```
from google.colab import drive
```

New Section

```
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
```

```
data = pd.read_csv('/content/drive/MyDrive/Hyderabad-Data2.csv')
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder, OneHotEncoder
from sklearn.impute import SimpleImputer
```

```
data.head()
```

	active	amenities	balconies	bathroom	combineDescription	completeStreetName	deposit	facing	facir
0	True	{"LIFT":true,"GYM":false,"INTERNET":false,"AC"...	3	3	NaN	Shreya carnation, Block I, NCB Enclave, Gachib...	90000	W	
1	True	{"LIFT":false,"GYM":false,"INTERNET":false,"AC...	1	2	NaN	Inner Ring Rd, near RTO Bandlaguda South Zone	45000	E	
2	True	{"LIFT":true,"GYM":true,"INTERNET":false,"AC":...	3	3	NaN	Rd Number 2, Shirdi Sai Nagar, Manikonda, Hyde...	80000	E	
3	True	{"LIFT":false,"GYM":false,"INTERNET":false,"AC...	1	2	NaN	Plot No. 44, Road No. 1/A, kakatiya colony, LB...	18000	W	
4	True	{"LIFT":true,"GYM":false,"INTERNET":false,"AC"...	2	2	NaN	Madhapur HUDA Techno Enclave, Near MaxCure Su...	80000	E	

5 rows × 36 columns

```
data.describe()
```

	bathroom	combineDescription	deposit	floor	property_age	property_size	rent_amount	totalFloor	weight
count	9820.000000	0.0	9.820000e+03	9820.000000	9820.000000	9820.000000	9820.000000	9820.000000	0.0
mean	1.934216	NaN	3.426805e+04	2.184725	3.759165	1087.364460	15291.173931	4.044501	NaN
std	0.747437	NaN	4.276260e+04	2.533791	3.477064	542.146286	9587.311735	4.199083	NaN
min	1.000000	NaN	1.000000e+00	0.000000	-1.000000	0.000000	0.000000	0.000000	NaN
25%	1.000000	NaN	1.600000e+04	1.000000	1.000000	720.000000	9000.000000	2.000000	NaN
50%	2.000000	NaN	2.600000e+04	2.000000	3.000000	1030.000000	13000.000000	3.000000	NaN
75%	2.000000	NaN	4.000000e+04	3.000000	5.000000	1350.000000	19000.000000	5.000000	NaN
max	8.000000	NaN	2.000000e+06	31.000000	10.000000	10000.000000	100000.000000	80.000000	NaN

```
New_Data = data.drop(['active', 'amenities', 'balconies',
       'completeStreetName', 'facing', 'facingDesc',
        'id', 'isMaintenance', 'lift', 'loanAvailable',
         'ownerName','waterSupply',
       'reactivationSource',
        'shortUrl', 'swimmingPool',
        'weight','combineDescription','deposit','furnishingDesc','gym','parking','parkingDesc',
                     'location','localityId','propertyTitle','propertyType','sharedAccomodation']
,axis=1)
```

```
# after removing irrelevant columns
New_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9820 entries, 0 to 9819
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   bathroom           9820 non-null   int64 
 1   floor              9820 non-null   int64 
 2   locality           9814 non-null   object
 3   maintenanceAmount  9820 non-null   object
 4   property_age       9820 non-null   int64 
 5   property_size      9820 non-null   int64 
 6   rent_amount        9820 non-null   int64 
 7   totalFloor         9820 non-null   int64 
 8   type_bhk           9820 non-null   object
dtypes: int64(6), object(3)
memory usage: 690.6+ KB
```

```
New_Data.replace({'parking':{'BOTH':0,'TWO_WHEELER':1,'FOUR_WHEELER':2,'NONE':3}},inplace=True)
New_Data.replace({'type_bhk':{'RK1':0.5,'BHK1':1,'BHK2':2,'BHK3':3,'BHK4':4,'BHK4PLUS':5}},inplace=True)
New_Data.replace({'maintenanceAmount':{'None':int(0)}},inplace=True)
New_Data.replace({'furnishingDesc':{'Unfurnished':0,'Semi':1,'Full':2}},inplace=True)
```

```
# Assuming 'object_column' contains strings and 'int_column' contains integ
New_Data['total_price'] = New_Data['maintenanceAmount'].astype(str) + New_Data['rent_amount'].astype(str)
```

```
New_Data['total_price']
```

```
0       200028000
1          015000
2       100016000
3         5009000
4       200032500
          ...    
9815       013000
9816        05000
9817      2008500
9818        08000
9819        17000
Name: total_price, Length: 9820, dtype: object
```

```
New_Data = pd.concat([New_Data.iloc[:, :-1], New_Data['total_price']], axis=1)
```

```
New_Data.head()
```

	bathroom	floor	locality	maintenanceAmount	property_age	property_size	rent_amount	totalFloor	type_bhk	total_price
0	3	3	Gachibowli	2000	5	2200	28000	5	3.0	200028000
1	2	2	Chandrayangutta	0	1	1200	15000	2	3.0	015000
2	3	0	Manikonda	1000	0	1800	16000	3	3.0	100016000
3	2	2	LB Nagar	500	0	750	9000	2	2.0	5009000
4	2	2	HITEC City	2000	5	1250	32500	5	2.0	200032500

```
New_Data = New_Data.drop(['rent_amount', 'maintenanceAmount'],axis=1)
```

```
New_Data.head()
```

	bathroom	floor	locality	property_age	property_size	totalFloor	type_bhk	total_price
0	3	3	Gachibowli	5	2200	5	3.0	200028000
1	2	2	Chandrayangutta	1	1200	2	3.0	015000
2	3	0	Manikonda	0	1800	3	3.0	100016000
3	2	2	LB Nagar	0	750	2	2.0	5009000
4	2	2	HITEC City	5	1250	5	2.0	200032500

```
# after removing irrelevant columns
New_Data.isnull().sum()
```

```
bathroom         0
floor            0
locality         6
property_age     0
property_size    0
totalFloor       0
type_bhk         0
total_price      0
dtype: int64
```

```
# Fill missing values in the "locality" column with 0
New_Data['locality'].fillna(0, inplace=True)

# Check for missing values after filling
missing_values = New_Data.isnull().sum()

# Print the updated DataFrame with missing values filled
print(New_Data)

# Print the count of missing values in each column
print("Missing Values Count:\n", missing_values)
```

```
      bathroom  floor                        locality  property_age  \
0            3      3                      Gachibowli             5
1            2      2                 Chandrayangutta             1
2            3      0                       Manikonda             0
3            2      2                         LB Nagar             0
4            2      2                      HITEC City             5
...        ...    ...                             ...           ...
9815         2      1                       King Koti             5
9816         1      0                  Jagadgiri Gutta             5
9817         1      0  Jyothi Nagar, Ramachandra Puram            10
9818         1      1                      Kukatpally            10
9819         1      3                       Yusufguda            10

      property_size  totalFloor  type_bhk total_price
0              2200           5       3.0   200028000
1              1200           2       3.0      015000
2              1800           3       3.0   100016000
3               750           2       2.0     5009000
4              1250           5       2.0   200032500
...             ...         ...       ...         ...
9815            800           2       2.0      013000
9816            150           0       0.5       05000
9817            550           3       1.0     2008500
9818            500           2       1.0       08000
9819            560           4       2.0       17000

[9820 rows x 8 columns]
Missing Values Count:
 bathroom         0
floor            0
locality         0
property_age     0
property_size    0
totalFloor       0
type_bhk         0
total_price      0
dtype: int64
```

```
New_Data.isnull().sum()
```

```
bathroom         0
floor            0
locality         0
property_age     0
property_size    0
totalFloor       0
type_bhk         0
total_price      0
dtype: int64
```

```
New_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9820 entries, 0 to 9819
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   bathroom       9820 non-null   int64
 1   floor          9820 non-null   int64
 2   locality       9820 non-null   object
 3   property_age   9820 non-null   int64
 4   property_size  9820 non-null   int64
 5   totalFloor     9820 non-null   int64
 6   type_bhk       9820 non-null   float64
 7   total_price    9820 non-null   object
dtypes: float64(1), int64(5), object(2)
memory usage: 613.9+ KB
```

```
New_Data.head()
```

	bathroom	floor	locality	property_age	property_size	totalFloor	type_bhk	total_price
0	3	3	Gachibowli	5	2200	5	3.0	200028000
1	2	2	Chandrayangutta	1	1200	2	3.0	015000
2	3	0	Manikonda	0	1800	3	3.0	100016000
3	2	2	LB Nagar	0	750	2	2.0	5009000
4	2	2	HITEC City	5	1250	5	2.0	200032500

```
# Assuming 'data' is your DataFrame
New_Data = pd.get_dummies(New_Data, columns=['locality'], prefix='locality')
```

```
New_Data.head()
```

	bathroom	floor	property_age	property_size	totalFloor	type_bhk	total_price	locality_0	locality_ Chanda Nagar	locality_ Nacharam	...	locality_
0	3	3	5	2200	5	3.0	200028000	0	0	0	...	
1	2	2	1	1200	2	3.0	015000	0	0	0	...	
2	3	0	0	1800	3	3.0	100016000	0	0	0	...	
3	2	2	0	750	2	2.0	5009000	0	0	0	...	
4	2	2	5	1250	5	2.0	200032500	0	0	0	...	

5 rows × 1779 columns

```
X = New_Data.drop(columns=['total_price'], axis=1)
y = New_Data['total_price']
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.linear_model import LinearRegression

# Create a Linear Regression model
model = LinearRegression()

# Train the model on the training data
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
# Make predictions
predictions = model.predict(X_test)


from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

# Evaluate the model
mae = mean_absolute_error(y_test, predictions)
mse = mean_squared_error(y_test, predictions)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, predictions)




print(f"Mean Absolute Error: {mae}")
print(f"Mean Squared Error: {mse}")
print(f"Root Mean Squared Error: {rmse}")
print(f"R-squared: {r2}")
```

```
Mean Absolute Error: 331739597084320.9
Mean Squared Error: 1.0515424152685784e+31
Root Mean Squared Error: 3242749474240306.5
R-squared: -527799281337713.7
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Split your data into features (X) and the target variable (y)
X = New_Data.drop('total_price', axis=1)
y = New_Data['total_price']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create a Linear Regression model
model = LinearRegression()

# Train the model on the training data
model.fit(X_train, y_train)

# Now, the model is fitted, and you can make predictions on new data

# Example usage:
input_features = X_test.sample(1)  # Get a random sample from your test data

predicted_rent = model.predict(input_features)
print(f"Predicted Rent Price: {predicted_rent[0]}")
```

```
Predicted Rent Price: -920311959769818.2
```