Food Demand Forecasting for Food Delivery Company using IBM Cloud

INTRODUCTION

1.1 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 waste, and not enough could lead to out-of-stocks - and push customers to seek solutions from your competitors. The replenishment of the majority of raw materials is done on a 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.

1.2 Purpose

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 the 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.

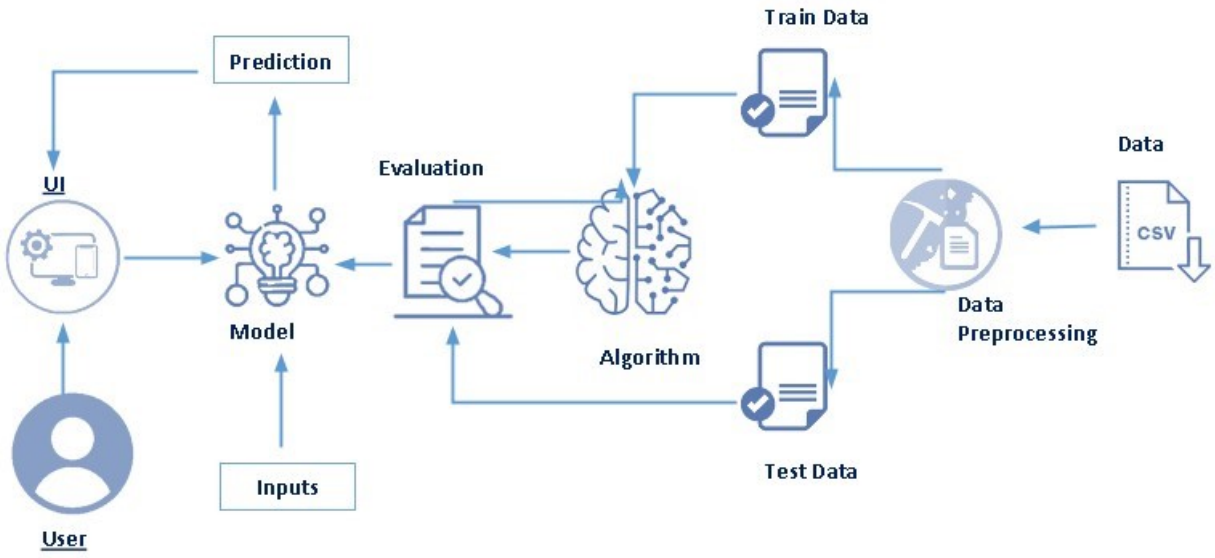
LITERATURE SURVEY

Existing problem

The replenishment of the majority of raw materials is done weekly and since the raw material is perishable, the procurement planning is of utmost importance. Secondly, staffing of the centers is also one area wherein accurate demand forecasts are really helpful. This is done through a manual process.
Proposed solution
Given the following information, the task is to predict the demand for the next 10 weeks (Weeks: 146-155) for the center-meal combinations in the test set using machine learning model.

THEORITICAL ANALYSIS

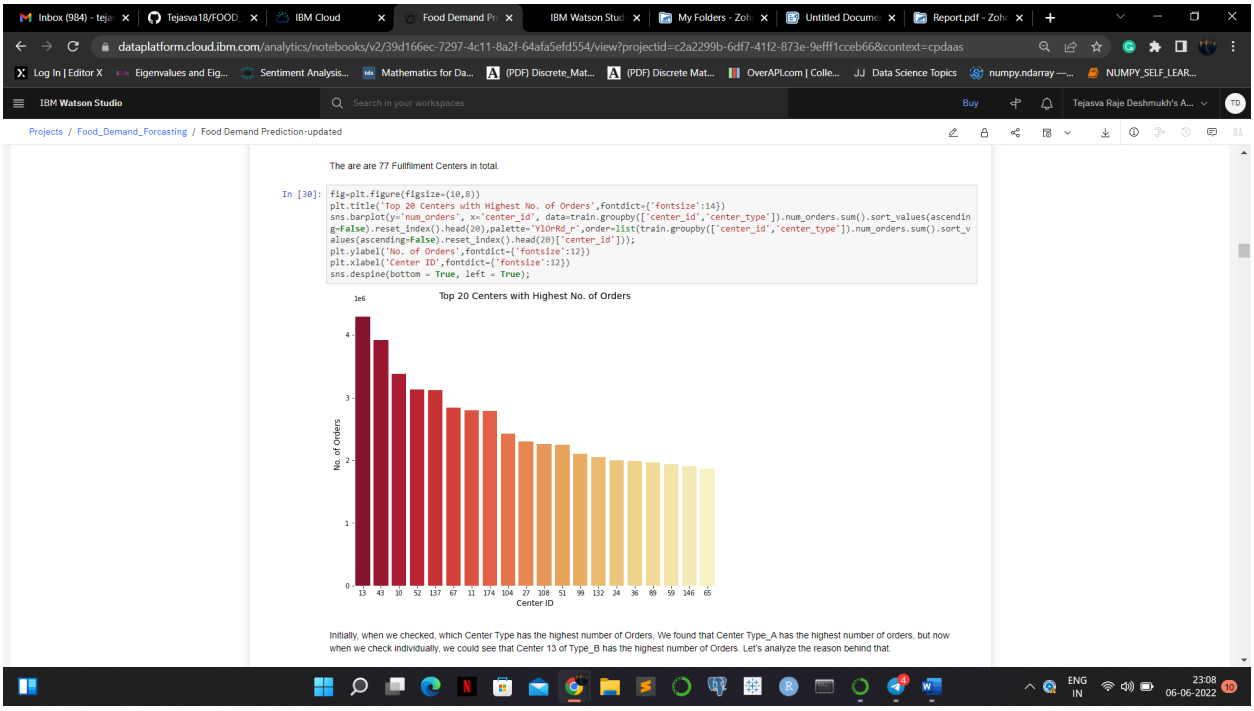
Block diagram

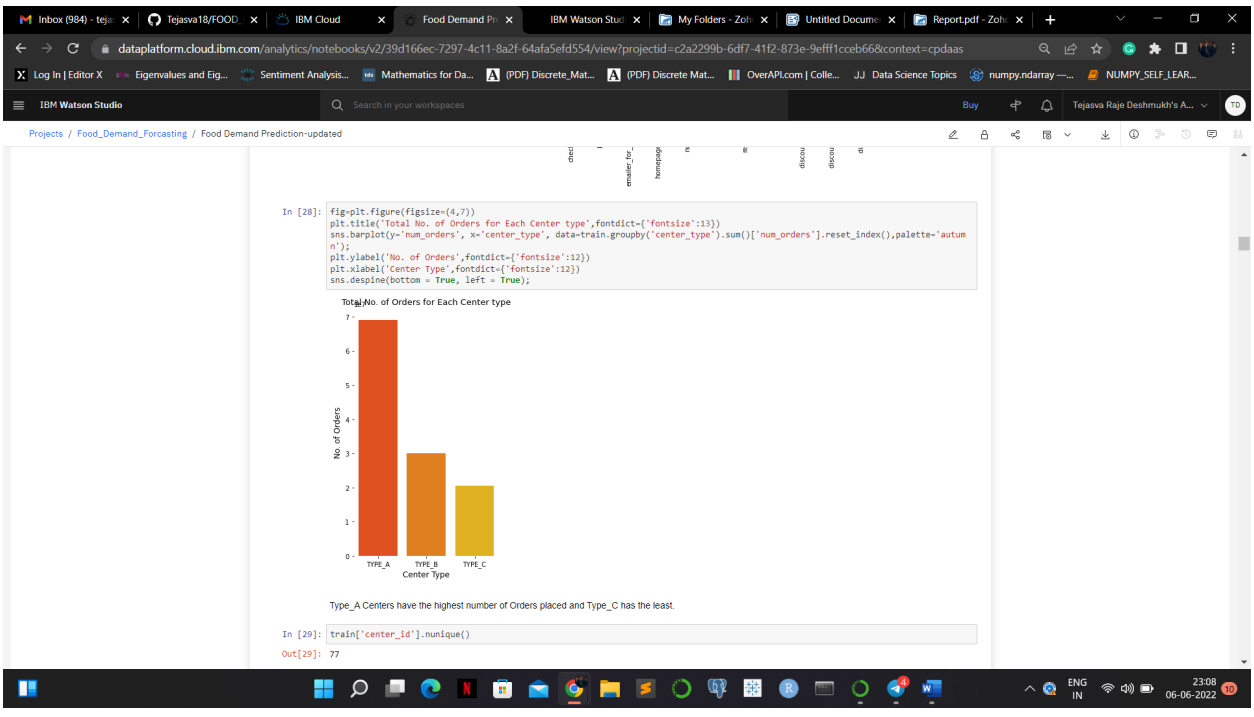


Hardware / Software designing

Anaconda.
Jupyter notebook.
Spyder IDE.
Numpy.
Pandas.
Matplotlib.
Seaborn.
Sklearn/Scikit-learn.
Flask.

EXPERIMENTAL INVESTIGATIONS

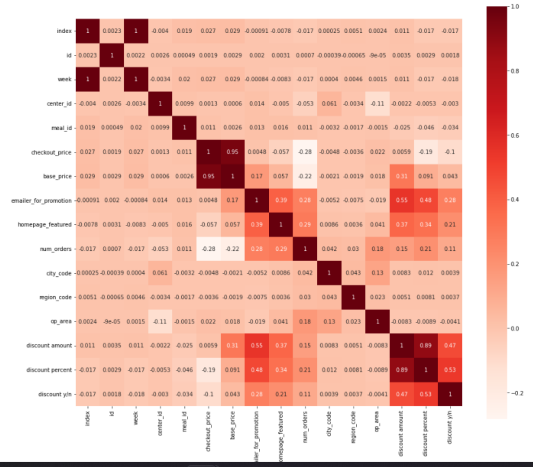




The Shape of Test dataset : (32573, 19)

In [27]:
```
plt.figure(figsize=(16,14))
sns.heatmap(train.corr(),annot=True, square=True, cmap='Reds')
```

Out[27]: <AxesSubplot:>



In [11]:
```
meal.head()
```

Out[11]:

	meal_id	category	cuisine
0	1885	Beverages	Thai
1	1993	Beverages	Thai
2	2539	Beverages	Thai
3	1248	Beverages	Indian
4	2631	Beverages	Indian

In [12]:
```
data=pd.concat([data,test],axis=0)
```

In [13]:
```
data.shape
```

Out[13]: (489121, 9)

In [14]:
```
data=data.merge(center,on='center_id',how='left')
```

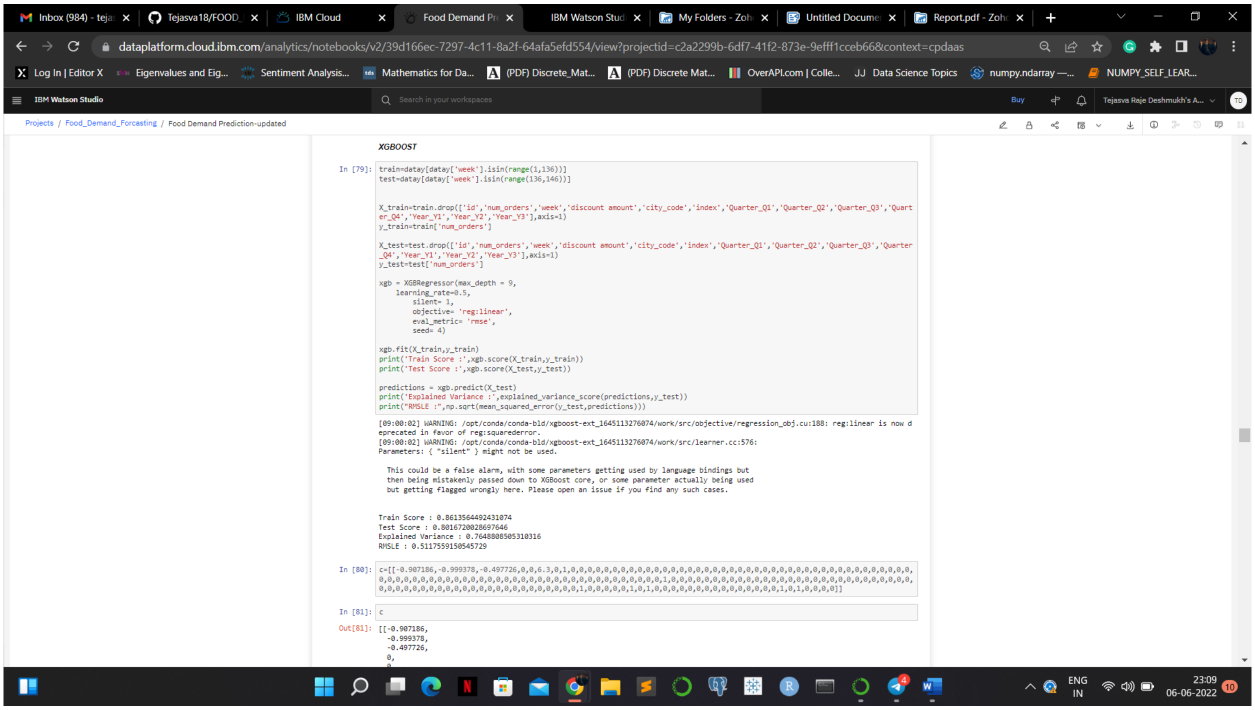
In [15]:
```
data
```

Out[15]:

	id	week	center_id	meal_id	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	city_code	region_code	center
0	1379560	1	55	1885	136.83	152.29	0	0	177	647	56	TY
1	1466964	1	55	1993	136.83	135.83	0	0	270	647	56	TY
2	1346989	1	55	2539	134.86	135.86	0	0	189	647	56	TY
3	1338232	1	55	2139	339.50	437.53	0	0	54	647	56	TY
4	1448490	1	55	2631	243.50	242.50	0	0	40	647	56	TY
...	...	...	...	...	...	...	...	...	...	...	...	...
489116	1250230	155	61	1543	482.09	484.09	0	0	123456	473	77	TY
489117	1039516	155	61	2304	483.09	483.09	0	0	123456	473	77	TY
489118	1158107	155	61	2864	322.07	323.07	0	0	123456	473	77	TY
489119	1444235	155	61	2569	322.07	323.07	0	0	123456	473	77	TY
489120	1291295	155	61	2490	276.45	276.45	0	0	123456	473	77	TY

489121 rows × 13 columns

In [16]:
```
data=data.merge(meal,on='meal_id',how='left')
```



XGBOOST

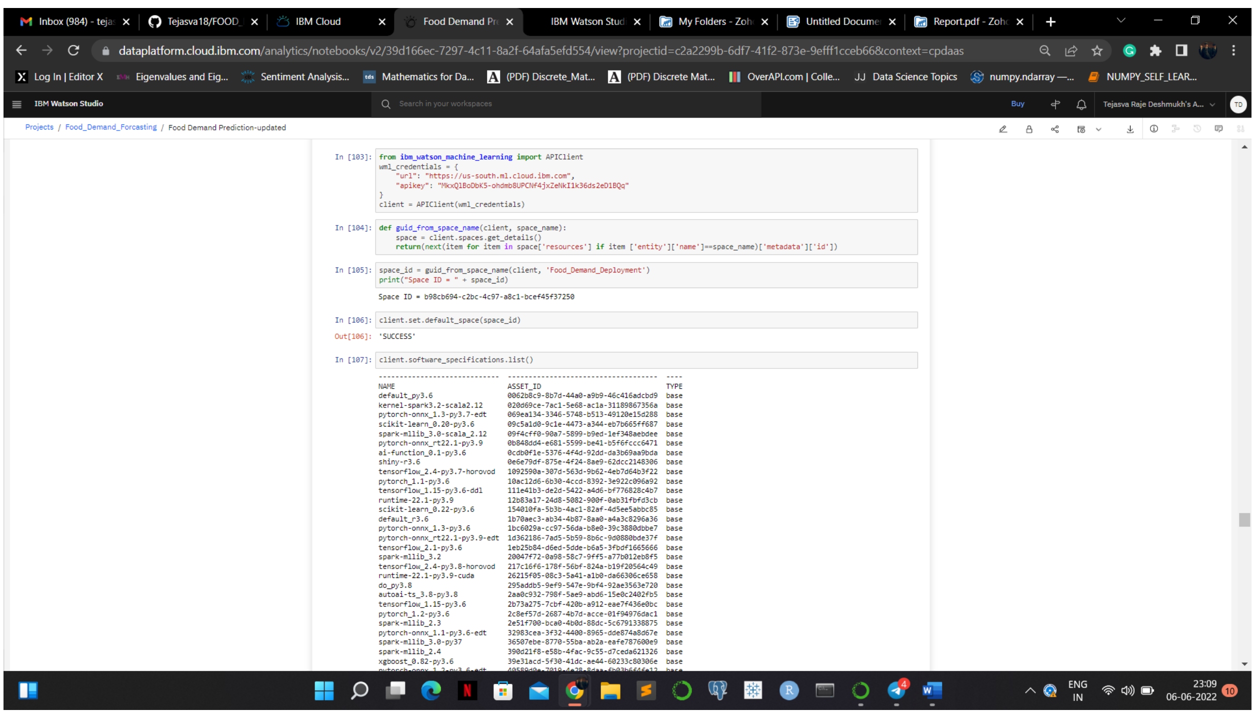
In [79]:
```python
train=datay[datay['week'].isin(range(1,136))]
test=datay[datay['week'].isin(range(136,146))]


X_train=train.drop(['id','num_orders','week','discount amount','city_code','index','Quarter_Q1','Quarter_Q2','Quarter_Q3','Quarter_Q4','Year_Y1','Year_Y2','Year_Y3'],axis=1)
y_train=train['num_orders']

X_test=test.drop(['id','num_orders','week','discount amount','city_code','index','Quarter_Q1','Quarter_Q2','Quarter_Q3','Quarter_Q4','Year_Y1','Year_Y2','Year_Y3'],axis=1)
y_test=test['num_orders']

xgb = XGBRegressor(max_depth = 9,
    learning_rate=0.5,
        silent= 1,
        objective= 'reg:linear',
        eval_metric= 'rmse',
        seed= 4)

xgb.fit(X_train,y_train)
print('Train Score :',xgb.score(X_train,y_train))
print('Test Score :',xgb.score(X_test,y_test))

predictions = xgb.predict(X_test)
print('Explained Variance :',explained_variance_score(predictions,y_test))
print("RMSLE :",np.sqrt(mean_squared_error(y_test,predictions)))
```

```
[09:00:02] WARNING: /opt/conda/conda-bld/xgboost-ext_1645113276074/work/src/objective/regression_obj.cu:188: reg:linear is now deprecated in favor of reg:squarederror.
[09:00:02] WARNING: /opt/conda/conda-bld/xgboost-ext_1645113276074/work/src/learner.cc:576:
Parameters: { "silent" } might not be used.

  This could be a false alarm, with some parameters getting used by language bindings but
  then being mistakenly passed down to XGBoost core, or some parameter actually being used
  but getting flagged wrongly here. Please open an issue if you find any such cases.


Train Score : 0.8613564492431074
Test Score : 0.8016720028697646
Explained Variance : 0.7648808505310316
RMSLE : 0.5117559150545729
```

In [80]:
```python
c=[[-0.907186,-0.999378,-0.497726,0,0,6.3,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0]]
```

In [81]:
```python
c
```

Out[81]:
```
[[-0.907186,
  -0.999378,
  -0.497726,
  0,
```



In [103]:
```python
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "MkxQ1BoDbK5-ohdmb8UPCNf4jxZeNkI1k36ds2eD1BQq"
}
client = APIClient(wml_credentials)
```

In [104]:
```python
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item ['entity']['name']==space_name)['metadata']['id'])
```

In [105]:
```python
space_id = guid_from_space_name(client, 'Food_Demand_Deployment')
print("Space ID = " + space_id)
```

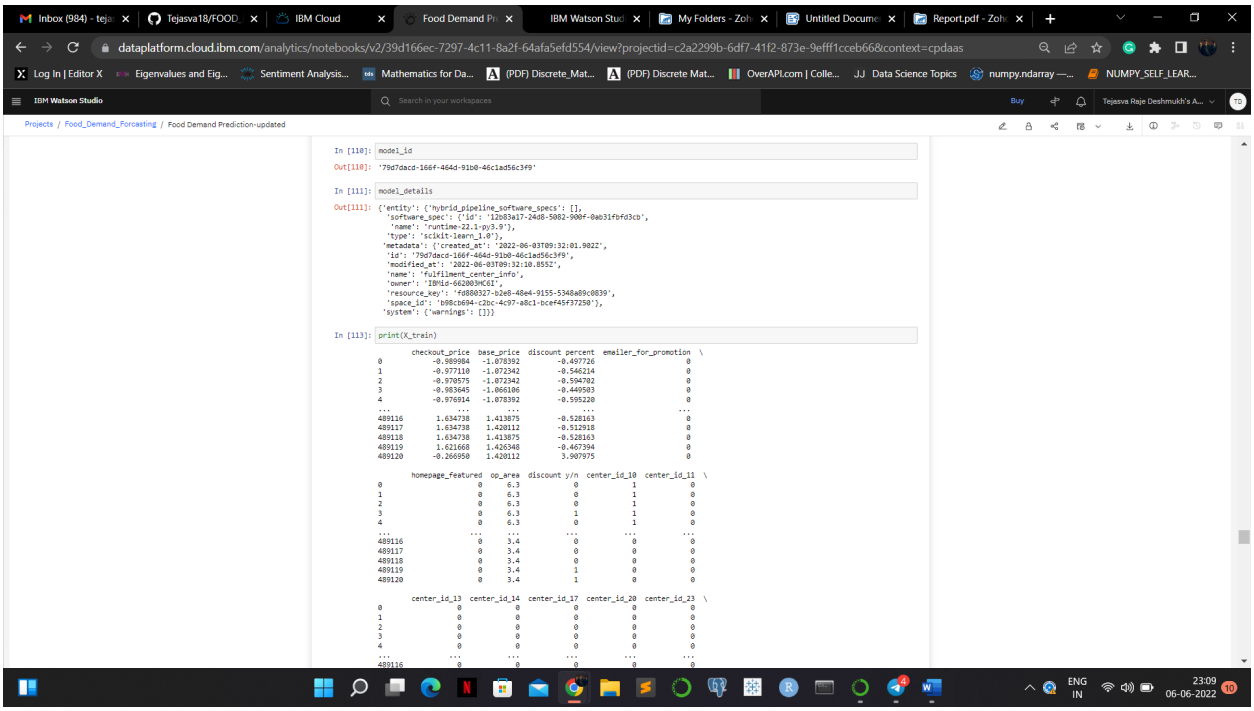
```
Space ID = b98cb694-c2bc-4c97-a8c1-bcef45f37250
```

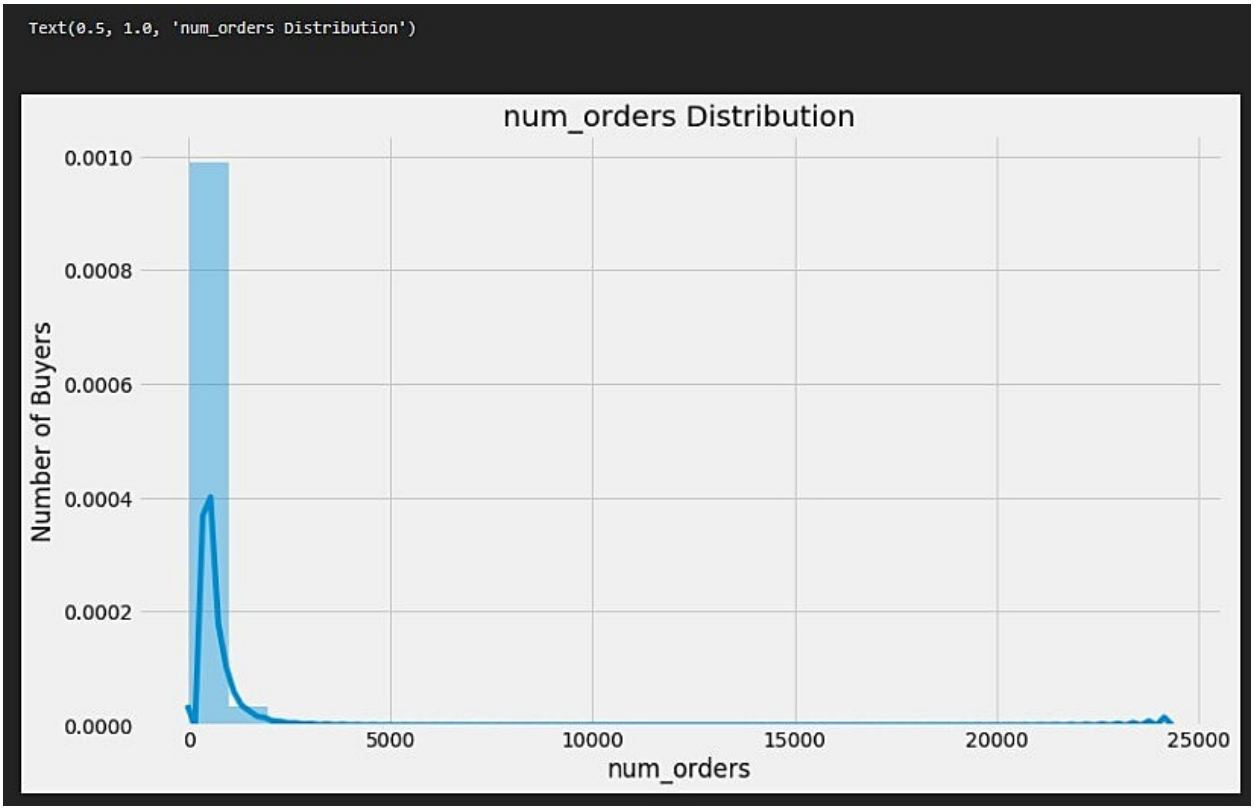
In [106]:
```python
client.set.default_space(space_id)
```

Out[106]: 'SUCCESS'

In [107]:
```python
client.software_specifications.list()
```

```
----------------------------  ------------------------------------  ----
NAME                          ASSET_ID                              TYPE
default_py3.6                 0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12     020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt    069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6       09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12    09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9     0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6         0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                    0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod  1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6             10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl     111e41b3-de2d-5422-a4d6-bf776828c4b7  base
runtime-22.1-py3.9            12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6       154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                  1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6        1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
pytorch-onnx_rt22.1-py3.9-edt 1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6          1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2               20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod  217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda       26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                      295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8           2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6         2b73a275-7cbf-420b-a912-eae7f436e0bc  base
pytorch_1.2-py3.6             2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3               2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt    32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37          36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4               390d21f8-e58b-4fac-9c55-d7ceda621326  base
xgboost_0.82-py3.6            39e31acd-5f30-41dc-ae44-60233c80306e  base
```

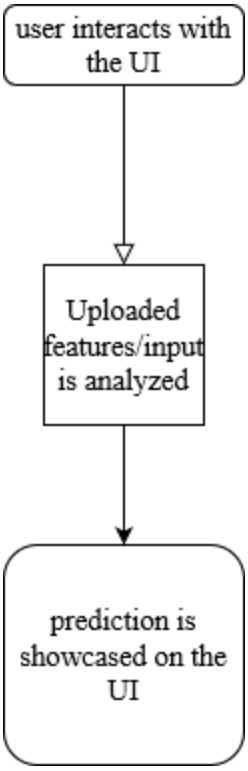

dataplatform.cloud.ibm.com/analytics/notebooks/v2/39d166ec-7297-4c11-8a2f-64afa5efd554/view?projectid=c2a2299b-6df7-41f2-873e-9efff1cceb66&context=cpdaas
IBM Watson Studio
Projects / Food_Demand_Forcasting / Food Demand Prediction-updated
In [110]: model_id
Out[110]: '79d7dacd-166f-464d-91b0-46c1ad56c3f9'
In [111]: model_details
Out[111]: {'entity': {'hybrid_pipeline_software_specs': [],
'software_spec': {'id': '12b83a17-24d8-5082-900f-0ab31fbfd3cb',
'name': 'runtime-22.1-py3.9'},
'type': 'scikit-learn_1.0'},
'metadata': {'created_at': '2022-06-03T09:32:01.902Z',
'id': '79d7dacd-166f-464d-91b0-46c1ad56c3f9',
'modified_at': '2022-06-03T09:32:10.855Z',
'name': 'fulfilment_center_info',
'owner': 'IBMid-662003HC6I',
'resource_key': 'fd880327-b2e8-48e4-9155-5348a89c0839',
'space_id': 'b98cb694-c2bc-4c97-a8c1-bcef45f37250'},
'system': {'warnings': []}}
In [113]: print(X_train)
checkout_price base_price discount percent emailer_for_promotion \
0 -0.989984 -1.078392 -0.497726 0
1 -0.977110 -1.072342 -0.546214 0
2 -0.970575 -1.072342 -0.594702 0
3 -0.983645 -1.066106 -0.449503 0
4 -0.976914 -1.078392 -0.595220 0
... ...
489116 1.634738 1.413875 -0.528163 0
489117 1.634738 1.420112 -0.512918 0
489118 1.634738 1.413875 -0.528163 0
489119 1.621668 1.426348 -0.467394 0
489120 -0.266950 1.420112 3.907975 0
homepage_featured op_area discount y/n center_id_10 center_id_11 \
0 0 6.3 0 1 0
1 0 6.3 0 1 0
2 0 6.3 0 1 0
3 0 6.3 1 1 0
4 0 6.3 0 1 0
... ...
489116 0 3.4 0 0 0
489117 0 3.4 0 0 0
489118 0 3.4 0 0 0
489119 0 3.4 1 0 0
489120 0 3.4 1 0 0
center_id_13 center_id_14 center_id_17 center_id_20 center_id_23 \
0 0 0 0 0 0
1 0 0 0 0 0
2 0 0 0 0 0
3 0 0 0 0 0
4 0 0 0 0 0
... ...
489116 0 0 0 0 0
ENG
IN
23:09
06-06-2022


Text(0.5, 1.0, 'num_orders Distribution')
num_orders Distribution
Number of Buyers
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
5000
10000
15000
20000
25000
num_orders

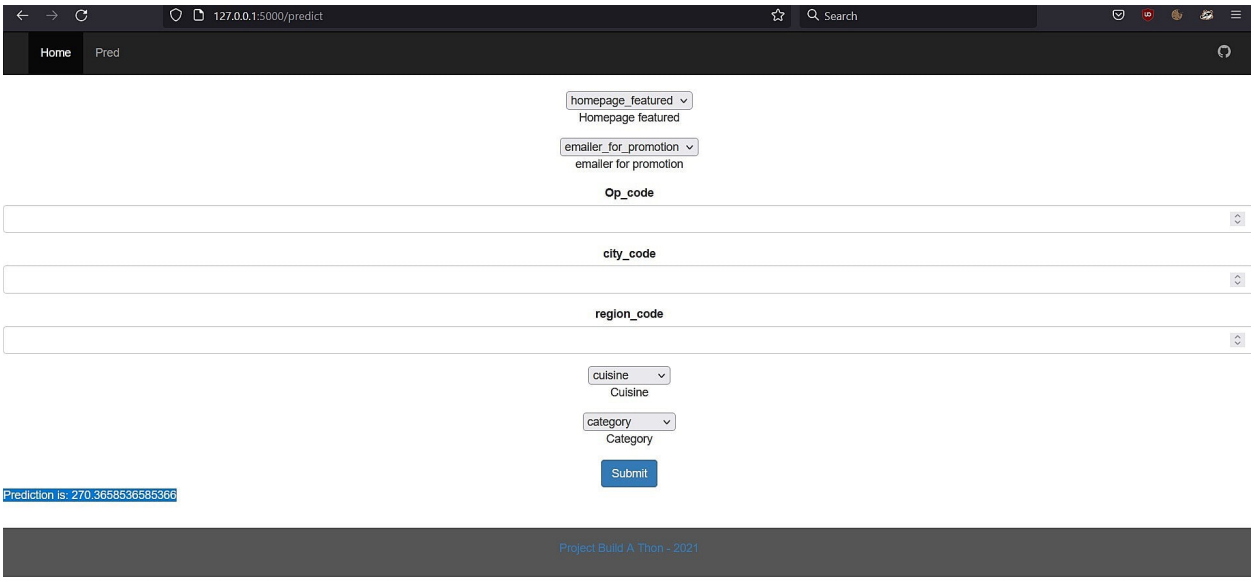
FLOWCHART

Project Work Flow:

The user interacts with the UI (User Interface) to upload the input features.
Uploaded features/input is analysed by the model which is integrated.
Once the model analyses the uploaded inputs, the prediction is showcased on the UI.



RESULT



ADVANTAGES & DISADVANTAGES

Advantages:

Easy to use and deploy.

Disadvantages:

Requires maintaining dataset and regular update and testing.

APPLICATIONS

Food demand forecasting for meal delivery company which helps the firm to predict the number of upcoming food demand orders from the target audience so they can prepare accordingly.

CONCLUSION

Preformed food demand forecasting for food delivery company using IBM Cloud Watson studio where we created our machine learning space and storage on cloud to deploy and test our machine learning model which is created by using Logistic Regressor, Libraries used are pandas, numpy for mathematicl analysis and sklearn, matplotlib for data visualization. We have used XGBoost library to test and train our data model and to get the test and train accuracy. After getting the accuracy we have integrated it with the flask where user can feed input and get their desired prediction in seconds.

FUTURE SCOPE

Improvement in UI/UX and setting up ml-pipeline can be done as future improvement.
Also, it will help restaurants to reduce their food waste and help them grow and know their audience and their most liked preparation.

BIBILOGRAPHY

https://smartinternz.com/ibm-build-a-thon-2021 .

https://www.w3schools.com/

https://towardsdatascience.com/

Source Code.

[FOOD_DEMAND_FORECASTING NOTEBOOK]