Оглавление

Глава 1. Предобработка данных

В предоставленном датасете имеются данные по 2-ум разным линиям. Но проблема заключается в том, что у одной линии есть такие параметры, которых нету у другой. Поэтому было принято решение использовать только данные с линии, по которой известно больше всего характеристик – с 1-ой линии.

В датасете также было количество nan значений, которое не превышало в общей сложности 20 тысяч строк. Так как большинство из этих nan значений связаны с неисправностью прибора – все пропуски были удалены.

Столбец “Время” был удален и заменен на два столбца с целыми значениями: столбец часы и столбец минуты.

Также стоит отметить, что для обучения всех алгоритмов данные были перемешаны, с разделением на тестовые данные с коэффициентом 0.2.

Глава 2. Выбор моделей и работа с ними

Выбор лучших моделей

Для задачи предсказания гранулометрии – в контексте машинного обучения задачи регрессии, было принято решение протестировать на предоставленных данных такие модели как: *XGBoost*, *CatBoost*, *LigthGBM*, *RandomForestRegressor*, *HistGradientBoostingRegressor*, KNN. Результаты тестирования каждой из этих моделей приведены ниже по порядку. Также стоит отметить, что для обучения, тестирования и инференса модели не использовались лабораторные данные, так как по результатам тестирования они увеличивают ошибки на валидационных данных из-за своей маленькой дискретности.

Для наглядности своих тестов я буду вставлять кусочки графиков предсказаний – это будет стабильно один и тот же кусочек, который алгоритм МО во время обучения не видел, но с наложением на него результатов разных моделей (предсказания модели – желтые точки, верные значения – зеленые точки).

- *RandomForestRegressor* из библиотеки *sklearn*.

Метрики на *Train* выборке: *R2 Score* = 0.9992, *MAE* = 0.3183, *MSE* = 1.1708.

Метрики на *Test* выборке: *R2 Score* = 0.9990, *MAE* = 0.3662, *MSE* = 1.3940.

Часть графика предсказаний, представлена на рисунке 1.

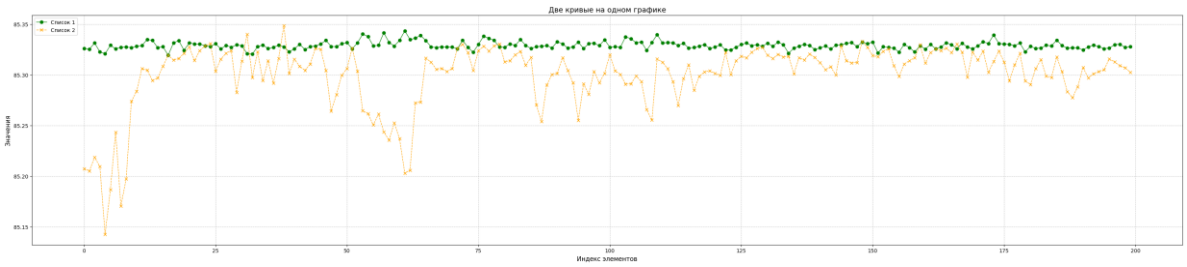

Рисунок 1 – Часть графика предсказаний *RandomForestRegressor*

На рисунке 2 ниже представлено распределение признаков по важности:

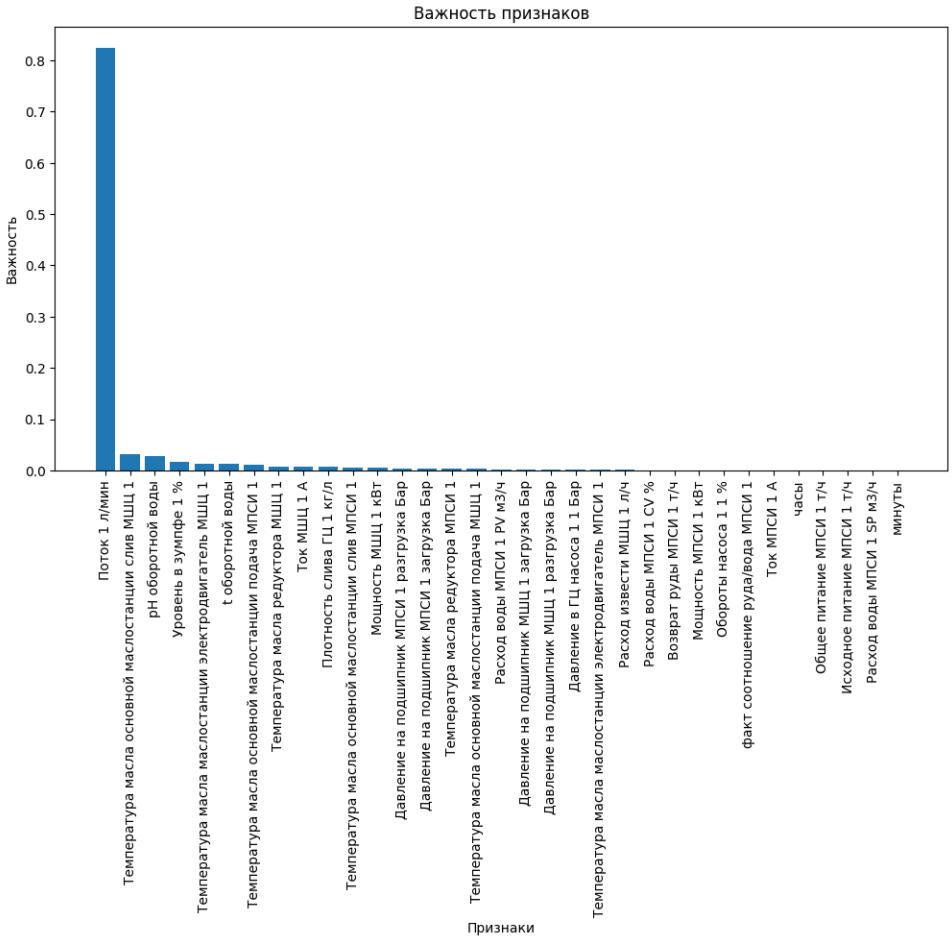

Рисунок 2 – Распределение признаков по важности в *RandomForestRegressor*

- *XGBoostRegressor* из библиотеки *xgboost*.

Метрики на *Train* выборке: *R2 Score* = 0.9999, *MAE* = 0.1542, *MSE* = 0.0808.

Метрики на *Test* выборке: *R2 Score* = 0.9993, *MAE* = 0.3201, *MSE* = 0.9310.

Часть графика предсказаний представлена на рисунке 3 ниже.

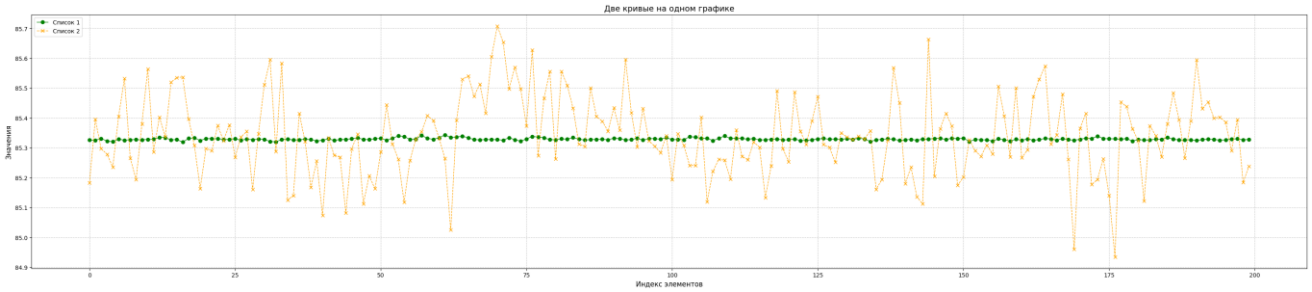

Рисунок 3 – Часть графика предсказаний *XGBoostRegressor*

На рисунке 4 представлено распределение признаков после обучения по важности.

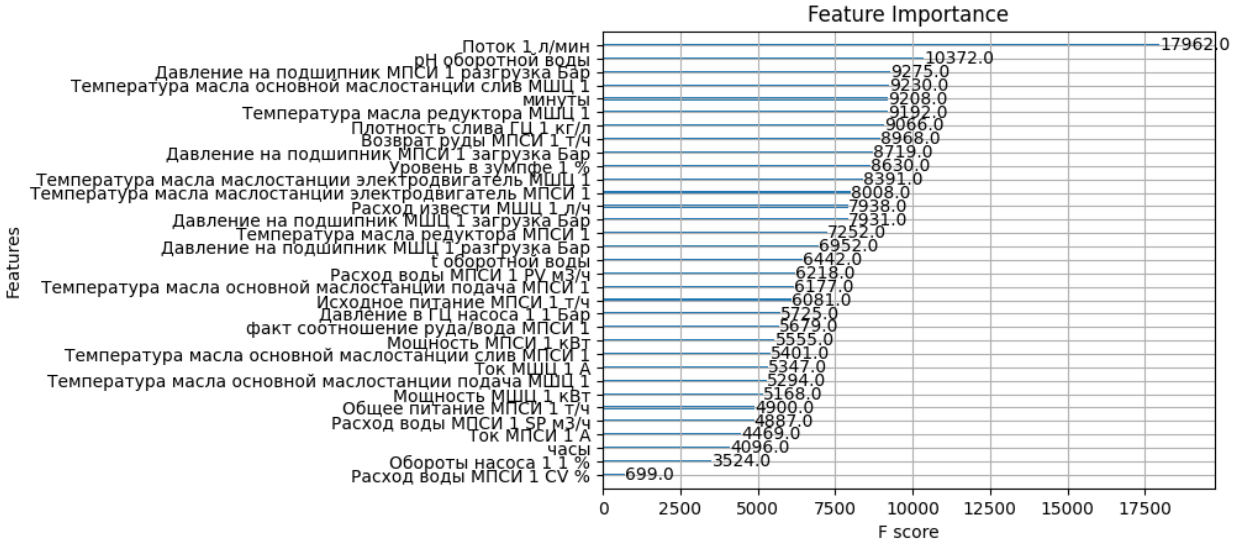

Рисунок 4 – Распределение признаков после обучения по важности

- *CatBoostRegreesssor* из библиотеки *CatBoost*.

Метрики на *Train* выборке: *R2 Score* = 0.9996, *MAE* = 0.3813, *MSE* = 0.6048.

Метрики на *Test* выборке: *R2 Score* = 0.9992, *MAE* = 0.4341, *MSE* = 1.1327.

Часть графика предсказаний представлена на рисунке 5 ниже.

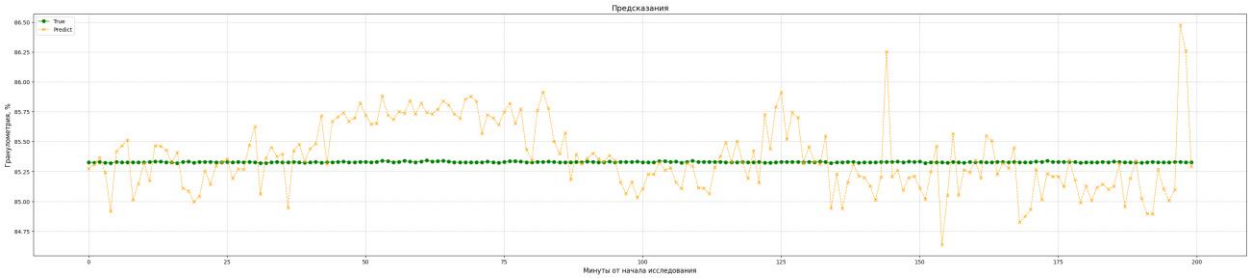

Рисунок 5 – Часть графика предсказаний *CatBoostRegressor*

Ниже на рисунке 6 представлено распределение признаков по важности.

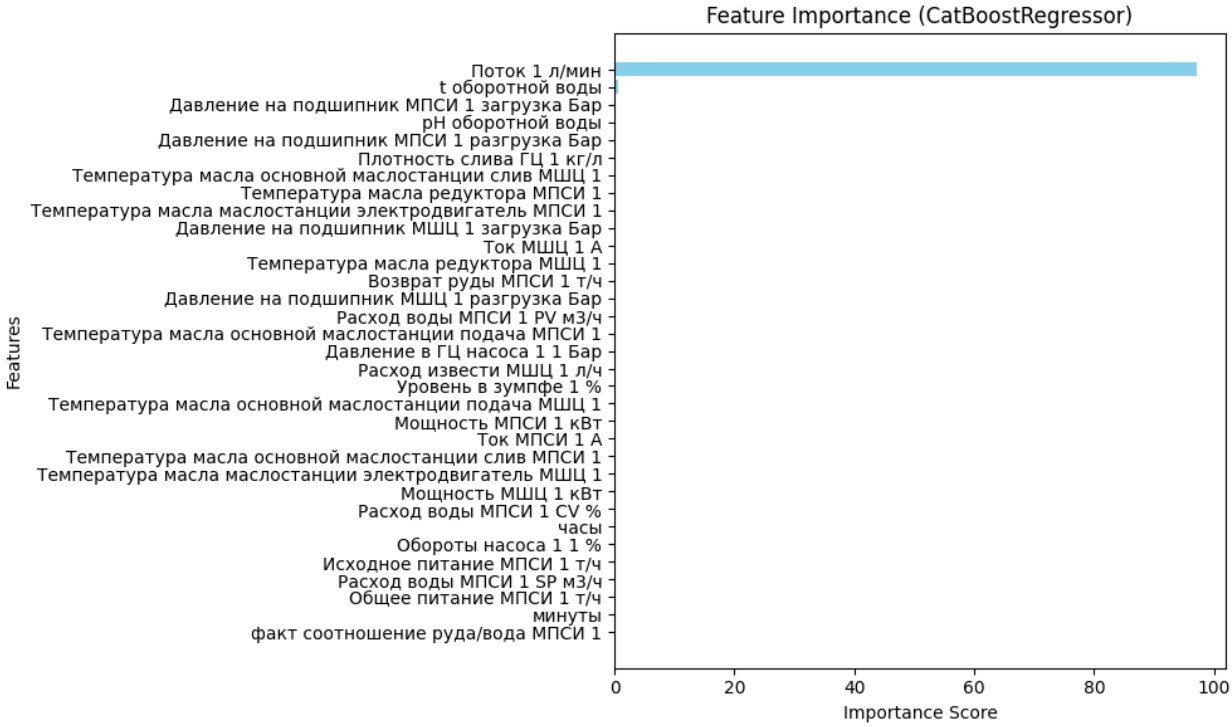

Рисунок 6 – Распределение признаков после обучения по важности

- *HistGradientBoostingRegressor* из библиотеки *sklearn*.

Метрики на *Train* выборке: *R2 Score* = 0.9997, *MAE* = 0.2754, *MSE* = 0.3677.

Метрики на *Test* выборке: *R2 Score* = 0.9993, *MAE* = 0.3868, *MSE* = 1.0138.

Часть графика предсказаний представлена на рисунке 7 ниже.

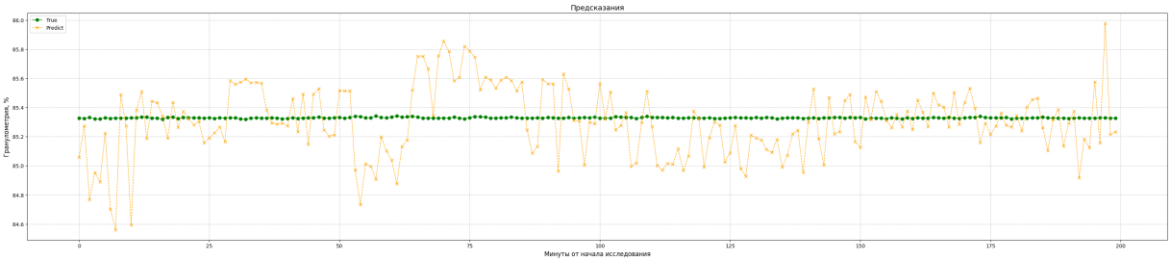

Рисунок 7 – Часть графика предсказаний *HistGradientBoostingRegressor*

Ниже на рисунке 8 представлено распределение признаков после обучения по важности.

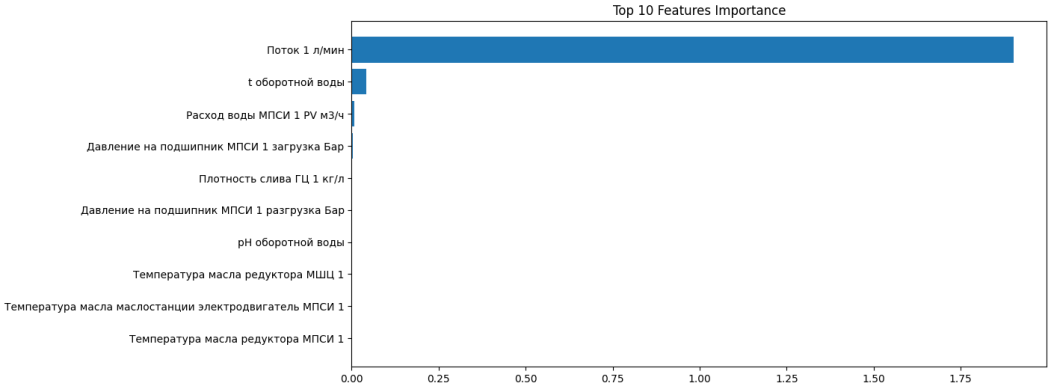

Рисунок 8 – Распределение признаков по важности после обучения

- *LightGBM* из библиотеки *lightGBM*.

Метрики на *Train* выборке: *R2 Score* = 0.9999, *MAE* = 0.0927, *MSE* = 0.0230.

Метрики на *Test* выборке: *R2 Score* = 0.9995, *MAE* = 0.3111, *MSE* = 0.6975.

Часть графика предсказаний представлена на рисунке 9 ниже.

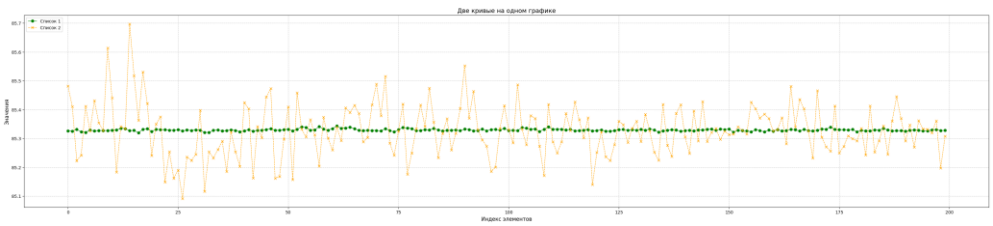

Рисунок 9 – Часть графика предсказаний *LightGBM*

Ниже на рисунке 10 представлено распределение признаков после обучения по важности.

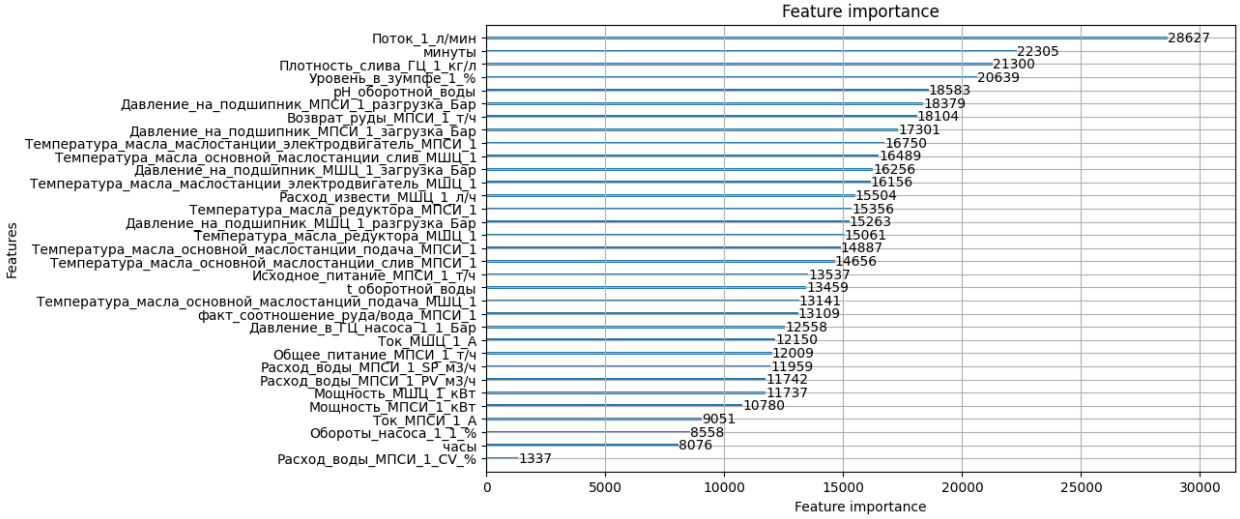

Рисунок 10 – Распределение признаков по важности после обучения

- *KNN* из библиотеки *sklearn*.

Метрики на *Train* выборке: *R2 Score* = 0.9999, *MAE* = 3.27e-07, *MSE* = 5.79e-12.

Метрики на *Test* выборке: *R2 Score* = 0.9971, *MAE* = 0.4722, *MSE* = 4.3742.

Часть графика предсказаний представлена на рисунке 11 ниже.

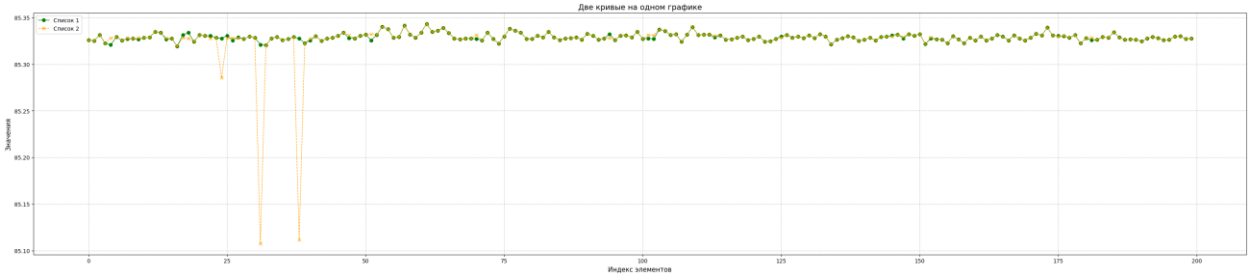

Рисунок 11 – Часть графика предсказаний *KNN*

Вывод: выбирая модели по лучшим метрикам и графику предсказаний, путем тестирования были выбраны лучшие модели для последующего блендинга, а именно: *CatBoost* и *HistGradientBoostingRegressor*.

Блендинг лучших моделей

Итак, если объеденить предсказания *CatBoost* и *HistGradientBoostingRegressor*, а точнее привести их к средним значениям, то получим явно улучшенную метрику *MSE* = 0.9480 с максимально маленьким *MAE* = 0.3650, что обеспечивает лучшую воспроизводимость тренда процентов гранулометрии и точности предсказаний. На рисунке 12 представлена часть графика предсказаний при приведении двух предсказаний разных моделей к среднему числу. Синяя линия – среднее значения предсказаний (лучше всего описывает тренд всего графика).

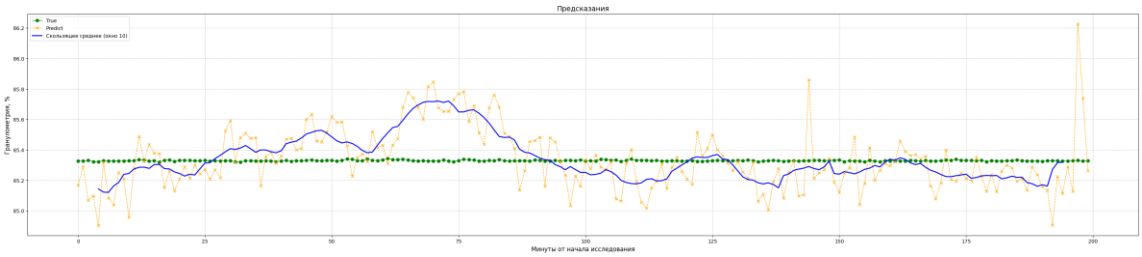

Рисунок 12 – Часть графика предсказаний при приведении двух предсказаний разных моделей к среднему числу

Полная картинка графика предсказания блендинг значений (график на весь промежуток, на который предоставлены данные) будет в репозитории на *GitHub*.

Также было произведено обучение еще с помощью двух способов: с помощью *Tensorflow* (*Sequential API*) и с помощью штабелирования разных вариаций моделей. Но ничего из этого не оказало положительный эффект по сравнению с блендингом.

Глава 3. Бизнесприменяемость

Краткий план внедрения

Для внедрения данной модели машинного обучения на производство, необходимо:

- установить на производстве серверное оборудование для развертывания модели;
- произвести интегрирование *IoT*-датчиков, чтобы данные с них перенаправлялись сразу на сервер, формируя *csv/excel* таблицы или же просто массив данных;
- реализовать скрипт на сервере, который будет подавать на вход модели данные, полученные с датчиков. Модель представляет собой файлы расширения *cbm* и *pkl* где для загрузки в решение первой модели необходимо инициализировать *CatBoostRegressor()*, потом к этому способу применить *.load_model('file_path')*. Для загрузки второй модели необходим метод *load('file_path')* из библиотеки *joblib*. Каждый *load* в обоих случаях нужно поместить в переменную, которая и будет являться обученной моделью.
- около места работы оператора установить сенсорное или другого рода устройство, способное вводить, выводить, получать и передавать данные. На этом устройстве должно быть установлено специализированное ПО или же приложение для отслеживания результатов работы модели, а точнее для отслеживания процентов гранулометрии в реальном времени;
- рассматривая перспективу бизнесприменения, будет весьма полезно реализовать процесс предсказания гранулометрии одновременно для нескольких линий. Для этого будет необходимо докупить серверное оборудование и написать скрипты для серверов, которые одновременно обращаются к модели и вытягивают оттуда предсказания для нескольких линий. Также, в перспективе можно реализовать самоодообучение прогнозной модели за счет того, что она на постоянной основе будет собирать доп. данные.

Макет приложения для оператора

На рисунке 13 представлен макет графического интерфейса, через который будет вестись отслеживание оператором гранулометрии в реальном времени.

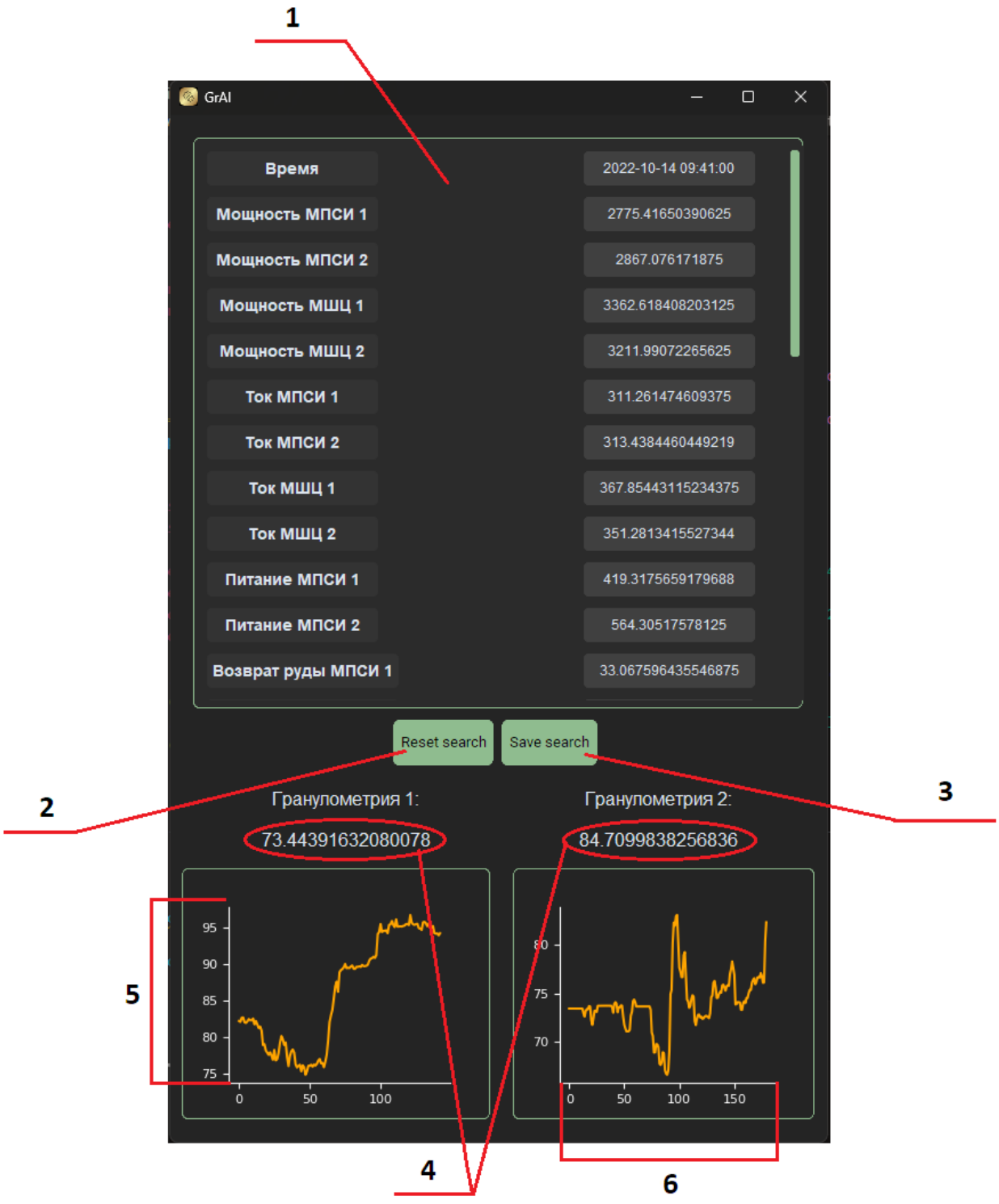

Рисунок 13 – Макет графического интерфейса

На данном макете имеется:

1. Скроллбар со всеми нужными параметрами линии;
2. Кнопка чтобы сбросить текущее исследование, обнулить графики и начать новое;
3. Кнопка сохранения текущих графиков и параметров в один файл;
4. Значения гранулометрии предсказанные на текущую минуту;
5. Ось %-ов гранулометрии;
6. Ось количества минут с начала нового исследования.

Данный макет довольно минималистичен и содержит в себе много важной информации, необходимой для работы с пульпой. Такой подбор цветов не режет глаза, а также все нужные параметры находятся недалеко друг от друга, что позволит оператору не водить глазами по всему экрану в поисках какого-то значения. А значит, исходя из всего этого - при долгой смене усталость на глазах от экрана будет минимальной, что в свою очередь не будет снижать концентрацию оператора. Также в приложении представлена визуализация предсказаний сразу для двух линий – это показывают два графика снизу и двойные параметры в скроллбаре. Ну и немного символики в честь проделанной работы – приложение носит название *GrAI* => Гранулометрия + искусственный интеллект.

Макет создавался с помощью *Python* библиотеки *customtkinter*.