

В работе было осуществлено написание парсеров для сбора данных, в необходимой связке: решение суда первой инстанции + результат рассмотрения апелляционной жалобы (download.ipynb, parse\_all.ipynb). Были разного рода сложности с разными сайтами - источниками данных, в том числе проблемы с выкладыванием решений судов без уникального идентификатора дела, что слишком затрудняло процесс поиска апелляционного решения по делу первой инстанции, или же наоборот, апелляционного определения по делу нижестоящей инстанции. Необходимые данные были собраны с сайта <https://www.mos-gorsud.ru/>, в количестве около 100000 объектов. Страницы, содержащие информацию о первой и второй инстанции отдельного дела, были распарсены и сохранены в формате csv для удобства дальнейшего их исследования и аппробации различных моделей машинного обучения.

Данные лежат в [Yandex Object Storage](https://cloud.yandex.ru/services/storage). Синхронизация данных делается через [dvc](https://dvc.org) (mosgorsud00.csv.dvc, mosgorsud01.csv.dvc, mosgorsud02.csv.dvc, mosgorsud03.csv.dvc, mosgorsud04.csv.dvc)

Бинарная классификация длинных текстов (предсказание отмены решения суда) осуществлялась с помощью логистической регрессии. Для этого тексты решений суда первично были очищены от цифр, дат, и одиночных букв (содержащихся, например в Фамилия И.О. или Ф.И.О.), затем производилась лемматизация, далее данные обрабатывались с использованием tfidfvectorizer, с параметрами, подобранными с условием - максимальное качество обученной модели при размере файла обученного tfidfvectorizer не более 1Гб. Минимальная цель – преодолеть значения метрики на константной модели, у которой ROC AUC = 0.5 (save\_clean\_data.py, save\_sparse\_data.py, version\_1.ipynb)

Метрика ROC AUC на тестовых данных была получена 0.72

Взаимодействие с пользователем осуществляется посредством телеграмм-бота @predict\_decision\_bot, в котором реализованы минимальная проверка на предмет является ли присланный объект судебным решением, очистка входных данных, модель предсказания успешной апелляции. Модель реализована как отдельный сервис, в отдельном докер контейнере, через Fast Api. ( serverapp/)

С помощью инструмента airflow настроены еженедельная закачка и обработка новых данных, проверка метрики ROC AUC на закачанных данных для принятия решения о замене модели или дообучении имеющейся. ([mosgorsud\_dag.py](https://github.com/Medvezhik/annulment_court_decision/blob/main/mosgorsud_dag.py))

На новых данных ROC AUC была получена 0.72, модель показывает себя довольно устойчивой.