# DCS 640 Data Presentation & Visualization (DSC640-T302 2231-1)

## Bellevue University

## 5.2 Exercises: Heat Map, Spatial Charts, and Contour Charts

## Author: Jake Meyer

## Date: 02/10/2023

## Assignment Instructions:

Submit 1 heat map, 1 spatial chart, and 1 contour chart with Python

In [55]:
```
'''
Import the necessary libraries to complete Exercise 2.2.
'''
import numpy as np
import pandas as pd
import seaborn as sns
import scipy.stats
import matplotlib
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import plotly.express as px
from matplotlib import cm
%matplotlib inline
```

In [2]:
```
'''
Check the versions of the packages.
'''
print('numpy version:', np.__version__)
print('pandas version:', pd.__version__)
print('seaborn version:', sns.__version__)
print('matplotlib version:', matplotlib.__version__)
```

```
numpy version: 1.20.3
pandas version: 1.3.4
seaborn version: 0.11.2
matplotlib version: 3.4.3
```

## Dataset Understanding

In [24]:
```
'''
Import the datasets.
Note: A copy of the CSV file was placed into the same directory as this notebook.
Utilize pd.read_csv() to read the file as a pandas data frame.
'''
df1 = pd.read_csv('ppg2008.csv')
df2 = pd.read_csv('costcos-geocoded.csv')
```

In [25]:
```
'''
Use head() function to display the first 5 rows of data of df1.
```

```
'''
```

Out[25]:

| | Name | G | MIN | PTS | FGM | FGA | FGP | FTM | FTA | FTP | ... | 3PA | 3PP | ORB | DRB | TRB | AST | STL | BLK | TO | PF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Dwyane Wade | 79 | 38.6 | 30.2 | 10.8 | 22.0 | 0.491 | 7.5 | 9.8 | 0.765 | ... | 3.5 | 0.317 | 1.1 | 3.9 | 5.0 | 7.5 | 2.2 | 1.3 | 3.4 | 2.3 |
| **1** | LeBron James | 81 | 37.7 | 28.4 | 9.7 | 19.9 | 0.489 | 7.3 | 9.4 | 0.780 | ... | 4.7 | 0.344 | 1.3 | 6.3 | 7.6 | 7.2 | 1.7 | 1.1 | 3.0 | 1.7 |
| **2** | Kobe Bryant | 82 | 36.2 | 26.8 | 9.8 | 20.9 | 0.467 | 5.9 | 6.9 | 0.856 | ... | 4.1 | 0.351 | 1.1 | 4.1 | 5.2 | 4.9 | 1.5 | 0.5 | 2.6 | 2.3 |
| **3** | Dirk Nowitzki | 81 | 37.7 | 25.9 | 9.6 | 20.0 | 0.479 | 6.0 | 6.7 | 0.890 | ... | 2.1 | 0.359 | 1.1 | 7.3 | 8.4 | 2.4 | 0.8 | 0.8 | 1.9 | 2.2 |
| **4** | Danny Granger | 67 | 36.2 | 25.8 | 8.5 | 19.1 | 0.447 | 6.0 | 6.9 | 0.878 | ... | 6.7 | 0.404 | 0.7 | 4.4 | 5.1 | 2.7 | 1.0 | 1.4 | 2.5 | 3.1 |

5 rows × 21 columns

In [26]:
```
'''
Use head() function to display the first 5 rows of data of df1.
'''
df2.head()
```

Out[26]:

| | Address | City | State | Zip Code | Latitude | Longitude |
|---|---|---|---|---|---|---|
| **0** | 1205 N. Memorial Parkway | Huntsville | Alabama | 35801-5930 | 34.743095 | -86.600955 |
| **1** | 3650 Galleria Circle | Hoover | Alabama | 35244-2346 | 33.377649 | -86.812420 |
| **2** | 8251 Eastchase Parkway | Montgomery | Alabama | 36117 | 32.363889 | -86.150884 |
| **3** | 5225 Commercial Boulevard | Juneau | Alaska | 99801-7210 | 58.359200 | -134.483000 |
| **4** | 330 West Dimond Blvd | Anchorage | Alaska | 99515-1950 | 61.143266 | -149.884217 |

In [27]:
```
'''
Understand the shape of the df1.
'''
print('There are {} rows and {} columns in the df1.'.format(df1.shape[0], df1.shape[1]))
```

```
There are 50 rows and 21 columns in the df1.
```

In [28]:
```
'''
Understand the shape of the df1.
'''
print('There are {} rows and {} columns in the df1.'.format(df2.shape[0], df2.shape[1]))
```

```
There are 417 rows and 6 columns in the df1.
```

In [29]:
```
'''
Find the type of data within each df1 column initially.
'''
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 21 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Name    50 non-null     object
 1   G       50 non-null     int64
 2   MIN     50 non-null     float64
 3   PTS     50 non-null     float64
 4   FGM     50 non-null     float64
 5   FGA     50 non-null     float64
 6   FGP     50 non-null     float64
 7   FTM     50 non-null     float64
```

```
 8   FTA     50 non-null     float64
 9   FTP     50 non-null     float64
 10  3PM     50 non-null     float64
 11  3PA     50 non-null     float64
 12  3PP     50 non-null     float64
 13  ORB     50 non-null     float64
 14  DRB     50 non-null     float64
 15  TRB     50 non-null     float64
 16  AST     50 non-null     float64
 17  STL     50 non-null     float64
 18  BLK     50 non-null     float64
 19  TO      50 non-null     float64
 20  PF      50 non-null     float64
dtypes: float64(19), int64(1), object(1)
memory usage: 8.3+ KB
```

In [30]:

```
'''
Find the type of data within each df1 column initially.
'''
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 417 entries, 0 to 416
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Address    417 non-null    object
 1   City       417 non-null    object
 2   State      417 non-null    object
 3   Zip Code   417 non-null    object
 4   Latitude   417 non-null    float64
 5   Longitude  417 non-null    float64
dtypes: float64(2), object(4)
memory usage: 19.7+ KB
```

In [31]:

```
'''
Understand if there are any missing values in df1.
'''
df1.isna().sum().sort_values(ascending = False)
```

Out[31]:

```
Name      0
3PA       0
TO        0
BLK       0
STL       0
AST       0
TRB       0
DRB       0
ORB       0
3PP       0
3PM       0
G         0
FTP       0
FTA       0
FTM       0
FGP       0
FGA       0
FGM       0
PTS       0
MIN       0
PF        0
dtype: int64
```

In [32]:

```
'''
Understand if there are any missing values in df1.
```

```
'''
```

Out[32]:
```
Address      0
City         0
State        0
Zip Code     0
Latitude     0
Longitude    0
dtype: int64
```

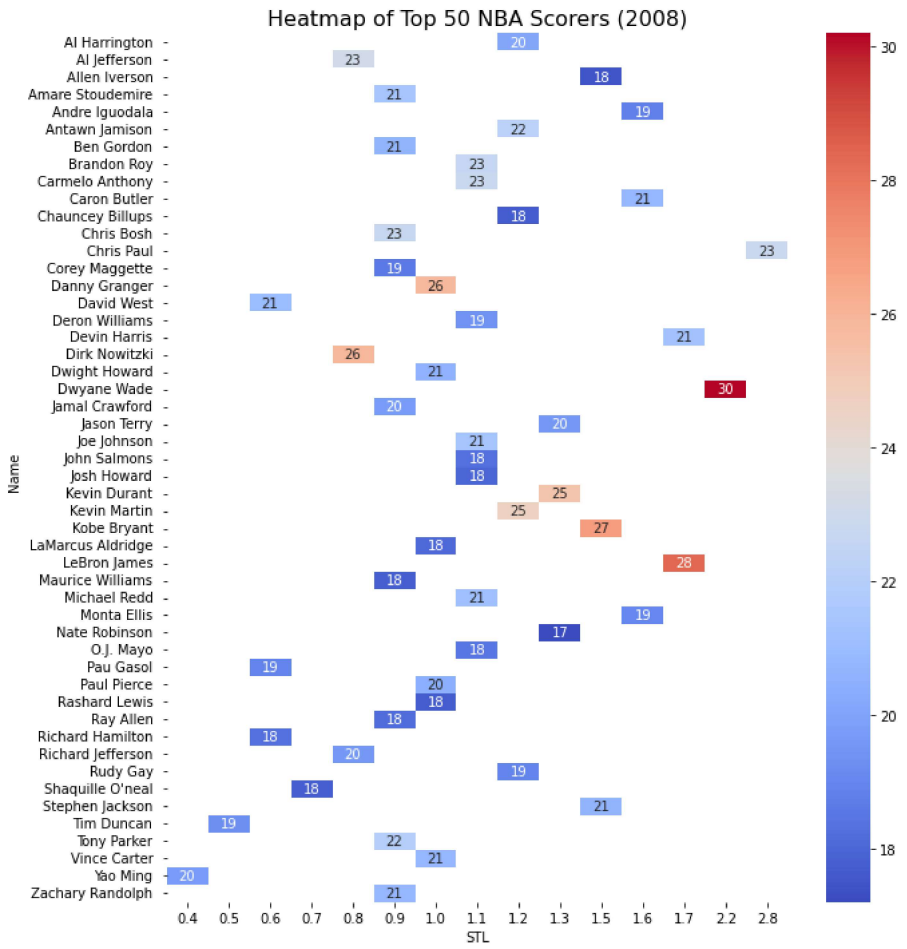
# Chart Creation from the Datasets.

## Heat Map

In [58]:
```
'''
Get the data in the correct format using pivot().
'''
bball = df1.pivot('Name  ','STL', 'PTS')
```

In [59]:
```
'''
Create the heat map using seaborn.
'''
fig, ax = plt.subplots(1, 1, figsize=(10,10), tight_layout = True)
sns.heatmap(bball, annot = True, cmap = 'coolwarm')
plt.title('Heatmap of Top 50 NBA Scorers (2008)', fontsize = 16)
```

Out[59]:
```
Text(0.5, 1.0, 'Heatmap of Top 50 NBA Scorers (2008)')
```

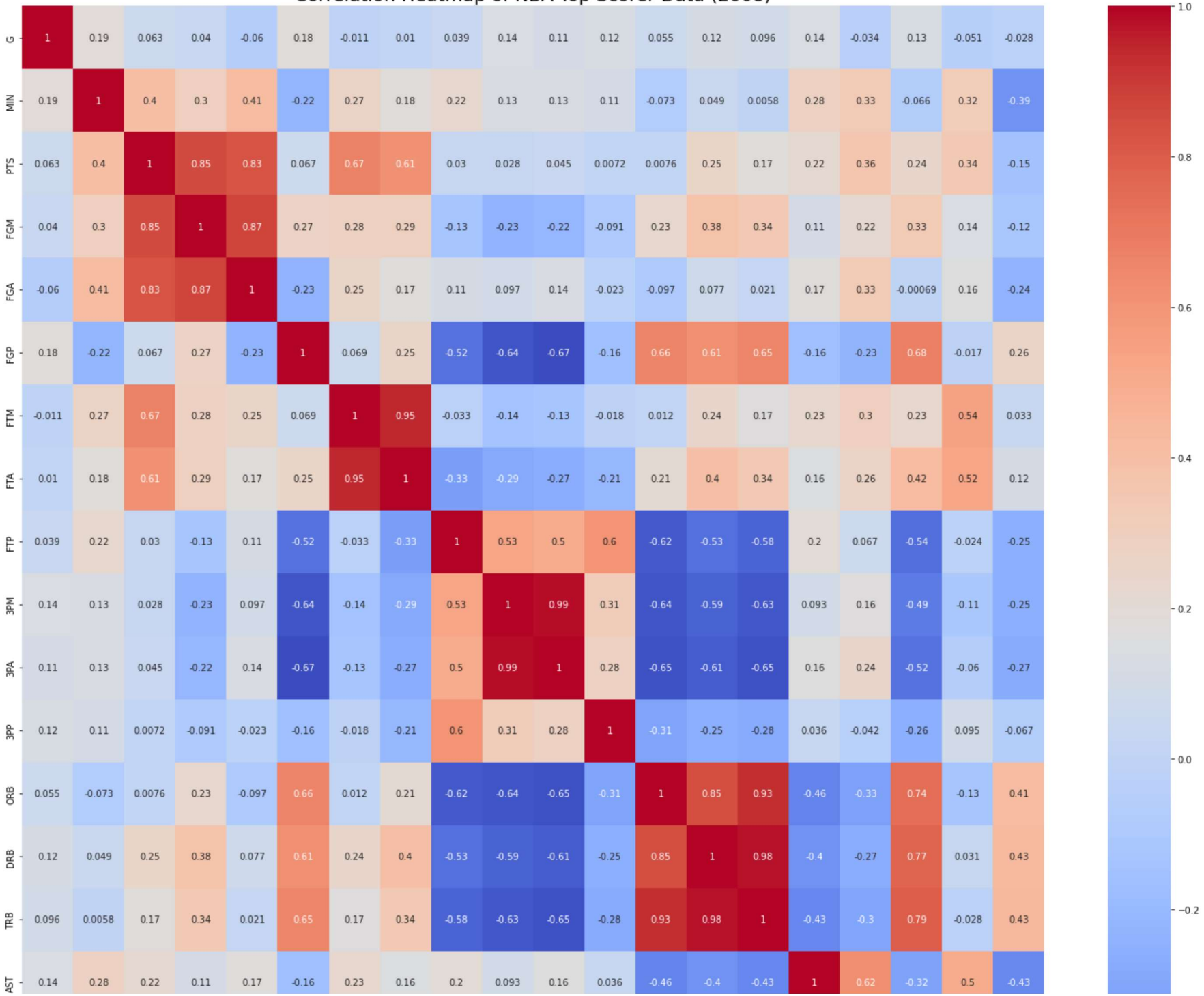



In [61]:
```
'''
Create a data frame with only numeric values.
'''
bball = df1.drop(['Name  '], axis = 1)
```

In [64]:
```
'''
Create a correlation heat map.
'''
# Calculate the correlation coefficient with corr().
correlation_number = bball.corr()

# Create the heatmap for the correlation coefficients calculated above.
fig, ax = plt.subplots(1, 1, figsize=(20,20), tight_layout = True)
sns.heatmap(correlation_number, annot = True, cmap = 'coolwarm')
plt.title('Correlation Heatmap of NBA Top Scorer Data (2008)', fontsize = 20)
```

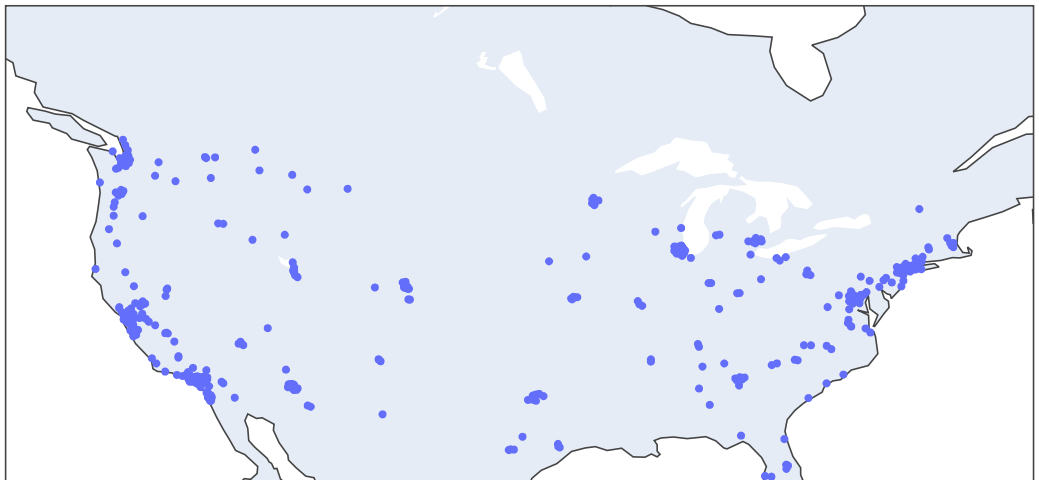
Out[64]:
```
Text(0.5, 1.0, 'Correlation Heatmap of NBA Top Scorer Data (2008)')
```

# Correlation Heatmap of NBA Top Scorer Data (2008)



## Spatial Chart

In [67]:
```
'''
Create the spatial chart using plotly.
'''
fig = px.scatter_geo(df2, lat = df2['Latitude'], lon = df2['Longitude'], hover_name = "City")
fig.show()
```

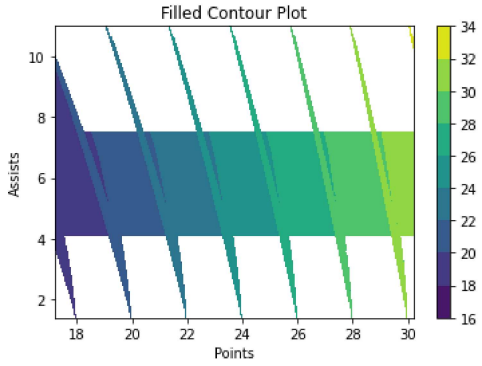


In [72]:
```
'''
Try a Density Plot with plotly
'''

fig = px.density_mapbox(df2, lat = 'Latitude', lon = 'Longitude', radius = 3, center = dict(lat = 0, lon = 180),
                        zoom = 2, mapbox_style = "stamen-terrain")

fig.show()
```


## Contour Plot

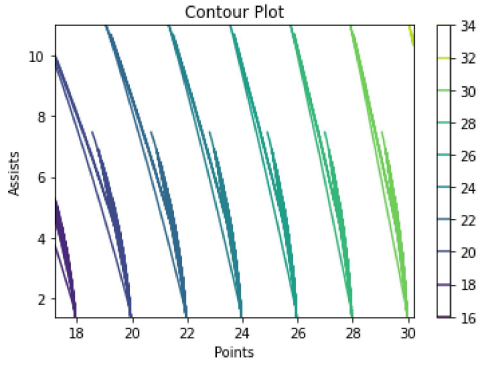
In [73]:
```
'''
Create a meshgrid using points and assists.
'''
X,Y = np.meshgrid(df1['PTS'], df1['AST'])
Z = np.sqrt(X**2 + Y**2)
```

In [89]:
```
'''
Create the contour plot.
'''
fig, ax = plt.subplots(1,1)
cp = ax.contourf(X, Y, Z)

fig.colorbar(cp)
ax.set_title('Filled Contour Plot')
plt.xlabel('Points')
plt.ylabel('Assists')
plt.show()
```




In [87]:
```
'''
Create the contour plot.
'''
fig, ax = plt.subplots(1,1)
cp = ax.contour(X, Y, Z)
```

```
fig.colorbar(cp)
ax.set_title('Contour Plot')
plt.xlabel('Points')
plt.ylabel('Assists')
plt.show()
```




In [76]:
```
'''
Try creating a contour plot with ellipse.
'''

ellipses = X*X/7 + Y*Y/5 - 1
```

In [78]:
```
'''
Create the ellipse contour plot.
'''
cnt = plt.contour(ellipses)
plt.clabel(cnt);
plt.colorbar()
plt.show()
```