

Passenger train delay prediction

1st Suhasini Aswath

Department of Software Engineering
Blekinge Institute of Technology
Karlskrona, Sweden
suas23@student.bth.se

2nd Ziaul Islam Chowdhury

Department of Software Engineering
Blekinge Institute of Technology
Karlskrona, Sweden
zich18@student.bth.se

3rd Christina Larsson

Department of Software Engineering
Blekinge Institute of Technology
Karlskrona, Sweden
chla24@student.bth.se

Abstract—This document is a model and instructions for L^AT_EX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.

Index Terms—component, formatting, style, styling, insert

I. INTRODUCTION

...

II. BACKGROUND AND RELATED WORK

...

III. PROJECT DESCRIPTION

...

IV. PROJECT EXECUTION

A. Data collection

B. Main requirements

- 1) Data Requirements - input data specification:
- 2) Functional Requirements:
- 3) Performance Requirements:
- 4) System Requirements:
- 5) Compliance and standards:

C. Overall System Architecture

The high-level architecture of the machine learning application is shown in Fig. 1.

The architecture of the train delay prediction system is designed in a way so that the training, testing can be continued any number of times without impacting the production backend service. This means that model weights are dumped to files after training and testing phase. The prediction backend service can load the model weights anytime depending on the necessity without any downtime.

It can be seen from the architecture diagram that data is loaded with requested API key from REST API of Trafikverket.se. Response of the REST API is in the format of JSON and contains nested JSON structure. This JSON content is then normalized in Pandas DataFrame. Data preprocessor class does the preprocessing of the normalized content and does feature engineering to make it meaningful for the machine learning model. After completion of data processing, the problem is transformed as a binary classification model with binary label where values are either late or on-time (0/1).

In the next step, Random Forest classifier from Scikit-learn Python package is chosen as the binary classification model. The model is then trained with the dataset and tested. After applying GridSearch, best model parameters are found. The weights of the best classifier is then exported as Pickle file to file storage.

Train delay prediction model loads model weights and initializes Random Forest classifier model. Additionally, label encoders are also initialized as these are required to transform user input coming from the frontend application. In the next step, Flask based web server starts and it offers REST API for the prediction of train delay. When user selects specific train route, date and time, REST service is called to the web server with HTTP POST method. The service then predicts if the train will be delayed or not and returns a JSON data back to the frontend application.

D. Data Engineering

A detailed pipeline of data engineering is shown in Fig. 2.

Figure 2 shows that the TRV REST API is used to get train announcements. API key is required to make REST call and registration is required to get an API key. TRV's Open API supports various types of data and train announcement is one of them. Table 1 shows the data model of train announcement containing the following attributes (description is taken from the schema model).

It has already been mentioned that the response is returned as JSON and contains many fields containing repetitive information. Additionally, some attributes do not bring any value such as the ID fields or the web links. Hence, data preprocessing is required. At first, JSON data is converted to Pandas DataFrame and the nested JSON structure is normalized into columns. From the normalized columns, a subset of columns is chosen to be a part of the final dataset.

From the advertised_time_at_location column, nine temporal features are extracted. These include advertised_year, advertised_month, and advertised_day, which represent the date components. Time components are captured as advertised_hour and advertised_minute. Additionally, features like advertised_day_of_week, advertised_is_weekend, advertised_season, and advertised_quarter provide insights into the weekly, seasonal, and quarterly aspects of the time data.

TABLE I
DATA MODEL OF TRAIN ANNOUNCEMENT

Name	Type	Description
ActivityId	string	Unique ID of the activity.
ActivityType	string	Type of the activity (Ankomst/Avgang)
Advertised	boolean	Indicates whether the arrival/departure is announced in the timetable.
AdvertisedTimeAtLocation	dateTime	Timetable time.
AdvertisedTrainIdent	string	Advertised train number (the train number on the ticket).
Booking	[]	Code for booking information and booking information, eg: "Carriage 4 unregistered."
Canceled	boolean	Indicates whether the arrival/departure is cancelled.
Deleted	boolean	Indicates that the data record has been deleted.
DepartureDateOTN	dateTime	Expiry date of the Operational train number.
Deviation	[]	Any discrepancy with full reason code, e.g.: ABC023 and description, e.g.: "Bus replaces", "Track changed", "Short train", "Not served" etc.
EstimatedTimeAtLocation	dateTime	Time of estimated arrival or departure.
EstimatedTimeIsPreliminary	boolean	Indicates whether an estimated time is preliminary.
FromLocation	[]	From the station of the train in order and in which priority to be displayed.
InformationOwner	string	The name of the traffic information owner.
LocationDateOTN	dateTime	The operating train's arrival or departure time according to the timetable.
LocationSignature	string	Signature for the station..
MobileWebLink	string	URL to the traffic owner's mobile website.
ModifiedTime	dateTime	Time when the data record was changed.
NewEquipment	int	Indicates the order in which the train is equipped. If no new equipment has taken place, the value will be zero.
OperationalTrainNumber	string	Operational train number (OTN).
Operator	string	The railway company that carries out railway traffic, i.e. runs the train, for a traffic organizer.
OtherInformation	[]	Code for other advertising information and other advertising information, e.g. "Nice trip!", "Rear vehicle goes locked!", "No boarding".
PlannedEstimatedTimeAtLocation	dateTime	Specifies a planned delay and its validity is specified with the PlannedEstimatedTimeAtLocationIsValid flag.
PlannedEstimatedTimeAtLocationIsValid	boolean	Indicates whether PlannedEstimatedTime is valid. Will be set to false when an operational estimated time report, time report or slope report is created.
ProductInformation	[]	Code for description of the train and description of the train, ex. "Tågkompaniet", "SJ InterCity", "TiB/Tågkomp".
ScheduledDepartureDateTime	dateTime	The train's announced departure date
Service	[]	Service code and a little extra in addition to product information, e.g. "Bistro", "Sleep and bed".
TimeAtLocation	dateTime	When the train has arrived or departed
TimeAtLocationWithSeconds	dateTime	When the train has arrived or departed, with seconds.
ToLocation	[]	To station for the train in order and in which priority to be displayed. Note that it refers to what is to be advertised to travelers, i.e. what is to be shown on signs etc. In other words, ToLocation can have different content for the same train at different stations and different content for arrivals and departures respectively. The field specifies how to-stations are to be advertised.
TrackAtLocation	string	Track
TrainComposition	[]	Code for train composition and train composition, ex: "Wagon order 7, 6, 5, 4, 2, 1".
TrainOwner	string	The owner of the current train position.
TypeOfTraffic	[]	The type of traffic, e.g. "Bus", "Commuter", "Taxi", "Train".
WebLink	string	URL to the traffic owner's website.
WebLinkName	string	Name of the traffic info owner to use in links.

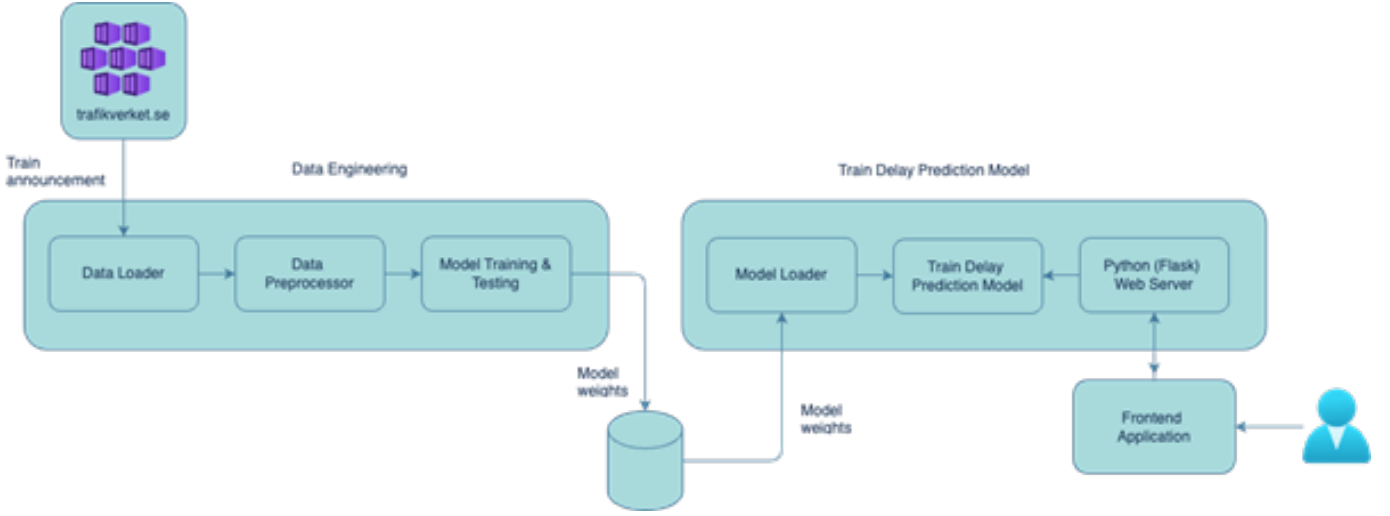


Fig. 1. Overall architecture.

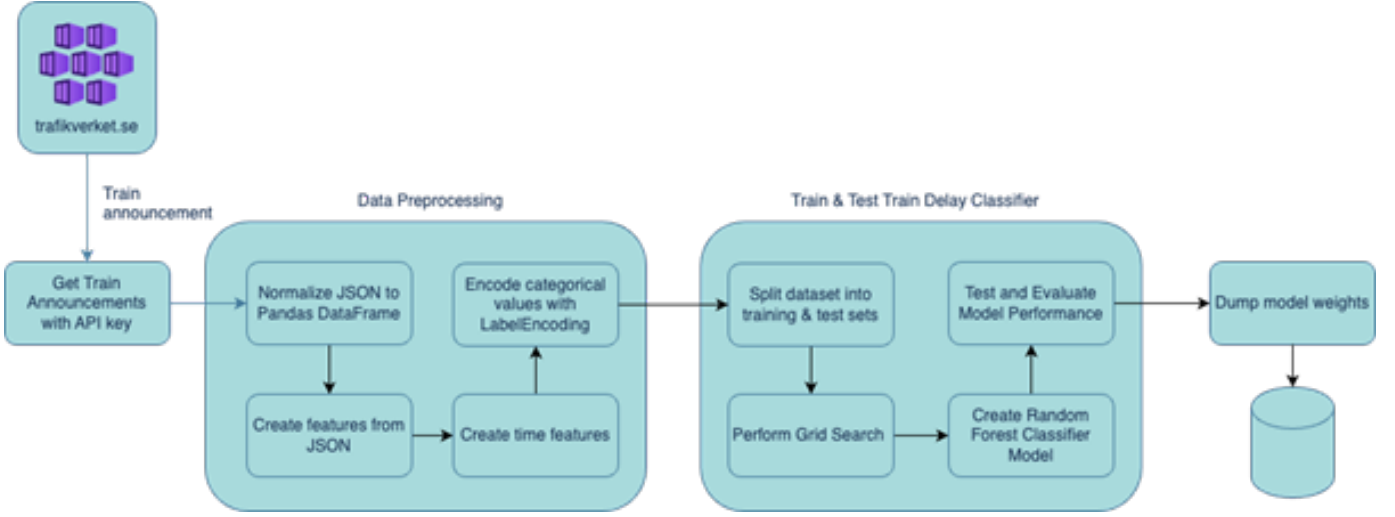


Fig. 2. Data engineering pipeline.

Categorical features within the dataset are transformed using the label encoding technique.

The final dataset after preprocessing is split into training and testing sets with 75 percent and 25 percent. With the training and testing sets, grid search algorithm is applied to find best parameters of Random Forest classifier such as maximum depth of the tree. The final Random Forest classifier model is constructed with the hyper parameters. The model is then trained with the training dataset. Finally, testing dataset is applied to the model for prediction and from there confusion matrix and other performance metrics such as F1 score, true positive rate, false positive rate is generated.

As the train delay prediction backend service is separated from the data engineering (training and testing), model weights are then dumped to the file system.

E. Train Delay Prediction Model

A detailed pipeline of train delay prediction model is shown in Fig. 3.

Fig. 3 shows that the Random Forest Classifier model is initialized from the stored weights along with the label encoders. A Flask server runs on port 8080 and provides 2 REST services to get train schedules and prediction train delay. To assist development and keep track of the APIs, Open API documentation is added with Swagger. Swagger UI lists all available REST services and provides options to execute REST call from the user interface.

Frontend application loads list of available train schedules from the web servers and displays them to the client accessing the application. When a client selects train route along with the departure date and time, /v1/predict-train-delay service is called with HTTP POST method. The prediction service makes a call to the request preprocessor which converts the incoming

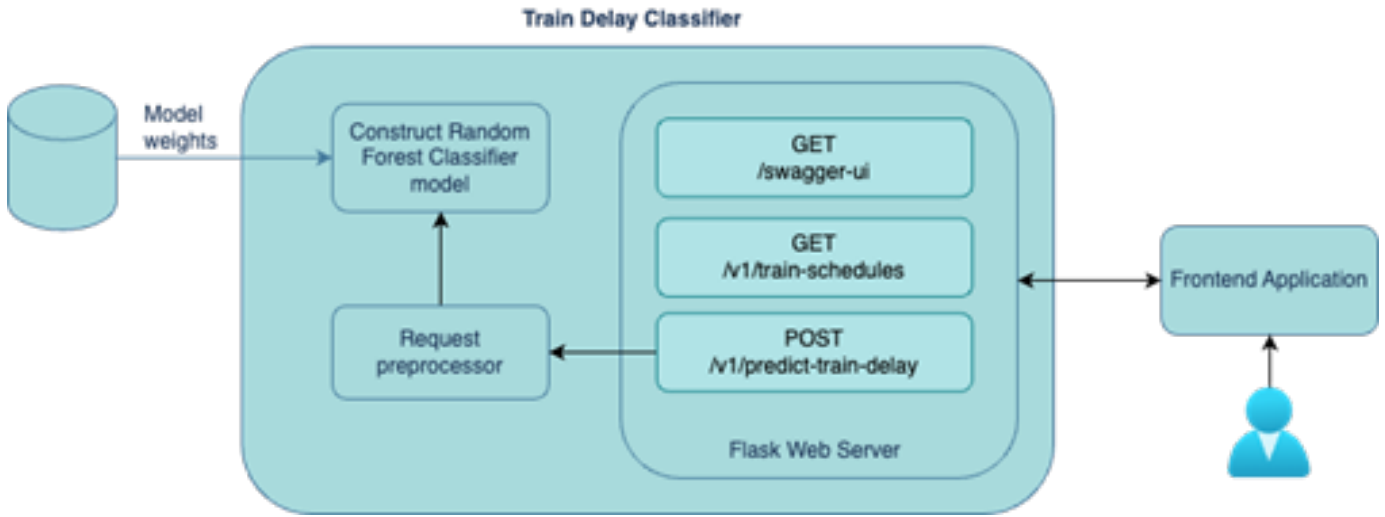


Fig. 3. Architecture Diagram of Train Delay Prediction Model.

data to the format which is understood by the machine learning model. The model then makes the prediction if the train will be delayed or not. The response is then returned to the frontend application as JSON format.

F. Outcomes of testing

When testing the developed prototype, the outcome was successful.

G. Deployment and Build Pipeline

To automate the build and deployment of data engineering and train delay prediction system, Jenkins tool was installed on local Docker. A multibranch build pipeline was created and linked to the JenkinsFile located under data engineering part of the project. Idea is to automate build containing pipeline states shown in Fig. 4. Unfortunately, the pipeline does not work as expected and it was not possible to fix the Docker related issue (inside of the Docker image of Jenkins) due to time constraints.

H. Maintenance

Trafikverket publishes the API and ... If there are updates to the API there will be information from Trafikverket...

I. User interface

The purpose of the user interface is to allow for the user to choose a train departure with SJ and predict if it will arrive on time. The attentive reader may notice that the model predicts delay, but the user interface is called "train arrival predictor"; it has a nicer ring to it. The frontend app is built for use cases centered on passengers. One use case is that a future passenger with a ticket for a specific train departure would like to know if the train is likely to arrive on time. Another use case it that a potential ticket buyer finds information before purchasing a ticket whether the train will arrive on time. The user finds a departure at sj.se and writes for their chosen departure "From-To", "Date" and "Time" of departure. The first version of the

user interface was coded in Python and then it was improved to HTML to enhance the usability.

V. RESULTS

The authors has created a backend service based on machine learning that correctly predicts future train delays based on the time frame set up. There is a frontend service aimed at passengers with a HTML user interface. The developed programs fulfill the main requirements that the team set up for the project.

VI. CONCLUSIONS

The authors have created a novel service that helps passengers choose a departure with high chance of timely arrival. If the planned departure is expected to arrive late, the service can help passengers prepare with extra snacks, beverages, and patience for a delay.

One difficulty in choosing a training set is that there are many factors that affect whether a train arrives on time. Issues that surface quickly can be problems with the infrastructure, such as damaged rails or electricity supply, or animal collisions.

Future work can include increase the training data to a larger time span, for example one year, so that for a search on a Wednesday train departure train the model on one earlier non-holiday Wednesday departures.

Another future development of the model is to predict how large the delay will be in spans of 5 minutes.

VII. ACKNOWLEDGMENT

The authors would like to thank Emil Klasson Svensson at SJ for his patient and kind support during the project.




Fig. 4. CI/CD pipeline of data engineering.


Welcome to the train arrival predictor


This tool uses a machine learning model to predict if a specific train departure will arrive on time at the final station. You can use it to predict if the train you have bought a ticket for will arrive on time or use it to choose which departure to buy a ticket for.

Find your departure at sj.se Write for your chosen departure "From-To", "Date" and "Time" of departure.

Type of train and From - To:

Please select your train route 

åååå - mm - dd 

-- : -- 




Fig. 5. The user interface.

VIII. REFERENCES

Interview with Johanna Törnquist Krasemann at Blekinge Institute of Technology 2024-04-30 Interviews with Per Ced-erström at SJ Business Control, several during March 2024 Interviews with Emil Klasson Svensson at SJ Traffic Planning, several during March and April 2024

Webbapp “Mina tåg” by Trafikverket <https://www.trafikverket.se/e-tjanster/mina-tag-den-tillgangliga-appen-for-trafikinformation/>

BELOW IS SUPPORT TEXT FOR GUIDING IN THE FORMAT. WILL BE REMOVED.

- [2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
- [3] I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
- [4] K. Elissa, “Title of paper if known,” unpublished.
- [5] R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
- [6] Y. Yoroza, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetism Japan, p. 301, 1982].
- [7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove the template text from your paper may result in your paper not being published.

A. *L^AT_EX*-Specific Advice

Please use “soft” (e.g., `\eqref{Eq}`) cross references instead of “hard” references (e.g., (1)). That will make it possible to combine sections, add equations, or change the order of figures or citations without having to go through the file line by line.

L^AT_EX does not have precognitive abilities. If you put a `\label` command before the command that updates the counter it’s supposed to be using, the label will pick up the last counter to be cross referenced instead. In particular, a `\label` command should not go before the caption of a figure or a table.

REFERENCES

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first . . .”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

REFERENCES

- [1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.