Using Machine learning to predict Formula One Podiums (2020 & 2021 seasons)

Mahmoud Abdelhadi
Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, Canada
melsafi@student.ubc.ca

Abstract

In this report, I will be discussing the results I got while using Multi-Layer Perception (MLP) to predict the winner of each race in the 2020 and 2021 seasons in Formula One. Some complications include the unpredictability of races due to crashes, weather, and car failures. I will also explain how I used the Weights and Biases (WandB) API to vary certain hyper parameters in my models to achieve the most optimal hyper parameters for testing data.

1 Introduction

To introduce the problem, my goal is to build and train a neural network to predict the race winner for each race during each season. with the winner being the driver that has the highest probability to win each race.

In the context of tuning the hyper-parameters for the neural network, I will be using Weights and Biases API to change the parameters of the neural network to try and find the one with the lowest loss.

In this paper, the methods I used, limitations, and my results will be discussed.

The remainder of this paper is structured as follows: Section 2 provides the methods used to obtain data, what I used to engineer the data, how I built my neural network and how I evaluated the success of the predictions. Section 3 presents the results of my model. Section 4 discusses the results and implications. Finally, Section 5 provides a summary of this paper.

2 Methods

IN this section, I will go through the methods I used to extract data from Ergast API. Wikipedia and the official Formula One website archive. The steps I took to make the data ready to be used as training data, and what machine learning model I picked to build my model.

2.1 Data extraction

To get the data needed for my data set, I used the Ergast API to collect data on the previous seasons, specifically from 2000 to 2021. for the missing data such as qualifying time, I used and HTML parser to get data from the official F1 website archives. Finally, for the weather, I used the wikipedia url for the races to get data on the weather on the day of the race.

I saved all the dataframes into a .csv files so I would not need to recollect the data every runtime.

2.2 Data engineering

To get the data ready to be used as training data for the neural network, first I joined all the data into one data frame.

I then calculated the age of the drivers instead of date of birth and dropped the date of birth column.

to get the qualifying times from minutes and seconds to just seconds (from X:XX.XXX to the form XX.XXX) so they can be compared better, we also calculated the cumulative difference in qualifying times so we know by how much faster the first car on the grid is compared to the other ones for each race.

Finally, I converted categorical variables into dummy/indicator variables to use as training and testing data.

I used data from the year 2000 to the year 2019 as my training data, and the year 2020 and 2021 as my testing data.

2.3 Building the networks

I build 2 approaches that reach the same goal, one that I made the model from scratch using Pytorch modules, and the second by training an MLPclassifier with my training data.

2.3.1 neural network classifier using Pytorch.nn modules

For this network, I used 4 linear layers, 1 dropout layer, Identity activation and softmax as my activation to get the probabilities of each driver winning and losing each race between zero and one.

I then used Weights and Biases API to tune my hyper parameters to this network and varied the hyper parameters according to table 1 below.

Table 1: Hyper-parameter tuning

Hyper-parameter	values
Optimizer	SGD & Adam
First layer size	65,70,75,80
Second layer size	15,20,25,30
Third layer size	40,50,60
Fourth layer size	5,8,10
Dropout probability	0.1 - 0.3
Learning rate	0 - 0.1 uniform
Batch size	1
loss criterion	Negative log-likelihood, Cross Entropy

With the hyper parameters, I used a random sweep and measured the training loss of the model, validation loss , and validation accuracy. my metric used to evaluate the models was cross varied using WandB API and the API's goal is to minimize the loss.

2.3.2 neural network classifier using MLPclassifier module

For this network, I used the MLPclassifier framework provided by Pytorch, I converted my data frame to a Numpy array and used the array to fit the model, I them used the fitted model to predict the 2020 and 2021 races.

Table 2: MLPclassifier Hyper-parameter tuning

parameter	values
Activation	Identity, Logistic, tanh, relu
First layer size	65,70,75,80
Second layer size	15,20,25,30
Third layer size	40,50,60
Fourth layer size	5,8,10
solver	lbfgs, SGD, Adam
alpha	0 - 0.1 uniform

2.4 Models Evaluation

The way each model was evaluated was by selecting each driver in each circuit in the season, calculating the probability of each driver winning in that circuit, and then organizing the drivers in descending order of probabilities, where the podium (winner) is the driver that has the highest probability of winning that race, and second, third, forth etc.. is what follows that winning driver in order of probability.

3 Results

3.0.1 neural network classifier using Pytorch.nn modules

To summarize the results, the best model trained using the nn.sequential() module with 4 linear layers and nn.Identity() as activation and sigmoid() at the end to get the output as probability between 0 and 1. The model hyper-parameter variations are shown in the figure below

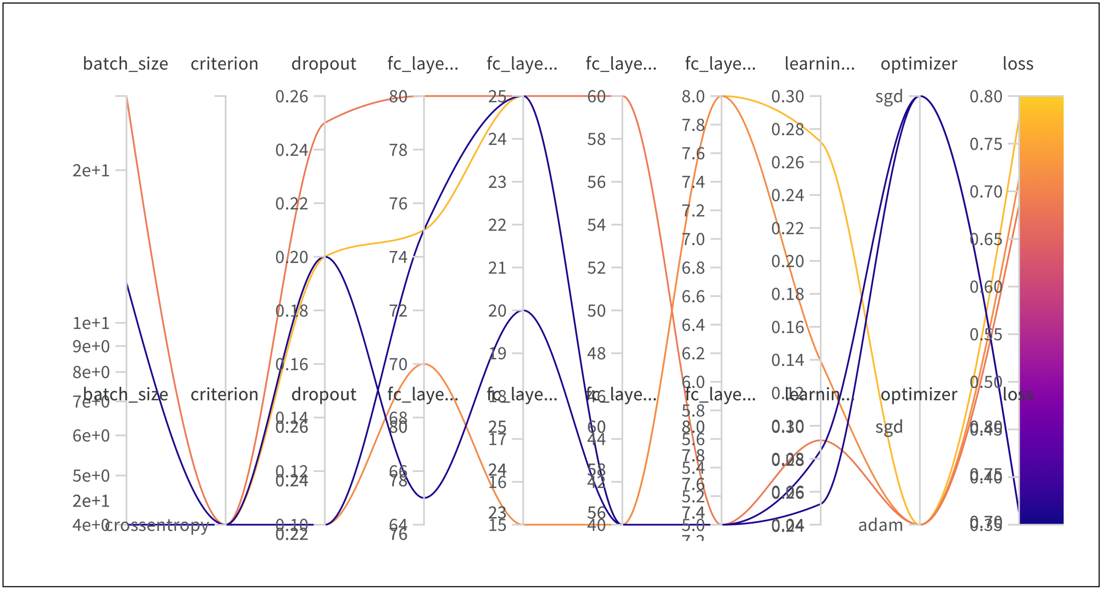

Figure 1: nn.Pytorch modules Sweep summary with different hyper-parameters

My best model yielded an accuracy of 58%, which means that the model predicted 13 out of the 22 races in the 2020 season.

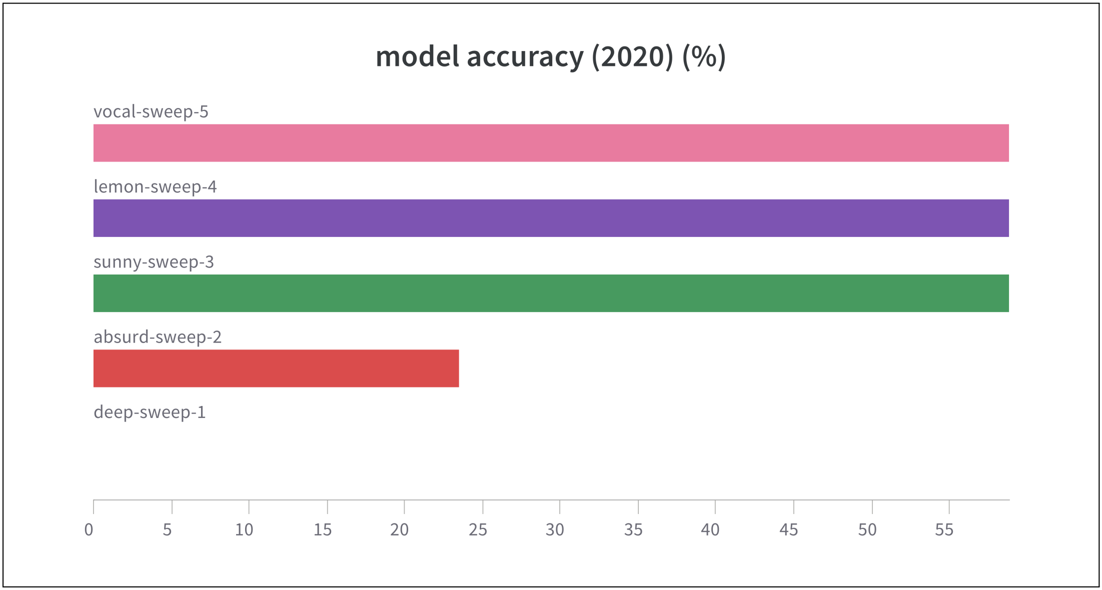

Figure 2: Results for 2020 Season

Alternatively, for the 2021 season, I got an accuracy of 52%, meaning the model predicted the podium winner for 9 out of the 17 races in the 2021 season.

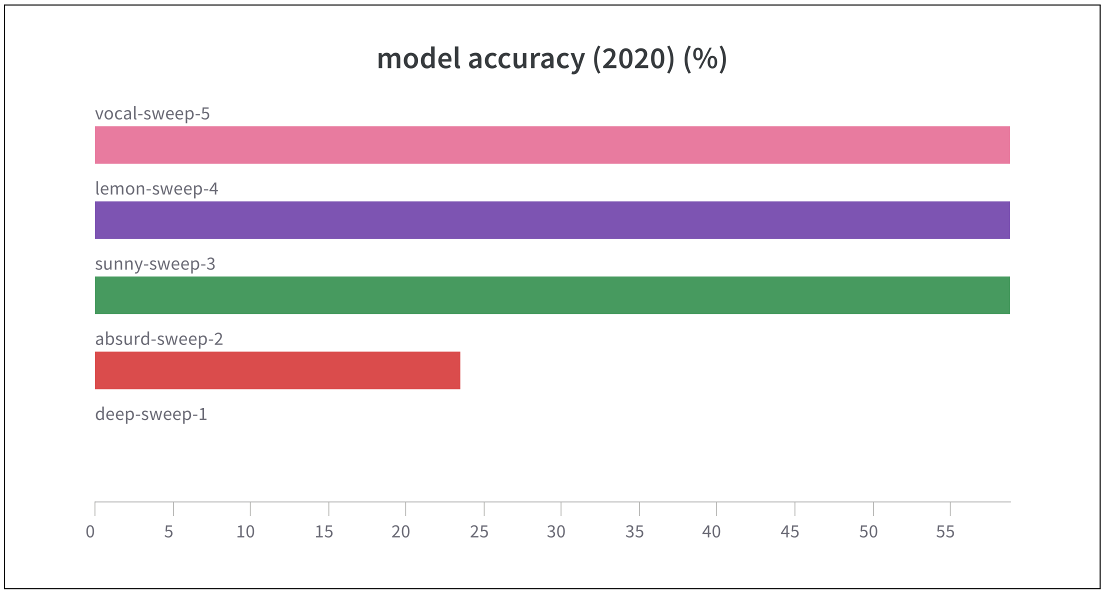

Figure 3: Results for 2021 Season

The hyper-parameters that yielded these results are summarized in the table below.

Table 3: nn.Pytorch Hyper-parameter tuning

parameter	values
Batch Size	28
Criterion	Cross Entropy
Dropout rate	0.25
First layer size 80	
Second layer size	25
Third layer size	60
Fourth layer size	5
Optimizer	Adam
learning rate	0.09116830834680612

3.0.2 neural network classifier using MLPclassifier module

Using the MLPclassifier, I sweeped the model with WandB API with the following parameters which yielded the losses shown in figure 2 below.

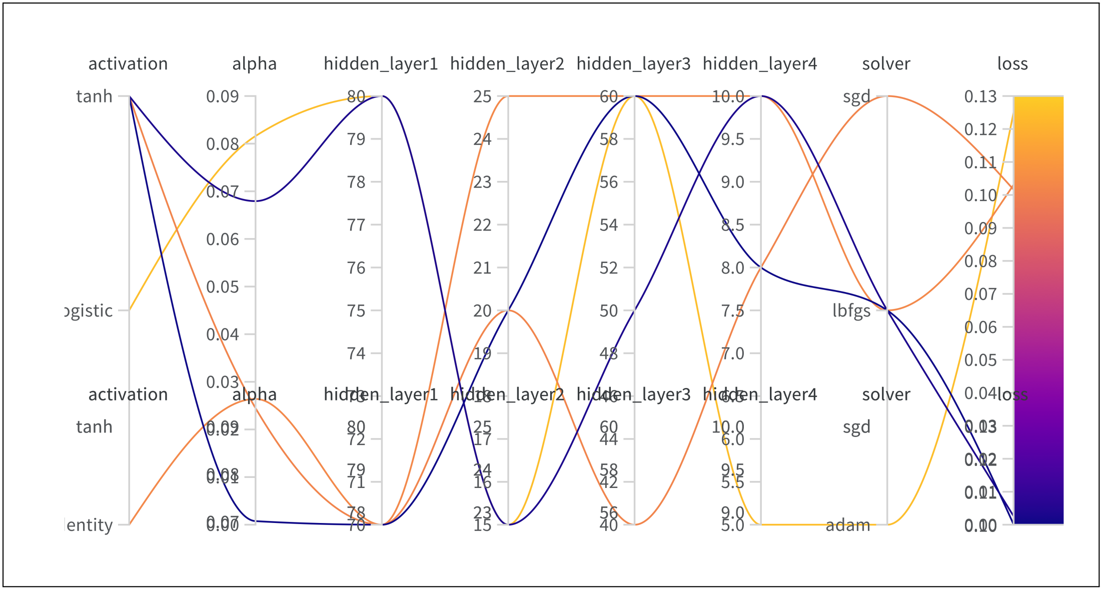

Figure 4: MLPclassifier Sweep summary with different hyper-parameters

My best model yielded an accuracy of 64.7%, which means that the model predicted 14 out of the 22 races in the 2020 season.

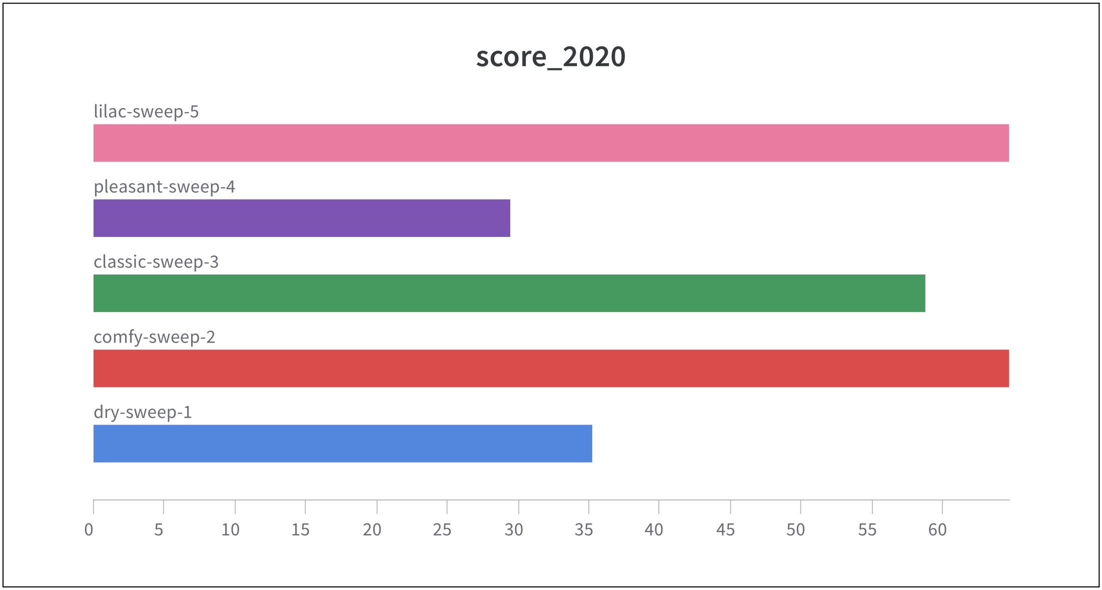

Figure 5: Results for 2020 Season

Alternatively, for the 2021 season, I got an accuracy of 63.6%, meaning the model predicted the podium winner for 11 out of the 17 races in the 2021 season.

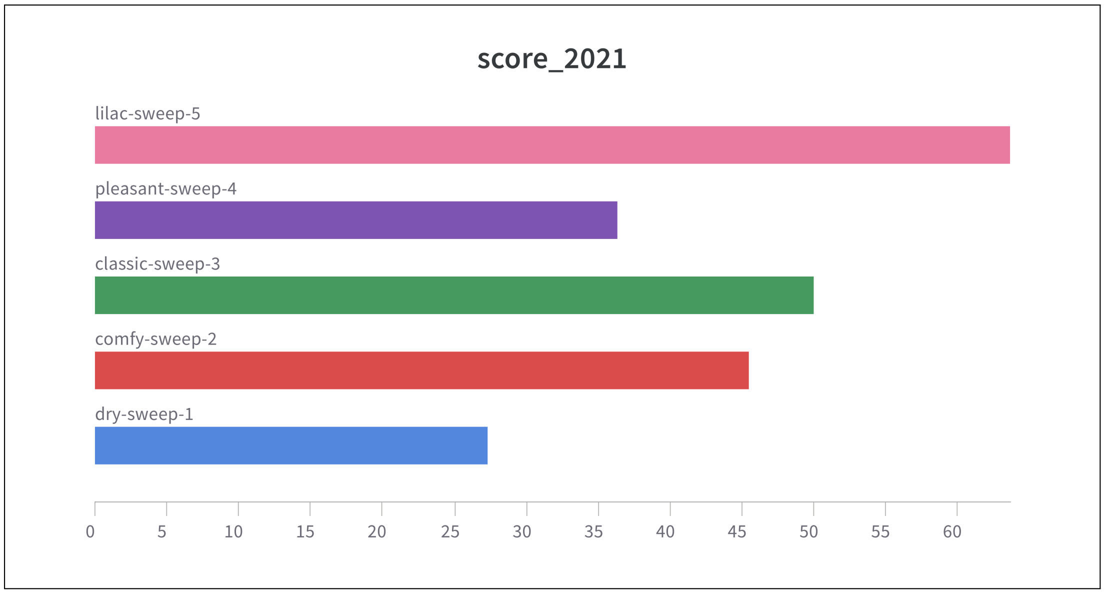

Figure 6: Results for 2021 Season

The hyper-parameters that yielded these results are summarized in the table below.

Table 4: MLPclassifier Hyper-parameter tuning

parameter	values
Activation	Logistic
First layer size 80	
Second layer size	15
Third layer size	60
Fourth layer size	5
solver	Adam
alpha	0.08160642765958119

4 Discussions

Overall, my models showed success in either one of the 2020 season or the 2021 season, however, due to the limitations and unpredictability of motor racing, crashes, safety cars and regulation changes, I was unable to get a model that was more that 60% accurate for both seasons.

Furthermore, the regulations changed between 2020 and 2021, more specifically, a cost cap was introduced between the 2020 and 2021 season. Cars for each constructor agreed that the budget for the development & manufacturing of cars for 2021 will be lowered from the original figure of $175m per year to $145m, with further reductions in the following years.

The FIA has also approved a measure to limit downforce on the 2021 cars. The new rule means teams will have to trim part of the floor to reduce the downforce created, largely so a new tyre compound will not be needed for next season, before a switch to 18 inch tyres in 2022.

Lastly, From the beginning of the 2021 season, there will be a limitation on the number of upgrades a power unit manufacturer can do over the course of a season and the minimum weight increases 3kg to 749kg.

These new regulations evened the playing field for constructors that are at the bottom of the grid, hence why I was unable to get a model that was more than 60% accurate in predicting the podium for each race in the season for 2020 and 2021. FIA simply introduced a new playing field for motor racing and made it easier for bottom-tier teams to compete.

Finally, if you, the reader, do watch Formula One, you would be aware of the unpredictability of the 2021 season. Especially the absurd violations that race control awarded to some drivers on the grid. The race director, Michael Massi, was actually asked to step down over the unfair advantages he awarded some constructors on the grid.

All the above points introduced limitations into my machine learning models, where the model is trained for races with very similar regulations, and then tested on a season where regulations change, causing the accuracy to drop around 14%.

5 Conclusion

In this paper, we addressed the steps I took to extract data, manipulate the data, and train 2 machine learning models to predict the accuracy. My best model hd around 64% accuracy for both 2020 and 2021 seasons. We also discussed limitations in using machine learning for motor racing predictions, especially with the evolving nature of regulations and unpredictability of cars travelling at 300 km/h.