Report: Predicting House Prices with Simple Linear Regression

This project was to develop a model using Python from scratch utilizing the ideas of linear regression that were discussed in class. I analyzed the model using a dataset, which provided the accuracy performance as determined by the Mean Squared Error (MSE) metric.

Dataset

The dataset was first retrieved from a CSV file: datasets_house_prices; next, datasets were used in dataFrame loops. csv and the first stage of data preprocessing, which addresses the removal of missing values, have also been made available. In order to deal with missing data, missing values were imputed using the median of the corresponding column. This approach of estimating was less vulnerable to outliers and preserved the distribution of the data set. Then, feature scaling was started by using MinMaxScaler to set the features' positions to [0,1] for Size (sqft), Number of Bedrooms, and Age. This was a significant scaling step because it increased the rate of resolution of the linear regression and aligned the feature space to other spaces.

Model Implementation:

The linear regression model in this case was developed using the Normal Equation technique to calculate for the model coefficients, or theta. The predict function then used the above computed parameters to make the predictions with a help of the intercept term. This approach offers the advantage of giving a closed-form solution for defining the connection between features as well as the target variable or the Price. Using the least squares method, the linear regression model was applied to determine the best line that can be used to estimate house prices. I have developed the model from the ground while instead of it we could have used existing Scikit-learn’s libraries. This involved estimating the values of the model parameters on the basis of minimizing the sum of squared errors of the actual house prices and the forecasted ones. Besides, we also used these parameters to define the prediction function where we forecasted the price of a house depending on the features provided.

Model Training:

To train the model, we split the dataset into two parts: test and training data where the test data comprised only 20% and the training data 80%. The training set was used to train the models by making the necessary adjustments on the model parameters while the test set was retained for model testing with the actual new data. After training we compute the Mean Squared Error (MSE) on the training data to verify the correctness of model prediction.

Model Evaluation:

While testing the model, we employed the MSE, calculated on the test set that was used to determine the model's accuracy when trained on unseen data. For test MSE the calculated value was 167882080. 37, which was not very far from the training MSE which indicates low chances of overfitting and high chances of the model generalize well. Also, scatter plot of actual vs predicted prices of the test set was depicted which made us realize that the actual values and the predicted values of the model are quite close, which may also be an evidence of its performance.

Conclusion:

Using size, number of bedrooms, and age as features, the linear regression model offered relatively accurate decisions about the house prices. The fact that the MSE values of the two sets were almost equal also showed that the model was able to generalize on different data sets. An issue that was encountered was the scaling of the features so as to ensure that they are at a similar scale for training purposes for the improved model. The model might be augmented in the future with additional features, for instance, location or the neighborhood to predict even better.

Challenges:

Handling the various feature scales provided one of the biggest challenges during this project. The house size was measured in thousands of square feet, while the numbers for the number of bedrooms and age were much less. The house size and the number of bedrooms were on very different scales. If this hadn't been fixed, it would have resulted in incorrect predictions. To try to address this, we used a MinMaxScaler to normalize the data. This ensured that each feature contributed equally to the model by turning all of the features to the same scale. I also have to act with careful to prevent the model from becoming overfit with training data. In order to prevent overfitting, we separated the data into training and testing sets so that we could evaluate our model using fresh, untouched test data. To make sure there was no overfitting, we used statistical analysis to assess the Mean Squared Error (MSE) of the models for the training and test sets. Finally, it was difficult to implement the model without using libraries for Machine Learning such as Scikit-learn because of calculating and creating parameters using least squares. This is a relatively simple concept, it was pretty straight forward code wise to use linear regression on any given dataset but required an understanding of the mathematics behind how we calculate this. I know, it was complicated but working with that complexity made me realize how an implementation from scratch can boost your understanding of the model and sharpen up those problem-solving skills.