Lead Scoring

By
Felix Philip, Hitesh Singhal, Ranjan N Lokegaonkar

Agenda

- The purpose of the case study is to build a logistic regression model which will optimize and output the variables that will contribute towards a lead being converted for X education
- The variables outputted by the model will be beneficial for the sales team for lead generation, lead nurturing and retention
- The univariate and bivariate analysis of categorical and non categorical variables from the data collected in order to have a overview of the variables of the variables

Problem Statement

- X Education's lead conversion rate is currently below the industry average. This means that for every 100 leads that X Education generates, only a small percentage of them are actually converting into customers.
- X Education is spending a significant amount of money to acquire leads. However, because the conversion rate is low, the company is not seeing a good return on its investment

Approach and Methodology

- Source the data for analysis
- Read and Understand the data
- Data Cleaning and EDA
- Select the categorical columns.
- Convert categorical variables into dummy variables.
- Split the dataset into features (X) and target variable (y)
- Split the data into training and testing sets
- Perform feature scaling
- Feature Selection
- Evaluate the model
- Use the model to predict the probability of leads converting

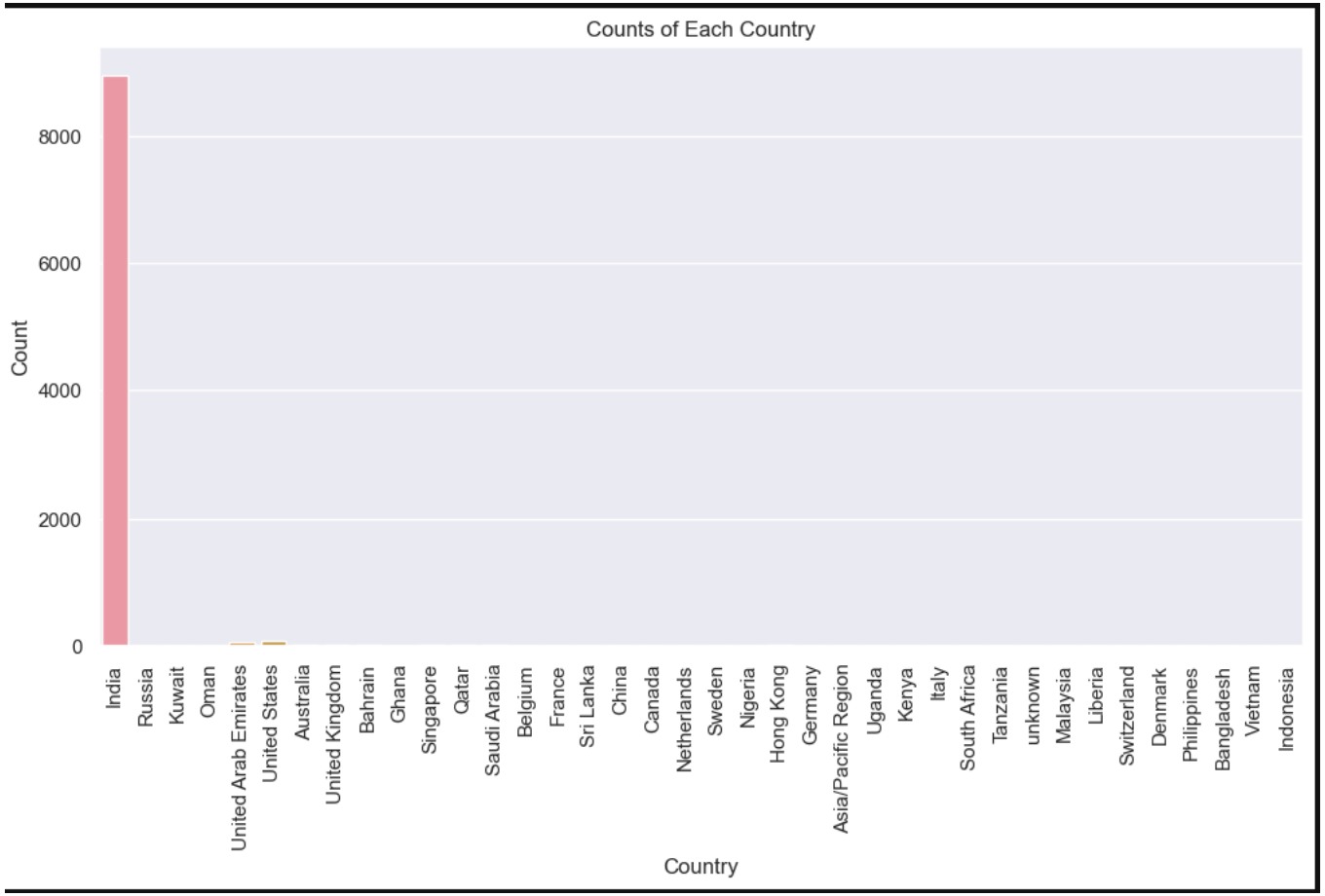
Data Sourcing, Cleaning and Preparation

- Read the data from CSV file
- Outlier treatment
- Data cleaning handling null values & handling null values data
- Removing redundant columns
- Imputing Null Values
- Feature standardization

Exploratory Data Analysis

- After Exploratory Data Analysis and handling of missing data. Univariate and Bivariate Analysis have been done on the categorical and non categorical variables
- The Results of Univariate and Bivariate Analysis on the provided data are displayed in successive slides

Univariate analysis of Country

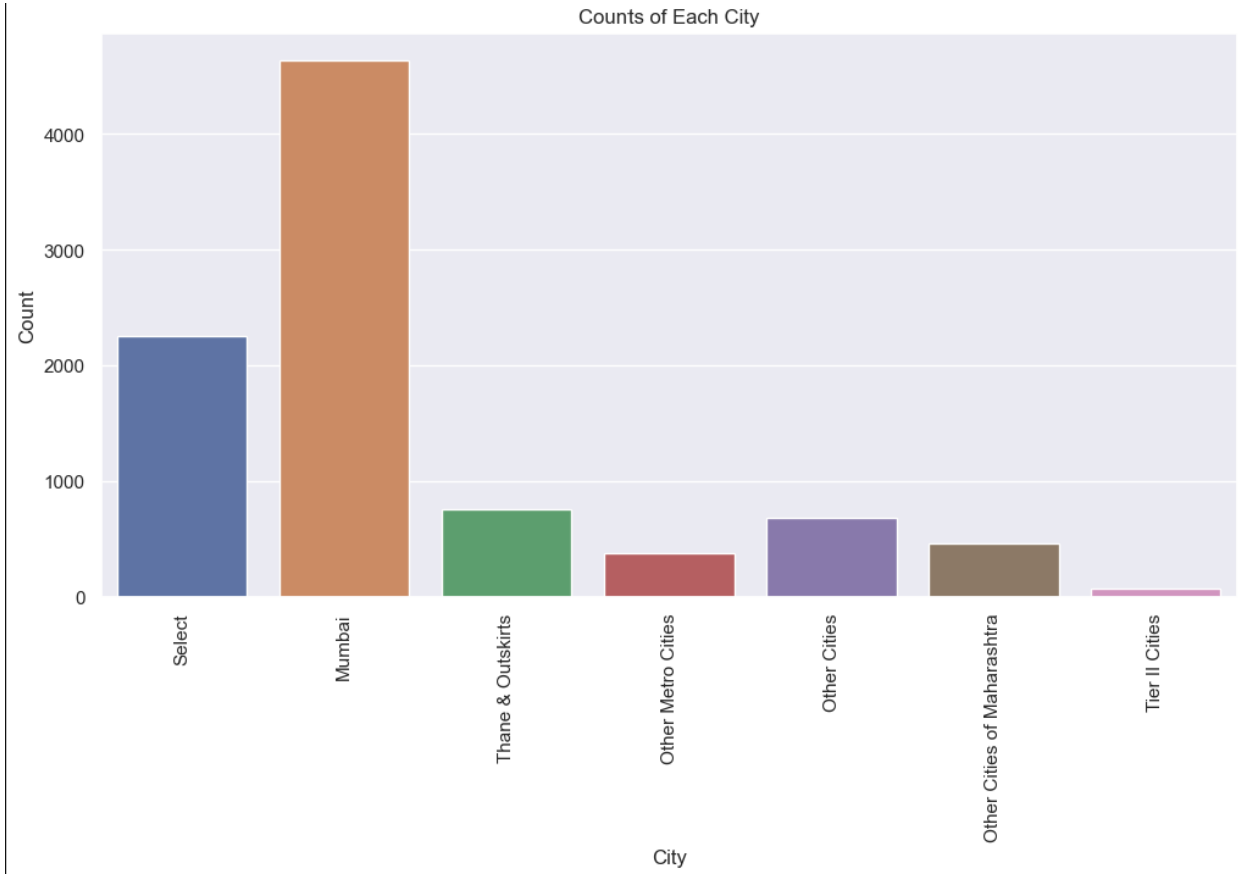


We can infer from the Univariate analysis most of the students are from India and few from UAE and USA

More sales and lead generation should be done on UAE and USA to increase the business

Lead retention and nurturing should be done in India

Univariate Analysis Count of Each City

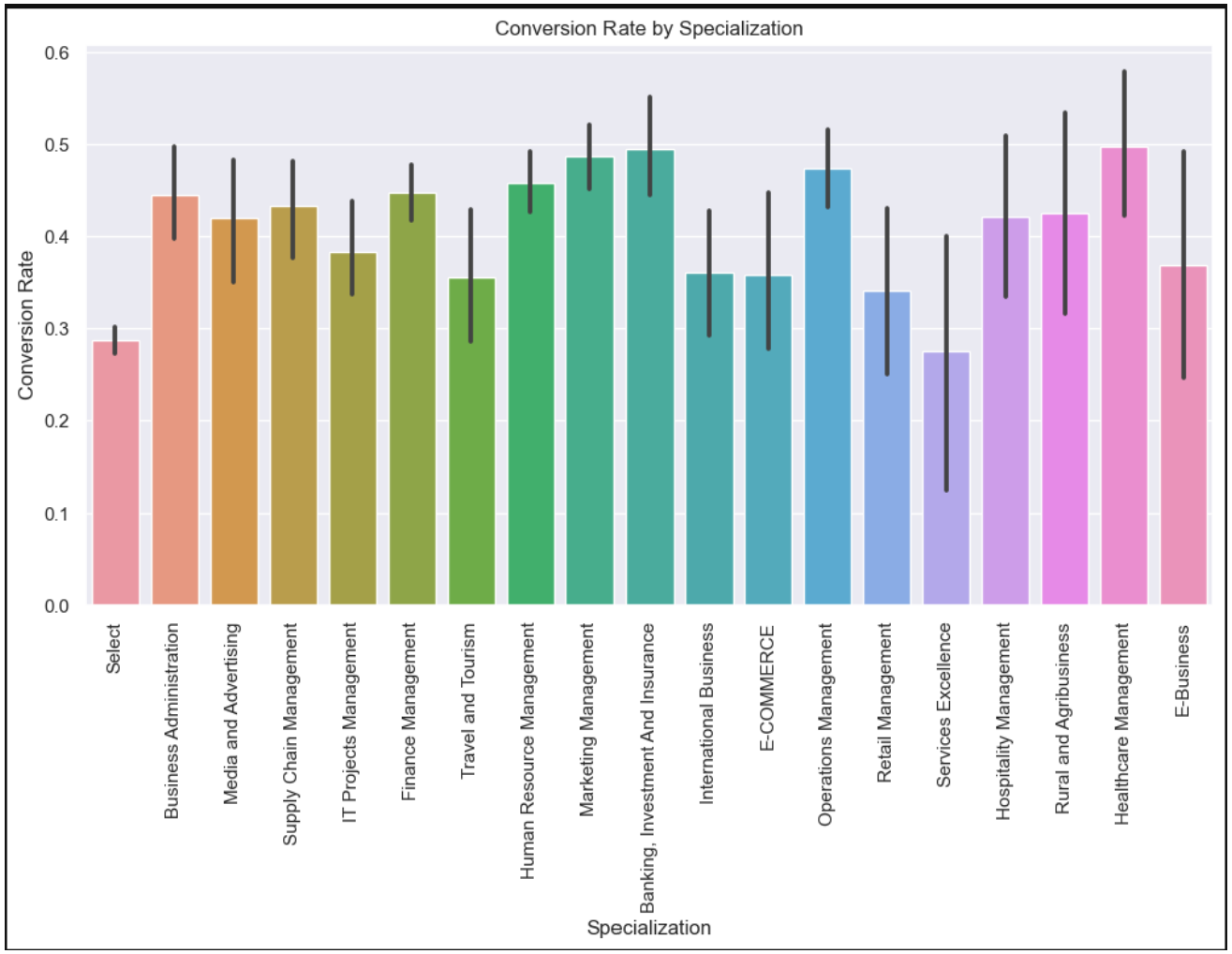


Its evident from the graph most of the students enrolled from Maharashtra

Lead nurturing and retention should be done for students from Maharashtra

Lead generation should be done on other major states as well in order to increase business in India

Univariate Analysis Specialization

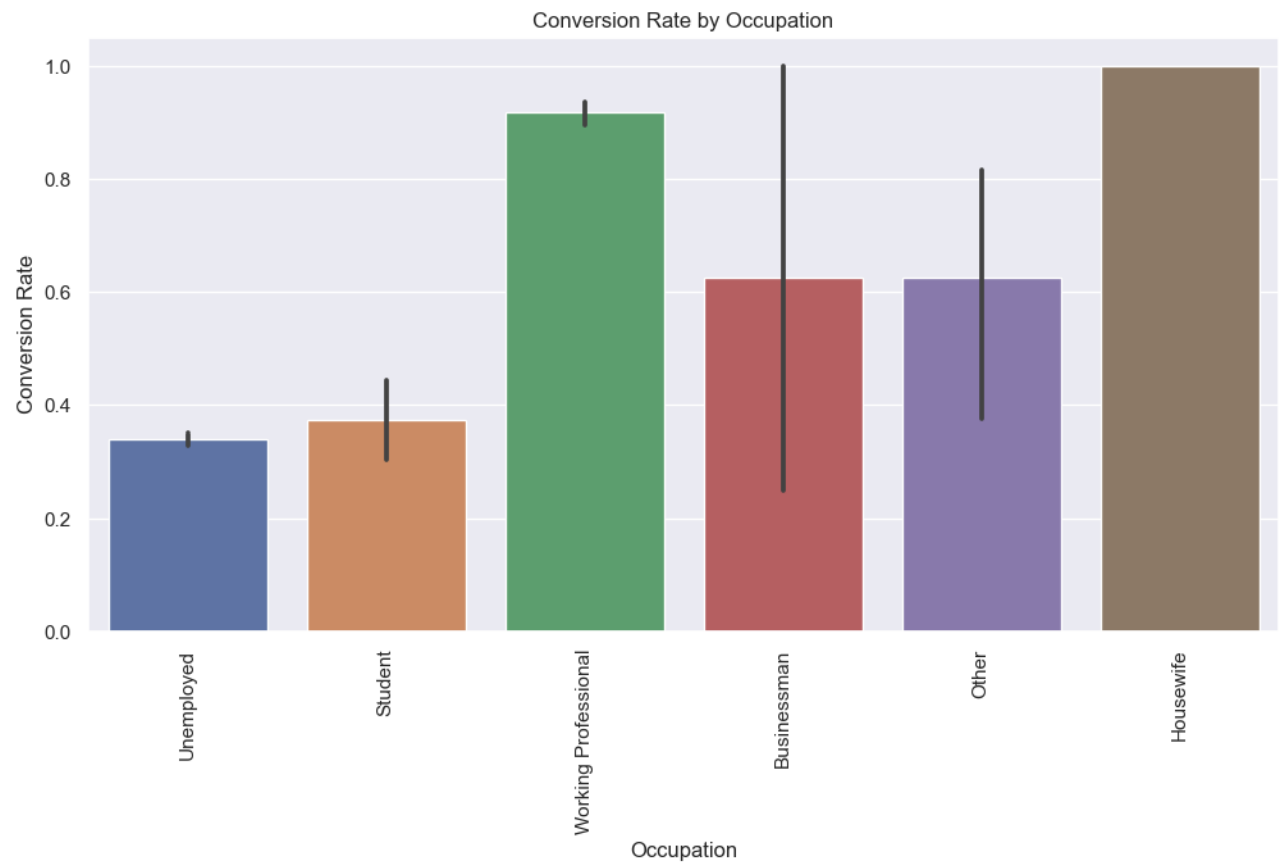


We can see most of the specialization enrolled by the students are from management domain

Lead nurturing and retention should be done for leads from management domain

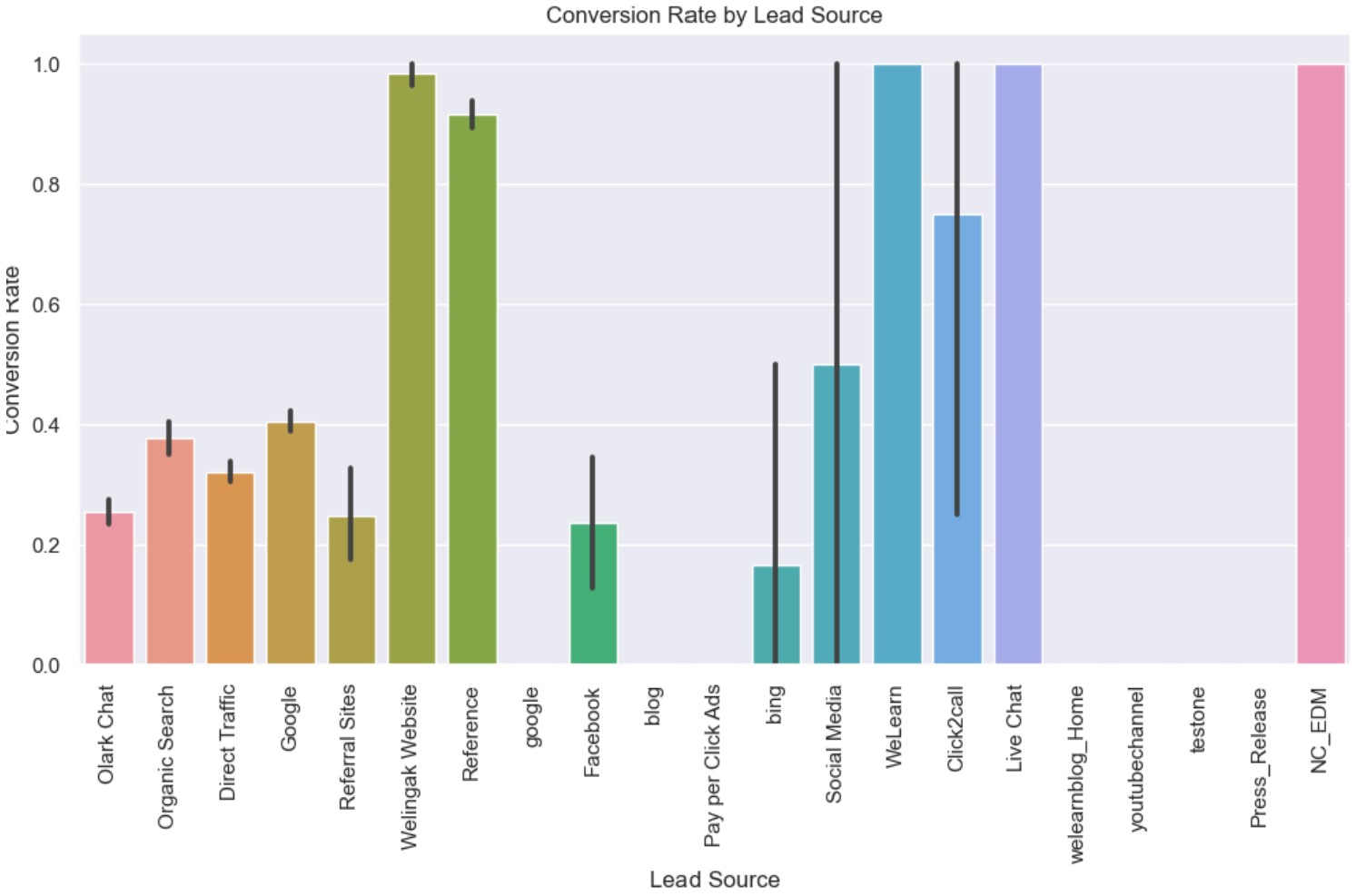
Lead generation should be done for courses in other domains

Bivariate Analysis Conversion vs Occupation



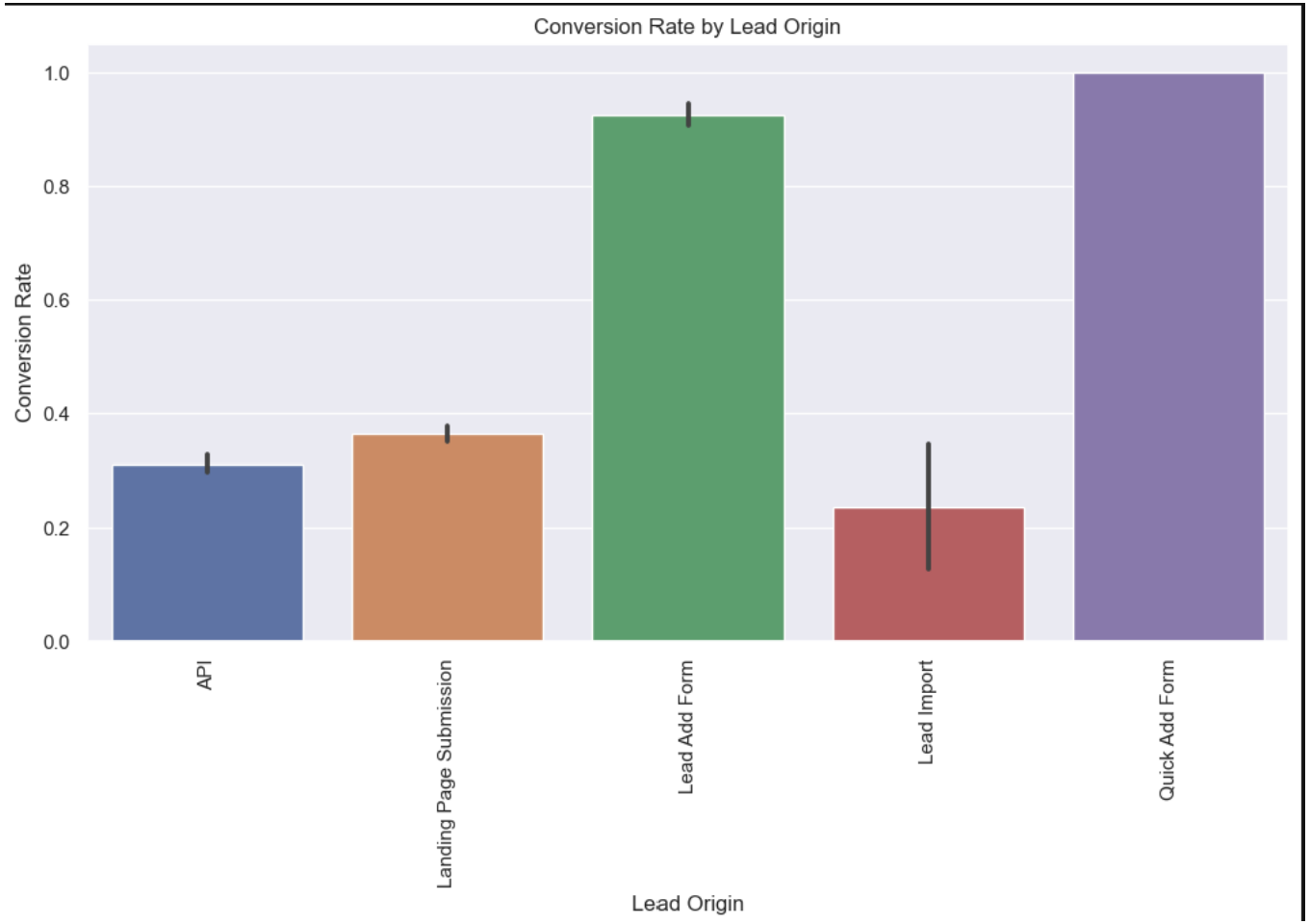
We can infer from graph that working professionals are more likely to enrol

Bivariate Analysis Conversion vs Lead Source



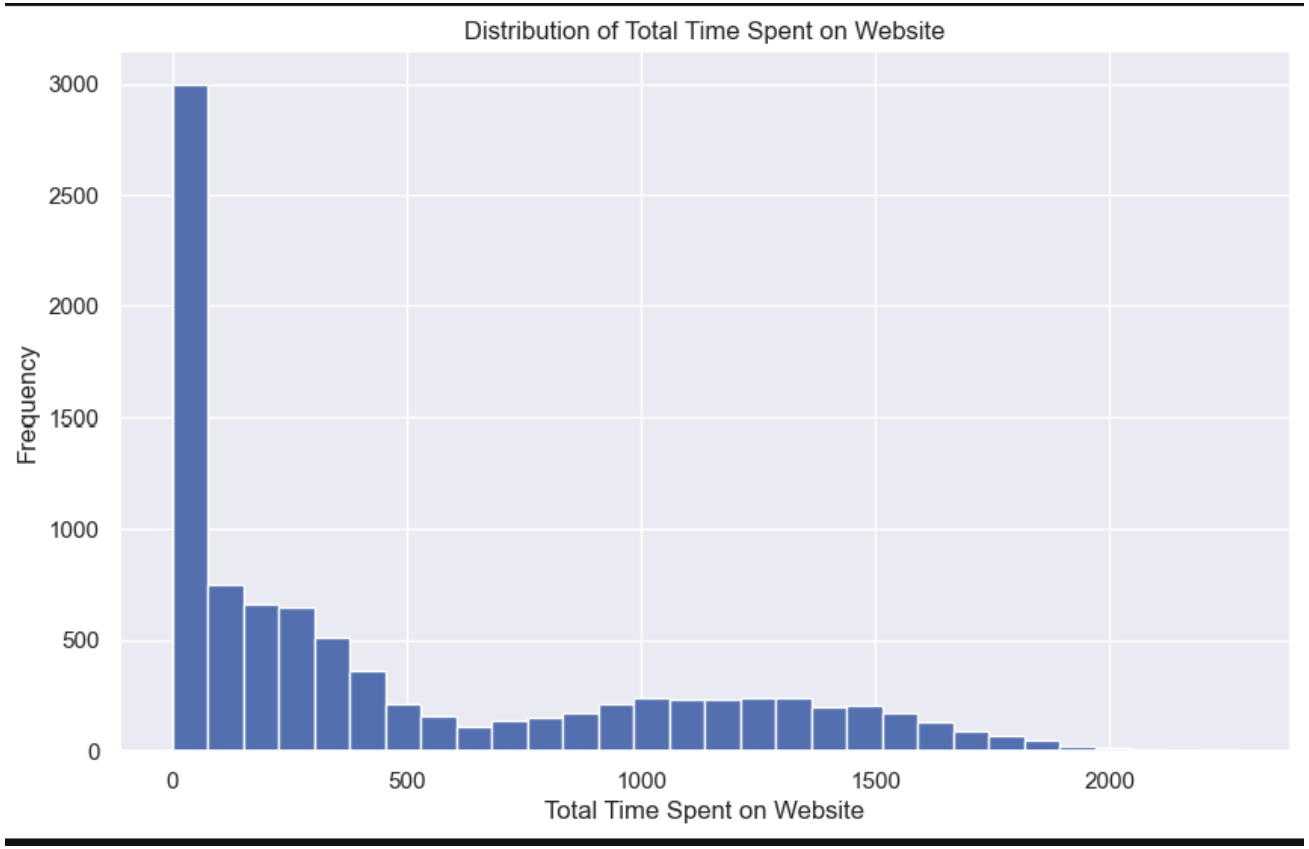
We can infer from the graph the following are good sources for getting students
Welingak Website,Reference,Welearn ,Click2call,LiveChat

Bivariate Analysis Conversion Vs Lead Origin



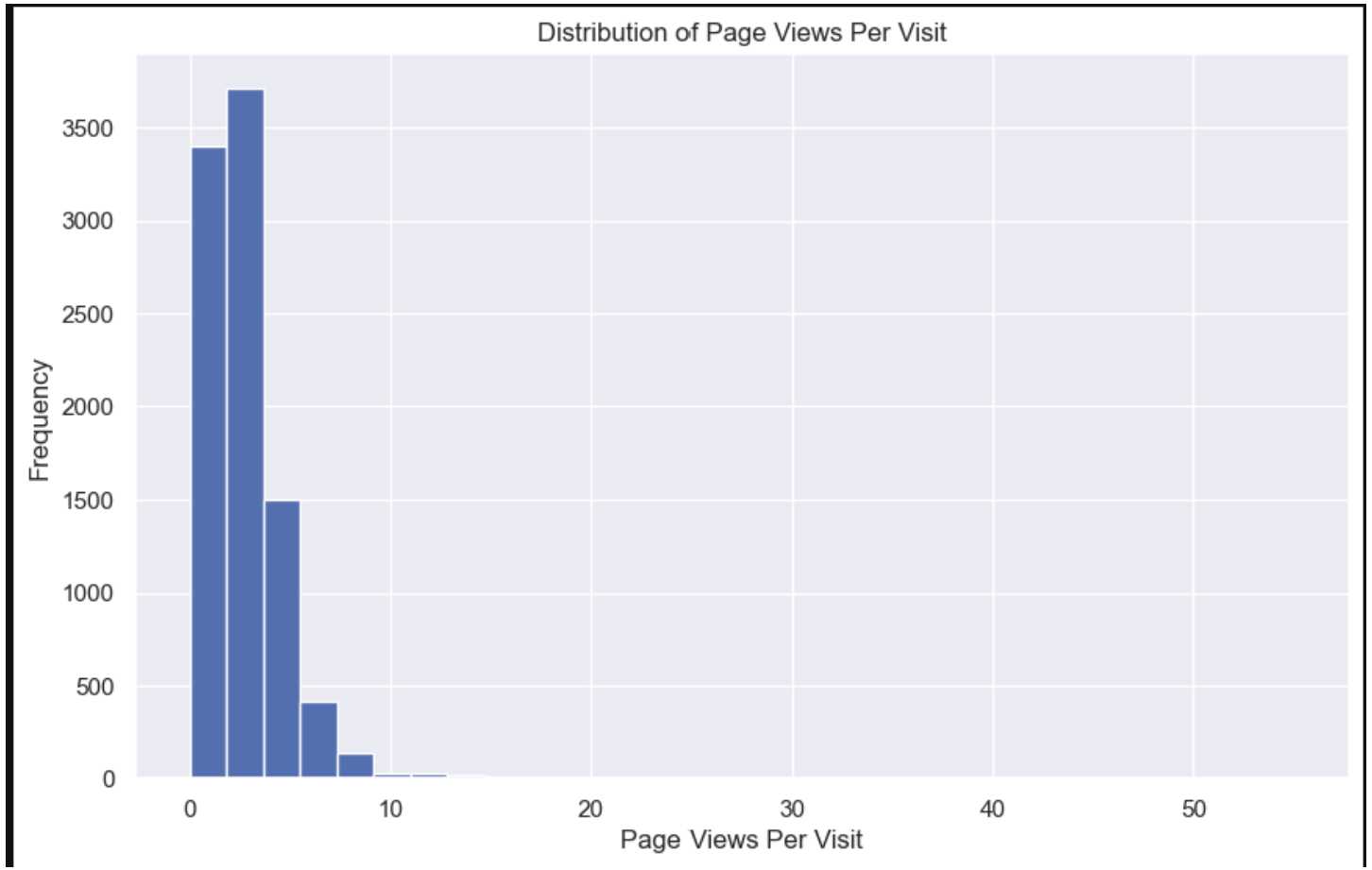
We can infer from graph the highest lead origin conversions are from forms which are lead add form and quick add form

Univariate Analysis Total Time On Page



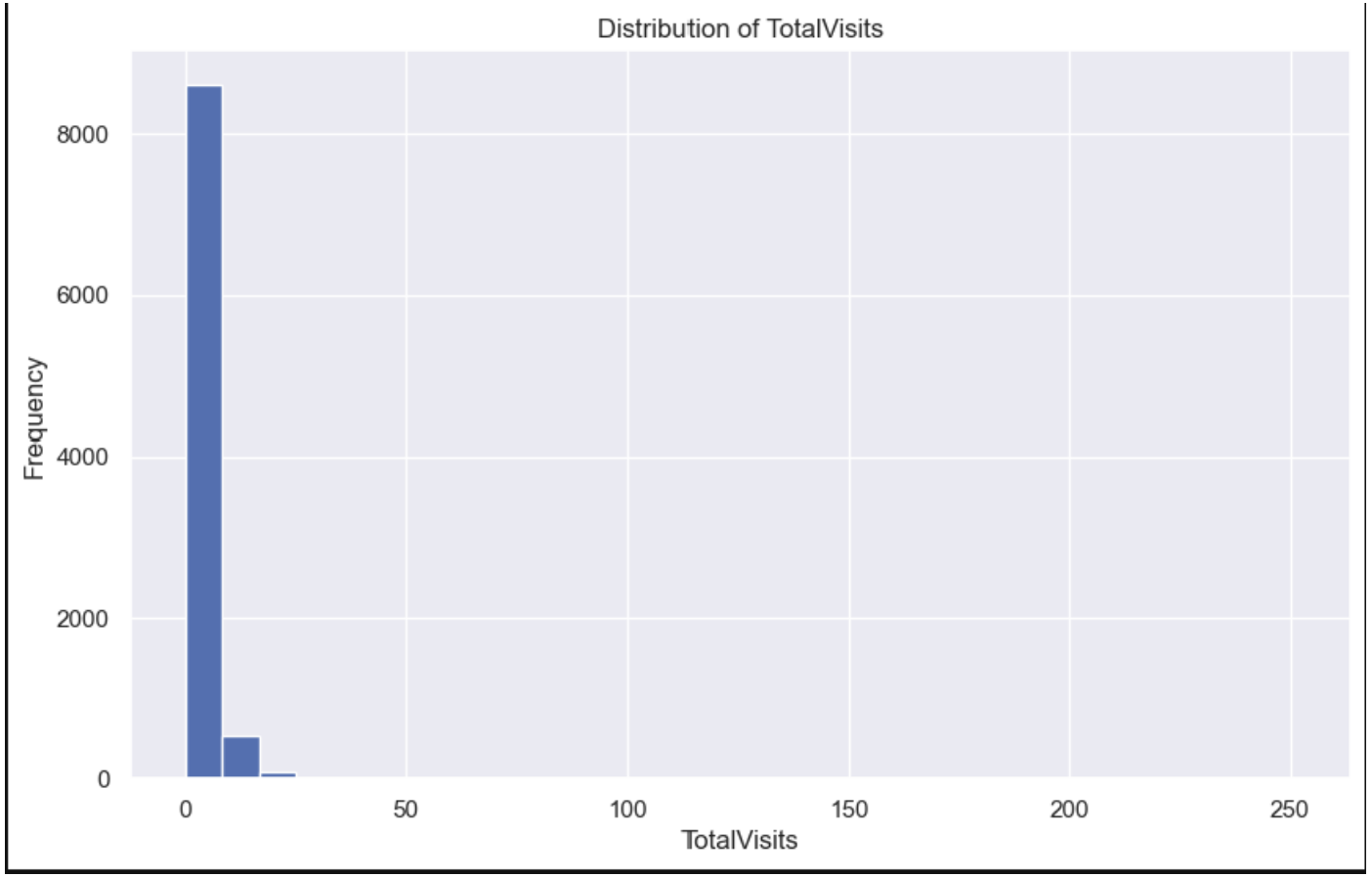
We can infer most of the users fall in the 0 category and anything above 0 should be considered as a prospect

Univariate Analysis Page View Per visit



We can consider any page views greater than 0 as a prospect and there is a correlation between Total time on page and page views per visit

Univariate Analysis Distribution Of Total visits



Most of the users fall in the 0 total visits category so any users with greater than 0 should be considered as a prospect and should be eligible for lead generation

Select the categorical columns & Convert categorical variables into dummy variables

- Select the categorical columns from the dataset. The select_dtypes function from the pandas library is used to do this
- Converts the categorical columns into dummy variables. The get_dummies function from the pandas library is used to do this. The drop_first argument is set to True to remove the first dummy variable for each categorical column. This is done to avoid the dummy variable trap

Split the dataset into features (X) and target variable (y)

- The first line of code drops the Converted column from the dataset. The drop function from the pandas library is used to do this. The axis argument is set to 1 to specify that the column should be dropped from the columns axis.
- The Second line of code assigns the Converted column to the variable y. This is done so that the Converted column can be used as the target variable for the model.
- By splitting the dataset into features and target variable, we can train the model on the features and then evaluate the model's performance on the target variable

```
X = data.drop('Converted', axis=1)
y = data['Converted']
```

Split the data into training and testing sets

- The train_test_split function from the scikit-learn library is used to split the data.
- The X and y arguments are used to specify the features and target variable, respectively.
- The test_size argument is set to 0.3 to specify that 30% of the data should be used for the testing set.
- The random_state argument is set to 42 to ensure that the data is split randomly.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
```

Perform feature scaling

- The first line of code creates a StandardScaler object.
- The second line of code fits the StandardScaler object to the training set. And transforms the training set using the StandardScaler object.
- The third line of code transforms the testing set using the StandardScaler object.

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

- We perform feature scaling to bring all the features to a common scale so that they have the same weight and importance when training the model.

Feature Selection

- The code you provided selects the features that are most important for predicting the target variable. The features are selected using a recursive feature elimination (RFE) algorithm.
- The RFE algorithm starts with all of the features and then iteratively removes features that are not important for predicting the target variable. The algorithm stops when the desired number of features has been selected.

```
selected_features = X_train.columns[rfe.support_]
print("Selected Features:")
print(selected_features)
```

Evaluate the model (GLM)

- The model was trained using a Generalized Linear Model (GLM) with the Binomial family and Logit link function
- The dataset used for training consisted of 6,468 observations.
- The model achieved convergence after 24 iterations.
- The pseudo R-squared (CS) value, a measure of model fit, is 0.6073, indicating a good fit
- The overall significance of the model is supported by the p-value (<0.001) of the Wald chi-square test.

Generalized Linear Model Regression Results

Generalized Linear Model Regression Results

Dep. Variable:	Converted	No. Observations:	6468
Model:	GLM	Df Residuals:	6452
Model Family:	Binomial	Df Model:	15
Link Function:	Logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-1285.0
Date:	Fri, 19 May 2023	Deviance:	2570.0
Time:	23:11:49	Pearson chi2:	1.05e+04
No. Iterations:	24	Pseudo R-squ. (CS):	0.6073
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.7595	0.107	-25.792	0.000	-2.969	-2.550
Total Time Spent on Website	0.0016	9.5e-05	17.321	0.000	0.001	0.002
Lead Source_Welingak Website	25.3514	1.91e+04	0.001	0.999	-3.74e+04	3.74e+04
Last Activity_SMS Sent	2.1985	0.114	19.288	0.000	1.975	2.422
Tags_Closed by Horizzon	7.4264	0.725	10.241	0.000	6.005	8.848
Tags_Diploma holder (Not Eligible)	-23.3773	3e+04	-0.001	0.999	-5.87e+04	5.87e+04
Tags_Interested in other courses	-2.1994	0.392	-5.610	0.000	-2.968	-1.431
Tags_Lost to EINS	6.6418	0.631	10.521	0.000	5.404	7.879
Tags_Ringing	-3.3410	0.232	-14.422	0.000	-3.795	-2.887
Tags_Will revert after reading the email	5.0046	0.180	27.749	0.000	4.651	5.358
Tags_number not provided	-25.3450	3.86e+04	-0.001	0.999	-7.56e+04	7.55e+04
Tags_switched off	-4.2284	0.731	-5.786	0.000	-5.661	-2.796
Tags_wrong number given	-25.0022	3.07e+04	-0.001	0.999	-6.01e+04	6.01e+04
City_Select	1.5174	0.126	12.010	0.000	1.270	1.765
Asymmetrique Activity Index_03.Low	-2.0173	0.474	-4.254	0.000	-2.947	-1.088
Last Notable Activity_Modified	-1.5951	0.122	-13.037	0.000	-1.835	-1.355

Evaluate the model(Variance Inflation Factor)

- The model evaluation includes an assessment of the Variance Inflation Factor (VIF) to check for multicollinearity among the predictor variables
- VIF measures the degree of multicollinearity between a predictor and the other predictors in the model
- A VIF value of 1 indicates no multicollinearity, while values above 1 suggest increasing levels of multicollinearity.
- The model have an VIF values below 5 indicating suggesting that there is no substantial multicollinearity present

Variance Inflation Factor

Variance Inflation Factor (VIF):

	Features	VIF
0	const	4.361078
1	Total Time Spent on Website	1.139038
2	Lead Source_Welingak Website	1.048954
3	Last Activity_SMS Sent	1.135049
4	Tags_Closed by Horizzon	1.078293
5	Tags_Diploma holder (Not Eligible)	1.026801
6	Tags_Interested in other courses	1.101395
7	Tags_Lost to EINS	1.037088
8	Tags_Ringing	1.147463
9	Tags_Will revert after reading the email	1.342399
10	Tags_number not provided	1.004472
11	Tags_switched off	1.033616
12	Tags_wrong number given	1.007707
13	City_Select	1.083695
14	Asymmetrique Activity Index_03.Low	1.034579
15	Last Notable Activity_Modified	1.165223

Predicting the probabilities

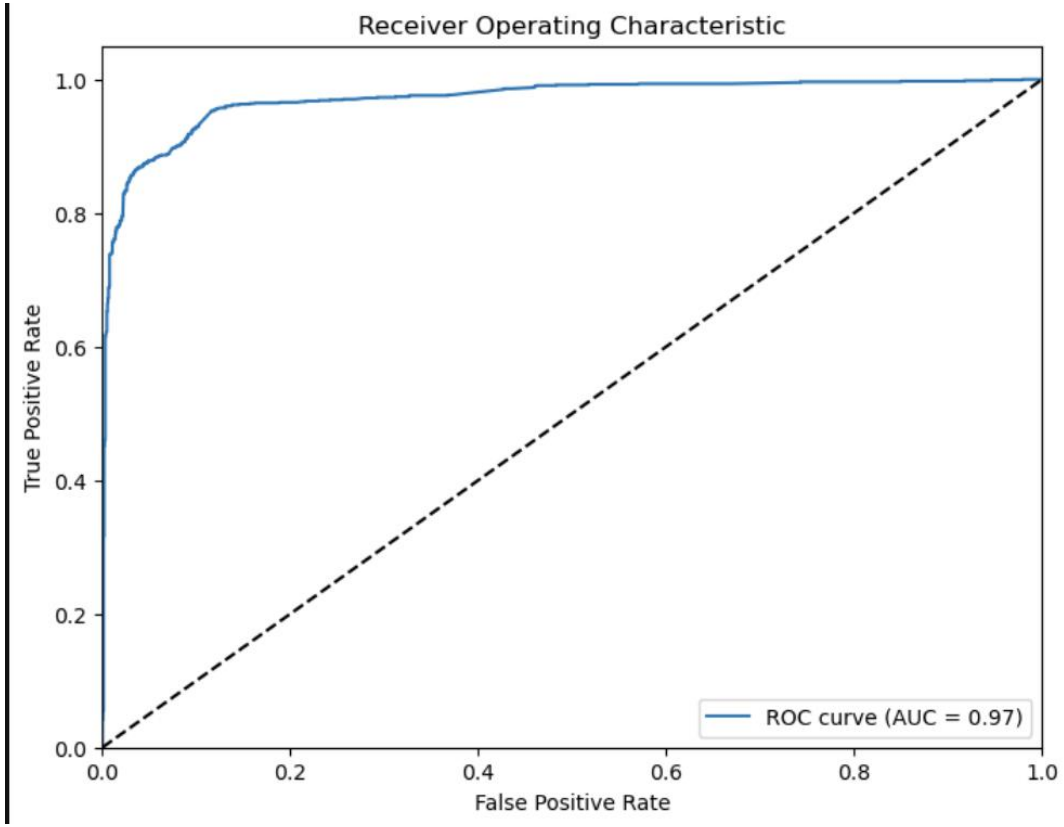
- y_train_pred = y_train_pred.values.reshape(-1) reshapes the predicted probabilities to a 1-dimensional array. This is necessary because the predicted probabilities are currently stored in a 2-dimensional array.
- y_train_pred_final = pd.DataFrame({'Converted':y_train.values, 'Conversion_Prob':y_train_pred}) creates a Pandas DataFrame that contains the predicted probabilities of churn and the actual churn values.
- y_train_pred_final['Predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 else 0) creates a new column in the DataFrame called Predicted and assigns it the values from the new Series.
- y_train_pred_final.head() prints the first 5 rows of the DataFrame

Use the model to predict the probability of leads converting

	Converted	Conversion_Prob	Predicted
0	0	0.059551	0
1	1	0.986605	1
2	0	0.036791	0
3	0	0.316595	0
4	0	0.518130	1

Model Performance

- The ROC curve is a good way to visualize the performance of a binary classification model. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR). The TPR is the percentage of instances that were correctly classified as positive, and the FPR is the percentage of instances that were incorrectly classified as positive



An AUC value of 0.97 is very good. It means that the model is very good at distinguishing between instances that will churn and instances that will not churn.

Thank You