FlightRank 2025: Aeroclub RecSys Cup

- Team Name: Mikhail Golubchik
- Private Leaderboard Score: 0.530796.
- Private Leaderboard Place: 8

- Name: Голубчик Михаил Игоревич
- Location: Россия, г. Краснодар
- Email: mikhail.golubchik@gmail.com

Немного о себе: По образованию я программист. Долгое время работал системным администратором в основном в банках. А также создавал и поддерживал свой сайт, касающийся банковской тематики. У меня уже был опыт соревнований kaggle. В частности, по теме предсказания производства и потребления электричества, где так же использовались бустинги и табличные данные, и этот опыт помог мне в данном соревновании.

Что привлекло. Данное соревнование привлекло меня упоминанием в группе телеграмм Kaggle Alchemists. И тем, что в нем некоторые из России решили поучаствовать. Включая одного грандмастера. Соревнование показалось мне интересным, тем более что было указано, что организаторы будут платить призовые в Россию. К тому времени я еще не знал что и само соревнование так же проводит компания из России.

Затраченное время. Как обычно, я потратил довольно много времени на соревнование. По несколько часов в день, за полтора - два месяца. С некоторым перерывом на одну неделю для решения другой задачи.

Описание решения задачи

Приступая к описанию самого решения, прежде всего, хотелось бы поблагодарить организаторов соревнования за интересную задачу и настоящие реальные данные.

Бейслайн. В качестве основы я взял решение от Kirill Khoruzhii (@ka1242): AeroClub RecSys 2025 - XGBoost Ranking Baseline (https://www.kaggle.com/code/ka1242/xgboost-ranker-with-polars). Хотелось бы поблагодарить его за хороший набор признаков. Для основы было выбрано именно это решение, так как оно имело лучший публичный скор на тот момент. И

решения с заметно более высоким публичным скором не появилось до конца соревнования.

Использованные модели. Дополнительно я попробовал так же использовать LightGBM вместо и в дополнение к XGBoost, но получал примерно те же результаты, и к концу соревнования отказался от использования LightGBM.

Важные признаки. Наиболее важными фичами как и ожидалось оказались цены билетов и производные от этих цен, включая долю от медианной цены в группе и другие. А так-же, признаки времени полета и числа пересадок. Существенными оказались и признаки возможности отмены или обмена билетов, класс билета.

Время тренировки моделей. Тренировка одиночной модели занимала порядка 10-15 минут. Тренировка итогового ансамбля из 10 моделей на полных данных заняла около 2-х часов.

Выбор признаков

Всего при обучении использовалось 330 признаков. Ниже список наиболее важных из них:

№	Признак	Важность (Feature Importance)
0	is_min_segments	18038.068359
1	free_exchange	525.814697
2	legs0_segments1_cabinClass	282.423248
3	avg_cabin_class	167.914963
4	legs0_segments1_departureFrom_airport_iata	166.922791
5	is_cheapest	162.626770
6	free_cancel	145.830124
7	legs0_segments0_cabinClass	138.951385
8	legs0_segments1_baggageAllowance_weightMeasurementType	116.882790
9	pricingInfo_isAccessTP	108.453781

Можно предположить, что данная таблица не полностью отражает важность различных факторов. Так как, например, число сегментов полета и пересадок было сконцентрировано в основном в одном признаке is_min_segments. В то время как фактор цены был разбит на десятки разных признаков средних, ранговых, что давало улучшение точности предсказания.

Выбор признаков: Для обучения признаки выбирались в основном интуитивно. Ожидая что будет важным для человека при выборе билетов. Затем выбор проверялся на валидации и если предположение о полезности признака подтверждалось, то такой признак сохранялся для обучения. В ряде случаев, например при добавлении категориальных признаков, касающихся истории выбора клиентов приводил к сильному переобучению. И, хотя использование таких признаков интуитивно казалось полезным. Но фактически использовать их не получалось. Поэтому, например фактор какой клиент/компания выбирали билет ранее по цене оказывался полезным. А Если попытаться учесть какой тип самолета или какого перевозчика выбирал клиент/компания ранее, это приводило к переобучению.

Преобучение: Признаки истории выбора клиентов приводили к переобучению, поэтому участник соревнования, занявший первое место, @goldenlock, указывал в своем решении, что не использовал их совсем до окончания соревнования. Однако, уже после соревнования, проверив предложенный мной подход к их использованию некоторой части этих признаков смог сильно увеличить скор своего финального решения.

Преодоление переобучения: Подход заключался в оконном принципе формирования признаков истории выбора клиентов. То есть каждой группе сохранялись признаки сформированная только по данным тех групп с выбором билета, где делали выбор те же клиенты до этой текущей группы с выбором билетов.

Исторические оконные признаки: Для тех признаков, по которым считалась история выбора опций в группе, конкретной компании или пользователя, формировались: средняя, медиана, число групп в которых ранее делался выбор и присутствовал данный признак. Так же формировались квантили 25% и 75%. Исторические признаки формировались на основе следующих фич:

- 'legs0_departureAt_hour', 'legs1_departureAt_hour', 'legs0_arrivalAt_hour', 'legs1_arrivalAt_hour'
- 'price_rank', 'price_from_median', 'duration_rank', 'duration_from_median',
- 'avg_cabin_class', 'baggage_total', 'l0_seg',
- 'legs0_segments0_seatsAvailable', 'legs0_segments0_baggageAllowance_quantity', 'legs0_segments0_cabinClass',
- 'miniRules1_statusInfos', 'miniRules0_statusInfos'

Тренировка моделей

Валидация: При подборе признаков данные разбивались на тренировочную и тестовую часть, которая была совмещена с валидацией. Что как я сейчас понимаю видимо было ошибкой и валидацию и тест все же следовало разделить. В то же время скор получаемый на валидации практически не отличался принципиально от скора полученного на Public и Private Leaderboard. И рост в локальной валидации чаще коррелировал с ростом в LB. Локальная валидация давала порядка 0.535. Для локальной валидации использовался последний месяц тренировочных данных.

Обучение: Обучение проходило на полных тренировочных данных без какого-либо использования данных из test. В том числе test не использовался для формирования статистик или кодирования категориальных признаков.

Фактор новых данных в тесте. Для того чтобы модель не сильно искажалась при появлении новых категорий в категориальных признаках, при обучении, примерно 1% категориальных данных для обучения случайно заменялся пропусками. Если таких пропусков было менее 1%. А затем, на тесте, неизвестные категории в категориальных данных так же отмечались как пропуски (-1), при кодировании категориальных данных.

Сокращение размера групп: Размышляя о возможности предварительного выбора наиболее перспективных кандидатов для финального выбора трех наиболее вероятных опций, которые выберет клиент, я попробовал просто случайно сократить все группы до максимум 100 опций. Сделав даунсамплинг негативных примеров (тех что клиент не выбрал). И это неожиданно улучшило скор. Как я предположил модели стало легче ранжировать, меньше стал дисбаланс классов. Потому что вероятность правильно угадать выбор пользователей в большой группе все равно был низким. Но без сокращения групп модели, возможно, было труднее выучить что-то полезное на такой большой группе. Так же побочным эффектом стало уменьшение памяти, используемой при обучении и повышение скорости обучения. А так-же, возможность использовать из-за этого больше признаков. При этом сокращение размера групп проводилось уже после формирования всех признаков и сбора статистики. То есть признаки формировались по полным группам. Сокращения групп до формирования признаков ухудшало результаты обучения моделей.

Ансамбль и функции потерь: В Финальном решении использовался ансамбль из 10 моделей. Это давало некоторый небольшой рост над результатом одной модели. Буквально на 0,001- 0,003. Но скорее давало не рост, а некоторую несколько большую стабильность результата. В ансамбле использовались две функции потерь у восьми моделей была функция потерь 'rank:pairwise'. А у двух оставшихся 'rank:ndcg'. Предсказания моделей в ансамбле просто суммировалось, подбор весов не осуществлялся.

Время работы моделей

Подготовка данных: формирования значительной части признаков на полных данных занимало около минуты. Однако формирование оконных признаков истории выбора отдельных клиентов, а также отдельных компаний занимало дополнительно по 5 минут. Итого подготовка данных занимала около 15 минут.

Тренировка моделей: Тренировка одной модели занимала около 15 минут. Тренировка ансамбля из 10 моделей около 2-х часов 30 минут.

Генерация предсказаний: Время генерации предсказаний 10 моделями ансамбля на представленных тестовых данных 7 минут.

Гипер параметры.

Для обучения моделей ансамбля использовались два набора гиперпараметров, полученных с помощью optuna:

```
base_params_1 = {
    'objective': 'rank:pairwise',
    'eval_metric': 'ndcg@3',
    'learning_rate': 0.03430629350470104,
    'max_depth': 20,
    'min_child_weight': 24,
    'subsample': 0.9680833434687813,
    'colsample_bytree': 0.1859969541434444,
    'lambda': 13.73979976725866,
    'alpha': 4.315374828140574,
    'gamma': 0.145064313893779,
    'n_jobs': -1,
}
```

```
base_params_2 = {
    'objective': 'rank:pairwise',
    'eval_metric': 'ndcg@3',
```

```
    'learning_rate': 0.01874682748459287,
    'max_depth': 17,
    'min_child_weight': 14,
    'subsample': 0.9910104516963302,
    'colsample_bytree': 0.24822082790651878,
    'lambda': 14.476928639040858,
    'alpha': 0.06344642592268934,
    'gamma': 0.2556097003945161,
    'n_jobs': -1,
}
```

Это были базовые параметры, на основании которых формировались конкретные параметры для каждой модели ансамбля:

- Параметр 'num_boost_round' варьировался вокруг значения 1200 +- 100. У двух моделей ансамбля число деревьев составляло 1700.
- У двух моделей функция потерь заменялась на 'rank:ndcg'

Возможные улучшения

Для улучшения качества предсказаний очевидным образом было бы хорошо получить более обширную и длительную историю по каждому клиенту и компании. Однако, такой истории именно по авиабилетам, для значительного числа клиентов может не быть в принципе. Например, потому что многие люди совершают не так много авиаперелетов и их предпочтения, статус, могут оставаться не ясными. Поэтому перспективным мне бы представлялось получение некоей дополнительной информации о клиентах за пределами информации о выбранных ими билетах. Что возможно было при синергии их выбора и предпочтений. Может быть, даже использование каких-то относительно открытых данных вроде кредитной истории. Если есть доступ к каким-то банковским данным, покупкам товаров - то возможно эти данные.

Может быть даже просто фото людей с паспорта, место прописки, паспортным данным, записей голоса. Чтобы по сформированным эмбедингам попытаться учесть степень состоятельности клиента и может быть что-то еще.

Так же нужно найти какой-то способ обучать модель без переобучения не только на числовых данных, но и на исторических категориальных данных. Что в рамках данного решения сделать не удалось.

Очень перспективными кажутся подходы в формировании дополнительных признаков с помощью языковых моделей. Чтобы с механизмом внимания что-то получать из всего описания предложенных опций. И попытки применения данного подхода были упомянуты рядом участников соревнований. Пока без особых результатов.

Репозиторий

Ссылка на github решения: https://github.com/Michael-Golubchik/flight_rank_doc