Title: Predicting California Housing Prices Using the XGBoost Regressor Model

Abstract

The demand for accurate housing price predictions has grown significantly with real estate's importance to economic development and individual investment decisions. This study investigates the application of machine learning techniques, specifically the XGBoost Regressor model, to predict California housing prices using various socioeconomic and geographic features. The model achieved a high accuracy score, showcasing its efficacy in predictive real estate analysis.

1. Introduction

The housing market is a complex ecosystem driven by numerous variables such as location, socioeconomic factors, and economic conditions. Machine learning provides an effective toolset for capturing patterns within these variables, allowing for precise predictions of housing prices. This study employs the XGBoost Regressor model, a robust and widely used machine learning algorithm, to predict housing prices based on features from the California Housing dataset.

The primary goal of this study is to analyze the predictive power of the XGBoost Regressor and demonstrate its potential for practical applications in housing market analytics.

2. Dataset and Preprocessing

This project utilizes the **California Housing dataset**, which includes features such as average rooms, population, median income, and other socioeconomic factors. The target variable is the median housing price in different locations in California.

2.1 Data Loading and Exploration

The dataset was loaded using the `fetch_california_housing` function, and the data was stored in a Pandas DataFrame. A preliminary analysis was performed to inspect the structure and summary statistics of the data. Additionally, a target column, `Price`, was added to represent the median housing price.

```
# Code snippet for loading and examining data
house_price_dataframe = pd.DataFrame(house_price_dataset.data, columns = house_price_dataset.feature_names)
house_price_dataframe['Price'] = house_price_dataset.target
```

2.2 Data Cleaning and Analysis

An examination for missing values revealed a clean dataset with no null values. A correlation matrix was generated to explore relationships between the features and the target variable, providing insight into features with potential predictive significance.

```
# Code snippet for correlation heatmap

plt.figure(figsize = (10,10))

sns.heatmap(correlation, annot = True, cmap = 'Blues')
```

3. Methodology

This study used the XGBoost Regressor model, chosen for its high performance and efficiency in regression tasks. XGBoost (Extreme Gradient Boosting) is an optimized gradient-boosting algorithm that builds models in stages, focusing on minimizing prediction errors by correcting the residuals from previous models. This iterative process is particularly effective for tabular data, as in this study.

3.1 Splitting the Data

The dataset was divided into independent features (`X`) and the target feature (`Price`). A training-test split of 80-20 was applied to separate the dataset, with a random state of 2 to ensure consistency across different runs.

```
# Code for splitting data
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 2)
```

4. Model Training and Evaluation

The XGBoost Regressor model was trained on the training dataset using default hyperparameters. After training, the model's performance was evaluated based on its predictions for both the training and test datasets.

4.1 Training and Evaluating the Model on Training Data

The model's performance was assessed on the training data using the R-squared error and Mean Absolute Error (MAE) metrics. The R-squared score measures the proportion of variance explained by the model, while MAE calculates the average error in absolute terms.

```
# Code for training and calculating accuracy
model = XGBRegressor()
model.fit(X_train, Y_train)
```

```
training_data_prediction = model.predict(X_train)
error_score = metrics.r2_score(Y_train, training_data_prediction)
mae = metrics.mean_absolute_error(Y_train, training_data_prediction)
```

- **R-squared Score (Training): 0.95**
- **Mean Absolute Error (Training): 0.2**

4.2 Model Performance on Test Data

For unbiased evaluation, the model was also tested on unseen data. The predictions were compared against the actual values using the same metrics.

```
# Code for evaluation on test data
test_data_prediction = model.predict(X_test)
error_score = metrics.r2_score(Y_test, test_data_prediction)
mae = metrics.mean_absolute_error(Y_test, test_data_prediction)
```

- **R-squared Score (Test)**: 0.82
- **Mean Absolute Error (Test)**: 0.4

The high R-squared and relatively low MAE indicate that the model generalizes well and provides reliable predictions on unseen data.

4.3 Visualization

To further analyze model accuracy, a scatter plot was generated comparing the actual housing prices with the predicted prices, allowing a visual assessment of prediction accuracy.

```
# Code for visualization
plt.scatter(Y_train, training_data_prediction)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual Price vs Predicted Price")
plt.show()
```

5. Discussion

The XGBoost Regressor model demonstrated strong performance in predicting California housing prices, achieving high R-squared scores on both training and test data. The feature selection process, focusing on socioeconomic and geographical attributes, proved effective. However, the model may benefit from hyperparameter tuning, which could further optimize accuracy. Additionally, including more location-based features or updating the dataset could improve its relevance for real-time applications.

6. Conclusion

This study successfully applied the XGBoost Regressor model to predict California housing prices based on multiple influential features. The model's performance confirms its suitability for predictive real estate analysis and highlights its potential for deployment in the real estate industry. Future work may include incorporating additional datasets and experimenting with different regression models for performance comparisons.

References

1. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794).
2. Pace, R. K., & Barry, R. (1997). Sparse Spatial Autoregressions. *Statistics & Probability Letters*, *33*(3), 291-297.