[Dota 2](https://en.wikipedia.org/wiki/Dota_2) — многопользовательская компьютерная игра жанра [MOBA](https://en.wikipedia.org/wiki/Multiplayer_online_battle_arena). Игроки играют между собой матчи. В каждом матче, как правило, участвует 10 человек. Матчи формируются из живой очереди, с учётом уровня игры всех игроков. Перед началом игры игроки автоматически разделяются на две команды по пять человек. Одна команда играет за светлую сторону (The Radiant), другая — за тёмную (The Dire). Цель каждой команды — уничтожить главное здание базы противника, трон.

Вам нужно построить модель, которая по данным о первых пяти минутах матча будет предсказывать его исход — то есть определять команду-победителя.

Чтобы выполнить это задание, вам необходимо провести ряд исследований, сравнить несколько алгоритмов машинного обучения и проверить эффект от ряда манипуляций с признаками.

К заданию приложены следующие файлы:

* final-statement.ipynb и final-statement.html — постановка задачи, описание данных, инструкции по выполнению
* features.zip — архив с обучающей выборкой
* features\_test.zip — архив с тестовой выборкой
* data.zip — полный архив с сырыми данными и скриптом для извлечения признаков (этот архив понадобится вам только для участия в kaggle; для выполнения данного задания он не нужен)
* extract\_features.py — скрипт, извлекающий признаки из сырых данных

Будет удобно выполнять это задание в IPython/Jupyter Notebook — интерактивной среде, которая устанавливается, например, вместе с пакетом Anaconda. При этом мы не настаиваем на его использовании, и в качестве файла с кодом можно отправлять обычный py-файл, подготовленный в вашей любимой среде.

В сданном файле для каждого пункта задания должен быть код, с помощью которого получен ответ. Постарайтесь обозначить части кода, отвечающие на разные вопросы задания — так будет проще проверять вашу работу.

Вам необходимо провести описанные в документе final-statement.html (или final-statement.ipynb) два этапа исследования (для двух подходов к решению задачи), написать по результатам каждого этапа небольшой отчет (ниже указаны вопросы, ответы на которые должны содержаться в отчете), и предоставить для ревью данный отчет и код, с помощью которого вы выполнили задание.

Не забывайте, что в выборке есть признаки, которые "заглядывают в будущее" — они помечены в описании данных как отсутствующие в тестовой выборке. Их прямое использование в модели приведет к переобучению, поэтому не забудьте исключить их из выборки.

**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. В данном разделе предлагается попробовать градиентный бустинг для решения нашей задачи.

В отчете по данному этапу должны содержаться ответы на следующие вопросы:

1. Какие признаки имеют пропуски среди своих значений (приведите полный список имен этих признаков)? Что могут означать пропуски в этих признаках (ответьте на этот вопрос для двух любых признаков)?
2. Как называется столбец, содержащий целевую переменную?
3. Как долго проводилась кросс-валидация для градиентного бустинга с 30 деревьями? Инструкцию по измерению времени можно найти выше по тексту. Какое качество при этом получилось?
4. Имеет ли смысл использовать больше 30 деревьев в градиентном бустинге? Что можно сделать, чтобы ускорить его обучение при увеличении количества деревьев?

**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. В данном разделе предлгается применить ее к данным, а также попробовать различные манипуляции с признаками.

В отчете по данному этапу должны содержаться ответы на следующие вопросы:

1. Какое качество получилось у логистической регрессии над всеми исходными признаками? Как оно соотносится с качеством градиентного бустинга? Чем можно объяснить эту разницу? Быстрее ли работает логистическая регрессия по сравнению с градиентным бустингом?
2. Как влияет на качество логистической регрессии удаление категориальных признаков (укажите новое значение метрики качества)? Чем можно объяснить это изменение?
3. Сколько различных идентификаторов героев существует в данной игре?
4. Какое получилось качество при добавлении "мешка слов" по героям? Улучшилось ли оно по сравнению с предыдущим вариантом? Чем можно это объяснить?
5. Какое минимальное и максимальное значение прогноза на тестовой выборке получилось у лучшего из алгоритмов?

Следует понимать, что конкретные показатели метрик качества могут отличаться в зависимости от конкретных разбиений выборки, значений параметров и версий библиотек. Ответы следует проверять на адекватность — в правильную ли сторону изменяется показатель качества при том или ином изменении модели или выборки, корректные ли выводы делаются из соответствующих результатов.