DEMAND EST

AI-POWERED FOOD DEMAND FORECASTER

TEAM LEADER : S.D.INIYAN
TEAM MEMBER 1 :J.ASHVIK
TEAM MEMBER 2 :R.DHARANI
TEAM MEMBER 3 :V.VINOBALA
TEAM MEMBER 4 :SABESAN

1. INTODUCTION

1.1 PROJECT OVERVIEW

A food delivery service has to deal with a lot of perishable raw materials which makes it all, the most important factor for such a company is to accurately forecast daily and weekly demand. Too much inventory in the warehouse means more risk of wastage, and not enough could lead to out-of-stocks - and push customers to seek solutions from your competitors. The replenishment of majority of raw materials is done on weekly basis and since the raw material is perishable, the procurement planning is of utmost importance, the task is to predict the demand for the next 10 weeks.

1.3 PURPOSE

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

The main aim of this project is to create an appropriate machine learning model to forecast the number of orders to gather raw materials for next ten weeks. To achieve this, we should know the information about of fulfilment center like area, city etc., and meal information like category of food sub category of food price of the food or discount in particular week. By using this data, we can use any classification algorithm to forecast the quantity for 10 weeks. A web application is built which is integrated with the model built.

- Prediction of Food Production Using Machine Learning Algorithms of Multilayer Perceptron and ANFIS - One of the limitations of this study is that forecasts for agricultural and livestock production are based only on time series data. Another limitation of this article is the generalization of the finding that the ANFIS model outperforms the MLP model because this finding is limited to the time series data of Iran and the result may differ in data related to another country.
- Food Demand Prediction Using the Nonlinear Autoregressive Exogenous Neural Network - The main limitation of the developed models is the lack of the possibility of analyzing small datasets (below 100 rows of data).
- Food Demand Prediction Using Machine Learning - Refined prediction can be done based on many other factors like cultural habits, religious holiday, consumer preferences etc. This method can be used for predicting work force requirement, automated food ordering based on forecasting results.

2.2. REFERENCE

[1] https://www.irjet.net/archives/V7/i6/IRJET-V7I6686.pdf

[2] https://www.mdpi.com/2077-0472/11/5/408

[3]https://iopscience.iop.org/article/10.1088/1742-6596/1918/4/042012/pdf

[4] https://www.hindawi.com/journals/complexity/2019/9067367/

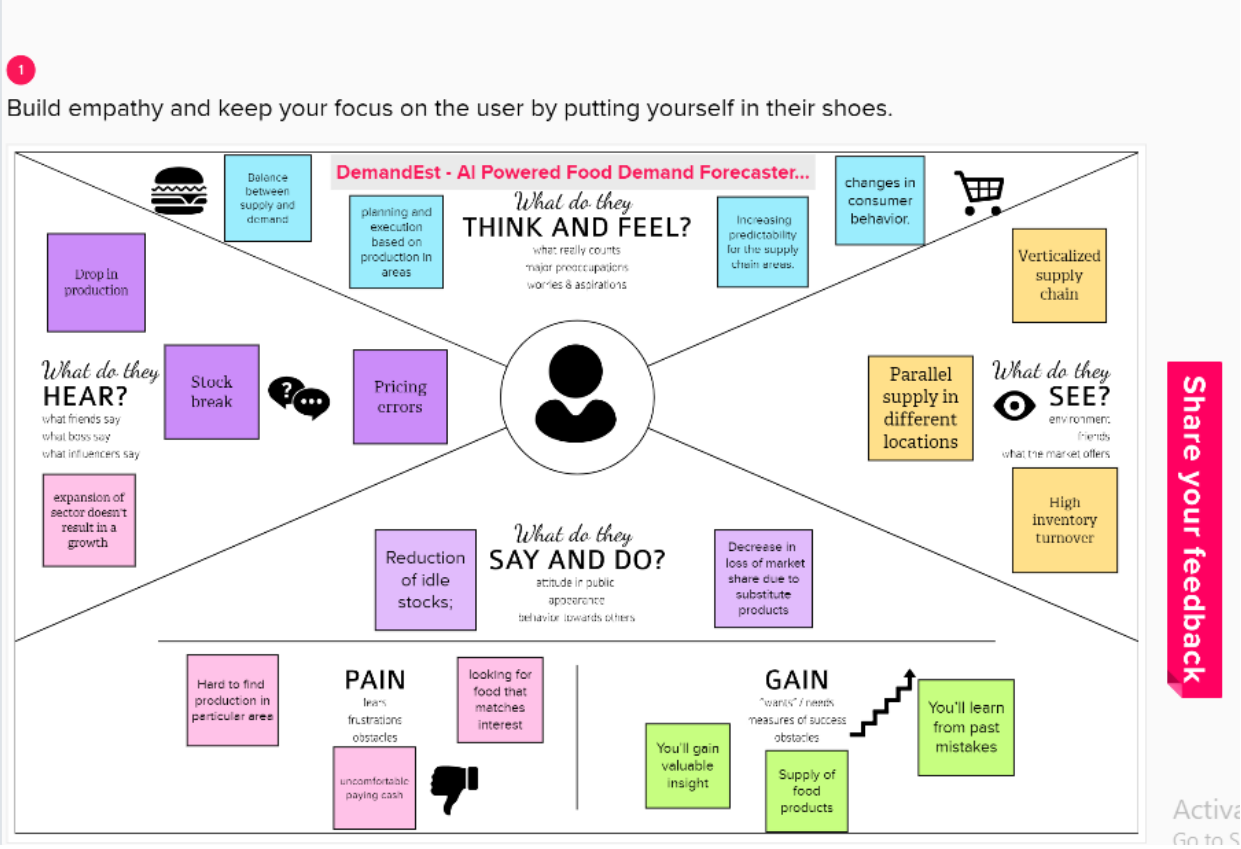
[5] https://ieeexplore.ieee.org/document/9585704

2.3.PROBLEM STATEMENT DEFINITION

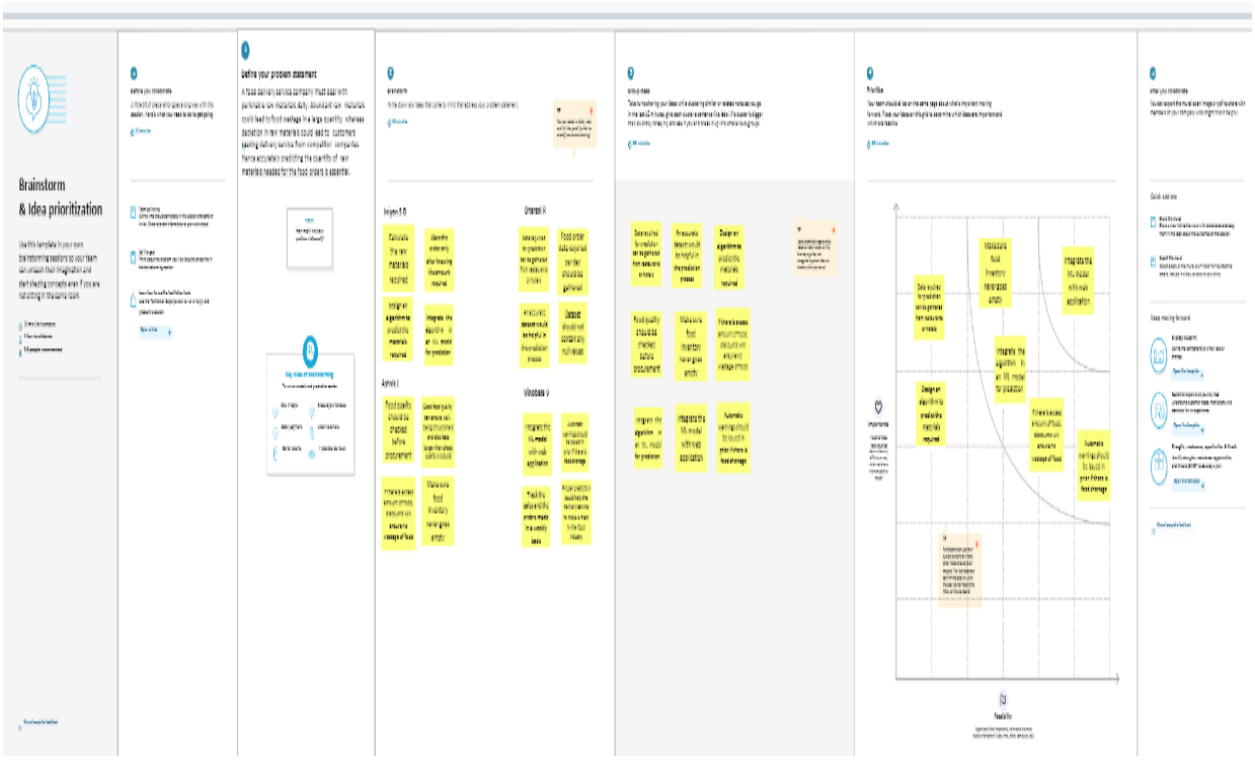
Restaurateur needs a way to overcome inventory shrinkage and wastage so that a proper amount of investment can be made in running a profitable restaurant.

Inventory shrinkage is a major issue that causes shortage of food which should be analyzed and controlled to prevent unavailability of eatery. Hotels are facing many difficulties that include wastage of inventories and foods so we need to predict the proper quantity of raw materials. Share market values play a vital role in restaurants so proper supply chain management can be made to prevent loss in market share. Restaurants are dealing with poor demand prediction so we need to make a adequate prediction by collecting numerous internal data sources from several proprietary systems and historical mission data The gap between the long lead times and when the order is placed creates a considerable amount of uncertainty and thus requires an accurate forecasting model to serve the customers.

IDEATION & PROPOSED SOLUTION



3.2 IDEATION & BRAINSTROMING



3.3 PROPOSED SOLUTION

Project Design Phase-I
Proposed Solution Template

Date	28 October 2022
Team ID	PNT2022TMID42688
Project Name	Demand Est-AI Powered Food Demand Forecaster
Maximum Marks	

Proposed solution:

The main aim of this project is to create an appropriate machine learning model to forecast the number of orders to gather raw materials for next ten weeks.

S. No	Parameter	Description
1	Problem statement (problem to be solved)	• Perishable raw materials must be handled daily by a food delivery service provider. • Therefore, it is crucial to forecast the number of raw materials required for meal orders.
2	Idea / Solution description	• The main objective of food demand forecaster project is to build a machine learning model which uses classification algorithm to forecast the number of orders to gather raw materials for the next 10 weeks. • Appropriate data is gathered from relevant datasets which includes information about food delivery services in any area, meal information, price for each meal and discount of meals in a particular week.
3	Novelty / Uniqueness	• The system automatically updates customer information. • Data is evaluated to forecast the raw materials. • User friendly interface.
4	Social Impact / Customer Satisfaction	• The amount of food wasted in the food sector will be reduced. • Increase in client profits. • Decrease raw material waste.
5	Business Model (financial Benefit)	• After examining the food-related data for each location, it will determine which location was most in demand • Highly profitable. • High inventory turnovers can be made with proper analysis.
6	Scalability of Solution	• The customer gains advantages from the analysis of industry data. • It offers predictions on the day-to-day analysis of the food that is sold.

3.4 PROPOSED SOLUTION FIT

3.4 PROPOSED SOLUTION FIT

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIR3MENTS

Project Design Phase-II

Solution Requirements (Functional & Non-functional)

Date	03 October 2022
Team ID	PNT2022TMID42688
Project Name	Project – DemandEst – AI powered Food Demand Forecaster
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Home page	The user is directed to the home page
FR-4	Sample use	The user would use the web application for calculating sample ideas so that he would get an idea of using it
FR-5	Adding sub-users and creating network	The user could add his co-workers in his application page and form a network.
FR-6	Feedback and support	After deployment, continuous customer support using the feedback

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements:

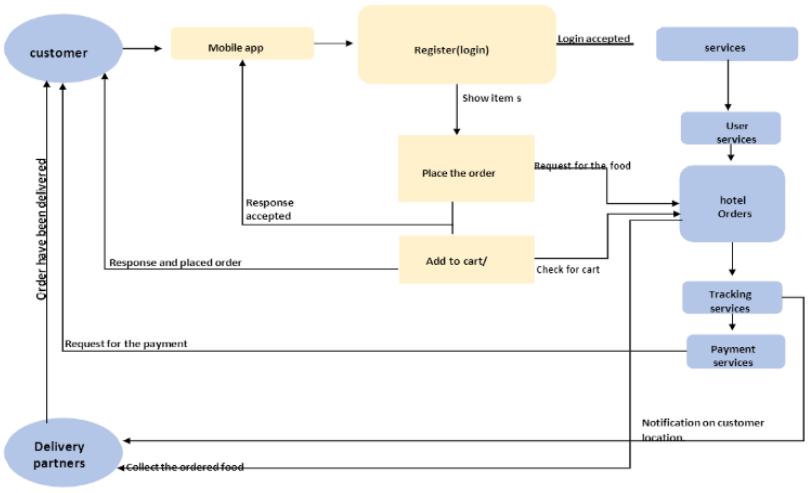
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The web application is required for the people working in food industry for calculating the required amount of food for a particular time period
NFR-2	**Security**	The passwords and emails of the users are stored in the encrypted form. Only if the password matches the encrypted form, the user would able to access their database
NFR-3	**Reliability**	The data stored in the web application is safe as it needs the correct password and verification to access the stored information
NFR-4	**Performance**	The web application is designed in such a way that, no matter how many users using it at a time, the performance of the application remains the same
NFR-5	**Availability**	The web application is available in all platforms
NFR-6	**Scalability**	It works in a fixed scalability

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

Data Flow Diagrams:



5.2 SOLUTION & TECHNICAL ARCHITECTURE SOLUTION ARCHITECTURE

Project Design Phase-I
Solution Architecture

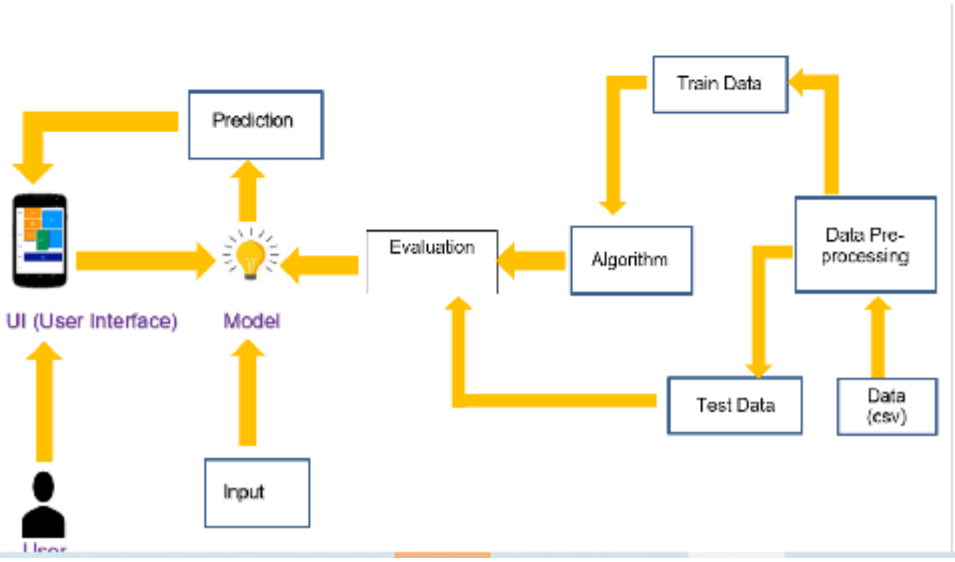
Date	28 October 2022
Team ID	PNT2022TMID42688
Project Name	Project - Demand Est - AI powered FoodDemand Forecaster
Maximum Marks	4 Marks

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gapbetween business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the softwareto project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Solution Architecture Diagram:



TECHNOLOGY ARCHITECTURE

Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	03 October 2022
Team ID	PNT2022TMID42688
Project Name	AI Powered- Food Demand Forecaster
Maximum Marks	4 Marks

Technical Architecture:

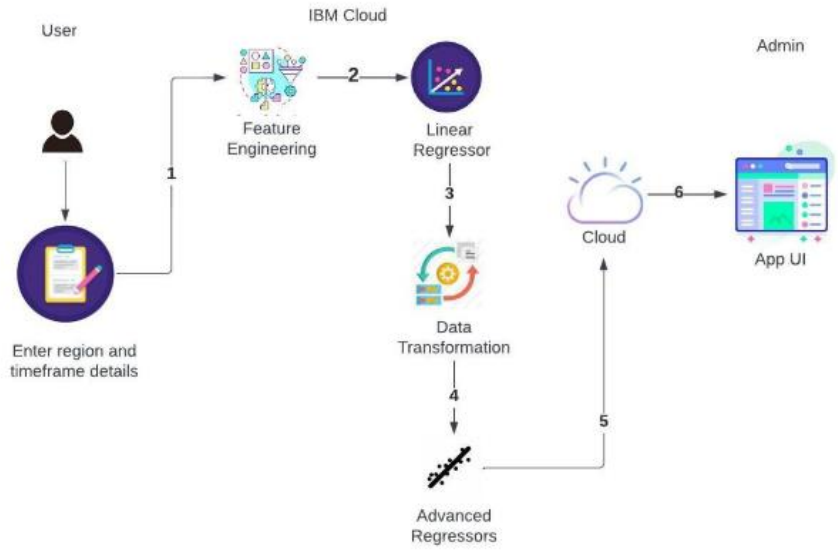


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User Interface for Food Demand Prediction which asks for registration credentials and requirement details.	HTML, CSS, JavaScript
2.	Input and output	Gets inputs(region and timeframe) from user and displays the output using python. It uses get and post http methods in backend for processing.	Python
3.	Libraries	Python libraries like Numpy, pandas, matplotlib, sklearn, seaborn for processing the dataset.	Google Colab
4.	Algorithm	Linear model using linear regression and advanced model using XGBoost, CATBoost and LightBoost regression are implemented.	Regression models and ensemble techniques.
5.	Database	Csv file	
6.	Machine Learning Model	Regression models are used to increase the speed at which data is processed and analysed.	Advanced Regression models.

Table-2: Application Characteristics:

S.No	Characteristics	Description
1.	Scalable Architecture	Highly-customizable infrastructure according to specific customer's needs
2.	Availability	The system is monitored for bugs and so it is highly reliable.
3.	Performance	Feature Engineering extracts valuable features from raw data which significantly improves the efficiency.

5.3 USER STORIES

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Customer Registration.	USN-1	As a user, I can register for the application by entering my email, password, and confirming by OTP.	I can access my account / dashboard	High	Sprint-1
Business staff	Confirmation.	USN-2	As a user, I will receive confirmation email once I have registered for the application.	I can receive confirmation email & click confirm	High	Sprint-1
Tourist people	Accessibility.	USN-3	As a user, I can register for the application through websites, and applications.	I can register & access the dashboard with apps and websites.	High	Sprint-2
College students	Customer access through mail.	USN-4	As a user, I can confirmation through Gmail.	I can confirm the order and get the OTP through e mail.	Medium	Sprint-2
Customer (websites)	Login	USN-5	As a user, I can log into the application by entering email & password and enabling location.	I can log into application to check either precise location or approximate location.	Medium	Sprint-2
	Dashboard	USN - 6	Choosing the menu, Restaurant, and paymentprocess. after receiving the food rating process.	Hazard analysis and critical control point system.	Low	Sprint -1
111	Customer orders	USN - 7	Delivery partner simply tracks the order and let the customer know when it will arrive.	Tracking through GPS make sure whether the given timeslot achieved.	Medium	Sprint-2
Customer Care Executive	Customer order delivery	USN - 8	Door step delivery. easy process to get the order.	Rating on delivery partner and food quality.	Medium	Sprint-2
Administrator	Hotel management, website holders	USN - 9	Choosing the restaurant. Multiple choice for restaurant profile.	Advertising through websites.	Low	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

PNT2022TMID17645

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	22 October 2022
Team ID	PNT2022TMID42688
Project Name	Project – DemandEst - AI Powered Food Demand Forecaster
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-2		USN-4	As a user, I can register for the application through Gmail	2	Medium	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-1	Dashboard	USN-6	As a user, I can access the services and information provided in the dashboard	2	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

AV (Sprint 1) = 7/6 = 1

AV (Sprint 2) = 4/6 = 1

AV (Sprint 3) = 6/6 = 1

AV (Sprint 4) = 2/6 = 1

AV (Total) = 21/24 = 1

PNT2022TMID17645

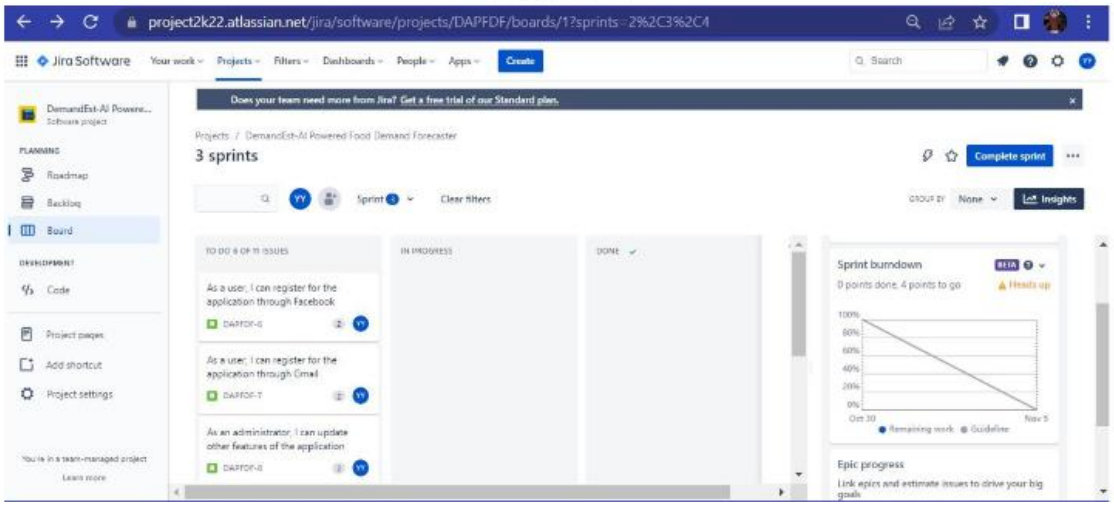
Sprint-1	Login	USN-7	As a user, I can log into the web application and access the dashboard	1	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-4	Helpdesk	USN-8	As a user, I can get the guidance from the customer care	1	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Management	USN-9	As an administrator, I can collect new datasets and keep the model trained	2	High	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-3		USN-10	As an administrator, I can update other features of the application	2	Medium	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-3		USN-11	As an administrator, I can maintain the information about the user	2	Medium	INIYAN S D ASHVIK J DHARANI R VINOBALA V
Sprint-4		USN-12	As an administrator, I can maintain third-party services	1	Low	INIYAN S D ASHVIK J DHARANI R VINOBALA V

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	7	6 Days	24 Oct 2022	29 Oct 2022	7	29 Oct 2022
Sprint-2	4	6 Days	31 Oct 2022	05 Nov 2022	4	05 Nov 2022
Sprint-3	6	6 Days	07 Nov 2022	12 Nov 2022	6	12 Nov 2022
Sprint-4	2	6 Days	14 Nov 2022	19 Nov 2022	2	19 Nov 2022

Burndown Chart:

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



7 CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 FEATURE 1

In Demand Est AI-Powered Food Demand Forecaster the datasets are collected datasets are preprocessed and splitted into train and test data. Models are evaluated and the output is predicted using the model.

Data Preprocessing

Importing the libraries

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
import sklearn as sk
```

Reading the dataset

In [4]:
```
from google.colab import files
uploaded=files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving test.csv to test (1).csv
```

In [8]:
```
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
```

Exploratory Data Analysis

In [9]:
```
train.head()
```

Out[9]:

	id	week	center_id	meal_id	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders
0	1379560	1	55	1885	136.83	152.29	0	0	177
1	1466964	1	55	1993	136.83	135.83	0	0	270
2	1346989	1	55	2539	134.86	135.86	0	0	189
3	1338232	1	55	2139	339.50	437.53	0	0	54
4	1448490	1	55	2631	243.50	242.50	0	0	40

Out[10]:

	id	week	center_id	meal_id	checkout_price	base_price	emailer_for_promotion	homepage_featured
0	1028232	146	55	1885	158.11	159.11	0	0
1	1127204	146	55	1993	160.11	159.11	0	0
2	1212707	146	55	2539	157.14	159.14	0	0
3	1082698	146	55	2631	162.02	162.02	0	0
4	1400926	146	55	1248	163.93	163.93	0	0

In [11]:
```
train.info()
```

```
RangeIndex: 456548 entries, 0 to 456547
Data columns (total 9 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   id                    456548 non-null  int64
 1   week                  456548 non-null  int64
 2   center_id             456548 non-null  int64
 3   meal_id               456548 non-null  int64
 4   checkout_price        456548 non-null  float64
 5   base_price            456548 non-null  float64
 6   emailer_for_promotion 456548 non-null  int64
 7   homepage_featured     456548 non-null  int64
 8   num_orders            456548 non-null  int64
dtypes: float64(2), int64(7)
memory usage: 31.3 MB
```

In [12]:
```
train['num_orders'].describe()
```

Out[12]:
```
count    456548.000000
mean        261.872760
std         395.922798
min          13.000000
25%          54.000000
50%         136.000000
75%         324.000000
max       24299.000000
Name: num_orders, dtype: float64
```

Checking for Null Values

In [13]:
```
train.isnull().sum()
```

Out[20]:

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type	op_area	Unnamed:
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C	2.0	NaN
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C	2.0	NaN
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C	2.0	NaN
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C	2.0	NaN
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C	2.0	NaN

In [21]:
```
cols=trainfinal.columns.tolist()
print(cols)
```

```
['id', 'week', 'checkout_price', 'base_price', 'emailer_for_promotion', 'homepage_featured', 'num_orders', 'category', 'cuisine', 'city_code', 'region
_code', 'center_type', 'op_area', 'Unnamed: 5']
```

In [22]:
```
cols=cols[:2]+cols[9:]+cols[7:9]+cols[2:7]
print(cols)
```

```
['id', 'week', 'city_code', 'region_code', 'center_type', 'op_area', 'Unnamed: 5', 'category', 'cuisine', 'checkout_price', 'base_price', 'emailer_for
_promotion', 'homepage_featured', 'num_orders']
```

In [23]:
```
trainfinal=trainfinal[cols]
```

In [24]:
```
trainfinal.dtypes
```

Out[24]:
```
id                      int64
week                    int64
city_code               int64
region_code             int64
center_type            object
op_area               float64
Unnamed: 5            float64
category               object
cuisine                object
checkout_price        float64
base_price            float64
emailer_for_promotion   int64
```


Reading and Merging .csv files

In [17]:
```
meal_info = pd.read_csv("meal_info.csv")
center_info = pd.read_csv("fulfilment_center_info.csv")
```

In [18]:
```
trainfinal = pd.merge(train, meal_info, on="meal_id", how="outer")
trainfinal = pd.merge(trainfinal, center_info, on="center_id", how="outer")
```

In [19]:
```
trainfinal.head()
```

Out[19]:

	id	week	center_id	meal_id	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type
0	1379560	1	55	1885	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C
1	1018704	2	55	1885	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C
2	1196273	3	55	1885	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C
3	1116527	4	55	1885	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C
4	1343872	5	55	1885	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C

Dropping Columns

In [20]:
```
trainfinal=trainfinal.drop(['center_id','meal_id'],axis=1)
trainfinal.head()
```

Out[20]:

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type	op_area	Unnamed:
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C	2.0	NaN
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C	2.0	NaN
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C	2.0	NaN
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C	2.0	NaN
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C	2.0	NaN

Label Encoding

In [25]:
```
from sklearn.preprocessing import LabelEncoder
```

In [26]:
```
lb1=LabelEncoder()
trainfinal['center_type']=lb1.fit_transform(trainfinal['center_type'])
lb2=LabelEncoder()
trainfinal['category']=lb1.fit_transform(trainfinal['category'])
lb1=LabelEncoder()
trainfinal['cuisine']=lb1.fit_transform(trainfinal['cuisine'])
```

In [27]:
```
trainfinal.head()
```

Out[27]:

	id	week	city_code	region_code	center_type	op_area	Unnamed: 5	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders
0	1379560	1	647	56	2	2.0	NaN	0	3	136.83	152.29	0	0	177
1	1018704	2	647	56	2	2.0	NaN	0	3	135.83	152.29	0	0	323
2	1196273	3	647	56	2	2.0	NaN	0	3	132.92	133.92	0	0	96
3	1116527	4	647	56	2	2.0	NaN	0	3	135.86	134.86	0	0	163
4	1343872	5	647	56	2	2.0	NaN	0	3	146.50	147.50	0	0	215

In [28]:
```
trainfinal.shape
```

Out[28]:
```
(456548, 14)
```

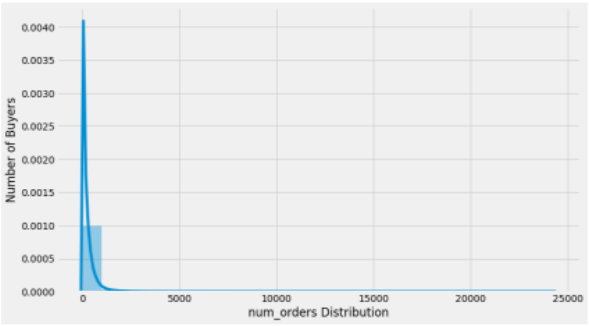
Data Visualisation

In [29]:
```
plt.style.use('fivethirtyeight')
plt.figure(figsize=(12,7))
sb.distplot(trainfinal.num_orders,bins=25)
plt.xlabel("num_orders")
plt.ylabel("Number of Buyers")
plt.xlabel("num_orders Distribution")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function fo
r histograms).
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function fo
r histograms).
  warnings.warn(msg, FutureWarning)
```

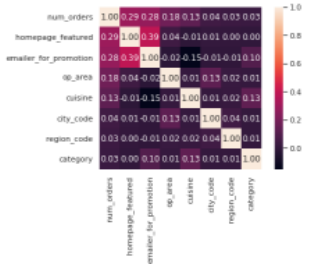
Out[29]:
```
Text(0.5, 0, 'num_orders Distribution')
```



In [30]:
```
trainfinal2=trainfinal.drop(['id'],axis=1)
correlation=trainfinal2.corr(method='pearson')
columns=correlation.nlargest(8,'num_orders').index
columns
```

Out[30]:
```
Index(['num_orders', 'homepage_featured', 'emailer_for_promotion', 'op_area',
       'cuisine', 'city_code', 'region_code', 'category'],
      dtype='object')
```

In [31]:
```
correlation_map=np.corrcoef(trainfinal[columns].values.T)
sb.set(font_scale=1.0)
heatmap=sb.heatmap(correlation_map, cbar=True, annot=True, square=True, fmt='.2f', yticklabels=columns.values, xticklabels=columns.values)
```



Splitting the dataset into Dependent and Independent variables

In [32]:
```
features=columns.drop(['num_orders'])
trainfinal3=trainfinal[features]
x=trainfinal3.values
y=trainfinal['num_orders'].values
```

In [33]:
```
trainfinal3.head()
```

Out[33]:

	homepage_featured	emailer_for_promotion	op_area	cuisine	city_code	region_code	category
0	0	0	2.0	3	647	56	0
1	0	0	2.0	3	647	56	0
2	0	0	2.0	3	647	56	0
3	0	0	2.0	3	647	56	0
4	0	0	2.0	3	647	56	0

Split the Dataset into Train and Test Set

In [34]:
```
from sklearn.model_selection import train_test_split
x_train, x_val, y_train, y_val=train_test_split(x,y,test_size=0.25)
```

Model Building

Train and Test Model Algorithms

In [35]:
```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.linear_model import ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import GradientBoostingRegressor
```

In [36]:
```
LR = LinearRegression()
LR.fit(x_train,y_train)
y_pred = LR.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 129.0834122813798
```

In [37]:
```
L=Lasso()
L.fit(x_train,y_train)
y_pred=LR.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 129.0834122813798
```

In [38]:
```
EN=ElasticNet()
EN.fit(x_train,y_train)
y_pred=EN.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 130.67395928486418
```

In [39]:
```
DT=DecisionTreeRegressor()
DT.fit(x_train,y_train)
y_pred=DT.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 62.85258748656859
```

In [40]:
```
KNN=KNeighborsRegressor()
KNN.fit(x_train,y_train)
y_pred=KNN.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 66.934119819693
```

In [41]:
```
GB=GradientBoostingRegressor()
GB.fit(x_train,y_train)
y_pred=GB.predict(x_val)
y_pred[y_pred<0]=0
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 98.13731815769488
```


```
from sklearn import metrics
print('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val,y_pred)))
```

```
RMSLE: 98.13731815769488
```


Save the Model

In [42]:
```
import pickle
pickle.dump(DT,open('fdemand.pkl','wb'))
```

In [42]:
```
import pickle
pickle.dump(DT,open('fdemand.pkl','wb'))
```

Predicting the output using the model

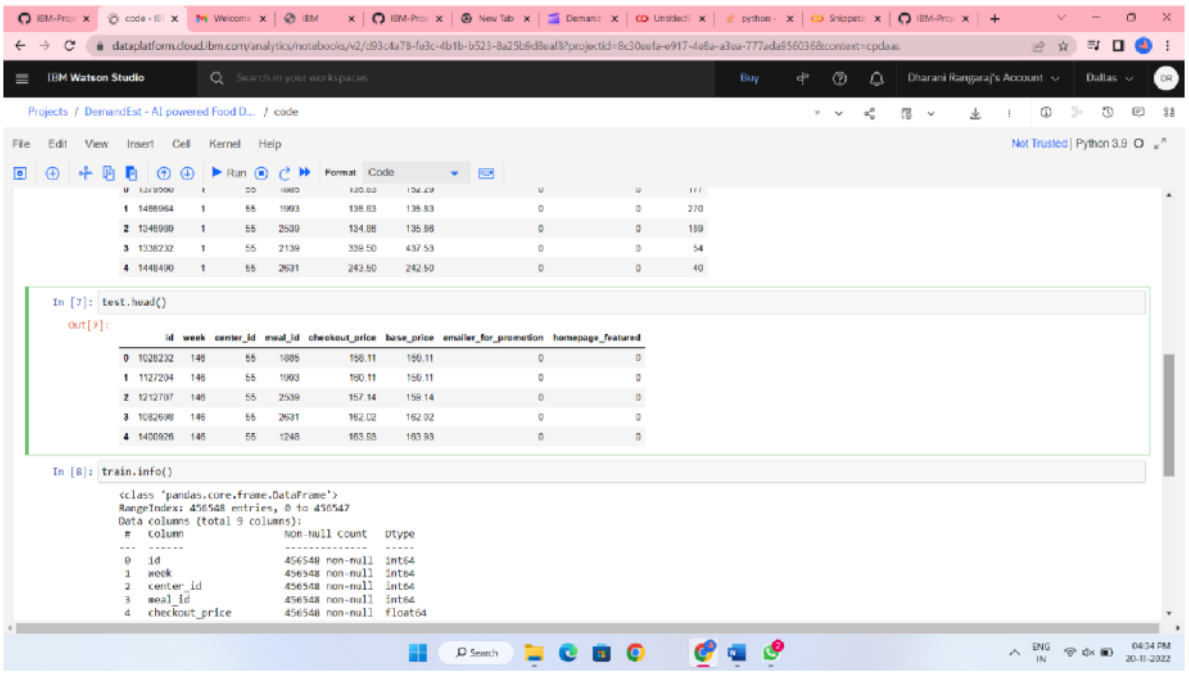
In [43]:
```
testfinal=pd.merge(test,meal_info, on="meal_id", how="outer")
testfinal=pd.merge(testfinal, center_info, on="center_id", how="outer")
testfinal=testfinal.drop(['meal_id','center_id'],axis=1)
tcols=testfinal.columns.tolist()
tcols=tcols[:2]+tcols[8:]+tcols[6:8]+tcols[2:6]
testfinal=testfinal[tcols]
lb1=LabelEncoder()
testfinal['center_type']=lb1.fit_transform(testfinal['center_type'])
lb2=LabelEncoder()
testfinal['category']=lb1.fit_transform(testfinal['category'])
lb1=LabelEncoder()
testfinal['cuisine']=lb1.fit_transform(testfinal['cuisine'])
x_test=testfinal[features].values
```

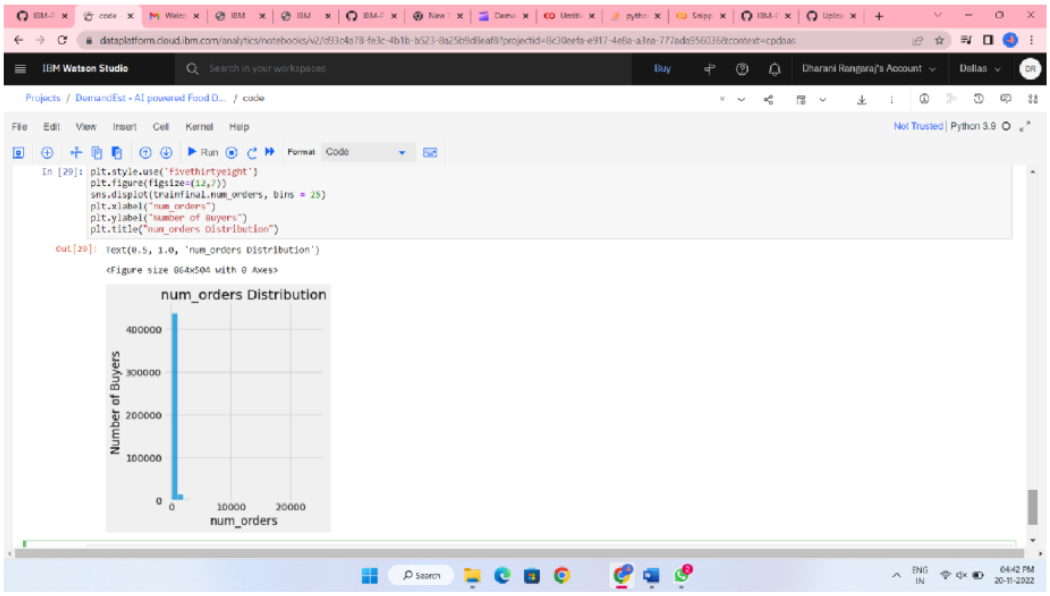
In [44]:
```
pred=DT.predict(x_test)
pred[pred<0]=0
submit=pd.DataFrame({'id':testfinal['id'],'num_orders':pred})
```

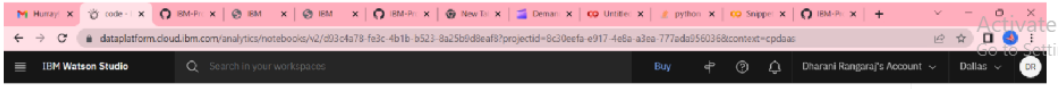
In [45]:
```
submit.to_csv("submission.csv",index=False)
submit.describe()
```

Out[45]:

	id	num orders
count	3.257300e+04	32573.000000
mean	1.248476e+06	262.788519
std	1.441580e+05	366.894816
min	1.000085e+06	15.446429
25%	1.123969e+06	64.892430
50%	1.247296e+06	145.723684
75%	1.372971e+06	323.404984
max	1.499996e+06	6124.523810







8.TESTING

8.1 TEST CASES

				Date	03-Nov-22			
				Team ID	PNT2022TMID40397			
				Project Name	Project - DemandEst AI Powered Food Demand Forecaster			
				Maximum Marks	4 marks			
Test case ID	**Feature Type**	**Component**	**Test Scenario**	**Test data**	**Steps To Execute**	**Expected Result**	**Actual Result**	**Status**	**Executed By**
HomePage_TC_001	UI	Home Page	Verify user is able to view the landing page of the prediction webpage	http://127.0.0.1:5000/	1.Enter URL and click go	The home page should be displayed	Working as expected	Pass	Jayadevi M
HomePage_TC_002	Functional	Home Page	Verify the user is able to click the predict button available in home page to go to the predict page	http://127.0.0.1:5000/	1.Enter URL and click go	User should navigate to the predict page by clicking predict button	Working as expected	Pass	Bhavatharani B
PredictPage_TC_003	UI	Predict page	Verify user is able to view the prediction page	http://127.0.0.1:5000/pred	1.Enter the URL and click go	Predict page should have the UI components such as emailer for promotion, pincode,cuisine type , predict button etc	Working as expected	Pass	Seetha A
PredictPage_TC_004	Functional	Predict page	Verify user is able to enter the credentials to predict the orders	http://127.0.01.:5000/pred	1.Enter URL and click go	User should be able to enter credentials in respective fields . And able to click predict button	Working as expected	Pass	Hemalatha B R
PredictPage_TC_005	Functional	Predict page	Verify user is able to click the predict button and get the prediction of number of orders	http://127.0.0.1:5000/predict	1.Enter URL and click go	Application should show the number of orders after clicking the predict button in predict page.	Working as expected	Pass	Jayadevi M

9. RESULTS

9.1 PERFORMANCE MATRIX

Project Planning Phase

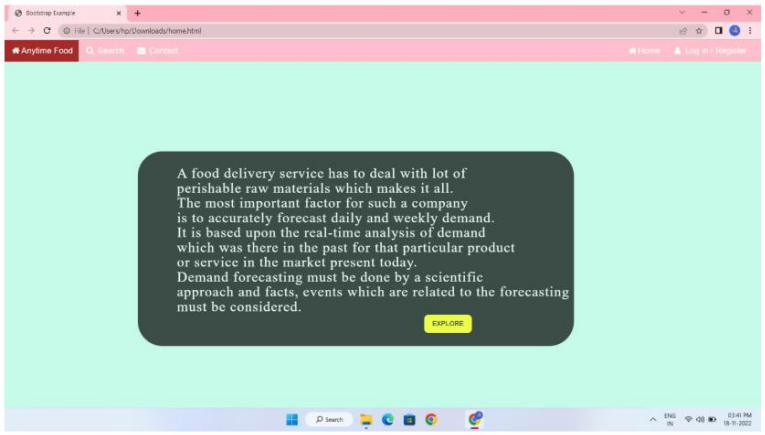
Milestone and Activity list

Date	02 November 2022
Team ID	PNT2022TMID42688
Project Name	DemandEst - AI powered Food Demand Forecaster
Maximum Marks	8 Marks

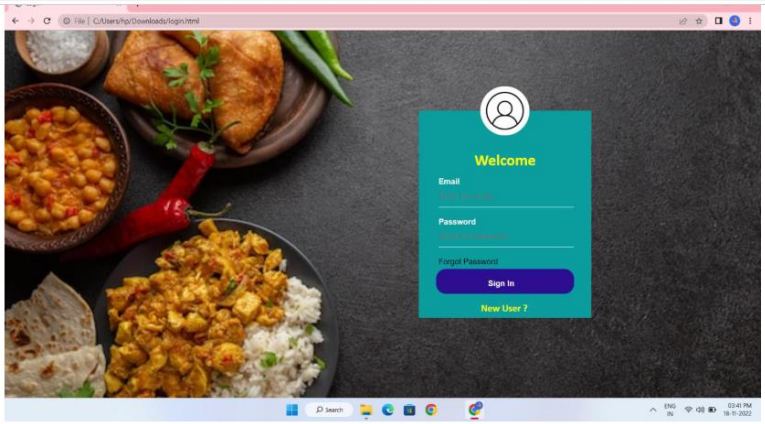
Completed Tasks:

MILESTONES	ACTIVITY	DESCRIPTION
Ideation phase	Literature survey	Literature survey on selected project and gathering information.
	Empathy map	Prepare empathy map to capture the user pains and gains, prepare a list of problem statement.
	Ideation	Organizing the brainstorming session and prioritize the top three ideas based on feasibility hand importance.
Project design phase-1	Proposed solution	Prepare proposed solution document which includes novelty, feasibility of ideas, business model, social impact, scalability of solution.
	Problem solution fit	Prepare problem solution fit Documents.
	Solution architecture	Prepare solution architecture

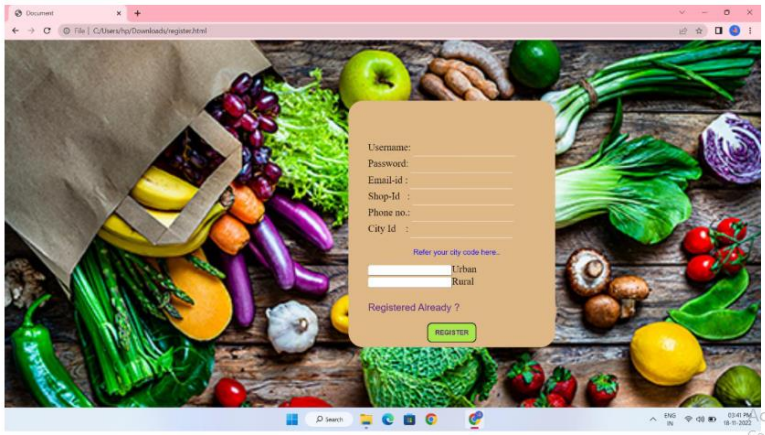
Team ID - PNT2022TMID42688



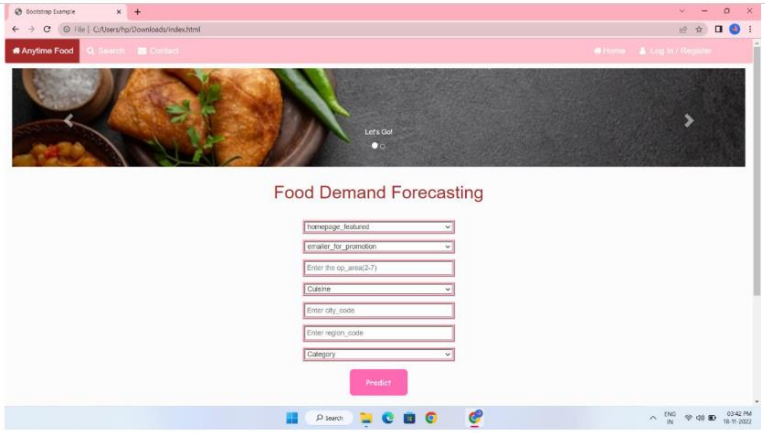
Home Page(home.html)



Login page(login.html)



Register page(register.html)



Predicting Page(index.html)

10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

- In this paper we presented penalized regression method, Bayesian Linear Regression K-nearest Neighbour, Decision tree approach as a food demand method. As we go through different algorithm for prediction the accuracy rate keeps on improving.
- There was not big difference other than precision rate of forecasting. XG boost is a decision-based boosting algorithm which is used for increasing the accuracy rate.
- These models contribute to improve the production plans of organizations, reduce product losses and increase customer satisfaction.
- The model was able to predict the number of orders for requested and given next 10 weeks of input data as a test validation data.

DISADVANTAGES

- There is a huge demand and need of Data Science in Food Industry as meals are one of important basic need for a human being.
- Reducing food wastage is an challenge and absolute way of making profit for the food delivery or manufactures.
- Manufacturing of raw products generally depends upon features like price, discount, area and many things.

11.CONCLUSION

This project presents a novel application to accurately forecast the demand for selected foods. The proposed approach can be practically applied as a component of a restaurant's intelligent management system . It can support the rational control of food inventory and production while reducing waste and costs in the supply chain.

Demand prediction plays a crucial role in planning operations for restaurant's management. Having a reliable estimation for a menu items future demand is the basis for other analysis. Various forecasting techniques have been developed, each one with its particular advantages and disadvantages compared to other approaches

12.FUTURE WORK

For further future aspects many more terms need to be considered like customer reach, feedback,cultural habits,religious holiday, consumer preferences as well as use of Neural Networks would lead to more understanding and predicting accurate number of stocks and customer reach considering many more features .

In future, this method can be used for predicting work force requirement, automated food ordering based on forecasting results. As a result, the system will push the company toward effective distribution of meals in the best possible way.

13. APPENDIX

SOURCE CODE

1.Code.py

```
Import panda as np
import numpy as np
import pickle
import os
from flask import Flask,request, render_template
app=Flask(__name__,template_folder="templates")
@app.route('/file:///D:/Nalaiyathiran%20Project/Pre-processing%20-%20development/Flask/Templetes/home.html',
methods=['GET'])
def index():
  return render_template('home.html')
@app.route('/file:///D:/Nalaiyathiran%20Project/Pre-processing%20-%20development/Flask/Templetes/home.html',
methods=['GET'])
def about():
  return render_template('home.html')
@app.route('/file:///D:/Nalaiyathiran%20Project/Pre-processing%20-%20development/Flask/Templetes/upload.html',
methods=['GET'])
def page():
  return render_template('upload.html')
@app.route('/y_predict', methods=['GET','POST'])
def y_predict():
  print("[INFO] loading model...")
  model = pickle.loads(open('fdemand.pkl',"rb").read())
  input_features = [float(x) for x in request.form.values()]
  features_value = [np.array(input_features)]
  print(features_value)
  features_name = ['homepage_featured', 'emailer_for_promotion','op_area','cuisine','city_code','region_code','category']
  prediction = model.predict(features_value)
  output=prediction[0]
  print(output)
  return render_template('upload.html', prediction_text=output)
if __name__ == '__main__':
  app.run(host='0.0.0.0',port=8000, debug=False)
```

2.app.py

```
Import pandas as np
import numpy as np
import pickle
import os
from flask import Flask,request, render_template
app=Flask(__name__,template_folder="templates")
@app.route('/', methods=['GET'])
def index():
    return render_template('home.html')
@app.route('/home', methods=['GET'])
def about():
    return render_template('home.html')
@app.route('/pred',methods=['GET'])
def page():
    return render_template('upload.html')
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    print("[INFO] loading model...")
    model = pickle.load(open('fdemand.pkl', 'rb'))
    input_features = [float(x) for x in request.form.values()]
    features_value = [np.array(input_features)]
    print(features_value)

    features_name = ['homepage_featured', 'emailer_for_promotion', 'op_area', 'cuisine',
      'city_code', 'region_code', 'category']
    prediction = model.predict(features_value)
    output=prediction[0]
    print(output)
    return render_template('upload.html', prediction_text=output)

if __name__ == '__main__':
     app.run(debug=False)
```

3.home.html

```
<!DOCTYPE
```

```
html>
                    <html lang="en">
                    <head>
                    <meta name="viewport" content="width=device-width, initial-scale=1">
                    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
                    awesome.min.css">
                      <title>Bootstrap Example</title>
                      <meta charset="utf-8">
                      <meta name="viewport" content="width=device-width, initial-scale=1">

                      <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
                      <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
                     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-
                    awesome.min.css">
                    <meta http-equiv="X-UA-Compatible" content="IE=edge">
                    <meta name="viewport" content="width=device-width, initial-scale=1.0">
                    <title>Document</title>
                    <style>
                    :root {
                    --icon: #b@bfd8;
                    }
                    .navbar {
                      width: 100%;
                      background-color:pink;
                    overflow: hidden;
                      position: fixed; /* Set the navbar to fixed position */
                      top: 0; /* Position the navbar at the top of the page */
                    left:0;
                    }
                    .navbar a {
                      float: left;
                      padding: 12px;
                      color: white;
                      text-decoration: none;
                      font-size: 17px;
                    }
                    .navbar a:hover {
                      background-color: #000;
                    }
                    .active1 {
                      background-color: brown;
                    }
                    @media screen and (max-width: 500px) {
```

```
  .navbar a {
    float: none;
    display: block;
  }
}
span{
background-color:yellow;
 font-size: 40px;
font-weight: bolder;
border-radius: 10px;
 margin-left: 200px;
 margin-bottom: 500px;
}
body{
margin:50px;
padding:20px;
background: url("chicken.png");
 background-size:100%;
 background-repeat: no-repeat;
 font-family: sans-serif;
background-color: rgb(198, 251, 233);
}
p{
margin-left:200px;
 margin-top:100px;
}
.content .par{
padding-left: 80px;
padding-bottom: 25px;
font-family:Garamound;
 letter-spacing: 1.2px;
 line-height: 30px;
margin-right: 310px;
color:white;
padding-top:30px;
font-size:29px;
background-color: black;
 border-radius: 50px;
opacity:70%;
}
button{
width:100px;
 height: 40px;
```

```
border-radius: 10px;
background-color:yellow;
 color:black;
margin-left: 500px;
 font-weight: bolder;
cursor: pointer;
}
</style>
</head>
<body>
<div class="navbar">
<a class="active1" href="#"><i class="fa fa-fw fa-home"></i>Anytime Food</a>

  <a href="#"><i class="fa fa-fw fa-search"></i> Search</a>
  <a href="#"><i class="fa fa-fw fa-envelope"></i> Contact</a>
<a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a
href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#">
<a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a
href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#">
<a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a href="#"><a
href="#"><a href="#"><a href="#"><a href="#"><a href="#">
 <a href="#"><i class="fa fa-fw fa-home"></i>Home</a>
  <a href="login.html" class="left"><i class="fa fa-fw fa-user"></i> Log In / Register</a>

</div>
<br><br><br>
<div class="content">
<p class="par">A food delivery service has to deal with lot of <br> perishable raw materials which makes
it all.<br>
The most important factor for such a company<br>
is to accurately forecast daily and weekly demand.<br> It is based upon the real-time analysis of
demand<br>
which was there in the past for that particular product<br> or service in the market present today. <br>
Demand forecasting must be done by a scientific<br>
approach and facts, events which are related to the forecasting must be
considered.
<a href="index.html"><button>EXPLORE</button></a>
</p>
</body>
</html>
</div>
```

4.login.html

```
<!DOCTYPE
html>
                    <html lang="en">
                    <head>
                    <meta charset="UTF-8">
                    <meta http-equiv="X-UA-Compatible" content="IE=edge">
                    <meta name="viewport" content="width=device-width, initial-scale=1.0">
                    <title>Login</title>
                    <style>
                    body{
                    margin:50px;
                    padding:20px;
                    background: url("food1.jpg");
                     background-size:100%;
                    background-repeat: no-repeat;
                     font-family: sans-serif;
                    }
                    .loginbox{
                    position: absolute;
                    top:50%;
                    right:170px;
                    transform:translate(-50%,-50%);
                    width:350px;
                    height:420px;
                     padding:80px 40px;
                     box-sizing:border-box;
                    background: rgb(11, 157, 157);
                    color:#fff;
                    }
                    h2{
                    margin:0;
                     padding:0 0 15px;
                    color:yellow;
                    font-size:32px;
                     text-align: center;
                    font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
                    }


                    .loginbox p{
                    padding:0;
                    margin:0;
                    font-weight: bold;
```

```
color:#fff;
}
.loginbox input{ width:100%;
margin-bottom: 20px;
}
.loginbox input[type="text"], .loginbox input[type="password"]
{
border:none;
border-bottom: 1px solid #fff;
 background: transparent;
 outline: none;
height:40px; color:#fff;
font-size:16px;
}
#sign
{
border:none;
 outline: none;
 height: 30px;
color:#fff;
font-size: 16px;
background: rgb(43, 15, 144);
cursor:pointer;
border-radius: 20px;
 position: absolute;
top:55%;
left:10%; right: 10%;
padding:20px 80px;
padding-left:70px;
margin-top:90px;
 padding-bottom:30px;
box-sizing:border-box;
}
#sign1
{
color:yellow;
 font-size: 16px;
cursor:pointer;
font-size:20px;

font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
position: absolute; padding-left:40px;
margin-top:80px;
```

```
 padding:10px 50px;
 box-sizing:border-box;

}
a{
color: black;
text-decoration: none;
}
.user{
width:100px;
height: 100px;
overflow: hidden;
position: absolute;
top:calc(-100px/2);
left: calc(50% - 50px);
border-radius: 50px;
}
#inp{
color:rgb(236, 75, 75);
}
</style>

</head>
<body>
<div class="loginbox" >
<img src="user.jpg" class="user">
<h2>Welcome</h2>
<form>
<p>Email</p>
<input type="text" id="inp" name="" placeholder="Enter the email" onclick="checkemail()">
<div id="d1"></div>
<p>Password</p>
<input type="password" name="" placeholder="Enter the password">
<p><a href="home.html" id="sign">        Sign In</a></p>
<p><a href="register.html" id="sign1">        New User
?</a></p>
<a href="#">Forgot Password</a>
</form>
</div>
</body>
</html>
```

5.register.html

```
<!DOCTYPE
html>
                <html lang="en">
                <head>
                <meta charset="UTF-8">
                <meta http-equiv="X-UA-Compatible" content="IE=edge">
                <meta name="viewport" content="width=device-width, initial-scale=1.0">
                <title>Document</title>
                <style>
                body{
                background-color:sandybrown; background-image:url("food4.jpg");
                align:center;
                background-size:100%;
                }
                p{
                padding-top: 7px; font-size: 20px;
                font-family: Arial, Helvetica, sans-serif;
                }
                .loginbox{
                position: absolute; top:50%;
                margin-left: 900px; transform:translate(-50%,-50%); width: 420px;
                height:500px; padding:80px 40px; box-sizing:border-box; color:#fff;
                background-color:burlywood; background-size: 90%; border-radius: 30px;
                font-size: 14px;
                }
                .loginbox input[type="text"], .loginbox input[type="password"]
                {
                border:none;
                border-bottom: 1px solid #fff; background: transparent;

                outline: none; height:30px; color:crimson; font-size:16px; padding-left:10px;
                }
                span{
                font-size: 20px;
                color:black;
                }
                button{
                width:100px; height: 40px; border-radius: 10px;
                background-color:rgb(169, 231, 75); color:black;
                margin-left: 120px;
                font-weight: bolder; cursor: pointer;
                }
                #sp{
```

```
margin-top:30px; margin-bottom: 30px;
}
#city{
font-size: 15px; padding-top:5px;
}
a{
text-decoration: none;
}
</style>
</head>
<body>
<div class="loginbox" >
<form>
<span>Username: </span>

<input type="text" id="inp" name="" >
<span>Password: </span>
<input type="password" name="" ><br>
<span>Email-id : </span>
<input type="text" id="inp" name="" ><br>
<span>Shop-Id   : </span>
<input type="text" id="inp" name="" ><br>
<span>Phone no.:</span>
<input type="text" id="inp" name="" ><br>
<span>City Id    : </span>
<input type="text" id="inp" name="" ><br>
<p id="city"><a href="cities.html">           
         
Refer your city code here..</a></p>
<input type="radio” value="Rural" id="radio" name=""><span
id="sp">Urban</span><br>
<input type="radio” value="urban" id="radio" name=""><span
id="sp">Rural</span><br>
<p><a href="index.html"> Registered Already ?</a></p>
<button><a href="login.html">REGISTER</a></button>
</form>
</div>
</body>
</html>
```

GITHUB LINK :

https://github.com/IBM-EPBL/IBM-Project-53347-1661337301

DEMO LINK :

VIDEO: https://clipchamp.com/watch/7LDp40H1doW