Predicting Online Shopper Intentions

Shilong Dai, Patrick Schmitt, Puyao Ge

Introduction

Our analysis is of the dataset from the paper "Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks" by C. Okan Sakar, S. Oakley Polat, Mete Katircioglu and Yomi Kastro in *Neural Computing and Applications*. The data records specific web sessions of users of a Turkish general purpose e-commerce store. The aim is to predict whether shoppers will purchase anything during a session to tailor store recommendations. We were able to successfully separate part of the data via transformations. We applied PCA and KMeans to identify patterns in the dataset, and attempted several supervised classification methods. We found that QDA and RBF SVM performed well on the testing set. Our methods outperformed the reported metrics in the comparable models present in the aforementioned paper. Thus, they may be useful as components of a recommendation system.

Data

Overview

There are 10 numerical variables and 8 categorical variables, one of which is the boolean "Revenue", indicating whether the customer purchased anything during that session. There are 12,330 sessions in the dataset. The empirical priors are that ~85% of sessions do not result in revenue and 15% do.

Feature name	Feature description	Min. value	Max. value	SD
Administrative	# of administrative pages visited in a session	0	27	3.32
Administrative duration	Amount of time spent in the administrative pages in second	0	3398	176.70
Informational	# of informational pages visited by the visitor. These pages contain information about the platform, address etc.	0	24	1.26

Informational duration	Total amount of time (in seconds) spent by the visitor on informational pages	0	2549	140.64
Product related	Number of pages visited by visitor about product related pages	0	705	44.45
Product related duration	Total amount of time (in seconds) spent by the visitor on product related pages	0	63973	1912.25
Bounce rate	Google analytic variable. See below.	0	0.2	0.04
Exit rate	Google analytic variable. See below.	0	0.2	0.05
Page value	Google analytic variable. See below.	0	361	18.55
Special day	Closeness of the site visiting time to a special day	0	1.0	0.19

Table 1 Numerical features found in the dataset

Table 1 shows the numerical features along with their statistical parameters. The bounce rate is defined as the number of single page sessions divided by all sessions that start with a given page for each page. Hence, it tracks the sessions where a user enters the site on a given page, and exits immediately. On the other hand, exit rate is the percentage of view that would be the last in the session, calculated for each page. Hence, the exit rate can identify the outlet for the site in user sessions. The page values metric is more e-commerce centric. It keeps track of the unique page view for each page, and the commerce transactions during which the user passed through the page. For each page, the total commerce value is divided by its unique page views. Hence, for a page that is often on the path to a purchase would have a higher page value. In the data, the Google analytics variables are the average of the pages during the session. Finally, the variable special day is a measure of distance in days from the session to the nearest holiday. It is normalized by the length of an e-commerce transaction from the payment to delivery decided by the e-commerce firm. Besides the numerical variables, the following categorical variables are also present in the dataset.

Feature	Description	Levels
Operating System	The OS of the visitor reported to the site	8
Browser	The browser of the visitor reported to the site	13
Region	The geographical region of the visitor	20
Visitor Type	New visitor, returning visitor, or other	3

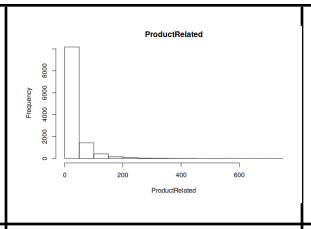
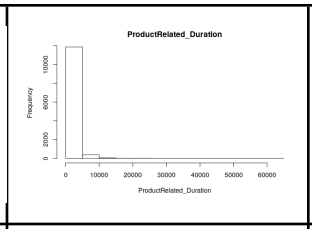
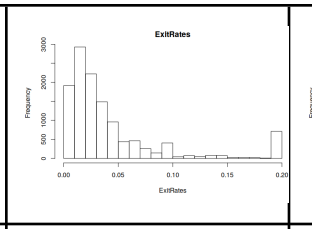
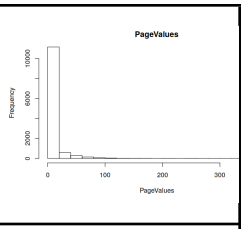
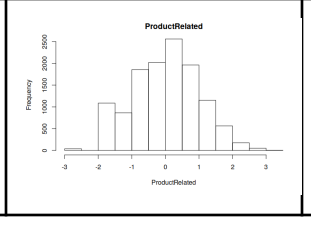
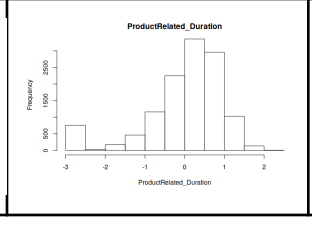
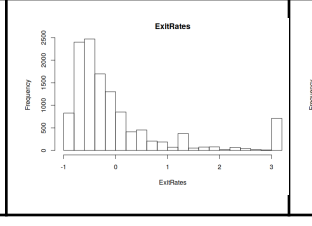
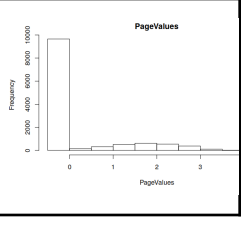
Weekend	Whether the session was during weekend	2
Traffic Type	The source of the visitor, whether from banner, text message, direct link, etc	20
Month	The month of the session	12
Revenue	Whether the session resulted in a purchase	2

The provider of the dataset also took care in ensuring the sanity and completeness of the data. Each record was selected from the database of the platform such that for each user, only one transaction is present. This ensures a greater independence between the entries.

A potential concern is data leakage. This is worth considering for the Google analytics variables, since the values came from every user who visited the site. However, the other variables are separate for each user and each session. Furthermore, given that the Google analytics values are available at all times, and especially before the user makes the decision to purchase, it could still be used before a purchase is made. Hence, it would be fair game for our model to use.

Transformations

We found that all of the numerical variables cover multiple orders of magnitude and are entirely positive. Furthermore, they are all strongly skewed to the right. So, a logarithmic transformation (x->log(x+1) since there are 0s in most columns) is applied. Since we intend to apply SVM and KMeans to the dataset later, we also standardized the numerical features by converting each value to its associated z-score. Below is a sample of feature histograms:

before				
after				

The features Administrative Duration, Informational Duration, Product Related, Product Related Duration, and Page Values become either normal, or normal with a concentration of values near 0. The features Administrative, Informational, and ExitRates appear to be skewed

normal distributions. The transformation is unsuccessful on the Bounce Rates, and Special Days feature, and they are still notably skewed. This transformation is useful for methods like linear and quadratic discriminant analysis, which assume a normal distribution of features. It also makes visualizations easier to interpret.

Learning Methods

Principal Component Analysis

We intended to apply PCA to the numerical features in attempts to find an alternative subspace or representation which has sharper distinctions between the patterns in the data. It also allows visualization of the numerical data. Since PCA is not concerned about any categorical variable of interest, it is an unsupervised learning method.

PCA works by minimizing the distance of each data point to the chosen subspace V spanned by the principal component vectors forming a basis. The error function is given below.

$$\text{Err}(\{\mathbf{x}_i\}, V) = \sum_{i=1}^{n} \|\mathbf{x}_i - \text{proj}_V(\mathbf{x}_i)\|^2$$

It can be shown that the optimal solution is the eigen basis of the sample covariance matrix. Furthermore, the eigenvalues represent the variance of the projected data points on the corresponding eigenvector. Hence, the principal component is said to preserve the variance of the original data, and the quality can be measured by:

$$\gamma_k = \frac{\sum_{i=1}^{n} \|\text{proj}_{V_k}(\mathbf{x}_i)\|^2}{\sum_{i=1}^{n} \|\mathbf{x}_i\|^2} = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{j=1}^{p} \lambda_j}$$

It is said to be the proportion of variance explained by the given principal component.

KMeans

Besides PCA, we also intended to try KMeans clustering with Euclidean distance on the dataset in order to find patterns that may be helpful. The method is very intuitive. However, the downside is that we can only examine numerical variables. With the distance metric, the KMeans algorithm finds the cluster by picking K center points such that all points in a cluster are the closest to its center.

$$A_j = \{\mathbf{x} : ||\mathbf{x} - \mathbf{c}_j|| \leq ||\mathbf{x} - \mathbf{c}_s|| \text{ all } s \neq j\}$$

Naturally, a metric for the fit of the cluster is the sum of the distance between each point to its center.

$$\text{Cost}(\mathbf{c}_1, \ldots, \mathbf{c}_k) \;=\; \sum_{i=1}^{n} \min_{1 \le j \le k} ||\mathbf{x}_i - \mathbf{c}_j||^2$$

Realistically, it is infeasible to find the true centers, but there are algorithms to iteratively approximate the centers.

QDA

Quadratic discriminant analysis (QDA) provides a parametric approach. Like LDA, the QDA classifier results from assuming that the observations from each class are drawn from a Gaussian distribution, and plugging estimates for the parameters into Bayes' theorem in order to perform prediction. However, unlike LDA, QDA assumes that each class has its own covariance matrix. That is, it assumes that an observation from the kth class is of the form $f_k = \mathcal{N}_d(\mu_k, \Sigma_k)$, where Σ_k is a covariance matrix for the kth class. The discriminant functions $\delta_k(x) = \log(\pi_k\, f_k(x))$ have the form

$$\delta_k(x) = -\frac{1}{2} x^t \Sigma_k^{-1} x + \langle x, \Sigma_k^{-1} \mu_k \rangle - \frac{1}{2} \left[\log(2\pi)^d \pi_k^{-2} \det(\Sigma_k) + \mu_k^t \Sigma_k^{-1} \mu_k \right]$$

Therefore, the decision boundary of for this method is a quadratic surface

$$B = \left\{ x : -\frac{1}{2} x^t (\Sigma_1^{-1} - \Sigma_0^{-1}) x + x^t (\Sigma_1^{-1} \mu_1 - \Sigma_0^{-1} \mu_0) + (c_0 - c_1) = 0 \right\}$$

.

Logistic Regression

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. It assumes a linear relationship between the log odds ratio between the classes.

$$\log \frac{\eta(x)}{1 - \eta(x)} \;=\; \beta_0 + \sum_{i=1}^{d} \beta_i\, x_i \;=\; \langle \beta, x \rangle$$

Thus, the problem of fitting a logistic curve over the observation is reduced to finding β, which are the coefficients in the linear relationship. In practice, the fitting is done via the maximum likelihood optimization, which can be expressed as minimizing negative log likelihood:

$$\mathrm{L}_{log} = -\ln(L) = -\sum_{i=1}^{N} \left[-\ln(1 + e^{(\beta_0 + \beta_1 x_i)}) + y_i\, (\beta_0 + \beta_1 x_i) \right]$$

In our case, we are more interested in using logistic regression to select the features. Hence, we adopted a penalized modification (LASSO) to prune the feature set.

LASSO

LASSO (least absolute shrinkage and selection operator) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. It adds an L1 loss with respect to the size of the coefficients to the loss function, so that the optimizer must trade off between model fit and size of coefficients. In the case of logistic regression, the negative log likelihood to minimize now becomes

$$L_{log} + \lambda \sum_{j=1}^{p} |\beta_j|$$

The parameter λ controls the trade off between the size of coefficient and the fit. Due to the geometry of the L1 loss, the coefficient would be set to zero if the loss induced by its magnitude is too high. Therefore, we can use it to guide our feature selection by examining the coefficients not set to 0.

SVM

Linear

Support vector machine (SVM) classifier is a discriminant-based algorithm which aims to find the optimal separation boundary called hyperplane to discriminate the classes from each other. 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:

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

where w is a weight vector defining the discriminant, C the regularization parameter, $\xi = (\xi_1, \xi_2, \ldots, \xi_k)$ vector of slack variables, and r^t the actual value of sample t. The slack variables are defined to tolerate the error on the training set in order to avoid overfitting and so improve the generalization ability of the model. The regularization (cost) parameter, C, is a hyperparameter of the algorithm which is used to control the complexity of the model that is fitted to the data. Higher values of C decrease the tolerance of the model on training set instances and hence may cause overfitting on the training set.

RBF

Although SVM is a linear classifier, it is capable of modeling nonlinear interactions by mapping the original input space into a higher dimensional feature space using a kernel function. Thus, the

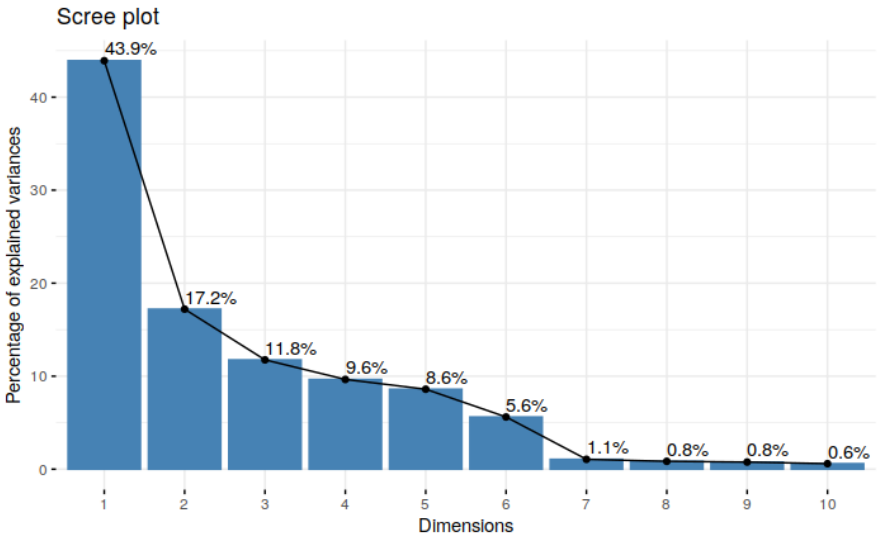
linear model in the new space corresponds to a nonlinear model in the original space. In this study, linear and radial basis function (RBF) kernels are used. The RBF is defined as

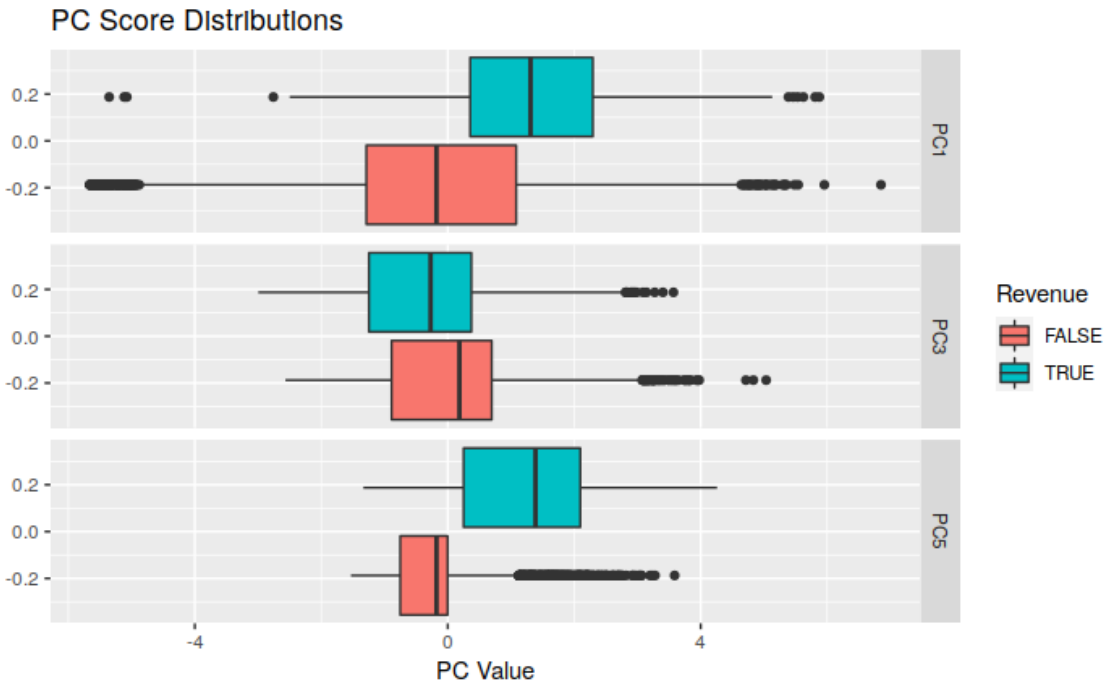
$$K(\boldsymbol{x}^t, \boldsymbol{x}) = \exp\left[-\frac{\|\boldsymbol{x}^t - \boldsymbol{x}\|}{2s^2}\right]$$

where $\boldsymbol{x}^t$ is the center and s defines the radius. To avoid overfitting and report unbiased results, the values of hyperparameters, C and s, are optimized using grid search on a randomly selected single train/validation partition, and the specified values are used for the rest of the partitions.

Unsupervised Results

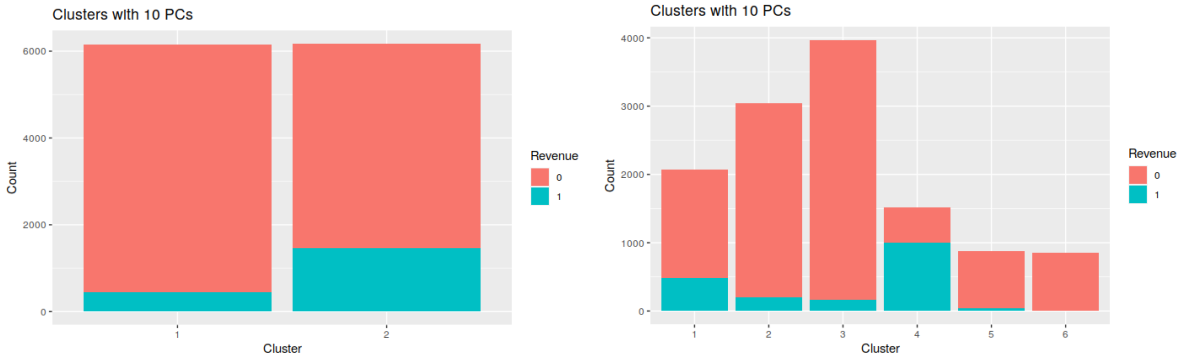
We found that the first 6 PCs were able to take into account 97% of variability in the numerical features of the transformed dataset. On an interesting note, the first PC was able to explain about 98% of variations on the untransformed data set. This shows that the log transformation with standardization was able to bring out more details from the dataset, so that it is no longer essentially one dimensional.





Then, we plotted the PCs against Revenue to see if a change of basis can reduce the noise in the dataset and differentiate between the classes of our concern. We found that PC1, PC3, and PC5 show some separation between the classes. However, the most drastic difference is in PC5, which only explains about 9% of the variation in the dataset. Hence, it is evident that Revenue does not align with the most clear and distinct variations in the dataset.

Afterwards, we applied KMeans to the numerical transformed features as well. We first attempted K = 2, since there are 2 levels in Revenue. Then, we applied the elbow method, which involved looking at a chart of the total sum of squares for each K and deciding on a K with an appropriate sum of squares relative to the others. In our case, we found K = 6 to be a reasonable choice from the sum of squares chart. We plotted the clusters against Revenue to see how well the patterns in the dataset align with our categorical variable of interest.



In the case of K = 2, KMeans was able to produce 2 clusters of similar sum of squared errors and size. It can be seen that cluster 2 had more buying customers than cluster 1. However, in both cases, the buying customers are still in the minority. In the case of K = 6, the sum of squared

errors and size are more diverse, and we found one cluster with mostly buying customers. Furthermore, the cluster 4 also has only 14% of the sum of squared error compared to all of the clusters. Looking at the center of cluster 4, the most distinguishing fact is that it has a page value significantly higher than the other clusters. This is in consistency with the description from Google analytics, since page values would only be high on the pages that lead to purchase.

We ran a similar assessment with clustering on the PCs. However, in this case, the PCs did not generate clusters that are much different in composition or size.

Feature Selection

We determined the features to include in our supervised models in the end via a mixed approach of examining plots, PCA vectors, LASSO regression, and cross validation. In our exploratory data analysis, we had examined the plots for the distribution of the features so that we were able to determine which features become normally distributed after transformation. We also have a large training set, so we were able to do extensive plotting and cross validation on it in addition to the whole set.

We also applied LASSO logistic regression to all of our features and potential interactions on the training set. The lambda hyperparameter was selected with 10-fold cross validation. We checked the coefficients for both the optimal lambda, as well as the lambda with the minimum number of variables, but within one standard error of the optimal lambda. Then, we compared potential variables with their plots. Since PCs capture the same information as the regular features, we decided to treat the PCs and regular features as separate, and produced two sets of variables. For the regular features, we ended up with ProductRelated, PageValues, ExitRates, MonthMay, MonthNov, ReturningVisitor, and the interaction between {ProductRelated, PageValues} x {May, Nov}. For the PCs, we decided on PC1, 3, 5, 6, and the same set of categorical variables. For interaction, we have {PC1, PC6} x {Nov}. In the subsequent sections, the result on both would be reported.

After we picked our initial variables, we experimented with them on the training set with cross validation. It turns out that numerical variables with interactions outperforms numerical variables with both interaction and categorical features. Therefore, we decided to drop the categorical variables in the end.

Learning Results

As said previously, the empirical priors are highly skewed. A model only outputting False would achieve ~0.85 accuracy. Three alternative metrics will be used for assessing model accuracy: balanced accuracy (the average of the two class accuracies), Cohen's kappa (a score between -1 and 1 where 0 is normalized to be equivalent with a random model based on the empirical priors) and the F1 score (a scored between 0 and 1 based on "positive" accuracy, in this case accuracy of True predictions). To start we made 75% of the data training and 25% for testing. Cross validation was used to pick the hyperparameters with a tuning grid, and also to

probe for promising models. We found the following procedures to have the best Kappa in CV, and their results on the testing set is listed below. During testing, we are mostly interested in the F1 score in order to compare to existing literature. However, we also looked at Kappa for comparison because it is a more balanced metric that is not focused on the positive cases.

SVM Linear

	Kappa	Balanced Accuracy	F1	Accuracy
Feature	0.6228	0.8071	0.6802	0.9027
PCA	0.6261	0.8136	0.6841	0.902

SVM RBF

	Kappa	Balanced Accuracy	F1	Accuracy
Feature	0.6307	0.8009	0.6844	0.9079
PCA	0.621	0.8017	0.6775	0.9037

Quadratic discriminant analysis

	Kappa	Balanced Accuracy	F1	Accuracy
Feature	0.6232	0.8476	0.6887	0.89
PCA	0.6118	0.8342	0.678	0.8891

Logistic regression with all features, numerical and categorical, was also tried. Overall, regression underperformed compared to SVM and QDA by around ~0.1 F1 score, despite having access to categorical data. When trying the above methods without the initial logarithmic transform and standardization, the accuracy is decreased by a large amount (~0.15 decrease in F1 score).

Results and Discussion

Overall, we achieved significant improvement over the models used in the original paper. For comparison, the multilayer perceptron used by the original authors in table 4 had accuracy of ~0.87, balanced accuracy of ~0.735 and F1 of ~0.57. Our QDA model had 0.89, 0.85 and 0.69 respectively.

The original authors used both random oversampling and the Synthetic Minority Oversampling Technique (SMOTE) to oversample their dataset for training better on the minority class. This resulted in a massive (~.55 to ~.8 F1 score in most models) increase in model accuracy. We attempted to replicate their multilayer perceptron model trained on both normal and random oversampled data, but the increase in F1 score was negligible (~0.01). When we oversampled the entire dataset first, then split it into training and testing data we found a similar increase to .86 F1 score.

Conclusion

We were able to obtain high scores in most metrics with SVM and QDA models through transformation and variable selection. Further areas to optimize may include changing or parametrizing the initial logarithm transformations to remove skewness and increase normality, and changing how categorical variables are integrated with the numerical models.

References

Sakar, Okan, Oclay Polat, Mete Katircioglu, and Yomi Kastro. 2018. “Real-Time Prediction of Online Shoppers’ Purchasing Intention Using Multilayer Perceptron and LSTM Recurrent Neural Networks.” *Neural Computing and Applications* 31. https://link.springer.com/article/10.1007/s00521-018-3523-0/.

n.d. “Online Shoppers Purchasing Intention Dataset.” https://archive.ics.uci.edu/ml/datasets/Online+Shoppers+Purchasing+Intention+Dataset.