

Project Development Phase Model Performance Test

Date	18 November 2022
Team ID	PNT2022TMID23900
Project Name	Project – DemandEst-AI Powered Food Demand Forecaster
Maximum Marks	10 Marks

Model Performance Testing:

S.No.	Parameter	Values	Screenshot
1.	Metrics	Regression Model: MAE 89.10334778841495, MSE - 43129.82977026746, RMSLE -207.67722496765856, R2 score -0.6946496854280233,	 <p>Evaluating the model</p> <pre> In [33]: from sklearn.metrics import mean_squared_error In [34]: RMSLE=np.sqrt(mean_squared_error(y_test,pred)) RMSLE Out[34]: 209.71961740201198 In [39]: from sklearn import metrics from sklearn.metrics import mean_absolute_error In [40]: MSE=print(metrics.mean_squared_error(y_test,pred)) MSE 43982.31792324628 In [41]: R2S=print(metrics.r2_score(y_test,pred)) R2S 0.6886142448276894 In [42]: MAE=print(mean_absolute_error(y_test,pred)) MAE 89.10334778841495 </pre>

2.

Tune the Model

Hyperparameter Tuning

RMSLE- 52.85812511759974

avg R-squared- 0.123

MSE: -64230.918

In [38]:

print("R-Squared:{}".format(grid_cv_dtm.best_score_))
print("Best Hyperparameters:{}".format(grid_cv_dtm.best_params_))

R-Squared:0.760137863085842

Best Hyperparameters:
{'max_leaf_nodes': None, 'min_samples_leaf': 4, 'min_samples_split': 16}

In [39]:

df = pd.DataFrame(data=grid_cv_dtm.cv_results_)
df.head()

Out[39]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_max_leaf_nodes	param_min_samples_leaf	param_min_samples_split	params
0	5.324027	1.065213	0.006586	0.028995	None	1	2	{'max_leaf_nodes': None, 'min_samples_leaf': 1, 'min_samples_split': 2}
1	4.932083	0.489172	0.006634	0.005248	None	1	4	{'max_leaf_nodes': None, 'min_samples_leaf': 1, 'min_samples_split': 4}
2	4.607915	0.326080	0.050024	0.005244	None	1	8	{'max_leaf_nodes': None, 'min_samples_leaf': 1, 'min_samples_split': 8}
3	4.148544	1.038443	0.043753	0.018894	None	1	16	{'max_leaf_nodes': None, 'min_samples_leaf': 1, 'min_samples_split': 16}
4	4.017265	0.764551	0.050551	0.005479	None	2	2	{'max_leaf_nodes': None, 'min_samples_leaf': 2, 'min_samples_split': 2}

In [42]:

r2_scores = cross_val_score(grid_cv_dtm.best_estimator_, X, y, cv=10)
mse_scores = cross_val_score(grid_cv_dtm.best_estimator_, X, y, cv=10, scoring='neg_mean_squared_error')

print("avg R-squared:{}".format(np.mean(r2_scores)))
print("MSE:{}".format(np.mean(mse_scores)))

avg R-squared:0.123

MSE:-64230.918

In [45]:

grid_cv_dtm.best_estimator_.fit(X_train, y_train)
y_pred = grid_cv_dtm.best_estimator_.predict(X_test)
y_pred[y_pred<0] = 0
from sklearn import metrics
print("RMSLE: ", 100*np.sqrt(metrics.mean_squared_log_error(y_test, y_pred)))

RMSLE: 52.85812511759974

In []:

Tuning the model Using GridSearchCV

In [33]:

from sklearn import preprocessing
from sklearn.model_selection import GridSearchCV, cross_val_score, cross_val_predict
import seaborn as sns
import matplotlib.pyplot as plt
sns.set_style('whitegrid')
sns.set_context('talk')
params = {'legend.fontsize': 'x-large',
 'figure.figsize': (30, 10),
 'axes.labelsize': 'x-large',
 'axes.titlesize': 'x-large',
 'xtick.labelsize': 'x-large',
 'ytick.labelsize': 'x-large'}

plt.rcParams.update(params)

In [37]:

param_grid = {'min_samples_split': [2, 4, 8, 16], 'min_samples_leaf': [1, 2, 3, 4], 'max_leaf_nodes': [None, 10, 20, 100]}

grid_cv_dtm = GridSearchCV(model, param_grid, cv=5)

grid_cv_dtm.fit(X_train, y_train)

Out[37]:

GridSearchCV(cv=5, estimator=DecisionTreeRegressor(),
 param_grid={'max_leaf_nodes': [None, 10, 20, 100],
 'min_samples_leaf': [1, 2, 3, 4],
 'min_samples_split': [2, 4, 8, 16]})

