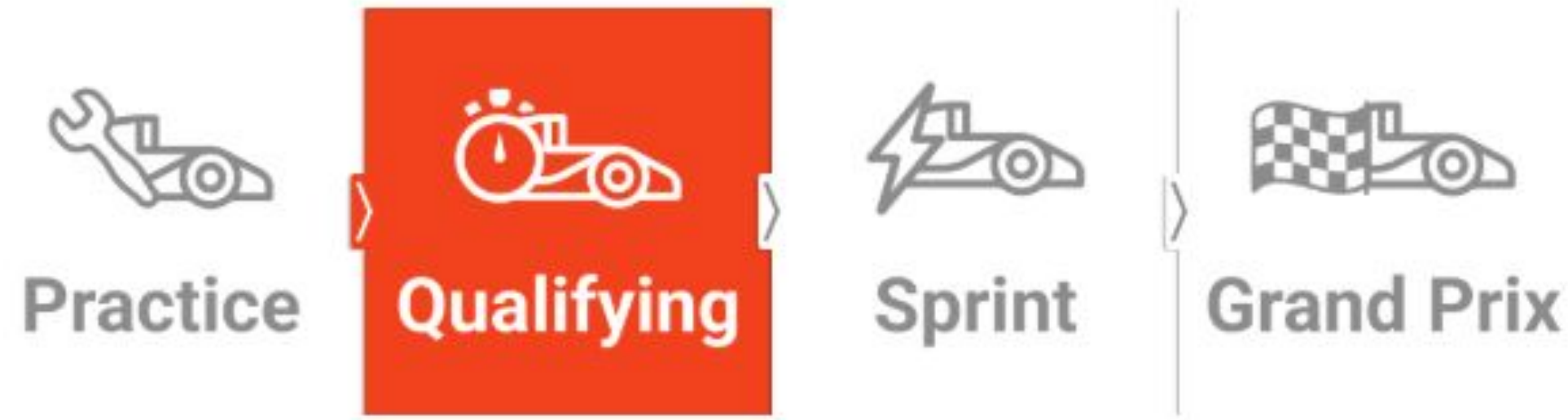
Beyond the Pit Wall: Machine Learning in Formula 1

Phillip Vieth Konrad Olsen Sagar Clemensen Aske Roelshøj

Goals

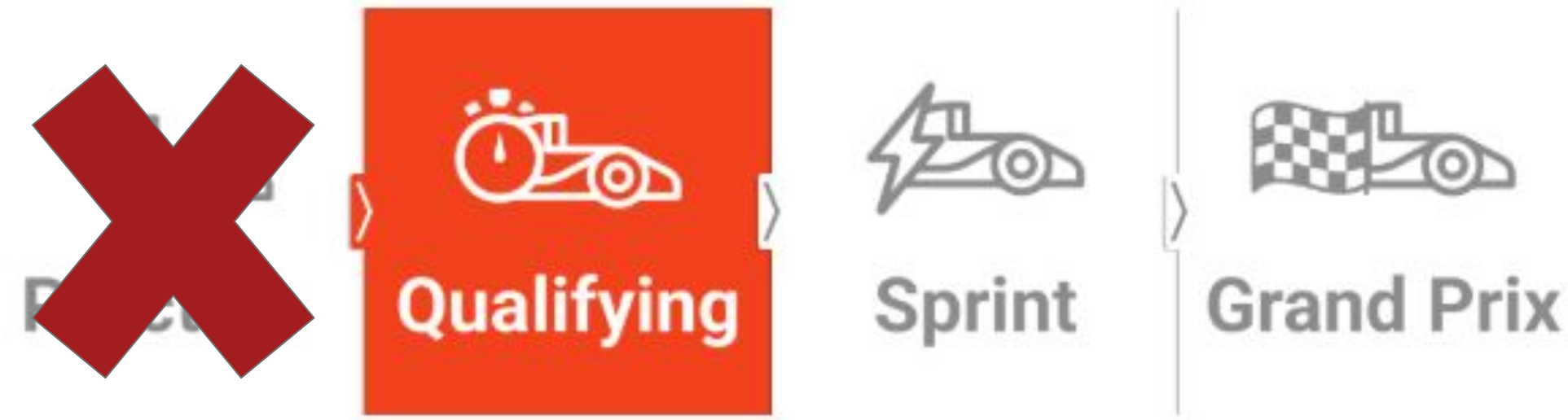
- Predict tyre compound with classification
- Clustering, differences in driving?
- Predict lap duration with regression

Types of race



From fanamp.com

Types of race



From fanamp.com

The Data

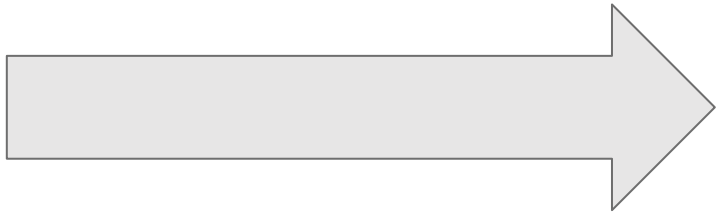
- Data downloaded from OPEN F1
- Data between 2023-2025
- Data sampled real-time / per lap?
- 33 features in total

The Data

- Grand Prix~400k-600k rows/session (9 features)
- Sprint ~ 100k-160k rows/session (9 features)
- Qualifying ~ 25k-90k rows/session (9 features)
- Smallest features ~ 20 rows/laps (~40-80 laps in Grand Prix, ~15-20 in sprint)
- Season volume: 22 Grand Prix & Qualifying, 12 sprint
- ~50 GBs in total

Data in details

Open F1 documentation



Click here

Data we used

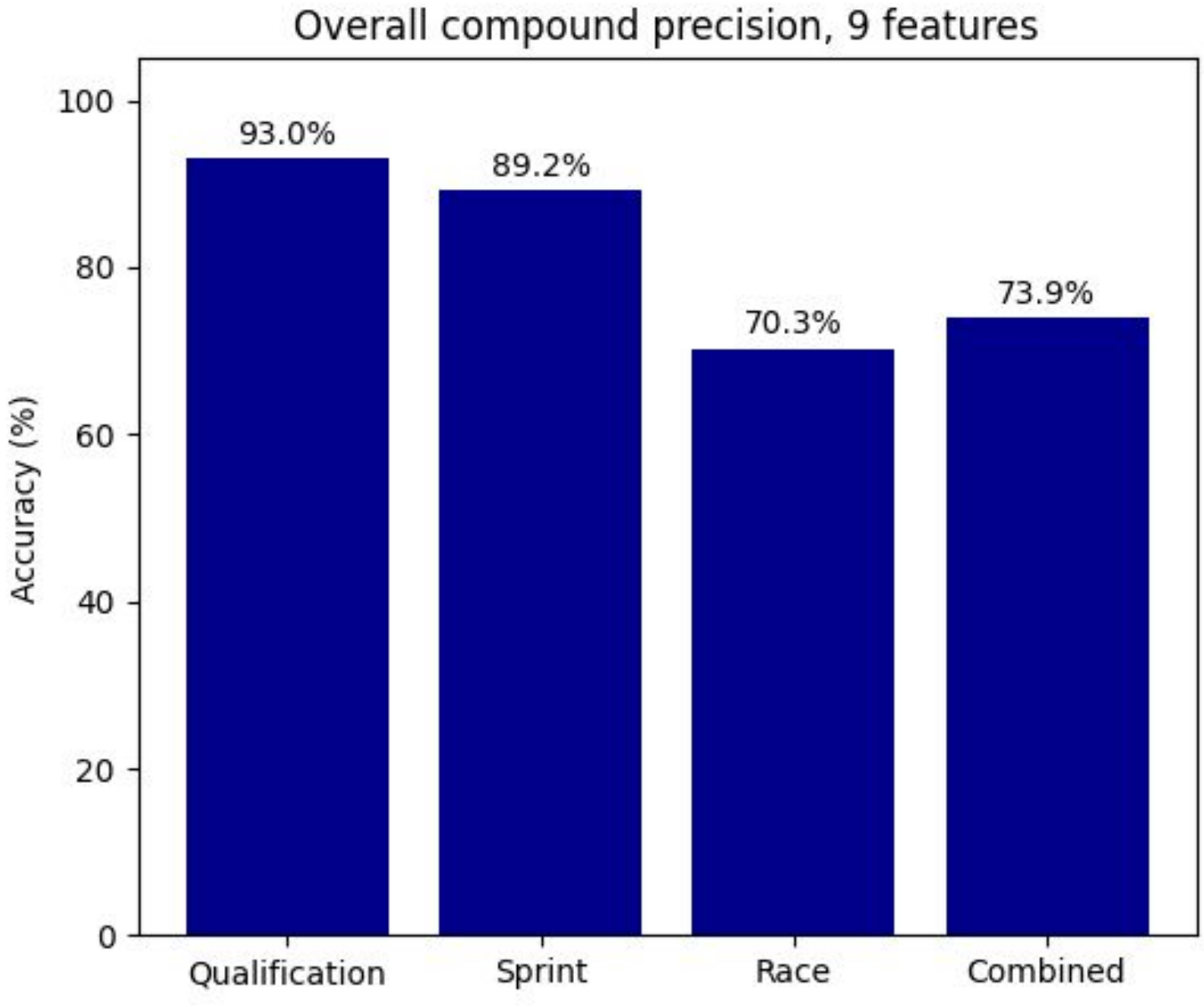
Click here

Classification of compounds

- Five types of compounds
- Different uses and drawbacks

Classification of compounds - Model 1

- 9 features
- Roughly 100k data points
- Simple decision tree & NN
- Unable to generalize

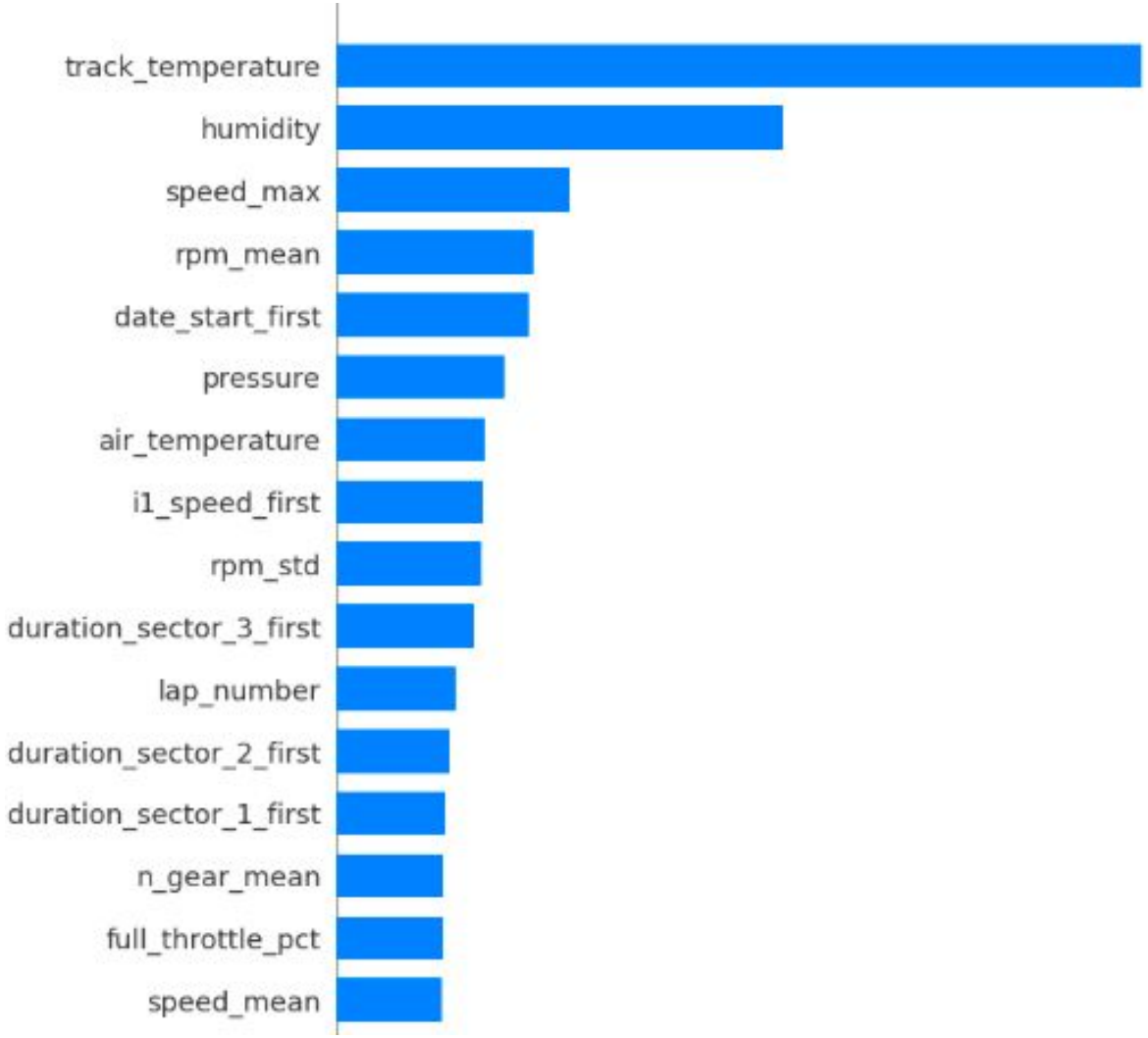


Classification of compounds - Model 2

- Only Races
- XGBoost
- Different Weather/Car telemetry
- 80% correct classification

Some variables

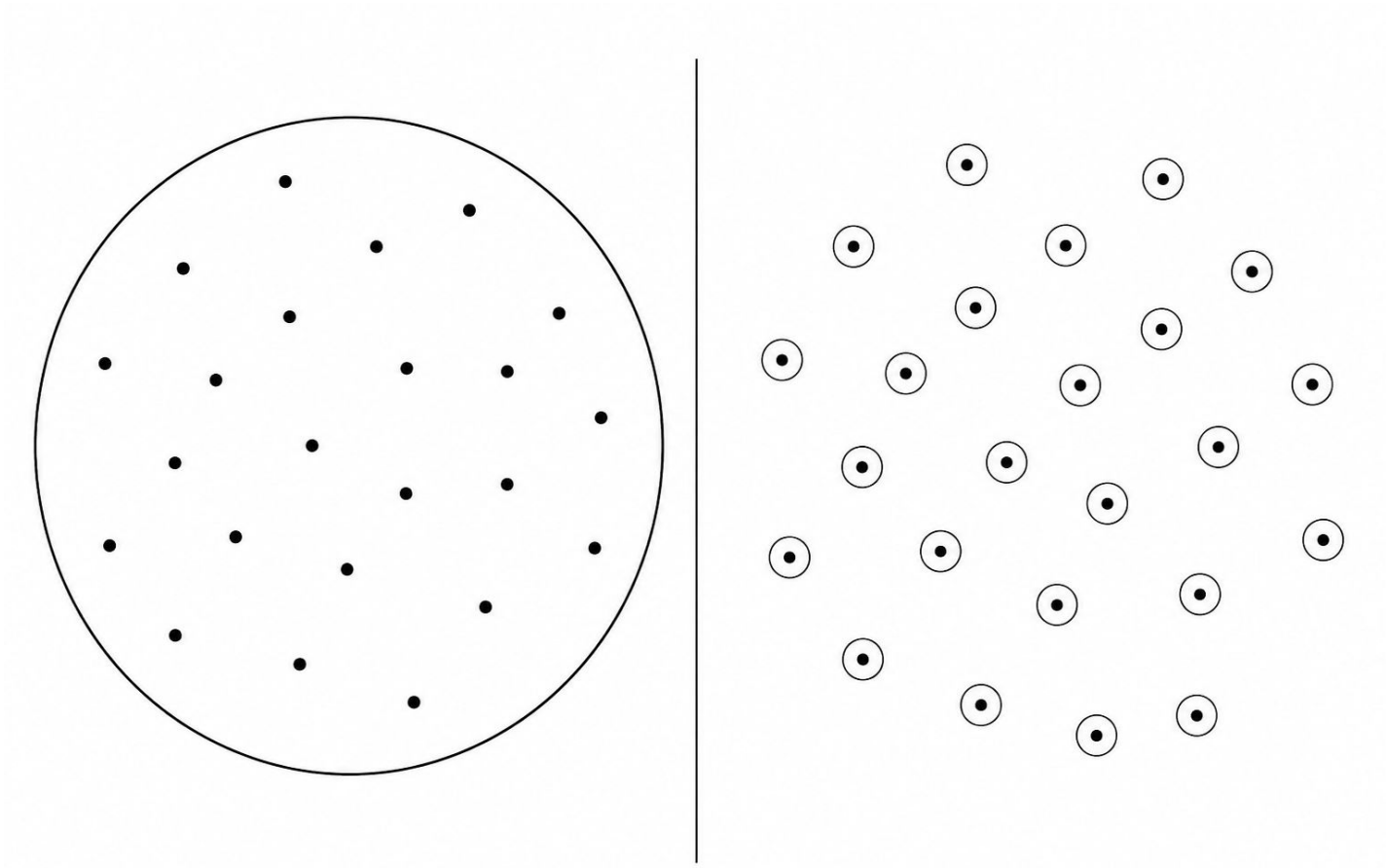
- Lap number
- Speed
- Throttle
- Brake
- Rpm
- n_gear
- Drs
- Date
- Duration_sector_n_first
- Air temperature
- Humidity
- Rainfall
- Track temperature

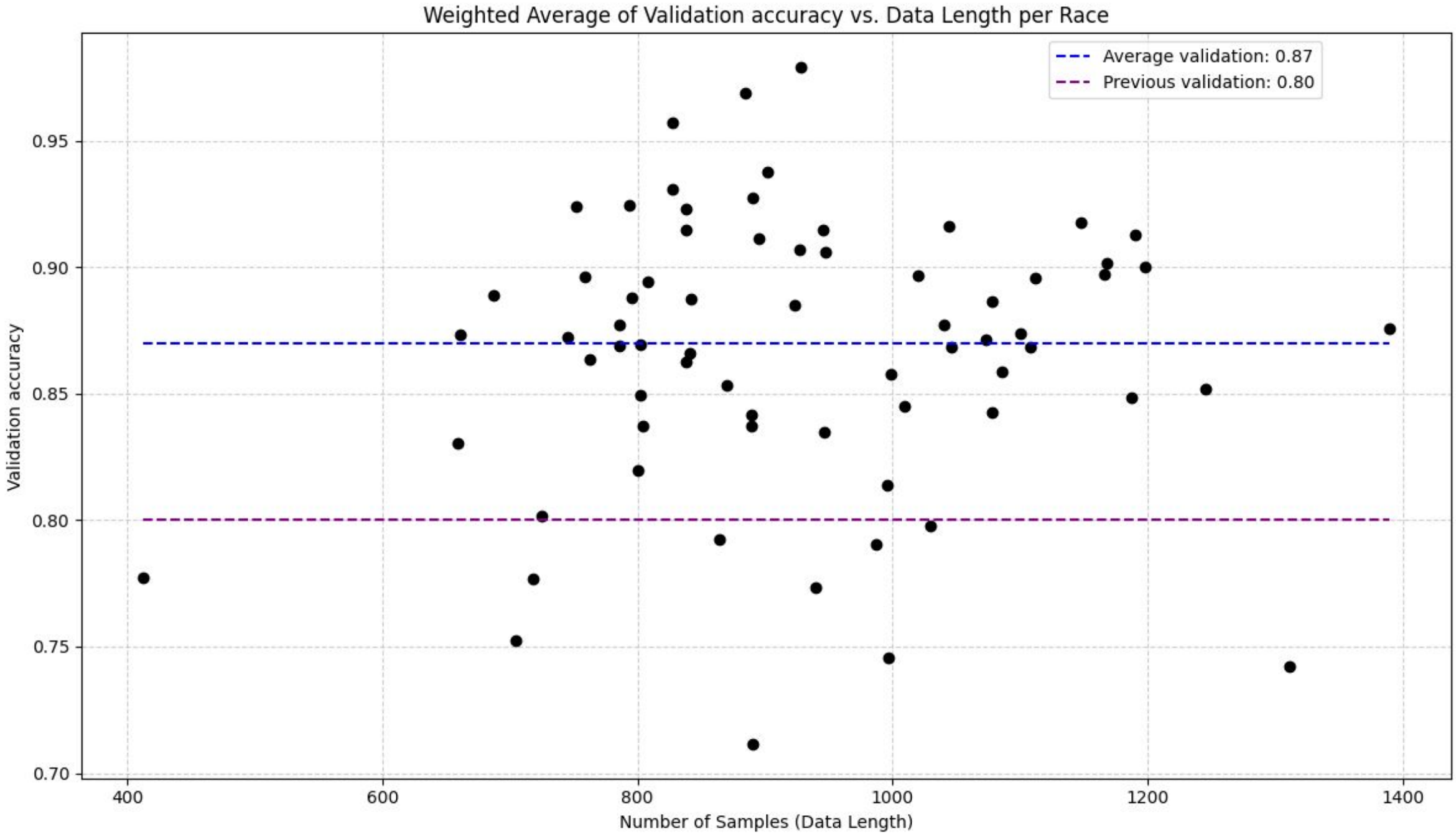


Training on all races combined vs a single race

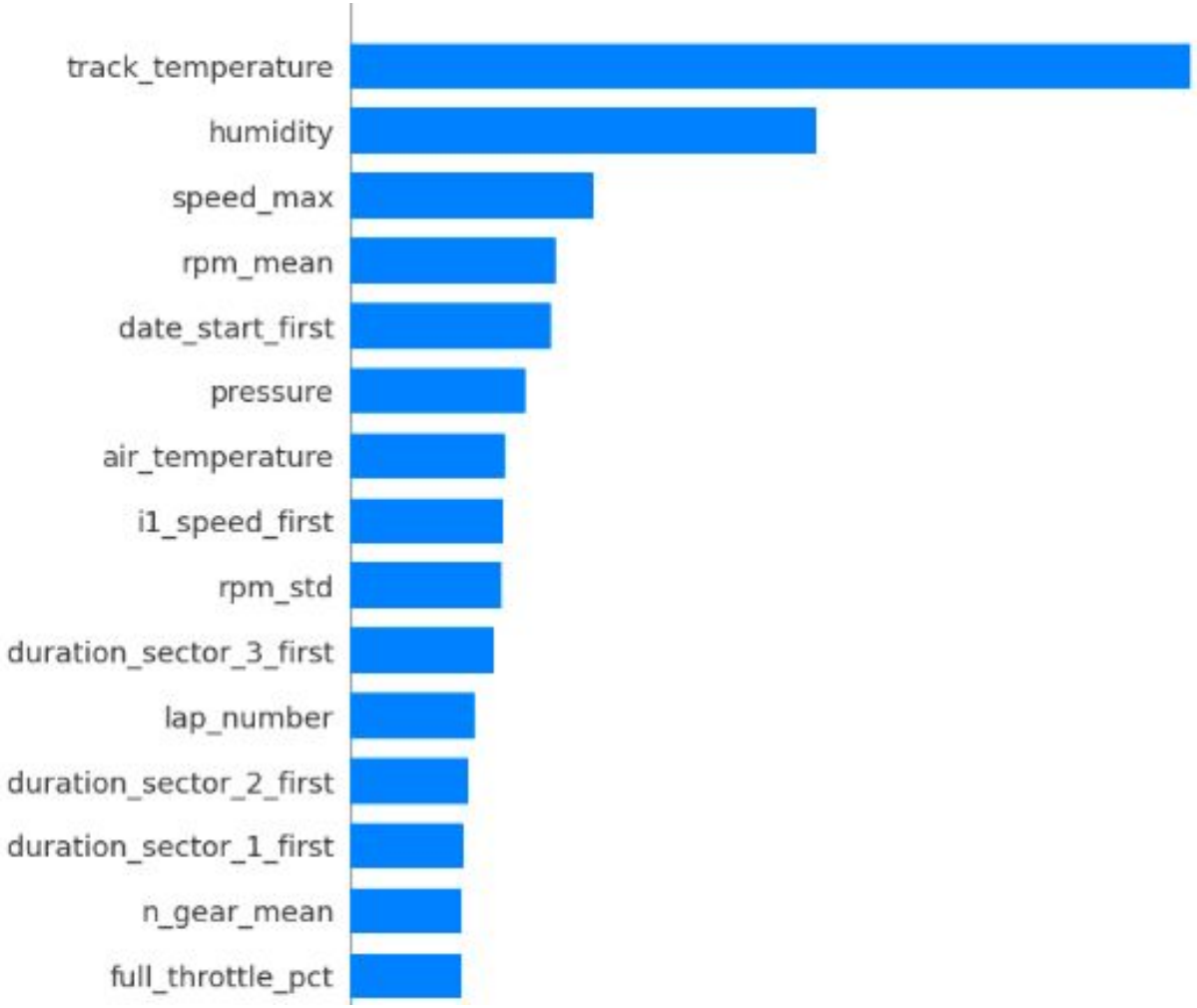
Pros: more data

Cons: lower quality

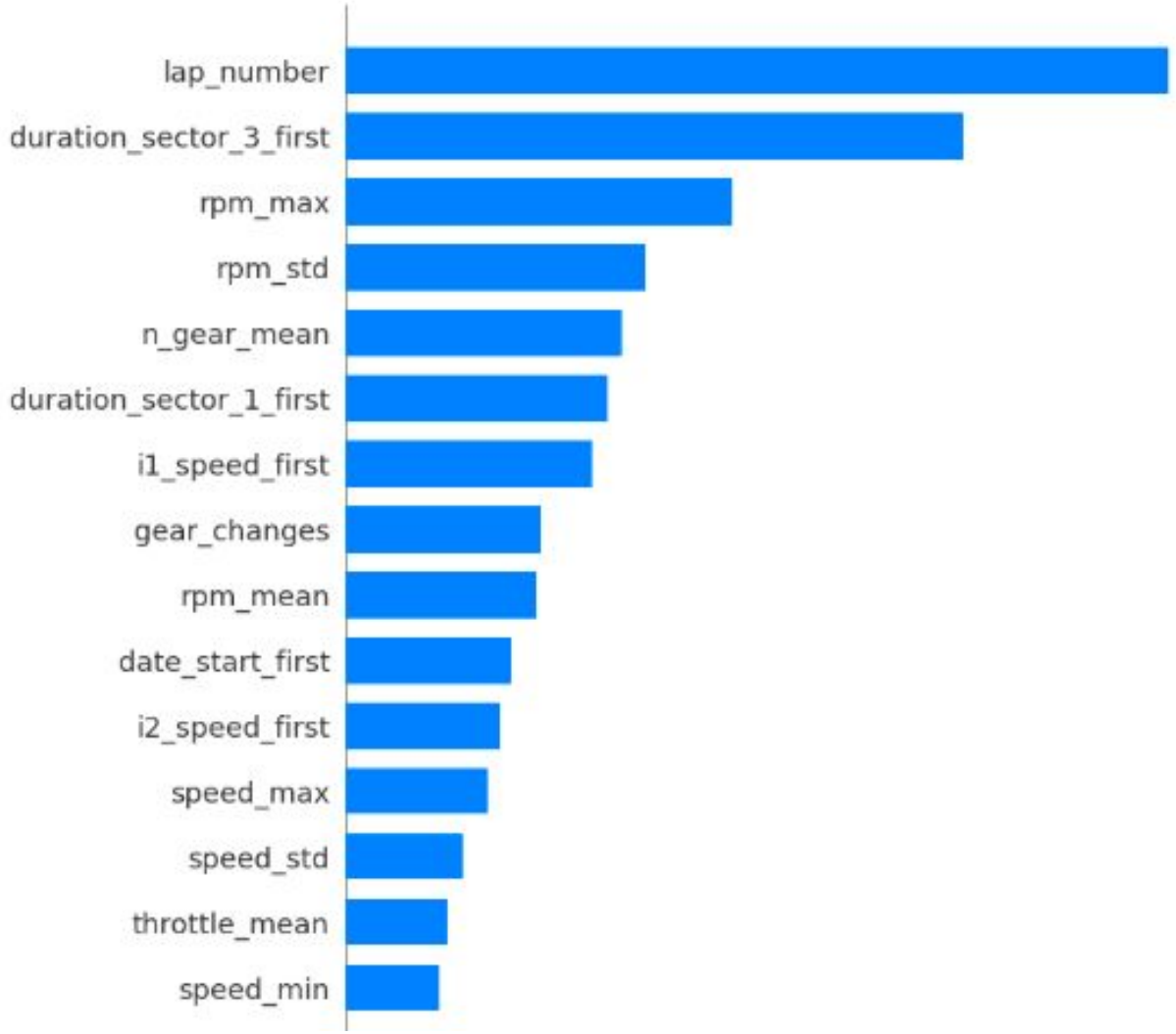



Weighted Average of Validation accuracy vs. Data Length per Race
Average validation: 0.87
Previous validation: 0.80
Validation accuracy
0.95
0.90
0.85
0.80
0.75
0.70
400
600
800
1000
1200
1400
Number of Samples (Data Length)

All Race data combined

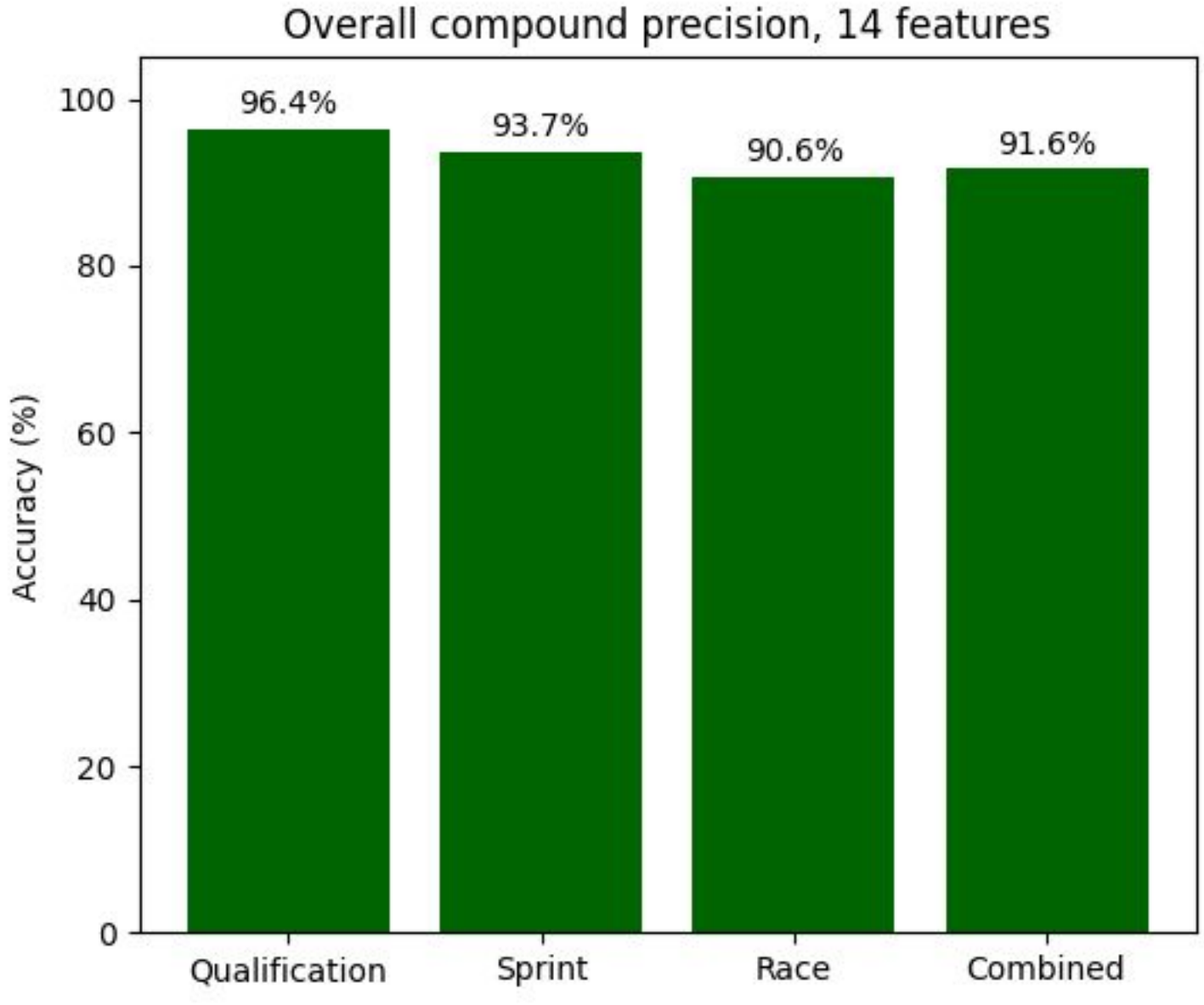


Race Melbourne

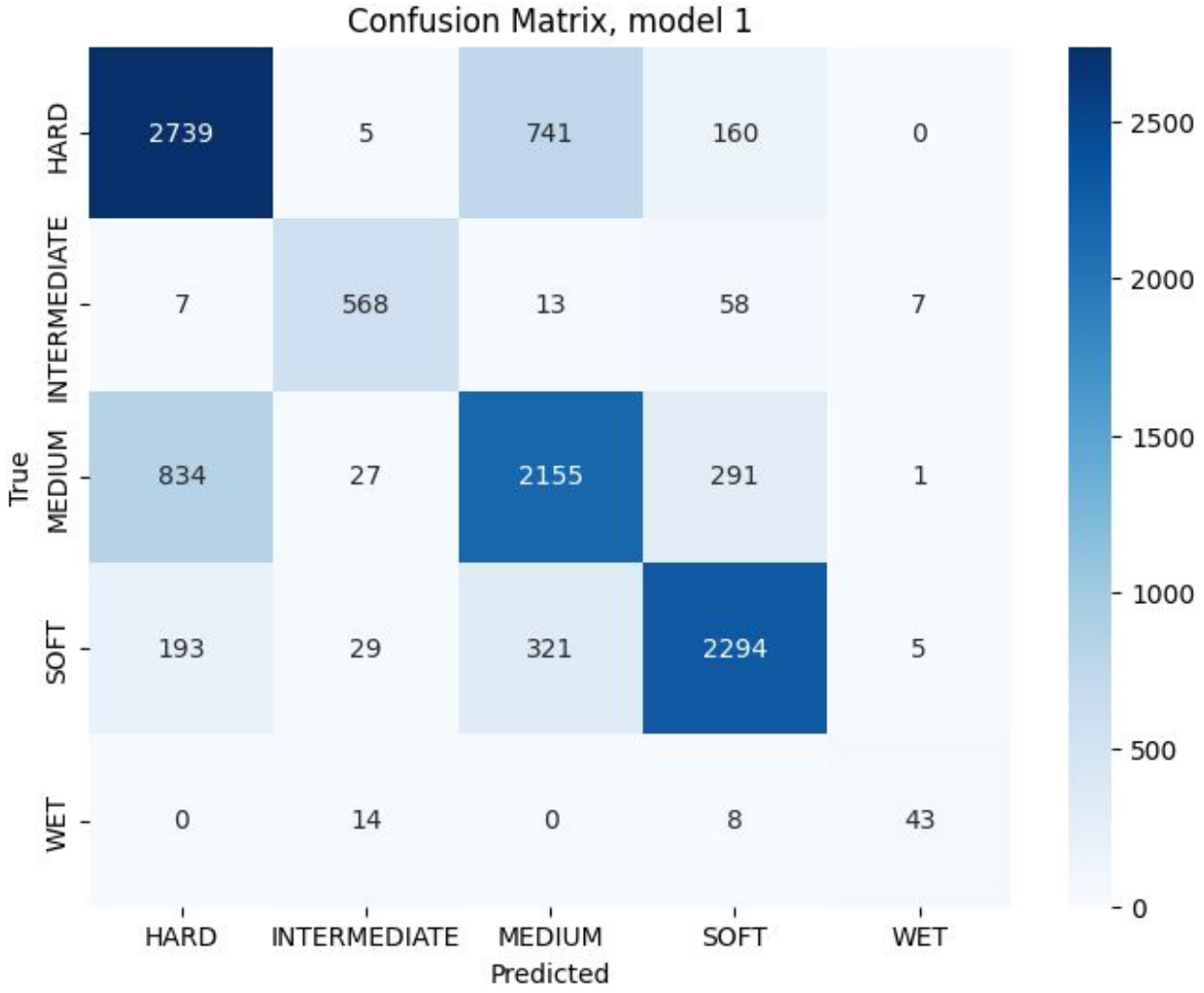


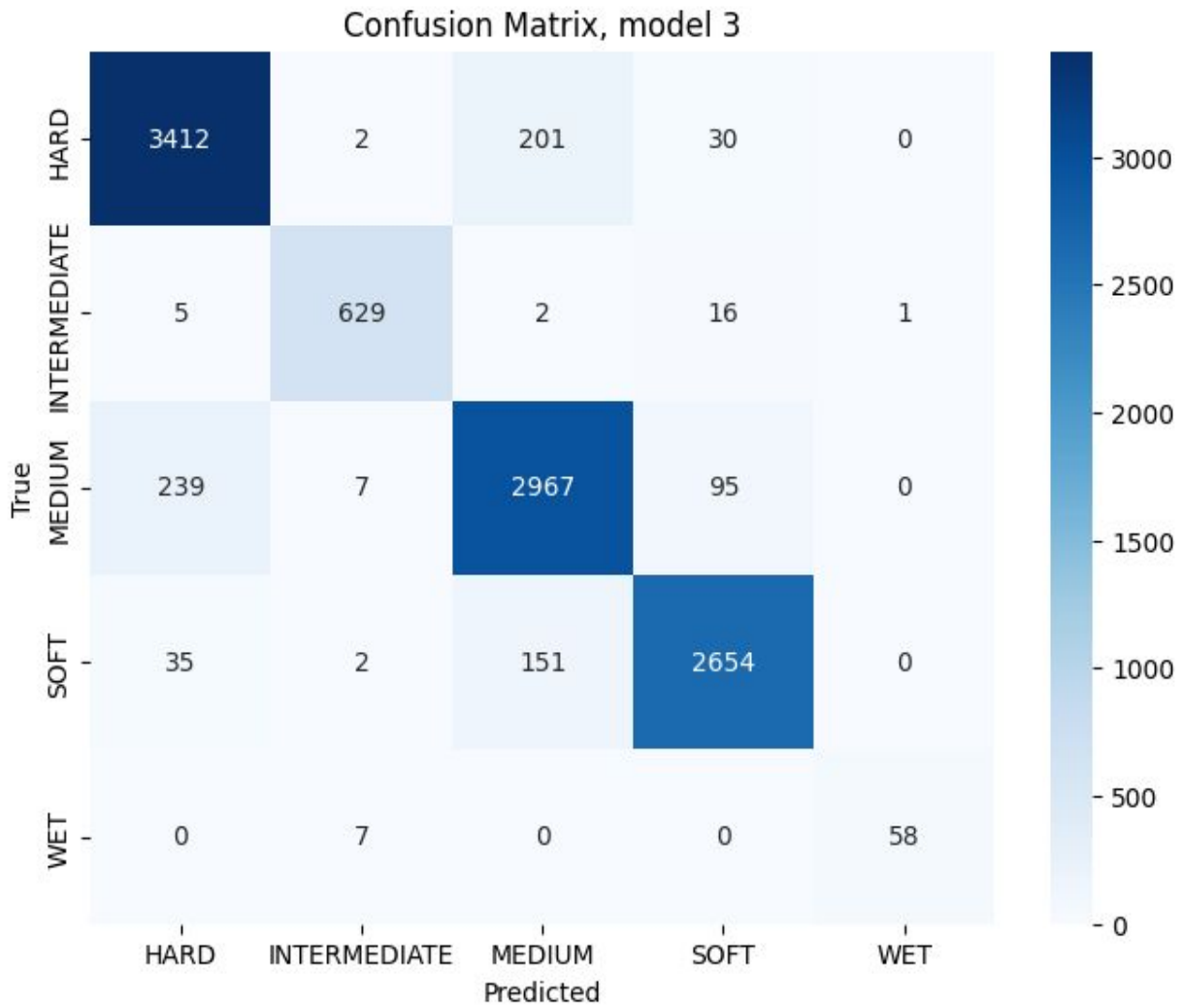
Classification of compounds - Model 3

- Goal to generalize
- One-hotting
- Feature engineering



Comparison of confusion matrices





Clustering of Car telemetry - driving styles

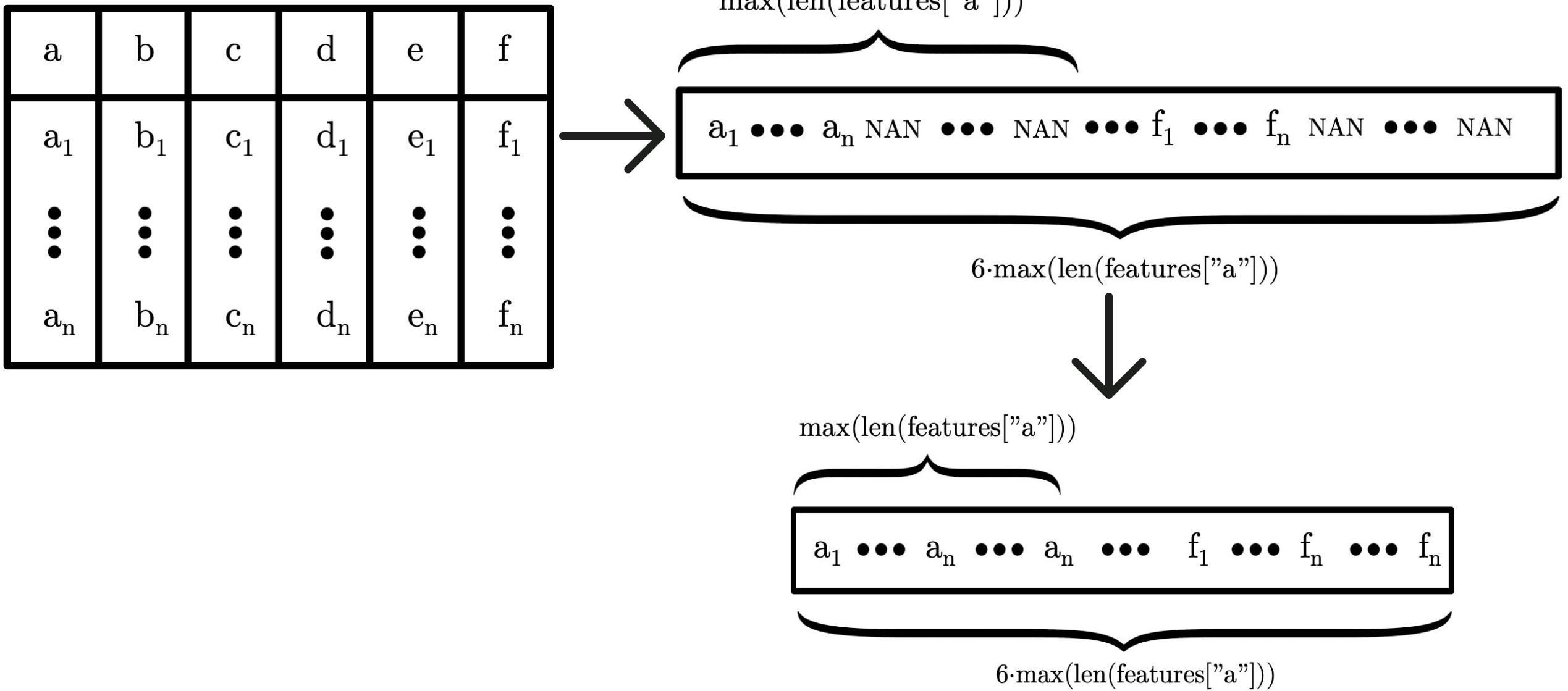
Baku Race 2023 - 9070

~400 entries pr. driver pr. lap. Interval of 373-507

51 laps in total, 20 drivers → ~400,000 entries in total

Goal: Summarize all telemetry data pr. driver pr. lap into one entry

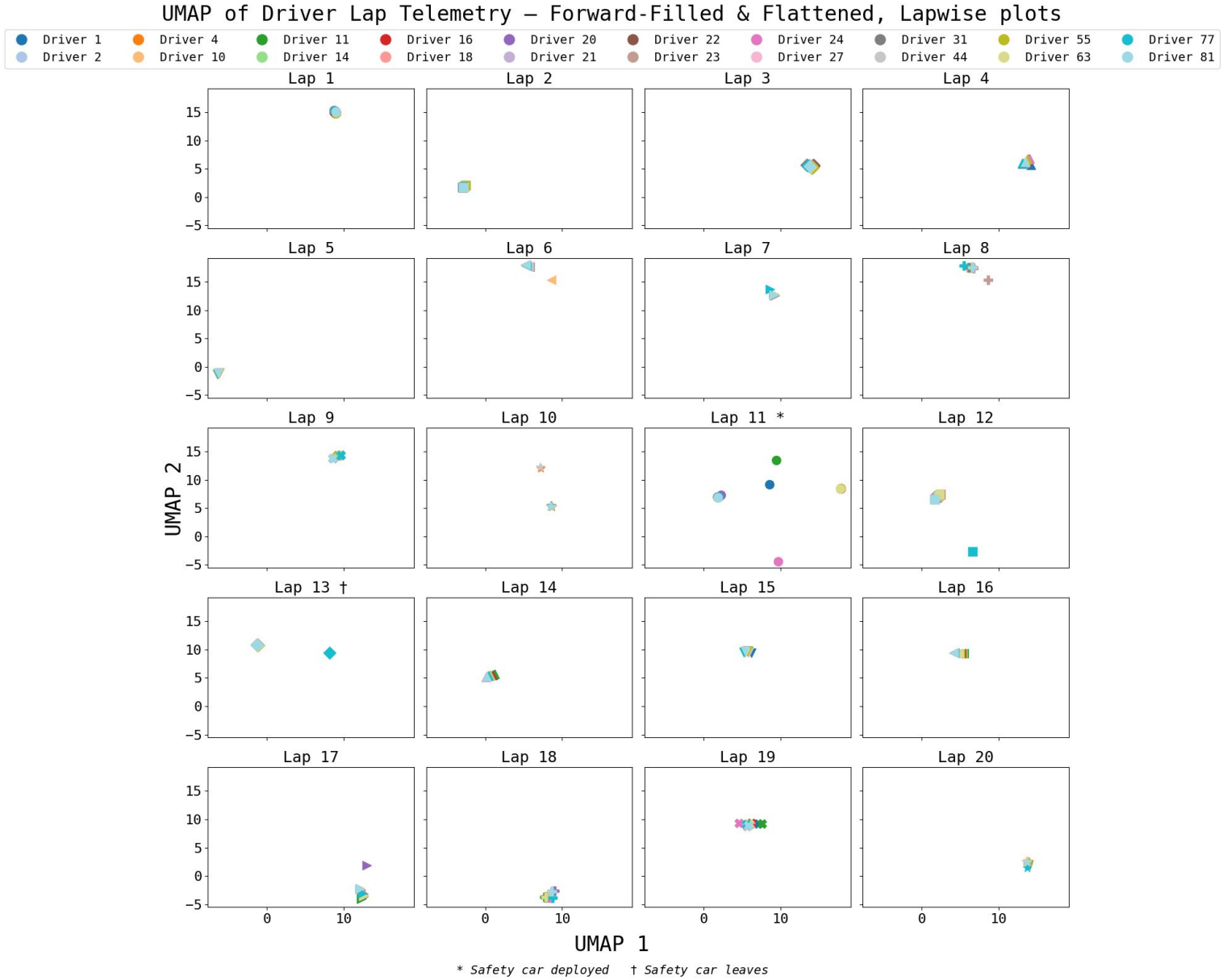
Initial idea: Ravelling data → padding with NANs → replacing NANs with last value



Ravelling approach - conclusions

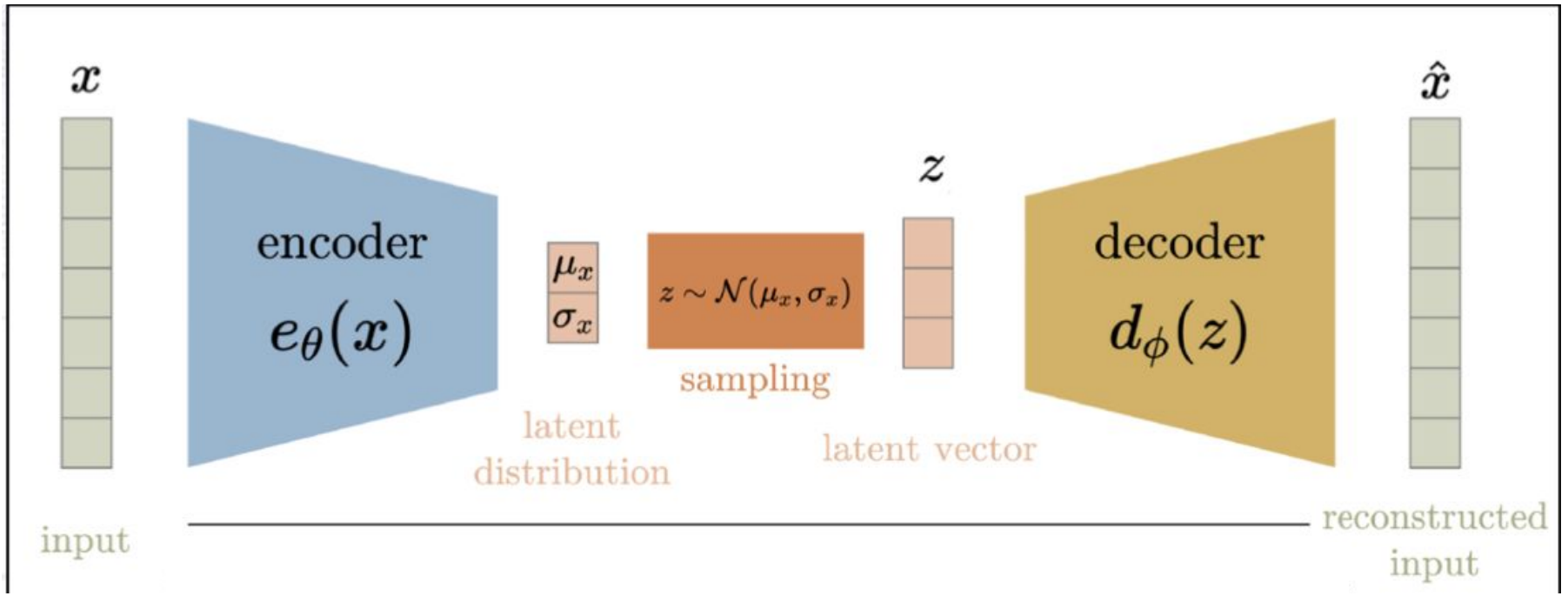
Handles variable length poorly

Nearly identical driving profiles



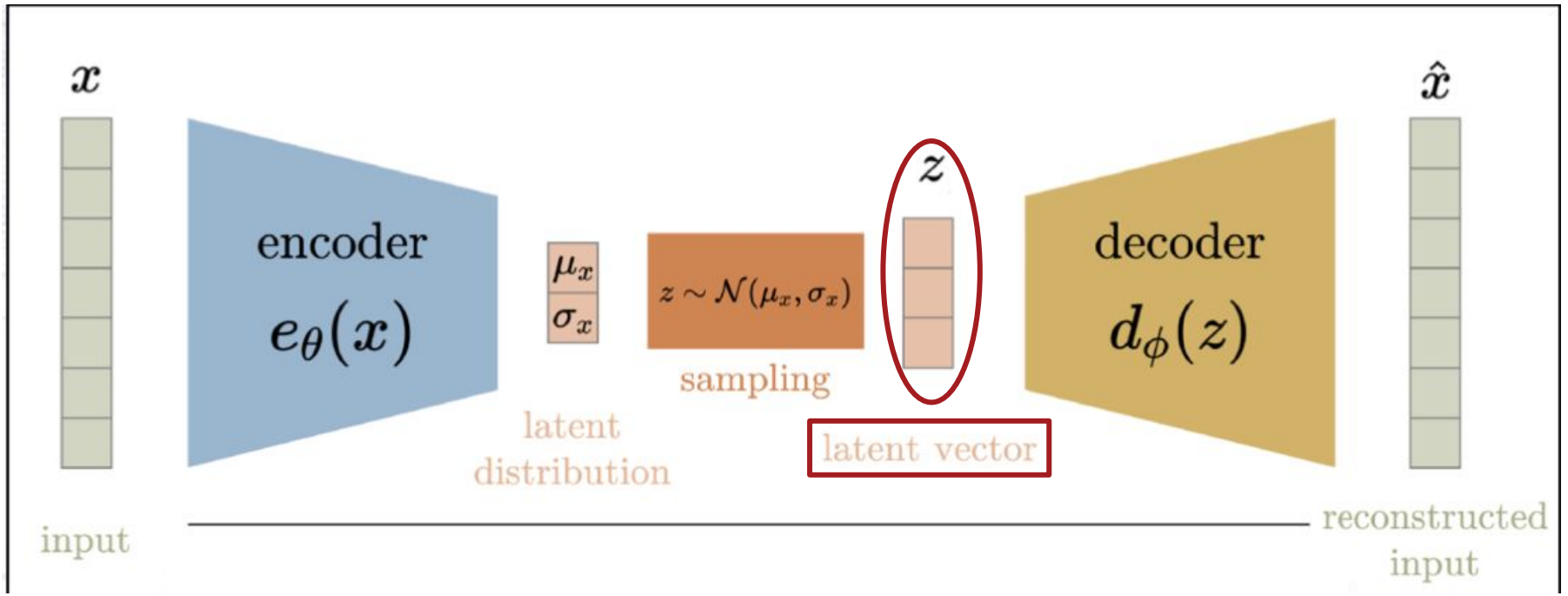
Introducing VAEs

Source: "AutoEncoders", AML slides (https://www.nbi.dk/~petersen/Teaching/ML2026/Week4/ML2026_AutoEncoders_GANs.pdf)



Introducing VAEs

Source: "AutoEncoders", AML slides (https://www.nbi.dk/~petersen/Teaching/ML2026/Week4/ML2026_AutoEncoders_GANs.pdf)



Introducing VAEs

Handles variable length

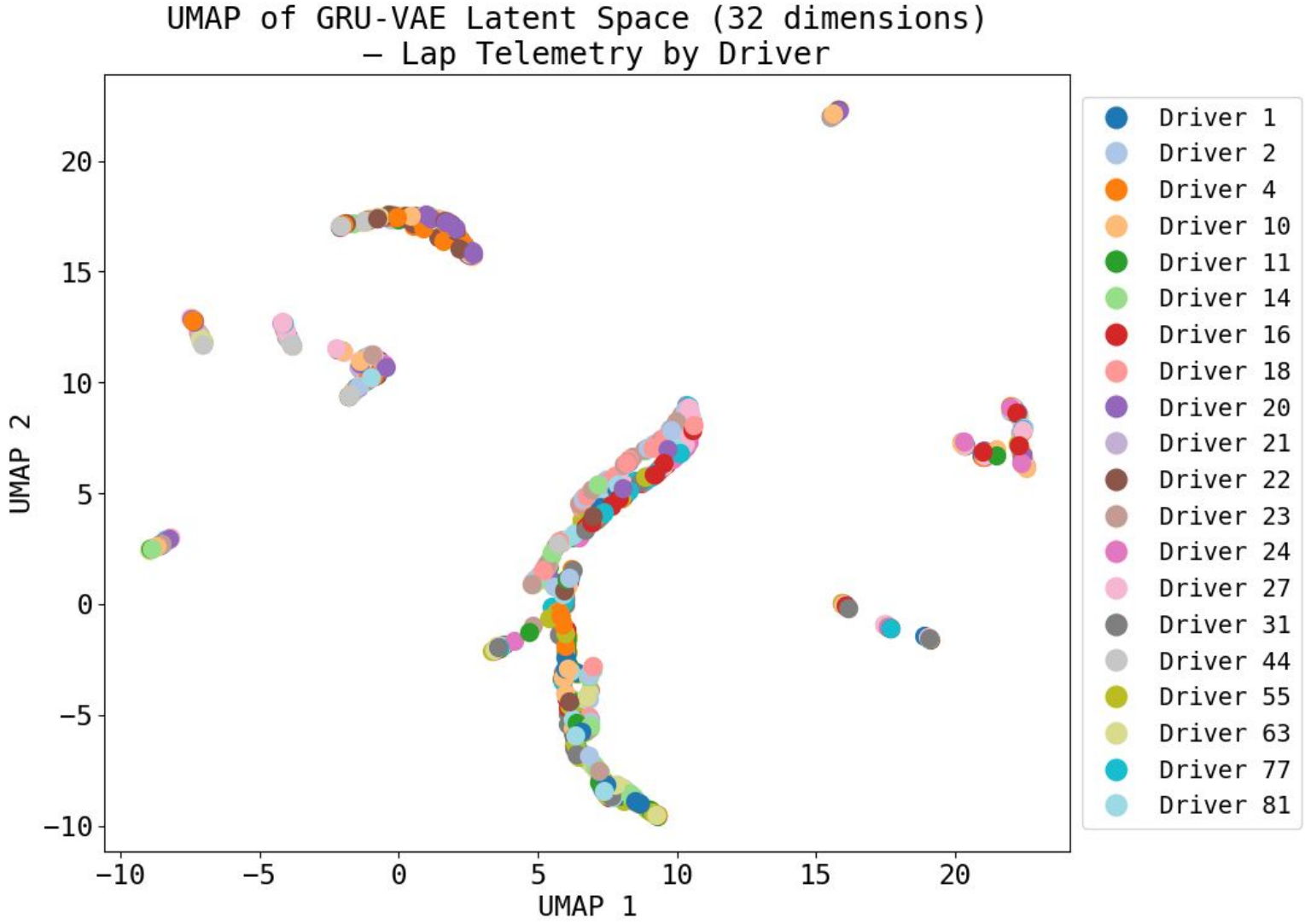
Compresses all telemetry data into a single latent vector with 32 dimensions (pr. driver pr. lap)

GRU-VAE model

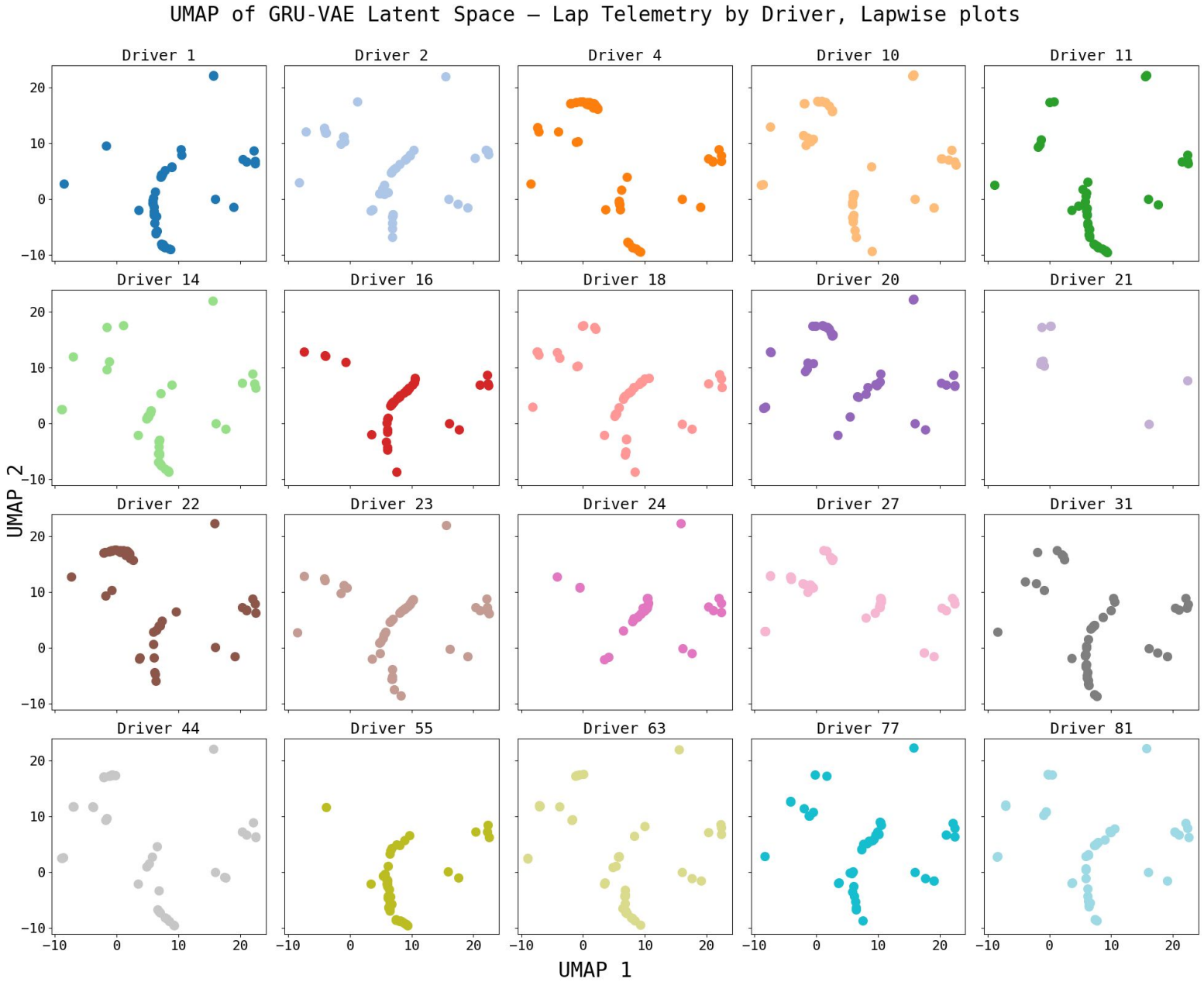
Trained on the first 10 and last 10 laps of the race - interpolation

VAE results

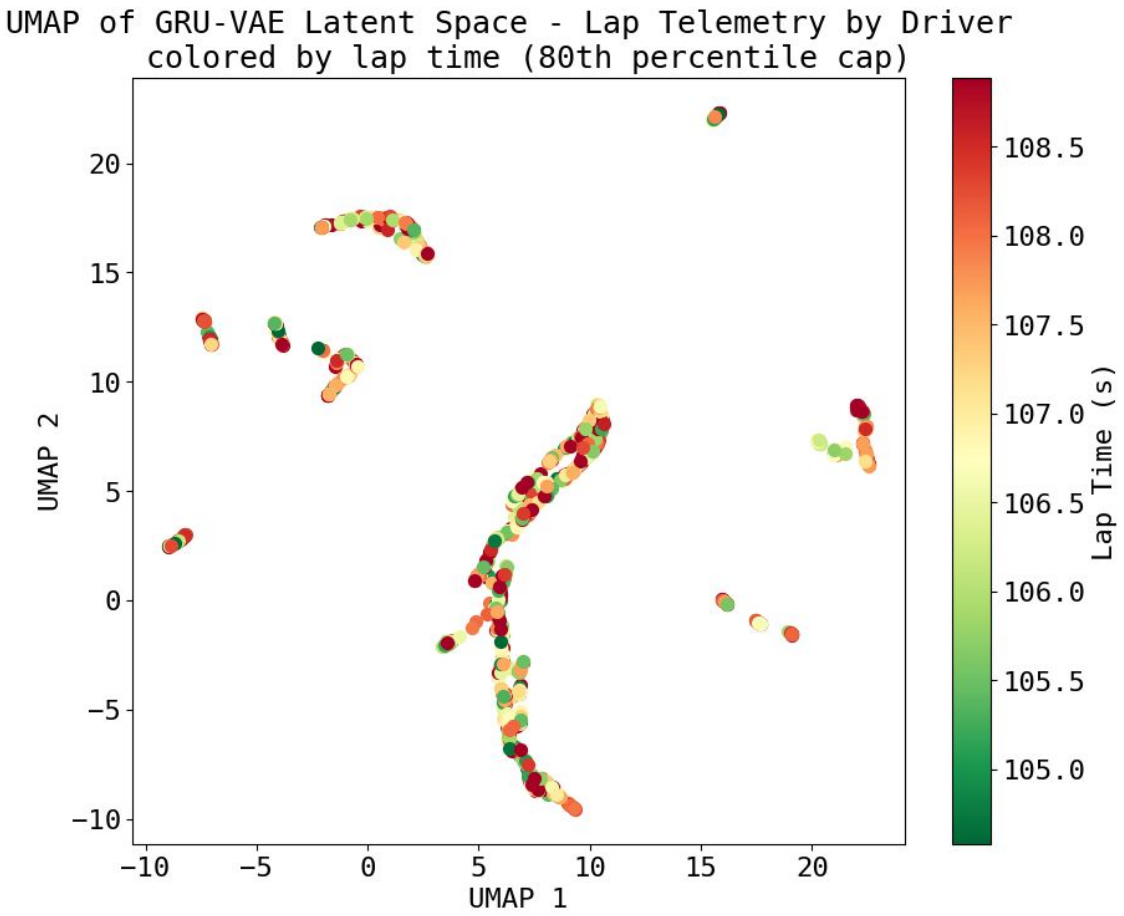
No driver seems to have a distinct driving style

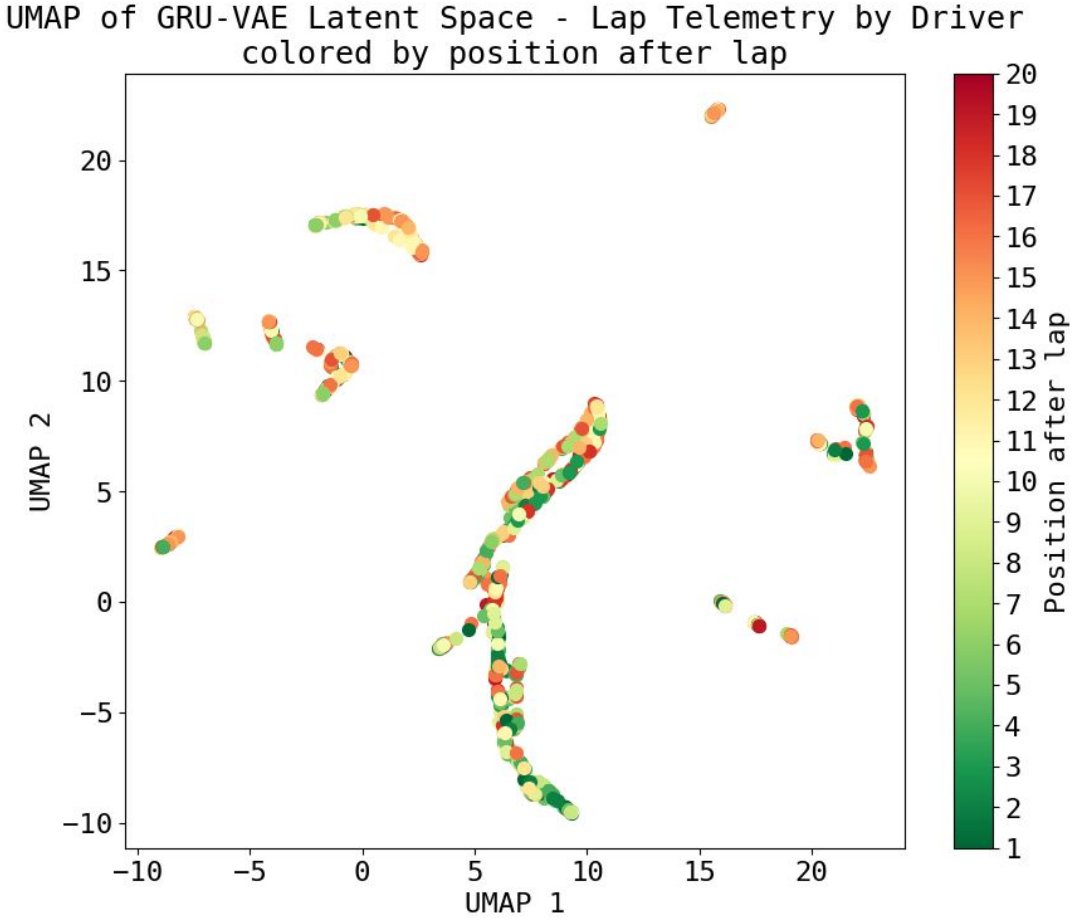


VAE results


UMAP of GRU-VAE Latent Space – Lap Telemetry by Driver, Lapwise plots
Driver 1
Driver 2
Driver 4
Driver 10
Driver 11
Driver 14
Driver 16
Driver 18
Driver 20
Driver 21
Driver 22
Driver 23
Driver 24
Driver 27
Driver 31
Driver 44
Driver 55
Driver 63
Driver 77
Driver 81
UMAP 2
UMAP 1

What determines the driving style?





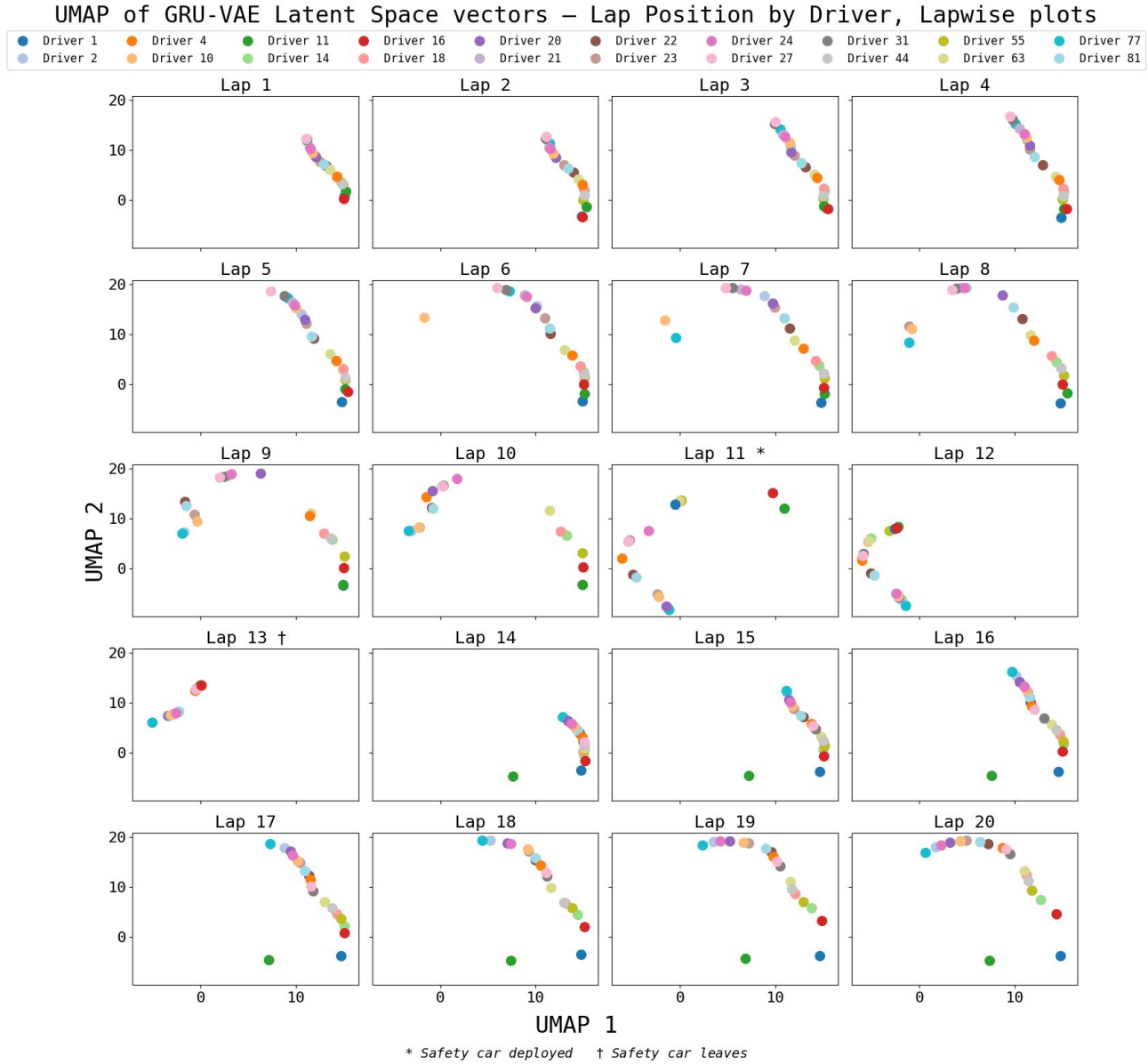
Introducing Pelotons: Turning location and time (x, y, t) into latent vectors

Using Silhouette score and inertia

to determine amount of clusters (pelotons)

Silhouette score requires 2+ clusters

Graduate student approach



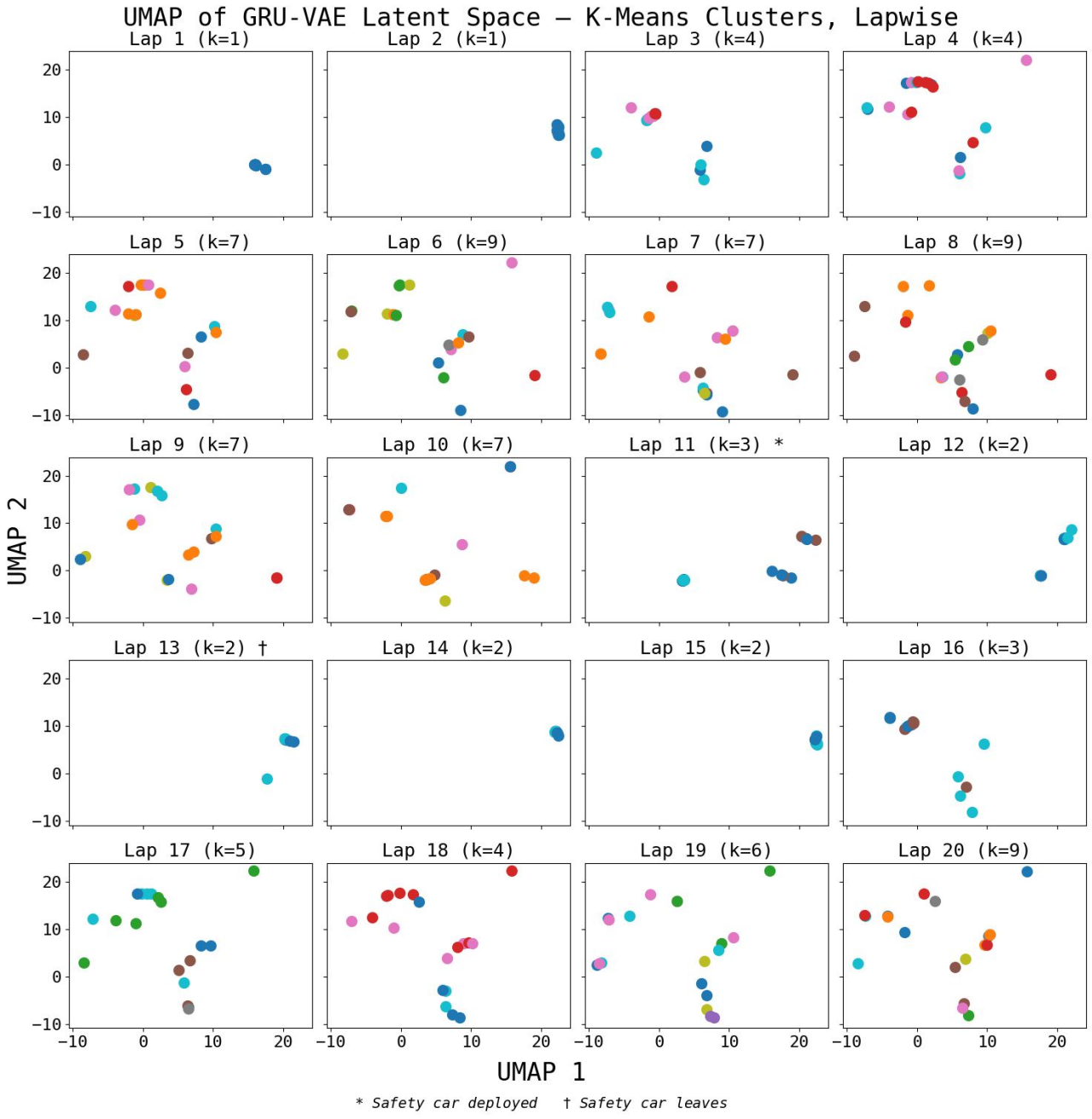
Clustering by Peloton

Peloton clustering explains some of the structure but not all

Presumably circuit layout, tires, and weather may explain some of it

Start of race + safety car

Order → disorder



Lap-time regression w/ XGBoost on VAE-GRU LVs Setup

Initial idea:

Use VAE-GRU LVs as features to predict lap-times on the fly.

Predict lap *j* w/ model trained on *j*-1 prior laps

Information leakage problem:

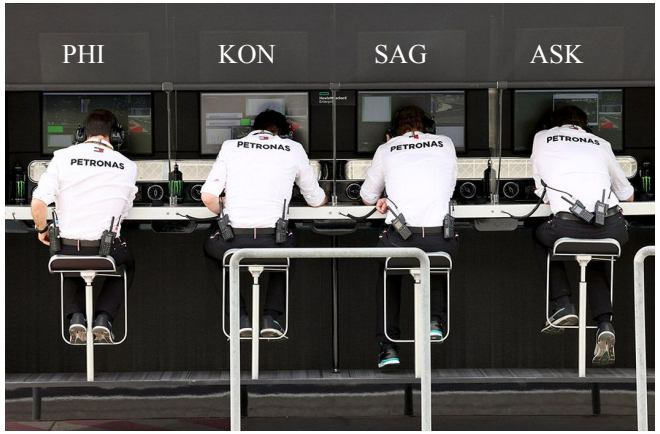
VAE-GRU LVs constructed from initial and last 10 laps

Fix = Retrain VAE-GRU model only on 10 first laps

Approach:

Initially train ONLY on first 10 laps

Then employ expanding window approach



Picture from:
(https://www.crowdstrike.com/en-us/blog/pit-stop-vs-pit-wall-teamwork-metaphor/)

Lap-time regression w/ XGBoost on VAE-GRU LVs Overview

Model:

Simple XGBoostRegressor w/ hyperparameters:

```
{'n_estimators': 300, 'max depth': 3, 'learning rate': 0.05, 'subsample': 0.8,
            'colsample_bytree': 0.6, 'colsample_bylevel': 0.8}
```

Data:

2023 LVs = 959 rows x 32 cols

2024 LVs = 816 rows x 32 cols

Three sets:

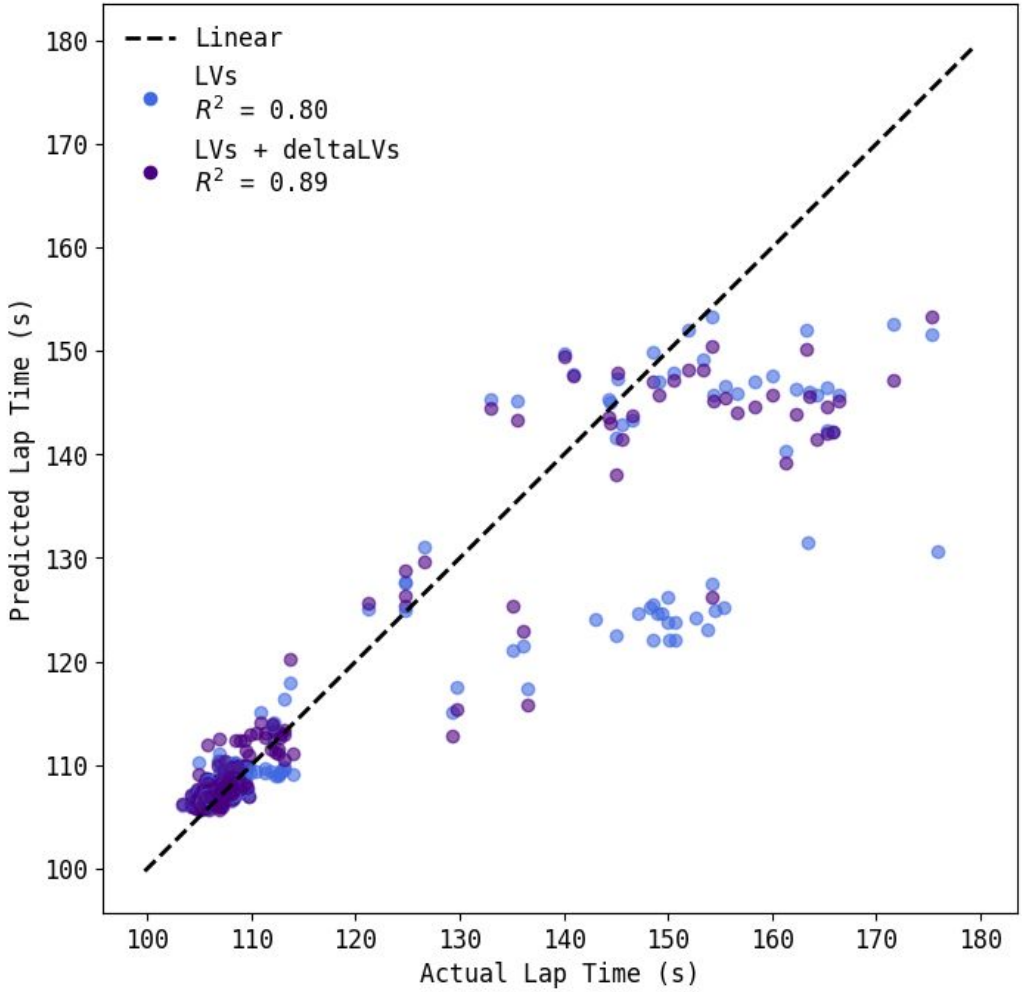
Predictions for 2023 (2024) trained solely on LVs for 2023 (2024)

Predictions for 2023 (2024) trained on LVs + deltaLVs for 2023 (2024)

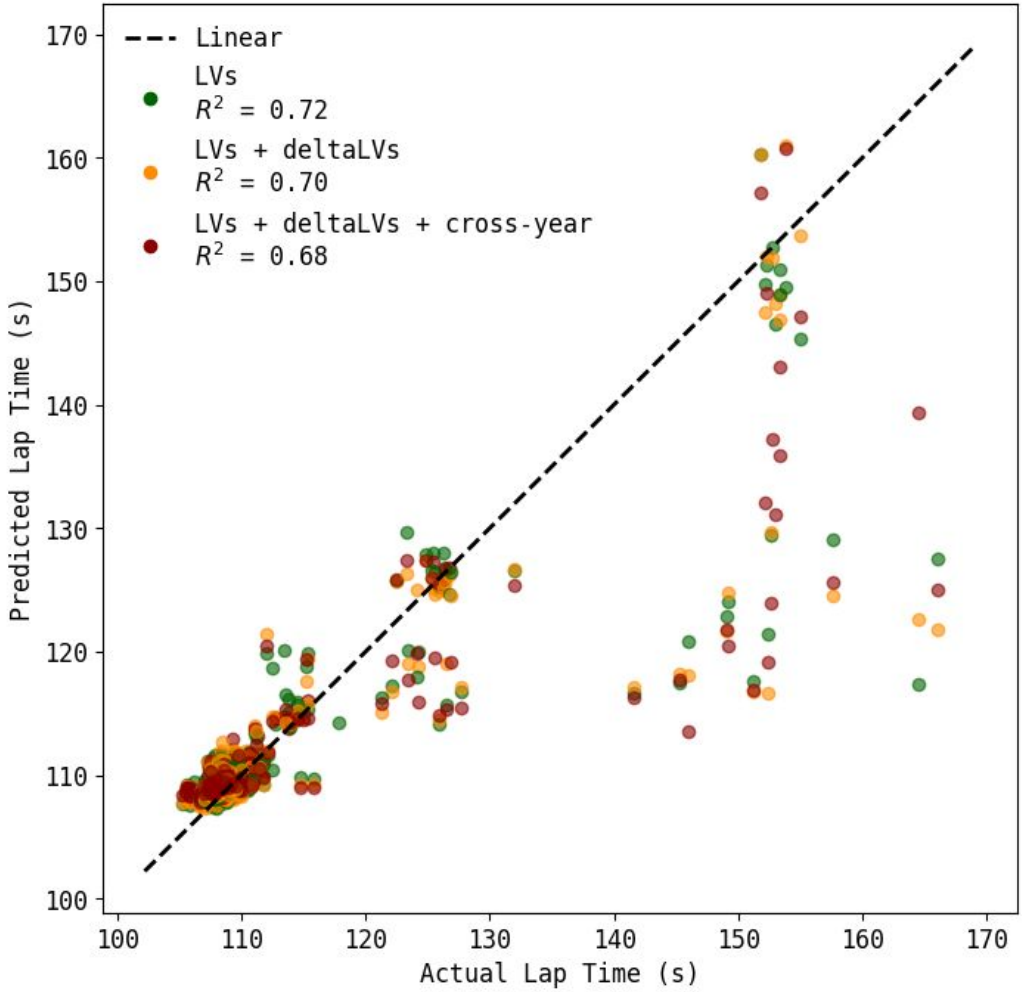
Predictions for 2024 trained on 2023 and 2024 LVs + deltaLVs

Lap-time regression: All driver results

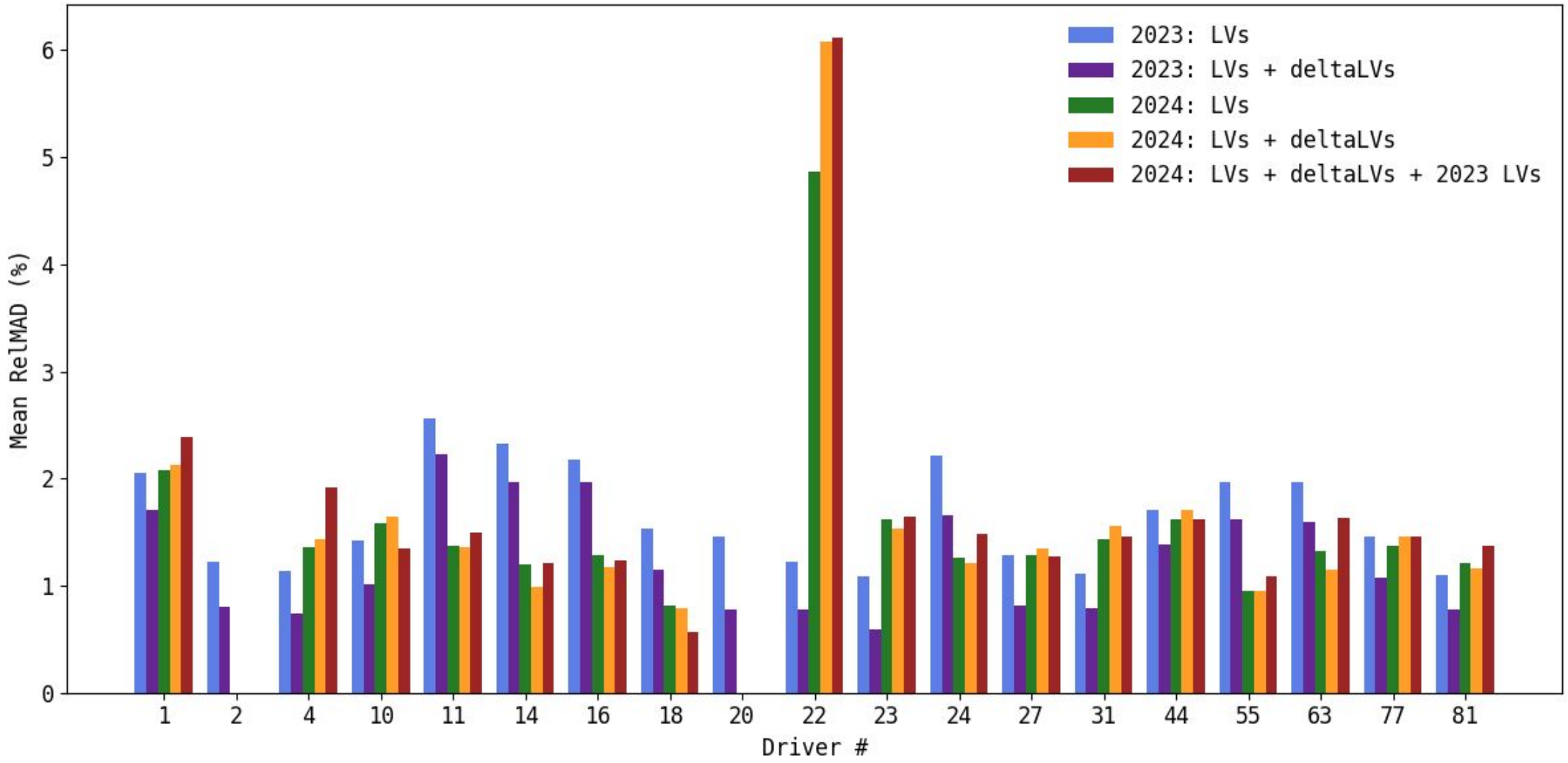
2023



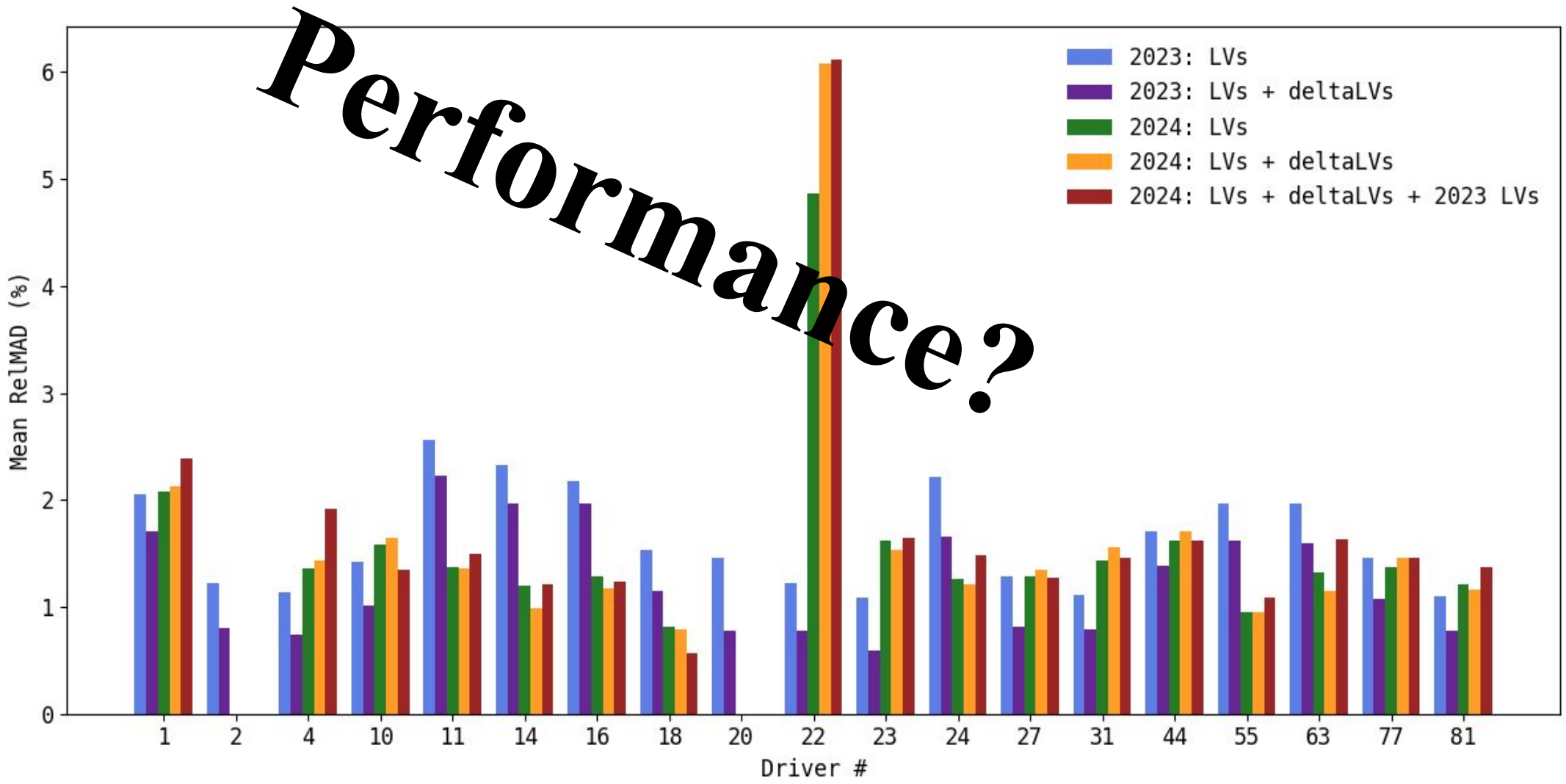
2024



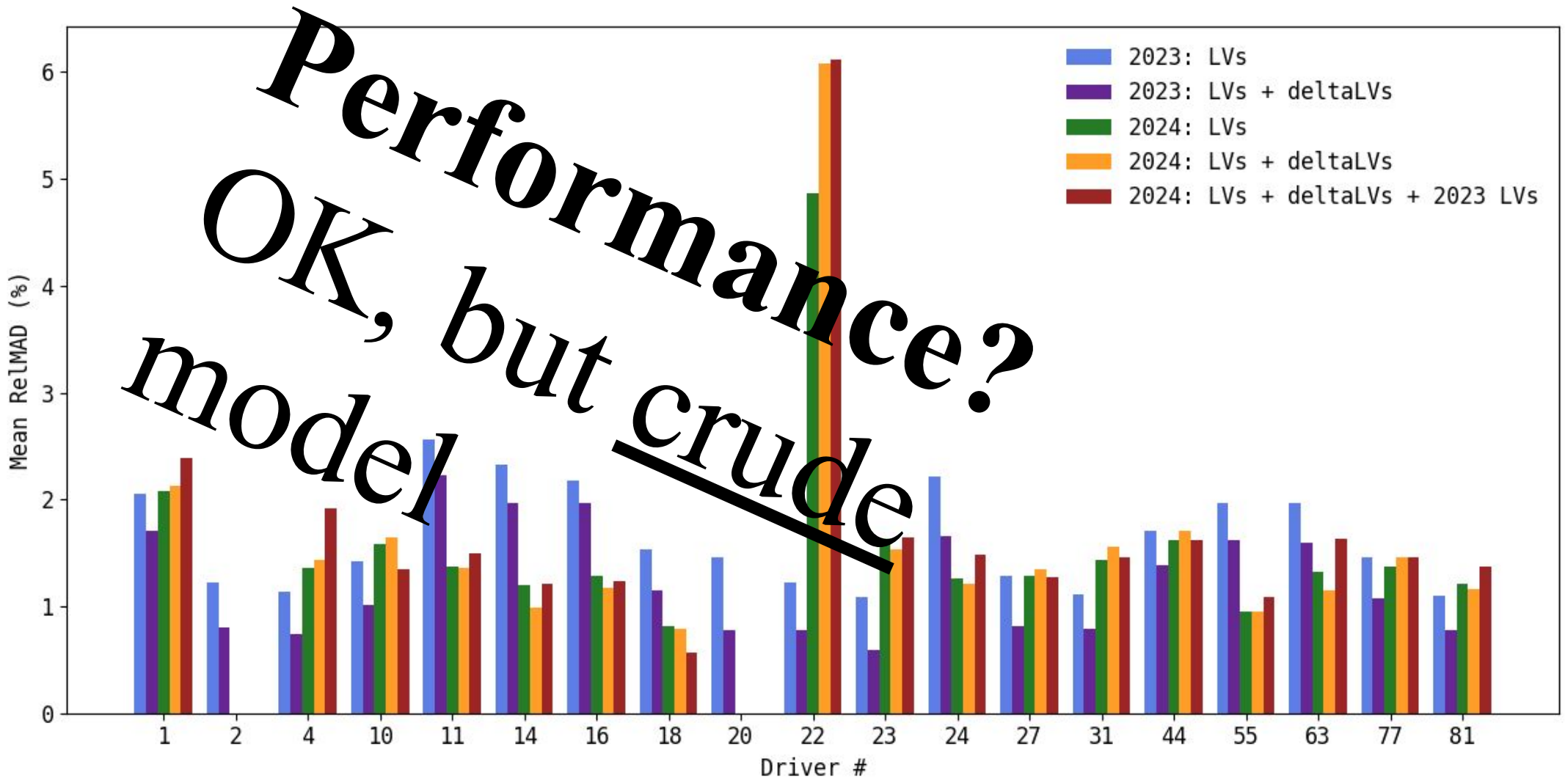
Lap-time regression: Mean RelMAD, all predictions



Lap-time regression: Mean RelMAD, all predictions



Lap-time regression: Mean RelMAD, all predictions



Lap-time regression w/ XGBoost on VAE-GRU LVs
Outlook

Model adjustments:

- Hyperparameter optimization – on the fly or?
- Time-series appropriate model (LSTM, etc.)
- Safety-car fix - one-hot encode?

With more time:

- Predict sector times instead (S1, S2, S3)
- Generalizing?
- Additional features – compounds, weather, etc.?

Magnum opus:

- On the fly predict race champion

Picture: Kimi Antonelli last weekend at Monaco 5th win in a row

Questions

Appendix

So… what is a **G**ated **R**ecurrent **U**nit (GRU)?

Source: Dobilas, Saul, "GRU Recurrent Neural Networks - A smart way to predict sequences in Python" in TowardDataScience (https://towardsdatascience.com/gru-recurrent-neural-networks-a-smart-way-to-predict-sequences-in-python-80864e4fe9f6/)

Overview:

An RNN made to solve vanishing/exploding gradient problem

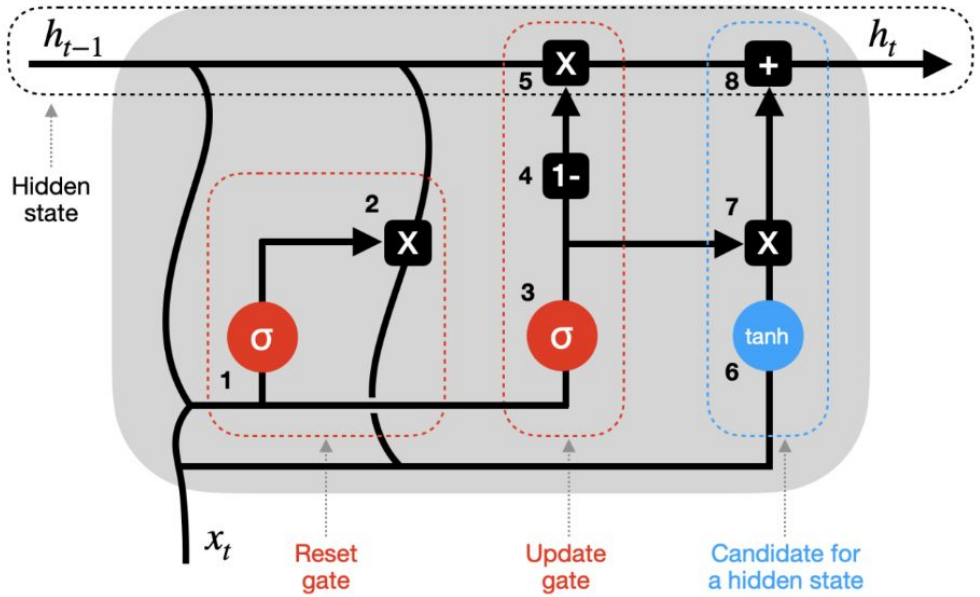
Similar to LSTM but simpler with fewer gates

The gist:

At time *t*, use:

Reset gate, *r(t)*, to decide balance of *h(t-1)* and candidate *h(t)*

Update gate, *z(t)*, to decide information retention from hidden state, *h(t-1)*, when constructing *h(t)*



$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$
$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t$$

Source: From Wikipedia, "Gated Recurrent Unit" (https://en.wikipedia.org/wiki/Gated_recurrent_unit)

Model training of GRU-VAE

```
def vae_loss(recon, x, mu, logvar):
    recon_loss = nn.functional.mse_loss(recon, x, reduction='mean')
    kl_loss    = -0.5 * torch.mean(1 + logvar - mu.pow(2) - logvar.exp())
    return recon_loss + 0.001 * kl_loss
```

The loss function combines reconstruction error with a regularisation term that keeps the latent space structured. A KL weight of 0.001 was chosen to prioritise proper reconstruction of the latent vectors resulting in more meaningful latent vectors.

Telemetry:	**Location:**
Best loss: 0.4449	Best loss: 0.0641
Training time: 22.48 minutes	Training time: 16.63 minutes

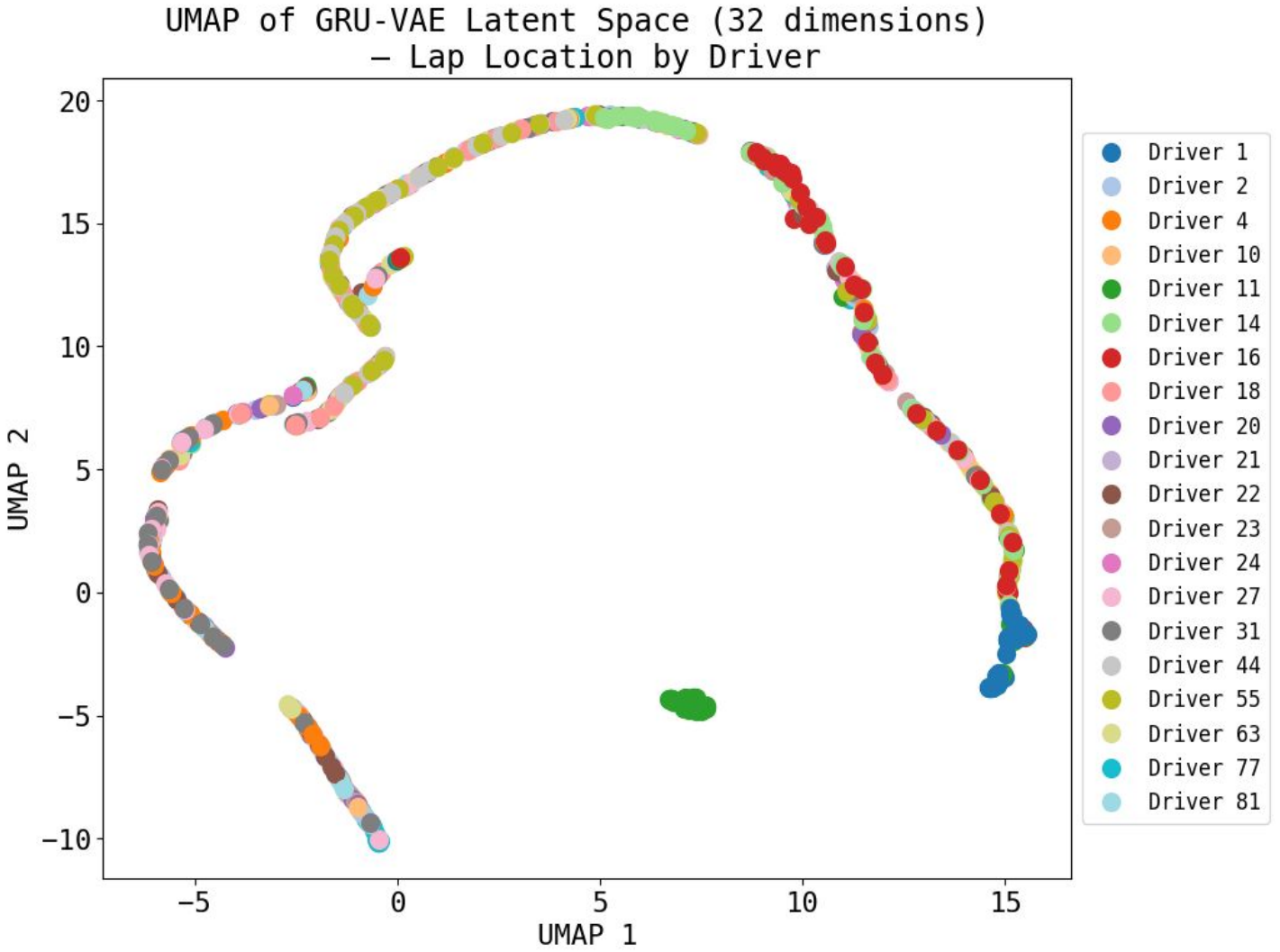
Both models trained on CPU - Metal (Apple GPU) not optimized for GRU

Clustering of driving styles
Continued

Clustering of location (x, y, t)

Pre-processing

Time is monotonically increasing across all laps. Time is normalized so that for each lap, the lowest value of t is subtracted from all others. This data is standardized. Despite the normalization μ=-0.0198 and var=0.8759 across all t_norm



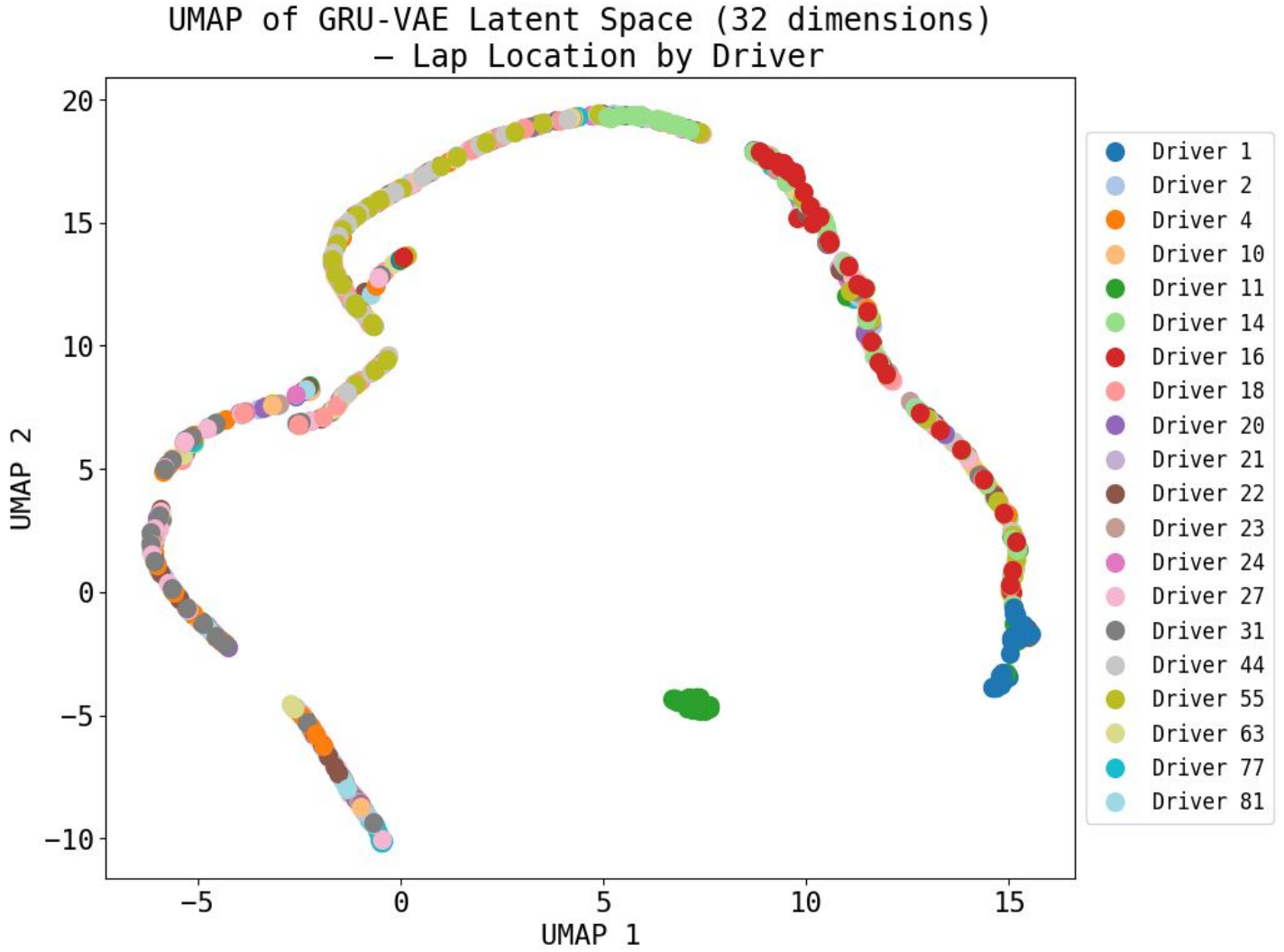
Clustering of location (x, y, t)

Interesting things to note

The darkgreen driver (driver 11)

is in a cluster by himself.

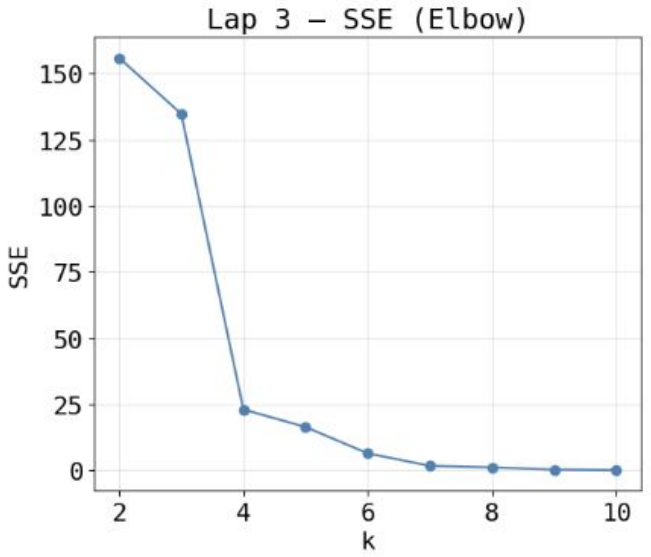
He is also the winner of the race

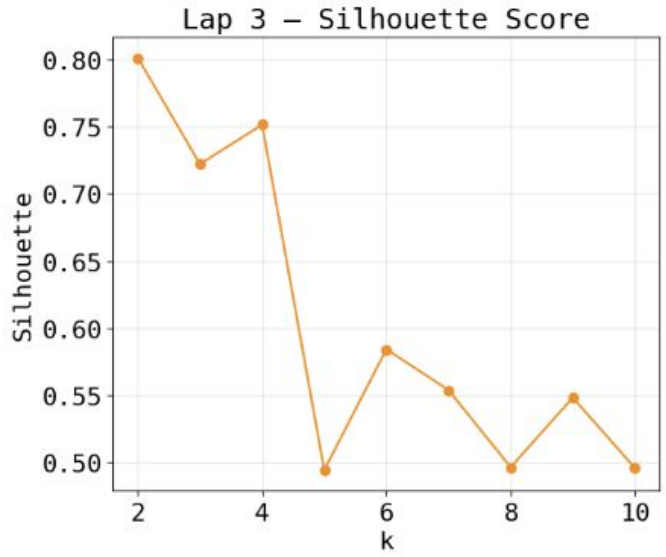


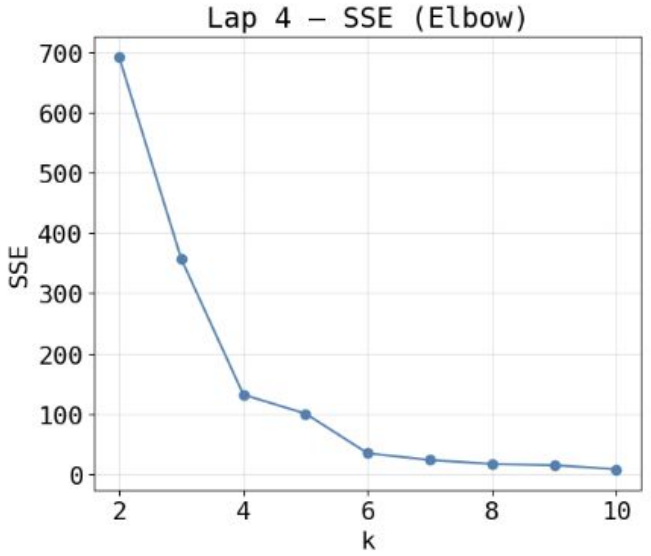
Clustering of location (x, y, t): Determining Pelotons

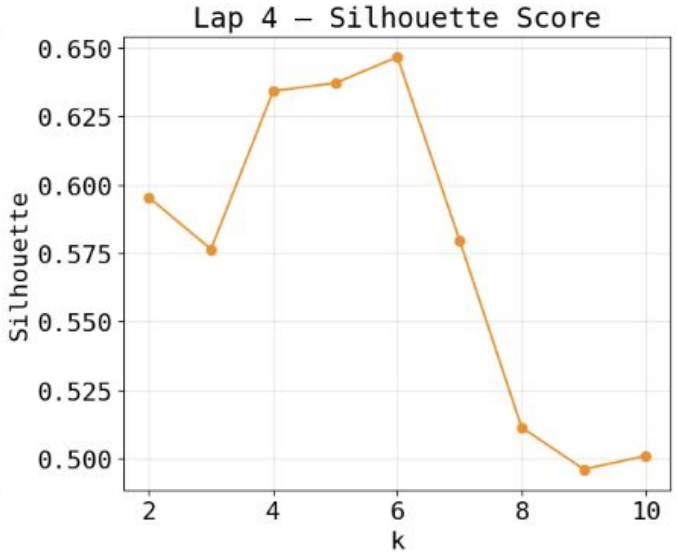
Two of the SSE and Silhouette plots used to determine the optimal amount of clusters

For both of these laps it was determined to be 4 (graduate student approach)



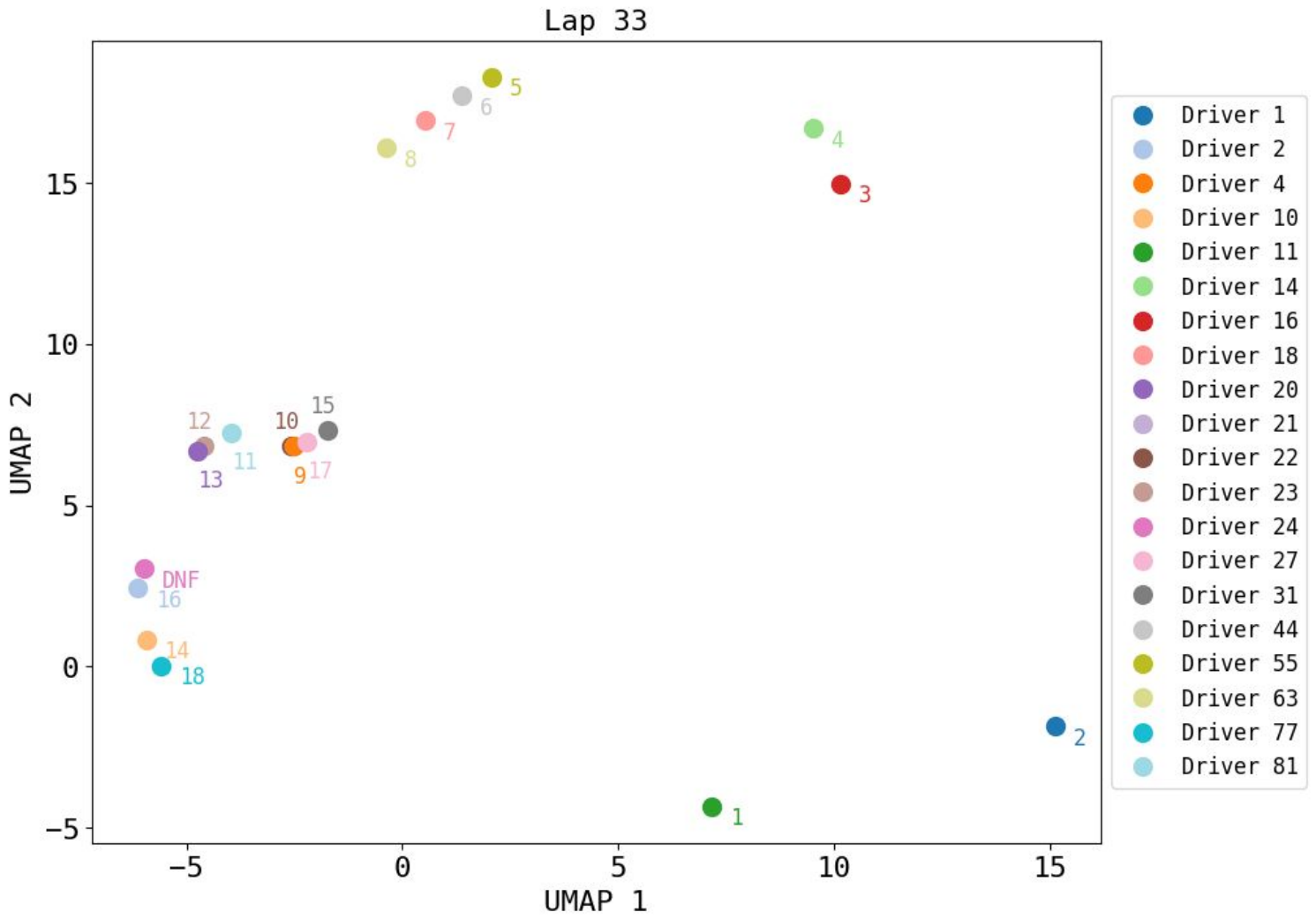




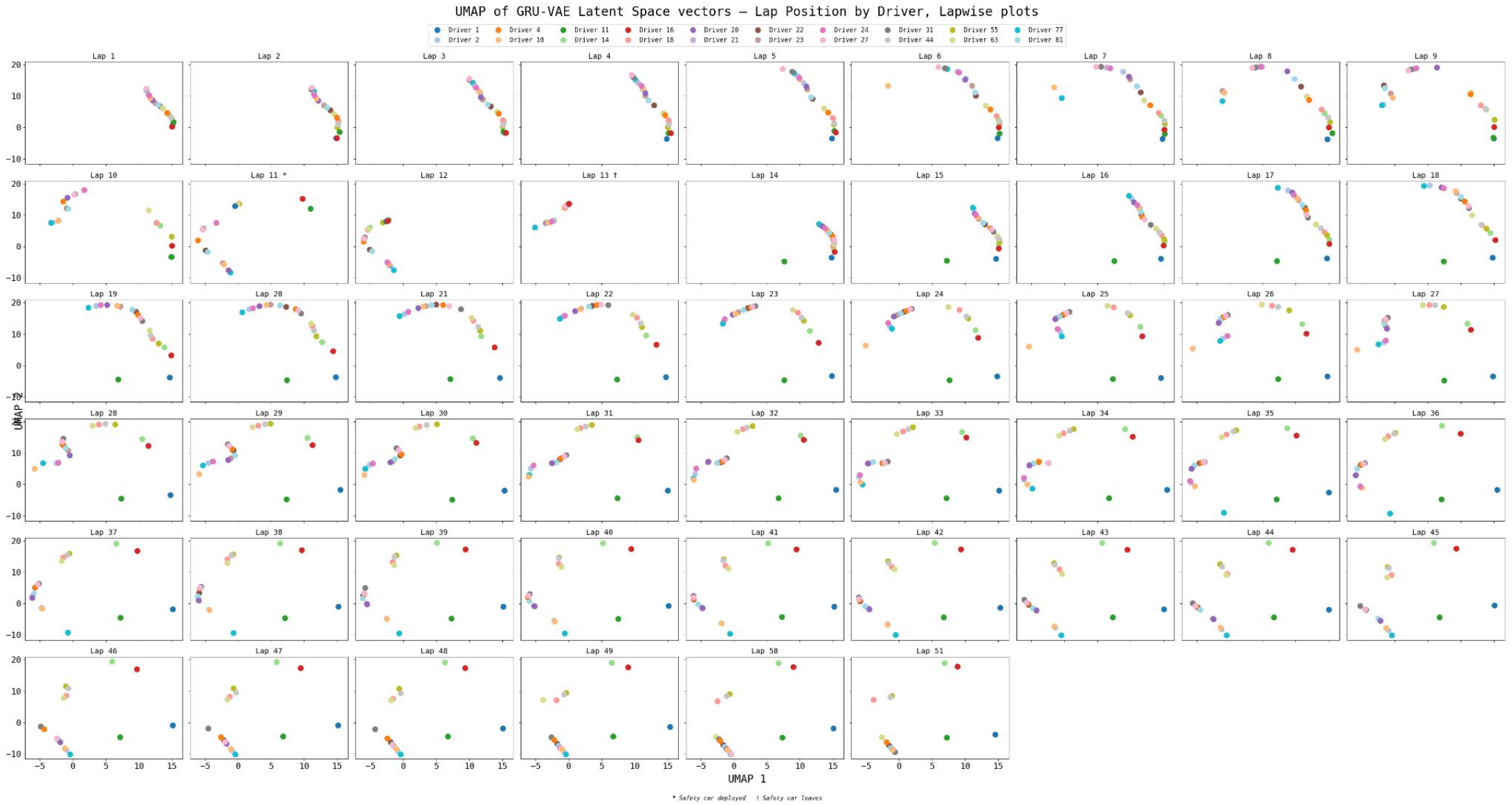


Clustering of location (x, y, t): Structured by finishing positions

Near lap 33, driver positions around the plot closely mirror their finishing positions (annotated), suggesting an equilibrium that holds for most drivers through to the end. The next slide shows all 51 laps.



Clustering of location (x, y, t): Lapwise plots for all 51 laps



Clustering of car telemetry and positions: How well does the model generalize?

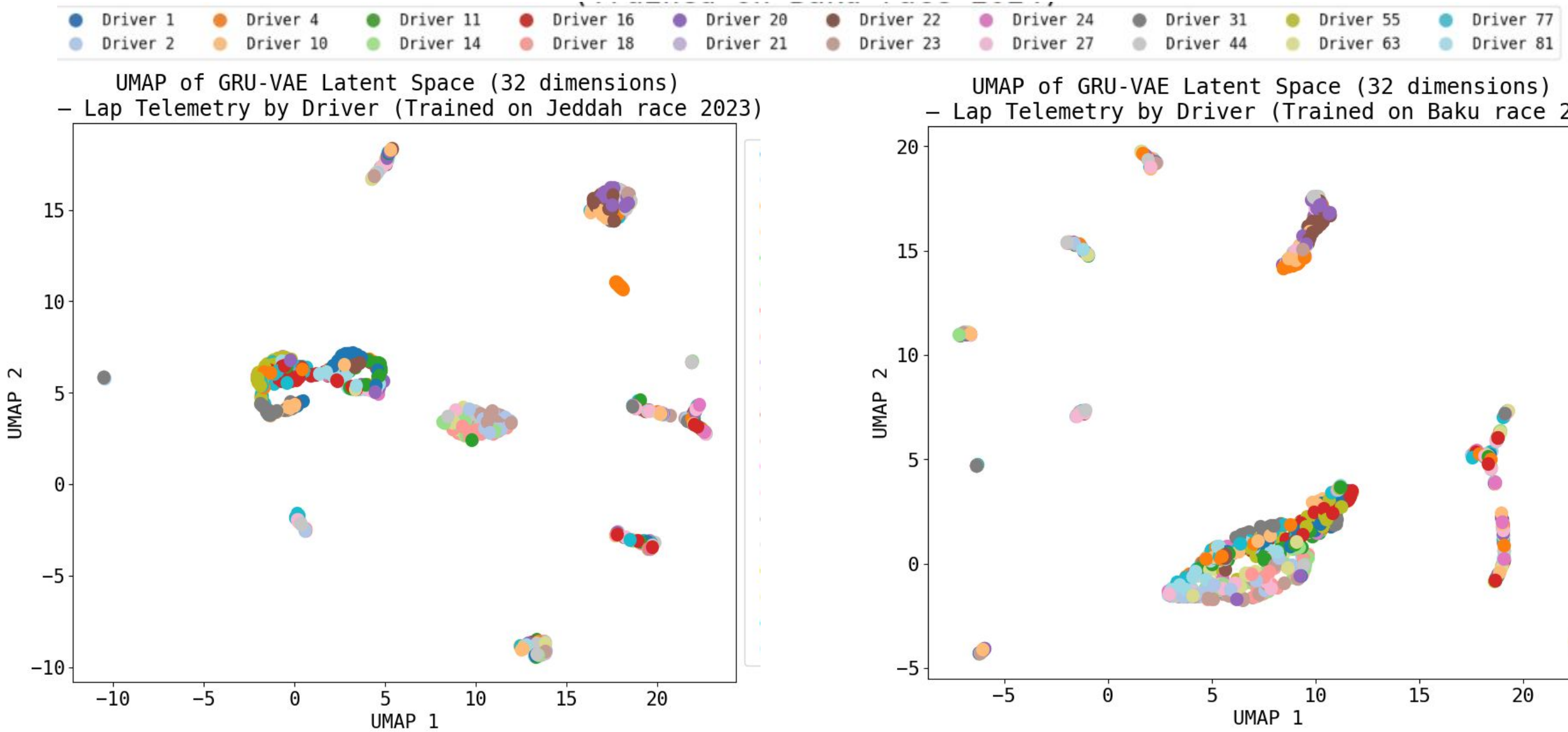
We wanted to examine how well the model generalizes.

Therefore we trained a model on a race from the same year (Jeddah Race 2023 - 7779) and a race on the same circuit from a different year (Baku Race 2024 - 9598)

We then extracted the latent vectors for the Baku Race 2023 - 9070, using both of the other models and comparing them with the original latents vectors.

As before we use a GRU-VAE model with 32 latent dimensions. The model is on the first 10 and last 10 laps of the race

How well does the model generalize: Comparison of the two models on telemetry

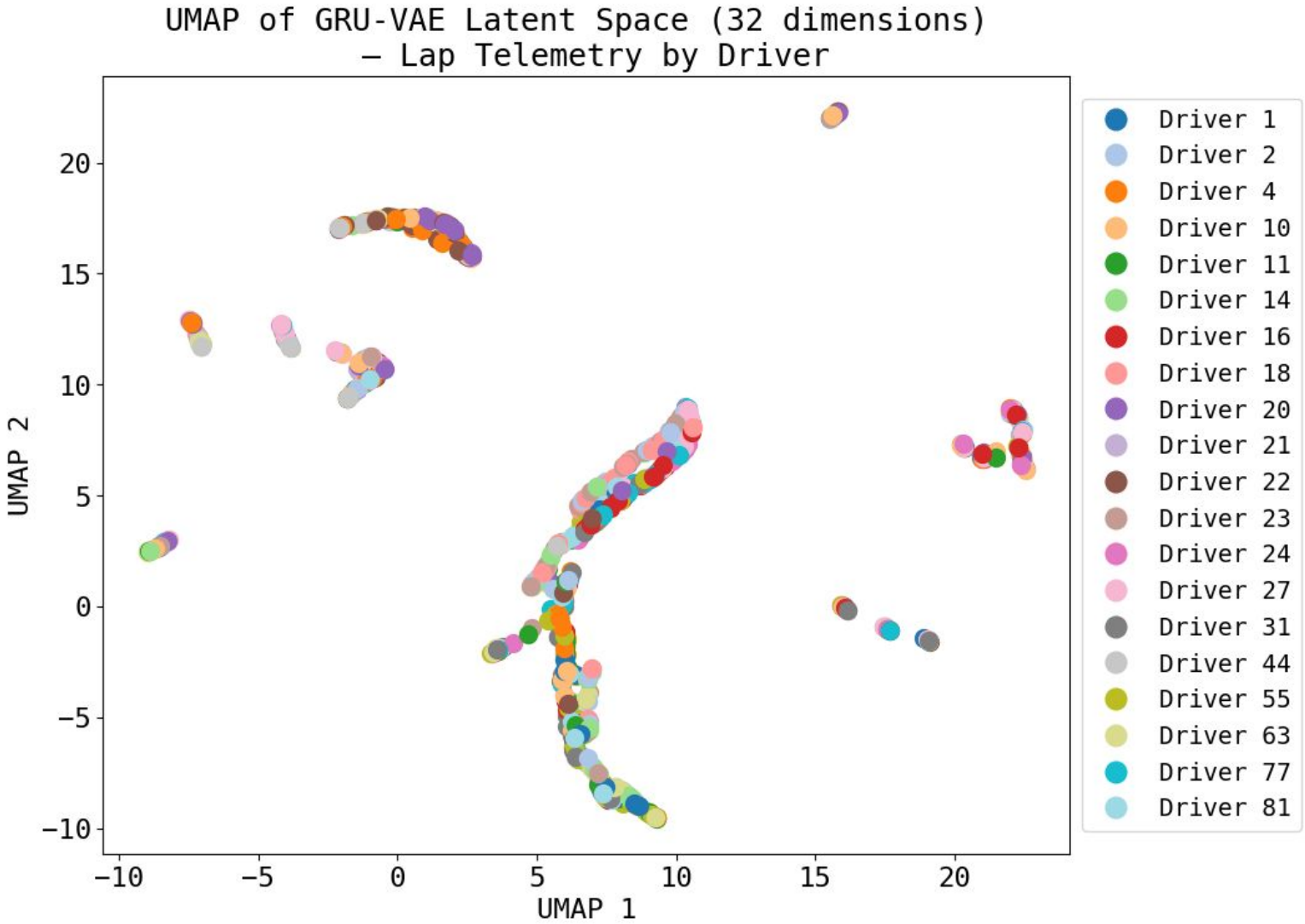


How well does the model generalize: Comparison of the two models on telemetry

Both models trained on different sessions from 9070 produce different latent vectors.

Visually it seems that the model trained on the same circuit, but different year (9598) it closer to the original latent vectors.

How do we quantify this?



How well does the model generalize: Comparison of the two models on telemetry

Averaging the UMAP distances between latent vectors from models trained on circuits 7779 (9598) and 9070 quantifies which model is closer. In the table below blue shows distances from 7779, red from 9598, black labels driver numbers. The 9598 distances are smaller for every driver, indicating that circuit layout plays a key role in determining driving style.

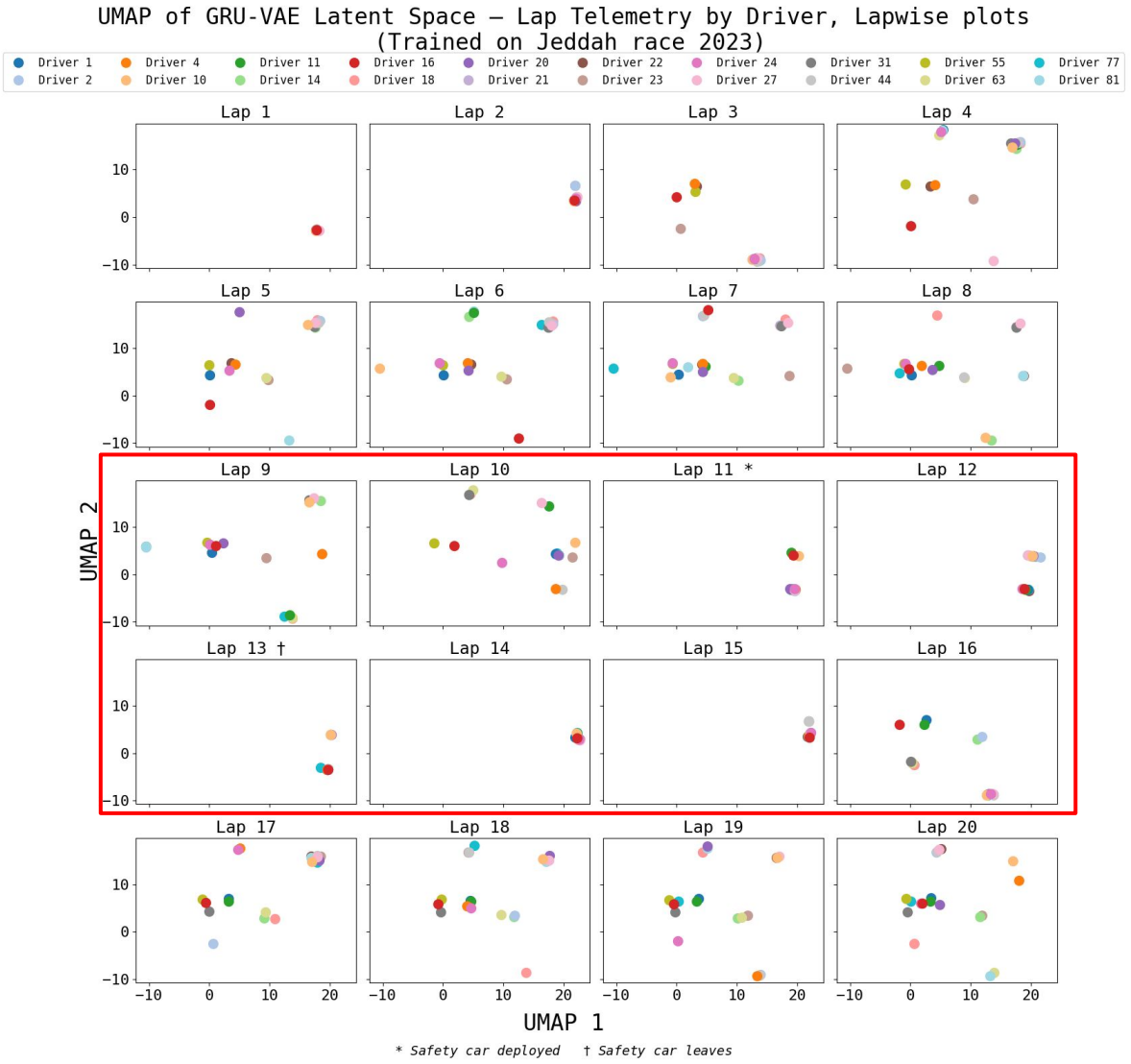
1	2	4	10	11	14	16	18	20	21	22	23	24	27	31	44	55	63	77	81
10.7	9.7	14.6	14.1	12.2	9.1	9.1	8.3	12.8	15.5	13.5	8.8	10.5	13.2	11.2	14.0	12.3	11.7	12.7	11.1
7.8	7.3	7.5	8.8	8.4	6.9	5.5	6.8	9.5	9.3	8.7	8.1	8.2	8.0	6.2	9.2	7.2	7.8	7.4	7.4

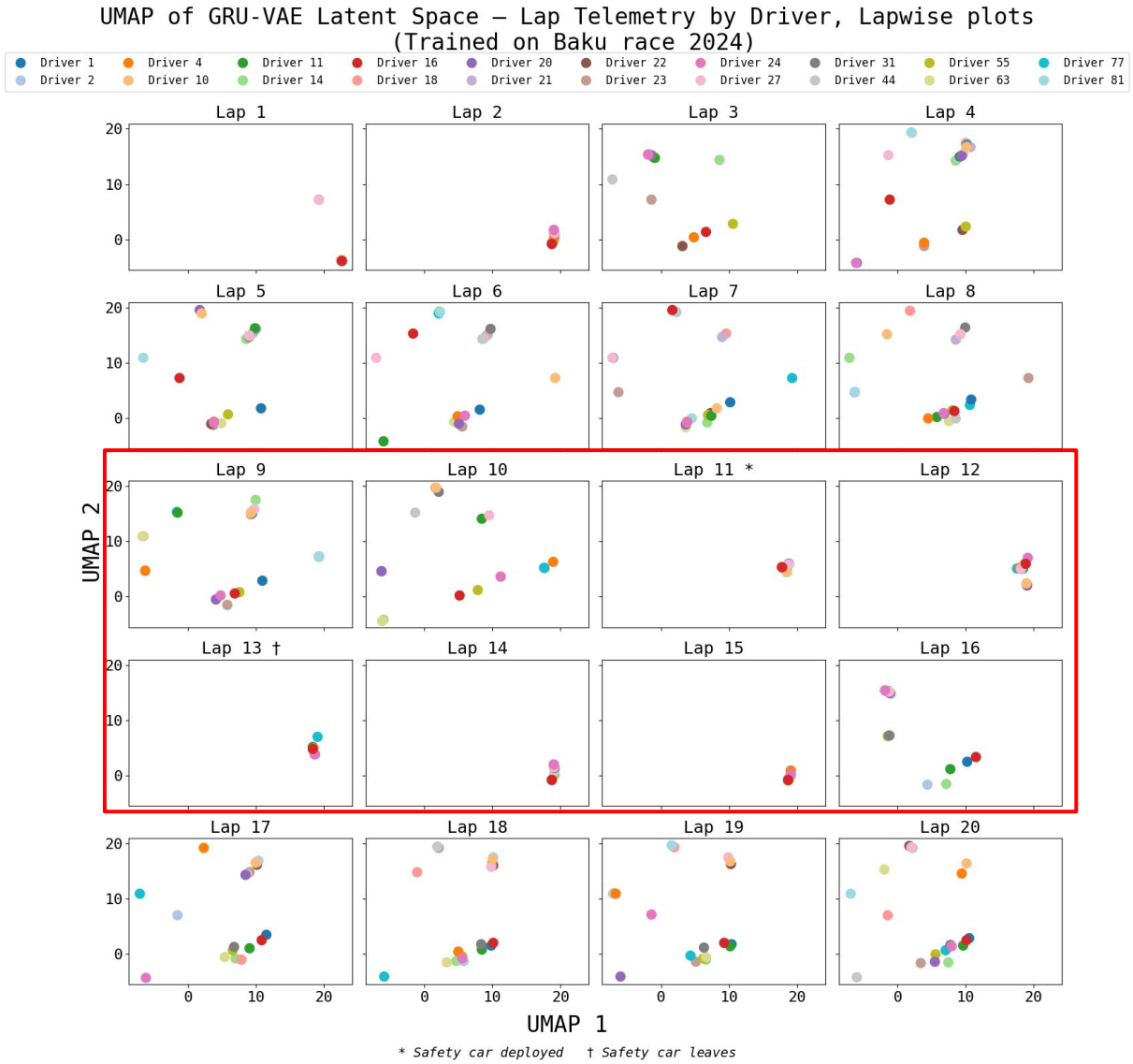
How well does the model generalize: Telemetry, order and disorder

Despite being trained on different races both of the alternate models preserve the order→disorder structure seen in the beginning of the race and the disorder→order → disorder structure when a safety car enters and leaves the circuit.

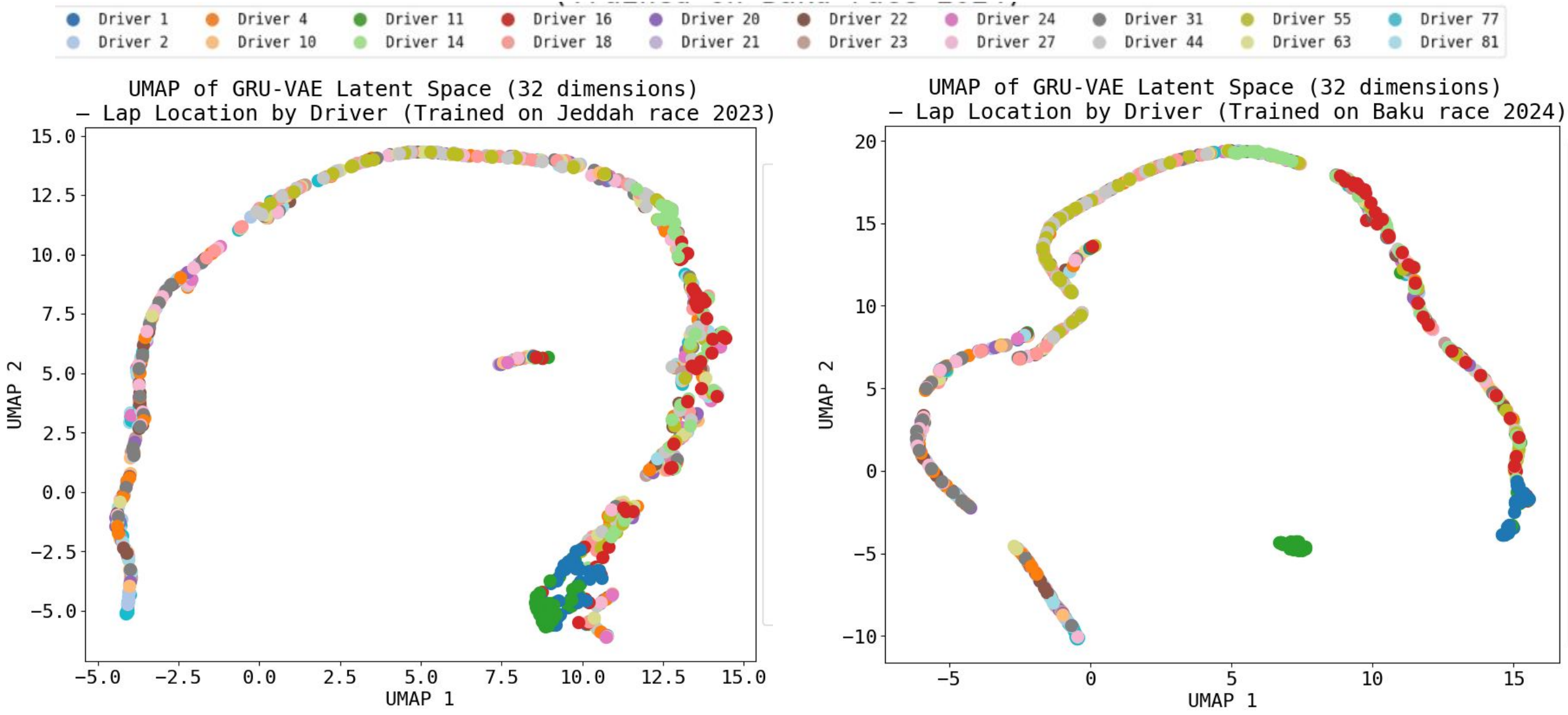
This is shown in the two plots featured in the next slide

How well does the model generalize: Telemetry, order and disorder



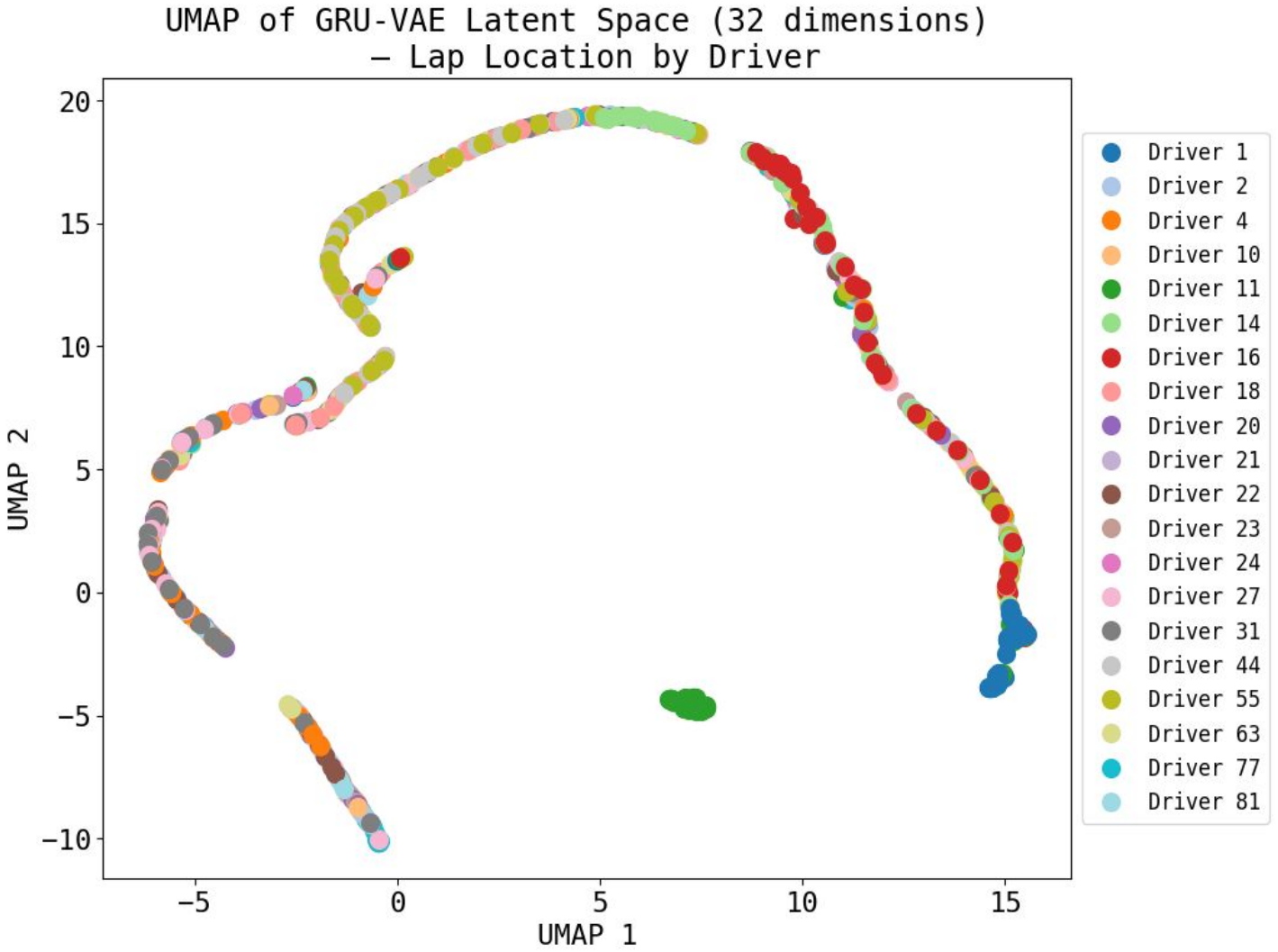


How well does the model generalize: Comparison of the two models on location



How well does the model generalize: Comparison of the two models on location

The Baku 2024 model (9598) produces identical latent vectors to the model trained on the race itself, suggesting the VAE has learned the circuit layout. The Jeddah model (7779) yields similar latent vectors overall, but driver 11 is no longer in his own cluster.

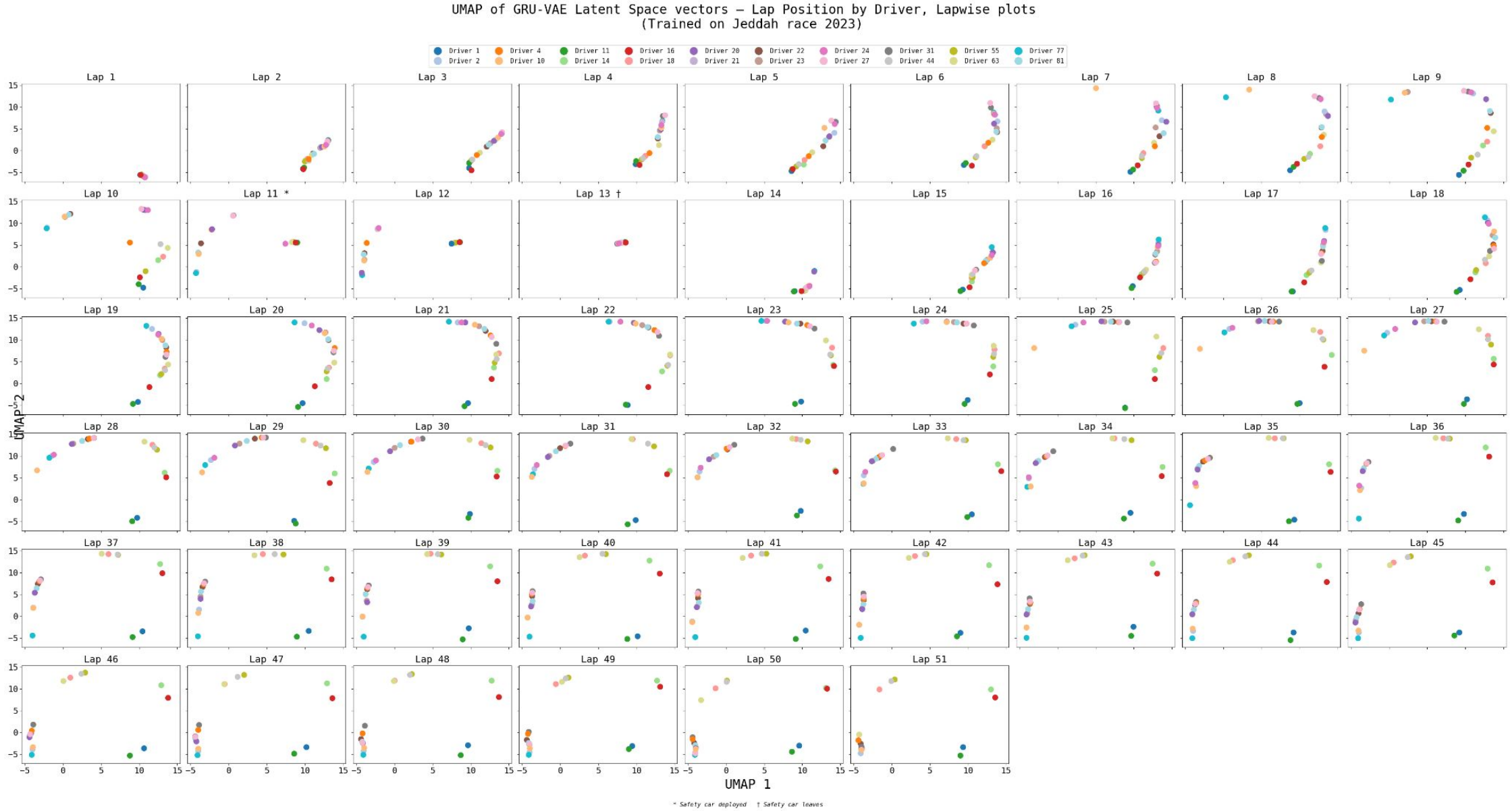


How well does the model generalize: Structured by finishing positions

The Jeddah model (7779) also preserves the structure where the drivers are primarily located in order (clockwise with the first position furthest along the clock) after their finishing position

This is shown in the the plots featured in the next slide

How well does the model generalize: Structured by finishing positions



Laptime Regression w/ VAE-GRU LVs Continued

Lap-time regression
Overview:

We aim to predict the lap time for each driver for each lap, and we want to do it for Baku 2023 and 2024. We use the VAE-GRU 32 dim latent vectors as features, and employ a very crude XGBoostRegressor.

We use a expanding window approach, i.e. we train our regression model on the first 10 laps and predict the next, and expand the next training window with the 11th latent vector. In that way, the prediction for lap *j* is based on all prior laps.

Furthermore, we want to explore if including residuals as features. So we calculate so-called deltaLVs which are calculated for each lap for each driver simply by taking the difference between the current LV and the prior. This in turn eliminates the first lap entry for all drivers.

Lap-time regression
Immediate concerns:

The 32 dim VAE-GRU LVs are interpolated from a model **trained on the first and last 10 laps of the race**. In that way, information about future laps are encoded in the latent vectors, thus leaking information.

To counteract this, we trained a new VAE-GRU model on only the first 10 laps of Baku 2023 and 2024, and then interpolated LVs for all laps. In that way when we train on the first 10 laps, the model is not fed information about the future.

Furthermore for the 2023 race there is a safety car deployed on laps 11 to 13. This would probably result in bad predictions at the start – which might bias the model.

Lap-time regression
Model:

The model is an XGBoostRegressor with the following hyperparameters:

```
{'n_estimators': 300, 'max_depth': 3, 'learning rate': 0.05, 'subsample': 0.8, 'colsample_bytree':
0.6, 'colsample_bylevel': 0.8}
```

We have not done any hyperparameter optimization. This is partly due to the fact that we want the model to adapt to each driver. We could implement an Optuna algorithm say every 5 laps, but this would be very time-consuming – and if the idea is to do on-the-fly predictions this simply wouldn't be applicable.

Lap-time regression
Data to predict on:

We test the approach in several ways.

For both 2023 and 2024, we try:

- Simple expanding window method purely on LVs
- Simple expanding window method purely on LVs and deltaLVs

For 2024 we also try to train our model on LVs from 2023, and then do the normal expanding window approach for the 2024 data.

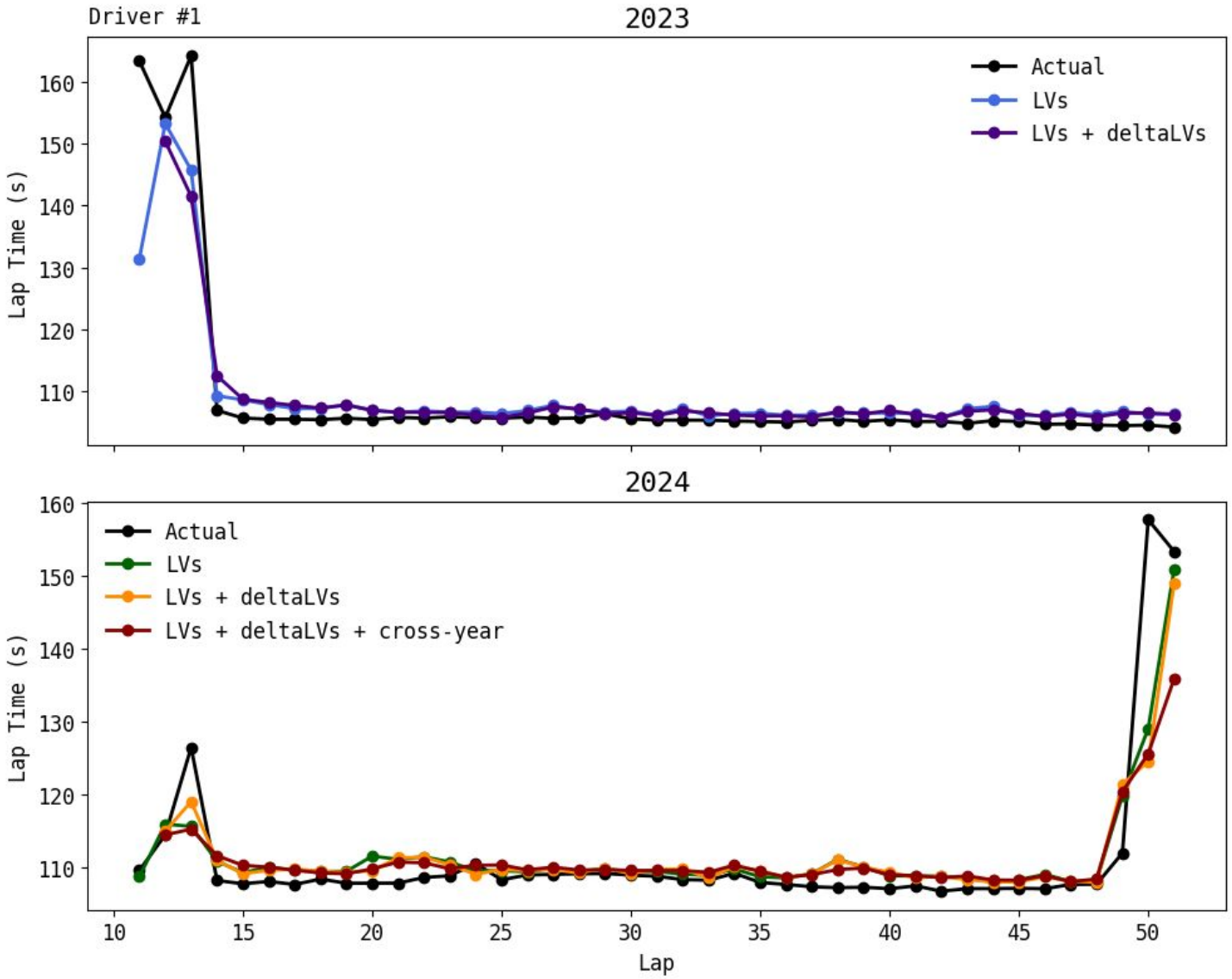
Laptime Regression w/ VAE-GRU LVs

ALL RESULTS

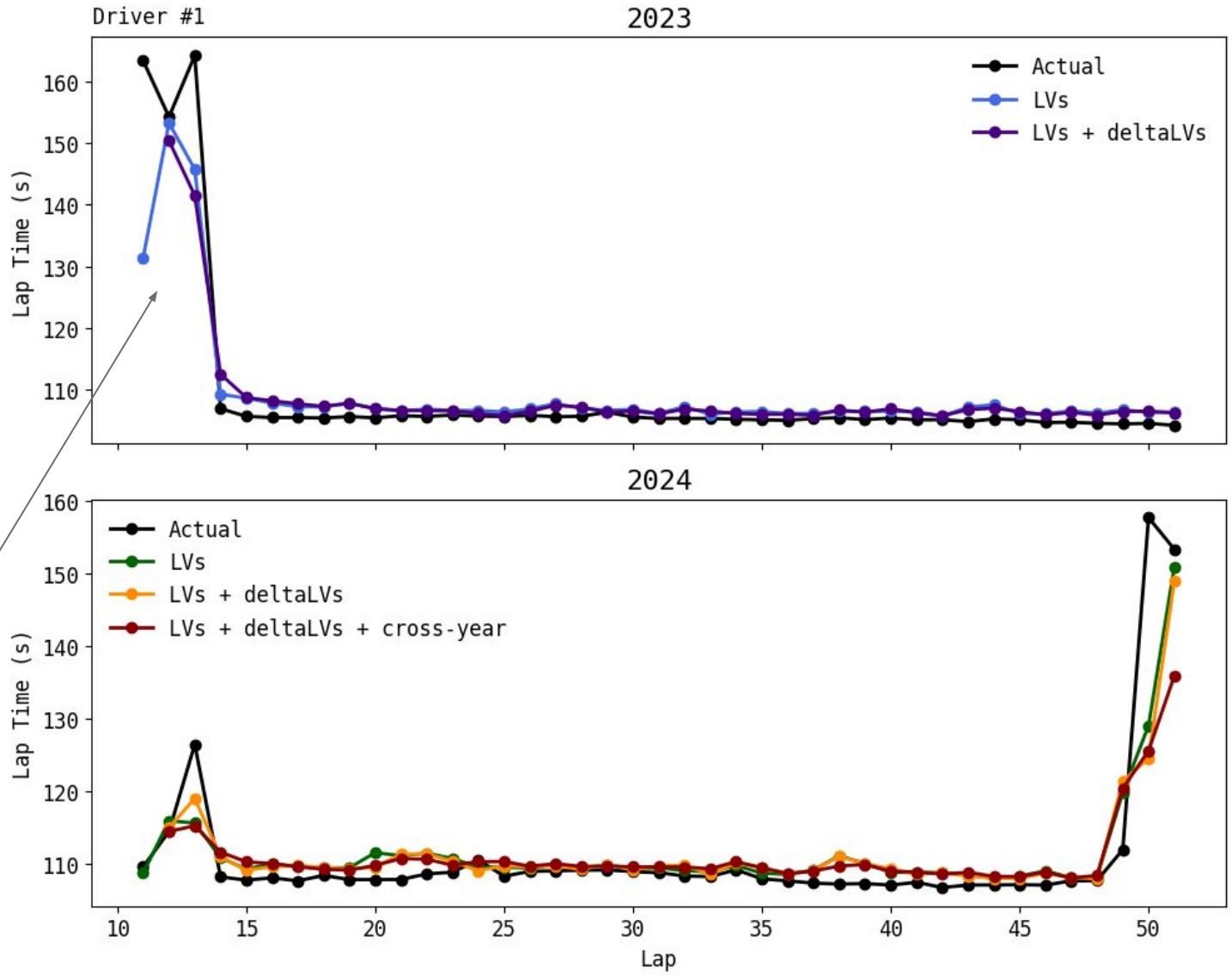
Lap-time regression
Results comment

In the follow all the results will be presented. Along the way there are notes that briefly explain any kinks or interesting things to see in the results.

Lap-time regression: Driver #1 results



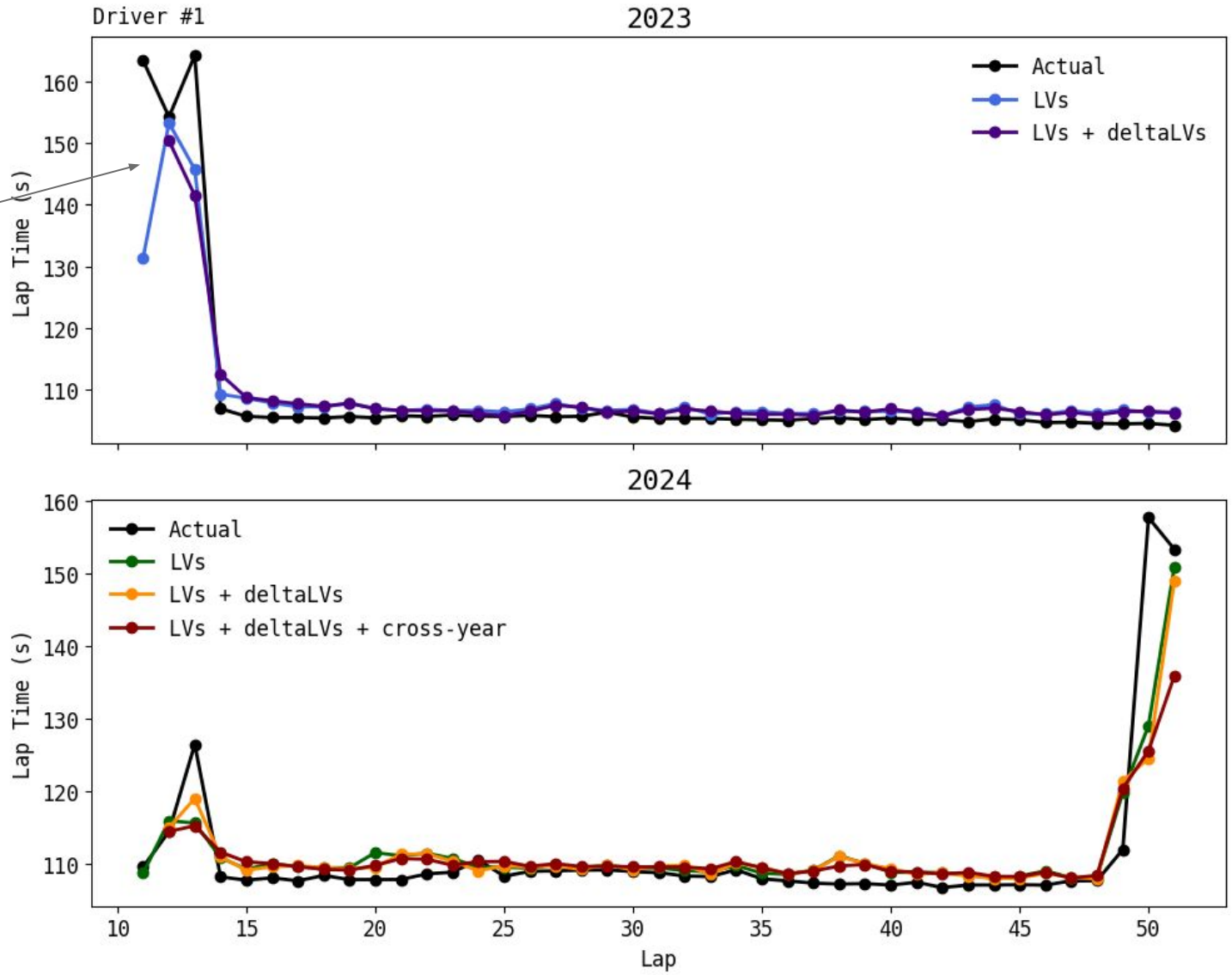
Lap-time regression: Driver #1 results



Note:
#21 crash at lap 11 resulting in a safety car for lap 11 to 13

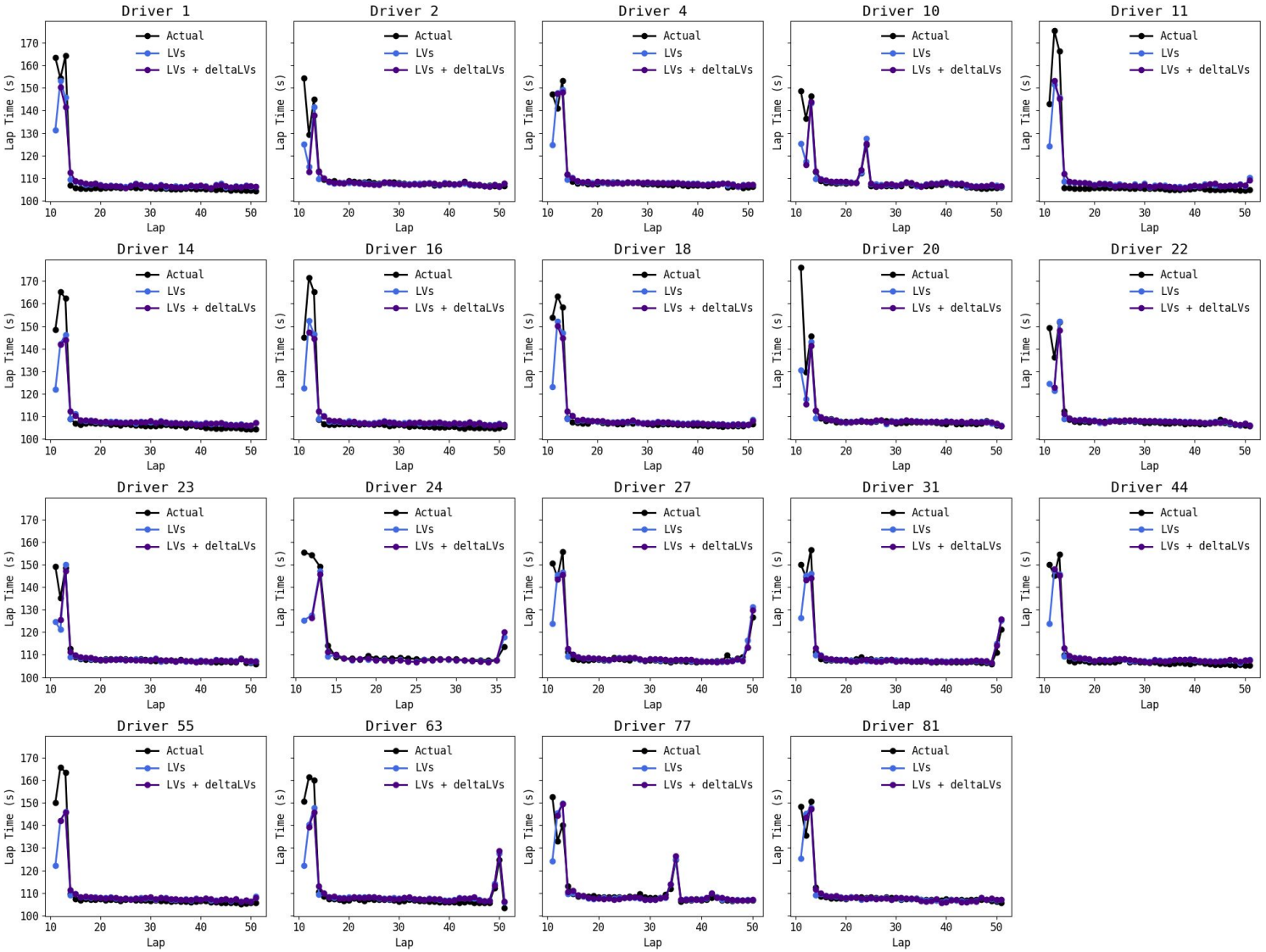
Lap-time regression: Driver #1 results

The model w/ delta LVs has lap 11 in its training window why is slightly better in the start

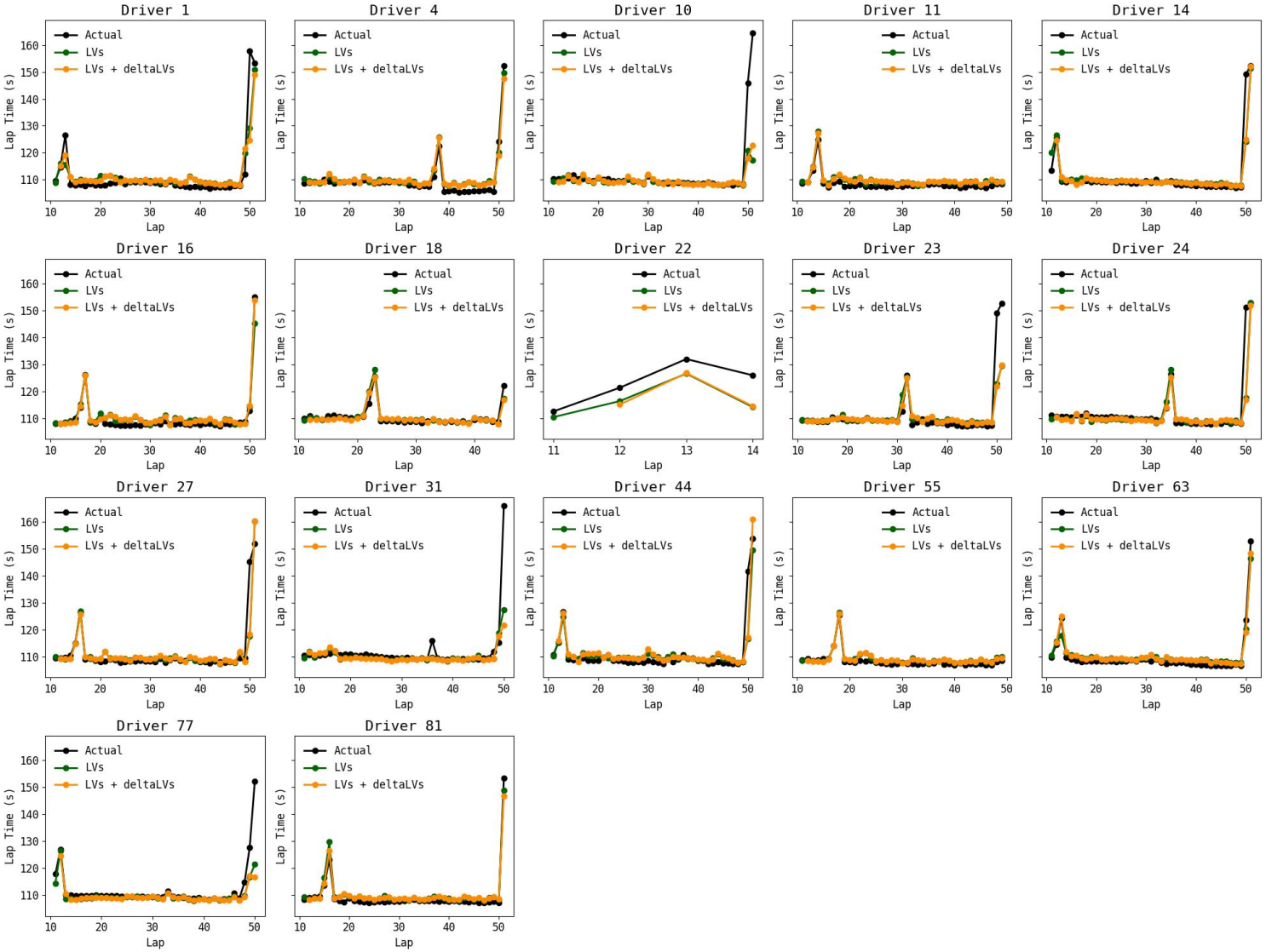


Note:
#21 crash at lap 11 resulting in a safety car for lap 11 to 13

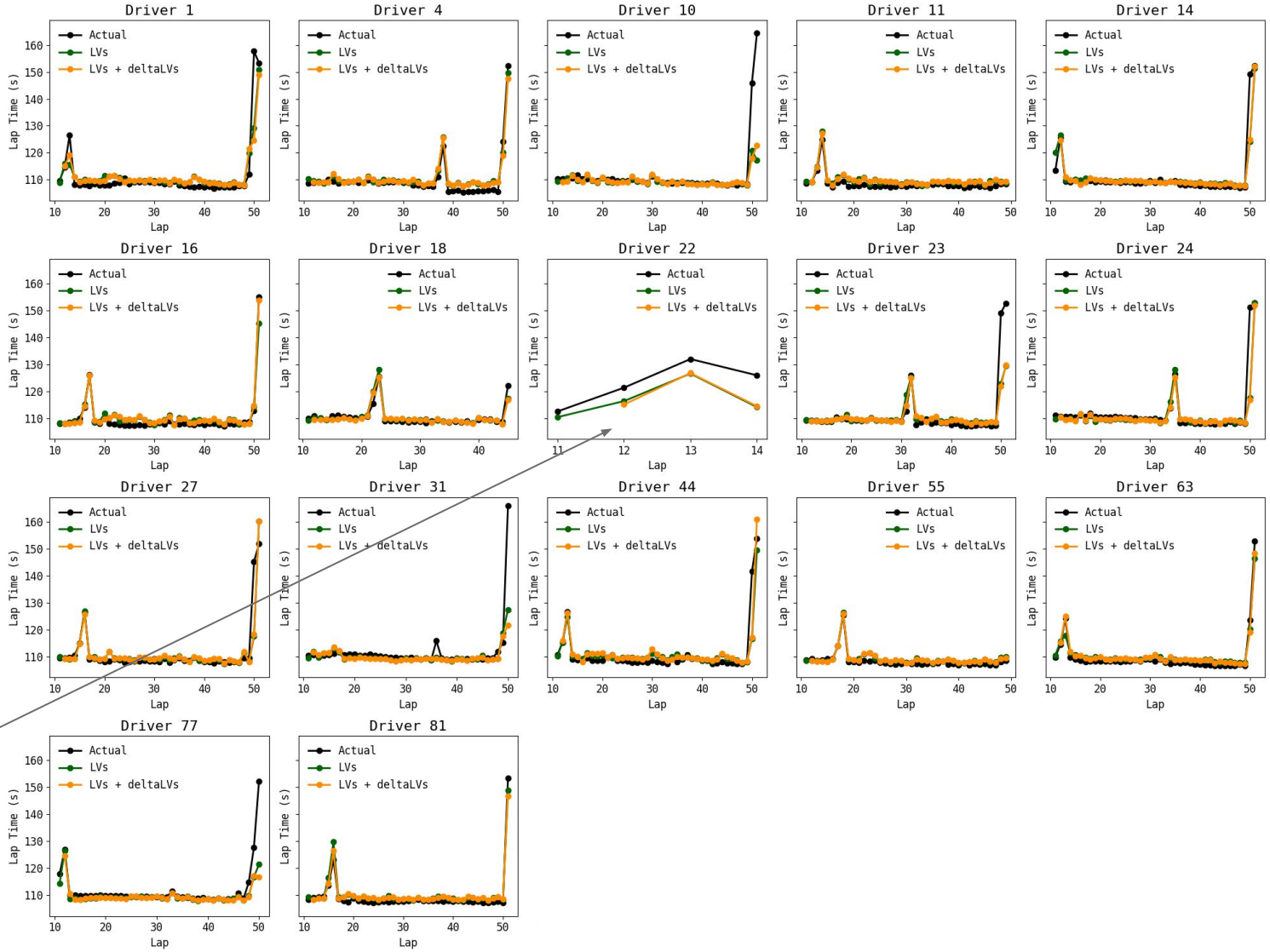
Lap-time regression: Individual Driver Results 2023



Lap-time regression: Individual Driver Results 2024

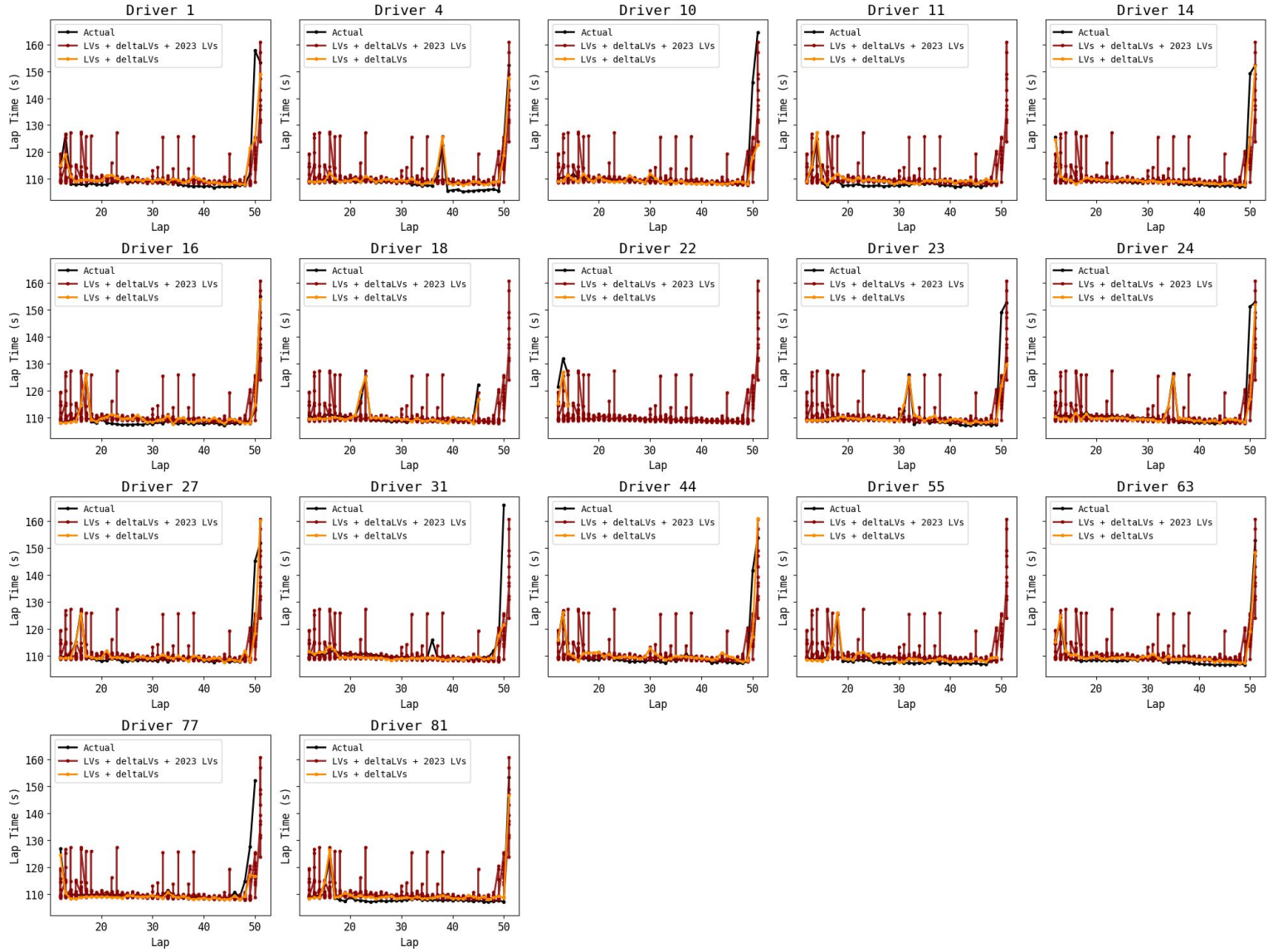


Lap-time regression: Individual Driver Results 2024



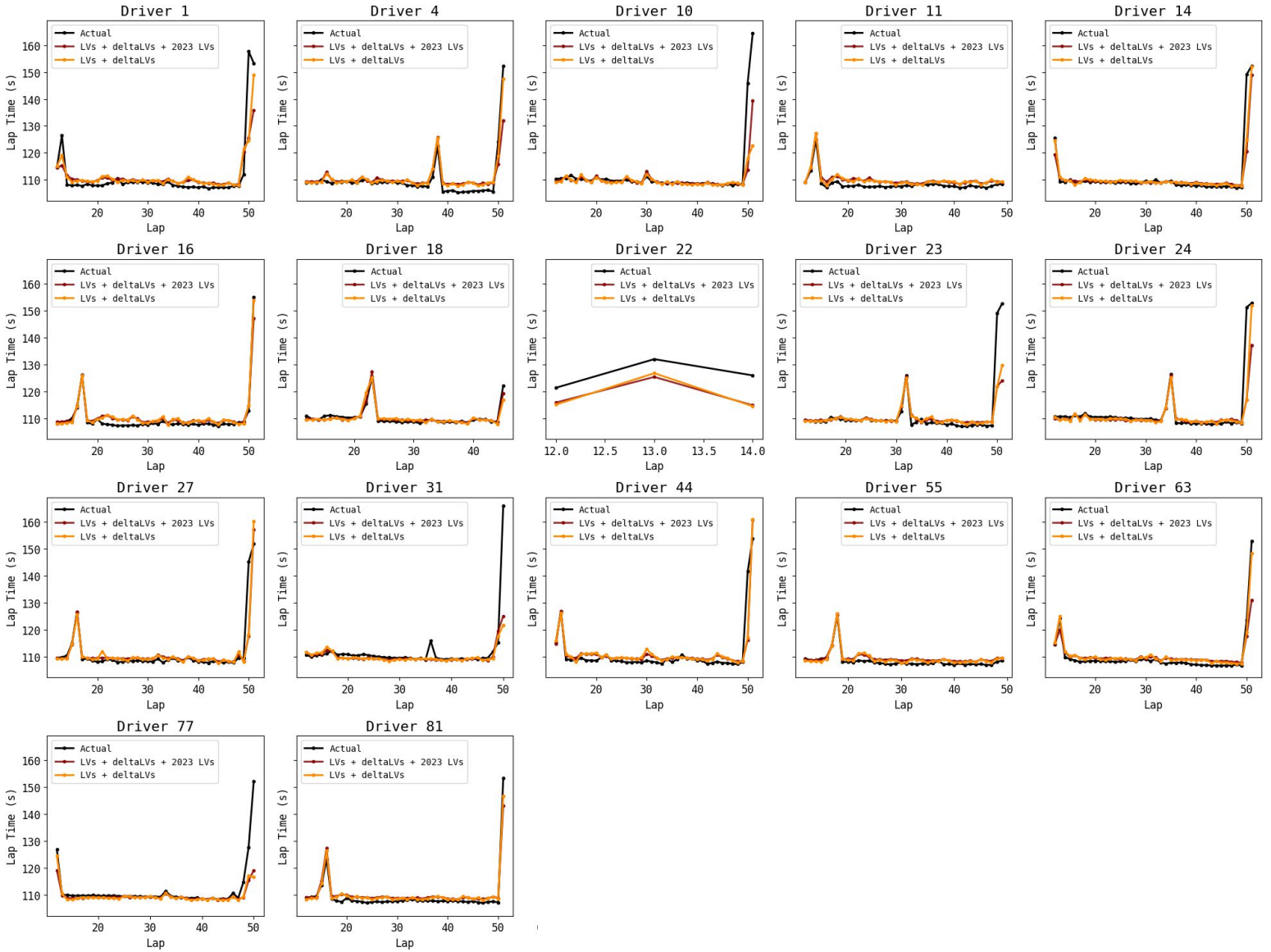
Note:
#22
stopped at
lap 14

Lap-time regression: Individual Driver Results 2024 w/2023 LVs

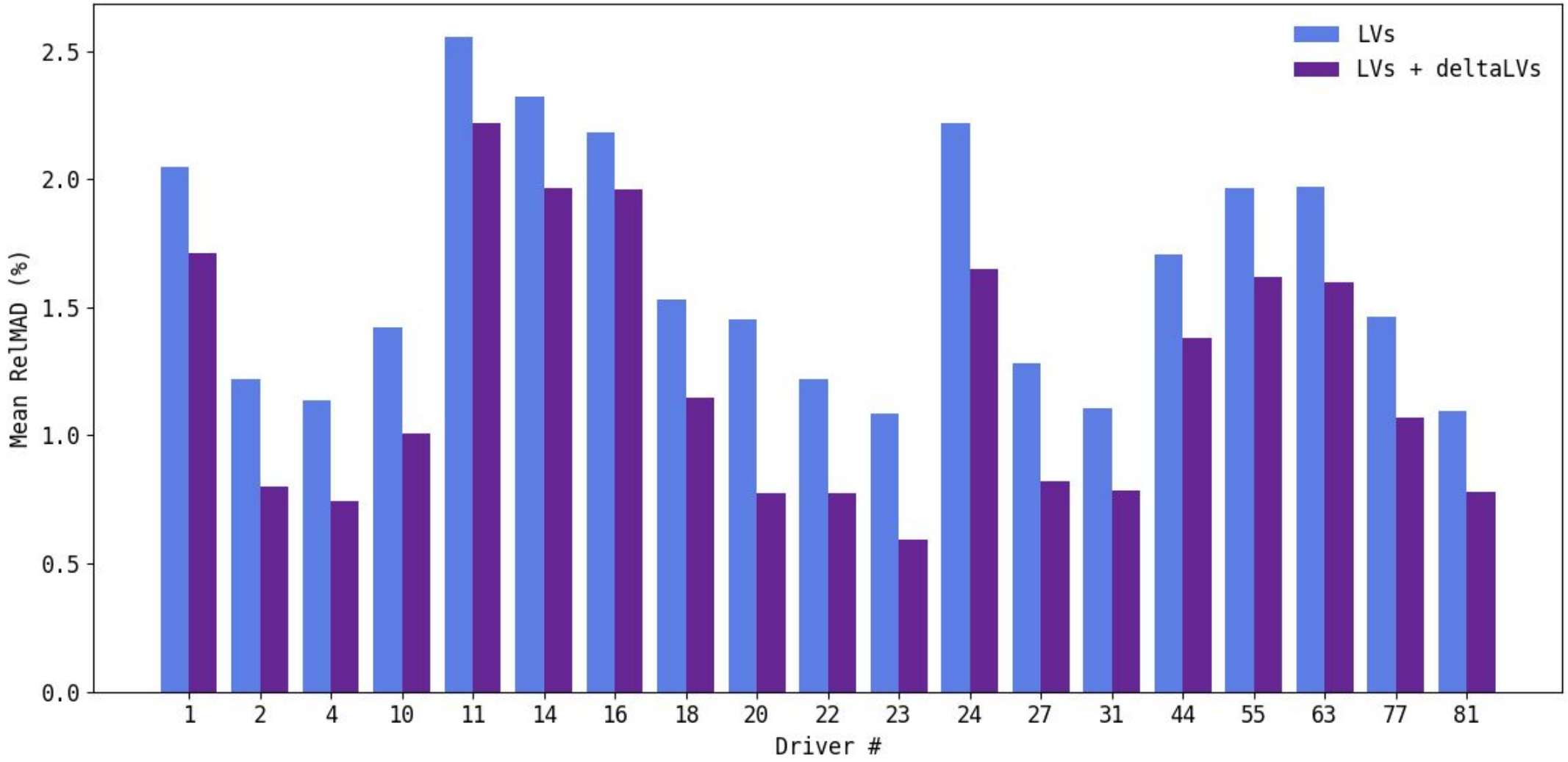


Lap-time regression: Individual Driver Results 2024 w/2023 LVs

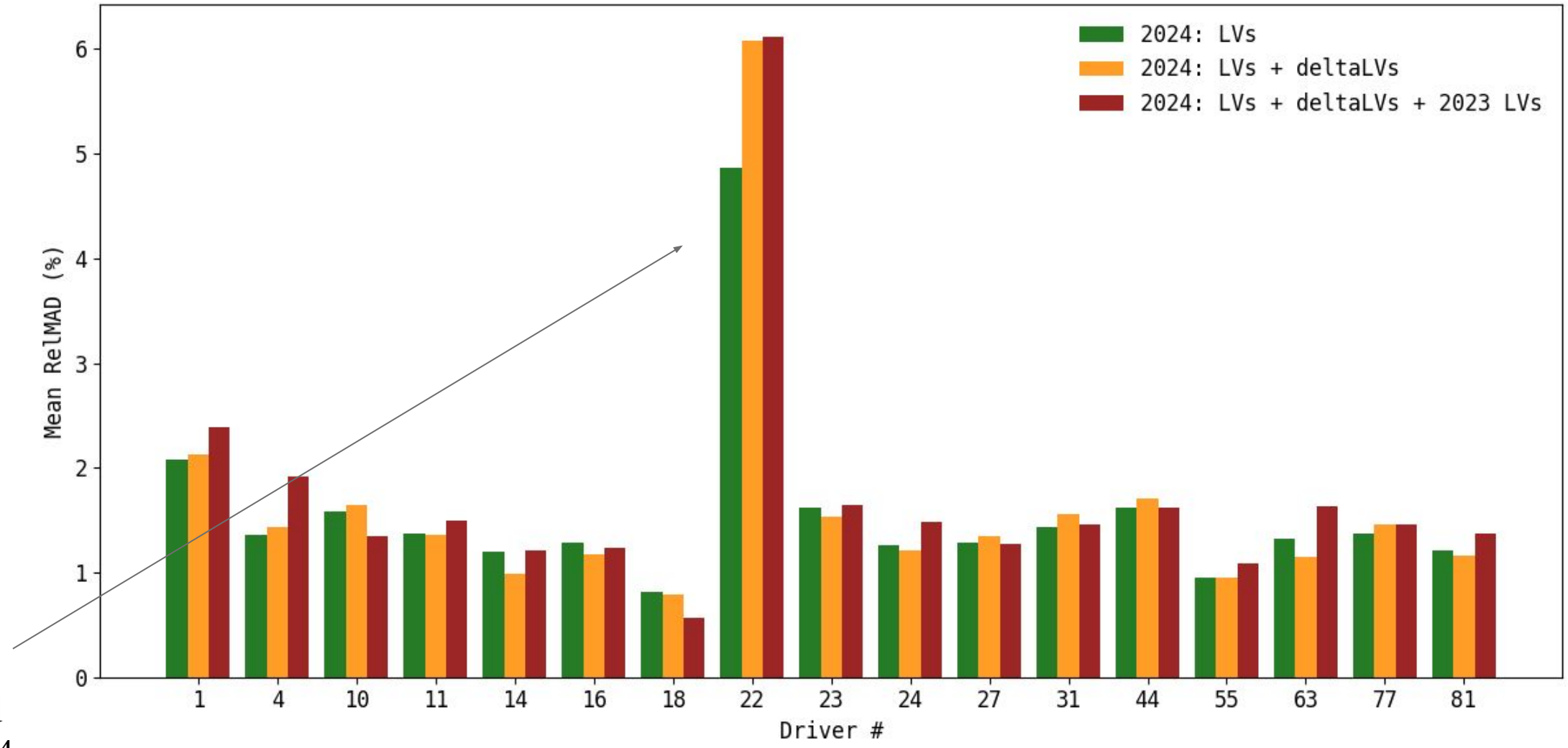
Note:
Using 2023 LVs does not improve the model. Might even bias to events prior year…



Lap-time regression: RelMAD Results 2023



Lap-time Regression: RelMAD Results 2024



Note: #22 stopped at lap 14

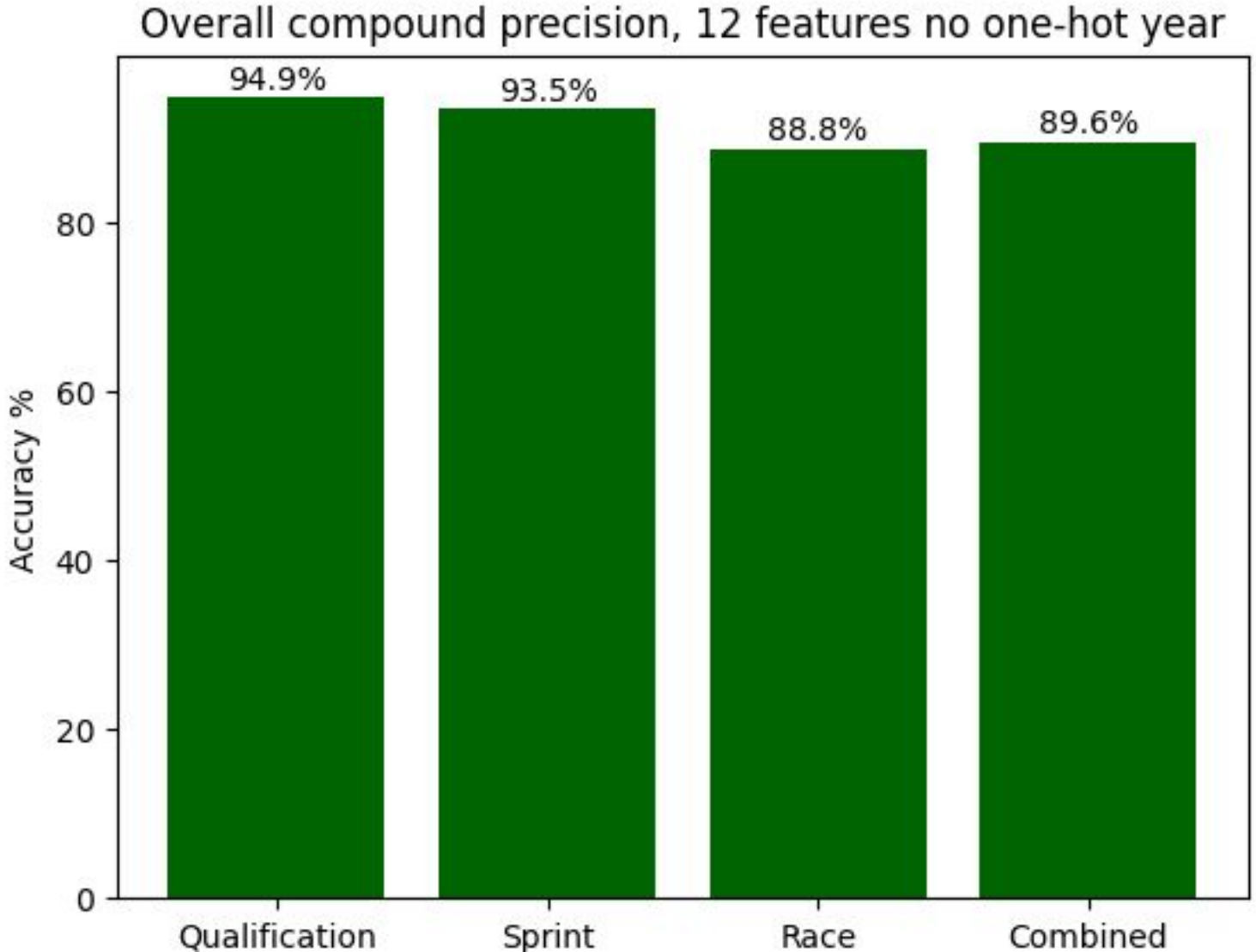
Compounds

Classification of unknown compounds

- 29 laps have unknown compounds
- Model 1 predicts the compound for the driver during each of the 29 laps as soft
- Model 3 predicts one of them as medium and the rest as soft
- Overall average score improvement from 0.932 to 0.999 for the predicted compound

DT results without One-hotting year & Type

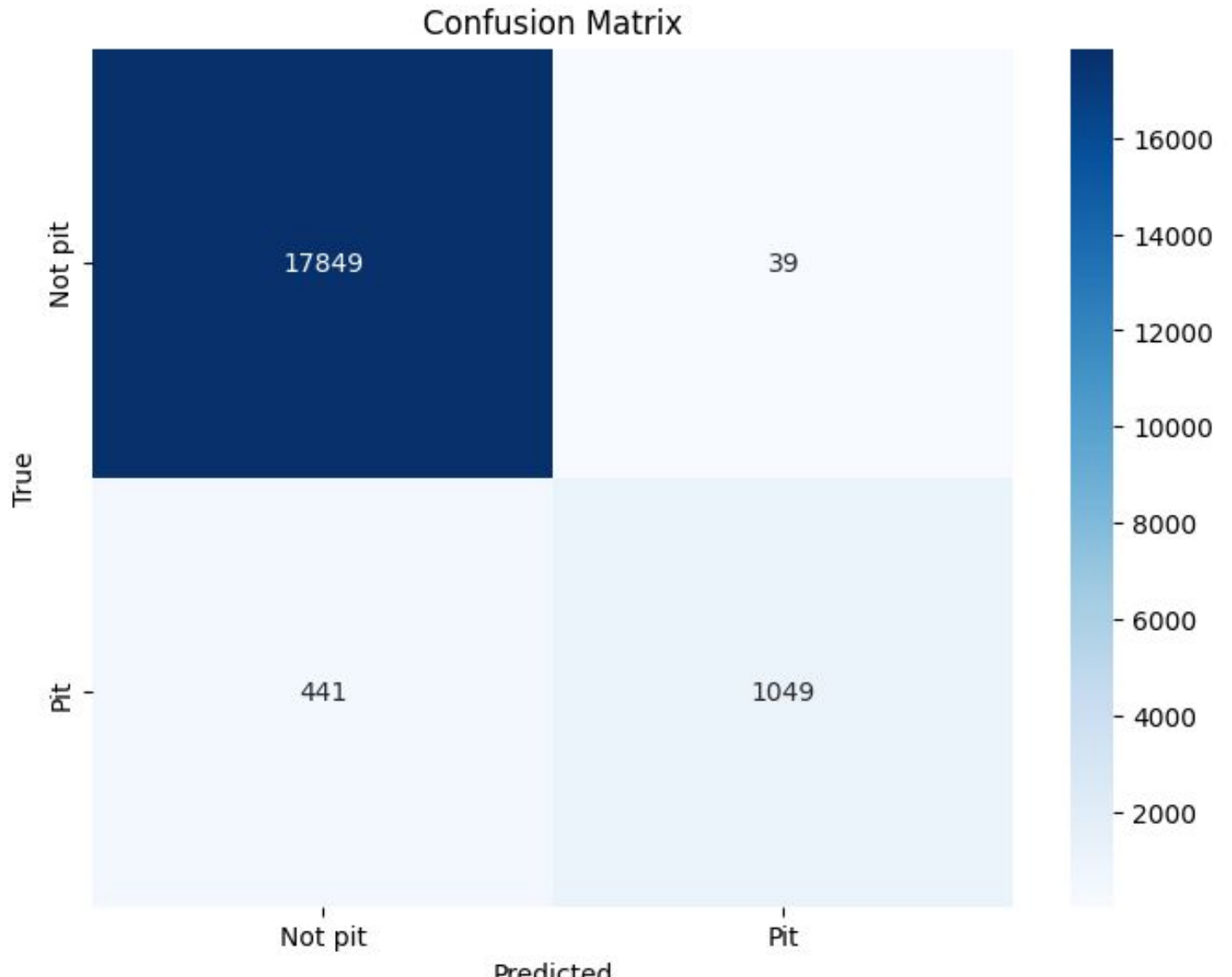
While a Boosted DT wouldn't typically need to be one-hotted, it gives a slight uneven increase in precision



Prediction on pitting

Pit prediction

- Uneven data, hard to predict
- Model primarily guessed
- Attempted weighting, did not work
- Not enough time to fix



Features

Features: **Category** and features

Car_telemetry
- Brake
- Drs
- n_gear
- rpm
- speed
- throttle

Position
- position

Location
- x
- y
- z

Intervals
- gap_to_leader
- interval

Weather
- air_temperature
- humidity
- pressure
- rainfall
- tracktemperature
- winddirection
- windspeed

Laps
- duration_sector_1
- duration_sector_2
- duration_sector_3
- i1_speed
- i2_speed
- is_pit_out_lap
- lap_duration
- lap_number
- st_speed

Stints
- compound
- lap_end
- lap_start
- stint-Number
- tyre_age_at_start