PARKINSON'S DISEASE DETECTION USING MACHINE LEARNING ALGORITHMS

Project Report submitted by
Ritika Salimath
220953428

ABSTRACT

Parkinson’s disease is a progressive brain disorder that causes movement problems, mental health issues and other health problems. It affects the nervous system and parts of the body that are controlled by nerves. Symptoms include tremors, stiffness, slowness of movement, difficulty speaking, poor balance and coordination and painful muscle contractions.
Detecting the disease early can significantly improve treatment outcomes and quality of life. Speech impairment is one of the earliest symptoms and is a very common symptom that affects more than 89% of the patients.
Speech analysis is a promising area due to the vocal impairments commonly observed in patients suffering from parkinson’s disease. Machine learning algorithms are applied to analyze speech features, including jitter, shimmer, and harmonic-to-noise ratios, to differentiate between healthy individuals and those with Parkinson’s Disease..

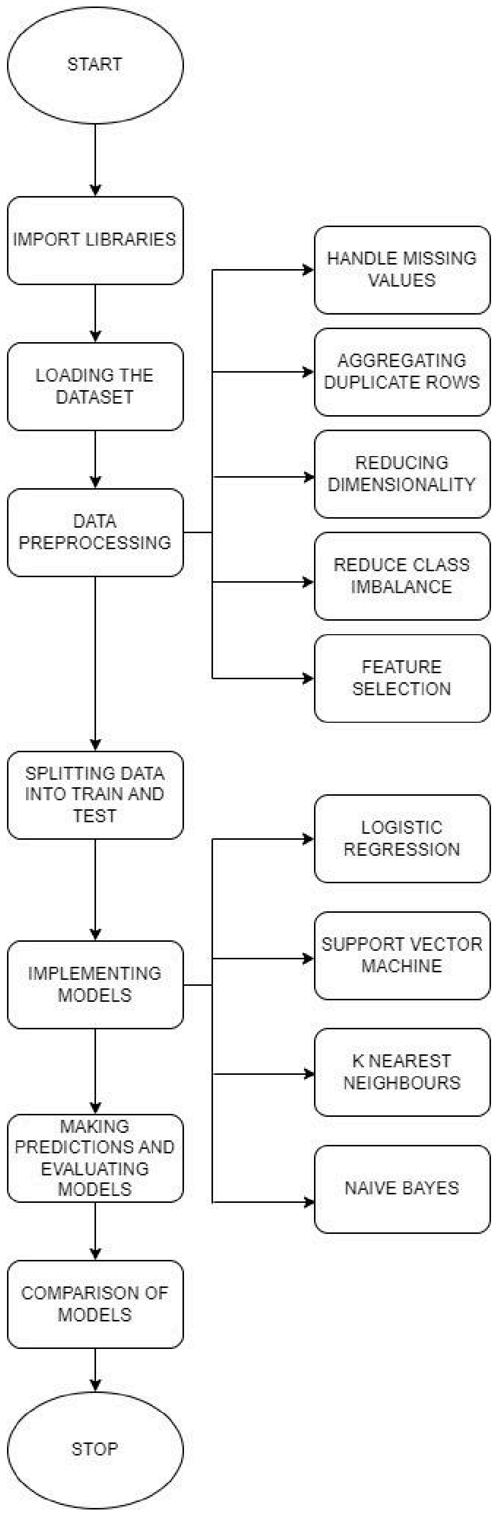
TABLE OF CONTENTS

1.INTRODUCTION

This project focuses on evaluating the performance of various machine learning models for the classification of parkinson’s disease on the basis of speech features. The data used in the project is taken from UCI Machine learning Repository- “Parkinson’s disease classification”. The data has been gathered from 188 patients with parkinson’s disease with ages ranging from 33 to 87 years. Various speech signal processing algorithms including Time Frequency Features, Mel Frequency Cepstral Coefficients (MFCCs), Wavelet Transform based Features, Vocal Fold Features and TWQT features have been applied to the speech recordings of the patients to extract clinically useful information for PD assessment.
Several Machine Learning algorithms such as Logistic regression, Support vector classifiers, K Nearest Neighbour Classifiers and the Naive Bayes method have been used on the same dataset and are compared on the basis of their ROC AUC accuracy scores and their F1 Scores. Data preprocessing has been done by handling missing values, feature scaling and the removal of highly correlated features to reduce dimensionality. Techniques such as random over sampling have been used to reduce class imbalance. Statistical methods such as Chi-Square tests are applied to the feature selection, determining which attributes are the most appropriate for classification.
The models have been trained using a portion of the dataset and their performance has been evaluated on metrics such as accuracy, precision, recall and F1 score. The results indicate that Logistic Regression demonstrates the highest performance among the four algorithms.

2.Methodology

2.1 Flowchart of the proposed method



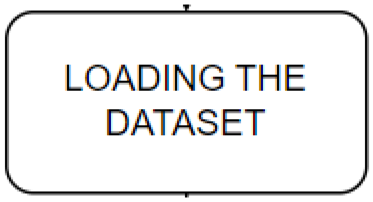
2.2 Explanation of each phase of the block diagram

a)

- Pandas as pd: Used for data manipulation and analysis
- seaborn as sb: Used to create statistical graphics for data visualization
- matplotlib.pyplot as plt: Used for data visualization typically in the form of plots, graphs and charts
- from imblearn.over_sampling import RandomOverSampler: Used to handle imbalanced datasets
- from sklearn.preprocessing import MinMaxScaler: Scales features to a specified range, typically [0, 1], by transforming the data proportionally to its minimum and maximum values, ensuring consistent scaling across all features.
- from sklearn.feature_selection import SelectKBest: Selects the top k features based on a specified statistical test, helping to reduce dimensionality by keeping only the most relevant features for the target variable.
- from sklearn.feature_selection import chi2: A statistical test used with SelectKBest to score features based on their chi-squared statistic, measuring the dependency between categorical features and the target variable.
- from sklearn.metrics import ... : Imports specific metrics functions from scikit-learn for evaluating machine learning models.
  -roc_auc_score as ras: Computes the Area Under the Receiver Operating Characteristic Curve (ROC AUC) to evaluate a model's ability to distinguish between classes, with values closer to 1 indicating better performance.
  - accuracy_score: Computes the accuracy of a classification model.
  -confusion_matrix: Generates a confusion matrix to evaluate classification performance.
  - precision_score: Calculates the precision of a classification model.
  - f1_score: Computes the F1 score, a harmonic mean of precision and recall.
  - recall_score: Calculates the recall of a classification model.
- from sklearn.model_selection import train_test_split: Imports the train_test_split function from scikit-learn to split data into training and testing sets.
- from sklearn.svm import SVC: Used to implement Support Vector Classifier
- from sklearn.linear_model import LogisticRegression: Used to implement logistic regression
- from sklearn.neighbors import KNeighborsClassifier: Used to implement K Nearest Neighbours algorithm

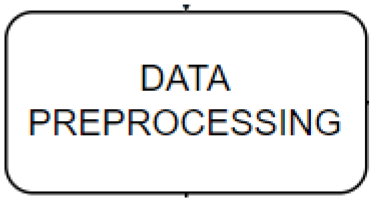
- from sklearn.naive_bayes import GaussianNB: Used to implement Gaussian Naive Bayes for classification, assuming features are normally distributed and conditionally independent given the class label.

b)



The Parkinson's disease classification dataset was downloaded from the UCI Machine Learning repository.[https://archive.ics.uci.edu/dataset/470/parkinson+s+disease+classification]
This dataset is in csv file format. The function “read_csv” part of the pandas library was used to load the file into a dataframe for further analysis.

c)



Data Preprocessing ensures that the data is standardized, devoid of noise and is suitable for model training. Following are the various techniques that have been implemented in this project.

i)

Missing data can cause results to be skewed and cause errors. The dataframe was checked to see if it contains any missing values or null values using the following code:

```
missing_values = df.isnull().sum()
print("Missing values:\n", missing_values)
```

It was seen that there are no missing or null values and hence imputation of values was not necessary.

ii)

AGGREGATING
DUPLICATE ROWS

The data contains duplicate entries for the same ID. We need to get rid of the duplicity by aggregating the rows with the same ID. This is done by grouping the rows by ID and calculating the average of the features to condense data and remove duplicates. The followig code is used to achieve this:

```
df=df.groupby('id').mean().reset_index()
```

The ID column is then dropped as it is of no use for modelling and this would reduce the feature space.

```
df.drop("id",axis=1,inplace=True)
```

iii)

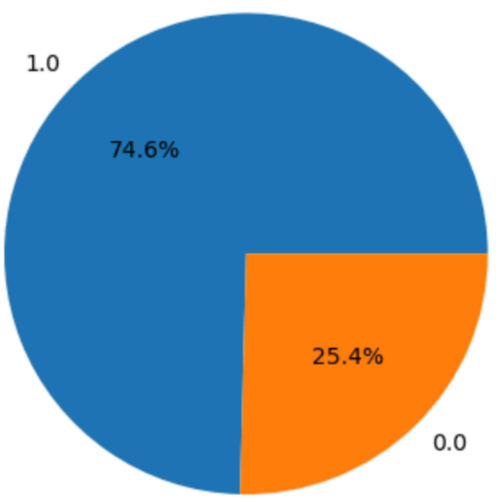
REDUCING
DIMENSIONALITY

Reducing dimensionality is done to eliminate redundant features to improve computational efficiency as well as reduce overfitting. Removing highly correlated columns helps in reducing dimensionality. In this project, columns having correlation more than 0.7 are filtered out.

iv)

REDUCE CLASS
IMBALANCE

An imbalanced dataset is a problem where the distribution of the examples across the classes is not equal, i.e, it is biased or skewed. Techniques to handle imbalanced datasets include oversampling and undersampling. Oversampling involves increasing the number of samples in the minority class. Undersampling is removing some observations of the majority class.
It is observed that the majority class in this project is “1.0”.



The imbalanced dataset is handled by adding repetitive rows of the minority class. The following code handles the imbalance:

```
ros = RandomOverSampler(sampling_strategy='minority',random_state=0)
X, Y = ros.fit_resample(X_train, Y_train)
```

v)

Feature selection is the process of identifying and selecting the most relevant features from the dataset. This is done to improve the model's performance and efficiency.
The features are scaled to a uniform range of [0,1] which ensures compatibility with statistical methods like the chi square test.

```
X_norm = MinMaxScaler().fit_transform(X)
```

The SelectKBest function with chi2 is then used to evaluate and select the 30 most relevant features based on their statistical dependency with the target variable.

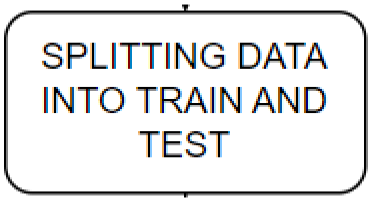
```
selector = SelectKBest(chi2, k=30)
selector.fit(X_norm, df['class'])
```

The "get_support()" functions returns a boolean array indicating the features which were selected

```
filtered_columns = selector.get_support()
```

The features which are not selected are then removed and the selected features are retained in the dataframe.

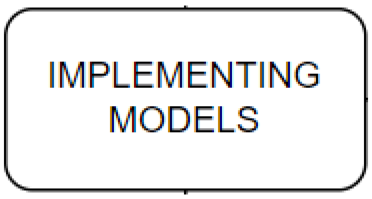
d)



Here we use the `train_test_split` function from the `sklearn.model_selection` module in Python. Here is a breakdown of what each part of the code does:
- X_train: This variable will store the training data for the features after splitting.
- X_val: This variable will store the testing data for the features after splitting.
- Y_train: This variable will store the training data for the target variable after splitting.
- Y_val: This variable will store the testing data for the target variable after splitting.
- train_test_split: This function is used to split the dataset into training and testing sets.
- features: This variable contains the features of the dataset.
- target: This variable contains the target variable of the dataset.
- test_size: This parameter specifies the proportion of the dataset that should be included in the testing set.
- random_state: This parameter sets the random seed for reproducibility.

We have split the data in such a way that 20% of it is used for testing and 80% of it is used for training.

e)

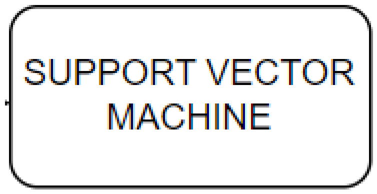


The models implemented in this project are Logistic Regression, Support Vector Machine, K Nearest Neighbours and Naive Bayes.

i)

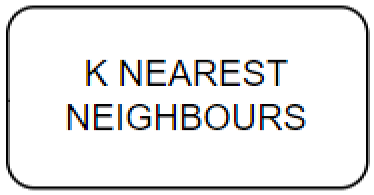
Logistic regression is a supervised machine learning algorithm used for classification tasks. It predicts the output of a categorical dependent variable and hence the outcome can be either 0 or 1. It gives the probabilistic values which lie between 0 and 1.
Logistic regression uses a sigmoid function to map the predicted values to probabilities.

ii)



Support vector machine is a supervised machine learning algorithm which can be used for both regression and classification tasks. They focus on finding the maximum separating hyperplane between the target classes. The algorithm ensures that the margin between the closest points of different classes, known as support vectors, is maximized.
The kernel is a mathematical function used to map input data into higher dimensional feature space. This allows the SVM to find a hyperplane in cases where data points are not linearly separable in the original space. The kernel used in this project is the radial basis function (RBF).

iii)



K nearest neighbours is a supervised machine learning algorithm which can be used for both classification and regression problems.The K-NN algorithm works by finding the K nearest neighbors to a given data point based on a distance metric, such as Euclidean distance. The class or value of the data point is then determined by the majority vote or average of the K neighbors. K-NN is less sensitive to outliers compared to other algorithms.
This project uses K-NN where the number of neighbours is set as 5.

iv)

NAIVE BAYES

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. It is used for classification problems. This model predicts the probability of an instance belonging to a class with a given set of feature values. It is a probabilistic classifier. It uses Bayes theorem in the algorithm for training and prediction.
Gaussian naive bayes algorithm is implemented in this project. Gaussian Naive Bayes is a type of classification algorithm working on continuous normally distributed features that is based on the Naive Bayes algorithm.

f)

MAKING PREDICTIONS AND EVALUATING MODELS

After splitting the data into training and validation sets, multiple models were trained on the training data. Predictions were then made on both the training and validation datasets. Metrics such as ROC AUC, Accuracy, Precision,recall, confusion matrix and F1 score were used to evaluate the performance of all the models. They help determine how well the models differentiate between patients suffering from parkinson's disease and healthy individuals.

g)

COMPARISON OF MODELS

To compare the performance of different machine learning models used in the project, a bar graph was plotted for each metric.
The bar graph illustrates the respective metric values for each of the four models: Logistic Regression, Support Vector Classifier (SVC), K-Nearest Neighbors (KNN), and Naive Bayes. Each bar represents the metric value of a model, and the bars are color-coded for clarity. The

y-axis is scaled between 0 and 1, representing the metric value as a percentage, while the x-axis lists the model names.
This visualization allows for an easy comparison of the models, showing which algorithm performs best in correctly identifying positive cases of Parkinson's disease. The results of this comparison serve as a basis for selecting the most effective model for the project.
The metrics considered are F1 Score, Accuracy, Precision and Recall(Sensitivity),

F1 Score: It is an evaluation metric which combines precision and recall into a single value. An F1 score can range from 0 to 1. The closer the F1 score of a model is to 1, the better the model is.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Accuracy: It is an evaluation metric that measures how often a model correctly predicts the outcome. The range of the accuracy is between 0 and 1. The closer the accuracy of a model is to 1, the better the model is. Accuracy is calculated by dividing the number of correct predictions to the total number of predictions.

$$\textbf{Accuracy} = \frac{\textbf{TP} + \textbf{TN}}{\textbf{TP} + \textbf{TN} + \textbf{FN} + \textbf{FP}}$$

Precision: It is an evaluation metric used to measure how well a model predicts positive instances. It is the ratio of true positives to the total number of positive predictions. The range of the precision is between 0 and 1. The closer the precision of a model is to 1, the better the model is.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

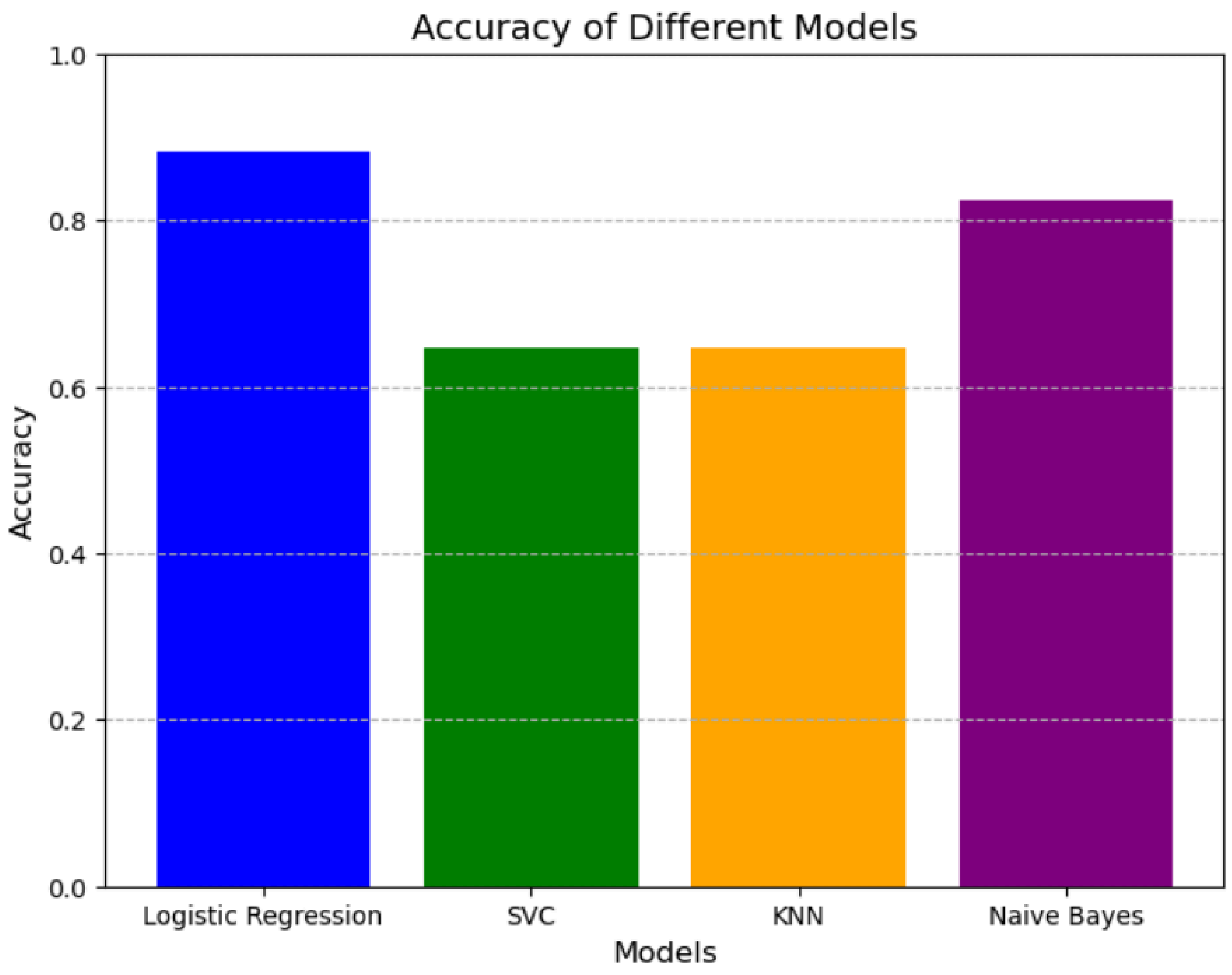
Recall: It is an evaluation metric that measures the proportion of actual positives that are correctly classified as positives. The range of the recall is between 0 and 1. The closer the recall of a model is to 1, the better the model is.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

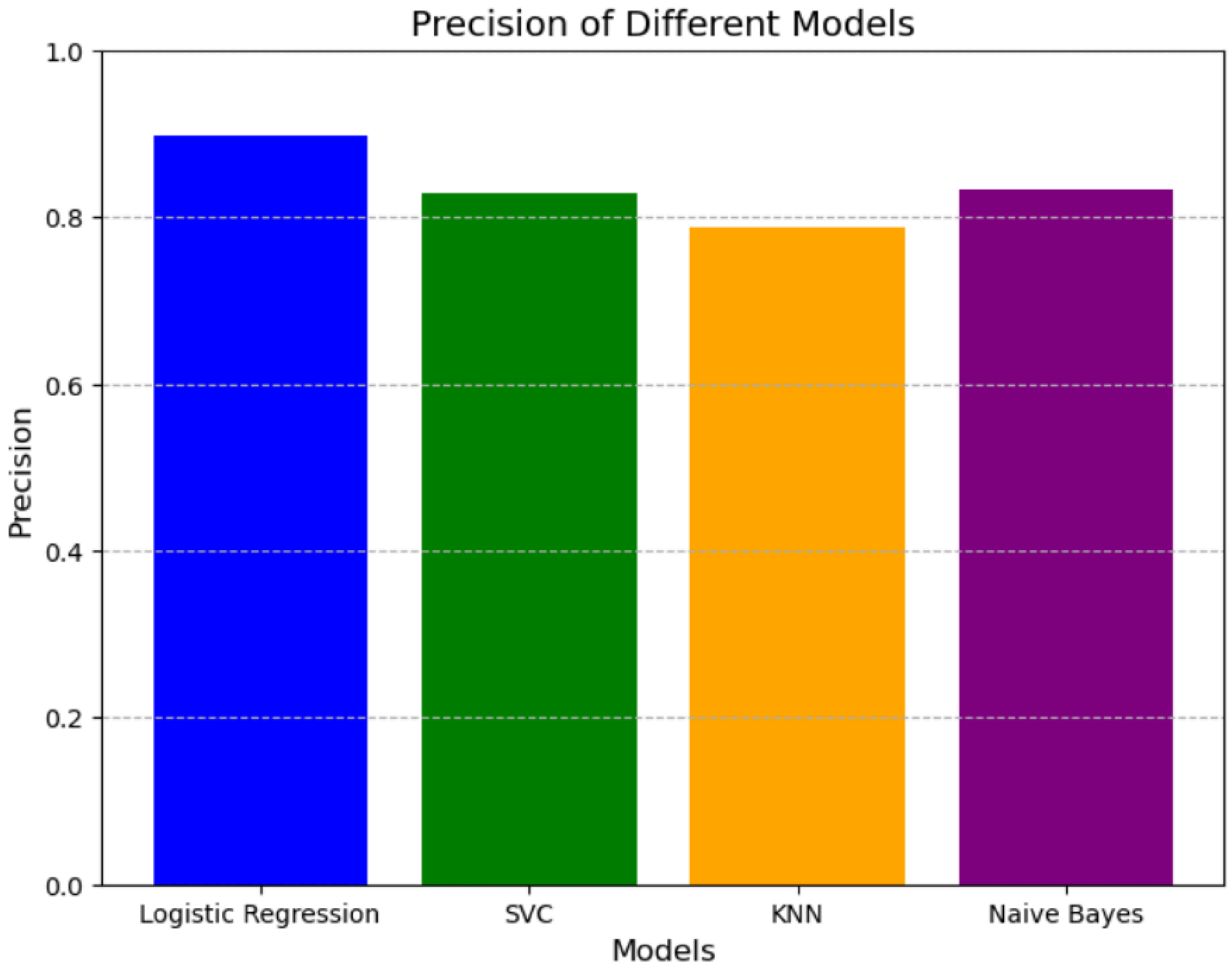
5.RESULTS AND DISCUSSION

The following bar graphs show us the comparison of the metrics on the various models used.

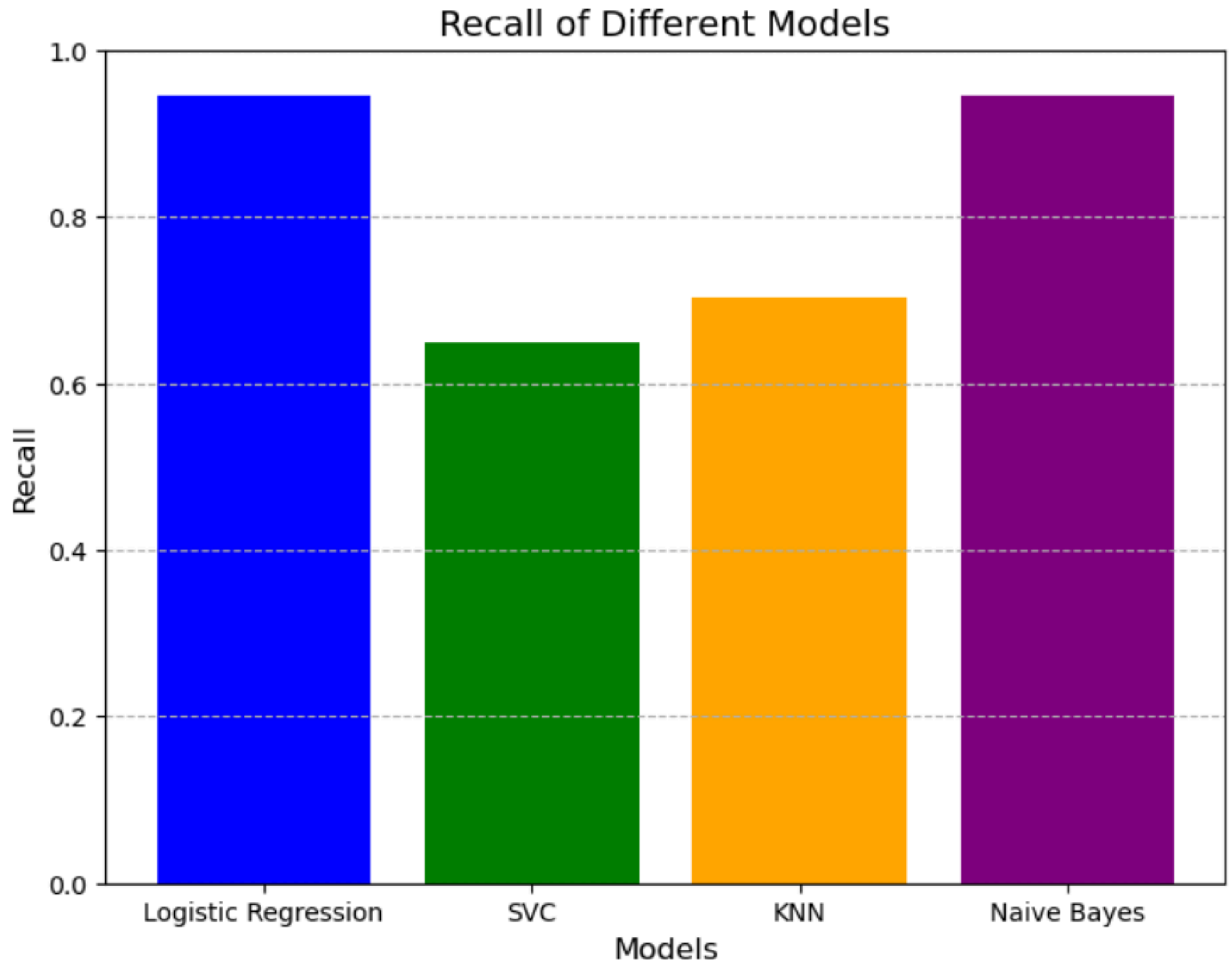
Accuracy:



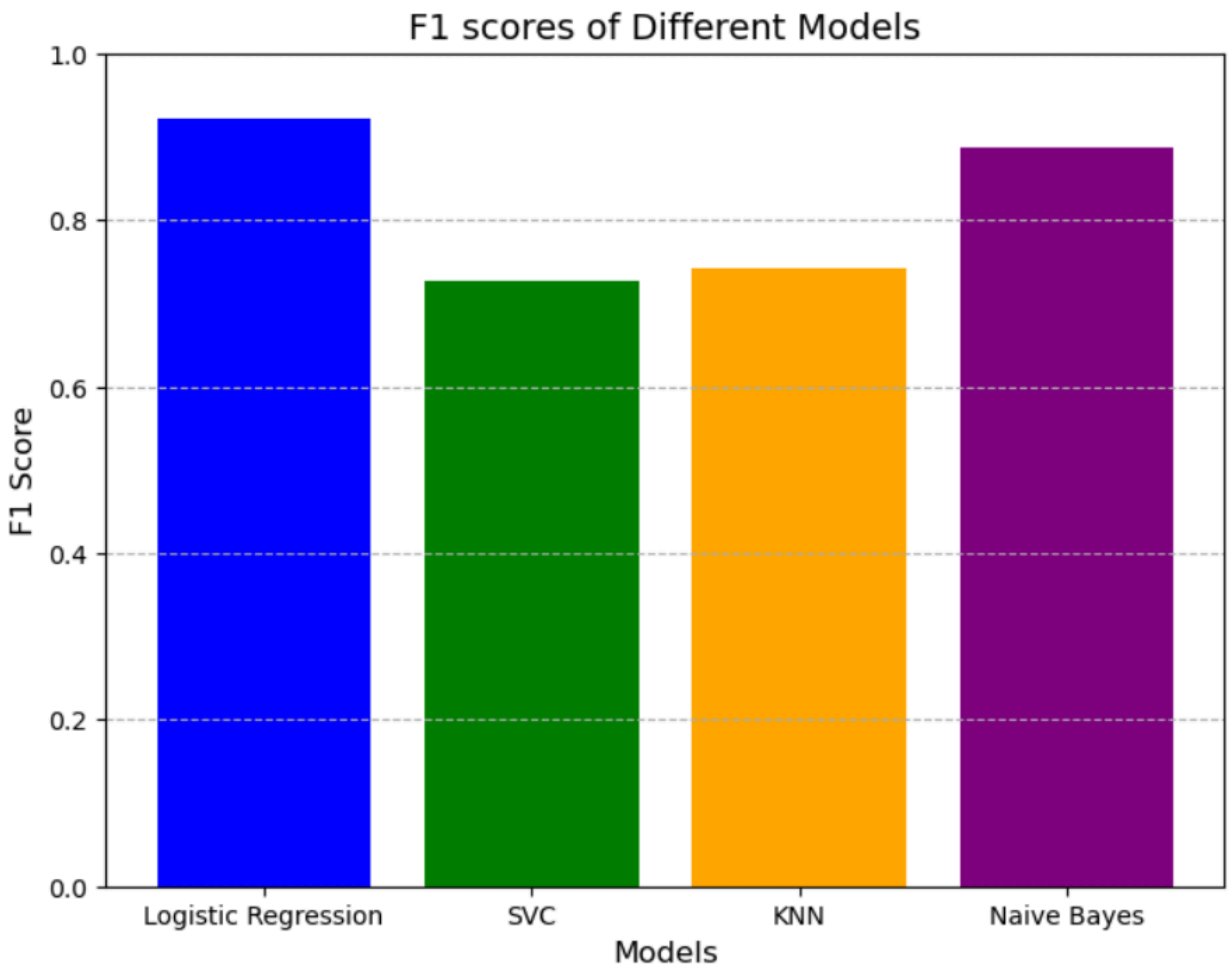
Precision:


Precision of Different Models
Precision
Models
Logistic Regression
SVC
KNN
Naive Bayes
1.0
0.8
0.6
0.4
0.2
0.0

Recall:

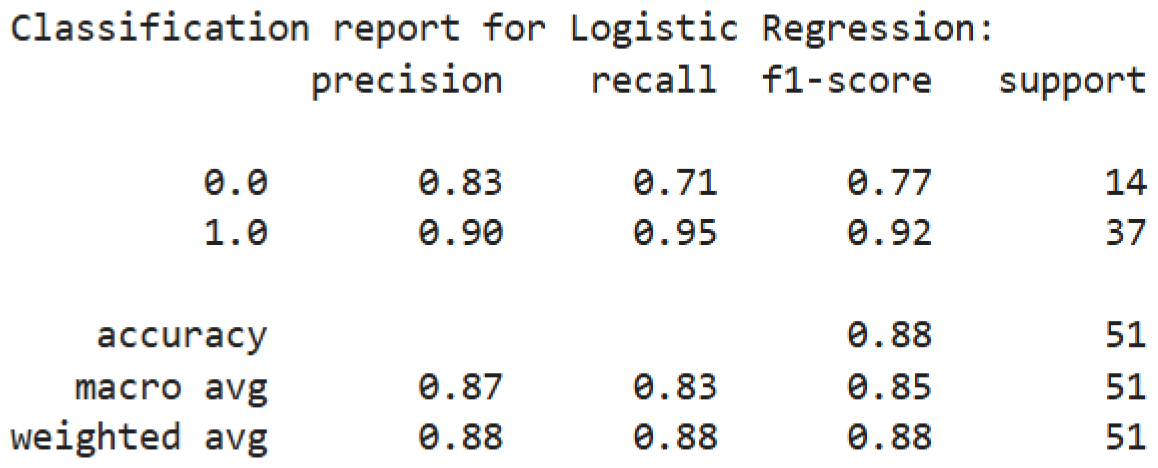

Recall of Different Models
Recall
1.0
0.8
0.6
0.4
0.2
0.0
Logistic Regression
SVC
KNN
Naive Bayes
Models

F1 Score:


F1 scores of Different Models
F1 Score
1.0
0.8
0.6
0.4
0.2
0.0
Logistic Regression
SVC
KNN
Naive Bayes
Models

The classification reports of the various models with the values of all the metrics are as follows:

Logistic regression:

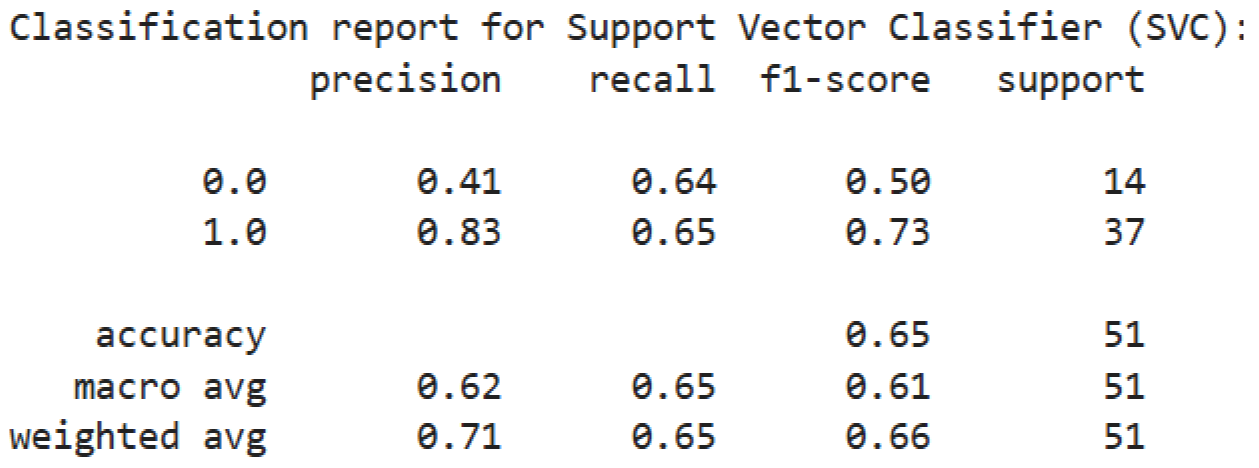


```
Classification report for Logistic Regression:
              precision    recall  f1-score   support

         0.0       0.83      0.71      0.77        14
         1.0       0.90      0.95      0.92        37

    accuracy                           0.88        51
   macro avg       0.87      0.83      0.85        51
weighted avg       0.88      0.88      0.88        51
```

Support vector machine:

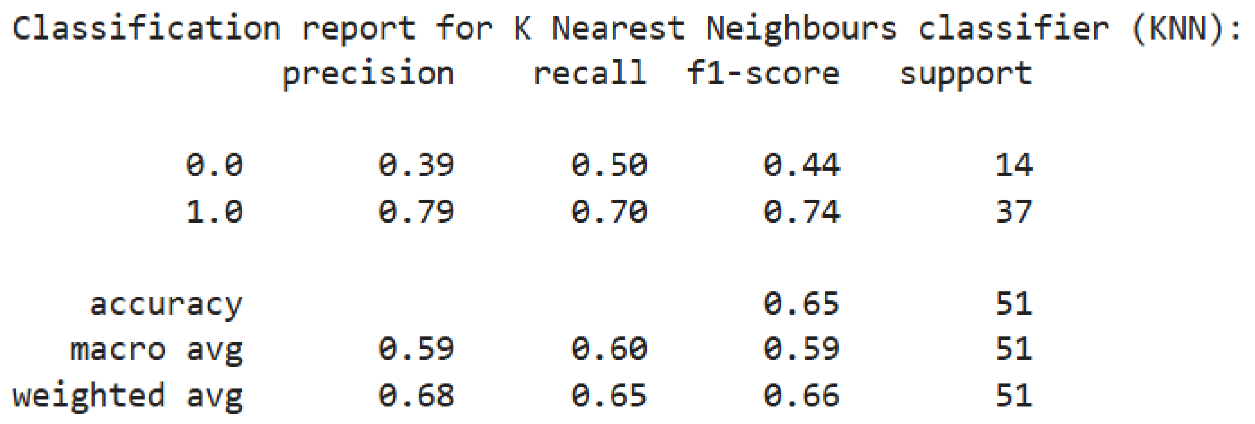


```
Classification report for Support Vector Classifier (SVC):
              precision    recall  f1-score   support

         0.0       0.41      0.64      0.50        14
         1.0       0.83      0.65      0.73        37

    accuracy                           0.65        51
   macro avg       0.62      0.65      0.61        51
weighted avg       0.71      0.65      0.66        51
```

K Nearest Neighbours:

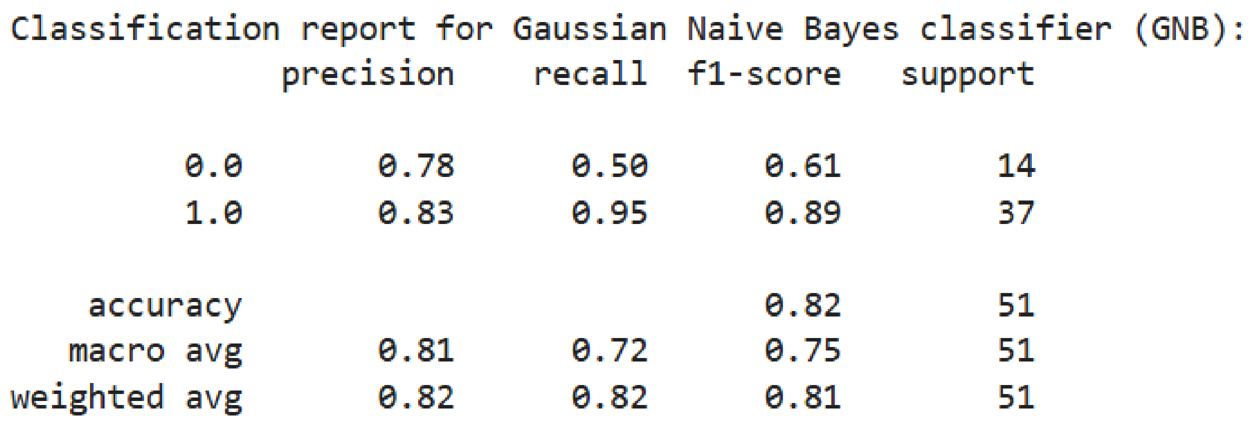


```
Classification report for K Nearest Neighbours classifier (KNN):
              precision    recall  f1-score   support

         0.0       0.39      0.50      0.44        14
         1.0       0.79      0.70      0.74        37

    accuracy                           0.65        51
   macro avg       0.59      0.60      0.59        51
weighted avg       0.68      0.65      0.66        51
```

Naive Bayes:



```
Classification report for Gaussian Naive Bayes classifier (GNB):
              precision    recall  f1-score   support

         0.0       0.78      0.50      0.61        14
         1.0       0.83      0.95      0.89        37

    accuracy                           0.82        51
   macro avg       0.81      0.72      0.75        51
weighted avg       0.82      0.82      0.81        51
```

In this project, for detecting Parkinson's syndrome, a higher sensitivity(recall) is preferred as it indicates lesser false negatives and a higher precision as it indicates lesser false positives.

The following conclusions can be made based on the above graphs and values:

- Logistic regression has the highest accuracy(0.88), precision(0.90) and F1 Score(0.92).
- Logistic regression and naive bayes have the same recall value (0.95)
- Based on the above observations Logistic Regression can be said to be well-suited for early detection of Parkinson's Disease from speech features.
- The second best is the naive bayes model which is followed by K Nearest neighbours and the least is Support vector machine. Although it can be observed that the evaluation metrics of the KNN model and the SVC model are more or less similar.

6. CONCLUSION

This project explores the application of machine learning techniques for the detection of Parkinson's disease using speech-based features. The data was preprocessed and the most relevant features were extracted using feature selection techniques. The performance of various machine learning models such as Logistic Regression, Support Vector Machines, K-Nearest Neighbors , and Naive Bayes were compared. Important evaluation metrics like recall, ROC AUC, accuracy, precision, F1 Score and confusion matrices helped understand how well each model did when classifying those with Parkinson's disease.The results show the capability of machine learning to evaluate subtle patterns in speech features towards early and accurate detection of Parkinson's disease.