View on GitHub

CustomerPurchaseBehavior

Customer Purchase Behavior Group 13

0. Midterm Checkpoint Preview

After receiving feedback on our proposal, our group has now changed our project. Consequently, we have redone our proposal (and ignored literature review as we deemed it wouldn't be beneficial to completely redo that section). Additionally, our group now has two people after the majority of the group has dropped the course. We have tried our best to make the most of the project but there may be some sections that seem a bit "lacking".

1. Introductory and Background Information

Data Set Link: https://www.kaggle.com/datasets/rabieelkharoua/predict-customer-purchase-behavior-dataset
Data Set Description: The dataset has a handful of variables including age, gender, income, and purchase history, and a value of either 1 or 0 representing whether a customer made a purchase.

2. Problem and Motivation

Motivation: Customer purchase behavior is useful for business in order to improve their marketing strategies and which audience they should appeal to. More importantly, by understanding how certain variables/demographic factors affect whether customers purchase or not, we can optimize business strategies. This type of information helps companies increase their ROI and understand their consumers better.

Problem: In this project, we aim to use a variety of different approaches to build models that can forecast customer purchase behavior.

3. Methods

3.1 Data Preprocessing Method: We carried out at least one of the following preprocessing steps for every model:

1. **Categorical Data:** We transformed our categorical data using the OneHotEncoder function (from sklearn library) to convert our categorical data into numerical. There were three categorical variables we had to consider, gender, product category, and loyalty program. Since we decided to one hot encode these variables, it created more input variables for our model. This is because the gender had two options, the product category had five, and the loyalty had two options. Thus, after transforming our categorical data, there ended up being six additional variables.

2. **Scaling Numerical Features:** We decided to use the StandardScaler to standardize our numerical features.

3. **Data Splitting:** We split the dataset into training and testing sets (usually an 80-20 ish ratio).

4. **Building the Preprocessing Pipeline:** We created a preprocessing pipeline that includes both the scaling of numerical features and the encoding of categorical features. This pipeline ensures that all preprocessing steps are applied consistently to both the training and testing datasets.

For future models, we may implement different approaches to help with the data preprocessing.

3.2 Models: The main approaches our group are going to try are: logistic regression, gradient boosting machine, and random forest.
3.2.1 Model 1 (Logistic Regression): The first model we did was a linear regression. We chose this model since we are doing a binary classification problem and are given variables we can assign weights to as coefficients. We trained our model on all the 14 inputs and processed the categorical data using one-hot encoding and standardized the numerical features. A visual representation of our results can be seen in figure 4.1.1 and quantitative metrics can be seen in figure 4.1.2. After preprocessing our data as explained in 3.1, we then used a pipeline and got a final model. We then played around with the threshold value and found the default value of 0.5 wasn't necessarily the best. We found that 0.4 had slightly better results.
3.2.2 Model 2 (Gradient Boosting Machine): The second approach we took on was using a gradient boosting machine. We decided to preprocess the data in a similar way to the linear regression model. This included one hot encoding all the categorical data and using a pipeline. Then, we used the GradientBoostingClassifier from the sklearn library and used built in functions to minimize our loss function. The results were good, as can be seen in figure 4.2.1 but we expected slightly higher results.
3.2.3 Model 3 (Random Forest): Our best model yet was the random forest model. There wasn't much preprocessing needed for this model, just splitting the data into test and training data sets. Then we initialized a random forest classifier and fitted the model. There was overfitting encountered with the training accuracy which we had to adjust the random forest classifier to correct. The results for this model can be seen in figure 4.3.1.

4. Result and Discussion

4.1 Logistic Regression Results

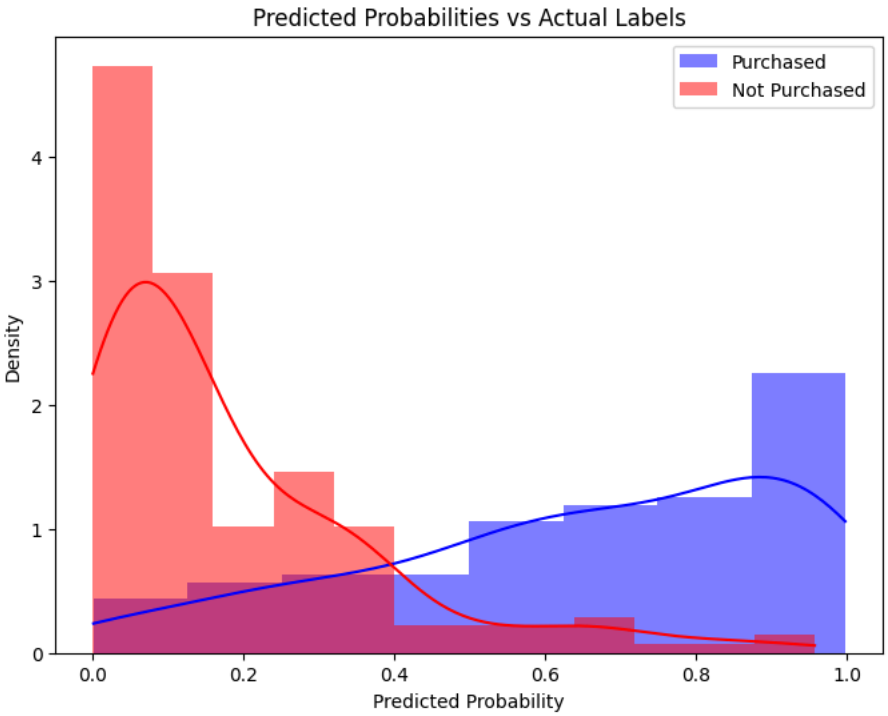


Figure 4.1.1: Logistic Regression Visualization of Predicted Probabilities
Figure 1 provides a visual representation of the predicted probabilities vs actual labels generated by our regression model. More precisely, the x-axis shows the predicted probabilities, the y-axis represents the density, the blue line represents the density of predicted probabilities for the "purchased" class and the red line represents the density of predicted probabilities for the "no purchase" class. Note, we chose our cutoff value where the lines intersect. This is important since when we are below 0.4, we want our model to predict "no purchase" and when we are above 0.4 we want to predict a "purchase." As can be expected, when the model becomes more inaccurate when a predicted customer receives a value close to 0.4 but is practically always right when the value is close to 0 or 1.

```
Accuracy: 0.8533333333333334
Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.90      0.88       172
           1       0.86      0.79      0.82       128

    accuracy                           0.85       300
   macro avg       0.85      0.85      0.85       300
weighted avg       0.85      0.85      0.85       300
```

Figure 4.1.2: Logistic Regression Quantitative Metrics Results
The logistic regression model achieved an overall accuracy of 85%, with a precision of 0.85, and recall of 0.9 for the "no purchase" group, and a precision of 0.86 and recall of 0.79 for the "purchased" group.

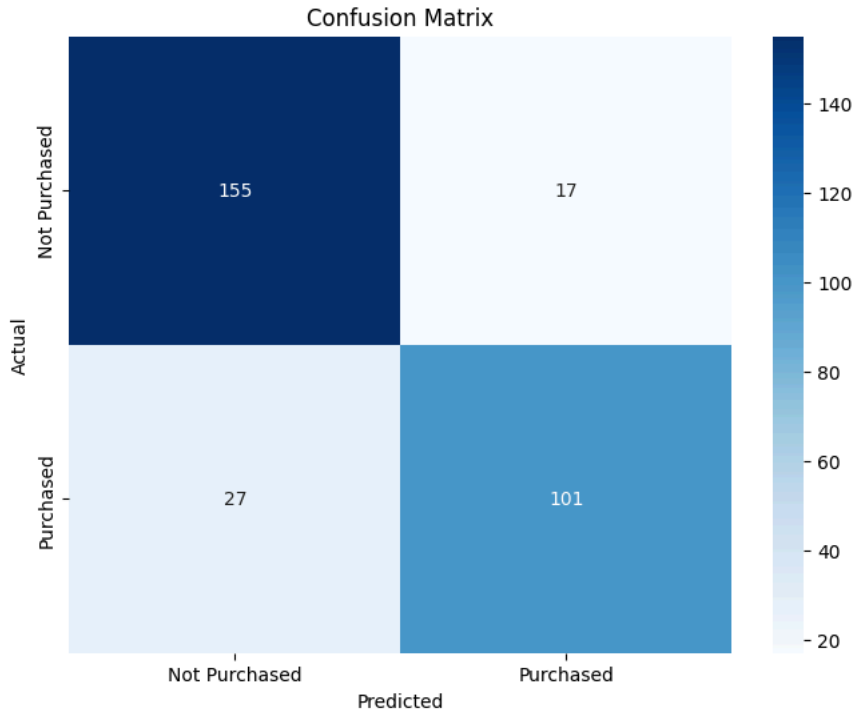


Figure 4.1.3: Logistic Regression Confusion Matrix
The following visual shows the confusion matrix generated from our logistic regression. It indicates our model had 17 false positive predictions and 27 false negatives predictions. Calculating the recall and precision metrics, we can see that they match with the results from 4.1.2.

4.1.4 Analysis of logistic regression model: The high recall of 0.92 for class 0 (Not Purchased) indicates that the model is very effective at identifying customers who will not make a purchase, which helps in minimizing wasted marketing efforts. Conversely, the precision of 0.88 for class 1 (Purchased) shows that when the model predicts a purchase, it is often correct, which is crucial for effective targeted marketing. Depending on the company, there may be instances when we would want to increase or decrease the threshold value from 0.4. For example, if a company wants to target consumers that have high probability of purchasing, they would want to choose a logistic regression with a higher cutoff value. Meanwhile if a company wants to target a bigger audience, they would want to lower their threshold value. Depending on the companies goals, modifying the threshold can provide them information that's more suitable to their goals.

4.2 Gradient Boosting Machine Results

```
Accuracy: 0.8222222222222222
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.89      0.85       282
           1       0.84      0.73      0.78       213

    accuracy                           0.82       495
   macro avg       0.83      0.81      0.82       495
weighted avg       0.82      0.82      0.82       495
```

Figure 4.2.1: Gradient Boosting Machine Quantitative Metrics
The following figure depicts our results for our Gradient Boosting Machine. It seemed this model had the lowest results with an accuracy of 0.82.

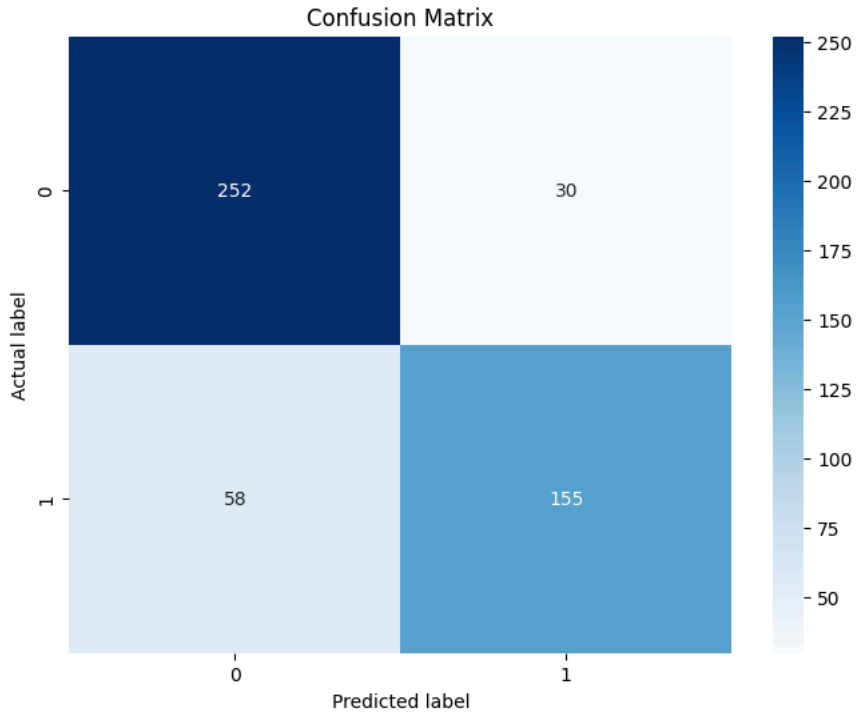


Figure 4.2.2 Gradient Boosting Machine Confusion Matrix
The following visual shows the results of our confusion matrix for our model. We can see that we had 58 false negatives and 30 false positives.

4.2.2 Analysis of Gradient Boosting Machine: The gradient boosting machine had results that were slightly below that of the logistic regression. Perhaps if we had more time, we could find a way to slightly better optimize this model and see if it would surpass the logistic regression. This would include playing around with the parameters like learning rate, number of trees, depth of trees, etc…

4.3 Random Forest Results

```
Accuracy: 0.9355555555555556
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.98      0.95       257
           1       0.97      0.88      0.92       193

    accuracy                           0.94       450
   macro avg       0.94      0.93      0.93       450
weighted avg       0.94      0.94      0.94       450
```

Figure 4.3.1 Random Forest Quantitative Metrics
As can be seen in this image, our random forest had the most successful results with an accuracy greater than 90%!

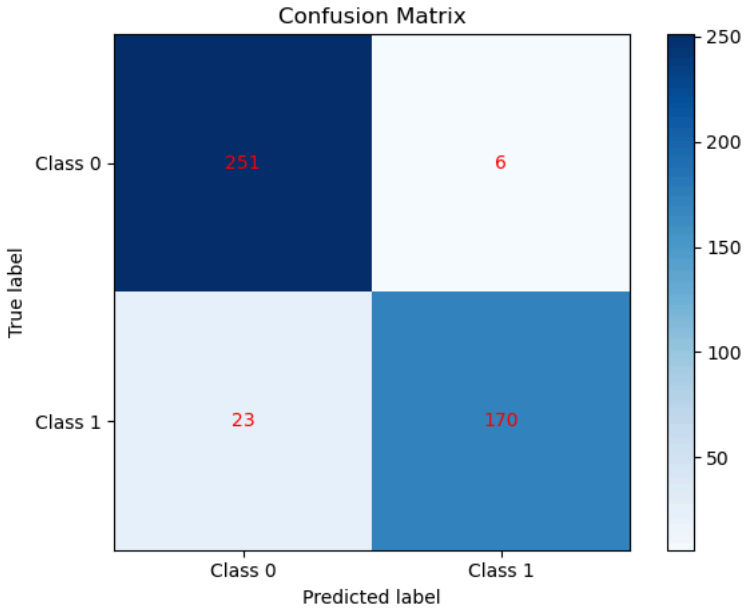


Figure 4.2.1 Random Forest Confusion Matrix
The confusion matrix for our random forest reinforces just how accurate this model turned out to be. It had a total of 23 false negatives and 6 false positives.

4.2.3 Analysis of Random Forest The random forest had the best results and managed to get an accuracy above 90%. Similar to our last model, there may be potential to get even more out of this model using hyperparameter tuning. However, we don't believe there is that much improvement left since our accuracy is already above 90% so we must be very close to an optimal model.

4.4 Precision vs Recall Let's consider the trade off between precision and accuracy. We want higher precision when the cost of a false positive is high, in this case it means targeting a non purchasing customer. On the other hand, we want to prioritize recall when the cost of a false negative is high, in this case it means when missing a potential purchaser is high. Depending on the company, we may want to use a model that balances the trade off between the precision an accuracy. For example, if a company wants to make sure they reach out to enough customers, they would prioritize recall to ensure that fewer potential customers are missed. Additionally, if a company wants to get a lower bound on the number of potential purchasers they would get, they would want to prioritize precision.

4.5 Comparison of Models Now lets compare how our models performed taking into consideration accuracy, precision, recall, and F1 score. For many projects, it can be tricky to compare models because they have very similar performance and it comes down to tradeoffs. However in our project, we found out there was a clear winner which was the random forest. The linear regression model seemed to be overall slightly better than the gradient boosting machine but not by much. More formally we have the following:

- Accuracy: Random Forest > Logistic Regression > Gradient Boosting Machine
- Precision: Random Forest > Logistic Regresson > Gradient Boosting Machine
- Recall: Random Forest > Logistic Regression > Gradient Boosting Machine

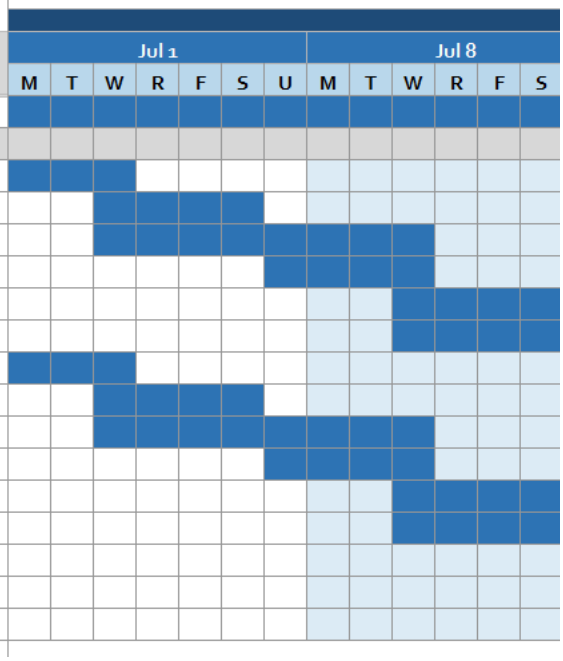
4.6 Next steps Some possible next steps if we were to continue this project would be to see how much we can milk out of our random forest model. We would also want to try gradient boosting a random forest and seeing if we can further optimize our gradient boosting machine. Overall, we are very happy with our results since we were able to get a model with above 90% accuracy and the other two models had good performance. However, we know which areas we could make possible improvements for the future.

Gantt Chart

GANTT CHART

PROJECT TITLE	Customer Purchase Behavior Group

TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION
Project Team Composition	All	1/17/2024	2/2/2024	15
Final Report				
Model 2 (M2) Design & Selection	Ojas	7/1/2024	7/3/2024	2
M2 Data Cleaning	Ojas	7/3/2024	7/6/2024	3
M2 Data Visualization	Ojas	7/3/2024	7/10/2024	7
M2 Feature Reduction	Ojas	7/7/2024	7/10/2024	3
M2 Coding & Implementation	Ojas	7/10/2024	7/16/2024	6
M2 Results Evaluation	Ojas	7/10/2024	7/16/2024	6
Model 3 (M3) Design & Selection	Joseph	7/1/2024	7/3/2024	2
M3 Data Cleaning	Joseph	7/3/2024	7/6/2024	3
M3 Data Visualization	Joseph	7/3/2024	7/10/2024	7
M3 Feature Reduction	Joseph	7/7/2024	7/10/2024	3
M3 Implementation & Coding	Joseph	7/10/2024	7/16/2024	6
M3 Results Evaluation	Joseph	7/10/2024	7/16/2024	6
M1-M3 Comparison	All	7/15/2024	7/21/2024	6
Video Creation & Recording	All	7/15/2024	7/21/2024	6
Final Report	All	7/15/2024	7/21/2024	6



Video

Video Link

Contributions

Name	Proposal Contributions
Ojas	Linear Regression Model, Gradient Boosting Machine, Github Page Writing
Joseph	Data cleaning, Data preprocessing, Random Forest, Video

CustomerPurchaseBehavior maintained by okalra3

Published with GitHub Pages