Solution

(2) Case Study: Source and investigate usable data sources

(GenAI Life Cycle Phase 2: Data Understanding self-practice)

Note on EDA

When performing Exploratory Data Analysis (EDA), the specific techniques used are less important than the overall goal: gaining a solid understanding of the data before transforming or modeling it. The primary objective of EDA is to uncover insights about the structure, quality, and relationships within the dataset to inform subsequent steps effectively.

- Load the head of each file to view the first few entries of each dataset.

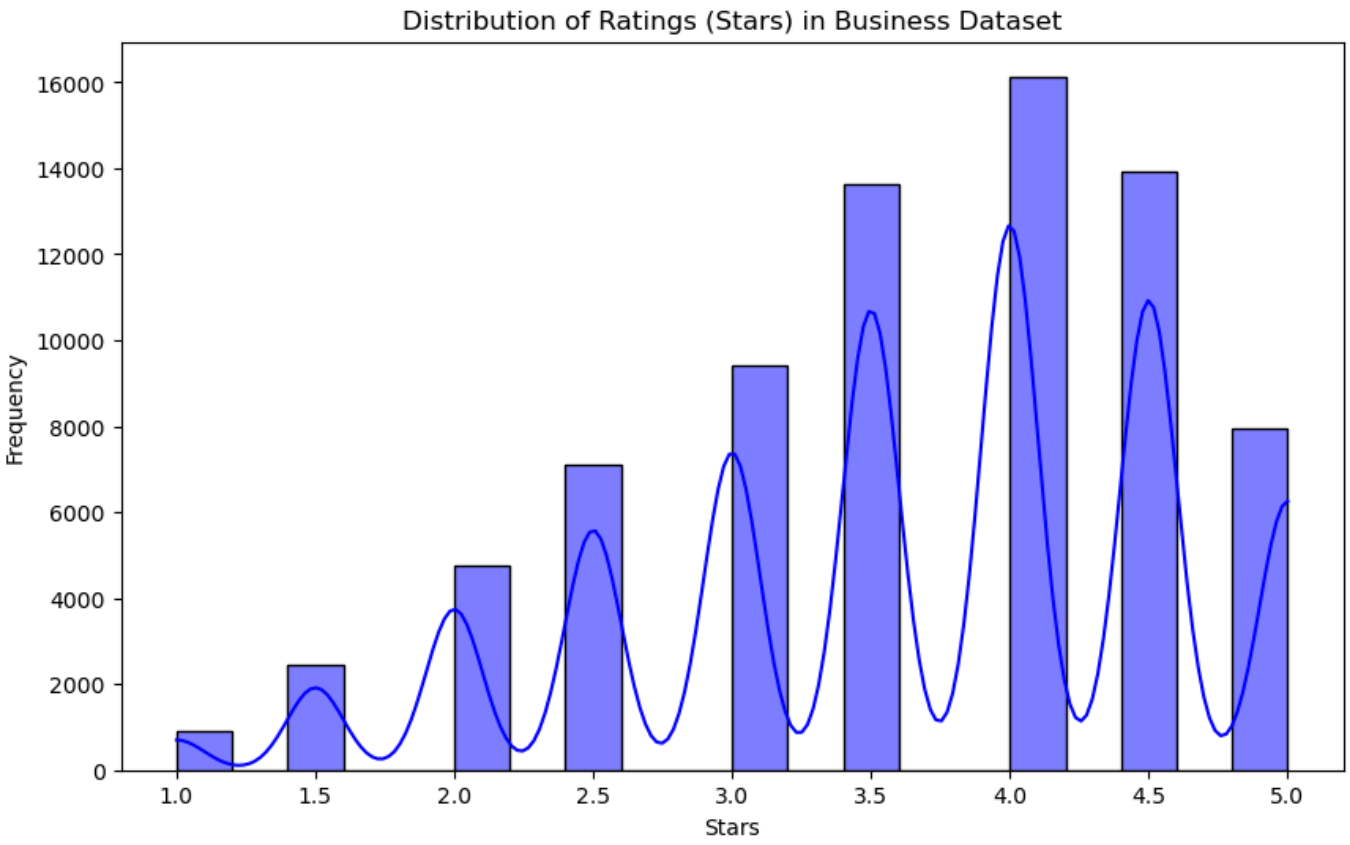
In [9]:
```
import pandas as pd

# Load datasets
df_business = pd.read_csv('yelp_academic_dataset_business.csv')
df_review = pd.read_csv('yelp_academic_dataset_review.csv')
df_user = pd.read_csv('yelp_academic_dataset_user.csv')

# Perform EDA
print("Business Dataset Head:")
print(df_business.info())
print("\nReview Dataset Head:")
print(df_review.info())
print("\nUser Dataset Head:")
print(df_user.info())
```

```
Business Dataset Head:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76214 entries, 0 to 76213
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   business_id   76214 non-null  object
 1   name          76214 non-null  object
 2   address       73808 non-null  object
 3   city          76214 non-null  object
 4   state         76214 non-null  object
 5   postal_code   76183 non-null  object
 6   latitude      76214 non-null  float64
 7   longitude     76214 non-null  float64
 8   stars         76214 non-null  float64
 9   review_count  76214 non-null  int64
 10  is_open       76214 non-null  int64
 11  attributes    69677 non-null  object
 12  categories    76170 non-null  object
 13  hours         64671 non-null  object
dtypes: float64(3), int64(2), object(9)
memory usage: 8.1+ MB
None

Review Dataset Head:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2360015 entries, 0 to 2360014
Data columns (total 9 columns):
 #   Column       Dtype
---  ------       -----
 0   review_id    object
 1   user_id      object
 2   business_id  object
 3   stars        float64
 4   useful       float64
 5   funny        float64
 6   cool         float64
 7   text         object
 8   date         object
dtypes: float64(4), object(5)
memory usage: 162.0+ MB
None

User Dataset Head:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 496974 entries, 0 to 496973
Data columns (total 22 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   user_id             496974 non-null  object
 1   name                496966 non-null  object
 2   review_count        496974 non-null  int64
 3   yelping_since       496974 non-null  object
 4   useful              496974 non-null  int64
 5   funny               496974 non-null  int64
 6   cool                496974 non-null  int64
 7   elite               49631 non-null   object
 8   friends             355801 non-null  object
 9   fans                496974 non-null  int64
 10  average_stars       496974 non-null  float64
 11  compliment_hot      496974 non-null  int64
 12  compliment_more     496974 non-null  int64
 13  compliment_profile  496974 non-null  int64
 14  compliment_cute     496974 non-null  int64
 15  compliment_list     496974 non-null  int64
 16  compliment_note     496974 non-null  int64
```

```
 17  compliment_plain    496974 non-null  int64
 18  compliment_cool     496974 non-null  int64
 19  compliment_funny    496974 non-null  int64
 20  compliment_writer   496974 non-null  int64
 21  compliment_photos   496974 non-null  int64
dtypes: float64(1), int64(16), object(5)
memory usage: 83.4+ MB
None
```

1. First file: `yelp_academic_dataset_business.csv`

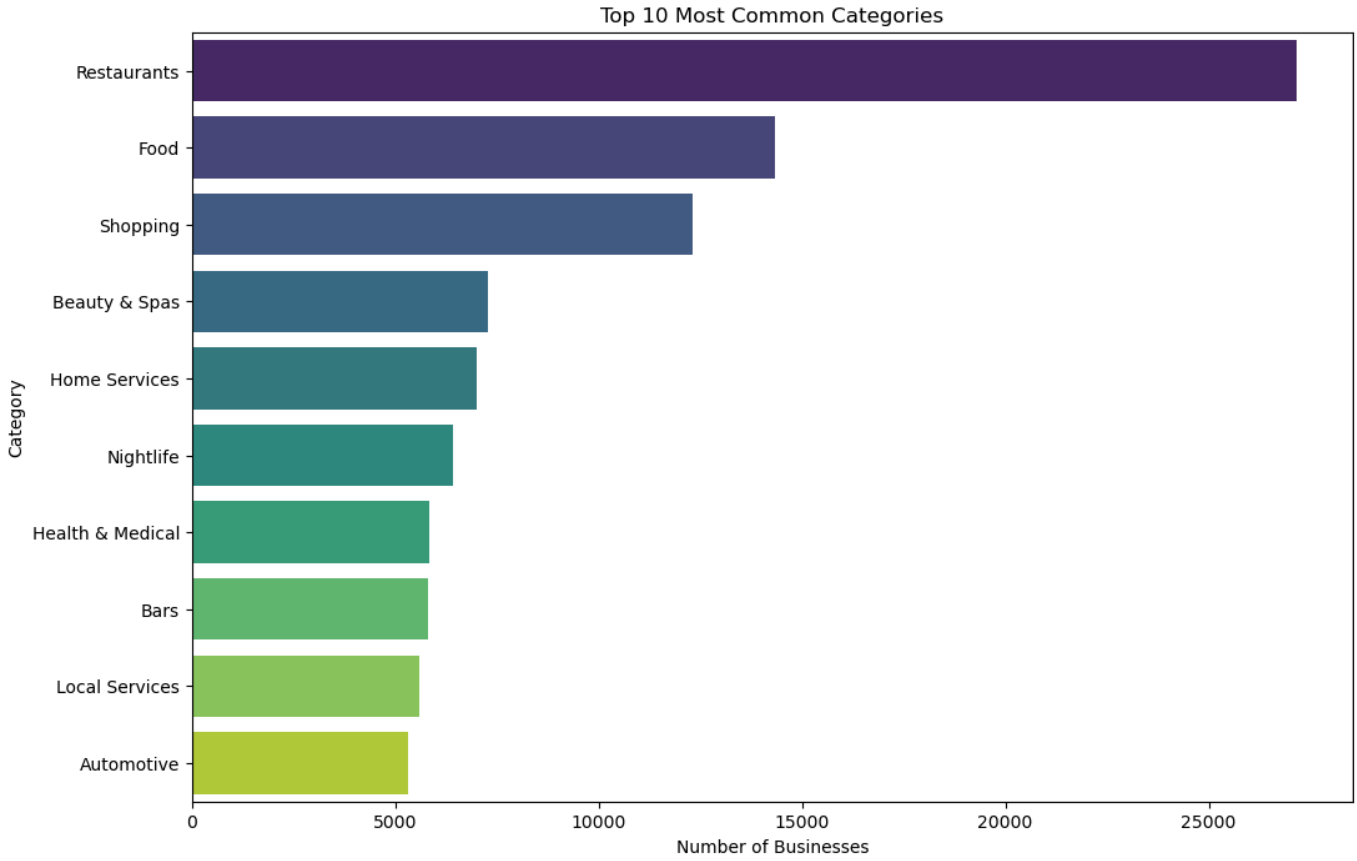
In [10]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load business dataset
df_business = pd.read_csv('yelp_academic_dataset_business.csv')

# Distribution of Ratings (Stars)
plt.figure(figsize=(10, 6))
sns.histplot(df_business['stars'], bins=20, kde=True, color='blue')
plt.title('Distribution of Ratings (Stars) in Business Dataset')
plt.xlabel('Stars')
plt.ylabel('Frequency')
plt.show()
```



In [11]:
```python
# Extract top categories from the categories column
df_business['categories'] = df_business['categories'].fillna('')
categories = df_business['categories'].str.split(',').explode().str.strip()

# Count the occurrences of each category
category_counts = categories.value_counts().head(10)

# Plot the top 10 categories
plt.figure(figsize=(12, 8))
sns.barplot(x=category_counts.values, y=category_counts.index, palette='viridis')
plt.title('Top 10 Most Common Categories')
plt.xlabel('Number of Businesses')
```

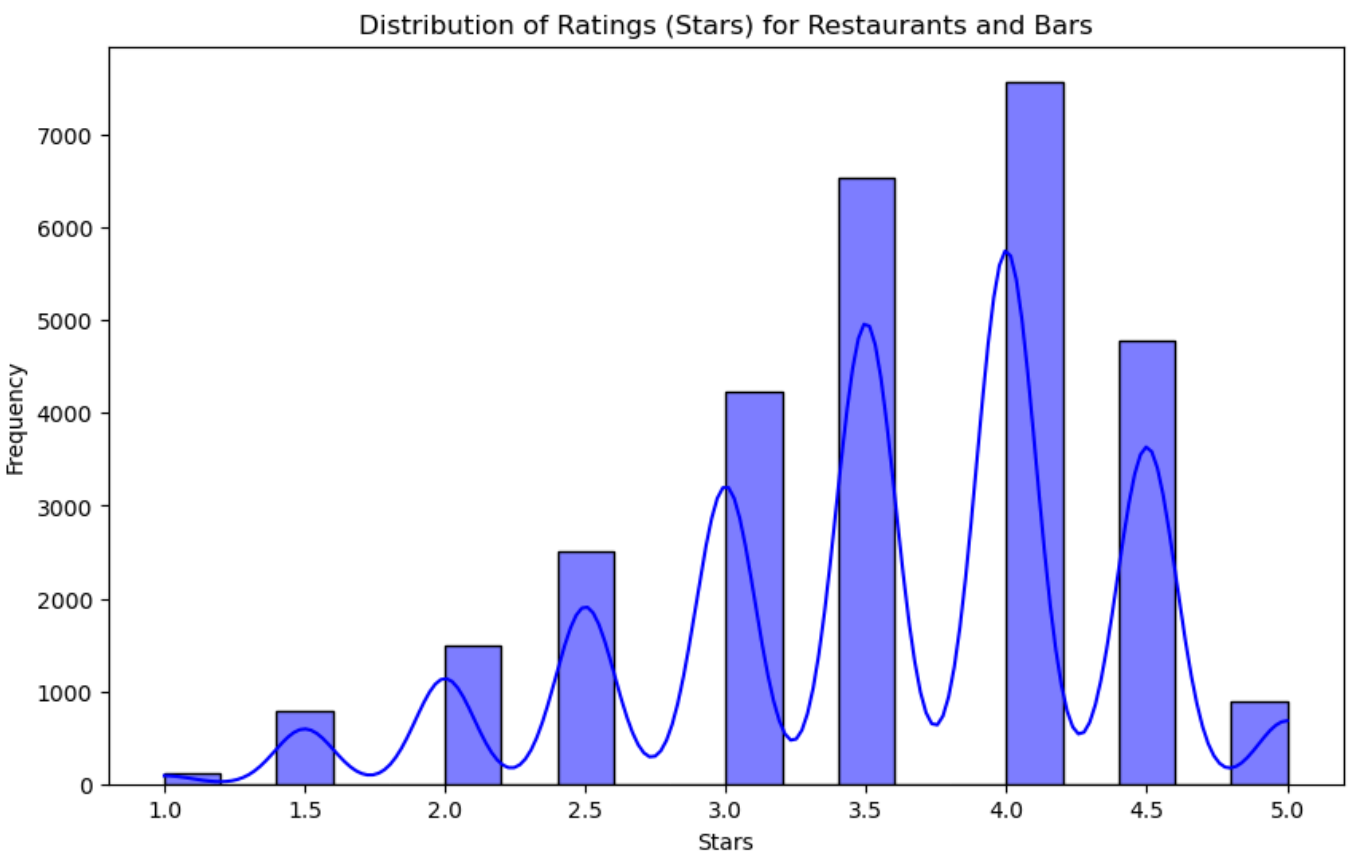
```
plt.ylabel('Category')
plt.show()
```

```
/var/folders/hj/877lyhb1715fltx1jm9dkwxw0000gn/T/ipykernel_45659/2859398872.py:10: Fut
ureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.
0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=category_counts.values, y=category_counts.index, palette='viridis')
```



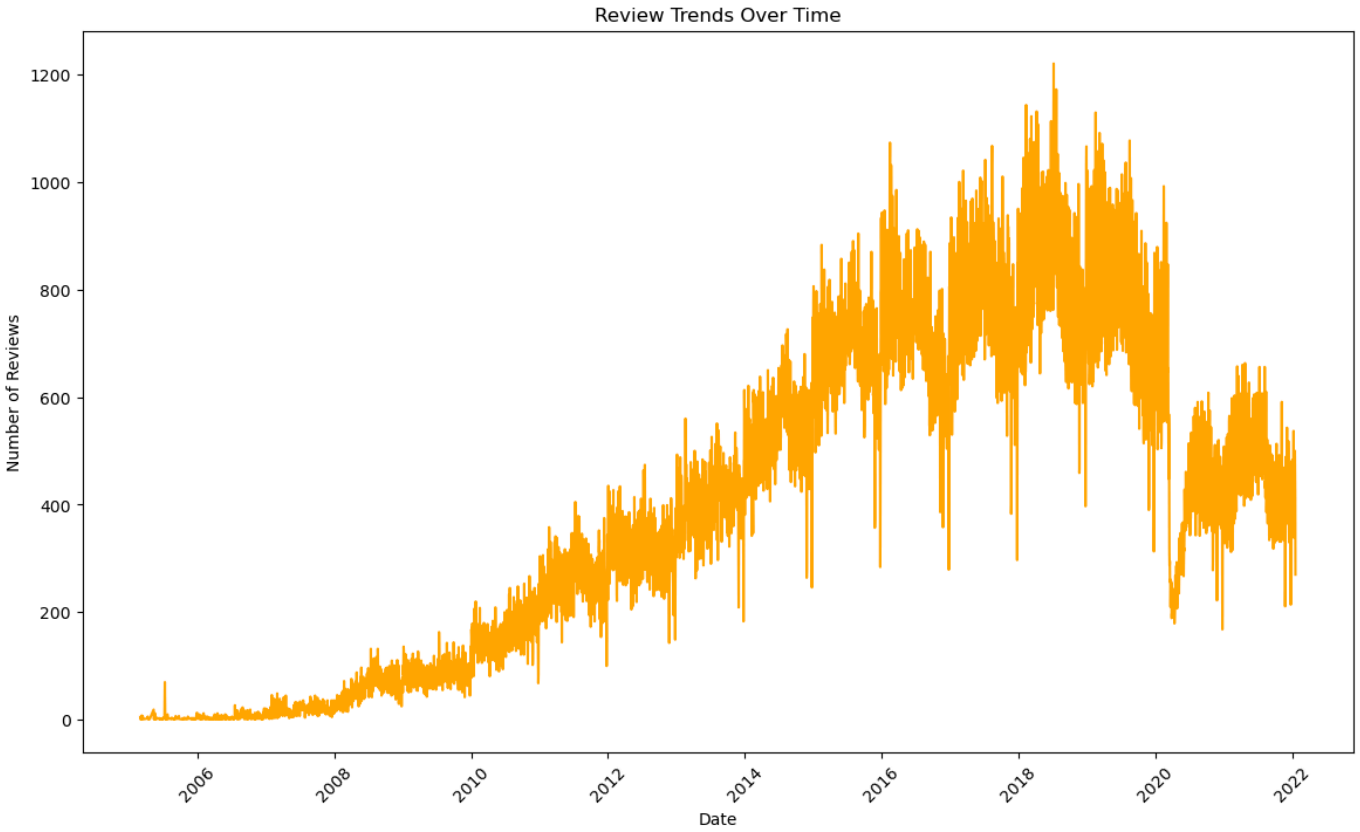
- The scope of this project encompasses restaurants and bars so let's revisit 'Distribution of Ratings (Stars) in Business Dataset' focused on those two categories

In [12]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load business dataset
df_business = pd.read_csv('yelp_academic_dataset_business.csv')

# Filter for Restaurants and Bars
df_filtered = df_business[df_business['categories'].str.contains('Restaurants|Bars',

# Distribution of Ratings (Stars) for Restaurants and Bars
plt.figure(figsize=(10, 6))
sns.histplot(df_filtered['stars'], bins=20, kde=True, color='blue')
plt.title('Distribution of Ratings (Stars) for Restaurants and Bars')
plt.xlabel('Stars')
plt.ylabel('Frequency')
plt.show()
```



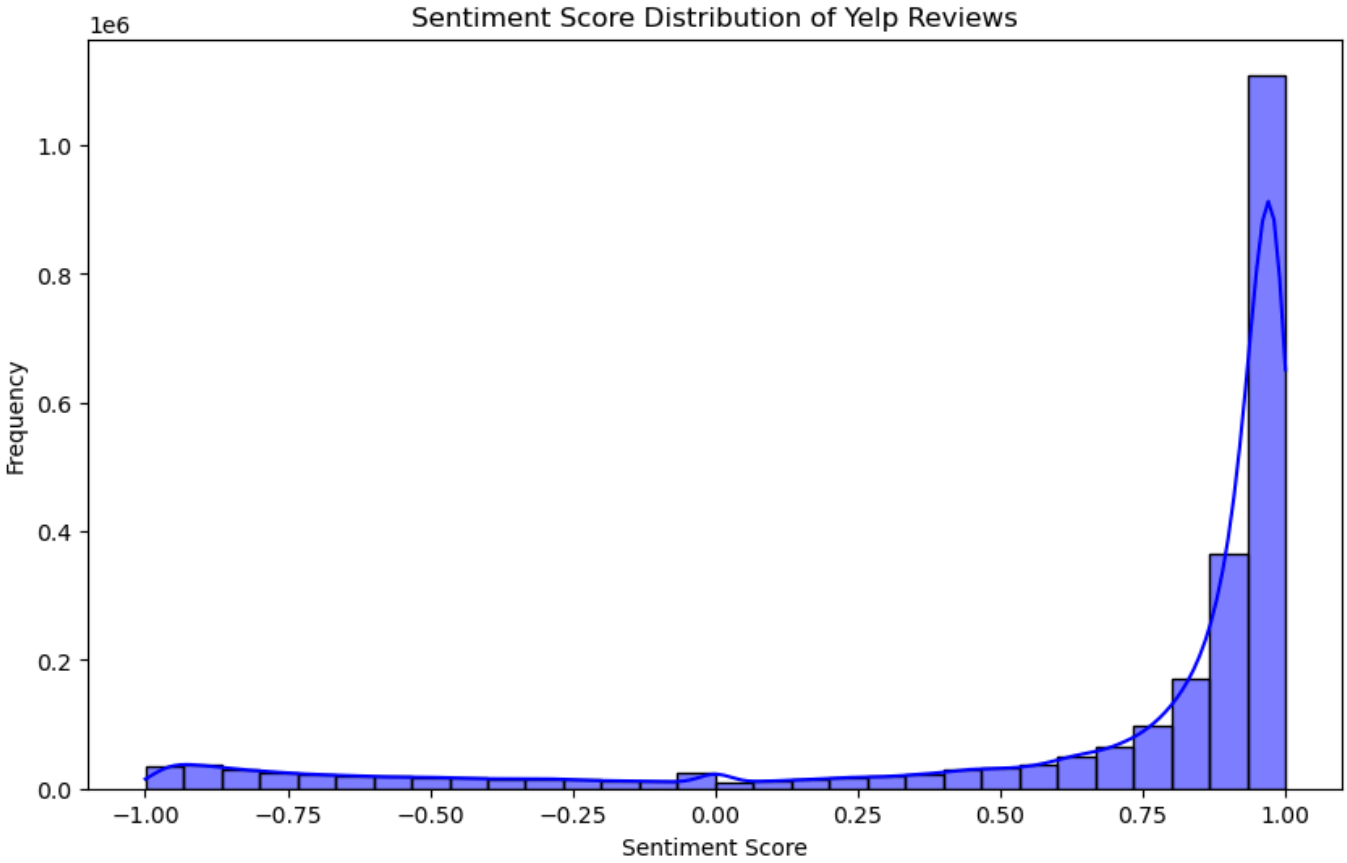
2. Second file: `yelp_academic_dataset_review.csvv`

In [13]:
```python
# Load review dataset
df_reviews = pd.read_csv('yelp_academic_dataset_review.csv')

# Convert date to datetime format
df_reviews['date'] = pd.to_datetime(df_reviews['date'])

# Group by date and count reviews
reviews_by_date = df_reviews.groupby(df_reviews['date'].dt.date).size()

# Line plot of reviews over time
plt.figure(figsize=(14, 8))
reviews_by_date.plot(kind='line', color='orange')
plt.title('Review Trends Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Reviews')
plt.xticks(rotation=45)
plt.show()
```



- We can look at the sentiment scores of the reviews

In [14]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer

# Download VADER Lexicon
nltk.download('vader_lexicon', quiet=True)

# Initialize Sentiment Analyzer
sid = SentimentIntensityAnalyzer()

# Load Reviews Dataset
df_reviews = pd.read_csv('yelp_academic_dataset_review.csv')

# Sentiment Analysis on Review Text
df_reviews['sentiment_score'] = df_reviews['text'].apply(lambda x: sid.polarity_score

# Plot Sentiment Score Distribution
plt.figure(figsize=(10, 6))
sns.histplot(df_reviews['sentiment_score'], bins=30, kde=True, color='blue')
plt.title("Sentiment Score Distribution of Yelp Reviews")
plt.xlabel("Sentiment Score")
plt.ylabel("Frequency")
plt.show()
```

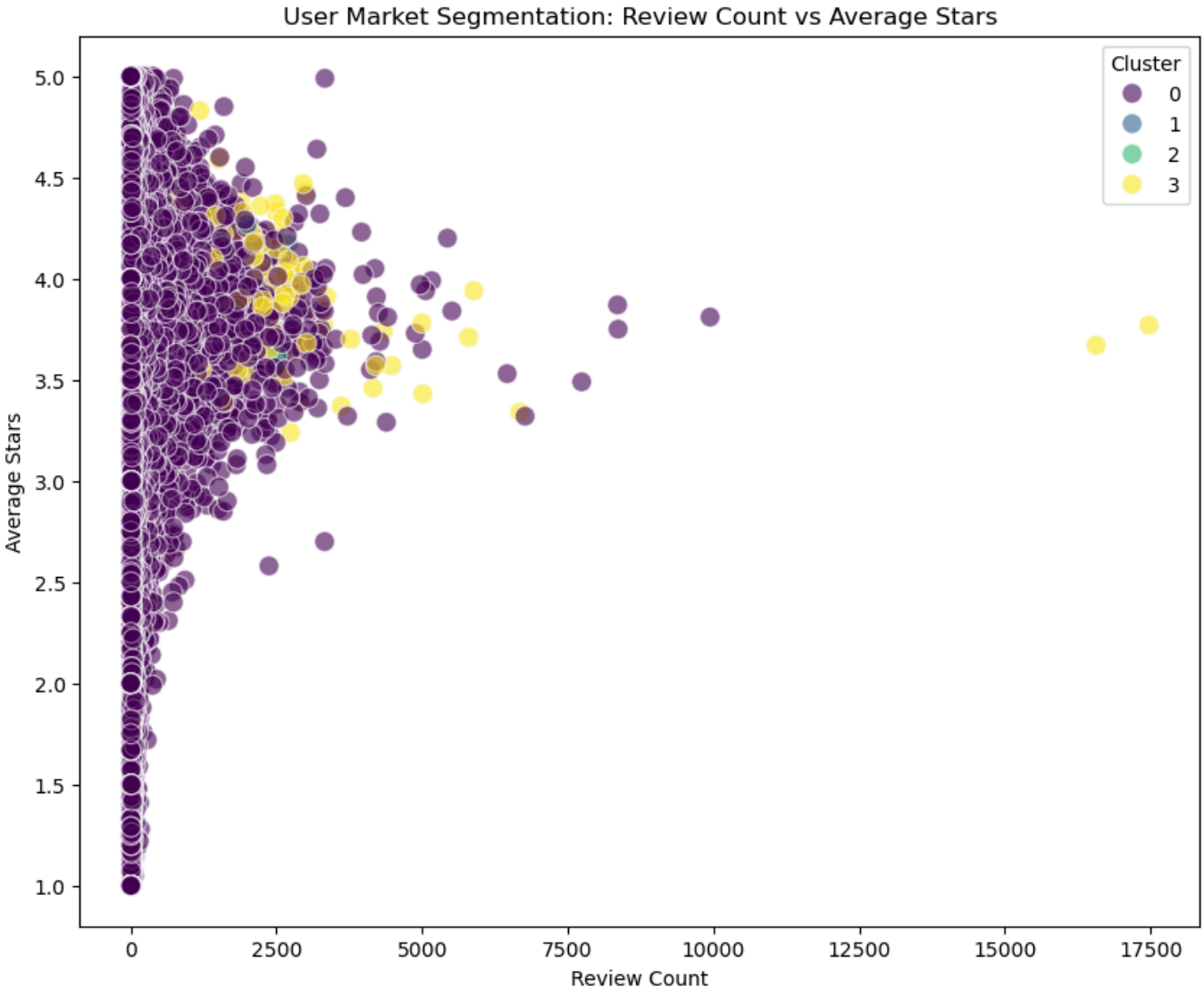


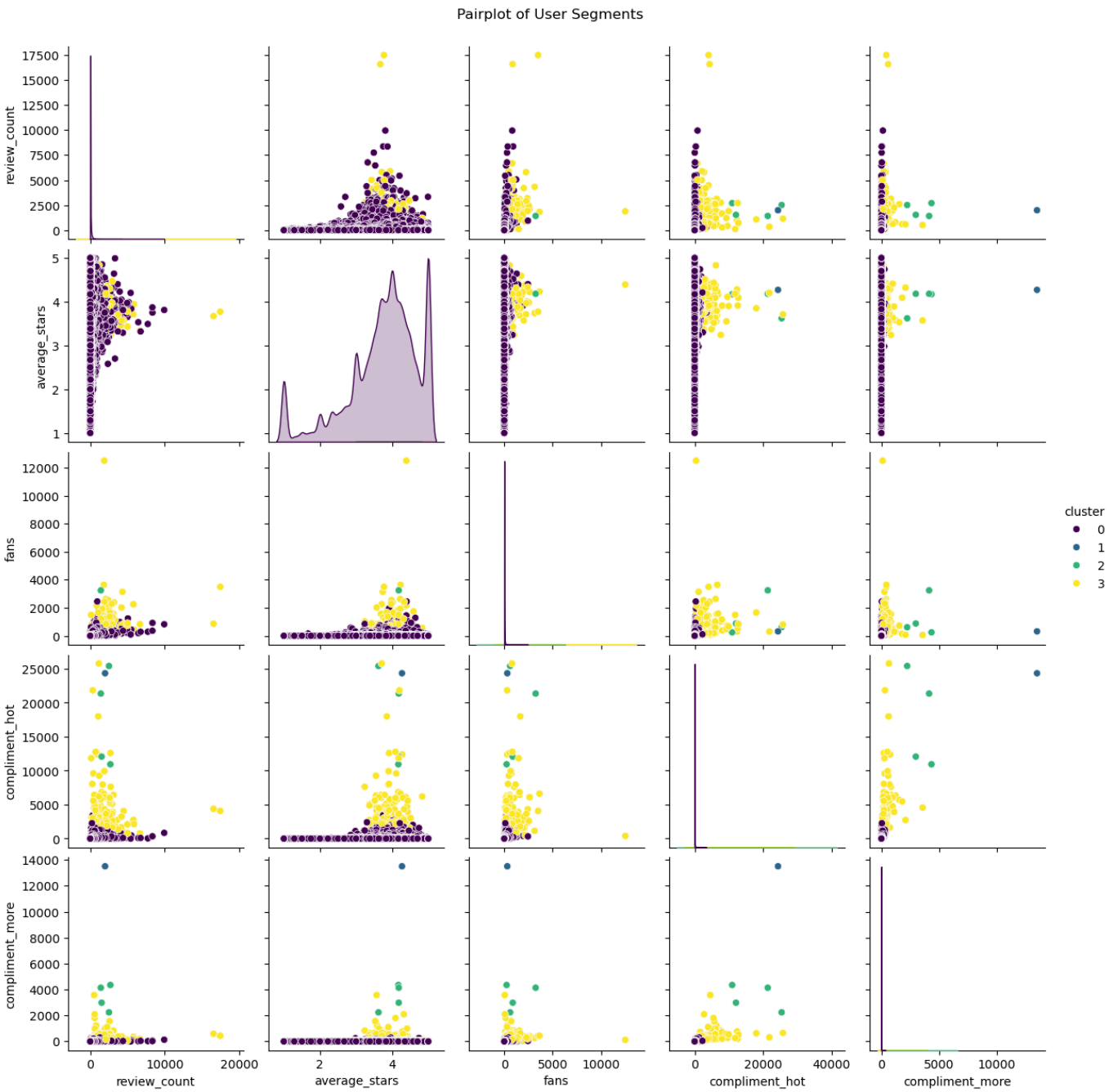
3. Third file: `yelp_academic_dataset_review.csvv`

We hadn't dealt with users in the Practice Learning Activity of Chapter 2. One useful way to get an overview of users is to segment into groups them by their different information.

In [15]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# Load user dataset
df_users = pd.read_csv('yelp_academic_dataset_user.csv')

# Feature selection: Choose relevant features for market segmentation
features = ['review_count', 'average_stars', 'fans',
            'compliment_hot', 'compliment_more', 'compliment_profile',
            'compliment_cute', 'compliment_list', 'compliment_note',
            'compliment_plain', 'compliment_cool', 'compliment_funny',
            'compliment_writer', 'compliment_photos']

# Fill missing values (if any)
df_users[features] = df_users[features].fillna(0)

# Normalize/Scale the data
scaler = StandardScaler()
df_users_scaled = scaler.fit_transform(df_users[features])

# Perform K-means clustering
kmeans = KMeans(n_clusters=4, random_state=42)  # Change the number of clusters based
df_users['cluster'] = kmeans.fit_predict(df_users_scaled)

# Visualize the segmentation
plt.figure(figsize=(10, 8))
sns.scatterplot(x=df_users['review_count'], y=df_users['average_stars'], hue=df_users
plt.title('User Market Segmentation: Review Count vs Average Stars')
```

```
plt.xlabel('Review Count')
plt.ylabel('Average Stars')
plt.legend(title='Cluster')
plt.show()

# Pairplot to visualize clusters
sns.pairplot(df_users[['review_count', 'average_stars', 'fans', 'compliment_hot', 'co
plt.suptitle("Pairplot of User Segments", y=1.02)
plt.show()

# Display cluster centers
centers = pd.DataFrame(kmeans.cluster_centers_, columns=features)
print("Cluster Centers (User Segments):")
print(centers)

# Segment Analysis
for i in range(4):  # Change the number of clusters here if necessary
    print(f"\nCluster {i} Summary:")
    segment = df_users[df_users['cluster'] == i]
    print(segment[features].describe())
```




Pairplot of User Segments
review_count
average_stars
fans
compliment_hot
compliment_more
cluster
0
1
2
3

Cluster Centers (User Segments):

	review_count	average_stars	fans	compliment_hot	compliment_more	compliment_profile	compliment_cute	compliment_list	compliment_note	compliment_plain	compliment_cool	compliment_funny	compliment_writer	compliment_photos
0	-0.005121	-0.000088	-0.009681	-0.012950	-0.007478	-0.006956	-0.005612	-0.004320	-0.008613	-0.011420	-0.013469	-0.013469	-0.011905	-0.008252
1	14.160863	0.557050	9.539340	204.724998	563.805120	498.999517	627.388900	644.257478	157.884846	115.505556	187.139740	187.139740	280.733949	476.403812
2	14.513021	0.313668	37.532019	146.674197	143.122070	183.296452	96.354081	95.507036	79.785281	171.754471	174.135525	174.135525	177.690291	141.865106
3	13.077898	0.222626	24.600368	29.851632	13.642348	11.764079	9.391955	5.925746	20.116056	25.771430	30.726084	30.726084	26.045859	16.166851

Cluster 0 Summary:

	review_count	average_stars	fans	compliment_hot	compliment_more	compliment_profile	compliment_cute	compliment_list	compliment_note	compliment_plain	compliment_cool	compliment_funny	compliment_writer	compliment_photos
count	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000	496780.000000
mean	45.030951	3.737772	3.032383	2.860033	0.514900	0.260971	0.203176	0.093569	2.618086	5.227741	4.831126	4.831126	1.984277	1.651437
std	127.563026	0.955450	19.504478	36.439235	3.864106	3.395599	3.694333	1.740106	23.374778	71.127031	52.751364	52.751364	17.798570	28.531025
min	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.000000	3.290000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	12.000000	3.900000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	33.000000	4.400000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000
max	9941.000000	5.000000	2451.000000	3448.000000	369.000000	502.000000	609.000000	321.000000	2265.000000	6510.000000	3502.000000	3502.000000	1351.000000	4141.000000

Cluster 1 Summary:

	review_count	average_stars	fans	compliment_hot	compliment_more
count	1.0	1.00	1.0	1.0	1.0
mean	1996.0	4.27	319.0	24348.0	13501.0
std	NaN	NaN	NaN	NaN	NaN
min	1996.0	4.27	319.0	24348.0	13501.0
25%	1996.0	4.27	319.0	24348.0	13501.0
50%	1996.0	4.27	319.0	24348.0	13501.0
75%	1996.0	4.27	319.0	24348.0	13501.0
max	1996.0	4.27	319.0	24348.0	13501.0

	compliment_profile	compliment_cute	compliment_list	compliment_note
count	1.0	1.0	1.0	1.0
mean	14180.0	13654.0	12669.0	15927.0
std	NaN	NaN	NaN	NaN
min	14180.0	13654.0	12669.0	15927.0
25%	14180.0	13654.0	12669.0	15927.0
50%	14180.0	13654.0	12669.0	15927.0
75%	14180.0	13654.0	12669.0	15927.0
max	14180.0	13654.0	12669.0	15927.0

	compliment_plain	compliment_cool	compliment_funny	compliment_writer
count	1.0	1.0	1.0	1.0
mean	24943.0	30008.0	30008.0	15446.0
std	NaN	NaN	NaN	NaN
min	24943.0	30008.0	30008.0	15446.0
25%	24943.0	30008.0	30008.0	15446.0
50%	24943.0	30008.0	30008.0	15446.0
75%	24943.0	30008.0	30008.0	15446.0
max	24943.0	30008.0	30008.0	15446.0

	compliment_photos
count	1.0
mean	82630.0
std	NaN
min	82630.0
25%	82630.0
50%	82630.0
75%	82630.0
max	82630.0

Cluster 2 Summary:

	review_count	average_stars	fans	compliment_hot
count	4.000000	4.000000	4.000000	4.000000
mean	2044.500000	4.037500	1245.250000	17445.250000
std	659.010116	0.278373	1356.857736	7067.555088
min	1424.000000	3.620000	247.000000	10944.000000
25%	1506.500000	4.032500	520.000000	11790.750000
50%	2025.000000	4.175000	745.500000	16713.000000
75%	2563.000000	4.180000	1470.750000	22367.500000
max	2704.000000	4.180000	3243.000000	25411.000000

	compliment_more	compliment_profile	compliment_cute	compliment_list
count	4.000000	4.000000	4.000000	4.000000
mean	3427.750000	5209.000000	2097.250000	1878.250000
std	992.533585	1825.104381	783.733533	743.987175
min	2240.000000	3144.000000	1277.000000	1096.000000
25%	2799.500000	3967.500000	1535.750000	1321.750000
50%	3562.000000	5326.500000	2069.000000	1905.000000
75%	4190.250000	6568.000000	2630.500000	2461.500000
max	4347.000000	7039.000000	2974.000000	2607.000000

	compliment_note	compliment_plain	compliment_cool	compliment_funny
count	4.000000	4.000000	4.000000	4.000000
mean	8050.250000	37086.000000	27923.250000	27923.250000
std	4335.925612	43096.807005	15586.868434	15586.868434

```
min        4751.000000      11231.000000     13280.000000      13280.000000
25%        4835.750000      11624.750000     20927.000000      20927.000000
50%        6740.000000      18008.000000     24223.000000      24223.000000
75%        9954.500000      43469.250000     31219.250000      31219.250000
max       13970.000000     101097.000000     49967.000000      49967.000000

       compliment_writer  compliment_photos
count           4.000000           4.000000
mean         9777.500000       24608.000000
std          4151.460024       23716.764549
min          7260.000000        1284.000000
25%          7296.750000        9950.250000
50%          7958.000000       20522.000000
75%         10438.750000       35179.750000
max         15934.000000       56104.000000

Cluster 3 Summary:
       review_count  average_stars          fans  compliment_hot  \
count    189.000000     189.000000    189.000000      189.000000
mean    1846.851852       3.950529    817.354497     3554.021164
std     1940.731821       0.276661   1077.000690     3282.363014
min      123.000000       3.240000     60.000000      383.000000
25%      816.000000       3.770000    298.000000     1729.000000
50%     1438.000000       3.950000    531.000000     2612.000000
75%     2331.000000       4.130000   1021.000000     4109.000000
max    17473.000000       4.830000  12497.000000    25784.000000

       compliment_more  compliment_profile  compliment_cute  compliment_list  \
count       189.000000          189.000000       189.000000       189.000000
mean        327.359788          334.746032       204.719577       116.703704
std         365.768831          540.162337       280.928023       250.832840
min          50.000000            3.000000         4.000000         0.000000
25%         161.000000          116.000000        59.000000        24.000000
50%         226.000000          211.000000       105.000000        57.000000
75%         368.000000          357.000000       230.000000       122.000000
max        3575.000000         5662.000000      1744.000000      2261.000000

       compliment_note  compliment_plain  compliment_cool  compliment_funny  \
count       189.000000        189.000000       189.000000        189.000000
mean       2032.296296       5571.222222      4932.793651       4932.793651
std        4275.366581       4061.812300      3306.769358       3306.769358
min         453.000000        685.000000       914.000000        914.000000
25%        1073.000000       3017.000000      2849.000000       2849.000000
50%        1501.000000       4511.000000      3887.000000       3887.000000
75%        2238.000000       6919.000000      5840.000000       5840.000000
max       59031.000000      28974.000000     20141.000000      20141.000000

       compliment_writer  compliment_photos
count         189.000000         189.000000
mean         1435.439153        2807.042328
std          1233.288647        3359.535934
min            80.000000          35.000000
25%           725.000000         601.000000
50%          1104.000000        1730.000000
75%          1643.000000        3509.000000
max          9821.000000       20573.000000
```

- You may customize the code above to try and group the users by different segments.