Can We Predict the Amount of a Black Friday Purchase?

Edgar Garcia
Thinkful Data Science Bootcamp

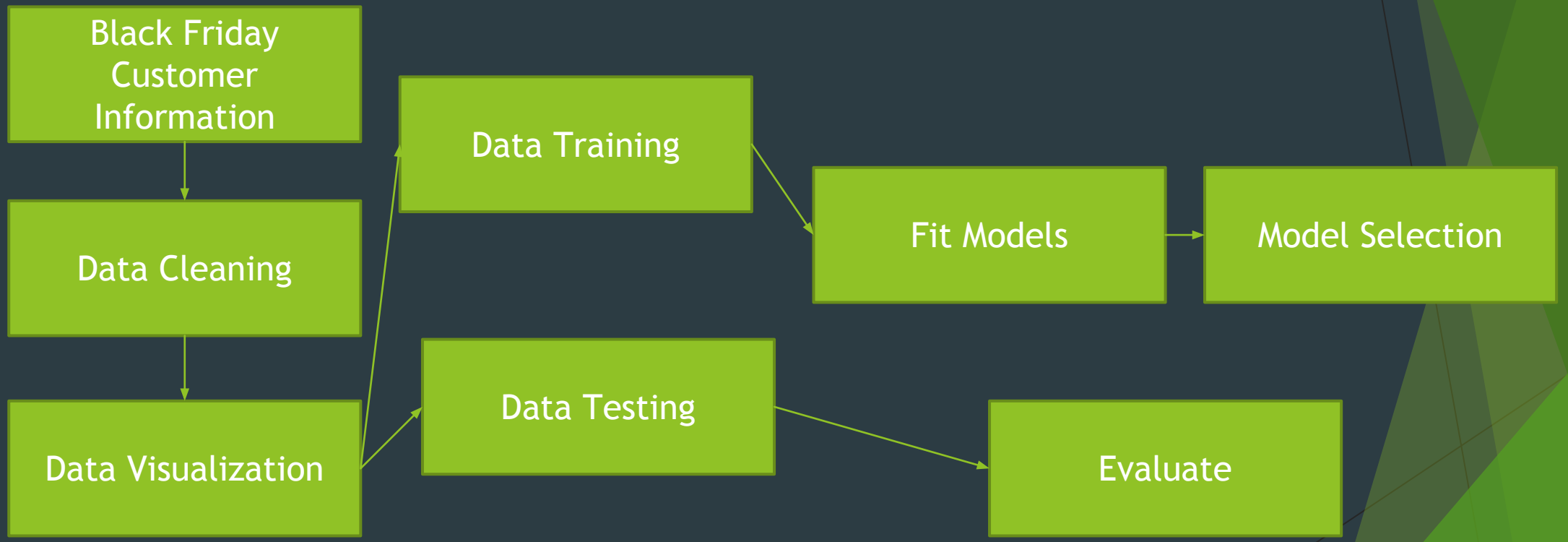
Black Friday

- Black Friday is the day after Thanksgiving in which retailers offers special deals for consumers for them to spend their hard earned money on things they most likely don’t need.
- This is good way for retailers to acquire new customers and generate some extra sales during the year.

Experiment

- In this experiment, I will try to predict the amount of a Black Friday purchase based on customer information such as user ID, product ID, gender, age, marital status, city, amount of time living in the city, and product category.
- The data was chosen from Kaggle and was generated this year, so I will assume it is recent and based on purchases from one year. The information was collected from customer transactions from one unspecified store.
- This information could prove useful to a business by helping to determine who are their target customers and to predict how much a specific customer will spend during Black Friday.

Data Processing



The Data

- I had to do some data cleaning for some columns such as age, gender, and city category.
- Unfortunately, we do know any specific details of the city or the type of product that was purchased. I will have to use the data as is in order to try to predict the amount of purchase.
- Some features will be not be very useful due to the lack of details given.
- Product ID and Customer ID alone will not help us in predicting the purchase amount, but I will group them by average purchase amount to determine if these features will help in improving my prediction model.

```
In [15]: #Recoding the catergorical values using replace
         Age_cleanup = {'Age':   {'0-17': 0, '18-25': 1,'26-35': 2, '36-45': 3, '46-50': 4, '51-55': 5,'55+':6}}
         df.replace(Age_cleanup, inplace=True)
         df.head(20)

         City_cleanup ={'City_Category':   {'A': 0, 'B': 1,'C': 2}}
         df.replace(City_cleanup, inplace=True)

         Gender_cleanup = {'Gender': {'M': 1, 'F': 0}}
         df.replace(Gender_cleanup, inplace=True)
```

The Data

- Our target variable will be purchase since we are trying to predict the purchase amount per customer. Our features will include gender, age, occupation, city category, stay in current city years, product category 1, product category 2, product category 3, average purchase per product, and average purchase per user.

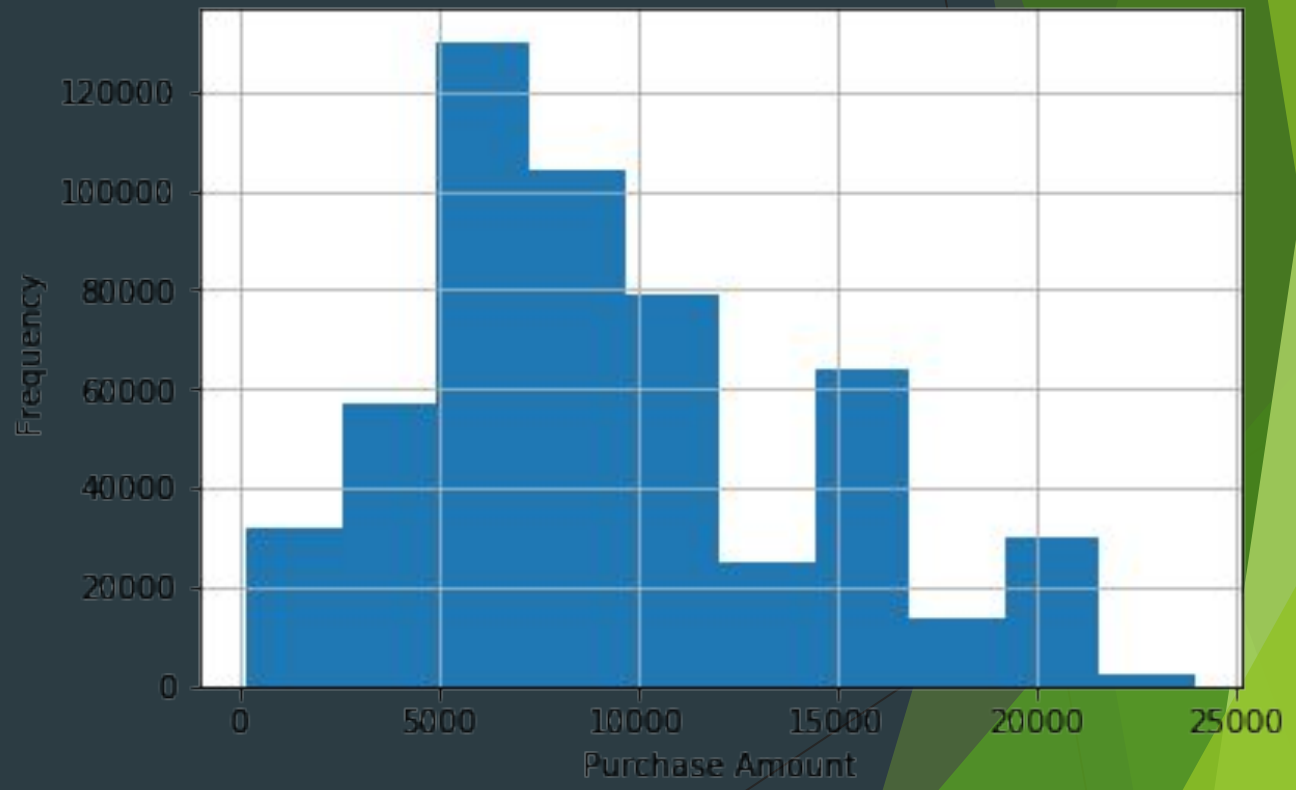
```
In [96]: temp.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 403182 entries, 0 to 403181
Data columns (total 13 columns):
User_ID                       403182 non-null int64
Gender                        403182 non-null int64
Age                           403182 non-null int64
Occupation                    403182 non-null int64
City_Category                 403182 non-null int64
Stay_In_Current_City_Years    403182 non-null object
Product_Category_1            403182 non-null int64
Product_Category_2            403182 non-null float64
Product_Category_3            403182 non-null float64
Marital_Status                403182 non-null int64
Purchase                      403182 non-null int64
Avg_Purchase_Per_Product      403182 non-null float64
Avg_Purchase_Per_User         403182 non-null float64
dtypes: float64(4), int64(8), object(1)
memory usage: 41.5+ MB
```

- Next, we will visualize the data.

Average Purchase Amount

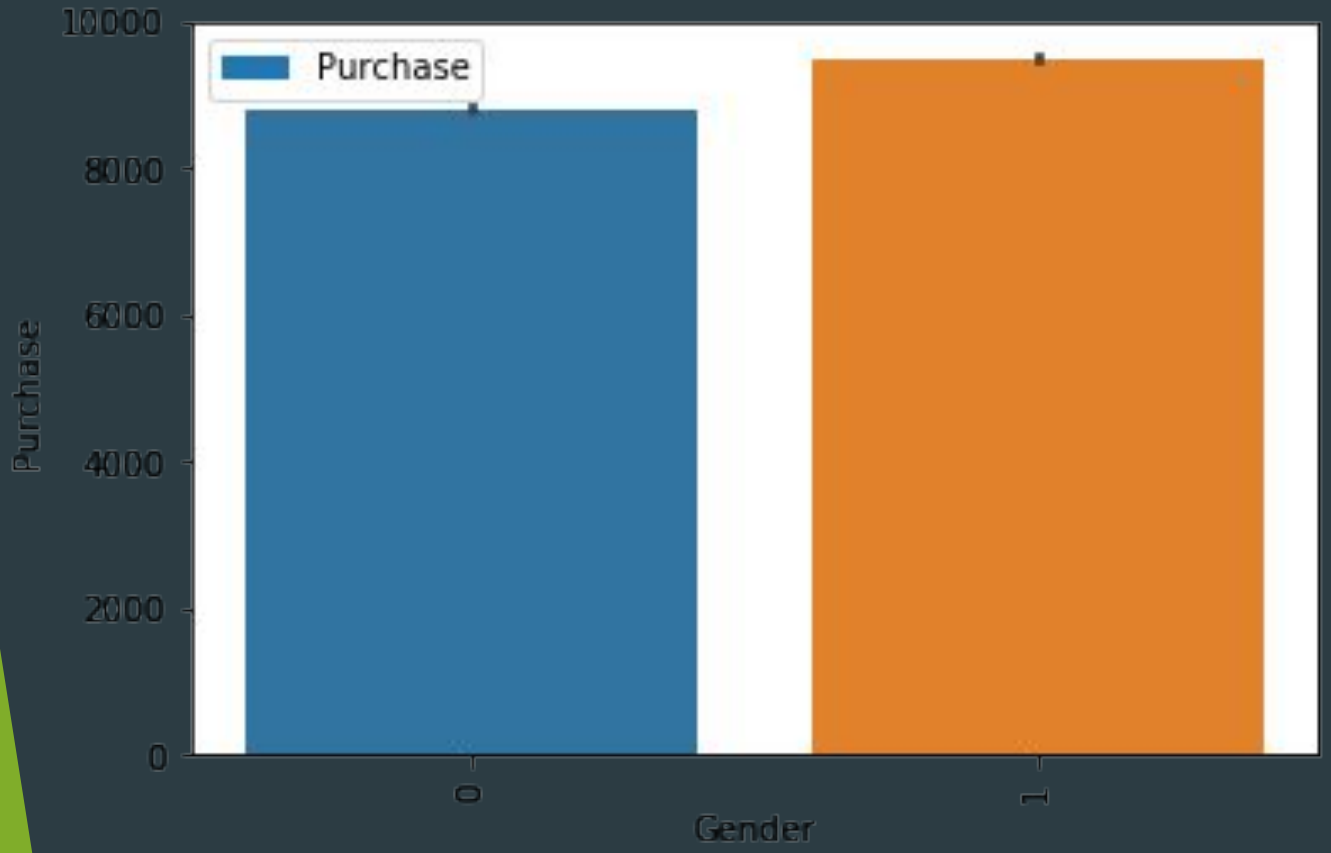
To the right, we are looking at the histogram for purchase amount. We can see that the data is almost normally distributed.

We can also see that the average purchase amount was between $5,000 and $10,000. Not many people made purchases that were more that $20,000.

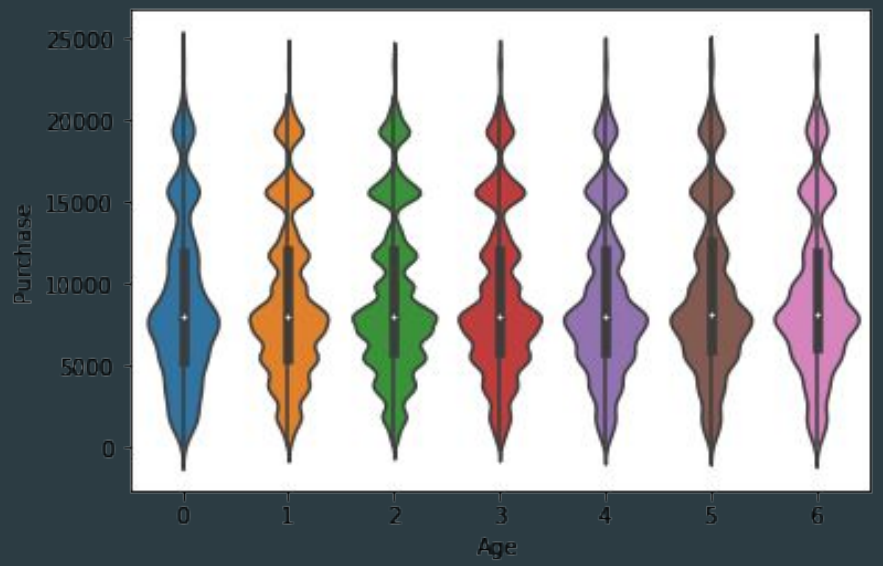


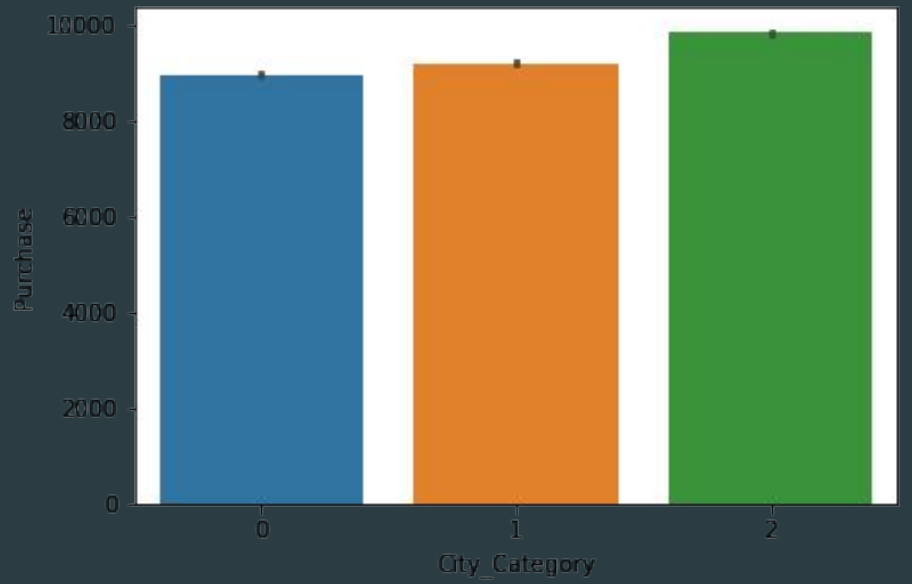
Males vs. Females

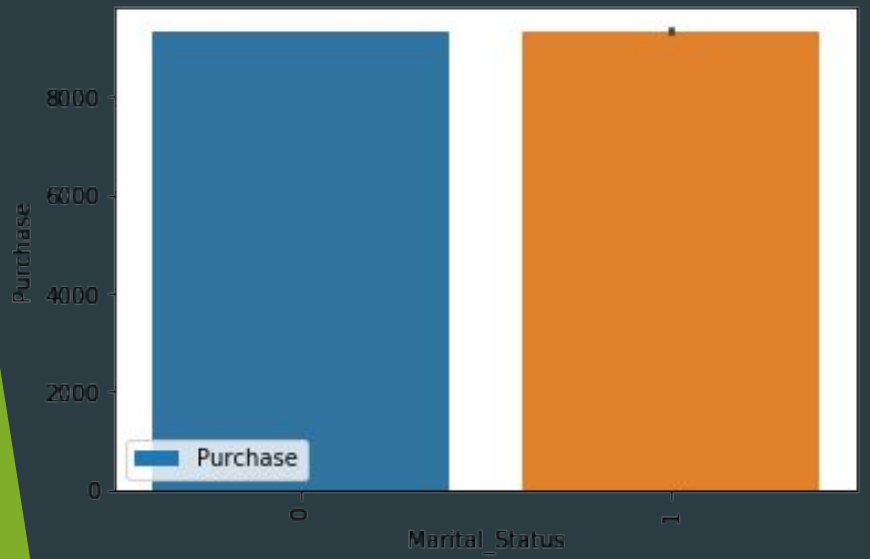
In the bar plot below, we can see that on average, males made greater purchases than females during Black Friday. This means that gender slightly impacts the total amount of purchase.



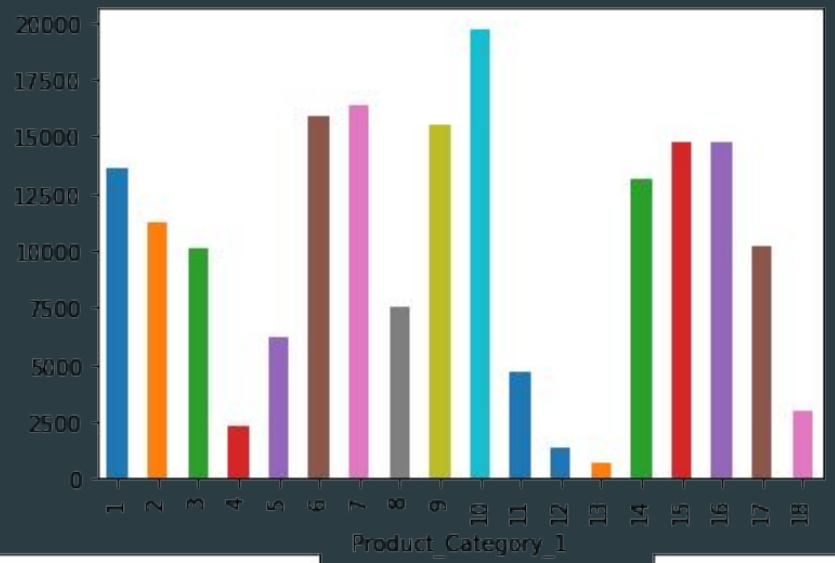
Age, City Category, Marital Status, & Occupation

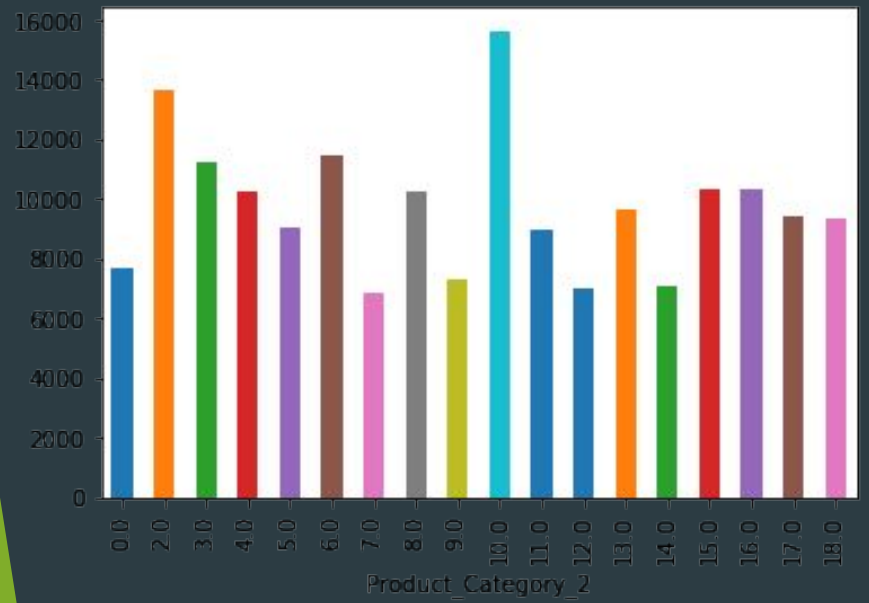


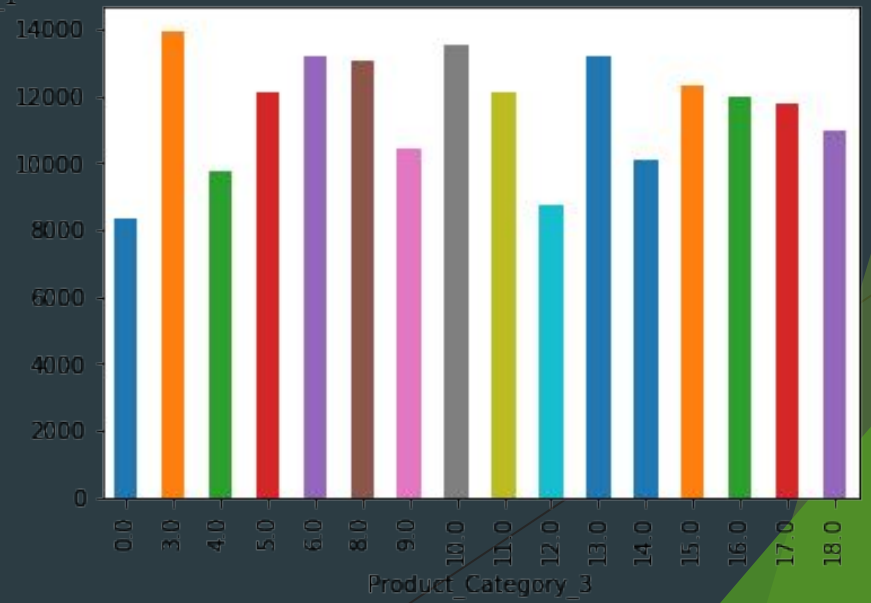




Purchase by Product Categories

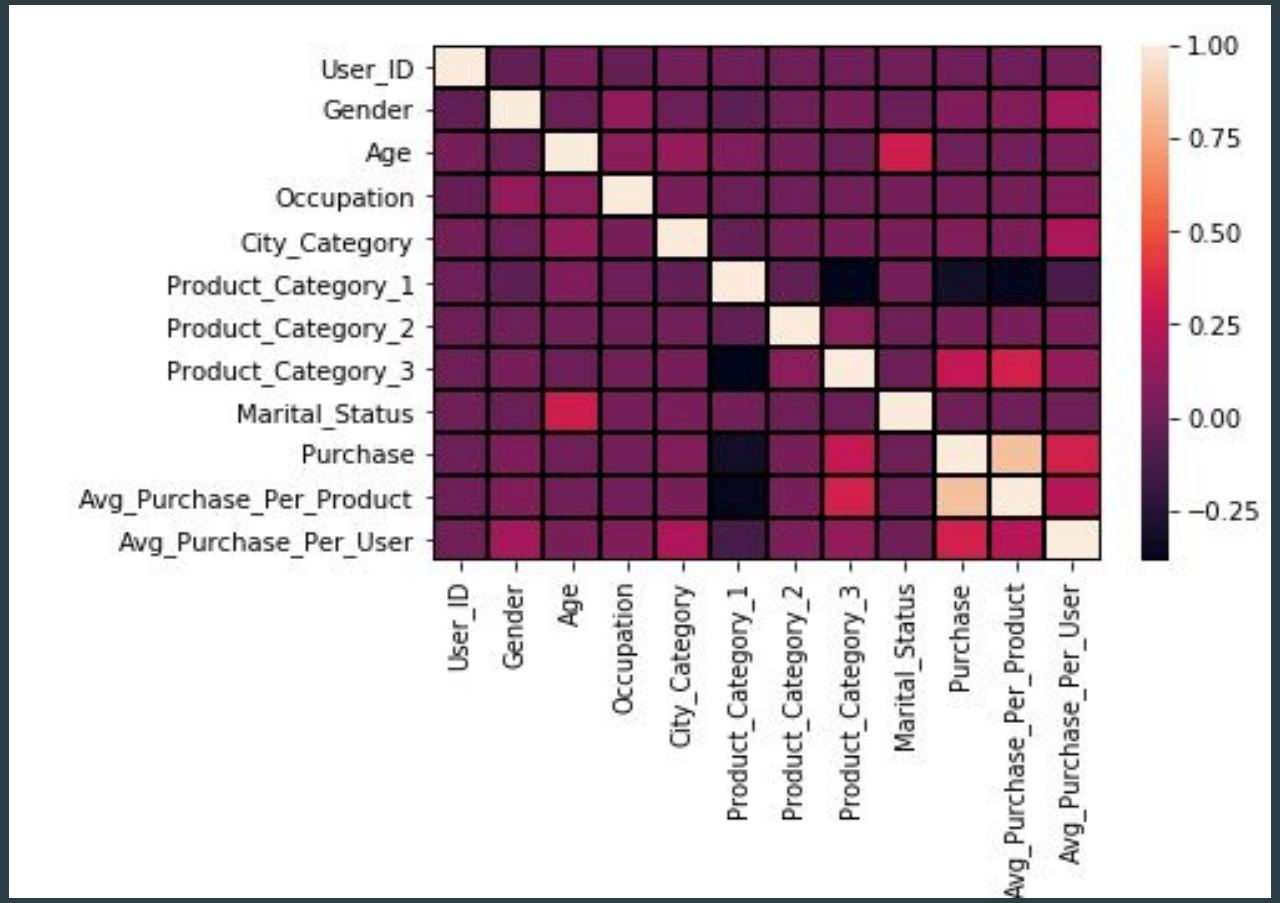






Correlation

As we can see, there is not a high correlation between any of the original features and purchase, but there is some correlation between purchase and the average purchase amounts per product and user.
Multicollinearity should not be an issue in predicting the purchase amount.



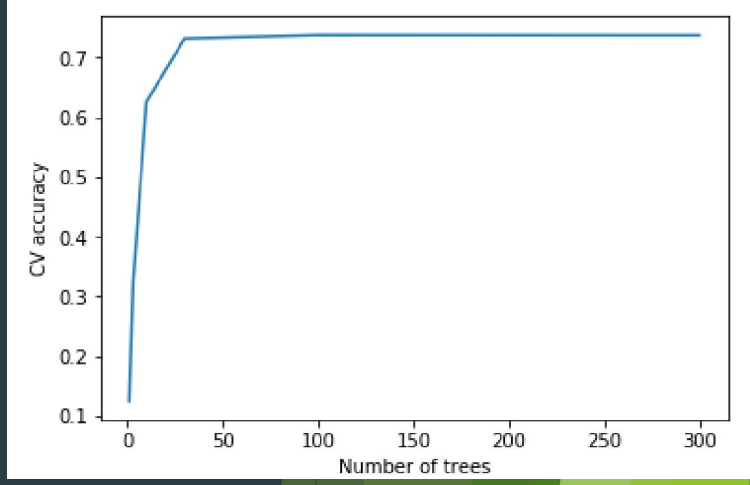
Prediction Model

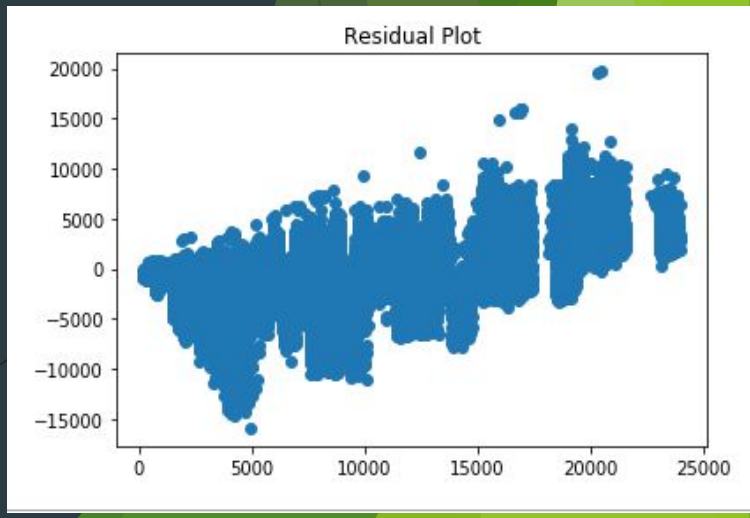
- I used different models to try to predict the purchase amount by customer. The best model was the gradient boosting model.

```
In [81]: # gradient boosting
         gb = GradientBoostingRegressor()
         gb.fit(X_train, y_train)
         # scores_gb = cross_val_score(gb, X_train,y_train, cv=5)
         # score_gb = cross_val_score(gb, X_train, y_train, cv=5).mean()
         # print('Cross-Validation Scores: \n', scores_gb)
         # print('Averaged Cross-Validation Scores: {:.2%}.\n'.format(score_gb))
         print ('Score on test', (gb.score(X_test,y_test)))

Score on test 0.7328444606232539
```

- In order to achieve this R-squared value of 73%, I included the average purchase per user and average purchase per product features which raised the accuracy score by more than 10%.
- I used a grid search for this model to determine that the best parameter was 100.
- We can also see in residual plot how the predicted values differ from the actual values.

```
Best parameter: {'n_estimators': 100}
Best score: 0.74
```





Other Models

I tried using other models such as linear regression, KNN, and random forest, but they were not as successful as the gradient boosting model.

In [75]:
```
## linear regression
lr = linear_model.LinearRegression()
lr.fit(X_train,y_train)
print ('Score on test', (lr.score(X_test,y_test)) )
```

```
Score on test 0.7261501053859667
```

In [76]:
```
## KNN
knn = KNeighborsRegressor()
knn.fit(X_train,y_train)
print ('Score on test', (knn.score(X_test,y_test)) )
```

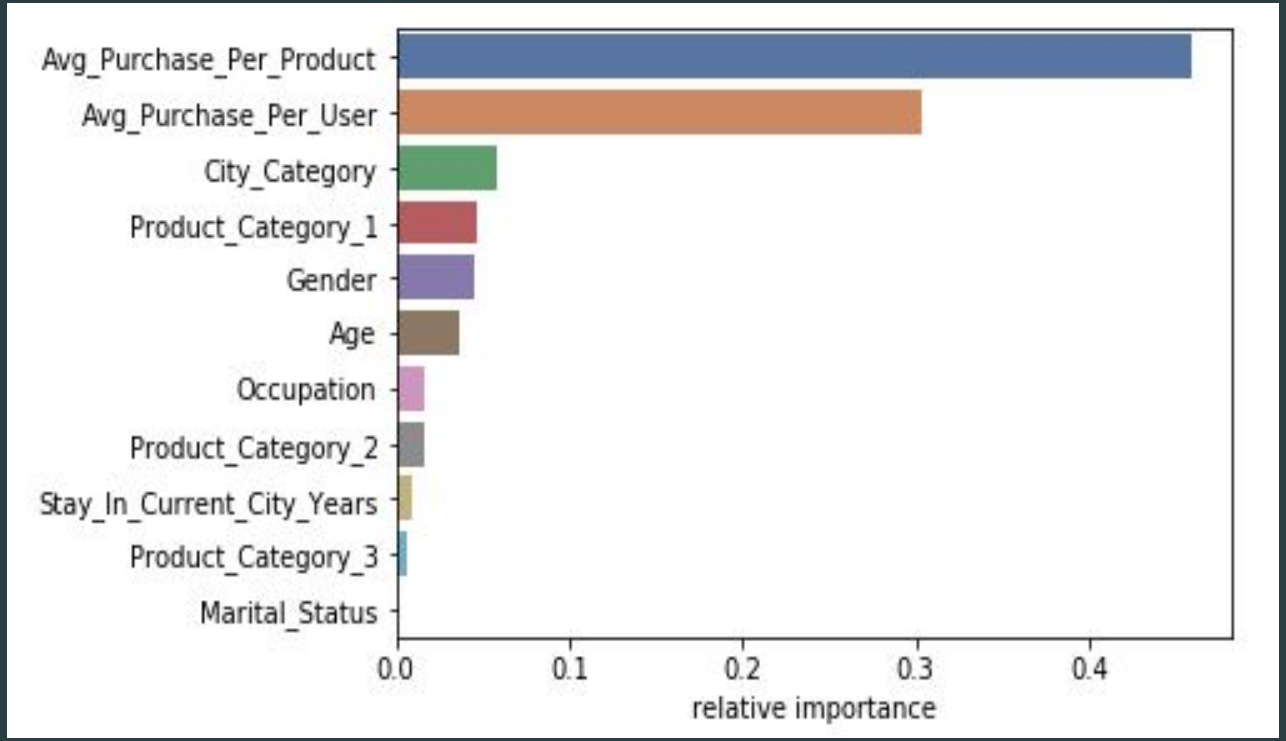
```
Score on test 0.6842528628771378
```

In [77]:
```
#random forest
rf = ensemble.RandomForestRegressor()
rf.fit(X_train,y_train)
print ('Score on test', (rf.score(X_test,y_test)))
#cross_val_score(rf, X_train, y_train, cv=5)
```

```
Score on test 0.7062841344820567
```

Most Important Features

The bar plot below confirms that the two most important features in our prediction model were average purchase per product and average purchase per user. Some features such as city category, product category 1, and gender had some importance, but not as much as the top two.




```
Winning Model is Gradient Boosting
==================================
Feature rank among 7 features:
Avg_Purchase_Per_Product      0.460
Avg_Purchase_Per_User         0.304
City_Category                 0.058
Product_Category_1            0.047
Gender                        0.045
Age                           0.036
Occupation                    0.017
Product_Category_2            0.016
Stay_In_Current_City_Years    0.009
Product_Category_3            0.006
Marital_Status                0.000
dtype: float64
```

Conclusion

- After trying different models for predicting the amount of a Black Friday purchase amount, it seems that the gradient boosting model is the best one with an accuracy score of 73%.
- I tried other models such as linear regression, lasso regression, KNN, and random forest but they were not as effective as the gradient boosting model.
- It is possible to improve our score perhaps by making new features or grouping purchase amount by product category.
- This model could be improved upon and prove useful to retailers in order to determine who are their target customers to increase Black Friday sales.

Next Steps

- Some next steps for this data would be to collect data in order to acquire details for products, categories, cities, and the year in order better analyze the data.
- I would like to create a better model for prediction with a better accuracy score and new features.
- I would also like to conduct further research to determine the national average purchase amount during Black Friday to compare to this data.