Final Report - Ramp Challenge

predict the number of air passengers

Jianing CHEN, Qiang Zhu(Team name: kiten/Derek)

1. Introduction

Given the limited datasets about early booking time, departure, and arrival information, our goal is to predicting air passengers. Obviously, we need more external datasets to more preciously interpret the target variable('log_PAX').

We first consider the influential factors from the inherent characteristics of airports, and then think about potential economic factors, demographic factors, geographical factors and time factors. As we have been provided with the comprehensive details about the weather for corresponding days, we collect data on airport characteristics, including total airline operations, departure and arrival situations, routes arrangements and operations between different airport pairs, including on-time flights and delays, etc. All data are collected from the Federal Aviation Administration (FAA)'s databases. In addition, we consider data about the economic and demographic conditions, including GDP, population, personal income from the city or state perspective, etc., which are obtained from Kaggle website and the Federal Reserve Bank. As season changes and an upsurge in flights due to holidays, we have also taken the major public holidays into account, which is directly provided by python library. Appendix A shows a list of data sources.

2. Data Preprocessing and Model Selection

Firstly, we merged all external data through the keys 'Airport' and 'Date' into a single file. For better explanations, we used daily data on air traffic and transformed the departure and destination airports with different prefixes in order to add condition variables of pairs, then turning the number of years, months, days, weekdays and weeks into a series of 0-1 dummy variables. Especially, we computed the next workday and the last workday to represent the leisure time, as well as distances between different pairs of airports by longitude and latitude data to represent the flight distance. For the economics data, we considered the GDP and census data in different years of both the corresponding state and the city, for the reason that most of the airports are at state capitals. Extreme weather indicators that might lead to flight delays or cancellations like minimum temperature, maximum wind, visibility and so on were also added. and the 'event' variable was also changed into dummies columns, including thunderstorms and snowstorms. When dealing with missing values, we use the mean or 'ffill' (like for oil price) method corresponding to the actual meaning of the variable to fill the missing values. Then, we did experiments with several different approaches, such as Multi-regression, Random Forest, neural network and etc.

linear regression and Random forest

With respect to Multi-regression, we tried Linear regression, Ridge, and Lasso from Scikit-Learn packages that provide automatic feature selection. The Average RMSE remains above 0.43, which indicates that simple linear regression may not take effect here.

Neural Network

Afterward, we tried a feed-forward neural network model. By slightly changing the final dimension of the output layer, we could turn the Neural network from the classical classifier to a non-linear regression method with one output at the final layer to do the regression issue. At the very beginning, we used only one hidden layer and choose the ReLu function as the activation function and Adam as the optimizer. The main parameters are the batch size and loss function, for which here we choose 64 and mse respectively. The RMSE result is more than 0.5, which acting not pretty well. We added more layers and increased the batch size to 128, as a smaller batch-size tends to decrease the generation error. However, RMSE remained above 0.4, even though with more layers and parameter tuning. Taking into account the limits of this simple structured neural network, and also the fact that it will be difficult to interpret the final models with multiple layer neural networks, we decide to switch to another model.

XGBoost

We thought about using boosting algorithms since we have hundreds of features in total, while boosting can integrate weak learners into a powerful learner by learning higher-order interactions between features. As one of the most powerful boosting algorithms, XGBoost has the training process that every iteration tries to fit the residual of the previous k-1 base learners, so as to enforce the newly base learner move greatly towards the direction of steepest gradient-descent, which lower RMSE for sure. Besides, XGBoost adds regularization, which can help prevent overfitting problems.

First, we tried the default parameter using the XGBoost package in Python, where we got RMSE lower than 0.4. Accordingly, we decided to do the feature selection by ranking all the features with recursive feature elimination and cross-validated selection (cv=3) of the best number of features. We finally got 56 parameters out of 208 parameters. By using departure and destination as parameter prefixes, the parameters all appear in pairs. Luckily, we got a nearly symmetric parameter for two sides. Selected features include time period (weekday, week, month, days, next workday, last workday, etc., air travel and airport analysis data (total operations, actual departure and arrival amounts, delayed arrivals, etc.), economic and demographic factors (GDP, populations, birth and death rates, etc.) and geographic attributes (longitude, latitude, elevation, distance).

After feature selection, we decreased our RMSE by around 0.05.

3. Parameter Tuning

We first looked for a reasonable n_estimator with other fixed parameters. From the previous loss curve, we knew that the XGBoost method might rush to over-fitting if using inappropriate parameters of base learners. So we fix max-depth at 7 and min_child_weight at 5. We choose n_estimator within the range $[500,2000]$, step size=100 to compute the RMSE. From the computation, we can find that the RMSE will converge when the n_estimator is larger than 1700. So we first set n_estimator to 1700. As to the learning rate, we conduct the same things as to n_estimator. We get the optimal learning rate for our cases, which is 0.05.

Thirdly, we focused on other parameters including ‘max_depth’ and ‘min_child_weight’. The former limits the depth of decision trees and the latter prevents an arbitrary node with few samples from being further split. We used the grid search method to get the optimal combination: max_depth of 6 and min_child_weight of 3. We tried to tune other parameters including regularization alpha and lambda, and gamma using the same approach.

Regressor type	**Cross-validation Score**
Lasso Regression	0.5482 +/- 0.0144
Random Forest (60 estimators)	0.4340 +/- 0.0267
Random Forest (300 estimators)	0.4310 +/- 0.0257
Neural network (batch size=128)	0.4736
XGBoost (n_estimator=1700)	0.3317 +/- 0.0174
Final Submission of XGBoost	0.265

4. Result Interpretation

Most sophisticated models are criticized for their bad interpretability as to a single feature, especially for our prior trial in Neural Network. When it comes to the XGBoost with the decision tree as a base learner, we can even plot a single tree easily when the max depth is smaller than 4. In our case, It may not simple to give a comprehensive plot towards that. Fortunately, Python offers us a tool to plot the difference and ranking in our model. (see **Figure1.1**)

The graph shows clearly that the most powerful predictor was the Weeks to Departure for a designated flight. This was followed by Total Operations of the airport, Days and Weeks of the airline, which at the same time represent leisure time difference. From the model we have chosen, the time metrics generally works as a strong indicator of airline passengers. The closer to the leisure time, the more passengers take airplanes to. What we did not expect though was that Altitude played a role and that a lower altitude was favored by passengers. Also, socioeconomic conditions generally have a moderate impact on the results. Except for distance, geographical features make a difference in final results but not too much.

As to a pair of datasets, we can conclude that the airlines between two prosperous, indicating with respect to GDP and Population earn more airline passengers. A longer distance between the two cities also contributes to a higher number of passengers. It seems that our model is consistent with our simple instincts.

In all, the external data and XGBoost model have successfully lowered the RMSE to 0.265. Given the comprehensive external data and multiple parameters tuning, we get seemingly reasonable features and thus an adoptable way to future prediction.

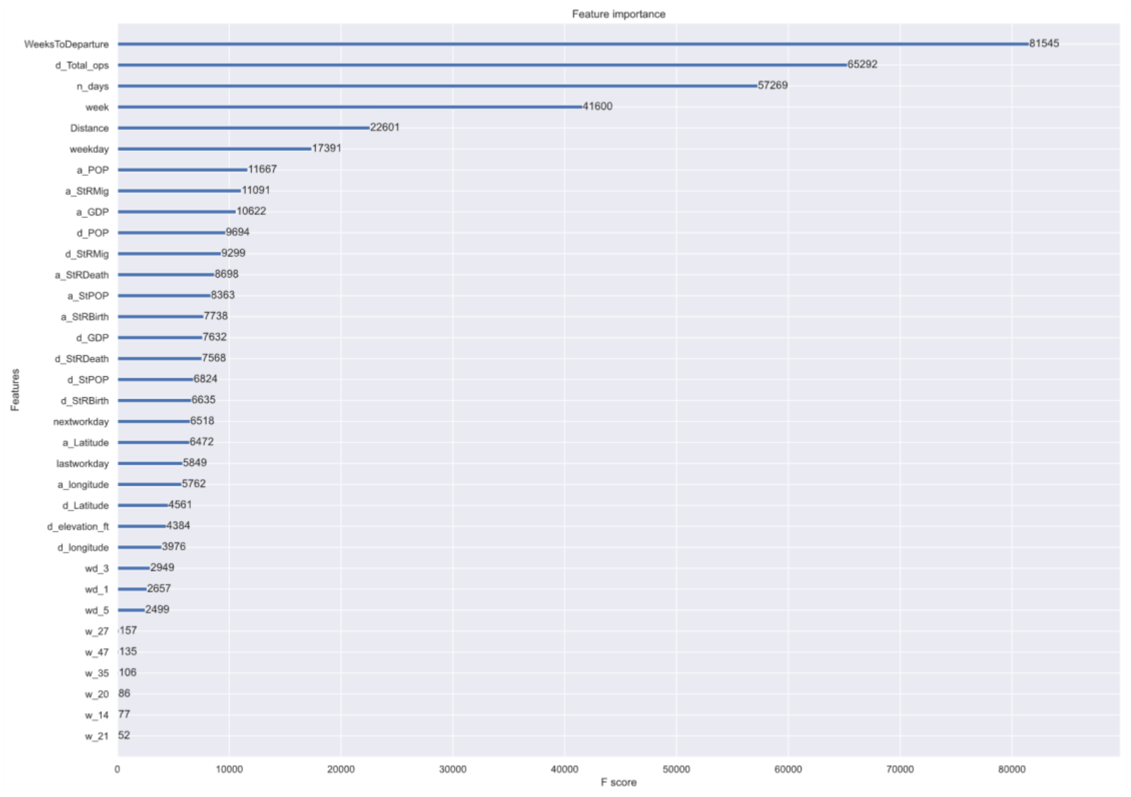

Figure1.1 Feature importance

5. Model Review

In this final project, we replenish the thin datasets after brainstorming in aspects of socioeconomic, airport details and etc. After experimenting with different models from Scikit-learn and Keras, and methods to do feature selection, we have chosen the XGBoost as our starting point to improve.

Finally, after parameter tuning, it seems that our feature sets are good and possess high prediction accuracy. However, XGBoost is a time-consuming way, especially when max depth reaches 5 or more, although it should save more time than most boosting algorithms from a general understanding. When conducting feature selection, we tend to spend hours running the code, finding no way to improve the process with limited computing resources. And we lack methods to visualize and interpret the results more preciously, especially on the real effects of variables.

At the same time, we failed in neural networks, though its potential. Despite its bad interpretability and constraints in normalization, we have not gotten the good RMSE in it.

Appendix A - Detailed information about selected features

*d refers to departure and a refers to arrival. d/a means the feature has two value in a record to indicate information about departure/arrival airport.

Feature name	type	Pair	Description
WeeksToDeparture	integer		the weeks to the departure date when the passenger booked the ticket
Min_TemperatureC	integer	d	the minimum temperature of the date at the departure airport
Total_opearations	integer	d/a	Total operations of Flights in a designated airport
Total Delays	integer	d/a	Total delay time for a specific day
Actual Departures	integer	d/a	the actual departures that the corresponding airport had at the corresponding date
Actual Arrivals	integer	d/a	the actual arrivals that the corresponding airport had at the corresponding date
Delayed Arrivals	integer	d/a	the delayed arrivals that the corresponding airport had at the corresponding date
Division	dummy	d/a	the division the state lies within, according to the 2010 census. 1 = New England, 2 = Middle Atlantic, 3 = East North Central, 4 = West North Central, 5 = South Atlantic, 6 = East South Central, 7 = West South Central, 8 = Mountain, 9 = Pacific
StPOP	integer	d/a	the population of the state which the airport is located
StRBirth	float64	d/a	the birth rate of the state which the airport is located
StRDeath	float64	d/a	the death rate of the state which the airport is located
StRMig	float64	d/a	the migration gate of the state which the airport is located
elevation_ft	float64	d/a	
GDP	integer	d/a	the GDP of the city which the airport is located
POP	integer	d/a	the population of the city which the airport is located
RPI	integer	d/a	The average real personal income of the city at the corresponding year
latitude	float64	d/a	the latitude of the corresponding airport
longitude	float64	d/a	the longitude of the corresponding airport
next workday	integer		measure the days to the next workday
last workday	integer		measure the days from the last workday

distance	float64		the distance between the departure and the arrival airport, measured in KM
month	ordinal		from 1 to 12, represent the month in a year
day	ordinal		from 1 to 31, represent the day in a month
n_days	integer		number of days count from 01/01/1970
[weekday] ‘wd_1’, ‘wd_3’, ‘wd_5’, ‘wd_6’	integer		wd_x refers to the x^{th} day in a week
[week] ‘w_1’, ‘w_14’, ‘w_20’, ‘w_21’, ‘w_27’, ‘w_35’, ‘w_47’, ‘w_52’	integer		w_x refers to the x^{th} week in a year

Appendix B - Resources of external data List of external data sources

Airport Analysis of Designated Airports	FAA: https://aspm.faa.gov/apm/sys/AnalysisAP.asp
Oil Price	Kaggle: https://www.kaggle.com/mabusalah/brent-oil-prices
Airport Operations between Designated Airports	FAA: https://aspm.faa.gov/opsnet/sys/Airport.asp
Delay Information about Airlines	FAA: https://aspm.faa.gov/opsnet/sys/Delays.asp
Airport Coordinates	https://github.com/datasets/airport-codes
Energy Census and Economic Data US 2010-2014	https://www.kaggle.com/lislejoem/us_energy_census_gdp_10-14
Real Personal Income for States and Metropolitan Areas	https://apps.bea.gov/regional/histdata/releases/0615rpi/index.cfm
GDP and Personal Income	https://apps.bea.gov/itable/iTable.cfm?ReqID=70&step=1