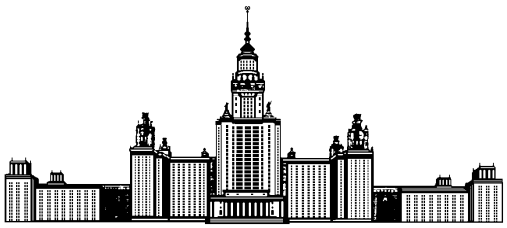

Московский государственный университет
имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Хальман Михаил Анатольевич

Методы персонализации показа объявлений в рекламной сети

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
д.ф.-м.н., профессор
К.В. Воронцов

Москва, 2015

Содержание

1 Введение

Контекстная реклама является основным источником дохода крупнейших IT-компаний.

Задача предсказания вероятности клика (вероятность клика сокращённо называют CTR) на объявление — ключевая задача в этой отрасли, хорошо решив которую мы сможем таргетировать рекламу точно и качественно. Это делает задачу прогноза CTR одной из центральных проблем для многих компаний. По решению этой задачи проводятся множество соревнований по машинному обучению с ценными призами. [1] [2]

Текстовая информация, которая содержится в заголовках баннеров и поисковых запросах пользователей может быть эффективно использована для улучшения качества прогноза.

Модель DSSM была предложена в статье [3] и была успешно использована для ранжирования поисковой выдачи. Модель восстанавливает скрытую семантическую структуру текстового запроса и текста страницы, строя векторное представление в пространстве маленькой размерности при помощи многослойной нейронной сети. Параметры сети настраиваются таким образом, чтобы предсказывать по какой из страниц кликнет пользователь.

Целью данной работы является адаптация модели DSSM для задачи предсказания CTR, реализация алгоритма обучения а также обзор методов построения признаковых описаний на основе текстов.

2 Постановка задачи и обозначения

2.1 Обозначения

Введём некоторые понятия, которые будут необходимы нам в дальнейшем.

- **Клик** — событие перехода пользователя по ссылке на рекламном объявлении.
- **Баннер** — конкретное рекламное объявление. Характеризуется отображаемым текстом рекламного предложения, ставкой (ценой, которую рекламодатель готов платить за клик) информацией о заказчике и сайте. Может содержать изображение или другой медиаконтент.
- **Хит** — одна загрузка страницы рекламной выдачи. Один хит содержит несколько событий загрузки баннеров.

- **Контекст** — дополнительная информация, которую можно использовать для предсказания. Под контекстом понимается введённый поисковый запрос, различная информация о пользователе (такая как регион, социально демографический сегмент), время суток или любые данные, которые могут использоваться для предсказания вероятности клика.

- **CTR (click-through rate)** — вероятность клика на баннер при условии его показа в рекламной выдаче.

- **CPC (cost-per-click)** — цена, которую рекламодатель платит за один клик по баннеру.

- **CPM (cost-per-mile)** — математическое ожидание заработанных денег за тысячу показов объявления. CPM = 1000 × CTR × CPC

2.2 Описание задачи

Задачу отбора рекламных объявлений для показа можно сформулировать следующим образом. Дано множество баннеров D и множество всевозможных контекстов $\mathcal{X}$ Требуется для заданного контекста $x \in \mathcal{X}$ отобрать некоторое маленькое подмножество баннеров $\{d_1, d_2, \ldots, d_k\} = D_{rel} \subset D$, релевантных x. В данной работе для упрощения будем считать, что число баннеров k фиксировано и задаётся «сверху». Обычно полагают $k \ll |D|$. Типичным соотношением являются $k = 3$ при $|D| \approx 10^5$.

Одним из основных критериев качества алгоритма является «кликабельность» (CTR) показанной выдачи. Также рассматривают другие критерии, такие как CPM (математическое ожидание денег, полученных за тысячу показов).

Стоит заметить, что точность предсказания CTR важна для корректного вычисления CPM. Таким образом важно не только хорошо отобрать объявления, но и правильно оценить вероятность клика. Поэтому на практике задачу отбора баннеров решают в формулировке предсказания CTR.

Пусть теперь дан контекст $x \in \mathcal{X}$ и баннер $d \in D$. Требуется, обучившись на исторических данных, предсказать вероятность клика $\mathrm{CTR}(x, d)$ пользователя с заданным контекстом x на рекламное объявление d.

Задача прогноза CTR характеризуется:

- Колоссальными (сотни гигабайт) объёмами данных.

- Большим количеством категориальных и текстовых признаков.

- Высокими требованиями к времени работы алгоритма предсказания.
- Наличием нескольких критериев качества.

3 Обзор существующих методов

Одними из наиболее распространённых методов машинного обучения, используемых для предсказания CTR, являются логистическая регрессия с хешированием признаков, а также различные композиции решающих деревьев. Полный и подробный обзор существующих методов приведён в статье [4].

3.1 Логистическая регрессия

Введём обозначения. Пусть обучающий лог состоит из n событий показов баннеров.

$\{x_i\}_{i=1}^n$, $\{d_i\}_{i=1}^n$ соответственно контекст и баннер i-ого события.

$\{y_i\}_{i=1}^n$ — вектор меток. $y_i = 1$, если для i-го события произошёл клик по рекламе, $y_i = -1$ иначе.

$\mathbf{f}(x, d) \in \mathcal{X} \times D \to \mathbb{R}^d$ — векторная функция, которая принимает на вход контекст и баннер и строит по ним вектор признаков для обучения.

$\mathbf{w} \in \mathbb{R}^d$ — вектор настраиваемых по данным коэффициентов.

Логистическая регрессия моделирует CTR по следующей формуле:

$$\mathrm{CTR}(x, d) = \frac{1}{1 + \exp\left(-\mathbf{w}^{\mathbf{T}}\mathbf{f}(x, d)\right)} \tag{1}$$

,

Веса $\mathbf{w}$ находятся как максимум регуляризованного правдоподобия:

$$\sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^{\mathbf{T}}\mathbf{f}(x_i, d_i)\right) + \frac{1}{2}\lambda\|\mathbf{w}\|_2^2 \to \min_{\mathbf{w}} \tag{2}$$

В статье [5] описан алгоритм хеширования признаков, который был реализован авторами статьи в библиотеке с открытым кодом Vowpal Wabiit, которая пользуется большой популярностью для задачи предсказания CTR. [6]

3.1.1 Хеширование признаков

Стандартный способ использования категориальных признаков заключается в модели dummy-кодирования, которая подразумевает преобразование категориального признака f_i, принимающего значение из конечного множества $\mathcal{F} : |\mathcal{F}| = d_i$ в бинарный вектор c_i длины d_i, в котором на всех позициях кроме одной (соответствующей значению признака) стоят нули. Например, пусть признак f_i принимает целые значения от 1 до 5. Тогда, по значению $f_i = 2$, мы построим вектор $(0, 1, 0, 0, 0)$

Однако, используя такую модель, очень сложно работать с признаками большой арности [1]. Например, если f_i это признак типа идентификатор пользователя или домен, который может принимать до 10^9 различных значений, по нему придётся строить и хранить вектор размерности 10^9, что не позволительно много.

Элегантное решение, предложенное в статье [5], заключается в построении вектора фиксированного размера для всех признаков и определении позиции единицы в этом векторе как значение хеш-функции, взятое по модулю размерности вектора. Таким образом для нескольких разных значений f_i может получиться один и тот же вектор. Техника получила название «хеширование признаков» или «hashing-trick».

Метод обладает рядом преимуществ. Во-первых, хеширование позволяет использовать произвольные признаки, даже принимающие миллиард различных значений и не думать о размерности пространства, задавая её априори исходя из вычислительных возможностей. Во-вторых, ошибки связанные с коллизиями при хешировании выполняют роль регуляризации и не позволяют настроить слишком переобученную линейную модель.

3.1.2 Достоинства метода логистической регрессии

- Хеширование признаков позволяет эффективно обрабатывать сколь угодно разреженные данные и произвольное число признаков.
- Существуют высокоэффективные параллельные онлайновые алгоритмы обучения. [7]
- Высокая скорость вычислений в итоговой формуле предсказания (предполагает сильную разреженность вектора $\mathbf{f}$)

[1] Под арностью признака подразумевается количество различных значений, которые может принимать данный признак

- Огромный объём данных позволяет настраивать сложные линейные модели с миллионами коэффициентов без переобучения.

3.2 Композиции решающих деревьев

Градиентный бустинг на решающих деревьях [8][9] — один из наиболее успешных и часто применяющихся на практике алгоритмов машинного обучения обладает высокой обобщающей способностью и хорошо работает на практике. Метод хорошо зарекомендовал себя и используется для задачи предсказания вероятности клика на рекламные объявления. [10] [11]

Решающие деревья отличаются тем, что крайне плохо работают с категориальными признаками, принимающими большое число значений. На вход такой композиции необходимо подавать веществennозначные признаки.

При помощи композиции решающих деревьев можно эффективно объединять результаты работы нескольких алгоритмов машинного обучения (например, линейных моделей). Для этого выборка делится на 2 части. Первая используется для обучения регрессионных моделей, а вторая — для обучения композиции.

Обучение происходит в два этапа.

Сначала регрессионные модели обучаются на первом наборе данных. Затем вычисляются предсказания этих моделей для всех объектов из второй выборки, после чего эти предсказания используются как признаки для обучения композиции решающих деревьев.

В результате получается модель, по качеству не уступающая каждой из исходных моделей.

Таким образом при помощи композиции решающих деревьев можно объединять предсказания нескольких линейных моделей, нейронных сетей или других алгоритмов, используя их предсказания на ряду с остальными признаками, используемыми для обучения.

4 Генерация признаков для обучения

Рекламное объявление d и контекст x содержат очень много информации, которая может быть использована для улучшения качества прогноза.

При помощи композиции деревьев можно объединять предсказания нескольких линейных моделей. Встаёт вопрос о том как построить вектор признаков $\mathbf{f}(x, d)$ для обучения логистической регрессии.

Вид признака	Пример	Способ представления для линейных моделей
Вещественные	Рейтинг сайта, число слов в запросе	Дискретизация, превращение в категориальный
Категориальные	Регион пользователя, URL страницы	Хеширование признаков
Текстовые	Текст запроса, заголовок объявления	Bag-of-words, TF-IDF и т.д.

Таблица 1: Виды признаков и методы обработки

Все признаки, которые характеризуют пару (x, d) можно условно разделить на три группы: вещественные, категориальные и текстовые. (Таблица 1).

Категориальные признаки преобразовываются в векторный вид при помощи хеширования. Для вещественных признаков хорошо работает алгоритм дискретизации: множество допустимых значений признака разбивается на корзинки, и признак преобразуется в категориальный, соответствующий индексу корзинки, в которую попало значение данного признака.

Такой способ предобработки вещественных признаков позволяет восстановить сложные нелинейные зависимости между значением признака и значением целевой функции. Нетрудно видеть, что настройка вектора коэффициентов линейной модели на таком признаке соответствует аппроксимации целевой функции кусочно-постоянной функцией одного аргумента (значения признака).

О построении текстовых признаков будет изложено ниже.

4.1 Методы векторного представления текстов

4.1.1 Модель мешка слов

Модель мешка слов подразумевает представление документа как категориальный признак, который принимает сразу несколько значений, соответствующих словам, которые встретились в тексте. Наличие того или иного слова в тексте является информативным признаком, характеризующим намерение пользователя и предложение рекламного объявления. Модель мешка слов имеет недостатки, связанные в первую очередь с слишком большой размерностью пространства и возможным наличием в тексте общеупотребимых и неинформативных слов, таких как союзы и предлоги, а также слов написанных с опечатками.

4.1.2 TF-IDF

Модель TF-IDF (term frequency — inverse document frequency), [12] [13] [14] [15] борется с проблемой общеупотребимых слов домножением частот на функцию монотонно убывающую по общей встречаемости слова в языке. Примером такой функции может быть $-\log(\frac{N_W}{N})$, где N_W — количество документов, в которых встретилось данное слово, N — количество документов в коллекции (в нашем случае это количество рекламных объявлений в базе).

Для коротких текстов, таких как поисковый запрос можно делать фильтрацию, отбрасывая слова со слишком маленьким IDF.

Все модели, основанные на простом сопоставлении слов обладают следующими недостатками:

- Слишком большая размерность пространства (размерность равна количеству слов в словаре, которое может достигать нескольких сотен тысяч на реальных данных) приводит к тому, что для каждого слова не набирается достаточно статистики для обучения адекватных закономерностей.
- Такие модели не могут восстановить похожести между разными словами, имеющими схожий смысл или обозначающими одно и то же. (Например такие как «машина» и «автомобиль»)
- В реальных данных существует проблема опечаток в поисковых запросах. Наличие опечатки в тексте приводит к тому, что слово в тексте интерпретируется моделью как совершенно новое, что приводит к сильной деградации метода.

4.1.3 Латентный семантический анализ (LSA)

Модель LSA [16] строит сингулярное разложение матрицы частот слов в документах, позволяя эффективно уменьшить размерность пространства параметров. Строится линейное преобразование векторов частот слов в текстах, минимизирующее среднеквадратичную ошибку при восстановлении исходного текста.

4.1.4 Вероятностное тематическое моделирование (PLSA, LDA, ARTM)

Модель PLSA (probabilistic latent semantic analysis) [17] и её модифицированная версия LDA (latent dirichlet allocation) [18] обладают рядом

преимуществ по сравнению с моделью LSA. Они также основаны на матричном разложении матрицы частот слов, но строят более интерпретируемую модель.

Идея заключается в том, что обучив вероятностную тематическую модель на одном наборе данных, можно использовать вектор тематического профиля документа вместо вектора частот слов в качестве вектора признаков для линейных моделей.

Также можно напрямую использовать расстояние между вектором запроса и вектором документа как вещественнозначный признак. Расстояние между векторами можно считать по косинусной метрике, расстоянию Кульбака-Лейблера или любой другой функции расстояния.

Различные supervised улучшения [19] [20] позволяют использовать информацию о метке класса в матричном разложении, что позволяет улучшить качество модели.

Одним из недостатком вероятностных тематических моделей является то, что они плохо работают для очень коротких текстов. Эту проблему можно решать разными способами, например увеличивая число тем или склеивая запросы в истории одного пользователя или заголовки баннеров одного заказа (рекламной кампании) одного рекламодателя. Всё это может приводить к деградации модели.

4.1.5 Глубинные нейросетевые модели

В последнее время большую популярность приобрели модели, снижающие размерность пространства параметров, основанные на обучении глубоких нейронных сетей. [21] [22] [23] В качестве сжатого представления используется вектор значений в нейронах на скрытом слое.

Глубинные нейросетевые модели обладают большим преимуществом в ситуации когда данных очень много, а тексты очень короткие. Как раз с такой ситуацией я столкнулся в задаче предсказания CTR.

Все вышеописанные методы построения признаковых описаний обладают рядом недостатков:

- Методы не учитывают информацию о целевой функции. Другими словами, параметры моделей оптимизируются таким образом, чтобы точнее восстанавливать исходные документы, а не для того, чтобы отделять релевантные данному запросу объявления от нерелевантных.

- Все описанные методы страдают от опечаток в текстовых запросах, а также требуют значительной предобработки текстов (восстановления главных форм слов, лемматизации)

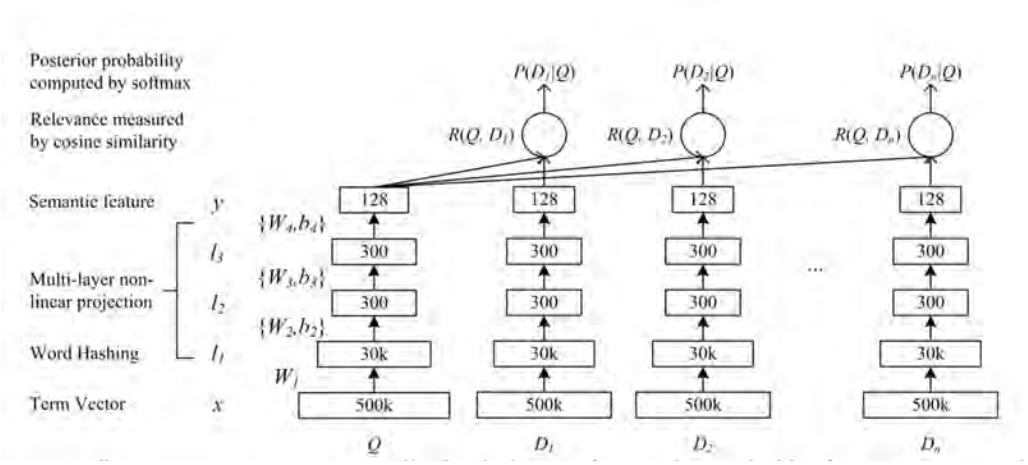

Рис. 1: Архитектура нейронной сети DSSM. На нижнем слое строится преобразование запроса в вектор частот трёхбуквенных n-грамм. Дальше идут три полносвязных слоя. На выходе релевантность моделируется как косинусное расстояние между преобразованием запроса и заголовка объявления. Нейронная сеть учится определять, по какому из показанных объявлений произошёл клик.

5 Deep Structured Semantic Model

Модель **Deep Structured Semantic Model (DSSM)** была предложена в 2013 году в статье [3] для поискового ранжирования. Она также строит скрытое векторное представление текста в пространстве меньшей размерности, но оптимизирует его таким образом, чтобы научиться предсказывать на какую из ссылок кликнет пользователь на основе косинусного расстояния между скрытыми представлениями для запроса и для текста заголовка объявления.

В начале для каждого текста строится векторное представление на основе трёх-буквенных n-грамм. В первую очередь текст склеивается в одно слово (удаляются все пробелы и любые другие символы кроме букв английского и русского алфавита). После чего строится векторное представление по аналогии с моделью мешка слов, только вместо слов выступают такие трёхбуквенные сочетания.

Такое преобразование позволяет решить несколько проблем.

Во-первых, оно решает проблему опечаток в тексте. Длинное слово, в котором сделана ошибка в одной букве, будет иметь схожее скрытое представление с оригинальным словом и мы сможем как-то использовать эту информацию. В отличие от модели мешка слов, где никакой связи

между такими словами мы не увидим и для слова с опечаткой просто не будет собрано никакой статистики.

Во-вторых, эта модель решает проблему длинных редкоупотребимых составных слов, таких как «кораблестроение». Модель, на подобии того как это делает маленький ребёнок, предполагая значение слова, которое он не знает, предположит, что это слово чем-то похоже на слова «корабли» и «строительство» и построит похожее скрытое представление как будто ей на вход пришёл запрос «строение кораблей».

Таким образом данное преобразование позволяет строить сколь-угодно масштабируемые модели для словарей произвольных размеров.

Далее над полученным скрытым представлением строится трёхслойная полносвязная нейронная сеть, которая преобразовывает вектор мешка 3-ёх-грамм x в вектор скрытого представления текста y размерности 128. По одной и той же формуле строится скрытое представление для текста запроса y_Q и для текста объявления y_D.

Обозначим через $\mathbf{f}(x; W)$ функцию, которая осуществляет преобразование текста x в вектор скрытого представления y_x. W — веса нейронной сети.

Определим релевантность между запросом Q и баннером D по следующей формуле.

$$R(D, Q) = \cos(y_D, y_Q) = \frac{y_D^T y_Q}{||y_D||||y_Q||} \tag{3}$$

Нетрудно видеть, что $-1 \leqslant R(D, Q) \leqslant 1$

Заметим, что если текст запроса Q совпадает с текстом объявления D, $R(D, Q) = 1$, т.к. $y_Q = y_D$.

Алгоритм обучения нейронной сети состоит в следующем. На вход подаётся запрос Q, текст объявления D_1, по которому пользователь с заданным запросом кликнул и тексты четырёх объявлений, по которым не было клика $D_2, \ldots, D_5$. В статье была предложена схема выбора текстов некликнутых ссылок случайно из всей базы существующих баннеров.

По формуле (3) вычисляется релевантность запроса текстовому документу. После softmax-преобразования получаем вероятность клика на документ D для запроса Q по формуле:

$$P(D|Q) = \frac{\exp(R(D, Q))}{\sum_{D'} \exp(R(D', Q))} \tag{4}$$

Веса нейронной сети обучаются методом стохастического градиента так, чтобы максимизировать логарифм правдоподобия нашей модели.

5.1 DSSM для предсказания CTR

Была предложена следующая идея. Обучить DSSM и использовать значение функции релевантности $R(D, Q)$как признак для готовой композиции решающих деревьев.

В экспериментах я пробовал следующие улучшения оригинальной модели:

- К словарю трёхбуквенных n-грамм я добавил вектор мешка слов простой конкатенацией. Это позволило улучшить качество модели, хотя немного замедлило обучение
- Было опробовано две схемы обучения.
  - Как в оригинальной статье, где алгоритм учился выбирать заголовок по которому произошёл клик среди нескольких заголовков, по которым не происходило клика случайно выбранных из базы всех объявлений.
  - Мотивированный целью повысить качество ранжирования баннеров, я пытался обучить нейронную сеть выбирать баннер, по которому произошёл клик среди всех баннеров, показанных пользователю.

6 Эксперименты

6.1 Данные и постановка эксперимента

Все эксперименты проводились с реальными данными о кликах на рекламу, показанную на страницах поиска над поисковой выдачей. Перед обучением была выполнена предобработка данных, удалены повторные события, а также события, для которых было предсказано, что они сделаны роботом. Были также отброшены все события показа рекламы, для которых не произошло ни одного клика ни на один баннер, показанный при данной загрузке страницы.

Эксперимент проводился по следующей схеме. Нейронная сеть обучалась на одних данных, после чего её прогноз добавлялся как признак к существующим признакам для использования жадным алгоритмом стохастического градиентного бустинга. В результате производилось сравнение качества на отложенной выборке двух алгоритмов — алгоритма, использующего признак, полученный при помощи нейронной сети и алгоритма, не использующего этот признак.

В случае, если первый алгоритм покажет лучшее качество, чем второй, можно будет сделать вывод о том, что предсказание нейронной сети приносит новую информацию, полезную для прогноза.

Данные были поделены на 3 части в хронологическом порядке. Нейронная сеть DSSM обучалась на истории за первые 3 недели февраля, тестировалась на последней неделе февраля. Обучение композиции решающих деревьев производилось на данных о кликах за март.

В совокупности для обучения нейронной сети использовалось около 100млн. записей общим объёмом 27 гигабайт. Для обучения бустинга использовалось 36 миллионов записей. Также наряду с предсказанием нейронной сети использовалось 254 других признака.

Исторически сложилось, что основной метрикой качества алгоритма предсказания является нормированный логарифм правдоподобия прогноза:

$$ll_p = \frac{ll_0 - ll}{C}$$

, где ll_0 — логарифм правдоподобия константного прогноза, ll — логарифм правдоподобия модели предсказания, C — число событий в выборке, по которым произошёл клик.

Основной метрикой качества является $\Delta ll_p = \frac{ll_p^1 - ll_p^0}{ll_p^0} \times 100\%$ — относительный прирост качества ll_p^1 алгоритма, использующего признак по сравнению с качеством ll_p^0 алгоритма, не использующего этот признак.

Также интересной метрикой качества является средний AUC внутри хита. Он вычисляется как усреднённое по всем хитам отношение количества правильно-упорядоченных пар баннеров к количеству всех пар баннеров. Пара баннеров считается правильно-упорядоченной, если среди двух баннеров тот, по которому произошёл клик имеет предсказание вероятности выше, чем другой.

Для настройки гиперпараметров нейронной сети я использовал ошибку классификации на отложенной выборке. Ошибку можно посчитать как долю событий, для которых баннер, по которому произошёл клик угадан неверно.

6.2 Обучение и подбор гиперпараметров модели

Обучение нейронной сети производилось пакетным методом стохастического градиентного спуска. Размер одного пакета для обучения составлял 1024 события. Темп обучения был выбран по ошибке на отложенной выборке. Оптимальное значение составило 0.01.

Все настраиваемые веса, согласно рекомендациям данным в [24] в нейронной сети инициализировались независимо случайными значениями из

Модель	Против случайного	Ранжирование для запроса
AUC по показам	0.59	0.526
Ошибка нейросети[2]	1.7%	32%
Δll_p	0.5%	0.01 %

Таблица 2: Основные результаты

равномерного распределения на интервале между $-\sqrt{6/(fanin + fanout)})$ и $\sqrt{6/(fanin + fanout)})$, где *fanin* и *fanout* — количество входных и выходных нейронов для данного слоя.

6.3 Результаты

Первая модель показала хорошее качество. Нейронная сеть научилась отличать заголовок, по которому произошёл клик от случайного заголовка, показав качество классификации 98.3% на тестовой выборке.

В моих экспериментах вторая модель работала гораздо хуже. Ожидания, что она будет лучше ранжировать рекламную выдачу не оправдались. Это видно из того, что AUC внутри одной выдачи крайне низкий (0.526). По всей видимости, это связано с тем, что все объявления внутри одного показа уже релевантны запросу, а одной только текстовой информации недостаточно для того, чтобы определить на какое из трёх объявлений кликнет пользователь.

7 Заключение

- Модель DSSM была успешна адаптирована для задачи предсказания CTR.
- Было показано, что применение объединённого признакового описания из вектора частот триграмм и вектора частот слов даёт выигрыш качества предсказания CTR.
- При помощи DSSM удалось повысить нормированный логарифм правдоподобия итоговой формулы на 0.5%.

Список литературы

[1] Display Advertising Challenge. — URL: http://http://www.kaggle.com/c/criteo-display-ad-challenge.

[2] Click-Through Rate Prediction. — URL: http://http://www.kaggle.com/c/avazu-ctr-prediction.

[3] Learning deep structured semantic models for web search using clickthrough data / Po-Sen Huang, Xiaodong He, Jianfeng Gao et al. // Proceedings of the 22nd ACM international conference on Conference on information & knowledge management / ACM. — 2013. — P. 2333–2338.

[4] Chapelle Olivier, Manavoglu Eren, Rosales Romer. Simple and scalable response prediction for display advertising // ACM Transactions on Intelligent Systems and Technology (TIST). — 2014. — Vol. 5, no. 4. — P. 61.

[5] Feature hashing for large scale multitask learning / Kilian Weinberger, Anirban Dasgupta, John Langford et al. // Proceedings of the 26th Annual International Conference on Machine Learning / ACM. — 2009. — P. 1113–1120.

[6] Langford John, Li L, Strehl A. Vowpal wabbit // URL https://github.com/JohnLangford/vowpal_wabbit/wiki. — 2011.

[7] Ad click prediction: a view from the trenches / H Brendan McMahan, Gary Holt, David Sculley et al. // Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining / ACM. — 2013. — P. 1222–1230.

[8] Friedman Jerome H. Greedy function approximation: a gradient boosting machine // Annals of statistics. — 2001. — P. 1189–1232.

[9] Friedman Jerome H. Stochastic gradient boosting // Computational Statistics & Data Analysis. — 2002. — Vol. 38, no. 4. — P. 367–378.

[10] Trofimov Ilya, Kornetova Anna, Topinskiy Valery. Using boosted trees for click-through rate prediction for sponsored search // Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy / ACM. — 2012. — P. 2.

[11] Dembczynski Krzysztof, Kotlowski Wojciech, Weiss Dawid. Predicting ads click-through rate with decision rules // Workshop on targeting and ranking in online advertising. — Vol. 2008. — 2008.

[12] Sparck Jones Karen. A statistical interpretation of term specificity and its application in retrieval // Journal of documentation. — 1972. — Vol. 28, no. 1. — P. 11–21.

[13] Salton Gerard, Buckley Christopher. Term-weighting approaches in automatic text retrieval // Information processing & management. — 1988. — Vol. 24, no. 5. — P. 513–523.

[14] Salton Gerard. Developments in automatic text retrieval // Science. — 1991. — Vol. 253, no. 5023. — P. 974–980.

[15] Manning Christopher D, Raghavan Prabhakar, Schütze Hinrich. Scoring, term weighting and the vector space model // Introduction to Information Retrieval. — 2008. — Vol. 100.

[16] Indexing by latent semantic analysis / Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer et al. // JAsIs. — 1990. — Vol. 41, no. 6. — P. 391–407.

[17] Hofmann Thomas. Probabilistic Latent Semantic Analysis // In Proc. of Uncertainty in Artificial Intelligence, UAI'99. — 1999. — P. 289–296.

[18] Blei David M, Ng Andrew Y, Jordan Michael I. Latent dirichlet allocation // the Journal of machine Learning research. — 2003. — Vol. 3. — P. 993–1022.

[19] Mcauliffe Jon D, Blei David M. Supervised topic models // Advances in neural information processing systems. — 2008. — P. 121–128.

[20] Zhu Jun, Ahmed Amr, Xing Eric P. MedLDA: maximum margin supervised topic models for regression and classification // Proceedings of the 26th Annual International Conference on Machine Learning / ACM. — 2009. — P. 1257–1264.

[21] Hinton Geoffrey E, Salakhutdinov Ruslan R. Reducing the dimensionality of data with neural networks // Science. — 2006. — Vol. 313, no. 5786. — P. 504–507.

[22] Bengio Yoshua. Learning deep architectures for AI // Foundations and trends® in Machine Learning. — 2009. — Vol. 2, no. 1. — P. 1–127.

[23] Salakhutdinov Ruslan, Hinton Geoffrey. Semantic hashing // International Journal of Approximate Reasoning. — 2009. — Vol. 50, no. 7. — P. 969–978.

[24] Orr Genevieve B, Müller Klaus-Robert. Neural networks: tricks of the trade. — Springer, 2003.