Final Project

Madeline Witters

December 4, 2022

Project Title: Predicting Customer Churn and Identifying Attributes of At-Risk Customers

Team Member Names: Madeline Witters

Problem Statement

One of the most important metrics of success in any business is customer retention. A business's customer churn rate can have very significant financial impacts that affect the company in a multitude of ways. A Harvard Business Review article recently stated that merely increasing a company's customer retention rates by 5% can increase profits upwards of 25%; conversely, the cost of obtaining a new customer can be anywhere from 5 to 25 times as expensive as retaining an existing one.

It therefore follows that if a company is able to predict which customers are at risk of leaving, they can use this information to better position themselves in a variety of ways, such as: creating an intervention plan for customers at risk of leaving, calculating potential loss of revenue in the next quarter, or simply better understanding their customer demographic and various market segments.

In this project, there are two central research questions that I will aim to answer:

1. Can I create a model that will predict customer churn with a reasonable accuracy rate?
    - Furthermore, does one model type outperform another in predicting customer churn?
2. What features/customer attributes (present in the dataset) are most important in predicting whether a customer will churn?
    - Additionally, what does interpretation of these attributes reveal (for example, are younger customers more at risk of churning)?

Data Source

For this project I am using the "Bank Customer Churn" dataset, sourced from Kaggle. The dataset has 10,000 data points and 12 variables. Each row represents an individual customer. Variables are both categorical and quantitative, and include demographic variables (age, gender, country), as well as industry-specific variables (balance, credit_card, etc.). The response variable "churn" is categorical, containing a 1 if the client left the bank and a 0 if they remained a customer.

Methodology

My methodology for this project consisted of three distinct parts:

- Exploratory Data Analysis
- Variable Selection

- Modeling

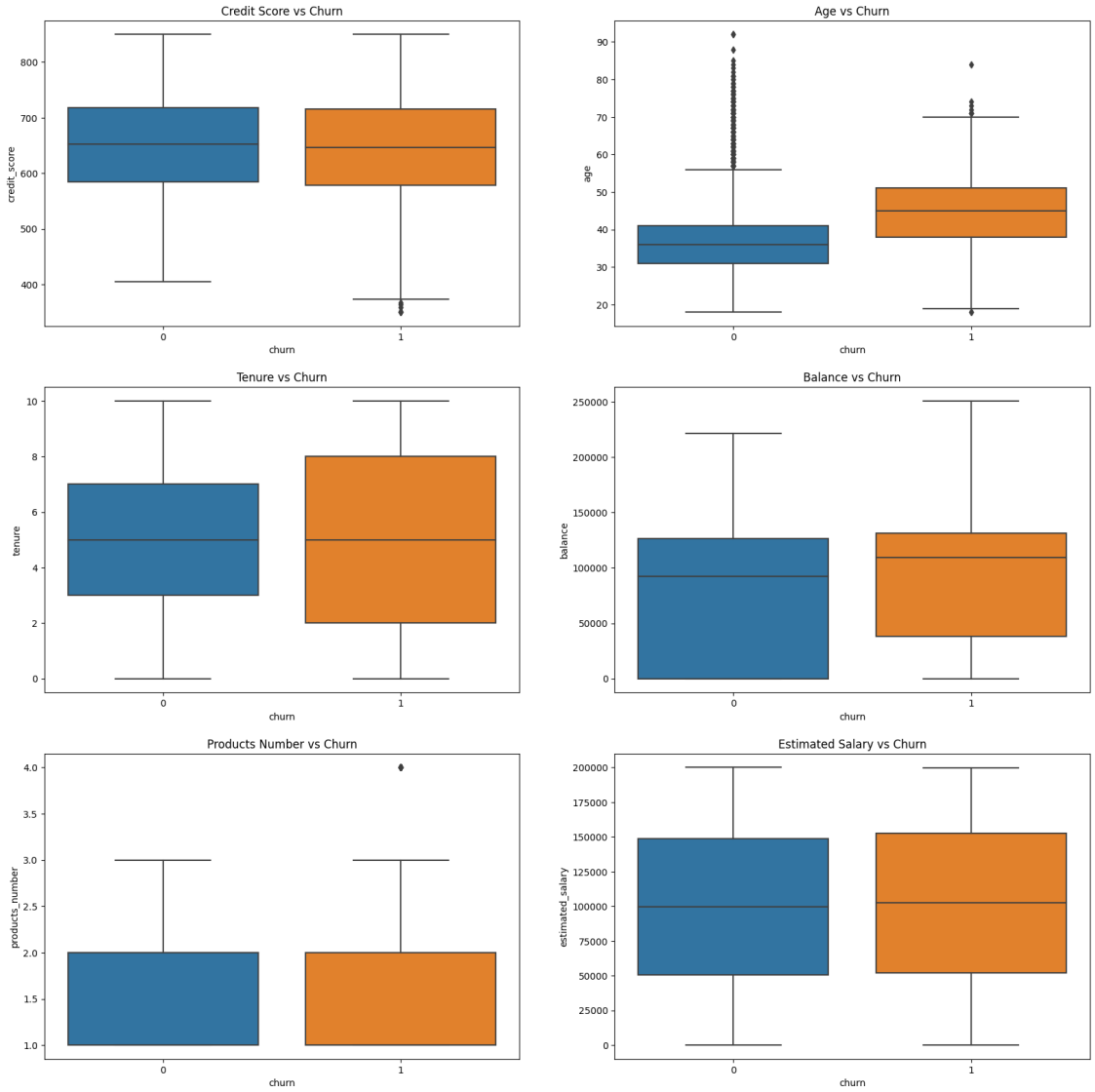
I will begin with the exploratory data analysis.

Exploratory Data Analysis

I began by performing some basic data cleaning. This primarily consisted of checking for any missing data, and dropping irrelevant features present in the dataset. There was no missing data in the dataset, so no imputation or row removal was necessary. There was only one irrelevant feature in the dataset, customer_id, a unique identification number for each customer. I used customer_id to confirm there were no duplicates present in the dataset, and then dropped it.

Next, I explored the remaining predictor variables via visual analysis. I began by creating boxplots for all the numeric variables. See below:

Boxplots of Numeric Dependent Variables

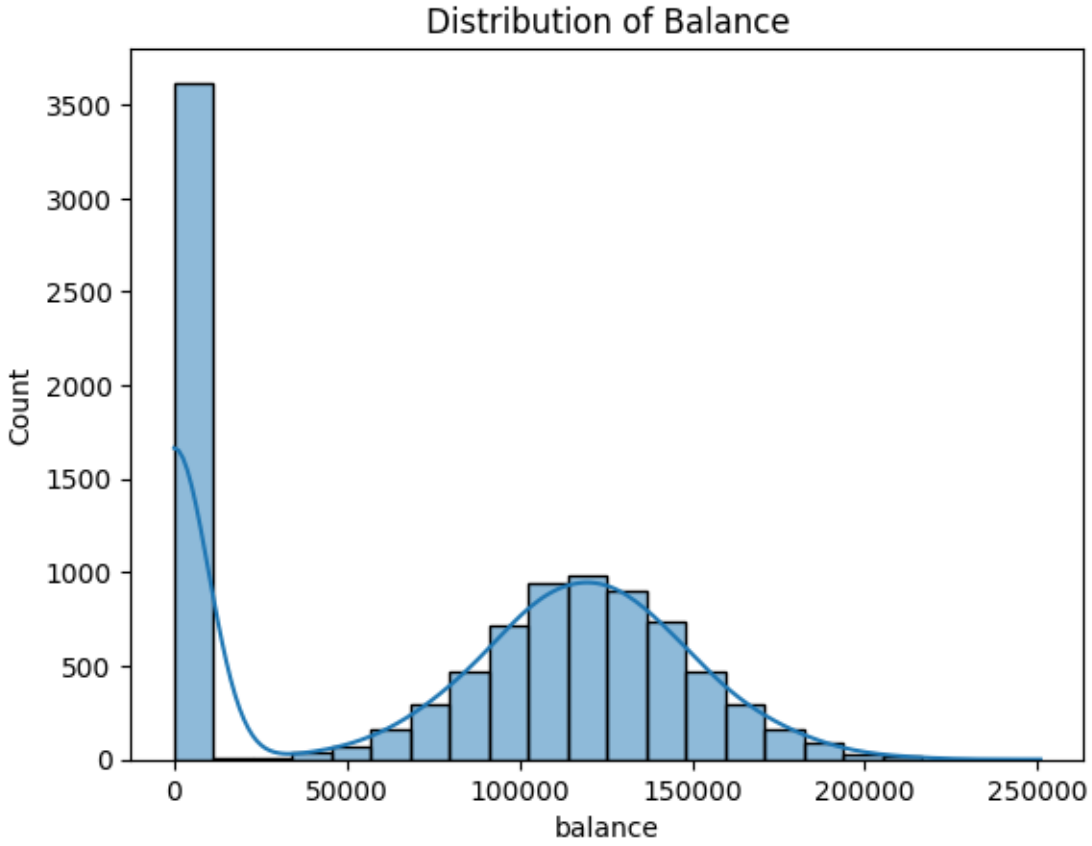


The boxplots revealed a couple interesting findings. First, it appears that the variables Credit Score, Products Number, and Estimated Salary all have very little impact on whether or not a customer churns. The boxplots for all of these variables were nearly identical when comparing the customer churn group (1) to the non-churn group (0). There was no visible distinction between the medians, and the inner-quartile range was nearly identical. Based off these boxplots alone, it would seem that these three variables will have little role in predicting churn.

Two variables that do appear to potentially have an impact, however, are Age and Balance. With the variable Age, it appears that older customers are more likely to churn, with the median for the churn group much higher than the non-churn group. With the variable Balance, it appears that those with a higher account balance are more likely to churn than those with a lower balance.

There are a couple additional items of note present in these boxplots. First, when examining the Balance boxplot for the non-churn group, it appears that there is a very large group of customers who have an account balance of zero. This may be a sign of a skewed distribution of the Balance variable. Second, several of the boxplots appear to show potential outliers. This is particularly notable in the Age boxplots. I decided to further investigate these anomalies.

I began by examining the distribution of the Balance variable via a histogram/KDE plot. See below:

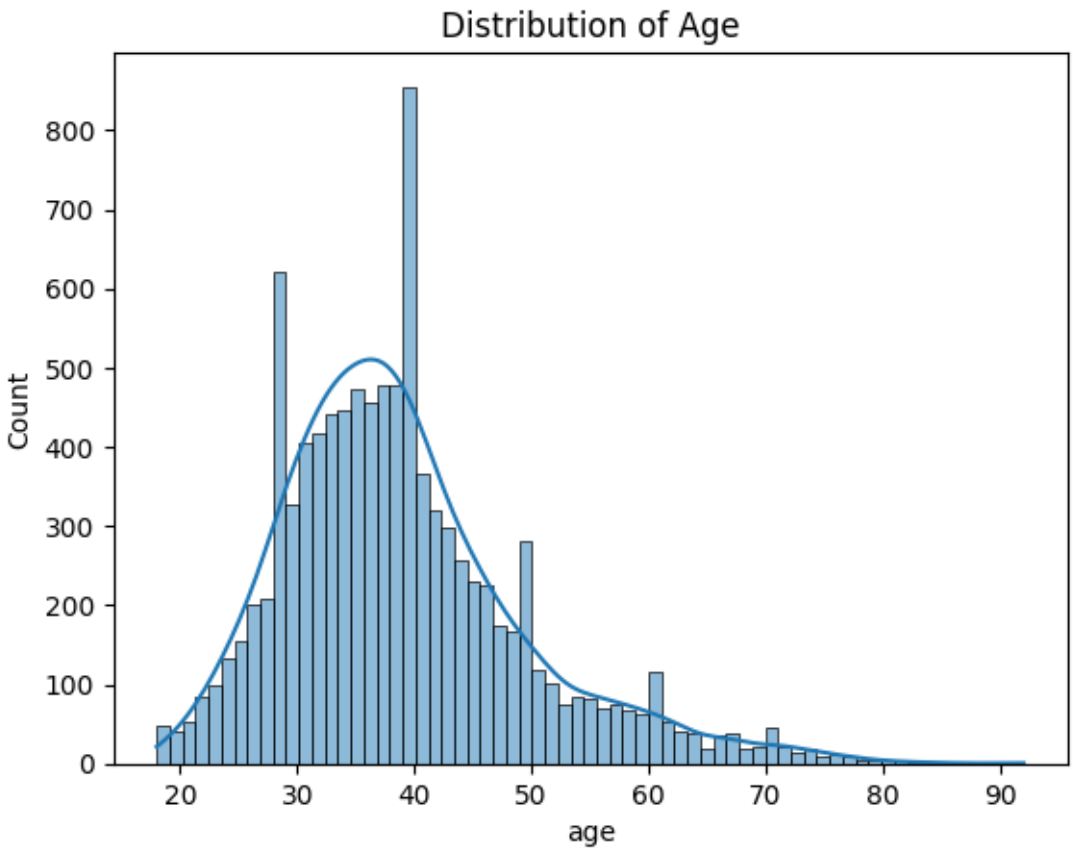


The histogram for Balance indeed confirmed a skewed distribution, with over 3500 customers having a balance of 0 in their account. While this seems odd, it could be the case that anyone who opens a new account with the bank automatically enters the system with a balance of 0 before their money is transferred over (which could also explain the shape of the non-churn balance boxplot above– those who just opened an account would be less likely to churn). Still, due to the distribution of the Balance variable, I decided to transform it to a categorical variable called "zero_balance", with a 1 indicating the customer had a balance of $0, and a 0 indicating otherwise.

Next, I used the z-score test to check for outliers present in the numeric dependent variables. Using a threshold of 3 standard deviations away from the mean, the z-score test identified approximately 133 outliers present in the Age variable, and 60 in the Products Number variable. No outliers were identified in any of the remaining variables. Having identified the outliers, I further investigated them in order to determine whether they should be removed from the dataset.

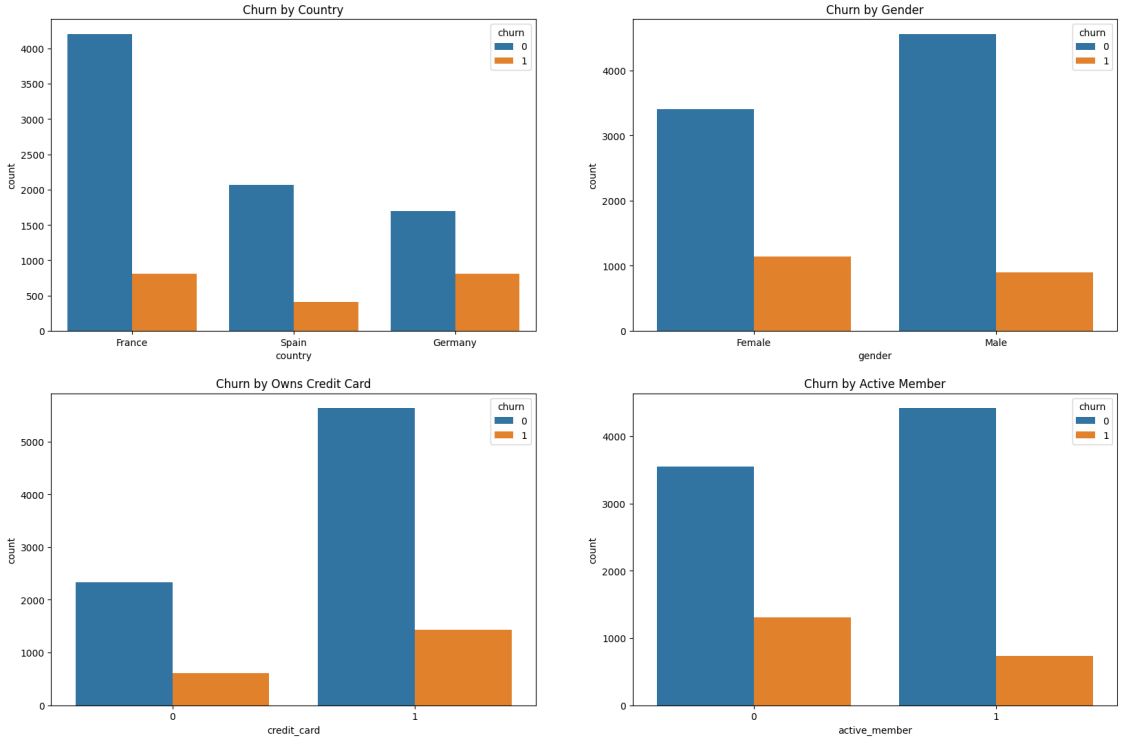
Most notably, none of the outliers appeared to represent incorrect data or data entry errors. For example, in the Age category, if one of the outliers had been an individual with the age 300, that would be a clear result of an error and could be thrown out. However, the outliers in Age primarily consisted of individuals in their 80s and 90s, who, while they may considered outliers, nonetheless represent real customers. A histogram/KDE plot of the Age variable may be found below illustrating this distribution. Likewise, the outliers in the Products Number variable also did not appear to indicate any obvious errors. Instead, it was simply the case that most customers had 1-3 products, with a small minority having 4 products.

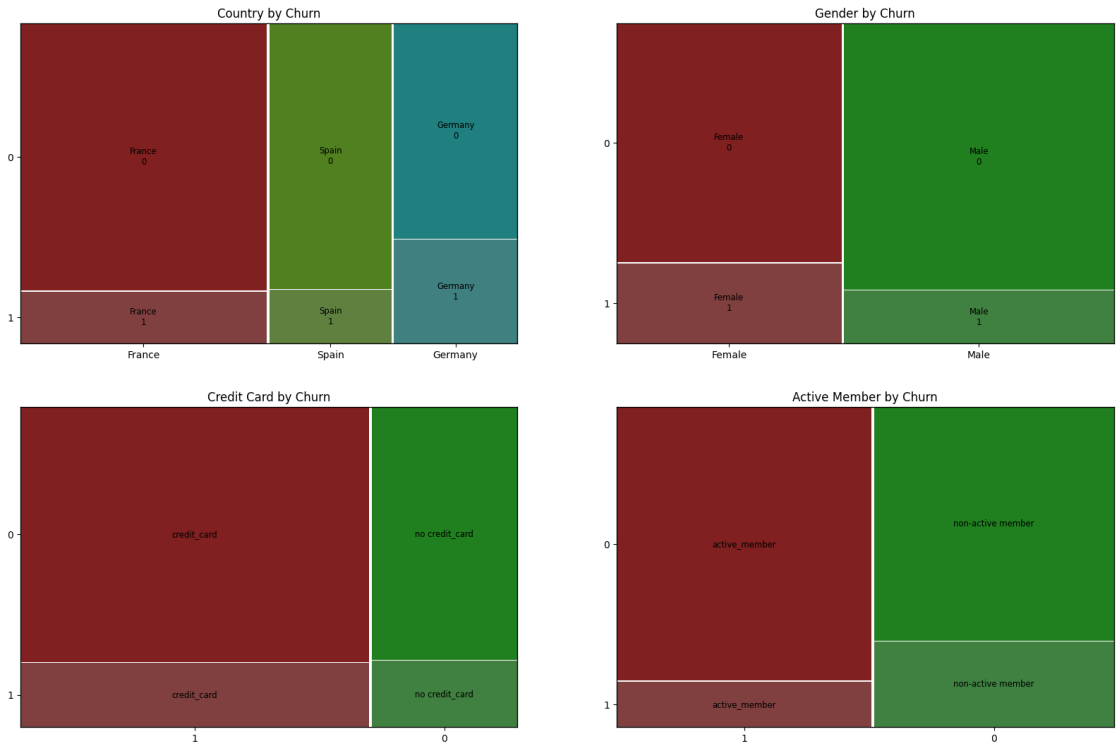
Ultimately, I decided not to remove any of the outliers from the dataset due to the fact that all of the outliers contained real customer information and did not appear to be the result of errors. Still, it should be noted that these outliers could still impact model fit. I address this issue further in the Conclusion section when discussing ways to improve and extend this project.



Having thoroughly explored the numeric dependent variables, I next turned to the categorical variables. I began by creating bar charts and mosaic plots for a basic visual analysis. See below:

Bar Charts for Categorical Dependent Variables

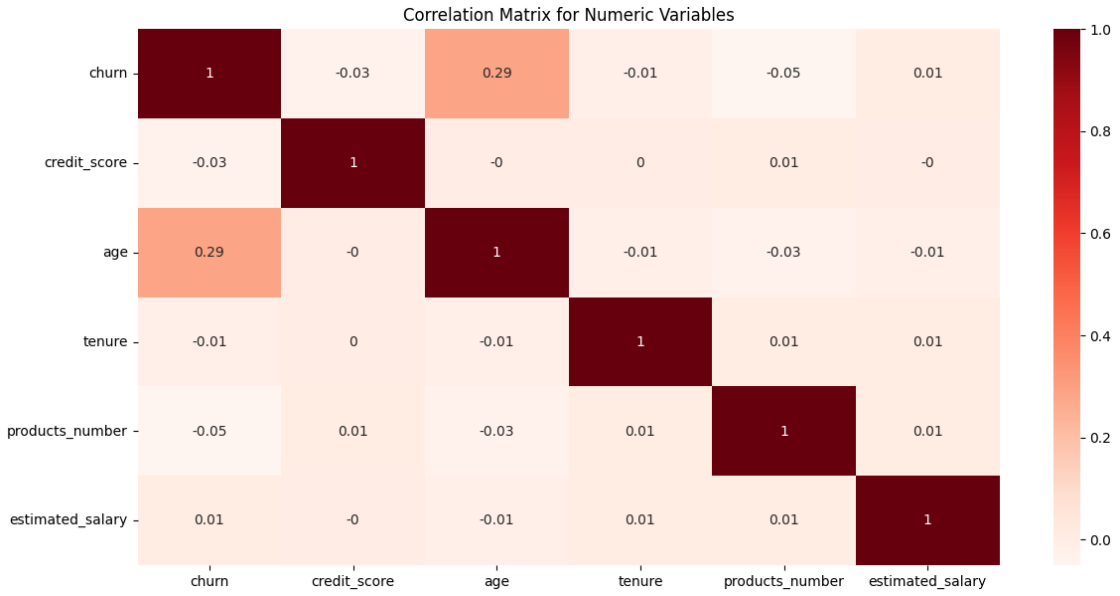




These plots contained a few interesting findings, and indicate that the variables Country, Gender, and Active Member may all play a role in predicting whether or not a customer will churn. It appears that those from Germany are more likely to churn than those from France or Spain; Females appear more likely to churn than males; and Non-Active Members are more likely to churn than Active Members. Whether or not an individual owns a Credit Card does not appear to have a large relationship with Churn.

Another interesting finding revealed by this plots is the fact that the dataset has a much higher proportion of data for customers who did not churn (0) than those who did churn (1). The official value counts for Churn are 7,963 customers who did not churn (0) and 2,037 who did churn (1). This unbalanced split has some implications for modeling which I will discuss later on.

The last part of exploratory data analysis I conducted was creating a correlation matrix for the numeric variables. High correlation between predicting variables may indicate that multicollinearity is present in the dataset, which has implications for variable selection and model building. However, the correlation matrix revealed very low correlations between numeric variables, indicating that multicollinearity will likely not be a large issue for this dataset. See below:



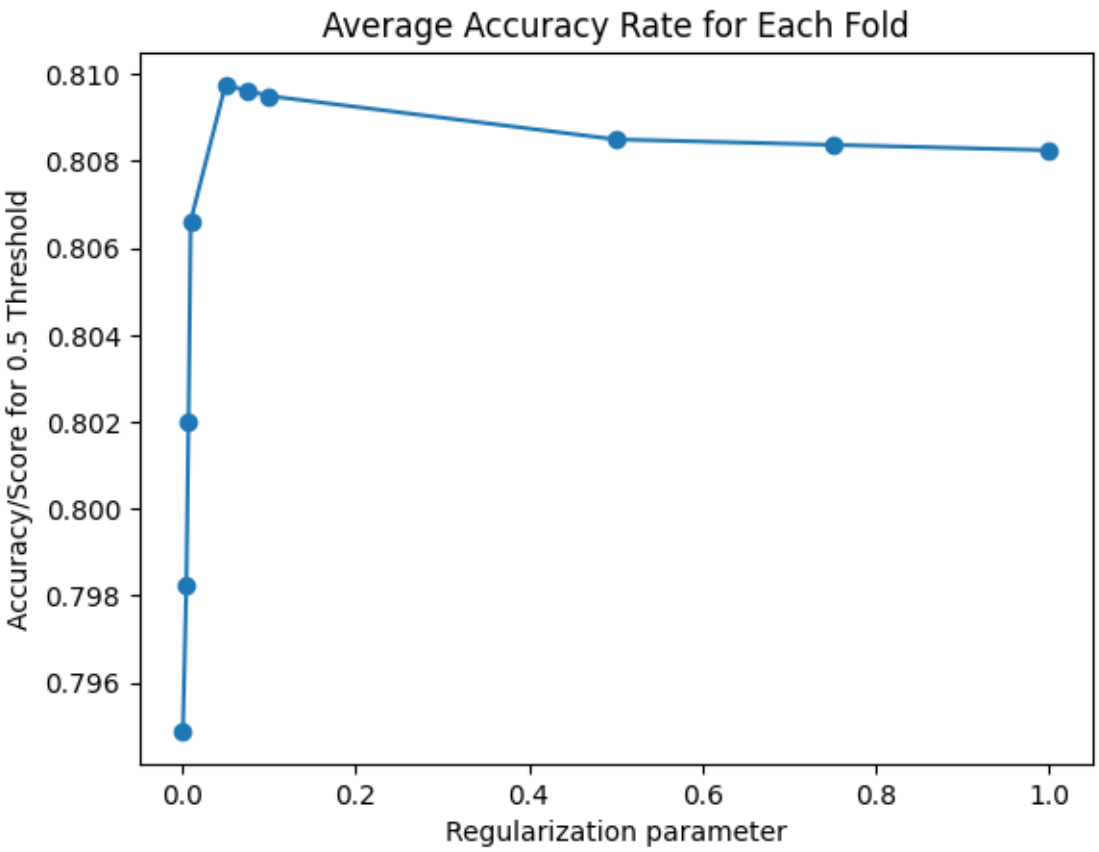
Having completed the exploratory data analysis, I conducted some one-hot encoding for the relevant categorical variables, and split the data into an 80/20 train-test split.

Variable Selection

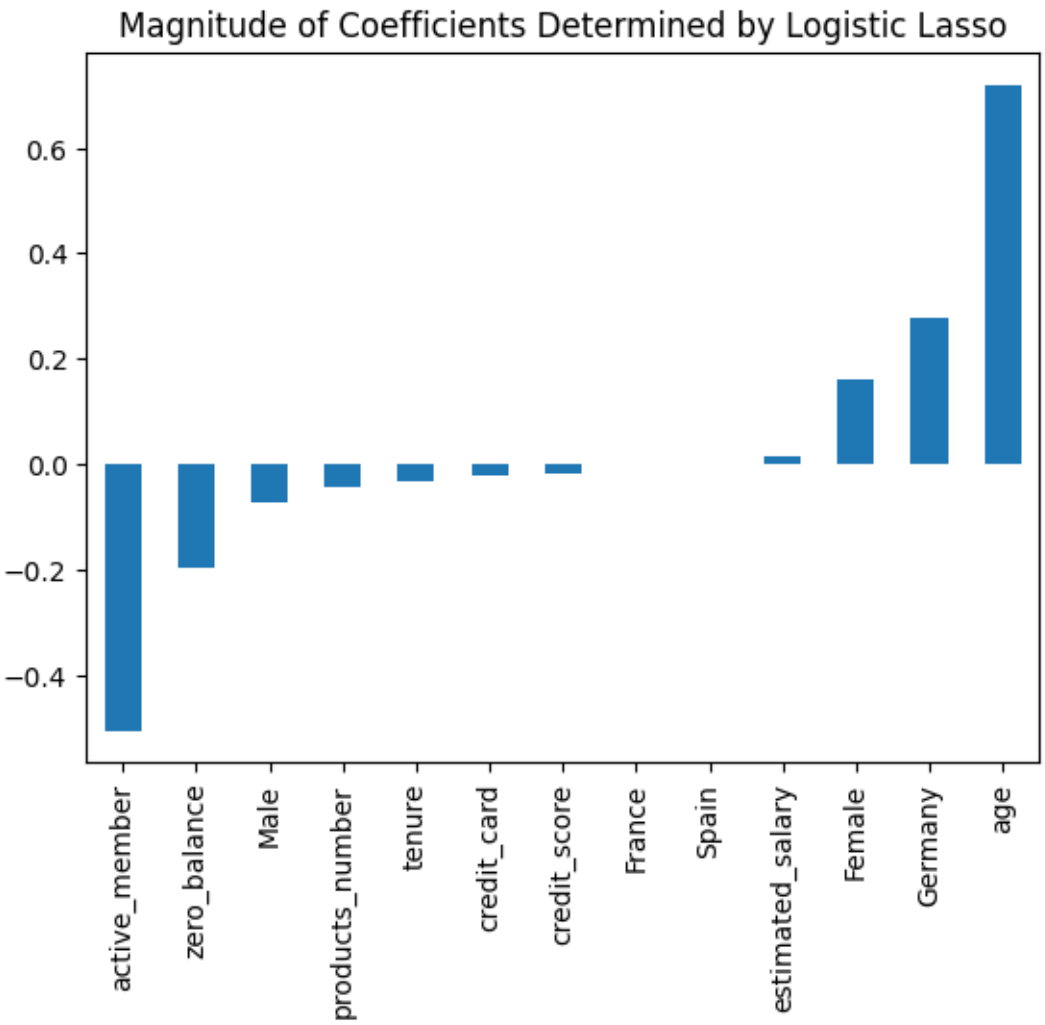
I next turned to variable selection. While the correlation matrix revealed that multicollinearity may not be an issue for this dataset, variable selection can still aid by both removing irrelevant features (those with no predictive or explanatory power), and further by identifying the most important features.

I decided to use Lasso for variable selection. Traditional Lasso optimizes the least squares problem with a L1 penalty. However, my response variable is binary; thus I must perform Logistic Lasso regression to perform variable selection (essentially, this means fitting a traditional Logistic Regression model but adding in an L1 penalty). The data was standardized and scaled prior to building the Logistic Lasso regression. I tuned the penalization parameter for the Logistic Lasso model via 10-fold cross-validation. Ultimately, the best regularization parameter was found to be 0.05. It should be noted that in sklearn's LogisticRegressionCV, the regularization parameter describes the inverse of regularization strength; thus smaller values specificy stronger regularization.

Below is a plot showing each tested regularization parameter against the average accuracy score of that particular model (using a default classification threshold of 0.5 – more on this later).



Below is a plot displaying the magnitude of the coefficients selected for the best regularization parameter 0.05. We can see that Logistic Lasso only removed two variables - the one hot encoded variables for France and Spain. Additionally, the coefficients with the largest magnitude correspond to the variables for Age, Germany, Female, Active Member, and Zero Balance. The signs of these coefficients indicate that invidiuals who are older, live in Germany, or are Female are more likely to Churn, while those who are Active Members, have a Zero Balance account and Male, are less likely to Churn.



With this information in hand, I dropped the variables France, Spain, and Male from the dataset (while the variable Male was not removed by Lasso, the information captured by that variable was redundant; hence it was only necessary to keep in either Female or Male for further modeling).

Modeling

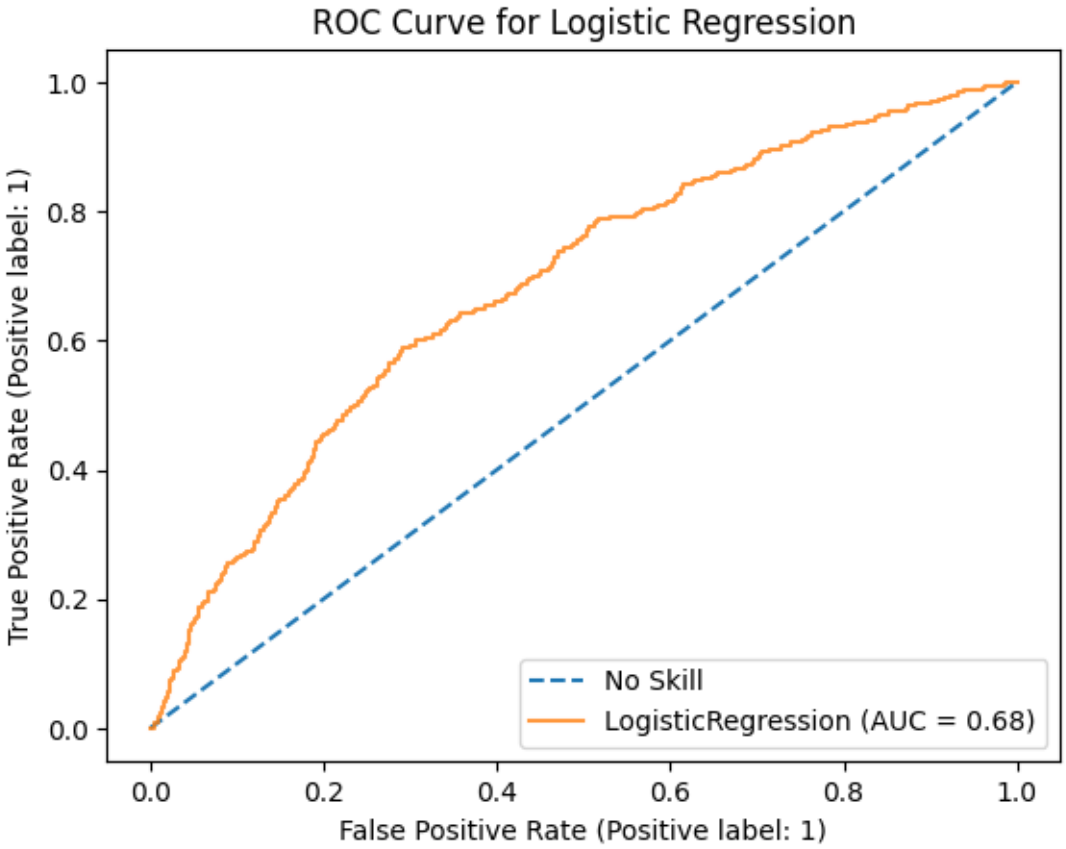
Logistic Regression

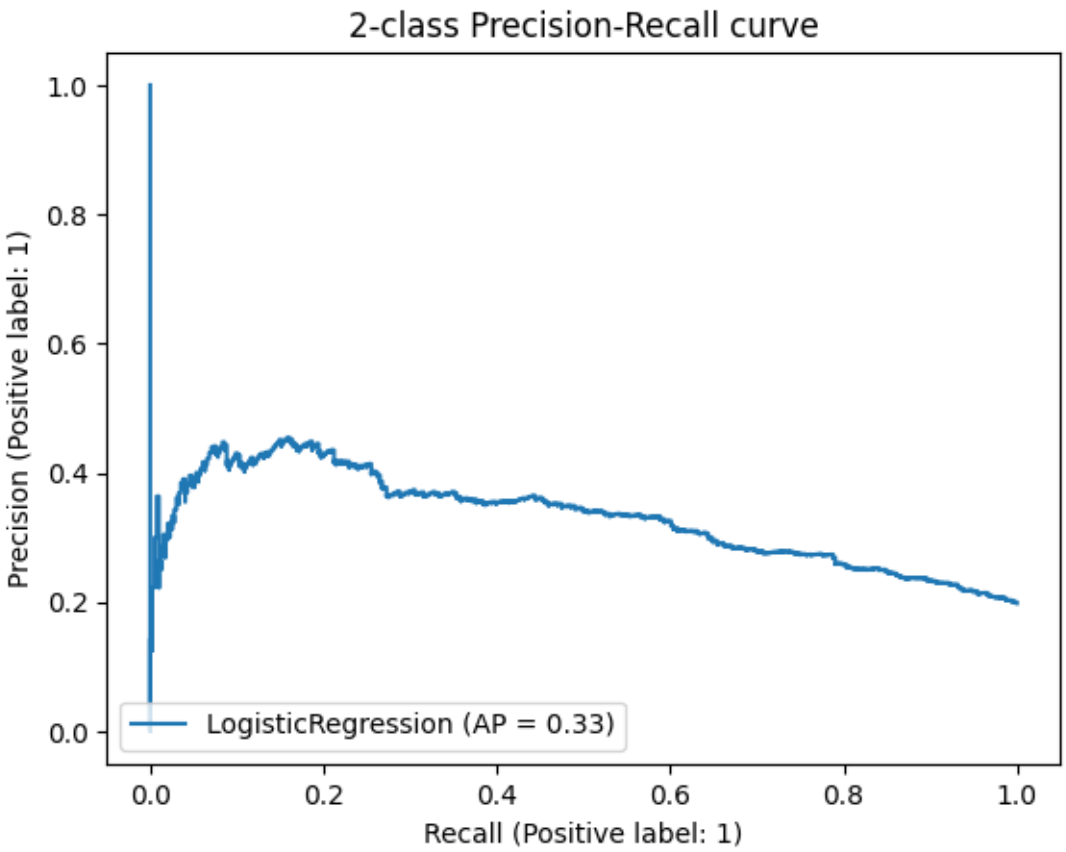
The first model I decided to build was a standard Logistic Regression. Logistic regression lends itself well to this dataset in two ways: it can be easily used for binary classification/prediction, and it is also very interpretable (meaning we can extract information about which attributes have the most predictive power). This is especially important for business-related data, as oftentimes one needs to explain the results from a model to non-technical stakeholders.

As mentioned above, I dropped the variables France, Spain, and Male from the dataset prior to fitting the Logistic Regression model. I also fed the model the orginal unscaled/unstandardized data, both to aid in interpretation of the model, but also because standardizing/scaling is unnecessary

for Logistic Regression.

The fitted Logistic Regression model displayed medicore classification performance. The accuracy score for the test data was 0.7965 (or in other words 79.65% of the test data points were correctly classified). The ROC Curve, AUC, and Precision-Recall Curve displayed fair/poor performance as well. Perfect AUC and AP scores are 1.0, so scores of 0.68 and 0.33 do not reflect a particularly strong model. Plots displaying these metrics may be found below:





In addition to evaluating the model based off the above metrics, I decided to also more closely examine the Confusion Matrix and corresponding Precision/Recall/F1-Scores of the Logistic Regression model for different thresholds. The above model used a default threshold of 0.5 which is the standard threshold for Logistic Regression models, however, sometimes better predictive performance can be achieved using a different threshold. Below is a summary of results for the test data for the thresholds 0.1, 0.25, 0.3, 0.5, 0.6. During my actual model building I tested even more thresholds, however, I felt that these 5 provided an adequate representation of the varying results one can get by altering threshold.

```
******** Logistic Regression model where threshold = 0.1 ******
Accuracy/Score is 0.291
Confusion Matrix:
[[ 200 1404]
 [  14  382]]
              precision    recall  f1-score   support

           0       0.93      0.12      0.22      1604
           1       0.21      0.96      0.35       396

    accuracy                           0.29      2000
```

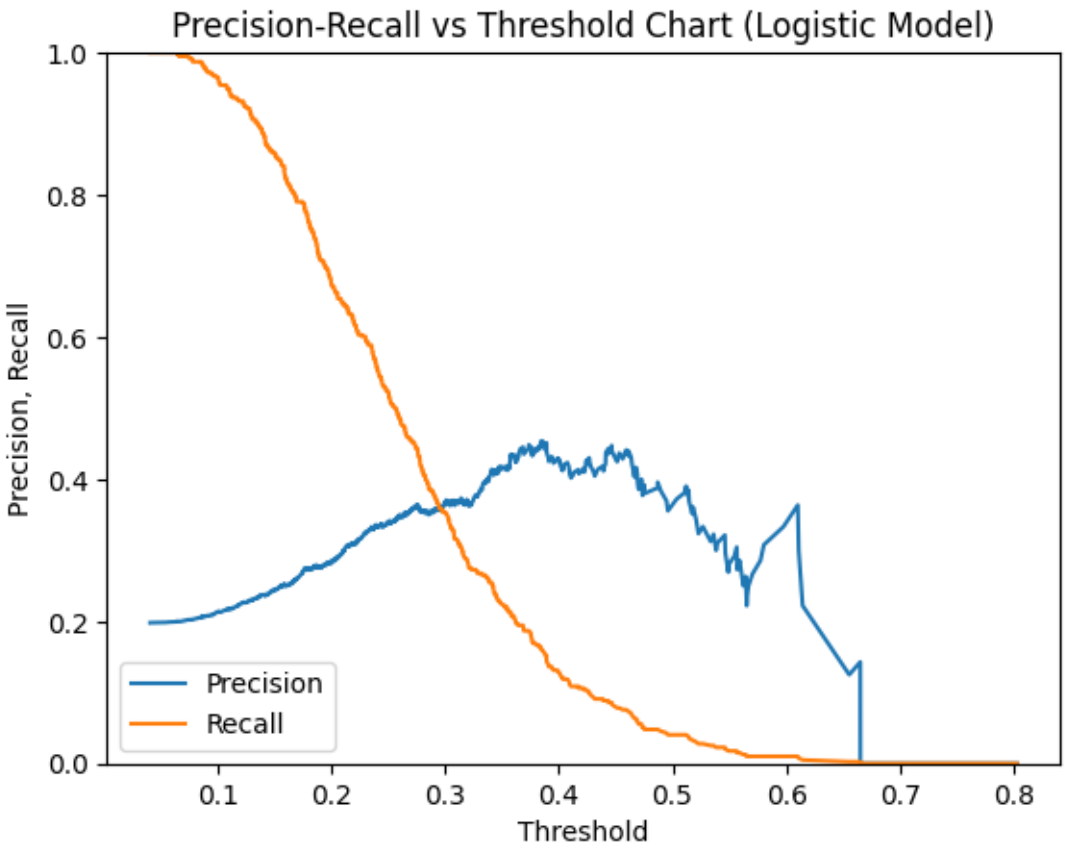
```
   macro avg       0.57      0.54      0.29      2000
weighted avg       0.79      0.29      0.25      2000


******** Logistic Regression model where threshold = 0.25 ******
Accuracy/Score is 0.7045
Confusion Matrix:
[[1203  401]
 [ 190  206]]
              precision    recall  f1-score   support

           0       0.86      0.75      0.80      1604
           1       0.34      0.52      0.41       396

    accuracy                           0.70      2000
   macro avg       0.60      0.64      0.61      2000
weighted avg       0.76      0.70      0.73      2000


******** Logistic Regression model where threshold = 0.3 ******
Accuracy/Score is 0.7515
Confusion Matrix:
[[1363  241]
 [ 256  140]]
              precision    recall  f1-score   support

           0       0.84      0.85      0.85      1604
           1       0.37      0.35      0.36       396

    accuracy                           0.75      2000
   macro avg       0.60      0.60      0.60      2000
weighted avg       0.75      0.75      0.75      2000


******** Logistic Regression model where threshold = 0.5 ******
Accuracy/Score is 0.7965
Confusion Matrix:
[[1577   27]
 [ 380   16]]
              precision    recall  f1-score   support

           0       0.81      0.98      0.89      1604
           1       0.37      0.04      0.07       396

    accuracy                           0.80      2000
   macro avg       0.59      0.51      0.48      2000
weighted avg       0.72      0.80      0.72      2000
```

```
******** Logistic Regression model where threshold = 0.6 ******
Accuracy/Score is 0.8005
Confusion Matrix:
[[1597    7]
 [ 392    4]]
              precision    recall  f1-score   support

           0       0.80      1.00      0.89      1604
           1       0.36      0.01      0.02       396

    accuracy                           0.80      2000
   macro avg       0.58      0.50      0.45      2000
weighted avg       0.72      0.80      0.72      2000
```

First, I'll highlight the results from the logistic regression model where the threshold was 0.5. Again, this is the standard threshold that was discussed above. We can see that the accuracy score for this model is 0.7965. However, closer examination of the Confusion Matrix reveals that the recall score for customers who churned is just 4% (or in other words, only 16 out of 396 customers who churned were correctly identified). This is pretty abysmal performance, and is a sign that the model would be almost entirely useless to a business who wanted to predict customer churn. This result also emphasizes the importance of not just relying on a single metric when evaluating classification models, but evaluating results holistically. I'll also add that because the dataset was unbalanced to begin with (roughly 20% of the customers in the dataset represented customers who churned, and 80% didn't), this means that looking at metrics such as Accuracy on their own can be very deceptive in terms of evaluating model strength. While I had initially planned to choose a threshold for the logistic regression model via cross-validation optimizing for Accuracy, these results revealed that choosing a threshold that way would be antithetical to the ultimate goal of predicting customer churn.

Interestingly, however, if were to lower the classification threshold, it appears that we can get a more balanced model that doesn't sacrifice too much accuracy. For example, for a threshold of 0.25, the recall for customers who churned is 52% (a significant improvement over 4%); the overall accuracy is 70.45%. So even though the overall acuraccy has decreased by ~9%, this is a model that may actually provide more value to a business than the model with the 0.5 threshold.

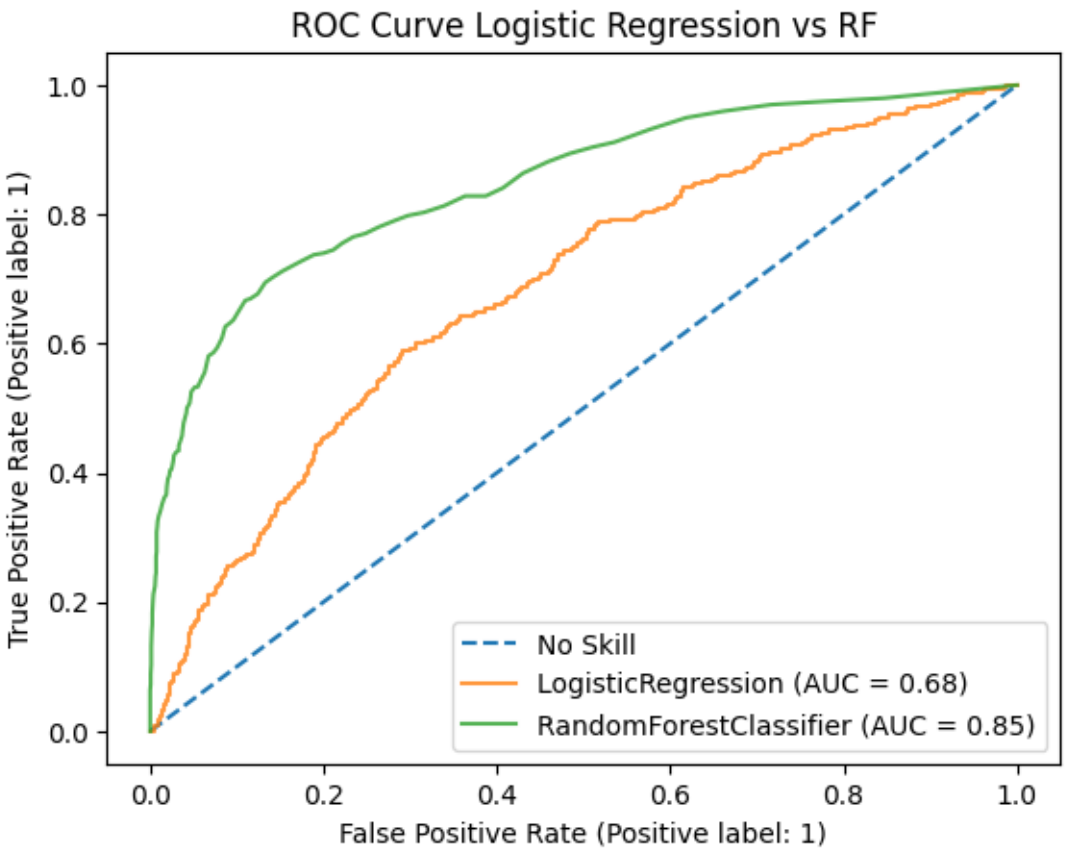
Further solidifying this idea is the plot below showing the differeing Precision/Recall scores as the threshold for the logistic regression model changes. This again reinforces the notion that for this particular model, a lower threshold may provide more value in identifying customers who churn.

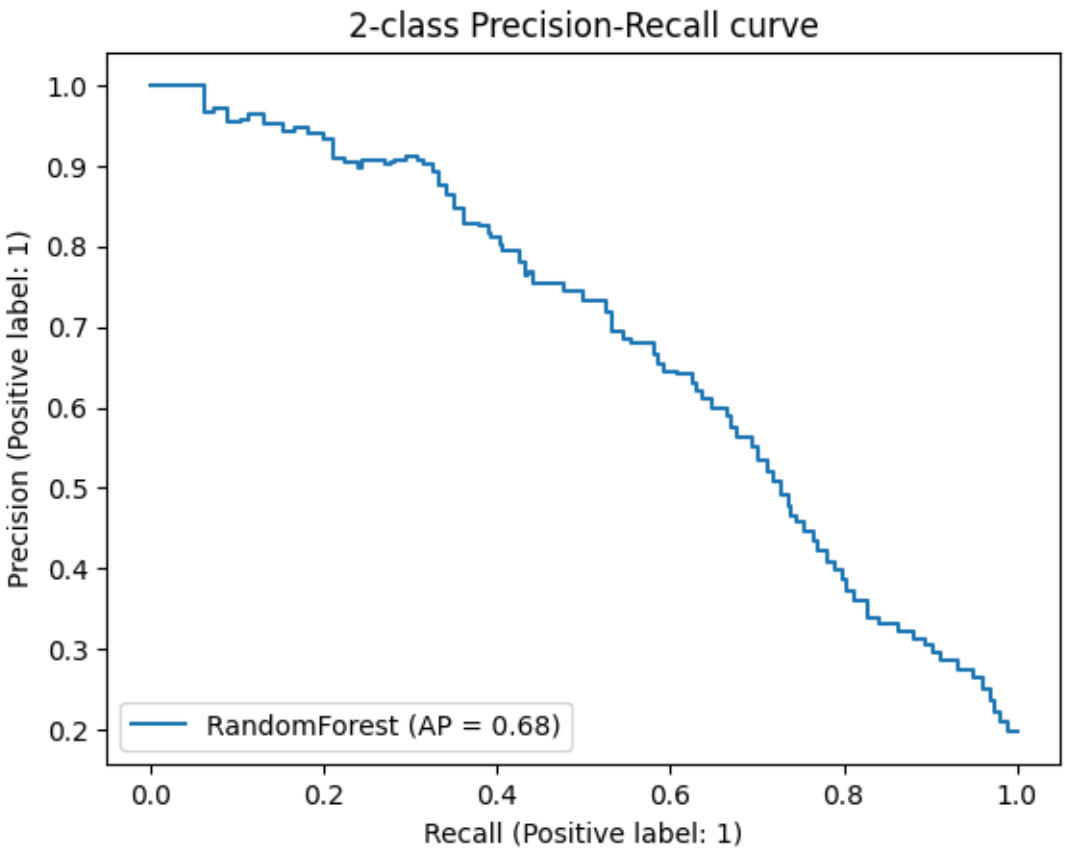


Ultimately, selecting the best threshold for this model would come down to the needs and costs of that business. In the real world, a business could assign actual costs related to customer churn which would then be used in conjunction with model building and selection. For example, costs could be assigned to incorrectly identifying a loyal customer as one who'd be likely to churn, or and costs for not identifying a customer who will leave (where presumably intervention could have prevented churn). All of this information could be inrcorporated into choosing the final model.

Random Forest

The next classification model I built was a Random Forest. I was hopeful that the Random Forest would display better predictive performance than the Logistic Regression, as Random Forests are known for being particularly robust and having strong predictive prowess. This did indeed prove to be the case. See below for the ROC/AUC, Precision-Recall Curve, and Confusion Matrix for the final Random Forest model. The final random forest was built using 100 trees, and was fit to the same variables as the Logistic Regression model (those chosen by Lasso).


ROC Curve Logistic Regression vs RF
True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
0.0
0.2
0.4
0.6
0.8
1.0
No Skill
LogisticRegression (AUC = 0.68)
RandomForestClassifier (AUC = 0.85)




```
Random Forest Results
Accuracy/Score is 0.866
Confusion Matrix:
[[1537   67]
 [ 201  195]]
              precision    recall  f1-score   support

           0       0.88      0.96      0.92      1604
           1       0.74      0.49      0.59       396

    accuracy                           0.87      2000
   macro avg       0.81      0.73      0.76      2000
weighted avg       0.86      0.87      0.86      2000
```

The above plots and output clearly show that the Random Forest is superior to the Logistic Regression model in a multitude of ways. First, the AUC and AP scores are significantly improved. Second, the overall Accuracy of the model is better at 86.6%, but perhaps even more importantly, the recall for churn individuals is now 49%. Thus, the Random Forest model was more accurate than the Logistic Regression model on every front.

Additionally, I extracted the feature importances from the Random Forest model and identified those that were considered by the Random Forest model to be the "most important". A list of those features identified as most important are below:

- Credit Score
- Age
- Products Number
- Estimated Salary

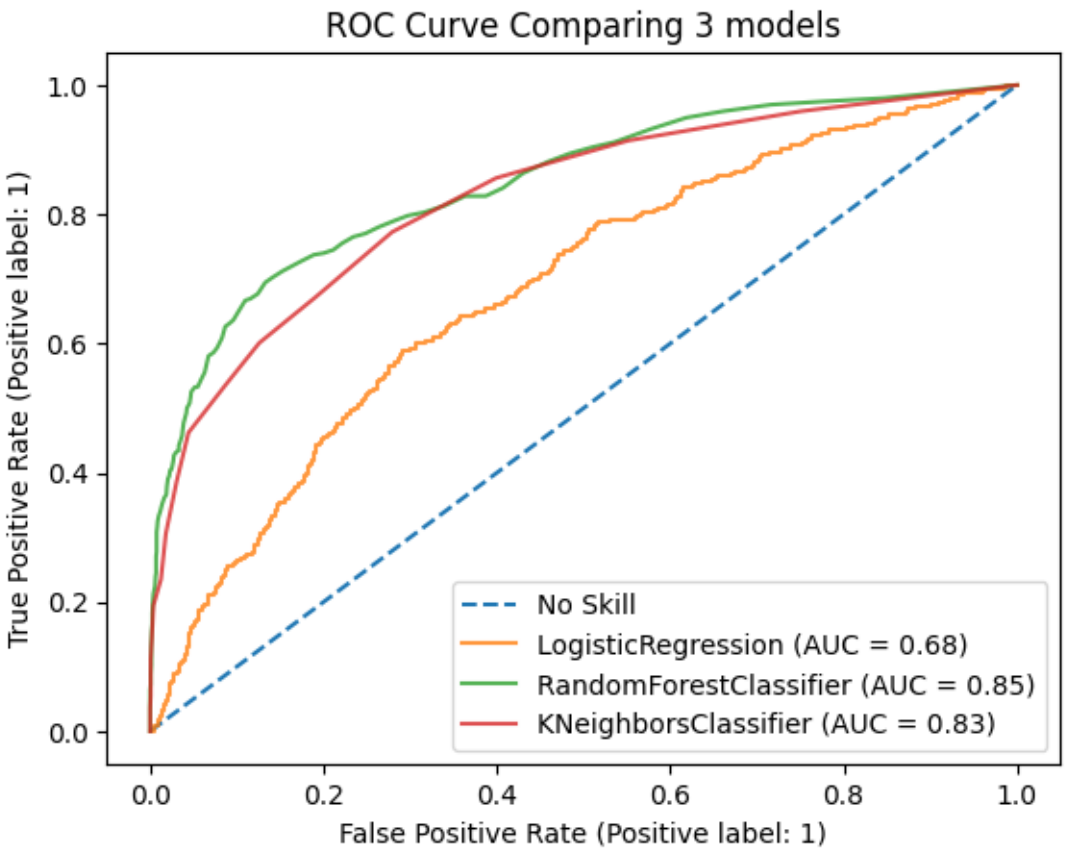
Interestingly, these features don't overlap much with the top 4 features (magnitude-wise) identified by Lasso (with the exception of the Age variable). Perhaps this is not surprising, however. It should be noted that the most important features identified by the Random Forest are all numeric. It could be the case that these variables were identified as most important because these features allowed the Random Forest the most flexibility in terms of splitting out different groups. I will discuss this further in the Evaluation and Final Results section.

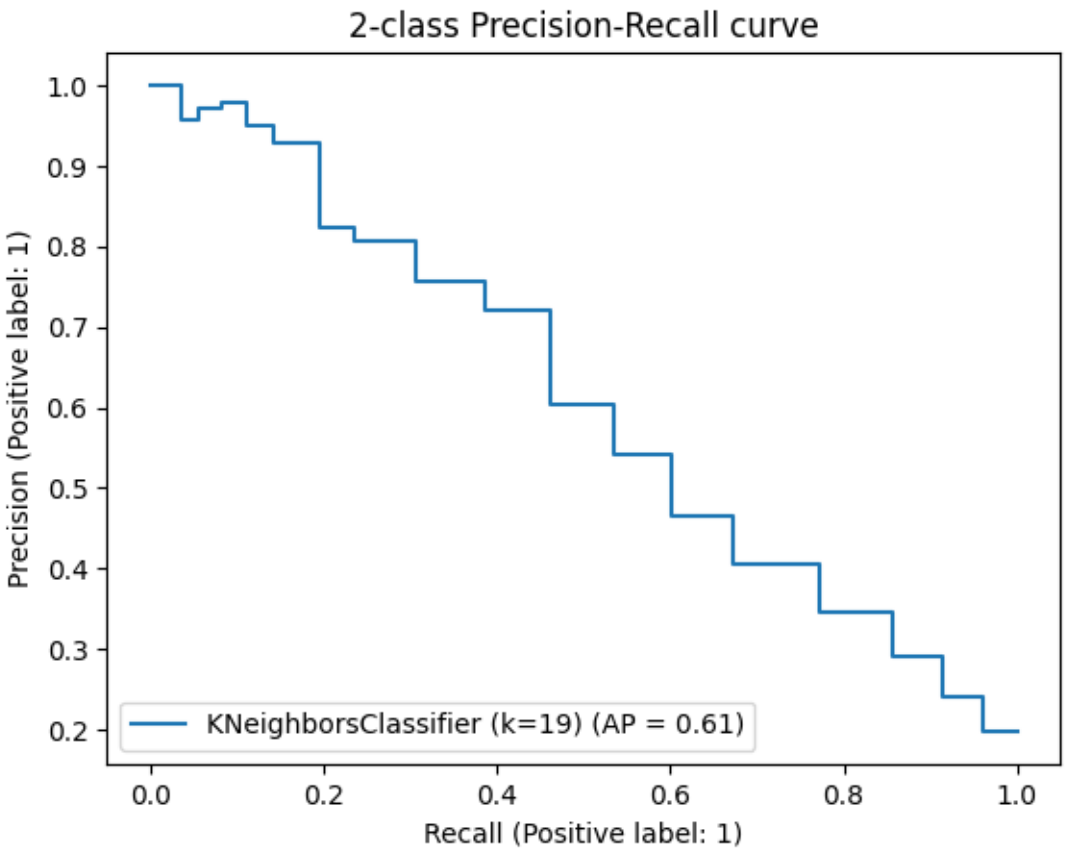
Refitting of Models

Before moving on to additional model building, I experimented with some model refitting for the Logistic Regression and Random Forest models. Given that the Random Forest had identified a smaller subset of variables than Lasso, I decided to re-fit the Logistic Regression and Random Forest using the Random Forest "most important" features (credit_score, age, products_number, estimated_salary) to see whether this would have a postive impact on either model. Ultimately, however, this did not improve performance for either model – both still achieved best performance when fit to the Lasso chosen variables. Given the size of the dataset, and the fact that multi-collinearity was not detected during the exploratory data analysis phase, this is perhaps not a surprising result. Still, it provided confirmation that Lasso had identified a solid group of variables to use for model fit.

KNN Classifier

The next model I built was a k-nearest neighbors classifier. Again, this model was fit to the variables chosen by Lasso, however, because KNN is a distance-based algorithm, the data was also scaled before being fed into the model. The best value of k was selected via 10-fold GridSearchCV using "accuracy" as the scoring metric. A k=19 was chosen for the final model. Below is a summary of the results:


ROC Curve Comparing 3 models
True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
0.0
0.2
0.4
0.6
0.8
1.0
No Skill
LogisticRegression (AUC = 0.68)
RandomForestClassifier (AUC = 0.85)
KNeighborsClassifier (AUC = 0.83)




```
KNN Results when k=19
Accuracy/Score is 0.848
[[1575   29]
 [ 275  121]]
              precision    recall  f1-score   support

           0       0.85      0.98      0.91      1604
           1       0.81      0.31      0.44       396

    accuracy                           0.85      2000
   macro avg       0.83      0.64      0.68      2000
weighted avg       0.84      0.85      0.82      2000
```

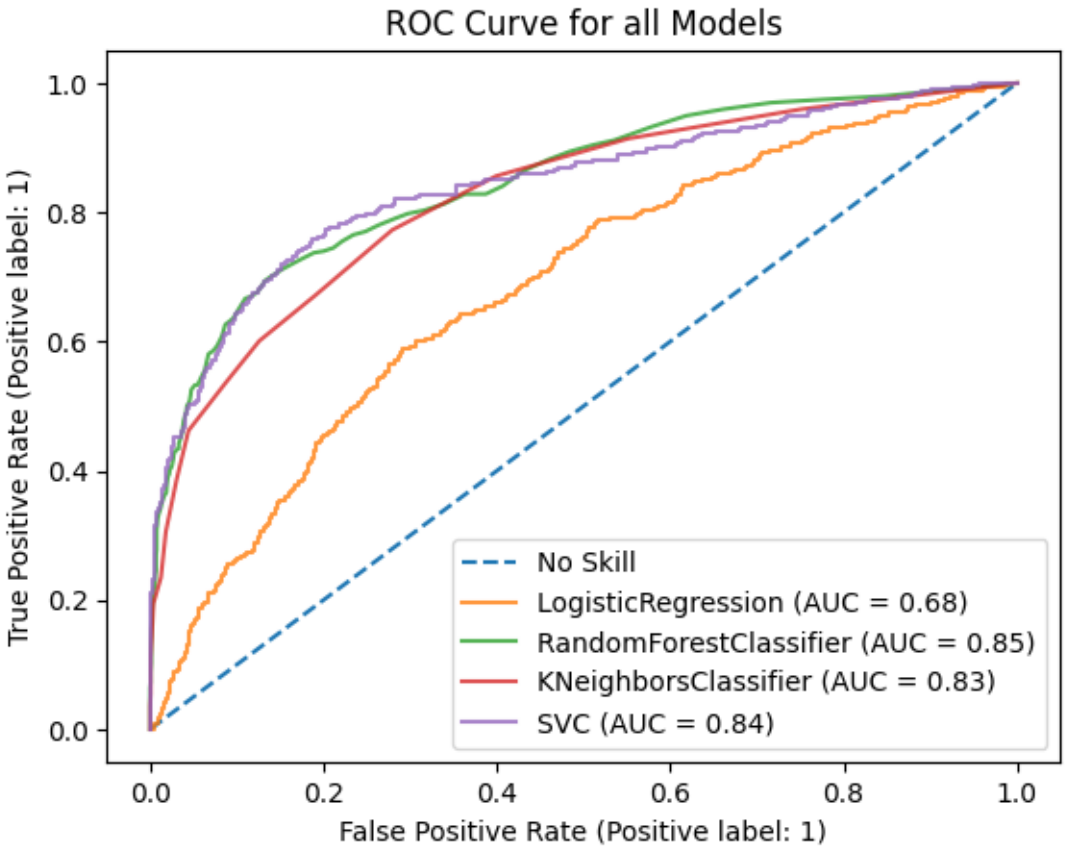
As can be seen above, the K-nearest neighbor model achieved performance somewhere between the Logistic Regression and Random Forest models. Accuracy, AUC, and AP scores were all fairly close to the Random Forest scores, however, the recall for the churn group was 31%. This recall was better than that achieved by the Logistic Regression, but still not as good as the Random Forest.

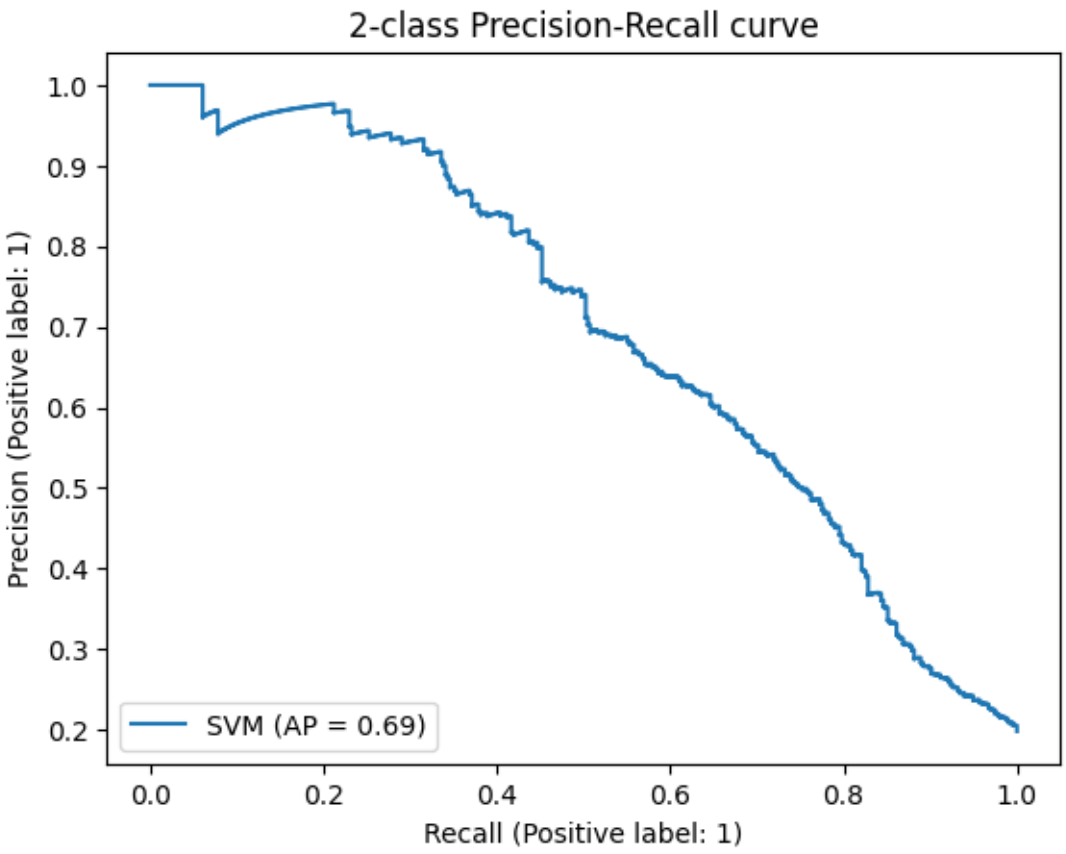
However, I would note that depending on the business's needs and priorities, one could tune the k parameter to achieve different results. Through some experimentation, I found that recall could

be improved if one was willing to sacrifice some of the accuracy.

Support Vector Machine

The final model I built was a Support Vector Machine using the . This model was again fit to the Lasso chosen variables, and the data was scaled before being fed into the model. 10-fold GridSearchCV was used for the regularization parameter C, with a C of 0.5 and the 'rbf' kernel ultimately being chosen. A summary of the results may be found below:






```
SVM Results
Accuracy/Score is 0.8675
[[1573   31]
 [ 234  162]]
              precision    recall  f1-score   support

           0       0.87      0.98      0.92      1604
           1       0.84      0.41      0.55       396

    accuracy                           0.87      2000
   macro avg       0.85      0.69      0.74      2000
weighted avg       0.86      0.87      0.85      2000
```

In terms of model performance, the Support Vector Machine achieved a close second behind the Random Forest. It had nearly identical Accuracy, AUC, and AP scores to the Random Forest. The only area where the Random Forest outperformed the Support Vector Machine was with recall for the churn group. The Support Vector Machine had a recall of 41% for this group, whereas the Random Forest achieved 49%.

Evaluation and Final Results

In this section, I will dive a little deeper into comparing and constrating the 4 classification models that I built. I will also include a discussion and interpretation of the dataset variables.

Model Comparison

A summary of the metrics used to evaluate each model can be found in the table below. I've also included a list summarizing some additional pros and cons of each model type.

Model Type	Accuracy	AUC	AP	Recall
Logistic Regression	0.7965	.68	.33	.04
Random Forest	0.866	.85	.68	.49
K-Nearest Neighbors	0.848	.83	.61	.31
Support Vector Machine	0.8675	.84	.69	.41

Logistic Regression

Pros:

- Easy to implement/understand
- Predictors/coefficients are interpretable
- Can output predictions as probabilities

Cons:

- Assumes predictor variables have a linear relationship with the logit of the response

Random Forest

Pros:

- Can handle linear and non-linear relationships
- Balances bias-variance tradeoff well
- Typically have high accuracy
- Less influenced by outliers

Cons:

- Computationally intensive
- Recognizes important predictors, but they are not easily interpretable

K-Nearest Neighbors

Pros:

- Easy to implement/understand
- Non-parametric model (no model assumptions)

Cons:

- Computationally intensive
- Sensitive to outliers
- Low predictor interpretability

Support Vector Machine

Pros:

- Memory efficient
- Works well for high-dimensional data

Cons:

- Computationally intensive
- Low predictor interpretability

Based off the above table, I believe it is clear that the Random Forest and Support Vector Machine were the top two performing models. These two models achieved reasonably high accuracy without sacrificing recall or AP. If choosing between these two models, I would select the Random Forest due to two reasons. First, its recall score is 8% higher than the Support Vector Machine, which in the scope of things, could have significant cost savings for a business. Second, I believe the output of a Random Forest is slightly more interpretable than a Support Vector Machine. We can look at the Random Forest's "most important" predictors as a way to glean which information is most useful in predicting churn. With Support Vector Machines, it's difficult to interpert and assign meaning to the feature weights, particular when a non-linear kernel is used (as was in this case).

Still, Random Forest models have their drawbacks. In this case, I believe the largest drawback is that fact that (even though we can identify the most important predictors) the predictors are not interpretable in the way they would be in a Logistic Regression model, for example. I will discuss this further in the next section.

Interpretation of Variables

Conclusion

Citations

- "Bank Customer Churn", https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset
- "The Value of Keeping the Right Customers", Amy Gallo, https://hbr.org/2014/10/the-value-of-keeping-the-right-customers