REPORT ON

# LEADS SCORING

DOMAIN UNDERSTANDING

By

Suyog Karpe

## ACKNOWLEDGEMENT

I am highly intended my project guide Akash Hajari sir, for guiding and giving me timely advice and suggestions in successful completion of project work “Leads Scoring Project”.

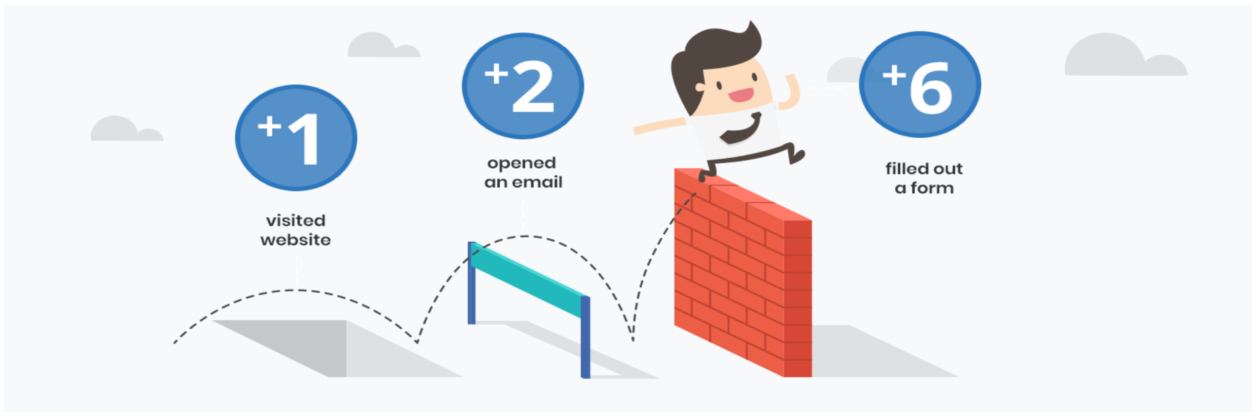
My sincere thanks to Siddharth Pense HR of Persistent Systems Ltd for his wholehearted support in completion of project and his motivation and for the project. Finally, I thank the two partners in this mentorship program who directly or indirectly helped me to complete this project.

## ABSTRACT

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

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

## CONTENTS



## DOMAIN UNDERSTANDING

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

X Education wants to select the most promising leads, i.e. the leads that are most likely to convert into paying customers. The company requires you 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 h 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%.

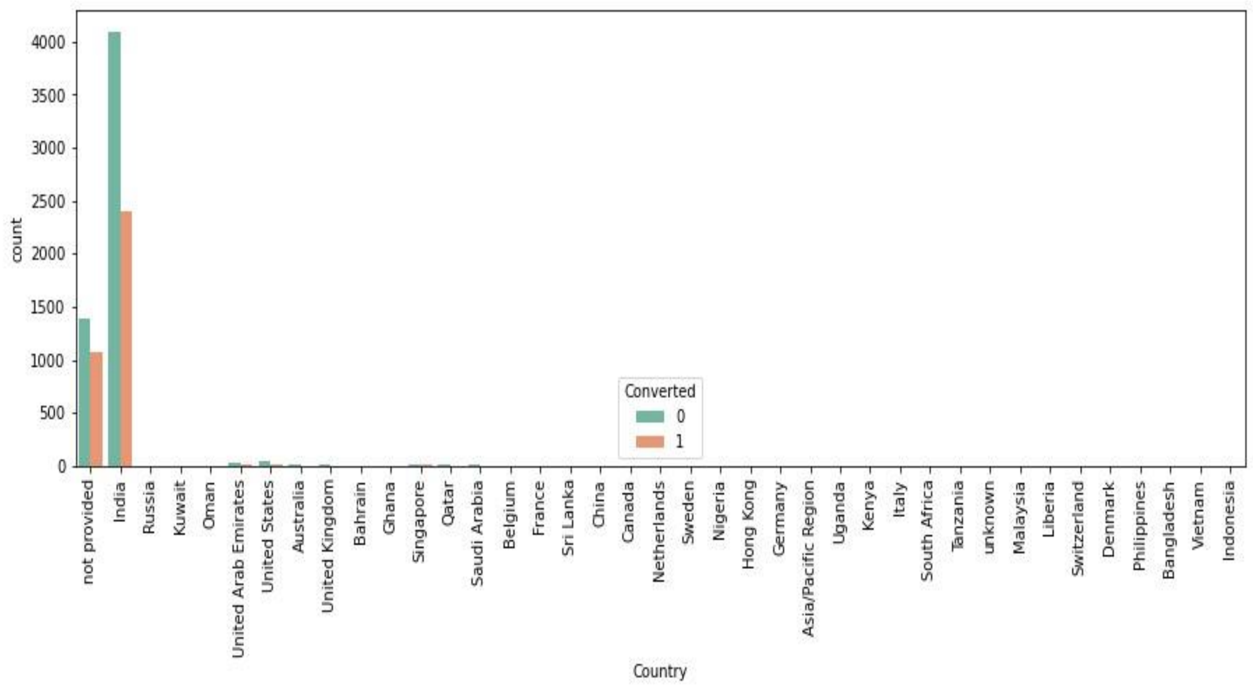
Build a logistic regression model to assign a lead score between 0 and 100 to each of the leads which can be used by the company to target potential leads. A higher score would mean that the lead is hot, i.e. is most likely to convert whereas a lower score would mean that the lead is cold and will mostly not get converted.

## Exploring Features

1. Univariate Analysis-

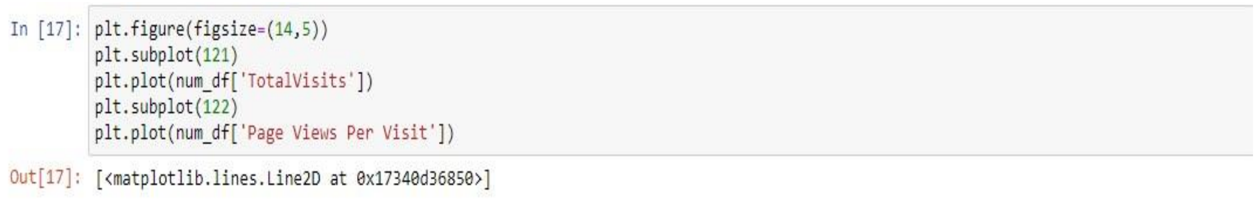
The term univariate analysis refers to the analysis of one variable. You can remember this because the prefix “uni” means “one.”

The purpose of univariate analysis is to understand the distribution of values for a single variable. You can contrast this type of analysis with the following:

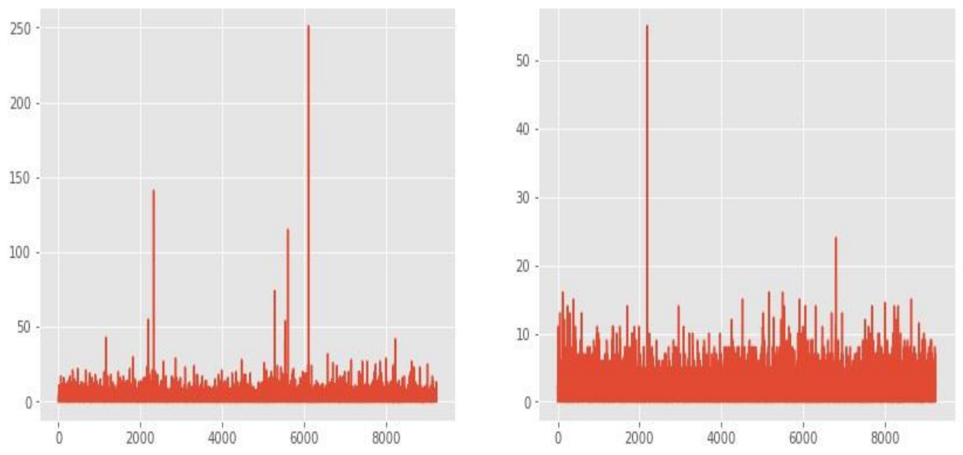


2. Bivariate Analysis-

Data in statistics is sometimes classified according to how many variables are in a particular study. For example, “height” might be one variable and “weight” might be another variable. Depending on the number of variables being looked at, the data might be univariate, or it might be bivariate.



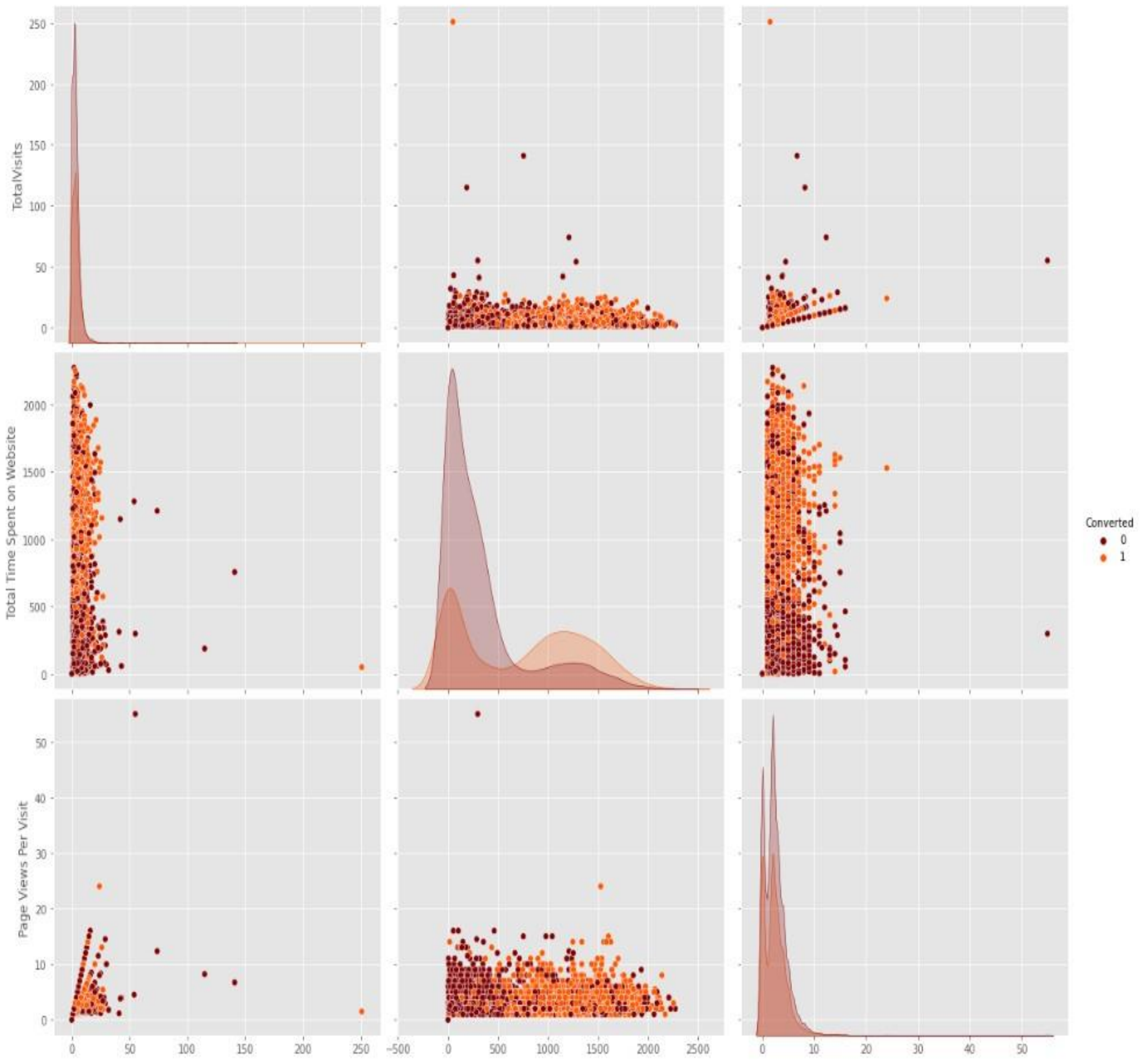
```
In [17]: plt.figure(figsize=(14,5))
         plt.subplot(121)
         plt.plot(num_df['TotalVisits'])
         plt.subplot(122)
         plt.plot(num_df['Page Views Per Visit'])

Out[17]: [<matplotlib.lines.Line2D at 0x17340d36850>]
```



3. Multivariate Analysis-

When the data involves **three or more variables**, it is categorized under multivariate. Example of this type of data is suppose an advertiser wants to compare the popularity of four advertisements on a website, then their click rates could be measured for both men and women and relationships between variables can then be examined. It is similar to bivariate but contains more than one dependent variable. The ways to perform analysis on this data depends on the goals to be achieved. Some of the techniques are regression analysis, path analysis, factor analysis and multivariate analysis of variance (MANOVA).

There are a lots of different tools, techniques and methods that can be used to conduct your analysis. You could use software libraries, visualization tools and statistic testing methods. However, this blog we will be compare Univariate, Bivariate and Multivariate analysis.

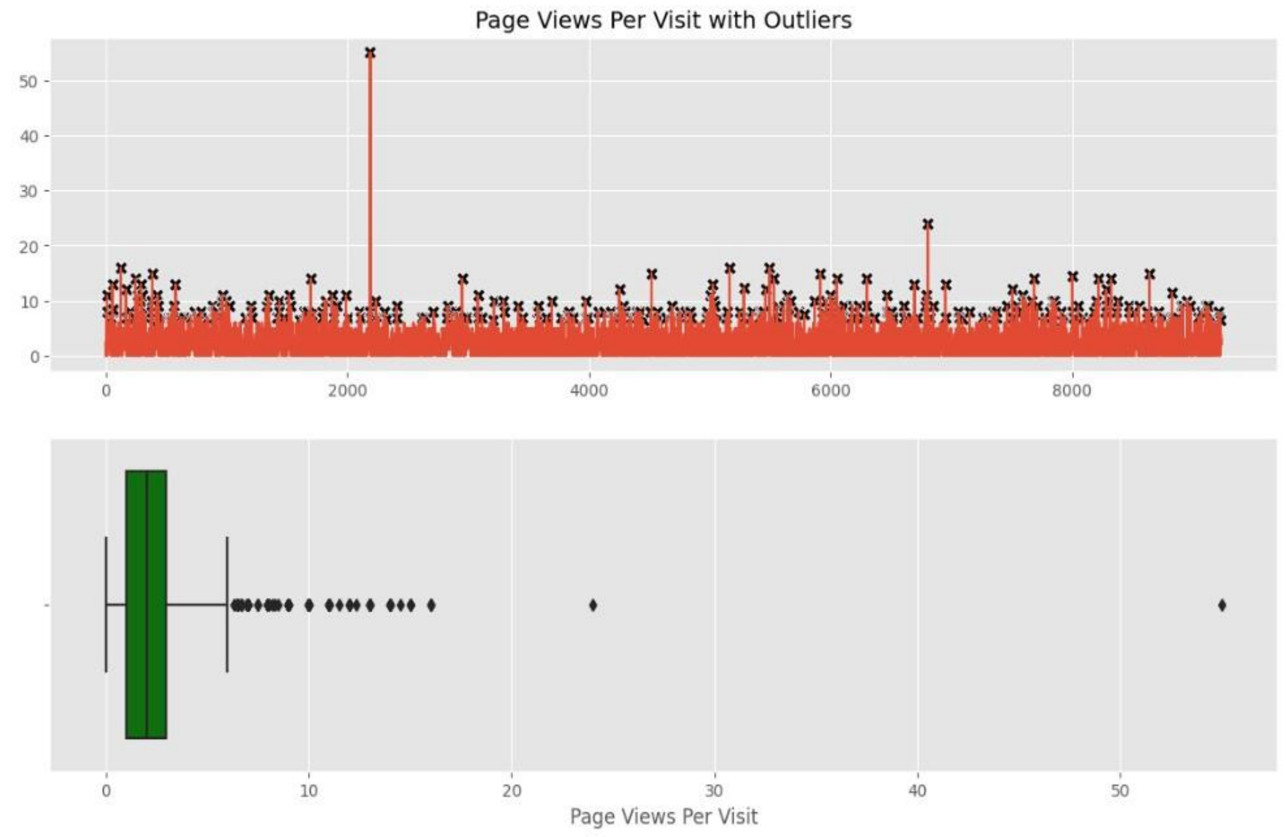

TotalVisits
Total Time Spent on Website
Page Views Per Visit
Converted
0
1

## Outlier Analysis

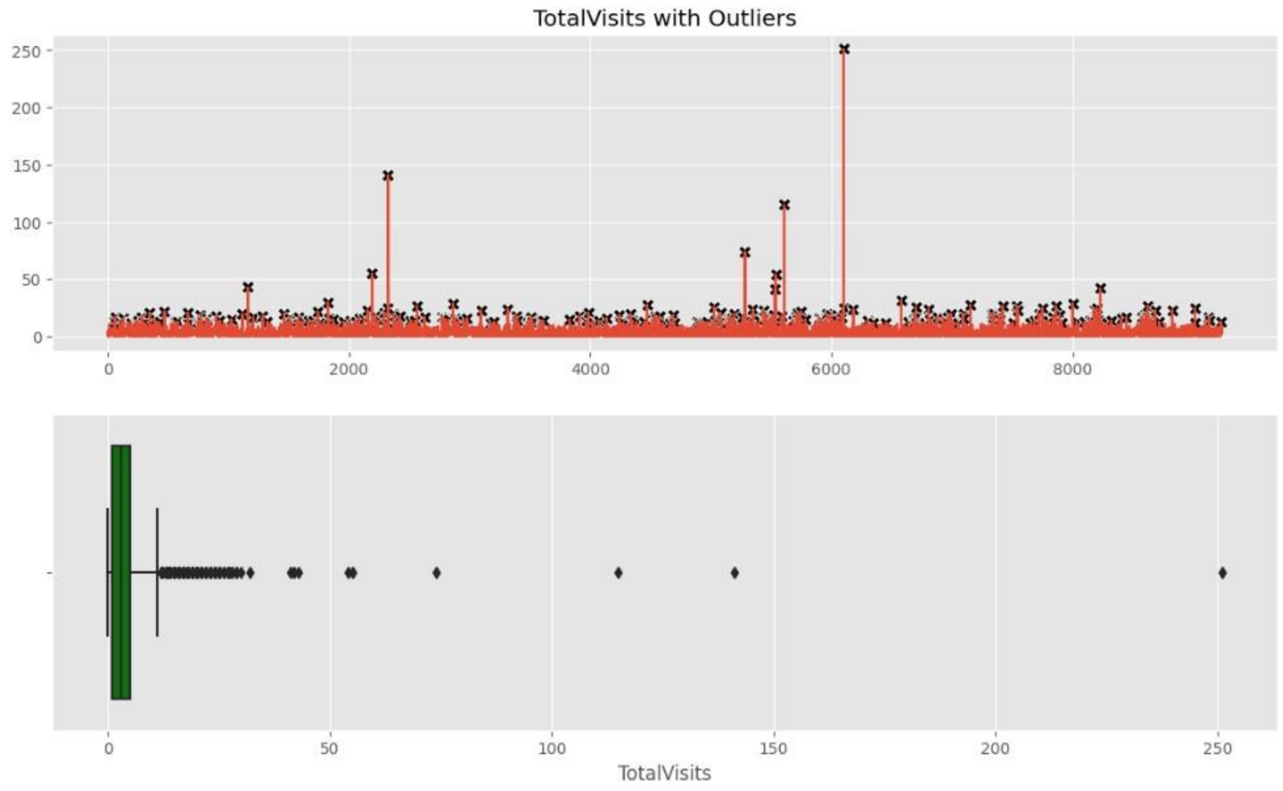
*Outlier analysis* is a **data analysis process** that involves identifying abnormal observations in a dataset. Outlier analysis is the process of identifying outliers, or abnormal observations, in a dataset. Also known as outlier detection, it’s an important step in data analysis, as it removes erroneous or inaccurate observations which might otherwise skew conclusions.

As mentioned, outlier analysis should be performed as part of any data analysis procedure. In this case, outlier analysis should be one of the first — if not the first — steps in data analysis. This way, when the dataset reaches steps that truly involve assessing and interpreting the data, any outliers will have already been removed.

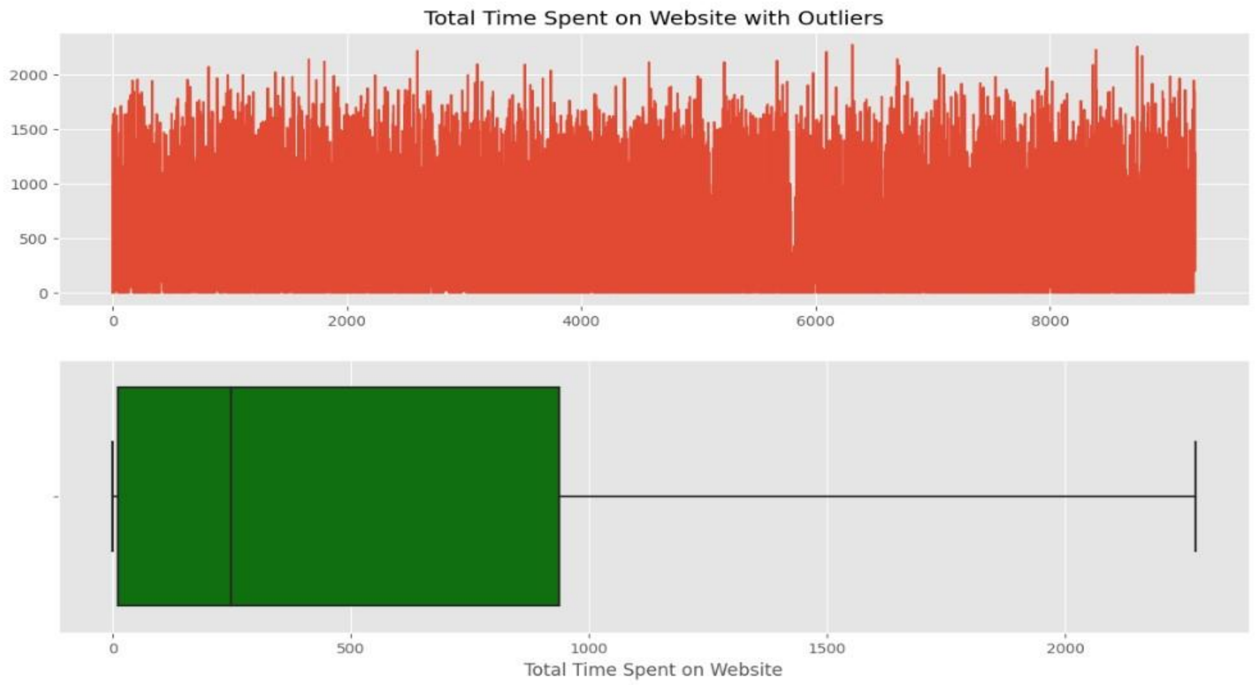
```
.00]: check_outlier(final_df, 'Page Views Per Visit')
```



```
check_outlier(final_df, 'TotalVisits')
```



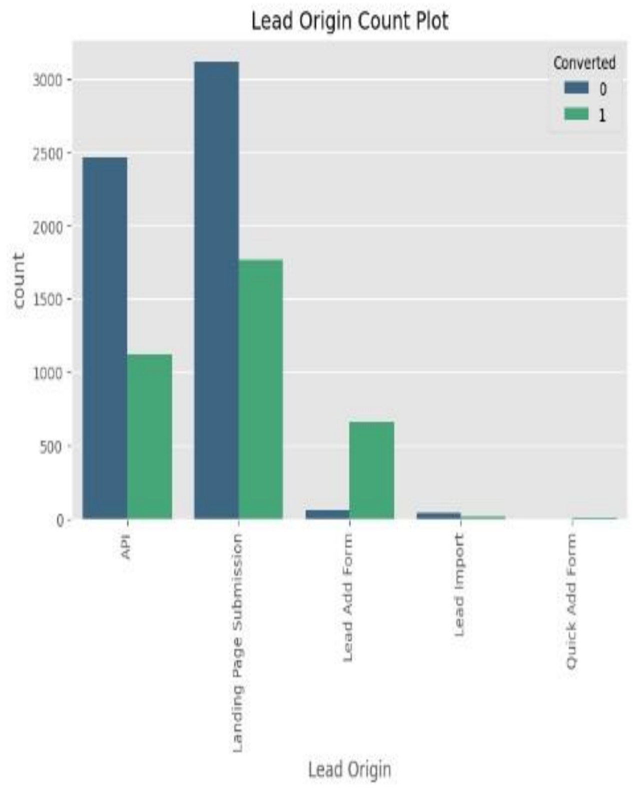
```
check_outlier(final_df, 'Total Time Spent on Website')
```

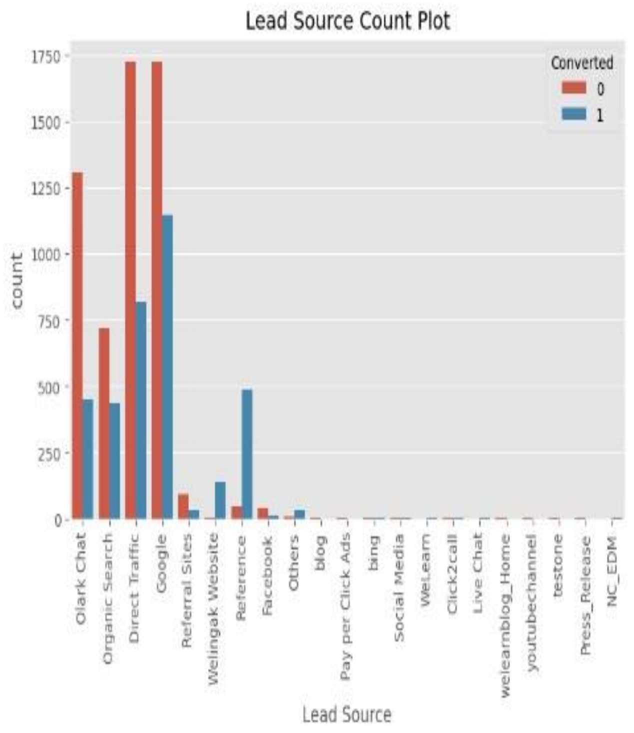


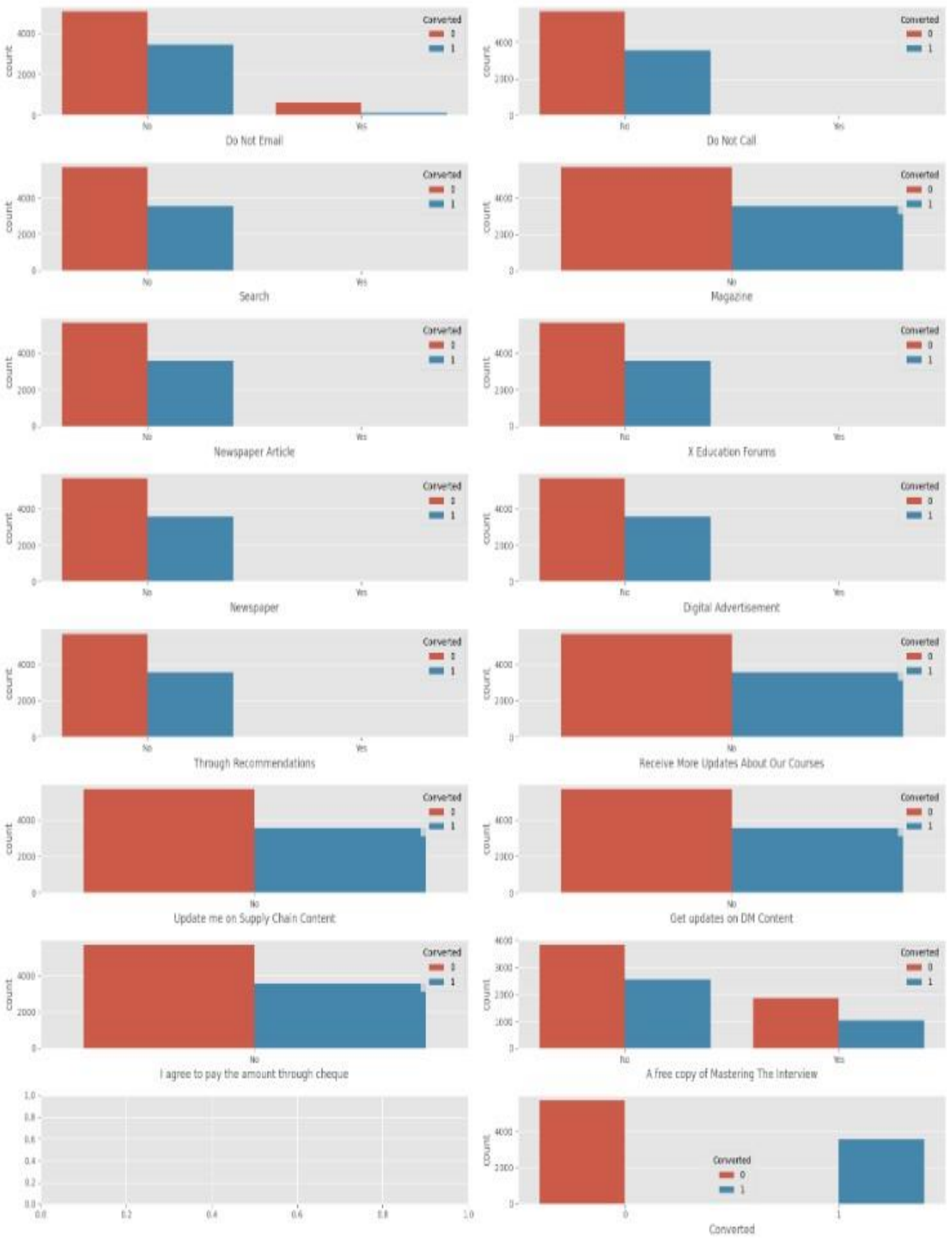
Different types of Plots Used:

- Bar Plot-

For Comparision of different variables

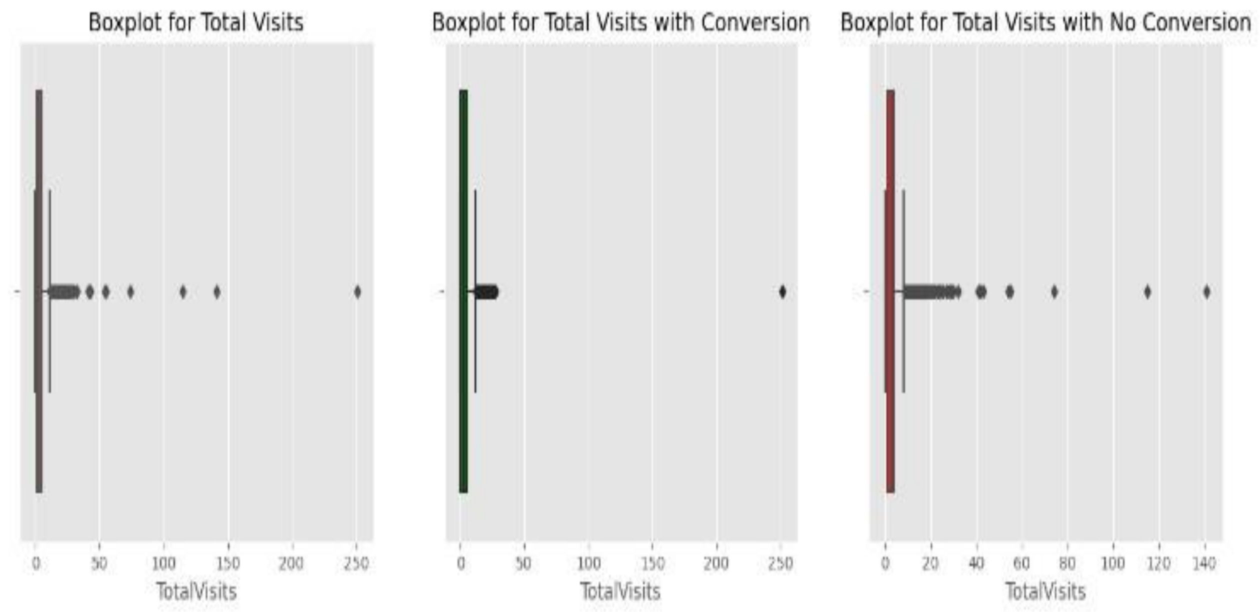





Do Not Email
Do Not Call
Search
Magazine
Newspaper Article
X Education Forums
Newspaper
Digital Advertisement
Through Recommendations
Receive More Updates About Our Courses
Update me on Supply Chain Content
Get updates on DM Content
I agree to pay the amount through cheque
A free copy of Mastering The Interview
Converted
count
No
Yes
Converted
0
1

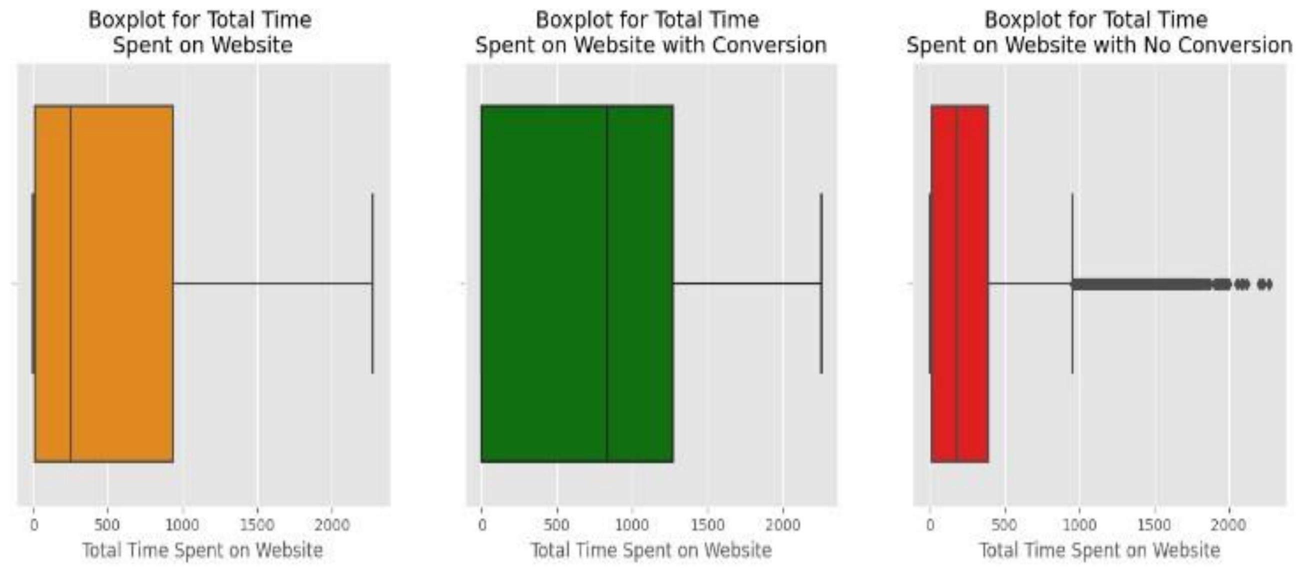
- Boxplot Analysis-

For Total Visits



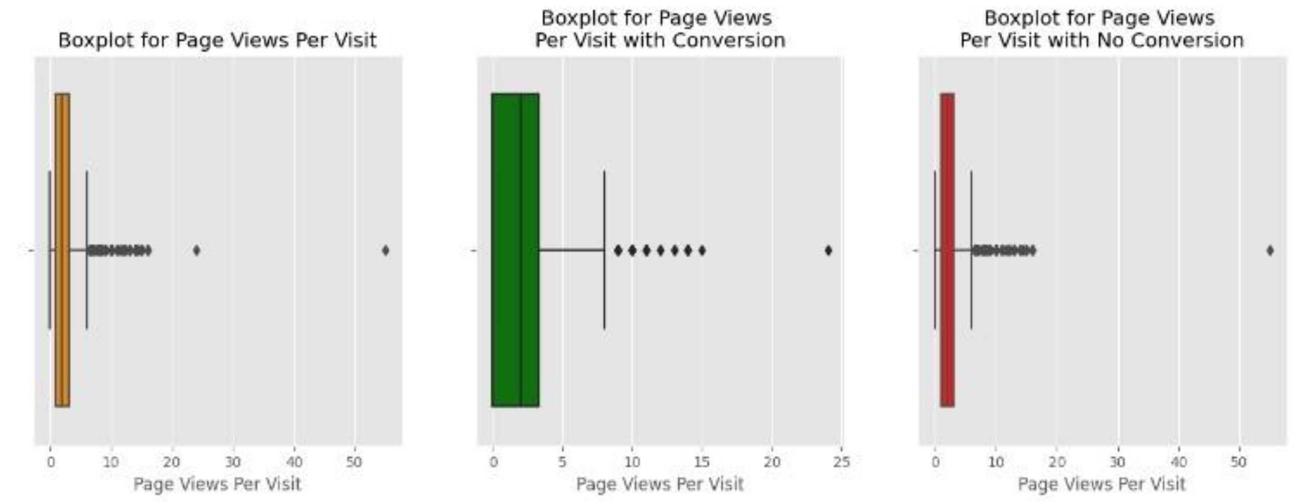
As we already see in our pairplot for the extreme values but via this boxplot, now we are clearly see some high extreme values in our data for the Total Visits.

For Total Time



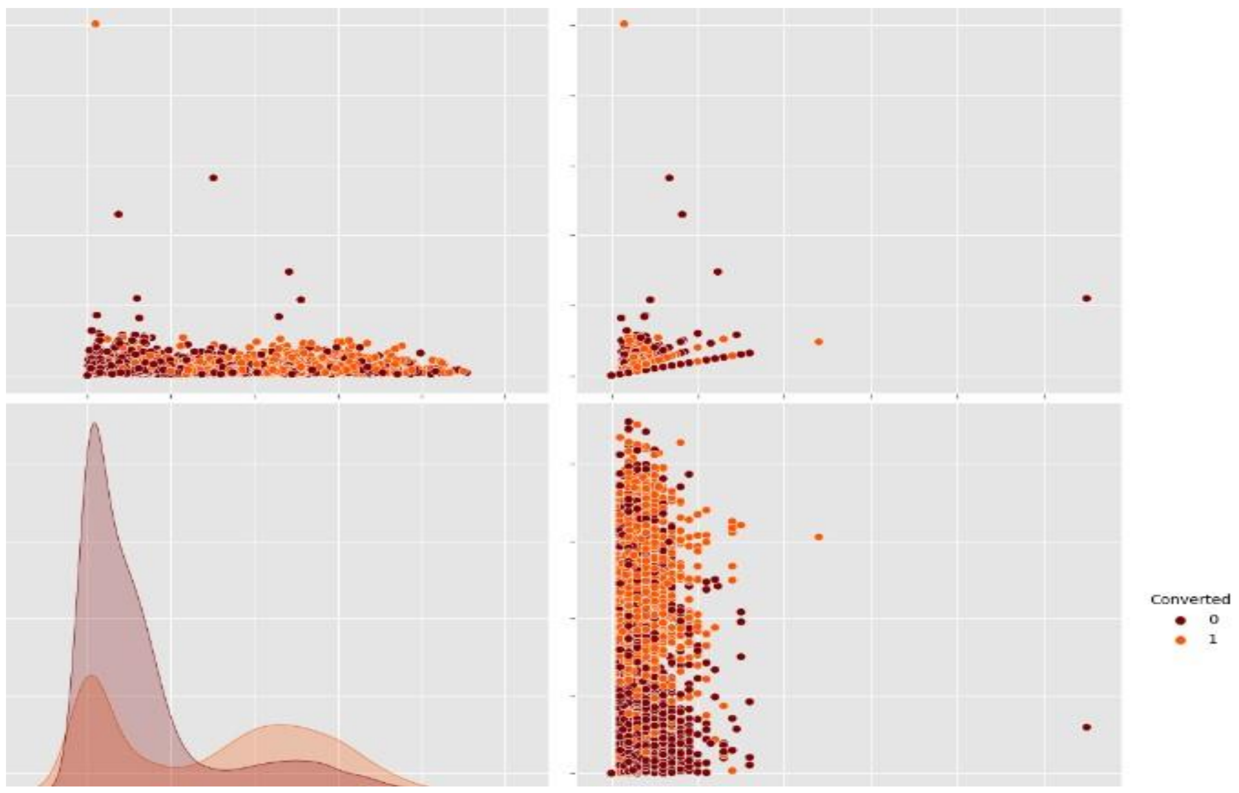
Here we can see some potential outliers for the class with **"No Conversion"** but when we look data at the aggregate level the effect of **"No Conversion"** seems to be neutralize because of **"Conversion** class seems to be far less skewed as compared to **"No Conversion"** class.

For Page Views

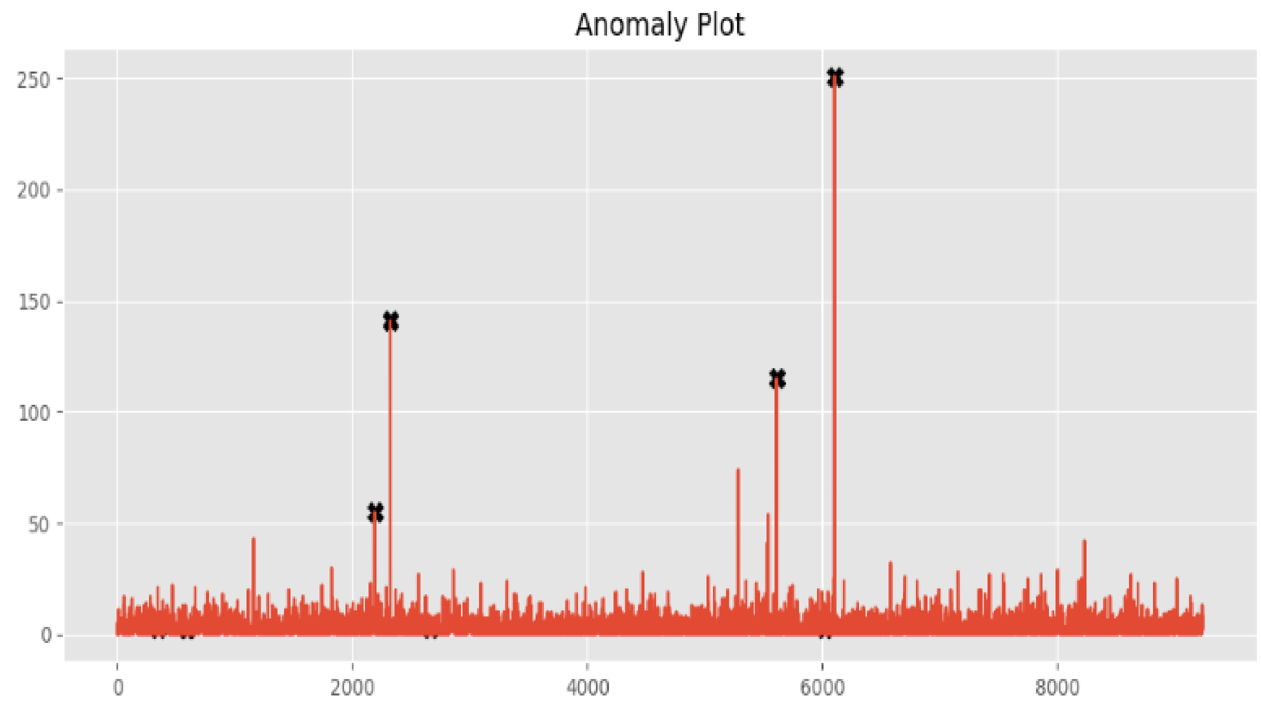


We can also see some extreme values here as well which indicated that we need to fix these outliers before procedding with our analysis.

- Scatterplot-



- Anamoly plot



# Lead Scoring Classification Model

Classification is a common task in machine learning that involves assigning a label or class to a given input data. It is a type of supervised learning, where the algorithm is trained on a labeled dataset to make predictions on new, unseen data. In this article, we'll take a closer look at classification models and how they can be used in real-world applications.

## How to Choose the Right Model

The choice of a classification model will depend on the specific problem you're trying to solve and the characteristics of your data. Some factors to consider when choosing a model include:

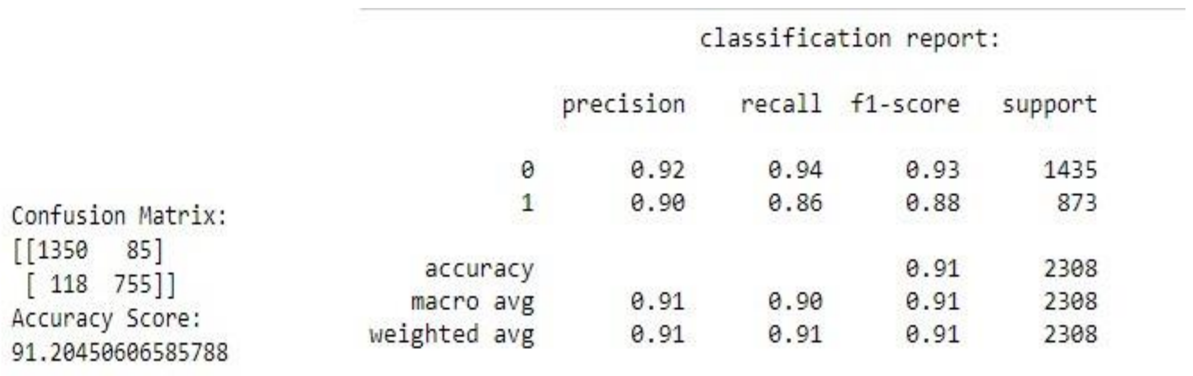
4. Number of classes: If you have a large number of classes, a model like k-NN or Random Forest may be more appropriate.
5. Size of the dataset: If you have a large dataset, a more complex model like Random Forest or SVM may be more appropriate, as they can handle larger amounts of data.
6. Complexity of the relationships between the input features and the target class: If the relationships between the input features and the target class are complex, a more complex model like Random Forest or SVM may be more appropriate.
7. Computational resources: Some models, like SVM, require a lot of computational resources, so you'll need to consider the resources you have available when choosing a model.

## Implementing a Classification Model

Once you've chosen a classification model, you can implement it using one of the many machine learning libraries available, such as scikit-learn or TensorFlow. The basic steps for implementing a classification model include:

8. Preparing the data: This involves cleaning and preprocessing the data to get it into a format that the model can use.
9. Splitting the data into training and test sets: This involves dividing the data into two parts, one part to train the model and one part to test the model.
10. Training the model: This involves feeding the training data into the model and allowing it to learn the relationships between the input features and the target class.
11. Evaluating the model: This involves using the test data to evaluate the accuracy of the model and make any necessary adjustments.
12. Making predictions: Once the model is trained and evaluated, you can use it to make predictions

### Logistic Regression

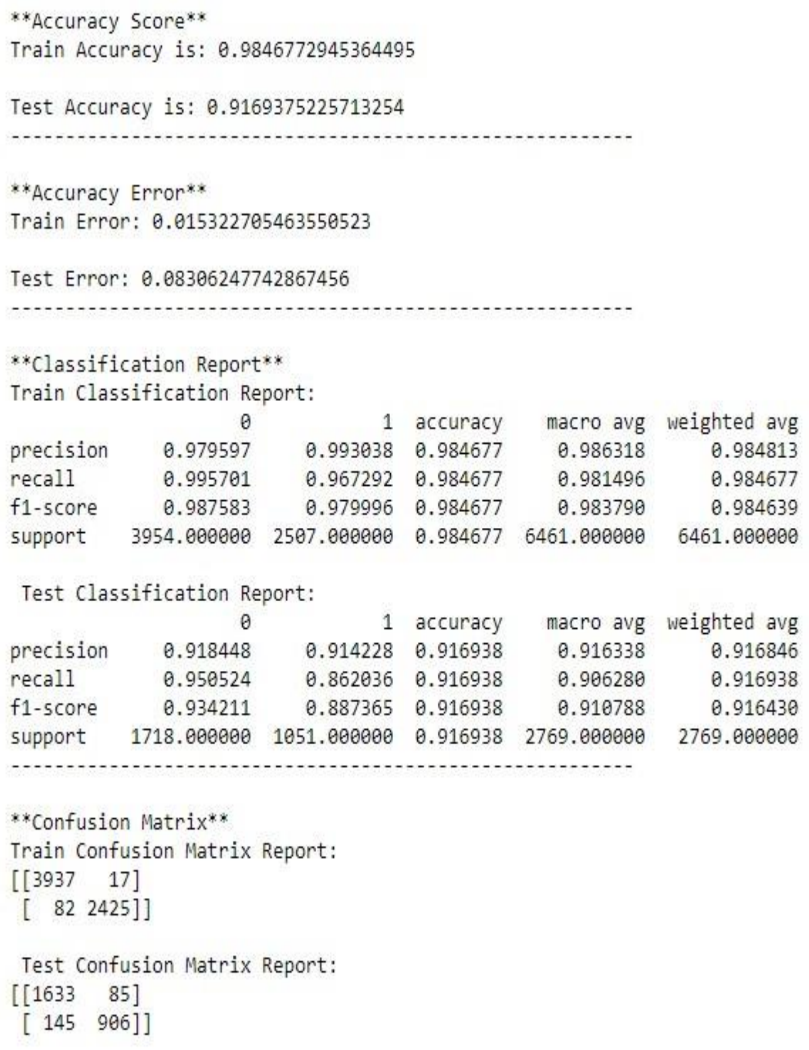


```
Confusion Matrix:
[[1350   85]
 [ 118  755]]
Accuracy Score:
91.20450606585788
```

```
              classification report:

              precision    recall  f1-score   support

           0       0.92      0.94      0.93      1435
           1       0.90      0.86      0.88       873

    accuracy                           0.91      2308
   macro avg       0.91      0.90      0.91      2308
weighted avg       0.91      0.91      0.91      2308
```

**Random Forest Classifier**

This is an extension of the decision tree model that uses multiple decision trees to make predictions. The final prediction is made by aggregating the results from all the trees.

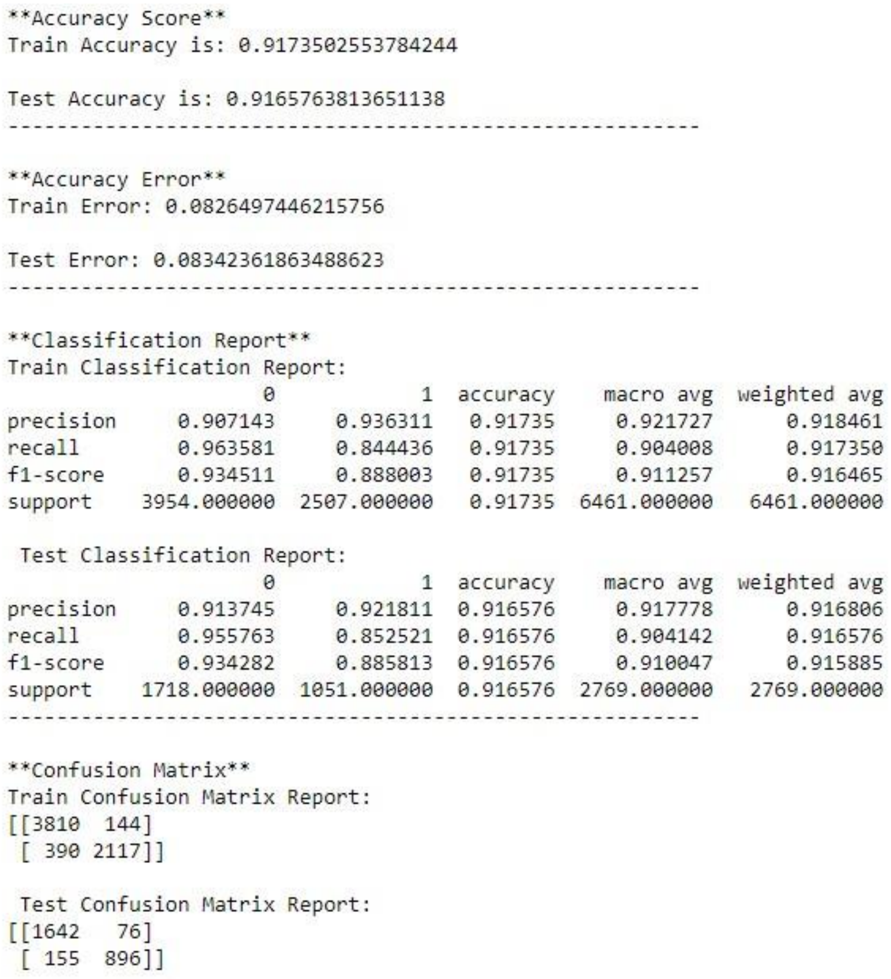


```
**Accuracy Score**
Train Accuracy is: 0.9846772945364495

Test Accuracy is: 0.9169375225713254
-------------------------------------------------------

**Accuracy Error**
Train Error: 0.015322705463550523

Test Error: 0.08306247742867456
-------------------------------------------------------

**Classification Report**
Train Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.979597     0.993038  0.984677     0.986318      0.984813
recall        0.995701     0.967292  0.984677     0.981496      0.984677
f1-score      0.987583     0.979996  0.984677     0.983790      0.984639
support    3954.000000  2507.000000  0.984677  6461.000000   6461.000000

 Test Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.918448     0.914228  0.916938     0.916338      0.916846
recall        0.950524     0.862036  0.916938     0.906280      0.916938
f1-score      0.934211     0.887365  0.916938     0.910788      0.916430
support    1718.000000  1051.000000  0.916938  2769.000000   2769.000000
-------------------------------------------------------

**Confusion Matrix**
Train Confusion Matrix Report:
[[3937   17]
 [  82 2425]]

 Test Confusion Matrix Report:
[[1633   85]
 [ 145  906]]
```

## Gradient Boosting Classifier

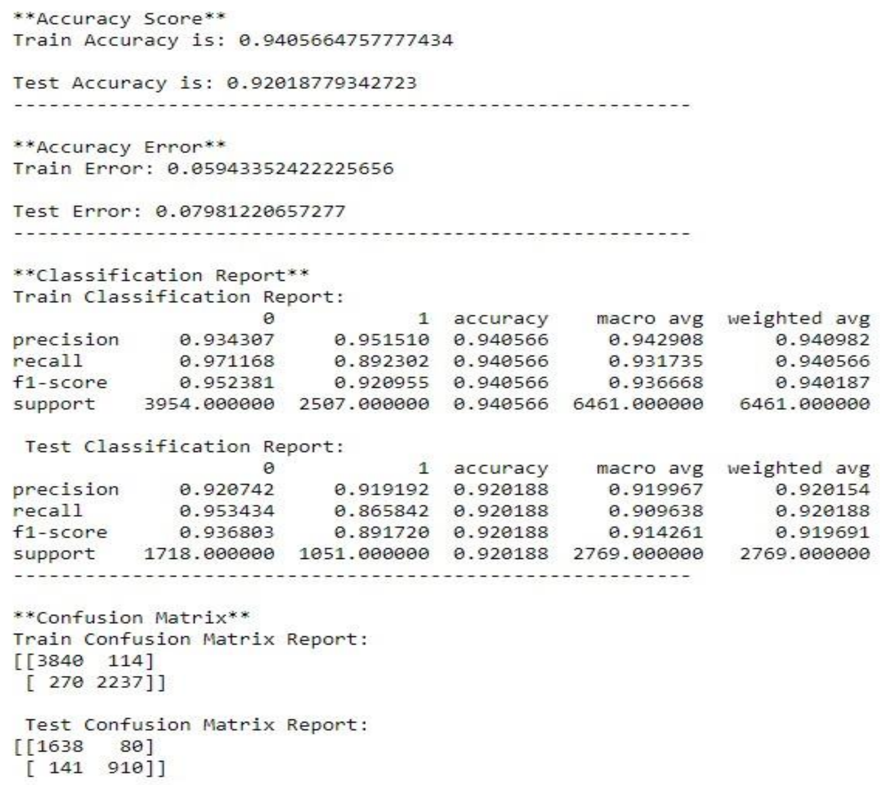
Gradient Boosting is the grouping of **Gradient descent and Boosting**. In gradient boosting, each new model minimizes the loss function from its predecessor using the Gradient Descent Method. This procedure continues until a more optimal estimate of the target variable has been achieved

Unlike other ensemble techniques, the idea in gradient boosting is that they build a series of trees where every other tree tries to correct the mistakes of its predecessor tree.



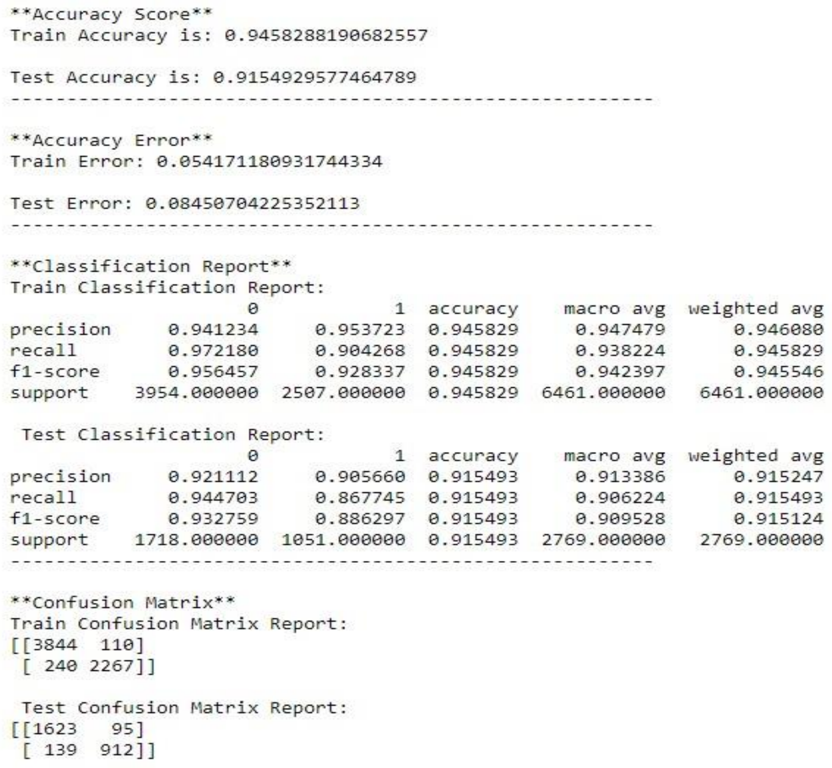
```
**Accuracy Score**
Train Accuracy is: 0.9173502553784244

Test Accuracy is: 0.9165763813651138
--------------------------------------------------------

**Accuracy Error**
Train Error: 0.0826497446215756

Test Error: 0.08342361863488623
--------------------------------------------------------

**Classification Report**
Train Classification Report:
                    0            1  accuracy    macro avg  weighted avg
precision    0.907143     0.936311   0.91735     0.921727      0.918461
recall       0.963581     0.844436   0.91735     0.904008      0.917350
f1-score     0.934511     0.888003   0.91735     0.911257      0.916465
support   3954.000000  2507.000000   0.91735  6461.000000   6461.000000

 Test Classification Report:
                    0            1  accuracy    macro avg  weighted avg
precision    0.913745     0.921811  0.916576     0.917778      0.916806
recall       0.955763     0.852521  0.916576     0.904142      0.916576
f1-score     0.934282     0.885813  0.916576     0.910047      0.915885
support   1718.000000  1051.000000  0.916576  2769.000000   2769.000000
--------------------------------------------------------

**Confusion Matrix**
Train Confusion Matrix Report:
[[3810  144]
 [ 390 2117]]

 Test Confusion Matrix Report:
[[1642   76]
 [ 155  896]]
```

## Catboost Classifier



```
**Accuracy Score**
Train Accuracy is: 0.9405664757777434

Test Accuracy is: 0.92018779342723
---------------------------------------------------------

**Accuracy Error**
Train Error: 0.05943352422225656

Test Error: 0.07981220657277
---------------------------------------------------------

**Classification Report**
Train Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.934307     0.951510  0.940566     0.942908      0.940982
recall        0.971168     0.892302  0.940566     0.931735      0.940566
f1-score      0.952381     0.920955  0.940566     0.936668      0.940187
support    3954.000000  2507.000000  0.940566  6461.000000   6461.000000

 Test Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.920742     0.919192  0.920188     0.919967      0.920154
recall        0.953434     0.865842  0.920188     0.909638      0.920188
f1-score      0.936803     0.891720  0.920188     0.914261      0.919691
support    1718.000000  1051.000000  0.920188  2769.000000   2769.000000
---------------------------------------------------------

**Confusion Matrix**
Train Confusion Matrix Report:
[[3840  114]
 [ 270 2237]]

 Test Confusion Matrix Report:
[[1638   80]
 [ 141  910]]
```

## LightGBM Classifier



```
**Accuracy Score**
Train Accuracy is: 0.9458288190682557

Test Accuracy is: 0.9154929577464789
---------------------------------------------------------

**Accuracy Error**
Train Error: 0.054171180931744334

Test Error: 0.08450704225352113
---------------------------------------------------------

**Classification Report**
Train Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.941234     0.953723  0.945829     0.947479      0.946080
recall        0.972180     0.904268  0.945829     0.938224      0.945829
f1-score      0.956457     0.928337  0.945829     0.942397      0.945546
support    3954.000000  2507.000000  0.945829  6461.000000   6461.000000

 Test Classification Report:
                     0            1  accuracy    macro avg  weighted avg
precision     0.921112     0.905660  0.915493     0.913386      0.915247
recall        0.944703     0.867745  0.915493     0.906224      0.915493
f1-score      0.932759     0.886297  0.915493     0.909528      0.915124
support    1718.000000  1051.000000  0.915493  2769.000000   2769.000000
---------------------------------------------------------

**Confusion Matrix**
Train Confusion Matrix Report:
[[3844  110]
 [ 240 2267]]

 Test Confusion Matrix Report:
[[1623   95]
 [ 139  912]]
```

## Model Evaluation

**A) Model Accuracy:**

**1) Random Forest:** When it comes to train accuracy, Random Forest have the accuracy of 98.4677% while test accuracy has been declined to 91.693% which is significant drop.

**2) Gradient Boosting:** For train dataset, we have a accuracy score of 91.7350% while for test dataset, we have a accuracy score of 91.657% which is pretty good as there is no much accuracy drop as compared to Random Forest.

**3) LightGBM:** The LightGBM algorithm gives us a train accuracy of 94.582% while test accuracy of 91.549%.

**4) CatBoost:** Under Catboost, we have a train accuracy of 94.05% while test accuracy of 92.018%. In Catboost algorithm, we have the highest test accuracy as compared to Random Forest, Gradient Boosting, LightGBM.

**B) Model Precision:**

**1) Random Forest:** When it comes to train precision for our class labels, we have a precision score of 97.95% for class label "0" and 99.30% for class label "1" while on test dataset this has been reduced. On testing dataset, precision score for class label "0" is coming out to be 91.84% while for class label "1" it is coming out to be 91.42%.

This indicating that our model requires parameters needs to be change as the score has come down significantly on the testing dataset.

**2) Gradient Boosting:** On our training data for class label "0" we have a precision score of 90.71% while for class label "1" we have a precision score of 93.63%,

On testing dataset for our class label "0" this has been increased from 90.71% to 91.37% while for class label "1" this is slightly down i.e; 92.18% but still it is pretty good as compared to Random Forest.

**3) Light GBM:** When it comes to Light GBM, our training precision score for class label "0" is coming out to be 94.12% while for class label "1" it is coming 95.37%.

As far as the testing dataset concern, the precision score of class label "0" is coming out to be 92.11% while for class label "1" it is coming out to be 90.56%.

**4) CatBoost:** Under CatBoost, for class label "0" under training dataset our precision score is coming out to be 93.43% while for class label "1" it is coming out to be 95.15%.

For testing dataset, the precision score class label "0" it is slightly down from 93.43% to 92.07% while for class label "1" it is coming out to be 91.91%.

**C) F1-Score:**

**1) Random Forest:** If we take a look at the F1-Score for Random Forest Classifier on training dataset, it is coming out to be 98.75% for class label "0" while 97.99% for class label "1".

On testing dataset, our F1-score has come down from 98.75% to 93.42% for class label "0" while for class label "1" it is coming out to be 88.73% which is again huge drop.

**2) Gradient Boosting:** On training dataset for class label "0" our F1-score is coming out to be 93.45% while for class label "1" it is coming as 88.80%. For testing dataset, the F1-score for class label "0" has been reduced to 93.42% while for class label "1" it is 88.58%.

**3) LightGBM:** On training dataset for class label "0" our F1-score is coming out to be 95.64% while for class label "1" it is coming as 92.83%. For testing dataset, the F1-score for class label "0" has been reduced to 93.27% while for class label "1" it is 88.62%.

**4) CatBoost:** On training dataset for class label "0" our F1-score is coming out to be 95.23% while for class label "1" it is coming as 92.09%. For testing dataset, the F1-score for class label "0" has been increased to 93.68% while for class label "1" it is 89.17%.

Also when it comes to confusion matrix, we are looking to increase our TP (True Positive) & TN (True Negative) as well as aiming to reduce the FN (False Negative). So for further analysis, we are taking random forest classifier and catboost classifier on which we're going to perform the hyper parameter tuning.

## CONCLUSION

In,Conlusion the project was a valuable experience for me.We have succesfully implemented a model that has the highest accuracy among others and explored different features in this project with the help from our mentor and guide.

## REFRENCES

- https://github.com/mukulsinghal001/lead-scoring-model-python/blob/main/Lead%20Scoring%20Data%20-%20EDA%20%2B%20Feature%20Engineering%20%2B%20Outlier%20Analysis.ipynb
- https://pestleanalysis.com/outlier-analysis/
- https://www.analyticsvidhya.com/blog/2020/11/popular-classification-models-for-machine-learning/
- https://visme.co/blog/types-of-graphs/