Filing a Better Claim against the TSA

Overview

The Transportation Security Administration was created in 2002, largely motivated as a change in flight security policy aftre the September 11 attacks. Passing through TSA checkpoints has been routine and ubiquitous in most United States airports since then, as every of the 2 million US passengers that fly daily must be checked at TSA security screening since then. Additionally, all checked baggage is screened by TSA as well. TSA processes 6.9 million bags daily.

Sometimes passengers encounter unwanted incidents when going through the TSA process. They may leave an item behind, suffer damage to property, have an item stolen, or more. As mandated by congressional law, TSA has setup a claims process where passengers may file a claim seeking compensation from TSA for alleged wrongdoing.

TSA provides data of each claim, including the category of incident, date it occurred, date it happened, category of item damaged/stolen/lost, the amount of compensation the claimee is seeking, the decision of the claim, and how much money compensated (if applicable).

While a claim sysem is in place, it is cumbersome and inconvient to complete. 5 pages of paperwork requesting witnesses, insurance information, receipts, and appraisals of lost/damaged items are all requested. For a layperson it is hard to know how likely a claim filed will receive compensation and be successful.

Blog post on the topic can be found here.

Project Design

Upon obtaining the claim data from the U.S. Department of Homeland Security website, extensive cleaning was needed to get the messy data into the same format. Once that was in place, I simplified the model building by indicating any money received from TSA as a "successful" claim, while no money received is "unsuccessful".

Once all the data was cleaned, I engineered a feature which consists of the number of days from when the incident took place to when the claim was actually filed. Additionally, I consolidated some item categories, and reduced the number of airports from ~380 to 38 and airlines from ~170 to 14. Both of those categories were reduced down to the most popular airports and airlines.

The **goal** of this project is to create a good model that accurately and consistently predicts whether a claim filed to TSA will be **successful**. A publicly available version of this app - such as in a website - could potentially save people time in helping them understand their odds of successful before undertaking the effort to file a claim in the first place.

Tools

- Python:
  - Data cleaning: Pandas, Joblib, NumPy, Tabula (extract tabular data from PDF files)
  - Data analysis: Numpy, Pandas, Jupyter
  - Modeling: Scikit-learn, Yandex's Catboost
  - Presentation: Matplotlib, Seaborn, Flask, Bokeh
- JavaScript:
  - JQuery
  - JSON
- Google Slides

Data

TSA publishes all of the claims they have received, some features of the claims, as well as the outcome. The data is posted on their website here. The data is either stored in Microsoft Excel's .xlsv format or in PDF format.

The xlsv files are imported into a Panda's DataFrame. The Tabula library is used to convert and extract the tables from the PDF files into Pandas as well.

There are over 200,000 claims from 2002-2017. After cleaning, parsing, and simplifying the data, I reduced the number of claims to about 93,000.

Additionally, the FAA publishes enplanement data which I gathered. I was also to sort this data to determine the most popular airports, by passenger boardings, in the United States in 2017. I used this as a proxy for TSA passenger throughput as only passengers can go through TSA screenings as a boarding pass is required. This data was used to limit my list of airports from over 370 down to the top 40, which still covered 87% of passenger boardings in the USA in 2017.

Results

I tried numerous classifier model algorithms as well as different variations of the dataset to create a good classifier model. I tried different parameter settings of sklearn's RandomForestClassifier, Support Vector Machines, Decision Trees, Logistic Regression, and Extra Trees algorithms. I also tried XGBoost Classifier and Yandex's Catboost (a Gradient Boosting Decision Tree algorithm). I feature engineered the number of days waited after incident to file the claim, the square of that number, which month, week, and year the incident happened, and how many items were considered damaged in each "claim".

I used Area Under a ROC Curve as my metric of choice since I made my problem into a binary classifier and equally weighed both True Positives and True Negatives. The CatBoost Classifier model algorithm performed best with an AUC of 62.7%. While it stands to be improved substantially, this model is better significantly better than random guessing.

To deploy the model I utilized the Flask library to create a JavaScript and HTML rendering. The rendering incorporates the Vega plotting system to make a visual line plot once the features for the model have been selected by user. This allows users to input the features for their unique TSA case, and see a plot showing them the probability of filing a successful claim depending on how many days they wait after the incident to file.

What I would do differently

To improve the model's performance I believe other features would need to be gathered. Although not published by TSA publicly, the organization does gather information regarding witnesses to the incidents, written statements from claimees, personal insurance information, receipts, and appraisals. The counts of the number of witnesses, TSA-written statements, receipts, and appraisals I believe would be indicative of more successful claims. Additionally, whether it's a full blown NLP analysis of the claimee's written statement or as simple as a word count, I believe this feature would be predictive as well. I would like to gather this data and incorporate in my model.

Additionally, I would like to create a regression model to predict the dollar value one may expect in return from TSA. I simplified the original model by transforming both *Settled* amounts and *Approved in Full* amounts as *Successes* to make it a binary problem. However, this ignores that the TSA sometimes refunds in full, something in between, or zero dollars. Unfortunately, TSA only provides the amount someone requested for a claim in the 2002-2009 period. I would like to still use that information to create the regression model in the future.

Learning JavaScript better and how to use that information with Python would help me on the app deployment part. This was quite frustrating to try to create a Flask app that worked back and forth between Python and JavaScript.

Appendix

Dataset Features

claim_number	date_received	incident_date	airport_code	airport_name	airline_name	claim_type	claim_site	item	status	close_amount
string	datetime	datetime	string	string	string	string	string	string	string	integer