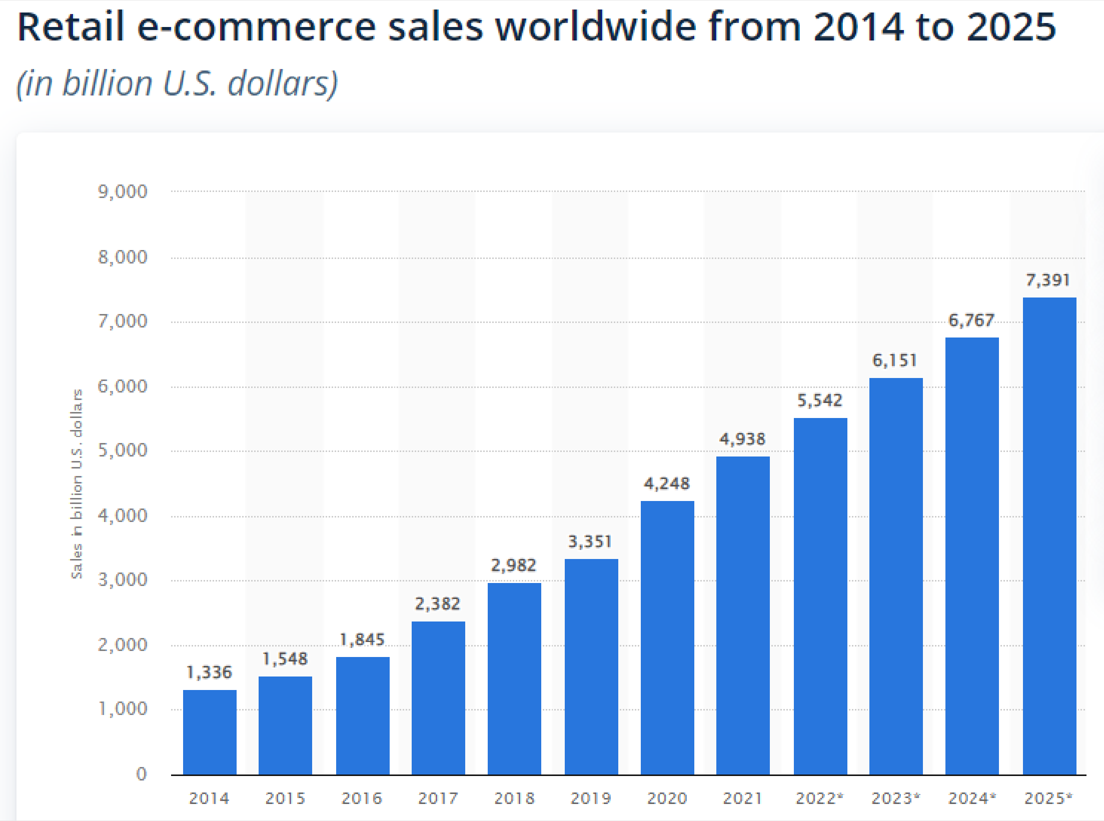
Will the Online Shopper Purchase?

Group 7

A0231906J HUANG, HAI-HSIN
A0231947Y TSAO, KAI-TING
A0231971E XIANG ZISHAN
A0232012E GUAN ZHILING
A0231904M KUAN JU LIN

Background & Problem Statement

- E-commerce has been a booming market these years.
- As shown in the figure, people are more and more accustomed to shopping online.
- In Singapore, about 3.3 million people shop in the e-commerce market and the expected revenue from the market in 2021 is USD 2,793m.
- To predict whether a user is going to make a purchase based on data from a user's current session.



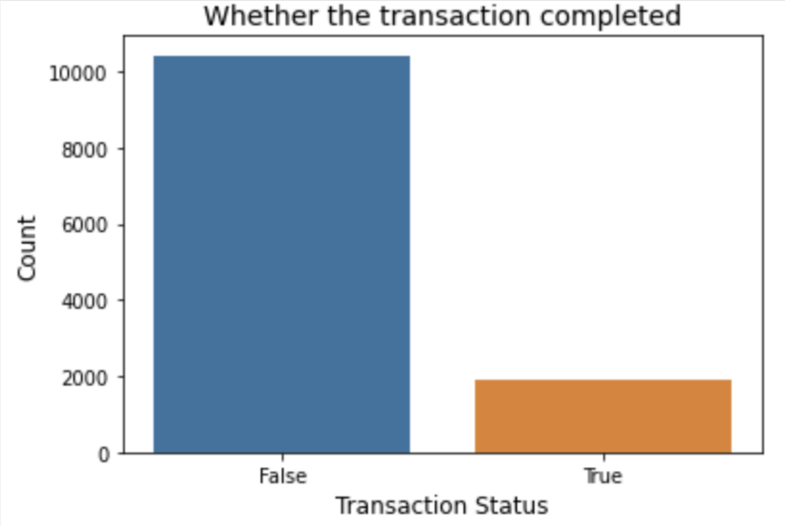
Exploratory Data Analysis

Missing Data

	Total	Percent		Total	Percent
Administrative	0	0.0	SpecialDay	0	0.0
Administrative_Duration	0	0.0	PageValues	0	0.0
Weekend	0	0.0	ExitRates	0	0.0
VisitorType	0	0.0	BounceRates	0	0.0
TrafficType	0	0.0	ProductRelated_Duration	0	0.0
Region	0	0.0	ProductRelated	0	0.0
Browser	0	0.0	Informational_Duration	0	0.0
OperatingSystems	0	0.0	Informational	0	0.0
Month	0	0.0	Revenue	0	0.0

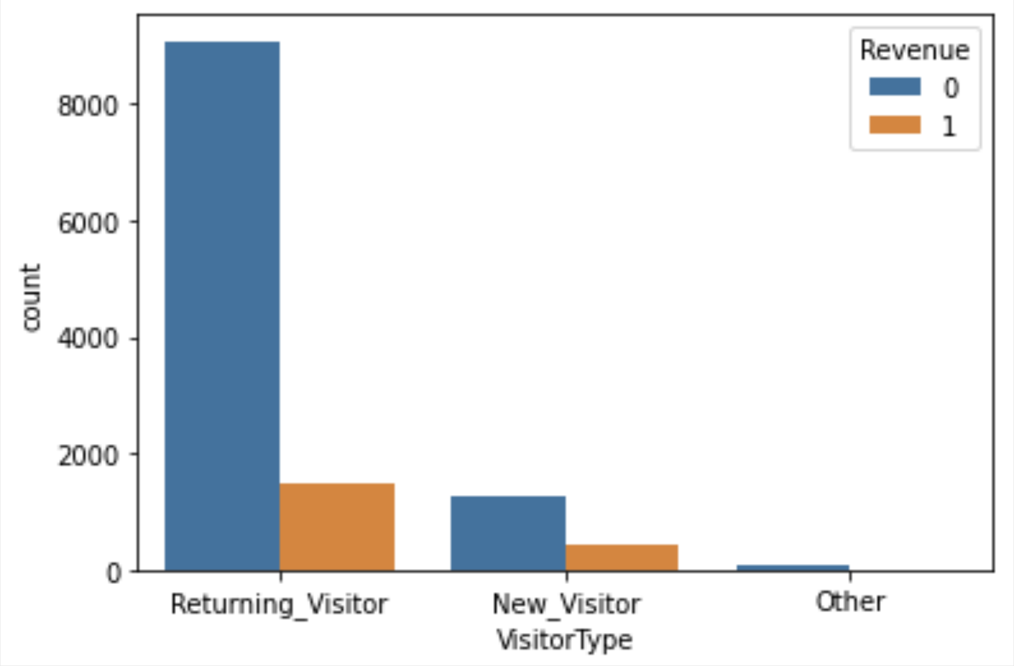
- There are no missing data

Target Variable

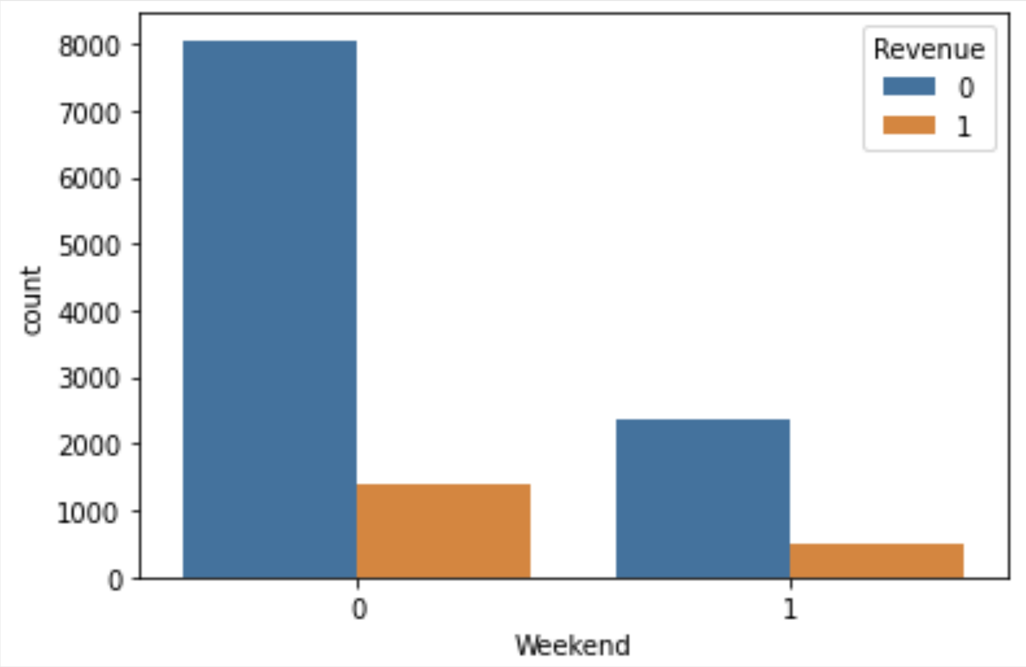


- Dataset is imbalance
- Uncompleted transaction: 0.845255
- Completed transaction: 0.154745

EDA: Bivariate Analysis

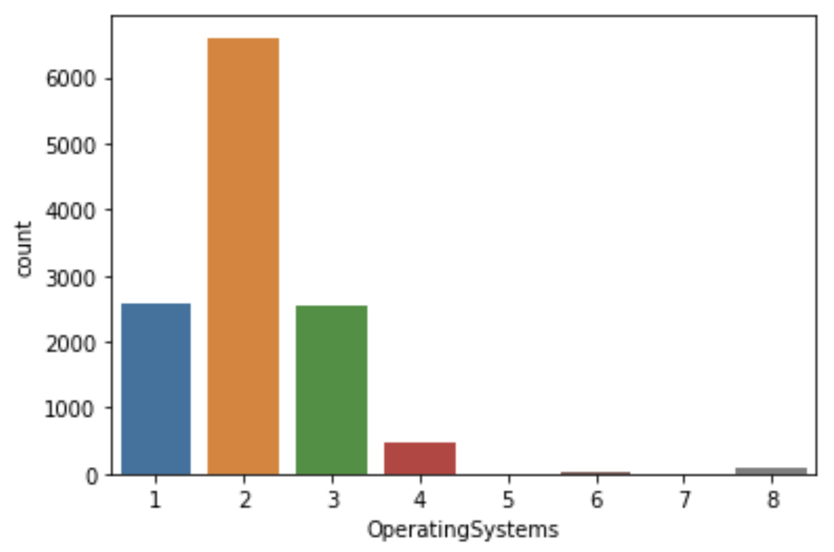


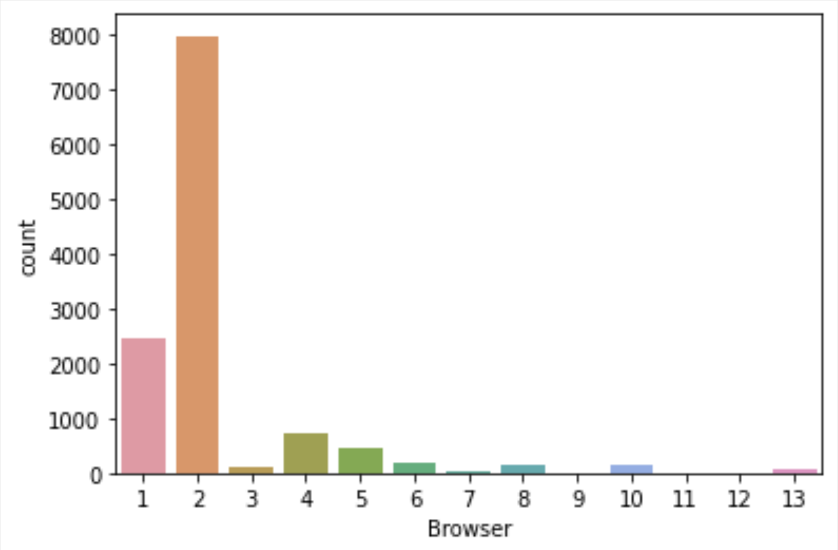
Find the relationship between "Visitor Type" and "Revenue"

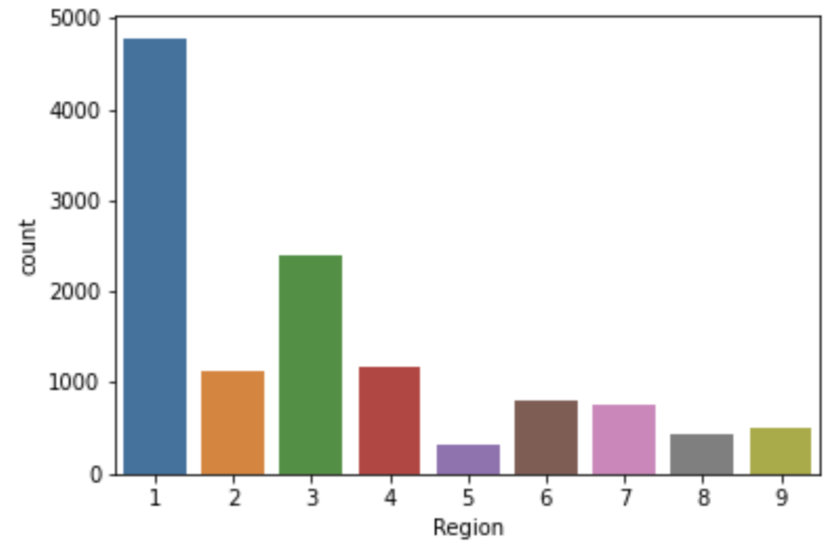


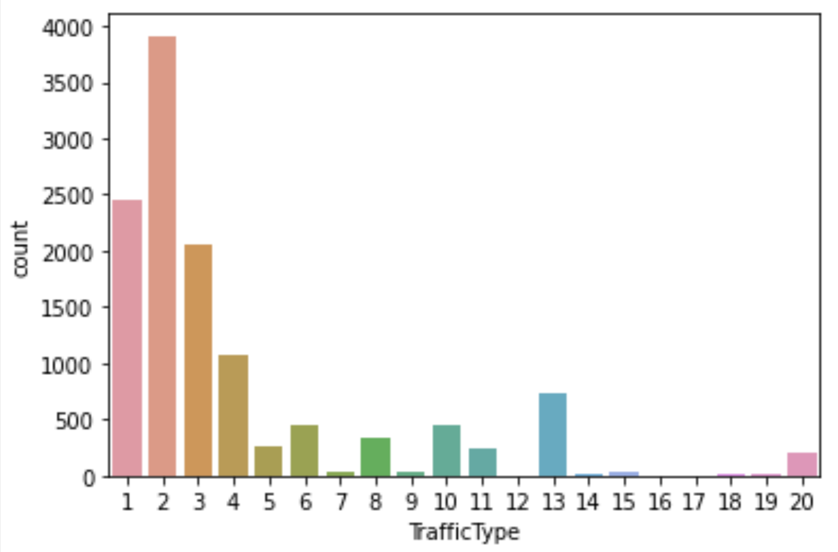
Find the relationship between "Weekend" and "Revenue"

Imbalanced Categorical Features

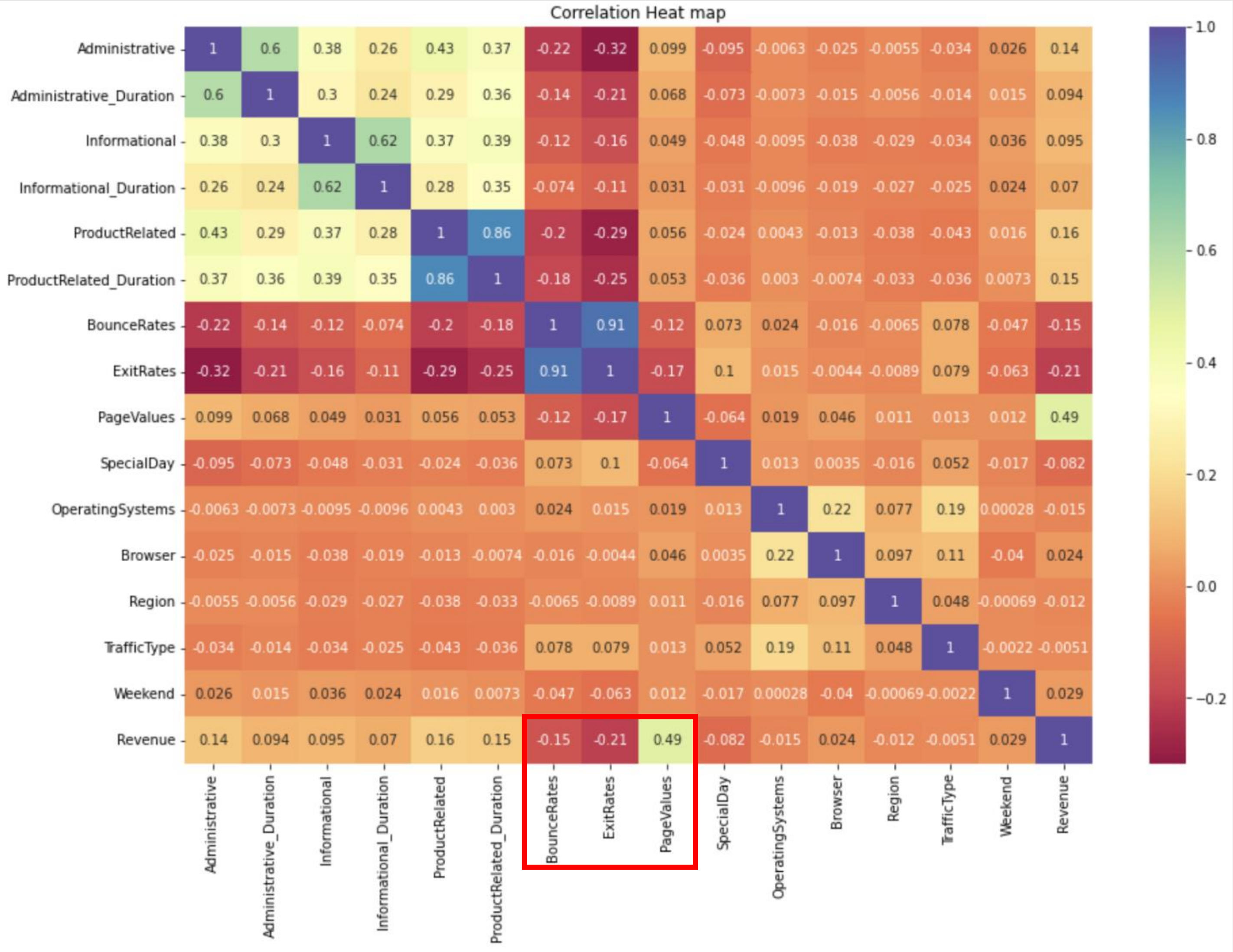






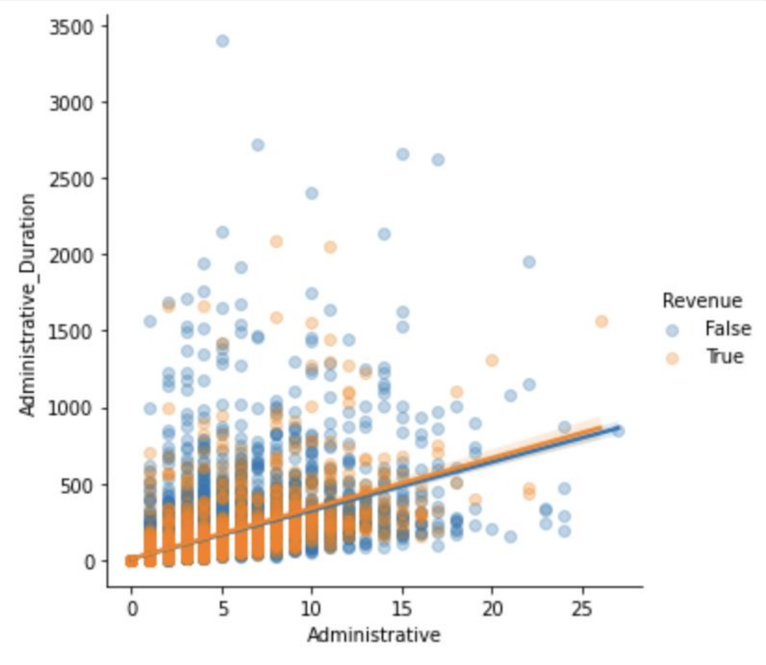


EDA: Heat Map

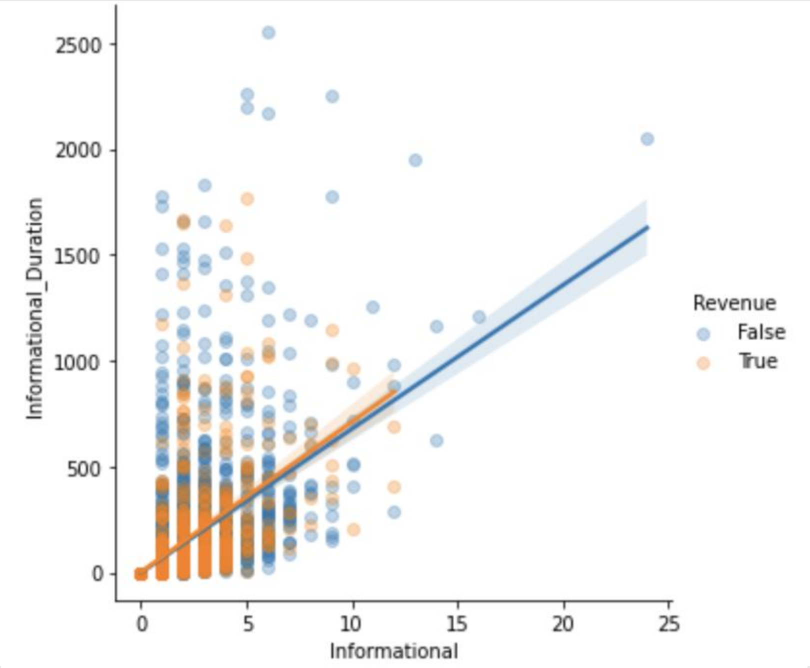


- “Revenue” and “PageValues” has a positive relationship
- “Revenue” and “BouceRates” has a negative relationship
- “Revenue” and “ExitRates” has a negative relationship

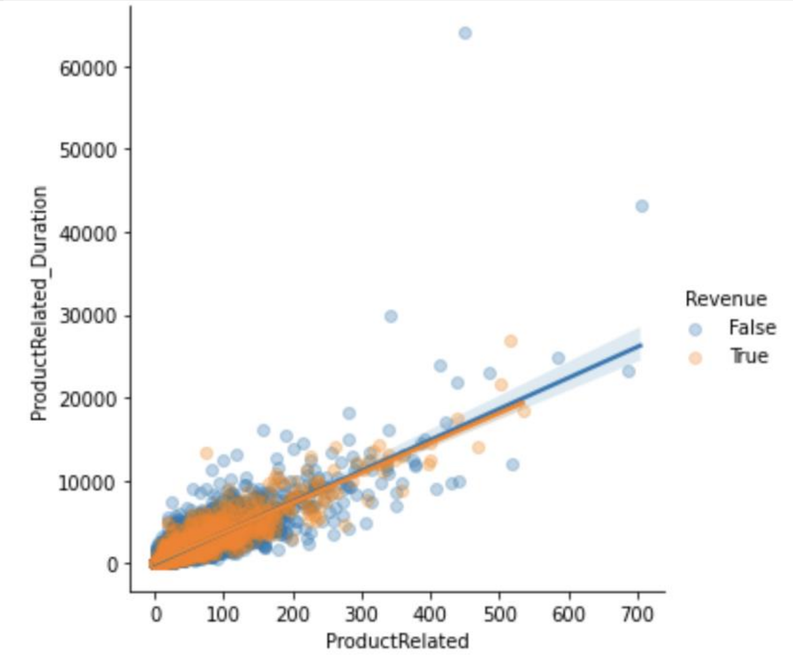
EDA: Correlation



Compare "Administrative" and "Administrative_Duration"



Compare "Informational" and "Informational_Duration"



Compare "ProductRelated" and "ProductRelated_Duration"

Data preprocessing

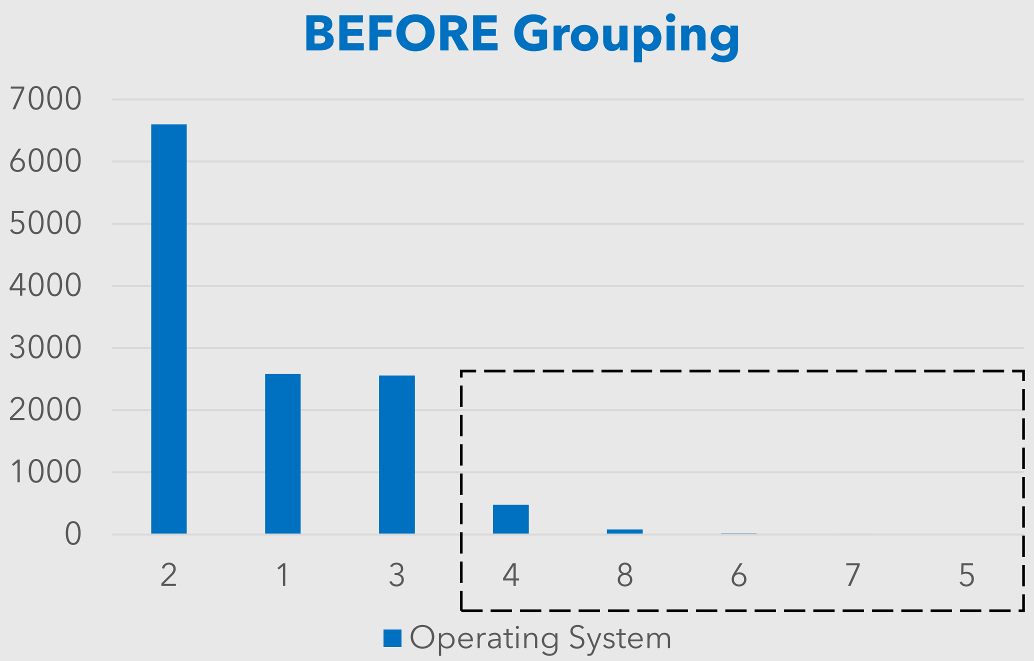
Feature Engineering: Average page view duration
Calculate the time spent by each customer on one page

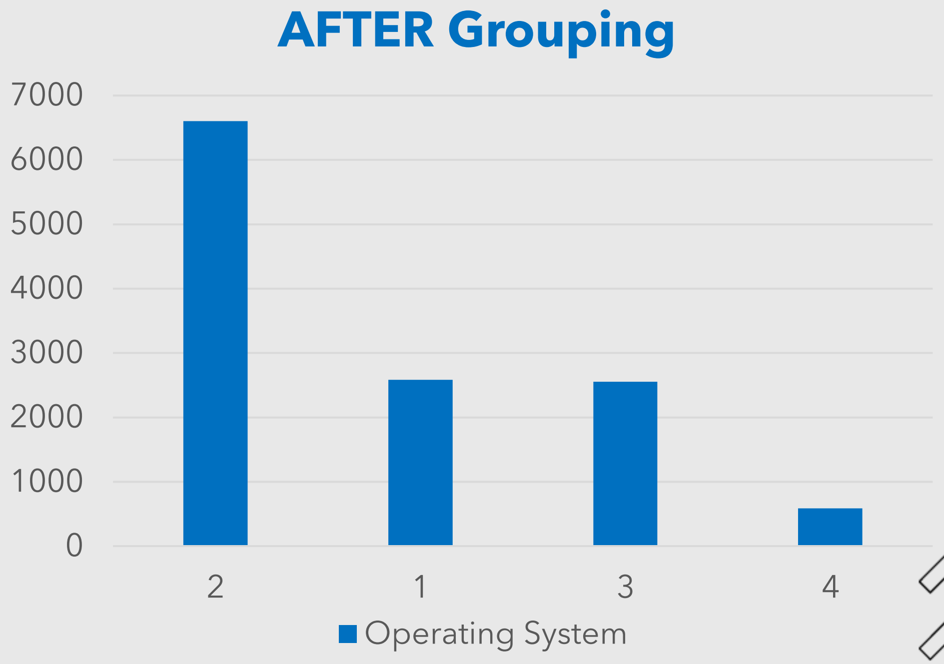
$$\frac{\text{Administrative\_Duration}}{\text{Administrative}} \qquad \frac{\text{Informational\_Duration}}{\text{Informational}} \qquad \frac{\text{ProductRelated\_Duration}}{\text{ProductRelated}}$$

1. Replace with 0 if number of pages visited is 0.
2. Drop the original columns
3. New feature names: Avg_Administrative_Duration, Avg_Informational_Duration, Avg_ProductRelated_Duration

Data preprocessing

Feature Engineering: Categories Grouping
For Categorical variables with many categories, we group the categories with lower counts into one.



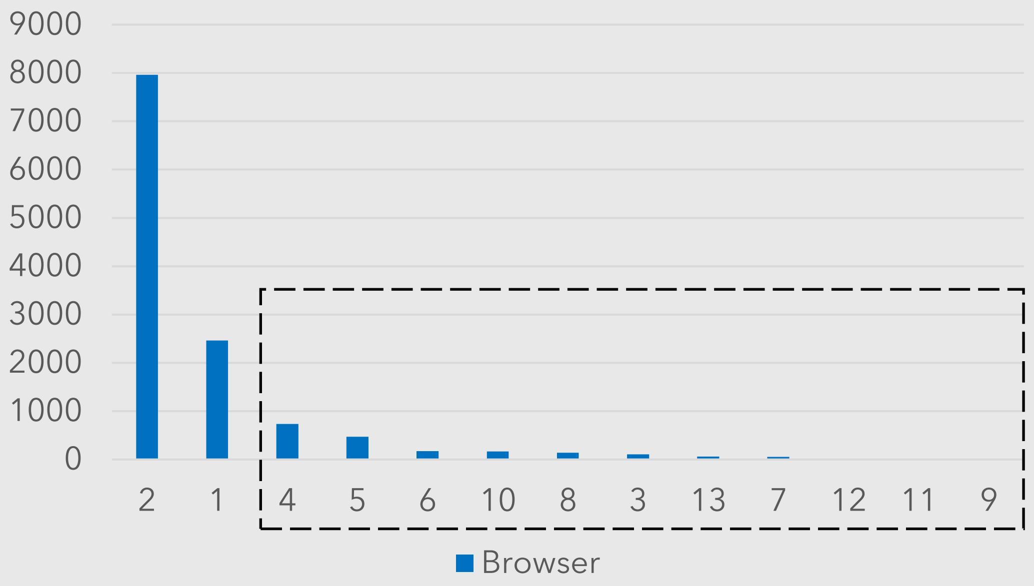


For example, we group Operating System's category 8, 6, 7, 5 into category 4, so there are only 4 categories left in Operating System.

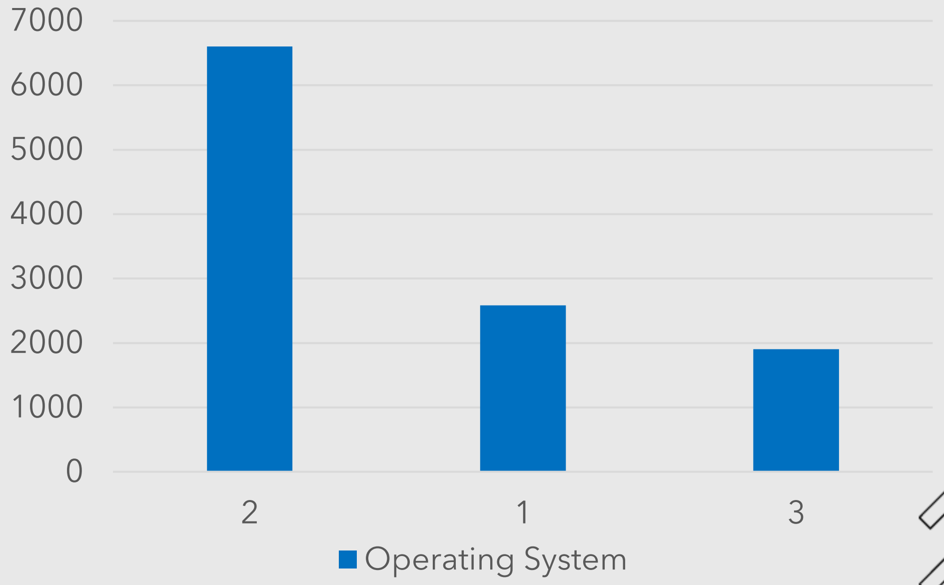
Data preprocessing

Feature Engineering: Categories Grouping
For Categorical variables with many categories, we group the categories with lower counts into one.

BEFORE Grouping



AFTER Grouping

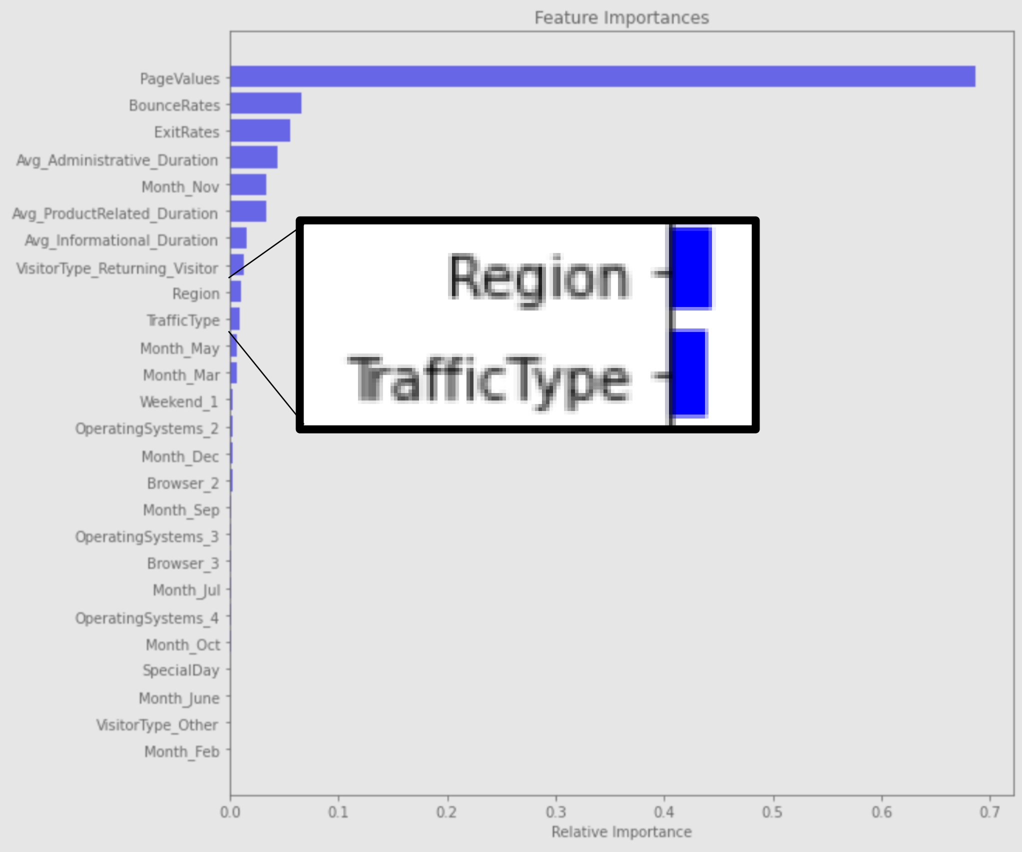


We also group categories 3,4,5,6,7,8,9,10,11,12,13 in Browser Type into category 3.

Feature Selection

Dropped **"Traffic type"** and **"Region"** because:

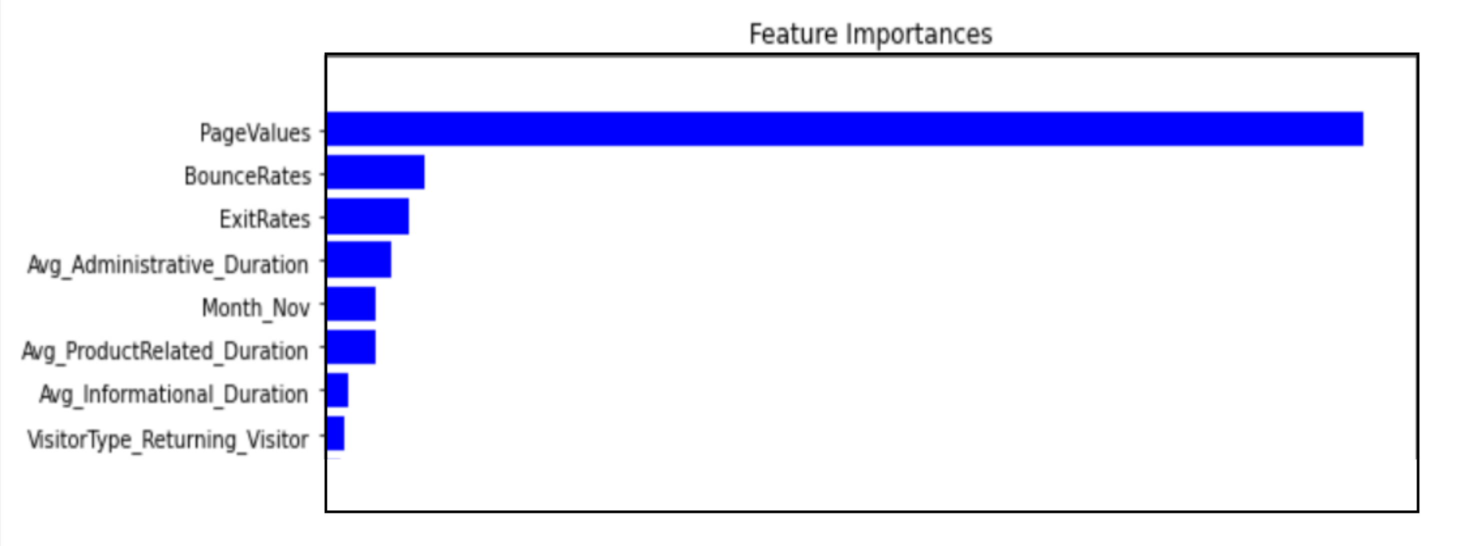
1. Lower correlation to "Revenue" according to heatmap.
2. Low feature importance according to the random forest naïve model we ran.
3. There are many categories in both "Traffic type" and "Region".



Feature Selection

**THE TOP 8
INFLUENTIAL FEATURES:**

PageValues
BounceRates
ExitRates
Avg_Administrative_Duration
Month_Nov
Avg_ProductRelated_Duration
Avg_Informational_Duration
VisitorType_Returning_Visitor



Feature Selection

INDEPENDENT VARIABLES

BounceRates
ExitRates
PageValues
SpecialDay
Avg_Administrative_Duration
Avg_Informational_Duration
Avg_ProductRelated_Duration

NUMERIC

Month
OperatingSystems
Browser
VisitorType
Weekend

CATEGORICAL

DEPENDENT VARIABLES

Revenue

Data preprocessing

Categorical Variables

One hot encoding

We used “Pandas get_dummies” and “Scikit-learn label encoder” to encode categorical features into dummy variables

Numeric Variables

Standardization

We used “Scikit-learn StandardScaler” to calculate the Z-score of the numeric variables

Data preprocessing

Train test split

We split the data into training and validation data set with the ratio of 0.7 and 0.3.

Imbalanced data handling

There are less customers who had finalized with transactions, so we use "Synthetic Minority Oversampling Technique" (SMOTE) to increase the data points of the purchase class.

BEFORE SMOTE

```
Train purchase class = 1336
Train non purchase class = 7295
Test purchase class = 572
Test non purchase class = 3127
```

AFTER SMOTE

```
Train purchase class = 7295
Train non purchase class = 7295
Test purchase class = 572
Test non purchase class = 3127
```

Model Introduction

Ensemble learning:
by combining multiple weak learners, a strong learner is created.

Boosting

LightGBM
XGBoost

Bagging

Random Forest

Why LightGBM?

- LightGBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other learning tasks.
- LightGBM carries out **leaf-wise growth** that results in more loss reduction and in turn higher accuracy while being faster.

Advantages:

- Faster training speed and higher efficiency
- Lower memory usage
- Better accuracy
- Support of parallel and GPU learning
- Capable of handling large-scale data

Hyperparameter Tuning

GridSearchCV

It helps to loop through predefined hyperparameters and fit the estimator (model) on the training set. So, in the end, we can select the best parameters from the listed hyperparameters.

Pros vs Cons

Manually input parameter; trial and error
BUT save computation time than Randomized Search

Implementation

grid = GridSearchCV(estimator=lightgbm, param_grid=params, scoring='roc_auc', cv=5, verbose=1)

Parameters Tuned

'max_depth': [1, 3, 6, 8, 10]
'num_leaves':[30, 40, 50]
'learning_rate': [0.01, 0.05, 0.1]
'min_child_samples':[10, 15, 20]

LightGBM – parameters after tuning

Parameters tuned	Why this parameter	Best parameter
max_depth	To prevent overfitting, we set as 10.	10
num_leaves	We increased the number of leaves to better capture the patterns of data.	50
learning_rate	Control the shrinkage rate.	0.1
min_child_samples	We loosen the minimum number of data needed for a leaf to 10 to better capture the patterns of data.	10

LightGBM - evaluation

Accuracy on the test set: 0.870
AUC on the test set : 0.918

Classification report:

	precision	recall	f1-score	Support
0	0.95	0.89	0.92	3127
1	0.56	0.75	0.64	572
accuracy			0.87	3699
macro avg	0.75	0.82	0.78	3699
weighted avg	0.89	0.87	0.88	3699

Why XGBoost?

REASON 1

Traditional boosting methods are prone to overfitting, but XGBoost overcomes this problem because of the **optimized regularization.**

REASON 2

The power of XGBoost has been proved and recognized by the community. It is often adopted by the winning teams of Data competitions.

Hyperparameter Tuning

Implementation

grid = GridSearchCV(estimator=xgboost, param_grid=params, scoring='roc_auc', cv=5, verbose=1)

Parameters Tuned

'max_depth': [24, 26, 28, 30]
'n_estimators':[100, 150, 200, 250]
'learning_rate': [0.001, 0.01, 0.1]
'min_child_weight':[0.5, 1, 2]

XGBoost-Parameters

Parameters tuned	Why this parameter	Best parameter
n_estimators	Control the number of boosting rounds, to be better capture the underlying pattern.	250
max_depth	Maximum tree depth for base learner. Help us better capture the underlying pattern as well.	30
learning_rate	Control the shrinkage rate.	0.1
min_child_weight	Minimum sum of instance weight(hessian) needed in a child. Loosen to better capture the patterns of data.	0.5

XGBoost - evaluation

Accuracy on the test set: 0.874
AUC on the test set : 0.915

Classification report:

	precision	recall	f1-score	Support
0	0.95	0.90	0.92	3127
1	0.57	0.72	0.64	572
accuracy			0.87	3699
macro avg	0.76	0.81	0.78	3699
weighted avg	0.89	0.87	0.88	3699

Why Random Forest?

REASON 1

Decision trees are helpful and intuitive ways to classify data, but they are prone to overfitting. To reduce overfitting, we use random forests.

REASON 2

By creating more decision tree, each tree receives a vote in terms of how to classify. In this way, the classification returned by the most trees is very likely to be the most accurate.

Hyperparameter Tuning

Implementation

```
grid = GridSearchCV(estimator=rf_clf,
param_grid=params,
scoring='roc_auc', cv=5, verbose=1)
```

Parameters Tuned

```
'max_depth': [24, 26, 28, 30, 32]
'n_estimators':[100, 200, 250, 300, 350]
'max_samples': [0.1, 0.5, 0.9]
```

Random Forest-Parameters

Parameters tuned	Why this parameter	Best parameter
estimators	The number of trees in the forest.	250
max_depth	The maximum depth of the tree. Default is nodes are expanded until all leaves are pure.	32
max_samples	The ratio of samples to draw from X to train each base estimator.	0.9

Random Forest-Evaluation

Accuracy on the test set: 0.869
AUC on the test set : 0.916

Classification report:

	precision	recall	f1-score	Support
0	0.95	0.89	0.92	3127
1	0.56	0.75	0.64	572
accuracy			0.87	3699
macro avg	0.75	0.82	0.78	3699
weighted avg	0.89	0.87	0.88	3699

Evaluation Metrics

AUC (Area Under Curve)

Measures the **ability** of a classifier to distinguish between **classes**

Care equally about positive and negative classes

Recall

TP/ (TP+**FN**)

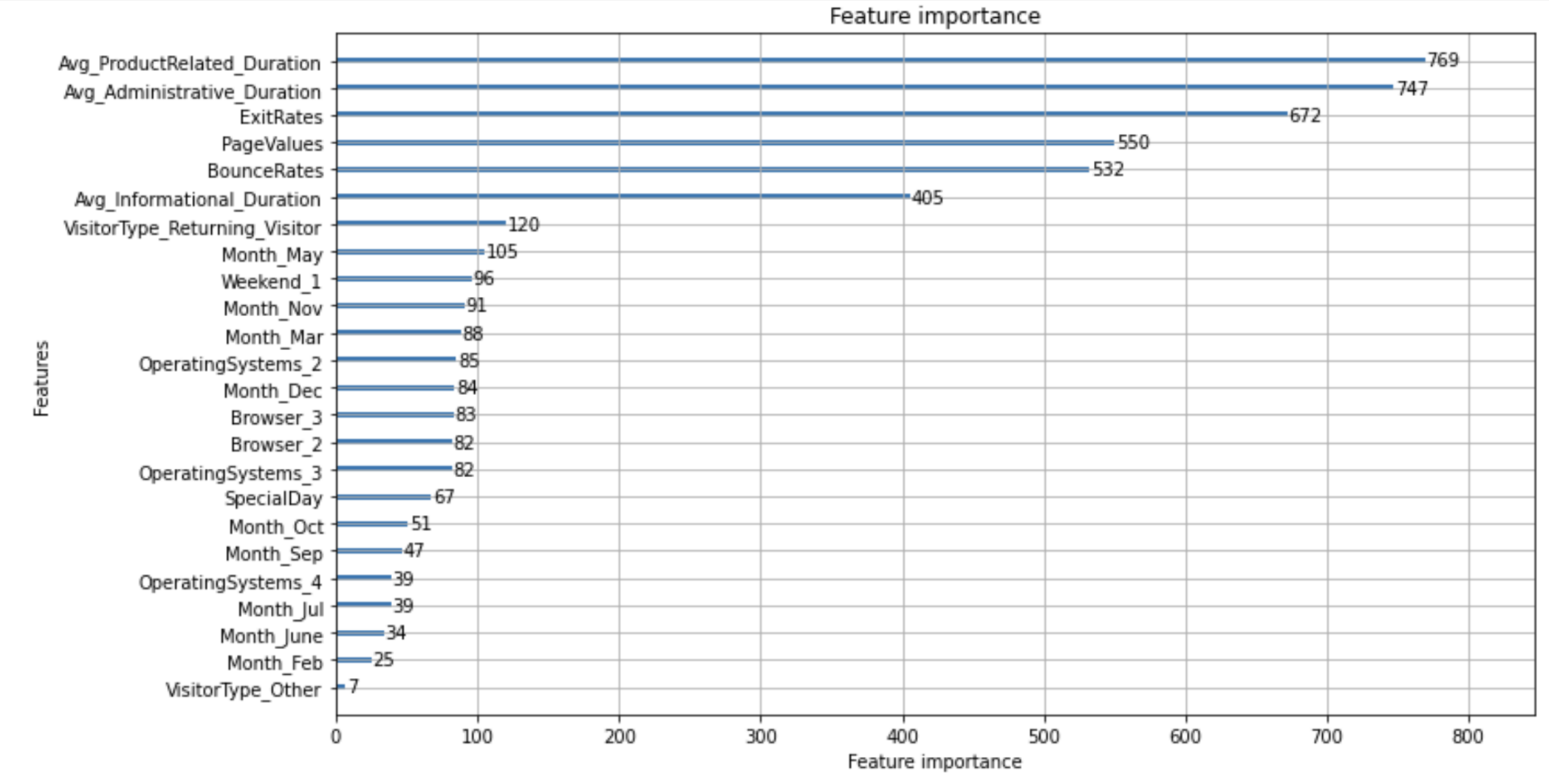
FN: user with buying intention wrongly labeled as non-buy
FP: user not buying wrongly labeled as buying

Tradeoff: cost of **losing** potential customer vs waste of **marketing efforts**
Assumption: **Customer churn is more costly**

Model Comparison

	LightGBM		XGBoost		RandomForest	
	Train	Test	Train	Test	Train	Test
Accuracy	0.957	0.870	1.000	**0.874**	1.000	0.867
ROC-AUC	0.994	**0.918**	1.000	0.915	1.000	0.916
Precision	0.947	0.559	1.000	**0.573**	1.000	0.557
Recall	0.968	**0.748**	1.000	0.724	1.000	**0.748**
F1 score	0.957	0.640	1.000	**0.640**	1.000	0.638

Recommendation by features importance



Recommendation by features importance

Webpage related features

Improve UI/UX design

Conduct A/B testing

Visitor Type

Precision marketing strategy

Customer churn & retention

Seasonality in purchase

Time-varied promotional campaigns

Thank you for your attention