Project Mcnulty Write-up – TSA Claims

Goal / Design:

Predict whether a TSA claim will be approved, settled, or denied (i.e. full / partial / no payment). With an accurate predictor, there would be a few uses:

- Passenger / insurance – Predict if a claim will be approved.
- TSA – Partially automate handling of claims, infer areas that can be targeted to avoid cost (or prevent passenger dissatisfaction)

I used accuracy to test the model since the target was pretty well balanced and accuracy should provide a good model for a general use case. However, it would have been best to pick one of these goals and do more nuanced evaluation (e.g. maximize precision / recall / f1 for one of the classes)

Data:

Kaggle dataset ~200k rows & 13 columns. Data was messy with missing values, and inconsistent use of category labels in different years (e.g. “approval” vs “approved” vs “approve in full”).

I would have liked to add passenger volume data but wasn’t able to find useful numbers (split by airport, airline, etc.)

Model:

EDA / Cleaning

- Removed rows with >5 null values since they are missing too much information
- Removed nulls for target & numerical columns (claim value, dates)
- Set categorical nulls to blanks

Feature engineering

- Date features:
  - Delay – [Report Date – Incident Date]
  - Seasonality – Month, Day of month, Day of year
  - Excluded year since new claims will be out of sample for the tree based methods
- Categorical features:
  - Tried dummy variable, frequency rank, and frequency count encoding
  - Frequency count made the most intuitive sense, as it ends up grouping together smaller categories (e.g. airlines with 5 claims will end up with the same “category” encoding)

Selection / Validation

- Feature selection with baseline + various feature combinations
  - Random forest with default parameters, 5-fold CV on 80% validation set
    - Also tried logistic regression, but adding / removing features had very little impact; probably due to non-linear relationships / feature interaction
  - Frequency count did best overall (better than dummy encoding, slightly better than rank)
  - Using all features was best and likely have room to improve with additional features
- Model selection with Date + Categorical variables
  - Models: Gaussian NB, Logistic Regression, Random Forest, XGBoost
  - Also attempted SVM, but training took too long

Testing / Results

- XGBoost was the best model overall
- Slight improvement after grid search via Amazon AWS
- Identified important features for inference using feature importance (by gain)
  - Created tableau charts for class distributions of important features