Assignment - Report

Implementation can be viewed on github (as well):
https://github.com/deepankkartikey/CSI-5155-Assignments/blob/master/Assignment-1/Assignment%20Implementation.ipynb

1. **Summary of preprocessing**
   Before implementing any of the algorithms on our dataset, the most important preprocessing task is feature engineering. In our dataset for this assignment, the preprocessing steps included:

   a. Feature transformation
      This was necessary because most of the machine learning algorithms work on numeric data only. The dataset had 2 categorical variables namely **VisitorType** (3 labels) and **Month** (10 labels). Both of the features were converted to numerics using one-hot-encoding.

   b. Feature Selection
      To remove highly correlated features from the dataset, Pearson Correlation method with a threshold of 0.8 (i.e. features that are more than 80% related) was used. This helped to remove highly correlated features which may also be called duplicate features (in layman language), that do not affect the prediction of machine learning models. There were 3 highly correlated features found namely, **ExitRates**, **Returning_Visitor**, **ProductRelated_Duration**

   c. Feature Scaling
      As we know that the K-Nearest Neighbor algorithm works on the basis of distance calculation between multiple points, all features need to be scaled to similar magnitude to facilitate ease of calculation. Standard Scaler has been used to scale the features of the dataset. StandardScaler follows Standard Normal Distribution (SND). Therefore, it makes *mean = 0* and scales the data to unit variance.

2. **Implemented algorithms (on imbalanced dataset)**
   As part of the assignment, 4 machine learning algorithms were trained and used to predict labels.
   a. Decision Tree Classifier
   b. Random Forest Learner
   c. Support Vector Machine
   d. K-Nearest Neighbours

3. **Training and testing on algorithms**
   Holdout method has been utilized to train and test the implemented models. In this method, the data set is partitioned, such that – maximum data belongs to the training set and the remaining data belongs to the test set. If maximum possible data items are placed in a training set for construction of a model/classifier, the classifier's error rates and estimates would be very low and accuracy would be high, which is a sign of a good classifier/model. Our working dataset has been divided into training and testing sets in a ratio of 7 : 3.

4. **Impact of Oversampling and Undersampling**

Due to imbalance in the dataset, the results of training models on the same dataset may result in inclination towards a specific target label (False in this case), since the number of 'False' labels is greater than 5 times the number of 'True' labels.

The imbalance in dataset has been tackled using any of the below two methodologies:

a. Oversampling - Duplicate or create synthetic examples in Minority class
b. Undersampling - Delete or merge examples in Majority class

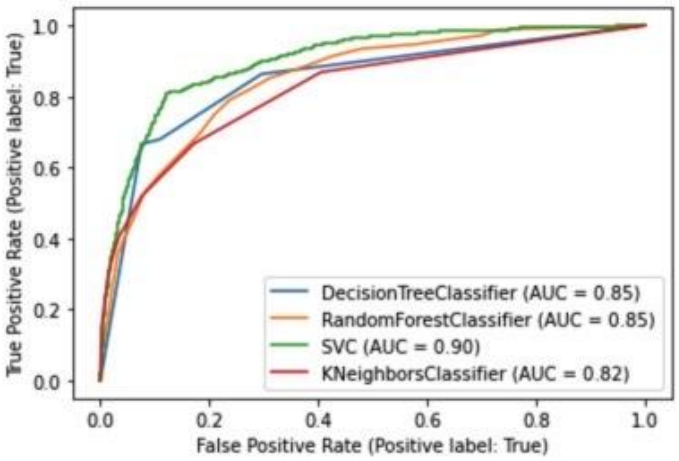
Machine Learning algorithm	**Accuracy (when Imbalanced)**	**Accuracy (After Oversampling)**	**Accuracy (After Undersampling)**
Decision Tree	66%	86%	96%
Random Forest	76%	96%	98%
Support Vector Machine	78%	82%	97%
K-Nearest Neighbour	88%	59%	61%

In case of Imbalanced dataset, K-Nearest neighbour performs the best whereas once the dataset is balanced using Oversampling/Undersampling Random Forest classifier is the best performer, followed by Support Vector Classifier and Decision Tree Classifier.

5. **Conclusion from ROC curves**

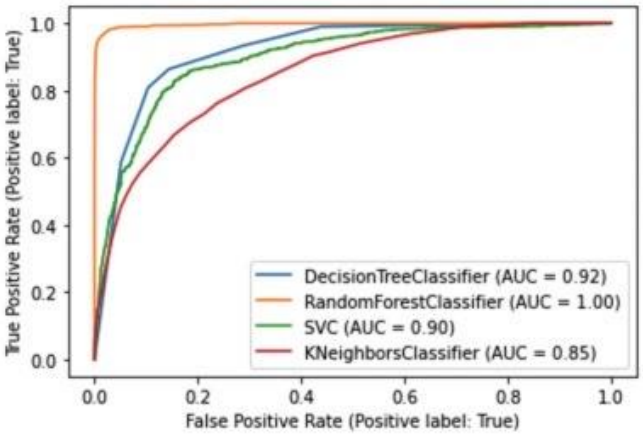
ROC curves show a trade-off between TPR (True Positive Rate) and FPR (False Positive Rate). Classifiers having curves nearer to top-left corner are an indication of better performance at predictions.

ROC curve for Imbalanced Dataset

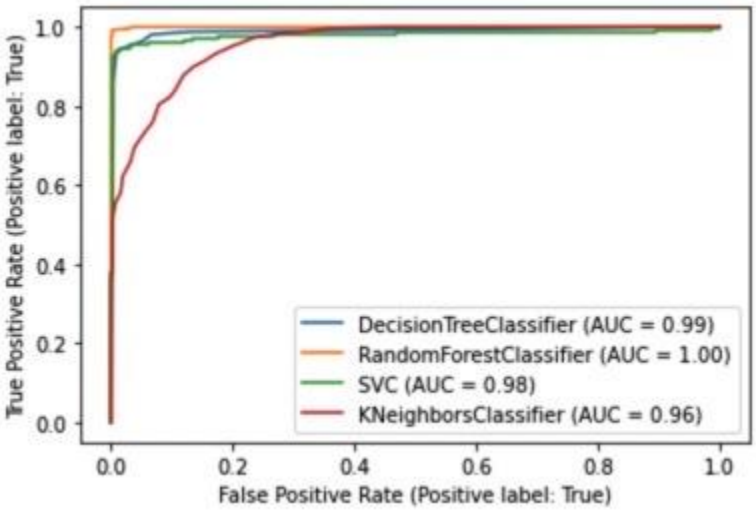


In case of imbalance dataset, Support Vector Machine (SVM) algorithm performs better than remaining algorithms with AUC score of 0.9, whereas KNN is not performing that good with a score of 0.82. Decision Tree and Random Forest Classifiers have similar performance per their AUC score while doing predictions.

ROC Curve after Oversampling Minority



ROC Curve after Undersampling Majority



It is evident from plotted ROC curves that Undersampling leads to better performance than Oversampling.

6. Comparison With reference paper

Algorithm	Accuracy (Research Paper)	Accuracy (My Implementation)
Random Forest Classifier	89.51%	98% (after Undersampling Majority Class)
Support Vector Machine (RBF)	84.88%	97% (after Undersampling Majority Class)

The comparison of accuracy for implemented algorithms is better than the classifiers used in the research paper.