Multivariate Statiscal Analysis for Super Stores

Shu-Cheng Zheng

July 29, 2023

1 Super Stores

The goal is to predict whether customers will purchase the offer. However, the data is imbalanced, label 0 accounting for around 85% and label 1 accounting for 15%,meaning that we cannot solely rely on accuracy as it can be easily biased towards the majority class. The company's objective is to minimize the cost of the campaign while maximizing the number of customers who will actually make a purchase. Therefore, we need to consider both precision and recall scores, amd the F_1-score provides a balanced evaluation metric.

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

takes into account both precision (the ability to correctly identify positive instances) and recall (the ability to capture all positive instances). Moreover, to address the issue of imbalanced data, I also employed the under-sampling method known as random under-sampling. This technique randomly selects samples from the majority class until the number of samples in each class becomes equal within each fold. By applying random under-sampling, we create a more balanced training set, allowing the models to learn from an equal representation of both the majority and minority classes. This helps prevent the models from being biased towards the majority class and enables them to better capture patterns and make accurate predictions for both classes.

After deciding on the evaluation metric, the next step is to design an experiment pipeline to assess our candidate models. First, we split the data into a training set and a test set, using a ratio of 80% for training and 20% for testing. Furthermore, to obtain reliable and robust results, we employ nested and stratified 5-fold cross-validation on the training set. This approach helps us fine-tune the model's hyperparameters and mitigate the impact of random data splits. In the inner loop of cross-validation, we perform a random search to explore different hyperparameter configurations. On the other hand, in the outer loop, we evaluate the model's performance using the chosen hyperparameters on each fold of the cross-validation. This allows us to identify the most promising hyperparameters and to get an unbiased estimate of the model's generalization ability on unseen data.

By following this experiment pipeline, we can effectively evaluate our candidate models, select the best hyperparameters, and assess their performance using the F1 score.

1.1 Candidate Models

1.1.1 LDA

LDA, which stands for Linear Discriminant Analysis, is a statistical technique that aims to find a linear combination of features to maximize the separation between different classes in the data. This is achieved by modeling the distribution of features within each class and calculating the optimal projection that maximizes the ratio of between-class scatter to within-class scatter. LDA assumes that the data follows a Gaussian distribution and that the classes have equal covariance matrices. By computing the eigenvalues and eigenvectors of the covariance matrix, LDA determines the optimal projection direction for achieving the desired separation between classes.

1.1.2 QDA

QDA, which stands for Quadratic Discriminant Analysis, is a classification algorithm that extends the principles of LDA by relaxing the assumption of equal covariance matrices across classes. Unlike LDA, QDA allows each class to have its own covariance matrix. This approach enables QDA to capture more complex relationships in the data compared to LDA. However, it is important to note that QDA may be more prone to overfitting when the number of features is large relative to the number of observations.

1.1.3 Support Vector Machine

Support Vector Machine (SVM) finds the hyperplane that maximizes the margin between classes, with data points closest to the hyperplane called support vectors. SVM can handle linearly separable data and uses kernel functions to handle non-linearly separable data. It is known for its ability to handle high-dimensional data, its robustness against overfitting, and its effectiveness with small datasets.

1.1.4 Decision Tree

The decision tree algorithm builds the tree by recursively partitioning the data based on the values of different features. At each node, the algorithm selects the best feature that maximizes the information gain or decreases the impurity of the data. This process continues until a stopping criterion is met, such as reaching a maximum depth or having a minimum number of samples in a leaf node. However, decision trees are prone to overfitting, especially when the tree becomes too complex. This can be mitigated by techniques such as pruning, setting constraints on the tree growth, or using ensemble methods like random forests. Overall, decision trees are versatile and widely used machine learning models for both classification and regression tasks.

1.2 Random Forest

Random Forest is an ensemble learning algorithm with the bagging method that combines multiple decision trees to create a robust and accurate model. It uses random subsets of data and features to train each tree, reducing overfitting and improving generalization. During prediction, the model combines the predictions of individual trees to make a final decision. Random Forest is known for its robustness, feature importance analysis, ability to handle non-linear relationships, scalability, and resistance to overfitting.

1.3 Adaboost

Adaboost is an ensemble learning algorithm with the boosting method that combines multiple weak classifiers to create a strong classifier which works by iteratively training weak classifiers on different subsets of the data, assigning higher weights to misclassified samples in each iteration. This allows the subsequent weak classifiers to focus on the previously misclassified samples, effectively improving the overall accuracy of the model. Adaboost is particularly effective in handling complex classification problems and has been widely used in various domains.

1.4 How Difference of Models and Data Influence Result

1.4.1 Incomplete Data-Consumption Behavior

In this dataset, our candidate models are LDA, QDA, Decision Tree, and SVM. The consumption behavior data consists of numeric variables. To ensure that the scale of the data does not influence the results of SVM, it is important to normalize the data. By normalizing the data, we bring all the variables to a similar scale. Additionally, we need to assess whether the data follows a Gaussian distribution, as LDA and QDA assume this distribution. Therefore, I do the kernel density estimation (KDE) plot, which is similar to histogram, to observe the data. Moreover, I also use the Henze-Zirkler test, which measures the distance between two distribution functions. If the data is distributed as multivariate normal, the test statistic is expected to be approximately log-normally distributed. The test involves calculating the mean, variance, and smoothness parameter, log-normalizing the mean and variance, and estimating the p-value. In our case, the p-value of the Henze-Zirkler test is 0, indicating that the data does not follow a Gaussian distribution. This is further supported by the kernel density estimation plot, which shows deviations from a Gaussian shape.

To address the non-Gaussian nature of the data and the presence of outliers, we first attempted a transformation using the Yeo-Johnson method, generelized transformation of Box-Cox, which can handle zero and negative values. However, the data still exhibited significant deviations from a normal distribution, as indicated by the Henze-Zirkler test. To further address the outliers issue, we applied the Local Outlier Factor (LOF) algorithm, which measures the local density deviation of each data point relative to its neighbors. This approach is effective in identifying anomalies in datasets with irregular shapes and varying densities.

Considering these findings, we decide to use the normalized original data since it is more likely to find a normal distribution that approximates the original data after outliers detection.

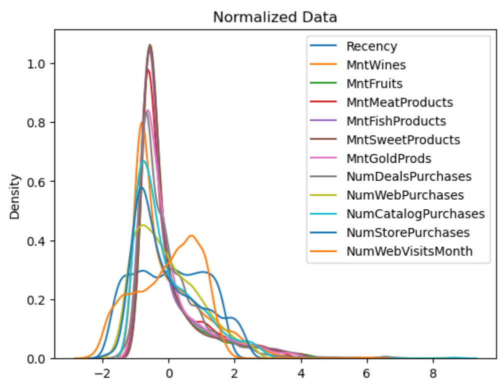

Figure 1: KDE Plot of Normalized Data

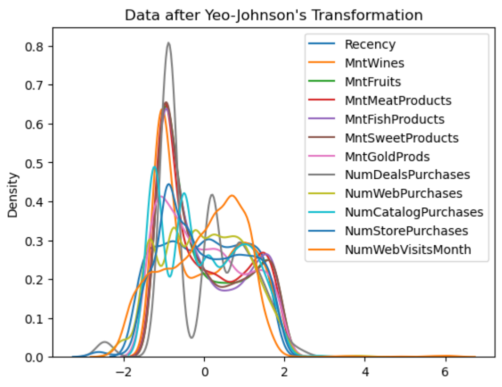

Figure 2: KDE Plot of YJ's Transformed Data

The confusion matrices of each model for the test data are as follows. We observe that QDA has the fewest false negatives and the highest false positives, indicating that it incurs a significant cost in predicting whether customers will make a purchase, although it successfully captures almost all customers who will buy the offer. Except for QDA, LDA has the fewest false positives and false negatives, followed by SVM, and Decision Tree performs the least well in terms of false positives and false negatives. Furthermore, Table 1 demonstrates that LDA outperforms the other models in terms of both F_1-score and accuracy in the repeated 5-stratified folds and F_1-score in the test data. Based on these results, I recommend using the LDA model in this situation.

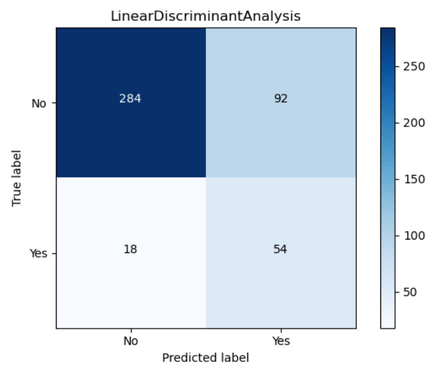

Figure 3: Confusion Matrix of LDA

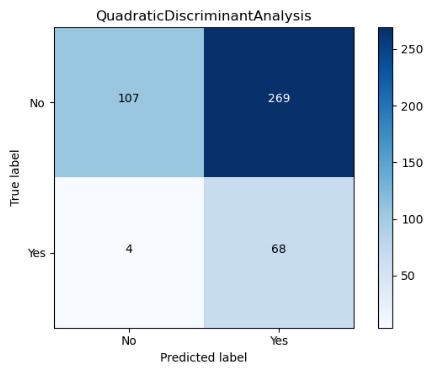

Figure 4: Confusion Matrix of QDA

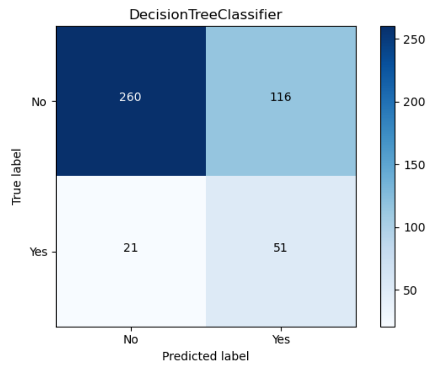

Figure 5: Confusion Matrix of Decision Tree

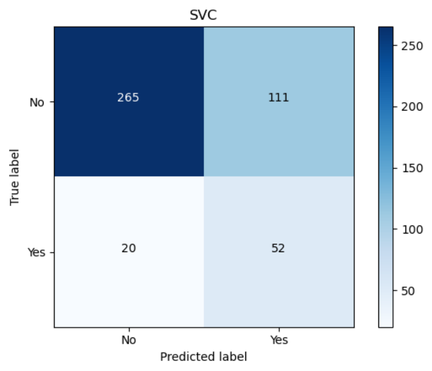

Figure 6: Confusion Matrix of SVC

	LDA	QDA	DT	SVC
F_1-Score in CV	**0.461**	0.367	0.383	0.451
Accuracy in CV	**0.755**	0.582	0.706	0.723
F_1-Score in Test	**0.484**	0.332	0.440	0.444

Table 1: The Mean Values of Metric Scores in Consumption Data

1.4.2 Complete Data

The advantage of using the Decision Tree model is its effectiveness in handling categorical data. Unlike LDA and QDA, Decision Trees and SVM do not assume a Gaussian distribution for the data. Therefore, in our case, we exclude LDA and QDA and choose the Decision Tree and SVM models. However, when working with categorical features, using one-hot encoding can increase the number of features significantly, which may affect the performance of SVM. Similarly, label encoding does not provide any inherent meaning to the encoded values, but it can still influence the results of SVM. To address these issues, I decided to perform target encoding on the categorical features. Target encoding utilizes the information from the target variable to assign suitable values to represent each class of the categorical features. This encoding method provides a meaningful representation of the categories without introducing a large number of additional features. Additionally, I normalized the columns to ensure that the data is on the same scale. Standardization is beneficial for SVM as it maximizes the margin by minimizing the norm of the kernel.

By using target encoding and standardization, we can effectively handle categorical features and ensure that the data is appropriately scaled for both the Decision Tree and SVM models.

The confusion matrix reveals a high number of false positives in the Decision Tree model, resulting in a lower F1-score. This behavior is expected as Decision Trees tend to overfit the data. Conversely, the SVC model performs better overall, despite the conventional belief that tree-based models are more suitable for categorical data. This difference in performance could be attributed to the target encoding used in the models, which captures categorical feature information instead of relying on meaningless numbers from one-hot encoding. However, both models have lower F1-scores compared to models trained solely on consumption data. This discrepancy may be due to the presence of uninformative features or collinearity issues. To address these challenges, feature selection and engineering techniques can be employed to extract more useful information from the data.

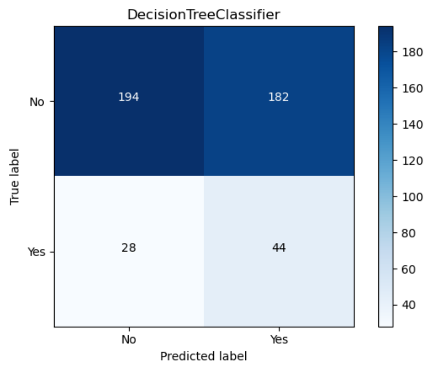

Figure 7: Confusion Matrix of Decision Tree

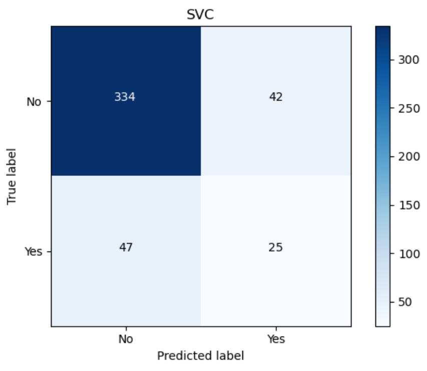

Figure 8: Confusion Matrix of SVC

	DT	SVC
F_1-Score in CV	0.235	**0.261**
Accuracy in CV	0.727	**0.856**
F_1-Score in Test	0.295	**0.360**

Table 2: The Mean Values of Metric Scores in Consumption Data in Whole Data

1.4.3 Ensemble Model

Ensemble models combine the predictions of multiple individual models to improve overall performance and accuracy. They can be categorized into bagging, boosting, and stacking methods. Bagging trains multiple instances of the same model on different subsets of data, while boosting focuses on correcting errors made by previous models. Stacking combines predictions of different models using a meta-model. Ensemble models are effective in reducing overfitting and improving robustness.

Both Adaboost and Random Forest outperform the individual learner, both in terms of cross-validation and test data. Adaboost, as a boosting method, continuously improves the results by iteratively creating weak classifiers to handle the remaining samples that are difficult to classify correctly. On the other hand, Random Forest, as a bagging method, generates multiple decision trees and combines their classifications, mitigating the risk of overfitting and achieving better generalization and robustness.

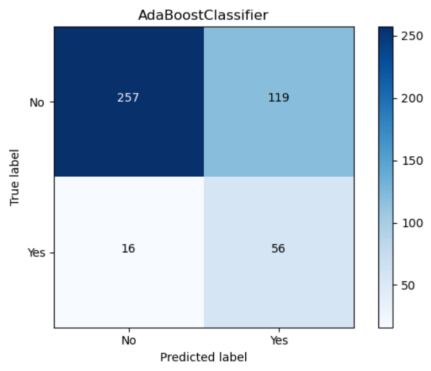

Figure 9: Confusion Matrix of Adaboost

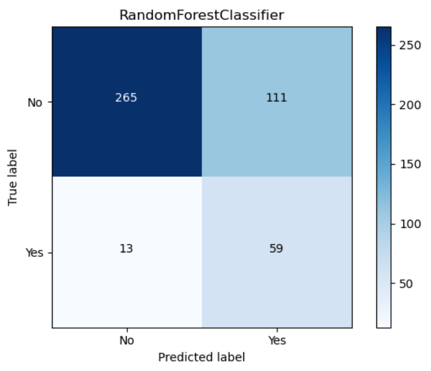

Figure 10: Confusion Matrix of Random Forest

	RF	ADB
F_1-Score in CV	0.441	**0.450**
Accuracy in CV	0.730	**0.727**
F_1-Score in Test	0.453	**0.488**

Table 3: The Mean Values of Metric Scores in Consumption Data

1.5 Conclusion

In summary, Adaboost and Random Forest outperform individual learners in both cross-validation and test data. Adaboost iteratively improves by creating weak classifiers, while Random Forest reduces overfitting through ensemble decision trees. However, lower performance compared to models trained solely on consumption data suggests the presence of uninformative features or collinearity. Additional steps, such as feature selection and engineering, may be needed to help extract more relevant information from the data.

	LDA-C	QDA-C	DT-C	SVC-C	DT-W	SVC-W	RF-C	ADB-C
F_1-Score in CV	**0.461**	0.367	0.383	0.451	0.235	0.261	0.441	0.450
Accuracy in CV	0.755	0.582	0.706	0.723	0.727	**0.856**	0.730	0.727
F_1-Score in Test	0.484	0.332	0.440	0.444	0.295	0.360	0.453	**0.488**

The model is fitted by only consumption data
The model is fitted by whole data

Table 4: The Mean Values of Metric Scores