```
#Multiple Linear Regression
#Problem Statement: Predict house prices based on multiple features (e.g., area, number of
#rooms, location) using Multiple Linear Regression.
#Activities:
#• Feature selection
#• Model training
#• Residual analysis
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
import pandas as pd
import os
import kagglehub

# Download and locate dataset
path = kagglehub.dataset_download("lespin/house-prices-dataset")

for file in os.listdir(path):
    if file.endswith('.csv'):
        file_path = os.path.join(path, file)
        break

df = pd.read_csv(file_path)
print("✅ Dataset Loaded from:", file_path)
print("\n📋 Columns in the dataset:\n")
print(df.columns.tolist())
print("\n🧭 First 5 rows:\n")
print(df.head())
```

```
Using Colab cache for faster access to the 'house-prices-dataset' dataset.
✅ Dataset Loaded from: /kaggle/input/house-prices-dataset/sample_submission.csv

📋 Columns in the dataset:

['Id', 'SalePrice']

🧭 First 5 rows:

     Id      SalePrice
0  1461  169277.052498
1  1462  187758.393989
2  1463  183583.683570
3  1464  179317.477511
4  1465  150730.079977
```

```
# ---------------------------------------------
# 🏡 MULTIPLE LINEAR REGRESSION - HOUSE PRICES
# ---------------------------------------------

# STEP 1: Import required libraries
import kagglehub
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from statsmodels.stats.outliers_influence import variance_inflation_factor
import os

# STEP 2: Download dataset from KaggleHub
path = kagglehub.dataset_download("lespin/house-prices-dataset")
print("✅ Path to dataset files:", path)

# STEP 3: Locate and load the training dataset
for file in os.listdir(path):
    if "train" in file.lower() and file.endswith(".csv"):
        file_path = os.path.join(path, file)
        break

df = pd.read_csv(file_path)
print("✅ Loaded file:", file_path)
print("\n📋 Columns available:\n", df.columns.tolist())
print("\n🧭 First 5 rows:\n", df.head())
```

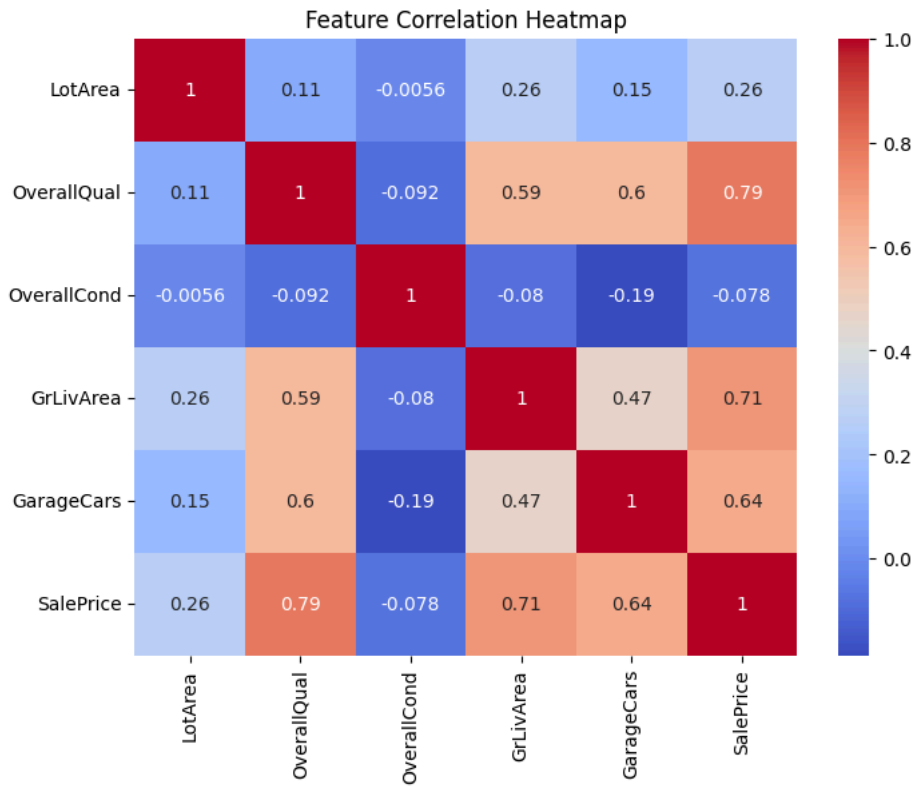
```
Using Colab cache for faster access to the 'house-prices-dataset' dataset.
✅ Path to dataset files: /kaggle/input/house-prices-dataset
✅ Loaded file: /kaggle/input/house-prices-dataset/train.csv

📋 Columns available:
 ['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilit

🧭 First 5 rows:
    Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0   1          60       RL         65.0     8450   Pave   NaN      Reg
1   2          20       RL         80.0     9600   Pave   NaN      Reg
2   3          60       RL         68.0    11250   Pave   NaN      IR1
3   4          70       RL         60.0     9550   Pave   NaN      IR1
4   5          60       RL         84.0    14260   Pave   NaN      IR1

  LandContour Utilities  ... PoolArea PoolQC Fence MiscFeature MiscVal MoSold  \
0         Lvl    AllPub  ...        0    NaN   NaN         NaN       0      2
1         Lvl    AllPub  ...        0    NaN   NaN         NaN       0      5
2         Lvl    AllPub  ...        0    NaN   NaN         NaN       0      9
3         Lvl    AllPub  ...        0    NaN   NaN         NaN       0      2
4         Lvl    AllPub  ...        0    NaN   NaN         NaN       0     12

  YrSold  SaleType  SaleCondition  SalePrice
0   2008        WD         Normal     208500
1   2007        WD         Normal     181500
2   2008        WD         Normal     223500
3   2006        WD        Abnorml     140000
4   2008        WD         Normal     250000

[5 rows x 81 columns]
```

```
# STEP 4: Basic data info
print("\n📊 Dataset Info:")
print(df.info())

# STEP 5: Select useful numeric features for regression
# (We'll choose some continuous and relevant columns)
selected_features = ['LotArea', 'OverallQual', 'OverallCond', 'GrLivArea', 'GarageCars']
target = 'SalePrice'

# Drop missing values
df = df[selected_features + [target]].dropna()
```

```
📊 Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          91 non-null     object
 7   LotShape       1460 non-null   object
 8   LandContour    1460 non-null   object
 9   Utilities      1460 non-null   object
 10  LotConfig      1460 non-null   object
 11  LandSlope      1460 non-null   object
 12  Neighborhood   1460 non-null   object
 13  Condition1     1460 non-null   object
 14  Condition2     1460 non-null   object
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
 21  RoofStyle      1460 non-null   object
 22  RoofMatl       1460 non-null   object
 23  Exterior1st    1460 non-null   object
 24  Exterior2nd    1460 non-null   object
 25  MasVnrType     588 non-null    object
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1423 non-null   object
 31  BsmtCond       1423 non-null   object
 32  BsmtExposure   1422 non-null   object
```

```
 33  BsmtFinType1   1423 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1422 non-null   object
 36  BsmtFinSF2     1460 non-null   int64
 37  BsmtUnfSF      1460 non-null   int64
 38  TotalBsmtSF    1460 non-null   int64
 39  Heating        1460 non-null   object
 40  HeatingQC      1460 non-null   object
 41  CentralAir     1460 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1460 non-null   int64
 44  2ndFlrSF       1460 non-null   int64
 45  LowQualFinSF   1460 non-null   int64
 46  GrLivArea      1460 non-null   int64
 47  BsmtFullBath   1460 non-null   int64
 48  BsmtHalfBath   1460 non-null   int64
 49  FullBath       1460 non-null   int64
 50  HalfBath       1460 non-null   int64
```

```
# STEP 6: Check correlation
plt.figure(figsize=(8,6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Feature Correlation Heatmap")
plt.show()
```




```
# STEP 7: Multicollinearity check (VIF)
X = df[selected_features]
vif = pd.DataFrame()
vif["Feature"] = X.columns
vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print("\n🔍 Variance Inflation Factor (VIF):\n", vif)
```

```
🔍 Variance Inflation Factor (VIF):
        Feature        VIF
0      LotArea   2.283834
1  OverallQual  32.668664
2  OverallCond  11.249573
3    GrLivArea  15.662862
4   GarageCars  10.944820
```

```
# STEP 8: Split data into train and test
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# STEP 9: Train model
model = LinearRegression()
```

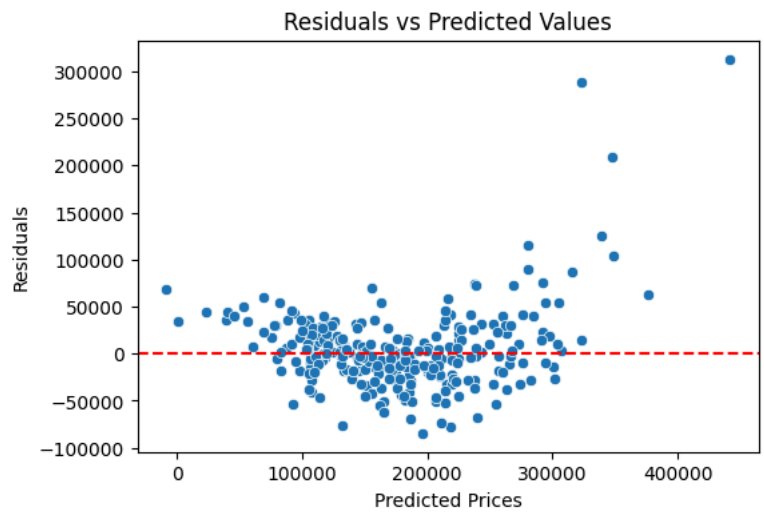
```
model.fit(X_train, y_train)

# STEP 10: Evaluate model
y_pred = model.predict(X_test)
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)

print("\n📈 Model Evaluation Metrics:")
print(f"R² Score: {r2:.3f}")
print(f"RMSE: {rmse:.2f}")
print(f"MAE: {mae:.2f}")
```

```
📈 Model Evaluation Metrics:
R² Score: 0.765
RMSE: 42413.86
MAE: 26982.19
```

```
#STEP 11: Display regression coefficients
coeff_df = pd.DataFrame({
    'Feature': selected_features,
    'Coefficient': model.coef_
})
print("\n📋 Model Coefficients:\n", coeff_df)
print(f"Intercept: {model.intercept_:.2f}")

# STEP 12: Residual Analysis
residuals = y_test - y_pred
```

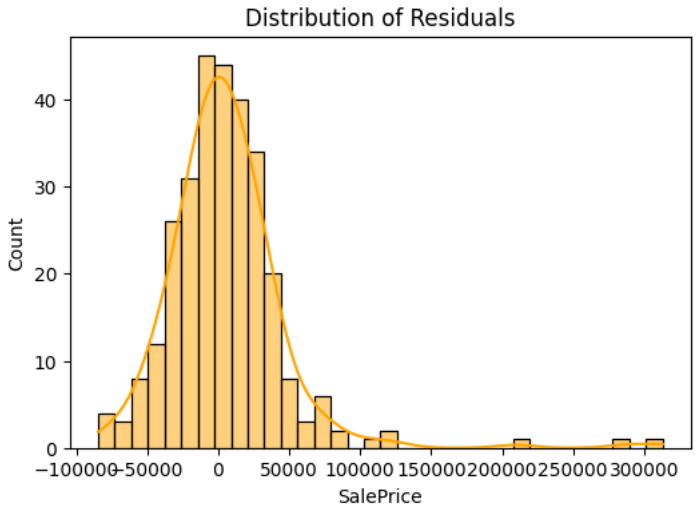
```
📋 Model Coefficients:
        Feature   Coefficient
0      LotArea      0.777139
1  OverallQual  27488.585324
2  OverallCond   1529.867736
3    GrLivArea     42.436690
4   GarageCars  21730.353113
Intercept: -107217.55
```

```
# Residual plot
plt.figure(figsize=(6,4))
sns.scatterplot(x=y_pred, y=residuals)
plt.axhline(0, color='red', linestyle='--')
plt.title("Residuals vs Predicted Values")
plt.xlabel("Predicted Prices")
plt.ylabel("Residuals")
plt.show()
```




```
#Distribution of residuals
plt.figure(figsize=(6,4))
sns.histplot(residuals, kde=True, color='orange')
plt.title("Distribution of Residuals")
```

```
plt.show()

print("\n✅ Residual analysis completed successfully!")
```




```
✅ Residual analysis completed successfully!
```