|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| Институт кибербезопасности и цифровых технологий |
| Кафедра КБ-2 «Прикладные информационные технологии» |

**ПРАКТИЧЕСКАЯ РАБОТА №1 по дисциплине**

**«Методы анализа данных»**

***(наименование дисциплины)***

**Тема практической работы   
«Предсказания победителя в онлайн-игре»**

Студент группы: БИСО-01-20 Давыдова Д.А.

*(шифр учебной группы) (Фамилия И.О)*

Преподаватель: Лесько С.А.

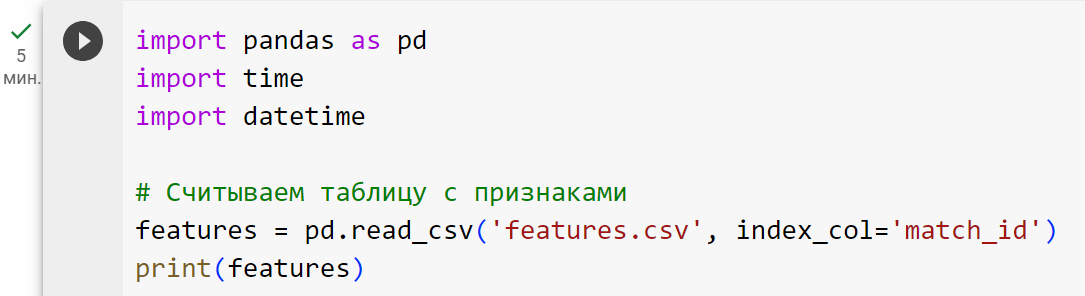
*(Фамилия И.О)*

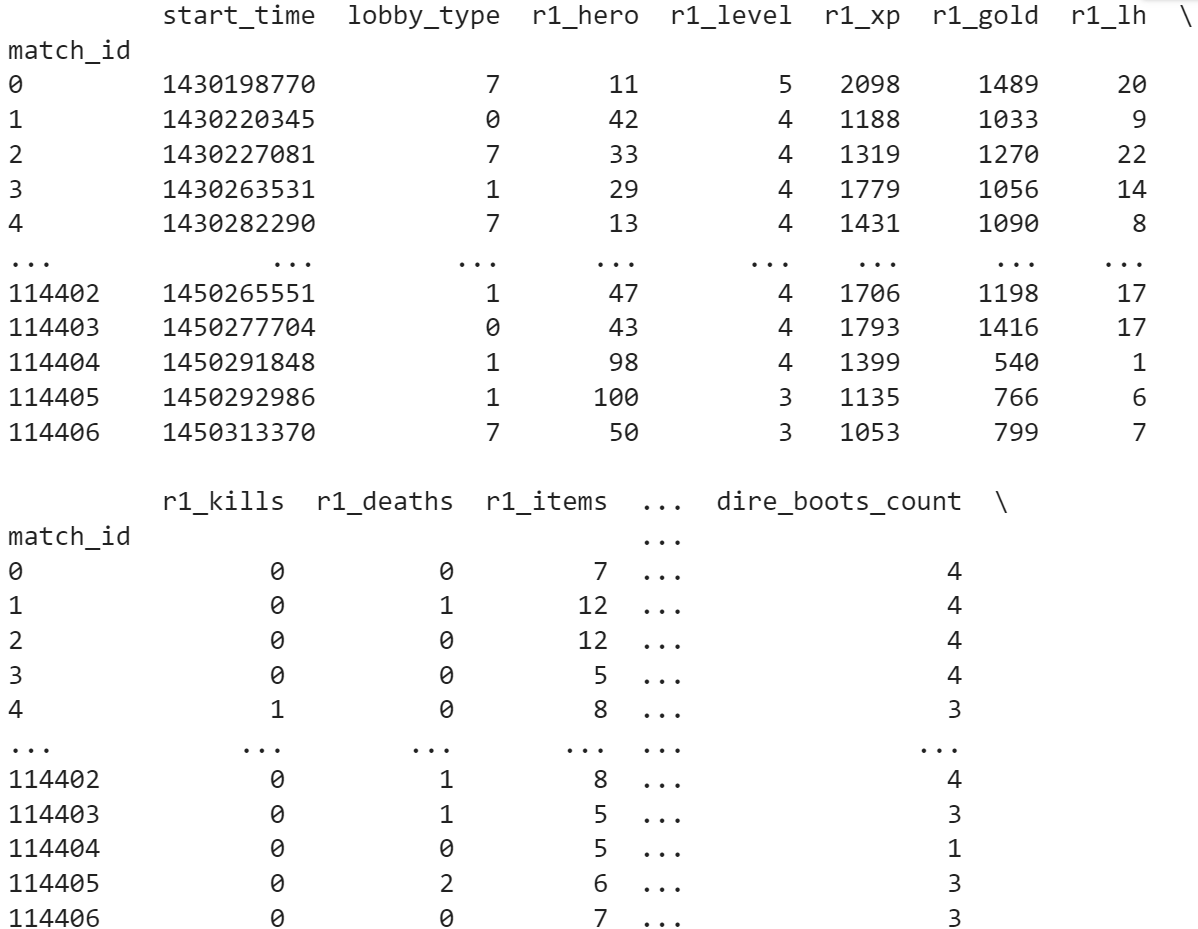
Москва, 2023

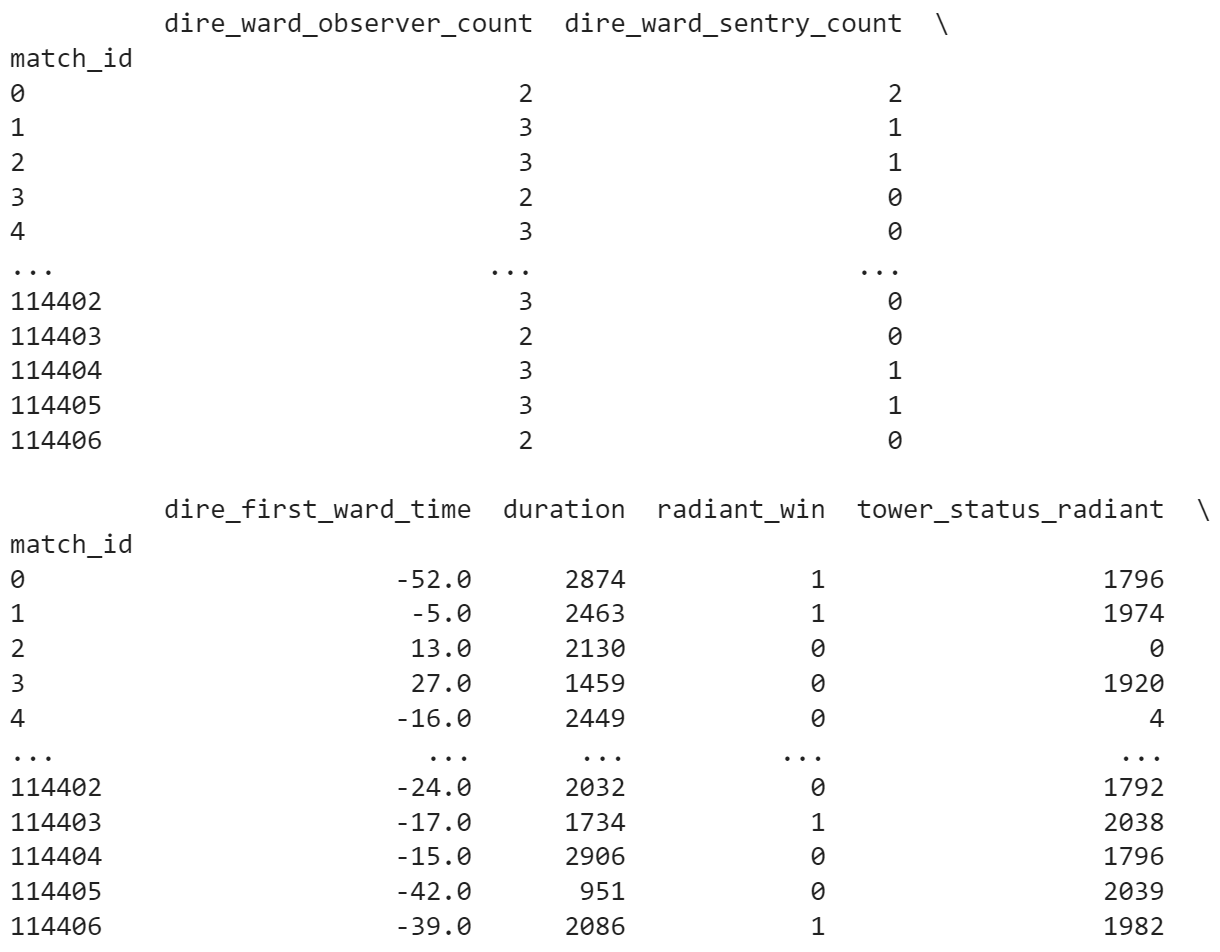
ПОДХОД 1: ГРАДИЕНТНЫЙ БУСТИНГ "В ЛОБ"

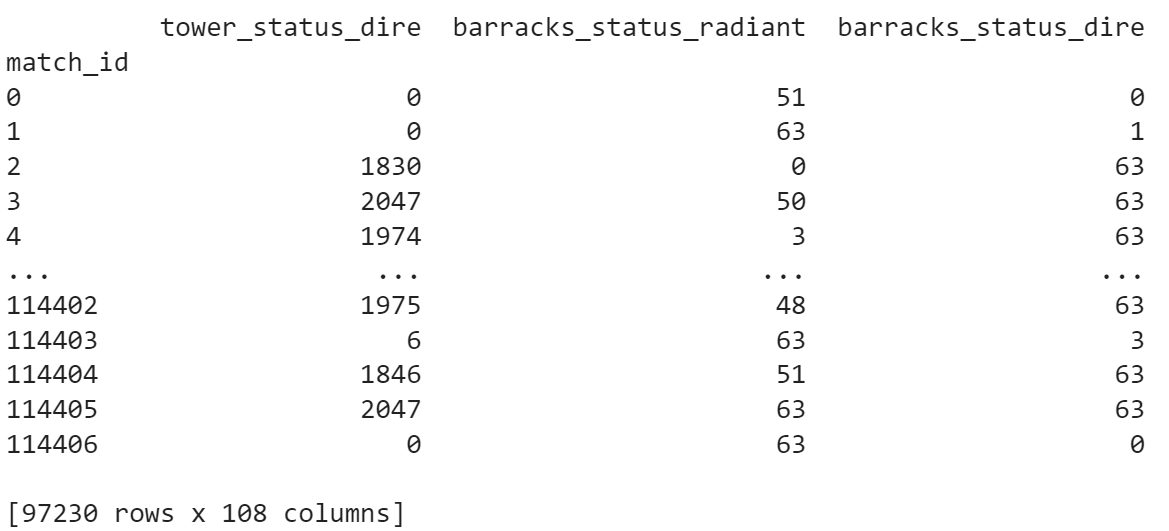
Один из самых универсальных алгоритмов, изученных в нашем курсе, является градиентный бустинг. Он не очень требователен к данным, восстанавливает нелинейные зависимости, и хорошо работает на многих наборах данных, что и обуславливает его популярность. Вполне разумной мыслью будет попробовать именно его в первую очередь.

1. Считайте таблицу с признаками из файла features.csv

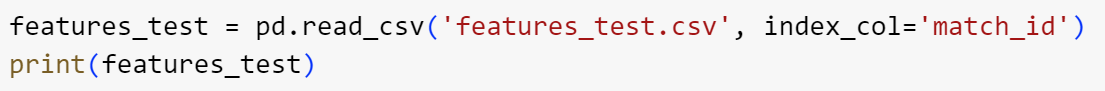


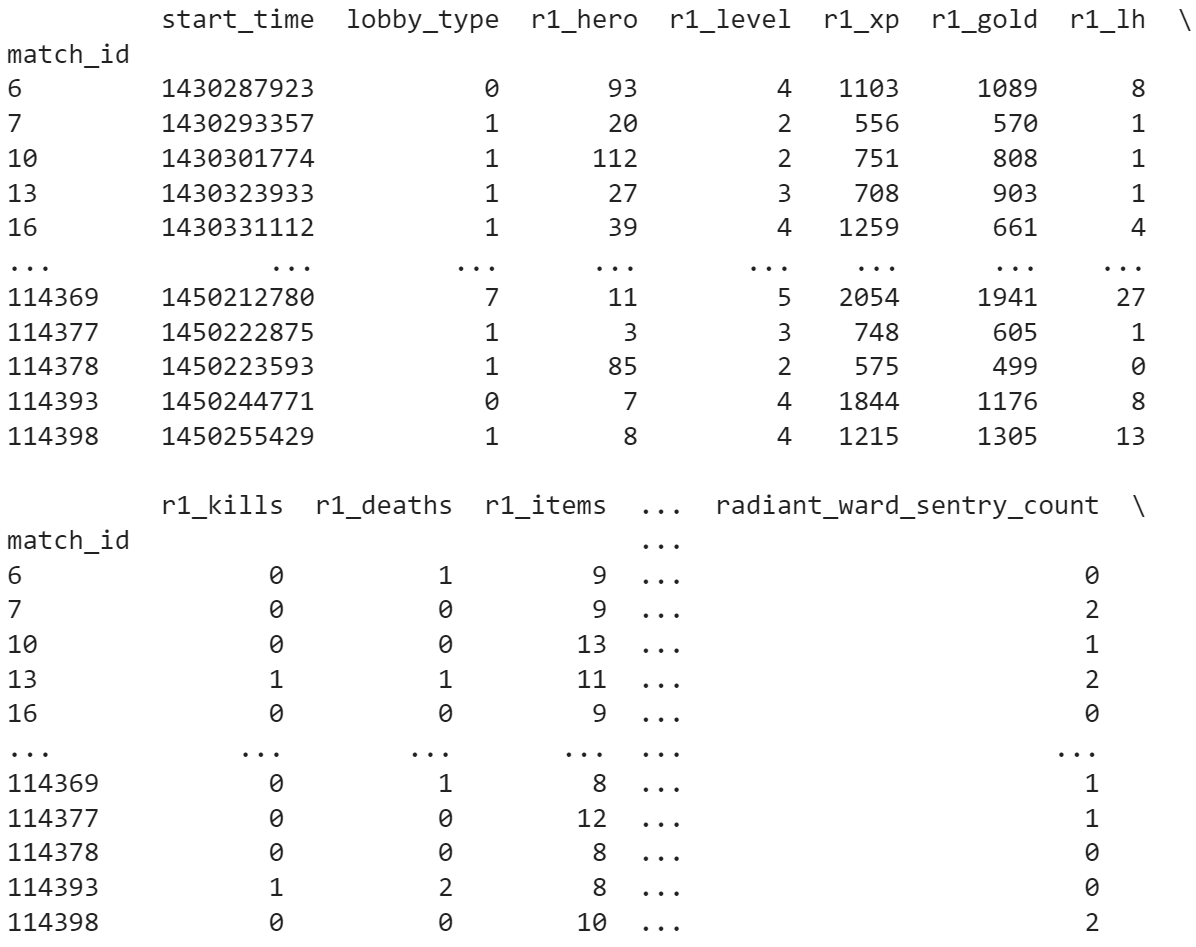


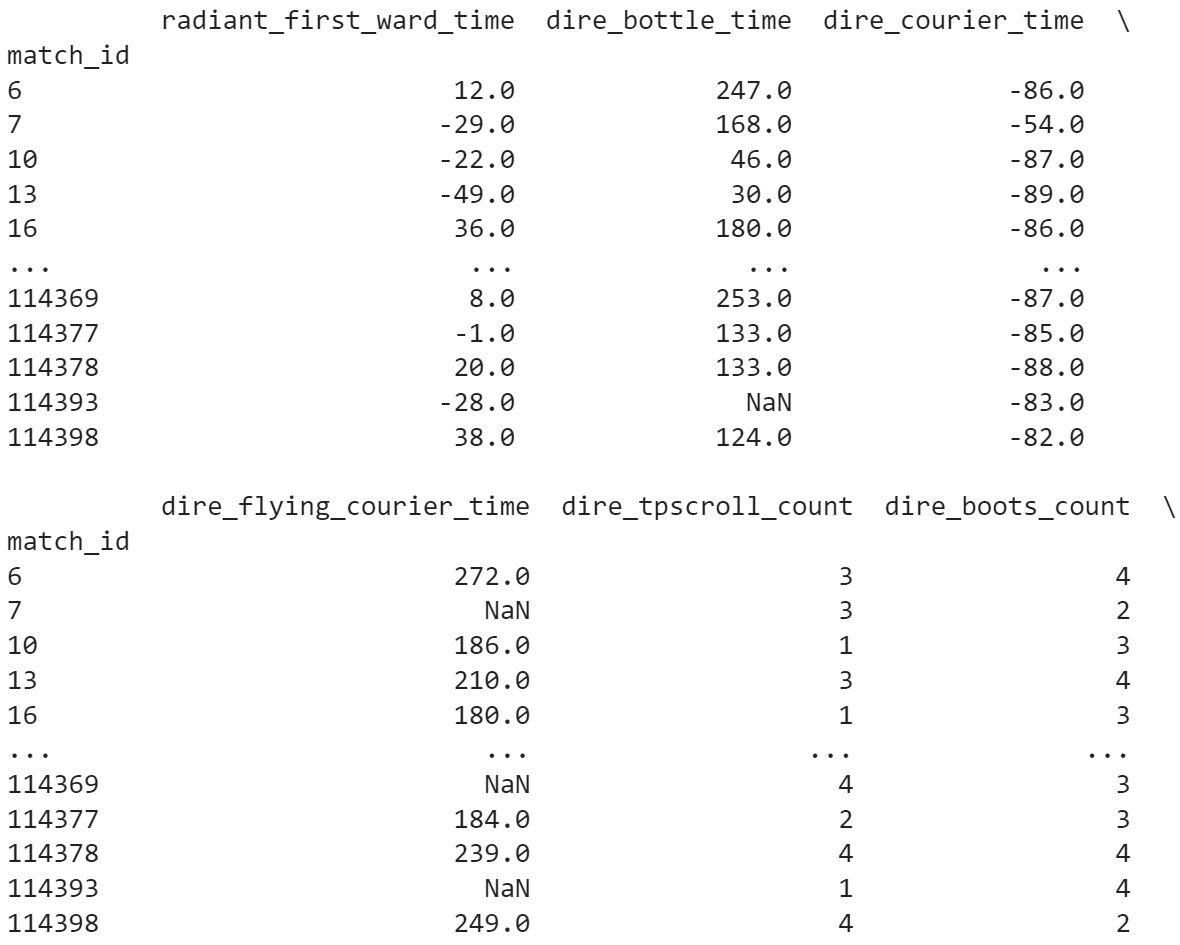


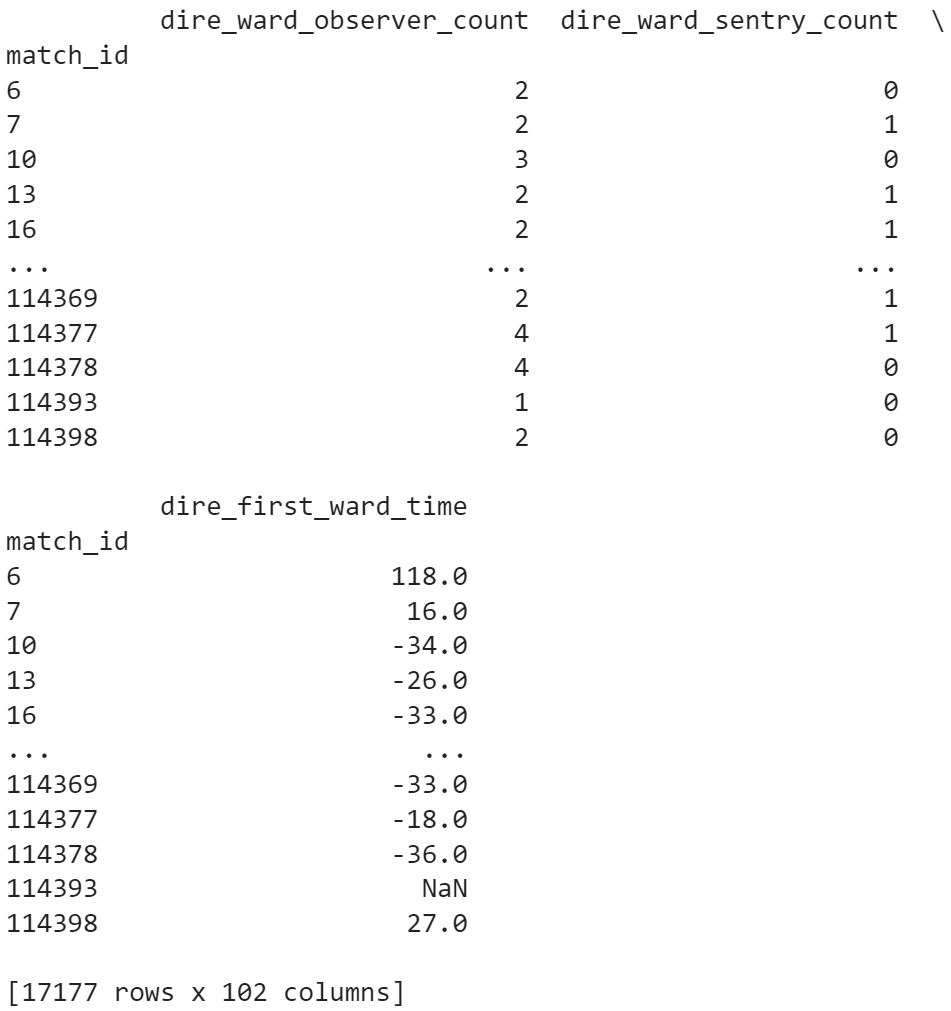


Считываем файл features\_test с тестовой выборкой

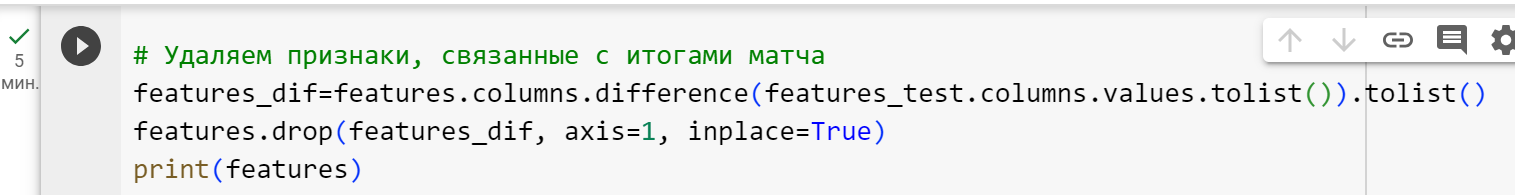


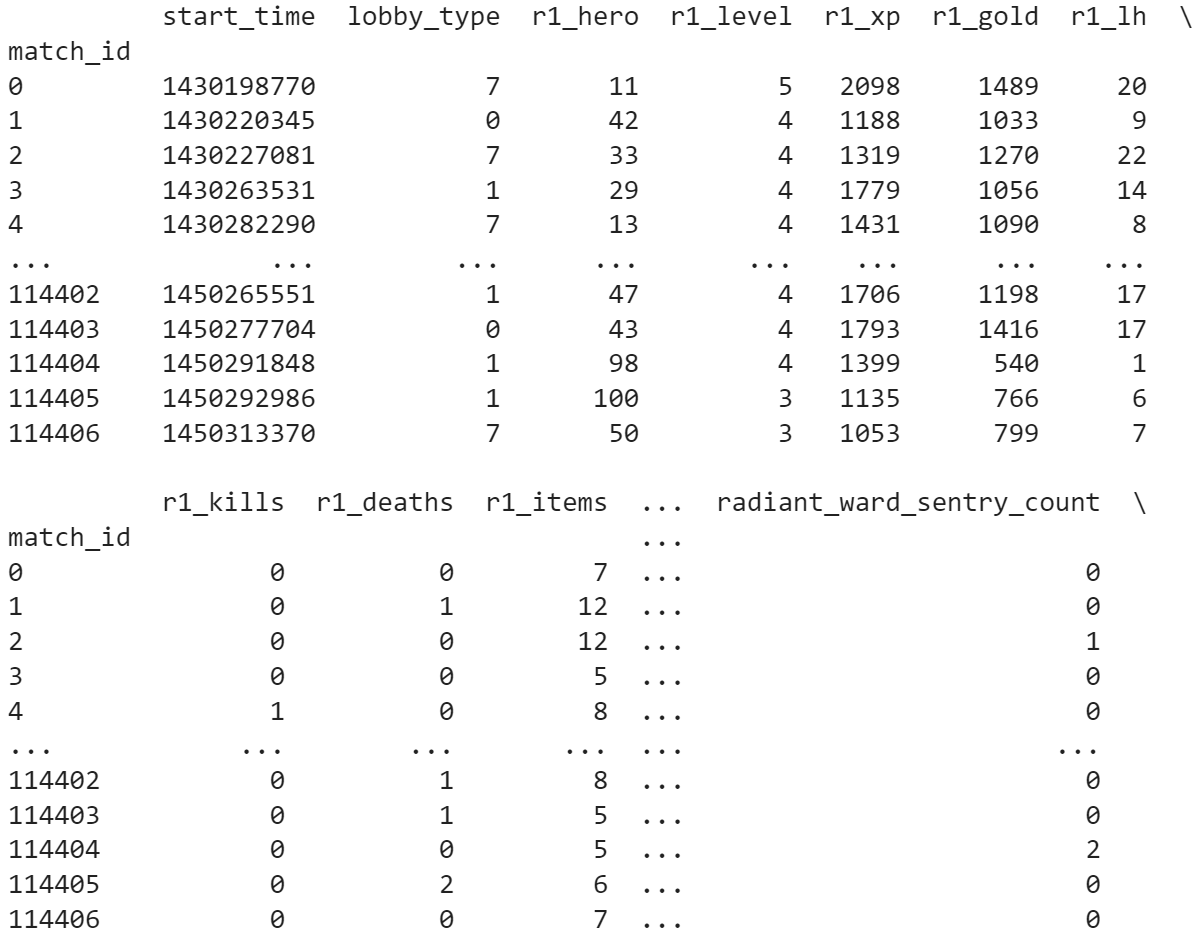


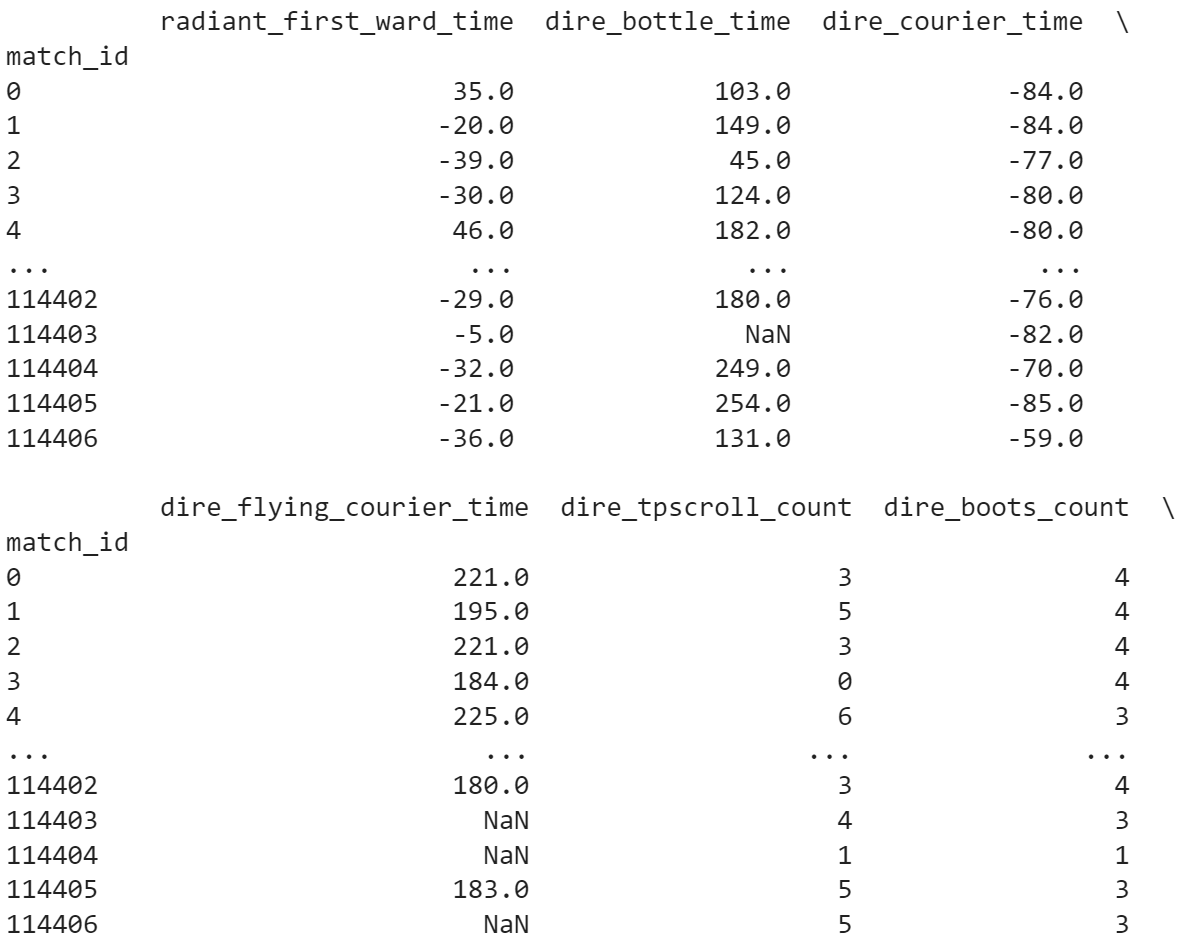


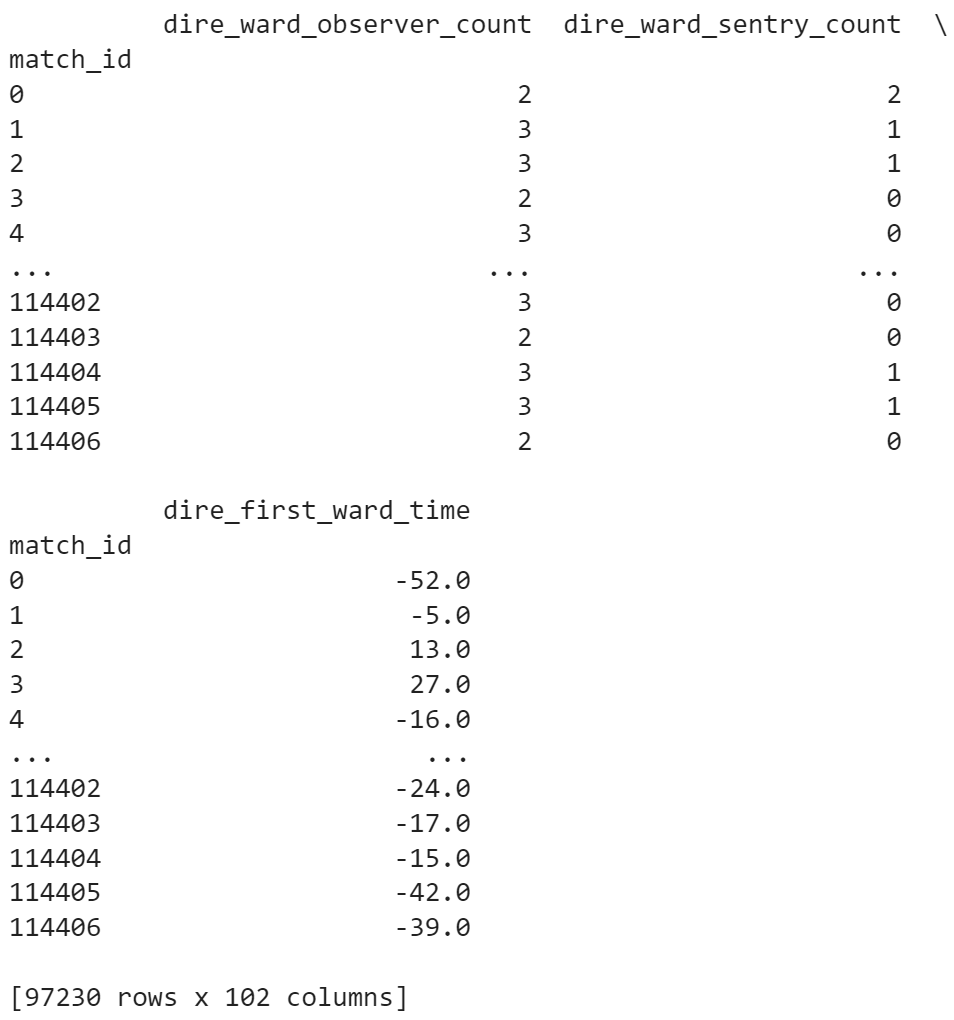


Удалите признаки, связанные с итогами матча (они помечены в описании данных как отсутствующие в тестовой выборке).

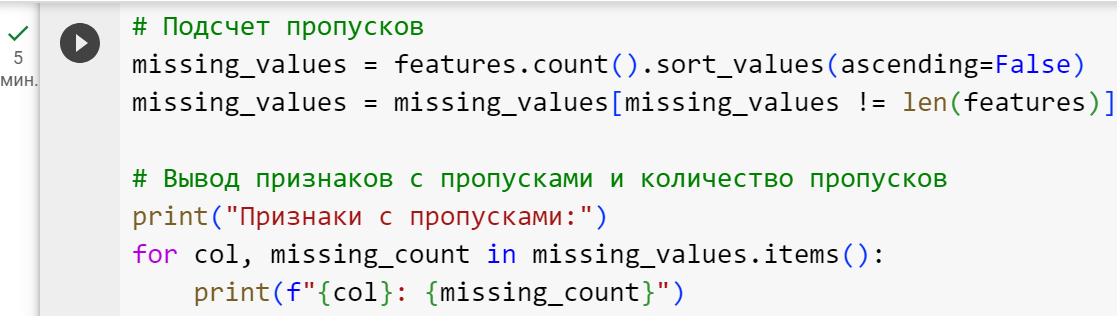


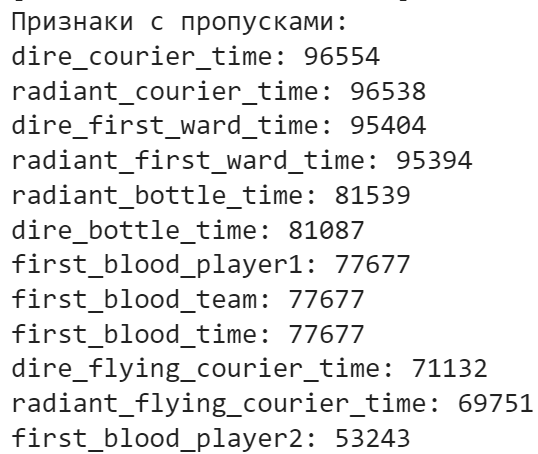




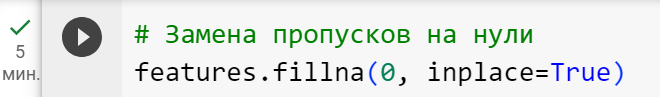


1. Проверьте выборку на наличие пропусков с помощью функции count(), которая для каждого столбца показывает число заполненных значений.





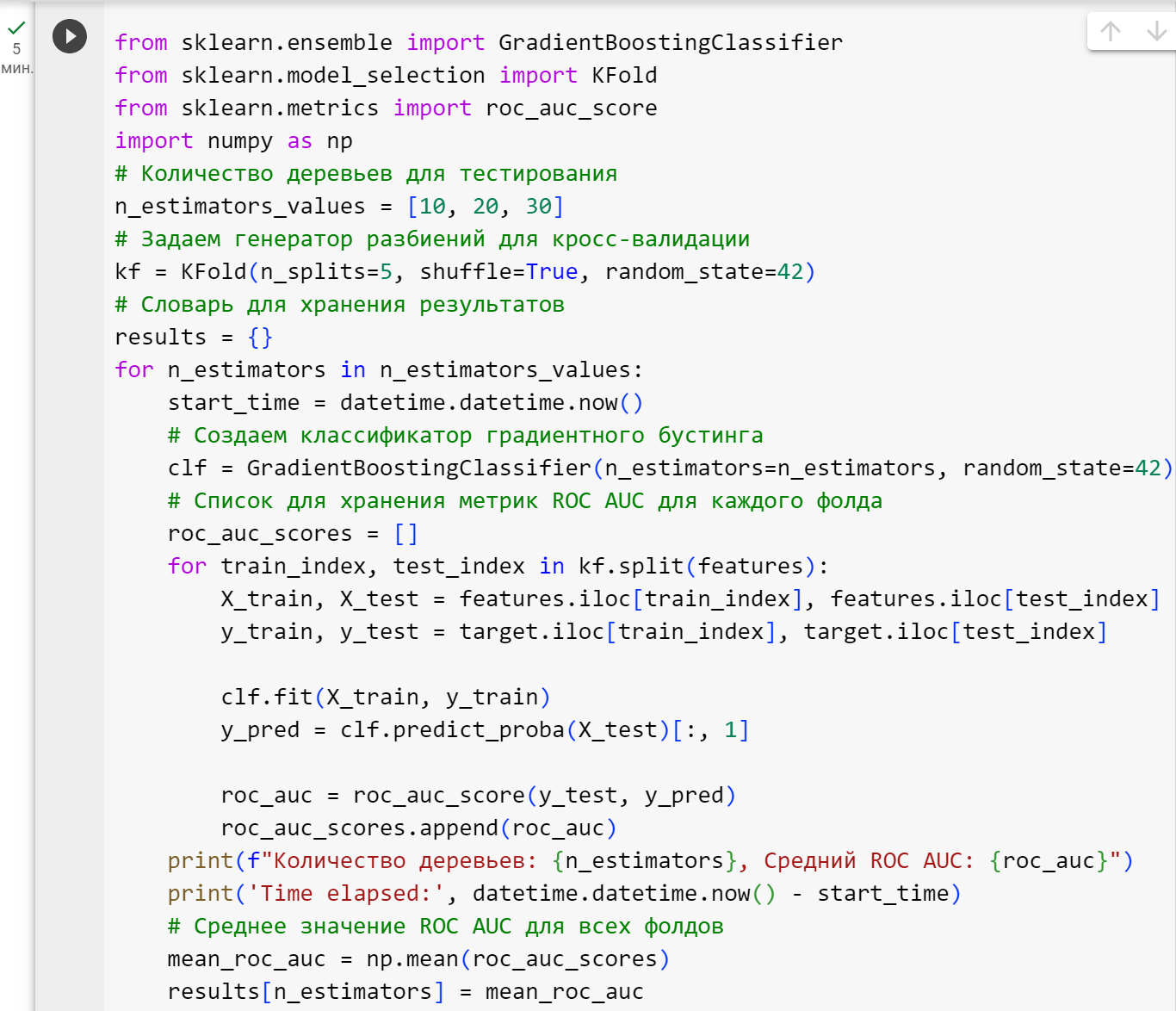
1. Замените пропуски на нули с помощью функции fillna().

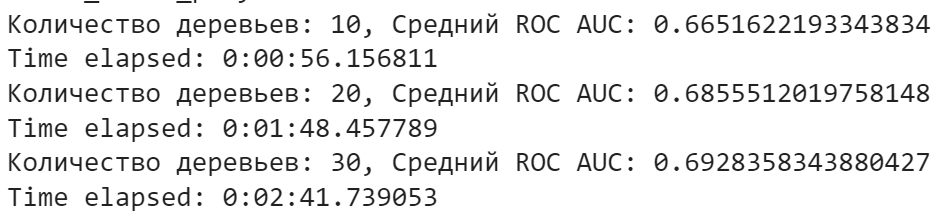


1. Какой столбец содержит целевую переменную? Запишите его название.

Целевая переменная 'radiant\_win' – 1, если победила команда Radiant, иначе – 0.

1. Забудем, что в выборке есть категориальные признаки, и попробуем обучить градиентный бустинг над деревьями на имеющейся матрице "объекты-признаки".





1) Какие признаки имеют пропуски среди своих значений? Что могут означать пропуски в этих признаках (ответьте на этот вопрос для двух любых признаков)?

Пропуски присутствуют у признаков: first\_blood\_time, first\_blood\_team, first\_blood\_player1,first\_blood\_player2, radiant\_bottle\_time, radiant\_courier\_time, radiant\_flying\_courier\_time, radiant\_first\_ward\_time, dire\_bottle\_time:dire\_courier\_time, dire\_flying\_courier\_time, dire\_first\_ward\_time.

Пропуски означают, что эти события не произошли в первые 5 минут игры.

2) Как называется столбец, содержащий целевую переменную?

Столбец, содержащий целевую переменную – 'radiant\_win'

3) Как долго проводилась кросс-валидация для градиентного бустинга с 30 деревьями? Инструкцию по измерению времени можно найти ниже по тексту. Какое качество при этом получилось? Напомним, что в данном задании мы используем метрику качества AUC-ROC.

1 мин 52 сек, качество – 0,69

4) Имеет ли смысл использовать больше 30 деревьев в градиентном бустинге? Что бы вы предложили делать, чтобы ускорить его обучение при увеличении количества деревьев?

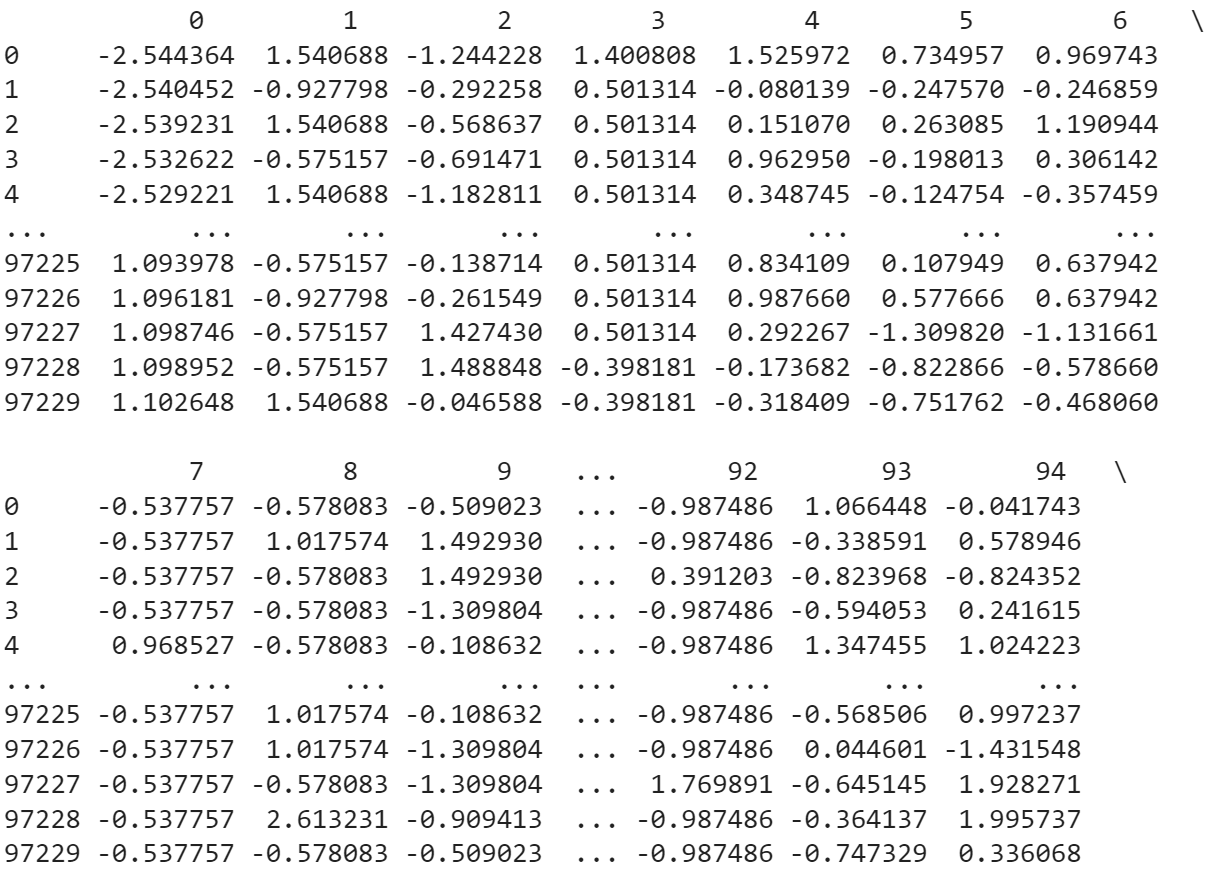
Увеличивать количество деревьев больше 30-ти стоит при наличии достаточного времени или вычислительных мощностей. В случае 99 деревьев качество увеличивается меньше, чем на 2%, а время вычислений становится больше примерно в 3 раза. Для ускорения процесса при увеличении числа деревьев следует уменьшать глубину деревьев и/или уменьшать вклад каждого дерева. Так же, Градиентный бустинг можно обучать параллельно на нескольких ядрах процессора с помощью параметра **n\_jobs**. Установка **n\_jobs** на большее значение позволит использовать многозадачность для ускорения обучения.

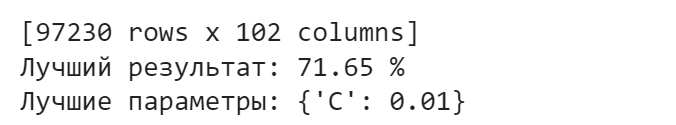
ПОДХОД 2: ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ

Линейные методы работают гораздо быстрее композиций деревьев, поэтому кажется разумным воспользоваться именно ими для ускорения анализа данных. Одним из наиболее распространенных методов для классификации является логистическая регрессия.

1. Оцените качество логистической регрессии (sklearn.linear\_model.LogisticRegression с L2-регуляризацией) с помощью кросс-валидации по той же схеме, которая использовалась для градиентного бустинга. Подберите при этом лучший параметр регуляризации (C). Какое наилучшее качество у вас получилось? Как оно соотносится с качеством градиентного бустинга? Чем вы можете объяснить эту разницу? Быстрее ли работает логистическая регрессия по сравнению с градиентным бустингом?

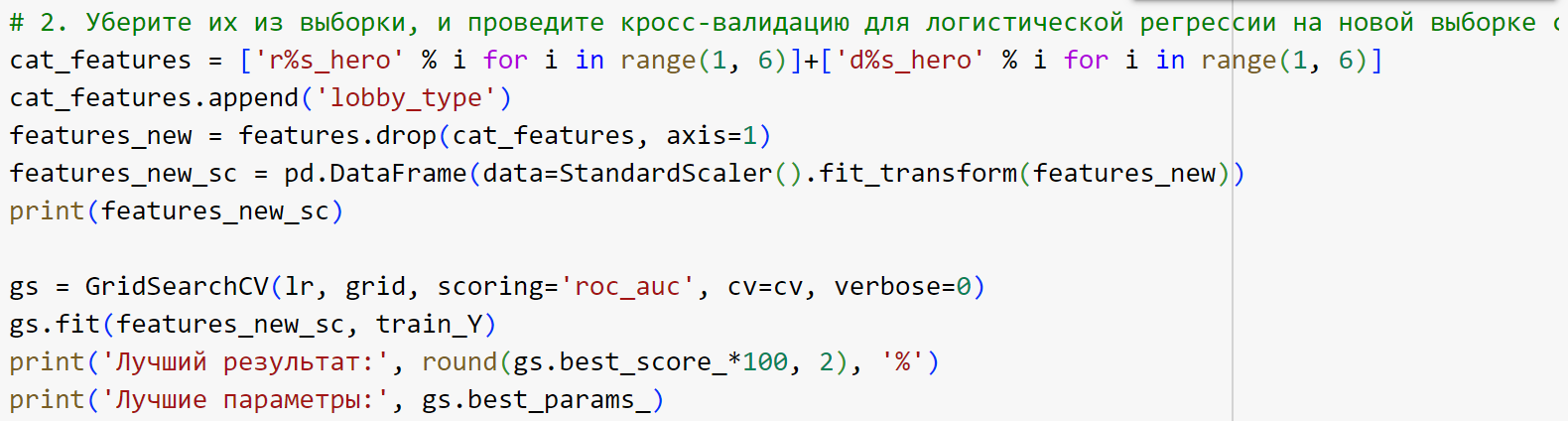


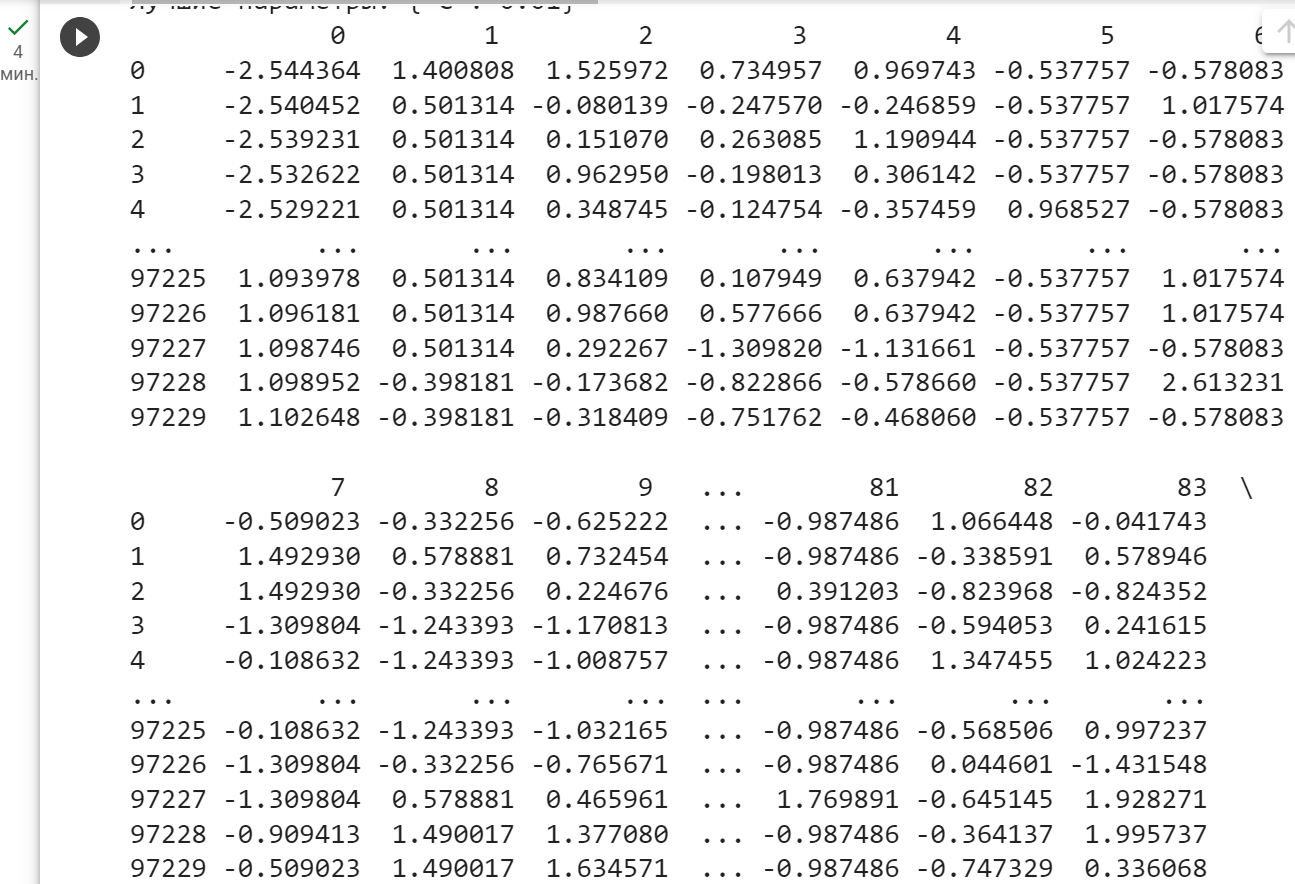


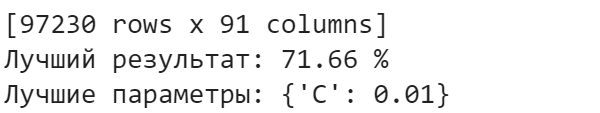


Для логистической регрессии был выбран более подходящий гиперпараметр, что позволило ей достичь лучшего качества на данном наборе данных. Логистическая регрессия работает быстрее, чем градиентный бустинг, потому что она является линейной моделью и не требует построения большого количества деревьев.

1. Среди признаков в выборке есть категориальные, которые мы использовали как числовые, что вряд ли является хорошей идеей. Категориальных признаков в этой задаче одиннадцать: lobby\_type и r1\_hero, r2\_hero, ..., r5\_hero, d1\_hero, d2\_hero, ..., d5\_hero. Уберите их из выборки, и проведите кросс-валидацию для логистической регрессии на новой выборке с подбором лучшего параметра регуляризации. Изменилось ли качество? Чем вы можете это объяснить?



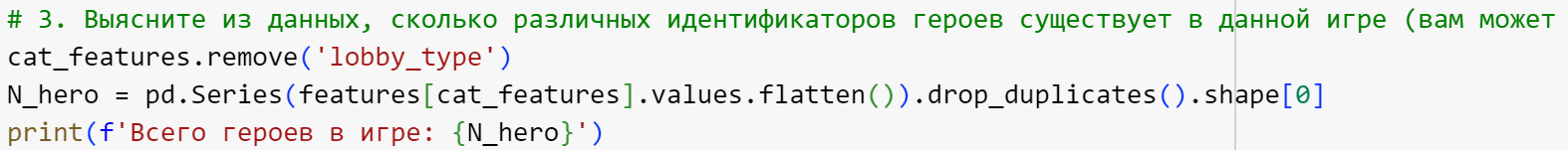




Убирая категориальные признаки, мы теряем информацию о конкретных героях, но при этом можем получить лучшие результаты, так как числовые представления этих категорий не содержат смысла. Номер героя не имеет никакого отношения к их характеристикам.

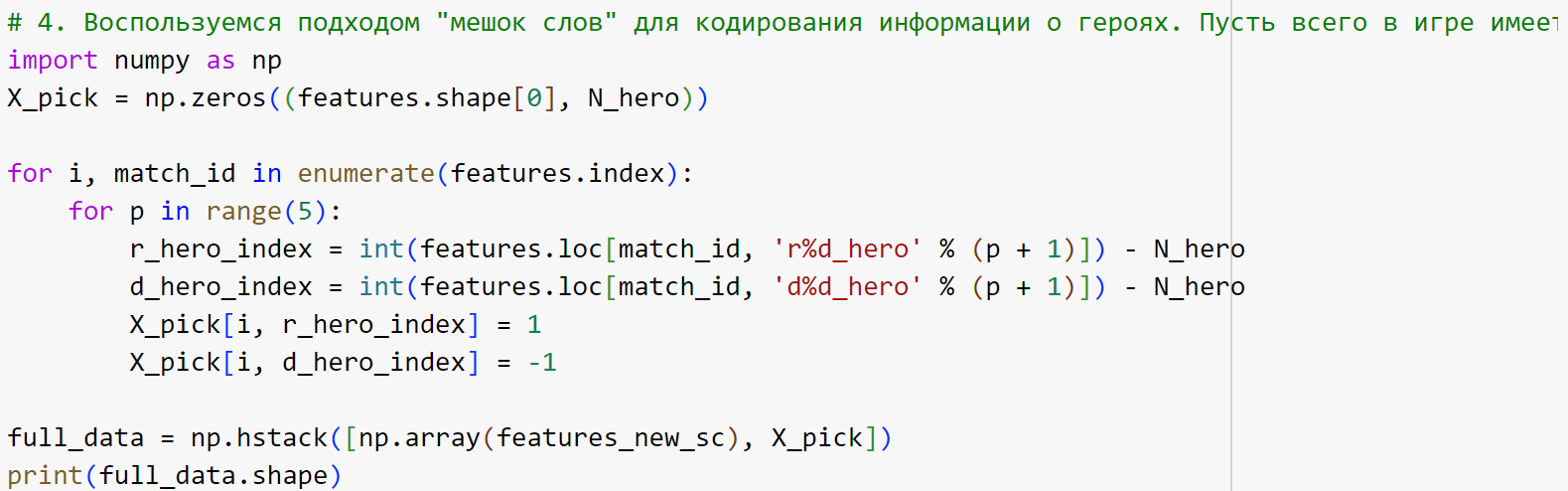
Это может улучшить работу модели, потому что логистическая регрессия будет использовать более информативные признаки для принятия решения. Результат получился чуть лучше, чем в предыдущем случае

1. На предыдущем шаге мы исключили из выборки признаки rM\_hero и dM\_hero, которые показывают, какие именно герои играли за каждую команду. Это важные признаки — герои имеют разные характеристики, и некоторые из них выигрывают чаще, чем другие. Выясните из данных, сколько различных идентификаторов героев существует в данной игре (вам может пригодиться фукнция unique или value\_counts).



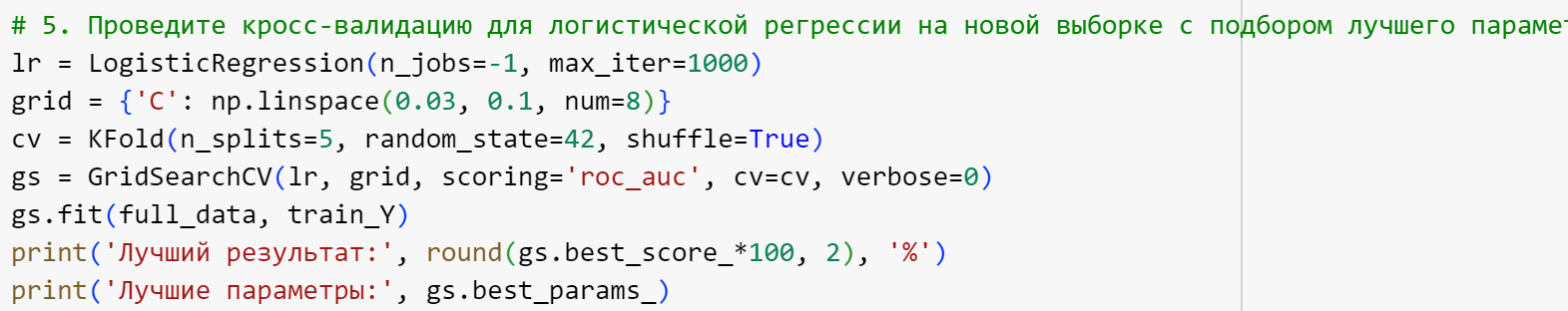


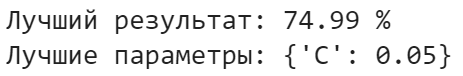
1. Воспользуемся подходом "мешок слов" для кодирования информации о героях. Пусть всего в игре имеет N различных героев. Сформируем N признаков, при этом i-й будет равен нулю, если i-й герой не участвовал в матче; единице, если i-й герой играл за команду Radiant; минус единице, если i-й герой играл за команду Dire. Добавьте полученные признаки к числовым, которые вы использовали во втором пункте данного этапа.





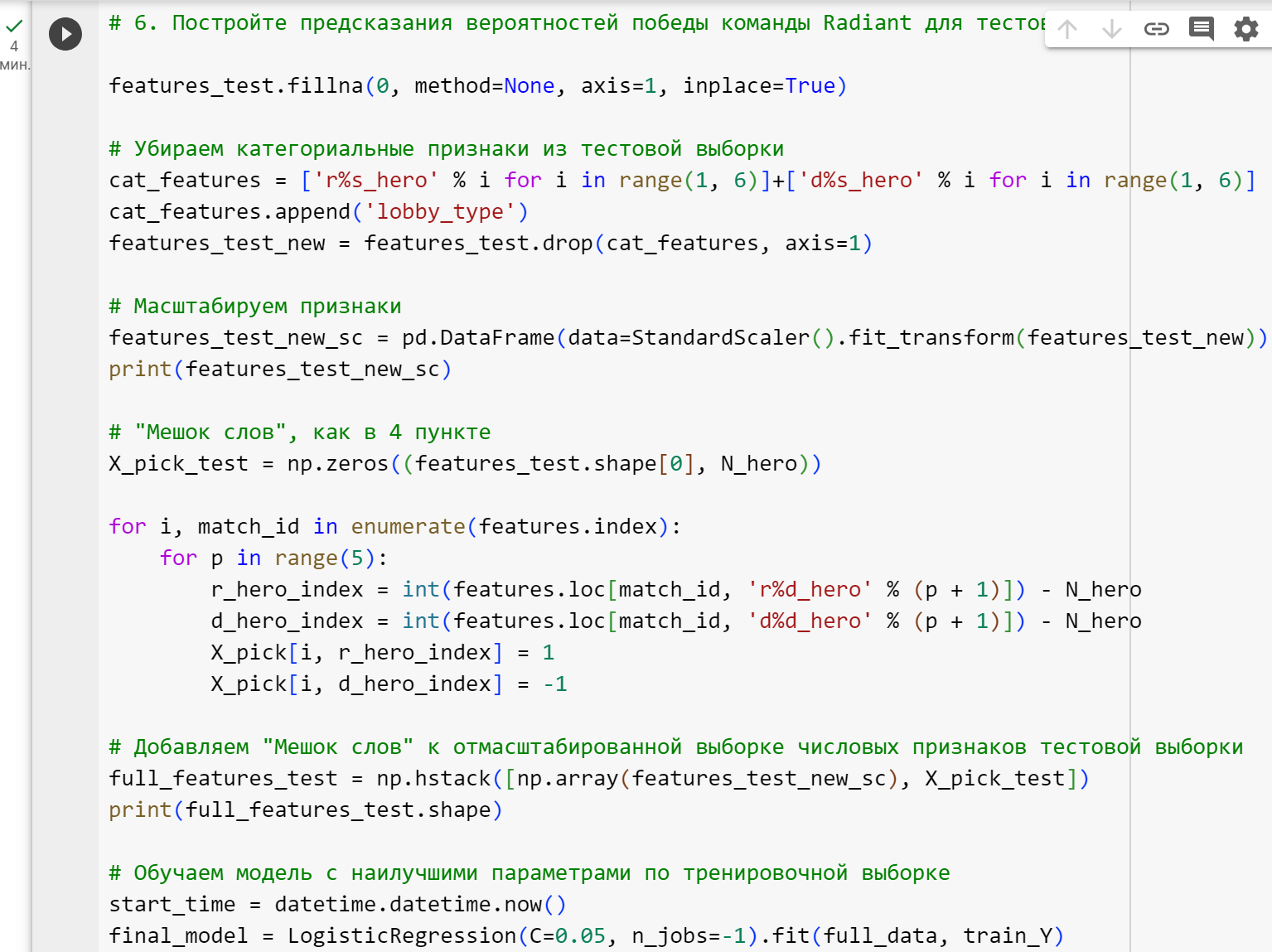
1. Проведите кросс-валидацию для логистической регрессии на новой выборке с подбором лучшего параметра регуляризации. Какое получилось качество? Улучшилось ли оно? Чем вы можете это объяснить?

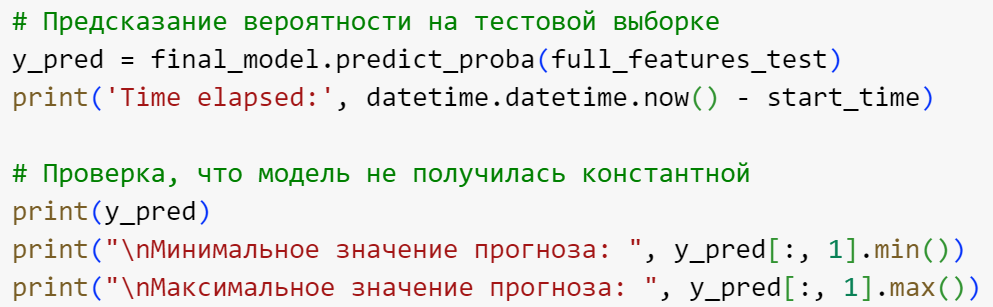


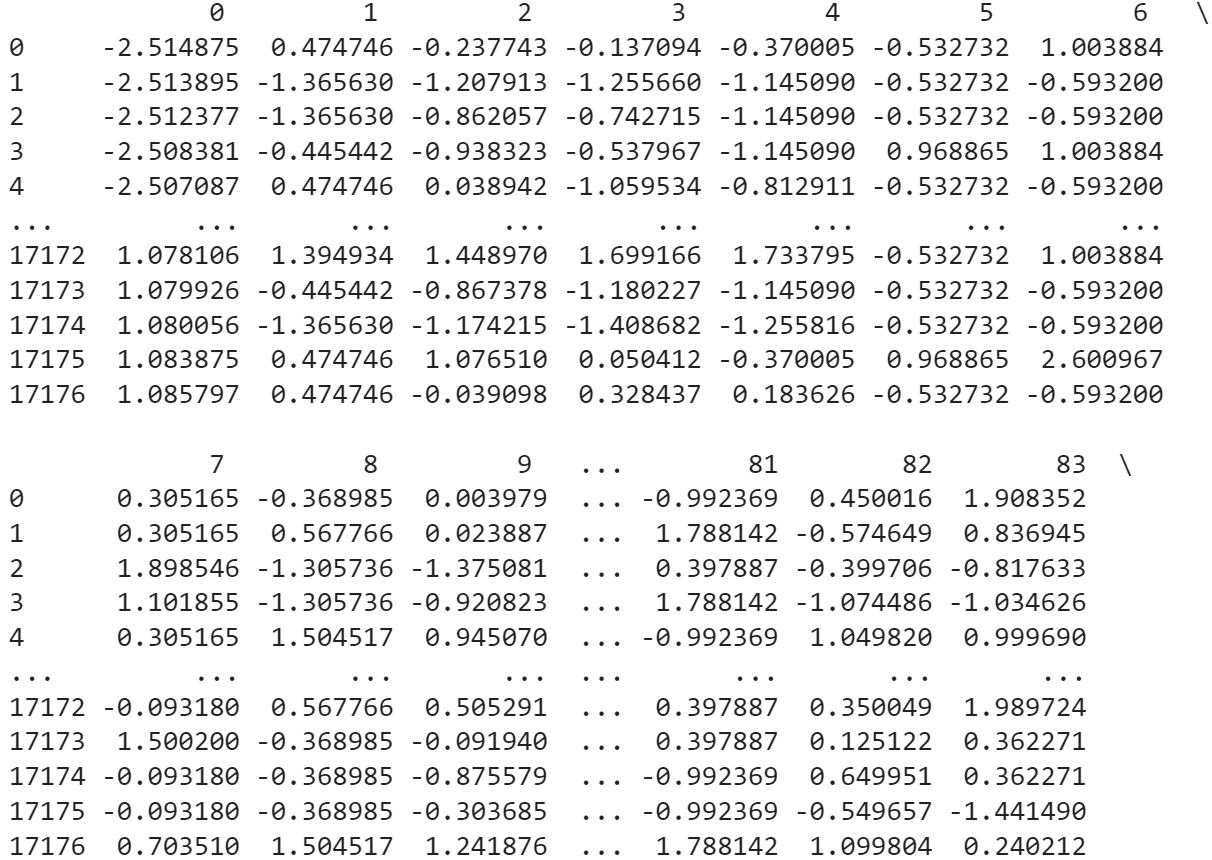


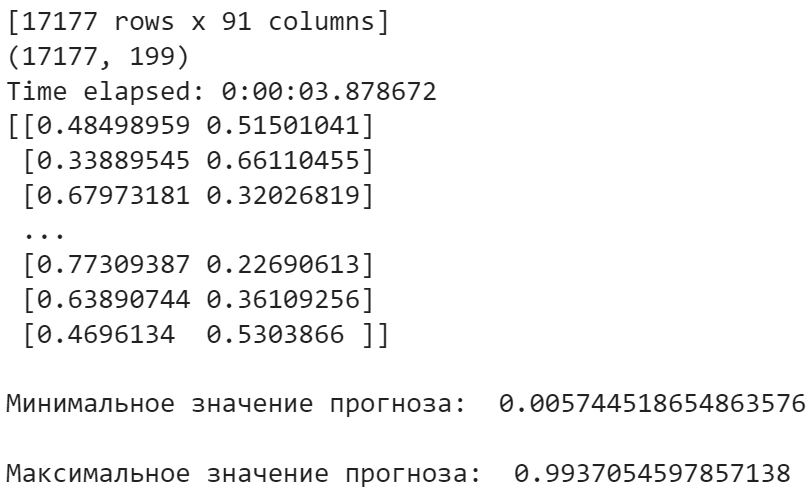
Качество улучшилось, это связано с тем, что удаление категориальных признаков и масштабирование данных помогли модели лучше обучиться.

1. Постройте предсказания вероятностей победы команды Radiant для тестовой выборки с помощью лучшей из изученных моделей (лучшей с точки зрения AUC-ROC на кросс-валидации). Убедитесь, что предсказанные вероятности адекватные — находятся на отрезке [0, 1], не совпадают между собой (т.е. что модель не получилась константной).









1) Какое качество получилось у логистической регрессии над всеми исходными признаками? Как оно соотносится с качеством градиентного бустинга? Чем вы можете объяснить эту разницу? Быстрее ли работает логистическая регрессия по сравнению с градиентным бустингом?

Логистическая регрессия, примененная к исходным признакам без дополнительной обработки, показала качество на кросс-валидации в 0.7165, что превосходит результат градиентного бустинга в 0.6928 . Эту разницу можно объяснить особенностями исходных данных.

Логистическая регрессия работает быстрее, чем градиентный бустинг, потому что она является линейной моделью и не требует построения большого количества деревьев.

2) Как влияет на качество логистической регрессии удаление категориальных признаков (укажите новое значение метрики качества)? Чем вы можете объяснить это изменение?

Убирая категориальные признаки, мы теряем информацию о конкретных героях, но при этом можем получить лучшие результаты, так как числовые представления этих категорий не содержат смысла. Номер героя не имеет никакого отношения к их характеристикам.

Это может улучшить работу модели, потому что логистическая регрессия будет использовать более информативные признаки для принятия решения. Полученный результат чуть лучше, чем в предыдущем случае - на 0,0001

3) Сколько различных идентификаторов героев существует в данной игре?

108 различных героев.

4) Какое получилось качество при добавлении "мешка слов" по героям? Улучшилось ли оно по сравнению с предыдущим вариантом? Чем вы можете это объяснить?

При добавлении "мешка слов" по героям, качество улучшается и составляет 0.7499. Это связано с переводом категориальных признаков в числовые на работу с которыми рассчитана линейная регрессия.

5) Какое минимальное и максимальное значение прогноза на тестовой выборке получилось у лучшего из алгоритмов?

Минимальное значение прогноза - 0.0057445186924018005

Максимальное значение прогноза - 0.993705459876053