

# TEAM ID: PNT2022TMID15280

The image displays two screenshots of the IBM Watson Studio interface, showing a notebook titled "FOODDEMANDEST".

**Top Screenshot: Model evaluation**

The notebook content includes:

- Model evaluation**
- Text: "We're going to use `x_train` and `y_train` obtained above in `train_test_split` section to train our regression model. We're using the `fit` method and passing the parameters as shown below."
- Text: "Finally, we need to check to see how well our model is performing on the test data."
- Regression Evaluation Metrics: RMSE(Root Mean Square Error)**
- Text: "RMSE is the square root of the averaged squared difference between the target value and the value predicted by the model. It is preferred more in some cases because the errors are first squared before averaging which poses a high penalty on large errors. This implies that RMSE is useful when large errors are undesired."
- Code cell [83]:

```
XG = XGBRegressor()
XG.fit(X_train, y_train)
y_pred = XG.predict(X_val)
y_pred[y_pred<0] = 0
from sklearn import metrics
print('RMSE:', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
```

RMSE: 70.95830804827709
- Code cell [84]:

```
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('RMSE:', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
```

RMSE: 130.16386189713575
- Code cell [85]:

```
L = Lasso()
L.fit(X_train, y_train)
y_pred = L.predict(X_val)
y_pred[y_pred<0] = 0
from sklearn import metrics
print('RMSE:', 100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
```

**Bottom Screenshot: Model Building**

The notebook content includes:

- Model Building**
- Text: "Model building includes the following main tasks"
- List of tasks:
  - Train and test model algorithms
  - Evaluation of Model
  - Save the model
  - Predicting the output using the model
- Train and Test Model Algorithms**
- Code cell [80]:

```
pip install xgboost
```

Requirement already satisfied: xgboost in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.5.2)  
Requirement already satisfied: numpy in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from xgboost) (1.20.3)  
Requirement already satisfied: scipy in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from xgboost) (1.7.3)  
Note: you may need to restart the kernel to use updated packages.
- Code cell [81]:

```
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
from xgboost import XGBRegressor
```
- Model evaluation**
- Text: "We're going to use `x_train` and `y_train` obtained above in `train_test_split` section to train our regression model. We're using the `fit` method and passing the parameters as shown below."
- Text: "Finally, we need to check to see how well our model is performing on the test data."

FOODDEMANDEST - IBM Wat...

dataplatform.cloud.ibm.com/analytics/notebooks/v2/071c51ff-2c68-45b9-8c37-c4ee52ee407f?projectId=f1f58cae-a900-429f-8a44-539445c6a016&context=cpdaas

IBM Watson Studio

Search in your workspaces

Buy

MAHALAKSHMI G's Account

Dallas

MG

Projects / FOOD DEMAND / FOODDEMANDEST

File Edit View Insert Cell Kernel Help

Not Connected Trusted Python 3.9

Save the Model

In [90]:

```
import pickle
pickle.dump(DT, open('fdemand.pkl', 'wb'))
```

Predicting the output using the model

Here, we are creating X\_test which we are using to test the model to predict the number of orders by giving input to the model build.

In [91]:

```
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)

tools = testfinal.columns.tolist()
tools = tools[:2] + tools[8:] + tools[6:8] + tools[2:6]
testfinal = testfinal[tools]

lbl = LabelEncoder()
testfinal['center_type'] = lbl.fit_transform(testfinal['center_type'])
testfinal['category'] = lbl.fit_transform(testfinal['category'])
testfinal['cuisine'] = lbl.fit_transform(testfinal['cuisine'])

X_test = testfinal[features].values
```

In [92]:

```
testfinal.dtypes
```

Out[92]:

id	int64
week	int64
city_code	int64
region_code	int64
center_type	int64
op_area	float64
category	int64
cuisine	int64