Virgin Atlantic Reviews Notebook 2: Data Analysis

The data for this notebook was obtained from Skytrax.com. We carried out the webscraping exercise in notebook one and in this notebook we continue with data analysis

In [27]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

In [28]:
```
reviews = pd.read_csv(r'C:\Users\imoge\AllMLProjects\Data\AirlineReviewsScraped.csv',
                      index_col=0
                     )
```

In [178…
```
reviews.shape
```

Out[178…
```
(500, 16)
```

In [29]:
```
reviews.head()
```

Out[29]:

	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Food Beverag
0	2025-06-23	Screen monitor is old	Maybe because of the route, food did not ca...	2025-06-01 00:00:00	Economy Class	London Heathrow	Mumbai	3	3	
1	2025-06-23	Seat for economy was comfortable	Food was decent. Seat for economy was comfo...	2025-06-01 00:00:00	Economy Class	Miami	London Heathrow	5	5	
2	2025-06-11	a great experience	First time with Virgin in over a decade but f...	2025-06-01 00:00:00	Premium Economy	Manchester	New York	4	5	
3	2025-05-23	most people wanted to just sleep	Really good experience in	2025-05-01 00:00:00	Premium Economy	New York	London Heathrow	5	5	

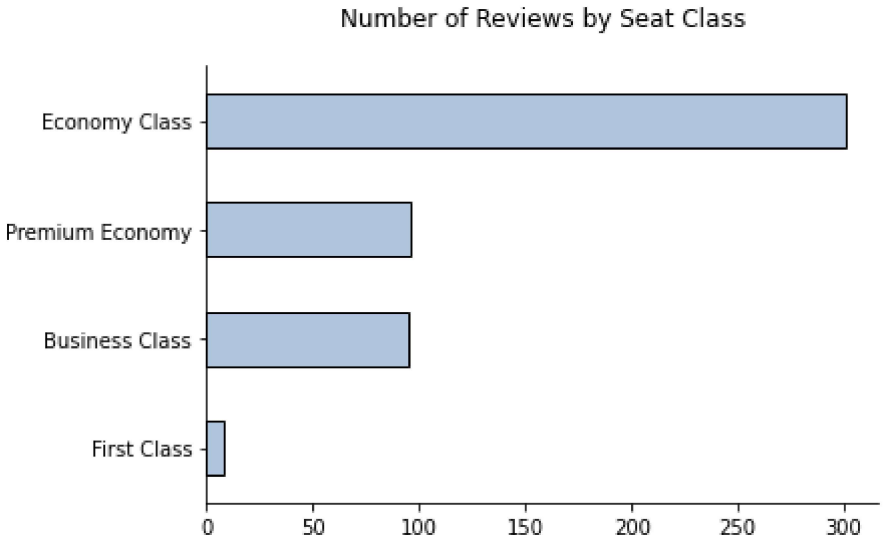
	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Food Beverag
			Premium Economy. ...							
4	2025-05-11	Good selection of movies	Premium economy had its own dedicated check...	2025-05-01 00:00:00	Premium Economy	London Heathrow	Orlando	5	4	

In [30]:
```
# Reset the review date to a datetime object
reviews['Review Date'] = pd.to_datetime(reviews['Review Date'])
```

In [31]:
```
print(reviews['Review Date'].min())
print(reviews['Review Date'].max())
```

```
2015-01-26 00:00:00
2025-06-23 00:00:00
```

In [32]:
```
### Seat Classes
seats_classes = reviews.groupby('Seat Class', as_index = False)['Review Date'].count().
ax = seats_classes.plot(kind = 'barh', color = 'lightsteelblue', ec = 'k', legend = Non
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.title('Number of Reviews by Seat Class', pad = 20)
plt.ylabel("");
```



In [33]:
```
# Number of reviews by departure location
from_location = reviews.groupby('From', as_index = False)['Review Date'].count().sort_v
from_location.head()
```

Out[33]:

	From	Review Date
25	London Heathrow	167
33	Not Provided	65
27	Manchester	39
24	London Gatwick	39
31	New York	24

In [34]:

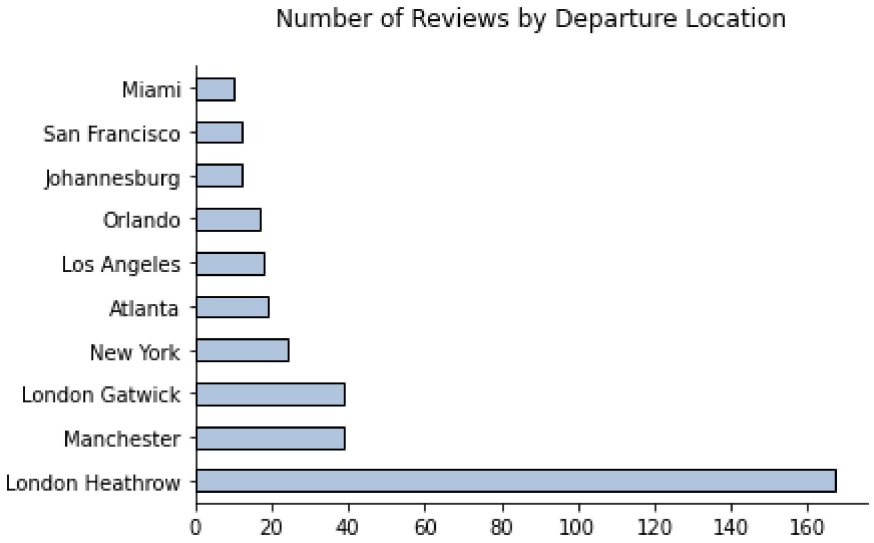
```
# Drop the 'Not provided'
from_location = from_location[from_location['From']!='Not Provided'].set_index('From').
from_location
```

Out[34]:

From	Review Date
London Heathrow	167
Manchester	39
London Gatwick	39
New York	24
Atlanta	19
Los Angeles	18
Orlando	17
Johannesburg	12
San Francisco	12
Miami	10

In [35]:

```
ax = from_location.plot(kind = 'barh', color = 'lightsteelblue',
                        ec = 'k',
                        legend = None)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.gca().set_ylabel("")
plt.title('Number of Reviews by Departure Location', pad = 20);
```



In [36]:

```
# Number of reviews by arrival location
to_location = reviews.groupby('To', as_index = False)['Review Date'].count().sort_value
to_location.head()
```

Out[36]:

	To	Review Date
23	London Heathrow	131
33	Not Provided	65
31	New York	48
34	Orlando	30
27	Manchester	19

In [37]:

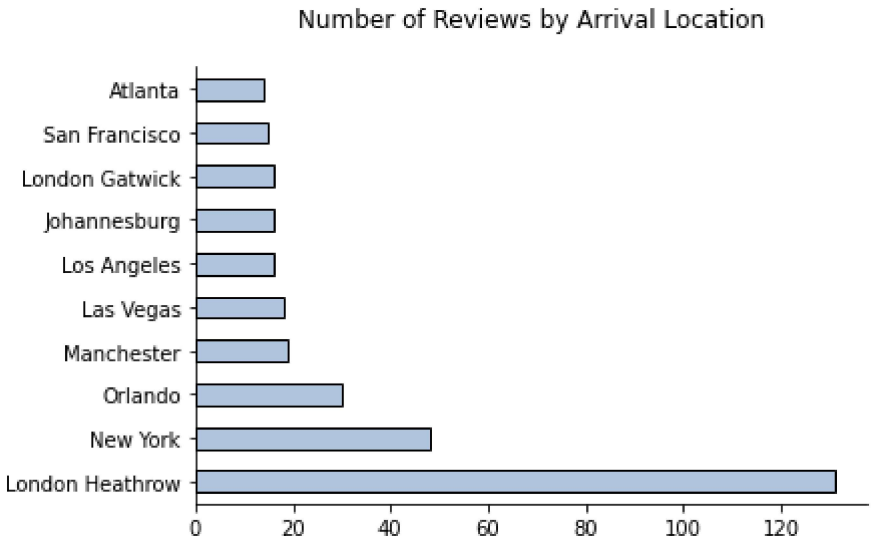
```
to_location = to_location[to_location['To']!='Not Provided'].set_index('To').head(10)
to_location
```

Out[37]:

To	Review Date
London Heathrow	131
New York	48
Orlando	30
Manchester	19
Las Vegas	18
Los Angeles	16
Johannesburg	16
London Gatwick	16
San Francisco	15
Atlanta	14

In [38]:

```
ax = to_location.plot(kind = 'barh',
                      color = 'lightsteelblue',
                      ec = 'k',
                      legend = None)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.gca().set_ylabel("")
plt.title('Number of Reviews by Arrival Location', pad = 20);
```

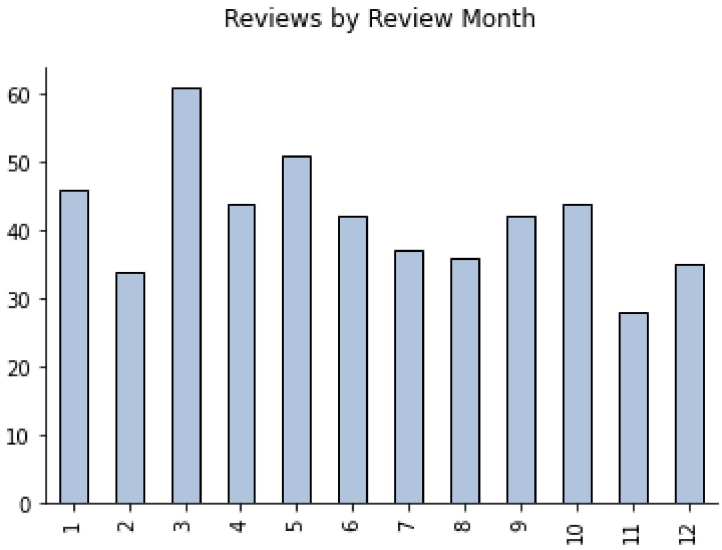


Most routes are to and from London Heathrow, which is not surprising given that this is the main airport Virgin operates from

In [39]:

```
# Reviews by Month
reviews['Month'] = reviews['Review Date'].dt.month
```

In [43]:
```
# Groupby month
ax = reviews.groupby('Month', as_index = False)['Review Date'].count().set_index('Month

plt.gca().set_xlabel("")
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.title('Reviews by Review Month', pad = 20);
```



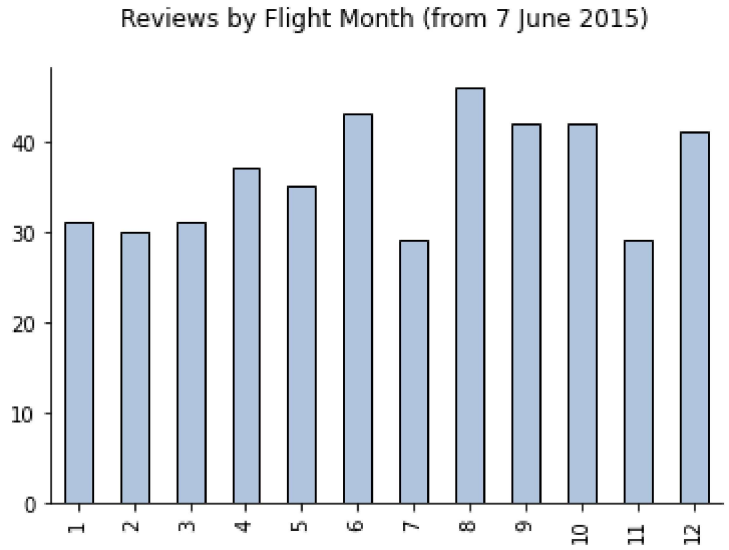
Peak review times for reviews to be left are in March with the least being left in November

We can analyse the flight dates for those flights we have information for

In [44]:
```
# Flight dates for those review we have information for
reviews_flight_dates = reviews[reviews['Flight Date']!='Not available']

# Convert the string column to datetime
reviews_flight_dates['Flight Date'] = pd.to_datetime(reviews_flight_dates['Flight Date'

# Extract the month
reviews_flight_dates['Flight Month'] = reviews_flight_dates['Flight Date'].dt.month
```

```
C:\Users\imoge\Anaconda3\envs\MachineLearning\lib\site-packages\ipykernel_launcher.py:5:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  """
C:\Users\imoge\Anaconda3\envs\MachineLearning\lib\site-packages\ipykernel_launcher.py:8:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [60]:
```
# Groupby month
ax = reviews_flight_dates.groupby('Flight Month', as_index = False)['Flight Date'].coun



plt.title('Reviews by Flight Month (from 7 June 2015)', pad = 20)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xlabel("");
```



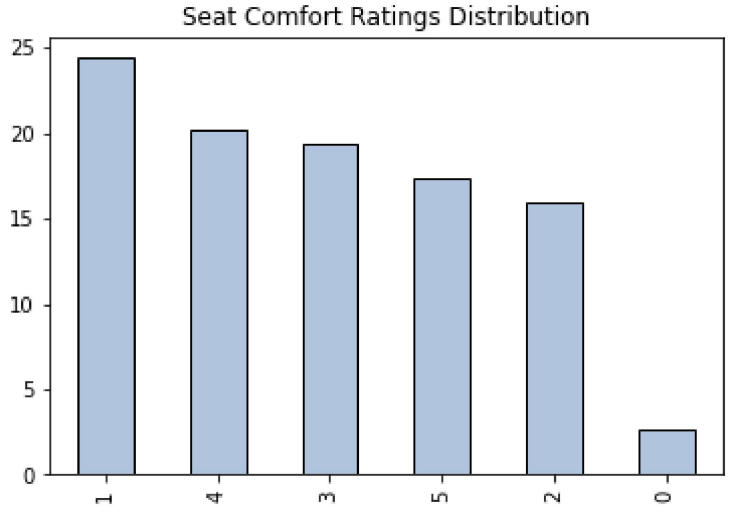
Most flights were the flight date was provided were taken in August and the least in July and November

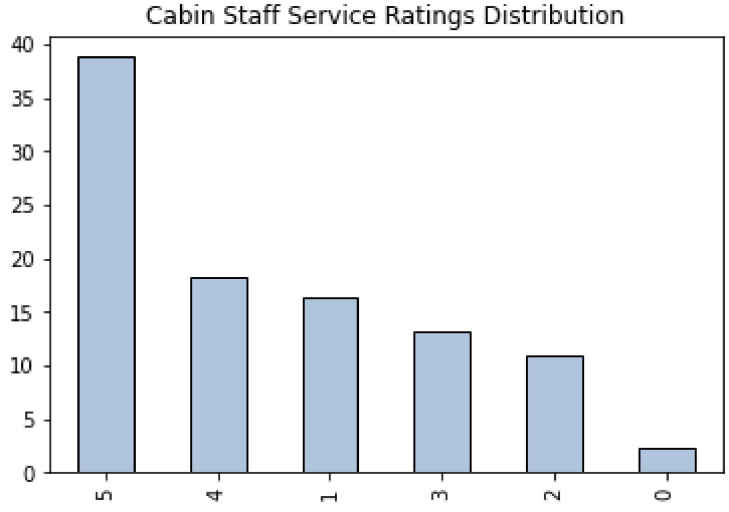
Let's have a look at the starred categories. We filled any nan values with zeros (as it was not possible to leave a zero rating, this was not a problem for our data) so we will filter our dataframe to just pull out those reviews which have a score above 0 in those columns

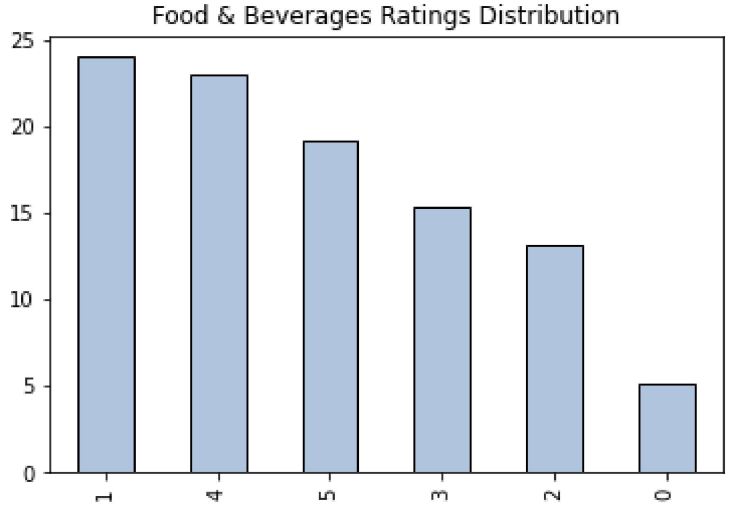
In [62]:
```
# Categories given ratings
list_cats = reviews.columns[7:-1]
```

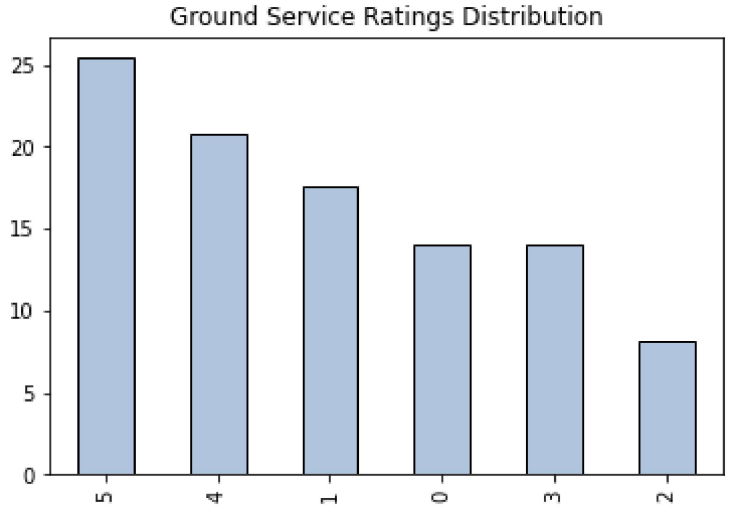
In [63]:
```
# Plot the review ratings for the categories - a rating of zero means the category wasn
def review_ratings(cat):
    pd.DataFrame(reviews[cat].value_counts(normalize = True)*100).plot(kind = 'bar',
                                                                       color = 'l
                                                                       ec = 'k',
                                                                      legend = Fa
                                                                       title = ca
```

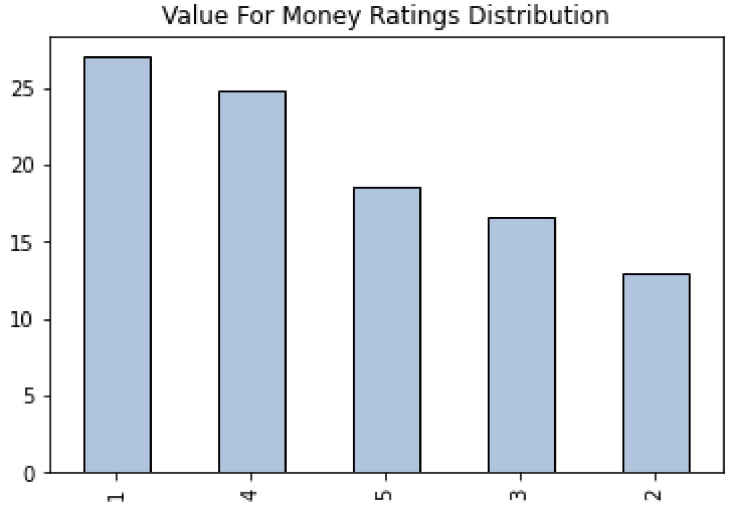
In [64]:
```
for i in list_cats:
    review_ratings(i)
```

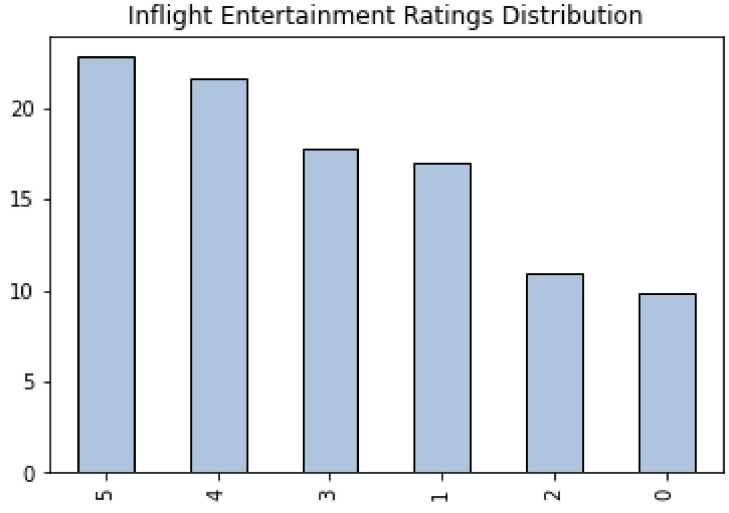

Seat Comfort Ratings Distribution
25
20
15
10
5
0
1
4
3
5
2
0

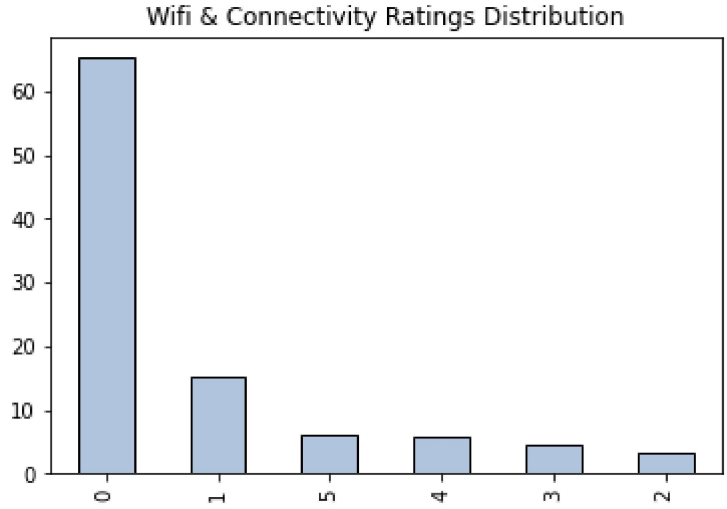

Cabin Staff Service Ratings Distribution
40
35
30
25
20
15
10
5
0
5
4
1
3
2
0

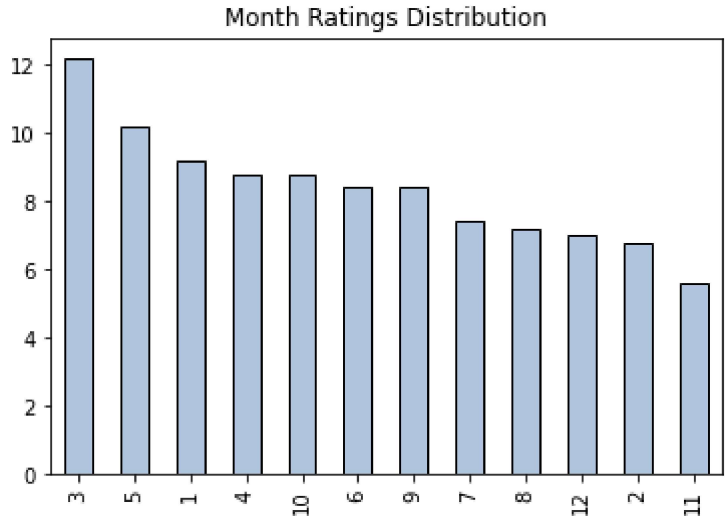

Food & Beverages Ratings Distribution
25
20
15
10
5
0
1
4
5
3
2
0


Ground Service Ratings Distribution
25
20
15
10
5
0
5
4
1
0
3
2


Value For Money Ratings Distribution
25
20
15
10
5
0
1
4
5
3
2


Inflight Entertainment Ratings Distribution
20
15
10
5
0
5
4
3
1
2
0





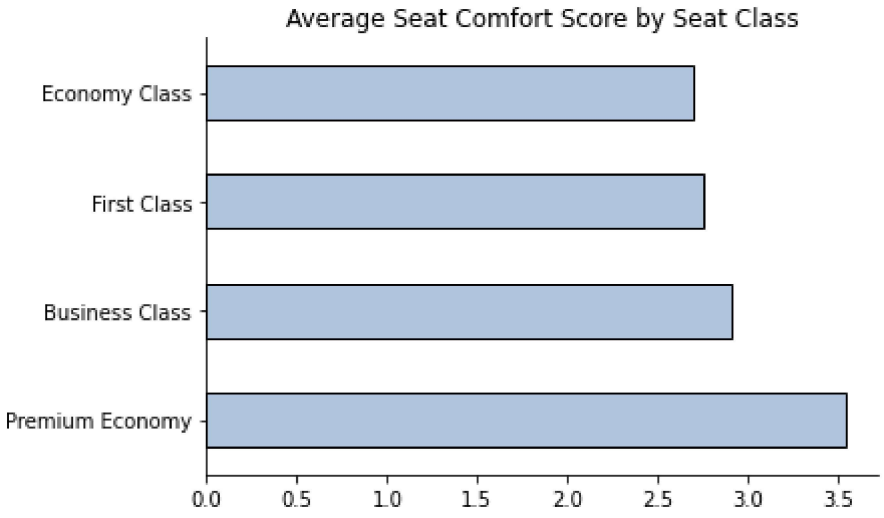
Most ratings given for seat comfort were 1 and the most ratings for food and beverages were 5 stars. Let's have a more in-depth look at this

In [65]:
```
selected = reviews[['Seat Class','Seat Comfort','Cabin Staff Service','Food & Beverages
        'Wifi & Connectivity','Value For Money']]
```

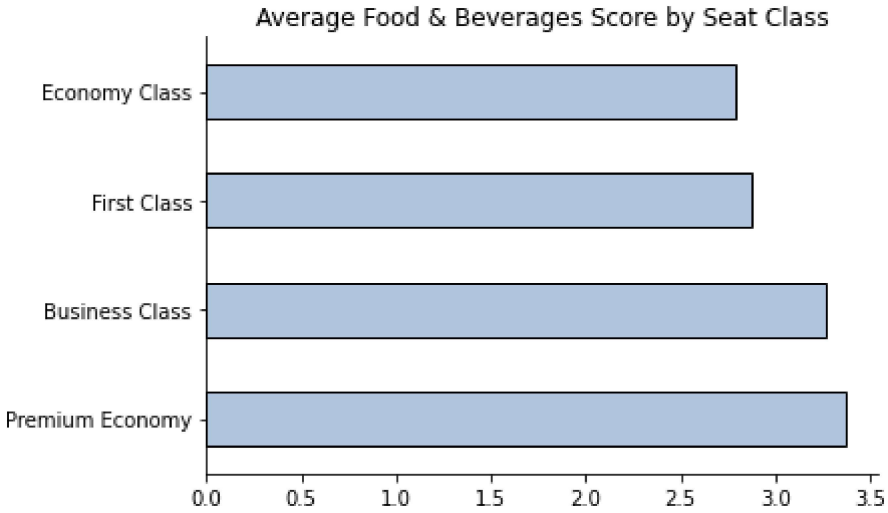
As they seem to be particularly unhappy with a key part of the service, which is the comfort of the seat, we can see if this varies by the type of seat class

In [76]:
```
# Function to plot the mean score for each category where a review has been given
def plot_scores(cat):
    rev = selected[reviews[cat]>0]
    score = rev.groupby('Seat Class', as_index = False)[cat].mean().sort_values(by = ca
    ax = score.plot(kind = 'barh',
              color = 'lightsteelblue',
              ec = 'k',
              legend = False,
              title = 'Average ' + cat + ' ' + 'Score by Seat Class')
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    plt.gca().set_ylabel("");
```

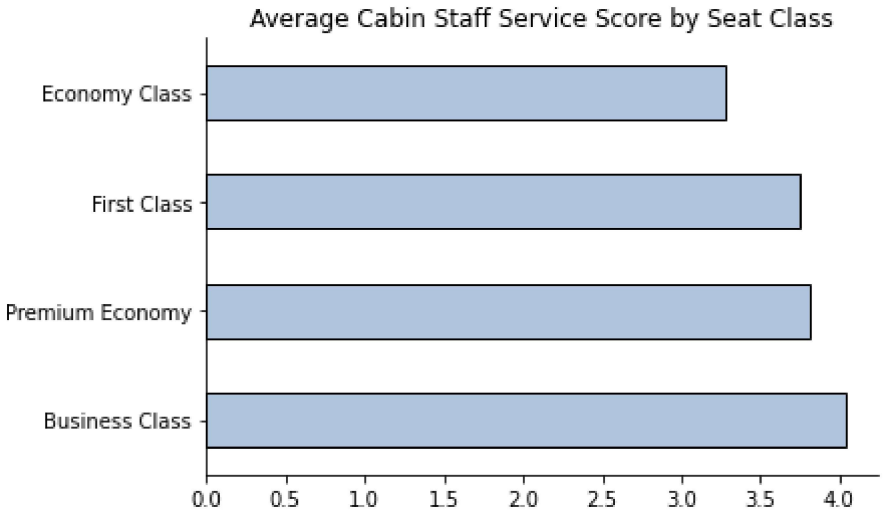
In [77]:
```
plot_scores('Seat Comfort')
```



In [78]:
```
plot_scores('Food & Beverages')
```

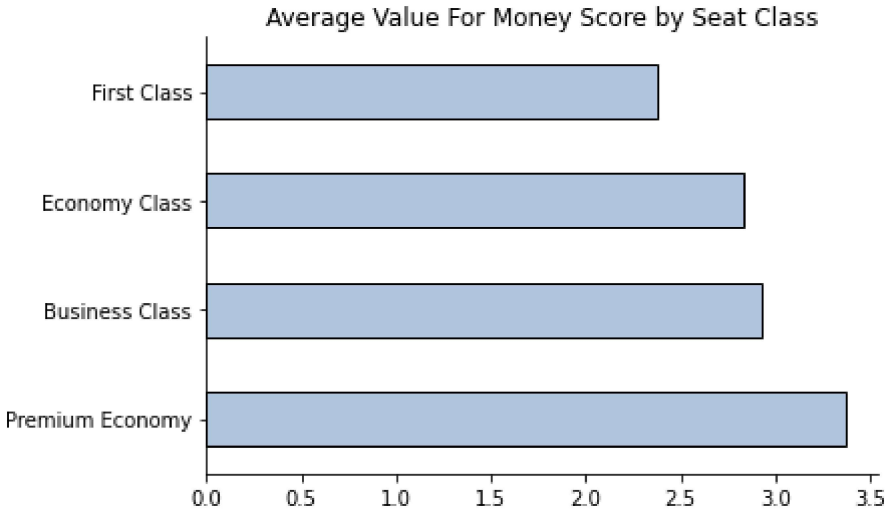


In [79]:
```
plot_scores('Cabin Staff Service')
```

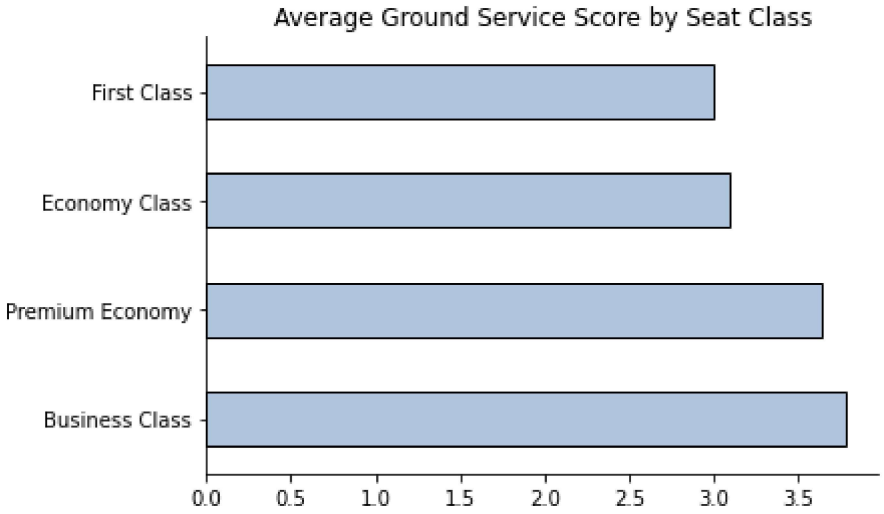


First class passengers get a lot of individual attention from cabin crew with dedicated staff, so they seemed to rate this higher than the other passengers

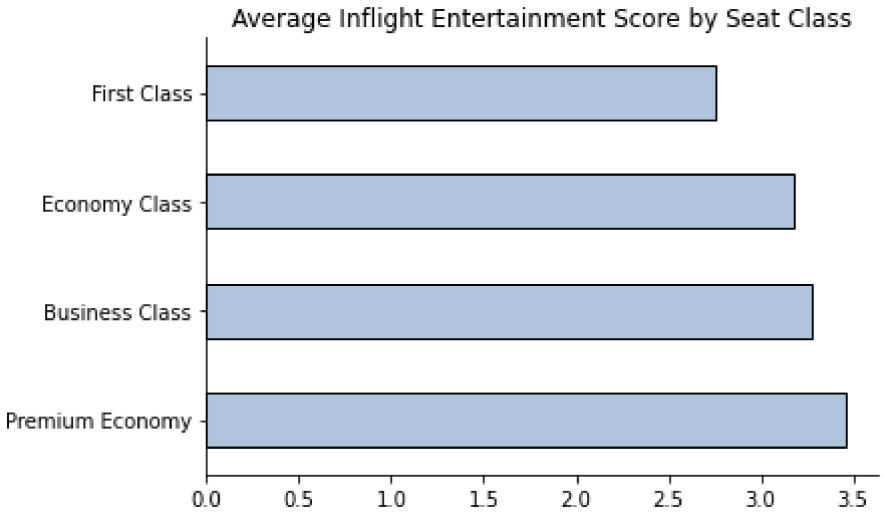
In [80]:
```
plot_scores('Value For Money')
```



In [82]:
```
plot_scores('Ground Service')
```

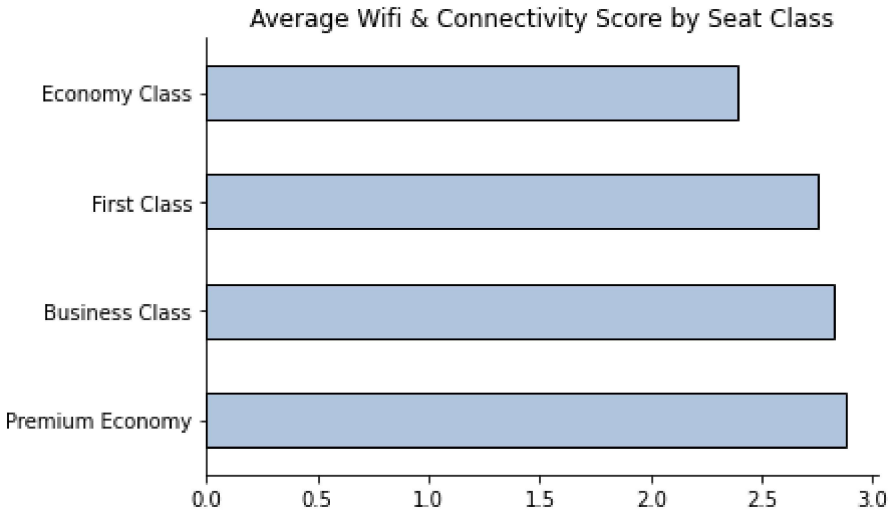


In [83]:
```
plot_scores('Inflight Entertainment')
```



There is little difference for inflight entertainment which is to be expected as all passengers generally get the same video options onboard but the rating for first class is the lowest where expectations might be higher.

In [84]:
```
plot_scores('Wifi & Connectivity')
```



Wifi may not be used by most passengers. For those that did use it and gave a rating, the highest was by business class passengers and the lowest by first class

In [85]:
```
# Looking at economy which is our biggest category
selected2 = reviews[['Seat Class','From','Seat Comfort','Cabin Staff Service','Food & B
        'Wifi & Connectivity','Value For Money']]
```

In [86]:
```
# Group by departure airport and get the places with the lowest mean rating for seat co
selected2[(selected2['Seat Class']== 'Economy Class') & (selected2['From']!= 'Not Provi
```

Out[86]:

	From	Seat Comfort
18	Kagi	1.0
24	Melbourne	1.0
32	Tampa	1.0
31	St Lucia	1.0
29	Perth	1.0

In [87]:

```
# Group by departure airport and get the places with the highest mean rating for seat c
selected2[(selected2['Seat Class']== 'Economy Class') & (selected2['From']!= 'Not Provi
```

Out[87]:

	From	Seat Comfort
15	Honolulu	5.000000
0	Aberdeen	4.000000
4	Barbados	4.000000
11	Dulles	4.000000
27	New York	3.692308

Text Analysis

We can find out average length of review, use an out of box sentiment analyser to get sentiment on each review and then we can analyse this by class. We could also determine topics to see if there are any other categories that people are complaining about and also whether the average sentiment we calculate from the text compares to the overall sentiment for the categories for the airline as a whole

In [88]:

```
reviews2 = reviews.copy()
```

In [89]:

```
# Function to count words
def count_words(text):
    return len(text.split())

# Apply the function to the 'text' column
reviews2['Length'] = reviews2['Review Text'].apply(count_words)
```

In [90]:

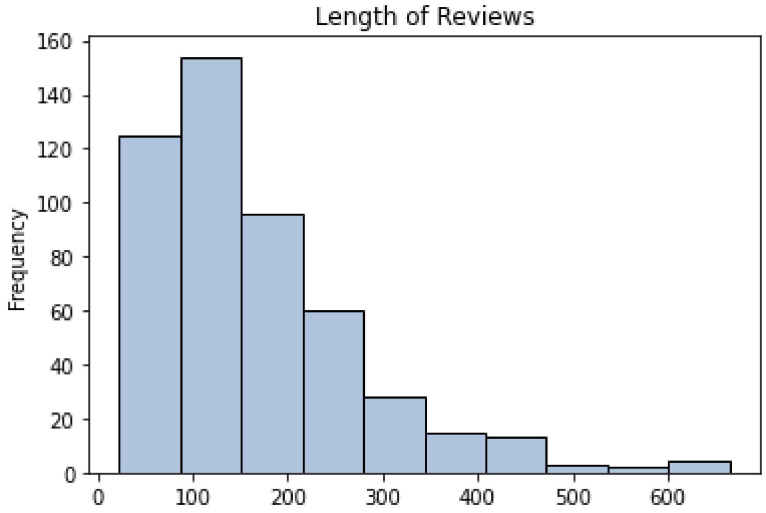
```
# Get stats
reviews2['Length'].describe()
```

Out[90]:

```
count    500.000000
mean     165.848000
std      110.263652
min       23.000000
25%       87.750000
```

```
50%      137.000000
75%      215.250000
max      665.000000
Name: Length, dtype: float64
```

In [91]:
```
reviews2['Length'].plot(kind = 'hist', color = 'lightsteelblue',ec = 'k', title = 'Leng
```



The maximum length of a review is 665 words and the minimum 23 words. The mean is more than the median, indicating some skew which is not surprising and is clearly shown in the histogram

In [92]:
```
# Reviews longer than 500 words
reviews2[reviews2['Length']>500]
```

Out[92]:

	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Be
87	2022-01-05	Virgin has cut back everything	Arrived at LHR to be greeted by Virgin groun...	2021-10-01 00:00:00	Premium Economy	London Heathrow	Antigua	4	4	
56	2018-12-04	I thought it a good value product	Bridgetown to London. After leaving a cruise...	2018-12-01 00:00:00	Economy Class	Barbados	London Heathrow	3	4	
4	2017-08-14	not be travelling Virgin again	Verified Review Gatwick to Orlando return. In...	2017-07-01 00:00:00	Economy Class	Orlando	London Gatwick	2	1	
36	2017-02-28	lost the extra legroom seats	Verified Review My booking was with Virgin At...	2017-02-01 00:00:00	Economy Class	Melbourne	London Heathrow	1	2	
47	2017-01-17	never flying	Verified Review	2016-12-01	Business Class	Atlanta	London Heathrow	3	3	

	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Be
		Virgin again	Atlanta to London. I've long ...	00:00:00						
51	2017-01-11	we have had no apology	Verified Review London Heathrow to Johannesbu...	2016-12-01 00:00:00	Economy Class	London Heathrow	Johannesburg	2	4	
8	2016-04-08	squeezed in as many seats as possible	Flew Los Angeles to London Heathrow. It's been...	2016-04-01 00:00:00	Premium Economy	Los Angeles	London Heathrow	1	1	
85	2015-03-04	Virgin Atlantic customer review	Flew ABZ-LHR-EWR returning JFK-LHR-ABZ. Outwar...	Not available	Economy Class	Not Provided	Not Provided	3	4	

In [97]:
```
# Does the length of review vary by seat class?
lengths = pd.DataFrame(reviews2.groupby('Seat Class')['Length'].mean()).sort_values(by
lengths['Length'] = round(lengths['Length'],1)
lengths
```

Out[97]:

Seat Class	Length
Business Class	174.0
Premium Economy	168.9
Economy Class	162.7
First Class	152.5

People flying business seem to leave the longest reviews and those flying by first class the shortest

In [98]:
```
# import SentimentIntensityAnalyzer class from vaderSentiment.vaderSentiment module.
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

In [99]:
```
# Initialize the VADER sentiment intensity analyzer
sia = SentimentIntensityAnalyzer()

# Function to get sentiment scores
def get_sentiment_scores(text):
    return sia.polarity_scores(text)
```

```
# Apply the function to the 'text' column
reviews2['sentiment_scores'] = reviews2['Review Text'].apply(get_sentiment_scores)

# Optionally: Split the sentiment scores into separate columns
reviews_sentiment = reviews2['sentiment_scores'].apply(pd.Series)

# Combine the original DataFrame with the sentiment scores
reviews_all = pd.concat([reviews2, reviews_sentiment], axis=1)

# Drop the combined columm
reviews_all.drop(columns = ['sentiment_scores'],axis = 1, inplace = True)

# Display the result
reviews_all.head()
```

Out[99]:

	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Food Beverag
0	2025-06-23	Screen monitor is old	Maybe because of the route, food did not ca...	2025-06-01 00:00:00	Economy Class	London Heathrow	Mumbai	3	3	
1	2025-06-23	Seat for economy was comfortable	Food was decent. Seat for economy was comfo...	2025-06-01 00:00:00	Economy Class	Miami	London Heathrow	5	5	
2	2025-06-11	a great experience	First time with Virgin in over a decade but f...	2025-06-01 00:00:00	Premium Economy	Manchester	New York	4	5	
3	2025-05-23	most people wanted to just sleep	Really good experience in Premium Economy. ...	2025-05-01 00:00:00	Premium Economy	New York	London Heathrow	5	5	
4	2025-05-11	Good selection of movies	Premium economy had its own dedicated check...	2025-05-01 00:00:00	Premium Economy	London Heathrow	Orlando	5	4	

5 rows × 21 columns

Score Range: The compound score ranges from -1 to +1:

-1 indicates extremely negative sentiment. 0 indicates neutral sentiment. +1 indicates extremely positive sentiment.

Thresholds for Interpretation:

Negative Sentiment: A compound score less than 0 (e.g., -1 to -0.01) suggests a negative sentiment.
Neutral Sentiment: A compound score around 0 (typically -0.05 to +0.05) indicates neutral sentiment.
Positive Sentiment: A compound score greater than 0 (e.g., 0.01 to +1) suggests a positive sentiment.

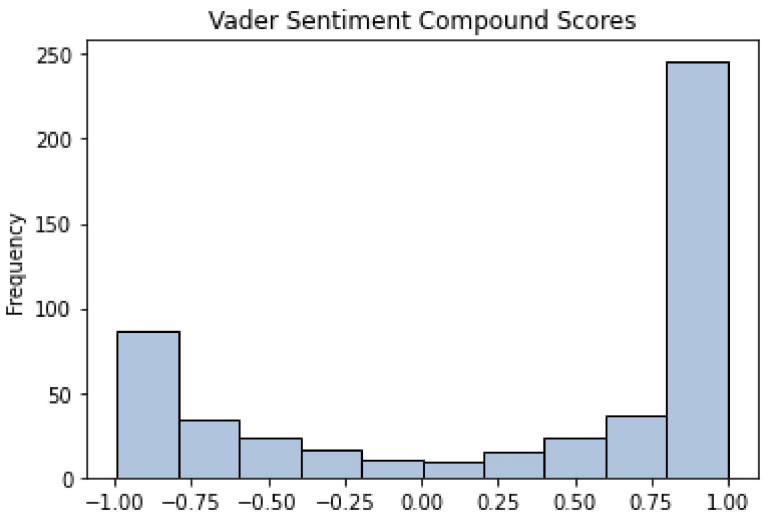
General Guidelines:

Scores closer to -1 or +1 indicate stronger sentiments, while scores closer to 0 indicate weaker sentiments.

In [100…
```
# Mean score for the compound
reviews_all['compound'].mean()
```

Out[100… 0.31143940000000003

In [101…
```
reviews_all['compound'].plot(kind = 'hist', ec = 'k', color = 'lightsteelblue', title =
```



Overall from the text, we can see that most comments are falling into being positive

In [102…
```
# Sentiment by length of review - are negative reviews more likely to be longer or shor
print("Negative reviews length:",reviews_all[reviews_all['compound']< -0.05]['Length'].
print("Positive reviews length:",reviews_all[reviews_all['compound']> 0.01]['Length'].m
```

```
Negative reviews length: 177.12048192771084
Positive reviews length: 160.62006079027356
```

Negative reviews are a bit longer than positive ones on average

In [355…
```
# Check a sample
reviews_all.iloc[0]['Review Text']
```

Out[355…

```
'   I’m extremely disappointed with the appalling customer service I’ve received from Vi
rgin Atlantic. It has been over 35 days since I submitted a refund request for a duplica
te charge and an issue with the condition of my seat, and despite numerous emails, phone
calls, and promises of resolution, I’m still no closer to receiving the refund I’m owed.
The way my case has been handled has been nothing short of frustrating. I’ve been repeat
edly told that the refund is “being processed” or that they’re “waiting for my bank deta
ils” – despite the fact that I’ve already provided this information multiple times. It’s
incredibly frustrating to be met with silence, delays, and unsatisfactory gestures of co
mpensation, including the offer of points that require me to spend even more money to re
deem. It’s clear that Virgin Atlantic doesn’t take customer issues seriously,'
```

In [356…

```
reviews_all.iloc[0]['compound']
```

Out[356…

```
-0.6575
```

In [358…

```
reviews_all.iloc[25]['Review Text']
```

Out[358…

```
"  Great flights. Very good food, the mile high tea on the lhr-jfk leg was a really nice
touch. A huge selection of music and movies to choose from. Friendly flight attendants a
vailable and happy to provide an extra drink or snack, including a fine selection of alc
oholic beverages free. Good selection of items in the onboard duty free. The great servi
ce even starts on the ground, i.e. I'd injured my hand shortly before flying, and was al
lowed to check my carryon for free. Virgin sets the bar for great customer service, all
airlines should aspire to be this good."
```

In [359…

```
reviews_all.iloc[25]['compound']
```

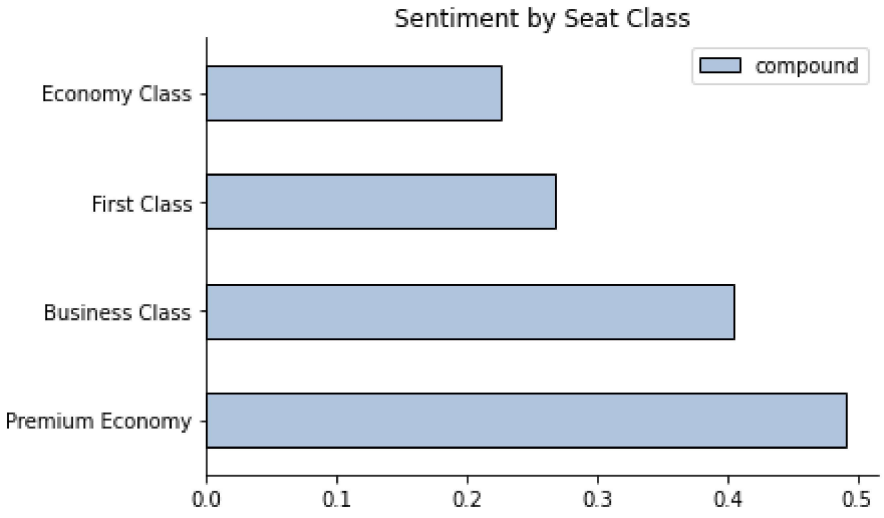
Out[359…

```
0.9929
```

Looks pretty good from a sample of a couple of the reviews

In [104…

```
# Lets look at the reviews by seat class
sent_seat = reviews_all.groupby('Seat Class', as_index = False)['compound'].mean().sort
ax = sent_seat.plot(kind = 'barh', title = 'Sentiment by Seat Class', color = 'lightste
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

plt.gca().set_ylabel("");
```



The most positive scores are associated with premium economy seats

In [105…

```
# Lets look at the reviews by departure location
sent_dep = reviews_all.groupby('From', as_index = False)['compound'].mean().sort_values
sent_dep.columns = ['Compound Sentiment Score']
sent_dep
```

Out[105…

From	Compound Sentiment Score
Antigua	0.98720
Honolulu	0.97900
Tampa	0.96590
Aberdeen	0.95605
Dulles	0.94670
Barbados	0.93080
Shanghai	0.88665
Cork	0.87810
Cancun	0.86810
Newark	0.81560

Flights from Antigua seem to have a high positive score which is good. However, we need to be careful as there may not be many reviews. Lets look.

In [106…

```
# Number of reviews from Antigua
reviews_all[reviews_all['From']=='Antigua']
```

Out[106…

	Review Date	Review Title	Review Text	Flight Date	Seat Class	From	To	Seat Comfort	Cabin Staff Service	Food & Beverages	…	N
9	2025-02-26	Very cramped space for legs	Very cramped space for legs, I am 186 (6"1) a...	2025-02-01 00:00:00	Economy Class	Antigua	London Heathrow	2	5	3	…	

1 rows × 21 columns

We can see that there is just one review making up this mean compound score, so we should be careful in making conclusions from this limited data. It might be better to have a look at the scores for the top airports that flights fly from and to.

In [107…

```
# Mean score for reviews for flights leaving from or arriving at Heathrow
round(reviews_all[(reviews_all['From'] =='London Heathrow') | (reviews_all['To'] == 'Lo
```

Out[107… 0.26

Quite a low score relative to other locations. Does it differ as to whether you are leaving or arriving at London?

In [108…

```
# Mean score for reviews for flights leaving from Heathrow
round(reviews_all[reviews_all['From'] =='London Heathrow']['compound'].mean(),2)
```

Out[108… 0.34

In [109…

```
# Mean score for reviews for flights arriving to Heathrow
round(reviews_all[reviews_all['To'] =='London Heathrow']['compound'].mean(),2)
```

Out[109… 0.16

There is a difference here suggesting that passengers arriving back into London Heathrow give a more negative review than those leaving from Heathrow. We need to be careful here as there might be all kinds of reasons for this. People heading out, especially for holidays are likely to be more positive. Many of the flights out of Heathrow will be day flights and flights landing back are often night flights when people are tired and this could also affect the rating. Also, the airport itself can obviously have an impact, whether there are delays, weather issues out etc and this might get reflected in the rating for the airline even if it has nothing to do with it.

In [110…

```
# Simple wordcloud to see what people are raising in the text
from wordcloud import WordCloud, STOPWORDS
```

In [213…

```
# Create wordcloud
def create_cloud(text):

    stop_words = ["Virgin","Atlantic","airline","flight","plane","told"] + list(STOPWOR

    wordcloud = WordCloud(width=800, height=400,
                      background_color='white',
                      colormap='viridis',
                      stopwords = stop_words,
                      collocation_threshold = 3,
                      ).generate(text)

    # Display the generated image
    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.title("All Reviews WordCloud", fontsize=18, pad=20)
    plt.axis('off')  # Turn off axis numbers and ticks
    plt.show()
```

In [214…

```
all_text = ''.join(reviews_all['Review Text'])
create_cloud(all_text)
```

In [215…

```
# Step 1: Create bigrams and trigrams
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english')
X = vectorizer.fit_transform(reviews_all['Review Text'])

# Step 2: Get frequencies
freqs = X.toarray().sum(axis=0)
ngram_freq = dict(zip(vectorizer.get_feature_names(), freqs))

# Step 3: Generate wordcloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_fr

# Step 4: Show it
plt.figure(figsize=(14, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("All Reviews Bigrams and Trigrams WordCloud", fontsize=18, pad=20)
plt.show()
```

All Reviews Bigrams and Trigrams WordCloud

We can see that the seat, the food and the level of service are the most frequent words raised in the reviews in line with the categories that the review uses and what we have already derived from the analysis of star ratings. We could split out the positive and negative reviews and see if different categories are highlighted.

In [174…

```
# Split dataframe based on thresholds
neg = reviews_all[reviews_all['compound']< -0.05]
pos = reviews_all[reviews_all['compound']> 0.01]
```

In [175…

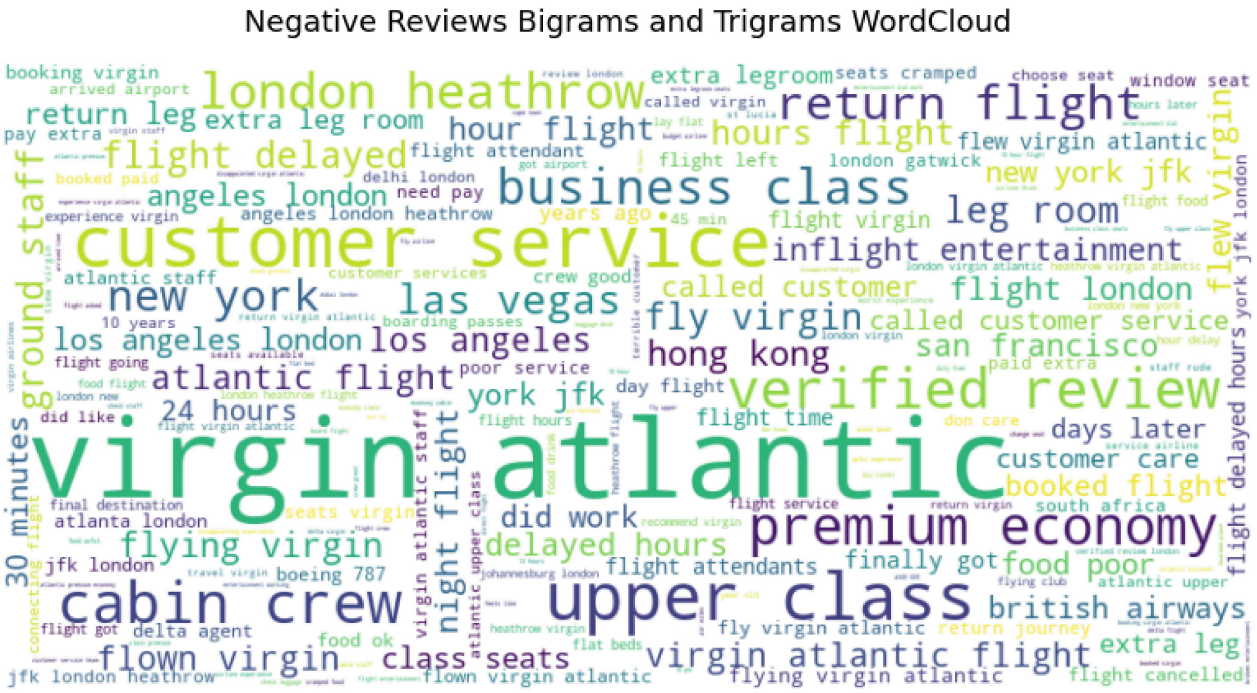
```
# Create text
text_neg = ''.join(neg['Review Text'])
text_pos = ''.join(pos['Review Text'])
```

In [176…

```
create_cloud(text_neg)
```

In [217…

```
# Step 1: Create bigrams and trigrams
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english')
X = vectorizer.fit_transform(neg['Review Text'])

# Step 2: Get frequencies
freqs = X.toarray().sum(axis=0)
ngram_freq = dict(zip(vectorizer.get_feature_names(), freqs))

# Step 3: Generate wordcloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_fr

# Step 4: Show it
plt.figure(figsize=(14, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Negative Reviews Bigrams and Trigrams WordCloud", fontsize=18, pad=20)
plt.show()
```



In [177…

```
create_cloud(text_pos)
```

In [218…

```
# Step 1: Create bigrams and trigrams
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english')
X = vectorizer.fit_transform(pos['Review Text'])

# Step 2: Get frequencies
freqs = X.toarray().sum(axis=0)
ngram_freq = dict(zip(vectorizer.get_feature_names(), freqs))

# Step 3: Generate wordcloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_fr

# Step 4: Show it
plt.figure(figsize=(14, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Positive Reviews Bigrams and Trigrams WordCloud", fontsize=18, pad=20)
plt.show()
```

We could extend this analysis further using a topic model such as LDA, NMF or one of the newer models leveraging LLMs such as BERTopic. Topic models generally work best with lots of data, so perhaps we will return to this at another time after scraping many more reviews possibly across different airlines.

A Deeper Dive

Investigating the word "time"

In [191…
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [ ]:
```
time_text = neg[neg['Review Text'].str.contains("time")]['Review Text']
```

In [194…
```
# Step 1: Create bigrams and trigrams
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english')
X = vectorizer.fit_transform(time_text)

# Step 2: Get frequencies
freqs = X.toarray().sum(axis=0)
ngram_freq = dict(zip(vectorizer.get_feature_names(), freqs))

# Step 3: Generate wordcloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_fr

# Step 4: Show it
plt.figure(figsize=(14, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Bigrams and Trigrams WordCloud", fontsize=18, pad=20)
plt.show()
```

Bigrams and Trigrams WordCloud

In [198…

```
heathrow_text = neg[neg['Review Text'].str.contains("Heathrow")]['Review Text']
```

In [199…

```
# Step 1: Create bigrams and trigrams
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english')
X = vectorizer.fit_transform(heathrow_text)

# Step 2: Get frequencies
freqs = X.toarray().sum(axis=0)
ngram_freq = dict(zip(vectorizer.get_feature_names(), freqs))

# Step 3: Generate wordcloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_fr

# Step 4: Show it
plt.figure(figsize=(14, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Bigrams and Trigrams WordCloud", fontsize=18, pad=20)
plt.show()
```