GeekBrains

Мой первый ML курсовой проект

Подтема

Предсказание цен московской недвижимости

Дмитрий Яковлев

Закончил курсы повышения квалификации в МГТУ им. Баумана.
Закончил факультет искусственного интеллекта GeekBrains..
Закончил Сибирскую аэрокосмическую академию по специальности «Системы управления ракетно-космическими объектами и комплексами летательных аппаратов»

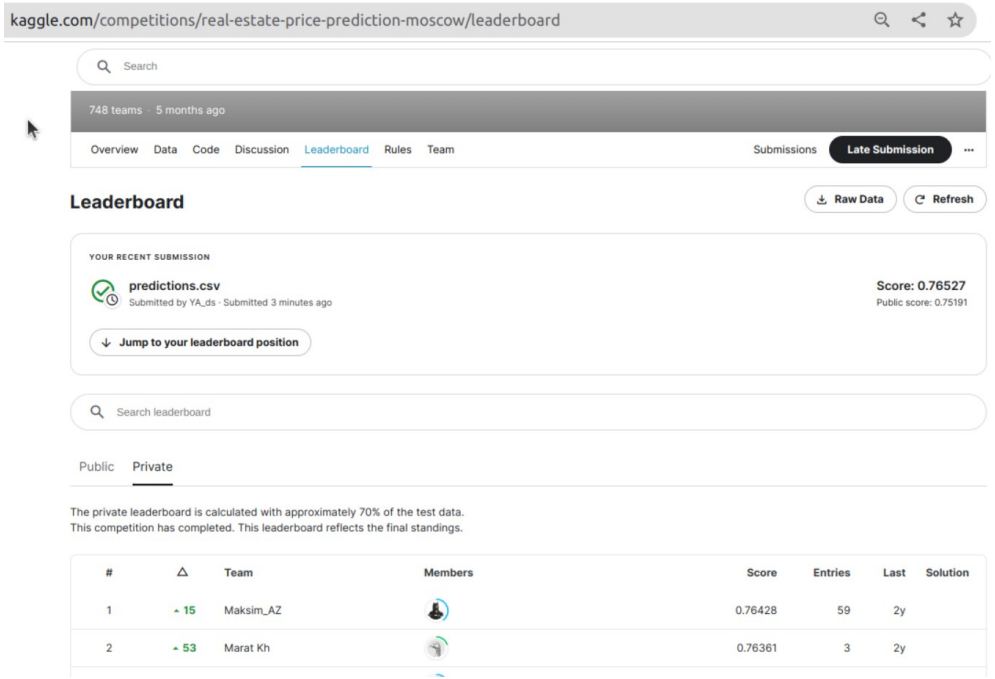
Немного о себе.

Инженер по работе с проектировщиками и клиентами 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
9. Будущий этап
10. Будущий этап
11. Будущий этап
12. Будущий этап
13. Будущий этап
14. Будущий этап
15. Будущий этап
16. Будущий этап

Достигнутые цели



Удалось найти наилучшее предсказание на лидерборде соревнования на Каггл по этому проекту.

Поставил для себя задачу достичь результата на базовом регрессоре (для небольших датасетов лучше всего Catboost) и без фиксации random_state и seed (чтобы не было повода думать что я подгонял свои результаты под эти гиперпараметры).

Кроме того выяснилось, что приватный лидерборд отлично коррелирует с кроссвалидацией. Корреляция пропала после генерации похожих наблюдений нейросеткой GAN (возникло переобучение), но это неизбежно из-за изменения распределений признаков в датасете. И не факт, что лидеры лидерборда не пользовались подобными методами.

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.
- 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 привела к искажению распределений 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.

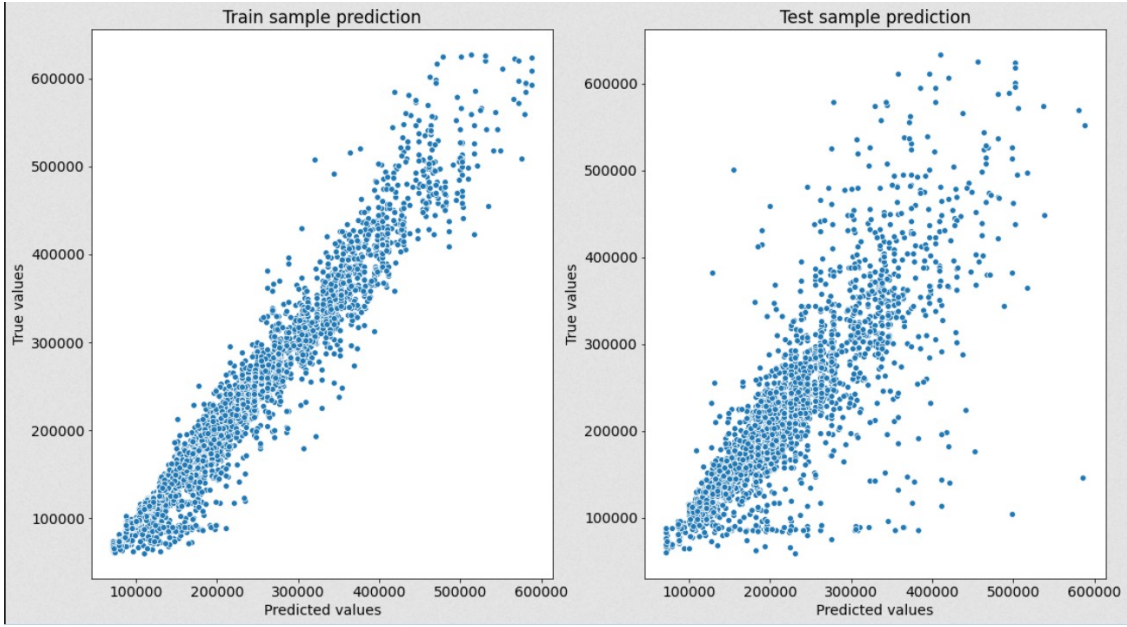
Проблему представляет разброс цен на похожие объекты. Здесь сказывается недостаток информации. Нужны дополнительные признаки. Ведь одинаковые квартиры даже на одной площадке могут различаться по цене в зависимости от стоимости ремонта либо даже на какую сторону окна.

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.

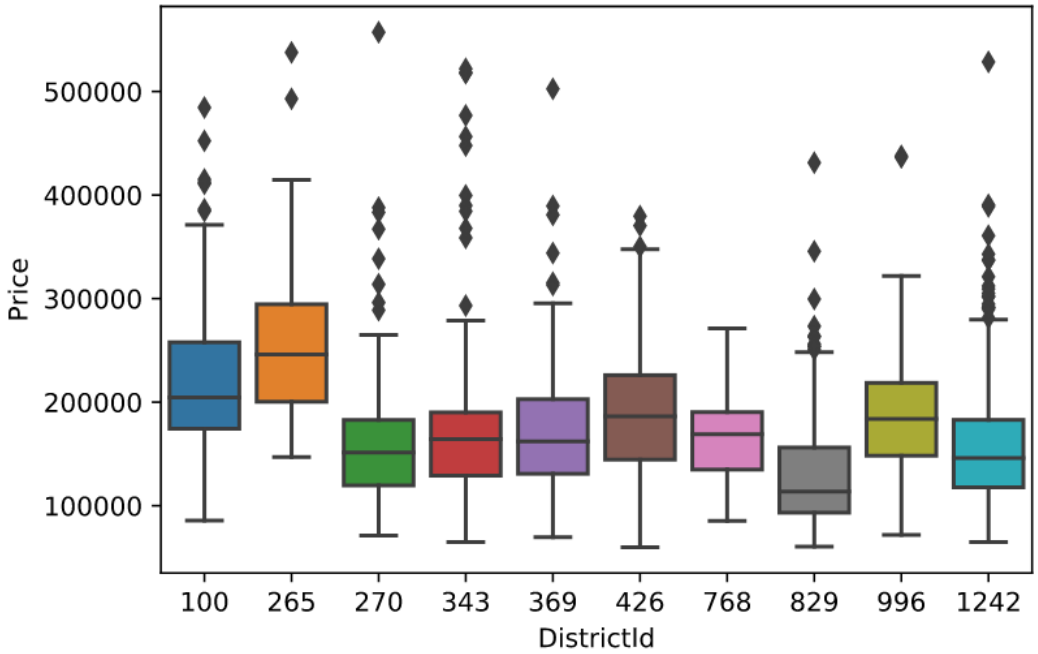
Это мой первый ML-проект, поэтому здесь EDA и предобработка данных не на высоте. Переделывать не стал. Тем более и на этих данных удалось найти наилучшее решение в лидерборде на Каггле.

Baseline



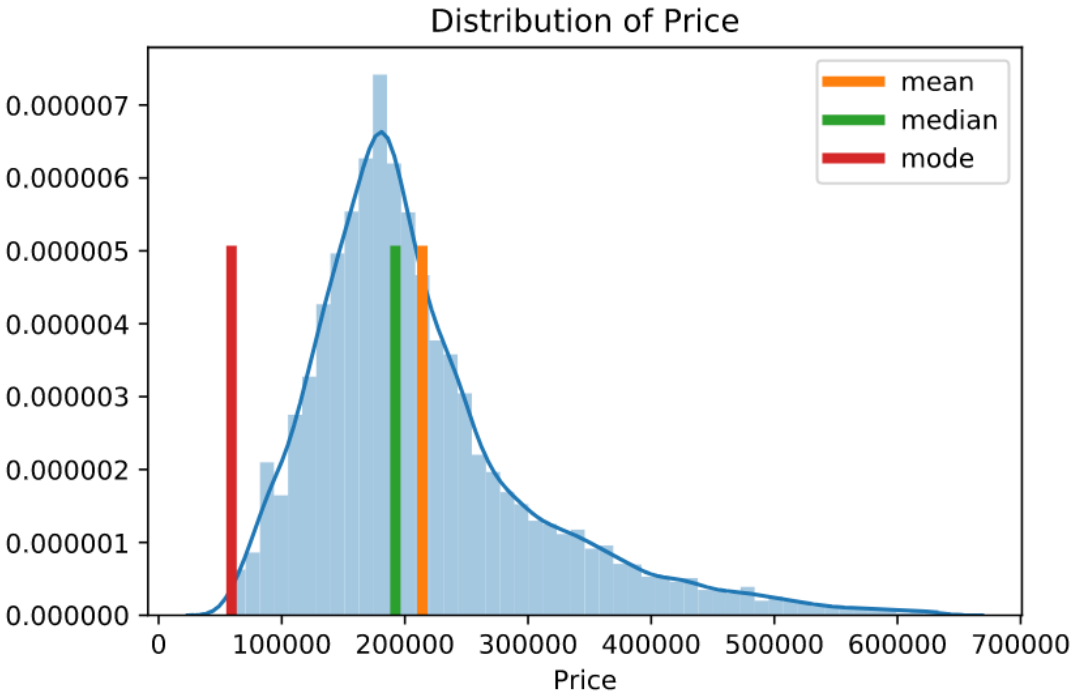
baseline-решатель, предложенный преподавателем имел R2 на валидации 0,7, но на тренировочных данных 0,952 (дикое переобучение, что не удивительно в связи с недостатком необходимой информации). ВАЖНО! Это метрика на тренировочных данных, на Kaggle меньше.

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

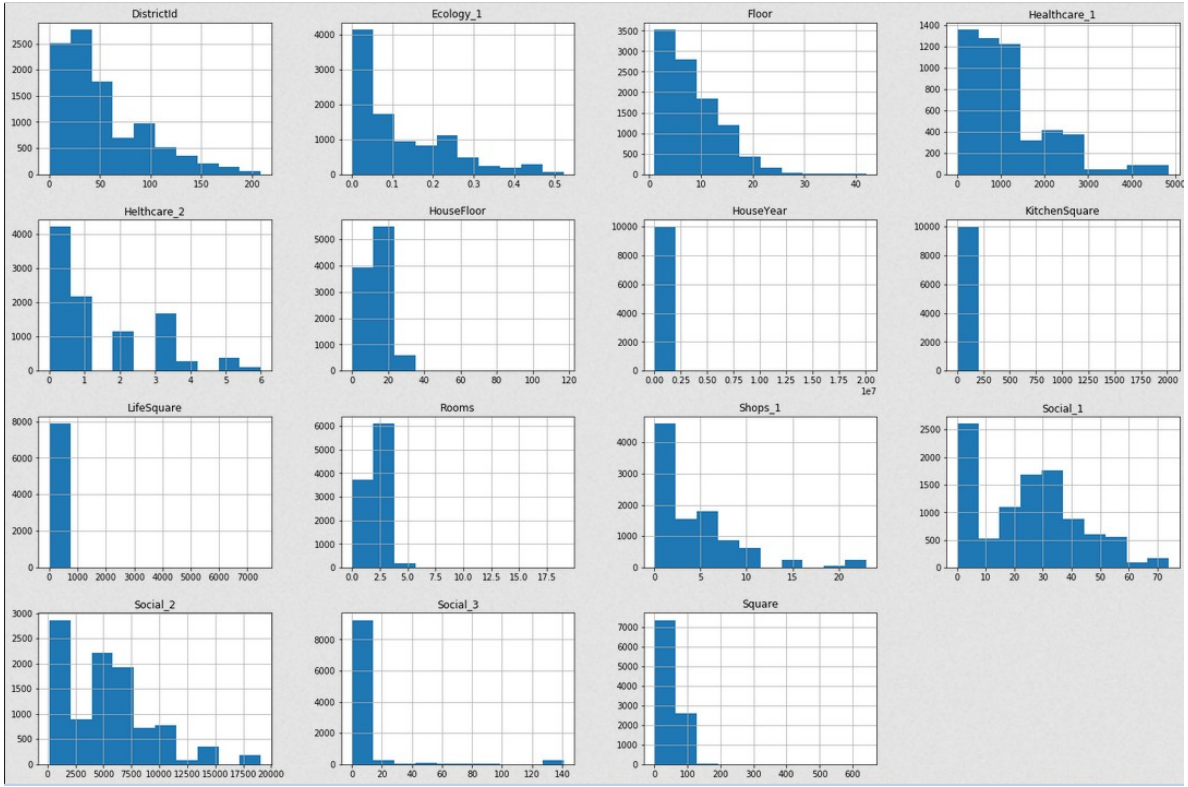


Анализ достаточно обширный чтобы его весь описать. Обычный статистический анализ здесь не очень помог, так как много аномалий.

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



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

В результате пришлось делать класс FeatureImputer, чтобы исправить эту ситуацию в пределах критичности для «деревянных» моделей.

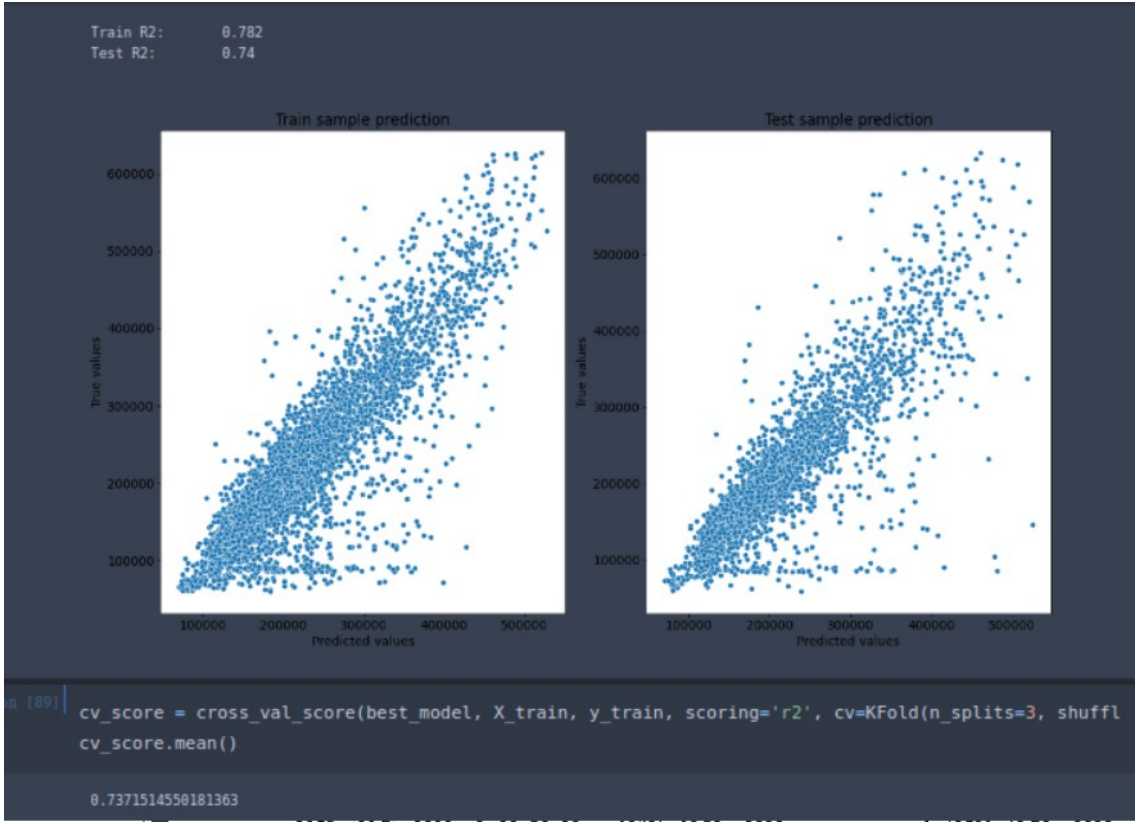
FeatureGeneration

Были сгенерированы следующие работающие признаки:

Энкодинг средних значений по признаку ‘DistrictId’:
'DistrictId_mark', 'DistrictId_E1_mark', 'DistrictId_Healthcare_2_mark‘
Категориальный из вещественного:
'LS_cat‘

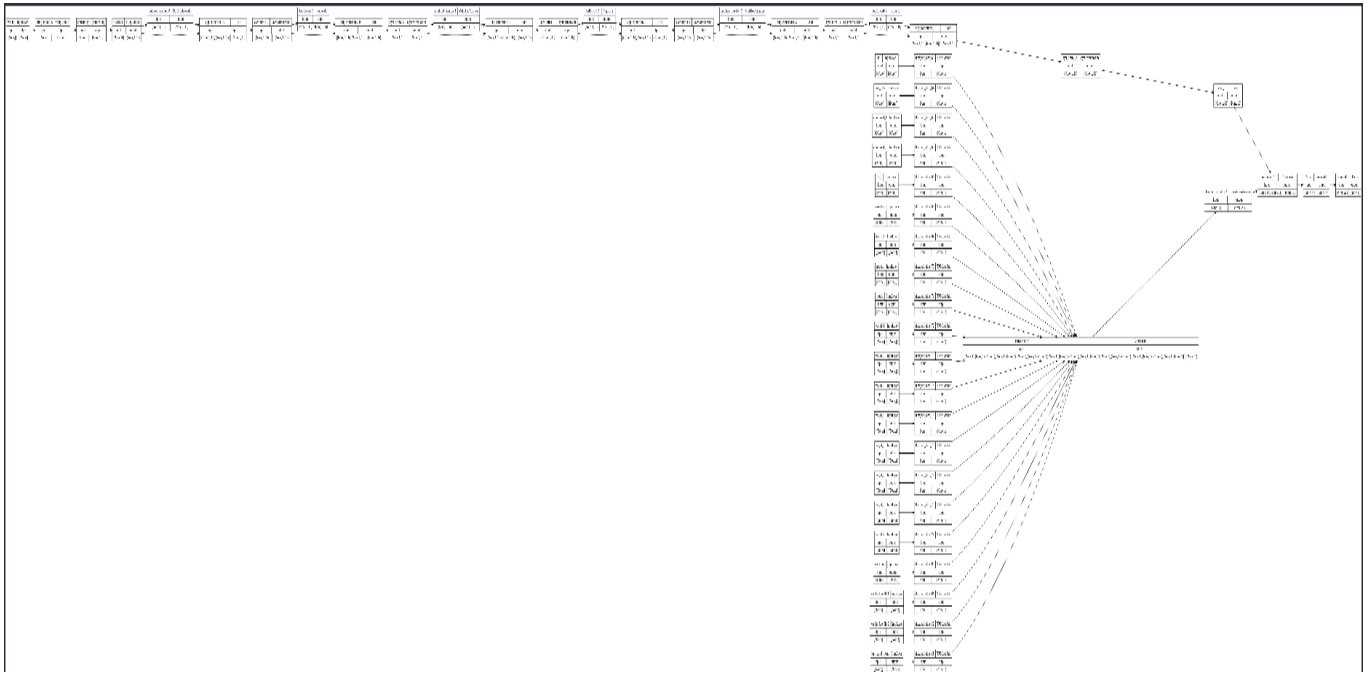
Говорящих сами за себя:
'DistrictId_count'
Признаки добавлялись по наличию влияния на метрики при валидации.

Выбор класификатора



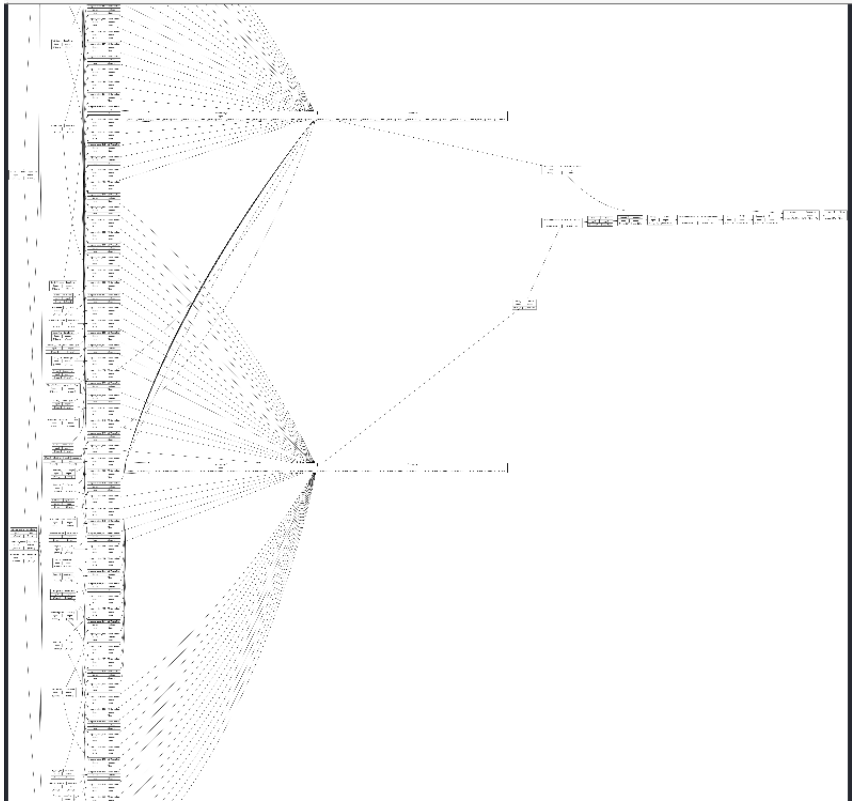
В качестве базового решения выбрал решатель CatBoostRegressor. Катбуст в качестве базового решателя выгоден тем, что он более устойчив без подбора гиперпараметров по сравнению с другими бустерами. Кроме того имеет достаточно сильный внутренний энкодер для категориальных признаков. Подбор гиперпараметров осуществлял встроенным методом Катбуста randomized_search по локальным максимумам на графике.

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 немного хуже чем Катбуст, но должна иметь низкую корреляцию с предсказаниями бустинговых моделей. И по этой причине она должна хорошо зайти при стекинге предсказаний нескольких моделей.

Выбор класификатора



Еще одна перспективная композиция глубоких моделей для стекинга. Показала немного меньшую метрику относительно предыдущей модели.

Предложения по проекту

В первую очередь попытаться настроить валидацию на основании похожести объектов в тренировочном датасете на объекты в тестовом датасете. Провести более качественный EDA и предобработку данных. Предобработку делал для «деревянных» моделей, поэтому сделать еще предобработку для нейронных сетей. Можно применить генератор признаков, различные разбиения. Сложный таргет энкодинг. Более глубоко проработать нейронные сети. Проработать стэкинг моделей.