Lead Scoring Case Study

Presented by: - Anhad Saini

Introduction

An education company named X Education sells online courses to industry professionals. On any given day, many professionals who are interested in the courses land on their website and browse for courses

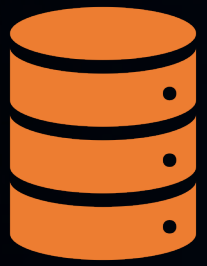
The company markets its courses on several websites and search engines like Google. Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos. When these people fill up a form providing their email address or phone number, they are classified to be a lead. Moreover, the company also gets leads through past referrals. Once these leads are acquired, employees from the sales team start making calls, writing emails, etc. Through this process, some of the leads get converted while most do not. The typical lead conversion rate at X education is around 30%.

Problem Statement

X Education gets a lot of leads, its lead conversion rate is very poor. For example, if, say, they acquire 100 leads in a day, only about 30 of them are converted. To make this process more efficient, the company wishes to identify the most potential leads, also known as 'Hot Leads'. If they successfully identify this set of leads, the lead conversion rate should go up as the sales team will now be focusing more on communicating with the potential leads rather than making calls to everyone.

As we can see, there are a lot of leads generated in the initial stage (top) but only a few of them come out as paying customers from the bottom. In the middle stage, we need to nurture the potential leads well (i.e., educating the leads about the product, constantly communicating etc.) in order to get a higher lead conversion.

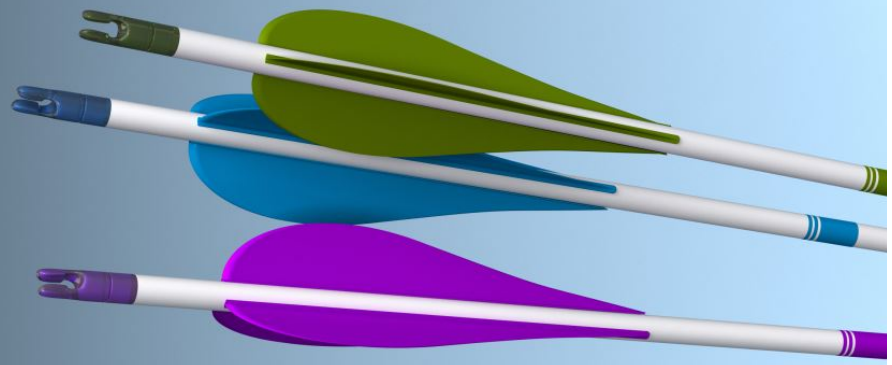
Data Structure



This dataset consists of various attributes : -

- Lead Source
- Total Visits
- Total Time Spent on Website
- Last Activity, etc.

These attributes may or may not be useful in ultimately deciding whether a lead will be converted or not. The target variable, in this case, is the column 'Converted' which tells whether a past lead was converted or not wherein 1 means it was converted and 0 means it wasn't converted. You can learn more about the dataset from the data dictionary provided in the zip folder at the end of the page. Another thing that you also need to check out for are the levels present in the categorical variables.



Targets and Requirements

The company requires a model wherein we need to assign a lead score to each of the leads such that the customers with higher lead score have a higher conversion chance and the customers with lower lead score have a lower conversion chance. The CEO has given a ballpark of the target lead conversion rate to be around 80%.

Let's Deep dive and fetch solutions and inferences

Research Methodology

We have taken the secondary bank loan data from UpGrad Portal.

Initially we'll clean the data and remove the null values as well as convert values according to our research so that we can fetch inference from it.

Research Methodology

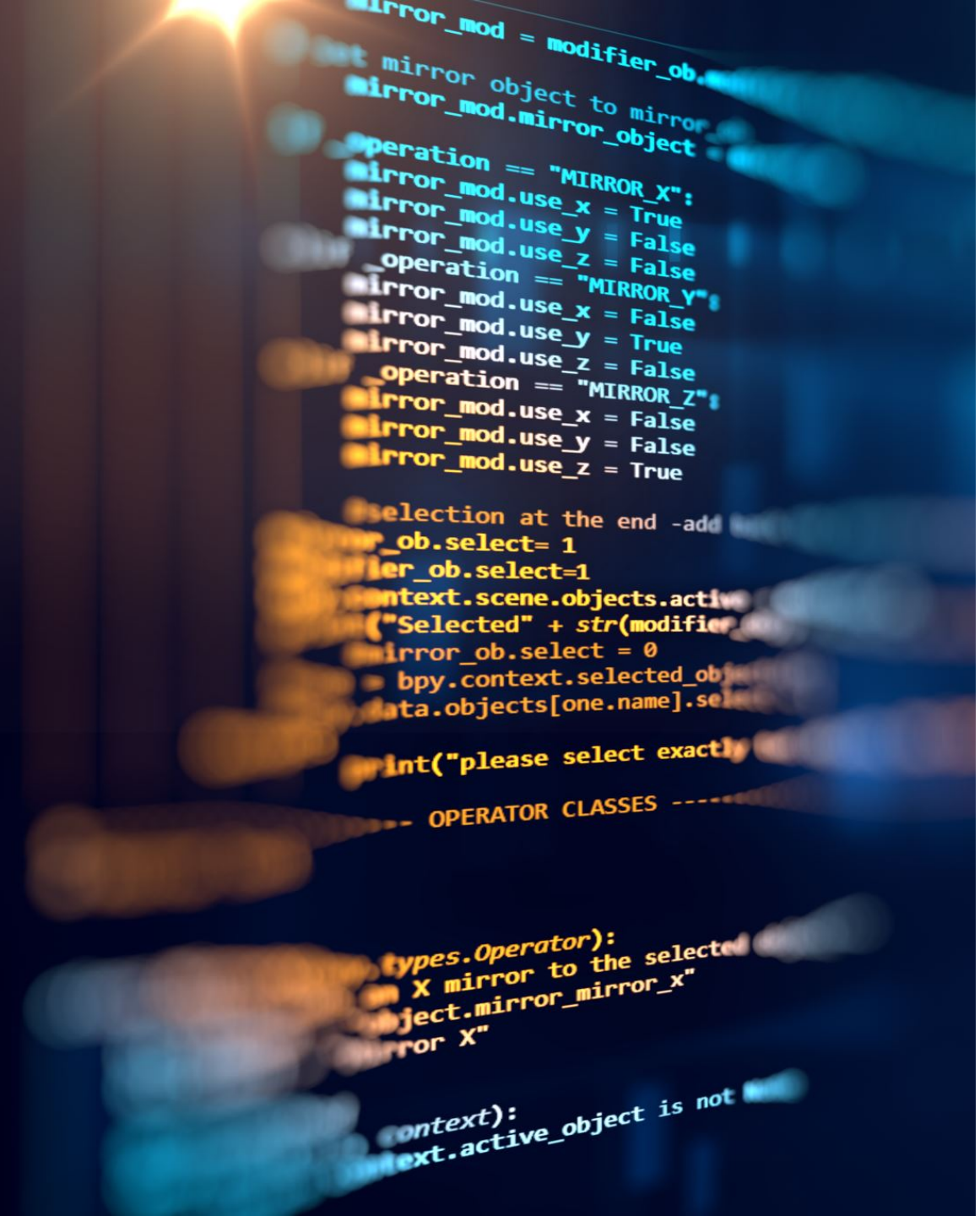
- Then we'll check for abnormalities/inconsistencies and Then we'll drop null values.
- Categories the quality of leads and change the note applicable values (N/A) in not sure.
- Prepare all the attribute like cites and promotional platforms in favor to research.
- Change the data type (Qualitative and quantitative) according to the need.

Language and Libraries used

Language — Python

Libraries — Numpy, Pandas

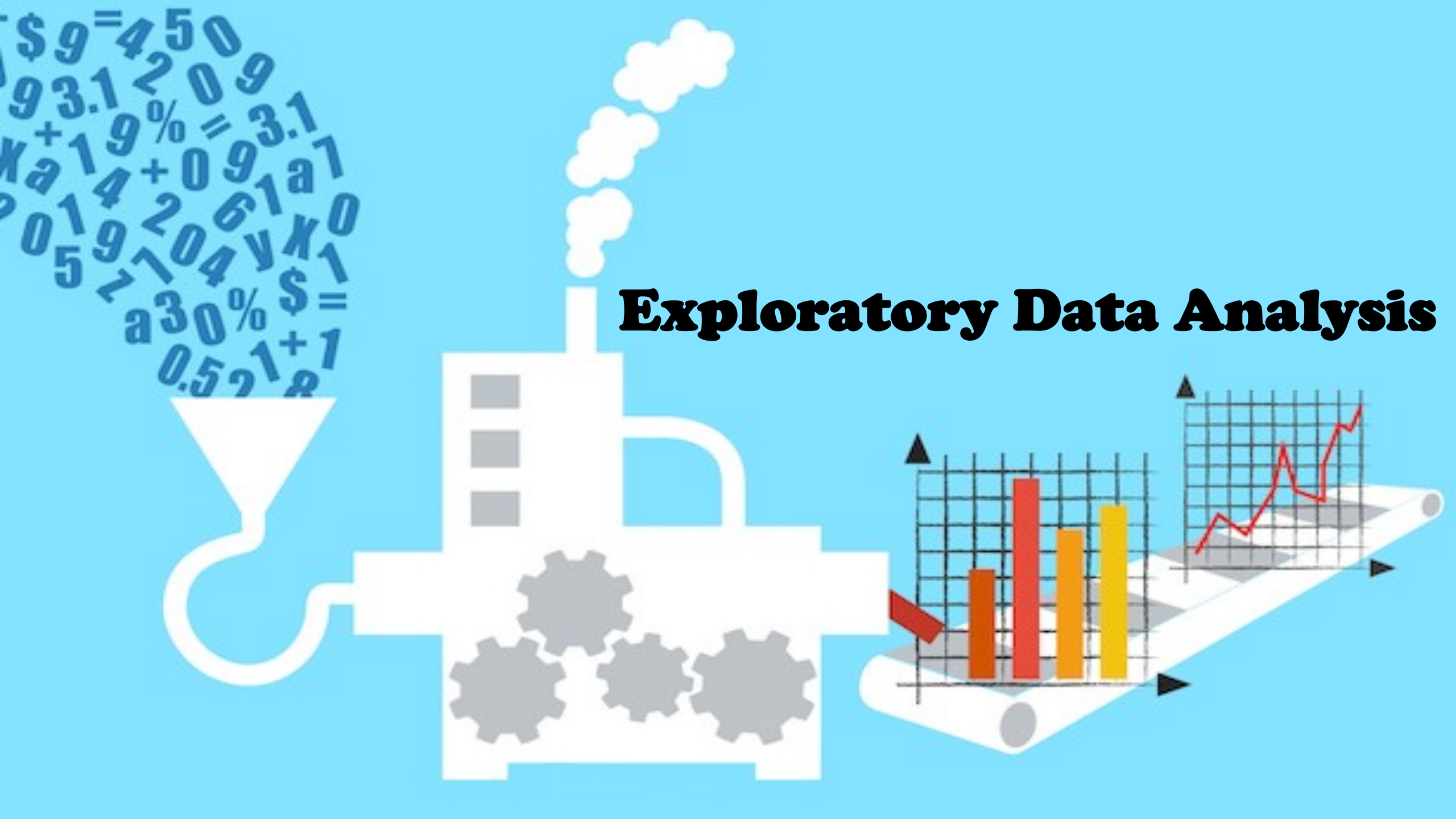
Data visualization tools — Matplotlib, Seaborn

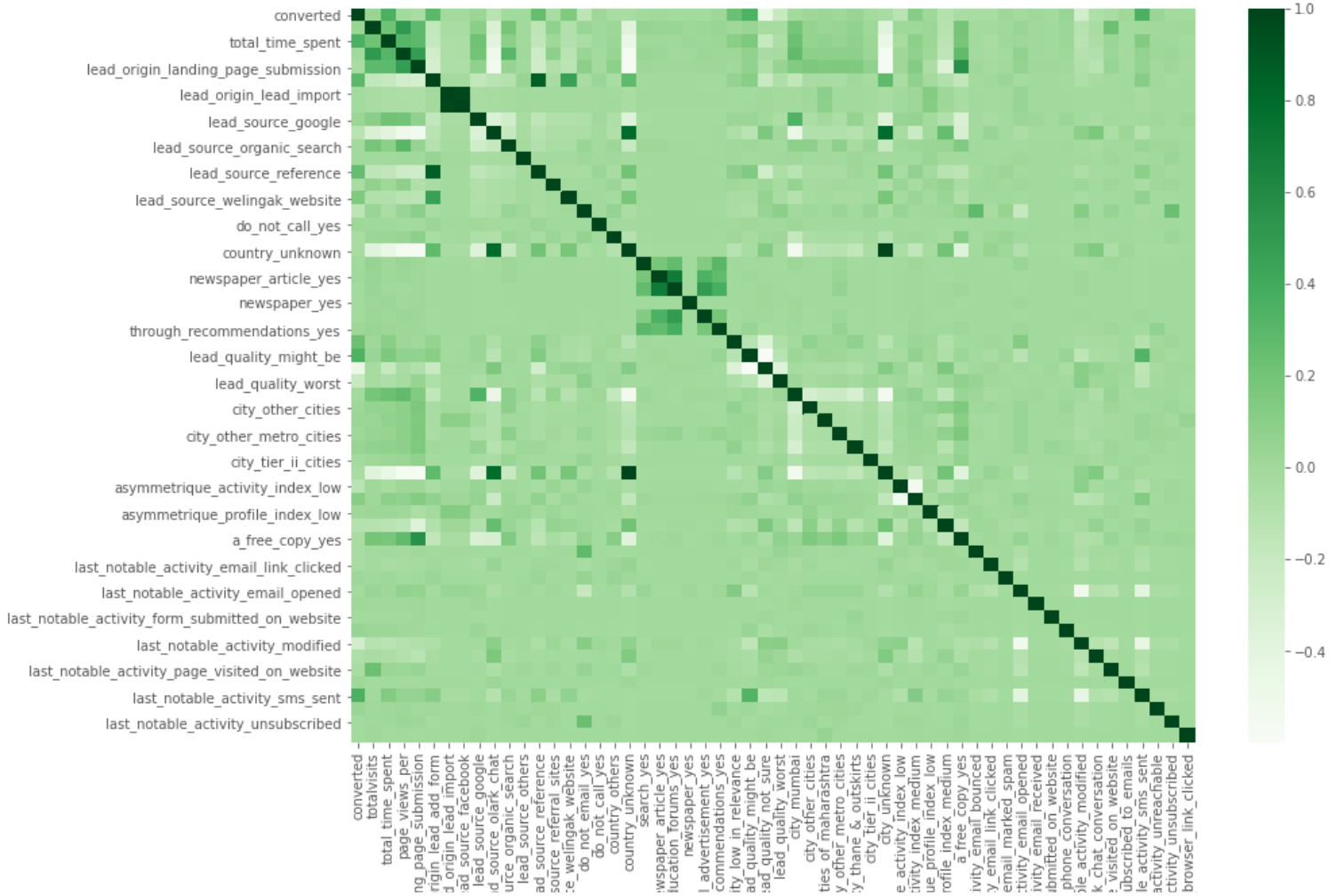


Research

- After importing all the libraries, we have called the data in our EDI with the help of python computer language.
- We have tested few basic operations like calling the headers, checking shape of the data, so that we can make sure that the database is working properly.

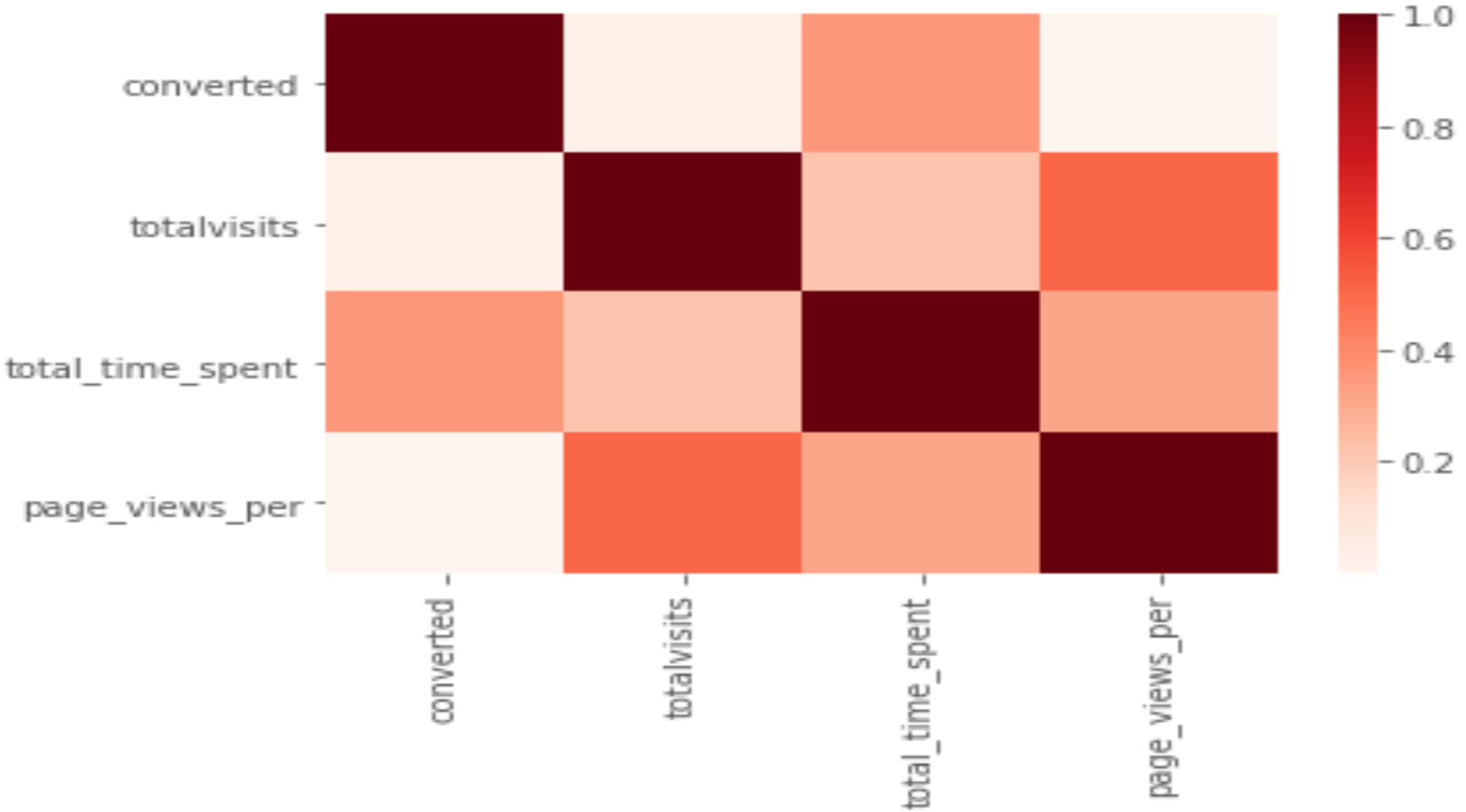
Exploratory Data Analysis





Checking the correlation with all the variables

Visualizing the continuous variables



columns pairs in order of highest absolute correlation

last_notable_activity_view_in_browser_link_clicked	last_notable_activity_view_in_browser_link_clicked	1.000000
lead_source_facebook	lead_origin_lead_import	0.983684
lead_origin_lead_add_form	lead_source_reference	0.866191
country_unknown	lead_source_olark_chat	0.803772
x_education_forums_yes	newspaper_article_yes	0.707068
lead_quality_not_sure	lead_quality_might_be	0.597667
city_unknown	lead_origin_landing_page_submission	0.566471
lead_origin_landing_page_submission	a_free_copy_yes	0.564863
page_views_per	country_unknown	0.556781
country_unknown	city_mumbai	0.536886
lead_source_olark_chat	lead_origin_landing_page_submission	0.528424
asymmetrique_activity_index_low	asymmetrique_activity_index_medium	0.526720

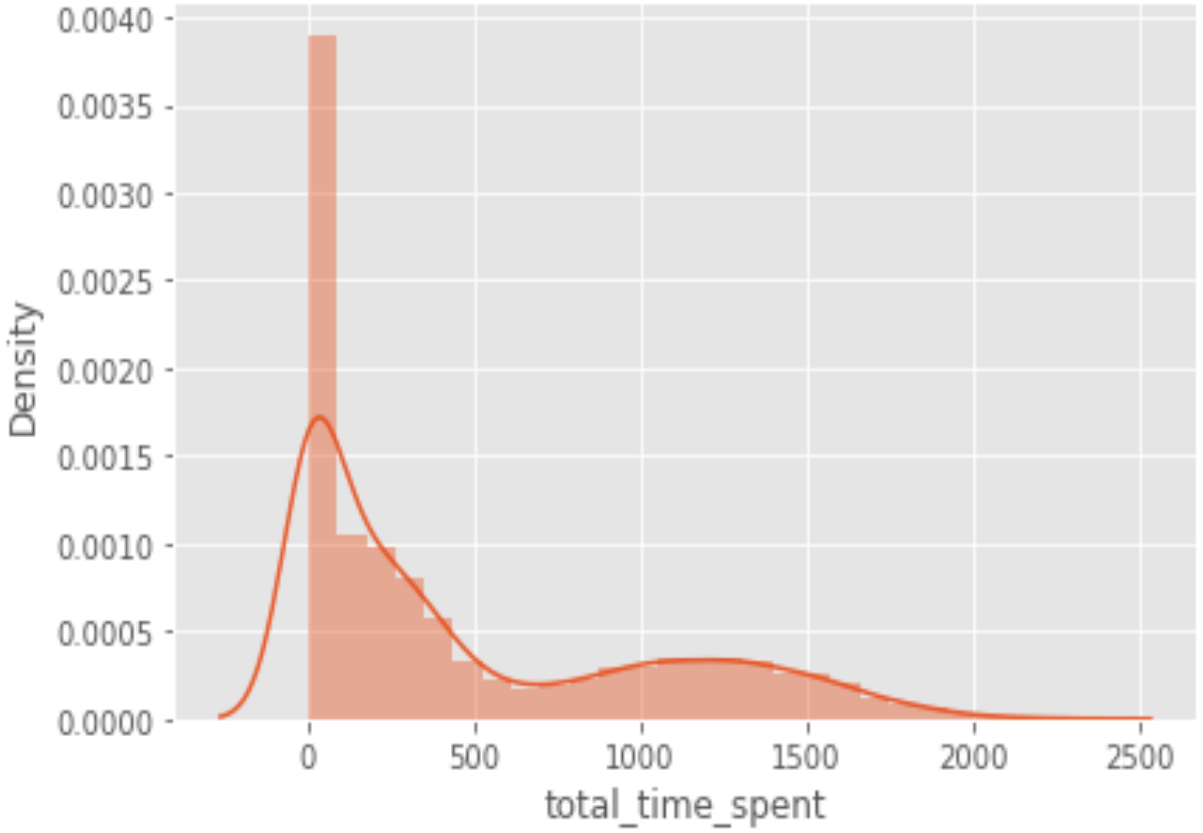
Dropping variables with high multi-collinearity

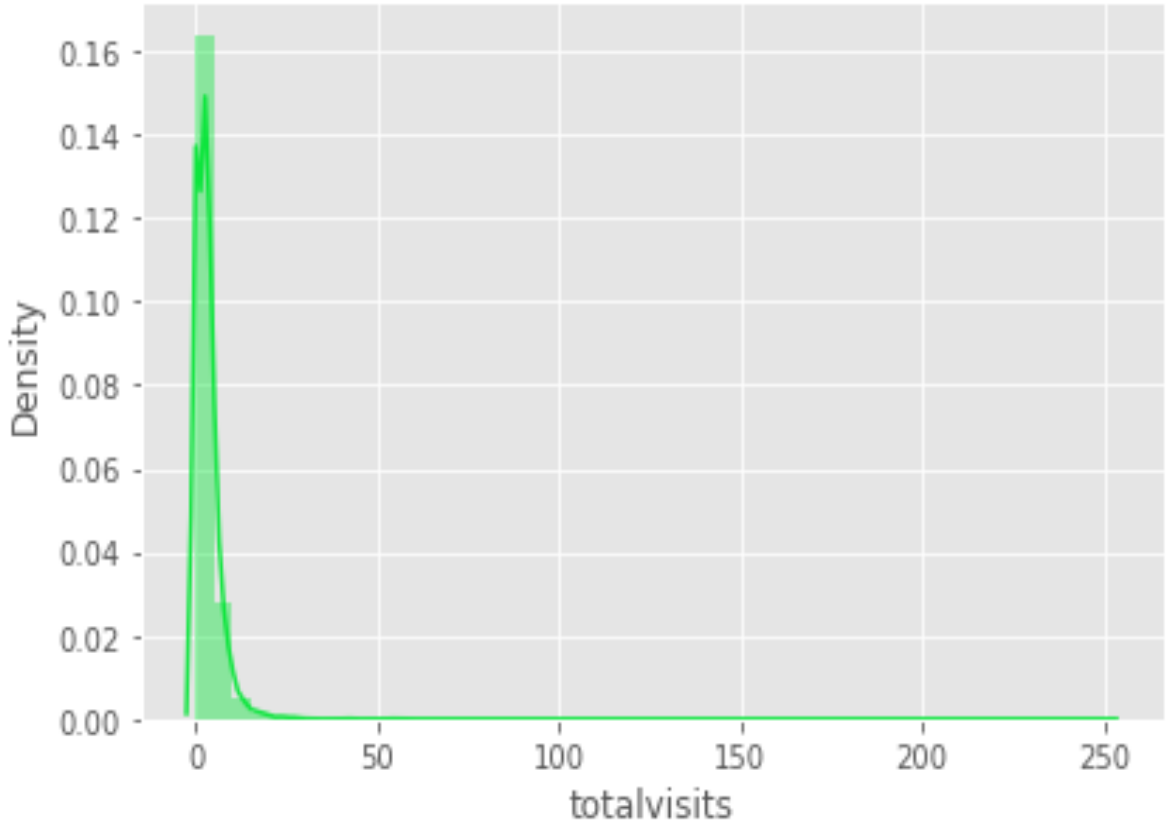
```
# Dropping variables with high multi-collinearity
clean_df.drop(['lead_source_facebook', 'lead_origin_lead_add_form', 'lead_source_olark_chat'], axis=1, inplace=True)

# Top 5 features correlated with target variable
clean_df.corr()['converted'].abs().sort_values(ascending=False).head(6)[1:]
```

```
lead_quality_not_sure             0.443920
last_notable_activity_sms_sent    0.360233
total_time_spent                  0.359261
lead_quality_might_be             0.349936
last_notable_activity_modified    0.263532
Name: converted, dtype: float64
```

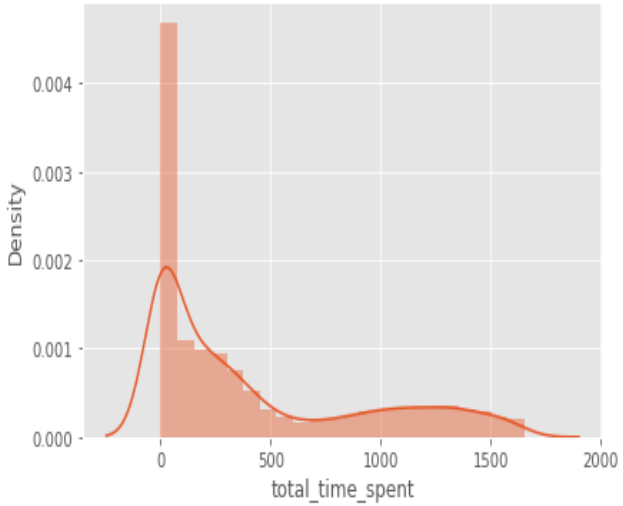
Univariate Analysis and Outlier Analysis

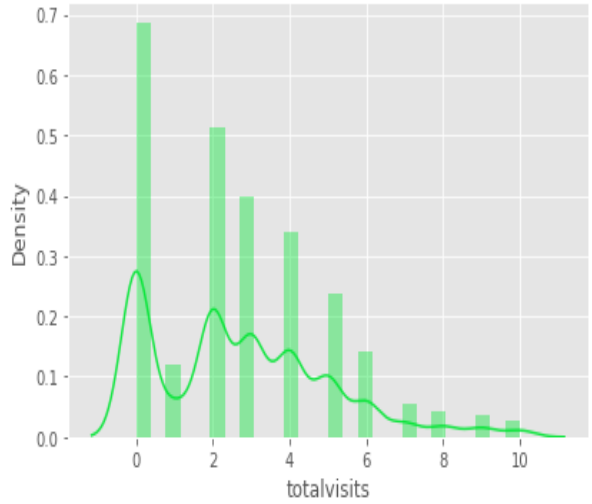


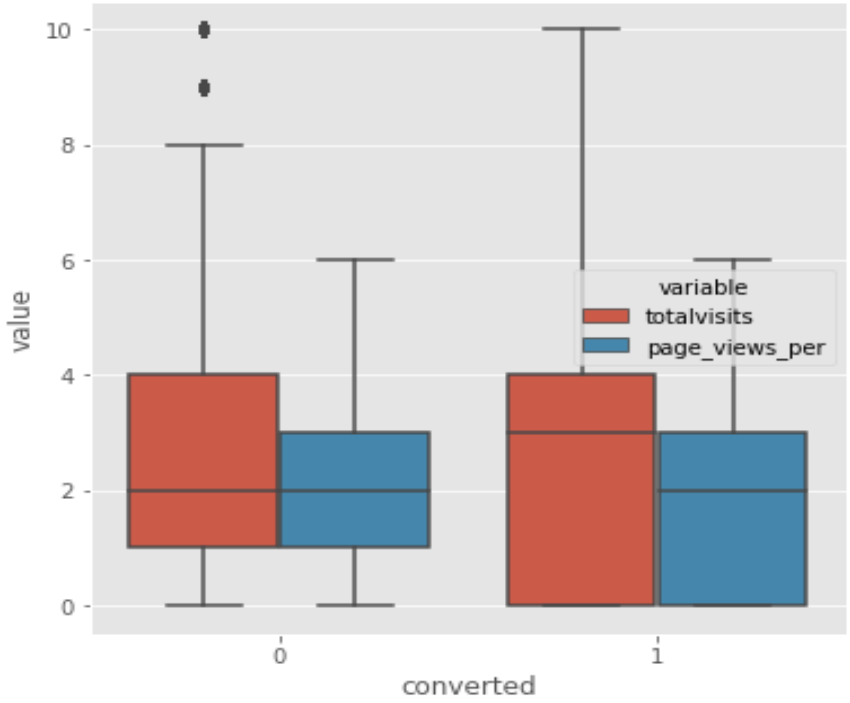


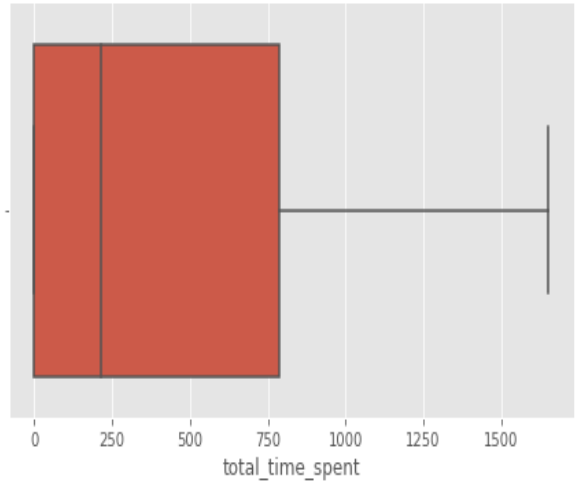
Removing outliers

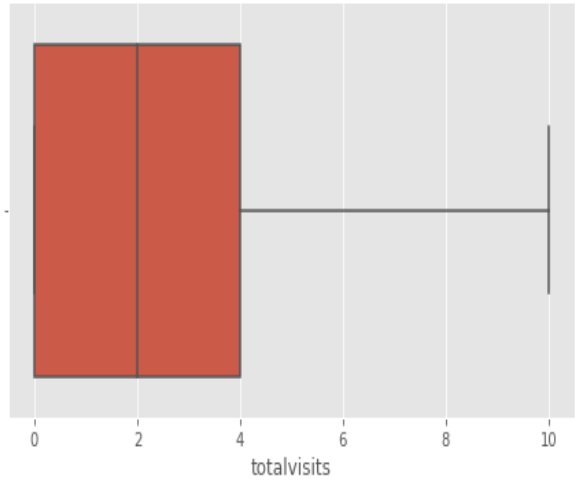
(88.78% data has been retained after outlier removal)



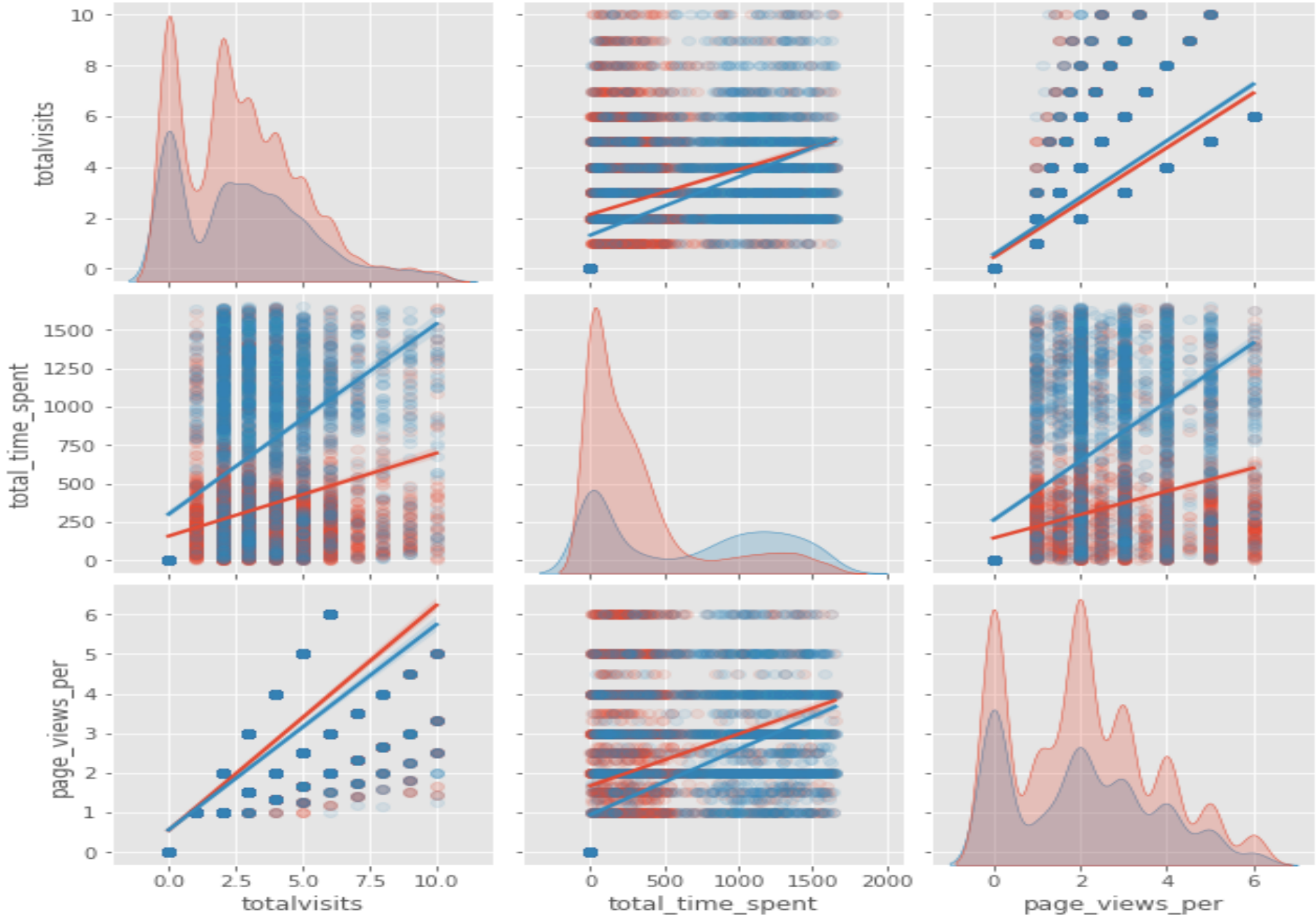








Bivariate Analysis



Splitting into Train and Test

(We will perform stratified split to prevent the effect of class imbalance of target variable, thus preventing bias.)

```
# Stratified Train Test Split
X = clean_df.drop('converted', axis=1)
y = clean_df['converted']
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.33, random_state=42)
```

Normal Scaling

```
# Min Max Scaling
scaler = MinMaxScaler()
cols = X_train.columns
scaled_X_train = pd.DataFrame(scaler.fit_transform(X_train[cols[:3]]), columns=cols[:3])
scaled_X_train = pd.concat([scaled_X_train, X_train.drop(cols[:3], axis=1).reset_index(drop=True)], axis=1)

scaled_X_test = pd.DataFrame(scaler.transform(X_test[cols[:3]]), columns=cols[:3])
scaled_X_test = pd.concat([scaled_X_test, X_test.drop(cols[:3], axis=1).reset_index(drop=True)], axis=1)
```

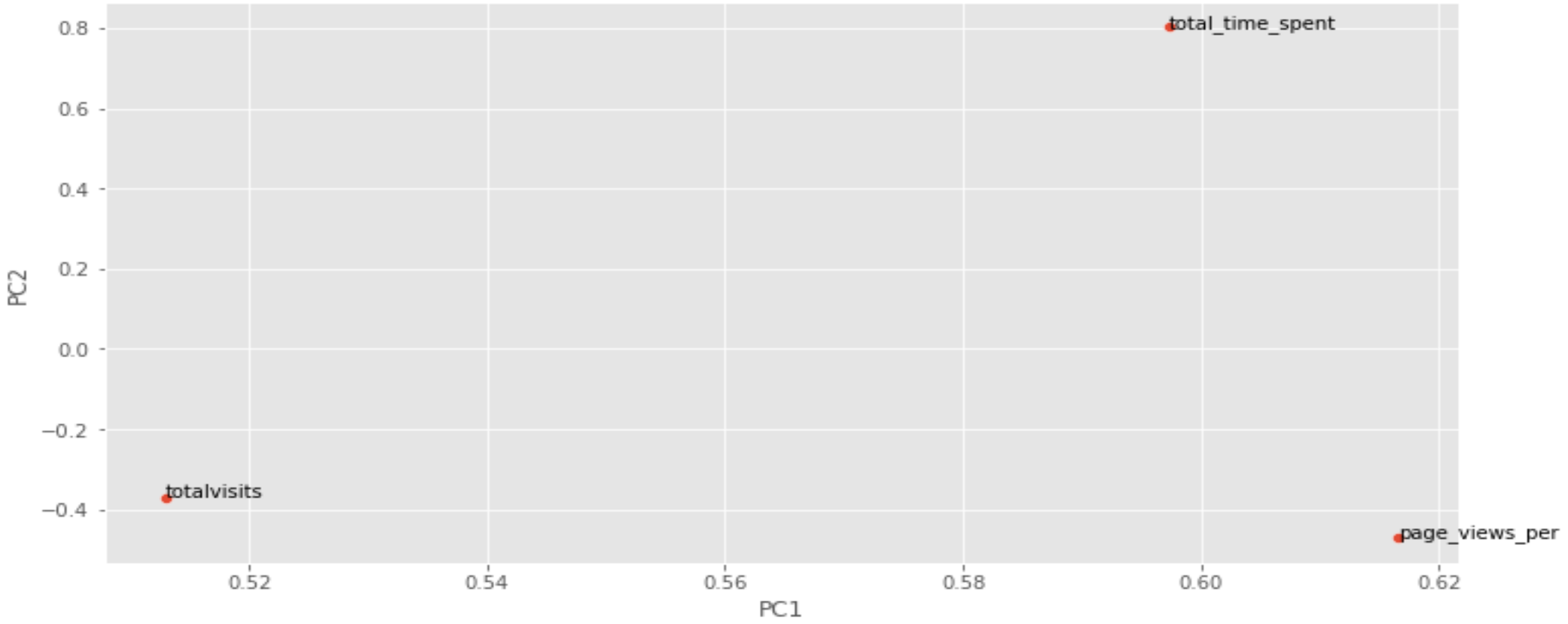
PCA

Generating principal components of continuous variables

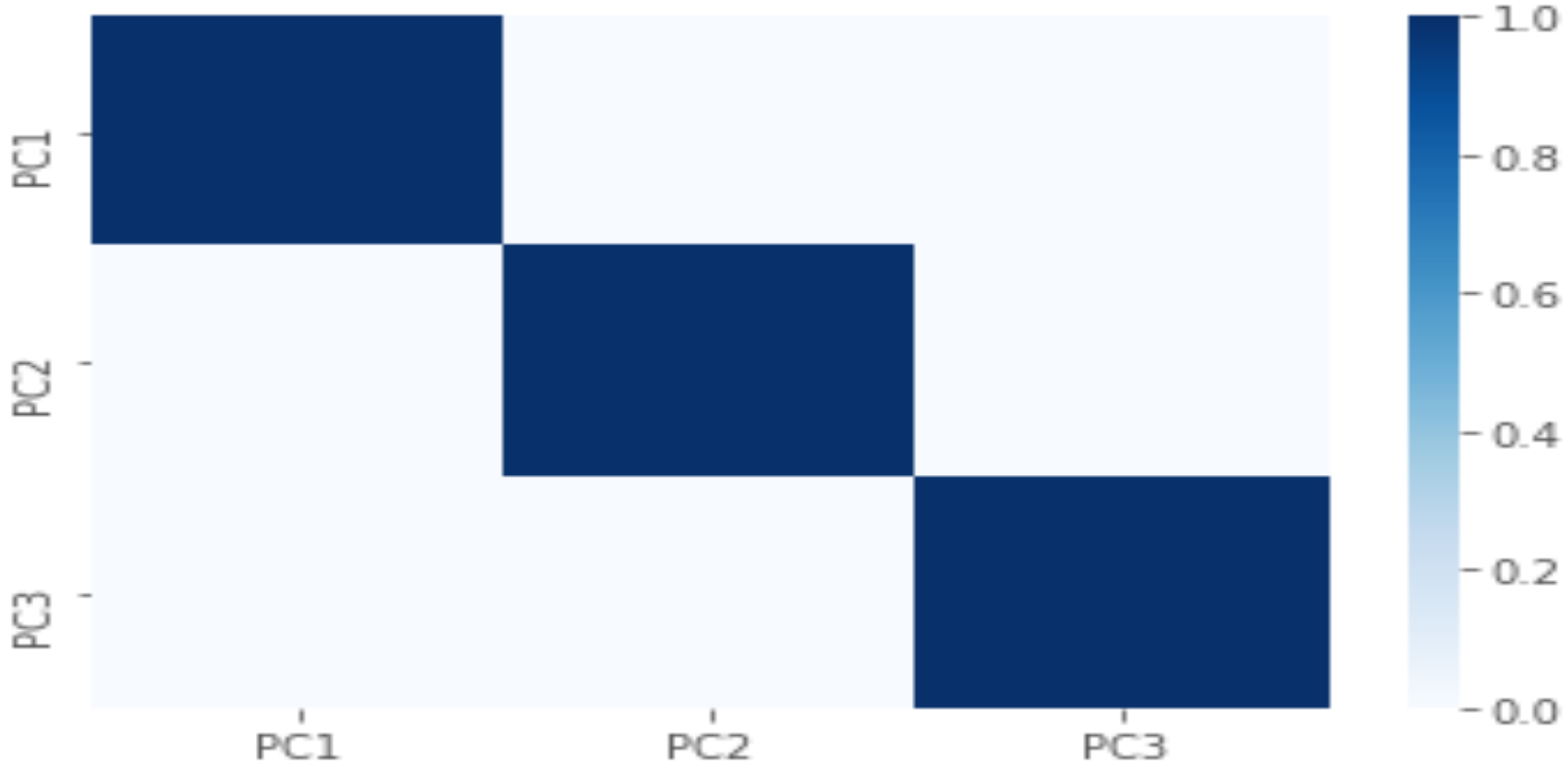
```
# Generating principal components of continuous variables
pca = PCA(random_state=42)
decomp_df = pca.fit_transform(scaled_X_train[cols[:3]])
pc_frame = pd.DataFrame({'Features': cols[:3], 'PC1': pca.components_[0], 'PC2': pca.components_[1], 'PC3': pca.compone
pc_frame
```

	Features	PC1	PC2	PC3
0	totalvisits	0.512978	-0.369512	0.774800
1	total_time_spent	0.597301	0.801911	-0.013018
2	page_views_per	0.616510	-0.469467	-0.632073

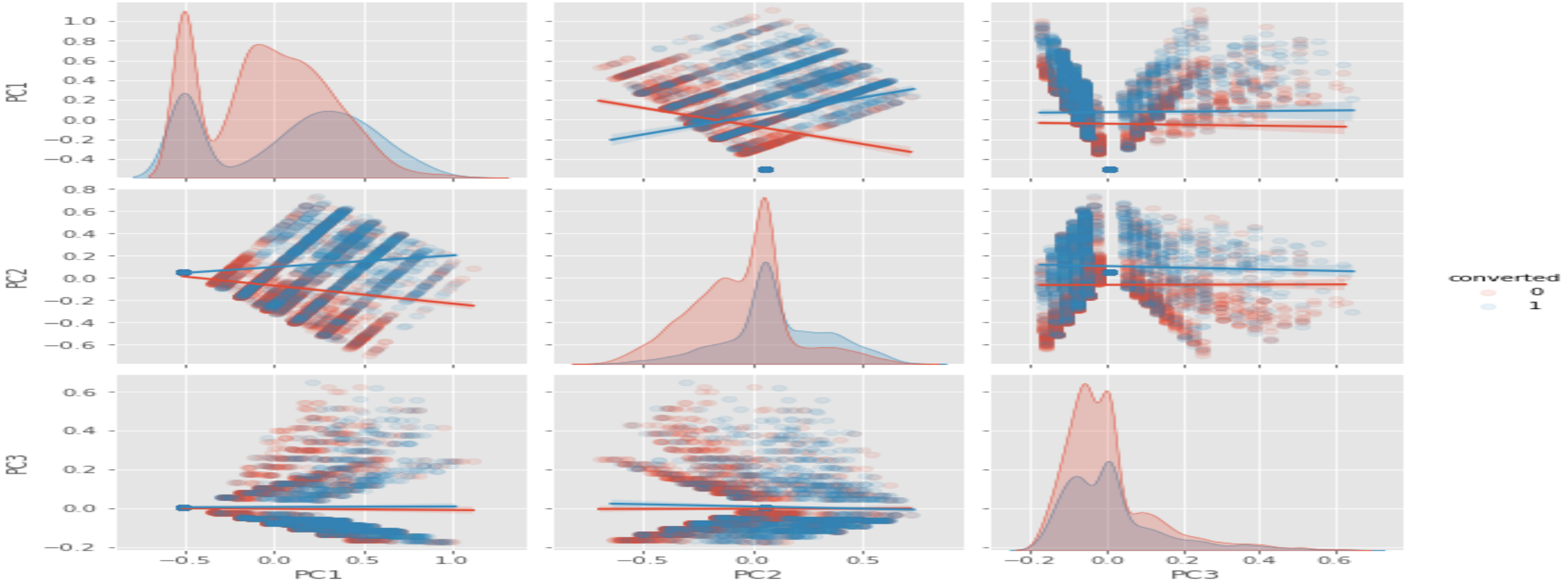
Plotting 2 principal components for the numeric variables



No correlation exists between the PCs

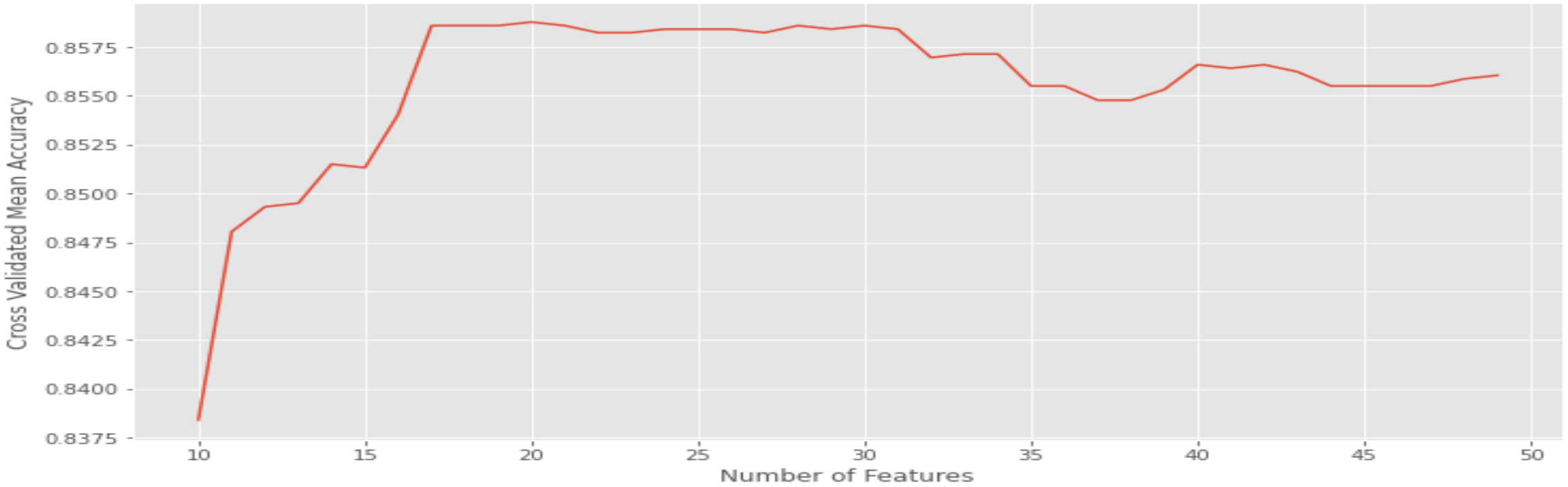


Pair plot Of all 3 PCs



Recursive Feature Elimination and Cross Validation

Plotting how accuracy changes with number of features considered



Optimal number of features to use is 20.0 which gives 0.8587807097361237 accuracy

Model Fitting

```
# Cross Validation
scores = cross_validate(log_reg, scaled_X_train[cols], y_train, return_train_score=True, cv=5, scoring=['accuracy'])
print(f"Cross validated mean accuracy: {round(scores['test_accuracy'].mean(), 3)}")

log_reg.fit(scaled_X_train[cols], y_train)
pred = log_reg.predict(scaled_X_train[cols])
prob_est = log_reg.predict_proba(scaled_X_train[cols])
```

```
Cross validated mean accuracy: 0.859
```

Measuring Model Performance

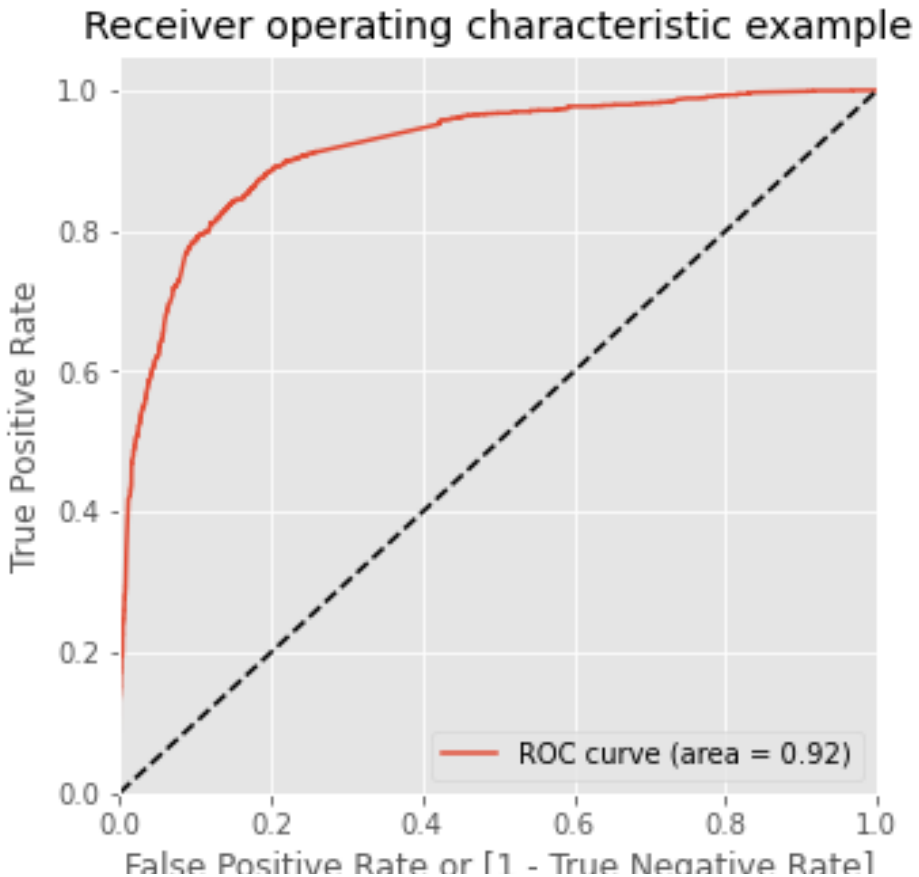
- The data available at hand has class imbalance and therefore accuracy is not a good enough metric to measure if model is good enough.
- Sensitivity (Recall) tells us what percentage of leads that were converted, were correctly identified as converted.
- Specificity tells is what percentage of leads that were NOT converted were correctly identified.

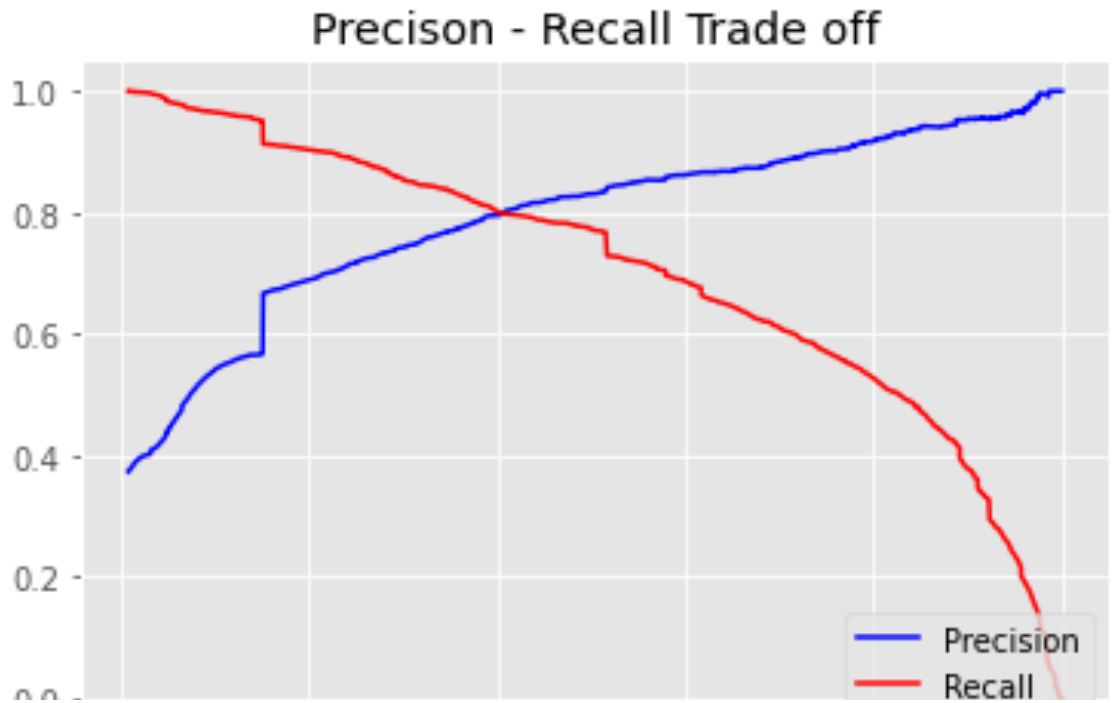
Measuring Model Performance

- Precision is, given a positive test result, the sample is positive.
- If correctly identifying positives is important for us, then we should choose a model with higher Sensitivity. However, if correctly identifying negatives is more important, then we should choose specificity as the measurement metric.
- F1 score is the weighted average of the precision and recall, and is a good metric to hold the model against.

Predicted Values

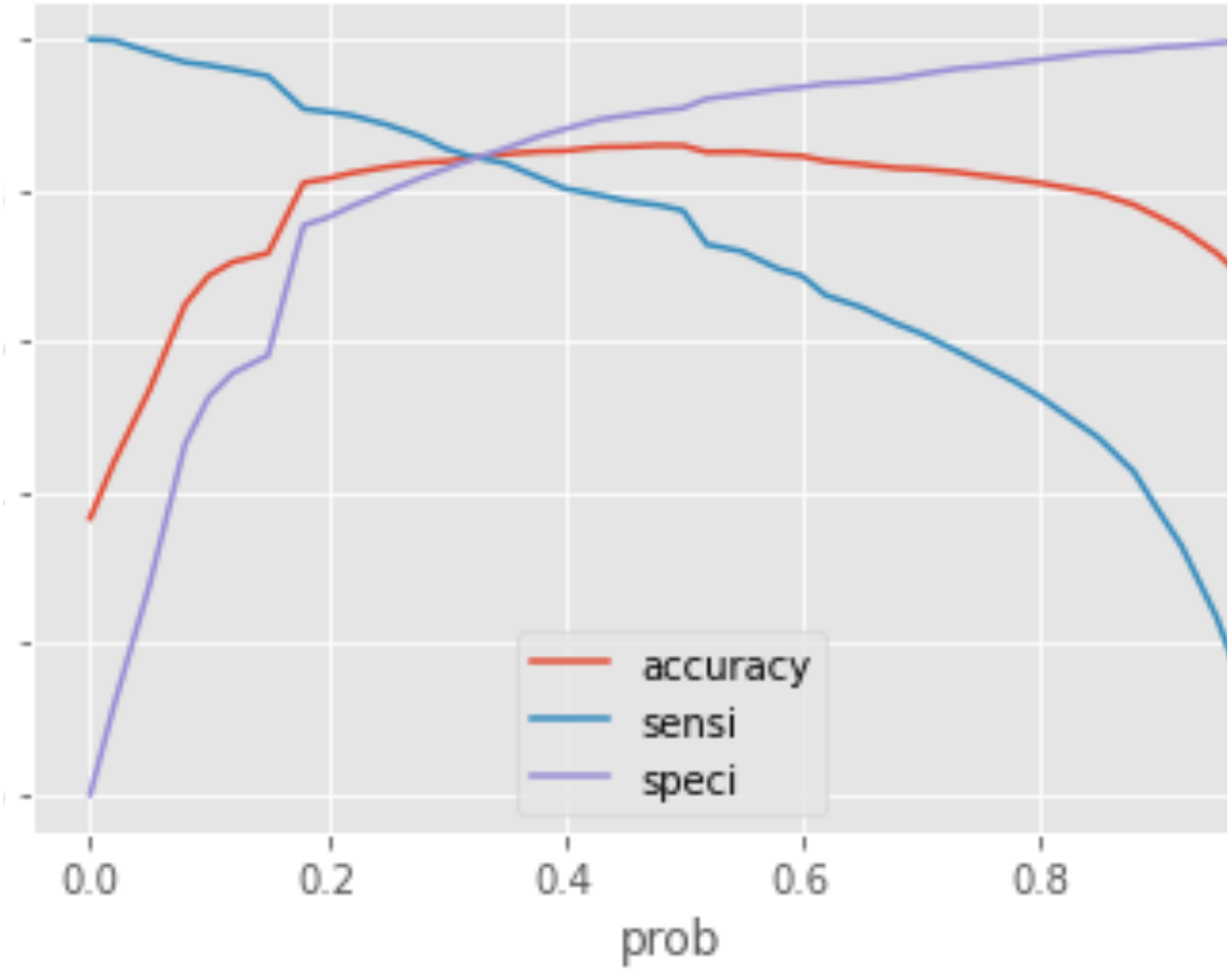
Sensitivity (Recall): 0.773247140726007
Specificity: 0.9090126291618829
Precision: 0.8306623931623932
F-Score: 0.8009271182075715





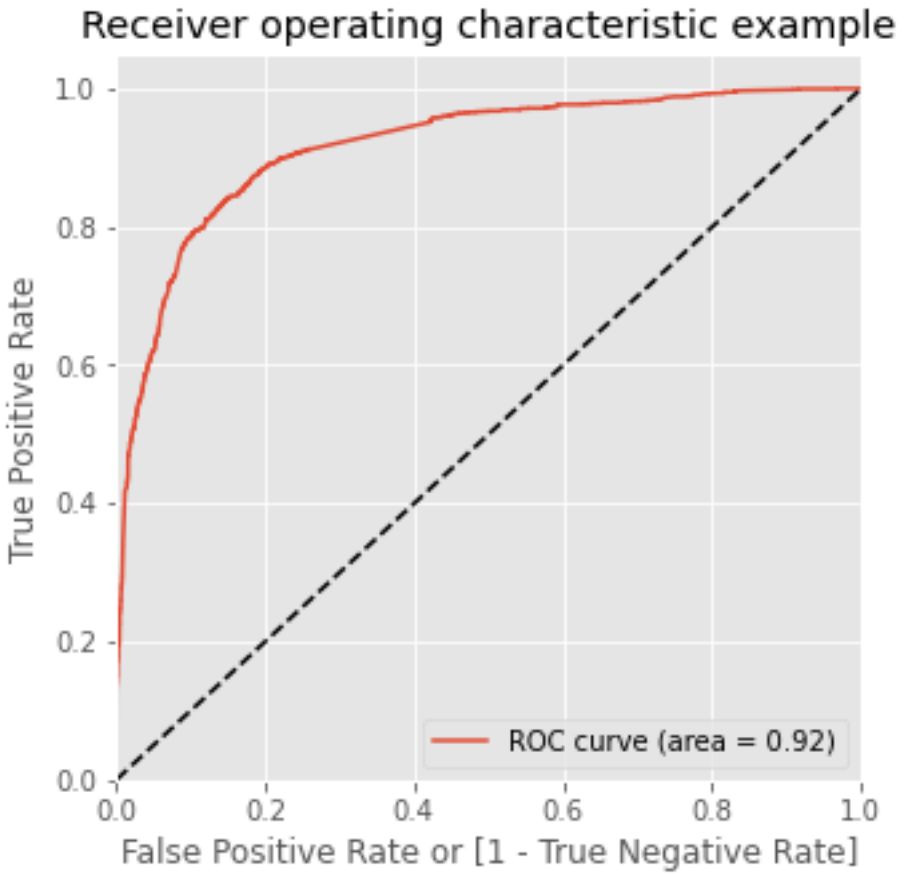
Finding Optimal Cutoff

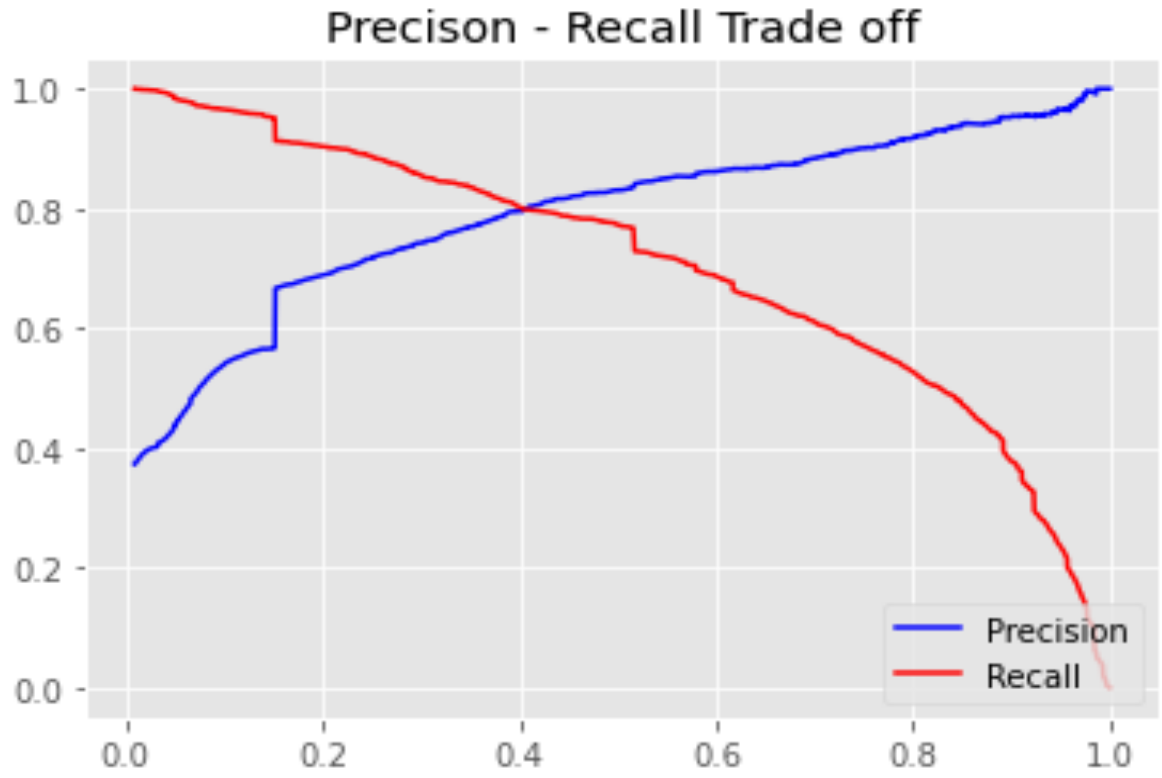
Optimum cut-off value is: 0.32



New predicted values based on cut-off

- Accuracy: 0.8420382165605096
- Sensitivity (Recall): 0.8453505718547986
- Specificity: 0.8401262916188289
- Precision: 0.7532122286220647
- F-Score: 0.796260543580131

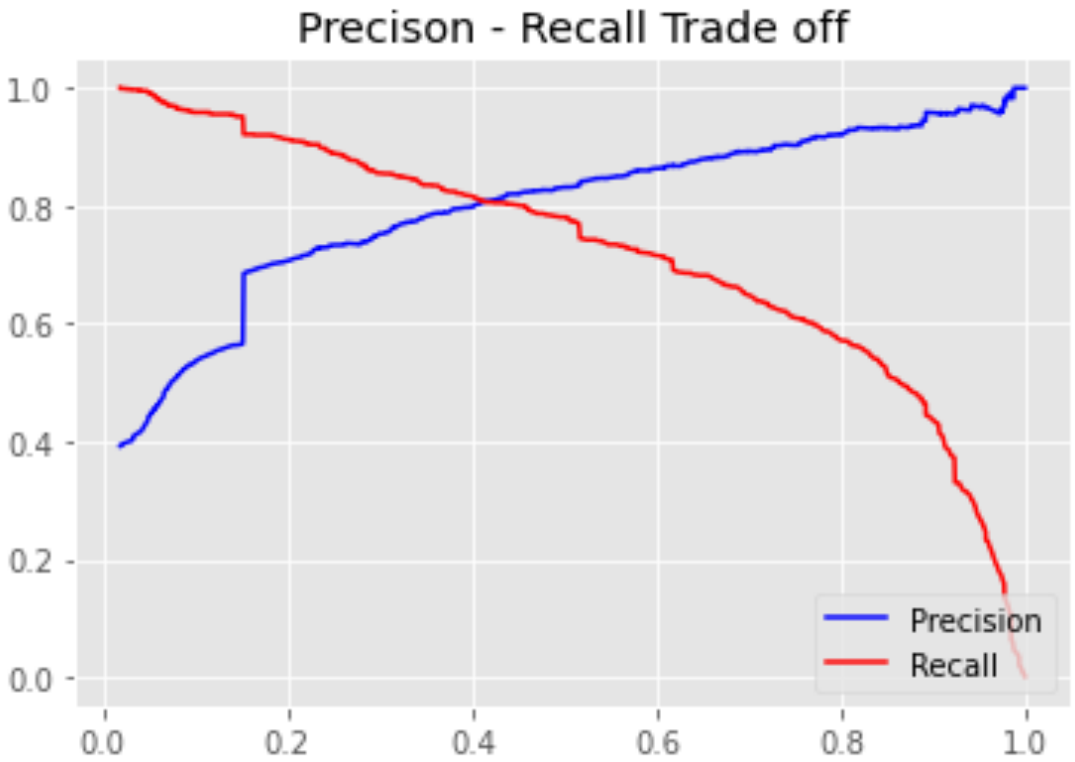


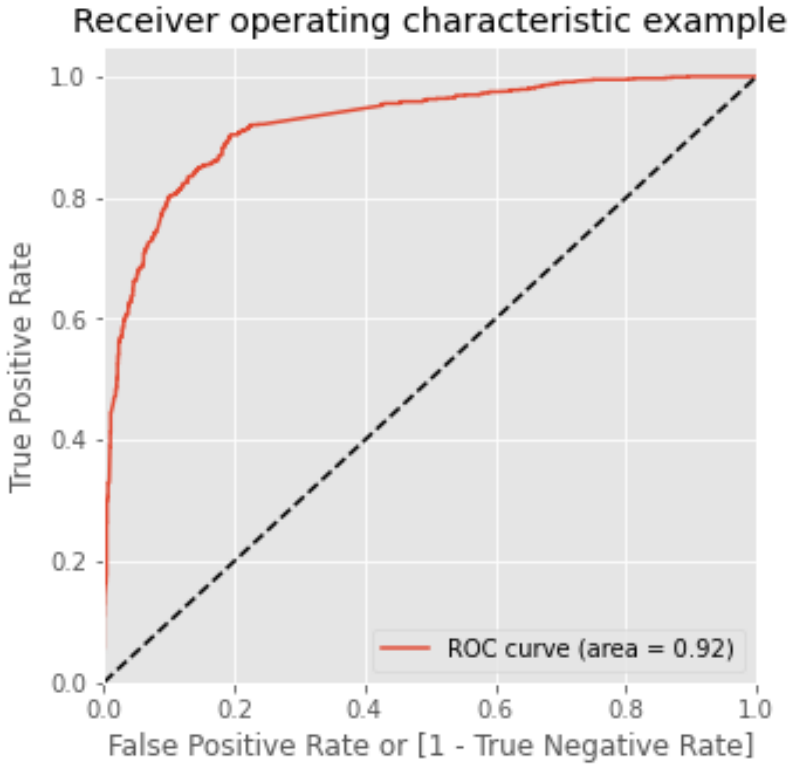


Analyzing the model and its performance on the test set

Analyzing the model and its performance on the test set

- accuracy: 0.8511267085334319
- Sensitivity (Recall): 0.8505050505050505
- Specificity: 0.8514851485148515
- Precision: 0.7675478577939836
- F-Score: 0.8068998562529947





Conclusion

In our Model, we have an overall accuracy of about 0.85 on our Logistic Regression model. That is, there is 85% chance that our predicted leads will be converted. This meets the CEO's target of at least 80% lead conversion.