CS210 - Data Management for Data Science
Final Project Interim Report
December 2, 2025
By: Rhea Yadav & Kaajal Yadav

Github Repository:

https://github.com/R-Yadav123/Driver-Drowsiness-Detection-Management-System

SQL Queries & Data Visualization Components:

Progressing with the project, background research, initial dataset research, database creation, dataset analysis, and visualization has been completed so far. A Github repository has been created in order to store all code files in an organized way. We have used artificial intelligence in order to generate randomized datasets. This is for the tables we have created within the Driver_Management database in order to generate synthetic data. So far, we have utilized the following standard Python libraries:

```
import sqlite3
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

The tables include: Driver, Driver_Drowsiness_Event, Trip, and Vehicle. The tables include both numerical and categorical data in order to effectively portray the analysis. Once the tables were properly populated with the accurate data we utilized a database tool known as DbVisualizer. Through this tool, we can organize our database through object and connection management.

From this we used various SQL queries in order to return various outputted data. Such SQL queries consist of joins, grouping/aggregation, and counting/filtering. We also explored the datasets by creating various plots in order to visualize the data. For instance, we created a scatterplot to visualize `Miles Traveled vs. Trip Duration` and `Age vs. Daily Avg Driving Hours`, a bar plot to see the `Count of vehicles by model` and `Count of vehicles by vehicle type`, and a boxplot to see `Age vs. Model`.

We have also created a prediction model with a training dataset and a testing dataset for the Driver_Drowsiness_Event table. The testing data has removed the column “Alert_Sent” as that is what is going to be predicted. The column is a boolean value column where the possible values of Alert_Sent are 1 or 0. If the value is 1 this means that the Alert_Sent is true and the alert was sent. If the value is 0 this means that the Alert_Sent is false and the alert was not sent. By fitting a prediction model using linear regression, and a 0.5 threshold we were able to get the predictions with a model accuracy of 58%.

Machine Learning Components:

After the data visualizations have been completed, in the future iterations of the project we will be focusing on the machine learning aspect. In order to both test and train the model, we will be using the datasets we have collected. These datasets contain images that show the different states that a person is in while driving.

`Dataset 1: Mendeley Data - Annotated Drowsiness Detection Dataset Captured Using Raspberry Pi 5`

`Link:` https://data.mendeley.com/datasets/chvz7vh2dc/1?utm_source=chatgpt.com

This will be implemented via Python's Pytorch library. We have decided to choose Pytorch as this is the industry standard and will be useful in developing the model.

`Dataset 2: DMD - Driving Monitoring Dataset via Vicomtech Open Datasets (Drowsiness category)`

`Link:` https://dmd.vicomtech.org/#Download

This will be implemented via Google Teachable Machine.

In the final project report, we will be displaying the code, outputs of the specific queries, analysis of the plots, and in-depth analysis of the machine learning models. In regards to the Google Teachable Machine, we are using this external tool in order to additionally support our data and to provide a further analysis. This allows for a different perspective of classifying images and we will be using it as supplementary material.

Steps for defining the model in teachable machine:

- First collect all data sets
- Sort data into specific classes/labels
- The images itself will be differentiating in terms of if the person's eyes are closed, looking down, eye contact, etc.
- Teachable machine performs the extractions of features, recognizes any specific patterns, and then we will perform multiple iterations to receive higher accuracies