КОНСТРУККТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

* 1. Описание поисковых запросов, на которых будет подсчитана метрика

Задача Метрики скорости реакции — измерить качество поисковой

выдачи на запросах, который задавались сразу после наступления новостного события. Таким образом мы можем сформулировать следующие критерии для подходящего запроса (ниже будут даны соответствующие пояснения критериям):

1. С запросом должно быть связано какое-либо новостное событие. Новостное событие должно удовлетворять следующим критерием:
   1. Событие по запросу не является запланированным.
   2. Для объекта запроса произошедшее событие значимо.
2. За день до запроса не должно происходить значимых событий, подходящих под формулировку запроса.
3. Запрос должен быть взят из потока, т.е. являться настоящим существующим запросом, который пользователь задал в какое-либо время в Яндекс Поиск.

Запрос, отвечающий критериям выше будем называть «***Ультрасвежим***». Примеры незапланированных и значимых событий:

* ЧП
* Катаклизм
* Неожиданные заявления
* Происшествия
* Неожиданные смерти

Примеры запланированных событий (запросы, связанные с данными событиями не являются ультрасвежими):

* Телепередача/эфир/стрим
* Спортивный матч
* Премьера фильма / выход новой серии
* Конференции / заседания

Примеры незначимых событий (запросы, связанные с данными событиями не являются ультрасвежими):

* Частые непланируемые события
* Незаметные новости с фронта
* Регулярные заявления

Такие запросы на начальном этапе будут определяться асессорами — сотрудниками, чья работа заключается в выполнении различных заданий для разметки данных. В дальнейшим планировалось обучить модель, которая по временному ряду задавания запроса смогла бы классифицировать запрос на ультрасвежесть.

Для любых самых громких новостных событий характерно следующее поведение: сразу после наступления такого события, количество поисковых запросов, связанных с этим событием, резко возрастает. С какого-то момента ажиотаж вокруг события начинает затухать и динамика запросов по событию падает, ведь люди все меньше и меньше интересуются таким событием.

Проиллюстрируем динамику таких запросов:

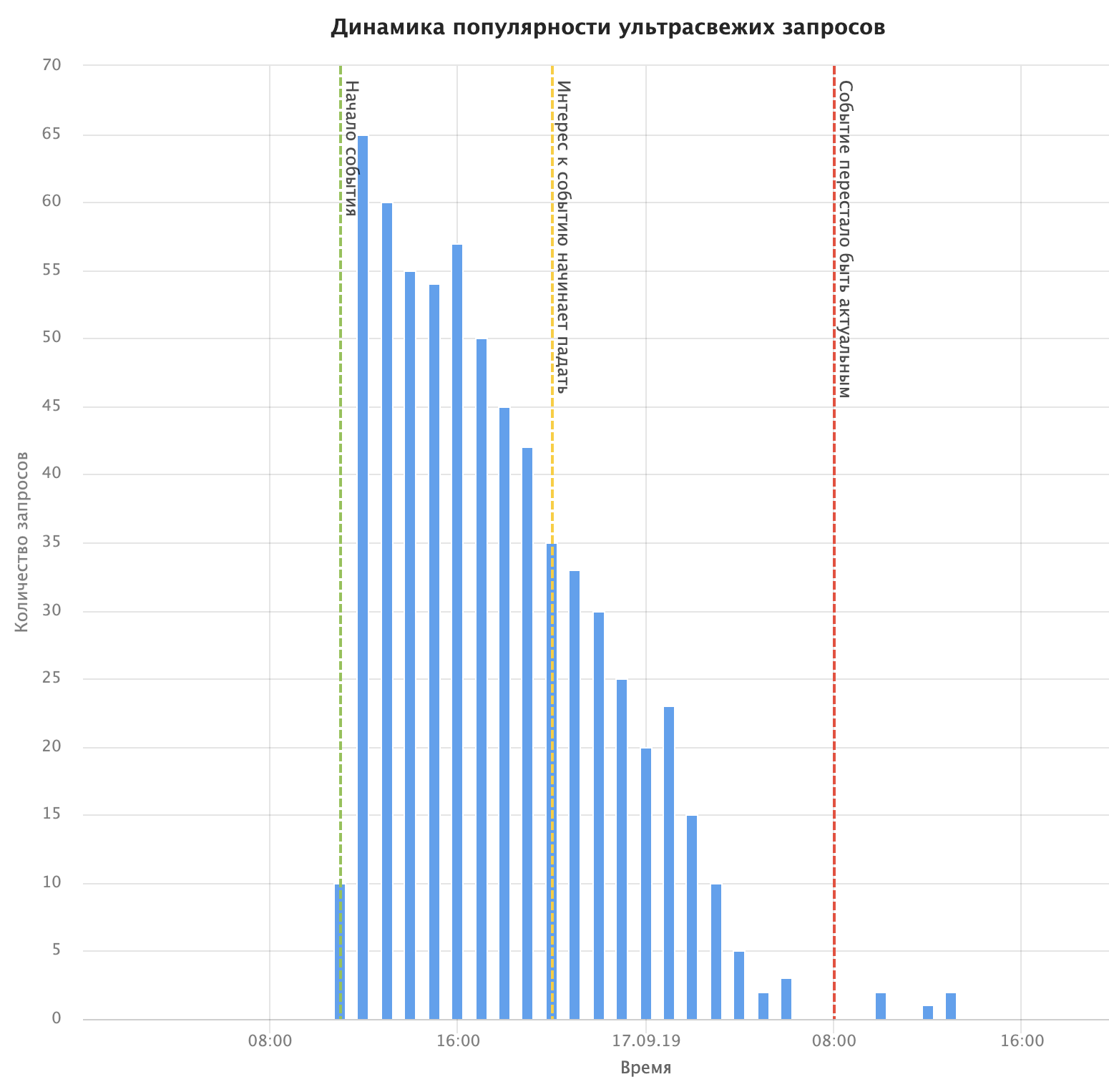


Рисунок 14 — Динамика популярности ультрасвежих запросов

По графику (Рисунок 14) видна отличительная особенность всех ультрасвежих запросов — всплеск популярности и плавный количества поисковых запросов по событию. Чем строже будет соблюдаться такой вид динамики запроса, тем легче в дальнейшем их будет классифицировать модели. Поэтому было решено не рассматривать запланированные события, ведь, поскольку о событии будет известно заранее, люди будут им интересоваться и до его наступления и выделить их среди потока с необходимой точностью не получится. Дополнительным условием для запроса является значимость события. Если событие незначимо, то оно не вызовет резонанс общественности, характерный всплеск не будет наблюдаться и моделе будет сложнее такой запрос верно классифицировать. Чтобы наблюдался резкий всплеск запроса также необходимо, чтобы за день до по данному запросу не было релевантных событий.

По нашим предположениям, если учтены нюансы, о которых сказано выше, то модель сможет с достаточной точностью классифицировать запросы на ультрасвежесть и тогда затраты на необходимый объем разметки асессорами будут укладываться в бюджет компании на соответствующее полугодие.

* 1. Запуск разметки на ультрасвежесть

После анализа природы необходимых запросов и формализованного описания термина ультрасвежести необходимо набрать необходимое количество асессоров и составить проект разметки в системе Yang.

Yang — краудсорсинговый проект для разметки большого количества данных, которые затем, как правило, используются для машинного обучения.

Асессоры должны были ознакомиться с инструкцией, пройти составленные мной обучающие и экзаменационные задания. Асессоры успешно справившееся с экзаменом допускаются до разметки реальных данных (Рисунок 15).

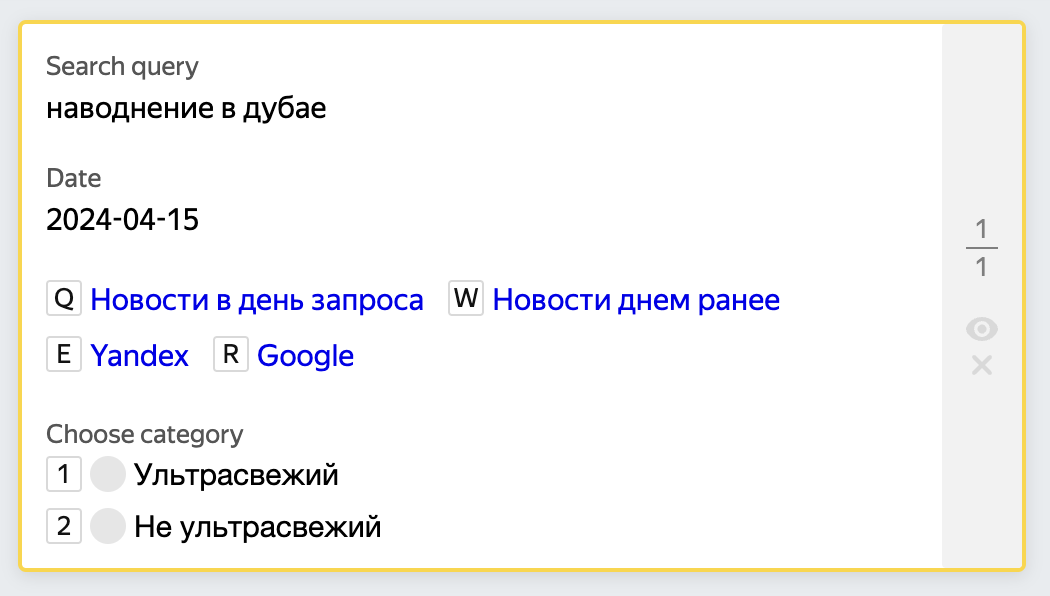


Рисунок 15 — Интерфейс асессорской разметки

* 1. Ансамбль для классификации ультрасвежих запросов

После получения необходимого количества разметки необходимо понять,

как на этом датасете обучить модель для классификации ультрасвежих запросов.

В нашем распоряжении были логи самого поиска и вектор признаков запроса, который автоматически считается Яндекс Поиском. Для классификации необходимо было обработать динамику запроса в соответствующий день (пример Рисунок 14), а также использовать уже имеющийся посчитанный вектор признаков, который описывает смысл запроса.

Было решено использовать LSTM для обработки временных рядов динамики запроса, т.к. эта модель хорошо себя показывает на работе с последовательностями, и ее архитектура не такая сложная, как, например, BERT, чтобы быстро переобучиться на нашем небольшом датасете (асессорский ресурс — небесплатный).

Для финальной классификации поверх LSTM было решено использовать CatBoost т.к. эта модель, как правило, показывает самые высокие результаты на табличных данных. В нашем случае данными для классификации будут признаковое описание запроса и получившееся предсказания LSTM.

Итоговый пайплайн можно проиллюстрировать следующим образом (Рисунок 16).

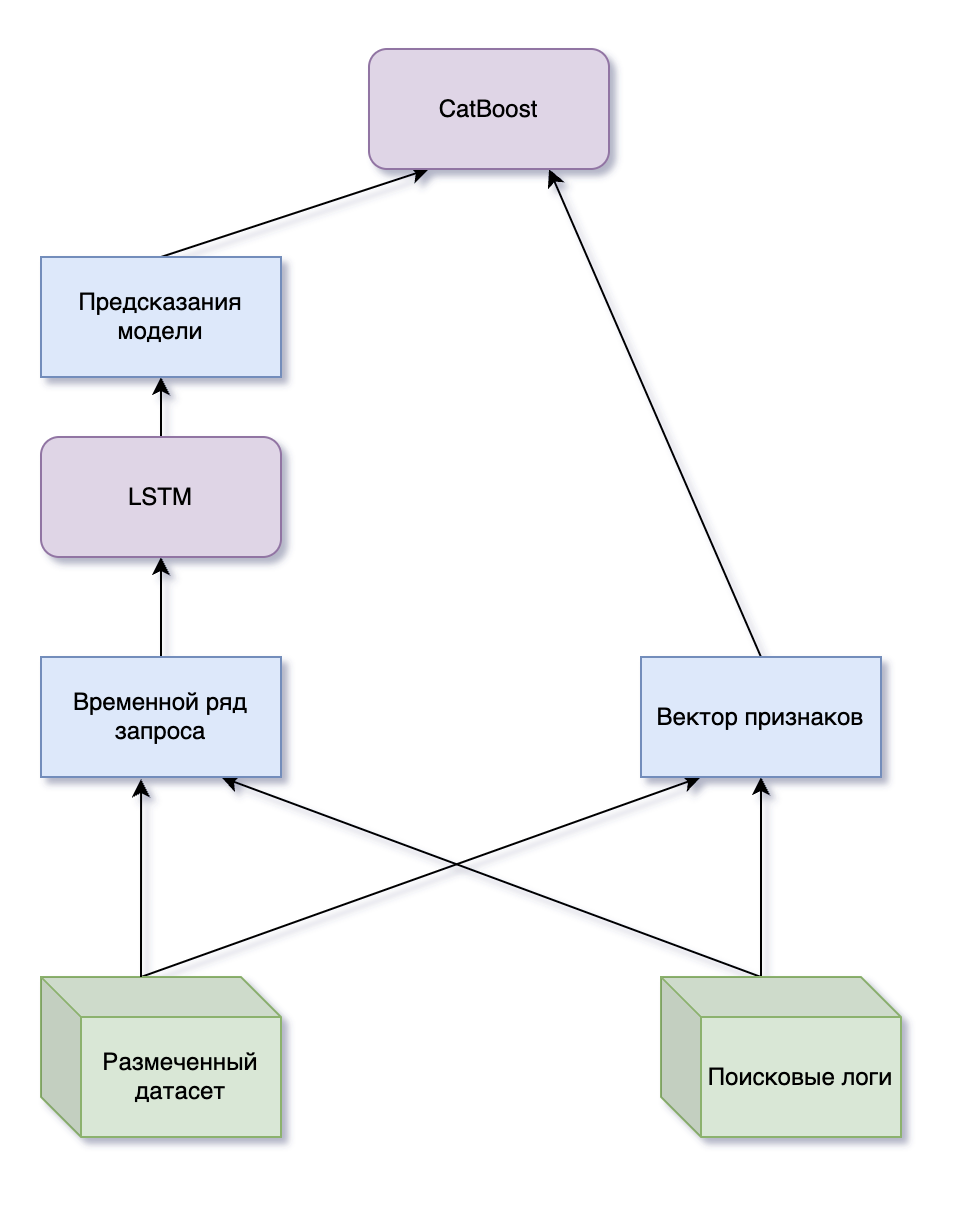


Рисунок 16 — Пайплайн классификации на ультрасвежесть

* 1. Обучение LSTM

По временному ряду необходимо предсказать вероятность отнесения данного запроса к ультрасвежим [5-11].

Пример временного ряда ультрасвежего запроса:

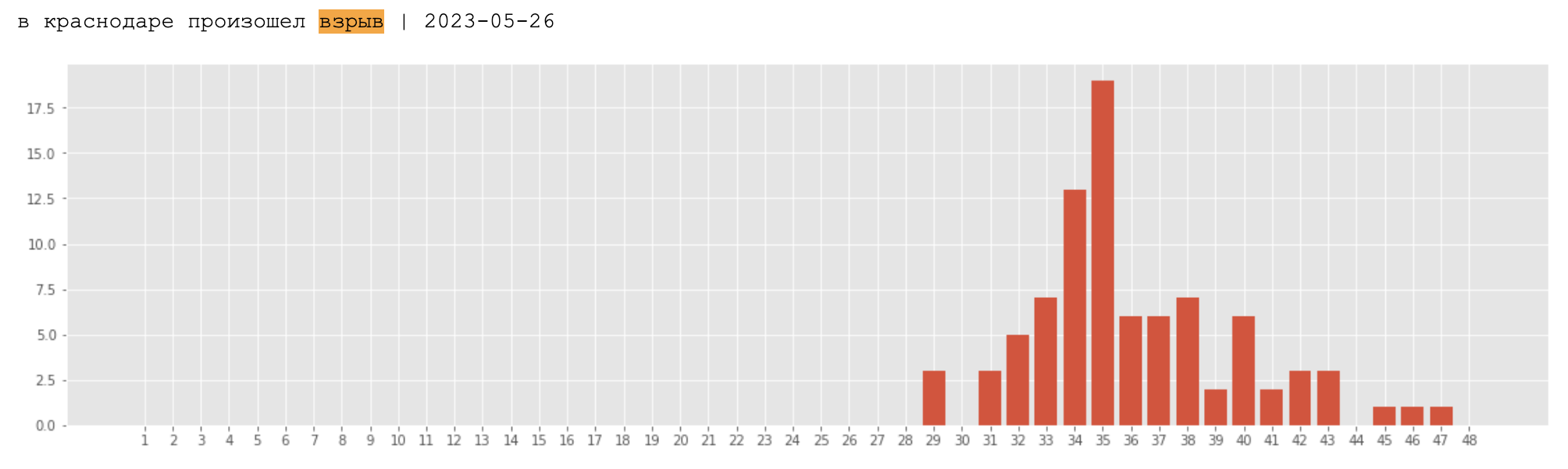


Рисунок 17 — временной ряд ультрасвежего запроса

Пример временного ряда новостного, но не ультрасвежего запроса про заранее спланированное спортивное событие (видим отсутствие всплеска):

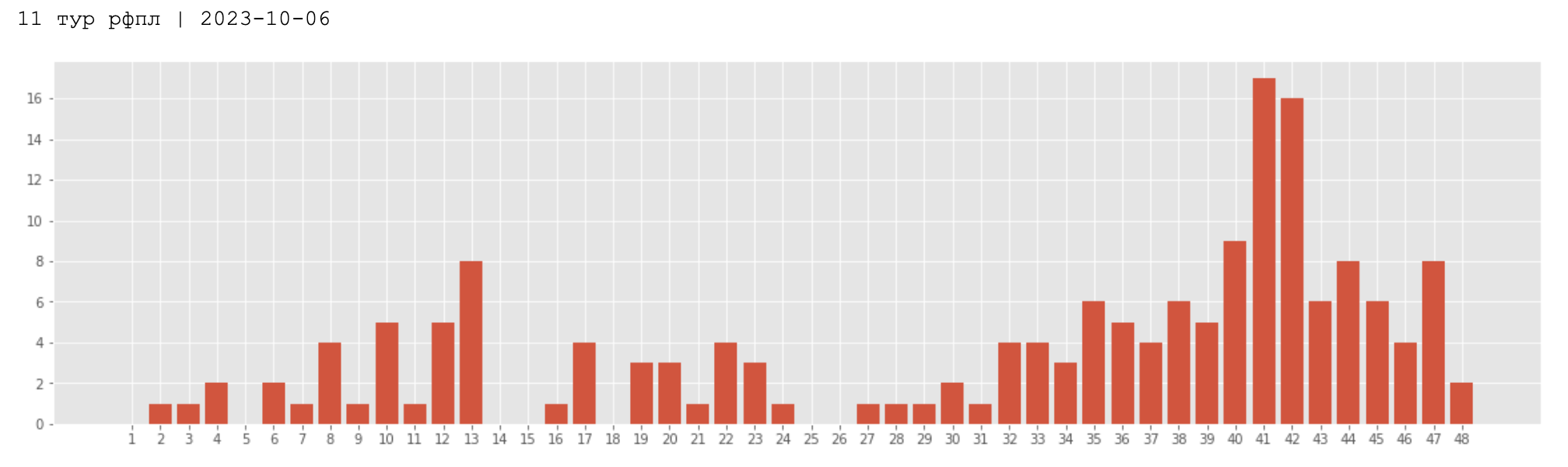


Рисунок 18 — временной ряд запроса, связанного с заранее спланированным событием

Такой запрос не будет являться ультрасвежем по нашему определению, так как о событии заранее известно.

Нормализовав такой временной ряд можно попробовать обучить первую модель нашего ансамбля для классификации запроса — LSTM. Ниже представлен листинг кода основного класса модели.

class LSTMClassifier(nn.Module):

"""Implementation of LSTM-based time-series classifier."""

def \_\_init\_\_(self, input\_dim, hidden\_dim, layer\_dim, output\_dim):

super().\_\_init\_\_()

self.hidden\_dim = hidden\_dim

self.layer\_dim = layer\_dim

self.rnn = nn.LSTM(input\_dim, hidden\_dim, layer\_dim, batch\_first=True)

self.fc = nn.Linear(hidden\_dim, output\_dim)

self.batch\_size = None

self.hidden = None

def forward(self, x):

h0, c0 = self.init\_hidden(x)

out, (hn, cn) = self.rnn(x, (h0, c0))

out = self.fc(out[:, -1, :])

return out

def init\_hidden(self, x):

h0 = torch.zeros(self.layer\_dim, x.size(0), self.hidden\_dim)

c0 = torch.zeros(self.layer\_dim, x.size(0), self.hidden\_dim)

return [t.cuda() for t in (h0, c0)]

В качестве оптимизитора градиентного спуска использовался Adam т.к. он обеспечивает самую быструю сходимость и умеет преодолевать точки локального минимума функции потерь. Поверх выхода нейронной сети применялась функции активации сигмойды т.к. решаем задачу бинарной классификации.

Основные параметры модели:

* Размер скрытого полносвязного слоя идущего сразу после LSTM — 256 (hidden\_dim в коде).
* Количество эпох — 100
* Learning rate — 0.0005

При обучении использовался шедулер, засчет которого можно было циклически изменять learning\_rate, чтобы не застревать в точках локального минимума и с наилучшем приближением придти к глобальному экстремуму. Ниже представлен график коэффицента, на который с течением времени будет убножаться learning rate:

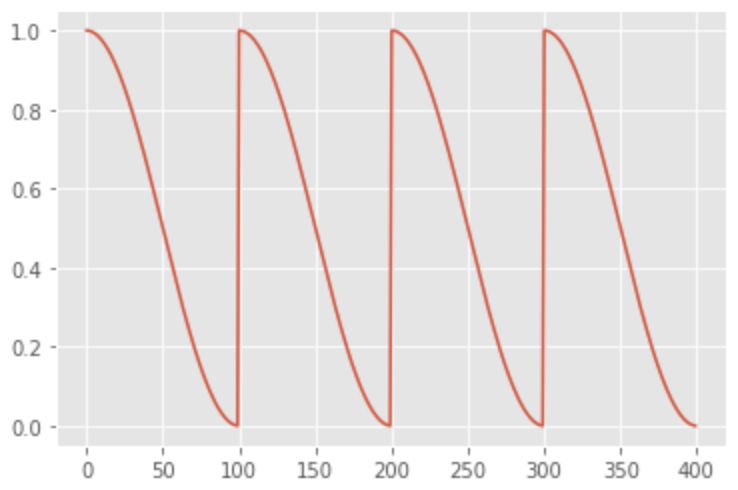


Рисунок 19 — коэффициент умножения скорости обучения

Итоговые метрики при пороге 0.403 при бейзлайне (предсказание ультрасвежести на всех запросах) precision = 0.05, recall = 1:

|  |  |  |
| --- | --- | --- |
| Метрика | Precision | Recall |
| Значение | 0.14 | 0.96 |

Стоит отметить, что бизнес требованием к данной задаче являлась высокая полнота классификации запросов [4]. Идея в том, чтобы как можно более честно брать ультрасвежие запросы из потока, без смещений на модели. В случае низкого recall часть запросов бы терялось, потому что их не различила бы модель, а возможно именно на этих запросах у поисковой системы низкое качество.

Учитывая, что раньше лишь 0.05 запросов по разметке было ультрасвежими делаем вывод, что засчет данной модели необходимо в 3 раза меньше асессорской разметки, чтобы набрать такое же количество ультрасвежих запросов за определенный день. Поэтому обучение модели можно считать успешным.

* 1. Обучение градиентного бустинга

В качестве фреймворка градиентного бустинга был выбран CatBoost, на данный момент это лучший технологический фреймворк, который позволяет гибко обучить модель и настроить визуализацию обучения, необходимую тебе.

Датасет на котором обучалась модель содержал разные нейросетевые признаки запроса такие как: вероятность отнесения запроса, к новостным или спортивным, является ли запрос медицинским и т.д. Всего более 300 колонок с разными текстовыми признаками. Временной ряд каждого запроса был обработан LSTM, ее выход был добавлен в качестве дополнительного признака.

За отложенное нам время нам удалось разметить почти 20 тысяч разных запросов. В датасете была всего 1 тысяча ультрасвежих запросов, поэтому необходима была стратификация по предсказываемому параметру перед делением данных на тренировочную и тестовую части. Несмотря на стратификацию обучение катбуста и его финальные метрики очень сильно зависили от рандома разбиения датасета на тестовую и тренировочную части. Становилось ясно, что необходимо кросс-валидация — метод оценки производительности модели, состоящий в том, чтобы разбить обучающую выборку на несколько фолдов (частей), проверить качество модели на каждом фолде, используя остальные фолды для обучения, и усреднить результаты. Этот метод позволяет получить более точную оценку производительности модели, т.к. использует все доступные данные для обучения и валидации.

Т.к. выборка несбалансирована, метрика точности классификации не подходит т.к. ее значение будет не информативным, ведь точность бейзлайна (предсказание константы 0) достигало бы уже 95% т.к. в исходном датасете всего 5% ультрасвежих запросов.

Продуктово нам было необходимо, чтобы модель как можно меньше пропускала ультрасвежих запросов для более точного и объективного подсчета метрики, поэтому было принято решение максимизировать precision при recall >= 0.95.

Ниже представлен график обучение градиентного бустинга на каждом из фолдов кросс-валидации (всего их было 50).

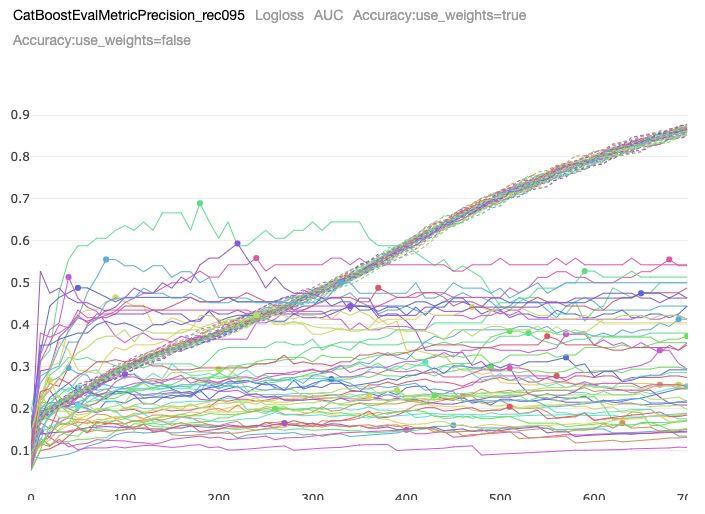


Рисунок 20 — график precision модели при recall >= 0.95 на каждом фолде кросс-валидации

Такие графики генерирует фреймворк CatBoost при указании необходмой конфигурации обучения. По оси Ox показывается количество обученных деревьев по оси Oy значение выделенной жирным метрики (на Рисунке 20 выбрана самописная метрика — значение precision при кecoll не меньше 0.95). Сплошной линией показывается значение метрик на тестовом датасете, пунктирной — на тренировочном. Точкой отмечено количество деревьев в градиентном бустинге при котором значение метрик максимальной на данном фолде. Как только значение метрик на тестовом множестве перестает падать градиентный бустинг переобучается и обучение следует остановить, но на каждом фолде необходимо обучить разное количество деревьев, прежде чем модель начнет переобучаться, поэтому было зафиксировано, что на каждом фолде мы последовательно обучаем 700 деревьев.

Ниже представлен график метрики LogLoss, где каждый цвет также определяет определенный фолд кросс-валидации, обозначения, описанные выше для предыдущего графика также верны:

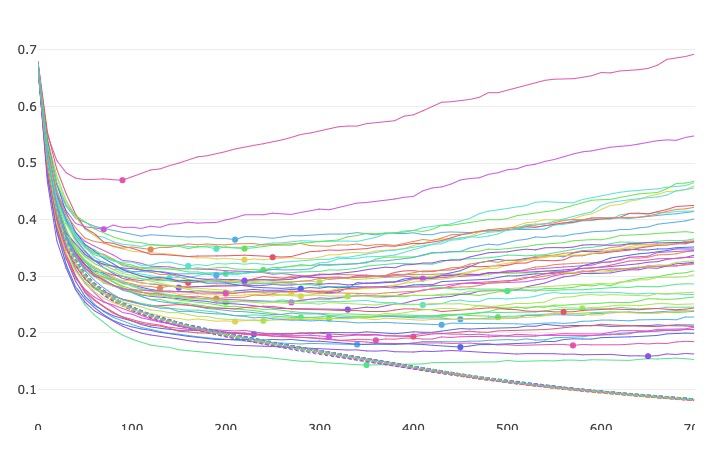


Рисунок 21 — значение LogLoss на каждом фолде кросс-валидации

Чтобы выбрать итоговое количество деревьев необходимо выбрать количество итераций обучения деревьев, при котором среднее значение целевой метрики precision (Рисунок 20) больше усредненного значения метрики на любом другом количестве итераций обучения деревьев.

Итоговые метрики градиентного бустинга:

|  |  |  |
| --- | --- | --- |
| Метрика | Precision | Recall |
| Значение | 0.3 | 0.95 |

Взятое количество деревьев — 410, глубина каждого дерева — 5. Порог при котором посчитаны метрики — 0.29 (градиентный бустинг предсказывает вероятность класса ультрасвежести для запроса).

Сравнивая разные подходы предсказания были получены следующие метрики (recall в каждом случае был не меньше 0.95):

1. LSTM, обученная на временном ряде запроса показывала precision = 0.15.
2. CatBoost, обученный на нейросетевых признаках, используемых в продакшене Яндекс Поиска показал precision = 0.25.
3. Если добавить предсказания LSTM, как дополнительный признак и обучить CatBoost, то precision = 0.3, что является наилучшим показателем.
   1. Поиск начала события

После того, как мы научились определять запросы, на которых хотим замерять качество необходимо реализовать итоговый процесс подсчета метрики. Для каждого ультрасвежего запроса необходимо определить момент начала этого события, точнее, момент, когда по данному событию был создан первый релевантный документ. Начиная с этого момента, мы будем считать, что миру стало известно о данном событии, а значит, если это событие громкое, то в скором времени ожидается пик запросов по данному событию, на таких запросах мы хотим померить качество.

Чтобы определить, когда был создан первый релевантный документ к этому запросу необходимо взять из базы индексированных страниц поиска страницы, созданные в день запроса, а далее в автоматическом режиме определить, какие из этих страниц релевантны запросу. Эту задачу мы реализовали с помощью архитектуры BERT нескольких типов: запросный BERT, и документный BERT. На выходе каждой модели мы получали вектор признаков, описывающий запрос и страницу соответственно. Значение скалярного произведения будет являться мерой схожести запроса и документа (Рисунок 22). Подобрав порог скалярного произведения, можно автоматически отбирать страницы, релевантные запросу.



Рисунок 22 — архитектура процесса, определяющего схожесть между запросом и страницей.

Страницы, релевантные запросу отправляются на запросно-документную асессорскую разметку (Рисунок 22), где асессоры отвечают на вопрос «Содержит ли данная страница актуальную новость по указанному запросу?».

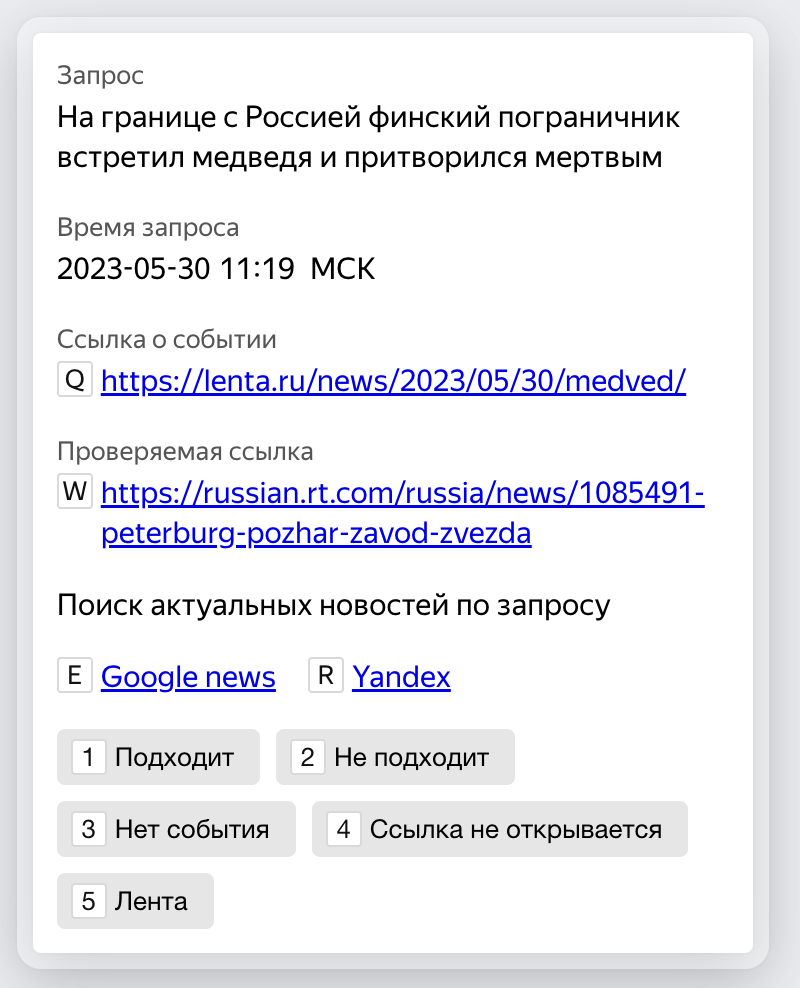


Рисунок 23 — асессорская разметка на соответствие страницы ультрасвежему запросу

Таким образом, разметив асессорами первые 100 созданных релевантных по BERT-у страниц мы сможем найти страницу, которая первая описала произошедшее событие. Время создания такой страницы и будет временем начала события.

* 1. Итоговый подсчет метрики

Итак, у нас есть ультрасвежий запрос, с ним связанно какое-то внезапное громкое новостное событие, мы знаем, в какой момент времени появилась первая новость, описывающая это событие. После момента начала данного события в первые два часа обычно происходит пик запросов пользователей, интересующихся данной новостью. Нам необходимо каким-то образом померить качество выдачи этих запросов в первые два часа после происшествия события.

В Яндексе есть множество внутренних замеров и метрик, измеряющих релевантность выдачи. Мы знаем, что показатель релевантности выдачи Яндекс Поиска очень высокий и на запрос «милые котики» мы не будем показывать собачек или не милых котиков. Поэтому можно априори считать, что первые топ-5 страниц по запросу имеют высокую релевантность, если запрос не являлся бредом или случайным набором букв. Поэтому, зная время начала события по запросу и зная время создания показанных страниц можно оценить их актуальность: страница будет актуальна, если она будет создана после момента начала события. Например: по запросу «землетрясение в Турции», заданному 6 февраля 2023 году можно показать страницу из википедии с землетрясением 2014 года, эта страница будет релевантной, но не актуальной. Пользователь хочет увидеть информацию про актуальное ЧП, ведь запрос был задан именно в день трагедии.

Итак, зная время происшествия события по запросу, время создания страниц на выдачи, мы можем определить актуальность новостной выдачи, как долю страниц в топ-5, созданных после происшествия события (Рисунок 24).



Рисунок 24 — подсчет метрики на одном запросе

Итоговые процесс вычислений после получения множества ультрасвежих запросов можно визуализировать следующим образом:



Рисунок 25 — процесс вычисления метрики после получения множества ультрасвежих запросов

Таким, образом, если доля актуальной выдачи в Яндекс Поиске в первые два часа после огласке в СМИ события — низкая, значит Яндекс Поиск не удовлетворяет потребности пользователя и обладает недостаточной скоростью реакции на новостные внезапные события. Если на большинстве запросов Яндекс Поиск показал актуальные страниц, созданные после момента происшествия событий — значит, метрика скорости реакции будет показывать высокие значения.

ЗАКЛЮЧЕНИЕ

В результате выполнения Выпускной квалификационной работы бакалавриата был разработан процесс, вычисляющий метрику скорости реакции поисковой системы Яндекс Поиск. Метрика отражает качество выдачи на запросах, заданных в первое время после момента происшествия новостного события. В ходе работы была также исследована динамика таких запросов, обучены модели машинного и глубокого обучения для того чтобы классифицировать такие запросы из общего потока запросов. Создано несколько проектов по разметке данных, разработан пайплайн предсказания таких запросов с помощью рекуррентной нейронной сети LSTM, обученной по временному ряду задавания запроса и модели машинного обучения — градиентного бустинга, обученного на выходе LSTM и нейросетевых поисковых признаках.

Благодаря разработанному мной модулю метрики скорости реакции поисковой системы Яндекс Поиск разработчики понимают текущее качество выдачи на таких запросах, а также появляется возможность более детально анализировать причины низких значений в отдельные дни на некоторых поисковых запросах. По итогу разработки данной метрики появилась возможность проводить AB эксперименты, которые способны увеличить метрику скорости реакции Яндекс Поиска на новостные события. В результате чего создается огромный потенциал к улучшению алгоритмов ранжирования Яндекс Поиска, а также повышению качества пользовательского опыта использования поисковой системы.

Благодаря разработанному мной модулю метрики скорости реакции поисковой системы Яндекс Поиск разработчики понимают текущее качество выдачи на таких запросах, а также появляется возможность более детально анализировать причины низких значений в отдельные дни на некоторых поисковых запросах. По итогу разработки данной метрики появилась возможность проводить AB эксперименты, которые способны увеличить метрику скорости реакции Яндекс Поиска на новостные события. В результате чего создается огромный потенциал к улучшению алгоритмов ранжирования Яндекс Поиска, а также повышению качества пользовательского опыта использования поисковой системы.