1. A brief on approach

There were several approach used for this problem statement :

a) Neural Network : This was the first approach where I used a neural network for this problem . The architecture of the neural network included one input layer and two dense layers , last being the output layer .
Relu was used as the activation function for the layers except the last layer where I used the sigmoid activation function.
Model was compiled with Adam optimizer having the learning rate initialized as 0.001. Sparse categorical cross entropy loss was used because the categorical columns were not hot one encoded. They were label encoded as the processing is faster because of less number of columns.
Implementation was in Keras using the Sequential class and not Keras functional API.

AUC score of .50 was achieved using this approach.

b) Random Forest Classifier : This was the second approach . I used H20 automl implemented random forest as well as Random Forest algorithm from Sklearn using gridsearchCV . Both gave maximum AUC score of 0.76 and 0.75 respectively.
Both were initialised with a number of trees as 200 and 300 interchangeably .

c) Ensemble Model : This was achieved using H2O automl , where it tested data with 10 classification algorithms.
Top performing models were :
- Stacked Ensemble with auc 0.8234
- XGBoost with auc 0.8121

d) LightGBM :Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks.

AUC on precision recall curve was achieved to be ~85

Stratified K fold validation was used to validate the dataset along with the training process.

Hyperparameters :
Boosting type : gbdt (gradient boosting)
Objective : binary (for binary classification)

Metric : auc_roc
Learning rate : 0.12345

Light Gradient Boosting Machine model was selected as final model for inferencing

2. Performance of the model :

- AUC : 0.88
- AUC PR : 0.851

3.What data-preprocessing / feature engineering ideas really worked?

a) **Finding the behaviour of the data** : Using pandas describe functionality to get some statistical analysis of the dataset.
b) Box Plots : Plotted the boxplots for all the columns for outlier detection

c) **Outliers** : Outlier removal using Z score statistics .I could use sklearn too but considering a large number of rows , the former was a better option.

d) **Label encoding** : Label encoder all the categorical columns of the dataset using Sklearn Label Encoder with lambda functions for each column because of large data .

e) **Negative Values** : Using the label encoder, some negative values were inserted into the categorical columns because of the presence of NaN. So , to resolve this issue , all the negative values were replaced by a new category number .

f) **Normalization** : All the non categorical columns were normalized by dividing each value with the max value of that column to make uniform normal distribution and faster training . Other way using KBinsDiscretizer was also implemented