COSC 3337 : Data Science I

N. Rizk

College of Natural and Applied Sciences

Department of Computer Science

University of Houston

Heatmap

What is a heatmap?

The heatmap is a way of representing the data in a 2-dimensional form. The data values are represented as colors in the graph. The goal of the heatmap is to provide a colored visual summary of information.
(rows features such as exam1, exam2…)
(column instances such as student1, student2)

Create a heatmap

To create a heatmap in Python, we can use the seaborn library. The seaborn library is built on top of Matplotlib. Seaborn library provides a high-level data visualization interface where we can draw our matrix.

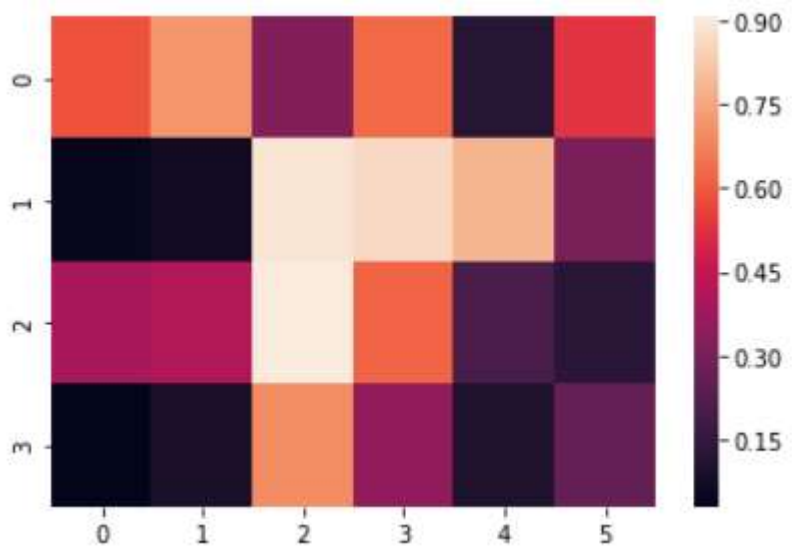
```
In [1]: import numpy as np

        import seaborn as sb

        import matplotlib.pyplot as plt
```

```
In [3]: data = np.random.rand(4, 6)
```

```
In [5]: heat_map = sb.heatmap(data)
        plt.show()
```

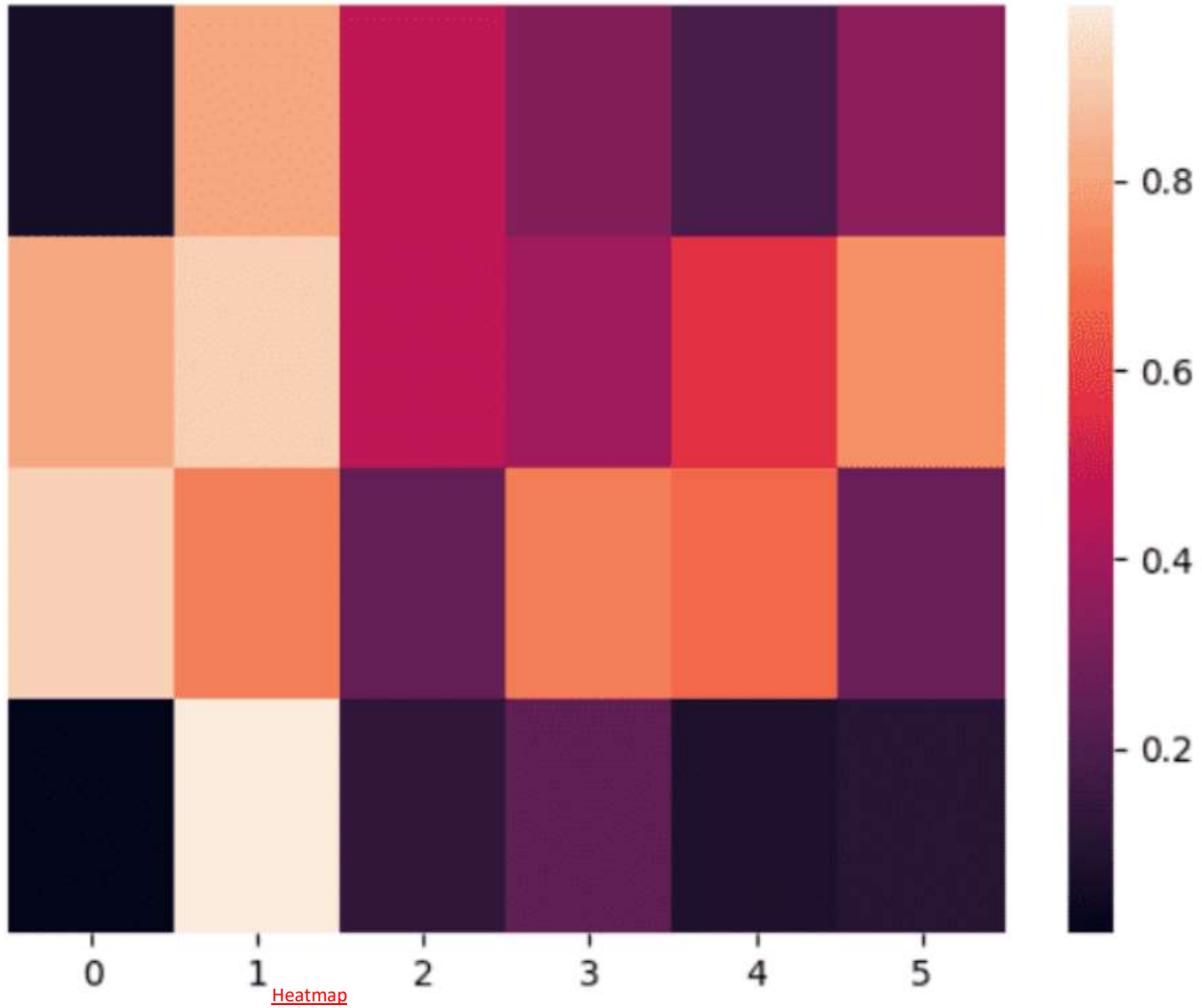


Remove heatmap x tick labels

```
heat_map = sb.heatmap(data, xticklabels=False, yticklabels=False)
```

Remove heatmap y tick labels

```
heat_map = sb.heatmap(data, yticklabels=False)
```



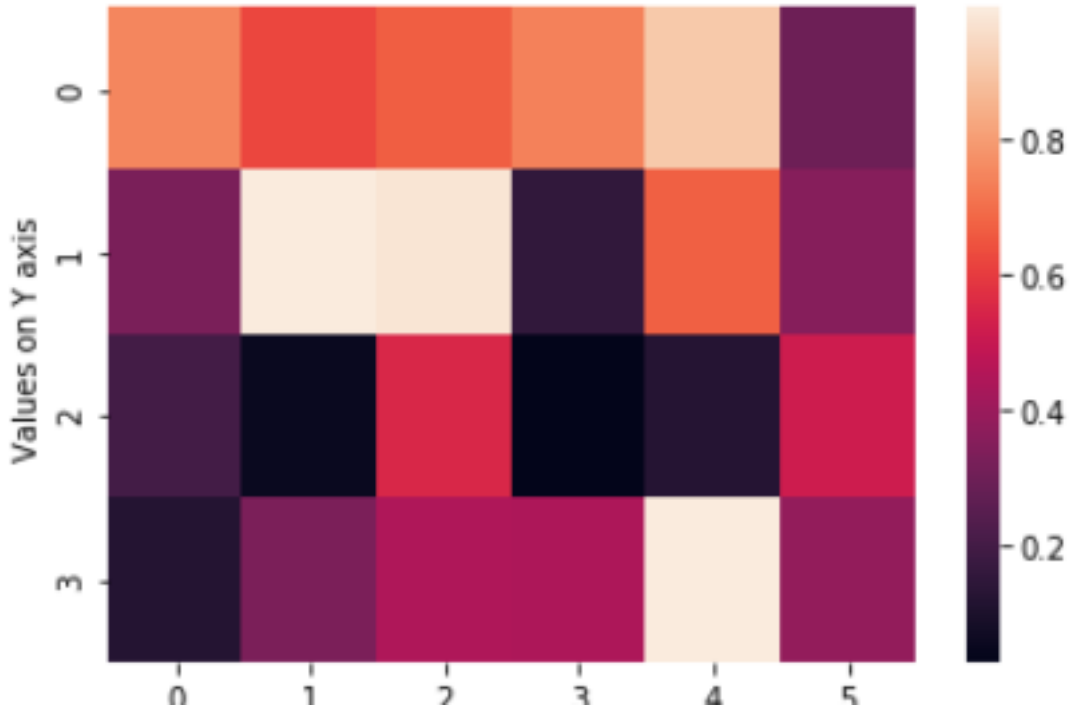
Heatmap

In [16]:
```
data = np.random.rand(4, 6)

heat_map = sb.heatmap(data)

plt.ylabel('Values on Y axis')
```

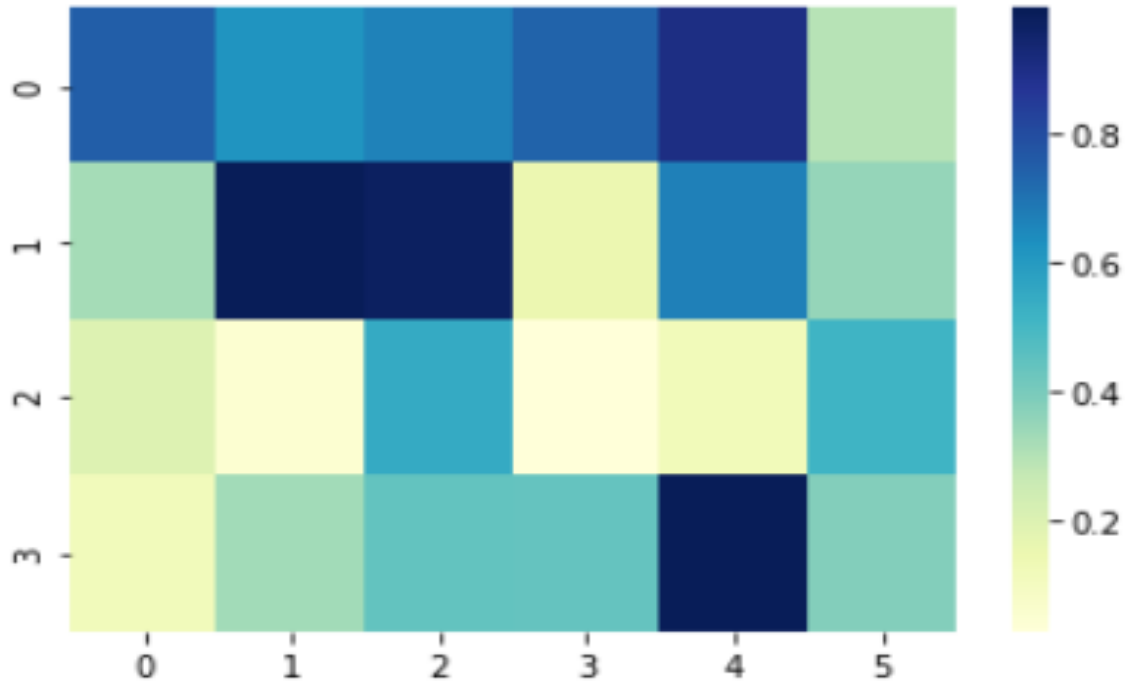
Out[16]: Text(33.0, 0.5, 'Values on Y axis')



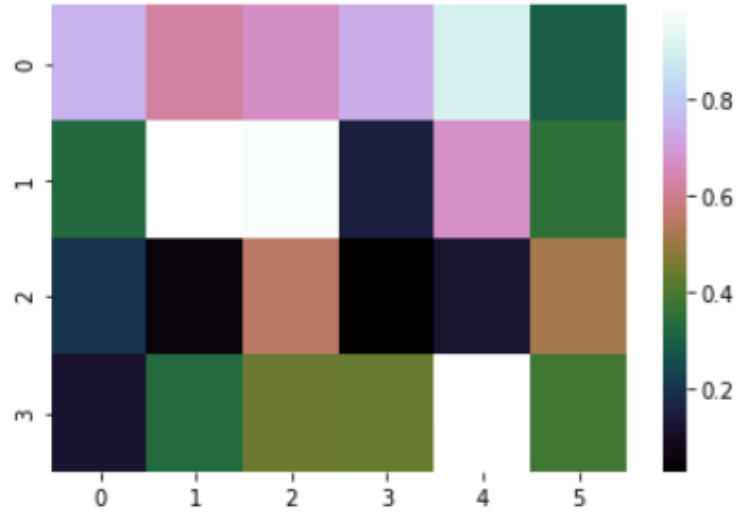
Changing heatmap color

```
heat_map = sb.heatmap(data, cmap="YlGnBu")

plt.show()
```



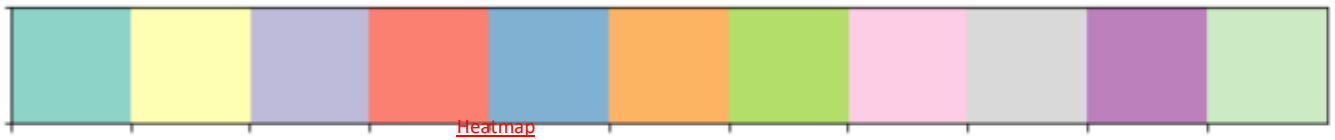
In [19]:
```
#Sequential cubehelix palette
heat_map = sb.heatmap(data, cmap="cubehelix")
```



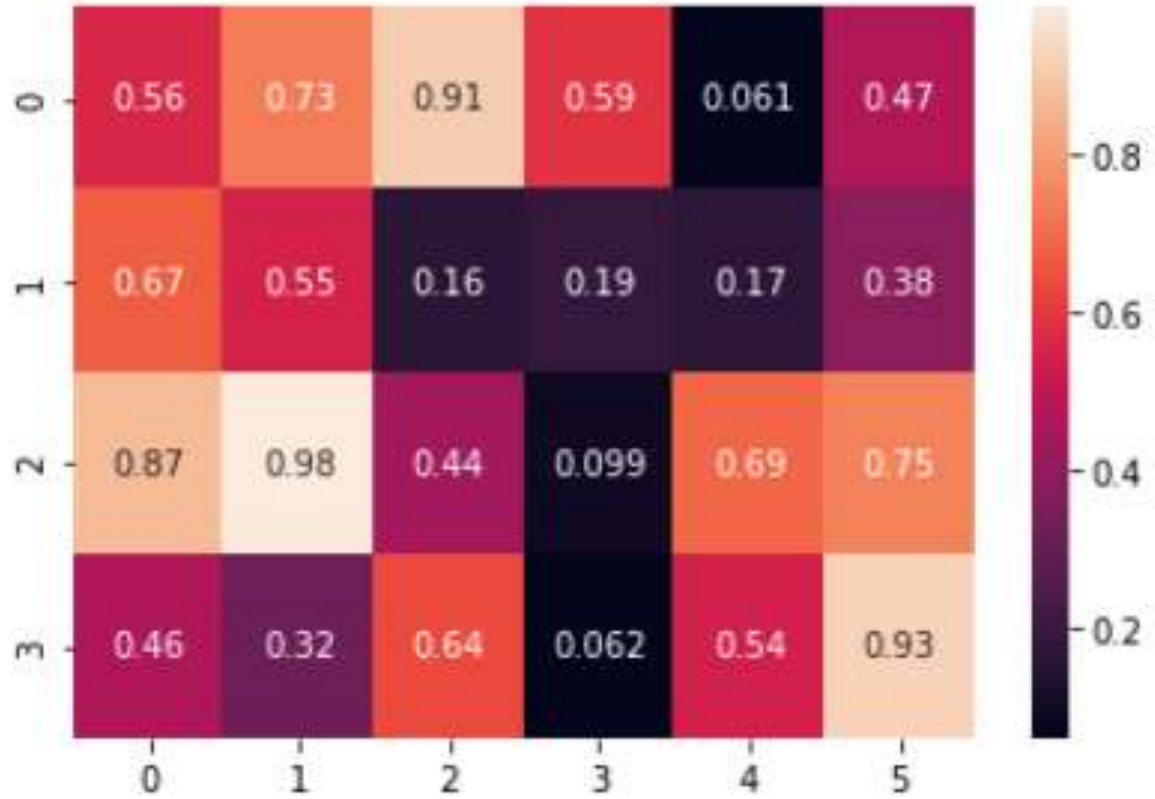
In [21]:
```
#Diverging color palette
import seaborn as sb

import matplotlib.pyplot as plt
sb.palplot(sb.diverging_palette(200, 100, n=11))
plt.show()
```

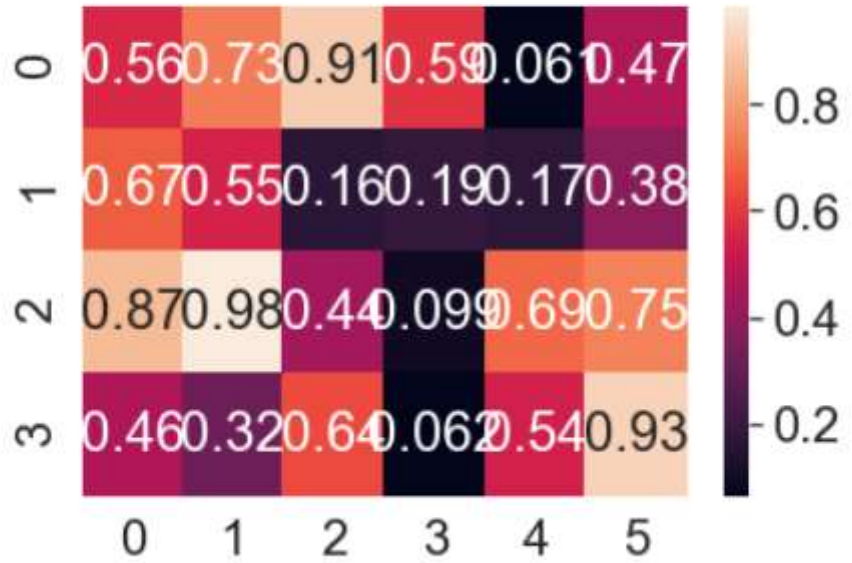
In [22]:
```
#discrete color palette
sb.palplot(sb.mpl_palette("Set3", 11))
plt.show()
```



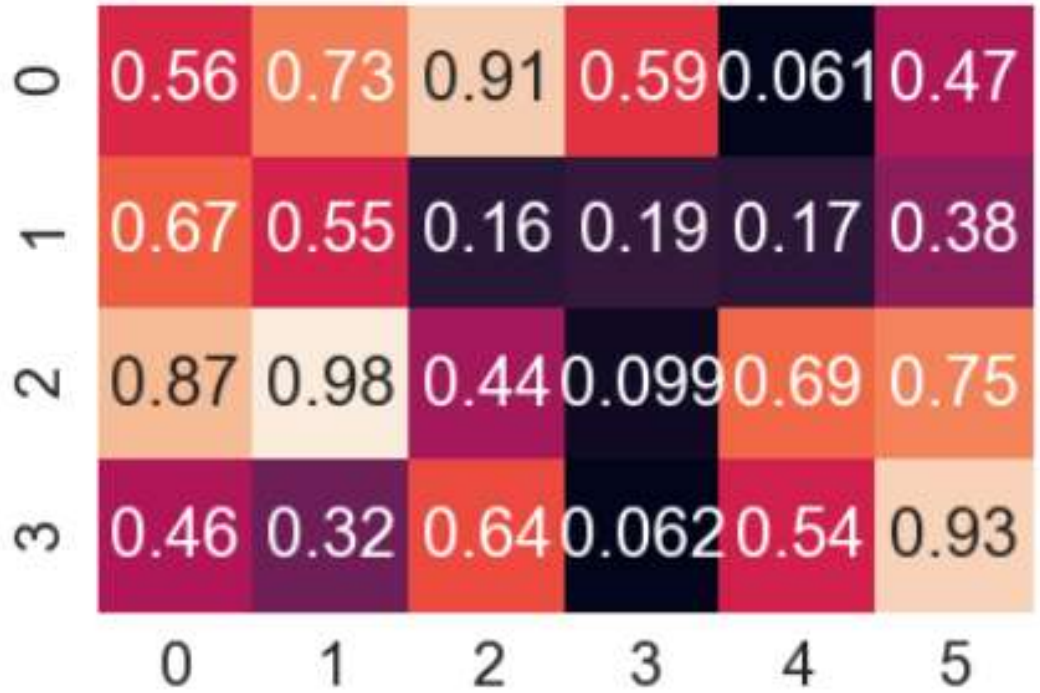
```
#add text over heatmap
data = np.random.rand(4, 6)
heat_map = sb.heatmap(data, annot=True)
plt.show()
```



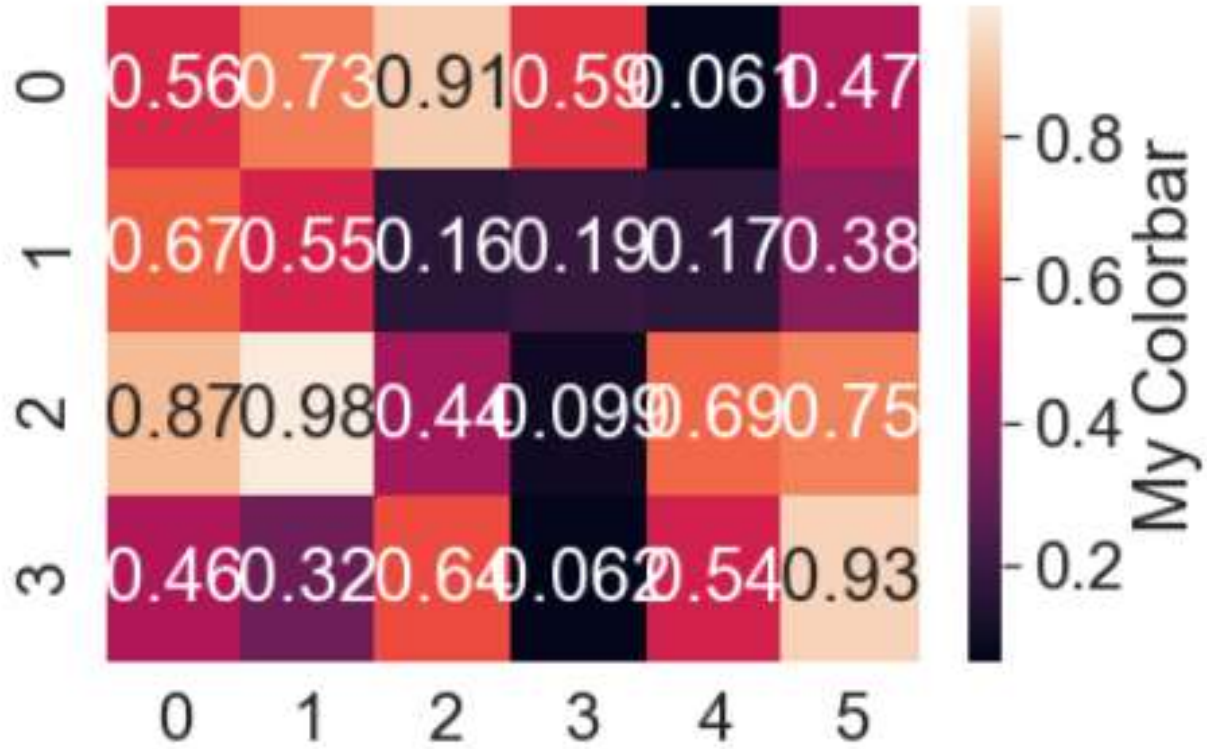
```
#Adjust heatmap font size
sb.set(font_scale=2)
heat_map = sb.heatmap(data, annot=True)
plt.show()
```



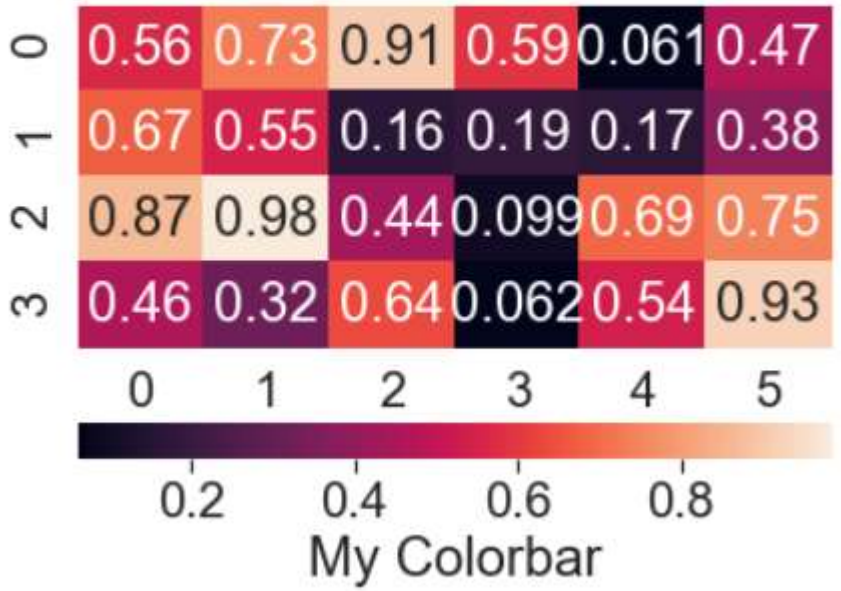
```
#Seaborn heatmap colorbar
heat_map = sb.heatmap(data, annot=True, cbar=False)
plt.show()
```



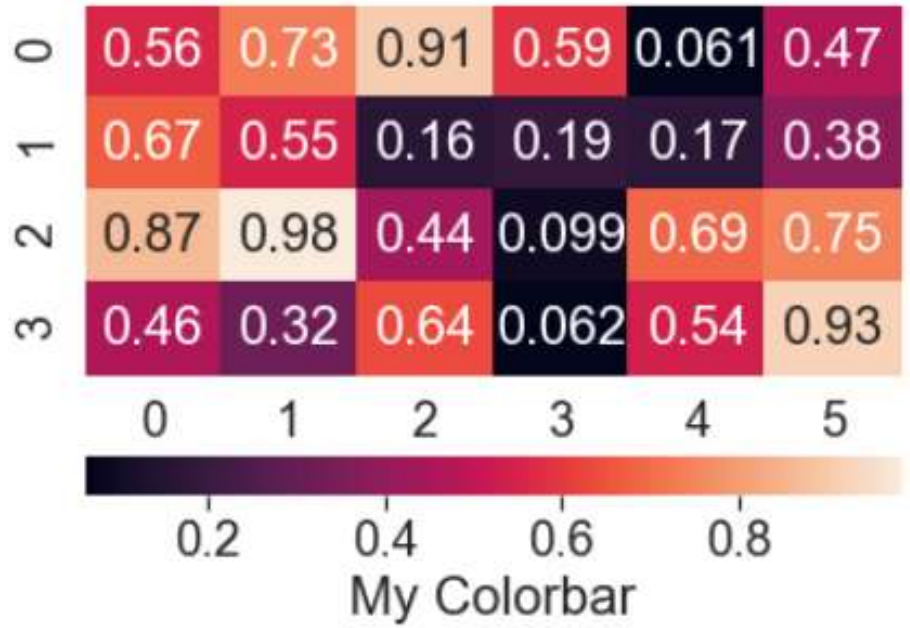
```
#add a color bar
heat_map = sb.heatmap(data, annot=True, cbar_kws={'label': 'My Colorbar'})
plt.show()
```



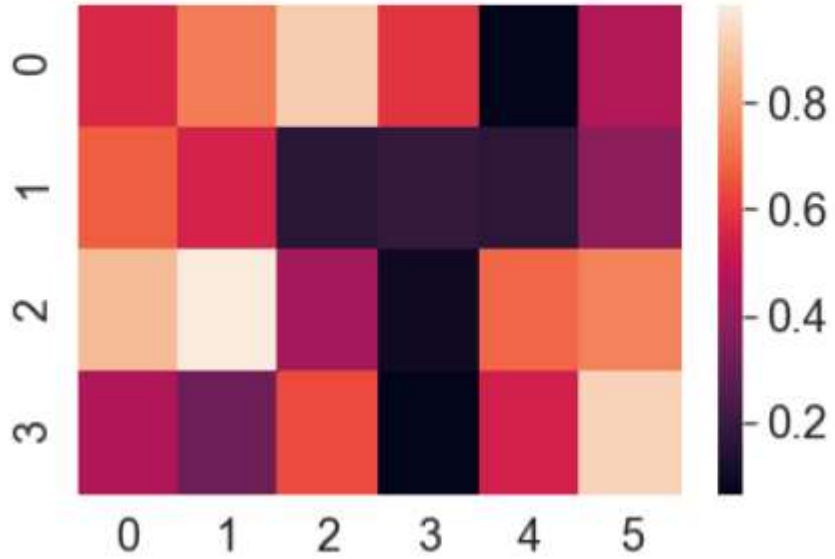
```
#horizontal bar
heat_map = sb.heatmap(data, annot=True, cbar_kws={'label': 'My Colorbar', 'orientation': 'horizontal'})
plt.show()
```



```
#Change heatmap colorbar font size
sb.set(font_scale=1.8)
heat_map = sb.heatmap(data, annot=True, cbar_kws={'label': 'My Colorbar', 'orientation': 'horizontal'})
plt.show()
```



```
#change rotation of x axis
heat_map = sb.heatmap(data)
plt.show()
```



```
heat_map.set_yticklabels(heat_map.get_yticklabels(), rotation=0)
```

```
]: [Text(0, 0.5, '0'), Text(0, 1.5, '1'), Text(0, 2.5, '2'), Text(0, 3.5, '3')]
```

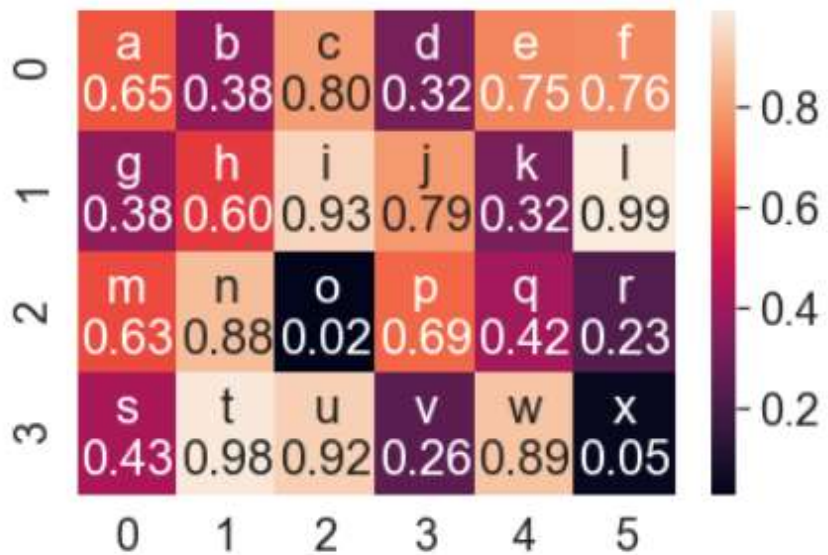
```
heat_map.set_yticklabels(heat_map.get_yticklabels(), rotation=35)
```

```
]: [Text(0, 0.5, '0'), Text(0, 1.5, '1'), Text(0, 2.5, '2'), Text(0, 3.5, '3')]
```

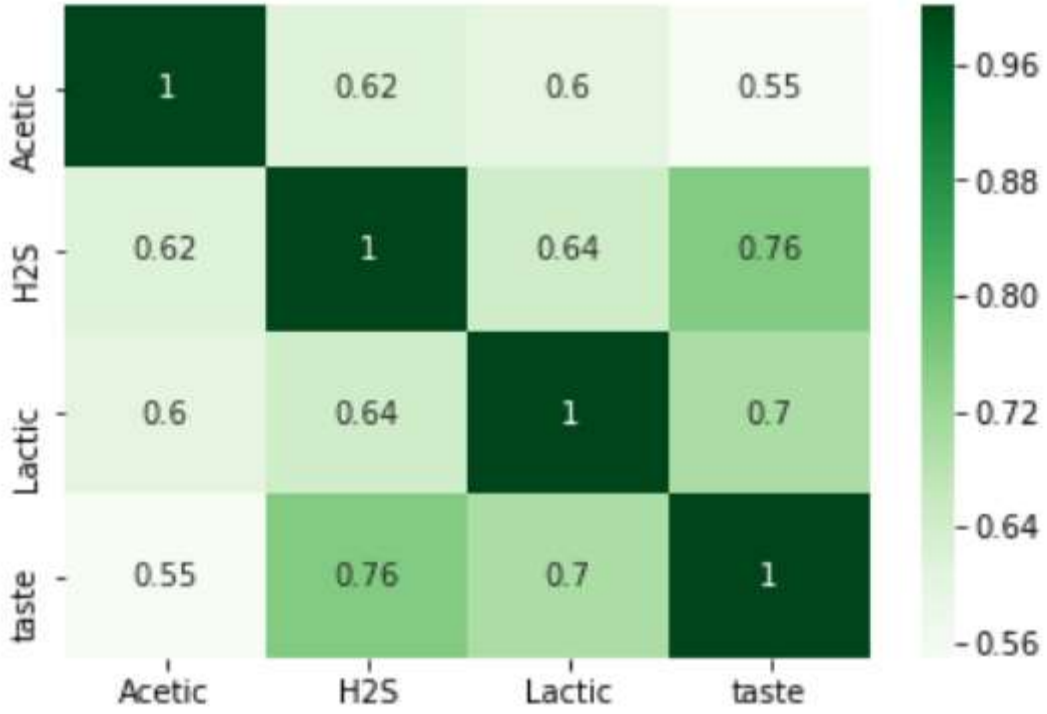
```
#Add text and values on the heatmap
data = np.random.rand(4, 6)
text = np.asarray([['a', 'b', 'c', 'd', 'e', 'f'], ['g', 'h', 'i', 'j', 'k', 'l'], ['m', 'n', 'o', 'p', 'q', 'r'], ['s', 't'
```

```
labels = (np.asarray(["{0}\n{1:.2f}".format(text,data) for text, data in zip(text.flatten(), data.flatten())])).reshape(4,6)
```

```
heat_map = sb.heatmap(data, annot=labels, fmt='')
```



Heatmap



Each square shows the correlation between the variables on each axis.

Correlation ranges from -1 to +1. Values closer to zero means there is no linear trend between the two variables. The close to 1 the correlation is the more positively correlated they are;

that is as one increases so does the other and the closer to 1 the stronger this relationship is.

A correlation closer to -1 is similar, but instead of both increasing one variable will decrease as the other increases.

The diagonals are all 1/dark green because those squares are correlating each variable to itself (so it's a perfect correlation).

For the rest the larger the number and darker the color the higher the correlation between the two variables.

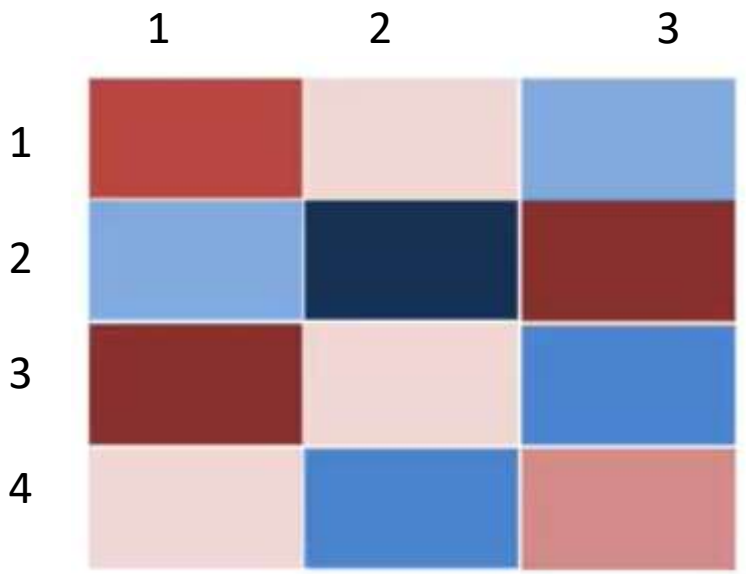
The plot is also symmetrical about the diagonal since the same two variables are being paired together in those squares.

Hierarchical clustering reorder rows (features) and columns (samples)

based on their similarities

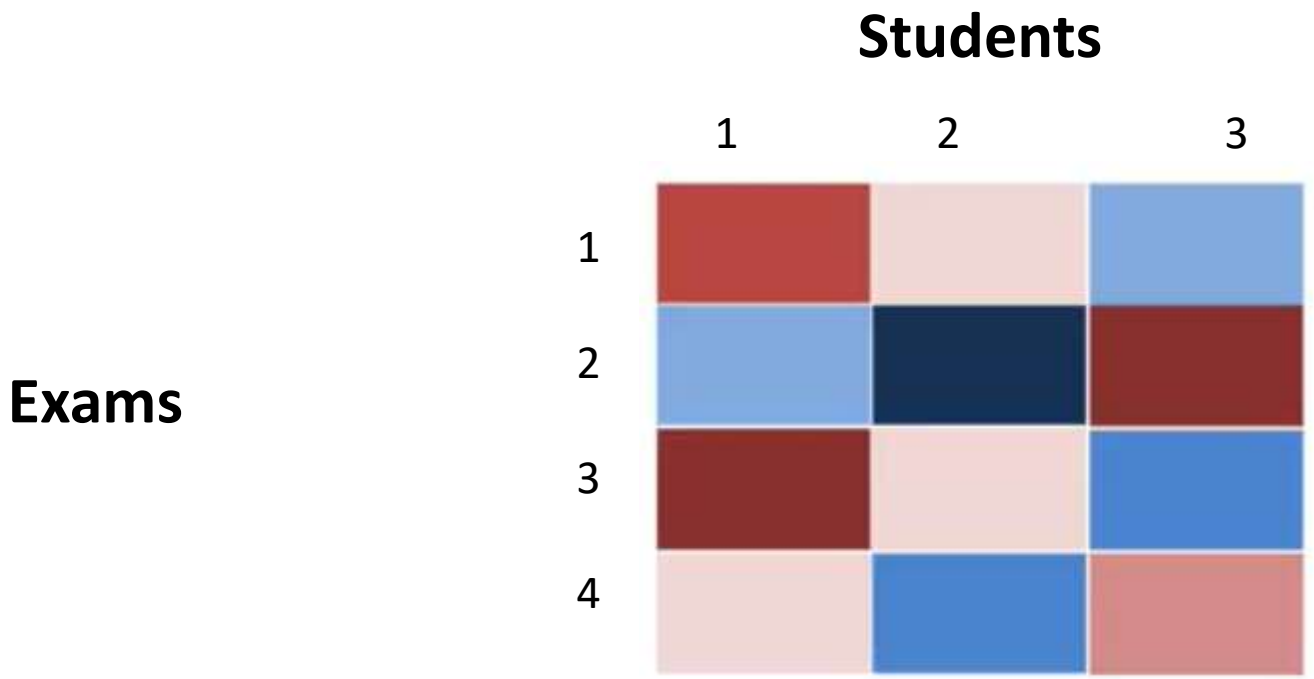
Students

Exams

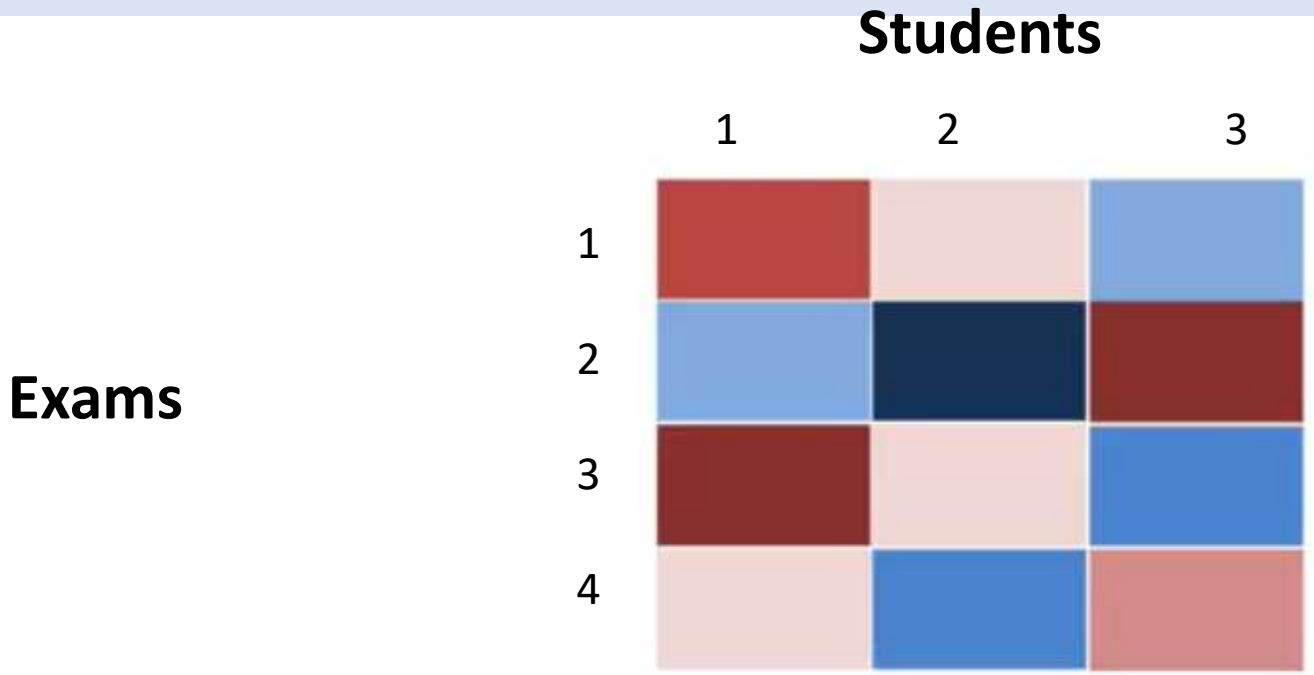


How to reorder exams based on the similarity?

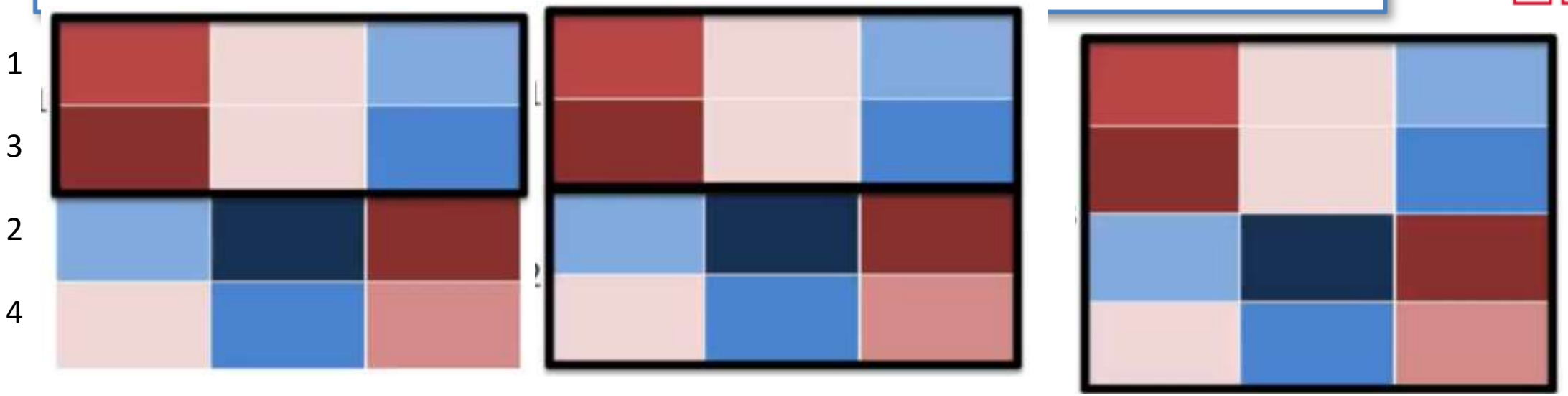
Exam1 and Exam 3 are similar (highly expressed for student 1 and lowly expressed for student 3

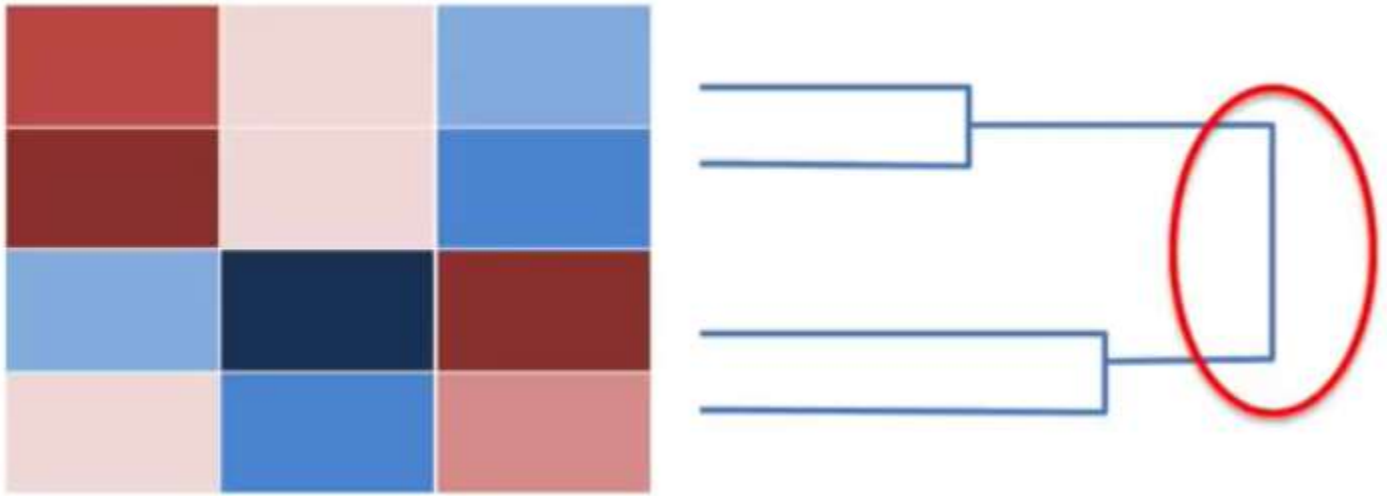


Exam2 and Exam 4 are similar
Exam3 and ……
Exam4 and ……..



Forming Clusters

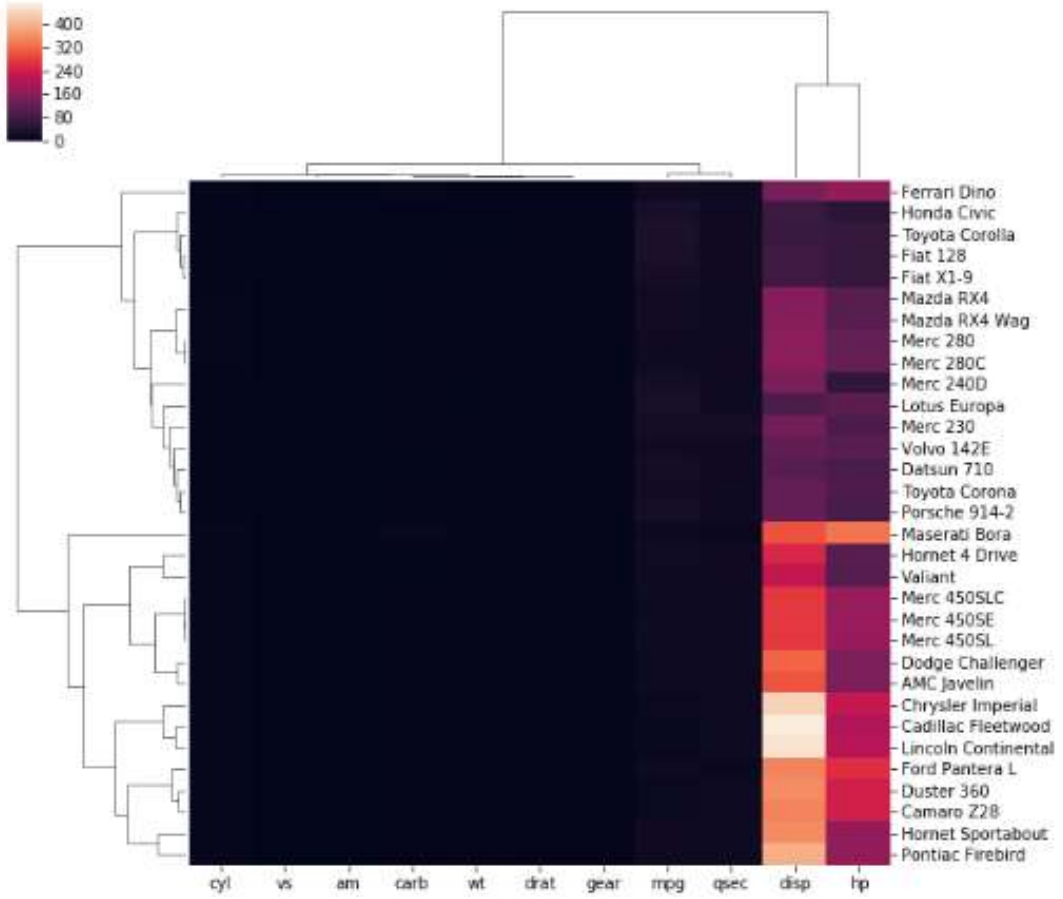




Heatmap

Dendrogram

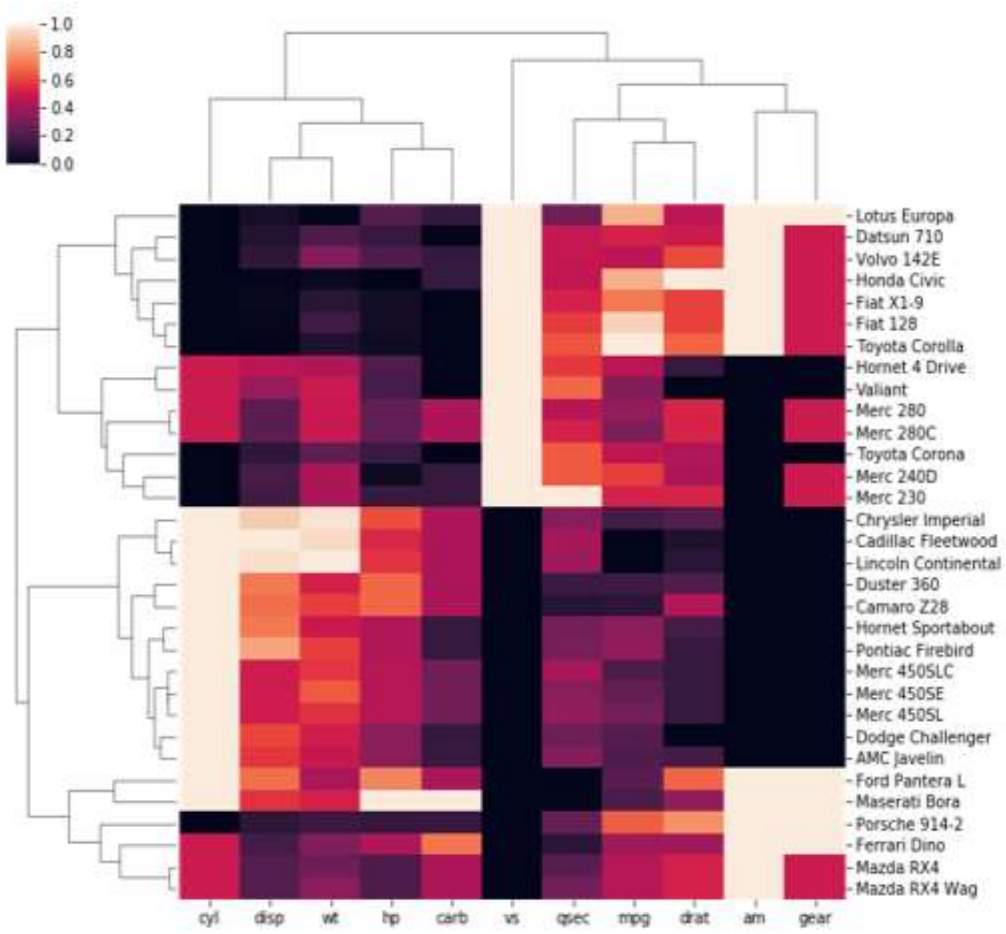
```
# Libraries
import seaborn as sns
import pandas as pd
from matplotlib import pyplot as plt
# Data set
url = 'https://python-graph-gallery.com/wp-content/uploads/mtcars.csv'
df = pd.read_csv(url)
df = df.set_index('model')
del df.index.name
df
# Default plot
sns.clustermap(df)
plt.show()
```

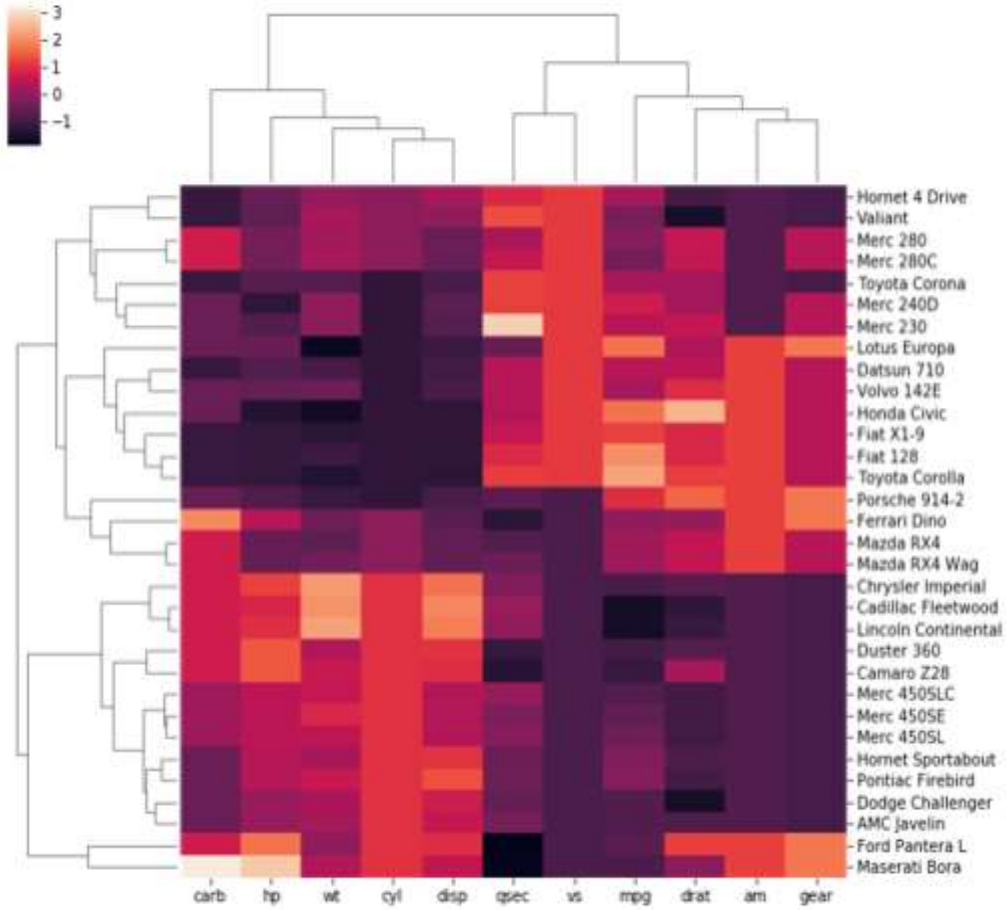


Heatmap

# Standardize:
sns.clustermap(df, standard_scale=1)

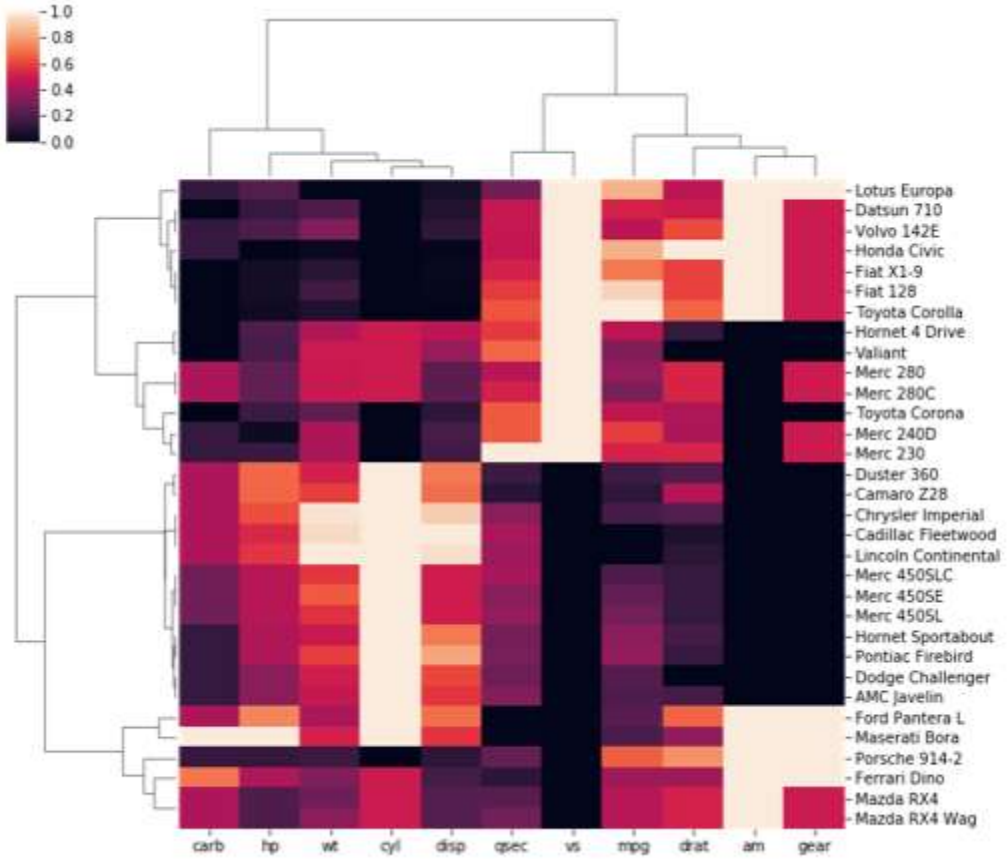
# Normalize
sns.clustermap(df, z_score=1)

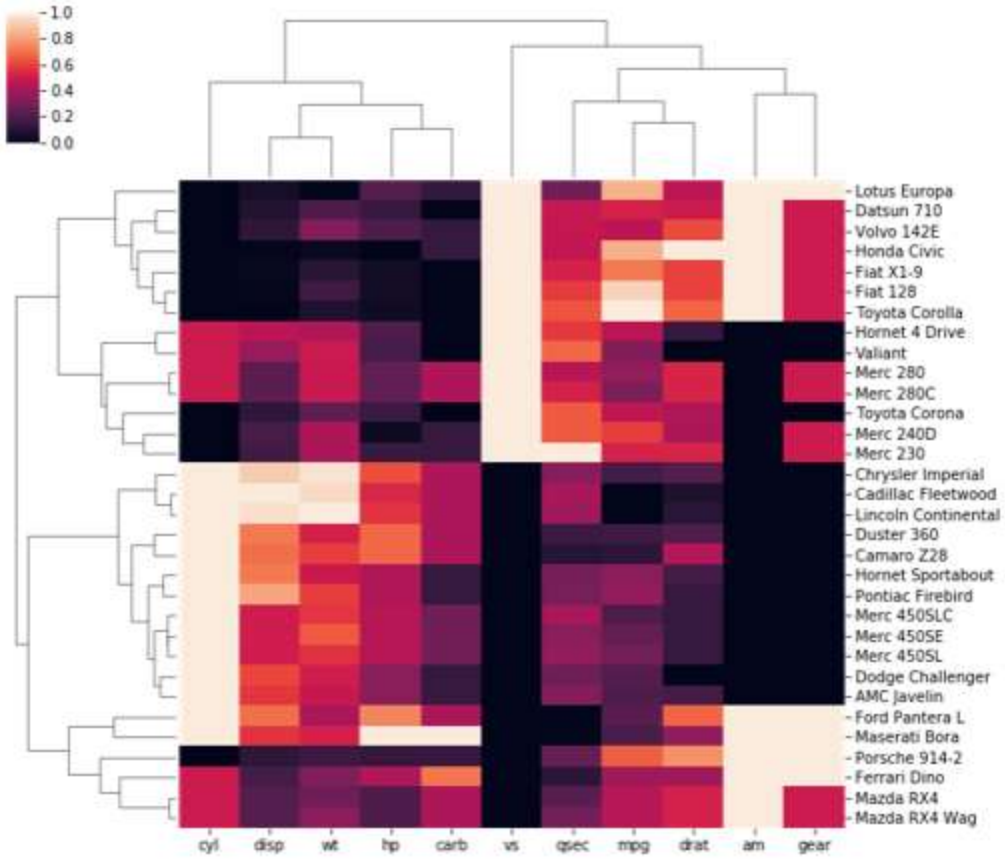




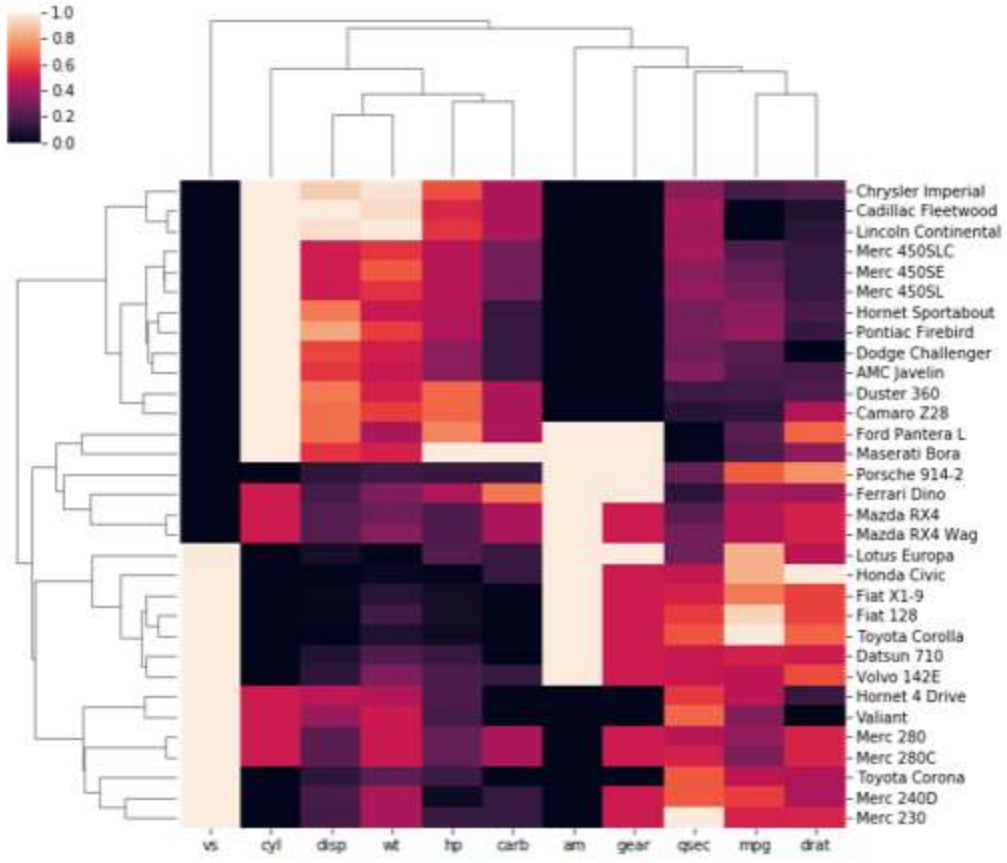
correlation and euclidean distance?

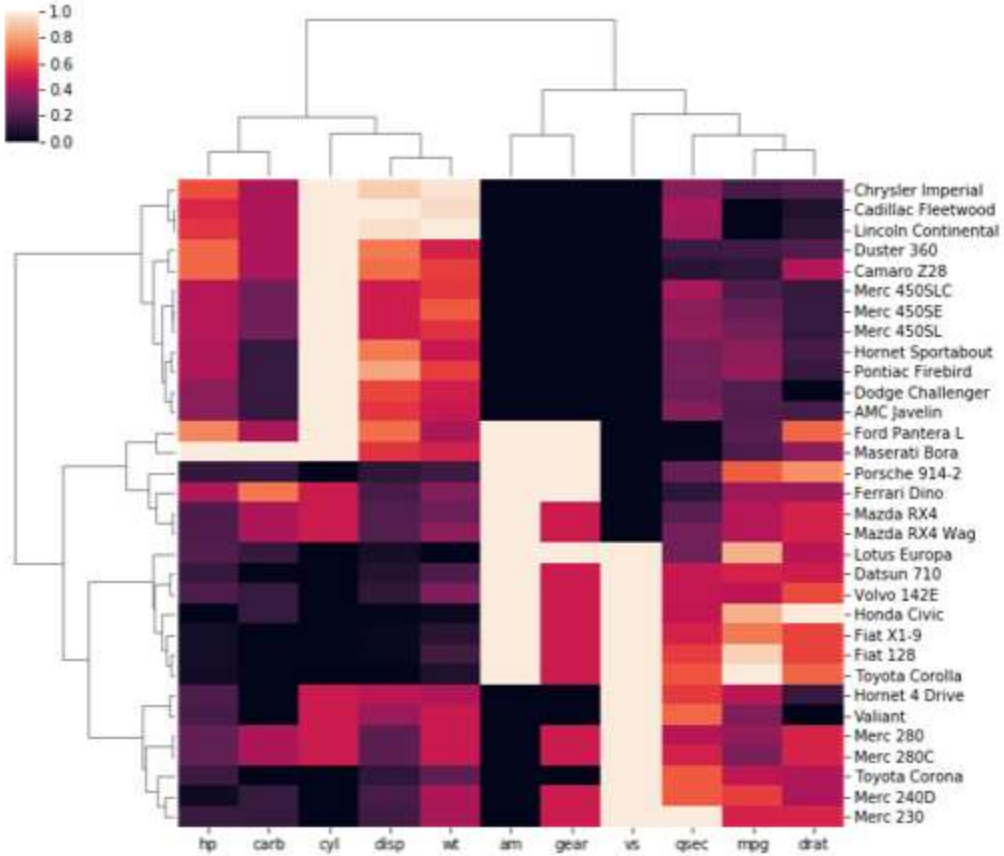
```
sns.clustermap(df, metric="correlation", standard_scale=1)
sns.clustermap(df, metric="euclidean", standard_scale=1)
```



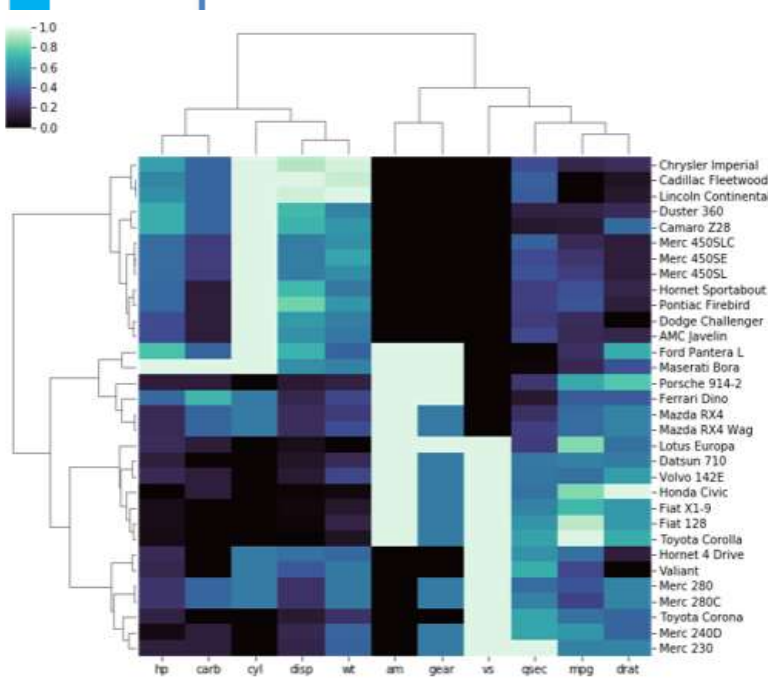


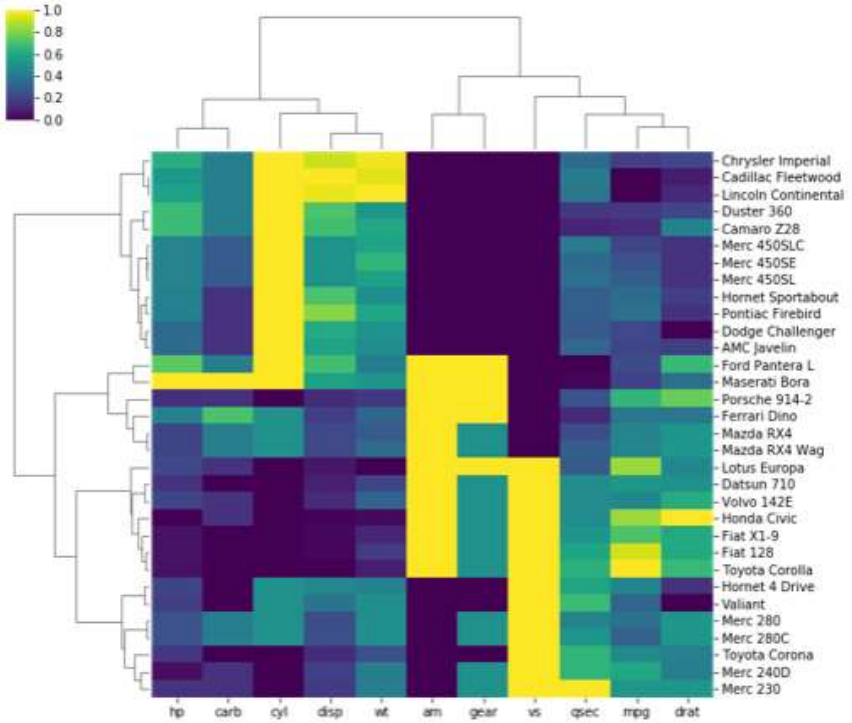
```
sns.clustermap(df, metric="euclidean", standard_scale=1, method="single")
sns.clustermap(df, metric="euclidean", standard_scale=1, method="ward“)
```

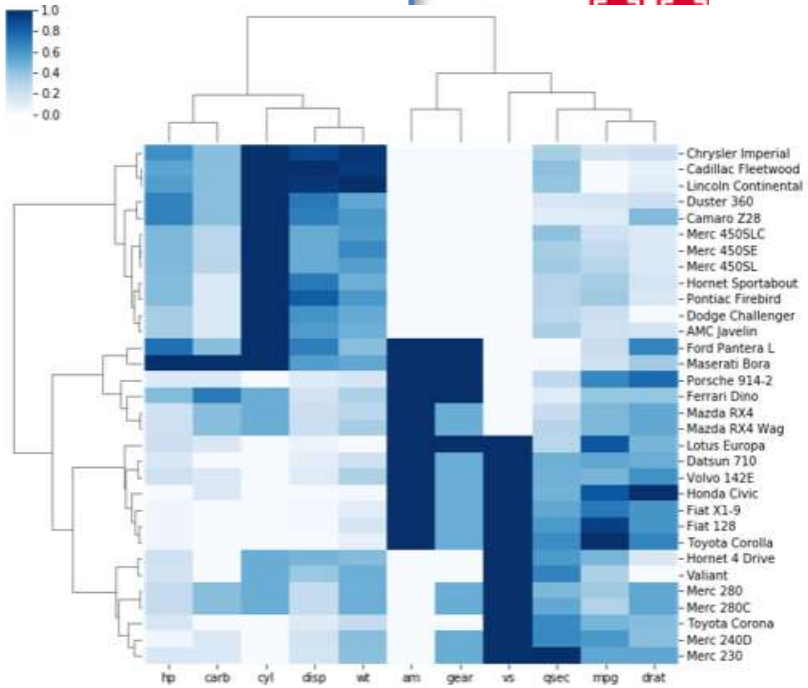




sns.clustermap(df, metric="euclidean", standard_scale=1, method="ward", cmap="mako") “viridis“ “Blues"







use the outlier detection sns.clustermap(df, robust=True) , robust=False)