GeekBrains

Мой первый ML курсовой проект на факультете AI GB

Подтема

Предсказание цен московской недвижимости

Дмитрий Яковлев

Закончил факультет искусственного интеллекта GeekBrains..
Закончил курсы повышения квалификации в МГТУ им. Баумана.

Немного о себе.

- Проживаю в г. Новосибирск, ищу работу в г. Москва.
- Инженер по работе с клиентами и проектировщиками на предприятии, производящем высоковольтное оборудование.

План проекта

1. Пройденный этап: DataPreprocessing
2. Пройденный этап: EDA
3. Пройденный этап: FeatureGeneration
4. Пройденный этап: настройка регрессоров
5. Пройденный этап: создание общего пайплайна
6. Пройденный этап: создание product ready приложения
7. Будущий этап: доработка проекта
8. Будущий этап
9. Будущий этап
10. Будущий этап
11. Будущий этап
12. Будущий этап
13. Будущий этап
14. Будущий этап
15. Будущий этап
16. Будущий этап

Решение задачи / План работы

- Иследование данных на платформе Kaggle.
- Исследование рабочего датасета: поиск аномалий, анализ распределений, корреляций, исследование методов работы с количественными и категориальными признаками, исследование выбросов, поиск оптимального алгоритма построения регрессора, определение полезных признаков.
- Генерация новых признаков, исследование их корреляции, попытка нахождения полезных кластеров (ноутбук Featureengeneering)
- Разработка регрессора.
- Разработка product ready приложения на основе фреймворка Flask

Трудности

Очень значимое количество пропусков в признаках. Попытка заполнения пропусков с помощью ML-моделей или KNN_Imputer может привести к искажению данных, подгонке данных под решатель. Поэтому пытался заполнить пробелы используя статистику и сопоставления признака с пропусками с наиболее коррелирующим признаком.

Также имелось много искаженных данных: не соответствующих статистике либо конфликты между похожими признаками.

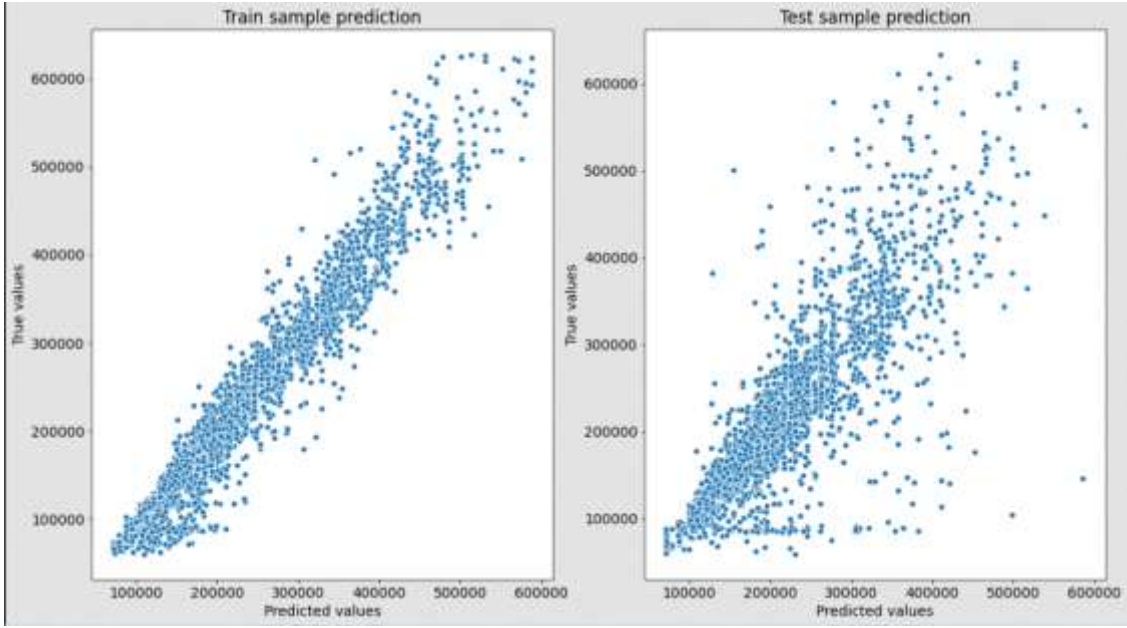
Проблему представляет разброс цен на похожие объекты. Здесь сказывается недостаток информации. Нужны дополнительные признаки. Ведь одинаковые квартиры даже на одной площадке могут различаться по цене в зависимости от стоимости ремонта либо даже на какую сторону окна.

Хотелось бы также узнать расстояние до центра, ближайшего метро, про общественный транспорт и т.д. Но пришлось работать с имеющимися данными. Возможно часть такой информации удалось добавить с помощью признаков, полученных на основе признаков датасета. Впрочем далеко не все сгенерированные признаки увеличили метрики.

В датасете имеется значительное количество выбросов.

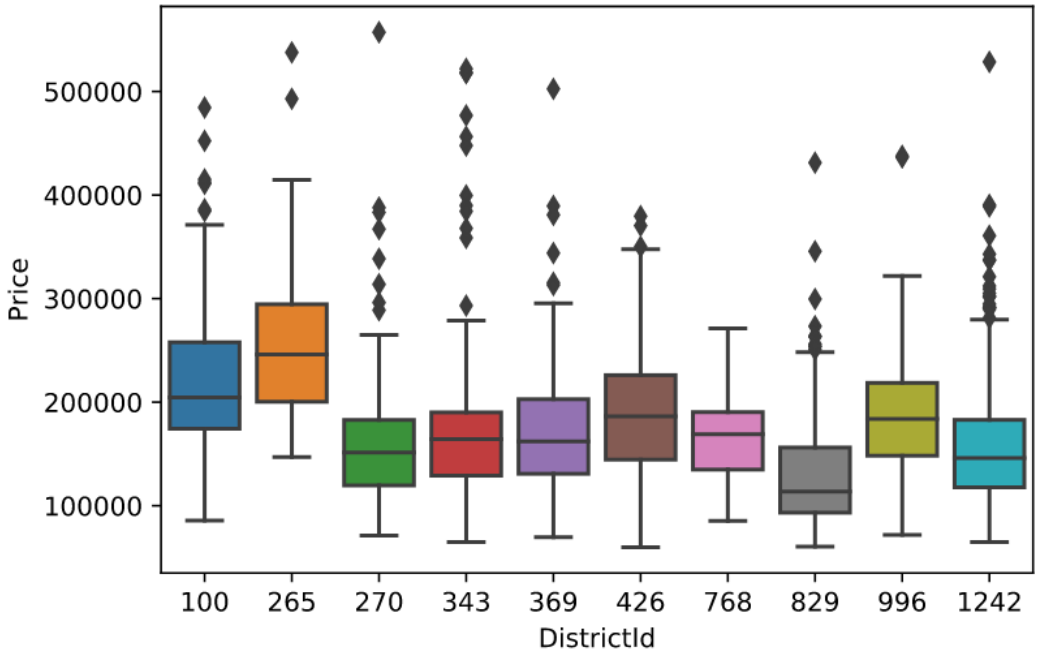
Это мой первый ML-проект, поэтому здесь EDA и предобработка данных не на высоте. Но класс с предобработкой данных состоит из 184 строчек кода, поэтому переделывать не стал. Тем более решатель работает хорошо.

Baseline



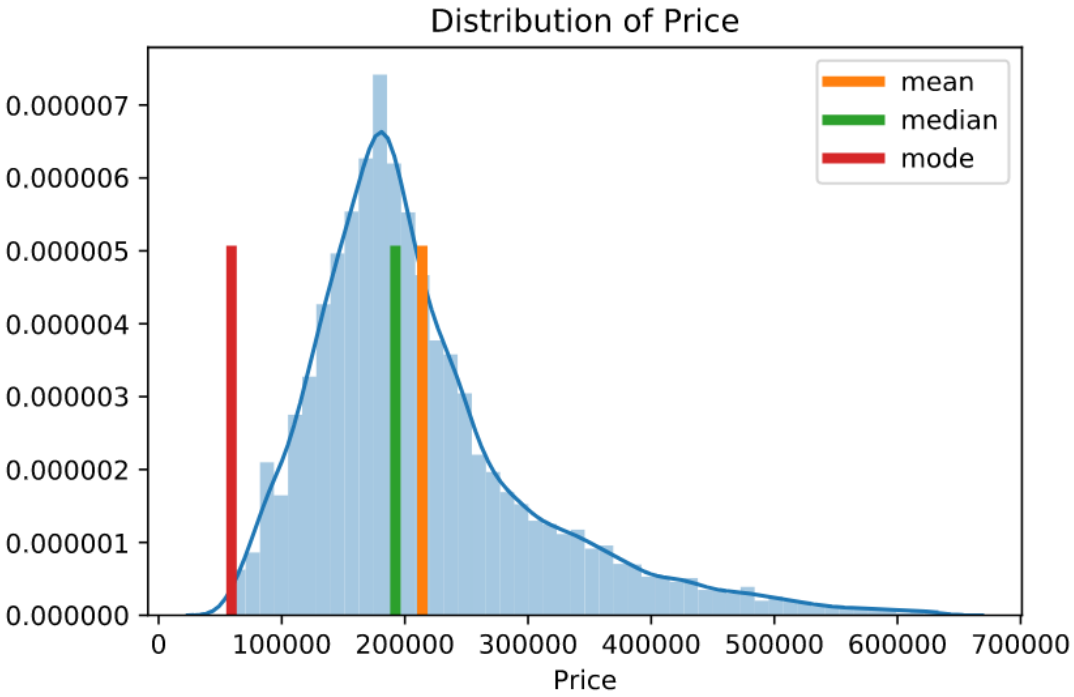
baseline-решатель, предложенный преподавателем имел R2 на валидации 0,7, но на тренировочных данных 0,952 (дикое переобучение, что не удивительно в связи с недостатком необходимой информации).

Выбросы



В датасете много выбросов. Вот, например, выбросы цен по районам. Но эти выбросы свидетельствуют о недостатке информации. Ведь цены в зависимости от различных параметров могут значительно разниться.

EDA

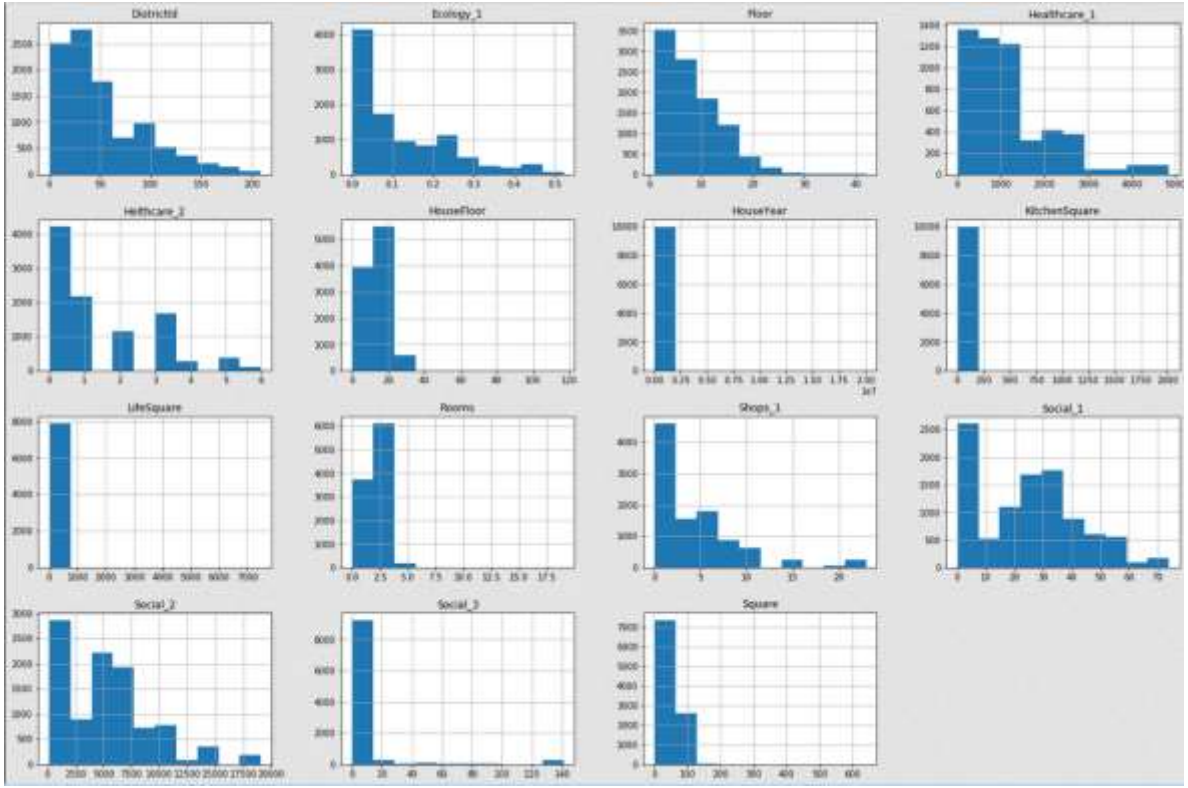


Анализ достаточно обширный чтобы его весь описать. Обычный статистический анализ здесь не очень помог, т.ак как много аномалий.

Стоит отметить очень сильное смещение по средним на распределении признака Price. Что впрочем не удивительно. Народ ест капусту, а слуги народа мясо, а в среднем весь народ ест голубцы.

Ну и судя по средним, ориентироваться надо на медианные значения.

EDA



И в целом распределения признаков оставляют желать лучшего. Нормального распределения практически нигде не наблюдается. И выбросы видны даже невооруженным взглядом.

В результате пришлось делать класс DataPreprocessing на 168 строчек кода, чтобы исправить эту ситуацию.

FeatureGeneration

Были сгенерированы следующие признаки:
Некачественных наблюдений в соответствующих признаках: 'HouseFloor_outlier', 'HouseYear_outlier', 'LifeSquare_nan',

Энкодинг средних значений по соответствующим признакам:
'DistrictId_mark', 'DistrictId_E1_mark', 'DistrictId_Social2_mark', 'DistrictId_Healthcare_2_mark', 'Helthcare_2', 'MedPriceByKitchenLS', 'MedPriceByBFF', 'MedPriceBySocial1', 'MedPriceBySocial2', 'MedPriceByShop', MedPriceByDistrict‘

Категориальных из вещественных:
'Social1_cat', 'Social2_cat', 'Shop_cat', 'Ecol1_cat', 'year_cat', 'kitch_cat'
'LS_cat‘

Говорящих сами за себя:
'DistrictId_count', 'new_district', 'DistrictId_popular‘

Признаки добавлялись по наличию влияния на метрики. Думаю было сгенерировано достаточное количество признаков. Лучшее – враг хорошего.

Сжатие признаков

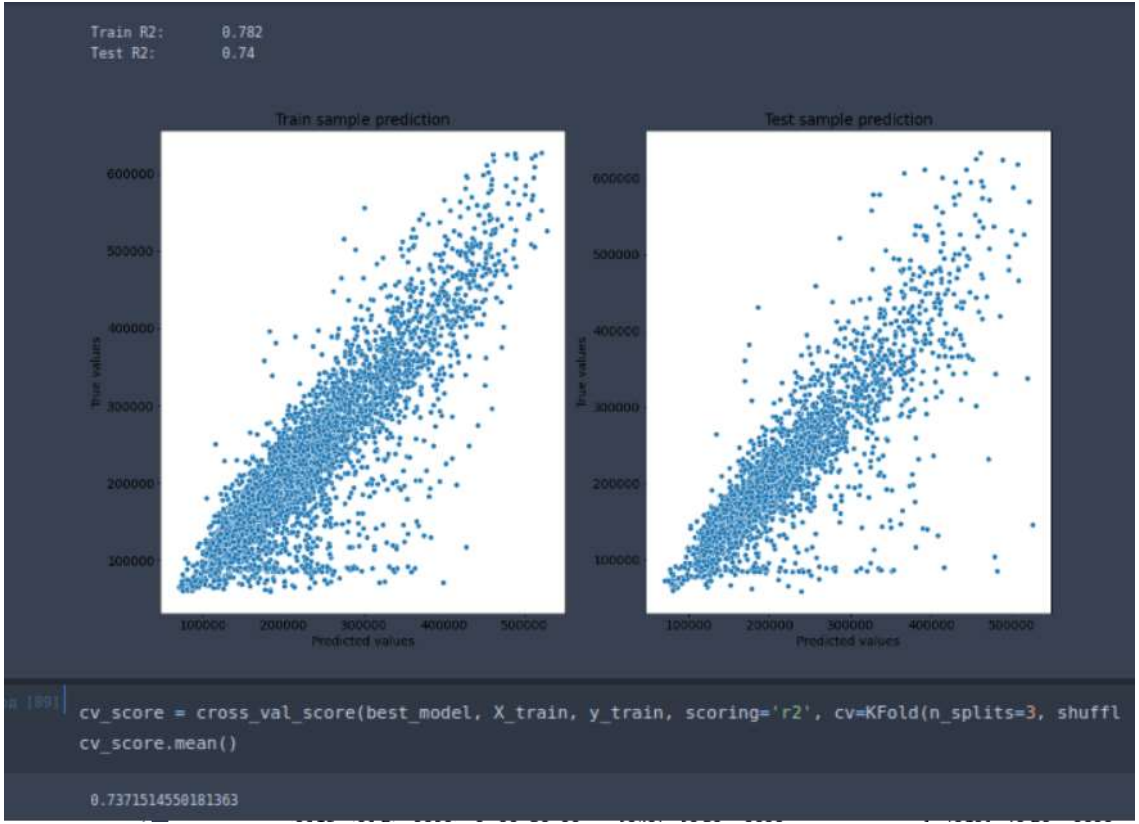
```
step_2 = My_pca(last_col, out_col)

X_train_scaled = step_2.fit_transform(X_train)
X_valid_scaled = step_2.transform(X_valid)
test_df_scaled = step_2.transform(test_df)
```

```
choise_features(X_train_scaled, y_train, X_valid_scaled, y_valid)
```

Для нормализации датасета и сжатия малозначимых признаков создал класс My_pca. В нем открыл классы MinMaxScaler и PCA. Не стал наследоваться от этих классов чтобы не запутать для понимания код.

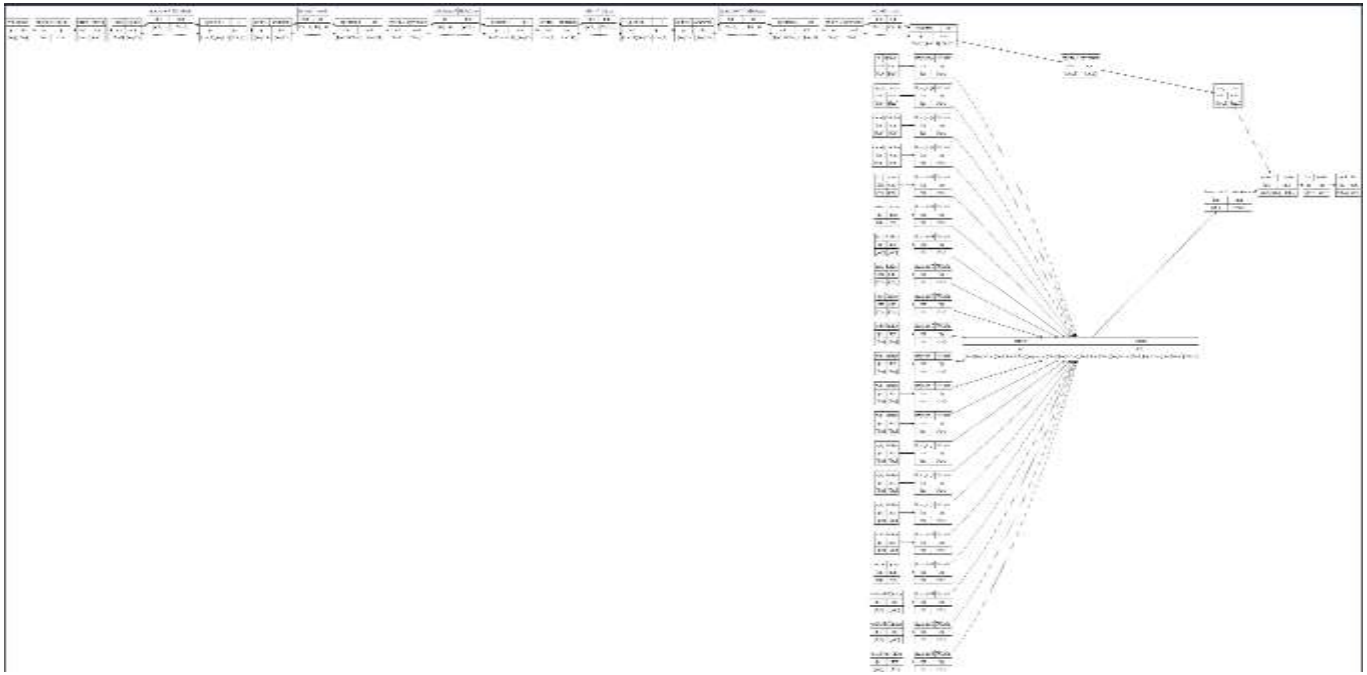
Проверка после преобразования показала что метрики не ухудшились.

Выбор класификатора



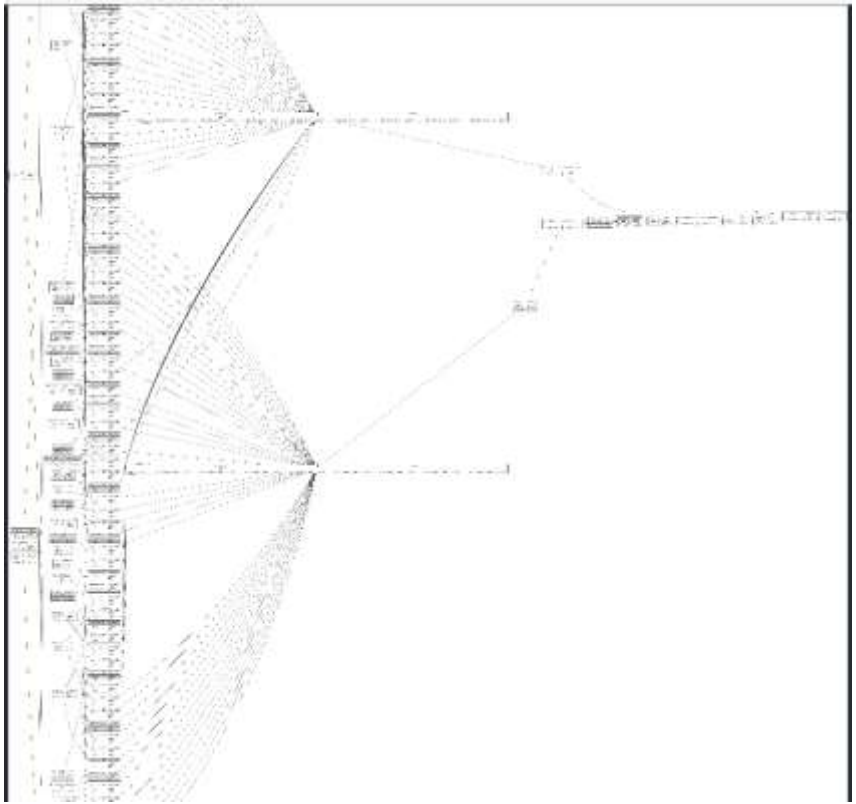
Из арсенала ML использовал XGBRegressor, CatBoostRegressor, LGBMRegressor и LGBM. Лучшие результаты показал CatBoostRegressor. Подбор гиперпараметров осуществлял встроенным методом Катбуста randomized_search.

Выбор класификатора



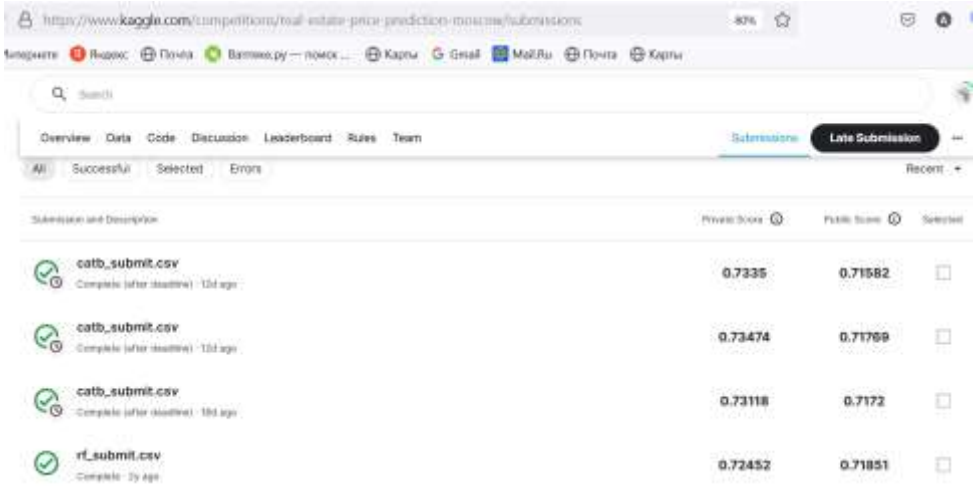
В ноутбуке real_es_price_msk_MLP подбирал архитектуру Multi Layer Perceptron для этой задачи. Лучшая архитектура оказалась такой. Но она показала метрику хуже чем Катбуст. R2 на тестовой выборке 0,69.

Выбор класификатора



Такая композиция глубоких моделей показала максимальный R2=0,65.

Достигнутые цели



Удалось перебить baseline по целевой метрике более чем на 3%, при этом переобучение отсутствует в отличие от baseline. При чем на приватном лидерборде. А на публичном лидерборде метрика значительно ниже чем на приватном.

Из истории сабмитов видно, что я не подгонял предсказание под приватный датасет. А то, что мой решатель работает в приложении с единичными предсказаниями говорит о том, что не подгонял предсказание и под тестовый датасет.

В результате более глубокой проработки проекта после окончания курса повышения квалификации в МГТУ им. Баумана удалось на 1% улучшить метрику по сравнению с решением двухгодичной давности.

Предложения по проекту

Нет предела совершенству и можно улучшать решение до бесконечности. В первую очередь провести более качественный EDA. Можно усложнить структуру решателя. Можно применить генератор признаков, различные разбиения. Более сложный таргет энкодинг. Более глубоко проработать нейронные сети.