ШИФТ
Экосистема

Решение задачи прогнозирования цены недвижимости

Команда "Софтмаксеры"

Клюева Виктория, Скибин Константин, Трофимов Ярослав

Содержание

1. Представление команды

Команда "Софтмаксеры"

Клюева Виктория

Придумала некоторые новые фичи, подбирала модели, гипермараметры, приняла решение использовать блендинг.

Скибин Константин

Помогал на всех этапах решения задачи.

Трофимов Ярослав

Занимался обработкой данных, поиском закономерностей и проверкой гипотез. Придумал новые фичи, алгоритм подбора коэффициентов блендинга. Занимался обучен ием моделей.

2. Работа с данными

Обработка и генерация
новых фич, поиск закономерностей

Признак full_sq

Самое большое влияние на качество модели

```
dff['full_sq'][(dff['full_sq'] > 1000)] = 45
dff['full_sq'][(dff['full_sq'] > 250)] = dff['full_sq'] / 10
dff['full_sq'][(dff['full_sq'] == 0)] = 1
dff['full_sq'][(dff['full_sq'] == 1)] = 66
dff['full_sq'][(dff['full_sq'] == 5) | (dff['full_sq'] == 6)] = 45
dff['full_sq'][(dff['full_sq'] == 9)] = 52
```

Было замечено, что в подавляющем большинстве квартиры больше 250 кв.м. стоили очень мало (<10млн.) .

Это выглядело так, будто их площади были преувеличены в 10 раз.

Так же были обработаны аномалии в данных.

Новые фичи

На основе мониторинга домов с наибольшей ошибкой модели

Некоторые новые фичи во многом связаны с мониторингом домов, в которых модель очень сильно ошибается. Выводились графики распределения фич и сравнивались с графиками для полных данных.

Иногда получалось находить некоторые взаимосвязи. Они и стали основой следующих фич:

```
dff['ExpensiveMeter1'] = dff['full_sq'] * dff['num_room']
dff['ExpensiveMeter2'] = dff['radiation_raion'] * np.sqrt(dff['full_sq'])
dff['Test4'] = (dff['product_type'] + 2) * np.sqrt(dff['full_sq'])
dff['Test5'] = np.minimum(dff['cemetery_km'], dff['oil_chemistry_km']) * dff['full_sq']
```

Новые фичи

Которые дали наибольший прирост к результату

Единственный ощутимый пророст дала обработка фичи full_sq.

Все остальные фичи обычно давали прирост на уровне погрешности, поэтому было очень сложно оценивать их пользу.

Наиболее ощутимый прирост показали следующие фичи:

```
dff['Room_area'] = dff['full_sq'] / (dff['num_room'] + 1)
dff['build_age'] = dff['year'] - dff['build_year']
```

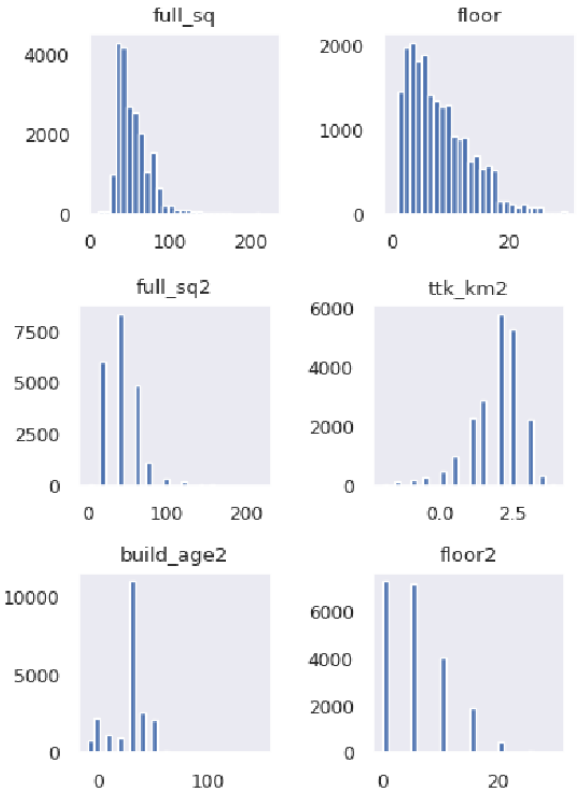
Фичи - клоны

Идея округления непрерывной фичи до нескольких групп

Многократно было выявлено, что удаление, казалось бы, бесполезных фич не улучшает метрики.

Появилась идея создать фичи - клоны. Их суть заключается в округлении всей фичи до 5-10 групп. Ожидалось, что модель сможет лучше понять некоторые скрытые взаимосвязи между фичами.

Получилось немного улучшить метрики.



Идеи, от которых мы отказались

1. Удаление выбросов
2. Отбор признаков через матрицу корреляции или специально обученную модель

3. Обучение моделей

Опробованные модели и ансамбли из них

Модели, не показавшие хорошего результата

1. Любые линейные модели
2. SVM
3. Random Forest

Так же не удалось добиться результатов с использованием AutoML

1-я идея. Ансамбли

HistGradientBoostingRegressor и CatBoost

С подобранными гиперпараметрами на преобработанном датасете позволяли добиться значения RMSE в районе 2 900 000

Идея с переобучением

Еще на этапе обработки данных была замечена высокая схожесть тренировочных и тестовых данных.

Эффективно оказалось переобучать модели, но до некоторого порога.

Такое решение уменьшило RMSE на 50K.

2-я идея. Блендинг

Обучить еще больше эффективных моделей и усреднить их предсказания.

```
model_xgboost = XGBRegressor(
    min_child_weight=0,
    subsample=0.7,
    colsample_bytree=0.7,
    objective='reg:squarederror',
    nthread=-1,
    scale_pos_weight=1,
    reg_alpha=0.00006,
    n_estimators=2500,
    max_depth=6,
    learning_rate= 0.009,
    gamma= 0.01
  )
```

```
params_lgb = {
    'objective': 'regression',
    'metric': 'rmse',
    'boosting': 'gbdt',
    'verbose': -1,
    'num_leaves': 15,
    'n_estimators': 3000,
    'max_depth': 10,
    'learning_rate': 0.01,
    'bagging_fraction': 0.5
}
model_lgb = lgb.LGBMRegressor(**params_lgb)
```

```
model_cat = CatBoostRegressor(
    learning_rate= 0.1,
    l2_leaf_reg=5,
    iterations=550,
    depth=8,
    loss_function='RMSE',
    verbose=0
  )
```

Гиперпараметры были подобраны с помощью GridSearch и RandomSearch.

RMSE снизилась до уровня 2 800 000.

3-я идея. Подбор коэфициентов блендинга

Вручную эксперементальным путем и с помощью написанного алгоритма

Разные комбинации коэффициентов могли улу
шить метрику на 35K - 50K для одинаково
обученных моделей.

```
# Коэффициенты блендинга подобраны экспериментальным путем
prediction = 0.1 * y_pred_model_lgb + 0.7 * y_pred_model_xgboost + 0.2 * y_pred_model_cat
```

4. Заключение

Решения, вошедшие в итоговый ноутбук и результат соревнования

Структура финальной версии ноутбука

1. Исправление аномалий и генерация новых фич
2. Обучение XGBoost, lightGBM и CatBoost моделей
3. Блендинг результатов

Результат соревнования

4	Softmaxers123	10	07/11/24	2773132.63 (3)

5. Впечатления от ШИФТ интенсива

Впечатления

Интенсив очень понравился, особенно формат, в котором проводились занятия и формат соревнования, что, безусловно, добавляло дополнительную мотивацию, ведь каждый стремился победить.

Интенсив помог погрузиться в мир машинного обучения, узнать много нового и получить ценный практический опыт.