**Freelancing Customer Segmentation**

# Low level Design Document

**Version 1.0**

RESTRICTED DISTRIBUTION

The information is standard Company Confidential, but due to its sensitivity, it has restricted distribution and viewing within iNeuron.

Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Date Issued** | **Version** | **Description** | **Author** |
|  | **1.0** | Initial Draft |  |
|  | **2.0** |  |  |

Contributors

The content of this document has been authored with the combined input of the following group of key individuals.

|  |  |
| --- | --- |
| **Name** | **Section Worked Upon** |
| Anoop Parashar |  |
| Robin Trakroo |  |
| Akshay Rahate |  |
| Nisikanta Dash |  |
| Shubhangi Shinde |  |
| Aditya Tandon |  |

Document Classification

|  |  |
| --- | --- |
| **Classification** | Company Confidential |
| **Definition** | Information is Group confidential and needs to be protected |
| **Context** | Where the loss of information confidentiality would result in significant harm to the interests of the organisation, financial loss, embarrassment or loss of information |

Contents

[1 Technical Design Document 2](#_Toc55920495)

[1. Introduction 4](#_Toc55920496)

[High level objectives 5](#_Toc55920497)

[2 Workflow Overall 6](#_Toc55920498)

[Application Flow 6](#_Toc55920499)

[Exception Scenarios Overall 8](#_Toc55920500)

# Introduction

The goal here is to build an end to end automated Machine Learning solution for customer segmentation where the user will only give the data and the result will be the best performing hyper tuned Machine Learning model.

This project shall be delivered in two phases:

Phase1: Integration of UI to all the functionalities.

The High level design document gives a design blueprint of the freelancing customer segmentation project. This document communicates the technical details of the solution proposed.

In addition, this document also captures the different workflows involved to build the solution, exceptions in the workflows and any assumptions that have been considered.

Once agreed as the basis for the building of the project, the flowchart and assumptions will be used as a platform from which the solution will be designed.

Changes to this business process may constitute a request for change and will be subject to the agreed agility program change procedures.

**Note: All the code will be written in python version 3.8**

## Low level objectives

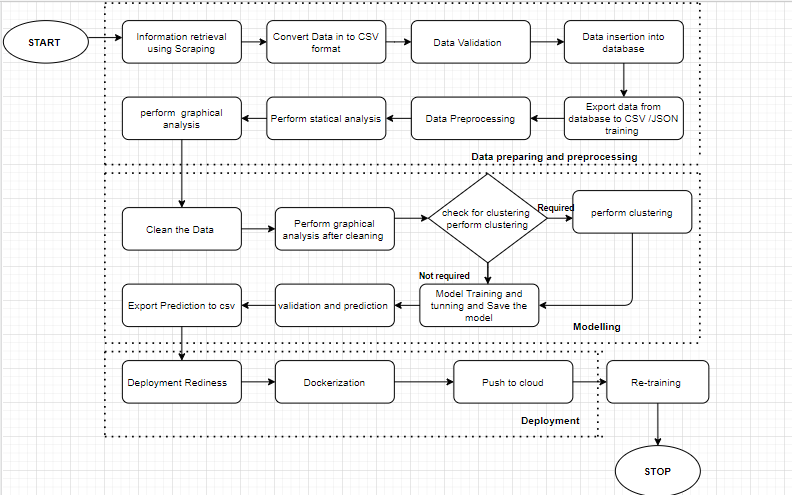
The Low-level objectives are:

1. Enable reading/loading of data from the various sources and convert them into panda’s data frame (details mentioned in the Data Ingestion Section).
2. Enable reading various file formats and convert them into panda’s data frame (details mentioned in the Data Ingestion Section).
3. Give user the filter option to specify feature and target columns.
4. Perform statistical analytics of the data and prepare a table for the analysis
5. Perform graphical analysis for the data and Showcase the results (graphs) on the screen.
6. Perform data cleaning operation with all the steps required and showcase a report on screen.
7. After data cleaning showcase the graphical analysis once again for comparison.
8. Check whether clustering is required or not.
9. Choose the appropriate ML model for training.
10. Perform model Tuning.
11. Create a list of top 3 models and show multiple metrics for them.
12. Give option for prediction.
13. Give options for docker container creation.
14. Give option for automatic cloud deployment.

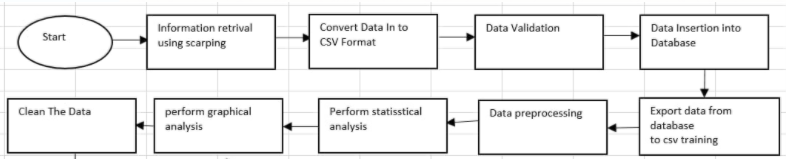
**Phase 1:** Create UI

# Workflow Overall

# Application Flow



Data preparing and Pre-processing:



1. **Information Retrieval using Scrapping**: This is an initial step which is crucial to collect the data. As a part of this module, we need to collect the data based on identified features w.r.t employer, employees, and Job description. We have identified these features based on Competitor mode of operation, Competitors Business Research, Competitors Leaders Research, Information source analysis.
2. **Convert Data into CSV format:** The information which we have scrapped we have converted that data into CSV format.
3. **Store data into tabular format:** As a next step, we need to fetch the data from the csv files and store the relative data into following tables.
   1. Employers\_Information\_tab – This table consists the information about employers, which are responsible to posting the job’s into the website.
   2. Employees\_Information\_tab – This table consists the information about employee’s information, which are applying for the jobs.
   3. Job\_description\_tab- This table consists the information about job information based on technology, location etc.
4. **Data Pre-processing: -** In this step, application will fetch the data from database and perform the data pre-processing techniques (cleaning and organizing), perform statistical analysis to identify the relationship between data and categorized in to define categories. This data is suitable for a building and training Machine Learning models.

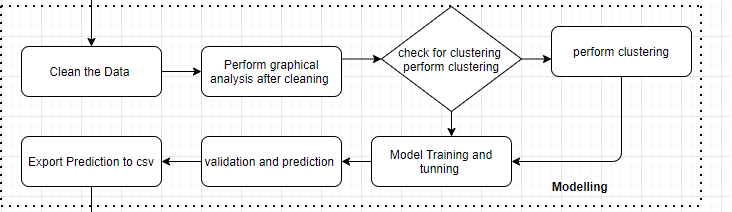
**Technology stack-**

* **Python -** The three core Python libraries used for this data set Numpy , Pandas, Matplotlib
* **Java script**
* **Graph database**
* **Database**

**Modelling:**

Feature engineering improves the quality of performance of machine learning algorithms. It is a process of using domain knowledge to extract features from row data via data mining technique. By scraping the data, we extracted some of the feature which is important to build a good model.

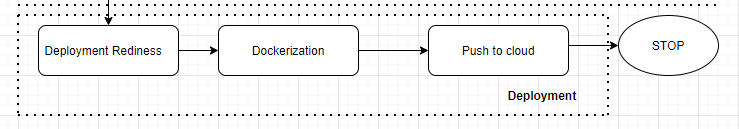
**Model Training**:



A model is a file that has been trained to recognize certain types of patterns. You train a model over a set of data, providing it an algorithm that it can use to reason over and learn from those data.

* K-Means Clustering
* Density-Based Spatial Clustering of Applications with Noise
* Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)
* Agglomerative Hierarchical Clustering
* OPTICS Clustering
* Affinity Propagation
* Birch Clustering

**Deployment**:



Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data.

Some deployment platform Google Cloud Platform (GCP) is one of the primary options for cloud-based deployment of ML models, along with others such as AWS, Microsoft Azure,etc

- Training a machine learning model on a local system.

- Wrapping the inference logic into a flask application.

- Using docker to containerize the flask application.

- Hosting the docker container on an AWS ec2 instance and consuming the web-service.

## Exception Scenarios Overall

|  |  |  |
| --- | --- | --- |
| **Step** | **Exception** | **Mitigation** |
| User gives Wrong Data Source | Give proper error message | Ask the user to re-enter the details |
| User gives corrupted data | Give proper error message |  |
| If the cluster contains only one class | No error message required | Handle this exception internally. User doesn’t know. |
| Deployment credentials are wrong | Give proper error message | Ask for the details to be entered again |

# 

# **Workflow Data Ingestion and File** Conversion

**Data Sources:**

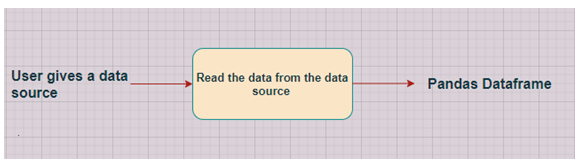
|  |  |
| --- | --- |
| Data Connector Utils | File Conversion Utils |
| CSV | CSV |

The input data file provided is of .CSV format (comma separated values) but with the delimiter being a semi-colon. We first manually replace the semi-colons with commas and prepare our input file.

we can read it and store the data in a Data Frame for further processing for both freelancer and job segmentation.

Freelancer:

## Technical solution design



|  |  |  |
| --- | --- | --- |
| **Class Name** | **Data Getter** |  |
| **Method Name** | read\_data\_from\_asc |  |
|  | Method Description | This method will be used to read data from a asc file or a flat file |
|  | Input parameter names | File / select features |
|  | Input Parameter Description | file: csv file containing the input parameters  form: user gives input on website in a form selecting appropriate values. |
|  | output | A pandas Dataframe |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |
| **Method Name** | **Connect\_to\_SQLite DB** |  |
|  | Method Description | This method will be used to connect to a SQLite Databases |
|  | Input parameter names | self,host,port, username, password |
|  | Input Parameter Description | host: the server hostname/IP where the DB server is hosted  Port: the port at which the DB Server is running  username: The username to connect to the DB server  password: The password to connect to the DB server |
|  | output | A DB connection object |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |
| **Method Name** | **read\_data\_from\_SQLite DB** |  |
|  | Method Description | This method will be used to read data from SQLite Databases |
|  | Input parameter names | self,db\_name,host,port, username, password, schema\_name, query\_string |
|  | Input Parameter Description | db\_name: For example,  SQLite  host: the server hostname/IP where the DB server is hosted  Port: the port at which the DB Server is running  username: The username to connect to the DB server  password: The password to connect to the DB server  schema\_name: The name of the DB schema the user wants to connect to.  query\_string: the query to be executed to load the data |
|  | output | A Pandas Dataframe |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |
|  | output | A Pandas Dataframe |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |

## Exceptions Scenarios

|  |  |  |
| --- | --- | --- |
| **Step** | **Exception** | **Mitigation** |
| User gives Wrong Data Source | Give proper error message | Ask the user to re-enter the details |
| User gives corrupted data | Give proper error message |  |

# **Data Profiling**

**Data profiling** is the process of examining the data available from an existing information source (e.g. a database or a file) and collecting statistics or informative summaries about that data.

After reading the data, automatically the following details should be shown:

1. The number of rows
2. The number of columns
3. Number of missing values per column and their percentage
4. Total missing values and it’s percentage
5. Number of categorical columns and their list
6. Number of numerical columns and their list
7. Number of duplicate rows
8. Number of columns with zero standard deviation and their list
9. Size occupied in RAM

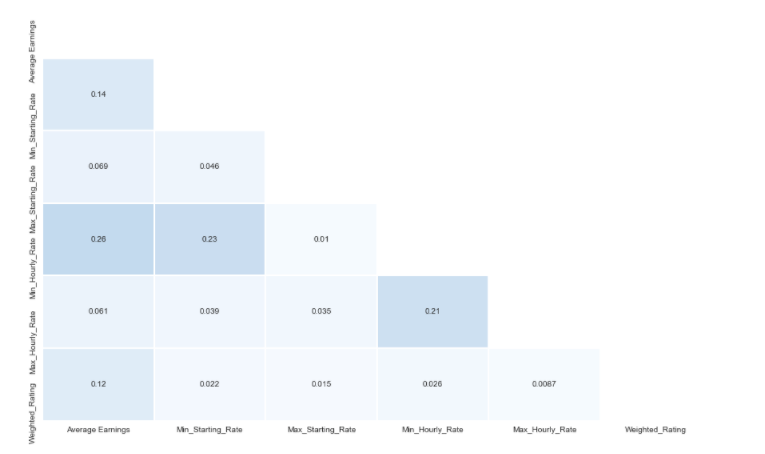
## **Method Definition**

|  |  |  |
| --- | --- | --- |
| **Class Name** | **DataProfiler** |  |
| Method Name | get\_data\_profile |  |
|  | Method Description | This method will be used to give various insighst about data. |
|  | Input parameter names | self, dataframe |
|  | Input Parameter Description | dataframe: the inpt data just loaded from source |
|  | ouptput | 1. The number of rows 2. The number of columns 3. Number of missing values per column and their percentage 4. Total missing values and it’s percentage 5. Number of categorical columns and their list 6. Number of numerical columns and their list 7. Number of duplicate rows 8. Number of columns with zero standard deviation and their list 9. Size occupied in RAM |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |

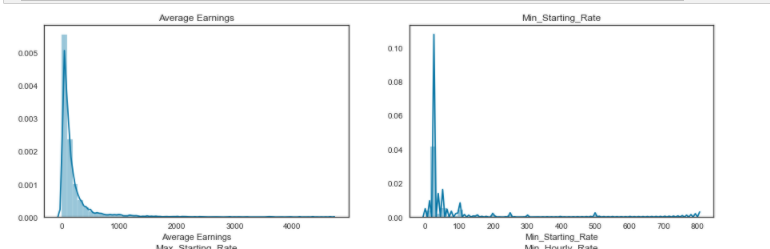
# 

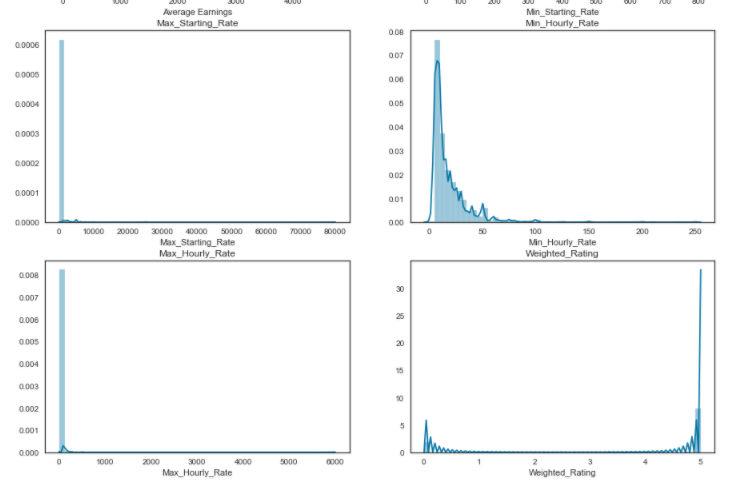
# **Graph Based EDA**

1. Checking Correlation

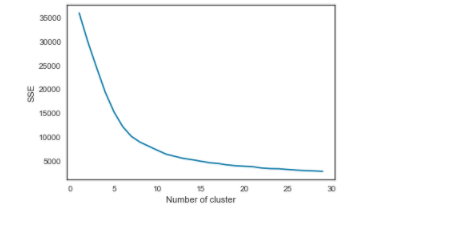


1. Checking the skewness





### Scaling & Normalize Dataset



**6 . Library Based Utils**

**Required Libraries:**

* **NumPy**
* **Pandas**
* **Scikit-Learn**
* **SqLite3**
* **Flask**
* **pycaret**
* **Joblib**

**7. Data Transformers (Pre-processing steps**)

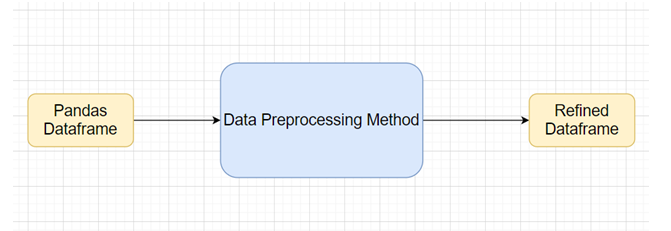
1. Converting Columns in the list
2. Checking for Null Values
3. Checking Data Type
4. Handling columns with standard deviation zero or below threshold
5. Correcting the Skewness
6. Handling the Outliers
7. Scaling and Normalize dataset

**Data pre-processing for JOB Segmentation**

Data pre-processing handles the basic processing of data before we begin the statistical and graphical analysis.

If our data has null values, we replace them with knn imputer of the respective feature. We then separate categorical and numerical features as further processing involves specific methods pertaining to type of feature.

## 7.1 Technical solution design



## 

## 7.2 Method Definitions

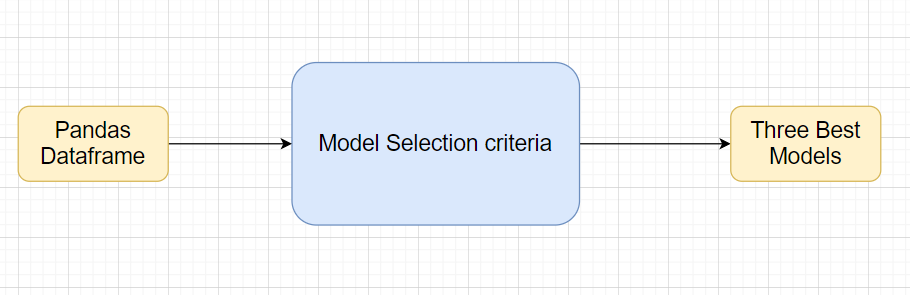
## 

|  |  |  |
| --- | --- | --- |
| **Class Name** | **DataPreprocessor** |  |
| Method Name | impute\_missing\_values |  |
|  | Method Description | This method will be used to read data from a csv file or a flat file |
|  | Input parameter names | self,file\_name, header,names, use\_cols, separator |
|  | Input Parameter Description | file\_name: name of the file to be read  header: Row number(s) to be used as column names  names : array-like, optional  List of column names to use. If file contains no header row, then you  should explicitly pass ``header=None``.  Use\_cols: To load a subset of columns  Separator: Delimiter to use |
|  | ouptput | A pandas Dataframe |
|  | On Exception | Write the exception in the log file.  Raise an exception with the appropriate error message |

## Exceptions Scenarios Module Wise

|  |  |  |
| --- | --- | --- |
| **Step** | **Exception** | **Mitigation** |
| Wrong parameters passed to the methods | Handle Internally | Code should never give a wrong input |

# **ML Model Selection**

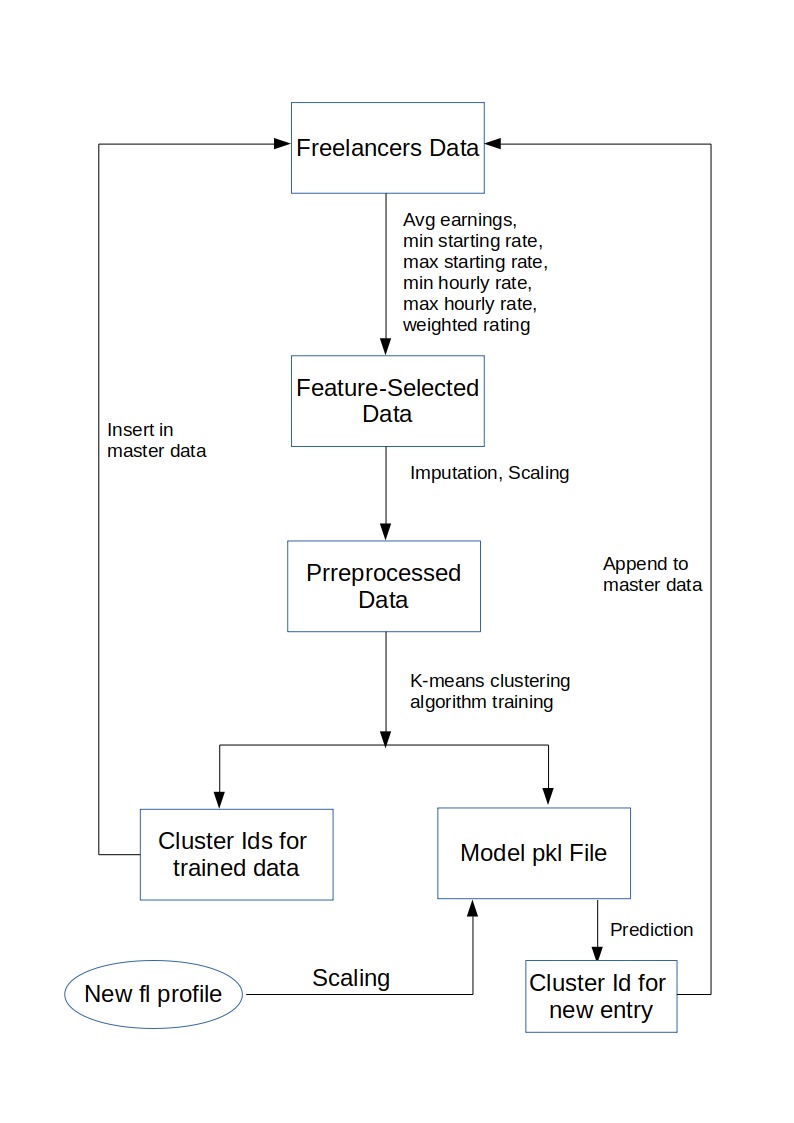


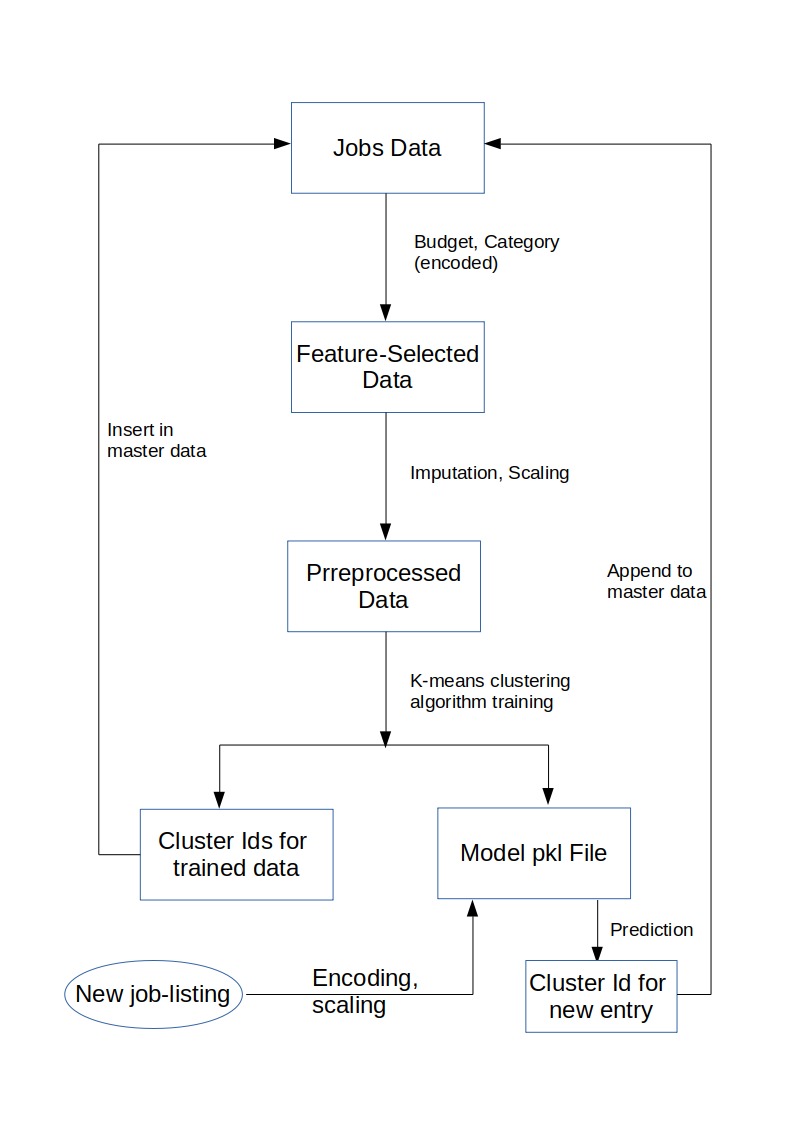
* K Means Clustering
* Agglomerative Clustering
* DBSCAN

Clustering is the task of grouping a set of objects in such a way that those in the same group (called a cluster) are more similar to each other than to those in other groups. It is an exploratory data mining activity, and a common technique for statistical data analysis used in many fields including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression and computer graphics. Some common real life use cases of clustering are:

* Customer segmentation based on purchase history or interests to design targeted marketing campaigns.
* Cluster documents into multiple categories based on tags, topics, and the content of the document.
* Analysis of outcome in social / life science experiments to find natural groupings and patterns in the data.

We have verified with 3 Models for job segmentation—k means, Dbscan,Mean shift. Kmeans algorithm provide us optimal result.





# **Testing Modules**

Divide the training data itself into train and test sets.

Use test data to have tests run on the three best models.

Give the test report.

1. R2 Score
2. Adjusted R2 score
3. MSE
4. Accuracy
5. Precision
6. Recall
7. F Beta
8. Cluster Purity
9. Silhouette score

Model selection criteria:

we have choose Silhouette score for finding optimise model.

**Phase 2**

AIC

BIC

**Note**: Save the best model after validation is completed.

# **Prediction Pipeline**

Use the existing data read modules

Use the existing pre-processing module

Load the model into memory

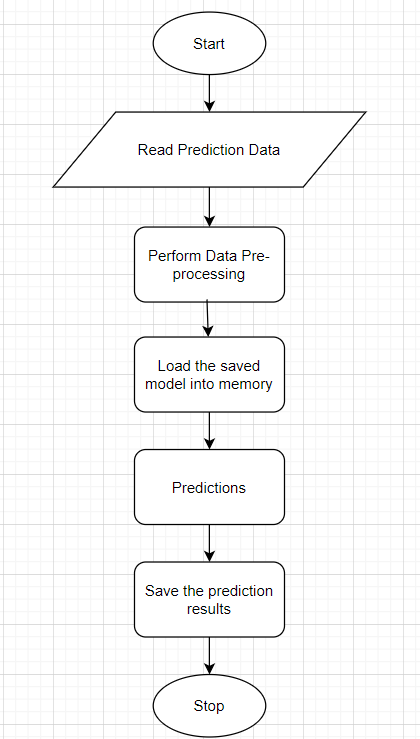
Do predictions

Store prediction results(show sample predictions)

Phase 2:

UI for predictions

## 10.1.Technical solution design



## Exceptions Scenarios Module Wise

|  |  |  |
| --- | --- | --- |
| **Step** | **Exception** | **Mitigation** |
| Columns don’t match in training and Prediction data | Show error message | The user enters the correct data |
|  |  |  |

# **Logging**

Separate Folder for logs

Logging of every step

Entry to the methods

Exit from the methods with success/ failure message

Error message Logging

Model comparisons

Training start and end

Prediction start and end

Achieve asynchronous logging

**Phase 2:**

Options for Logging in DB

Options for Log Publish

## Technical solution design



# **Hardware Requirements**

## Requirements for model training

The minimum configuration should be:

* 8 GB RAM
* 2 GB of Hard Disk
* Intel Core i5 Processor

## Requirements for model testing

The minimum configuration should be:

* GB RAM
* GB of Hard Disk Space
* Intel Core i5 Processor

**13. Self-Reflection**

14.1 What we Learned

* Learnt the basics of html and css to make the UI
* Learnt to build Docker container and deployment.
* Learnt about the flask monitoring Dashboard
* Management
* Learnt to write production level code
* Learnt working in team with different and unknown team members

14.2 Hurdles faced

* Creating a clean and modular level code
* Managing time
* Writing Documentation