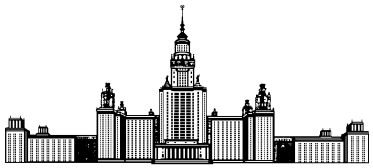
Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования



Отчет по преддипломной практике

«Изучение работы высоконагруженных рекомендательных систем»
«Studying the operation of highloaded recommender systems»

Выполнил:
студент 4 курса 417 группы
Борисов Алексей Антонович

Научный руководитель:
д. ф.-м. н., профессор
Дьяконов Александр Геннадьевич

Москва, 2022

Содержание

1 Введение

Преддипломная практика проходила в составе команды, занимающейся рекомендательными системами музыки в «ООО» ВК. Наша команда работает над рекомендациями музыки в нескольких сервисах: в социальной сети ВКонтакте, в приложении VK Музыка и в Марусе (голосовой помощник).

Основной целью данной практики было изучение работы рекомендательных систем на примере стриминговых сервисов музыки, а также разобраться с функционированием отдельных компонент рекомендательных систем, решая задачи по их разработке и улучшению.

Выполнение задания преддипломной практики должно дать необходимые знания для последующего написания ВКР бакалавра по теме «Решение проблемы использования одного аккаунта несколькими пользователями в рекомендательных системах».

2 Описание предметной области

В общем виде предметная область представляет собой стриминговый сервис, позволяющий пользователям прослушивать музыкальные композиции (треки), а также составлять собственные плейлисты (списки треков). У каждого трека имеется один или более исполнителей, также у исполнителей могут быть альбомы, которые также представляют собой списки треков.

Примеры стриминговых сервисов музыки: Яндекс Музыка, VK Музыка, Spotify, Apple Music.

3 Устройство рекомендательных систем

Во время прохождения практики, осуществлялась работа с двумя отдельными рекомендательными системами, что позволило лучше понять устройство таких систем, выделить общие компоненты.

Вообще говоря, в сервисах с которыми происходило взаимодействие существует большое количество разных рекомендаций: рекомендации по треку, рекомендации по исполнителю, рекомендации определенного жанра, общие рекомендации музыки для конкретного пользователя и т.д. Но в большинстве случаев можно выделить общую структуру получения рекомендаций:

1. Отбор кандидатов
2. Ранжирование
3. Постобработка

Кроме собственно получения рекомендаций можно выделить еще несколько элементов, без которых сложно представить себе работу рекомендательного сервиса:

- Автоматическое переобучение моделей машинного обучения

- Автоматическое обновление данных
- Инфраструктура для проверки гипотез (проведение A/B экспериментов)

3.1 Отбор кандидатов

Количество объектов, которые теоретически можно порекомендовать может достигать сотен миллионов. Поэтому в онлайн сервисе использовать тяжелые модели на всем множестве объектов не представляется возможным. Необходимо сперва более легковесными методами выделить множество объектов, которые могут быть релевантными.

Кандидаты могут поступать из разных источников, например, при наличии векторных представлений можно выявлять кандидатов на основе близости этих представлений между собой. Также можно брать популярные объекты, так как они с большой вероятностью могут быть релевантны пользователю.

В системах, с которыми производилась работа имелись векторные представления треков, которые получались из разложения матрицы сопрослушиваний треков (сколько раз треки слушали вместе) или матрицы пользователь-трек . И в большинстве случаев отбор кандидатов происходит на основе векторных представлений.

3.2 Ранжирование

После того как этап отбора кандидатов завершен, необходимо упорядочить их по релевантности. На данном этапе могут использоваться более сложные методы, например, градиентный бустинг над деревьями решений (gradient boosted decision trees) или нейросетевые методы.

Этап отбора кандидатов может быть многоуровневым, например, после получения N кандидатов происходит их ранжирование некоторым способом, после чего оставляют только M лучших кандидатов ($M < N$). По мере уменьшения количества кандидатов появляется возможность использовать все более сложные методы, что позволяет получить более качественные рекомендации.

3.3 Постобработка

Часто появляются ограничения на итоговый список рекомендованных объектов, связанные с некоторой бизнес-логикой. Приведу несколько примеров из систем, над которыми велась работа:

- В детском режиме недопустимо, чтобы среди рекомендованных треков были те, что нельзя слушать детям.
- После того как пользователь только прослушал трек странно включать в качестве рекомендаций тот же самый трек.
- Могут быть треки, запрещенные на территории некоторых стран. Недопустимо рекомендовать трек, если известно, что пользователь находится на территории страны, в которой этот трек запрещен.

4 Рекомендации для детей

В рекомендациях музыки в Марусе большое внимание уделено рекомендациям для детей, и часть практики решались задачи, связанные с этим.

4.1 Классификатор детских треков

Для решения некоторых задач, о которых речь пойдет дальше, была необходима модель способная определять является ли трек детским или нет. В качестве данных для обучения имелись названия треков, их исполнители и векторные представления треков и исполнителей. Также имелась асессорская разметка некоторого количества детских треков. Размеченные треки являются «очень детскими», поэтому фактически задача классификации сводится к отделению очень детских треков от всех остальных, и должна с хорошим качеством решаться простыми методами. Сразу стоит отметить, что хотелось использовать данную модель в онлайн режиме, что накладывало дополнительные ограничения на ее время работы, идеальным вариантом казались линейные модели.

В качестве метода классификации была использована логистическая регрессия, а в качестве признаков были взяты скалярные произведения векторов трека и исполнителя на средний вектор трека и исполнителя по всем детским трекам соответственно. Всего получилось 4 признака: левый и правый вектор трека из матричного разложения треки на треки, и вектор исполнителя и трека из разложения исполнители на треки. Ниже представлена таблица 1 со значениями метрик при разных порогах бинаризации вероятностей, а также рисунок 1 с изображением ROC-кривой. (оценивание производилось на сбалансированных данных).

threshold	accuracy	precision	recall	f1-score
0.10	0.904	0.872	0.947	0.908
0.20	0.932	0.927	0.938	0.933
0.30	0.939	0.949	0.928	0.938
0.40	0.945	0.965	0.923	0.943
0.50	0.947	0.975	0.917	0.945
0.60	0.946	0.981	0.909	0.943
0.70	0.945	0.988	0.902	0.943
0.80	0.940	0.993	0.886	0.936
0.85	0.929	0.996	0.861	0.924
0.90	0.915	0.997	0.833	0.908
0.95	0.887	0.997	0.776	0.873

Таблица 1: Значения метрик при разных порогах бинаризации.

Стоит отметить, что использовать такой классификатор или любой другой для определение какие треки можно включать в детском режиме не представлялось возможным, для этого классификатор должен иметь точность 1.0, так как абсолютно

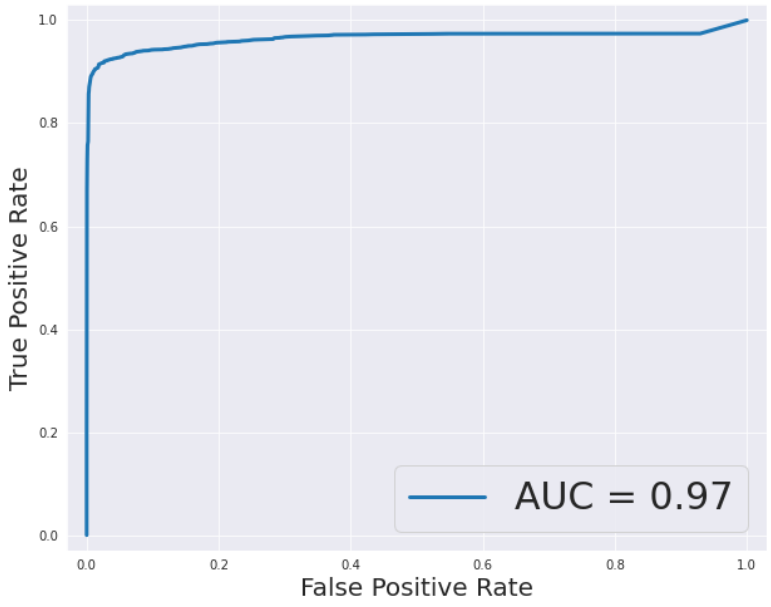


Рис. 1: ROC-кривая классификатора детских треков.

недопустимо разрешить проигрывание в детском режиме трека, который на самом деле нельзя детям. Однако данный классификатор вполне подходит для того чтобы определить, например, насколько пользователю нравится детское (насколько хорошо он слушает детские треки). Подробнее такое использование модели будет описано ниже.

4.2 Исключение детских треков из рекомендаций

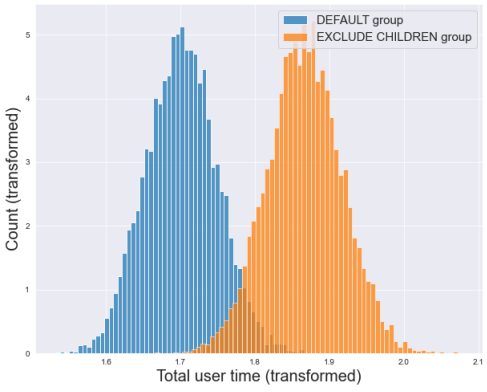


Рис. 2: Результаты A/B эксперимента по исключению детских треков.

Было замечено, что если дети и взрослые слушают музыку с одного устройства (колонки), то впоследствии взрослые могут в рекомендациях получать детские треки, из-за чего их опыт пользования сервисом может быть испорчен. В качестве возможного решения данной проблемы была использована следующая эвристика: если пользователь «плохо» слушает детские треки в рекомендациях (часто пропускает их), то вероятно не стоит рекомендовать ему детские треки.

Для определения того нужно ли оставлять детское использовался следующий алгоритм: если доля пропусков детских треков из рекомендаций выше некоторого порога, то детские треки исключаются из рекомендаций. Данный по-

рог подбирался при помощи A/B эксперимента, последующий анализ которого заключался в том, что если в эксперименте использовать порог t, то можно получить результаты для любого порога больше чем t просто сузив выборку на пользователей с долей пропусков больше t.

На рисунке 2 изображены результаты A/B эксперимента при использовании значения порога t близкого к оптимальному. Удалось добиться статистически значимого улучшения рекомендаций, например, среднее время прослушивания для одного пользователя увеличилось на 9%. Здесь и далее статистическая значимость имеется в виду с уровнем значимости 0.1.

4.3 Улучшение рекомендаций

Отдельными задачами, связанными с рекомендациями для детей, являлось добавление персонализации в детские рекомендации по треку, а также создание общих рекомендаций в детском режиме. Как уже было упомянуто, в детском режиме нельзя рекомендовать треки, которые нельзя детям. Результатом выполнения этих задач должно было стать улучшение рекомендаций для детей.

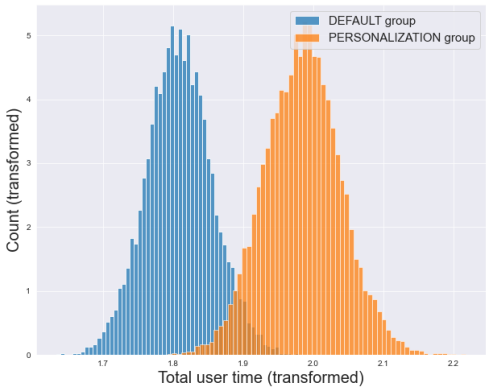


Рис. 3: Результаты A/B эксперимента по добавлению персонализации в детские рекомендации.

Задача добавления персонализации в детские рекомендации по треку представляет собой применение градиентного бустинга над деревьями решений для переранжирования кандидатов с учетом персонализации. Отличие от обычных рекомендаций по треку заключается в том, что в данном случае используется редуцированное множество треков, из него исключены треки которые нельзя детям. Вообще говоря, имело смысл обучить новую модель для ранжирования, но был интересен эффект от применения уже имеющейся модели ранжирования, которая обучалась на другом множестве треков. Таким образом, получаем нечто напоминающее transfer learning. На рисунке 3 изображены результаты A/B эксперимента при добавлении персонализации. Как и ожидалось это дало статистически значимое улучшение качества рекомендаций, например, среднее время пользователя увеличилось на 9%.

Под общими рекомендациями понимается то, что пользователю нужно порекомендовать список треков, без дополнительных условий. В детском режиме появляется ограничение на итоговый список треков, не допускаются треки, которые нельзя слушать детям. Основная вариативность заключается в формировании так называемых «якорных треков» (anchors), по которым впоследствии будут отбираться кандидаты. Рассматривались два способа:

- В качестве якорей брать детские треки из прослушиваний пользователя. В данном случае трек считается детским, если он был размечен асессорами как

детский. Хотя в дальнейшем можно попробовать определять детский трек или нет с помощью модели. Этот подход на графике обозначен как ALPHA.

- В качестве якорей брать треки, которые проигрывались после запроса детским голосом. В Марусе существует функция «Узнавать детей», при включении которой разрешается использовать распознавание голоса на детский/взрослый. Этот подход на графике обозначен как BETA.

В качестве бейзлайнового решения использовались рекомендации из заранее зафиксированного списка детских треков, который случайным образом перемешивался. На графике данный подход обозначен как DEFAULT.

Как и следовало ожидать, оба способа генерации кандидатов статистически значимо превзошли бейзлайновое решение, но использование в качестве якорей детских треков показало лучший результат, хотя разница между ALPHA и BETA оказалась не статистически значимой. При оценивании среднего времени прослушивания на пользователя получили следующее относительное улучшение: BETA лучше DEFAULT на 12%, ALPHA лучше DEFAULT на 18%.

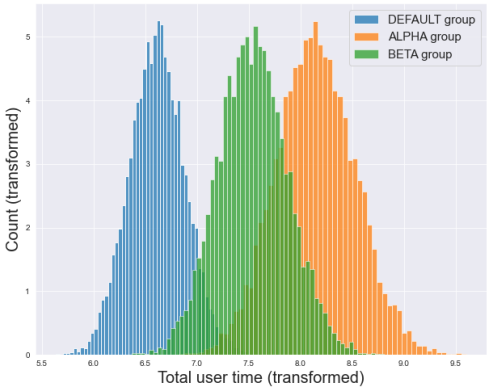


Рис. 4: Результаты A/B/C эксперимента по созданию общих рекомендаций для детей.

5 Создание новых типов рекомендаций

Как уже было упомянуто, в рекомендательных системах, с которыми производилась работа, имеются разные виды рекомендаций. Одной из задач, которые были решены во время прохождения практики, является создание нескольких новых типов рекомендаций, а именно: рекомендации по исполнителю, рекомендации по альбому/плейлисту.

5.1 Рекомендации по исполнителю

Задача формулируется следующим образом: «Для заданного исполнителя и заданного пользователя сформировать список из N треков, которые были бы релевантны как треки в том же жанре/стиле, что у данного исполнителя». В этом списке допускаются треки самого исполнителя. В приложении VK Музыка это называется «микс по артисту».

$$\text{artist\_id} \longrightarrow \text{track\_id}_1, \ldots, \text{track\_id}_N$$

Исполнитель определяется своими треками, фактически задача сводится к получению списка треков (рекомендации) по списку треков (треки исполнителя). Ниже данная задача будет рассмотрена в более общем виде.

5.2 Рекомендации по альбому/плейлисту

В данном случае можно логически разделить рекомендации по плейлисту и по альбому, потому что под плейлистом понимается заданный список треков, который пользователь может сформировать произвольным образом, а альбом это зачастую треки одного исполнителя в одном жанре. Наличие такой специфики может быть использовано при построении рекомендаций. Далее для определенности будем использовать понятие «плейлист».

Задача формулируется следующим образом: «Для заданного плейлиста и заданного пользователя сформировать список из N треков, которые были бы релевантны для включения после окончания плейлиста, причем в этом списке не должно быть треков из этого плейлиста». Последнее замечание связано с тем, что странно включать те же треки, которые человек только что прослушал.

$$\text{playlist\_id} \longrightarrow \text{track\_id}_1, \ldots, \text{track\_id}_N$$

Фактически снова имеем задачу получения списка треков (рекомендации) по списку треков (треки из плейлиста).

5.3 Рекомендации по списку треков

Основная вариативность и сложность заключается в отборе кандидатов, так как после получения списка кандидатов можно использовать общую модель ранжирования с персонализацией на основе градиентного бустинга над деревьями решений (gradient boosted decision trees).

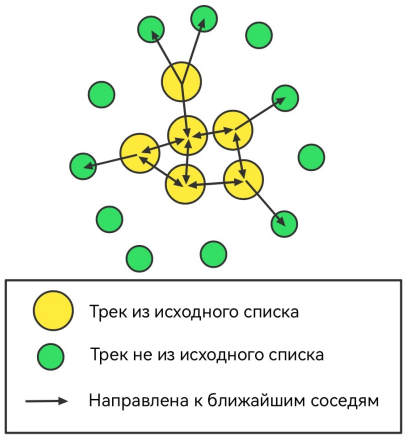


Рис. 5: Иллюстрация недостатка первого варианта отбора кандидатов.

Первый вариант для отбора кандидатов заключается в том, чтобы брать по каждому треку из списка M ближайших к нему треков в смысле близости векторных представлений. Но у такого подхода есть проблема, если в альбоме K треков, и для каждого трека брать M ближайших к нему, то может быть получено на самом деле намного меньше чем $K \cdot M$ кандидатов. Это связано с тем, что если треки из списка некоторым образом связаны, например, это треки одного исполнителя или это треки из одного альбома, то векторные представления треков будут располагаться очень близко друг к другу. В таком случае ближайшие соседи для треков из списка будут сильно пересекаться. Тогда для гарантированного получения итогового списка из N треков нужно брать M непозволительно большим. На рисунке 5 проиллюстрирован этот недостаток: если брать 3 ближайших соседей для представленных 6 треков, то уникальных кандидатов будет всего 10, а не 18. А при необходимости фильтрации треков из исходного списка это число уменьшится до 5.

Второй вариант заключается в построении общего векторного представления для списка треков. Для списка из треков одного исполнителя этот вектор можно интерпретировать как вектор данного исполнителя, а для треков из одного альбома/плейлиста как вектор альбома/плейлиста. Самым простым вариантом получения такого вектора, является усреднение векторов треков. Но у такого подхода также есть недостаток: если треки в списке находятся на большом расстоянии друг от друга (что вполне возможно так как плейлисты формируются пользователями), то вектор среднего может попасть в случайную область пространства и не давать качественного описания для исходного списка треков. Данный недостаток проиллюстрирован на рисунке 6.

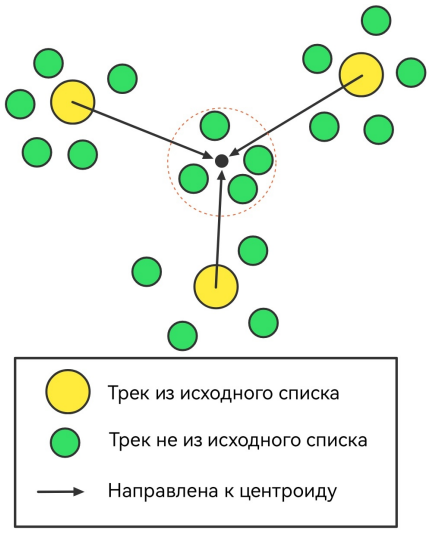


Рис. 6: Иллюстрация недостатка второго варианта отбора кандидатов.

Комбинированный вариант. Первые два варианта можно естественным образом скомбинировать. Вначале получить $\frac{N}{K}$ ближайших треков для каждого трека из списка (всего в списке K треков) и посмотреть сколько всего уникальных треков нашлось. Если нашлось недостаточно, то можно построить средний вектор и взять N ближайших треков по нему, добавив полученные треки в уже частично сформированный список. Таким образом, если треки в списке слабо связаны между собой, то после первого шага уже получится достаточное количество треков, а если они сильно связаны (один исполнитель/альбом), то использование вектора среднего позволит найти релевантных кандидатов.

Вариант с кластеризацией. Вообще говоря, можно произвести кластеризацию треков в пространстве их векторных представлений, найти центр каждого кластера и отбирать кандидатов как ближайшие треки к центрам кластеров, причем можно для каждого центроида брать число кандидатов пропорционально размеру кластера. Теоретически этот подход кажется наиболее правильным, но и более сложным, чем варианты выше. Возникают вопросы: «Как выбирать число кластеров?», «Какой алгоритм кластеризации использовать?». На рисунке 7 проиллюстрирован отбор кандидатов с использованием данного подхода.

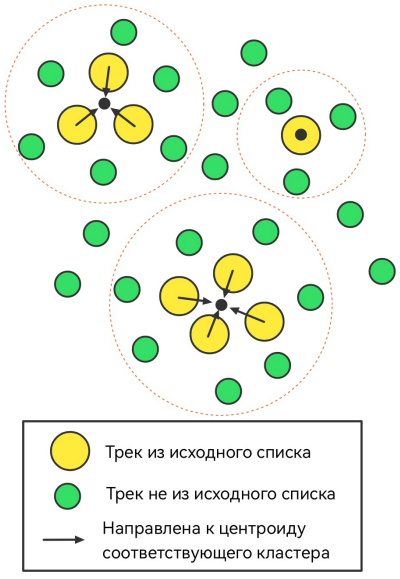


Рис. 7: Иллюстрация отбора кандидатов с использованием кластеризации.

Был реализован комбинированный вариант для рекомендаций по исполнителю, альбому и плейлисту. Однако в будущем планируется подробнее исследовать и реализовать вариант с использованием кластеризации.

6 Другие задачи

Выше были подробно описаны некоторые из задач, которые решались при прохождении преддипломной практики, однако кроме них были и другие задачи, которые сложно объединить в какие-то группы. Ниже перечислены некоторые из них:

- Изменение динамики весов при «прогреве» «холодных» пользователей, проведение соответствующего A/B эксперимента.
- Анализ эксперимента по формированию персонализированных жанровых рекомендаций.
- Автоматизация обновления моделей и данных при помощи инструмента Apache Airflow.

7 Заключение

В процессе выполнения задания преддипломной практики был решен ряд задач по улучшению рекомендательных систем, с которыми происходило взаимодействие. Работа над решением данных задач позволила разобраться в устройстве рекомендательных систем, а также в рассматриваемой предметной области. Это дало необходимые знания для работы над ВКР бакалавра по теме «Решение проблемы использования одного аккаунта несколькими пользователями в рекомендательных системах».

Некоторые задачи включали в себя проведение анализа данных и проверку гипотез (проведение A/B экспериментов), благодаря чему были улучшены соответствующие навыки. Также был получен опыт работы с ClickHouse, инструментом для выполнения аналитических запросов.

Кроме того, были обретены навыки использования Apache Airflow для управления потоками обработки данных и обучения моделей машинного обучения. Также был получен опыт программирования на языке PHP.

Список литературы

[1] Yuanzhe Peng. A survey on modern recommendation system based on big data, 2022.

[2] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback, 2012.

[3] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Defu Lian. A survey on session-based recommender systems, 2019.