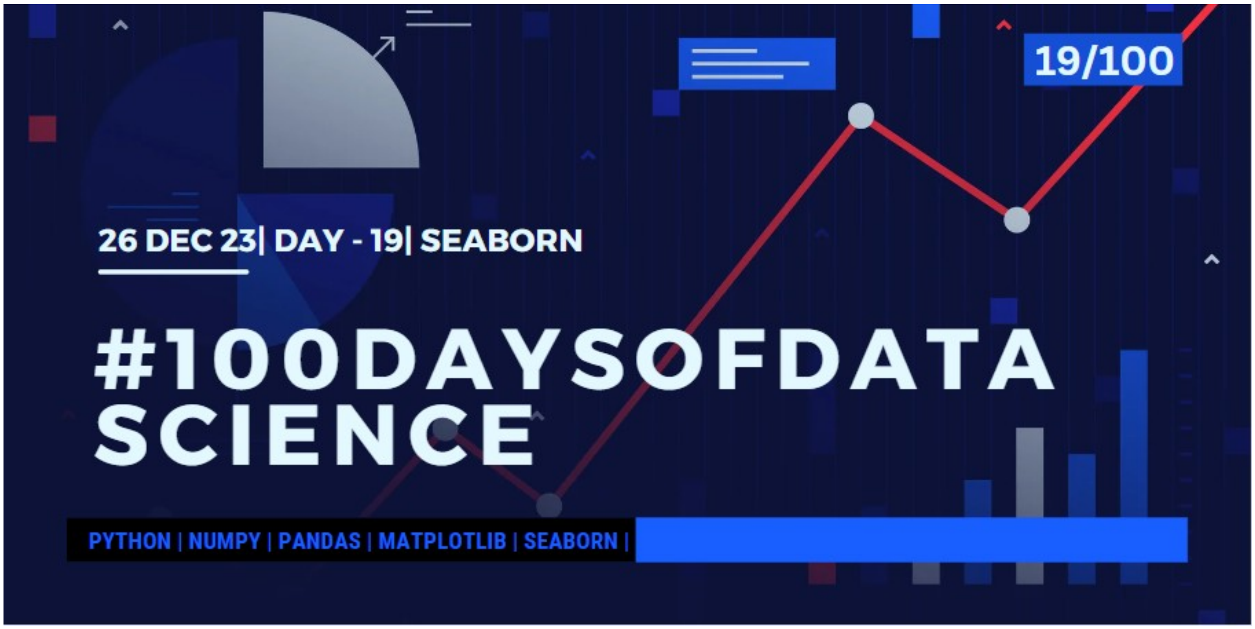

19/100
26 DEC 23| DAY - 19| SEABORN
#100DAYSOFDATA SCIENCE
PYTHON | NUMPY | PANDAS | MATPLOTLIB | SEABORN |

```
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

iris = sns.load_dataset("iris")
```

Distribution Plots

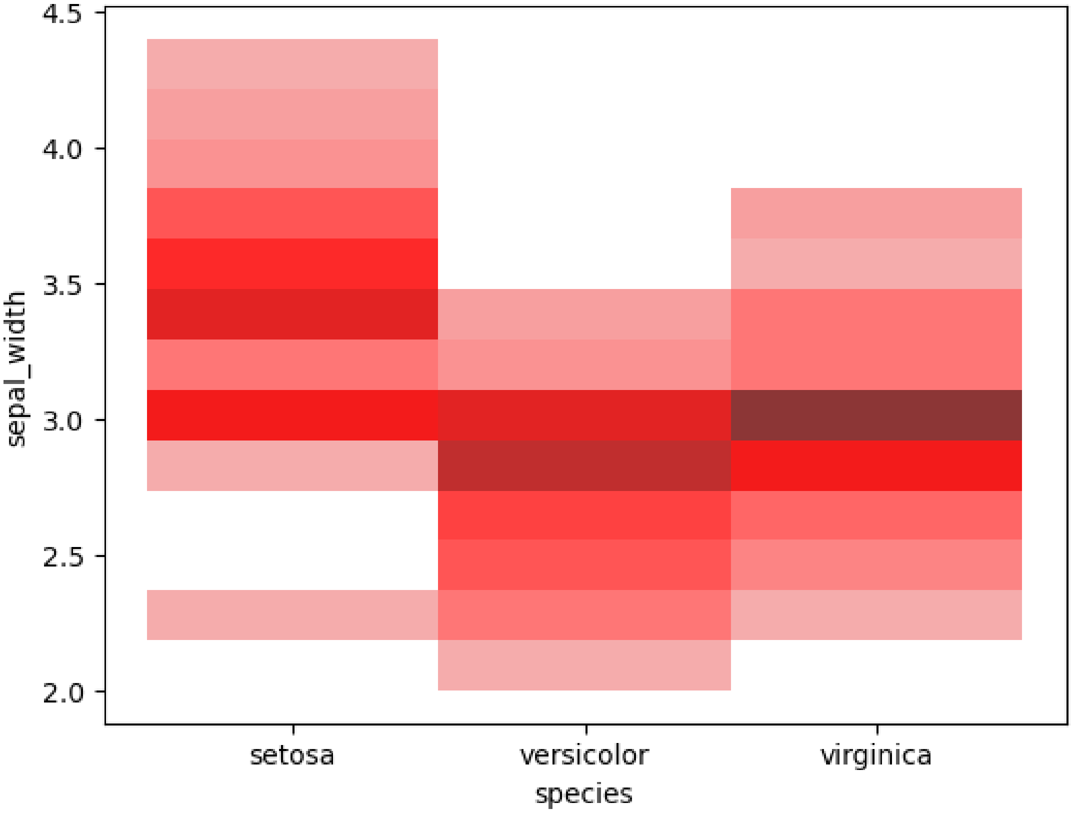
- Distribution Plots are used for examining univariate and bivariate distributions meaning such distributions that involve one variable or two discrete variables.
- There are various types of categorical plots:
  a. Histogram
  b. Distplot
  c. Pairplot
  d. KDE Plot

1. Histogram:

- A histogram is basically used to represent data provided in a form of some groups.
- It is accurate method for the graphical representation of numerical data distribution.
- It can be plotted using the histplot() function.
- Syntax:

- `histplot(data=None, *, x=None, y=None, hue=None,  **kwargs)`

```
sns.histplot(x='species', y='sepal_width', data=iris, color='red')
plt.show()
```



2. Distplot:

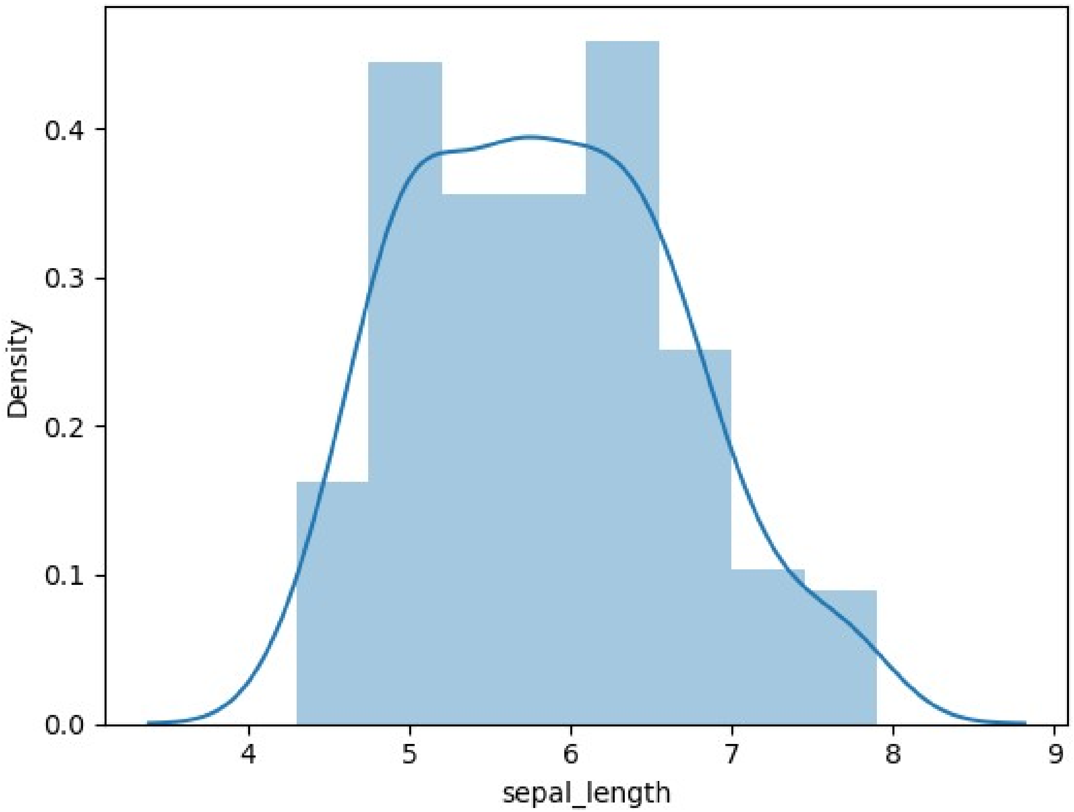
- Distplot is used basically for univariant set of observations and visualizes it through a histogram i.e. only one observation and hence we choose one particular column of the dataset.

- It is potted using the distplot() method.

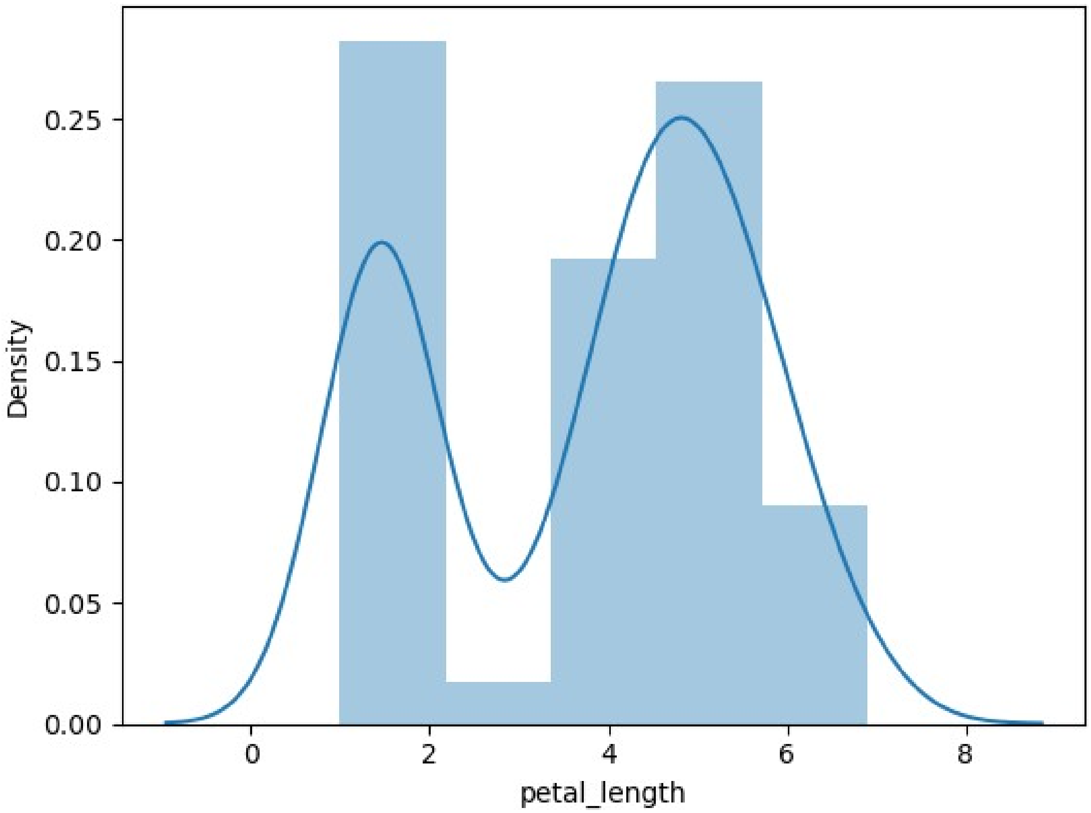
- Syntax:

  - `distplot(a[, bins, hist, kde, rug, fit, …])`

```
sns.distplot(iris['sepal_length'])
plt.show()
```



```
sns.distplot(iris['petal_length'])
plt.show()
```

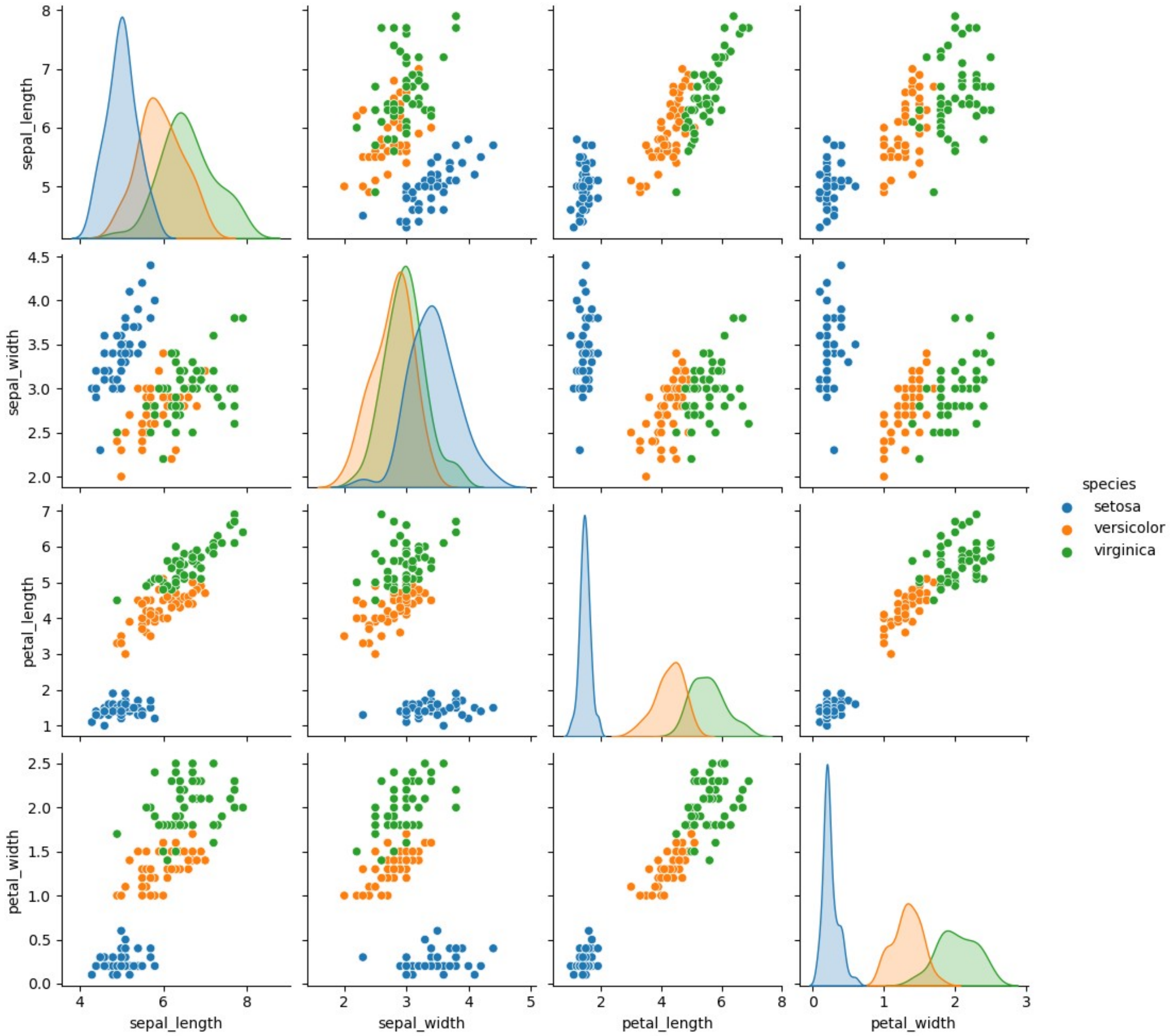


3. Pairplot:

- Pairplot represents pairwise relation across the entire dataframe and supports an additional argument called hue for categorical separation.
- It is plotted using the pairplot() method.
- Syntax:
  - `pairplot(data[, hue, hue_order, palette, …])`

```
sns.pairplot(data=iris, hue='species')
```

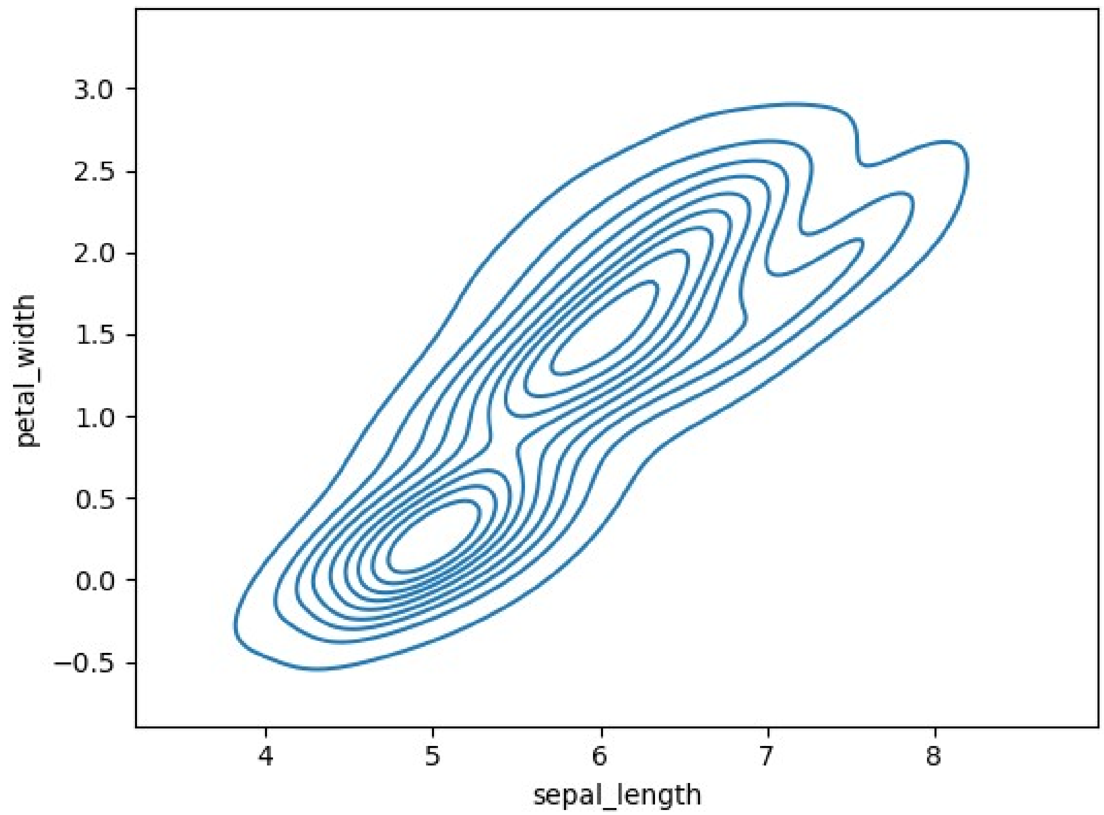
```
<seaborn.axisgrid.PairGrid at 0x1f8b8006510>
```



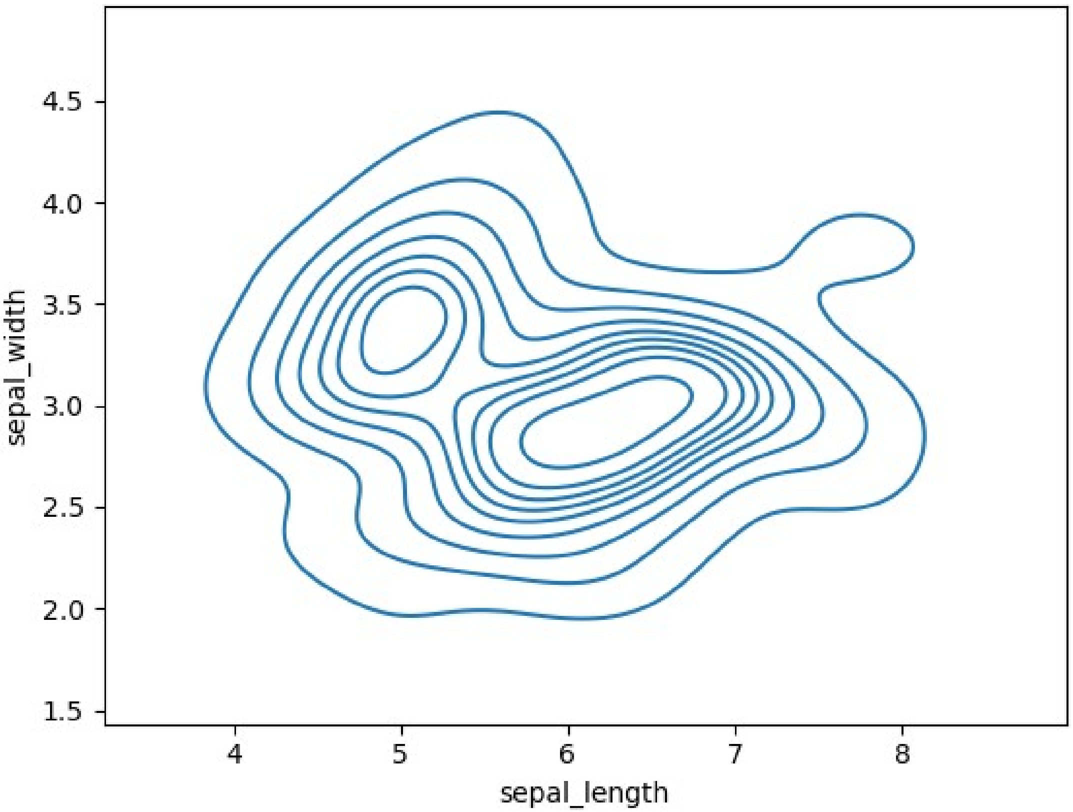
4. KDE Plot:

- KDE Plot described as Kernel Density Estimate is used for visualizing the Probability Density of a continuous variable.
- It depicts the probability density at different values in a continuous variable.
- We can also plot a single graph for multiple samples which helps in more efficient data visualization.
- Syntax:
  - `seaborn.kdeplot(x=None, *, y=None, vertical=False, palette=None, **kwargs)`

```
sns.kdeplot(x='sepal_length', y='petal_width', data=iris)
plt.show()
```



```
sns.kdeplot(x='sepal_length', y='sepal_width', data=data)
plt.show()
```



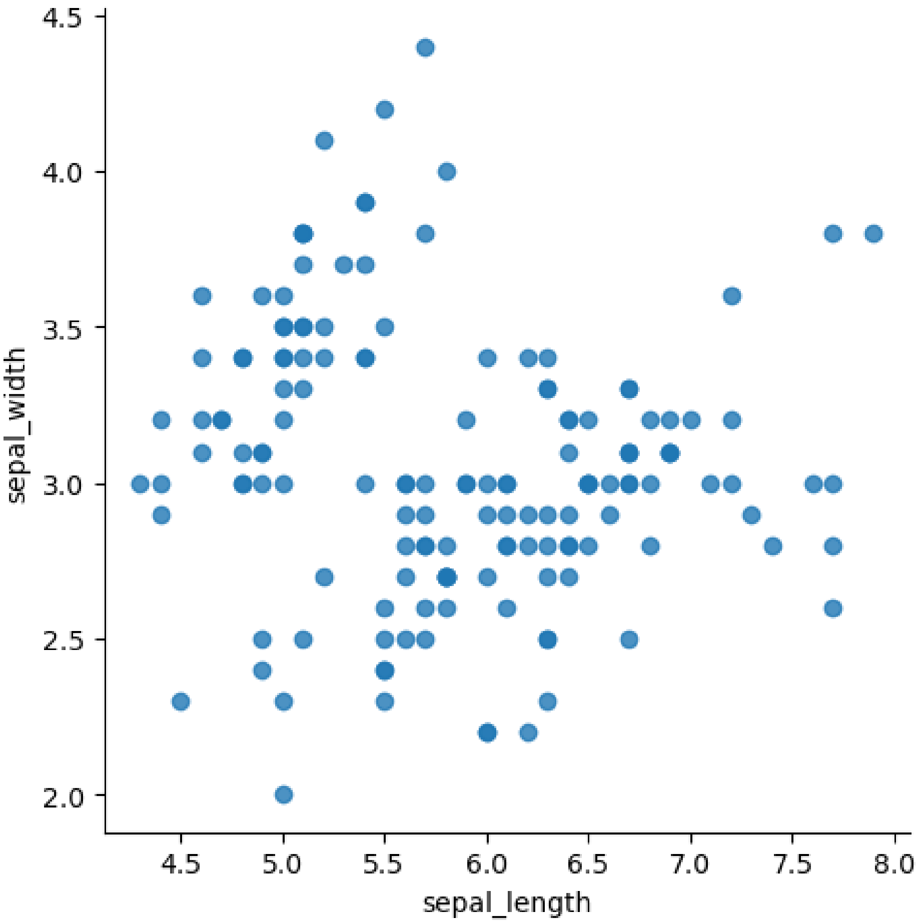
Regression Plots

- Regression plots as the name suggests creates a regression line between two parameters and helps to visualize their linear relationships.
- There are various types of categorical plots:
    a. lmplot
    b. Regplot

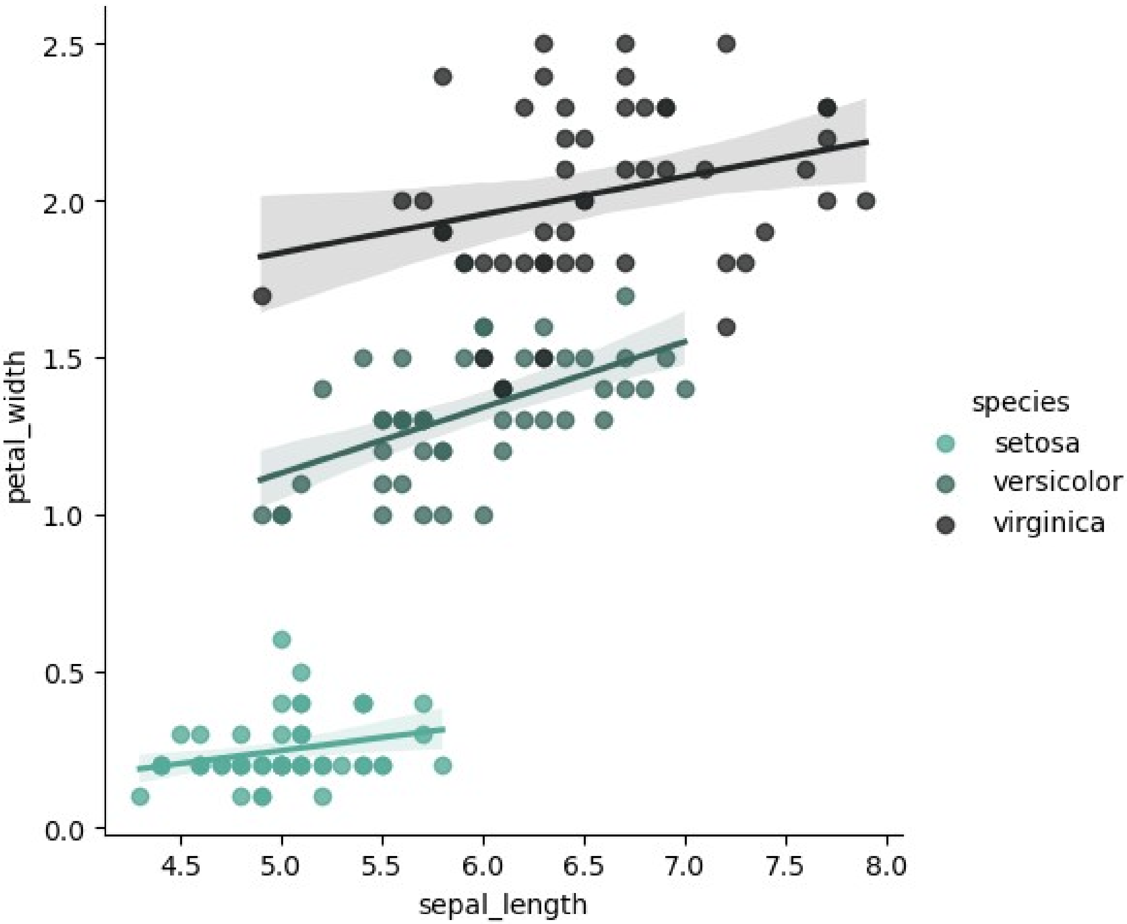
1. lmplot:

- lmplot() method can be understood as a function that basically creates a linear model plot.
- It creates a scatter plot with a linear fit on top of it.
- Syntax:
    – `seaborn.lmplot(x, y, data, hue=None, col=None, row=None, **kwargs)`

```
sns.lmplot(x='sepal_length', y='sepal_width', data=iris,
fit_reg=False)
plt.show()
```



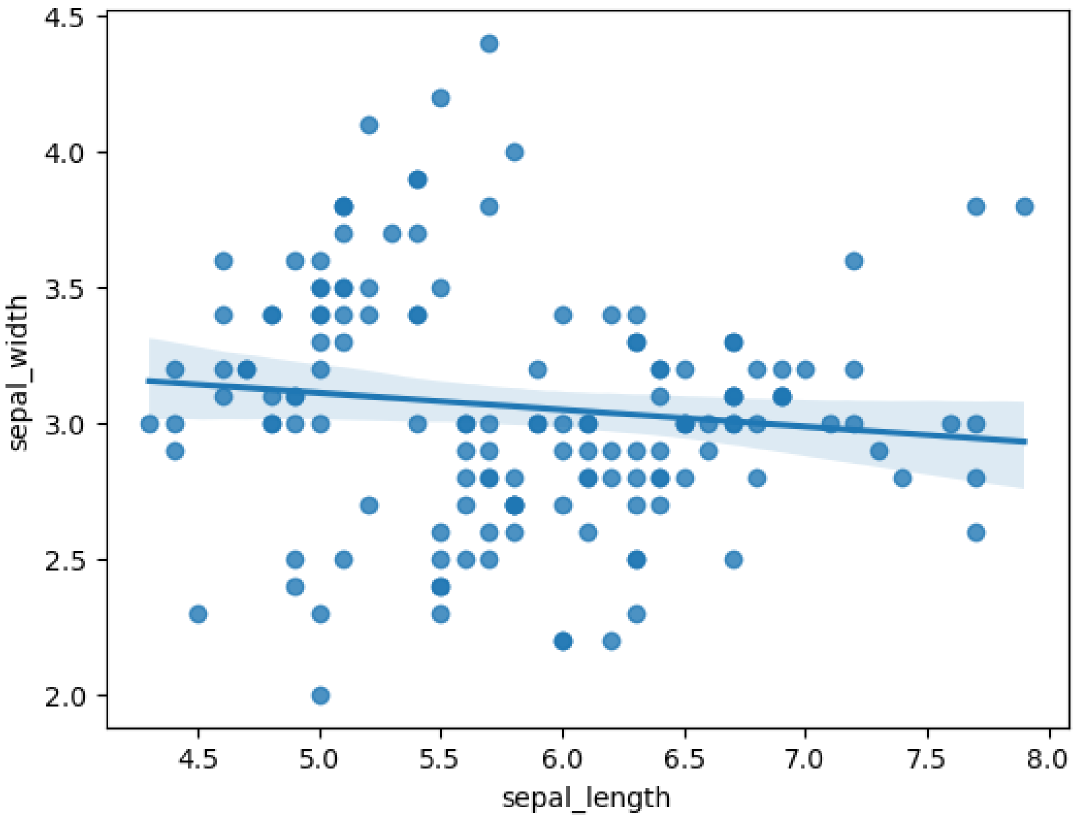
```
sns.lmplot(x='sepal_length', y='petal_width', data=iris,
hue='species', palette='dark:#5A9_r')
plt.show()
```



2. Regplot:

- regplot() method is also similar to lmplot which creates linear regression model.
- Syntax:
  - `seaborn.regplot( x,  y,  data=None, x_estimator=None, **kwargs)`

```
sns.regplot(x='sepal_length', y='sepal_width', data=iris)
plt.show()
```



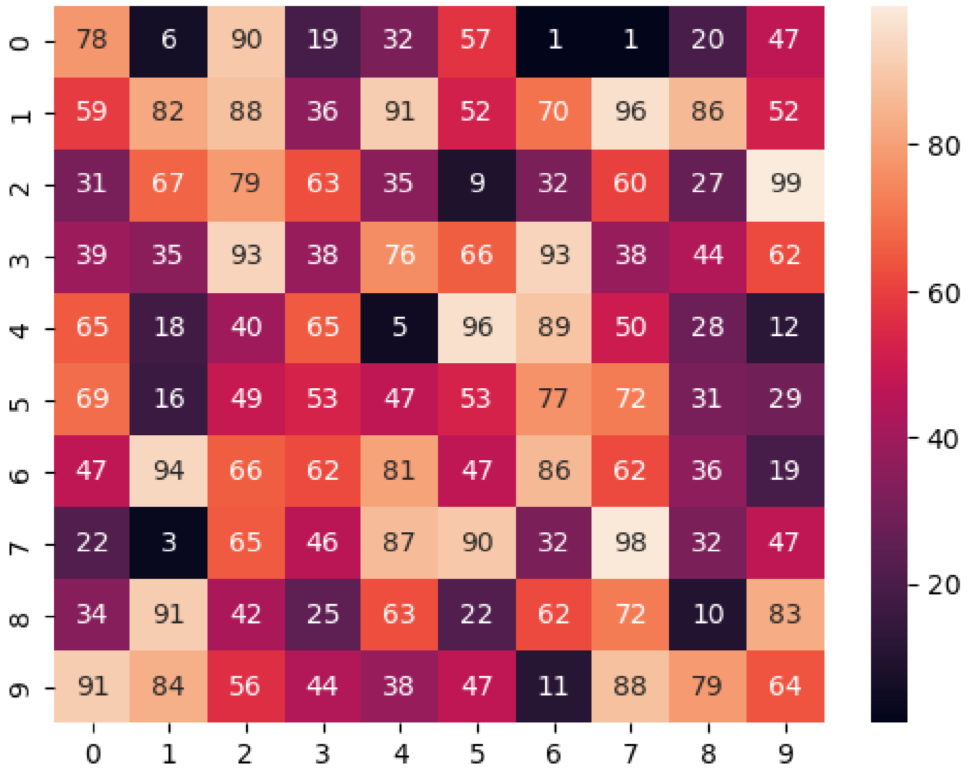
Matrix Plots

- A matrix plot means plotting matrix data, where color coded diagrams shows rows data, column data and values.
- It can shown using the heatmap and clustermap.
- There are various types of categorical plots:
    a. Heatmap
    b. Clustermap

1. Heatmap:

- Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix.
- In this, to represent more common values or higher activities brighter colors basically reddish colors are used and to represent less common or activity values, darker colors are preferred.
- It can be plotted using the heatmap() function.
- Syntax:
    - `seaborn.heatmap(data, *, vmin=None, vmax=None, cmap=None, linecolor=’white’, cbar=True, **kwargs)`

```
import numpy as np
import pandas as pd
data = np.random.randint(low=1, high=100, size=(10, 10))
hm = sns.heatmap(data=data, annot=True)
plt.show()
```

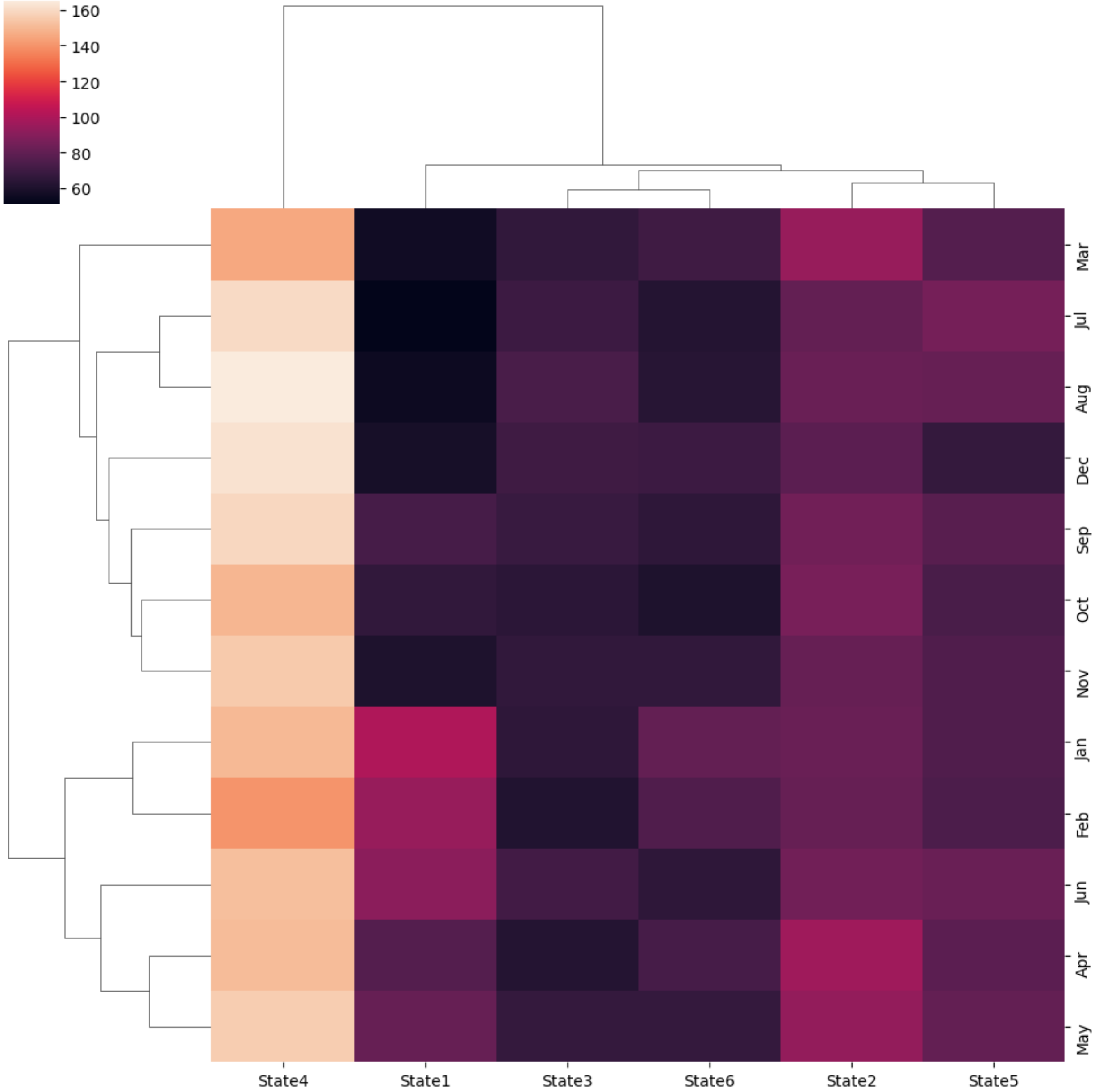


2. Clustermap:

- The clustermap() function of seaborn plots the hierarchically-clustered heatmap of the given matrix dataset.
- Clustering simply means grouping data based on relationship among the variables in the data.
- Syntax:
  - `clustermap(data, *, pivot_kws=None,  **kwargs)`

```
s1 = [100, 94, 56, 76, 81, 91, 51, 55, 72, 66, 60, 58 ]
s2 = [82, 81, 94, 96, 93, 84, 80, 82, 84, 86, 81, 78]
s3 = [65, 61, 66, 62, 67, 71, 69, 73, 68, 64, 66, 70]
s4 = [150, 140, 145, 151, 156, 152, 160, 165, 159, 149, 155, 162]
s5 = [75, 74, 76, 78, 80, 82, 85, 81, 77, 73, 75, 67]
s6 = [80, 75, 70, 72, 67, 65, 62, 63, 65, 60, 66, 69]

months= ["Jan", "Feb", "Mar", "Apr",
          "May", "Jun", "Jul", "Aug",
          "Sep", "Oct", "Nov", "Dec"]

d1 = {"State1":s1,
      "State2":s2,
      "State3":s3,
      "State4":s4,
      "State5":s5,
      "State6":s6};

df = pd.DataFrame(data=d1, index=months)
print(df)
print(df.columns)
print(df.index)

sns.clustermap(df)
plt.show()
```

```
     State1  State2  State3  State4  State5  State6
Jan     100      82      65     150      75      80
Feb      94      81      61     140      74      75
Mar      56      94      66     145      76      70
Apr      76      96      62     151      78      72
May      81      93      67     156      80      67
Jun      91      84      71     152      82      65
Jul      51      80      69     160      85      62
Aug      55      82      73     165      81      63
Sep      72      84      68     159      77      65
Oct      66      86      64     149      73      60
Nov      60      81      66     155      75      66
Dec      58      78      70     162      67      69
Index(['State1', 'State2', 'State3', 'State4', 'State5', 'State6'],
dtype='object')
Index(['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep',
'Oct',
       'Nov', 'Dec'],
      dtype='object')
```


160
140
120
100
80
60
Mar
Jul
Aug
Dec
Sep
Oct
Nov
Jan
Feb
Jun
Apr
May
State4
State1
State3
State6
State2
State5