Understanding Why Churn prediction ?

1. Retention of customers is important because of 2 factors

    A. Growth perspective

        a. Are you able to onboard new users and able to retain them?
        b. Is your product able to strike an interest/value with users
        c. Important indicator of mapping between product and market need

    B. Value perspective

        a. Marketting is a costly procedure
        b. Acquiring a new user is costlier than retaining an old user
        c. Business point of view it makes more sense to retain these older users

2. If you know which customer is going to leave the platform you can use this knowledge to create attractive offers or discounts to retain them

3. This will make the customer feel that company cares about their interest.
4. Which inturn will be a value add.

Translating given business problem into a machine learning problem

`Classification Problem`

1. Given features of a user, services provided by telecom service can we predict if the customer is going to churn out or not?

`Regression Problem`

1. Given features of a user, services provided by telecom service can we predict tenure of the customer for using the service?

EDA observations and insights

1. Talking about `Churn` label it has imbalance
    - counts of people churning out is less than the people not churning
2. Talking about `tenure` target it has 2 distinct peaks
    - one users who have very less tenure
    - other users who have high tenure
    - latter peak is lower than the prior
    - meaning there are more people with less tenure than people with more tenure
3. We can see most of the features are categorical in nature with 2-3 categories at max

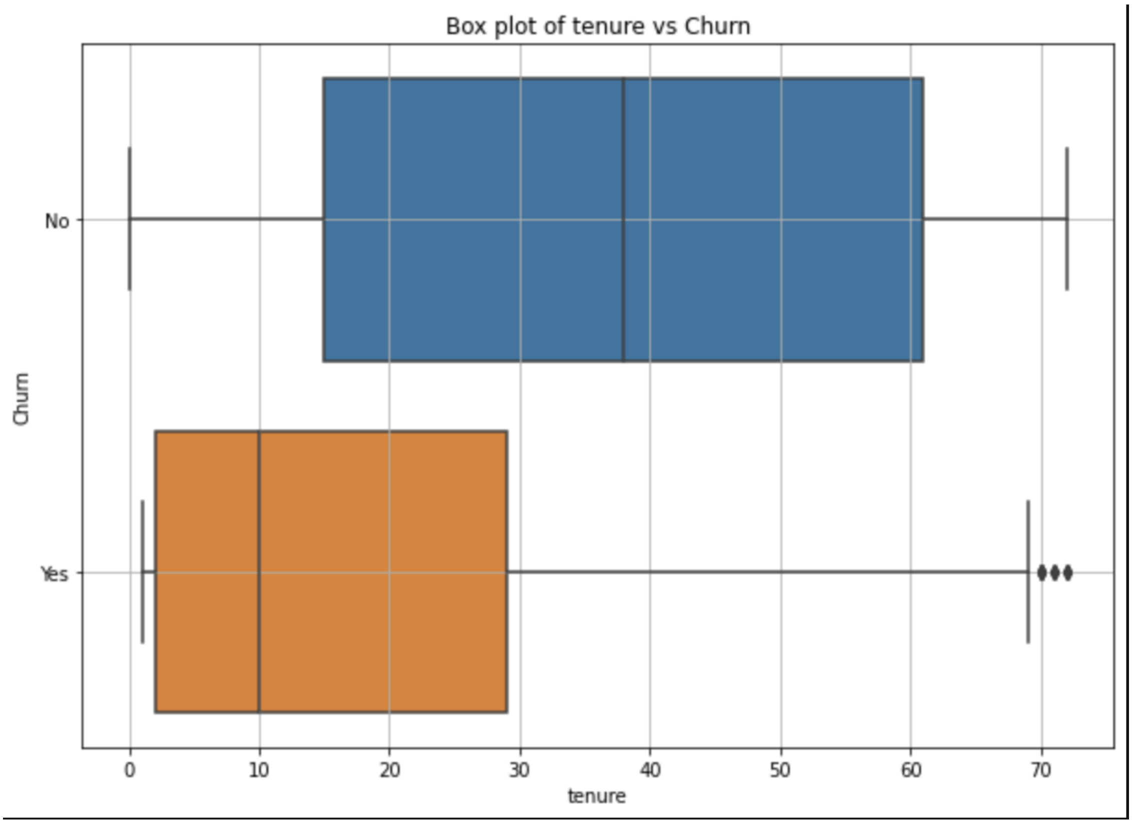
1. In monovariate analysis we accounted the following:
    A. Categorical features:
        a. occurences - distributions using bar charts

b. unique categories in each variable
c. balance/ imbalance in categories

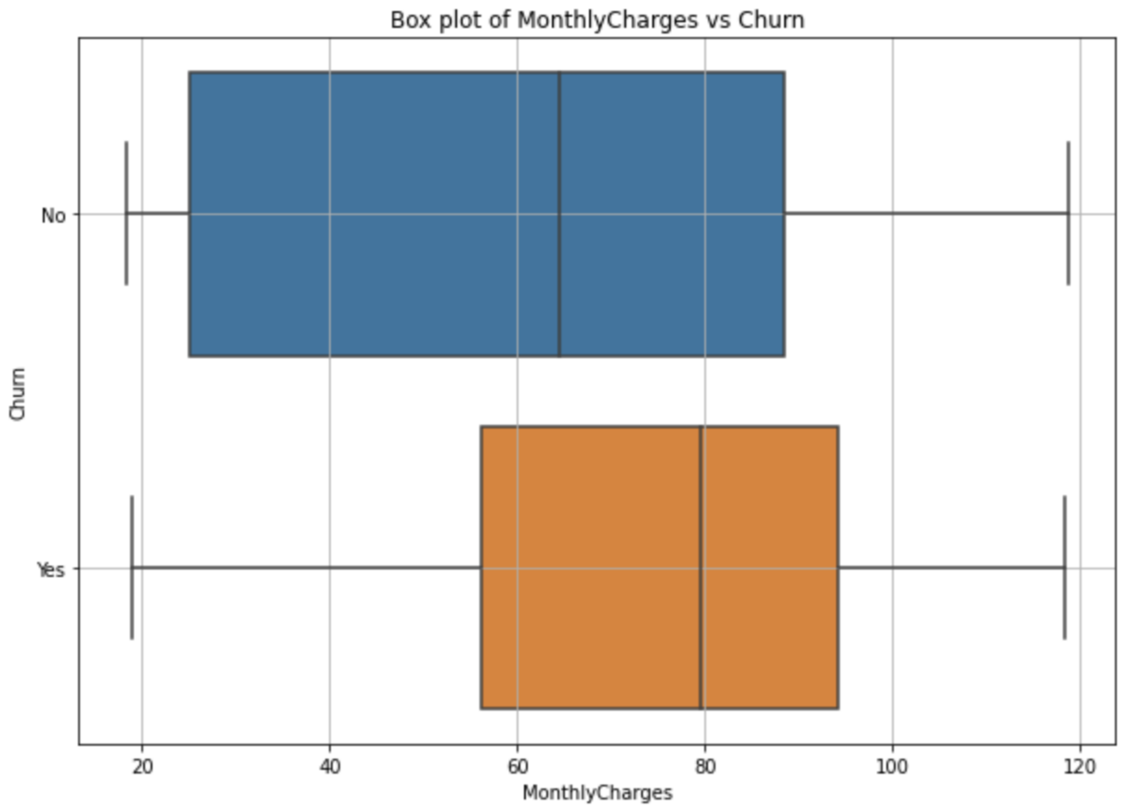
B. Real features:
    a. frequency - histograms
    b. nature of histograms

1. In Bivariate analysis we tried to answer the following:

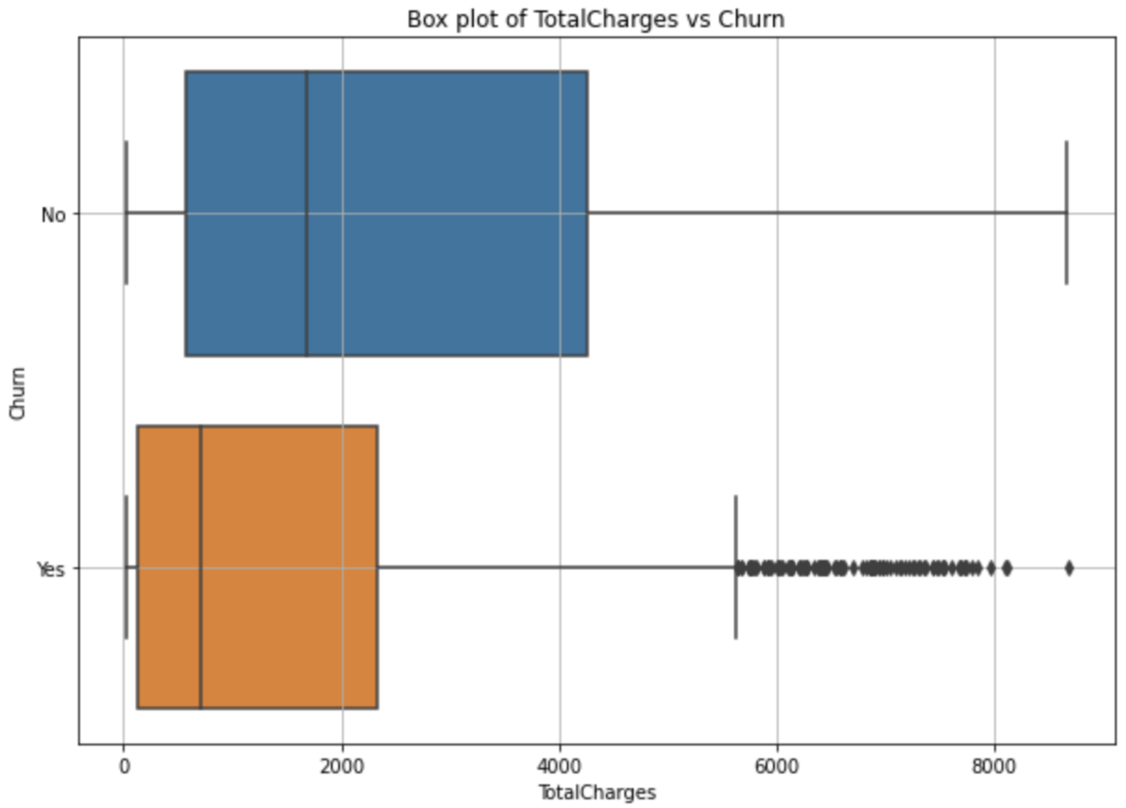
    A. Are we able to establish some connection in between features and the label/target variable
    B. We also saw the distributions of tenure w.r.t churn

        a. We could see that people having tenure in between 20-60 months are less likely to churn
        b. We could see that people having tenure in between 0-30 months are more likely to churn
        c. 

    C. We also saw that variables like `MonthlyCharges` and `TotalCharges` brought some distinct patterns out w.r.t to churn and tenure



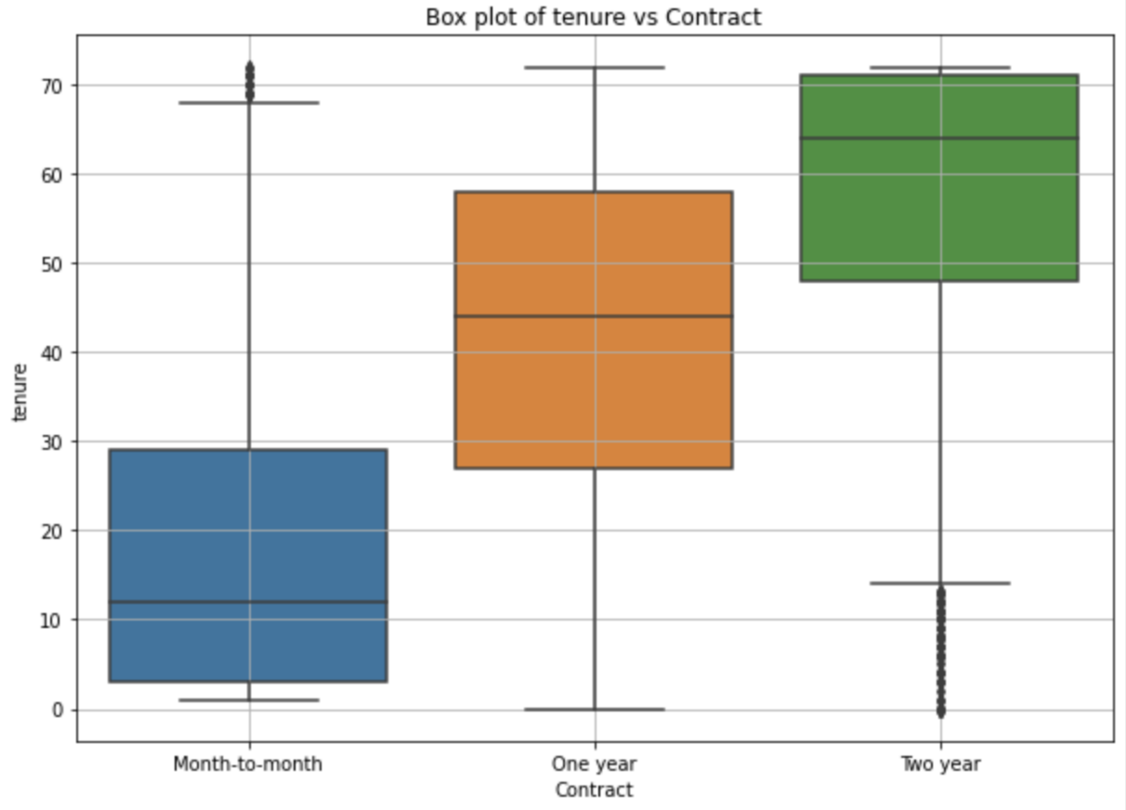
a.



b.

c. Similar patterns could be observed in tenure

D. We also saw distinct pattern in how contract is related to tenure

a. 

Some thoughts that will help in hypothesis building

Thinking from a user perspective.

What would make a user churn from a telecom service?

1. is user getting additional service the user is getting by staying with telecom.
2. is user getting better cost of service else where?
3. is billing easier or not for current telecom service

Initial hypotheses

1. Can we say that cost of service is contributor to churn?
2. Can we say that having additional internet services would contribute to churn?
3. Can we say that having a certain method of payment contribute to churn?
4. Can we say that longer the contract the customer is less likely to churn?
5. Can we say that senior citizen churn less as there is some effort required to change the service?

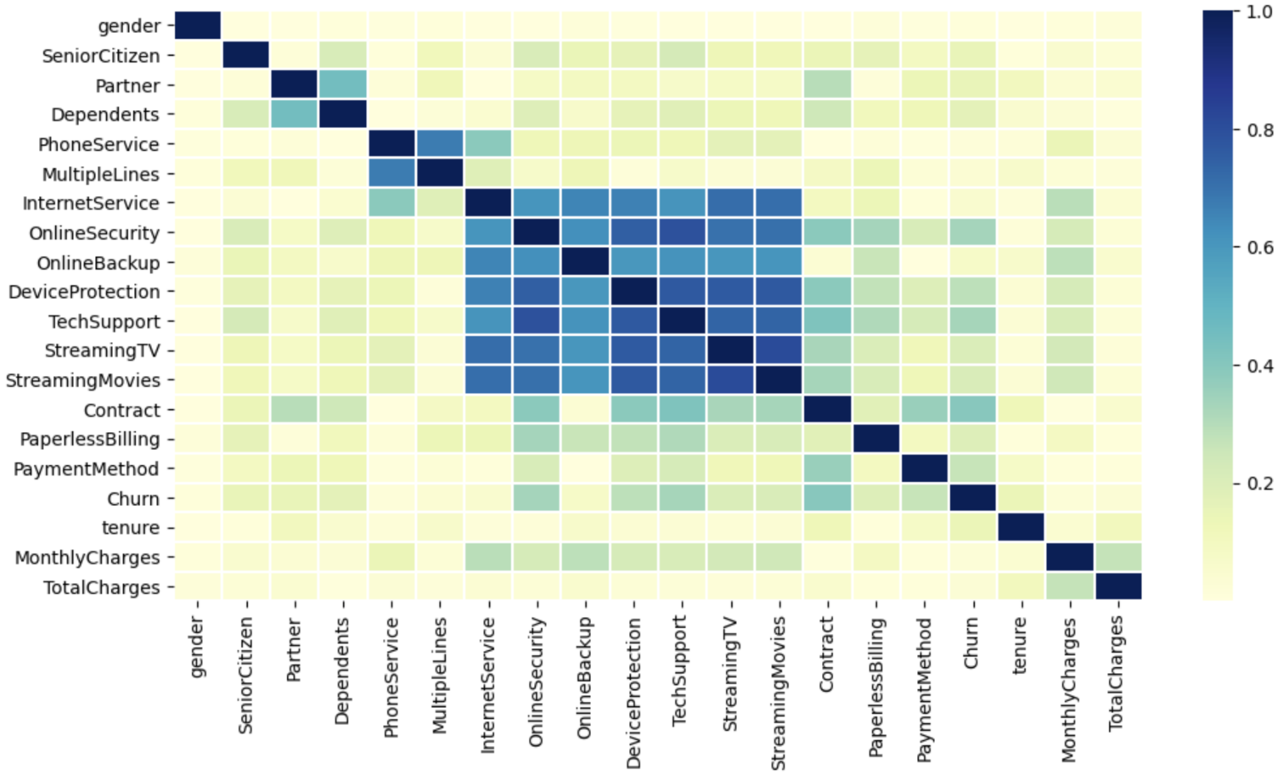
`NOTE` similar hypotheses can be created for tenure.

Correlation analysis

1. We generally remove highly correlated features
2. Reason say if x (input feature), y (target) are highly correlated
3. We might miss out on some other explanatory feature

4. Chances are we might not learn the target variable well enough due to this highly correlated feature reference

---



1. Plotted a correlation heatmap between features and dropped highly correlated features using a certain threshold

We could see that there is high correlation in:

1. `StreamingMovies` and `[InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingMovies]`

Feature engineering

Boolean features

1. Boolean features were encoder using simple mapping of 0,1

Categorical features

1. Categorical features were encoded using binary encoder

cyclical Time features

As mentioned in the problem statement, we had to add a time column, randomly assigned dates in between a certain range and designed features on top of these timestamps.

Understanding Cyclical time features

Let us use `sin_hour = sin((hour/24)*(2*pi))` for tracing an example.

Let `r = hour/24`

1. What are the values that `r` can take?
2. r = {0/24, 1/24, ... 23/24, 24/24}
3. r = {0, 0.041, ... , 0.95, 1}
4. r has range of [0,1]

`sin(2*pi)` = `sin(360 degrees)`

1. we will be multiplying `r` with `2*pi`
2. essentially we are calculating what fraction of `2*pi` are we looking at

Scaling

Real features were min max scaled and saved.

Feature Selection

1. Feature selection was done 2 times one for classfication task and other for regression task
2. Both the tasks were done using respective derivatives of random forest i.e random forst classifier and regressor respectively

Feature Selection - Classification

We can see that top features for classification according to random forest classifier's feature importance are:

1. 'TotalCharges',
2. 'MonthlyCharges',
3. 'Contract_0',
4. 'cos_day',
5. 'sin_day'
6. 'TechSupport_0'

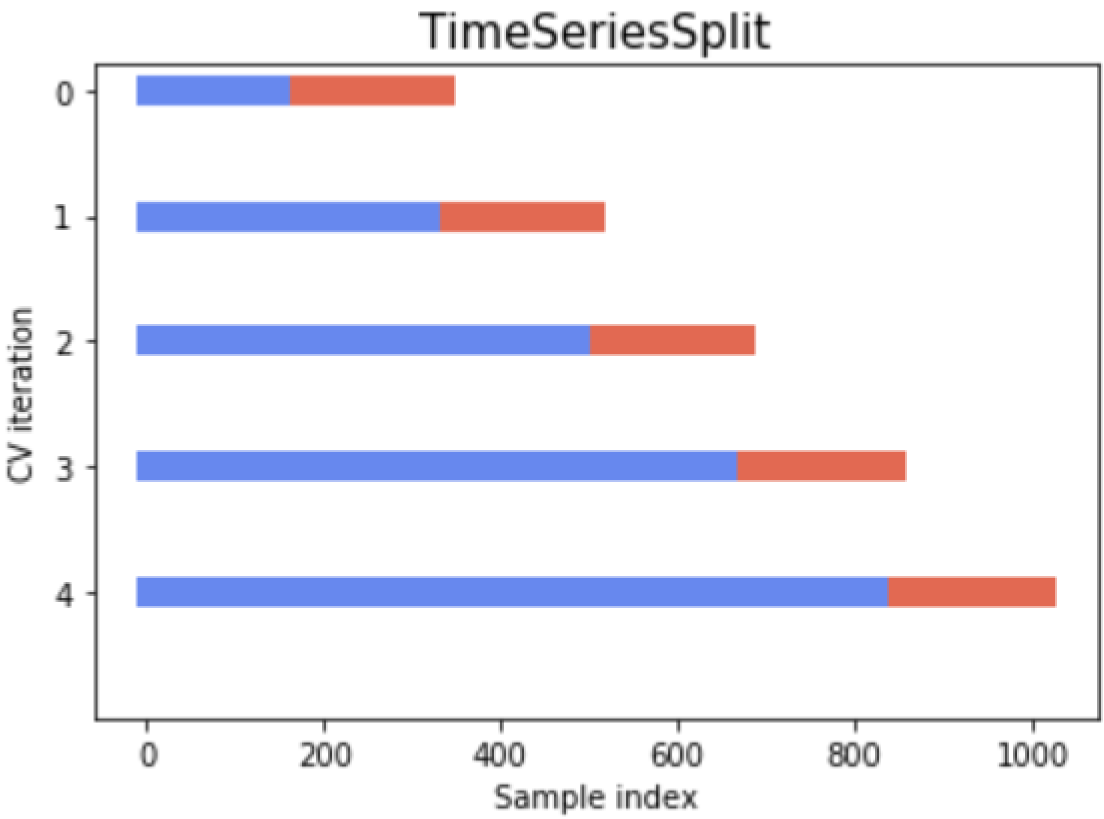
Feature Selection - Regression

We can see that top features for regression according to random forest classifier's feature importance are:

1. 'TotalCharges'
2. 'MonthlyCharges'
3. 'Contract_0'
4. 'Contract_1'
5. 'InternetService_1'
6. 'InternetService_0'

Splitting data

1. Data was split into two parts
2. train, test sets
3. train set was further divided into cv set using TimeSplitCV
4. Image of time series split
   A. blue = train, red = cv
   B. Scores of each iteration will be averaged to get final cv score



Modelling and Model evaluation

As mentioned in the problem statement used top 6 features according to feature importances for modelling Churn and tenure.

What should we look for in classification?

> Precision or recall?

What is more harmful to have?

1. FP → person who is not going to churn being predicted as 1
2. FN → person who is going to churn being predicted as 0
3. FN is more harmful

Hence recall is more important.

- recall = TP/(TP+FN) = TP/P
- out of total positives how many are actually(true) positives

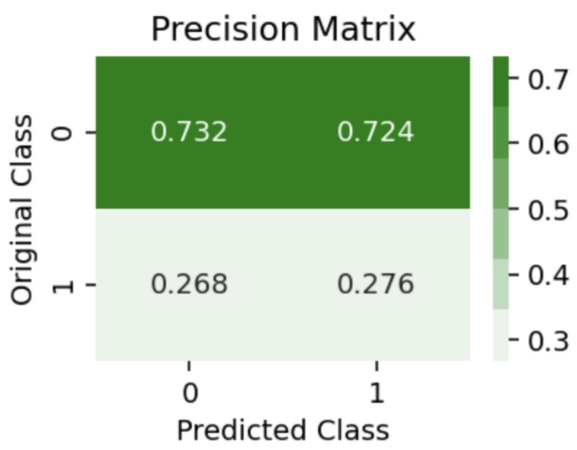
To be more specific recall on class 1 is more important

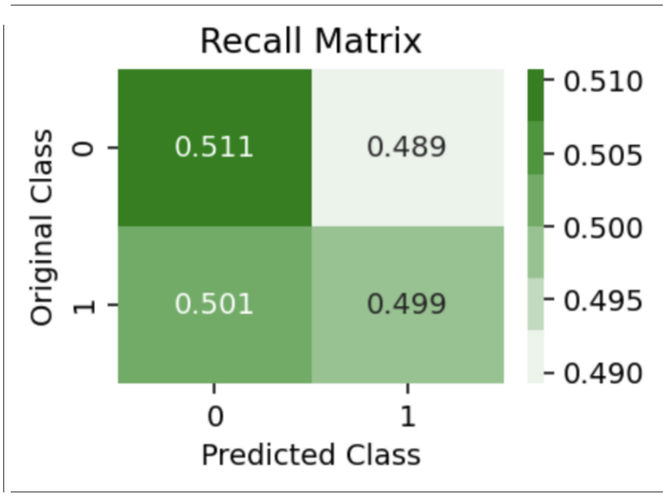
Setting baseline for classifier

1. Used a random model to get target labels

Observations:

1. log_loss on train and test sets was very close to 1 - (train 0.98 , test 0.94)
2. Precision and recall matrix of both train and test sets were very similar attaching PR matrix of predictions on test set





1. We can see how random model is predicted each class with equal probability in recall matrix
2. Whereas in Precision matrix we can see that how imbalance is affecting the predictions
3. because churn labels are in ratio of ≈ 1:3 (yes:no) probability of predicting No label i.e 0 class is around 70%

We had established from random model that log_loss should be better than 1.0

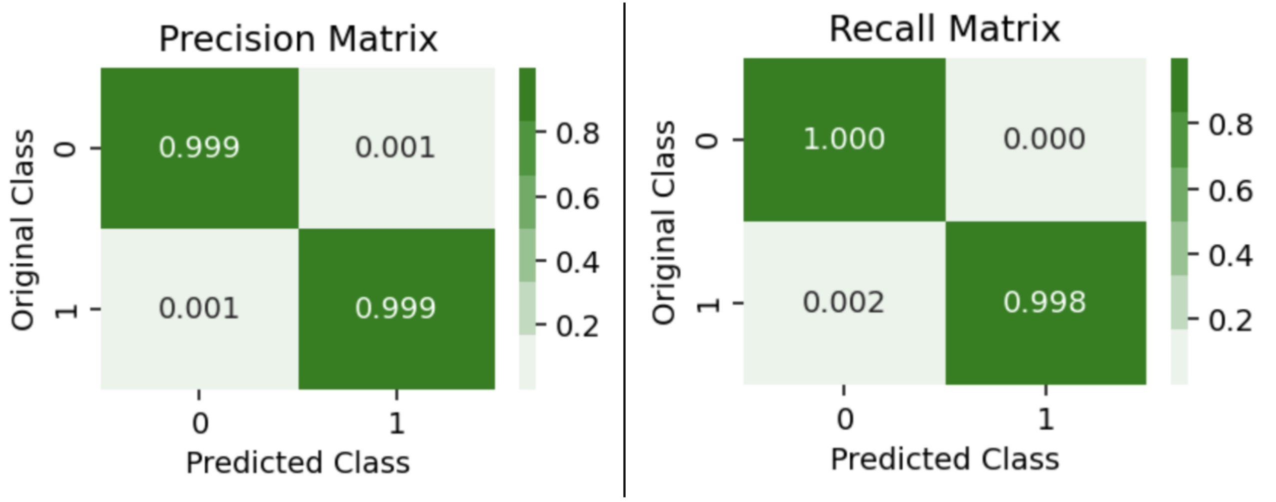
log_loss will be primary metric that we will try to optimise for, precision and recall matrix will help us understand where we misclassified.

Classification using RandomForestClassifier

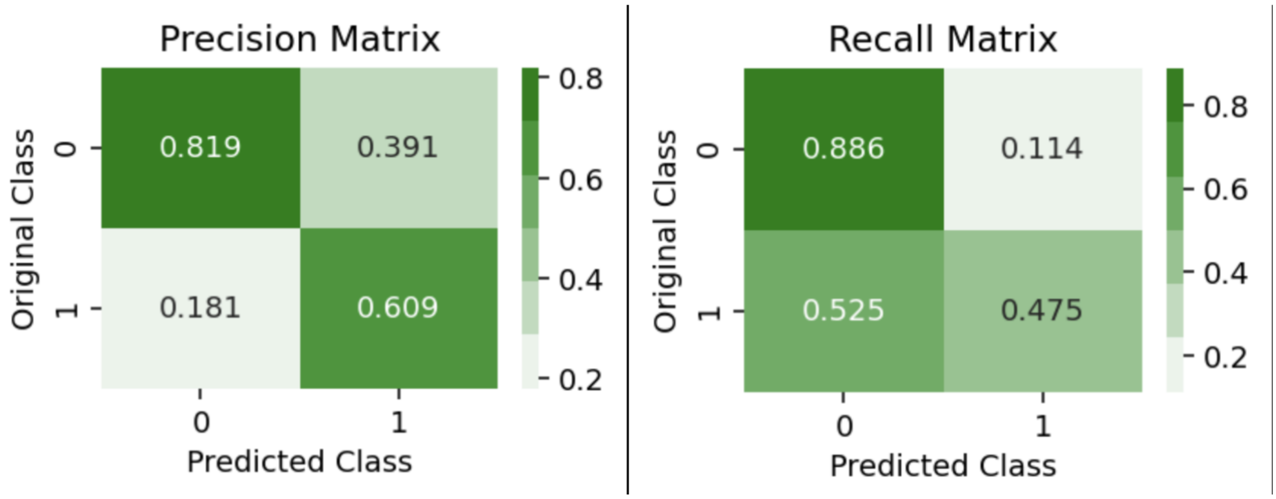
1. Trained a random forest model to predict labels
2. as n_estimators drastically affects generalization of model
3. First this model was trained to give best train validation log_loss by iterating over various n_estimators
4. Once we found out best n_estimators then use grid search on top of this model with best n_estimators

Observations

Train set



Test set



log loss train 0.11384032771799346

log loss test 0.5404848995450136

1. We can see that model is fitting training data very well
2. But is not performing well on Test set
3. We can see that log loss is drastically less than that of random model
   A. log loss train is 0.11
   B. log loss test is 0.54
4. We can see that difference in recall of class 1 in train and test is very high (~52%)
5. [Precision matrix] In test set we can see that almost 40% of predicted positives are misclassified
6. [Recall matrix] In test set we can see that almost 52% of actual positives are misclassified
7. Model was serialised with small improvements in logloss after fine tuning

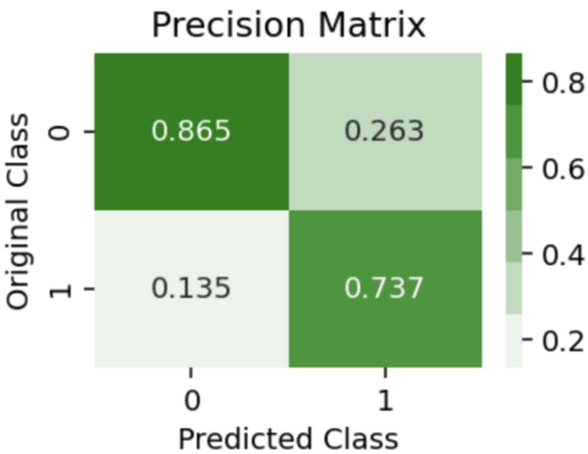
Classification using XgBoostClassifier

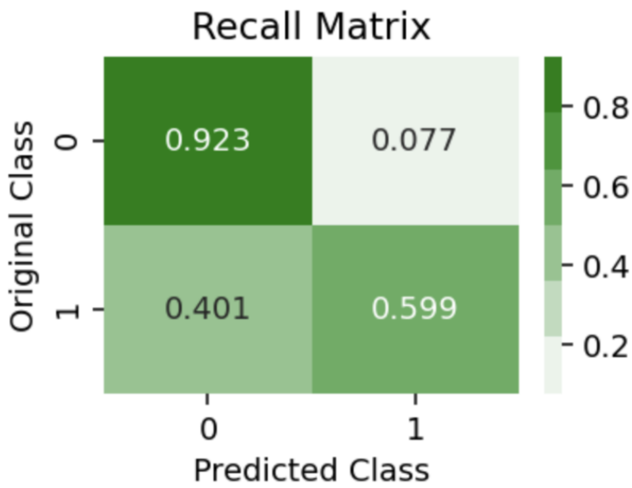
1. Trained a xgboost model to predict labels
2. as n_estimators drastically affects generalization of model
3. First this model was trained to give best train validation log_loss by iterating over various n_estimators

4. Once we found out best n_estimators then use grid search on top of this model with best n_estimators

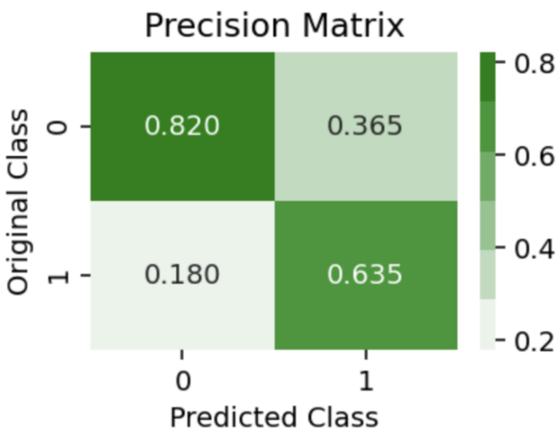
Observations

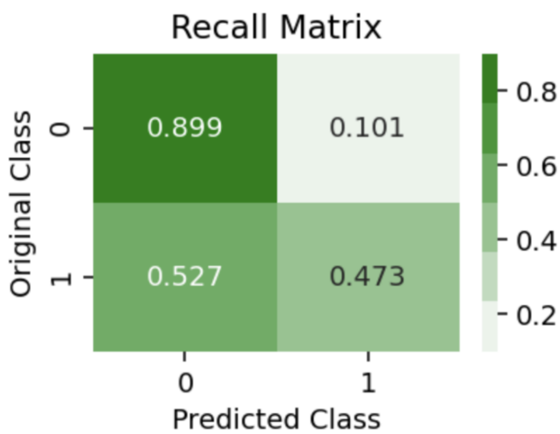
Train set





Test set





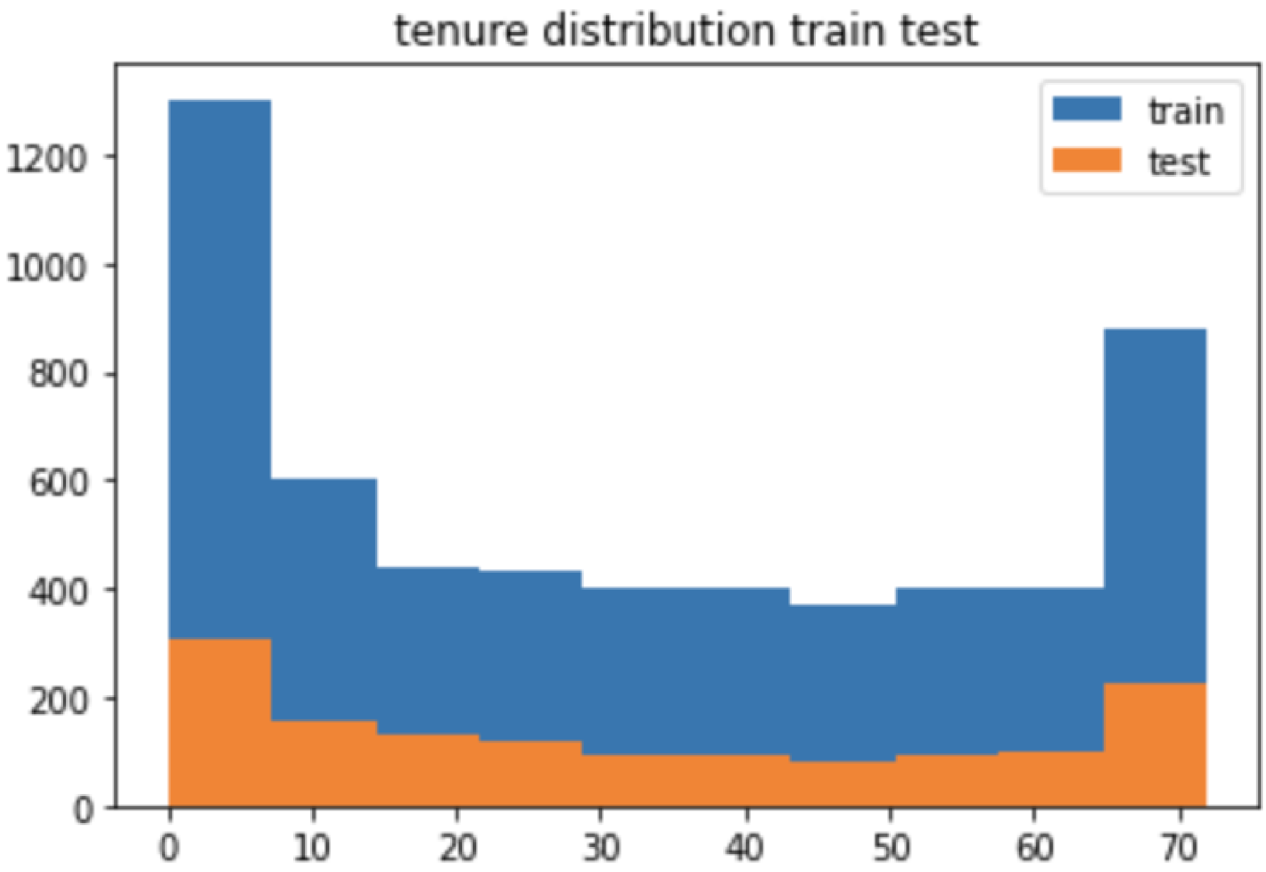
log loss train 0.3509486212673173

log loss test 0.44877092160860893

1. We can see that model is fitting training data very well
2. But is not performing well on Test set
3. We can see that log loss is drastically less than that of random model
   A. log loss train 0.35
   B. log loss test 0.44
4. We can see that difference in recall of class 1 in train and test is relatively low (~12%)
5. [Precision matrix] In test set we can see that almost 36% of predicted positives are misclassified
6. [Recall matrix] In test set we can see that almost 52% of actual positives are misclassified
7. Model was serialised with small improvements in logloss after fine tuning

Setting baseline for regressor

Distributions of target variable `Tenure`



For creating a baseline let us predict random values between min(tenure), max(tenure) with equal probabilities.

- mse random train 1040.17, mse random test 1025.55
- rmse random train 32.25, rmse random test 32.02

Our regression models should atleast perform better that rmse = 32 and mse = 1040

Classification using RandomForestRegressor

Observations

1. Trained a randomforest regressor model to predict tenure
2. as n_estimators drastically affects generalization of model
3. First this model was trained to give best train validation mse and rmse by iterating over various n_estimators
4. Once we found out best n_estimators then use grid search on top of this model with best n_estimators

- mse train 0.73, mse test 5.53
- rmse train 0.85 , rmse test 2.35

1. rmse of random model is around -> 32
2. After training random forest we are getting rmse around -> 2

3. It is almost 16 times better than random model

Classification using XgBoostRegressor

Observations

1. Trained a xgboost model to predict tenure
2. as n_estimators drastically affects generalization of model
3. First this model was trained to give best train validation mse and rmse by iterating over various n_estimators
4. Once we found out best n_estimators then use grid search on top of this model with best n_estimators

- mse train 1.10, mse test 5.20
- rmse train 1.05, rmse test 2.28

1. rmse of random model is around -> 32
2. After training random forest we are getting rmse around -> 2
3. It is almost 16 times better than random model

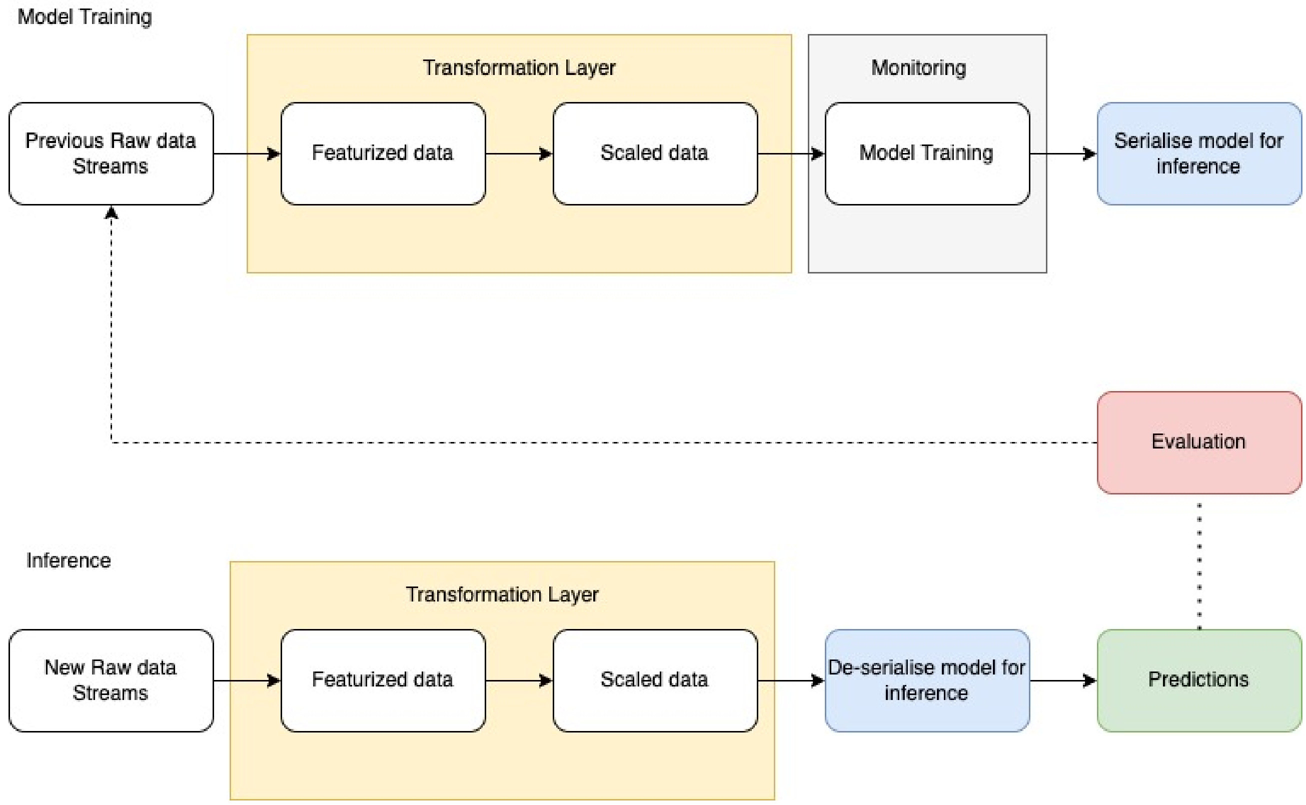
Choosing final models

1. based on log loss and recall % for class 1 xgboost is the better model
2. based on rmse and mse for predicting tenure xgboost is better model by slight margin

How could the losses and metrics be improved?

1. We restricted ourselves to top 6 derived or raw features in this case study
2. By increasing the number of features we feed to our model we can boost the evaluation metrics
3. In order to incorporate multiple features we could also use dimensionality reduction techniques but explanability of model (w.r.t input features) will be gone.
4. In churn prediction identifying variables contributing to churn is important, so that business teams can take actions to alter these variables to reduce the churn

How will the model/s work in production?


Model Training
Transformation Layer
Monitoring
Previous Raw data Streams
Featurized data
Scaled data
Model Training
Serialise model for inference
Evaluation
Inference
Transformation Layer
New Raw data Streams
Featurized data
Scaled data
De-serialise model for inference
Predictions