Interactive System for ML Model Implementations

Bhavyadeep Purswani[1], Lekhana G[2], Harshitha A[3], Krithi Hegde[4]

[5]Under the guidance of **Smt. Jalaja G, Associate Professor,**

[1-5]Department of Computer Science & Engineering, BNM Institute of technology, Bangalore – 560070

---***---

***Abstract* —** Machine learning (ML) is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. ML finds its application not only in the field of Computer Science but also in various other fields like Biological studies, Ecology, Military etc. A subset of artificial intelligence (AI), machine learning (ML) is the area of computational science that focuses on analyzing and interpreting patterns and structures in data to enable learning, reasoning, and decision making outside of human interaction. Simply put, machine learning allows the user to feed a computer algorithm an immense amount of data and have the computer analyze and make data-driven recommendations and decisions based on only the input data. If any corrections are identified, the algorithm can incorporate that information to improve its future decision making. Data is the lifeblood of all business. Data-driven decisions increasingly make the difference between keeping up with competition and falling further behind. Machine learning can be the key to unlocking the value of corporate and customer data and enacting decisions that keep a company ahead of the competition. Building a system that can perform ML tasks for user without writing any lines of code can help incorporate ML in each sector easily and without much investment. The user can visually select the tasks to be performed and see the results directly on screen. The system can help people learn ML without learning to code.

***Keywords* —** Artificial intelligence, biomedical, reaction site, reaction sequence, reaction performance

I. INTRODUCTION

Importance of ML in every sector is growing every day, hence more people are investing time to learn it. Computer scientists are hired specifically for ML in other sectors. Hence, each sector has to either train people for tasks of ML or hire a Computer Scientist for the same. Machine learning has applications in all types of industries, including manufacturing, retail, healthcare and life sciences, travel and hospitality, financial services, and energy, feedstock, and utilities. Use cases include:

• Manufacturing: Predictive maintenance and condition monitoring

• Retail: Upselling and cross-channel marketing

• Healthcare and life sciences: Disease identification and risk satisfaction

• Travel and hospitality: Dynamic pricing

• Financial services: Risk analytics and regulation

• Energy: Energy demand and supply optimization

Building a system that can perform ML tasks for user without writing any lines of code can help incorporate ML in each sector easily and without much investment. The user can visually select the tasks to be performed and see the results directly on screen. The system can help people learn ML without learning to code. The main aim is to develop a system using which the user can work on the dataset without having to design or develop the ML algorithms and code in person.

The system would contain the following functional units-

▯ Dataset upload- This unit would allow the user to upload the dataset and to manage attributes like test/train split.

▯ Preprocessing- This unit would make suggestions to the user about various preprocessing techniques that could be applied to the dataset to increase the accuracy.

▯ Data analysis- This unit would allow the users to plot graphs between different attributes and analyze the relation between them.

▯ Application of ML algorithms- This unit would suggest the user about the best algorithm that can be applied on the dataset to get best results.

▯ Result analysis- Comparing the accuracy score of various algorithms and selecting the best one.

We shall note that the preprocessing suggestions made could be accepted or declined by the user. And these preprocessing suggestions could include normalization, standardization etc..,

We can thus infer that the user need not know into depths about the development of the model and basic knowledge about the concept of the algorithm to be used is sufficient to operate on the model which helps save time, cost and money put into the entire process of model development and usage.

Our system provides the user with an interface which accepts dataset, predicts the kind of problem, and suggests the kind of pre-processing that is to be done, implements the algorithm and provides the user with an understandable output. Therefore, using this system shall help an organization by providing it a smart all in one ML algorithm implementation toolkit.

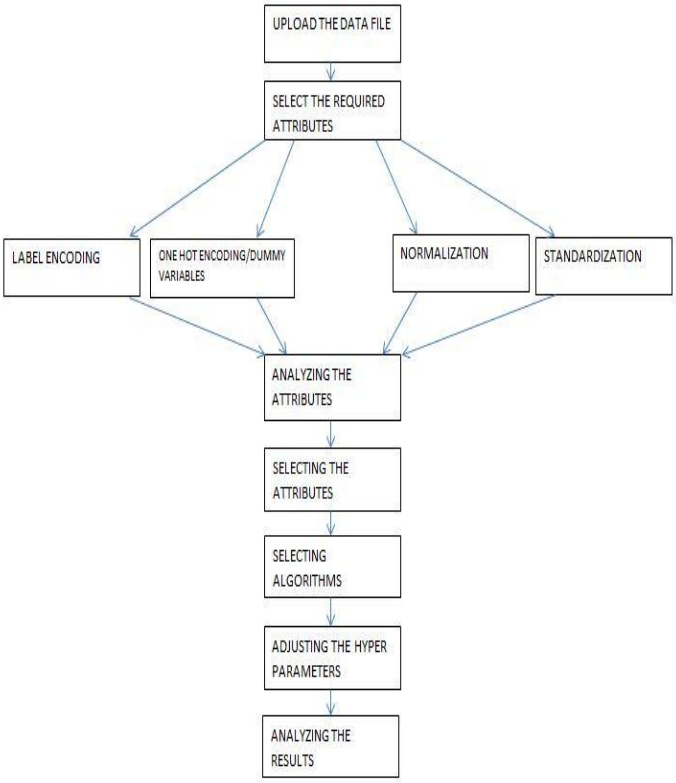


Figure 1.1 Process Flow of the model

II. WORKING

Functional Requirements

The main functional requirements of the system are as follows,

 The application should provide easy interface for user to upload the data.

The entire application relies on processing the data given by the user, hence the application should provide easy and convenient way to upload the data.The system has certain restrictions on the data that can be uploaded such as the limit of the file size should be maximum 10Mb.

 The application should allow flexible data pre-processing

Every machine learning workflow requires the data to be in certain format before feeding into the algorithm. The application should allow the user to manipulate and process the data according to the user needs and also according to the requirements of the algorithm.

 The application should allow the user to analyse the data

The machine learning algorithm cannot be applied until the data is analysed and the best way to do that is provide a means to visualize the data. The application must contain a means of visualizing the data graphically before or after pre-processing the data. A minimum of bar graph, line graph and scatter graph are required to visualize the data for machine learning application.

 The application should allow user to apply various machine learning algorithm

The application main goal is that the user can try out various algorithms and decide which is the best algorithm for the data. The application should allow the user to select from different types of algorithms. The system should allow both regression and classification type of algorithms for the data.

 The system should help user with application of algorithm

One of the objectives of the application is to make applying machine learning easy for the user. This can be done if the system is suggested with the possible algorithm that can be applied by the user on the data provided by them. The user should be allowed to either apply algorithm of his choice or

 The application should allow the user to see the results of the applied algorithm

The end goal of applying machine learning algorithm is to know the accuracy and efficiency of the algorithm and also to obtain the predicted values. The application should allow the user to download and view the predicted values and also obtain the accuracy and efficiency of the model.

Non- Functional Requirements

 The application should provide an easy user interface.

The user should be able to easily see understand the procedure and the should be able to access and understand the system.

 The errors should be handled.

The possible errors should be handled properly. Every possible scenario where the system could fail to produce the correct output should be anticipated as much as possible and should be resolved.

 Displaying proper messages

The system should display proper and appropriate messages for the user so that the user can navigate through the application conveniently. If the user has made an error it should be displayed properly and should be alerted so that the user doesn't repeat it again.

 The UI should be convenient for the user

The User Interface should be clear and visible to the user. The font size should be subtle and convenient for the user to interact with the system. The buttons and pages should be clearly marked and the content on the screen should be neatly organized so that it is clearly understood to the user.

The proposed system consists of following elements:

 Flask Server: Flask is a library in python which allows the developer to host a REST service. It uses Jinja templating, which is a template engine for the Python programming language and is used for generating markups from python itself. In the proposed system, Flask server is used to host a service that is utilized by the User Interface (UI) for performing all the Machine Learning (ML) related tasks. This server is the place where the logic of the application resides. The UI makes asynchronous calls to various services of the Flask server to perform actions like uploading the dataset, pre-processing the dataset, selecting the appropriate ML algorithm, viewing different graphs and training the model for selected algorithm.

 Architecture: The proposed system follows a combination of API Gateway and Chain of Responsibility Microservice deign patterns. The API Gateway Microservice design pattern involves a Gateway through which all the requests are directed to the various services, here, the Flask Server acts as a Gateway and ever request made by the UI is directed from this server to the corresponding service hosted withing the flask server.

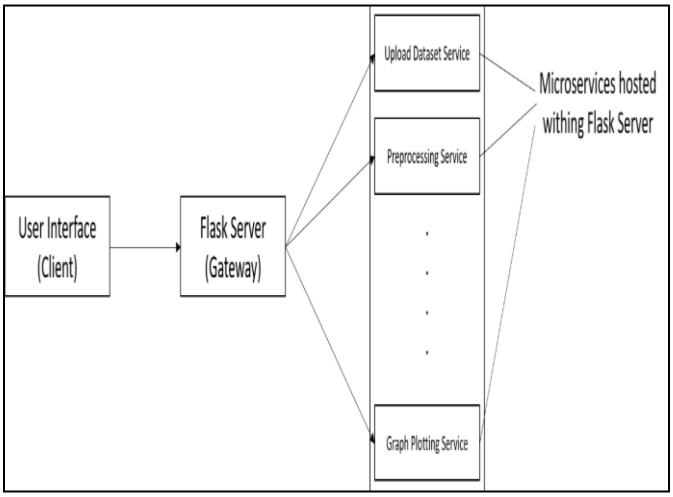


Figure 2.1: API Gateway Architecture Basic

In Chain of Responsibility Microservice design pattern, the result of one microservice is fed as an input to another microservice. Here, the output of the Interactive system for upload dataset microservice will be fed as an input to the pre-processing microservice, the output of this pre-processing microservice will be fed as an input to the graph plotting microservice and so on.

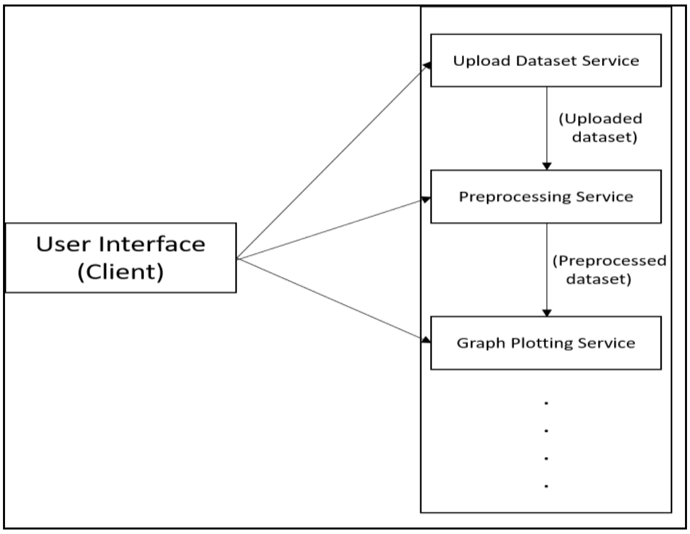


Figure 2.2: Chain of Responsibility Architecture

A combination of both the design patterns is shown in Figure 4.3.

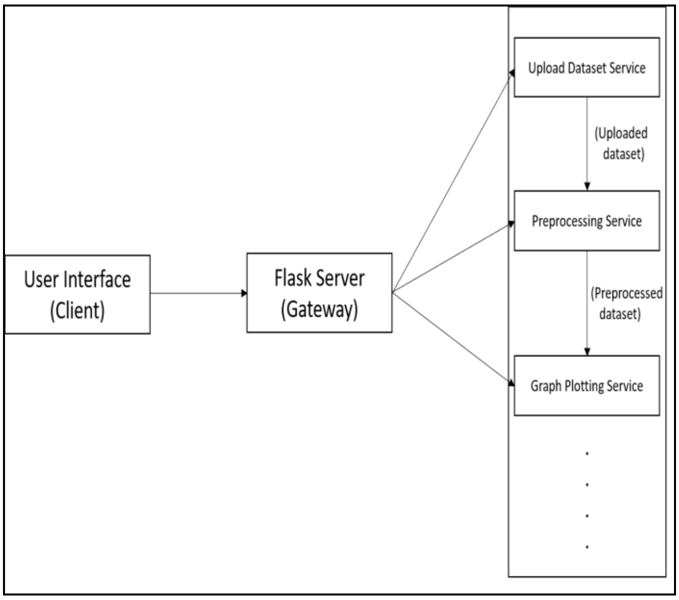


Figure 2.3: Combination of API Gateway and Chain of Responsibility Architecture

 User Interface: The interface of the application will be a web interface made using technologies like, HTML, CSS and JavaScript. All the service request made to the server would be asynchronous and done using JavaScript. For each different module, a separate webpage will be displayed which would carry forward the state from the previous webpage.

File Processing Service:

This service is responsible for uploading the datasets into the application server. The uploaded dataset is stored in the data folder of the server. The application performs two types of check before uploading the file. Those are the following checks,

 File Extension type: The current extension type allowed in the system is only csv.

 File Size: The application performs a check for the size of the uploaded file, the allowed size currently 10MB.

The file processing service is also responsible to display the file head, i.e. the first five rows of the uploaded file. This ensures the user that the correct file has been uploaded.

The file processing service is also responsible for adding a header to the rows if there no headers present in the file. The user is supposed to mention if the file contains a header, if the file contains a header, those header names are used, if not the numbers starting from 0 are given as the headers for the dataset.

Pre-processing Service:

This service consists of 4 modules,

 Null values: The dataset is not eligible for further processing if it contains null values or NaN values. All the algorithms do not behave as expected if datasets contain NaN values. The elimination of NaN values through the application can be done in the following ways:

 Fill Backward: The values in the previous cell would be filled

 Fill Forward: The values in the next cell will be filled in the current cell.

 Drop Null Rows: The rows containing null values will be dropped.

 Fill Most Common: The most common value will be filled in the current cell.

 Fill Mean: The mean values will be filled in the current cell

 Fill Median: The median value will be filled in the current cell.

 Fill Custom: A custom value specified by the user will be filled in the current cell.

 Label Encoding: It refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning. The application performs label encoding on the all the strings present in the current state of dataset. This process is automatic and it does not involve any user interaction for this process.

 One-Hot Encoding: A one hot encoding is a representation of categorical variables as binary vectors. This first requires that the categorical values be mapped to integer values. Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1. The user selects the columns which require one hot encoding and only those columns are one hot encoded and the columns are appended to the dataset. The column which is one hot encoded is removed from the dataset.

 Standardization: It typically means rescales data to have a mean of 0 and a standard deviation of 1 (unit variance). Standardization in the application can be performed in two ways. The standardization can be applied to each column where the variance of each row is calculated separately or the standardization is applied to the entire dataset. The user is given the option to choose the type of standardization to be applied. At this point, before standardization is applied the data would not contain any strings.

ML Processing Service

1. Algorithms

The algorithms available in the application for the users to select from are:

 Support Vector Machine (SVM): SVM models are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. The application has both types of SVM, SVMRegressor, SVMClassifier.

 Neural Networks: A neural network is a series of algorithms that endeavours to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so, the network generates the best possible result without needing to redesign the output criteria. The application supports use of Neural Networks for regression and classification i.e. MLPClassifier, MLPRegressor.

 Random Forest: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. The application supports both classification and regression in random forest i.e. RandomForestRegressor, RandomForestClassifier.

 Gaussian Naïve 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. The application supports the Naïve Bayes classifier.

 Linear Regression: Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model.

For selecting the algorithm and to train the model, the user and client undergo the following steps, the user is required to send the class of algorithm (Regression or Classification), the list algorithms under the particular class are returned and the user selects the algorithm and return to server. The server returns the list of hyperparameters for that algorithm. The user selects the hyperparameter values and returns to server for it to form the model.

2. Process flow of Model training and Evaluation:

The model training and evaluation takes place in 4 steps,

 Split data: The data is split into training and test set as specified by the user and this data is stored in the memory

 Train Model: The algorithm selected and hyperparameters are all given to training service which builds the model and trains it and stores the trained model in the memory.

 Predict using trained model: The test data is fetched from the memory and also the trained model to perform the predictions, these predictions are stored in the memory.

 Evaluate predictions: The predictions are fetched from memory and the accuracy (Classification) and RMSE(Regression) are calculated.

Prediction Service :

This service is responsible for predicting the newly uploaded dataset. The following steps are performed in this service:

 Pre-processing: The pre-processing actions performed on the train set are stored and converted as a function, this function can be used to perform the same set of pre-processing actions on the test dataset.

 Prediction: After the pre-processing actions are performed the values are predicted using the model which is stored in the memory. The predicted values are once again stored in the memory.

 Download Predictions: The predicted values which are stored in the memory can be downloaded as a csv file, the csv file will contain index values and the predicted values.

Problem Type Prediction Service:

This service is a part of the prediction layer. This service is responsible for predicting the class type of the uploaded dataset. The two types of class of algorithms available in the application are:

 Classification

 Regression

This service predicts the type of problem based on the target attributes, features of target attributes such as the if values are continuous or discreet or if they are strings. To predict the problem type unique ratio of the set is calculated. Unique ratio is the number of unique target outputs to the total number of outputs. After the unique ratio is calculated, the dataset size is checked, if the size is more the trained KNN model is used to predict the problem type of the dataset, this predictor takes the unique ratio of the dataset. The predicted problem type is displayed to the user. If the user chooses other problem type, this is added to dataset maintained for problem type prediction this is used to predict further problem types.

Null Value Handling Predictor Service:

The null value percentage is calculated for each column containing null values. If these null values are less than 4 percent the user is recommended to drop the drop the rows else other conditions are checked. If the null value percentage is greater than 70 percent the user is recommended to drop the columns as that attribute may not provide much information. If the column contains categorical value and null percentage greater than 20 percent than the fill custom option is recommended.

Algorithm Type Predictor Service:

The algorithm type service is responsible for suggesting the user the type of algorithm can be applied on the current dataset. The model runs through all the availablealgorithms for the given problem type. All the different algorithms run on different threads parallelly to avoid high time consumption. The different accuracies are compared and the best algorithm with high accuracy are chosen as the output. The calculated accuracy along with algorithm is suggested to the user, if the user wishes to he can proceed with this suggested algorithm or can choose any other algorithm from the available list.

Graph Service:

The Graph service is completely individual component of the application. Once the service is started a separate API call is made to fetch the current data into this module as the graph service does not share the data as the other services do. This service is used to perform data analysis on the current dataset so that the user has better idea about the type of algorithms, pre-processing selections the user makes on the dataset.

Currently available graphs in the application are

 Line Graph

 Box Plot

 Scatter Plot

 Bar Plot

 Correlation Plot.

The following steps are performed in the Graph Service:

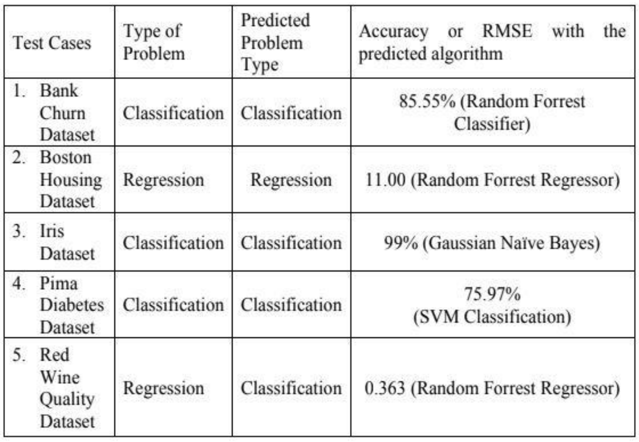
 Plot Type Selection: The user selects the type of graph to be plotted.

 Attributes Selection: The server responds with list of attributes in the dataset that can be plotted using the selected plot type from the previous step.

 Viewing the graph: After the attributes and graph type are provided the service plots the graph and return the graph in a new tab of the interface.

III. THE SUCCESS RATES

Performance Evaluation of the test cases has been presented in the form of table. The table contains a row for each data set and contains columns such as the algorithm type, predicted algorithm type, Accuracy or RMSE with the predicted algorithm, Accuracy or RMSE with any of the user selected algorithm.



Test Cases	Type of Problem	Predicted Problem Type	Accuracy or RMSE with the predicted algorithm
1. Bank Churn Dataset	Classification	Classification	85.55% (Random Forrest Classifier)
2. Boston Housing Dataset	Regression	Regression	11.00 (Random Forrest Regressor)
3. Iris Dataset	Classification	Classification	99% (Gaussian Naïve Bayes)
4. Pima Diabetes Dataset	Classification	Classification	75.97% (SVM Classification)
5. Red Wine Quality Dataset	Regression	Classification	0.363 (Random Forrest Regressor)

IV. CONCLUSIONS

This project was chosen because importance of ML in every sector is growing every day, hence more people are investing time to learn it. Computer scientists are hired specifically for ML in various other sectors. So, to help people who are not computer scientist or to help companies by reducing hiring of professionals this application has been developed.

This application has mainly five objectives which have been achieved in the following ways,

 Providing a completely automated tool for user, this hasbeen implemented using a Web User interface where the user simply clicks and selects and does not have to code.

 Incorporating mechanism for pre-processing, this has been implemented by providing means to label encode, one-hot encode, remove null values and standardizing the data.

 Enabling the user to plot graphs, this has been implemented through the graph service which lets user select attributes and see their graphical visualisation.

 Suggesting ML algorithm, this has been implemented by the Prediction layer service which predicts for the user the type of problem and also the best algorithm for the problem.

 Displaying the accuracy, the application displays accuracy or RMSE for the problem given by the user as the final result.

Acknowledgment

We would like to thank Shri. Narayan Rao R. Maanay, Secretary, BNMIT, Bengaluru for providing excellent academic environment in the college. We would like to sincerely thank Prof. T. J. Rama MurthyDirector, BNMIT, Bengaluru for having extended his support and encouraging us during the course of the work. We would like to express our gratitude to Prof. Eishwar N Maanay, Dean, BNMIT, Bengaluru for his relentless support, guidance and assistance.

We would like to thank Dr. Krishnamurthy G.N, Principal, BNMIT, Bengaluru for his constant encouragement.

We would like to thank Dr. Sahana D Gowda, Professor, Head, Department of Computer Science and Engineering, who has shared her opinions and thoughts which helped us in the completion of our Project successfully. We consider it is a privilege to express gratitude to our guide Smt. Jalaja G, Associate Professor, BNMIT who has given us all the support and guidance in completing the Project successfully.

References

[1] Rosas–Arias Leonel, Sanchez–Perez Gabriel, Toscano–Medina Linda K.,Perez–Meana,Hector M., and Portillo–Portillo Jose, "A Graphical User Interface for Fast Evaluation and Testing of Machine Learning Models Performance", IEEE, 2019.
[2] Matthias Feurer,Aaron Klein,Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum ,Frank Hutter," Efficient and Robust Automated Machine Learning" ,NIPS,2015.
[3] Felix Mohr, Marcel Wever,Eyke Hüllermeier," ML-Plan: Automated machine learning via hierarchical planning",Springer,2017.
[4] Amanpreet Singh, Narina Thakur, Aakanksha Sharma," A Review of Supervised Machine Learning Algorithms",IEEE,2016.
[5] Patrick Vogel, Thijs Klooster, Vasilios Andrikopoulos, Mircea Lungu," A Low-Effort Analytics Platform for Visualizing Evolving Flask-Based Python Web Services",IEEE,2017.
[6] Nur Atikah Shamat, Shahida Sulaiman and Jacline Sudah Sinpang," A Systematic Literature Review on User Interface Design for Web Applications",JTEC,2017.
[7] Suad A. Alasadi and Wesam S. Bhaya, "Review of data preprocessing techniques in Data Mining", Journal of Engineering and applied sciences, ResearchGate, 2017.
[8] Bin Wei, Minqing Zang, Longfei Liu and Jing Zhao, "Feature Selection on the Basis of Rough Set Theory and Univariate Marginal Distribution Algorithm", International Conference on Applied Mathematics, Modelling and Statistics Application, 2017.
[9] Scikit Library, scikit-learn.org/
[10] Kaggle for Datasets, (Kaggle.com/datasets)
[11]Dataset Preprocessing techniques, (towardsdatascience.com/data-pre-processing-techniques-you-should-know)
[12]API Gateway Design Pattern, (microservices.io/patterns/apigateway)
[13] Flask Documentation,(flask.palletsprojects.com/en/1.1.x/quickstart/)
[14] Graph Visualization with Bokeh (bokeh.org/)
[15] Bootstrap Designing (getbootstrap.com/)
[16] G. Holmes; A. Donkin; I.H. Witten (1994). "Weka: A machine learning workbench"(PDF). Proc Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia. Retrieved 2007-06-25.