Delay and preference based flights recommendations

Rodrigo Hanuch
rihanuch@uc.cl
Pontificia Universidad Católica de Chile
Santiago, Chile

Enrique Waugh
ewaugh1@uc.cl
Pontificia Universidad Católica de Chile
Santiago, Chile

Sebastián Ricke
sricke@uc.cl
Pontificia Universidad Católica de Chile
Santiago, Chile

Abstract

In today's world, recommender systems are a fundamental part of the choices we make while shopping online, and normally, the more expensive an item, the more important the recommendations is for a user. In this same way, nowadays, airlines are recommended, to the best of our knowledge, mainly by the price, without having any consideration for the user's preferences and the possible delays of a determined airline and flight.

This allows much room for improvement. Some users might be very interested on the quality of the food or overall service, while others might prefer seat comfort. Even more, for a user that has a tight schedule (maybe has to catch another flight), punctuality might be crucial. All of this can be modeled and taken into account when building a flight recommender system to maximize customer satisfaction and perceived airline quality [4].

This leads to two problems: airline recomendation and delay prediction. In the following work, we build a recommender system based on user's preferences and flight classification in conjunction with a flight delay prediction model. Afterwards, these models are joined to make a recommendation based on the user's preferences and the probability of having a delay in the flight. We will present the datasets that were used, methodology, evaluation and finally conclusions and future work.

Keywords: airlines, fastfm, recommender system, slim, neural networks, delay prediction

1 Introduction

In this paper we contribute to the determination if a recomender system can give better recommendations given a user's relevance factor of having a delayed flight and the probability of such delay occurring. This recommendations can be particularly impactful on users that travel often and want to minimize their delays to prevent the loss of connections and their loss of time, which, according to Efthymiou [4], can affect the way a user perceives a company.

Bearing in mind the previous idea, we hypothesised that a personalized recommender system working in conjunction with a flight delay predictor can generate better recommendations for the passengers to which airline they should take. We will build separately a classic personalized recommender system and a deep learning model to merge them afterwards in our dataset.

The structure of the rest of the paper is as such. First we present a dataset explanation, followed by the methodology used in our work. Later, we show the results (in which we compare the recommendation task before and after the introduction of our predictions), and finally, we present conclusions and future work.

2 Datasets

To build our main recommender system we used two sources. One dataset for the "classic" recommendation task and another dataset for the flight delay prediction. Below we will describe both of them.

On one hand, the dataset used for the airline recommendation [6] consists of 27284 reviews from different users that flew in one or more airlines, with a breakdown from each component of the main score that was given to the airline and an overall score of the flight. This dataset contains airlines such as South West, Delta, American Airlines, SkyWest, United Jetblue, etc.

On the other hand, the second dataset [3] consists of all the flight delays in the U.S. of 2015, having approximately 6000000 flight records. All airline carriers have a similar number of flights, but their delays vary according to figure 1.

Something worth mentioning about this dataset is that on it we can find a class imbalance. This produced very poor results in the preliminary classifier testing, so this needs to be handled in order to produce a good delay predictor.

Another important factor to consider is the fact that the intersection of our datasets produces a reduced space to work with (airlinewise). This mainly due to the fact that our first dataset consists of global airlines, while our second dataset has a U.S. focus. Nevertheless, these datasets were chosen as there is a scarcity of user-rated airlines that have a direct rating insted of having to do an interpretation on *tweets* or sentiment analysis.

3 Methodology

The way the recommender system was built was by utilizing two approaches for the classic recommending task and a deep neural network for the flight prediction problem. For the classic recommendation problem, we tested FastFM [1]

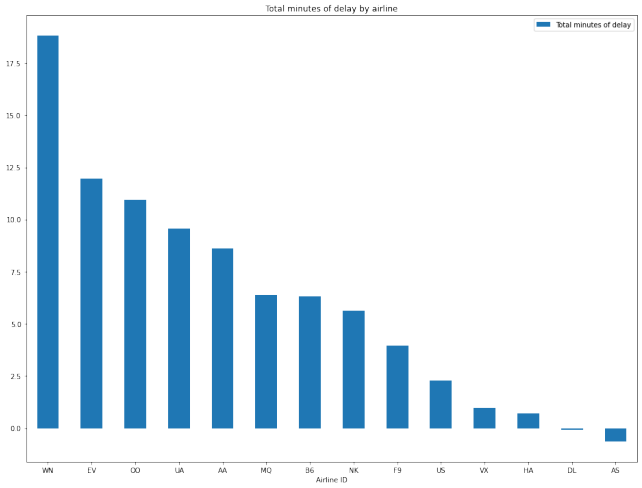

Figure 1. Total minutes of delay by airline.

and SLIM [5], with FastFM being used in a single and multi style approach.

One main disadvantage of using SLIM compared to FastFM in being restricted to using only the rating given by the user and not adding a multi style approach. Naturally, our idea to add some context information with the inclusion of flight delay information fits better with a multi style approach given by FastFM.

For the flight delay problem, we built a simple linear deep neural network with customized input layer for each attribute. This was done to preprocess the attributes in a customized way. The details of the implementation are presented in section 3.4.

The main reason why we had to build a model to predict the flight delay was because, as described in the previous section, we did not have the information of the flight delays in the user ratings dataset. After having the three base models, we predicted the delays for the user's flights and reincorporated these predictions as a parameter in our first dataset in conjunction with a weight of the importance that a user gives to delays that was inferred from the data.

3.1 Importance of delay

We interpreted the relative importance of any delay for a user as equation 1, where δ_{Mi} is the global minimum and δ_{Ma} is the global maximum, of all δ_{M_u}.

$$\delta_{R_u} = \frac{\delta_{M_u} - \delta_{Mi}}{\delta_{Ma} - \delta_{Mi}} \tag{1}$$

At the same time, in equation 2, $OR_{u,f}$ represents the overall rating, and $AR_{u,f}$ represents the real average rating, each for a given user u and a particular flight f. Finally, F_u represents all the flights of a particular user u.

Table 1. Optimized FastFM

Type	MSE	NDCG@5	MAE	L2W	L2V
Single	1.8598100	0.8611048	1.1247318	0.56	0.55
Multi	2.1772958	0.9020624	1.2037653	0.44	0.43

$$\delta_{M_u} = \frac{\sum_{f \in F_u} OR_{u,f} - AR_{u,f}}{|F_u|} \tag{2}$$

The interpretation of this data normalization is that $\delta_{R_u} \in [0, 1]$, where 0 represents no importance and 1 represents absolute importance of a flight delay for a particular user.

3.2 FastFM

Two variants of FastFM were used, single and multi style approaches. For the single style approach the style was defined as `cabin_type`, while for the multi style strategy the styles were defined as `air_date` and `cabin_type`.

The premise of using FastFM as a benchmark is that it allows us to introduce arbitrarily many styles as we want, which, in theory, fits well with our problem as we have multiple possible styles to use. In this way, the main constraint of the problem will not be the method, but the dataset used, which has two main styles, that are the ones that we used to test this recommender system with.

We optimized our parameters based on NDCG@5 by trying to get as close as possible to IDCG@5 as there are not many airlines that are both in the prediction and recommendation datasets, so, in this context, being accurate is the most important parameter for the recommendations. This can also be extrapolated to a real world system, as although there are many airlines, users tend to stick with the ones they most like.

We tested our parameters in two steps with local optimization and with a graphical approach (figure 2) to get the best parameters within 2 decimal places to finally get the best ones via a sorted list. As we can see in table 1, there is not much difference in using multi style FastFM in terms of MAE and MSE, while there is a minor gain in NDCG@5.

The presented preliminary results of table 1 were expected as FastFM without any kind of contextual information (style) will tend to underperform in comparison with an implementation that gives context about the items. This can bee seen our results as the multi style approach benefits in terms of NDCG@5, but at the expense of MSE and MAE.

For the incorporation of the predictions into FastFM we created a new column for the rating that would be given to a flight in terms of delay[1]. This delay rating is given by:

$$RD_i = (1 - \delta_{R_i})(1 - PD_i) \tag{3}$$

[1]Bigger numbers are a better rating.

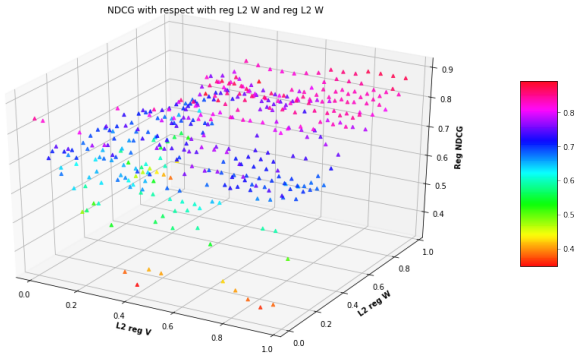

Figure 2. First iteration of graphical method for multi style FastFM optimization.

In equation 3, RD_i represents the entry i of the dataset, δ_{R_i} represents the relative delay of the corresponding user of entry i and PD_i represents the predicted delay (in terms of probability) for entry i. After generating the column, the entries are normalized with min max and multiplied by five to get a comparable rating with the existing ones of other columns. This can be represented by RD'_i, as the following equation:

$$RD'_i = 5 \times \left(\frac{RD_i - RD_{Mi}}{RD_{Ma} - RD_{Mi}} \right) \tag{4}$$

In equation 4, RD_{Mi}, represents the global minimum and RD_{Ma} the global maximum, of all RD_i.

3.3 SLIM

These methods aim to obtain the low-density user-item matrix A by l1- norm and l2-norm optimization. For the implementation we used a more standard approach that can be found in Github [2], and we filtered ratings higher than 3 to obtain better results.

Following the same procedure as in FastFM, we optimized L1 and L2 for SLIM based on NDCG@5 for consistency. Through a standard grid search and local optimization of these parameters we ended with the results shown in table 2. As expected, SLIM underperformed compared to FastFM since the last one adapts better to this context, though we didn't expect such a substantial difference.

For the incorporation of delays into SLIM we followed the same steps 3 and 4 as in FastFM to obtain the delay rating. Since SLIM takes a single rating to train, we took the equally weighted average between the average rating the user gave and the delay rating obtained. It is worth mentioning that we run another hyperparameter optimization after incorporating delay ratings, where we obtained L1 = 0.001 and L2 = 0.0001.

Table 2. Optimized SLIM

NDCG@5	MAE	MSE	L1	L2
0.4288816	3.1829868	11.1702993	0.001	0.0046

3.4 Flight delay prediction

To tackle this problem we builded a linear deep neural network architecture. This was composed by:

1. A single input layer for each feature
2. Only for the string-type features, a transformation to numerical categorical value.
3. A single personalized encoding layer for each feature that transforms the numerical categorical value of the input into a one-hot encoded version of the feature.
4. A concatenation layer that joins all the previous information into a single one-dimensional layer.
5. A dense layer of length 256 and Relu activation function.
6. A dropout layer with a probability of 30% on each position.
7. A second dense layer of length 256 and Relu activation function.
8. A second dropout layer with a probability of 30% on each position.
9. A last dense layer of length 1 and softmax activation function that acts as output layer. It's interpretation is the probability for flight delay.

The architecture also contemplates and deals with the class imbalance present in the dataset (aproximately 20% of the examples are delayed flights) with class weights. The weight of each class is defined as follows:

$$w_a = \frac{N}{count(a)} \tag{5}$$

Where w_a is the weight of the class a, N is the total number of examples on the dataset, and $count(a)$ is the number of examples belonging to class a. Figure 3 presents a graphical representation of the architecture.

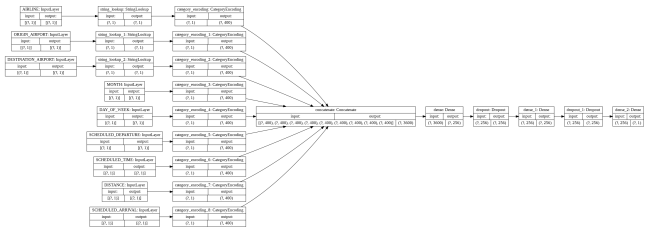

Figure 3. Deep Neural Netwotk Architecture

The baseline used in this sections was a simple probability functions, calculated as:

$$p_a = \frac{delays_a}{total_a}$$

Where p_a is the delay probability assigned to airline a, $delays_a$ is the number of delays registered for airline a, and $total_a$ is the total number of examples for airline a.

This baseline performed poorly because of the class imbalance present in the dataset, with a high accuracy score, but predicting only non delayed flights.

4 Evaluation and results

On the flight delay prediction, we split the dataset into separated train, validation and test sets (70%, 15% and 15% respectively). Our model performed better than the preliminary models, raising accuracy from 61% to 73%. Other metrics like precision and recall were improved as well. This being a good improvement over the preliminary models, we must say that the overall performance is not satisfactory. For a model to be good at the delay prediction task, it would need an accuracy score close to at least 90%. Putting that aside, we still achieved positive results on improving the basic personalized recommender system using the output of this flight delay prediction model.

For the prediction with FastFM was made through a test-train relation of 20-80, in which the optimized NDCG@5 score was of 0.9162867. This poses a gain in terms of precision in NDCG@5 with respect to our original score, but at the same time this is achieved by trading off precision in terms of MSE and MAE. This exchange of MSE with NNDCG@5 presents decreasing gains, as each unit of additional NDCG@5 comes at the cost of a bigger overall MSE and MAE error increase.

The results we got from FastFM, which are presented in table 3, were not as good as we expected, as we only managed to increase our NDCG@5 score by 1,57% with respect to our multi style approach without prediction. In this sense, our results are inconclusive, as this increase in score is not enough to simply attribute it to the nature of the dataset, specially considering its small size and the possibility that there may have been overfitting of the model.

Another thing to consider about the FastFM result is the fact that the increase of MSE and MAE with respect with the increase of NDCG@5 is not acceptable enough. This is mainly due to the fact that the gain made was not proportional to the loss obtained, so the implementation of this model in a real world system would, overall, decrease precision if the user did not like the first 5 recommendations that are given.

In the case of SLIM, we found a similar improvement as in FastFM where we managed to improve our NDCG@5 score by 2,9%, as seen in table 3. Our conclusions from this small improvement are similar as in FastFM where we find that this small improvement is inconclusive given the nature of the dataset, though given that we find similar improvements in two different models we gain confidence of our approach to this problem and expect to gain better results if the quality of the dataset is improved, but we will go into further detail in the next section.

Table 3. SLIM & FastFM with (w) and without (wo) delay prediction.

Type	NDCG@5	MAE	MSE
Most Popular	0.0885310	-	-
SLIM wo/p	0.4288816	3.1829868	11.1702993
SLIM w/p	0.4413291	3.0817207	10.3539178
FastFM multi wo/p	0.9020624	1.2037653	2.1772958
FastFM multi w/p	0.9162867	1.2182207	2.2300922

If we compare MAE and MSE metrics in table 3 between SLIM and FastFM we can see a clear difference in performance favoring FastFM, though given that SLIM is a top-N recommender system, and that the user generally looks at the first recommended items, NDCG@5 is a better metric to compare both.

One important result is the substantial difference in accuraccy between FastFM and SLIM where we corroborate our hypothesis that the first one would perform better in this situation because the context information given by inclusion of flight delay fits better with a multi style approach.

5 Conclusions and future work

As seen by our results, our recommender system didn't perform as expected and did not result in enough gain in terms of precision that can be solely attributed to the model and not the dataset. Though we believe that adding flight delays information as context for a recommender system is a good path to improve flights recommendations, the main constraint we faced in our task was finding the right datasets. Naturally the best case scenario would have been to have for each rating additional information about the specific flight concerning its arrival on time or not.

Another possible approach would have been, instead of using a prediction model, to use only the average flight delay per airline, but we were also faced with the problem that we only had this information for a small portion of the airlines in the ratings dataset. Because of this, a prediction model appeared as the best solution, but because of the inaccurate predictions in our delay model this was translated into a big gap of information to be filled, that naturally led to poor results in our combined recommender system.

The last point could be, potentially, solved by using a bigger dataset. This data would have to be obtained via talks with airlines to gain access to some of their data or to do one ourselves with online surveys and user approval. Both paths are difficult as, on one hand, as it is very probable that airlines will guard their data as industrial secrets, and on the

other, the user form will take a long time to build a decently sized dataset.

Because of this, our plan for future work would be to achieve better synergy between our ratings dataset and flight delays. Since the last one includes airlines from all over the globe, the best way to achieve this would be to get delay information from airlines within the U.S. Once achieved this, we could replicate the steps described in this paper to compare results. Since we faced problems achieving good precision results in the flight delay prediction model, we recommend as a first approach, to use the average delay information per airline, but also, look further into improving the delay prediction model performance, and comparing both results.

References

[1] Bayer, I. fastfm: A library for factorization machines. *CoRR abs/1505.00641* (2015).

[2] Bidart, R. Slim and sslim recommendation methods - toy implementations. https://github.com/ruhan/toyslim, 2017.

[3] DoT. 2015 flight delays and cancellations. Electronic, https://www.kaggle.com/usdot/flight-delays, February 2017.

[4] Efthymiou, M., Njoya, E. T., Lo, P. L., Papatheodorou, A., and Randall, D. The Impact of Delays on Customers' Satisfaction: an Empirical Analysis of the British Airways On-Time Performance at Heathrow Airport. *Journal of Aerospace Technology and Management 11* (00 2019).

[5] Ning, X., and Karypis, G. Sparse linear methods with side information for top-n recommendations. In *Proceedings of the Sixth ACM Conference on Recommender Systems* (New York, NY, USA, 2012), RecSys '12, Association for Computing Machinery, p. 155–162.

[6] Patel, A. Airline dataset mining. Electronic, https://www.kaggle.com/arjhbholu/airline-dataset-mining, February 2018.