Flight Delay Claims Prediction

Charles Siu

Goal

To predict the claim amount for flight delays

Initial Analysis

- ~900K Rows of data in the training dataset with 10 columns
- is_claim is calculated from delay_time, so it is highly possible that both fields will need to be predicted in the hidden dataset
- Dataset is relatively clean except for NULL values that exist in the Airline column
- Correlation is low between the original features and is_claim, which means additional data is required
- Some airlines and some routes have longer mean delay times than others.

Features Used

Original Features

- Flight_no – Flight number of each flight
- Week – Week of year is the departure date in
- Departure – Location of departure
- Arrival – Location of arrival
- Std_hour – Scheduled departure time, in 24-hour format

Added Features

- flight_date_year – Year of flight date
- flight_date_month – Month of flight date
- flight_date_day – Day of flight date
- flight_date_dow – Day of week of flight
- is_public_holiday – Is flight date a Hong Kong public holiday?
- mean_pressure – Mean air pressure
- mean_temp – Mean temperature
- mean_dew_point – Mean dew point temperature
- mean_humidity – Mean humidity
- mean_cloud – Mean amount of cloud
- total_rainfall – Total amount of rainfall

Approach to Model Training

- Determining the Target Variable – I decided to do modeling on delay_time because it seems to be a more quantitative measurement and a good proxy for is_claim
- Data Cleaning – Fix the format and fill NULL values
- Feature Engineering – Derive new features from existing features as well as retrieve external data
- Data Encoding – Map categorical variables to integers and map numerical variables using Standard Scaler
- Train/Test Split – Split the original dataset into a 80% training portion and 20% validation portion to simulate making predictions for the hidden dataset

Possible Improvements

- I have jumped straight to a neural network-based model without first experimenting with a linear or tree-based model. I could spend some time with those models and see if there are any improvements to the model performance.
- The model I have trained can only produce validation R-squared values at around 0.36, which is relatively low compare to the standard 0.8 or above.
- There are possibly other ways to store model files, but for the sake of time and easy sharing, I have saved them as individual files on disk.
- Although I have mentioned the possibility of imbalanced dataset in the EDA notebook, I never got around to implement an upsampling procedure during my model training process.
- It is possible to use Python scripts (.py) rather than Jupyter Notebooks, but the changes in Python scripts are more difficult to realize and to debug.
- In the HKO Weather Data, there is a column of wind speed and wind direction but I did not include it in the list of features due to difficulties in parsing. Including that feature might improve model performance.