Big Data Team Project

Students:
Kirill Batyshchev
Nagim Isyanbaev
Viktor Kovalev

Course:	Big Data
Semester:	Spring
Year:	2024

Contents

1. Introduction

In Russia, many people are involved in buying and selling properties online every day, making online real estate marketing very popular. However, figuring out the right price for a property is tough because the real estate market is complex and always changing. So, our team is working on a solution: a predictive model that can analyze property listings and help users decide if a property is priced too high or too low. This model could help users make smarter choices when buying or selling real estate, picking out the best deals. Also, it could be useful for sellers who aren't sure how to price their properties accurately.

Our business objective is to create a service that gives accurate estimates for real estate prices. To make money from this service, we offer it as a subscription or selling it to online real estate platforms. If these platforms use our service, they can provide better value to their users by helping them make better pricing decisions. This might also help them make more money from the fees they charge for listing properties.

2. Data Description

Our team choose the "Russian Real Estate 2021"[1] dataset from Kaggle for our project. This dataset collects information from various websites like avito.ru, realty.yandex.ru, cian.ru, sob.ru, youla.ru, n1.ru, moyareklama.ru. The data was collected in 2021 by a service called ads-api.ru. The dataset was cleaned by the author to remove duplicate ads. The original dataset contains information about 11,358,150 real estate properties in Russia. The target is to predict the price column (Regression task)

Features

Dataset includes 2 geo-spatial features, 1 time feature, 6 numerical features, and 6 categorical features, making a total of 15 different features.

- **date** – date of publication of the announcement.
- **price** – price of a real estate in rubbles.
- **level** – apartment floor.
- **levels** – number of stories.
- **rooms** – number of living rooms. If equals to -1 then this is "studio apartment".
- **area** – the total area of the apartment.
- **kitchen_area** – area of the kitchen. (-100 means no kitchen)
- **geo_lat** – latitude of the building.
- **geo_lon** – longitude of the building.
- **building_type** – facade type (0 - Unknown, 1 - Other, 2 - Panel, 3 - Monolithic, 4 - Brick, 5 - Blocky, 6 - Wooden).
- **object_type** – apartment type (0 - secondary real estate market, 1 - new building).
- **postal_code** – postal code.
- **street_id** – anonymized street ID.
- **id_region** – region of Russia. There are 85 subjects in the country.

– **house_id** – anonymized house ID.

3. Architecture of Data Pipeline

Preprocessing

Splitting the Dataset into Houses and Announcements Dataframes: We divided the dataset into two parts to save space and make the data easier to work with.

- The first part includes data about apartments and announcements (*announcement id, date, price, level, levels, rooms, area, kitchen area, house id*), named **real_estate_announcements**.
- The second part contains data only about houses (*street, house id, postal code, latitude, longitude, region, building type, object type*), named **houses**.

We assigned a unique ID to each announcement.

Removing Outdated Announcements: We removed samples with an unknown *object_type* which have copy announcement with known *object_type*.

Generating House ID for Houses with Missing IDs: Houses without IDs were given synthetic IDs for identification.

Setting Data Types for Each Column: We manually adjusted the data types for each column based on their values.

Input

Unstructured dataset in the form of the **input.csv** file from Kaggle, which was uploaded to Yandex Disk.

Output

The preprocessing resulted in two CSV files: **real_estate_announcements.csv** and **houses.csv**.

Stage 1

Input

Preprocessed dataframes **real_estate_announcements.csv** and **houses.csv**.

Output

Tables **houses** and **real_estate_announcements** in a cluster PostgresSQL database. The same tables in .avro format stored in HDFS in *user/team14/project/warehouse*. Schema files **houses.avsc** and **real_estate_announcements.avsc**. ORM class files **houses.java** and **real_estate_announcements.java**

Stage 2

Input

`*.avsc` and `*.java` files for building Hive tables.

Output

Hive database named `team14_projectdb` with tables: `houses` and `real_estate_announcements`, `houses_part` partitioned version of `houses`, and `real_estate_announcements_buck` bucketed version of `real_estate_announcements`. Extracted 5 different insights from the data and automatized the extraction process via Hive queries, creating tables for each analysis result and importing them to .csv files. Manually created Apache Superset charts for each insight.

Stage 3

Input

Two tables from the hive database named team14_projectdb: houses_part (partitioned version of the table of with features of houses) and real_estate_announcements_buck (bucketed version of the table with real estate listings).

Output

Two model instances in hdfs models folder (model1 - linear regression with best hyperparameters, model2 - random forest with best hyperparameters). Train and test splits of data (with 70-30 ratio) in hdfs data folder. Two .csv files (model1_prediction.csv and model2_prediction.csv) with predictions of the two best models along with the correct labels in hdfs in output folder. In the same folder in hdfs, there is also evaluation.csv which contains the comparison of R-squared score and Root Mean Squared Error of the two best models.

Stage 4

Input

SQL tables of initial data, `*.csv` files of models predictions and metrics, `*.json` files with model's input features, data insights charts.

Output

Hive tables of models predictions(**model1_predictions**, **model2_predictions**), their metrics (**evaluation**) and input features(**test_data**, **train_data**), `stage4.sh` which automate their creation. Charts for data description and model evaluation, published dashboard, final report.

4. Data Preparation

Following preprocessing, our dataset was divided into two tables to occupy less space and achieve second normal form (all non-key columns depend on an entire primary key), resulting in the creation of the Entity-Relationship (ER) diagram depicted in Figure 1.

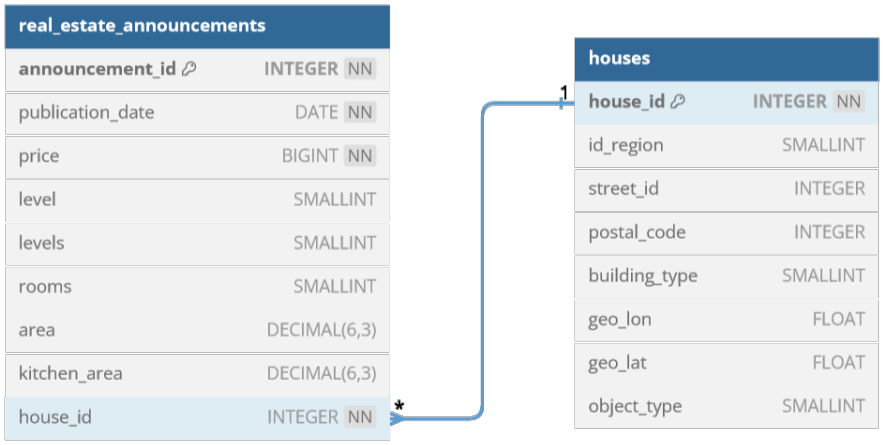

Figure 1: Entity-Relationship Diagram of the prepared dataset.

The content of each table is illustrated in Figure 2 and Figure 3. These figures were generated by querying a subset of 10 random samples from each table in PostgreSQL.

house_id	id_region	street_id	postal_code	building_type	geo_lon	geo_lat	object_type
1549541	23	232513	352630	4	39.865139	44.767944	0
1549548	33	355229	602267	4	42.0430485	55.57568739999999	0
1549550	23	330667	353445	5	37.3326501	44.892316799999996	0
1549553	10	128960	185034	4	34.4478109	61.754488	0
1549555	11	451890	167019	6	50.8358151	61.66866170000001	0
1549567	18	435328	427011	4	53.31147320000001	56.91720920000001	0
1549570	47	378113	188683	0	30.698532	59.785776	0
1549582	36	498038	396005	4	39.189542700000004	51.7926023	2
1549588	16	563306	420004	1	49.0292403	55.8239975	0
1549596	74	327619	456316	0	60.1230946	55.0656201	0

Figure 2: 10 samples from `houses` table.

announcement_id	publication_date	price	level	levels	rooms	area	kitchen_area	house_id
6252341	2021-07-27	3420000	12	18	2	48.77	0	3193786
960775	2021-02-12	2737900	20	25	1	40.8	8	3189281
892283	2021-02-10	13080000	22	24	2	56	9.5	2298254
9687846	2021-11-07	7200000	10	24	1	38.7	0	1239632
4322819	2021-05-28	3950000	2	5	3	57.1	5.6	3082864
1578609	2021-03-01	1950000	4	5	1	31.1	5.5	762392
2760081	2021-04-06	3767400	6	24	1	27.3	13.9	2381326
4018599	2021-05-18	4200000	6	14	1	42	-100	2089648
11081830	2021-12-18	5490000	2	45	-1	31	-100	831406
4403207	2021-05-31	1940000	5	5	2	43.3	5.6	2951090

Figure 3: 10 samples from `real_estate_announcements` table.

The PostgreSQL tables were migrated to Hive using Sqoop. For the `houses` table, we used partitioning, while for the second table, bucketing was utilized to facilitate faster interactions with the data. These Hive tables were then used for data analysis, and the resulting analysis outputs were converted into `.csv` format for integration into Superset for further visualization.

5. Data Analysis

Listings show various relationships related to the price, and not all of them are obvious.

1. **Floor Level**: The influence of floor level on real estate pricing (see Figure 4) shows a pattern of price growth with higher floor levels. This phenomenon can be attributed to various factors. Firstly, high-rise apartment buildings often demand higher prices due to their association with luxury living. Moreover, taller buildings offer beautiful views. In summary, the combination of prestigious location and impressive views associated with high floors contributes to the observed trend of rising real estate prices.

2. **Number of Rooms**: The relationship between the number of rooms in a property and its price (see Figure 5) shows a positive correlation. However, there are dip in price at the maximum room number. It means that some of the cases with big number of rooms do not increase living space. Properties with an unusually high room count may have unique characteristics that affect their price. For instance, these properties can belong to dormitories or communal housing arrangements, which could impact their attraction and pricing.

3. **Area**: The relationship between property area and price (see Figure 6) shows no strict linear increase in price with the expansion of area, but shows a general upward trend in pricing.

4. **Building Material**: This feature related to the facade of the building. On the Figure 7 plotted relationship between material and average price. Properties constructed with panel, brick, or blocky materials tend to have lower prices, while monolithic materials are associated with higher price. Monolithic materials are often associated with higher quality and durability. Buildings with monolithic facades often have a more luxurious or prestigious appearance compared to those with panel, brick, or blocky facades. This aesthetic can justify higher prices. Moreover, the cost of materials for monolithic facades is higher compared to other ones.

5. **Property Type**: The distinction between new construction and secondary housing (see Figure 8) reveals that secondary housing commands higher prices, attributed in part to factors such as enhanced furnishings or historical significance. Moreover, the majority of secondary housing is located near to the center of city.

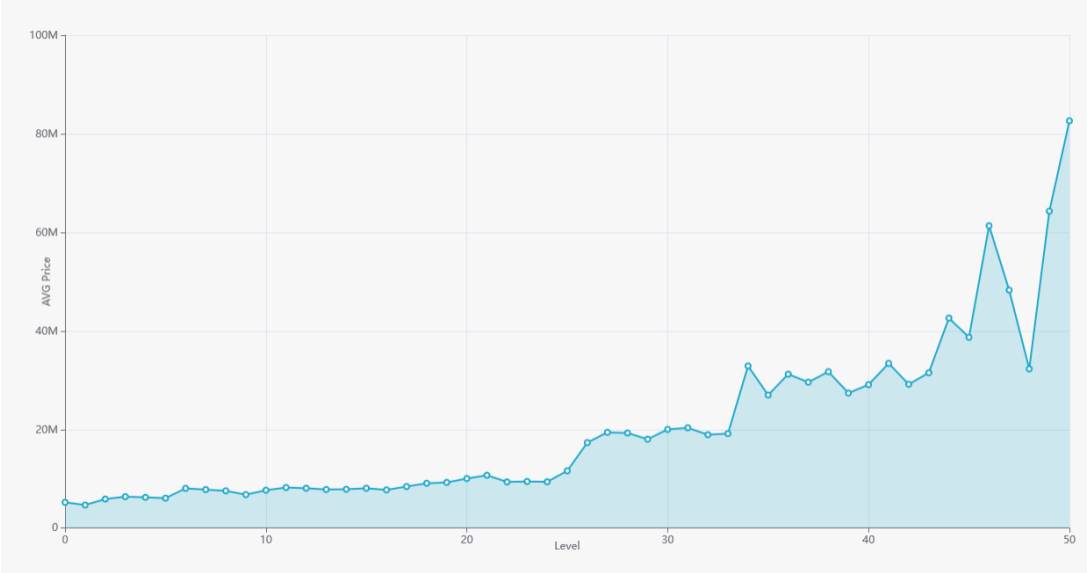

Figure 4: Average Price by Floor Level

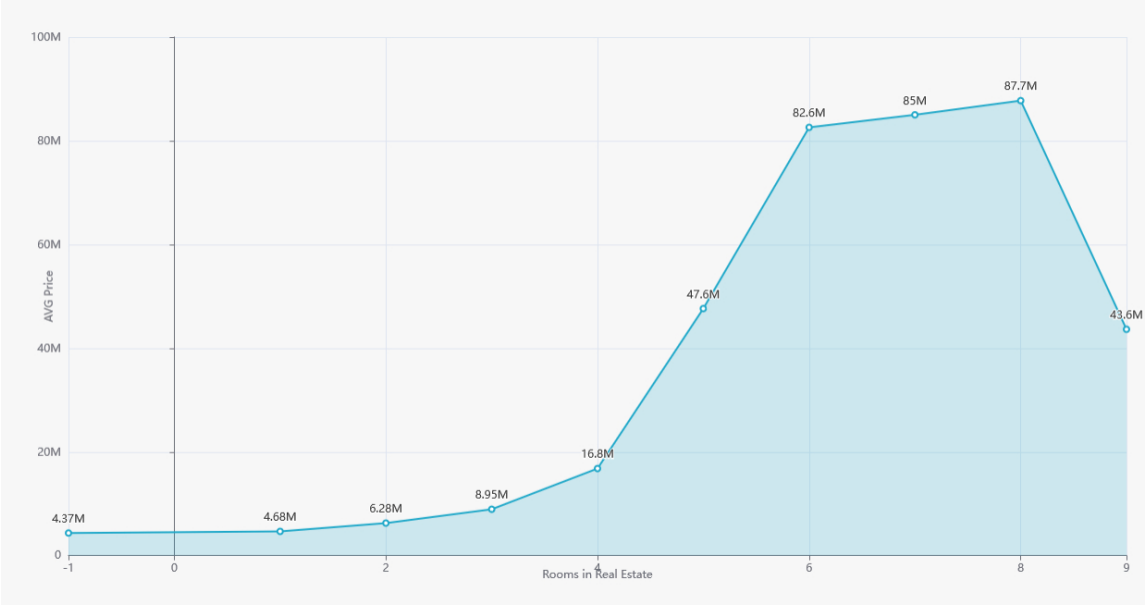

Figure 5: Average Price by Number of Rooms

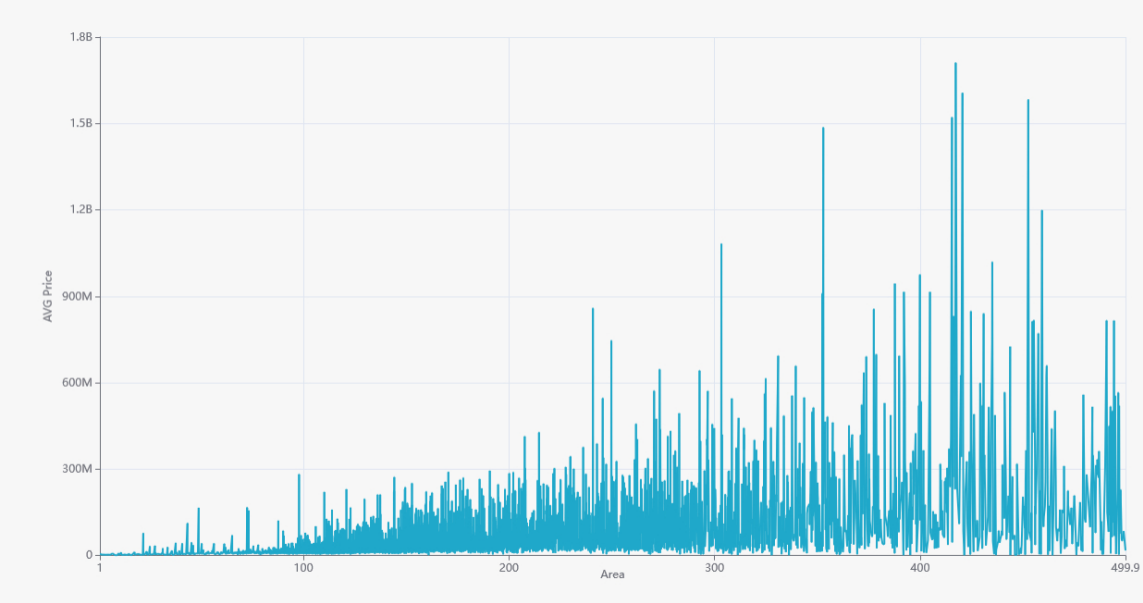

Figure 6: Average Price by Area

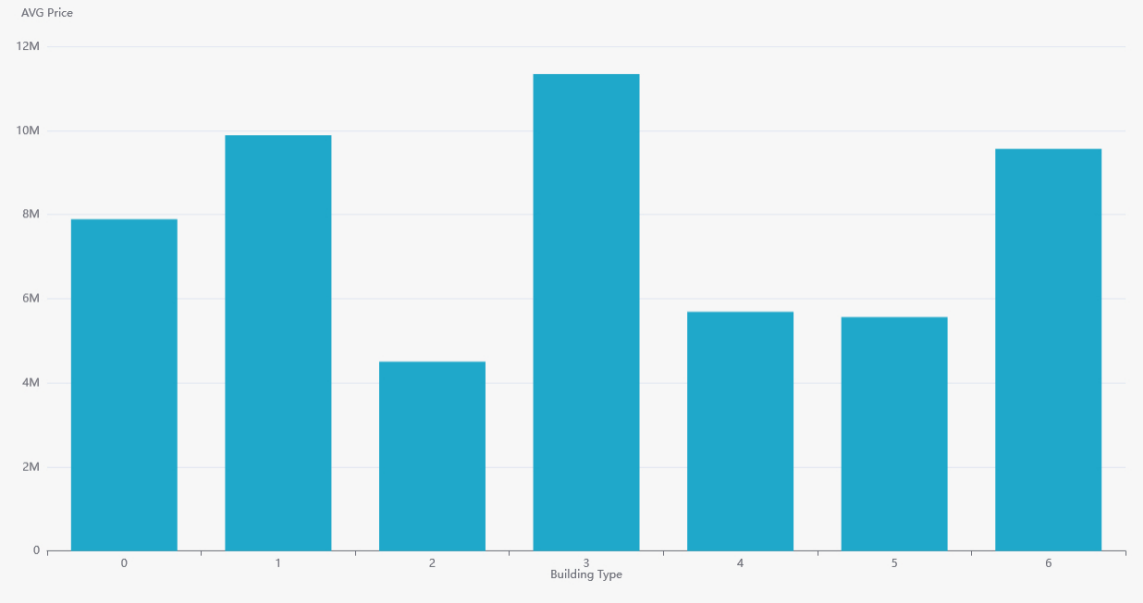

Figure 7: Average Price by Building Material

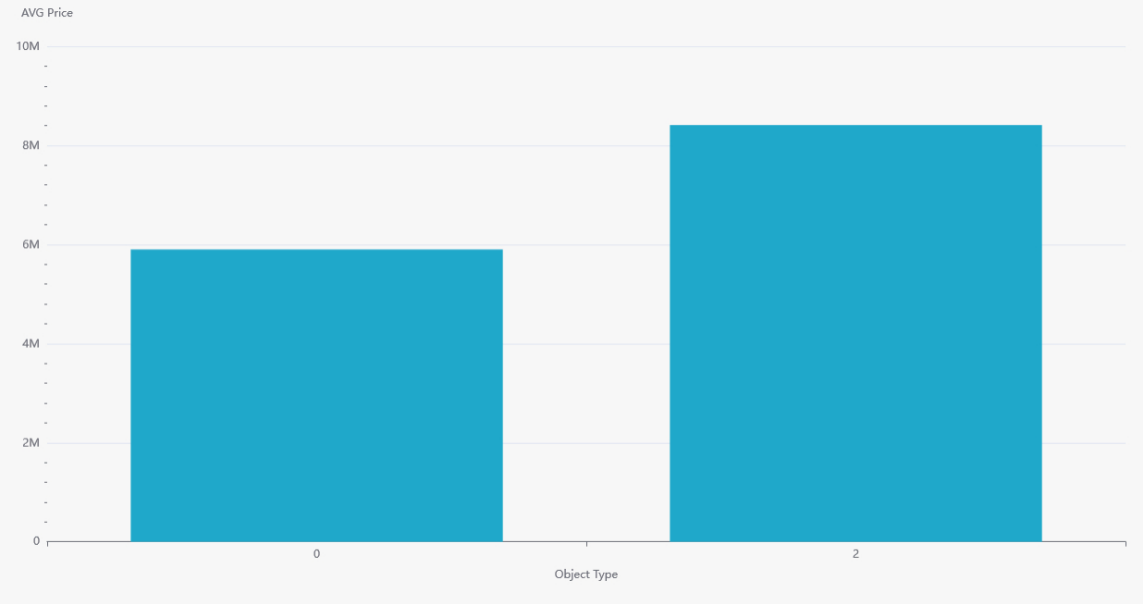

Figure 8: Average Price by Property Type

The analysis of the data suggests several key insights into the factors influencing real estate pricing. These insights provide valuable information for stakeholders in the real estate market, enabling them to make informed decisions regarding property valuation, investment, and development.

6. ML Modeling

Data preprocessing, feature extraction

In the beginning of the stage we had a table with houses and a table with real estate listings. The data had more than 8M instances, which turned out to load the cluster too significantly. Therefore, it was decided to shrink the task to predicting the prices of real estates in the Republic of Tatarstan(region id 16). Then, we joined the tables getting a single dataframe. Feature "house_id" was dropped, as it is not present in the dataset and has no meaning by itself, it was only needed as a joint point. "id_region" feature was dropped as well, as after filtering the data it has only one value. Then, the data was cleaned in the following ways:

- the rows with null values were dropped
- the duplicate rows were dropped
- special numbers (-100 for kitchen_area and -1 for rooms) for studio-flats were substituted with zeros
- the rows with inadequate prices were dropped. We dropped listings with price less than 500,000 because even the smallest studio-flats in small cities cannot have such a small cost, so the listings are probably something else (or incorrectly filled). We also did not predict the rows with price greater than 100,000,000, because these are some highly elite real estates, and the price for them is not determined by simple area or features of the dataset, but some other factors. So, these prices will only confuse the model.

The number of samples left in the dataset after cleaning and sampling is 161,425, which still satisfies the requirements. After that, I worked with columns:

- the categorical features (object type and building type) were encoded using the OneHotEncoder. We did not count street_id and postal_code were not treated as categorical (even though they are), as there are way too many values in them. Yet, they were not dropped in case Tree-based techniques can handle some important info from those features.
- the geo features were translated to ECEF coordinates and treated as numerical features
- the date feature was translated to month (sin and cos) and day (sin and cos). Year is the same for all listings, time is not present in the dataset

- the features were assembled into a single vector and scaled using the RobustScaler.
- the final feature list was indexed.

The final dataset with 22 features was split randomly into training and testing subsets with 70% of training data and 30% of testing data.

Training and hyperparameter tuning

For the task of predicting the real estate prices we have chosen two types of models:

- Linear Regression
- Random Forest Regressor

First we trained the models we default set of hyperparameters and received the following results:

Table 1: Performance Metrics for base models

Model	R-squared	MAE	RMSE
Linear Regression	0.627062	1,020,680	2,090,230
Random Forest Regressor	0.681059	934,979	1,889,344

Then, we have run Grid search with Cross Validation factor 3 on the following sets of hyperparameters:
for Linear Regression:

- fitIntercept (True, False). Whether to fit slope + intercept, or just slope
- aggregationDepth (2, 3, 4). The number of terms gathered

for Random Forest:

- numTrees (25, 50). Number of decesion trees trained
- maxDepth (5, 10). Maximal depth of each tree

Evaluation

Here are the parameters and results of the best models:

Table 2: Performance Metrics for best models

Model	Parameter 1	Parameter 2	R-squared	MAE	RMSE
Linear Regression	fitIntercept=True	aggregationDepth=3	0.651599	994,592	1,976,613
Random Forest	numTrees=50	maxDepth=10	0.822866	677,721	1,409,394

Random Forest Regression has shown quite good results. Even though they may be not good enough to use the model in production, the results are still promising. Additional data cleaning and increasing models' complexities (which was limited due to cluster recources limitaition) may lead to a result that could be deployed and could be useful for predicting the real life real estate prices.

7. Data Presentation

The dashboard description

Our dashboard consists of the several tabs each include information about different parts of our project.

Data description tab

This tab include the description of the dataset characteristics (features, size of dataset), provides schema of the dataset and demonstrates some samples from the resulting database tables.

Content of tab:

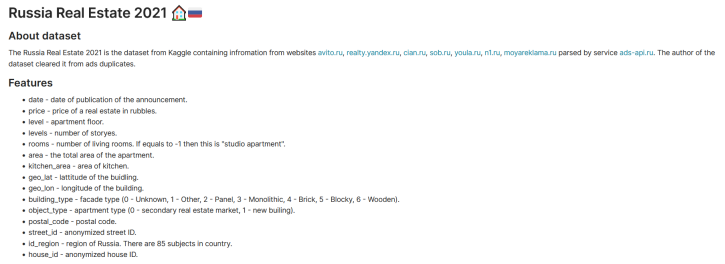

Figure 9: Description of dataset

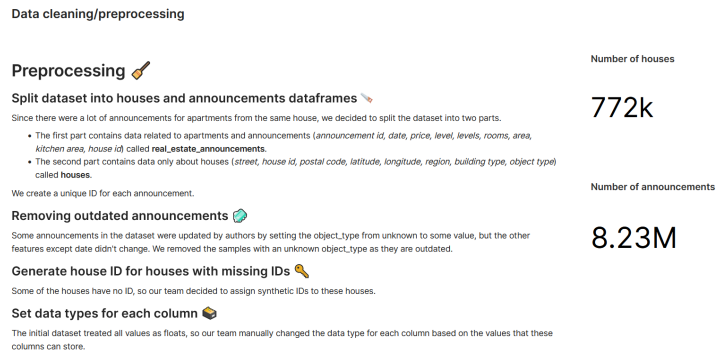

Figure 10: Description of preprocessing

Data types

Real estate announcements

column_name	data_type
announcement_id	integer
area	numeric
house_id	integer
kitchen_area	numeric
level	smallint
levels	smallint
price	bigint
publication_date	date
rooms	smallint

Houses

column_name	data_type
house_id	integer
id_region	smallint
street_id	integer
postal_code	integer
building_type	smallint
geo_lon	double precision
geo_lat	double precision
object_type	smallint

Figure 11: Datatypes

Dataset samples

Real estate announcements

announcement_id	publication_date	price	level	levels	rooms	area	kitchen_area	house_id
6252341	2021-07-27	3420000	12	18	2	48.77	0	3193786
960775	2021-02-12	2737900	20	25	1	40.8	8	3189281
892283	2021-02-10	13080000	22	24	2	56	9.5	2298254
9687846	2021-11-07	7200000	10	24	1	38.7	0	1239632
4322819	2021-05-28	3950000	2	5	3	57.1	5.6	3082864
1578609	2021-03-01	1950000	4	5	1	31.1	5.5	762392
2760081	2021-04-06	3767400	6	24	1	27.3	13.9	2381326
4018599	2021-05-18	4200000	6	14	1	42	-100	2089648
11081830	2021-12-18	5490000	2	45	-1	31	-100	831406
4403207	2021-05-31	1940000	5	5	2	43.3	5.6	2951090

Houses

house_id	id_region	street_id	postal_code	building_type	geo_lon	geo_lat	object_type
1549541	23	232513	352630	4	39.865139	44.767944	0
1549548	33	355229	602267	4	42.0430485	55.57568739999999	0
1549550	23	330667	353445	5	37.3326501	44.892316799999996	0
1549553	10	128960	185034	4	34.4478109	61.754488	0
1549555	11	451890	167019	6	50.8358151	61.66866170000001	0
1549567	18	435328	427011	4	53.31147320000001	56.91720920000001	0
1549570	47	378113	188683	0	30.698532	59.785776	0
1549582	36	498038	396005	4	39.189542700000004	51.7926023	2
1549588	16	563306	420004	1	49.0292403	55.8239975	0
1549596	74	327619	456316	0	60.1230946	55.0656201	0

Figure 12: Samples

Data insights tab

This part of the dashboard consists of plot charts with text justification (Figure 14).

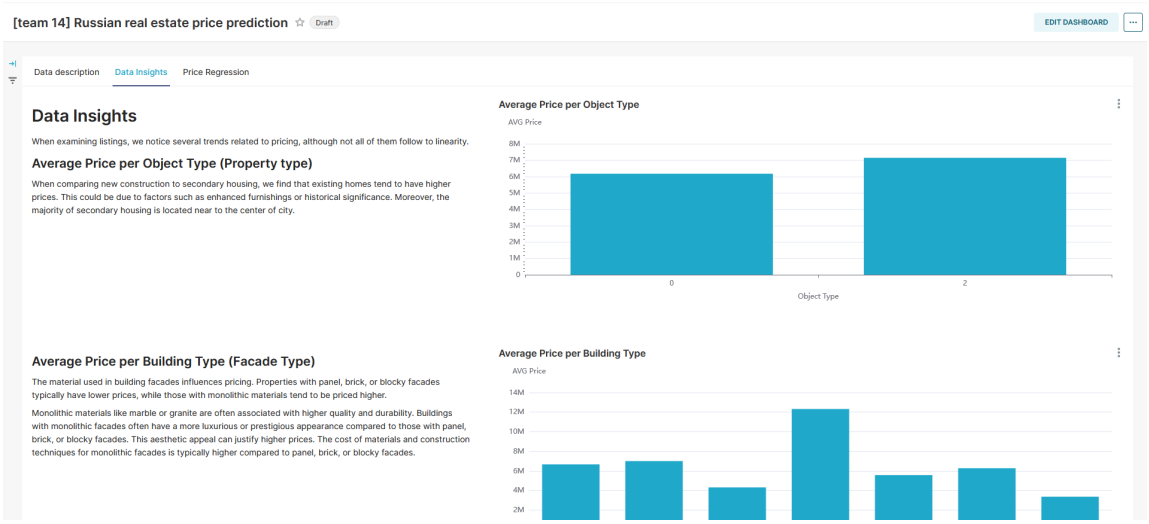

Figure 13: Part of "Data Insights" tab.

Price Regression

In this tab we present the data after feature extraction and preprocessing, present the metrics of the models, their parameters and sample predictions along with textual comments. Here is a sample picture of the dashboard (detailed description of all the pictures is given in the next subsection)

Figure 14: Part of ”Price regression” tab.

The charts description

Charts for data description

Figure 15: Chart that shows how much houses (samples) we have in **houses** table

Figure 16: Chart that shows how much announcements (samples) we have in **real_estate_announcements** table.

Real estate announcements

column_name	data_type
announcement_id	integer
area	numeric
house_id	integer
kitchen_area	numeric
level	smallint
levels	smallint
price	bigint
publication_date	date
rooms	smallint

Figure 17: Chart that describes each column of **real_estate_announcements** table.

Houses

column_name	data_type
house_id	integer
id_region	smallint
street_id	integer
postal_code	integer
building_type	smallint
geo_lon	double precision
geo_lat	double precision
object_type	smallint

Figure 18: Chart that describes each column of **houses** table.

announcement_id	publication_date	price	level	levels	rooms	area	kitchen_area	house_id
6252341	2021-07-27	3420000	12	18	2	48.77	0	3193786
960775	2021-02-12	2737900	20	25	1	40.8	8	3189281
892283	2021-02-10	13080000	22	24	2	56	9.5	2298254
9687846	2021-11-07	7200000	10	24	1	38.7	0	1239632
4322819	2021-05-28	3950000	2	5	3	57.1	5.6	3082864
1578609	2021-03-01	1950000	4	5	1	31.1	5.5	762392
2760081	2021-04-06	3767400	6	24	1	27.3	13.9	2381326
4018599	2021-05-18	4200000	6	14	1	42	-100	2089648
11081830	2021-12-18	5490000	2	45	-1	31	-100	831406
4403207	2021-05-31	1940000	5	5	2	43.3	5.6	2951090

Figure 19: Chart that shows samples from **real_estate_announcements** table.

house_id	id_region	street_id	postal_code	building_type	geo_lon	geo_lat	object_type
1549541	23	232513	352630	4	39.865139	44.767944	0
1549548	33	355229	602267	4	42.0430485	55.57568739999999	0
1549550	23	330667	353445	5	37.3326501	44.892316799999996	0
1549553	10	128960	185034	4	34.4478109	61.754488	0
1549555	11	451890	167019	6	50.8358151	61.66866170000001	0
1549567	18	435328	427011	4	53.31147320000001	56.91720920000001	0
1549570	47	378113	188683	0	30.698532	59.785776	0
1549582	36	498038	396005	4	39.189542700000004	51.7926023	2
1549588	16	563306	420004	1	49.0292403	55.8239975	0
1549596	74	327619	456316	0	60.1230946	55.0656201	0

Figure 20: Chart that shows samples from **houses** table.

Charts for data insights

Data Insights

When examining listings, we notice several trends related to pricing, although not all of them follow to linearity.

Average Price per Object Type (Property type)

When comparing new construction to secondary housing, we find that existing homes tend to have higher prices. This could be due to factors such as enhanced furnishings or historical significance. Moreover, the majority of secondary housing is located near to the center of city.

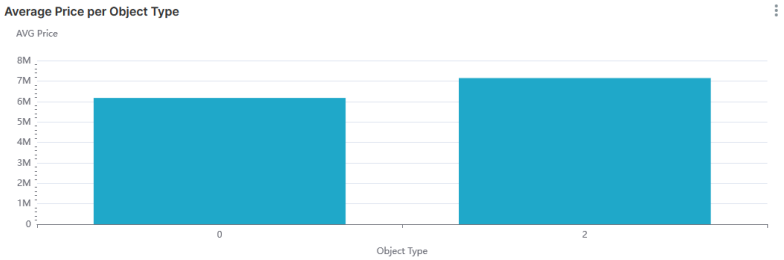

Figure 21: Average Price per Object Type Pair of plot and justification.

Average Price per Building Type (Facade Type)

The material used in building facades influences pricing. Properties with panel, brick, or blocky facades typically have lower prices, while those with monolithic materials tend to be priced higher.

Monolithic materials like marble or granite are often associated with higher quality and durability. Buildings with monolithic facades often have a more luxurious or prestigious appearance compared to those with panel, brick, or blocky facades. This aesthetic appeal can justify higher prices. The cost of materials and construction techniques for monolithic facades is typically higher compared to panel, brick, or blocky facades.

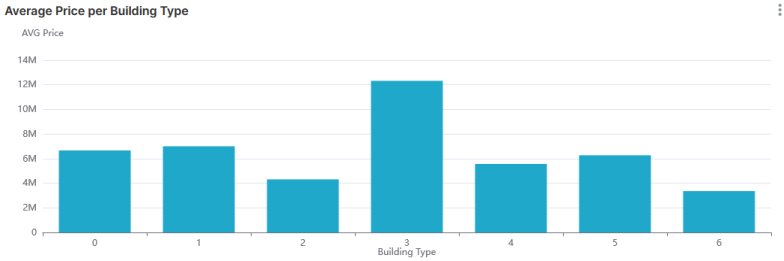

Figure 22: Average Price per Building Type Pair of plot and justification.

Average Price per Area

When analyzing the relationship between property area and price, we observe that there isn't a consistent linear correlation where larger properties always command higher prices. However, there is a discernible overall trend of prices increasing with larger property sizes.

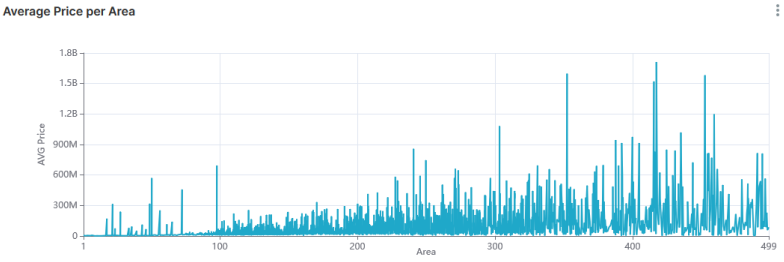

Figure 23: Average Price per Area Pair of plot and justification.

Average Price per Number of Rooms

Analyzing the relationship between the number of rooms in a property and its price reveals a positive correlation. Generally, as the number of rooms increases, so does the price. However, an intriguing anomaly arises at the maximum room count, where we observe a dip in the average price.

This suggests that while more rooms typically lead to higher prices due to increased living space and utility, there are exceptions. Properties with an unusually high number of rooms may possess unique characteristics that affect their market value. For instance, these properties might be classified as dormitories or communal housing arrangements, which could impact their desirability and consequently their pricing. This anomaly underscores the importance of considering property characteristics beyond just the number of rooms when evaluating pricing trends.

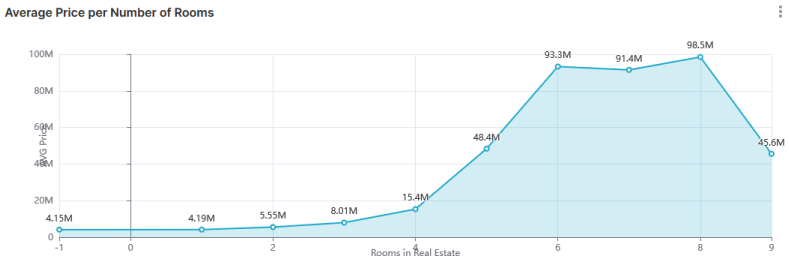

Figure 24: Average Price per Number of Rooms Pair of plot and justification.

Floor Level to Average Price

Examining the impact of floor level on real estate pricing reveals a clear pattern: prices tend to escalate with higher floor levels. Several factors contribute to this phenomenon.

Firstly, high-rise apartment complexes, commonly found in upscale neighborhoods, typically demand higher prices. These areas often boast luxurious amenities and desirable surroundings, which attract affluent buyers willing to pay a premium for properties on higher floors.

Moreover, taller buildings often provide superior views, whether of city skylines, scenic landscapes, or bodies of water. These panoramic vistas enhance the appeal of properties on higher floors, further driving up their market value.

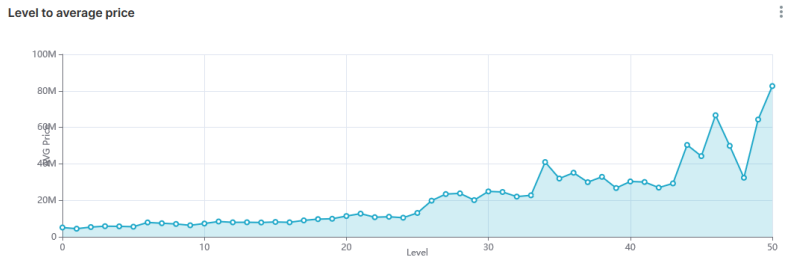

Figure 25: Average Price per Average Price Pair of plot and justification.

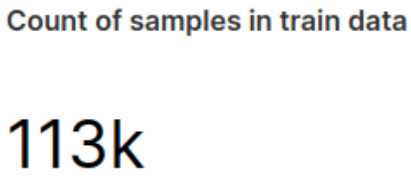

Figure 26: Number of rows in training data.

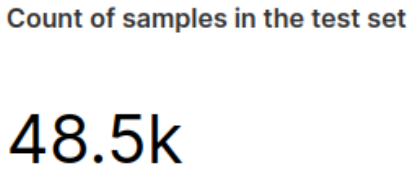

Figure 27: Number of rows in testing data.

Train dataset sample

type	vector_values
1	[0.5057421818987656,133.52961562696913,0.2,1.8,1.0,1.2890625,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.8660254037844377,-0.28867513459481325,0.0735146600458892,-0.687194940122474,21.42372
1	[0.5057421818987656,133.52961562696913,0.2,1.8,1.0,1.2890625,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-2.4492935982947044E-16,0.577350269189626,-0.35264802355333374,-0.6039452181310054,21.4
1	[0.5057421818987656,133.52961562696913,0.2,2.4000000000000004,1.0,1.4140625,0.87,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.9999999999999991,3.535250795749691E-17,0.41511599483613687,0.5669332
1	[0.5057421818987656,133.52961562696913,0.2,2.4000000000000004,1.0,1.421875,0.87,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.4999999999999993,-0.5000000000000002,0.5746226519476533,-0.4228054
1	[0.5057421818987656,133.52961562696913,0.2,2.8000000000000003,1.0,1.3671875,0.8500000000000001,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.4999999999999993,-0.5000000000000002,-0.4733243142
1	[0.5057421818987656,133.52961562696913,0.2,2.8000000000000003,1.0,1.40625,1.1,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-2.4492935982947044E-16,0.577350269189626,0.41511599483613687,0.566933234
1	[0.5057421818987656,133.52961562696913,0.2,2.8000000000000003,1.0,1.40625,1.1,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.8660254037844379,-0.28867513459481287,-0.14627496019847797,0.6765994322
1	[0.5057421818987656,133.52961562696913,0.4,1.8,3.0,1.9921875,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.8660254037844378,0.2886751345948130 3,0.1462749601984778,0.6765994322070602,21.423725
1	[0.5057421818987656,133.52961562696913,0.4,1.8,3.0,2.265625,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.4999999999999999,-0.5000000000000001,-0.5746226519476534,-0.422805411861869,21.423725
1	[0.5057421818987656,133.52961562696913,0.4,1.8,3.0,2.265625,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.8660254037844378,0.2886751345948130 3,-0.526675685729772,0.4758962504607053 3,21.42372
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,2.0,2.03125,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.9999999999999991,3.535250795749691E-17,-0.21753428131778277,-0.6590610
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,5.0,4.0625,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.9999999999999991,-1.0605752387249072E-16,-0.526675685729772,0.4758962
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,5.0,4.0625,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.4999999999999995,0.5000000000000002,-0.7182773362410615,0.1045970561
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,5.0,4.0625,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.4999999999999999,-0.5000000000000001,0.3526480235533339,-0.6039452181
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,5.0,4.0625,0.86,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.4999999999999993,-0.5000000000000002,-0.14627496019847797,0.6765994
1	[0.5057421818987656,133.52961562696913,0.4,2.4000000000000004,5.0,4.0625,0.86,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.4999999999999999,-0.5000000000000001,-0.07351466004588904,-0.6871949
1	[0.5057421818987656,133.52961562696913,0.4,2.8000000000000003,3.0,2.578125,0.9500000000000001,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.4999999999999993,-0.5000000000000002,-0.4733243142

Figure 28: Sample rows from the training data after feature extraction.

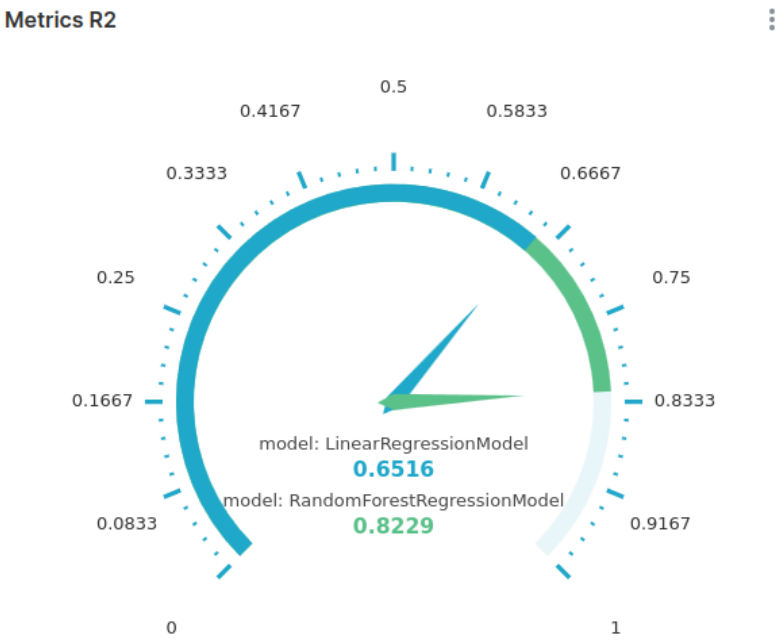

Figure 29: Comparison of R-squared scores of the best models.

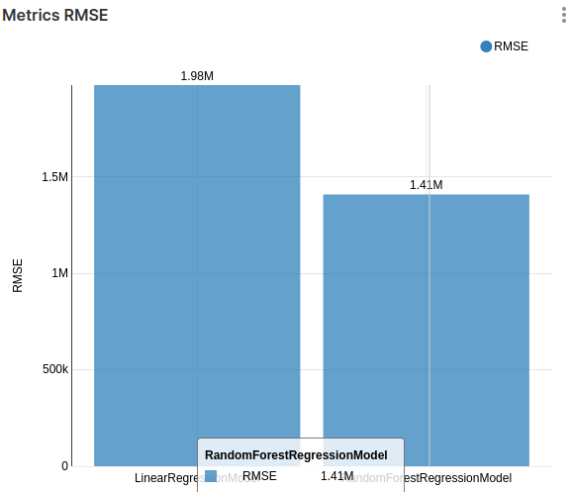

Figure 30: Comparison of RMSE metrics of the best models.

Best models' characteristics

Model	Parameter1	Parameter2	Number of features
LinearRegression	fitIntercept=True	aggregationDepth=3	numFeatures=22
RandomForestRegression	maxDepth=10	numTrees=50	numFeatures=22

Figure 31: Hyperparameters and number of features of the best models.

Model1 Predictions

label	prediction	Error	Relative Error
4084430	4428284.5	-343854.5	-0.08418665517587522
2558646	2483824.2	74821.75	0.029242712747288995
4700000	4777800.5	-77800.5	-0.016553297872340427
3550080	4694266.5	-1144186.5	-0.32229879326663063
3550080	4791486.5	-1241406.5	-0.3496840916261042
2877630	4311941	-1434311	-0.4984348231009546
3100000	4421005.5	-1321005.5	-0.4261308064516129
2150000	1447670	702330	0.32666511627906974
2300000	1667053.5	632946.5	0.27519413043478264
2700000	5946609	-3246609	-1.2024477777777778
4300000	4575540.5	-275540.5	-0.06407918604651162

Figure 32: Comparison of best model 1 predictions with actual labels.

Model2 Predictions

label	prediction	Error	Relative Error
4084430	4438417	-353987	-0.08666741748542636
2558646	2297362	261284	0.10211807338725247
4700000	4351517.5	348482.5	0.07414521276595745
3550080	4219703	-669623	-0.18862194654768344
3550080	4398986	-848906	-0.23912306201550387
2877630	3391484	-513854	-0.17856847475179227
3100000	3391484	-291484	-0.09402709677419355
2150000	2252841.8	-102841.75	-0.047833372093023255
2300000	2256082.8	43917.25	0.01909445652173913
2700000	5569605.5	-2869605.5	-1.062816851851852
4300000	4380741.5	-80741.5	-0.018777093023255816

Figure 33: Comparison of best model 2 predictions with actual labels.

Our Findings

Our analysis has uncovered several key findings:

- When a user views a listing featuring specific attributes (e.g., 3 rooms with an area of 50 square meters) and the property has a monolithic facade, we can recommend similar listings with panel or brick facades. Our data indicates that listings with panel and brick facades tend to have lower prices. This recommendation could potentially help users in finding more affordable options that match their criteria.
- In cases where a user is searching for properties with a specific number of rooms, such as 5 rooms, we can suggest exploring listings with a higher number of rooms, such as 9 rooms. Surprisingly, our analysis reveals that these larger properties often have similar prices to those with fewer rooms. This insight could provide users with the opportunity to explore larger properties within their budget constraints.
- Real estate listings of premium segment does not follow the same price formation rules as the usual real estates.

These findings underscore the value of data insights to enhance user experiences and help to make decisions in the real estate market.

8. Conclusion

In conclusion, our project journey involved setting clear business goals, sourcing relevant data, and successfully constructing a comprehensive data processing ML pipeline within the cluster. Initially, we uploaded the selected dataset in compressed format to the University Hadoop cluster. Subsequently, we conducted thorough analysis, extracting five key insights regarding the relationships between various features and pricing. Furthermore, we loaded this dataset into Hive, optimizing it through partitioning and bucketing techniques. Notably, we also stored the insights we discovered in Hive for future reference. Additionally, we preprocessed the data and developed two machine learning models using the Apache Spark framework with the Python library, PySpark. We further refined our models through hyperparameter tuning and rigorous evaluation. Finally, we leveraged Apache Superset to craft a comprehensive dashboard showcasing different facets of our project and the attained results. Along the way, we encountered challenges such as lengthy queue times for Spark applications and managing large dataset sizes. However, we navigated these challenges, gaining valuable experience in cluster environments. Despite the hurdles, we achieved satisfactory metrics, with an R^2 exceeding 0.8 for the second model, marking a successful outcome.

9. Reflections on Own Work

Regression model score

The main challenge of our project was to train a regression model. The problem that we encounterted was that none of the models we tried has shown somehow sufficient results (R-squared greater than 0). Correlations between the price and some obviously important features like area was almost 0 as well. We tried several solutions, but the one that helped was to remove the outliers. We removed listings with too low price (less than 500,000 which seemed to be unreal in the modern world) and listings with too high prices (greater than 100,000,000, that is a price of premium class estate, the price for which depends on some other factors than just simply area or level). It helped us to achieve sufficient results.

Problems with cluster

During the second stage we faced problem with Apache Superset. Their engine has poor performance. When we created dashboard with data insights, we added there 5 charts. Such action crashed the whole Superset server. The solution was to cache charts and use their saved versions.

Cluster problems:

- Large waiting time for the recources. Solution: wait more
- Uncontrolled recource consumption by grid search. Solution: prohibit dynamic recource allocation
- Insufficient recources to train the model on the whole data. Solution: limit training data to a single region (Republic of Tatarstan).
- Large waiting time for grid search (which prevented other people to use the cluster). Solution: train at night, decrease model complexity.

9.1 Recommendations

- Acquire more features, extend dataset (pictures, text description of listing)
- Increase model complexity
- Do additional data cleaning paying attention to more details

- Try more different models, try using a DL model.

9.2 The Table of Contributions of Each Team Member

Table 3: The Table of Contributions of Each Team Member

Project tasks	**Task description**	**Nagim Isyanbaev**	**Viktor Kovalev**	**Kirill Batyschev**	**Deliverables**	**Actual Hours Spent**
Complete stage 1 of project	Preprocess the dataset for a pipeline and complete stage 1 by following the instructions for stage 1. Create github and start report	100%	0%	0%	houses.java, houses.avsc, real_estate_ announcements.java, real_estate_ announcements.avsc, stage1.sh	9
Complete stage 2 of project	Create hive tables from the data, partition and bucket them, perform EDA (find 5 different insights)	0%	100%	0%	stage2.sh, 2 Hive tables, 5 charts in Superset	10
Complete stage 3 of project	Do feature extraction and data cleaning. Train two models. Tune the hyperparameters of the two models. Evaluate the models	0%	0%	100%	Two trained models. Hive and csv/json tables with train and test data, predicitons of best models, evaluation of the best models	15 active, 40 busy waiting
Complete stage 4 of project	Design a Superset dashboard	33%	34%	33%	Dashboard in Superset	5
Write a report	Complete the report that describes the project by following the given report structure	34%	33%	33%	Report in pdf format	12
Check with pylint	Fix the warnings highlighted by pylint	33%	33%	34%	Cleaned python code	3

References

[1] Daniil Agniashvili. Russia real estate 2021, 2021. URL https://www.kaggle.com/datasets/mrdaniilak/russia-real-estate-2021.