Здравствуйте! Меня зовут Быков Илья. Моя работа называется Прогнозирование потенциальной прибыли компьютерных игр. В работе я использовал датасет с данными об играх из платформы Steam. И в начале я провел разведочный анализ данных, чтобы не только проанализировать данные, но и определить какую именно задачу и каким образом можно решить в рамках общей постановки задачи, основываясь на доступных данных. Взглянув на данные и проведя первоначальный анализ, мной было принято решение, поставить задачу таким образом: используя исключительно метки игры, или таги, попытаться предсказать насколько это возможно, ее успешность. Причина поставить задачу таким образом была во первых доступность данных для решения подобной задачи, во вторых, уже имеющееся подобное решение той же задачи на рынке, что увеличивает вероятность того что такая задача вообще разрешима.

Другими словами - спрогнозировать некую метрику успеха игры, основываясь исключительно на метках этой игры, или тагах. Метка игры, это ключевое слово которое лучше всего ее описывает. Это может быть жанр, какой-либо аспект геймплея или графики, например Шутер от первого лица 3D. У каждой игры есть набор таких меток, по которым они будут ранжироваться в поиске Steam, но которыми так же можно емко описать игру. Путем проб и ошибок в итоге было решено использовать только первые 3 метки, на то есть множество причин, самая важная это уменьшение шума в данных, что значительно улучшило в итоге результат.

Итак, ключевыми аспектами разведочного анализа явились: избавление от дубликатов, избавление от пропусков, создание новых полей, которые понадобятся в итоге для вычисления итоговой целевой метрики успеха для наших игр, анализ и исследование методов работы с выбросами, и построение графиков для анализа целевой переменной.

Итак, первым делом мы загрузили датасет, изучили его и правильно поставили задачу, после этого мы должны удалить все ненужные записи, например нам не нужны все игры, у которых меньше 3-х меток, так как это будут для нас пропуски, все они были удалены. Так как количество отзывов на игру это ключевой показатель, на котором будет основана в итоге наша метрика успеха игры, мы удаляем все игры у которых нет информации об общем числе отзывов. Далее очень важный аспект, это дубликаты.

Подавать в модели мы будем исключительно 3 первых метки игры, а количество отзывов это ключевой признак по которому мы будем в итоге вычислять метрику успеха игры, поэтому все те игры, у которых совпадают первые три метки и количество отзывов, это для нас дубликаты, все они были удалены, можно их увидеть тут на первой картинке. Однако проблема в том, что те игры у которых все 3 первые метки совпадают, но количество отзывов не совпадает, по сути получаются тоже дубликаты, которые будут указывать на различные целевые значения. Поэтому было решено поступить так, все дубликаты удалить, и назначить той уникальной комбинации меток что осталась значение количества отзывов среднее от всех тез что были у удаленных дубликатов, + значение оставшейся уникальной комбинации. В итоге у нас

получился датасет с исключительно уникальными комбинациями первых 3-х меток. Что в привело в итоге, к удалению существенного количества записей.

Единственный способ как определить успешность игры на платформе steam, это посмотреть на количество отзывов на игру от пользователей, так как общее количество владельцев игры можно определить умножив общее количество отзывов примерно на 50. Поэтому, в качестве метрики успеха было решено использовать среднее количество отзывов в месяц которые игра получила, то есть общее количество отзывов деленное на общее количество дней с момента релиза и умноженное на 30. И, как видно на рисунке справа, построив диаграмму ящик с усами этой новой переменной, количество отзывов в месяц, у нее слишком большой диапазон значений и большое количество выбросов. В итоге было решено ее логарифмировать, и анализировать дальше график ее логарифма, гистограмму ее логарифма мы видим на рисунке слева, можно сказать что это распределение Пуассона.

Далее на ящиках с усами внизу можно увидеть эффект применения различных алгоритмов избавления от выбросов на логарифм нашей переменной отзывы за 30 дней. Верхний это просто логарифм нашей пременной без изменений, ниже это метод избавления от выбросов основанный на zscore, где режется все что имеет модуль zscore выше 3-х, и последний это квантильный метод, где режется грубо говоря все за пределами 1.5 умноженный на межквартильный диапазон. В итоге от выбросов было решено не избавляться, так как в этом случае мы удалим слишком много полезных данных.

Затем было решено разбить переменную на 10 квантилей, чтобы посмотреть как она по ним распределена. Мы видим из рисунка, что большая часть высоких значений она в последнем десятом квантиле. Так же опять были протестированы оба метода избавления от выбросов. И в общем итоге было решено их оставить. Далее так же был определен бинарный класс is_succesful, который будет определять успешна игра или нет, и было решено все игры которые входят в последние два квантиля чтобы они приняли значения этого класса 1, а остальные 0. потому что в общем итоге, после проб и попыток было решено делать бинарную классификацию, и это то как мы определили два класса на которые мы будем все игры делить.

Далее мы делаем итоговую предобработку данных для подчаи в модели, для этого мы создаем датафрейм, с первыми 3-мя метками игр, и теперь мы должны эти строковые значения закодировать и превратить в численные при помощи метода TagetEncoder библиотеки category_encoders. Этот метод создает численные значения для категориальных данных на основе целевой переменной, которая у нас является бинарным классом. Затем мы делим выборку на тренировочную и тествую с сохранением баланса классов, путем добавления параметра startify=True к методы train_test_split.

Далее у нас был выбор методов машинного обучения. На самом деле весь этот процесс выбора и исследования и смены парадигм что мы собираемся делать, происходил не строго по этапам а в целом, и довольно много раз, изначально я вообще хотел делать регрессию, эксперементировал с кучей разных способ преобразования наших категориальных тагов в численные, и так

далее. Но в результате пришлось стабилизироваться на том чтобы делать бинарную классификацию.

Итак согласно нашей задаче а это бинарная классификация с небольшим несбалансированным датасетом, в итоге я нашел наиболее подходящими вот эти методы, на экране, это Логистическая регрессия так как она, хорошо подходит для бинарной классификации, имеет неплохую точность на маленьких датасетах, может быть использована с несбалансированными датасетами, однако недостаток, что она чувствительна к выбросам. Деревья решений просты в понимании и интерпретации и визуализации, и не так тербовательно к предобработке данных, однако к сожалению они не так устойчивы к переобучению и дисбалансу классов. Гауссовский байесовский классификатор, опять же потому что он простой но эффективный для задача из реальной жизни, и не требователен к обьему данных, однако он плох в оценке вероятностей классов.

Метод К-ближайших соседей, потмоу что опять же он прост но эффективен, и хорош в работе с небольшими датасетами, однако он не может оценить вероятность классов. Метод опорных векторов, потому что он может показать хорошие результаты на несбалансированных классах, имеет высокую гибкость в настройке, однако он чувствителен к шуму в данных. Ну и случайный лес, главными достоинствами которого для нас являются устойчивость к выбросам, и меньше слконности к переобучению чем скажем в дервьев решений. Однако самый главный его минус это затраты времени и ресурсов на обучение, то есть долгий процесс обучения.

Для оценки результатов работы алгоритмов были использованы не много н е обычные метрики, не была использована метрика accuracy, так как она по сути не работает с задачами сильно насбалансированных классов, это f1_score, котоырй влется средним гармоническим между полнотой и точностью, и их относительный вклад этих обоих метрик он одинаков в этот показатель. Это точност — количество правильно распознаных позитивных классов к неправильно распознанным, и полнота общее количество распознанных позитивных классов. Ну и последний это ко’ффициент корреляции мэтьюса, который является корреляцией между предсказанными значениями и правдой.