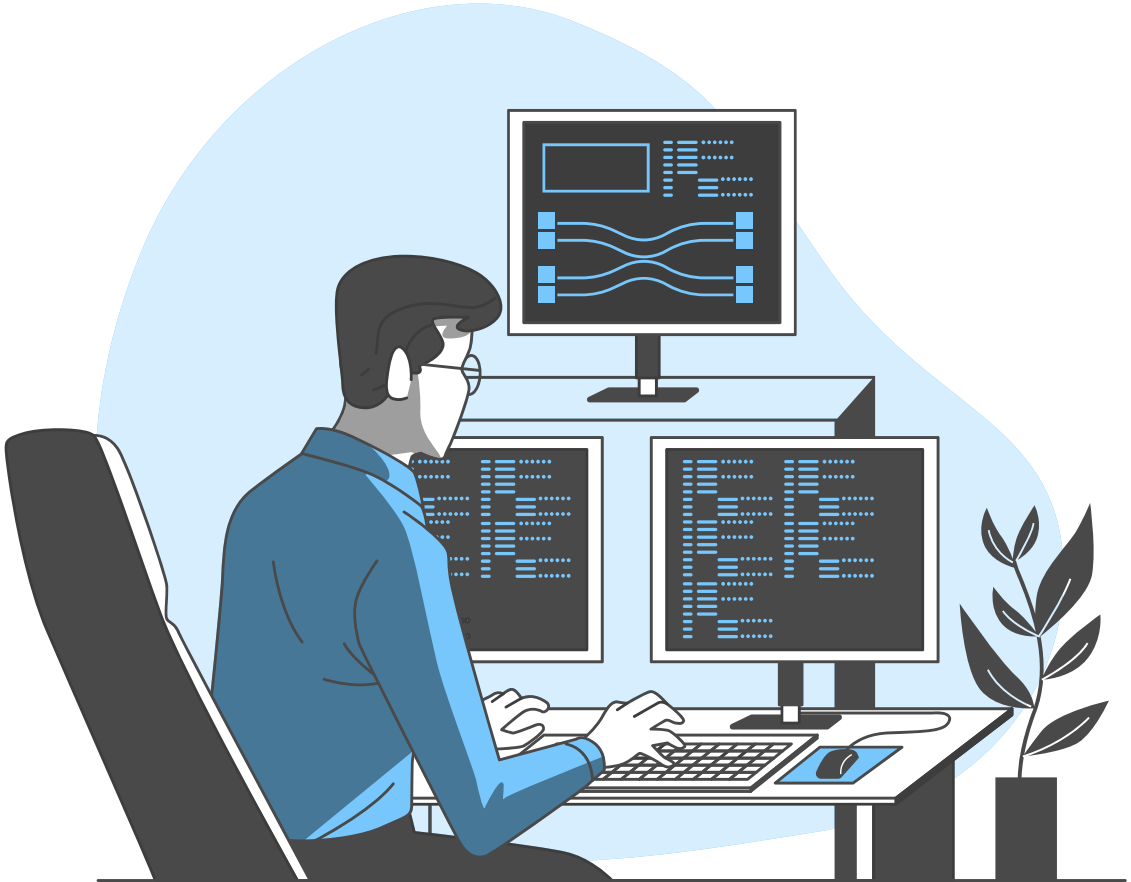


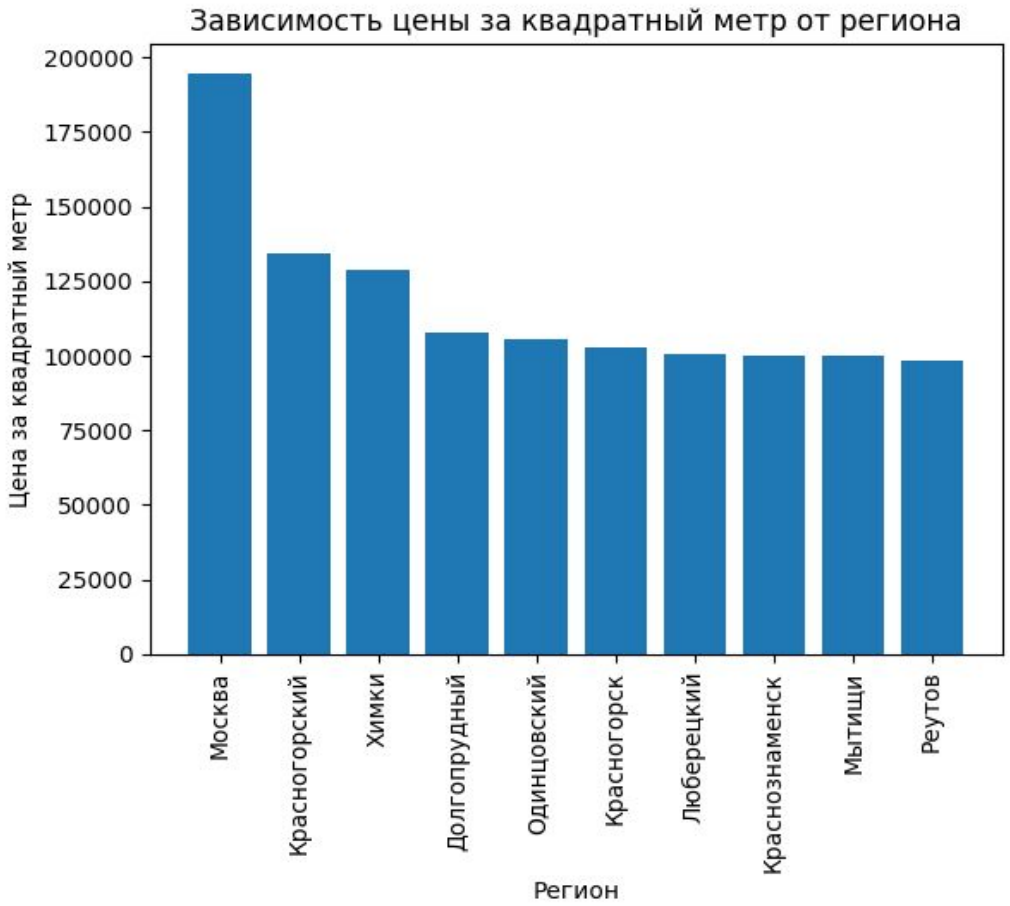
ML FINAL PROJECT

Gleb Kudoyarov MDI 212
Eva Magakelyan MDI 212
Anastasiia Volkova MDI 212

Problems description

1. Linear Regression
2. KNN
3. GBM

Data Analysis



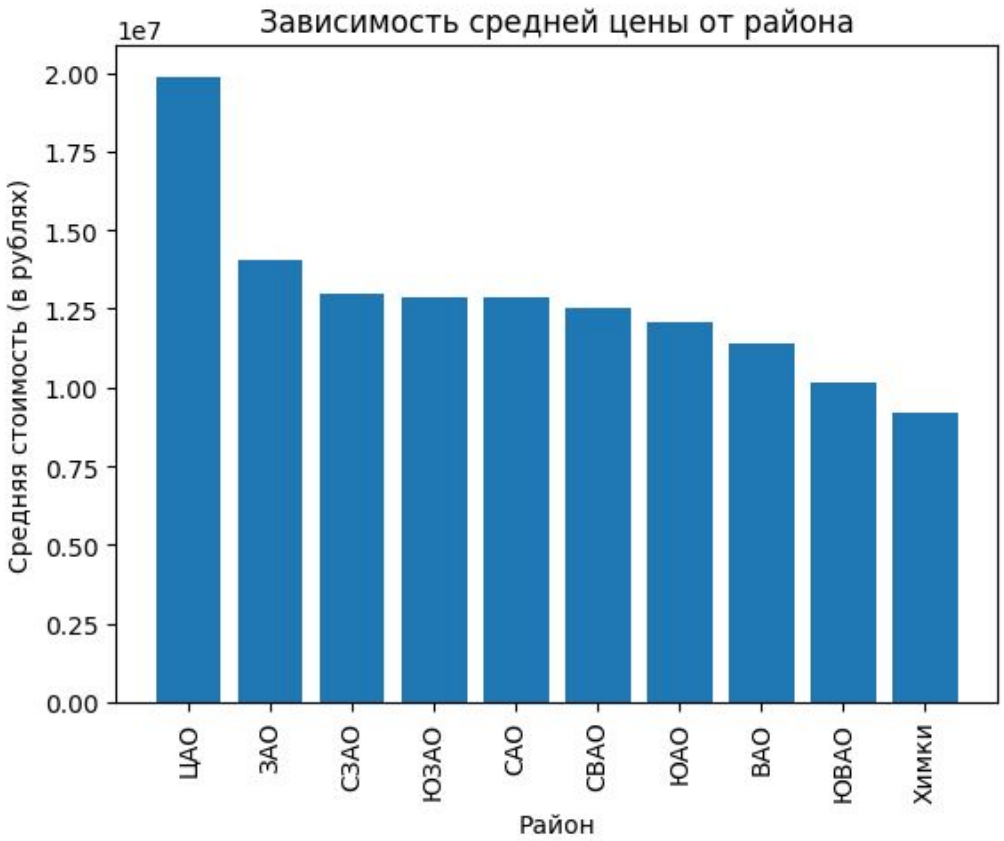
Ввод [55]:

```
import pandas as pd
import matplotlib.pyplot as plt

price_per_sqm = p_df['Стоимость'] / p_df['Общая площадь']
region_prices = p_df.groupby('Регион')['Стоимость'].sum() / p_df.groupby('Регион')['Общая площадь'].sum()
sorted_region_prices = region_prices.sort_values(ascending=False)[:10]

plt.bar(sorted_region_prices.index, sorted_region_prices)
plt.xlabel('Регион')
plt.ylabel('Цена за квадратный метр')
plt.title('Зависимость цены за квадратный метр от региона')
plt.xticks(rotation=90)
plt.show()
```

Data Analysis



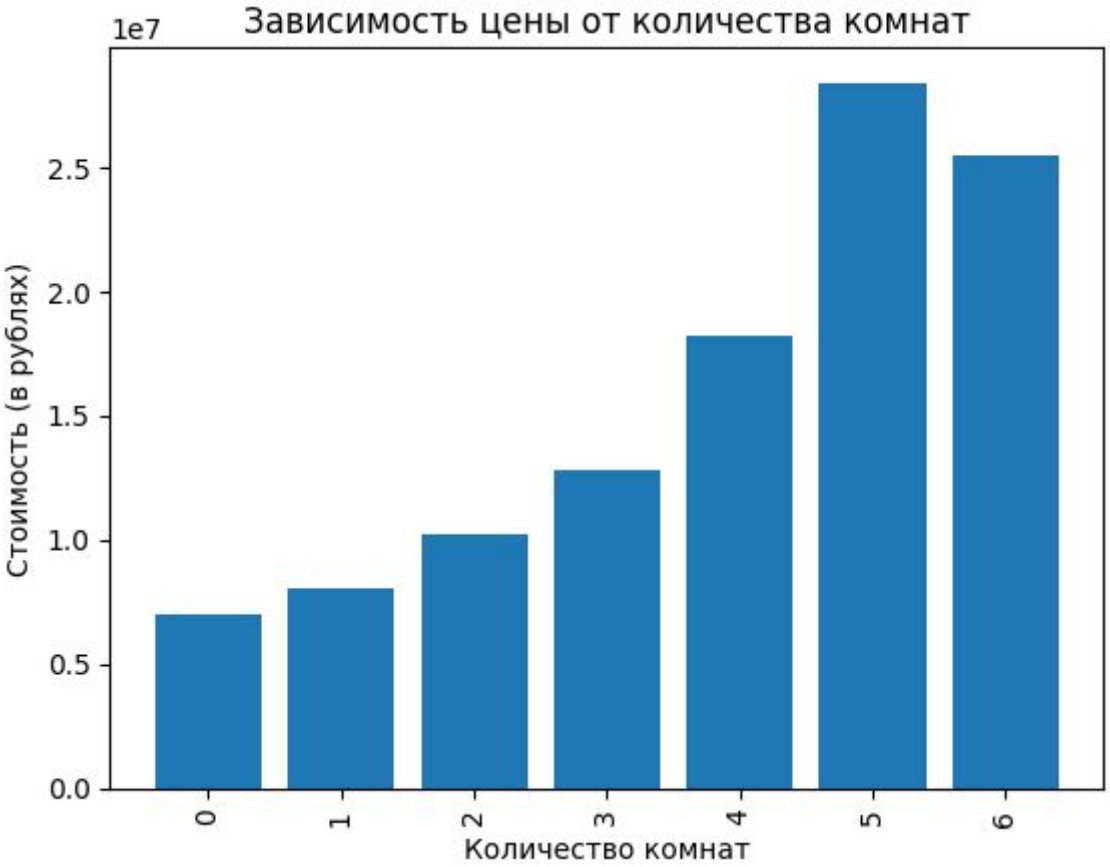
Ввод [58]:

```
import pandas as pd
import matplotlib.pyplot as plt

district_prices = p_df.groupby('Район')['Стоимость'].mean()
sorted_district_prices = district_prices.sort_values(ascending=False)[:10]

plt.bar(sorted_district_prices.index, sorted_district_prices)
plt.xlabel('Район')
plt.ylabel('Средняя стоимость (в рублях)')
plt.title('Зависимость средней цены от района')
plt.xticks(rotation=90)
plt.show()
```

Data Analysis



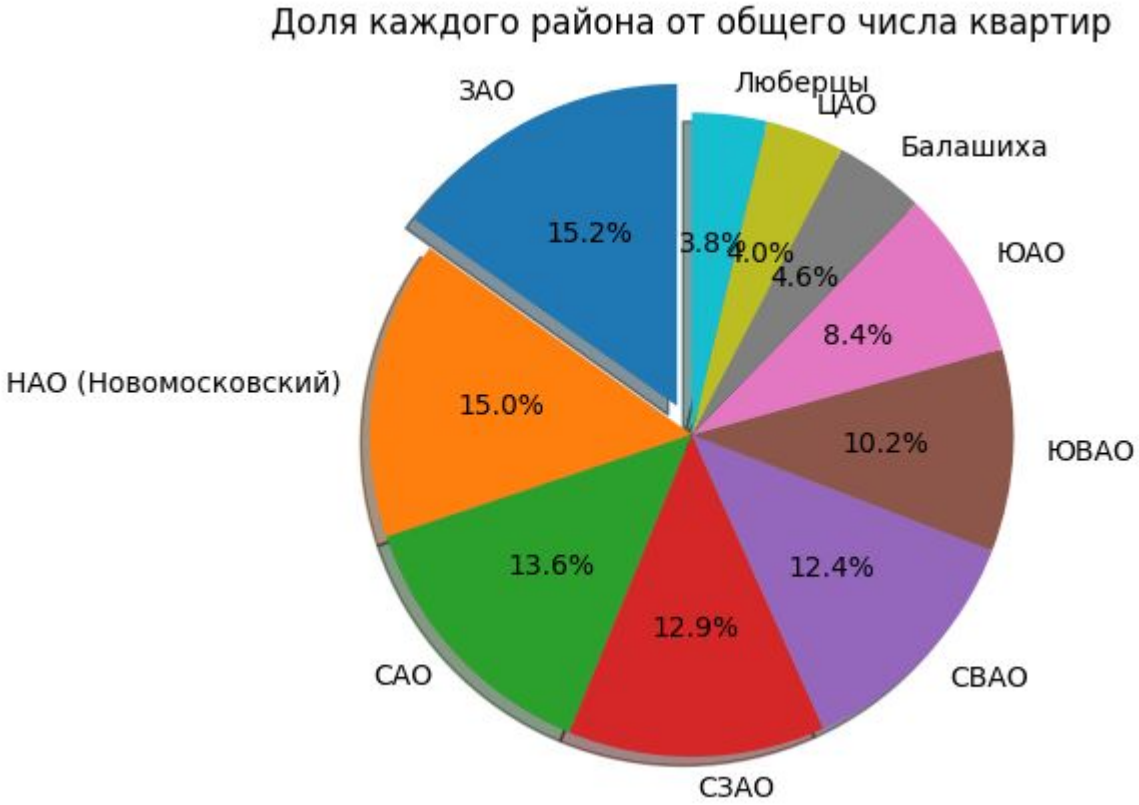
Ввод [57]:

```
import pandas as pd
import matplotlib.pyplot as plt

room_prices = p_df.groupby('Кол-во комнат')['Стоимость'].mean()
sorted_room_prices = room_prices.sort_values(ascending=False)[:10]

plt.bar(sorted_room_prices.index, sorted_room_prices)
plt.xlabel('Количество комнат')
plt.ylabel('Стоимость (в рублях)')
plt.title('Зависимость цены от количества комнат')
plt.xticks(rotation=90)
plt.show()
```

Data Analysis



Ввод [59]:
```
import pandas as pd
import matplotlib.pyplot as plt

district_counts = p_df['Район'].value_counts()
sorted_district_counts = district_counts.sort_values(ascending=False)[:10]

labels = sorted_district_counts.index.tolist()

sizes = sorted_district_counts.values.tolist()

explode = [0.1] + [0] * (len(labels) - 1)

plt.pie(sizes, labels=labels, explode=explode, autopct='%1.1f%%', shadow=True, startangle=90)
plt.axis('equal')
plt.title('Доля каждого района от общего числа квартир')
plt.show()
```

Understanding the Problem

KNN

...

LINEAR REGRESSION

...

GRADIENT BOOSTING

...

KNN

knn

Ввод [66]:
```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import accuracy_score, mean_squared_log_error
from sklearn.preprocessing import MinMaxScaler

features = encoded_df.drop("Стоимость", axis=1)
target = encoded_df["Стоимость"].values

scaler = MinMaxScaler()
scaled_features = scaler.fit_transform(features)

X_train, X_test, y_train, y_test = train_test_split(scaled_features, target, test_size=0.2, random_state=42)

regressor = KNeighborsRegressor(n_neighbors=5)
regressor.fit(X_train, y_train)

y_pred = regressor.predict(X_test)

rmsle = np.sqrt(mean_squared_log_error(y_test, y_pred))
print("RMSLE:", rmsle)
```

```
RMSLE: 0.1907009718947283
```

LINEAR REGRESSION

```
linear regression
```

Ввод [65]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, mean_squared_log_error

features = encoded_df.drop('Стоимость', axis=1)
target = encoded_df['Стоимость']

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

linear_reg = LinearRegression()
linear_reg.fit(X_train_scaled, y_train)

y_pred = linear_reg.predict(X_test_scaled)

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("RMSE:", rmse)
```

```
RMSE: 4063225.4339280673
```

GRADIENT BOOSTING

gradient boosting (библиотека через xgboost)

Ввод [67]:

```
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_log_error
from sklearn.preprocessing import MinMaxScaler
import numpy as np

X = encoded_df.drop("Стоимость", axis=1)
y = encoded_df["Стоимость"].values

scaler = MinMaxScaler()
X = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=22)

dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test)

params = {
    "objective": "reg:squarederror",
    "eval_metric": "rmsle"
}

model = xgb.train(params, dtrain)

y_pred = model.predict(dtest)

rmsle = np.sqrt(mean_squared_log_error(y_test, y_pred))
print("RMSLE:", rmsle)
```

```
RMSLE: 0.18558027378228312
```

Thank You For Your Attention!