Examining the Effect of Review Count and Location on Stars Earned by Businesses on Yelp

Introduction

Yelp is company that publishes Yelp.com, a website dedicated to crowd-sourced reviewes about businesses. Yelp has many key features that help users consume products and services in the most effective way possible. The website and mobile application contains information regarding location (street, city, state/province, country, latitude, longitude), opening and closing hours, services provided, number of reviews, average reviews, business attributes (e.g. accepts credit cards), and taggable categories (e.g. resturant, health & medical, good for kids, etc.).

The following data exploration seeks to identify a statistical relationship between location and review count against number of stars (out of 5.0, with 5.0 being the best). More specifically, I examine business from Toronto and Montreal (nominal categorical variables) and look to determine whether the number of reviews (discrete variable) has any preictive power when it comes to number of stars given to a business (ordinal categorical variable).

Research Question: In Toronto, does the location and number of reviews of a business on Yelp influence the number of stars it receives from users?

Importing and Cleaning the Data

The following analysis utilizes four commonly known python packages: pandas (which is used to import and clean the data), matplotlib (which is used for formatting), pyplot (from matplotlib, which is used for plotting), and seaborn (which is used for plotting).

In [66]:

```python
# import packages for analysis
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
```

The Yelp data used in this project was sourced from Kaggle. According to their description, "This dataset is a subset of Yelp's businesses, reviews, and user data...put together for the Yelp Dataset Challenge." The data is stored as both CSV and JSON files, and can be found at the following link: https://www.kaggle.com/yelp-dataset/yelp-dataset/version/6.

In [67]:

```python
# reading in the data
yelp_business = pd.read_json("yelp_academic_dataset_business.json", lines = True)
yelp_business.head()
```

Out[67]:

	business_id	name	address	city	state	postal_code	latitude	longitude	stars	review_count	is_open	
0	f9NumwFMBDn751xgFiRbNA	The Range At Lake Norman	10913 Bailey Rd	Cornelius	NC	28031	35.462724	-80.852612	3.5	36	1	{'Busines
1	Yzvjg0SayhoZgCljUJRF9Q	Carlos Santo, NMD	8880 E Via Linda, Ste 107	Scottsdale	AZ	85258	33.569404	-111.890264	5.0	4	1	'By
2	XNoUzKckATkOD1hP6vghZg	Felinus	3554 Rue Notre-Dame O	Montreal	QC	H4C 1P4	45.479984	-73.580070	5.0	5	1	
3	6OAZjbxqM5ol29BuHsil3w	Nevada House of Hose	1015 Sharp Cir	North Las Vegas	NV	89030	36.219728	-115.127725	2.5	3	0	{'Busines
4	51M2Kk903DFYI6gnB5I6SQ	USE MY GUY SERVICES LLC	4827 E Downing Cir	Mesa	AZ	85205	33.428065	-111.726648	4.5	26	1	{'Busines

Because we want to only look at certain variables in the dataset, we filter out some of them in the following code chunks. Namely, we want to keep a business' latitude, longitude, city, and state (Toronto data only), review count, and number of stars.

In [68]:

```python
# removing unnecessary categories
# leaves city, state, latitude, longitude, stars, and review count only
yelp_business_cleaned = yelp_business[["city", "state", "latitude", "longitude", "stars", "review_count"]]
```

```
yelp_business_cleaned.head()
```

Out[68]:

	city	state	latitude	longitude	stars	review_count
0	Cornelius	NC	35.462724	-80.852612	3.5	36
1	Scottsdale	AZ	33.569404	-111.890264	5.0	4
2	Montreal	QC	45.479984	-73.580070	5.0	5
3	North Las Vegas	NV	36.219728	-115.127725	2.5	3
4	Mesa	AZ	33.428065	-111.726648	4.5	26

In [69]:

```
# taking Toronto data only
yelp_tor = yelp_business_cleaned["city"].isin(["Toronto"])
yelp_tor = yelp_business_cleaned[yelp_tor]

yelp_tor.head()
```

Out[69]:

	city	state	latitude	longitude	stars	review_count
9	Toronto	ON	43.624539	-79.529108	3.0	16
26	Toronto	ON	43.656542	-79.381308	4.0	9
29	Toronto	ON	43.603232	-79.538424	4.0	8
38	Toronto	ON	43.633291	-79.531768	3.0	13
52	Toronto	ON	43.727189	-79.293008	3.5	7

Summary Statistics

Now that we have our working dataframe of Toronto yelp data only, we can conduct some summary statistics on each numeric variable below.

In [70]:

```
# summary statistics for Toronto businesses
yelp_tor.describe()
```

Out[70]:

	latitude	longitude	stars	review_count

	latitude	longitude	stars	review_count
count	20366.000000	20366.000000	20366.000000	20366.000000
mean	43.679511	-79.394862	3.414367	28.651282
std	0.043918	0.060457	0.944043	65.240050
min	43.584846	-79.713930	1.000000	3.000000
25%	43.650579	-79.418920	3.000000	4.000000
50%	43.664456	-79.394136	3.500000	9.000000
75%	43.691593	-79.375347	4.000000	25.000000
max	43.881942	-79.019777	5.000000	2758.000000

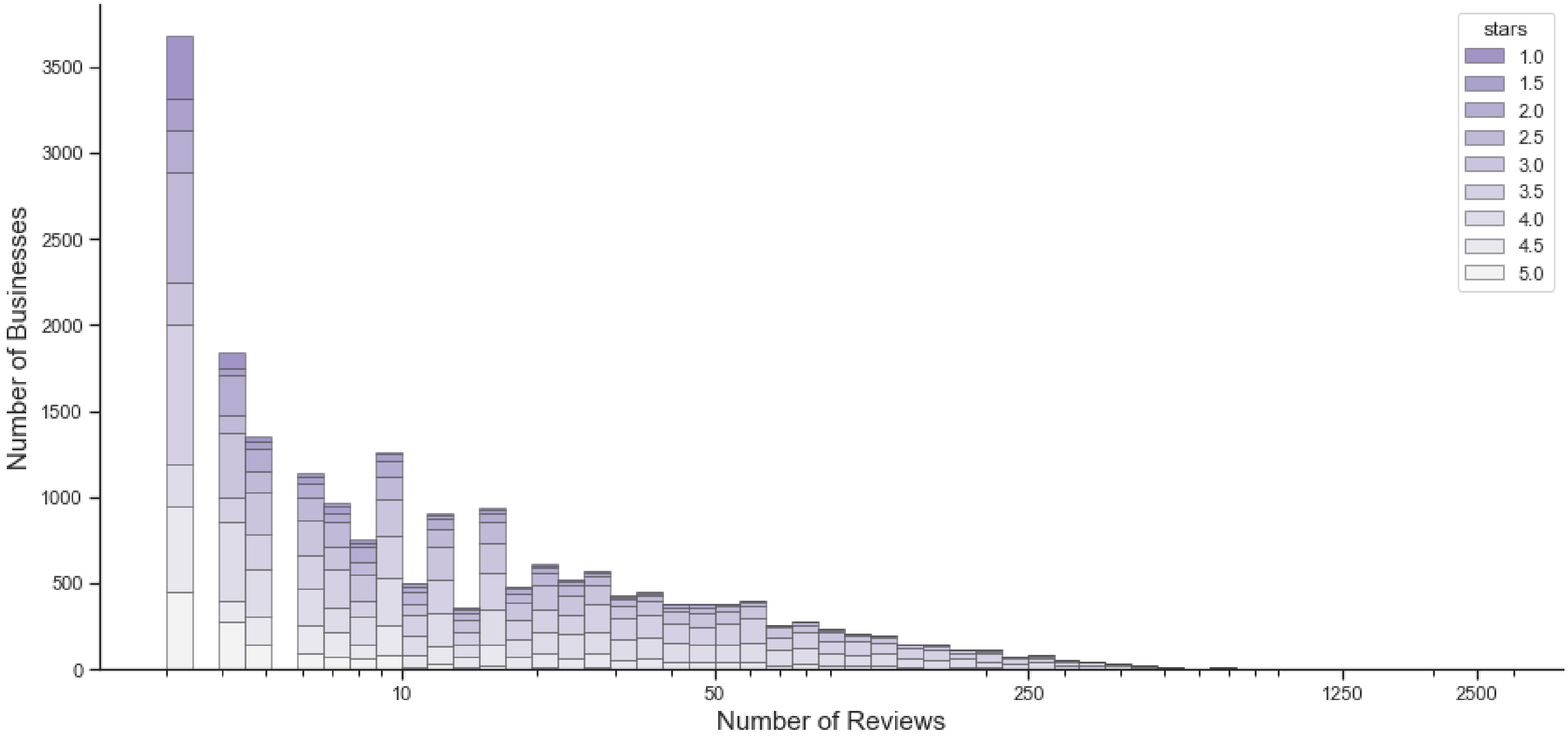
Although the summary statistics for latitude and longitude mean little, they do confirm that there are no outlier datapoints. That is, no typos, data entry mistakes, or businesses from outside of Toronto in the dataframe. Looking at stars, we see that the mean number of stars a Toronto business receives on Yelp is ~3.41, with a median of 3.5, both reasonable ratings for an "average" business. As for number of reviews, we see that the mean number of reviews is ~28.7, whereas the median number is only 9. Naturally, there are some businesses that have far more customers and active reviewers than others (e.g. an instagram friendly coffee shop in the Kensington Market), which drag the mean upwards. To see the full distribution of the number of reviews, we create a stacked bar chart below.

Distribution of the Number of Reviews (Stacked Bar Chart)

In [76]:

```
# distribution of the number of reviews per business, sorted by stars
sns.set_theme(style="ticks")

f, ax = plt.subplots(figsize=(15, 7))
sns.despine(f)

sns.histplot(
    yelp_tor,
    x="review_count", hue="stars",
    multiple="stack",
    palette="light:m_r",
    edgecolor=".3",
    linewidth=.5,
    # using a logarithmic scale for the x axis since that is how review count is likely distributed
    log_scale=True,
)
```

```python
ax.xaxis.set_major_formatter(mpl.ticker.ScalarFormatter())
ax.set_xlabel("Number of Reviews", fontsize = 15)
ax.set_ylabel("Number of Businesses", fontsize = 15)

# custom x-axis labels for clarity
ax.set_xticks([10, 50, 250, 1250, 2500]);
```



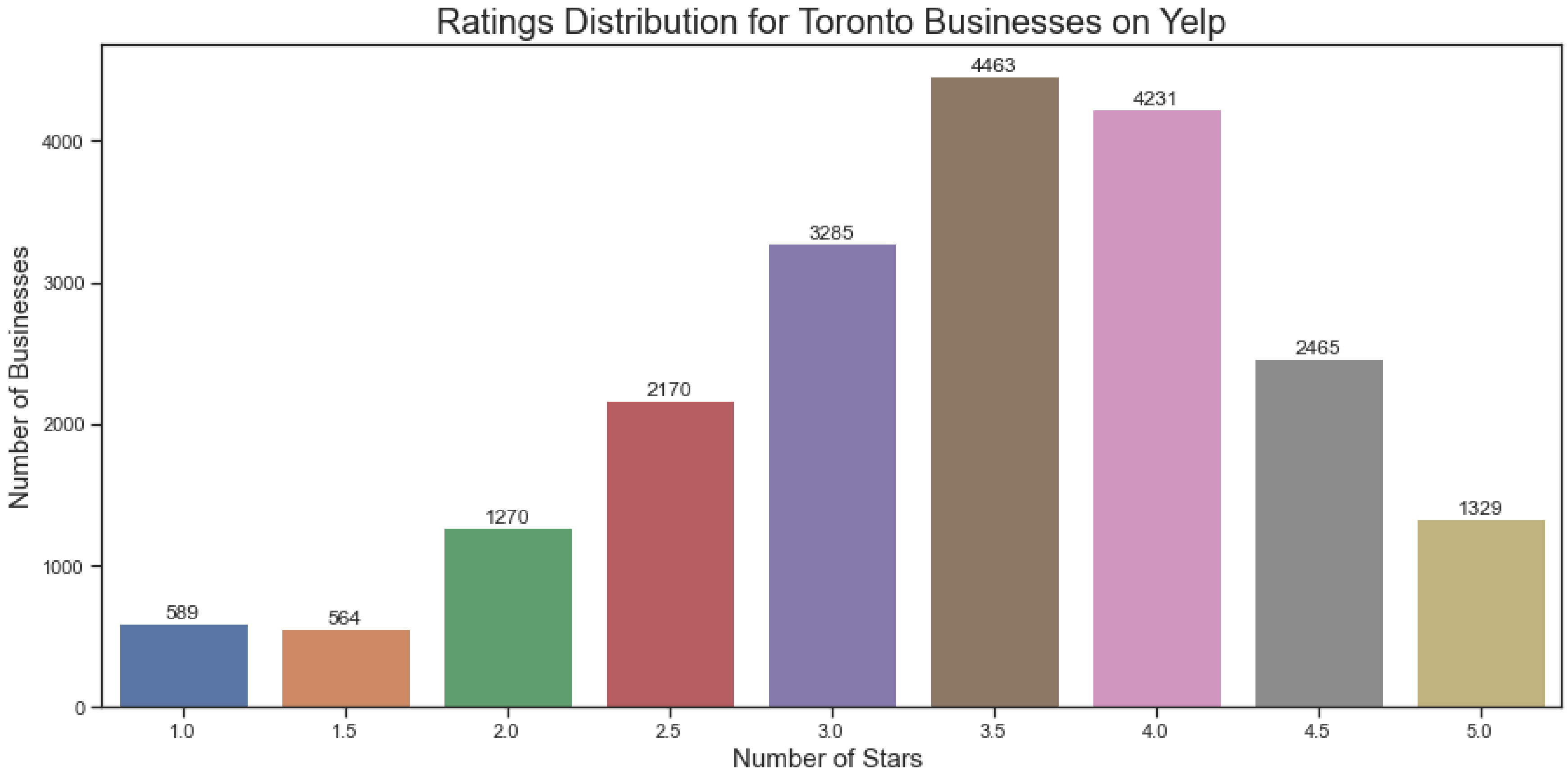
As expected, the majority of businesses in Toronto have been reviewed no more than 10 times on Yelp (left side of the plot). As we move along the right tail of the distribution, we notice that fewer and fewer businesses receive more and more review counts. The x-axis of the curve has been scaled logarithmically as a result. This curve is somewhat reminscent of a Pareto distribution, which reflects the Pareto or 80/20 principle. Intuitively, it is not unreasonable to suggest that ~80% of the businesses on Yelp receive ~20% of the reviews.

Distribution of the Number of Stars (Histogram)

Since we have thoroughly looked at the distributions of both x variables (location and number of reviews), it may be helpful to look at the distribution of star ratings for Toronto businesses. Since we have even bin sizes and no outliers (stars range from 1-5 in increments of 0.5), we can create a histogram below.

In [72]:

```
# Ratings distribution of Toronto Businesses on Yelp
tor = yelp_tor['stars'].value_counts()
tor = tor.sort_index()

#plot
plt.figure(figsize=(15,7))
ax= sns.barplot(x = tor.index, y = tor.values)
plt.title("Ratings Distribution for Toronto Businesses on Yelp", fontsize = 20)
plt.ylabel('Number of Businesses', fontsize=15)
plt.xlabel('Number of Stars ', fontsize=15)

#adding the text labels
rects = ax.patches
labels = tor.values
for rect, label in zip(rects, labels):
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2, height + 5, label, ha='center', va='bottom')

plt.show()
```



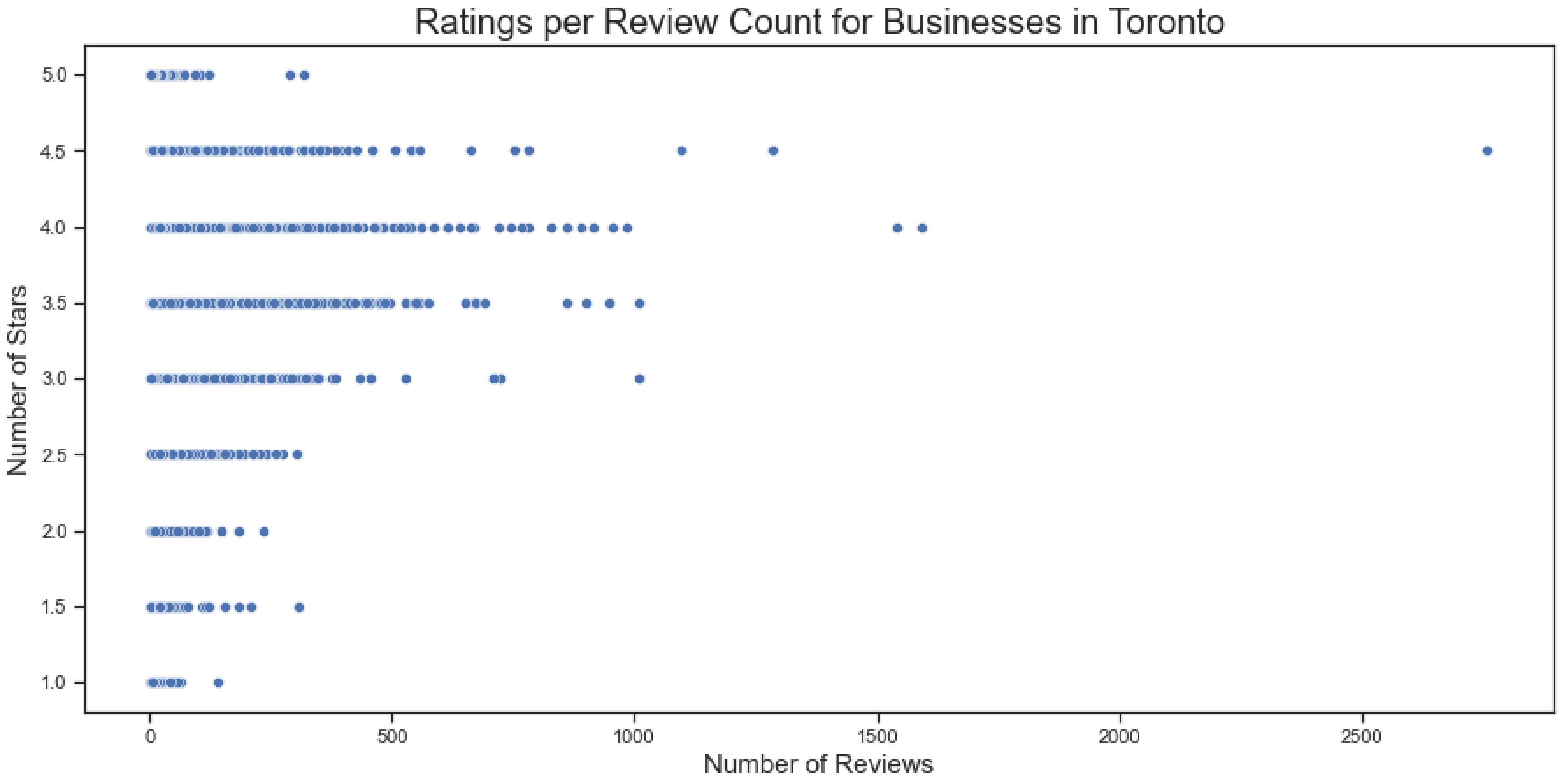
The distribution of the histogram reflects what the summary statistics had already alluded to previously. The most common rating for a business in Toronto is 3.5 stars, followed by 4 stars and 3 stars.

Number of Reviews and Number of Stars (Scatterplot)

Having looked at each variable independent of one another, we can examine whether any relationship between the explanatory variables (location and number of reviews) and the response variable (number of stars) exists. First, we create a scatter plot which plots every business in the Yelp dataset based on the number of reviews and the number of stars it has on Yelp.

In [73]:

```
# Star Ratings Based on Number of Reviews for Toronto Businesses on Yelp

#plot
plt.figure(figsize=(15,7))
ax= sns.scatterplot(x = yelp_tor.review_count, y = yelp_tor.stars)
plt.title("Ratings per Review Count for Businesses in Toronto", fontsize = 20)
plt.ylabel('Number of Stars', fontsize=15)
plt.xlabel('Number of Reviews', fontsize=15)

#adding the text labels
rects = ax.patches
labels = tor.values
for rect, label in zip(rects, labels):
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2, height + 5, label, ha='center', va='bottom')

plt.show()
```



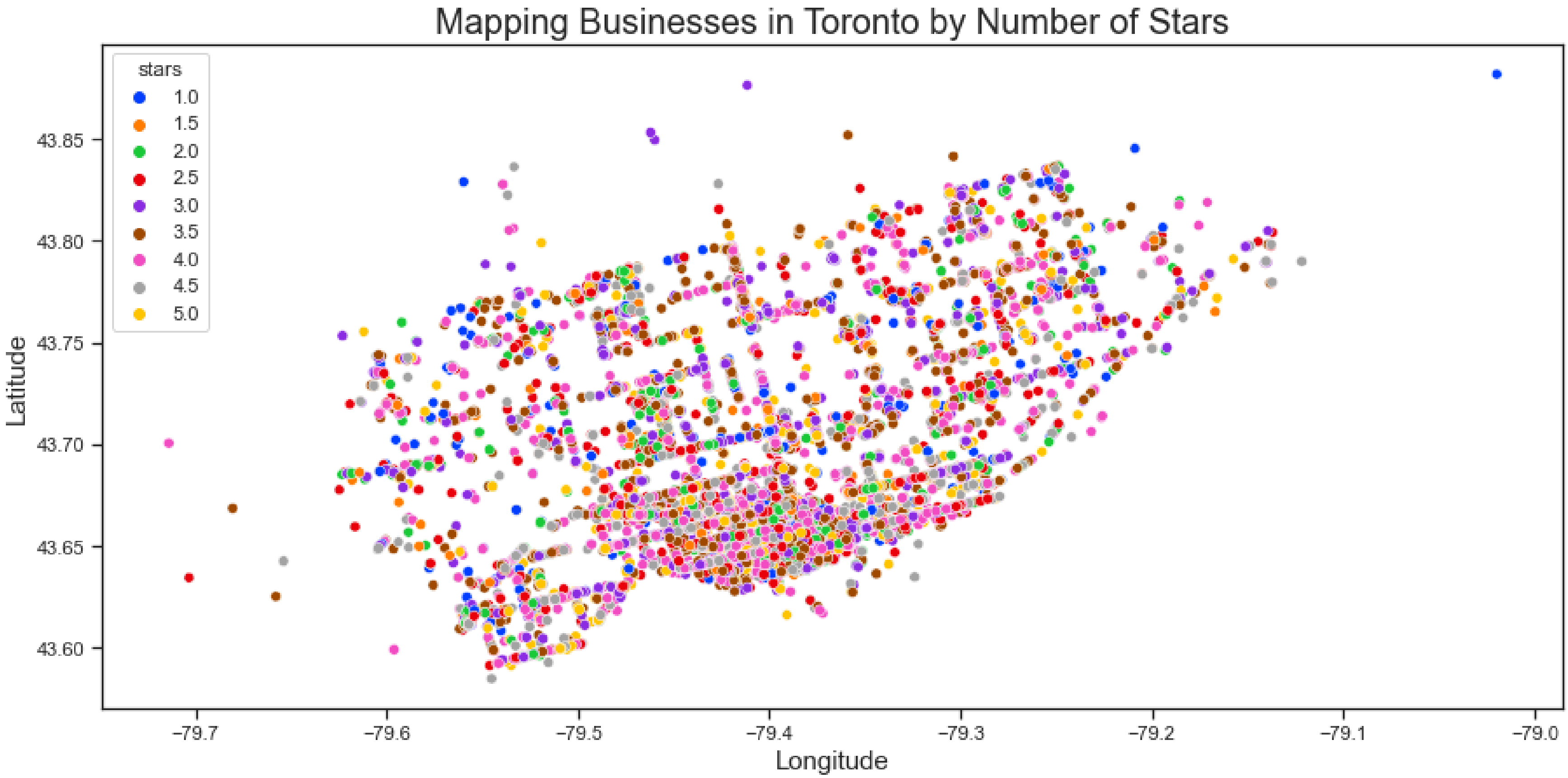
Based on the plot above, there are a few observations that can be made. Firstly, the businesses that have the least number of reviews have either the highest rating (5 stars), or the lowest three ratings (1, 1.5, and 2 stars). There may be several reasons for this. One is that it may be difficult for a business with many reviews to maintain a sky high rating. On the flip side, customers may be dissuaded from visiting a business with a super low rating already, and therefore be less likely to review the business.

Location and Number of Stars (Map Scatterplot)

In order to assess whether location plays any role in business ratings, we generate a "map" of toronto by creating a scatterplot of businesses on Yelp. Since latitude as an angle measures a locations' North-South position on Earth, and longitude as an angle measures a locations' East-West position on Earth, these variables are plotted as the y and x variables respectively below. As for the number of stars a business has on Yelp, ratings have been color coordinated according to the legend on the left.

In [74]:

```
# Star Ratings Based on Number of Reviews for Toronto Businesses on Yelp

#plot
palette = sns.color_palette("bright", 9)
plt.figure(figsize=(15,7))
ax = sns.scatterplot(x = yelp_tor.longitude, y = yelp_tor.latitude, palette = palette, hue = yelp_tor.stars)
plt.title("Mapping Businesses in Toronto by Number of Stars", fontsize = 20)
plt.ylabel('Latitude', fontsize=15)
plt.xlabel('Longitude', fontsize=15)
new_title = "Number of Stars"

plt.show()
```



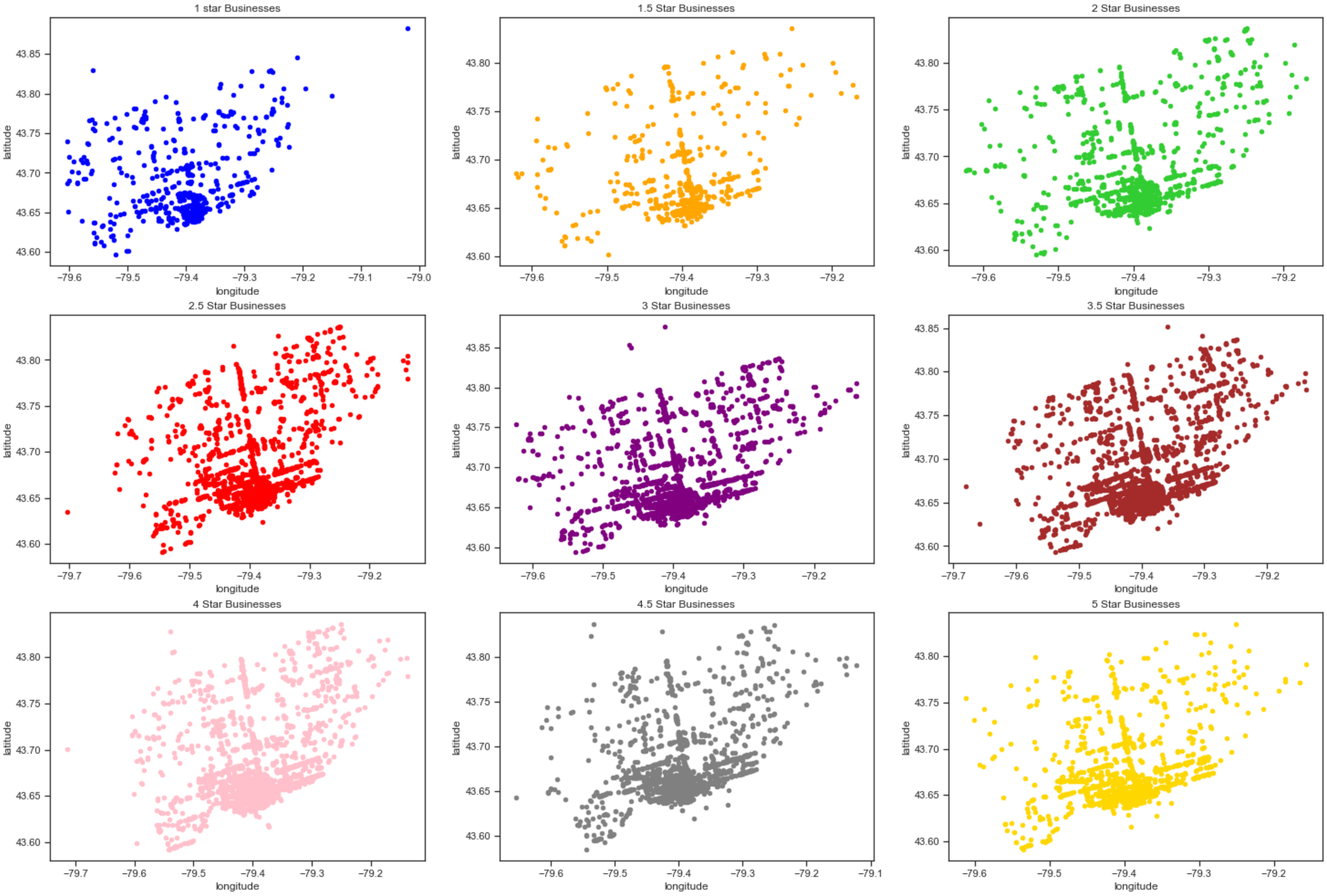
Having plotted all of the businesses from the dataset above, we see a general shape of Toronto forming, with the curvature of the waterfront at the bottom, and several straighlines of points resembling the city's gridlike nature. However, since there are so many datapoints, it might be easier to generate 9 different maps, one for each rating (from 1 star to 5 star). In the 3x3 grid below, ratings have been sorted from 1 star in the top left to 5 star in the bottom right.

In [75]:

```
plots, [[ax1, ax2, ax3], [ax4, ax5, ax6], [ax7, ax8, ax9]] = plt.subplots(3, 3, figsize = (25, 17))


yelp_tor1 = yelp_tor["stars"].isin(["1.0"])
yelp_tor1 = yelp_tor[yelp_tor1]
yelp_tor1.plot(kind = "scatter", x = "longitude", y = "latitude", color = "blue", ax = ax1)
ax1.set_title("1 star Businesses")

yelp_tor15= yelp_tor["stars"].isin(["1.5"])
yelp_tor15 = yelp_tor[yelp_tor15]
yelp_tor15.plot(kind = "scatter", x = "longitude", y = "latitude", color = "orange", ax = ax2)
```

```
ax2.set_title("1.5 Star Businesses")

yelp_tor2= yelp_tor["stars"].isin(["2.0"])
yelp_tor2 = yelp_tor[yelp_tor2]
yelp_tor2.plot(kind = "scatter", x = "longitude", y = "latitude", color = "limegreen", ax = ax3)
ax3.set_title("2 Star Businesses")

yelp_tor25= yelp_tor["stars"].isin(["2.5"])
yelp_tor25 = yelp_tor[yelp_tor25]
yelp_tor25.plot(kind = "scatter", x = "longitude", y = "latitude", color = "red",  ax = ax4)
ax4.set_title("2.5 Star Businesses")

yelp_tor3= yelp_tor["stars"].isin(["3.0"])
yelp_tor3 = yelp_tor[yelp_tor3]
yelp_tor3.plot(kind = "scatter", x = "longitude", y = "latitude", color = "purple", ax = ax5)
ax5.set_title("3 Star Businesses")

yelp_tor35= yelp_tor["stars"].isin(["3.5"])
yelp_tor35 = yelp_tor[yelp_tor35]
yelp_tor35.plot(kind = "scatter", x = "longitude", y = "latitude", color = "brown", ax = ax6)
ax6.set_title("3.5 Star Businesses")

yelp_tor4= yelp_tor["stars"].isin(["4.0"])
yelp_tor4 = yelp_tor[yelp_tor4]
yelp_tor4.plot(kind = "scatter", x = "longitude", y = "latitude", color = "pink", ax = ax7)
ax7.set_title("4 Star Businesses")

yelp_tor45= yelp_tor["stars"].isin(["4.5"])
yelp_tor45 = yelp_tor[yelp_tor45]
yelp_tor45.plot(kind = "scatter", x = "longitude", y = "latitude", color = "grey", ax = ax8)
ax8.set_title("4.5 Star Businesses")

yelp_tor5= yelp_tor["stars"].isin(["5.0"])
yelp_tor5 = yelp_tor[yelp_tor5]
yelp_tor5.plot(kind = "scatter", x = "longitude", y = "latitude", color = "gold", ax = ax9)
ax9.set_title("5 Star Businesses")
```

```
Out[75]: Text(0.5, 1.0, '5 Star Businesses')
```



The 3x3 grid of maps above showcases the density of businesses in the downtown area of Toronto (the cluster around -79.4 degrees of longitude and 43.65 degrees of latitude). One observation that can be made is that there are very few 1 star rated businesses in the eastern half of Toronto (blue scatter). Another observation is that it seems as though many lower rated businesses exist in the western half of Toronto (see the points between -79.7 and -79.6 degrees of longitude for every scatter).

Conclusion

In the following notebook, data from Yelp was taken from Kaggle to answer the following research question: In Toronto, does the location and number of reviews of a business on Yelp influence the number of stars it receives from users? This project focused on importing, cleaning, sorting, summarizing, plotting, and analyzing data for business reviewed on Yelp in Toronto. While there are some interesting observations to be made, this project is rudimentary in nature, and is perhaps best read as a template to clean and present data from the web.