Machine Learning Predictive Analysis of Warsaw Housing Rental Prices

Research goal:

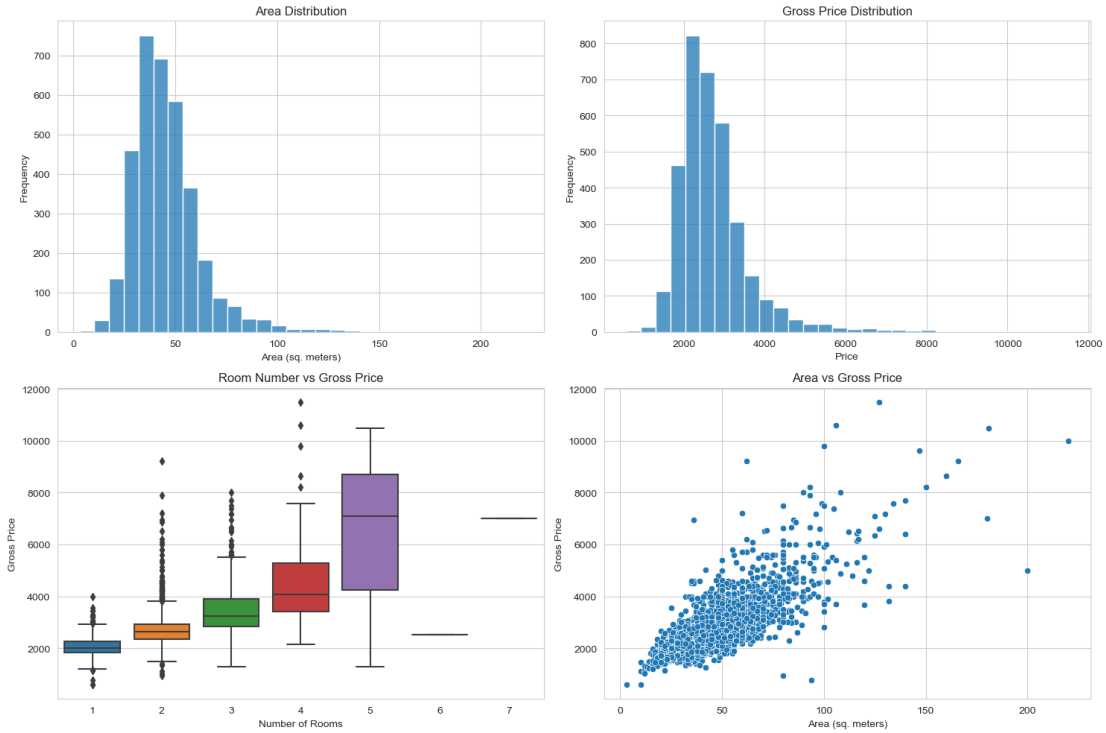
Rent in Warsaw is a topic unavoidable for us. If you plan to stay here in the future, unless you win the lottery and can fully purchase an apartment in Zlota 44, your priority will be considering renting a house.
Our research aims to apply and compare four different machine learning models to predict rental prices in Warsaw. We want to identify which model is most suitable for handling real estate market data.

Description of the data set

The data is downloaded from Kaggle. We made minor adjustments to the data csv file, removing some columns that we thought were completely useless, such as the type of window in the room, the material of the door etc.
The dataset includes various features related to house rents, such as area, number of rooms, floor, construction year, etc., along with the target variable 'gross_price' (rental price). The dataset contains a total of 3472 records.

Data Preprocessing and Exploratory Data Analysis (EDA):

We verified that there are no missing values in the dataset, and most features are numeric.



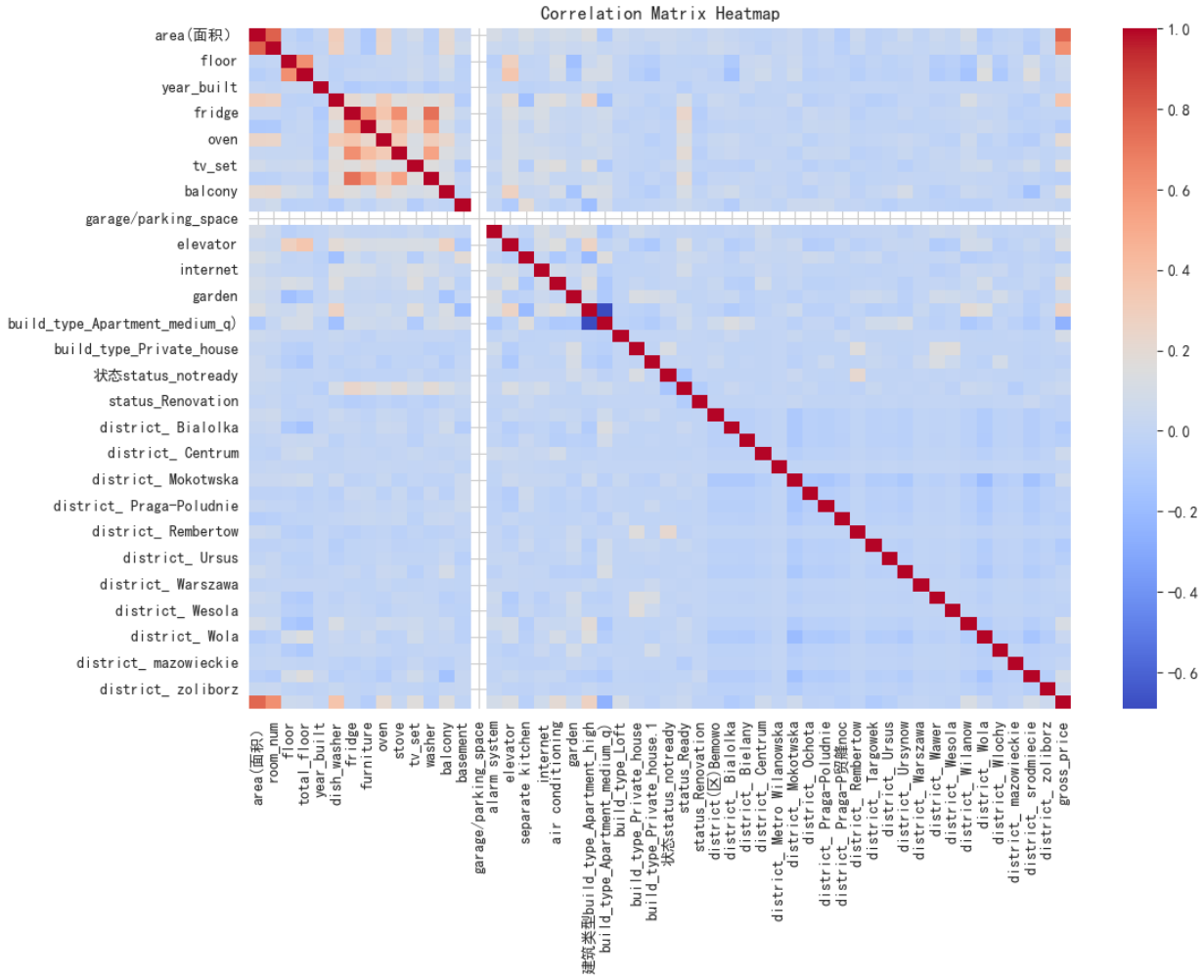
Area Distribution: Displays the distribution of house areas. Most houses have areas concentrated within a smaller range.

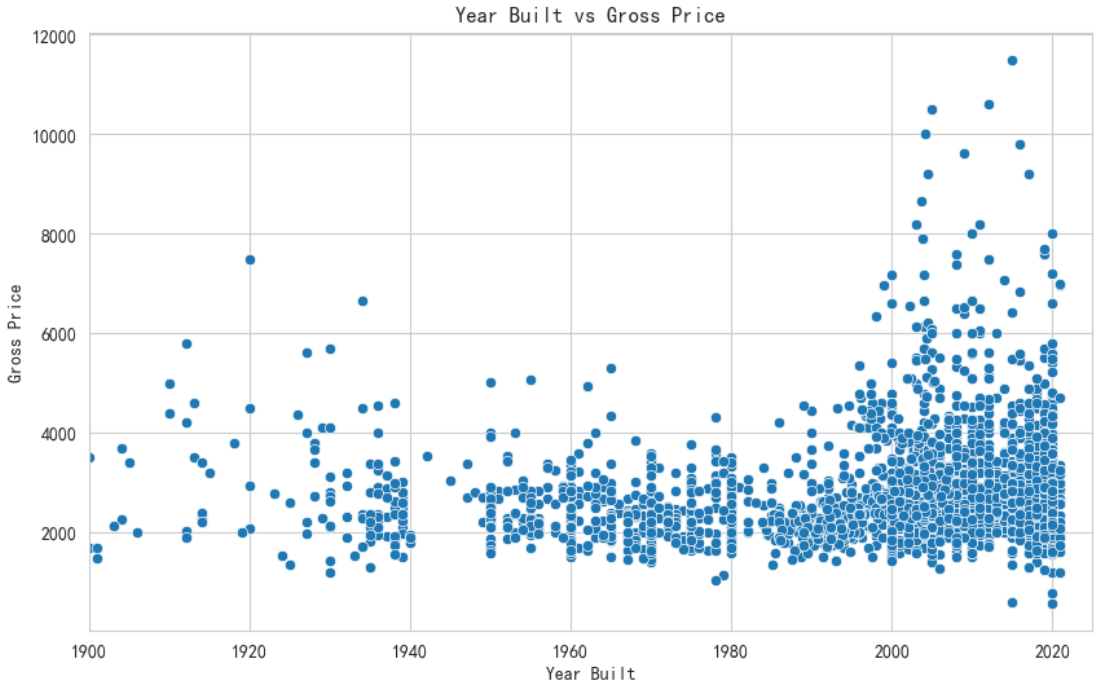
Rental Price Distribution: Shows the distribution of rental prices for houses. Prices seem to be concentrated within a specific range, but there are also some high values.

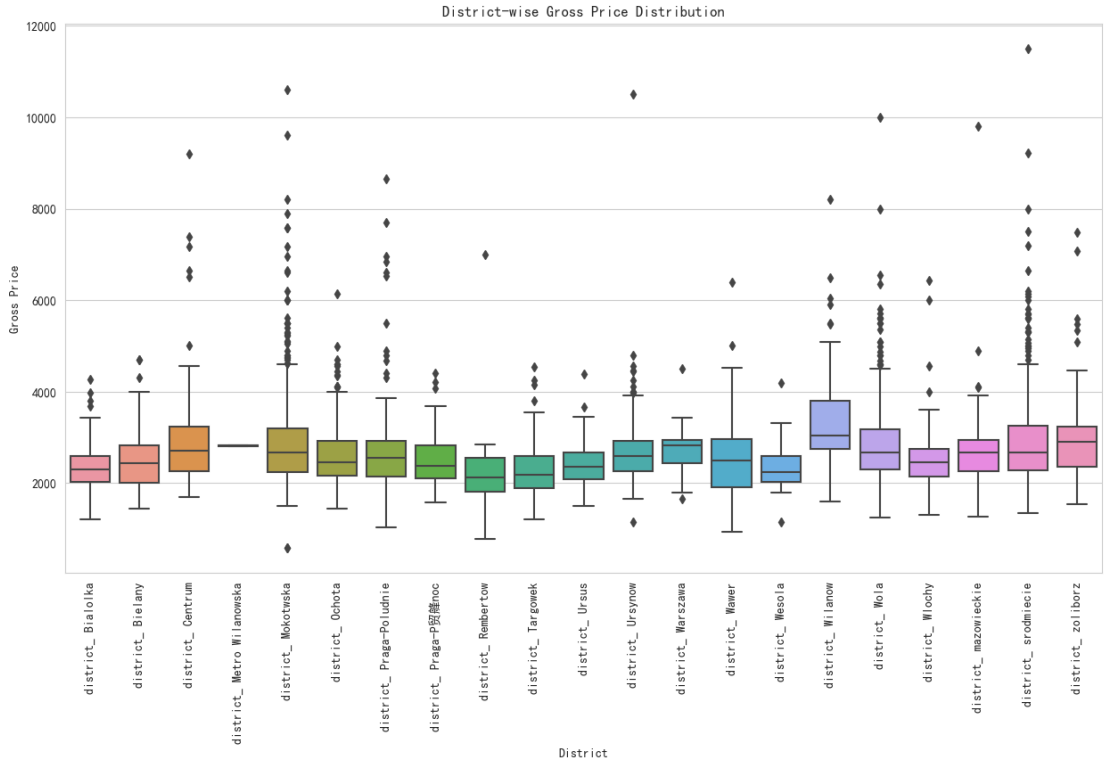
Number of Rooms and Rental Prices: The box plot shows the distribution of rental prices for houses with different numbers of rooms, shows the relationship between the number of rooms and rental prices.

Area and Rental Prices: The scatter plot shows the relationship between house area and rental prices.

Rental prices are mainly concentrated in lower ranges, but there are also listings with higher prices. There is a positive correlation between the number of rooms and rental prices.


Correlation Matrix Heatmap
area(面积)
floor
year_built
fridge
oven
tv_set
balcony
garage/parking_space
elevator
internet
garden
build_type_Apartment_medium_q)
build_type_Private_house
状态status_notready
status_Renovation
district_ Bialolka
district_ Centrum
district_ Mokotwska
district_ Praga-Poludnie
district_ Rembertow
district_ Ursus
district_ Warszawa
district_ Wesola
district_ Wola
district_ mazowieckie
district_ zoliborz
area(面积)
room_num
floor
total_floor
year_built
dish_washer
fridge
furniture
oven
stove
tv_set
washer
balcony
basement
garage/parking_space
alarm system
elevator
separate kitchen
internet
air conditioning
garden
建筑类型build_type_Apartment_high
build_type_Apartment_medium_q)
build_type_Loft
build_type_Private_house
build_type_Private_house.1
状态status_notready
status_Ready
status_Renovation
district(区)Bemowo
district_ Bialolka
district_ Bielany
district_ Centrum
district_ Metro Wilanowska
district_ Mokotwska
district_ Ochota
district_ Praga-Poludnie
district_ Praga-Północ
district_ Rembertow
district_ Targowek
district_ Ursus
district_ Ursynow
district_ Warszawa
district_ Wawer
district_ Wesola
district_ Wilanow
district_ Wola
district_ Włochy
district_ mazowieckie
district_ srodmiecie
district_ zoliborz
gross_price
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


Year Built vs Gross Price
Gross Price
12000
10000
8000
6000
4000
2000
1900
1920
1940
1960
1980
2000
2020
Year Built



Correlation Matrix Heatmap: shows the correlation between all the features in the dataset. Dark areas indicate strong correlations, while light areas indicate weak correlations. We can identify the features that are most correlated with rental prices.

Relationship between year of construction and rental prices: the relationship between year of construction and rental prices is demonstrated through a scatter plot. The effect of building year on rental prices can be observed.

Distribution of Rental Prices in Different Regions: The box plot shows the distribution of rental prices in different districts. This helps to understand which district have higher

or lower rental prices .It seems rents of house in Centurm, Mokotwo and Willanow are relatively higher.
Relationship between floors and rental prices: The scatter plot shows the relationship between floors and rental prices. It shows whether the height of the floor affects the rental price.

Models

Linear Regression

In [8]:
```
from sklearn.linear_model import LinearRegression

# Initialize Linear Regression
linear_regressor = LinearRegression()

# Training the Linear Regression model
linear_regressor.fit(X_train, y_train)

# Predicting on the test set
y_pred_linear = linear_regressor.predict(X_test)

# Calculating the evaluation metrics for Linear Regression
mse_linear = mean_squared_error(y_test, y_pred_linear)
mae_linear = mean_absolute_error(y_test, y_pred_linear)
r2_linear = r2_score(y_test, y_pred_linear)

mse_linear, mae_linear, r2_linear
```

Out[8]: (289881.97438202205, 362.97789479795216, 0.696086524345958)

The first model we used is the linear regression model. As a basic model, we want to use it as a benchmark for comparison with other models.

Boosting Model

Adaboost

```
# Initialize AdaBoostRegressor
ada_regressor = AdaBoostRegressor(random_state=42)

# Define the range of values for n_estimators to search
param_grid = {'n_estimators': [50, 100, 150, 200]}

# Initialize GridSearchCV
grid_search = GridSearchCV(ada_regressor, param_grid, cv=5, scoring='neg_mean_squared

# Perform the grid search on the training data
grid_search.fit(X_train, y_train)

# Get the best parameters
best_params = grid_search.best_params_

# Initialize AdaBoostRegressor with the best parameters
ada_regressor = AdaBoostRegressor(n_estimators=best_params['n_estimators'], random_st

# Train the model with the best parameters
ada_regressor.fit(X_train, y_train)

# Predict on the test set
y_pred =ada_regressor.predict(X_test)

# Calculate the evaluation metrics with the best model
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

best_params, mse, mae, r2
```

```
({'n_estimators': 50},
 403476.0772305412,
 505.4540763489867,
 0.5769939913104261)
```

Xgboost

```
# Define the range of values for n_estimators to search
param_grid = {'n_estimators': [50, 100, 150, 200]}

# Initialize GridSearchCV
grid_search_xgb = GridSearchCV(xgb_regressor, param_grid, cv=5, scoring='neg_mean_sq

# Perform the grid search on the training data
grid_search_xgb.fit(X_train, y_train)

# Get the best parameters
best_params_xgb = grid_search_xgb.best_params_

# Initialize XGBRegressor with the best parameters
xgb_regressor_best = XGBRegressor(n_estimators=best_params_xgb['n_estimators'], rand

# Training the model with the best parameters
xgb_regressor_best.fit(X_train, y_train)

# Predicting on the test set
y_pred_xgb = xgb_regressor_best.predict(X_test)


# Calculating the evaluation metrics
mse_xgb = mean_squared_error(y_test, y_pred_xgb)
mae_xgb = mean_absolute_error(y_test, y_pred_xgb)
r2_xgb = r2_score(y_test, y_pred_xgb)

best_params_xgb,mse_xgb, mae_xgb, r2_xgb
```

```
({'n_estimators': 50},
 230051.71140340948,
 321.2138773104524,
 0.7588128225571144)
```

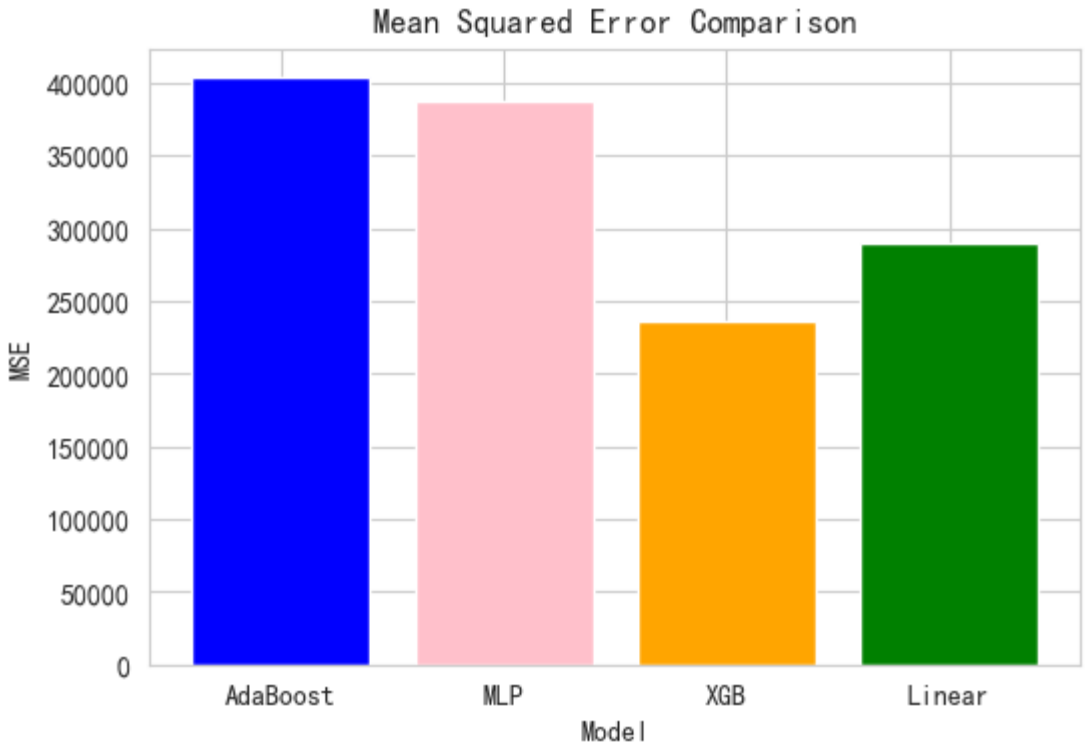
Neural network model

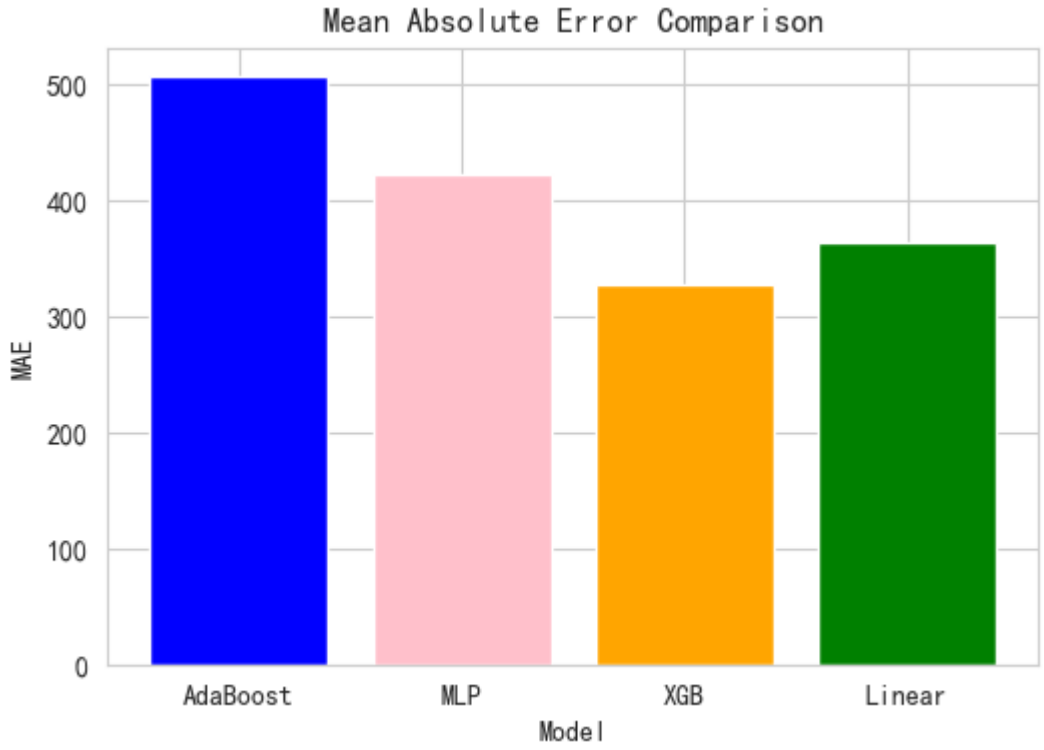
Multi-layer Perceptron Regressor

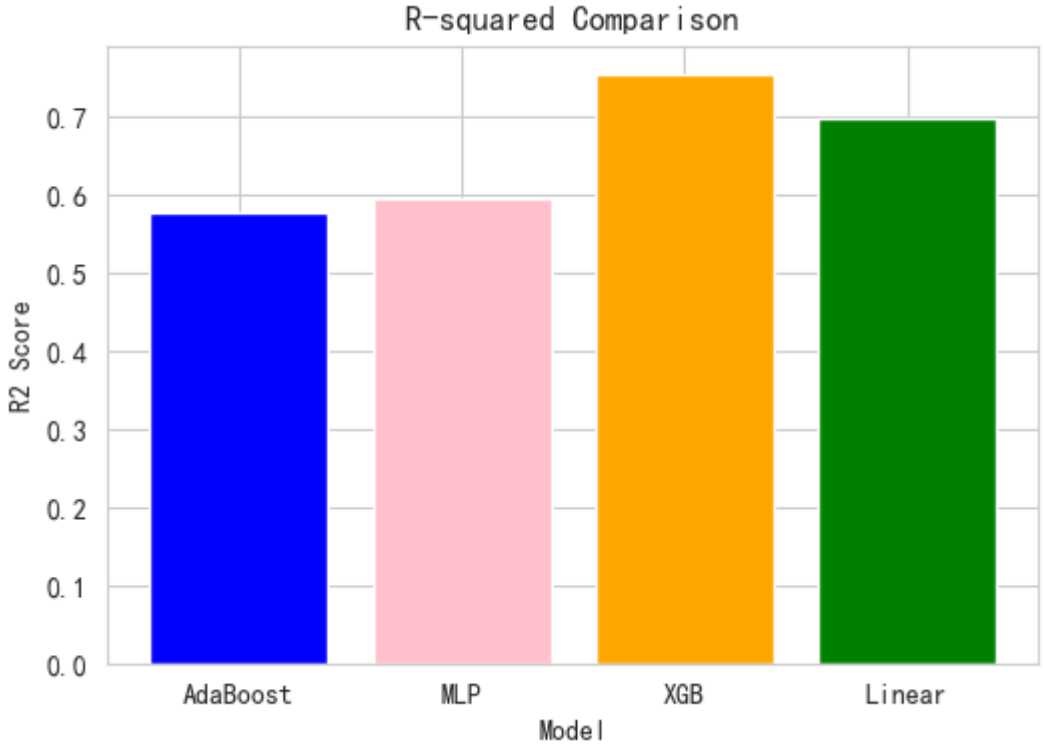
```
from sklearn.neural_network import MLPRegressor

# Initialize MLPRegressor
mlp_regressor = MLPRegressor(hidden_layer_sizes=(100, 50), max_iter=500, random_state=42)

# Training the MLP Regressor
mlp_regressor.fit(X_train, y_train)

# Predicting on the test set
y_pred_mlp = mlp_regressor.predict(X_test)

# Calculating the evaluation metrics for MLP Regressor
mse_mlp = mean_squared_error(y_test, y_pred_mlp)
mae_mlp = mean_absolute_error(y_test, y_pred_mlp)
r2_mlp = r2_score(y_test, y_pred_mlp)

mse_mlp, mae_mlp, r2_mlp


在 4s 的 15 Jan at 16:48:25 执行
```

```
(387930.3729716862, 421.30920852368854, 0.5932921727439919)
```

Comparison of model results







AdaBoostRegressor:
Mean Square Error (MSE): 403476.08
Mean Absolute Error (MAE): 505.45
R^2 score: 0.577
AdaBoostRegressor is a regression model based on the boosting method, which enhances performance by combining multiple weak learners. In this case, it provides a

moderate level of performance, with an R^2 score indicating its ability to explain approximately 57.7% of the data variation.

MLPRegressor:

Mean Squared Error (MSE): 393739.14
Mean Absolute Error (MAE): 395.07
R^2 Score: 0.587
MLPRegressor is a regression model based on neural networks. It performs slightly better than AdaBoostRegressor in this case, particularly in terms of MSE and R^2 score. However, neural network models typically require careful parameter tuning and sufficient training time.

Linear Regression

Mean Square Error (MSE): 289881.97
Mean Absolute Error (MAE): 362.98
R^2 score: 0.696
Linear regression is the simplest model but performs exceptionally well on this dataset. It exhibits good MSE and R^2 scores among all tested models, indicating that even a simple model can provide strong performance under appropriate data conditions.

XGBRegressor:

Mean Square Error : 230051.71
Mean Absolute Error :321.2138
R^2 score: 0.7588

XGBRegressor is an ensemble learning algorithm based on gradient boosting, known for its excellent performance in many machine learning competitions .In this case, it outperforms all other models, with the lowest MSE and MAE, as well as the highest R^2 score, demonstrating its effectiveness in explaining data variations.

Conclusion:

In this study, we used four different machine learning models—Linear Regression, AdaBoostRegressor, MLPRegressor, and XGBRegressor—to predict rental prices in the Warsaw housing leasing market. After training and evaluation, we found that XGBoost outperformed all other models across all evaluation metrics. This result highlights the efficiency and strong generalization capabilities of the XGBoost when dealing with complex datasets. The performance of the linear regression model was also quite satisfactory, while AdaBoostRegressor and MLPRegressor, lagged behind XGBoost in all three key metrics .

Although XGBoost performes well, improve and adjust the paraments may improve its performance even further. If we consider using ensemble learning methods such as stacking or blending, combining predictions from different models may produce more accurate results.

Hanwen Miao

Yue Zhou