National College of Ireland

Identifying Factors Contributing to Lead Conversion Using Machine Learning to Gain Business Insights

MSc Research Project
MSCDADJAN22

Mansi Sharma
Student ID: x21143315

School of Computing
National College of Ireland

Supervisor: Qurrat Ul Ain

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……Mansi Sharma……

Student ID: ……x21143315@student.ncirl.ie……..……

Programme:……Data Analytics…… **Year:** ……2022……..

Module: …… MSc. Research Project ……..……

Lecturer: …… Qurrat Ul Ain ……..……

Submission Due Date: ……01-02-2023……..……

Project Title: …… Identifying Factors Contributing to Lead Conversion Using Machine Learning to Gain Business Insights ……..……

Word Count: ……1356…… **Page Count:** ……13……..……..……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Mansi Sharma……

Date: ……29-01-2023……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Identifying Factors Contributing to Lead Conversion Using Machine Learning to Gain Business Insights

Mansi Sharma
Student ID: x21143315

1 Introduction

This document provides a concise overview of the steps taken in order to implement this research project. The aim of this project is to predict lead conversion using machine learning and to identify the most influential attributes in this process. The performance of machine learning models was evaluated and the best models were selected for calculating feature importance to generate visualizations.
This paper is split into the sections mentioned below: The hardware and software requirements for the implementation of this project are detailed in Section 2. Section 3 describes the environment setup for installing essential tools. The numerous execution processes for this project are detailed in Section 4. Finally, Section 5 shows the steps to view the Tableau workbook in the Tableau Desktop application.

2 Hardware and Software Requirements

All the necessary tools and software required for this research can be installed on any computer system with the following configurations:

Operating System	Windows 11
RAM	8 GB
Processor	Intel i5 10^{th} Gen
Hard Disk	512GB SSD

All the basic tools and software required for the implementation of this research are as below:

- Google Colab
- Tableau Desktop
- Microsoft Office Suite

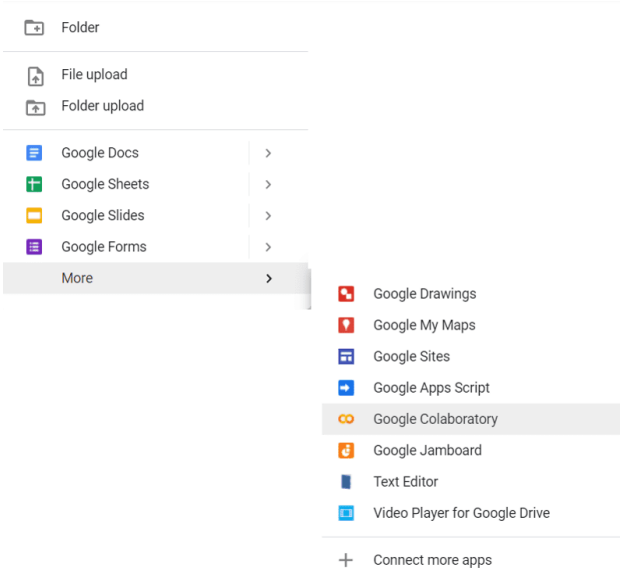
The code was written and executed in Google Colab which is a cloud bases framework. All it needs is a Google account and it can be installed on the cloud itself from Google drive. The advantage of Google Colab is that it is fast since it allows free access to Jupyter notebooks with Graphics Processing Units (GPU's)/ Tensor Processing Units (TPU's) and requires no

configuration or software installation prior to use. For creating visualization dashboards, Tableau Desktop (version 2022.2) has been used. It was installed in the local drive from the official website of Tableau. MS Word and MS Excel were used from the Microsoft Office Suite for creating reporting and viewing the CSV data respectively.

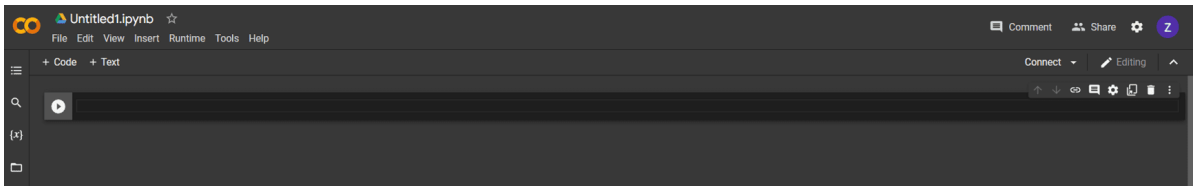
3 Installing Required Tools

3.1 Google Colab

1. To create Google Colab file and begin using Google Colab, visit Google Drive and establish a Google Drive account, if it is not already created.
2. Now, click the "New" button in the upper-left corner of Google Drive page, followed by More Google Collaboratory. If it's not listed, go to "Connect more apps" and install.

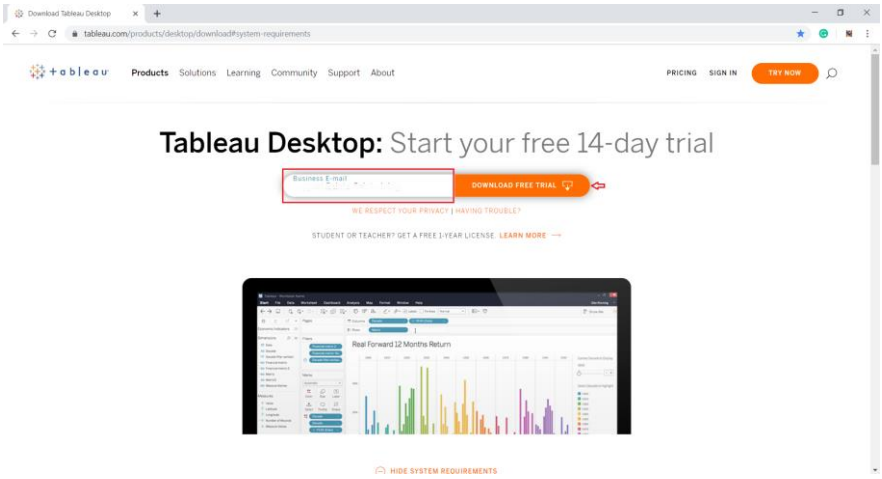


3. The website for your new Google Colab file will then load. From here, code can be written in the cells and can be executed. Files can also be shared across with other users. *(Chng, 2022)*

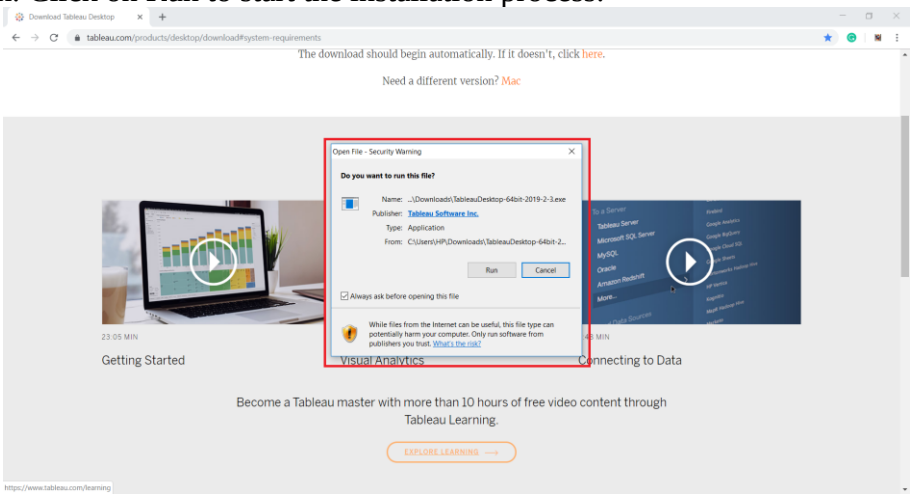


3.2 Tableau Desktop

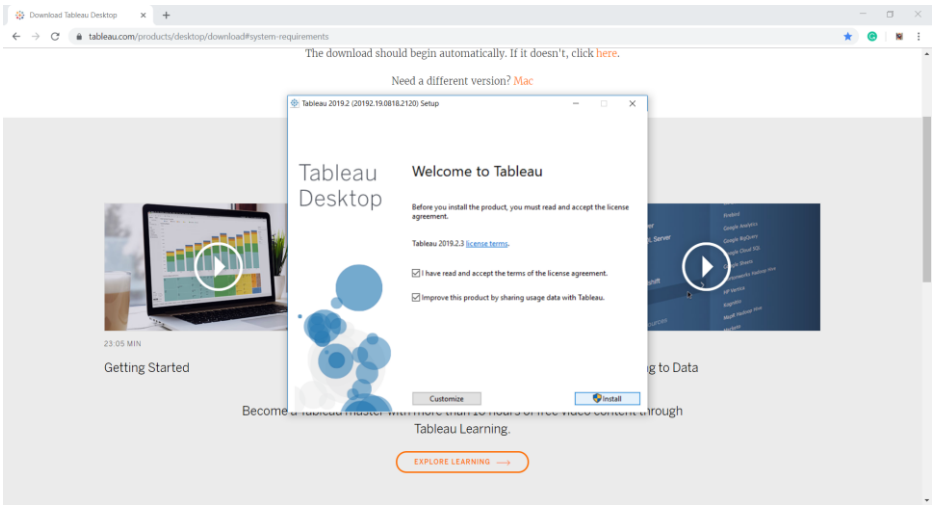
1. To download Tableau Desktop, we must first open a computer browser and navigate to the Tableau Website.
2. This will take us to the Tableau Desktop download page.
3. At the top of this page, you will be prompted to input your organization email address.



4. As soon as the download is completed, open the exe file. The install wizard will launch. Click on Run to start the installation process.



5. Check the box next to License conditions and click Install. This begins the installation procedure.



6. Next are three activation phases. In this box, pick Start trial immediately or Activate. Start trial now is a free 14-day trial, while Activate installs the commercial version with a product key.
7. Next, you'll be asked to register. Name, Email, Organization, Job Title, etc. Click Register after filling out the form.
8. After activation is complete, navigate to your desktop and double-click the Tableau icon to launch the application. *(Team, 2019)*

4 Code Execution

4.1 Importing required libraries

All the libraries required for the code are shown below.

```
import pandas as pd, numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score,recall_score, precision_score
import statsmodels.api as sm
import random
from scipy.stats import norm, skew
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import plot_roc_curve
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
from sklearn.model_selection import RandomizedSearchCV
!pip install catboost
import catboost as ctb
!pip install xgboost!
import xgboost as xgb
from xgboost import XGBClassifier
```

4.2 Importing DataSet

The data was in CSV format which was stored in the Google drive and was fetched from there. It was then changed into a dataframe as shown below. The cell will ask for permission when fetching the file from Google drive. Every user, while running the code, needs to first store the CSV file in their Google drive and change the file path to fetch it.

```
[3] from google.colab import drive
    drive.mount('/content/gdrive')
    drive.mount('/content/drive')

Mounted at /content/gdrive
Mounted at /content/drive
```

```
[4] leadcov=pd.read_csv('/content/drive/MyDrive/Lead Scoring.csv', on_bad_lines='skip')
```

```
[5] #leadcov = pd.read_csv("C:\\Users\\91823\\Desktop\\Research Project\\Lead Scoring Dataset\\Lead Scoring.csv")
    leadcov.head()
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	...	Get updates on DM Content	Lead Profile	City	Asymmetrique Activity Index	Asymmetrique Profile Index	Asymmetrique Activity Score
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	...	No	Select	Select	02.Medium	02.Medium	15.0
1	2a272436-5132-4136-			Organic													

4.3 Data Pre-Processing

In this section, all the steps taken from data cleaning to normalization will be shown.

4.3.1 Data Cleaning

1. The columns were checked for null values.

```
leadcov.isnull().sum()

Prospect ID                                       0
Lead Number                                       0
Lead Origin                                       0
Lead Source                                      36
Do Not Email                                      0
Do Not Call                                       0
Converted                                         0
TotalVisits                                     137
Total Time Spent on Website                       0
Page Views Per Visit                            137
Last Activity                                   103
Country                                        2461
Specialization                                 1438
How did you hear about X Education             2207
What is your current occupation                2690
What matters most to you in choosing a course  2709
Search                                            0
Magazine                                          0
Newspaper Article                                 0
X Education Forums                                0
Newspaper                                         0
Digital Advertisement                             0
Through Recommendations                           0
Receive More Updates About Our Courses            0
Tags                                           3353
```

2. Columns with more than 3000 null values were dropped so that model performance wouldn't get negatively impacted.

```
[15] #Removing columns with more than 3000 null values
     for x in leadcov.columns:
         if leadcov[x].isnull().sum() > 3000:
             leadcov=leadcov.drop(x, 1)
     leadcov.isnull().sum()

<ipython-input-15-9da4dd5be032>:4: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  leadcov=leadcov.drop(x, 1)
Prospect ID                                       0
Lead Number                                       0
Lead Origin                                       0
Lead Source                                      36
Do Not Email                                      0
Do Not Call                                       0
Converted                                         0
TotalVisits                                     137
Total Time Spent on Website                       0
Page Views Per Visit                            137
Last Activity                                   103
Country                                        2461
Specialization                                 1438
How did you hear about X Education             2207
What is your current occupation                2690
What matters most to you in choosing a course  2709
Search                                            0
Magazine                                          0
Newspaper Article                                 0
```

3. In order to treat missing values, mean and mode imputation was applied to the columns.

```
[18] #Checking mode value
     leadcov['Lead Source'].value_counts()
```

```
Google               2873
Direct Traffic       2543
Olark Chat           1755
Organic Search       1154
Reference             534
Welingak Website      142
Referral Sites        125
Social Media           58
Bing                    6
Click2call              4
Press_Release           2
Live Chat               2
Blog                    2
WeLearn                 1
Testone                 1
Pay per Click Ads       1
NC_EDM                  1
Name: Lead Source, dtype: int64
```

```
[19] #Imputing mode value
     leadcov['Lead Source']=leadcov['Lead Source'].fillna('Google')
     leadcov['Lead Source'].value_counts()
```

```
Google               2909
Direct Traffic       2543
```

4. Columns with more than 90% same values were removed as it would have affected the model performance.

```
[46] #Checking what percent of space is occupied by a single value as compared to others in columns
     categorical_columns = leadcov.select_dtypes(include=['object']).columns
     for x in leadcov[categorical_columns]:
         print(x,": ",round(int(list(leadcov[x].value_counts())[0])/int(sum(leadcov[x].value_counts()))*100,2),"%")
```

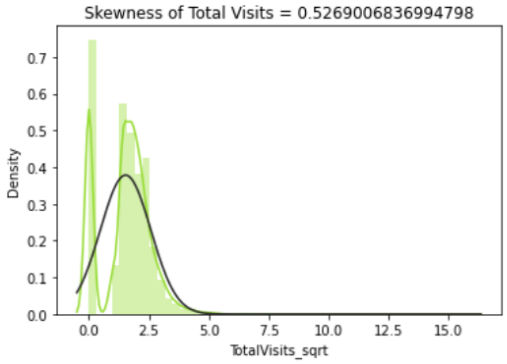
```
Prospect ID :  0.01 %
Lead Origin :  52.88 %
Lead Source :  31.48 %
Do Not Email :  92.06 %
Do Not Call :  99.98 %
Last Activity :  38.31 %
Country :  96.89 %
Specialization :  36.58 %
How did you hear about X Education :  78.46 %
What is your current occupation :  89.72 %
What matters most to you in choosing a course :  99.97 %
Search :  99.85 %
Magazine :  100.0 %
Newspaper Article :  99.98 %
X Education Forums :  99.99 %
Newspaper :  99.99 %
Digital Advertisement :  99.96 %
Through Recommendations :  99.92 %
Receive More Updates About Our Courses :  100.0 %
Update me on Supply Chain Content :  100.0 %
Get updates on DM Content :  100.0 %
```

4.3.2 Checking Skewness

Skewness for the numerical variables was checked and fixed.

```
[ ] leadcov['TotalVisits_sqrt'] = np.sqrt(leadcov['TotalVisits'])
    skewness = str(skew(leadcov['TotalVisits_sqrt']))
    sns.distplot(leadcov['TotalVisits_sqrt'],fit = norm,color = randomcolor())
    plt.title("Skewness of " + 'Total Visits'+ ' = '+ skewness)
    plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot`
  warnings.warn(msg, FutureWarning)
```

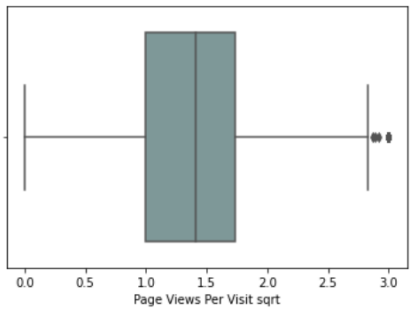


4.3.3 Outlier Analysis

Outliers were checked and removed for the numerical variables to enhance the model performance.

```
[ ] sns.boxplot(out_num_leadcov['Page Views Per Visit sqrt'],color=randomcolor())
    plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable
  warnings.warn(
```




```
[ ] #Creating new dataframe with variables free from outliers
    cleaned_leadcov=pd.concat([other_leadcov,out_num_leadcov],axis = 1)
    cleaned_leadcov.head()
```

4.3.4 One-Hot Encoding

For the model fitting, all the categorical variables were converted into numerical format using One-hot encoding.

```
[ ] #Removing 'Prospect ID' column from variables to create dummies.
    leadcov_2=cleaned_leadcov.drop("Prospect ID",axis=1).select_dtypes(include=['object'])
    leadcov_2.columns
```

```
Index(['Lead Origin', 'Lead Source', 'Last Activity', 'Specialization',
       'How did you hear about X Education', 'What is your current occupation',
       'Lead Profile', 'City', 'A free copy of Mastering The Interview',
       'Last Notable Activity'],
      dtype='object')
```

```
[ ] #Converting categorical variables into numerical form
    leadcov_num = pd.get_dummies(leadcov_2)
    leadcov_num.head()
```

	Lead Origin_API	Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Origin_Quick Add Form	Lead Source_Bing	Lead Source_Blog	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Google	...	Last Notable Activity_Form Submitted on Website	Last Activ Conv
0	1	0	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	0	0	0	...	0	
2	0	1	0	0	0	0	0	0	1	0	...	0	
3	0	1	0	0	0	0	0	0	1	0	...	0	

4.3.5 Normalization

Normalization has been applied to the variables to convert the values of numeric columns in the dataset to a similar scale without distorting range differences or losing data.

```
[ ] #Normalization using MinMaxScaler
    from sklearn import preprocessing
    scaler = preprocessing.MinMaxScaler()
    names = leadcov_num4.columns
    d = scaler.fit_transform(leadcov_num4)
    scaled_df = pd.DataFrame(d, columns=names)
    scaled_df.head()
```

	Total Time Spent on Website sqrt	Converted	Page Views Per Visit sqrt	TotalVisits_sqrt	Lead Origin_API	Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Origin_Quick Add Form	Lead Source_Bing	...	Last Notable Activity_Form Submitted on Website	Last Notable Activity_Had a Phone Conversation	Acti
0	0.000000	0.0	0.000000	0.000000	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
1	0.544660	0.0	0.527046	0.527046	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
2	0.821155	1.0	0.471405	0.333333	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
3	0.366392	0.0	0.333333	0.235702	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
4	0.792793	1.0	0.333333	0.333333	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	

5 rows × 110 columns

4.4 Train-Test Split

Data was distributed into train and test sets in a ratio of 7:3.

```
[211] Y = leadcov_num4.Converted
      X = leadcov_num4.drop("Converted",axis = 1)
```

```
[212] X_train, X_test,Y_train,Y_test = train_test_split(X, Y, test_size = 0.3, random_state = 42)
```

```
[213] X_train.shape
```

```
(6468, 109)
```

```
[214] X_test.shape
```

```
(2772, 109)
```

4.5 Logistic Regression

1. Calculating Logistic Regression coefficients after first fitting.

```
[485] #Finding features importance by calculating coefficients
      feature= pd.DataFrame()
      feature['Column']= X_train.columns
      feature['Coefficient_LR']= logit_model.coef_[0]
      feature.sort_values('Coefficient_LR', ascending=False, inplace=True)
      feature
```

	Column	Coefficient_LR
24	Lead Source_Welingak Website	2.243303
5	Lead Origin_Lead Add Form	2.095012
25	Last Activity_Approached upfront	1.909369
77	Lead Profile_Dual Specialization Student	1.522627
106	Last Notable Activity_Unreachable	1.369466
100	Last Notable Activity_Had a Phone Conversation	1.186306
76	What is your current occupation_Working Profes...	1.126254
80	Lead Profile_Potential Lead	1.118291
78	Lead Profile_Lateral Student	0.985894
72	What is your current occupation_Housewife	0.788216
37	Last Activity_SMS Sent	0.773689
33	Last Activity_Had a Phone Conversation	0.689837
15	Lead Source_Olark Chat	0.573539

2. Calculating VIF values to improve the model performance.

```
[492] vif = pd.DataFrame()
      vif['Features'] = X_train_new.columns
      vif['VIF'] = [variance_inflation_factor(X_train_new.values, i) for i in range(X_t
      vif['VIF'] = round(vif['VIF'], 2)
      vif = vif.sort_values(by = "VIF", ascending = False)
      vif
```

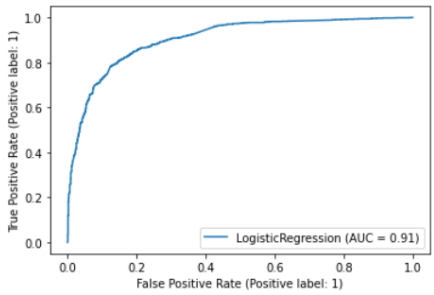
```
/usr/local/lib/python3.8/dist-packages/statsmodels/stats/outliers_influence.py:19
  vif = 1. / (1. - r_squared_i)
```

	Features	VIF
41	Last Activity_Email Marked Spam	inf
48	Last Notable Activity_Email Received	inf
47	Last Activity_Email Received	inf
42	Last Notable Activity_Email Marked Spam	inf
24	A free copy of Mastering The Interview_No	10.82
31	A free copy of Mastering The Interview_Yes	7.54
10	Last Activity_SMS Sent	4.09
14	Last Notable Activity_SMS Sent	3.63
26	TotalVisits_sqrt	2.91
12	Lead Source_Olark Chat	2.67
11	Last Activity_Had a Phone Conversation	2.04
5	Last Notable Activity_Had a Phone Conversation	2.02

3. Final fitting considering coefficients and VIF values.

```
[496] Logit=LogisticRegression(solver='liblinear')
      clf = Logit.fit(x_train_vif_adj,Y_train)
      Y_pred_lr1=Logit.predict(x_test_vif_adj)

      print(f"Accuracy of the Logistic Regression is: {accuracy_score(Y_test, Y_pred_lr1)}")
      print(f"Precision Score of Logistic Regression is: {precision_score(Y_test, Y_pred_lr1)}")
      print(f"Recall Score of Logistic Regression is: {recall_score(Y_test, Y_pred_lr1)}")
      print(f"F1 Score of Logistic Regression is: {f1_score(Y_test, Y_pred_lr1)}")
      plot_roc_curve(Logit, x_test_vif_adj, Y_test, drop_intermediate=False)
      plt.show()
```

```
Accuracy of the Logistic Regression is: 0.8326118326118326
Precision Score of Logistic Regression is: 0.8229715489989463
Recall Score of Logistic Regression is: 0.7251624883936861
F1 Score of Logistic Regression is: 0.7709772951628825
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Functi
  warnings.warn(msg, category=FutureWarning)
```



4.6 Decision Tree

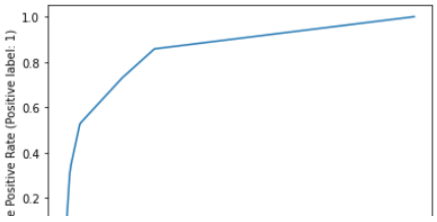
Applying decision tree to the original train and test sets and evaluating the model.

```
[226] dt = DecisionTreeClassifier(random_state=42, max_depth=3, min_samples_leaf=10)

[227] dt.fit(X_train, Y_train)
```

```
DecisionTreeClassifier(max_depth=3, min_samples_leaf=10, random_state=42)
```

```
[228] y_test_pred_dt = dt.predict(X_test)

[229] print(f"Accuracy of the Decision Tree is: {accuracy_score(Y_test,y_test_pred_dt)}")
      print(f"Precision Score of Decision Tree is: {precision_score(Y_test,y_test_pred_dt)}")
      print(f"Recall Score of Decision Tree is: {recall_score(Y_test,y_test_pred_dt)}")
      print(f"F1 Score of Decision Tree is: {f1_score(Y_test,y_test_pred_dt)}")
      plot_roc_curve(dt, X_test, Y_test, drop_intermediate=False)
      plt.show()
```

```
Accuracy of the Decision Tree is: 0.7911255411255411
Precision Score of Decision Tree is: 0.8854489164086687
Recall Score of Decision Tree is: 0.531104921077066
F1 Score of Decision Tree is: 0.6639582124201974
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning:
  warnings.warn(msg, category=FutureWarning)
```



4.7 Random Forest

1. Applying random forest to the original train and test sets and evaluating the model.

```
[230] # Instantiate and fit the RandomForestClassifier
      forest = RandomForestClassifier()
      forest.fit(X_train, Y_train)

      RandomForestClassifier()
```

```
[231] # Make predictions for the test set
      y_pred_test_rf = forest.predict(X_test)
```

```
[232] print(f"Accuracy of the Random Forest Classifier is: {accuracy_score(Y_test, y_pred_test_rf)}")
      print(f"Precision Score of Random Forest Classifier is: {precision_score(Y_test, y_pred_test_rf)}")
      print(f"Recall Score of Random Forest Classifier is: {recall_score(Y_test, y_pred_test_rf)}")
      print(f"F1 Score of Random Forest Classifier is: {f1_score(Y_test, y_pred_test_rf)}")

      Accuracy of the Random Forest Classifier is: 0.8398268398268398
      Precision Score of Random Forest Classifier is: 0.8063891577928364
      Recall Score of Random Forest Classifier is: 0.7734447539461468
      F1 Score of Random Forest Classifier is: 0.7895734597156399
```

2. Hyperparameter Tuning to enhance model performance.

```
[ ] n_estimators = [5,20,50,100] # number of trees in the random forest
    max_features = ['auto', 'sqrt'] # number of features in consideration at every split
    max_depth = [int(x) for x in np.linspace(10, 120, num = 12)] # maximum number of levels allowed in each decision tree
    min_samples_split = [2, 6, 10] # minimum sample number to split a node
    min_samples_leaf = [1, 3, 4] # minimum sample number that can be stored in a leaf node
    bootstrap = [True, False] # method used to sample data points

    random_grid = {'n_estimators': n_estimators,

    'max_features': max_features,

    'max_depth': max_depth,

    'min_samples_split': min_samples_split,

    'min_samples_leaf': min_samples_leaf,

    'bootstrap': bootstrap}
```

```
[ ] rf = RandomForestClassifier()
```

```
[ ] rf_random = RandomizedSearchCV(estimator = rf,param_distributions = random_grid,
                   n_iter = 100, cv = 5, verbose=2, random_state=35, n_jobs = -1)
```

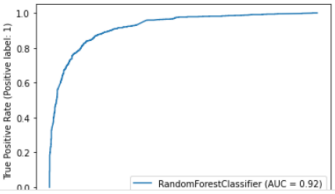
```
[ ] rf_random.fit(X_train, Y_train)

    Fitting 5 folds for each of 100 candidates, totalling 500 fits
    RandomizedSearchCV(cv=5, estimator=RandomForestClassifier(), n_iter=100,
                       n_jobs=-1,
                       param_distributions={'bootstrap': [True, False],
                                            'max_depth': [10, 20, 30, 40, 50, 60,
                                                          70, 80, 90, 100, 110,
```

3. Evaluating the model performance after hyperparameter tuning.

```
[ ] #Substituting the hyperparamerers and fitting the model
    randmf = RandomForestClassifier(n_estimators = 100, min_samples_split = 10, min_samples_leaf= 4, max_features = 'auto', max_depth= 50, bootstrap=True)
    randmf.fit( X_train, Y_train)

    RandomForestClassifier(max_depth=50, min_samples_leaf=4, min_samples_split=10)
```

```
[ ] y_pred_test_rf1 = randmf.predict(X_test)
```

```
[ ] print(f"Accuracy of the Random Forest is: {accuracy_score(Y_test,y_pred_test_rf1)}")
    print(f"Precision Score of Random Forest is: {precision_score(Y_test,y_pred_test_rf1)}")
    print(f"Recall Score of Random Forest is: {recall_score(Y_test,y_pred_test_rf1)}")
    print(f"F1 Score of Random Forest is: {f1_score(Y_test,y_pred_test_rf1)}")
    plot_roc_curve(randmf, X_test, Y_test, drop_intermediate=False)
    plt.show()

    Accuracy of the Random Forest is: 0.8528138528138528
    Precision Score of Random Forest is: 0.849529780564263
    Recall Score of Random Forest is: 0.754874651810585
    F1 Score of Random Forest is: 0.7994100294985251
    /usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_roc_curve is deprecated; Function :func:`plot_roc_
      warnings.warn(msg, category=FutureWarning)
```



4. Calculating Feature Importance of the random forest model.

```
[ ] importances = randmf.feature_importances_
    pd.options.display.float_format = '{:.10f}'.format
    feature_rf= pd.DataFrame()
    feature_rf['Column']= X_train.columns
    feature_rf['Feature_Importance_RF']= importances
    feature_rf.sort_values('Feature_Importance_RF', ascending=False, inplace=True)
    pd.set_option('display.max_rows', None)
    feature_rf
```

	Column	Feature_Importance_RF
0	Total Time Spent on Website sqrt	0.2298029640
80	Lead Profile_Potential Lead	0.0958590443
105	Last Notable Activity_SMS Sent	0.0775091070
37	Last Activity_SMS Sent	0.0711432890
82	Lead Profile_Unknown	0.0569541464
76	What is your current occupation_Working Profes...	0.0493173063
5	Lead Origin_Lead Add Form	0.0473441068
75	What is your current occupation_Unemployed	0.0333023905
19	Lead Source_Reference	0.0329138920
2	TotalVisits_sqrt	0.0310192486
101	Last Notable Activity_Modified	0.0268508335
1	Page Views Per Visit sqrt	0.0268397987
60	Specialization_Unknown	0.0175147490

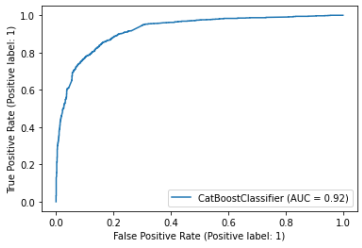
4.8 CatBoost Classifier

1. Applying Catboost classifier to the original train and test sets and evaluating the model.

```
[372] expected_y  = Y_test
      predicted_y = model_CBC.predict(X_test)
```

```
[373] print(f"Accuracy of the CatBoost Classifier is: {accuracy_score(expected_y, predicted_y)}")
      print(f"Precision Score of CatBoost Classifier is: {precision_score(expected_y, predicted_y)}")
      print(f"Recall Score of CatBoost Classifier is: {recall_score(expected_y, predicted_y)}")
      print(f"F1 Score of CatBoost Classifier is: {f1_score(expected_y, predicted_y)}")
      plot_roc_curve(model_CBC, X_test, Y_test, drop_intermediate=False)
      plt.show()
```

```
Accuracy of the CatBoost Classifier is: 0.8502886002886003
Precision Score of CatBoost Classifier is: 0.8277227722772277
Recall Score of CatBoost Classifier is: 0.776230269266481
F1 Score of CatBoost Classifier is: 0.8011499760421658
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function
  warnings.warn(msg, category=FutureWarning)
```



2. Calculating Feature Importance of the Catboost classifier model.

```
[374] # Compute feature importance dataframe
      feat_imp_list = list(zip (model_CBC.feature_names_, list(model_CBC.feature_importances_)) )
      feature_imp_df = pd.DataFrame(sorted(feat_imp_list, key=lambda x: x[0], reverse=True) , columns = ['Column','Importance_CB'])
      pd.set_option('display.max_rows', None)
      feature_imp_df
```

	Column	Importance_CB
0	What is your current occupation_Working Profes...	5.6672497659
1	What is your current occupation_Unemployed	2.2487410534
2	What is your current occupation_Student	0.1273797837
3	What is your current occupation_Other	0.0238751259
4	What is your current occupation_Housewife	0.0340460483
5	What is your current occupation_Businessman	0.0015160362
6	TotalVisits_sqrt	5.7696057703
7	Total Time Spent on Website sqrt	21.0562658616
8	Specialization_Unknown	1.8400636395
9	Specialization_Travel and Tourism	0.1824327044
10	Specialization_Supply Chain Management	0.2076446787
11	Specialization_Services Excellence	0.0074975652
12	Specialization_Rural and Agribusiness	0.0815927601
13	Specialization_Retail Management	0.1304458214
14	Specialization_Operations Management	0.5650361911
15	Specialization_Media and Advertising	0.4165479742
16	Specialization_Marketing Management	0.7068485841
17	Specialization_International Business	0.2244000157

4.9 XGBoost Classifier

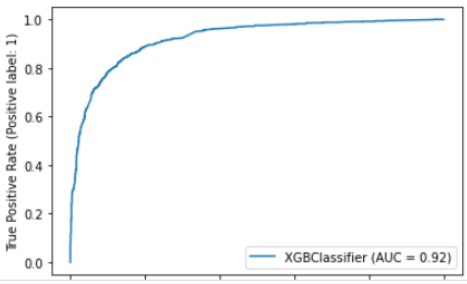
1. Applying XGBoost classifier to the train and test sets and evaluating the model.

```
[ ] # fit the model with the training data
    modelXGB.fit(X_train,Y_train)

    XGBClassifier()
```

```
[ ] predict_test_xg = modelXGB.predict(X_test)
```

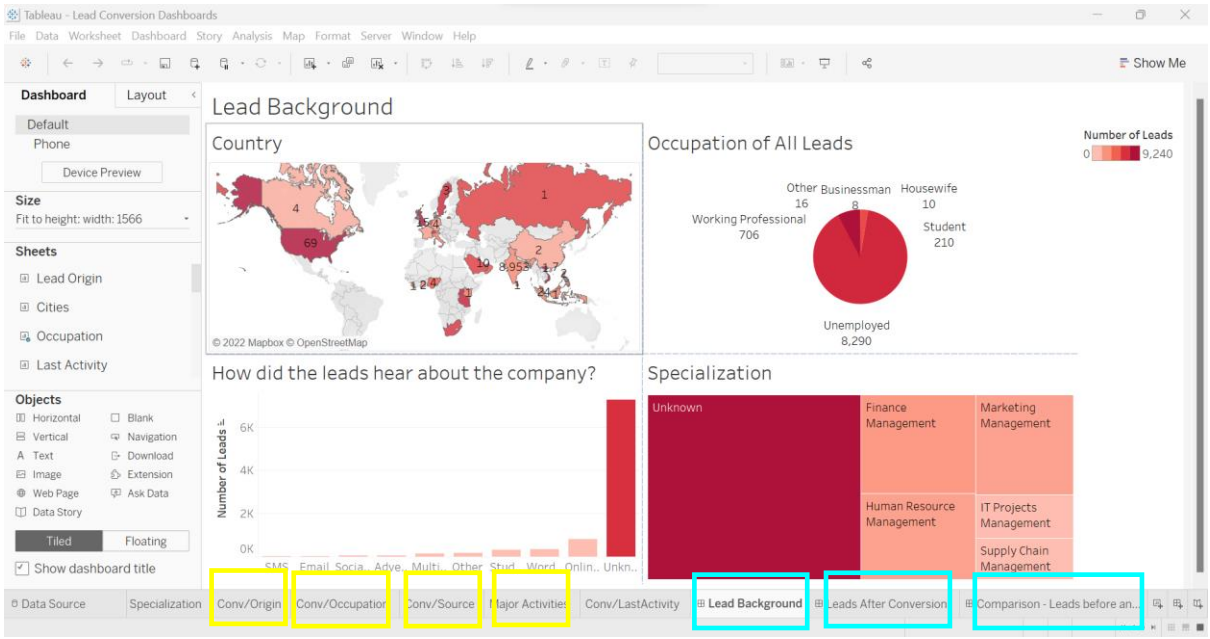
```
[ ] print(f"Accuracy of the XGBoost Classifier is: {accuracy_score(Y_test,predict_test_xg)}")
    print(f"Precision Score of XGBoost Classifier is: {precision_score(Y_test,predict_test_xg)}")
    print(f"Recall Score of XGBoost Classifier is: {recall_score(Y_test,predict_test_xg)}")
    print(f"F1 Score of XGBoost Classifier is: {f1_score(Y_test,predict_test_xg)}")
    plot_roc_curve(modelXGB, X_test, Y_test, drop_intermediate=False)
    plt.show()
```

```
Accuracy of the XGBoost Classifier is: 0.8517316017316018
Precision Score of XGBoost Classifier is: 0.8377281947261663
Recall Score of XGBoost Classifier is: 0.7669452181987001
F1 Score of XGBoost Classifier is: 0.8007755695588948
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Functic
  warnings.warn(msg, category=FutureWarning)
```



5 Tableau Dashboards

To view the Tableau dashboards, simply open the Tableau Workbook. On the bottom side of the window, several sheets and dashboards are there. The figure below depicts different dashboards created in Tableau. Tabs highlighted in yellow are the sheets (individual visualizations of variables) while tabs highlighted in blue are the dashboards.



References

Chng, Z. M. (2022, April 27). Google Colab for Machine Learning Projects. MachineLearningMastery.Com. https://machinelearningmastery.com/google-colab-for-machine-learning-projects/

Team, D. (2019, November 11). Tableau Installation Process—The easy way to set up Tableau. DataFlair. https://data-flair.training/blogs/tableau-installation/