Capstone Project 1
Predicting Customer Churn in the Telecom Industry

1 Overview

Client retention is one of the key drivers of profitability and longevity in many businesses. As such, it is a worthwhile endeavor to understand customer behavior and how it may be involved in helping companies better predict which customers are most likely to churn. This project performs a case study in churn by analyzing the customer data of a telecommunications company. This data includes information about what kinds of services the customer elected to sign up for, account information such as customer tenure, payment method and contract type, demographic information about the customer and of course a binary indicator for churn which indicates whether or not the customer left in the last month.

In order to perform this study, the project will pull from many strengths of a data scientist's toolbox including: data collection and wrangling, exploratory data analysis, applying machine learning models to find predictive value in the data and of course translating these insights and results for any reader or client to easily digest and interpret. In addition, while the goal of the project is to better predict which clients at a given moment in time are more likely to churn, survival analysis will also be performed and discussed, so that the client can have an idea of how likely a customer is to churn in the future.

2 Data Wrangling

2.1 Data Overview

The dataset consists of 21 columns and 7,044 rows of unique customer data. One of the columns was the customer's ID and thus contained no valuable information, so this column was dropped for data analysis and modeling purposes. Of the 20 remaining columns, one was the target variable, a binary value indicating whether or not the customer churned, and the other 19 customer attributes were a combination of 3 numerical columns and 16 categorical columns. In addition, the dataset provided is right censored, since tenure (how long a customer has been with the telecom company) is no longer recorded after 72 months; all clients who are still with the company after 72 months are therefore recorded as having not churned. The original dataset can be found here [1].

2.2 Data Importation & Cleaning

Data was imported into a pandas DataFrame after the CSV file was stored on the local machine used for this project. After the data was imported, standard steps were used to check for outliers, missing values and any other inconsistencies in our dataset. Missing values were denoted by an empty space. Since there were also spaces between words in some of our categorical variables and column names, the

appropriate blank spaces denoting missing values were converted to NaN values. After this, it was easy to identify that the dataset only contained 12 customer entries with NaN values, so these were simply excluded from the pandas DataFrame going forward. Upon excluding these rows, it was necessary to reset the index so that later on when indexes were referenced, all remaining indexes were valid. Additionally, appropriate type conversions were performed on almost all columns of the dataset (categorical, string, numeric).

3 Data Exploration

3.1 Exploratory Data Analysis (EDA)

With the dataset imported and properly formatted, exploratory data analysis is performed next to get familiar with the contents of the data. The purpose of this section is to explore some of the more important insights gained from the data analysis process. Histograms are a useful way to gain some insights into the composition of the telecom client's customer base. Figure 1 below shows the distribution of monthly charges, separated by what contract type they chose (month-to-month, one year or two year). A couple insights from this stand out. First, the size of the frequency bars of the month-to-month customers shows that this is the preferred contract type for all customers, across nearly all price points. In addition, there are quite a large number of month-to-month customers in the higher monthly charge range (~$70-100/month).

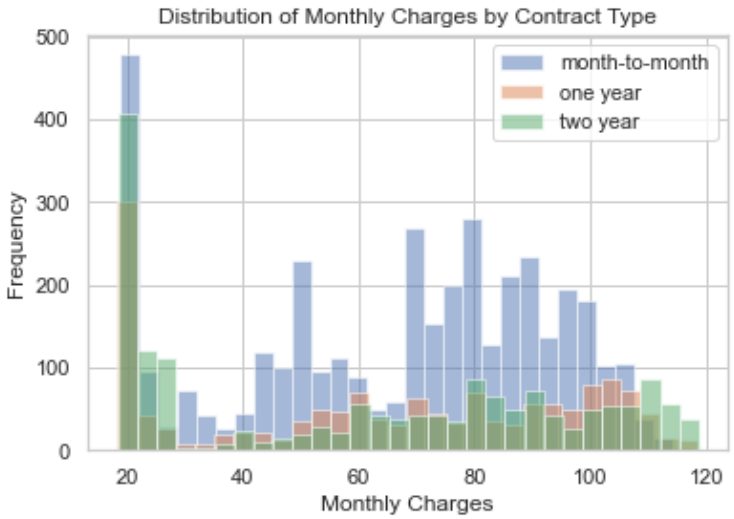

Figure 1: Histogram of monthly customer charges by contract type

To quantify and further visualize findings from Figure 1, it is useful to view normalized customer counts to compare the proportions of customers belonging to each contract type. Figure 2 shows that indeed over half of the customers chose a month-to-month contract (55.1%) and the remaining customers were closer to indifferent in choosing between one year contracts (24.0%) and two year contracts (20.9%).

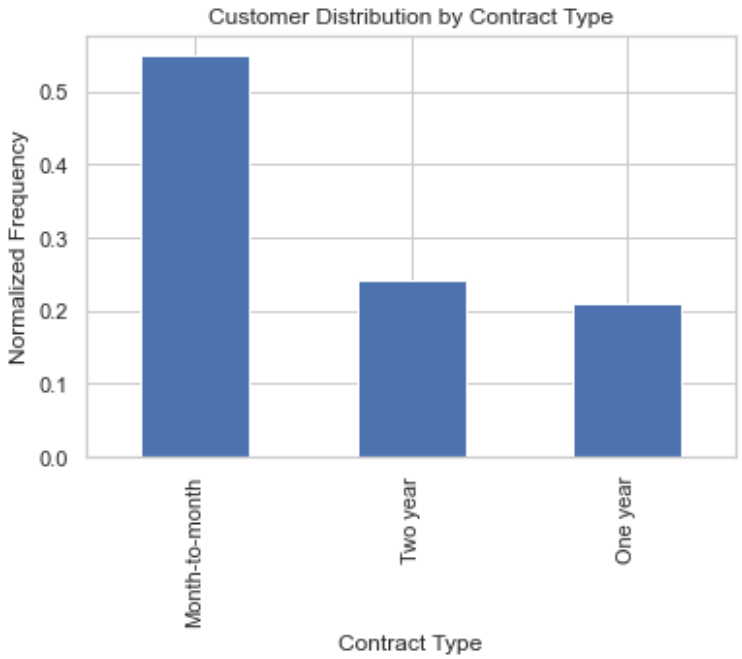

Figure 2: Normalized frequency bar graph of contract types

Throughout the EDA process, another variable of apparent importance is the payment method chosen by a customer. Figure 3 shows that the preferred payment method is an electronic check (33.6%) and the other 3 methods all have a similar percentage of customers choosing that method.

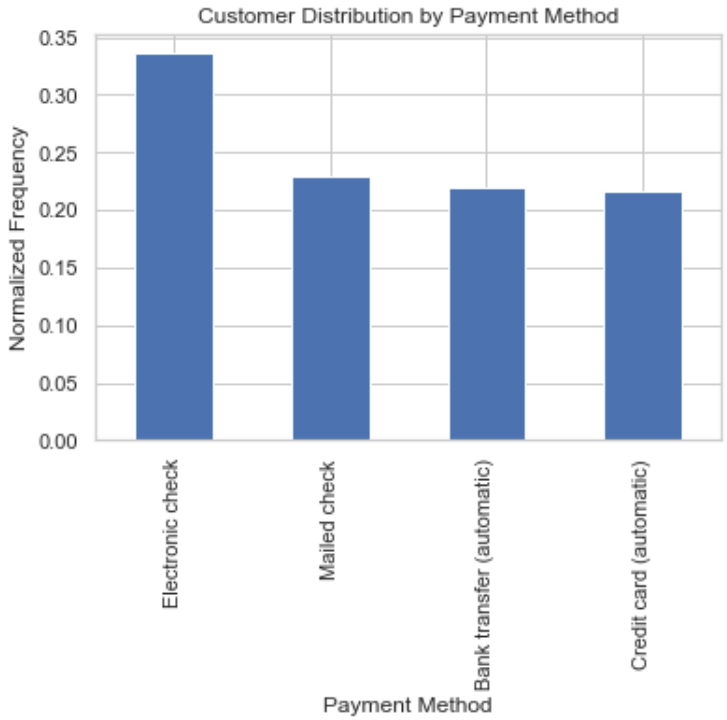

Figure 3: Normalized frequency bar graph of payment methods

After analyzing some of the basic distributions and frequencies in this dataset, the next step in the EDA process involves looking at some of these variables of interest and their effects on churn. Focusing again on contract type, Figure 4 shows that the month-to-month customers churn at by far the highest relative rate with almost half of this population churning (42.7%). Churn in the remaining contract type buckets is significantly lower with the one year customers churning at a rate of 11.3% and the two year contract customers churning at a rate of just 2.8%.

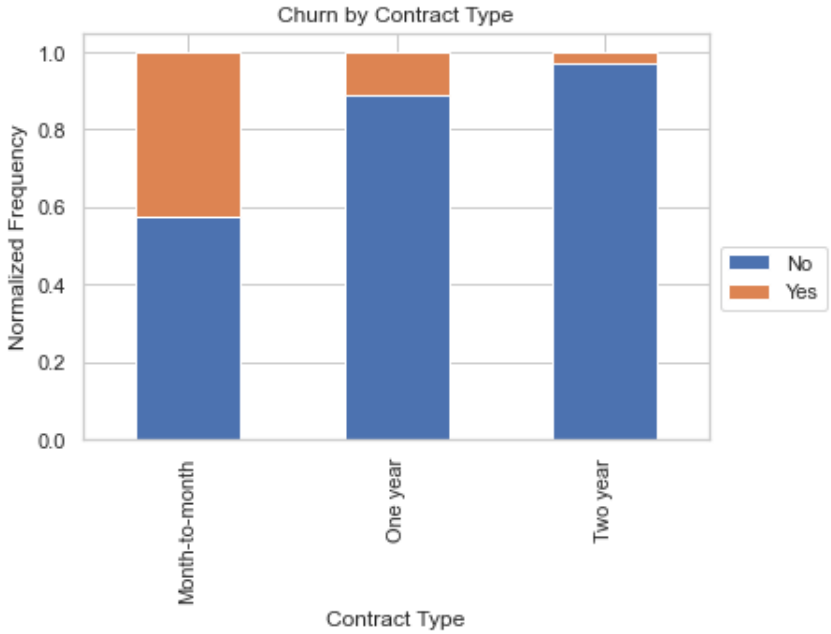

Figure 4: Stacked frequency bar graph of churn by contract type

Continuing to focus on the prediction variable, customer churn, Figure 5 again breaks down the dataset to examine churn by subcategories. When breaking down the population of customers that churn by payment method, electronic check is the standout payment method for highest relative churn at 45.3%. The other methods in order of descending churn were mailed check at 19.2%, bank transfer at 16.7%, and credit card at 15.2%.

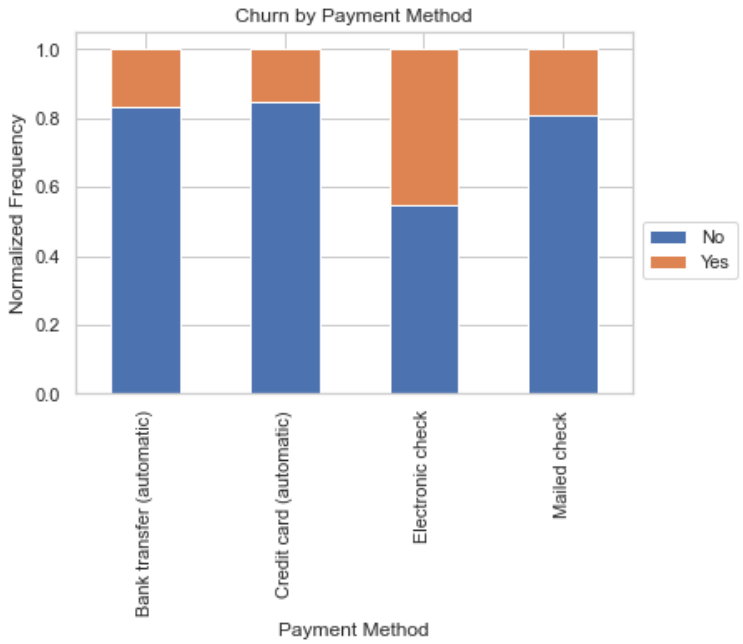

Figure 5: Stacked frequency bar graph of churn by payment method

The last portion of EDA for this case study examines a close counterpart to churn, survival. To conduct survival analysis, the Kaplan-Meier model is used to estimate survival over time; this model can be found within the PySurvival open-source python package. The Kaplan-Meier model offers two distinct benefits for this problem: it handles censored data (appropriate since our data stops after 72 months of tenure with the company) and it is non-parametric (requires no assumptions about underlying population) [2]. It is not uncommon for Kaplan-Meier survival curves to be plotted with an accompanying confidence interval (i.e. 95%), but for graphical and visual clarity, only the actual survival approximations are displayed in this section.

Figure 6 gives even more bad news for the telecom client as the month-to-month customer base survives at significantly lower rates across all tenures. It is also noteworthy that the survival rates of the one year contract customers start to exponentially decrease from about month 40 and beyond. The remaining bucket of two year contract customers is the highlight of the graph, as the survival rates across all tenures are quite high for this group.

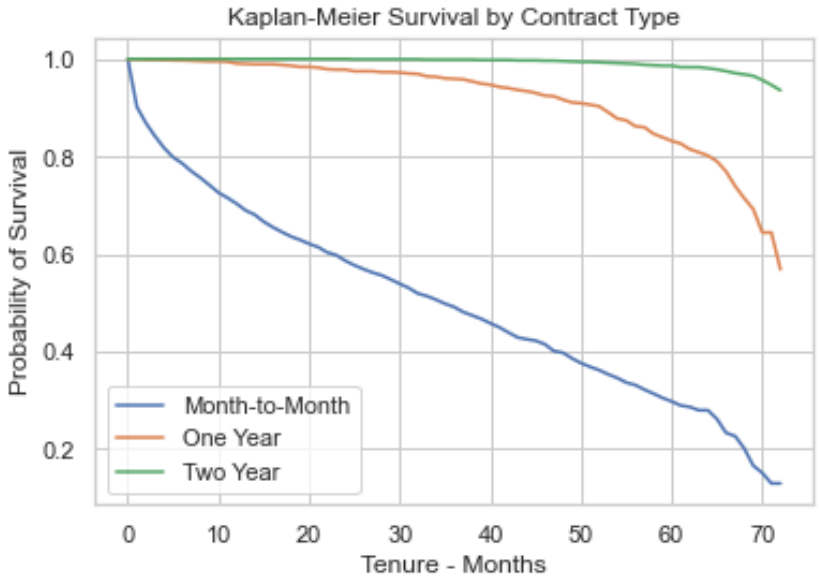

Figure 6: Kaplan-Meier survival curves by contract type

To complete the highlights of the EDA for this case study, Figure 7 looks at survival rates separated by payment method. Again, the insights from previous EDA show more bad news for the highest churn payment method, those opting to pay by electronic check. As was the case with the highest churn contract bucket, the electronic payment survival curve decays at a far higher rate than any of its other contenders. Mailed checks showed the second fastest drop off in survival over all time periods, but bank transfer and credit card survival rates were comparable and by far the best performers among the four payment methods.

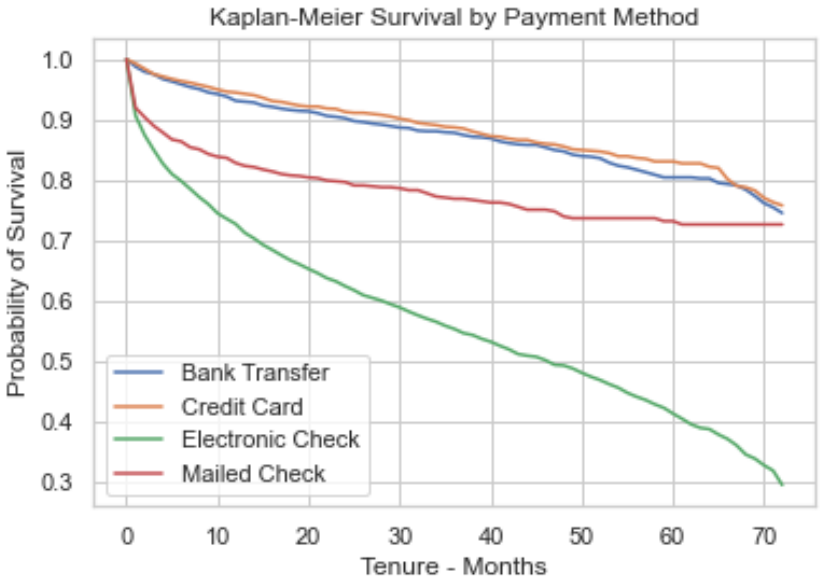

Figure 7: Kaplan-Meier survival curves by payment method

3.2 EDA Takeaways

Recapping the analysis in the EDA section and connecting the dots to tangible solutions, it's important to note that the majority of the customer base not only opts for month-to-month contracts, but also chooses their favorite payment method to be an electronic check. Unfortunately for the telecom company, those particular choices happen to be the highest churning option in their respective categories, which raises a few potential points of discussion for the stakeholders.

First is client sensitivity to payment method. It would be beneficial to weigh the pros and cons of removing this payment method altogether, as client sensitivity to payment method should not be that high if there are other reasonable methods for paying their bill. If there is indeed high sensitivity to different payment methods, there could be some kind of promotion offered to incentivize customer's preferences; perhaps something like $5 off of their bill per month for the first year or two on the notion that once they have been paying with that method, they are unlikely to want to switch when the promotion expires. Also, you could run some kind of A/B testing where when you onboard new clients, one cohort has the electronic check option and another doesn't and compare consumer sentiment around available payment methods.

The second big question up for discussion is what to do with the problem of the month-to-month customer base churning at such a high rate. Given industry trends and standards, it would certainly be worth discussing entirely removing the month-to-month contract option, or perhaps only allowing it after a two year contract has been completed. A cost-benefit analysis would certainly need to be performed here as it is imperative to ensure too many clients will not turn away from the telecom provider solely due to lack of a month-to-month option. In addition, there could be a marketing message along the lines of "we have removed our month-to-month option and lowered the price of our existing contracts in an effort to help us provide the highest quality service to our customers." With clever marketing, the shift could take on the narrative of generating value for the client and being in the best interest of both customers and the telecom provider.

The main point to draw from the EDA is that there is certainly an effort needed from the marketing team to find the best offering of payment methods and contract types so as to best incentivize the customer base towards options with lower historical churn rates. The discussion of these actionable plans and focus of marketing efforts will be continued upon conclusion of the in-depth analysis in the upcoming section on machine learning models. After a full analysis of the dataset, this case study can fully and properly consider actionable plans to present to stakeholders.

4 Machine Learning: In-Depth Analysis

4.1 Data Pre-Processing

Similar to getting data in the proper format before commencing EDA, there are some data pre-processing steps that must be taken before applying machine learning algorithms to the dataset. Since some data pre-processing steps vary by algorithm, an overview of all pre-processing steps will be discussed in this section. In later sections, model specific pre-processing steps will be addressed.

4.1.1 *Encode Labels*

Several of the algorithms used in the machine learning portion of this case study require categorical labels to be converted to integer representations of these categories. In order to accomplish this, the LabelEncoder method within the scikit-learn preprocessing library was utilized. LabelEncoder also contains methods that make it easy to convert the integer labels back to their corresponding labeled categories, so that model results can be easily interpreted.

4.1.2 *Split Data*

To ensure that the analysis of this case study adheres to machine learning best practices, an arbitrary split is used to divide the data into a training set and a test set. Specifically, 70% of the data will be used for training models and the remaining 30% will be used for testing the predictive power of the trained models. Additionally, a stratified split was specified so that approximately equal proportions of churned and non-churned customers remained in both the training and test sets.

4.1.3 *Cross-Validation*

While not necessarily a pre-processing step, the same cross-validation parameters were used for all models, so it is worthwhile to mention before discussing the specifics of each model. This study uses a 5-fold, stratified, cross-validation with 2 repeats. Again, stratified refers to the consistency of churned and non-churned customers being kept roughly equal across all folds. Cross-validation helps avoid over fitting the training set to the specific split generated by splitting the data by further dividing the training set into 5 folds. The 5 folds are then used in a manner such that each fold is held out for testing the remaining 4 folds on their ability to fit the data. Another way of stating the benefit of cross-validation is that it helps produce models that can generalize well to not only our test data, but also other unseen data in the future.

4.1.4 *Scaling*

Some of the machine learning algorithms used in this study are also sensitive to the values of the independent variables (i.e. variables with higher absolute values could have an unduly large influence on the model). For this reason,

some algorithms will acquire the additional pre-processing step of scaling the data. This study uses the StandardScaler method, again found in the scikit-learn preprocessing library, which scales variables so that all have mean 0 and standard deviation 1.

4.2 Modeling Pipeline and Hyperparameter Tuning

Once pre-processing steps have been performed on the dataset, there are some additional steps needed before models can be analyzed appropriately. First, the appropriate pipeline needs to be set up to feed into each algorithm. Common steps handled by the pipeline include imputing values for missing data points, (as a reminder, this step has already been taken care of in the data cleaning phase as there were only a handful of missing values and they were simply removed), scaling values as required, and specifying the type of model to be used. As mentioned in section 4.1.3, cross-validation will be performed to help increase the generalization power of models. One of the ways this is accomplished is via hyperparameter tuning. Hyperparameters vary by model and tuning them across all folds in the cross-validation process helps prevent tuning the hyparameters to just one split of the data. Fitting the hyperparameters to different splits of the data gives more confidence that model parameters and thus results can be generalized to test data as well as unseen data. Since the data was not heavily imbalanced in the case study, hyperparameters were optimized based on scoring accuracy of predictions. In practice, different scoring metrics can be used based on client specifications on how they intend to use the model, but for this study, optimizing on accuracy was appropriate.

4.3 Models

4.3.1 *k-Nearest Neighbors (kNN)*

The most common hyperparameter to tune in a kNN model is the number of nearest neighbors to be used in deciding where the classification boundary should be placed. This study optimized hyperparameters for the number of neighbors, the weights to be given to each neighbor (either uniform or by distance) and the definition of distance to be used. After using GridSearchCV to tune hyperparameters based on accuracy, the optimal parameters were determined to be:

- number of neighbors: 19
- weighting scheme: uniform
- formula for distance: Manhattan

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
kNN	0	0.81	0.89	0.85	0.79	0.57	0.77
	1	0.59	0.42	0.49			

Table 1: kNN summary statistics

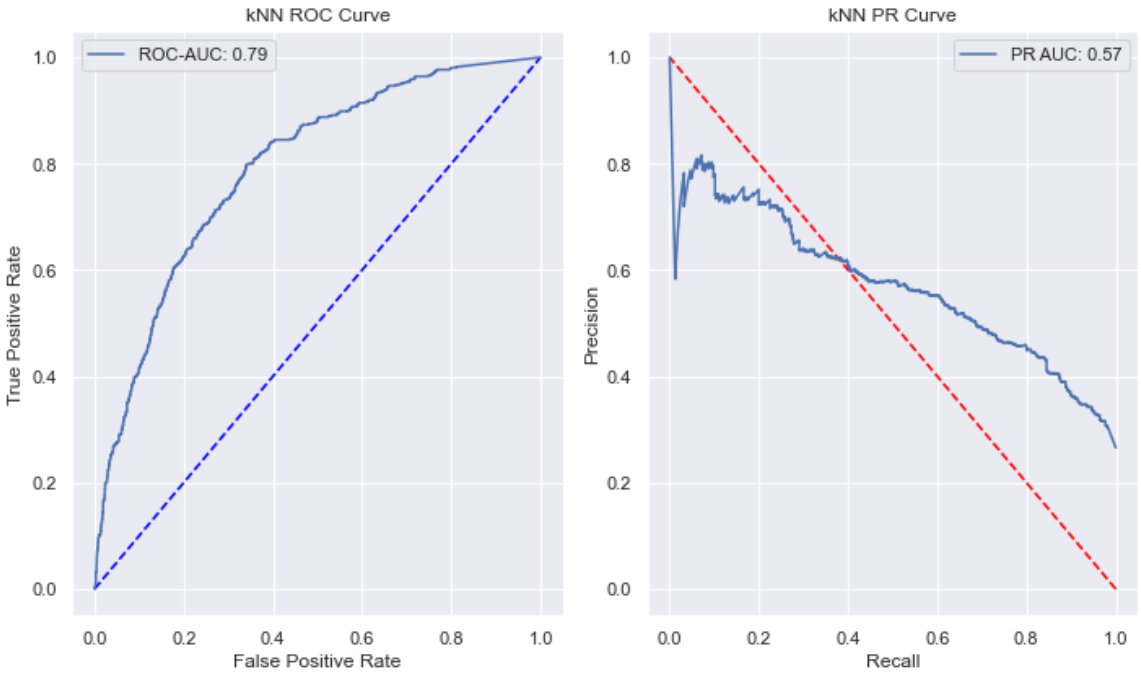

Figure 8: knn ROC (left), kNN PR Cuve (right)

From Table 1 above, the model has 77% accuracy, but most of the predictive power comes from the negative class (class 0, or those customers who did not churn) and struggles more with the positive class of customers who did churn.

The Receiver Operating Curve (ROC), depicted on the left-hand side in Figure 8, shows how the true positive rate (TPR) and false positive rate (FPR) change as the threshold for classification moves between [0,1]. As is the case with most classification problems, the threshold is assumed to be 0.5 unless specified otherwise. Focusing on the ROC curve, a perfect prediction model would be two perpendicular lines that intersect in the upper left corner and have an ROC area under the curve (AUC) of 1.0 as this would represent a TPR of 100% and a corresponding FPR of 0%. The blue dotted line has an AUC of 0.5 and signifies what the ROC curve would look like if all classifications were left to chance; anything above this line means the model is performing better than classifying based on chance.

The Precision Recall (PR) curve, displayed on the right-hand side in Figure 8, is another graphic for evaluating model performance but instead shows the relationship between precision and recall as the classification threshold varies between [0,1]. The perfect model for this graph would be two perpendicular lines, but this time intersecting at the upper right hand corner, which would signify 100% for both precision and recall. Precision quantifies the number of correct predictions made and recall quantifies the number of correct positive predictions. Since the precision recall curve focuses on the positive class (and can be structured more generally to focus on the minority class), it is useful for comparing binary classification models with imbalanced classes [3]. Since the

population of customers is imbalanced (73.4% churn, 26.6% do not churn), although not terribly so, it makes PR AUC a metric worth noting. AUC for both the ROC and PR curves is a way to boil down the effectiveness of a model into one number and make comparisons between models. The F1-score just represents the harmonic mean of precision and recall and is a metric to summarize those numbers into one for each class. Overall, the topline metrics of an ROC AUC score of 0.79, PR AUC score of 0.57 and accuracy score of 0.77 show that the model has some predictive power, but struggles with predicting customers who churn, so it stands as the first comparison point against future models.

4.3.2 *Logistic Regression*

Since logistic regression is designed to handle binary classification problems, it was a logical model to fit this dataset to. First, all data was normalized using the StandardScaler method in scikit-learn. In order to optimize the logistic regression algorithm, values for the hyperparameter C were tested using GridSearchCV. An optimal value of C = 0.1 was found.

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
Logistic Regression	0	0.84	0.89	0.87	0.83	0.61	0.80
	1	0.64	0.55	0.59			

Table 2: Logistic regression summary statistics

After tuning the model and scaling the data, the logistic regression model was fit and predictions on customer churn were generated. Compared to the kNN model, there were modest improvements across the board as the ROC AUC score improved from 0.79 to 0.83, the PR AUC score went from 0.57 to 0.61 and overall accuracy also saw an improvement, from 0.77 to 0.80. The class of interest, customer churn, also saw a noticeable improvement from the kNN, as indicated by the F1-score of 0.59 (compared to the F1-score of just 0.49 for the kNN model).

4.3.3 *Random Forest*

Random forests are one of the more robust and popular machine learning algorithms in use today and was a logical next choice in the model selection and evaluation process. Random forests build upon decision trees by not only introducing randomness into each tree via selection of a random subset of the feature variables, but also using a collection of many trees to generate predictions. While decision trees on their own are not great predictors, a collection of trees, or forest, significantly increases predictive power and lowers the test error of prediction.

Random forests have several hyperparameters that need to be tuned. Since there are several combinations of variables to be tried, computational time increases quickly. To help combat this a bit, RandomSearchCV was used initially, which

picks random combinations of specified parameter values to test and then reports back the optimal set of parameters based on tested values. After using RandomSearchCV to get in the correct general area of optimal hyperparameter values, GridSearchCV was then used for additional hyperparameter tuning closer to the values found from the random search. The final optimal set of hyperparameters for the random forest model were:

- number of trees: 200
- max features: 'sqrt' (square root of the length of the full feature set)
- max depth: 36
- min samples split: 4
- min samples leaf: 4
- bootstrap: True

"Min samples split" refers to the minimum number of data points placed in a node before it is split and "min samples leaf" refers to the minimum number of data points allowed in a leaf node. Bootstrap is simply the method for sampling data points with a value of true meaning that data points are sampled with replacement [4].

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
Random Forest	0	0.83	0.91	0.87	0.83	0.63	0.79
	1	0.65	0.49	0.56			

Table 3: Random forest summary statistics

While the random forest model did not show improvements across the board, there were some noteworthy takeaways. As the logistic regression model is the best seen so far, it serves as a useful model for comparison. The ROC AUC score was equivalent to that of the logistic regression model, but the PR AUC score of 0.63 was an improvement over the prior best score of 0.61. There was a small reduction in accuracy from 0.80 to 0.79, but given that the dataset is slightly imbalanced, the increase in the PR AUC score makes the random forest a contender for best model.

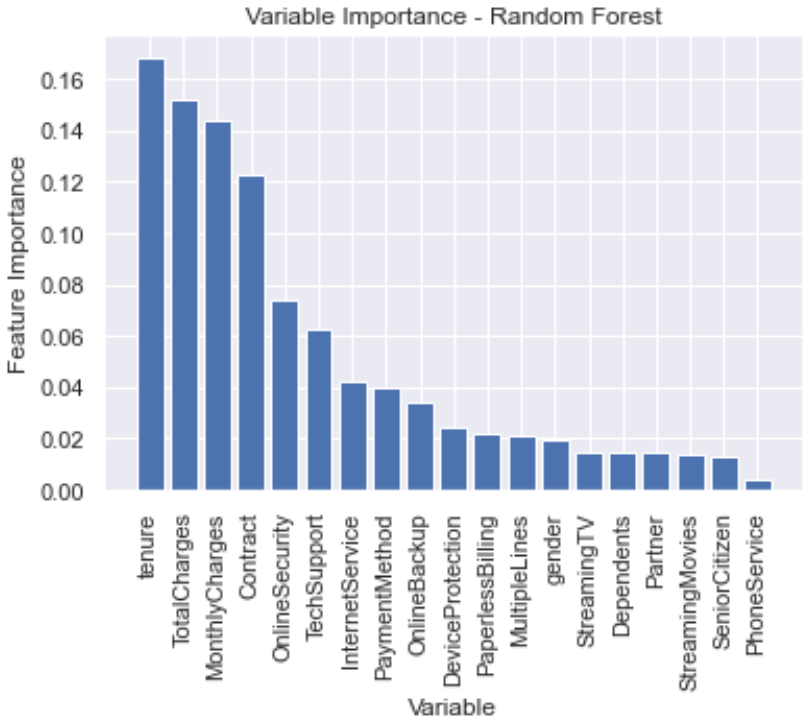

Figure 9: Random forest feature extraction

One other takeaway from the random forest is the ease with which feature importance can be analyzed. A model's predictive power can be thought of as how well of a job it does in explaining the variance of a dependent variable; in this case study, the variable of interest is again customer churn. From Figure 9 above, the 4 most important variables are tenure, total charges, monthly charges and contract type. Given the high correlation between total and monthly charges, one of these variables could simply be dropped for further dimensionality reduction in future models. Contract type shows up as the most important categorical variable, which supports the findings from the EDA section. Surprisingly, payment method comes in as much less important than some of the other variables, but could still serve as a useful cutoff for features to be included when considering dimensionality reduction models.

From Figure 9, it can be concluded that future analysis could be performed using a subset of features including: tenure, total or monthly charges, contract type, whether or not a customer had online security, if a customer used tech support, if a client opted to purchase internet service, and what their payment method was for their monthly bill. This would bring the example model to 7 features, from an initial feature space of 19 variables. To come up with optimal dimensionality reduction models, further analysis would need to be performed to quantify the predictive power sacrificed to achieve a lower dimension and computationally less expensive dataset.

4.3.4 *Support Vector Machine (SVM)*

Support vector machines are one of the most widely used algorithms for classification problems, which is the reason it is included in this case study. Since SVMs are not scale invariant, it is imperative to scale the data in the pre-processing stage. In addition, hyperparameters C and gamma are tuned, as these are two of the more important hyperparameters for establishing a well-tuned SVM model. Optimal hypertuned parameters are:

- C: 10
- gamma: 0.01

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
SVM	0	0.83	0.91	0.86	0.80	0.60	0.79
	1	0.64	0.47	0.54			

Table 4: SVM summary statistics

Unfortunately, the SVM model scores lower or equal to the best two models, logistic regression and the random forest. Overall prediction accuracy of 79% was on par with the other two models, but the ROC AUC score of 0.80 and the PR AUC score of 0.60 show that this model did struggle in a relative sense.

4.3.5 *AdaBoost*

The next model evaluated in this case study is a powerful ensemble method called AdaBoost, which is one of the most successful algorithms used for boosting. Similar to random forests, AdaBoost and other boosting methods get their roots from decision trees and seek to build upon the relatively weak predictive power of decision trees to produce better models. Boosting methods are usually composed of weaker trees because they have fewer nodes and leaves. However, they are notably different because they learn from each slice of the dataset, learning from the weaknesses of past iterations to make future iterations less prone to the largest areas of prior predictive weakness. AdaBoost does not require scaling the data, so the focus is on tuning two hyperparameters: the number of trees and the learning rate, which indicates how each individual tree contributes to the overall results [5]. GridSearchCV is used again for this process and yields optimal hyperparameters of:

- number of trees: 300
- learning rate: 0.1

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
AdaBoost	0	0.84	0.89	0.86	0.84	0.64	0.79
	1	0.64	0.51	0.57			

Table 5: AdaBoost summary statistics

Table 5 confirms that AdaBoost has produced the best model yet. With overall accuracy of 0.79, it is on par with the most accurate models, and boasts best scores for both ROC AUC and PR AUC at 0.84 and 0.64, respectively.

4.3.6 *Voting Classifier*

After tuning and analyzing 5 models, it is worth seeing if churn predictions can be better forecast by using another ensemble model, this time a voting classifier. With the VotingClassifier class in scikit-learn, many models can be combined into one to harness the forecasting powers of multiple models into one. There is a voting parameter that must be specified as "soft," since the models being fed into the voting classifier are well tuned. Since all 5 models being inputted have already been appropriately pre-processed, these steps are not needed for this algorithm.

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
Voting Classifier	0	0.82	0.91	0.87	0.83	0.63	0.79
	1	0.66	0.46	0.54			

Table 6: Voting classifier summary statistics

The results from this model are a little underwhelming, as it performs close to as well as the AdaBoost model, but does not show any improvements in top-line metrics and performs a bit poorer in the positive class (customers who churn); this is made evident when you take a look at the F1-scores for class 1 and see that AdaBoost has an F1-score of 0.57 and the voting classifier has an F1-score of 0.54.

4.4 Model Comparisons

Model	Class	Precision	Recall	F1-score	ROC AUC	PR AUC	Accuracy
kNN	0	0.81	0.89	0.85	0.79	0.57	0.77
	1	0.59	0.42	0.49			
Logistic Regression	0	0.84	0.89	0.87	0.83	0.61	0.80
	1	0.64	0.55	0.59			
Random Forest	0	0.83	0.91	0.87	0.83	0.63	0.79
	1	0.65	0.49	0.56			
SVM	0	0.83	0.91	0.86	0.80	0.60	0.79
	1	0.64	0.47	0.54			
AdaBoost	0	0.84	0.89	0.86	0.84	0.64	0.79
	1	0.64	0.51	0.57			
Voting Classifier	0	0.82	0.91	0.87	0.83	0.63	0.79
	1	0.66	0.46	0.54			

Table 7: All model summary statistics

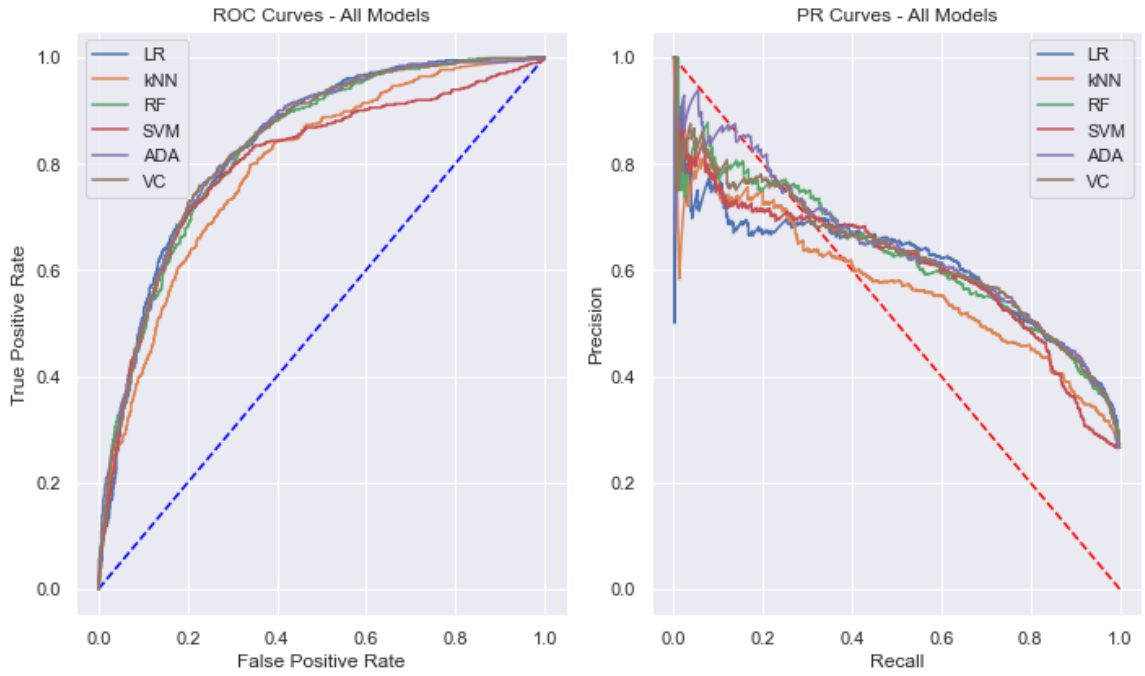

Figure 10: ROC and PR plots for all models

All model results are displayed in a single table upon completion of model analysis for this case study. While the AdaBoost model appears to be the best overall method with all factors considered, the logistic regression model could also be used for predictions in practice as it has high relative precision and recall scores for both classes, as is conveyed by the F1-scores.

Table 7 shows conclusions from the machine learning analysis more succinctly, but for visual comparison, Figure 10 shows the ROC (left) and PR (right) curves of all models graphed together. These graphs more or less show what was covered in the analysis sections on each of the individual models. From the ROC curves, it can be seen that the logistic regression, random forest, AdaBoost and voting classifier curves all do a similar job in predicting classification probabilities. On the right-hand side, it is a little bit clearer to see that the best model is the AdaBoost model, as it retains higher values of precision for low values of recall better than all other models.

5 Random Survival Forest (RSF) Analysis

Many prediction problems in the real world have censored datasets and this case study is an example of dealing with right censored data. Right censored in this case means that it is known when the customer joined the telecom company, but since the data only records up to 72 months of tenure with the company, it is not necessarily known when a customer will ultimately churn since after 72 months, all customers still with the company are simply marked as not having churned. Since it is also of interest to not only be able to give a prediction about how likely a customer is to churn today, but also how likely they will be to churn in the future,

this study concludes by taking a look at insights offered from a Random Survival Forest (RSF), which is an extension of the Random Forest model [5]. Since the random forest model from before had already hypertuned parameters, applicable carryover parameters used those values. Specifically, parameter values used were:

- number of trees: 200
- max features: 'sqrt'
- max depth: 36
- min samples leaf: 4

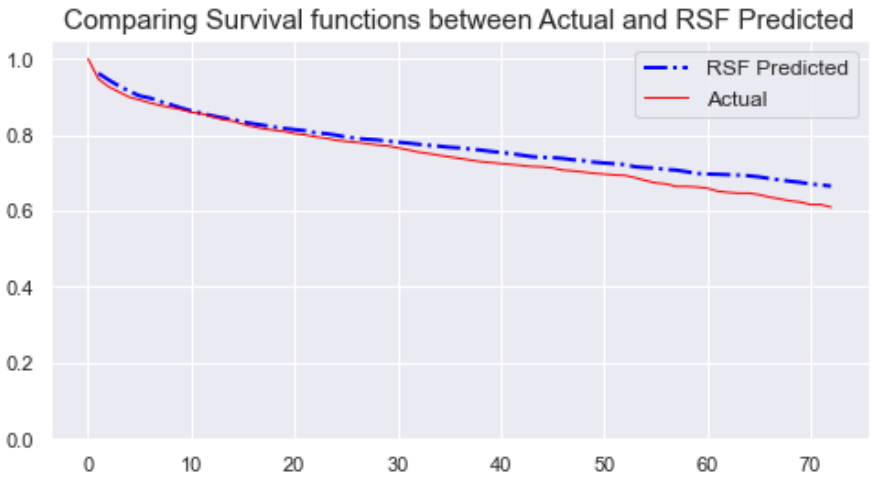

Figure 11: Actual vs. RSF Predicted Survival Curves
C-index: 0.88, Integrated Brier Score (IBS): 0.08

The plotted curves in Figure 11 show the RSF model does a very good job at predicting the survival curve. The concordance index (C-index above) is a generalization of the ROC AUC metric that takes into account censored data. Again, 1 is a perfect prediction and 0.5 is a completely random predicition, so the score of 0.88 confirms what is seen visually. The next metric for common evaluation of an RSF model is the integrated brier score (IBS), which can be thought of as a loss function[6][7]. Since the IBS looks at the squared difference from 1, a model predicting by chance would get a score of 0.25, so the closer to 0 the IBS score is, the better. Again, the IBS score of 0.08 for this model supports the finding that this model does a good job in predicting customer churn over time.

6 Future Work

6.1 Dimensionality Reduction

During the discussion surrounding the random forest model, it was mentioned that certain features that explained more of the variance in customer churn could be selected with the purpose of reducing dimensionality, and ultimately, computation time. Future models could use feature selection to optimize between a smaller feature set and lower computation time. One example of this would be the 7 feature model mentioned at the end of the analysis about random forests.

6.2 Exploration of Alternative Survival Models

The last section of this case study explored a potential use case for the client, which was using a random survival forest model to predict survival curves of clients. While this was useful, and the model was quite effective, there are other models that could have been used in addition that may have yielded better results (such as a conditional survival forest or extra survival trees).

6.3 Handle Class Imbalance

While the class imbalance in this dataset was not overwhelming, and was therefore not addressed in this study, predictive models still may have benefitted by addressing this imbalance via upsampling, downsampling or Synthetic Minority Oversampling Technique (SMOTE).

6.4 Integrate Client Specifications

Machine learning models in this analysis optimized hyperparameters based on a scoring metric of overall predictive accuracy. While this was a reasonable choice for the scoring metric, client specific requests may have mandated the use of optimization based on ROC AUC, PR AUC or some other metric. Changing the scoring metric for optimization may or may not have produced better results, but could be more in line with the client's need for the analysis. Obviously since this dataset was procured online, there was no interaction with the client.

7 Conclusions

While the logistic regression model produced slightly better minority class results than the AdaBoost, and the voting classifier produced results similar to AdaBoost, the AdaBoost model was the best all-around model and therefore most likely to be used in practice, if just one model were selected. Since the model not only classifies a customer as churned or not churned, but also computes associated probabilities, a client risk profile could be developed. This scale may have ranks that range from low risk (below 20%) to high risk (80-100%). Coordination with the marketing team would be the next step to properly address allocating marketing dollars on how to best retain customers with higher risk of leaving the company. The sweet spot for targeting marketing towards customers might be towards those who are 60-80% to churn for example (perhaps it is too difficult to change the minds of those who are 80-100% to leave). This example is just to highlight a possible use case for how the marketing team could analyze customer churn with marketing tactics built around the new churn model.

8 References

[1]: https://www.kaggle.com/blastchar/telco-customer-churn
[2]: https://en.wikipedia.org/wiki/Kaplan–Meier_estimator
[3]: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification/
[4]: https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
[5]: https://educationalresearchtechniques.com/2019/01/02/adaboost-classification-in-python/
[6]: https://square.github.io/pysurvival/models/random_survival_forest.html
[7]: https://square.github.io/pysurvival/metrics/c_index.html
[8]: https://en.wikipedia.org/wiki/Brier_score