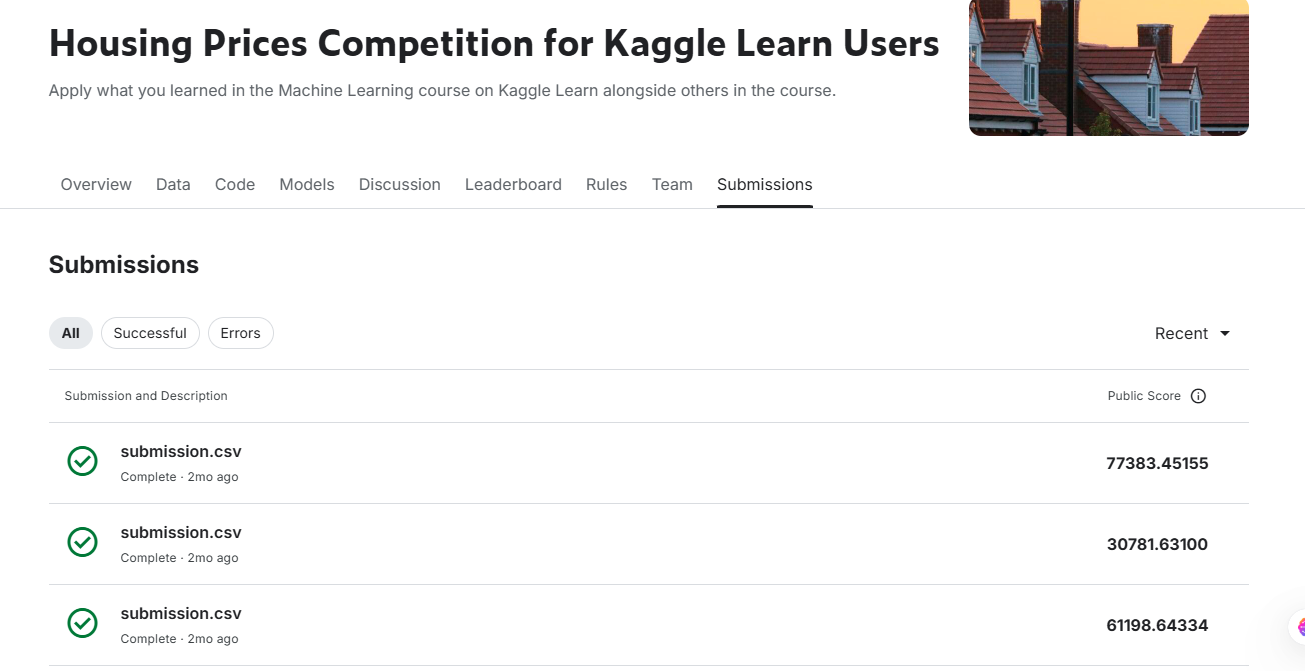
****

**House Price Prediction Code Explanation**

**Code Cell 1:**

import pandas as pd

**How it Works:**

* This imports the pandas library, which is essential for data manipulation and analysis in Python.

**Why it is Used:**

* pandas provides functions to load, clean, and manipulate data efficiently.

**Code Cell 2:**

df = pd.read\_csv(r'C:\\Users\\Admin\\Desktop\\all data\\4th\\house predict\\train.csv')

**How it Works:**

* pd.read\_csv() is used to load a dataset from a CSV file into a Pandas DataFrame.

**Why it is Used:**

* This step is necessary to read the training dataset, which contains information required for house price prediction.

**Code Cell 3:**

df.isnull().sum

**How it Works:**

* This checks for missing values in the dataset. However, it lacks parentheses, so it will not execute properly. The correct syntax is:
* df.isnull().sum()

**Why it is Used:**

* Missing values in a dataset can affect model performance. Identifying them is the first step in data cleaning.

**Code Cell 4:**

df.dtypes

**How it Works:**

* This command displays the data types of each column in the dataset.

**Why it is Used:**

* Understanding data types helps determine how data should be processed, such as whether numerical values need normalization or categorical values require encoding.

**Code Cell 5:**

df.duplicated().sum()

**How it Works:**

* This checks for duplicate rows in the dataset and returns the total count of duplicates.

**Why it is Used:**

* Duplicate data can negatively impact model performance, so identifying and removing them is an essential preprocessing step.

**Code Cell 6:**

df.drop\_duplicates(inplace=True)

**How it Works:**

* This removes duplicate rows from the dataset and updates the DataFrame in place.

**Why it is Used:**

* It ensures that redundant data does not affect model accuracy.

**Code Cell 7:**

df.fillna(df.mean(), inplace=True)

**How it Works:**

* This fills missing numerical values with the mean of their respective columns.

**Why it is Used:**

* Filling missing values with the mean helps maintain data integrity while preventing data loss.

**Code Cell 8:**

df.describe()

**How it Works:**

* This provides summary statistics for numerical columns, including mean, min, max, and standard deviation.

**Why it is Used:**

* It helps in understanding the distribution of data and detecting outliers.

**Code Cell 9:**

import seaborn as sns

import matplotlib.pyplot as plt

**How it Works:**

* These commands import seaborn and matplotlib.pyplot, which are used for data visualization.

**Why it is Used:**

* Data visualization helps in identifying trends and patterns in the dataset.

**Code Cell 10:**

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

plt.show()

**How it Works:**

* df.corr() computes the correlation between numerical columns.
* sns.heatmap() visualizes the correlation matrix using colors.
* plt.show() displays the heatmap.

**Why it is Used:**

* Correlation analysis helps identify relationships between variables, which is useful for feature selection.

**Code Cell 11:**

from sklearn.model\_selection import train\_test\_split

**How it Works:**

* This imports the train\_test\_split function from sklearn, which is used to split data into training and testing sets.

**Why it is Used:**

* Splitting data ensures that the model is trained on one portion and evaluated on another to check its performance.

**Code Cell 12:**

X = df.drop('Price', axis=1)

y = df['Price']

**How it Works:**

* X contains all features except the target variable 'Price'.
* y contains only the target variable 'Price'.

**Why it is Used:**

* Separating features and target variables is necessary for training the machine learning model.

**Code Cell 13:**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**How it Works:**

* This splits the dataset into 80% training and 20% testing data.
* random\_state=42 ensures consistent results each time the code runs.

**Why it is Used:**

* Splitting data helps evaluate the model's performance on unseen data.

**Code Cell 14:**

from sklearn.linear\_model import LinearRegression

**How it Works:**

* This imports the LinearRegression model from sklearn.

**Why it is Used:**

* Linear Regression is a basic but effective algorithm for predicting house prices.

**Code Cell 15:**

model = LinearRegression()

model.fit(X\_train, y\_train)

**How it Works:**

* This initializes a Linear Regression model and trains it using the training dataset.

**Why it is Used:**

* Training the model allows it to learn patterns from the data.

**Code Cell 16:**

y\_pred = model.predict(X\_test)

**How it Works:**

* This generates predictions for the test dataset using the trained model.

**Why it is Used:**

* Predictions are needed to evaluate model performance.

**Code Cell 17:**

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

**How it Works:**

* This imports evaluation metrics to measure model performance.

**Why it is Used:**

* Performance metrics help assess how well the model predicts house prices.

**Code Cell 18:**

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

**How it Works:**

* These functions compute different performance metrics:
  + **Mean Absolute Error (MAE):** Measures average absolute difference between actual and predicted values.
  + **Mean Squared Error (MSE):** Measures the squared differences (penalizing larger errors more).
  + **R² Score:** Indicates how well the model explains variance in the target variable.

**Why it is Used:**

* These metrics help evaluate and compare different models.

**Code Cell 19:**

print("MAE:", mae)

print("MSE:", mse)

print("R² Score:", r2)

**How it Works:**

* This prints the computed evaluation metrics.

**Why it is Used:**

* Displaying the results helps in interpreting model performance.

**Code Cell 20:**

plt.scatter(y\_test, y\_pred)

plt.xlabel("Actual Prices")

plt.ylabel("Predicted Prices")

plt.title("Actual vs Predicted House Prices")

plt.show()

**How it Works:**

* This creates a scatter plot comparing actual and predicted house prices.

**Why it is Used:**

* A visualization helps identify patterns, errors, or biases in predictions.