Heat Map – Python

```
# plot a heatmap for features' correlations
plt.figure(figsize=(18,8))
sns.heatmap(corr, linewidth=0.3, annot=True)
plt.show()
```

Background Information:

A heat map is created by using Python – Seaborn. The data set is ppg2008. In this data set, we can find the statistics of many popular basketball players. After removing two columns (players' name and game number), the rest of the columns is used as features, and their correlations are graphed in a heat map. This heat map allows the audience to see how each feature is correlated with other features.

Method:

- Load the libraries and the dataset into a data frame
- Remove the Name and G columns from the dataset
- Find the correlations between all other features with the corr() function
- Plot the heat map by using seaborn.heatmap() function

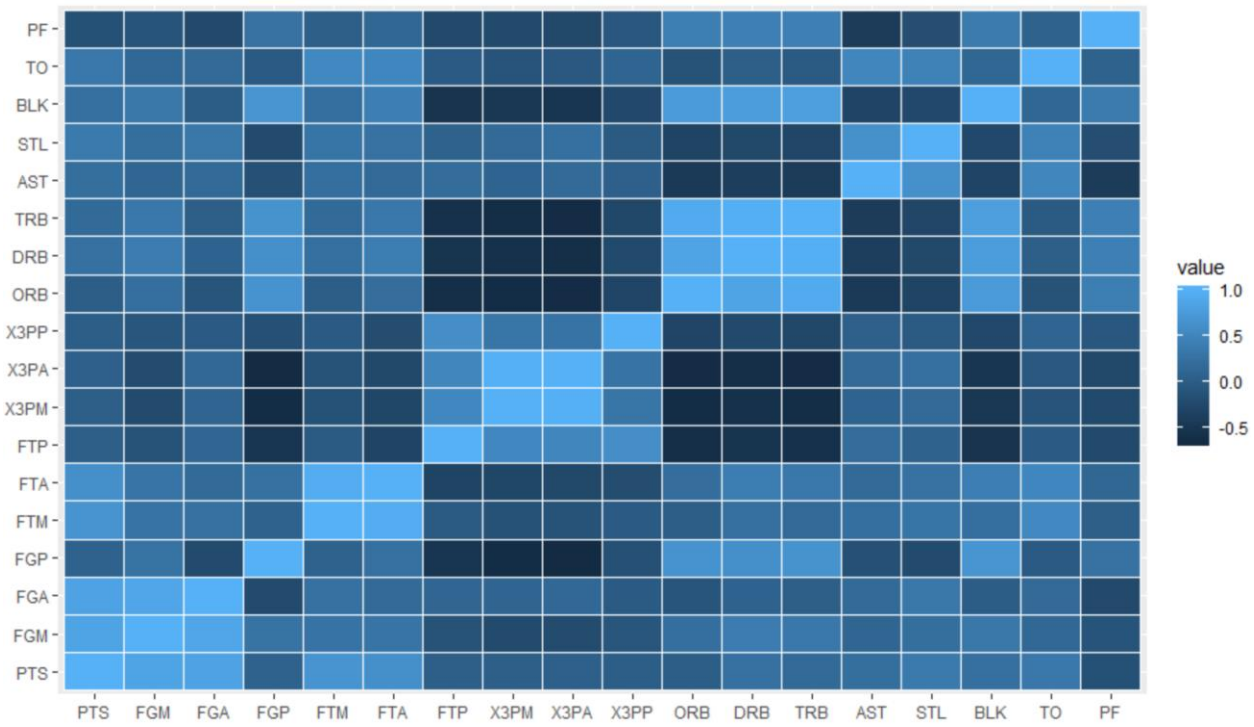
Heat Map – R

```
# load data
costco <- read.csv('costcos-geocoded.csv')
ppg <- read.csv('ppg2008.csv')

# remove first two columns
ppg <- ppg[ -c(1, 2)]

# compute correlation coefficients
cormat <- round(cor(ppg),2)
melted_cormat <- melt(cormat)

# plot the heat map
ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile(color = "white")
```



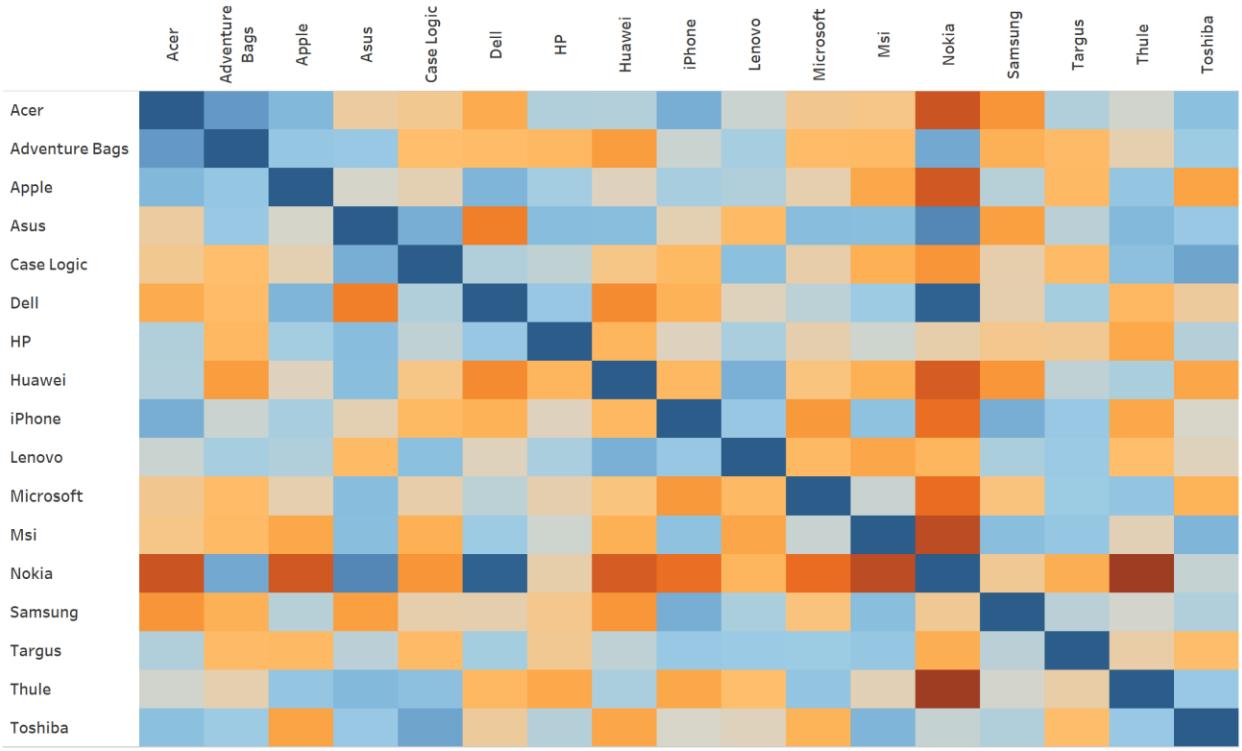
Background Information:

A heat map is created by using R – ggplot2. The data set is ppg2008. In this data set, we can find the statistics of many popular basketball players. After removing two columns (players' name and game number), the rest of the columns is used as features, and their correlations are graphed in a heat map. This heat map allows the audience to see how each feature is correlated with other features.

Method:

- Load the libraries and the dataset into a data frame
- Remove the Name and G columns from the dataset
- Find the correlations between all other features with the cormat() function
- Plot the heat map by using ggplot() and geom_tile() function

Heat Map – Tableau



Background Information:

A heat map is created by using Tableau. The data set is Tech-Sales-Data-18. In this data set, we can find the order numbers and the sale quantities for tech companies. This heat map shows the correlations between order numbers and sale quantities. The deep blue color represents correlation coefficient equal 1 whereas the deep red means the correlation coefficient equal -1.

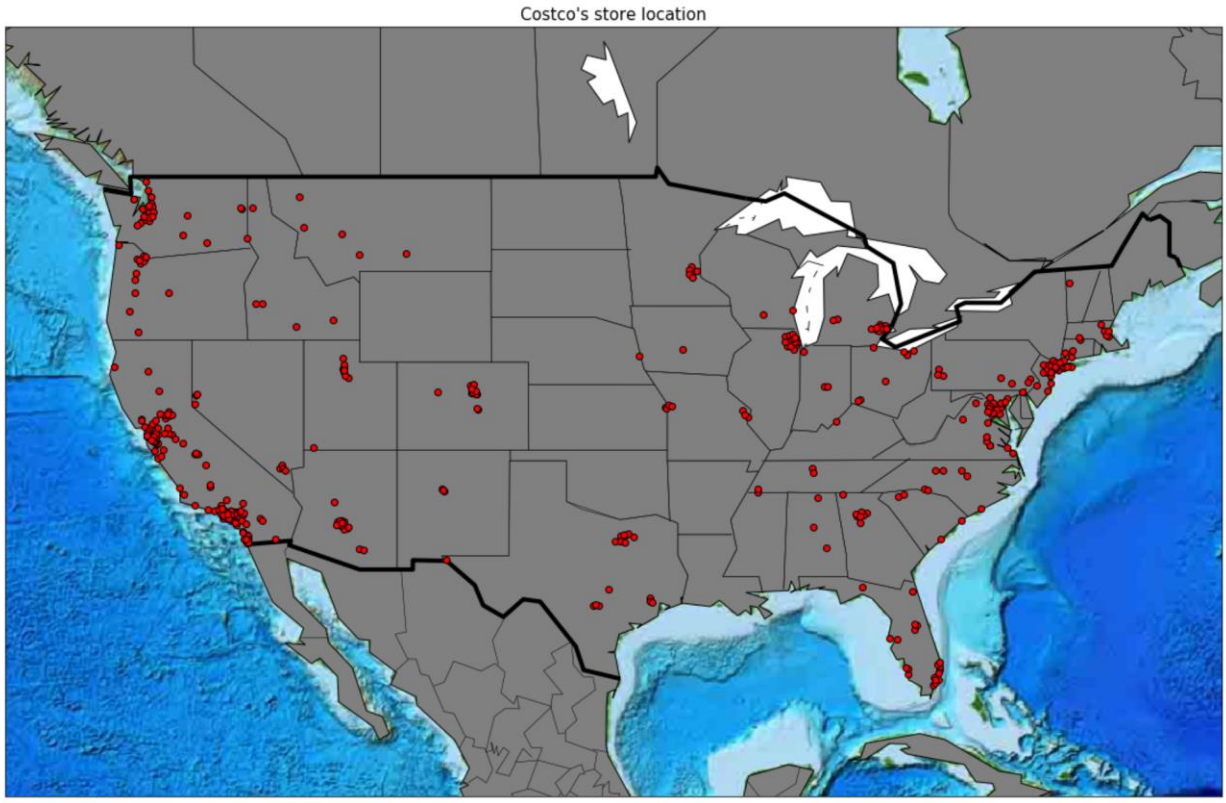
Method:

- Load the data from the data source
- Drag the dataset to the empty space on top of the table in the Data Source tab
- Drag the name from the data set to the columns tab
- Drag the name of the duplicated data set to the rows tab
- Right click on the column and select create calculated field
- Use this formula to calculate the correlation: CORR({INCLUDE [Order Number] : SUM([Quantity])}, {INCLUDE [Order Number] : SUM([Quantity (Tech Sales Data1)])})

Spatial Chart – Python

```
# plot a spatial map for costco's store location
plt.figure(figsize=(24,13))
m = Basemap(projection='mill', llcrnrlon=-130, llcrnrlat=20, urcrnrlon=-65, urcrnrlat=55 , resolution= 'c')
m.drawcoastlines()
m.drawcountries(color='black', linewidth=4)
m.drawstates()
m.fillcontinents(color='gray')
m.scatter(lon,lat, latlon=True, c='red', alpha=1, edgecolor='k', linewidth=1, zorder=2)
m.etopo()

plt.title("Costco's store location", fontsize=15)
plt.show()
```



Background Information:

A spatial chart is created by using Python – Basemap. The data set is costcos-geocoded. In this data set, the user can find the location of Costco stores from their longitudes and latitudes. After loading the data set, the columns of longitudes and latitudes are selected to plot. In this spatial plot, the audience can see where the Costco stores are located by the red dots.

Method:

- Load the libraries and the dataset into a data frame
- Select the longitude and latitude column from the data frame
- Using Basemap package to plot
- Resize the map by adjusting the longitudes and latitudes

Spatial Chart – R

```r
# pick the longitude and latitude from dataframe
df <- costco[c("Longitude","Latitude")]
# create spatial data
coordinates(chi_dat) <- df
crs.geo1 <- CRS("+proj=longlat")
proj4string(chi_dat) <- crs.geo1

plot(chi_dat, pch=20, col='steelblue')
# plot us map and the data
us <- readOGR(dsn="C:/map", layer="cb_2018_us_nation_5m")
plot(us)
points(chi_dat, pch=20, col="red")
```

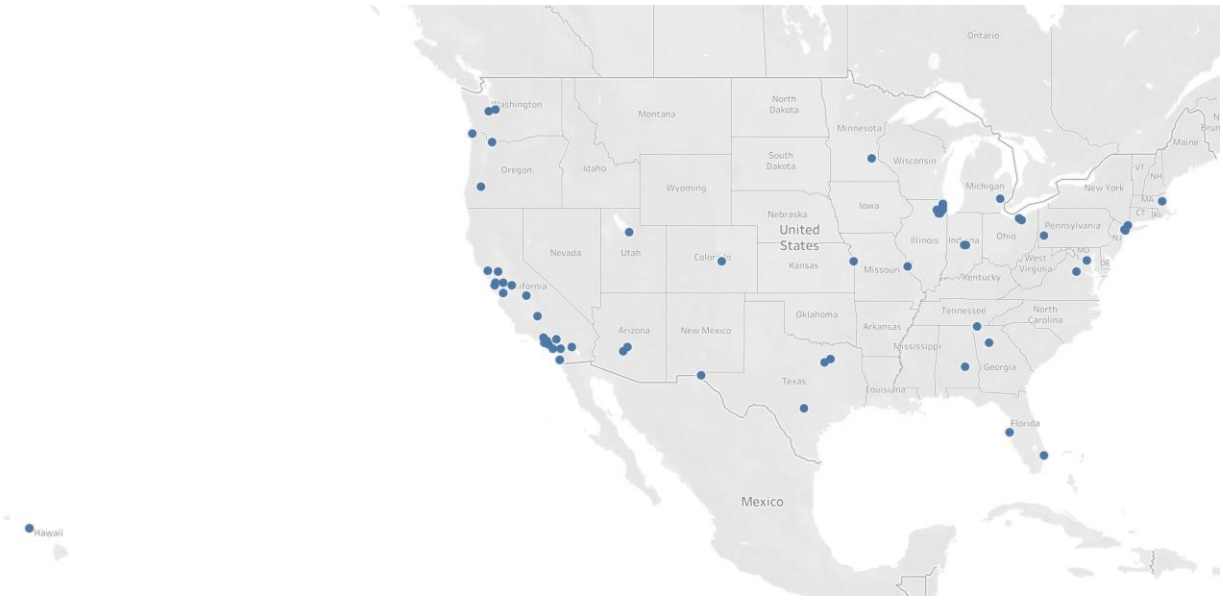
Background Information:

A spatial chart is created by using R – rgdal, raster, sp. The data set is costcos-geocoded. In this data set, the user can find the location of Costco stores from their longitudes and latitudes. After loading the data set, the columns of longitudes and latitudes are selected to plot. In this spatial plot, the audience can see where the Costco stores are located by the red dots. R does not allow me to adjust the size of the map to zoom in, and I can only use an existing map from the internet.

Method:

- Load the libraries and the dataset into a data frame
- Select the longitude and latitude column from the data frame
- Create spatial data
- Use readOGR() function to read the downloaded map
- Plot the US map and the data together

Spatial Chart – Tableau



Background Information:

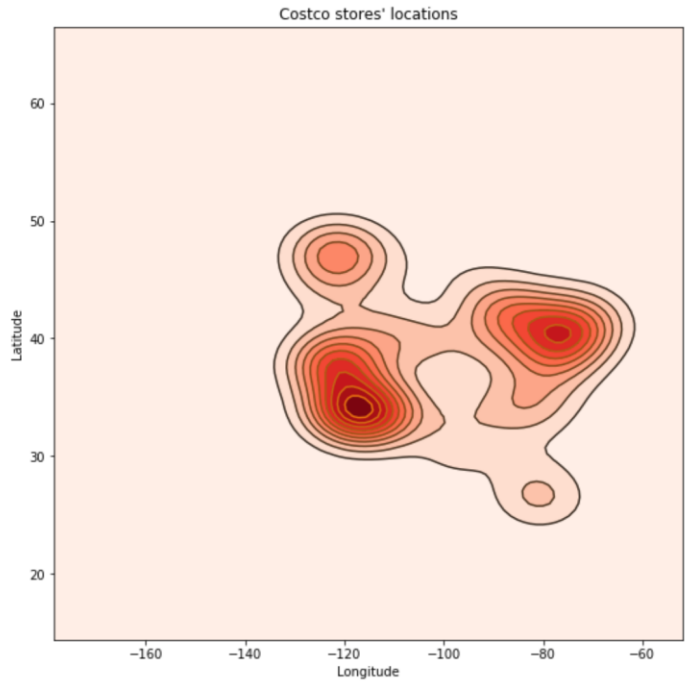
A spatial chart is created by using Tableau. The data set is costcos-geocoded. In this data set, the user can find the location of Costco stores from their longitudes and latitudes. In this spatial plot, the audience can see where the Costco stores are located by the blue dots. Method:

- Load the data set from the data source
- Drag the Longitude (generated) to the Columns tab
- Drag the Latitude (generated) to the Rows tab
- Drag the Zip Code (generated) to the Detail tab

Contour Chart – Python

```
# select x y
x = costco['Longitude'].values
y = costco['Latitude'].values
```

```
# plot contour chart
plt.figure(figsize=(9,9))
sns.kdeplot(x, y, cmap='Reds', shade=True)
sns.kdeplot(x, y)
plt.title("Costco stores' locations")
plt.xlabel("Longitude")
plt.ylabel("Latitude")
plt.show()
```



Background Information:

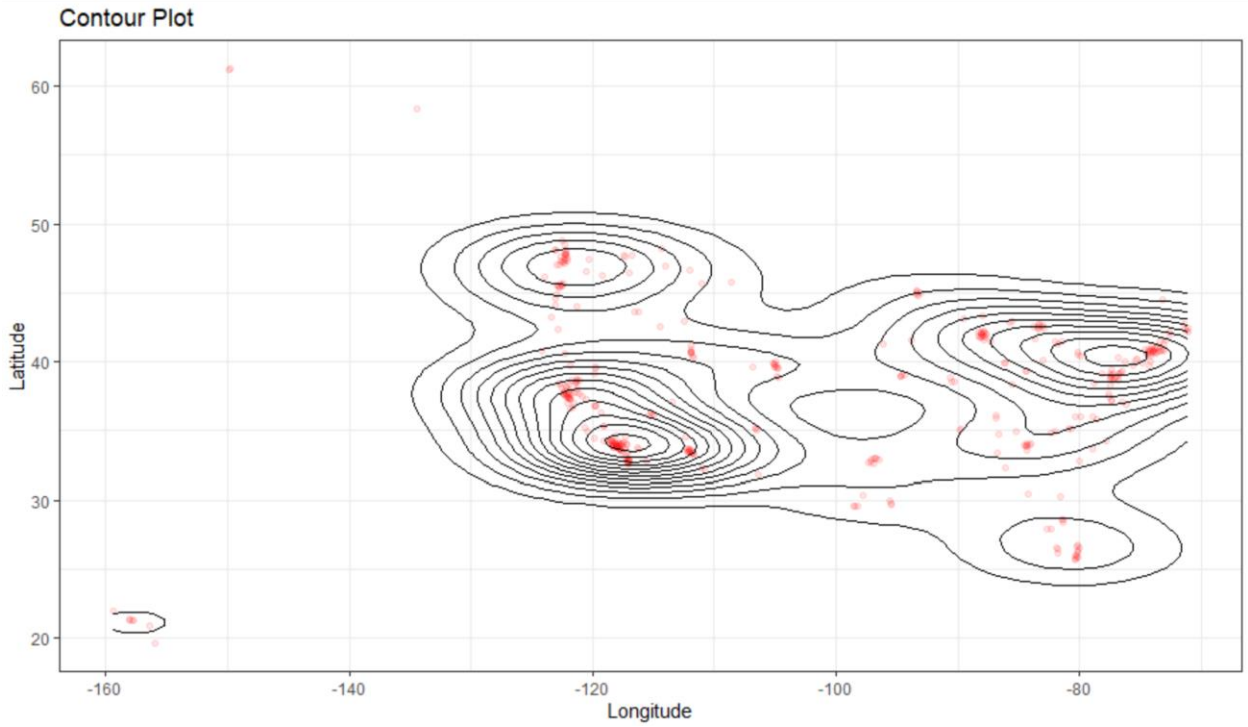
A contour chart is created by using Python – Seaborn. The data set is costcos-geocoded. In this data set, the user can find the concentration of Costco stores in the US, where it is present the most and the least. After loading the data set, the columns of longitudes and latitudes are selected to plot. In this contour chart, the audience can see the highest concentration of stores based on the deep red color and the lowest concentration of stores based on the orange color. An alternative is to use the contour() function in Matplotlib, which requires X, Y, Z to be in two-dimensional data.

Method:

- Load the libraries and the dataset into a data frame
- Select the longitude and latitude column from the data frame
- Using Seaborn package to plot
- Using the two kdeplot() functions to plot the values

Contour Chart – R

```
50  ggplot(data = df, aes(x = Longitude, y = Latitude)) +
51    geom_point(alpha=0.1, col="red") +
52    geom_density2d(color="black") +
53    ggtitle("Contour Plot") +
54    theme_bw()
```



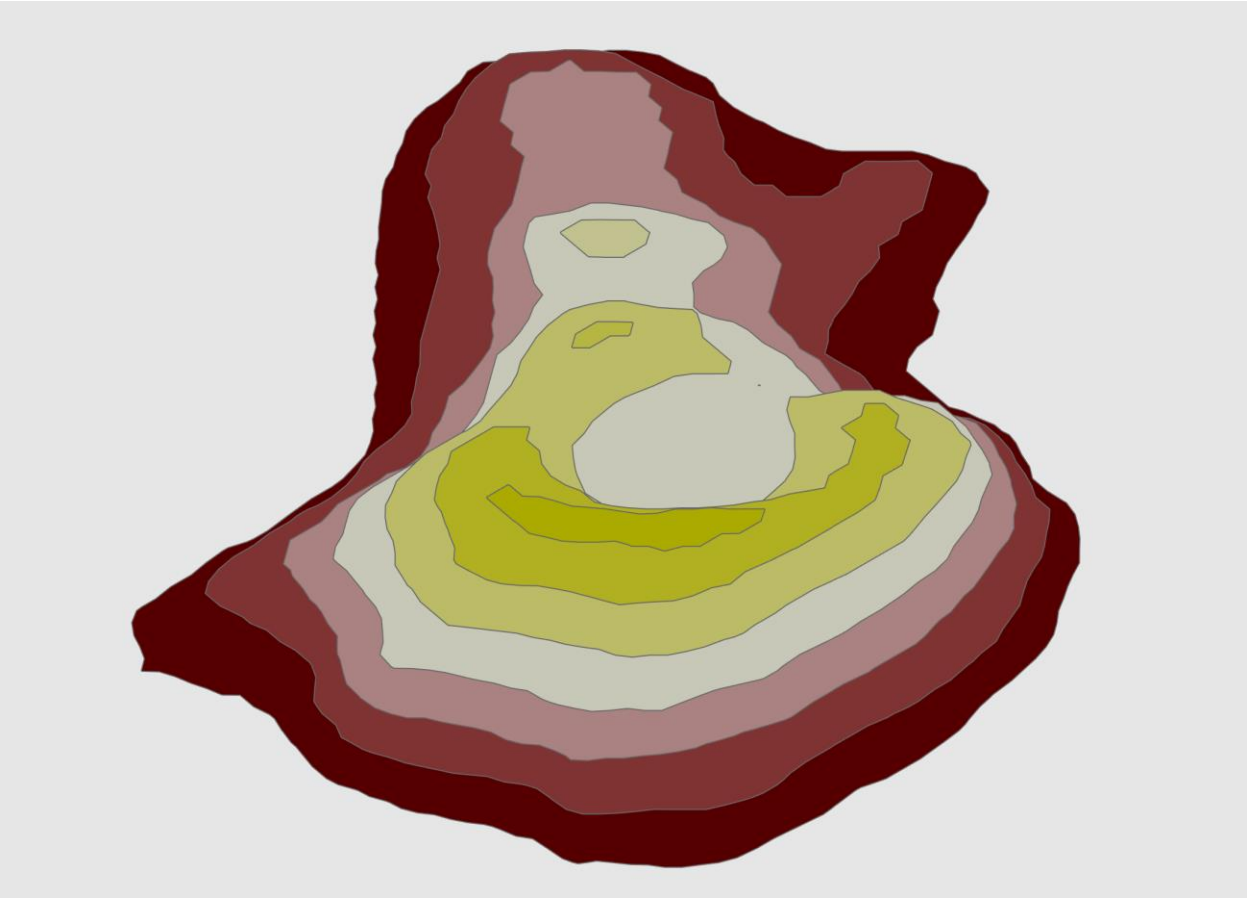
Background Information:

A contour chart is created by using R – ggplot2. The data set is costcos-geocoded. In this data set, the user can find the concentration of Costco stores in the US, where it is present the most and the least. After loading the data set, the columns of longitudes and latitudes are selected to plot. In this contour chart, the audience can see the highest concentration of stores based on the deep red color and the lowest concentration of stores based on the white color.

Method:

- Load the libraries and the dataset into a data frame
- Select the longitude and latitude column from the data frame
- Using ggplot2 package to plot
- Using the geom_density2d() and geom_point() to plot the data

Contour Chart – Tableau



Background Information:

A contour chart is created by using Tableau. The data set is caffeine molecule csv. In this data set, the user can see the concentration of atoms. The top of the image is dark yellow, which is the most concentrated location of atoms whereas the brown areas are where the atoms are more spreadout.

Method:

- Load data set from the data source
- Drag SUM(x_rotated) into the Rows tab
- Drag SUM(y_rotated) into the Columns tab
- Drag the id to the Detail tab
- Drag the NoName into the Path tab
- Drag the SUM(z) into the Color tab