VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF INFORMATION AND TECHNOLOGY

SOFTWARE METRICS

PROJECT REPORT

SWE2020-FALL 2021-22

CLASSIFICATION OF INDIAN LIVER PATIENT DATASET

SUBMITTED BY:

18MIS0246

Raveendar A

CLASSIFICATION OF INDIAN LIVER PATIENT

DATASET

ABSTRACT:

This project attempts to achieve efficient early detection of Liver disease through different algorithms. We collected Five hundred and eighty-three records related to the Indian Liver Patient Dataset from UCI repository. This data set contains 416 liver patient records and 167 non liver patient records collected from North East of Andhra Pradesh, India. The "Dataset" column is a class label used to divide groups into liver patient (liver disease) or not (no disease). This data set contains 441 male patient records and 142 female patient records.The ILPD dataset is divided into 70% for the training stage and 30% for the testing stage. Indian liver patient dataset Contains10 variable that are age, gender, total bilirubin, Direct bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alpo's. We will detect whether the person is affected by liver Disease and also find the overall accuracy.

KEYWORDS:

Random Forest Classifier, Gaussian Naive Bayes Classifier , Logistic Regression, Patients, Liver dataset.

1. INTRODUCTION:

One of the major cause of human death is liver disorder diseases. Liver is the second largest inside organ in the human body. Playing a key role in the metabolism and serving several imperative functions and its disorder has become one of the big issues of human diseases around the world. Liver diseases are one of the most killer diseases by the most cause of Viral Hepatitis in the world. Enhanced health analysis may be accomplished complete automatic diagnosis of patient record stored in health data, i.e., By learning from past experiences. And data mining is applied to this stored medical record to retrieve information from the data. According to data mining techniques categorized into Association, classification, and clustering. The patient taking the reports back to the hospital, where they are examined and the disease is identified. This project aims to somewhat reduce the time delay caused due to the unnecessary back and forth shuttling between the hospital and pathology lab. To overcome this problem, we have chosen and trained a dataset against an algorithm that

will be helpful to predict and classify liver disease in patients. Liver patient dataset is a multivariate dataset contain ten attributes that are: age, gender, total billirubin, direct billirubin, total proteins, albumin, a/g ratio, sgpt, sgot, and alkphos. This dataset contains 416 liver patient records and 167 non-liver patient records. Random Forest Classifier is applied to the dataset . Classification algorithms are used when the outputs are restricted to a limited set of values. For a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email. Regression algorithms are named for their continuous outputs, meaning they may have any value within a range. Examples of a continuous value are the temperature, length, or price of an object. In unsupervised learning, the algorithm builds a mathematical model from a set of data which contains only inputs and no desired output labels.

2. REQUIREMENTS SPECIFICATION:

I) PURPOSE :

We will detect whether the person is affected by liver Disease and also find the overall accuracy.

II) SCOPE :

Patients with Liver disease have been continuously increasing because of excessive consumption of alcohol, inhale of harmful gases, intake of contaminated food, pickles and drugs. This dataset was used to evaluate prediction algorithms in an effort to reduce burden on doctors.

III) FUNCTIONAL REQUIREMNTS:

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. Exact details of the number and types of algorithms used for training is included in the 'Algorithms and Techniques' sub-section of the 'Analysis' part.

IV) INTERFACE REQUIREMENTS:

Python is a very powerful programming language used for many different applications. Over time, the huge community around this open source language has

created quite a few tools to efficiently work with Python. Many other paradigms are supported via extensions, including design by contract and logic programming.

V) PERFORMANCE REQUIREMENTS:

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.

3. DESIGN METHODOLOGIES:

TOOLS USED:

- ➔ GOOGLE COLAB
- ➔ PANDAS:

  Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive.

- ➔ NUMPY:

  NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays.

- ➔ MATPLOTLIB:

  Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

- ➔ SEABORN:

  Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis.

- ➔ SKLEARN:

  Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy

APPROACHES USED:

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: 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. The algorithms are described below:

➔ RANDOM FOREST:

It comes under the category of ensemble methods. It employs 'bagging' and 'boosting' methods to draw a random subset from the data, and train a Decision Tree on that (hence, the name Random Forest). In this case, we don't know the relative importance of each feature while deciding the output, so a Random Forest can be successful as it will ensure training on different randomized subsets. Hyperparameters to be manipulated:

i) n_estimators(number of trees in a forest)

ii) max_depth(maximum depth of one single tree)

iii) max_features(decides how many features are to be used)

iv) oob_score(decides whether to include out-of-bag or prediction error)

➔ GAUSSIAN NAIVE BAYES CLASSIFIER

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values. The representation for naive Bayes is probabilities.

A list of probabilities are stored to file for a learned naive Bayes model.

This includes:

- Class Probabilities: The probabilities of each class in the training dataset.
- Conditional Probabilities: The conditional probabilities of each input value given each class value.

➔ 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.

ARCHITECTURE DIAGRAM:

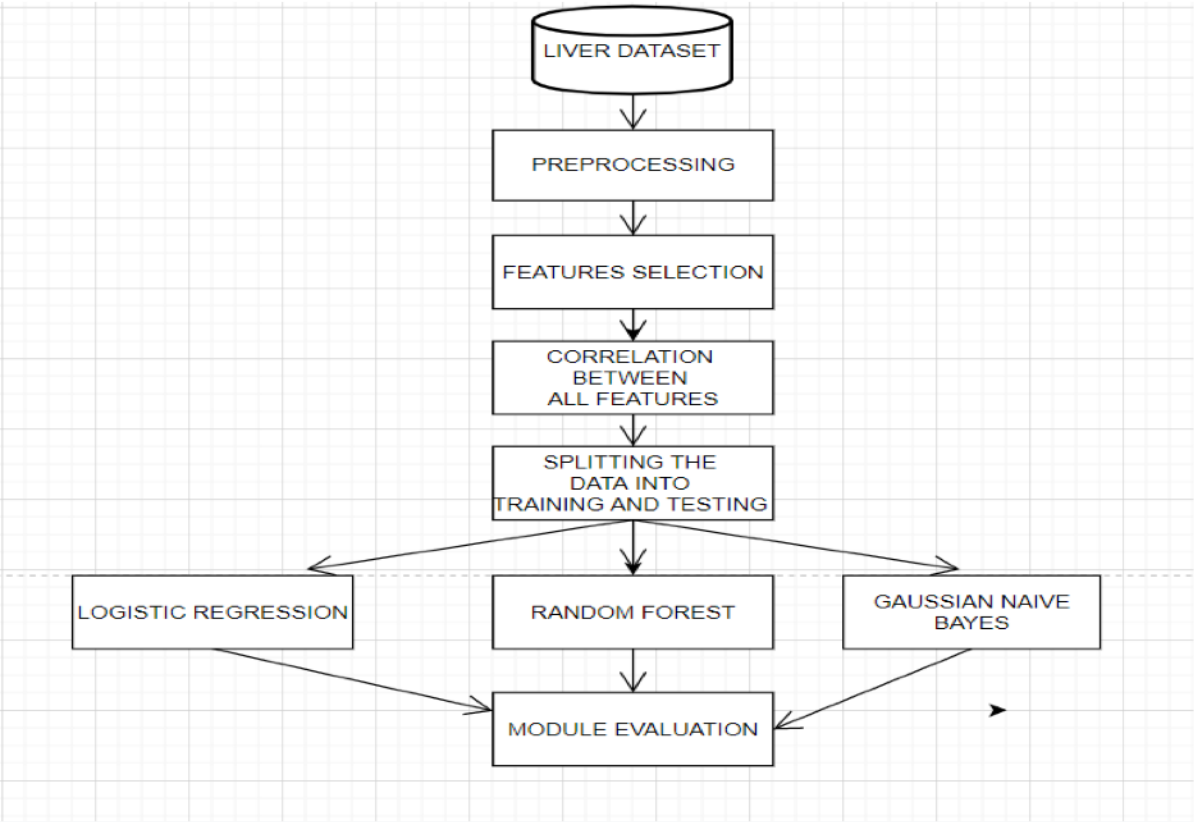

Fig 3.1 Architecture diagram for Liver dataset analysis

4. IMPLEMENTATION:

1. Importing Libraries
2. Reading the Data from the CSV file
3. Exploratory Data Analysis (EDA)

```
[ ] # Information about the dataset
    liver_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
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
```

Fig 4.1 Exploratory Data Analysis (EDA) for all features

4. Data Visualization:

```
# Plotting the Number of patients with liver disease vs Number of patients with no liver disease
sns.countplot(data=liver_df, x = 'Dataset', label='Count')

LD, NLD = liver_df['Dataset'].value_counts()
print('Number of patients diagnosed with liver disease: ',LD)
print('Number of patients not diagnosed with liver disease: ',NLD)
```

```
Number of patients diagnosed with liver disease:  416
Number of patients not diagnosed with liver disease:  167
```

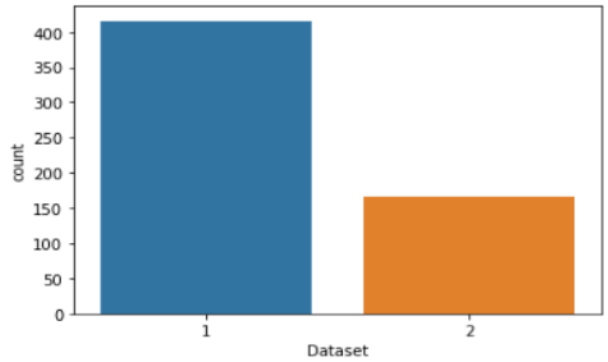

Fig 4.2 Data visualization for patients diagnosed with liver disease

```
# Plotting the Number of Male and Female patients
sns.countplot(data=liver_df, x = 'Gender', label='Count')

M, F = liver_df['Gender'].value_counts()
print('Number of patients that are male: ',M)
print('Number of patients that are female: ',F)
```

```
Number of patients that are male:  441
Number of patients that are female:  142
```

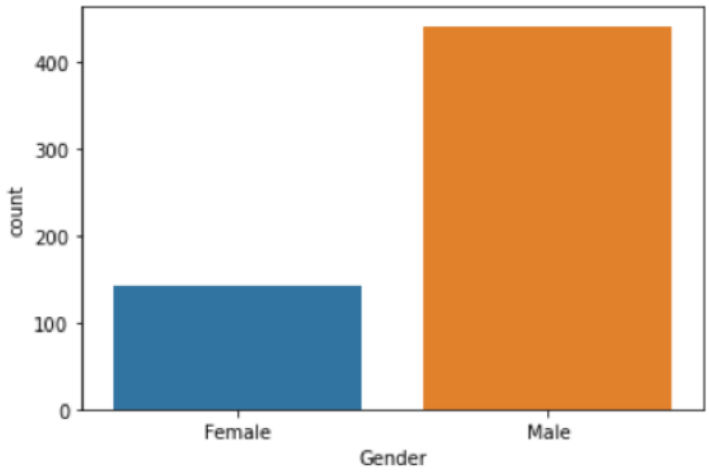

Fig 4.3 Data visualization for patients male and female

5. Feature engineering

```
liver_corr = X.corr()
liver_corr
```

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_ar
Age	1.000000	0.011763	0.007529	0.080425	-0.086883	-0.019910	-0.187461	-0.265924	
Total_Bilirubin	0.011763	1.000000	0.874618	0.206669	0.214065	0.237831	-0.008099	-0.222250	
Direct_Bilirubin	0.007529	0.874618	1.000000	0.234939	0.233894	0.257544	-0.000139	-0.228531	
Alkaline_Phosphotase	0.080425	0.206669	0.234939	1.000000	0.125680	0.167196	-0.028514	-0.165453	
Alamine_Aminotransferase	-0.086883	0.214065	0.233894	0.125680	1.000000	0.791966	-0.042518	-0.029742	
Aspartate_Aminotransferase	-0.019910	0.237831	0.257544	0.167196	0.791966	1.000000	-0.025645	-0.085290	
Total_Protiens	-0.187461	-0.008099	-0.000139	-0.028514	-0.042518	-0.025645	1.000000	0.784053	
Albumin	-0.265924	-0.222250	-0.228531	-0.165453	-0.029742	-0.085290	0.784053	1.000000	
Albumin_and_Globulin_Ratio	-0.216089	-0.206159	-0.200004	-0.233960	-0.002374	-0.070024	0.233904	0.686322	
Gender_Female	-0.056560	-0.089291	-0.100436	0.027496	-0.082332	-0.080336	0.089121	0.093799	
Gender_Male	0.056560	0.089291	0.100436	-0.027496	0.082332	0.080336	-0.089121	-0.093799	

Fig 4.4 Feature Engineering for all features

6. Splitting data into Train and Test:

▾ Splitting the data into Train and Test

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
print (X_train.shape)
print (y_train.shape)
print (X_test.shape)
print (y_test.shape)
```

```
(390, 11)
(390,)
(193, 11)
(193,)
```

Fig 4.5 splitting the data into train and test

7. Model Building

A machine learning model is built by learning and generalizing from training data, then applying that acquired knowledge to new data it has never seen before to make predictions and fulfill its purpose.

a. Logistic Regression

```
logreg = LogisticRegression()

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

# 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)

# Equation coefficient and Intercept
print('Logistic Regression Training Score: \n', logreg_score)
print('Logistic Regression Test Score: \n', logreg_score_test)

print('Accuracy: \n', accuracy_score(y_test,log_predicted))
print('Confusion Matrix: \n', confusion_matrix(y_test,log_predicted))
print('Classification Report: \n', classification_report(y_test,log_predicted))
```

Fig 4.6 predicting using logistic regression model

b. Gaussian Naive Bayes

```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
# Predict Output
gauss_predicted = gaussian.predict(X_test)

gauss_score = round(gaussian.score(X_train, y_train) * 100, 2)
gauss_score_test = round(gaussian.score(X_test, y_test) * 100, 2)
print('Gaussian Score: \n', gauss_score)
print('Gaussian Test Score: \n', gauss_score_test)
print('Accuracy: \n', accuracy_score(y_test, gauss_predicted))
print(confusion_matrix(y_test,gauss_predicted))
print(classification_report(y_test,gauss_predicted))
```

Fig 4.7 predicting using Gaussian naïve bayes model

c. Random Forest

```
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, y_train)
# Predict Output
rf_predicted = random_forest.predict(X_test)

random_forest_score = round(random_forest.score(X_train, y_train) * 100, 2)
random_forest_score_test = round(random_forest.score(X_test, y_test) * 100, 2)
print('Random Forest Score: \n', random_forest_score)
print('Random Forest Test Score: \n', random_forest_score_test)
print('Accuracy: \n', accuracy_score(y_test,rf_predicted))
print(confusion_matrix(y_test,rf_predicted))
print(classification_report(y_test,rf_predicted))
```

Fig 4.8 predicting using random forest model

8. Model Evaluation

```
# Comparing all the models
models = pd.DataFrame({
    'Model': [ 'Logistic Regression', 'Gaussian Naive Bayes','Random Forest'],
    'Score': [ logreg_score, gauss_score, random_forest_score],
    'Test Score': [ logreg_score_test, gauss_score_test, random_forest_score_test]})
models.sort_values(by='Test Score', ascending=False)
```

	Model	Score	Test Score
0	Logistic Regression	70.77	72.54
2	Random Forest	100.00	71.50
1	Gaussian Naive Bayes	53.59	57.51

Conclusion

Fig 4.9 comparing the accuracy for all models

5. METRICS ANALYSIS:

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.

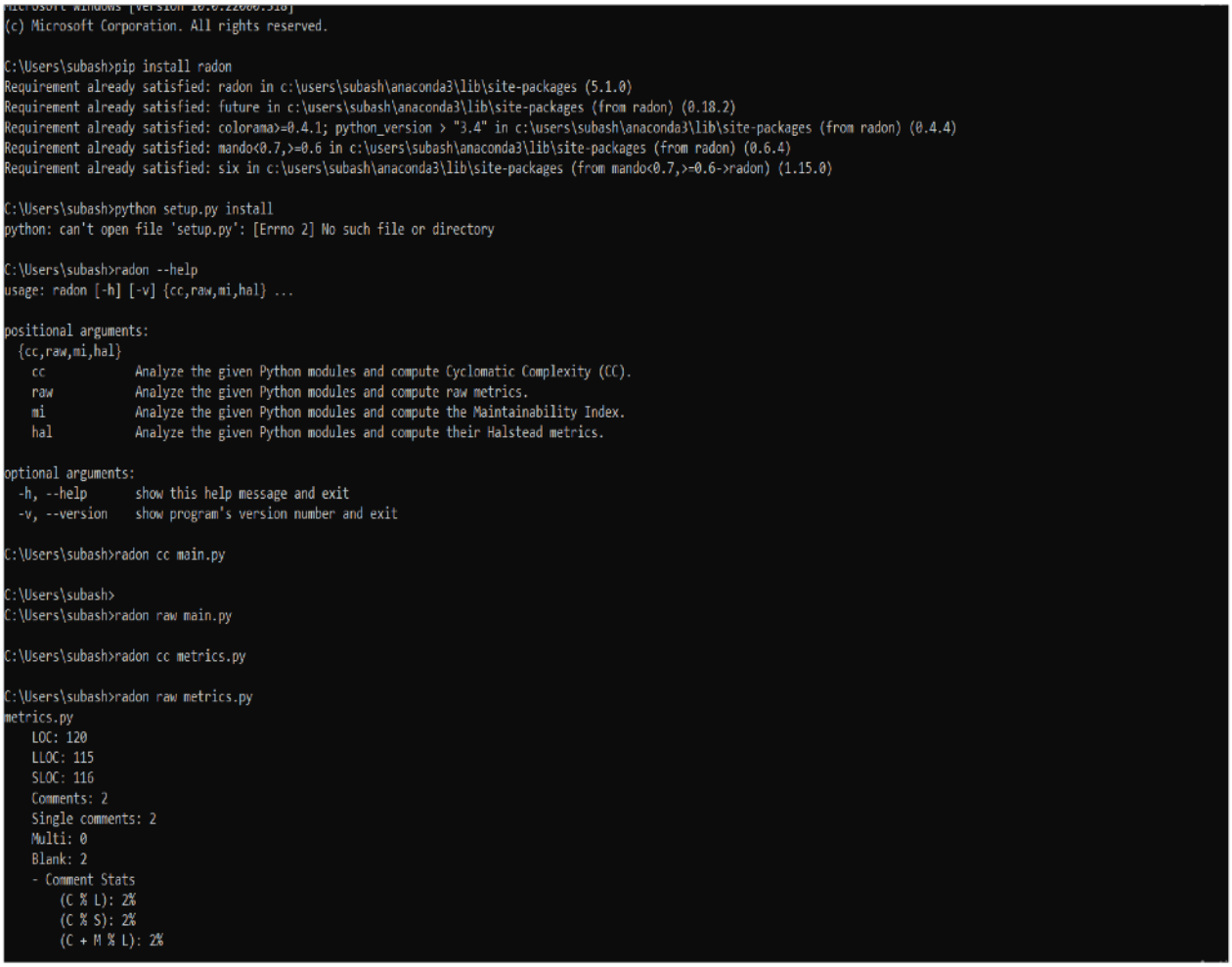

Fig 5.1 metrics like loc,lloc,sloc etc…

```
        (C + M % L): 2%

C:\Users\subash>radon cc metrics.py

C:\Users\subash>radon mi metrics.py
metrics.py - A

C:\Users\subash>radon hal metrics.py
metrics.py:
    h1: 1
    h2: 7
    N1: 6
    N2: 12
    vocabulary: 8
    length: 18
    calculated_length: 19.651484454403228
    volume: 54.0
    difficulty: 0.8571428571428571
    effort: 46.285714285714285
    time: 2.571428571428571
    bugs: 0.018

C:\Users\subash>radon cc metrics.py

C:\Users\subash>radon CC metrics.py
usage: radon [-h] [-v] {cc,raw,mi,hal} ...
radon: error: invalid choice: 'CC' (choose from 'cc', 'raw', 'mi', 'hal')

C:\Users\subash>radon cc metrics.py

C:\Users\subash>radon cc metrics.py -s

C:\Users\subash>radon cc metrics.py

C:\Users\subash>radon mi metrics.py -s
metrics.py - A (50.64)
```

Fig 5.2 metrics like halsted approach

LOGISTIC REGRESSION HEATMAP FOR CONFUSION MATRIX:

```
sns.heatmap(confusion_matrix(y_test,log_predicted),annot=True,fmt="d")
```

```
<AxesSubplot:>
```

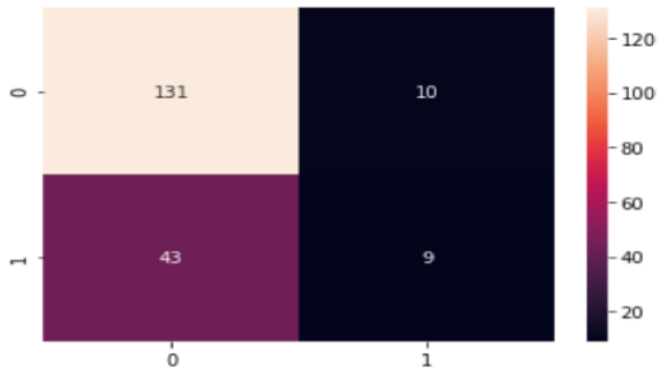

Fig 5.3 confusion matrix for logistic regression

GAUSSIAN NAIVE BAYES HEATMAP FOR CONFUSION MATRIX:

```
sns.heatmap(confusion_matrix(y_test,gauss_predicted),annot=True,fmt="d")
```

```
<AxesSubplot:>
```

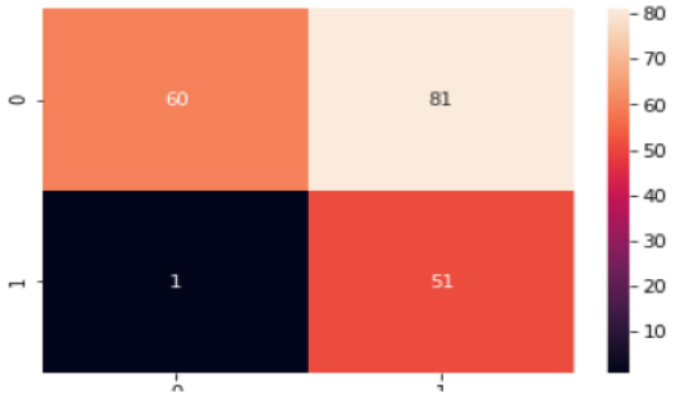

Fig 5.4 confusion matrix for Gaussian naïve bayes

RANDOM FOREST HEATMAP FOR CONFUSION MATRIX:

```
[ ]  sns.heatmap(confusion_matrix(y_test,rf_predicted),annot=True,fmt="d")
```

```
<AxesSubplot:>
```

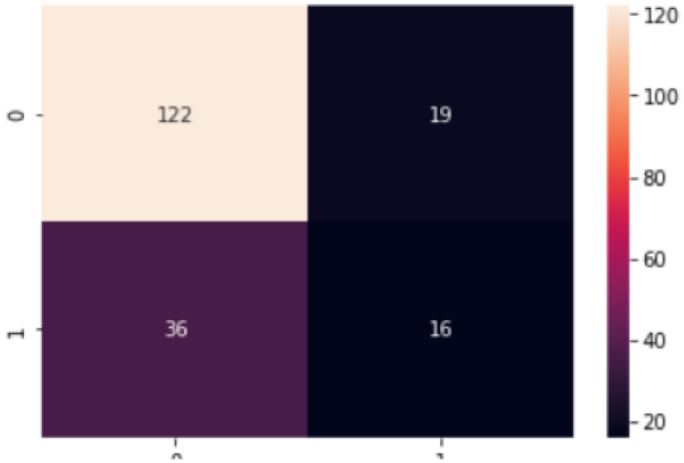

Fig 5.5 confusion matrix for random forest

6. 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.

7. REFERENCES:

[1] S. Muthuselvan1 , S. Rajapraksh2 , K. Somasundaram3, K. Karthik4 " Classification of Liver Patient Dataset Using Machine Learning Algorithms." International Journal of Engineering & Technology, September 2018, Issue no:03.

[2] M. Banu Priya1, P. Laura Juliet2, P.R. Tamilselvi3 "Performance Analysis of Liver Disease Prediction Using Machine Learning Algorithms" International Research Journal of Engineering and Technology (IRJET), Jan-2018 Issue: 01, Volume: 05. IRJET-V5I142.pdf

[3] Joel Jacob1, Joseph Chakkalakal Mathew2, Johns Mathew3, Elizabeth Issac4 "Diagnosis of Liver Disease Using Machine Learning Techniques ." International Research Journal of Engineering and Technology (IRJET), Apr-2018, Issue: 04 , Volume: 05 .IRJET-V5I4896.pdf

[4] Vasan Durai, Suyan Ramesh, Dinesh Kalthireddy "Liver disease prediction using machine learning" International Journal of Advance Research, Ideas and Innovations in Technology,sep-2019,issue : 2,Volume 5. Liver disease prediction using machine learning (ijariit.com).

[5] Vijay Panwar, Naved Choudhary, Sonam Mittal, Gaurav Sahu "REVIEW OF LIVER DISEASE PREDICTION USING MACHINE LEARNING ALGORITHM" Journal of emerging technologies and innovative research(JETIR) February 2021, Volume 8, Issue 2

JETIR2102026.pdf

[6] G.Compean D, J.Quintana , "M.Garza .Hepatogenous diabetes: current views of an ancient problem." Ann Hepatol ,pp 8:13,vol.20, 2019.

[7] S. Karthik, A. Priyadarishini and J. Anuradha and B. K. Tripathy,"Classification and Rule Extraction using Rough Set for Diagnosis of Liver Disease and its Types", Pelagia Research Library,Advances in Applied Science Research, 2017.

[8] T. Weber, K. Blin, S. Duddela, D. Krug, H. U. Kim, R. Bruccoleri, et al., "antiSMASH 3.0—a comprehensive resource for the genome mining of biosynthetic gene clusters," Nucleic acids research, vol. 43, pp. W237-W243, 2015.

[9] J. Ji, A. Zhang, C. Liu, X. Quan, and Z. Liu, "Survey: Functional module detection from protein-protein interaction networks," IEEE Transactions on Knowledge and Data Engineering, vol. 26, pp. 261- 277, 2017

[10] Yanwei, X.; Wang, J.; Zhao, Z.; Gao, Y., "Combination data mining models with new medical data to predict outcome of coronary heart disease". Proceedings International Conference on

[11] B. Schroeder and G. Gibson, The Computer Failure Data Repository (CFDR): collecting, sharing and analyzing failure data, SC 06 Proc. 2006 ACM/IEEE Conf. Supercomput., no. March, p. 154, 2016

[12]. Hastie T, Robert, T, Jerome F (2019). The Elements of Statistical Learning: Data mining, Inference andPrediction. Springer. 485–586.

[13]. P.Rajeswari, G.Sophia Reena (2017). Analysis of Liver Disorder using Data Mining Algorithm. Global journal of Computer Science and Technology. 10(14): 48-52.

[14]. BendiVenkataRamana, Prof. M.Surendra Prasad Babu and Prof. N. B. Venkateswarlu (2021). A Critical Study of Selected Classification Algorithms for Liver Disease Diagnosis. International Journal of Database Management Systems. 3(2): 101-111.

[15]. Dr. Soin. Liver diseases affect one in 5 Indians. Retrieved from:http://timesofindia.indiatimes.com/city/mumbai/Liver-diseases-affect-one-in-5-Indians/articleshow/31394640.cms

[16]. Min-Jung Song, Dong-Hwa Tun, Suk-In Hong (2009). An Electrochemical Biosensor Array for Rapid Detection of Alanine Aminotransferase and Aspartate Aminotransferase. Biosci. Biotechnol. Biochem. 73(4):474-478.

[17]. Fabricio Voznika, Leonardo Viana. Data Mining Classification. Retrieved from:http://courses.cs.washington.edu/courses/csep521/07wi/prj/leonardo_fabricio.pdf

[18]. Petr Somol, Bart Baesens, Pavel Pudil, Jan Vanthienen. Filter-versus Wrapper-based Feature Selection forCredit Scoring. 1-16. Retrieved from: http://library.utia.cas.cz/separaty/historie/somol-filter-%20versus%20wrapper-based%20feature%20selection%20for%20credit%20scoring.pdf

[19].Ashwani Kumar and Neelam Sahu, "Categorization of Liver Disease Using Classification Techniques", International Journal for Research in Applied Science & Engineering Technology

(IJRASET), vol. 5, no. V, May 2017.

[20.]Chandrasegar Thirumalai and Rashad Manzoor, "Cost Optimization using Normal Linear Regression Method for Breast Cancer Type I Skin", International Conference on Electronics Communication and Aerospace Technology ICECA, 2017.