WITH LIVER DISEASE:

****

COUNT PLOT ON TOTAL NUMBER OF THE GENDER:

****

**CO-RELATION GRAPH:**

****

**Algorithms and Techniques:**

One supervised learning approach is selected for this problem. Based on the accuracy we found that logistic regression is the best-suited algorithm for this problem statement.

For each algorithm, we will try out different values of a few hyperparameters to arrive at the best possible classifier. This will be carried out with the help of grid search cross validation technique. But logistic regression gave the maximum accuracy.

**Logistic Regression**:

Since the outcome is binary and we have a reasonable number of examples at our disposal compared to number of features, this approach seems suitable. At the core of this method is a logistic or sigmoid function that quantifies the difference between each prediction and its corresponding true value. When presented with a number of inputs, it assigns different weights to features (based on their relative importance).

Since for this data it already knows the output beforehand, it continuously adjusts the weights such that when these weights summed up with their features are introduced in the logistic function, the results are as near as possible to the actual ones. Once presented with a test value, it again inserts the value into our logistic function and returns the output as a number between 0 and 1, which represents the probability of that test value being in a particular class.

**Accuracy scored: 0.70**

**Conclusion:**

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 ('Logistic Regression') were run on the dataset to determine the benchmark value of accuracy. The greatest difficulty in the execution of this project was faced in two areas- determining the algorithms for training and choosing proper parameters for fine-tuning. Initially, I found it very vexing to decide upon 3 or 4 techniques out of the numerous options available in sklearn.

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.

**Appendix:**

Source Code for Data Preprocessing:

import the libraries:

**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** sklearn

pd.options.display.max\_columns **=** **None**

pd.options.display.max\_rows **=** **None**

import the dataset

ds**=**pd.read\_csv('indian\_liver\_patient.csv')

Ds

Taking Care of Missing Data

ds.isnull().sum()

ds['Albumin\_and\_Globulin\_Ratio'].isnull()

ds['Albumin\_and\_Globulin\_Ratio'].fillna((ds['Albumin\_and\_Globulin\_Ratio'].mean()),inplace=True)

Ds

ds.isnull().sum()

Label Encoding

In [47]:

**from** sklearn.preprocessing **import** LabelEncoder

In [48]:

lb**=**LabelEncoder()

x[:,1]**=**lb.fit\_transform(x[:,1])

X

y[:,0]=lb.fit\_transform(y[:,0])

y

Data Visualization

ds['Gender'].value\_counts().plot.bar(color='yellow')

ds['Dataset'].value\_counts().plot.bar(color='red')

corr=ds.corr()

plt.figure(figsize=(20,10))

sns.heatmap(corr,cmap="Greens",annot=True)

Splitting train and testing data

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.metrics **import** classification\_report,confusion\_matrix

x\_train,x\_test,y\_train,y\_test**=**train\_test\_split(x,y,test\_size**=**0.2,random\_state=0)

Source Code for Logistic Regression :

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

x\_train = sc.fit\_transform(x\_train)

x\_test = sc.transform(x\_test)

from joblib import dump

dump(sc,"scalar.save")

from sklearn.linear\_model import LogisticRegression

from sklearn.pipeline import Pipeline

pipe = Pipeline([

('rescale',StandardScaler()),

('classifer',LogisticRegression())

])

pipe.fit(x\_train,y\_train)

import pickle

pickle.dump(pipe,open('decision.pk1','wb'))

y\_pred=pipe.predict(x\_test)

y\_pred

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test,y\_pred)

Confusion Matrix:

print('Confusion Matrix: \n', confusion\_matrix(y\_test,y\_pred))

Classification Report:

print('Classification Report: \n', classification\_report(y\_test,y\_pred))