Customer Churn

Springboard Capstone 2 Milestone Report 2

Brock Nosbisch

10.16.2019

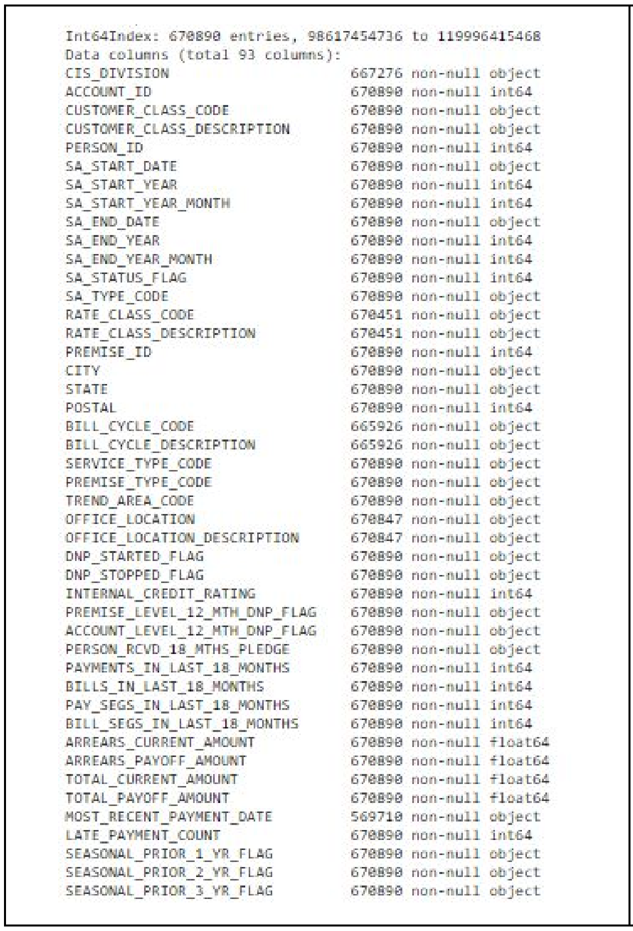
Problem Statement

My client is a natural gas company. Being able to predict when a customer will turn off is crucial to the natural gas business. By knowing when a customer will turn off, workload planning and field operations can forecast future turn off work and plan for enough resources to be available to do the required work. Also, knowing when a customer turns off enables salesreps to reach out to those customers to try to get them to stay on but if the customer does want to turn off still, the salesrep can schedule that customer to turn off during a non-peak time. This will save the company in overtime costs and ensure the workers are working at 100% utilization.

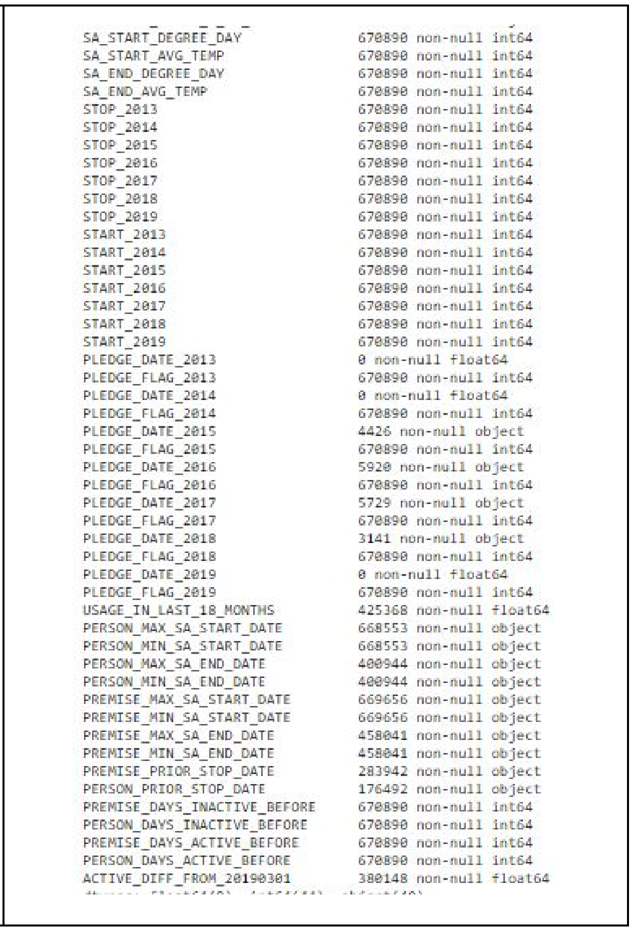
The Dataset

The dataset used in this project comes from both internal data and external data sources. The data will be merged inside our internal data warehouse and then the dataset will be manually downloaded and imported into the Data Wrangling notebook.

The original data is sourced from multiple applications and external sources. It is combined in a data warehouse and then exported with the below columns. The imported data set contains 670,000 records with 93 columns.



```
Int64Index: 670890 entries, 98617454736 to 119996415468
Data columns (total 93 columns):
CIS_DIVISION                       667276 non-null object
ACCOUNT_ID                         670890 non-null int64
CUSTOMER_CLASS_CODE                670890 non-null object
CUSTOMER_CLASS_DESCRIPTION         670890 non-null object
PERSON_ID                          670890 non-null int64
SA_START_DATE                      670890 non-null object
SA_START_YEAR                      670890 non-null int64
SA_START_YEAR_MONTH                670890 non-null int64
SA_END_DATE                        670890 non-null object
SA_END_YEAR                        670890 non-null int64
SA_END_YEAR_MONTH                  670890 non-null int64
SA_STATUS_FLAG                     670890 non-null int64
SA_TYPE_CODE                       670890 non-null object
RATE_CLASS_CODE                    670451 non-null object
RATE_CLASS_DESCRIPTION             670451 non-null object
PREMISE_ID                         670890 non-null int64
CITY                               670890 non-null object
STATE                              670890 non-null object
POSTAL                             670890 non-null int64
BILL_CYCLE_CODE                    665926 non-null object
BILL_CYCLE_DESCRIPTION             665926 non-null object
SERVICE_TYPE_CODE                  670890 non-null object
PREMISE_TYPE_CODE                  670890 non-null object
TREND_AREA_CODE                    670890 non-null object
OFFICE_LOCATION                    670847 non-null object
OFFICE_LOCATION_DESCRIPTION        670847 non-null object
DNP_STARTED_FLAG                   670890 non-null object
DNP_STOPPED_FLAG                   670890 non-null object
INTERNAL_CREDIT_RATING             670890 non-null int64
PREMISE_LEVEL_12_MTH_DNP_FLAG      670890 non-null object
ACCOUNT_LEVEL_12_MTH_DNP_FLAG      670890 non-null object
PERSON_RCVD_18_MTHS_PLEDGE         670890 non-null object
PAYMENTS_IN_LAST_18_MONTHS         670890 non-null int64
BILLS_IN_LAST_18_MONTHS            670890 non-null int64
PAY_SEGS_IN_LAST_18_MONTHS         670890 non-null int64
BILL_SEGS_IN_LAST_18_MONTHS        670890 non-null int64
ARREARS_CURRENT_AMOUNT             670890 non-null float64
ARREARS_PAYOFF_AMOUNT              670890 non-null float64
TOTAL_CURRENT_AMOUNT               670890 non-null float64
TOTAL_PAYOFF_AMOUNT                670890 non-null float64
MOST_RECENT_PAYMENT_DATE           569710 non-null object
LATE_PAYMENT_COUNT                 670890 non-null int64
SEASONAL_PRIOR_1_YR_FLAG           670890 non-null object
SEASONAL_PRIOR_2_YR_FLAG           670890 non-null object
SEASONAL_PRIOR_3_YR_FLAG           670890 non-null object
```



```
SA_START_DEGREE_DAY                670890 non-null int64
SA_START_AVG_TEMP                  670890 non-null int64
SA_END_DEGREE_DAY                  670890 non-null int64
SA_END_AVG_TEMP                    670890 non-null int64
STOP_2013                          670890 non-null int64
STOP_2014                          670890 non-null int64
STOP_2015                          670890 non-null int64
STOP_2016                          670890 non-null int64
STOP_2017                          670890 non-null int64
STOP_2018                          670890 non-null int64
STOP_2019                          670890 non-null int64
START_2013                         670890 non-null int64
START_2014                         670890 non-null int64
START_2015                         670890 non-null int64
START_2016                         670890 non-null int64
START_2017                         670890 non-null int64
START_2018                         670890 non-null int64
START_2019                         670890 non-null int64
PLEDGE_DATE_2013                   0 non-null float64
PLEDGE_FLAG_2013                   670890 non-null int64
PLEDGE_DATE_2014                   0 non-null float64
PLEDGE_FLAG_2014                   670890 non-null int64
PLEDGE_DATE_2015                   4426 non-null object
PLEDGE_FLAG_2015                   670890 non-null int64
PLEDGE_DATE_2016                   5920 non-null object
PLEDGE_FLAG_2016                   670890 non-null int64
PLEDGE_DATE_2017                   5729 non-null object
PLEDGE_FLAG_2017                   670890 non-null int64
PLEDGE_DATE_2018                   3141 non-null object
PLEDGE_FLAG_2018                   670890 non-null int64
PLEDGE_DATE_2019                   0 non-null float64
PLEDGE_FLAG_2019                   670890 non-null int64
USAGE_IN_LAST_18_MONTHS            425368 non-null float64
PERSON_MAX_SA_START_DATE           668553 non-null object
PERSON_MIN_SA_START_DATE           668553 non-null object
PERSON_MAX_SA_END_DATE             400944 non-null object
PERSON_MIN_SA_END_DATE             400944 non-null object
PREMISE_MAX_SA_START_DATE          669656 non-null object
PREMISE_MIN_SA_START_DATE          669656 non-null object
PREMISE_MAX_SA_END_DATE            458041 non-null object
PREMISE_MIN_SA_END_DATE            458041 non-null object
PREMISE_PRIOR_STOP_DATE            283942 non-null object
PERSON_PRIOR_STOP_DATE             176492 non-null object
PREMISE_DAYS_INACTIVE_BEFORE       670890 non-null int64
PERSON_DAYS_INACTIVE_BEFORE        670890 non-null int64
PREMISE_DAYS_ACTIVE_BEFORE         670890 non-null int64
PERSON_DAYS_ACTIVE_BEFORE          670890 non-null int64
ACTIVE_DIFF_FROM_20190301          380148 non-null float64
```

Data Wrangling

As mentioned, most of the data manipulation and joins were done in SQL prior to loading the data into the Python notebook.

The dataset contains Service Agreement (gas service) level data for random cities. The aggregated data is as of 3/1/2019 for this project's purpose. This will later be updated so that Python will dynamically generate the data being brought in. For now the data is generated from the “Queries - Used in Data Wrangling.sql” file in this repository. Some code was removed for masking and PII concerns. There are several temp tables created in the mentioned .sql file. At a high level, the data contains:

- Service Agreement level data
- Weather information
- Turn Ons
- Turn Offs
- Credit Scores

- DNP flags
- Number of pledges, payments, bills, late payments, and usage in the past 18 months
- Total money owed
- Prior 3 years of being a seasonal customer
- Prior premise and person level stops/starts.

The data will be read into the Python notebook through a csv (data.csv).

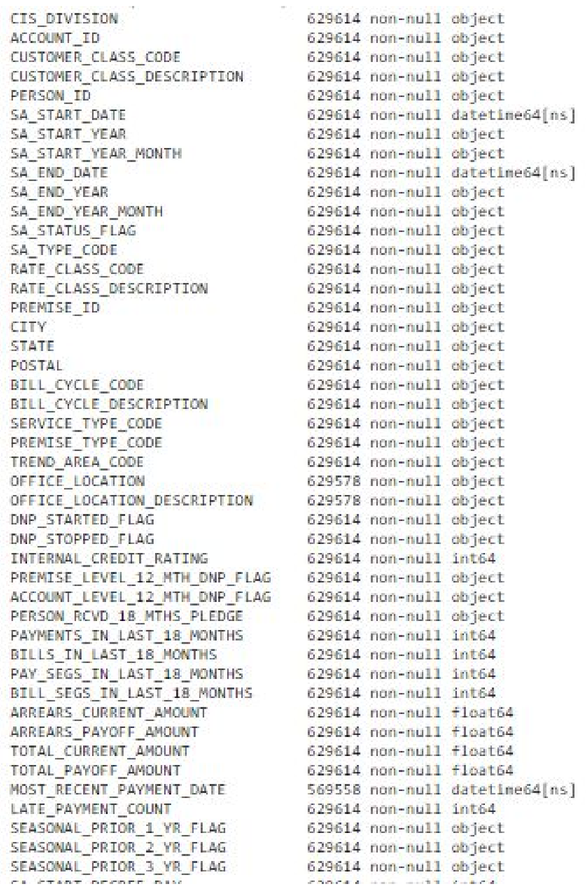
I will then set the index for the dataset to the unique identifier.

There are several fields that will need to have their data types updated and I will remove all Pending Starts, Reactivated, and Cancelled SAs since these should not be considered for Stops.

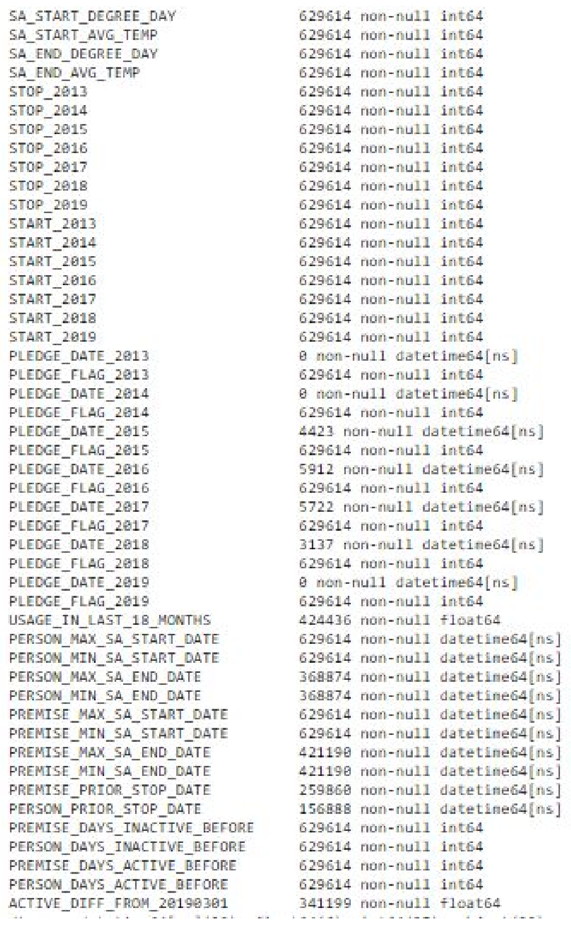
I will also remove a few bad records that have Null Rate Class, Company, and Bill Cycle.

Updated Dataset After Data Wrangling

For the most part, the columns are the same as what was read into the notebook. The big difference is in the data types and removal of some of the records.



```
CIS_DIVISION                       629614 non-null object
ACCOUNT_ID                         629614 non-null object
CUSTOMER_CLASS_CODE                629614 non-null object
CUSTOMER_CLASS_DESCRIPTION         629614 non-null object
PERSON_ID                          629614 non-null object
SA_START_DATE                      629614 non-null datetime64[ns]
SA_START_YEAR                      629614 non-null object
SA_START_YEAR_MONTH                629614 non-null object
SA_END_DATE                        629614 non-null datetime64[ns]
SA_END_YEAR                        629614 non-null object
SA_END_YEAR_MONTH                  629614 non-null object
SA_STATUS_FLAG                     629614 non-null object
SA_TYPE_CODE                       629614 non-null object
RATE_CLASS_CODE                    629614 non-null object
RATE_CLASS_DESCRIPTION             629614 non-null object
PREMISE_ID                         629614 non-null object
CITY                               629614 non-null object
STATE                              629614 non-null object
POSTAL                             629614 non-null object
BILL_CYCLE_CODE                    629614 non-null object
BILL_CYCLE_DESCRIPTION             629614 non-null object
SERVICE_TYPE_CODE                  629614 non-null object
PREMISE_TYPE_CODE                  629614 non-null object
TREND_AREA_CODE                    629614 non-null object
OFFICE_LOCATION                    629578 non-null object
OFFICE_LOCATION_DESCRIPTION        629578 non-null object
DNP_STARTED_FLAG                   629614 non-null object
DNP_STOPPED_FLAG                   629614 non-null object
INTERNAL_CREDIT_RATING             629614 non-null int64
PREMISE_LEVEL_12_MTH_DNP_FLAG      629614 non-null object
ACCOUNT_LEVEL_12_MTH_DNP_FLAG      629614 non-null object
PERSON_RCVD_18_MTHS_PLEDGE         629614 non-null object
PAYMENTS_IN_LAST_18_MONTHS         629614 non-null int64
BILLS_IN_LAST_18_MONTHS            629614 non-null int64
PAY_SEGS_IN_LAST_18_MONTHS         629614 non-null int64
BILL_SEGS_IN_LAST_18_MONTHS        629614 non-null int64
ARREARS_CURRENT_AMOUNT             629614 non-null float64
ARREARS_PAYOFF_AMOUNT              629614 non-null float64
TOTAL_CURRENT_AMOUNT               629614 non-null float64
TOTAL_PAYOFF_AMOUNT                629614 non-null float64
MOST_RECENT_PAYMENT_DATE           569558 non-null datetime64[ns]
LATE_PAYMENT_COUNT                 629614 non-null int64
SEASONAL_PRIOR_1_YR_FLAG           629614 non-null object
SEASONAL_PRIOR_2_YR_FLAG           629614 non-null object
SEASONAL_PRIOR_3_YR_FLAG           629614 non-null object
```



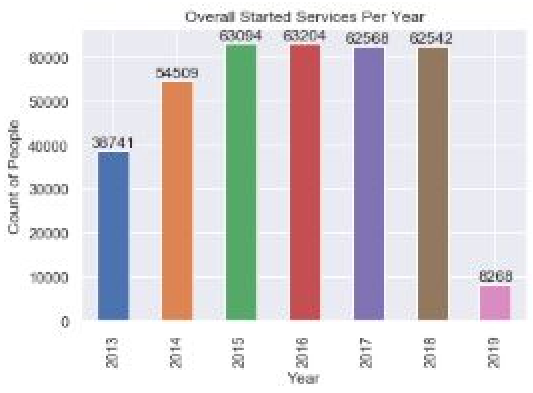
```
SA_START_DEGREE_DAY                629614 non-null int64
SA_START_AVG_TEMP                  629614 non-null int64
SA_END_DEGREE_DAY                  629614 non-null int64
SA_END_AVG_TEMP                    629614 non-null int64
STOP_2013                          629614 non-null int64
STOP_2014                          629614 non-null int64
STOP_2015                          629614 non-null int64
STOP_2016                          629614 non-null int64
STOP_2017                          629614 non-null int64
STOP_2018                          629614 non-null int64
STOP_2019                          629614 non-null int64
START_2013                         629614 non-null int64
START_2014                         629614 non-null int64
START_2015                         629614 non-null int64
START_2016                         629614 non-null int64
START_2017                         629614 non-null int64
START_2018                         629614 non-null int64
START_2019                         629614 non-null int64
PLEDGE_DATE_2013                   0 non-null datetime64[ns]
PLEDGE_FLAG_2013                   629614 non-null int64
PLEDGE_DATE_2014                   0 non-null datetime64[ns]
PLEDGE_FLAG_2014                   629614 non-null int64
PLEDGE_DATE_2015                   4423 non-null datetime64[ns]
PLEDGE_FLAG_2015                   629614 non-null int64
PLEDGE_DATE_2016                   5912 non-null datetime64[ns]
PLEDGE_FLAG_2016                   629614 non-null int64
PLEDGE_DATE_2017                   5722 non-null datetime64[ns]
PLEDGE_FLAG_2017                   629614 non-null int64
PLEDGE_DATE_2018                   3137 non-null datetime64[ns]
PLEDGE_FLAG_2018                   629614 non-null int64
PLEDGE_DATE_2019                   0 non-null datetime64[ns]
PLEDGE_FLAG_2019                   629614 non-null int64
USAGE_IN_LAST_18_MONTHS            424436 non-null float64
PERSON_MAX_SA_START_DATE           629614 non-null datetime64[ns]
PERSON_MIN_SA_START_DATE           629614 non-null datetime64[ns]
PERSON_MAX_SA_END_DATE             368874 non-null datetime64[ns]
PERSON_MIN_SA_END_DATE             368874 non-null datetime64[ns]
PREMISE_MAX_SA_START_DATE          629614 non-null datetime64[ns]
PREMISE_MIN_SA_START_DATE          629614 non-null datetime64[ns]
PREMISE_MAX_SA_END_DATE            421190 non-null datetime64[ns]
PREMISE_MIN_SA_END_DATE            421190 non-null datetime64[ns]
PREMISE_PRIOR_STOP_DATE            259860 non-null datetime64[ns]
PERSON_PRIOR_STOP_DATE             156888 non-null datetime64[ns]
PREMISE_DAYS_INACTIVE_BEFORE       629614 non-null int64
PERSON_DAYS_INACTIVE_BEFORE        629614 non-null int64
PREMISE_DAYS_ACTIVE_BEFORE         629614 non-null int64
PERSON_DAYS_ACTIVE_BEFORE          629614 non-null int64
ACTIVE_DIFF_FROM_20190301          341199 non-null float64
```

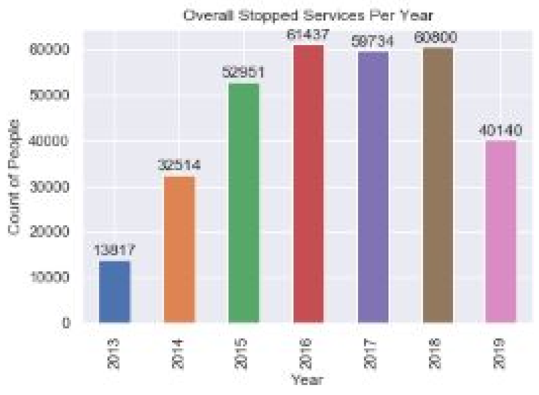
Field Definitions

- **CIS Division**: Company
- **Account ID**: An account is owned by 1 person but can have multiple Service Agreements.
- **Customer Class**: Account level customer class.
- **Person ID**: A person can have multiple accounts.
- **SA Start**: The Service Agreement's Start Date (when gas started).
- **SA End**: The Service Agreement's End Date (when gas stopped).
- **SA Status**: The current SA status.
- **SA Type**: The Service Agreement's type (example: Gas Residential).
- **Rate Class**: Determines how the SA is billed.
- **Premise ID**: Unique identifier for an address.
- **City**: Premise's City
- **State**: Premise's State
- **Postal**: Premise's Zip Code
- **Bill Cycle**: How we know when to bill an Account.
- **Service Type**: Always Gas.
- **Premise Type**: Premise Type (example: Commercial).
- **Office Location**: Closest office to premise.
- **DNP Started Flag**: Was the previous stop a DNP.
- **DNP Stopped Flag**: Did the current SA Stop because of DNP.
- **Internal Credit Rating**: The internal credit rating for an Account.
- **Premise Level 12 Month DNP Flag**: Has the Premise had a DNP in the last 12 months.
- **Account Level 12 Month DNP Flag**: Has the Account had a DNP in the last 12 months.
- **Person Received 18 Months Pledge**: Has the Person received a Pledge in the last 18 months.
- **Payments in Last 18 Months**: How many payments have been made for the SA in the last 18 months.
- **Bills in Last 18 Months**: How many bills were billed for the SA in the last 18 months.
- **Arrears Current/Payoff Amount**: How much money is past due for the SA.
- **Total Current/Payoff Amount**: How much money is owed for the SA.
- **Most Recent Payment Date**: What was the most recent Payment Date.
- **Late Payment Count**: How many late payments were charged to the SA in the past 18 months.
- **Seasonal Flags (1 year ago, 2 years ago, 3 years ago)**: Did the premise turn off between 3/1 and 6/30 and then turn on between 8/1 and 12/31.
- **SA Start/End Degree**: What was the Degree Days on the SA Start and SA End dates.
- **Stop/Start Flags**: Flags to show if the SA started/stopped in a given year.
- **Pledge Date/Flag**: The Pledge Date and Flag for each year.
- **Usage in Last 18 Months**: The Usage billed in the past 18 months.
- **Person/Premise Min/Max SA Start/End Dates**: What are the minimum and maximum start and end dates at a Person or Premise level.

- **Person/Premise Days Active/Inactive Before**: How many days was the previous SA for the Person or Premise Active or Inactive for.
- **Stop Days from 20190301**: Flags to use for training/testing. This should be dynamic but we are hardcoding the value for now.

Overall Started/Stopped Services Per Year

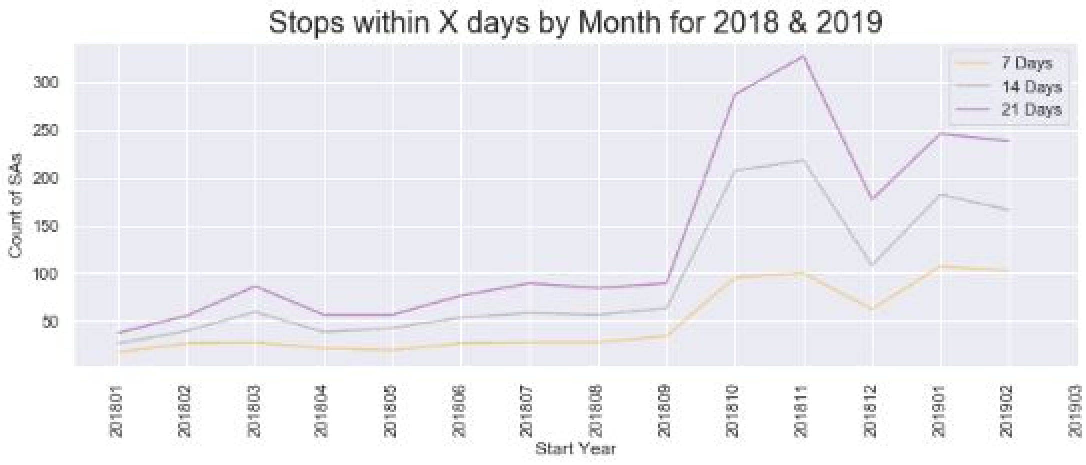




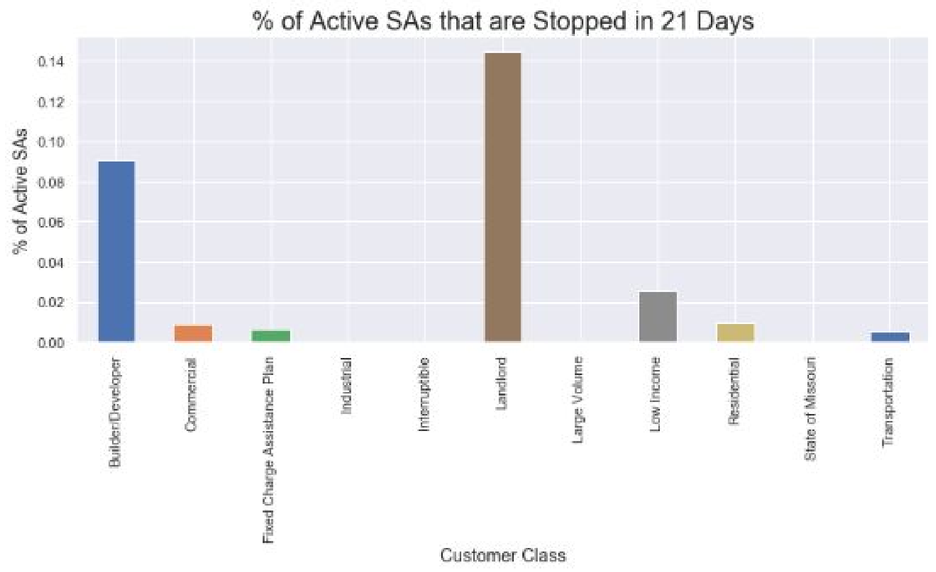
Exploratory Data Analysis

During EDA, several observations of interest were found.

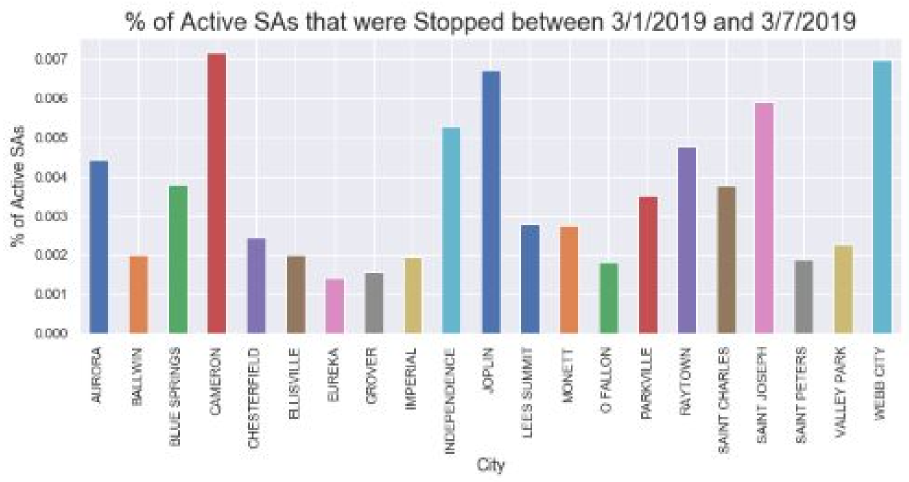
1) The most frequent Start Date of the SAs that were stopped in March had started gas service in October, November, and December of the previous year.

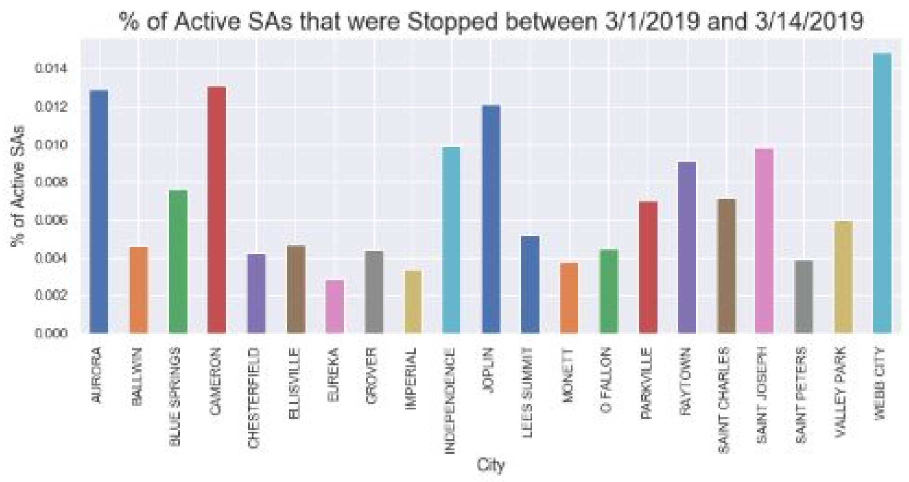


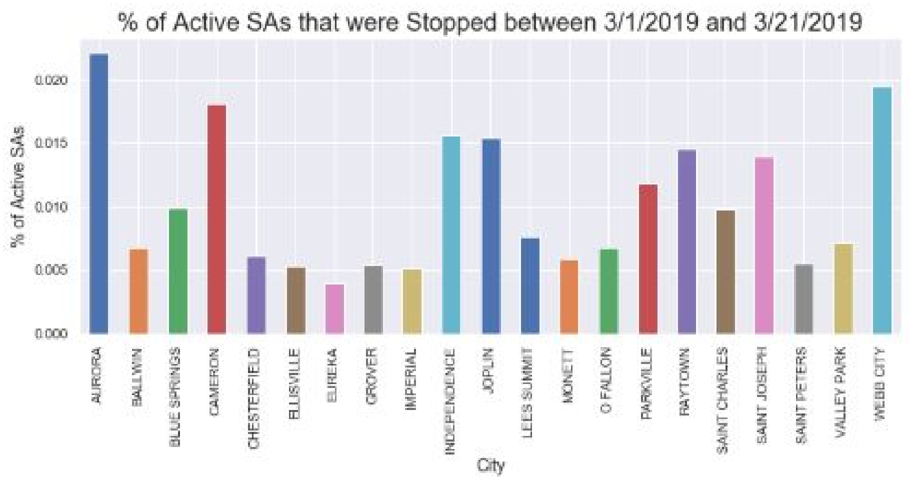
2) The Residential Customer Class has the most stops during all 3 of the time windows but less than 1% of all active Residential SAs were stopped. The biggest offenders of stops during the 3 windows are Landlords and Builder/Developers. 14% of Landlord SAs stopped during the 21 day window and 9% of Builder/Developer SAs stopped during the same window.



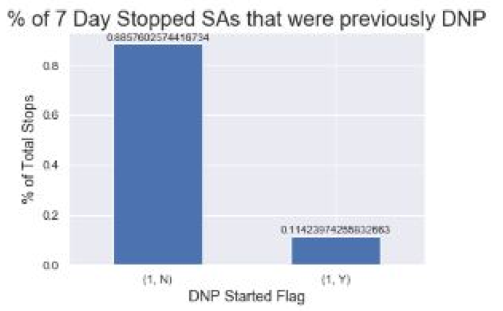
3) When looking at stops during the 7 day, 14 day, and 21 day windows by City, Webb City, Joplin, Cameron, and Aurora have the highest % of Active SAs that were stopped. Over 2% of all Active SAs in Aurora are stopped during the 21 day window.

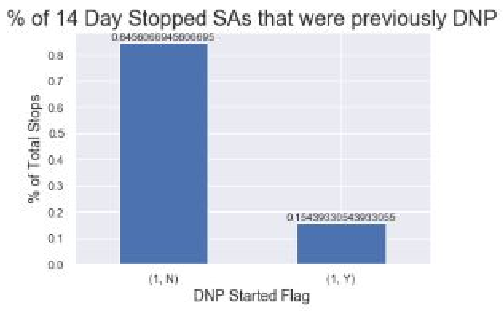


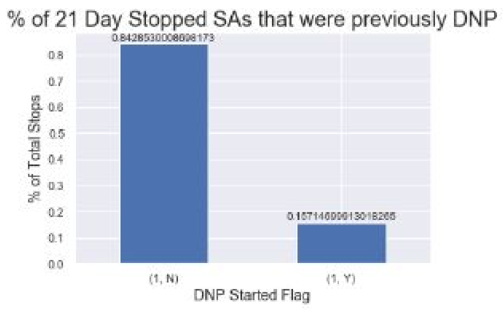




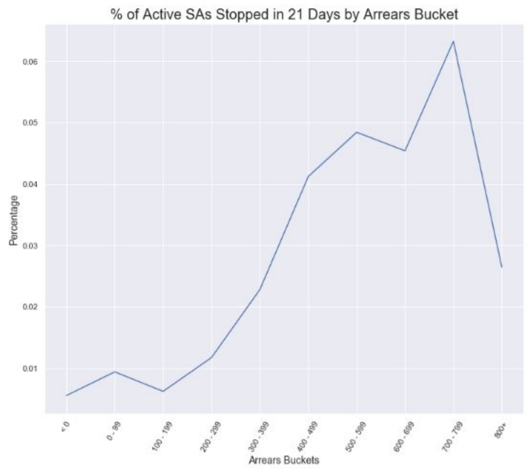
4) The percentage of stopped SAs that were DNPs previously during the 7 day window was 11%, 14 day window was 15%, and 21 day window was also 15%.

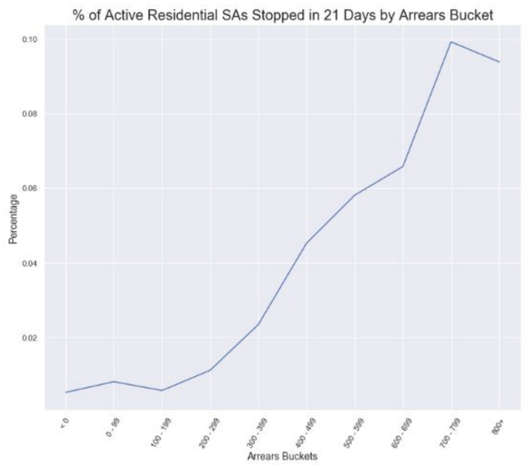




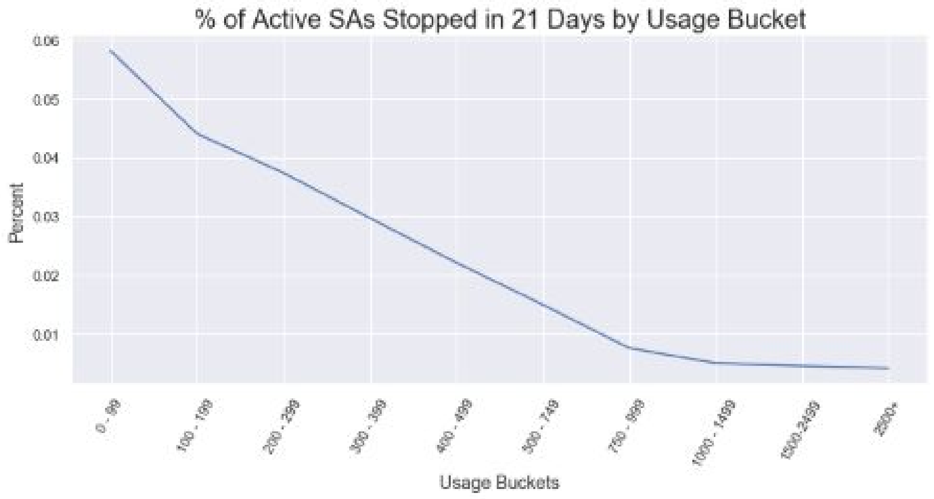


5) The higher the arrears amount, the more likely you are to stop. For example, if you owe 100 or less then you have less than a 1% chance of stopping between 3/1/19 - 3/21/19. If you owe 700−799 then you have over a 6% chance of stopping in the same timeframe. Residential customers have almost a 10% chance of stopping in the same bucket/timeframe.

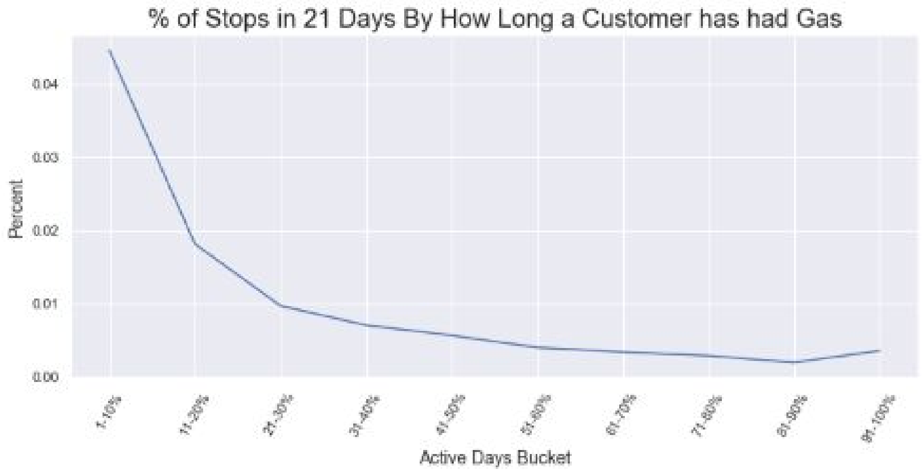




6) For Usage, it appears that the less you use, the higher chance you have of stopping service. Overall, if you have used < 100 therms then you have almost a 6% chance of stopping.



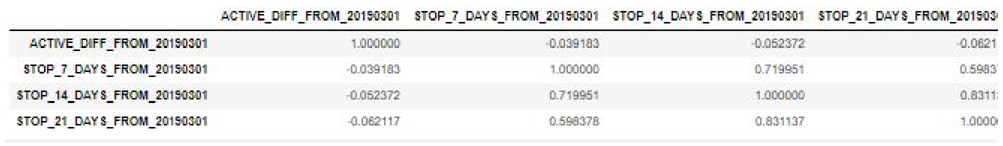
7) The shorter amount of time a customer has had gas service, the more likely the customer will stop gas.



Statistical Inference

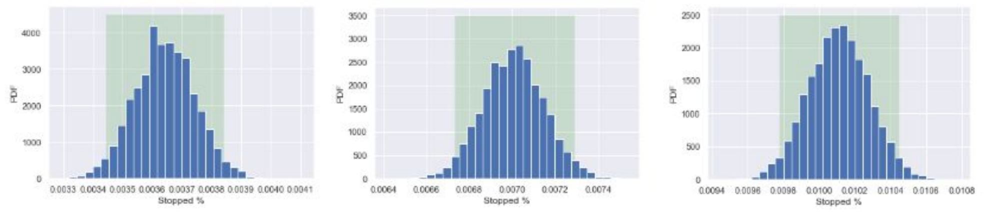
To confirm some of the above observations that may prove to be useful, I applied inferential statistics and found the correlations that are of statistical significance and a few that have practical significance. Here are a few observations:

- When comparing the Pearson correlation between Late Payment Counts and Stopping 21 Days from 3/1/2019, it has a p-value > .05 so the Null Hypothesis holds true. We can assume there is no significant correlation between these 2 values.
- # of Days Active as of 3/1/2019 has a negative correlation with Stopping. So in other words, the less days you have been active, the more likely you are to stop within 21 days of 3/1/2019.

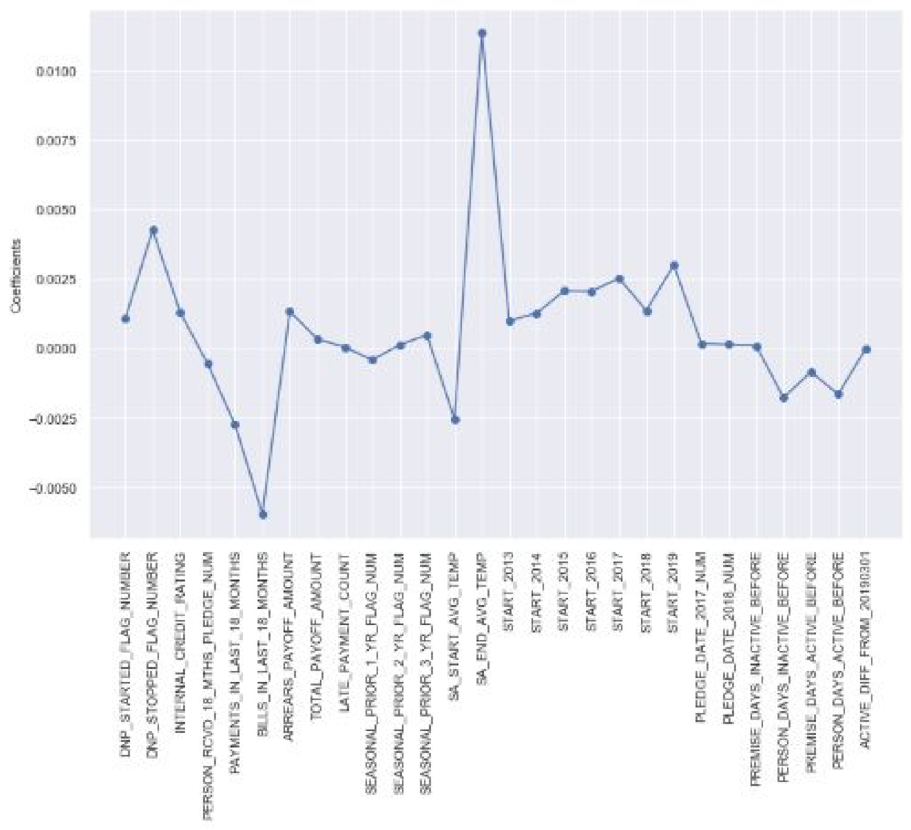


	ACTIVE_DIFF_FROM_20190301	STOP_7_DAYS_FROM_20190301	STOP_14_DAYS_FROM_20190301	STOP_21_DAYS_FROM_201903
ACTIVE_DIFF_FROM_20190301	1.000000	-0.039183	-0.052372	-0.0621
STOP_7_DAYS_FROM_20190301	-0.039183	1.000000	0.719951	0.5983
STOP_14_DAYS_FROM_20190301	-0.052372	0.719951	1.000000	0.8311
STOP_21_DAYS_FROM_20190301	-0.062117	0.598378	0.831137	1.0000

- The Bootstrap Estimate of SA Stops shows the 95% confidence interval for SAs stopping between 3/1/2017 and 3/7/2019 is from .34% to .38%.. Similarly, 3/1/2017 to 3/14/2019 is from .67% to .72% and 3/1/2019 to 3/21/2019 is from .98% to 1.04%.



- I used Lasso for Feature Selection. At a low value, the results show the Bills in last 18 months, DNP Stopped Flag, and SA End Avg Temperature as the top features.



Machine Learning

Custom Scorer

I created a custom scorer to show the amount of savings. For simplicity, I made the savings for scheduling a turn-off as 100 dollars and the cost of making a non-productive call as -10 dollars. This will account for cases of saving money for scheduling a turn-off (or a customer staying on) and also account for a CSR or salesreps' time they spent calling customers that were never going to turn-off.

Cost Factors:

- Scheduling a Turn-Off = $100 savings
- Time Spend Calling a non-Stop = -$10

Calculations:

- If the scorer correctly predicts a Stop then the savings is $100.
- If the scorer incorrectly labels a non-Stop as a Stop then the cost is -$10.

Machine Learning Algorithms

The original data brought in is for a handful of random cities. This was done to reduce the number of records brought into the original Data Wrangling notebook. From the Active SAs as of 3/1/2019, I upsampled stops to 20,000 and downsampled the Active SAs (non-stops) to 40,000. This was done to reduce training time. If I was doing this for the real thing and it was going to inform decisions then I would take the time to train the whole sample and use a server to make the training run in a reasonable amount of time. For the purpose of this capstone, I am using a smaller dataset so that it completes.

I ran the resampled dataset using Logistic Regression, k-NN, and Random Forest. I had originally included SVM but the results were subpar and the algorithm took a very long time to return results. The Random Forest algorithm was the best performing by far. The other algorithms were outputting negative savings. Do remember that the Savings values were picked at random by myself so the actual values a company chooses will be different.

Results for each model tested:

Algorithm	Savings
Logistic Regression	-53,360.00
Logistic Regression (Balanced Class Weight)	-292,720.00
k-NN (Best Neighbors, Uniform Weight)	-26,260.00
Random Forest	24,520.00
Random Forest (Max Features = 1)	19,850.00
Random Forest (Max Features = 2)	21,380.00
Random Forest (Max Features = 3)	23,970.00
Random Forest (Max Features = 4)	24,680.00
Random Forest (Max Features = 5)	24,520.00
Random Forest (Balanced, Max Features = 5, Max Depth = 22)	25,120.00

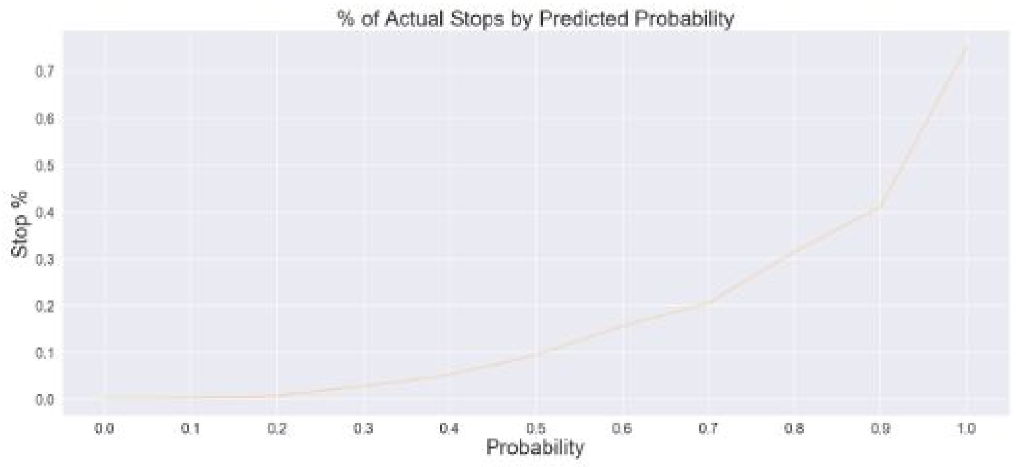
Since the Random Forest algorithm was the best performing, I chose the "Random Forest (Balanced, Max Features = 5, Max Depth = 22)" algorithm to dig into the details and see the individual probabilities. The Actual Stop % is good at the higher predicted probabilities but there are fewer customers at the higher predicted probabilities.

You can see that if the company wanted to keep the accuracy of calling people who will actually stop over a threshold (ie. 30%), the company can choose to only call those people with more than an 80% chance of stopping. The stats for the 80%+ chance of stopping predictions are a total of 376 predicted stops and 131 actual stops. The overall actuals stop % would be around 35%.

This is the breakdown of Predicted Stop Count, Actual Stop Count, and Predicted Stop % by Predicted Probability:

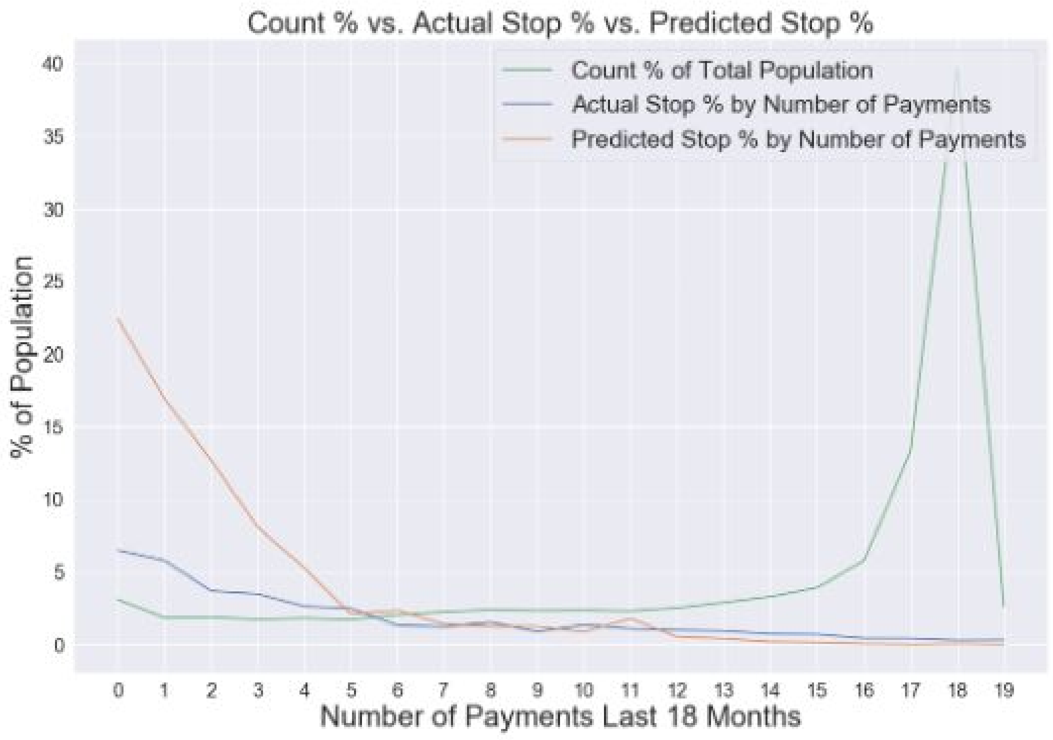
Predicted Probability	**Predicted Stop Count**	**Actual Stop Count**	**Predicted Stop %**
0%	0	66	0%
10%	0	367	0%
20%	0	199	0%
30%	0	138	0%
40%	0	121	0%
50%	611	127	20.8%
60%	895	140	15.6%
70%	556	114	20.5%
80%	273	86	31.5%
90%	95	39	41.1%
100%	8	6	75.0%

And here is a visual the % of Actual Stops by Predicted Probability. Like the percentages above show, the model is able to predict 75% of the stops that were given a 100% prediction.



Metrics by Subgroups (using Random Forest w/ Balanced Weight, 5 Features, 22 Depth)

For an analysis of metrics by subgroups, I am using the Random Forest with Balance Weight Class and 5 Max Features model since it was one of the best models. You can see from the below graph that the predictions are by far one of the best when compared to the other algorithms. The overpredictions are the highest when a customer has not made any payments but it seems to still be acceptable since the prediction is only 15% higher than the actual stop %.



Recommendations

Here is a list of my recommendations:

- The Random Forest algorithm is the best overall algorithm to use to get the most savings.
- Targeting the high turn-off probability customers is best since they have the greatest chance to be an actual turn-off.
- The most people turning off between 3/1/2019 and 3/21/2019 had gas started in October, November, and December 2018. These turn-offs are most likely financial in nature and the customers most likely only have a furnace that uses gas and nothing else.
- The less gas someone uses, the more likely they are to stop service. For example, if a customer has used < 100 therms then they have ~6% chance of stopping.
- The Residential Customer Class had the most stops during all 3 of the time windows but less than 1% of all active Residential SAs were stopped. The biggest offenders of stops during the 3 windows are Landlords and Builder/Developers. 14% of Landlord SAs are stopped during the 21 day window and 9% of Builder/Developer SAs are stopped during the same window.
- Having a good balance between correctly predicted stops and wrongly predicted stops is key. With the current scorer configuration, it takes 10 wrongly predicted stops to zero out a correctly predicted stop.

Conclusion

The Random Forest algorithm is the best model to use to calculate savings. The model may be somewhat biased against customers who haven't made as many payments but that bias is based on the fact that a higher percentage of people with less payments made stop gas service more often. In the end, the model does do a good job in being able to predict who will stop. As mentioned before, if a company only wants to call customers that have a 30% chance or more to stop service then the company can choose which customers to call based on the probabilities.

Possible Next Steps:

- If time permitted, I would add additional algorithms (ex. xgboost).
- The original dataset was reduced by only including certain cities. All cities should be included though.
- The amount of data was resampled so that less customers would be analyzed. This was done to reduce training time. If I was doing this for the real thing and it was going to inform decisions then I would take the time to train on all Active SAs and still upsample the Stops. I would use a server to make the training run in a reasonable amount of time.
- Most of the data manipulation was done in SQL against a Data Warehouse database. It would be ideal to be able to dynamically generate the SQL so that any date(s) can be used.
- The effect of weather should be explored more. Then I can pick up relationships more on that margin. The current models are only using the average temperature on the day the customer started service.