Project Report

Black Friday

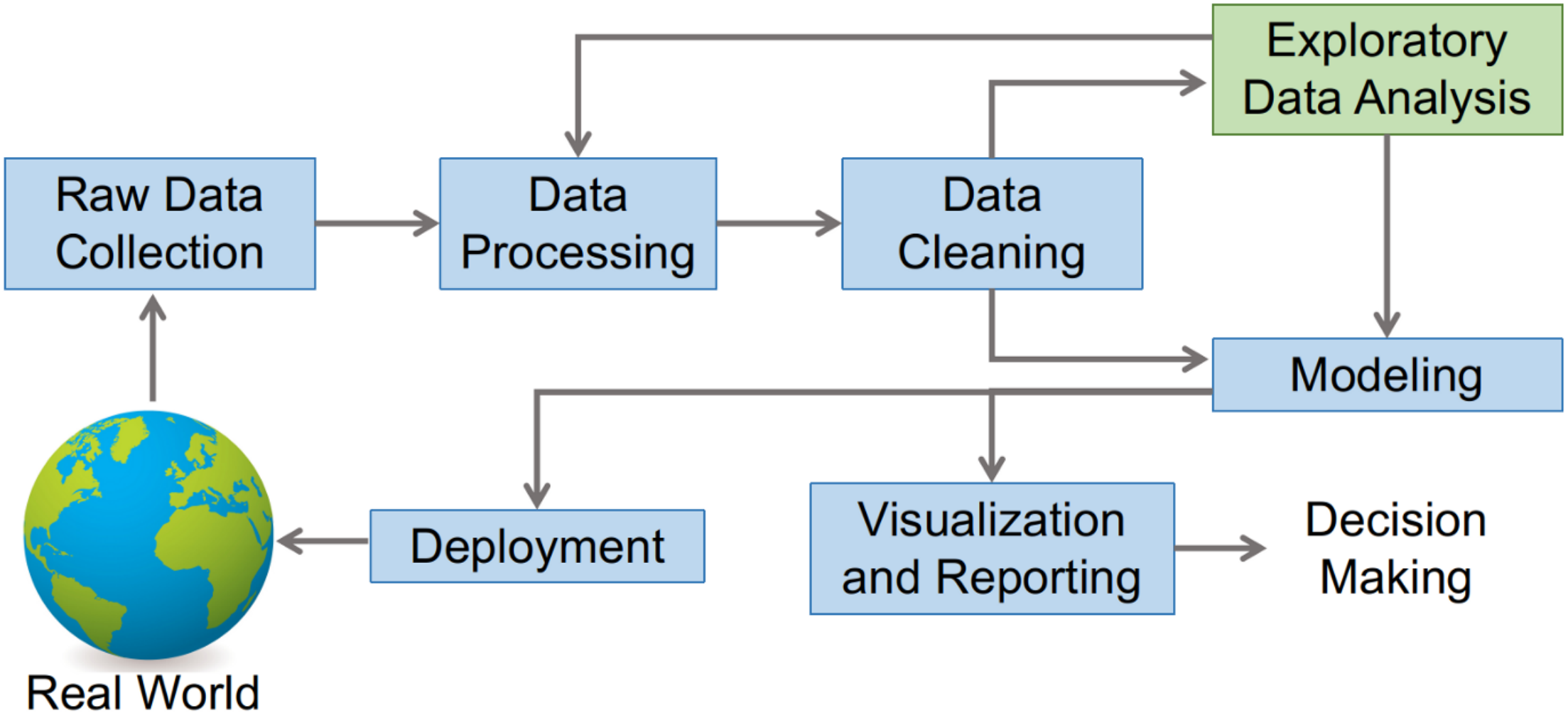

BLACK
FRIDAY
Sale

BackGround

Black Friday is a colloquial term for the Friday after Thanksgiving in the United States. It traditionally marks the start of the Christmas shopping season in the United States. Many stores offer highly promoted sales at discounted prices and often open early, sometimes as early as midnight[2] or even on Thanksgiving. Some stores' sales continue to Monday ("Cyber Monday") or for a week ("Cyber Week").

Black Friday occurs on the day after Thanksgiving in the United States. Black Friday has routinely been the busiest shopping day of the year in the United States.

Data Science Process



Exploratory Data Analysis

Training Dataset

- Rows and Columns: 550068, 12
- Features: 'User_ID', 'Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category','Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category_1', 'Product_Category_2', 'Product_Category_3'
- Target: 'Purchase'
- Null values: Product_Category_3 383247
- Product_Category_2 1 73638

User_ID	int64
Product_ID	object
Gender	object
Age	object
Occupation	int64
City_Category	object
Stay_In_Current_City_Years	object
Marital_Status	int64
Product_Category_1	int64
Product_Category_2	float64
Product_Category_3	float64
Purchase	int64

DataTypes

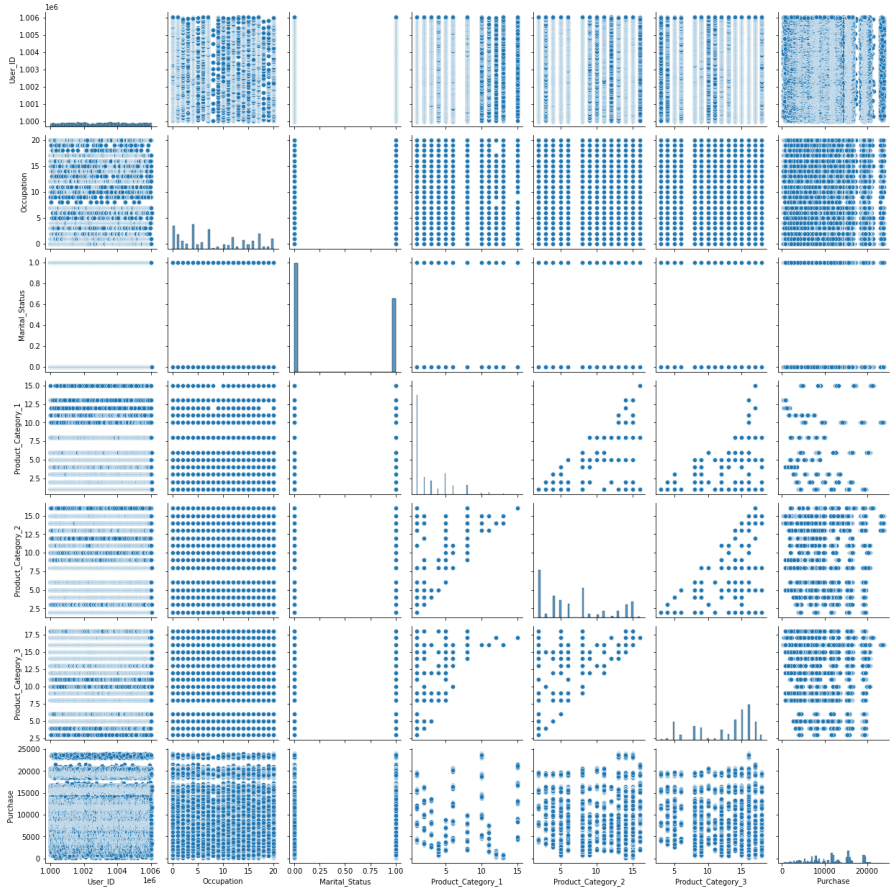
	count	mean	std	min	25%	50%	75%	max	range
User_ID	550068.0	1.003029e+06	1727.591586	1000001.0	1001516.0	1003077.0	1004478.0	1006040.0	6039.0
Occupation	550068.0	8.076707e+00	6.522660	0.0	2.0	7.0	14.0	20.0	20.0
Marital_Status	550068.0	4.096530e-01	0.491770	0.0	0.0	0.0	1.0	1.0	1.0
Product_Category_1	550068.0	5.404270e+00	3.936211	1.0	1.0	5.0	8.0	20.0	19.0
Product_Category_2	376430.0	9.842329e+00	5.086590	2.0	5.0	9.0	15.0	18.0	16.0
Product_Category_3	166821.0	1.266824e+01	4.125338	3.0	9.0	14.0	16.0	18.0	15.0
Purchase	550068.0	9.263969e+03	5023.065394	12.0	5823.0	8047.0	12054.0	23961.0	23949.0

```
df.describe(include='object').T
```

	count	unique	top	freq
Product_ID	550068	3631	P00265242	1880
Gender	550068	2	M	414259
Age	550068	7	26-35	219587
City_Category	550068	3	B	231173
Stay_In_Current_City_Years	550068	5	1	193821

Descriptive statistics

Data Visualization and relationships

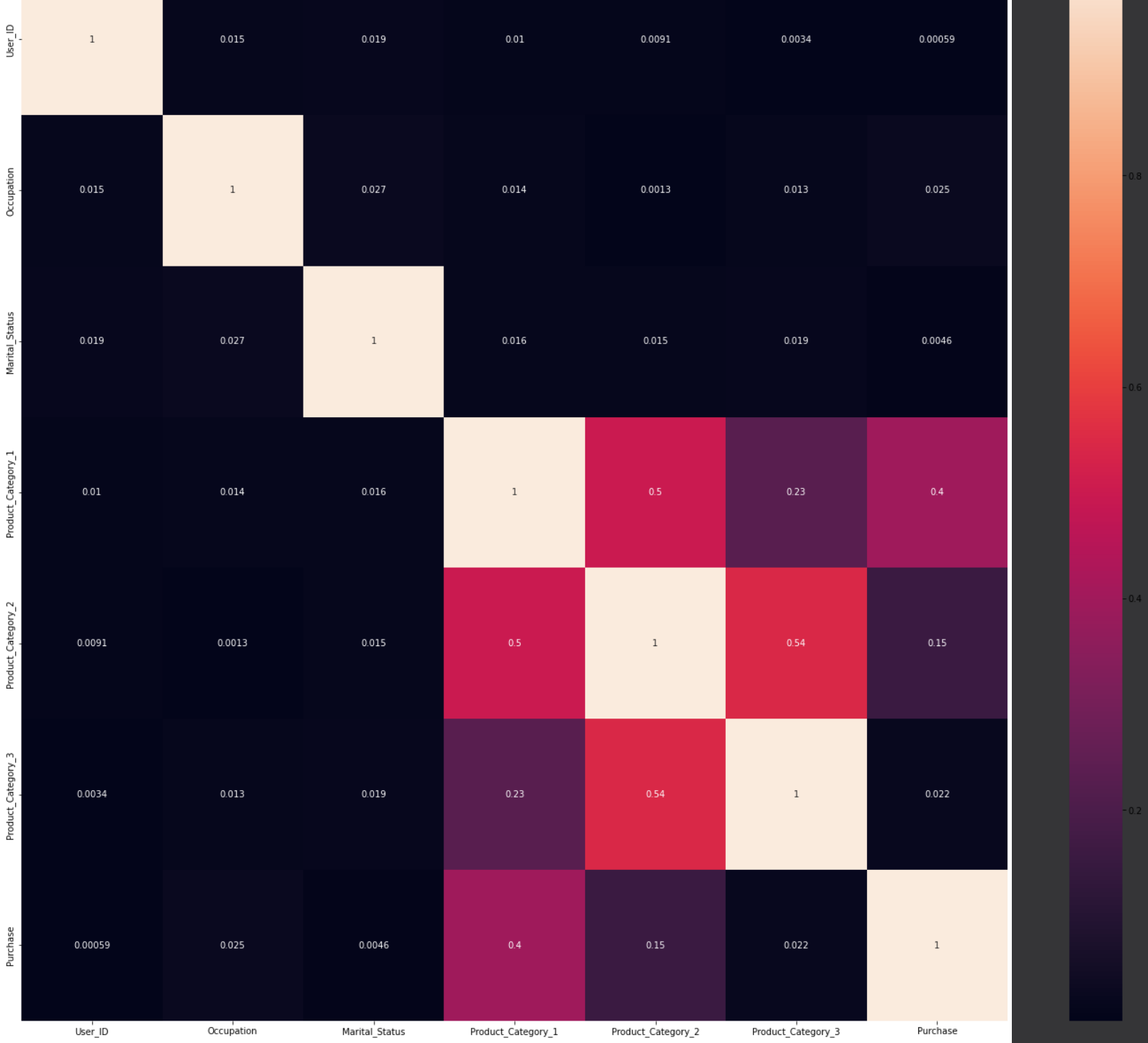


Pairplots

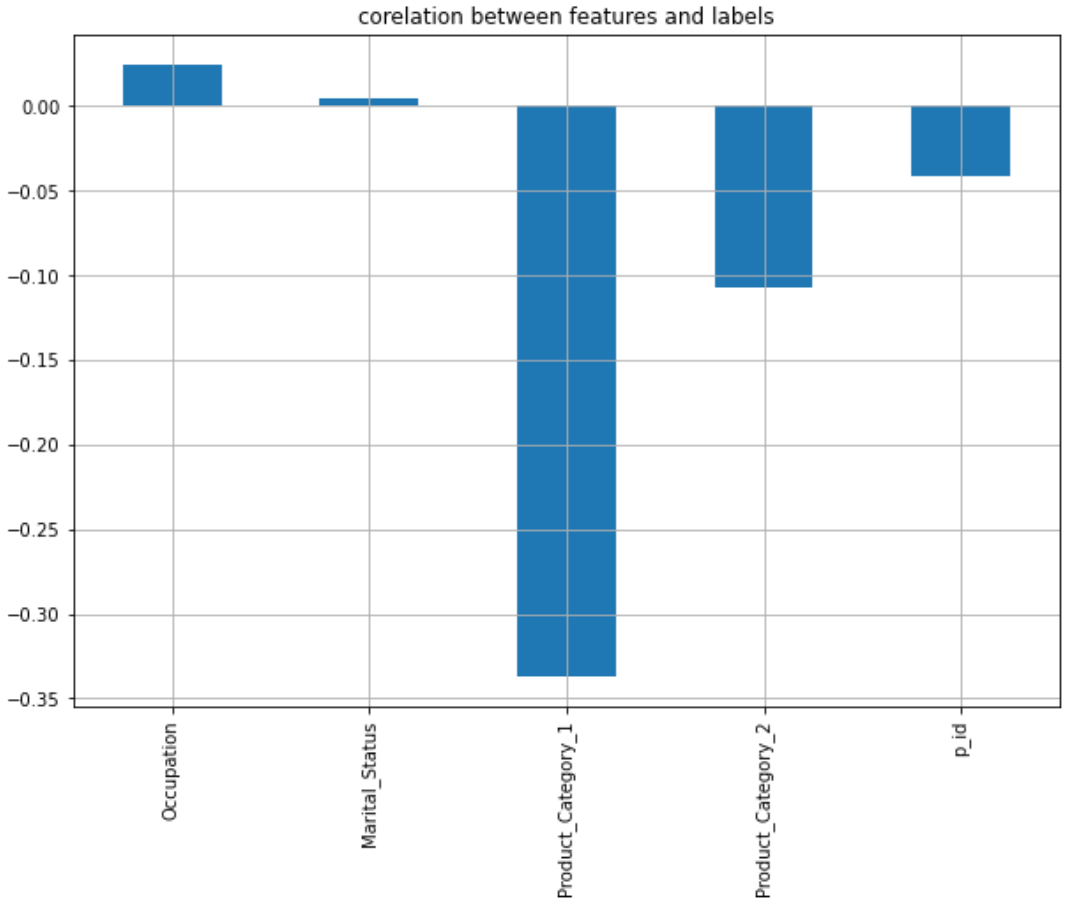
Pair plot shows No sign of strong relation between features. As the data is pattered roughly.

Correlation and Multicollinearity

correlation and heat map show product category features have a problem of multicollinearity and that needs to be treated.



Correlation between features and label



Variance inflation factor

```
]: from statsmodels.stats.outliers_influence import variance_inflation_factor


vif = pd.DataFrame()
vif['vif'] = [variance_inflation_factor(df1[cont_columns],i) for i in range(df1[cont_columns].shape[1])]
vif['features'] = df1[cont_columns].columns
vif.sort_values(by='vif',ascending=False)
```

]:

	vif	features
0	22.823305	User_ID
5	14.918952	Product_Category_3
6	7.517701	Purchase
4	6.022380	Product_Category_2
7	3.581712	p_id
3	3.311656	Product_Category_1
1	2.593746	Occupation
2	1.677069	Marital_Status

The columns user ID product category 3 purchase and product category two columns have multicollinearity issue.

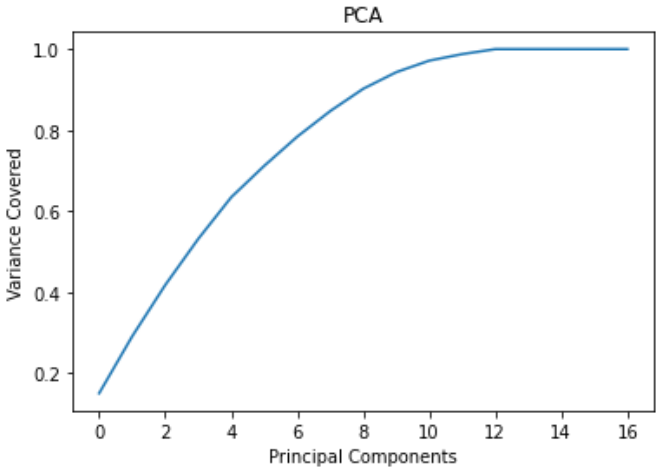
Chi square test

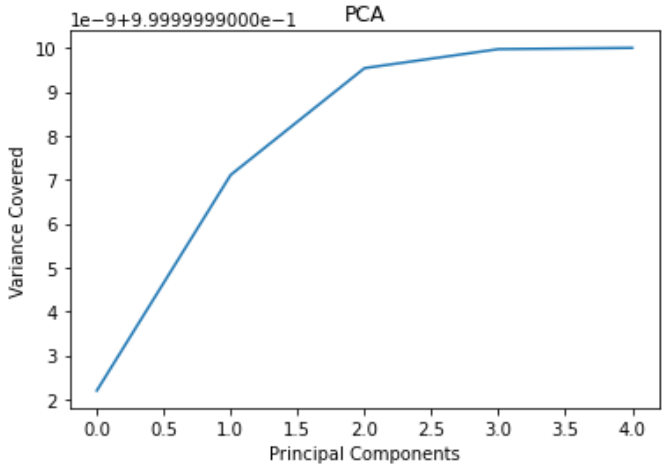
```
data_scores = pd.DataFrame(fit.scores_)
data_columns = pd.DataFrame(X[cont_columns].columns)

features_score = pd.concat([data_columns,data_scores],axis=1)

features_score.columns = ['Features','Scores']
print(features_score.nlargest(5,'Scores'),'\n') # print 5 best features

# Here we are getting top 5 features we got based on f_classify that uses ANOVA test of statistics.
```

	Features	Scores
2	Product_Category_1	27.240240
3	Product_Category_2	3.041683
4	p_id	1.366393
0	Occupation	1.027642
1	Marital_Status	0.983043

Since it is better to understand the feature importance in the analysis process, we have chosen the Chi Square test and found that the product category one feature is the most important in predicting the purchase amount of customers during Black Friday sale.

Feature Selection





As our data it's a mix of categorical and continuous features so before proceeding with the model development process we are going to first convert our categorical features into continuous once using encoding techniques and further we will do the principal component analysis so that we can understand the components that are most responsible and important in covering the variance present in the data set and then we will select them and drop any unnecessary components and this will be done using principal component analysis and by plotting the scree plot.

So here we have plotted scree plot for both categorical features and continuous features separately and selected the best components for further analysis

Model building

- Feature and Target selection
- Data Transformation
- Model selection and development
- Model Evaluation
- Model Improvement and Hyperparameter tuning

Algorithms used

- Linear regression
- Ridge and lasso regression
- decision tree regression
- random forest regressor
- xg boost regressor
- ada boost regressor
- support vector regressor
- K neighbours regressor

Performance (mean squared error)

- Linear Regression 4617.99
- Ridge Regression 4687.75
- Lasso Regression 4694.14
- Decision Tree Regressor 3363.87
- Random Forest Regressor 3062.72
- XG Boost 2713
- KNN Model 2703

Best model


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

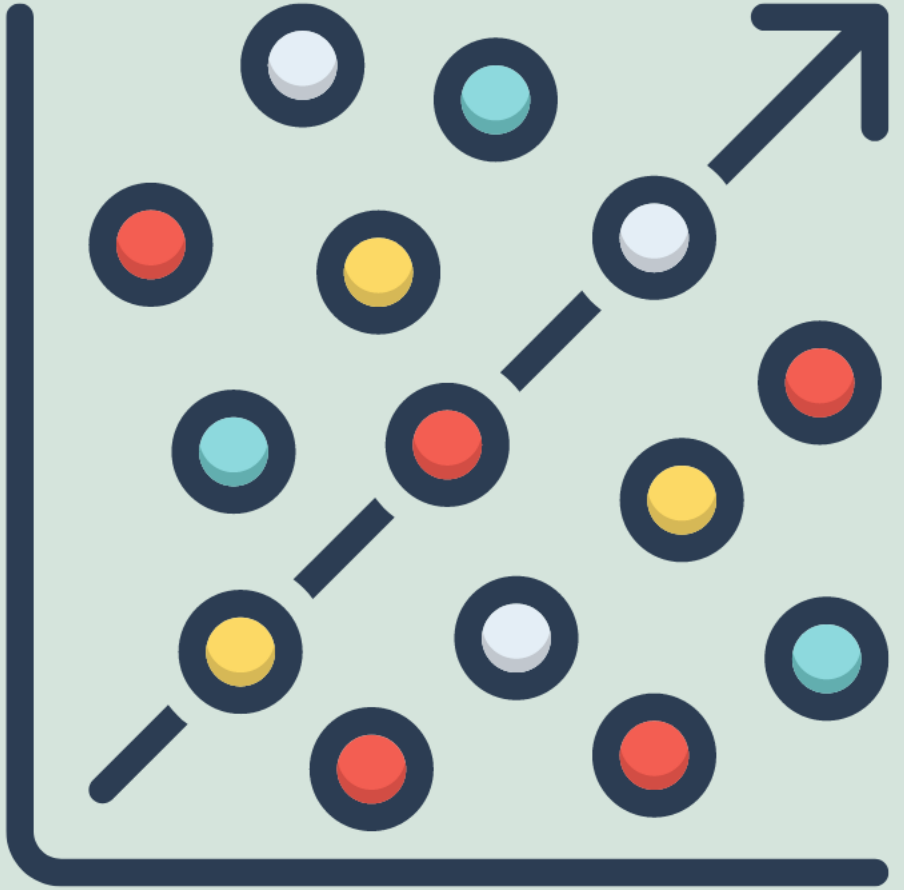
Detailed result (Training and Model score)

- # Using XGBoost:

- xgb_reg = xgb.XGBRegressor(n_estimators=200,learning_rate=0.1)
- xgb_reg.fit(x_train,y_train)
- print('Training score: ',xgb_reg.score(x_train, y_train))
- y_predx = xgb_reg.predict(x_test)
- print('R2 score: ',r2_score(y_test,y_predx))

- `Training score:  0.6028498580163906`
- `R2 score:  0.5711187448028863`

Regression Metrics in Machine Learning



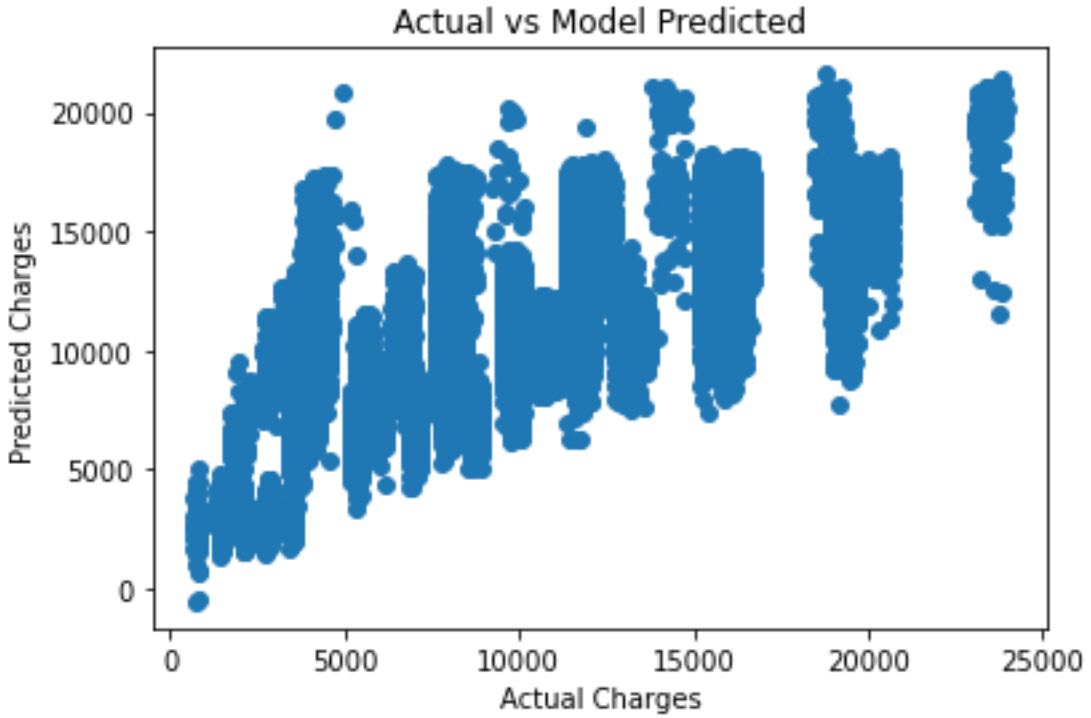
metrics

Model Evaluation: MAE , MSE , RMSE

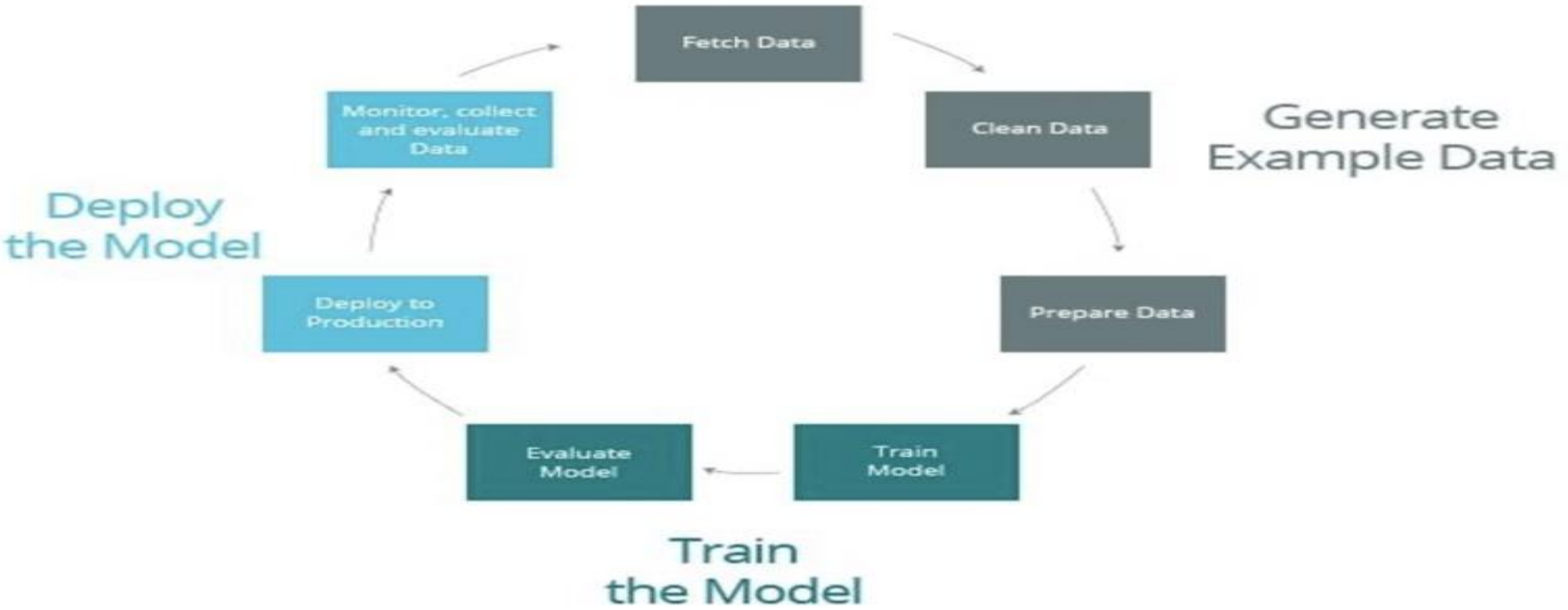
- from sklearn.metrics import mean_absolute_error,mean_squared_error
- print("MAE: ", metrics.mean_absolute_error(y_test, y_predx))
- print("MSE: ", metrics.mean_squared_error(y_test, y_predx))
- print("RMSE: ", metrics.mean_squared_error(y_test, y_predx, squared=False))
- print("R2: ", metrics.r2_score(y_test, y_predx), "\n")
- print("Score: ", xgb_reg.score(x_test, y_predx))

- **MAE: 2538.0027214224583**
- **MSE: 10619406.010134248**
- **RMSE: 3258.7430107534174**
- **R2: 0.5711187448028863**
- **Score: 1.0**

Actual vs predicted values



Machine Learning Pipeline



Pipelining

```
from sklearn.pipeline import Pipeline

pipe1 = Pipeline([('pca',PCA(n_components=17)),('base_model1',xgb.XGBRegressor(n_estimators=200,learning_rate=0.1))])

pipe1.fit(x_train,y_train)

y_predx = xgb_reg.predict(x_train)
y_pred = pipe1.predict(x_test)

print('Training accuracy: ', r2_score(y_train,y_predx)*100)
print('Testing accuracy: ', r2_score(y_test,y_pred)*100)
print("MAE: ", metrics.mean_absolute_error(y_test, y_pred))
print("MSE: ", metrics.mean_squared_error(y_test, y_pred))
print("RMSE: ", metrics.mean_squared_error(y_test, y_pred, squared=False))
print("R2: ", metrics.r2_score(y_test, y_pred), "\n")
print("Score: ", pipe1.score(x_test, y_pred))

# Saving regression model to pickle string

import pickle
saved_model1 = pickle.dumps(pipe1)
pipe_pickle1 = pickle.loads(saved_model1)
pipe_pickle1.predict(x_test) # predicting testing data
```

```
Training accuracy:  60.28498580163906
Testing accuracy:  56.62258985314824
MAE:  2559.1852021096165
MSE:  10740556.376282398
RMSE:  3277.2788066141698
R2:  0.5662258985314824

Score:  1.0
```

Pipelining can be used to automate whole process of Data preprocessing and model building.

deployment

```
: df_test
```

:

	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Product_Category_2	p_id	Purchase
4	F	26-35	1	C	1	0	4	5.0	53842	2875.656494
5	M	46-50	1	C	3	1	2	3.0	350442	11581.344727
6	M	46-50	1	C	3	1	1	11.0	155442	11933.896484
7	M	46-50	1	C	3	1	2	4.0	94542	11371.570312
8	M	26-35	7	A	1	0	10	13.0	161842	13001.013672
...	...	...	...	...	...	...	...	...	...	...
233584	M	26-35	0	C	2	1	1	11.0	262242	12299.335938
233586	M	36-45	6	C	1	1	1	2.0	110742	16578.753906
233588	M	26-35	17	C	1	1	6	8.0	129842	14053.278320
233591	M	51-55	13	B	1	1	1	2.0	127642	13567.117188
233596	F	26-35	15	B	4+	1	1	5.0	31842	11816.208008

69677 rows × 10 columns

conclusion

- With traditional methods not being of much help to business growth in terms of revenue, the use of Machine learning approaches proves to be an important point for the shaping of the business plan taking into consideration the shopping pattern of consumers.
- The evaluation measure used is Mean Squared Error (RMSE).
- XG Boost Regressor is best suitable for the prediction of sales based on a given dataset. Thus, the proposed model will predict the customer purchase on Black Friday and give the retailer insight into customer choice of products. This will result in a discount based on customer-centric choices thus increasing the profit to the retailer as well as the customer