Report on machine learning models to predict e-commerce visitors’ purchasing intention

The dataset contains shoppers’ online activity information including clickstream and session information data, where the last column Revenue represents visitors’ purchasing intention. A machine learning models is developed to predict e-commerce visitors’ purchasing intention.

Data exploration

1. By using pandas-profiling (3rd column of the table on next page)
2. Comparing all features with the median of ExitRates (2nd column )

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| Chart, histogram  Description automatically generated | Yellow dots represent that the customers did not buy anything and blue dots represent customers buying intention. Both features ProductRelated and ProductRelated\_Duration are linearly co-related. Also, as the number of pages or duration spent on Product related pages increases, there are more chances that the customer will buy the product. On the other hand, higher the number of ExitRates and BounceRates lesser is the chance that the customer will purchase. |

In ExitRates above the value of 0.1, almost all customers do not have purchasing intention i.e. any new customer falling in the region right to 0.1 in the histogram will not have purchasing intention.

Among all features, only ExitRates shows a distribution similar to Normal distribution. A comparison between ExitRates variable with the categorical features of the dataset is made in the table given below. Median of ExitRates is used instead of mean because the feature contains lots of outliers.

|  |  |  |
| --- | --- | --- |
| **1**)**Month:** Even though the month Feb comprises only 1.5% of the dataset, yet the ExitRates is highest in it. | Chart, bar chart  Description automatically generated | Table  Description automatically generated |
| **2)OperatingSystems:** In OperatingSystems =7 only makes 0.1% of total yet in it there is highest number of ExitRates. | A picture containing icon  Description automatically generated | Chart, histogram  Description automatically generated |
| **3)Browser:** In this 9th browser makes less than 0.1% of total, yet it is major contributor in ExitRates | Chart, bar chart  Description automatically generated | Chart, histogram  Description automatically generated |
| **4)Region:** 1st and 3rd have half of the total customers, yet their ExitRates are comparable to others. | Icon  Description automatically generated with medium confidence | Chart, histogram  Description automatically generated |
| **5)TrafficType:** Customers arrived from 12th , 13th, and 17th source shows maximum ExitRates. | Chart, bar chart  Description automatically generated | Chart, histogram  Description automatically generated |
| **6)VisitorTypes:**  Category Other comprises only 0.7% of the total, yet it is a major contributor in ExitRates. | A picture containing logo  Description automatically generated | Chart, pie chart  Description automatically generated |

Data Pre-Processing

To avoid over-fitting, the dataset is split into train and test datasets by using train\_test\_split from sklearn. 20% of the data is kept in test dataset and rest into train dataset.

1. Encoding Categorical Features

There are two variables in the dataset which have data-type object:

1. Month
2. VisitorType

**1.1 Encoding Categorial Feature: Month - by using Target Guided Ordinal Encoding**

As we know, there are specific months/seasons in which there are more sales (near to Christmas/New Year). Therefore, We need to rank them accordingly. For that, Ordinal Encoding (Target Guided) is used. It calculates the mean of the target for each label/category, then order the labels according to these mean from the smallest to the highest.

It captures information within the label, therefore rendering more predictive features. It creates a monotonic relationship between the variable and the target as seen in the diagram below. Higher the Month number ('Feb': 0, 'Mar': 1, 'June': 2, 'May': 3, 'Dec': 4, 'Jul': 5, 'Aug': 6, 'Sep': 7, 'Oct': 8, 'Nov': 9), the more likely the visit has been finalized with a transaction. One major drawback is that it is prone to cause over-fitting.

|  |  |
| --- | --- |
| Chart, line chart  Description automatically generated | Chart, line chart  Description automatically generated |

* 1. **Encoding Categorical Feature: VisitorType - by using One Hot Encoding**

One hot encoding, consists of replacing the categorical variable by 0 or 1, to indicate whether or not a certain category / label of the variable was present for that observation.

Apart from these two variables, Revenue (Target) and Weekend have Boolean values. They can be easily converted into 1 or 0 respectively by using astype(int) -a method in numpy.ndarray.

1. Feature Selection

Two methods are used for feature selection:

1. By creating new variables
2. By using Pearson Correlation
3. By using Fisher Score- Chi-square Test

**2.1 Creating New Variables**

|  |  |
| --- | --- |
| A picture containing square  Description automatically generated | As we can see in the key findings, that Administrative is highly related to Administrative\_Duration. Instead of using two features we can create a new variable: Administrative\_Rate which is equal to (Administrative\_Duration/Administrative) |

i.e. Total amount of time (in seconds) spent by the visitor **per** account management related page. This new variable is capable of sharing same amount of information. Similarly, we can also create Informational\_Rate for Informational and Informational\_Duration, and ProductRelated\_Rate for ProductRelated\_Duration and ProductRelated.

This way we can reduce the number of dimensions of the dataset from six to three. Hence, diminishing the need for more data.

**2.2 Pearson Correlation**

It is the ratio between covariance of two variables and the product of their standard deviation (normalised measurement of the covariance) such that the result always has a value between −1 and 1. Here it is used Selection to identify independent variables having strong correlations among themselves. Generally, |0.85| value in Pearson Correlation represent a strong Correlation between two variables.

When two variables are strongly related, they provide almost same information about the target variable. In order to avoid ‘Curse of dimensionality’, it is better to remove one of the variable which has lesser correlation with target variable or lesser resemblance of its distribution towards bell-curve.

Chart, timeline, treemap chart

Description automatically generatedIn this dataset, BounceRates and ExitRates has a strong correlation of 0.91 and Returning\_Visitor and New\_Visitor has -0.97 inter-dependence. Since, ExitRates has more bell-curve like distribution, it is better to drop BounceRates in Feature Selection.

|  |  |
| --- | --- |
|  |  |
| 1. ExitRates | 1. BounceRates |

A picture containing timeline

Description automatically generated

Similarly, Returning\_Visitor has more counts than New\_Visitor thus representing more data than the other.

**2.3 Fisher Score- Chi-square Test**

|  |  |
| --- | --- |
| Chart, histogram  Description automatically generated | This score can be used to select the features with the highest values for the test chi-squared statistic from X, which must contain only non-negative features such as Booleans or frequencies relative to the classes. The chi-square test measures dependence between stochastic variables, this function “weeds out” the features that are the most likely to be independent of class and therefore irrelevant for classification.  OperatingSystems has the highest value for the test chi-squared statistic . Hence, We can drop this column. |

1. Feature Scaling-QuantileTransformer (uniform output)

Both [**StandardScaler**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler) and [**MinMaxScaler**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html#sklearn.preprocessing.MinMaxScaler) are very sensitive to the presence of outliers. [**RobustScaler**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html#sklearn.preprocessing.RobustScaler) and [**QuantileTransformer**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html#sklearn.preprocessing.QuantileTransformer) are robust to outliers in the sense that adding or removing outliers in the training set will yield approximately the same transformation.

|  |  |
| --- | --- |
|  | But contrary to [RobustScaler](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html#sklearn.preprocessing.RobustScaler), [QuantileTransformer](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html#sklearn.preprocessing.QuantileTransformer) will also automatically collapse any outlier by setting them to the a priori defined range boundaries (0 and 1). This can result in saturation artifacts for extreme values.  Avoiding to use QuantileTransformer with Gaussian output because it will less represent the real situation which is not beneficial for business purposes.  Let's look at how ExitRates transformed  looks like as compared to the original variable: |

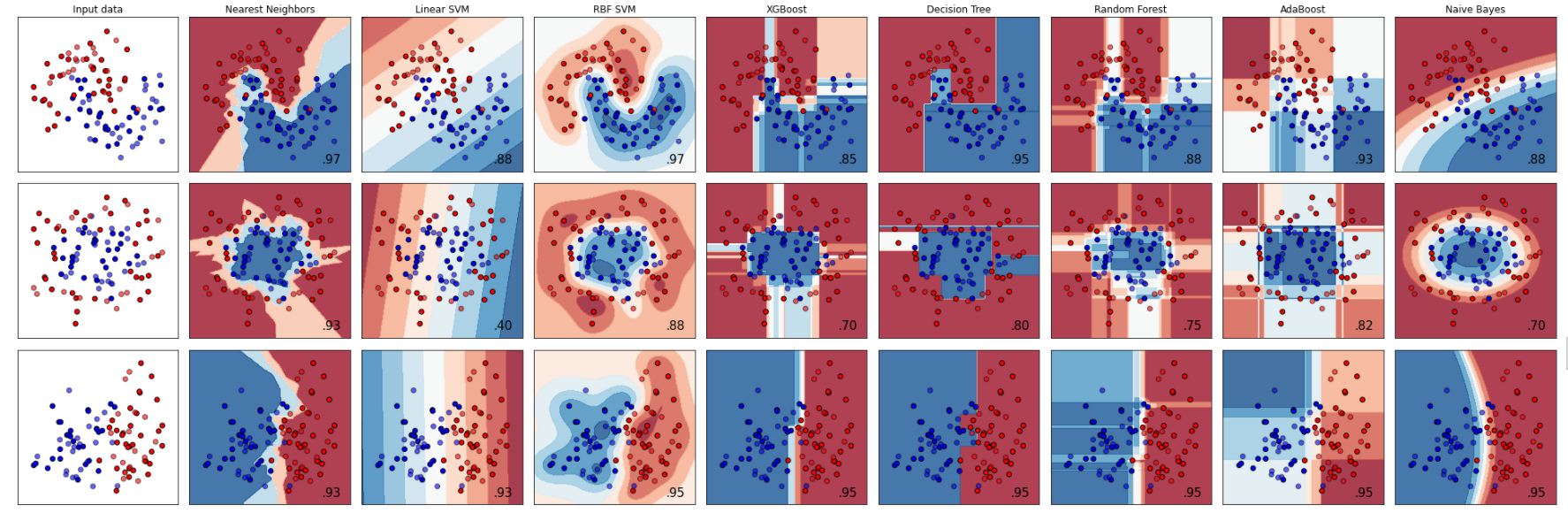
1. Sampling- using SMOTEENN

SMOTE allows to generate samples. However, this method of over-sampling does not have any knowledge regarding the underlying distribution. Therefore, some noisy samples can be generated, e.g. when the different classes cannot be well separated. Hence, it can be beneficial to apply an under-sampling algorithm to clean the noisy samples. Two methods are usually used in the literature: (i) Tomek’s link and (ii) edited nearest neighbours cleaning methods. Imbalanced-learn provides two ready-to-use samplers SMOTETomek and SMOTEENN. In general, SMOTEENN cleans more noisy data than SMOTETomek.

Resampled X\_train dataset shape- 1: 7772, 0: 6180

Resampled y\_train dataset shape- 1: 1920, 0: 1482

Model Implementation



Three best performing models chosen are:

1. XGBoost
2. Random Forest.
3. KNN
4. eXtreme Gradient Boosting

**Reason for Selection:**

XGBoost is a tree based ensemble machine learning algorithm which has higher predicting power and performance and it is achieved by improvisation on Gradient Boosting framework by introducing some accurate approximation algorithms. XGBoost tends to do well when there is a mixture of categorical and numeric features.

xgb train roc-auc: 0.993723604997013

xgb test roc-auc: 0.988396954113924

Hyperparameter Optimization is done using RandomizedSearchCV.best\_estimator\_

**best hyperparameter grid for the given dataset:**

XGBClassifier(learning\_rate=0.05, max\_depth=10, n\_estimators=1100)

2) RandomForestClassifier

Random forest can detect non-linearity. Random forest, being a complex model, will tend to fit the data with less bias. Random forest will not overfit if it is trained on totally uncorrelated trees. It can handle outliers on its own

Train set

Random Forests roc-auc: 0.973615598419203

Test set

Random Forests roc-auc: 0.9706327465717299

Hyperparameter optimization is done using GridSearchCV

**best hyperparameter grid for the given dataset:**

RandomForestClassifier(max\_depth=8,n\_estimators=10))

3) KNeighborsClassifier

The KNN algorithm can compete with the most accurate models because it makes highly accurate predictions. The quality of the predictions depends on the distance measure. Therefore, the KNN algorithm is suitable for applications for which sufficient domain knowledge is available. This knowledge supports the selection of an appropriate measure. KNN is the better choice for applications where predictions are not requested frequently but where accuracy is important.

|  |  |
| --- | --- |
| Table  Description automatically generated | Table  Description automatically generated |
| With K=1 | With K=15 |

Accuracy scores without and with hyperparameter optimization respectively.

Shape

Description automatically generated with medium confidence

Hyperparameter Optimization: Around k=15, the Error Rate becomes almost constant

**best hyperparameter grid for the given dataset:**

KNeighborsClassifier(n\_neighbors=15)

Performance evaluation- by using K Fold Cross Validation

In K Fold Cross Validation, we are making 10 splits and for all of them the machine learning model will be applied and mean of accuracy will be given out.

For ROC curve: Classifiers that give curves closer to the top-left corner indicate a better performance. One common approach is to calculate the area under the ROC curve, which is abbreviated to AUC. It is equivalent to the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance

1) eXtreme Gradient Boosting

[0.9665272 0.9693166 0.97280335 0.9713887 0.95464061 0.94277739

0.99860433 0.99441731 0.99092812 0.99371947]

Mean = 0.9755123065498351

Chart

Description automatically generated with medium confidence

XGBoost ROC curve is closest to top left corner, it has highest accuracy.

2) RandomForestClassifier

[1. 1. 1. 0.8 0.9 0.9 1. 1. 1. 1. ]

Mean = 0.96

Graphical user interface

Description automatically generated with medium confidence

1. KNeighborsClassifier

[0.92189679 0.90167364 0.92608089 0.93091417 0.94207955 0.97836706

0.97766923 0.97278437 0.98046057 0.97278437]

Mean = 0.9504710641085159

Graphical user interface

Description automatically generated with medium confidence

Result analysis and discussion

All of three models have performed well. The accuracy of both training data and test data is high. It means that the models have low variance and low bias and does not indicate towards over-fitting or under-fitting. XGBoost has performed best with mean cross\_val\_score 0.9755. RandomForestClassifier has second best: 0.96 and KNN has least with 0.9504.

PageValues has the highest importance in Random Forest classifier. ProductRelated\_Rate, Informational\_Rate, Administrative\_Rate, Month\_Num, and ExitRates are other significantly important features.

Chart, histogram, waterfall chart

Description automatically generated