

СЕРВИС ДЛЯ ОЦЕНКИ СТОИМОСТИ НЕДВИЖИМОСТИ

Дополнительное задание
ко второму туру

Ларин Кирилл Андреевич – 10 класс – Кузбасс (Капитан, анализ проблемы)

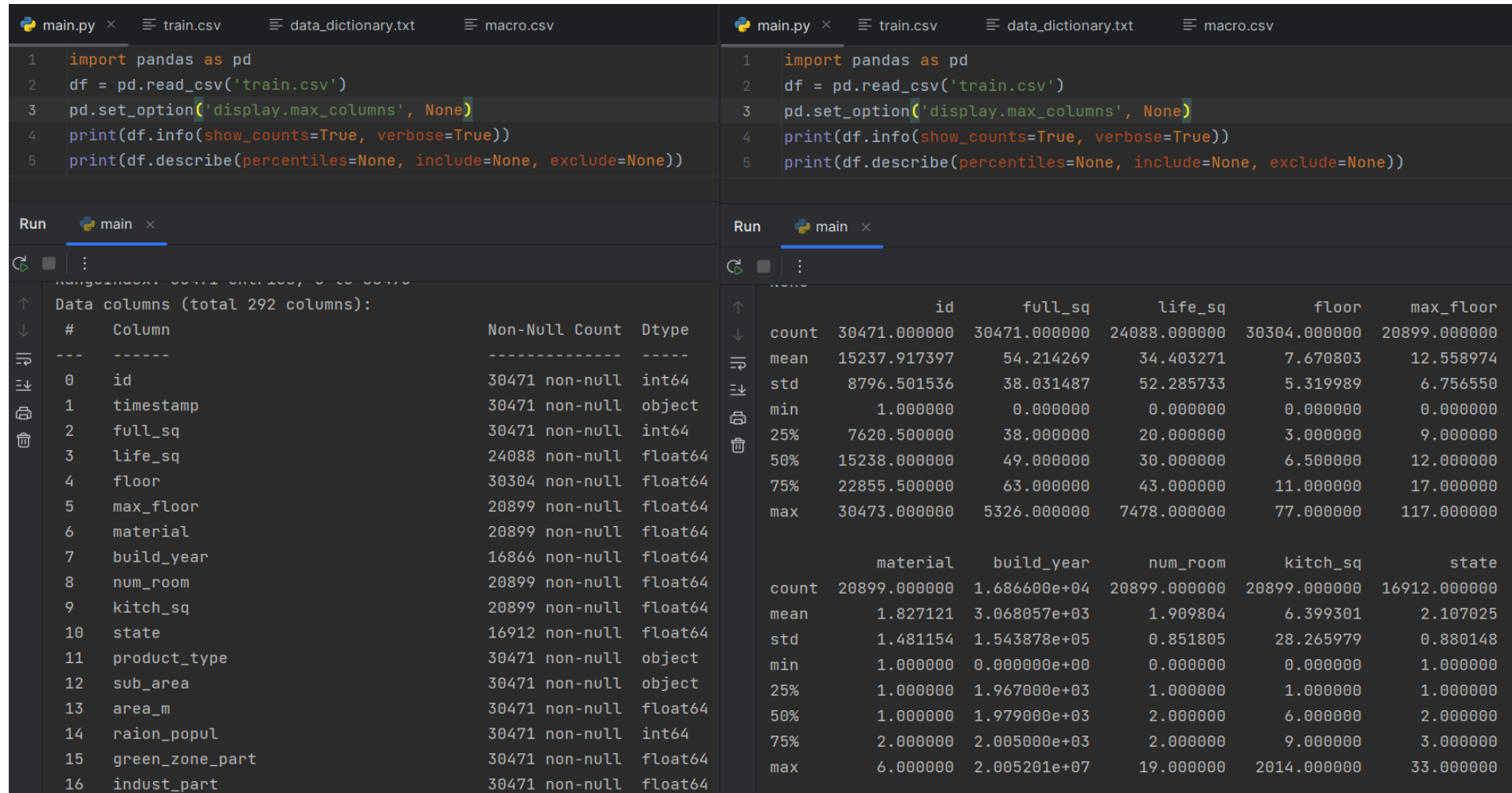
Филиппов Семён Сергеевич – 9 класс – Кузбасс (Front-end)

Сахибов Холмухаммад Фирдавсович – 9 класс – Кузбасс (Обучение модели)

Исаков Илья Михайлович – 8 класс – Кузбасс (Сбор и подготовка данных)

Загрузка данных, вывод статистик

Для работы с данными выбрана библиотека pandas. Создан датафрейм, выведены основные метрики по каждому столбцу таблицы для дальнейшего анализа.



```
1 import pandas as pd
2 df = pd.read_csv('train.csv')
3 pd.set_option('display.max_columns', None)
4 print(df.info(show_counts=True, verbose=True))
5 print(df.describe(percentiles=None, include=None, exclude=None))
```

Run main

Data columns (total 292 columns):

#	Column	Non-Null Count	Dtype
0	id	30471 non-null	int64
1	timestamp	30471 non-null	object
2	full_sq	30471 non-null	int64
3	life_sq	24088 non-null	float64
4	floor	30304 non-null	float64
5	max_floor	20899 non-null	float64
6	material	20899 non-null	float64
7	build_year	16866 non-null	float64
8	num_room	20899 non-null	float64
9	kitch_sq	20899 non-null	float64
10	state	16912 non-null	float64
11	product_type	30471 non-null	object
12	sub_area	30471 non-null	object
13	area_m	30471 non-null	float64
14	raion_popul	30471 non-null	int64
15	green_zone_part	30471 non-null	float64
16	indust_part	30471 non-null	float64

Run main

	id	full_sq	life_sq	floor	max_floor
count	30471.000000	30471.000000	24088.000000	30304.000000	20899.000000
mean	15237.917397	54.214269	34.403271	7.670803	12.558974
std	8796.501536	38.031487	52.285733	5.319989	6.756550
min	1.000000	0.000000	0.000000	0.000000	0.000000
25%	7620.500000	38.000000	20.000000	3.000000	9.000000
50%	15238.000000	49.000000	30.000000	6.500000	12.000000
75%	22855.500000	63.000000	43.000000	11.000000	17.000000
max	30473.000000	5326.000000	7478.000000	77.000000	117.000000

	material	build_year	num_room	kitch_sq	state
count	20899.000000	1.686600e+04	20899.000000	20899.000000	16912.000000
mean	1.827121	3.068057e+03	1.909804	6.399301	2.107025
std	1.481154	1.543878e+05	0.851805	28.265979	0.880148
min	1.000000	0.000000e+00	0.000000	0.000000	1.000000
25%	1.000000	1.967000e+03	1.000000	1.000000	1.000000
50%	1.000000	1.979000e+03	2.000000	6.000000	2.000000
75%	2.000000	2.005000e+03	2.000000	9.000000	3.000000
max	6.000000	2.005201e+07	19.000000	2014.000000	33.000000

Обработка отсутствующих значений

```
11 df = df.drop(index=1)
12 print(df.shape)
13 columns = list(df.columns)
14 lst = list(df.select_dtypes(include=['object']).columns)
15 for column in columns:
16     if column not in lst:
17         df[column] = df[column].fillna(df[column].mean())
18 print(df.info(show_counts=True, verbose=True))
```

Run main x

207	church_count_1500	30471	non-null	int64
208	mosque_count_1500	30471	non-null	int64
209	leisure_count_1500	30471	non-null	int64
210	sport_count_1500	30471	non-null	int64
211	market_count_1500	30471	non-null	int64
212	green_part_2000	30471	non-null	float64
213	prom_part_2000	30471	non-null	float64
214	office_count_2000	30471	non-null	int64
215	office_sqm_2000	30471	non-null	int64
216	trc_count_2000	30471	non-null	int64
217	trc_sqm_2000	30471	non-null	int64
218	cafe_count_2000	30471	non-null	int64
219	cafe_sum_2000_min_price_avg	30471	non-null	float64
220	cafe_sum_2000_max_price_avg	30471	non-null	float64
221	cafe_avg_price_2000	30471	non-null	float64
222	cafe_count_2000_na_price	30471	non-null	int64

Проанализировав количество непустых ячеек в каждом столбце, было принято решение об удалении неинформативных столбцов и строк (количество непустых записей в которых соответственно <21330 и <204 (70% от общего числа)). Таким образом удалено 10 колонок и 0 строк. Оставшиеся пустые ячейки были заполнены средним значением по столбцу.

Обработка лишних значений

В ходе проведенного анализа можно заметить, что данные коррелируются с ценой.

В последнем столбце 'price_doc' указана зависимость от каждой строки.

В очищенном датасете больше всего коррелируют строки 'full_sq', 'num_room', 'life_sq', имея значения: 0.608, 0.502, 0.456.

	id	full_sq	life_sq	floor	max_floor	material	build_year	num_room	kitch_sq	state	price_doc
id	1.000000	0.046391	0.058977	0.002082	0.010992	-0.003783	0.032047	0.021957	-0.046112	-0.071205	0.083090
full_sq	0.046391	1.000000	0.863599	0.145572	0.164839	-0.017983	0.232453	0.798658	0.170791	-0.062420	0.608341
life_sq	0.058977	0.863599	1.000000	0.141682	0.105360	-0.033365	0.228419	0.677133	-0.183024	-0.274163	0.456044
floor	0.002082	0.145572	0.141682	1.000000	0.494951	-0.027113	0.352762	-0.011255	0.018450	-0.086640	0.102398
max_floor	0.010992	0.164839	0.105360	0.494951	1.000000	-0.022665	0.508753	-0.028003	0.172568	-0.052193	0.083708
material	-0.003783	-0.017983	-0.033365	-0.027113	-0.022665	1.000000	-0.043965	-0.042910	0.126893	-0.017265	0.017868
build_year	0.032047	0.232453	0.228419	0.352762	0.508753	-0.043965	1.000000	-0.057281	0.102672	-0.212218	0.018242
num_room	0.021957	0.798658	0.677133	-0.011255	-0.028003	-0.042910	-0.057281	1.000000	0.145338	0.076479	0.502347
kitch_sq	-0.046112	0.170791	-0.183024	0.018450	0.172568	0.126893	0.102672	0.145338	1.000000	0.386853	0.249087
state	-0.071205	-0.062420	-0.274163	-0.086640	-0.052193	-0.017265	-0.212218	0.076479	0.386853	1.000000	0.138883
price_doc	0.083090	0.608341	0.456044	0.102398	0.083708	0.017868	0.018242	0.502347	0.249087	0.138883	1.000000

Выявление аномалий

Аномалии и некорректность в данных присутствует. Это может быть связано с человеческим фактором. Сразу после загрузки данных и вывода статистики по ним мы заметили, что в некоторых столбцах присутствуют нулевые значения (например, `full_sq`, `life_sq` соответственно показывают общую и жилую площадь недвижимости, эти значения не могут равняться нулю, или год постройки здания не может быть позже 2015 года). В подобных случаях, необходимо удалить строку с некорректным значением целиком.

```
main.py x train.csv data_dictionary.txt macro.csv
1 import pandas as pd
2 df = pd.read_csv('train.csv')
3 pd.set_option('display.max_columns', None)
4 print(df.info(show_counts=True, verbose=True))
5 print(df.describe(percentiles=None, include=None, exclude=None))
```

Run main x

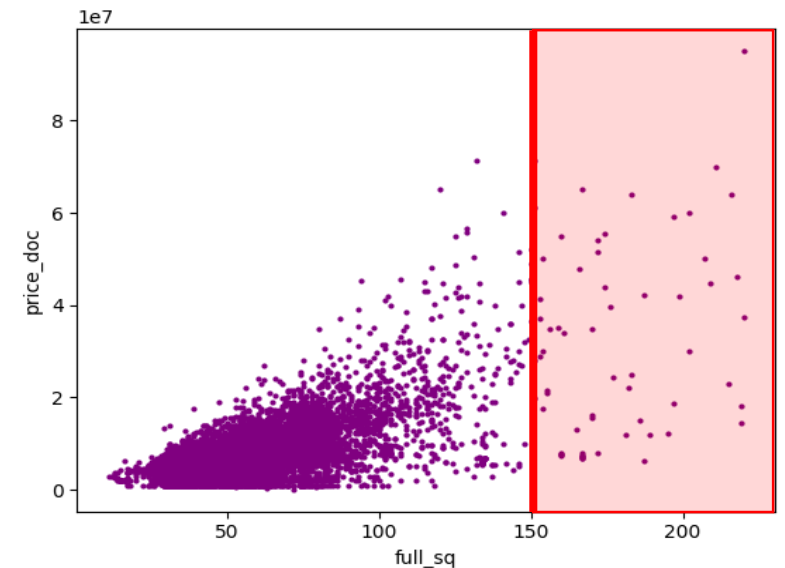
		id	full_sq	life_sq	floor	max_floor
count	30471.000000	30471.000000	24088.000000	30304.000000	20899.000000	
mean	15237.917397	54.214269	34.403271	7.670803	12.558974	
std	8796.501536	38.031487	52.285733	5.319989	6.756550	
min	1.000000	0.000000	0.000000	0.000000	0.000000	
25%	7620.500000	38.000000	20.000000	3.000000	9.000000	
50%	15238.000000	49.000000	30.000000	6.500000	12.000000	
75%	22855.500000	63.000000	43.000000	11.000000	17.000000	
max	30473.000000	5326.000000	7478.000000	77.000000	117.000000	

	material	build_year	num_room	kitch_sq	state
count	20899.000000	1.686600e+04	20899.000000	20899.000000	16912.000000
mean	1.827121	3.068057e+03	1.909804	6.399301	2.107025
std	1.481154	1.543878e+05	0.851805	28.265979	0.880148
min	1.000000	0.000000e+00	0.000000	0.000000	1.000000
25%	1.000000	1.967000e+03	1.000000	1.000000	1.000000
50%	1.000000	1.979000e+03	2.000000	6.000000	2.000000
75%	2.000000	2.005000e+03	2.000000	9.000000	3.000000
max	6.000000	2.005201e+07	19.000000	2014.000000	33.000000

Сбалансированность

Датасет не сбалансирован. Медианное значение должно примерно соответствовать среднему арифметическому данных по столбцу, чего во многих случаях не наблюдается. Также прослеживается не прямо пропорциональное увеличение данных в метриках 25%, 50%, 75%, что говорит о большом среднеквадратичном отклонении (std), т.е. несбалансированности. В качестве выхода из этой ситуации можно установить верхние и нижние границы по некоторым признакам. Например, по общей площади недвижимости, стоит брать записи $<150 \text{ м}^2$.

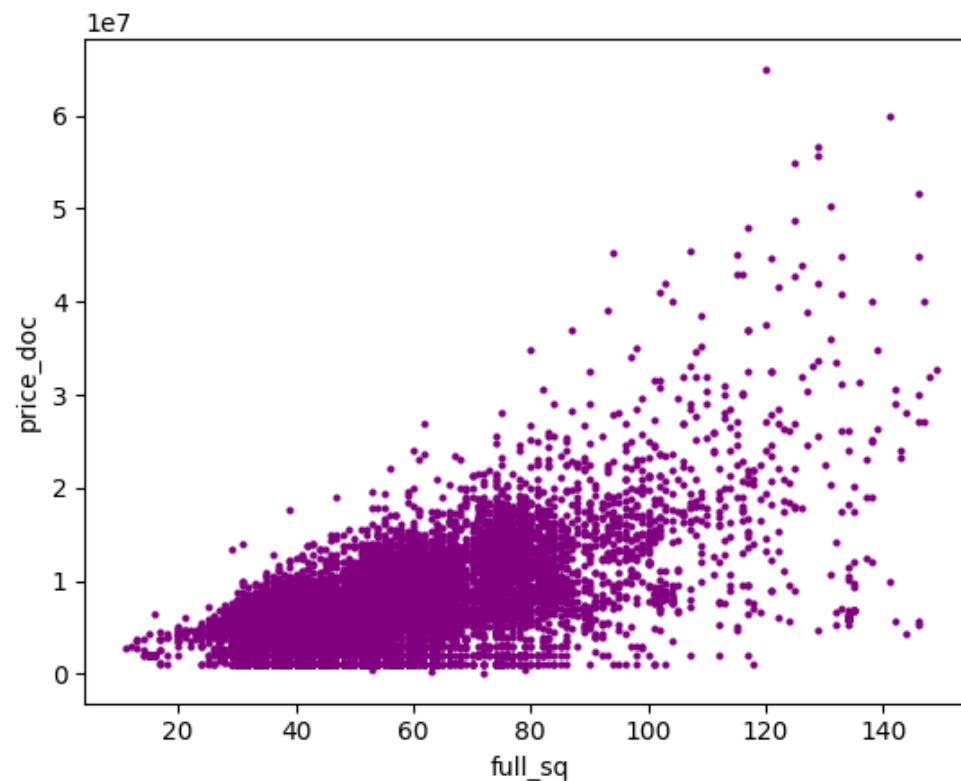
	shopping_centers_raion	office_raion	full_all	male_f \
count	17792.000000	17792.000000	1.779200e+04	17792.000000
mean	4.526922	8.755058	1.574488e+05	72319.010286
std	4.783980	23.993245	2.999139e+05	137109.545250
min	0.000000	0.000000	2.546000e+03	1208.000000
25%	1.000000	0.000000	4.379500e+04	21145.000000
50%	4.000000	2.000000	8.946700e+04	41288.000000
75%	6.000000	5.000000	1.253540e+05	58773.000000
max	23.000000	141.000000	1.716730e+06	774585.000000



Базовый отбор признаков

```
df.drop(labels: ['ID_railroad_station_walk', 'ID_railroad_station_avto', 'ID_big_road1', 'ID_big_road2',  
               'ID_railroad_terminal', 'ID_bus_terminal', ], axis=1, inplace=True)
```

Лишними колонками в данном датасете являются столбцы в роде «ID_railroad_station_walk», который содержит в себе информацию об *идентификационном номере* ближайшей ж/д станции. Данная информация никак не влияет и не поможет для оценки стоимости недвижимости.



Подобные графики были построены с разными признаками

Статистики

Было проведено исследование датасета предоставленного в условиях, с найденным нами на Kaggle [датасетом](#). По итогам сравнения было выявлено, что стоимость на недвижимость с идентичными характеристиками отличаются. Недвижимость в новом датасете при тех же данных стоит дороже, чем в выданном датасете. Причиной этому может служить инфляция, санкции (повышение издержек на строительство новостроек застройщиками).

Выводы

Проведено полноценное исследование датасета по итогам которого можно отметить, что подготовка данных для машинного обучения достаточно сложный и трудоёмкий процесс, который ранее нами недооценивался. Данные для обучения модели нейронной сети следует тщательно очищать от ячеек, которые могут ухудшить точность работы НС, проверять их на сбалансированность, корреляцию между собой и т.д. Для этого используются разнообразные методы, статистики