Мини-отчет по прошедшему контесту

Паша Коваленко

Машинное обучение | ММП ВМК

Постановка задачи

Дается вся информация о первых 5 минутах матча в DOTA. Требуется предсказать, какая команда одержит победу.

Некоторые фичи были изначально выделены, в частности:

- Выбранные персонажи ($\times 10$)
- Уровень, достигнутый за первые 5 минут ($\times 10$)
- Общая стоимость всех купленных предметов ($\times 10$)
- Накопленный опыт ($\times 10$)
- Число убийств ($\times 10$)
- Число смертей ($\times 10$)
- И еще много непонятных признаков

На этих признаках я и обучался.

Метрика качества — AUC-ROC.

Герои

Согласно моим наблюдениям, в выборке были герои с номерами от 1 до 112, но некоторых из них ни разу не выбрали.

Выбранные герои — категориальные признаки, с которыми неудобно работать. Оказалось эффективным сделать для них мешок слов. То есть 10 категориальных признаков героев заменить на 112 признаков вида:

$$x_i = \begin{cases} 1, & \text{если i-й герой играет за radiant} \\ -1, & \text{если i-й герой играет за dire} \\ 0, & \text{если i-й герой не участвует в игре} \end{cases}$$

Остается единственный категориальный признак — `lobby_type`, к которому можно применить one-hot кодирование.

Среднее и разброс

Для каждого игрока известны 7 вещественных признаков — уровень, опыт, число убийств и прочие. Представляется логичным аггрегировать их внутри команды.

Для каждой из команд добавляется 7×2 новых признаков — для каждого из исходных признаков среднее и разброс.

Но исходные признаки тоже остаются на всякий случай.

Теперь ко всем получившимся признакам можно применить StandartScaler.

Пары героев

Говорят, что одни герои хорошо работают в тандеме, а другие хорошо действуют против определенных героев противника. Это из парное взаимодействие хочется учитывать.

Введем новые признаки следующего вида:

$$a_{ij} = \begin{cases} 1, & \text{если i-й и j-й герои играют за radiant} \\ -1, & \text{если i-й и j-й герои играют за dire} \\ 0, & \text{если один из этих героев не участвует в игре} \end{cases}$$

$$b_{ij} = \begin{cases} 1, & \text{если i-й герой играет за radiant, а j-й — за dire} \\ -1, & \text{если i-й герой играет за dire, а j-й — за radiant} \\ 0, & \text{если один из этих героев не участвует в игре} \end{cases}$$

Многовато

Получается 112×111 новых признаков. В сумме со всеми предыдущими получается порядка 12 000 признаков. В выборке 100 000 объектов. Матрица огромная, но разреженная.

Чтоб сэкономить память, можно информацию о парах героев хранить в разреженном виде (например, `scipy.sparse.csr_matrix` — Compressed Sparse Row matrix). Логистическая регрессия из sklearn умеет работать с таким типом данных напрямую, не конвертируя в `numpy.array`.

Пора что-нибудь обучить

Лучший результат на кросс-валидации показала логистическая регрессия с l2-регуляризацией, обученная на всех фичах (включая парные взаимодействия).

Итоговый ответ является взвешенной комбинацией ответов 5 алгоритмов:

- Логистическая регрессия с l2-регуляризацией, обученная на всех фичах
- Логистическая регрессия с l1-регуляризацией, обученная на всех фичах
- Логистическая регрессия с l2-регуляризацией, обученная на фичах без парных взаимодействий
- Логистическая регрессия с l1-регуляризацией, обученная на фичах без парных взаимодействий
- Градиентный бустинг, обученный на фичах без парных взаимодействий

Спасибо за внимание!