SYRIA TEL CUSTOMER CHURN

Syria Tel is one of the three telecommunication company in Syria. It is a mobile network provider in Syria. Syria Tel needs competitive edge against its competitors including data driven insights on how to improve their business.
The company has recently observed that customers are churning, discounting their servies. Loosing customers is a great concern to the company. As a result, the company is loosing money. The company is interested in reducing the amount of money it looses because a customer will soon not be here.
The company needs a data driven approach to this issue. The company needs to know the features that determine if a customer will soon quit the company. These should be used to predict if the customer would soon quit the company or not so that the company can make the appropriate decisions to reduce money lost because of customers who will not stay for long.

PROBLEM STATEMENT

Syria Tel has discovered that their customers have started to churn, discontinue their services. This poses a major risk to the company. The challenge is to Build a classifier to predict whether a customer will ("soon") stop doing business with SyriaTel, a telecommunications company

OBJECTIVES

Determine what features will indicate if a customer will soon discontinue their services with Syria Tel

DATA UNDERSTANDING

This project uses data from Churn in Telcom Dataset
The data has 3333 rows, customers, and 21 columns, features.
The dataset has no null values or duplicated values
Here is a summary of the contents of the columns:
1. state: The state of the customer.
2. account length: The length of the account in days or months.
3. area code: The area code of the customer's phone number.
4. phone number: The phone number of the customer.
5. international plan: Whether the customer has an international plan or not.
6. voice mail plan: Whether the customer has a voicemail plan or not.
7. number vmail messages: The number of voicemail messages the customer has.
8. total day minutes: Total minutes of day calls.
9. total day calls: Total number of day calls.
10. total day charge: Total charge for the day calls.
11. total eve minutes: Total minutes of evening calls.
12. total eve calls: Total number of evening calls.
13. total eve charge: Total charge for the evening calls.

14. total night minutes: Total minutes of night calls.
15. total night calls: Total number of night calls.
16. total night charge: Total charge for the night calls.
17. total intl minutes: Total minutes of international calls.
18. total intl calls: Total number of international calls.
19. total intl charge: Total charge for the international calls.
20. customer service calls: Number of times the customer called customer service.
21. churn: Whether the customer churned or not (True/False)

DATA CLEANING

Data cleaning is useful in finding duplicates and null values that may mess with our machine learning algorithms.
This is a decent dataset. There are no duplicates nor null values.

DATA EXPLORATION AND FEATURE ENGINEERING

All columns has a normal distribution, except area code and number of voice mail messages.
There is no big difference in the account length of the customers who churned compared to customers who stayed.
Area 510 and 415 look like the most likely to churn.
Drop unnecessary columns like phone number and columns with multicolliniearity.
Transform the target variable to 0s and 1s.
One hot encode the categorical variables.
Model validate by splitting the data to training and test data by a 75/25 split.
Use Min Max Scalers because there are some features whose data are not normally distributed.

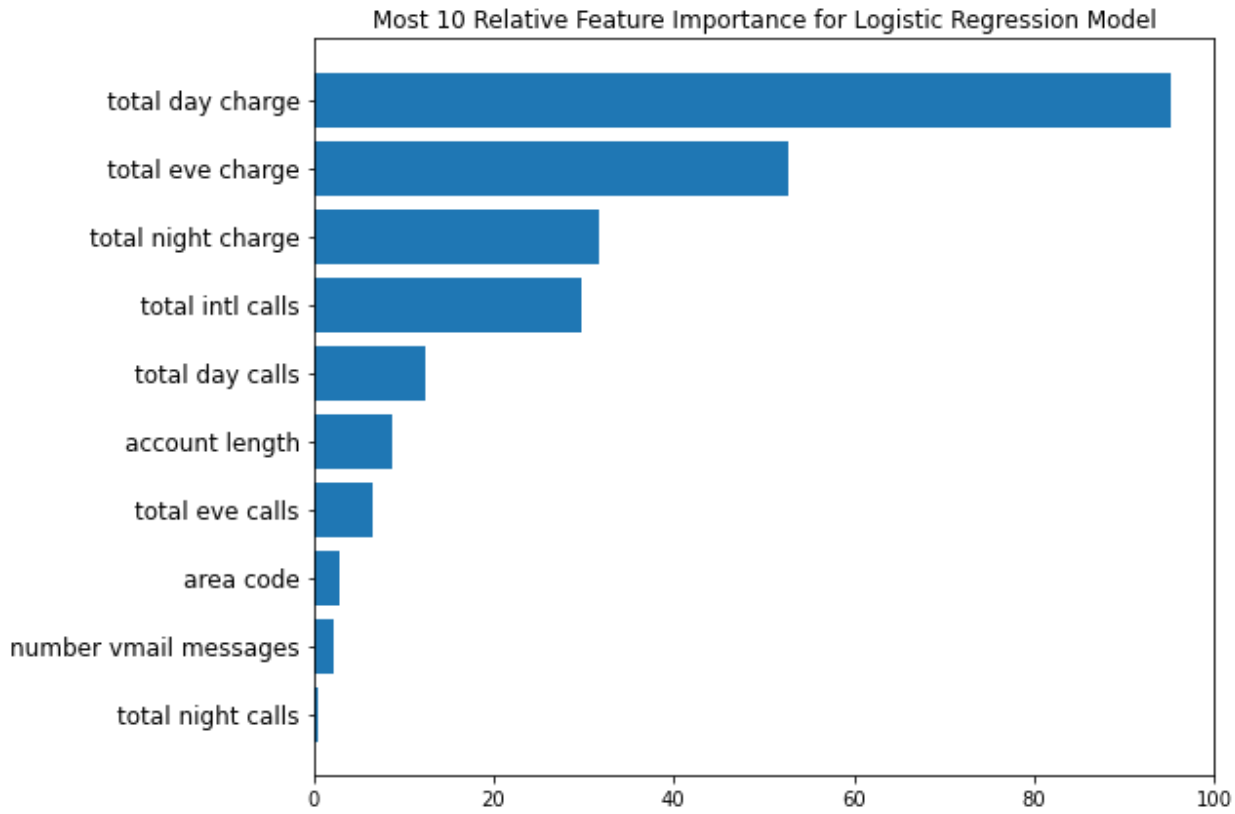
MODELLING

We use modelling rather than analysis because:
1. We want to predict beyond the observed data that we have right now.
2. We would like to capture complex relationships and dependancies within the data that may not be easily discernible with simple data analysis techniques.
3. Going forward, we would like to automate our decision making process beyond manual intervention.

BASELINE MODEL - LOGISTIC REGRESSION

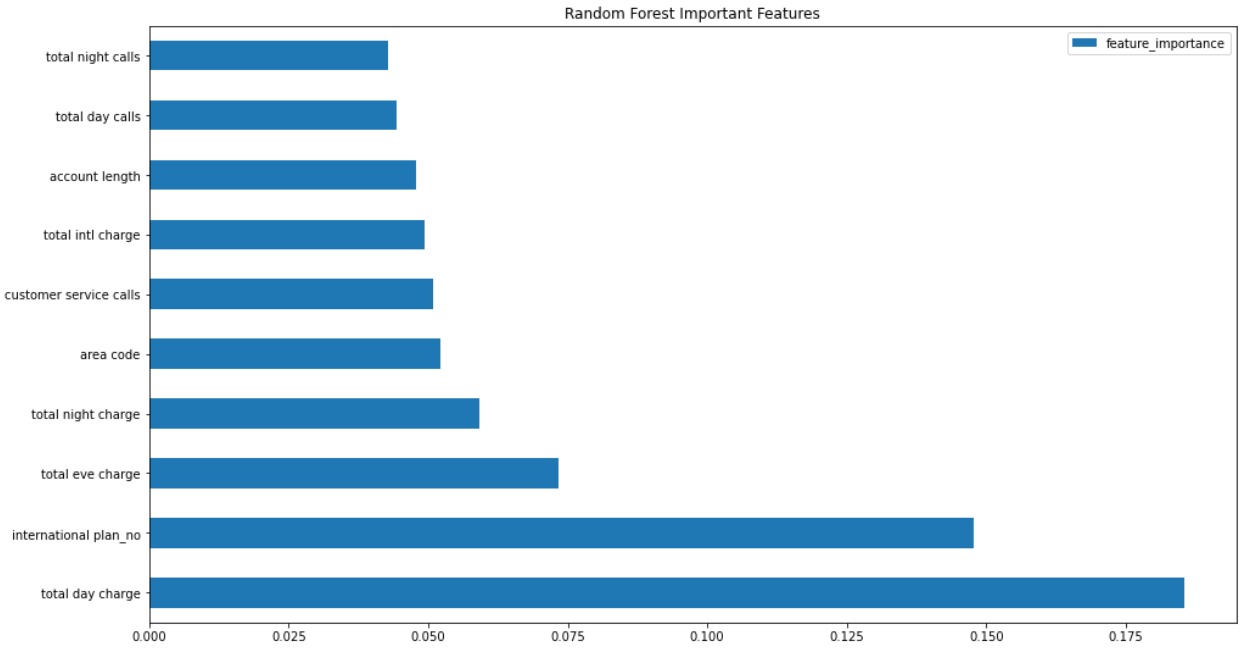
Logistic regression is a supervised classifier algorithm.
It performs a classification of your data by finding the probability of a model belonging to one group vs another.
It is transformed in a way that the outcomes takes a value between 0 and 1.
We will use logistic regression as our baseline model because it is a simple model.
The model has an accuracy of 0.87.
It performs moderately well.
The train AUC is 0.92 while the train AUC is 0.63 The drop in performance in the test data is an indication of overfitting.

Being a linear model, the model may fail to capture relationships in the data that are not linear. We could try a more complex model and try and improve our scores such as f1score and accuracy as well as trying to handle the overfitting.
The most important features are:



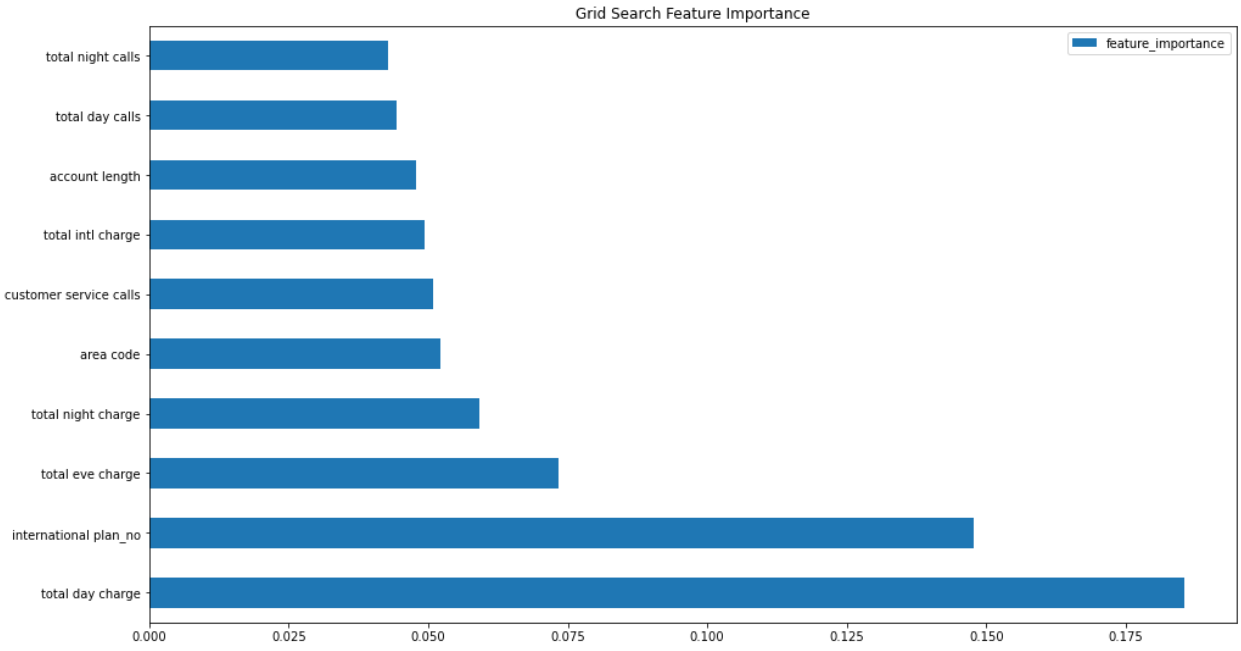
RANDOM FOREST

Random Forest is a supervised classification algorithm.
It is an ensemble method for decision trees.
Fit the model with default values for the hyperparameters
The model has a 0.924 accuracy.
The model has a train AUC of 1 and a test AUC of 0.83
The difference between the train and the test scores indicate that our model overfits on training data.
We could try hyperparameter tuning to try and fix the overfitting, improve the
The most important features here are:



GRID SEARCH - HYPERPARAMETER TUNING

Each dataset is different, and the chances that the best possible parameters for a given dataset also happen to be the default parameters set by scikit-learn at instantiation is very low.
We need to try hyperparameter tuning.
The Model has test accuracy 91.93% score.
The model has a train AUC of 0.98 and a test auc 0f 0.81
The model reduces the overfitting compared to the randomforest model.
Could we improve our model? The random forest we used above uses bagging. However we could try and use Gradient boosting's powerful method, ensemble method to try and improve our model.
The cross validation score of our model is 92%



XGBOOST

XGBoost(eXtreme Gradient Boosting) is a stand alone algorithm that implements popular gradient boosting algorithms in the fastest, most performant way possible.
It is a great choice for classification tasks.
The model test accuracy score is 0.9546.
The model training AUC is 0.99 while the test AUC is 0.87
This is a good metric , however, there is an indication of overfitting the training data.
We could try hyperparameter tune the XGBoost to try and reduce the overfitting and improve model performance

GRIDSEARCH - XGBOOST

Each dataset is different, and the chances that the best possible parameters for a given dataset also happen to be the default parameters set by scikit-learn at instantiation is very low.
We need to try hyperparameter tuning.
The model has a 94% test accuracy score.
The difference in the test and train scores indicate the model may be indicating.
The model has a test auc score of 0.856 and a training auc of 0.99.

EVALUATION

I evaluate the models using recall and AUC.
Precision measures how many of the predicted positives are actually positive. This is a good metric to look at because we are concerned of false negatives more than false positives. If we fail to predict a customer will churn, we risk loosing money spent on that customer. We want a model with the best precision.

AUC measures the ability of a model to distinguish between classes. We want the model that will distinguish the two classes best.

model	test recall	
0	logistic regression	0.32
1	Random Forest	0.66
2	Grid Search Random Forest	0.67
3	XGBoost	0.77
4	Grid Search XGBoost	0.74