Carrier Recommendation System

Gabriel Giangi (40174314) Tomas Pereira (40128504) Aryamann Mehra (40127106)

April 2022

Abstract

In this project, two machine learning models were engineered which aimed to speed up the dispatch process in the shipping industry (currently a time intensive, manual process). The final result was a recommendation system which predicted the top N carriers for a specific route and the prices they would charge, based on historical shipment data collected from a shipping company. The data used for training is a collection of around 5000 historical shipments with key feature columns such as origin, destination, weight of shipment, etc. Preprocessing included trimming whitespaces, dropping cells with important features missing, removing entries where the carrier was a logistic company, and more. After testing several different classification models and regression models, the random forest classifier model architecture and random forest regressor model architecture was chosen for the final version, as they had the best testing accuracies. After hyperparameter validation on these models, our classification model had a top-5 accuracy of 50.20% (due to an imbalanced dataset). The regression model had an accuracy of 77.50%. Further, we connected the two models in order to give a combined carrier-price prediction for a user-given input. Upon testing the model with a more balanced dataset we achieved a top-5 accuracy of 71.77%, implying that this model could perform much better given a more diverse and balanced dataset. While the model did not perform up to our initial expectations, the scalability of the model - given more data - seems to be a promising avenue to explore.

1 Introduction

The transportation industry is behind in terms of technological innovations. Many tasks are done manually, and are inefficient in terms of what is capable in today's world. The focus of this report is dispatching, which is the process of finding trucks to move specific shipments. Currently, customers call in to brokerage firms when they need to move a shipment. The brokerage then spends hours (ranges from 1 to 3) to find a suitable carrier that has the equipment, availability and great rate for that specific route and shipment. There are hundreds to thousands of unique carriers, all with their own respective equipment set and competitive lane prices. Due to the immense size of the carriers and their differentiables, it is impossible to completely memorize and know all of the intricacies, which is why dispatching is currently such a lengthy process.

Thus, this seemingly is a perfect problem to solve with machine learning, where we can "teach" a model through large amounts of shipment data, and hopefully it can pick up on these differences and trends. The goal of this project is to first determine if this is a possible task, how accurate we can get the model to be with our limited data, and how much time this can save the dispatch process.

2 Methodology

2.1 Multilayer Perceptron

We tried to fit an MLP classifier to predict carriers for the given data and initial accuracy was quite low. This was boosted by normalizing the data. For categorical data, one-hot encoding was used and for numerical data we used StandardScaler. This provided a to the accuracy. We further used hyperparameter optimization and found the best learning rate to be 0.01 (a balance between the computational intensity and boost to accuracy). The hidden layer size was best at (64,16) and max iterations were set to 2500 to allow the MLP to converge. The best possible accuracy found from the model was 32%. Since our RandomForest Model scored better with hyperparameter optimization, we chose to scrap this model for the carrier predictions.

2.2 Random Forest Classfier and Regressor

After testing with a neural network and not getting the performance we were looking for, we decided to construct a decision tree classifier. Decision trees are known to generally overfit the dataset, and it was no different in our case. Despite trying to tweak hyperparameters, we were getting similar if not worse performance compared to the neural network. The next intuitive step was to construct a random forest classifier, which generalizes data much better due to the "voting" mechanism of multiple decision trees, and we saw a significant increase in our testing accuracies. Given the context of our data, and the problem we are trying to solve, random forest architectures make sense as it splits our labels dependent on the feature values. In a real life scenario, we look at each feature and then narrow down the list of carriers that have those capabilities, so because of this, and the increased performance, we stuck with the random forest classifier and started hyperparameter validation to get the best performance computationally possible.

We ran k-fold validation to iteratively determine an approximate best fold split for random search. We decided on random search over grid search due to the increased performance and decreased execution time we witnessed in our labs. Because we saw that the random forest classifier best described our data, we then built another random forest model, this time optimized for regression. The exact same procedures and methodologies were applied for the regression model. When it came to normalizing our data, it was not a difficult process due to the inherent nature of the random forest models. Features don't need to be completely normalized (relative scale to each other), rather, we had to ensure that the categorical feature variables were set as one-hot-encoded integers whilst continuous variables were set as float values. That way,

during the splits, continuous variables were split at a certain threshold (ex. Weight > 100.00) and categorical variables were split based on active features (ex. Split if shipment is LTL ("Less Than Truckload").

2.3 Custom Scoring Function

Due to the nature of the carrier recommendation classifier being that it will be used to speed up the process of dispatching, it was developed with the idea in mind that the predictions will be used to recommend which carriers should first be considered. As the given prediction will not always result in a shipment being successfully assigned to a carrier, a custom loss function was developed which considers a prediction to be correct if the true label is present within the 5 top recommendations. While we are aware that this no longer measures whether the model can accurately predict the best suited carrier on its first attempt, this measure was considered as it would be more representative of its practical use. Using this modified criteria, the accuracy of the model rose by approximately 20% to a total of 50.2% on the validation set.

3 Experimental Results

3.1 Multilayer Perceptron

The initial best carrier accuracy of the MLP Classifier (with one-hot encoded categorical values and uniformly scaled numerical values) with default parameters was approximately 21% (obtained using accuracy_score). After hyperparameter optimization, the accuracy was boosted to approximately 28.5%. The best hyperparameters were found to be: Learning Rate = 0.01, Max Iterations: 2500, Hidden Layer Size = (64,16), Activation: 'tanh'. In the end, this model was replaced by a Random Forest Classifier due to better performance.

3.2 Random Forest Classfier

As this has been a term length project, and without thought, the initial accuracies obtained from the decision tree and random forest models were not recorded, as the focus was building the best performing system and not so much the final report (at the time). This part of the report will focus on the final results obtained, and a discussion about them. After running k-fold validation for the random forest classifier, we obtained a best approximate fold of 5/6/7 (all same) that gave us a held out accuracy of 30.2%. From multiple sources, (such as StackOverflow, Medium, and Towards Data Science), if we set random search to 60 iterations, 95% of the time we will find a hyperparameter configuration in the top 5% of best performing configurations. We randomly searched with n_iter at 30 (because 60 took long to compute), and cv = 6. It was found that a relative best performing model had 250 estimators, all with a max depth of 15. The validation set accuracy with this model was recorded at 32.7%. After training the model with those hyperparameters, we obtained a test accuracy of 30.43% and training accuracy of 85.47% for our classifier.

3.3 Random Forest Regressor

After running k-fold validation for the random forest regressor, we obtained a best approximate fold of 11 that gave us a held out accuracy of 79.43%. We randomly searched with n_iter at 30 (because 60 took long to compute), and cv = 11. It was found that a relative best performing model had 500 estimators, all with a max depth of 45. The validation set accuracy with this model was recorded at 79.5%. After training the model with those hyperparameters, we obtained a test accuracy of 77.5% and training accuracy of 97.0% for our classifier.

3.4 Discussion of Data Imbalance

A strong limitation observed in the development of our models came from the size of our dataset in conjunction with the high number of different carriers (labels) within it. In analyzing the whole dataset, it was found that the 1549 distinct labels were present in the 5057 data points, and of this number, 942 labels appeared only once. On the opposite side, the label associated with the most entries appeared a total of 219 times. To solve this issue random over-sampling was considered, however the large disparity between the majority and minority classes made this highly prone to overfitting. To measure the effect of this data bias, 3 additional carrier classification models were trained using datasets which removed the lowest occurrence labels. The performance of each of these models was plotted side by side and can be seen in the appendix. Figure 1 plots both testing and training accuracy and shows that removing the lowest occurrence labels led to significant increase in model accuracy. When using the Top 5 scoring mentioned previously (Figure 2) the greatest performing model, which was left with 271 of the 1549 labels, had an accuracy of 71.77% on the validation set. From this we conclude that given a much larger and more balanced dataset, our carrier classification model may have been able to achieve performance even beyond our expectations.

4 Conclusions

In planning this project, it was hoped that we could achieve an accuracy of 95% for our carrier classification model when considering a top 5 performance measure. While initially it was planned to use a neural network to achieve this, we quickly found that the size of our dataset was insufficient to achieve good generalization and thus had to reconsider and use a random forest classifier. As our greatest performing classifier model achieved only 71.77% accuracy, we can not say that we reached our goal but we believe that this would have been achievable given a more balanced dataset.

Although the accuracy can be described as “mediocre”, the time saved is where we see the most success from the model. As previously mentioned, it takes a dispatcher on average 1-3 hours to find a truck that has proper capabilities. With the system, we see an absolutely drastic time decrease of -99.95% with the model capable of outputting a capable carrier at around 5 seconds on average.

Appendix: Figures

Figure 1: Bar Graph Plotting Accuracy of 4 Carrier Classifiers using Modified Datasets

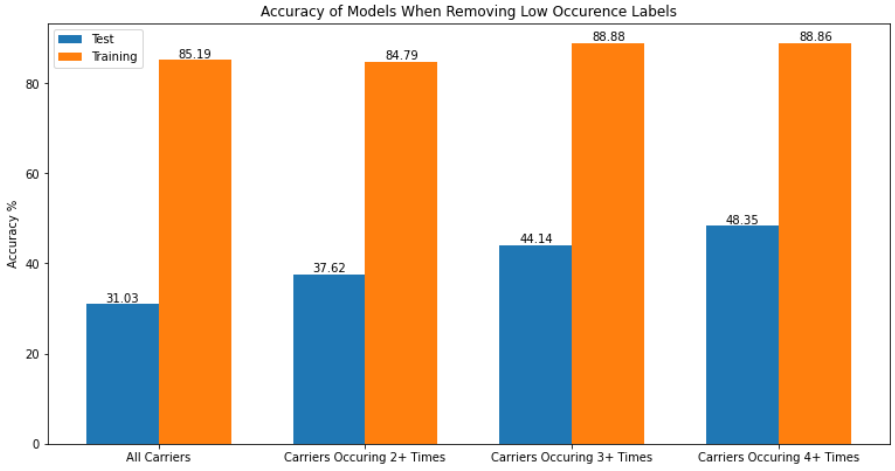

Figure 2: Bar Graph of Model Comparison using Top 5 Scoring

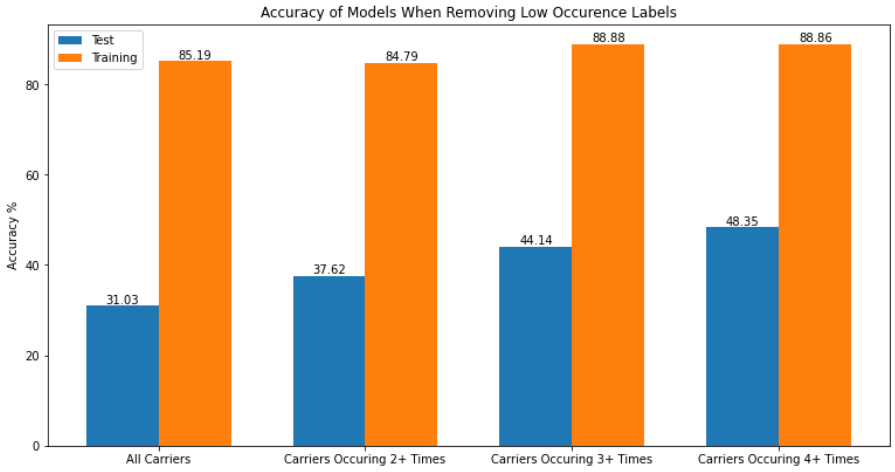

Figure 3: Route Lane Data Plot (Data visualization)