Predictive Modeling Of Customer Attrition

Tracey Zicherman

## Executive Summary

***SIDEKICK: TO DO - Complete after other sections.***

## Approach and Data

### Overall Approach

Our overall approach to solving the problem of predictive modeling of customer churn can be summarized as follows:

1. Perform basic cleaning and manipulating of raw customer data.
2. Explore raw customer data to understand features of customer behavior and possible relationships to customer churn, with an eye towards predictive modeling.
3. Engineer new customer features from pre-existing features.
4. Perform additional steps to maximize the predictive values of both pre-existing and engineered features
5. Process the resulting data to prepare for predictive modeling with machine learning algorithms.
6. Fit an tune 5 baseline classification machine learning algorithms which be used to predict whether a given customer will churn based on features.
7. Evaluate the tuned baseline models on a number of different metrics to determine which is the best candidate for a final model.
8. Select a final model and interpret it to gain insights into the relationship between customer features and customer churn.

### Analytic and Informational Goals

The overall, high level goal, is to learn as much as possible about the relationship between customer behavior, as measured by the features present in the dataset, and customer churn (attrition), and also to predict whether a customer will churn or not, based on a given set of values for these features, as accurately as possible.

Both interpretative and predictive modeling have clear value – for example, both can be used to guide business decisions and policy, to and understand the customer base. It is perhaps helpful to think of them as complementary, since they may inform different decisions. For example, interpreting the relationship between demographic features and customer churn can inform marketing campaigns or retention policy, while predicting whether particular customers will churn can inform other strategic decisions.

When the two goals are in conflict, as they sometimes are in predictive modeling, we will tend to prefer making choices that favor the predictive modeling side.

### Data

#### Data Dictionary

This is data dictionary for this dataset, which includes each customer feature and a brief description.

We note the following:

1. The client number is a unique identifier for internal purposes, and should have no real predictive value for the customer attrition.
2. Customer attrition is measured by a binary (true/false) feature which says whether or not the given account is closed. This will be our predictive target, the feature we are aiming to predict.
3. There are 4 demographic features, corresponding to customer’s age, gender, number of dependents, education level, marital status, and level of income.
4. A single product feature described the customer’s credit card category.
5. The remaining 12 features describe various aspects of the customer’s credit card related behavior.

We observe features which fall into a number of different categories. For example, there are demographic features (which *a proiri* contain no time information) and financial features, with time information (e.g. Total\_Trans\_Amt) and without time information (e.g. Credit\_Limit).

Some features with time information measure customer behavior over the last twelve months (e.g. Months\_Inactive\_12\_mon) while others measure on a longer time scale (e.g. Months\_on\_book) so we can assume that data are a snapshot of customer behavior over different timescales.

The predictive modeling task can be thought of as - given this snapshot of customer features at a moment in time, what is the likelihood their account will be closed at that same moment.

Given such a disparate mix of features, we were optimistic that we would be able to predict customer churn with a reasonable high level of accuracy, and this was borne out in our investigation.

#### Load Data

Here is a snapshot of the dataset

## # A tibble: 10,127 × 21  
## CLIENTNUM Attrition\_Flag Customer\_Age Gender Dependent\_count Education\_Level  
## <int> <chr> <int> <chr> <int> <chr>   
## 1 768805383 Existing Custo… 45 M 3 High School   
## 2 818770008 Existing Custo… 49 F 5 Graduate   
## 3 713982108 Existing Custo… 51 M 3 Graduate   
## 4 769911858 Existing Custo… 40 F 4 High School   
## 5 709106358 Existing Custo… 40 M 3 Uneducated   
## 6 713061558 Existing Custo… 44 M 2 Graduate   
## 7 810347208 Existing Custo… 51 M 4 Unknown   
## 8 818906208 Existing Custo… 32 M 0 High School   
## 9 710930508 Existing Custo… 37 M 3 Uneducated   
## 10 719661558 Existing Custo… 48 M 2 Graduate   
## # ℹ 10,117 more rows  
## # ℹ 15 more variables: Marital\_Status <chr>, Income\_Category <chr>,  
## # Card\_Category <chr>, Months\_on\_book <int>, Total\_Relationship\_Count <int>,  
## # Months\_Inactive\_12\_mon <int>, Contacts\_Count\_12\_mon <int>,  
## # Credit\_Limit <dbl>, Total\_Revolving\_Bal <int>, Avg\_Open\_To\_Buy <dbl>,  
## # Total\_Amt\_Chng\_Q4\_Q1 <dbl>, Total\_Trans\_Amt <int>, Total\_Trans\_Ct <int>,  
## # Total\_Ct\_Chng\_Q4\_Q1 <dbl>, Avg\_Utilization\_Ratio <dbl>

#### Data Preparation

In preliminary work, we investigation determined that errors and duplicates did not appear to be an issue. The amount of data processing needed was minimal, and included:

1. ***Removing unnecessary or problematic features***. Since the client number has no known predictive value, this was removed from the data before modeling.
2. ***Enforcing the correct data types for features***. This step enabled the features to be recognized correctly by modeling algorithms.
3. ***Encoding missing values***. Since roughly 30% of the data contained missing values, these were encoded using the “Unknown” label, which could be processed as an additional possibility for the predictive modeling step.

After preliminary cleaning steps 2 the dataset looks ready for feature engineering. All remaining features are relevant, and are represented by the correct data types, namely fct for categorical features, int for discrete numerical (integer) features, and dbl for continuous numerical features.

#### Descriptive Statistics and Visualizations

Now we present some statistics and plots relevant to the overall predictive modeling task, specifically with an eye towards feature engineering.

***SIDEKICK: What we have included is mostly univariate exploratory data analysis [EDA], which looks at individual features. Under normal circumstances, given enough time we would also include more bivariate analysis, which looks at the relationship between pairs of features, and specifically where one member of the pair is the target feature, customer attrition. What we have should be sufficient for the course, given the instructor’s response to the Preliminary Steps.***

##### Categorical Features

The categorical features present in the dataset are

## [1] "Attrition\_Flag" "Gender" "Education\_Level" "Marital\_Status"   
## [5] "Income\_Category" "Card\_Category"

Now, we visualize the distribution in the dataset of all categorical features. Our main goals with this visualization are:

1. To get a quick basic visual sense of the how these features are distributed among the customers represented in the data
2. Inform feature engineering, both the creation of new features, and the modification or transformation of existing ones.
3. Identify any potential issues or corrective action that might be advisable before using these features for predictive modeling.

From these distribution plots, we make the following observations:

* The proportion of churned customers (“attrited” customers, i.e. customers with closed accounts) is relatively low compared to customers who haven’t churned.
* The overall proportions of female and male customers are more balanced, but with a slightly higher proportion of females.
* The most frequently occurring educational level was graduate, followed by high school. There were relatively fewer customers with doctorate or post-graduate educational level. A reasonably high proportion of customers have unknown educational level
* Almost all customers were either married or single, with a relatively small minority being divorced or of unknown status.
* The most frequently occurring income level by far was the lowest, namely Income\_Category == 'Less than $40k. Whether this reflects something about the customer base or the underlying population is unknown.

We note that customer attrition, marital status, and card category stand out as somewhat unbalanced features. First we look at their individual distributions in the data.

## # A tibble: 2 × 3  
## Attrition\_Flag Count Proportion  
## <fct> <int> <dbl>  
## 1 Attrited Customer 1627 0.16  
## 2 Existing Customer 8500 0.84

## # A tibble: 4 × 3  
## Marital\_Status Count Proportion  
## <fct> <int> <dbl>  
## 1 Divorced 748 0.07  
## 2 Married 4687 0.46  
## 3 Single 3943 0.39  
## 4 Unknown 749 0.07

## # A tibble: 4 × 3  
## Card\_Category Count Proportion  
## <fct> <int> <dbl>  
## 1 Blue 9436 0.93  
## 2 Gold 116 0.01  
## 3 Platinum 20 0   
## 4 Silver 555 0.05

Since customer attrition is the predictive target, and the imbalance is relatively moderate, we won’t take any corrective measures here. Card category is however extremely unbalanced, so we will re-code this into two categories.

Finally for marital status, divorced and unknown values only represent small proportions of customers, but it doesn’t make sense to bin these (since information will be erased). Since the proportions aren’t too small (single digit percentages), we won’t take any corrective measures for this feature either.

##### Numerical Features

Now we turn our attention to numerical (continuous or integer) features. As with the categorical features, we aim to get a sense of how the numerical features are distributed, inform feature engineering, and identify and correct any issues with these features that may affect predictive modeling.

These are numerical features present in the dataset

## [1] "Customer\_Age" "Dependent\_count"   
## [3] "Months\_on\_book" "Total\_Relationship\_Count"  
## [5] "Months\_Inactive\_12\_mon" "Contacts\_Count\_12\_mon"   
## [7] "Credit\_Limit" "Total\_Revolving\_Bal"   
## [9] "Avg\_Open\_To\_Buy" "Total\_Amt\_Chng\_Q4\_Q1"   
## [11] "Total\_Trans\_Amt" "Total\_Trans\_Ct"   
## [13] "Total\_Ct\_Chng\_Q4\_Q1" "Avg\_Utilization\_Ratio"

Now let’s look at histograms for the first seven numerical features.

We note that Customer\_Age, Dependent\_Count, Total\_Relationship\_Count and Contacts\_Count\_12\_mon seem relatively well behaved (e.g, symmetric).

The distribution of Credit\_Limit has some curious features - there is clearly very large proportion of customers with small credit limits. There is also an extreme right skew, and a curious spike at the maximum of the range. This is an excellent example of a situation in which binning is called for. For these reasons, we will bin Credit\_Limit, to create a new ordinal feature with 7 levels.

Furthermore, Months\_on\_book has a single strange spike near the middle of the range.

## # A tibble: 1 × 2  
## Months\_on\_book Count  
## <int> <int>  
## 1 36 2463

This represents of the dataset, which is very unusual. Since this is too large a percentage to drop, we will choose to bin this feature. We will also bin Months\_Inactive\_12\_mon to lump the values with small proportions together.

Now we plot the dataset distributions of the remaining eight numerical features

We note that Total\_Revolving\_Bal and Avg\_Utilization\_Ratio have a large spike around 0 and an extreme right skew. We will bin these as ordinal features with a low number of levels, with an extra lowest level for the zero value.

Avg\_Open\_To\_Buy has a similar distribution compared to Credit\_Limit with a high right skew and a spike at the lowest part of the range. Total\_Amt\_Chng\_Q4\_Q1 and Total\_Ct\_Chng\_Q4\_Q1 also both show extreme right skew. We will transform these features to try to reduce skew.

Finally, Total\_Trans\_Amt and Total\_Trans\_Ct are clearly multimodal distributions. For this reason, we will bin these features as well.

Here is a summary of numerical features which need further engineering:

1. Transformations: Credit\_Limit, Avg\_Open\_To\_Buy, Total\_Amt\_Chng\_Q4\_Q1, Total\_Ct\_Chng\_Q4\_Q1
2. Binning: Card\_Category, Months\_on\_book, Months\_Inactive\_12\_mon, Total\_Revolving\_Bal, Avg\_Open\_To\_Buy, Avg\_Utilization\_Ratio, Total\_Trans\_Amt, Total\_Trans\_Ct

#### Feature Engineering

Here the intention is to create features which are both meaningfully meaningfully interpretable and have additional predictive value. The exploratory analysis, in particular the distribution plots, as well as domain knowledge, informed the decision to create the following new features from pre-existing ones:

* Avg\_Trans = Total\_Trans\_Amt / Total\_Trans\_Ct - Average amount of customer spending per transaction.
* Avg\_Mon\_Spend = Total\_Trans\_Amt / Months\_on\_book - Average monthly spending of the customer over the duration of their recorded relationship with the bank.
* Trans\_Freq = Total\_Trans\_Ct / Months\_on\_book - Average number of transactions per month of the recorded customer duration.
* Credit\_Use\_Ratio = Total\_Revolving\_Bal / Credit\_Limit - Proportion of available credit utilized, a well known indicator of customer credit health.

Further exploratory analysis in the preliminary steps showed that these features all had sufficient predictive value to justify including them in the predictive modeling.

##### Create New Features

In this section, we create the new features. First we check for zero values, that could be a problem for division.

## 0 customers were removed before filtering for zero values

And here is a snapshot of the features.

# create tibble for modeling  
model\_data <- cleaned\_data  
  
# average transaction  
model\_data$Avg\_Trans <- model\_data$Total\_Trans\_Amt / model\_data$Total\_Trans\_Ct  
  
# average monthly spending  
model\_data$Avg\_Mon\_Spend <- model\_data$Total\_Trans\_Amt / model\_data$Months\_on\_book  
  
# average monthly transactions  
model\_data$Trans\_Freq <- model\_data$Total\_Trans\_Ct / model\_data$Months\_on\_book  
  
# average credit utilization ratio  
model\_data$Credit\_Use\_Ratio <- model\_data$Total\_Revolving\_Bal / model\_data$Credit\_Limit  
  
# get engineered columns  
eng\_feat <- model\_data %>%  
 select(Avg\_Trans, Avg\_Mon\_Spend, Trans\_Freq, Credit\_Use\_Ratio)  
  
head(eng\_feat)

## # A tibble: 6 × 4  
## Avg\_Trans Avg\_Mon\_Spend Trans\_Freq Credit\_Use\_Ratio  
## <dbl> <dbl> <dbl> <dbl>  
## 1 27.2 29.3 1.08 0.0612  
## 2 39.1 29.3 0.75 0.105   
## 3 94.4 52.4 0.556 0   
## 4 58.6 34.4 0.588 0.760   
## 5 29.1 38.9 1.33 0   
## 6 45.3 30.2 0.667 0.311

And finally, some basic summary statistics on the new customer features

## Avg\_Trans Avg\_Mon\_Spend Trans\_Freq Credit\_Use\_Ratio   
## Min. : 19.14 Min. : 10.00 Min. :0.1887 Min. :0.00000   
## 1st Qu.: 47.51 1st Qu.: 62.36 1st Qu.:1.2727 1st Qu.:0.02271   
## Median : 55.79 Median : 105.80 Median :1.8571 Median :0.17565   
## Mean : 62.61 Mean : 131.01 Mean :1.9231 Mean :0.27489   
## 3rd Qu.: 65.48 3rd Qu.: 141.36 3rd Qu.:2.3611 3rd Qu.:0.50269   
## Max. :190.19 Max. :1256.85 Max. :9.7692 Max. :0.99877

##### Visualize New Feature Distributions

Now we look at the dataset distributions of features we just engineered, with the same intentions as above when we visualized the preexisting categorical and numerical dataset Distributions.

Each of these distributions are highly right skewed, suggesting a somewhat exponential distribution. It can be helpful to reduce skew as a precautionary step, since some models are better able to handle skew than others (for example in the latter case, if it is assumed the data is normally distributed).These distributions are not surprising, given the properties of the distributions of the features they were derived from.

We will bin Credit\_Utilization\_Ratio and transform the remaining new features. We note a large spike in the distribution at Credit\_Utilization\_Ratio == 0, indicating a large number of customers are retaining a zero balance.

##### Binning

In this section, we bin the following features: Card\_Category, Months\_on\_book, Months\_Inactive\_12\_mon, Total\_Revolving\_Bal, Avg\_Utilization\_Ratio, Total\_Trans\_Amt, Total\_Trans\_Ct, Credit\_Use\_Ratio. The choice of bins are made by hand in this case, although there are a wide range of options for doing so, including more advanced techniques.

Note that this binning will result in an ordinal (not categorical) feature because of the natural ordering reflecting the ordering of the underlying continuous feature.

Also note that the resulting numerical feature encoding will not necessarily be easily interpreted - for example, we will encode Months\_on\_book >= 4 as Months\_on\_book = 4, so that any value 4 months before the encoding will have code 4 afterwards (in that case, essentially, we are re-binning the upper end of the range).

Here is a summary of the re-encoding:

1. Card category will be re-grouped into two categories, 1 corresponding to Blue card category, 2 corresponding to all other card types.
2. Months on book will be binned into 6 bins, roughly corresponding to the number of years of book, with an additional category for Months\_on\_book == 36, since this represents roughly 25% of customers.
3. Months Inactive in the last twelve will binned into 4 bins, corresponding to 0 and 1, 2, 3, and more than 4 months.
4. Total Revolving Balances will be binned into six bins, corresponding to 0-500, 500-1000, 1000-1500, 1500-2000, and > 2000
5. Credit utilization ratio and average utilization will be binned into 6 bins of equal width 0-0.2, 0.2-0.4, 0.4-0.6, 0.6-0.8, and 0.8-1.0.
6. Total transaction amount will be binned into 5 bins, 0, >0 - 3000, 3000-5000, 5000-10000, and > 10000
7. Total transaction count will be binned into 3 bins, 0-50, 5-80, and > 80.

Let us inspect the distributions of the binned features after binning.

These distributions look much more appropriate for predictive modeling, with the exception of Card\_Category which is still extremely unbalanced after the binning. To decide if retaining a feature with this large of an imbalance is useful, let us consider its relationship to the predictive target

## Warning in RColorBrewer::brewer.pal(N, "Set2"): minimal value for n is 3, returning requested palette with 3 different levels  
## Warning in RColorBrewer::brewer.pal(N, "Set2"): minimal value for n is 3, returning requested palette with 3 different levels

This is a tough call, but given that there are only 100 customers which churned which also have non-blue card category, there is a strong case that the relationship with the predictive target is too weak to justify including it. However, we note that overall, the ratio of features to observations is << 1.

## Ratio of features to observations is 0.002

Furthermore, advanced modern models such as xgboost may be able to make use of the slight amount of information this feature provides, so we choose to keep it for modeling.

##### Transformations

Here we apply the logarithm transformation to the following features: Credit\_Limit, Avg\_Open\_To\_Buy, Total\_Amt\_Chng\_Q4\_Q1, Total\_Ct\_Chng\_Q4\_Q1, Avg\_Trans, Avg\_Mon\_Spend, Trans\_Freq.

This transformation is intended reduce skew (note that the logarithm is the inverse of the exponential transformation). Note we cannot straightforwardly apply this transformation to any zero values of these features. To get around this issue, we can drop these rows, or perform a not uncommon step of imputing a tiny but negligible non zero value. Since these customers may be customers which churned and there are so few of these in the dataset, we take the latter approach.

Again, these distributions look much more suitable for predictive modeling. In particular, they are closer to normally distributed.

#### Predictive Modeling

In this section we model the relationship between the customer features (preexisting and engineered) and customer attrition, with an eye towards maximizing predictive accuracy.

We perform the final preprocessing steps, then split the data into train and test (holdout) data. Using the training data, we fit and tune five baseline classifier predictive models based on well-known machine learning classification algorithms. We then evaluate these using standard metrics, and comparison with a null (“worst-case”) model, thereby selecting a top choice.

##### Preprocessing

First we perform the final preprocessing steps necessary to get the data in a final form suitable for fitting machine learning algorithms.

###### Train Test Split

A fairly common 75%-25% train-test split is performed. The training data will be used to tune the baseline predictive models a process called cross-validation, which provides a more accurate estimate of the model on unseen data, while the test data will be held out until the last step, to provide a final estimate of future model predictive accuracy.

## Train set dimensions: 7595 24

## Test set dimensions: 2532 24

###### Preprocessing Recipe

We next perform the following preprocessing steps

1. step\_other - this pools infrequently occurring values of categorical features into another category called "other"
2. step\_dummy - this one-hot encodes all categorical features.
3. step\_center - this centers all features (which are all numeric by this stage in the processing pipeline) by subtracting them from their mean.
4. step\_scale - this scales all features by dividing them by their standard deviation.

##### Hyperparameter Tuning with Cross-Validation on Train Data

In this section we tune the hyperparameters of the 5 baseline classification algorithms: logistic regression, k-nearest neighbors, naive Bayes, random forest, and gradient boosted machines (specifically, XGBoost).

We use 5-fold cross-validation to tune the model to the training data, to find the best hyperparameters for each model. Optimization is with respect to the area under the receiver operating characteristic curve (ROC-AUC), a good metric for a balanced classifier with a binary target.

To save training and computational resource time, we only tune a single hyperparameter for each model. We also use 5 CV folds, rather than 10 as in the preliminary steps. In both cases, experimentation revealed little additional benefit to exploring more hyperparameters or using a greater number of folds.

Now, we tune the models

# Prepare and bake the recipe  
baked\_data <- rec %>%  
 prep(training = train\_set) %>%  
 bake(new\_data = NULL)  
  
# Check for missing values in the baked data  
na\_counts <- baked\_data %>%  
 summarise(across(everything(), ~sum(is.na(.), na.rm = FALSE)))  
  
na\_counts

## # A tibble: 1 × 35  
## Customer\_Age Dependent\_count Months\_on\_book Total\_Relationship\_Count  
## <int> <int> <int> <int>  
## 1 0 0 0 0  
## # ℹ 31 more variables: Months\_Inactive\_12\_mon <int>,  
## # Contacts\_Count\_12\_mon <int>, Credit\_Limit <int>, Total\_Revolving\_Bal <int>,  
## # Avg\_Open\_To\_Buy <int>, Total\_Amt\_Chng\_Q4\_Q1 <int>, Total\_Trans\_Amt <int>,  
## # Total\_Trans\_Ct <int>, Total\_Ct\_Chng\_Q4\_Q1 <int>,  
## # Avg\_Utilization\_Ratio <int>, Avg\_Trans <int>, Avg\_Mon\_Spend <int>,  
## # Trans\_Freq <int>, Credit\_Use\_Ratio <int>, Attrition\_Flag <int>,  
## # Gender\_M <int>, Education\_Level\_Graduate <int>, …

## Logistic Regression elasped tuning time approximately: 2.544603 seconds

## K-Nearest Neighbors elasped tuning time approximately: 31.38436 seconds

## Naive Bayes elasped tuning time approximately: 2.746739 seconds

## Random Forest elasped tuning time approximately: 1.89145 seconds

## XGBoost elasped tuning time approximately: 44.79925 seconds

We note some numerical issues with naive Bayes, already indicating that perhaps this model is not a good choice for the dataset.

## Detailed Findings and Evaluation

Here we provide an in-depth evaluation of models and description of analytical findings.

### Evaluate Tuned Models

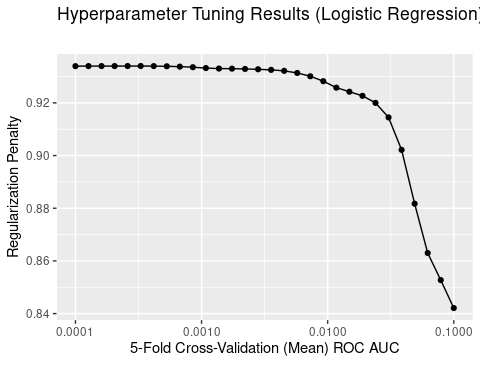
In this section, we look at the 5-fold CV estimates of the ROC-AUC for each classifier, highlighting the top 5 models (corresponding to the hyperparameter values searched in the CV tuning).

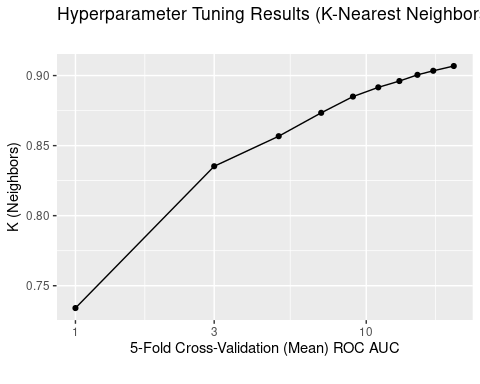
We then fit five models using the best hyperparameters obtained in the CV tuning step, obtain their predictions on the test data, and investigate them more deeply, plotting ROC and Calibration curves and calculating accuracy with respect to the model predictions.

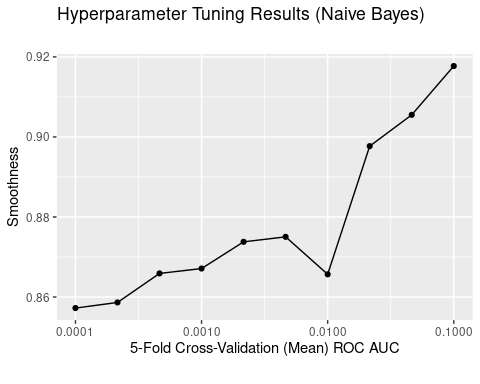
Finally, we use these results to select a top model.

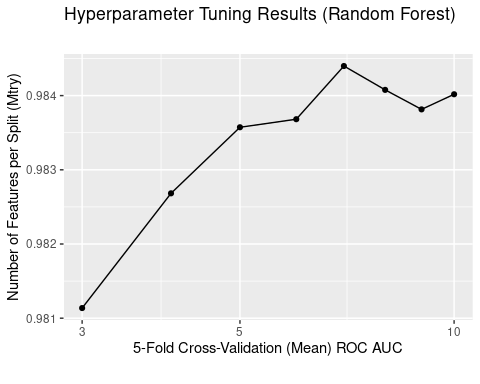
#### Tuning Results and CV ROC-AUC Estimates

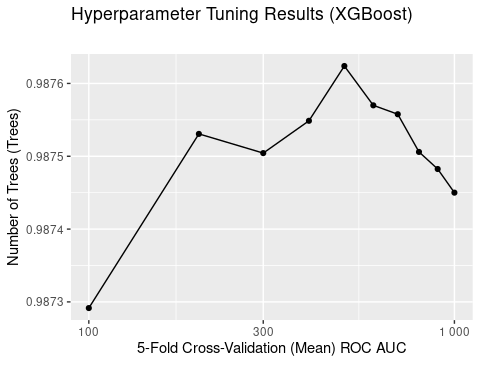
Here are plots of the hyperparameter values for each of the five baseline classification models, versus the mean ROC-AUC score over all 5 cross-validation folds.











Here we summarize the cross-validation tuning results, selecting the top five hyperparameters for each model, ranked by mean CV ROC-AUC, and sorting overall mean CV ROC-AUC in descending order.

## # A tibble: 25 × 5  
## Model Hyperparameter Hyperparameter\_Value Mean\_CV\_ROC\_AUC Std\_Err\_CV\_ROC\_AUC  
## <chr> <chr> <dbl> <dbl> <dbl>  
## 1 XGBoo… NumTrees 500 0.988 0.000652  
## 2 XGBoo… NumTrees 600 0.988 0.000650  
## 3 XGBoo… NumTrees 700 0.988 0.000658  
## 4 XGBoo… NumTrees 400 0.988 0.000706  
## 5 XGBoo… NumTrees 200 0.988 0.000657  
## 6 Rando… NumPredictors… 7 0.984 0.000596  
## 7 Rando… NumPredictors… 8 0.984 0.000661  
## 8 Rando… NumPredictors… 10 0.984 0.000733  
## 9 Rando… NumPredictors… 9 0.984 0.000779  
## 10 Rando… NumPredictors… 6 0.984 0.000873  
## # ℹ 15 more rows

From these results, there is a strong showing for the XGBoost gradient boosted machine model with all five top hyperparameter values having a better CV ROC-AUC then all other models, all just over 99%.

Random forest is a close second, again with all all five top hyperparameter values having a better CV ROC-AUC then the remaining models, all just under 99%.

Naive Bayes and logistic regression models took the next positions, with mixed ranking and accuracies between 92-94%.

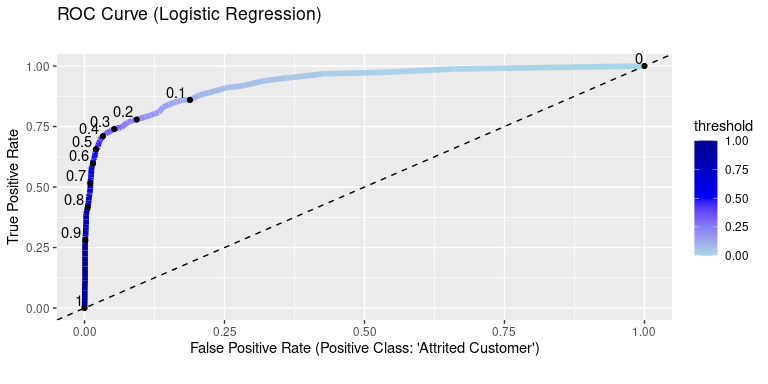
Finally K-nearest neighbors fared worst, with all five hyperparameters ranked lower than all other models, with CV ROC-AUC around 88-89%.

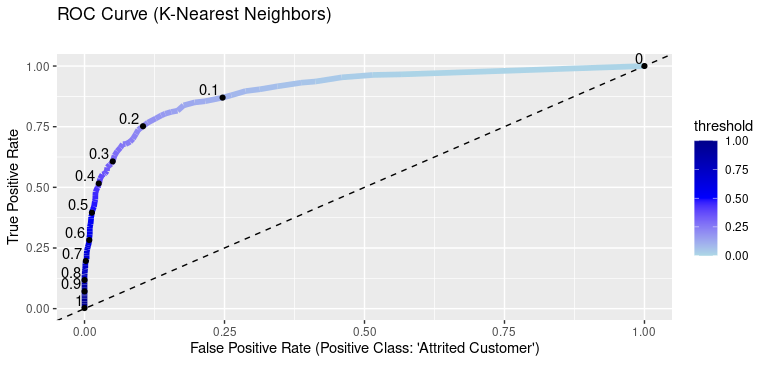
#### Null Model

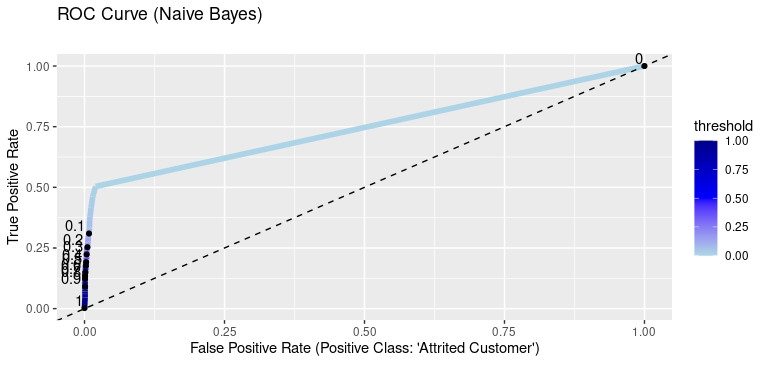
For this classification problem the classes are relatively unbalanced. For this reason, a good choice (and indeed, the standard choice) for the null model is the model which predicts the mode (most frequently occurring class), name "Existing Customer", in other words, predicting no customer attrition (this makes sense, as it is a relatively rare event).

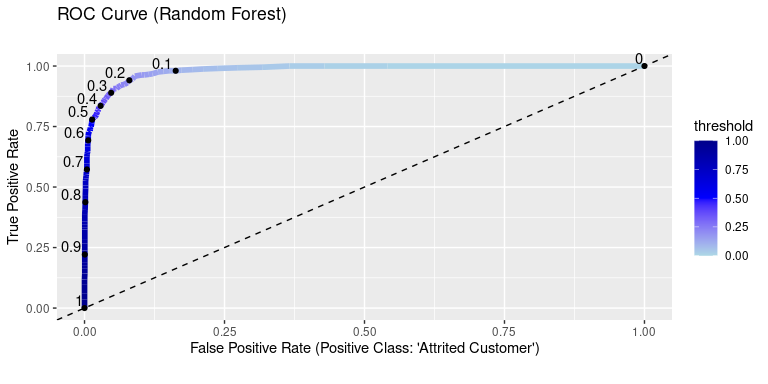
#### ROC Curves

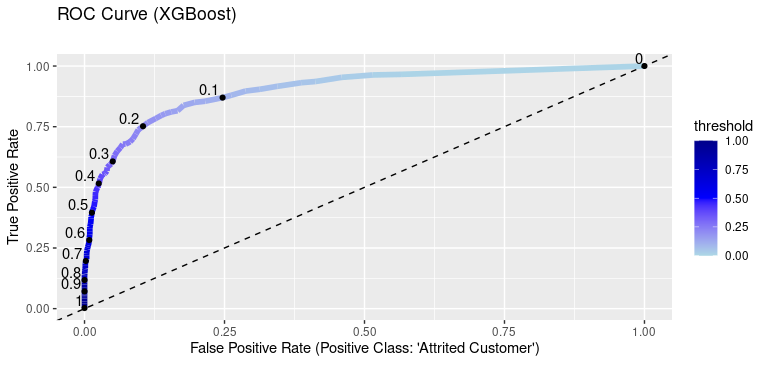
Here are the ROC curves for test data predictions of the five baseline models









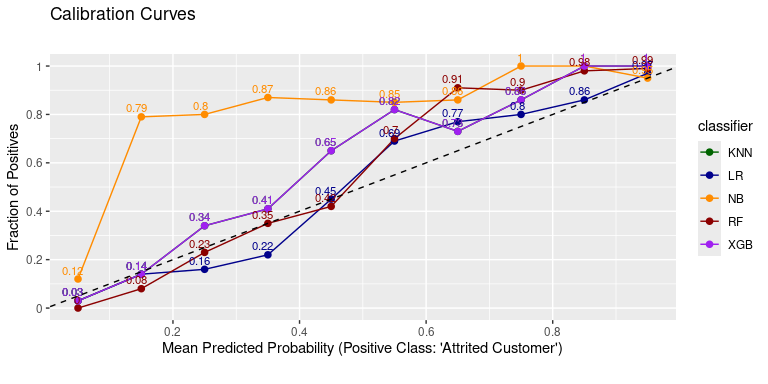


From these plots, we can see that the random forest classifier has the best ROC-AUC on the test data. This may be a bit expected, since the XGBoost mean ROC AUC found during cross-validation was highest.

This shows that random forest is much better than all other models at predicting the positive class at all probability thresholds, which indicates it is overall better suited to this data set as a predictive model.

#### Calibration Curves

Here are the calibration curves for test data predictions of the five baseline models

 The calibration curves provide strong evidence against the suitability of naive Bayes, which has a wild curve. The other models have more well-behaved calibration curves, with logistic regression and random forest clearly the best.

#### Accuracy

Here is the overall accuracy – i.e. the percentage of correctly classified customers – for the test data.

## # A tibble: 6 × 2  
## Model accuracy  
## <chr> <dbl>  
## 1 Random Forest 0.953  
## 2 Logistic Regression 0.928  
## 3 K-Nearest Neighbors 0.892  
## 4 XGBoost 0.892  
## 5 Naive Bayes 0.866  
## 6 Null Model 0.839

In terms of overall accuracy, random forest is the clear winner.

Note that the accuracy of the null model is relatively high, reflecting the imbalance of the classes, that is, the dominance of the existing customers in the data set. Note the proportion of the existing (non-attrited) customers in the data set is exactly the null model accuracy.

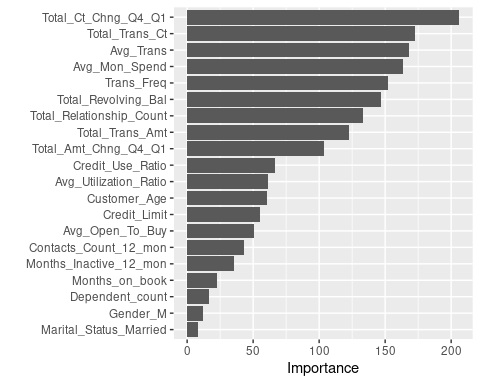
### Summary and Interpretation of Findings

The results of the last section indicate that the random forest model is the best choice, with best test ROC-AUC, a top two calibration curve, and best test accuracy, so we select this as the final model for interpretation.

#### Feature Importances

The first step for interpretation is the feature importances.

Note that, although feature engineering resulted in a total of 24 features, which is a low enough number to practically view all feature importances, the feature encoding, particularly the one-hot encoding, increased this number substantially, so it is not practically feasible all the importances.



A few observations and remarks:

1. Total amount and number of transactions were the most importance features, by a good margin, followed by total revolving balance, and total change in number of transactions from Q1 to Q4.
2. All four engineered features proved important, including transaction frequency, average amount per transaction, average monthly spending and credit use ratio.
3. Demographic variables were clearly less important than financial and transaction variables.

## Recommendations

***SIDEKICK: TO DO - Discuss***

### Future Work

Here are some suggested steps that could be taken to improve the predictive model.

1. Gather data on more customers
2. Add additional features which have predictive value (these could be tested against their relationship to churn before being added to the customer database)
3. Explore more thoroughly the relationship between features and the target, for example with bivariate plots and stastistical hypothesis tests.
4. Recode some of the categorical features with natural orderings as ordinal features, for example Education\_Level and Income\_Category.
5. Expand hyperparameter tuning to explore more of the search space or additional hyperparameters.

## Appendix [Optional]

Any additional details such as parts of the output you consider worthwhile but not appropriate for inclusion in the body of the report. It is important to make sure, however, that only outcomes that are referenced elsewhere in your report and/or are necessary to understand the details of the work are included - **inclusion of unrelated or unnecessary (to communicating or explaining of results) outcomes will adversely impact the final report grade.**

***SIDEKICK: TO DO - Discuss.***

***SIDEKICK: TO DO - Move technical discussion of feature engineering here, e.g. feature name, strategy taken (transformation or binning) and reason.***