

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 |
| 3. Введение | 8 |
| 4. Предыдущие работы | 9 |
| 5. Подготовка данных | 12 |
| 5.1. Сырые 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 | 15 |
| 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.

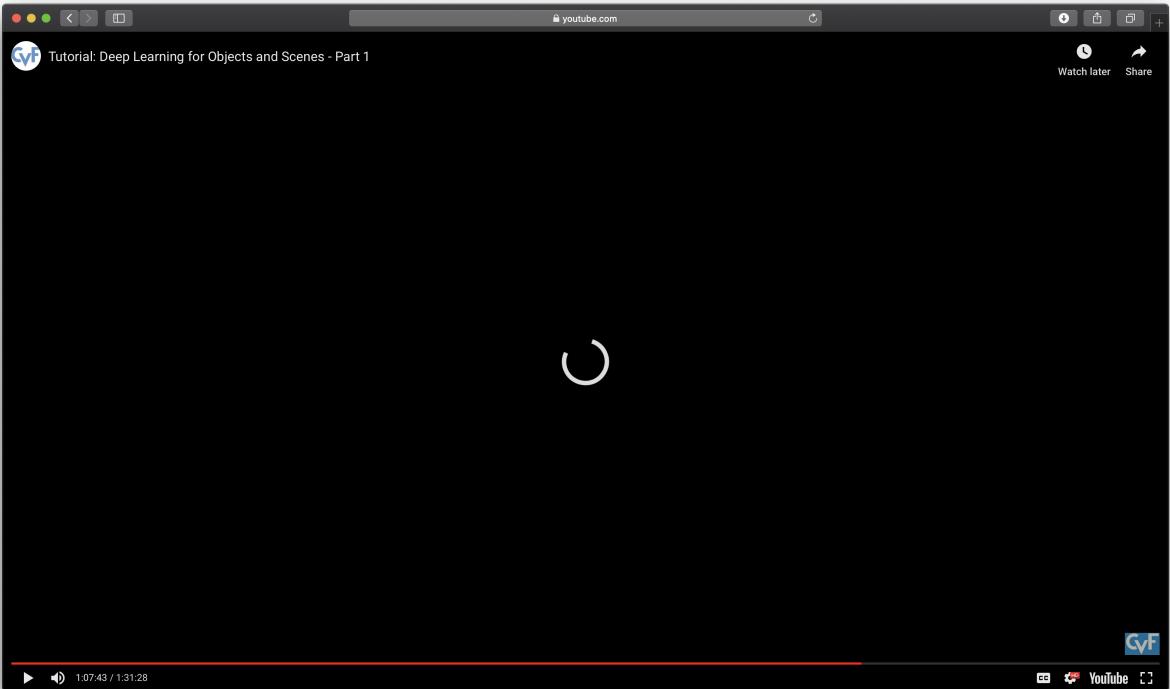


Рисунок 1 — Видеоплеер YouTube.

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/>.

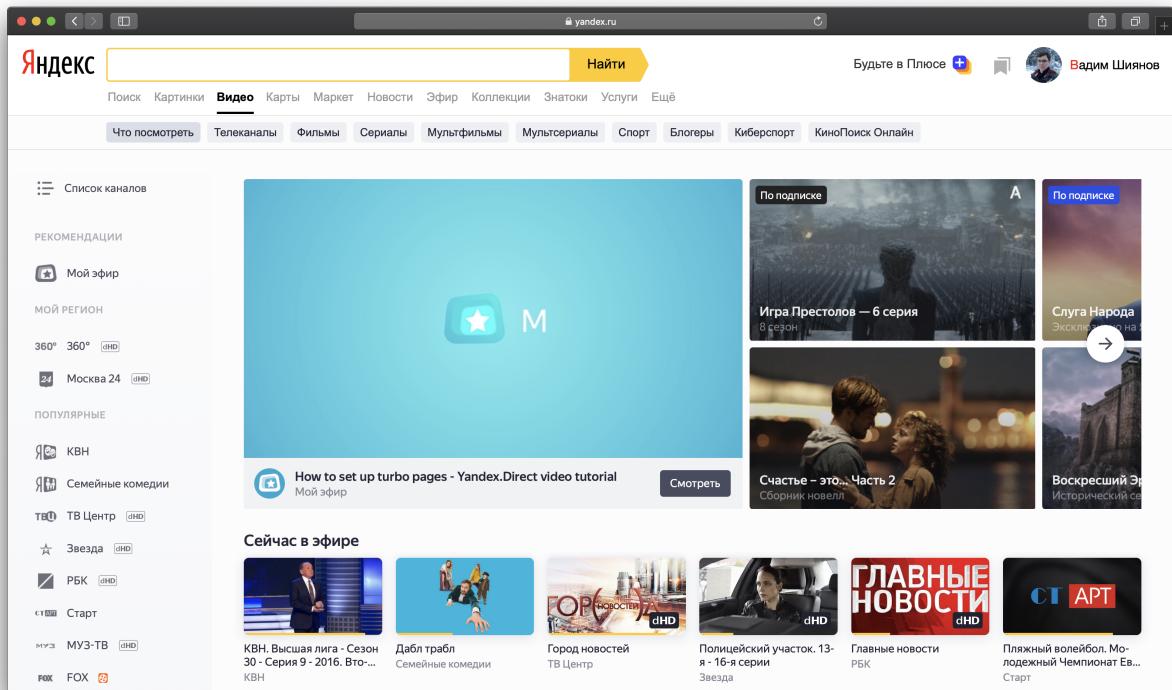


Рисунок 2 — Сервис Yandex Video.

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.

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

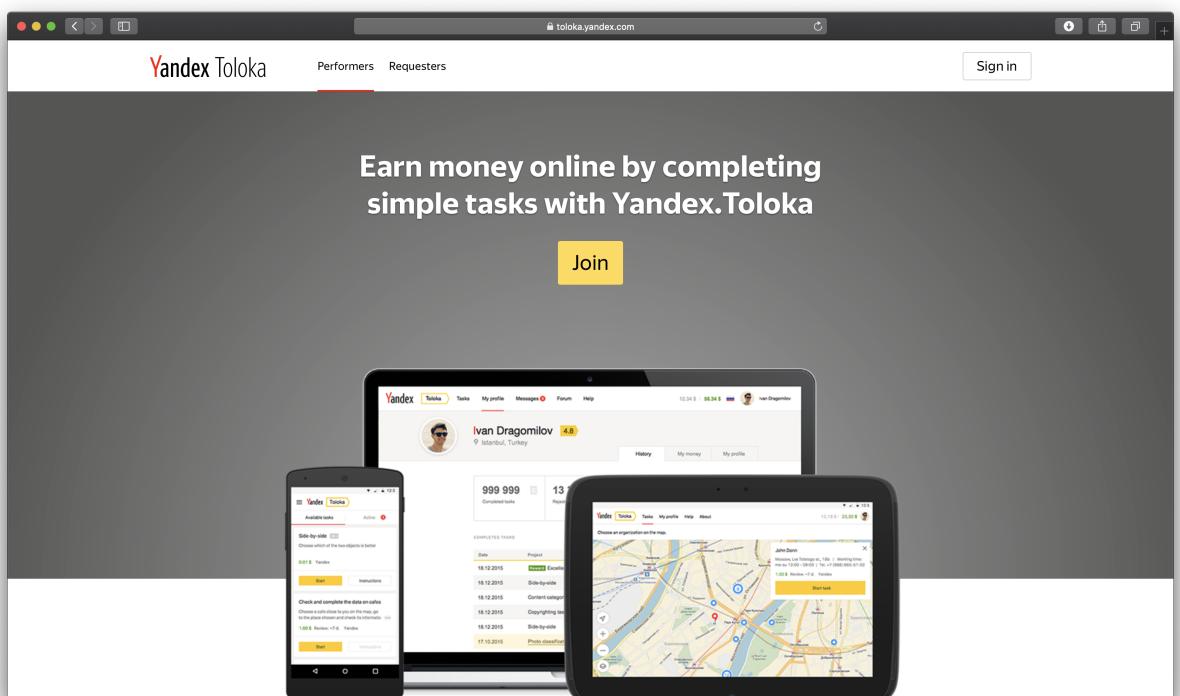


Рисунок 3 — Сервис Yandex Toloka.

Введение

Выбор темы обусловлен спецификой работы сервиса Видео Поиск Yandex, в команде разрабатывающей который я делал НИР. Сервис позволяет пользователю искать видео содержащиеся на различных ресурсах интернета. В силу большого количества контента, собранного с различных интернет ресурсов, хранить все видеозаписи не представляется возможным. Поэтому контент на выдаче предоставляется в виде iframe'ов. Это решает проблему хранения видеозаписей, однако порождает другую проблему, а именно необходимость контролировать работоспособность провайдеров контента. С этой целью по кликам пользователей собирается анонимная статистика: когда пользователь кликает по видеозаписи, включается таймер, когда же пользователь переключает видео, к нам приходит время просмотра предыдущей. Основной проблемой здесь является высокая степень загрязненности данных. Если во время просмотра или переключения видео у пользователя пропадет интернет соединение, нам может либо не прийти клик, либо прийти клик с невалидными данными. Также возможен случай, когда пользователь открывает видео и отходит от компьютера. В это время таймер продолжает считать время просмотра, несмотря на то, что видео может так и не загрузиться, например, в силу ошибки на сервере провайдера.

Задачей данной работы ставится создание модели, которая смогла бы предсказывать неработающие видео с высокой точностью, так чтобы видео можно было убирать из выдачи только на основании ее предсказания. В качестве метода исследования был выбран экспериментальный подход: сбор и разметка выборки, вычисление метрик точности и полноты классификатора с помощью кроссвалидации, обучение модели и применение ее на свежих данных с последующей разметкой результатов.

Предыдущие работы

Наиболее близкой задачей к поставленной является задача детектирования так называемого мягкого 404 (мягкой ошибки 404, soft 404). В соответствии с протоколом HTTP¹, когда при обращении к серверу клиент запрашивает документ, который более не доступен, сервер должен возвращать ошибку. Как правило это 404 (Not Found) код ошибки. Мягкий 404 чаще всего возникает в следствие попыток сайта предоставить пользователю более дружественный интерфейс. В таком случае, когда пользователь запрашивает недоступный документ, сервер возвращает не ошибку 404, а код 200 (OK) и страницу, на которой в более удобочитаемом формате написано, что документ недоступен.

В таком случае детектирование недоступных документов становится нетривиальной задачей. Мне удалось найти две статьи, авторы которых подходят к ее решению. В первой из них² предлагается "спровоцировать" сервер вернуть недоступный документ и сравнивать поведение сервера в случае обращения к искомому документу и к заведомо недоступному. Во второй статье³ авторы предлагают классифицировать страницы используя лексические сигнатуры, содержащиеся в заголовке страницы.

Схожесть задачи детектирования неработающих видео с задачей детектирования мягкого 404 обусловлена тем, что видеохостинги также нередко предпочитают информировать пользователя о недоступности видео с помощью сообщения в плеере (смотри рисунок 4). Особенно часто это проявляется в случае с видеозаписями, которые были доступны ранее, а затем стали недоступными для просмотра по той или иной причине.

К сожалению ни один из способов детектирования мягкого 404 не может быть применен к задаче детектирования неработающих видео. Это вызвано тем, что в данном случае текст почти всегда показывает человеку плеер, который работает внутри веб браузера. Таким образом простого скачивания html файла уже недостаточно. Требуется полноценный механизм, способный эмулировать нажатие на кнопку проигрывания и оценивать началось ли воспроизведение видео. Метод же, предлагаемый в данной работе, базируется на использовании статистики просмотров видеозаписи, что позволяет

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.

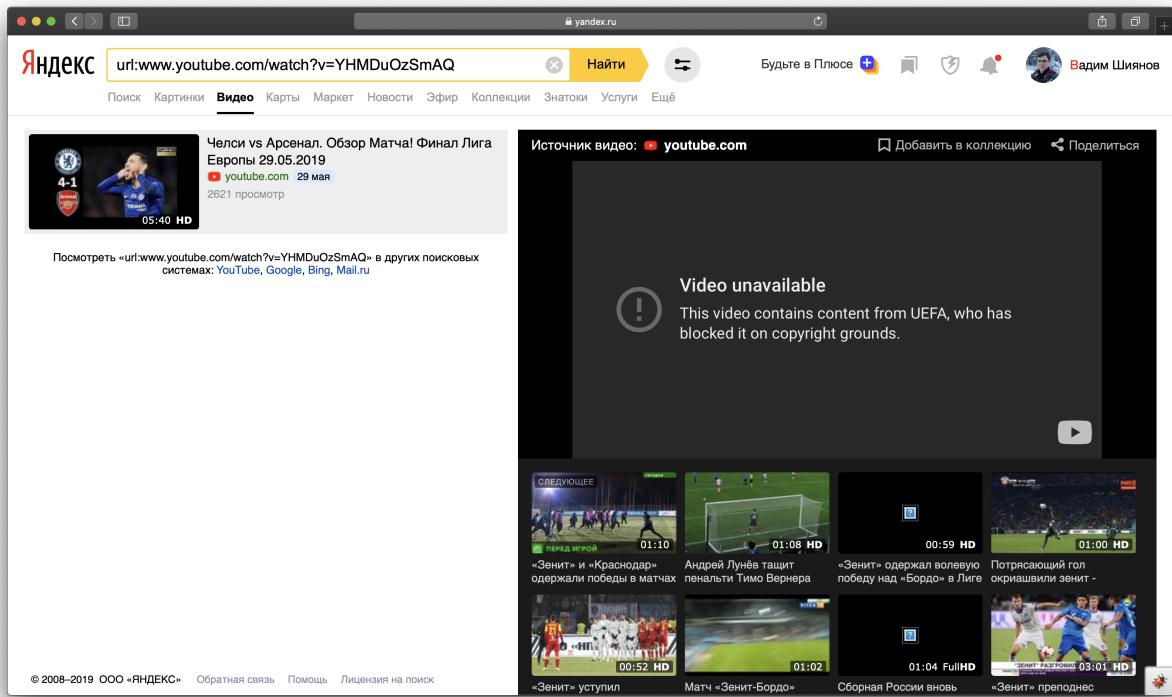
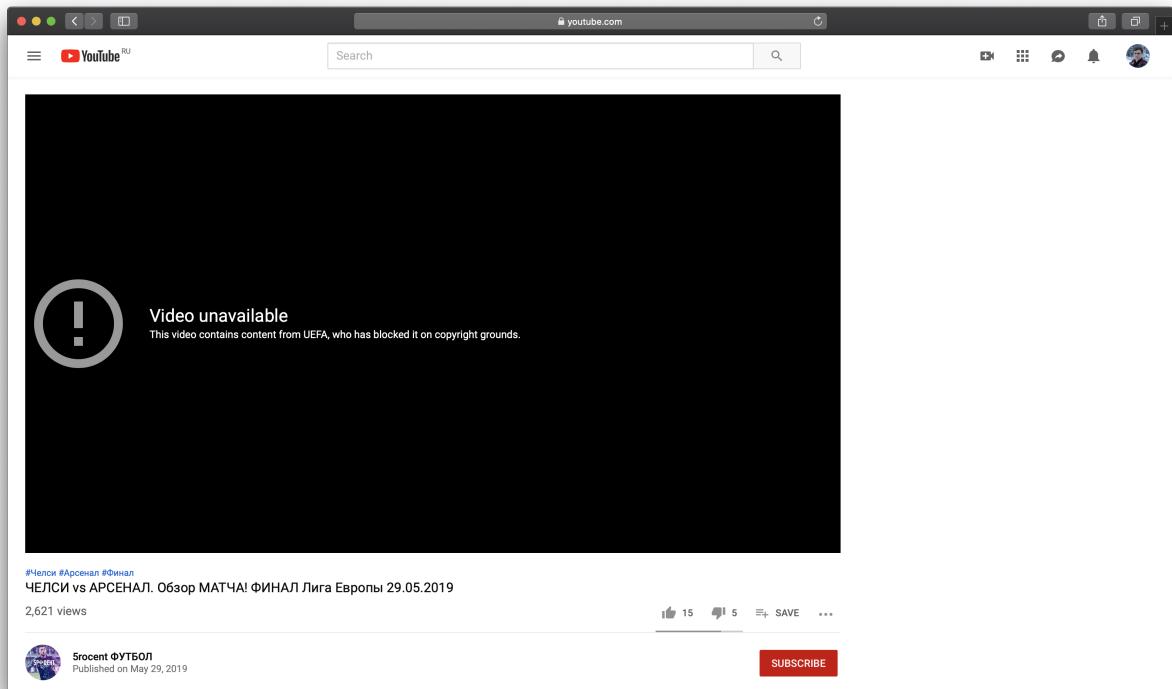


Рисунок 4. Пример неработающего видео на сайте www.youtube.com и на сайте видео-поиска.

избежать использования технически сложных средств.

Подготовка данных

5.1 Сырые данные

В качестве сырых данных имеется таблица, каждая строка которой содержит некоторый идентификатор плеера, идентификатор видео, а также время просмотра и плеерные события, если они доступны для данного плеера. Также из разметки сайта провайдера контента берется продолжительность видеозаписи. Здесь и далее строку в данной таблице я буду называть кликом.

Время просмотра вычисляется, как время между переключением видеозаписи или закрыванием страницы. Также здесь учитывается возможность того, что пользователь поставит видео на паузу. Плеерные события — это часть публичного API многих видеохостингов. Если провайдер контента позаботился об удобстве разработчиков, которые будут эмбедить их контент, они предоставляют возможность "подписаться" на их события, такие как смена разрешения видеозаписи, выставление плеера на паузу, ошибка воспроизведения и прочие. Такую функциональности предоставляет, например, сервис YouTube¹.

Разметка данных производилась при помощи сервиса Yandex Toloka².

5.2 Выбор признаков

Так как данные сильно зашумлены, клики необходимо усреднить. При этом плеерные события — достаточно полезный и надежный источник информации, однако они доступны не для всех плееров, поэтому хотелось бы сделать больший упор именно на время просмотра, как на более универсальный источник информации. В качестве признаков для модели были выбраны доля пользователей, которые смотрели видео в течении не менее 15, 30 и 45 секунд, а также доля просмотревших до 5, 10, 20, 30, 40, 50, 60, 70, 80 и 90 % видео. Также в качестве признаков использовались усредненные события старта и ошибки (если событие имело место, то клику ставилась единица, в случае отсутствия события 0 и усреднялось по доступным кликам). Кроме этого было замечено, что фактически мы имеем дело с времененным рядом времени просмотра от момента времени и пытаемся "поймать" момент, когда плеер перестал играть видеозапись. В таком случае полезными показались такие признаки как скользящее среднее

¹YouTube Player API Reference for iframe Embeds.

²Yandex Toloka.

времени просмотра и событий старта и ошибки.

Обучение классификатора

6.1 Выбор модели

Заметим, что пространство признаков имеет очень большую размерность. К тому же многие из признаков явно взаимосвязаны с другими (действительно, доля досмотревших до 90 % видео не может быть выше доли досмотревших до 50 %). Если с первой из этих проблем можно справиться, например, воспользовавшись методом главных компонент¹, то избавиться от второй гораздо сложнее.

Так как борьба с вышеозначенными проблемами сама по себе является нетривиальной задачей, было принято решение использовать модели машинного обучения, основанные на решающих деревьях². Это позволяет свести предобработку данных к минимуму. Однако решающие деревья в чистом виде редко используются из-за сложностей связанных с соблюдением баланса между сложностью модели и ее обобщающей способностью. Поэтому в качестве базовой модели был выбран случайный лес³. В качестве дополнительных моделей были выбраны чрезвычайно случайные деревья (*extremely randomized trees*)⁴, также метод градиентного бустинга (*gradient boosting*)⁵ основанный на решающих деревьях. Первые два алгоритма были взяты из библиотеки scikit-learn⁶, последний был взят из библиотеки CatBoost⁷.

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), c. 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.

⁵Jerome 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.

⁶F. Pedregosa и др. “Scikit-learn: Machine Learning in Python”. B: Journal of Machine Learning Research 12 (2011), с. 2825—2830.

⁷Prokhorenkova и др., “CatBoost: unbiased boosting with categorical features”.

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 1, 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 1 — 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 %.

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

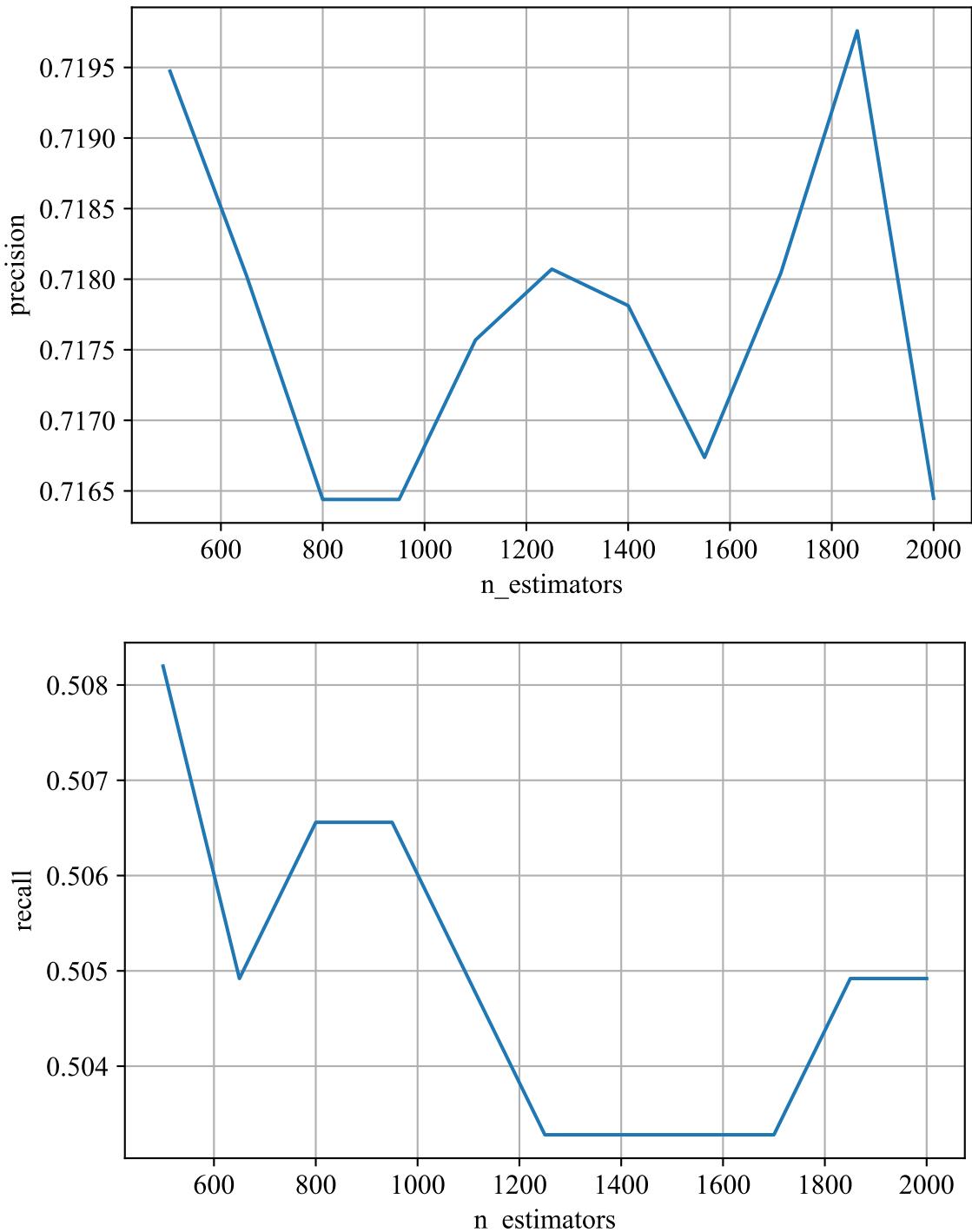


Рисунок 5 — Результаты сеточного поиска для модели RandomForestClassifier

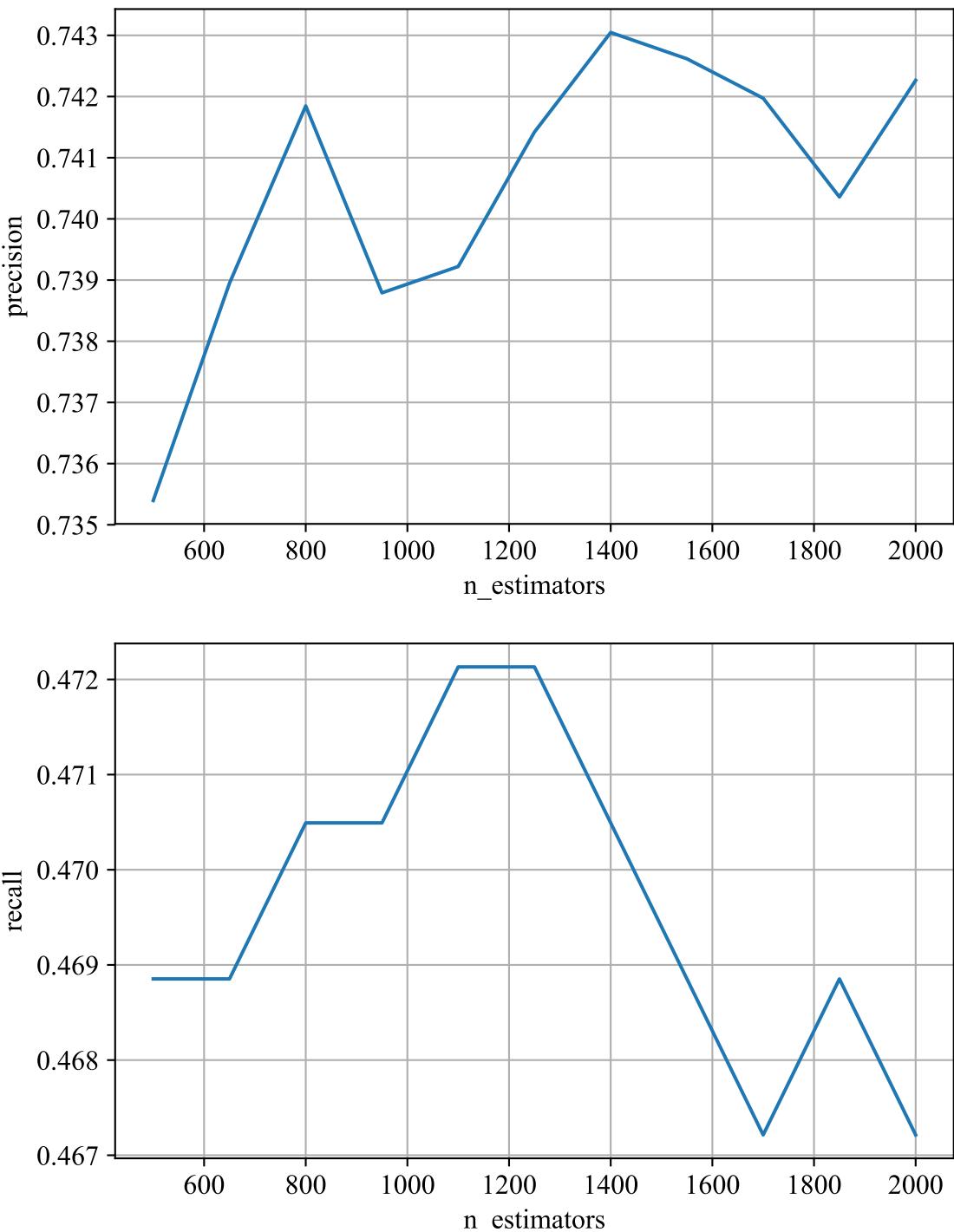


Рисунок 6 — Результаты сеточного поиска для модели ExtraTreesClassifier

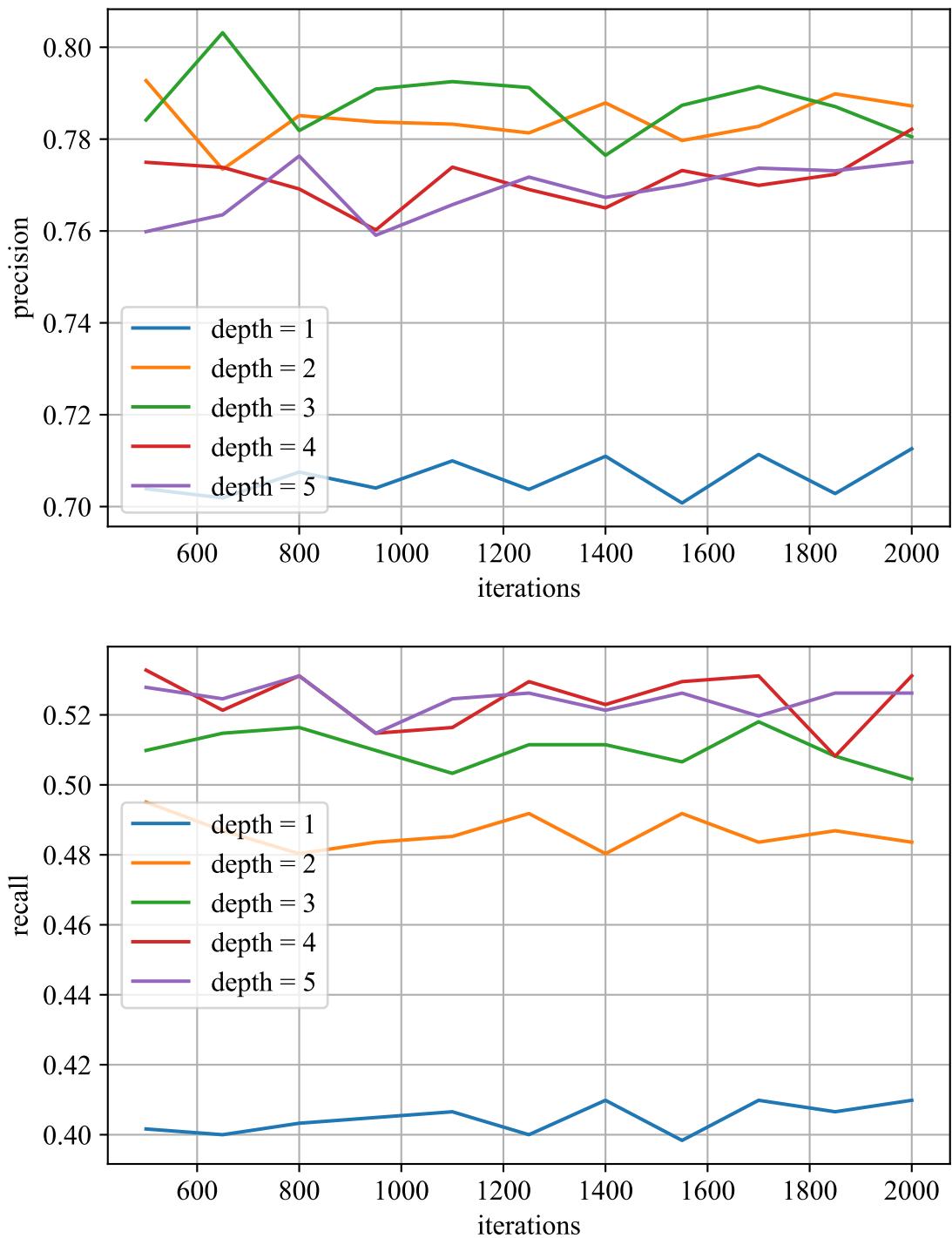


Рисунок 7 — Результаты сеточного поиска для модели CatBoostClassifier

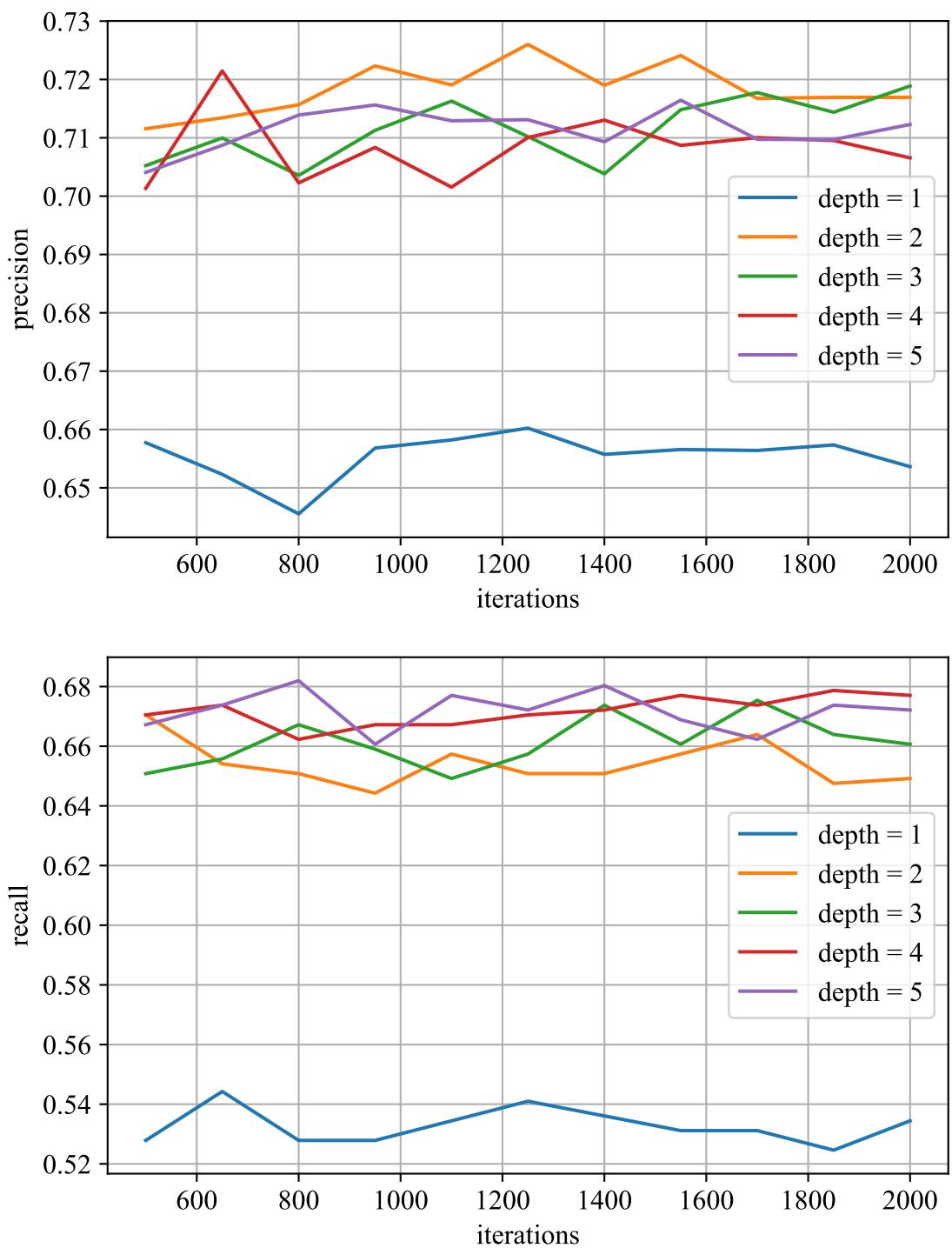


Рисунок 8. Результаты сеточного поиска для модели CatBoostClassifier с категориальным признаком "идентификатор плеера"

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 %.

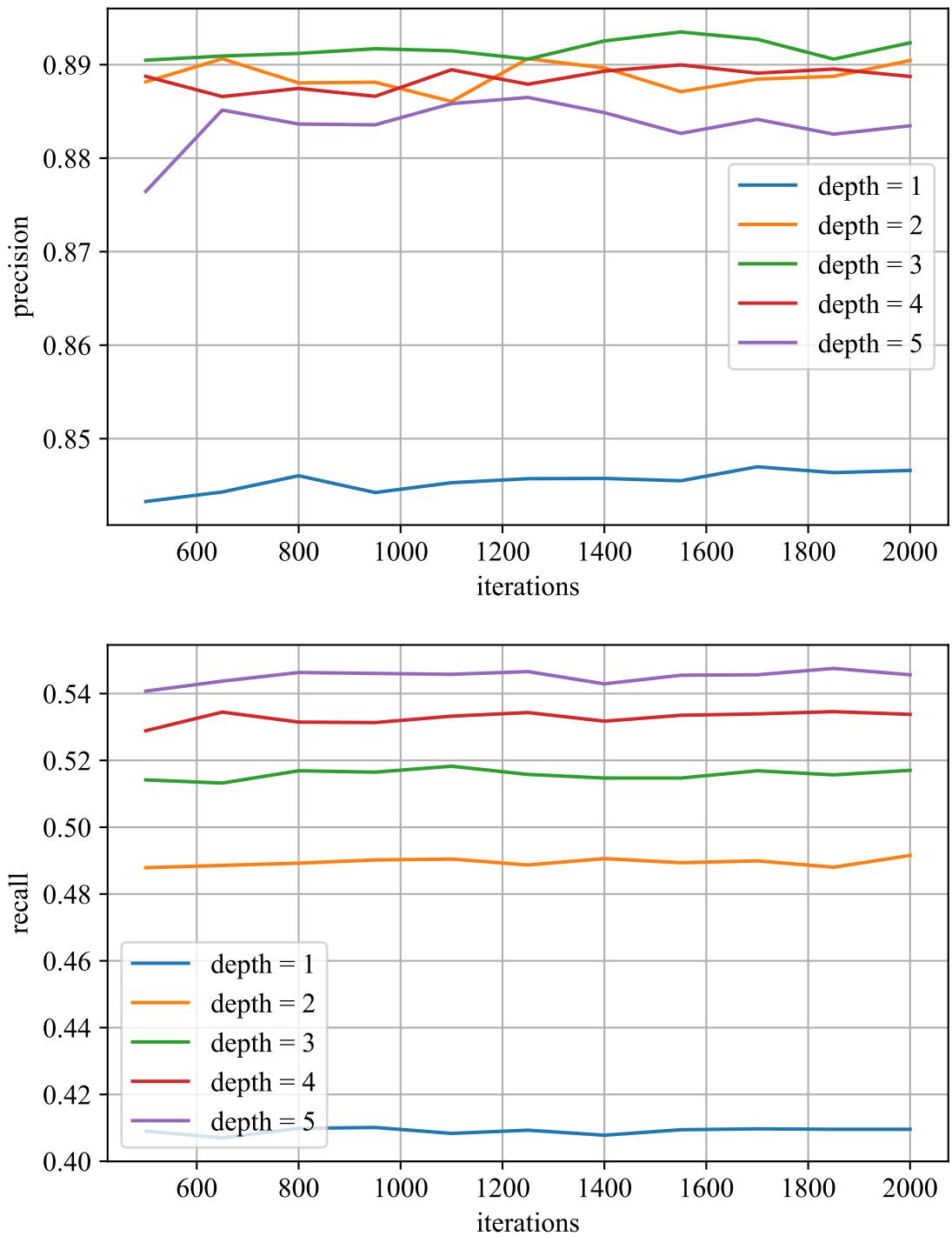


Рисунок 9. Результаты сеточного поиска для модели CatBoostClassifier обученной на данных обхода

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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/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 \[cs.LG\]](https://arxiv.org/abs/1706.09516v5).

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.