

Аннотация

Целью данной работы является изучение возможности определять неработающие видео по статистике просмотров. В ходе работы был предложен ряд моделей и проведены численные эксперименты. Наилучшей точности классификации удалось достичнуть с использованием классификатора из библиотеки CatBoost¹. Наилучший классификатор показал точность в 80.3 % и полноту в 51.4 %. Точность классификации удалось улучшить на 9 % с помощью добавления в данные событий плеера и создания отдельной модели для каждого плеера. В данном случае удалось достичнуть точности классификации в 89.3 % при той же полноте. На основании данной работы рекомендуется внедрение классификатора в сервис Yandex Video, а также дальнейшее изучение свойств модели и ее последующее улучшение.

¹Liudmila Prokhorenkova и др. “CatBoost: unbiased boosting with categorical features”. В: (28 июня 2017). arXiv: <http://arxiv.org/abs/1706.09516v5> [cs.LG].

Содержание

1. Аннотация	2
2. Используемые определения	4
2.1. Бинарная классификация	4
2.2. Видеоплеер	4
2.3. Yandex Video	5
2.4. Yandex Toloka	6
2.5. Неработающие видео	6
3. Введение	9
4. Предыдущие работы	10
5. Подготовка данных	12
5.1. Исходные данные	12
5.2. Выбор признаков	12
6. Обучение классификатора	14
6.1. Выбор модели	14
6.2. Обучение модели	15
6.3. Одноплеерная модель	20
7. Заключение	22
Список литературы	23

Используемые определения

2.1 Бинарная классификация

В данной работе рассматривается задача бинарной классификации, то есть разбиение множества объектов на два класса. Были выбраны следующие обозначения: под классом 0 (нулевым классом, отрицательным классом) подразумевается класс работающих видео, а под классом 1 (положительным классом) подразумевается класс неработающих видео, детектирование которых и является целью данной работы.

При анализе предсказания модели в задаче бинарной классификации принято говорить о четырех группах объектов: объекты, верно отнесенные к отрицательному классу (true negatives); объекты, верно отнесенные к положительному классу (true positives); объекты, неверно отнесенные к отрицательному классу (false negatives) и объекты, неверно отнесенные к положительному классу (false positives).

Для оценки модели в задаче бинарной классификации принято использовать метрики точности (precision) и полноты (recall), которые вводятся следующим образом:

$$P = \frac{TP}{TP+FP}, R = \frac{TP}{TP+FN},$$

где с помощью P обозначена точность (precision), R — полнота (recall), TP , FP и FN — числа объектов, относящихся к той или иной группе из четырех указанных выше. Так, TP — это число объектов, верно отнесенных к положительному классу (true positives), FP — это число объектов, неверно отнесенных кциальному классу (false positives), а FN — это число объектов, неверно отнесенных к отрицательному классу (false negatives). Здесь и далее под точностью и полнотой классификации подразумеваются именно введенные выше метрики.

2.2 Videopлеер

Под видеоплеером (плеером) в данной работе подразумевается элемент web страницы, позволяющий пользователю смотреть видеозаписи (рисунок 1). Один видеоплеер, как правило, умеет воспроизводить ряд видеозаписей, хранящихся на сервере провайдера контента. Различные видеохостинги могут использовать один общий плеер, однако внутри одного хостинга видео, как правило, воспроизводятся с помощью одного и того же плеера.

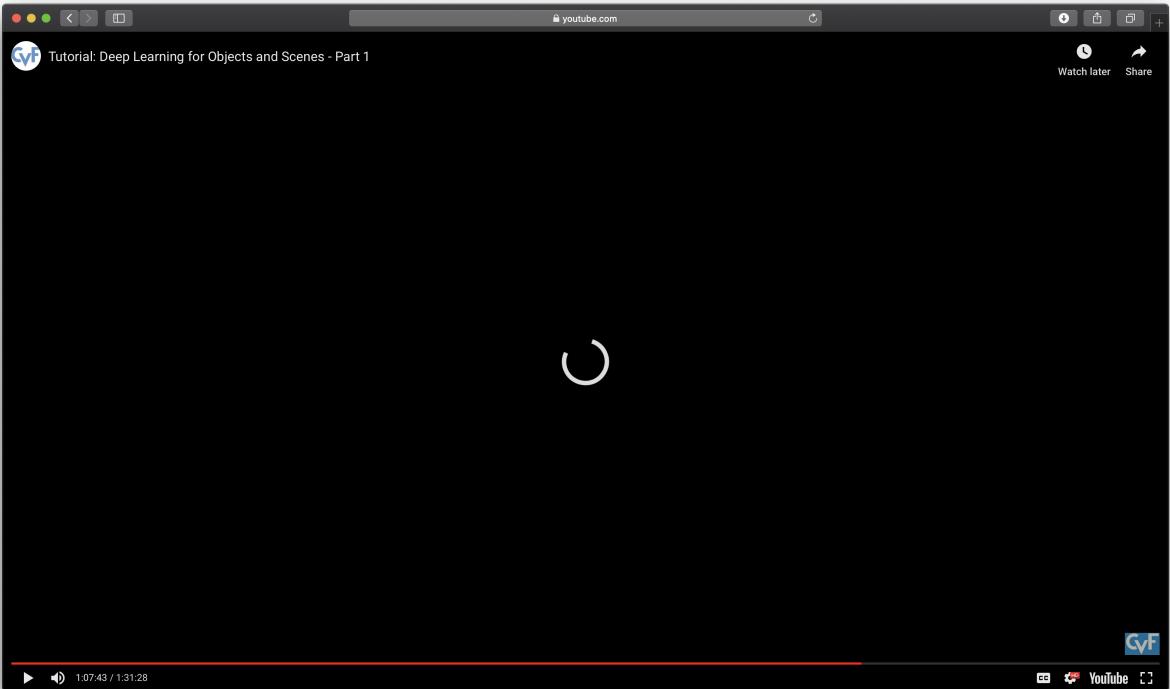


Рисунок 1 — Видеоплеер YouTube.

Помимо предоставления конечным пользователям возможности просматривать видеозаписи, видеохостинги часто предоставляют возможность другим интернет ресурсам интегрировать этот контент в свои страницы. Для этого, как правило, используется тэг iframe языка разметки HTML. Данный тэг, согласно спецификации¹, позволяет web странице открывать внутри себя вложенные страницы, например, с видеоплеером.

Кроме того ряд крупных видеохостингов предоставляют более обширное API для интеграции контента. В частности, некоторые плееры открывают доступ к так называемым событиям плеера, то есть оповещениям о некоторых событиях, которые дают возможность оценивать состояние контента. К таким событиям чаще всего относятся события старта, паузы и ошибки плеера. Примером плеера, который предоставляет подобную функциональность может служить плеер видеохостинга YouTube².

2.3 Yandex Video

Yandex Video³ (яндекс видео, видеопоиск) — это сервис, позволяющий пользователям смотреть видеозаписи, собранные с множества различных видеохостингов (рисунок 2). Из-за большого объема контента и невозможности моментально реагировать на его изменения, хранить видео на локальных серверах не представляется возможным, а по-

¹Scott O'Hara и др. HTML 5.3. W3C Working Draft. <https://www.w3.org/TR/2018/WD-html53-20181018/>. W3C, окт. 2018.

²YouTube Player API Reference for iframe Embeds. Май 2018. URL: https://developers.google.com/youtube/iframe_api_reference.

³Yandex Video. 2019. URL: <http://yandex.ru/video/>.

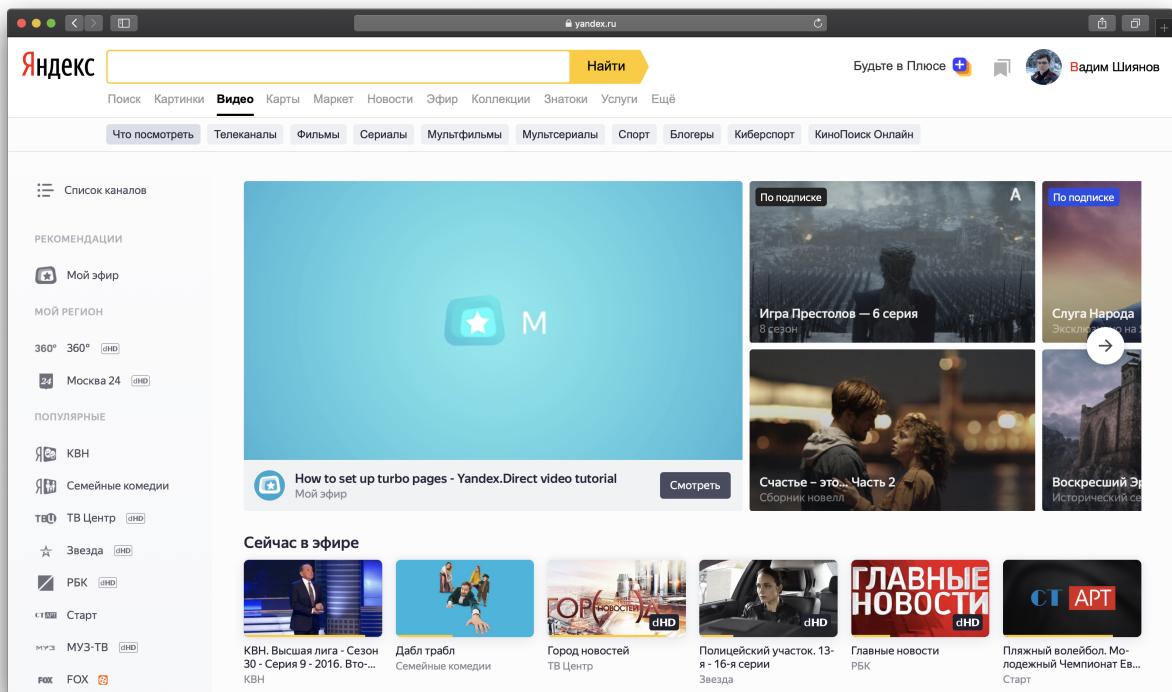


Рисунок 2 — Сервис Yandex Video.

тому видеозаписи показываются пользователям в виде интегрированных с помощью тэга `iframe` видеоплееров.

2.4 Yandex Toloka

Yandex Toloka⁴ (яндекс толока, толока) — это сервис, позволяющий публиковать некоторые несложные задания, которые другие пользователи могут выполнять за материальное вознаграждение (рисунок 3). Данный сервис использовался в данной работе для получения экспертной разметки данных: пользователям, взявшимся за выполнение задания, показывался `iframe` с видеозаписью и требовалось ответить, проигрывается эта видеозапись или нет.

2.5 Неработающие видео

Под неработающим видео (рисунок 4) в данной работе поднимается любая видеозапись, которую пользователь не может просмотреть. Видеозапись может прекратить работу по ряду причин: отключение сервера видеохостинга, на котором данная видеозапись хранилась или блокировка просмотра видеозаписи на основании обращения правообладателя контента.

⁴Yandex Toloka. 2019. URL: <https://toloka.yandex.com>.

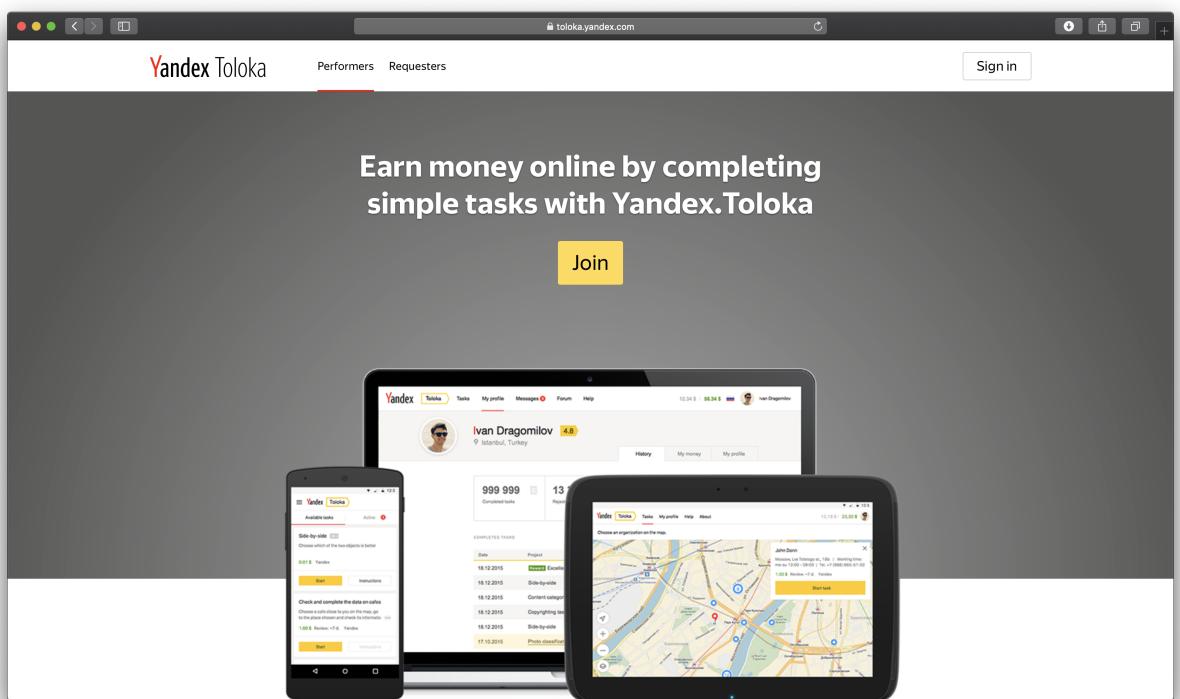


Рисунок 3 — Сервис Yandex Toloka.

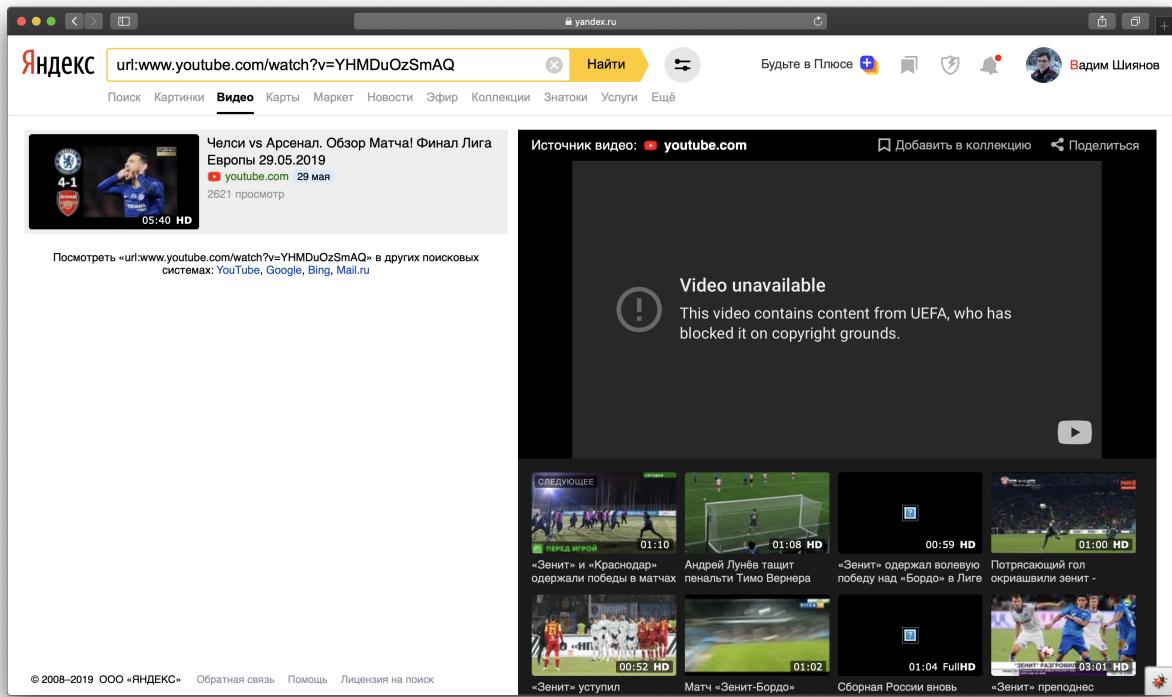
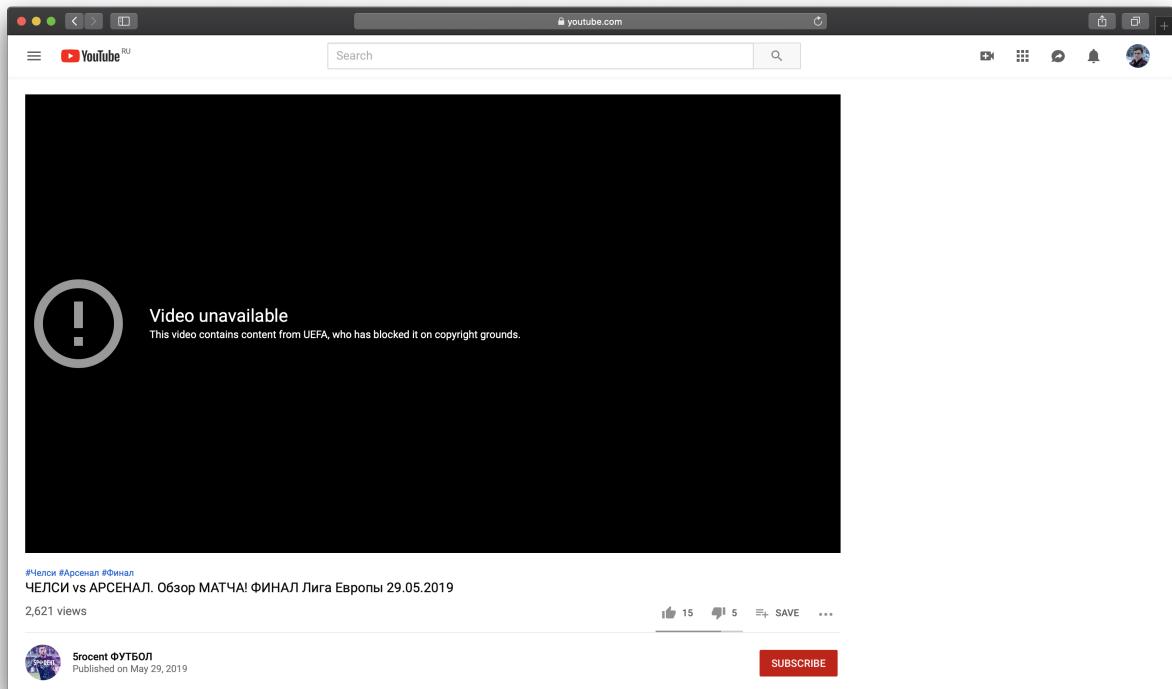


Рисунок 4. Пример неработающего видео на сайте www.youtube.com и на сайте Yandex Video.

Введение

Выбор темы обусловлен спецификой работы сервиса Yandex Video. В силу того, что контент данного сервиса выдается пользователю в виде плееров, интегрированных в сайт с помощью тэга iframe, у данного сервиса возникает необходимость вовремя детектировать неработающие видеозаписи и убирать их из выдачи. На данный момент эта задача решается с помощью периодического обхода видеозаписей с помощью специального механизма, умеющего эмулировать работу клиентского web браузера, а также умеющего нажимать на кнопку начала проигрывания видеозаписи и оценивать, началось ли воспроизведение. Данный механизм позволяет получать достаточно достоверный сигнал о недоступности видео, однако обладает рядом существенных минусов, например, невозможность проверить доступность видео в странах кроме России (так как все запросы территориально отправляются из России), то есть учесть специфику локальных блокировок сайтов и контента. Для устранения подобных недостатков было предложено использовать статистику просмотров для детектирования неработающих документов.

С этой целью по сессиям пользователей собирается анонимная статистика: когда пользователь начинает просмотр видеозаписи, включается таймер, когда же пользователь переключает видео или закрывает страницу сервиса, время просмотра записывается для дальнейшего анализа. Вместе со временем просмотра сохраняются и события плеера, если они доступны. Основной проблемой таких данных является высокая степень загрязненности. Если во время просмотра или переключения видео у пользователя пропадет интернет соединение, запись о просмотре может не прийти или содержать неверные данные. Также возможен случай, когда пользователь включает проигрывание видеозаписи, а затем отходит от компьютера. В это время таймер считает время просмотра, несмотря на то, что видео могло оказаться неработающим и так и не загрузиться.

Задачей данной работы ставится создание модели, которая смогла бы предсказывать неработающие видео с высокой точностью, чтобы данный механизм мог дополнить, а в перспективе и заменить механизм обхода видеозаписей.

В качестве метода исследования был выбран экспериментальный подход: сбор и разметка выборки, оценка точности и полноты классификации с помощью метода перекрестной проверки и анализ предсказаний полученной модели.

Предыдущие работы

Наиболее близкой задачей к поставленной является задача детектирования soft 404 страниц. В соответствии с протоколом HTTP¹, если при обращении к серверу клиент запрашивает документ, который не доступен по той или оной причине, сервер должен возвращать ошибку. Как правило это ошибка 403 (Forbidden Error) или 404 (Not Found). Тем не менее многие интернет ресурсы стремятся предоставить пользователю более дружественный интерфейс и вместо возврата ошибки имитируют нормальную работу, возвращая код возврата 200 (OK), а вместо запрашиваемого документа отображают страницу, оповещающую пользователя о причинах недоступности контента.

Подобное поведение делают задачу детектирования недоступных документов нетривиальной. Первая попытка борьбы с данной проблемой была предпринята в статье от 2004 года². Авторы данной статьи предлагают спровоцировать сервер вернуть заведомо недоступный документ путем прибавления к имеющемуся названию искомого документа произвольного окончания. После этого предлагается сравнивать поведение сервера в случае обращения к исходному документу и к заведомо недоступному. Второе упоминание о проблеме soft 404 датируется 2012 годом³. В отличии от первой статьи, данная работа целиком посвящена проблеме детектирования soft 404 страниц. Авторы этой статьи предлагают использовать классификатор, который использует в качестве данных лексические сигнатуры, содержащиеся в заголовке или в тексте страницы.

К сожалению ни один из предложенных способов детектирования soft 404 документов не может быть применен к задаче детектирования неработающих видео. Это вызвано тем, что, в отличии от других интернет ресурсов, видеохостинги при обращении к несуществующему или уже удаленному документу чаще всего предпочитают показывать страницу, полностью идентичную странице, которая была бы отображена в случае, если видео было доступно для просмотра. Сообщение о недоступности видеозаписи как правило располагается в самом видеоплеере. Таким образом подход, основанный на детектировании переадресаций на заранее заготовленные документы, содержащие оповещение о недоступности видео не применим так как в данном случае переадресаций не

¹Roy T. Fielding, Julian Reschke. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231. Июнь 2014. DOI: 10.17487/RFC7231. URL: <https://rfc-editor.org/rfc/rfc7231.txt>.

²Ziv Bar-Yossef и др. “Sic transit gloria telae”. B: Proceedings of the 13th conference on World Wide Web - WWW '04. ACM Press, 2004. DOI: 10.1145/988672.988716.

³Luis Meneses, Richard Furuta, Frank Shipman. “Identifying “Soft 404” Error Pages: Analyzing the Lexical Signatures of Documents in Distributed Collections”. B: Theory and Practice of Digital Libraries. Springer Berlin Heidelberg, 2012, c. 197–208. DOI: 10.1007/978-3-642-33290-6_22.

происходит. Анализ лексических сигнатур, содержащихся на странице и ее заголовке также неприменим в силу того, что и тот и другой текст не зависит от доступности видеозаписи, а само сообщение о возникшей проблеме находится в плеере. Таким образом для детектирования таких сообщений требуется технически сложный механизм, который умеет эмулировать доступ к странице из пользовательского web браузера и попытку воспроизведения видеозаписи.

Подготовка данных

5.1 Исходные данные

В качестве исходных данных имеется таблица, каждая строка которой содержит некоторый идентификатор плеера, идентификатор видеозаписи, а также время просмотра и события плеера, если они доступны. Также из разметки сайта провайдера контента берется продолжительность видеозаписи. Пример исходных данных представлен в таблице 1.

Плеер	Видео	Длительность, с	Просмотр, с	Старт	Ошибка
Плеер1	Видео1	100	10	True	
Плеер2	Видео2	200	180	True	
Плеер3	Видео3	0	5		True
...

Таблица 1 — Пример исходных данных.

5.2 Выбор признаков

Так как данные сильно зашумлены, клики необходимо усреднять. При этом события плеера — достаточно надежный источник информации, тем не менее они имеют несколько существенных недостатков. К этим недостаткам относится тот факт, что события плеера доступны не для всех плееров, а также каждомуциальному плееру характерен некоторый профиль событий для различных состояний видеозаписи, однако эти профили могут существенно отличаться для двух разных плееров. Существуют два различных способа борьбы с данными недостатками: во-первых можно сделать больший упор на время просмотра и производимые из него признаки, как на более универсальный источник информации, во-вторых можно попытаться сделать отдельную модель для каждого плеера, чтобы она смогла в большей мере использовать сигнал событий плеера, если они доступны. В ходе данной работы были опробованы оба подхода.

Также в силу высокого уровня зашумленности исходных данных понятно, что их необходимо каким-то образом усреднить. В случае с данным событием видеоплеера усреднение очевидно — если событие произошло, то данной записи ставится в соответствие единица, в противном случае ноль и в качестве признака используется среднее значение, то есть доля просмотров, в которых данное событие наблюдалось.

Из времени просмотра было выбрано сгенерировано большее количество признаков исходя из некоторых предположений о природе данных. Первое предположение заключается в том, что, если видеозапись не работает, большинство пользователей заметят это и переключать видео за довольно короткий промежуток времени, близкий к 30-40 секундам. Поэтому в качестве первой группы признаков были выбраны доли пользователей смотревших видео не менее 15, 30 и 45 секунд. Следующая идея говорит о том, что информация о том, как хорошо пользователи смотрят данное видео хранится в распределении доли просмотра контента. Для простоты и однообразности расчета распределение было введено в выборку в виде долей пользователей, досмотревших видео до 5, 10, 20, 30, 40, 50, 60, 70, 80 и 90 % длительности видеозаписи. Последнее наблюдение говорит о том, что если видеозапись попала на выдачу, то когда-то она была обнаружена обходом, а значит в какой-то момент времени она работала. Таким образом чаще всего мы занимаемся не поиском неработающего контента, а пытаемся поймать момент, в который видеозапись перестала работать. В таких случаях принято говорить о данных в форме временного ряда, то есть последовательных значениях, зависящих от времени. Однако для удобства работы с данными и расширения круга потенциальных алгоритмов классификации хотелось бы иметь данные в виде вектора фиксированной размерности. Также важно помнить о шуме в данных, а значит и необходимости усреднения данных. Для соблюдения всех ограничений было принято решение воспользоваться методом скользящего среднего. В таком случае мы получаем, например, такие признаки как среднее время просмотра за 5 последних просмотров заканчивая последним записанным просмотром и такое же среднее за 5 просмотров заканчивая предпоследним просмотром и так далее. Хотелось бы заметить, что подход со скользящим средним применим не только к времени просмотра. Скользящее среднее также считалось для событий плеера и долей смотревших видео хотя бы 15, 30, 45 секунд и так далее.

В итоге для каждого видео мы получили вектор признаков, размерность которого составила порядка 150. После того как были выбраны признаки и собрана выборка, разметка видеозаписей производилась с помощью сервиса Yandex Toloka. Таким образом был собран датасет из порядка 100000 размеченных объектов.

Обучение классификатора

6.1 Выбор модели

Заметим, что пространство признаков имеет очень большую размерность. К тому же многие из признаков сильно взаимосвязаны с другими (действительно, доля досмотревших до 90 % видео не может быть выше доли досмотревших до 50 %). Если с размерностью можно справиться, например, воспользовавшись методом главных компонент¹, то избавиться от зависимости между признаками гораздо сложнее. В данной работе было принято решение использовать модели машинного обучения, основанные на решающих деревьях². Это позволяет исключить из предобработки уменьшение размерности признаков, а также перестать беспокоиться о зависимостях в признаках. Однако решающие деревья в чистом виде используются редко из-за высокой чувствительности к значениям гиперпараметров и низкой обобщающей способностью модели. Однако решающие деревья показывают очень хорошие результаты, когда они используются в виде разных ансамблей.

В качестве базовой модели был выбран случайный лес³. Данный выбор обусловлен тем что метод зарекомендовал себя как неприхотливый алгоритм, который тем не менее показывает довольно высокие результаты практически во всех задачах. Также было замечено, что выборка в 100000 объектов не очень велика. Также важно понимать, что среди этих объектов только около 1 % видеозаписей были размечены как неработающие. Таким образом выборка содержит довольно мало сигнала, который мы хотим детектировать. Поэтому в качестве дополнительной модели был выбран метод чрезвычайно случайных деревьев (*extremely randomized trees*)⁴, который позволяет в

¹Karl Pearson. “LIII. On lines and planes of closest fit to systems of points in space”. B: The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2.11 (нояб. 1901), с. 559–572. DOI: 10.1080/14786440109462720.

²California USA) Breiman Jerome (Stanford University California USA) Friedman Charles J. (University of California Berkeley USA) Stone R. A. (Stanford California USA) Olshen Leo (Consultant Berkeley. Classification and Regression Trees. Taylor & Francis Ltd, 1 янв. 1984. 368 с. ISBN: 0412048418. URL: https://www.ebook.de/de/product/3606994/leo_consultant_berkeley_california_usa_breiman_jerome_stanford_university_california_usa_friedman_charles_j_university_of_california_berkeley_usa_stone_r_a_stanford_california_usa_olshen_classification_and_regression_trees.html.

³Tin Kam Ho. “Random decision forests”. B: Proceedings of 3rd International Conference on Document Analysis and Recognition. IEEE Comput. Soc. Press, 1995. DOI: 10.1109/icdar.1995.598994.

⁴Pierre Geurts, Damien Ernst, Louis Wehenkel. “Extremely randomized trees”. B: Machine Learning 63.1 (март 2006), с. 3–42. DOI: 10.1007/s10994-006-6226-1.

некоторых случаях уменьшить разброс предсказаний за счет небольшого увеличения смещения предсказания. В качестве последней модели был выбран метод градиентного бустинга (gradient boosting)⁵ на решающих деревьях, как наиболее точной хотя и наименее неприхотливый алгоритм.

Численные эксперименты производились с помощью языка программирования Python⁶. Реализации первых двух алгоритмов были взяты из библиотеки scikit-learn⁷, реализация последнего была взята из библиотеки CatBoost⁸.

6.2 Обучение модели

Для подбора гиперпараметров и получения оценки точности и полноты использовался сеточный поиск с оценкой точности и полноты с использованием 5-кратной кроссвалидации. Для случайного леса и чрезвычайно случайных деревьев подбиралось количество деревьев в ансамбле. Для классификатора CatBoost кроме этого подбиралась максимальная глубина дерева.

Результаты сеточного поиска представлены в таблице 2, а также на рисунках 5, 6 и 7.

Модель	Наилучшая точность, %	Наилучшая полнота, %
RandomForestClassifier	72.0	50.8
ExtraTreesClassifier	74.3	47.2
CatBoostClassifier	80.3	53.3

Таблица 2 — Результаты сеточного поиска

Наилучшей точности удалось достигнуть с использованием классификатора из библиотеки CatBoost (свыше 80.3 %). Полнота у данного классификатора получилась выше, чем у конкурентов (около 51.5 %).

При дальнейшем изучении предсказаний данной модели выяснились следующие факты: выборка содержит некоторое количество видео, неверно размеченных, как работающие, и плеерные события имеют большой вес в предсказании модели. Первая проблема требует другого способа разметки данных, и ее решение будет предложено далее. Для решения же второй проблемы было предложено добавить в признаки идентификатор плеера. Это позволило бы модели корректировать ожидаемый профиль событий в зависимости от конкретного плеера, который их шлет. Результаты этого эксперимента представлены на рисунке 8. К сожалению, точность классификатора упала до 72.6 %, однако полнота классификации возросла почти на 17 % и составила почти 68.2 %.

⁵ Jerome H. Friedman. “Greedy function approximation: A gradient boosting machine.” B: The Annals of Statistics 29.5 (окт. 2001), с. 1189–1232. DOI: 10.1214/aos/1013203451.

⁶ Python Programming Language. 2019. URL: <https://www.python.org>.

⁷ F. Pedregosa и др. “Scikit-learn: Machine Learning in Python”. B: Journal of Machine Learning Research 12 (2011), с. 2825–2830.

⁸ Prokhorenkova и др., “CatBoost: unbiased boosting with categorical features”.

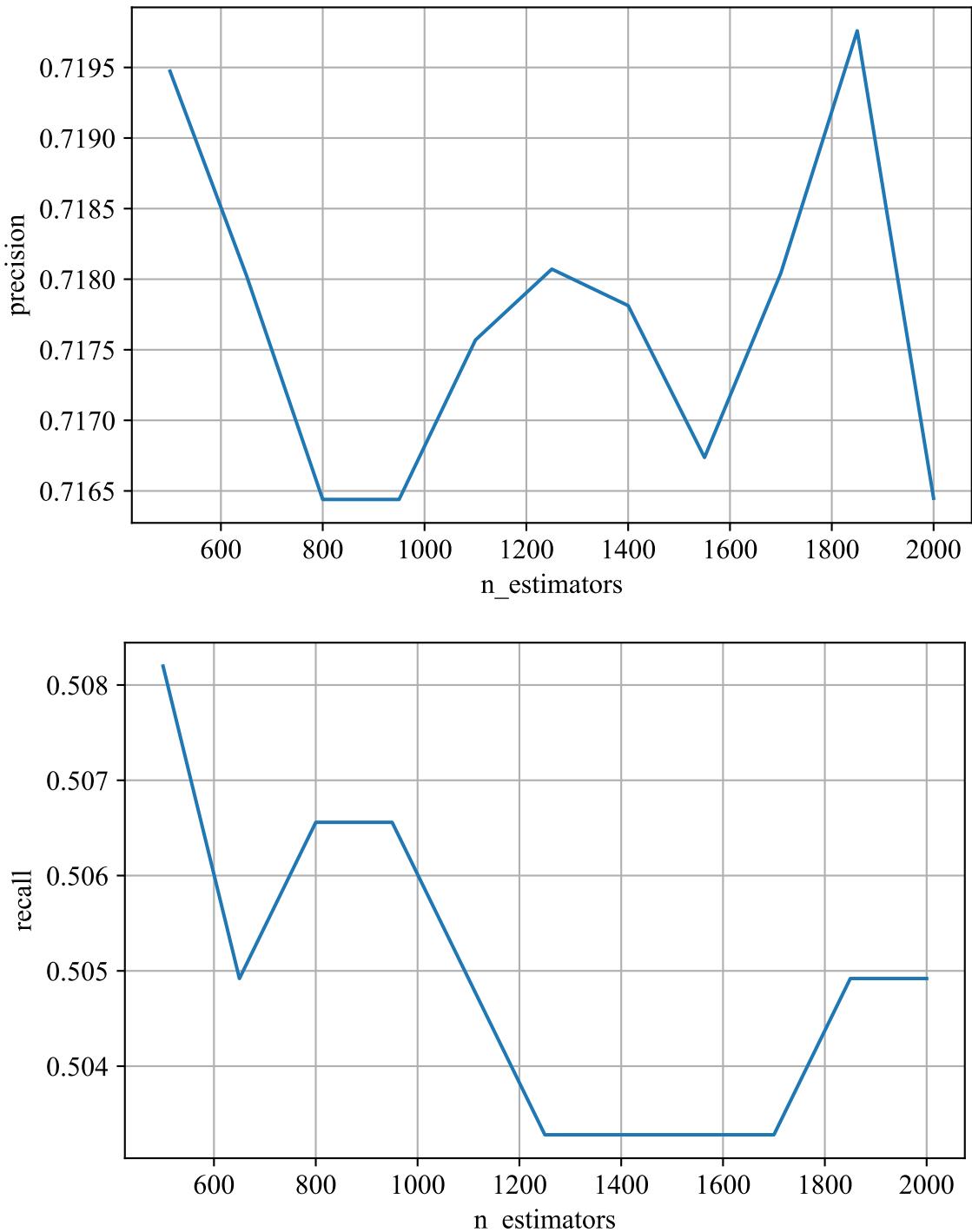


Рисунок 5 — Результаты сеточного поиска для модели RandomForestClassifier

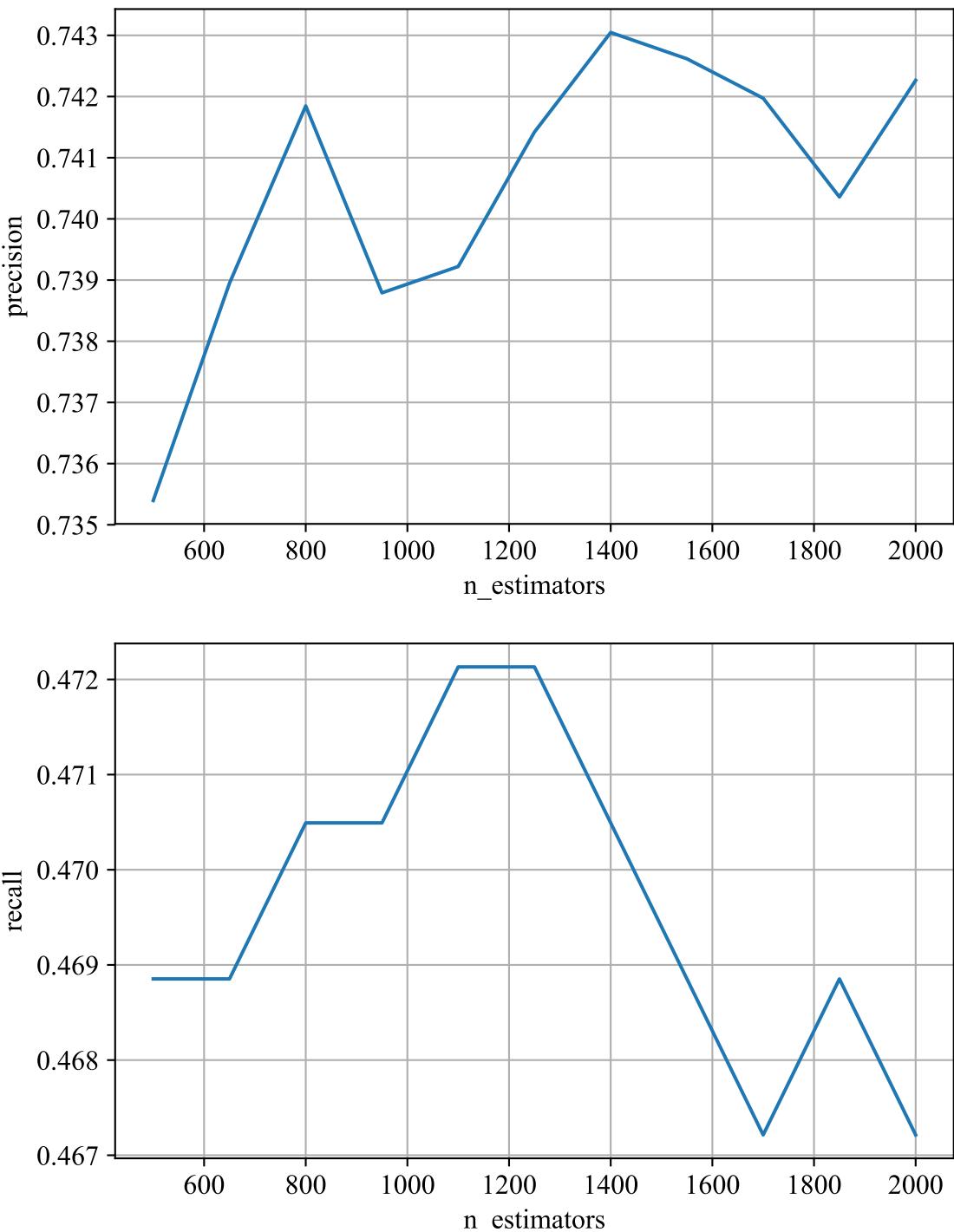


Рисунок 6 — Результаты сеточного поиска для модели ExtraTreesClassifier

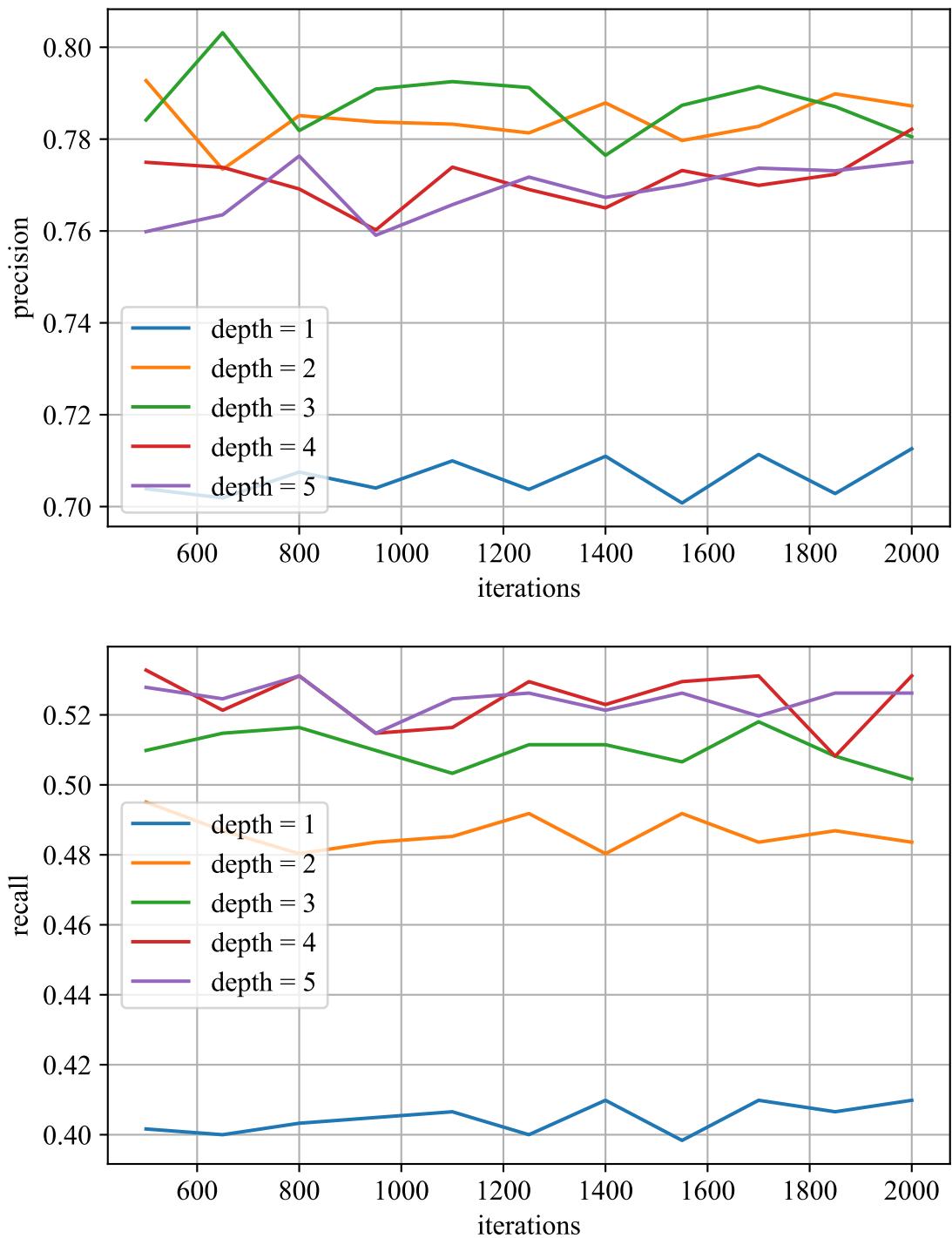


Рисунок 7 — Результаты сеточного поиска для модели CatBoostClassifier

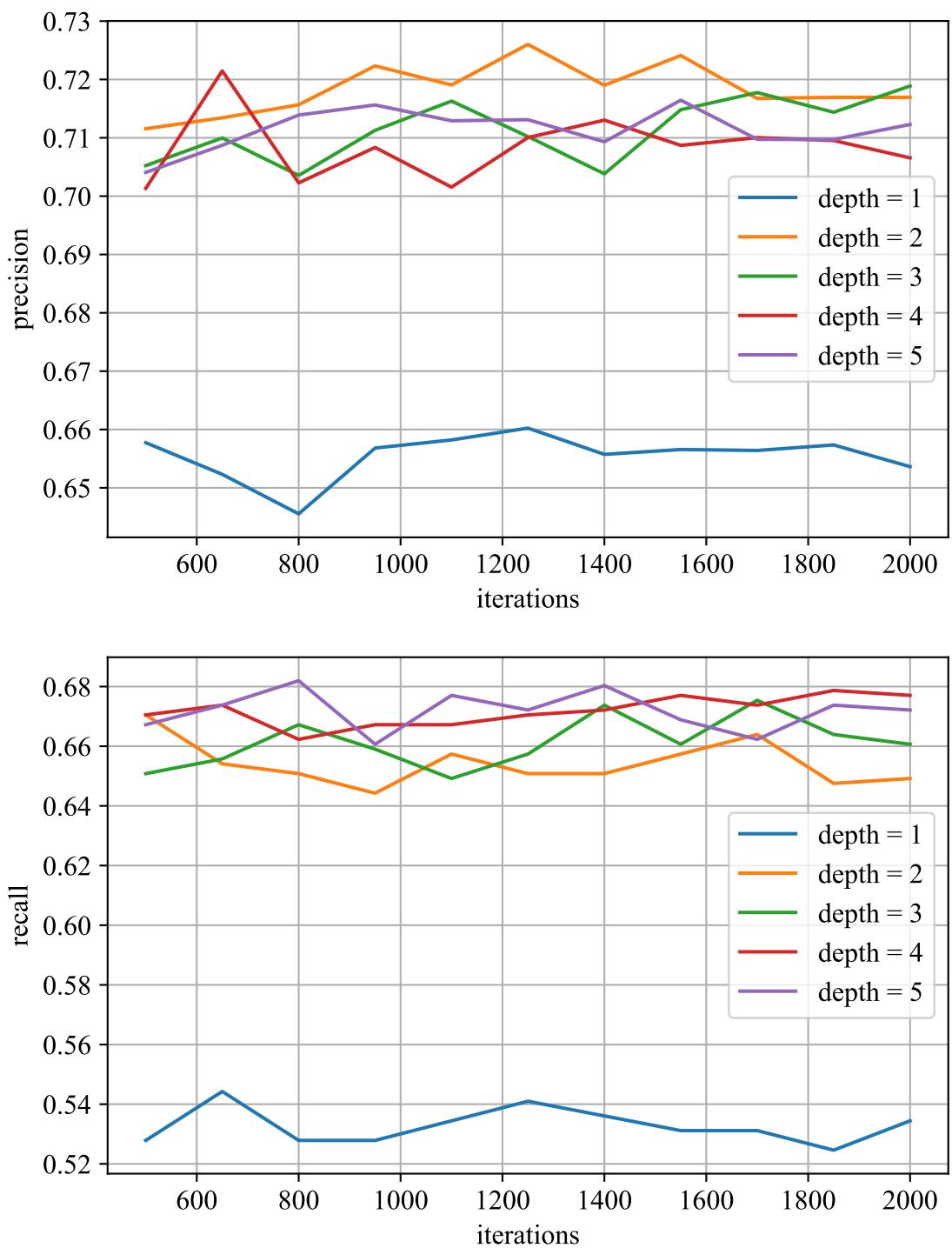


Рисунок 8. Результаты сеточного поиска для модели CatBoostClassifier с категориальным признаком "идентификатор плеера"

6.3 Одноплеерная модель

Для решения проблемы неаккуратной разметки данных, было принято решение размечать выборку с помощью данных обхода. В таком случае возникает обратная проблема: у нас есть информация о том, какие видео не работали в выбранный день, однако у нас нет достоверного обратного сигнала. Для решения данной проблемы был выбран следующий подход: известно, что среди видеозаписей, которые представлены на выдаче поиска, примерно 1 % не работает. Исходя из этого, при составлении выборки мы считаем количество достоверно неработавших видео исходя из логов обхода и им присваивается метка класса 1. Остальные видео ранжируются по среднему времени просмотра и выбирается в 99 раз большее количество видео, чем получившийся объем первого класса. Эти видео размечаются меткой класса 0 и также добавляются в выборку. Таким образом мы получаем размеченную выборку большого размера, так как нам доступна история обхода за довольно длительный период времени. Поэтому было решено учить модель только на данных одного плеера, что позволяет модели подстраиваться под конкретный профиль событий и тип контента. В качестве модели здесь выбран только CatBoostClassifier в силу того, что он зарекомендовал себя, а также должен лучше себя вести на сильно возросшей выборке. Результаты сеточного поиска в данном случае представлены на рисунке 9.

Мы видим заметный прирост точности классификации (более 9 %) при практически неизменной полноте по сравнению с классификацией данных, размеченных с помощью сервиса Yandex Toloka. В результате удалось обучить классификатор из библиотеки CatBoost, который достиг точности классификации 89.3 % и полноты 54.7 %.

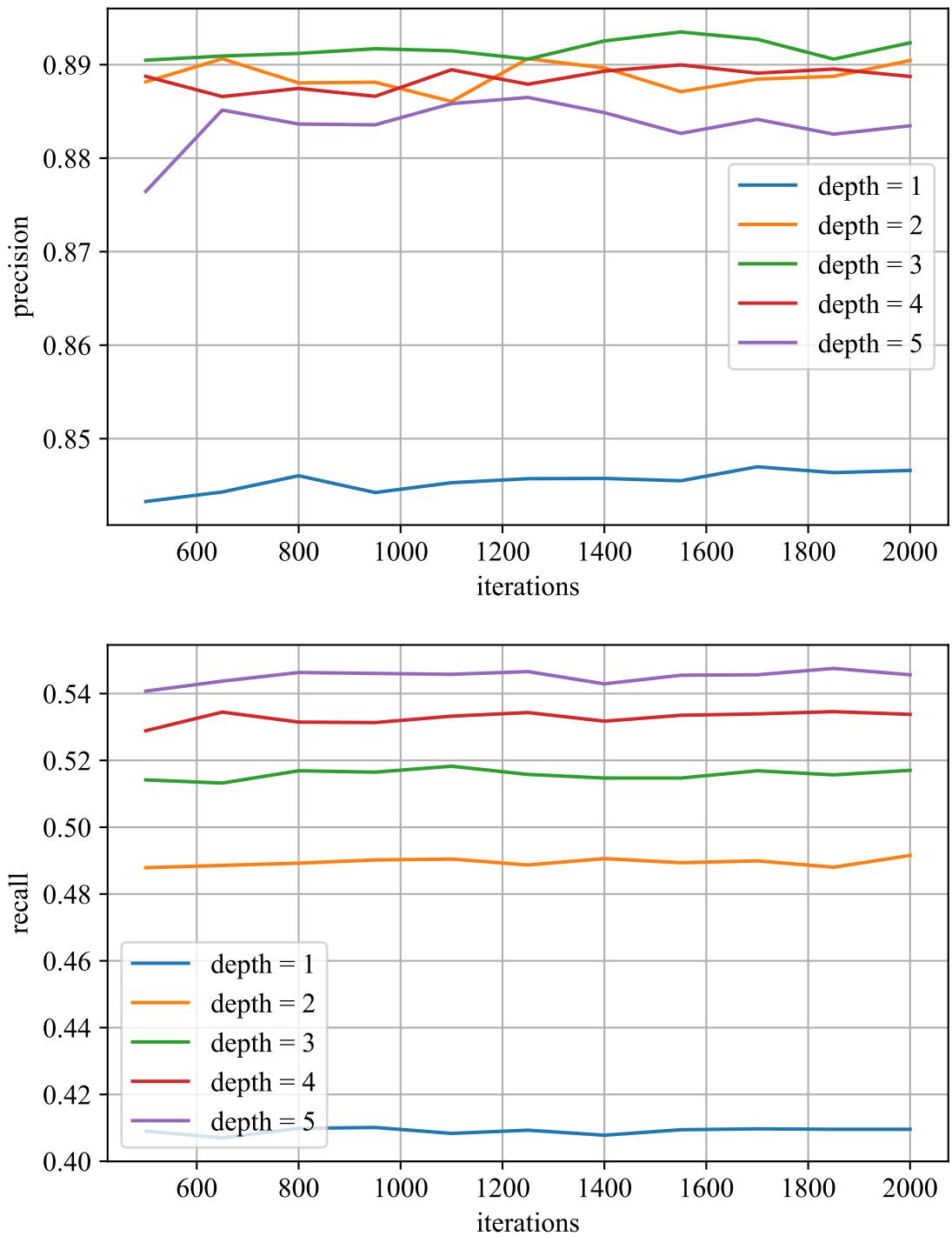


Рисунок 9. Результаты сеточного поиска для модели CatBoostClassifier обученной на данных обхода

Заключение

Список литературы

- Bar-Yossef, Ziv и др. “Sic transit gloria telae”. B: Proceedings of the 13th conference on World Wide Web - WWW '04. ACM Press, 2004. DOI: 10.1145/988672.988716.
- Fielding, Roy T., Julian Reschke. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231. Июнь 2014. DOI: 10.17487/RFC7231. URL: <https://rfc-editor.org/rfc/rfc7231.txt>.
- Friedman, Jerome H. “Greedy function approximation: A gradient boosting machine.” B: The Annals of Statistics 29.5 (окт. 2001), c. 1189—1232. DOI: 10.1214/aos/1013203451.
- Geurts, Pierre, Damien Ernst, Louis Wehenkel. “Extremely randomized trees”. B: Machine Learning 63.1 (март 2006), c. 3—42. DOI: 10.1007/s10994-006-6226-1.
- Ho, Tin Kam. “Random decision forests”. B: Proceedings of 3rd International Conference on Document Analysis and Recognition. IEEE Comput. Soc. Press, 1995. DOI: 10.1109/icdar.1995.598994.
- Leo (Consultant Berkeley, California USA) Breiman Jerome (Stanford University California USA) Friedman Charles J. (University of California Berkeley USA) Stone R. A. (Stanford California USA) Olshen. Classification and Regression Trees. Taylor & Francis Ltd, 1 янв. 1984. 368 c. ISBN: 0412048418. URL: https://www.ebook.de/de/product/3606994/leo_consultant_berkeley_california_usa_breiman_jerome_stanford_university_california_usa_friedman_charles_j_university_of_california_berkeley_usa_stone_r_a_stanford_california_usa_olshen_classification_and_regression_trees.html.
- Meneses, Luis, Richard Furuta, Frank Shipman. “Identifying “Soft 404” Error Pages: Analyzing the Lexical Signatures of Documents in Distributed Collections”. B: Theory and Practice of Digital Libraries. Springer Berlin Heidelberg, 2012, c. 197—208. DOI: 10.1007/978-3-642-33290-6_22.
- O’Hara, Scott и др. HTML 5.3. W3C Working Draft. <https://www.w3.org/TR/2018/WD-html53-20181018/>. W3C, окт. 2018.
- Pearson, Karl. “LIII. On lines and planes of closest fit to systems of points in space”. B: The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2.11 (нояб. 1901), c. 559—572. DOI: 10.1080/14786440109462720.
- Pedregosa, F. и др. “Scikit-learn: Machine Learning in Python”. B: Journal of Machine Learning Research 12 (2011), c. 2825—2830.
- Prokhorenkova, Liudmila и др. “CatBoost: unbiased boosting with categorical features”. B: (28 июня 2017). arXiv: <http://arxiv.org/abs/1706.09516v5> [cs.LG].

Python Programming Language. 2019. URL: <https://www.python.org>.

Yandex Toloka. 2019. URL: <https://toloka.yandex.com>.

Yandex Video. 2019. URL: <http://yandex.ru/video/>.

YouTube Player API Reference for iframe Embeds. Май 2018. URL: https://developers.google.com/youtube/iframe_api_reference.