Analysis of train delays and train delay prediction

HELI HUHTILAINEN, MILLA LINTUNEN, ROOPE NIEMI

University of Helsinki

In a busy world like ours, it is important for people to be able to plan their timetables accurately. Sometimes it feels that trains in Finland are always late, but are they and how much? We developed a machine learning model with the identified factors that could affect the train delays. Having a model that predicts if a train is late would help busy people to find out and plan their timetables. The predictions could be checked for a certain date, time and a station. The results would also help HSL and VR to recognize if there are some patterns that cause trains to be late.

1 Data collection and preprocessing

For this project we used daily train data collected in the years 2017-2019 from DigiTrafficAPI (https://rata.digitraffic.fi). In the collected data there was a json file for each day, containing data from all the trains that travelled that day. For each train there was a list of the stations the train came across on the way from departure station to its destination. In each of these was information related to the train passing the station, including scheduled time the train should arrive or leave, the actual time it arrived or left, and the difference of these two in minutes.

Data consists of all train information from all the stations in Finland during that time. Dataset included a large amount of information that was also unnecessary, such as operatorUICCode, stationcUICCode and trainNumber originally containing over 50 million rows of data. We decided to use local train commuter lines. Train delays were announced in data by minute and every train had both arrival and departure times for every station, except for departure stations that had only departure and destination stations that had only arrival. We decided to use only departure times for two main reasons; one being that data size would be somewhat reasonable and another reason was that we wanted to predict when a person should expect a train to leave the station. We also included only those stations for each train that the train stopped at, excluding the stations it only passed.

Expecting that weather is a big factor on train delays we also collected weather data that was merged with train data. Regions of train station locations remain quite wide, so multiple different weather stations were needed. Weather data was collected from Ilmatieteenlaitos open data (https://www.ilmatieteenlaitos.fi/avoin-data) for every hour. We ended up choosing eight different weather areas (Helsinki, Espoo, Kirkkonummi, Lahti, Hämeenlinna, Tampere, Hyvinkää, Kouvola). Weather factors were chosen to be temperature, wind gust, wind speed and rain intensity. From the beginning there were also some other factors included, but after research that future weather forecasts have only these four similar fields (when concerning possible relevant fields to involve) so that was the final reason these four factors were chosen. One problem occurred that there were some weather stations missing some of these field values because apparently all stations are not collecting all the data. For example when finding weather for Lahti area, there was one station that had temperature and rain but wind was missing, so it had to be collected from other weather stations data which did not have some other features. In the train data we added the weather area (1 to 8) joining the station name to the right area. On concatenated weather data we also added the weather area column and then in the end merged the cleaned train data to weather data by weather area, year, month, day and hour. After cleaning data and merging it we retained about 10,3 million rows of data.

One problem in addition to huge data amounts with different trains and stations remained that the weather values temperature, wind and rain had decimals. We changed them to int values in order to data being more simple and model learning easier. With train data we changed delay minutes into time slots of three (0 min, 1-2 min and over 3 min) as the long delays were rare and affected only few trains. We also examined results and made visualizations selecting four classes (0 min, 1-2 min, 3-5 min and over 6 min) instead of three. In addition we added weekdays and direction to the data. Direction was calculated from the first station the train departed from. If the first station was Helsinki, direction was set to 1, otherwise 0.

2 Exploratory analysis

To get a better understanding of the data and delays, we first decided to see at what scale and how many minutes trains were usually late. The delay distribution is shown in Figure 1. Turned out that even against general expectations of trains being quite often late, in local train traffic there were quite few large delays, and mostly the trains either were on time or just 1-2 minutes late. One huge problem was that some of the big delays are so unexpected that these are quite hard to predict. Because of this, the minutes late column had to be categorized to just a few different values. Reducing them to just 2 categories, late and not late was also considered. This would have reduced the problem to "predict whether a train is late or not". This didn't cause any significant improvement in accuracy, so we decided to use more than 2 categories to give the train predictor user a better scope of the predicted delay.

Only 105 000 data rows had details about the causes of the delays. These we could only use in the exploratory phase as the data was so scarce and corresponding data was not available for the real time predictions. Analysing the causes we found out that weather was part of the delay reasons, but there were many other factors as well, like other trains blocking the way or faulty equipment or infrastructure failures. Surely weather might be part of the reason in some of the other causes as well.

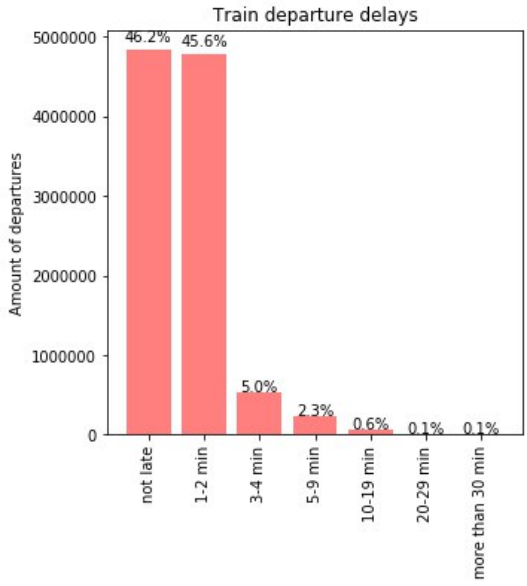

Fig. 1. Almost half of the trains are right on time. Most of the trains that are late have only small (1-2 min) delay. Long delays are rare.

3 Machine learning

Our focus ended up being in supervised learning methods, mainly tree based methods. We tested different models to see accuracy scores and tested out different hyperparameters. In addition to tree based methods we explored different kinds of neural networks. We also used Tpot with

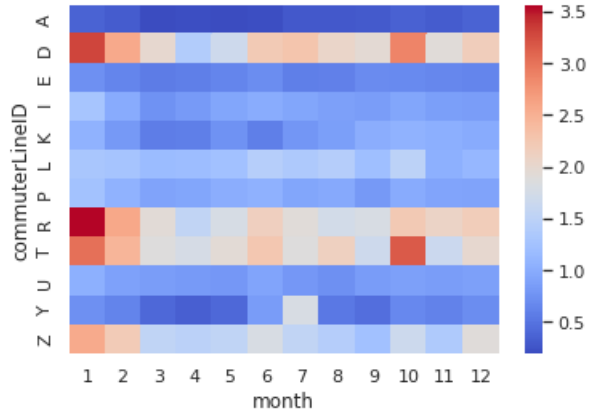

Fig. 2. Commuter lines T, R and D seem to have most delays. Also January, February, October and December have bit more delays.

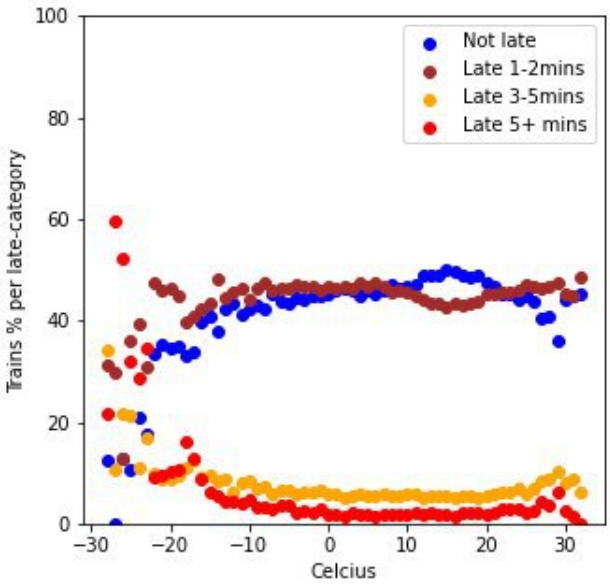

Fig. 3. Train delays affected by temperature when 4 delay classes are used.

a sample of 200 000 rows to determine what kind of model would suit best for us. Tpot recommended Extra Trees Classifier as the best model and some hyperparameters that would result in best accuracy. We continued developing Extra Tree classifiers and Random Forest classifiers. However, using Tpot recommended hyperparameters resulted in very large model files (around 2 GB). Also the Extra Trees classifier model build crashed when the build was done using minutes as target, and only worked when the target was restricted to time slots. The model sizes were reduced when the amount of classes was restricted, the features were rounded to ints instead of using floats and the maximum model depth was restricted. The size of the model varied also depending on used sample size as we also have found out in our own model builds. When trying

out different sample sizes we were able to use all the data, but as the model accuracy stayed the same in 2,5 million, 5 million and all data row models, we ended up using the 2,5 million row model in our application.

As we continued optimising the Random forest classifier, we found out that leaving max depth undefined was the main reason that produced huge model file sizes, but decreasing it also caused the model to be less accurate, as we can see in Figures 4 and 5. A useful tradeoff was to use max depth of 14. We also looked into the amount of trees used in the model (Figure 6). We found out that we don't actually need the default 100 trees for our model, so only 50 was used. Other hyperparameters were chosen according to Tpot recommendation. These were max features 0,70, criterion gini, no bootstrap, min samples leaf 10 and min samples split 8.

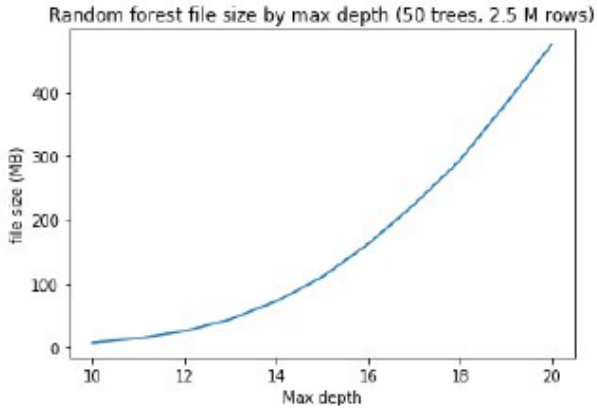

Fig. 4. Figure showing the increase of the model size when Random forest max depth is increasing.

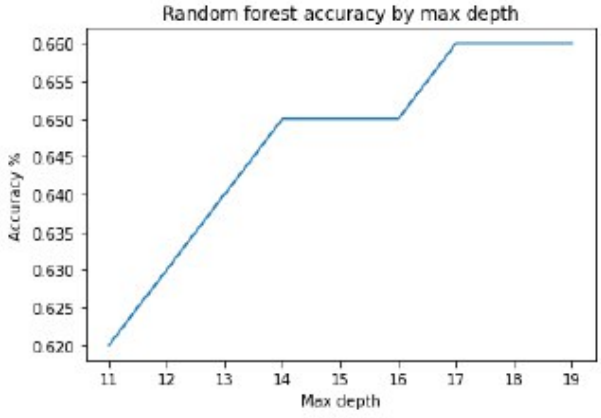

Fig. 5. Trying out differerent max depth to find a balance between model accuracy and model file size. Accuracy did not increase much even if max depth was increased.

4 Visualizations

A lot of different kinds of graphical representation of data were made helping to see patterns of the data more clearly. We used mainly matlibplot and seaborn in our visualizations. Plotting of trains late given for example day,

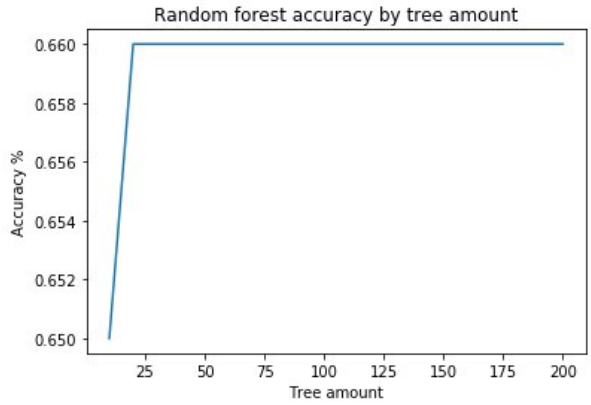

Fig. 6. Accuracy did not increase after a certain amount of trees.

month, train, station and temperature. In visualizations e.g. celsius degrees were divided into categories so that visualization would provide a more clear image of the situation. We had added weekdays so visualizing hour time slots and weekdays showed some variation with delays but not huge. At night times there seemed to be a bit more delays, especially on Saturday night.

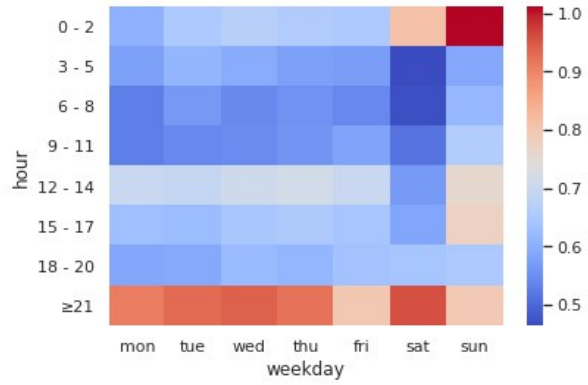

Fig. 7. Visualization of delays in different weekdays and by the hour. Middle of the day and later night have bit more delays in general.

5 Results and future

The results of train and weather data had less delays than expected. Although with long distance trains delays might occur more and application and model could be expanded to long distance trains. Results of the current local commuter line trains could potentially be improved by introducing other delay cause features.

Data was splitted training and testing sets with ratio 80/20. Random Forest Classifier and Extra Trees Classifier performed the best in our experiments. With Extra Trees Classifier we got accuracies between 53 and 70 percent. When predicting delays by the minute with Random forest (using all the minutes by themselves), accuracy remained 53 %. Using Extra trees for a model with minutes as target was not possible but when the Extra trees model was tested with boolean prediction that trains either are late or not and

with that managed to get accuracy of 65-70 %. When dividing minutes to three (under 0, 1-2 and over 3) or four (under 0, 1-2, 3-5, over 6) classes accuracy was around 63-65 %. Random Forest Classifier gave similar results. With neural networks we got accuracy 58-60 percent.

Initially in the application the prediction was performed by determining the train line, station, month, day and hour and direction and taking current weather into account in that certain area. This resulted in an accuracy of around 61 %. When also weather area and year features were introduced to the model, the accuracy improved a bit to 65 %. Some of the accuracy results and confusion matrices are visualized in the figure 8 and in this we can see that the models have trouble recognising the class 2 that is the longer delays (3 min or more). To improve the model's ability to predict these with more accuracy, the data would need to be enriched with some other delay causing factors, like track work, accidents and other surprising events. This would also enable using the causes in the history data when creating the model. How to recognize and incorporate these factors for the prediction proved too big a challenge for the time scope of this project.

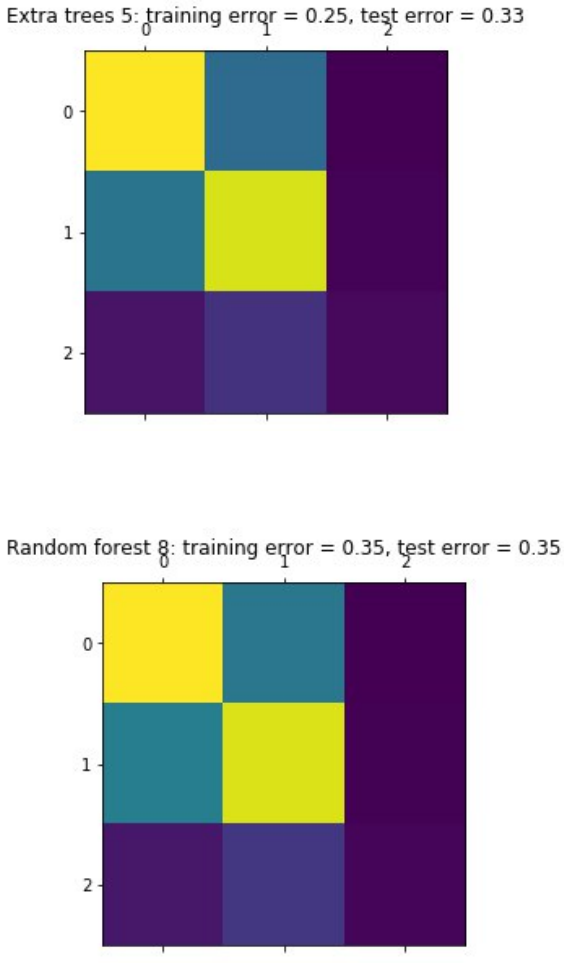

Fig. 8. Extra trees and random forest accuracy and confusion matrices were quite alike, and show that the class 2 (3 min or more) was hardest for the model to predict.

6 Challenges

One of the biggest challenges we had was the amount of the data. When first training the model we could only use ¼ of the data at the best which had some effect on accuracy and different scenarios missing in the training data. Especially with tree based models that suited best to our purposes the amount was a problem. Data preprocessing took a lot of time during the project. In train data there were lots of nested structures. Weather data was missing relevant data from certain stations and had to be collected from different stations. When writing the script for fetching the weather forecast the challenge was how to get data from the same areas as in the history data.

We had taken into consideration that rail constructions could be an affecting factor to train delays. Unfortunately that data was not available to be used. We found some data but a lot of time slots were missing, for example we found some data from 2020 but 2019 had none. Another challenge was that there seemed to be some factors that cannot be predicted at all, like accidents and some brakings. If ages of the trains would be available some of the brakings of the train could have been predicted. In the VR delay cause codes that were used in the train data some of the codes did not have explanation, and it is possible that weather could also be a factor in these. Also most of the rows with delays lacked cause information entirely.

Model size was challenging which was not that surprising after a huge amount of data and a bit complex model. Model size could be reduced by limiting the depth of the trees. The depth reduction could be well justified to make the model less prone to overfitting.

One consideration is the unusual nature of 2020, as the amount of passengers might be affected by the ongoing pandemic. To make the model more current, the year 2020 timetables might have been a good addition to it.

7 Application

Application is available in Heroku. It uses a random forest model that has approximately 65 % accuracy and also it fetches future 48 hour weather forecasts to make a prediction if a certain train will be departing late from the chosen station. The weather forecasts are fetched using fmiopendata library as a Python interface for FMI open data. The weather forecast station is chosen using the weather area that has been defined for each train station, and given to the query as coordinates.

Unfortunately for now the application does not contain validations so it will not announce wrong inputs. Month, day and time (hour) must be int values so that time slot is inside the future 48 hours or it fails to find weather forecast and make a prediction. Application also contains a map where you can explore what percentage of trains have been departing late in different months, days and time of the days. It shows with ten different colours the changes where darker color means a larger percentage of trains being late in that station in that certain time (usually) accord-

ing to data. There is also a statistics view where one can see heatmaps, barplot and explore differences.

The application prediction could be also visualised further, if the history visualization would be combined with the prediction model. Daily and even monthly percentages of late trains per station could also be valuable to catch certain patterns in data, at the moment visualization contains only late percentages per hour.

8 Conclusion

Overall this project provided interesting insights into the problem of the train delays and to achieve more accurate results would require more information for example on train conditions, construction works on rails or traffic volumes. We were able to achieve close to 70 % accuracy at the best, by restricting the problem to finding if a train is late or not. Introducing more classes to the model took the accuracy down a notch to 67 %, and getting the model to a more handleable size and less likely overfitted version to 65 %. Making a model with a larger dataset containing data from several years and trying to predict long distance train delays would have potential to improve results in the future. To be able to make more accurate predictions, more data from a longer period of time should be used, and other cause factors should be incorporated in the data. In local commuter lines there are quite a few longer delays and also factors that cannot be predicted.

9 Links

- https://github.com/millalin/Train-predictor
- https://train-predictor.herokuapp.com/home
- https://rata.digitraffic.fi/api/v1/trains/dumps/list.html
- https://www.ilmatieteenlaitos.fi/havaintojen-lataus