Домашнее задание 1

Реализовать API (REST и процедуры gRPC), которое умеет:

1. Обучать ML-модель с возможностью настройки гиперпараметров. При этом гиперпараметры для разных моделей могут быть разные. Минимальное количество классов моделей доступных для обучения == 2.
2. Возвращать список доступных для обучения классов моделей

Домашнее задание 1

3. Возвращать предсказание конкретной модели (как следствие, система должна уметь хранить несколько обученных моделей)

4. Обучать заново и удалять уже обученные модели

5. Отдельный эндпоинт для проверки статуса сервиса

6. Отдельный эндпоинт для получения списка загруженных датасетов

6. Интерактивный дашборд streamlit/gradio/dash

Дополнительные требования

1. Логгирование: все важные действия должны логгироваться через логгер
2. Доккументирование: должны быть не только сваггер и докстринги, но и понятный ридми, по которому можно понять, что у вас такое тут слеплено, и как это использовать
3. Для gRPC необходимо приложить скрипт или ноутбук с клиентом (для проверки вашего кода)
4. gRPC может запускать отдельным сервисом/скриптом, просто инструкция по его запуска должна быть в ридми

Дополнительные требования

- 5. Сервис + дашборд запускаются в миникубе
- 6. У вас должен быть Makefile/аналог для запуска всего этого дела одной командой локально в миникубе с драйвером=docker
- 7. Дашборд должен «тыкаться» в эндпоинты вашего сервиса:
  - Отдельная вкладка/таб/вьюшка для работы с датасетами – что посмотреть, удалить, загрузить (датасеты загружаются в json либо csv), выбирать должен из dvc датасетов, а модели – из Clearml моделей (про это след слайд),
  - Отдельная – для инференса
  - Отдельная – для обучения (гиперпараметры json)

Дополнительные требования

- 8. gRPC может запускать отдельным сервисом/скриптом, просто
- инструкция по его запуска должна быть в ридми
- 9. Датасеты должны версионироваться через dvc
- 10. Каждое обучение модели – отдельный эксперимент в clearml, а модель – отдельная модель в clearml
- 11. Веса и кэш хранятся на s3 (minio в том же minicube, загрузка осуществляется через cleaml и dvc)
- 12. ClearML – можно просто рядом в docker compose поднять, не обязательно в minicube его тащить тоже

Общий

- Таким образом у вас будет:
- * Ваш сервис + дашборд + minio в миникубе (+Clearml для доп баллов)
- * Датасеты версионируются через dvc, кэш на персистентном волюме + на s3 (dvc push)
  * Модели после обучения кладутся через ClearML в s3, для инференса берутся из ClearML
- * Дашборд – ваш фронт
- * Makefile и readme для запуска всего этого дела

Оценка

- [5 баллов] Работоспособность программы - то что ее можно запустить и она выполняет задачи, перечисленные в требованиях.
- [3 балла] Корректность и адекватность программы - корректная обработка ошибок, адекватный выбор структур классов, понятная документация (docstring-и адекватные здесь обязательны)
- [2 балла] Стиль кода - соблюдение стайлгайда.
- [2 балла, доп] – ClearML поднимается тоже в minicube

Дополнительные нюансы

- Принимать буду ссылкой на репозиторий (гитхаб, гитлаб, etc)
- Зависимости – фиксируйте
- Данные подаются вместе с запросом на отдельный эндпоинт
- Когда готовы сдавать – делаете MR в main либо просто ссылка на репозиторий

Дополнительные нюансы

- Можно будет поправить или обжаловать, на что укажу, до конца курса
- До дедлайна сдачи необходимо предоставить рабочий сервис, который я смогу запустить, и в котором будет хотя бы один эндпоинт.
- Дедлайны:
- Сдача ДЗ – **до 8:00 (8 утра) 30.11**