МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования «Московский государственный
технический университет имени Н. Э. Баумана (национальный
исследовательский университет)»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
по курсу
«Data Science»

Тема: «Исследование датасета “Оставшееся количество циклов зарядки аккумулятора”»

Слушатель Ларин Петр Михайлович

Москва, 2023

Содержание

Введение

Данная работа посвящена исследованию датасета «Оставшееся количество циклов зарядки аккумулятора». Датасет связывает параметры циклов зарядки и разрядки распространенной модели литий-ионных аккумуляторов (продолжительность, напряжение) с оставшимся полезным сроком службы аккумулятора. Актуальность темы обусловлена динамичным ростом глобального рынка литий-ионных батарей (совокупный среднегодовой темп роста в перспективе до 2030 г. оценивается от 13% до 20%).

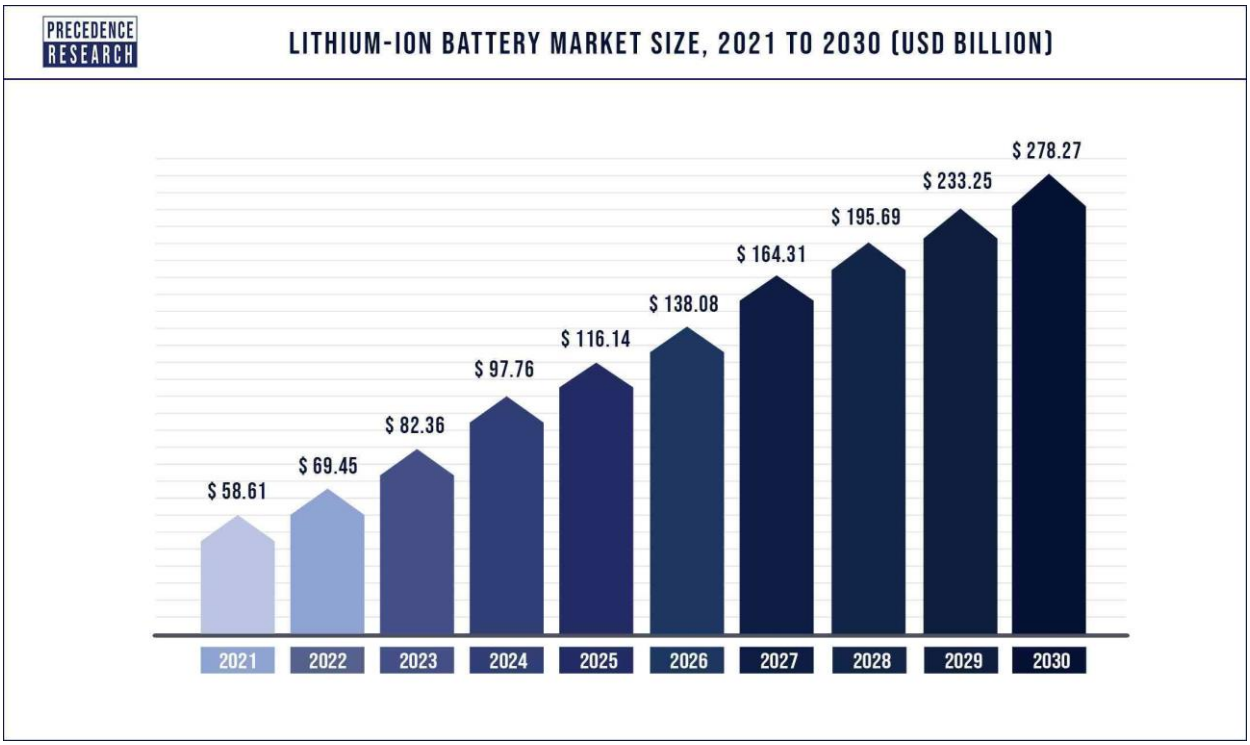

Рисунок 1 – Глобальный рынок литий-ионных батарей.

Источник – PrecedenceResearch.com

1. Описание датасета и постановка задачи

Датасет «Оставшееся количество циклов зарядки аккумулятора» (оригинальное название Battery Remaining Useful Life – RUL) создан аргентинским специалистом по машинному обучению Игнасио Виньюалесом (Ignacio Viñuales) на основе исследований, выполненных Гавайским Институтом природной энергии (HNEI). Исследование включало тестирование 14 аккумуляторов распространенного типа NMC-LCO 18650 номинальной емкости 2.8 А·ч. (Данный тип широко используется в самых разных изделиях, от ноутбуков до электромобилей.) Каждый аккумулятор был подвергнут более чем 1000 циклам разрядки/зарядки при температуре 25°C, включающим CC-CV зарядку в режиме C/2 и разрядку в режиме 1.5C. Переработав первичные данные исследования, автор датасета создал новые признаки, описывающие динамику напряжения и силы тока для каждого цикла, с целью предсказывания оставшегося количества циклов аккумулятора. Датасет содержит сводную информацию о тестировании 14 аккумуляторов.

Датасет и сопутствующая информация размещены на [1, 2].

В датасете 9 столбцов и 15064 строки, то есть более 1000 циклов для каждого из протестированных аккумуляторов. В столбцах приводится следующая информация:

- Индекс цикла (нумерация не сквозная, а повторяется для каждого аккумулятора)
- F1: Полная продолжительность разрядки (с)
- F2: Продолжительность разрядки с 3.6 В до 3.4 В (с)
- F3: Стартовое напряжение при разрядке (В)
- F4: Стартовое напряжение при зарядке (В)
- F5: Продолжительность зарядки до 4.15 В (с)
- F6: Продолжительность зарядки постоянным током (с)
- F7: Полная продолжительность зарядки (с)
- RUL: Целевая переменная (оставшееся количество циклов зарядки аккумулятора)

2. Используемые программные средства

Для анализа датасета использовалась среда Jupyter Notebook в составе пакета Anaconda для Windows. Для разработки приложения также использовалась среда разработки Microsoft Visual Studio. Серверная часть приложения – пакет TensorFlow Serving, размещенный в контейнере Docker Desktop.

3. Разведочный анализ и предобработка

В датасете нет пропусков, но даже при беглом осмотре очевидно наличие выбросов. Поскольку строки датасета содержат результаты последовательных измерений параметров циклов зарядки и разрядки, отличия в значениях соседних строк крайне невелики, порядка единиц процентов. Приведем примеры строк, которые выбиваются из данной нормы:

	F1	F2	F3	F4	F5	F6	F7	RUL
10	3228.58	1135.349333	3.689	3.485	5033.075692	5969.89	5969.89	1101
11	6019.90	1058.279724	4.045	3.475	5053.842846	5980.77	5980.77	1100
12	6026.59	1049.487845	4.047	3.477	5046.429500	5966.82	5966.82	1099
13	6008.07	1065.372059	4.045	3.480	5033.075769	5954.47	5954.47	1098
14	423271.35	168773.265000	4.270	3.108	219923.996000	430028.84	430028.84	1097
15	2261.34	883.200000	4.038	3.901	1949.664000	2922.69	6070.11	1096
16	2259.46	883.199000	4.042	3.373	5181.377000	6161.38	9310.98	1095
17	2256.61	878.400000	4.042	3.374	5181.375000	6154.37	9296.64	1094
18	2252.83	873.601000	4.043	3.374	5174.334000	6147.33	9243.58	1093

Рисунок 2 – пример выброса (строка 14)

На рисунке 2 приведен пример выброса, где значения у пяти признаков отличаются от соседних строк на 2 порядка. Данные признаки содержат время зарядки и разрядки в секундах, и в соседних с выбросом строках это время имеет порядок тысяч секунд, то есть нескольких часов. В строке с выбросом время имеет порядок суток. Нет никаких разумных способов объяснения таких величин чем-либо, кроме ошибки измерения или обработки.

	F1	F2	F3	F4	F5	F6	F7	RUL
7603	2188.81	818.375	4.044	3.384	5002.813	5965.38	9136.38	1049
7604	2186.38	818.438	4.044	3.385	4999.313	5958.31	9101.31	1048
7605	2186.38	818.375	4.045	3.385	4992.375	5958.38	9136.38	1047
7606	2184.00	806.375	4.026	3.705	4956.000	5922.00	9142.00	1046
7607	207013.99	-98271.841	4.272	3.159	73349.980	85080.00	85080.00	1045
7608	11755.36	807.000	4.005	3.380	5250.080	6200.48	9556.67	1044
7609	2184.00	814.400	4.008	3.389	5012.353	5948.35	9188.35	1043
7610	2183.74	816.000	4.009	3.388	5012.318	5948.32	9188.32	1042
7611	2172.00	816.000	4.009	3.389	5005.152	5948.35	9192.90	1041

Рисунок 3 – пример выброса (строка 7607)

На рисунке 3 приводится выброс не только с отличием значений на 2 порядка, но и с отрицательным значением.

	F1	F2	F3	F4	F5	F6	F7	RUL
11822	1017.06	273.371429	3.800	3.674	1457.375000	2096.38	7823.50	41
11823	1014.88	273.500000	3.800	3.673	1452.875000	2096.38	7793.94	40
11824	1008.00	272.457143	3.800	3.675	1448.375000	2060.38	7795.12	39
11825	1012.25	271.542857	3.799	3.676	1443.875000	2060.38	7792.44	38
11826	32.38	42.105263	3.114	3.674	114.250000	8.00	8.00	37
11827	1014.69	271.600000	3.799	3.671	1529.375000	2168.38	7963.75	36
11828	1014.62	271.657143	3.799	3.673	1448.375000	2060.38	7863.06	35
11829	1009.75	269.828571	3.799	3.673	1443.875000	2060.38	7856.69	34
11830	1007.00	268.914286	3.798	3.674	1432.884429	2060.31	7783.56	33

Рисунок 4 – пример выброса (строка 11826)

На рисунке 4 приводится выброс с отличием значений на 2 порядка в меньшую сторону. В отношении данных рисунков также можно сделать вывод об ошибке измерения либо обработки.

Поскольку у каждого выброса аномальные значения присутствуют сразу у 5 признаков, единственным разумным решением будет удаление таких выбросов.

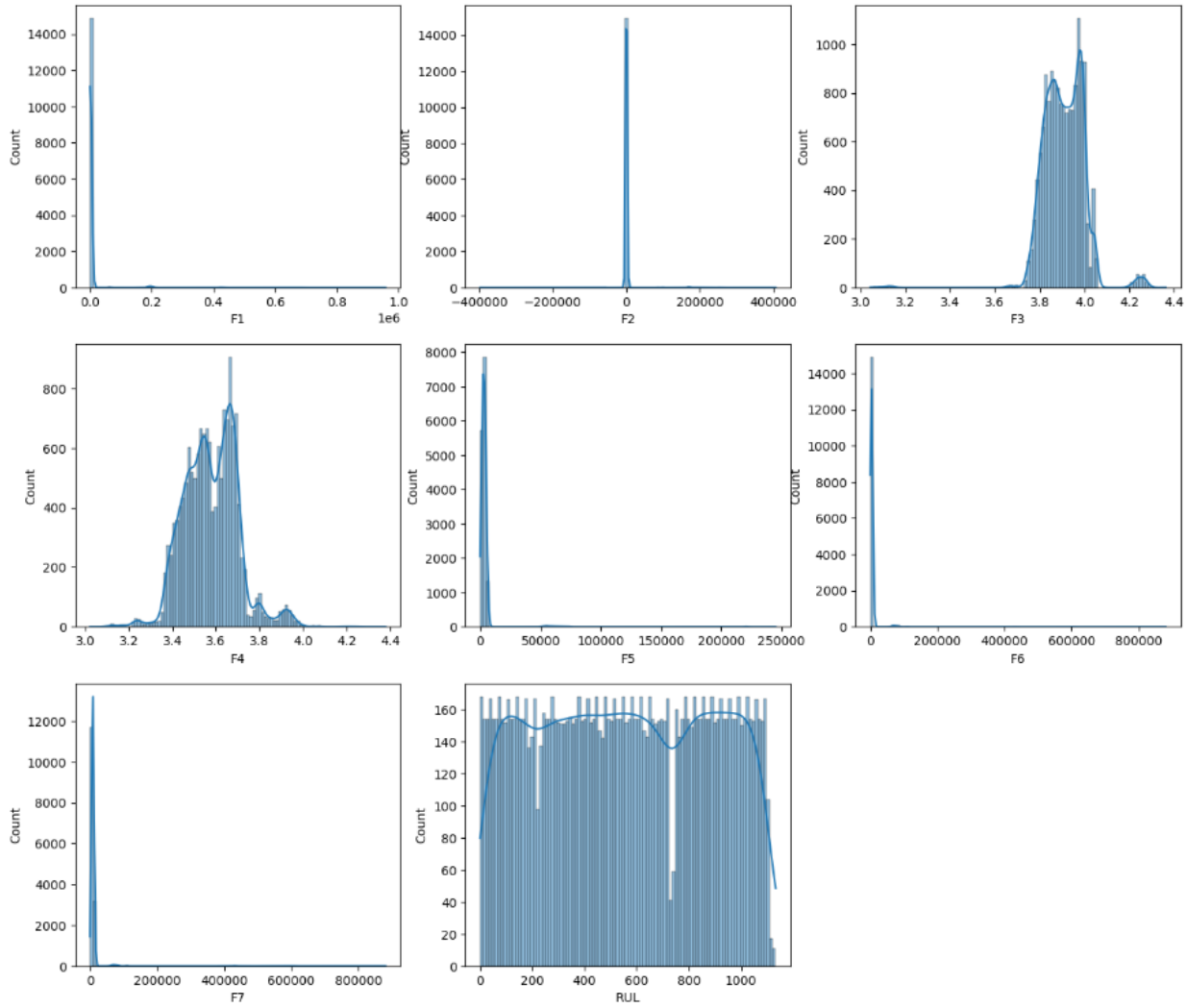

Рисунок 5 – распределения до удаления выбросов

На рисунках 5 и 6 хорошо видно присутствие масштабных выбросов у признаков F1, F2, F5, F6 и F7.

Для выбора порогов удаления выбросов используем графики квартилей данных пяти признаков. В итоге после достаточно консервативного выбора порогов в датасете остается 14845 из 15064 значений, т. е. отброшено 1.45% или 219 значений.

Видно, что выбросы можно было бы удалить более агрессивно, однако мы остановимся на этом.

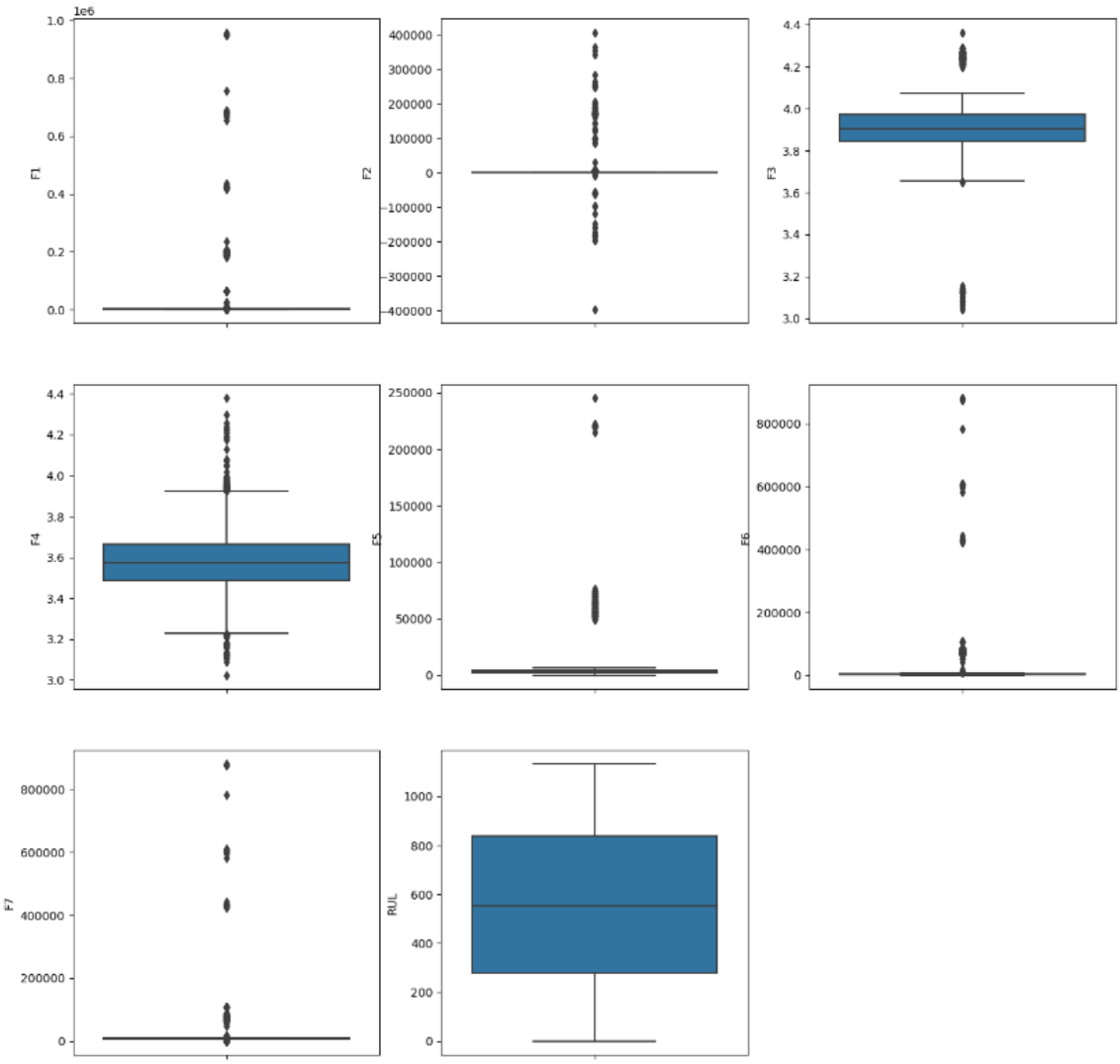

Рисунок 6 – графики boxplot до удаления выбросов

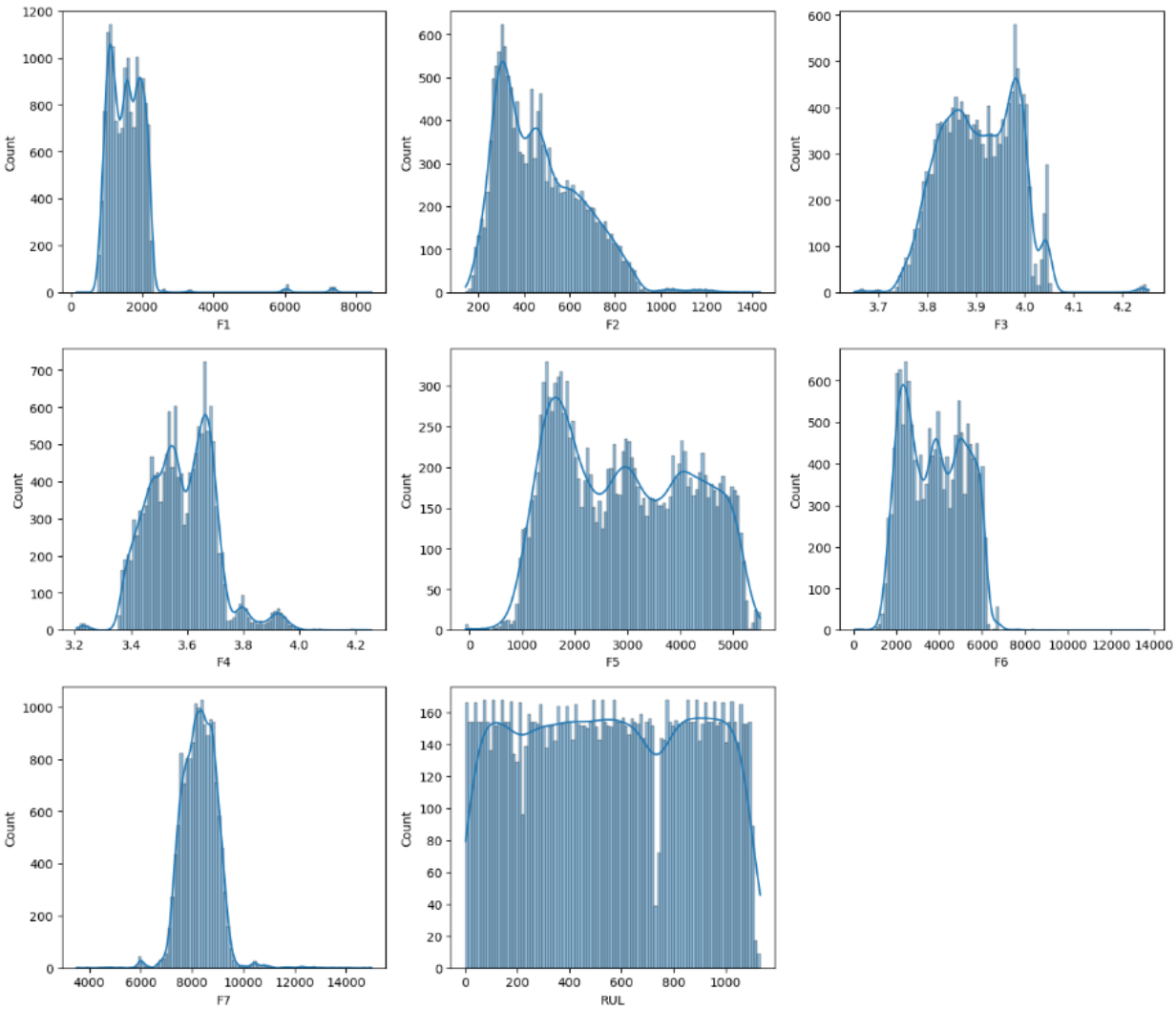

Рисунок 7 – распределения после удаления выбросов

Сравним ковариационные матрицы до и после удаления выбросов. Видно, что удаление выбросов радикально улучшило корреляции. Если до удаления выбросов признаки F1, F2, F5, F6 и F7 являлись фактически константами и их корреляция с целевой переменной лежала в диапазоне 1–18%, то после удаления корреляции лежат в диапазоне 73–98%.

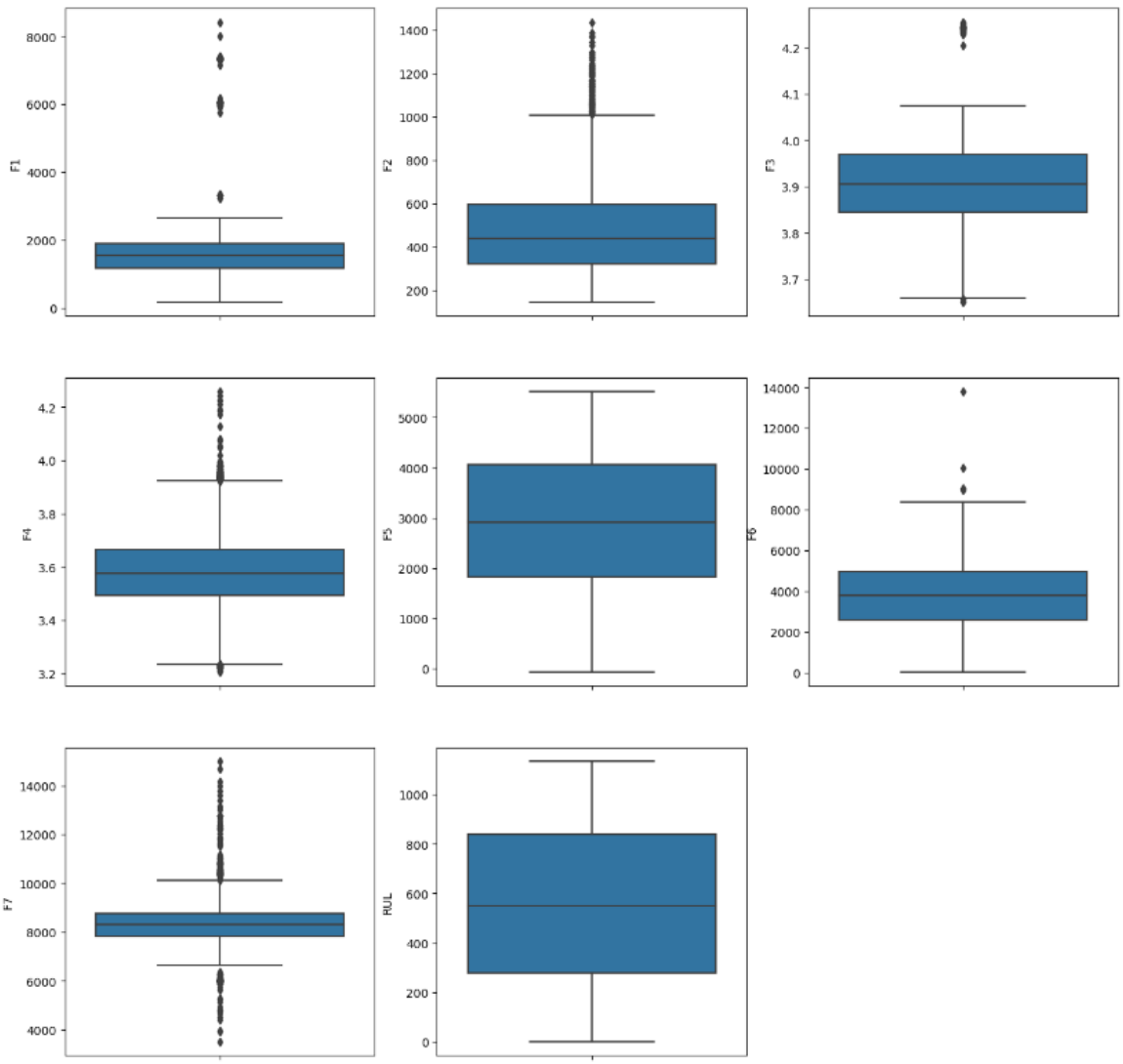

Рисунок 8 – графики boxplot после удаления выбросов

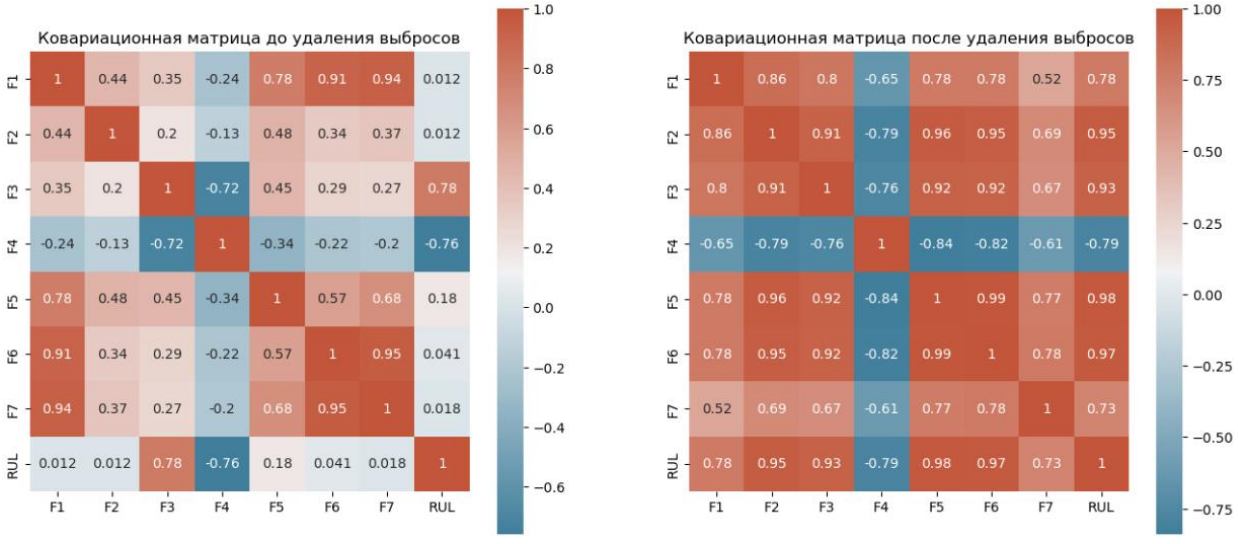

Рисунок 9 – ковариационные матрицы

Также мы наблюдаем высокую корреляцию между признаками, что естественно приводит нас к анализу главных компонент.

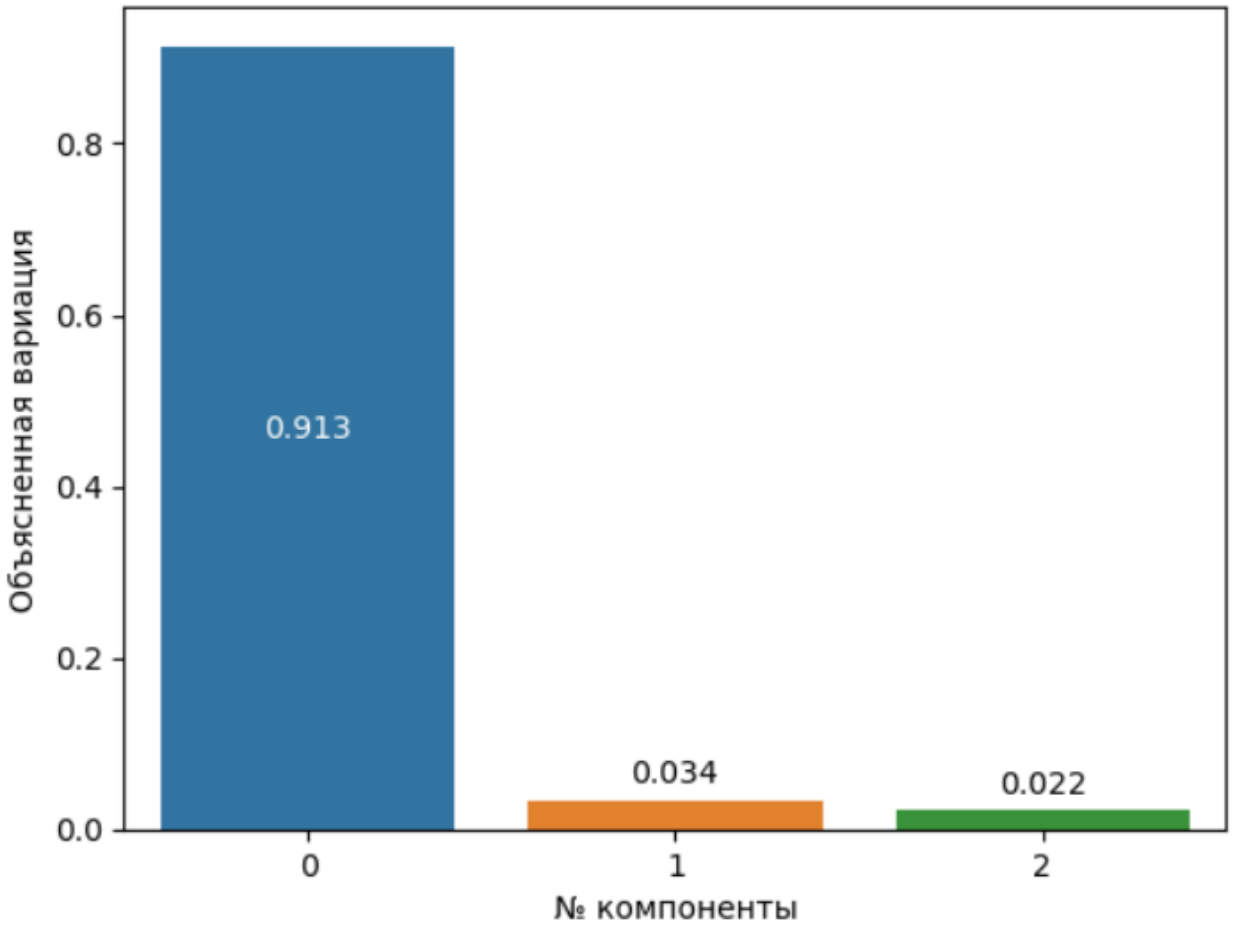

Рисунок 10 – метод главных компонент

Анализ главных компонент показывает, что датасет представляет собой практически одномерное многообразие в многомерном пространстве признаков, и что признаки в значительной степени избыточны.

4. Методика исследования

Для исследования датасета мы будем применять классические регрессоры, ансамблевые методы и полносвязную нейронную сеть.

Из библиотеки Scikit-learn:

- Линейная регрессия
- Ridge (L_2 – регуляризация, или регуляризация Тихонова)
- Полиномиальная регрессия
- Метод k-ближайших соседей

- RandomForest
- Градиентный бустинг

Из библиотеки TensorFlow:

- Полносвязная нейронная сеть

Замечание: включение Ridge-регрессора обусловлено высокой корреляцией признаков. В ситуациях высокой корреляции рекомендуется применять методы регуляризации: Lasso (L_1-регуляризация) либо Ridge (L_2-регуляризация), причем Lasso с байесовской точки зрения соответствует априорному распределению Лапласа для коэффициентов регрессии, а Ridge – нормальному распределению [3].

Для каждого регрессора и каждого $n \in [1, 7]$ проводится подбор n признаков, оптимальных по критерию R^2, из всех наборов из n признаков. На полученных оптимальных наборах проводится кросс-валидация и по ее результатам – усреднение R^2 и MAE. Таким образом, для каждого цикла исследования мы получаем следующий выходной набор данных:

- 7 наборов признаков, оптимальных по R^2
- 7 соответствующих значений R^2
- 7 соответствующих значений MAE

Заметим, что оптимизация производится по метрике R^2, поэтому значения MAE не обязательно будут оптимальными и должны рассматриваться как значения, соответствующие оптимальным значениям R^2. Из указанного выше набора сохраняются следующие значения:

- Лучшее значение R^2 из 7 полученных
- Лучшее значение MAE из 7 полученных
- Значения R^2, полученные при кросс-валидации на одном, двух и трех признаках, и соответствующие им значения MAE. Интерес к данным параметрам вызван сильной корреляцией между признаками и естественно возникающем при этом желанием оценить разброс качества модели на минимальных и максимальных наборах признаков.

– всего 8 параметров для каждого полного цикла подбора параметров и кросс-валидации.

Для линейной регрессии и Ridge-регрессии проводится по одному циклу. Для полиномиальной регрессии предварительно создаются дополнительные признаки до 7-й степени включительно, и также проводится один цикл. При этом оказывается, что лучшими признаками все равно остаются первые степени F5 и F6.

Для регрессора kNN проводится 5 циклов с количеством соседей от 2 до 6, при этом оптимальным оказывается цикл с 3 соседями.

Для RandomForest проводится 5 циклов с количеством деревьев от 60 до 1000, при этом оптимальным оказывается максимальное число деревьев, но с крайне незначительным перевесом.

Метод k-ближайших соседей

kNN с количеством соседей от 2 до 6:

```
In [40]: kNN1 = apply_regressor(KNeighborsRegressor(n_neighbors=2), df_X, dfc.RUL)
```

```
R2       RMSE    MAE     best features
-----------------------------------------------
0.9679   57.65   40.09   [1]
0.9729   52.92   35.23   [1 6]
0.9761   49.69   32.82   [1 5 6]
0.9862   37.75   20.74   [1 4 5 6]
0.9925   27.80   12.36   [1 2 4 5 6]
0.9950   22.66   8.03    [1 2 3 4 5 6]
0.9963   19.46   6.11    [ALL]
```

```
In [41]: kNN2 = apply_regressor(KNeighborsRegressor(n_neighbors=3), df_X, dfc.RUL)
```

```
R2       RMSE    MAE     best features
-----------------------------------------------
0.9711   54.65   38.35   [1]
0.9759   49.95   33.74   [1 6]
0.9781   47.60   32.26   [1 5 6]
0.9866   37.16   21.78   [1 4 5 6]
0.9926   27.64   13.35   [1 2 4 5 6]
0.9952   22.23   8.76    [1 2 3 4 5 6]
0.9965   19.10   6.59    [ALL]
```

Рисунок 11 – циклы подбора признаков

Для градиентного бустинга предварительно проводится подбор оптимального значения параметра learning_rate, и затем 5 циклов кросс-валидации в окрестности этого оптимального значения.

Для всех регрессоров, кроме нейронной сети, подбор признаков производится с помощью метода SequentialFeatureSelector.

На рисунке 11 приводится пример двух циклов подбора признаков.

5. Нейронная сеть

Для нейронной сети была выбрана архитектура с двумя полносвязными скрытыми слоями. Опытным путем был подобран размер слоев (по 50 нейронов), оптимизатор – Adam, который оказался заметно лучше остальных (за исключением разве что Adamax) и функция активации relu, которая показала себя лучше первоначально планировавшегося tanh.

При подборе признаков и кросс-валидации модель перекомпилировалась перед каждым запуском, чтобы исключить продолжение обучения сети, уже обученной на предыдущих циклах кросс-валидации.

6. Сравнение регрессоров

В результате проведения процедур подбора признаков получается следующая сводная таблица и графики параметров R^2 и MAE.

Таблица 1 – Сводная таблица оценки регрессоров

	R2_best	R2_1_feature	R2_2_features	R2_3_features	MAE_best	MAE_1_feature	MAE_2_features	MAE_3_features
Linear	0.965325	0.955305	0.959547	0.963458	41.890696	42.862455	41.890696	42.709987
Ridge	0.965053	0.955281	0.959377	0.963327	41.688289	42.879856	41.688289	42.694885
kNN	0.996457	0.971106	0.975873	0.978093	6.592141	38.345391	33.741462	32.260986
RandomForest	0.996642	0.969702	0.978442	0.981426	9.16303	38.430279	31.977396	29.740954
Polynomial	0.966741	0.955305	0.959547	0.963458	41.890696	42.862455	41.890696	42.709987
GradientBoosting	0.990813	0.975557	0.9786	0.978744	21.365446	35.433984	33.701959	33.727206
DenseNN	0.978636	0.975476	0.975248	0.977382	34.895249	36.38265	39.194288	34.896182

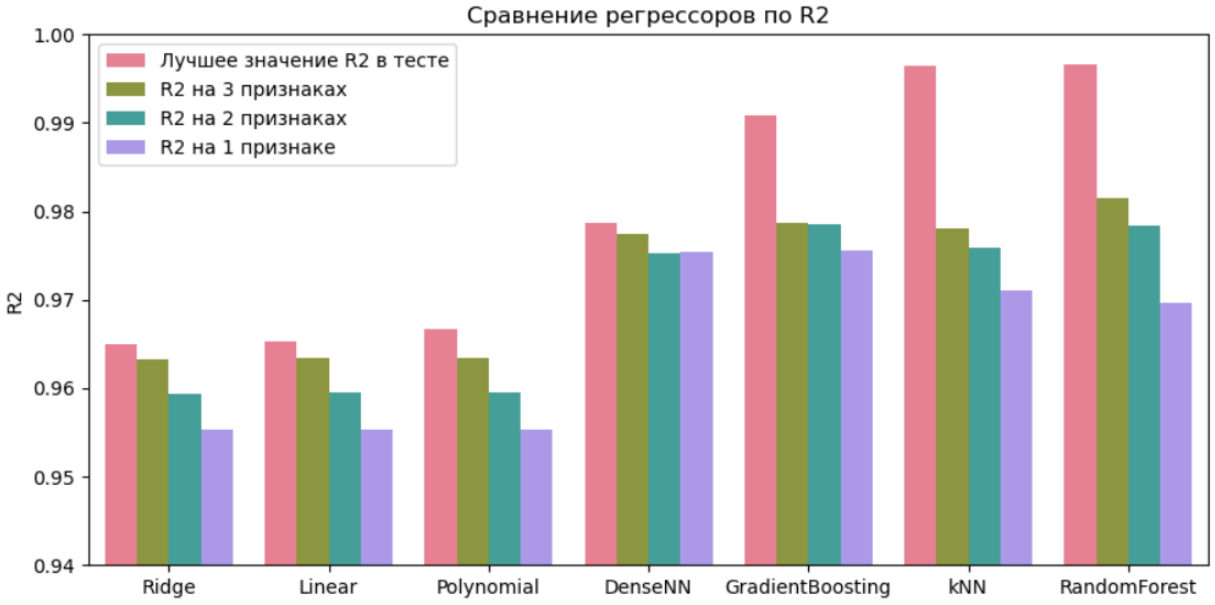

Рисунок 12 – сравнение регрессоров по R^2

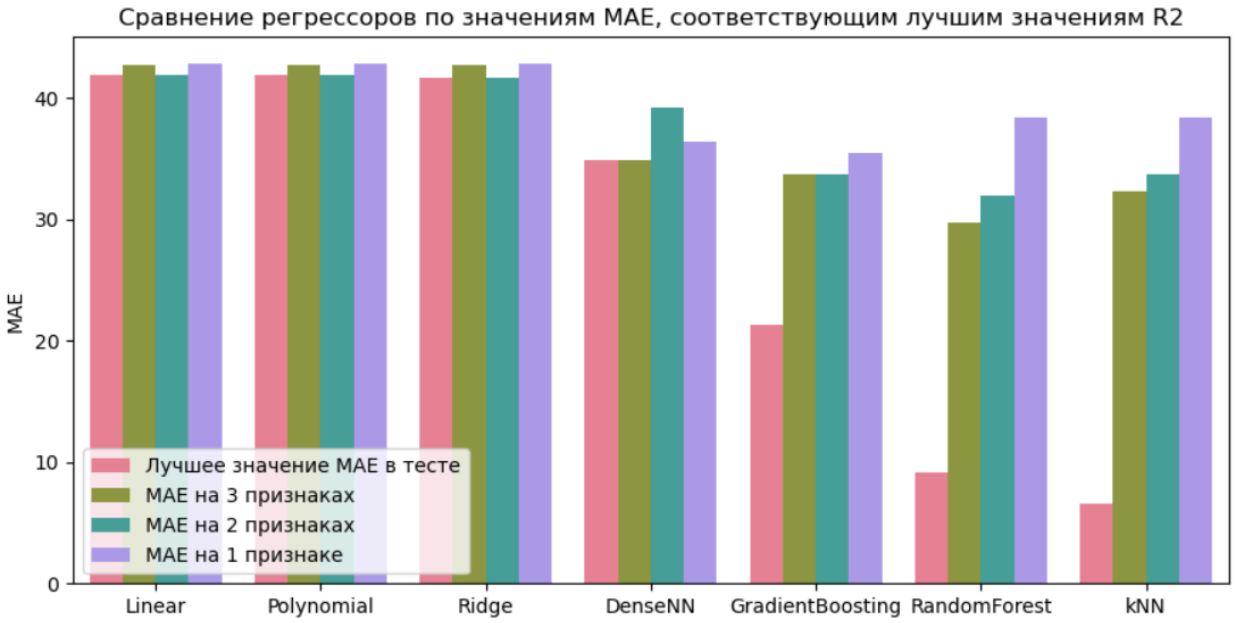

Рисунок 13 – сравнение регрессоров по MAE

7. Оптимальные наборы признаков

Интересным наблюдением является то, что регрессоры (за исключением нейронной сети) разделились на две группы по тому, какие наборы признаков они сочли оптимальными – см. рис. 14.

Регрессор	Линейная регрессия	Ridge	Полиномиальная регрессия
Оптимальные наборы признаков	[5] [4 5] [3 4 5] [3 4 5 7] [3 4 5 6 7] [2 3 4 5 6 7]	[5] [4 5] [3 4 5] [3 4 5 7] [3 4 5 6 7] [1 3 4 5 6 7]	[5] [4 5] [3 4 5] [3 4 5 10] [2 3 4 5 10] [2 3 4 5 9 10] [2 3 4 5 6 9 10] [1 2 3 4 5 6 9 10] [1 2 3 4 5 6 8 9 10]

Регрессор	RandomForest	kNN	Градиентный бустинг
Оптимальные наборы признаков	[1] [1 6] [1 5 6] [1 3 5 6] [1 2 3 5 6] [1 2 3 4 5 6]	[1] [1 6] [1 5 6] [1 4 5 6] [1 2 4 5 6] [1 2 3 4 5 6]	[1] [1 6] [1 5 6] [1 4 5 6] [1 3 4 5 6] [1 2 3 4 5 6]

Рисунок 14 – Оптимальные наборы признаков

В то же время, у нейронной сети не наблюдалось устойчивых предпочтений к подобным наборам.

Также следует отметить, что у всех регрессоров, за исключением нейронной сети, лучшие значения R^2 наблюдались на максимальном наборе признаков (7), в то время как у нейронной сети это значение варьировалось от 5 до 7 при повторении расчетов.

8. Выводы

- Ключевым моментом в исследовании датасета явилось удаление выбросов. Остается неизвестным, почему автор датасета не произвел подобное удаление.
- Высокая избыточность данных позволяет получать качественные предсказания на малых наборах признаков (что будет использовано при создании приложения).
- Регрессоры разделились на 3 группы: семейство линейных регрессоров, включающее Ridge и полиномиальную регрессию, оказалось чуть хуже нейронной сети, а семейство ансамблевых методов, дополненное kNN, оказалось чуть лучше. При этом разделение произошло не только по качеству предсказаний, но и по набору оптимальных признаков (см. выше).

- Наилучшие показатели демонстрируют регрессоры RandomForest и kNN, их показатели очень близки. При этом вычислительные ресурсы, требующиеся для RandomForest, существенно (на 1–2 порядка) выше ресурсов для kNN. Поэтому лидером данного исследования следует признать регрессор kNN.

9. Приложение

Использование всех 7 признаков для предсказания представляется избыточным. Для приложения построим нейронную сеть, которая будет делать предсказания по 3 признакам F1, F4 и F7, и предсказывать не только целевую переменную, но и остальные 4 признака.

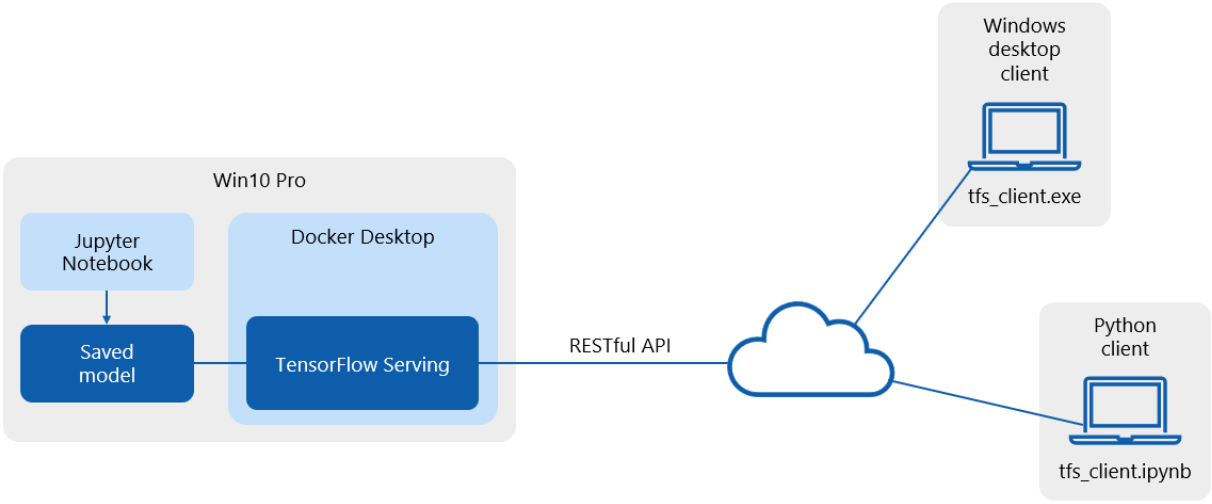

Рисунок 15 – Схема приложения

В приложении используется архитектура «Клиент – сервер», где сервером служит пакет TensorFlow Serving, размещенный в Docker Desktop для Windows, а клиентом – приложение для Windows, написанное на C++ с использованием новейшего фреймворка Microsoft WinUI3 для Windows 10/11. Изображение окна приложения приводится на рис. 16. Для использования приложения необходимо загрузить всю папку по адресу https://github.com/petr-larin/BMSTU-Graduate-Qualifying-Project/tree/main/tensorflow-serving-client-win/release, и, если будет выдано соответствующее сообщение, обновить библиотеку времени исполнения (запустить файл WindowsAppRuntimeInstall.exe).

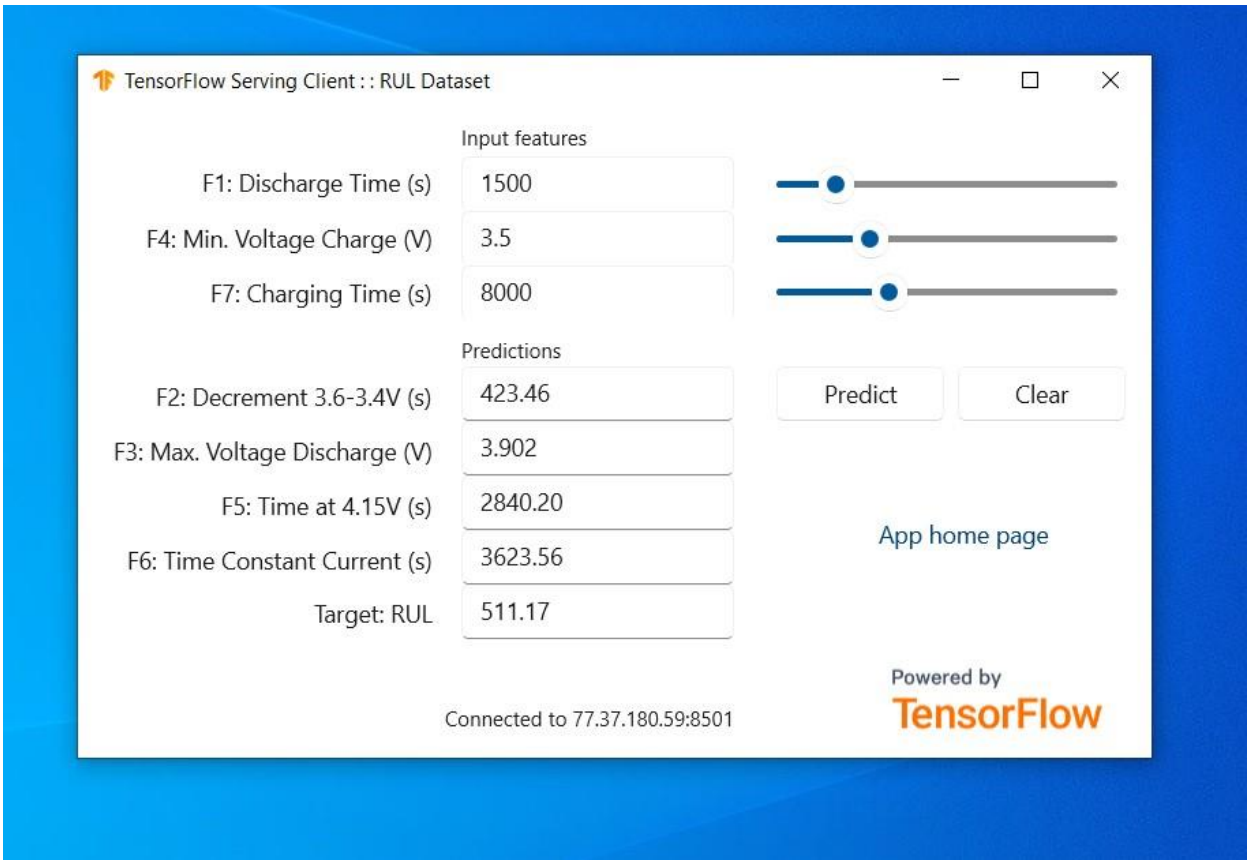

Рисунок 16 – Окно клиента приложения

Также, имеется клиент на основе Jupyter Notebook (файл tfs_client.ipynb), обладающий той же функциональностью, но без графического интерфейса.

Список литературы

[1] *Ignacio Viñuales*. Battery RUL prediction using PyTorch – Режим доступа: https://github.com/ignavinuales/Battery_RUL_Prediction (дата обращения: 25.04.2023).

[2] *Ignacio Viñuales.* Battery Remaining Useful Life (RUL) – Режим доступа: https://www.kaggle.com/datasets/ignaciovinuales/battery-remaining-useful-life-rul (дата обращения: 25.04.2023).

[3] Lasso (statistics) – Режим доступа: https://en.wikipedia.org/wiki/Lasso_(statistics) (дата обращения: 25.04.2023).