

Table of content

First: INTRODUCTION

1.1. PROBLEM STATEMENT

1.2. DATA

Second: METHODOLOGY

2.1. PREPROCESSING

2.2. MODEL DEVELOPMENT

Finally: CONCLUSION

First: INTRODUCTION

1.1. Problem Statement:

This project analyzes the content of an E-commerce database. Based on this analysis, we have to predict the bike rental count.

Data description:

Train dataset has 7165 rows and 11 columns.

Test dataset has 3530 rows and 10 columns.

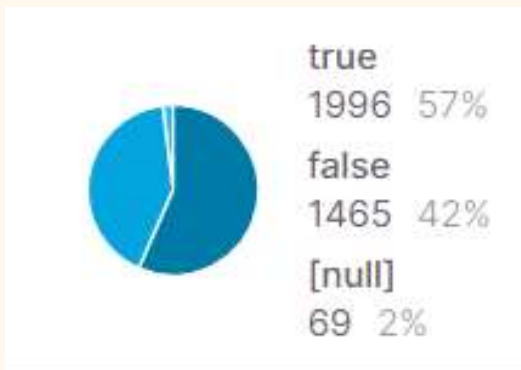
Columns Description:

Variable	Definition
ID	Unique ID
Gender	Gender of the customer
Ever_Married	Marital status of the customer
Age	Age of the customer
Graduated	Is the customer a graduate?
Profession	Profession of the customer
Work_Experience	Work Experience in years
Spending_Score	Spending score of the customer
Family_Size	Number of family members for the customer (including the customer)
Var_1	Anonymised Category for the customer
Segmentation	(target) Customer Segment of the customer

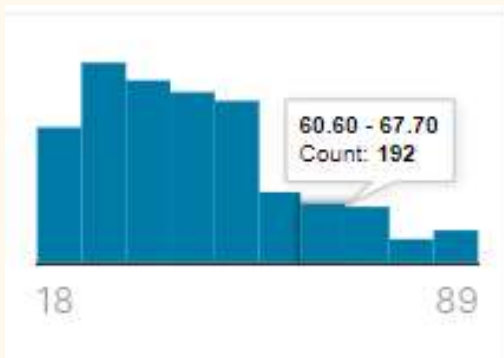
a. Gender:



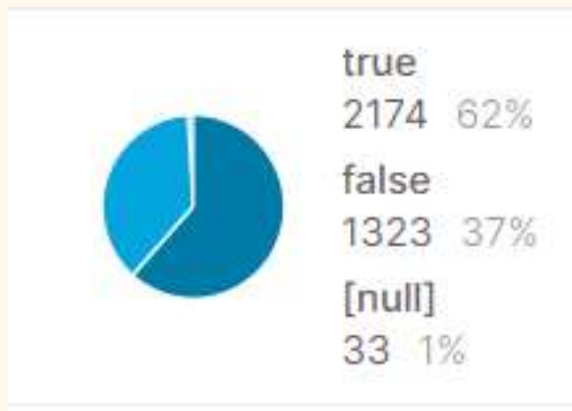
b. Ever married:



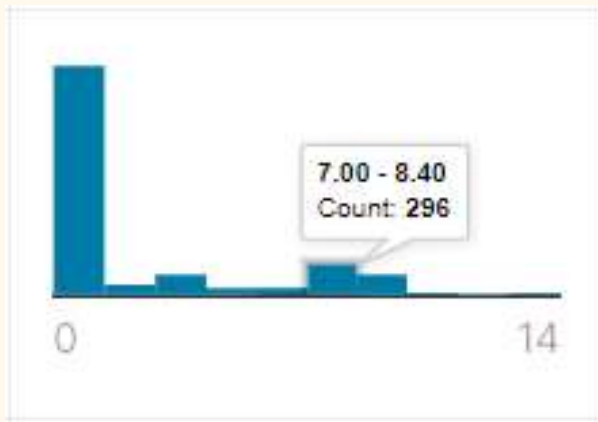
c. Age:



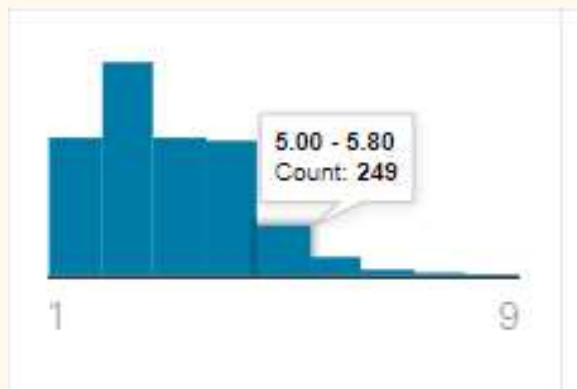
d. Graduated:



e. Work experience:



f. Family size:



Second: METHODOLOGY

2.1. Preprocessing:

Checking the datatypes of every column whether it's numerical or categorical.

- **Label encoding:**

We use label encoding because categorical data cant be processed by models(we used it with ordinal data & with data that has 2 attributes only)

'Spending_Score','Gender','Graduated','Ever_Married',
'Segmentation']

- **One-hot encoding:**

Because categorical data cant be processed by models(we used it with nominal data) and the columns are not ordinal so we do not want to assign weights to the values['profession','Var_1']

- **Filling Nan values:**

We filled Nan with KNN imputer for all models except Lightgbm ,CatBoost and XGBoost ,we left the lightgbm ,catboost and xgboost fill the nan implicitly due to their effective ability to fill the nan and that yielded to better results.

- **Splitting train data:**

Splitting data to train and validation data ,to evaluate how well the model makes predictions based on the new data(train →80%,test→20% validation,random_satate=42).

- **Correlation and Feature Selection:**



- **Feature selection:**

As shown in the above matrix, the 'Work_Experience' column has very low correlation with the 'Segmentation' column (0.052) so it was better to drop it and drop the spending score but unfortunately it made the model perform worse.

Model Selection:

Baseline Models:

```
ExtraTreesClassifier : 0.37264480111653875  
DecisionTree : 0.3600837404047453  
RandomForest : 0.3740404745289602  
AdaBoostClassifier : 0.45638520586182835  
GradientBoostingClassifier: 0.47871598046057223  
XGB : 0.47592463363572923  
CatBoost : 0.4466154919748779
```

Optimizing Models:

1. Gradient boosting classifier:

It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees. We used grid search with 10 fold cross validation.

```
print("gsearch1.best_score_",cv.best_score_)  
gsearch1.best_score_ 0.4678968203294599
```

2. XGBooster Classifier:

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. We used gridsearch to optimize hyperparameters and used ten fold cross validation. [snippet from code showing accuracy and confusion matrix]


```
[35] modelXg = XGBClassifier(  
        learning_rate=0.011, n_estimators=500, max_depth=5, min_child_weight=100,  
        nthread=-1, subsample=0.8, colsample_bytree=0.7, scoring = 'accuracy', seed=7  
    )  
modelXg.fit(X_train, y_train)  
predXg=modelXg.predict(X_test)  
accuracy_score(y_test, predXg)  
  
0.4856943475226797  
  
from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error  
cm = confusion_matrix(predXg, y_test)  
cm  
  
array([[188, 112, 55, 104],  
       [ 44,  74,  37,  18],  
       [ 62, 110, 176,  27],  
       [ 78,  52,  38, 258]])
```


Submission_xgb	0.47875	0.49065	<input type="checkbox"/>
6 hours ago by Seif Emad			
add submission details			

3. CatBoost

CatBoost is based on gradient boosted decision trees. During training, a set of decision trees is built consecutively. Each successive tree is built with reduced loss compared to the previous trees. The number of trees is controlled by the starting parameters.

We optimized the Catboost model by gridsearch and ten fold cross validation

Submission and Description	Private Score	Public Score	Use for Final Score
Submission_cat_filltest (1) (2)	0.47818	0.50028	<input type="checkbox"/>
5 hours ago by Seif Emad			
add submission details 			

```
[31] preds_cat = catmodel.predict(X_test)
# catmodel.best_svc.best_params_
from sklearn.metrics import accuracy_score
print("Val Acc:", accuracy_score(preds_cat, y_test))

Val Acc: 0.4961618981158409

from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error
cm = confusion_matrix(preds_cat, y_test)

cm
array([[182, 104, 56, 92],
       [ 57, 90, 39, 26],
       [ 55, 104, 174, 24],
       [ 78, 50, 37, 265]])
```

4.LightgbmClassifier:

Stands for Light Gradient Boosting Machine

It is based on decision tree algorithms and used for ranking, classification and other machine learning tasks.

For better performance, we optimized the model by gridsearch and ten fold cross validation.

```
cm

array([[155, 106, 56, 89],
       [ 62, 82, 46, 29],
       [ 56, 104, 161, 29],
       [ 99, 56, 43, 260]])

[44] accuracy_score(y_test, preds_lgbm_t)

0.45917655268667135
```

add submission details			
Submission_lgbm (1)	0.48045	0.49801	<input type="checkbox"/>
15 days ago by NouranHanyMoh			
add submission details			

5.ExtraTree classifier:

ExtraTreesClassifier is an ensemble learning method fundamentally based on decision trees.Performed poorly in private and public leaderboard.

Submission_xtra	0.39206	0.41076	<input type="checkbox"/>
7 days ago by NouranHanyMoh			
add submission details			

```
from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error
cm = confusion_matrix(xtree, y_test)
cm
```

```
array([[130, 101, 68, 95],
       [ 95, 85, 96, 47],
       [ 59, 105, 99, 53],
       [ 88, 57, 43, 212]])
```

6.VotingClassifier:

We ensembled the CatBoostClassifier and XGBoost but it yielded worse results than using XGBoost and CatBoost individually.

Submission_voting (1)	0.44589	0.46685	<input type="checkbox"/>
7 days ago by NouranHanyMoh			
add submission details			

Techniques to Improve Results:

We Merged train and test data to improve the generalization capability of the model which increased our score in the private leaderboard significantly.

Submission_cat_merge	0.47875	0.50594	<input type="checkbox"/>
10 hours ago by NouranHanyMoh			
add submission details			

Finally: Conclusion:

1. Our intuition is that merging the train data and test data got the highest score in private leaderboard but not in public because merging increased the generalization capability of the model.

add submission details			
Submission_cat_merge_grid	0.48725	0.49688	<input type="checkbox"/>
8 hours ago by NouranHanyMoh			
add submission details			

2. The problem needed generalization capability more than it needed higher score for the private leaderboard
3. Catboost and XGBoost performed best in this problem
4. Columns Ever_Married and spending score are highly correlated so Spending Score can be used to fill the nan in Ever_married