Parkinson’s Disease Detection Using Machine Learning

A Research-Based Technical Report

Author: Satyam Gajjar

Abstract
This research paper presents a machine learning-based approach for the early detection of Parkinson's disease using biomedical voice measurements. The study explores data preprocessing techniques, feature scaling, model training using Support Vector Machines (SVM), and evaluation of predictive performance. The resulting model demonstrates high accuracy and serves as a basis for clinical decision support systems.

1. Introduction

Parkinson’s disease is a progressive neurological disorder that affects movement and vocal characteristics. Early diagnosis plays an essential role in improving patient outcomes. Machine learning methods provide a powerful tool to classify biomedical features and assist clinical professionals in disease detection.

2. Dataset Description

The dataset contains 22 biomedical voice features extracted from sustained vowel phonation recordings. The target variable, *status*, indicates whether the subject is healthy (0) or diagnosed with Parkinson’s disease (1).

3. Methodology

The machine learning workflow includes: 1. Data Loading and Inspection 2. Data Cleaning and Preprocessing 3. Train-Test Split 4. Feature Standardization using StandardScaler 5. Model Training using SVM 6. Model Evaluation 7. Prediction on New Samples

4. Data Preprocessing

All non-numeric or irrelevant columns were removed. Features (X) and the target variable (y) were separated. A train-test split of 80/20 ensured unbiased evaluation.

5. Feature Scaling

StandardScaler was applied to normalize data distributions, which improves the performance of SVM models. Scaling parameters were learned from the training set and applied consistently to test and prediction data.

6. Model Training

Support Vector Machine (SVM) with a linear kernel was selected due to its effectiveness in binary classification tasks and ability to handle high-dimensional data. The model was trained using scaled feature inputs.

7. Model Evaluation

Accuracy metrics were computed using the test dataset. The model achieved strong performance, demonstrating its suitability for real-world diagnostic assistance.

8. Prediction Pipeline

To predict for a new patient: 1. Input values are converted to a NumPy array 2. Reshaped to 2D: (1, number_of_features) 3. Standardized using the trained scaler 4. Passed to the SVM model for classification

9. Conclusion

The machine learning model successfully detects Parkinson’s disease using vocal biomarkers. Future work may include deep learning models, larger datasets, and integration with real-time clinical tools.

Appendix: Notebook Code and Explanations

Import dependencies

```
import numpy as np
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import accuracy_score
```

Data Collection and Analysis

```
#loading the csv file and printing the head
parkinsons_data = pd.read_csv('/content/drive/MyDrive/Data
Science/Projects/14 Parkinsons Disease Detection/parkinsons.data')
parkinsons_data.head()
```

```
#number of rows and cols
parkinsons_data.shape
```

```
#getting some information for the dataset
parkinsons_data.info()
```

```
#checking for missing values
parkinsons_data.isnull().sum()
```

```
#getting some statistical measures about the data
parkinsons_data.describe()
```

```
#checking the count of status
parkinsons_data['status'].value_counts()
```

0 - healthy people, 1 - they have parkinsons disease

Grouping the data based on target variable

```
#Drop the categorical column
parkinsons_data_new = parkinsons_data.drop(['name'], axis=1)
```

```
parkinsons_data_new.groupby('status').mean()
```

Data Preprocessing, seperating the features and targets

```
x = parkinsons_data_new.drop(['status'], axis=1)
y = parkinsons_data_new['status']
```

Splitting the data to training and test data

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2,
random_state=2)
x.shape, x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

Data Standardization

```
scaler = StandardScaler()
scaler.fit(x_train)
```

```
x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)
```

```
x_train
```

Model Training, Support Vector Machine Model

```
model = svm.SVC(kernel='linear')
model.fit(x_train, y_train)
```

Model Evaluation

```
#Accuracy score on training data
x_train_prediction = model.predict(x_train)
x_train_accuracy = accuracy_score(y_train, x_train_prediction)
x_train_accuracy*100
```

```
#Accuracy score on test data
x_test_prediction =model.predict(x_test)
x_test_accuracy = accuracy_score(y_test, x_test_prediction)
x_test_accuracy*100
```

Building a predictive system

```
input_data = x_test[0].reshape(1,-1) #iloc will not work here just bcz we
did standard scalar which changed our dataframe to array
```

```
predictions = model.predict(input_data)
predictions
```

```
if predictions == 0:
print("The person is free from disease")
else :
print("The person has disease")
```

```
y_test
```

Author: Satyam Gajjar