IAD Flight Delays

Leo Wang
Computer Science
University of Virginia
Charlottesville, U.S.
yw7uc@virginia.edu

Alicia Wu
Computer Science
University of Virginia
Charlottesville, U.S.
yw7vv@virginia.edu

Simon Zhu
Computer Science
University of Virginia
Charlottesville, U.S.
mz4cr@virginia.edu

***Index Terms*—flight, delay, airport, airline, weather**

I. Abstract

For the sake of enhancing international students traveling experience, this project aims to establish the optimal model that classifies flights in terms of delay time. The three categories are "No Delay," "Slight Delay," and "Heavy Delay." As this project mainly concerns international students in UVA, we select Dulles International Airport (IAD) as the research object. The data to be used originate from two datasets, one describing daily weather conditions around IAD and the other describing flights departing from IAD.

We then conduct feature engineering by eliminating irrelevant features and combining to create new features such as "Frozen." We also add a label for each entry. After appropriate data cleaning such as undersampling the data to have a more balanced dataset, we split 20% of the data into test data and the rest into training data. Then we fit the training data on several models such as Linear Support Vector Machine Classifier and Softmax Regression, Random Forest, K-Nearest Neighbors, finetune the hyperparameters with cross-validation, and then evaluate each model with the test data in terms of the accuracy and the confusion matrix. In this way, we are able to find the best-performing model. We then analyze the results to see whether it fulfills our stated purpose and offer some ideas about future works.

II. Introduction

Flight delay has been a nuisance bothering travelers for years. For example, almost every traveler has experienced desperation when waiting for their flight in Chicago OHare Airport for 3 hours in snowy winter. Likewise, such nuisances have also bothered UVA students, namely the international students who fly frequently between home and UVA. Therefore, as international students in UVA ourselves, we wish to create a model through machine learning that classifies flights in terms of delay time, depending on features such as airlines, weather conditions, departure time intervals, distances and so on. In brief, we desire to establish an application that classifies flights in terms of delay time.

Before checkpoint, we have tried the softmax regression model and the linear SVM model, and used accuracy rate and confusion matrix to evaluate model. After that, when we found that the precision rate and recall rate of both models are very low, we figured that the imbalance of the dataset led to this problem. After some research, we decided to drop out some of the data and try some other classification models. We also use Random Forest and K-Nearest Neighbor model to improve the performance of our model. We end up with KNN model since it delivers the best result.

III. Method

During the data cleaning process, we conducted feature engineering, namely feature selection and extraction. From the original data, we select the features that will significantly affect the delay status of a flight, namely the airline name, the departing time, the distance of the flight, the amount of precipitation of rain (PRCP), the amount of snow (SNOW), and the lowest temperature (TMIN) on the date of flight. Besides, we extracted the features "TMIN", "PRCP", and "SNOW", and created a new feature called "Frozen" that signifies whether the ground was frozen on the date of flight. While the elimination of irrelevant features simplifies the data and the model, the feature extraction makes it more likely to classify correctly with the help of more highly correlated features.

In addition to feature engineering, we also use the pipeline to ensure that the incoming data would be transformed with the same order of operations and parameters. In this case, it ensures that all the missing values are filled with imputer and all the features are scaled with standard scaler, which would enable the models to be trained more effectively. Also, we use one hot encoding to deal with categorical variables like Airline names (Reporting_Airline) and Departure Time (DepTimeBlk) in our model, which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction.

After data pre-processing section, we use train_test_split function to do data splitting, which partitions the data into two sets, train set and test set so that we can train our model with train set and evaluate the models performance with test set. We decide to make 80% of the data as train set and 20% as test set.

For model selection and model training, weve tried different classification including softmax regression, Linear SVM Classifier, Random Forest and K-Nearest Neighbors. We end up with K-Nearest neighbors classifier. This model uses a database in which the data points are separated into several

classes to predict the classification of a new sample point, which means with KNN model, we can predict whether a flight is no-delay, slight-delay or heavy-delay given the information about weather, airline names, departure time and distance. In addition, KNN is a non-parametric and lazy algorithm, which are both useful features in our case. Its non-parametric feature means KNN doesnt make any assumptions on the underlying data distribution, so the model structure is determined from the data. In our case, like most data in the real world, our data will not obey the typical theoretical assumptions made (as in linear regression models, for example) and also, we have no prior knowledge about the distribution data since this is our first time to deal with this dataset. KNN is also a lazy algorithm, which means that it doesnt use the training data points to do any generalization. In other words, there is no explicit training phase, so the training phase is pretty fast. Thats a very helpful feature to our project. We have a relatively large dataset (more than 90,000 at first), but limited computing power.

Furthermore, we used cross-validation to fine tune the hyperparameters so that we find the optimal balance between bias and variance of the model. Cross-validation also enables us to make efficient use of our data. When evaluating each models performance, we used the confusion matrix as it offers a good view of how correct the classifiers predictions are. Finally, we use accuracy rate, precision rate, recall rate and F1 scores to evaluate our models. With accuracy rate, we can know how many times we predict the delay of flights correctly. With precision rate, taking the no-delay situation as an example, we know what percent of the predictions we mark as no-delay flights are actually no-delay. With recall rate, we know what percent of the actually no-delay flight are caught by our model. We evaluate our models by considering these four standards together.

IV. Experiments

Since our project aims to establish a classification model that classifies flights in terms of delay time, we first thought about using **logistic regression**. Because we have three categories “No Delay,” “Slight Delay,” and “Heavy Delay” in our model, we end up with using **softmax regression**, the generalization of logistic regression that is used for multi-class classification. In order to construct the model for our dataset, we directly used the implemented LogisticRegression from the scikit-learn library. We constructed models with different values of hyperparameter C and used cross-validation to train each model. Finally, we used accuracy to evaluate the models to find the optimal model. In our experiments, the best hyperparameter C is 0.01 with an accuracy score of 0.8728397. However, we find out we have 0% precision rate and recall rate for heavy delay flight and very low recall rate for slight delay flights. This is a serious problem since in the proposal, we realize the recall rate is very important for our project since we would rather mistakenly predict an on-time flight as a delayed flight than the other way around.

Respect to this problem, we find that it should be caused by our very imbalanced dataset. In our dataset, we have 64,700 no delay flights, but only 28,657 slight delay flights and 3,040 heavy delay flights. Therefore, we do some research about how to deal with imbalanced dataset. Based on our research result, we decided to delete some of the data to make it balanced. We end up with the updated dataset with 8,000 no delay flights, 8,000 slight delay flights and 3,040 heavy delay flights. On the updated dataset, we run softmax regression again, the model with the hyperparameter C of 0.01 has 83.19% precision rate and 17.01% recall rate for heavy delay flights, which shows our solution to fix the imbalanced data do improve the model performance in terms of recall, precision and F1 scores. The recall rate for slight delay flight increases from 4.44% to 62.65% and recall rate for heavy delay flight increases from 0% to 17.01%. However, it is at cost of the accuracy rate. Our accuracy rate drops from 66.43% to 56.03%. After we tune the hyperparameters for softmax regression, the best model we can get is the one with C of 10, in which the recall rate for heavy flight delay can reach to 19.07%, compared to 17.01% for the model with C of 0.01.

In order to improve the recall rate and accuracy rate, we need to investigate other models. We try the second classification model we have learned in class, **Linear SVM**. The best linear SVM model we get is the model a hyperparameter C of 0.01. The accuracy rate for that model is 55.94%; the recall rate for slight delay flights is 63.93% and for heavy delay flight is 15.63%. None of them are better than best softmax regression. Therefore, we give up the Linear SVM model.

The third classification model we try is **Random Forest Model**. We first try the model with 100 trees and 16 max leaf nodes, and get the results of 56.73% accuracy, 68.21% recall for slight delay and 18.56% recall for heavy delay flights, which are all slightly better than the best softmax regression. Therefore, we decided to further tune the hyperparameters to find the optimal model. We try all combinations of 100, 500, and 1000 trees, and 8, 16, and 64 max leaf nodes. The model with 500 trees and 64 max leaf nodes performs best. Its accuracy is 57.42%; its recall rate for slight delay flights and heavy delay flights are 65.22% and 21.48%. Compared to the softmax regression, the accuracy increases by 0.69%, recall rate for slight delay flight and for heavy delay flight increase by 2.73% and 2.41%. However, the accuracy rate still doesnt improve a lot and the recall rate are lower than our expectation. We decided to do further research about other classification model that we havent learned in class.

Finally, we try **K-Nearest Neighbor** model. The first time, we set the number of neighbors to 3 and get an accuracy of 51.61%, the recall rate for slight delay flights of 49.88% and for heavy delay of 43.30%. Although the accuracy rate are 5.81% less than the best random forest model, the recall rate for heavy delay flight doubly increases from 21.48% to 43.30%, which is a huge improve for the recall rate of heavy delay flight. Then, we try to set the number of neighbors to 5. 10, 20, 30, and 50. We find as the number of neighbors increases, the accuracy and the recall rate for slight delay flights increases, but the recall rate for the heavy delay flight decreases. Although accuracy rate is important, we do evaluate

more about the recall rate, especially for the heavy delay. Therefore, we decided that the final model is the KNN model with 10 neighbors. Its accuracy is 57.0%, and recall rate for slightly delay flights and heavy delay flights are 55.20% and 33.68%.

V. Results

Originally, when using the softmax regression model on the data, weve achieved an accuracy of 87.3%. However, as we found that the high accuracy was an illusion brought by the imbalanced data, we modified the data through undersampling and came up with a more balanced dataset. More specifically speaking, we subsetted 8000 no-delay flights, 8000 slight-delay flights and 3000 heavy-delay flights. After several trials and errors mentioned in the Experiments Section, weve found that the K-Nearest Neighbors model yields the highest accuracy as well as precision rate. To begin with, we set the number of neighbors, a hyperparameter to tune, to three. As a result, we get an overall accuracy of 51.6%, a precision rate of 54.75% for no delay flights, 53.93% for slight delay flights and 38.41% for heavy delays. We furthermore set the hyperparameter to other numbers. It turns out that when the number of neighbors is set to 10, we reached a balance between precision and recall rate. The precision rates for no-delay, slight-delay and heavy-delay flights are 56.97%, 56.61% and 58.68% respectively, while the recall rates are 67.38%, 55.19% and 33.67% respectively. That is, of all the heavy-delay flights that we predict, 58.68% of them are actually heavy-delay flights; of all the actual heavy delay flights, we successfully capture 33.68% of them. Then as the number of neighbors increase, although the overall accuracy keeps increasing, its at cost of decreasing precision rate and recall rate for the classification of slight-delay and heavy-delay flights. For instance, when the number of neighbors is set to 50, the overall accuracy increases to 57.47%, but the precision rate for heavy-delay category becomes 62.98% and its recall rate becomes 25.43%.

To put everything in a nutshell, the best model for our case is the K-Nearest Neighbors and the optimal hyperparameter is for the number of neighbors to be 10.

VI. Conclusion

The motivation of the project is to enhance international students traveling experience by predicting flight delays in advance for them. In this way, an international student can arrange his or her traveling time beforehand. For example, if the student knew that his first flight is going to have a heavy delay, then he would leave more time for connecting time so that he wouldnt miss the next one. Besides, when deciding which flight to take, he or she can foresee the flights likeliness of delay.

To some extent, the results of our model validate our original claim, which is to classify flights in terms of delay time. Take the results of the finalized model as an example. Thanks to the KNN model, the lazy and non-parametric nature of which allows us to construct the model quickly without knowing the data distribution in advance. Our model has a overall accuracy of 56.97%. Besides, the precision rates for all three categories are all around 57%. The recall rates for the three categories are 67.39%, 55.20% and 33.68% respectively. Though not perfect, our model does offer some advice on whether the flight being predicted would delay.

However, a precision rate of 56.61% and 58.68% and a recall rate of 55.19% and 33.68% for detecting slight-delay and heavy-delay flight are way too low, considering the cost of misprediction. That is, our model does not capture well all the delaying flights. Particularly, we weigh the recall rate more as we would rather mispredict an on-time flight as delayed than miss a heavy-delay flight. However, as the results showed, only 33% of heavy-delay flights are actually captured by our model. Besides, of all the slight-delay flights that we predict, only half of them would actually be delayed flights. If an international student solely depends on our prediction, then this outcome might not benefit him or her much.

As we further investigate, we realized that this shortcoming in our findings originates from many aspects. The classification model not being sufficiently sophisticated or optimal and the features we chose not sufficiently representative, all these problems contribute to our low accuracy, precision rate, and recall rate. Therefore, in future work, we shall tackle these problems one by one. For example, we can utilize ensemble learning by combining several models such as Random Forest and K-Nearest Neighbors. It turns out that although Random Forest doesnt produce a model of which the recall rate for heavy delays is as good as the K-Nearest Neighbors, it does yield a high accuracy as well as F1 scores. Therefore, Random Forest model might have more potential to be explored. Besides, we can do more feature engineering. For instance, the features that we chose currently might not be the most representative. Perhaps features such as taxing time, which we considered as irrelevant, would surprisingly have a huge effect on the performance of our model.

VII. Contribution

Simon Zhu: Author of Abstract, Introduction, Results and Conclusion section of the Final Report; Editor of the Project Video; Feature engineering; Author of the Abstract, Motivation and Method sections of checkpoint paper; Author of the Motivation, Dataset, and Example of Related Work sections of the proposal.

Alicia Wu: Author of Method section, Experiment section, partial Introduction section for the Final Report; Content constructor for the Video; Feature selection and feature extraction; Author of the Preliminary Experiments and Next Steps of Checkpoint paper; Author of the Intended Experiment and Evaluation of the proposal.

Leo Wang: Feature selection and extraction; Feature engineering; Author of all coding sections in Jupyter Notebooks for Checkpoint and Final Report; Manager of dataset on download, extraction, combination, and preprocessing; Document organizer with LaTex.

REFERENCES

[1] Boyle, Tara, "Dealing with Imbalanced Data," Towards Data Science.

[2] Bronshtein, Adi, "A Quick Introduction to K-Nearest Neighbors Algorithm," Noteworthy - The Journal Blog.