
Прогноз позиции и финиша в топ-10 в FIA Formula 2 с помощью линейной и логистической регрессии (реализация с нуля)

Острикова Мария
Ds-23b, AIU

1 Введение

FIA Formula 2 — это “вторая ступень” формульных гонок под эгидой FIA и важная серия для подготовки пилотов к Formula 1. Цель проекта — построить простую интерпретируемую модель, способную (1) оценивать ожидаемую итоговую позицию пилота и (2) предсказывать вероятность финиша в очковой зоне (топ-10), используя табличные данные гонки (стартовая позиция, команда, трасса, очки и т.д.).

2 Данные

В работе используется датасет Formula-2 Dataset с Kaggle, содержащий статистику по заездам FIA F2. В ноутбуке используется файл Feature-Race.csv. В текущем состоянии данные имеют 1420 строк и 15 исходных столбцов:

Тип	Признаки
Числовые	SEASON, ROUND, DRIVER, EVENT, STARTINGGRID, RACESPOSITION, POINTS
Категориальные	TEAM, CIRCUIT
Цели	POS (регрессия), $I_{[POS \leq 10]}$ (классификация)

3 Предобработка и формирование признаков

В ноутбуке выполнены следующие шаги:

- Удаление пропусков (dropna); в текущем файле пропусков нет.
- Масштабирование числовых признаков с помощью стандартизации (z-score).
- One-Hot кодирование категориальных признаков TEAM и CIRCUIT.
- После кодирования размерность признакового пространства составляет 47.
- Добавление bias-терма (единичного столбца) для моделей, реализованных “с нуля”.

Таблица 1: Метрики бинарной классификации (оценка на тех же данных, что и обучение, без train/test split).

Модель	Accuracy	Precision	Recall	F1	ROC AUC
Логистическая регрессия (с нуля)	0.699	0.704	0.759	0.730	0.752
Decision Tree (sklearn)	1.000	1.000	1.000	1.000	1.000

4 Модели

4.1 Линейная регрессия (с нуля)

Для предсказания позиции POS используется линейная модель:

$$\hat{y} = \mathbf{X}\mathbf{w}, \quad (1)$$

где $\mathbf{X} \in \mathbb{R}^{n \times d}$ — матрица признаков (включая bias), \mathbf{w} — вектор весов. Оптимизация выполняется градиентным спуском по MSE:

$$\mathcal{L}_{\text{MSE}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2. \quad (2)$$

В ноутбуке использованы настройки: learning rate 0.01 и 800 эпох (для full-batch GD), а также исследовано влияние шага обучения на сходимость.

4.2 Логистическая регрессия (с нуля) Целевая переменная классификации:

$$y = \mathbb{I}[\text{POS} \leq 10]. \quad (3)$$

Вероятность класса 1:

$$p(y=1|\mathbf{x}) = \sigma(\mathbf{x}^\top \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{x}^\top \mathbf{w})}. \quad (4)$$

Оптимизируется бинарная кросс-энтропия с L2-регуляризацией:

$$\mathcal{L}(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^n [y_i \log p_i + (1 - y_i) \log(1 - p_i)] + \lambda \|\mathbf{w}\|_2^2. \quad (5)$$

В ноутбуке: learning rate 0.1, 2000 эпох, $\lambda = 0.001$.

4.3 Дерево решений (sklearn)

В качестве базовой модели добавлено дерево решений (DecisionTreeClassifier) с параметром max_depth. В ноутбуке показан пример обучения и вычисления метрик.

5 Эксперименты и метрики

Для бинарной классификации используются стандартные метрики: Accuracy, Precision, Recall, F1-score и ROC AUC. В ноутбуке метрики вычисляются средствами sklearn.metrics.

6 Результаты

6.1 Классификация (Топ-10)

Таблица 1 содержит результаты, сохранённые в выходах ноутбука.

Интерпретация. Идеальные метрики дерева решений при in-sample оценке почти наверняка означают переобучение (модель “запомнила” данные) и/или утечку целевой информации. Для корректной оценки качества необходимы train/test split или кроссвалидация.

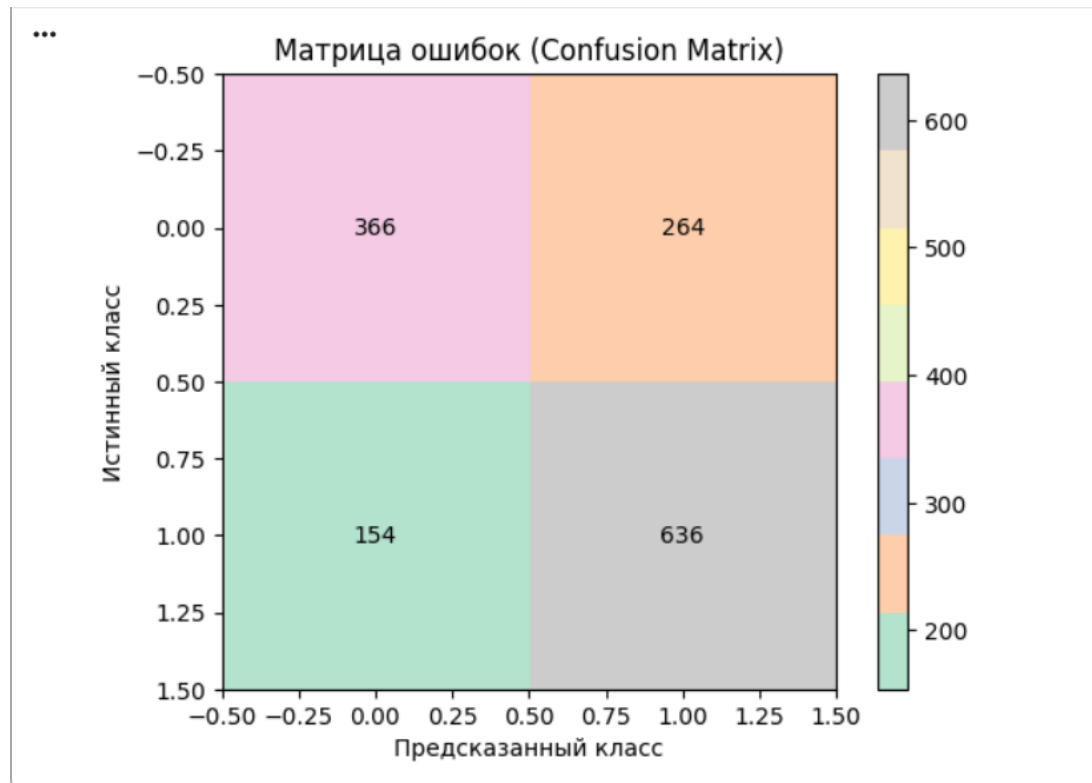


Рис. 1. Confusion Matrix логистической регрессии

Матрица ошибок — это таблица, которая показывает, сколько раз модель предсказала правильно и какие именно ошибки она делает. В ней сравнивается реальный класс и предсказанный: сколько случаев “топ-10” модель действительно распознала как “топ-10”, сколько “не топ-10” правильно отнесла к “не топ-10”, а также два типа ошибок — когда модель ошибочно относит “не топ-10” к “топ-10” и когда, наоборот, пропускает настоящие “топ-10” и предсказывает “не топ-10”. По этой таблице можно понять, каких ошибок больше и в чём именно модель чаще путается.

...

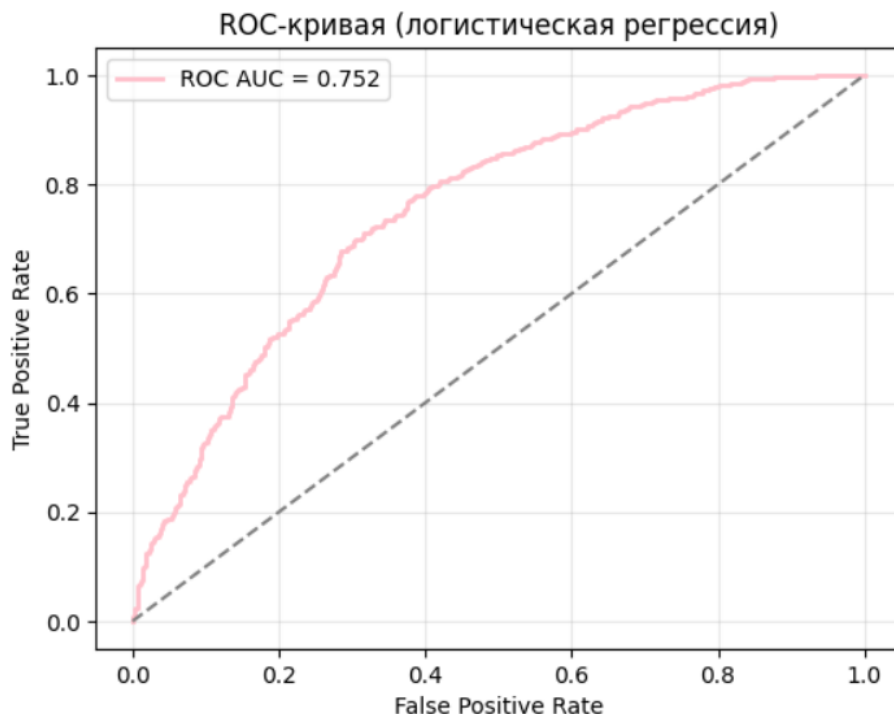


Рис. 2. ROC-кривая логистической регрессии для Top-10 ($POS \leq 10$).

Этот график показывает, насколько хорошо модель отличает два класса ("топ-10" и "не топ-10") при разных порогах. Чем ближе кривая к левому верхнему углу, тем лучше. У меня AUC около 0.75 — значит модель различает классы лучше случайного угадывания, но всё равно делает заметное количество ошибок.

6.2 Регрессия (POS)

В ноутбуке реализована линейная регрессия и построены графики MSE по эпохам (а также сравнение разных learning rate). Для отчёта в формате статьи рекомендуется дополнить этот блок численными метриками (например, MAE/MSE/ R^2) и оценкой на отложенной выборке.

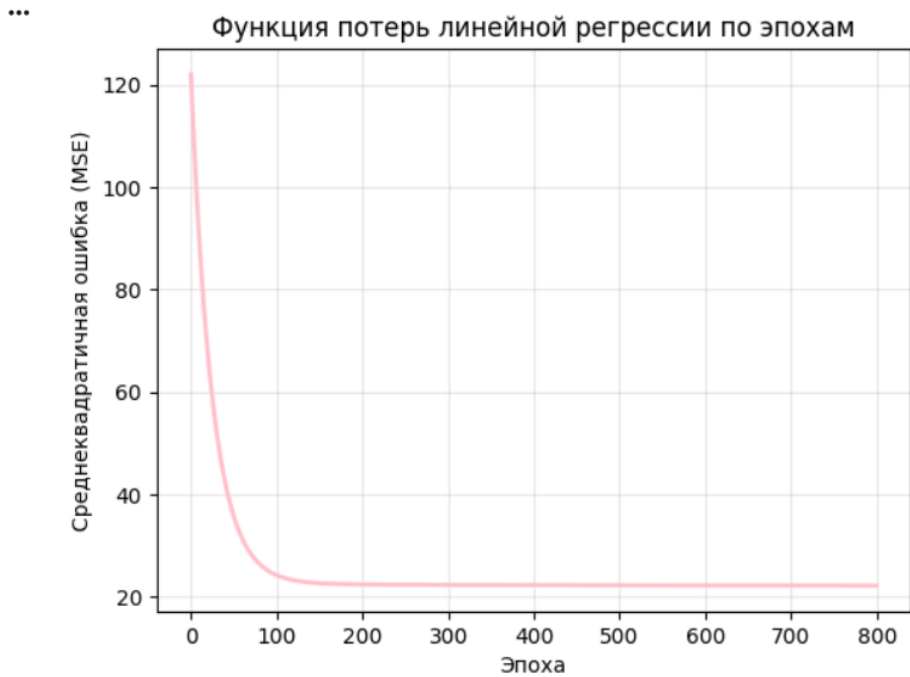


Рис. 3. Функция потерь (MSE) линейной регрессии по эпохам.

Этот график показывает, как обучается линейная регрессия: по горизонтали идут эпохи обучения, по вертикали — ошибка MSE. В начале ошибка быстро уменьшается, потому что модель подбирает веса и начинает лучше предсказывать, а затем кривая выравнивается и почти не меняется. Это означает, что обучение сошлось: модель дошла до состояния, где дальнейшие эпохи уже почти не улучшают результат при текущих данных и признаках.

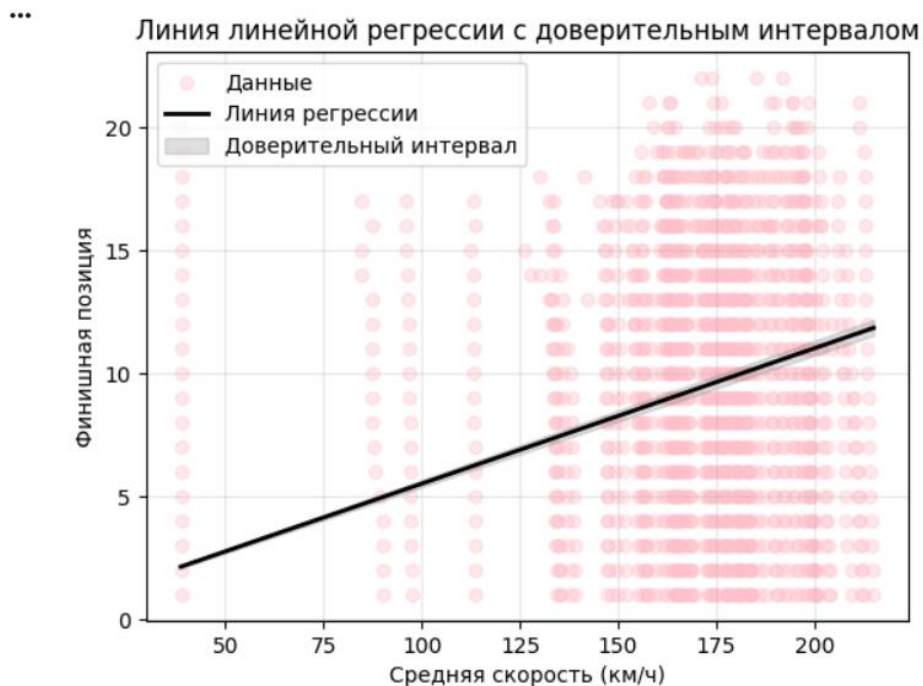


Рис. 4. Линия линейной регрессии с доверительным интервалом (POS vs скорость).

Это визуализация по одному признаку по скорости. Точки - реальные данные, линия - то, что предсказывает линейная регрессия. Серый диапазон показывает, что предсказания не точные и есть неопределённость. Видно общий тренд, но точки сильно разбросаны — значит одного признака недостаточно, и линейная модель объясняет результат не полностью.

7 Ограничения и дальнейшая работа

- Нет разделения на train/test: текущие метрики по классификации рассчитаны на обучающей выборке и завышают качество.
- Размер данных: один файл Feature-Race.csv содержит 1420 строк; при требовании ≥ 1500 объектов нужно объединить больше данных.
- Регрессия без метрик: в ноутбуке есть обучение и графики, но нет финальной численной оценки регрессии.
- Улучшения: добавить кросс-валидацию, подбор гиперпараметров, регуляризацию для линейной регрессии, а также сравнить с более сильными моделями (например, градиентным бустингом) при корректной валидации.

Список литературы

- [1] A. Larchemin. Formula 2 Dataset (Kaggle). <https://www.kaggle.com/datasets/alarchemn/formula-2-dataset>.
- [2] F. Pedregosa et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

А Приложение: соответствие ноутбуку

Ниже кратко перечислены ключевые элементы реализации из ноутбука:

- Загрузка данных из Feature-Race.csv.
- Предобработка: стандартизация числовых и One-Hot для TEAM/CIRCUIT.
- Реализация линейной регрессии (batch GD и SGD) и построение графиков MSE.
- Реализация логистической регрессии с регуляризацией и вывод метрик (Accuracy/Precision/Recall/F1/ROC AUC).
- Демонстрация дерева решений из sklearn и вывод метрик.