TOPIC : LEAD SCORING CASE STUDY

ABSTRACT:

Although lead scoring is an essential component of lead management, there is a lack of a comprehensive literature review and a classifcation framework dedicated to it. Lead scoring is an efective and efcient way of measuring the quality of leads. In addition, as a critical Information Technology tool, a proper lead scoring model acts as an alleviator to weaken the conficts between sales and marketing functions. Yet, little is known regarding lead scoring models and their impact on sales performance. Lead scoring models are commonly categorized into two classes: traditional and predictive. While the former primarily relies on the experience and knowledge of salespeople and marketers, the latter utilizes data mining models and machine learning algorithms to support the scoring process. This study aims to review and analyze the existing literature on lead scoring models and their impact on sales performance. A systematic literature review was conducted to examine lead scoring models. A total of 44 studies have met the criteria and were included for analysis. Fourteen metrics were identifed to measure the impact of lead scoring models on sales performance. With the increased use of data mining and machine learning techniques in the fourth industrial revolution, predictive lead scoring models are expected to replace traditional lead scoring models as they positively impact sales performance. Despite the relative cost of implementing and maintaining predictive lead scoring models, it is still benefcial to supersede traditional lead scoring models, given the higher efectiveness and efciency of predictive lead scoring models. This study reveals that classifcation is the most popular data mining model, while decision tree and logistic regression are the most applied algorithms among all the predictive lead scoring models. This study contributes by systematizing and recommending which machine learning method (i.e., supervised and/or unsupervised) shall be used to build predictive lead scoring models based on the integrity of diferent types of data sources.Additionally, this study ofers both theoretical and practical research directions in the lead scoring feld.

Keywords Lead scoring model · Sales performance · Data mining model · Machine learning algorithm · Systematic literature review

OBJECTIVE:

- X education wants to know most promising leads.
- For that they want to build a Model which identifies the hot leads.
- Deployment of the model for the future use

INTRODUCTION:

A lead is an essential raw material for sales organizations.Leads, being members of a target market segment, intentionally or unintentionally signal an interest in a company’s product(s)/service(s), regardless of whether that particular interest comes from a new prospect or an existing customer. Companies invest significantly in advertisements, web campaigns, and marketing to generate new leads and allocate enormous resources to nurture and convert these leads into customers. Conventional, outside sales (also called field sales) that are primarily based on in-person interactions with leads have been giving up the leading role to inside sales that mainly rely on remote sales conducted with the help of information and communication technologies (ICT) (e.g., phone, Internet). For some industries, inside sales became dominant and sometimes the only way to sell their products and services. The increasing cost of conventional sales, as well as advances in information technology (IT) tools and buyers’ higher demands and expectations, have contributed to the rapid growth of inside sales. For the last two decades, we have observed a significant shift from conventional field sales to the dominating inside sales enabled by ICT. The current COVID-19 pandemic forced many organizations to reduce costs and eliminate unnecessary spending. For this reason, it has become increasingly essential for organizations to maximize opportunities from new prospects and existing customers by taking advantage of inside sales. Lead Management System (LMS), an integrated information system of inside sales, became the“driving force”for operations with leads. LMS uses various IT tools to streamline and automate complicated lead management processes, for example, lead generation, lead nurturing, lead distribution, and lead scoring. However,not only the way of selling (i.e., traditional vs. ICT-enabled inside sales) has evolved during the last decades, but inside sales have further benefited by shifting from list-based (manually prioritizing and filtering of leads based on sales representatives’ knowledge and experience) to queue-based LMSs (an approach for prioritizing leads when the most promising leads are served first). The increased productivity,more efficient management control, and quicker response to leads have made queue-based LMSs the best solution for managing leads in inside sales. Lead scoring has been widely acknowledged as the most effective and efficient way of qualifying the quality of a large number of leads for queue-based LMSs. Lead scoring modeling is at the core of lead scoring, a qualifcation approach that assesses the leads'likelihood of making a purchase by ranking them against a scale to differentiate and prioritize them by generating a queue-based list for sales. A high-quality lead scoring model with superior predictive power could convince salespeople to contact more market-qualifed leads (MQLs) and convert those"ready-to-buy" leads to customers in a short time. From a long-term perspective, having a high-quality lead-scoring model can also improve the internal collaboration between the marketing and sales functions.

METHODOLOGY:

◆ Data cleaning and data manipulation.

1. Check and handle duplicate data.

2. Check and handle NA values and missing values.

3. Drop columns, if it contains large amount of missing values and not useful for the analysis.

4. Imputation of the values, if necessary.

5. Check and handle outliers in data.

◆ EDA

1. Univariate data analysis: value count, distribution of variable etc.

2. Bivariate data analysis: correlation coefficients and pattern between the variables etc.

◆ Feature Scaling & Dummy Variables and encoding of the data.

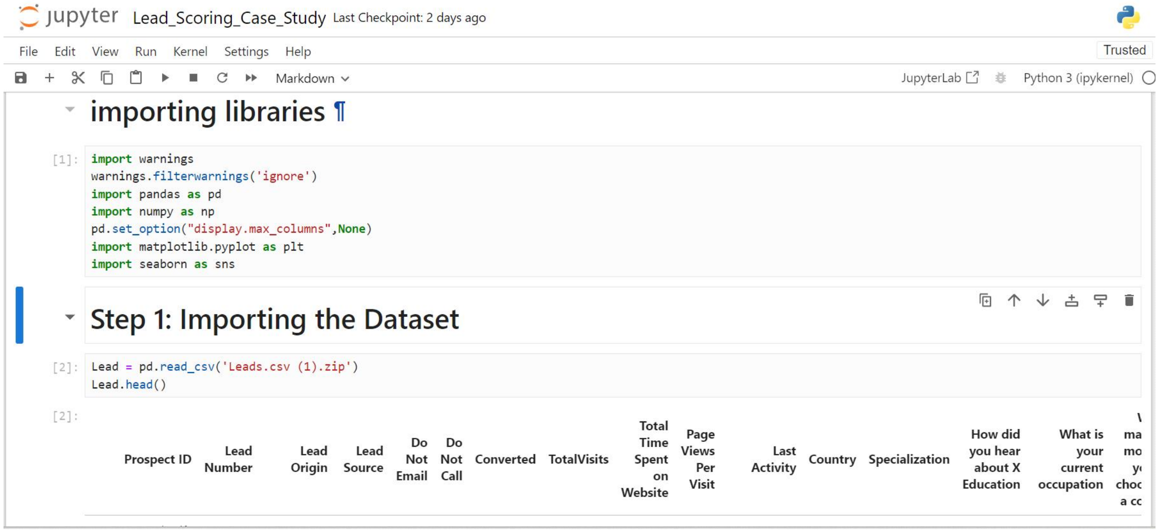
◆ Classification technique: logistic regression used for the model making and prediction.

◆ Validation of the model.

◆ Model presentation.

◆ Conclusions and recommendations

CODE:



jupyter Lead_Scoring_Case_Study Last Checkpoint: 2 days ago
File Edit View Run Kernel Settings Help
Trusted
Markdown
JupyterLab
Python 3 (ipykernel)

importing libraries ¶

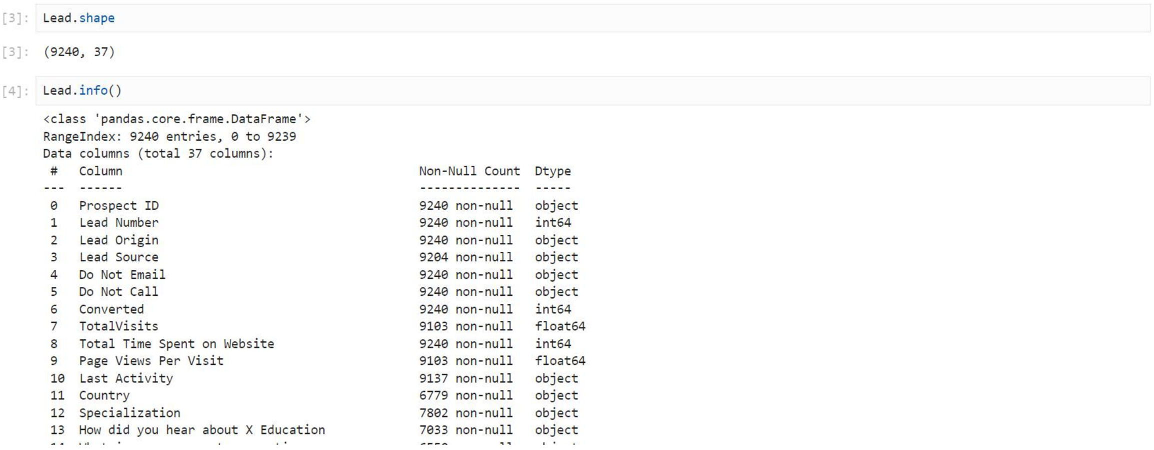
```
[1]: import warnings
     warnings.filterwarnings('ignore')
     import pandas as pd
     import numpy as np
     pd.set_option("display.max_columns",None)
     import matplotlib.pyplot as plt
     import seaborn as sns
```

Step 1: Importing the Dataset

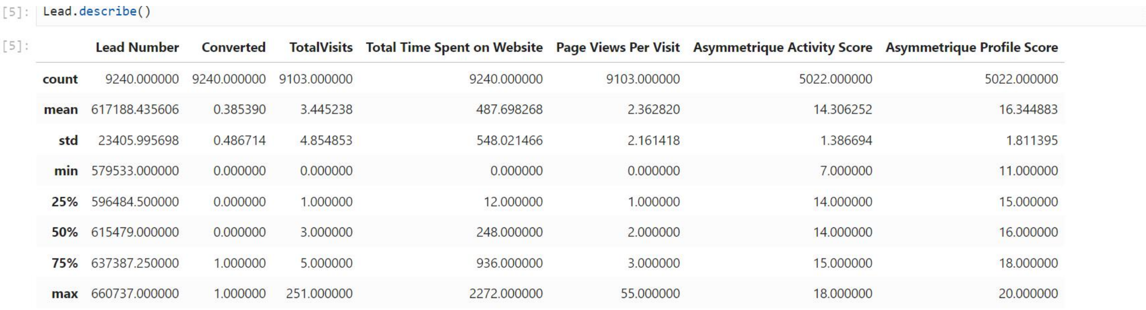
```
[2]: Lead = pd.read_csv('Leads.csv (1).zip')
     Lead.head()
```

[2]:

	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	Last Activity	Country	Specialization	How did you hear about X Education	What is your current occupation	\ ma mo y choc a cc



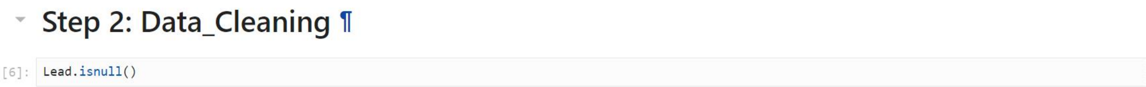
```
[3]: Lead.shape

[3]: (9240, 37)

[4]: Lead.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 9240 entries, 0 to 9239
     Data columns (total 37 columns):
      #   Column                                         Non-Null Count  Dtype  
     ---  ------                                         --------------  -----  
      0   Prospect ID                                    9240 non-null   object 
      1   Lead Number                                    9240 non-null   int64  
      2   Lead Origin                                    9240 non-null   object 
      3   Lead Source                                    9204 non-null   object 
      4   Do Not Email                                   9240 non-null   object 
      5   Do Not Call                                    9240 non-null   object 
      6   Converted                                      9240 non-null   int64  
      7   TotalVisits                                    9103 non-null   float64
      8   Total Time Spent on Website                    9240 non-null   int64  
      9   Page Views Per Visit                           9103 non-null   float64
      10  Last Activity                                  9137 non-null   object 
      11  Country                                        6779 non-null   object 
      12  Specialization                                 7802 non-null   object 
      13  How did you hear about X Education             7033 non-null   object 
```



```
[5]: Lead.describe()
```

[5]:

	Lead Number	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Asymmetrique Activity Score	Asymmetrique Profile Score
count	9240.000000	9240.000000	9103.000000	9240.000000	9103.000000	5022.000000	5022.000000
mean	617188.435606	0.385390	3.445238	487.698268	2.362820	14.306252	16.344883
std	23405.995698	0.486714	4.854853	548.021466	2.161418	1.386694	1.811395
min	579533.000000	0.000000	0.000000	0.000000	0.000000	7.000000	11.000000
25%	596484.500000	0.000000	1.000000	12.000000	1.000000	14.000000	15.000000
50%	615479.000000	0.000000	3.000000	248.000000	2.000000	14.000000	16.000000
75%	637387.250000	1.000000	5.000000	936.000000	3.000000	15.000000	18.000000
max	660737.000000	1.000000	251.000000	2272.000000	55.000000	18.000000	20.000000



Step 2: Data_Cleaning ¶

```
[6]: Lead.isnull()
```

```
[7]: Lead.isnull().mean()*100
```

```
[7]: Prospect ID                                      0.000000
     Lead Number                                      0.000000
     Lead Origin                                      0.000000
     Lead Source                                      0.389610
     Do Not Email                                     0.000000
     Do Not Call                                      0.000000
     Converted                                        0.000000
     TotalVisits                                      1.482684
     Total Time Spent on Website                      0.000000
     Page Views Per Visit                             1.482684
     Last Activity                                    1.114719
     Country                                         26.634199
     Specialization                                  15.562771
     How did you hear about X Education              23.885281
     What is your current occupation                 29.112554
     What matters most to you in choosing a course   29.318182
     Search                                           0.000000
     Magazine                                         0.000000
     Newspaper Article                                0.000000
     X Education Forums                               0.000000
     Newspaper                                        0.000000
     Digital Advertisement                            0.000000
     Through Recommendations                          0.000000
     Receive More Updates About Our Courses           0.000000
```

```
[8]: Lead.drop(columns=["Prospect ID","Lead Number"], axis = 1, inplace = True)
```

```
[9]: cat_cols = list(Lead.select_dtypes(include='object').columns)
     cat_cols
```

```
[9]: ['Lead Origin',
      'Lead Source',
      'Do Not Email',
      'Do Not Call',
      'Last Activity',
      'Country',
      'Specialization',
      'How did you hear about X Education',
      'What is your current occupation',
      'What matters most to you in choosing a course',
      'Search',
      'Magazine',
      'Newspaper Article',
      'X Education Forums',
      'Newspaper',
      'Digital Advertisement',
      'Through Recommendations',
      'Receive More Updates About Our Courses',
      'Tags',
```

```
[10]: for col in cat_cols:
          print(col, ":",Lead[col].value_counts())
          print("\n\n\t-------------------------\n\n")
```

```
Lead Source : Lead Source
Google               2868
Direct Traffic       2543
Olark Chat           1755
Organic Search       1154
Reference             534
Welingak Website      142
Referral Sites        125
Facebook               55
bing                    6
google                  5
Click2call              4
Press_Release           2
Social Media            2
Live Chat               2
youtubechannel          1
testone                 1
```

```
[11]: sel_cols = ["How did you hear about X Education","Lead Profile","City","Specialization"]
```

```
[12]: Lead[sel_cols] = Lead[sel_cols].replace("Select", np.nan,inplace = True)
```

```
[13]: Lead.isnull().mean()*100
```

```
[13]: Lead Origin                                        0.000000
      Lead Source                                        0.389610
      Do Not Email                                       0.000000
      Do Not Call                                        0.000000
      Converted                                          0.000000
      TotalVisits                                        1.482684
      Total Time Spent on Website                        0.000000
      Page Views Per Visit                               1.482684
      Last Activity                                      1.114719
      Country                                           26.634199
      Specialization                                   100.000000
      How did you hear about X Education               100.000000
      What is your current occupation                   29.112554
      What matters most to you in choosing a course     29.318182
      Search                                             0.000000
      Magazine                                           0.000000
      Newspaper Article                                  0.000000
      X Education Forums                                 0.000000
      Newspaper                                          0.000000
      Digital Advertisement                              0.000000
      Through Recommendations                            0.000000
```

```
[14]: cols = Lead.columns
      cols
```

```
[14]: Index(['Lead Origin', 'Lead Source', 'Do Not Email', 'Do Not Call',
             'Converted', 'TotalVisits', 'Total Time Spent on Website',
             'Page Views Per Visit', 'Last Activity', 'Country', 'Specialization',
             'How did you hear about X Education', 'What is your current occupation',
             'What matters most to you in choosing a course', 'Search', 'Magazine',
             'Newspaper Article', 'X Education Forums', 'Newspaper',
             'Digital Advertisement', 'Through Recommendations',
             'Receive More Updates About Our Courses', 'Tags', 'Lead Quality',
             'Update me on Supply Chain Content', 'Get updates on DM Content',
             'Lead Profile', 'City', 'Asymmetrique Activity Index',
             'Asymmetrique Profile Index', 'Asymmetrique Activity Score',
             'Asymmetrique Profile Score',
             'I agree to pay the amount through cheque',
             'A free copy of Mastering The Interview', 'Last Notable Activity'],
            dtype='object')
```

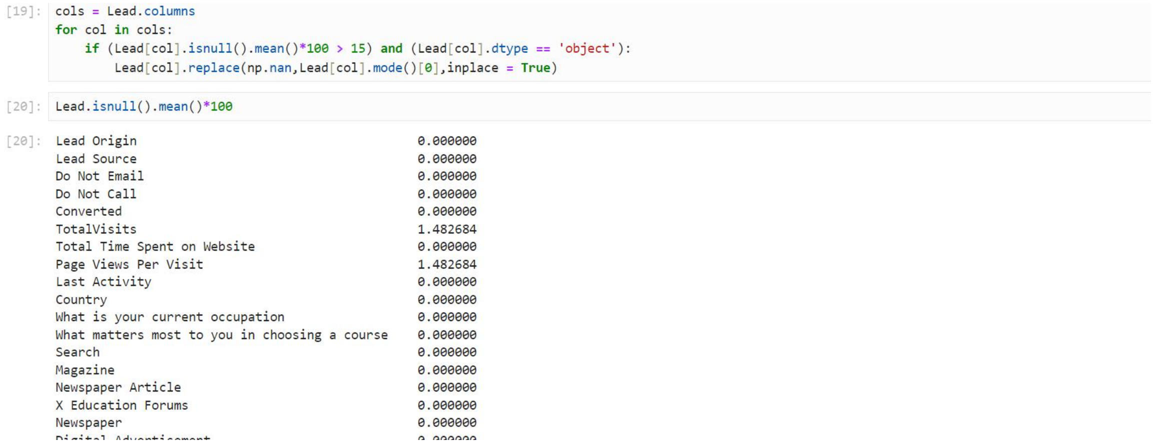
```
[15]: for col in cols:
          if Lead[col].isnull().mean()*100 > 40:
              Lead.drop(col,axis = 1, inplace = True)
```

```
[16]: Lead.isnull().mean()*100
```

```
[17]: cols = Lead.columns
      for col in cols:
          if (Lead[col].isnull().mean()*100 < 15) and (Lead[col].dtype == 'object'):
              Lead[col].replace(np.nan,Lead[col].mode()[0],inplace = True)
```

```
[18]: Lead.isnull().mean()*100
```

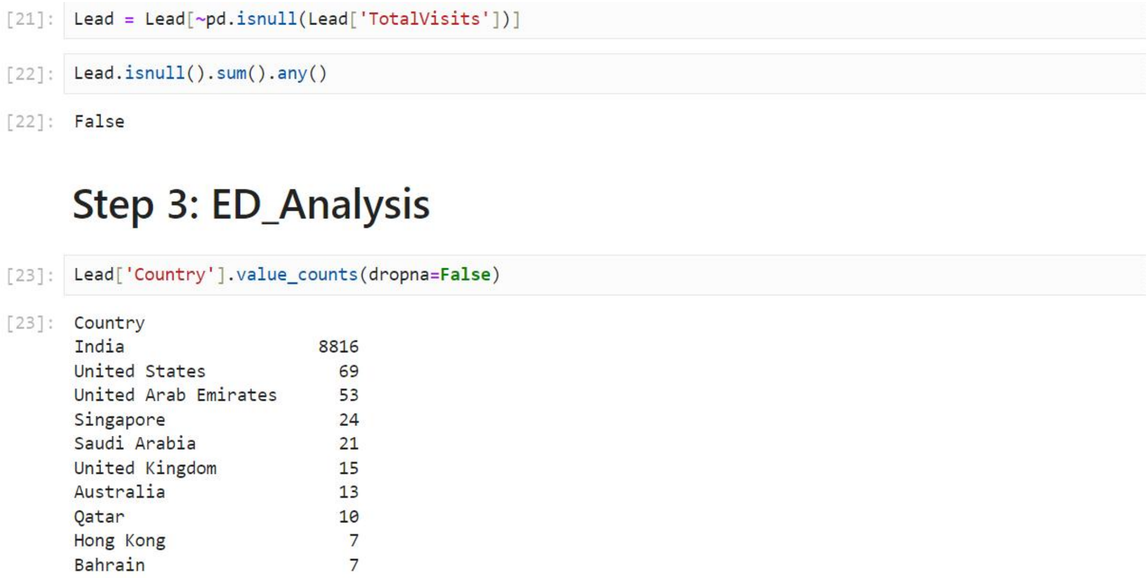
```
[18]: Lead Origin                                        0.000000
      Lead Source                                        0.000000
      Do Not Email                                       0.000000
      Do Not Call                                        0.000000
      Converted                                          0.000000
      TotalVisits                                        1.482684
      Total Time Spent on Website                        0.000000
      Page Views Per Visit                               1.482684
      Last Activity                                      0.000000
      Country                                           26.634199
      What is your current occupation                   29.112554
      What matters most to you in choosing a course     29.318182
      Search                                             0.000000
      Magazine                                           0.000000
      Newspaper Article                                  0.000000
      X Education Forums                                 0.000000
      Newspaper                                          0.000000
      Digital Advertisement                              0.000000
      Through Recommendations                            0.000000
```



```
[19]: cols = Lead.columns
      for col in cols:
          if (Lead[col].isnull().mean()*100 > 15) and (Lead[col].dtype == 'object'):
              Lead[col].replace(np.nan,Lead[col].mode()[0],inplace = True)
```

```
[20]: Lead.isnull().mean()*100
```

```
[20]: Lead Origin                                      0.000000
      Lead Source                                      0.000000
      Do Not Email                                     0.000000
      Do Not Call                                      0.000000
      Converted                                        0.000000
      TotalVisits                                      1.482684
      Total Time Spent on Website                      0.000000
      Page Views Per Visit                             1.482684
      Last Activity                                    0.000000
      Country                                          0.000000
      What is your current occupation                  0.000000
      What matters most to you in choosing a course    0.000000
      Search                                           0.000000
      Magazine                                         0.000000
      Newspaper Article                                0.000000
      X Education Forums                               0.000000
      Newspaper                                        0.000000
      Digital Advertisement                            0.000000
```



```
[21]: Lead = Lead[~pd.isnull(Lead['TotalVisits'])]
```

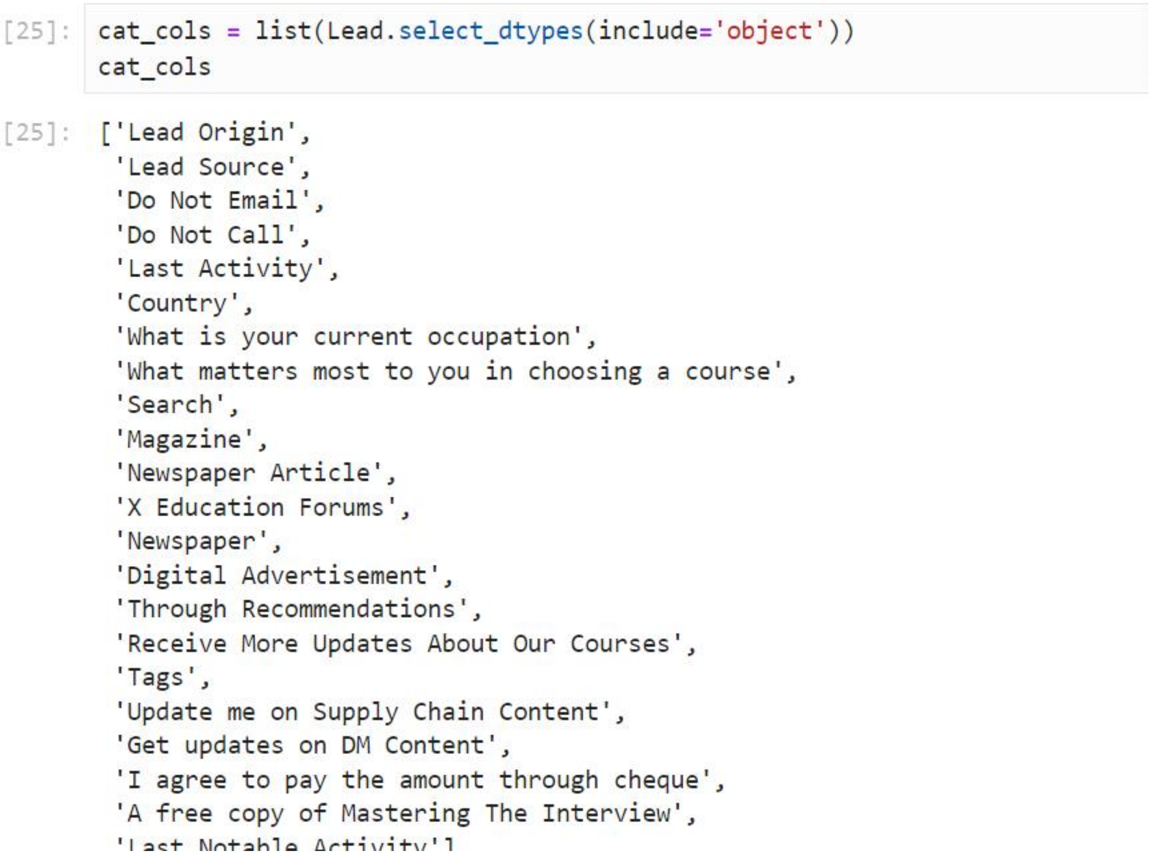
```
[22]: Lead.isnull().sum().any()
```

```
[22]: False
```

Step 3: ED_Analysis

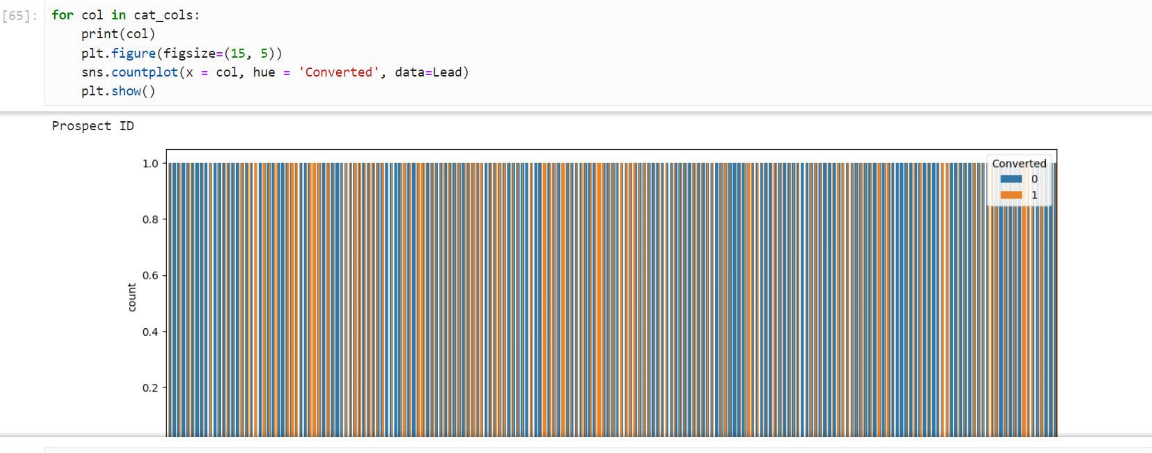
```
[23]: Lead['Country'].value_counts(dropna=False)
```

```
[23]: Country
      India                   8816
      United States             69
      United Arab Emirates      53
      Singapore                 24
      Saudi Arabia              21
      United Kingdom            15
      Australia                 13
      Qatar                     10
      Hong Kong                  7
      Bahrain                    7
```



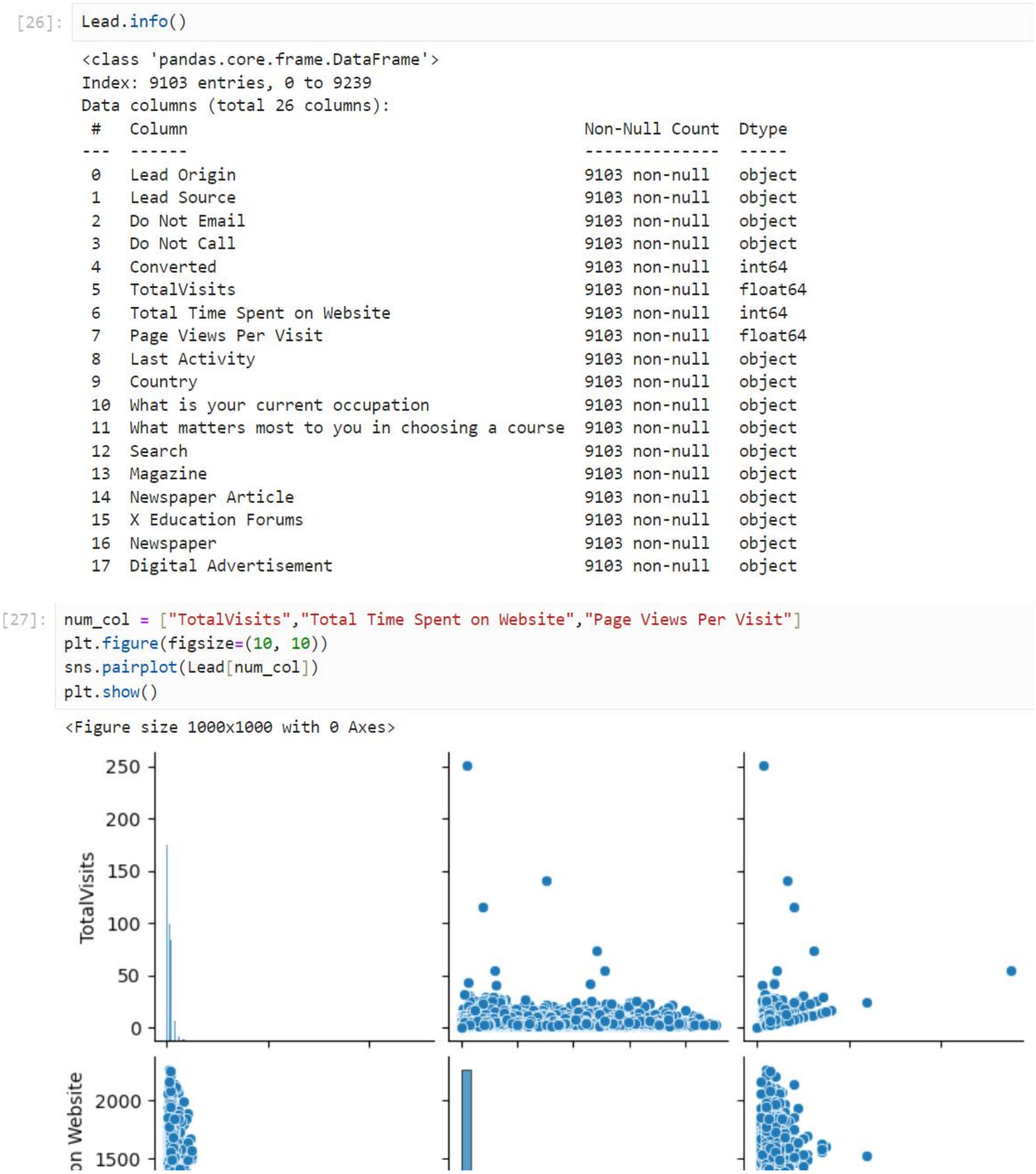
```
[25]: cat_cols = list(Lead.select_dtypes(include='object'))
      cat_cols
```

```
[25]: ['Lead Origin',
       'Lead Source',
       'Do Not Email',
       'Do Not Call',
       'Last Activity',
       'Country',
       'What is your current occupation',
       'What matters most to you in choosing a course',
       'Search',
       'Magazine',
       'Newspaper Article',
       'X Education Forums',
       'Newspaper',
       'Digital Advertisement',
       'Through Recommendations',
       'Receive More Updates About Our Courses',
       'Tags',
       'Update me on Supply Chain Content',
       'Get updates on DM Content',
       'I agree to pay the amount through cheque',
       'A free copy of Mastering The Interview',
       'Last Notable Activity']
```



```
[65]: for col in cat_cols:
          print(col)
          plt.figure(figsize=(15, 5))
          sns.countplot(x = col, hue = 'Converted', data=Lead)
          plt.show()
```

```
Prospect ID
```



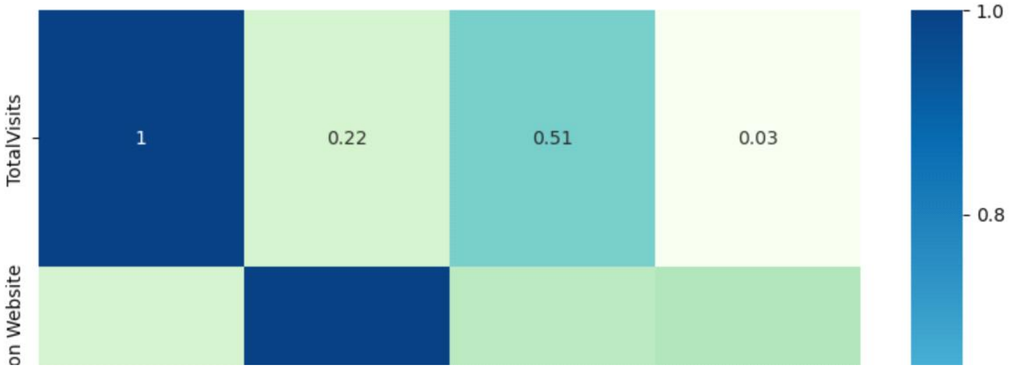
[26]:
```
Lead.info()
```

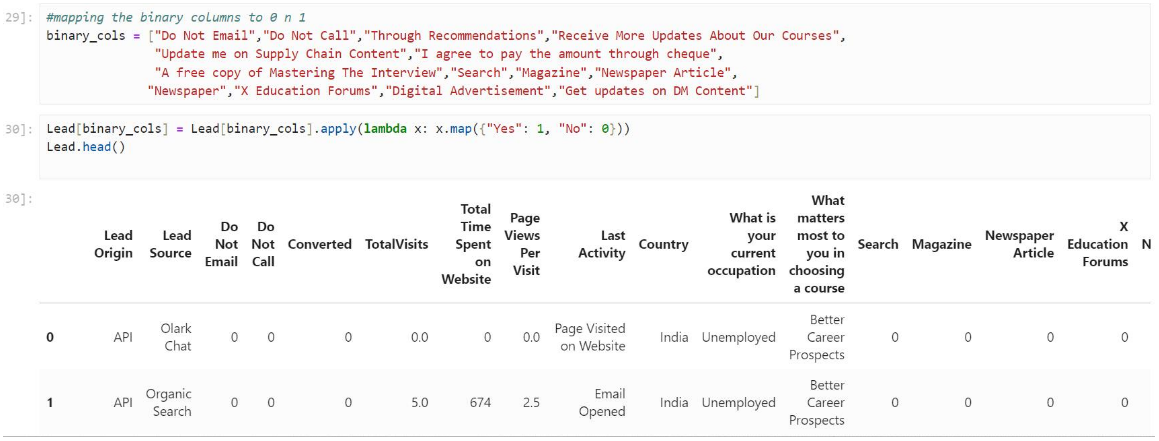
```
<class 'pandas.core.frame.DataFrame'>
Index: 9103 entries, 0 to 9239
Data columns (total 26 columns):
 #   Column                                         Non-Null Count  Dtype  
---  ------                                         --------------  -----  
 0   Lead Origin                                    9103 non-null   object 
 1   Lead Source                                    9103 non-null   object 
 2   Do Not Email                                   9103 non-null   object 
 3   Do Not Call                                    9103 non-null   object 
 4   Converted                                      9103 non-null   int64  
 5   TotalVisits                                    9103 non-null   float64
 6   Total Time Spent on Website                    9103 non-null   int64  
 7   Page Views Per Visit                           9103 non-null   float64
 8   Last Activity                                  9103 non-null   object 
 9   Country                                        9103 non-null   object 
 10  What is your current occupation                9103 non-null   object 
 11  What matters most to you in choosing a course  9103 non-null   object 
 12  Search                                         9103 non-null   object 
 13  Magazine                                       9103 non-null   object 
 14  Newspaper Article                              9103 non-null   object 
 15  X Education Forums                             9103 non-null   object 
 16  Newspaper                                      9103 non-null   object 
 17  Digital Advertisement                          9103 non-null   object 
```

[27]:
```
num_col = ["TotalVisits","Total Time Spent on Website","Page Views Per Visit"]
plt.figure(figsize=(10, 10))
sns.pairplot(Lead[num_col])
plt.show()
```

```
<Figure size 1000x1000 with 0 Axes>
```

```
[28]: num_col = Lead[["TotalVisits", "Total Time Spent on Website", "Page Views Per Visit", "Converted"]]

      # Create a heatmap for the correlation matrix
      plt.figure(figsize=(10, 10))
      sns.heatmap(num_col.corr(), annot=True, cmap='GnBu')
      plt.show()
```



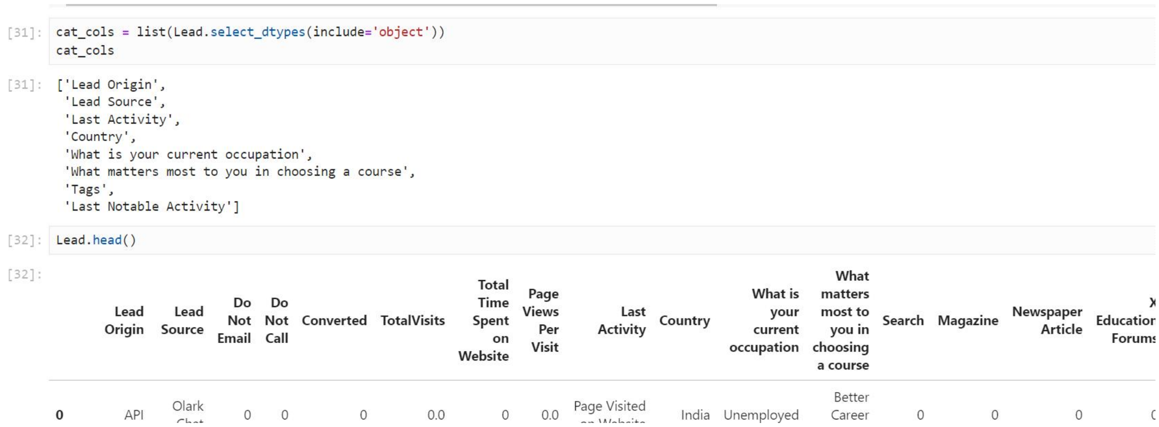


```
29]: #mapping the binary columns to 0 n 1
     binary_cols = ["Do Not Email","Do Not Call","Through Recommendations","Receive More Updates About Our Courses",
                    "Update me on Supply Chain Content","I agree to pay the amount through cheque",
                    "A free copy of Mastering The Interview","Search","Magazine","Newspaper Article",
                   "Newspaper","X Education Forums","Digital Advertisement","Get updates on DM Content"]
```

```
30]: Lead[binary_cols] = Lead[binary_cols].apply(lambda x: x.map({"Yes": 1, "No": 0}))
     Lead.head()
```

30]:

	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Country	What is your current occupation	What matters most to you in choosing a course	Search	Magazine	Newspaper Article	X Education Forums	N
0	API	Olark Chat	0	0	0	0.0	0	0.0	Page Visited on Website	India	Unemployed	Better Career Prospects	0	0	0	0	
1	API	Organic Search	0	0	0	5.0	674	2.5	Email Opened	India	Unemployed	Better Career Prospects	0	0	0	0	



```
[31]: cat_cols = list(Lead.select_dtypes(include='object'))
      cat_cols
```

```
[31]: ['Lead Origin',
       'Lead Source',
       'Last Activity',
       'Country',
       'What is your current occupation',
       'What matters most to you in choosing a course',
       'Tags',
       'Last Notable Activity']
```

```
[32]: Lead.head()
```

[32]:

	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Country	What is your current occupation	What matters most to you in choosing a course	Search	Magazine	Newspaper Article	X Education Forums
0	API	Olark	0	0	0	0.0	0	0.0	Page Visited	India	Unemployed	Better Career	0	0	0	0

[33]:

```
##import pandas as pd

# Create dummy variables for categorical columns
cat_cols_dum = pd.get_dummies(Lead[cat_cols], dtype=int)

# Check the result
cat_cols_dum.head()
```

[33]:

	Lead Origin_API	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	Lead Source_Olark Chat
0	1	0	0	0	0	0	0	0	0	0	1
1	1	0	0	0	0	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0	0	0	0
3	0	1	0	0	0	1	0	0	0	0	0

[34]:

```
Lead_final = pd.concat([cat_cols_dum,Lead],axis = 1)
```

[35]:

```
Lead_final
```

[35]:

	Lead Origin_API	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	Le Source_Ola Ch
0	1	0	0	0	0	0	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	0	

36]:

```
Lead_final.drop(cat_cols, axis= 1, inplace = True)
```

37]:

```
Lead_final
```

37]:

	Lead Origin_API	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	S
0	1	0	0	0	0	0	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	0	
2	0	1	0	0	0	1	0	0	0	0	
3	0	1	0	0	0	1	0	0	0	0	
4	0	1	0	0	0	0	0	1	0	0	

[38]:

```
x = Lead_final.drop("Converted",axis = 1)
```

[39]:

```
y = Lead_final["Converted"]
```

[40]:

```
x
```

[40]:

	Lead Origin_API	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	l Source_C
0	1	0	0	0	0	0	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	0	
2	0	1	0	0	0	1	0	0	0	0	
3	0	1	0	0	0	1	0	0	0	0	

Step 5: Train-Test-Split_ModelBuilding

```
[41]: from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import StandardScaler
```

```
[42]: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size= 0.3,random_state = 100)
```

```
[43]: scaler = StandardScaler()
      x_train[["TotalVisits","Total Time Spent on Website","Page Views Per Visit"]]= scaler.fit_transform(x_train[["TotalVisits","Total Time Spent on Website",
```

```
[44]: x_train.head()
```

[44]:

	Lead Origin_API	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Click2call	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	Le Source_Ola Cl
7962	0	1	0	0	0	0	0	1	0	0	
5520	0	1	0	0	0	1	0	0	0	0	

```
[45]: converted = sum(Lead['Converted'])/len(Lead['Converted'].index)
      converted
```

```
[45]: 0.38020432824343625
```

Step 6: ML_Model_Building

```
[46]: #import the libraries
      import statsmodels.api as sm
      from statsmodels.stats.outliers_influence import variance_inflation_factor
      import sklearn
      from sklearn.linear_model import LogisticRegression
      from sklearn.feature_selection import RFE
      from sklearn import metrics
      from sklearn.metrics import precision_recall_curve
```

```
[47]: x_train.shape
```

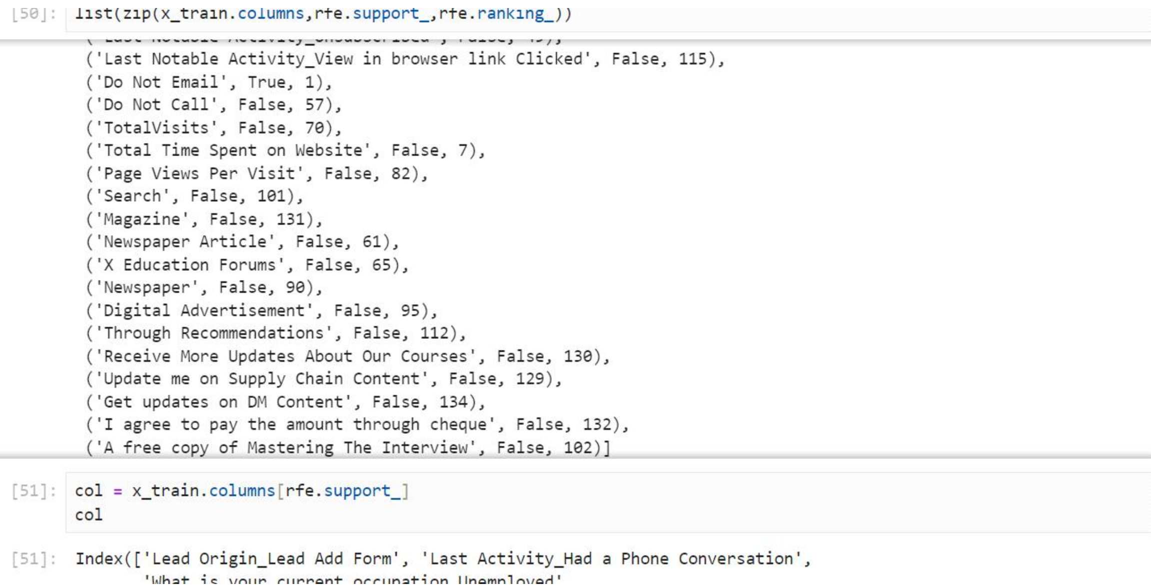
```
[47]: (6372, 148)
```

```
[48]: loggregg = LogisticRegression()
      rfe = RFE(loggregg, n_features_to_select= 15)
      rfe.fit(x_train,y_train)
```

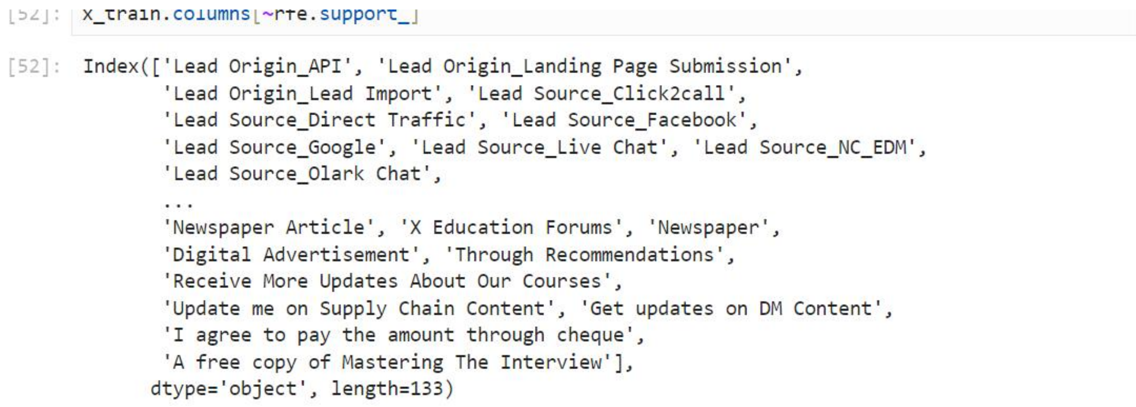
```
[48]: ▸              RFE
      ▸ estimator: LogisticRegression
          ▸ LogisticRegression
```

```
[49]: rfe.support_
```

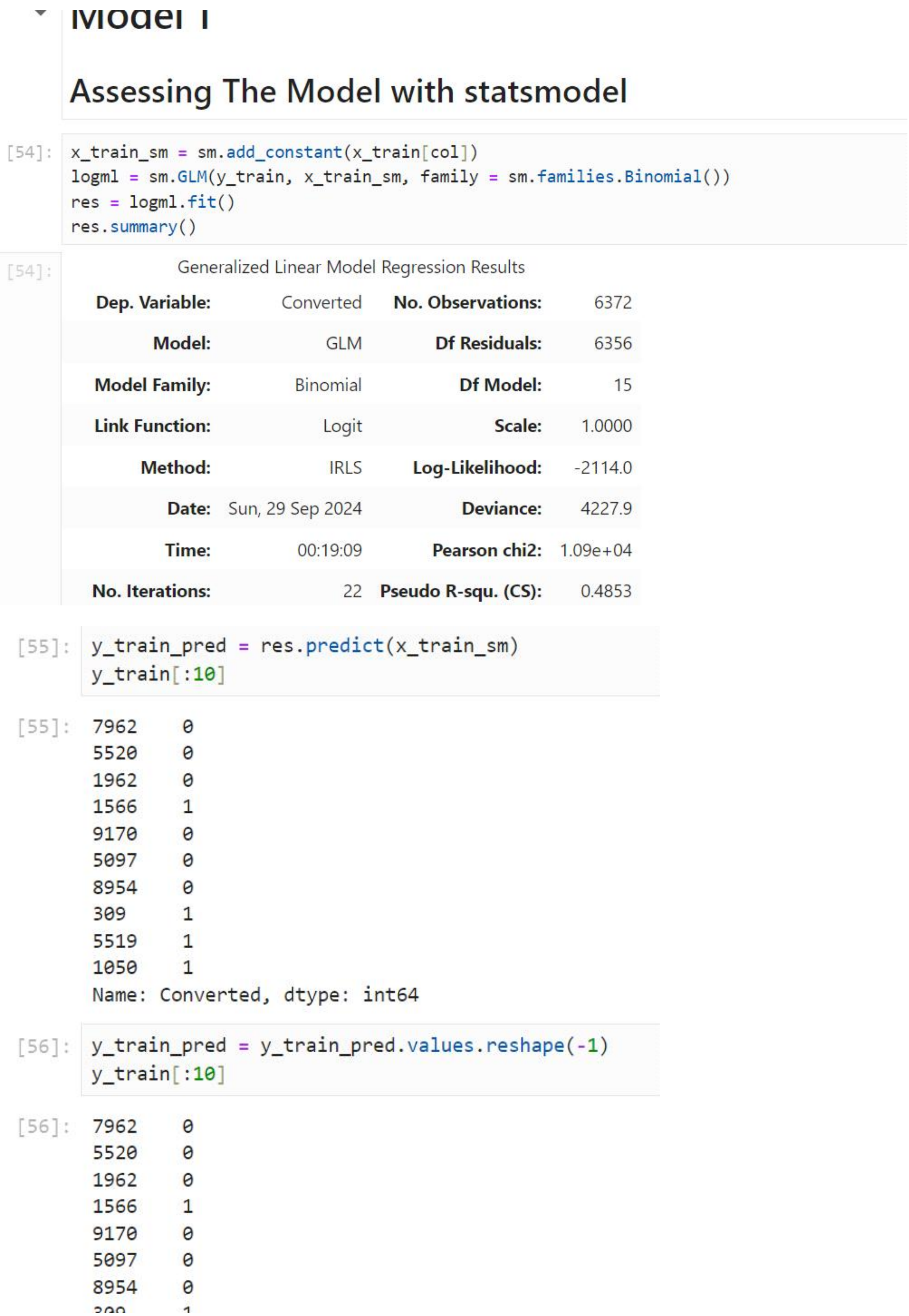
```
[49]: array([False, False,  True, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False,  True, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False, False, False, False, False, False, False,
              False, False, False,  True,  True, False, False, False,  True,
               True,  True, False, False, False, False, False, False,  True,
               True, False, False, False,  True, False, False, False, False,
               True, False, False, False, False,  True,  True, False, False,
              False, False, False, False, False, False, False, False, False,
              False,  True, False, False, False,  True, False, False, False,
              False, False, False, False, False, False, False, False, False,
```



```
[50]:  list(zip(x_train.columns,rfe.support_,rfe.ranking_))
        ('Last Notable Activity_Unsubscribed', False, 49),
        ('Last Notable Activity_View in browser link Clicked', False, 115),
        ('Do Not Email', True, 1),
        ('Do Not Call', False, 57),
        ('TotalVisits', False, 70),
        ('Total Time Spent on Website', False, 7),
        ('Page Views Per Visit', False, 82),
        ('Search', False, 101),
        ('Magazine', False, 131),
        ('Newspaper Article', False, 61),
        ('X Education Forums', False, 65),
        ('Newspaper', False, 90),
        ('Digital Advertisement', False, 95),
        ('Through Recommendations', False, 112),
        ('Receive More Updates About Our Courses', False, 130),
        ('Update me on Supply Chain Content', False, 129),
        ('Get updates on DM Content', False, 134),
        ('I agree to pay the amount through cheque', False, 132),
        ('A free copy of Mastering The Interview', False, 102)]

[51]:  col = x_train.columns[rfe.support_]
       col

[51]:  Index(['Lead Origin_Lead Add Form', 'Last Activity_Had a Phone Conversation',
              'What is your current occupation_Unemployed'
```



```
[52]:  x_train.columns[~rfe.support_]

[52]:  Index(['Lead Origin_API', 'Lead Origin_Landing Page Submission',
              'Lead Origin_Lead Import', 'Lead Source_Click2call',
              'Lead Source_Direct Traffic', 'Lead Source_Facebook',
              'Lead Source_Google', 'Lead Source_Live Chat', 'Lead Source_NC_EDM',
              'Lead Source_Olark Chat',
              ...
              'Newspaper Article', 'X Education Forums', 'Newspaper',
              'Digital Advertisement', 'Through Recommendations',
              'Receive More Updates About Our Courses',
              'Update me on Supply Chain Content', 'Get updates on DM Content',
              'I agree to pay the amount through cheque',
              'A free copy of Mastering The Interview'],
             dtype='object', length=133)
```



Model 1

Assessing The Model with statsmodel

```
[54]: x_train_sm = sm.add_constant(x_train[col])
      logml = sm.GLM(y_train, x_train_sm, family = sm.families.Binomial())
      res = logml.fit()
      res.summary()
```

[54]: Generalized Linear Model Regression Results

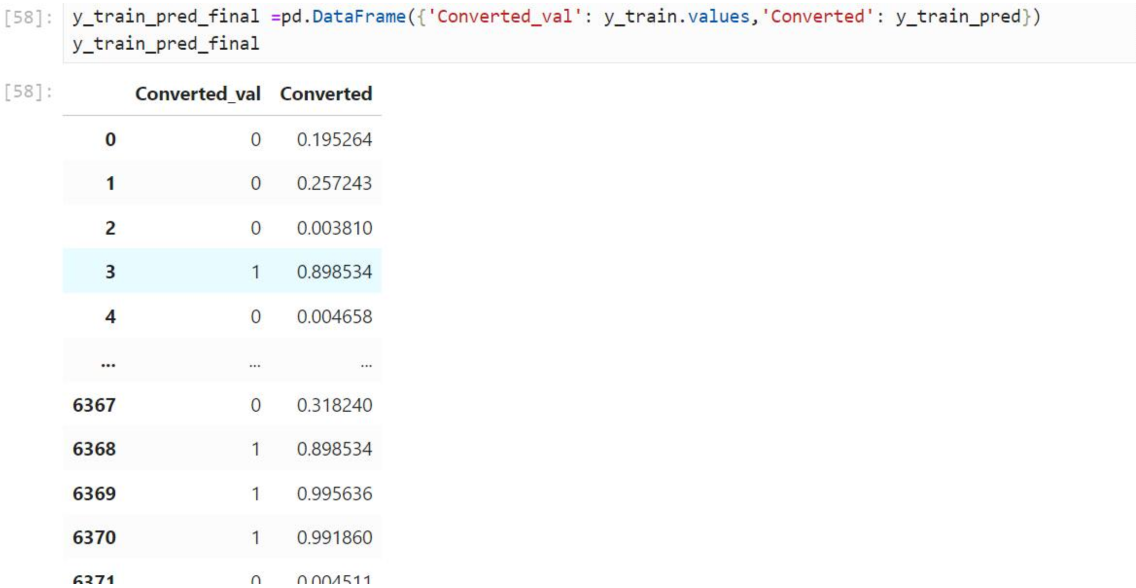
Dep. Variable:	Converted	**No. Observations:**	6372
Model:	GLM	**Df Residuals:**	6356
Model Family:	Binomial	**Df Model:**	15
Link Function:	Logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2114.0
Date:	Sun, 29 Sep 2024	**Deviance:**	4227.9
Time:	00:19:09	**Pearson chi2:**	1.09e+04
No. Iterations:	22	**Pseudo R-squ. (CS):**	0.4853

```
[55]: y_train_pred = res.predict(x_train_sm)
      y_train[:10]
```

```
[55]: 7962    0
      5520    0
      1962    0
      1566    1
      9170    0
      5097    0
      8954    0
      309     1
      5519    1
      1050    1
      Name: Converted, dtype: int64
```

```
[56]: y_train_pred = y_train_pred.values.reshape(-1)
      y_train[:10]
```

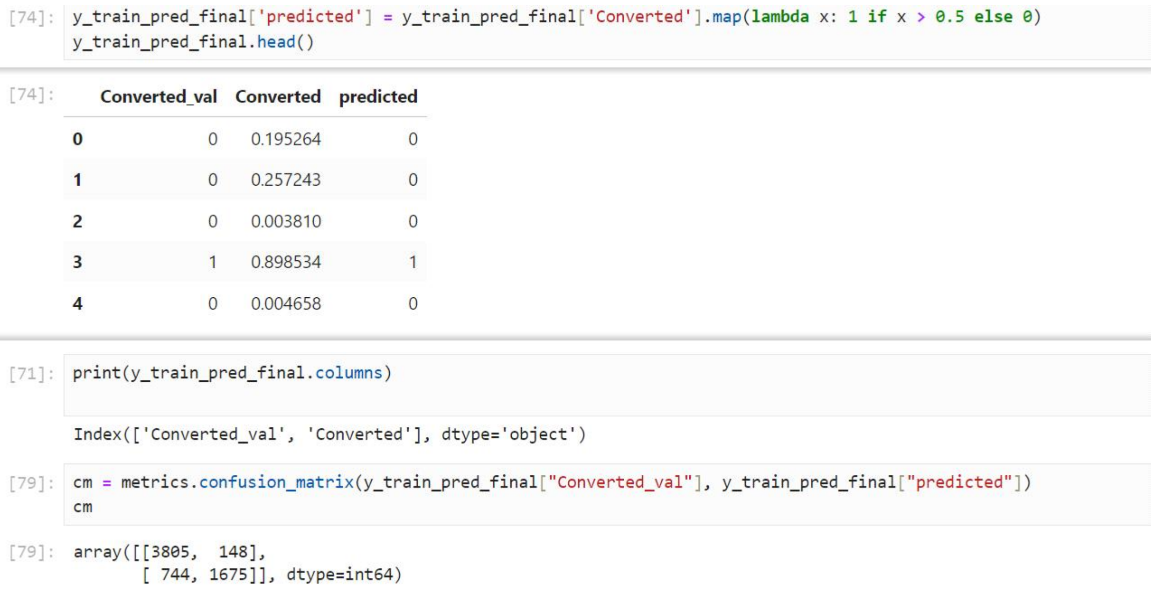
```
[56]: 7962    0
      5520    0
      1962    0
      1566    1
      9170    0
      5097    0
      8954    0
      309     1
```



[58]:
```
y_train_pred_final =pd.DataFrame({'Converted_val': y_train.values,'Converted': y_train_pred})
y_train_pred_final
```

[58]:

	Converted_val	Converted
0	0	0.195264
1	0	0.257243
2	0	0.003810
3	1	0.898534
4	0	0.004658
...	...	...
6367	0	0.318240
6368	1	0.898534
6369	1	0.995636
6370	1	0.991860
6371	0	0.004511



[74]:
```
y_train_pred_final['predicted'] = y_train_pred_final['Converted'].map(lambda x: 1 if x > 0.5 else 0)
y_train_pred_final.head()
```

[74]:

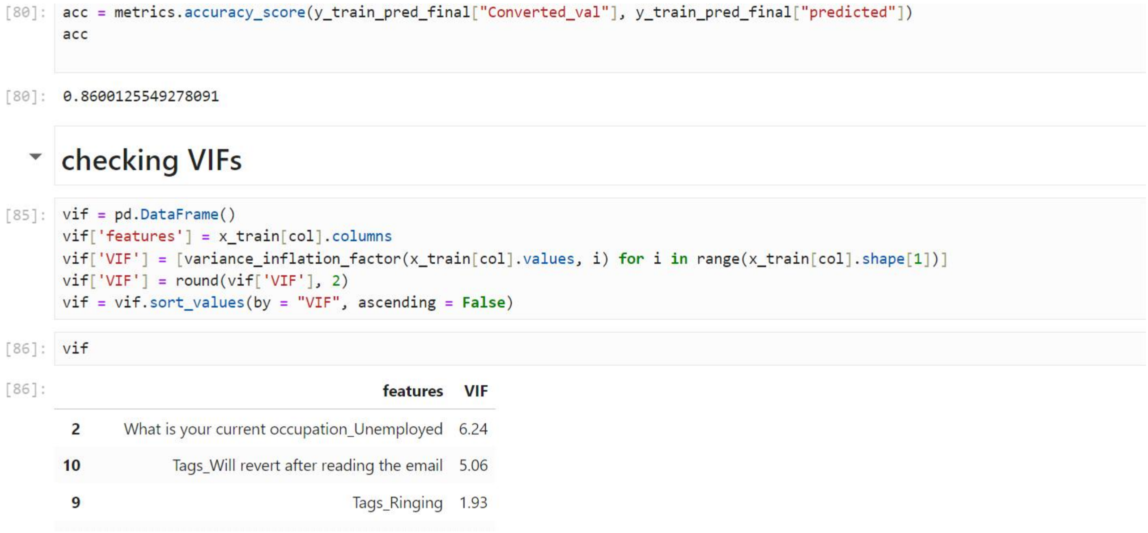
	Converted_val	Converted	predicted
0	0	0.195264	0
1	0	0.257243	0
2	0	0.003810	0
3	1	0.898534	1
4	0	0.004658	0

[71]:
```
print(y_train_pred_final.columns)
```

```
Index(['Converted_val', 'Converted'], dtype='object')
```

[79]:
```
cm = metrics.confusion_matrix(y_train_pred_final["Converted_val"], y_train_pred_final["predicted"])
cm
```

[79]:
```
array([[3805,  148],
       [ 744, 1675]], dtype=int64)
```



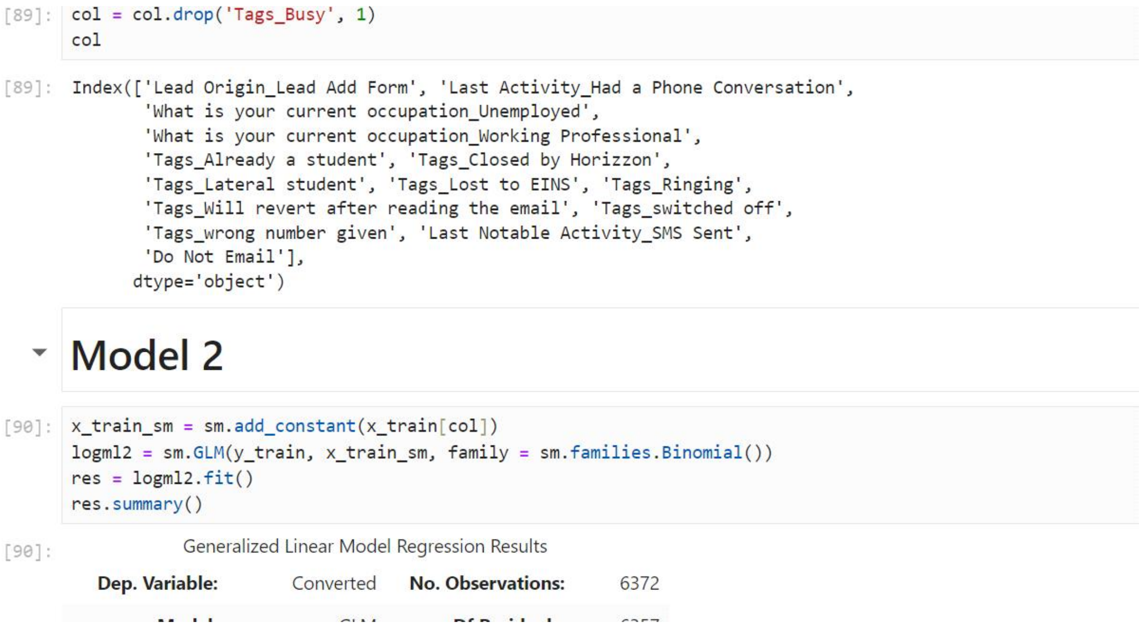
```
[80]: acc = metrics.accuracy_score(y_train_pred_final["Converted_val"], y_train_pred_final["predicted"])
      acc

[80]: 0.8600125549278091
```

checking VIFs

```
[85]: vif = pd.DataFrame()
      vif['features'] = x_train[col].columns
      vif['VIF'] = [variance_inflation_factor(x_train[col].values, i) for i in range(x_train[col].shape[1])]
      vif['VIF'] = round(vif['VIF'], 2)
      vif = vif.sort_values(by = "VIF", ascending = False)

[86]: vif
```

[86]:

	features	VIF
2	What is your current occupation_Unemployed	6.24
10	Tags_Will revert after reading the email	5.06
9	Tags_Ringing	1.93



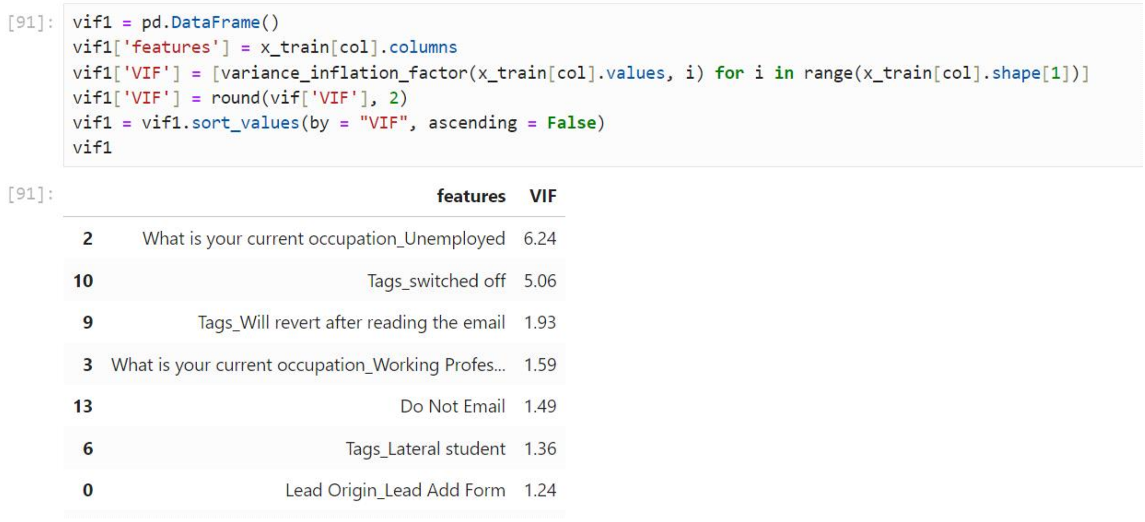
```
[89]: col = col.drop('Tags_Busy', 1)
      col

[89]: Index(['Lead Origin_Lead Add Form', 'Last Activity_Had a Phone Conversation',
             'What is your current occupation_Unemployed',
             'What is your current occupation_Working Professional',
             'Tags_Already a student', 'Tags_Closed by Horizzon',
             'Tags_Lateral student', 'Tags_Lost to EINS', 'Tags_Ringing',
             'Tags_Will revert after reading the email', 'Tags_switched off',
             'Tags_wrong number given', 'Last Notable Activity_SMS Sent',
             'Do Not Email'],
            dtype='object')
```

Model 2

```
[90]: x_train_sm = sm.add_constant(x_train[col])
      logml2 = sm.GLM(y_train, x_train_sm, family = sm.families.Binomial())
      res = logml2.fit()
      res.summary()
```

[90]: Generalized Linear Model Regression Results

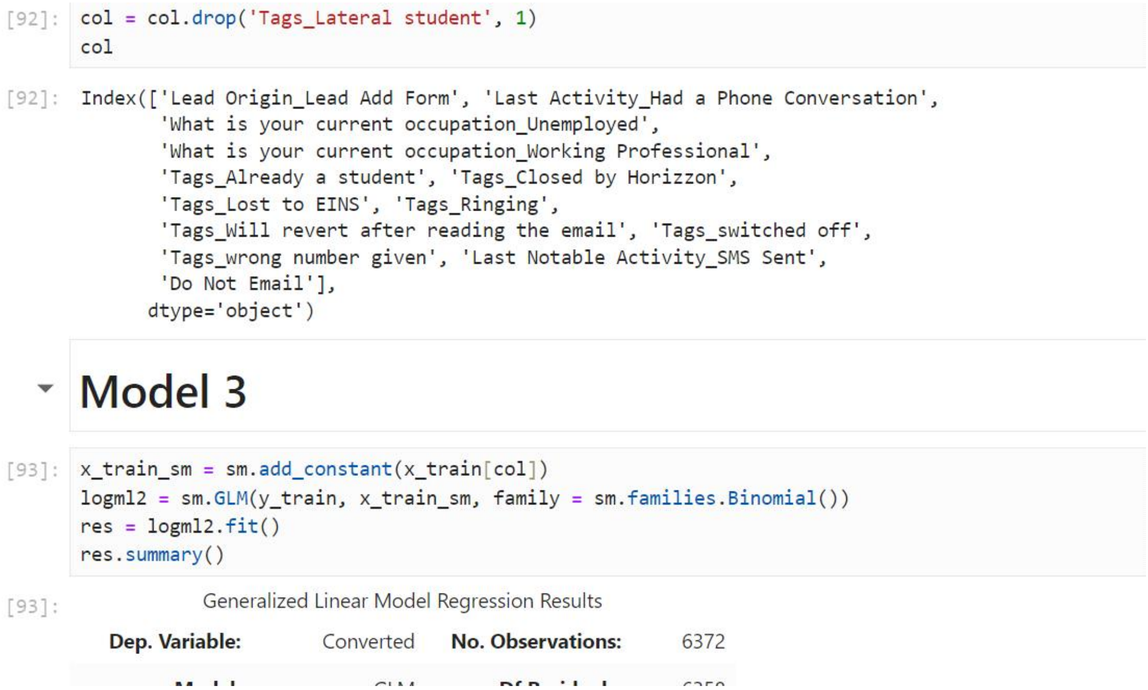
Dep. Variable:	Converted	**No. Observations:**	6372
Model:	GLM	**Df Residuals:**	6357



```
[91]: vif1 = pd.DataFrame()
      vif1['features'] = x_train[col].columns
      vif1['VIF'] = [variance_inflation_factor(x_train[col].values, i) for i in range(x_train[col].shape[1])]
      vif1['VIF'] = round(vif['VIF'], 2)
      vif1 = vif1.sort_values(by = "VIF", ascending = False)
      vif1
```

[91]:

	features	VIF
2	What is your current occupation_Unemployed	6.24
10	Tags_switched off	5.06
9	Tags_Will revert after reading the email	1.93
3	What is your current occupation_Working Profes...	1.59
13	Do Not Email	1.49
6	Tags_Lateral student	1.36
0	Lead Origin_Lead Add Form	1.24



```
[92]: col = col.drop('Tags_Lateral student', 1)
      col
```

```
[92]: Index(['Lead Origin_Lead Add Form', 'Last Activity_Had a Phone Conversation',
             'What is your current occupation_Unemployed',
             'What is your current occupation_Working Professional',
             'Tags_Already a student', 'Tags_Closed by Horizzon',
             'Tags_Lost to EINS', 'Tags_Ringing',
             'Tags_Will revert after reading the email', 'Tags_switched off',
             'Tags_wrong number given', 'Last Notable Activity_SMS Sent',
             'Do Not Email'],
            dtype='object')
```

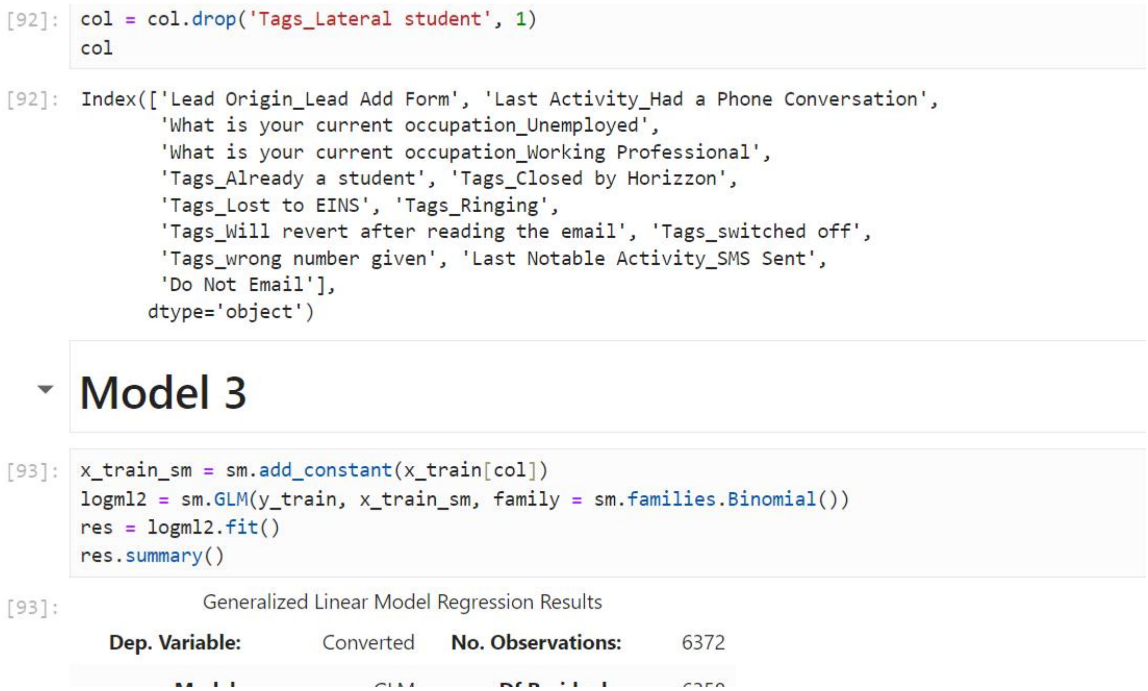
Model 3

```
[93]: x_train_sm = sm.add_constant(x_train[col])
      logml2 = sm.GLM(y_train, x_train_sm, family = sm.families.Binomial())
      res = logml2.fit()
      res.summary()
```

[93]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6372
Model:	GLM	**Df Residuals:**	6358



```
[92]: col = col.drop('Tags_Lateral student', 1)
      col
```

```
[92]: Index(['Lead Origin_Lead Add Form', 'Last Activity_Had a Phone Conversation',
             'What is your current occupation_Unemployed',
             'What is your current occupation_Working Professional',
             'Tags_Already a student', 'Tags_Closed by Horizzon',
             'Tags_Lost to EINS', 'Tags_Ringing',
             'Tags_Will revert after reading the email', 'Tags_switched off',
             'Tags_wrong number given', 'Last Notable Activity_SMS Sent',
             'Do Not Email'],
            dtype='object')
```

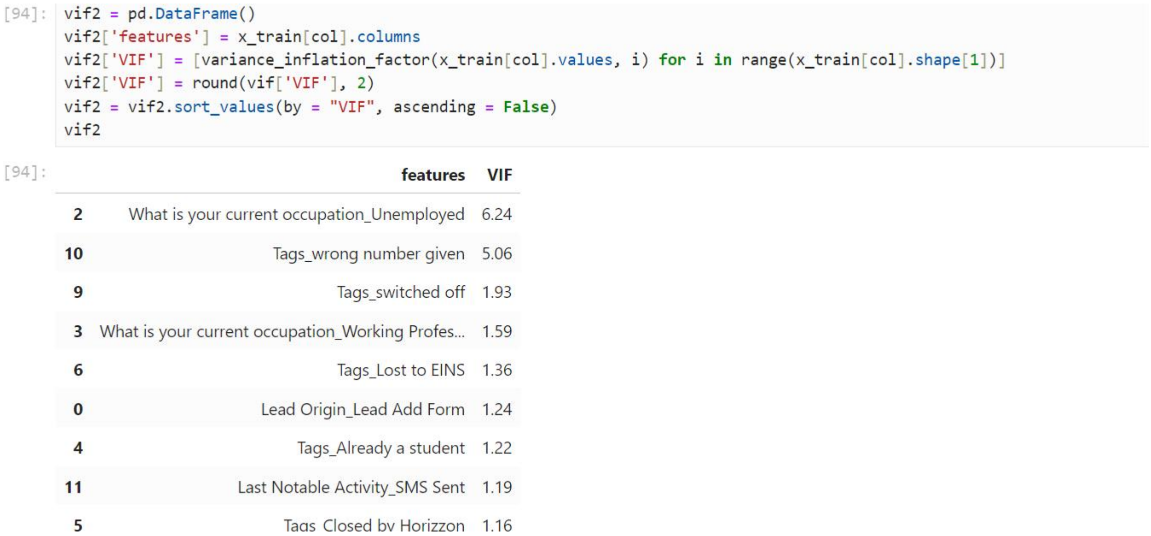
Model 3

```
[93]: x_train_sm = sm.add_constant(x_train[col])
      logml2 = sm.GLM(y_train, x_train_sm, family = sm.families.Binomial())
      res = logml2.fit()
      res.summary()
```

[93]:

Generalized Linear Model Regression Results

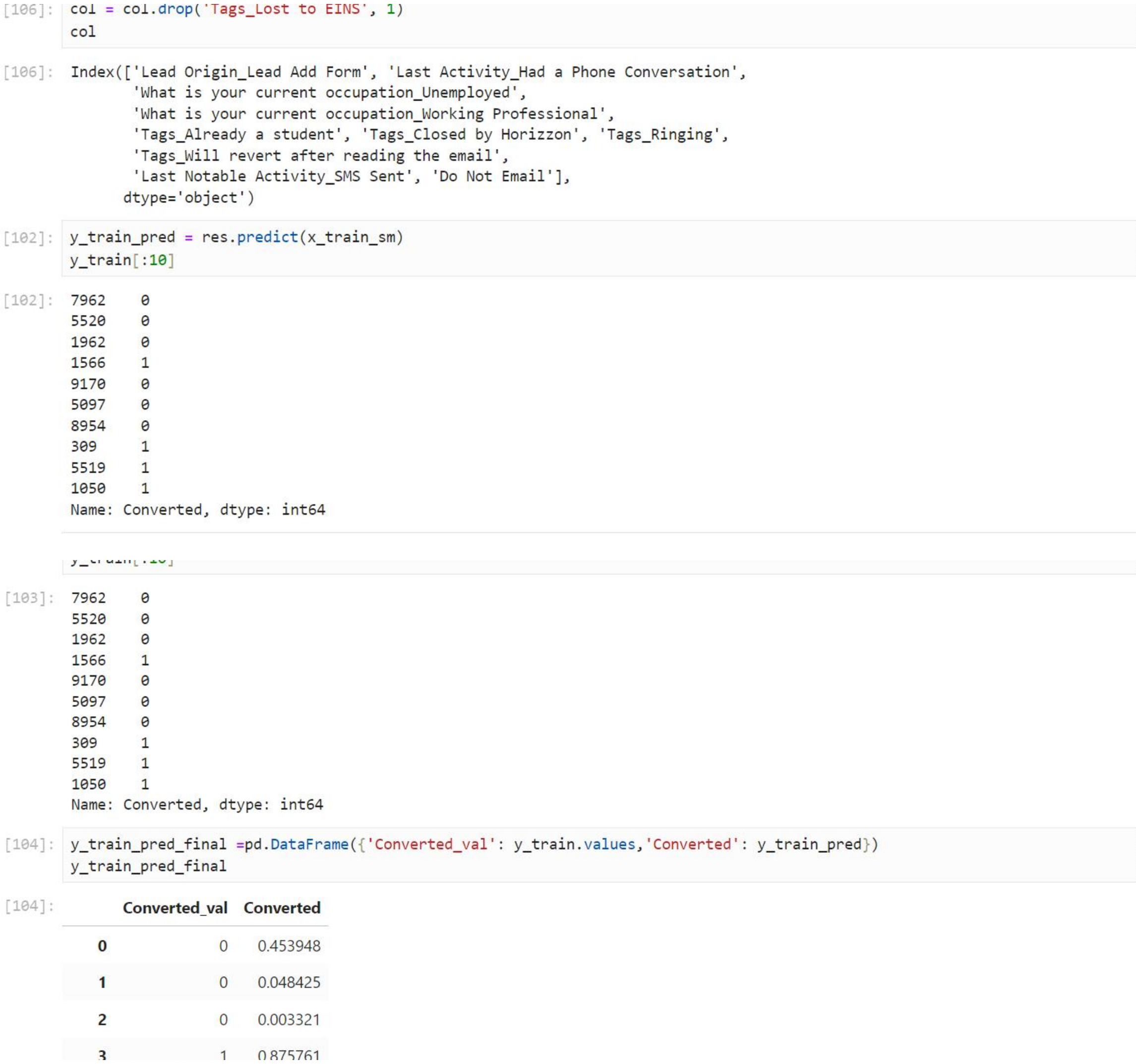
Dep. Variable:	Converted	**No. Observations:**	6372



```
[94]: vif2 = pd.DataFrame()
      vif2['features'] = x_train[col].columns
      vif2['VIF'] = [variance_inflation_factor(x_train[col].values, i) for i in range(x_train[col].shape[1])]
      vif2['VIF'] = round(vif['VIF'], 2)
      vif2 = vif2.sort_values(by = "VIF", ascending = False)
      vif2
```

[94]:

	features	VIF
2	What is your current occupation_Unemployed	6.24
10	Tags_wrong number given	5.06
9	Tags_switched off	1.93
3	What is your current occupation_Working Profes...	1.59
6	Tags_Lost to EINS	1.36
0	Lead Origin_Lead Add Form	1.24
4	Tags_Already a student	1.22
11	Last Notable Activity_SMS Sent	1.19
5	Tags_Closed by Horizzon	1.16



```
[106]: col = col.drop('Tags_Lost to EINS', 1)
       col
```

```
[106]: Index(['Lead Origin_Lead Add Form', 'Last Activity_Had a Phone Conversation',
              'What is your current occupation_Unemployed',
              'What is your current occupation_Working Professional',
              'Tags_Already a student', 'Tags_Closed by Horizzon', 'Tags_Ringing',
              'Tags_Will revert after reading the email',
              'Last Notable Activity_SMS Sent', 'Do Not Email'],
             dtype='object')
```

```
[102]: y_train_pred = res.predict(x_train_sm)
       y_train[:10]
```

```
[102]: 7962    0
       5520    0
       1962    0
       1566    1
       9170    0
       5097    0
       8954    0
       309     1
       5519    1
       1050    1
       Name: Converted, dtype: int64
```

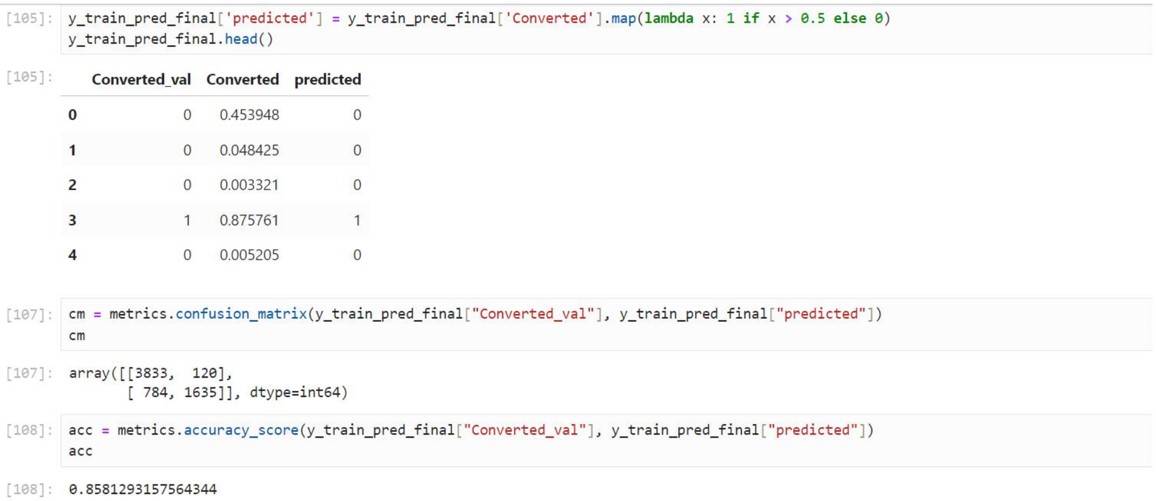
```
       y_train[:10]
```

```
[103]: 7962    0
       5520    0
       1962    0
       1566    1
       9170    0
       5097    0
       8954    0
       309     1
       5519    1
       1050    1
       Name: Converted, dtype: int64
```

```
[104]: y_train_pred_final =pd.DataFrame({'Converted_val': y_train.values,'Converted': y_train_pred})
       y_train_pred_final
```

[104]:

	Converted_val	Converted
0	0	0.453948
1	0	0.048425
2	0	0.003321
3	1	0.875761



```
[105]: y_train_pred_final['predicted'] = y_train_pred_final['Converted'].map(lambda x: 1 if x > 0.5 else 0)
       y_train_pred_final.head()
```

[105]:

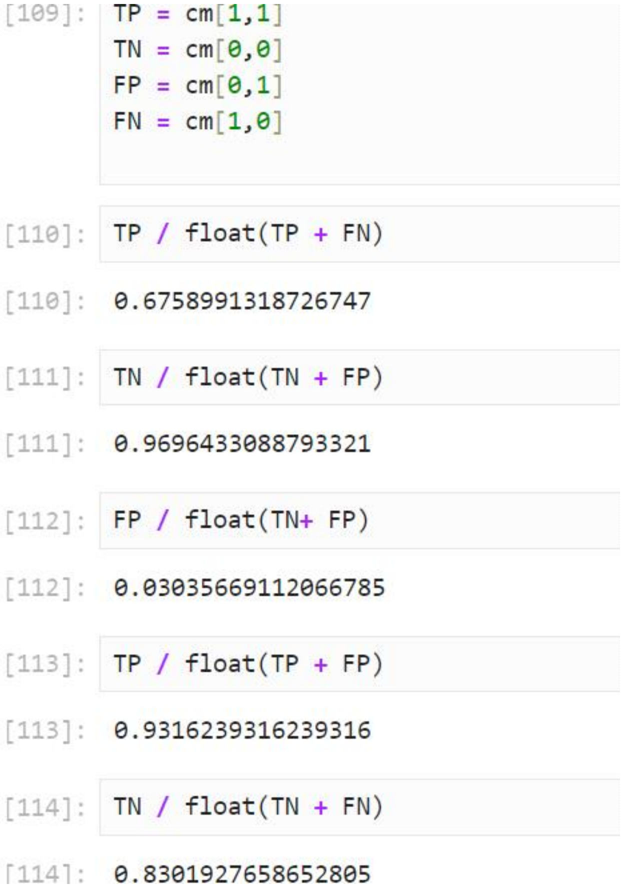
	Converted_val	Converted	predicted
0	0	0.453948	0
1	0	0.048425	0
2	0	0.003321	0
3	1	0.875761	1
4	0	0.005205	0

```
[107]: cm = metrics.confusion_matrix(y_train_pred_final["Converted_val"], y_train_pred_final["predicted"])
       cm
```

```
[107]: array([[3833,  120],
              [ 784, 1635]], dtype=int64)
```

```
[108]: acc = metrics.accuracy_score(y_train_pred_final["Converted_val"], y_train_pred_final["predicted"])
       acc
```

```
[108]: 0.8581293157564344
```



```
[109]: TP = cm[1,1]
       TN = cm[0,0]
       FP = cm[0,1]
       FN = cm[1,0]
```

```
[110]: TP / float(TP + FN)
```

```
[110]: 0.6758991318726747
```

```
[111]: TN / float(TN + FP)
```

```
[111]: 0.9696433088793321
```

```
[112]: FP / float(TN+ FP)
```

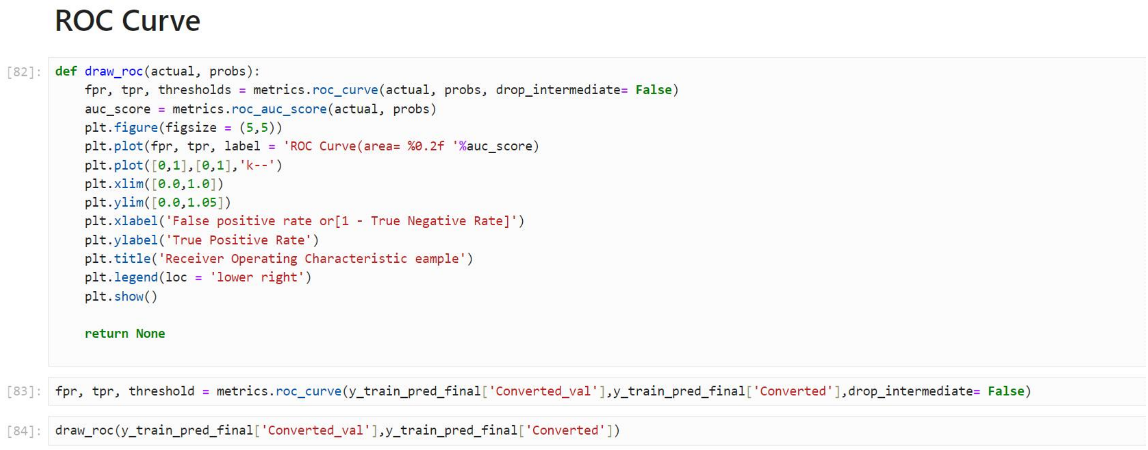
```
[112]: 0.03035669112066785
```

```
[113]: TP / float(TP + FP)
```

```
[113]: 0.9316239316239316
```

```
[114]: TN / float(TN + FN)
```

```
[114]: 0.8301927658652805
```

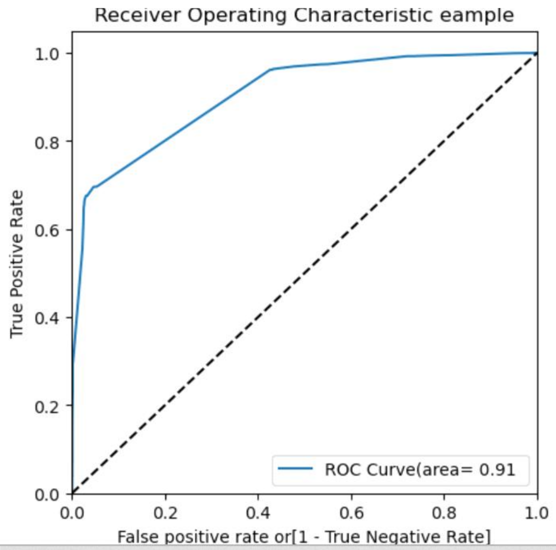


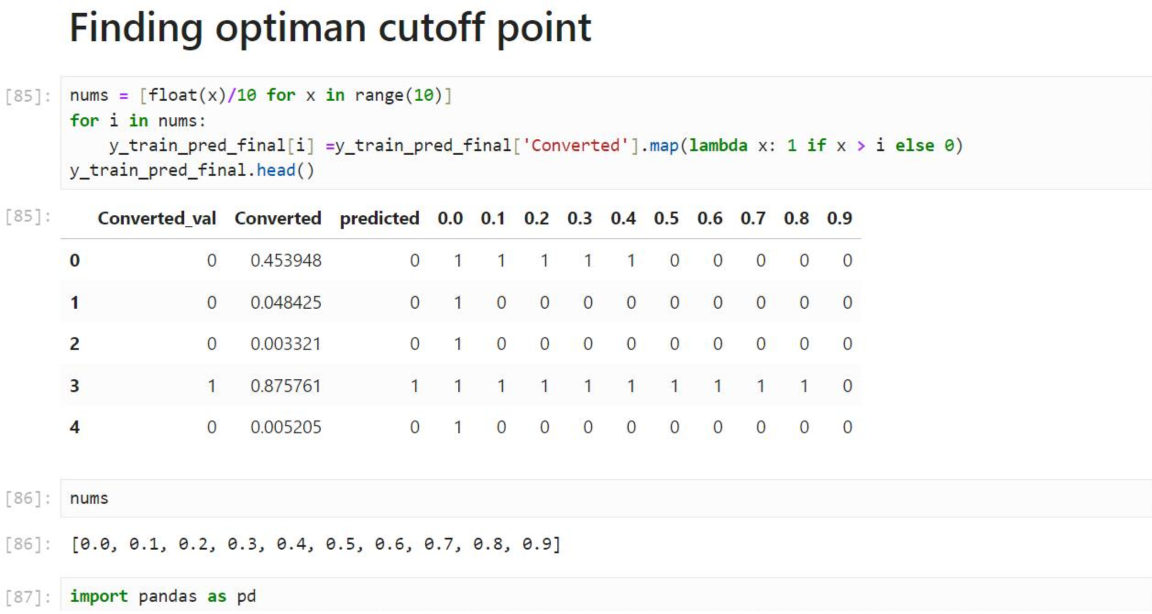
ROC Curve

```
[82]: def draw_roc(actual, probs):
          fpr, tpr, thresholds = metrics.roc_curve(actual, probs, drop_intermediate= False)
          auc_score = metrics.roc_auc_score(actual, probs)
          plt.figure(figsize = (5,5))
          plt.plot(fpr, tpr, label = 'ROC Curve(area= %0.2f '%auc_score)
          plt.plot([0,1],[0,1],'k--')
          plt.xlim([0.0,1.0])
          plt.ylim([0.0,1.05])
          plt.xlabel('False positive rate or[1 - True Negative Rate]')
          plt.ylabel('True Positive Rate')
          plt.title('Receiver Operating Characteristic eample')
          plt.legend(loc = 'lower right')
          plt.show()

          return None
```

```
[83]: fpr, tpr, threshold = metrics.roc_curve(y_train_pred_final['Converted_val'],y_train_pred_final['Converted'],drop_intermediate= False)
```

```
[84]: draw_roc(y_train_pred_final['Converted_val'],y_train_pred_final['Converted'])
```





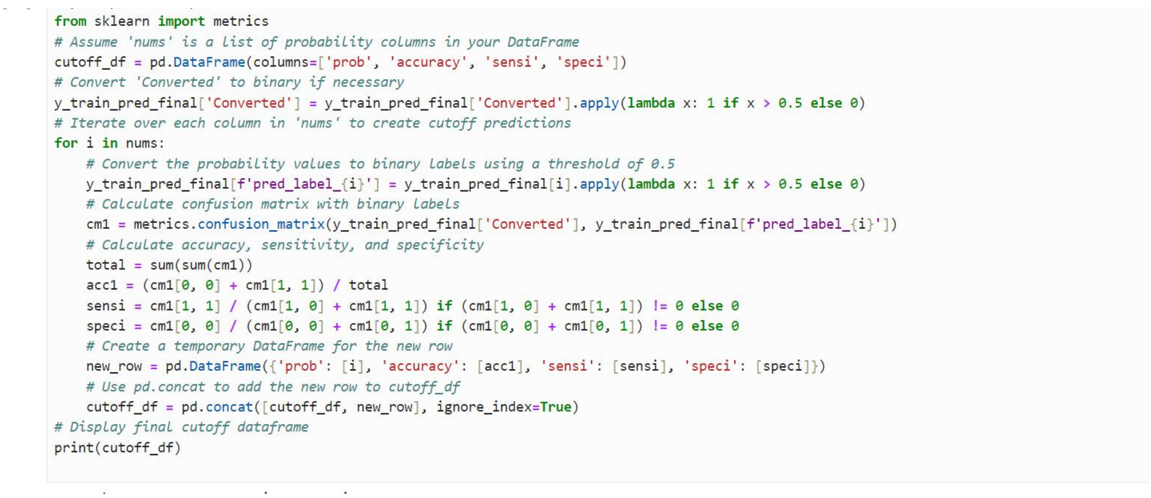
Finding optiman cutoff point

```
[85]: nums = [float(x)/10 for x in range(10)]
      for i in nums:
          y_train_pred_final[i] =y_train_pred_final['Converted'].map(lambda x: 1 if x > i else 0)
      y_train_pred_final.head()
```

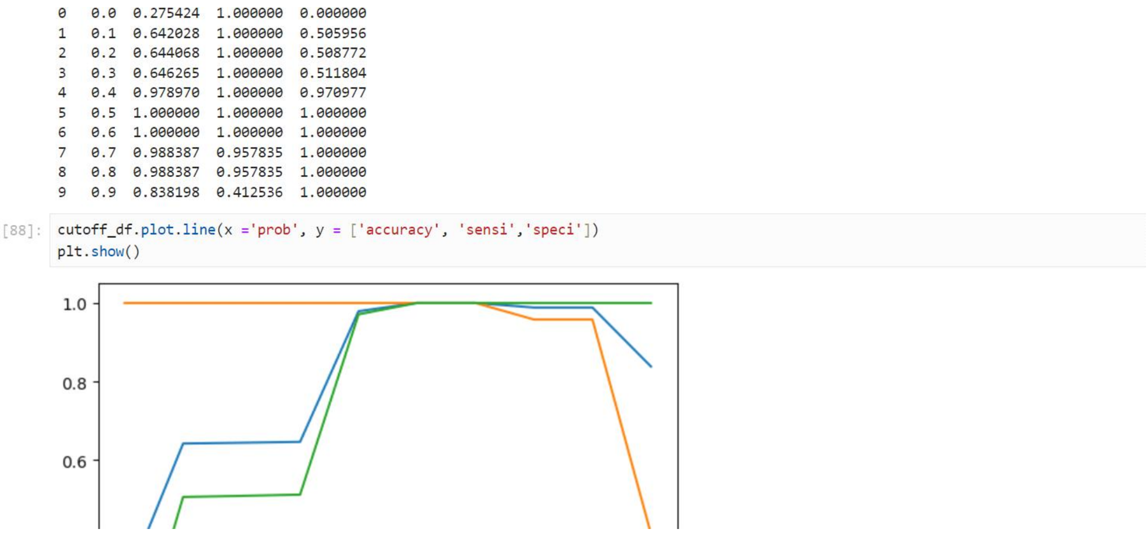
[85]:

	Converted_val	Converted	predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	0	0.453948	0	1	1	1	1	1	0	0	0	0	0
1	0	0.048425	0	1	0	0	0	0	0	0	0	0	0
2	0	0.003321	0	1	0	0	0	0	0	0	0	0	0
3	1	0.875761	1	1	1	1	1	1	1	1	1	1	0
4	0	0.005205	0	1	0	0	0	0	0	0	0	0	0

```
[86]: nums
[86]: [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
[87]: import pandas as pd
```



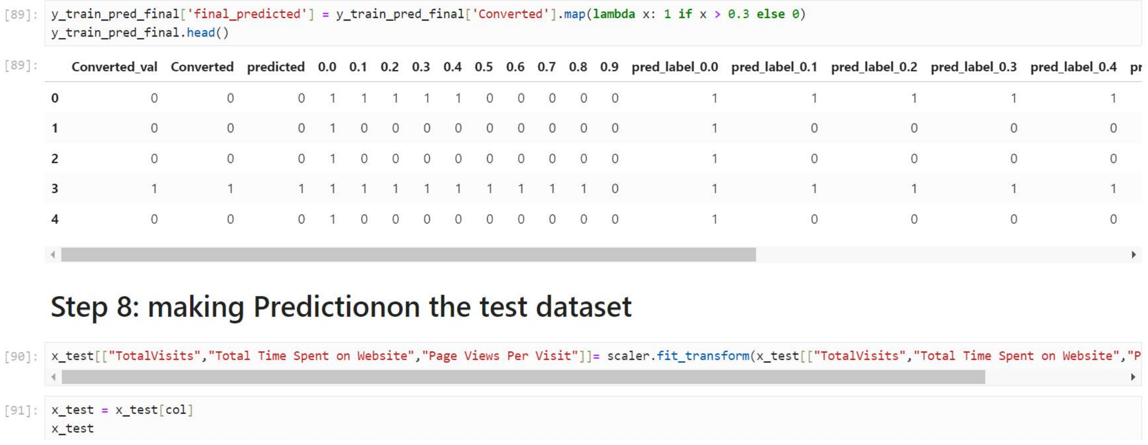
```
from sklearn import metrics
# Assume 'nums' is a list of probability columns in your DataFrame
cutoff_df = pd.DataFrame(columns=['prob', 'accuracy', 'sensi', 'speci'])
# Convert 'Converted' to binary if necessary
y_train_pred_final['Converted'] = y_train_pred_final['Converted'].apply(lambda x: 1 if x > 0.5 else 0)
# Iterate over each column in 'nums' to create cutoff predictions
for i in nums:
    # Convert the probability values to binary labels using a threshold of 0.5
    y_train_pred_final[f'pred_label_{i}'] = y_train_pred_final[i].apply(lambda x: 1 if x > 0.5 else 0)
    # Calculate confusion matrix with binary labels
    cm1 = metrics.confusion_matrix(y_train_pred_final['Converted'], y_train_pred_final[f'pred_label_{i}'])
    # Calculate accuracy, sensitivity, and specificity
    total = sum(sum(cm1))
    acc1 = (cm1[0, 0] + cm1[1, 1]) / total
    sensi = cm1[1, 1] / (cm1[1, 0] + cm1[1, 1]) if (cm1[1, 0] + cm1[1, 1]) != 0 else 0
    speci = cm1[0, 0] / (cm1[0, 0] + cm1[0, 1]) if (cm1[0, 0] + cm1[0, 1]) != 0 else 0
    # Create a temporary DataFrame for the new row
    new_row = pd.DataFrame({'prob': [i], 'accuracy': [acc1], 'sensi': [sensi], 'speci': [speci]})
    # Use pd.concat to add the new row to cutoff_df
    cutoff_df = pd.concat([cutoff_df, new_row], ignore_index=True)
# Display final cutoff dataframe
print(cutoff_df)
```



```
0   0.0  0.275424  1.000000  0.000000
1   0.1  0.642028  1.000000  0.505956
2   0.2  0.644068  1.000000  0.508772
3   0.3  0.646265  1.000000  0.511804
4   0.4  0.978970  1.000000  0.970977
5   0.5  1.000000  1.000000  1.000000
6   0.6  1.000000  1.000000  1.000000
7   0.7  0.988387  0.957835  1.000000
8   0.8  0.988387  0.957835  1.000000
9   0.9  0.838198  0.412536  1.000000
```

[88]:

```
cutoff_df.plot.line(x ='prob', y = ['accuracy', 'sensi','speci'])
plt.show()
```



[89]:

```
y_train_pred_final['final_predicted'] = y_train_pred_final['Converted'].map(lambda x: 1 if x > 0.3 else 0)
y_train_pred_final.head()
```

[89]:

	Converted_val	Converted	predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	pred_label_0.0	pred_label_0.1	pred_label_0.2	pred_label_0.3	pred_label_0.4	pr
0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	
1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
2	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
3	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	
4	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	

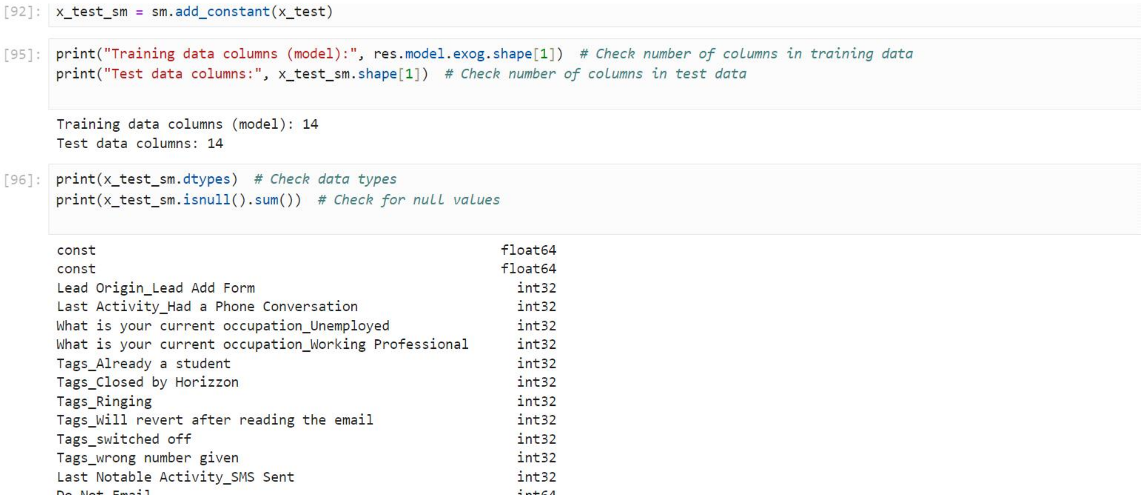
Step 8: making Predictionon the test dataset

[90]:

```
x_test[["TotalVisits","Total Time Spent on Website","Page Views Per Visit"]]= scaler.fit_transform(x_test[["TotalVisits","Total Time Spent on Website","P
```

[91]:

```
x_test = x_test[col]
x_test
```



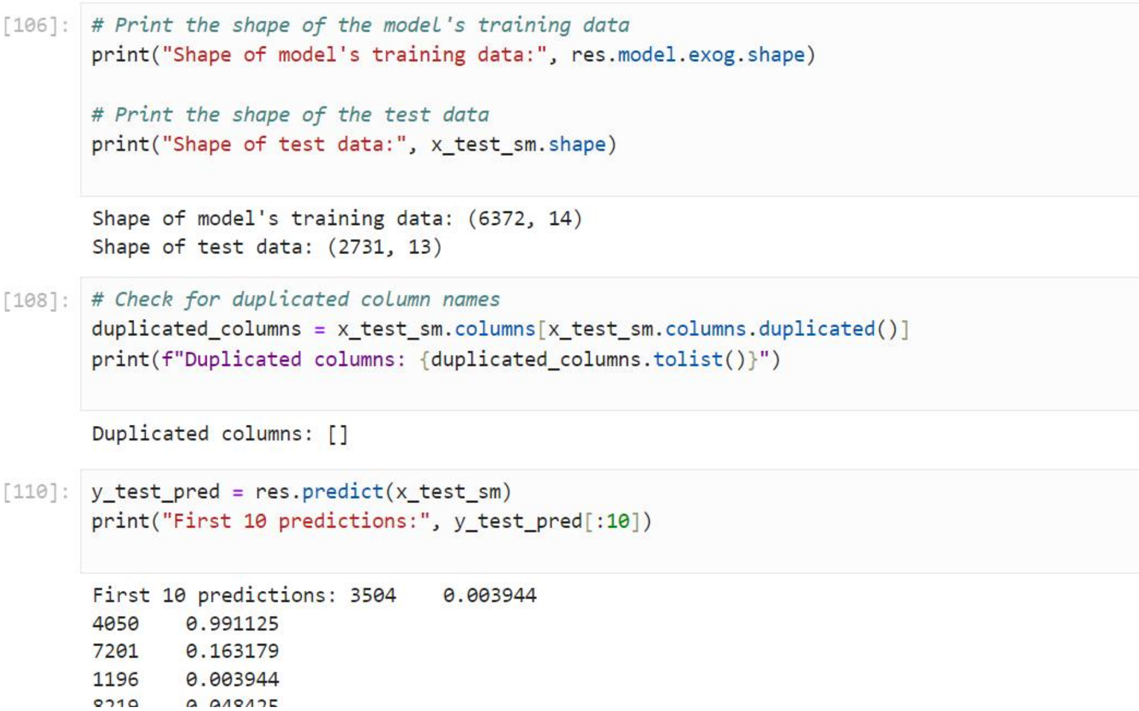
```
[92]: x_test_sm = sm.add_constant(x_test)
```

```
[95]: print("Training data columns (model):", res.model.exog.shape[1])  # Check number of columns in training data
      print("Test data columns:", x_test_sm.shape[1])  # Check number of columns in test data
```

```
Training data columns (model): 14
Test data columns: 14
```

```
[96]: print(x_test_sm.dtypes)  # Check data types
      print(x_test_sm.isnull().sum())  # Check for null values
```

```
const                                                  float64
const                                                  float64
Lead Origin_Lead Add Form                                int32
Last Activity_Had a Phone Conversation                   int32
What is your current occupation_Unemployed               int32
What is your current occupation_Working Professional     int32
Tags_Already a student                                   int32
Tags_Closed by Horizzon                                  int32
Tags_Ringing                                             int32
Tags_Will revert after reading the email                 int32
Tags_switched off                                        int32
Tags_wrong number given                                  int32
Last Notable Activity_SMS Sent                           int32
Do Not Email                                             int64
```



```
[106]: # Print the shape of the model's training data
       print("Shape of model's training data:", res.model.exog.shape)

       # Print the shape of the test data
       print("Shape of test data:", x_test_sm.shape)
```

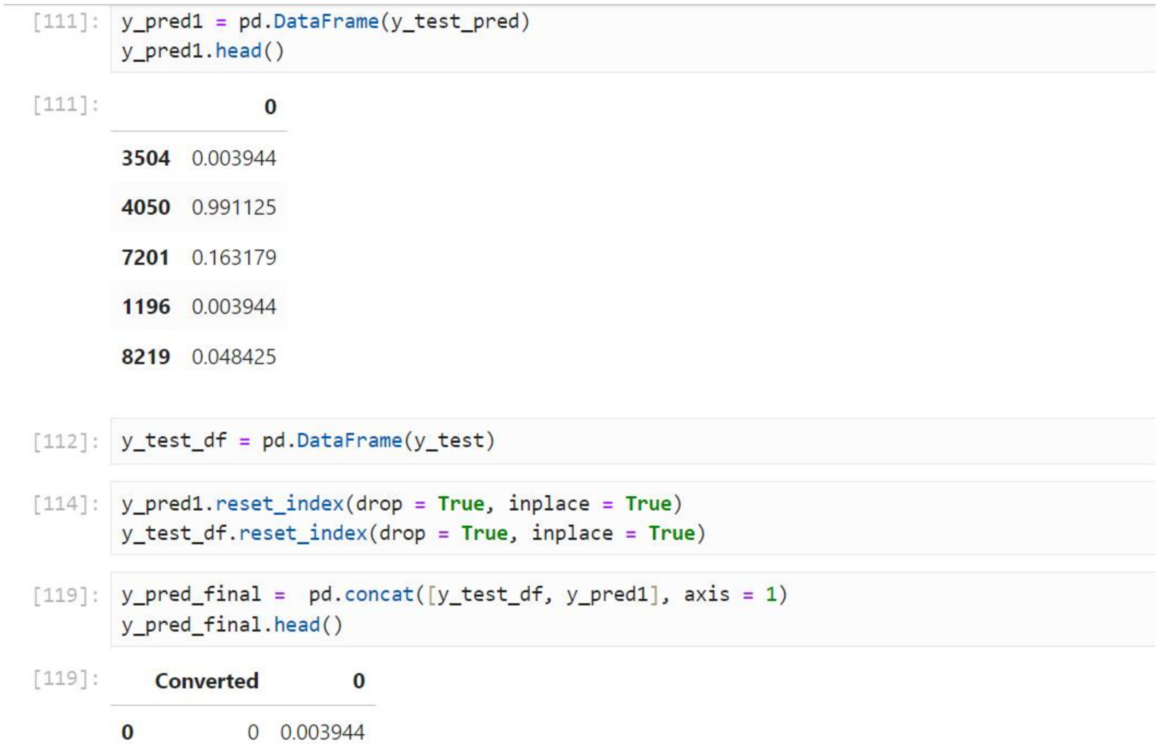
```
Shape of model's training data: (6372, 14)
Shape of test data: (2731, 13)
```

```
[108]: # Check for duplicated column names
       duplicated_columns = x_test_sm.columns[x_test_sm.columns.duplicated()]
       print(f"Duplicated columns: {duplicated_columns.tolist()}")
```

```
Duplicated columns: []
```

```
[110]: y_test_pred = res.predict(x_test_sm)
       print("First 10 predictions:", y_test_pred[:10])
```

```
First 10 predictions: 3504    0.003944
4050    0.991125
7201    0.163179
1196    0.003944
8219    0.048425
```



```
[111]: y_pred1 = pd.DataFrame(y_test_pred)
       y_pred1.head()
```

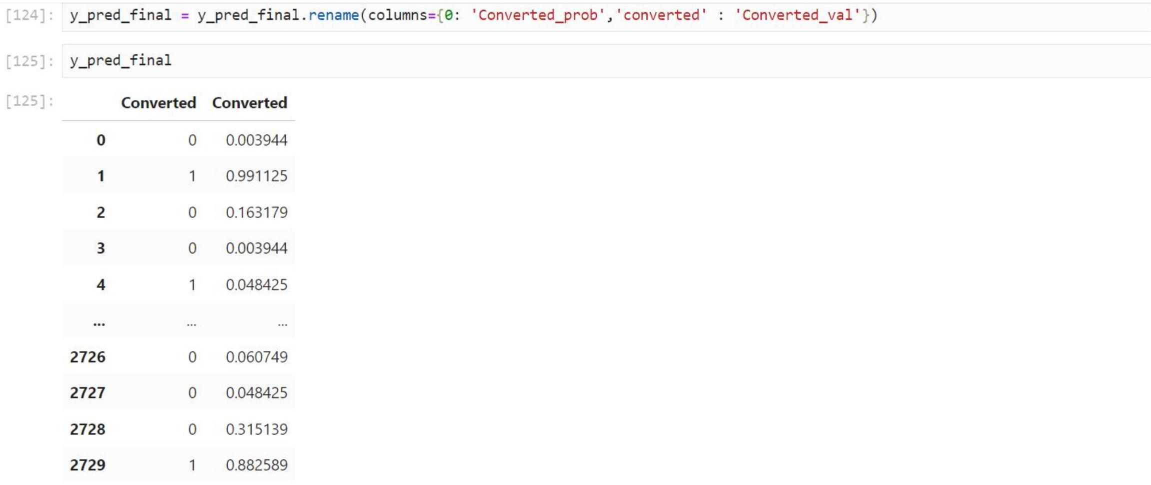
```
[111]:          0
       3504  0.003944
       4050  0.991125
       7201  0.163179
       1196  0.003944
       8219  0.048425
```

```
[112]: y_test_df = pd.DataFrame(y_test)
```

```
[114]: y_pred1.reset_index(drop = True, inplace = True)
       y_test_df.reset_index(drop = True, inplace = True)
```

```
[119]: y_pred_final =  pd.concat([y_test_df, y_pred1], axis = 1)
       y_pred_final.head()
```

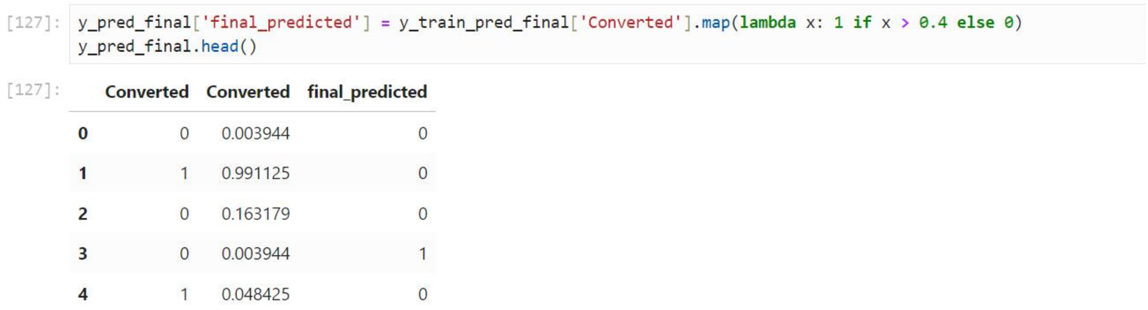
```
[119]:    Converted         0
       0          0  0.003944
```



```
[124]: y_pred_final = y_pred_final.rename(columns={0: 'Converted_prob','converted' : 'Converted_val'})
```

```
[125]: y_pred_final
```

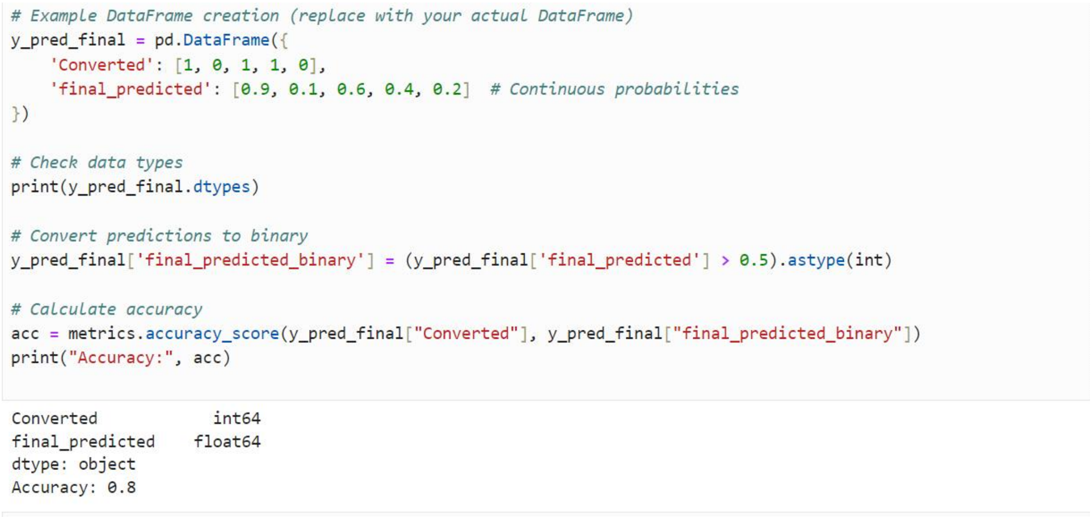
```
[125]:       Converted  Converted
        0            0   0.003944
        1            1   0.991125
        2            0   0.163179
        3            0   0.003944
        4            1   0.048425
      ...          ...        ...
     2726            0   0.060749
     2727            0   0.048425
     2728            0   0.315139
     2729            1   0.882589
```



```
[127]: y_pred_final['final_predicted'] = y_train_pred_final['Converted'].map(lambda x: 1 if x > 0.4 else 0)
       y_pred_final.head()
```

[127]:

	Converted	Converted	final_predicted
0	0	0.003944	0
1	1	0.991125	0
2	0	0.163179	0
3	0	0.003944	1
4	1	0.048425	0



```
# Example DataFrame creation (replace with your actual DataFrame)
y_pred_final = pd.DataFrame({
    'Converted': [1, 0, 1, 1, 0],
    'final_predicted': [0.9, 0.1, 0.6, 0.4, 0.2]  # Continuous probabilities
})

# Check data types
print(y_pred_final.dtypes)

# Convert predictions to binary
y_pred_final['final_predicted_binary'] = (y_pred_final['final_predicted'] > 0.5).astype(int)

# Calculate accuracy
acc = metrics.accuracy_score(y_pred_final["Converted"], y_pred_final["final_predicted_binary"])
print("Accuracy:", acc)
```

```
Converted            int64
final_predicted    float64
dtype: object
Accuracy: 0.8
```

Conclusion:

It was found that the variables that mattered the most in the potential buyers are (In descending order) :

◆ The total time spend on the Website.

◆ Total number of visits.

◆ When the lead source was:

a. Google

b. Direct traffic

c. Organic search

d. Welingak website

◆ When the last activity was:

a. SMS

b. Olark chat conversation

◆ When the lead origin is Lead add format.

◆ When their current occupation is as a working professional.

Keeping these in mind the X Education can flourish as they have a very high chance to get almost all the potential buyers to change their mind and buy their courses