Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(УНИВЕРСИТЕТ ИТМО)

Факультет «Систем управления и робототехники»

# ОТЧЕТ
# ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

По дисциплине «Теория идентификации»
на тему:
FLAT PRICE PREDICTION
Вариант 2025-2026

Выполнил студент:
Румянцев Алексей Александрович, 368731

Проверил преподаватель:
Ведяков Алексей Алексеевич

Санкт-Петербург

2025

## Задача

Дан датасет. Необходимо обучить модель, предсказывающую цены на квартиры. Метрикой оценки является RMSE. Цель – набрать RMSE < 500 000.

## Описание датасета

Два файла:

1. “data.csv” – обучающий набор,
2. “test.csv” – тестовый набор.

В датасете содержатся поля: index, kitchen_area, bath_area, other_area, gas, hot_water, central_heating, extra_area, extra_area_count, year, ceil_height, floor_max, floor, total_area, bath_count, extra_area_type_name, district_name, rooms_count, price.

## Обработка датасета

Поля gas, hot_water, central_heating содержат значения Yes или No. Заменим их логическими истиной и ложью – Yes=1, No=0.

Поля extra_area_type_name и district_name содержат различные строковые значения – категориальный признак. Закодируем его с помощью one-hot encoding. На каждый признак будет свой столбец с логическими 1 или 0. Если описываемая квартира принадлежит к данному району, то в нем будет 1, остальные столбцы нулевые.

Добавим в датасет новый признак floor_ratio=floor/floor_max.

Построим тепловую карту корреляции признаков:

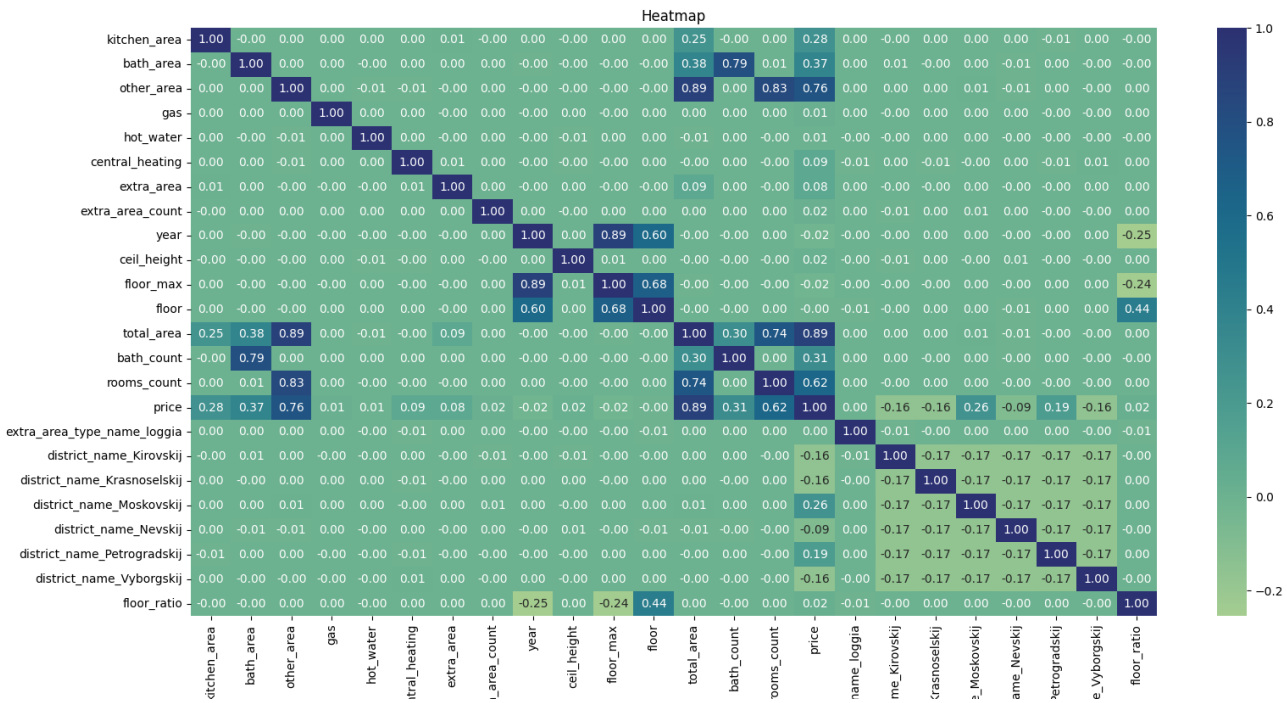


*Рисунок 1: Тепловая карта корреляции признаков*

Большинство признаков не коррелируют значительно с целевой переменной price. Удалим данные признаки. Также нам не нужны признаки, которые сильно коррелируют между собой – оставим те, что больше коррелируют с ценой.

Итоговая тепловая карта:

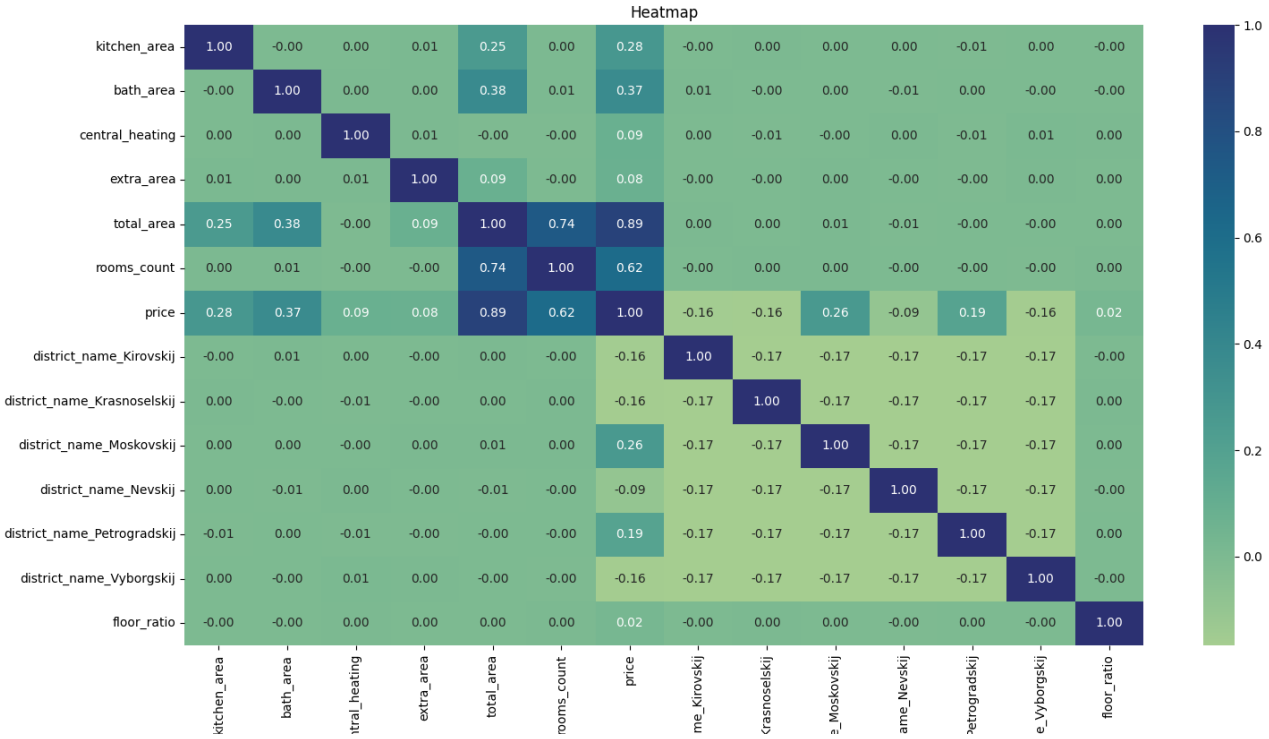


*Рисунок 2: Тепловая карта корреляции признаков после обработки датасета*

Несмотря на то, что добавленный признак floor_ratio почти не коррелирует с целевой переменной, было экспериментально выяснено, что его наличие уменьшает RMSE:

$$RMSE = \left( \frac{1}{N} \Sigma_{i=1}^{N} (price(i) - price_e(i))^2 \right)^{\frac{1}{2}}$$

## Выбор модели, ее обучение и проверка

Разделим обучающий набор на 70% обучающей выборки и 30% тестовой. В качестве модели выберем RandomForestRegressor – ансамблевый алгоритм машинного обучения для задач регрессии, основанный на методе случайного леса. Он строит множество решающих деревьев, каждое из которых обучается на случайной подвыборке данных и признаков. Прогноз итоговой модели считается как среднее предсказаний всех деревьев.

С помощью GridSearchCV были подобраны лучшие параметры для модели:

1. max_depth=20,
2. max_features='sqrt',
3. min_samples_leaf=1,
4. min_samples_split=2,
5. n_estimators=500,
6. random_state=42,

7. n_jobs=-1

RMSE для 30% тестового набора:

```
PS C:\Users\alexe\study\id-theory-labs> & C:/Users/alexe/AppData/Local/P
g.py
Index(['kitchen_area', 'bath_area', 'central_heating', 'extra_area',
       'total_area', 'rooms_count', 'price', 'district_name_Kirovskij',
       'district_name_Krasnoselskij', 'district_name_Moskovskij',
       'district_name_Nevskij', 'district_name_Petrogradskij',
       'district_name_Vyborgskij', 'floor_ratio'],
      dtype='object')
PS C:\Users\alexe\study\id-theory-labs> & C:/Users/alexe/AppData/Local/P
Test RMSE: 465389.72439069534
PS C:\Users\alexe\study\id-theory-labs>
```

*Рисунок 3:RMSE для 30% тестового набора*

Цель RMSE < 500 000 достигнута.

## Прогноз модели на тестовом датасете

Используем модель для предсказания на тестовом датасете, который обработаем аналогично обучающему. Составим таблицу в соответствии с шаблоном: index, price. Загрузим таблицу на сайт. Результат:

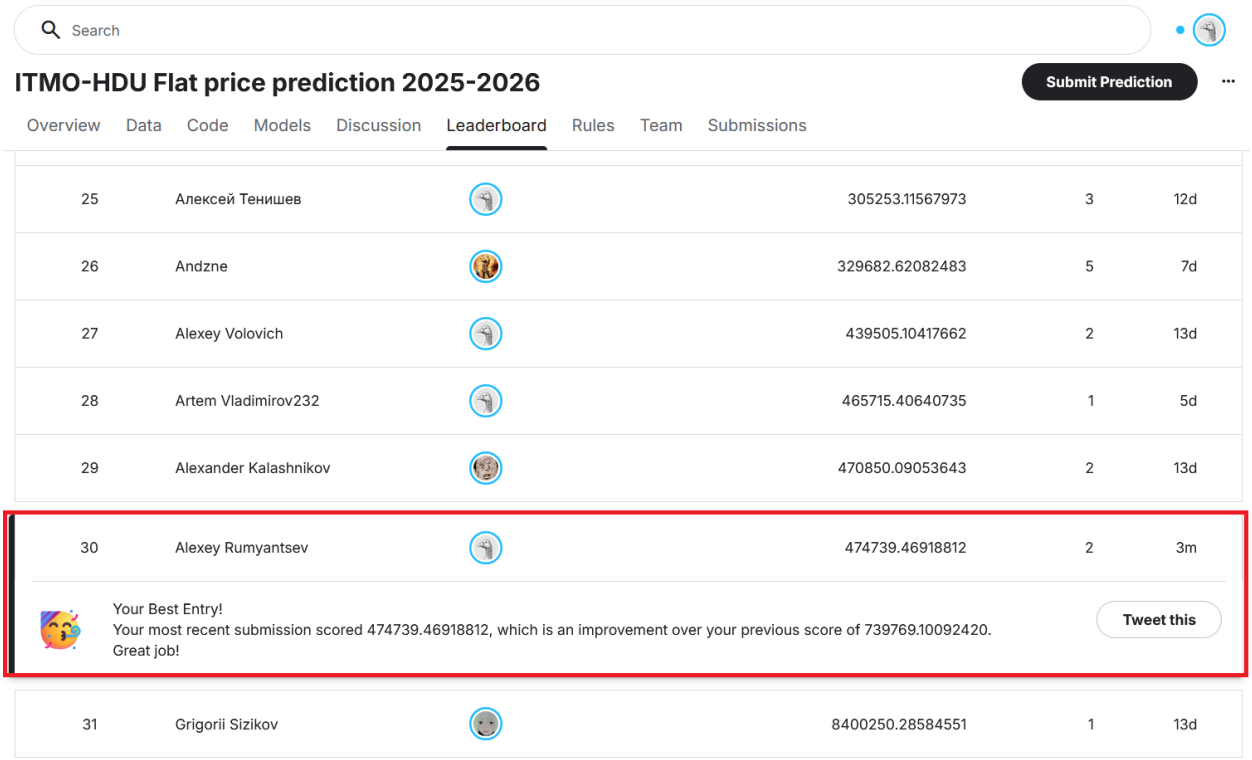


*Рисунок 4: RMSE для тестового датасета*

Цель достигнута – RMSE < 500 000. Результат почти такой же, как и на 30% тестовой выборки. Это означает, что модель хорошо обучилась – нет переобучения или недообучения.

## Вывод

В ходе выполнения лабораторной работы был обработан датасет, выбрана модель для предсказания целевой переменной, обучена и проверена на нем. Далее модель была проверена на тестовом датасете. Результат удовлетворяет заданному в условии неравенству.

## Приложение А

Код для обработки датасетов:

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

def process_data(data):
    for col in ['gas', 'hot_water', 'central_heating']:
        data[col] = data[col].map({'Yes': 1, 'No': 0})
    data = pd.get_dummies(
        data,
        columns=['extra_area_type_name'],
        drop_first=True
    )
    data = pd.get_dummies(
        data,
        columns=['district_name'],
        drop_first=True
    )
    return data.drop(columns=['index'])

def heatmap(data):
    corr_matrix = data.corr()
    plt.figure(figsize=(16, 9))
    sns.heatmap(corr_matrix, annot=True, cmap='crest', fmt='.2f')
    plt.title('Heatmap')
    plt.show()

df=pd.read_csv('lab3/Archive2025/data.csv')
df=process_data(df)

df['floor_ratio']=df['floor']/df['floor_max']

heatmap(df)

to_drop = ['gas', 'hot_water', 'ceil_height',
           'extra_area_count', 'floor', 'bath_count',
           'other_area', 'floor_max', 'year',
           'extra_area_type_name_loggia']

df = df.drop(columns=to_drop)
heatmap(df)
```

```
print(df.columns)

df.to_csv('lab3/processed_data.csv', index=False)

df_test = pd.read_csv('lab3/Archive2025/test.csv')
df_test = process_data(df_test)
df_test['floor_ratio']=df_test['floor']/df_test['floor_max']
df_test = df_test.drop(columns=to_drop)
df_test.to_csv('lab3/processed_test_data.csv', index=False)
```

*Листинг 1: Обработка датасета*

## Приложение Б

Код для тренировки и проверки модели:

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
import joblib

def get_rmse(y_true, y_pred):
    mse = np.mean((y_true - y_pred) ** 2)
    return np.sqrt(mse)

df=pd.read_csv('lab3/processed_data.csv')

m, n = df.shape
idx = np.random.permutation(m)
P = 0.7

split = int(P * m)
train_df = df.iloc[idx[:split]]
test_df = df.iloc[idx[split:]]

X_train = train_df.drop(columns=['price'])
y_train = train_df['price']

X_test = test_df.drop(columns=['price'])
y_test = test_df['price']

model = RandomForestRegressor(
    max_depth=20,
    max_features='sqrt',
    min_samples_leaf=1,
    min_samples_split=2,
    n_estimators=500,
    random_state=42,
    n_jobs=-1
)
model.fit(X_train, y_train)
```

```
y_pred_model = model.predict(X_test)

rmse = get_rmse(y_test.values, y_pred_model)
print("Test RMSE:", rmse)

joblib.dump(model, 'lab3/model.joblib')
```

*Листинг 2:Тренировка и проверка модели*

## Приложение В

Пример кода для поиска лучших параметров модели:

```
from sklearn.model_selection import GridSearchCV
rf = RandomForestRegressor(random_state=100, n_jobs=-1)
param_rf = {'n_estimators': [100, 200],
            'max_depth': [10, 20, None],
            'min_samples_leaf': [1, 2],
            'min_samples_split': [2, 5],
            'max_features': ['sqrt', 'log2']}
rf_grid = GridSearchCV(rf,
                       param_rf,
                       cv=4,
                       scoring='neg_root_mean_squared_error',
                       n_jobs=-1)
rf_grid.fit(X_train, y_train)
best_rf = rf_grid.best_estimator_
print("Best RMSE:", -rf_grid.best_score_)
print("Best params:", rf_grid.best_params_)
y_pred_rf = best_rf.predict(X_test)
rmse_rf = get_rmse(y_test.values, y_pred_rf)
print("Test RMSE:", rmse_rf)
```

*Листинг 3: GridSearchSV поиск лучших параметров модели*

## Приложение Г

Код для прогноза на тестовом датасете и сохранения результатов в соответствии с шаблоном:

```
import pandas as pd
import joblib

saved_model = joblib.load('lab3/model.joblib')
df_test = pd.read_csv('lab3/processed_test_data.csv')
y_test_pred = saved_model.predict(df_test)
price_table = pd.DataFrame({
    'index': range(len(y_test_pred)),
    'price': y_test_pred
})
price_table.to_csv('lab3/test_pred.csv', index=False)
```

*Листинг 4: Прогноз на тестовом датасете, сохранение результатов*