Multiple ML Models on Prediction of Online Shoppers Purchasing Intension

Kangrui Li kl3350
Yuhang Wang yw3733
Linyang Han lh3096
Zheyuan Song zs2527

Abstract

This report is based on the requirements of the EECS6690 final project and our goal is to implement a literature review of a paper relating to machine learning and try to reproduce the results of the original paper. What's more, a variety of new methods will be used in this project. The original paper [1] discusses a real-time online shopper behavior analysis system consisting of several modules that simultaneously predicts a visitor's shopping intent and the likelihood of site abandonment. The following models are implemented and used in the modules: random forest(RF), support vector machine(SVM), multi-layer perceptron(MLP). Use the three models to predict a visitor's purchase intent and display a probability estimate of the visitor's intention to leave the site without completing a transaction within the predicted horizon. Use all the modules together to identify visitors who are in-market but likely to leave your site within the predicted range. Besides that, new methods are also successfully implemented online shoppers purchasing intention, covering LDA (Linear Discriminant Analysis), Neural Network, AdaBoost (Adaptive Boosting).

Keywords: *SVM; RandomForest; MLP; LSTM; KNN; DecsionTree; Bagging; Adaboost; Naive Bayes; Neural Network; LDA; Online shopper behavior; Clickstream data.*

1. INTRODUCTION

As the scale and scope of data collection in various fields continues to expand, machine learning has become an important tool for people to study data. By building models to process large-scale data and forecasts, it is possible to gain a deep understanding of the current situation and speculate on future events. Our report is mainly divided into the following parts: The first part introduces the background of the project and what we will do. The data set section will introduce the source of the data and the process of processing the data. The reproduction part of the original paper will describe the details of the article and code implementation. The data will then be processed using the new methods. Finally, compare the various data processing methods to draw conclusions and look forward to future work.

With the continuous rise of the Internet retail industry, more and more consumers choose to shop online. Using the behavioral data left by consumers on the Internet to predict purchase intentions is of great significance

for enterprises to achieve precision marketing[2]. For online shopping websites, the conversion rate for visitors who are interested in the product to those who make the purchases could be less than ideal. Finding areas where your website can be improved to convert more "visitors" into "buyers" becomes crucial. Not only that, but being able to more accurately predict potential "buyers" will allow websites to more appropriately distribute their ads. The goal of our project is to find a better model for predicting a visitor's purchase intention and to identify important features that contribute to the prediction.

2. DATASET

2.1. Data Description

The data we use in our project includes features such as how close a site visit was to a particular date, how many pages a visitor visited about a product, etc., extracted from access logs of online shopping sites.

According to 18 characteristics such as Google Analytics[3] indicators and basic user information, predict whether a visit (session) will be converted into a valid order. It is essentially a binary classification problem. The data set contains a total of 12330 sessions and 18 features. A session can be understood as a user visit, starting from entering the landing page until the user completes the final operation (such as leaving the webpage, closing the browser, etc.), this period is recorded as a session[4].

10 continuous features out of 18 features:

• Administrative: The number of pages related to account management viewed by the user

• Administrative_Duration: The duration of staying on the account management page (unit: second)

• Informational: the number of informational pages (such as contact information) that users have browsed

• Informational_Duration: The duration of staying on informational pages (unit: seconds)

•ProductRelated: The number of product-related pages viewed by the user

• ProductRelated_Duration: The duration of staying on the product page (unit: second)

• BounceRates: bounce rate

• ExitRates: exit rate

• PageValues: page value

• SpecialDay: Whether the access date is close to special days such as holidays (value 0 to 1)

And 8 discrete features:

• Month: Access month (10 months in total)

• OperatingSystems: operating system (8 categories in total)

• Browser: browser (13 categories in total)

• Region: region (9 categories in total)

• TrafficType: traffic source (20 in total)

• VisitorType: visitor type (3 types in total)

• Weekend: Whether the access time is a weekend (2 types in total)

• Revenue: whether to place an order (2 categories in total) this is the Y variable to be predicted

First clear the environment, import the library and data, and then check the data type and structure. Clean up the data, check for missing values and convert data types, and convert discrete features from int to factor. After that, EDA and visual analysis can be done on the processed data.

2.2. Data Visualization

In the data visualization section we created a heatmap to see the correlation between 10 numerical variables. We found a relatively high correlation between the type of page and the time spent on that page.

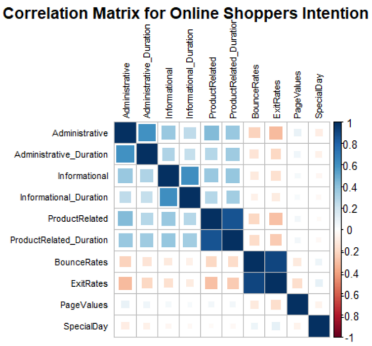

Figure1: Correlation Matrix for Online Shoppers Intention

Then we use pairs.panels to create the correlation, density and histogram of the two features. At this point we find that ProductRelated&ProductRelated_Duration, have the second highest correlation value.

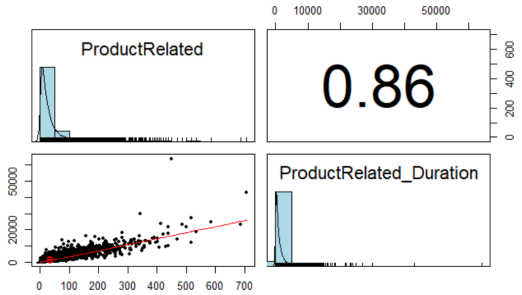

Figure2: Correlation of 2 features

Pairs.panels are used to display the highest correlation value of BounceRates and ExitRates.

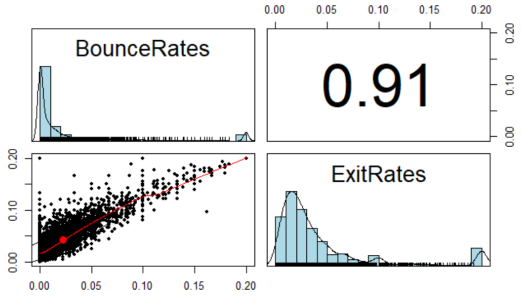

Figure3: Correlation of 2 features

Barplot is used to display popular shopping months and found out “May” is the most popular one.

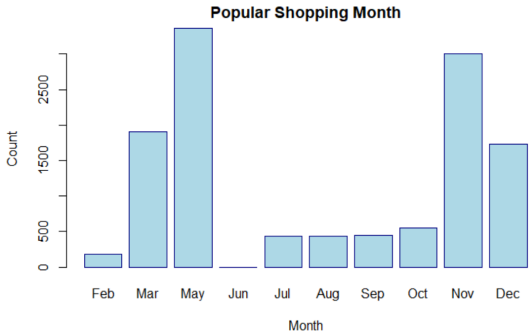

Figure4: Popular Shopping Month

Different types of visitors are displayed and it is found that t “returning_visitor” has many more numbers.

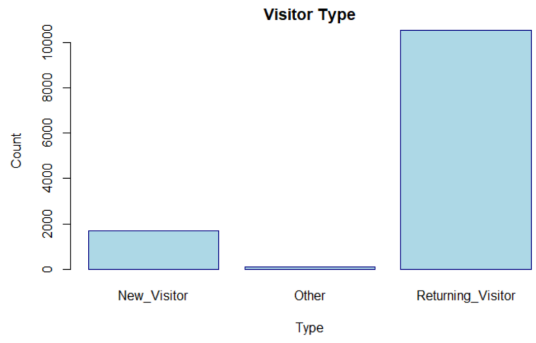

Figure5: Visitor Type

3. ORIGINAL PAPER REPRODUCTION

3.1. Literature Review

In the original paper, a real-time online shopper behavior analysis system is proposed. The proposed system uses aggregated page view data tracked during a visitor's visit along with some session and user information to predict a visitor's purchase intention [5]. The extracted features are fed back to Random Forest (RF), Support Vector Machine (SVM) and Multilayer Perceptron (MLP) classifiers as input. Oversampling and feature selection preprocessing steps are then used to improve the scalability and performance of the classifier. The results show that MLP computed using the elastic backpropagation algorithm with weight backtracking produces significantly higher accuracy and F1-score than RF and SVM. Another interesting finding is that clickstream data [6] obtained from navigational paths followed during online visits can be combined with session-based features that have unique information about purchasing interests to improve success rates. The analytics system supports the feasibility of accurate and scalable purchase intent prediction for virtual shopping environments using clickstream and session information data. The original paper uses online retailer data and compares the performance of various machine learning algorithms under different conditions. To sum up, we tried to reproduce the methods in the original paper, implemented new methods, analyzed their performance and completed the comprehensive analysis of the results.

3.2. Model Evaluation

There are 4 important evaluation metrics used in the original paper to estimate the performance: accuracy, true-positive rate, true-negative rate and F1-score.

Among them, accuracy is the most widely-used one, which is the percentage of the number of corrected prediction samples over whole samples. And the other three metrics formula are given below:

$$TPR = \frac{TP}{TP + FN}$$

$$TNR = \frac{TN}{TN + FP}$$

$$F1\ Score = 2 * \frac{precision * recall}{precision + recall}$$

where TP, TN, FP, FN represent the number of true positive samples, true negative samples, false positive samples and false negative samples respectively. And

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN}$$

3.3. Reproduced Results

In the original paper, mainly three models are applied: the support vector machines (SVM), random forest and multiple layer perceptron (MLP). And these models are used for the imbalanced datasets and oversampled datasets. We reproduced the above results and made some comparison in the following sections.

3.3.1 Support Vector Machine

Support vector machine (SVM) classifier is a very traditional but widely-used machine learning method, especially for supervised learning, which is the target of the original paper. In the original paper dataset, the training samples are labeled as belonging to two classes, so the SVM algorithm could build a model making a binary linear classification. Although SVM does not have a straightforward implementation for online learning, an online passive–aggressive implementation can be used to dynamically update the SVM model with new examples if it achieves significantly higher accuracies than the other classifiers used in the original paper. SVM is a discriminant-based algorithm which aims to find the optimal separation boundary called hyperplane to discriminate the classes from each other [7]. The closest samples to these hyperplanes are called support vectors, and the discriminant is represented as the weighted sum of this subset of samples which limits the complexity of the problem. The optimization problem to find an optimal separating hyperplane is defined as:

$$min\frac{1}{2}\left||w^2|\right| + C\sum_{i+1}^{k}\xi_i subject\ to\ r^t(w^T x^t + w_0) \geq 1 - \xi_i$$

Besides the linear SVM, we can also design kernel-based SVM with selected kernel function, with which we could make SVM capable of modeling nonlinear interactions by mapping the original input space into a higher dimensional feature space using a kernel function [9]. In the original paper, the radial basis function (RBF) kernel is applied, which is defined below:

$$K(x^t, x) = exp[-\frac{\left||x^t - x|\right|}{2s^2}]$$

So basically, we reproduced the two SVM classifiers with imbalanced and oversampled datasets. Here are some results comparisons:

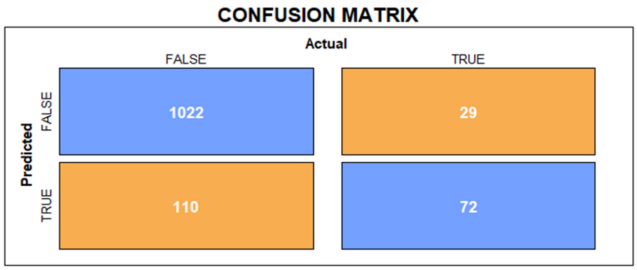

Figure6: Confusion matrix of linear SVM with imbalanced dataset

SVM vanilla

Model	Acc(%)	TPR	TNR	F1
Paper	88.3	0.42	0.97	0.52
Proposed	88.7	0.79	0.90	0.51

Table1: Evaluation Metrics of linear SVM with imbalanced dataset

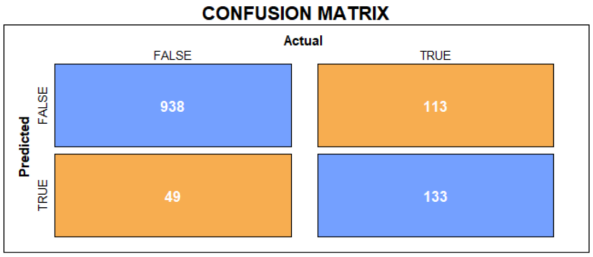

Figure7: Confusion matrix of linear SVM with oversampled dataset

SVM vanilla

Model	Acc(%)	TPR	TNR	F1
Paper	84.3	0.75	0.93	0.82
Proposed	87.5	0.54	0.95	0.63

Table2: Evaluation Metrics of linear SVM with oversampled dataset

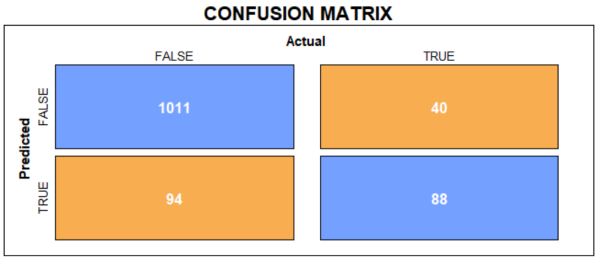

Figure8: Confusion matrix of RBF SVM with imbalanced dataset

SVM rbf

Model	Acc(%)	TPR	TNR	F1
Paper	86.14	0.46	0.92	0.53
Proposed	89.1	0.69	0.91	0.57

Table3: Evaluation Metrics of RBF SVM with imbalanced dataset

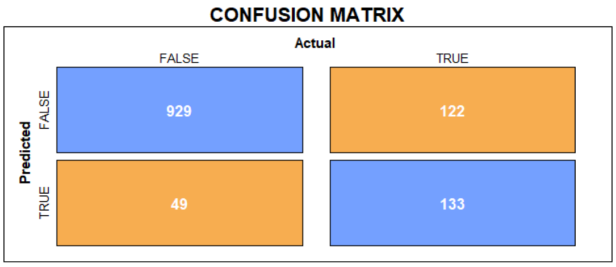

Figure9: Confusion matrix of RBF SVM with oversampled dataset

SVM rbf				
Model	Acc(%)	TPR	TNR	F1
Paper	84.9	0.75	0.94	0.82
Proposed	86.1	0.52	0.95	0.61

Table4: Evaluation Metrics of RBF SVM with oversampled dataset

From the results we can see that for both imbalanced and oversampled data, our reproduced model could achieve a better accuracy than the original paper and RBF kernel SVM could achieve the best performance with 89.1% accuracy.

3.3.2 Random Forest

Random forest is an ensemble method of tree-based models. It is based on constructing a forest, e.g., a set of diverse and accurate classification trees, using bagging resampling technique and combining the predictions of the individual trees using a voting strategy. The steps of the random forest construction algorithm are given below: Step 1: Given N instances in the original training set, create a subsample with bagging, i.e., choose N instances at random with replacement from the original data which constitutes the training set. Step 2: Suppose that each instance is represented with M input variables in the original input space. A number m is specified, which is much less than M, such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. Step 3: According to the predetermined stopping criteria, each tree is grown to the largest extent possible without pruning. Step 4: Repeat it until the desired number of trees is obtained for the forest.

This model is applied in the original study because it is proved to be effective in many classification tasks. In the model, the hyperparameter is set to be $log_2 M$, where M is the input and the parameter determines the size of each bag, and the number of trees is set to 100 [9].

We implement this method and here are the results of two datasets. It is great that we achieved nearly the same accuracy as the original paper with 89.5% and much higher accuracy when applying oversampling, with 89.9% compared to 82.3%.

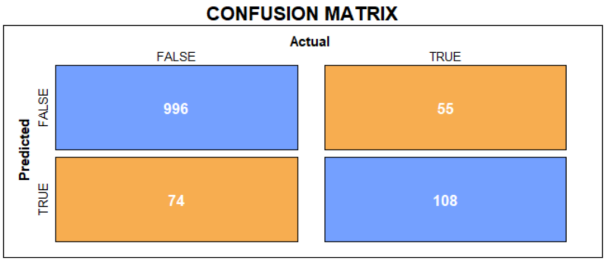

Figure10: Confusion matrix of Random Forest with imbalanced dataset

Random Forest				
Model	Acc(%)	TPR	TNR	F1
Paper	89.5	0.57	0.96	0.58
Proposed	89.5	0.66	0.93	0.63

Table5: Evaluation Metrics of Random Forest with imbalanced dataset

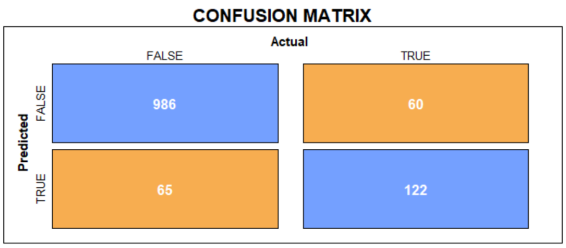

Figure11: Confusion matrix of Random Forest with oversampled dataset

Random Forest

Model	Acc(%)	TPR	TNR	F1
Paper	82.3	0.74	0.90	0.81
Proposed	89.9	0.67	0.94	0.66

Table6: Evaluation Metrics of Random Forest with oversampled dataset

3.3.3 Multiple Layer Perceptron

MLP is a feedforward artificial neural network model that is composed of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. The elements of the hidden and output layers are called neurons, which are a set of processing units. Both the MLP model in the original paper and in our reproduced design consist of an input, an output, and a single hidden layer. MLP is capable of modeling complex nonlinear problems with the use of a nonlinear activation function in its hidden layer [10].

The metric of errors would be calculated with the given equation and the output of the network is calculated as below:

$$E(W, v|X) = \sum_{t=1}^{T} (r^t - y^t)^2$$

$$y^t = \sum_{h=1}^{H} v_h z_h^t + v_0$$

where W and v denote the set of first- and second-layer weights, respectively, T the number of training set samples, r^t the actual value of sample t, and r^t the output of the network, i.e., the predicted value, for sample t. And the updates of parameters are calculated with gradient descent method and chain rule deduction.

In our reproduced experiment, we implemented this model with 40 hidden layers with an imbalanced dataset and an oversampled one, because we do think a deep structure could achieve a good performance with the computational power guaranteed. Here are the confusion matrix and evaluation metrics of both two datasets. From the results, we can see that our reproduced model can achieve 86.7% accuracy with imbalanced data, which is only a little bit lower than the original results. And for the oversampled one, we could obtain 88.4% accuracy, which is much higher than the original paper.

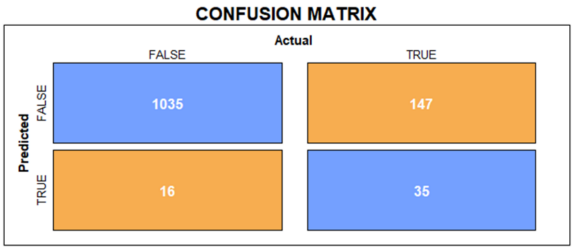

Figure12: Confusion matrix of MLP with imbalanced dataset

MLP(# of hidden layers: 40)

Model	Acc(%)	TPR	TNR	F1
Original Paper	87.1	0.54	0.93	0.56
Proposed	86.7	0.19	0.98	0.30

Table7: Evaluation Metrics of MLP with imbalanced dataset

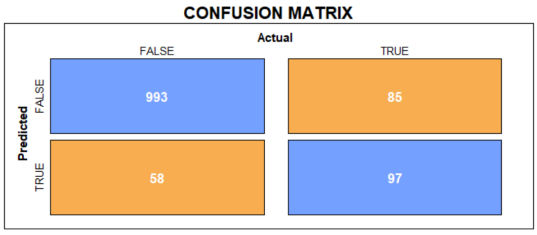

Figure13: Confusion matrix of MLP with oversampled dataset

MLP(# of hidden layers: 40)

Model	Acc(%)	TPR	TNR	F1
Original Paper	82.2	0.82	0.83	0.82
Proposed	88.4	0.53	0.94	0.58

Table8: Evaluation Metrics of MLP with oversampled dataset

4. OTHER METHODS BEYOND THE ORIGINAL PAPER

We also implemented several other models beyond the original paper and the details are given below.

4.1 Decision Tree (without pruning)

The first algorithm we tried to fit on the data is the decision tree, which is a tree-variant classification model. Generally, Decision tree is an efficient nonparametric method that can be used for both classification and regression.[11] A decision tree has two main components: internal decision nodes and terminal leaves. For the node that won't reach the leaf, it will be branched based on some criterion. The whole process is like this: first an impure internal node is selected, then the reduction in impurity is estimated based on every feature, after which the feature with maximum reduction to create two child nodes are chosen so the tree can be branched, then we make judgment that if

all the stopping criteria meets for all nodes. A decision tree is built with multiple repeats of the above processing.

So basically, the decision tree can be applied to our task and we built a tree from top to end and it seems to have multiple branches because we increased the number of nodes greedily. Here are the results of our plain decision tree without pruning, showing that the accuracy could be 84.8 %.

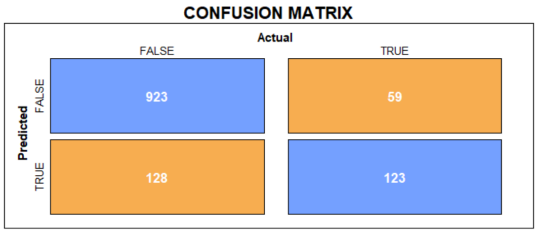

Figure14: Confusion matrix of Decision Tree with oversampled dataset

Decision Tree

Classifier	Acc(%)	TPR	TNR	F1
Proposed	84.8	0.68	0.88	0.56

Table9: Evaluation Metrics of Decision Tree with oversampled dataset

4.2 Decision Tree (with pruning)

Pruning is widely used in decision trees because of the problem of overfitting. With pruning, each node would be replaced with the most popular class, from which we can reconstruct this tree. By doing so, we greatly decrease the dimension of trees and we found the optimal size of leaves in the pruned tree is 6. For pruning trees, the accuracy is 86.2 %, TNR is 0.88 and F1 score is 0.56.

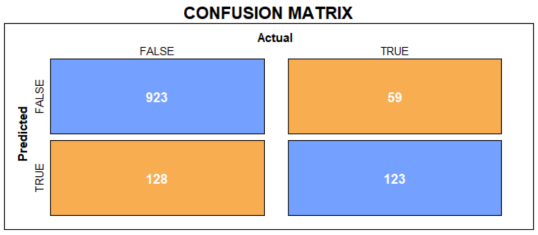

Figure15: Confusion matrix of pruned Decision Tree with oversampled dataset

Decision Tree

Classifier	Acc(%)	TPR	TNR	F1
Proposed	86.2	0.68	0.88	0.56

Table10: Evaluation Metrics of pruned Decision Tree with oversampled dataset

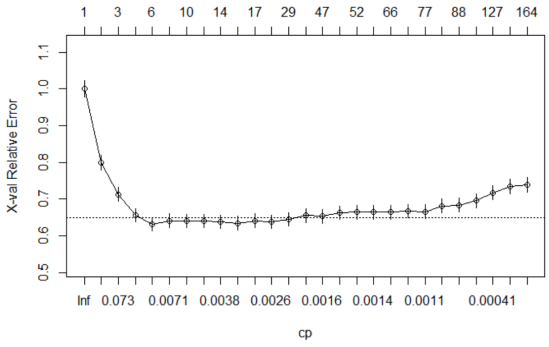

Figure16: Cross-Validation of pruning process

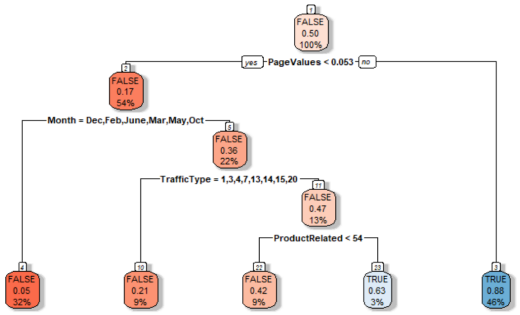

Figure17: Decision Tree with pruning

4.3 K-nearest neighbor

K-nearest Neighbors are widely used in classification and clustering tasks. Based on KNN method, an input consists of the k closest training examples in a dataset and the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.[12]

In our design, we applied the library "kknn" in R and set the kernel and distance to be triangular and 1 respectively. We trained the model on oversampling datasets and here are the results. The confusion matrix is given below and the training accuracy is 84.2%.

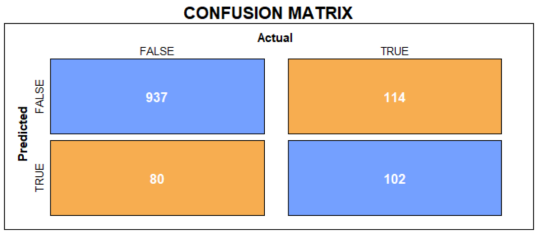

Figure18: Confusion matrix of KNN with oversampled dataset

KNN

Classifier	Acc(%)	TPR	TNR	F1
Proposed	84.2	0.47	0.92	0.62

Table11: Evaluation Metrics of KNN with oversampled dataset

4.4 Deep Neural Net

There are multiple breakthroughs and outcomes in Deep Neural Network and its performance of classification has been proved to be effective. A complex model like deep network could perform well with more parameters, also causing some problems like overfitting. Based on this idea, we tried to build a deep neural network to work on our target dataset and to solve the overfitting problem we introduced a dropout layer.

In our designed deep neural network, as shown in Figure 19, we basically implemented the architecture consisting of 3 dense layers, 2 dropout layers and some activation function. For the structure, the input is first fed into a dense layer of 400 units, then it is processed by the 'Relu' activation and dropout layers. The output is then used as the input of another combination of dense layer, activation and dropout layers. Finally, this output is scaled into 2 classes and activated by the 'Softmax' function to make classification.

With this predefined model, we trained our model with oversampling training datasets by specifying the optimizer, loss function and other hyperparameters. The training and

validation process is given as Figure 20. From the accuracy and loss curve, it is clear that our model converges well and could achieve good performance.

We also use this trained model to make predictions of the test datasets and the results of the confusion matrix and some evaluation metrics are given below: we achieved prediction accuracy of more than 90%, which is the highest of all the models we designed.

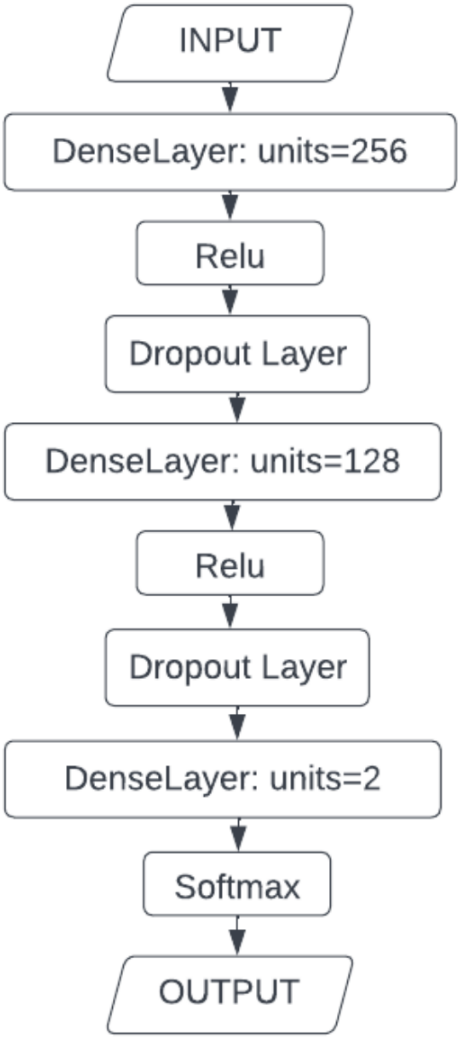

Figure19: Deep Neural Net Architecture

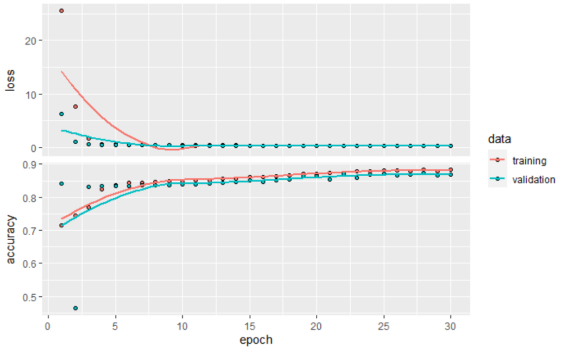

Figure20: Accuracy and Loss while training and validation

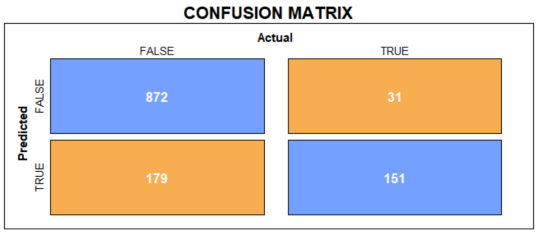

Figure21: Confusion matrix of Deep Neural Net with oversampled dataset

Neural Net

Classifier	Acc(%)
Proposed	90.2

Table12: Evaluation Metrics of Deep Neural Net with oversampled dataset

4.5 Naive Bayesian

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models, but coupled with kernel

density estimation, they can achieve high accuracy levels. But in our project, Naïve Bayes' accuracy is just about 64.2%.

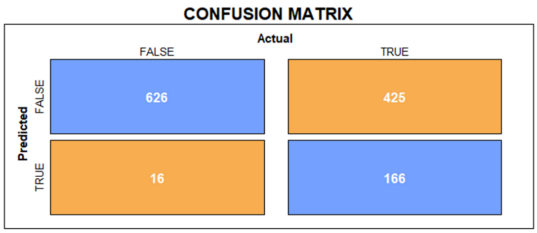

Figure22: Confusion matrix of Naive Bayes

Bayes

Classifier	Acc(%)	TPR	TNR	F1
Proposed	64.2	0.29	0.98	0.42

Table13: Evaluation Metrics of Naive Bayes with oversampled dataset

4.6 Adaboost

AdaBoost, short for Adaptive Boosting, is a statistical classification meta-algorithm formulated by Yoav Freund and Robert Schapire in 1995, who won the 2003 Gödel Prize for their work. It can be used in conjunction with many other types of learning algorithms to improve performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. Usually, AdaBoost is presented for binary classification, although it can be generalized to multiple classes or bounded intervals on the real line.

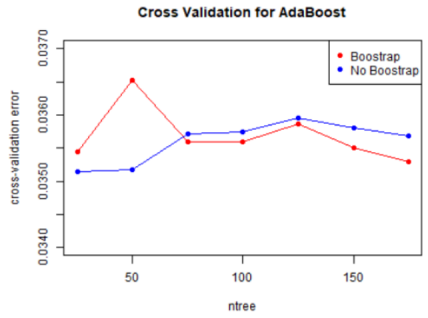

Figure23: Cross Validation for AdaBoost

First we test a different parameter model's error rate. From the above digraph we can see that when the tree is 50 and doesn't use bootstrap we can get a suitable error rate. After setting up our model, we get the result shown down here. We get a very high accuracy(89.1%) and F1 Score(0.61), which means this model can handle this problem well.

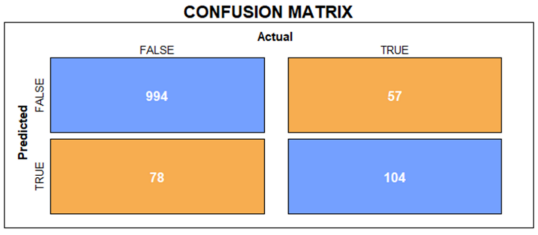

Figure24: Confusion matrix of Adaboost

Adaboost

Classifier	Acc(%)	TPR	TNR	F1
Proposed	89.1	0.65	0.93	0.61

Table14: Evaluation Metrics of Adaboost with oversampled dataset

4.7 LDA

LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities. In an oversampling dataset, LDA's accuracy is not that high, just 77.9%. LDA and Naïve Bayes' result shows that maybe some class is not a linear relationship in our dataset.

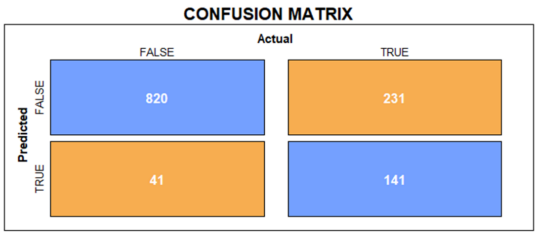

Figure25: Confusion matrix of LDA

Adaboost

Classifier	Acc(%)	TPR	TNR	F1
Proposed	77.9	0.38	0.95	0.51

Table15: Evaluation Metrics of LDA with oversampled dataset

4.8 Bagging

Bagging algorithm (bootstrap aggregation) is a method to establish a basic classifier on each self-service sample set and obtain the final category of test samples through voting allocation.

By randomly selecting samples from the returned dataset and generating multiple self-service sample sets, the size of each self-service sample set is consistent with the original data set. Therefore, some samples may appear multiple times in the same self-service sample set. Train one basic trainee for each self-service sample set. A common basic learner is a binary decision tree. For problems with complex decision boundaries, the performance of binary decision trees is unstable, which can be overcome by combining multiple decision tree models. Finally, for the regression problem, the result is the average value of basic learners. For the classification problem, the result is the probability or average value of each category obtained from the percentage of different categories.

Our algorithm uses the bagging() function in the R language adabag library to implement the bagging algorithm, and the base classifier selected in this function is a tree. We train the model according to the divided training set, and then analyze the results, listing the importance of each parameter, confusion matrix and the accuracy of the algorithm. Unmentioned feature classes like OperatingSystems, VisitorType and so on show no significant relationship with the prediction of Revenue feature.

Feature Class	Importance
Administrative_Duration	0.33228264
Informational_Duration	0.03451416
ProductRelated	1.61764305
BounceRates	6.08534149
Month	3.35516073
ProductRelated_Duration	0.43758439
Informational	0.03595865
PageValues	86.78681809
TrafficType	0.37372105

Table16: Feature class's importance

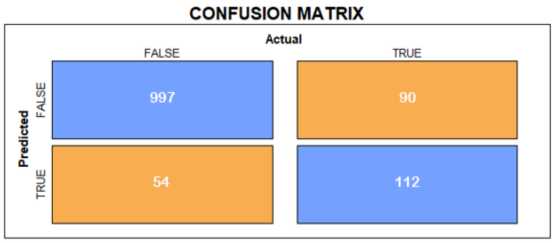

Figure26: Confusion matrix of bagging feature with oversampled dataset

Bagging Accuracy

Classifier	Error Rate(%)	Accuracy(%)
Proposed	10.1	89.9

Table17: Evaluation Metrics of bagging accuracy with oversampled dataset

4.9 Ensemble Method

In machine learning algorithms, ensemble represents the combination of multiple machine learning algorithms, just like its literal meaning. Given a group of classifiers, each of these classifiers has different functions, principles and properties. With the combination of these classifiers, we can make the algorithm have new advantages and properties, as well as higher prediction performance than a single classifier.

The combination of classifiers usually has two strategies: average method and boosting method. Bagging and adaboost belong to these two strategies respectively. The former uses multiple classifiers for average estimation, and the result is more accurate by reducing variance; The latter combines different classifiers in order to build and optimize a powerful model through simple models. The bagging and adaboost methods in adabag previously used in this paper are based on the combination of decision trees: bagging trains multiple models by extracting and creating multiple data sets, and finally combines these models; adaboost method is to adjust the weight of multiple models according to the error rate to finally combine a better classifier.

Here we use the boosting method, which includes three algorithms we used in this paper previously: support vector machine, random forest, and multilayer perceptron (ANN). Let them process the model sequentially to produce an effect superior to the three alone. From the results (the accuracy is up to about 93%), we can also see that this combination algorithm has a great effect. The following is the representation of the resemble algorithm

Ensemble method		
Classifier	Accuracy(%)	F1 Score
Proposed	92.7	0.73

Table18: Evaluation Metrics of ensemble method with oversampled dataset

4.10 Feature Selection

Generally speaking, a machine learning task will predict one or more values or states according to the complex characteristics of samples. It can be said that the number, correlation and other attributes of various eigenvalues are closely related to the results of machine learning and the selection of models. Generally speaking, we will encounter a situation where the number of sample eigenvalues is higher than the number of actual effective eigenvalues for prediction. Then we need to identify those feature values that are more effective, that is, exclude irrelevant features and more than features, so as to optimize the training and prediction results.

Among them, irrelevant features refer to the feature value that has nothing to do with the prediction. For example, if we predict the stock price, then the gender ratio of shareholders is an irrelevant feature. The second is the redundant feature, which refers to the redundant eigenvalue. When we have shown the influence of a factor through an eigenvalue, if other eigenvalues are set for this factor, then this factor may be the redundant feature. Take forecasting stock prices as an example. If we have stock prices and transaction prices at each time, then one of them is redundant, because they describe the same factor: the stock price at that time.

The significance of reducing features is very important. It can reduce overfitting to make the model more reliable and can be applied to a wider range of scenes; It can reduce the dimension of the model and make the training faster and more accurate; You can also make the model more readable.

However, it is difficult to select the most effective feature set from a large number of features only based on the common sense of the experimenter, so it needs some engineering methods to help. Here we use the Random Forest model without oversampling to determine the effectiveness of the eigenvalues in the dataset. Following is the result figure of feature importance.

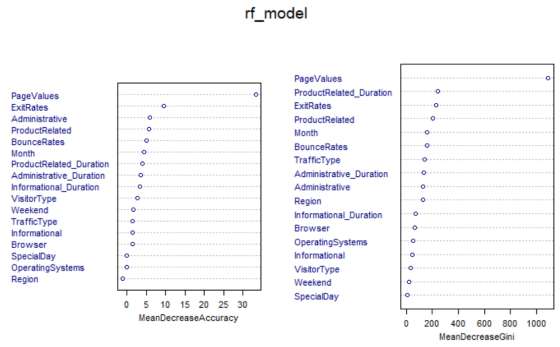

Figure27: Result of feature importance

Next, we verify the results of feature analysis. We use the Gini coefficient as a reference to judge the accuracy of the model prediction. From the test results, we can see that when the dependence of prediction on PageValues decreases, the Gini coefficient, that is, the performance of the model decreases rapidly, while other features do not change so quickly. From the decreasing speed of Gini coefficient, that is, the

weakening speed of model performance, we can see that our previous analysis of the importance of eigenvalues is correct. And compare the importance of the features we output in the bagging method. Because the model strategies are different, we exclude some small differences in the weak correlation features, and the results are basically consistent. For example, PageValues is the most important factor that determines Revenue far exceeds other features. And the number of pages a viewer goes through and the time length of a viewer looking at related pages will also be relatively important factors to decide whether they will make a final purchase. Followings are partial dependence figures of different factors which represent Gini coefficient shifts.

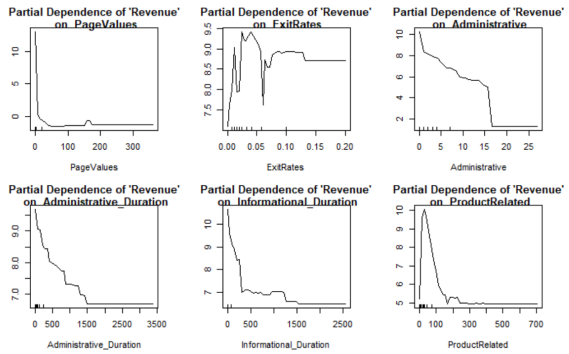

Figure28: Different factors which represent Gini coefficient shifts

5. CONCLUSION AND DISCUSSION

5.1. Results Comparison

	Imbalanced data		Oversampled data	
model	Acc(%)	F1	Acc(%)	F1
SVM(rbf)	89.1	0.57	86.1	0.61
Random Forest	89.5	0.58	89.9	0.66
MLP	86.7	0.56	88.4	0.58
Decision Tree	89.8	0.65	84.8	0.56
KNN	87.4	0.64	84.2	0.62
Deep Neural Net	90.2	*	87.3	*
Naive Bayes	81.8	0.54	64.2	0.42
Adaboost	89.1	0.61	89.1	0.61
LDA	88.8	0.49	77.9	0.51
Bagging	*	*	89.9	0.67
Ensemble method	*	*	**92.1**	**0.73**

Table 19: Results comparison

5.2. Model Prediction

Oversampling may help with improving the performance of the models. The Ensemble using the prediction of the three models yields the best accuracy and f1 score among all.

5.3. Feature Selection

Salient features are: Page Value; Product Related (number of related pages have viewed); Product Related Duration(total time spent on related pages). We can

hypothesize that the more attractive the pages are, the more pages the viewer reads, the more time the viewer spends on the pages, the more likely they are going to make a purchase.

6. FUTURE WORK

For future research, different methods of data augmentation should be experimented to improve model performance. Do more feature importance analysis in different models. More parameter tuning should be done to improve the performance of the models. We could try more ensemble strategies and combinations of models to reach a better score and accuracy.

ACKNOWLEDGMENT

Thanks to Professor Predrag Jelenkovic and all teaching assistants for patiently and enthusiastically teaching us to learn machine learning algorithms and R language, which provided us with sufficient basic knowledge for project research.

REFERENCES

[1]Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. Neural Comput & Applic 31, 6893–6908 (2019).

[2]Zhang Jingxuan, Zhang Weiwei. Prediction of Repurchase Behavior of E-commerce Consumers Based on Neural Network [J]. Advances in Applied Mathematics, 2021, 10(10): 3374-3380.

[3]Clifton B (2012) Advanced web metrics with Google Analytics. Wiley, New York.

[4]Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Neural Comput & Applic (2018).

[5]Moe WW (2003) Buying, searching, or browsing: differentiating between online shoppers using in-store navigational clickstream.J Consum Psychol 13(1–2):29–39.

[6]Budnikas G (2015) Computerised recommendations on e-transaction finalisation by means of machine learning. Stat Transit New Ser 16(2):309–322.

[7]Salcedo-Sanz S, Rojo-A´ lvarez JL, Martı´nez-Ramo´n M, CampsValls G (2014) Support vector machines in engineering: an overview. Wiley Interdiscip Rev Data Min Knowl Discov 4(3):234–267

[8]Vapnik V (2013) The nature of statistical learning theory. Springer, Berlin

[9]Breiman L (2001) Random forests. Mach Learn 45(1):5–32

[10]Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366

[11]Tan PN (2006) Introduction to data mining. Pearson Education, New Delhi

[12]Fix, Evelyn; Hodges, Joseph L. (1951). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties (PDF) (Report). USAF School of Aviation Medicine, Randolph Field, Texas. Archived from the original on September 26, 2020.