LEAD SCORING CASE-STUDY

Submitted By :

1. Mohammad Aftab Alam
2. Praveen Kumar Natikar

Abstract

Problem Statement :

An education company named X Education sells online courses to industry professionals. The company markets its courses on several websites and search engines like Google.

Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos. When these people fill up a form providing their email address or phone number, they are classified to be a lead. Moreover, the company also gets leads through past referrals.

Once these leads are acquired, employees from the sales team start making calls, writing emails, etc. Through this process, some of the leads get converted while most do not. The typical lead conversion rate at X education is around 30%.

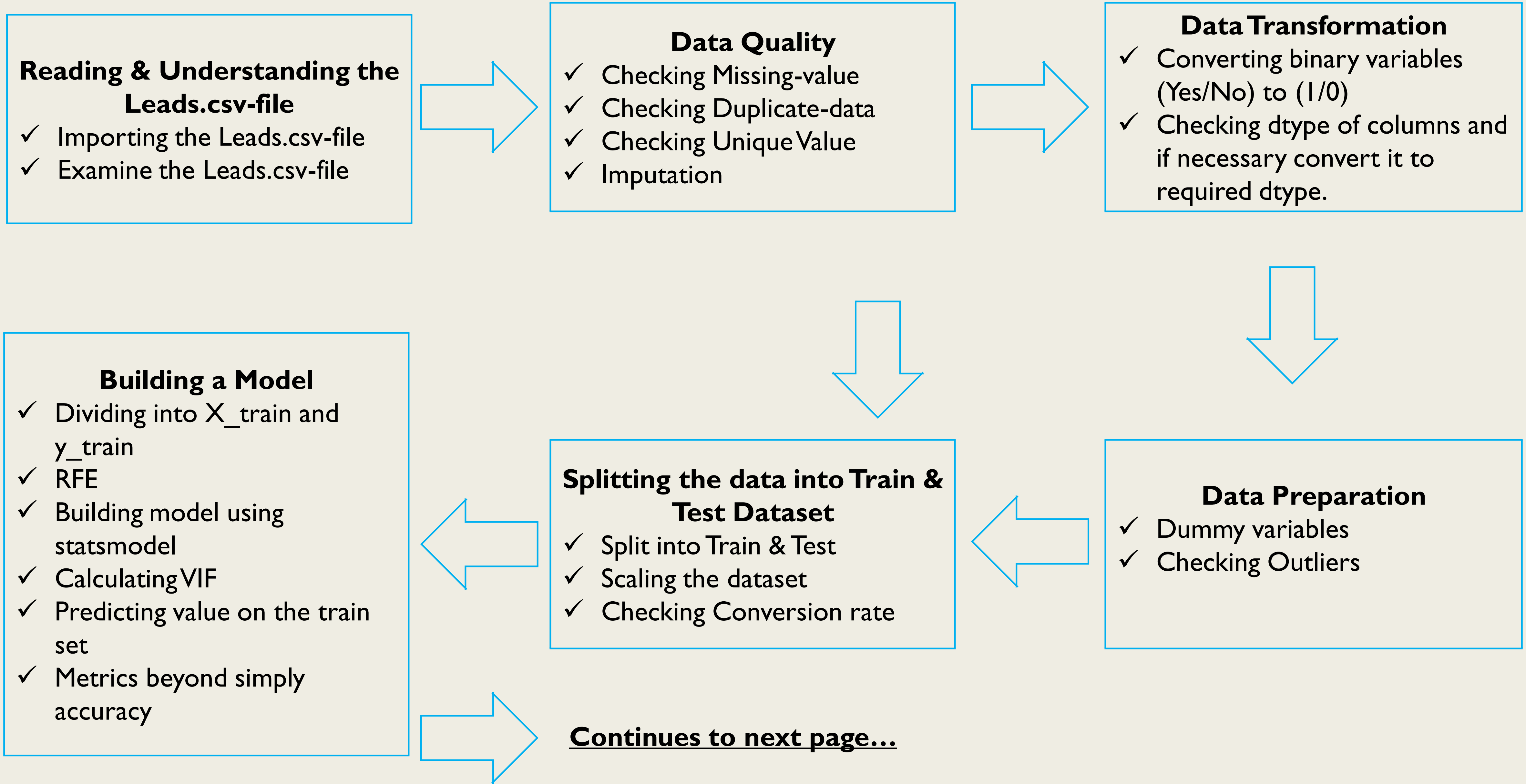
Business Goal :

X Education need help in selecting the most promising leads, i.e. the leads that are most likely to convert into paying customers.

The company need to build a model wherein you need to assign a lead score to each of the leads such that the customers with higher lead score have a higher conversion chance and the customers with lower lead score have a lower conversion chance.

The CEO, in particular, has given a ballpark of the target lead conversion rate to be around 80%.

Analysis Methodology



Reading & Understanding the Leads.csv-file

- ✓ Importing the Leads.csv-file
- ✓ Examine the Leads.csv-file

Data Quality

- ✓ Checking Missing-value
- ✓ Checking Duplicate-data
- ✓ Checking Unique Value
- ✓ Imputation

Data Transformation

- ✓ Converting binary variables (Yes/No) to (1/0)
- ✓ Checking dtype of columns and if necessary convert it to required dtype.

Data Preparation

- ✓ Dummy variables
- ✓ Checking Outliers

Splitting the data into Train & Test Dataset

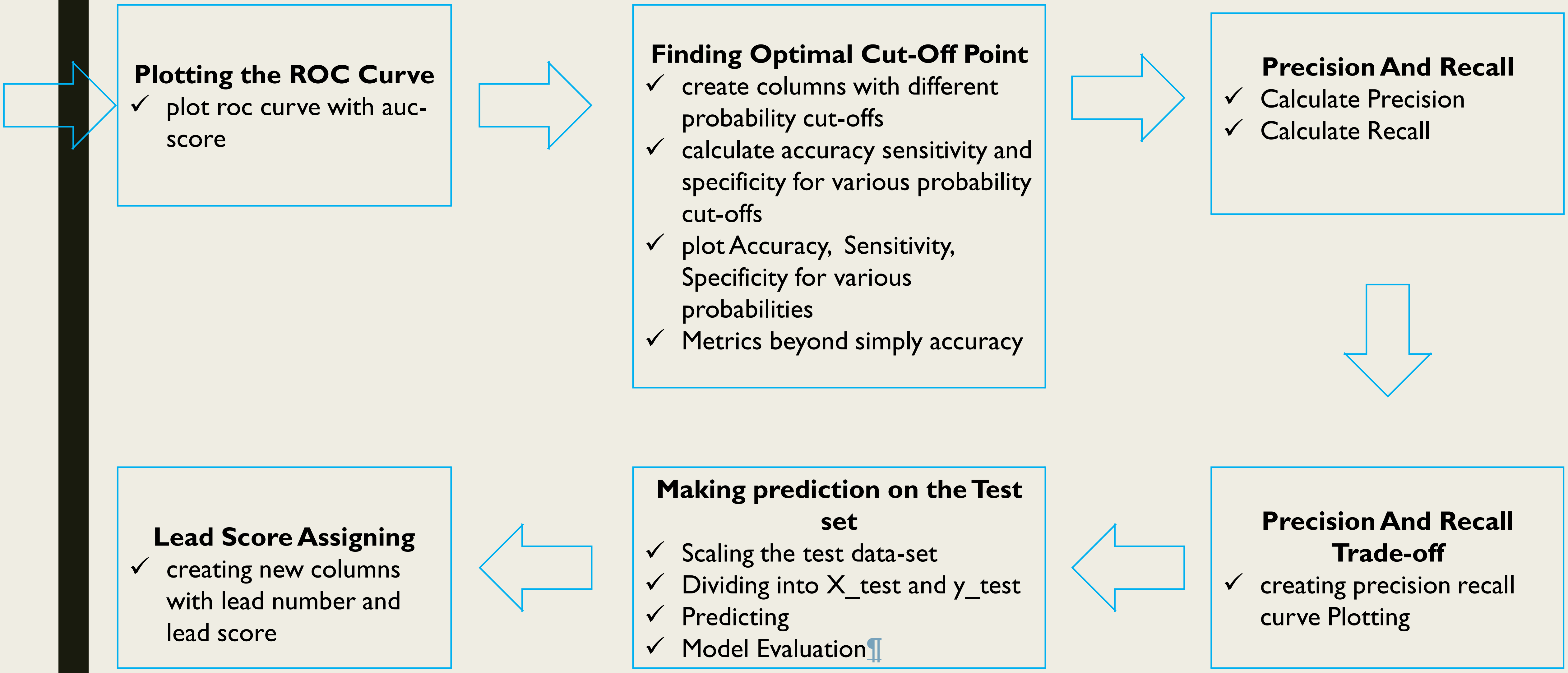
- ✓ Split into Train & Test
- ✓ Scaling the dataset
- ✓ Checking Conversion rate

Building a Model

- ✓ Dividing into X_train and y_train
- ✓ RFE
- ✓ Building model using statsmodel
- ✓ Calculating VIF
- ✓ Predicting value on the train set
- ✓ Metrics beyond simply accuracy

Continues to next page…

Analysis Methodology Cont…



Plotting the ROC Curve

- ✓ plot roc curve with auc-score

Finding Optimal Cut-Off Point

- ✓ create columns with different probability cut-offs
- ✓ calculate accuracy sensitivity and specificity for various probability cut-offs
- ✓ plot Accuracy, Sensitivity, Specificity for various probabilities
- ✓ Metrics beyond simply accuracy

Precision And Recall

- ✓ Calculate Precision
- ✓ Calculate Recall

Precision And Recall Trade-off

- ✓ creating precision recall curve Plotting

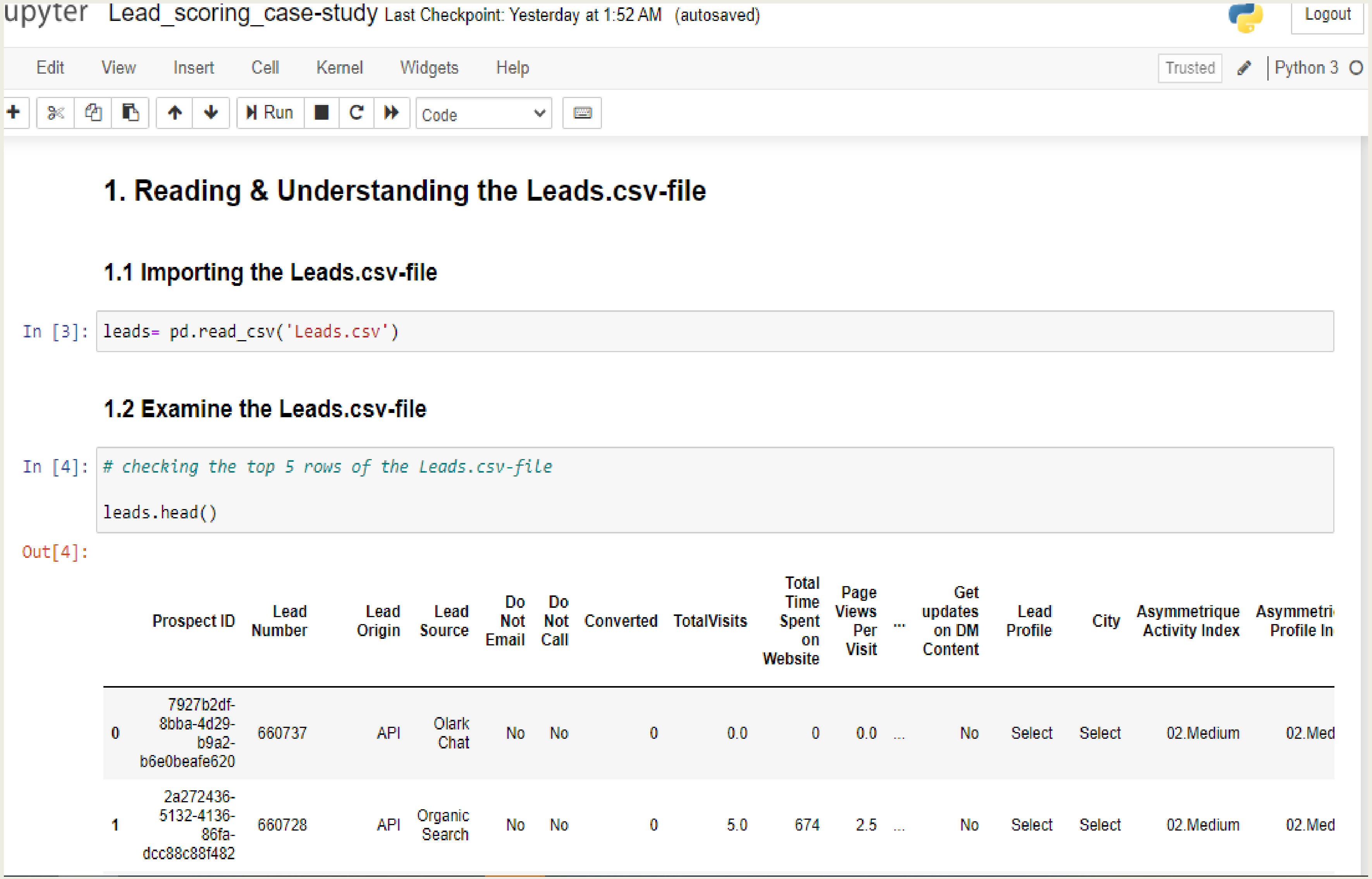
Making prediction on the Test set

- ✓ Scaling the test data-set
- ✓ Dividing into X_test and y_test
- ✓ Predicting
- ✓ Model Evaluation¶

Lead Score Assigning

- ✓ creating new columns with lead number and lead score

Reading & Understanding the



upyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Logout
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

1. Reading & Understanding the Leads.csv-file

1.1 Importing the Leads.csv-file

```
In [3]: leads= pd.read_csv('Leads.csv')
```

1.2 Examine the Leads.csv-file

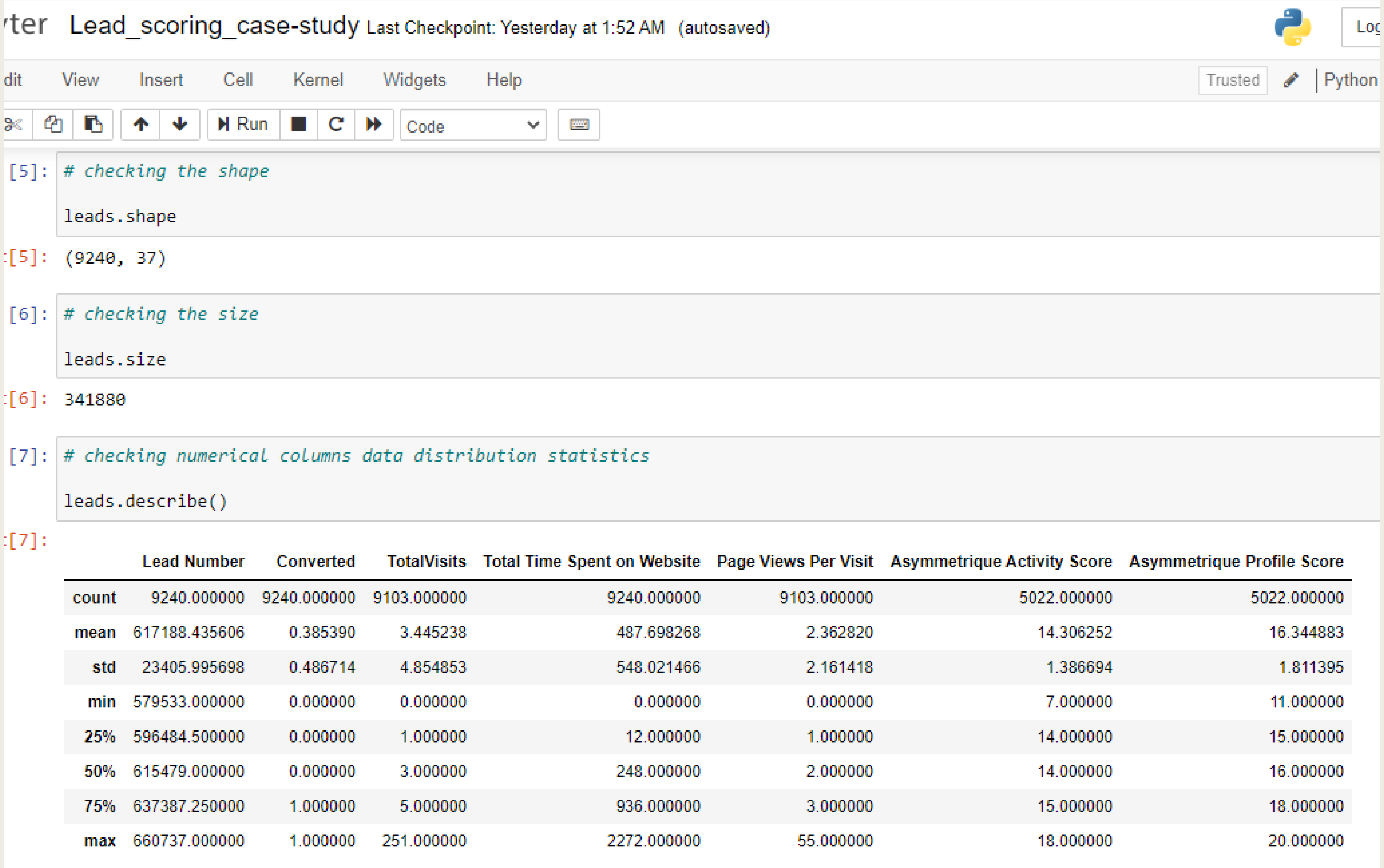
```
In [4]: # checking the top 5 rows of the Leads.csv-file

        leads.head()
```

Out[4]:

	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	Asymmetri Profile In
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	...	No	Select	Select	02.Medium	02.Med
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0	674	2.5	...	No	Select	Select	02.Medium	02.Med

- ✓ Importing the leads.csv-file.
- ✓ Examine the leads.csv-file

Reading & Understanding the Cont…



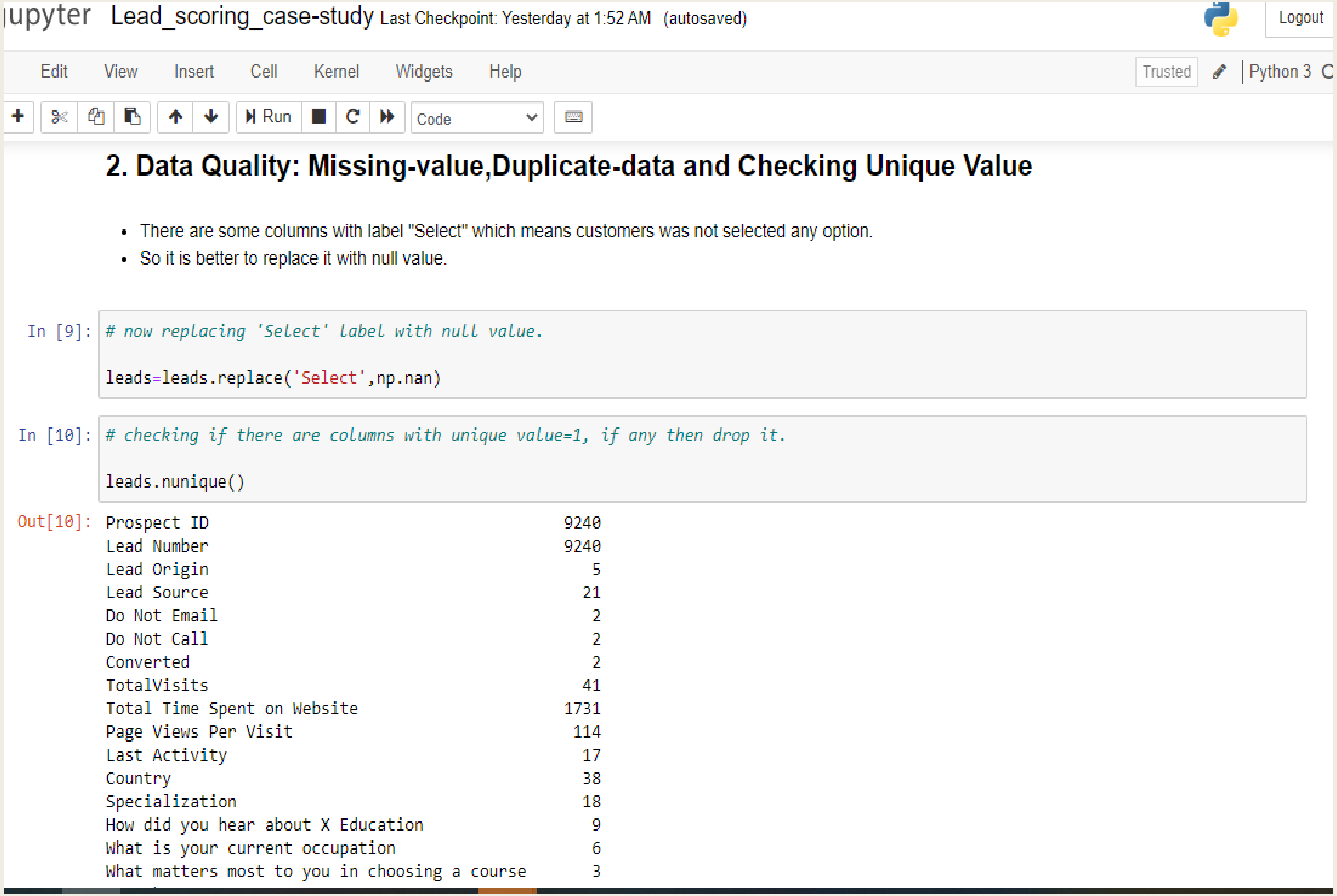
ter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Log
dit View Insert Cell Kernel Widgets Help
Trusted
Python
Run
Code

```
[5]: # checking the shape

     leads.shape
:[5]: (9240, 37)

[6]: # checking the size

     leads.size
:[6]: 341880

[7]: # checking numerical columns data distribution statistics

     leads.describe()
:[7]:
```

	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

- ✓ Checking shape
- ✓ Checking size
- ✓ Checking describe (numerical columns distribution)
- ✓ Checking info.

Data Quality



jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Logout
Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

2. Data Quality: Missing-value,Duplicate-data and Checking Unique Value

- There are some columns with label "Select" which means customers was not selected any option.
- So it is better to replace it with null value.

```
In [9]:  # now replacing 'Select' label with null value.

         leads=leads.replace('Select',np.nan)
```

```
In [10]: # checking if there are columns with unique value=1, if any then drop it.

         leads.nunique()
```

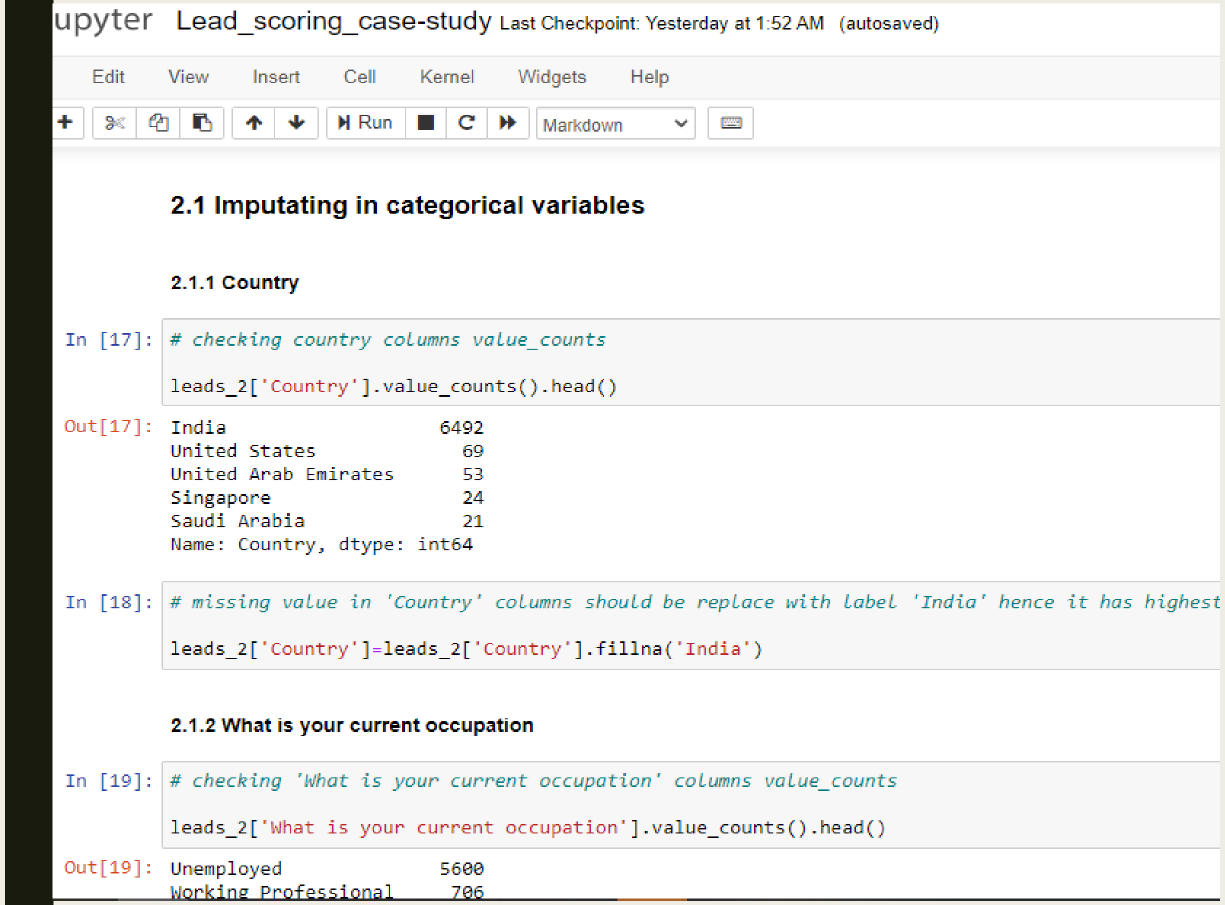
```
Out[10]: Prospect ID                                      9240
         Lead Number                                      9240
         Lead Origin                                         5
         Lead Source                                        21
         Do Not Email                                        2
         Do Not Call                                         2
         Converted                                           2
         TotalVisits                                        41
         Total Time Spent on Website                      1731
         Page Views Per Visit                              114
         Last Activity                                      17
         Country                                            38
         Specialization                                     18
         How did you hear about X Education                  9
         What is your current occupation                     6
         What matters most to you in choosing a course       3
```

- ✓ replacing 'Select' label with null value.
- ✓ checking if there are columns with unique value=1, if any then drop it

Conclusion

After seeing above we find that there are some columns with unique value = 1 are named as 'Magazine', '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'

Data Quality Cont…



upyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Edit View Insert Cell Kernel Widgets Help
Run
Markdown

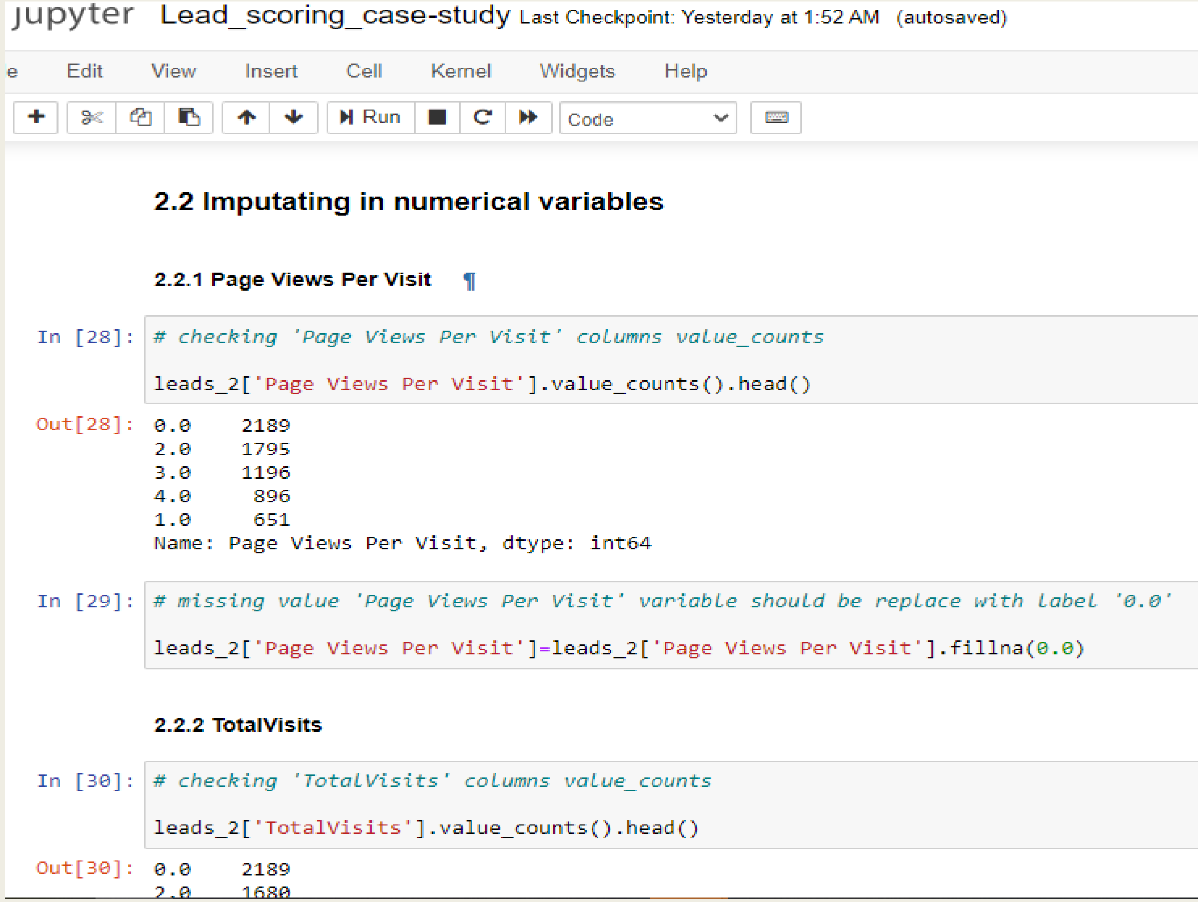
2.1 Imputating in categorical variables

2.1.1 Country

```
In [17]: # checking country columns value_counts

         leads_2['Country'].value_counts().head()

Out[17]: India                   6492
         United States             69
         United Arab Emirates      53
         Singapore                 24
         Saudi Arabia              21
         Name: Country, dtype: int64

In [18]: # missing value in 'Country' columns should be replace with label 'India' hence it has highest

         leads_2['Country']=leads_2['Country'].fillna('India')
```

2.1.2 What is your current occupation

```
In [19]: # checking 'What is your current occupation' columns value_counts

         leads_2['What is your current occupation'].value_counts().head()

Out[19]: Unemployed              5600
         Working Professional     706
```



jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
e Edit View Insert Cell Kernel Widgets Help
Run
Code

2.2 Imputating in numerical variables

2.2.1 Page Views Per Visit ¶

```
In [28]: # checking 'Page Views Per Visit' columns value_counts

         leads_2['Page Views Per Visit'].value_counts().head()

Out[28]: 0.0    2189
         2.0    1795
         3.0    1196
         4.0     896
         1.0     651
         Name: Page Views Per Visit, dtype: int64

In [29]: # missing value 'Page Views Per Visit' variable should be replace with label '0.0'

         leads_2['Page Views Per Visit']=leads_2['Page Views Per Visit'].fillna(0.0)
```

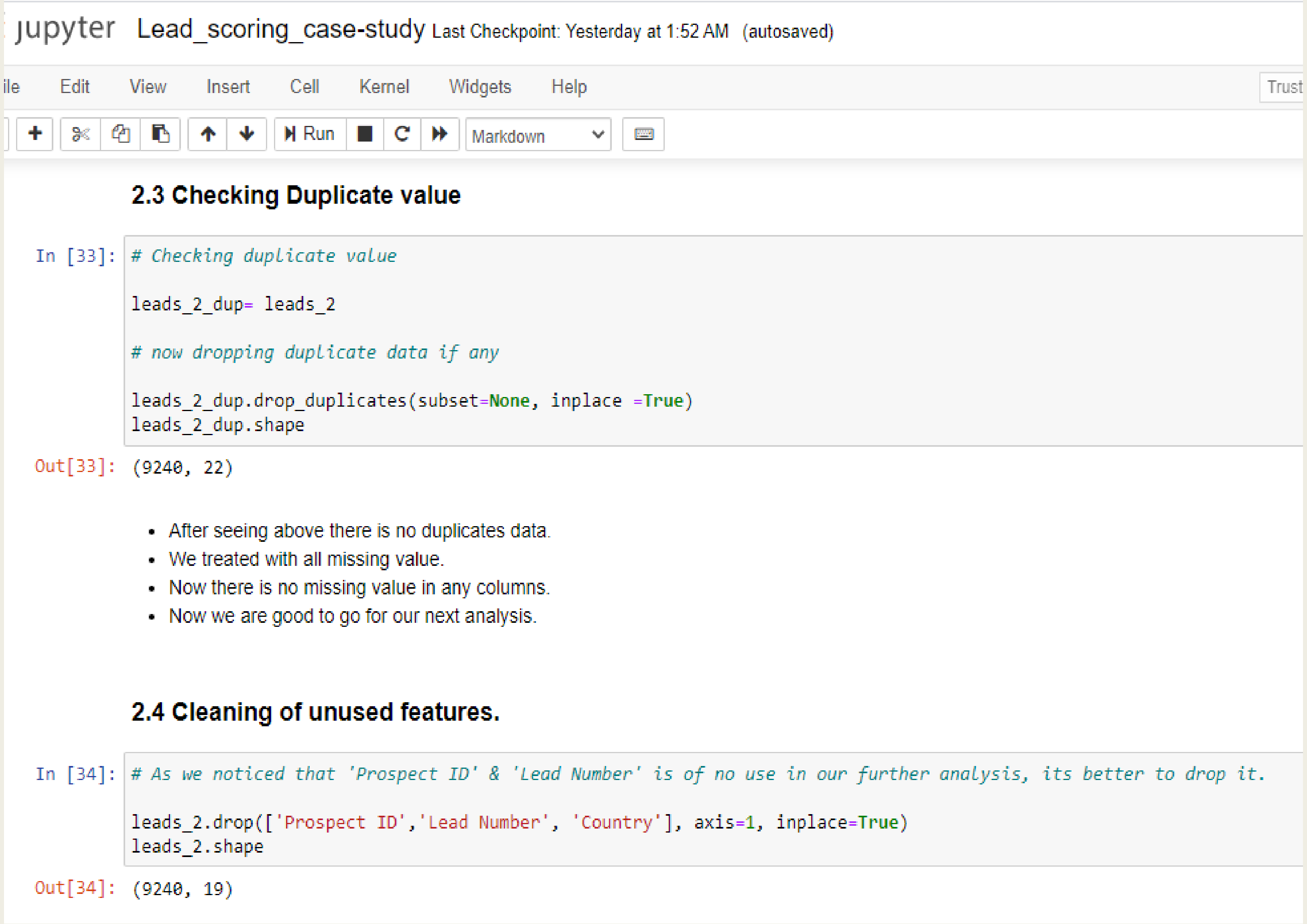
2.2.2 TotalVisits

```
In [30]: # checking 'TotalVisits' columns value_counts

         leads_2['TotalVisits'].value_counts().head()

Out[30]: 0.0    2189
         2.0    1680
```

Imputating in categorical variables

Imputating in categorical variables

Data Quality Cont…


jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
ile Edit View Insert Cell Kernel Widgets Help
Trust
Run
Markdown

2.3 Checking Duplicate value

In [33]:
```
# Checking duplicate value

leads_2_dup= leads_2

# now dropping duplicate data if any

leads_2_dup.drop_duplicates(subset=None, inplace =True)
leads_2_dup.shape
```

Out[33]:
```
(9240, 22)
```

- After seeing above there is no duplicates data.
- We treated with all missing value.
- Now there is no missing value in any columns.
- Now we are good to go for our next analysis.

2.4 Cleaning of unused features.

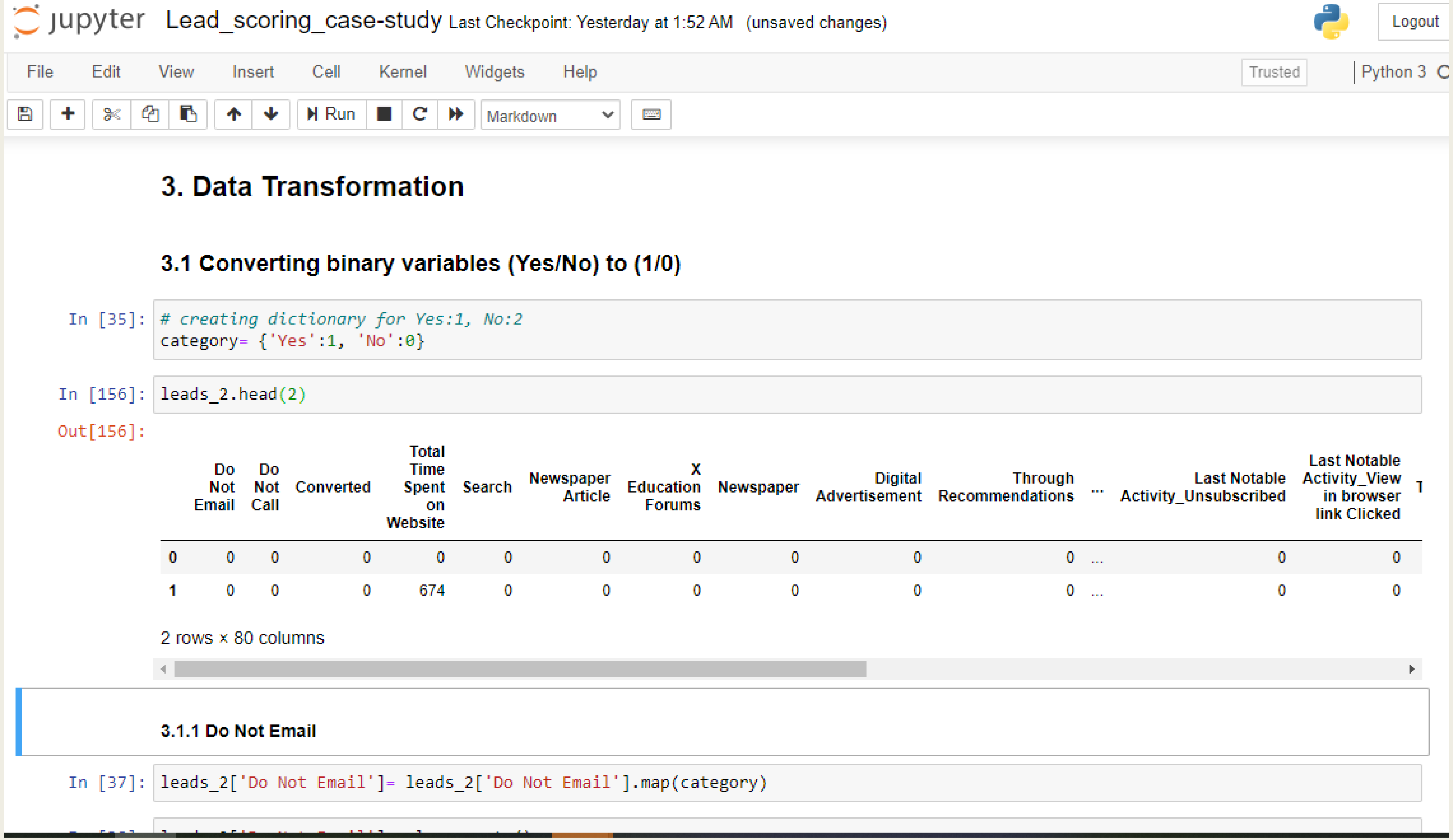
In [34]:
```
# As we noticed that 'Prospect ID' & 'Lead Number' is of no use in our further analysis, its better to drop it.

leads_2.drop(['Prospect ID','Lead Number', 'Country'], axis=1, inplace=True)
leads_2.shape
```

Out[34]:
```
(9240, 19)
```

- ✓ There is no duplicates data.
- ✓ We treated with all missing value.
- ✓ Now there is no missing value in any columns.
- ✓ Now we are good to go for our next analysis.

Checking Duplicate value and Cleaning of unused features

Data Transformation



jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Markdown

3. Data Transformation

3.1 Converting binary variables (Yes/No) to (1/0)

```
In [35]: # creating dictionary for Yes:1, No:2
         category= {'Yes':1, 'No':0}
```

```
In [156]: leads_2.head(2)
```

Out[156]:

	Do Not Email	Do Not Call	Converted	Total Time Spent on Website	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	Through Recommendations	...	Last Notable Activity_Unsubscribed	Last Notable Activity_View in browser link Clicked	T
0	0	0	0	0	0	0	0	0	0	0	...	0	0	
1	0	0	0	674	0	0	0	0	0	0	...	0	0	

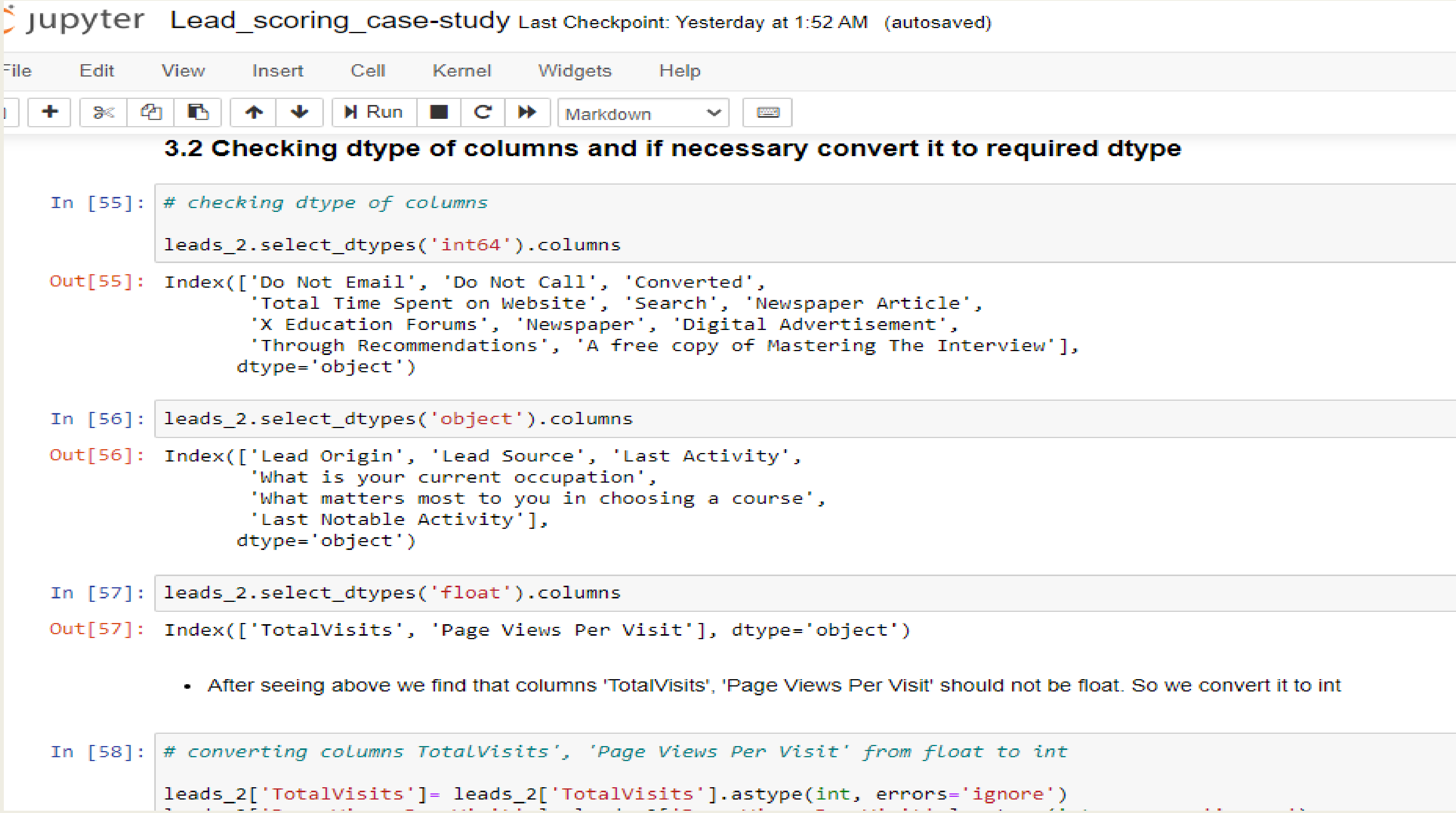
2 rows × 80 columns

3.1.1 Do Not Email

```
In [37]: leads_2['Do Not Email']= leads_2['Do Not Email'].map(category)
```

Converting binary variables (Yes/No) to (1/0)

Data Transformation Cont…


jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Markdown

3.2 Checking dtype of columns and if necessary convert it to required dtype

```
In [55]: # checking dtype of columns

         leads_2.select_dtypes('int64').columns

Out[55]: Index(['Do Not Email', 'Do Not Call', 'Converted',
                'Total Time Spent on Website', 'Search', 'Newspaper Article',
                'X Education Forums', 'Newspaper', 'Digital Advertisement',
                'Through Recommendations', 'A free copy of Mastering The Interview'],
               dtype='object')

In [56]: leads_2.select_dtypes('object').columns

Out[56]: Index(['Lead Origin', 'Lead Source', 'Last Activity',
                'What is your current occupation',
                'What matters most to you in choosing a course',
                'Last Notable Activity'],
               dtype='object')

In [57]: leads_2.select_dtypes('float').columns

Out[57]: Index(['TotalVisits', 'Page Views Per Visit'], dtype='object')
```

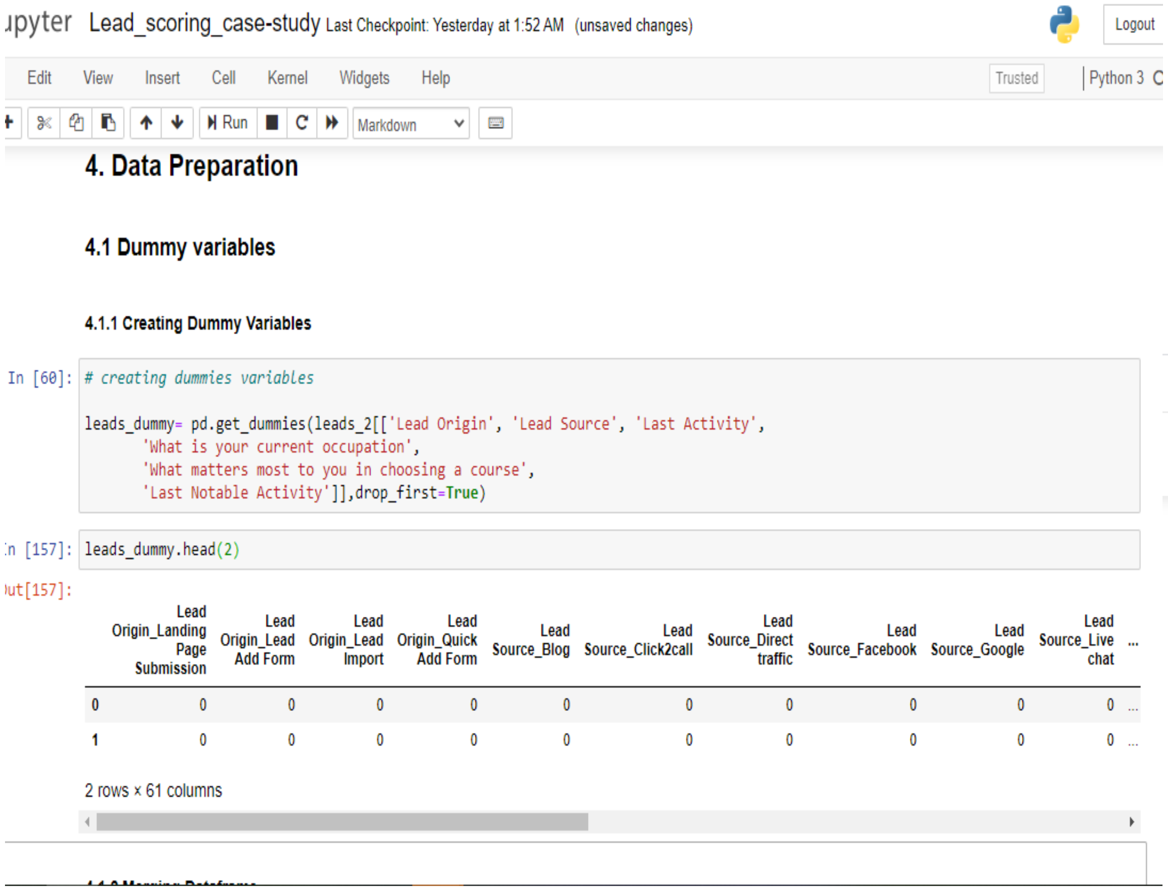
- After seeing above we find that columns 'TotalVisits', 'Page Views Per Visit' should not be float. So we convert it to int

```
In [58]: # converting columns TotalVisits', 'Page Views Per Visit' from float to int

         leads_2['TotalVisits']= leads_2['TotalVisits'].astype(int, errors='ignore')
```

Checking dtype of columns and if necessary convert it to required dtype

Data Preparation

Creating Dummy Variables



jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (unsaved changes)
Logout
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Markdown

4. Data Preparation

4.1 Dummy variables

4.1.1 Creating Dummy Variables

```
In [60]: # creating dummies variables

         leads_dummy= pd.get_dummies(leads_2[['Lead Origin', 'Lead Source', 'Last Activity',
                'What is your current occupation',
                'What matters most to you in choosing a course',
                'Last Notable Activity']],drop_first=True)
```

```
In [157]: leads_dummy.head(2)
Out[157]:
```

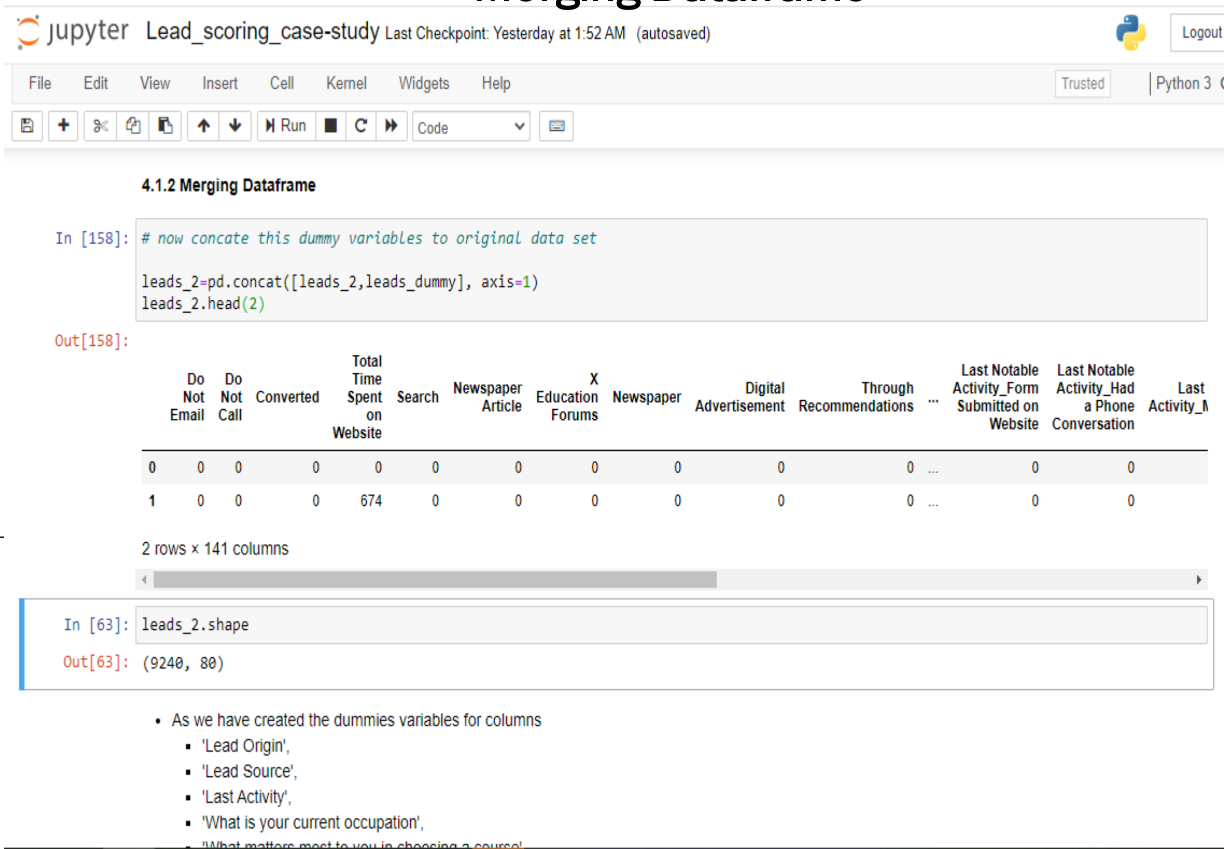
	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Origin_Quick Add Form	Lead Source_Blog	Lead Source_Click2call	Lead Source_Direct traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live chat	...
0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	...

2 rows × 61 columns

4.1.2 Merging Dataframe

As we have created the dummies variables for columns 'Lead Origin', 'Lead Source', 'Last Activity', 'What is your current occupation', 'What matters most to you in choosing a course', 'Last Notable Activity'

Merging Dataframe



4.1.2 Merging Dataframe

```
In [158]: # now concate this dummy variables to original data set

          leads_2=pd.concat([leads_2,leads_dummy], axis=1)
          leads_2.head(2)
Out[158]:
```

	Do Not Email	Do Not Call	Converted	Total Time Spent on Website	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	Through Recommendations	...	Last Notable Activity_Form Submitted on Website	Last Notable Activity_Had a Phone Conversation	Last Activity_M
0	0	0	0	0	0	0	0	0	0	0	...	0	0	
1	0	0	0	674	0	0	0	0	0	0	...	0	0	

2 rows × 141 columns

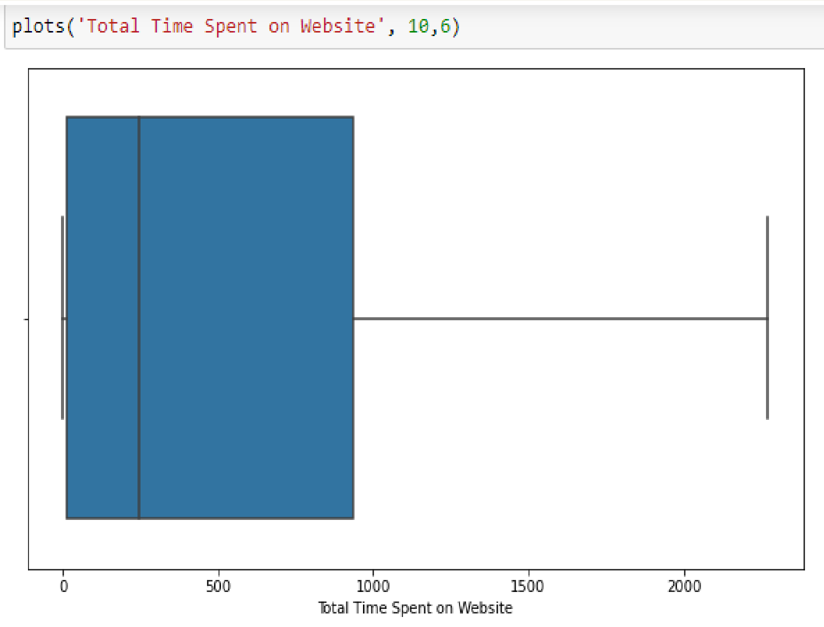
```
In [63]: leads_2.shape
Out[63]: (9240, 80)
```

- As we have created the dummies variables for columns
  - 'Lead Origin',
  - 'Lead Source',
  - 'Last Activity',
  - 'What is your current occupation',
  - 'What matters most to you in choosing a course'

And after that merge it with the original data and remove this above original columns as it is of no use.

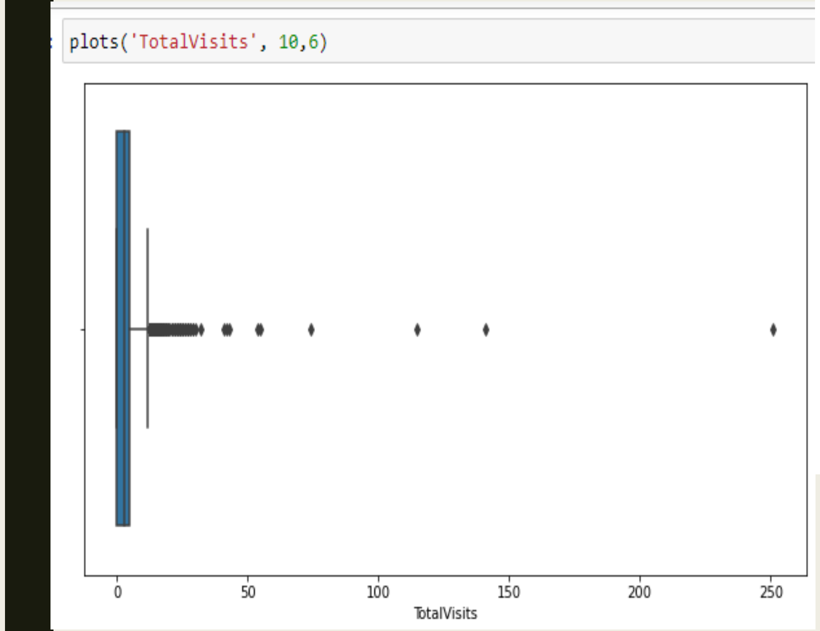
Data Preparation Cont…

Checking outliers by Plotting Boxplot



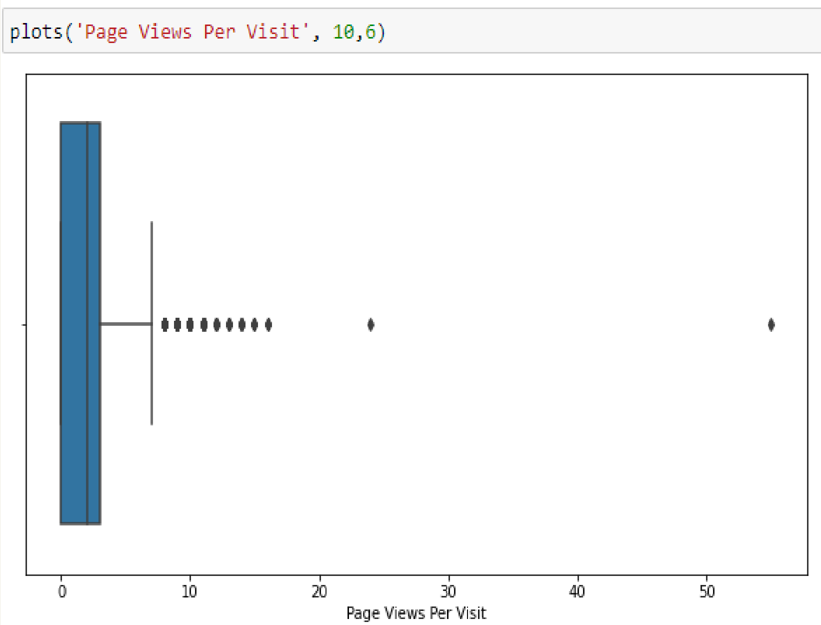
```
plots('Total Time Spent on Website', 10,6)
```

'Total Time Spent on Website'



```
plots('TotalVisits', 10,6)
```

'TotalVisits'



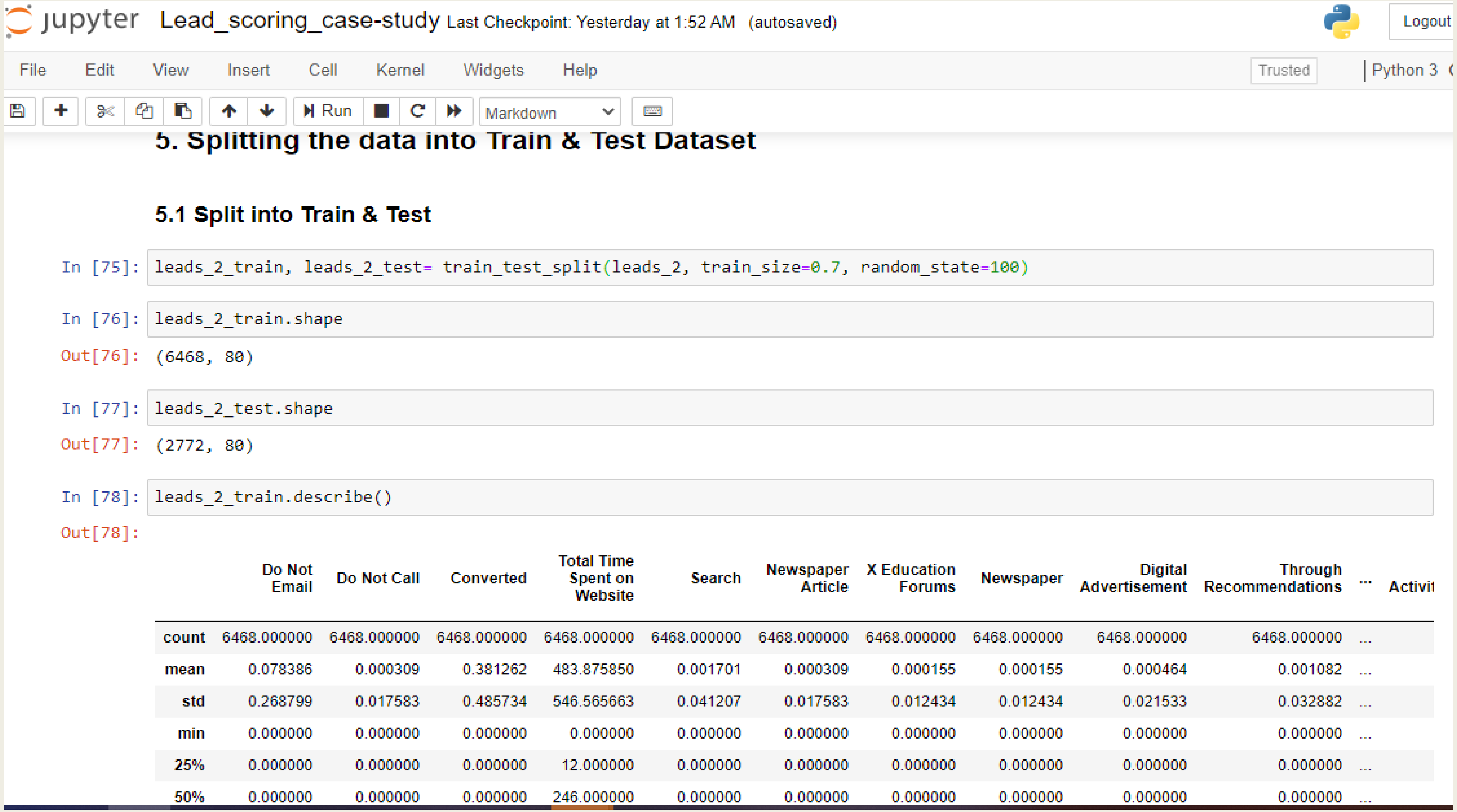
```
plots('Page Views Per Visit', 10,6)
```

'Page Views per Visit'

- ✓ After seeing boxplot we can clearly find that there is 2 outliers variables that's 'TotalVisits' and 'Page Views per Visit'.
- ✓ But as per business requirement we cannot drop these outliers as it may impact our analysis/model.
- ✓ So we will create bins for these outliers.

Splitting the data into Train & Test Dataset

Split into Train & Test



5. Splitting the data into Train & Test Dataset

5.1 Split into Train & Test

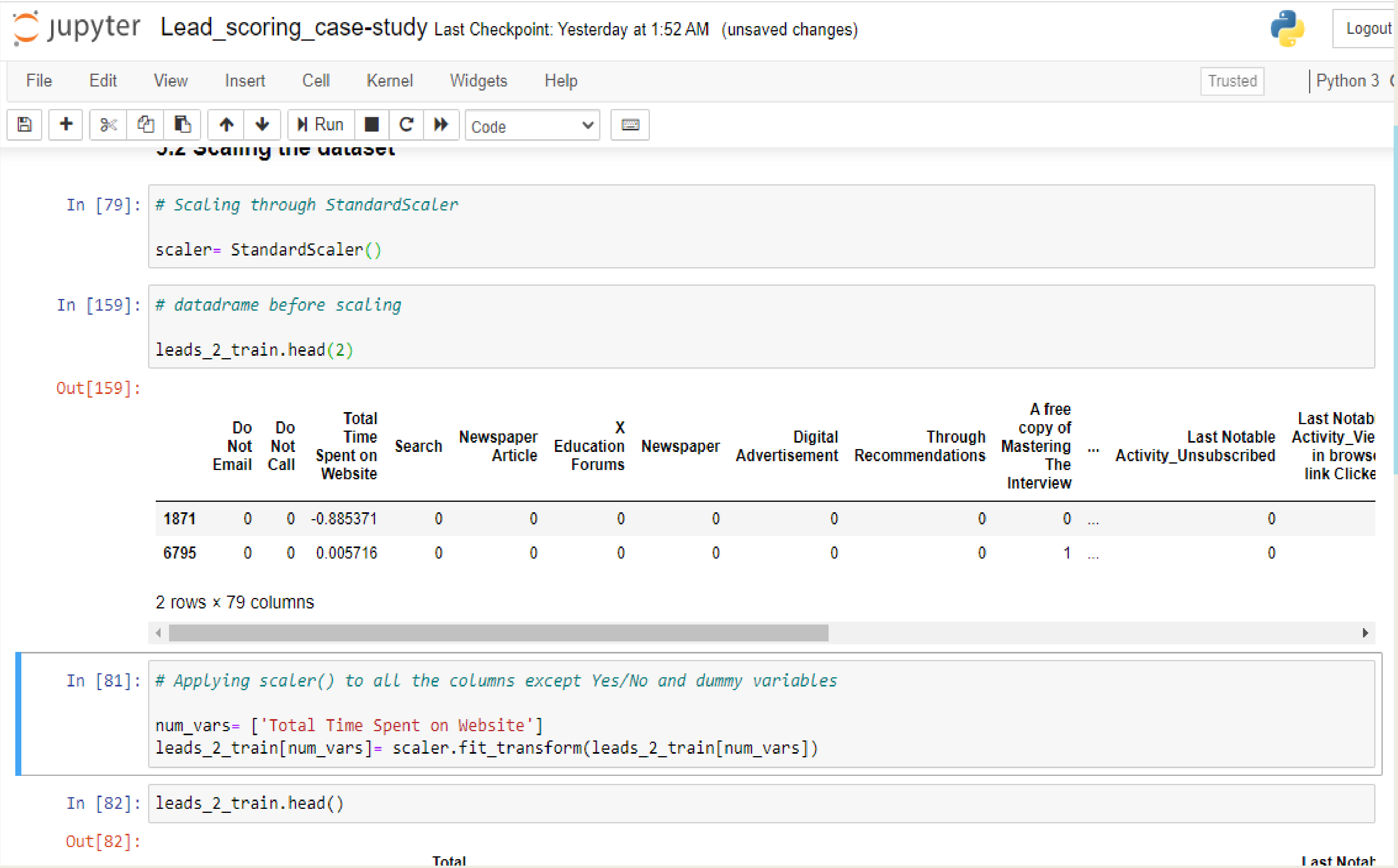
```
In [75]: leads_2_train, leads_2_test= train_test_split(leads_2, train_size=0.7, random_state=100)

In [76]: leads_2_train.shape

Out[76]: (6468, 80)

In [77]: leads_2_test.shape

Out[77]: (2772, 80)

In [78]: leads_2_train.describe()

Out[78]:
```

	Do Not Email	Do Not Call	Converted	Total Time Spent on Website	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	Through Recommendations	...	Activit
count	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	6468.000000	...	
mean	0.078386	0.000309	0.381262	483.875850	0.001701	0.000309	0.000155	0.000155	0.000464	0.001082	...	
std	0.268799	0.017583	0.485734	546.565663	0.041207	0.017583	0.012434	0.012434	0.021533	0.032882	...	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	
25%	0.000000	0.000000	0.000000	12.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	
50%	0.000000	0.000000	0.000000	246.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	

✓ Based on the split (70% - 30%) between train and test dataset we got 6468 rows in train dataset and 2772 rows in test dataset.

Splitting the data into Train & Test Dataset Cont…

Scaling the dataset


jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

5.2 Scaling the dataset

```
In [79]: # Scaling through StandardScaler

         scaler= StandardScaler()
```

```
In [159]: # datadrame before scaling

          leads_2_train.head(2)
```

Out[159]:

	Do Not Email	Do Not Call	Total Time Spent on Website	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	Through Recommendations	A free copy of Mastering The Interview	...	Last Notable Activity_Unsubscribed	Last Notab Activity_Vie in brows link Clicke
1871	0	0	-0.885371	0	0	0	0	0	0	0	...	0	
6795	0	0	0.005716	0	0	0	0	0	0	1	...	0	

2 rows × 79 columns

```
In [81]: # Applying scaler() to all the columns except Yes/No and dummy variables

         num_vars= ['Total Time Spent on Website']
         leads_2_train[num_vars]= scaler.fit_transform(leads_2_train[num_vars])
```

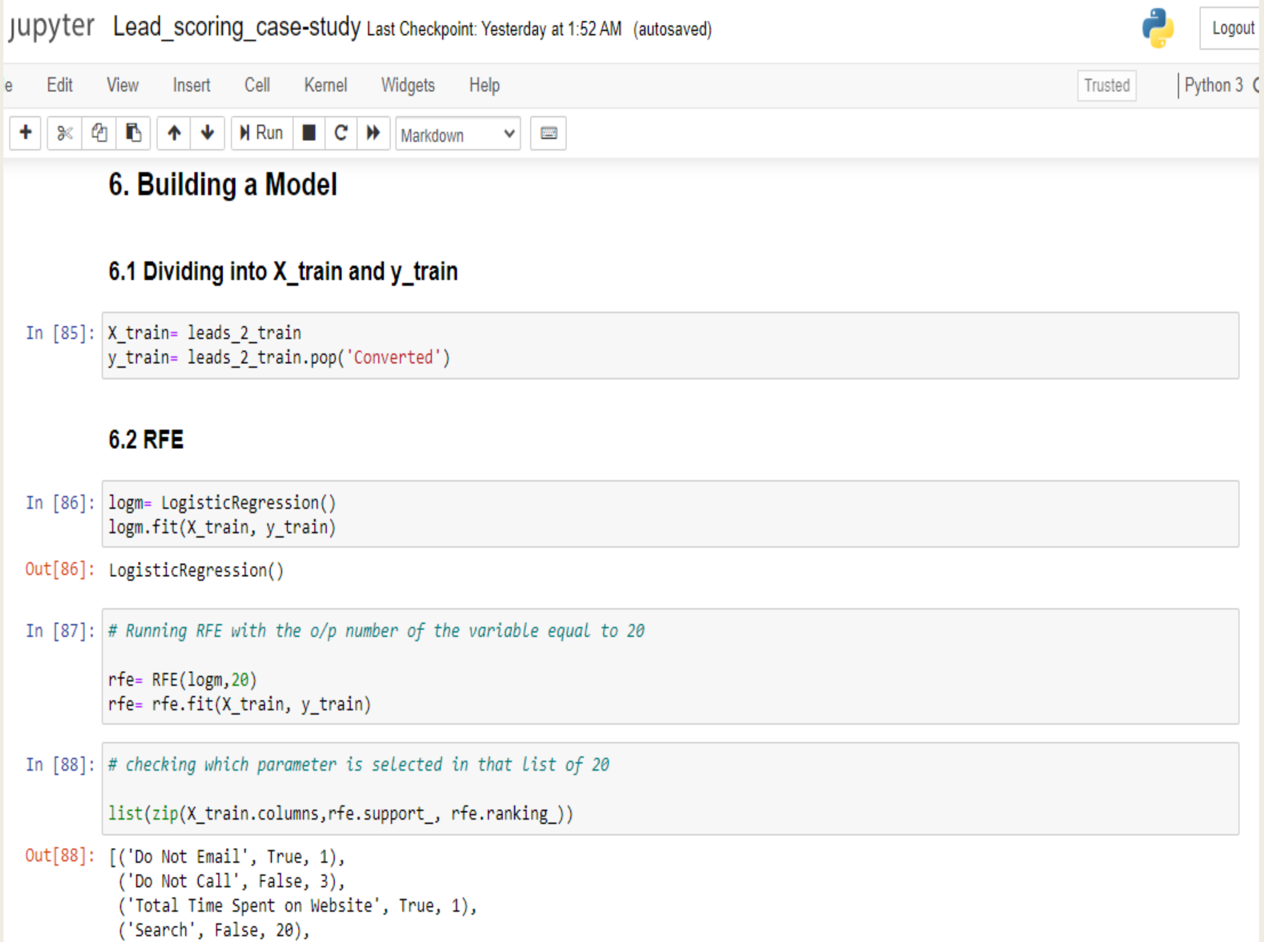
```
In [82]: leads_2_train.head()
```

Out[82]:

Total Last Notab

- ✓ Scaling through StandardScale
- ✓ Applying scaler() to all the columns except Yes/No and dummy variables
- ✓ After that we should Check the Conversion rate and we get almost 39% of conversion rate

Building a Model


jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Logout
e Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

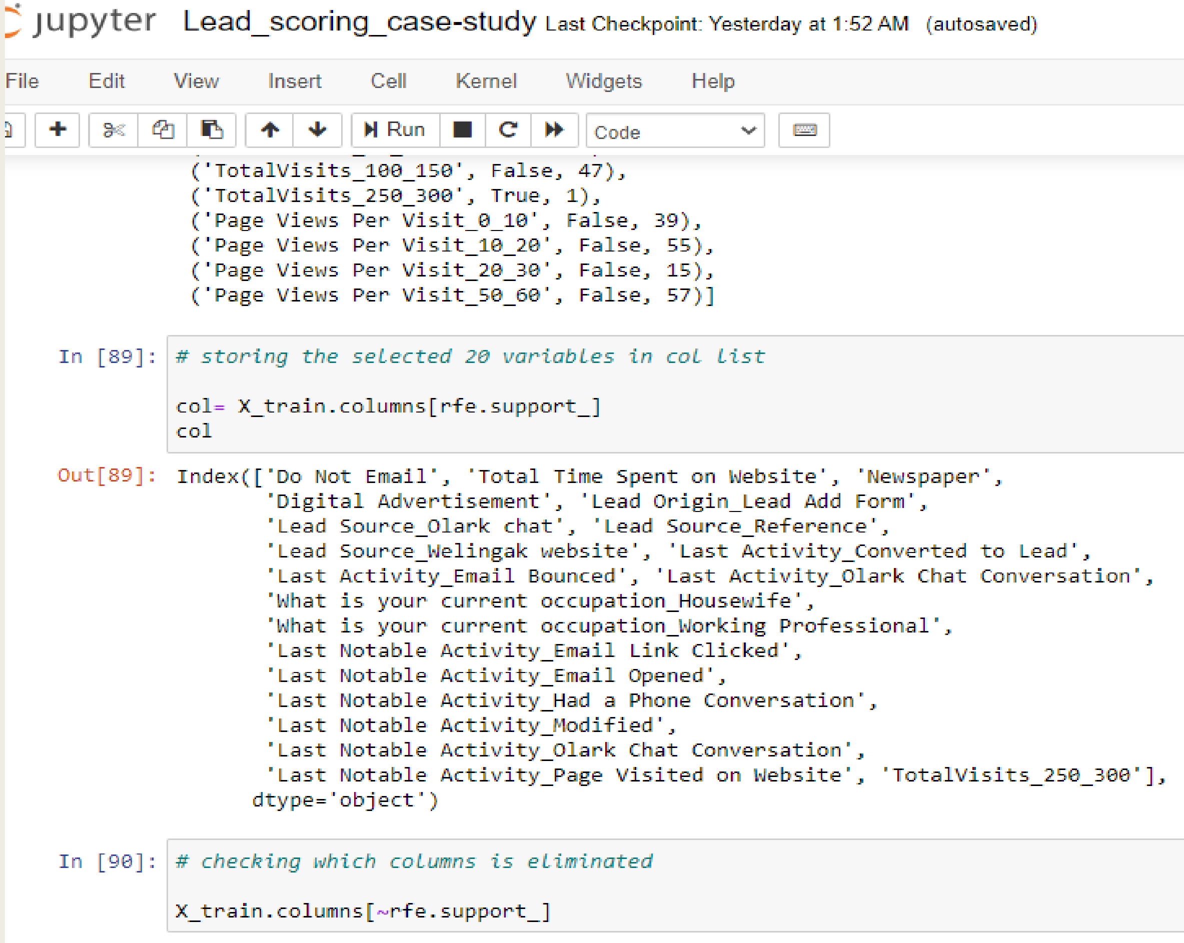
6. Building a Model

6.1 Dividing into X_train and y_train

```
In [85]: X_train= leads_2_train
         y_train= leads_2_train.pop('Converted')
```

6.2 RFE

```
In [86]: logm= LogisticRegression()
         logm.fit(X_train, y_train)

Out[86]: LogisticRegression()

In [87]: # Running RFE with the o/p number of the variable equal to 20

         rfe= RFE(logm,20)
         rfe= rfe.fit(X_train, y_train)

In [88]: # checking which parameter is selected in that list of 20

         list(zip(X_train.columns,rfe.support_, rfe.ranking_))

Out[88]: [('Do Not Email', True, 1),
          ('Do Not Call', False, 3),
          ('Total Time Spent on Website', True, 1),
          ('Search', False, 20),
```


jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Code

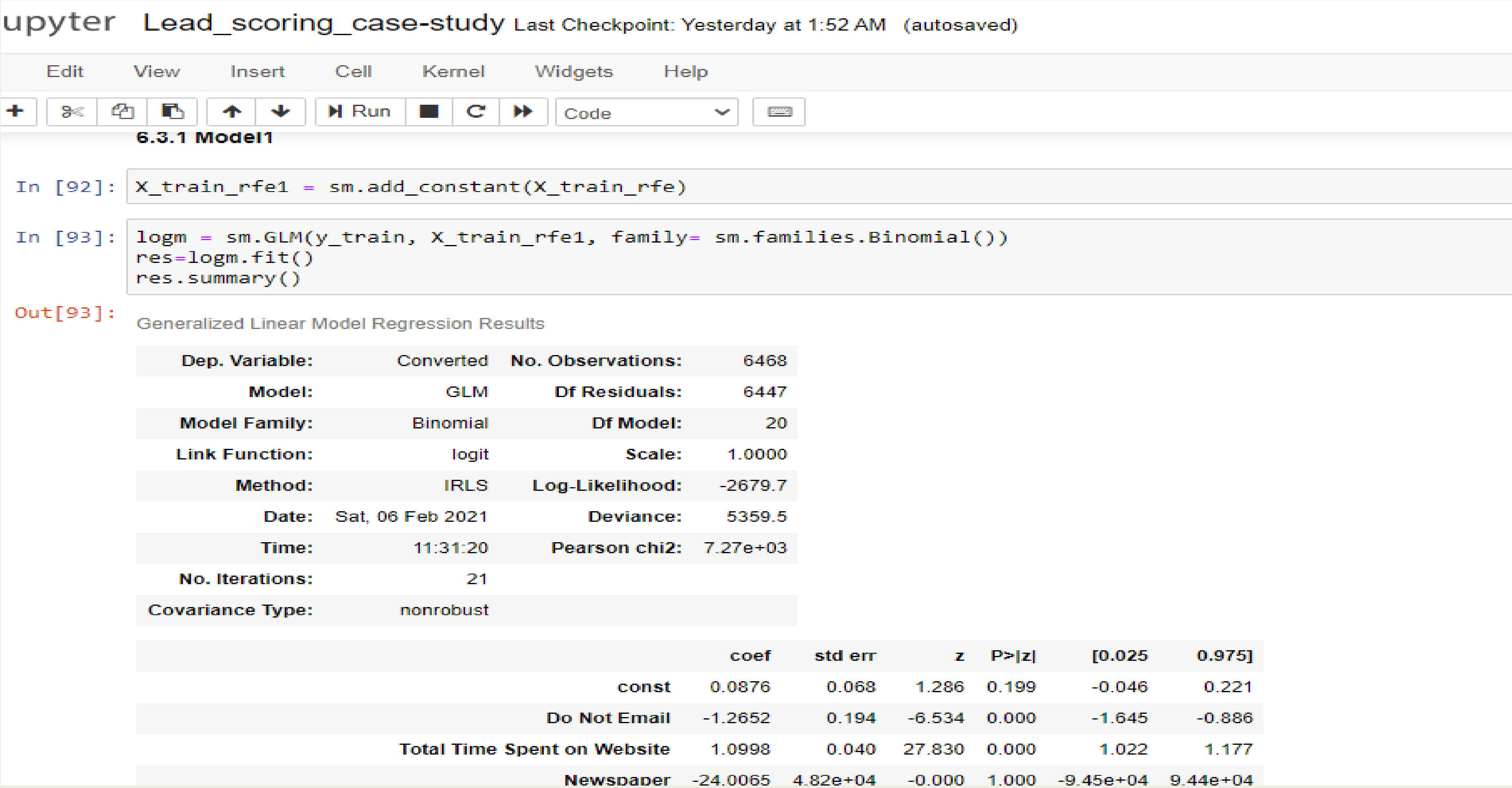
```
          ('TotalVisits_100_150', False, 47),
          ('TotalVisits_250_300', True, 1),
          ('Page Views Per Visit_0_10', False, 39),
          ('Page Views Per Visit_10_20', False, 55),
          ('Page Views Per Visit_20_30', False, 15),
          ('Page Views Per Visit_50_60', False, 57)]

In [89]: # storing the selected 20 variables in col list

         col= X_train.columns[rfe.support_]
         col

Out[89]: Index(['Do Not Email', 'Total Time Spent on Website', 'Newspaper',
                'Digital Advertisement', 'Lead Origin_Lead Add Form',
                'Lead Source_Olark chat', 'Lead Source_Reference',
                'Lead Source_Welingak website', 'Last Activity_Converted to Lead',
                'Last Activity_Email Bounced', 'Last Activity_Olark Chat Conversation',
                'What is your current occupation_Housewife',
                'What is your current occupation_Working Professional',
                'Last Notable Activity_Email Link Clicked',
                'Last Notable Activity_Email Opened',
                'Last Notable Activity_Had a Phone Conversation',
                'Last Notable Activity_Modified',
                'Last Notable Activity_Olark Chat Conversation',
                'Last Notable Activity_Page Visited on Website', 'TotalVisits_250_300'],
               dtype='object')

In [90]: # checking which columns is eliminated

         X_train.columns[~rfe.support_]
```

- ✓ Dividing into X_train and y_train
- ✓ Running RFE with the o/p number of the variable equal to 20.
- ✓ checking which parameter is selected in that list of 20.
- ✓ storing the selected 20 variables in col list.
- ✓ creating X_train dataframe with RFE selected Variables

Building a Model Cont…

Model I

Generalized Linear Model Regression



Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)

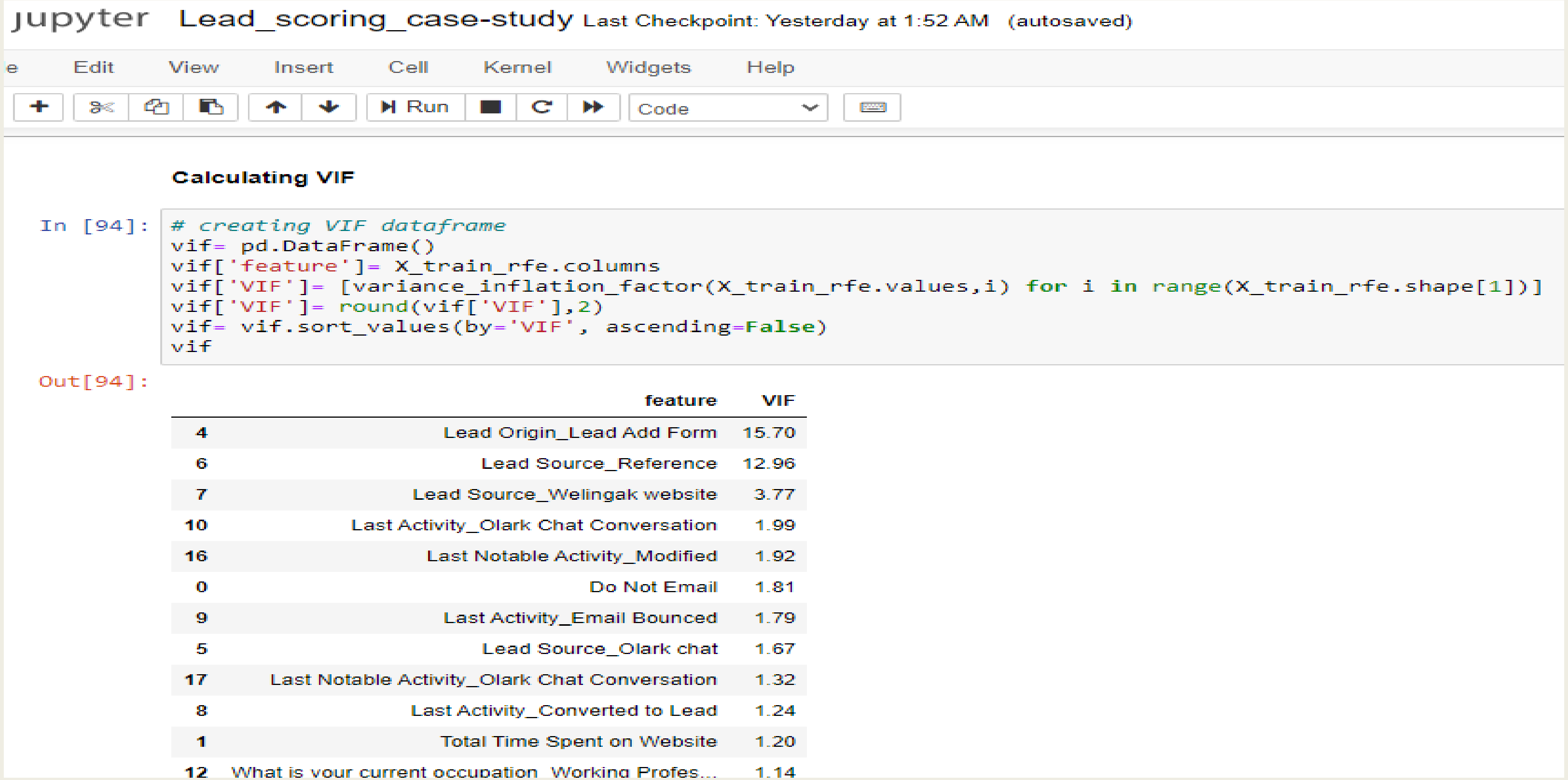
6.3.1 Model1

```
In [92]:  X_train_rfe1 = sm.add_constant(X_train_rfe)

In [93]:  logm = sm.GLM(y_train, X_train_rfe1, family= sm.families.Binomial())
          res=logm.fit()
          res.summary()

Out[93]:
          Generalized Linear Model Regression Results

               Dep. Variable:         Converted  No. Observations:      6468
                      Model:               GLM      Df Residuals:      6447
               Model Family:          Binomial          Df Model:        20
               Link Function:             logit             Scale:    1.0000
                     Method:              IRLS    Log-Likelihood:   -2679.7
                       Date:  Sat, 06 Feb 2021          Deviance:    5359.5
                       Time:          11:31:20     Pearson chi2:  7.27e+03
              No. Iterations:                21
             Covariance Type:         nonrobust

                                          coef    std err        z   P>|z|     [0.025     0.975]
                               const    0.0876      0.068    1.286   0.199     -0.046      0.221
                        Do Not Email   -1.2652      0.194   -6.534   0.000     -1.645     -0.886
         Total Time Spent on Website    1.0998      0.040   27.830   0.000      1.022      1.177
                           Newspaper  -24.0065   4.82e+04   -0.000   1.000  -9.45e+04   9.44e+04
```

- ✓ we see the p-value, there are some insignificant variables i.e p-value should be less than 5%(0.05).
- ✓ so it is good to drop that variables one by one and creating model again and again upto final model

Building a Model Cont…

Calculating VIF


jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
e Edit View Insert Cell Kernel Widgets Help
Run
Code

Calculating VIF

In [94]:

```
# creating VIF dataframe
vif= pd.DataFrame()
vif['feature']= X_train_rfe.columns
vif['VIF']= [variance_inflation_factor(X_train_rfe.values,i) for i in range(X_train_rfe.shape[1])]
vif['VIF']= round(vif['VIF'],2)
vif= vif.sort_values(by='VIF', ascending=False)
vif
```

Out[94]:

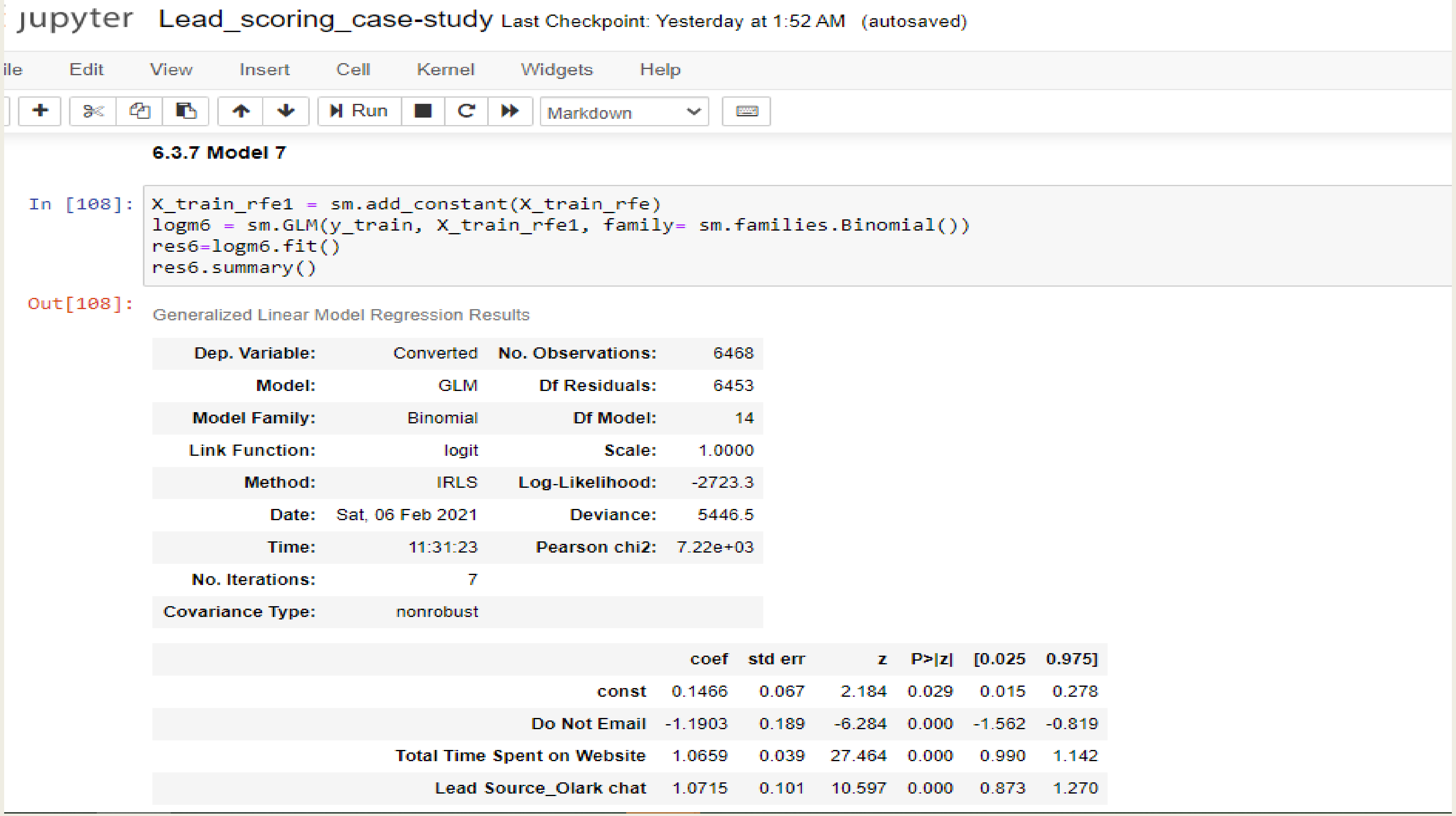
	feature	VIF
4	Lead Origin_Lead Add Form	15.70
6	Lead Source_Reference	12.96
7	Lead Source_Welingak website	3.77
10	Last Activity_Olark Chat Conversation	1.99
16	Last Notable Activity_Modified	1.92
0	Do Not Email	1.81
9	Last Activity_Email Bounced	1.79
5	Lead Source_Olark chat	1.67
17	Last Notable Activity_Olark Chat Conversation	1.32
8	Last Activity_Converted to Lead	1.24
1	Total Time Spent on Website	1.20
12	What is your current occupation_Working Profes...	1.14

- ✓ As we seen there are few variables with high VIF.
- ✓ So it is better to drop that variables as they not going help more in prediction.
- ✓ So it is good to drop that variables one by one and creating model again and again up to final model

Building a Model Cont…

Model I

Generalized Linear Model Regression



jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
ile
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Markdown

6.3.7 Model 7

```
In [108]: X_train_rfe1 = sm.add_constant(X_train_rfe)
          logm6 = sm.GLM(y_train, X_train_rfe1, family= sm.families.Binomial())
          res6=logm6.fit()
          res6.summary()
```

```
Out[108]:
Generalized Linear Model Regression Results

   Dep. Variable:        Converted   No. Observations:       6468
           Model:              GLM       Df Residuals:       6453
    Model Family:         Binomial           Df Model:         14
   Link Function:            logit              Scale:     1.0000
          Method:             IRLS     Log-Likelihood:    -2723.3
            Date: Sat, 06 Feb 2021           Deviance:     5446.5
            Time:         11:31:23      Pearson chi2:    7.22e+03
 No. Iterations:                 7
Covariance Type:         nonrobust

                              coef   std err       z   P>|z|  [0.025  0.975]
                     const  0.1466     0.067   2.184   0.029   0.015   0.278
              Do Not Email -1.1903     0.189  -6.284   0.000  -1.562  -0.819
Total Time Spent on Website 1.0659     0.039  27.464   0.000   0.990   1.142
      Lead Source_Olark chat 1.0715     0.101  10.597   0.000   0.873   1.270
```

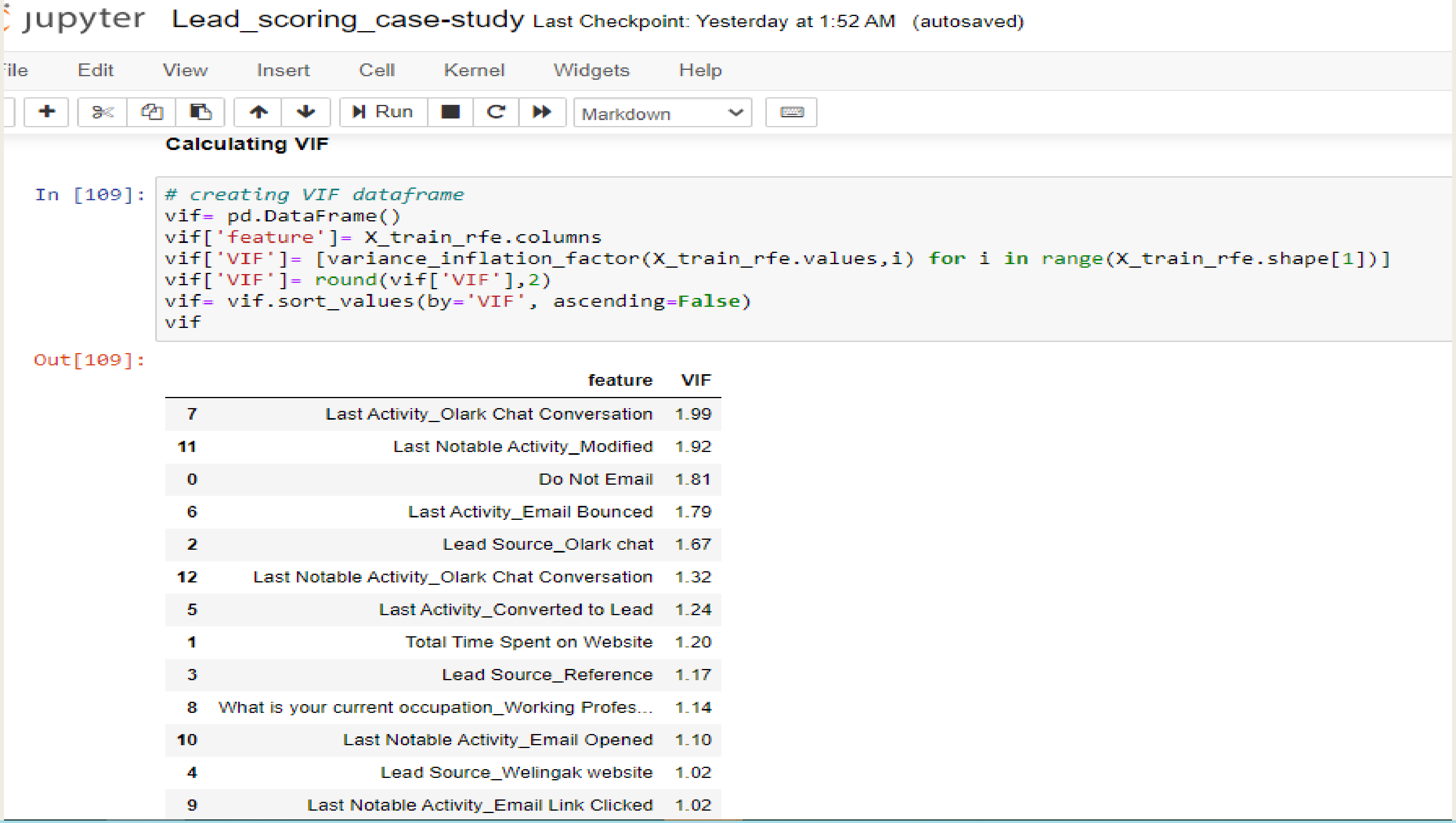
✓ This is our final model(model 7).

✓ We removed all insignificants variables whose p-value more than 5%(0.05).

✓ Now it is good to go to our next analysis.

Building a Model Cont…

Model 7,VIF



jupyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Markdown

Calculating VIF

```
In [109]: # creating VIF dataframe
vif= pd.DataFrame()
vif['feature']= X_train_rfe.columns
vif['VIF']= [variance_inflation_factor(X_train_rfe.values,i) for i in range(X_train_rfe.shape[1])]
vif['VIF']= round(vif['VIF'],2)
vif= vif.sort_values(by='VIF', ascending=False)
vif
```

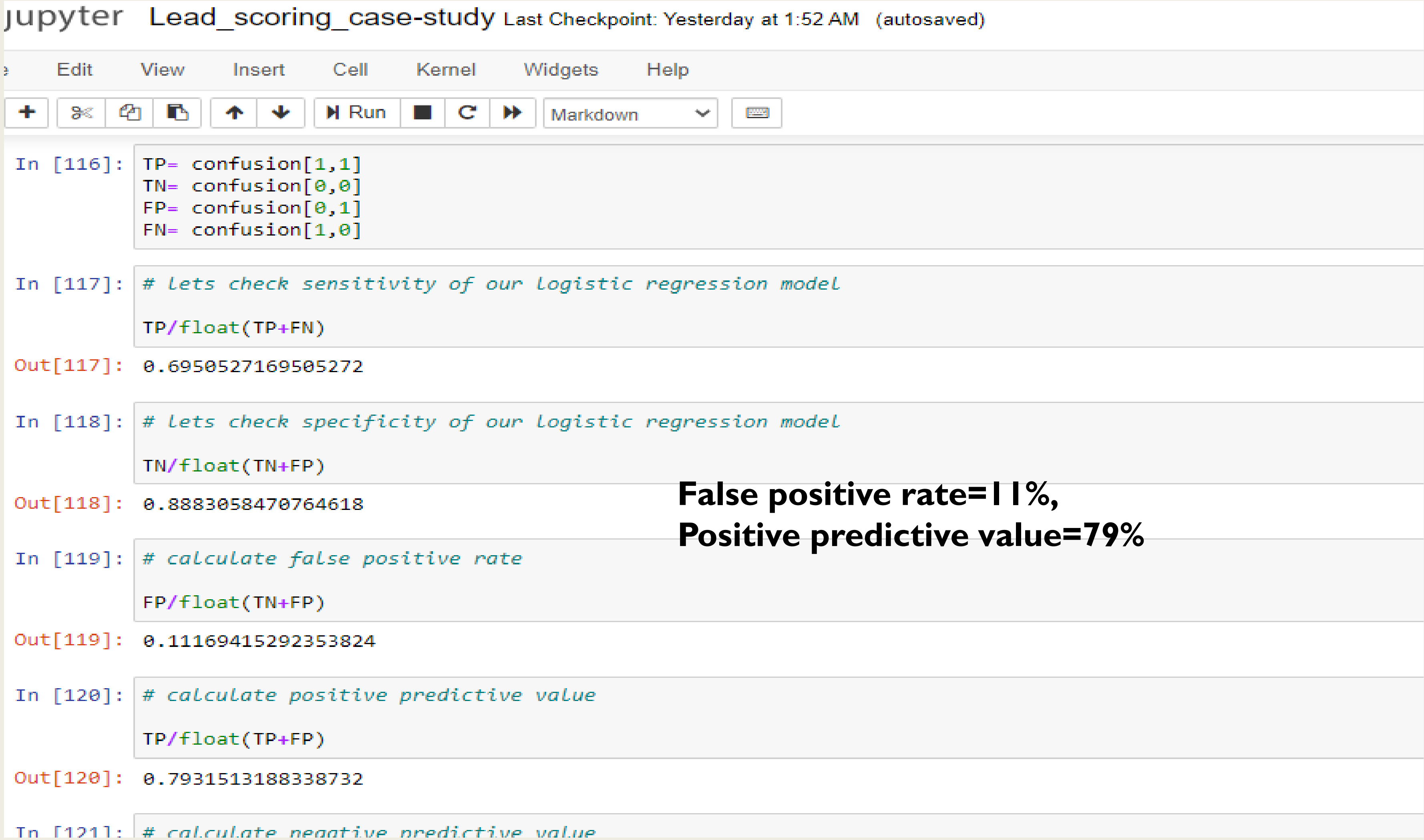
Out[109]:

	feature	VIF
7	Last Activity_Olark Chat Conversation	1.99
11	Last Notable Activity_Modified	1.92
0	Do Not Email	1.81
6	Last Activity_Email Bounced	1.79
2	Lead Source_Olark chat	1.67
12	Last Notable Activity_Olark Chat Conversation	1.32
5	Last Activity_Converted to Lead	1.24
1	Total Time Spent on Website	1.20
3	Lead Source_Reference	1.17
8	What is your current occupation_Working Profes...	1.14
10	Last Notable Activity_Email Opened	1.10
4	Lead Source_Welingak website	1.02
9	Last Notable Activity_Email Link Clicked	1.02

- ✓ This is our final model VIF value which is less than 2.
- ✓ Now this model is good to go for further steps.
- ✓ Now we Predict value on the train set

Building a Model Cont…

Metrics beyond simply accuracy

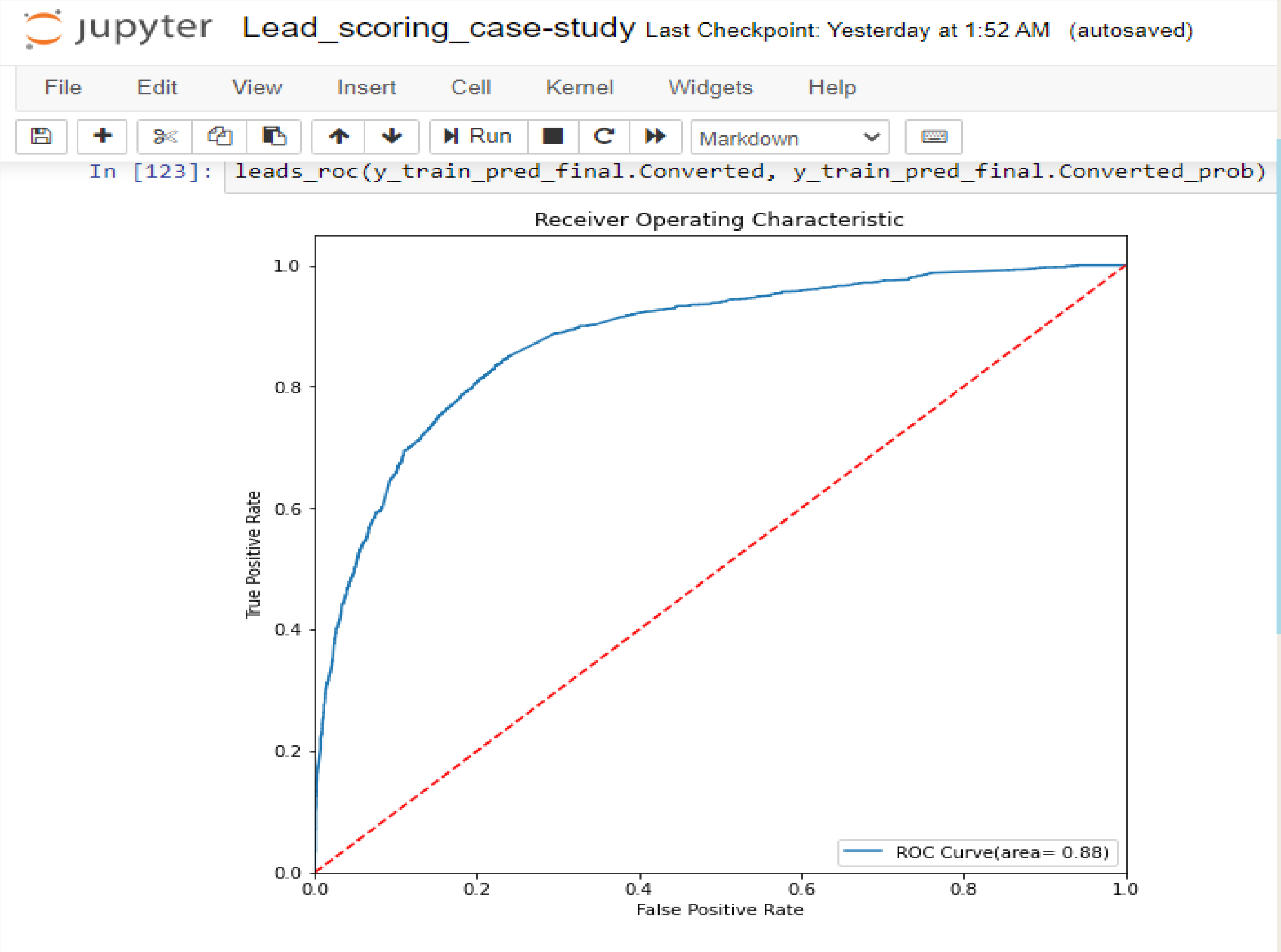

jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Markdown

```
In [116]: TP= confusion[1,1]
          TN= confusion[0,0]
          FP= confusion[0,1]
          FN= confusion[1,0]

In [117]: # lets check sensitivity of our logistic regression model

          TP/float(TP+FN)

Out[117]: 0.6950527169505272

In [118]: # lets check specificity of our logistic regression model

          TN/float(TN+FP)

Out[118]: 0.8883058470764618

In [119]: # calculate false positive rate

          FP/float(TN+FP)

Out[119]: 0.11169415292353824

In [120]: # calculate positive predictive value

          TP/float(TP+FP)

Out[120]: 0.7931513188338732

In [121]: # calculate negative predictive value
```

False positive rate=11%,
Positive predictive value=79%

Sensitivity=70%, Specificity=89%, False positive rate=11%, Positive predictive value=79%

Plotting the ROC Curve

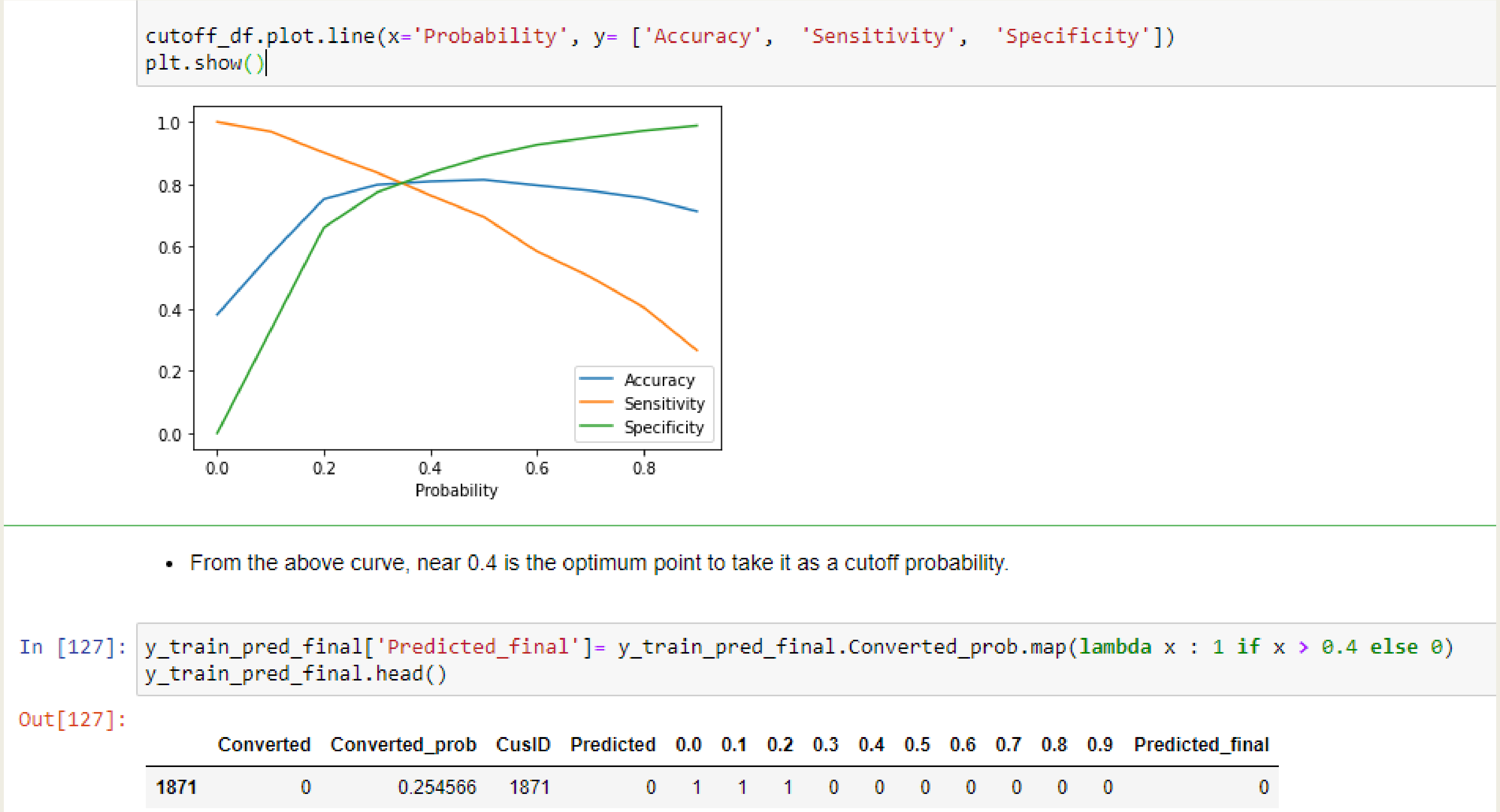


```
In [123]: leads_roc(y_train_pred_final.Converted, y_train_pred_final.Converted_prob)
```

Point to be concluded from the above curve

- ✓ The curve is closer to the left side of the border of the roc curve than to the right curve.
- ✓ Hence our model is having great accuracy.
- ✓ The area under the curve is 88% of the total area.

Finding Optimal Cut-Off Point



```
cutoff_df.plot.line(x='Probability', y= ['Accuracy',  'Sensitivity',  'Specificity'])
plt.show()
```

- From the above curve, near 0.4 is the optimum point to take it as a cutoff probability.

```
In [127]: y_train_pred_final['Predicted_final']= y_train_pred_final.Converted_prob.map(lambda x : 1 if x > 0.4 else 0)
          y_train_pred_final.head()
```

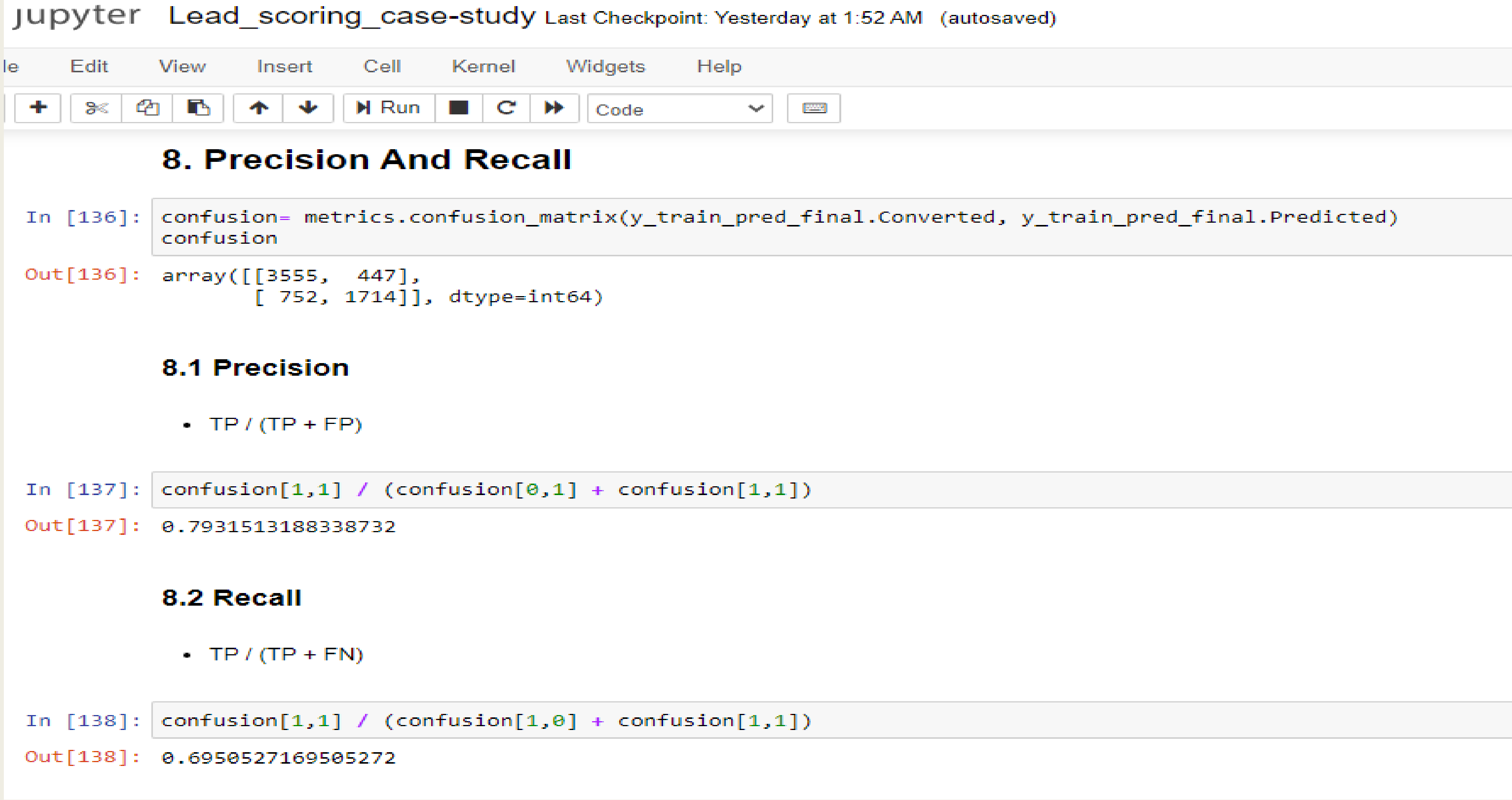
Out[127]:

	Converted	Converted_prob	CusID	Predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Predicted_final
1871	0	0.254566	1871	0	1	1	1	0	0	0	0	0	0	0	0

Plot Accuracy, Sensitivity, Specificity for various probabilities

✓ Near 0.4 is the optimum point to take it as a cutoff probability

Precision And Recall



jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
le Edit View Insert Cell Kernel Widgets Help
Run
Code

8. Precision And Recall

```
In [136]: confusion= metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.Predicted)
          confusion

Out[136]: array([[3555,  447],
                 [ 752, 1714]], dtype=int64)
```

8.1 Precision

- TP / (TP + FP)

```
In [137]: confusion[1,1] / (confusion[0,1] + confusion[1,1])

Out[137]: 0.7931513188338732
```

8.2 Recall

- TP / (TP + FN)

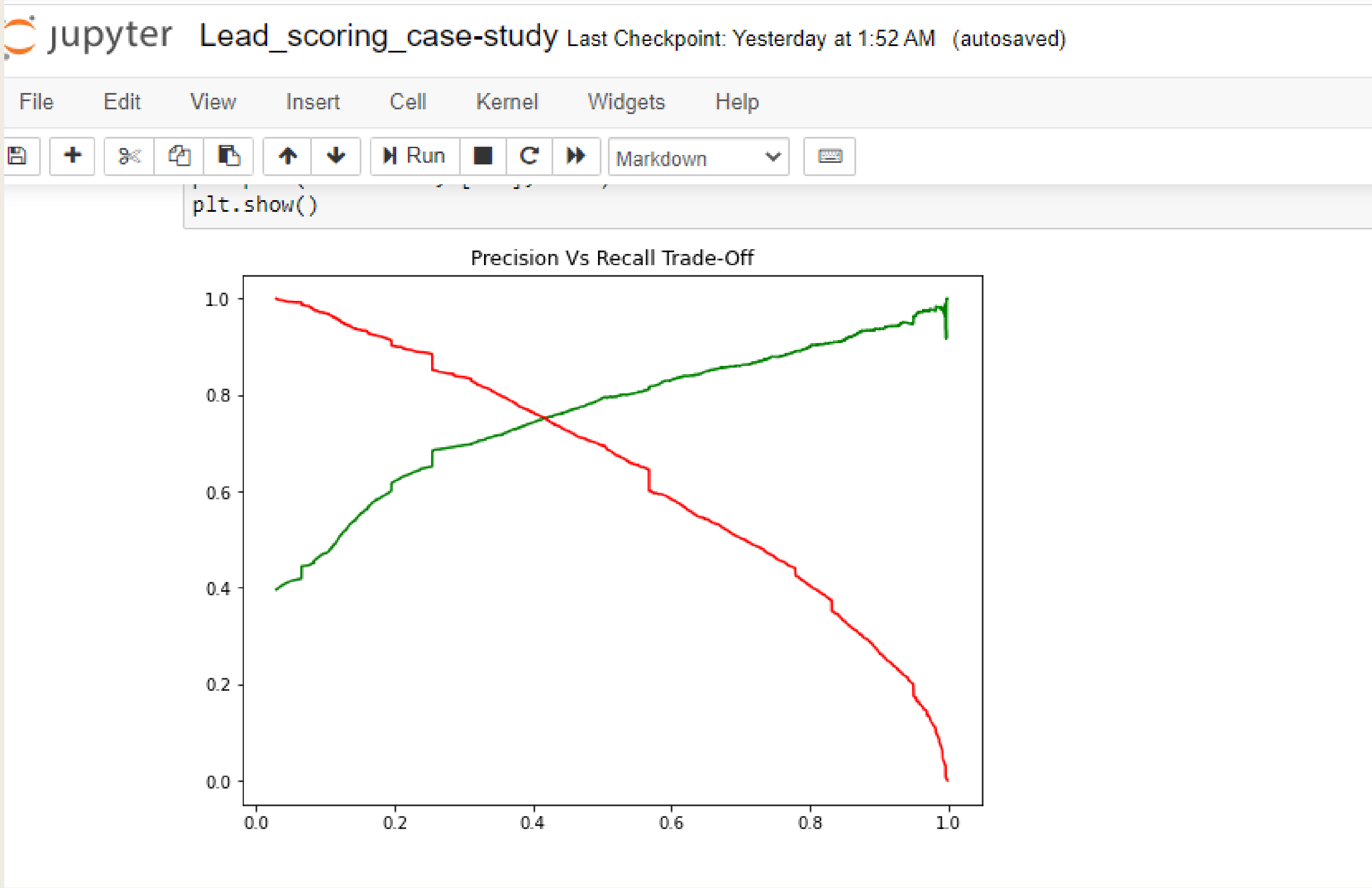
```
In [138]: confusion[1,1] / (confusion[1,0] + confusion[1,1])

Out[138]: 0.6950527169505272
```

Important point to be noted from the outcomes for Precision and recall are:

✓ Our precision percentage is approximately 79%.

✓ And our recall percentage is approximately 69%.

Precision And Recall Trade-off



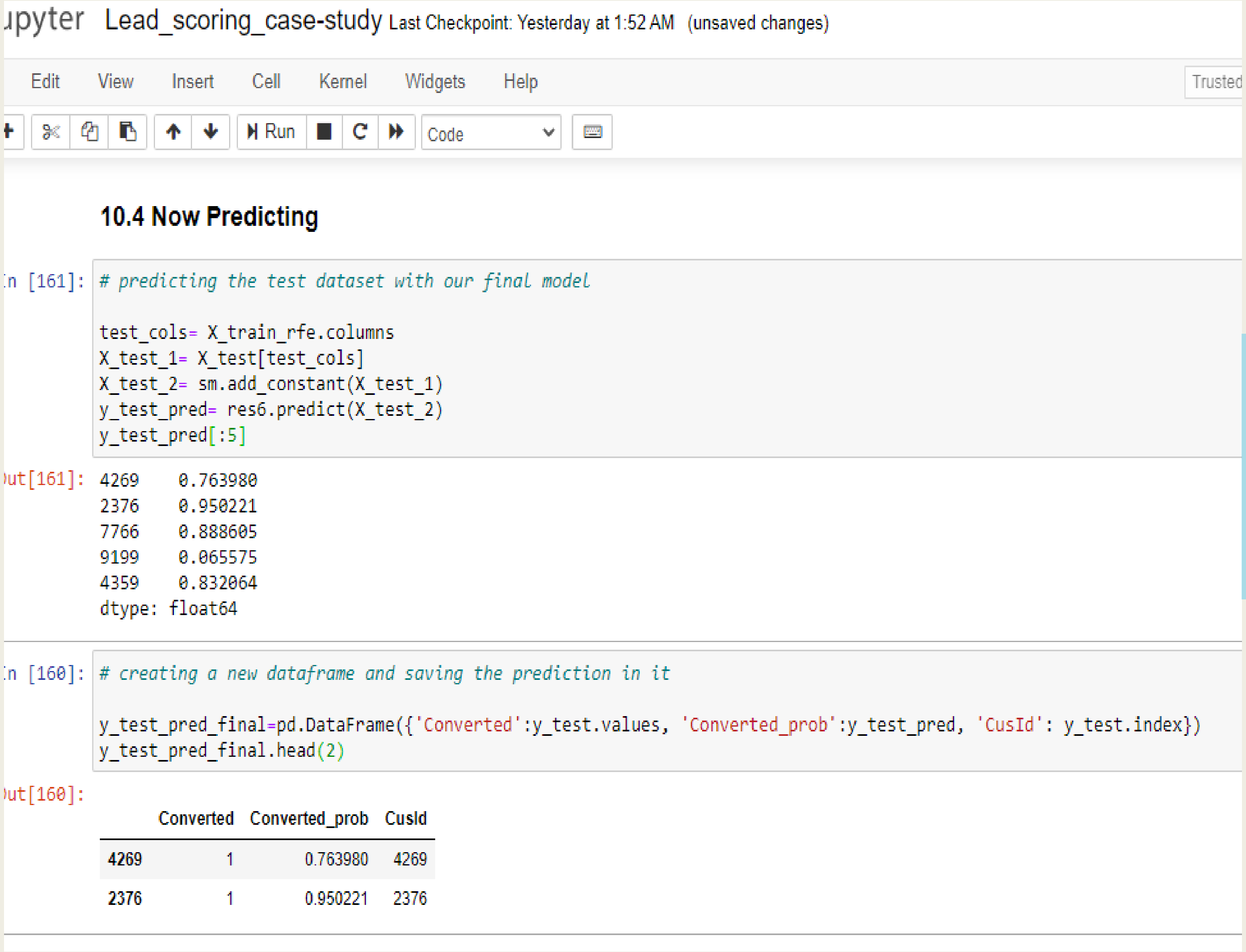
jupyter
Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Markdown

```
plt.show()
```

Precision Vs Recall Trade-Off

As we see that there is a trade-off between Precision and Recall and meeting points is nearby at 0.4

Making prediction on the Test set


upyter Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (unsaved changes)
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Run
Code

10.4 Now Predicting

n [161]:

```
# predicting the test dataset with our final model

test_cols= X_train_rfe.columns
X_test_1= X_test[test_cols]
X_test_2= sm.add_constant(X_test_1)
y_test_pred= res6.predict(X_test_2)
y_test_pred[:5]
```

ut[161]:

```
4269    0.763980
2376    0.950221
7766    0.888605
9199    0.065575
4359    0.832064
dtype: float64
```

n [160]:

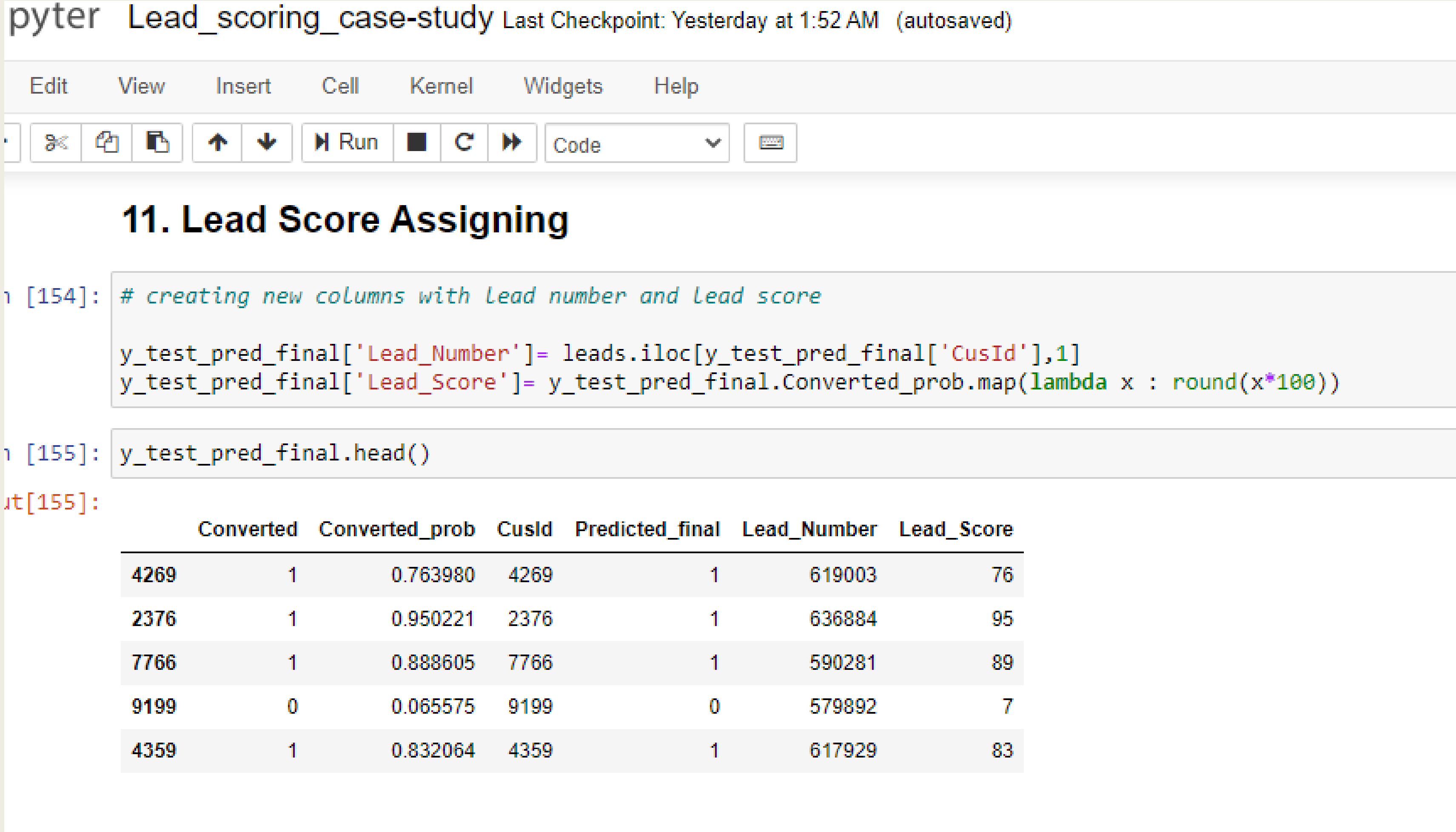
```
# creating a new dataframe and saving the prediction in it

y_test_pred_final=pd.DataFrame({'Converted':y_test.values, 'Converted_prob':y_test_pred, 'CusId': y_test.index})
y_test_pred_final.head(2)
```

ut[160]:

	Converted	Converted_prob	CusId
4269	1	0.763980	4269
2376	1	0.950221	2376

- ✓ Predicting the test dataset with our final train model
- ✓ Model Evaluation of this test data.
- ✓ Accuracy of the test_pred data = 82%
- ✓ Precision_score of the test_pred data=77%
- ✓ Recall_score of the test_pred data =76%

Lead Score Assigning



Lead_scoring_case-study Last Checkpoint: Yesterday at 1:52 AM (autosaved)

11. Lead Score Assigning

```
[154]: # creating new columns with lead number and lead score

       y_test_pred_final['Lead_Number']= leads.iloc[y_test_pred_final['CusId'],1]
       y_test_pred_final['Lead_Score']= y_test_pred_final.Converted_prob.map(lambda x : round(x*100))
```

```
[155]: y_test_pred_final.head()
```

t[155]:

	Converted	Converted_prob	CusId	Predicted_final	Lead_Number	Lead_Score
4269	1	0.763980	4269	1	619003	76
2376	1	0.950221	2376	1	636884	95
7766	1	0.888605	7766	1	590281	89
9199	0	0.065575	9199	0	579892	7
4359	1	0.832064	4359	1	617929	83

Conclusion

- ✓ Accuracy (82%), Precision_Score(77%) and Recall_Score(76%) we got from test set in acceptable range.
- ✓ In business terms, this model has an ability to adjust with the company requirements in coming future.
- ✓ Model is in stable state.

Important features responsible for good conversion rate are:

- ✓ Lead Source_Welingak website
- ✓ Lead Source_Reference
- ✓ What is your current occupation_Working Professional