Analysis of Yelp Business Intelligence Data

We will analyze a subset of Yelp's business, reviews and user data. This dataset comes to us from Kaggle although we have taken steps to pull this data into a publis s3 bucket: `s3://cis9760-yelpdataset/yelp-light/*business.json`

Installation and Initial Setup

Begin by installing the necessary libraries that you may need to conduct your analysis. At the very least, you must install `pandas` and `matplotlib`

In [1]:
```
%%info
```

Current session configs: `{'conf': {'spark.pyspark.python': 'python3', 'spark.pyspark.virtualenv.enabled': 'true', 'spark.pyspark.virtualenv.type': 'native', 'spark.pyspark.virtualenv.bin.path': '/usr/bin/virtualenv'}, 'kind': 'pyspark'}`

No active sessions.

In [2]:
```
# Installing all libraries I might need
sc.install_pypi_package("pandas==1.0.3")
sc.install_pypi_package("matplotlib==3.2.1")
sc.install_pypi_package("scipy==1.7.1")
sc.install_pypi_package("seaborn==0.11.2")
```

▸ Spark Job Progress

Starting Spark application

ID	YARN Application ID	Kind	State	
10	application_1651323976290_0011	pyspark	idle	Link (http://ip-17 22.ec2.internal:20888/proxy/application_165132397629

SparkSession available as 'spark'.

```
Collecting pandas==1.0.3
  Using cached https://files.pythonhosted.org/packages/4a/6a/94b219b8ea0f2d580169e85ed1edc0163743f55aaeca8a44c2e8fc1e344e/pandas-1.0.3-cp37-cp37m-manylinux1_x86_64.whl (https://files.pythonhosted.org/packages/4a/6a/94b219b8ea0f2d580169e85ed1edc0163743f55aaeca8a44c2e8fc1e344e/pandas-1.0.3-cp37-cp37m-manylinux1_x86_64.whl)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/site-packages (from pandas==1.0.3)
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib64/python3.7/site-packages (from pandas==1.0.3)
Collecting python-dateutil>=2.6.1 (from pandas==1.0.3)
  Using cached https://files.pythonhosted.org/packages/36/7a/87837f39d0296e723bb9b62bbb257d0355c7f6128853c78955f57342a56d/python_dateutil-2.8.2-py2.py3-none-any.whl (https://files.pythonhosted.org/packages/36/7a/87837f39d0296e723bb9b62bbb257d0355c7f6128853c78955f57342a56d/python_dateutil-2.8.2-py2.py3-none-any.whl)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/site-packages (from python-dateutil>=2.6.1->pandas==1.0.3)
Installing collected packages: python-dateutil, pandas
Successfully installed pandas-1.0.3 python-dateutil-2.8.2

Collecting matplotlib==3.2.1
  Using cached https://files.pythonhosted.org/packages/b2/c2/71fcf957710f3ba1f09088b35776a799ba7dd95f7c2b195ec800933b276b/matplotlib-3.2.1-cp37-cp37m-manylinux1_x86_64.whl (https://files.pythonhosted.org/packages/b2/c2/71fcf957710f3ba1f09088b35776a799ba7dd95f7c2b195ec800933b276b/matplotlib-3.2.1-cp37-cp37m-manylinux1_x86_64.whl)
Requirement already satisfied: python-dateutil>=2.1 in /mnt/tmp/1651341799496-0/lib/python3.7/site-packages (from matplotlib==3.2.1)
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 (from matplotlib==3.2.1)
  Using cached https://files.pythonhosted.org/packages/d9/41/d9cfb4410589805cd787f8a82cddd13142d9bf7449d12adf2d05a4a7d633/pyparsing-3.0.8-py3-none-any.whl (https://files.pythonhosted.org/packages/d9/41/d9cfb4410589805cd787f8a82cddd13142d9bf7449d12adf2d05a4a7d633/pyparsing-3.0.8-py3-none-any.whl)
Collecting cycler>=0.10 (from matplotlib==3.2.1)
  Using cached https://files.pythonhosted.org/packages/5c/f9/695d6bedebd747e5eb0fe8fad57b72fdf25411273a39791cde838d5a8f51/cycler-0.11.0-py3-none-any.whl (https://files.pythonhosted.org/packages/5c/f9/695d6bedebd747e5eb0fe8fad57b72fdf2541
```

```
1273a39791cde838d5a8f51/cycler-0.11.0-py3-none-any.whl)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib64/python3.7/site-p
ackages (from matplotlib==3.2.1)
Collecting kiwisolver>=1.0.1 (from matplotlib==3.2.1)
  Using cached https://files.pythonhosted.org/packages/51/50/9a9a94afa26c50fc5d
9127272737806990aa698c7a1c220b8e5075e70304/kiwisolver-1.4.2-cp37-cp37m-manylinu
x_2_5_x86_64.manylinux1_x86_64.whl (https://files.pythonhosted.org/packages/51/
50/9a9a94afa26c50fc5d9127272737806990aa698c7a1c220b8e5075e70304/kiwisolver-1.4.
2-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.whl)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/site-packag
es (from python-dateutil>=2.1->matplotlib==3.2.1)
Collecting typing-extensions; python_version < "3.8" (from kiwisolver>=1.0.1->m
atplotlib==3.2.1)
  Using cached https://files.pythonhosted.org/packages/75/e1/932e06004039dd670c
9d5e1df0cd606bf46e29a28e65d5bb28e894ea29c9/typing_extensions-4.2.0-py3-none-an
y.whl (https://files.pythonhosted.org/packages/75/e1/932e06004039dd670c9d5e1df0
cd606bf46e29a28e65d5bb28e894ea29c9/typing_extensions-4.2.0-py3-none-any.whl)
Installing collected packages: pyparsing, cycler, typing-extensions, kiwisolve
r, matplotlib
Successfully installed cycler-0.11.0 kiwisolver-1.4.2 matplotlib-3.2.1 pyparsin
g-3.0.8 typing-extensions-4.2.0

Collecting scipy==1.7.1
  Using cached https://files.pythonhosted.org/packages/b5/6b/8bc0b61ebf824f8c39
79a31368bbe38dd247590049a994ab0ed077cb56dc/scipy-1.7.1-cp37-cp37m-manylinux_2_5
_x86_64.manylinux1_x86_64.whl (https://files.pythonhosted.org/packages/b5/6b/8b
c0b61ebf824f8c3979a31368bbe38dd247590049a994ab0ed077cb56dc/scipy-1.7.1-cp37-cp3
7m-manylinux_2_5_x86_64.manylinux1_x86_64.whl)
Requirement already satisfied: numpy<1.23.0,>=1.16.5 in /usr/local/lib64/python
3.7/site-packages (from scipy==1.7.1)
Installing collected packages: scipy
Successfully installed scipy-1.7.1

Collecting seaborn==0.11.2
  Using cached https://files.pythonhosted.org/packages/10/5b/0479d7d845b5ba410c
a702ffcd7f2cd95a14a4dfff1fde2637802b258b9b/seaborn-0.11.2-py3-none-any.whl (htt
ps://files.pythonhosted.org/packages/10/5b/0479d7d845b5ba410ca702ffcd7f2cd95a14
a4dfff1fde2637802b258b9b/seaborn-0.11.2-py3-none-any.whl)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib64/python3.7/site-p
ackages (from seaborn==0.11.2)
Requirement already satisfied: scipy>=1.0 in /mnt/tmp/1651341799496-0/lib/pytho
n3.7/site-packages (from seaborn==0.11.2)
Requirement already satisfied: matplotlib>=2.2 in /mnt/tmp/1651341799496-0/lib/
python3.7/site-packages (from seaborn==0.11.2)
Requirement already satisfied: pandas>=0.23 in /mnt/tmp/1651341799496-0/lib/pyt
hon3.7/site-packages (from seaborn==0.11.2)
Requirement already satisfied: python-dateutil>=2.1 in /mnt/tmp/1651341799496-
0/lib/python3.7/site-packages (from matplotlib>=2.2->seaborn==0.11.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /mn
t/tmp/1651341799496-0/lib/python3.7/site-packages (from matplotlib>=2.2->seabor
n==0.11.2)
Requirement already satisfied: cycler>=0.10 in /mnt/tmp/1651341799496-0/lib/pyt
hon3.7/site-packages (from matplotlib>=2.2->seaborn==0.11.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /mnt/tmp/1651341799496-0/li
b/python3.7/site-packages (from matplotlib>=2.2->seaborn==0.11.2)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/site-pa
ckages (from pandas>=0.23->seaborn==0.11.2)
```

```
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/site-packag
es (from python-dateutil>=2.1->matplotlib>=2.2->seaborn==0.11.2)
Requirement already satisfied: typing-extensions; python_version < "3.8" in /mn
t/tmp/1651341799496-0/lib/python3.7/site-packages (from kiwisolver>=1.0.1->matp
lotlib>=2.2->seaborn==0.11.2)
Installing collected packages: seaborn
Successfully installed seaborn-0.11.2
```

Importing

Now, import the installed packages from the previous block below.

In [3]:
```
# Importing libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
```

Loading Data

We are finally ready to load data. Using spark load the data from S3 into a dataframe object that we can manipulate further down in our analysis.

In [4]:
```
# Importing files from S3
business = spark.read.json('s3://xinyujiang-project02/yelp_academic_dataset_busir
```

▸ Spark Job Progress

In [5]:
```
# Previewing business dataframe
business.show(5)
```

▸ Spark Job Progress

```
+--------------------+--------------------+--------------------+--------------
-----+-------------+--------------------+-------+----------+-----------+------
-------------+-----------+------------+-----+-----+
|             address|          attributes|         business_id|          categ
ories|         city|               hours|is_open|  latitude|   longitude|
name|postal_code|review_count|stars|state|
+--------------------+--------------------+--------------------+--------------
-----+-------------+--------------------+-------+----------+-----------+------
-------------+-----------+------------+-----+-----+
|1616 Chapala St, ...|[,,,,,,,,,,, True...|Pns2l4eNsfO8kk83d...|Doctors, Tradit
io...|Santa Barbara|                null|      0|34.4266787|-119.7111968|Abby R
appoport, L...|      93101|           7|  5.0|   CA|
|87 Grasso Plaza S...|[,,,,,,,,, True,,...|mpf3x-BjTdTEA3yCZ...|Shipping Center
s,...|       Affton|[8:0-18:30, 0:0-0...|      1| 38.551126|  -90.335695|
The UPS Store|      63123|          15|  3.0|   MO|
|5255 E Broadway Blvd|[,,,,,,, True,, T...|tUFrWirKiKi_TAnsV...|Department Stor
es...|       Tucson|[8:0-23:0, 8:0-22...|      0| 32.223236| -110.880452|
Target|      85711|          22|  3.5|   AZ|
|         935 Race St|[,, u'none',,,,, ...|MTSW4McQd7CbVtyjq...|Restaurants, Fo
od...| Philadelphia|[7:0-21:0, 7:0-20...|      1|39.9555052| -75.1555641|  St H
onore Pastries|      19107|          80|  4.0|   PA|
|       101 Walnut St|[,,,,,,, True,, T...|mWMc6_wTdE0EUBKIG...|Brewpubs, Brewe
ri...|   Green Lane|[12:0-22:0,, 12:0...|      1|40.3381827| -75.4716585|Perkio
men Valley ...|      18054|          13|  4.5|   PA|
+--------------------+--------------------+--------------------+--------------
-----+-------------+--------------------+-------+----------+-----------+------
-------------+-----------+------------+-----+-----+
only showing top 5 rows
```

Overview of Data

Display the number of rows and columns in our dataset.

In [6]:
```
columns = len(business.columns)
rows = business.count()

print('Number of columns in Business table: ',columns)
print('Number of rows in Business table: ',rows)
```

▸ Spark Job Progress

```
Number of columns in Business table:  14
Number of rows in Business table:  150346
```

Display the DataFrame schema below.

In [7]:
```
business.printSchema()
```

```
root
 |-- address: string (nullable = true)
 |-- attributes: struct (nullable = true)
 |    |-- AcceptsInsurance: string (nullable = true)
 |    |-- AgesAllowed: string (nullable = true)
 |    |-- Alcohol: string (nullable = true)
 |    |-- Ambience: string (nullable = true)
 |    |-- BYOB: string (nullable = true)
 |    |-- BYOBCorkage: string (nullable = true)
 |    |-- BestNights: string (nullable = true)
 |    |-- BikeParking: string (nullable = true)
 |    |-- BusinessAcceptsBitcoin: string (nullable = true)
 |    |-- BusinessAcceptsCreditCards: string (nullable = true)
 |    |-- BusinessParking: string (nullable = true)
 |    |-- ByAppointmentOnly: string (nullable = true)
 |    |-- Caters: string (nullable = true)
 |    |-- CoatCheck: string (nullable = true)
 |    |-- Corkage: string (nullable = true)
 |    |-- DietaryRestrictions: string (nullable = true)
 |    |-- DogsAllowed: string (nullable = true)
 |    |-- DriveThru: string (nullable = true)
 |    |-- GoodForDancing: string (nullable = true)
 |    |-- GoodForKids: string (nullable = true)
 |    |-- GoodForMeal: string (nullable = true)
 |    |-- HairSpecializesIn: string (nullable = true)
 |    |-- HappyHour: string (nullable = true)
 |    |-- HasTV: string (nullable = true)
 |    |-- Music: string (nullable = true)
 |    |-- NoiseLevel: string (nullable = true)
 |    |-- Open24Hours: string (nullable = true)
 |    |-- OutdoorSeating: string (nullable = true)
 |    |-- RestaurantsAttire: string (nullable = true)
 |    |-- RestaurantsCounterService: string (nullable = true)
 |    |-- RestaurantsDelivery: string (nullable = true)
 |    |-- RestaurantsGoodForGroups: string (nullable = true)
 |    |-- RestaurantsPriceRange2: string (nullable = true)
 |    |-- RestaurantsReservations: string (nullable = true)
 |    |-- RestaurantsTableService: string (nullable = true)
 |    |-- RestaurantsTakeOut: string (nullable = true)
 |    |-- Smoking: string (nullable = true)
 |    |-- WheelchairAccessible: string (nullable = true)
 |    |-- WiFi: string (nullable = true)
 |-- business_id: string (nullable = true)
 |-- categories: string (nullable = true)
 |-- city: string (nullable = true)
 |-- hours: struct (nullable = true)
 |    |-- Friday: string (nullable = true)
 |    |-- Monday: string (nullable = true)
 |    |-- Saturday: string (nullable = true)
 |    |-- Sunday: string (nullable = true)
 |    |-- Thursday: string (nullable = true)
 |    |-- Tuesday: string (nullable = true)
 |    |-- Wednesday: string (nullable = true)
 |-- is_open: long (nullable = true)
```

```
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
 |-- name: string (nullable = true)
 |-- postal_code: string (nullable = true)
 |-- review_count: long (nullable = true)
 |-- stars: double (nullable = true)
 |-- state: string (nullable = true)
```

Display the first 5 rows with the following columns:

- business_id
- name
- city
- state
- categories

In [8]:
```
business.createOrReplaceTempView("business")
spark.sql("""
SELECT
    business_id,
    name,
    city,
    state,
    categories
FROM business
""").show(5, truncate=True)
```

▸ Spark Job Progress

```
+--------------------+--------------------+------------+-----+--------------------
----+
|         business_id|                name|        city|state|          catego
ries|
+--------------------+--------------------+------------+-----+--------------------
----+
|Pns2l4eNsfO8kk83d...|Abby Rappoport, L...|Santa Barbara|   CA|Doctors, Traditi
o...|
|mpf3x-BjTdTEA3yCZ...|       The UPS Store|       Affton|   MO|Shipping Center
s,...|
|tUFrWirKiKi_TAnsV...|              Target|       Tucson|   AZ|Department Store
s...|
|MTSW4McQd7CbVtyjq...|  St Honore Pastries| Philadelphia|   PA|Restaurants, Foo
d...|
|mWMc6_wTdE0EUBKIG...|Perkiomen Valley ...|   Green Lane|   PA|Brewpubs, Brewer
i...|
+--------------------+--------------------+------------+-----+--------------------
----+
only showing top 5 rows
```

Analyzing Categories

Let's now answer this question: **how many unique categories are represented in this dataset?**

Essentially, we have the categories per business as a list - this is useful to quickly see what each business might be represented as but it is difficult to easily answer questions such as:

- How many businesses are categorized as `Active Life` , for instance
- What are the top 20 most popular categories available?

Association Table

We need to "break out" these categories from the business ids? One common approach to take is to build an association table mapping a single business id multiple times to each distinct category.

For instance, given the following:

business_id	categories
abcd123	a,b,c

We would like to derive something like:

business_id	category
abcd123	a
abcd123	b
abcd123	c

What this does is allow us to then perform a myriad of rollups and other analysis on this association table which can aid us in answering the questions asked above.

Implement the code necessary to derive the table described from your original yelp dataframe.

In [9]:
```
# Install the necessary libraries here
# SQL functions
from pyspark.sql.functions import explode, split, col, mean, countDistinct
```

In [10]:
```
business.createOrReplaceTempView("business")
spark.sql("""
SELECT
    business_id,
    categories
FROM business
""").show(5, truncate=True)
```

Spark Job Progress

```
+--------------------+--------------------+
|         business_id|          categories|
+--------------------+--------------------+
|Pns2l4eNsfO8kk83d...|Doctors, Traditio...|
|mpf3x-BjTdTEA3yCZ...|Shipping Centers,...|
|tUFrWirKiKi_TAnsV...|Department Stores...|
|MTSW4McQd7CbVtyjq...|Restaurants, Food...|
|mWMc6_wTdE0EUBKIG...|Brewpubs, Breweri...|
+--------------------+--------------------+
only showing top 5 rows
```

Display the first 5 rows of your association table below.

In [11]:
```
business = business.withColumn('categories', explode(split('categories',', ')))

business.createOrReplaceTempView("business")
spark.sql("""
SELECT
    business_id,
    categories
FROM business
""").show(5, truncate=True)
```

Spark Job Progress

```
+--------------------+--------------------+
|         business_id|          categories|
+--------------------+--------------------+
|Pns2l4eNsfO8kk83d...|             Doctors|
|Pns2l4eNsfO8kk83d...|Traditional Chine...|
|Pns2l4eNsfO8kk83d...|Naturopathic/Holi...|
|Pns2l4eNsfO8kk83d...|         Acupuncture|
|Pns2l4eNsfO8kk83d...|    Health & Medical|
+--------------------+--------------------+
only showing top 5 rows
```

Total Unique Categories

Finally, we are ready to answer the question: **what is the total number of unique categories available?**

Below, implement the code necessary to calculate this figure.

In [12]:
```
business.createOrReplaceTempView("business")
spark.sql("""
SELECT
    DISTINCT categories
FROM business
""").count()
```

▸ Spark Job Progress

```
1311
```

Top Categories By Business

Now let's find the top categories in this dataset by rolling up categories.

Counts of Businesses / Category

So now, let's unroll our distinct count a bit and display the per count value of businesses per category.

The expected output should be:

category	count
a	15
b	2
c	45

Or something to that effect.

In [13]:
```
business.groupBy('categories').count().show()
```

Spark Job Progress

```
+--------------------+-----+
|          categories|count|
+--------------------+-----+
|      Paddleboarding|   98|
|      Dermatologists|  336|
|         Hobby Shops|  552|
|          Bubble Tea|  477|
|             Embassy|    3|
|             Tanning|  667|
|            Handyman|  356|
|      Aerial Fitness|   19|
|             Falafel|  103|
|        Summer Camps|  232|
|       Outlet Stores|  182|
|     Clothing Rental|   37|
|      Sporting Goods| 1662|
|     Cooking Schools|   76|
|  Lactation Services|   27|
|Ski & Snowboard S...|   40|
|             Museums|  413|
|              Doulas|   31|
|                Food|27781|
|         Halotherapy|   23|
+--------------------+-----+
only showing top 20 rows
```

Bar Chart of Top Categories

With this data available, let us now build a barchart of the top 20 categories.

HINT: don't forget about the matplotlib magic!

```
%matplot plt
```

If you want, you can also use seaborn library

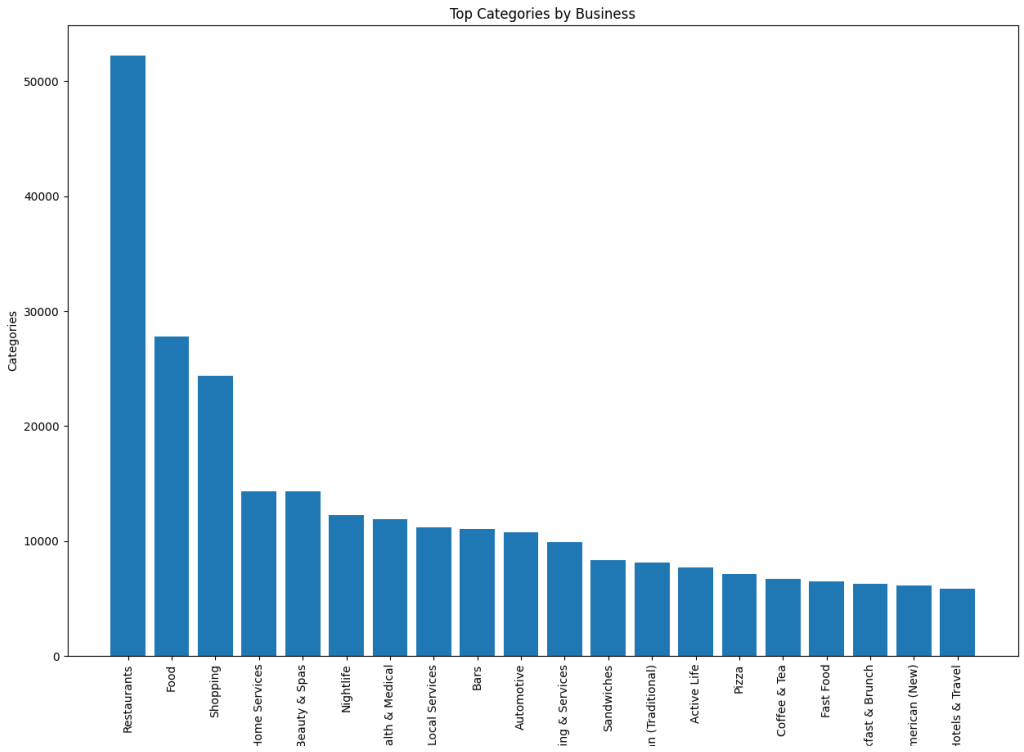
In [14]:
```
business.groupBy('categories').count().orderBy(col('count').desc()).show()
```

▸ Spark Job Progress

```
+--------------------+-----+
|          categories|count|
+--------------------+-----+
|         Restaurants|52268|
|                Food|27781|
|            Shopping|24395|
|       Home Services|14356|
|       Beauty & Spas|14292|
|           Nightlife|12281|
|    Health & Medical|11890|
|      Local Services|11198|
|                Bars|11065|
|          Automotive|10773|
|Event Planning & ...| 9895|
|          Sandwiches| 8366|
|American (Traditi...| 8139|
|         Active Life| 7687|
|               Pizza| 7093|
|        Coffee & Tea| 6703|
|           Fast Food| 6472|
|  Breakfast & Brunch| 6239|
|      American (New)| 6097|
|     Hotels & Travel| 5857|
+--------------------+-----+
only showing top 20 rows
```

In [15]:
```
top_business = business.groupBy('categories').count().orderBy(col('count').desc()
pandas_top_business = top_business.toPandas()
x = pandas_top_business.head(20)['categories'].values
y = pandas_top_business.head(20)['count'].values
plt.figure(figsize=(15,10))
plt.bar(x,y)
plt.xticks(rotation=90)
plt.xlabel('Count')
plt.ylabel('Categories')
plt.title('Top Categories by Business')
%matplot plt
```

▸ Spark Job Progress



Loading User Data

Begin by loading the user data set from S3 and printing schema to determine what data is available. `s3://cis9760-yelpdataset/yelp-light/*review.json`

In [16]:
```
review = spark.read.json('s3://xinyujiang-project02/yelp_academic_dataset_review.
```

▸ Spark Job Progress

In [17]:
```
review.printSchema()
```

```
root
 |-- business_id: string (nullable = true)
 |-- cool: long (nullable = true)
 |-- date: string (nullable = true)
 |-- funny: long (nullable = true)
 |-- review_id: string (nullable = true)
 |-- stars: double (nullable = true)
 |-- text: string (nullable = true)
 |-- useful: long (nullable = true)
 |-- user_id: string (nullable = true)
```

Let's begin by listing the `business_id` and `stars` columns together for the user reviews data.

In [18]:
```
review.createOrReplaceTempView("review")
spark.sql("""
SELECT
    business_id,
    stars
FROM review
""").show(5, truncate=True)
```

▸ Spark Job Progress

```
+--------------------+-----+
|         business_id|stars|
+--------------------+-----+
|XQfwVwDr-v0ZS3_Cb...|  3.0|
|7ATYjTIgM3jUlt4UM...|  5.0|
|YjUWPpI6HXG530lwP...|  3.0|
|kxX2SOes4o-D3ZQBk...|  5.0|
|e4Vwtrqf-wpJfwesg...|  4.0|
+--------------------+-----+
only showing top 5 rows
```

Now, let's aggregate along the `stars` column to get a resultant dataframe that displays *average stars* per business as accumulated by users who **took the time to submit a written review**.

In [19]:
```
review.createOrReplaceTempView("review")
spark.sql("""
SELECT
    business_id,
    AVG(stars)
FROM review
GROUP BY business_id
""").show(5, truncate=True)
```

Spark Job Progress

```
+--------------------+------------------+
|         business_id|        avg(stars)|
+--------------------+------------------+
|HSzSGdcNaU7heQe0N...|3.3333333333333335|
|skW4boArIApRw9DXK...|2.3947368421052633|
|zJErbOQMKX-MwHs_u...|2.9279279279279278|
|I0053JmJ5DEFUWSJ8...|2.3956043956043955|
|wS-SWAa_yaJAw6fJm...| 3.357142857142857|
+--------------------+------------------+
only showing top 5 rows
```

Now the fun part - let's join our two dataframes (reviews and business data) by `business_id` .

In [20]:
```
business = business.select('business_id','stars','name','city','state')
review = review.groupby('business_id').avg('stars')

merged = business.join(review, ['business_id'], "inner")
```

Let's see a few of these:

In [21]:
```
merged.show(5)
```

Spark Job Progress

```
+--------------------+-----+--------------------+-------+-----+----------------
--+
|         business_id|stars|                name|   city|state|        avg(star
s)|
+--------------------+-----+--------------------+-------+-----+----------------
--+
|HSzSGdcNaU7heQe0N...|  3.0|Gillane's Bar & G...|Ardmore|   PA|3.33333333333333
35|
|HSzSGdcNaU7heQe0N...|  3.0|Gillane's Bar & G...|Ardmore|   PA|3.33333333333333
35|
|HSzSGdcNaU7heQe0N...|  3.0|Gillane's Bar & G...|Ardmore|   PA|3.33333333333333
35|
|HSzSGdcNaU7heQe0N...|  3.0|Gillane's Bar & G...|Ardmore|   PA|3.33333333333333
35|
|HSzSGdcNaU7heQe0N...|  3.0|Gillane's Bar & G...|Ardmore|   PA|3.33333333333333
35|
+--------------------+-----+--------------------+-------+-----+----------------
--+
only showing top 5 rows
```

Compute a new dataframe that calculates what we will call the *skew* (for lack of a better word) between the avg stars accumulated from written reviews and the *actual* star rating of a business (ie: the average of stars given by reviewers who wrote an actual review **and** reviewers who just provided a star rating).

The formula you can use is something like:

```
(row['avg(stars)'] - row['stars']) / row['stars']
```

If the **skew** is negative, we can interpret that to be: reviewers who left a written response were more dissatisfied than normal. If **skew** is positive, we can interpret that to be: reviewers who left a written response were more satisfied than normal.

In [22]:
```
# Creating "skew" column
merged = merged.toPandas()
merged['skew'] = (merged['avg(stars)'] - merged['stars']) / merged['stars']

# Converting back to spark dataframe
merged = spark.createDataFrame(merged)
merged.show(5)
```

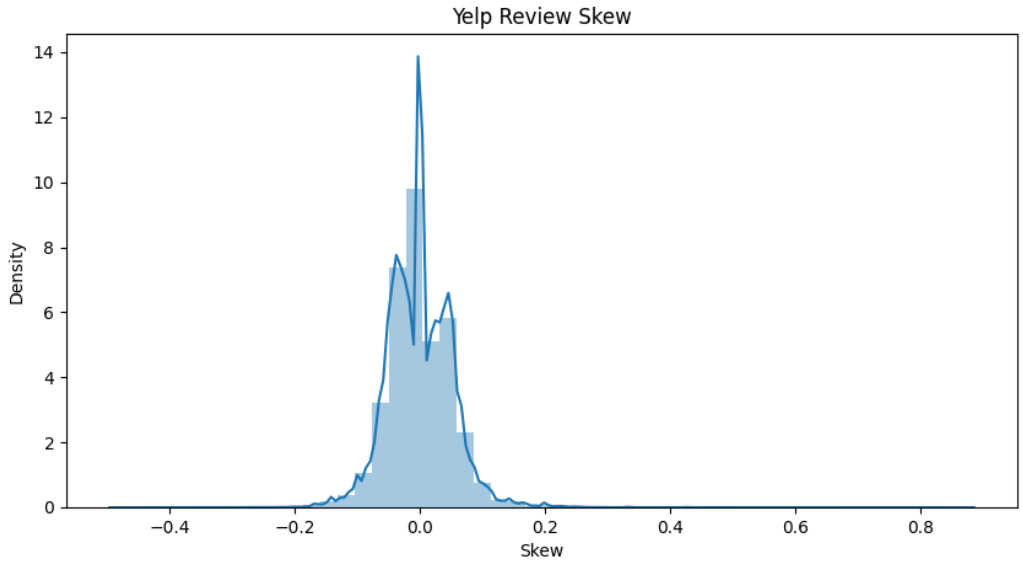
Spark Job Progress

```
+--------------------+-----+--------------------+------------+-----+-----------
-------+--------------------+
|         business_id|stars|                name|        city|state|        avg
(stars)|                skew|
+--------------------+-----+--------------------+------------+-----+-----------
-------+--------------------+
|zJErbOQMKX-MwHs_u...|  3.0|Philadelphia Marr...|Philadelphia|   PA|2.927927927
9279278|-0.02402402402402...|
|zJErbOQMKX-MwHs_u...|  3.0|Philadelphia Marr...|Philadelphia|   PA|2.927927927
9279278|-0.02402402402402...|
|zJErbOQMKX-MwHs_u...|  3.0|Philadelphia Marr...|Philadelphia|   PA|2.927927927
9279278|-0.02402402402402...|
|zJErbOQMKX-MwHs_u...|  3.0|Philadelphia Marr...|Philadelphia|   PA|2.927927927
9279278|-0.02402402402402...|
|RZ-FNTXvqHKngyLGD...|  3.0|Gaetano's of West...| West Berlin|   NJ|2.882352941
1764706|-0.03921568627450981|
+--------------------+-----+--------------------+------------+-----+-----------
-------+--------------------+
only showing top 5 rows
```

And finally, graph it!

In [23]:
```
# Using seaborn for the line on distribution plot
merged = merged.toPandas()
plt.figure(figsize=(10,5))
sns.distplot(merged['skew'], kde=True)
plt.title('Yelp Review Skew')
plt.xlabel('Skew')
plt.ylabel('Density')
%matplot plt
```

▸ Spark Job Progress



So, do Yelp (written) Reviews skew negative? Does this analysis actually prove anything? Expound on implications / interpretations of this graph.

IMPLICATIONS

Type your answer here:

From the graph above, it's hard to tell the skew because the density plot is nearly symmetric. However, it does seem like the graph is **very slightly** left skewed (negative). This means user reviews of restaurants tend to be more negative (lower user satisfaction).

If I want to explore further, I can use something like Pearson's Skewness Formula (https://www.investopedia.com/terms/s/skewness.asp#measuring-skewness) I found on Investopedia to obtain the exact numbers to establish skewness.

Should the Elite be Trusted?

How accurate or close are the ratings of an "elite" user (check Users table schema) vs the actual business rating? `s3://cis9760-yelpdataset/yelp-light/*user.json`

Feel free to use any and all methodologies at your disposal. You must render one visualization in your analysis and interpret your findings.

In [24]:
```
# Importing files from S3
user = spark.read.json('s3://xinyujiang-project02/yelp_academic_dataset_user.jsor
```

▸ Spark Job Progress

In [25]:
```
user.printSchema()
```

```
root
 |-- average_stars: double (nullable = true)
 |-- compliment_cool: long (nullable = true)
 |-- compliment_cute: long (nullable = true)
 |-- compliment_funny: long (nullable = true)
 |-- compliment_hot: long (nullable = true)
 |-- compliment_list: long (nullable = true)
 |-- compliment_more: long (nullable = true)
 |-- compliment_note: long (nullable = true)
 |-- compliment_photos: long (nullable = true)
 |-- compliment_plain: long (nullable = true)
 |-- compliment_profile: long (nullable = true)
 |-- compliment_writer: long (nullable = true)
 |-- cool: long (nullable = true)
 |-- elite: string (nullable = true)
 |-- fans: long (nullable = true)
 |-- friends: string (nullable = true)
 |-- funny: long (nullable = true)
 |-- name: string (nullable = true)
 |-- review_count: long (nullable = true)
 |-- useful: long (nullable = true)
 |-- user_id: string (nullable = true)
 |-- yelping_since: string (nullable = true)
```

In [26]:
```
user.createOrReplaceTempView("user")
spark.sql("""
SELECT
    user_id,
    elite,
    average_stars,
    yelping_since,
    review_count
FROM user
""").show(10, truncate=True)
```

Spark Job Progress

```
+--------------------+--------------------+------------+------------------+--
----------+
|             user_id|               elite|average_stars|      yelping_since|re
view_count|
+--------------------+--------------------+------------+------------------+--
----------+
|qVc8ODYU5SZjKXVBg...|                2007|         3.91|2007-01-25 16:47:26|
585|
|j14WgRoU_-2ZE1aw1...|2009,2010,2011,20...|         3.74|2009-01-25 04:35:42|
4333|
|2WnXYQFK0hXEoTxPt...|2009,2010,2011,20...|         3.32|2008-07-25 10:41:00|
665|
|SZDeASXq7o05mMNLs...|      2009,2010,2011|         4.27|2005-11-29 04:38:33|
224|
|hA5lMy-EnncsH4JoR...|                    |         3.54|2007-01-05 19:40:59|
79|
|q_QQ5kBBwlCcbL1s4...|2006,2007,2008,20...|         3.85|2005-03-14 20:26:35|
1221|
|cxuxXkcihfCbqt5By...|                    |         2.75|2009-02-24 03:09:06|
12|
|E9kcWJdJUHuTKfQur...|                    |         3.73|2008-12-11 22:11:56|
358|
|lO1iq-f75hnPNZkTy...|                    |         4.04|2008-12-29 22:40:56|
40|
|AUi8MPWJ0mLkMfwbu...|                    |          3.4|2010-01-07 18:32:04|
109|
+--------------------+--------------------+------------+------------------+--
----------+
only showing top 10 rows
```

In [27]:
```
# From the above we see that there are missing values for elite
# So I will filter only for values that exist and are NOT missing
user = user.filter(user['elite'] != '')
user.createOrReplaceTempView("user")
spark.sql("""
SELECT
    user_id,
    elite,
    average_stars,
    yelping_since,
    review_count
FROM user
""").show(10, truncate=True)
```

Spark Job Progress

```
+--------------------+--------------------+------------+-------------------+--
----------+
|             user_id|               elite|average_stars|      yelping_since|re
view_count|
+--------------------+--------------------+------------+-------------------+--
----------+
|qVc8ODYU5SZjKXVBg...|                2007|         3.91|2007-01-25 16:47:26|
585|
|j14WgRoU_-2ZE1aw1...|2009,2010,2011,20...|         3.74|2009-01-25 04:35:42|
4333|
|2WnXYQFK0hXEoTxPt...|2009,2010,2011,20...|         3.32|2008-07-25 10:41:00|
665|
|SZDeASXq7o05mMNLs...|      2009,2010,2011|         4.27|2005-11-29 04:38:33|
224|
|q_QQ5kBBwlCcbL1s4...|2006,2007,2008,20...|         3.85|2005-03-14 20:26:35|
1221|
|xoZvMJPDW6Q9pDAXI...| 2009,2010,2011,2012|         3.89|2009-05-27 06:12:10|
535|
|SgiBkhXeqIKl1PlFp...|2007,2008,2009,20...|         3.75|2006-08-25 16:47:25|
682|
|QF1Kuhs8iwLWANNZx...|2010,2011,2012,20...|         4.11|2009-04-27 20:25:54|
607|
|FT9CFS39sjZxVjCTr...|           2015,2016|         3.52|2010-06-14 21:44:28|
201|
|MGPQVLsODMm9ZtYQW...|2008,2009,2010,20...|         4.06|2008-01-19 22:50:00|
1807|
+--------------------+--------------------+------------+-------------------+--
----------+
only showing top 10 rows
```

In [28]:
```
# Reimporting and selecting columns
business = spark.read.json('s3://xinyujiang-project02/yelp_academic_dataset_busir
business = business.select('business_id','name','city','state')

review = spark.read.json('s3://xinyujiang-project02/yelp_academic_dataset_review.
review_userID = review.select('user_id','business_id','stars')
review = review.groupby('business_id').avg('stars')

merged = business.join(review, ['business_id'], "inner")
```

▸ Spark Job Progress

In [29]:
```
# Merging
elite_user_reviews = user.join(review_userID, ['user_id'], 'inner')
elite_user_revuews = elite_user_reviews.drop(elite_user_reviews['stars'])
elite_user_reviews = elite_user_reviews.join(review, ['business_id'], 'inner')
elite_user_reviews = elite_user_reviews.select('user_id','elite','stars','average

# elite_user_reviews.createOrReplaceTempView("elite_user_reviews")
# spark.sql("""
# SELECT
#     user_id,
#     elite,
#     average_stars,
#     business_id,
#     avg(stars)
# FROM elite_user_reviews
# """).show(5, truncate=True)

elite_user_reviews.show(5)
```

▸ Spark Job Progress

```
+--------------------+--------------------+-----+------------+---------------
----+----------------+
|             user_id|               elite|stars|average_stars|         busines
s_id|       avg(stars)|
+--------------------+--------------------+-----+------------+---------------
----+----------------+
|fen9BWC39ul9SJZfQ...|2017,2018,2019,20...|  4.0|         3.87|--gJkxbsiSIwsQKb
i...|4.833333333333333|
|7jOaJw3txVFlkHB7Y...|           2015,2016|  5.0|         4.49|-02xFuruu85XmDn2
x...| 4.68595041322314|
|_VZlDBtCT_Qb3_OOT...|           2018,2019|  5.0|         4.07|-02xFuruu85XmDn2
x...| 4.68595041322314|
|EO7u_L1_ZgRdawMrb...|2017,2018,2019,20...|  5.0|         3.93|-02xFuruu85XmDn2
x...| 4.68595041322314|
|wqeGcKWbtQLyavwtq...|                2021|  5.0|         3.81|-02xFuruu85XmDn2
x...| 4.68595041322314|
+--------------------+--------------------+-----+------------+---------------
----+----------------+
only showing top 5 rows
```

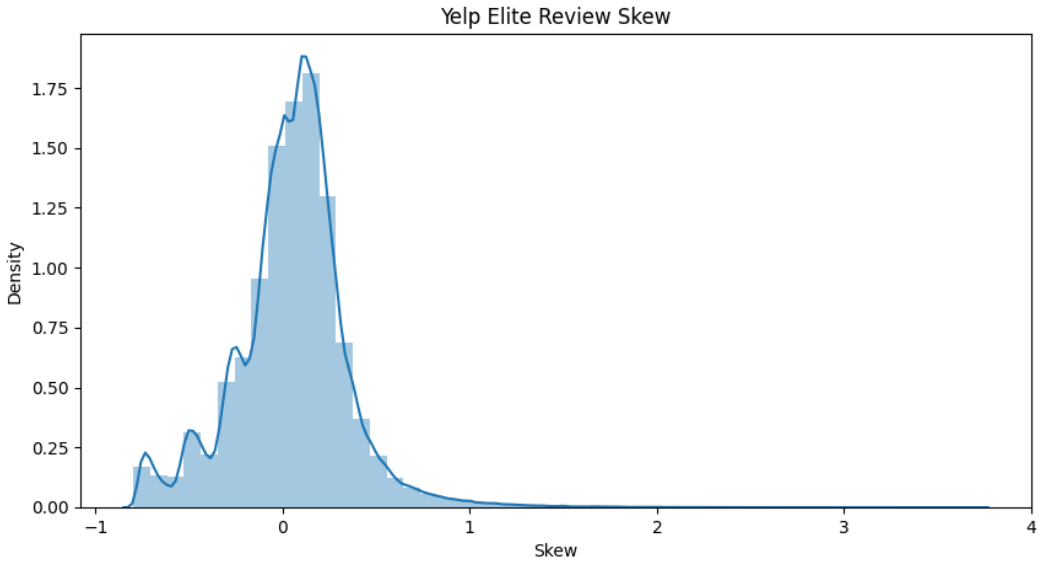
In [30]:
```
# Skew column, same as previous section
elite_user_reviews = elite_user_reviews.toPandas()
elite_user_reviews['skew'] = (elite_user_reviews['stars'] - elite_user_reviews['a
elite_user_reviews = spark.createDataFrame(elite_user_reviews)
elite_user_reviews.show(5)
```

▸ Spark Job Progress

```
+--------------------+--------------------+-----+------------+---------------
----+----------------+--------------------+
|             user_id|               elite|stars|average_stars|        busines
s_id|       avg(stars)|                skew|
+--------------------+--------------------+-----+------------+---------------
----+----------------+--------------------+
|fen9BWC39ul9SJZfQ...|2017,2018,2019,20...|  4.0|        3.87|--gJkxbsiSIwsQKb
i...|4.833333333333333|-0.17241379310344823|
|7jOaJw3txVFlkHB7Y...|           2015,2016|  5.0|        4.49|-02xFuruu85XmDn2
x...| 4.68595041322314| 0.06701940035273375|
|_VZlDBtCT_Qb3_OOT...|           2018,2019|  5.0|        4.07|-02xFuruu85XmDn2
x...| 4.68595041322314| 0.06701940035273375|
|EO7u_L1_ZgRdawMrb...|2017,2018,2019,20...|  5.0|        3.93|-02xFuruu85XmDn2
x...| 4.68595041322314| 0.06701940035273375|
|2gyrlO8oOuGf5JMOe...|2017,2018,2019,20...|  5.0|        4.06|-02xFuruu85XmDn2
x...| 4.68595041322314| 0.06701940035273375|
+--------------------+--------------------+-----+------------+---------------
----+----------------+--------------------+
only showing top 5 rows
```

In [31]:
```
# Seaborn distribution of elite skews
elite_user_reviews = elite_user_reviews.toPandas()
plt.figure(figsize=(10,5))
sns.distplot(elite_user_reviews['skew'], kde=True)
plt.title('Yelp Elite Review Skew')
plt.xlabel('Skew')
plt.ylabel('Density')
%matplot plt
```

▸ Spark Job Progress



IMPLICATIONS

The graph above is skewed left - meaning elite users left more negative reviews than normal users. This could be due to having the "Elite" status, these users could be more critical of the restaurant.

It is worth pointing out that the skewness is not too strong - meaning leaving negative reviews may not be intentional (just for the sake of leaving negative reviews).

Extra Credit (3 points)

Try and analyze some interesting dimension to this data. **Requirements:**

You must use the **Users** dataset and join on either the "**business** or **reviews** dataset.

You must render **one visual**

In [ ]: