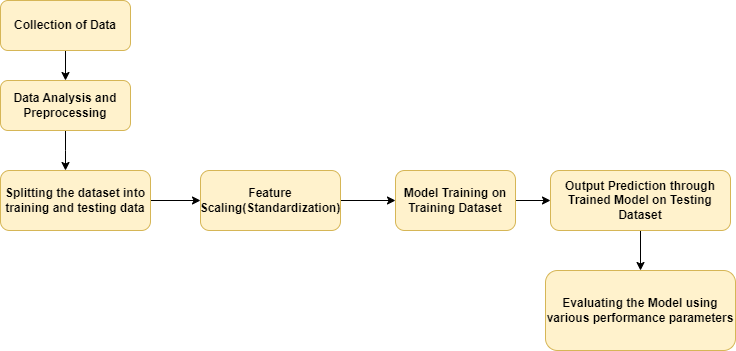
***Parkinson’s Disease Detection***

***using***

***Machine Learning and Data Mining***

**

**Flowchart**



**Flowchart Explanation**

* As a first step we look for the appropriate dataset. Since data collection would be bit difficult due to time constraint and proper biomedical signal processing knowledge would be required for the same so we will use the already available data sets.
* After that we will process and analyze the data. In the data processing and analyzing following steps are followed:
  + Handling and replacing the missing values in our dataset(no missing value was found in our dataset)
  + Handling the Categorical values in the dataset(our dataset had only numerical values)
  + Extraction of dependent and independent variable from the dataset.(all other variables except status variable are independent and status variable is dependent variable)
* Next step is splitting the dataset into trainng and testing data(80% data is given to training data and 20% is given to testing data).
* The data is scaled using Standardization approach so that all the data values belong to a particular range
* The training Data is trained using differentt models
* The model is then tested on testing dataset using the predict method
* The accuracy and precision of different Models is evaluated and the best model is choosen.

**Model Related Explanation**

We have used 6 models in our implementation. The models used with their mathematical approach are as:

Sure, here are some further details behind the mathematics used in each of the models:

* **Logistic Regression**: The logistic regression model uses the logistic function (also called sigmoid function) to map the input variables to the probability of the binary response variable. The logistic function is given by:

p(x) = 1 / (1 + e^(-z)), where z = b0 + b1x1 + b2x2 + ... + bnxn

where p(x) is the probability of the positive response variable, e is the base of natural logarithm, and b0, b1, b2, ..., bn are the coefficients of the model that are estimated using the maximum likelihood method.

* **KNN**: The KNN algorithm classifies a data point based on the majority class of its k-nearest neighbors. The distance between two data points is measured using a distance metric such as Euclidean distance or Manhattan distance. The optimal value of k is usually determined using cross-validation.
* **Naive Bayes**: The Naive Bayes algorithm uses Bayes' theorem to estimate the probability of a class given a set of features. It assumes that the features are conditionally independent of each other given the class. The probability of a class given the features can be calculated using:

P(class | features) = P(class) \* P(features | class) / P(features)

where P(class) is the prior probability of the class, P(features | class) is the likelihood of the features given the class, and P(features) is the evidence.

* **SVM**: The SVM algorithm separates the data points into different classes by finding the hyperplane that maximizes the margin between the two classes. The hyperplane is defined by:

w^T \* x + b = 0

where w is the weight vector, b is the bias term, and x is the input vector. The optimal values of w and b are found by solving a constrained optimization problem using Lagrange multipliers. When the data is not linearly separable, SVM uses kernel functions to transform the data into higher dimensions where it can be separated by a hyperplane.

* **Decision Trees**: The decision tree algorithm partitions the data into smaller subsets based on the values of different features. The partition is done using a set of rules based on the information gain or Gini impurity criteria. The information gain measures the reduction in entropy after splitting the data, while the Gini impurity measures the probability of misclassifying a random sample.
* **XGBoost**: The XGBoost algorithm is an ensemble learning method that combines multiple decision trees to improve the predictive power of the model. It uses a gradient boosting algorithm to minimize the loss function. The loss function is defined as the sum of the loss for each data point and the complexity of the model. The complexity of the model is penalized by a regularization term to prevent overfitting.

**Dataset Description**

This dataset is composed of a range of biomedical voice measurements from 31 people, 23 with Parkinson's disease (PD). Each column in the table is a particular voice measure, and each row corresponds to one of 195 voice recordings from these individuals ("name" column). The main aim of the data is to discriminate healthy people from those with Parkinson’s Disease, according to the "status" column which is set to 0 for healthy and 1 for Parkinson’s Disease.

**Attribute Information:**

Matrix column entries (attributes):

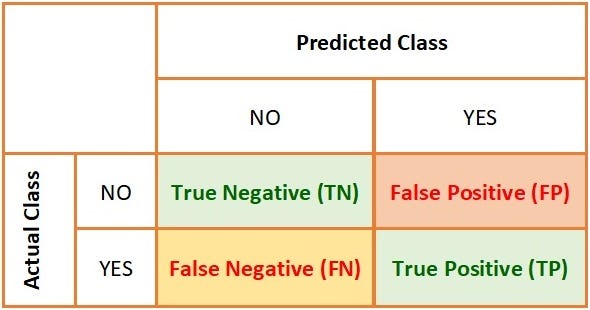
* name - ASCII subject name and recording number
* MDVP:Fo(Hz) - Average vocal fundamental frequency
* MDVP:Fhi(Hz) - Maximum vocal fundamental frequency
* MDVP: Flo(Hz) - Minimum vocal fundamental frequency
* MDVP:Jitter(%) , MDVP:Jitter(Abs) , MDVP:RAP , MDVP:PPQ , Jitter:DDP - Several measures of variation in fundamental frequency
* MDVP:Shimmer , MDVP:Shimmer(dB) , Shimmer:APQ3 , Shimmer:APQ5 , MDVP:APQ , Shimmer:DDA - Several measures of variation in amplitude
* NHR , HNR - Two measures of ratio of noise to tonal components in the voice
* status - Health status of the subject (one) - Parkinson's, (zero) - healthy
* RPDE , D2 - Two nonlinear dynamical complexity measures
* DFA - Signal fractal scaling exponent
* spread1 , spread2 , PPE - Three nonlinear measures of fundamental frequency variation

**Experimental Result Submission**

Peformance Metrics

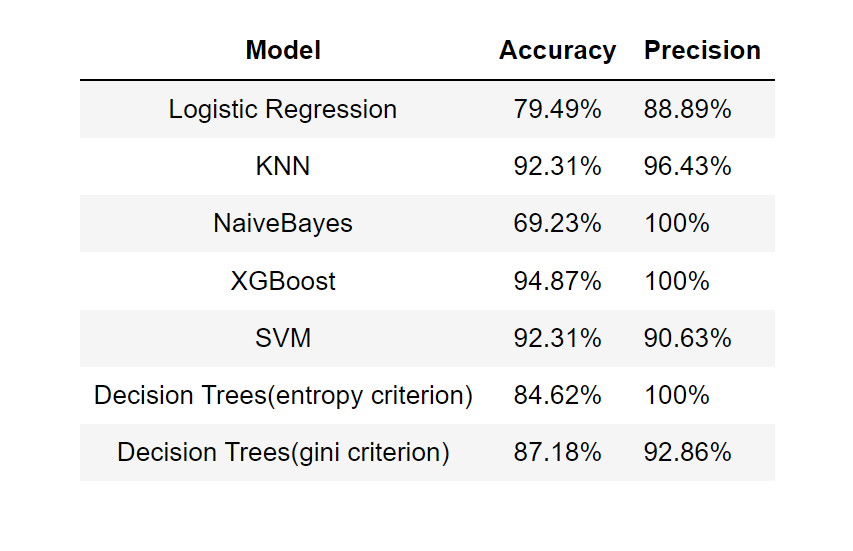
The model is tested using varrious performance parametres such as :

* Accuracy: Accuracy is the proportion of correct predictions among all predictions made by the model. It is calculated as (TP + TN) / (TP + TN + FP + FN), where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.
* Precision: Precision is the proportion of true positives among all positive predictions made by the model. It is calculated as TP / (TP + FP).
* Recall (also called Sensitivity or True Positive Rate): Recall is the proportion of true positives among all actual positive cases in the dataset. It is calculated as TP / (TP + FN).
* F1 Score: F1 Score is the harmonic mean of precision and recall. It is calculated as 2 \* precision \* recall / (precision + recall).
* Specificity (also called True Negative Rate): Specificity is the proportion of true negatives among all actual negative cases in the dataset. It is calculated as TN / (TN + FP).



Performance of the Models used in the Implementation

We have mainly focused on the Accuracy and Performance of various models used in our project.Here is the summary:



Based on this we can see that **XGBoost** is best implemented model.

* 1. Comparative analysis

|  |  |  |
| --- | --- | --- |
| **Reference** | **Precision (%)** | **Accuracy (%)** |
| [Dr. Anupam Bhatia[1]Ms. Raunak Sulekh[2]](http://csjournals.com/IJCSC/PDF9-1/32.%20Raunak.pdf) | 99.75 | 98.50 |
| [C K Gomathy](https://www.researchgate.net/publication/357448942_THE_PARKINSON'S_DISEASE_DETECTION_USING_MACHINE_LEARNING_TECHNIQUES#:~:text=There%20is%20a%20model%20for,affected%20person%20by%20Parkinson's%20disease.) | 94.87 | 100 |
| Propose Model: XG Boost | 94.87 | 100 |