Department of Computer Science and Engineering
Asansol Engineering College
Vivekananda Sarani, Asansol - 713305

Project Report

a NEW PARADIGM of MACHINE LEARNING in BIOINFORMATICS

Submitted to Asansol Engineering College in Partial Fulfilment for the degree of
Bachelor of Technology
(Computer Science and Engineering)
of
Maulana Abul Kalam Azad University of Technology
KOLKATA – 700064

By

Arkadeep Bagal – 10800116108
Farooq Ansari – 10800116102
Rohan Kumar Singh – 10800116062
Rohit Kumar Majee – 10800116061

Under the guidance of
Mr. Sabyasachi Mukherjee
Assistant Professor
Department of Computer Science and Engineering

DEPARTMENTOF XXXX ENGINEERING

ASANSOL ENGINEERING COLLEGE

Vivekananda Sarani, Kanyapur, Asansol, West Bengal – 713305

Certificate of Recommendation

I hereby recommend that the thesis entitled **“A New Paradigm of Machine Learning in Bioinformatics”** submitted by

Arkadeep Bagal	(Roll No. 10800116108, Reg No. 161080110028),
Farooq Ansari	(Roll No. 10800116102, Reg No. 161080110034),
Rohan Kumar Singh	(Roll No. 10800116062, Reg. No. 161080110074)
Rohit Kumar Majee	(Roll No. 10800116063, Reg. No. 161080110075),

has been carried out under my guidance and supervision and may be accepted in partial fulfillment for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** of **Maulana Abul Kalam Azad University of Technology, Kolkata-700064.**

Mr. Sabyasachi Mukherjee
Assistant Professor
Dept. of Computer Science and Engineering
Asansol Engineering College
Asansol-713305, West Bengal.

Dr. Monish Chatterjee
Head of Department
Dept. of Computer Science and Engineering
Asansol Engineering College
Asansol-713305, West Bengal

DEPARTMENTOF XXX ENGINEERING

ASANSOL ENGINEERING COLLEGE

Vivekananda Sarani, Kanyapur, Asansol, West Bengal – 713305

Certificate of Approval

The forgoing thesis is hereby approved as creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose for which it is submitted.

…………………………………

Mr. Sabyasachi Mukherjee

Thesis Supervisor
Dept. of Computer Science and Engineering
Asansol Engineering College
Asansol-713305, West Bengal.

Acknowledgement

We express my sincere gratitude to **Mr. Sabyasachi Mukherjee**, our guide for his affectionate and valuable guidance without whose help the present work could not have been successful. We are also indebted to him as a teacher who introduced us to the topics related to the project. We thank **Dr. Monish Chatterjee**, the Head of the Department of Computer Science and Engineering for his constant encouragement and permission to work in the Departmental laboratory and use the various resources of the Department of CSE whenever required without which our work would not have been possible.

We are also grateful to the other teachers of the Department of CSE who have taken the pain of teaching us various core subjects of Computer Science for the last few years. We are also thankful to all other staffs of the Department for clearing the various technical doubts during the Laboratory Sessions which boosted our confidence.

Arkadeep Bagal
University Roll No. 10800116108
University Reg. No. 161080110028

Farooq Ansari
University Roll No. 10800116102
University Reg. No. 161080110034

Rohan Kumar Singh
University Roll No. 10800116062
University Reg. No. 161080110074

Rohit Kumar Majee
University Roll No. 10800116061
University Reg. No. 161080110075

Abstract

Chronic liver diseases impact humans around the globe. Through the research done in the medical field, the professional are able to diagnose it properly due to the indicators that the disease leaves in the body.

Through this project, we are trying to predict the diagnosis of liver disease based on those indicating parameters as accurately as possible. This would help in early diagnosis of liver disease in its preliminary stages, which would be really helpful in getting early treatment.

Another one of our goals for this project to come up with a Machine Learning Pipeline that includes a clear set of steps for the development of a prediction model.

Using the modern machine learning techniques implemented in Classification algorithms, we're trying to come up with an optimal model that provides maximum performance with feature engineering and hyper-parameter tuning on standard classification algorithm such as Logistic Regression.

The classification algorithm that is used here is Logistic Regression. We used a combinatorial approach for feature selection.

List of Contents

List of Figures

1. Preface

1.1. Introduction

Through this project, we explored the possibilities of extracting meaningful information from seemingly random and ordinary dataset. We tried to unleash the hidden potential of using our experiences to learn, and improve our methodology. No other framework in Computer Science's literature except Machine Learning and Data Science is more perfect for our goal in this regard.

Bioinformatics is a field of study that uses computation to extract knowledge from biological data. It includes the collection, storage, retrieval, manipulation and modelling of data for analysis, visualization or prediction through the development of algorithms and software.

Our implementation also explores the importance of extracting high performances from ordinary Machine Learning algorithms through proper data preprocessing and feature engineering. It would prove to be an important contribution in the ever-growing literature and research in the of aforementioned feature engineering and hyper-parameter tuning.

1.2. Motivation

Our purpose of delving into this project is to device a methodology that can be used as a tool in the field of Bioinformatics. This tool, which is our context a model that would predict whether a person has Liver Disease on not based on some of the biological test results that he can input.

This model that combines data science and biology, could be a stepping-stone in a world where more and more services are turning to Artificial Intelligence for error-free implementations. It would remove the human bias and error in our current diagnosis methodology and make way for truly automated systems.

For fields as severe, important and critical such as disease diagnosis, we can combine years of developmental research in the field of predictive algorithms with the ever growing computational power of modern system to come up with state-of-the-art prediction models that provide impeccable essential services.

1.3. Description

This project mainly deals with the ways of how we can improve upon the existing models for this particular dataset implemented using Logistic Regression algorithm, and achieving a better Test Set accuracy than them.

To make our Logistic Regression model robust, we have developed a pipeline of various data science techniques, which will be discussed in detail.

One particularly important stage of our pipeline is Hyper-parameter tuning, for which we have selected Genetic Algorithm as our method.

As the field of study has observed, the proper way to achieving a well performing model is to feed it the most cleaned and processed dataset. Feature Engineering in that context is a field on itself. It has been observed in the past that for most, if not all, of the popular machine learning dataset, some noise in form of either missing or false values, or irrelevant features are present.

This particular dataset that we used, was mostly used in literature in its raw form i.e. without the exclusion of any of its features in the input dataset. What we have tried to do is select the best possible set of features from this dataset that outputs the highest accuracy and therefore better performance.

Our resolve is that even a small dataset with proper feature selection and engineering and with a simple machine learning algorithm such as Logistic Regression can result in high model performance with proper feature selection and hyper-parameter tuning techniques.

2. Literature Review

2.1. General

While doing our background research for this project, we went through various research papers that specifically dealt with the dataset that we used. We also looked into the research papers that used Genetic Algorithm for both feature selection and hyper-parameter tuning.

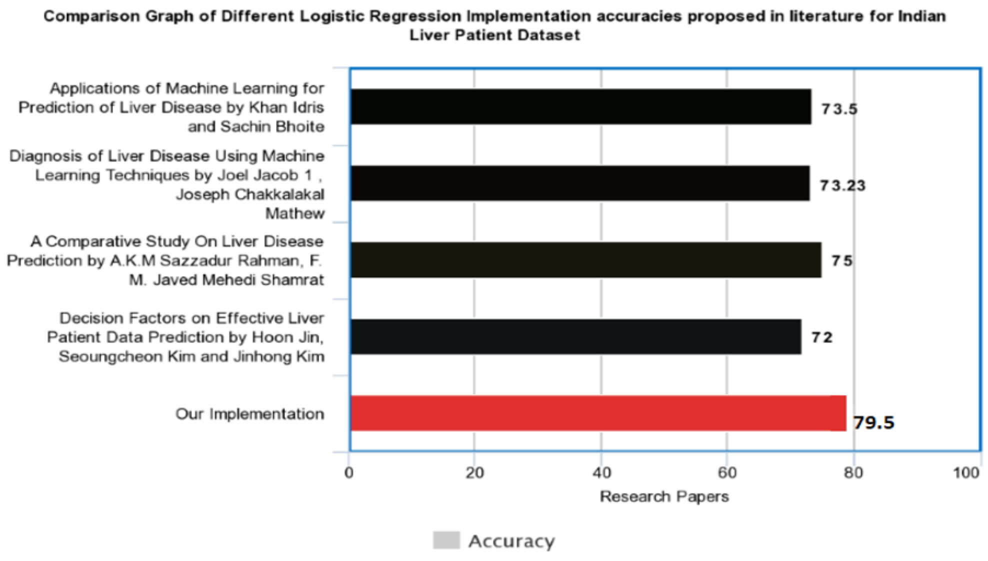
We were, however, unable to find any pipeline implementation such as ours that, for example, used Genetic Algorithm as a method of hyper-parameter tuning for Logistic Regression.

2.2. Comparative Study

Our implementation of the pipeline provides Logistic Regression Model that performs better than any other such Logistic Regression Classification Model implemented for this particular dataset.

The dataset was divided into a training set and test set, and our metric was test set accuracy percentage i.e. out of a given data inputs, how many one is able to diagnose accurately.

Figure 2.2.1 Comparative Study



3. Related Theories and Algorithms

3.1. Machine Learning

Machine Learning is a methodology that enables IT systems to recognize patterns on the basis of existing algorithms and data sets and to develop adequate solution concepts. Therefore, in Machine Learning, artificial knowledge is generated on the basis of experience.

Arthur Samuel's Definition:

Machine Learning is a science of getting computers to learn without being explicitly programmed.

Tom Mitchell's Definition:

A computer program is said to learn from experience (E) with respect to some task (T) and some performance measure (P), if its performance measure on T, as measured by P, improves with E.

Advantages of Machine Learning:

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviours and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

As Machine Leaning Algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. As a contextual example, if we’re making a disease predictor such as ours, we can keep increasing the dataset that the model trains on by adding data repositories from different sources and our model will keep getting better.

Self-learning machines can perform complex tasks. These include, for example, the recognition of error patterns. This is a major advantage, especially in areas such as the manufacturing industry. The industry relies on continuous and error-free production. While even experts often cannot be sure where and by which correlation a production error in a plant fleet arises, Machine Learning offers the possibility to identify the error early - this saves downtimes and money.

Self-learning programs are now also used in the medical field. In the future, after "consuming" huge amounts of data (medical publications, studies, etc.), apps will be able to warn a patient in case his doctor wants to prescribe a drug that he cannot tolerate. This "knowledge" also means that the app can propose alternative options which for example also take into account the genetic requirements of the respective patient.

Machine Learning algorithms are of following types.

3.1.1. Supervised Learning

Supervised learning is when the model is getting trained on a labelled dataset. **Labelled** dataset is one which have both input and output parameters. In this type of learning both training and validation datasets are labelled.

Example: Linear Regression, Logistic Regression

Types of Supervised Learning

Supervised learning problems further grouped into Classification and Regression problems.

3.1.1.1 Classification

It is a Supervised Learning task where output is having defined labels (discrete value).

Example:
Given a patient with a tumor, we have to predict whether the tumor is malignant or benign.

3.1.1.2 Regression

Regression predictive modeling is the task of approximating a mapping function (f) from input variables (X) to a continuous output variable (y).

Example:
Given data about the size of the house on the real estate market, try to predict their price. Price as a function of size is a continuous output, so this is a regression problem.

3.1.2. Unsupervised Learning

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.

Types of Unsupervised Learning

Unsupervised learning problems further grouped into clustering and association problems.

3.1.2.1 Clustering

Clustering is an important concept when it comes to unsupervised learning. It mainly deals with finding a structure or pattern in a collection of uncategorized data. Clustering algorithms will process your data and find natural clusters (groups) if they exist in the data. You can also modify how many clusters your algorithms should identify. It allows you to adjust the granularity of these groups.

K-means clustering

K means it is an iterative clustering algorithm which helps you to find the highest value for every iteration. Initially, the desired number of clusters are selected. In this clustering method, you need to cluster the data points into k groups. A larger k means smaller groups with more granularity in the same way. A lower k means larger groups with less granularity.

The output of the algorithm is a group of "labels." It assigns data point to one of the k groups. In k-means clustering, each group is defined by creating a centroid for each group. The centroids are like the heart of the cluster, which captures the points closest to them and adds them to the cluster.

3.2. Logistic Regression

It is a classification algorithm. Instead of fitting a straight line or hyperplane, the logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1.

Here, theta (θ) is the parameter array that is attached with every data sample, or the features.

x is the array of values that provides the weight for every values. This is the variable that gets adjusted in every iteration of Gradient Descent and is optimized.

In such problems we have to predict whether the result is 0 or 1.

Logistic Regression: $0 <= h\ \theta(x) <= 1$

$Y \in \{0,1\}$
0: for negative class (Diagnosis: safe)
1: for positive class (Diagnosis: diseased)

Hypothesis representation for Logistic regression

h $\theta(x) = (\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \ldots)$
h $\theta(x) = \theta^T x$ such that $0 <= h\ \theta(x) <= 1$

This is accomplished by putting $\theta^T x$ into logistic function.

3.2.1 Sigmoid Function

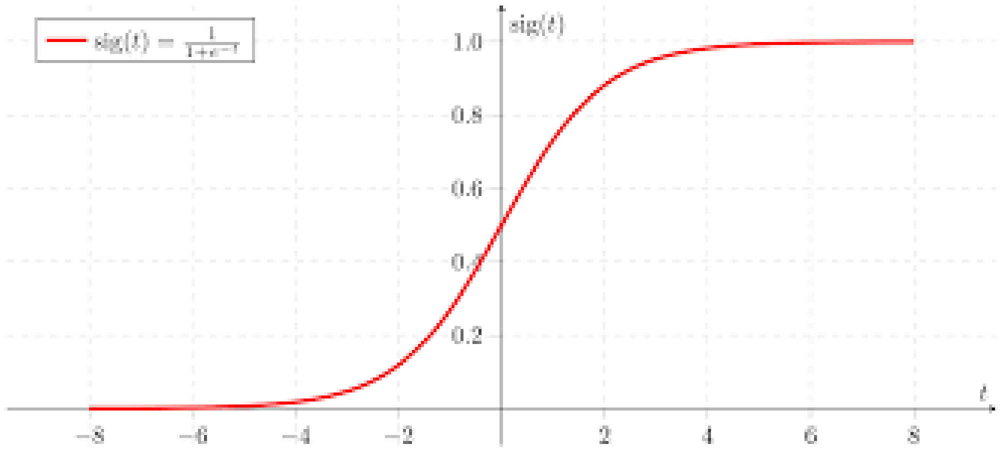
New form uses sigmoidal function or logistic function

$h(\theta) = g(\theta^T x)$

$g(z) = 1/1+e^z$ where $z = \theta^T x$

the function g(z) looks like this:

Figure 3.2.1 Sigmoid Function



Threshold classifier output h $\theta(x)$ at 0.5:
If h $\theta(x) >= 0.5$, predict y=1
If h $\theta(x) <= 0.5$, predict y=0

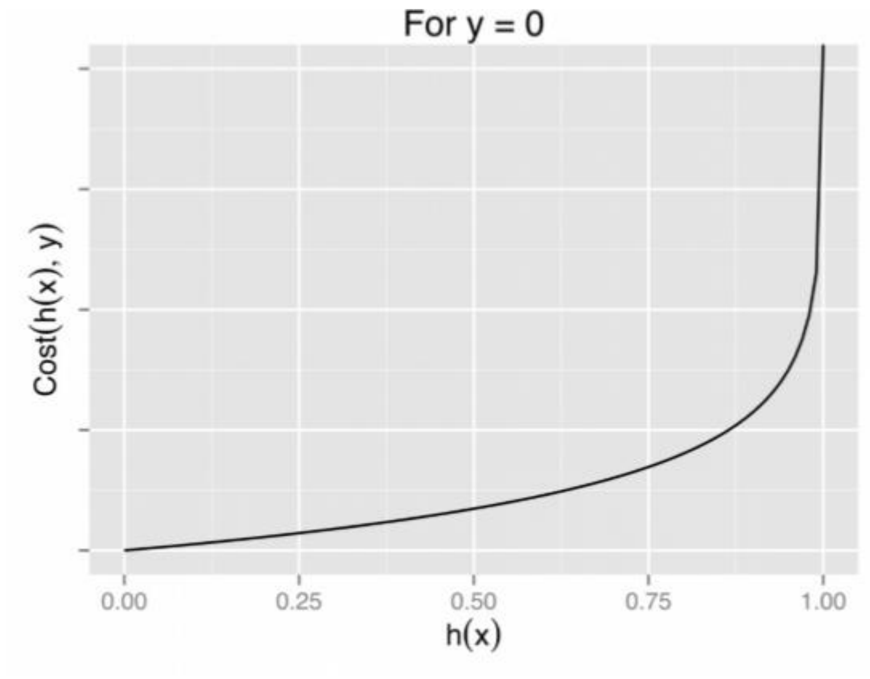
Now, if hθ (x) will give the probability that our output is 1 or 0.
Example: 0.7 means the probability of having output 1 is 70%

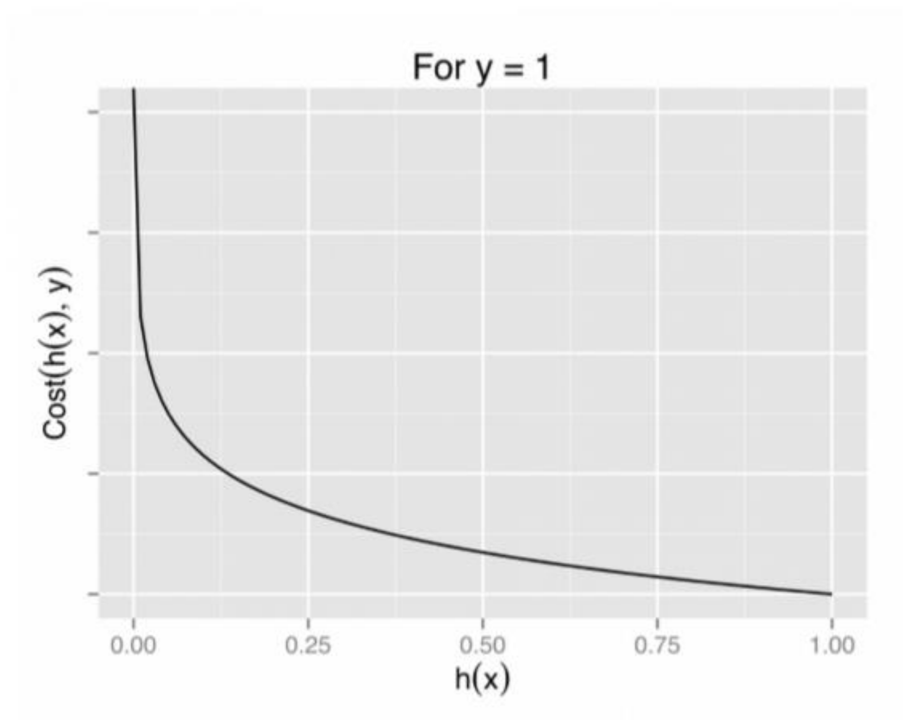
3.2.2 Cost Function

The Cost Function represent the optimization objective. Our aim is to minimize this cost function so that we can develop an accurate model with minimum error.

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$$\begin{aligned} J(\theta) &= \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) \\ &= -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))] \end{aligned}$$





3.2.3 Gradient Descent

Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. In machine learning, we use gradient descent to update the parameters of our model. The primary goal of Gradient Descent is to minimize the cost function. The value of alpha here is used to subtract at each step in the direction of Gradient J(θ).

Want $\min_\theta J(\theta)$:

Repeat $\{$

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update all θ_j)

$\}$

3.3. Genetic Algorithms

A Genetic Algorithms is a type of optimization algorithms.
It is a stochastic method for function optimization based on the mechanics of natural genetics and biological evolution.

Genetic Algorithm is inspired by Darwin's theory of natural evolution, which reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation and in the process of breeding random mutations are also introduced which is in compliance with Darwin's theory.

The process of natural selection starts with the selection of fittest individuals from a population. They produce offspring which inherit the characteristics of the parents and will be added to the next generation. If parents have better fitness, their offspring will be better than parents and have a better chance at surviving. This process keeps on iterating and at the end, a generation with the fittest individuals will be found.

Components of Genetic Algorithm.

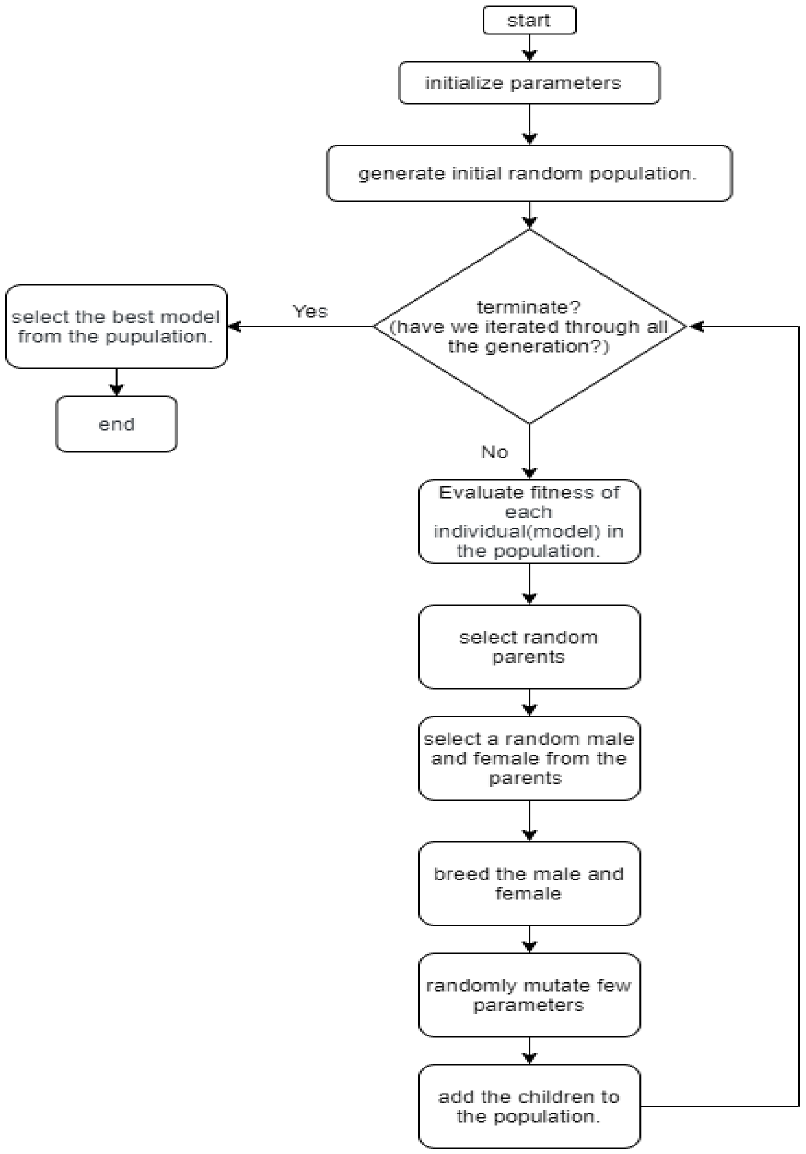
- Random initial population.
- Fitness Function
- Selection of random parents
- Breeding
- Mutation

3.3.1. Random initial population

The process begins with a set of individuals which is called a Population.
In the context of our implementation, each individual corresponds to a model. An each individual(model) is characterized by a set of parameters known as Genes.
for example, each of our individual will be characterized by the random values of the following parameters.

- ➔ C
- ➔ tol
- ➔ penalty
- ➔ solver

Figure 3.3.1: Genetic Algorithm


start
initialize parameters
generate initial random population.
terminate?
(have we iterated through all the generation?)
Yes
select the best model from the pupulation.
end
No
Evaluate fitness of each individual(model) in the population.
select random parents
select a random male and female from the parents
breed the male and female
randomly mutate few parameters
add the children to the population.

Random values for the parameters are selected from their corresponding range specified below.

```
 Params = {
          'C' : [0.1, 1, 10, 100, 1000, 10000, 100000],
          'tol' : [0.001, 0.01, 0.1, 1, 10, 100],
        'penalty': ['l1', 'l2'],
        'solver': ['liblinear', 'lbfgs','newton-cg']
   }
```

3.3.2. Fitness Function

The fitness function determines how fit an individual is (the ability of an individual to compete with other individuals). It gives a fitness score to each individual. The probability that an individual will be selected for reproduction is based on its fitness score.

In our implementation, the degree of fitness for a particular individual (model) is based on the accuracy attribute of the model or individual.

3.3.3. Selection of Random Parents

The idea of selection phase is to select the fittest individuals and let them pass their genes to the next generation.
Individuals with high fitness have more chance to be selected for breeding.
In our implementation we select a group of individuals from the papulation based on their fitness scores. The selected group of individuals are termed parents

A pair of two parents (one male and one female) are randomly selected from the available parents and are allowed to breed.

```
Male = random.randint(0, parents_length-1)
female = random.randint(0, parents_length-1)
```

3.3.4. Breeding

Breeding is the most significant phase in a genetic algorithm. For each pair of parents to be mated, two children are being produced.

Where each child will have their values of the parameters randomly inherited from either of the two parent's values for the corresponding parameters.

```
For param in self.params:
            child[param] = random.choice(
                [mother.model[param], father.model[param]]
            )
…
…
```

3.3.5. Mutation

In certain offspring or child formed, a random parameter is allowed to mutate, in other words, a random parameter is selected and a random value is assigned to that selected parameter of the offspring.

```
 # Choose a random key.
   Mutation = random.choice(list(self.params.keys()))

 # Mutate one of the params.
   Model.model[mutation] = random.choice(self.params[mutation])
```

The decision about whether or no mutation will take place is again randomly decided.

```
If self.mutate_chance > random.random():
      model = self.mutate(model)
```

Here are some of the terms that further explains the algorithm nomenclature.

- *Population*

  It is the collection of different Logistic Regression models that are randomly selected at the start of a generation. This continues to grow as more and more breeding happens in every generation.

- *Chromosome*

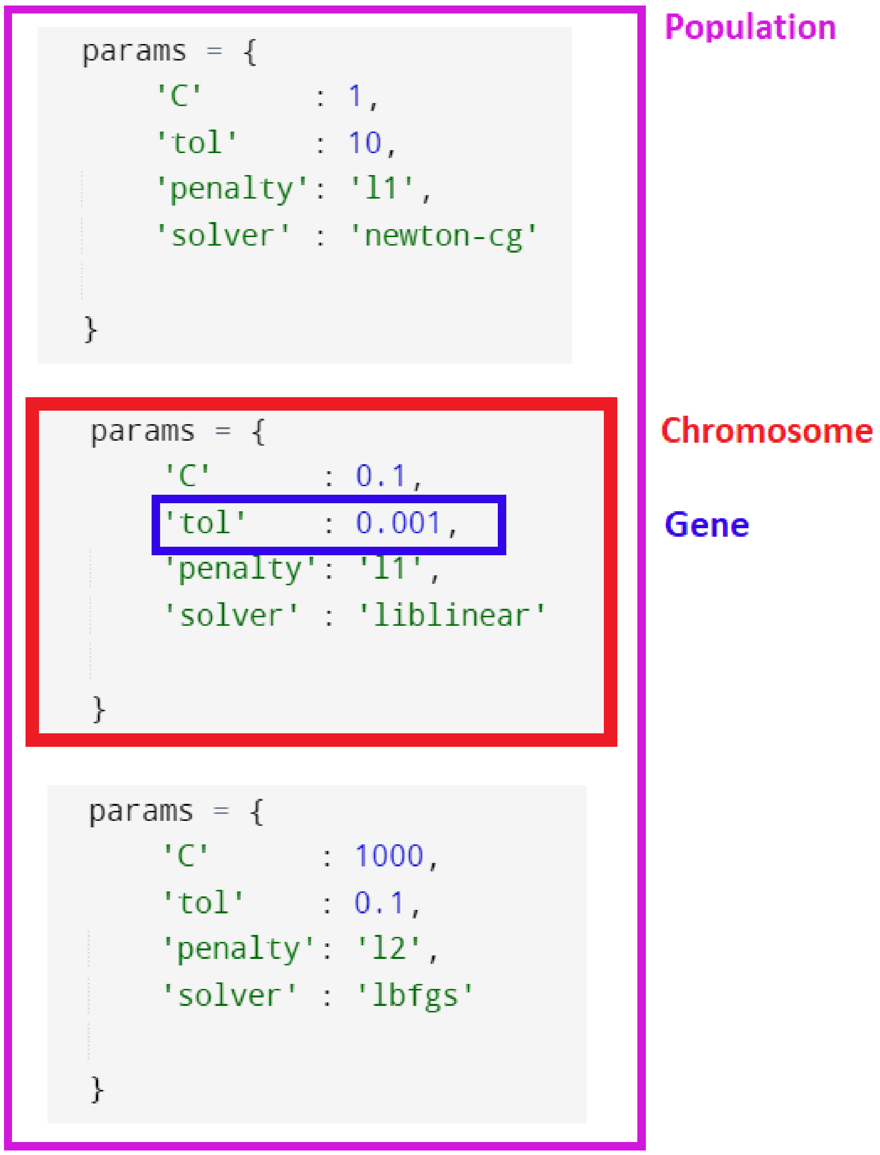
  Any single one of the Logistic Regression models in the population is the chromosome. As the characteristics of every human being is determined by their DNA, and by extension, the Chromosomes in the

DNA. Similarly here the characteristics of a model is determined by the set of parameters chosen.

- *Gene*

Any single one of the parameters in the Chromosome is knows as the Gene here. It this context, they are parameters like *C*, *tol*, *penalty* and *solver*.

Figure 3.3.2 Nomenclature Explained



```
params = {
    'C'       : 1,
    'tol'     : 10,
    'penalty': 'l1',
    'solver'  : 'newton-cg'

}
```

```
params = {
    'C'       : 0.1,
    'tol'     : 0.001,
    'penalty': 'l1',
    'solver'  : 'liblinear'

}
```

```
params = {
    'C'       : 1000,
    'tol'     : 0.1,
    'penalty': 'l2',
    'solver'  : 'lbfgs'

}
```

4. Dataset

4.1. Description

The Dataset that we would be using is called *Indian Liver Patient Dataset*. It is a popular dataset referenced in literature various times through different research papers on the classification algorithms. There are total 583 records out of which 416 have the disease and 167 do not.

The features present in the dataset are as follows:

1. **AGE -** Age of the patient
2. **GENDER -** Gender of the patient
3. **TB** - Total Bilirubin
4. **DB** - Direct Bilirubin
5. **ALKPHOS -** Alkaline Phosphotase
6. **SGPT** - Alamine Aminotransferase
7. **SGOT** - Aspartate Aminotransferase
8. **TP** - Total Proteins
9. **ALB** - Albumin
10. **A/G** - Ratio Albumin and Globulin Ratio
11. **Target -** Selector field used to split the data into two sets (labeled by the experts)

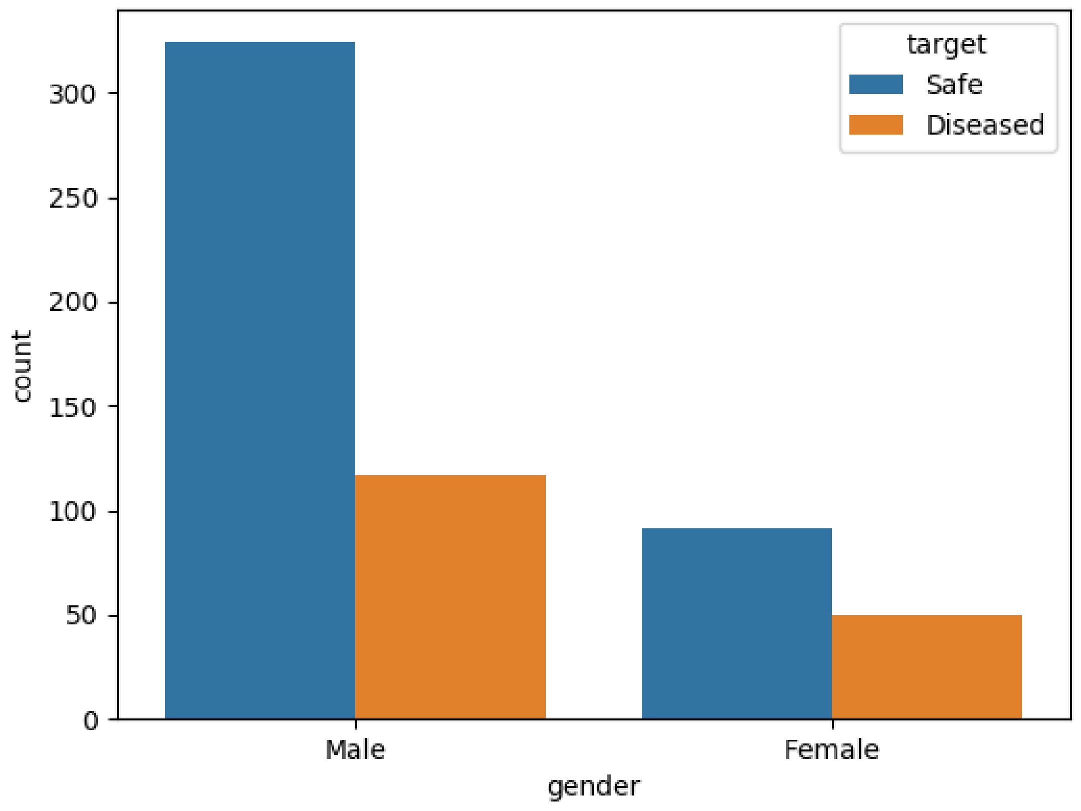
4.2. Explanatory Visual Analysis

This is a technique that helps people understand and make sense of large amounts of **data**. The **data** is often displayed in a story format that visualizes patterns, trends and correlations that may otherwise go unnoticed.

Let us explore the dataset and its discrete and continuous values to see if we can find some meaningful relationships among the features.

For **GENDER**, which is a discrete feature. We have made a countplot (Figure 4.2.1) that depicts how many of both the genders present in the dataset have the disease and how many are safe. As visible from the plot, among the diseased, there's a majority of Males. The whole dataset is overall biased with a higher count
Of males.

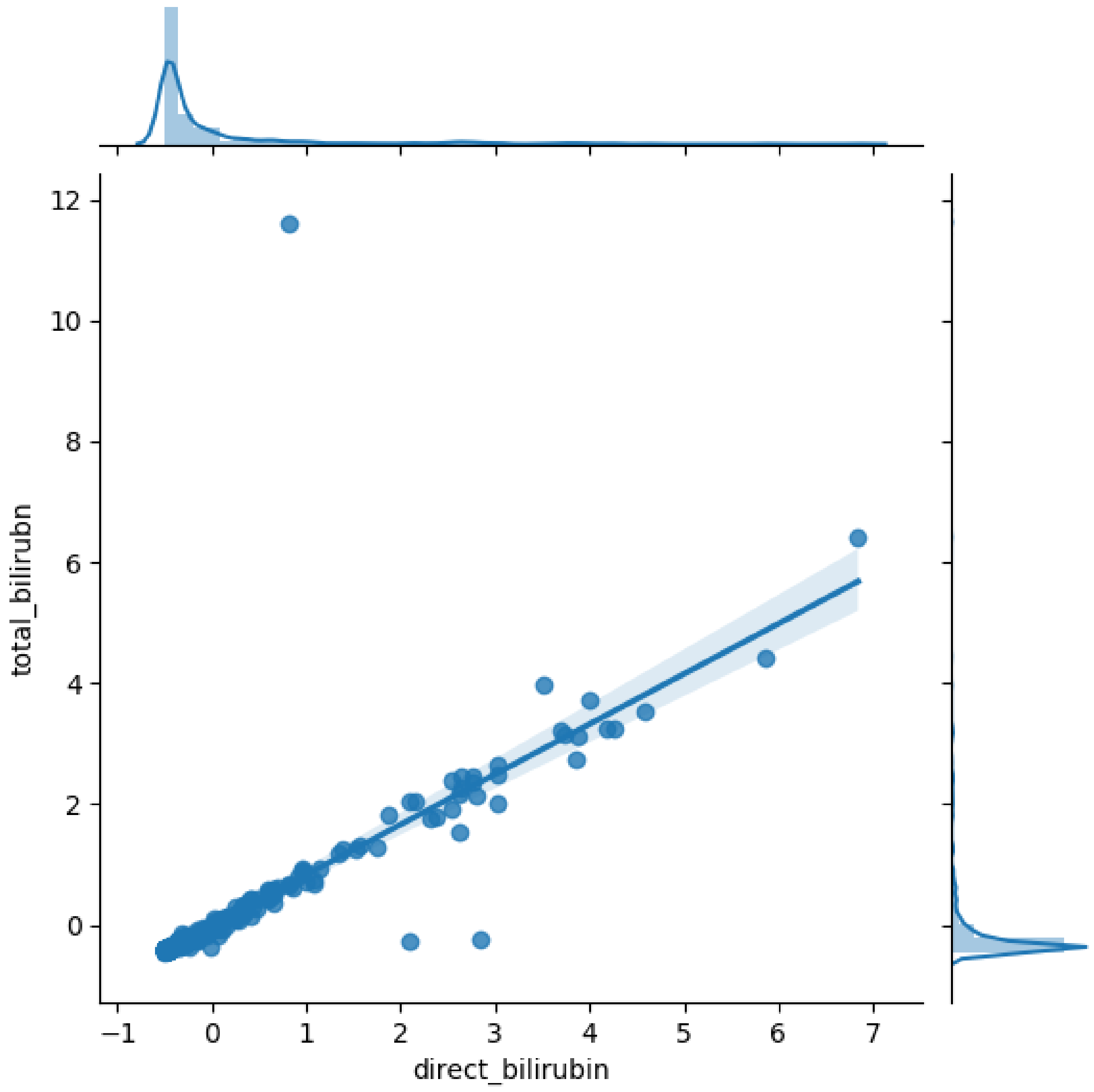
Figure 4.2.1: Gender Distribution



We plotted the relationship between **TB** (Total Bilirubin) and **DB** (Direct Bilirubin) using a Scatterplot.

We found a linear relationship between them.

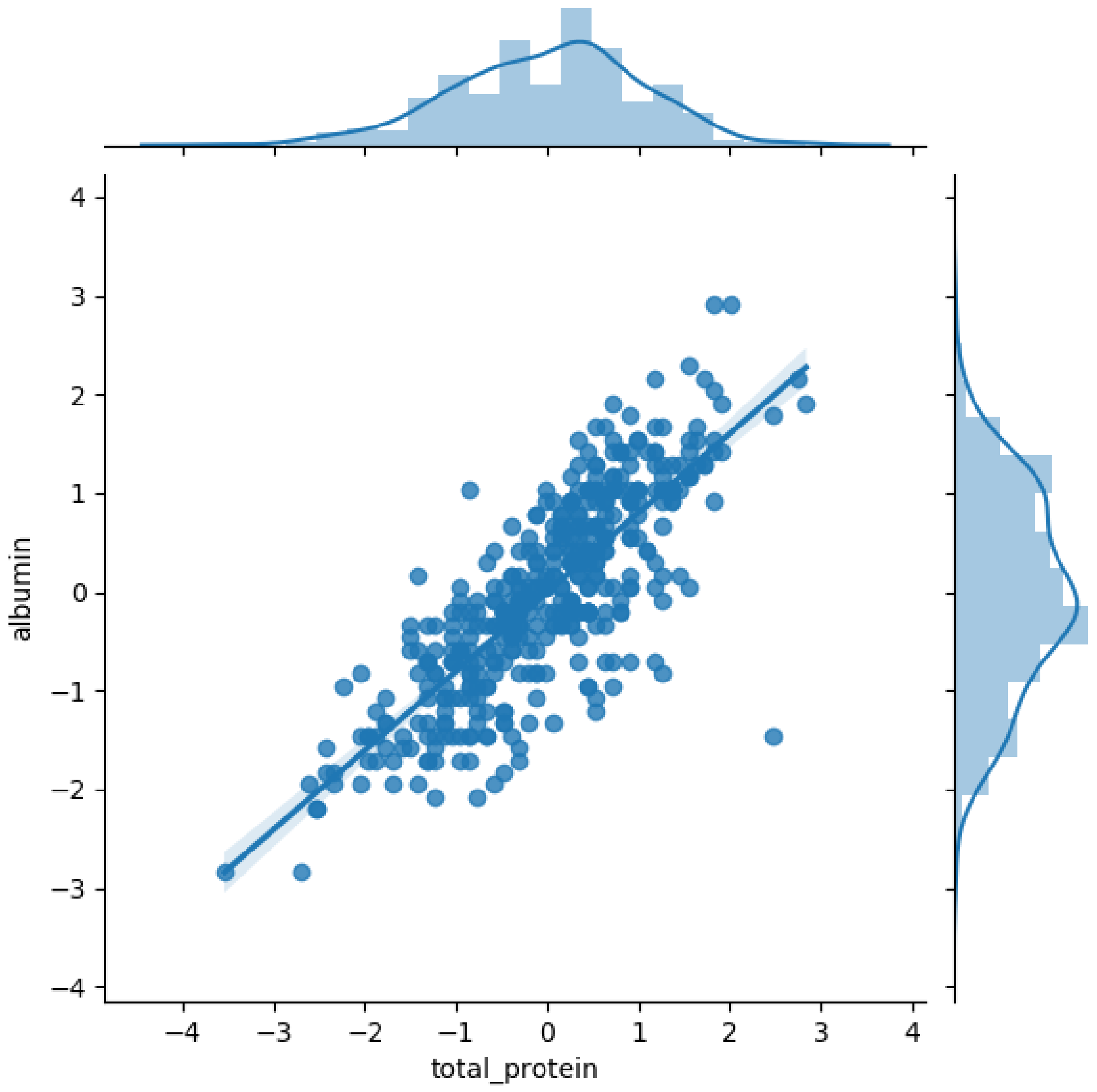
Figure 4.2.2: TB vs DB



We plotted the relationship between **TP** (Total Protein) and **ALB** (Albumin) using a Scatterplot.

We found a somewhat linear relationship between them.

Figure 4.2.3: TP vs ALB

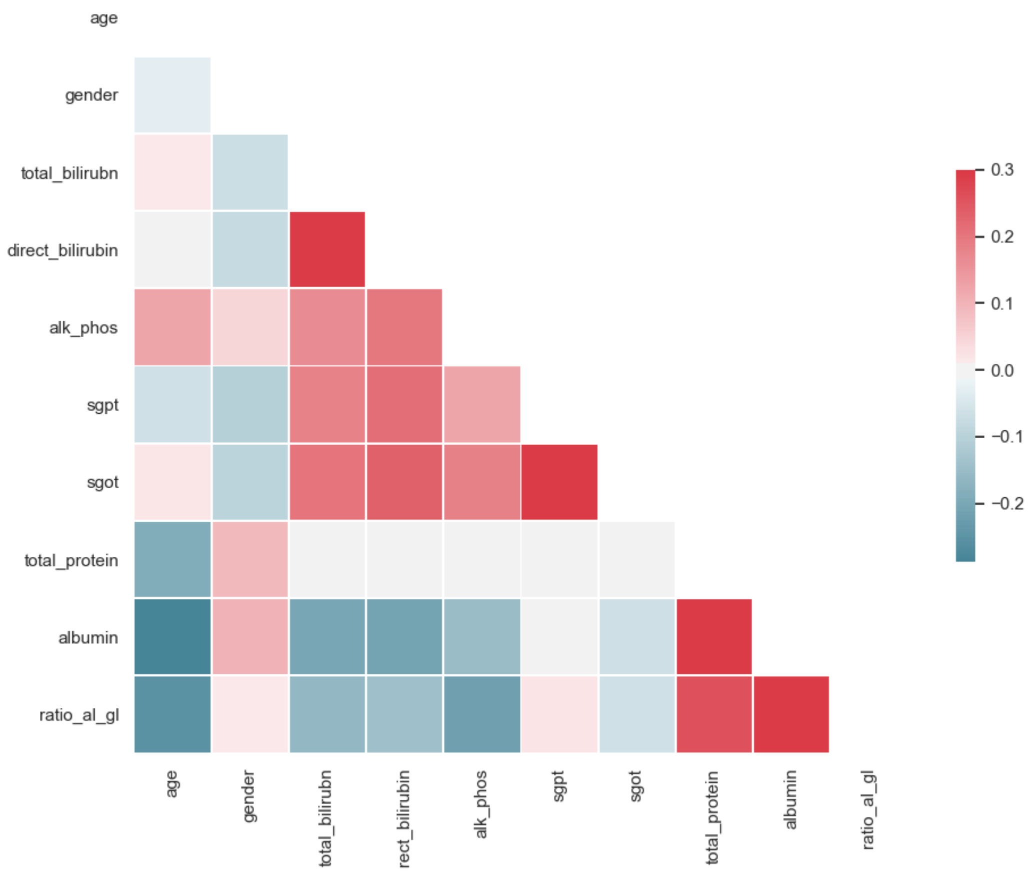


Here's a visualization of the correlation matrix among the features.

As we can observe, there's strong relationship between some combinations of features, such as Direct Bilirubin with Total Bilirubin, SGPT with SGOT, Albumin with Total Protein and Albumin-Globulin Ratio with Albumin.

This information is going to be relevant in further analysis.

Figure 4.2.4: Correlation Matrix

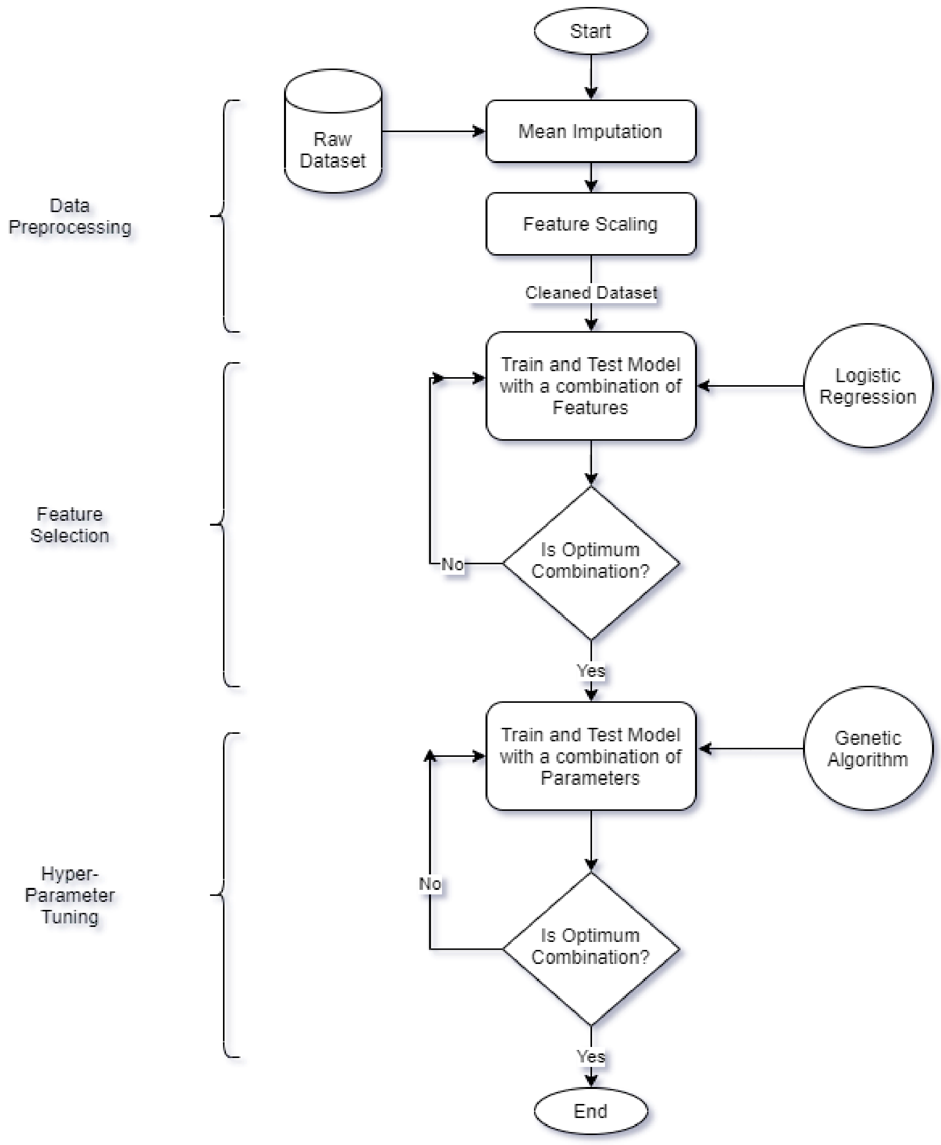


5. Proposed Pipeline

5.1.Description

A Machine Learning Pipeline is a well-defined set of steps that usually include data cleaning and preprocessing methods, as well feature engineering and model selection methodologies that can be used in any dataset. The final output of any such pipeline is an optimal machine learning classification or regression model that is fine-tuned to perform most efficiently for the input dataset.

Figure 5.1.1 Pipeline



5.2. Components

The following are the components of the pipeline that we have structured for this project.

5.2.1. Data Pre-processing

In this step, the input data is cleaned and processed in a way that enables our machine learning model to run efficiently.

There are several steps that we took in this process.

a. The dataset was encoded to turn the values in the features that are of String datatype to Integer values. Since Logistic Regression is mathematical algorithm, this steps becomes necessary if we want our dataset to be processed.

b. The values that were corrupted were removed, and missing values were replaced by using Imputation. The method that we used was Mean Imputation where the mean of the feature or column is inserted at the place of the missing value.

c. The dataset was scaled using a standard scaler that standardizes features by removing the mean and scaling to unit variance.

   Since all the fields are measured and documented in different units, we are using a mean normalization technique to put all the values in the same scale for better performance of our model.

   The standard score of a sample x is calculated as: $z = (x - u) / s$ where u is the mean of the training samples and s is the standard deviation of the training samples.

5.2.2. Feature Selection

Many of the popular dataset, though painstakingly acquired through different means and put together to further the research happening in the field of Data Analysis and prediction, have fields and values that might not prove relevant or useful for the optimum prediction model.

Here, we tried to find the optimum combination among the different features of the dataset that results in the maximum accuracy of the model.

For that we are iterating through all the possible combination of all the possible sizes of the features, and choosing the combination that results in the optimum accuracy.

Following is the pseudo code for our approach.

```
data = open ('dataset.csv’)

X = data.features
y = data.target

max_columns = data.feature_length
max_accuracy = -1
no_of_columns = 1

while no_of_columns < max_columns
loop
    all_combinations = every combination of    that
size

    # If no_of_columns = 3, then
    # all_combinations = [1, 2, 3], [2, 3, 4], [5,
6, 7]

    for combination in all_combinations
    loop

         X = X[combination]

         # selects just those columns from X

         Model = LogisticRegressionModel(X, y)
         accuracy = model.accuracy()
         if accuracy > max_accuracy
            max_accuracy = accuracy
            best_combination = combination

    no_of_columns++

Return max_accuracy, best combination
```

Through this methodology, we found that the combination of features **AGE, GENDER, DB, SGPT, SGOT** and **ALB** are the ones that performed the best. It can be argued that the rest of the features provides either no help or worse, error to our prediction model.

Interestingly, this result was also consistent with our explanatory visual analysis where we found correlation of left-out features such as **TB** and **A/G** with the selected features, therefore rendering them of not much use.

5.2.3. Model Optimization

The classification that used that predicts whether a data sample, or a person, has liver disease or not is Logistic Regression.

The Logistic Regression class that we imported from the popular 'sklearn' library has many parameter in its Constructor that can be tuned to improve the prediction accuracy. Some of them being:

C - Inverse of regularization strength; must be a positive float. Smaller values specify stronger regularization.

Tol - Tolerance for stopping criteria.

Penalty - Used to specify the norm used in the penalization. It is also known as method of regularization. It is the process which regularizes or shrinks the coefficients towards zero to prevent overfitting.

Solver - Algorithm to use in the optimization problem.

By using **Genetic Algorithm** that has been mentioned earlier in the report, we carried out the process of hyper-parameter tuning that searches for the optimum combination of the aforementioned parameters.

Our results showed the combination of values of **100000, 0.1, 'L1'** and **'liblinear'** of parameters **C, Tol, Penalty** and **Solver** respectively performed the best of the highest accuracy score.

Figure 5.2.3.1 Pseudo Code for Genetic Algorithm

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

6. Simulation of Hypotheses

6.1. Setup

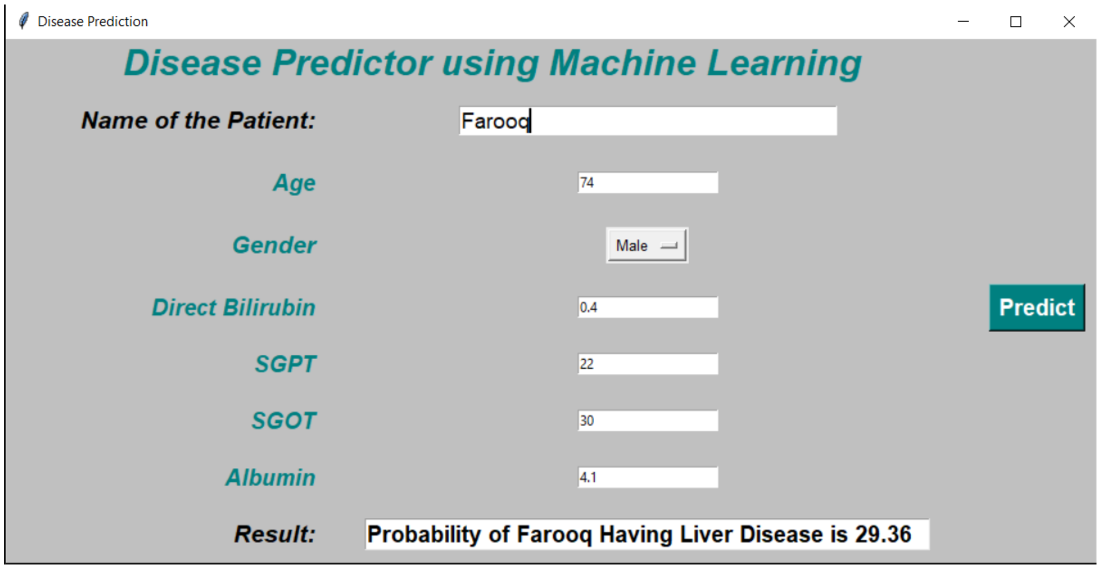
For our implementation, we used the following software and hardware components and tools.

a. A System with Intel-8th Generation processor and 8 GB of RAM.
b. Text Editor named Sublime Text for coding.
c. Python Programming Language.
d. Popular Python libraries and modules such as **pandas, sklearn, numpy** and **tkinter.**
e. GitHub as a version control and code collaboration tool, where all of our code is hosted.

6.2. Graphical User Interface

In order to make our prediction model interactive and responsive to a query based interface, we used tkinter to make a Graphical User Interface to the model. Here, it is possible to input a variety of details about any patient and then the user will be informed of the percentage likelihood of that patient developing a liver disease.

Figure 6.2.1: GUI



Name of the Patient: Farooq

Age: 74

Gender: Male

Direct Bilirubin: 0.4

SGPT: 22

SGOT: 30

Albumin: 4.1

Result: Probability of Farooq Having Liver Disease is 29.36

6.3. Results

In our implementation, we obtained some positive results.

We split our original dataset of **582** data points (with 6 attributes each) into two sets of data, a training set of size **436** data points and a test set of **146**. We followed a split ratio of 1/4.

The model was trained on the aforementioned 436 data points, and then tested on the ‘unknown’ 146 data points.

When we ran our optimized Logistic Regression model, it returned with a **79.45%** accuracy. It implies that out of the 146 cases that we tested, the model could **correctly** predict whether the person with those characteristics would get a heart disease or not 116 times.

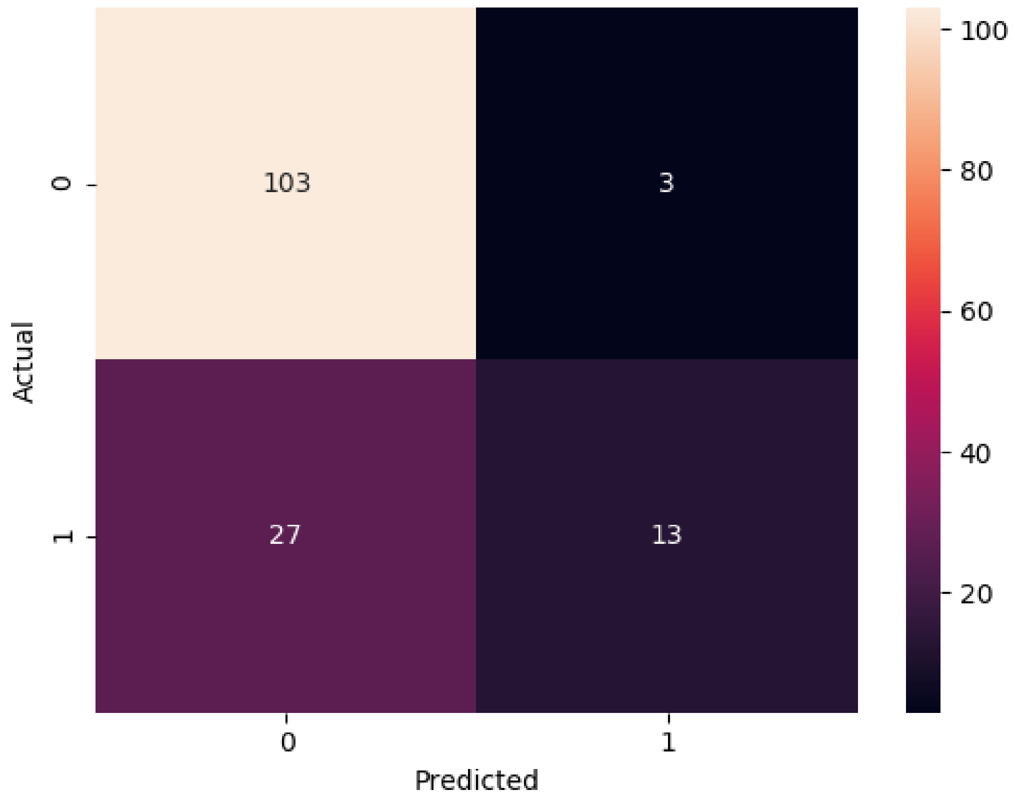
We can see a breakdown of those results in the following Heat-map of the **Confusion Matrix.**

Confusion Matrix is a metric used for prediction models, some relevant terms are as follows:

True Positive*: Actually positive, and predicted to be positive.*
True Negative*: Actually negative, and predicted to be negative*
False Positive*: Actually negative, and predicted to be positive.*
False Negative*: Actually positive, and predicted to be negative.*

Here prediction ‘0’ implies that the person is safe and ‘1’ implies that he is diseased.

Figure 6.3.1: Confusion Matrix



We can also see how our model improved at every stage of the pipeline.

The two metrics depicted here are accuracy and F1 score. Accuracy is the total percentage of correct predictions in the test data. F1 score is calculated by the following formula.

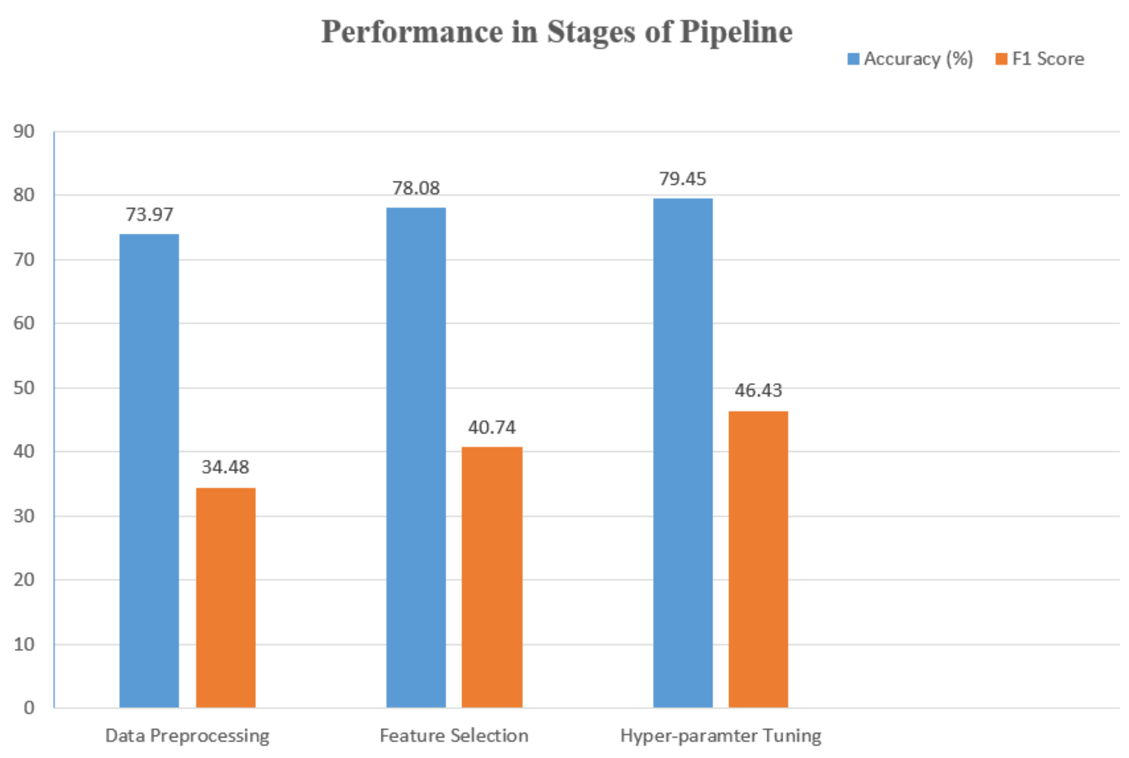
$$PRE = \frac{TP}{TP + FP}$$

$$REC = TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

$$F_1 = 2 \cdot \frac{PRE \cdot REC}{PRE + REC}$$

Where TP, FP and FN are True Positive, False Positive and False Negative respectively.

Figure 6.3.2: Performance in Stages



7. Inferences

7.1. *Discussion*

Our findings through the project solidifies the fact that at the end of the day, the input to a Machine Learning model is just as important as the model itself. We can extract meaningful information from a small dataset to get high accuracies too, if we have a functional methodology in place.

The accuracy that we got from using Logistic Regression, which is considered a light-weight, fast and amateur classification algorithm rivalled the performance of complicated algorithms such Neural Network and Random Forest Classifier implemented in the literature. Our implementation also obtained the highest accuracy recorded from a Logistic Regression Model on this dataset.

One setback found in our implementation was a relatively high number of *False Negatives* in our predictions. A *False Negative* is a result that was actually positive, but predicted to be negative. One can assume why this sort of error would be undesirable in a classifier that is supposed to diagnose diseases.

At last, more than replacing our current health care systems and personnel, our short term goal is to provide them a tool to use in their diagnosis.

7.2. *Future Work*

We intend to further refine our implementation as there is always room for improvement in the field of data science. There are a lot of components that we can add to our implementation such as a *sampler* that would add more samples to our dataset to train our model on, or *Principal Component Analysis* stage for feature selection.

We can also modify the *Fitness Function* of our Genetic Algorithm to make it so that it would reduce the number of False Negatives, which is, as we discussed, undesirable in the field of Automated Diagnosis.

We would also like to apply our pipeline to different datasets to see how we can contribute further to the ongoing research and improve upon the current status quo for a lot of other implementations.

7.3. Conclusion

As we can observe, there is plenty of scope in the field of Bio-Medical Informatics for us, as budding Computer Science students to explore and implement ground-breaking algorithms & machine learning pipelines to extract meaningful inferences that can directly help people.

With the help of Modern Computer Science tools, we can use the abundance of data available in this "Age of Information" to find patterns in raw numbers and help Medical Professionals take potentially life-saving informed decisions.

Not only in disease diagnosis, we can take the help of Machine Learning and Data Analysis in the field of Genetics too. As the DNA information that defines the blueprint of the Human Body, is a seemingly never-ending well of raw data and information that is in dire need of prediction models and analytical tools.

It is our conviction that we would contribute to this ever-growing field and improve our initial understandings by doing further research and improving our existing pipeline.

8. References

8.1. Dataset was taken from the popular UCI Machine Learning Repository
https://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset)

8.2. The Python in-built machine learning module "Scikit-Learn"
https://scikit-learn.org/stable/

8.3. Understanding of the Algorithms and various Data Science costructs from the popular MOOC by Andrew Ng
https://www.coursera.org/learn/machine-learning-with-python

8.3. Genetic Algorithm Implementation
https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-15-S16-S11

8.4. Genetic Algorithm Implementation
https://blog.coast.ai/lets-evolve-a-neural-network-with-a-genetic-algorithm-code-included-8809bece164