SAS Viya CODE  
proc contents data=WORK.IMPORT;   
run;  
data house\_data\_cleaned;  
    set WORK.IMPORT;  
    length Amount\_Lac 8;  
    /\* Fix: Reference the column using name literal \*/  
    Amount\_Upper = upcase(strip('Amount(in rupees)'n));  
    if index(Amount\_Upper, 'LAC') then do;  
        Amount\_Lac = input(scan(Amount\_Upper, 1, ' '), best.);  
    end;  
    else if index(Amount\_Upper, 'CR') then do;  
        Amount\_Lac = input(scan(Amount\_Upper, 1, ' '), best.) \* 100;  
    end;  
    drop 'Amount(in rupees)'n Amount\_Upper;  
run;  
  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    length Carpet\_Num 8;  
    /\* Step 1: Handle variable with space using name literal \*/  
    Carpet\_Lower = lowcase(strip('Carpet Area'n));  
    /\* Step 2: Extract numeric value \*/  
    Area\_Value = input(compress(scan(Carpet\_Lower, 1, ' '), ','), best.);  
    /\* Step 3: Convert to sqft \*/  
    if index(Carpet\_Lower, 'acre') then Carpet\_Num = Area\_Value \* 43560;  
    else if index(Carpet\_Lower, 'bigha') then Carpet\_Num = Area\_Value \* 27225;  
    else if index(Carpet\_Lower, 'cent') then Carpet\_Num = Area\_Value \* 435.6;  
    else if index(Carpet\_Lower, 'ground') then Carpet\_Num = Area\_Value \* 2400;  
    else if index(Carpet\_Lower, 'kanal') then Carpet\_Num = Area\_Value \* 5445;  
    else if index(Carpet\_Lower, 'marla') then Carpet\_Num = Area\_Value \* 272.25;  
    else if index(Carpet\_Lower, 'sqm') or index(Carpet\_Lower, 'sq meter') then Carpet\_Num = Area\_Value \* 10.7639;  
    else if index(Carpet\_Lower, 'sqyrd') or index(Carpet\_Lower, 'sq yard') then Carpet\_Num = Area\_Value \* 9;  
    else if index(Carpet\_Lower, 'sqft') then Carpet\_Num = Area\_Value;  
    else Carpet\_Num = .;  
    drop 'Carpet Area'n Carpet\_Lower Area\_Value;  
run;  
  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    length Floor\_Num 8;  
    /\* Use lowercase version for easier checks \*/  
    Floor\_Lower = lowcase(strip(Floor));  
    /\* Extract numeric floor or convert label \*/  
    if index(Floor\_Lower, 'ground') then Floor\_Num = 0;  
    else if index(Floor\_Lower, 'lower basement') then Floor\_Num = -1;  
    else if index(Floor\_Lower, 'upper basement') then Floor\_Num = -2;  
    else Floor\_Num = input(scan(Floor\_Lower, 1, ' '), best.);  
    drop Floor Floor\_Lower;  
run;  
  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    length Title\_Lower Place\_From\_Title $100;  
    /\* Convert to lowercase for case-insensitive search \*/  
    Title\_Lower = lowcase(Title);  
    /\* Look for "sale in" first \*/  
    pos\_in = index(Title\_Lower, "sale in");  
    /\* If "sale in" is found, extract text after it \*/  
    if pos\_in > 0 then Place\_From\_Title = substr(Title, pos\_in + 8);  
    /\* Else look for just "sale" \*/  
    else do;  
        pos\_sale = index(Title\_Lower, "sale");  
        if pos\_sale > 0 then Place\_From\_Title = substr(Title, pos\_sale + 5);  
        else Place\_From\_Title = "";  
    end;  
    drop Title\_Lower pos\_in pos\_sale;  
run;  
  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    /\* Fill Society ONLY if it's missing AND Place\_From\_Title is available \*/  
    if missing(Society) then Society = Place\_From\_Title;  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    length Super\_Area\_Num 8;  
    /\* Step 1: Standardize case and remove extra space \*/  
    Super\_Lower = lowcase(strip('Super Area'n));  
    /\* Step 2: Extract numeric value from string \*/  
    Area\_Value = input(compress(scan(Super\_Lower, 1, ' '), ','), best.);  
    /\* Step 3: Apply conversion to sqft based on unit \*/  
    if index(Super\_Lower, 'acre') then Super\_Area\_Num = Area\_Value \* 43560;  
    else if index(Super\_Lower, 'bigha') then Super\_Area\_Num = Area\_Value \* 27225;  
    else if index(Super\_Lower, 'cent') then Super\_Area\_Num = Area\_Value \* 435.6;  
    else if index(Super\_Lower, 'ground') then Super\_Area\_Num = Area\_Value \* 2400;  
    else if index(Super\_Lower, 'kanal') then Super\_Area\_Num = Area\_Value \* 5445;  
    else if index(Super\_Lower, 'marla') then Super\_Area\_Num = Area\_Value \* 272.25;  
    else if index(Super\_Lower, 'sqm') or index(Super\_Lower, 'sq meter') then Super\_Area\_Num = Area\_Value \* 10.7639;  
    else if index(Super\_Lower, 'sqyrd') or index(Super\_Lower, 'sq yard') then Super\_Area\_Num = Area\_Value \* 9;  
    else if index(Super\_Lower, 'sqft') then Super\_Area\_Num = Area\_Value;  
    else Super\_Area\_Num = .;  
    drop 'Super Area'n Super\_Lower Area\_Value;  
run;  
  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    length Carpet\_Num 8;  
    if missing(Carpet\_Num) and not missing(Super\_Area\_Num) then  
        Carpet\_Num = 0.75 \* Super\_Area\_Num;  
run;  
  
proc means data=house\_data\_cleaned median;  
    var Bathroom;  
run;  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    if missing(Bathroom) then Bathroom = 2;  
run;  
proc means data=house\_data\_cleaned median;  
    var Floor\_Num;  
run;  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    if missing(Floor\_Num) then Floor\_Num = 2;  
run;  
  
  
data house\_data\_cleaned;  
  
    set house\_data\_cleaned;  
    length BHK 8 Property\_Type $20 BHK\_Combined $30;  
    /\* Step 1: Extract BHK number from Title \*/  
  
    if prxmatch("/\d+\s\*BHK/i", Title) then  
  
        BHK = input(scan(Title, 1, ' '), best.);  
    /\* Step 2: Extract Property Type from Title \*/  
  
    if index(upcase(Title), 'FLAT') then Property\_Type = 'Flat';  
  
    else if index(upcase(Title), 'VILLA') then Property\_Type = 'Villa';  
  
    else if index(upcase(Title), 'APARTMENT') then Property\_Type = 'Apartment';  
  
    else Property\_Type = 'Other';  
    /\* Step 3: Merge into BHK\_Combined \*/  
  
    if not missing(BHK) and not missing(Property\_Type) then  
  
        BHK\_Combined = cats(put(BHK, 1.), ' BHK ', Property\_Type);  
  
    else if not missing(Property\_Type) then  
  
        BHK\_Combined = Property\_Type;  
  
    else if not missing(BHK) then  
  
        BHK\_Combined = cats(put(BHK, 1.), ' BHK');  
  
    else BHK\_Combined = 'Unknown';  
  
run;  
  
data house\_data\_cleaned;  
    set house\_data\_cleaned;  
    if missing(Amount\_Lac) then delete;  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;  
  
    /\* Handle missing character fields \*/  
    if missing(Transaction) then Transaction = "Unknown";  
    if missing(Furnishing) then Furnishing = "Unknown";  
    if missing(Ownership) then Ownership = "Unknown";  
    if missing(Facing) then Facing = "Unknown";  
    if missing(Overlooking) then Overlooking = "Unknown";  
  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;

    length Price\_Per\_Sqft\_Cat $15;

    if missing('Price (in rupees)'n) then   
        Price\_Per\_Sqft\_Cat = "Unknown";  
    else   
        Price\_Per\_Sqft\_Cat = strip(put('Price (in rupees)'n, 8.));  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;

    length Balcony\_Cat $10;

    if missing(Balcony) then   
        Balcony\_Cat = "Unknown";  
    else   
        Balcony\_Cat = strip(put(Balcony, 8.));  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;

    length Car\_Parking\_Cat $10;

    if missing('Car Parking'n) then   
        Car\_Parking\_Cat = "Unknown";  
    else   
        Car\_Parking\_Cat = strip(put('Car Parking'n, 8.));  
run;

data house\_data\_cleaned;  
    set house\_data\_cleaned;

    length Carpet\_Area\_Cat $15;

    if missing(Carpet\_Num) then   
        Carpet\_Area\_Cat = "Unknown";  
    else   
        Carpet\_Area\_Cat = strip(put(Carpet\_Num, 8.));  
run;

Regression\_model\_python\_code

# Import required libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

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

from sklearn.preprocessing import OneHotEncoder, PolynomialFeatures

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline, make\_pipeline

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

# Load cleaned dataset

df = pd.read\_csv('HOUSE\_DATA\_CLEANED\_v10\_Flat\_dataset.csv')

# Convert relevant columns to numeric

df['Carpet\_Area\_Cat'] = pd.to\_numeric(df['Carpet\_Area\_Cat'], errors='coerce')

df['Balcony\_Cat'] = pd.to\_numeric(df['Balcony\_Cat'], errors='coerce')

# Drop rows with missing Carpet or Balcony info for simplicity

df = df.dropna(subset=['Carpet\_Area\_Cat', 'Balcony\_Cat'])

# Define target and features

target = 'Amount\_Lac'

features = ['Transaction', 'Furnishing', 'Ownership', 'BHK\_Combined',

'Floor\_Num', 'Carpet\_Area\_Cat', 'Balcony\_Cat']

X = df[features]

y = df[target]

# Train-test split

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

# Define preprocessing for categorical and numeric features

cat\_cols = ['Transaction', 'Furnishing', 'Ownership', 'BHK\_Combined']

num\_cols = ['Floor\_Num', 'Carpet\_Area\_Cat', 'Balcony\_Cat']

preprocessor = ColumnTransformer(

transformers=[

('cat', OneHotEncoder(handle\_unknown='ignore'), cat\_cols),

('num', 'passthrough', num\_cols)

]

)

# -------------------------

# Linear Regression Model

# -------------------------

linear\_pipeline = Pipeline(steps=[

('preprocessor', preprocessor),

('regressor', LinearRegression())

])

linear\_pipeline.fit(X\_train, y\_train)

linear\_preds = linear\_pipeline.predict(X\_test)

linear\_mse = mean\_squared\_error(y\_test, linear\_preds)

linear\_rmse = np.sqrt(linear\_mse)

# -------------------------

# Polynomial Regression (degree 2) on log(Amount\_Lac)

# -------------------------

X\_poly\_train = preprocessor.fit\_transform(X\_train)

X\_poly\_test = preprocessor.transform(X\_test)

poly\_model = make\_pipeline(

PolynomialFeatures(degree=2, include\_bias=False),

LinearRegression()

)

y\_train\_log = np.log1p(y\_train)

poly\_model.fit(X\_poly\_train, y\_train\_log)

poly\_preds\_log = poly\_model.predict(X\_poly\_test)

poly\_preds = np.expm1(poly\_preds\_log)

poly\_mse = mean\_squared\_error(y\_test, poly\_preds)

poly\_rmse = np.sqrt(poly\_mse)

# -------------------------

# Visualization of predictions

# -------------------------

plt.figure(figsize=(10, 5))

# Linear Regression Plot

plt.subplot(1, 2, 1)

plt.scatter(np.log1p(y\_test), np.log1p(linear\_preds), alpha=0.3)

plt.plot([np.log1p(y\_test).min(), np.log1p(y\_test).max()],

[np.log1p(y\_test).min(), np.log1p(y\_test).max()], color='red')

plt.xlabel('Actual log(Amount\_Lac)')

plt.ylabel('Predicted log(Amount\_Lac)')

plt.title('Linear Regression: log(Amount\_Lac)')

# Polynomial Regression Plot

plt.subplot(1, 2, 2)

plt.scatter(np.log1p(y\_test), poly\_preds\_log, alpha=0.3)

plt.plot([np.log1p(y\_test).min(), np.log1p(y\_test).max()],

[np.log1p(y\_test).min(), np.log1p(y\_test).max()], color='red')

plt.xlabel('Actual log(Amount\_Lac)')

plt.ylabel('Predicted log(Amount\_Lac)')

plt.title('Polynomial Regression: log(Amount\_Lac)')

plt.tight\_layout()

plt.savefig('Actual\_vs\_Predicted\_Log\_Amount\_Lac.png')

plt.show()

# Print evaluation results

print(f'Linear Regression RMSE: {linear\_rmse:.2f}')

print(f'Polynomial Regression RMSE: {poly\_rmse:.2f}')