Task

Implement Linear and Logistic Regression on real-world datasets. Implementing Linear Regression

Upload file

Subtask:

Create a cell for uploading the csv files.

Reasoning: Create a code cell to handle the file upload process using google.colab.files.upload and print a message to the user.

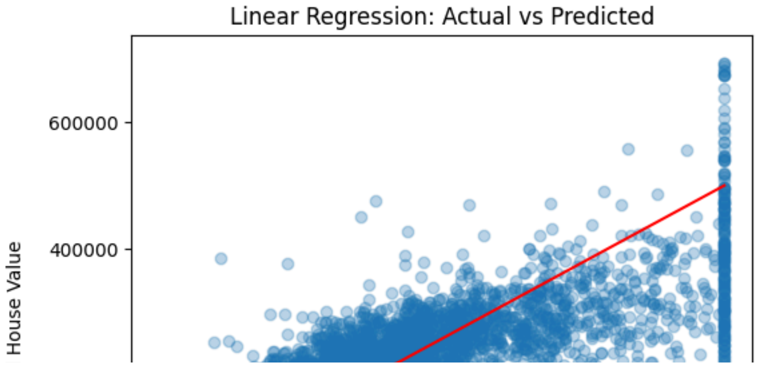
```
# Linear Regression on California Housing Dataset

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# ---------------------------
# 1. Load dataset
# ---------------------------
df = pd.read_csv("housing.csv")

print(df.head())
print(df.info())

# ---------------------------
# 2. Handle missing values
# ---------------------------
df['total_bedrooms'] = df['total_bedrooms'].fillna(df['total_bedrooms'].median())

# ---------------------------
# 3. Features & target
# ---------------------------
X = df.drop("median_house_value", axis=1)
y = df["median_house_value"]

# ---------------------------
# 4. Identify column types
# ---------------------------
numeric_features = X.select_dtypes(include=[np.number]).columns
categorical_features = ['ocean_proximity']

# ---------------------------
# 5. Preprocessing pipeline
# ---------------------------
numeric_transformer = Pipeline(steps=[
    ("scaler", StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ("onehot", OneHotEncoder(handle_unknown="ignore"))
])

preprocessor = ColumnTransformer(
    transformers=[
        ("num", numeric_transformer, numeric_features),
        ("cat", categorical_transformer, categorical_features)
    ]
)

# ---------------------------
# 6. Train-test split
# ---------------------------
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

```
# ---------------------------
# 7. Linear Regression model
# ---------------------------
model = Pipeline(steps=[
    ("preprocessor", preprocessor),
    ("regressor", LinearRegression())
])

# ---------------------------
# 8. Train model
# ---------------------------
model.fit(X_train, y_train)

# ---------------------------
# 9. Predictions
# ---------------------------
y_pred = model.predict(X_test)

# ---------------------------
# 10. Evaluation
# ---------------------------
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse:.2f}")
print(f"R² Score: {r2:.4f}")

# ---------------------------
# 11. Visualization
# ---------------------------
plt.figure(figsize=(6,6))
plt.scatter(y_test, y_pred, alpha=0.3)
plt.xlabel("Actual House Value")
plt.ylabel("Predicted House Value")
plt.title("Linear Regression: Actual vs Predicted")
plt.plot([y_test.min(), y_test.max()],
         [y_test.min(), y_test.max()],
         color="red")
plt.show()
```

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0    -122.23     37.88                41.0        880.0           129.0   
1    -122.22     37.86                21.0       7099.0          1106.0   
2    -122.24     37.85                52.0       1467.0           190.0   
3    -122.25     37.85                52.0       1274.0           235.0   
4    -122.25     37.85                52.0       1627.0           280.0   

   population  households  median_income  median_house_value ocean_proximity  
0       322.0       126.0         8.3252            452600.0        NEAR BAY  
1      2401.0      1138.0         8.3014            358500.0        NEAR BAY  
2       496.0       177.0         7.2574            352100.0        NEAR BAY  
3       558.0       219.0         5.6431            341300.0        NEAR BAY  
4       565.0       259.0         3.8462            342200.0        NEAR BAY  
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
None
RMSE: 70060.52
R² Score: 0.6254
```


Linear Regression: Actual vs Predicted
House Value