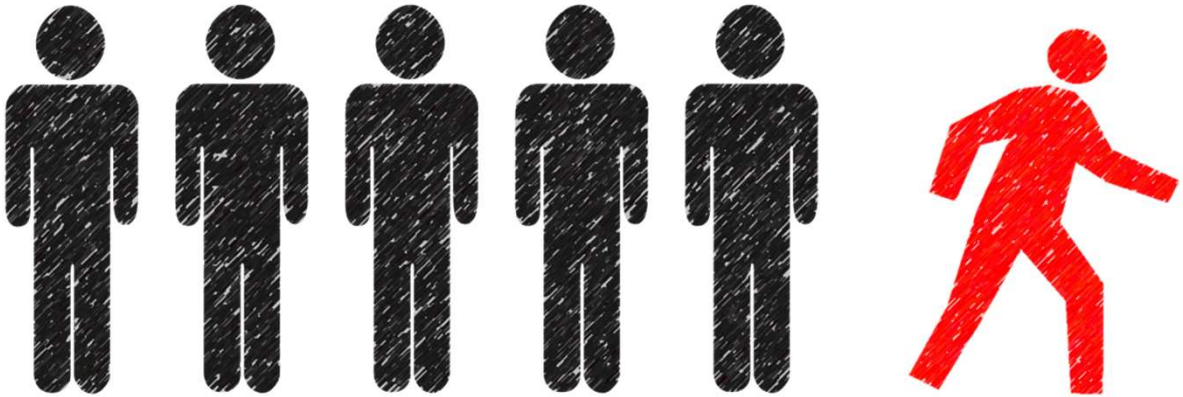
Predicting Customer Turnover

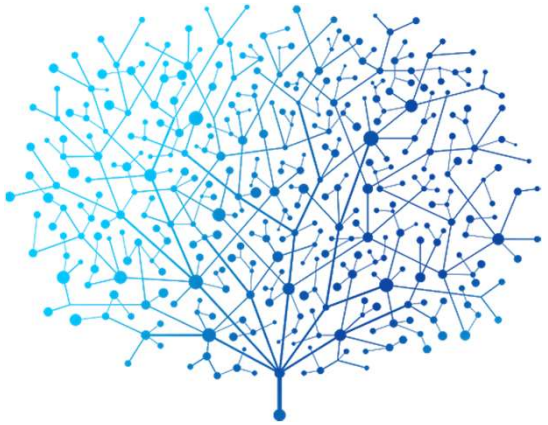
Vivian Dang and Joe Buzzelli



April 17, 2020

Machine learning empowers Syriatel to mitigate customer turnover

- Customer turnover risks Syriatel's profitibility
- After developing our statistical model, we succeeded our goal of predicting customer churn with greater than **95% precision**



Assumptions and data sources

Assumptions

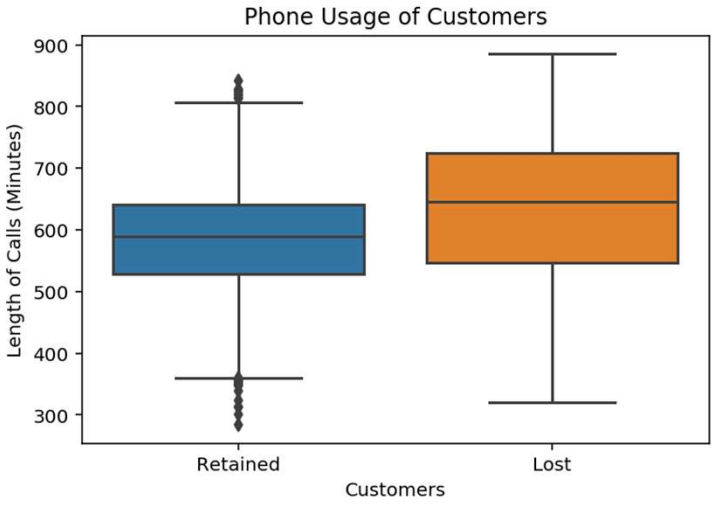
- All customers received similar products or services
- No time period is provided for this data, it is assumed that Syriatel can take actions to improve customer retention

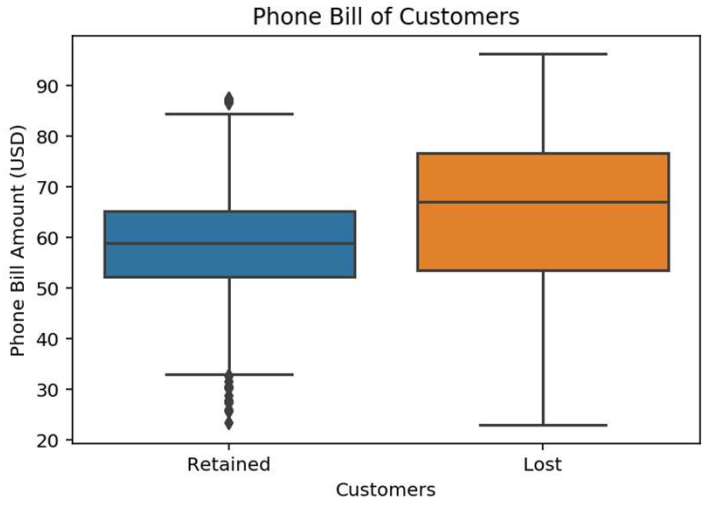
Data sources

- Syriatel's customer turnover data set, no dates provided

Lost customers incurred higher charges and talked longer

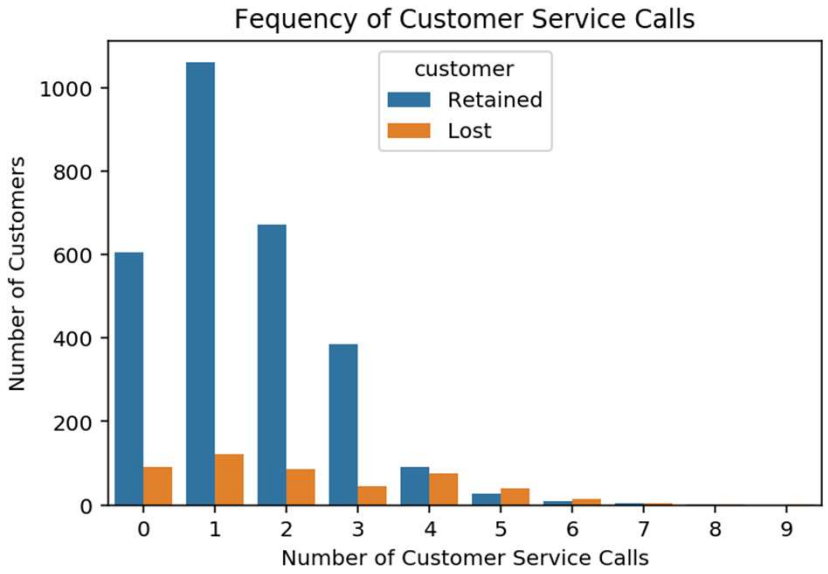
- Lost customers have higher total charges and use more minutes than their counterparts
- Extra talk yields higher total charges exhibited by lost customers





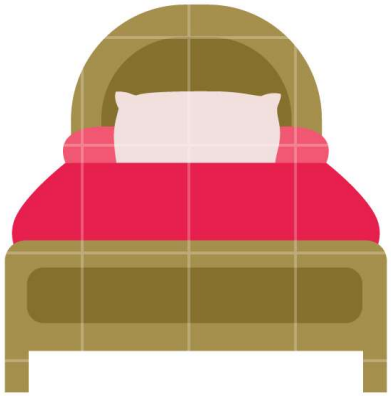
Syriatel appears likely to lose frequent customer service callers

- Despite representing only **14%** of Syriatel's customer base, lost customers call customer service **disproportionally** more often



We deployed the "Goldilocks" approach to find the right model

- In an iterative fashion, we assessed several different models until we found the best match with Syriatel's customer data
- This is similar to…



If our model was an archer…

The data identifies three key factors for customer turnover

1. Total bill
   - Customers paying over **$60** are more likely to turnover

2. Total minutes
   - Customers using over **600 minutes** are more likely to turnover

3. Customer service calls
   - Lost customers typically place **more than 4 calls**

Recommendations

- Monitor customers that use **over 600 minutes** with a **bill over $60**
- Offer promotions to customers who have made **over four** customer service calls
- Provide a survey to customers when they leave to augment existing data

Next Steps

- Improve the current model with additional data from Syriatel to **better predict** customer turnover
- Explore machine learning solutions for Sytriatel's **customer service department** to mitigate customer attrition
- Conduct a **sentiment analysis** of Syriatel's customers to discover issues not present in operational databases


Thank you for
your time.
Are there any
questions?

Backup Slides

We aimed to maximize precision scores in our models

Precision calculation:

$$\frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}}$$

In total, we used nine different models in this project

Models include:

- Gradient Boosting Classifier
  - Default and tuned
- Random Forest
  - Default and tuned
- Logistic Regression cross validation estimator
  - Default and tuned
- Nearest neighbor (KNN)
  - Default and tuned
- Decision tree

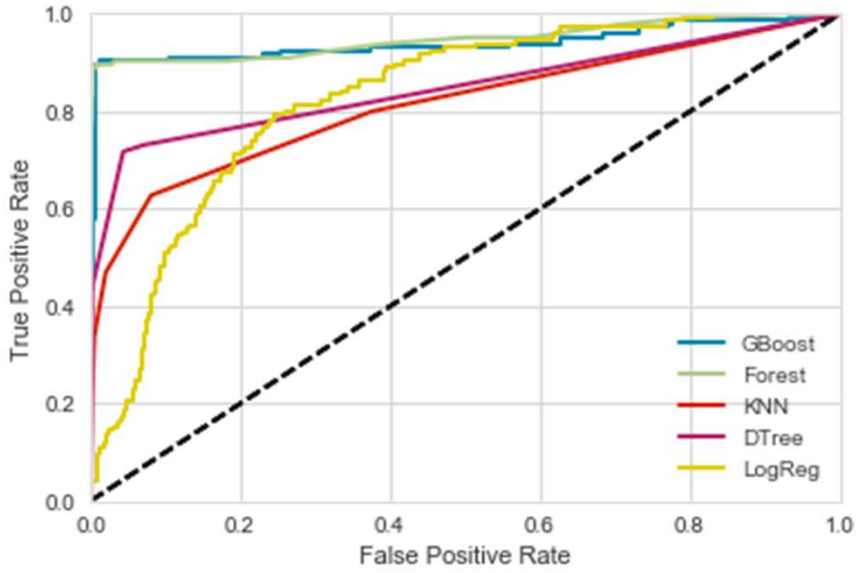
For each model type, the best performer was tested

The table below outlines the best performing models per type:

Best Model by Type	Precision	Accuracy	Recall	F1
Gradient Boosting Classifier	0.970	0.980	0.890	0.928
KNN (n=7)	0.925	0.900	0.338	0.495
Random Forest	0.977	0.981	0.890	0.931
Decision Tree (max_depth = 1)	0.970	0.917	0.441	0.607
Logistic Regression (solver = 'lbfgs', Cs=50, penalty='l2')	0.528	0.857	0.131	0.210

ROC Curves for each tested model

The chart below includes the ROC curve for each model vs the test data



Random Forest Notes

- Strengths
  - RFs train each tree independently, using a random sample of the data. This randomness helps to make the model more robust than a single decision tree, and less likely to overfit on the training data -> one of the most accurate learning algorithms available
  - RF is much easier to tune than GBM. There are typically two parameters in RF: number of trees and number of features to be selected at each node.
  - RF is harder to overfit than GBM
- Weaknesses
  - Overfit for some datasets with noisy classification/regression tasks
  - Classifications made by random forests are difficult for humans to interpret
  - Categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels
- Default parameters
  - Default: (n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None

Gradient Boosting Classifier

- Strengths
  - Build trees one at a time, where each new tree helps to correct errors made by previously trained tree.
  - Great for very unbalanced data
- Weaknesses
  - Sensitive to overfitting if the data is noisy.
  - Training generally takes longer because of the fact that trees are built sequentially.
  - Harder to tune than RF. There are typically three parameters: number of trees, depth of trees and learning rate, and each tree built is generally shallow.
- Default parameters
  - (loss='deviance', learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=None, max_features=None, verbose=0, max_leaf_nodes=None, warm_start=False, presort='deprecated', validation_fraction=0.1, n_iter_no_change=None, tol=0.0001, ccp_alpha=0.0)