Санкт-Петербургский государственный университет

Кафедра Системного Программирования

Сулягина Анастасия Александровна

Персональное ранжирование новостной ленты в социальных сетях

Бакалаврская работа

Научный руководитель:
к. ф.-м. н. Бугайченко Д. Ю.

Рецензент:
Ведущий разработчик ООО “НМТ — Новые Мобильные Технологии” Невоструев К. Н.

Санкт-Петербург
2017

SAINT-PETERSBURG STATE UNIVERSITY

Software Engineering

Anastasia Sulyagina

Personalized news feed ranking in social networks

Graduation Thesis

Scientific supervisor:
professor Bugaychenko Dmitry

Reviewer:
Senior Developer, NMT LLC Nevostruev Konstantin

Saint-Petersburg
2017

Оглавление

Введение

Интернет стремительно растет и развивается, и с каждым днем в нем появляется все больше информации. Но не вся информация интересна всем пользователям, в связи с этим одной из важнейших задач IR в настоящее время является предоставление пользователю персонализованного контента. Персонализация - обязательный элемент любой современной интернет-площадки, поскольку предоставляет пользователю возможность потреблять наиболее интересный контент, быстро находить нужную информацию по запросу, приобретать наиболее подходящие товары.

Социальные сети в настоящее время являются основной платформой для общения и получения информации. В новостной ленте среднего пользователя социальной сети в день появляется 2000 записей, все из которых он физически не может прочитать. Задача любой социальной сети состоит в том, чтобы пользователь получал более интересный для него контент среди огромного количества записей и информационного шума. Для решения этой задачи предлагается ранжировать записи новостной ленты пользователя в порядке релевантности. С внедрением подобной технологии в социальную сеть увеличивается время, проведенное пользователем в социальной сети, его удовлетворенность, вероятность что он вернется на ресурс снова. Таким же образом можно ранжировать контекстную рекламу, показываемую пользователю, с целью сделать ее более релевантной и менее раздражающей.

Многие задачи в области Information Retrieval по своей природе являются задачами ранжирования, так как именно оно лежит в основе рекомендаций и персонализации. Одна из наиболее интересных в настоящее время задач в области ранжирования заключается в том, чтобы, используя некоторые тренировочные данные, построить ранжирующую модель машинного обучения так, чтобы модель могла отсортировать новые данные по их релевантности. Подобная задача и будет рассматриваться в работе.

1. Постановка задачи

Целью данной работы является реализация системы персонального ранжирования записей в новостной ленте пользователя социальной сети. Для достижения этой цели были сформулированы следующие задачи.

1. Провести анализ существующих решений в области ранжирования документов.

2. Разработать архитектуру ранжирующей системы.

3. Провести обработку пользовательских данных с выделением признаков для тренировки ранжирующей модели.

4. Реализовать комбинированную ранжирующую модель на основе наиболее успешных подходов в области ранжирования.

5. Провести апробацию реализованной системы и оценить ее качество.

2. Обзор

2.1. Алрогитмы ранжирования

На данный момент существует множество алгоритмов ранжирования. Среди них выделяют 3 типа: точечные, попарные и списочные.

2.1.1. Точечные алгоритмы

Точечными алгоритмами ранжирования называют алгоритмы, оценивающие релевантность отдельных документов относительно запроса и затем сортирующие их в порядке релевантности. Алгоритмы этого вида наиболее просты, потому что для решения подобной задачи применимы многие уже существующие алгоритмы машинного обучения.

Недостатки подобных алгоритмов заключаются в следующем: так как они принимают на вход отдельные документы, относительный порядок документов в результате никак не учитывается, а так как задача ранжирования - задача определения наилучшего относительного порядка выдачи докуметов по запросу, можно сказать, что точечные алгоритмы не решают ее напрямую, и следовательно, могут быть использованы только как субоптимальное решение. Примерами подобных алгоритмов могут служить любые классические алгоритмы машинного обучения, решающие задачи регрессии или классификации.

2.1.2. Попарные алгоритмы

Попарные алгоритмы ранжирования фокусируются на предсказании относительного порядка пары документов. Ранжирование сводится к задаче классификации на парах документов. Цель - минимизировать количество неправильно классифицированных пар. В отличие от точечных методов, попарные учитывают относительный порядок документов, поэтому можно считать, что они моделируют задачу ранжирования напрямую. Тем не менее, так как за эталон берется именно релевантность пары документов, а в выборках для задачи ранжирования данные о релевантности конкретной пары документов относительно друг друга в чистом виде встречаются редко, подобные зависимости нужно искусственно извлекать из данных, в процессе чего при наличии более 2х категорий теряется информация о различии между классами. Так как при попарном сравнении документов количество данных увеличивается до квадрата изначального, на больших объемах данных затруднительно использовать информацию о всех парах, присутствующих в данных. В таких случаях используется сэмплирование данных, то есть используется лишь малая часть данных. Хотелось бы не извлекать много лишней информации и по максимуму использовать данные.

Подобные алгоритмы также могут строиться на различных классических алгоритмах машинного обучения: RankSVM на основе Support Vector Machine, RankBoost на основе градиентного бустинга, RankNet на основе нейронной сети.

2.1.3. Списочные алгоритмы

Списочные алгоритмы ранжирования решают задачу ранжирования напрямую: определяют оптимальный порядок документов в данном списке. На текущий момент существуют два основных способа решения этой задачи.

Оптимизация метрик напрямую

Оптимизация метрик, используемых для оценки точности ранжирования выглядит как самый очевидный способ решения проблемы. Но задача на самом деле не настолько проста: Существующие метрики оценки качества ранжирования (NDCG, MAP) основаны на позициях документов в списке ранжирования, из чего следует их дискретность и недифференцируемость, а большинство существующих техник оптимизации функций рассчитаны на работу с непрерывными и дифференцируемыми случаями. Было представлено множество способов решения проблемы: Оптимизация дифференцируемого и непрерывного приближения существующих метрик (SoftRank, AppRank, SVMmap, SVMndcg, PermuRank); Использование технологий, позволяющих оптимизировать сложные целевые функции (AdaRank, LambdaMART - gradient boosting, RankGP - генетические алгоритмы)

Минимизация потерь при ранжировании

Результатом работы алгоритмов второй категории является перестановка документов. Функция потерь подобных алгоритмов измеряет неконсистентность между выводом модели и ground truth перестановкой. Хоть метрики и не оптимизируются напрямую, подобный подход дает хорошие результаты. Примерами таких алгоритмов могут служить ListNet и ListMLE - методы, основанные на нейронных сетях.

Результаты исследований показывают, что точность списочных алгоритмов превышает точность точечных и попарных. Тем не менее для тренировки подобных алгоритмов требуется большее время.

2.2. Сравнение наиболее интересных подходов

Для сравнения решено было выбрать наиболее показательные алгоритмы разных типов и классов. Линейные алгоритмы представлены точечной логистической регрессией и попарным RankSVM, сетевые - списочными ListNet и ListMLE, деревья - списочным LambdaMART.

2.2.1. Логистическая регрессия

Логистическая регрессия - классическая вероятностная модель машинного обучения. Решающая функция выглядит следующим образом:

$$f(t) = \frac{1}{1 + e^{-t}}$$

где t - линейная комбинация независимых переменных $x_1..x_n$ и коэффициентов регрессии $\theta_1..\theta_n$. Коэффициенты подбираются методом максимального правдоподобия так, чтобы они максимизировали функцию правдоподобия на обучающей выборке. Максимизировать ее можно, например, градиентным спуском.

Для ранжирования используется следующим образом: определяет вероятность релевантности каждого документа, после чего документы сортируются по релевантности

2.2.2. RankSVM

RankSVM [6] - Попарный алгоритм, использующий SVM для задачи классификации пары документов. Целевая функция сходна с классическим SVM, отличие от классического SVM в функции потерь. Функция потерь RankSVM - hinge loss [8] относительно пар документов

$$f(y) = max(0, 1 - t * y)$$

где t - предсказанный класс (в данном случае +1, если первый документ пары релевантнее второго, или -1 в обратном случае), а y - оценка модели. Подобная функция потерь обеспечивает максимальное расстояние между результирующими классами.

2.2.3. ListNet

ListNet [10] - первый из представленных списочных подходов. Работает на основе Luce model

$$P(i) = \frac{w(i)}{\sum_j W(j)}$$

- модели, представляющей распределение вероятностей перестановок. Данная модель на основе оценок, данных оценочной функцией, определяет вероятности всех возможных перестановок документов, основанное на цепном правиле [7]. ListNet определяет распределение вероятностей на основе оценивающей функции, затем определяет распределение, основанное на эталонной функции. На следующем шаге использует расхождение Кульбака–Лейблера [9]

$$D_{\mathrm{KL}}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$$

между двумя этими распределениями для определения потерь списочного ранжирования. Эти потери минимизируются градиентным спуском в нейронной сети.

2.2.4. ListMLE

ListMLE [2] - еще один списочный метод ранжирования, основанный на распределениях вероятностей. Тоже основан на Luce model, но использует логарифмическую функцию правдоподобия для определения потерь ранжирования. По заявлению автора алгоритма, логарифмическая функция правдоподобия обладает лучшими свойствами и кроме улучшения результатов модели ListNet позволяет сократить время тренировки модели. Потери минимизируются градиентным спуском в нейронной сети

2.2.5. LambdaMART

LambdaMART [1] - подход, основанный на методе градиентного бустинга. В обычном ансамбле бустинговых деревьев каждое следующее дерево пересчитывает градиенты, определяя направление, минимизирующее функцию потерь. В LambdaMART же вместо стандартных градиентов лямбды - градиенты, вычисленные способом, представленным в LambdaRANK [1] относительно пар документов. Алгоритм учитывает изменение оптимизируемой метрики при замене позиций пары документов при ранжировании и на основе этого изменения измеряет потери.

2.3. Метрики

Для того, чтобы уметь сравнивать алгоритмы, нужно научиться определять их качество. Для этого необходимо ввести понятие "эталонная перестановка (π^{true})" - перестановка документов, соответствующая порядку убывания их реальной релевантности. Метрики качества ранжирования должны уметь определить разницу между перестановкой документов, полученной при помощи алгоритма, и эталонной перестановкой. В настоящее время для оценки точности ранжирующих алгоритмов используются две метрики: MAP [3] и NDCG [4].

Пусть $\text{r}^{\text{true}}(\text{e}) = 1$, если элемент e релевантен, иначе $\text{r}^{\text{true}}(\text{e}) = 0$

2.3.1. MAP

$$Ap@K = \frac{1}{K} \sum_{k=1}^{K} (r^{\text{true}}(\pi_{\text{k}}) * \sum_{n=1}^{k} r^{\text{true}}(\pi_{\text{n}}))$$

Находит среднюю долю релевантных элементов для k = 1..K, таким образом учитывая порядок объектов.

MAP@K используется при наличии множества объектов (в данной работе пользователей), для которых считается Ap@K и представляет собой среднее для всех посчитанных Ap@K

2.3.2. NDCG

Normalized discounted cumulative gain учитывает позиции релевантных элементов с весом, обратно пропорциональным логарифму позиции элемента.

$$NDCG@K = \frac{DCG@k}{IDCG@K}$$

где

$$DCG@K = \sum_{k=1}^{K} \frac{2^{\mathrm{r}^{\mathrm{true}}(\mathrm{pi_k})} - 1}{log(k+1)}$$

а IDCG@k - DCG@K для эталонной перестановки

2.4. Определение релевантности контента

Типичные задачи ранжирования представляют собой ранжирование документов в соответствии с запросом. При такой постановке задачи большое внимание уделяется содержимому документа, в то же время отсутствует акцент на предпочтениях пользователя. В задаче ранжирования записей в новостной ленте пользователя социальной сети наиболее интересны три аспекта: отсутствие явного запроса, отсутствие явной оценки и влияние окружения пользователя на его предпочтения. Поэтому одна из ключевых задач работы - определить, что же именно хочет увидеть пользователь и какие признаки влияют на его отношение к контенту. Для этого необходимо провести исследование структуры социальных сетей и выделить наиболее важные аспекты взаимодействия пользователя с социальной сетью.

2.4.1. Социальные признаки

Существует прямая связь между интересами пользователя и его социальным окружением. Так, пользователь охотнее потребляет контент, созданный людьми со схожими с ним интересами. Также пользователи, объединяющиеся в некоторое сообщество, подвержены схожим идеям. Исходя из этого можно сказать, что реакция пользователя на контент во многом будет зависеть от социальных связей и взаимодействий с автором контента.

2.4.2. Интересы и паттерны поведения пользователя

Структура социальных сетей достаточно сложна и пользователю предоставлено множество методов взаимодействия с контентом. Каждый пользователь взаимодействует с контентом по-разному и при персонализации очень важно учесть эти предпочтения. Следует принять во внимание предпочтения пользователя по типу контента и его авторству.

3. Архитектура

В данной работе была разработана и реализована модульная архитектура, обеспечивающая эффективное взаимодействие компонентов системы. Предложенная архитекрура представлена на рисунке.

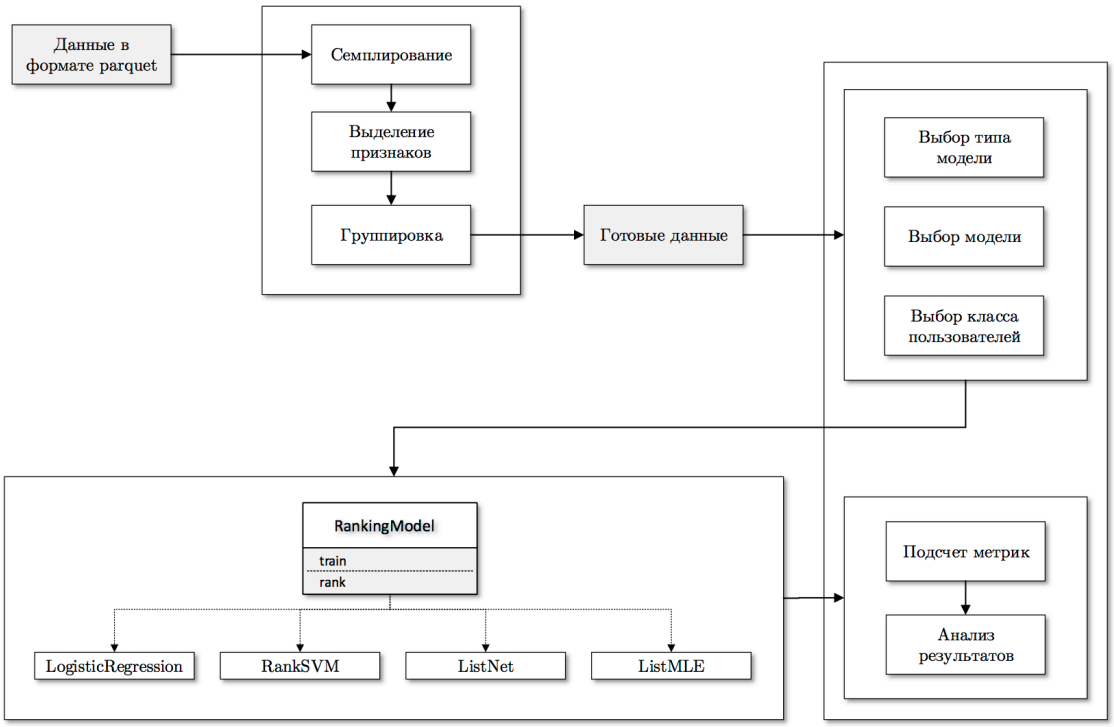

Рис. 1: Архитектура системы

В рамках системы реализованы следующие модули.

3.1. Модуль, обрабатывающий данные

Занимается приведением исходных пользовательских данных к пригодному для ранжирования виду. Для достижения наибольшей гибкости для экспериментов модуль разделен на две компоненты. Первая из них проводит семплирование данных, выделяет из данных признаки и проводит оценивание записей, вторая группирует данные по пользователю, выделяет нужные временные промежутки и разделяет данные на тренировочную и тестовую выборки.

3.2. Ранжирующий модуль

В рамках модуля реализованы ранжирующие модели логистическая регрессия, SankSVM, ListNet, ListMLE с возможностью обучения, тестирования, установки параметров обучения.

3.3. Модуль, оценивающий точность модели

В рамках модуля реализованы метрики NCDG и MAP, с помощью которых по выводу модели рассчитывается ее точность. Модуль позволяет проводить тестирование модели на данных, отфильтрованных по различным признакам и поддерживает построение общей и персональной модели для алгоритмов, реализованных в рамках работы. Также в модуле присутствует функциональность, позволяющая сравнивать различные модели по разным признакам и визуализировать сравнение.

4. Обработка данных

4.1. Исходные данные

Данные, использованные при ранжировании, включали информацию о всех записях в новостной ленте n пользователей за промежуток времени в 2 месяца (Ноябрь - Декабрь 2016 года). Для каждой записи известна следующая информация: демографические данные, метаданные, счетчики взаимодействий между пользователем и автором контента, отношения между пользователем и автором контента, реакция пользователя на запись.

4.2. Признаки релевантности контента

В процессе изучения структуры социальных сетей были выделены признаки, влияющие на отношение пользоваля к контенту. Ниже перечислены разбитые на категории признаки, определенные как релевантные.

4.2.1. Взаимодействия пользователя с автором записи

1. Количество посланных сообщений автору записи
2. Количество просмотров страницы автора записи
3. Количество лайков автора записи
4. Количество комментариев пользователя на странице автора поста
5. Количество просмотров различного контента пользователем со страницы автора поста.
6. Вес автора контета, посчиатнный на основе статистики прошлых взаимодействий

Признаки этой категории были нормализованы по времени экспоненциальным скользящим средним.

Обратные метрики взяты не были, так как после изучения существующих работ по выделению социальных связей из соцсетей, были сделаны выводы, что пользовательский интерес не является взаимным и в данной работе представляет интерес только отношение пользователя к автору поста

4.2.2. Отношения между пользователем и автором

В пользовательских данных присутствовала информация об отношениях пользователя и автора контента. Она представлялась в виде маски, которая включала в себя 20 видов отношений. Мной было принято решение сгруппировать виды отношений следующим образом:

1. Член семьи: LOVE, SPOUSE, PARENT, CHILD, BROTHER_SISTER, UNCLE_AUNT, RELATIVE, NEPHEW, GODPARENT, GODCHILD, GRANDPARENT, GRANDCHILD, PARENT_IN_LAW, CHILD_IN_LAW

2. Коллега: COLLEAGUE, COLLEGE_UNIVERSITY_FELLOW

3. Друг: CLOSE_FRIEND, PLAYING_TOGETHER

4. Одноклассник: SCHOOLMATE

4. Товарищ из армии: ARMY_FELLOW

Каждая из вышеописанных групп составила отдельный бинарный признак.

5. Добавлен ли автор поста в избранное пользователем

4.2.3. Демографические признаки

1. Совпадение пола
2. Совпадение возраста с погрешностью 5 лет
3. Совпадение места проживания автора контента и пользователя.

4.2.4. Реакция других пользователей на запись

1. Количество лайков от друзей пользователя, нормализованное по времени экспоненциальным скользящим средним.

2. Количество комментариев от друзей пользователя, нормализованное по времени экспоненциальным скользящим средним.

3. Аггрегированный с гиперболическим убаванием вес пользователей, лайкнувших запись на основе статистики прошлых взаимодействий.

4. Общий Click-Through Rate

5. Click-Through Rate по пользователям схожего с целевым пользователем пола

6. Click-Through Rate по негативной реакции

4.2.5. Тип записи

Данные содержали информацию об авторстве записи. Авторство может быть следующим: USER, GROUP_OPEN_OFFICIAL, GROUP_OPEN, USER_APP, GROUP_CLOSED. Каждый тип был выделен в отдельный бинарный признак. Также при определении типа записи учитывалось, является ли запись репостом и опубликована ли запись в рамках альбома.

1. Запись опубликована пользователем
2. Запись опубликована открытой группой
3. Запись опубликована официальным пабликом
4. Запись опубликована приложением
5. Запись опубликована приватной группой
6. Является ли запись репостом

7. Является ли запись частью альбома

4.2.6. Содержимое записи

Важной категорией признаков также является контент записи.

1. Наличие видео
2. Наличие опросов
3. Наличие картинок
4. Наличие анимационного контента
5. Наличие текста
6. Наличие ссылок
7. Наличие геолокации
8. Совпадение карты интересов пользователя с семантикой поста

Было принято решение не учитывать количество объектов каждого вида в записи, так как для некоторых видов контента данные о количестве были неточны и наличие было определено как более показательный признак.

4.3. Оценка реакции пользователя на контент

Для того, чтобы ранжирование записей было возможным, следовало численно оценить записи так, чтобы самый релевантный контент имел больший вес, а наименее релевантный, соответственно, меньший. В выгруженных данных присутствовали следующие виды реакции пользователя на контент:

1. комментарий
2. пересылка
3. лайк
4. нажатие
5. просмотр
6. дислайк
7. игнорирование.

Каждой записи соответствовало несколько видов реакции: пользователь листал ленту несколько раз или одноразово среагировал несколькими способами.

Такие виды реакции как нажатие, просмотр, комментарий, были в итоге признаны неоднозначными, так как для каждого пользователя для различных записей подобная реакция может означать разную степень релевантности и численное оценивание подобных признаков может затруднить принятие решения моделью. Поэтому после множества экспериментов для чистоты эксперимента было принято решение использовать наиболее простую конфигурацию. Реакция пользователя на обучающих данных была оценена следующим образом:

1 (Релевантный), если в списке реакций присутствовал лайк

-1 (Нерелевантный), если в списке реакций присутствовал дислайк

0 (Нейтральный), если реакции, упомянутые выше, отсутствовали.

4.4. Группировка данных

Так как персональная модель создает уникальную рекомендацию для каждого пользователя, данные были сгруппированы по пользователям.

Кроме того, было принято решение разбить данные на двухнедельные промежутки. Для каждого промежутка данные за первые 11 дней были использованы для обучения, а последующие 3 дня - для тестирования. Группировка была произведена именно таким образом, так как за 2 недели средний пользователь просматривает достаточно записей для тренировки персональной модели, в то же время за этот временной промежуток не успевают кардинально поменяться интересы пользователя и его реакция на контент.

Если же в обучающей или тестовой выборке пользователя было меньше 50 записей, пользователь не участвовал в построении модели, так как если за 2 недели пользователь просмотрел меньше 50 записей, это мало что о нем говорит и не имеет смысла обучать модель, если же в тестовой выборке меньше 50 записей, считаем, что пользователь в любом случае просмотрит все новые записи и сортировка не нужна.

5. Реализация модели

5.1. Выбор оптимальной комбинации моделей

В рамках работы были реализованы следующие модели:

1. Точечный алгоритм ранжирования на основе логистической регрессии
2. Попарный алгоритм ранжирования на основе SVM
3. Списочный алгоритм ранжирования ListNet на основе нейронной сети
4. Списочный алгоритм ранжирования ListMLE на основе нейронной сети

Также в сравнении участвовал списочный алгоритм ранжирования LambdaMART на основе градиентного бустинга, реализованный в рамках опенсорсной библиотеки RankLib [5]

Для упрощения интерпретации результатов пользователи были отсортированы в порядке возрастания количества записей в обучающей выборке, после чего была построена логарифмически нормализованная гистограмма, при помощи которой пользователи были разделены на группы.

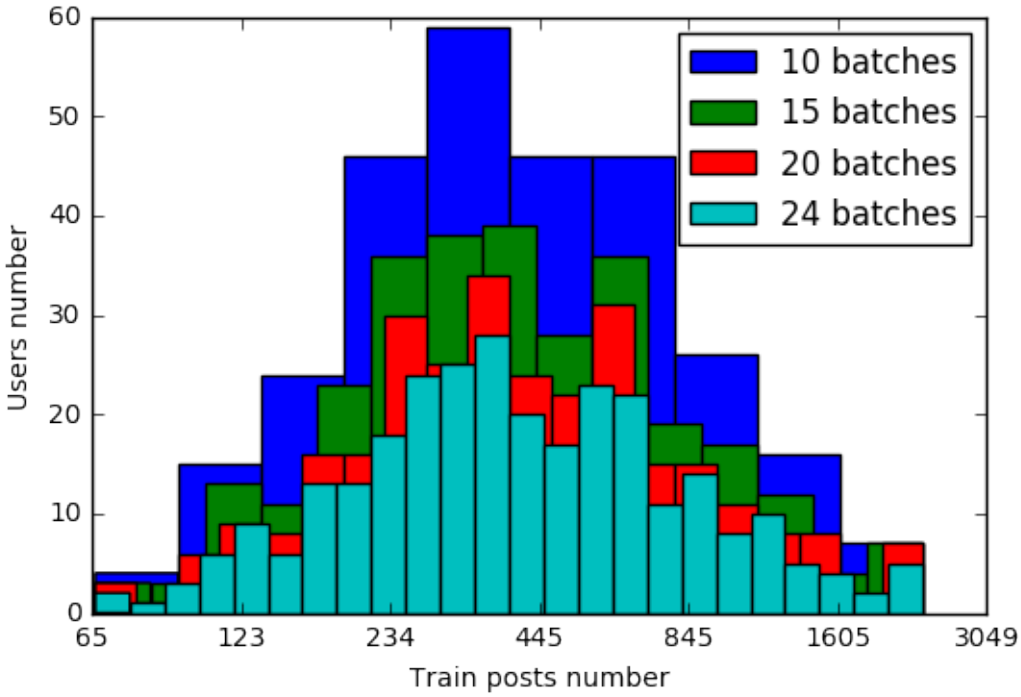

Рис. 2: Гистограмма пользователей по размеру обучающей выборки

5.1.1. Сравнение общих моделей

Для того, чтобы построить оптимальную модель персонализации, было проведено сравнение точности выбранных подходов на различных подмножествах данных.

Сначала было проведено сравнение точности моделей, обученных на данных для всех пользователей

	NDCG@5	NDCG@10	NDCG@30	NDCG@50	MAP@50
Логистическая регрессия	0,5122	0,5069	0,5915	0,6801	0,4220
RankSVM	0,5235	0,5164	0,6018	0,6861	0,4334
ListNet	0,4901	0,4933	0,5807	0,6676	0,4457
ListMLE	0,4954	0,4967	0,5864	0,6761	0,4461
LambdaMART	0.4902	0.4480	0.4519	0.5333	0.4311

Таблица 1: Сравнение общих моделей

Из Таблицы 1 видно, что по всем метрикам линейные общие модели дают лучший результат, чем более сложные.

5.1.2. Применимость персональных моделей

Следующая задача - исследование применимости персональных ранжирующих моделей для решения поставленной задачи. С целью выяснения целесообразности использования персональных моделей было проведено попарное сравнение результатов общей и персональной моделей одного вида. Ниже приведены таблицы сравнения моделей по взятым метрикам и графики разности результатов персональных и общей модели для пользователей, отсортированных по количеству данных в обучающей выборке.

	NDCG@5	NDCG@10	NDCG@30	NDCG@50	MAP
Общая	0,5122	0,5069	0,5915	0,6801	0,4220
Персональная	0,5339	0,5279	0,6028	0,7001	0,4439

Таблица 2: Логистическая регрессия. Персональные vs общая на всех данных

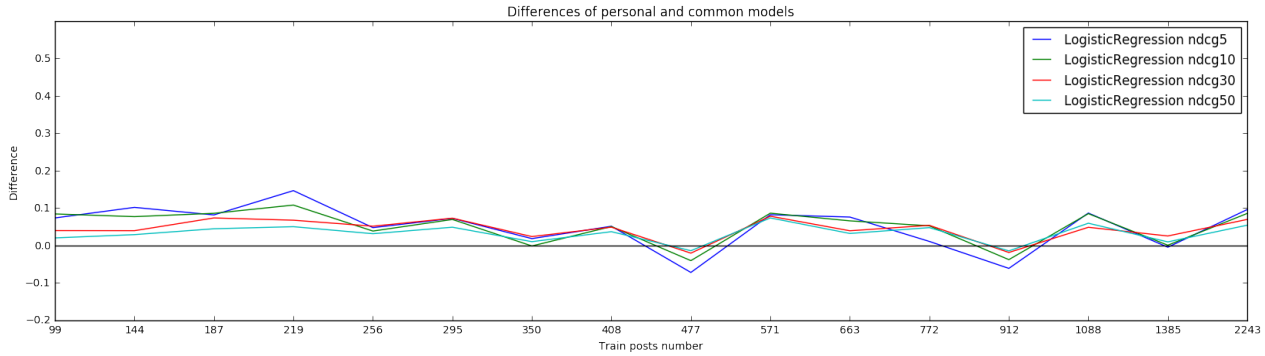

Рис. 3: Сравнение персональной и общей моделей Логистической Регрессии

Для логистической регрессии видно, что персональные модели дают незначительно лучший результат на всем диапазоне значений по большинству метрик.

	NDCG@5	NDCG@10	NDCG@30	NDCG@50	MAP
Общая	0,5313	0,5336	0,6093	0,6927	0,4334
Персональная	0,5412	0,5408	0,6145	0,7018	0,4453

Таблица 3: RankSVM. Персональные vs общая на всех данных

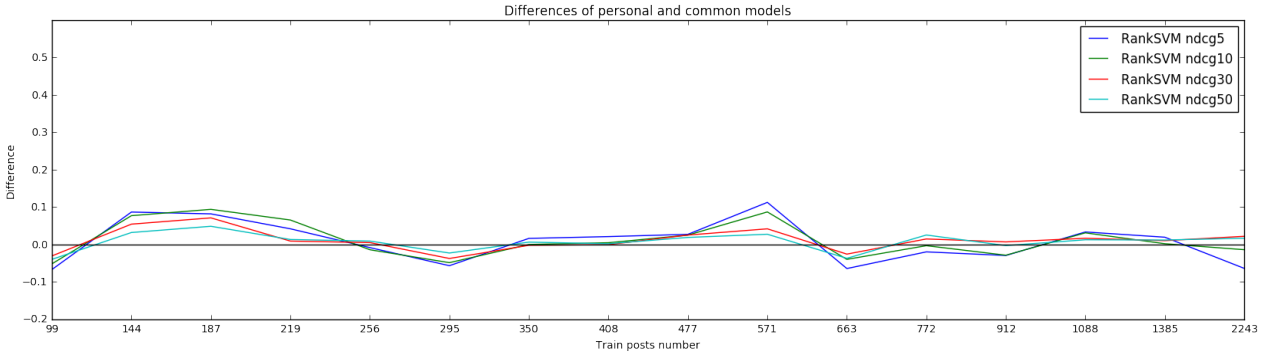

Рис. 4: Сравнение персональной и общей моделей RankSVM

Для модели RankSVM этот тренд не сохраняется, разность результатов близка к нулю для всех метрик. Так как в общей попарной модели в обучающую выборку входят только пары документов одного и того же пользователя, можно сказать, что персонализация в RankSVM присутствует уже и в общей модели, именно поэтому прирост не замечен.

	NDCG@5	NDCG@10	NDCG@30	NDCG@50	MAP
Общая	0,4901	0,4933	0,5807	0,6676	0,4457
Персональная	0,5579	0,5541	0,6334	0,7126	0,4739

Таблица 4: ListNet. Персональные vs общая на всех данных

Для списочной модели ListNet персональная модель работает сильно лучше общей по всем используемым метрикам, следовательно есть смысл использовать данную модель как часть итоговой комбинированной модели.

Для списочной модели ListMLE персональная модель работает лучше общей, в то же время результаты персональной модели ListNet лучше результатов персональной ListMLE, поэтому эта персональная модель в итоговый ансамбль выбрана не будет.

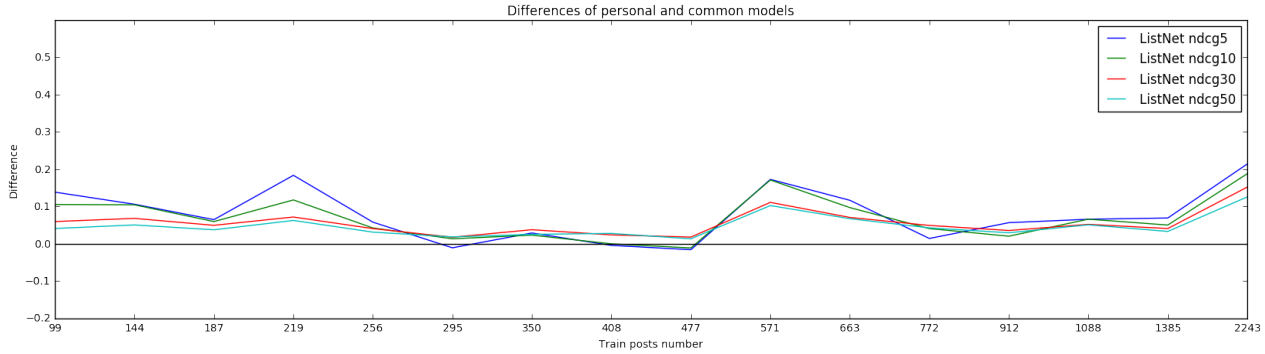

Рис. 5: Сравнение персональной и общей моделей ListNet

	NDCG@5	NDCG@10	NDCG@30	NDCG@50	MAP
Общая	0,4954	0,4967	0,5864	0,6761	0,4461
Персональная	0,5216	0,5229	0,6062	0,6927	0,4621

Таблица 5: ListMLE. Персональные vs общая на всех данных

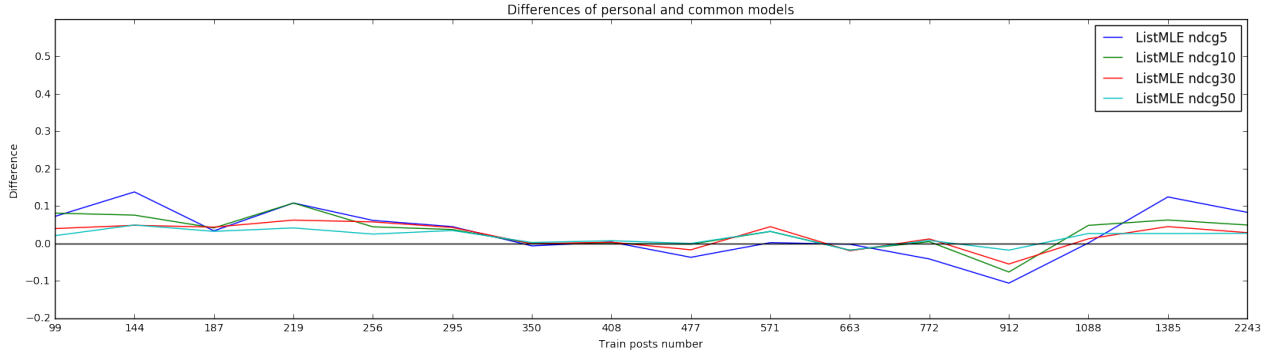

Рис. 6: Сравнение персональной и общей моделей ListMLE

Из проведенного исследования можно сделать несколько выводов:

1. Для списочных моделей персонализация дает больший прирост, чем для линейных.

2. Использование попарных персональных моделей нецелесообразно: с гораздо более длительным обучением прироста точности не произойдет.

3. Строгой зависимости разницы между персональной и общей моделями и размером обучающей выборки не наблюдается ни для одной из моделей.

4. Тем не менее для различных пользователей разность точности персональной и общей моделей варьируется.

5.1.3. Выбор итоговых моделей

При сравнении общих сетевых списочных моделей и линейных моделей лучшие результаты показали линейные модели, но так как при использовании персональной списочной модели наблюдается сильный прирост точности, имеет смысл сравнить с

общими линейными моделями уже персональные списочные и выяснить, для какого класса пользователей целесообразно использовать более сложные модели.

Сравнение будет проводиться по метрике ndcg@50 так как 50 записей достаточно для оценки актуальности ленты, в то же время средний пользователь гарантированно просмотрит 50 записей, листая ленту.

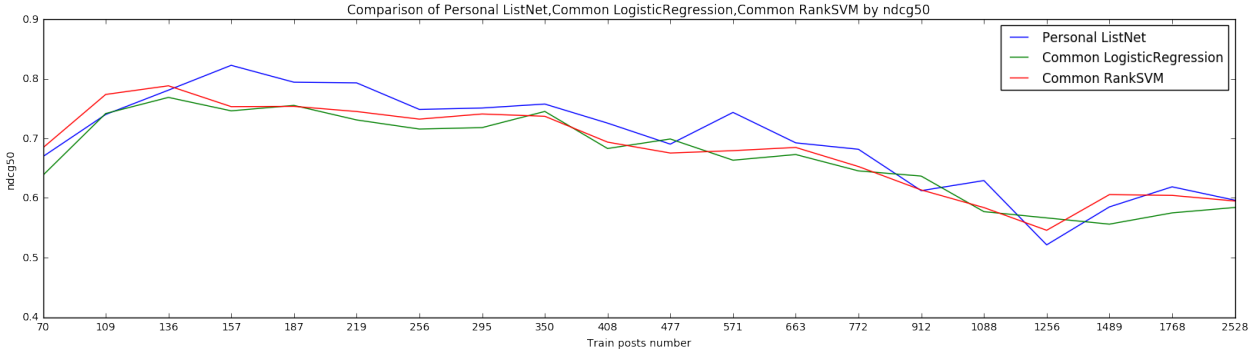

Рис. 7: Сравнение ListNet и общих линейных моделей

На графике можно увидеть, что для очень малого и очень большого количества данных целесообразно использовать линейные модели.

5.1.4. Итоги

После изучения полученных результатов было принято решение использовать комбинацию общей модели RankSVM и персональной модели ListNet

5.2. Построение модели

Итоговая модель является комбинацией общей и персональной моделей. В качестве персональной модели был взят списочный алгоритм ListNet, так как он показал наилучшие результаты для среднего активного пользователя. Данная модель использовалась для пользователей с размером обучающей выборки от 140 до 900. Модель обучалась на данных каждого пользователя по отдельности.

В качестве общей модели был взят попарный алгоритм RankSVM, так как он показал лучшие результаты для групп неактивных и гиперактивных пользователей. Модель использовалась для пользователей с размером обучающей выборки от 500 до 140 и от 900.

Тренировочные данные для алгоритма RankSVM были сформированы следующим образом: для каждого пользователя, для которого записей каждой из категорий присутствовало больше двух, случайно выбирались 3 поста каждой категории и из всех постов разных категорий составлялись пары. Пары составлялись следующим образом: признаки записи из худшей категории вычитались из признаков записи лучшей,

label проставлялся как 1 если первый пост лучше второго и 0 в противоположном случае. Так как записи из лучшей категории всегда берутся первыми, для того, чтобы избежать несбалансированности классов в 50% случаев порядок записей менялся.

5.3. Особенности реализации

5.4. Использованные технологии

1. Scala

На языке Scala написан код системы, представленной в работе.

2. Spark

Библиотека для обработки данных. Применялась по причине удобства при обработке больших объемов данных, которые невозможно единоразово загрузить в память.

3. Spark Mllib

Библиотека, в рамках которой реализованы базовые алгоритмы машинного обучения. Применялась при написании алгоритмов RankSVM и логистическая регрессия.

4. RankLib

Библиотека, реализованный в рамках которой алгоритм LambdaMART использовался в работе

5. Python

На языке Python написана часть оценочного модуля, т.к. при помощи следующих библиотек можно очень быстро и удобно сравнить разные модели по разным показателям.

6. Pandas

Библиотека для обработки табличных данных. Использовалась для представления и обработки результатов моделей, реализованных в модуле.

7. MatPlotLib

Библиотека, использованная для построения графиков в рамках работы

6. Апробация

Модель была запущена на данных за ноябрь и декабрь 2016 года, предоставленных социальной сетью ”Одноклассники”. Были проведены множественные эксперименты, в ходе которых для моделей варьировались параметры обучения, применялись различные виды сэмплирования данных.

Итоговая комбинация настраиваемых параметров обучения - ListNet: 200 итераций, шаг обучения 0.001; ListMLE - толерантность - 0.03, шаг обучения 0.0002; RankSVM - 50 итераций.

6.1. Сравнение результатов моделей

	NDCG@5	NDCG@10	NDCG@30	NDCG@50
Комбинация моделей	0,5597	0,5641	0,6352	0,7204
Общая ЛогистическаяРегрессия	0,5122	0,5069	0,5915	0,6801
Общий RankSVM	0,5235	0,5164	0,6018	0,6861
Персональный ListNet	0,5579	0,5541	0,6334	0,7126
Персональный ListMLE	0,5216	0,5229	0,6062	0,6927
LambdaMART	0.4902	0.4480	0.4519	0.5333
Без сортировки	0.1965	0.1978	0.2873	0.3763

Таблица 6: Сравнение рассмотренных в работе моделей с итоговой комбинированной моделью

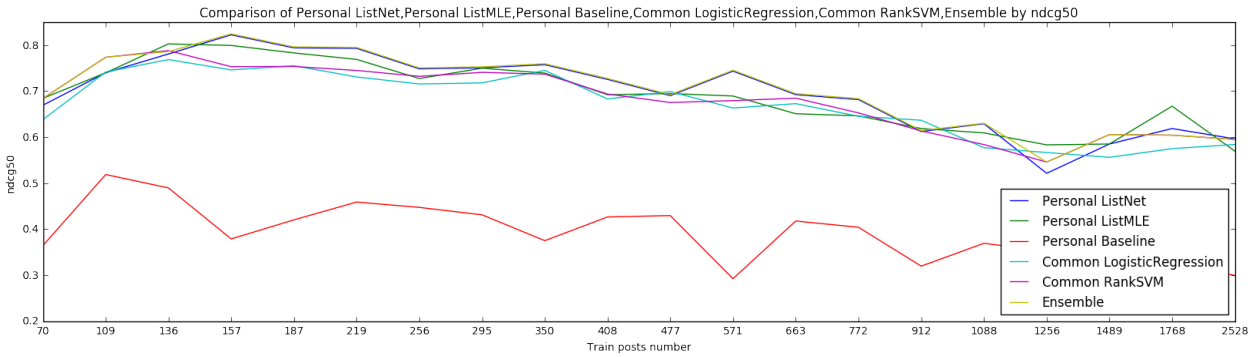

Рис. 8: Сравнение рассмотренных в работе моделей с бейзлайном

6.2. Разбиение пользователей

Интересно, что персональная модель ранжирования работает хуже линейной, если у пользователя слишком много тренировочных данных. Это говорит о том, что если пользователь беспорядочно реагирует на любой контент, в его поведении пропадают закономерности и персональная модель не может их уловить

Заключение

6.3. Результаты

В ходе работы были достигнуты следующие результаты.

1. Проведен анализ существующих решений в области ранжирования документов, выделены наиболее перспективные и представляющие наибольший интерес алгоритмы.

2. Разработана модульная архитектура ранжирующей системы.

3. Проведена обработка пользовательских данных, включающая в себя выделение признаков для тренировки ранжирующей модели, группировку данных и оценку реакции пользователя.

4. Реализована комбинированная ранжирующая модель на основе алгоритмов RankSVM и ListNet.

5. Проведена апробация реализованной системы, в ходе которой была достигнута точность NDCG@50 0,7204, что превышает точность остальных рассмотренных в работе моделей.

6.4. Выводы

Из результатов работы можно сделать следующие выводы. 1. Алгоритмы, используемые для ранжирования документов, могут использоваться для построения персонализованной новостной ленты пользователя социальной сети.

2. Для большинства умеренно активных пользователей персональная модель работает лучше, в то же время она не может уловить зависимости для малоактивных и гиперактивных пользователей.

3. Комбинация общей модели RankSVM и персональной модели ListNet решает поставленную задачу лучше, чем остальные рассмотренные в работе подходы.

6.5. Дальнейшее развитие

Один из возможных способов улучшения модели - кластеризация пользователей по интересам и по способу взаимодействия с контентом и построение отдельных моделей для получившихся групп пользователей.

В рамках данной работы не были решены задачи настройки временных параметров и параметров соотношения постов от групп и от пользователей. Сложность состоит в том, что на настоящее время подобная настройка производится при помощи длительных A/B тестов с проверкой таких показателей как количество ежедневно/ежемесячно заходящих пользователей, время, проведенное пользователями на

сайте, среднее количество заходов в день, а техники оптимизации по историческим данным нет. Провести же A/B тестирование возможности не было.

Также было бы интересно применить техники NLP и распознавания изображений/видео для выделения семантики записей при наличии данных о содержимом поста.

Список литературы

[1] Burges Christopher J.C. From RankNet to LambdaRank to LambdaMART: An Overview. — 2010.

[2] Fen Xia Tie-Yan Liu Jue Wang Wensheng Zhang Hang Li. Listwise Approach to Learning to Rank - Theory and Algoritm. — 2008.

[3] Kaggle. Mean Average Precision. — URL: `https://www.kaggle.com/wiki/MeanAveragePrecision`.

[4] Kaggle. Normalized Discounted Cumulative Gain. — URL: `https://www.kaggle.com/wiki/NormalizedDiscountedCumulativeGain`.

[5] Project The Lemur. RankLib. — URL: `https://sourceforge.net/p/lemur/wiki/RankLib/`.

[6] R. Herbrich K. Obermayer, Graepel T. "Large margin rank boundaries for ordinal regression," in Advances in Large Margin Classifiers, pp. 115–132. — 2000.

[7] Wikipedia. Chain rule. — URL: `https://en.wikipedia.org/wiki/Chain_rule`.

[8] Wikipedia. Hinge loss. — URL: `https://en.wikipedia.org/wiki/Hinge_loss`.

[9] Wikipedia. Kullback–Leibler divergence. — URL: `https://en.wikipedia.org/wiki/Kullback-Leibler_divergence`.

[10] Zhe Cao Tao Qin Tie-Yan Liu Ming-Feng Tsai Hang Li. Learning to Rank: From Pairwise Approach to Listwise Approach. — 2007.