МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«Южный федеральный университет»

Институт математики, механики
и компьютерных наук им. И. И. Воровича

Кафедра прикладной математики и программирования

Игнатова Анастасия Алексеевна

ИСПОЛЬЗОВАНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ПРОГНОЗИРОВАНИЯ ВЕРОЯТНОСТИ ЗАИНТЕРЕСОВАННОСТИ ПОЛЬЗОВАТЕЛЯ КОНТЕНТОМ В РАМКАХ ВЕБ САЙТА

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
по направлению подготовки
02.03.02 – Фундаментальная информатика и информационные технологии

Научный руководитель –
к.т.н. доцент Чердынцева Марина Игоревна

Допущено к защите:
заведующий кафедрой _____ Угольницкий Г.А.

Ростов-на-Дону – 2020

ОТЗЫВ

о выпускной квалификационной работе бакалавра

Игнатовой Анастасии Алексеевны на тему

«Использование нейронных сетей для прогнозирования вероятности заинтересованности пользователя контентом в рамках веб сайта»

Работа Игнатовой А.А посвящена проблеме решения задач прогнозирования с использованием нейронных сетей. В качестве предметной области рассматривалась задача построения рекомендательной системы на основе данных о действиях пользователей сайта «Яндекс.Дзен». Выбор предметной области определялся возможностью получить данные для обучения нейронной сети и для проверки качества результатов.

На основе сопоставления возможных методов решения поставленной задачи, а также анализа исходных данных студенткой был выбран метод градиентного бустинга. Разработаны две модели с разными параметрами. Также предложено расширить исходные наборы данных путем введения агрегатного показателя, что позволило повысить качество предсказаний моделей.

Все поставленные задачи были решены. Работа выполнялась студенткой регулярно. Она проявила большую заинтересованность и самостоятельность в процессе выполнения работы. Некоторые замечания по оформлению текста работы не являются существенными и не влияют на качество и объем проделанной работы

Считаю, что работа Игнатовой А.А. выполнена на достаточном профессиональном уровне и заслуживает оценку «отлично» (95 баллов).

16 июня 2020 г.

Доцент кафедры ПМиП, к.т.н.  Чердынцева М.И.

Задание на выпускную квалификационную работу

Направление подготовки: прикладная математика и информатика

Образовательная программа бакалавриата: Фундаметальная информатика и информационные технологии

Соискатель: Игнатова А.А.

Научный руководитель: доцент Чердынцева М.И.

Год защиты: 2020

Тема работы: Использование нейронных сетей для прогнозирования вероятности заинтересованности пользователя контентом в рамках веб сайта

Цель работы: Исследование методов построения рекомендательной системы на основе данных о действиях пользователей

Задачи работы:

- изучение литературы по теме выпускной квалификационной работы;

- программная реализация моделей для предсказания кликов пользователя;

- экспериментальная проверка качества решений применением моделей к тестовому набору данных;

- сравнительный анализ и интерпретация результатов исследований

Заведующий кафедрой Г.А. Угольницкий

Научный руководитель М.И. Чердынцева

Студент А.А. Игнатова

13 февраля 2020 г.

АНТИПЛАГИАТ
ТВОРИТЕ СОБСТВЕННЫМ УМОМ

Южный Федеральный Университет

СПРАВКА
о результатах проверки текстового документа на наличие заимствований

Проверка выполнена в системе Антиплагиат.ВУЗ

Автор работы	**Игнатова Анастасия Алексеевна**
Подразделение	Институт математики, механики и компьютерных наук им. И. И. Воровича
Тип работы	Выпускная квалификационная работа
Название работы	Использование нейронных сетей для прогнозирования вероятности заинтересованности пользователя контентом в рамках веб сайта
Название файла	ВКР_Игнатова_Анастасия_Алексеевна.pdf
Процент заимствования	**9.84 %**
Процент самоцитирования	**0.00 %**
Процент цитирования	**13.04 %**
Процент оригинальности	**77.12 %**
Дата проверки	**13:04:20 17 июня 2020г.**
Модули поиска	Модуль поиска ИПС "Адилет"; Модуль выделения библиографических записей; Сводная коллекция ЭБС; Модуль поиска "Интернет Плюс"; Коллекция РГБ; Цитирование; Модуль поиска переводных заимствований; Модуль поиска переводных заимствований по elibrary (EnRu); Модуль поиска переводных заимствований по интернет (EnRu); Коллекция eLIBRARY.RU; Коллекция ГАРАНТ; Коллекция Медицина; Диссертации и авторефераты НББ; Модуль поиска перефразирований eLIBRARY.RU; Модуль поиска перефразирований Интернет; Коллекция Патенты; Модуль поиска "ЮФУ"; Модуль поиска общеупотребительных выражений; Кольцо вузов
Работу проверил	Чердынцева Марина Игорьевна
	ФИО проверяющего
Дата подписи	
	Подпись проверяющего

Чтобы убедиться в подлинности справки, используйте QR-код, который содержит ссылку на отчет.

Ответ на вопрос, является ли обнаруженное заимствование корректным, система оставляет на усмотрение проверяющего. Предоставленная информация не подлежит использованию в коммерческих целях.

Оглавление

Введение

В ходе выполнения выпускной квалификационной работы мной была рассмотрена прикладная задача, в рамках которой были построены рекомендательные системы на основе данных о действиях пользователей в персональной ленте рекомендаций сервиса «Яндекс.Дзен».

В экономике и маркетинге актуальной является задача упрощения обработки больших массивов данных с информацией о поведении пользователей на веб-сайтах, поскольку это позволяет определить эффективность рекламной кампании. В частности, для оценки релевантности контекстных статей, объявлений и баннеров запросам целевой аудитории.

Поиск баннеров для показа на странице начинается, когда авторизованный пользователь вводит запрос в пользовательскую систему. На практике места на странице для рекламы меньше, чем количество баннеров-кандидатов на показ. Для того, чтобы отобрать наиболее подходящие, выбираются баннеры, которые имеют наибольшую оценку релевантности для этого пользователя. Релевантность определяется как произведение стоимости рекламы на вероятность нажатия на баннер этим пользователем (клик). Экономический смысл формулы заключается в том, что произведение вероятности клика и ставки — это ожидаемая прибыль от показа наиболее релевантного предложения. Предсказание вероятности клика на баннер является важной задачей, решение которой необходимо для организация эффективного отбора баннеров.

С этой задачей успешно справляются современные нейронные сети, позволяющие проводить анализ целевой аудитории и предсказание их платежеспособности, а иногда и ускорить их.

1 Использование нейронных сетей для решения задачи регрессии

1.1 Постановка задачи

Целью выпускной квалификационной работы является исследование методов построения рекомендательной системы на основе данных о действиях пользователей в рамках сайта «Яндекс.Дзен».

Для достижения поставленной цели необходимо было решить следующие задачи:

- изучить литературу по теме выпускной квалификационной работы, а именно получить теоретические знания о нейронных сетях и маркетинге;
- изучить возможные топологии нейронных сетей и библиотек для работы с ними;
- реализовать модели для предсказания поведения пользователя вебсайта;
- экспериментально проверить качество решений применением моделей к тестовому набору данных;
- провести сравнительный анализ и интерпретация результатов исследований.

1.2 Обзор использованных источников и технологий

Машинное обучение используется для решения множества задач, а именно: ее обработки (ярким примером можно назвать распознавание текста, используемое on-line переводчиками Google Translate и Яндекс Переводчик), поиску (например сервис «BBC News Text Search» запущенный Oxford Visual Geometry group, который производит поиск релевантных видеосюжетов BBC по заданному ключевому слову) и предсказанию (компания LBS Capital Management достигла больших успехов, используя нейросети для прогнозирования цен акций).

Для работы с нейронным сетями традиционно используется язык программирования Python, для которого реализовано большинство библиотек для машинного обучения, например NumPy (в ней реализованы оптимизированные для работы с многомерными массивами вычислительные алгоритмы) и Pandas (областью применения являются задачи анализа и моделирования данных).

Помимо общих сведений об основных методах машинного обучения, примеров алгоритмов, которые применяются на практике, для работы с нейронными сетями необходимо иметь сведения о критериях оценки качества моделей обучения. Важной частью является математика нейронных сетей, а именно алгеброй, теорией вероятности, вычислительной математикой.

1.3 Изучение предметной области

Для решения задачи используется набор данных, содержащий информацию о действиях пользователей в персональной ленте рекомендаций «Яндекс.Дзена». Набор данных (датасет) предлагалось использовать для решения задачи «Предсказание кликов пользователя» чемпионата по программированию machine learning трека. Эти данные иллюстрируют взаимодействие пользователей с некоторыми документами, показанными им в ленте рекомендаций. Они содержат такие столбцы-идентификаторы: id взаимодействия; id документа, показанного пользователю; id автора статьи; id пользователя; id i-ой темы документа и степень принадлежности документа данной теме; а также столбец, показывающий факт клика пользователя на документ.

Таким образом исходный тестовый датасет состоит из 14 столбцов и 112 тысяч строк, тренировочный из 15 столбцов и 1 миллион и 900 тысяч строк. Разница объема информации обусловлена стандартным разбиением исходных полных данных, когда выбираются 80% данных и на их основе обучается модель. Оставшиеся 20% используют для проверки корректности обучения.

Предсказание поведения пользователя – это задача регрессионного анализа. Существует множество математических методов, комбинация которых может привести к решению. Однако, такой метод требует аккуратности и часто сводится к перебору, поэтому задачи регрессионного анализа решаются нейросетевым подходом.

2 Решение задачи от Яндекса

2.1 Исследование задачи

Традиционно предоставляются наборы данных, которые самостоятельно необходимо разделить на тестовые и обучающие. В этой задаче сразу предоставлялись два датасета, при этом по условию гарантировалось, что все встречающиеся в тестовом датасете значения встречаются и в тренировочном наборе данных.

Каждая строка соответствует взаимодействию некоторого пользователя с некоторым документом, показанным ему в ленте рекомендаций.

Необходимо построить модель для предсказания выбора рекламы пользователем и обучить эту модель на тренировочном и применить к тестовому датасету.

Учитывая, что все сущности (документы, пользователи, источники, темы) одинаковые как в тренировочном, так и в тестовом датасете, данные не требуют дополнительного анализа случая, когда информация о взаимодействии пользователей с документами из тестового датасета неизвестна.

Существует множество методов для решения данной задачи. Самый популярный – коллаборативная фильтрация (Collaborative Filtering) [1]. Поскольку нет информации о том, как были собраны данные, нет уверенности в том, что данные не разрежены. Это может повлечь проблему холодного старта [1].

Поэтому в качестве модели использованы взвешенные суммы деревьев регрессии. Модель была построена методом градиентного бустинга.

2.2 Градиентный бустинг

Рассмотрим формальное описание метода градиентного бустинга. Пусть дана выборка пар разрядности N , где 1-ый элемент принадлежит пространству объектов, а второй — пространству ответов. Наша задача найти функцию $F(x)$, действующую из пространства объектов в пространство ответов, такую,

чтобы минимизировать суммарную ошибку, подсчитанную на основе функции потерь.

$$\text{Loss} = \sum_{\text{i}=1}^{N} \Psi\big(y_i, \text{F}(x_i)\big), \qquad \text{где } \Psi(\text{y}, \text{F}) — \text{ функция потерь.}$$

Эту функцию $F(x)$ будем строить итеративно для $m = 1, 2, \ldots, M$. На каждой итерации m будем добавлять к уже построенной функции F дерево регрессии, которое задается параметрами: вес дерева β_m и функция $h(x, \alpha_m)$ от пространства объектов и параметра, определяющего дерево регрессии, а именно предикатов в его вершинах и значений в листьях. При этом все добавляемые деревья имеют одинаковую форму, которая фиксируется заранее. Такой параметр настраивается так, чтобы минимизировать функционал ошибки:

$$(\beta_m, \alpha_m) = \arg min_{\beta,\alpha} \sum_{i=1}^{N} \Psi(y_i, \text{F}_{m-1}(x_i) + \beta\text{h}(x_i, \alpha))$$

Для ее оптимизации воспользуемся методом градиентного спуска, то есть параметр α будем искать так, чтобы $h(x, \alpha)$ было как можно ближе по направлению к антиградиенту функционала ошибки:

$$\alpha_m = \arg min_{\beta,\alpha} \sum_{i=1}^{N} [-g_m(x_i) - \beta\text{h}(x_i, \alpha)]^2$$

$$g_m(x_i) = [\frac{\partial \Psi(y_i, \text{F}(x_i))}{\partial\ \text{F}(x_i)}]_{\text{F}(x)=\text{F}_{m-1}(x)}, i = 1..N$$

Осталось вычислить оптимальную длину шага:

$$\rho_m = \arg min_{\rho} \sum_{i=1}^{N} \Psi(y_i, \text{F}_{m-1}(x_i) + \rho\text{h}(x_i, \alpha_m))$$

Таким образом, общая схема градиентного бустинга имеет следующий вид, опишем с помощью картинки (рис.1):

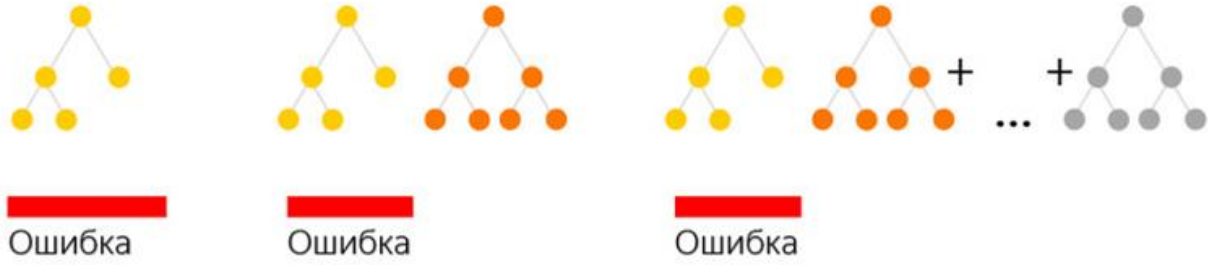

Рис.1. Общая схема градиентного бустинга

Произвольно созданные деревья, заданной формы с высокой ошибкой складываем поэтапно N раз, улучшая таким образом решающее дерево и уменьшая значение функции ошибки.

На основе этого алгоритма инженерами Яндекса была разработана библиотека машинного обучения CatBoost [3–10], которая используется для поиска, рекомендательных систем, персонального помощника, самоуправляемых автомобилей, прогнозирования погоды и многих других задач в Яндексе и в других компаниях. Данная разработка доступна как библиотека с открытым исходным кодом.

2.3 Определение топологии сети

Построение нейронной сети начинается с определения топологии сети. Принимать решение об архитектуре нужно исходя из определенной задачи. Представление данных формируют задачу регрессии [11]. Модель для решения задач этого класса тренируют с использованием учителя.

Выделяют два основных подхода в выборе количества слоев нейронной сети: поверхностное обучение, где используется несколько слоев (обычно не более трех) и глубокое или многослойное обучение, где обычно используются десятки и сотни слоев. Стоит отметить, что данные недостаточно обширные для использования глубокого обучения, поскольку это может повлечь за собой переобучение модели.

Для сравнения результативности моделей будут использоваться разные функции потерь.

Класс задач регрессии предполагает предсказание значения некоторой функции, определяющей поведение пользователя относительно статей сайта.

При обучении с учителем (supervised learning) учителю обучаемой сети подается тренировочный набор данных из исходного датасета. Его задача в том, чтобы продолжить известные ответы на новый опыт, выраженный в виде тестового набора данных. Важным условием в обучении модели является похожесть тренировочных и тестовых наборов на данные, где будет применяться обученная модель, для успешного обобщения данных.

После выбора общей структуры необходимо определить параметры сети. Подбор осуществляется экспериментально.

Для понимания эффективности работы нейронной сети нужно иметь возможность измерить расхождение реальных результатов и ожидаемых [12–13]. Для этого используется функция потерь, которая вычисляет оценку расстояния между предсказанием сети и истинным значением. Эта характеристика отражает, насколько хорошо сеть справилась с данным конкретным примером. Параметры сети корректируются с учетом весов обучающего вектора для уменьшения потерь.

Изначально веса сети имеют произвольные значения и оценка потерь очень высока, так как реализована последовательность случайных преобразований. Но чем больше данных обрабатывает сеть, то есть, чем больше проходит циклов обучения, тем лучше веса корректируются и минимизируется функция потерь. Таким образом, обученная сеть имеет наименьшие потери, а следовательно, возвращает результаты, максимально неотличимые от исходных.

Сети, обучающиеся с учителем, просматривают выборку множество раз, при этом один полный проход по выборке называется эпохой обучения. То есть исходная выборка делится на n частей по m образцов в каждой. При обучении сеть совершает n итераций, которые и называются эпохами, для каждой итерации сеть определяет и изменяет значения весов на ней. Если после какой-либо эпохи сеть показывает хороший результат на обучающих данных и

плохой на новых, значит, сеть перестала выполнять обобщение и просто «запоминает» обучающие данные. Такие ситуации называют переобучением сети (overfitting), тогда обучение сети прекращают.

В качестве модели для данного класса задач подходит использование взвешенной суммы деревьев регрессии, построенной методом градиентного бустинга.

2.4 Выбор инструментария

Для написания программы был выбран язык программирования Python версии 3.6. Код был написан в веб-приложении Jupyter Notebook, поскольку он позволяет запускать код модульно, сохраняя промежуточные результаты. Использовался дистрибутив Anaconda для упрощения работы с пакетами и их развертыванием.

Один из способов решить поставленную задачу — использовать пакет CatBoost — библиотеку для градиентного бустинга, основным преимуществом которой является возможность работать и с числовыми признаками, и с категориальными признаками. Бустинг — подход, который комбинирует слабые функции, то есть имеющие невысокую обобщающую способность, строящуюся в ходе итеративного процесса (на каждой итерации новая модель обучается с использованием данных об ошибках предыдущих). Результирующая функция — это линейная комбинация базовых, слабых моделей.

Исходный датасет содержит числовые и категориальные признаки [14], а поскольку задача регрессии, где по заданному набору признаков наблюдаемого объекта необходимо спрогнозировать вероятность наблюдения бинарной переменной. Для предсказания выбора рекламы пользователем использование библиотеки CatBoost оправдано.

3 Проведение эксперимента

3.1 Подготовка данных

Библиотека CatBoost использует в качестве входных данных числовые и категориальные признаки. Рассмотрим исходный датасет (рис. 2)

	sample_id	item	publisher	user	topic_0	topic_1	topic_2	topic_3	topic_4	weight_0	weight_1	weight_2	weight_3	weight_4	target
0	0	531	147	2925	411	477	618	249	460	27	18	9	8	7	0
1	1	1574	260	2981	212	287	382	302	51	27	11	2	1	0	0
2	2	940	394	1230	145	150	212	170	174	7	6	6	5	5	0
3	3	52	520	2597	201	283	618	249	617	35	33	30	11	9	1
4	4	766	55	1680	362	150	477	305	388	51	15	13	10	9	1

Рис. 2. Изначальный вид данных для обучения

Видно, что данные изначально полные, то есть не содержат NULL значений или другого мусора, а значит на них уже можно обучать модель.

Для сравнения работы моделей, построение которых будет описано ниже, удобно добавить вспомогательный столбец. Это приводит к улучшению работы моделей.

Введем параметр CTR (click-through rate, показатель кликабельности), для подсчета которого существует формула [15]:

$$CTR = \frac{total\ number\ of\ clicks}{total\ number\ of\ impressions} * 100\%$$

Коэффициент CTR — это соотношение кликов по рекламе к числу её показов.

Код добавления столбца:

```
Counter_list_of_train_items = Counter(List_of_train_items)
for n in range(train_count):
    if train_df['target'][n]==1:
      List_of_test_items.append(train_df['item'][n])
Counter_list_of_test_items = Counter(List_of_test_items)
for n in range(train_count):
    CTR_list_train.append(
Counter_list_of_test_items.get(train_df['item'][n])
          /Coun-
ter_list_of_train_items.get(train_df['item'][n]))
for n in range(test_count):
    CTR_list_test.append(
Counter_list_of_test_items.get(test_df['item'][n])
/Counter_list_of_train_items.get(test_df['item'][n]))
MyFile=open('output_train.txt', 'w')
```

```
for element in CTR_list_train:
     MyFile.write(str(round(element*1000000)))
     MyFile.write('\n')
MyFile.close()
MyFile2=open('output_test.txt', 'w')
for element in CTR_list_test:
     MyFile2.write(str(round(element*1000000)))
     MyFile2.write('\n')
MyFile2.close()
```

Поскольку CatBoost не работает с типом данных float, выполним округление значений в столбце CTR, а для увеличения точности увеличиваем его в 100 раз перед округлением. Заменяем содержимое тестового и тренировочного датасета.

Теперь данные выглядят следующим образом (рис. 3)

	sample_id	item	publisher	user	topic_0	topic_1	topic_2	topic_3	topic_4	weight_0	weight_1	weight_2	weight_3	weight_4	target	ctr
0	0	531	147	2925	411	477	618	249	460	27	18	9	8	7	0	153005
1	1	1574	260	2981	212	287	382	302	51	27	11	2	1	0	0	137838
2	2	940	394	1230	145	150	212	170	174	7	6	6	5	5	0	218391
3	3	52	520	2597	201	283	618	249	617	35	33	30	11	9	1	190272
4	4	766	55	1680	362	150	477	305	388	51	15	13	10	9	1	312749

Рис. 3. Тестовые данные после изменений

3.2 Настройка параметров

Для построения модели в библиотеке CatBoost используется класс классификатор CatBoostRegressor, требующий определение параметров модели.

Параметр eval_metric вычисляет заданную метрику по необработанным аппроксимированным значениям формулы и значениям метки. AUC — Area Under Curve — площадь под кривой, статистический показатель, эквивалентный вероятности, что классификатор присвоит больший вес случайно выбранной положительной сущности, чем случайно выбранной отрицательной. Здесь показатель AUC используется для того, чтобы сравнивать модели, полученные на основе обучающей выборки.

Параметр learning_rate — скорость обучения, является, который определяет размер шага на каждой итерации, двигаясь в направлении минимума функции потерь, влияет на то, в какой степени вновь полученная информация переопределяет старую информацию.

Максимальное количество деревьев, которые могут быть построены при решении указывается в параметре iterations. При использовании других параметров, ограничивающих число итераций, конечное число деревьев может быть меньше числа, указанного в этом параметре.

Случайное значение, которое используется для обучения задается в параметре random_seed.

Параметр od_type задает детектор переобучения, который будет использоваться. CatBoost может остановить обучение раньше, чем это диктуют параметры. Например, он может быть остановлен до того, как будет построено заданное количество деревьев. Количество итераций для продолжения обучения после итерации с оптимальным значением метрики задается в параметре od_wait. Его назначение различается в зависимости от указанного od_type.

При od_type = IncToDec детектор переполнения при достижении порогового значения игнорируется и обучение продолжается для заданного числа итераций после итерации с оптимальным значением метрики.

Если значения параметра od_type = Iter обучение прекращается после заданного количества итераций, начиная с итерации с оптимальным значением метрики.

Сформируем модель с таким набором параметров:

```
model1 = CatBoostRegressor (
    eval_metric='AUC',
    learning_rate=0.8,
    iterations=500,
    random_seed=42,
    od_type='Iter',
    od_wait=20
)
```

Далее обучаем классификатор используя функцию fit:

```
model1.fit(
    X_train, y_train,
    eval_set=(X_validation, y_validation),
    cat_features=cat_features,
    logging_level='Silent',
    plot=True
)
```

В переменной X_train содержатся данные, которые используются как входной обучающий набор данных. В y_train — целевые переменные (другими словами, значения меток объектов) для обучающего набора данных.

В параметр eval_set указывается набор данных проверки.

Параметр cat_features — вектор индексов категориальных признаков.

Уровень протоколирования для вывода в stdout задается в logging_level. Возможные значения:

'Silent' — не выводит никакой регистрационной информации.

'Verbose' — вывод оптимизации метрики, время обучения и оставшееся время обучения.

'Info' — вывод дополнительной информации и количества деревьев.

'Debug' — вывод отладочной информации.

Поскольку используется Jupyter NoteBook для написания кода, можно через параметр plot=True указать построить график со следующей информацией: значения метрики; пользовательские значения потерь; время прошедшее с начала обучения; оставшееся время до окончания обучения.

График обучения данной модели выглядит следующим образом (рис.4)

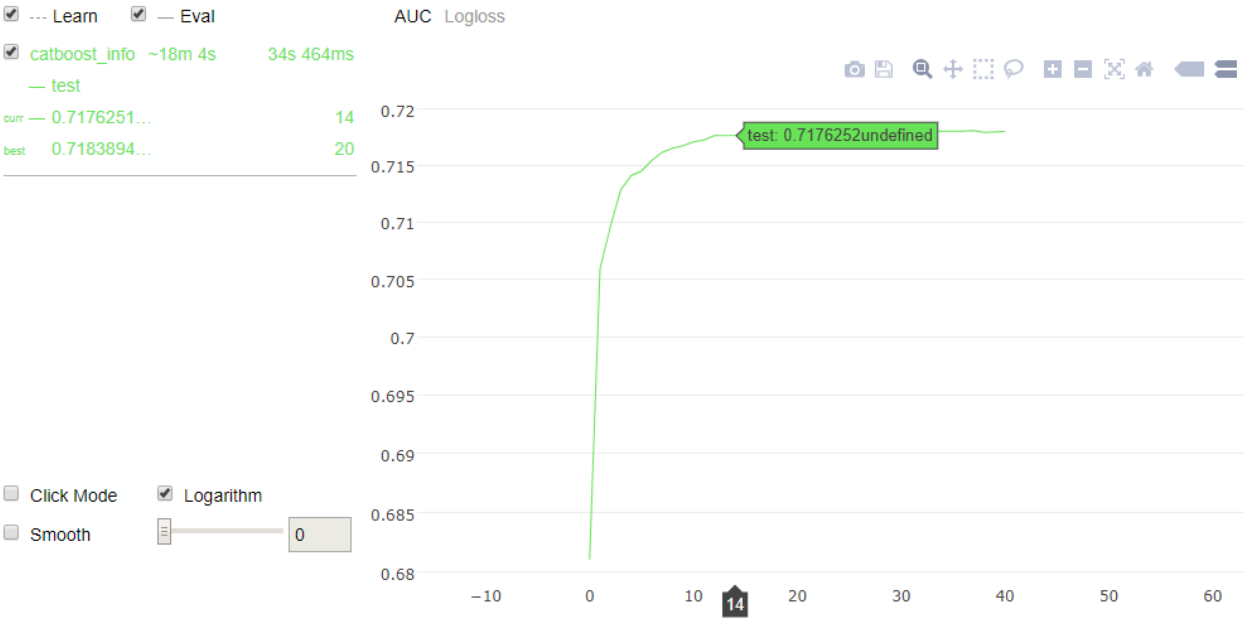

Рис. 4. График обучения модели

На графике построены метрики для всех обучений, оценок метрик и запусков перекрестной проверки, которые были использованы для обучения модели. По оси x можно видеть i-ую итерацию, по оси y значения тестовой выборки и предсказанное значение.

Рассмотрим модель с другими заданными параметрами.

```
model2 = CatBoostRegressor (
    thread_count =2,
    iterations=300,
    learning_rate=0.1,
    random_seed=63,
    custom_loss=['AUC', 'Accuracy'],
    use_best_model=True
)
```

Для реализации данного классификатора были заданы другие параметры:

Параметр thread_count — задание количества запущенных потоков.

Параметр custom_loss позволяет задать свою функцию потерь, в данном случае используется композиция функций 'AUC', 'Accuracy'.

При use_best_model=True количество деревьев, сохраненных в результирующей модели, определяется следующим образом: строятся деревья, их количество определено параметрами обучения. Используется набор данных проверки для идентификации итерации с оптимальным значением метрики, указанным в eval_metric. После этой итерации деревья не сохраняются.

Функция обучения fit не содержит параметров, влияющих на результат обучения, поэтому они остаются неизменны.

3.3 Анализ результатов

Для сравнения разных моделей были использованы различные метрики качества [16–21]:

- Время обучения
- Коэффициент детерминации (R^2) :

$$1 - \frac{\sum_{i=0}^{m} |a_i - y_i|^2}{\sum_{i=0}^{m} |\bar{y} - y_i|^2}, \text{где } \bar{y} = \frac{1}{m} \sum_{i=0}^{m} y_i$$

- MSLp1E(Mean squared logarithmic error):

$$\frac{1}{m}\sum_{i=0}^{m}(\log(a_i+1)-\log(y_i+1))^2$$

- MAE(Mean Absolute Error) :

$$\frac{1}{m}\sum_{i=0}^{m}|a_i-y_i|$$

- MSE(Mean Squared Error) :

$$\frac{1}{m}\sum_{i=0}^{m}|a_i-y_i|^2$$

- MedAE(Median absolute error):

$$\text{медианное значение от}\sum_{i=0}^{m}|a_i-y_i|$$

Сравнение этих метрик для обучения двух описанных моделей представлены на рисунке 5.

	model #1	model #2
working time:	0:01:51.894483	0:16:32.931793
mae:	0.7986315798317545	0.7976084168673939
mse:	0.6534059898067504	0.6520924417691101
r2_score:	-1.0374905646463848	-1.0308652171294055
MedAE:	805905.5162608621	804873.0343282222
MSLp1E:	0.33013467560547066	0.3294971063648194

Рис.5. Результат обучения моделей

По всем метрикам 2-я модель показала меньший, а следовательно лучший результат. При этом интересно заметить, что 2-я модель обучалась значительно дольше 1-ой.

Обучив, рассмотренные ранее, модели на данных с CTR столбцом имеется следующий результат (рис. 6).

	model #2.1	model #2.2
working time:	0:04:55.895495	0:24:05.322802
mae:	0.7884434343388699	0.790002501424827
mse:	0.6435668862072111	0.644164196413606
r2_score:	-0.9603515669399165	-0.9720008980229735
MedAE:	795624.5771388283	797197.8442078043
MSLp1E:	0.32479127642303396	0.32536290912109184

Рис. 6. Результат обучения моделей

Как можно заметить, введение столбца CTR улучшило результат обучения по заданным метрикам, но увеличило время обучения.

Для интерпретации полученных результатов предсказанные значения округляются и сравниваются с ожидаемыми значениями в целевой переменной. Таким образом, было установлено, что 1-ая и 2-ая модели, не использующие столбец CTR, правильно предсказывают результат в 78% и 78,9% случаев соответственно. Модели, к исходным данным которых, был добавлен столбец CTR, показали результат 81% и 81,7%.

Заключение

В рамках выпускной квалификационной работы решена задача, в рамках которой были построены рекомендательные системы с использованием метода градиентного бустинга. В качестве набора данных для обучения использовалась информация о действиях пользователей в персональной ленте рекомендаций сервиса «Яндекс.Дзен».

Сформированы две модели, для которых было проведено обучение на исходных наборах и расширенных данных. Расширение происходило за счет введения параметра CTR.

Выполнен сравнительный анализ качества предсказаний моделей на основе указанного набора характеристик.

Для модели, имеющей наилучший набор характеристик, полученное итоговое значение округлялось и сравнивалось со значением в целевой переменной.

В результате проведённого проекта по прогнозированию заинтересованности пользователей, мной было выявлено, что реализованная модель способна решать данную задачу с вероятностью 81,7%. Это позволяет уменьшать экономические потери от неправильного предсказания.

Проект был представлен на студенческой научной конференции «Неделя науки — 2020», посвящённой столетию со дня рождения И.И. Воровича, где был отмечен дипломом 3-ей степени в рамках секции «Обработка и генерация данных».

Литература

1. Build a Recommendation Engine With Collaborative Filtering. – URL: https://realpython.com/build-recommendation-engine-collaborative-filtering/ (дата обращения 20.05.2020)
2. Школа анализа данных Яндекса. – URL: https://yandex.ru/promo/academy/data_analysis (дата обращения 15.02.2019)
3. Exploration of Yandex CatBoost in Python. – URL: https://github.com/KwokHing/YandexCatBoost-Python-Demo/blob/master/README.md (дата обращения 10.10.2019)
4. Викиконспект ИТМО CatBoost. – URL: https://neerc.ifmo.ru/wiki/index.php?title=CatBoost (дата обращения 12.10.2019)
5. CatBoost от Яндекса. Разбираемся. – URL: https://xn--80ajickj6abfedo.xn--p1ai/2019/04/29/catboost (дата обращения 12.10.2019)
6. Дипломная работа:«Прогнозирование вероятности кликов на новые баннеры». – URL: http://www.machinelearning.ru/wiki/images/4/45/Kolesnikov-2012-master-thesis.pdf (дата обращения 12.11.2019)
7. CatBoost documentation. – URL: https://catboost.ai/docs/features/visualization_jupyter-notebook.html#visualization_jupyter-notebook (дата обращения 12.11.2019)
8. catboost/tutorials. – URL: https://github.com/catboost/tutorials/blob/master/ru/kaggle_amazon_tutorial_ru.ipynb(дата обращения 12.11.2019)
9. Применение моделей CatBoost внутри ClickHouse. Лекция Яндекса. – URL: https://habr.com/ru/company/yandex/blog/347696/ (дата обращения 25.12.2019)

10. Catboost для самых маленьких. – URL: https://habr.com/ru/sandbox/122369/ (дата обращения 25.12.2019)
11. Базовые принципы машинного обучения на примере линейной регрессии. – URL: https://habr.com/ru/company/ods/blog/322076/ (дата обращения 15.02.2020)
12. Методы оценки параметров моделей. – URL: https://forecasting.svetunkov.ru/forecasting_toolbox/estimation-simple-methods/ (дата обращения 13.04.2019)
13. Оценка качества прогнозных моделей. – URL: https://forecasting.svetunkov.ru/forecasting_toolbox/models_quality/ (дата обращения 13.04.2019)
14. Работа с категориальными признаками. – URL: https://habr.com/ru/company/ods/blog/326418/#label-encoding (дата обращения 15.04.2019)
15. CTR. – URL: https://www.calltouch.ru/glossary/ctr-click-through-rate-pokazatel-klikabelnosti/ (дата обращения 12.04.2019)
16. How to Make Predictions with scikit-learn. – URL: https://machinelearningmastery.com/make-predictions-scikit-learn/ (дата обращения 13.04.2019)
17. sklearn.metrics.mean_squared_error. – URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html (дата обращения 25.12.2019)
18. Методы оценки качества прогноза. – URL: https://habr.com/ru/post/19657/ (дата обращения 25.12.2019)
19. Простые методы оценки параметров моделей. – URL: https://forecasting.svetunkov.ru/etextbook/forecasting_toolbox/estimation-simple-methods/ (дата обращения 26.12.2019)
20. pandas.to_numeric. – URL: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_numeric.html#pandas-to-numeric (дата обращения 26.12.2019)

21. Formula For R2_Score. – URL: https://www.kaggle.com/c/mercedes-benz-greener-manufacturing/discussion/34567 (дата обращения 20.04.2020)