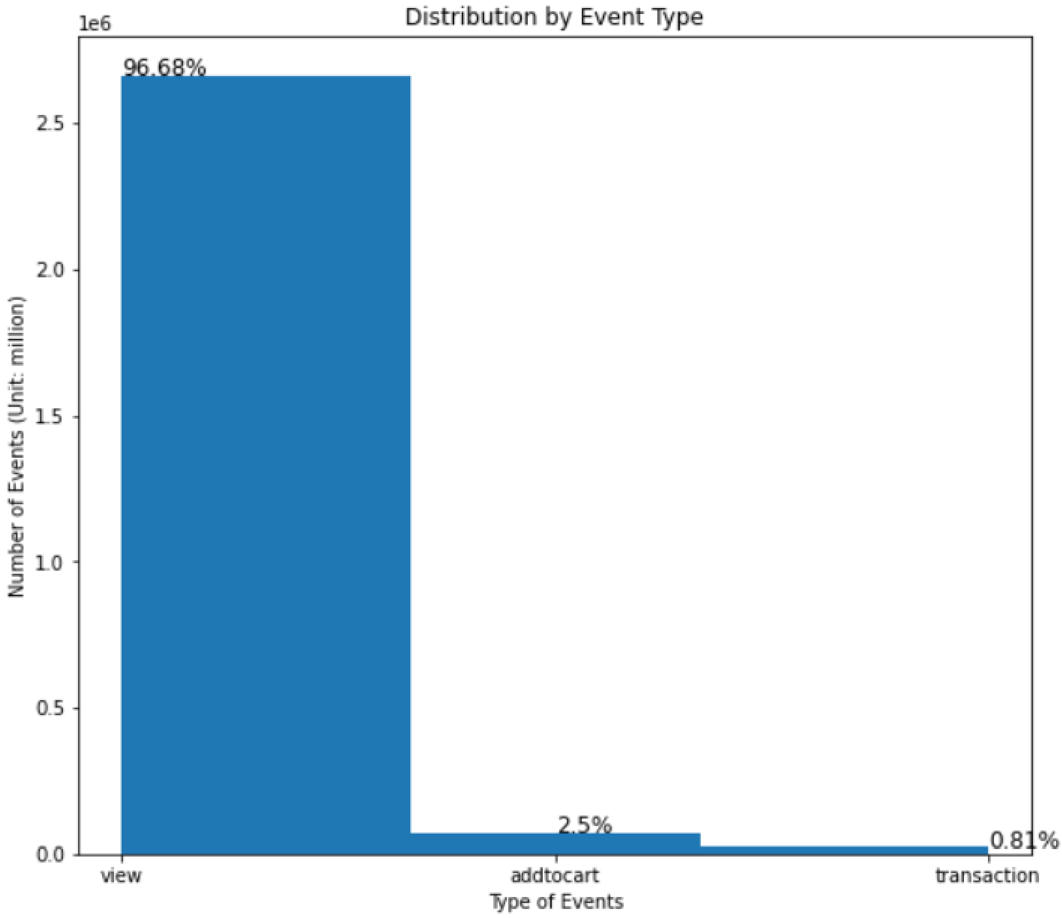
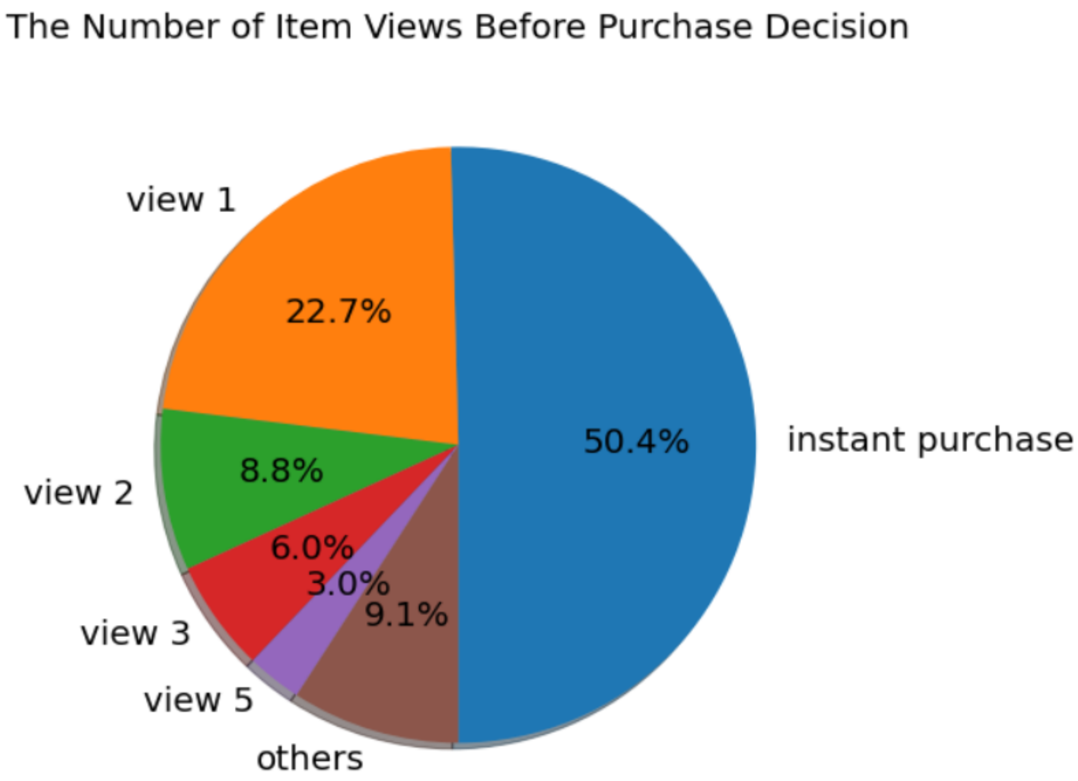
Insights from data

- The timestamp portion is in Unix format
- Visitor Id is the unique user currently browsing the website
- Event is what the user is currently doing in that current timestamp
- Transaction ID will only have value if the user made a purchase as shown below, Else it will be Null



When analyzing the distribution of events, 'View' occupies 96.67%, 'Add to cart' 2.52%, 'Transaction' 0.81%.

-
- Start Date of Dataset: 2015-05-03 03:00:04.384
- End Date of Dataset: 2015-09-18 02:59:47.788

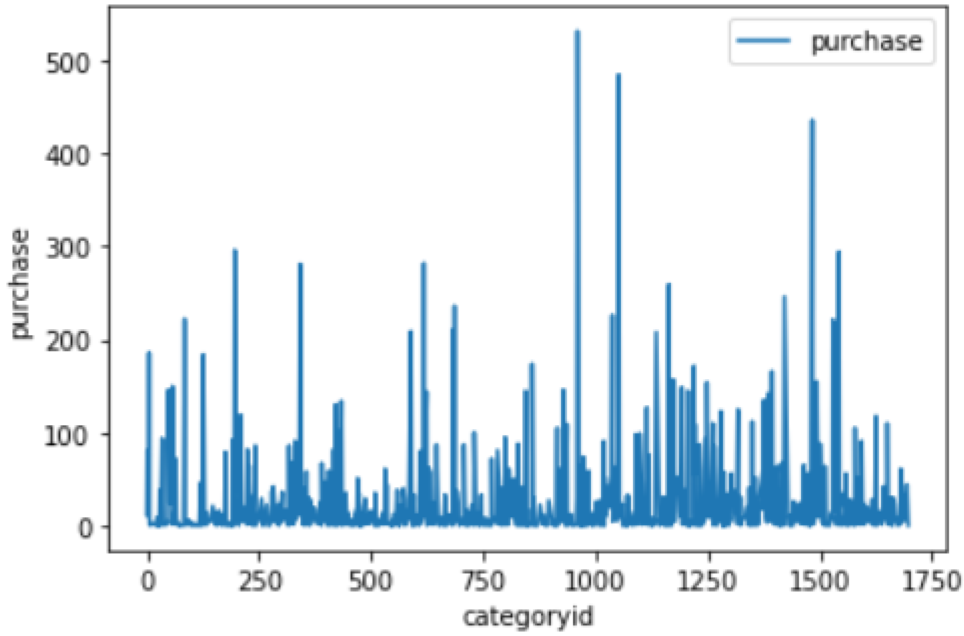


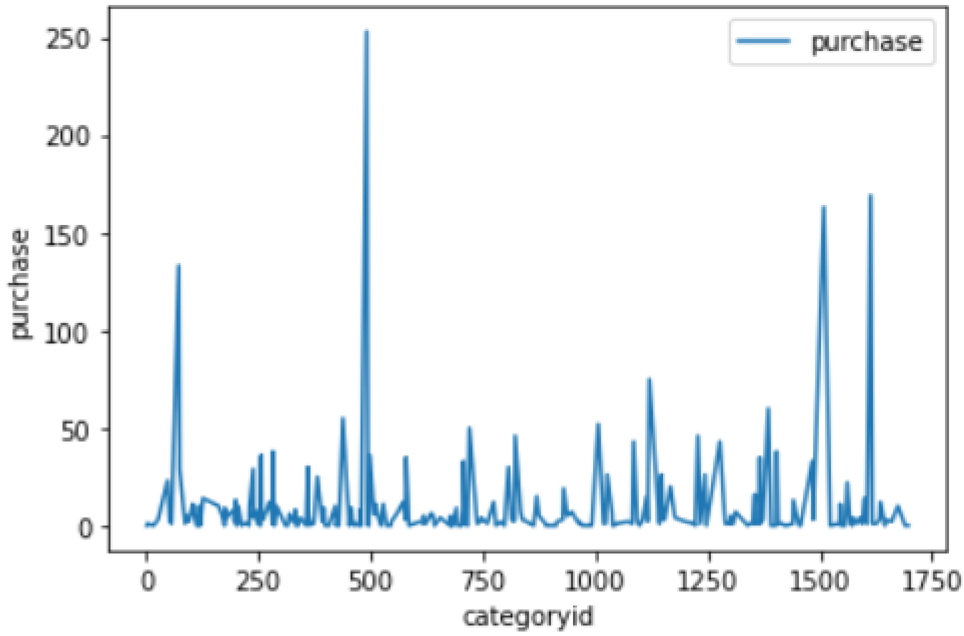
When analyzing the item view numbers, I found 50.4% of transactions were made without a more-than-once view: a visitor checked an item, added to cart and checked out.

About 30% of transactions were made after a buyer viewed an item once or twice.

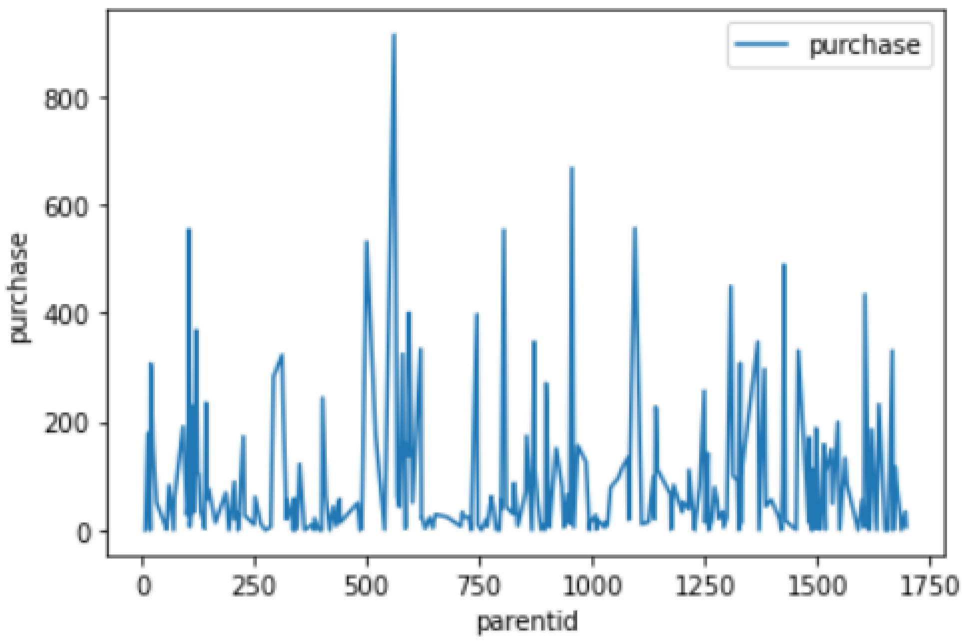
In summary, 80% of total transactions were made after less-than-three-times item views.

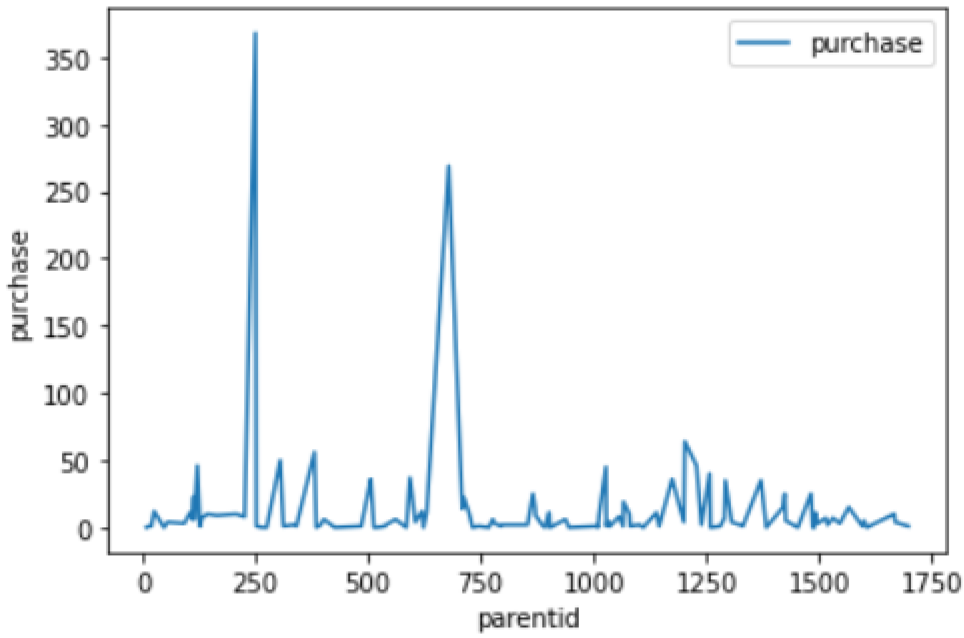
Relationship between feature and purchase



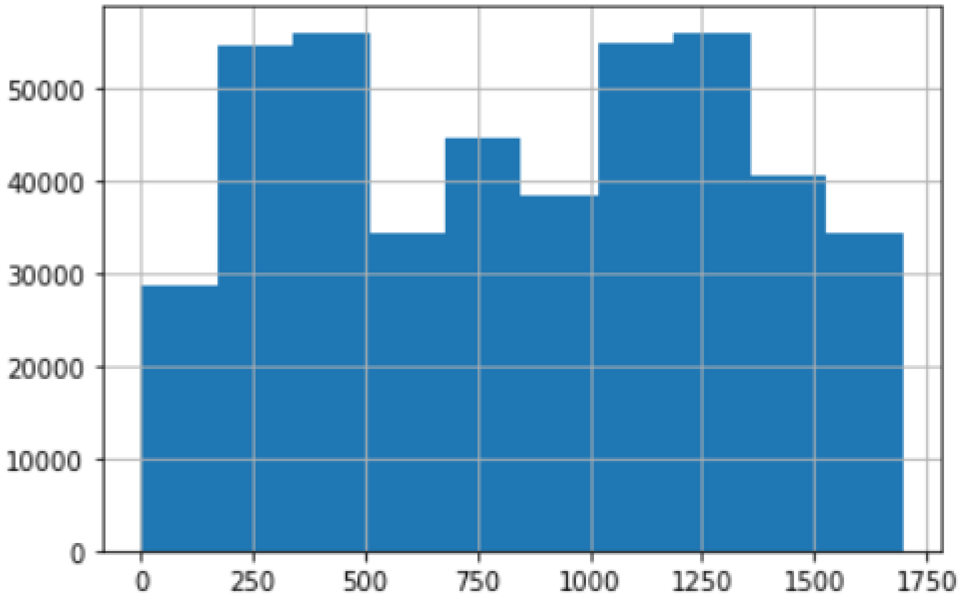


There is no general trend in the number of sales along category id.

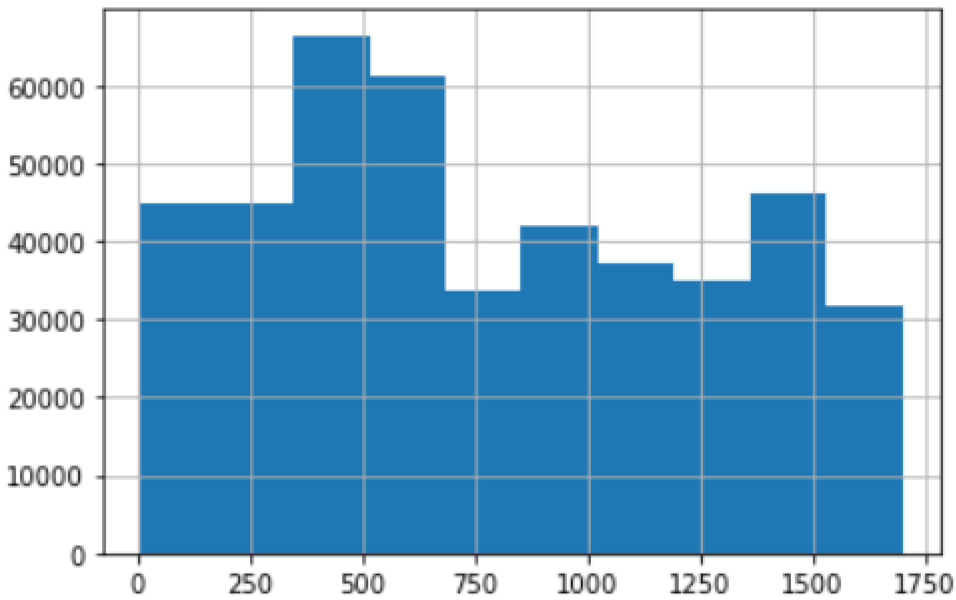




There is no general trend in the number of sales along parent id.

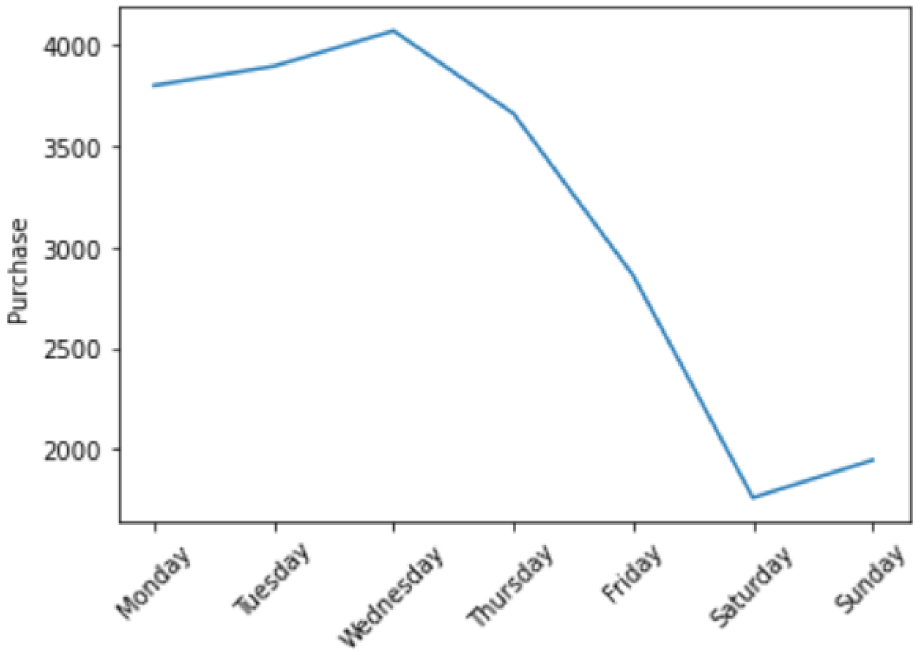


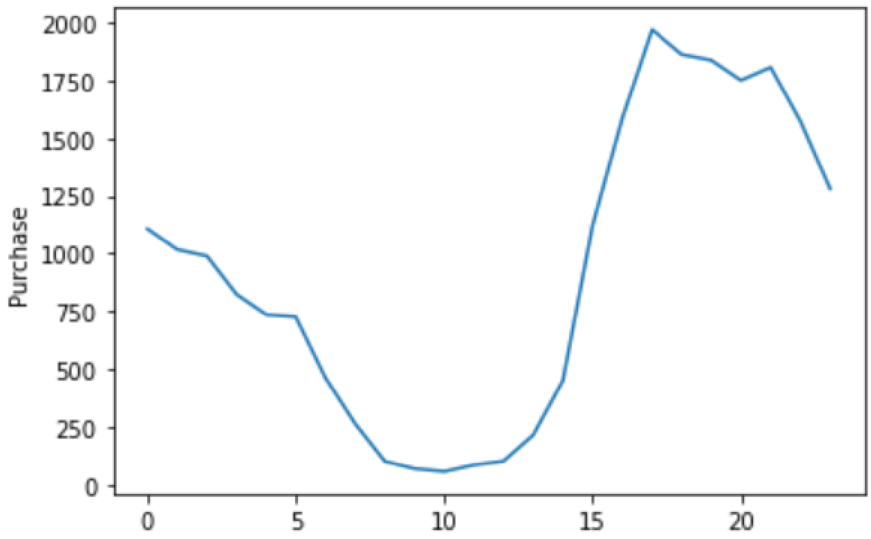
distribution by category id



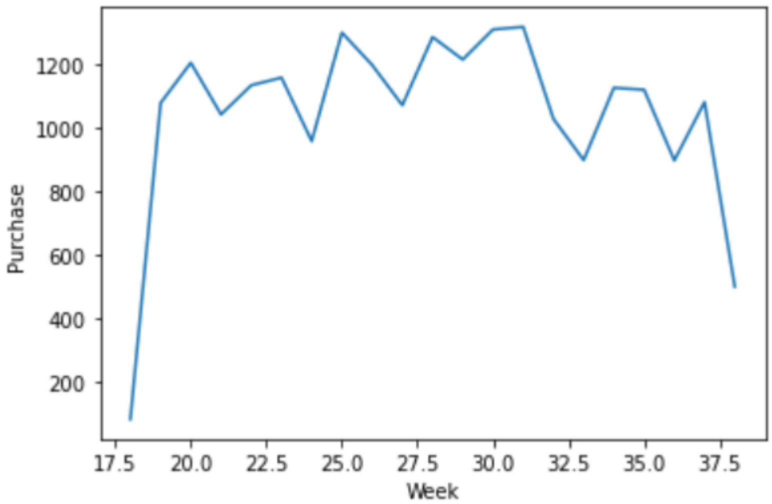
There was no clear relationship between category and transaction (sales)

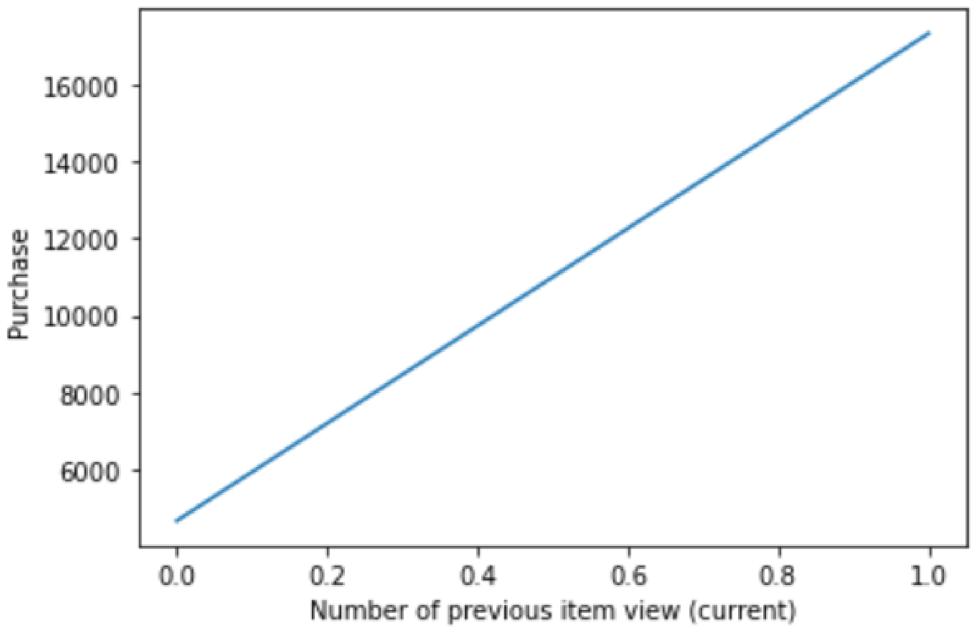
Is there any relationship between View Time and Transaction?



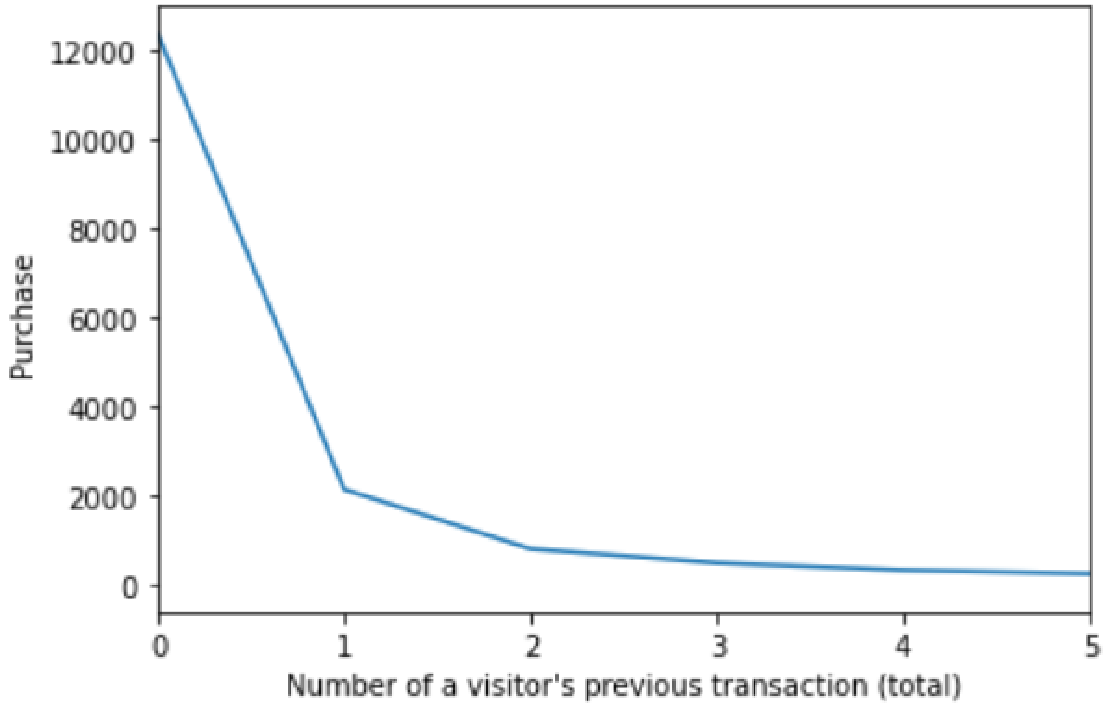


Hours





We can see that there is a very good relation between Number of previous view and purchase



For modeling we should use (dayofweek, hour, previous view, previous transaction)

Recommendation system:-

simply we can offer our visitors a list of what previous visitors bought together with the item they are currently viewing

```
recommender(302422, purchased_items)
```

```
{12836, 15335, 25353, 80582, 105792, 200793, 237753, 317178, 380775, 400969}
```

It’s working by firstly grouping every similar object or (every bought-together product) in one list

```
purchased_items[:5]
```

```
[[356475],
 [15335,
  380775,
  237753,
  317178,
  12836,
  400969,
  105792,
  25353,
  200793,
  80582,
  302422],
 [81345],
 [150318, 49521],
 [310791, 299044]]
```

The recommender function will help the visitor by recommending (similar products) which other visitors bought together, which will increase the revenue.

Modeling part

- choosing 25,000 samples only from the non- transaction records (under sampling technique to not deal with unbalanced data)

- We can use machine learning to predict if the visitor will make a purchase decision or not
- that will increase the revenue if we make offers for predicted not making a purchase

Selected features

After looking to the correlation between all features and our label (purchase)those features have been chosen for modeling.

```
modeling_data.head()
```

	dayofweek	hour	previous_view	previous_transaction
0	6	3	0	0
1	6	3	0	0
2	6	3	1	0
3	6	3	0	0
4	6	4	0	0

- After looking carefully to the data, Decision Tree have been chosen for its advantages to the numeric data

- using Grid search to choose best hyper-parameters

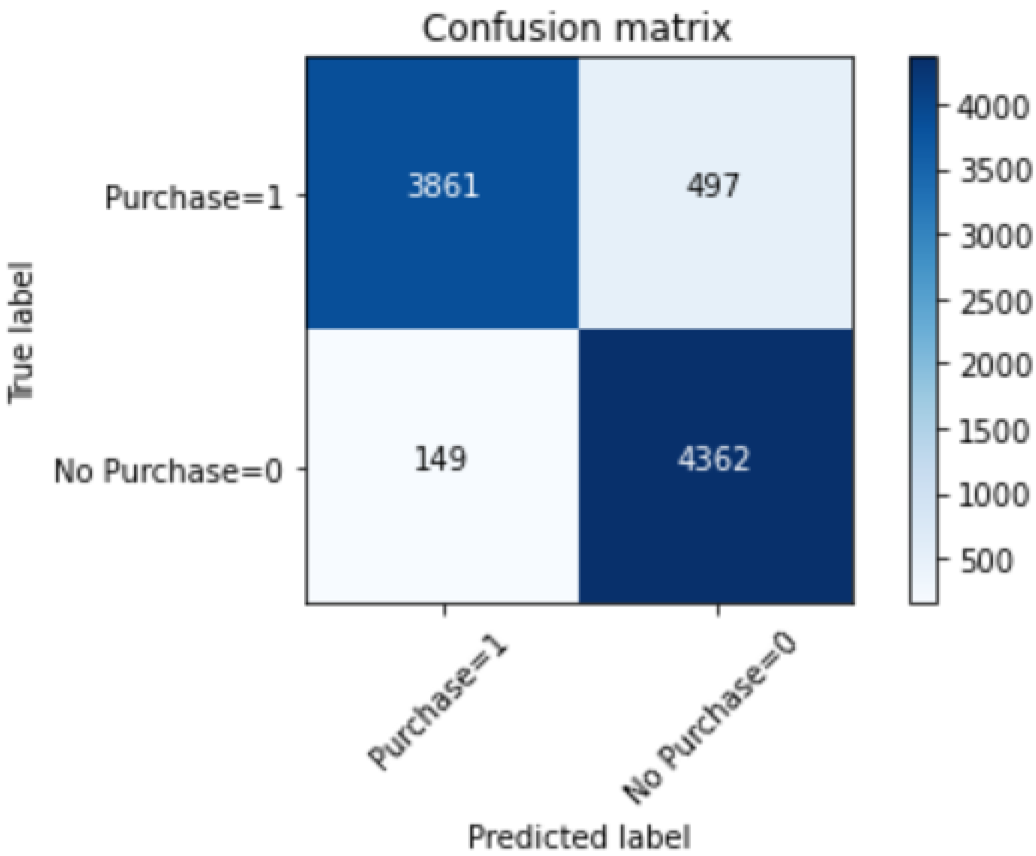
```
GridSearchCV(estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': array([ 2,  4,  6,  8, 10, 12])})
```

- Best hyper-parameters:-
  criterion = "gini"
  max_depth = "2"

Model evaluation
choosing F1 score for evaluation beside precision and recall to ensure that the results are matching business needs.

Classification Report

	precision	recall	f1-score	support
0	0.90	0.97	0.93	4511
1	0.96	0.89	0.92	4358
accuracy			0.93	8869
macro avg	0.93	0.93	0.93	8869
weighted avg	0.93	0.93	0.93	8869



At conclusion we can increase revenue by

1- We can increase the probability of purchase decision by giving offers for those who has predicted probability near 50%

2- we also can increase revenue by recommending similar products

3- Making offers in the days which has less number of purchase