CSci 4521 Final Project Proposal

Group Members: Baanee Singh & Shaan Sidhu

In this project, we use machine learning techniques to predict whether a Formula 1 driver will finish on the podium (top 3) in a given race. This task will be framed as a binary classification problem where a driver in positions 1, 2, or 3 will be given the label 1, and a driver in positions 4 or lower will be given the label 0. Predictions will be based on various features such as historical performance, constructor team, qualifying position, race location, and other statistics. Along with this, we will be testing different models to see which model would perform the best and why.

The dataset we will use is the Formula 1 World Championship 1950-2020 dataset from Kaggle, which includes races (metadata about each race and circuit), driver information, constructor team data, results of each race (postions, points, laps), starting grid position, and other features (status, pit stops, lap times). We will be cleaning and merging the relevant tables in order to fit our goal of predicting whether a driver will be on the podium.

Several projects have used this dataset, focusing on various analytical and predictive tasks. For example, the projects *"Formula 1 Analysis 1995–2025"* and *"F1 EDA"* examine trends in driver and team performance. *"PitStops"* and *"Basic Race & Pit Stop Analysis"* analyze race strategy and its impact on the outcomes of the races. Other projects attempt to predict race outcomes using machine learning models like Random Forest, similar to our approach. These projects achieve success, but the method by which they do their classification differs from our attempt.

For this project, we plan to implement two models: a PyTorch-based neural network and a classic machine learning model for comparison. The primary model will be a multi-layer model built using PyTorch. This neural network will take in a feature vector composed of relevant statistics. The architecture will consist of 3 to 4 fully connected layers with ReLU activation functions, and we will incorporate regularization techniques such as dropout and batch normalization to improve generalization and training stability. The output layer will use a single sigmoid unit to perform binary classification, predicting whether a driver will finish on the podium. Key hyperparameters we intend to tune include the learning rate, dropout rate, batch size, number of neurons per layer, and the type of optimizer used.

To provide a baseline and allow for comparison, we will also train a another machine learning model, specifically, a Random Forest classifier. This model is well-suited for structured data and provides strong interpretability. For the Random Forest, we will experiment with hyperparameters such as the number of decision trees, maximum depth of each tree, the number of features considered at each split, and the minimum number of samples required at a leaf node. Comparing the performance of both models will help us determine the strengths and limitations of each approach in the context of predicting podium finishes in Formula 1 races.

To evaluate our models, we will employ a range of metrics. Accuracy will be used as a general indicator of model performance, providing an overall sense of how often predictions are

correct. However, given the imbalance between podium (top 3) and non-podium finishes in Formula 1 races, precision and recall are especially important. Precision will help us assess how many of the drivers predicted to finish on the podium do, while recall will indicate how many actual podium finishes were correctly identified. We will also use the F1 score, which is particularly useful for imbalanced classification problems. Additionally, the ROC-AUC score will be used to evaluate the model's performance across different classification thresholds, offering insight into how well the model distinguishes between classes. A confusion matrix will further break down the number of true positives, true negatives, false positives, and false negatives, giving a detailed view of the model's classification behavior.

For our final report and presentation, we will include a diagram of our neural network, illustrating the input layer, hidden layers, and output layer to clarify the structure of our PyTorch model. With our neural network, we will show the testing and training loss to show how the model's loss changes over our decided number of epochs. We will also display confusion matrix visualizations to better communicate the distribution of correct and incorrect classifications. ROC curves will be plotted to compare the performance of both models across thresholds. Finally, we will present summary statistics to provide an overview of the dataset's composition and help inform our modeling decisions.

Given the division of tasks, we will likely meet once a week to discuss any updates or questions we may have. We will communicate using messages to get timely responses. Shaan will be focusing primarily on the neural network, while I will be handling the Random Forest Model. We will work together on loading the dataset, identifying relevant features, and creating the binary classification before splitting up our tasks from there. We will be responsible for creating the necessary metrics for each model and providing the explanation for the final report (Shaan will do the write-up/metrics for the neural network, and I will do the write-up/metrics for the Random Forest Classifier). Similarly, for creating the project poster, we will each individually put in relevant information from our specific parts of the project, and work together on explanations that are relevant to the project as a whole. For the presentation of our project, we will work together to create a speech that evenly divides the tasks and analysis between the two of us.

If there are any challenges to the project, we will fall back on the following options. If training the PyTorch neural network proves to be unstable or results in significant overfitting, we will explore simpler alternatives such as logistic regression or a smaller multi-layer model with fewer layers and parameters. In terms of features, if using complex variables becomes too difficult or time-consuming, we will instead focus on performance-based metrics such as qualifying position, constructor, and historical race outcomes. Additionally, to manage computation expenses, we will restrict the dataset to between 2010-2020. Lastly, if team responsibilities become uneven due to changing schedules or unexpected time constraints, we will redistribute tasks so that one member focuses more on visualizations or writing to ensure consistent and timely progress throughout the project, while the other focuses solely on code or fixing the code.