

Ensemble approach on customer churn prediction

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

POREDDY AKSHITHA REDDY
(Reg. No. 38110411)
RAGHAVARAJU SRAVANI
(Reg No.38110439)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119 TAMILNADU.
MARCH - 2022**



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Raghavaraju Sravani (38110439), P. Akshitha Reddy (38110411) who have done the Project work as a team who carried out the project entitled " Ensemble approach on customer churn prediction " under my supervision from Dr. Meera Gandhi . G

Internal Guide

Dr. MEERA GANDHI.G, ME., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I Raghavaraju sravani and P. Akshitha reddy (Name/s of the individual Candidate) hereby declare that the Project Report entitled Ensemble approach on customer shopping churn prediction done by me under the guidance of Dr.Meera Gandhi . G(Internal) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Computer science engineering .

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

We convey our thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing **Dr. L. Lakshmanan M.E., Ph.D.** , and **Dr.S.Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

We convey our thanks to **Dr. S. VIGNESHWARI, M.E., Ph.D.**, Head of Department of Computer Science for providing us necessary support and details at right time during the progressive views.

We would like to express our sincere and deep sense of gratitude to my Project Guide **Dr. MEERA GANDHI.G, ME., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Online shopping grows along with the growing population. An ensemble approach has been drawn for better shopping churn prediction. The algorithms used are KNN, Stacking, Random forest, XGBoost, and Logistic Regression. An accurate prediction of (90.65%) has been achieved for our ensemble approach as the best result. Customer churn refers to the number of customers who have ceased utilizing the company's product or service over a period of time.

The number of customers lost within a certain time frame divided by the number of active customers at the start of the period is one way to compute a churn rate. For example, if you gained 1000 clients last month but lost 50, the monthly churn rate would be 5%. Every month, the active customer base is fed into a Machine Learning Predictive Model, which calculates the likelihood of each client churning, will be sorted from highest to lowest probability value (or score. Clients with a low likelihood of turnover (or, in other words, customers for whom the model forecasts no churn) are satisfied customers.

TABLE OF CONTENTS

CHAPTER	CHAPTER NAME	PAGE NUMBER
	Abstract	iv
	List of Figures	vii
	List of Abbreviations	vii
1	INTRODUCTION	01
	1.1 OUTLINE OF PROJECT	01
	1.2 PURPOSE OF THE PROJECT	01
	1.3 RELATED WORK	02
	1.4 SCOPE OF PROJECT	03
	1.5 PROJECT OVERVIEW	03
	1.6 TECHNIQUES USED?	04
	1.7 SYSTEM REQUIREMENTS	05
2	LITERATURE SURVEY	06
	2.1 DIFFERENT SURVEYS FR ONLINE PREDICTION	06
3	METHODOLOGY	09
	3.1 EXISTING METHODOLOGY	09
	3.2 PROPOSED METHODOLOGY	09
	3.3 SYSTEM ARCHITECTURE	10
	3.3.1 TAKING INPUT DATA	11
	3.3.2 DATA PREPROCESSING AND ACQUISITION	13
	3.3.3 VALIDATION OF DATASET	17
	3.3.4 USING DIFFERENT ML ALGORITHMS	19
	3.3.4.1 KNN	19
	3.3.4.2 RANDOM FOREST	21
	3.3.4.3 XGBOOST	24

	3.3.4.4 LOGISTIC REGRESSION	26
	3.3.4.5 STACKING	29
	3.3.5 TEST OF DATASET	32
	3.3.6 METRICS USED	32
	3.4 WORKFLOW -DIAGRAM	36
4	RESULTS AND DISSCUSSIONS	37
	4.1 RESULTS	37
	4.2 REQUIRED ANALYSIS	38
5	CONCLUSION AND FUTURE WORK	39
	5.1 CONCLUSION	39
	5.2 SCOPE OF FUTURE WORK	39
	REFERENCES	40
	APPENDICIES	
	A. SAMPLE CODE	41
	B. SCREENSHOTS	
	C. PLAGIARISM REPORT	

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO.
3.3.1	System Architecture	10
3.3.2.1	Data pre-processing and Acquisition	13
3.3.4.1.1	KNN Processing	19
3.3.4.2.1	Random Forest Processing	22
3.3.4.3.1	XGBoost Processing	24
3.3.4.4.1	Logistic Regression Processing	27
3.3.4.5.1	Stacking Process	30
3.3.4.5.2	Model training of stacking	31
3.3.6.1	Classification Graph	33
3.4.1	Work-Flow Diagram	36

LIST OF TABLES

TABLE NO	TITLE	PAGE NO.
4.1	RESULT TABLE	37

LIST OF GRAPHS

GRAPH NO	TITLE	PAGE NO.
4.1	TITLE COMPARSION FOR PERFORMANCE METRICS	38

LIST OF ABBRIVATIONS

- 1 KNN - K NEAREST NEIGHBOURS
- 2 XGBOOST- XTREME GRADIENT BOOSTING
- 3 TP – TRUE POSITIVE
- 4 TN - TRUE NEGATIVE
- 5 FP - FALSE POSITIVE
- 6 FN – FALSE NEGATIVE

CHAPTER 1

INTRODUCTION

1.1 OUTLINE OF PROJECT

Online Shopping is continuously growing in India. People depend on online shopping since it is easier time saving and effortless. Getting the menu at our fingertips makes it a lot time saving. An online shopping system that permits a customer to submit online orders for items and/or services from a store that serves both walk-in customers and online customers.

Today, web-based shopping is the quickest and internet promoting tool. The is vital when shopping. quantity of engineers who purchase their items on the web with the nature of online administrations and items on the web. In this way, the internet shopping climate assumes a significant part in the connection among shippers and purchasers.

ML Algorithms give incredible assets to unraveling huge, uproarious information. Machine preparing permits analysts likely comprehend buyer conduct and work on the presentation of promoting efforts. We have attempted to change the fulfillment of shoppers who shop on the web/online, also attempted to make a perfect framework for deciding shopper purchasing conduct and online store fulfillment levels. The main goal of this work is to use ML algorithmsfor better recommendations.

1.2 PURPOSE OF PROJECT

Accurate online prediction for a product buying or not involves expert knowledge, because price usually depends on many distinctive features and factors. Typically, most significant ones are brand, Expiry date, Tenure, Membership and Average Salary of person. The main objective of the Online Shopping System is to manage the details of Shopping, Internet, Payment, Bills, Customer. It manages all the

information about Shopping, Products, Customer, Shopping. The project is totally built at administrative end and thus only the administrator is guaranteed the access based on the products purchased.

A typical online store enables the customer to browse the firm's range of products and services, view photos or images of the products, along with information about the product specifications, features and prices.

Online stores usually enable shoppers to use "search" features to find specific models, brands or items. Online customers must have access to the Internet and a valid method of payment in order to complete a transaction, such as a credit card, debit card, or a service such as PayPal. For physical products (e.g., paperback books or clothes), the e-tailer ships the products to the customer; for digital products, such as digital audio files of songs or software, the e-tailer usually sends the file to the customer over the Internet.

People in large number are doing online shopping today, and it is not only because it is convenient as one can shop from home, but also because there is an ample number of varieties available, with a high competition of prices, and also it is easy to navigate for searching regarding any particular item.

1.3 RELATED WORK

Online shopping is a form of electronic commerce which allows consumers to directly buy goods or services from a seller over the Internet using a web browser or a mobile app. Consumers find a product of interest by visiting the website of the retailer directly or by searching among alternative vendors using a shopping search engine, which displays the same product's availability and pricing at different e-retailers.

K Maheshwari,P Packia Amutha Priya has discussed, in their paper in 2018 written for Master thesis [2], that regression model that was built using Support Vector Machines (SVM). Customer buying behaviour is identified by people's personality and character. These personality characters vary from person to person. The character includes quality, motivation, occupation and income level, perception,

psychological, personality, reference groups and demographic reasons learning, beliefs, attitude, Culture and social forces. Nowadays, Data mining normally used to investigate the customer activities on shopping by using various algorithms and methods. Data mining has gradually raised and it gains numerous industries which applies this technology. Each and every activity of a customer is stored as a byte of data in a database to collect information such as how the customer spends their valuable time, day in buying decision.

Another approach by Cheng Lin in 2020 was we studied the size prediction problem and proposed a method to improve the predication accuracy. We capture the subtle semantics of customer reviews by using Bert model to learn the latent representations. Deep features are learned from the review data, combining with customers' body features and the product's information to predict the product's fitness for a customer. The experiments on the real online clothing retailer dataset validate the effectiveness of the proposed method.

1.4 SCOPE OF PROJECT

This Online prediction system can be implemented to any shop in the locality or to multinational branded shops having retail outlet chains. The system recommends a facility to accept the orders 24*7 and a home delivery system which can make customers happy.

If shops are providing an online portal where their customers can enjoy easy shopping from anywhere, the shops won't be losing any more customers to the trending online shops such as flipcart or ebay. Since the application is available in the Smartphone it is easily accessible and always available.

1.5 PROJECT OVERVIEW

The Online Customer churn prediction form predicts whether the customer buys product from the respective website or not based on feedback or any other factors of product. The central concept of the application is to allow the customer to shop virtually using the Internet and allow customers to buy the items and articles of their desire from the store. The information pertaining to the products are stores on an RDBMS at the server side (store).

The end user of this product is a departmental store where the application is hosted on the web and the administrator maintains the database. The application which is deployed at the customer database, the details of the items are brought forward from the database for the customer view based on the selection through the menu and the database of all the products are updated at the end of each transaction. Data entry into the application can be done through various screens designed for various levels of users. Once the authorized personnel feed the relevant data into the system, several reports could be generated as per the security.

The Server process the customer to the demo page to enter the details i.e, age ,membership, has any offers, coupons , time spent on website etc things to fill it and predict the score whether they are willing to buy the product or not based on the churn score we get on the screen.

1.6 TECHNIQUES USED?

This project is an attempt to provide the advantages of online shopping to customers of a real shop. It helps buying the products in the shop anywhere through internet by using an android device by showing there the Front-End Development connecting to the backend server to store the information using Flask etc. Thus the customer will get the service of online shopping and home delivery from his favorite shop based on the tracker of the address and and may buy the product depending on the feedback of other customers. This project is a web-based shopping system for an existing shop. The project objective is to deliver the online shopping application into android platform.

Supervised machine learning is a technique whose task is to deduce a function from tagged training samples. The training samples for supervised learning consist of large set of examples for a particular topic. In supervised learning, every example training data comes in a pair of input (vector quantity) and output value (desired result). These algorithms analyze data and generate an output function, which is used to mapped new data sets to respective classes. Different algorithms used in prediction of this analysis are:-

- 1.KNN (K Nearest Neighbor)
- 2.Random Forest
- 3.XGBoost (Xtreme Gradient Boosting)
- 4.Logistic Regression
- 5.Stacking (Ensemble Method)

1.7 SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Monitor : 15"LED
- Input Devices : Keyboard, Mouse
- Ram : 1GB.

SOFTWARE REQUIREMENTS

- Operating system : Windows 7.
- Coding Language : python
- Toolkit : Jupiter Notebook
- DATABASE : EXCEL

CHAPTER 2

LITERATURE SURVEY

2.1 DIFFERENT SURVEYS FOR ONLINE PREDICTION

[1]Theresa Maria Rausch, Nicholas Daniel Derra, Lukas Wolf in the year 2020 Predicting online shopping cart abandonment with machine learning approaches was proposed by using the methodology of k-nearest neighbour algorithm, Decision Tree And Boosting Algorithm proposing that Thereby, we provide methodological insights to gather a comprehensive understanding of the practicability of classification methods in the context of online shopping cart abandonment prediction: our findings indicate that gradient boosting with regularization outperforms the remaining models yielding an F1-Score of 0.8569 and an AUC value of 0.8182. Nevertheless, as gradient boosting tends to be computationally infeasible, a decision tree or boosted logistic regression may be suitable alternatives, balancing the trade-off between model complexity and prediction accuracy.

[2] Cheng Lin; Qinpei Zhao; Jiangfeng Li; Weixiong Rao in the year 2020 has proposed that Size Prediction for Online Clothing Shopping with Heterogeneous Information by a methodology of Naïve Bayes Algorithm by saying in their paper, we studied the size prediction problem and proposed a method to improve the predication accuracy. We capture the subtle semantics of customer reviews by using Bert model to learn the latent representations. Deep features are learned from the review data, combining with customers' body features and the product's information to predict the product's fitness for a customer. The experiments on the real online clothing retailer dataset validate the effectiveness of the proposed method.

[3] Yuanbang Liang; Yunyu Jia; Jinglin Li; Meiyi Chen; Yifan Hu; Yinan Shi; Fei Ma in the year 2020 has proposed that Online Shop Daily Sale Prediction Using Adaptive Network-Based Fuzzy Inference System by saying that This paper proposes a method for online shop daily sale forecasting. In the method, Kalman Filter was firstly applied on the historic sale data to smooth the data. Adaptive Network-based Fuzzy Inference System (ANFIS) was then built to achieve time series forecasting. Sale histories of an online shop was used to evaluate the method. Overall performance of

ANFIS was evaluated with MAPE and ACC($\pm$), and were found to be 1.0725 and 40%, respectively. The results were found to be slightly better than Back Propagation Neural Network which has MAPE 4.9891 and ACC($\pm$) 39.85%. 200 products were divided into 6 groups based on corresponding MAPE and ACC($\pm$). The results showed that daily sales of 22.86% products were predicted accurately, while that of 34.29% products were badly predicted. The results also indicated that membership function inserted in the model causes little impact on prediction accuracy. It was tested that the average performance indicator ACC($\pm$) (%) stays stable with different membership functions.

[4] Zui Zhang, Hua Lin, Kun Liu, Dianshuang Wu, Guangquan Zhang, Jie Lu in the year of 2019 has proposed that A hybrid fuzzy-based personalized recommender system for telecom products/services by saying that This study develops a hybrid recommendation approach which combines user-based and item-based collaborative filtering techniques with fuzzy set techniques and applies it to mobile product and service recommendation. It particularly implements the proposed approach in an intelligent recommender system software called Fuzzy-based Telecom Product Recommender System (FTCP-RS). The major limitations of CF methods are the cold start problem for new users and new items, the sparsity problem.

[5] Gopalakrishnan T.; Ritesh Choudhary; Sarada Prasad in the year of 2019 has proposed that Prediction of Sales Value in Online shopping using Linear Regression by saying that the aim of this paper is to analyze the sales of a big superstore, and predict their future sales for helping them to increase their profits and make their brand even better and competitive as per the market trends by generating customer satisfaction as well. The technique used for prediction of sales is the Linear Regression Algorithm, which is a famous algorithm in the field of Machine Learning.

[6] K. Maheswari; P. Packia Amutha Priya in the year 2018 has proposed that Predicting customer behavior in online shopping using SVM classifier by saying that Customer buying behavior is identified by people's personality and character. These personality characters vary from person to person. The character includes quality,

motivation, occupation and income level, perception, psychological, personality, reference groups and demographic reasons learning, beliefs, attitude, Culture and social forces. Nowadays, Data mining normally used to investigate the customer activities on shopping by using various algorithms and methods. Data mining has gradually raised and it gains numerous industries which applies this technology. Each and every activity of a customer is stored as a byte of data in a database to collect information such as how the customer spends their valuable time, day in buying decision.

[7] Jesus Serrano-Guerrero a, Enrique Herrera-Viedma b , Jose A. Olivas a , Andres Cerezo a , Francisco P. Romero in the year 2018 has proposed that A google wave-based fuzzy recommender system to disseminate information in University Digital Libraries 2.0 by saying that Nowadays Digital Libraries 2.0 are mainly based on the interaction between users through collaborative applications such as wikis, blogs, etc. or new possible paradigms like the waves proposed by Google. In this case a fuzzy linguistic recommender system based on the Google Wave capabilities is proposed as tool for communicating researchers interested in common research lines. The system allows the creation of a common space by means a wave as a way of collaborating and exchanging ideas between several researchers interested in the same topic. One of the main disadvantages of the proposed approach is its complexity.

[8] Zhoubao Sun¹ , Lixin Han¹ , Wenliang Huang¹ , Xueting Wang¹ , Xiaoqin Zeng¹ , Min Wang¹ and Hong Yan in the year 2018 has proposed that Recommender systems based on social networks by saying that in their paper, we propose a social regularization approach that incorporates social network information to benefit recommender systems. Both users' friendships and rating records (tags) are employed to predict the missing values (tags) in the user-item matrix. Especially, we use a biclustering algorithm to identify the most suitable group of friends for generating different final recommendations. Empirical analyses on real datasets show that the proposed approach achieves superior performance to existing approaches. With the limitation of these methods have several inherent limitations and

weaknesses that need to be solved. Another limitation is that two different items are indistinguishable if they are represented by the same tags.

CHAPTER 3

METHODOLOGY

3.1 EXISTING METHODOLOGY

In this current method when the customer chooses to buy a product there aren't proper and efficient prediction given as his needs or requirements. E-commerce functions or features that are commonly listed in the literature include product presentation, electronic catalogs, order entry, order confirmation, electronic payment, order fulfillment, and customer service. In order to Utilize the items and different choices to get what you are utilizing. Use BG/NBD-based NBD to apportion online buy costs and make an internet-based price tag model.

3.2 PROPOSED METHODOLOGY

The suggested technique predicts and recommends the best prediction system using several machine learning strategies such as KNN, Random Forest, Logistic Regression, XG Boost, and Stacking. Let's begin the training process by using the compare models functionality now that the data has been prepared and the architecture shown in in Fig.1 This function uses cross-validation to train all of the algorithms in the model library and to evaluate numerous number of performance measures.

After data collection, numerous procedures are taken to investigate the data. The goal of this step is to learn about the data structure, perform preliminary pre-processing, clean the data, discover patterns and inconsistencies in the data (such as skewness, outliers, and missing values), and develop and test hypotheses. A customer chooses to acquire a product, there is n o proper and efficient forecast made as to his demands or requirements under the current technique.

To apportion online purchase costs and create an internet-based price tag model, use BG/NBDbased NBD. To gain a better grasp of the feature values, the console is printed with the unique values in the feature fig for each feature. A general idea of the features can be gained by looking at the columns and their unique values.

3.3 SYSTEM ARCHITECTURE

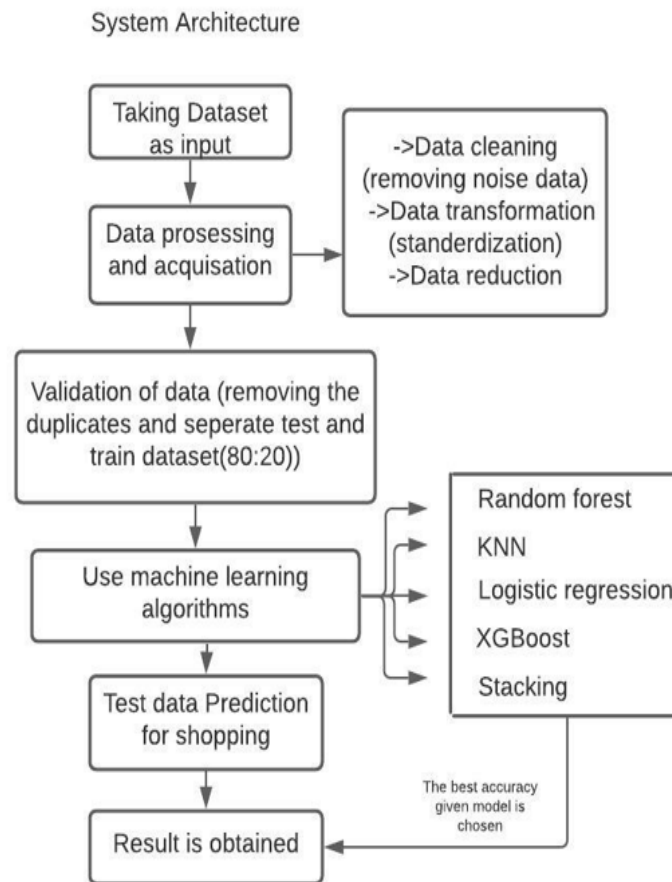


Fig 3.3.1 System Architecture

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the information to recommend improvements on the system. System analysis is a problem-solving activity that requires intensive communication between the system users and system developers.

System analysis or study is an important phase of any system development process. The system is viewed as a whole, the inputs are identified and the system is subjected to close study to identify the problem areas.

The solutions are given as a proposal. The proposal is reviewed on user request and suitable changes are made. This loop ends as soon as the user is satisfied with the proposal. They are many steps involved in this process and has importance to each and every step here.

3.3.1 TAKING INPUT DATA:

First of all, we need to take the dataset from different surveys and convert it into excel sheet format to make it as .csv file. So that we can import the dataset file .csv using pandas into the application for prediction. Using Pandas we can import the file as using the below syntax as:-

```
Eg:-      import pandas as pd
          df=pd.read_csv("dataset file.csv")
          where read_csv is a function to take the dataset,
          df=some randomly chosen file name.
```

ABOUT PANDAS

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. Pandas is built on top of two core Python libraries—matplotlib for data visualization and NumPy for mathematical operations. Pandas acts as a wrapper over these libraries, allowing you to access many of matplotlib's and NumPy's methods with less code. For instance, pandas'.plot()' combines multiple matplotlib methods into a single method, enabling you to plot a chart in a few lines. Many methods available in Pandas as of example are:

1.read_csv(): The function helps to read a comma-separated values (csv) file into a Pandas Data Frame.

2. head(n): it is used to return the first n rows of a dataset.

3. `describe()`: it is used to generate descriptive statistics of the data in a Pandas Data Frame or Series.
4. `memory_usage()`: it returns a Pandas Series having the memory usage of each column (in bytes) in a Pandas Data Frame.
5. `astype()`: it is used to cast a Python object to a particular data type.
6. `loc[]`: it helps to access a group of rows and columns in a dataset, a slice of the dataset, as per our requirement.
7. `to_datetime()`: it converts a Python object to datetime format. It is very powerful when the dataset has time series values or dates.
8. `value_counts()`: it returns a Pandas Series containing the counts of unique values.
9. `groupby()`: it is used to group a Pandas Data Frame by 1 or more columns, and perform some mathematical operation on it.
10. `merge()`: it is used to merge 2 Pandas Data Frame objects or a Data Frame and a Series object on a common column (field).
11. `sort_values()`: it is used to sort column in a Pandas Data Frame (or a Pandas Series) by values in ascending or descending order.
12. `fillna()`: it helps to replace all NaN values in a Data Frame or Series by imputing these missing values with more appropriate values.

ABOUT NUMPY

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open-source project and you can use it freely. NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices. It stands for 'Numerical Python'. Some methods are:

1.arange(): This method takes the start index of the array, the end index, and the step size (which is optional).

2.zeros(): Apart from custom arrays with pre-filled data, you can also create arrays with a simpler set of data. For instance, you can use the method to create an array of all zeros.

3.linspace(): This method takes three arguments: a start index, end index, and the number of linearly-spaced numbers that you want between the specified range.

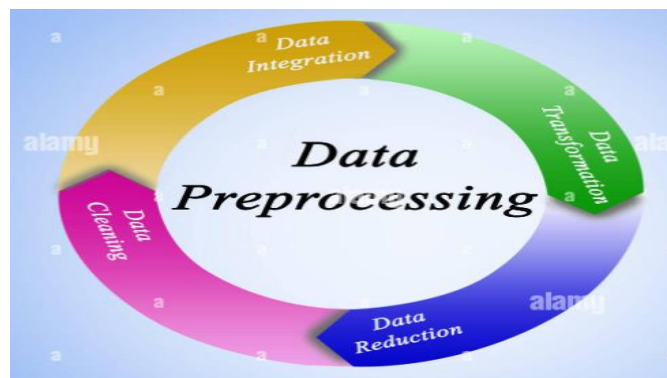
4.random(): to take random numbers in a given range.

5.reshape(): you can convert a one-dimensional array into a two-dimensional array using the reshape method.

3.3.2 DATA PRE-PROCESSING AND ACQUISITION

Data are acquired online after preprocessing. The data can be vibration, voltage, current, acoustic emission, sound signals, etc. Corresponding to the object, different preprocessing approaches are used, such as filtering (high-, low-, and band-pass), wavelet transform, and averaging.

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.



3.3.2.1 Data preprocessing steps

Steps involved are: 1] Data Cleaning

2] Data Integration

3] Data Reduction

4] Data Transformation

DATA CLEANING

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct.

There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset. But it is crucial to establish a template for your data cleaning process so you know you are doing it the right way every time. Essentially, garbage data in is garbage analysis out. Various Steps involved are:

- 1.Remove Duplicate or irrelevant observations: Remove unwanted observations from your dataset, including duplicate observations or irrelevant observations.
- 2.Fix structural errors: Structural errors are when you measure or transfer data and notice strange naming conventions, typos, or incorrect capitalization.
- 3.Filter Unwanted Outliers: If an outlier proves to be irrelevant for analysis or is a mistake, consider removing it.
- 4.Handle Missing Data: Missing data can be handled as-
 - 1.Remove missing values from dataset if there are very small number.
 - 2.Replace the missing data with mean, median or mode observations.
- 5.Validate the data so that data can be free from all the cases mentioned above.

DATA INTEGRATION

Data Integration is a data preprocessing technique that involves combining data from multiple heterogeneous data sources into a coherent data store and provide a unified view of the data. These sources may include multiple data cubes, databases, or flat files. The data integration approaches are formally defined as triple $\langle G, S, M \rangle$ where,

G stand for the global schema,

S stands for the heterogeneous source of schema,

M stands for mapping between the queries of source and global schema.

There are mainly 2 major approaches for data integration – one is the “tight coupling approach” and another is the “loose coupling approach”.

Tight Coupling:

- Here, a data warehouse is treated as an information retrieval component.
- In this coupling, data is combined from different sources into a single physical location through the process of ETL – Extraction, Transformation, and Loading.

Loose Coupling:

- Here, an interface is provided that takes the query from the user, transforms it in a way the source database can understand, and then sends the query directly to the source databases to obtain the result.
- And the data only remains in the actual source databases.

DATA REDUCTION

Data Reduction or Dimensionality reduction refers to techniques for reducing the number of input variables in training data. When dealing with high dimensional data, it is often useful to reduce the dimensionality by projecting the data to a lower dimensional subspace which captures the “essence” of the data. Data reduction techniques are used to obtain a reduced representation of the dataset that is much smaller in volume by maintaining the integrity of the original data. By reducing the data, the efficiency of the data mining process is improved, which produces the same analytical results. The most used method in this project is Dimensionality Reduction.

Dimensionality Reduction: Whenever we encounter weakly important data, we use the attribute required for our analysis. Dimensionality reduction eliminates the attributes from the data set under consideration, thereby reducing the volume of original data. It reduces data size as it eliminates outdated or redundant features. Here are three methods of dimensionality reduction.

- i. **Wavelet Transform:** In the wavelet transform, suppose a data vector A is transformed into a numerically different data vector A' such that both A and A' vectors are of the same length. Then how it is useful in reducing data because the data obtained from the wavelet transform can be truncated. The compressed data is obtained by retaining the smallest fragment of the strongest wavelet coefficients. Wavelet transform can be applied to data cubes, sparse data, or skewed data.
- ii. **Principal Component Analysis:** Suppose we have a data set to be analyzed that has tuples with n attributes. The principal component analysis identifies k independent tuples with n attributes that can represent the data set.
In this way, the original data can be cast on a much smaller space, and dimensionality reduction can be achieved. Principal component analysis can be applied to sparse and skewed data.
- iii. **Attribute Subset Selection:** The large data set has many attributes, some of which are irrelevant to data mining or some are redundant. The core attribute subset selection reduces the data volume and dimensionality. The attribute subset selection reduces the volume of data by eliminating redundant and irrelevant attributes. The attribute subset selection ensures that we get a good subset of original attributes even after eliminating the unwanted attributes. The resulting probability of data distribution is as close as possible to the original data distribution using all the attributes.

DATA TRANSFORMATION

Data transformation is the process of converting raw data into a format or structure that would be more suitable for model building and also data discovery in general. It is an imperative step in feature engineering that facilitates discovering insights. This article will cover techniques of numeric data transformation: log transformation, clipping methods, and data scaling. The scaling transformation techniques are:

Min Max Scaler — normalization

MinMaxScaler() is usually applied when the dataset is not distorted. It normalizes the data into a range between 0 and 1 based on the formula:

$$x' = (x - \min(x)) / (\max(x) - \min(x))$$

Eg: `sklearn.preprocessing import MinMaxScaler.fit(train_data)`

Standard Scaler — standardization

We use standardization when the dataset conforms to normal distribution. StandardScaler() converts the numbers into the standard form of mean = 0 and variance = 1 based on z-score formula:

$$x' = (x - \text{mean}) / \text{standard deviation.}$$

Eg: `sklearn.preprocessing import StandardScaler.fit(train_data)`

Robust Scaler

RobustScaler() is more suitable for dataset with skewed distributions and outliers because it transforms the data based on median and quantile:

$$x = (x - \text{median}) / \text{inter-quartile range.}$$

Eg: `sklearn.preprocessing import RobustScaler.fit(train_data)`

3.3.3 VALIDATION OF DATASET

Data validation means checking the accuracy and quality of source data before using, importing or otherwise processing data. Different types of validation can be performed depending on destination constraints or objectives. Data validation is a form of data cleansing. It focuses on checking that the statistics of the new data are as expected (e.g. feature distribution, number of categories, etc). Separate the train and test data sets using sklearn library.

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm.

The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

- Train Dataset: Used to fit the machine learning model.
- Test Dataset: Used to evaluate the fit machine learning model.

The objective is to estimate the performance of the machine learning model on new data: data not used to train the model.

The procedure has one main configuration parameter, which is the size of the train and test sets. This is most commonly expressed as a percentage between 0 and 1 for either the train or test datasets. For example, a training set with the size of 0.67 (67 percent) means that the remainder percentage 0.33 (33 percent) is assigned to the test set. Nevertheless, common split percentages include:

- Train: 80%, Test: 20%
- Train: 67%, Test: 33%
- Train: 50%, Test: 50%

Eg: `from sklearn.model_selection import train_test_split`

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
```

When to Use the Train-Test Split

The idea of “sufficiently large” is specific to each predictive modeling problem. It means that there is enough data to split the dataset into train and test datasets and each of the train and test datasets are suitable representations of the problem domain. This requires that the original dataset is also a suitable representation of the problem domain.

A suitable representation of the problem domain means that there are enough records to cover all common cases and most uncommon cases in the domain. This

might mean combinations of input variables observed in practice. It might require thousands, hundreds of thousands, or millions of examples.

Conversely, the train-test procedure is not appropriate when the dataset available is small. The reason is that when the dataset is split into train and test sets, there will not be enough data in the training dataset for the model to learn an effective mapping of inputs to outputs. There will also not be enough data in the test set to effectively evaluate the model performance. The estimated performance could be overly optimistic (good) or overly pessimistic (bad). If you have insufficient data, then a suitable alternate model evaluation procedure would be the k-fold cross-validation procedure.

In addition to dataset size, another reason to use the train-test split evaluation procedure is computational efficiency.

Some models are very costly to train, and in that case, repeated evaluation used in other procedures is intractable. An example might be deep neural network models. In this case, the train-test procedure is commonly used.

Alternately, a project may have an efficient model and a vast dataset, although may require an estimate of model performance quickly. Again, the train-test split procedure is approached in this situation.

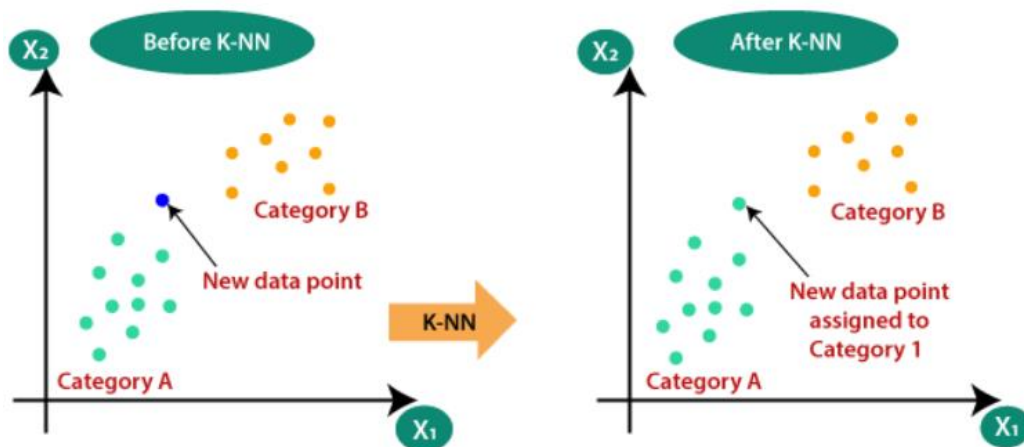
Samples from the original training dataset are split into the two subsets using random selection. This is to ensure that the train and test datasets are representative of the original dataset.

3.3.4 USING DIFFERENT MACHINE LEARNING ALGORITHMS

In this Project we have used different types of Machine Learning Algorithms among all we have selected the one that give us best accuracy and fix it as best model. The different models used are:

3.3.4.1.KNN(K-Nearest Neighbour)

The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slower as the size of the data in use grows. The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.



3.3.4.1 KNN Processing

Steps Involved in KNN are:

- Step-1: Select the number K of the neighbours
- Step-2: Calculate the Euclidean distance of K number of neighbours
- Step-3: Euclidean Distance = $\sqrt{(x - x_i)^2 + (y - y_i)^2}$
- Step-4: Take the K nearest neighbours as per the calculated Euclidean distance.
- Step-5: K-value depends on hyperparameter optimization based on elbow method
- Step-6: Among these k neighbours, count the number of the data points in each category

- Step-7: Assign the new data points to that category for which the number of the neighbour is maximum.
- Step-8: Our model is ready.

Choosing the right value for K

To select the K that's right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before.

Here are some things to keep in mind:

1.As we decrease the value of K to 1, our predictions become less stable. Just think for a minute, imagine $K=1$ and we have a query point surrounded by several reds and one green (I'm thinking about the top left corner of the colored plot above), but the green is the single nearest neighbor. Reasonably, we would think the query point is most likely red, but because $K=1$, KNN incorrectly predicts that the query point is green.

2.Inversely, as we increase the value of K, our predictions become more stable due to majority voting / averaging, and thus, more likely to make more accurate predictions (up to a certain point). Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far.

3.In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.

Advantages

1. The algorithm is simple and easy to implement.
2. There's no need to build a model, tune several parameters, or make additional assumptions.

3. The algorithm is versatile. It can be used for classification, regression, and search (as we will see in the next section).

Disadvantages

1. The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x_1 , so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.

Applications of KNN

- Text mining.
- Agriculture.
- Finance.
- Medical.
- Facial recognition.

3.3.4.2 RANDOM FOREST

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

As the name suggests, *"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."* Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The

greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

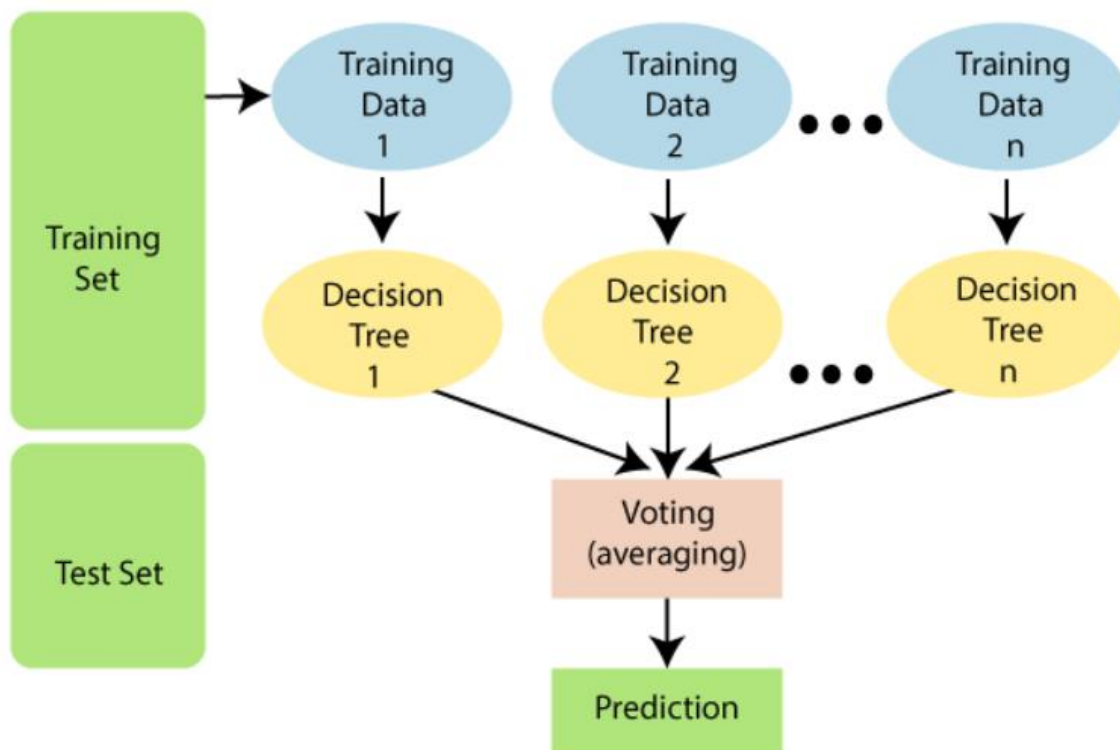


Fig 3.3.4.2.1 Random Forest Processing

Steps involved in this method are:

- Step-1: Select random K data points from the training set.
- Step-2: Build the decision trees associated with the selected data points (Subsets).
- Step-3: Choose the number N for decision trees that you want to build.
- Step-4: Number of Decision trees used are depends on the hyperparameterisation .
- Step-4: Repeat Step 1 & 2.
- Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes. Majority voting is given priority, this is called Row Sampling with replacement.
- Step-6: When the dataset is changed like out of 1000 records 200 records got changed, it does not have more impact on models, as only some random samples will be transferred for the algorithm, but it has less impact on accuracy and error loss.

Advantages

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

Applications

There are mainly four sectors where Random forest mostly used:

1. Banking: Banking sector mostly uses this algorithm for the identification of loan risk.
2. Medicine: With the help of this algorithm, disease trends and risks of the disease can be identified.

3. Land Use: We can identify the areas of similar land use by this algorithm.
4. Marketing: Marketing trends can be identified using this algorithm.

3.3.4.3. XGBOOST (XTREME GRADIENT BOOSTING)

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models.

To calculate the gain in each split, XGBoost uses CPU cache to store calculated gradients and Hessians (cover) to make the necessary calculations fast. When data does not fit into the cache and main memory, then it becomes important to use the disk space. The XGBoost library provides two wrapper classes that allow the random forest implementation provided by the library to be used with the scikit-learn machine learning library. They are the XGB RF Classifier and XGB RF Regressor classes for classification and regression respectively.

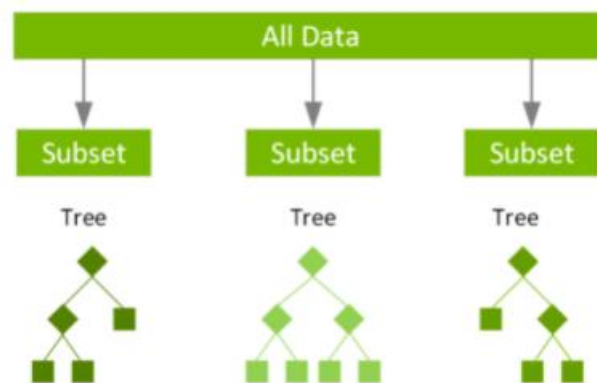


Fig 3.3.4.3.1 XGBoost Processing

Steps involved in this process is:

- Step-1: An initial model F_0 is defined to predict the target variable y . This model will be associated with a residual $(y - F_0)$
- Step-2: A new model h_1 is fit to the residuals from the previous step, Similarity weight = $\frac{\sum (\text{Residual})^2}{\sum (\text{prob}(1-\text{prob})) + \lambda}$ (λ depends on hyperparameter)
- Step-3: Now, F_0 and h_1 are combined to give F_1 , the boosted version of F_0 . The mean squared error from F_1 will be lower than that from F_0 :

calculate gain = sum of all sub trees similarity weight - similarity weight of root

- To improve the performance of F_1 , take the one which has highest gain and divide based on that we could model after the residuals of F_1 and create a new model F_2 : POST PRUNING-Whether the split is to do or not is said by this method.
- This can be done for ' m ' iterations, until residuals have been minimized as much as possible.

Advantages:

- It is Highly Flexible.
- It uses the power of parallel processing.
- It is faster than Gradient Boosting.
- It supports regularization.
- It is designed to handle missing data with its in-build features.
- The user can run a cross-validation after each iteration.
- Often provides predictive accuracy that cannot be trumped.

Disadvantages

- XGBoost does not perform so well on sparse and unstructured data
- Gradient Boosting is very sensitive to outliers since every classifier is forced to fix the errors in the predecessor learners.
- Overall method is hardly scalable.

- When the number of training samples is significantly smaller than the number of features.
- Unlike LightGBM, in XGB, one has to manually create dummy variable/ label encoding for categorical features before feeding them into the models.

Applications

- Can be used to solve regression, classification, ranking, and user-defined prediction problems.
- Portability: Runs smoothly on Windows, Linux, and OS X. Languages: Supports all major programming languages including C++, Python, R, Java, Scala, and Julia.

3.3.4.4 LOGISTIC REGRESSION

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets. Logistic Regression can be used to classify the observations using different

types of data and can easily determine the most effective variables used for the classification.

Mathematically, a logistic regression model predicts $P(Y=1)$ as a function of X . It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc. Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

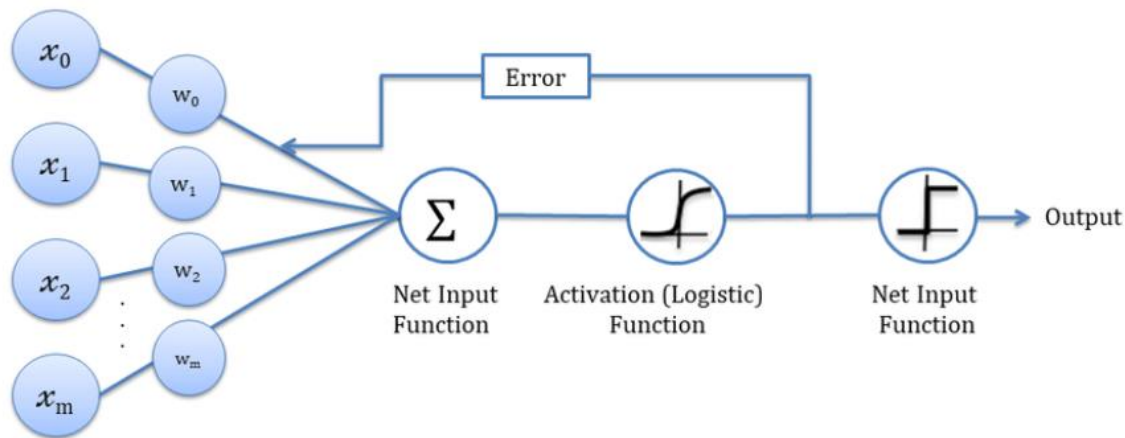


Fig 3.3.4.4 Logistic Regression Processing

Steps involved in this method are:

- Step-1: When there are points on the graph then the line is drawn in such a way that has the minimum distance between line and points i.e, R^2
- Step-2: Now when the new data points are added then the minimized line has to change and the line equation will be in form of $y=mx+c$ where m =slope, c =intercept.
- Step-3: But when the outlier comes into model then the line total changes towards the outlier to make it minimize ass it results in high error rate .so we use Logistic Regression here.

- Step-4: Then it starts dividing the points based on y values. There are few conditions that apply is :

i) $y > 0$, the slope is positive

ii) $y < 0$, the slope is negative

Step-5: Finally the overall cost function is calculated as: $\max \sum_{i=1}^n y_i + w^t * x_i$.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Advantages

- Logistic regression is easier to implement, interpret, and very efficient to train.
- It can easily extend to multiple classes (multinomial regression) and a natural probabilistic view of class predictions.
- It is very fast at classifying unknown records.
- It can interpret model coefficients as indicators of feature importance.
- Good accuracy for many simple data sets and it performs well when the dataset is linearly separable.
- It not only provides a measure of how appropriate a predictor(coefficient size)is, but also its direction of association (positive or negative).

Disadvantages

- If the number of observations is lesser than the number of features, Logistic Regression should not be used, otherwise, it may lead to overfitting.

- The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.
- Non-linear problems can't be solved with logistic regression because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.
- It is tough to obtain complex relationships using logistic regression. More powerful and compact algorithms such as Neural Networks can easily outperform this algorithm.
- It constructs linear boundaries.

Applications

- Credit scoring
- Medicine
- Text editing
- Hotel Booking
- Gaming

3.3.4.5 STACKING

Stacking is a way to ensemble multiple classifications or regression model. There are many ways to ensemble models, the widely known models are *Bagging* or *Boosting*. Bagging allows multiple similar models with high variance are averaged to decrease variance. Boosting builds multiple incremental models to decrease the bias, while keeping variance small.

Stacking (sometimes called *Stacked Generalization*) is a different paradigm. The point of stacking is to explore a space of different models for the same problem. The idea is that you can attack a learning problem with different types of models which are capable to learn some part of the problem, but not the whole space of the problem. So, you can build multiple different learners and you use them to build an intermediate prediction, one prediction for each learned model. Then you add a new model which learns from the intermediate predictions the same target.

This final model is said to be stacked on the top of the others, hence the name. Thus, you might improve your overall performance, and often you end up with a model which is better than any individual intermediate model. Notice however, that it does not give you any guarantee, as is often the case with any machine learning technique. They can improve the existing accuracy that is shown by individual models. We can get most of the Stacked models by choosing diverse algorithms in the first layer of architecture as different algorithms capture different trends in training data by combining both of the models can give better and accurate results.

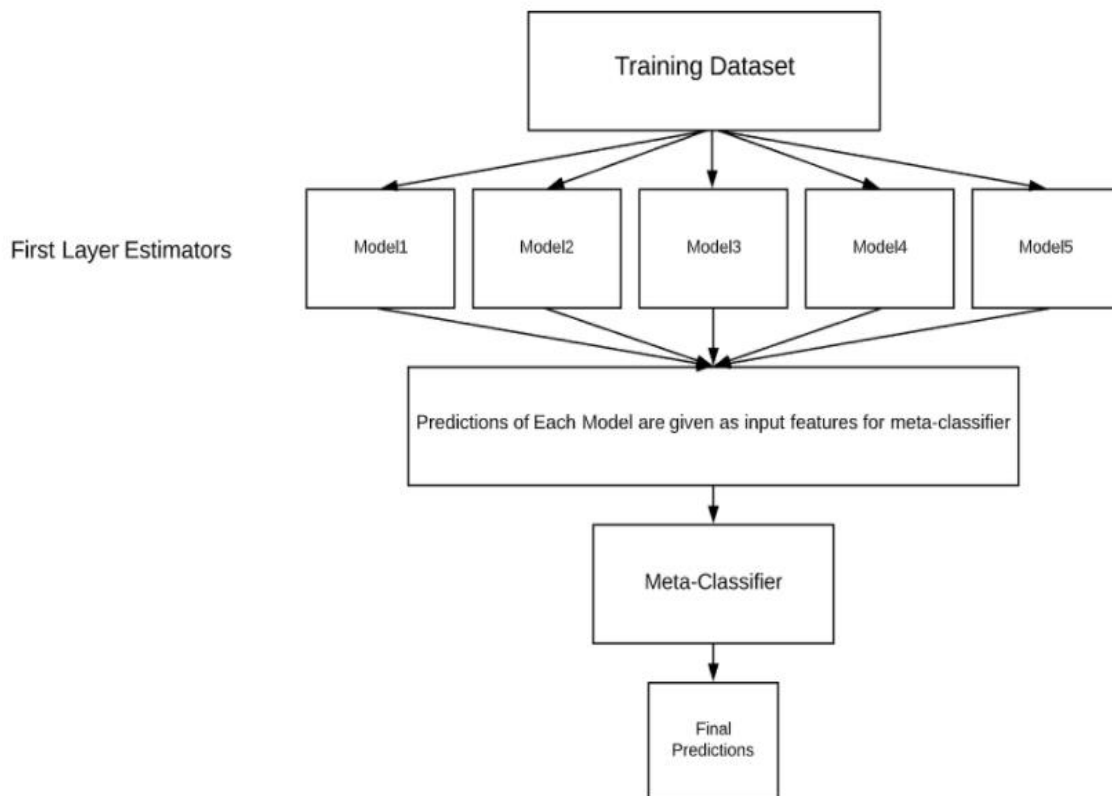


Fig 3.3.4.5.1 Stacking Processing

The first layer consists of all the baseline models that are used to predict the outputs on the test datasets. The second layer consists of Meta-Classifier or Regressor which takes all the predictions of baseline models as an input and generate new predictions.

Steps involved in this method are:

- We split the training data into K-folds just like K-fold cross-validation.
- A base model is fitted on the K-1 parts and predictions are made for Kth part.
- We do for each part of the training data.
- The base model is then fitted on the whole train data set to calculate its performance on the test set.
- We repeat the last 3 steps for other base models.
- Predictions from the train set are used as features for the second level model.
- Second level model is used to make a prediction on the test set.

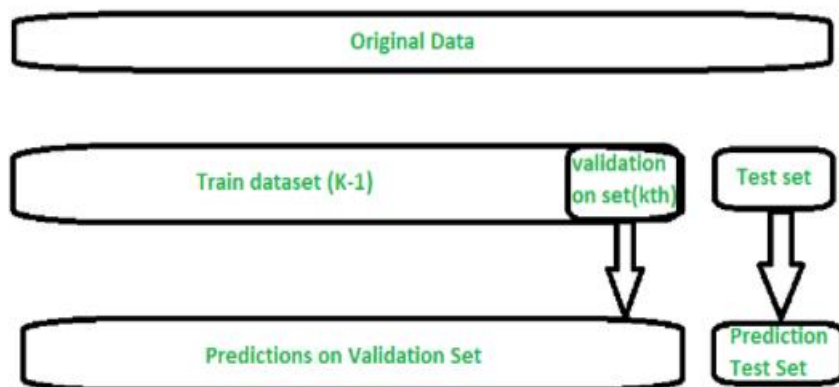


Fig 3.3.4.5.2 Model training of stacking

Advantages

- The benefit of stacking is that it can harness the capabilities of a range of well-performing models on a classification or regression task and make predictions that have better performance than any single model in the ensemble.
- Strengthens different models to combine their predictions.
- Capability to use as many as models it required.
- Majority of all algorithms output is considered as final output.

- Stacking reduces variance and creates a more robust model by combining the predictions of multiple models.

Disadvantages

- Stacked models can take significantly longer to train than simpler models and require more memory.
- Generating predictions using stacked models will usually be slower and more computationally expensive. This drawback is important to consider if you are planning to deploy a stacked model into production.

Applications

- Ensemble model so applications also depends on the applications of algorithms it uses.

3.3.5 TEST OF DATASET

A simple evaluation method is a train test dataset where the dataset is divided into a train and a test dataset, then the learning model is trained using the train data and performance is measured using the test data. In a more sophisticated approach, the entire dataset is used to train and test a given model. Because the data in the testing set already contains known values for the attribute that you want to predict, it is easy to determine whether the model's guesses are correct.

The Test dataset provides the gold standard used to evaluate the model. It is only used once a model is completely trained(using the train and validation sets). The test set is generally what is used to evaluate competing models (For example on many Kaggle competitions, the validation set is released initially along with the training set and the actual test set is only released when the competition is about to close, and it is the result of the the model on the Test set that decides the winner). Many a times the validation set is used as the test set, but it is not good practice. The test set is generally well curated. It contains carefully sampled data that spans the various classes that the model would face, when used in the real world.

3.3.6 METRICS USED FOR THIS DATASET

Metrics are used to train a machine learning model (using some kind of optimization like Gradient Descent), and they're usually differentiable in the model's parameters. Metrics are used to monitor and measure the performance of a model (during training and testing), and don't need to be differentiable. Some of metrics used here are Accuracy, F1 Score, Confusion Matrix etc.,

CONFUSION MATRIX

A Confusion matrix is an $N \times N$ matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**.

Need for Confusion Matrix in Machine learning

- It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is.
- It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error.
- With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

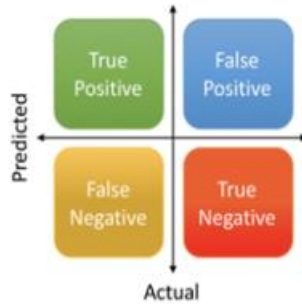


Fig 3.3.6.1 Classification graph

- **True Positive (TP)** -The predicted value matches the actual value and the actual value was positive and the model predicted a positive value.
- **True Negative (TN)** -The predicted value matches the actual value and the actual value was negative and the model predicted a negative value.
- **False Positive (FP) – Type 1 error**-The predicted value was falsely predicted and the actual value was negative but the model predicted a positive value. Also known as the **Type 1 error**.
- **False Negative (FN) – Type 2 error**-The predicted value was falsely predicted and the actual value was positive but the model predicted a negative value. Also known as the **Type 2 error**.

ACCURACY

Model accuracy is defined as the number of classifications a model correctly predicts divided by the total number of predictions made. It's a way of assessing the performance of a model, but certainly not the only way. It is the measurement used to determine which model is best at identifying relationships and patterns between variables in a dataset based on the input, or training, data.

There are many instances in which a first pass at model training is insufficient such that the accuracy needs to be increased. This is normal in machine learning—it's nearly impossible to create a model that perfectly fits all the objectives you initially set out to achieve on your first try, and some iteration on your initial models is necessary. One way to improve model accuracy is by tuning model

parameters. For example, if one has a 3-layer neural network classifier with 100 nodes in its hidden layer, one could retry training the neural network with more or fewer nodes. This is called a parameter search, and is best implemented by first making relatively large changes to the parameter one is tuning, and then successively smaller changes as one settles closer to an optimum value.

Model accuracy is a very important component in assessing model performance, but is in no way the only metric. When evaluating a model, it is important to do so holistically, taking into account a variety of metrics and heuristics in order to assess how a model is performing on a specific dataset. Otherwise the accuracy metrics is not complete.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{False Negative} + \text{True Negative}}$$

PRECISION

Precision is defined as the *ratio of correctly classified positive samples (True Positive) to a total number of classified positive samples* (either correctly or incorrectly). Precision is one indicator of a machine learning model's performance – the quality of a positive prediction made by the model.

- Precision refers to the number of true positives divided by the total number of positive predictions (i.e., the number of true positives plus the number of false positives).
- The precision is imported as `sklearn.metrics import precision_score` into the implementation.

$$\text{Precision} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Positive (FP)}}$$

RECALL

The recall is calculated as the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples. The *recall measures the model's ability to detect positive samples*. The

higher the recall, the more positive samples detected. Recall is the ability of a model to find all the relevant cases within a data set.

- Mathematically, we define recall as the number of true positives divided by the number of true positives plus the number of false negatives.
- The Recall is imported as `sklearn.metrics import recall_score` into the implementation.

$$\text{Recall} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Negative (FN)}}$$

F1-SCORE

It is also called the F Score or the F Measure. Put another way, the F1 score conveys the balance between the precision and the recall. The F1 for the All No Recurrence model is $2*((0*0)/(0+0))$ or 0. For example, for our dataset, we can consider that achieving a high recall is more important than getting a high precision – we would like to detect as many heart patients as possible.

There are also a lot of situations where both precision and recall are equally important. For example, for our model, if the doctor informs us that the patients who were incorrectly classified as suffering from heart disease are equally important since they could be indicative of some other ailment, then we would aim for not only a high recall but a high precision as well.

$$\text{F1 score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

3.4 WORKFLOW DIAGRAM

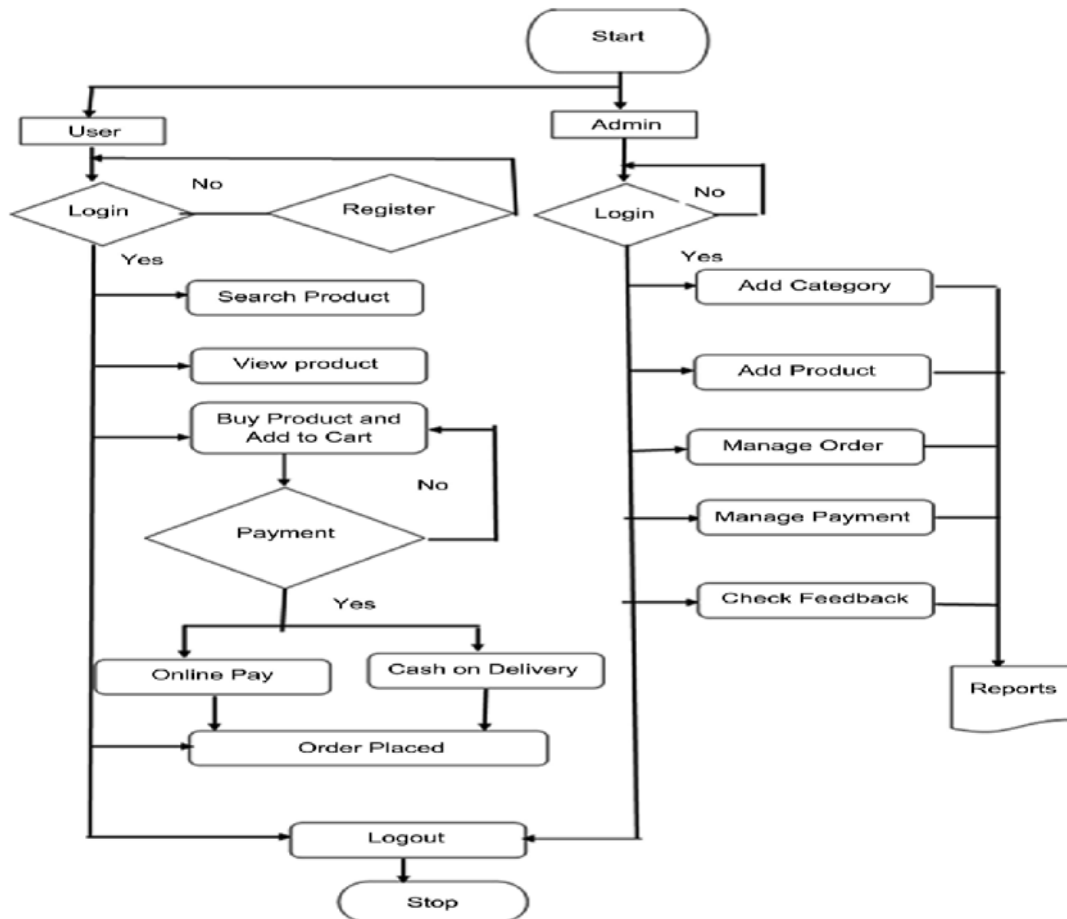


Fig 3.4.1 Work-Flow Diagram

- Display all the available categories for shopping on the home page.
- Display all the subcategories on the home page, that are associated with any particular item.
- Admin has the authority to add new particulars to the items list whenever needed.
- Permission to the administrator to remove items, anytime.
- Allows the admin to modify the price of each item, whenever required or felt like.
- Admin has the authority to update the description of each item.
- Permission to the admin to view information about each customer who checkouts the items list.

CHAPTER 4

RESULTS AND DISSCUSSIONS

4.1 RESULTS

We have achieved the maximum accuracy as (90.65%) for our prediction model using Stacking amongst all the algorithms that has been used to predict the outcome Stacking gave us the best accuracy. The results of our tests were quantified in terms of the accuracy score and F1-score of our predictions. score is a statistical Accuracy -measure of how close the data are to the fitted classification class. With different models the analysis are-like this:

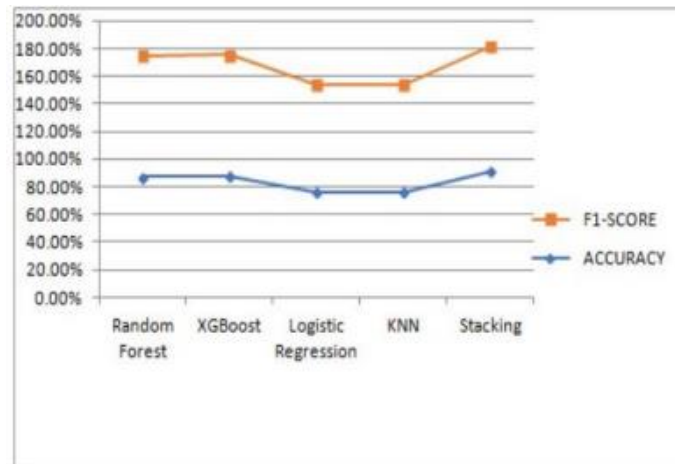
Fig 4.1 Result Table

MODEL	ACCURACY	F1 SCORE
Random forest	86.98%	87.79%
XGBoost	87.44%	88.20%
Logistic Regression	76.34%	77.42%
KNN	76.30%	77.50%
Stacking	90.65%	90.82%

According to Result table we have used five different algorithms to get the best accuracy and F1 Score. Among them the best one we have chosen is Stacking is a ensemble method that combines all the model predictions and try to give the accuracy which is best among all.

Stacking enable us to use as many different model combinations as we can use and combine it using MLP method where it works in the same way as the Neural Network works with the activation function and even with the input, output functions.

Graph 4.1 Title Comparison Chart for the performance metrics



4.2 REQUIRED ANALYSIS

Compared to Logistic Regression, most Decision-Tree based methods did perform comparably very well. This can be attributed to the apparent linearity of the dataset. We believe that it can also be attributed to the easily in tuning the hyperparameters for most gradient boost methods.

The exception to this is the Random Forest method which marginally outperforms Logistic Regression. However Random Forests tend to overfit the dataset due to the tendency of growing longer trees. This was worked upon by restricting the depth of trees to different values and it was observed that beyond limiting depth to 32 resulted in negligible improvement in prediction performance but progressively increased overfitting.

Building up from the relatively good performance of stacking, Logistic Regression + KNN Learning Method (with K = depends) + Random

Forest + XGBoost produced the best accuracy score on test data without high variance and with low bias. Finally Among all the stacking gives the best accuracy of (90.65%) and F1-Score of (90.85%).

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

In this study, we have developed an algorithm which gives Ensemble approach on Online shopping churn prediction system that people can utilize. The model developed in this study seeks to provide the best suggestion based on the user's daily consuming / regularly shopping needs with an accuracy of 90.65 percent. The proposed approach might be highly beneficial to shoppers since it provides precise suggestions, which means there is a strong probability of Increased online shopping.

In this, we have discussed that how our proposed system predicted shopping system using machine learning algorithms. The proposed system is also scalable for predicting shopping whether it is satisfied or not. The appeal of the proposed work is related to the need to set up a model that forecasts the visitor's shopping intent as soon as the e-commerce website is visited. The advantage behind that is to avoid the risk of abandon implied by each visit on the website. In this scope, the challenging issue consisted in finding out the most appropriate machine learning or which could reach this purpose.

5.2 SCOPE OF FUTURE WORK

The things need to be done in future are:

- The current system can be extended to allow the users to create accounts and save products in to wish list.
- The users could subscribe for price alerts which would enable them to receive messages when price for products fall below a particular level.

- The current system is confined only to the shopping cart process. It can be extended to have an easy to use check out process.
- Users can have multiple shipping and billing information saved. During checkout they can use the drag and drop feature to select shipping and billing information.
- Based on shopping levels users are allowed to given coupons , gifts and festive offers an a lot of offers.

REFERENCES

- [1] Libai, B.; Bart, Y.; Gensler, S.; Hofacker, C.F.; Kaplan, A.; Kötterheinrich, K.; Kroll, E.B. Brave new world? On AI and the management of customer relationships. *J. Interact. Mark.* 2020, 51, 44–56.
- [2] Ballestar, M.T.; Grau-Carles, P.; Sainz, J. Predicting customer quality in e-commerce social networks: A machine learning approach. *Rev. Manag. Sci.* 2018, 13, 589–603.
- [3] Goldstein, A.; Oestreicher-Singer, G.; Barzilay, O. Deep into the funnel? Predicting online conversion using search diversity. *SSRN Electron. J.* 2017.
- [4] Trusov, M.; Ma, L.; Jamal, Z. Crumbs of the cookie: User profiling in customer-base analysis and behavioral targeting. *Mark. Sci.* 2016, 35, 405–426.
- [5] Liu, X.; Lee, D.; Srinivasan, K. Large-scale cross-category analysis of consumer review content on sales conversion leveraging deep learning. *J. Mark. Res.* 2019, 56, 918–943.
- [6] Chakraborty, I.; Kim, M.; Sudhir, K. attribute sentiment scoring with online text reviews: Accounting for language structure and attribute self-selection. *SSRN Electron. J.* 2019.
- [7] Kakalejczyk, L.; Bucko, J.; Vejacka, M. differences in buyer journey between high- and low-value customers of e-commerce business. *J. Theor. Appl. El. Comm.* 2019, 14, 47–58.

APPENDICIES

A.SAMPLE CODE

ML CODE

```
import pandas as pd
from matplotlib import pyplot as plt
import numpy as np
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold, StratifiedKFold, RepeatedStratifiedKFold
from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import accuracy_score, f1_score, recall_score, precision_score
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import StackingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegression
df = pd.read_csv('C:\\Users\\Akshitha reddy\\Desktop\\customer_churn.csv')
df.head()
df.shape
columns = df.columns
for i in range(len(columns)):
    if df[columns[i]].dtypes!=object:
```

```

    print("unique number of {} -> {}".format(columns[i], len(df[columns[i]].unique()))
print("Cateogrical fiedls")
for i in range(len(columns)):
    if df[columns[i]].dtypes==object:
        print("unique number of {} -> {}".format(columns[i], len(df[columns[i]].unique()))
df.info()
df.isnull().sum()
df.describe()
df.drop('customerID',axis='columns',inplace=True)
pd.to_numeric(df.TotalCharges,errors='coerce').isnull()
df[pd.to_numeric(df.TotalCharges,errors='coerce').isnull()]
df['TotalCharges'] = df['TotalCharges'].replace(' ', np.nan, regex=True)
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'])
sns.countplot(x = "Churn", data = df)
df.loc[:, 'Churn'].value_counts()
categorical_feature = df.dtypes==object
final_categorical_feature = df.columns[categorical_feature].tolist()
numeric_feature = df.dtypes!=object
final_numeric_feature = df.columns[numeric_feature].tolist()
df[final_numeric_feature].plot(kind='density', subplots=True, layout=(2,3), sharex=False,
figsize= (20,8))
plt.show()
print("Mode: {}".format(df.TotalCharges.mode()))
print("Mean: {}".format(df.TotalCharges.mean()))
print("Median: {}".format(df.TotalCharges.median()))
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
def lower_outlier(df, col):
    q1_col = Q1[col]
    iqr_col = IQR[col]
    q3_col = Q3[col]
    lower = df[(df[col] < (q1_col - 1.5 * iqr_col))]
    return lower
def upper_outlier(df, col):
    q1_col = Q1[col]
    iqr_col = IQR[col]
    q3_col = Q3[col]
    upper = df[(df[col] > (q3_col + 1.5 * iqr_col))]
    return upper
for i in range(len(final_numeric_feature)):
    preprocess(df[final_numeric_feature], final_numeric_feature[i])
for i in range(len(final_categorical_feature)):
    print("*****".format(final_categorical_feature[i]))
    print(df[final_categorical_feature[i]].unique())
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import RobustScaler
le = LabelEncoder()

```

```

df_dummy[final_categorical_feature] = df_dummy[final_categorical_feature].apply(lambda
col: le.fit_transform(col))
df_dummy.head(5)
mask = np.triu(np.ones_like(df_dummy.corr()))
fig, ax = plt.subplots(figsize=(16,10),dpi=80, facecolor='w', edgecolor='k')
sns.heatmap(df_dummy.corr(), mask= mask, cmap="YlGnBu", annot = True, center =
0,annot_kws={"fontsize":8})
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state =100 ,stratify=y, test_size
= 0.2)
print(y_train.value_counts())
scale = MinMaxScaler()
col = X_train.columns
X_train[col] = scale.fit_transform(X_train[col])
X_test[col] = scale.transform(X_test[col])
X_train_randOverSample = X_train.copy()
y_train_randOverSample = y_train.copy()
from imblearn.over_sampling import SMOTE, ADASYN, BorderlineSMOTE, SVMSMOTE,
RandomOverSampler
print("Before OverSampling- counts of label '1': {}".format(sum(y==1)))
print("Before OverSampling- counts of label '0': {} \n".format(sum(y==0)))
print("*****")
randOverSample = RandomOverSampler(random_state=100,)
X = df_dummy.drop("Churn", axis = 1)
y = df_dummy['Churn']
X_train_randOverSample, y_train_randOverSample =
randOverSample.fit_resample(X_train_randOverSample, y_train_randOverSample.ravel())
print('After OverSampling with RandomOverSampler- X_train:
{}'.format(X_train_randOverSample.shape))
print('After OverSampling with RandomOverSampler- y_train: {}
\n'.format(y_train_randOverSample.shape))
print("After OverSampling with RandomOverSampler- counts of label '1':
{}".format(sum(y_train_randOverSample==1)))
print("After OverSampling with RandomOverSampler- counts of label '0':
{}".format(sum(y_train_randOverSample==0)))
CV = StratifiedKFold(n_splits=10, random_state=0, shuffle=True)
def RandomSearch_Tune(model, Data, X, y, params):
    clf = RandomizedSearchCV(model, params, scoring='accuracy', cv = CV, n_jobs=-1)
    clf.fit(X, y)
    print("best score is :", clf.best_score_)
    print("best estimator is :", clf.best_estimator_)
    print("best Params is :", clf.best_params_)
    return (clf.best_score_)
sfs_randOverSample = SFS(GradientBoostingClassifier(n_estimators=100, random_state=0),
    k_features = 9,
    forward= True,
    floating = False,
    verbose= 2,
    scoring= 'accuracy',
    cv=CV,
    n_jobs= -1).fit(X_train_randOverSample, y_train_randOverSample)
print("Feature Names: ",sfs_randOverSample.k_feature_names_)

```

```

print("Score: ", sfs_randOverSample.k_score_)
MLP = MLPClassifier(activation='tanh', hidden_layer_sizes=20, learning_rate_init=0.025,
max_iter=100, random_state=0)
cross_val_MLP = cross_val_score(MLP, X_train_randOverSample, y_train_randOverSample
, cv = CV, scoring = 'accuracy')
cross_val_MLP_f1 = cross_val_score(MLP, X_train_randOverSample,
y_train_randOverSample , cv = CV,
scoring = 'f1')
estimators = [('DT', DT),
('RF', RF),
('MLP', MLP),
('KNN',KNN),
('LR', LR),
('XGB', XGB)]
SC_tune = StackingClassifier(estimators = estimators, final_estimator = MLP)
cross_val_SC_tune = cross_val_score(SC_tune, X_train_randOverSample,
y_train_randOverSample , cv = CV, scoring = 'accuracy' )
cross_val_SC_tune_f1 = cross_val_score(SC_tune, X_train_randOverSample,
y_train_randOverSample , cv = CV, scoring = 'f1' )
compare_models = [('Stacking',cross_val_SC_tune.mean(), cross_val_SC_tune_f1.mean())]
predict = pd.DataFrame(data = compare_models, columns=['Model','Accuracy',
'F1_score']),predict
import pickle
file=open('stacking.pkl' , 'wb')
pickle.dump(MLP,file)

```

App.py

```

from flask import Flask, escape, request ,render_template
import pickle
model = pickle.load(open('GB_Model.pkl', 'rb'))
app = Flask(__name_)
@app.route('/')
def home():
    return render_template("index.html")
@app.route('/analysis')
def analysis():
    return render_template("churn.html")
@app.route('/prediction', methods=['GET' , 'POST'])
def prediction():
    if request.method == "POST" :
        age = request.form['age']
        last_login = int(request.form['last_login'])
        avg_time_spent = float(request.form['avg_time_spent'])
        avg_transaction_value = float(request.form['avg_transaction_value'])
        points_in_wallet = float(request.form['points_in_wallet'])
        date = request.form['date']
        avg_frequency_login_days = float(request.form['avg_frequency_login_days'])
        gender = request.form['gender']
        region_category = request.form['region_category']
        membership_category = request.form['membership_category']
        joined_through_referral = request.form['joined_through_referral']
        preferred_offer_types = request.form['preferred_offer_types']

```

```

medium_of_operation = request.form['medium_of_operation']
internet_option = request.form['internet_option']
used_special_discount = request.form['used_special_discount']
offer_application_preference = request.form['offer_application_preference']
past_complaint = request.form['past_complaint']
complaint_status = request.form['complaint_status']
feedback = request.form['feedback']
if gender == "M":
    gender=1
else:
    gender=0
# region_category
if region_category == 'Town':
    region_category=1
elif region_category == 'Village':
    region_category=2
else:
    region_category=0
# membership_category
if membership_category == 'Basic Membership':
    membership_category=0
elif membership_category == 'No Membership':
    membership_category=2
elif membership_category == 'Gold Membership':
    membership_category=1
elif membership_category == 'Platinum Membership':
    membership_category=3
elif membership_category == 'Silver Membership':
    membership_category=5
else:
    membership_category=4
# joined_through_referral
if joined_through_referral == 'Yes':
    joined_through_referral=1
else:
    joined_through_referral=0
# preferred_offer_types
if preferred_offer_types == 'Gift Vouchers/Coupons':
    preferred_offer_types=1
elif preferred_offer_types == 'Credit/Debit Card Offers':
    preferred_offer_types=0
else:
    preferred_offer_types=2
# medium_of_operation
if medium_of_operation == 'Desktop':
    medium_of_operation=1
elif medium_of_operation == 'Smartphone':
    medium_of_operation=2
else:
    medium_of_operation=0
# internet_option

```

```

if internet_option == 'Wi-Fi':
    internet_option=2
elif internet_option == 'Mobile_Data':
    internet_option=1
else:
    internet_option=0
# used_special_discount
if used_special_discount == 'Yes':
    used_special_discount=1
else:
    used_special_discount=0
# offer_application_preference
if offer_application_preference == 'Yes':
    offer_application_preference=1
else:
    offer_application_preference=0
# past_complaint
if past_complaint == 'Yes':
    past_complaint=1
else:
    past_complaint=0
# complaint_status
if complaint_status == 'Not Applicable':
    complaint_status=1
elif complaint_status == 'Solved in Follow-up':
    complaint_status=3
elif complaint_status == 'No Information Available':
    complaint_status=0
elif complaint_status == 'Unsolved':
    complaint_status=4
else:
    complaint_status=2

# feedback
if feedback == 'Poor Product Quality':
    feedback=2
elif feedback == 'No reason specified':
    feedback=0
elif feedback == 'Too many ads':
    feedback=7
elif feedback == 'Poor Website':
    feedback=3
elif feedback == 'Poor Customer Service':
    feedback=1
elif feedback == 'Reasonable Price':
    feedback=6
elif feedback == 'User Friendly Website':
    feedback=8
elif feedback == 'Products always in Stock':
    feedback=4
else:

```

```

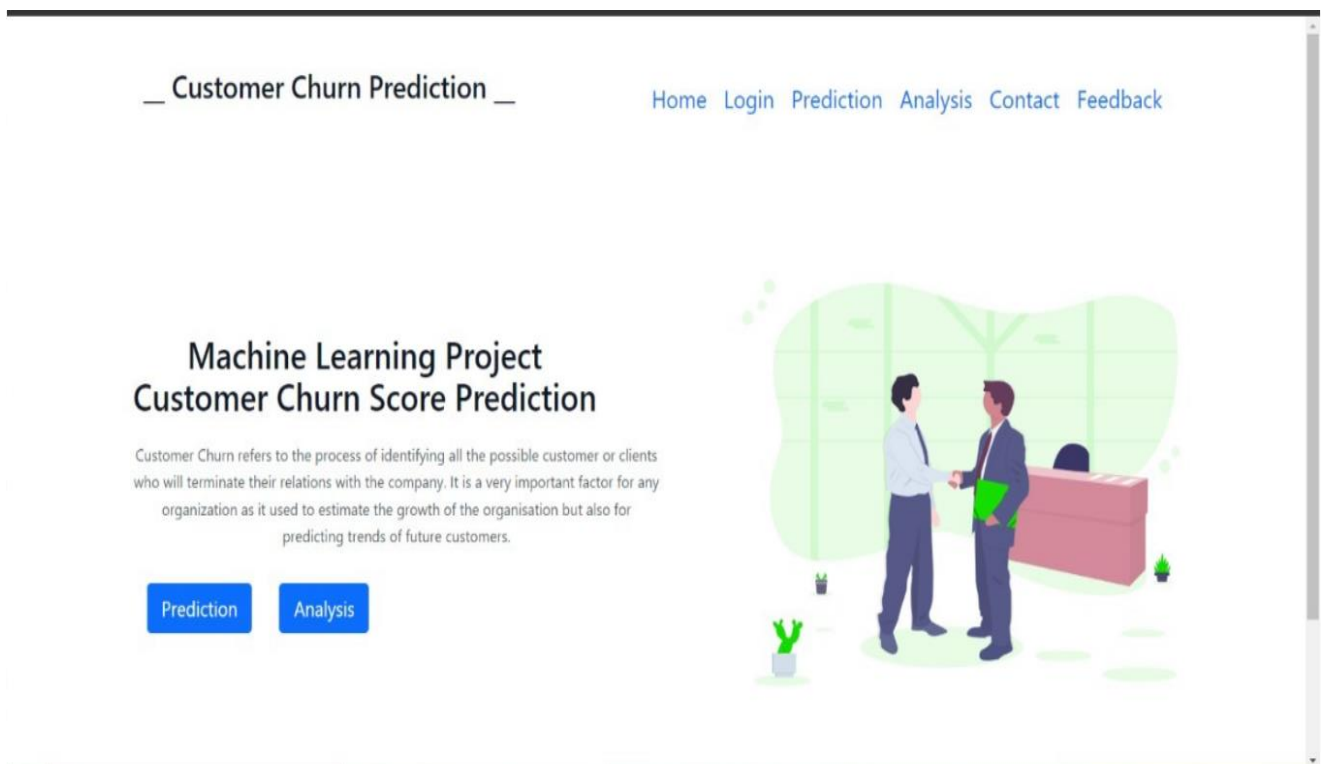
        feedback=5
    date2 = date.split('-')
    joining_day = int(date2[0])
    joining_month = int(date2[1])
    joining_year = int(date2[2])
    data = { 'age':[age], 'gender':[gender],
    'region_category':[region_category], 'membership_category':[membership_category],
    'joined_through_referral':[joined_through_referral],
        'medium_of_operation':[medium_of_operation],
        'internet_option':[internet_option],
        'days_since_last_login':[last_login],
        'avg_time_spent':[avg_time_spent],
        'avg_transaction_value':[avg_transaction_value],
        'avg_frequency_login_days':[avg_frequency_login_days],
        'points_in_wallet':[points_in_wallet],
        'used_special_discount':[used_special_discount],
        'offer_application_preference':[offer_application_preference],
        'past_complaint':[past_complaint],
        'complaint_status':[complaint_status],
        'feedback':[feedback],
        'joining_day':[joining_day],
        'joining_month':[joining_month], 'joining_year':[joining_year]
    }
    import pandas as pd
    df = pd.DataFrame.from_dict(data)
    pred = model.predict(df)
    #print(pred)
    if(pred<0):
        return render_template("prediction.html" , pred_text="Churn Score is {} not
churned".format(pred))
    else:
        return render_template("prediction.html" , pred_text="Churn Score is {}
churned".format(pred))
    else :
        return render_template("prediction.html")
if __name__ == "__main__":
    app.run(debug=True)

```

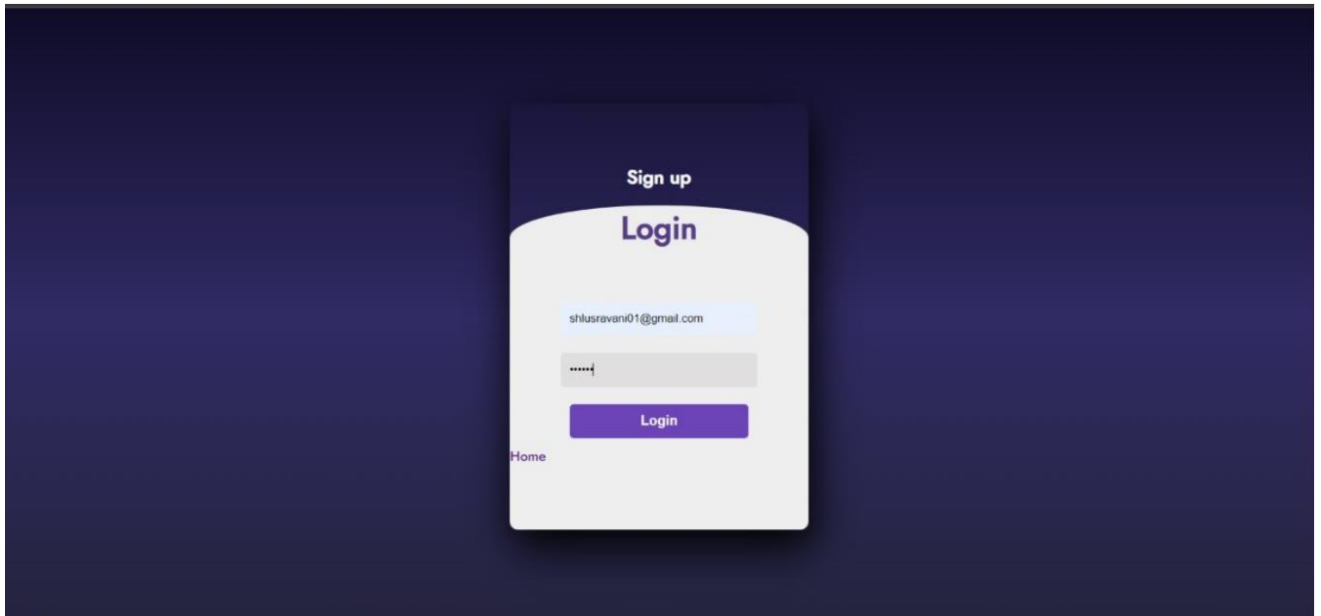
B. SCREENSHOTS

```
Anaconda Prompt (anaconda3) - python app.py
C:\Users\raju\anaconda3\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator GradientBoostingClassifier from version 0.24.1 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
C:\Users\raju\anaconda3\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator DummyClassifier from version 0.24.1 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  warnings.warn(
C:\Users\raju\anaconda3\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator DecisionTreeRegressor from version 0.24.1 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  warnings.warn(
C:\Users\raju\anaconda3\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator GradientBoostingClassifier from version 0.24.1 when using version 1.0.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  warnings.warn(
* Debugger is active!
* Debugger PIN: 945-209-968
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

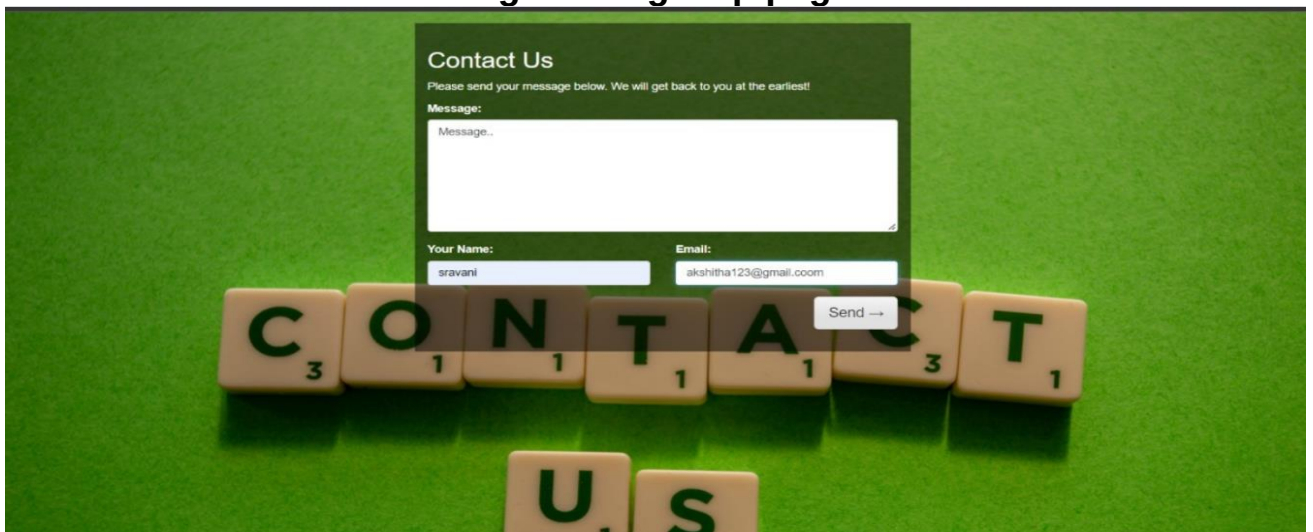
IP Address Generated



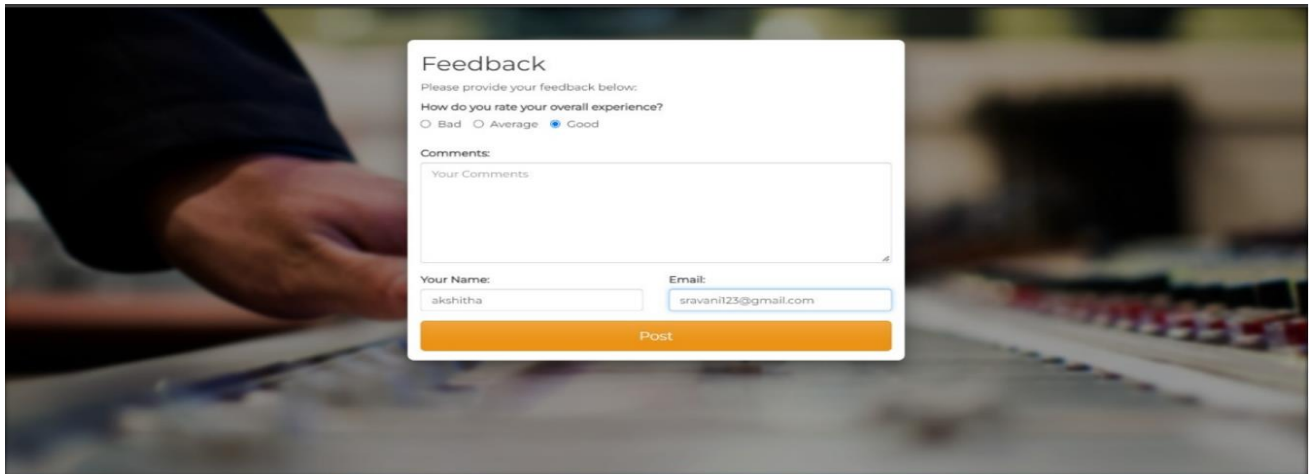
Home Page



Login or Sign-Up page



Contact Us Page



Feedback

Please provide your feedback below:

How do you rate your overall experience?

☐ Bad ☐ Average ☒ Good

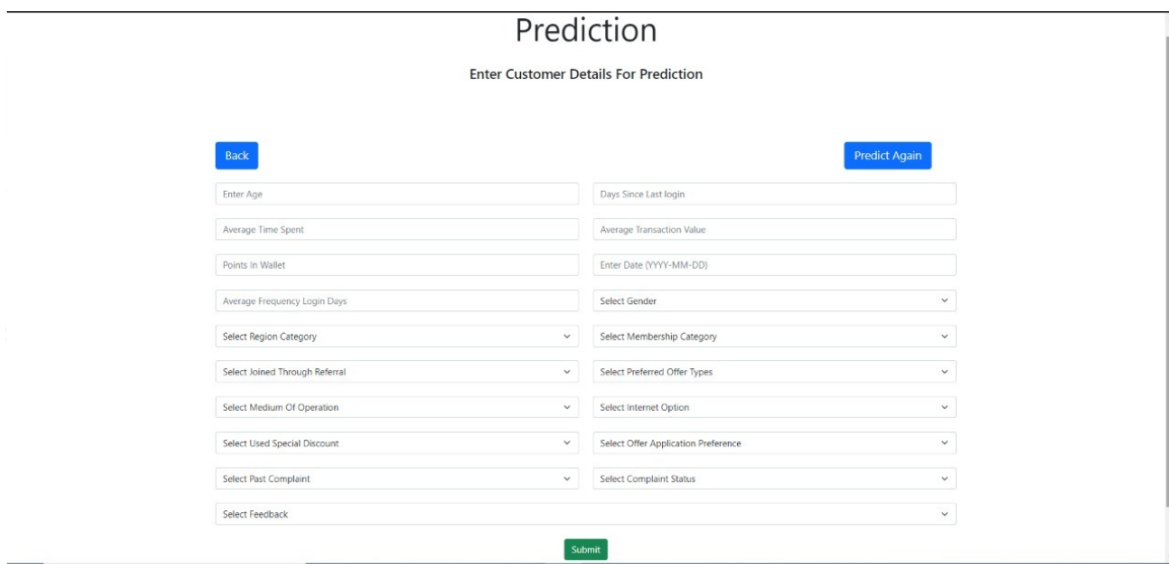
Comments:

Your Comments:

Your Name:

Email:

Feedback Form on products



Prediction

Enter Customer Details For Prediction

Enter Age	Days Since Last login
Average Time Spent	Average Transaction Value
Points In Wallet	Enter Date (YYYY-MM-DD)
Average Frequency Login Days	Select Gender
Select Region Category	Select Membership Category
Select Joined Through Referral	Select Preferred Offer Types
Select Medium Of Operation	Select Internet Option
Select Used Special Discount	Select Offer Application Preference
Select Past Complaint	Select Complaint Status
Select Feedback	

Prediction Page

Prediction

Enter Customer Details For Prediction

Churn Score is [2] churned

Back

Predict Again

30	12
72.52	10000
100	2022-03-02
34	Female
Town	No Membership
No	Credit/Debit Card Offers
Desktop	Mobile_Data
Yes	Yes
No	Unsolved
Reasonable Price	

Submit

After Predicted churn Score