Notebook

June 17, 2021

Question 2.3: `Translated_Review` contains the text of the reviews. Create a new column in `reviews_df` called `Review_Length` that contains the number of words in each review (for this question assume that words are seperated by white space). Then use the DataFrame `describe()` function to print descriptive statistics about the length of the reviews.

[131]:
```
reviews_df["Review_Length"] = reviews_df["Translated_Review"].map(lambda x:␣
↪len(x.split()))
reviews_df
```

[131]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity	Review_Length
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking …	Positive	1.000000	0.533333	21
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive	0.250000	0.288462	7
3	10 Best Foods for You	Works great especially going grocery store	Positive			
4	10 Best Foods for You	Best idea us	Positive			
5	10 Best Foods for You	Best way	Positive			
…	…	…	…			
64220	Housing-Real Estate & Property	Most ads older many agents ..not much owner po…	Positive			
64221	Housing-Real Estate & Property	If photos posted portal load, fit purpose. I'm…	Positive			
64224	Housing-Real Estate & Property	Dumb app, I wanted post property rent give opt…	Negative			
64225	Housing-Real Estate & Property	I property business got link SMS happy perform…	Positive			
64228	Housing-Real Estate & Property	Useless app, I searched flats kondapur, Hydera…	Negative			

```
3                0.400000                0.875000                6
4                1.000000                0.300000                3
5                1.000000                0.300000                2
…                     …                       …                …
64220            0.173333                0.486667               22
64221            0.225000                0.447222               29
64224           -0.287500                0.250000               15
64225            0.800000                1.000000               15
64228           -0.316667                0.400000               19

[37425 rows x 6 columns]
```

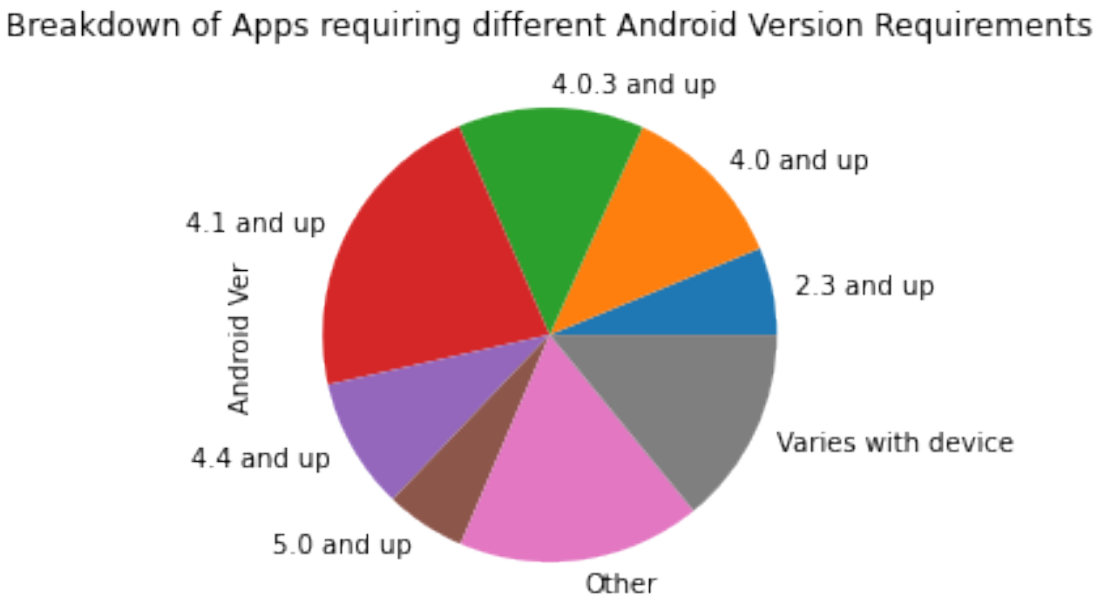
[132]:
```
reviews_df["Review_Length"].describe()
```

[132]:
```
count    37425.000000
mean        18.350007
std         16.646923
min          1.000000
25%          6.000000
50%         14.000000
75%         26.000000
max        345.000000
Name: Review_Length, dtype: float64
```

Question 2.4: Based on the descriptive statistics, do you think the mean is roughly equal to, higher, or lower the median value. Justify

Higher (Mean = 18.35, 50%/Median = 14.00)

Question 3.1: Produce a pie chart with the Android Ver requirements for the different apps. Group together all versions that make up less than 5% of the total apps into a single `Other` category. This should look similar to



. Don't forget to include a title for the figure.

Hint 1: You will find the df.value counts() function useful for solving this problem.

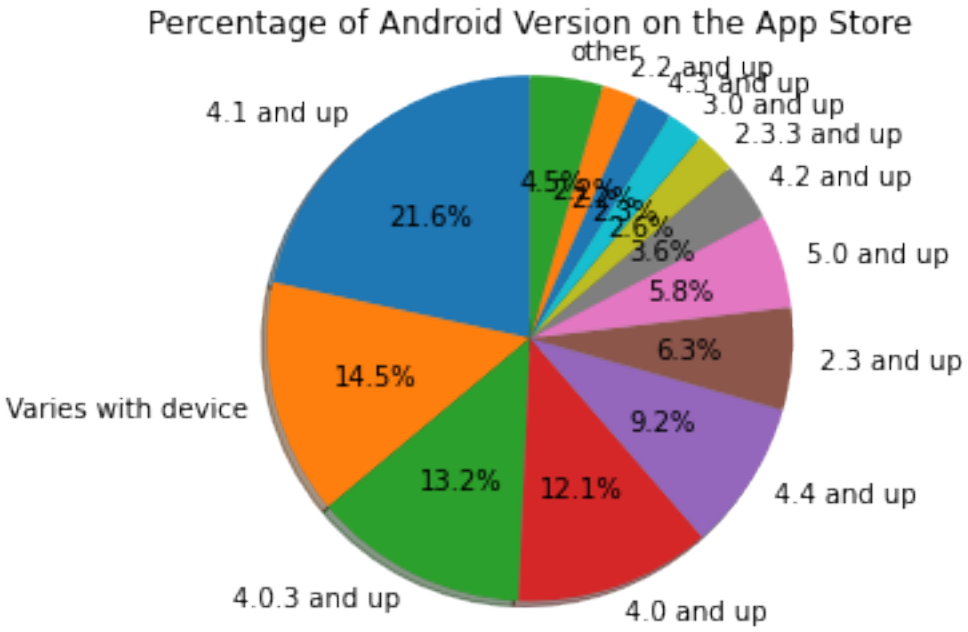
Hint 2: This stackoverflow answer will be useful.

[134]:
```
versions = apps_df["Android Ver"].value_counts()
summation = sum(apps_df["Android Ver"].value_counts())
other = sum(versions[12:])
chart = versions[:12]
chart["other"] = other
chart
```

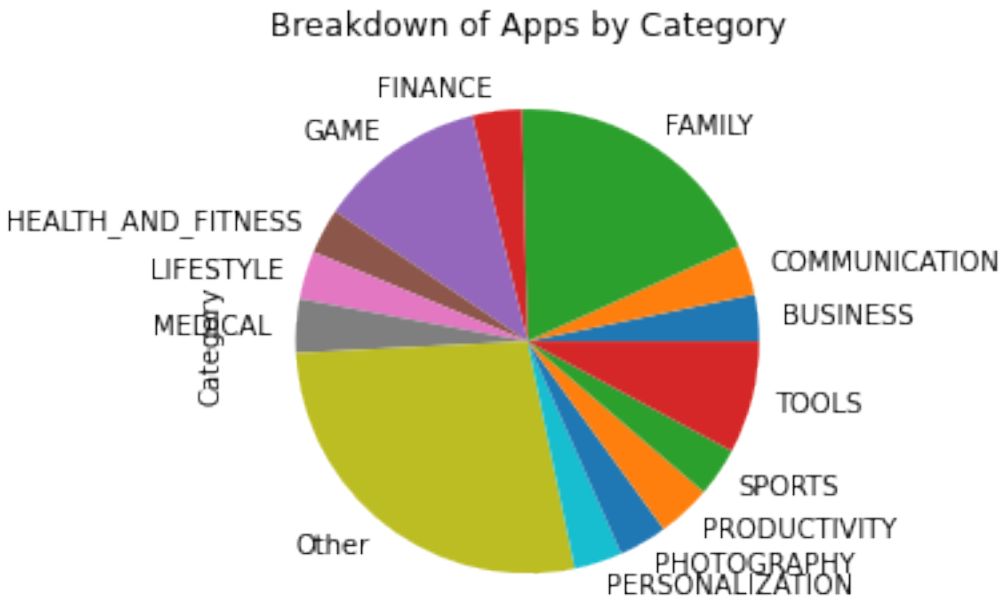
[134]:
```
4.1 and up            1962
Varies with device    1318
4.0.3 and up          1201
4.0 and up            1096
4.4 and up             840
2.3 and up             572
5.0 and up             526
4.2 and up             325
2.3.3 and up           234
3.0 and up             206
4.3 and up             203
2.2 and up             201
other                  406
Name: Android Ver, dtype: int64
```

[135]:
```
import matplotlib.pyplot as plt

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
```

```
labels = chart.index
sizes = chart

fig1, ax1 = plt.subplots()
ax1.pie(sizes, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title("Percentage of Android Version on the App Store")


plt.show()
```



Question 3.2: Create a similar pie chart for app `Category`. In this case, group together categories that make up less than 3% of the apps. The resulting graph should look something like



.

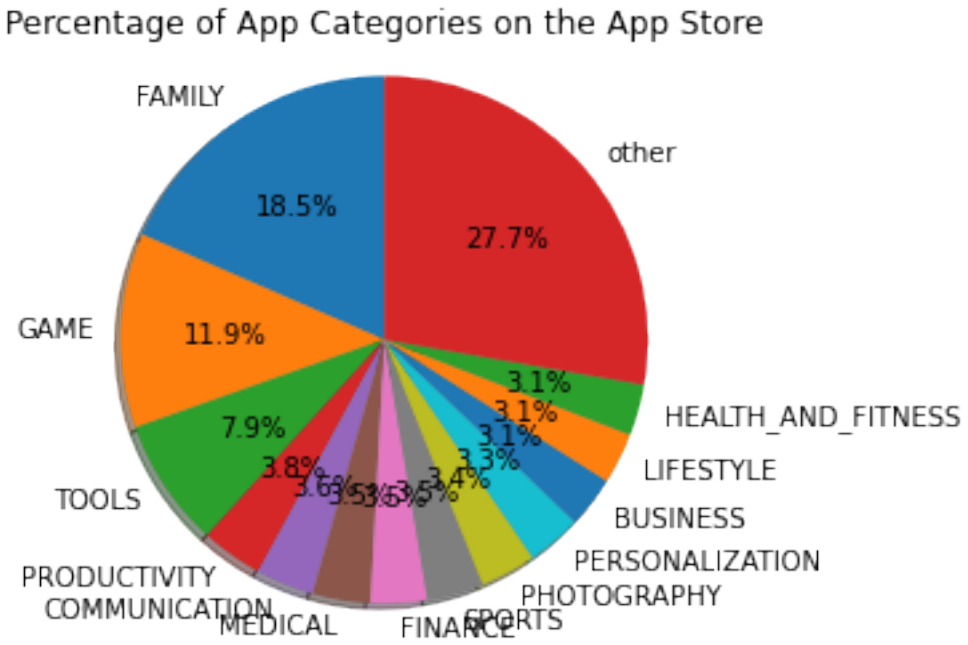
[136]:
```
categories = apps_df["Category"].value_counts()
summation = sum(apps_df["Category"].value_counts())
other2 = sum(categories[13:])
chart2 = categories[:13]
chart2["other"] = other2
chart2
```

[136]:
```
FAMILY                1680
GAME                  1085
TOOLS                  717
PRODUCTIVITY           343
COMMUNICATION          323
MEDICAL                322
FINANCE                315
SPORTS                 315
PHOTOGRAPHY            311
PERSONALIZATION        304
BUSINESS               285
LIFESTYLE              285
HEALTH_AND_FITNESS     285
other                 2522
Name: Category, dtype: int64
```

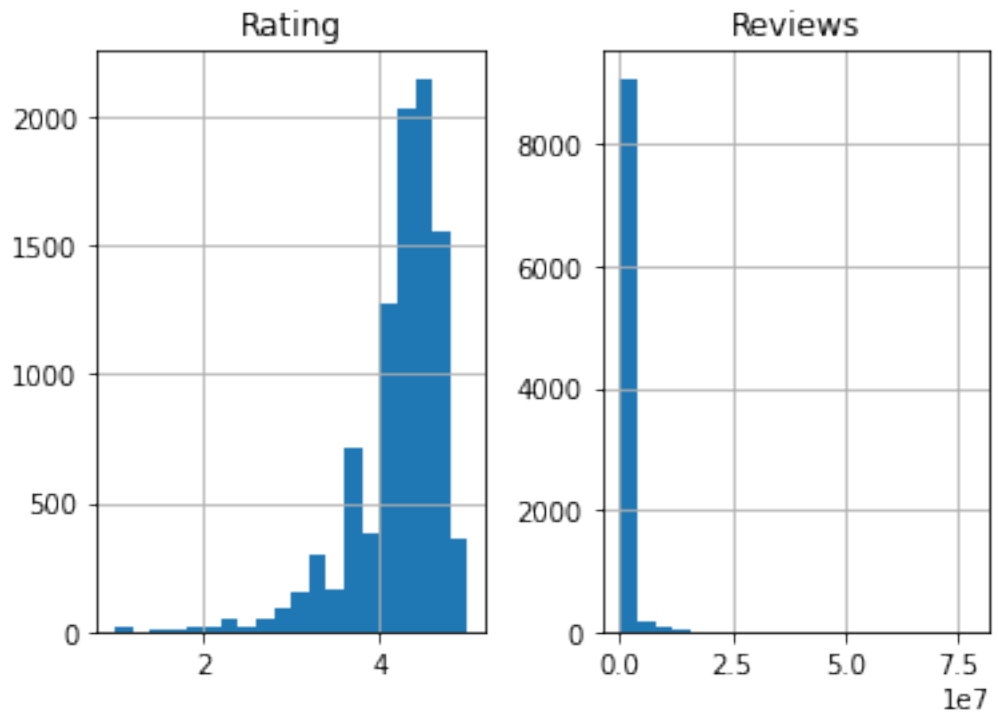
[29]:
```
#Check to see where .03 boundary is
#sum(categories[27:])/sum(categories)
sum(chart2) == sum(categories)
```

[29]:
```
True
```

[137]:
```
import matplotlib.pyplot as plt

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
labels = chart2.index
sizes = chart2

fig1, ax1 = plt.subplots()
ax1.pie(sizes, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title("Percentage of App Categories on the App Store")


plt.show()
```

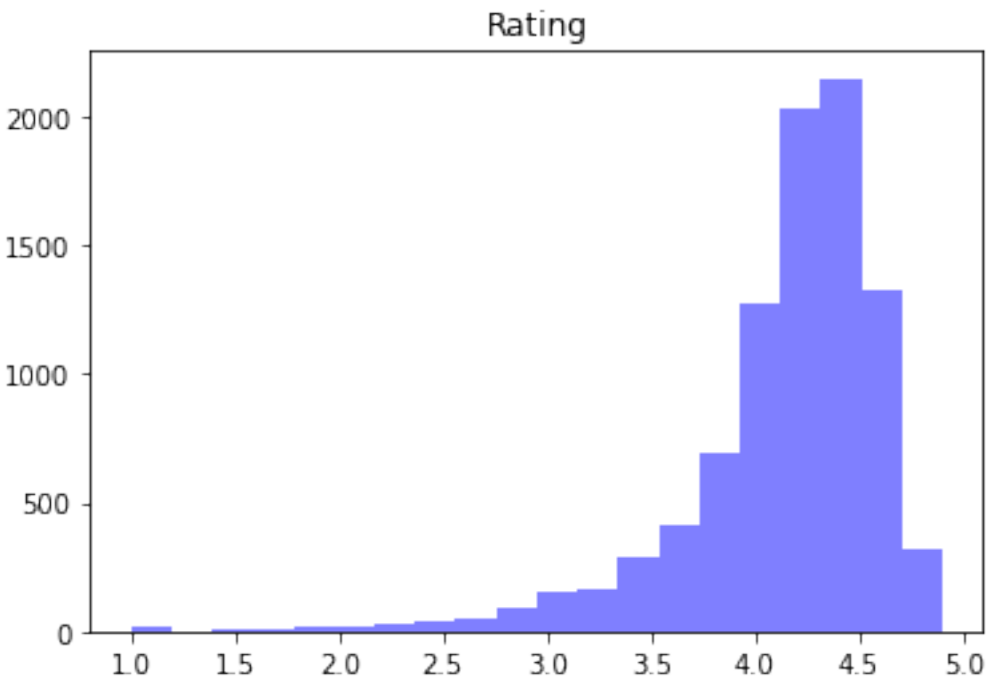


Question 3.3: Generating histograms of the Rating and Reviews across all apps, with 20 bins each. The histgrams should look like



Hint: Remember that histograms are used for numeric data. You might need to convert the values in one of the columns to a numeric type.

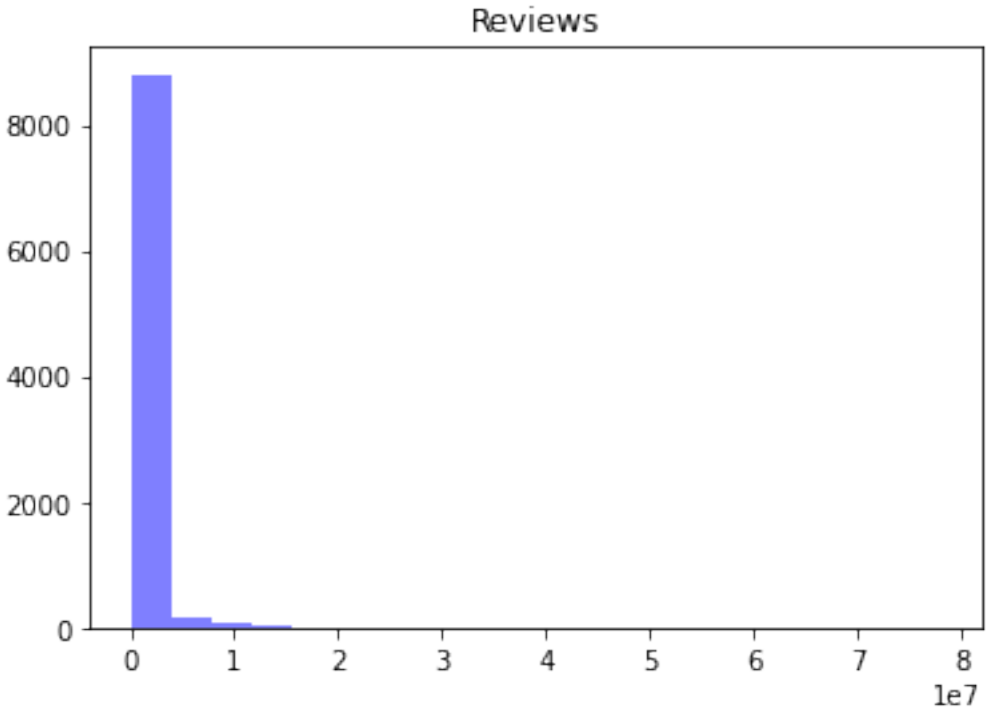
[141]:
```
import matplotlib.mlab as mlab

x = apps_df["Rating"]
num_bins = 20
n, bins, patches = plt.hist(x, num_bins, facecolor='blue', alpha=0.5)
plt.title("Rating")
plt.show()
```



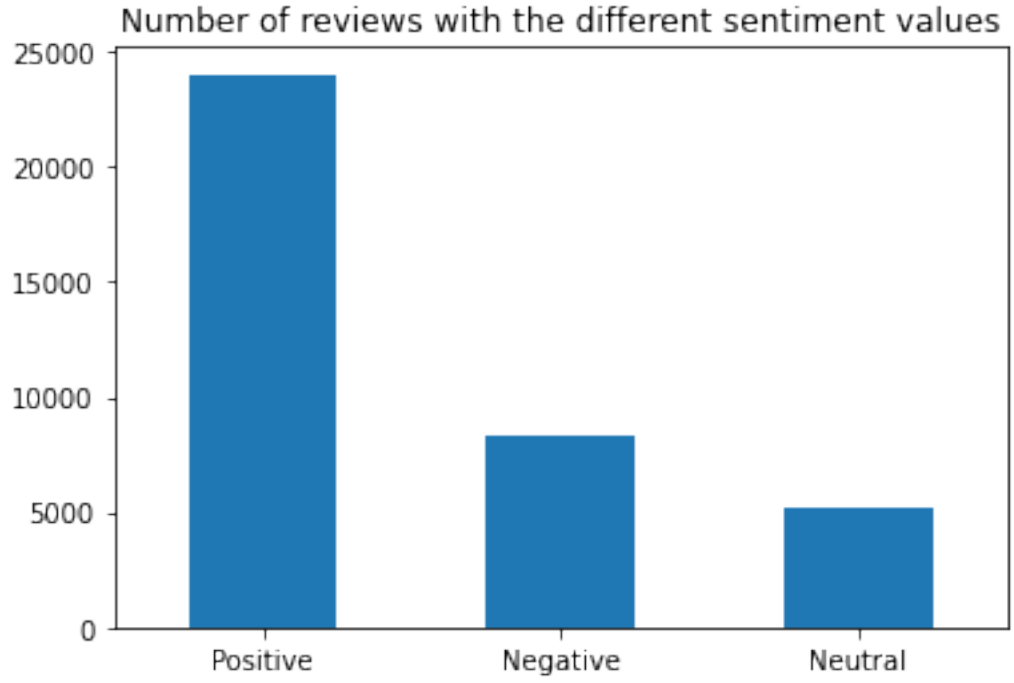
[139]:
```
type(apps_df["Reviews"].astype(int)[0]), type(apps_df["Reviews"][0])
```

[139]:
```
(numpy.int64, str)
```

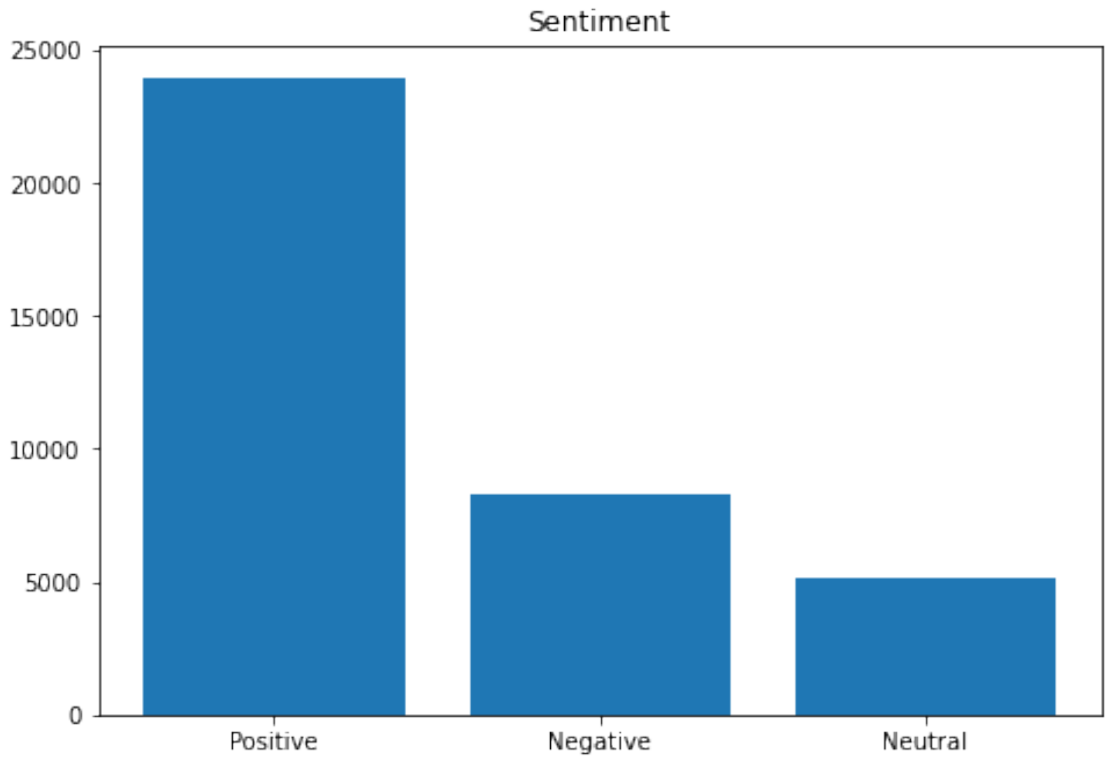
[143]:
```
x = apps_df["Reviews"].astype(int)
num_bins = 20
n, bins, patches = plt.hist(x, num_bins, facecolor='blue', alpha=0.5)
plt.title("Reviews")
plt.show()
```



Question 3.4: Plot a bar chart with the number of reviews that received the different `Sentiment` values. The sentiments chart should look similar to



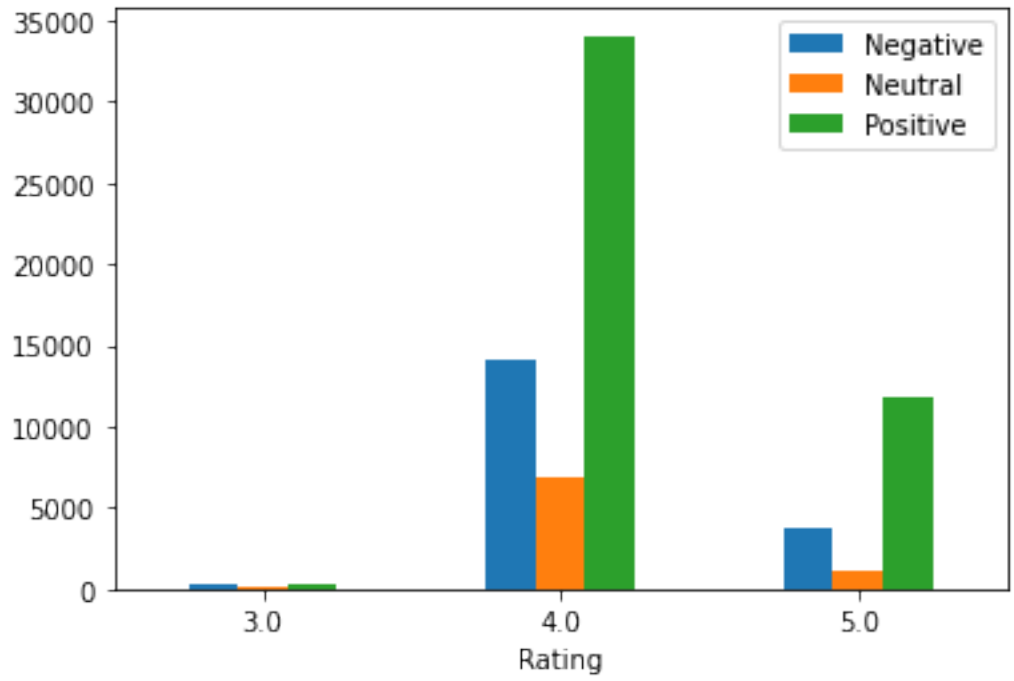
[144]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
sentiment_type = ["Positive", "Negative", "Neutral"]
amount = reviews_df["Sentiment"].value_counts()
ax.bar(sentiment_type, amount)
plt.title("Sentiment")
plt.show()
```



[145]:
```
reviews_df["Sentiment"].value_counts()
```

[145]:
```
Positive    23997
Negative     8270
Neutral      5158
Name: Sentiment, dtype: int64
```

Question 4.2: Group the Sentiment by rounded Rating, and produce a bar chart where you display the different sentiments grouped by rating. The chart should look like



Hint: You might find the `np.round`, `pd.groupby` and `df.unstack` functions helpful for this task.

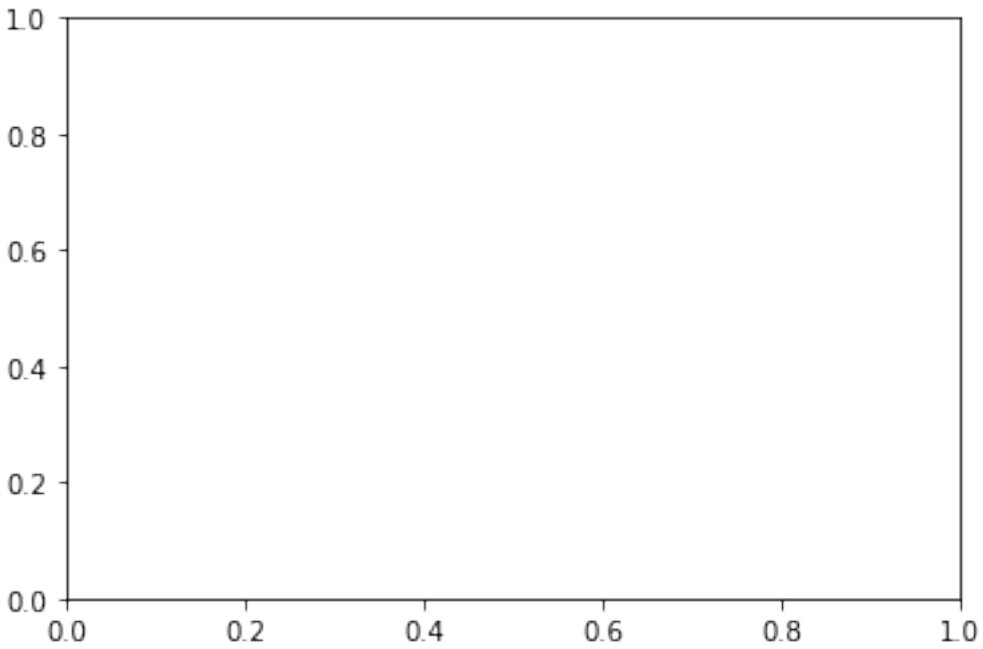
[213]:
```
merged_df_sentiment = merged_df[["Sentiment","Rating"]]
list_rounded = list(merged_df_sentiment["Rating"].round())
list_sentime = merged_df_sentiment["Sentiment"].tolist()
new_merged_df_sentiment = pd.DataFrame({"Sentiment":list_sentime, "Rating":
 ↪list_rounded})
new_merged_df_sentiment = new_merged_df_sentiment.dropna()
described_3 = new_merged_df_sentiment[new_merged_df_sentiment["Rating"] == 3.0].
 ↪groupby("Sentiment").count()
described_4 = new_merged_df_sentiment[new_merged_df_sentiment["Rating"] == 4.0].
 ↪groupby("Sentiment").count()
described_5 = new_merged_df_sentiment[new_merged_df_sentiment["Rating"] == 5.0].
 ↪groupby("Sentiment").count()
#merged_df_sentiment[merged_df_sentiment["Sentiment"] == 'Negative']
```

[214]:
```
labels = [3.0, 4.0, 5.0]
three = described_3
four =described_4
five = described_5

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
rects1 = ax.bar(x - width/3, pos_sent, width, label='Positive')
rects2 = ax.bar(x + width/3, neg_sent, width, label='Negative')
```

```
rects3 = ax.bar(x, neut_sent, width, label='Netural')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Total Number of Apps')
ax.set_title('Rating')
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)
ax.bar_label(rects3, padding=3)

fig.tight_layout()

plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-214-ce6b4871cebb> in <module>
      8
      9 fig, ax = plt.subplots()
---> 10 rects1 = ax.bar(x - width/3, pos_sent, width, label='Positive')
     11 rects2 = ax.bar(x + width/3, neg_sent, width, label='Negative')
     12 rects3 = ax.bar(x, neut_sent, width, label='Netural')

/opt/conda/lib/python3.8/site-packages/matplotlib/__init__.py in inner(ax, data␣
 ↪*args, **kwargs)
   1445     def inner(ax, *args, data=None, **kwargs):
   1446         if data is None:
-> 1447             return func(ax, *map(sanitize_sequence, args), **kwargs)
   1448
   1449         bound = new_sig.bind(ax, *args, **kwargs)

/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_axes.py in bar(self, x,␣
 ↪height, width, bottom, align, **kwargs)
   2428                 yerr = self._convert_dx(yerr, y0, y, self.convert_yunits)
   2429
-> 2430         x, height, width, y, linewidth = np.broadcast_arrays(
   2431             # Make args iterable too.
   2432             np.atleast_1d(x), height, width, y, linewidth)

<__array_function__ internals> in broadcast_arrays(*args, **kwargs)

/opt/conda/lib/python3.8/site-packages/numpy/lib/stride_tricks.py in␣
 ↪broadcast_arrays(subok, *args)
```

```
    536     args = [np.array(_m, copy=False, subok=subok) for _m in args]
    537
--> 538     shape = _broadcast_shape(*args)
    539
    540     if all(array.shape == shape for array in args):

/opt/conda/lib/python3.8/site-packages/numpy/lib/stride_tricks.py in␣
 ↪_broadcast_shape(*args)
    418     # use the old-iterator because np.nditer does not handle size 0␣
 ↪arrays
    419     # consistently
--> 420     b = np.broadcast(*args[:32])
    421     # unfortunately, it cannot handle 32 or more arguments directly
    422     for pos in range(32, len(args), 31):

ValueError: shape mismatch: objects cannot be broadcast to a single shape
```



To double-check your work, the cell below will rerun all of the autograder tests.

[114]:
```
grader.check_all()
```

[114]:
```
q1_1 passed!

q1_2 results:
```

```
Trying:
    assert apps_df.shape == (10841, 13)
Expecting nothing
**********************************************************************
Line 1, in q1_2 1
Failed example:
    assert apps_df.shape == (10841, 13)
Exception raised:
    Traceback (most recent call last):
      File "/opt/conda/lib/python3.8/doctest.py", line 1336, in __run
        exec(compile(example.source, filename, "single",
      File "<doctest q1_2 1[0]>", line 1, in <module>
        assert apps_df.shape == (10841, 13)
    AssertionError


q1_3 passed!

q1_4 passed!

q2_1 results:

Trying:
    assert reviews_df['Translated_Review'].isna().value_counts().shape[0] == 1
Expecting nothing
**********************************************************************
Line 2, in q2_1 0
Failed example:
    assert reviews_df['Translated_Review'].isna().value_counts().shape[0] == 1
Exception raised:
    Traceback (most recent call last):
      File "/opt/conda/lib/python3.8/doctest.py", line 1336, in __run
        exec(compile(example.source, filename, "single",
      File "<doctest q2_1 0[0]>", line 1, in <module>
        assert reviews_df['Translated_Review'].isna().value_counts().shape[0] ==
1
    AssertionError

Trying:
    assert reviews_df['Sentiment'].isna().value_counts().shape[0] == 1
Expecting nothing
**********************************************************************
Line 2, in q2_1 1
Failed example:
    assert reviews_df['Sentiment'].isna().value_counts().shape[0] == 1
Exception raised:
```

```
Traceback (most recent call last):
  File "/opt/conda/lib/python3.8/doctest.py", line 1336, in __run
    exec(compile(example.source, filename, "single",
  File "<doctest q2_1 1[0]>", line 1, in <module>
    assert reviews_df['Sentiment'].isna().value_counts().shape[0] == 1
AssertionError
```

```
q2_2 passed!
```

```
q4.1 passed!
```

0.1 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

[239]:
```
# Save your notebook first, then run this cell to export your submission.
grader.export()
```

```
<IPython.core.display.HTML object>
```