

Project report on

Food Demand Forecasting for Food Delivery Company

Submitted to - SmartInternz

Prepared by

Nagaraj S (20BCE1187)

T. DEVENDHAR 20BPS1041

K. ESHWAR 20BCE1963

S.V. AKHIL 20BPS1146

INTRODUCTION

OVERVIEW

The primary challenge for a food delivery service is accurately forecasting daily and weekly demand due to the perishable nature of raw materials. Maintaining the right balance of inventory is crucial as excess inventory increases the risk of wastage, while insufficient inventory can lead to out-of-stock situations and drive customers to competitors. Since most raw materials are replenished on a weekly basis, effective procurement planning is vital. Therefore, the task at hand is to predict the demand for the next 10 weeks.

PURPOSE

The main objective of this project is to build an appropriate machine learning model to forecast the number of orders required to buy raw materials over the next 10 weeks. To do this, we need to know details about the meal fulfillment facility, such as the neighbourhood, city, etc., as well as details about the meals themselves, such as the food category, food subclass, price of the food, or discount during a certain week. Any classification system that makes use of this data allows us to predict the quantity for 10 weeks. For this, a web application that is integrated with the model is made.

LITERATURE SURVEY

EXISTING PROBLEM

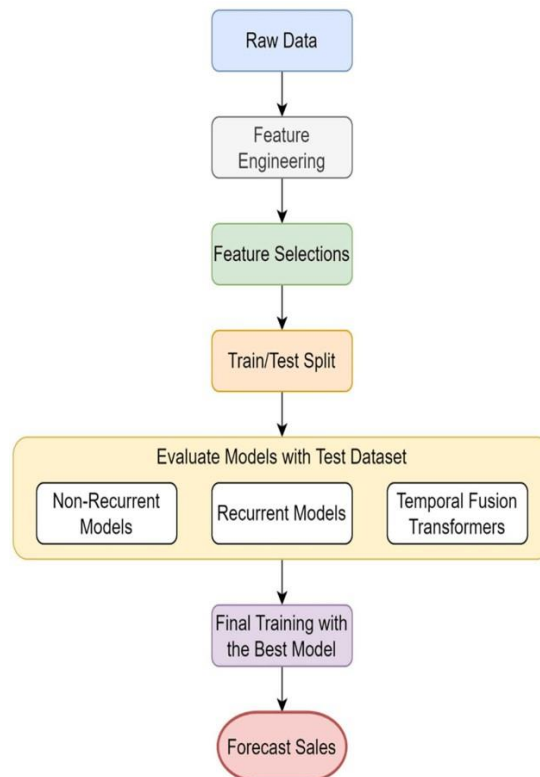
Accurate procurement planning is crucial for a food delivery service that replenishes the majority of its raw materials on a weekly basis. Considering the perishable nature of these materials, careful forecasting becomes paramount. Additionally, predicting orders can aid in the recruitment of staff members at the fulfillment center. While this process can be performed manually, it is essential to acknowledge the potential benefits of automation and technology in streamlining these operations.

PROPOSED SOLUTION

The main goal of this project is to create a machine learning model to forecast the demand for the center-meal combinations in the test set over the course of the next tenweeks given the following information.

THOERITICAL ANALYSIS

BLOCK DIAGRAM



HARDWARE/SOFTWARE DESIGNING

Jupyter notebook

Spyder

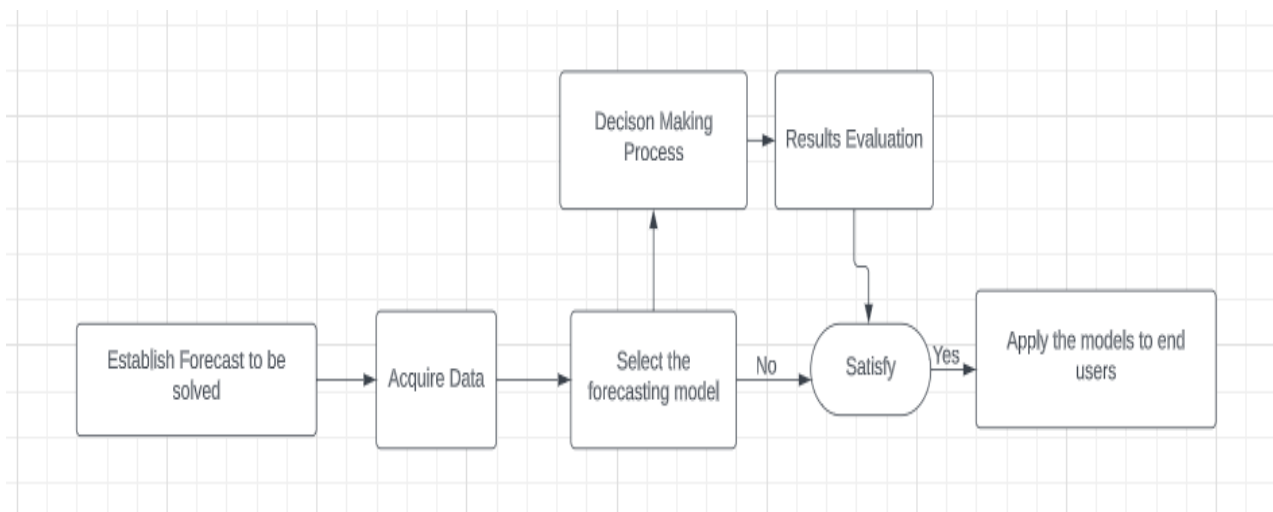
IBM Cognos Analytics

IBM Cloud

EXPERIMENTAL INVESTIGATION

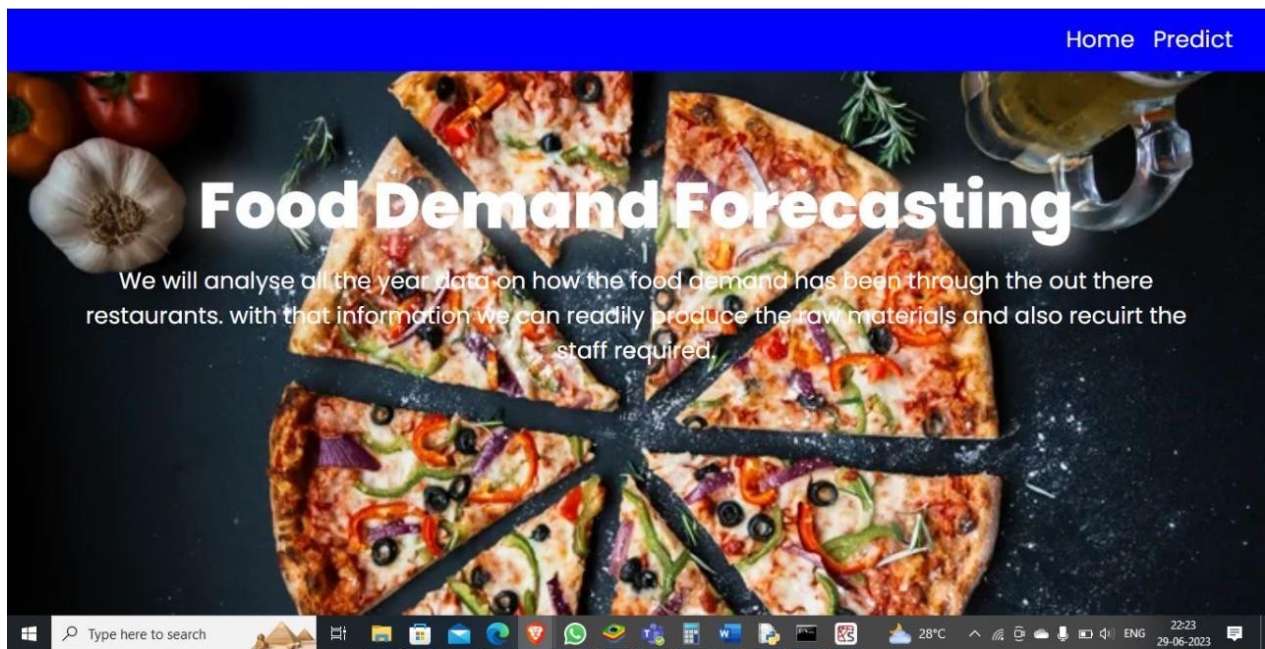
This experimental investigation acknowledges potential limitations, such as the availability and quality of historical data, assumptions made during modeling, and external factors affecting demand. Future research could explore advanced machine learning algorithms, incorporate real-time data sources, and consider external factors such as weather conditions and marketing promotions to enhance the accuracy of food demand forecasting models.

FLOWCHART

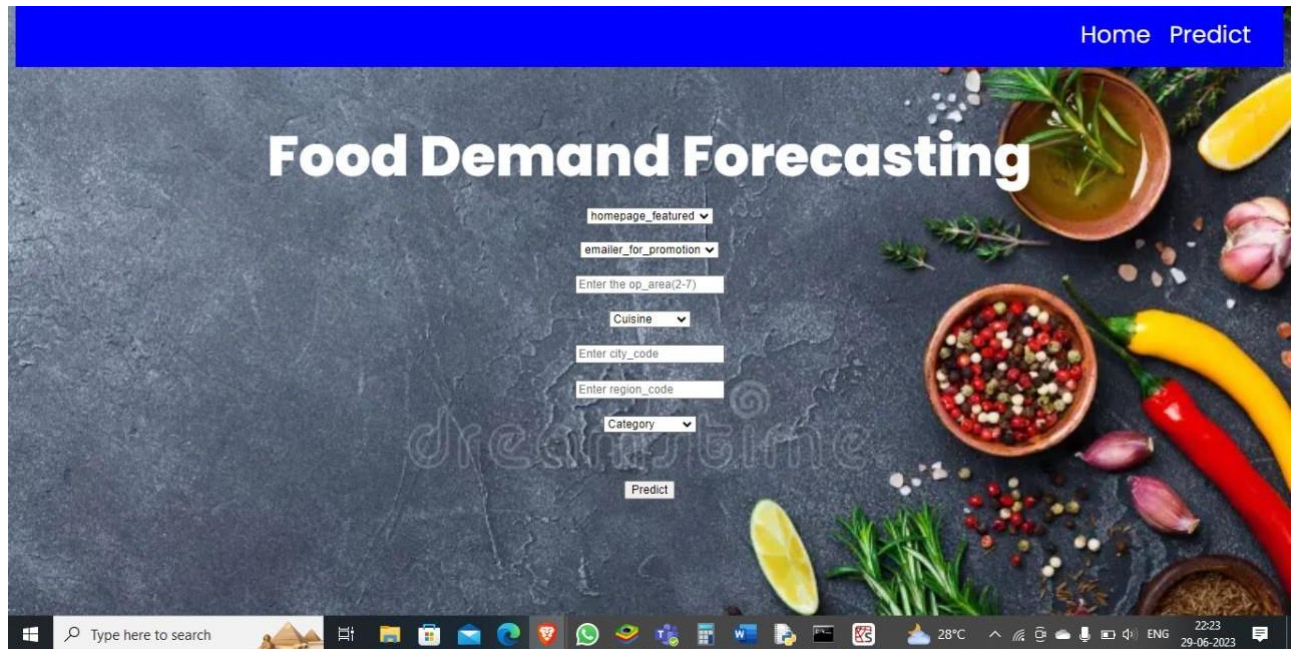


RESULTS

We have developed a precise predictive system for the study and forecasting of the demand for various food items at various locations.



(fig1. Homepage)



Home Predict

Food Demand Forecasting

homepage_featured ▼

email_for_promotion ▼

Enter the op_area(2-7)

Cuisine ▼

Enter city_code

Enter region_code

Category ▼

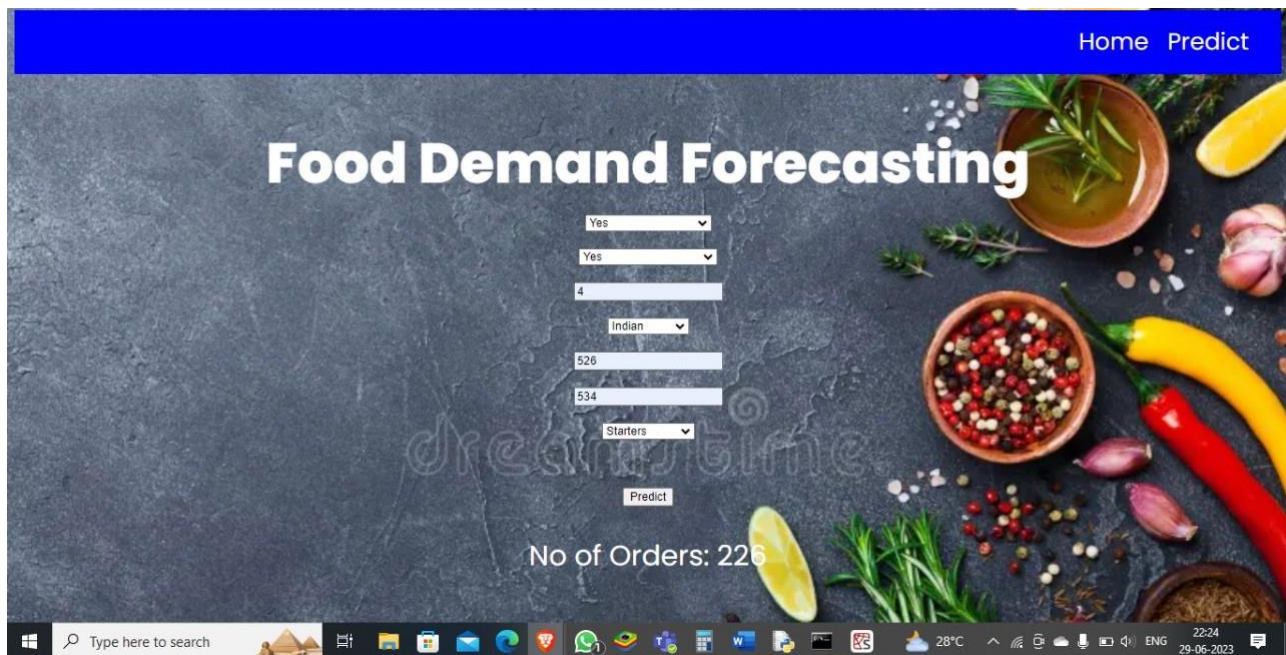
Predict

Type here to search

28°C

22:23
29-06-2023

(fig2. Predict page)



The screenshot displays a web application interface for 'Food Demand Forecasting'. The page has a dark blue header with 'Home' and 'Predict' links. The main content area features a dark, textured background with a collage of food items like lemons, herbs, and spices. The title 'Food Demand Forecasting' is prominently displayed in white. Below the title, there are several input fields: two dropdown menus set to 'Yes', a text input with the value '4', a dropdown menu set to 'Indian', a text input with '526', another text input with '534', and a dropdown menu set to 'Starters'. A 'Predict' button is located below these inputs. The result of the prediction is shown as 'No of Orders: 226'. The bottom of the image shows a Windows taskbar with various application icons, a search bar, and system status information including temperature (28°C) and date (29-06-2023).

Home Predict

Food Demand Forecasting

Yes
Yes
4
Indian
526
534
Starters
Predict

No of Orders: 226

(fig3. Output)

ADVANTAGES/DISADVANTAGES

Advantages:

There will be less food waste.

A basic and user-friendly structure.

Disadvantages:

Due to the use of small datasets, the results may not be exact.

APPLICATIONS

Optimal Procurement: Food demand forecasting enables accurate procurement planning, helping food delivery services optimize their raw material purchases and reduce the risk of wastage or stockouts.

Staffing Optimization: By predicting future orders, food demand forecasting aids in staffing decisions, allowing fulfillment centers to efficiently allocate resources and ensure adequate staff coverage during peak demand periods.

CONCLUSION

The primary goal of this project is to combat food wastage by developing an accurate food demand forecasting model. By accurately predicting the number of food orders, the model aims to assist companies in improving their service to customers and minimizing food waste. The availability of food items is essential for creating a better society, and our proposed model will contribute to achieving this objective. By leveraging the predicted demand, companies can optimize their operations, reducing excess inventory and avoiding stockouts, ultimately enhancing customer satisfaction.

This project aims to provide a practical solution that helps companies serve their customers more effectively and make a positive impact on food sustainability.

FUTURE SCOPE

Increasing the framework's liveliness by working on the frontend. We also have future plans to increase forecasting accuracy and do research on store management effectiveness.

BIBLIOGRAPHY

<https://www.kaggle.com/datasets/kannanaikkal/food-demand-forecasting>
<https://datahack.analyticsvidhya.com/contest/genpact-machine-learning-hackathon-1/>

APPENDIX:

Building model code:

```
import pandas as pd
import numpy as np
import pickle
```

```
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
meal_info = pd.read_csv("meal_info.csv")
center_info = pd.read_csv("fulfilment_center_info.csv")
trainfinal = pd.merge(train, meal_info, on="meal_id", how="outer")
trainfinal = pd.merge(trainfinal, center_info, on="center_id", how="outer")
trainfinal = trainfinal.drop(['center_id', 'meal_id'], axis=1)
cols = trainfinal.columns.tolist()
cols = cols[:2] + cols[9:] + cols[7:9] + cols[2:7]
```

```
trainfinal = trainfinal[cols]
from sklearn.preprocessing import LabelEncoder
lb1 = LabelEncoder()
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
```

```
lb2 = LabelEncoder()
trainfinal['category'] = lb1.fit_transform(trainfinal['category'])
```

```
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb1.fit_transform(trainfinal['cuisine'])
```

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

```
features = columns.drop(['num_orders']) trainfinal3 = trainfinal[features]  
X = trainfinal3.values  
y = trainfinal['num_orders'].values
```

```
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)
```

```
from sklearn.tree import DecisionTreeRegressor DT =  
DecisionTreeRegressor()  
DT.fit(X_train, y_train) pickle.dump(DT, open('fdemand.pkl', 'wb')) 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)))
```

```
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'])
```

```
lb3 = LabelEncoder()  
testfinal['cuisine'] = lb1.fit_transform(testfinal['cuisine'])  
X_test = testfinal[features].values y_test = DT.predict(X_test)
```

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

submit.to_csv("submission.csv", index=False)
```