Tools for data analytics

**B8IT106**

**Team members:**

1. **Romina fratoni**
2. **Gregory Cannella**
3. **Shakir Aleem**

**Teacher:**

**Kunwar Madan**

**Submit Date: 27/04/2020**

Table of Contents

Part 1

1 – Data preparation

We familiarised ourselves with the different variables and identified our target variable to be the “Transaction” column.

We could confirm that the dataset had no missing values and that the data types were identical within each column.

We then went on to convert the categorical columns “Month” and “Visitor\_type” to 0 and 1 using the “get\_dummies()” function from pandas’ library. This splits each categorical value into its own column with 1 and 0 as values depending on whether or not the value is present for a particular row.

We split our dataset into a training set (70% of the records) and a test set (30% of the records), so that we can train our model on the training set and then test our model on the test set. We also normalised our numerical features so that each feature has a mean of 0 and variance of 1

Finally, because we are dealing with an imbalanced dataset, we used the Synthetic Minority Oversampling Technic (SMOTE) on our training set to increase the number of records in the minority class (i.e: transaction takes place).

2 – Model Evaluation Strategy

What we are trying to achieve here is to correctly predict when a transaction takes place. In other words, we are trying to build a model that can correctly predict true positives.

Knowing that false negatives is the complement of our true positives, if we add them together, this gives us the total number of “Real” positives or transactions that take place in our dataset. By minimising our false negatives this by definition increases our number of true positives.

For that reason, we decided for our GridSearchCV() function to pass in the “recall” in our scoring parameter in order to minimise the false negatives.

Model accuracy would not have been a good indicator here, since we are dealing with a very much imbalanced dataset.

3 – Model building and testing

In order to find the correct number of trees we should be building in our Random Forest model, we used the Grid Search algorithm using a range from 50 to 400 with an interval of 50. We also passed in “recall” in our scoring parameter in order to minimise false negatives.

Since our result came at the lower boundary (50), we then decided to change change our range from 20 to 50 using an interval of 5.

The number 45 ended up being the best parameter to use for our Random Forest model.

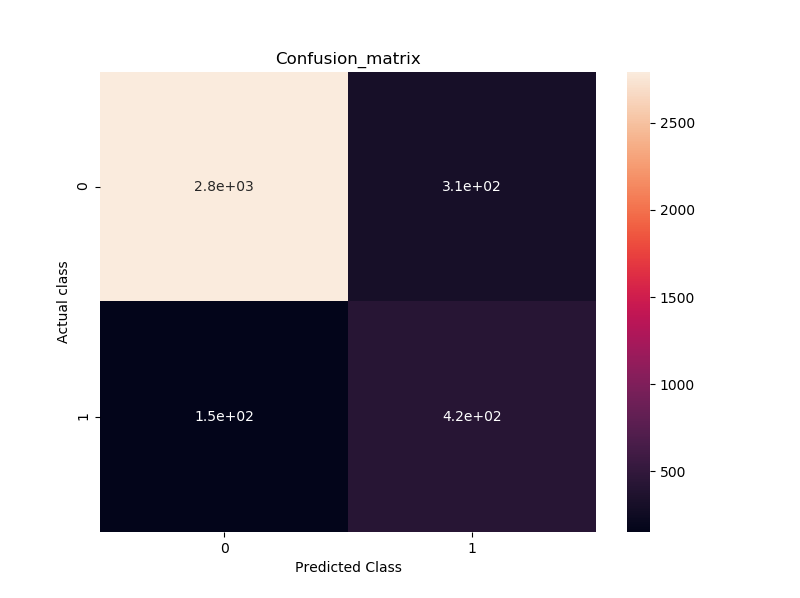
Please find below the performance for the model using all features and for the model using a subset of the features.

Average percentage of variation for the target variable that is explained by each feature:

|  |  |
| --- | --- |
| Page Value | 0.379987 |
| ExitRate | 0.087512 |
| ProductRelated\_Duration | 0.086798 |
| ProductRelated | 0.075005 |
| Administrative | 0.073419 |
| BounceRate | 0.065927 |
| Administrative\_Duration | 0.063739 |
| Month\_Nov | 0.033586 |
| Informational | 0.026658 |
| Informational\_Duration | 0.021566 |
| Month\_May | 0.015099 |
| Weekend | 0.010825 |
| Month\_Mar | 0.010393 |
| VisitorType\_Returning\_Visitor | 0.009245 |
| VisitorType\_New\_Visitor | 0.008097 |
| Month\_Dec | 0.006331 |
| SpecialDay | 0.005430 |
| Month\_Oct | 0.005324 |
| Month\_Sep | 0.005245 |
| Month\_Jul | 0.003825 |
| Month\_Aug | 0.003610 |
| Month\_June | 0.001749 |
| Month\_Feb | 0.000631 |

Performance using all features:

|  |  |
| --- | --- |
| TP | 419 |
| TN | 2793 |
| FP | 312 |
| FN | 150 |

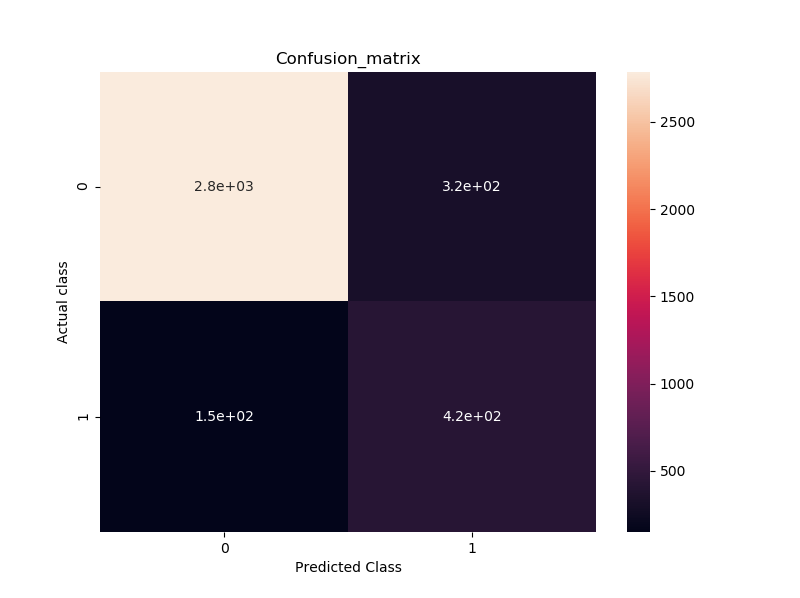


Performance using a subset of features:

List of features used in correct significance order: Page Value, ExitRate, ProductRelated\_Duration, ProductRelated, Administrative, BounceRate, Administrative\_Duration, Month\_Nov, Informational, Informational\_Duration, Month\_May, Weekend

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| number of features | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| True Positives | 345 | 414 | 410 | 401 | 412 | 414 | 420 | 415 | 415 | 415 | 415 | 422 |
| False Negatives | 224 | 155 | 159 | 168 | 157 | 155 | 149 | 154 | 154 | 154 | 154 | 147 |
| percentage of correct prediction | 0,61 | 0,73 | 0,72 | 0,70 | 0,72 | 0,73 | 0,74 | 0,73 | 0,73 | 0,73 | 0,73 | 0,74 |

|  |  |
| --- | --- |
| TP | 420 |
| TN | 2786 |
| FP | 319 |
| FN | 149 |



Based on the above figures the total number of “Real” positives values in our test dataset is: True Positives + False Negatives = Real positives

420 + 149 = 569

Percentage of correct prediction of “Real” positives = 420 / 569 = 0.7381

We can conclude that our model was able to correctly predict 73.81 % of the “Real” positives in our test dataset.

5 – Identifying best model

Our best final classification model uses only the following 7 features in order of significance: Page Value, ExitRate, ProductRelated\_Duration, ProductRelated, Administrative, BounceRate, Administrative\_Duration

It uses a RandomForestClassifer with the following parameters:

RandomForestClassifier(n\_estimators=45, criterion='entropy', max\_features='auto', random\_state=1)

The model can correctly predict 73.81 % of the “Real” positives in our test dataset.