Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)

Functional Requirements and Data flow Diagram for Parkinson's Disease detection using ML

Abstract:

The Functional Requirements Document (FRD) is a formal statement of an application's functional requirements. It serves the same purpose as a contract. Here, the developers agree to provide the capabilities specified. The client agrees to find the product satisfactory if it provides the capabilities specified in the FRD.

Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. The document should be tailored to fit a particular project's need. They define things such as system calculations, data manipulation and processing, user interface and interaction with the application.

The Functional Requirements Document (FRD) has the following characteristics −

It demonstrates that the application provides value in terms of the business objectives and business processes in the next few years.

It contains a complete set of requirements for the application. It leaves no room for anyone to assume anything which is not stated in the FRD.

It is solution independent. The ERD is a statement of what the application is to do— not of how it works. The FRD does not commit the developers to a design. For that reason, any reference to the use of a specific technology is entirely inappropriate in an FRD.

The functional requirement should include the following −

Descriptions of **data** to be entered into the system

Descriptions of **operations** performed by each screen

Descriptions of **work-flows** performed by the system

Descriptions of **system reports** or other outputs

Who can enter the **data** into the system?

How the system meets applicable **regulatory requirements?**

The functional specification is designed to be read by a general audience. Readers should understand the system, but no technical knowledge should be required to understand this document.

Functional Requirements Deliverables:

A Business Requirements Document (BRD) consists of −

Functional Requirements − A document containing detailed requirements for the system being developed. These requirements define the functional features and capabilities that a system must possess. Be sure that any assumptions and constraints identified during the Business Case are still accurate and up to date.

About Detecting Parkinson's Disease Project:

In this Python machine learning project, we will build a model to detect Parkinson's disease using one of the Classifier techniques known as RandomForestClassifier as our output contains only 1's and 0's. We'll load the dataset, get the features and targets, split them into training and testing sets, and finally pass them to RandomForestClassifier for prediction.

Business Process Model − A model of the current state of the process ("as is" model) or a concept of what the process should become ("to be" model)

Project Prerequisites:

Install the following libraries using pip :

pip install numpy , pandas , matplotlib , sklearn

The versions which are used in this parkinson's Disease Detection project for python and its corresponding modules are as follows:

1) python: 3.8.5
2) numpy: 1.19.5
3) pandas: 1.1.5
4) matplotlib: 3.2.2
5) sklearn: 0.24.2

Steps for detecting Parkinson's disease:

1) Import Libraries

Firstly we will import all the required libraries.

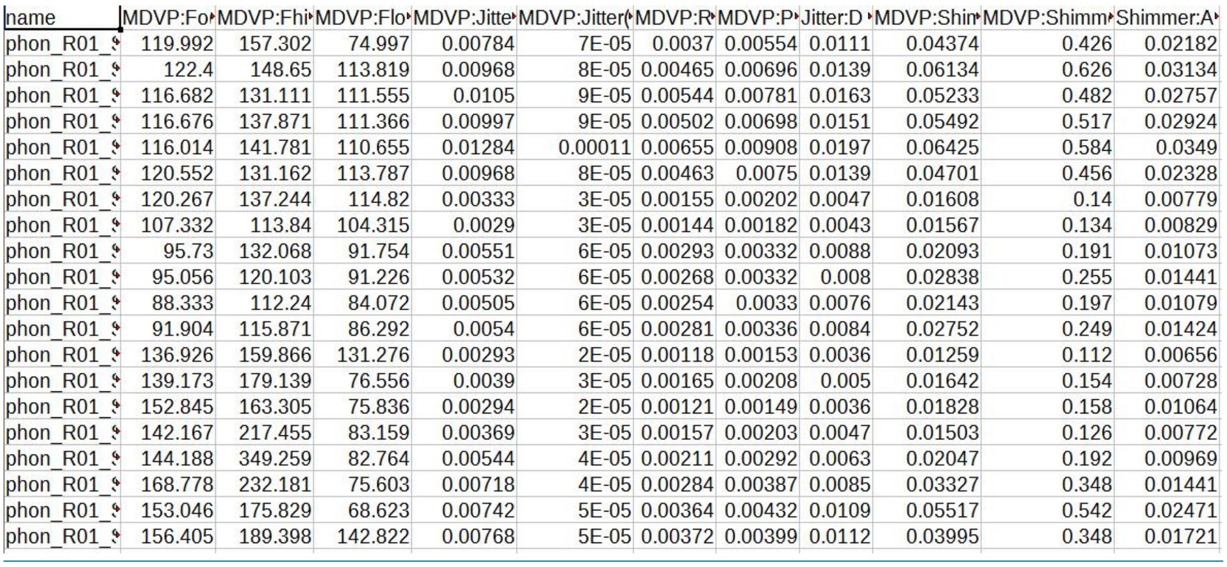
2) Preprocessing

Here the preprocessing process under goes in which read the File in dataframe using the pandas library.

dataset file :

This is the Demo dataset file.

Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)



name	MDVP:Fo	MDVP:Fhi	MDVP:Flo	MDVP:Jitte	MDVP:Jitter(	MDVP:R	MDVP:P	Jitter:D	MDVP:Shin	MDVP:Shimm	Shimmer:A
phon_R01_S	119.992	157.302	74.997	0.00784	7E-05	0.0037	0.00554	0.0111	0.04374	0.426	0.02182
phon_R01_S	122.4	148.65	113.819	0.00968	8E-05	0.00465	0.00696	0.0139	0.06134	0.626	0.03134
phon_R01_S	116.682	131.111	111.555	0.0105	9E-05	0.00544	0.00781	0.0163	0.05233	0.482	0.02757
phon_R01_S	116.676	137.871	111.366	0.00997	9E-05	0.00502	0.00698	0.0151	0.05492	0.517	0.02924
phon_R01_S	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.0197	0.06425	0.584	0.0349
phon_R01_S	120.552	131.162	113.787	0.00968	8E-05	0.00463	0.0075	0.0139	0.04701	0.456	0.02328
phon_R01_S	120.267	137.244	114.82	0.00333	3E-05	0.00155	0.00202	0.0047	0.01608	0.14	0.00779
phon_R01_S	107.332	113.84	104.315	0.0029	3E-05	0.00144	0.00182	0.0043	0.01567	0.134	0.00829
phon_R01_S	95.73	132.068	91.754	0.00551	6E-05	0.00293	0.00332	0.0088	0.02093	0.191	0.01073
phon_R01_S	95.056	120.103	91.226	0.00532	6E-05	0.00268	0.00332	0.008	0.02838	0.255	0.01441
phon_R01_S	88.333	112.24	84.072	0.00505	6E-05	0.00254	0.0033	0.0076	0.02143	0.197	0.01079
phon_R01_S	91.904	115.871	86.292	0.0054	6E-05	0.00281	0.00336	0.0084	0.02752	0.249	0.01424
phon_R01_S	136.926	159.866	131.276	0.00293	2E-05	0.00118	0.00153	0.0036	0.01259	0.112	0.00656
phon_R01_S	139.173	179.139	76.556	0.0039	3E-05	0.00165	0.00208	0.005	0.01642	0.154	0.00728
phon_R01_S	152.845	163.305	75.836	0.00294	2E-05	0.00121	0.00149	0.0036	0.01828	0.158	0.01064
phon_R01_S	142.167	217.455	83.159	0.00369	3E-05	0.00157	0.00203	0.0047	0.01503	0.126	0.00772
phon_R01_S	144.188	349.259	82.764	0.00544	4E-05	0.00211	0.00292	0.0063	0.02047	0.192	0.00969
phon_R01_S	168.778	232.181	75.603	0.00718	4E-05	0.00284	0.00387	0.0085	0.03327	0.348	0.01441
phon_R01_S	153.046	175.829	68.623	0.00742	5E-05	0.00364	0.00432	0.0109	0.05517	0.542	0.02471
phon_R01_S	156.405	189.398	142.822	0.00768	5E-05	0.00372	0.00399	0.0112	0.03995	0.348	0.01721

Fetch the features and targets from the dataframe. Features will be all columns except ‘name’ and ‘status’. Therefore we will drop these two columns. And our target will be ‘status’ column which contains 0’s(no parkinson’s disease) and 1’s(has parkinson’s disease)

3) Normalization

We will scale our feature data in the range of -1(minimum value) and 1(maximum value). Scaling is important because variables at different scales do not contribute equal fitting to the model which may end up creating bias. For that we will be using ‘MinMaxScaler()’ to fit and then transform the feature data.

4) Training and Testing

Split dataset into 80:20 ratio where 80 rows for training and 20 rows for testing purposes. For this we will pass scaled features and target data to ‘train_test_split()’.

5) Building the classifier model

We will use Random forest Classifier for the classification of our data points. So let’s see what random forest is.

What is a Random Forest Algorithm?

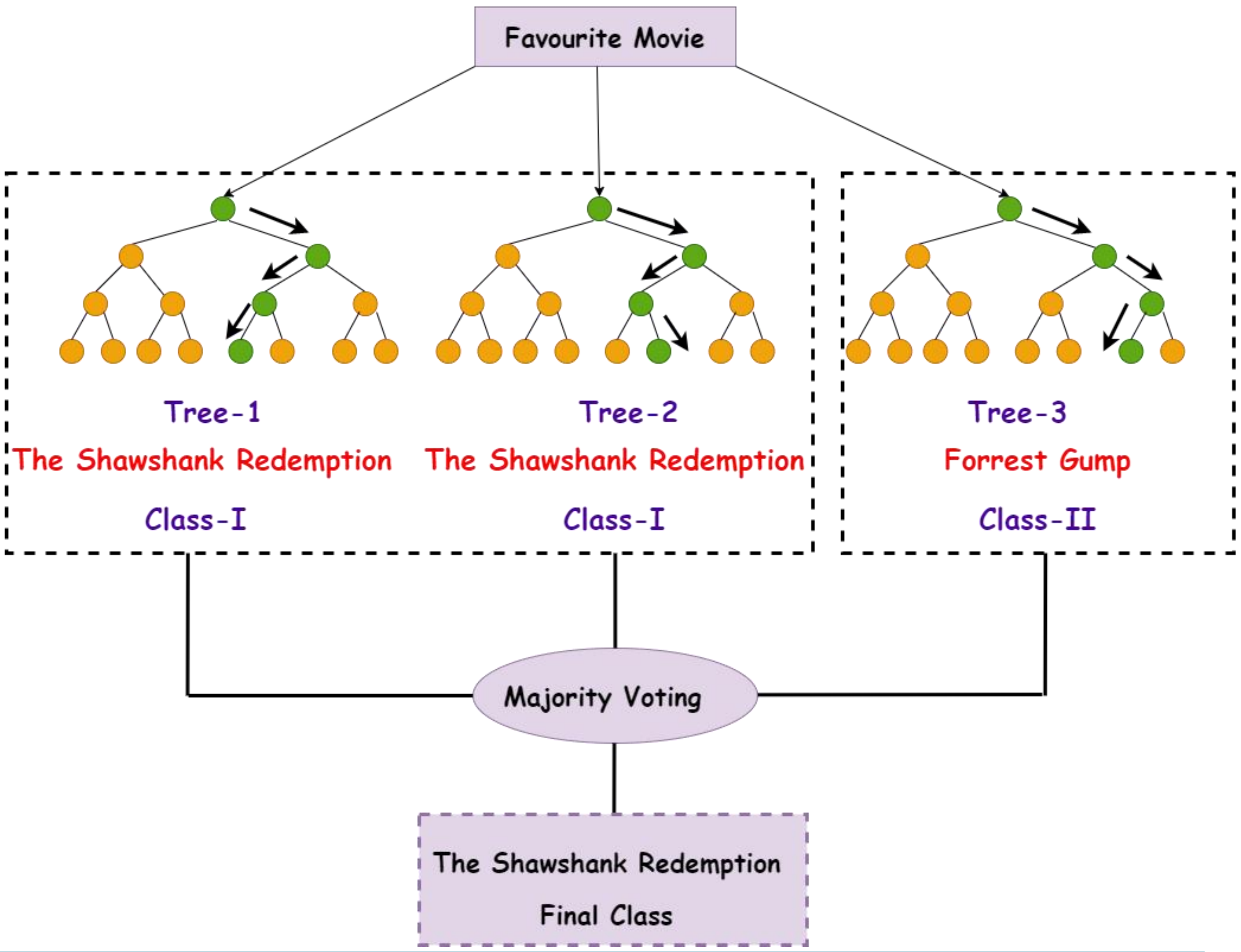
Random forest is a supervised learning algorithm. It can be used for both Classification and Regression problems. It uses an ensemble learning method known as ‘bagging’ (Bootstrap Aggregation) which is a process of combining multiple classifiers to solve a complex problem.

Random forest creates various random subsets of the given dataset and passes them to different numbers of decision trees and takes the prediction from each tree. Based on the majority votes of prediction, random forest takes the average to predict the output and also to increase the accuracy and overall result. As there are greater numbers of trees in random forests, it prevents the problem of overfitting.

Random forest searches for the most important feature while splitting a node which helps in building a better model.

Random forest example:

Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)



As you can see in the above image we are building a random forest for the topic "Favourite movie". We have used three trees for the classification. Let's say these three trees are three different persons. So after asking different questions and applying the conditions we can see that the first two people said their favorite movie is "The Shawshank Redemption" and one said "Forrest Gump".

As Random forest classifier uses majority voting technique, we get "The Shawshank Redemption" as our Final Class output.

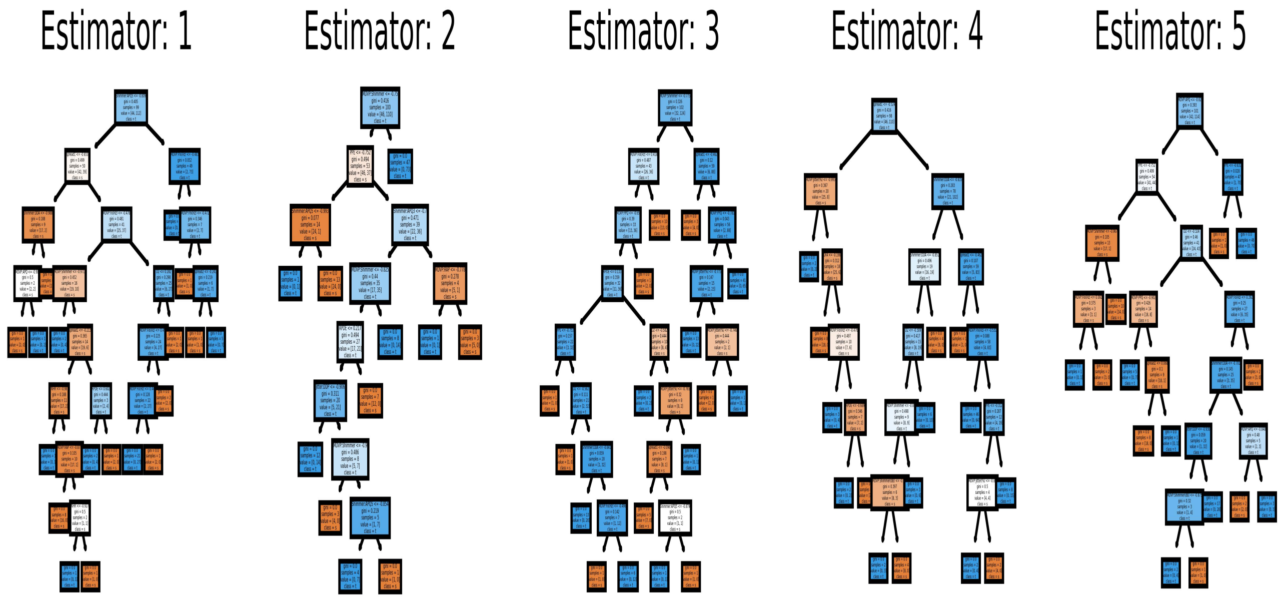
Initialize 'RandomForestClassifier' and train the model.

Output:

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=2, verbose=0,
                       warm_start=False)
```

After we train our model we can see how RandomForestClassifier fits our features and targets of the dataset. We will plot the first 5 trees of our classifier using the 'matplotlib' library.

Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)

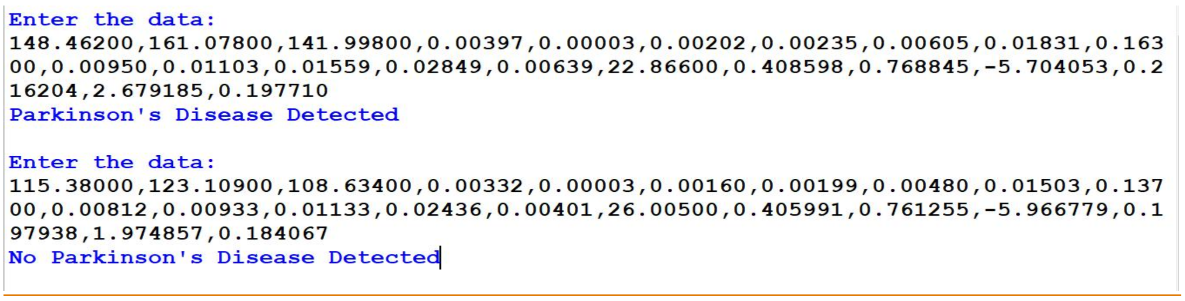


6) Prediction and Accuracy

Now we will predict our output(y_pred) for testing data(x_test) which is 20% of the dataset using the model which we have trained. Also, we will calculate accuracy, mean absolute error and root mean square error of our model.

7) Building the Prediction System.

Finally, we will take the user's input and check whether data has Parkinson's disease or not.



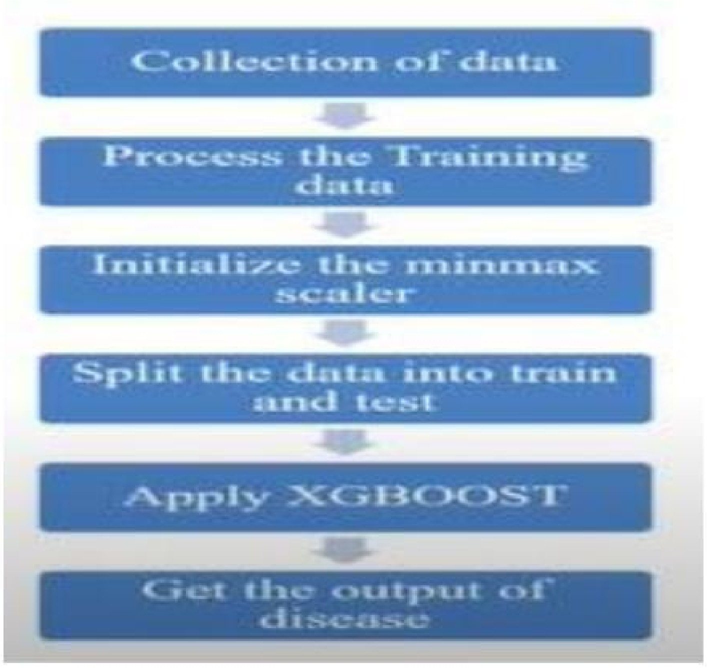
```
Enter the data:
148.46200,161.07800,141.99800,0.00397,0.00003,0.00202,0.00235,0.00605,0.01831,0.163
00,0.00950,0.01103,0.01559,0.02849,0.00639,22.86600,0.408598,0.768845,-5.704053,0.2
16204,2.679185,0.197710
Parkinson's Disease Detected

Enter the data:
115.38000,123.10900,108.63400,0.00332,0.00003,0.00160,0.00199,0.00480,0.01503,0.137
00,0.00812,0.00933,0.01133,0.02436,0.00401,26.00500,0.405991,0.761255,-5.966779,0.1
97938,1.974857,0.184067
No Parkinson's Disease Detected
```

System Context Diagram − A Context Diagram shows the system boundaries, external and internal entities that interact with the system, and the relevant data flows between these external and internal entities.

Flow Diagrams (as-is or to-be) − Diagrams graphically depict the sequence of operations or the movement of data for a business process. One or more flow diagrams are included depending on the complexity of the model.

Data Flow Diagram:

Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)


Collection of data
Process the Training data
Initialize the minmax scaler
Split the data into train and test
Apply XGBOOST
Get the output of disease

Conclusion:

In this Machine learning project, we developed a model using the RandomForestClassifier of the sklearn module of python to detect if an individual has Parkinson’s Disease or not. We got the machine learning model with 97.43% accuracy, which is good as our dataset contains less records.

Sreevarshan S(TL)
Vishwa S(M1)
Sreehari Pranesh K(M2)
Vinith Kumar S(M3)