Predicting Formula 1 Race Performance from Free Practice Data Using Machine Learning

Simen S. Gåsland
University of Stavanger

Introduction

This project investigates how well Free Practice (FP) data can predict Formula 1 race outcomes. Using FastF1, I gathered FP statistics, weather information, and race results, and tested three models: an SVC for point scoring, a Beta-Binomial model for teammate comparisons, and a regularized linear regression for the fastest lap prediction. The goal was to determine how much information FP sessions actually reveal about race-day performance.

Data Collection, Cleaning and Feature Engineering

All the data in this project were collected using the FastF1 Python package which fetches historic Formula 1 data via the **jolpica-f1** API [1]. A custom **DataAquisition** class was implemented to automate data retrieval, cleaning and feature generation. The dataset covers the 2024, 2023, 2022, 2021 and 2019 seasons.

For all selected seasons, the script produced a .csv file containing one row per driver per Grand Prix (GP). Instead of storing every FP lap aggregated statistics were calculated, including the mean, standard deviation and fastest FP lap times, as well as the delta to the session's best lap.

A "faster-than-teammate" feature was added by comparing teammates fastest FP lap time, and average weather variables (track temperature, air temperature, rainfall) were included for both FP and Race sessions. The resulting dataset combines driver performance, weather conditions, and race outcomes.

Key features:

- **Driver performance:** FastestFPLap, MeanFPLaps, StdFPLaps, DeltaBestFPLap
- **Weather:** TrackTempAvgFP, AirTempAvgFP, RainAvgFP
- **Race outcomes:** FastestLapRace, FasterThanTeammateRace, PointFinishRace

Predicting Point-Scoring Drivers Using an SVC

A Support Vector Classifier (SVC) was trained to predict whether a driver would finish in the top ten based solely on FP data. The features included the fastest FP lap, mean and standard deviation of lap times, delta from best FP lap, adn whether the driver was faster than their teammate. Team and driver information were intentionally excluded, since includuing it the model would quickly learn that teams like Ferrari and McLaren almost always score points. This defeats the goal to test whether FP performance alone could predict race results.

To implement the SVC, I used scikit-learn. Two main classes are available: **sklearn.SVC** and **sklearn.LinearSVC** [2]. Since I wanted to test different kernel functions I used sklearn.SVC. Unlike LinearSVC, I cannot change the loss function or penalty type. These are fixed as hinge loss and L2 regularization. However, the amount of regularization can be tuned using the misclassification penalty C [2].

$$C\sum_{i=1}^{n}\mathcal{L}(f(x_i),y_i)+\tfrac{1}{2}\|w\|^2 \tag{1}$$

The model was evaluated using different kernels and misclassification penalties, and accuracy was used as the comparison metric since it provides an overall indicator of performance. The best result was obtained with the linear kernel and $C = 0.1$.

Class	Precision	Recall	F1-score
No Points	0.72	0.74	0.73
Scored Points	0.72	0.70	0.71

TABLE I: Classification report for SVC model

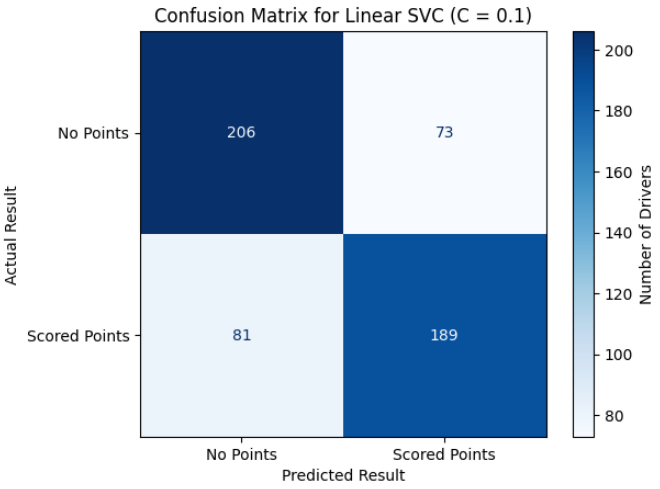

Fig. 1: Confusion matrix

This result suggests that the relationship between FP features and race outcomes is approximatly linear, and that a smaller

C prevented overfitting, and improved performance on unseen race data. In other words, a simple linear decision boundary using FP metrics is sufficient to separate drivers likely to score points from those who are not.

The confusion matrix in figure 1 shows that the model performs similarly for both classes, but is slightly better at identifying drivers who will not score points than those who will.

Modeling Teammate Performance Using a Beta-Binomial Model

A Beta-Binomial model was used to estimate the probability that a driver finishes ahead of their teammate in the race, given whether they were faster or slower in FP. As a prior belief, I assumed no knowledge about the relationship between FP and race performance.

$$\textbf{Prior:} \quad \theta \sim Beta(1, 1) \tag{2}$$

From the dataset, I used the Faster-than-Teammate metric for both FP and race sessions to compute the posterior distributions.

$$\textbf{Posterior:} \quad \theta \mid \text{Slower in FP} \sim Beta(426, 674) \tag{3}$$

$$\textbf{Posterior:} \quad \theta \mid \text{Faster in FP} \sim Beta(674, 426) \tag{4}$$

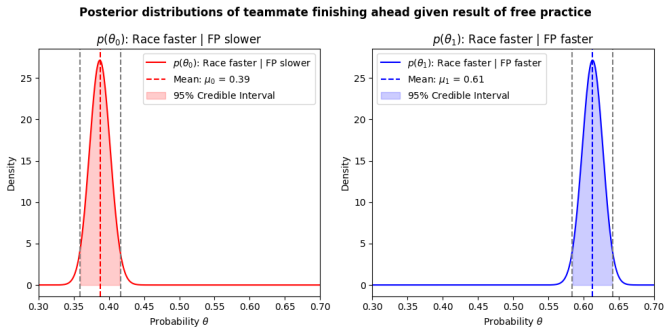

Fig. 2: Posterior distributions

Figure 2 shows the resulting posterior distributions. When a driver was faster than their teammate in FP, the posterior mean probability of also being faster in the race was about 0.61, compared to 0.39 when slower in FP. This indicates that drivers who outperform their teammates in FP are statistically more likely to do so in the race as well.

Predicting the Fastest Race Lap Using Linear Regression

The objective was to predict the fastest race lap using FP data, so I began by evaluating linear regression as a simple yet effective modelling approach. To improve performance, I compared several regularized linear models across different polynomial degrees. As shown in table II, the ElasticNet model with first-degree features produced the strongest cross-validation results. I selected ElasticNet with an l_1-ratio of 0.7 as the final model, as this places more weight on the Lasso penalty. Lasso helps remove weak or noisy coefficients, while the Ridge component keeps the model stable. This combination provided a good balance of accuracy and consistency. Although the second-order model performed well on unseen 2025 race data, its instability across runs, made it unsuitable as the final choice.

Degree	Model	Best α	MAE $\pm$ std	R^2 $\pm$ std
1	LinearRegression	–	1.528 $\pm$ 0.481	0.895 $\pm$ 0.098
	Lasso	0.142	1.561 $\pm$ 0.638	0.908 $\pm$ 0.099
	Ridge	25.950	1.528 $\pm$ 0.281	0.909 $\pm$ 0.069
	ElasticNet	0.013	1.419 $\pm$ 0.452	0.819 $\pm$ 0.153
	ElasticNet (l_1=0.7)	**0.022**	**1.488 $\pm$ 0.310**	**0.922 $\pm$ 0.055**
2	LinearRegression	–	1.917 $\pm$ 0.677	0.706 $\pm$ 0.407
	Lasso	0.027	1.814 $\pm$ 1.058	0.735 $\pm$ 0.335
	Ridge	79.248	1.841 $\pm$ 1.147	0.688 $\pm$ 0.441
	ElasticNet	0.081	1.586 $\pm$ 0.842	0.662 $\pm$ 0.553
	ElasticNet (l_1=0.7)	**0.056**	**1.759 $\pm$ 0.739**	**0.786 $\pm$ 0.279**
3	LinearRegression	–	3.041 $\pm$ 1.689	-1.116 $\pm$ 3.749
	Lasso	0.171	1.652 $\pm$ 0.465	-0.974 $\pm$ 3.750
	Ridge	14.850	2.888 $\pm$ 2.231	-4.138 $\pm$ 10.039
	ElasticNet	0.039	3.336 $\pm$ 2.947	-1.987 $\pm$ 5.764
	ElasticNet (l_1=0.7)	**0.171**	**1.801 $\pm$ 0.358**	**0.853 $\pm$ 0.140**

TABLE II: Model comparison across polynomial degrees.

To assess real-world performance, I tested the final model on the most recent GP in São Paulo, which occurred after the training data were retrieved from the FastF1 API and was therefore fully unseen. I also evaluated additional races from the 2025 season (table III). These races occurred before the API call but were **not** included in the training data or any previous testing, so they still provide a strong indication of how well the model generalizes to a new race weekend.

Grand Prix	Predicted (s)	Actual (s)	Error (s)	Note
Australian	78.687	82.167	3.480	Worst
São Paulo	71.534	72.400	0.866	Last GP
Azerbaijan	103.045	103.388	0.343	Best
Monaco	73.486	73.221	0.265	Best

TABLE III: Predictions for selected 2025 Grand Prix races.

Conclusion

The results show that FP data provides clear signals for predicting race outcomes. The SVC identified point-scoring drivers reliably, and the Bayesian model reveled a strong link between FP pace and teammate race results. ElasticNet regression also gave stable fastest lap predictions, even on unseen 2025 races. Overall, despite many race-day factors like weather, temperature, and tyres, simple models with good FP features still generalize well and capture meaningful patterns.

References

[1] theOehrly, *Fastf1: A python package for accessing and analysing formula 1 results, schedules, timing data and telemetry*, https://github.com/theOehrly/Fast-F1, GitHub repository, accessed: 12 November 2025, Sep. 2025.

[2] scikit-learn Developers, *Sklearn.svm.svc — c-support vector classification*, https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, Accessed: 12 November 2025, Nov. 2025.