Predicting Customer Churn

Quick introduction

WHAT WE WILL COVER

1. Data Preparation and pre-prediction
2. Predictive Modeling
   a. Classification using Decision Tree
   b. Classification using Naive Bayes
   c. Classification using Random Forest
   d. Model comparison
3. Recommendation

WHO WE ARE

Group 4

- Patrick Codrington
- Stacey Jovcic
- Ben Polasek
- Kenneth Brandt

The problem we are solving

We have been tasked with assisting a phone company to characterize customer churn through data analytics methods

Executive **Summary**

The Random Forest algorithm showed the best performance correctly estimating 96% of churners and non-churners (accuracy) and 69% of actual churners were identified (recall)

Key Insights where that key predictors of churn included:

- High frequency interaction with customer (typically greater than 3 call)
- High total charges (typically customers with total charges > $80)
- Enrolled in the International Calling Plan
- Not enrolled in the Voicemail Plan
- Members with really high tenure churning more

Recommendation: Using the Random Forest model the phone company can apply a score to each customer and take proactive initiatives to reduce the number of churners.

1

Data Preparation and pre-prediction

Preparing the data

Before

21 attributes

Categorical

State, Area Code, Phone, Int'l Plan (binary), VMail Message (binary), Churn (class variable)

Numerical

Continuous: Account Length, Day Mins, Day Charge, Eve Mins, Eve Charge, Night Mins, Night Charge, Intl Mins, Intl Charge

Discrete: VMail Message, Day Calls, Eve Calls, Night Calls, Intl Calls, CustServ Calls

After

8 attributes

State

Int'l Plan

VMail Plan

VMail Message

Total Charge

Total Charge > 80

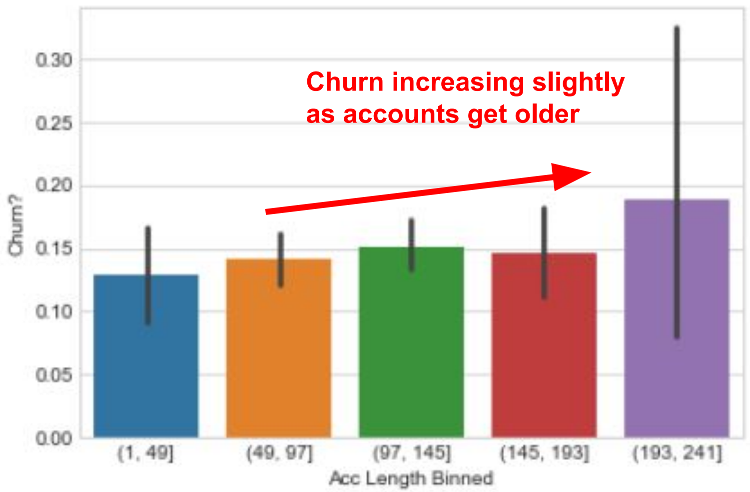
CustServ Calls >3

Churn (class variable)

Feature Selection

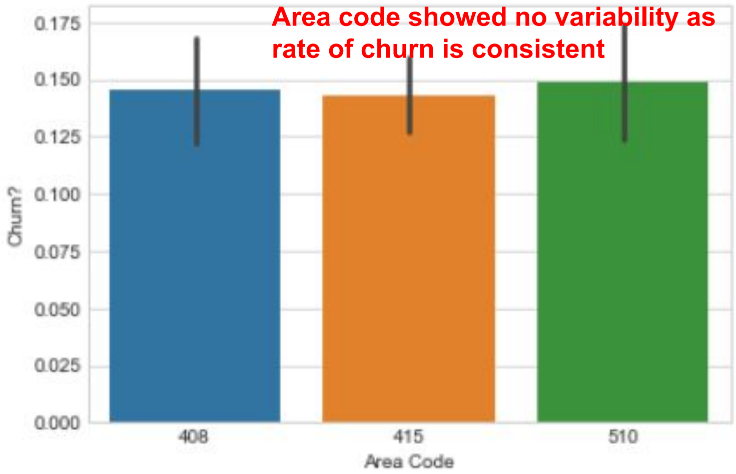
Features were selected after pivoting the attributes against the class variable. Below are plot of three examples of the effect each attribute was having on the class variable.

Churn x Acc Length Binned



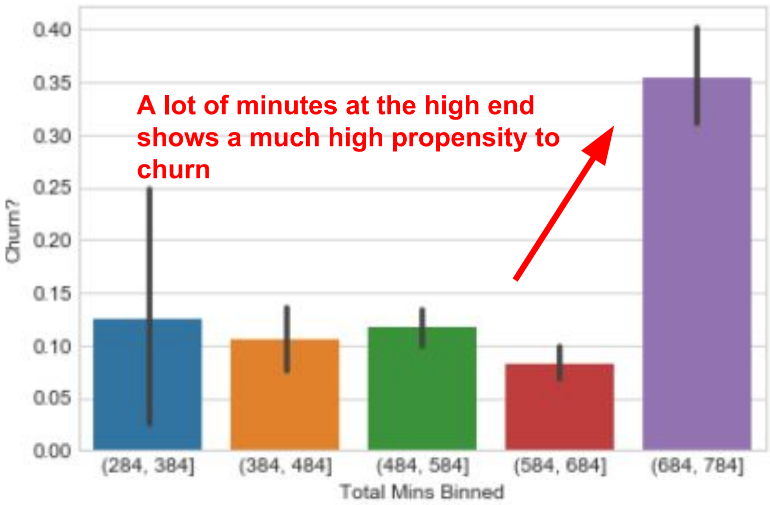
Feature used in modeling

Churn x Area Code



Feature removed

Churn x Total Min Binned

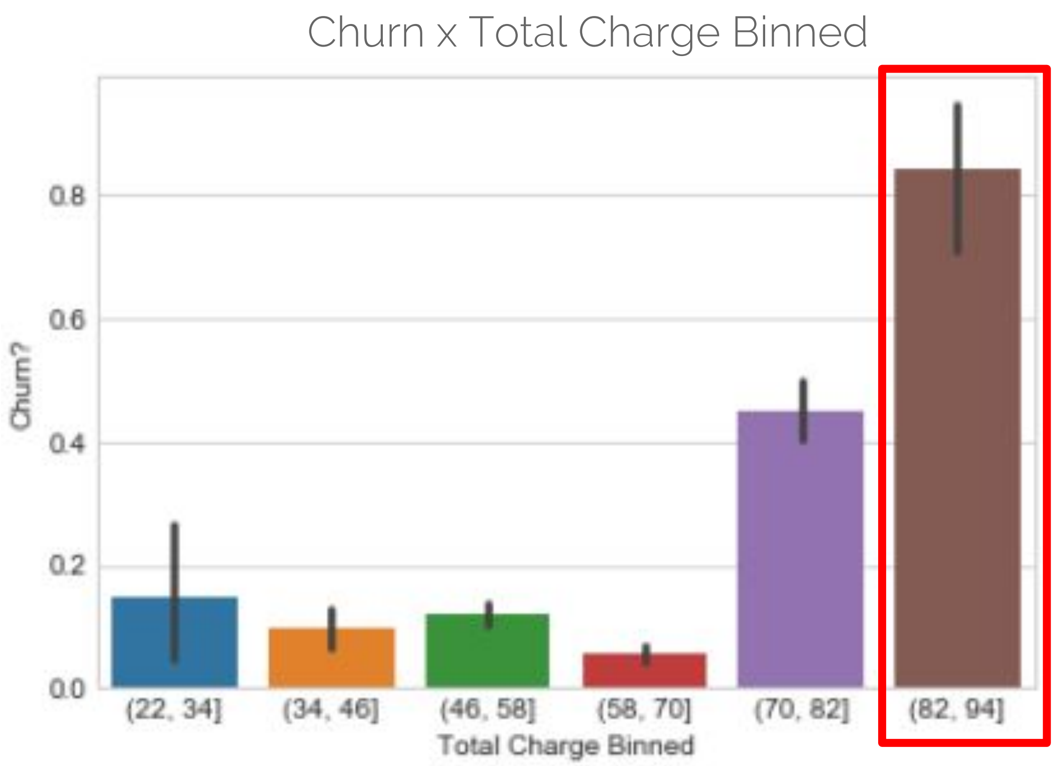


Feature used in modeling

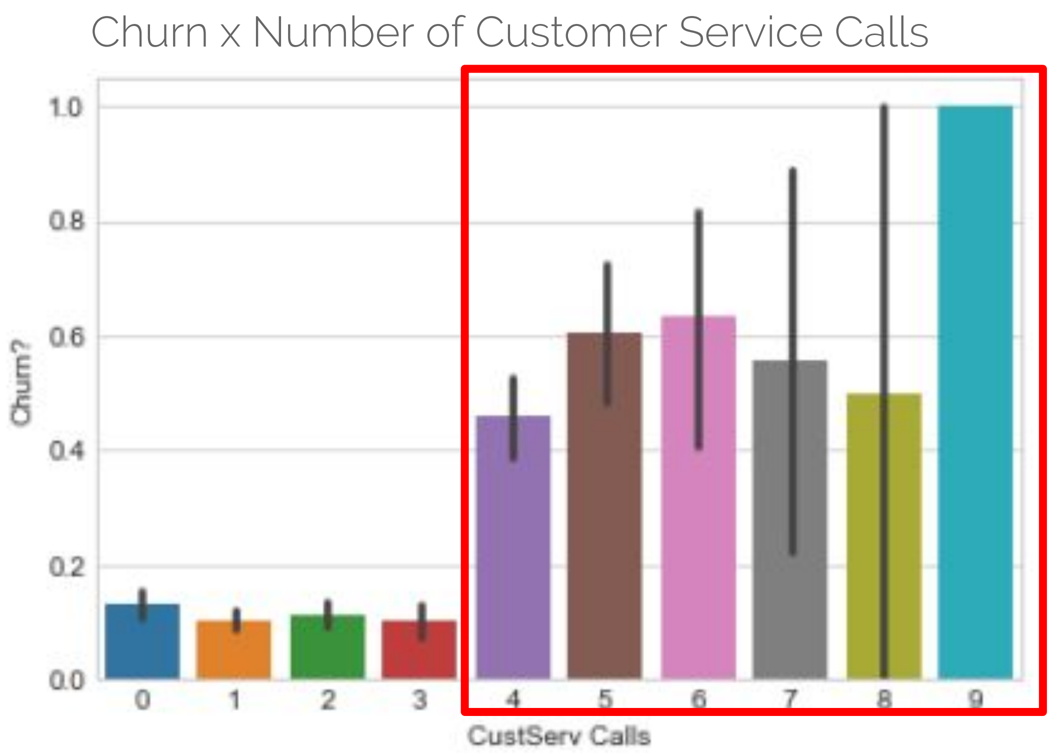
Creating new features

Each new feature created was based on observable patterns that identified increased rate of churn

Total Charge > 80



CustServ Calls >3



2

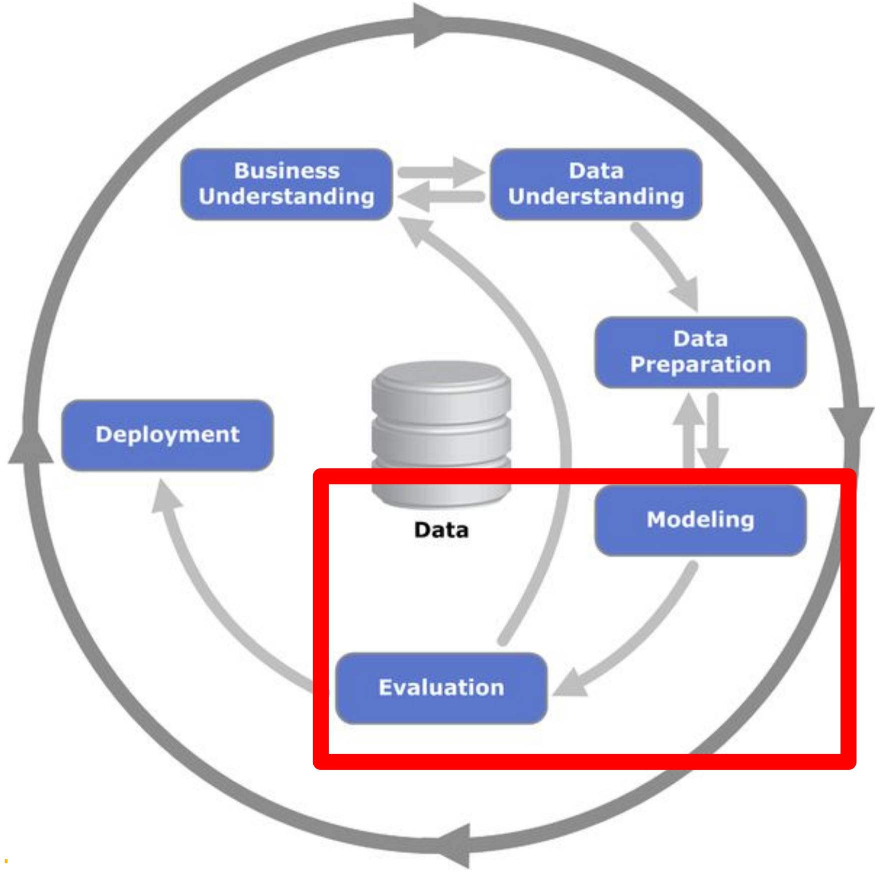
Predictive Modeling (Classification)

Modelling and Evaluation

Decision Tree

Naive Bayes Classifier

Random Forest



Crisp-DM Image Source: https://www.sv-europe.com/crisp-dm-methodology/

Methodology

Modeling Steps

Model training: 60% train set
Model optimization: 20% test set
Model performance evaluation/comparison: 20% validation set

Model Optimization Steps

Features - Leveraged raw feature data, created binned variables (Total Charge > 80, CustServ Calls >3), created dummy variables (State dummy)

Algorithm variants - Assessed performance of algorithm variants

- Bernoulli vs Gaussian (Naives Bayes)
- Gini vs Entropy (Random Forest/Decision)

Hyperparameter tuning - Optimized performance by leveraging gridsearchCV

- Tree Depth, Minimum split (Random Forest/Decision)

Model optimization steps increased model performance (based on accuracy measure)

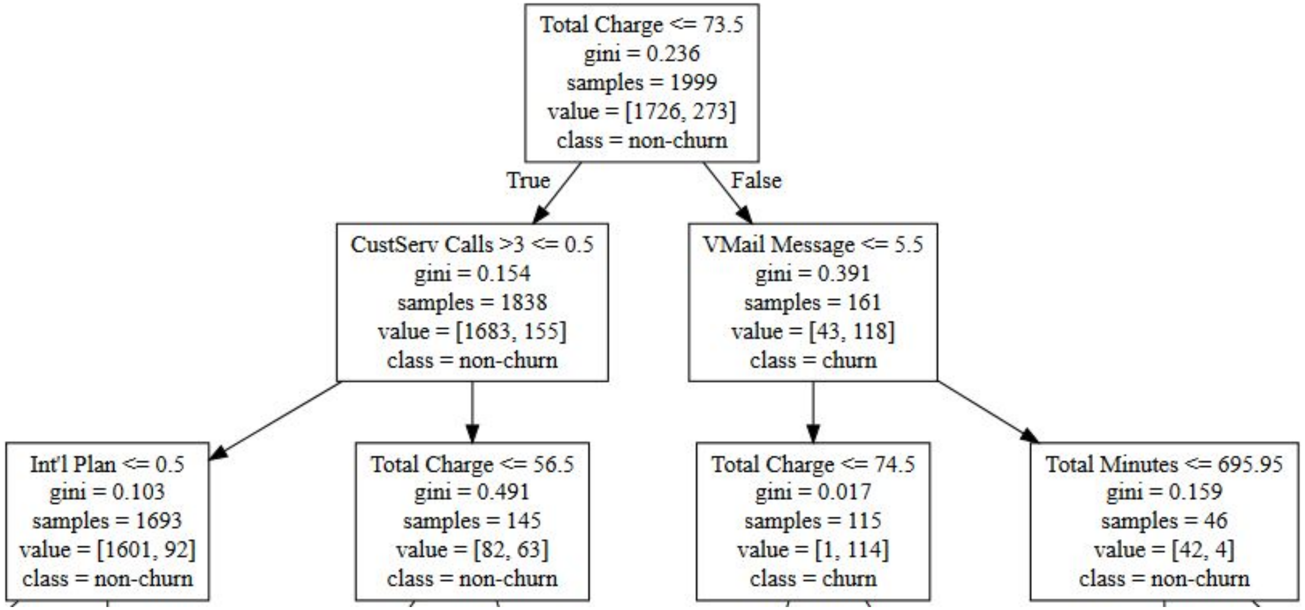
Decision Tree

Benefits:

- Robust predictions, even with limited preprocessing, non-linear relationships and correlated features
- Implicitly does feature selection, i.e., non/less predictive features excluded
- Outputs can be easily explained/actioned, i.e., turned into simple business rules

Final model

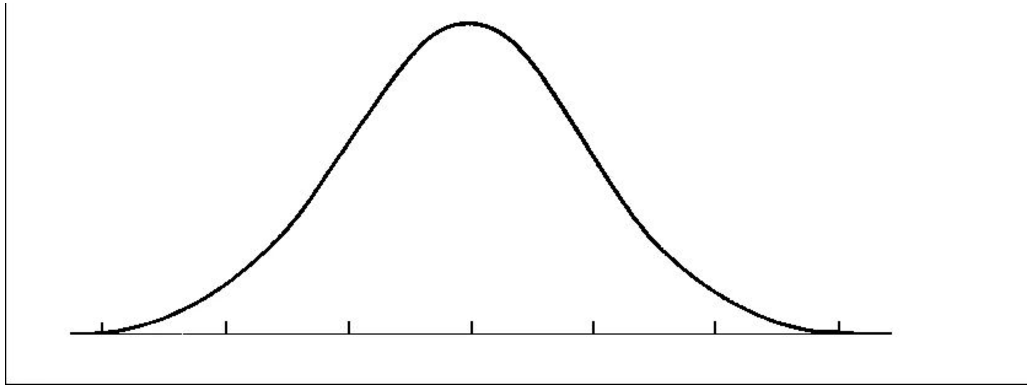
- Hyperparameter settings
  - criterion = 'gini',
  - splitter='best'
  - max_depth=3
  - min_samples_split=5,
  - min_samples_leaf=5
- Features dropped from final model
  - State (dummy variables)
  - VMail Plan (binned variable)
  - Total Charge > 80 (binned variable)



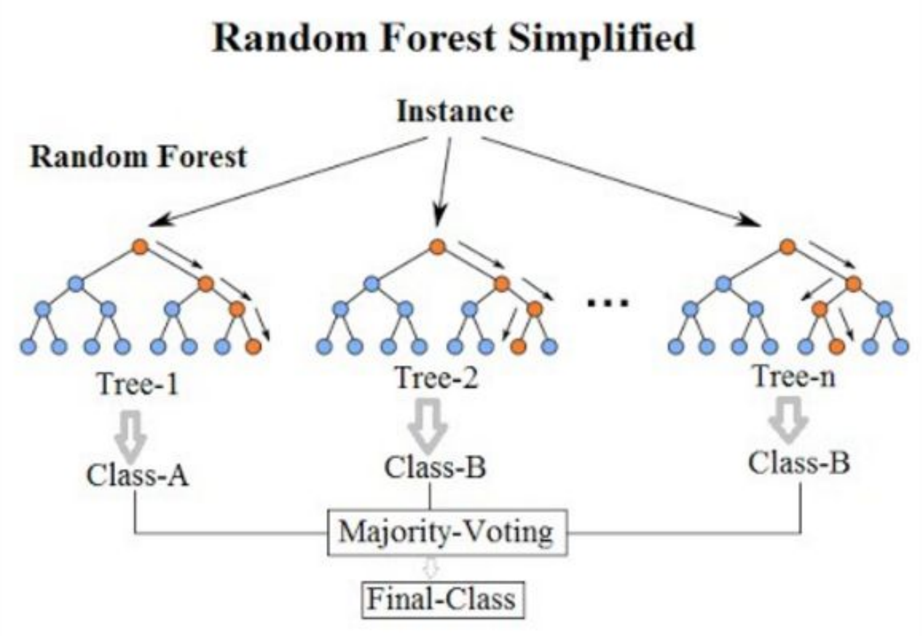
Final model: **95.6% accuracy**

Naive Bayes

- **Benefits:**
  - Very good with small data sets (3333 rows)
  - Computationally fast and simple to implement
- **Drawbacks:**
  - Didn't perform as well as Decision Tree or Random Forest
  - Features were not normally distributed
- **Features Used:**
  - Int't Plan
  - VMail Plan
  - Total Charge > 80
  - CustServ Calls > 3
- **Gaussian Model:**
  - 85.33% Accuracy
- **Bernoulli Model:**
  - Performed slightly better than Gaussian
  - 86.83% Accuracy
- **Binomial Model:**
  - Not applicable



Random Forest



- **Benefits:**
  - Performance at least as good as a Decision Tree
  - Significantly lower risk of overfitting
- **Draw backs:**
  - More computationally taxing
  - Hard to visualize

- **Features Used:**
  - Int't Plan
  - VMail Plan
  - VMail Messages
  - Total Charge
  - Total Charge > 80
  - CustServ Calls > 3
  - State

- Optimized key hyperparameter for model performance settings based on Grid Search
  - max_features = 0.9
  - n_estimators = 2000
  - min_sample_leaf = 2
  - max_depth = 6

- Optimized hyper parameters resulted in a increase in model performance of almost 10% to 96% accuracy.

Evaluation

Metrics considered

- **Accuracy**: Proportion of total true observations to total observations
- **Precision**: Proportion of true churn to total predicted churn
    - Good if cost of false positive is high - Non-churner predicted as churner
    - Better service
- **Recall**: Proportion of true churn to total actual churn
    - Good if cost of false negative is high - Churner predicted not churner
    - Likely to lose their business
- **F1**: Arithmetic mean of Precision & Recall
    - Good when cost of false positives and false negatives are very different

    - **Accuracy > Recall > Precision**

		PREDICTED	
		NOT CHURN	**CHURN**
ACTUAL	**NOT CHURN**	TRUE NOT CHURN	FALSE CHURN
	CHURN	FALSE NOT CHURN	TRUE CHURN

Accuracy

The **Random Forest algorithm** optimized through gridsearchCV **provided the best performance, 0.961 accuracy**

Other algorithm performance

Decision tree algorithm 0.956 accuracy

Naives Bayes algorithm 0.853 accuracy

3

Recommendation

Recommendation

The Random Forest algorithm should be used to identify potential churners moving forward

Key predictors of churn identified were:

Ranking	Feature	Importance
1	Charge	54.46%
2	CustServ Calls > 3	14.90%
3	Int'l Plan	10.39%
4	VMail Plan	9.65%
5	VMail Messages	7.93%

The business should immediately conduct a review on customer charges and enhance their customer service process for customers who have called more than 3 times..

Thanks!

Any questions?

Appendix

Appendix A

Limitations

- Limited time within which to fully optimize model hyperparameters.
- No time series data, e.g., how long after customer service calls did member churn. Key area for future study.
- Limited business context, greater context would have helped determine key model evaluation KPI. Accuracy not the optimal KPI in light of imbalanced target.
- Limited dataset size, with larger data set, additional variables may have entered the predictive model, e.g., state dummies.

Sample Python Code

Data Binning - How was data binning done?

```
#Bin account Length
binwidth_al = int((max(df['Account Length'])-min(df['Account Length']))/5)
bins_al = range(min(df['Account Length']), max(df['Account Length']), binwidth_al)
al_names = ['Newest', 'Avg', "Oldest"]
df['Acc Length Binned'] = pd.cut(df['Account Length'], bins_al)
```

Dummy Variable Creation - How were dummy features created?

```
## Create dummy variables for state to feed into decision tree

s = df['State']
state_dummies = pd.get_dummies(s)
```

Creating bins and dummy variables

Binning variables is the process of dividing continuous variables that are hard to analyze due to small intervals.

Attributes binned:

- Account length
- Mins
- Calls
- Charges

Dummy variables are indicator variables that convert categorical data into numeric data for modeling

Attribute with dummy variables:

- State

Appendix D

Confusion Matrices

Confusion matrices based on optimized version of algorithm

Naive Bayes

		Predicted	
Actual		Non-Churn	Churn
	Non-Churn	564	2
	Churn	86	14

Decision Tree

		Predicted	
Actual		Non-Churn	Churn
	Non-Churn	555	2
	Churn	39	70

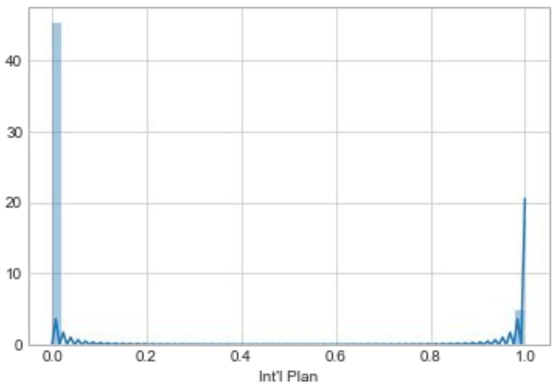
Random Forest

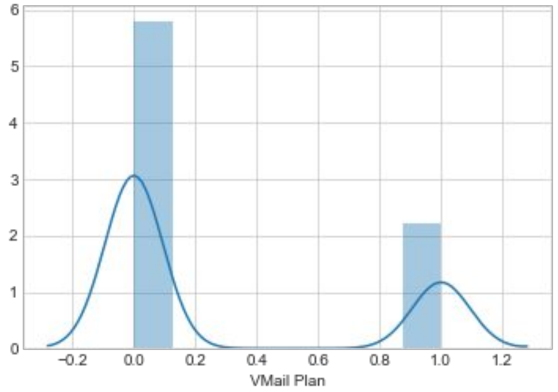
		Predicted	
Actual		Non-Churn	Churn
	Non-Churn	561	1
	Churn	35	79

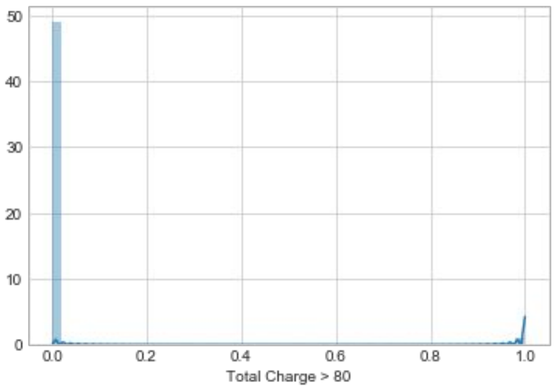
Appendix E - Naive Bayes

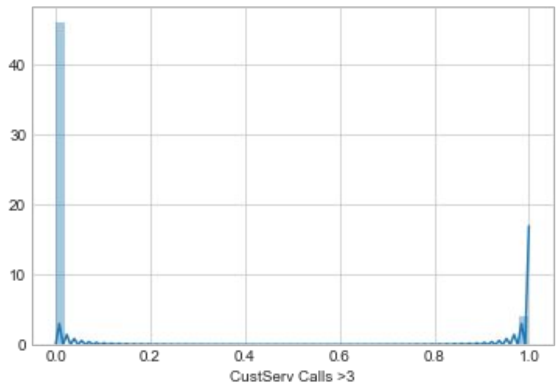
Identifying Weaknesses

The Gaussian model assumes that features follow a normal distribution.



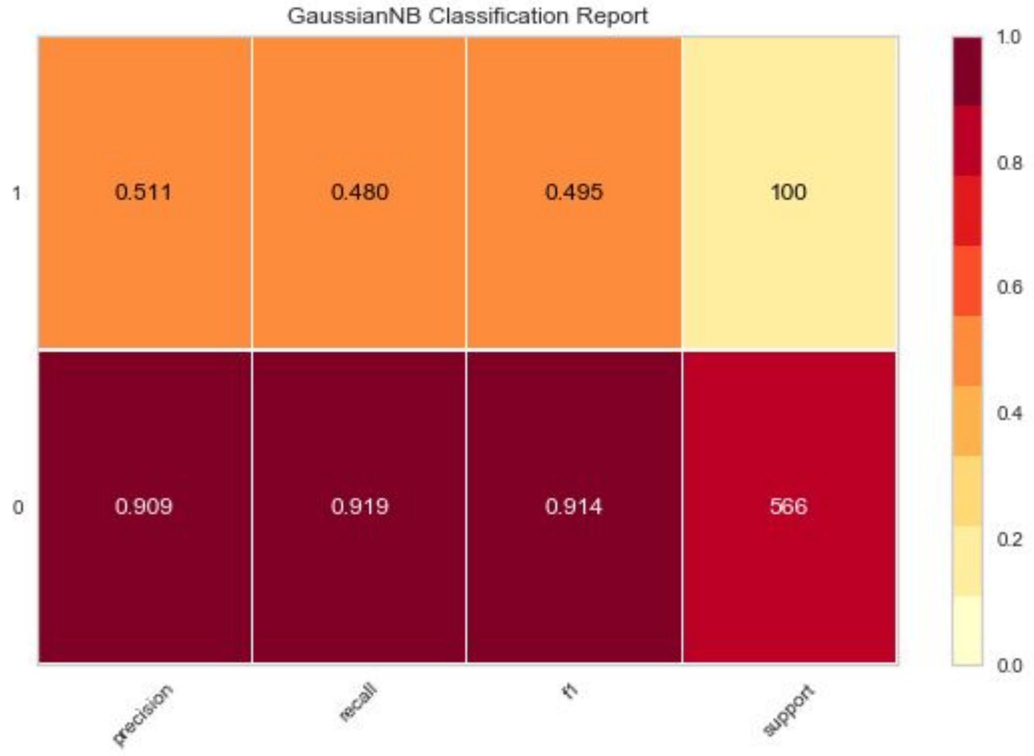


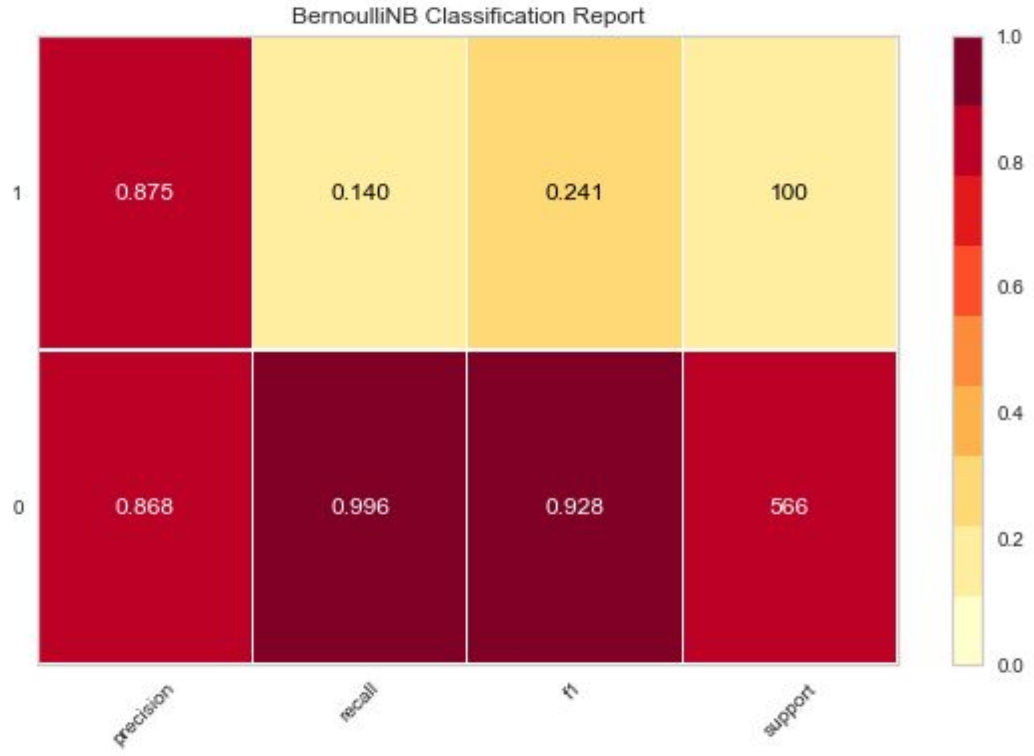




Appendix F - Naive Bayes

Performance Metrics





Appendix G

Feature Importance for Random Forest

```
In [172]: train.columns

Out[172]: Index(['Int'l Plan', 'VMail Plan', 'VMail Message', 'Total Charge',
              'Total Charge > 80', 'CustServ Calls >3', 'AK', 'AL', 'AR', 'AZ', 'CA',
              'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN', 'KS',
              'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'MT', 'NC', 'ND',
              'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'RI', 'SC',
              'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY'],
             dtype='object')
```

```
In [165]: importances = rf.feature_importances_
          std = np.std([tree.feature_importances_ for tree in rf.estimators_],
                       axis=0)
          indices = np.argsort(importances)[::-1]

          print("Feature ranking:")

          for f in range (5):
              print("%d. feature %d (%f)" % (f + 1, indices[f], importances[indices[f]]))

          Feature ranking:
          1. feature 3 (0.544660)
          2. feature 5 (0.149036)
          3. feature 0 (0.103856)
          4. feature 2 (0.096536)
          5. feature 1 (0.079295)
```