Вылегжанин Евгений.

Краткий отчет по вступительному испытанию в команду VK CoreML.

Основная часть (то, что указано в задании).

1. Метрики качества.

В качестве метрик для оценки моделей я выбрал несколько метрик:

- Precision@k

На стандартную метрику “Precision” не влияет порядок рекомендованных элементов. Релевантный элемент в топе списка и в конце будут оценены одинаково, но обычно рекомендательные системы создают упорядоченный список рекомендаций для пользователя. Исправить эту проблему можно с помощью precision@k – метрика precision, рассчитанная на подмножестве рекомендованных объектов с ранга 1 по ранг k. С помощью precision@k рассчитывается метрика Average precision.

- Recall@k

Аналогично precision@k – recall, рассчитанная на подмножестве рекомендованных объектов с ранга 1 по ранг k.

- F1-score

Выбрал эту метрику, чтобы оценивать две предыдущие метрики одновременно.

- Average Precision
- Метрика Average Precision учитывает порядок релевантных элементов в списке рекомендаций. Метрика рассчитывается как среднее значение precision@k на релевантных элементах (!).

Как оценивалась модель:

Для каждого пользователя выделялся тестовый набор данных. Например, 30 взаимодействий (фильмов с рейтингом). Из 30 взаимодействий модель создает какое-то количество лучших рекомендаций, например 10. Затем из этих 10 фильмов определяется количество и порядок релевантных рекомендаций (рекомендации считаются релевантными, если истинный рейтинг, поставленный пользователем, больше, чем его средний рейтинг по всем фильмам). Последним этапов рассчитываются метрики.

2. Способ разбиения данных на обучение и валидацию.

Для каждого пользователя данных о взаимодействиях сортируются в хронологическом порядке по timestamp признаку. Далее из конца списка берется определенное количество записей для валидационной и тестовой выборок. Остальная часть относится к тренировочной части. Таким образом, сохраняется порядок действий пользователя.

3. Для реализации задания я использовал библиотеки Surprise (для коллаборативной фильтрации) и LightFM (для построения гибридной рекомендательной системы)

Подбор гиперпараметров осуществлялся на валидационной выборке с последующей проверкой на тестовой. К сожалению, не использовал оптимизационные фреймворки и написал поиск по сетке руками.

Гиперпараметры в Surprise и LightFM (можно было оптимизировать и остальные, их много):

- Размерность вектора скрытых признаков
- Скорость обучения

4. В моих экспериментах модель коллаборативной фильтрации с помощью Surprise оказалась намного лучше (около 10% по всем метрикам) гибридной модели.

Подробнее можно посмотреть в jupyter ноутбуке. Мне, к сожалению, не хватило времени на подбор гиперпараметров и дополнительных признаков фильмов для улучшения модели и поиск ошибки.

	Precision@10	Recall@10	F1-score	Average Precision
Random	0.49	0.65	0.53	0.6
CF	**0.59**	**0.81**	**0.65**	**0.77**
Hybrid	0.52	0.69	0.56	0.64

Как видно из таблицы, модель коллаборативной фильтрации показала лучший результат по всем 4 метрикам. Возможно, дело в метрике качества и способе оценки модели. Во многих статьях, в том числе, в оригинальной статье LightFM, сказано, что LightFM работает по крайней мере не хуже коллаборативной фильтрации. Из этого следует, что в гибридной модели есть ошибка и она работает некорректно, либо неверно подобраны гиперпараметры/метрика качества.

Дополнительная часть.

1. Предобработка данных.

В качестве предобработки данных я убрал все фильмы, у которых количество оценок меньше 500, а также пользователей с количеством просмотренных фильмов больше 5000 (слишком много информации, сложно выделить значимые признаки) и меньше 150 (недостаточно данных).

Для гибридной модели выделил из названия фильма название и год выпуска, каждому фильму добавил список жанров и релевантных тегов. Для рейтинга провел бинаризацию на положительное и негативное взаимодействие (рейтинг больше 4 – положительное взаимодействие, меньше 4 – негативное). Подход с бинаризацией взял из оригинальной статьи про LightFM.

2. Baseline решение

В качестве самого простого решения я выбрал рекомендацию случайных фильмов из тестового набора пользователя. Baseline решение помогает оценивать качество работы более сложных моделей.

Список источников:

1. Статья о Average Precision: https://sdsawtelle.github.io/blog/output/mean-average-precision-MAP-for-recommender-systems.html#:~:text=In%20recommendation%20systems%20MAP%20computes,relevant%22%20recommendations%20for%20that%20user
2. Статья о метриках в рекомендательных системах: https://ieeexplore.ieee.org/abstract/document/8550639
3. Статья от автора LightFM: https://arxiv.org/pdf/1507.08439.pdf
4. Статья от Авито про конкурс с рекомендательными системами: https://habr.com/ru/company/avito/blog/439206/
5. Тулбокс Surprise: https://surprise.readthedocs.io/en/stable/
6. Тулбокс LightFM: https://making.lyst.com/lightfm/docs/home.html
7. Kaggle notebooks: https://www.kaggle.com/code/sharthz23/implicit-lightfm https://www.kaggle.com/code/niyamatalmass/lightfm-hybrid-recommendation-system
8. Реализация LightFM «на пальцах»: https://towardsdatascience.com/how-i-would-explain-building-lightfm-hybrid-recommenders-to-a-5-year-old-b6ee18571309