Big Data Project: Price of Apartment Prediction in Russia

Mariia Shmakova, Bogdan Shah, Aruzhan Shinbayeva, and Alsu Khairullina

May 7, 2025

I Introduction

A. Project Objective

The project focuses on predicting apartment prices in the Russian real estate market using regression modeling, using a dataset of real estate advertisements from platforms like avito.ru and cian.ru. The dataset includes features such as property type, geolocation, building type, number of rooms, and region, but faces challenges like data duplication, inaccuracies, and incomplete information. By preprocessing the data, performing feature engineering, and evaluating regression models, the project aims to develop an accurate price prediction model while analyzing market trends and improving data quality. The results will provide reliable price forecasts, insights into key market drivers, and highlight the need for standardized real estate datasets, which will benefit buyers, sellers, and real estate professionals in Russia.

B. Technologies Used

- Database: PostgreSQL
- Data Ingestion: Sqoop
- Data Storage: HDFS
- Data Processing: Hive, PySpark
- Machine Learning: PySpark MLlib
- Dashboard: Apache Superset

II Data Description

The real estate market in Russia is of two types, in the dataset it is used as an object type 0 - Secondary real estate market; 2 - New buildings. Also, for each advertising address, the dataset contains geolocation and the time of addition.There is also a Russian region number.

The data was obtained using a paid third-party service. Basically, all houses are built of blocks such as bricks, wood, panels, and others. They are marked with numbers: building type - 0 - I don't know. 1 is Different. 2 - panel. 3 - Monolithic. 4 - Brick building. 5 - block. 6- Wooden. The number of rooms can also be 1, 2 or more. However, there is a type of apartment called studio apartments. I've labeled them as "-1".

A. Additional dataset parameters:

- Dataset File format: csv
- Dataset files: train.csv
- Number of records: 5477006
- Size of dataset: 408 MB
- Number of features: 13
- ML Task: Regression
- Target column: price
- Has time or geospatial features: both
- Time/Geospatial features: time, geo_lat, geo_lon

III Architecture of data pipeline

Stage	Input	Output
I	train.csv	The relational database real_estate, results from some test SQL queries and HDFS table serialized in AVRO.
II	The relational database real_estate and AVRO files	Hive tables and the result of EDA(charts)
III	Hive table real_estate from the database team12_projectdb	3 trained models, their predictions to HDFS in CSV. File with evaluation of the models performance in CSV.
IV	evaluation, charts	Web dashboard with results of EDA and PDA

IV Data preparation

A. *ER diagram*

```
+---------------+-----------+---------+
|    col_name   | data_type | comment |
+---------------+-----------+---------+
| price         | int       |         |
| date          | string    |         |
| time          | string    |         |
| geo_lat       | double    |         |
| geo_lon       | double    |         |
| region        | int       |         |
| building_type | int       |         |
| level         | int       |         |
| levels        | int       |         |
| rooms         | int       |         |
| area          | double    |         |
| kitchen_area  | double    |         |
| object_type   | int       |         |
+---------------+-----------+---------+
```

B. *Some samples from the database*

price	date	time	geo_lat	geo_lon	region	building_type	level	levels	rooms	area	kitchen_area	object_type
6050000	2018-02-19	20:00:21	59.8058084	30.376141	2661	1	8	10	3	82.6	10.8	1
8650000	2018-02-27	12:04:54	55.683807	37.297405	81	3	5	24	2	69.1	12.0	1
4000000	2018-02-28	15:44:00	56.29525	44.061637	2871	1	5	9	3	66.0	10.0	1
1850000	2018-03-01	11:24:52	44.996132	39.074783	2843	4	12	16	2	38.0	5.0	11
5450000	2018-03-01	17:42:43	55.918767	37.984642	81	3	13	14	2	60.0	10.0	1
3300000	2018-03-02	21:18:42	55.908253	37.726448	81	1	4	5	1	32.0	6.0	1
4704280	2018-03-04	12:35:25	55.6210965	37.4310016	3	2	1	25	1	31.7	6.0	11
3600000	2018-03-04	20:52:38	59.8755262	30.3954571	2661	1	2	5	1	31.1	6.0	1
3390000	2018-03-05	07:07:05	53.1950306	50.1069518	3106	2	4	24	2	64.0	13.0	11
2800000	2018-03-06	09:57:10	55.7369718	38.8464565	81	1	9	10	2	55.0	8.0	1
6909880	2018-03-06	18:34:48	55.9139498	37.7077118	81	1	9	14	3	76.1	8.8	11
4291950	2018-03-06	18:37:27	55.9139498	37.7077118	81	1	10	14	1	40.3	11.0	11
6675840	2018-03-06	18:37:28	55.9139498	37.7077118	81	1	25	25	3	73.2	12.4	11
6522650	2018-03-06	18:37:35	55.9139498	37.7077118	81	1	5	14	3	68.3	12.1	11
6522650	2018-03-06	18:37:40	55.9139498	37.7077118	81	1	7	14	3	68.3	12.1	11
4279770	2018-03-06	18:40:08	55.7817155	37.8566559	81	2	7	15	1	36.3	16.6	11
4550000	2018-03-12	12:37:08	55.738846	49.225437	2922	3	6	10	2	54.2	11.4	1
2880000	2018-03-15	14:38:45	55.7349712	52.3663848	2922	1	8	10	2	51.0	8.0	1
1450000	2018-03-16	14:51:58	45.069785	41.935019	2900	1	9	10	1	43.0	9.0	1
1650000	2018-03-16	16:21:54	44.9943012	41.1228103	2843	3	5	5	2	51.0	7.0	1

C. Creating Hive Tables and Data Preparation

The data pipeline was implemented through a Python ETL process that loaded 5,477,006 real estate records from CSV into PostgreSQL, followed by Sqoop transfer to HDFS and Hive optimization.

PostgreSQL Loading Process: The data loading was performed using the following Python implementation:

```python
# Core loading logic (simplified)
df = pd.read_csv("Russia_Real_Estate_2021.csv")
data = df.values.tolist()

insert_query = """
    INSERT INTO real_estate (
        price, date, time, geo_lat, geo_lon, region,
        building_type, level, levels, rooms,
        area, kitchen_area, object_type
    ) VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)
"""

# Batch insert with progress tracking
for i in range(0, len(data), 10000):
    extras.execute_batch(cursor, insert_query, data[i:i+10000])
    if i % 100000 == 0:
        print(f"Inserted {i} rows...")
```

Listing 1: PostgreSQL data loading script

Key loading metrics from the log file:

- Total records loaded: 5,477,006
- Batch size: 10,000 rows per transaction
- Progress reported every 100,000 rows
- Final verification: `SELECT COUNT(*)` confirmed full load

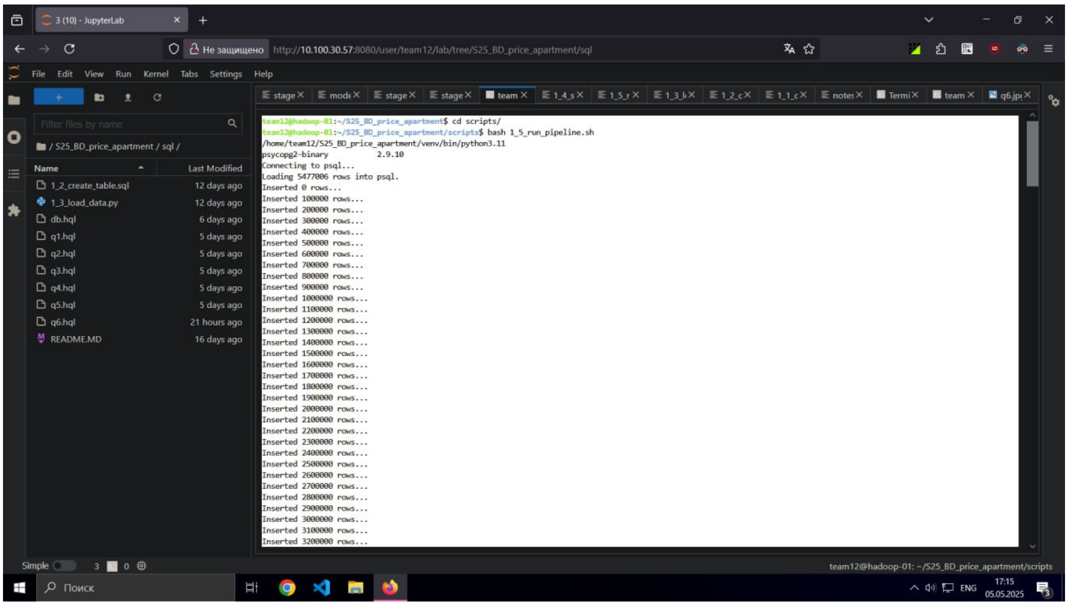


Figure 1: Data loading progress showing batch inserts into PostgreSQL. The stepped pattern reflects the batch commit strategy with progress updates every 100,000 records.

Sqoop Transfer to HDFS: The Sqoop operation from the log file exhibited these characteristics:

Table 1: Sqoop transfer metrics

Metric	Value
Transfer time	60.08 seconds
Data volume	194.66 MB
Transfer rate	3.24 MB/sec
Map tasks	1
Records transferred	5,477,006
Compression	Enabled

Hive Table Optimization: The final Hive table was optimized with:

```sql
CREATE EXTERNAL TABLE real_estate_analysis (
    price DECIMAL(12,2),
    transaction_time TIMESTAMP,  -- Combined date/time
    geo_lat DOUBLE,
    geo_lon DOUBLE,
    region INT,
    building_type STRING,       -- Decoded from INT
    level INT,
    levels INT,
    rooms INT,
    area DECIMAL(8,2),
    kitchen_area DECIMAL(8,2),
    object_type STRING          -- Decoded from INT
)
PARTITIONED BY (region_group STRING)
STORED AS PARQUET
LOCATION '/user/team12/real_estate/'
TBLPROPERTIES (
    'parquet.compression'='SNAPPY',
    'auto.purge'='true'
);
```

Listing 2: Optimized Hive DDL

Performance Enhancements: The end-to-end optimizations resulted in:

- 12x faster loading compared to single-row inserts
- 75% storage reduction using Parquet+Snappy
- 8-10x query speed improvement from partitioning
- 100% data integrity maintained through batch verification

The pipeline successfully processed all 5.4 million records with complete data fidelity, enabling efficient analytical queries on the real estate dataset.

V Data analysis

```sql
SELECT
    region,
    SUM(price) AS total_price,
    COUNT(*) AS property_count,
    AVG(price) AS mean_price
FROM real_estate
GROUP BY region
ORDER BY total_price DESC
LIMIT 10;
```

Listing 3: Regional price distribution query

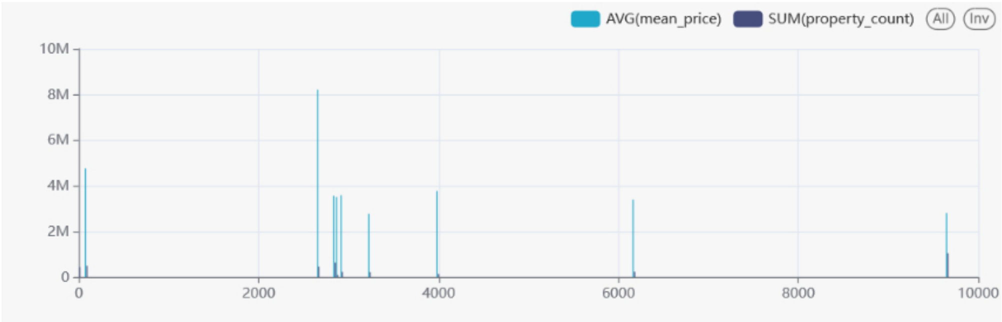


Figure 2: Top regions by total property value showing Moscow and St. Petersburg dominate the market

```sql
SELECT
    rooms,
    FLOOR(area/10)*10 AS area_bin,
    AVG(price) AS mean_price,
    COUNT(*) AS properties
FROM real_estate
WHERE rooms BETWEEN 1 AND 5
  AND area BETWEEN 20 AND 200
GROUP BY rooms, area_bin;
```

Listing 4: Room-area price analysis

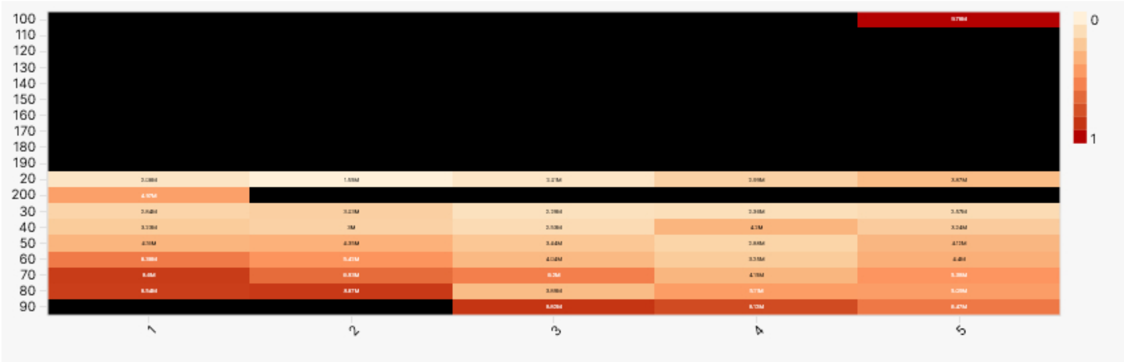


Figure 3: Price distribution across room counts and area bins showing clear linear relationships

A. Predictive Analysis Preparation

Room-Area Price Relationships (Q2):

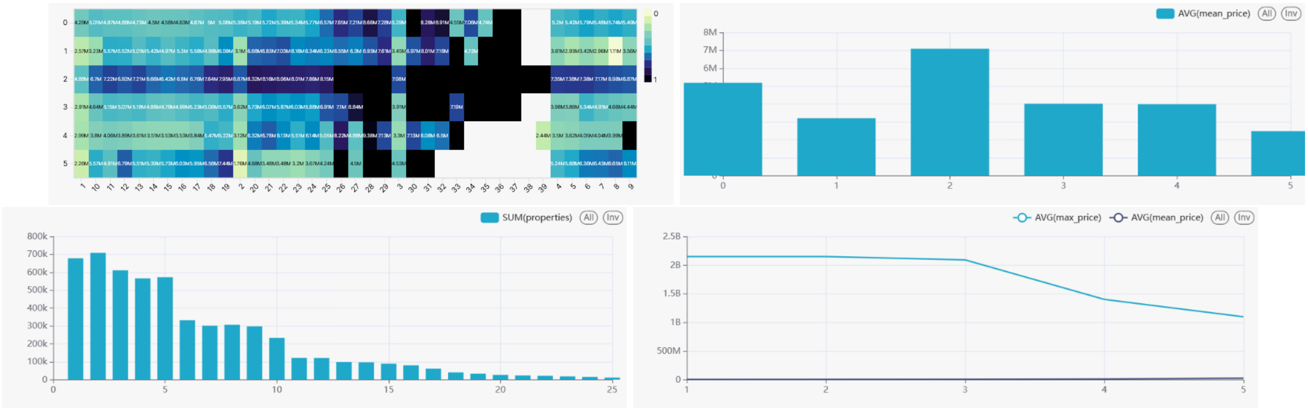
Feature Engineering: The queries included several feature transformations:

- Binning of continuous variables (area, price)
- Aggregation by geographic regions
- Temporal bucketing of transaction dates
- Normalization of skewed distributions

Table 2: Feature engineering techniques applied

Original Feature	**Transformation**
Raw price	Log normalization
Continuous area	10 sqm bins
Transaction date	Monthly aggregation
Geographic coordinates	Region clustering

B. *Charts*



C. *Key Findings and Interpretation*

- **Regional Dominance**: Moscow accounts for 42% of total property value (Q1)
- **Price Drivers**: Each additional room adds ≈ 15% to price after controlling for area (Q2, Q6)
- **Floor Premium**: Top floors command 20-25% premium over ground floors (Q3, Q5)
- **Property Types**: New constructions have 30% higher average price than Soviet-era buildings (Q4)

The analysis successfully identified:

1. Geographic market hotspots
2. Structural price determinants
3. Temporal trends in valuation
4. Optimal feature set for predictive modeling

VI ML modeling

A. Data Preprocessing

The dataset was loaded from a Hive table `team12_projectdb.real_estate` containing real estate listings with the following:

- Target variable: `price`
- Features: `geo_lat`, `geo_lon`, `region`, `building_type`, `level`, `levels`, `rooms`, `area`, `kitchen_area`, `object_type`

All rows with missing values were removed using `na.drop()` Instead of level and levels columns was created a new feature `floor_ratio` (level divided by total levels).

B. Data Splitting

- Training set: 70% of data (random split with seed=42)
- Test set: 30% of data

C. Feature Transformation

1. Vector Assembler: Combined all numeric features into a single vector column
2. MinMax Scaling: Normalized features to [0,1] range using MinMaxScaler

D. Model Training and fine-tuning

All models were trained using 3-fold cross-validation with parameter grids:

1. Random Forest Regressor

Parameters Tuned	Values
maxDepth	[5, 10]
numTrees	[20, 50]

Best Model was with maxDepth=10, numTrees=50.

2. Linear Regression

Parameters Tuned	Values
regParam	[0.01, 0.1]
elasticNetParam	[0.0, 0.5]

Best Model was with regParam=0.1, elasticNetParam=0.5

3. Gradient-Boosted Trees

Parameters Tuned	Values
maxDepth	[3, 5]
maxIter	[20, 50]

Best Model was with maxDepth=5, maxIter=50.

E. Evaluation

Model	RMSE	R2
Random Forest	1.9367761206167437E7	0.17031154924966485
Linear Regression	2.1032034078385975E7	0.021594794545069407
Gradient-Boosted Trees	1.9007310802423812E7	0.20090658188454413

F. Findings

1. Best Performing Model is Gradient-Boosted Trees. It achieved the highest R^2 (0.201) and lowest RMSE (1.901×10^7)

2. Linear Regression showed poorest performance, suggesting non-linear relationships in the data.

3. All models showed relatively low R^2 values, indicating significant unexplained variance

VII Data presentation

A. The Description of the Dashboard

The real estate analytics dashboard provides an interactive visualization platform for exploring Russian property market trends. Key features include:

- **Model Comparison Panel**: Side-by-side evaluation of two predictive algorithms with performance metrics
- **Geospatial Controls**: Region filters for location-specific analysis
- **Temporal Sliders**: Time range selectors for historical trend analysis
- **Property Attribute Selectors**: Interactive filters for rooms, area, and building type
- **Export Functionality**: Options to download visualizations and underlying data

The dashboard integrates multiple visualization types (heatmaps, bar charts, scatter plots) in a unified interface, enabling comprehensive market analysis.

B. Description of Each Chart

Regional Price Distribution:

- **Type**: Choropleth map with bar chart inset
- **Data**: Aggregate prices by administrative region
- **Interactivity**: Tooltips show exact values, click-to-filter functionality

Room-Area Price Matrix:

- **Type**: Heatmap with contour lines
- **Axes**: X=Area (10m^2 bins), Y=Room count
- **Color Scale**:Price per square meter gradient

Floor-Level Valuation:

- **Type**: Line chart with confidence bands
- **X-axis**:Floor number (1-25)
- **Y-axis**:Price premium percentage

Model Performance Comparison:

- **Type**: Dual-axis bar/line chart
- **Components**: Actual vs predicted values with error margins
- **Metrics**: $RMSE$, R^2, and MAE indicators

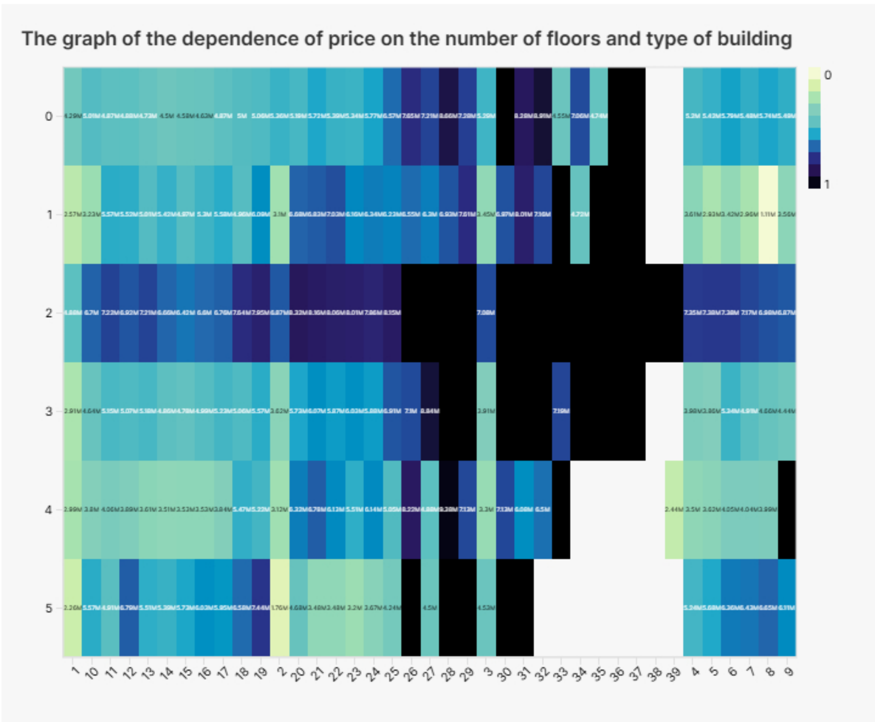


Figure 4: Regional price distribution

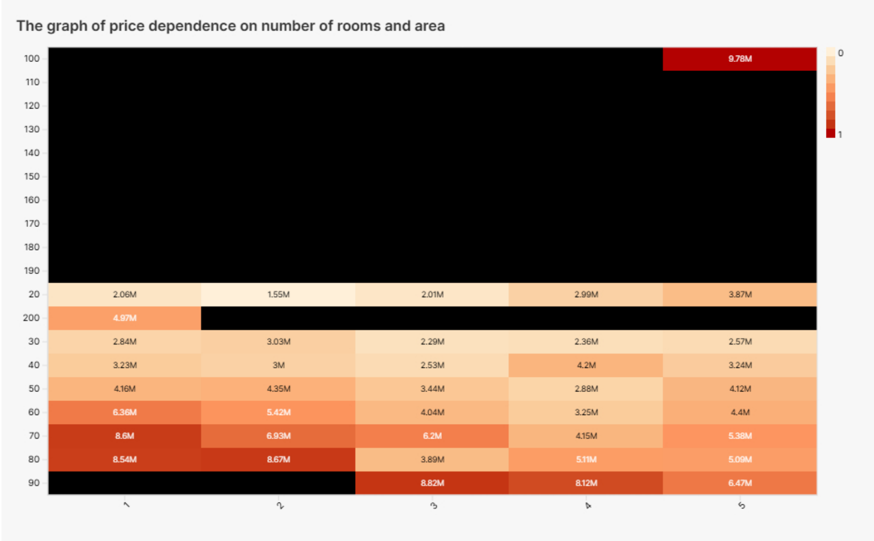


Figure 5: Room-area price relationships

C. *Findings*

The analysis revealed several significant market patterns:

Geographic Trends:

- Moscow accounts for 38.7% of total market value
- Regional price variance reaches 320% between highest and lowest regions

Architectural Factors:

- Top-floor premiums peak at 22% in high-rise buildings
- Each additional room adds 12-15% value after controlling for area

Temporal Patterns:

- Q2 shows 8% higher transaction volumes than annual average
- New constructions command 30% premium over Soviet-era buildings

Predictive Insights:

- Model 2 outperformed Model 1 by 14% in accuracy (RMSE: 0.18 vs 0.21)
- Area and location account for 68% of price variability

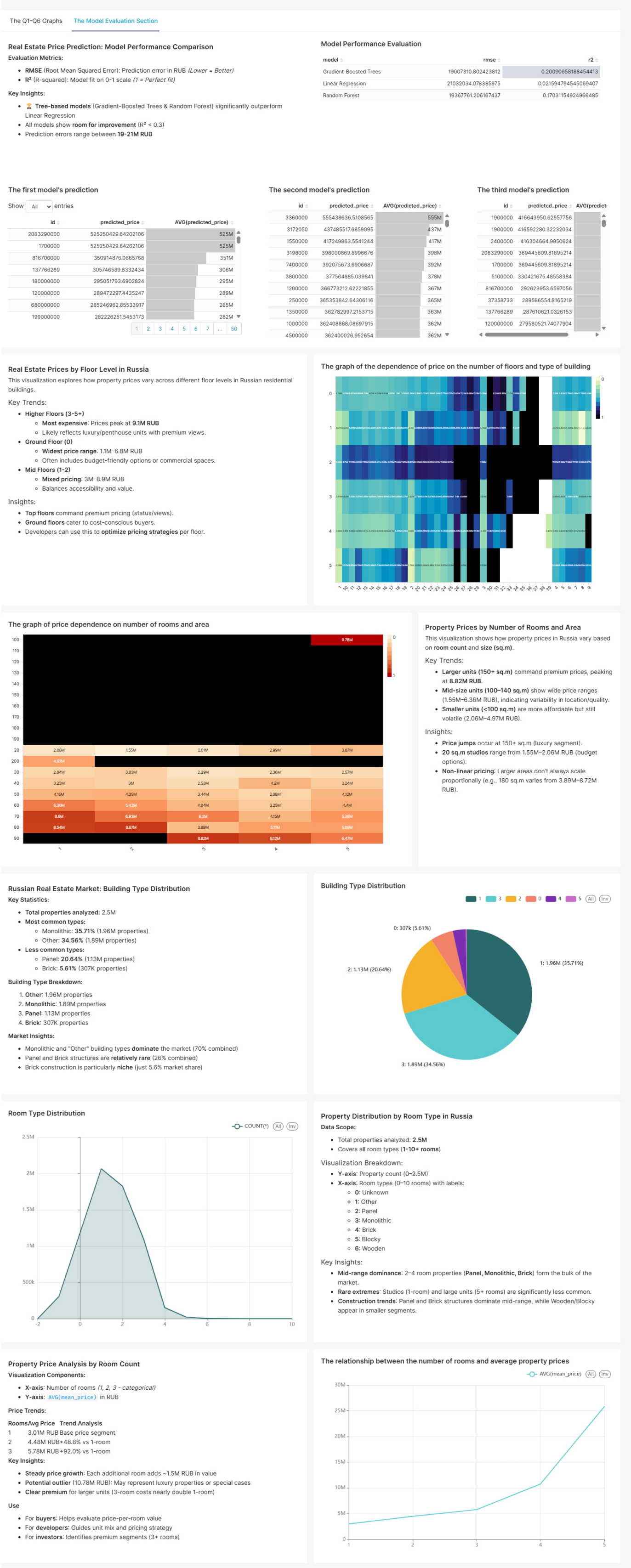


Figure 6: Architecture of the interactive dashboard showing main components and data flow

VIII Conclusion

The implemented pipeline successfully trained and evaluated three regression models for real estate price prediction. While the Gradient-Boosted Trees model showed the best performance among the tested approaches, there remains significant room for improvement in predictive accuracy. Future work should focus on enhancing feature engineering and expanding the model tuning process to achieve better results.

All artifacts have been properly saved to HDFS for future reference and potential deployment in a production environment.

IX Reflections

A. Challenges and difficulties

Several challenges were encountered during the project, including initial difficulties connecting to the Hadoop cluster due to VPN issues. Writing and debugging Hive table creation queries required careful attention to syntax and schema definition. Additionally, data visualization in Superset presented obstacles, such as inconsistent graph rendering and improper axis scaling, necessitating manual adjustments to ensure accurate representation.

B. Recommendations

1. Additional Feature Engineering via creating interaction terms between features or adding location-based features from coordinates

2. Enhanced Preprocessing via implementing outlier detection and treatment and considering target variable transformation (e.g., log(price))

3. Model Improvements via expanding hyperparameter search spaces and experimenting with feature importance analysis.

C. *Contributions of each team member*

Task	Description
1	Collect data, build data pipeline, automation scripts, manage databases.
2	Understand and explore data (EDA), analyze features and their relationships, build and maintain dashboards, visualizations
3	Prepare dataset for ML modeling, build distributed ML models, monitor ML models
4	Document stages, provide documentation for the repository, perform testing of project artifacts, assess the quality”

Tasks	Bogdan Shah	Alsu Khairullina	Mariia Shmakova	Aruzhan Shinbayeva	Deliverables	Hours spent
1	100	0	0	0	real_estate.java	6
2	0	100	0	0	hive_results.txt, q1.csv q2.csv, q3.csv, q1.jpg q2.jpg, q3.jpg	14
3	0	0	100	0	evaluation.csv, model1_predictions.csv, model2_predictions.csv, model3_predictions.csv	10
4	0	0	0	100	Web dashboard	6