F1 Safety car predictor

1. Abstract

This is an introductory project that features Formula 1 and Machine Learning. Formula 1 represents the pinnacle of auto-moto sport and as today it is the most popular racing championship in the world.

Goal of this project is to use machine learning algorithm and a given set of data in order to predict whether a safety car will be deployed during a specific race in calendar. Safety car occurs in various situations in Formula 1, and it means that one car (*not on the grid*) will go in front of the race leader and slow down the entire column, until the reason for deploying it is resolved. During the safety car period, overtaking is not allowed, but pit stops are more profitable as everyone is slower, so this can play a crucial role in fighting for valuable points.

The usual reason for a safety car is an **incident**, whether it is one car crashing in the wall or multiple cars colliding together. Rain can also bring it out for some time, however, if the rain is too heavy, the race will be stopped.

In order for this project to make more sense, it is decided that we will not be satisfied with the simple Boolean decision like, **yes**, there will be safety care, or **no** it will not, but rather, we want a percentage of that happening. So for example, in a circuit like Monaco, which is a street race, chance for a safety car is higher, like 70%, while on a traditional track like Austria, that percentage can go down to 35%. Of course, these numbers are just an assumptions at this point.

2. What will you need?

Like mentioned, there are various parameters that can increase the chance of a safety car deployment, and most of them are *static* and interchangeable over the years, meaning that there are very little dependencies between these variables. Since safety car was introduced to F1 in 1993, our dataset will consist of data from 1993 all the way to the present moment. Also you will encounter a column that holds information about *virtual safety car*. This is similar to an actual safety car, but for minor accidents. At this point, an actual car is not on a track, but a simple warning with yellow flags. Virtual safety car was introduced in 2015 and it will also be fed to a model for a better prediction. Accordingly, cell values for this column before 2015 will be assigned Nan.

In this repo, you will find 2 separate datasets, the main one named *safety_car_predictor.csv* and a so called dimension table named *circuit_data.csv*. You might find some information missing from the original one, but everything necessary that is not here is added in a Jupyter Notebook file called *F1-Safety-Car-Predictor.ipynb*. All information about that file is listed in it. In order to open it, you might want to use a tool like VS Code, and in its terminal run command *jupyter notebook.*

As mentioned, we use 2 tables for data storage. First parameters are stored in *circuit-_data.csv* and they are:

- **Circuit name** – Firstly, we have to know which track the race is held on due to various track parameters. Some corners are more difficult, some more tricky to handle and some just require experience from a driver. Also, pit lane exit can be important. If you go directly into a tricky part of the track, like on Monza, various accidents can occur (*Remember Hamilton and Verstappen in 2021).*
- **Circuit ID –** Number that is assigned to each circuit, so that ML model can use it as prediction. We do not use *string* for prediction.
- **Circuit type** – Track configuration also plays a role. Street circuits are more narrow and difficult to drive, creating a more risky situation to overtake. Traditional circuits have more space and therefore are easier for less experienced players not to crash their cars.
- **Physical difficulty of a circuit, for both driver and car** – This is an experimental parameter, and it indicates whether a circuit is particularly difficult for drivers due to extreme weather conditions like high temperature and humidity. Also, cars don't behave the same way on every track due to same parameters mentioned previously. Please understand that this parameter is more or less subjective when it comes to not so obvious races. There are 4 levels of difficulty, where 1 is the easiest. For example, most tracks that are held in high humidity and high

temperatures areas like Singapore and Malaysia are given a high level. Others are subjective so feel free to experiment for yourself.

Other parameters are stored in *safety_car_predictor.csv* and they are:

- **Year** and **date** are both just informational parameters. They do not affect our predictions.
- **Name** is important because, with date, it creates a unique entry for a single race in a history. In the code, these two tables are merged together on this column.
- **Weather information** – Weather is fairly important for this project since rain is the one of the biggest reasons for incidents and most likely, safety car deployment. Here, since this is an introductory project and we do not possess detailed weather information (*we are broke students*), we use 3 levels for weather representation. If there is not rain we say *dry,* if the rain is in normal doses (*usually this means green, intermediate tires*) we say *light rain.* In case of intensified rain, when blue, wet tires are used, we say *heavy rain*. Again, this parameter is more or less subjective with limited knowledge of internet and our memory, so if you know better, do let us know!
- **Number of DNFs** – DNF stands for "*Did not finish*" and simply means that the car did not make it to the end of the race. Reason could be an incident, but also a mechanical issue. We will use this information as well, since a lot of mechanical issues are connected to a geographical location of a track. It is important to address that in the actual prediction sample we can't input a number of DNFs for future race, as it didn't happen. Here we calculate an average number of DNFs per track and use that for prediction. Same logic is used for number of safety car and virtual safety car deployments.
- **Number of safety car and virtual safety car deployments** obviously matters a lot. It is not same if safety car happens once and if it happens 3 times. That just mean that circuit with 3 safety car occurrences is more difficult and prone to accidents.
- **Total** column represents the sum of previous two columns, and is mostly experimental parameters. It will also have value in data visualization later on.
- Finally, column **safety car** has Boolean value and it gives information whether a safety car was deployed or not on a race. This is our prediction target.

For this project, we will use Python and Jupyter Notebook to make it more readable. We will discuss the choice of a machine learning algorithm in the next chapter.

3. Choosing a ML algorithm

As we are still beginners in this area, initially, our choices were either Decision tree or Random Forest, probably the 2 most algorithms. When we started to dig up a little bit, we realized that Random Forest is a better choice. Some advantages over Decision Tree are given in a table 1.

Random Forest is an ensemble learning method that combines multiple decision trees to improve accuracy and reduce overfitting. It is widely used for both **classification and regression** tasks.

The key idea behind Random Forest is **bagging** (Bootstrap Aggregation), where multiple decision trees are trained on different random subsets of the dataset. This prevents overfitting. The final prediction is obtained by **averaging the outputs** (for regression) or taking a **majority vote** (for classification).

For training we used mentioned columns with some new ones that are created in the notebook file. Initial idea was to group all entries of an input track from the original table, so for example, if we want to predict Australian Grand Prix 2025, we would create a mini table with all Australian Grand Prix races in F1 history. On that set, we would train our model.

However, we quickly dropped that idea, and decided to train it on the **entire** table. Explanation of training and data manipulation is provided in Jupyter Notebook file.

Table 1 : Decision Tree versus Random Forest

Feature	Decision Tree	Random Forest
Overfitting	Highly prone to overfitting on training data.	Reduces overfitting by averaging multiple trees.
Bias-Variance Tradeoff	High variance – A small change in data can drastically change the tree.	Reduces variance while maintaining accuracy.
Robustness	Sensitive to noise in the dataset.	More robust as it aggregates multiple models.
Feature Importance	Can provide feature importance, but results are unstable.	Provides more reliable feature importance scores.
Performance on large datasets	Slower for large datasets due to depth and complexity.	Handles large datasets efficiently due to parallelization.

4. Dataset information to consider

Formula 1 calendar is changing from time to time, and some tracks are being removed from the rotation, while some are being added. In this chapter you can find some *odd* stuff when it comes to dataset.

1. **San Marino Grand Prix = Emilia Romagna Grand Prix.** San Marino Grand Prix was a track used in past until 2006. After that it was dropped from the calendar, only to return in 2020 as Emilia Romagna Grand Prix. Since this is the same track, we named all occurrences of it Emilia Romagna so the model can use that information as well.

2. **European Grand Prix.** We are aware that European Grand Prix change multiple tracks during its history, but since it is not used in the F1 calendar, and since there are no hints of it comeback, we decided to just let it slip, and named them all European Grand Prix. In case something happens, and it returns, or if any new grand prix starts to use tracks of past European Grand Prix, it will be divided and resolved.

3. **United States Grand Prix = Indianapolis Grand Prix.** Similar as point 1. United States Grand Prix used Indianapolis track until 2012 for Sunday race and COTA in Texas from 2012. Since it is a *different* track, model should not consider it as a same one (*since the name is same*), so we decided to rename it. We realize, of course, that there is no such thing as *Indianapolis Grand Prix.*

4. **Luxembourg Grand Prix.** Since this grand prix uses Nürburgring track that was rotated in European Grand Prix, all appearances of Luxembourg Grand Prix are replaces with European Grand Prix. If something changes, dataset will be modified.

5. **Adelaide Grand Prix = Australian Grand Prix.** Same logic as point 3. Melbourne became host of Australian Grand Prix, race didn't change its name, but it changed the circuit. In order to distinguish them, we gave different names.

We realize that some *abstractions* that we made during this project might twist around the prediction and model accuracy. Please understand that this is still a beginner project, with no intention of shutting down betting houses or something. We are just two curious F1 fans who happen to be interested in Machine Learning as well.