```
# Step 1: Import Libraries
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 mean_squared_error, r2_score

# Step 2: Load the dataset
data = pd.read_csv("house.csv")  # Replace with your dataset filename

# Step 3: Basic preprocessing
# Drop rows with missing target
data.dropna(subset=['SalePrice'], inplace=True)

# Select numeric features only
numeric_features = data.select_dtypes(include=[np.number])

# Fill missing values with median
numeric_features = numeric_features.fillna(numeric_features.median())

# Step 4: Feature selection
# Correlation with target
correlation = numeric_features.corr()
top_features                                                                    =
correlation['SalePrice'].sort_values(ascending=False).head(6).index.tolist()
top_features.remove('SalePrice')
print("Top correlated features with SalePrice:", top_features)

# Step 5: Define features (X) and target (y)
X = numeric_features[top_features]
y = numeric_features['SalePrice']

# Step 6: Split 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)

# Step 7: Train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Step 8: Make predictions
y_pred = model.predict(X_test)

# Step 9: Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"\nMean Squared Error: {mse:.2f}")
print(f"R² Score: {r2:.2f}")

# Step 10: Visualize predictions vs actual values
plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_test, y=y_pred)
```

```
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted House Prices")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--r')  # reference
line
plt.grid(True)
plt.show()
```