UNIVERSITÀ DEGLI STUDI DI BERGAMO
Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Formula 1 Lap Time Prediction

Optimization Project

Optimization 2024/2025

PRESENTER
Daniele Gotti - 1079011

LOCATION
Dalmine, BG

Contents

This project was developed in a **Jupyter Notebook** (*.ipynb*) using **VS Code** and **Python** within an **Anaconda** virtual environment.

The full project is available on **GitHub**: link.

What is Formula 1?

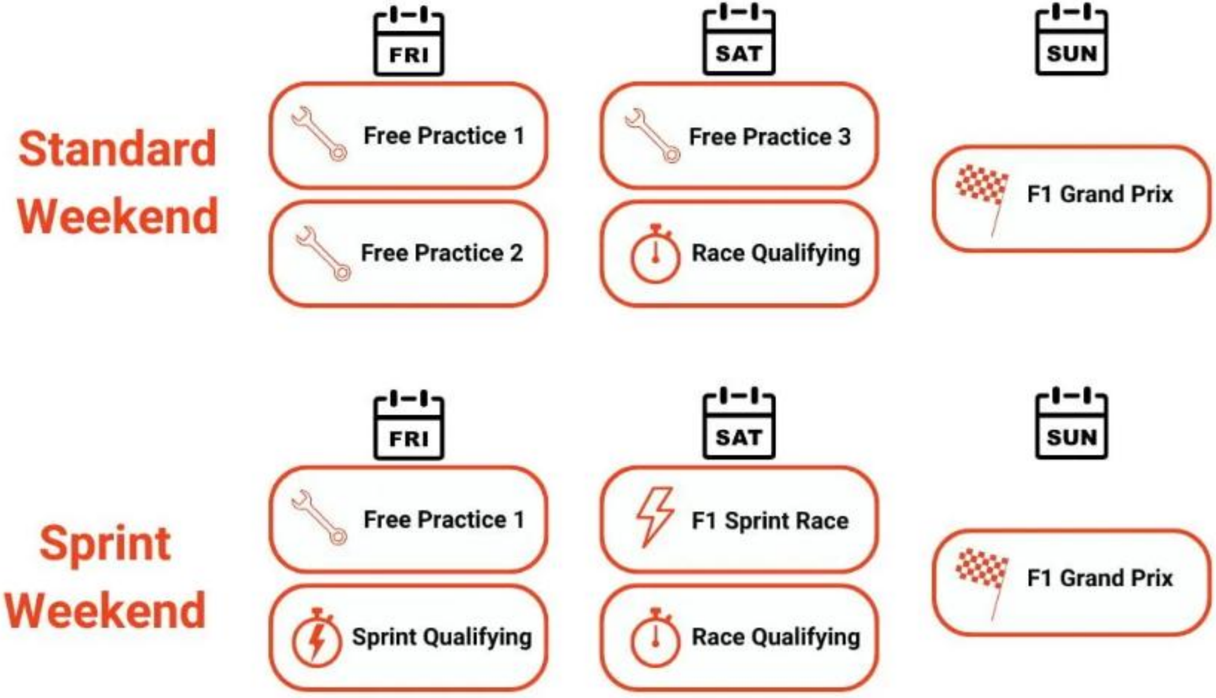
What is Formula 1?

Formula 1 (F1) is the highest class of international **motorsport**.

- It features open-wheel, single-seater race cars competing in a series of **Grand Prix** events worldwide.
- Each Grand Prix (24 GPs in 2025) is held on a permanent or street circuit in different countries and the race weekend usually spans **three days**.

Source: sportingnews.com

Race weekend



Source: fanamp.com

Teams, drivers and championships

- Each team (constructor) fields **two cars** and employs **two drivers**.
- **Points** are awarded based on finishing position. The driver and team with the most points at season's end are crowned champions.
- Teams compete in the **Constructors' Championship**, and drivers in the **Drivers' Championship**.

Source: sportingnews.com

Source: motorauthority.com

Cars

- F1 cars are technological masterpieces, with hybrid power units combining a turbocharged V6 engine and energy recovery systems.

- Made with carbon fiber, they're extremely light and fast: over **350 km/h top speed** and **0 - 100 km/h in ~2.5 seconds**.

Source: motorionline.com

- Each car is designed and built by the teams within strict FIA regulations.

Why is Formula 1 fascinating?

- Involves the **best of engineering**: aerodynamics, materials science, thermal management, mechanical design, information and electronic systems, etc.

- Drivers undergo intense **physical and mental training**, enduring G-forces, high temperatures, and long concentration spans.

- Anything can happen during a race, which is why **strategy** plays a crucial role (pit stops, accidents, etc.).

- It’s a sport where even **hundredths of a second can make the difference**.

Why data is crucial in F1?

- Each car has over **300 sensors** collecting real-time data generating **1.1 million telemetry data points per second**.

- Data is used to:
  - **optimize car** setup for each track.
  - **monitor performance** and reliability.
  - run simulations to **test strategies** .
  - train drivers using **advanced simulators**.

Source: formule1fr.com

- Data informs **race strategies**: data scientists make fast decisions during the race.

- **Machine learning** and **AI** are increasingly used to model race outcomes and competitor behavior.

UNIVERSITÀ DEGLI STUDI DI BERGAMO | Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Project Goals

Source: f1friend.com

Project goals

- Develop **linear regression** models to **predict lap times** based on multiple variables
- Implement **optimization techniques** derived from **gradient descent** methods to minimize prediction error
- Analyze model performance and interpret the **influence of different factors** on lap times

Why predict lap times?

- **Driver benchmarking**: provide realistic targets based on car and track conditions.
- **Simulator Training**: set lap time goals for driver development.
- **AI Calibration (Games)**: tune AI lap times to match real-world scenarios.
- **Car Development**: evaluate design changes through simulated performance.
- **Race Strategy**: support tire and fuel decisions with predicted pace.
- **Fan Engagement**: Enhance broadcasts with expected lap time insights.

UNIVERSITÀ DEGLI STUDI DI BERGAMO

Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Data Preparation

Custom Dataset Construction

- Pre-existing datasets on **Kaggle** or **UCI** were not suitable for my specific goal of predicting lap times under varying real-world conditions.

- Instead, I created a custom dataset using the approach from a Kaggle notebook (*Data Exploration with OCI*), which leverages the **FastF1 API** (a F1 data service) to collect telemetry and race data.

- The resulting dataset includes detailed lap-by-lap information from the **2023 and 2024** Formula 1 seasons.

References:

- Kaggle notebook *Data Exploration with OCI*: link.
- Code for dataset creation with FastF1 API: link.
- FastF1 API documentation: link.

Source: docs.fastf1.dev

Original dataset

- The result obtained using the API is a dataset (*dataset_original.csv*) containing **46 columns** and **121534 rows**. At first glance, many of these columns appear to be irrelevant.

```
Index(['Time', 'Driver', 'DriverNumber', 'LapTime', 'LapNumber', 'Stint',
       'PitOutTime', 'PitInTime', 'Sector1Time', 'Sector2Time', 'Sector3Time',
       'Sector1SessionTime', 'Sector2SessionTime', 'Sector3SessionTime',
       'SpeedI1', 'SpeedI2', 'SpeedFL', 'SpeedST', 'IsPersonalBest',
       'Compound', 'TyreLife', 'FreshTyre', 'Team', 'LapStartTime',
       'LapStartDate', 'TrackStatus', 'Position', 'Deleted', 'DeletedReason',
       'FastF1Generated', 'IsAccurate', 'RoundNumber', 'EventName', 'country',
       'session', 'EventDate', 'eventYear', 'original_index', 'TimeWeather',
       'AirTemp', 'Humidity', 'Pressure', 'Rainfall', 'TrackTemp',
       'WindDirection', 'WindSpeed'],
      dtype='object')
```

- The code for the following operations is contained in the file *1_Dataset_Preparation.ipynb*.

Editing rows (1/2)

- Removed rows where *IsAccurate* is false, as they do not meet the quality criteria for valid lap data (inlap or outlap, flags, safetycar, etc.).

- Removed rows in which all of the following fields were null: *Stint*, *Compound,* and *TyreLife*.

- Excluded drivers with minimal or unofficial participation.

- Removed laps recorded under yellow flag conditions (i.e., where *TrackStatus* ≠ 1).

- Excluded free practice sessions (FP1, FP2, FP3).

Editing rows (2/2)

- Removed warm-up laps in qualifying sessions, retaining only laps within 5 seconds of the driver's best time in the same session.

- Standardized team names (Alfa Romeo → Kick Sauber, AlphaTauri → RB).

- Removed unofficial tyre compounds like test compound.

- Converted *LapTime* to seconds.

Editing columns

- Dropped unnecessary columns at the beginning: *DriverNumber, Sector1Time, Sector2Time, Sector3Time, Sector1SessionTime, Sector2SessionTime, Sector3SessionTime, SpeedI1, SpeedI2, SpeedFL, SpeedST, IsPersonalBest, LapStartTime, LapStartDate, Position, DeletedReason, FastF1Generated, Country*.

- Added a new column *FuelLevel* based on estimated fuel consumption per lap.

- Dropped additional columns after filtering and row operations: *IsAccurate, PitOutTime, PitInTime, TrackStatus, Deleted, EventDate, Time, Stint, RoundNumber, TimeWeather, original_index.*

Final dataset

- The new data have been exported to the file *dataset_final.csv*.

- Changes include:
  - columns: 46 → **18**;
  - rows: 121534 → **52160**.

	Driver	LapTime	LapNumber	Compound	TyreLife	FreshTyre	Team	EventName	Session	EventYear	AirTemp	Humidity	Pressure	Rainfall	TrackTemp	WindDirection	WindSpeed	FuelLevel
0	VER	91.295	3.0	SOFT	3.0	False	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	24.2	16.0	1017.5	False	28.7	352	0.6	2.0
1	VER	90.503	8.0	SOFT	2.0	True	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.8	20.0	1017.7	False	27.8	342	0.5	2.0
2	VER	89.897	11.0	SOFT	2.0	True	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.1	31.0	1017.7	False	26.9	338	1.3	2.0
3	VER	89.708	14.0	SOFT	2.0	False	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.0	33.0	1017.9	False	26.6	329	0.8	2.0
4	PER	91.479	3.0	SOFT	3.0	False	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	24.3	16.0	1017.5	False	28.7	333	0.4	2.0
5	PER	90.746	8.0	SOFT	2.0	True	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.8	20.0	1017.7	False	27.8	342	0.5	2.0
6	PER	90.131	11.0	SOFT	2.0	True	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.1	31.0	1017.7	False	26.9	338	1.3	2.0
7	PER	89.846	14.0	SOFT	2.0	False	Red Bull Racing	Bahrain Grand Prix	Qualifying	2023	23.0	33.0	1017.9	False	26.6	329	0.8	2.0
8	LEC	91.094	5.0	SOFT	2.0	True	Ferrari	Bahrain Grand Prix	Qualifying	2023	24.2	16.0	1017.5	False	28.7	352	0.6	2.0
9	LEC	91.699	10.0	SOFT	7.0	False	Ferrari	Bahrain Grand Prix	Qualifying	2023	23.8	20.0	1017.7	False	27.8	342	0.5	2.0

UNIVERSITÀ DEGLI STUDI DI BERGAMO

Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Model Creation

Analysis of correlations between variables

- **Numerical** vs **numerical** variable: **Pearson correlation coefficient** (from -1 to 1)

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2}\sqrt{\sum(y_i - \bar{y})^2}}$$

- **Categorical** vs **numerical** variable: **Eta squared** (from 0 to 1)

$$\eta^2 = \frac{SS_{\text{between}}}{SS_{\text{total}}} = \frac{\sum_g n_g(\bar{y}_g - \bar{y})^2}{\sum_i (y_i - \bar{y})^2}$$

g = group,
n = number of observation in g.

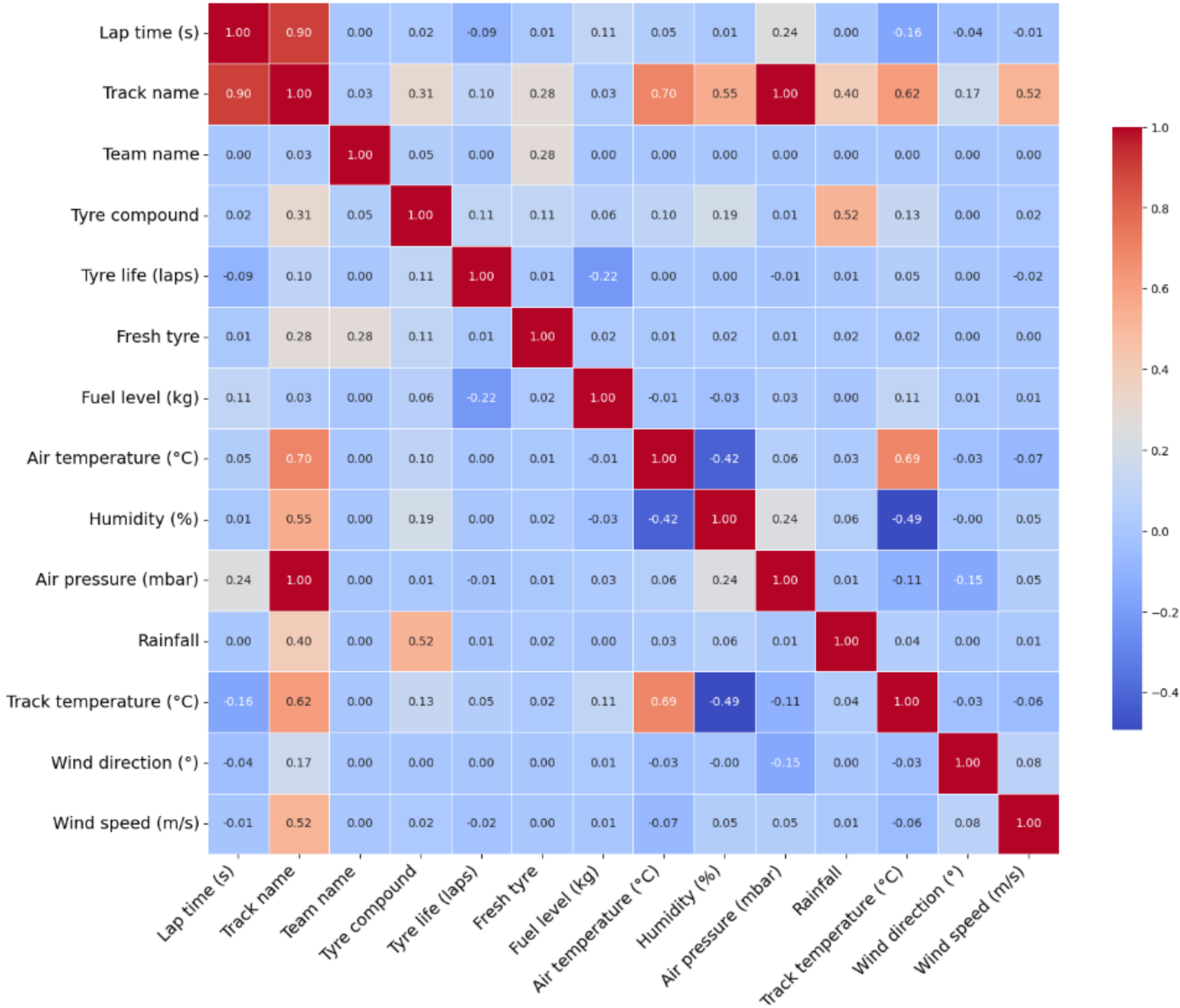
- **Categorical** vs **Categorical** variable: **Cramér V** (from 0 to 1)

$$V = \sqrt{\frac{\chi^2/n}{\min(k-1, r-1)}}$$

X^2 = Chi-squared statistic;
n = number of observation;
k = categories of one variable;
r = categories of the other variable.

Correlation matrix

- Lap times are highly correlated with the circuit.
- *Air pressure* is excessively correlated with the circuit, while variables such as *Air temperature* and *Humidity* convey overlapping information.
- Therefore, *Air pressure, Air temperature,* and *Humidity* will not be included in the model.



Model features

y = **Lap time (s)**

	Variable name	Values
0	**Track name**	Abu Dhabi Grand Prix, Australian Grand Prix, Austrian Grand Prix, Azerbaijan Grand Prix, Bahrain Grand Prix Belgian Grand Prix, British Grand Prix, Canadian Grand Prix, Chinese Grand Prix, Dutch Grand Prix Emilia Romagna Grand Prix, Hungarian Grand Prix, Italian Grand Prix, Japanese Grand Prix, Las Vegas Grand Prix Mexico City Grand Prix, Miami Grand Prix, Monaco Grand Prix, Qatar Grand Prix, Saudi Arabian Grand Prix Singapore Grand Prix, Spanish Grand Prix, São Paulo Grand Prix, United States Grand Prix
1	**Team name**	Alpine, Aston Martin, Ferrari, Haas F1 Team, Kick Sauber, McLaren, Mercedes, RB, Red Bull Racing, Williams
2	**Compound**	HARD, INTERMEDIATE, MEDIUM, SOFT, WET
3	**Fresh tyre**	False, True
4	**Rainfall**	False, True
5	**Tyre life (laps)**	/
6	**Track temperature (°C)**	/
7	**Wind direction (°)**	/
8	**Wind speed (m/s)**	/
9	**Fuel level (kg)**	/

Model creation

- Variables are standardized to have **mean 0** and **standard deviation 1**.
- The dataset is split **50% into training and testing** sets (kept fixed throughout the optimization process to compare the results of different optimization methods).
- Categorical variables are converted into dummies using **one-hot encoding**, with the first category of each variable set as the **baseline**.

```
Categorical variables (with dropped baseline):
'Track name': baseline = 'Abu Dhabi Grand Prix'
'Team name': baseline = 'Alpine'
'Tyre compound': baseline = 'HARD'
'Fresh tyre': baseline = 'False'
'Rainfall': baseline = 'False'
```

Linear regression

- **Loss function** (objective): **MSE** (Mean Squared Error)

$$L(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 = \frac{1}{2n} \|X\mathbf{w} - \mathbf{y}\|^2$$

- **Gradient**

$$\nabla L(\mathbf{w}) = \frac{1}{n} X^\top (X\mathbf{w} - \mathbf{y})$$

- **Hessian**

$$\nabla^2 L(\mathbf{w}) = \frac{1}{n} X^\top X$$

UNIVERSITÀ
DEGLI STUDI
DI BERGAMO
Dipartimento
di Ingegneria Gestionale,
dell'Informazione e della Produzione

Optimization

Gradient Descent (basic)

$$x_{t+1} := x_t - \gamma \nabla f(x_t)$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *gamma* = 0.6.

```
[TRAIN] Gradient Descent:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.45285 | ||grad|| = 2.19e-01
[STEP    3] f(x) = 0.42587 | ||grad|| = 1.97e-01
[STEP    4] f(x) = 0.40318 | ||grad|| = 1.88e-01
[STEP    5] f(x) = 0.38245 | ||grad|| = 1.81e-01
[STEP    6] f(x) = 0.36318 | ||grad|| = 1.75e-01
[STEP    7] f(x) = 0.34517 | ||grad|| = 1.69e-01
[STEP    8] f(x) = 0.32831 | ||grad|| = 1.64e-01
[STEP    9] f(x) = 0.31249 | ||grad|| = 1.59e-01
[STEP   10] f(x) = 0.29765 | ||grad|| = 1.54e-01
[STEP   11] f(x) = 0.28370 | ||grad|| = 1.49e-01
[STEP   12] f(x) = 0.27058 | ||grad|| = 1.45e-01
[STEP   13] f(x) = 0.25823 | ||grad|| = 1.40e-01
[STEP   14] f(x) = 0.24660 | ||grad|| = 1.36e-01
[STEP   15] f(x) = 0.23563 | ||grad|| = 1.32e-01
[STEP   16] f(x) = 0.22529 | ||grad|| = 1.28e-01
[STEP   17] f(x) = 0.21553 | ||grad|| = 1.25e-01
[STEP   18] f(x) = 0.20632 | ||grad|| = 1.21e-01
[STEP   19] f(x) = 0.19761 | ||grad|| = 1.18e-01
[STEP   20] f(x) = 0.18938 | ||grad|| = 1.15e-01
[STEP   21] f(x) = 0.18160 | ||grad|| = 1.12e-01
[STEP   22] f(x) = 0.17423 | ||grad|| = 1.09e-01
[STEP   23] f(x) = 0.16726 | ||grad|| = 1.06e-01
[STEP   24] f(x) = 0.16065 | ||grad|| = 1.03e-01
...
[STEP 3847] f(x) = 0.01908 | ||grad|| = 1.00e-04
[TEST] R2:   0.957
[TEST] MSE:  5.107 s²
[TEST] RMSE: 2.260 s
```

Gradient Descent (Lipschitz convex)

$$\gamma = \frac{R}{B\sqrt{T}} = \frac{\|x_0 - x^*\|}{\|\nabla f(x_0)\| \cdot \sqrt{T}}$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *x** = 0.01908 (GD result).

```
[TRAIN] Gradient Descent Lipschitz Convex:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.49961 | ||grad|| = 3.09e-01
[STEP    3] f(x) = 0.49922 | ||grad|| = 3.09e-01
[STEP    4] f(x) = 0.49883 | ||grad|| = 3.08e-01
[STEP    5] f(x) = 0.49844 | ||grad|| = 3.07e-01
[STEP    6] f(x) = 0.49806 | ||grad|| = 3.06e-01
[STEP    7] f(x) = 0.49768 | ||grad|| = 3.06e-01
[STEP    8] f(x) = 0.49730 | ||grad|| = 3.05e-01
[STEP    9] f(x) = 0.49692 | ||grad|| = 3.04e-01
[STEP   10] f(x) = 0.49654 | ||grad|| = 3.03e-01
[STEP   11] f(x) = 0.49616 | ||grad|| = 3.03e-01
[STEP   12] f(x) = 0.49579 | ||grad|| = 3.02e-01
[STEP   13] f(x) = 0.49542 | ||grad|| = 3.01e-01
[STEP   14] f(x) = 0.49505 | ||grad|| = 3.01e-01
[STEP   15] f(x) = 0.49468 | ||grad|| = 3.00e-01
[STEP   16] f(x) = 0.49432 | ||grad|| = 2.99e-01
[STEP   17] f(x) = 0.49395 | ||grad|| = 2.98e-01
[STEP   18] f(x) = 0.49359 | ||grad|| = 2.98e-01
[STEP   19] f(x) = 0.49323 | ||grad|| = 2.97e-01
[STEP   20] f(x) = 0.49287 | ||grad|| = 2.96e-01
[STEP   21] f(x) = 0.49251 | ||grad|| = 2.96e-01
[STEP   22] f(x) = 0.49215 | ||grad|| = 2.95e-01
[STEP   23] f(x) = 0.49180 | ||grad|| = 2.94e-01
[STEP   24] f(x) = 0.49145 | ||grad|| = 2.94e-01
...
[STEP 10000] f(x) = 0.04555 | ||grad|| = 3.68e-02
[TEST] R2:   0.904
[TEST] MSE:  11.249 s²
[TEST] RMSE: 3.354 s
```

Gradient Descent (smooth convex)

$$L = \frac{1}{n}\|X^\top X\| \qquad \gamma := \frac{1}{L}$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000.

```
[TRAIN] Gradient Descent Smooth Convex:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.45774 | ||grad|| = 2.27e-01
[STEP    3] f(x) = 0.43257 | ||grad|| = 2.02e-01
[STEP    4] f(x) = 0.41189 | ||grad|| = 1.91e-01
[STEP    5] f(x) = 0.39314 | ||grad|| = 1.84e-01
[STEP    6] f(x) = 0.37565 | ||grad|| = 1.79e-01
[STEP    7] f(x) = 0.35922 | ||grad|| = 1.73e-01
[STEP    8] f(x) = 0.34371 | ||grad|| = 1.69e-01
[STEP    9] f(x) = 0.32907 | ||grad|| = 1.64e-01
[STEP   10] f(x) = 0.31522 | ||grad|| = 1.59e-01
[STEP   11] f(x) = 0.30212 | ||grad|| = 1.55e-01
[STEP   12] f(x) = 0.28970 | ||grad|| = 1.51e-01
[STEP   13] f(x) = 0.27794 | ||grad|| = 1.47e-01
[STEP   14] f(x) = 0.26678 | ||grad|| = 1.43e-01
[STEP   15] f(x) = 0.25620 | ||grad|| = 1.40e-01
[STEP   16] f(x) = 0.24615 | ||grad|| = 1.36e-01
[STEP   17] f(x) = 0.23660 | ||grad|| = 1.33e-01
[STEP   18] f(x) = 0.22753 | ||grad|| = 1.29e-01
[STEP   19] f(x) = 0.21890 | ||grad|| = 1.26e-01
[STEP   20] f(x) = 0.21069 | ||grad|| = 1.23e-01
[STEP   21] f(x) = 0.20288 | ||grad|| = 1.20e-01
[STEP   22] f(x) = 0.19545 | ||grad|| = 1.17e-01
[STEP   23] f(x) = 0.18837 | ||grad|| = 1.14e-01
[STEP   24] f(x) = 0.18162 | ||grad|| = 1.12e-01
...
[STEP 4417] f(x) = 0.01908 | ||grad|| = 1.00e-04
[TEST] R2:   0.957
[TEST] MSE:  5.107 s²
[TEST] RMSE: 2.260 s
```

Gradient Descent (strongly convex)

$$L = \frac{1}{n}\|X^\top X\| \qquad \mu = \tfrac{1}{n}\lambda_{\min}(X^\top X)$$

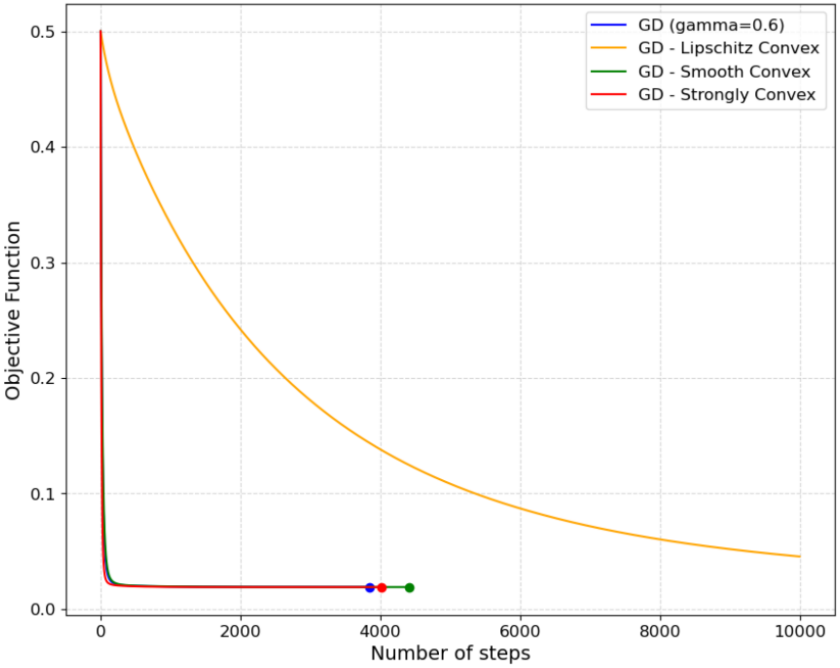
$$\gamma = \frac{2}{L+\mu}$$

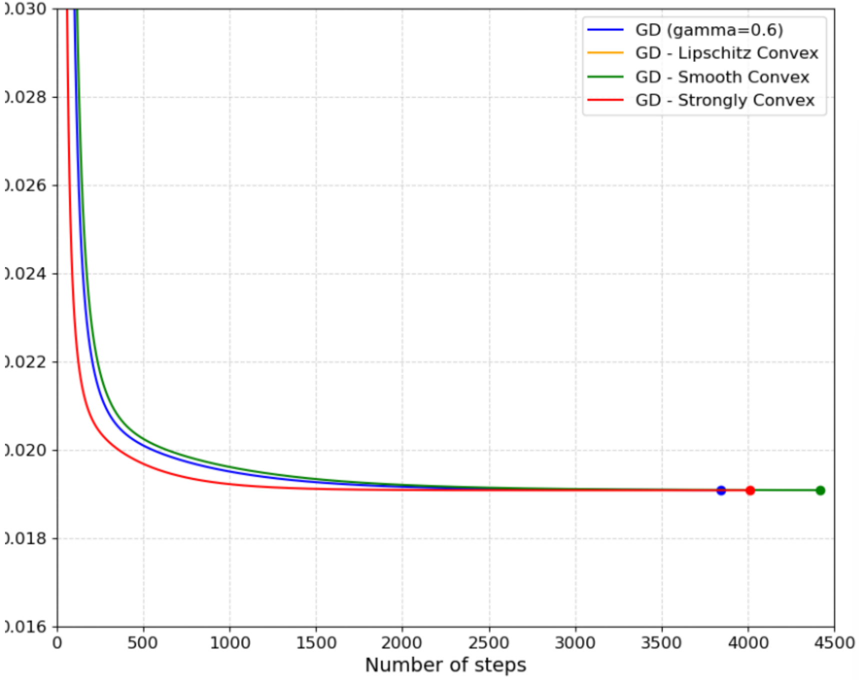
Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000.

```
[TRAIN] Gradient Descent Strongly Convex:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.43156 | ||grad|| = 2.02e-01
[STEP    3] f(x) = 0.39230 | ||grad|| = 1.87e-01
[STEP    4] f(x) = 0.35816 | ||grad|| = 1.76e-01
[STEP    5] f(x) = 0.32782 | ||grad|| = 1.67e-01
[STEP    6] f(x) = 0.30073 | ||grad|| = 1.58e-01
[STEP    7] f(x) = 0.27647 | ||grad|| = 1.50e-01
[STEP    8] f(x) = 0.25469 | ||grad|| = 1.43e-01
[STEP    9] f(x) = 0.23507 | ||grad|| = 1.36e-01
[STEP   10] f(x) = 0.21738 | ||grad|| = 1.30e-01
[STEP   11] f(x) = 0.20138 | ||grad|| = 1.24e-01
[STEP   12] f(x) = 0.18690 | ||grad|| = 1.18e-01
[STEP   13] f(x) = 0.17377 | ||grad|| = 1.13e-01
[STEP   14] f(x) = 0.16184 | ||grad|| = 1.08e-01
[STEP   15] f(x) = 0.15099 | ||grad|| = 1.04e-01
[STEP   16] f(x) = 0.14112 | ||grad|| = 9.97e-02
[STEP   17] f(x) = 0.13211 | ||grad|| = 9.57e-02
[STEP   18] f(x) = 0.12389 | ||grad|| = 9.20e-02
[STEP   19] f(x) = 0.11637 | ||grad|| = 8.85e-02
[STEP   20] f(x) = 0.10950 | ||grad|| = 8.52e-02
[STEP   21] f(x) = 0.10320 | ||grad|| = 8.21e-02
[STEP   22] f(x) = 0.09742 | ||grad|| = 7.92e-02
[STEP   23] f(x) = 0.09212 | ||grad|| = 7.65e-02
[STEP   24] f(x) = 0.08725 | ||grad|| = 7.39e-02
...
[STEP 4013] f(x) = 0.01908 | ||grad|| = 1.00e-04
[TEST] R2:   0.957
[TEST] MSE:  5.105 s²
[TEST] RMSE: 2.259 s
```

Comparison of GD methods





Stochastical Gradient Descent (basic)

$$x_{t+1} := x_t - \gamma \nabla f_i(x_t)$$

• Each epoch consists of several gradient updates.
• The number of updates per epoch is as *max_iter / epochs*.
• In each step, a mini-batch of size *batch_size* is selected uniformly at random without replacement.

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *gamma* = 0.05;
- *batch_size* = 32;
- *epochs* = 20.

```
[TRAIN] Stochastical Gradient Descent:
   Epoch: 1/20 | Step: 1/500
[STEP     1] batch loss = 0.51212 | f(x) = 0.50000 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 2/500
[STEP     2] batch loss = 0.56820 | f(x) = 0.49685 | ||grad|| = 3.05e-01
   Epoch: 1/20 | Step: 3/500
[STEP     3] batch loss = 0.47040 | f(x) = 0.49185 | ||grad|| = 2.96e-01
   Epoch: 1/20 | Step: 4/500
[STEP     4] batch loss = 0.43891 | f(x) = 0.48855 | ||grad|| = 2.91e-01
   Epoch: 1/20 | Step: 5/500
[STEP     5] batch loss = 0.45475 | f(x) = 0.48374 | ||grad|| = 2.80e-01
   Epoch: 1/20 | Step: 6/500
[STEP     6] batch loss = 0.43605 | f(x) = 0.48050 | ||grad|| = 2.76e-01
   Epoch: 1/20 | Step: 7/500
[STEP     7] batch loss = 0.53208 | f(x) = 0.47747 | ||grad|| = 2.72e-01
   Epoch: 1/20 | Step: 8/500
[STEP     8] batch loss = 0.59147 | f(x) = 0.47283 | ||grad|| = 2.61e-01
   Epoch: 1/20 | Step: 9/500
[STEP     9] batch loss = 0.67909 | f(x) = 0.47150 | ||grad|| = 2.67e-01
   Epoch: 1/20 | Step: 10/500
[STEP    10] batch loss = 0.54173 | f(x) = 0.46619 | ||grad|| = 2.56e-01
   Epoch: 1/20 | Step: 11/500
[STEP    11] batch loss = 0.34960 | f(x) = 0.46520 | ||grad|| = 2.60e-01
   Epoch: 1/20 | Step: 12/500
[STEP    12] batch loss = 0.39757 | f(x) = 0.46310 | ||grad|| = 2.55e-01
...
[STEP 10000] batch loss = 0.01760 | f(x) = 0.01971 | ||grad|| = 1.00e-02
[TEST] R2:   0.955
[TEST] MSE:  5.264 s²
[TEST] RMSE: 2.294 s
```

Stochastical Gradient Descent (Lipschitz convex)

$$\gamma = \frac{R}{B\sqrt{T}} = \frac{\|x_0 - x^*\|}{\|\nabla f(x_0)\| \cdot \sqrt{T}}$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *batch_size* = 32;
- *epochs* = 20.

```
[TRAIN] Stochastical Gradient Descent Lipschitz Convex:
   Epoch: 1/20 | Step: 1/500
[STEP     1] batch loss = 0.42238 | f(x) = 0.50000 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 2/500
[STEP     2] batch loss = 0.37405 | f(x) = 0.49977 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 3/500
[STEP     3] batch loss = 0.53735 | f(x) = 0.49959 | ||grad|| = 3.09e-01
   Epoch: 1/20 | Step: 4/500
[STEP     4] batch loss = 0.44097 | f(x) = 0.49930 | ||grad|| = 3.09e-01
   Epoch: 1/20 | Step: 5/500
[STEP     5] batch loss = 0.47149 | f(x) = 0.49907 | ||grad|| = 3.09e-01
   Epoch: 1/20 | Step: 6/500
[STEP     6] batch loss = 0.51910 | f(x) = 0.49871 | ||grad|| = 3.08e-01
   Epoch: 1/20 | Step: 7/500
[STEP     7] batch loss = 0.78506 | f(x) = 0.49856 | ||grad|| = 3.08e-01
   Epoch: 1/20 | Step: 8/500
[STEP     8] batch loss = 0.54381 | f(x) = 0.49766 | ||grad|| = 3.05e-01
   Epoch: 1/20 | Step: 9/500
[STEP     9] batch loss = 0.57174 | f(x) = 0.49722 | ||grad|| = 3.05e-01
   Epoch: 1/20 | Step: 10/500
[STEP    10] batch loss = 0.47979 | f(x) = 0.49668 | ||grad|| = 3.03e-01
   Epoch: 1/20 | Step: 11/500
[STEP    11] batch loss = 0.52904 | f(x) = 0.49609 | ||grad|| = 3.02e-01
   Epoch: 1/20 | Step: 12/500
[STEP    12] batch loss = 0.46823 | f(x) = 0.49556 | ||grad|| = 3.01e-01
...
[STEP 10000] batch loss = 0.10987 | f(x) = 0.04558 | ||grad|| = 3.74e-02
[TEST] R2:   0.904
[TEST] MSE:  11.259 s²
[TEST] RMSE: 3.355 s
```

Stochastical Gradient Descent (strongly convex)

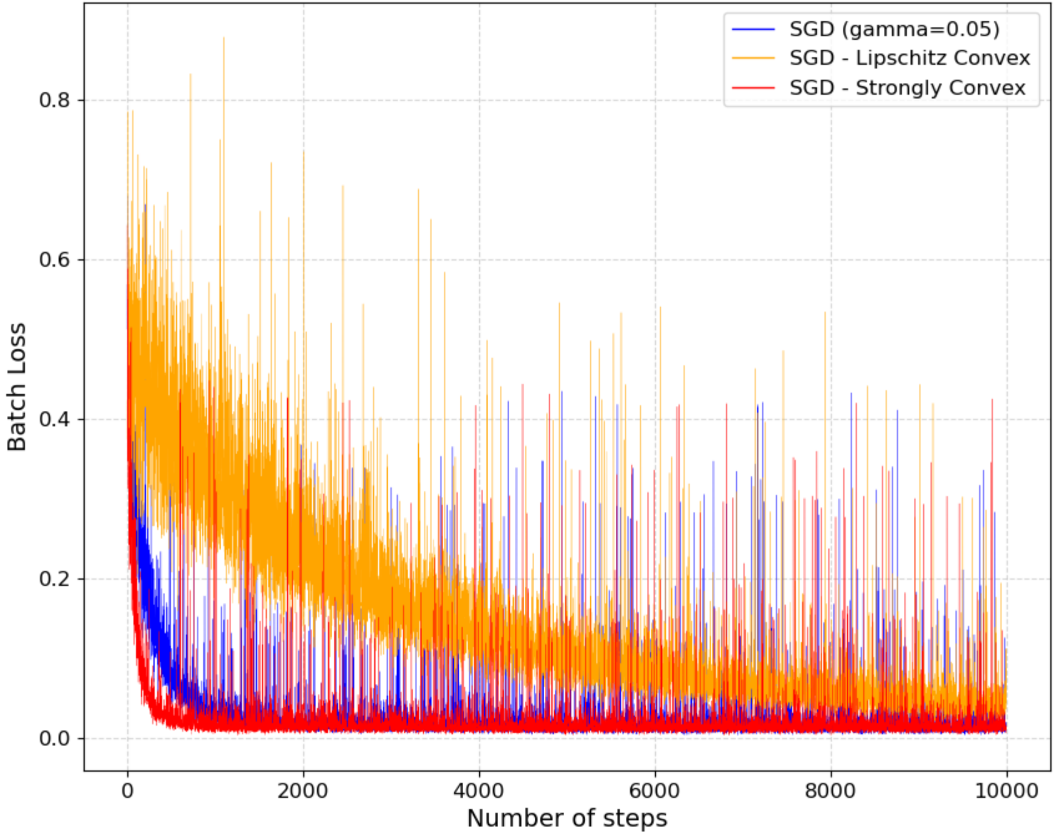
$$\mu = \frac{1}{n}\lambda_{\min}(X^\top X) \qquad \gamma_t = \frac{2}{\mu(t+1)}$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *batch_size* = 32;
- *epochs* = 20;
- *t0* = 10000.

```
[TRAIN] Stochastical Gradient Descent Strongly Convex:
[INFO] mu = 1.391e-03
   Epoch: 1/20 | Step: 1/500
[STEP     1] batch loss = 0.64239 | f(x) = 0.50000 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 2/500
[STEP     2] batch loss = 0.52280 | f(x) = 0.48161 | ||grad|| = 2.78e-01
   Epoch: 1/20 | Step: 3/500
[STEP     3] batch loss = 0.50150 | f(x) = 0.47486 | ||grad|| = 2.75e-01
   Epoch: 1/20 | Step: 4/500
[STEP     4] batch loss = 0.58777 | f(x) = 0.47008 | ||grad|| = 2.80e-01
   Epoch: 1/20 | Step: 5/500
[STEP     5] batch loss = 0.39681 | f(x) = 0.46852 | ||grad|| = 3.22e-01
   Epoch: 1/20 | Step: 6/500
[STEP     6] batch loss = 0.53487 | f(x) = 0.47740 | ||grad|| = 3.78e-01
   Epoch: 1/20 | Step: 7/500
[STEP     7] batch loss = 0.34691 | f(x) = 0.46028 | ||grad|| = 3.17e-01
   Epoch: 1/20 | Step: 8/500
[STEP     8] batch loss = 0.54930 | f(x) = 0.44895 | ||grad|| = 2.81e-01
   Epoch: 1/20 | Step: 9/500
[STEP     9] batch loss = 0.42627 | f(x) = 0.43792 | ||grad|| = 2.52e-01
   Epoch: 1/20 | Step: 10/500
[STEP    10] batch loss = 0.45343 | f(x) = 0.43380 | ||grad|| = 2.64e-01
   Epoch: 1/20 | Step: 11/500
[STEP    11] batch loss = 0.49409 | f(x) = 0.42654 | ||grad|| = 2.60e-01
   Epoch: 1/20 | Step: 12/500
...
[STEP 10000] batch loss = 0.01282 | f(x) = 0.01952 | ||grad|| = 2.59e-02
[TEST] R2:   0.956
[TEST] MSE:  5.195 s²
[TEST] RMSE: 2.279 s
```

Comparison of Stochastical GD methods



Adagrad

$$[\mathbf{G}_t]_i := \sum_{s=0}^{t} ([\mathbf{g}_s]_i)^2$$

$$[\mathbf{x}_{t+1}]_i := [\mathbf{x}_t]_i - \gamma \cdot \frac{[\mathbf{g}_t]_i}{\sqrt{[\mathbf{G}_t]_i} + \varepsilon} \quad \forall i$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *gamma* = 0.1;
- *epochs* = 20;
- *epsilon* = 1e-08.

```
[TRAIN] Adagrad:
   Epoch: 1/20 | Step: 1/500
[STEP     1] f(x) = 0.50000 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 2/500
[STEP     2] f(x) = 0.72197 | ||grad|| = 1.06e+00
   Epoch: 1/20 | Step: 3/500
[STEP     3] f(x) = 0.57555 | ||grad|| = 1.19e+00
   Epoch: 1/20 | Step: 4/500
[STEP     4] f(x) = 0.29119 | ||grad|| = 7.45e-01
   Epoch: 1/20 | Step: 5/500
[STEP     5] f(x) = 0.14026 | ||grad|| = 3.42e-01
   Epoch: 1/20 | Step: 6/500
[STEP     6] f(x) = 0.09360 | ||grad|| = 1.62e-01
   Epoch: 1/20 | Step: 7/500
[STEP     7] f(x) = 0.07334 | ||grad|| = 9.05e-02
   Epoch: 1/20 | Step: 8/500
[STEP     8] f(x) = 0.06107 | ||grad|| = 6.76e-02
   Epoch: 1/20 | Step: 9/500
[STEP     9] f(x) = 0.05242 | ||grad|| = 5.54e-02
   Epoch: 1/20 | Step: 10/500
[STEP    10] f(x) = 0.04598 | ||grad|| = 4.91e-02
   Epoch: 1/20 | Step: 11/500
[STEP    11] f(x) = 0.04105 | ||grad|| = 4.35e-02
   Epoch: 1/20 | Step: 12/500
[STEP    12] f(x) = 0.03722 | ||grad|| = 3.92e-02
...
[STEP   712] f(x) = 0.01908 | ||grad|| = 9.96e-05
[TEST] R2:   0.957
[TEST] MSE:  5.105 s²
[TEST] RMSE: 2.259 s
```

Adam

$$\mathbf{m}_t := \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t$$

$$[\mathbf{v}_t]_i := \beta_2 [\mathbf{v}_{t-1}]_i + (1-\beta_2)([\mathbf{g}_t]_i)^2 \quad \forall i$$

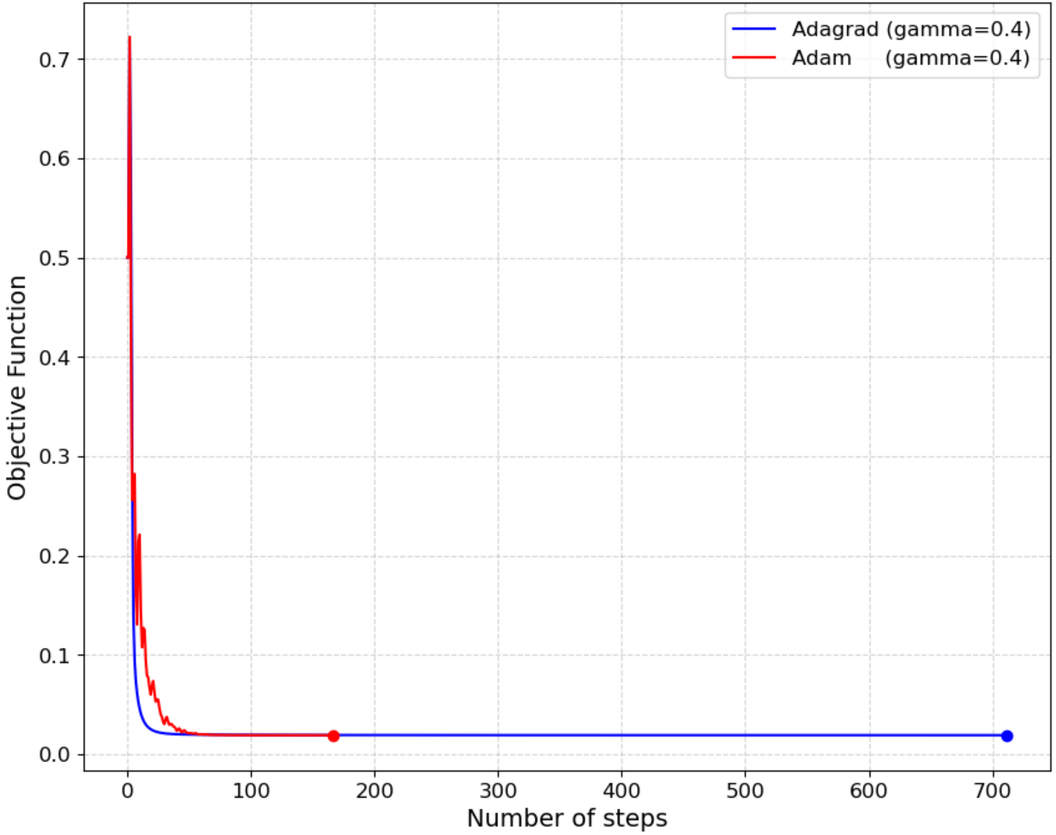
$$[\mathbf{x}_{t+1}]_i := [\mathbf{x}_t]_i - \gamma \cdot \frac{[\hat{\mathbf{m}}_t]_i}{\sqrt{[\hat{\mathbf{v}}_t]_i} + \varepsilon} \quad \forall i$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *gamma* = 0.4;
- *epochs* = 20;
- *epsilon* = 1e-08;
- *beta1* = 0.9;
- *beta2* = 0.999.

```
[TRAIN] Adam:
   Epoch: 1/20 | Step: 1/500
[STEP     1] f(x) = 0.50000 | ||grad|| = 3.10e-01
   Epoch: 1/20 | Step: 2/500
[STEP     2] f(x) = 0.72197 | ||grad|| = 1.06e+00
   Epoch: 1/20 | Step: 3/500
[STEP     3] f(x) = 0.44041 | ||grad|| = 8.99e-01
   Epoch: 1/20 | Step: 4/500
[STEP     4] f(x) = 0.25586 | ||grad|| = 6.28e-01
   Epoch: 1/20 | Step: 5/500
[STEP     5] f(x) = 0.25651 | ||grad|| = 6.38e-01
   Epoch: 1/20 | Step: 6/500
[STEP     6] f(x) = 0.28212 | ||grad|| = 7.91e-01
   Epoch: 1/20 | Step: 7/500
[STEP     7] f(x) = 0.16497 | ||grad|| = 5.15e-01
   Epoch: 1/20 | Step: 8/500
[STEP     8] f(x) = 0.13054 | ||grad|| = 2.77e-01
   Epoch: 1/20 | Step: 9/500
[STEP     9] f(x) = 0.21314 | ||grad|| = 5.51e-01
   Epoch: 1/20 | Step: 10/500
[STEP    10] f(x) = 0.22101 | ||grad|| = 5.88e-01
   Epoch: 1/20 | Step: 11/500
[STEP    11] f(x) = 0.14442 | ||grad|| = 3.79e-01
   Epoch: 1/20 | Step: 12/500
[STEP    12] f(x) = 0.10761 | ||grad|| = 2.85e-01
...
[STEP   168] f(x) = 0.01908 | ||grad|| = 5.22e-05
[TEST] R2:   0.957
[TEST] MSE:  5.105 s²
[TEST] RMSE: 2.259 s
```

Comparison of Adagrad and Adam



Newton-Raphson Method

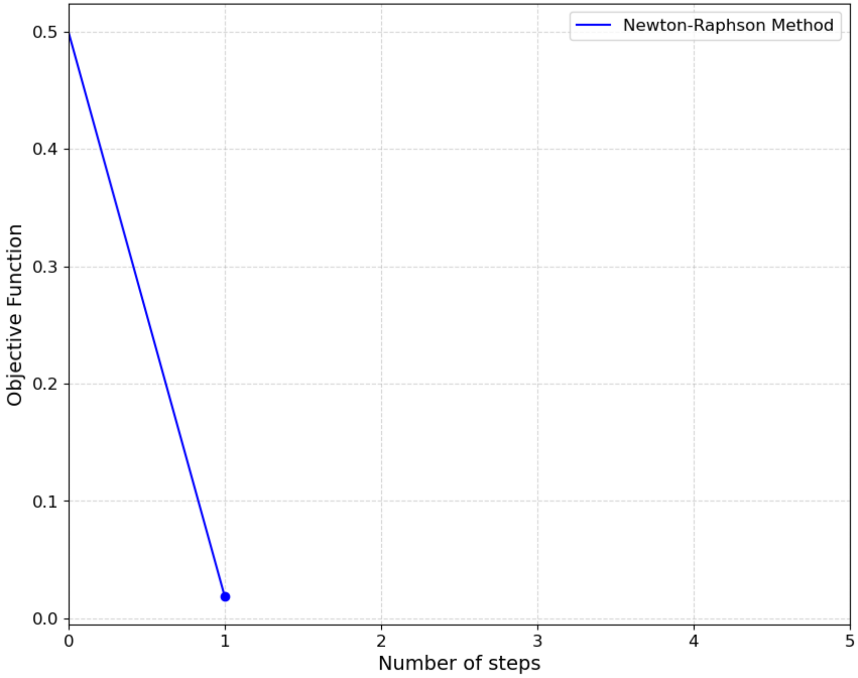
$$\mathbf{x}_{t+1} := \mathbf{x}_t - \nabla^2 f(\mathbf{x}_t)^{-1} \nabla f(\mathbf{x}_t)$$

Key property: because uses a quadratic approximation of the function, it converges in just one step if the function is exactly quadratic.

Assumptions:

- *tol* = 1.00e-04;
- *x0* = zero vector;
- *max_iter* = 10 000.

```
[TRAIN] Newton-Raphson Method:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.01908 | ||grad|| = 5.06e-16
[TEST] R2:   0.957
[TEST] MSE:  5.105 s²
[TEST] RMSE: 2.259 s
```



Random Coordinate Gradient Descent

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \frac{1}{L_{i_t}} \nabla_{i_t} f(\mathbf{x}_t) \mathbf{e}_{i_t}$$

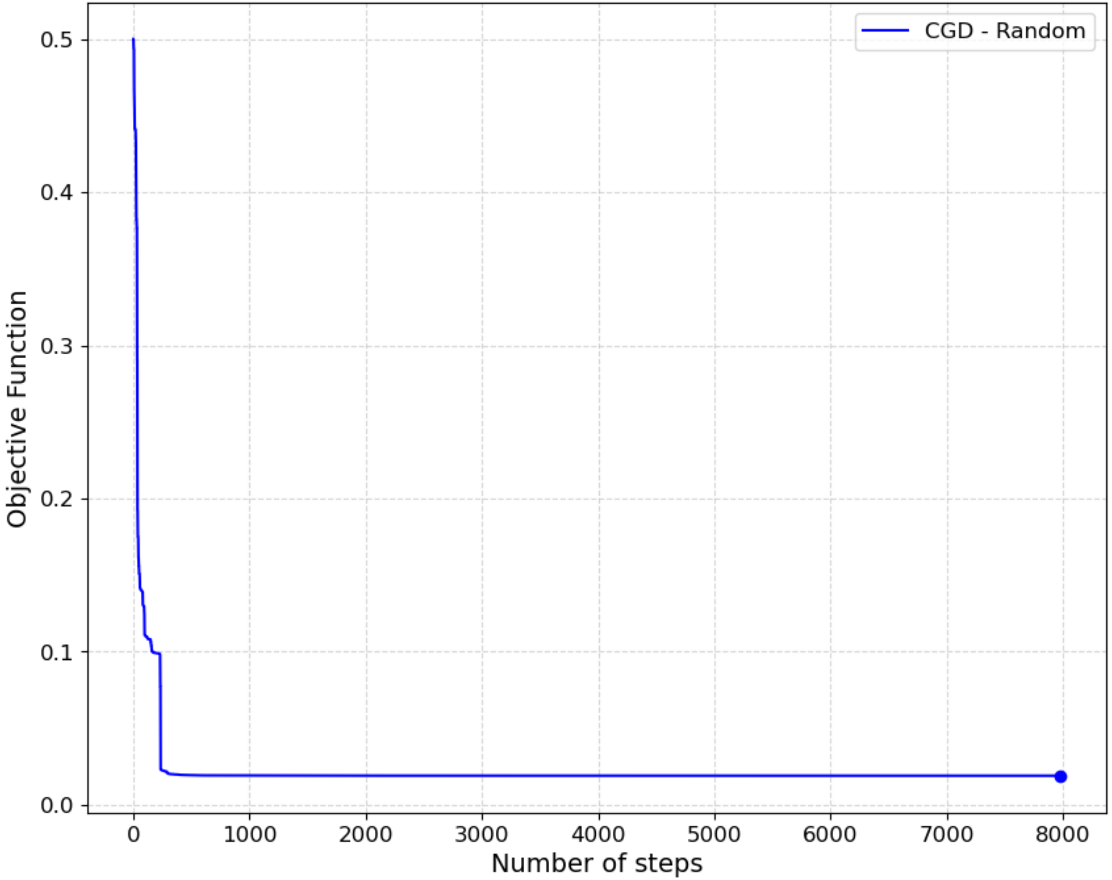
$$L_j = \frac{1}{n} \sum_{k=1}^{n} X_{k,j}^2$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000.

```
[TRAIN] Random Coordinate Gradient Descent:
[STEP    1] f(x) = 0.50000 | ||grad|| = 3.10e-01
[STEP    2] f(x) = 0.49999 | ||grad|| = 3.10e-01
[STEP    3] f(x) = 0.49418 | ||grad|| = 3.02e-01
[STEP    4] f(x) = 0.49377 | ||grad|| = 2.99e-01
[STEP    5] f(x) = 0.49358 | ||grad|| = 3.00e-01
[STEP    6] f(x) = 0.49358 | ||grad|| = 2.99e-01
[STEP    7] f(x) = 0.46822 | ||grad|| = 3.41e-01
[STEP    8] f(x) = 0.46750 | ||grad|| = 3.37e-01
[STEP    9] f(x) = 0.46311 | ||grad|| = 3.29e-01
[STEP   10] f(x) = 0.45466 | ||grad|| = 3.53e-01
[STEP   11] f(x) = 0.45466 | ||grad|| = 3.53e-01
[STEP   12] f(x) = 0.44208 | ||grad|| = 3.01e-01
[STEP   13] f(x) = 0.44203 | ||grad|| = 3.01e-01
[STEP   14] f(x) = 0.44202 | ||grad|| = 3.01e-01
[STEP   15] f(x) = 0.44103 | ||grad|| = 2.97e-01
[STEP   16] f(x) = 0.44101 | ||grad|| = 2.97e-01
[STEP   17] f(x) = 0.44094 | ||grad|| = 2.97e-01
[STEP   18] f(x) = 0.44094 | ||grad|| = 2.97e-01
[STEP   19] f(x) = 0.44094 | ||grad|| = 2.97e-01
[STEP   20] f(x) = 0.44088 | ||grad|| = 2.96e-01
[STEP   21] f(x) = 0.43766 | ||grad|| = 2.83e-01
[STEP   22] f(x) = 0.43171 | ||grad|| = 2.88e-01
[STEP   23] f(x) = 0.43171 | ||grad|| = 2.88e-01
[STEP   24] f(x) = 0.41689 | ||grad|| = 2.68e-01
...
[STEP 7972] f(x) = 0.01908 | ||grad|| = 9.90e-05
[TEST] R2:   0.957
[TEST] MSE:  5.105 s²
[TEST] RMSE: 2.259 s
```

Random Coordinate GD



Proximal Gradient Descent for Lasso

- **Lasso regression:** tends to zero out many coefficients, making the model more interpretable. The last term of the Lasso objective function is not differentiable; therefore, this problem is well-suited for Proximal Gradient Descent, as the first part is differentiable while the second is not.

$$f(\mathbf{w}) = \frac{1}{2n}\sum_{k=1}^{n}(y_k - \mathbf{x}_k^\top \mathbf{w})^2 + \lambda \sum_{j=1}^{d} w_j^2$$

- At each step of Proximal Gradient Descent, we apply the following proximal operator, the **soft-thresholding function**.

$$\mathrm{prox}_{\lambda\|\cdot\|_1,\gamma}(\mathbf{z}) = \mathrm{sign}(\mathbf{z})\max\left(|\mathbf{z}| - \gamma\lambda, 0\right)$$

Proximal Gradient Descent

$$\mathbf{x}_{t+1} = \mathrm{prox}_{h,\gamma}\left(\mathbf{x}_t - \gamma \nabla g(\mathbf{x}_t)\right)$$

Assumptions:

- *tol* = 1e-04;
- *x0* = zero vector;
- *max_iter* = 10000;
- *gamma* = 1;
- *lambda_min* = 0.005;
- *lambda_max* = 1;
- *lambdas* = 20.

The code takes as input a range of lambda values, defined by *lambda_min*, *lambda_max*, and *lambdas*, and performs Proximal GD for each of them. In the end, it returns the solution corresponding to the lambda that yields the best results.

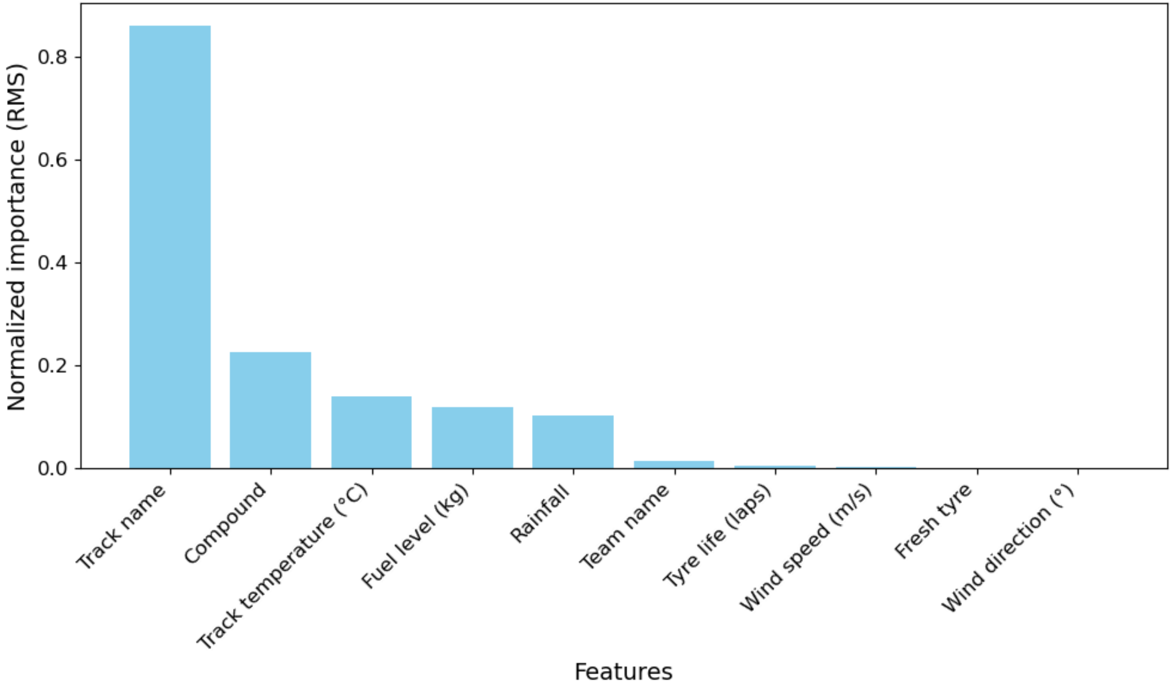
```
[TRAIN] Proximal Gradient Descent for Lasso function:

[TRAIN] Lambda = 5.000e-03
[TEST] R2:   0.927
[TEST] MSE:  8.547 s²
[TEST] RMSE: 2.924 s

[TRAIN] Lambda = 5.737e-02
[TEST] R2:   0.257
[TEST] MSE:  87.223 s²
[TEST] RMSE: 9.339 s

[TRAIN] Lambda = 1.097e-01
[TEST] R2:   0.016
[TEST] MSE:  115.559 s²
[TEST] RMSE: 10.750 s

[TRAIN] Lambda = 1.621e-01
[TEST] R2:   0.002
[TEST] MSE:  117.203 s²
[TEST] RMSE: 10.826 s

[TRAIN] Lambda = 2.145e-01
[TEST] R2:   -0.000
[TEST] MSE:  117.452 s²
...
[FINAL RESULTS] Best lambda = 5.000e-03
[TEST] R2:   0.927
[TEST] MSE:  8.547 s²
[TEST] RMSE: 2.924 s
```

	Feature	Coefficient
0	bias	-0.000
1	Track name [Australian Grand Prix]	-0.452
2	Track name [Austrian Grand Prix]	-1.483
3	Track name [Azerbaijan Grand Prix]	1.725
4	Track name [Bahrain Grand Prix]	0.555
5	Track name [Belgian Grand Prix]	1.980
6	Track name [British Grand Prix]	0.155
7	Track name [Canadian Grand Prix]	-0.799
8	Track name [Chinese Grand Prix]	0.926
9	Track name [Dutch Grand Prix]	-1.040
10	Track name [Emilia Romagna Grand Prix]	-0.311
11	Track name [Hungarian Grand Prix]	-0.141
12	Track name [Italian Grand Prix]	-0.000
13	Track name [Japanese Grand Prix]	0.719
14	Track name [Las Vegas Grand Prix]	0.465
15	Track name [Mexico City Grand Prix]	-0.322
16	Track name [Miami Grand Prix]	0.380
17	Track name [Monaco Grand Prix]	-0.621
18	Track name [Qatar Grand Prix]	-0.084
19	Track name [Saudi Arabian Grand Prix]	0.321
20	Track name [Singapore Grand Prix]	0.838
21	Track name [Spanish Grand Prix]	-0.653
22	Track name [São Paulo Grand Prix]	-0.843
23	Track name [United States Grand Prix]	1.193
24	Team name [Aston Martin]	-0.000
25	Team name [Ferrari]	-0.015
26	Team name [Haas F1 Team]	0.000
27	Team name [Kick Sauber]	0.015
28	Team name [McLaren]	-0.009
29	Team name [Mercedes]	-0.001
30	Team name [RB]	0.000
31	Team name [Red Bull Racing]	-0.034
32	Team name [Williams]	0.001
33	Compound [INTERMEDIATE]	0.437
34	Compound [MEDIUM]	-0.000
35	Compound [SOFT]	-0.122
36	Compound [WET]	0.000
37	Fresh tyre [True]	-0.000
38	Rainfall [True]	0.102
39	Tyre life (laps)	0.005
40	Track temperature (°C)	-0.139
41	Wind direction (°)	-0.000
42	Wind speed (m/s)	-0.004
43	Fuel level (kg)	0.118

Coefficient importance in Lasso

- Feature importance is computed using the **RMS** (Root Mean Square) of the coefficients of the dummy variables if the feature is **categorical**, and using the absolute value of the coefficient if the feature is **numerical**.

- The variables *Wind speed*, *Fresh tyre*, and *Wind direction* are not particularly relevant. We can remove them to simplify the model.



UNIVERSITÀ DEGLI STUDI DI BERGAMO | Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Conclusion

Conclusion

- The most effective method for training the model is the **Newton-Raphson method**, as it converges in a single step and reaches a more precise minimum.

- The overall **results are promising**, especially considering that the model does not account for factors such as driver skill, strategy, etc. which can significantly influence performance in this context.

- The final model is implemented in the file *3_Final_Model*, and is ready for use by the user. Here, the model is implemented using the **entire dataset** for training, in order to achieve greater robustness.

	Feature	Coefficient
0	bias	0.127
1	Track name [Australian Grand Prix]	-0.689
2	Track name [Austrian Grand Prix]	-1.761
3	Track name [Azerbaijan Grand Prix]	1.636
4	Track name [Bahrain Grand Prix]	0.770
5	Track name [Belgian Grand Prix]	1.985
6	Track name [British Grand Prix]	0.292
7	Track name [Canadian Grand Prix]	-1.054
8	Track name [Chinese Grand Prix]	1.163
9	Track name [Dutch Grand Prix]	-1.216
10	Track name [Emilia Romagna Grand Prix]	-0.783
11	Track name [Hungarian Grand Prix]	-0.568
12	Track name [Italian Grand Prix]	-0.414
13	Track name [Japanese Grand Prix]	0.760
14	Track name [Las Vegas Grand Prix]	0.885
15	Track name [Mexico City Grand Prix]	-0.650
16	Track name [Miami Grand Prix]	0.254
17	Track name [Monaco Grand Prix]	-1.060
18	Track name [Qatar Grand Prix]	-0.182
19	Track name [Saudi Arabian Grand Prix]	0.440
20	Track name [Singapore Grand Prix]	0.834
21	Track name [Spanish Grand Prix]	-0.872
22	Track name [São Paulo Grand Prix]	-1.319
23	Track name [United States Grand Prix]	1.032
24	Team name [Aston Martin]	-0.020
25	Team name [Ferrari]	-0.082
26	Team name [Haas F1 Team]	0.013
27	Team name [Kick Sauber]	0.051
28	Team name [McLaren]	-0.074
29	Team name [Mercedes]	-0.064
30	Team name [RB]	0.016
31	Team name [Red Bull Racing]	-0.096
32	Team name [Williams]	0.037
33	Compound [INTERMEDIATE]	0.892
34	Compound [MEDIUM]	0.009
35	Compound [SOFT]	-0.071
36	Compound [WET]	1.649
37	Rainfall [True]	0.205
38	Tyre life (laps)	0.051
39	Track temperature (°C)	0.031
40	Fuel level (kg)	0.122

References

- Lecture notes and slides provided during the Optimization course, academic year 2024 - 2025.
- Kaggle notebook Data Exploration with OCI: link.
- Code for dataset creation with FastF1 API: link.
- FastF1 API documentation: link.