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*—The authors highlight the punctuality issues faced by long-distance trains in Sweden, with a current rate of 68.3 percent against a target of 95 percent. Passengers cannot know in advance which departures are likely to arrive on time. The authors propose a machine learning-based tool to predict train delays, using historical data from Trafikverket. A backend service utilizing machine learning and the Python Flask web framework, along with a frontend HTML interface, has been developed to allow passengers to stay informed about potential future train delays.**

***Index Terms*—railway, machine learning, random forest**

I. Introduction

The Swedish railway system plays a crucial role in the nation's transportation infrastructure. Managed primarily by the government agency Trafikverket (Swedish Transport Administration) [1], it serves a vital role for commuters, business passengers, and leisure travelers. Notably, Sweden's railway passenger traffic has undergone significant deregulation and adaptation to a commercial market, surpassing many other European countries in this regard. Also, the maintenance being performed on the infrastructure is lower than needed.

In 1988, the Swedish railway landscape was dominated by one single operator: Statens Järnvägar. Since 2001, the organization is a state-owned company called SJ. Fast forward to 2022, and the railway ecosystem has evolved dramatically, with 12 passenger train operators utilizing the rail system. In addition, there are also 15 freight train operators [2]. This diversification has introduced prioritization and coordination complexities between stakeholders in the system.

Timely train services are essential for both daily commuters and occasional travelers. Whether it's the morning rush to work or a leisurely trip, passengers rely on punctual arrivals. Currently, long-distance trains aim for a punctuality rate of 95 percent. However, the reality falls short, and the actual punctuality figure remains a critical concern. For long-distance trains covering all operators in Sweden, the punctuality rate was only 68.3 percent during the first quarter of 2024 [3].

Collaboration between Trafikverket, passenger and freight train operators, and academic institutions seeks to address this challenge. Their much-needed joint efforts focus on improving punctuality, but until now, passengers lacked a reliable tool to predict whether their chosen departure would arrive on time. This project aims to fill that gap by developing a passenger train delay predictor, providing travelers with valuable information for improved planning possibilities.

II. Background and related work

One of the authors is employed at SJ. With this connection, the authors saw an opportunity to solve a lack of information problem with a machine learning engineering project. Also, the choice of topic offered available subject matter expertise [4].

Trafikverket gathers detailed information about all trains in their railway system. Information about passenger trains in Sweden is available through an application programming interface (API). The API is open to the public at no cost, only requiring the user to register on their site [5].

A literature review shows that work in this field is aimed at professionals in the business and researchers in academia. The material addresses both the active side of train scheduling, such as traffic planning and optimization measures [6], and the passive side, such as predicting delay without opportunity to affect the outcome [7].

The analysis methods have so far mainly been from the mathematical and statistical toolbox. Statistical regression is straightforward to understand but is limited in modelling complex and nonlinear relationships. In recent years methods from conventional machine learning have gained ground, but these are less interpretable and requires human-engineered features. Neural networks go further by learning automatically and giving flexibility to integrate different architectures into hybrid models [8].

III. Project description

Creation of a web-service aimed at informing train passengers planning to travel with SJ. The service predicts whether a specific train departure in the future will arrive on time or be delayed, based on historical data from Trafikverket.

IV. Project execution

A. Data collection

The team started with collecting information from Trafikverket's API through a download that SJ had done to a csv-file. In the next phase, the team instead downloaded the information directly from the API.

B. Main requirements

1) Data Requirements - input data specification:
`id11` Use publicly available historical rail data from the Swedish Transport Administration.

`id21` Historical timetable data, departure times and arrival times.
`id23` Use data for SJ's long-distance trains.
`id24` The front-end application shall prompt the user to choose from a list a departing station and an arrival station.

2) Functional Requirements:

`id31` The system shall predict if a train between two specific stations will arrive on time. Trafikverket's definition is that a deviation of more than 5:59 min after planned time is a delay.

3) Performance Requirements:

`id41` The model shall achieve a minimum accuracy of 90 percent.

4) System Requirements - Scalability:

`id55` The system shall be designed to scale with an increase in data volume (increasing the time period analyzed) and complexity (other departures, destinations and operators)

5) Compliance and standards:

`id61` Data privacy - The system shall use only publicly available data from Trafikverket, so avoid issues with data protection regulations.
`id65` Reporting - The system shall generate a response informing the user if the model predicts that a train between the named locations will arrive to the end location 1) on time 2) delayed.

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 phases. 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 (Representational State Transfer) API is in the format of JSON (JavaScript Object Notation) and contains nested JSON structure. This JSON content is then normalized into 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 to 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 to filter out unnecessary columns and process the meaningful ones to the format expected by the machine learning model. 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. 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.

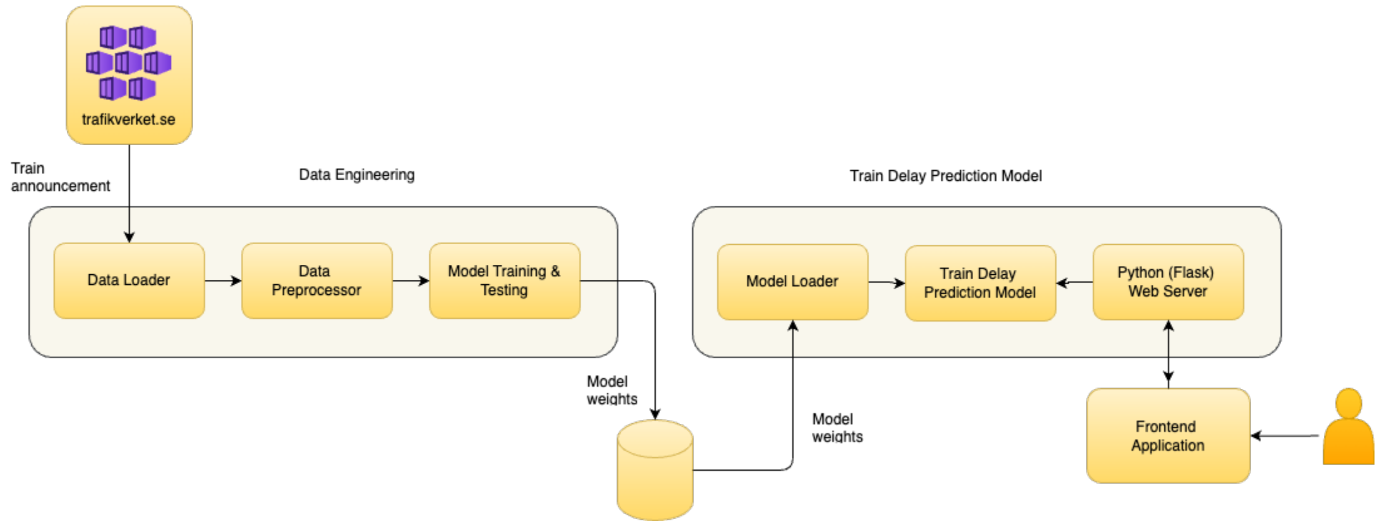


Fig. 1. Overall architecture.

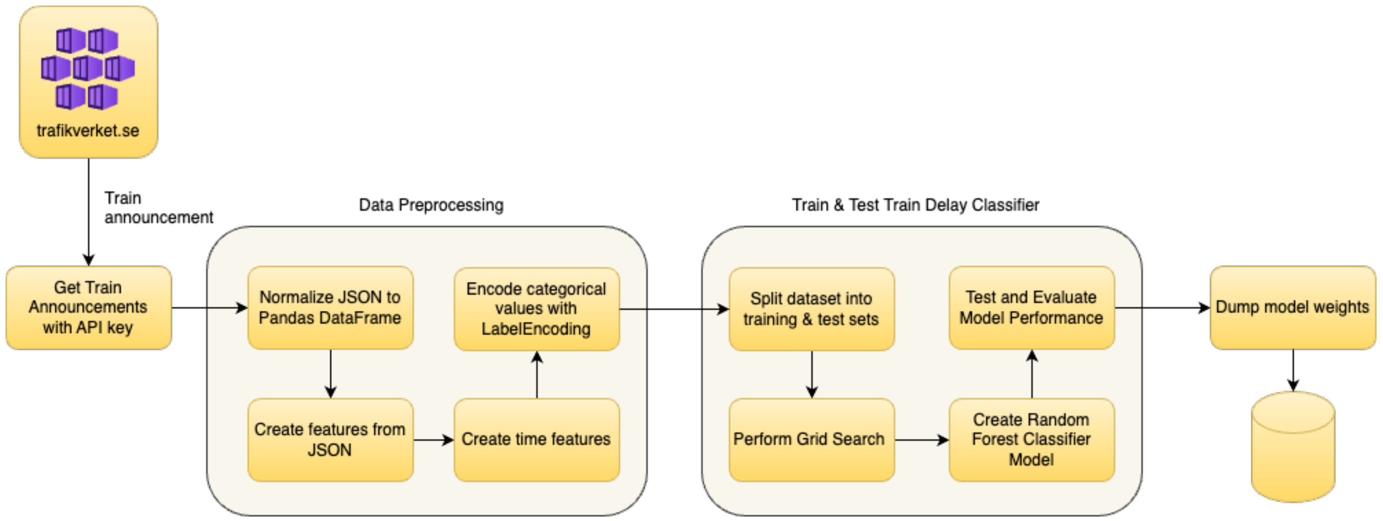

trafikverket.se
Train announcement
Get Train Announcements with API key
Data Preprocessing
Normalize JSON to Pandas DataFrame
Encode categorical values with LabelEncoding
Create features from JSON
Create time features
Train & Test Train Delay Classifier
Split dataset into training & test sets
Test and Evaluate Model Performance
Perform Grid Search
Create Random Forest Classifier Model
Dump model weights

Fig. 2. Data engineering pipeline.

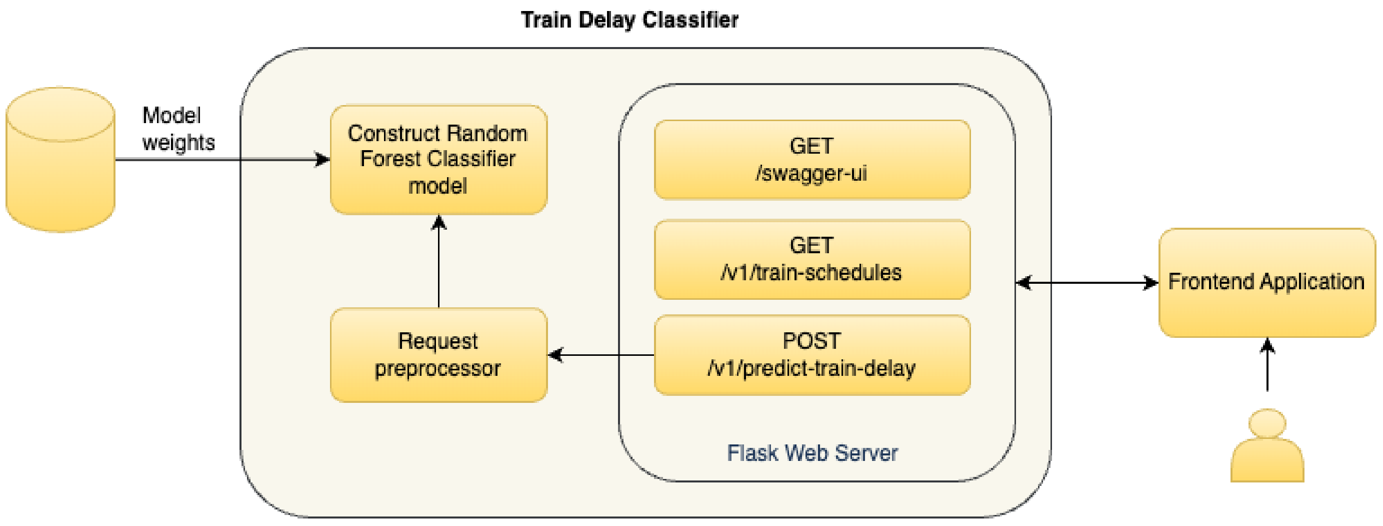


Fig. 3. Architecture Diagram of Train Delay Prediction Model.

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.
LocationDateTimeOTN	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.

It 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 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*

During the design phase of the software, various test strategies were considered such as unit testing of the Python backend and HTML/JavaScript front-ends, testing REST APIs with CURL or Postman tools, end to end integration testing by connecting frontend and back-ends, regression testing when new features are developed and deployed. All test strategies were implemented and executed in the project except regression testing as no new feature was developed after the initial project. 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 multi-branch 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 provides the API and publishes the XSD schema for the data model based on the API version. When the API is updated, Trafikverket will notify users, necessitating updates to the backend service implementation. If features used in the machine learning prediction model are removed from the data model, the model must be updated to reflect the new feature set and tested to ensure performance metrics are met. This process requires regular maintenance.

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 have 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.

By applying GridSearch, overall accuracy of the model is achieved 100 percent.

Table 2 shows the confusion matrix where precision, recall and F1-scores are described. It can be seen that the model achieved perfect score with limited test dataset [9].

TABLE II
Confusion matrix

Label	Precision	Recall	F1-Score
1 (Delay)	1	1	1
0 (On-time)	1	1	1

VI. Conclusions

The train arrival predictor tool is a novel service that helps passengers choose a departure with high chance of timely arrival. This type of service is not yet available to the public though other channels. If the planned departure is expected to arrive late, this tool can help passengers prepare with extra snacks and patience to handle the delay.

One difficulty in choosing a training set is that there are many factors that affect whether a train arrives on time. Delays can occur due to the activity of other trains. Other causes are issues that surface quickly, such as problems with the infrastructure, such as damaged rails or electricity supply.

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. Also, adding all passenger train operators is a possible development.

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.

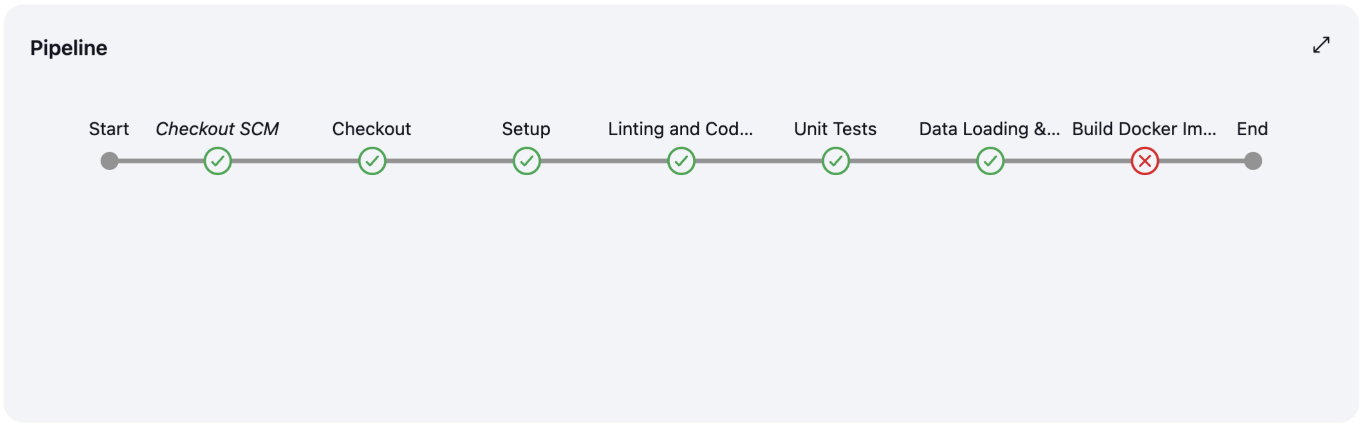
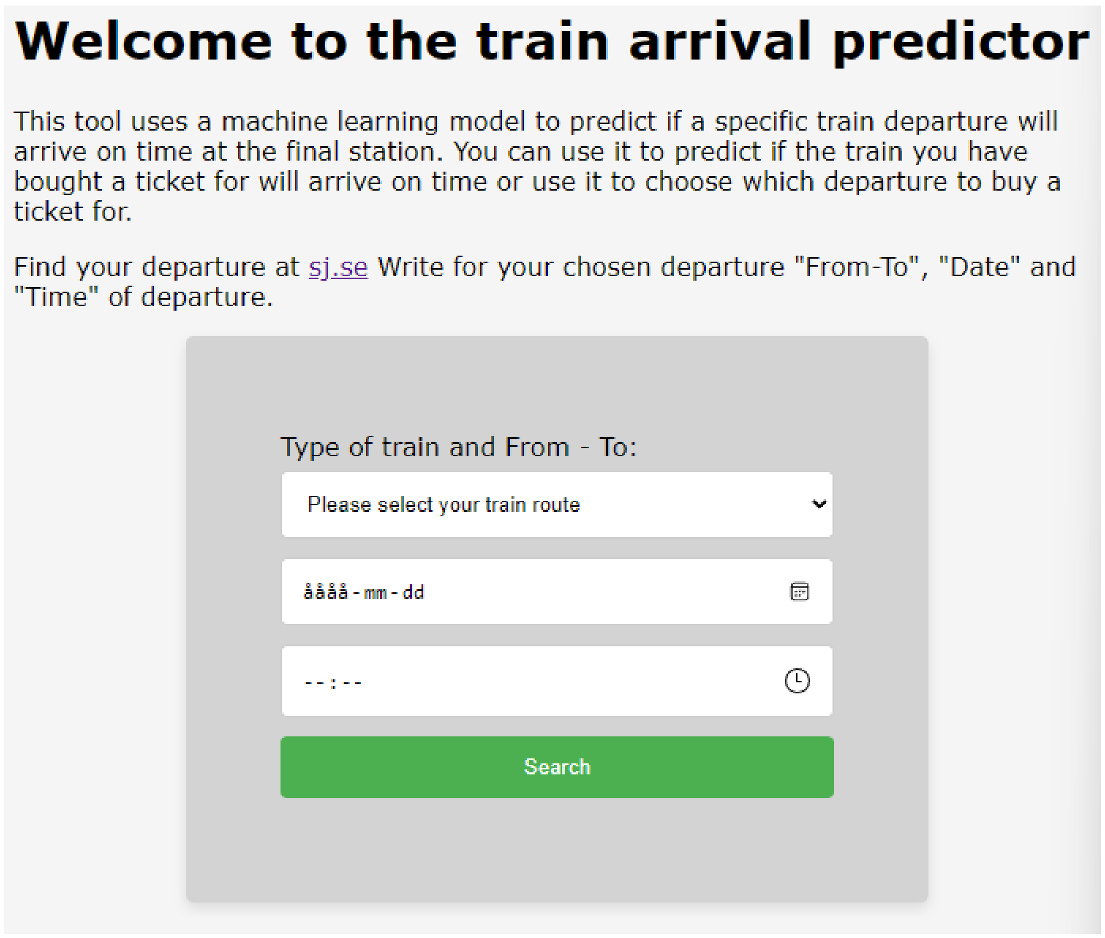


Fig. 4. CI/CD pipeline of data engineering.



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:

Fig. 5. The user interface.

References

[1] *Trafikverket: Vem gör vad av myndigheterna*, May 2024. [Online]. Available: https://www.trafikverket.se/om-oss/var-verksamhet-vision-och-uppdrag/vem-gor-vad-av-myndigheterna/.

[2] *Trafikverket: Järnkoll på tågresor*, May 2024. [Online]. Available: https://www.trafikverket.se/resa-och-trafik/jarnvag/jarnkoll–fakta-om-svensk-jarnvag/jarnkoll-pa-tagresor/.

[3] *Trafikanalys*, May 2024. [Online]. Available: https://www.trafa.se/bantrafik/punktlighet-pa-jarnvag/.

[4] E. Klasson Svensson and P. Cederström at SJ, private communication with subject matter experts, Feb - May 2024.

[5] *Trafikverket's Data Exchange Portal*, Feb - May 2024. [Online]. Available: https://data.trafikverket.se/documentation/api-railway.

[6] J. Törnquist Krasemann, private communication, Apr. 30, 2024.

[7] D. Jakobsson et al., "AIRT AI-baserad Realtidsprognostisering av Trafikinformation," March 20, 2023. https://fudinfo.trafikverket.se/fudinfoexternWebb/pages/PublikationVisa.aspx?PublikationId=6348

[8] T. Kah Yong, "Data Driven Methods for Train Delay Prediction," presented at the Kapacitet i Järnvägstrafiken (KAJT) spring seminar, Borlänge, Sweden, May 13, 2024.

[9] All code for the project is available for invited e-mail addresses at https://github.com/ziaulchowdhury/machine-learning-engineering-bth/tree/master