Consumer Complaints Resolution

Consumer complaint resolution is important to any business. In this particular case we have been given details consumer complaints along with whether consumer disputed with the conclusion. If we are able to predict this , consumer likely disputed can be given more attention as to how the complaints are handelled as well as how to convincingly convey the final conlusions to them.

Your target here is to build prediction model for column "Consumer disputed"

- Training Data = 'Consumer_Complaints_train.csv'
- Test Data = 'Consumer_Complaints_test_share.csv"

All the column names are sefl explanatory.You need to build your model on Train data . Test data doesnt have response column 'Consumer disputed', you need to predict those values and submit it a csv format. Submission CSV should resemble the file

- Sample Submission = 'sample_submission.csv'

Column names ,value types should exactly match. Also number of rows in the submission csv should be exactly same as test data. If this is not taken care of , your submission will not be graded.

Your submission should have AUC score atleast 0.54 for you to pass. The better you do further you move up in the leader board for this particular project.

Few Suggestions Before you begin:

- Do not use date columns as is , you can use them to create other features. For example which month of the year complaint was filed. Was it first week or last week of the month. How long it took between complaint filing and data being sent to the company. These are just ideas , feel free to make any other features out of these. You can convert strings/object type columns to date_time data using pd.to_datetime. Create cyclic features for date components
- You can handle Consumer Complaint Narrative, creatively. See if you can create some good feature from this column containing text data. [tfidf ?]
- It doesnt make sense to use Consumer ID as predictor.
- While parameter tuning with grid search or randomised search you will get cv performance score . That would be close to what you score might on the test data, but its necessary that it will be always close to that, especially in case if your model is overfitting.
- Before removing NAs from data, do check if there are columns which have too many NaN. See whether you need to impute those values or need to drop that column all together; before you start removing NA obs from the entire data.
- If you are creating any new features on your training data or modifying features in the train; you will have to do that for test data also , in order to use the model which you built on test data for making prediction.
- It doesnt make sense to use ZIP CODES as a numeric variable
- Its a large dataset , might take a lot of time to run
- You can discuss anything on QA forum. Although threads which explicitly disclose too much

information for a solution right away will be removed from QA forum.
- Consider making features for prsesence of NaNs itself

We have also uploaded a benchmark script, it gives you auc score on test data, slightly less than what is required to pass the course. You can include your ideas to make better predictions and make submissions. You can make as many submissions you want if you want to move up the leader board. [We might ask you to submit the script which was used to generate the submission at any time].