› Set Up

↳ 1 cell hidden

⌄ Data Preparation

```
# Locate the first row of 2019

# Seperate the dataset into smaller part to display
chunk_size = 700000
chunks = pd.read_csv('Combined Listing Data [Detailed].csv', chunksize=chunk_size)
df = next(chunks)

# # Create mask for rows in year 2019
df["year"] = pd.to_datetime(df['calendar_last_scraped'], errors="coerce").dt.year
mask_2019 = df['year'] == 2019

# # Grab the first row that meets the condition
first_2019_index = df[mask_2019].index[0]
print(first_2019_index)
```

```
<ipython-input-2-c0da483c02f7>:6: DtypeWarning: Columns (0,1,13,14,16,17,25,38,41,46,51,53,54,65,66,68,71,72,74,85,86,93,
  df = next(chunks)
628693
```

```
# Read data and clean data
CSV_FILE = 'Combined Listing Data [Detailed].csv'

start_row = 628694    # inclusive
chunk_size = 500000   # number of rows per chunk

wanted_columns = [
    'calendar_last_scraped',
    'review_scores_accuracy',
    'review_scores_cleanliness',
    'review_scores_checkin',
    'review_scores_communication',
    'review_scores_location',
    'review_scores_value',
    'host_response_rate',
    'host_is_superhost',
    'instant_bookable',
    'description',
    'cleaning_fee'
]

# We will keep reading until we can no longer get any rows
all_chunks = []         # list to store each chunk’s DataFrame
current_skip = start_row

while True:
    # skiprows=range(1, current_skip) means skip lines 1..(current_skip-1),
    # but keep line 0 as the header.
    skip_list = range(1, current_skip)  # can be large but 'engine="python"' can handle it

    df_chunk = pd.read_csv(
        CSV_FILE,
        skiprows=skip_list,
        nrows=chunk_size,
        usecols=wanted_columns,
        engine='python'
    )

    # If there are no rows returned, we're done
    if df_chunk.empty:
        break

    # DATA CLEANING
    # Clean all row contain NA value
    df_cleaned = df_chunk.dropna(axis=0, how="any")
    # Extract year
    df_cleaned['calendar_last_scraped'] = pd.to_datetime(
        df_cleaned['calendar_last_scraped'],
        errors='coerce'
    )
    df_cleaned['year'] = df_cleaned['calendar_last_scraped'].dt.year
    # Change the col from t/f to 1/0
    df_cleaned['host_is_superhost'] = df_cleaned['host_is_superhost'].replace({'t': 1, 'f': 0})
    df_cleaned['instant_bookable'] = df_cleaned['instant_bookable'].replace({'t': 1, 'f': 0})


    all_chunks.append(df_cleaned)

    # Advance the skip pointer by the chunk size
    current_skip += chunk_size
```

```
<ipython-input-3-88ac68c5a396>:53: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in
  df_cleaned['host_is_superhost'] = df_cleaned['host_is_superhost'].replace({'t': 1, 'f': 0})
<ipython-input-3-88ac68c5a396>:54: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in
  df_cleaned['instant_bookable'] = df_cleaned['instant_bookable'].replace({'t': 1, 'f': 0})
<ipython-input-3-88ac68c5a396>:53: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in
  df_cleaned['host_is_superhost'] = df_cleaned['host_is_superhost'].replace({'t': 1, 'f': 0})
<ipython-input-3-88ac68c5a396>:54: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in
  df_cleaned['instant_bookable'] = df_cleaned['instant_bookable'].replace({'t': 1, 'f': 0})
```

```
df_merged = pd.concat([all_chunks[0], all_chunks[1]], axis=0, ignore_index=True)
```

```
# Furthur Data Cleaning
# Change host response rate
df_merged["host_response_rate"] = (
  df_merged["host_response_rate"]
  .astype(str)  # convert to string
  .str.replace("%", "", regex=False)  # remove the % sign
  .astype(float)
  .div(100)                           # convert 67 → 0.67
)

# Change cleaning fee
df_merged["cleaning_fee"] = (
    df_merged["cleaning_fee"]
    .fillna("")                       # fill missing with empty string
    .astype(str)                      # make sure everything is a string
    .str.replace("$", "", regex=False)# remove '$'
    .str.replace(",", "", regex=False)# remove commas
    .astype(float)                    # finally convert to float
)

# Compute compsite score
df_merged["composite_rating"] = df_merged[
  [
    "review_scores_accuracy",
    "review_scores_cleanliness",
    "review_scores_checkin",
    "review_scores_communication",
    "review_scores_location",
    "review_scores_value"
  ]
].mean(axis=1)
```

```
sia = SentimentIntensityAnalyzer()

# Fill and convert description to string
df_merged["description"] = df_merged["description"].fillna("").astype(str)

# Ensure the DataFrame has a simple integer index 0..N-1
df_merged = df_merged.reset_index(drop=True)

# Create the new column ahead of time (optional but recommended)
df_merged["sentiment_compound"] = None

# Loop over rows
for i in range(len(df_merged)):
    text = df_merged.loc[i, "description"]
    score = sia.polarity_scores(text)["compound"]
    df_merged.loc[i, "sentiment_compound"] = score
```

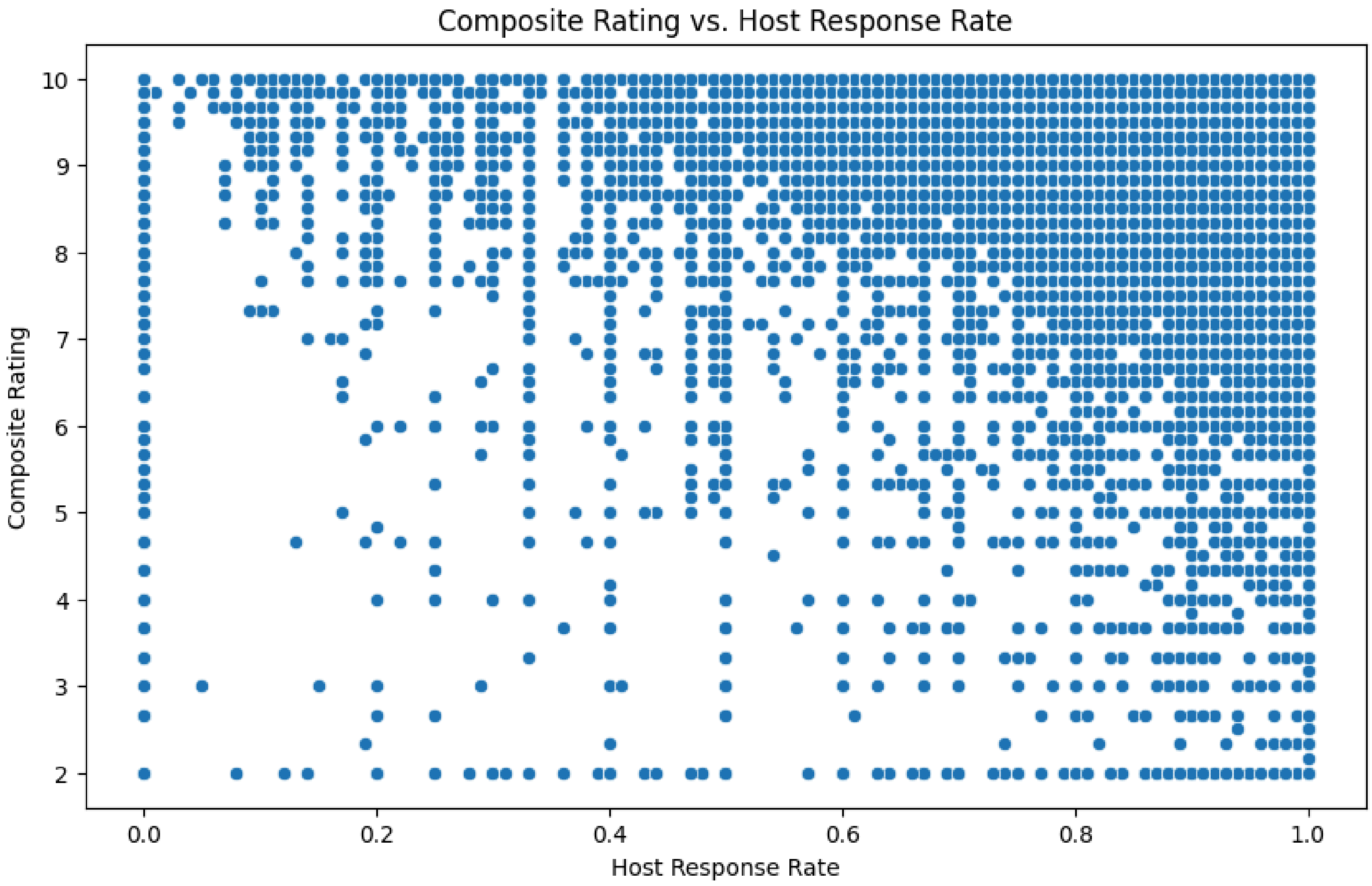
```
df_merged.to_csv('data.csv', index=True)
```

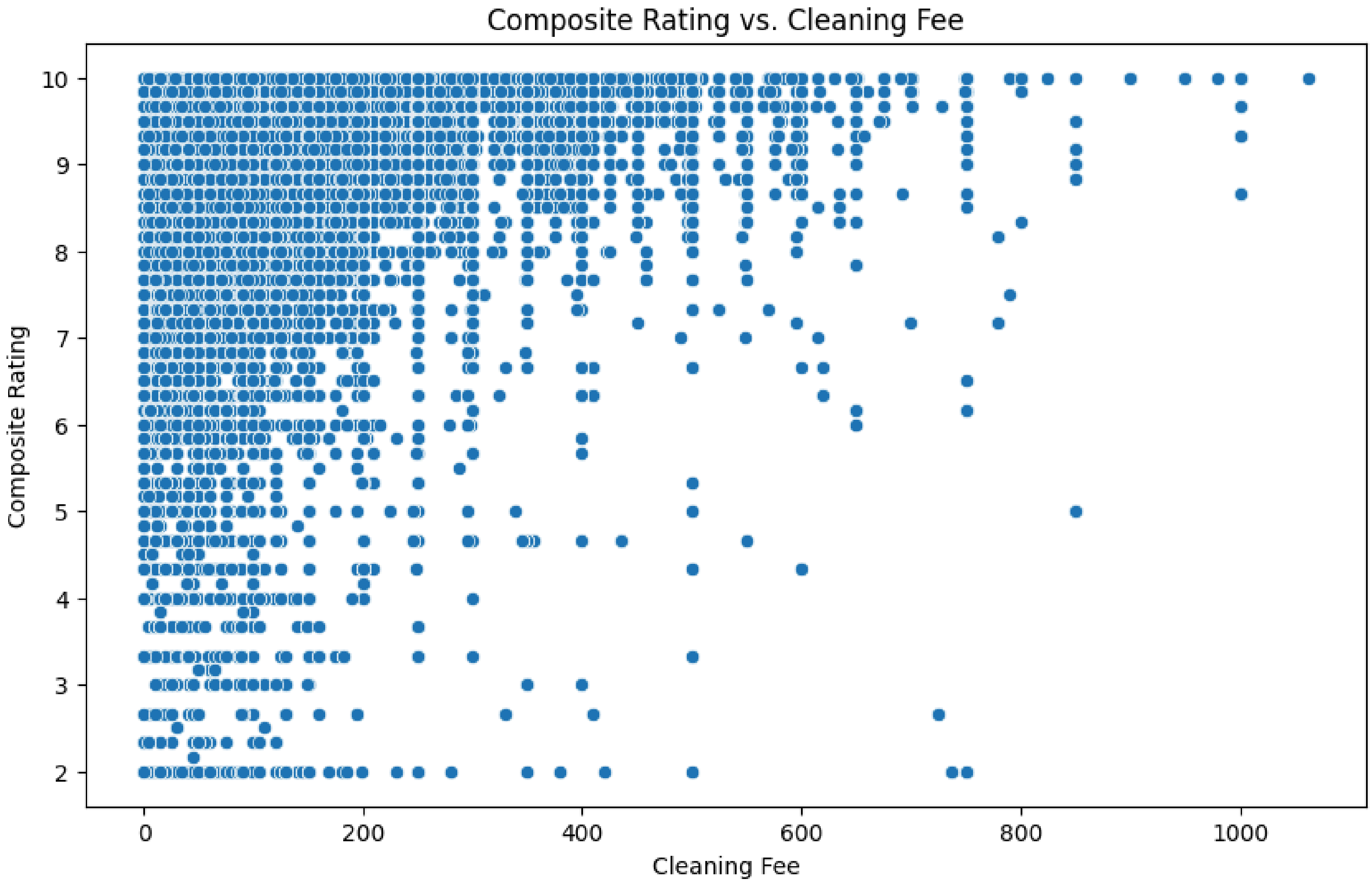
```
df = pd.read_csv('data.csv')
df.head()
```

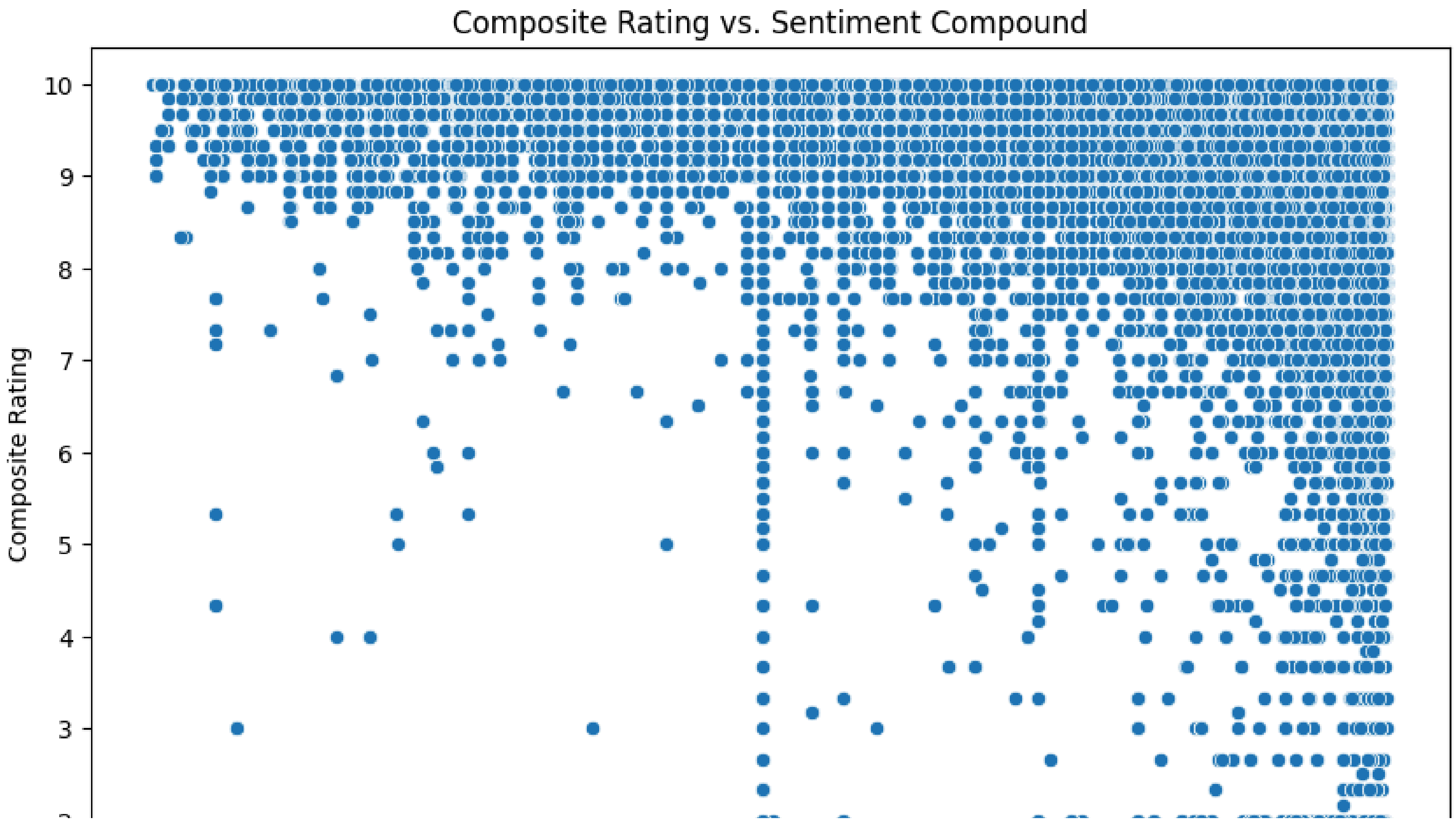
	Unnamed: 0	description	host_response_rate	host_is_superhost	cleaning_fee	calendar_last_scraped	review_scores_accurac
0	0	This home is perfect for families; aspiring ch...	0.67	0	100.0	2019-01-12	10
1	1	Our best memory foam pillows you'll ever sleep...	1.00	1	85.0	2019-01-11	10
2	2	This is a three story townhouse with the follo...	1.00	0	100.0	2019-01-12	9
3	3	A very Modern Hollywood Hills Zen style galler...	0.90	0	60.0	2019-01-12	8
4	4	Our distinctive bachelor's studio for 1-2 gues...	1.00	0	25.0	2019-01-11	8

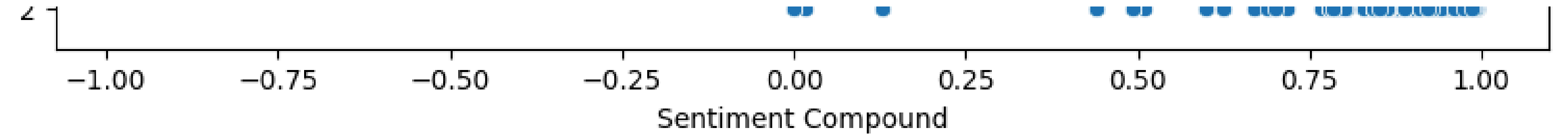
EDA

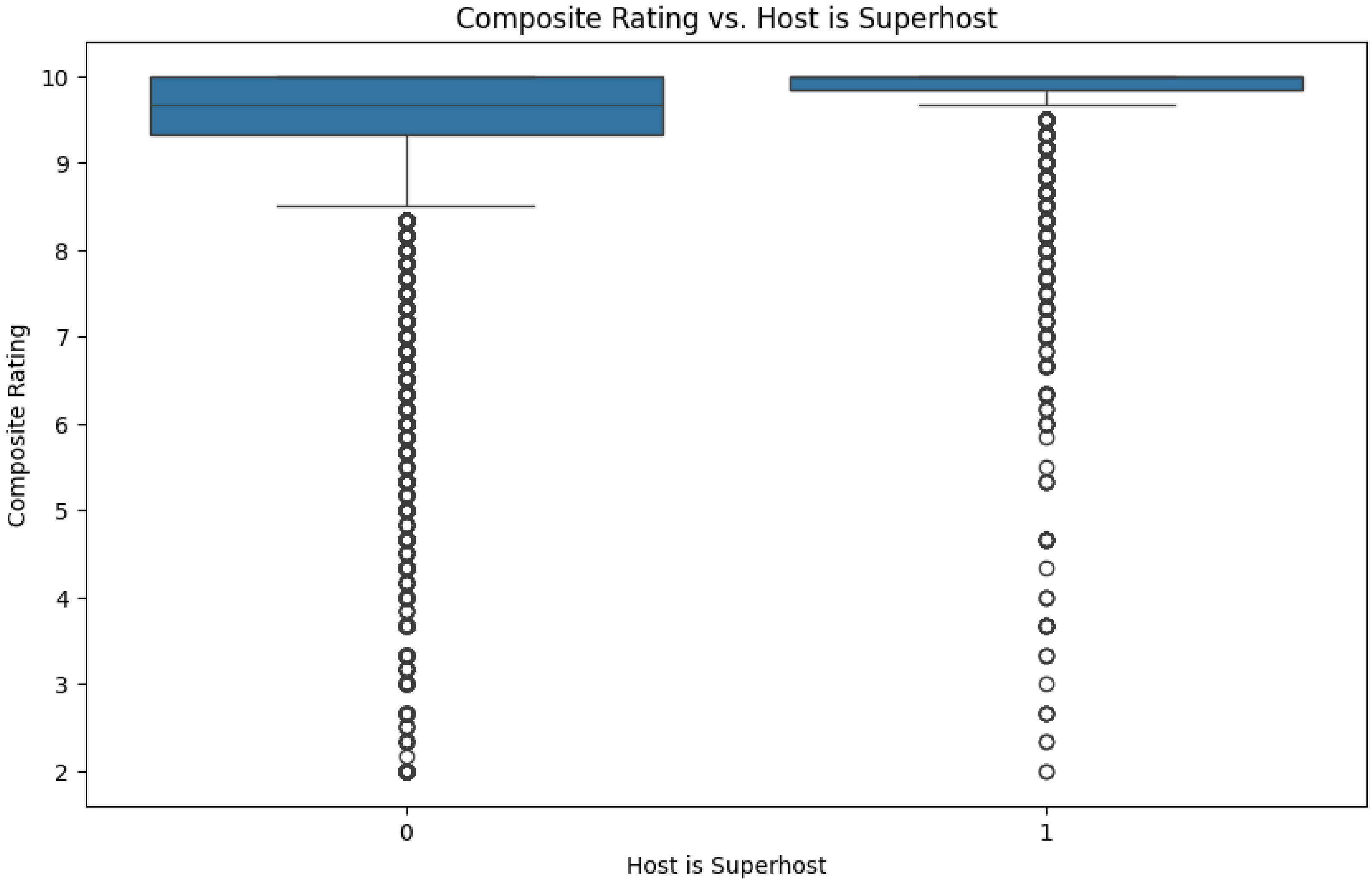
```
# prompt: plot composite_rating vs host_response_rate

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.scatterplot(x='host_response_rate', y='composite_rating', data=df)
plt.title('Composite Rating vs. Host Response Rate')
plt.xlabel('Host Response Rate')
plt.ylabel('Composite Rating')
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(x='cleaning_fee', y='composite_rating', data=df)
plt.title('Composite Rating vs. Cleaning Fee')
plt.xlabel('Cleaning Fee')
plt.ylabel('Composite Rating')
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(x='sentiment_compound', y='composite_rating', data=df)
plt.title('Composite Rating vs. Sentiment Compound')
plt.xlabel('Sentiment Compound')
plt.ylabel('Composite Rating')
plt.show()

plt.figure(figsize=(10, 6))
sns.boxplot(x='host_is_superhost', y='composite_rating', data=df)
plt.title('Composite Rating vs. Host is Superhost')
plt.xlabel('Host is Superhost')
plt.ylabel('Composite Rating')
plt.show()

plt.figure(figsize=(10, 6))
sns.boxplot(x='instant_bookable', y='composite_rating', data=df)
plt.title('Composite Rating vs. Instant Bookable')
plt.xlabel('Instant Bookable')
plt.ylabel('Composite Rating')
plt.show()
```

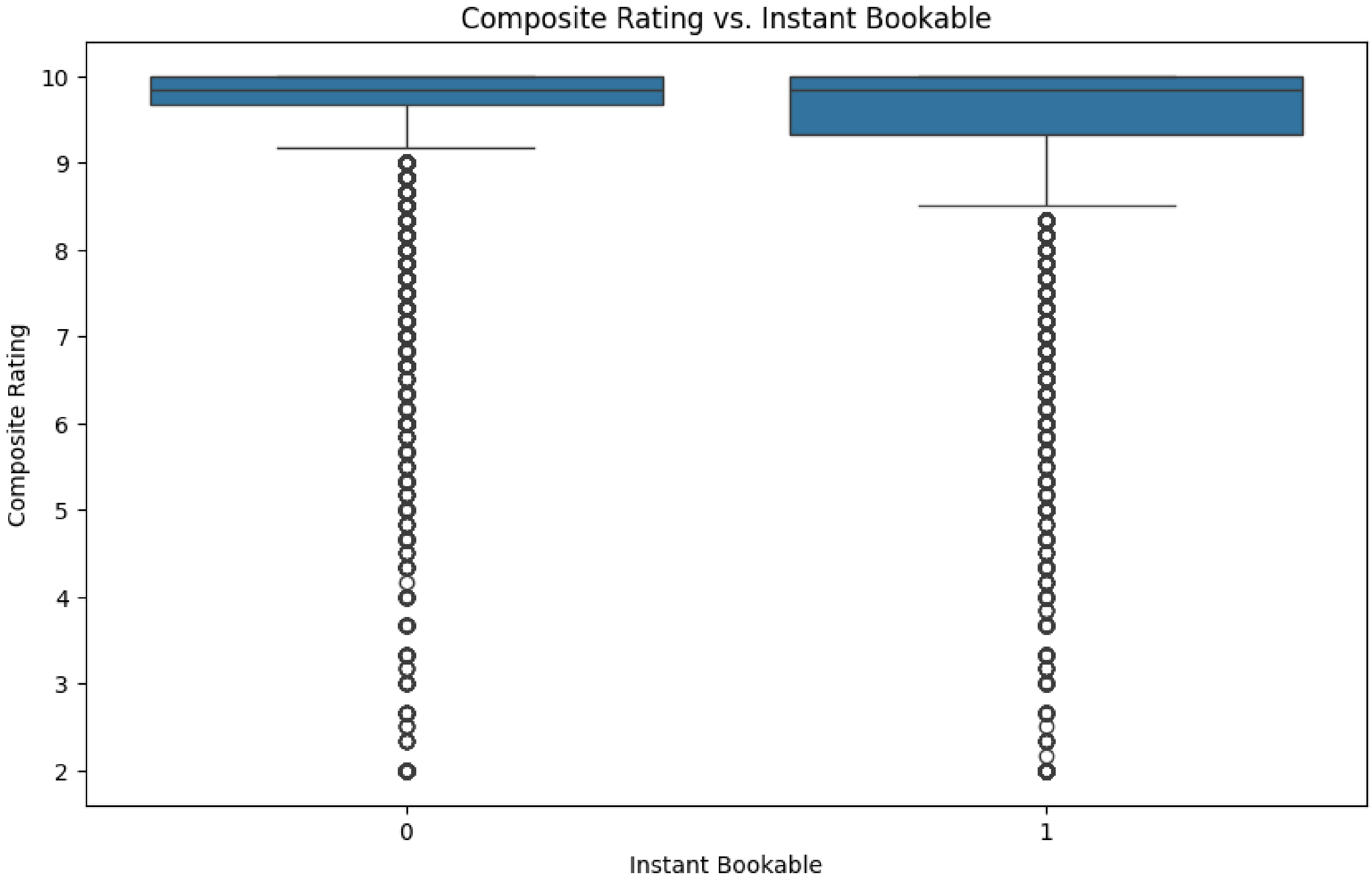

Composite Rating vs. Host Response Rate
Composite Rating
10
9
8
7
6
5
4
3
2
0.0
0.2
0.4
0.6
0.8
1.0
Host Response Rate


Composite Rating vs. Cleaning Fee
Composite Rating
10
9
8
7
6
5
4
3
2
0
200
400
600
800
1000
Cleaning Fee


Composite Rating vs. Sentiment Compound
Composite Rating
10
9
8
7
6
5
4
3


2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Sentiment Compound


Composite Rating vs. Host is Superhost
10
9
8
7
6
5
4
3
2
Composite Rating
0
1
Host is Superhost


Composite Rating vs. Instant Bookable
10
9
8
7
6
5
4
3
2
Composite Rating
0
1
Instant Bookable

MLR Model

```
import statsmodels.api as sm

# Fit the multiple linear regression model
X = df[['host_response_rate', 'cleaning_fee', 'sentiment_compound', 'host_is_superhost', 'instant_bookable']]
X = sm.add_constant(X)  # Add intercept
y = df['composite_rating']

mlr_model = sm.OLS(y, X).fit()

# Print the full summary (which includes the F-test result)
print(mlr_model.summary())

# Extract F-statistic and p-value
f_statistic = mlr_model.fvalue
p_value = mlr_model.f_pvalue

print(f"F-statistic: {f_statistic}")
print(f"P-value: {p_value}")

# Interpretation
if p_value < 0.05:
    print("The model is statistically significant (reject H0). At least one predictor explains variance in the dependent vari
else:
    print("The model is NOT statistically significant (fail to reject H0). The predictors collectively do not explain the out
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       composite_rating   R-squared:                       0.120
Model:                            OLS   Adj. R-squared:                  0.120
Method:                 Least Squares   F-statistic:                 1.302e+04
Date:                Sun, 02 Mar 2025   Prob (F-statistic):               0.00
Time:                        06:56:58   Log-Likelihood:            -4.0869e+05
No. Observations:              475581   AIC:                         8.174e+05
Df Residuals:                  475575   BIC:                         8.175e+05
Df Model:                           5
Covariance Type:            nonrobust
======================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                  9.1823      0.007   1336.962      0.000       9.169       9.196
host_response_rate     0.3013      0.007     45.967      0.000       0.288       0.314
cleaning_fee       -8.231e-05   9.88e-06     -8.335      0.000      -0.000    -6.3e-05
sentiment_compound     0.0838      0.003     25.622      0.000       0.077       0.090
host_is_superhost      0.3762      0.002    219.172      0.000       0.373       0.380
instant_bookable      -0.1240      0.002    -73.868      0.000      -0.127      -0.121
==============================================================================
Omnibus:                   479999.529   Durbin-Watson:                   1.939
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         39180428.632
Skew:                          -4.917   Prob(JB):                         0.00
Kurtosis:                      46.365   Cond. No.                     1.35e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.35e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
F-statistic: 13020.657353726368
P-value: 0.0
The model is statistically significant (reject H0). At least one predictor explains variance in the dependent variable.
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
import pandas as pd
```