

Team ID: PNT2022TMID23576

PROJECT NAME: DemandEst - AI powered Food Demand Forecaster

Team Leader

```
colab.research.google.com/drive/1nhHECd4iq--10dkcHMYdy_hDhPD0kx_#scrollTo=n757G6XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

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

[05:36:29] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
RMSLE: 100.175198689628

[ ] 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: 129.7404397040951

[ ] 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('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))

RMSLE: 129.22746395054898
```

```
colab.research.google.com/drive/1nhHECd4iq--10dkcHMYdy_hDhPD0kx_#scrollTo=n757G6XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

[ ] 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: 131.1336557664208

[ ] 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: 62.59878547492348

[ ] 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: 66.4669666824401
```

```
colab.research.google.com/drive/1nhHECd4iq--10dkcHMYdy_hDhPD0kx_#scrollTo=n757G6XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

[ ] 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: 100.16911494445236
```

Team Member 1



colab.research.google.com/drive/1nhHECd4iq-10dkcHMVdy_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

Connect Editing

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

[05:36:29] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
RMSLE: 100.175198689628

```
[ ] 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: 129.7404397040951

```
[ ] 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('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))
```

RMSLE: 129.22746395054898



colab.research.google.com/drive/1nhHECd4iq-10dkcHMVdy_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

Connect Editing

```
[ ] 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: 131.1336557664208

```
[ ] 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: 62.59878547492348

```
[ ] 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: 66.4669666824401



colab.research.google.com/drive/1nhHECd4iq-10dkcHMVdy_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text

Connect Editing

```
[ ] 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: 100.16911494445236

Team Member 2



This screenshot shows a Google Colab notebook titled "Data_preprocessing_and_Model_building (1).ipynb". The notebook contains three code blocks, each testing a different regression model and printing the Root Mean Squared Error (RMSE) on a validation set. The first block uses XGBoost, the second uses Linear Regression, and the third uses Lasso. A warning message is displayed above the Linear Regression code, indicating that the 'reg:linear' parameter is deprecated in favor of 'reg:squarederror'.

```
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)))

[05:36:29] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
RMSE: 100.175198689628

[ ] 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: 129.7404397040951

[ ] 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)))

RMSE: 129.22746395054898
```



This screenshot shows the continuation of the Colab notebook, testing three more regression models: ElasticNet, Decision Tree, and K-Nearest Neighbors. Each model is fitted to the training data, used to predict on the validation set, and its RMSE is printed. The ElasticNet model has an RMSE of approximately 131.13, the Decision Tree model has an RMSE of approximately 62.59, and the K-Nearest Neighbors model has an RMSE of approximately 66.46.

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

RMSE: 131.1336557664208

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

RMSE: 62.59878547492348

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

RMSE: 66.4669666824401
```



This screenshot shows the final code block in the notebook, testing a Gradient Boosting Regressor. The model is fitted to the training data, used to predict on the validation set, and its RMSE is printed, which is approximately 100.16.

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

RMSE: 100.16911494445236
```

Team Member 3



```
colab.research.google.com/drive/1nhHECd4iq--1OdkcHMVidY_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text
Connect Editing

[ ] 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('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))

[05:36:29] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
RMSLE: 100.175198689628

[ ] 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: 129.7404397040951

[ ] 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('RMSLE:',100*np.sqrt(metrics.mean_squared_log_error(y_val, y_pred)))

RMSLE: 129.22746395054898
```



```
colab.research.google.com/drive/1nhHECd4iq--1OdkcHMVidY_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text
Connect Editing

[ ] 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: 131.1336557664208

[ ] 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: 62.59878547492348

[ ] 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: 66.4669666824401
```



```
colab.research.google.com/drive/1nhHECd4iq--1OdkcHMVidY_hDhPD0lx_#scrollTo=n757G0XVO_VI

Data_preprocessing_and_Model_building (1).ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 19

+ Code + Text
Connect Editing

[ ] 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: 100.16911494445236
```