House Price Prediction

Submitted By:

-Swati Jaiswal

Overview

- Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement celling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

Used library in this system

- Raw data of houses taken from **Kaggle** website where **550** data of houses are taken.
- Uploaded to Jupiter notebook at the place of workstation
- Imported pandas which is used for working with datasets; allows us to analyze big data.

Used library in this system

- Import **math**: This module provides access to the mathematical functions
- Import **matplotlib** pyplot: Pyplot is a collection of functions in the popular visualization package Matplotlib. such as creating a figure, creating a plotting
- **Seaborn** is a library for making statistical graphics in Python area, plotting lines, adding plot labels, etc.

Used library in this system

- Import **numpy**: mathematical operations and scientific calculations.
- The **line warnings.filterwarnings**('ignore') is used in Python to suppress warning messages.
- **Sklearn**: sklearn is efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Code Overview

- Imported all required libraries required for the project.
- **pd.read_csv** helps to work with dataset.
- **house.head()** is used to display the data in the datasets.
- **house.shape():** It provides the dimensions of a DataFrame or Series.(rows and columns)

Code Overview

- **Info()** is used to get information of current data (i.e. Column and Datatype)
- **Isnull().sum()** is used to identify null value in datasets (i.e. among 550 data)

Code Overview

- **House.Price.describe():** to generate summary statistics for price column only.
- **House.dtypes**: to know the data type of different columns.

Code Overview

- **Duplicate().sum** is used to identify duplicate data in dataset. Hence, I have not found duplicate data (….set() to remove duplicate).
- house.select_dtypes(include=['int64', 'float64’]: This selects only the columns in the DataFrame house that contain numerical data types (int64 for integers and float64 for floating-point numbers).

Code Overview

Correlation Heatmap

- **plt.show():** Finds how strongly each number column is related to the others.
- The heatmap makes it easier to see which factors impact house prices.

Code Overview

Histogram of all numerical features

- Creates histograms(graphical representation) to show the distribution of different values in the dataset.

Function to plot categorical features

- Defines a function to make bar charts of categories like "air conditioning" and "furnishing status”.
- Helps understand how common different house features are.

Code Overview

Distribution of Price

- plt.show(): Shows how house prices are spread out (most houses are in what price range?)

Code Overview

- **Outlier**: It refers data point that significantly differs from the majority of the other data points in a dataset. These values can arise due to variability in the data, measurement errors, or other anomalies.

Code Overview

- Uses boxplots to check if any values are too high or too low compared to the normal range.
- House.get_dummies(house,columns=categorical_features, drop_first=True)
- Converts categorical features (like "furnishingstatus": 'furnished' or 'unfurnished') into numbers.
- The drop_first=True helps avoid unnecessary extra columns.

Code Overview

- X = df_encoded.drop(['price'], axis=1)#input
- y = df_encoded['price’]# output

[Separates the data into inputs (features) and output (house price to predict)]

Code Overview

- X_train, X_test, y_train, y_test = train_test_split (X,y,test_size=0.2)
- Splits the dataset into 80% training data (to train the model) and 20% test data (to check how well it predicts).

Code Overview

- Adjusts the scale of numerical values (important for Linear Regression).

Function to evaluate regression models

- Train Decision Tree Model, Train Random Forest Model, Train Linear Regression Model
- Prints how well each model performs.

Code Overview

Expected Accuracy (R^2 Score)

- ✅ Random Forest: ~85-92%
- ✅ Decision Tree: ~80-88%
- ✅ Linear Regression: ~75-85%

Code Overview

Predicting price for a new house

- Creates a new house with custom features.
- Uses the trained model to predict its price.


Thank You...