

RecSys

рекомендации фильмов

Авторы проекта – студенты МОВС:

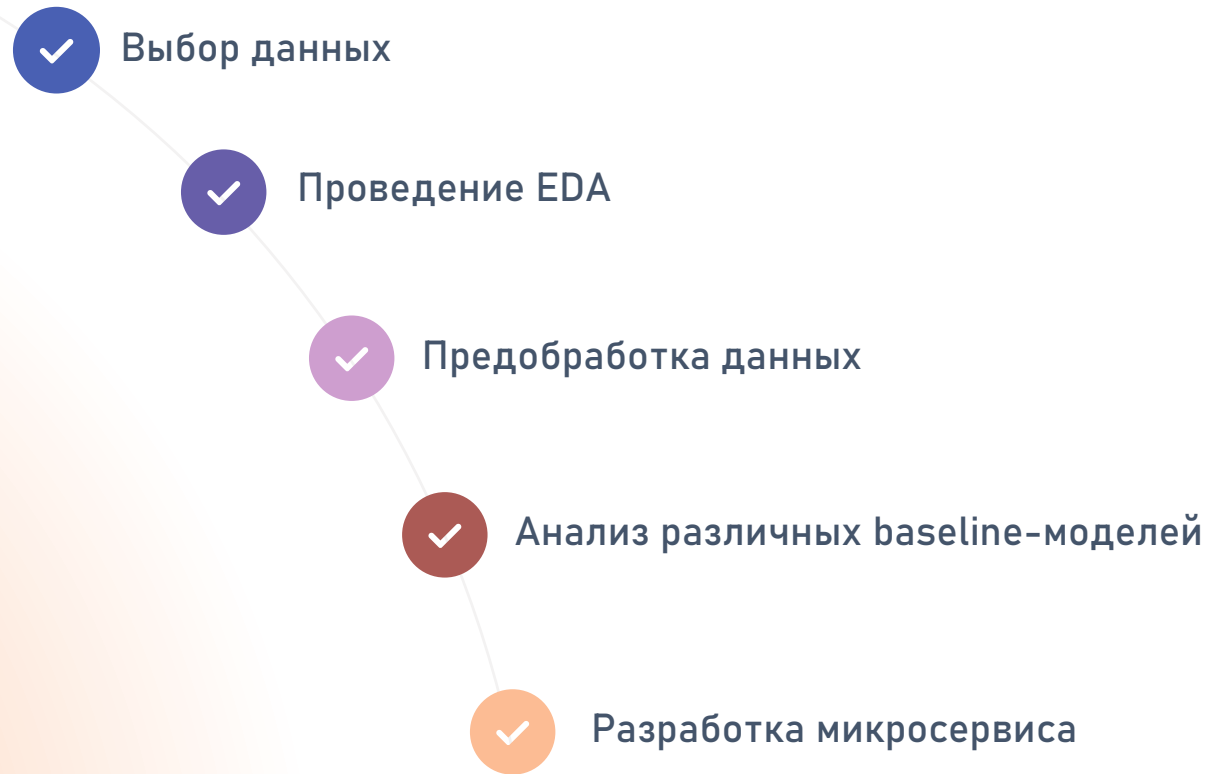


Владислав Панфиленко



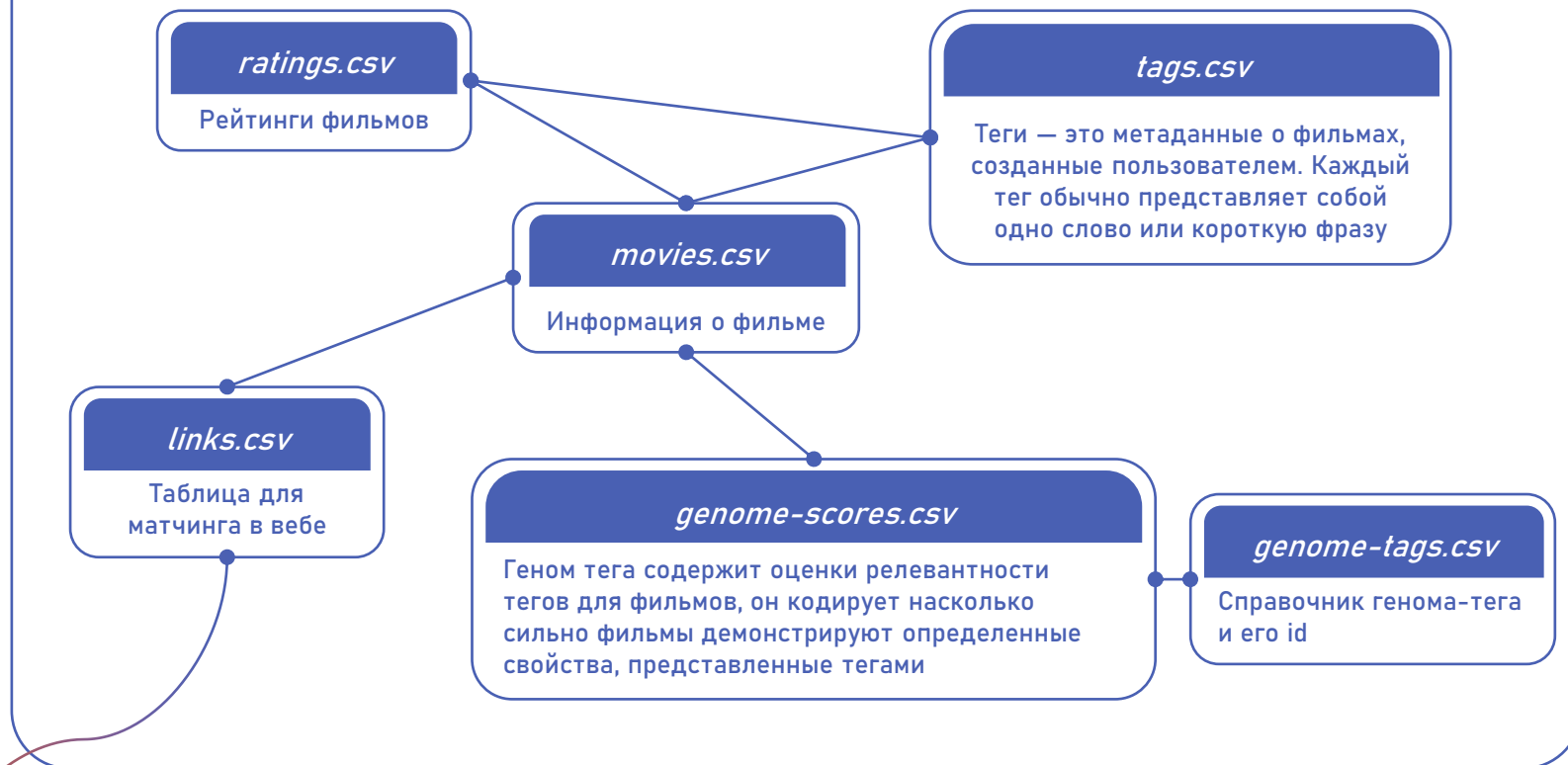
Лилия Хорошенина

Что было сделано?

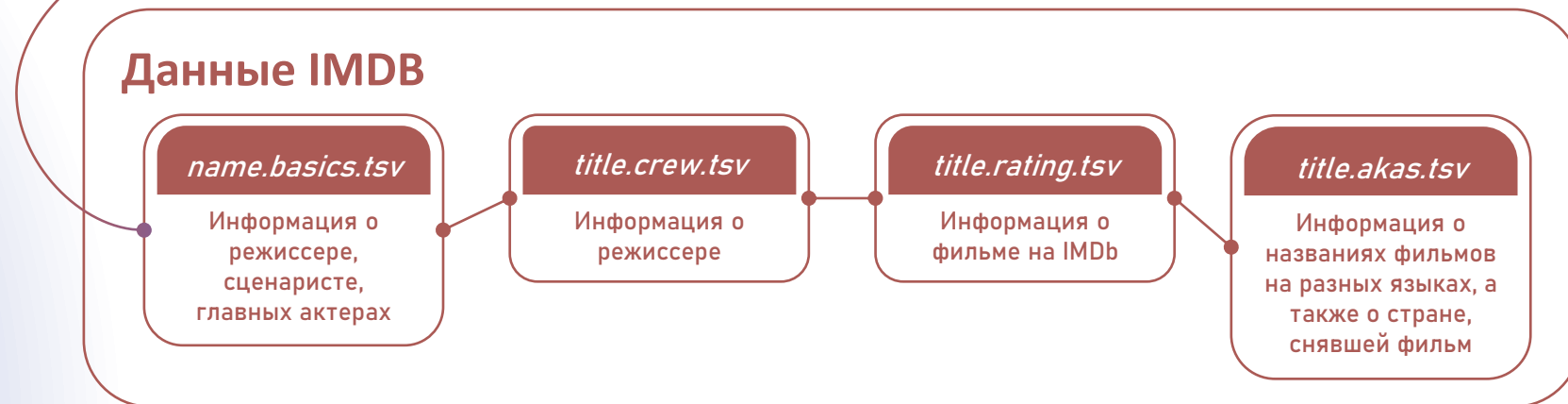


Данные

Данные MovieLens



Данные IMDb



EDA

~34 млн.
оценок

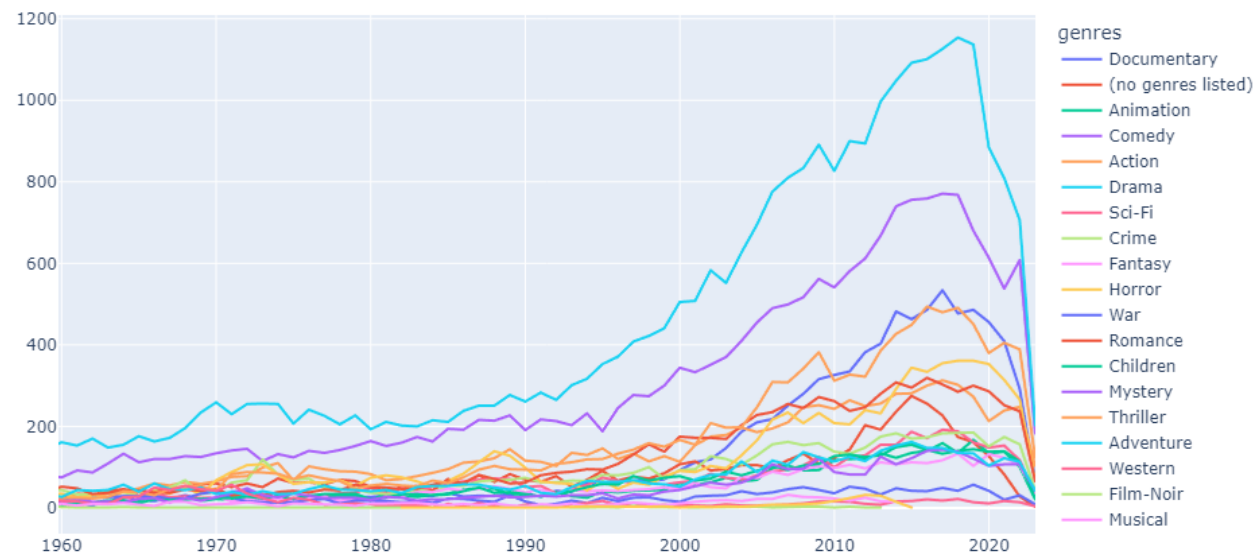
86,5 тыс.
уникальных фильмов
с оценкой

~331 тыс.
уникальных пользователей
поставили оценку

Количество оценок по жанрам, млн.

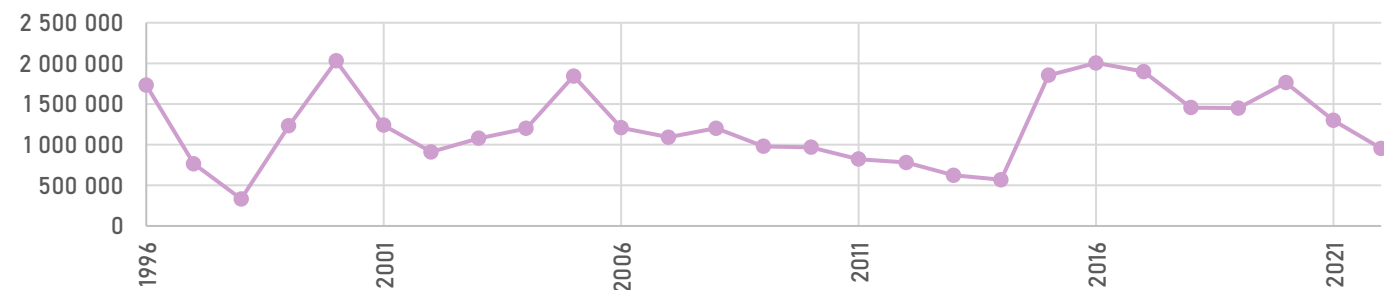


Количество выпущенных фильмов, шт.

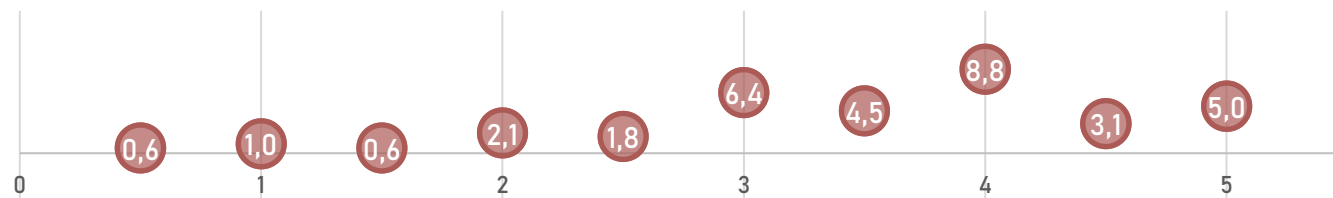


EDA

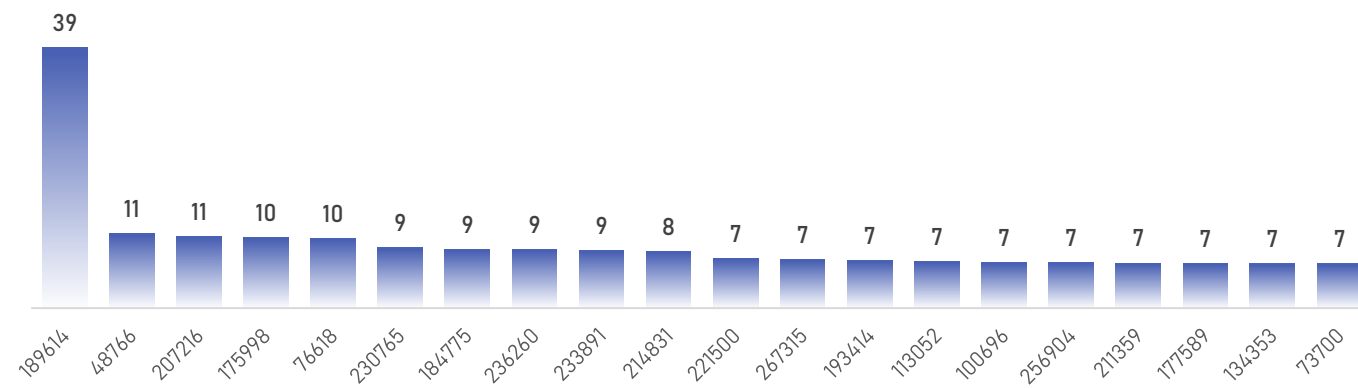
Количество оценок по годам



Динамика оценок пользователей по 5-бальной шкале, млн.



Пользователи, у которых больше всего оценок, в %



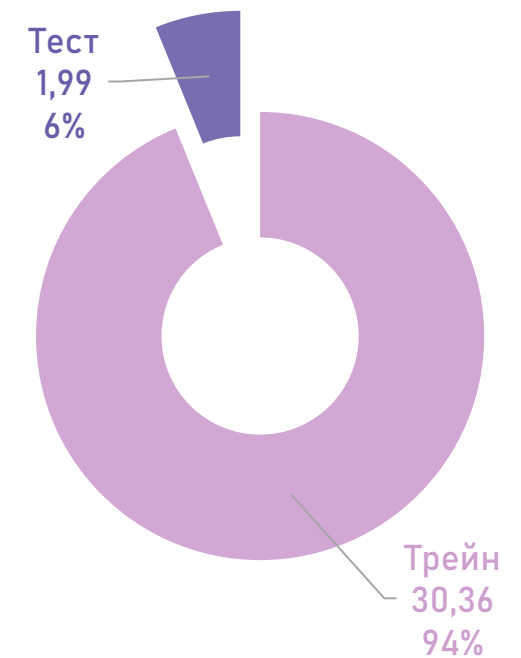
Предобработка данных

Сплит по данным



- Для модели брали **20 последних** оценок пользователя в тест, оставшиеся – в трейн;
- Среди пользователей отбирали только тех, у кого было **более 20 оценок** за все время, что соответствует 0,4 квантилю.

Сплит по данным в млн. строк и % от всех данных



Модель

При разработке модели мы учли, что все пользователи делятся на:

- «холодных» – по которым у нас нет информации;
- «известных» – по которым есть информация.

IMDb-формула
(«холодные» пользователи):

$$w = \frac{Rv + Cm}{v + m}$$

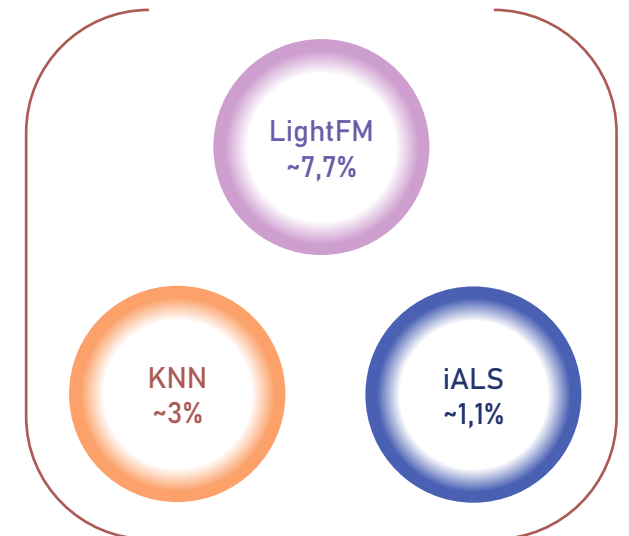
- w – взвешенный рейтинг по фильму
- v – количество оценок по фильму
- m – минимальное количество оценок, необходимое для попадания в число 250 лучших фильмов
- R – средняя оценка фильма
- C – среднее число оценок

Применение ML-моделей
(пользователи с данными):

Нами были рассмотрены 3 основные модели классического ML, подходящие под задачу рекомендательных систем:

- KNN
- iALS
- LightFM

Качество по
MAP@K=10





KNN: rectools + implicit

Для KNN были проанализированы 3 метода сходства между пользователями:

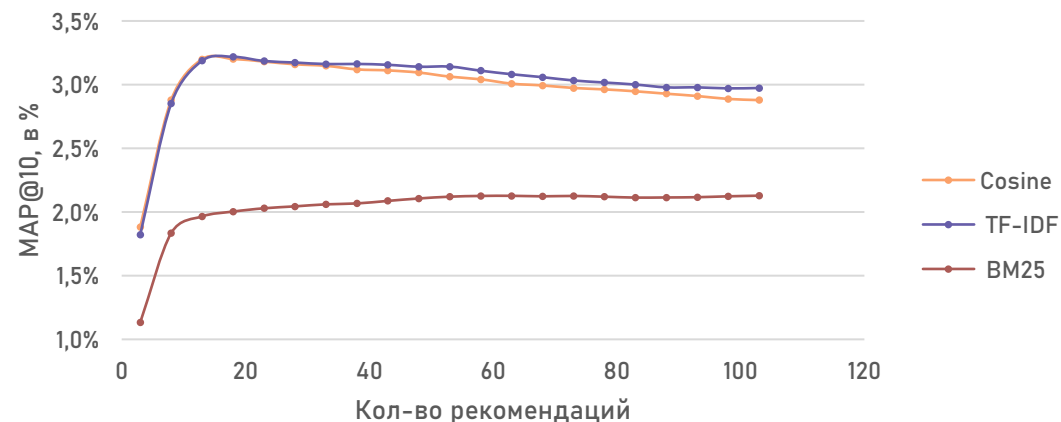
- Косинусное расстояние (Cosine);
- Мера TF-IDF;
- Мера Окари BM25.

Наилучший результат на MAP@10 показал TF-IDF.

Более подробная информация:
[recsys_part1/KNN.ipynb](#)



Качество модели KNN: ~3



iALS: implicit

Для iALS также был произведен подбор гиперпараметров, среди которых были:

- Кол-во факторов (factors);
- Кол-во итераций (iterations);
- Коэффициент регуляризации (regularization);
- Веса положительного класса (alpha).

Более подробная информация:
[recsys_part1/iALS.ipynb](#)



Качество модели iALS: ~1.1

Наилучшими гиперпараметрами были выбраны:

- factors = 3
- iterations=50
- regularization = 1.0
- alpha=4.0



LigthFM (библиотека)

Данная модель показала **наилучшее качество**.

Для начала было проанализировано качество модели на двух **функциях потерь**:

- WARP – взвешенный приближенный ранг;
- BPR – байесовский ранг.

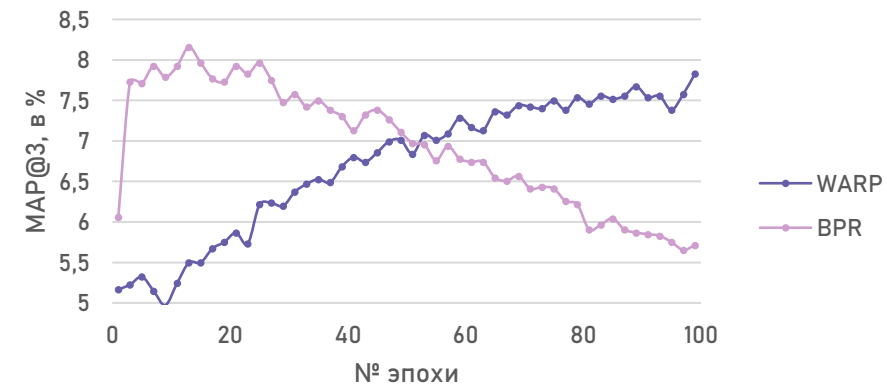
Затем были проанализированы различия в использовании оптимизаторов для обучения:

- Adagrad;
- Adadelata.

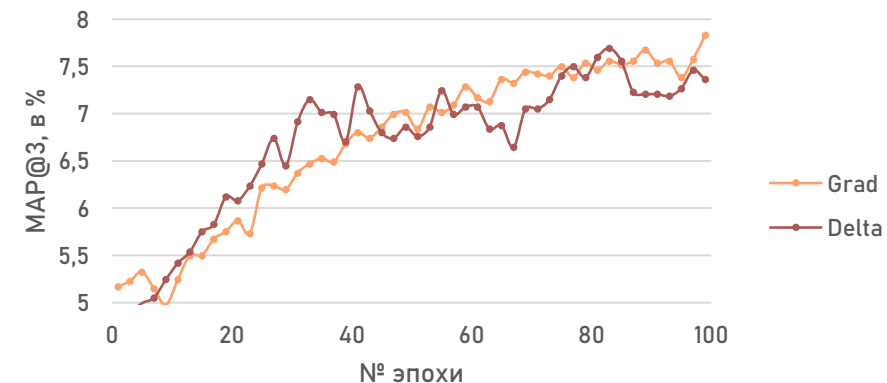
Более подробная информация:

[recsys_part1/LightFM.ipynb](#)

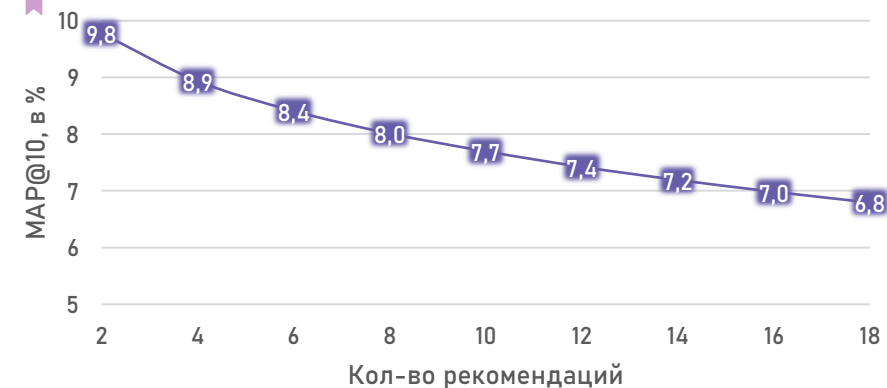
Сравнение loss-функций WARP и BPR



Сравнение оптимизаторов Adagrad и Adadelata



Качество модели на MAP@10: ~7.7



Микросервис telegram - бот

Постарались учесть простой клиентский путь, для этого разработали систему последовательных сообщений, которые учитывают статус нахождения пользователя с помощью системы конечного автомата FSM.

I. Авторизация

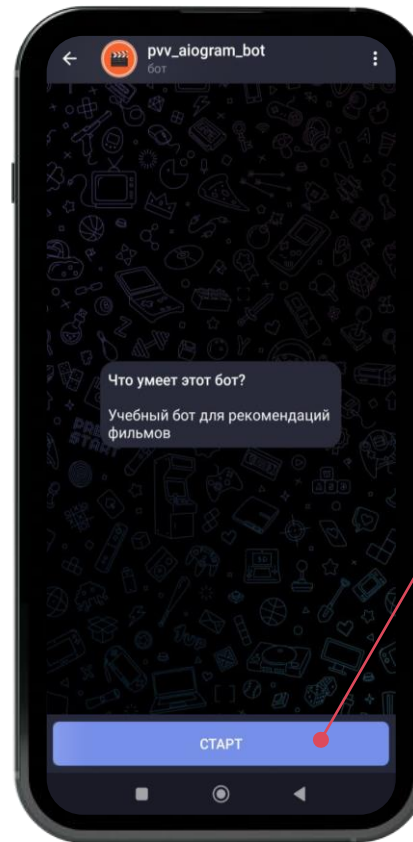


II. Основной функционал

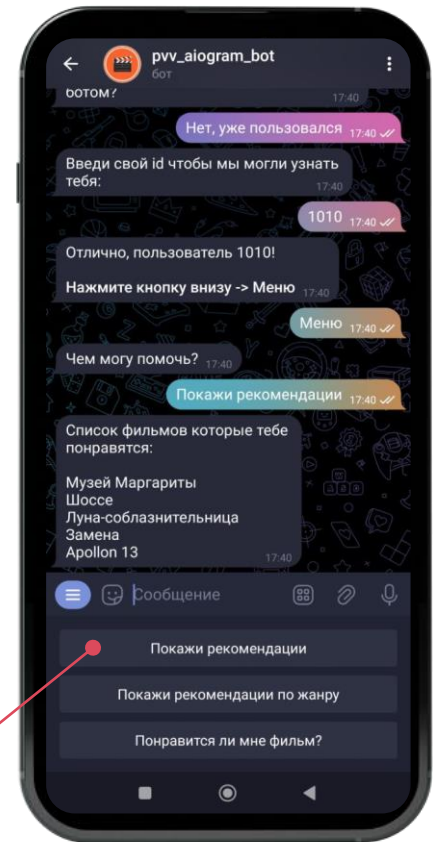


Пример работа бота

Постарались сделать удобство
пользования интуитивно
понятным

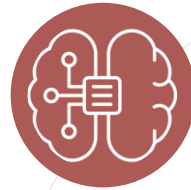


Для начала
работы бота
необходимо
нажать на
«Старт»



Далее
пользователь
пользуется
кнопками
снизу

1. Использование более сложных моделей с применением нейросетевых подходов



2. Использование расширенной информации при моделировании



Дальнейшие планы

4. Доработка микросервиса (БД, логирование, расширение функционала взаимодействия)



3. Реализация механизма валидации в модели

