# Learn to Build an End-to-End Machine Learning Pipeline - Part 1

## Overview

This project is the first part of a three-part series aimed at solving the truck delay prediction problem. In this initial phase, we will utilize PostgreSQL and MySQL in AWS RDS to store the data, set up an AWS Sagemaker Notebook, perform data retrieval, conduct exploratory data analysis, create feature groups with Hopsworks, data processing, and feature engineering. We aim to build a pipeline that prepares the data for model building.

## Problem Statement

The project addresses a critical challenge faced by the logistics industry. Delayed truck shipments not only result in increased operational costs but also impact customer satisfaction. Timely delivery of goods is essential to meet customer expectations and maintain the competitiveness of logistics companies. By accurately predicting truck delays, logistics companies can:

- Improve operational efficiency by allocating resources more effectively
- Enhance customer satisfaction by providing more reliable delivery schedules
- Optimize route planning to reduce delays caused by traffic or adverse weather conditions
- Reduce costs associated with delayed shipments, such as penalties or compensation to customers

## Topics

The following topics are covered in this

## Prerequisite

- Basic understanding of python and Machine Learning

- Basic understanding of SQL
- AWS Account
- MySQL Workbench
- pgAdmin4

### AWS Usage Charges

This project leverages the AWS cloud platform to build the end-to-end machine learning pipeline. While using AWS services, it's important to note that certain activities may incur charges. We recommend exploring the AWS Free Tier, which provides limited access to a wide range of AWS services for 12 months. Please refer to the AWS Free Tier page for detailed information, including eligible services and usage limitations. If you have exhausted the free tier services you may need to pay charges as per your usage.

## Folder structure

```
├─ Data
│  ├─ Database_backup
│  │  ├─ truck-eta-mysql.sql
│  │  └─ truck-eta-postgres.sql
│  ├─ data_description.pdf
│  └─ Training_data
│     ├─ city_weather.csv
│     ├─ drivers_table.csv
│     ├─ routes_table.csv
│     ├─ routes_weather.csv
│     ├─ traffic_table.csv
│     ├─ trucks_table.csv
│     └─ truck_schedule_table.csv
├─ Notebook
│  └─ Truck_Delay_Classification.ipynb
├─ References
│  ├─ readme.md
│  ├─ requirements.txt
│  └─ solution methdology.pdf
└─ SQL Commands
├─ truck-delay-mysql.sql
└─ truck-delay-postgres.sql
```

## Data Dump in DB

### How to create RDS instance?

Creating a MySQL RDS (Relational Database Service) instance in Amazon Web Services (AWS) involves several steps. Below is a general outline of the process to guide you through setting up a MySQL RDS instance:

- **Step 1: Sign in to your AWS Account**

If you don't already have an AWS account, you'll need to create one. Once you're signed in, follow these steps:

- **Step 2: Open the RDS Dashboard**

  Navigate to the AWS Management Console and search for "RDS" or find it in the "Database" section.

- **Step 3: Launch a DB Instance**

  Click the "Create database" button. Choose the "Standard create" option.

- **Step 4: Choose Engine Options**

  Select "MySQL"(or PostgreSQL) as the engine type. Choose the appropriate version of MySQL(Recommended: 8.0.33) or PostgreSQL(Recommended: 14.7-R1)

- **Step 5: Specify DB Details**

  Choose the use case that best describes your deployment scenario (Recommended: Production). Configure your DB instance settings:

  - DB instance class: Choose the instance type based on your performance and resource requirements
  - Multi-AZ deployment: Choose Single DB instance(or as per your requirements)
  - Storage: Set the storage capacity and type (General Purpose SSD, Provisioned IOPS, Magnetic).

  Provide a DB instance identifier and a master username and password. These will be used to access the MySQL/PostgreSQL database.

  Recommended: 100 GB storage, 1000 IOPS

- **Step 6: Configure Advanced Settings**

  Here, you can configure additional settings such as network, security group, database name, and backup retention. You can also enable automatic backups and set the maintenance window.

  Make sure you are giving public access to the DB so that you can connect with the DB otherwise you need to add specific IP's before accessing.

- **Step 7: Review and Create**

  Review your configuration settings and make sure everything is as you intend.

- **Step 8: Create the DB Instance**

  Click the "Create database" button to create your MySQL RDS instance. The instance creation process might take several minutes.

## Connect with DB

### Connect with MySQL RDS using MySQL Workbench

To connect to a MySQL RDS instance using MySQL Workbench, follow these steps:

- **Step 1: Open MySQL Workbench**

  After installing MySQL Workbench, open the application.

- **Step 2: Create a New Connection**

  Click on the "+" icon under the "MySQL Connections" section in the home screen of MySQL Workbench. Enter a connection name for your RDS instance (e.g., "truck-delay-classification").

  Enter the following details for the connection:

  - Hostname: The endpoint of your RDS instance (can be found in the RDS dashboard)
  - Port: Usually 3306 for MySQL
  - Username: The master username you specified when creating the RDS instance
  - Password: The password for the master username

  Click the "Test Connection" button to ensure that your MySQL Workbench can connect to the RDS instance successfully. If the test connection is successful, click the "OK" button to save the connection settings.

- **Step 3: Connect to the RDS Instance**

  In the MySQL Workbench home screen, you'll see your newly created connection under "MySQL Connections." Double-click on the connection name or select it and click the "Connect" button. MySQL Workbench will connect to your RDS instance, and you'll be able to explore and manage your database using the interface.

### Connect with PostgreSQL RDS using pgAdmin4

To connect to a PostgreSQL RDS instance using pgAdmin 4, follow these steps:

- **Step 1: Open pgAdmin 4**

  After installing pgAdmin 4, open the application.

- **Step 2: Add a New Server**

  In pgAdmin 4, click on the "Add New Server" button in the "Quick Links" section of the home screen. In the "General" tab, provide a name for your server connection (e.g., "truck-delay-classification").

- **Step 3: Configure Connection Details**

  Switch to the "Connection" tab. Enter the following connection details for your PostgreSQL RDS instance:

  - Host name/address: The endpoint of your RDS instance (can be found in the RDS dashboard).
  - Port: Usually 5432 for PostgreSQL.
  - Maintenance database: This is typically the default database that comes with your RDS instance. You might need to check this information in your RDS settings
  - Username: The master username you specified when creating the RDS instance.
  - Password: The password for the master username.

Click the "Save Password" checkbox if you want pgAdmin 4 to remember your password. Click the "Save" button to save the connection settings.

- **Step 4: Connect to the RDS Instance**

  In the pgAdmin 4 home screen, you'll see your newly added server connection under "Servers." Double-click on the server connection name or select it and click the "Connect" button. pgAdmin 4 will connect to your PostgreSQL RDS instance, and you'll be able to explore and manage your database using the interface.

## Data Analysis using SQL

### Data Analysis - MySQL Workbench

- **Step 1: Launch MySQL Workbench** Open MySQL Workbench on your computer.

- **Step 2: Connect to Your MySQL Server** In the "MySQL Connections" window, connect to your MySQL server by double-clicking the existing connection or creating a new one if necessary. Enter the connection details like hostname, username, password, and port as required.

- **Step 3: Open the SQL File** Click on the "File" menu at the top left corner of MySQL Workbench and select "Open SQL Script." Navigate to the location of your SQL file and select it. Make sure you have downloaded the SQL file from the download code section. This will open the SQL file in the MySQL Workbench editor.

  Alternatively, you can go to the "File" menu and select "New SQL Tab" to create a new tab for your SQL file.

- **Step 4:Execute the SQL Script** To execute the SQL script, you have several options:

  - Click the lightning bolt icon in the toolbar (labeled "Execute SQL Script").
  - Right-click within the SQL script editor and choose "Execute" from the context menu.
  - Use the keyboard shortcut `Ctrl + Enter` (Windows/Linux) or `Command + Enter` (macOS).

- **Step 5: View Execution Results:** MySQL Workbench will execute the SQL script, and the results will be displayed in the "Query Results" tab at the bottom of the SQL script editor. If there are any errors or warnings, they will be shown here.

### Data Analysis - pgAdmin4

- **Step 1: Open pgAdmin 4**

  Launch pgAdmin 4 and make sure you are connected to your PostgreSQL database server.

- **Step 2: Open the Query Tool**

  - In the navigation panel on the left, expand your server group and click on the server to which you want to upload the SQL file.

  - Under the selected server, expand the "Databases" node and select the specific database where you want to run the SQL script.

  - Right-click on the database, and from the context menu, select "Query Tool." This will open the Query Tool window.

- **Step 3: Load the SQL File**

  - In the Query Tool window, you'll see a toolbar with various options. Look for the "Open File" button (usually represented by an open folder icon). Click this button to open a file dialog.

  - Make sure you have downloaded the SQL file from the download code section. Navigate to the location of your SQL file, select it, and click the "Open" button in the file dialog. This will load the SQL script into the Query Tool's editor.

  - You can also open new sql file to write your own queries.

- **Step 4: Execute the SQL Script**

  - Once you're ready to execute the SQL script, click the "Execute" button in the Query Tool's toolbar. This button is typically represented by a green triangle icon.

  - Alternatively, you can use the keyboard shortcut `F5` to execute the script.

- **Step 5: View Execution Results** The Query Tool will execute the SQL script, and the results will be displayed in the output pane below the editor. You can see the execution status, any error messages, and the query results.

## AWS Sagemaker Setup

To create an Amazon SageMaker notebook instance and open Jupyter Notebook in it, follow these steps:

- **Step 1: Sign in to the AWS Management Console**

  - Go to the AWS Management Console.
  - Sign in with your AWS credentials.

- **Step 2: Open SageMaker Service**

  In the AWS Management Console, search for "SageMaker" in the search bar or navigate to "Services" > "Machine Learning" > "Amazon SageMaker."

- **Step 3: Create a Notebook Instance**

  - In the SageMaker console, click on "Notebook instances" in the left sidebar.

  - Click the "Create notebook instance" button.

- **Step 4: Configure the Notebook Instance**

  In the "Notebook instance settings" section, provide the following information:

  - Notebook instance name: Give your notebook instance a name.
  - Notebook instance type: Select the instance type that suits your needs.
  - IAM role: Choose an existing IAM role or create a new one with SageMaker permissions.
  - Click the "Create notebook instance" button.

- **Step 5: Wait for Notebook Creation**

  SageMaker will create the notebook instance. It may take a few minutes. You'll see the status change from "Pending" to "InService" when it's ready.

- **Step 6: Access Jupyter Notebook**

  Once the notebook instance is in the "InService" state, click on "Open Jupyter" next to the instance name. Jupyter Notebook will open in a new tab or window in your browser.

  - To upload a Jupyter Notebook file, click the "Upload" button in the top-left corner of the JupyterLab interface. It looks like an upward-pointing arrow.
  - Navigate to the location of your Jupyter Notebook file on your local machine and select it.
  - Click the "Upload" button in the file dialog to upload the notebook file to your SageMaker instance.

- **Step 7: Start Using Jupyter Notebook**

  You are now inside Jupyter Notebook running on your SageMaker notebook instance. You can create new notebooks, upload existing ones, and start running code and machine learning experiments.

- **Step 8: Stop the Notebook Instance (Important)**

  When you're done using the notebook instance, make sure to stop it to avoid incurring unnecessary charges. You can stop it by going back to the SageMaker console, selecting the instance, and clicking the "Stop" button.

## Execution Instructions - Local

### Python version 3.10.2

To create a virtual environment and install requirements in Python 3.10.2 on different operating systems, follow the instructions below:

### For Windows:

- Open the Command Prompt by pressing Win + R, typing "cmd", and pressing Enter.
- Change the directory to the desired location for your project:

  ```
  cd C:\path\to\project
  ```

- Create a new virtual environment using the venv module:

  ```
  python -m venv myenv
  ```

  Note: Replace myenv with your desired virtual environment name.

- Activate the virtual environment:

  ```
  myenv\Scripts\activate
  ```

- Install the project requirements using pip:

  ```
  pip install -r requirements.txt
  ```

### For Linux/Mac:

- Open a terminal.
- Change the directory to the desired location for your project:

  ```
  cd /path/to/project
  ```

- Create a new virtual environment using the venv module:

  ```
  python3 -m venv myenv
  ```

  Note: Replace myenv with your desired virtual environment name.

- Activate the virtual environment:

  ```
  source myenv/bin/activate
  ```

- Install the project requirements using pip:

  ```
  pip install -r requirements.txt
  ```

Note: These instructions assume you have Python 3.10.2 installed and added to your system's PATH variable.

## Execution Instructions if Multiple Python Versions Installed

If you have multiple Python versions installed on your system , you can use the Python Launcher to create a virtual environment with Python 3.10.2, you can specify the version using the -p or --python flag. Follow the instructions below:

### For Windows:

- Open the Command Prompt by pressing Win + R, typing "cmd", and pressing Enter.
- Change the directory to the desired location for your project:

  ```
  cd C:\path\to\project
  ```

- Create a new virtual environment using the Python Launcher:

  ```
  py -3.10 -m venv myenv
  ```

  Note: Replace myenv with your desired virtual environment name.

- Activate the virtual environment:

  ```
  myenv\Scripts\activate
  ```

- Install the project requirements using pip:

  ```
  pip install -r requirements.txt
  ```

### For Linux/Mac:

- Open a terminal.
- Change the directory to the desired location for your project:

  ```
  cd /path/to/project
  ```

- Create a new virtual environment using the Python Launcher:

  ```
  python3.10 -m venv myenv
  ```

  Note: Replace myenv with your desired virtual environment name.

- Activate the virtual environment:

  ```
  source myenv/bin/activate
  ```

- Install the project requirements using pip:

  ```
  pip install -r requirements.txt
  ```

  By specifying the version using py -3.10 or python3.10, you can ensure that the virtual environment is created using Python 3.10.2 specifically, even if you have other Python versions installed.

## Notes

- If you do not want to use AWS and want to execute code locally kindly follow the local instructions given. Also make sure to load csv files rather than featching data from RDS. `import pandas as pd df=pd.read_csv('csv_file_path')`
- Kindly change the credentials to connect with RDS before executing the code. The credentials provided in the notebook are dummy and won't work.