Global Scientific JOURNALS

GSJ: Volume 11, Issue 6, June 2023, Online: ISSN 2320-9186

www.globalscientificjournal.com

Early Parkinson's Disease Detection Using Machine Learning Algorithm.

Ghalib Mundhir Al Mandhary
Dept. of computing
University of the West of England
ghalib2.almandhary@live.uwe.ac.uk

Dr.Raza Hassan
Dept. of computing
University of the West of England
raza.h@live.uwe.ac.uk

***Abstract*—This following research paper showcases the possibility of having an early Parkinson's detection by utilizing and applying the study of machine learning algorithms. Parkinson's disease is a neurodegenerative disorder that effects many patients across the world in which early detection is critical for timely interventions and it can lead to a positive outcome for the patients. The following study uncovers the way to build an efficient model by the help of applying the collected data which contains the patients details like speech related feature which will help us analyze whether the patient is healthy or has the disease. Moreover, we will be exploring different techniques of supervised learning and the most effective and accurate one will be selected. The data within the dataset has 197 number of instances and 23 attributes in which it was a study that was done by the University of Oxford in cooperation with the national center for voice and speech. Moreover, the study was conducted on 31 patients in which 23 were diagnosed with PD (Parkinson's disease).**

Keywords—Parkinson's Disease;Machine Learning;supervised learning; Early Detection;Random Forest;Predictive Modeling; Multivariate selection;Algorithm Evaluation;Model Accuracy.

I. INTRODUCTION

Parkinson's disease or in short PD is a type of progressive neurological disorder which is most common after Alzheimer disease it manly targets the motion of the patient and speech ability. According to Dorsey, "Parkinson's Disease affects 1 % of the world's population over the age of 60" (Dorsey et al., 2007). To this day scientists have not yet fully understood the precise cause of the Parkinson's disease, but a theory suggests that it involves the combination of genetics and the environment of the person. Moreover, a permanent cure does not exist in today's world in which the availability of the treatment focuses on controlling the symptoms to help achieve a better lifestyle for the patient. These are the following symptoms of Parkinson's disease:

- Tremor/frequent shaking sensation in the hands and fingers.
- Stiffness in the muscles.
- Slowness of movement.
- Speech and swallowing difficulties.
- Fatigue and lack of energy.

Simuni and Sethi stated that, "Early diagnosis of PD is crucial, as it enables early intervention, which can significantly slow down the progression of the disease and improve patients' quality of life (Simuni & Sethi, 2008)."
Within the past few years, the medical field has relied a lot on machine learning techniques to help uncover better diagnosis and detecting diseases. Similar approach is used to early detect Parkinson's disease by analyzing the provided data accurately which helps the model perform the best data driven prediction which will result in doctors to make informative decisions regarding their patients care.

II. LITREATURE RIVIEW

- A paper that was published by Sakar and Areora in which they explore the potential of applying machine learning algorithms in the field of medicine where they stated that "Recent studies have demonstrated the potential of machine learning algorithms in early detection of Parkinson's disease." (Sakar et al., 2019; Arora et al., 2014). Moreover, the paper focuses on the two models; Random Forest and SVM (support vector machines) in which it was used to categorize patients by the help of features that were obtained from speech recognition and signals.
- The following research paper uncovers the possibility of utilizing the concept of "deep learning" in the field of early Parkinson's detection by using gait analysis. As per the author Eskofier, "employed convolutional neural networks (CNNs) to analyze gait patterns of Parkinson's patients and healthy subjects, achieving significant improvements in classification performance compared to traditional machine learning approaches" (Eskofier et al., 2016). This summarizes the potential of deep learning when it comes to diagnosing diseases like Parkinson's is quite effective.
- This paper highlights the growth today of machine learning in the field of medicine research and health care, additionally their effect in disease diagnosis and proper treatment. The research author emphasizes

that “machine learning promises to transform the ways we detect, manage, and treat human disease." (Deo, 2015, p. 1284). We can understand from this paper that machine learning plays a major role in decision-making which results in better patient care and positive health outcomes.

III. DATASET SELECTION

The selected dataset that is applied for this research paper was from UCI Machine Learning Repository, a website that contains many machine learning repositories across different topics. The dataset for our research is a study that was conducted on 31 patients by using biomedical speech measurement testing for the purpose of distinguishing between healthy patients and patients with the PD (Parkinson’s Disease). The detailed summary of the dataset is provided below:

Table1: Dataset Description:

Number:	Dataset Name	Source	Data Type	Attributes	Instances
1	Parkinsons Disease Data Set	UCI Machine Learning Repository	Multivariate	23	197

Table2: List of Attributes in the dataset:

Attribute Name	Description
Name	Patient 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

As we can observe from the attributes within the dataset, there are some multiple features like speech frequency measurements variables (MDVP, Shimmer etc..) that have dependency amongst each other, and these features are used to predict the status of the patient whether they are diagnosed by the disease or not. This proves that this dataset is most suited for multivariate analysis for the predictive model that will be created.

IV. PROPOSED IMPLEMENTATION PLAN

The proposed implementation plan to be followed as seen in the figure below, which involves generating the dataset, processing the data stage (data cleansing and removing null values), loading the processed data, apply various machine learning algorithms by training the model and finally based on the output result the model will be evaluated. Moreover, this entire work will be done in python 3.

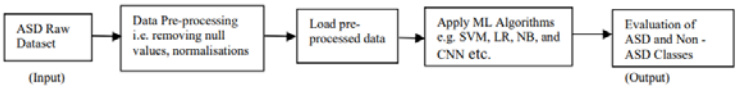

Figure 1 Implementation plan.

4.1 Loading the dataset:

Loading a dataset is a way of reading the data using external python libraries like csv or pandas which helps the data in the form of csv, Dat file or SQL to be displayed and processed. Once the dataset is loaded it can easily be preprocessed, engineered and manipulated. Furthermore, this process is crucial for analyzing the data and having an efficient machine learning development model.

```
#importing the csv file
df = pd.read_csv('Parkinssondata.csv') #Creation of the dataframe
df
```

Figure 2 Loading the dataset in python.

4.2 Data processing:

During this stage after the dataset was loaded and stored in a created data frame, it is now the time to perform data cleansing by removing “Null Values” and unnecessary columns if required. This step is crucial as having such values might cause disturbance for our ML algorithm in training, performing detection and providing a relevant output result. Luckily our selected dataset has no missing values and to confirm that we applied some python code to verify that as seen in the figure below. Moreover, after carefully checking each column within the dataset and its relevancy we concluded that we don’t need to remove any column within the dataset. However, for a more complex data set these steps need to be followed to ensure data accuracy and better performance for our model.

```
#droping Nan values in rows:
df = df.dropna() #function that is used to remove null values,
df
```

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitt
0	phon_R01_S01_1	119.992	157.302	74.997	0.0
1	phon_R01_S01_2	122.400	148.650	113.819	0.0
2	phon_R01_S01_3	116.682	131.111	111.555	0.0
3	phon_R01_S01_4	116.676	137.871	111.366	0.0
4	phon_R01_S01_5	116.014	141.781	110.655	0.0
...	...	...	...	...	
190	phon_R01_S50_2	174.188	230.978	94.261	0.0
191	phon_R01_S50_3	209.516	253.017	89.488	0.0
192	phon_R01_S50_4	174.688	240.005	74.287	0.0
193	phon_R01_S50_5	198.764	396.961	74.904	0.0
194	phon_R01_S50_6	214.289	260.277	77.973	0.0

195 rows × 24 columns

Figure 3 Data cleansing and manipulation.

4.3 Loading pre-processed data:
In this step it simply means that we will need to load our pre-processed dataset again after the cleansing and data manipulation process has been finalized to move to the next stage which is integrating this dataset in a machine learning model to solve a problem.

4.Types of machine learning Algorithm:
Before selecting a specific Algorithm, it is crucial to integrate our dataset and perform the training on several algorithms to compare between each model in terms of performance, accuracy, cost and reliability. Furthermore, For the following reasons we have focused mainly on supervised models because, our data within the dataset are labeled for example the targeted column “Status “indicates 1 the patient has the Parkinson’s disease or 0 the patient is healthy. Moreover, supervised learning supports predictive modeling in which it exceeds in predictive tasks based on the features that is available, they are trained clearly to generalize, adapt and perform prediction on unseen data. According to Russell and Norvig, “Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.” (Russell and Norvig, 2016). These are the following algorithms that will be used for comparison in which we will select the best and most efficient among the rest:

- Random Forest.
- Logistic Regression.
- SVM (Support Vector Machine).
- Decision Tree.
- KNN-KNeighborsClassifier.

This will be done by installing and importing a machine learning library in python that is called “ Sklearn“ as seen in the figure below:

```
from sklearn.linear_model import LogisticRegression #Logistic Regression Lil
from sklearn.svm import SVC #SVC Library
from sklearn.ensemble import RandomForestClassifier #Random Forest Library
from sklearn.tree import DecisionTreeClassifier #Decision Tree Library
from sklearn.neighbors import KNeighborsClassifier #KNN Library
from sklearn.metrics import accuracy_score
```

Figure 4 Machine learning Algorithms Python Libraries.

V. Training and Testing the Model:

Once the data has been prepared and a set of algorithms are selected the model will then be trained in the process that is known as “fitting “in which the parameters inside the model will be optimized to reduce the number of errors between the predicted values and the precise data in the training set. Hastie, Tibshirani, & Friedman have stated in their research paper that, “Training and testing are critical components of the machine learning workflow. The training phase is where the model learns from labeled data, while testing evaluates the model's ability to generalize to new, unseen data. This process is essential to ensure robust performance and avoid overfitting" (Hastie, Tibshirani, & Friedman, 2009). The training will be done by dividing the whole size of the dataset into two sections or parts which will form the 80:20 rule. The rule states that 80% size will be used for training purposes in which the model will be fed data and 20% size will be used to test the performance. This method ensures that the model will constantly learn and adapt to the patterns from the trained data and then apply what it learned in the form of prediction for example in our research, the model will predict the status of the patient.
After the training is done, the model will be tested based on the accuracy, precision, f1 score and recall level. If the result is not satisfactory then a tuning mechanism will be applied to fix the performance.

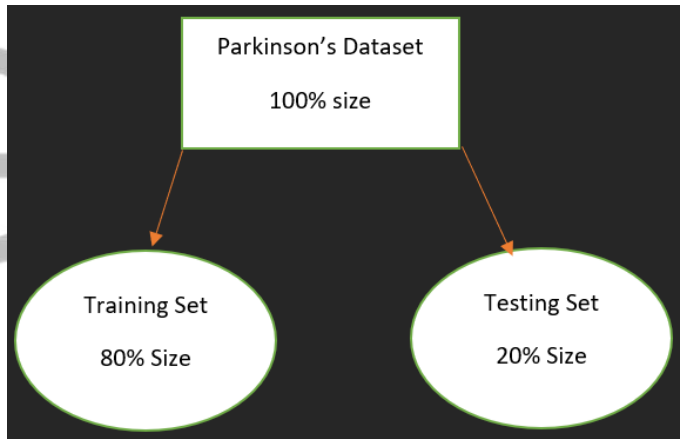

Figure 5 Training and testing dataset Representation

```
# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = SVC()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)
```

Figure 6 Splitting the dataset and training a model in python.

VI. Model Result and Evaluation:

Once the model is trained and tested the result will be evaluated in terms of the overall accuracy which will be displayed by using confusion matrix and a classification report. Furthermore, this report will contain data like the model accuracy, precision, F1 score and recall output. This data is very important to be represented as it will help us distinguish which model is more efficient than the others, moreover, it will also open a room for improvement for the model if it's required.

"Evaluating model performance is a critical step in the machine learning pipeline, as it helps us to assess how well the model generalizes to new, unseen data. Common evaluation metrics include accuracy, precision, recall, F1-score, and others, depending on the problem being addressed. 'An accurate and robust evaluation is crucial for determining the model's efficacy and guiding future improvements,"
(Kelleher, Mac Namee, & D'Arcy, 2015).

Table 3: Confusion Matrix Structure

	Predictive values	
	TP (True Positive)	FP (False Positive)
Actual Values	FN (False Negative)	TN (True Negative)

6.1 Model Accuracy calculation Formula:

This is how the accuracy of the model will be calculated based on the following formula:

$$\text{Accuracy} = \frac{TP + TN}{(TN + TP + FP + FN)}$$

Figure 7 Model Accuracy Formula.

Accuracy measures the ratio of right classifications both in negative and positive values out of the total number of instances. TP (True Positives) represents the value of instances that are correctly classified as positive, TN (True Negatives) means the value of instances that are correctly categorized as negative, FP (False Positives) are the value of instances that are incorrectly represented as positive. Finally, FN (False Negatives) are the total number of instances that are incorrectly classified as negative.

6.2 Model Results:

```python
# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = LogisticRegression(max_iter=1000)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 89.74%
```

Figure 7 Logistic Regression Python Code.

```python
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = SVC()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 84.62%
```

Figure 8 SVC Python Code.

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = RandomForestClassifier()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 94.87%
```

Figure 9 Random Forest Python Code.

```python
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = DecisionTreeClassifier()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 92.31%
```

Figure 10 Decision Tree Python Code.

```python
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Load the dataset
df = pd.read_csv('Parkinssondata.csv')

# Select the features and target variable
X = df.drop(['name', 'status'], axis=1)
y = df['status']

# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create the model
model = KNeighborsClassifier(n_neighbors=5)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
Accuracy: 82.05%
```

Figure 11 KNN Python Code.

Table 4: Total Accuracy Result for Parkinson's Detection Data:

Classifier	Accuracy Result %
Random Forest	94.87%
Decision Tree	92.31%
Logistic Regression	89.74%
Support Vector Machine (SVM)	84.62%
KNN-KNeighborsClassifier	82.05%

As we can observe from the accuracy result table above for the different classifications models that we have selected, Random Forest is on top with an accuracy level percentage of 94.87% which is a very high percentage which significantly displays that its most efficient model when compared to the rest in the table. However, we can simply just depend on one evaluation matrix to make this decision, we will need to consider other evaluation metrics, such as precision, recall, and F1-score to understand the model's performance and efficiency better.

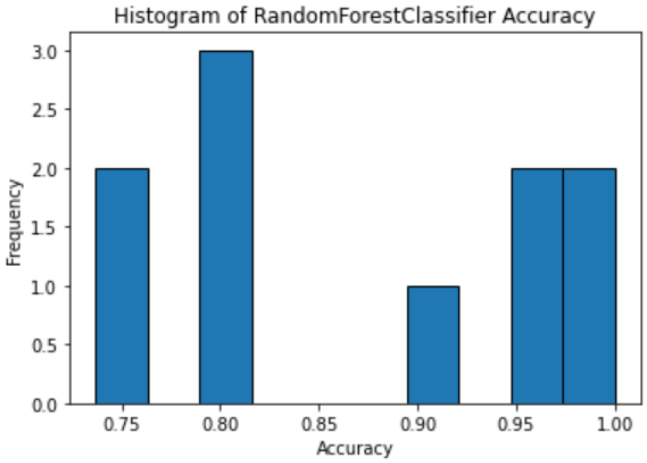

Figure12 Random Forest Accuracy Histogram.

```
Accuracy: 0.9487
Precision: 0.9412
Recall: 1.0000
F1-score: 0.9697

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.71      0.83         7
           1       0.94      1.00      0.97        32

    accuracy                           0.95        39
   macro avg       0.97      0.86      0.90        39
weighted avg       0.95      0.95      0.95        39


Confusion Matrix:
[[ 5  2]
 [ 0 32]]
```

Figure 13 Classification Report and confusion Matrix.

6.3 Model Evaluation:

As observed in the classification report and confusion Matrix in Figure 12, the model runs quite when it comes to predicting the patients who are diagnosed with the Parkinson's Disease with an accuracy of 94.87%. However, the model is evaluated as per the followings:

- Precision: The precision of the model is 0.94 for class 1 which signifies that it identifies 94% of the Parkinson's disease positive cases from the total number of predicted cases as positive. Similarly, for class 0 the precision score was 1 which means it identifies all negative cases (Healthy Patients) from the total number of predicted negative cases.
- Recall: For class 1 the model perfectly recalls with a score of 1.00 which means that it's capturing the entire positive Patients with disease cases. On the other hand, for class 0 the score was mildly low with a score of 0.71, as per the calculation the model overlooked 29% of the negative cases.

- F1-Score: The F1-Score is a combination of both results of precision and recall. As observed for class 1 the model scored a value of 0.97 which is a superior number indicates a great balance between the recall and precision. Furthermore, for class 0 we can observe that the model scored 0.83 which is slightly low due to the recall score not being flawless.
- Confusion Matrix: The purpose of having a confusion matrix is to measure the level of correctness and incorrection of a model's prediction. As we can observe, out of the total 39 samples the model was able to perfectly predict 5 TN (True Negative) values from class 0 and 32 values of TP (True Positive) from class 1. However, it overlooked and was not able to classify 2 FP (False Positive) values.

Overall, the model performs well and displays a great performance level when it comes to predicting and detecting a patient who's diagnosed with the Parkinson's Disease, but there is a small gap when it comes to identifying accurately the negative cases with healthy patients as per the evidence with the results of recall and F1-score for the class 0. However, there are always different methods to be used to improve the model. The most applicable approach to take for this dataset would either be:

- Tuning Parameters: in which the model's parameters will be optimized like the number of estimators in the classification of random forest and the total number of samples that are required to split the node.
- Augmented data: This method is applied by increasing the dataset size of the trained data by creating a completely new sample from the one that already exists, and this has proven to help the model to adapt better and perform well.

VII. Model Selection and Comparison:

The selected model will be Random Forest Model and the reason why it was favored then the rest of the other models is due to its fittingness in which it takes a combination of several decision trees to perform predictions. Moreover, this model is the most ideal when it comes to handling the Parkinson's Disease dataset as it can catch and establish complicated relationship between the features like for example the sound frequency measurement attributes since they have dependency amongst each other. Secondly, the model's performance since it outperforms different models that were selected in this research like SVM (Support Vector Machines) or KNN (K Neighbors Classifier) when it comes to accuracy as seen in Table 4 which makes this model the highest and the strongest candidate when compared to the rest of the other model. Moreover, other evaluation methods like precision, F1-Score and Recall as seen in figure 13.

Finally, in terms of suitability and cost factor, the Random Forest Model is considered the greatest option for this machine learning solution of a problem as mentioned earlier the performance and better data management, adding on to that, the cost of training is less when compared to KNN, SVM models.

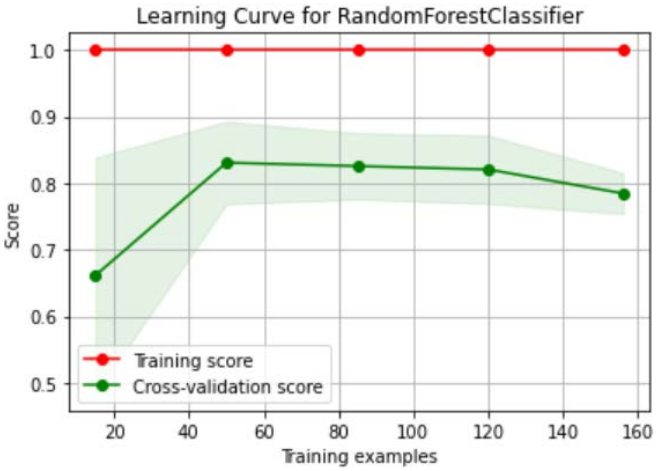

Figure 14 Learning Curve for Random Forest.

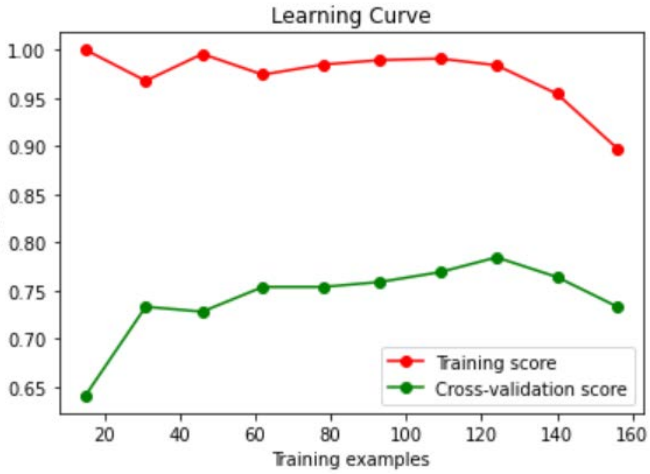

Figure 15 Learning Curve for KNN

The reason why we rejected the KNN model for instance is that despite the model accuracy of 82% or being known to for its simplicity. The performance of the model has a strong dependency for the total number of neighbors metrics. Moreover, the model possesses an assumption that instances that are similar or close to each other within the feature space will always have the same labels which is not always true, specifically when you're dealing with complicated targeted variables. Secondly, if the size of the dataset is huge, this model will have complexities to adapt to it, since it's computing the distance between training and test instances which can become relatively slow and expensive as the dataset size increases which is why this model is not ideal to large dataset and in real time the selected dataset for this research

paper will be much larger than this. In such cases it's crucial to select an efficient mode

Acknowledgement

I would like to express my sincere gratitude to my supervisor lecturer and colleagues in the classroom for their continues support, guidance, and encouragement throughout this journey in writing this research paper. Moreover, a special thanks to the providers of the dataset for allowing this data to be accessible to everyone to contribute and conduct research in this medical field of Parkinson's Disease by the help of using machine learning in applying a solution to possibly detect this disease much better and in an efficient way.

REFERENCES

[1] Ascherio, A. and Schwarzschild, M. A. (2016) 'The epidemiology of Parkinson's disease: risk factors and prevention', The Lancet Neurology, 15(12), pp. 1257-1272.

[2] Deo, R. C. (2015) 'Machine learning in medicine', Circulation, 132(20), pp. 1920-1930.

[3] Dorsey, E. R., Constantinescu, R., Thompson, J. P., Biglan, K. M., Holloway, R. G., Kieburtz, K., et al. (2007) 'Projected number of people with Parkinson disease in the most populous nations, 2005 through 2030', Neurology, 68(5), pp. 384-386.

[4] Eskofier, B. M., Lee, S. I., Daneault, J. F., Golabchi, F. N., Ferreira-Carvalho, G., Vergara-Diaz, G., et al. (2016) 'Recent machine learning advancements in sensor-based mobility analysis: Deep learning for Parkinson's disease assessment', in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 655-658.

[5] Jankovic, J. (2008) 'Parkinson's disease: clinical features and diagnosis', Journal of Neurology, Neurosurgery & Psychiatry, 79(4), pp. 368-376.

[6] Kalia, L. V. and Lang, A. E. (2015) 'Parkinson's disease', The Lancet, 386(9996), pp. 896-912.

[7] Little, M. A., McSharry, P. E., Roberts, S. J., Costello, D. A. E. and Moroz, I. M. (2007) 'Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection', BioMedical Engineering OnLine, 6(23).

[8] Postuma, R. B., Berg, D., Stern, M., Poewe, W., Olanow, C. W., Oertel, W., et al. (2015) 'MDS clinical diagnostic criteria for Parkinson's disease', Movement Disorders, 30(12), pp. 1591-1601.

[9] Sakar, B. E., Isenkul, M. E., Sakar, C. O., Sertbas, A., Gurgen, F., Delil, S., et al. (2013) 'Collection and analysis of a Parkinson speech dataset with multiple types of sound recordings', IEEE Journal of Biomedical and Health Informatics, 17(4), pp. 828-834.

[10] Saraswathi, S., Sundar, C. S. and Nagarajan, R. (2020) 'Machine learning techniques for the diagnosis of Parkinson's disease: a review', Biocybernetics and Biomedical Engineering, 40(4), pp. 1364-1378.

[11] Simuni, T. and Sethi, K. (2008) 'Nonmotor manifestations of Parkinson's disease', Annals of Neurology, 64(S2), pp. S65-S80.

[12] Tsanas, A., Little, M. A., McSharry, P. E. and Ramig, L. O. (2010) 'Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests', IEEE Transactions on Biomedical Engineering, 57(4), pp. 884-893.

[13] Little, M. A., McSharry, P. E., Roberts, S. J., Costello, D. A. E., and Moroz, I. M. (2007) 'Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection', BioMedical Engineering OnLine, 6(23).