YAP470 - ANALYTICS PROJECT
Predicting Road Accident Risk using Deep Regression on Synthetic Data

Esmanur Ulu - 231101024
Nehir Tıraş - 231101065
Zeynep Yetkin - 231101042

---

1. Clear Description of Our Project

The primary **aim of our project** is to develop a solution for a **Regression** task: predicting the **accident risk** (accident_risk, a continuous value between 0 and 1) based on a large-scale (570K records) synthetically generated tabular dataset from the Kaggle Playground Series. Our approach involves prioritizing the Root Mean Squared Error metric for performance evaluation. A core technical aspect of our project will be the comparative analysis of a traditional regression baseline against a Multi-Layer Perceptron architecture built in PyTorch.

---

2. Short Literature Search

-A STUDY ON TRAFFIC CRASH SEVERITY PREDICTION USING MACHINE LEARNING ALGORITHMS

This study focused on predicting traffic accident severity by testing several classical Machine Learning algorithms, including Random Forest and SVM, on a large dataset of US traffic crashes. The analysis emphasized metrics like Precision, Recall, and F1-Score to determine factors contributing to accidents. This research is directly relevant because it validates our domain accident prediction and provides strong classical Baselines (RF, SVM, etc.) for our project to compare against the performance of our Deep Learning model.

-Neural Network Models for Combined Classification and Regression

This article introduces multi-output neural networks, which can be designed to perform both regression and classification simultaneously using a single input. While our project focuses purely on the regression task, the structure and concepts detailed in this article—such as designing optimal layer connections and selecting appropriate loss functions for continuous output are directly applicable to building and optimizing our Multi-Layer Perceptron for deep regression in PyTorch.

-Deep Neural Networks for Regression Problems

This article is a guide on how to build a Deep Neural Network (DNN) to predict numbers. It teaches us important rules, like using only one final output node and the right error measure for our model. This knowledge is perfect for our project because we need to predict a number. This guide helps us design our PyTorch model correctly, making sure the model's setup is technically sound and ready to start learning from our large dataset.

---

3. Source of the Dataset for Our Project

Kaggle Competition:
https://www.kaggle.com/competitions/playground-series-s5e10/data

---

4. Description of the Dataset Used in Our Project

Domain	Road Safety and Risk Assessment. The objective of our project is to predict the continuous accident risk associated with various road and environmental conditions.
Data Nature	Synthetically-Generated Tabular Data. The dataset (both train and test) was generated by a deep learning model trained on the original Simulated Roads Accident dataset. Feature distributions are close to the original, but not identical.
Data Size	Train: 517,755 rows. Test: 172,586 rows. Our large scale dataset contains a total of 690,341 records (not including the original data).
Target Variable	accident_risk: The continuous ground truth value, ranging from 0 to 1. Our objective is to predict this risk for the test set.
Key Features (Columns)	id, road_type, num_lanes, curvature, speed_limit, lighting, weather, road_signs_present, public_road, time_of_day, holiday, school_season, num_reported_accidents.

5. Platforms, Python Packages, Frameworks, and Hardware We Plan to Use

Our project uses strong, standard tools to handle our large dataset and train our complex models easily.

We chose **PyTorch** to be our main tool for Deep Learning. Our team will use it to build and train all of our numerical prediction models. PyTorch is flexible, which we need for setting up our complex model architecture and customizing our training process on this large synthetic dataset.

For our simpler, Baseline Machine Learning test, we will use **Scikit-learn**. This tool is important for setting up the first model and calculating our main scores, such as RMSE and R2, so we can compare the simple model to the deep learning one.

To manage and clean the data, **Pandas** and **NumPy** are our essential tools. We use them to load the huge dataset and prepare it by scaling numbers and encoding categories before the models start training.

Finally, because our training data is so large, we must use **Google Colab**. This gives us the necessary fast hardware to run our Deep Learning models quickly and correctly, without causing problems on our personal computers.

6. Planned Division of Tasks Among Our Team

	Esmanur Ulu	**Zeynep Yetkin**	**Nehir Tıraş**
Data Preparation & Preprocessing	Data acquisition, initial loading, cleaning, and Exploratory Data Analysis.	Support with numerical scaling/normalization and basic preprocessing steps.	Categorical encoding and basic Feature Engineering.
Model Training & Experimentation	Implementation and training of the Classical Baseline Regression Model.	Implementation, training, and Hyperparameter Optimization of the Core PyTorch MLP Model.	Implementation and training of the Advanced PyTorch MLP Model.
Analysis & Reporting	Writing the Introduction, Data Description, and Baseline Results sections of the final paper.	Writing the Core MLP Model results and the general Model Comparison section.	Writing the Literature Review, Discussion/Interpretation, and Conclusion sections.

Note: This is a structured plan, but task assignments may be adjusted during the process as challenges arise.