Truck Delay Classification: Building an End-to-End ML Pipeline

Atul Kumar Maurya(23M1517)

Introduction

Truck delays pose significant challenges to the logistics industry, affecting operational efficiency, customer satisfaction, and profitability. The ability to predict delays accurately can lead to improved resource allocation, better route planning, and reduced costs associated with penalties and customer compensations. This project is the first installment in a three-part series aimed at addressing this critical issue by building an end-to-end machine learning (ML) pipeline for truck delay classification.

In this initial phase, the project focuses on the foundational tasks necessary to build the pipeline, including setting up databases, performing exploratory data analysis (EDA), and utilizing feature stores for data management. This report delves into the various steps involved in the process, from database setup to data preprocessing and feature engineering, providing a comprehensive overview of the project's approach and objectives.

Project Objectives

The primary objective of this project is to build a robust ML pipeline capable of predicting truck delays with high accuracy. By leveraging advanced tools and technologies such as AWS Redshift, Hopsworks feature store, and AWS SageMaker, the project aims to:

- **Improve Operational Efficiency**: By predicting potential delays, logistics companies can better allocate their resources, ensuring that trucks and drivers are utilized optimally.
- **Enhance Customer Satisfaction**: Reliable delivery schedules are crucial for maintaining customer trust. Accurate predictions allow companies to provide more precise delivery windows, reducing uncertainty for customers.
- **Optimize Route Planning**: By incorporating traffic data and weather conditions into the predictive model, companies can plan more efficient routes, minimizing the chances of delays.
- **Reduce Costs**: Delays often result in penalties or the need to compensate customers. By mitigating these delays, companies can save significantly on such costs.

Approach and Methodology

The project adopts a structured approach, beginning with database setup and culminating in data preprocessing and feature engineering. Below are the key steps involved:

Introduction to End-to-End Pipelines

The project starts with an understanding of end-to-end ML pipelines, emphasizing their importance in automating and streamlining the entire machine learning process. An end-to-end pipeline ensures that data flows smoothly from collection and preprocessing to model training and deployment, making the system scalable and maintainable.

Database Setup

The first technical task involves setting up the necessary databases using AWS RDS instances for MySQL and PostgreSQL. MySQL Workbench and pgAdmin4 are employed for database management, allowing for efficient data storage and retrieval. AWS RDS provides a scalable and secure environment for managing the large volumes of data typically involved in logistics operations.

Data Analysis

With the databases in place, the project proceeds to perform data analysis using SQL on MySQL Workbench and pgAdmin4. This step is crucial for understanding the underlying patterns and characteristics of the data, which will inform subsequent steps in the pipeline.

AWS SageMaker Setup

AWS SageMaker, a powerful tool for building, training, and deploying machine learning models at scale, is set up to facilitate the model development process. This setup includes configuring the environment and integrating it with other AWS services for seamless data flow.

Exploratory Data Analysis (EDA)

EDA is conducted to gain insights into the data's essential features and characteristics. This step involves visualizing data distributions, identifying correlations, and detecting potential outliers or anomalies. EDA provides a foundation for feature selection and engineering, which are critical for building a robust predictive model.

Feature Store

The concept of a feature store is introduced, with a focus on Hopsworks. A feature store is a centralized repository for storing and managing features used in machine learning models. It ensures consistency across different models and facilitates the reuse of features, improving efficiency. Hopsworks is employed to create the project, manage feature groups, and store the engineered features.

Data Retrieval from Feature Stores

After creating the feature store, the next step is to retrieve the necessary data for further analysis. This data will be used to train the predictive model, making it essential to ensure that the retrieved data is accurate and relevant.

Data Preprocessing and Feature Engineering

Data preprocessing involves cleaning and transforming the data to make it suitable for model training. This step includes handling missing values, encoding categorical variables, and normalizing numerical features. Feature engineering, on the other hand, involves creating new features or modifying existing ones to enhance the model's predictive power. The final engineered features are then stored in the feature store for consistency and easy access.

0.1 Model Building and Experimentation

Experiment tracking involves systematically recording and managing details related to each model experiment, including hyperparameters, metrics, and data versions. Benefits: Ensures reproducibility, supports comparison of different models, facilitates collaboration, and aids in informed decision-making.

A Model Registry is a centralized system for managing, versioning, and tracking machine learning models throughout their lifecycle. Benefits: Enables version control, streamlines collaboration, provides deployment features, maintains metadata, and enhances reproducibility.

Classification Evaluation Metrics

Classification evaluation metrics are used to evaluate the performance of a machine learning model that is trained for classification tasks. Some of the commonly used classification evaluation metrics are F1 score, recall score, confusion matrix, and ROC AUC score. Here's an overview of each of these metrics:

F1 score: The F1 score is a metric that combines the precision and recall of a model into a single value. It is calculated as the harmonic mean of precision and recall, and is expressed as a value between 0 and 1, where 1 indicates perfect precision and recall. F1 score is the harmonic mean of precision and recall. It is calculated as follows:

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

where precision is the number of true positives divided by the sum of true positives and false positives, and recall is the number of true positives divided by the sum of true positives and false negatives.

Recall: Use the recall score when the cost of false negatives (i.e., missing instances of a class) is high. For example, in a medical diagnosis problem, the cost of missing a positive case may be high, so recall would be a more appropriate metric. Recall score (also known as sensitivity) is the number of true positives divided by the sum of true positives and false negatives. It is given by the following formula:

$$Recall = \frac{TP}{TP + FN}$$

Precision: Precision is another important classification evaluation metric, which is defined as the ratio of true positives to the total predicted positives. It measures the accuracy of positive predictions made by the classifier, i.e., the proportion of positive identifications that were actually correct. The formula for precision is:

$$precision = \frac{true\ positive}{true\ positive + false\ positive}$$

where true positive refers to the cases where the model correctly predicted the positive class, and false positive refers to the cases where the model incorrectly predicted the positive class. Precision is useful when the cost of false positives is high, such as in medical diagnosis or fraud detection, where a false positive can have serious consequences. In such cases, a higher precision indicates that the model is better at identifying true positives and minimizing false positives.

Confusion Matrix: A confusion matrix is a table that is often used to describe the performance of a classification model. It compares the predicted labels with the true labels and counts the number of true positives, false positives, true negatives, and false negatives. Here is an example of a confusion matrix:

— — Actual Positive — Actual Negative — ————-————————-————————-— — Predicted Positive — True Positive (TP) — False Positive (FP) — — Predicted Negative — False Negative (FN) — True Negative (TN) —

ROC AUC Score ROC AUC (Receiver Operating Characteristic Area Under the Curve) score is a measure of how well a classifier is able to distinguish between positive and negative classes. It is calculated as the area under the ROC curve. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. TPR is the number of true positives divided by the sum of true positives and false negatives, and FPR is the number of false positives divided by the sum of false positives and true negatives.

$$ROC\ AUC\ Score = \int_0^1 TPR(FPR^{-1}(t))dt$$

where FPR^{-1} is the inverse of the FPR function.

The choice of evaluation metric depends on the specific requirements of the business problem. Here are some general guidelines:

F1 score: Use the F1 score when the class distribution is imbalanced, and when both precision and recall are equally important.

Recall score: Use the recall score when the cost of false negatives (i.e., missing instances of a class) is high. For example, in a medical diagnosis problem, the cost of missing a positive case may be high, so recall would be a more appropriate metric.

* Precision: Precision is useful when the cost of false positives is high, such as in medical diagnosis or fraud detection, where a false positive can have serious consequences. In such cases, a higher precision indicates that the model is better at identifying true positives and minimizing false positives.

* Confusion matrix: The confusion matrix is a versatile tool that can be used to visualize the performance of a model across different classes. It can be useful for identifying specific areas of the model that need improvement.

* ROC AUC score: Use the ROC AUC score when the ability to distinguish between positive and negative classes is important. For example, in a credit scoring problem, the ability to distinguish between good and bad credit risks is crucial.

Importance with respect to the business problem:

The importance of each evaluation metric varies depending on the business problem. For example, in a spam detection problem, precision may be more important than recall, since false positives (i.e., classifying a non-spam email as spam) may annoy users, while false negatives (i.e., missing a spam email) may not be as harmful. On the other hand, in a disease diagnosis problem, recall may be more important than precision, since missing a positive case (i.e., a false negative) could have serious consequences. Therefore, it is important to choose the evaluation metric that is most relevant to the specific business problem at hand.

A Model Registry is a centralized system for managing, versioning, and tracking machine learning models throughout their lifecycle. Benefits: Enables version control, streamlines collaboration, provides deployment features, maintains metadata, and enhances reproducibility.

Data Storage

The final step in this phase is to store the engineered features in the feature store. This ensures that the data is readily available for model training and can be easily accessed and reused for future projects. Storing the features in a centralized repository also helps maintain consistency across different models and versions, reducing the likelihood of errors.

Conclusion

The first phase of the truck delay classification project lays the groundwork for building a robust and scalable ML pipeline. By setting up the necessary databases, conducting EDA, and utilizing a feature store, the project ensures that the data is well-prepared for model training in the subsequent phases. The approach adopted in this project emphasizes the importance of a structured and methodical process in building ML pipelines, ensuring that each step contributes to the overall objective of accurately predicting truck delays.

The successful completion of this phase will enable logistics companies to improve their operational efficiency, enhance customer satisfaction, optimize route planning, and reduce costs associated with delayed shipments. As the project progresses into the next phases, the focus will shift towards model building, evaluation, and deployment, ultimately leading to a comprehensive solution for truck delay prediction.