

Disease Diagnosis Django Framework using Machine Learning

Mrinal Mayank¹, Ukshit Savaniya², Mahak Gawate³ and Payal Thakur⁴

¹⁻⁴Department of Computer Science & Engineering, Chandigarh University, Mohali, Punjab, India

Email: mrinalmayank7@gmail.com, ukshitsavaniya123@gmail.com, gawatemahak01@gmail.com, payal.e12720@cumail.in

Abstract— In the past it was very challenging for the doctors and specialist to predict the disease without the help of any specialized tools manually. Today with the help of futuristic technologies like AI and Machine Learning, it's now possible to make fast and accurate prediction of diseases without much effort. The proposed disease diagnosis framework is focused on training the Machine Learning (ML) models & integrating the serialized model with the user interface using django framework in python. The framework includes supervised machine learning using sklearn, Random Forest Algorithm, training of the ML model & serializing the trained ML model in order to integrate with the django framework. The model will be used to predict the binary classification of the disease using Random Forest Classifier & risk score using the Random Forest Regressor. We have identified the challenge of accurately and promptly diagnosing non-communicable diseases, as well as addressing the issues of limited accessibility and affordability of healthcare services.

Index Terms— Machine Learning, Django, Prediction, Python, Classification, Regression

I. INTRODUCTION

The intelligent machine learning models for disease diagnosis today is preferred over the manual prediction as it is more accurate and fast than the manual diagnosis. Manually it is difficult to handle huge amount of patient data which can be easily handled via emerging technologies like ML. In the proposed framework random forest algorithm [2], [6] is used for prediction of the disease. It is a widely used machine learning algorithm that falls under the category of supervised ML [3], which can be used as Regression as well as classification algorithm. The trained model is serialized as an object with the help of joblib library. Python is used for training the model as well as for the integration with the django framework [5], [9]. PIMA diabetes dataset [4] is used for training the model for predicting the occurrence of the diabetes. The dataset is divided into eight input measures and one outcome measure.

The trained model can be stored as a serialized object using joblib which can be used in backend of the interface for prediction of the disease. The accuracy of the Machine Learning algorithm depends on the dataset which is used for training the ML model. The trained model integration with the web framework is challenging and can be accomplished by following different methods. The research is particularly based on integrating the model at the backend of django framework [5], [9].

II. LITERATURE REVIEW

Manoj Diwakar et al. [11] proposed methodology to predict the heart disease using ML and Image Fusion where

he classified various types of machine learning techniques like Naive bayes, ANN, KNN and SVM. Preprocessing of the dataset is the most sensitive step which prepares the dataset to get more precise outcomes. Quality of the dataset plays the key role in forecasting the accuracy of the machine learning model. The importance of providing tailored care for specific types of heart diseases is emphasized, highlighting the need for a dataset consisting of appropriate samples and reliable data to create a predictive model. The paper stresses the significance of dataset pre-processing, which plays a critical role in optimizing the performance of the machine learning algorithm.

Sumathi M S et al. [12] elaborated analysis & performance of ML on diagnosis of diseases. Algorithms employed are MLP (a deep learning model), SVM, Decision Tree, KNN, Gradient Boosting, Random Forest Classifier and Logistic Regression. The paper shows the implementation of ensemble ML Models and also compared the various models like random forest, gradient boosting, logistic regression, decision tree, SVM, KNN etc. with respect to various parameters like accuracy, precision and recall. Based on these parameters and models, the paper presents the excellent example of classification algorithms in diagnosis of heart disease and cholera disease.

Chinmayi Thallam et al. [13] proposed methodology to predict the lungs cancer in the early stages with various machine learning algorithms. The research shows that SVM algorithm is not suitable for large dataset and dataset with more noise, Random Forest Model becomes slow in real time with large number of trees, KNN is only applicable for a balanced data. The research helps a lot to consider and evaluate the best model for prediction of lungs disease. Artificial Neural Networks (ANN) were found to present challenges in dealing with numerical data and network duration estimation.

Ankit Verma et al. [14] performed the integration of web application with machine learning. The work shows the web application integration with NLP model which is used for text analysis, the major challenge faced is the implementation of NLP through backend. The core technologies used were NLP, PostgreSQL, Django and cloud deployment with AWS cloud. React JS, Bootstrap, React, JavaScript were used in the frontend to make a responsive web application. The model is created using the SK-learn library in Python and is deployed through Flask. Expansion plans for the project involve targeting universities across the country. During the sign-up process, users can select their specific university from a dropdown list. This segmentation ensures that each user is directed to the appropriate university, avoiding any confusion or mismatches among students.

Mingxia Liu et al. [15] proposed the analysis of Brain morphometric pattern using both classification as well as the regression approaches using the deep multi-task & multi-channel CNN. The CNN model is used to diagnose the Alzheimer's Disease. In the model implementation both using both MRI data & demographic information like gender, age etc. were used. The proposed approach starts by identifying discriminative landmarks from MR images using a data-driven method. Multiple image patches are then extracted around these landmarks. The framework employs a deep multi-task multi-channel convolutional neural network to jointly perform classification and regression tasks. Importantly, the demographic information is explicitly incorporated into the learning process, enhancing the model's performance. The effectiveness of the DM2L framework is evaluated on four publicly available datasets. The experimental results demonstrate superior performance compared to several state-of-the-art approaches in both disease classification and clinical score regression tasks.

Suryadiputra Liawatimena et al. [16] proposed the detailed study of metrics measurement of django framework using pylint and radon like LOC, SLOC, Cyclomatic Complexity etc. The Django framework is highlighted as a compact Python framework designed for rapid development in fast-paced environments, particularly suited for working with relational databases. The article then delves into software metrics for Django, using Radon as a tool to provide cyclomatic complexity raw metrics. These metrics include measures such as source lines of code (SLOC), comment lines, number of blank lines, maintainability index, and Halstead metrics. Additionally, pylint is mentioned as a source code analyser that helps identify errors in programming and enforces adherence to coding standards.

Ajinkya Kunjir et al. [17] proposed the analysis of COVID-19 WHO Dataset, where the research is based on the regression techniques such as CNN, Decision Tree and LSTM & these techniques are used for the analysis of the COVID-19 dataset. The analysis can be used in providing the detailed and statistical idea of the data and is helpful in reducing the impact of the global epidemic.

Umar Sidiq et al. [18] used various data mining methods for the analysis of thyroid disease where three different approaches had been utilized which includes Stacking Ensemble, Vote Ensemble & Neural Net. In comparison of the three approaches Stacking Ensemble Classifier returns the highest accuracy rate.

Hadeel S. Obaid et al. [19] elaborated the influence of data preprocessing over the accuracy of the ML Models. In the research work three widely used data preprocessing techniques were described which includes normalization, dimension reduction and discretization.

Tri Doan et al. [20] proposed the analysis of different approaches particularly for regression models. The work presents the insights of CART (Classification & Regression Trees), Bagging CART, Random Forest etc. & the

approach which includes dimensionality reduction , meta features , evaluation of metrics and building the regression model.

III. PROPOSED METHODOLOGY

A. Training Phase

In training phase the model is to be trained with respect to the dataset and the algorithm. Before building the model there are sequence of steps involved to prepare data. The stages of the training phase are as follows:

- Gathering the Dataset: The dataset is required for training the model. Here we are using supervised Machine Learning [3], so the dataset must contain the outcome and based on that the model will make the prediction.
- Data Pre-processing: The raw data must be cleaned before training, as it may contain the null/ missing values which will create ambiguity in the prediction. The dataset must contain the numerical values [1] for making the prediction, hence before training, the string data must be mapped with the respective integer values. Also all the missing cells/ values must be replaced with some mean value of the desired column.
- Data Analysis: The processed data is analysed to observe the hidden correlation in the dataset and the statistical measures of the data like the mean, maximum, minimum etc.
- Building the model: The cleaned data is used to build the model with suitable algorithm and the accuracy is analysed. Random forest algorithm [2], [6] is used to build the model for the proposed framework.

B. Inference Phase

In the inference phase [1], trained model is used to predict the outcome with completely isolated data. The model will be serialized using joblib library. The serialized model is then integrated with the interface for prediction of the disease by general user.

IV. PREREQUISITE

- Numpy [8] & pandas [7]: These are powerful python libraries for manipulation with the dataset.
- Matplotlib [10] & seaborn : Used for data visualization in the form of interactive charts and graphs.
- train_test_split (sklearn.models) [6] : Used for splitting the data in 2 parts, one is the train data and other is the test data. Generally the train set is taken in the major proportion & test set in the minor proportion.
- accuracy_score (sklearn.metrics) [6]: Used for finding the accuracy of the trained model.
- Joblib : used for serializing and de-serializing the machine learning model for making prediction.
- Random Forest Classifier [6]: Imported from sklearn.ensemble. Used to build classification model.
- Random Forest Regressor [6]: Imported from sklearn.ensemble. Used to build regression model.
- Django [9]: It is a web framework which will be used to build the interface and integrate the serialized machine learning model with the frontend. The machine learning model is implemented in the backend of the framework.

V. DATA PRE-PROCESSING

The objective of the research work is to predict the occurrence of diabetes with the help of Machine Learning. The initial step is to read and load the raw dataset in order to get some insights of the dataset.

Syntax to load the raw dataset:

```
raw_data= pd.read_csv('diabetes.csv')
raw_data.head(5)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Figure 1. Diabetes Dataset

Now the next step is to determine the data type of the parameters and analyze the missing values. Here data.info() function is used to find the null values and data type of the attributes of the dataset. Later we will use this information for replacing the missing objects in the given dataset. In Fig.2 we can see that the dataset does not

contain any missing cells. But the missing values may be present in the form of zeroes as it is impossible to have the parameters Body Mass Index (BMI), Glucose, Blood Pressure (BP), Skin Thickness, Insulin equals to zero. Now we first need to visualize the number of zeroes (non-acceptable values) in the columns BMI, Glucose, Blood Pressure (BP), Skin Thickness and Insulin. The 0's are replaced with nan and then number of nan values are visualized in each column with the help of a bar graph.

#	Column	Non-Null Count	Dtype
0	Pregnancies	768 non-null	int64
1	Glucose	768 non-null	int64
2	BloodPressure	768 non-null	int64
3	SkinThickness	768 non-null	int64
4	Insulin	768 non-null	int64
5	BMI	768 non-null	float64
6	DiabetesPedigreeFunction	768 non-null	float64
7	Age	768 non-null	int64
8	Outcome	768 non-null	int64

dtypes: float64(2), int64(7)
memory usage: 54.1 KB

Figure 2. Data Information

```
Code snippet to plot a bar graph to visualize the missing values in the form of 0's:
non_acceptable_zero = ["Glucose", "Blood Pressure", "Skin Thickness", "Insulin", "BMI"]
for column in non_acceptable_zero:
    raw_data[column].replace(0, np.nan, inplace=True)
plt.figure(figsize=(10,4))
bargraph = sns.barplot(x=data.isna().sum(),y=data.columns)
for it in bargraph.patches:
    bargraph.annotate(text=f"{it.get_width():.0f}",
        xy=(it.get_width(), it.get_y()+it.get_height()/2),
        xytext=(5, 0), textcoords='offset points',
        ha="left", va="center", )
plt.show()
```

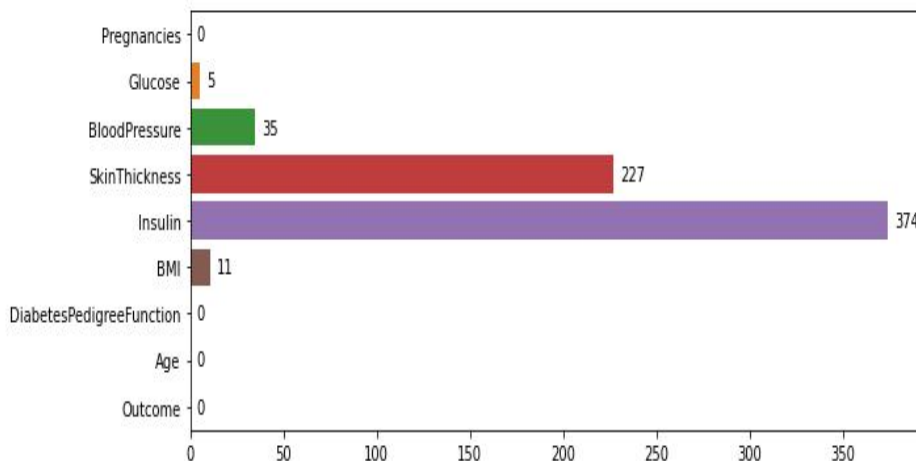


Figure 3. Bar graph with nan values

The bar graph in Fig.3 shows the number of nan values in each column. These values will disturb the prediction accuracy of the model, therefore it needs to be replaced with mean value of all the acceptable values in the desired column. The data does not need more pre-processing as all the fields are numeric and missing values are also replaced with the mean.

VI. DATA ANALYSIS

The preprocessing of data is followed by data analysis where the dataset pattern is visualized. It is important to analyze the data in order to gain data insights, find correlations and hidden patterns inside the dataset. Here are the list and usage of different functions we are using to analyze the dataset.

describe() function: It gives the statistical measures like mean, count, maximum, minimum value etc. of the dataset parameters. This helps us in generalizing the input insights so that we can use those insights in building a precise model.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	121.686763	72.405184	29.153420	155.548223	32.457464	0.471876	33.240885	0.348958
std	3.369578	30.435949	12.096346	8.790942	85.021108	6.875151	0.331329	11.760232	0.476951
min	0.000000	44.000000	24.000000	7.000000	14.000000	18.200000	0.078000	21.000000	0.000000
25%	1.000000	99.750000	64.000000	25.000000	121.500000	27.500000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.202592	29.153420	155.548223	32.400000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	155.548223	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

Figure 4. Data Description

Range of Outcome Values: The outcomes are classified as binary variables, where Zeroes represents the non-diabetic patients and Ones represents the patients suffering with diabetes. Thus we can use a binary classification machine learning model in this dataset in order to predict the outcomes.

Grouping outcome: The outcome is grouped with respect to the input values using the function `groupby()` concatenated with the `mean()` to obtain the mean input values. In Fig.6 the outcome values is mapped with the input values which shows the average input values with respect to the binary outcome classification. For example the person is more likely to suffer from diabetes if glucose level is 142 and may be not diabetic if glucose level is 110

```
1 data['Outcome'].value_counts() #count values
0    500
1    268
Name: Outcome, dtype: int64
```

Figure 5. Range of Outcome

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
Outcome								
0	3.298000	110.710121	70.935397	27.768851	142.210761	30.888434	0.429734	31.190000
1	4.865672	142.165573	75.147324	31.736944	180.431548	35.384757	0.550500	37.067164

Figure 6. Outcome Grouping

VII. BUILDING THE MODEL

A. Test Train Split

The processed data is trained in a sequence of steps from splitting of test train data to model fitting. Accuracy is obtained after training the Machine Learning (ML) model. The dataset is partitioned into 2 parts and termed as test data and train data. Both holds two subset X & Y where the subset X holds the input and Y holds the output. `train_test_split` [6] is used for partitioning of test and train data where the size of the test data is taken as 35% and size of train data is taken as 65%. Code snippet:

```
X = data.drop(columns='Outcome', axis=1)
```

```
Y = data['Outcome']
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.35, stratify=Y, random_state=3)
```

B. Fitting the Model

The train and test data are now obtained. Now it needs to be fitted to the model for the training. Here two models will be used, one is the random forest classifier [6] and other is the random forest regressor [6]. The classification algorithm will do the binary classification of the disease i.e. 0 or 1 and the regression algorithm will predict the mean of all the outcomes of the decision tree. Code snippet for fitting the model:

```

Random Forest Classifier [6] from sklearn.ensemble import RandomForestClassifier
RFCClassifierModel = RandomForestClassifier(random_state=16)
RFCClassifierModel.fit(X_train, Y_train)
Random Forest Regressor [6]
from sklearn.ensemble import RandomForestRegressor
RFRegressorModel = RandomForestRegressor(random_state=26)
RFRegressorModel.fit(X_train, Y_train)

```

C. Accuracy

accuracy_score() [6] function is used to find the accuracy of the models. The accuracy is the probability of correctness in the outcome. It depends on the size and quality of the dataset.

TABLE I. MODELS ACCURACY

Algorithm	Test Data / Train Data	Accuracy
Random Forest Classifier [6]	Train Data	100 %
	Test Data	80.669 %
Random Forest Regressor [6]	Train Data	100 %
	Test Data	80.408

D. Model Serialization

The trained model is serialized using joblib and can be used as a file in the backend of the interface for prediction. The following format should be followed while serializing the model using joblib library:

```

joblib.dump(Model_name,'file_name.pkl')
joblib.dump(RFCClassifierModel,'RFCClassifierDiabetes.pkl')
joblib.dump(RFRegressorModel,'RFRegressorDiabetes.pkl')

```

VIII. DJANGO INTEGRATION

The serialized model is integrated with django [9] where it is loaded with the help of joblib library. The implementation is done in django views.py in a series of steps.

A. Load the Model

```

RFCClassifier=joblib.load('RFCClassifierDiabetes.pkl')
RFRegressor=joblib.load('RFRegressorDiabetes.pkl')

```

B. User Input

In order to make the prediction the user need to provide the input data to the model. Django forms are used to get the input from the user, all the 8 input fields can be implemented in django forms with a unique name. The input name is then called in the views with GET and POST methods. Format to get the input values from the form is given below:

```

if request.method == 'POST':
var_name=request.POST.get('input_name')

```

C. Conversion into numpy array

```

input_data = (var1 , var2, .....var_n)
ny_arr= np.asarray(input_data)
input_data_reshaped = ny_arr.reshape(1,-1)

```

D. Making Predictions

The predictions are made by passing the reshaped numpy array input in the parameter of the predict function which is to be included with the loaded model as given below:

```

Predict1= RFCClassifier.predict(input_data_reshaped)
Predict2= RFRegressor.predict(input_data_reshaped)*100

```

IX. RESULT & DISCUSSION

The objective of the research work is to predict the occurrence of diabetes with the help of Machine Learning (ML) and integrate it with the suitable interface. This research is on supervised machine learning [3] and training the

model with random forest algorithm [2], [6]. The accuracy of both the trained models is noted in the Table- I and is calculated using python. This methodology can also be applied to predict some other diseases with supervised Machine Learning [3]. These models can be used in healthcare applications in order to predict the outcomes.

A. Predictions

In Table- II the sample input data is tested and the results are compared with the expected results and the actual results. The 0/1 prediction is made by the Random Forest Classifier [6] Model and Risk Score is generated by the Random Forest Regressor [6] Model.

TABLE II. PREDICTIONS

Input Array (8 input fields)	Expected Result (0/1)	Actual Result (0/1)	Risk Score
(1,89,66,23,94,28.1,0.167,21)	0	0	6%
(0,137,40,35,168,43.1,2.288,33)	1	1	86%
(3,78,50,32,88,31,0.248,26)	1	1	74%
(1,189,60,23,846,30.1,0.398,59)	1	1	57.9%
(1,103,30,38,83,43.3,0.183,33)	0	0	15%
(13,145,82,19,110,22.2,0.245,57)	0	0	23%
(0,162,76,56,100,53.2,0.759,25)	1	1	68%

B. Random Forest (RF) Classifier VS Regressor

Random forest algorithm [2], [6] works by generating the N decision trees and combines them to get the final outcome, the number of trees is directly proportional to accuracy. Random Forest being a supervised ML algorithm is used for both classification problems & regression problems where the classification algorithm termed as random forest classifier [6] which predict the binary outcome and the regression algorithm is termed as random forest regressor [6]. The output of the classifier algorithm is based on the majority voting whereas the output of the regressor algorithm is based on averaging or mean.

X. CONCLUSION

The applicability of Machine Learning Algorithms helps in fast, accurate and early prediction of various diseases which include diabetes, cardiovascular disease, cancer & many more. Many algorithms are used for prediction depending upon the dataset where Random Forest Algorithm [2], [6] proves to be more accurate in terms of correctness of the outcome and its capability to avoid data over fitting [3]. The inference phase includes the implementation of working model with the application for the user usage. In the paper we had discussed the integration of the model with django framework [5], [9] and object serialization with joblib. The model result depends upon the accuracy of the model, so far most of the predictions are correct. The future research works may include the improvement of current Algorithms and using more complex algorithms to increase the efficiency and accuracy of the model & model integration with some other frameworks.

REFERENCES

- [1] Venugopalkadamba, Disease Prediction Using Machine Learning, geeks for geeks, Available at- <https://www.geeksforgeeks.org/disease-prediction-using-machine-learning/>, 15 Sep, 2021
- [2] Pooja Anbuselvan, Heart Disease Prediction using Machine Learning Techniques, International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 9 Issue 11, November-2020
- [3] Marouane Fethi Ferjani, Disease Prediction Using Machine Learning, 10.13140/RG.2.2.18279.47521, December- 2020
- [4] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science.
- [5] Akash Chauhan, Creating a Machine Learning Based Web Application Using Django, Towards Data Science, Available at- <https://towardsdatascience.com/creating-a-machine-learning-based-web-application-using-django-5444e0053a09>, 1st September-2020
- [6] Pedregosa, F. & others, Scikit-learn (Machine learning in Python), Journal of Machine Learning Research, 2011, 2825–2830
- [7] McKinney, W., & others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).
- [8] Harris, C. R. & others (2020). Array programming with NumPy. Nature, 585, 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- [9] Django (Version 3.2) [Computer Software]. (2021). Retrieved from <https://www.djangoproject.com/>.
- [10] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90–95.

- [11] Manoj Diwakar & others, Latest trends on heart disease prediction using machine learning and image fusion, Volume 37, Part 2, 2021, Pages 3213-3218, Available at- <https://doi.org/10.1016/j.matpr.2020.09.078> , Elsevier
- [12] Sumathi M S & others, Analysis and Performance of Machine Learning Algorithms on Disease Diagnosis , 3rd International Conference on Energy, Power and Environment: Towards Clean Energy Technologies , DOI: 10.1109/ICEPE50861.2021.9404447, IEEE , 2021.
- [13] Chinmayi Thallam & others, Early Stage Lung Cancer Prediction Using Various Machine Learning Techniques, 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), DOI: 10.1109/ICECA49313.2020.9297576, IEEE, 2020
- [14] Ankit Verma & others, Web Application Implementation with Machine Learning, 2nd International Conference on Intelligent Engineering and Management (ICIEM), DOI: 10.1109/ICIEM51511.2021.9445368, IEEE , 2021
- [15] Mingxia Liu & others, Joint Classification and Regression via Deep Multi-Task Multi-Channel Learning for Alzheimer's Disease Diagnosis, DOI: 10.1109/TBME.2018.2869989, IEEE Transactions on Biomedical Engineering (Volume: 66, Issue: 5, May 2019)
- [16] Suryadiputra Liawatimena & others, Django Web Framework Software Metrics Measurement Using Radon and Pylint, Indonesian Association for Pattern Recognition International Conference (INAPR), DOI: 10.1109/INAPR.2018.8627009 , IEEE , 2018.
- [17] Ajinkya Kunjir & others , A Comparative Study of Predictive Machine Learning Algorithms for COVID-19 Trends and Analysis, IEEE International Conference on Systems, Man, and Cybernetics (SMC), DOI: 10.1109/SMC42975.2020.9282953, IEEE , 2020.
- [18] Umar Sidiq & others, Disease diagnosis through data mining techniques, International Conference on Intelligent Computing and Control Systems (ICCS), DOI: 10.1109/ICCS45141.2019.9065824 , IEEE, 2019
- [19] Hadeel S. Obaid & others, The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning, 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), DOI: 10.1109/IEMECONX.2019.8877011, IEEE, 2019
- [20] Tri Doan & others, Selecting Machine Learning Algorithms Using Regression Models, IEEE International Conference on Data Mining Workshop (ICDMW), DOI: 10.1109/ICDMW.2015.43 , IEEE , 2015.