Liver disease prediction with majority voting

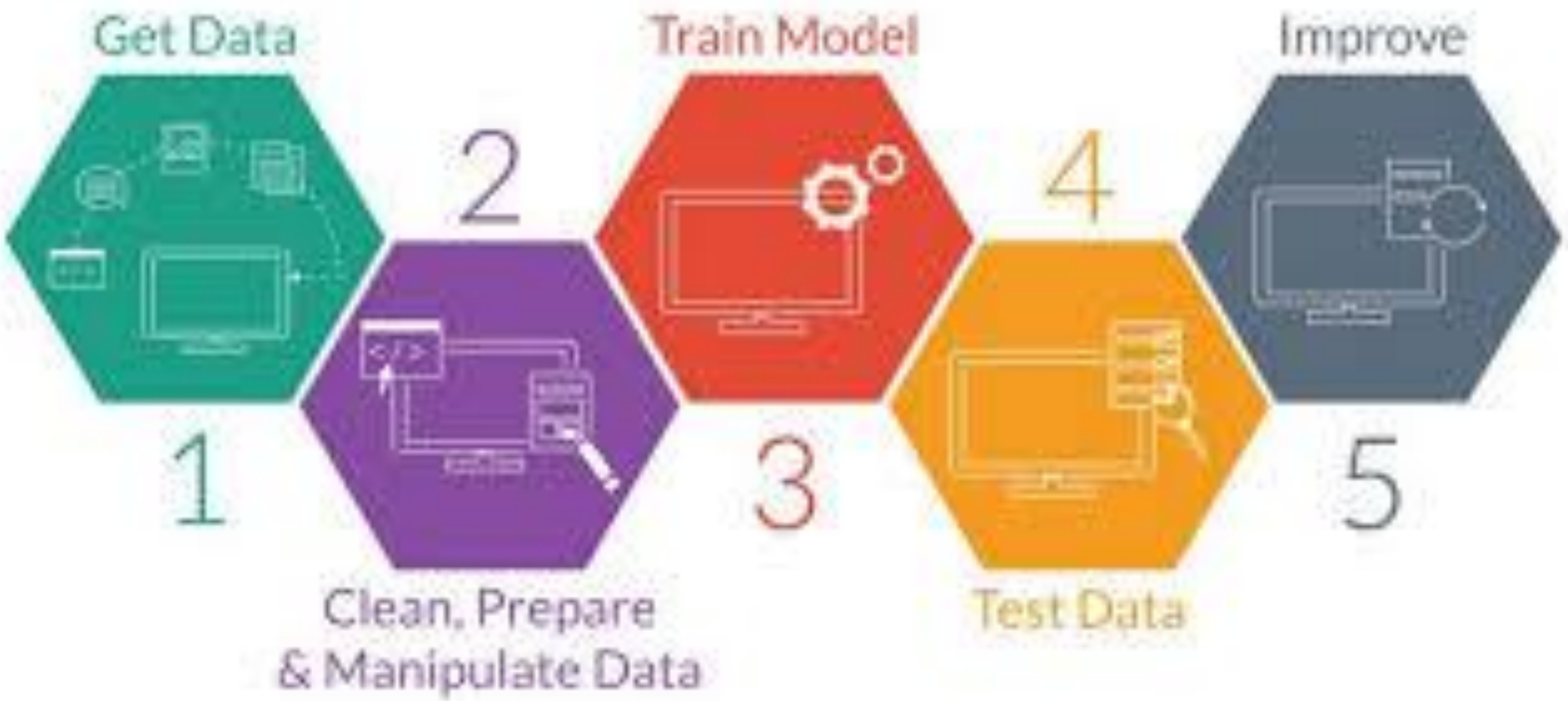
Team name- Fudgelicious

- Fahim Ishrak, 19141012, section 4
- Adiba Mahbub Proma, 15201014, section 4
- Tahsin Mostafa, 16101186, section 4
- Mohammad Waseq ul Islam, 19341014, section 4

Contributions of different members

- Mohammad Waseq ul Islam: Dataset analysis, presentation making
- Fahim Ishrak: Coding and environment set up
- Tahsin Mostafa: Algorithm Analysis and Coding
- Adiba Mahbub Proma: Algorithm Analysis, Result Analysis, Presentation making

Introduction

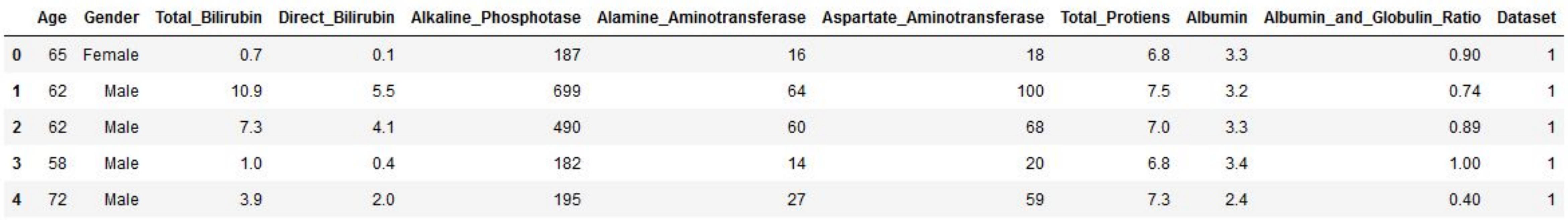


Introduction

- Using just one classifier can be risky as different classifiers learn differently, and so can give varied results.
- One classifier also has higher chance of predicting wrong on particular instances
- The goal of the topic is to be able to accurately predict whether a person might get liver diseases or not.
- Our area of interest is Supervised learning, which is **the** task of learning a function that maps an input to an output
- Specifically, we are targeting binary classification problems
- Our project covers predictions of diseases, data analytics, and pattern analysis, so that prediction is most accurate.

Details about the Dataset

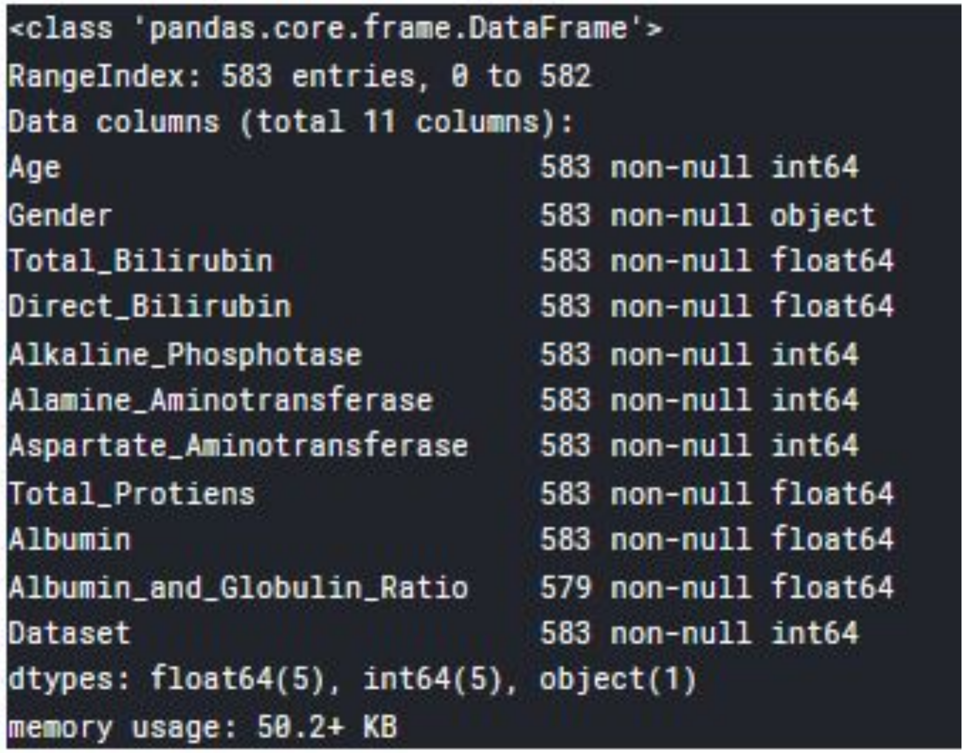
- Bilirubin level, alkaline phosphotase(ALP), alanine aminotranferase(ALT), aspertase aminotransferase (AST) features extracted.
- ALP- enzyme found throughout the body. Highest concentration in bone and liver
- ALT- breaks down food into energy. Low levels usually in blood but rises if there is liver damage
- Bilirubin- orange-yellow pigment when RBC breaks down. Bilirubin test is used to test the healthy condition of the liver.
- Outliers represented
- No null values in the data



	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Dataset
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

Preprocessing

- Gender is a categorical features, and so this is label encoded using label_encoder
- Albumin_and_Globulin_Ratio has 4 missing values, and so these rows are dropped
- Therefore, we are left with 579 samples and all numerical features



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
Age                           583 non-null int64
Gender                        583 non-null object
Total_Bilirubin               583 non-null float64
Direct_Bilirubin              583 non-null float64
Alkaline_Phosphotase          583 non-null int64
Alamine_Aminotransferase      583 non-null int64
Aspartate_Aminotransferase    583 non-null int64
Total_Protiens                583 non-null float64
Albumin                       583 non-null float64
Albumin_and_Globulin_Ratio    579 non-null float64
Dataset                       583 non-null int64
dtypes: float64(5), int64(5), object(1)
memory usage: 50.2+ KB
```

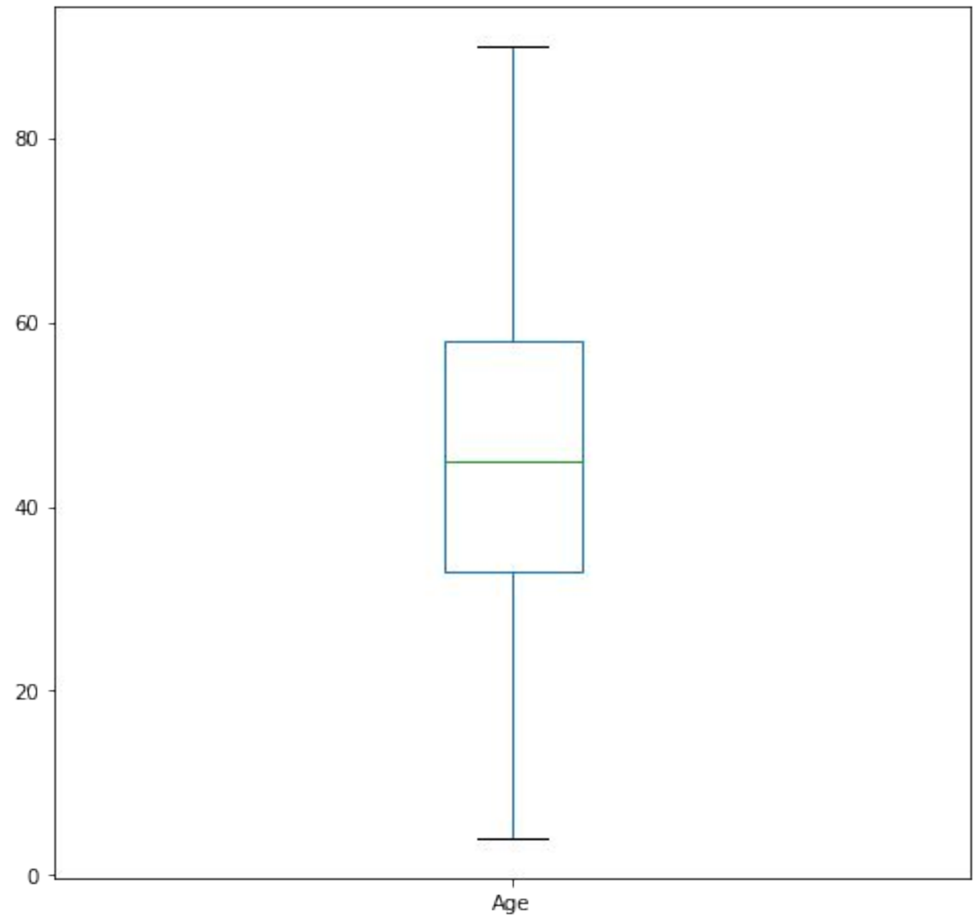
Exploratory Data Analysis

- Initial investigations on **data** is needed to discover patterns, spot anomalies,and test hypothesis using statistics and graphical representations.
- Libraries used - matplotlib, seaborn, pandas
- For every feature, box and whiskers plot, scatter plots and histograms are generated.

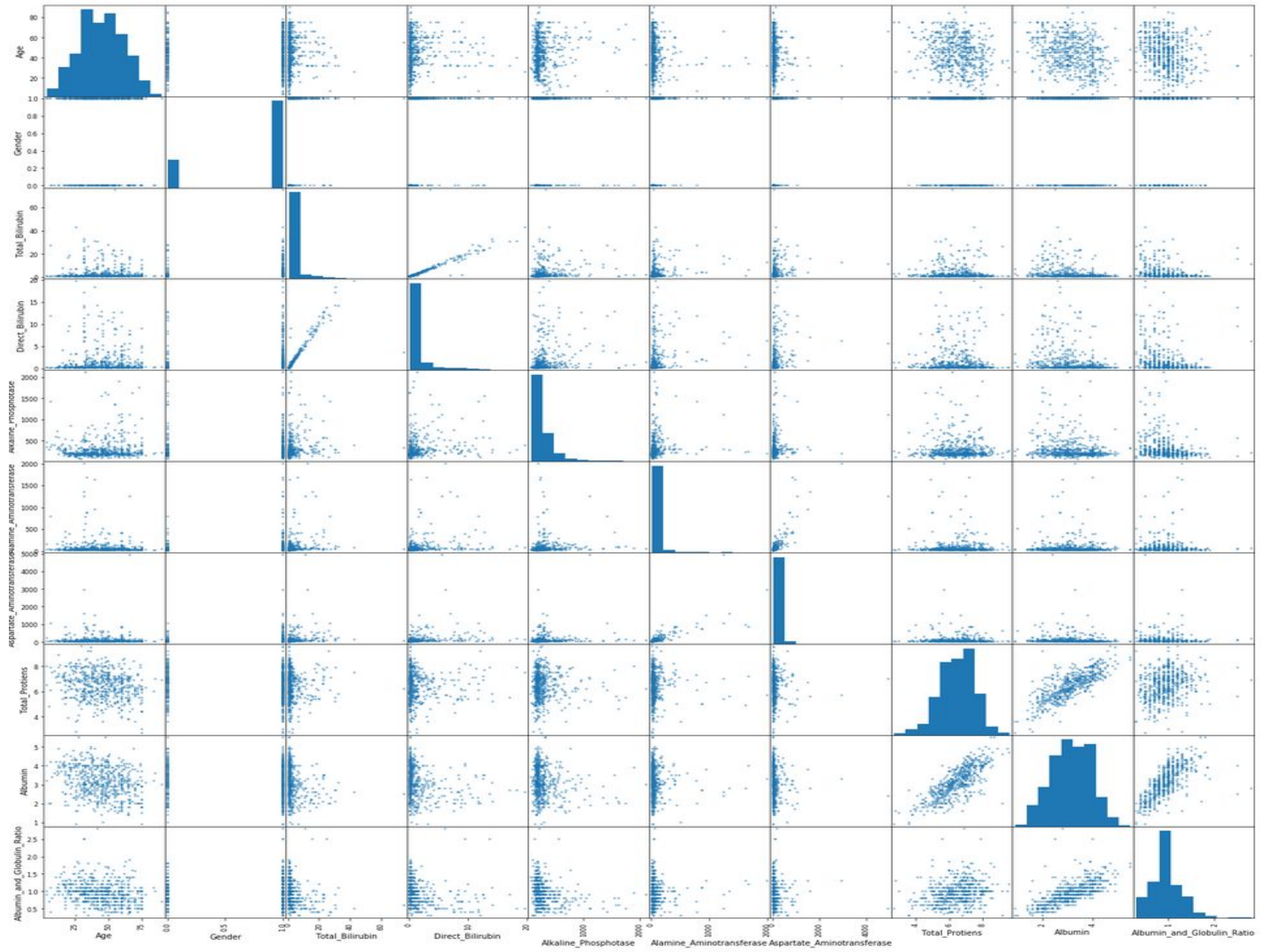
Exploratory Data Analysis

Box and whisker Plot

- it show variant the data is
- this is done using matplotlib library
- it helps us detect anomalies
- Those with lower variance consists of more outliers



Exploratory Data Analysis



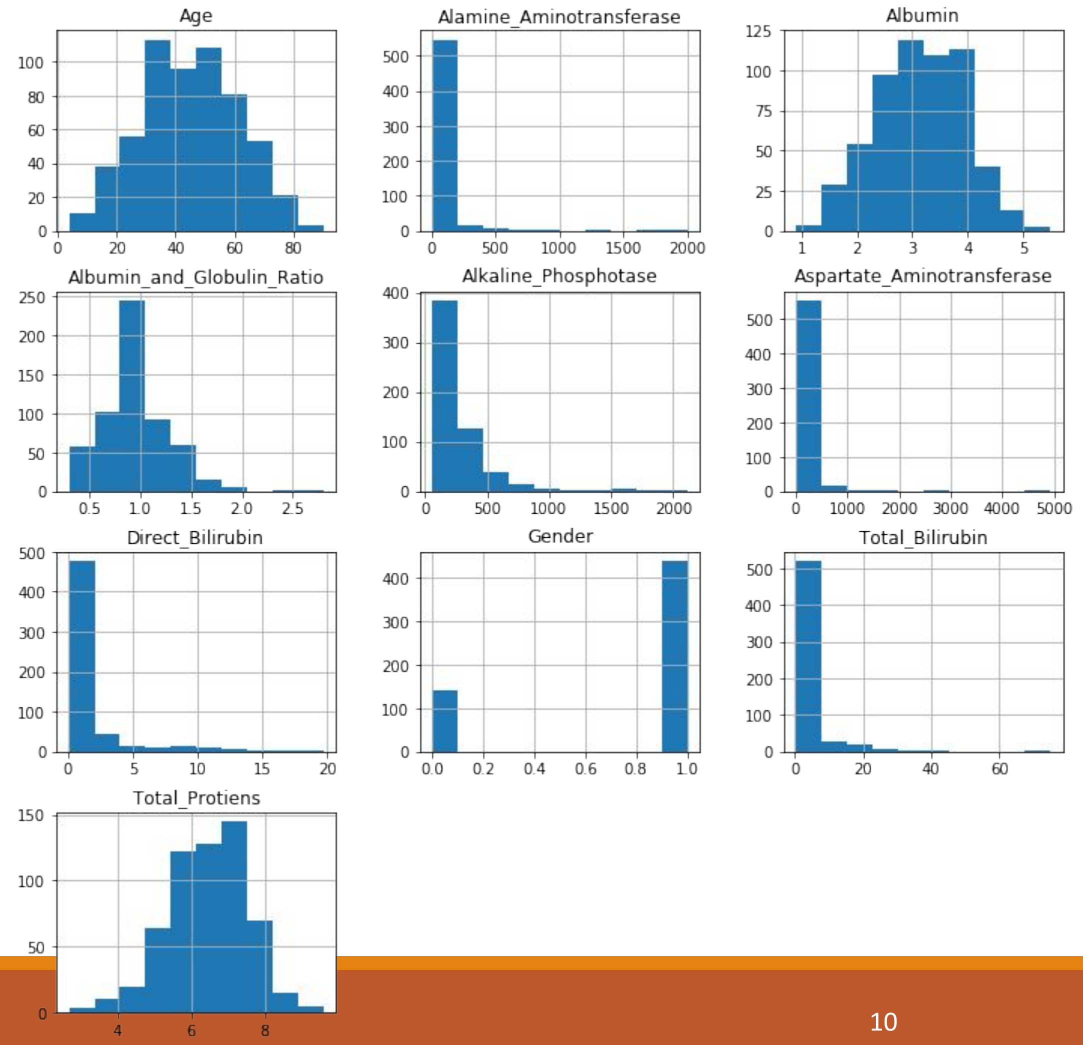
Scatter Plot

- It shows how scattered the data is
- this is done using pandas library
- the diagonals are since you cannot have scatter plots against the same features

Exploratory Data Analysis

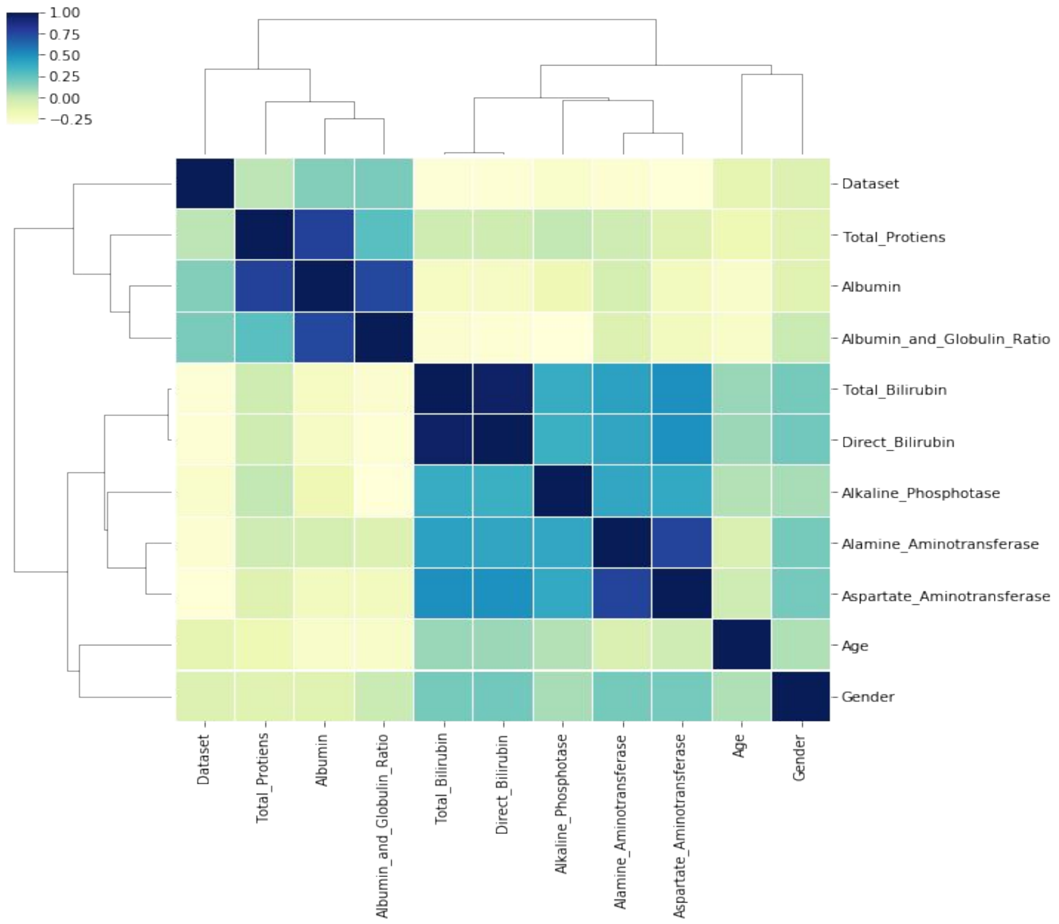
Histograms

- The histogram is used to see class distribution of the dataset
- this is done using matplotlib library



Feature Selection

- Correlation of every feature with each other is found
- A heatmap is drawn from it
- Seaborn library is used for this
- Features with higher correlation than 0.8 are dropped
- Therefore, Albumin, Aspartate_Aminotransferase and Total_Bilirubin are dropped



Principle Component Analysis

A standard way of reducing the dimension of a data is called principal component analysis (PCA). Geometrically speaking, PCA reduces the dimension of a dataset by squashing it onto a proper lower-dimensional line (or more generally a hyperplane, also often referred to as a subspace) which retains as much of the original data's defining characteristics as possible. The advantage in using PCA is that we get a set of uncorrelated features from a set of (possible) correlated features.

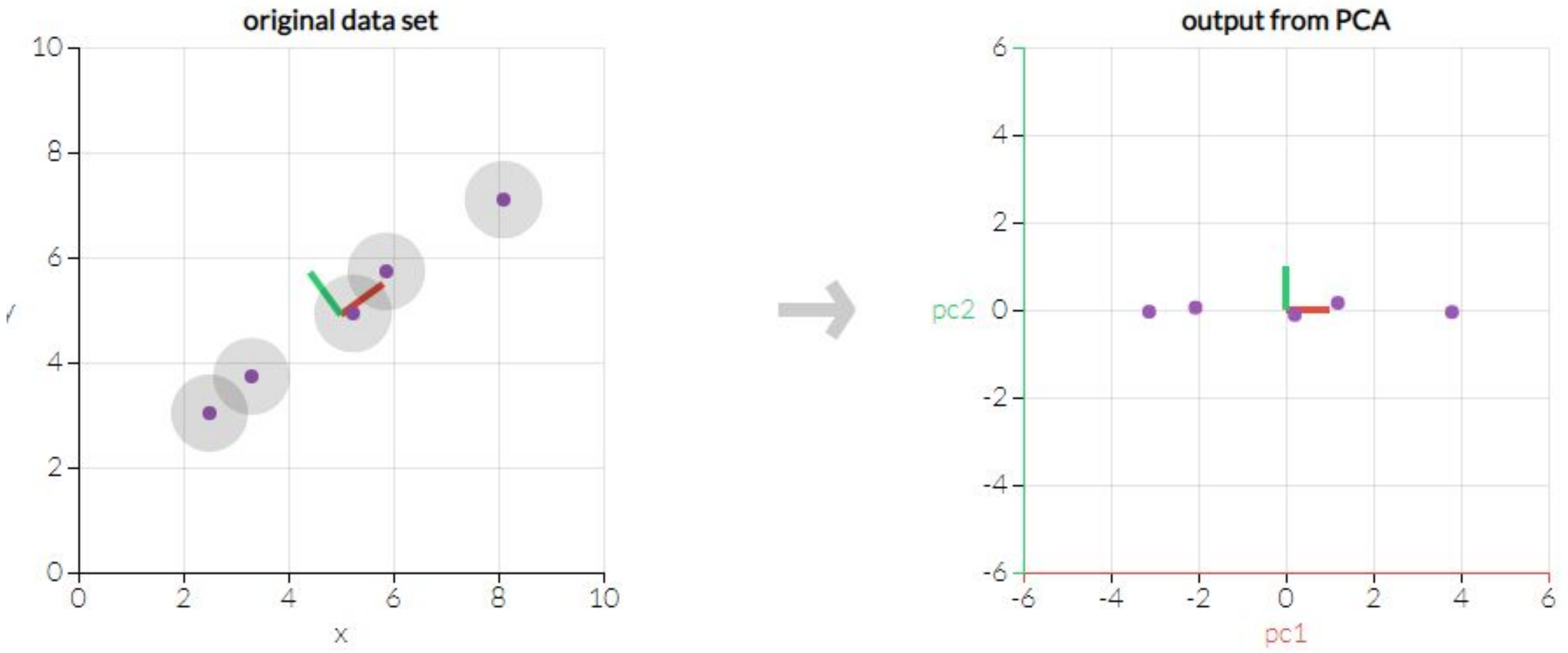
Libraries used

- from sklearn.decomposition import PCA

Parameters

- n_components = 3 -- This means that we are taking the first three principle components of the reduced space of features since, by theory, we know that the first 3 components consists of more than 95% of the information.

Principle Component Analysis



Grid Search

Definition:
Grid search is the process of performing hyper parameter tuning in order to determine the optimal values for a given model. This performs an exhaustive search based on the hyper parameter values specified on the dataset and outputs the best hyper parametre values

Libraries used

- from sklearn.model_selection import GridSearchCV

Algorithms used

- Random Forest
- Gradient Boosting
- Logistic regression
- Support vector machine

Each of the algorithms are run by default, then with PCA, then with both PCA and Grid Search to observe the changes in accuracy.

Then the most accurate among them is taken and put into Majority voting classifier

Libraries for the algorithms

- from sklearn.ensemble import GradientBoostingClassifier
- from sklearn.ensemble import RandomForestClassifier
- from sklearn.linear_model import LogisticRegression
- from sklearn.svm import SVC
- from sklearn.ensemble import VotingClassifier

Logistic Regression

"Logistic regression uses the logistic function, or the sigmoid function, to model output as binary values. It uses the maximum likelihood estimation (MLE) to obtain the model coefficients that relate predictors to the target."

Parameters used

- penalty : this is the regularization method than can be used. We tried both l1 and l2 regularization.
- C: A control variable that retains strength modification of Regularization. It's a penalty term, meant to disincentivize and regulate against Overfitting. values between 0.01 and 1 were used
- solver : We used liblinear and saga

Upon grid search, the best parameters were 0.1 for C, l2 for penalty and liblinear for solver.

SVM

“Support Vectors Classifier tries to find the best hyperplane to separate the different classes by maximizing the distance between sample points and the hyperplane.”

Parameters used

- gamma: gamma is a parameter for non linear hyperplanes. The higher the gamma value it tries to exactly fit the training data set.
  Gamma values used were between 0.1 and 1

- C : This is the penalty parameter of the error term. It controls the trade off between smooth decision boundary and classifying the training points correctly.
  C values used were between 0.1 and 1

- degree : this is the degree of the polynomial used to find the hyperplane to split the data. the degree chosen is 0.1 to 1

Upon grid search, the best parameters were 0.8 for C, 0.1 for Degree, 0.2 for Gamma

Random Forest

"Random forest is an extension of bagging method that builds a large set of de-correlated trees, and then averages their predictions.The input variables and features are selected at random with replacement to build the tree. This helps decrease the correlation of the trees."

Parameters used

- n_estimators: This is the number of trees in the forest. The number of trees used are 100 to 1000.
- min_samples_split: represents the minimum number of samples required to split an internal node. Our values were 1 to 9.
- max_depth : This is the number of levels of the tree. Our range was 1 to 9.

Upon grid search, the best parameters were 400 for n_estimators, 7 for min_samples_split, and max_depth to be 3.

Gradient Boosting

"Gradient boosting is an additive model which produces an algorithm in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion, and it generalizes them by optimization of a differentiable loss function."

Parameters used

- learning_rate : It shrinks the contribution of each tree. Values were taken between 0.01 and 0.1
- n_estimators: This is the number of trees in the forest. The number of trees used are 100 to 1000.
- min_samples_split: This represents the minimum number of samples required to split an internal node. Our values were 1 to 9.
- max_depth : This is the number of levels of the tree. Our range was 1 to 9.

Upon grid search, the best parameters were 0.03 for learning_rate, 1.0 for max depth, 0.1 for min samples leaf, 0.1 for min sample split and 70 for n_estimators.

Result Analysis

Libraries used

- from sklearn.model_selection import cross_val_score
- from sklearn.metrics import accuracy_score
- from sklearn.metrics import classification_report
- from sklearn.metrics import confusion_matrix
- import sklearn.metrics as metrics

Result Analysis

	Accuracy			
	Logistic Regression	SVM	Random Forest	Gradient Boosting
Original	0.7379310344827587	0.7310344827586207	0.7103448275862069	0.7172413793103448
PCA	0.7517241379310344	0.7310344827586207	0.7172413793103448	0.7034482758620689
PCA with Grid Search	0.7517241379310344	0.7310344827586207	0.7172413793103448	0.7034482758620689

Majority Voting Result

Original - 0.7172413793103448

PCA - 0.7172413793103448

PCA and grid search - 0.7379310344827587

Explanation about the code/library

1. We found enough resources like python library documentations to help with debugging our codes and tune our parameters.
2. We wrote our own code.
3. We collected ideas how to approach the problem from Kaggle Discussions and other Kernels
4. We spent time surfing Kaggle to see how other Data Scientists were solving the similar problems that we faced.

Conclusion

- Further work would be needed before this could be published. Better results might be achieved if neural networks was used.
- Through this project, we can get a better understanding of the theoretical and practical aspects of machine learning algorithms.
- Moreover, it would help doctors more accurately predict liver diseases, and thus save lives.

Reference

- S. Sontakke, J. Lohokare and R. Dani, "Diagnosis of liver diseases using machine learning," *2017 International Conference on Emerging Trends & Innovation in ICT (ICEI),* Pune, 2017, pp. 129-133. doi: 10.1109/ETIICT.2017.7977023
- https://www.kaggle.com/uciml/indian-liver-patient-records
- https://scikit-learn.org/