Assignment-1 Report: Linear Regression Model Analysis

This report summarizes the steps and findings of the linear regression model analysis performed on the `data.csv` dataset.

1. Data Loading and Exploration

Objective: Load the data and understand its structure and characteristics.

Code:

Summary:

Data Analysis Key Findings

- There are no missing values in the dataset.
- The data contains both numerical and object type columns.
- After preprocessing, the feature set `X` has 12 columns and the target variable `y` is 'price'.
- The data was split into training (80%) and testing (20%) sets, with `X_train` having a shape of (3680, 12), `X_test` having a shape of (920, 12), `y_train` having a shape of (3680,), and `y_test` having a shape of (920,).
- The linear regression model achieved a Mean Squared Error (MSE) of 986,921,767,056.10 and an R-squared score of 0.0323 on the test set, indicating a poor model fit.
- A scatter plot of 'sqft_living' versus 'price' shows a general positive relationship.
- A scatter plot of actual versus predicted prices shows that while there is some correlation, the predictions are not tightly clustered around the ideal diagonal line, confirming the low R-squared value.

```
display(df.isnull().sum())
display(df.info())
```

	0
date	0
price	0
bedrooms	0
bathrooms	0
sqft_living	0
sqft_lot	0
floors	0
waterfront	0
view	0
condition	0
sqft_above	0
sqft_basement	0
yr_built	0
yr_renovated	0
street	0
city	0
statezip	0
country	0

```
dtype: int64
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           4600 non-null   object
 1   price          4600 non-null   float64
 2   bedrooms       4600 non-null   float64
 3   bathrooms      4600 non-null   float64
 4   sqft_living    4600 non-null   int64
 5   sqft_lot       4600 non-null   int64
 6   floors         4600 non-null   float64
 7   waterfront     4600 non-null   int64
 8   view           4600 non-null   int64
 9   condition      4600 non-null   int64
 10  sqft_above     4600 non-null   int64
 11  sqft_basement  4600 non-null   int64
 12  yr_built       4600 non-null   int64
 13  yr_renovated   4600 non-null   int64
 14  street         4600 non-null   object
 15  city           4600 non-null   object
 16  statezip       4600 non-null   object
 17  country        4600 non-null   object
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
None
```

```python
X = df.drop(['date', 'street', 'city', 'statezip', 'country', 'price'], axis=1)
y = df['price']
display(X.head())
display(y.head())
```

	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated
0	3.0	1.50	1340	7912	1.5	0	0	3	1340	0	1955	2005
1	5.0	2.50	3650	9050	2.0	0	4	5	3370	280	1921	0
2	3.0	2.00	1930	11947	1.0	0	0	4	1930	0	1966	0
3	3.0	2.25	2000	8030	1.0	0	0	4	1000	1000	1963	0
4	4.0	2.50	1940	10500	1.0	0	0	4	1140	800	1976	1992

	price
0	313000.0
1	2384000.0
2	342000.0
3	420000.0
4	550000.0

dtype: float64

```python
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)

print("Shape of X_train:", X_train.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of y_test:", y_test.shape)
```

```
Shape of X_train: (3680, 12)
Shape of X_test: (920, 12)
Shape of y_train: (3680,)
Shape of y_test: (920,)
```

```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```python
from sklearn.metrics import mean_squared_error, r2_score

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

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

```
Mean Squared Error: 986921767056.0986
R-squared: 0.032283856632802865
```

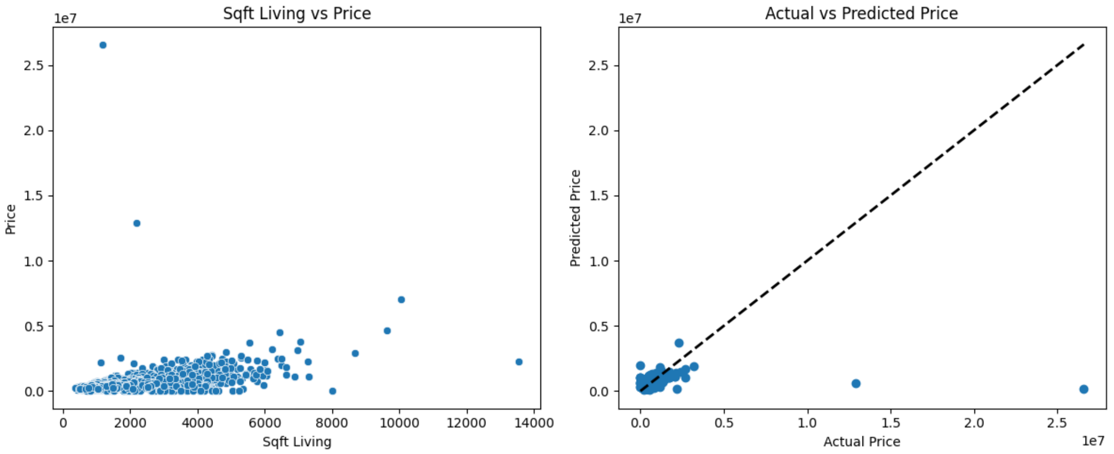
```python
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12, 5))

# Scatter plot of sqft_living vs price
plt.subplot(1, 2, 1)
sns.scatterplot(x='sqft_living', y='price', data=df)
plt.title('Sqft Living vs Price')
plt.xlabel('Sqft Living')
plt.ylabel('Price')

# Scatter plot of actual vs predicted prices
plt.subplot(1, 2, 2)
plt.scatter(y_test, y_pred)
plt.title('Actual vs Predicted Price')
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)

plt.tight_layout()
plt.show()
```



Linear Regression Model Analysis

This notebook performs a linear regression analysis on a dataset to predict house prices.

Dataset

The analysis uses data from `data.csv`. The dataset contains information about various house features and their corresponding prices.

Analysis Steps

1. **Data Loading and Exploration**: The data was loaded into a pandas DataFrame. Initial exploration included viewing the first few rows, checking for missing values, and examining data types. No missing values were found.
2. **Data Preprocessing**: Irrelevant columns (`date`, `street`, `city`, `statezip`, `country`) were dropped. The target variable (`price`) was separated from the features.
3. **Data Splitting**: The data was split into training (80%) and testing (20%) sets to train and evaluate the model.
4. **Model Building and Training**: A linear regression model from scikit-learn was instantiated and trained on the training data.
5. **Model Evaluation**: The model's performance was evaluated using Mean Squared Error (MSE) and R-squared (R2) on the test set.
    - Mean Squared Error: 986,921,767,056.10
    - R-squared: 0.0323
6. **Visualization**: Scatter plots were generated to visualize the relationship between 'sqft_living' and 'price', and to compare actual versus predicted prices.

Findings and Conclusion

The linear regression model achieved a very low R-squared score (0.0323), indicating that it explains only a small portion of the variance in the house prices. The scatter plot of actual versus predicted prices also shows a poor fit, with predictions not closely aligned with the actual values.

The results suggest that a simple linear regression model is not sufficient to accurately predict house prices with this dataset and the selected features.