STAT3622 Data Visualization (with Python)

Lecture 1

Wenbin Du
The University of Hong Kong

18 January 2021

Pycharm: Python IDE (Integrated Development Environment)

- code completion
- manage your packages automatically
- keyboard shortcuts
  - command+D
  - command+/
  - command+F
  - ...

Matplotlib: Visualization with Python

- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

```
import matplotlib.pyplot as plt
```

```
y= [3,2,3]
plt.figure()
plt.plot(y)
plt.ylabel('Y')
plt.title('Test')
plt.show()
```

Pandas: Visualization with Python

- pandas is an open source library providing high-performance, easy-to-use data structures and data analysis tools for Python

```
import pandas as pd
```

```
iris = pd.read_csv("iris.data")
```

Nelli F. Python data analytics: with pandas, numpy, and matplotlib[M]. Apress, 2018.

Pandas: DataFrame

```
type(iris)
```

pandas.core.frame.DataFrame

```
iris.index
```

RangeIndex(start=0, stop=150, step=1)

```
iris.columns
```

Index(['SepalLength', 'SepalWidth', 'PetalLength', 'PetalWidth', 'Name'], dtype='object')

```
iris.values[:3,:4]
```

array([[5.1, 3.5, 1.4, 0.2], [4.9, 3.0, 1.4, 0.2], [4.7, 3.2, 1.3, 0.2]], dtype=object)

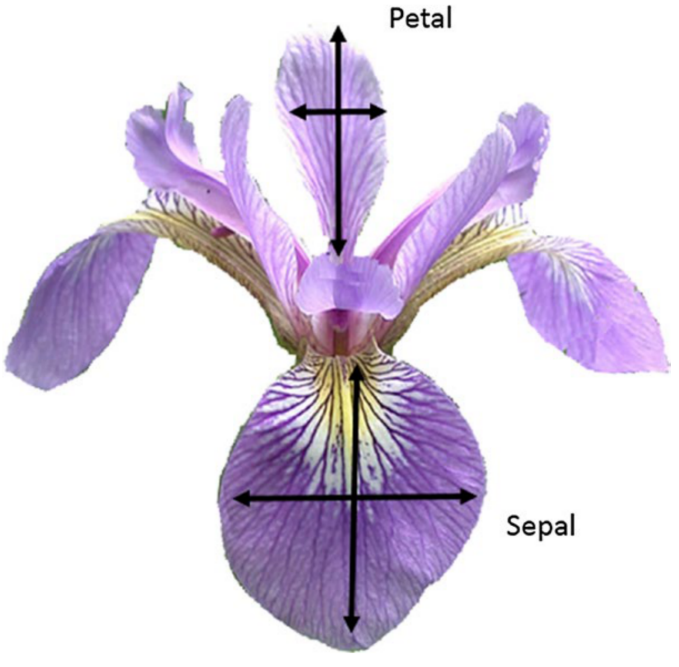
Pandas: DataFrame

	SepalLength	SepalWidth	PetalLength	PetalWidth	Name
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

column

value

index



Pandas: DataFrame attributes

```
iris.head()
```

	SepalLength	SepalWidth	PetalLength	PetalWidth	Name
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

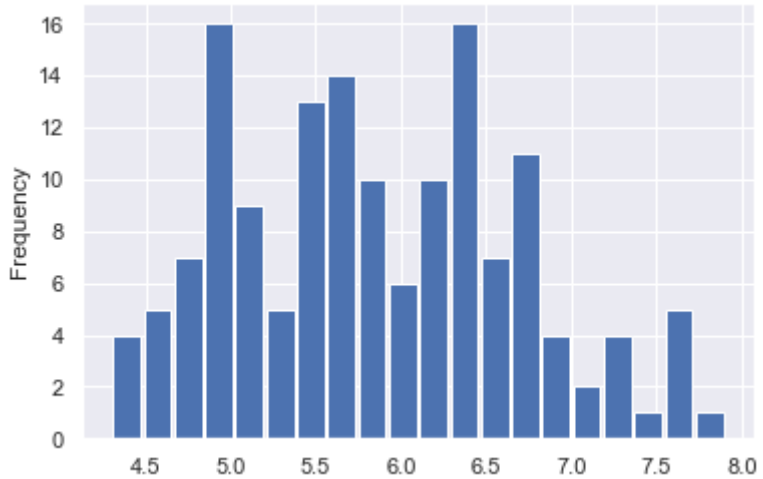
Pandas: DataFrame attributes

```
iris.describe()
```

	SepalLength	SepalWidth	PetalLength	PetalWidth
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

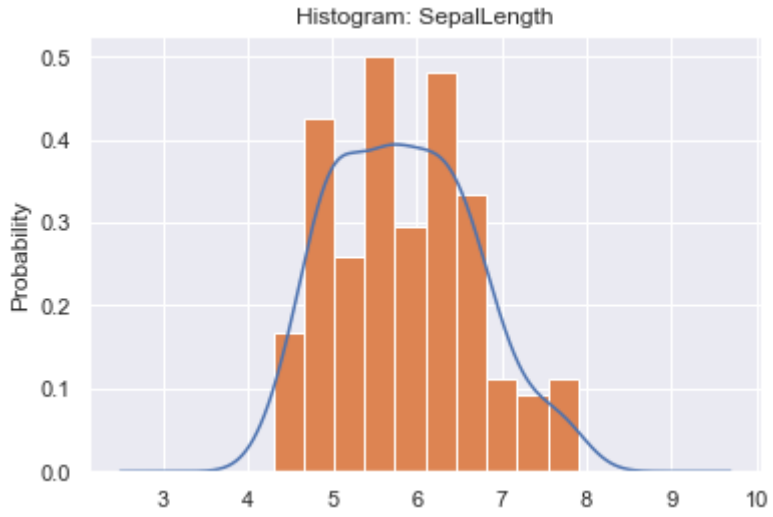
DataFrame: hist

```
x = iris["SepalLength"]
x.plot.hist( bins=20,rwidth=0.9)
```



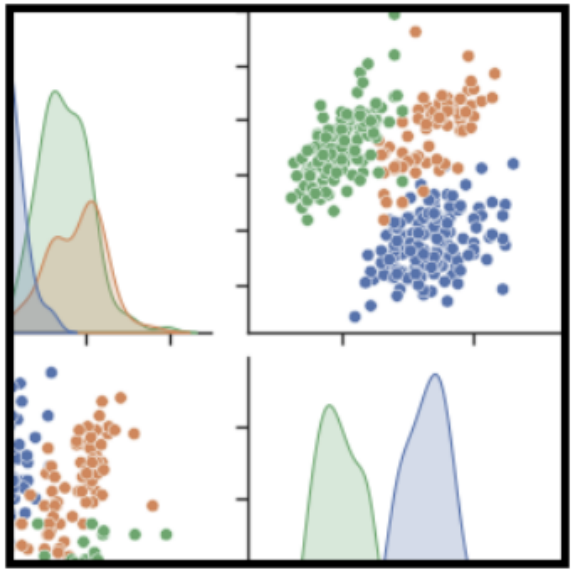
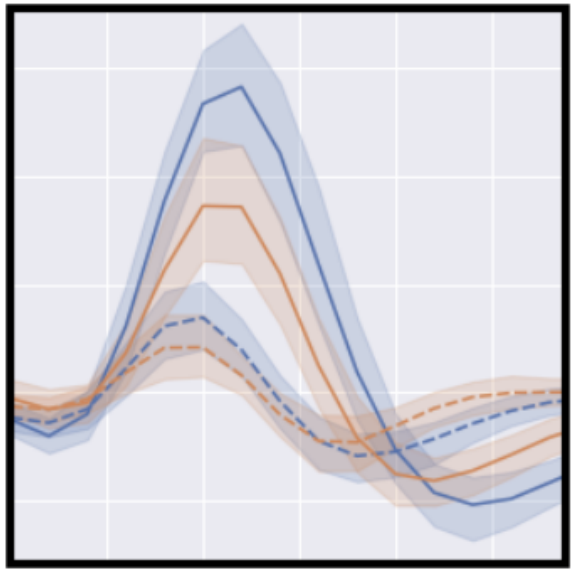
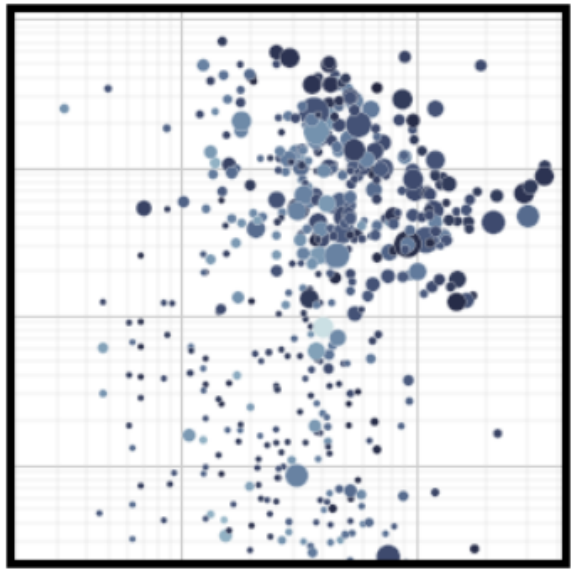
DataFrame: kde

```
fig, ax = plt.subplots(1,1)
x.plot.kde(ax=ax, legend=False, title='Histogram: SepalLength')
x.plot.hist(ax=ax,density=True)
ax.set_ylabel('Probability')
# ax.grid(axis='y')
# ax.set_facecolor('#d8dcd6')
```



Seaborn:statistical data visualization

- Seaborn is a Python data visualization library based on matplotlib.
- It provides a high-level interface for drawing attractive and informative statistical graphics.

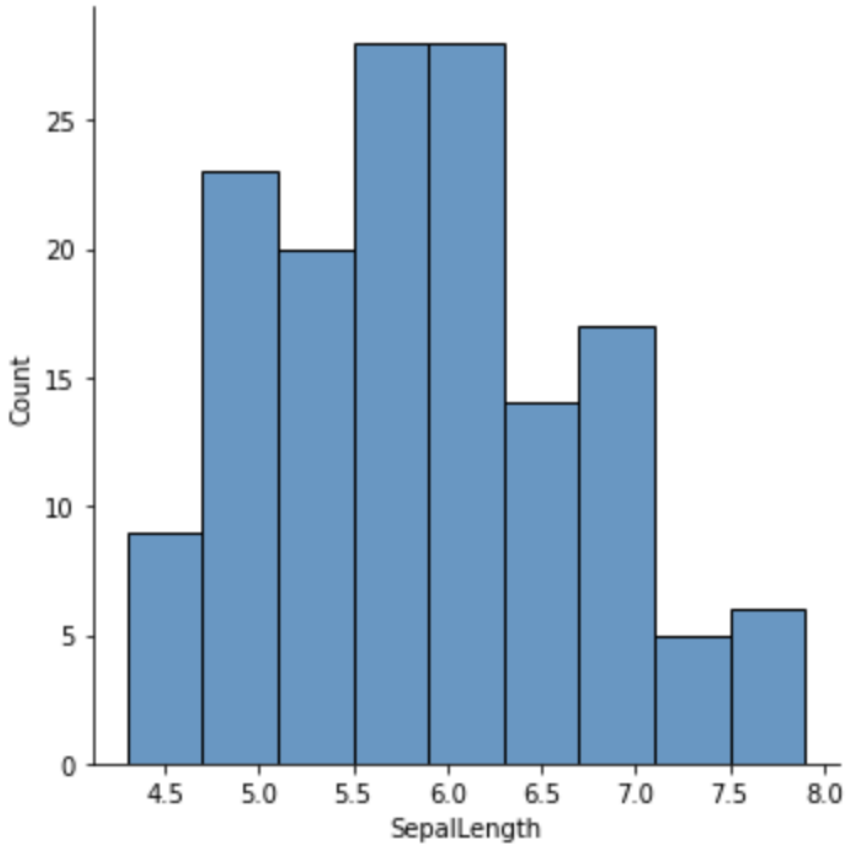




Seaborn

- Univariate Plots
  - displot
- Bivariate Plot
  - jointplot
  - airplot
  - displot
- Bivariate Plot with Conditioning
  - lmplot
- Trivariate Plot
  - 3D Plot
  - Heatmap

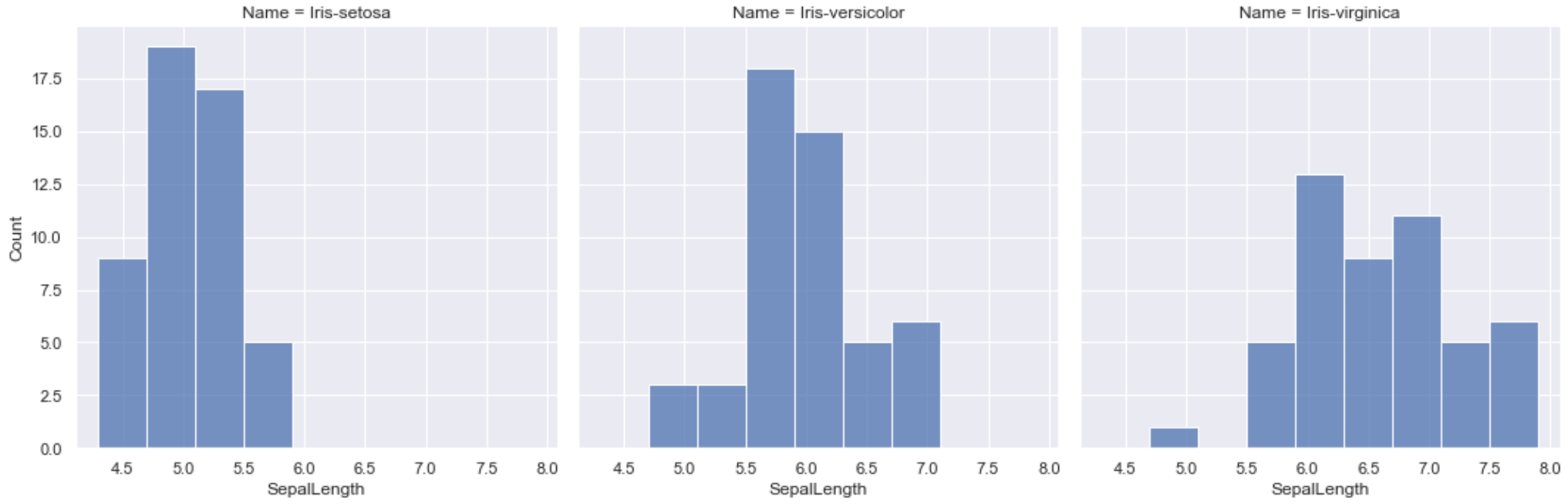
Univariate Plots: sns.displot

```
import seaborn as sns
sns.displot(data=iris,x="SepalLength")
```



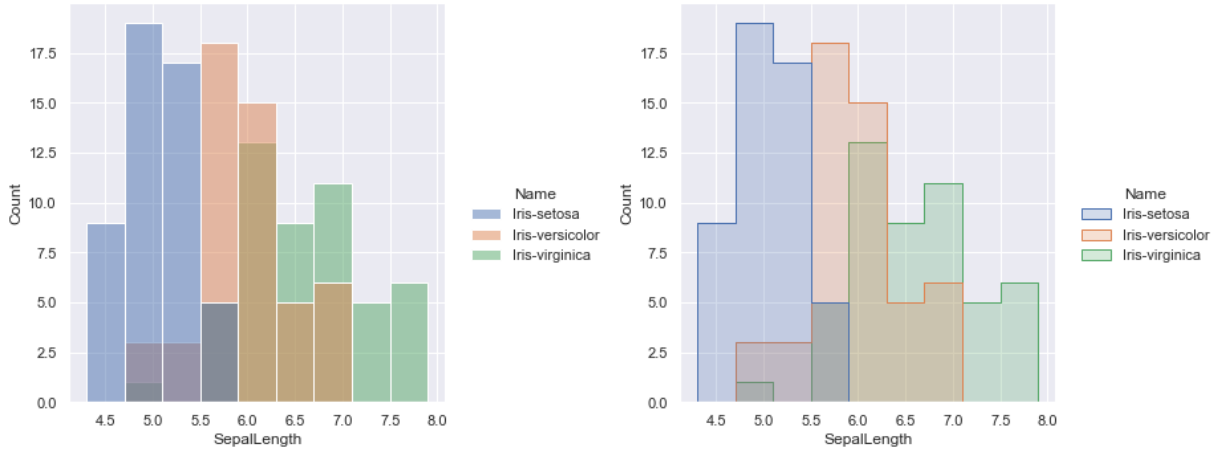
Univariate Plots: sns.displot

```
import seaborn as sns
sns.displot(data=iris,x="SepalLength",col="Name",multiple="dodge")
```



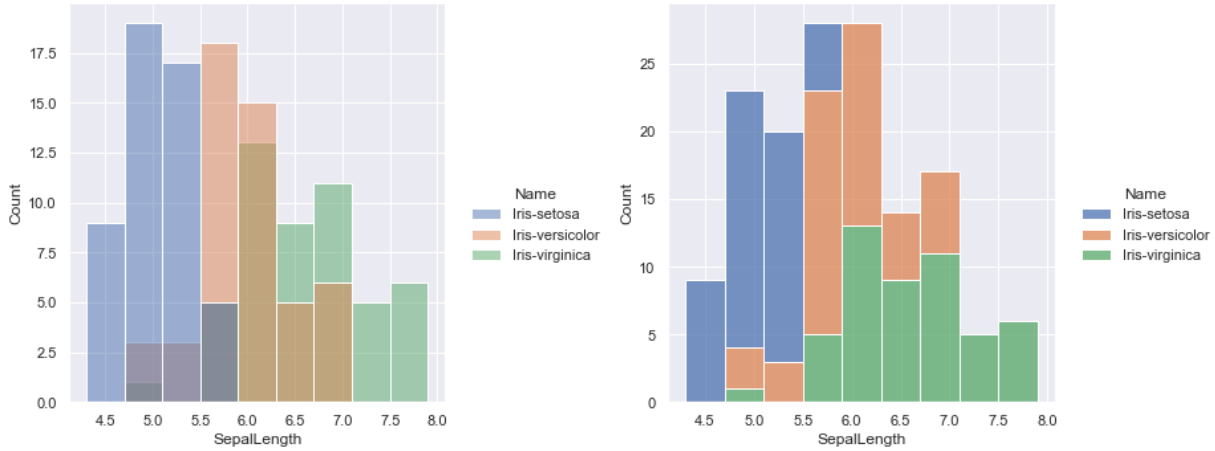
Univariate Plots: sns.displot

```
sns.displot(data=iris,x="SepalLength",hue="Name")
sns.displot(data=iris,x="SepalLength",hue="Name",element="step")
```



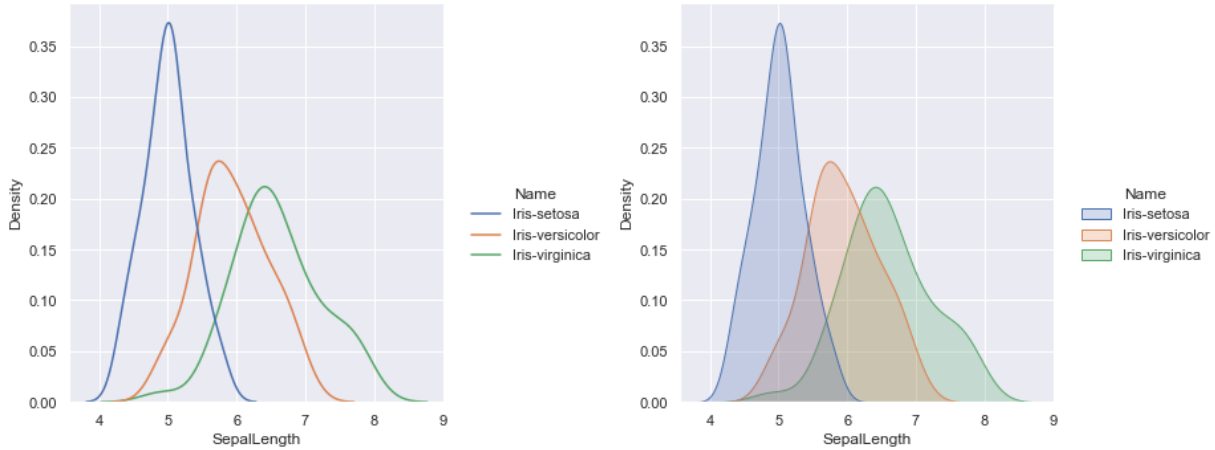
Univariate Plots: sns.displot

```
sns.displot(data=iris,x="SepalLength",hue="Name")
sns.displot(data=iris,x="SepalLength",hue="Name",multiple="stack")
```



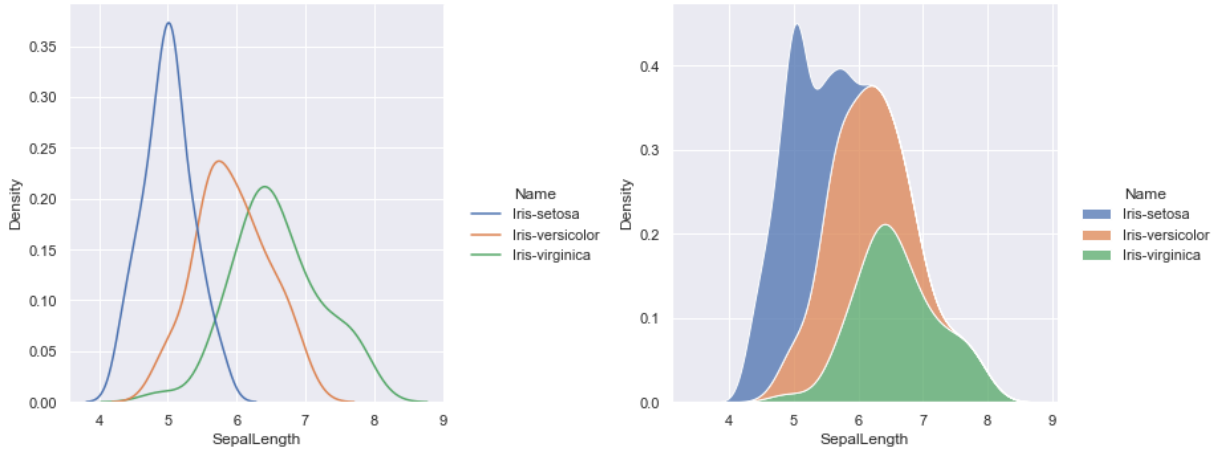
Univariate Plots: sns.displot

```
sns.displot(data=iris, x="SepalLength",hue="Name", kind="kde")
sns.displot(data=iris, x="SepalLength",hue="Name", kind="kde",fill=True)
```



Univariate Plots: sns.displot

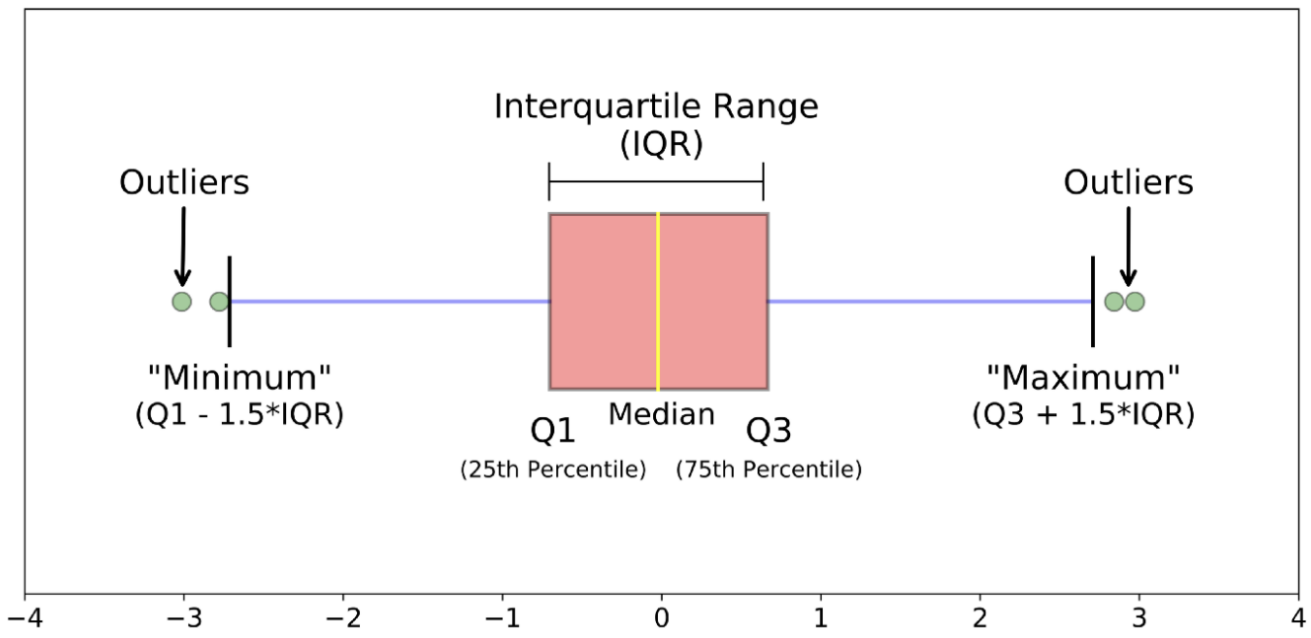
```
sns.displot(data=iris, x="SepalLength",hue="Name", kind="kde")
sns.displot(data=iris, x="SepalLength",hue="Name", kind="kde",multiple="stack")
```



Seaborn:boxplot

```
import seaborn as sns
```

A boxplot is a way of displaying the distribution of data based on a five number summary (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”).

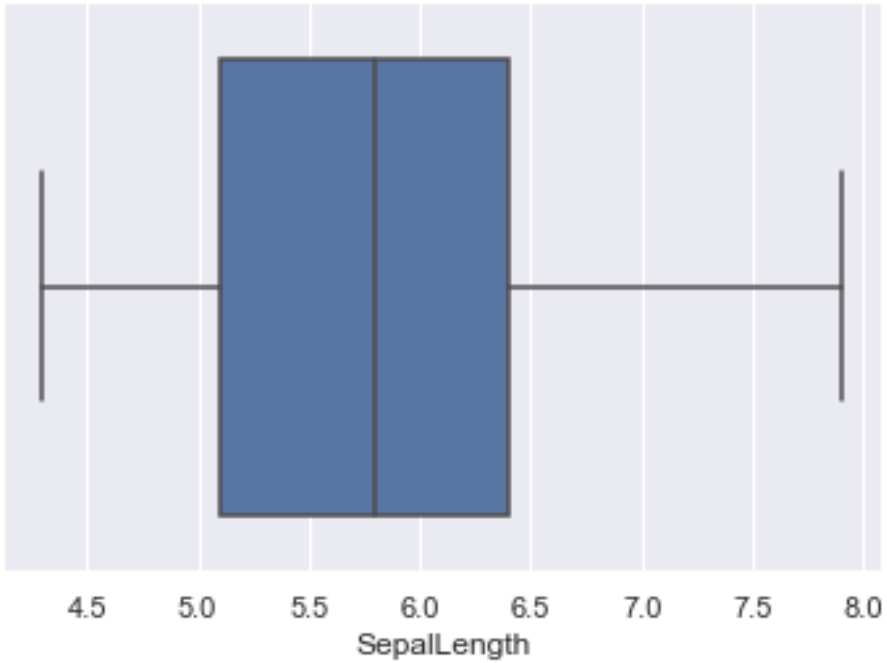


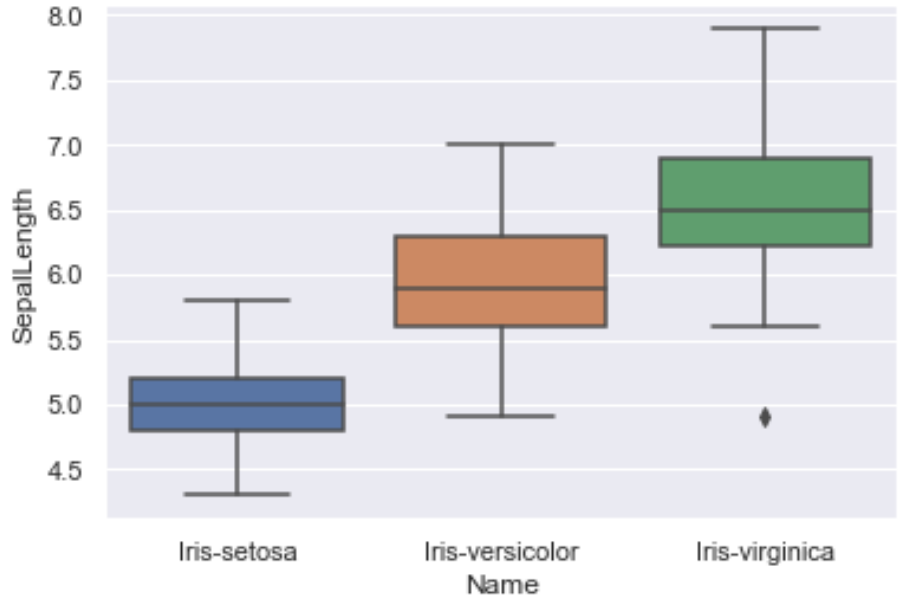
Different parts of a boxplot

https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

Seaborn:boxplot

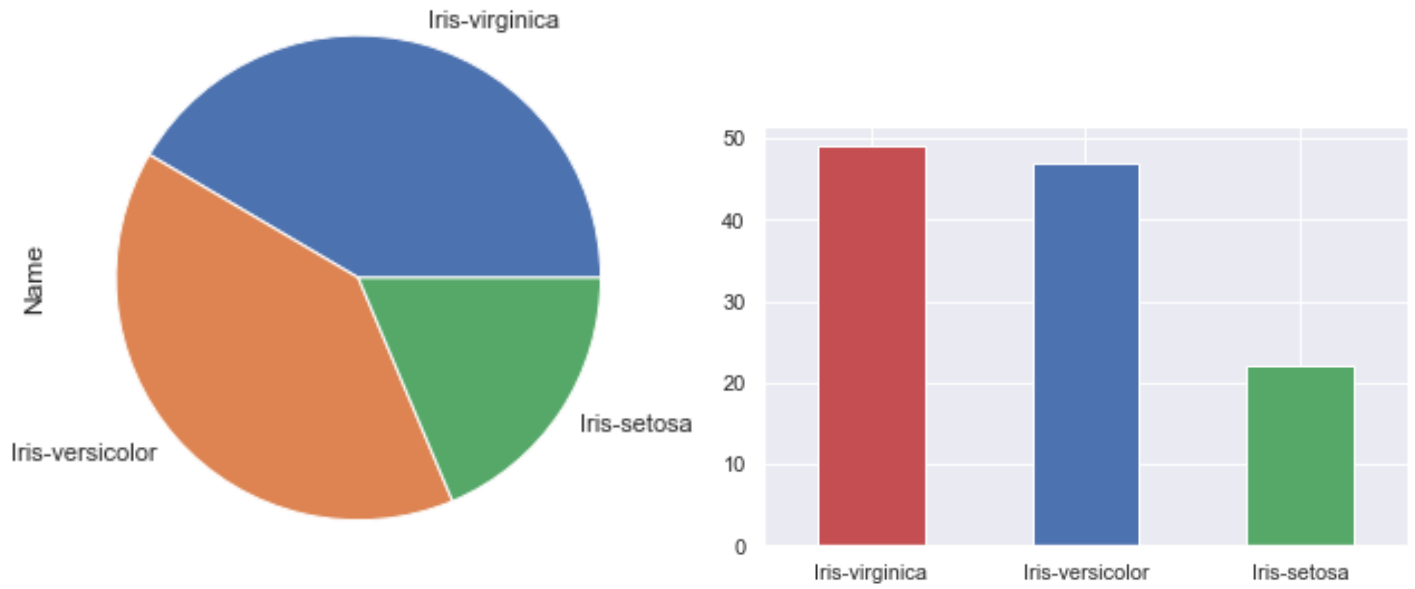
```
ax = sns.boxplot(data=iris, x="SepalLength")
ax = sns.boxplot(data=iris, x="Name",y="SepalLength")
```





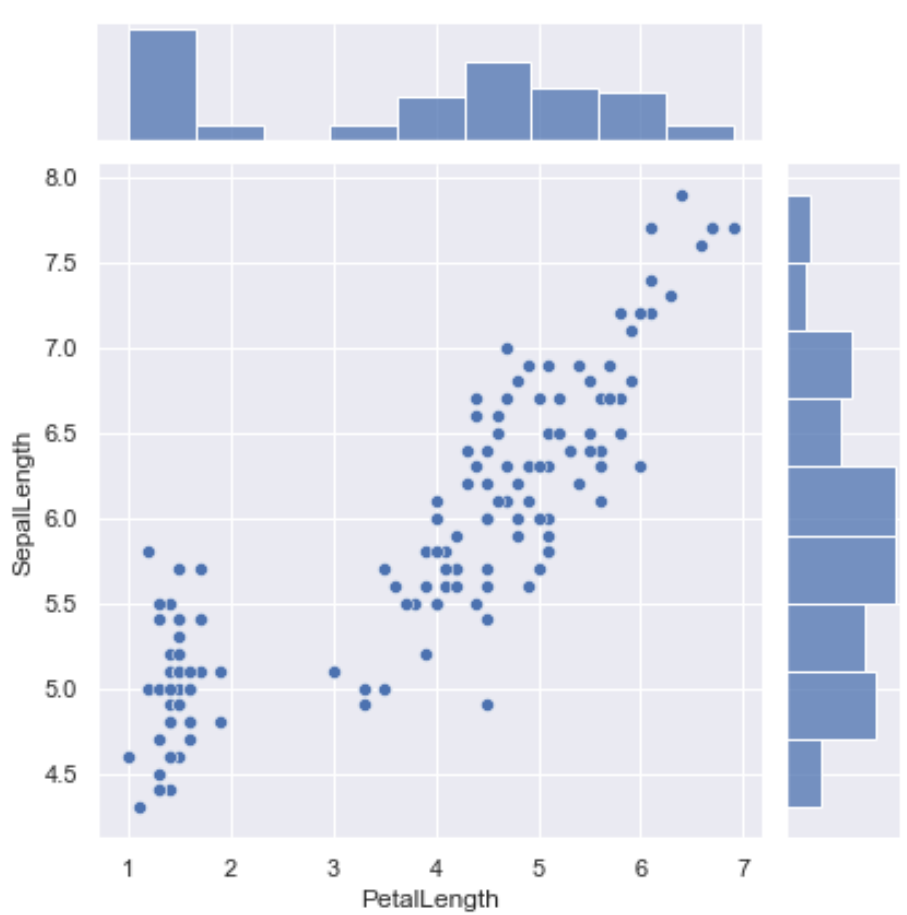
Pandas: pie&bar

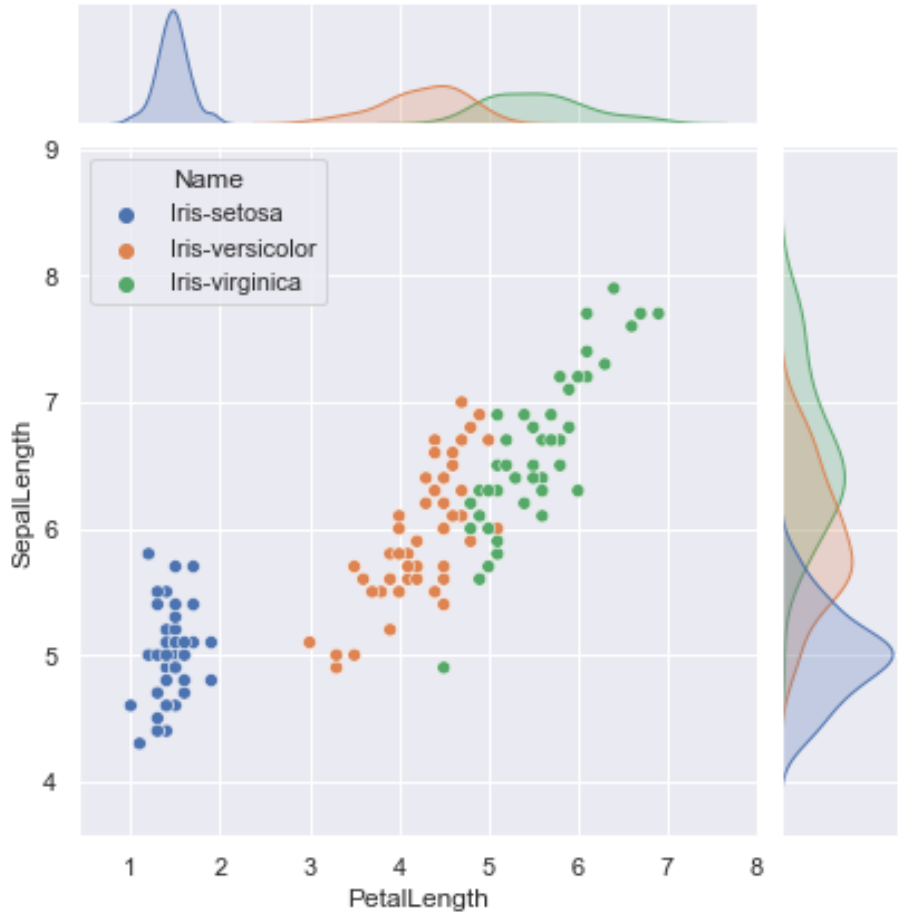
```
flag = x>5
species = iris["Name"]
species = species[flag]
species_data=pd.DataFrame(species)
data = species_data["Name"].value_counts()

data.plot.pie(y="Name",figsize=(5,5))
data.plot.bar(x=data.index,y=data.values,rot=0,color=['r','b','g'])
```



Bivariate plot : sns.jointplot

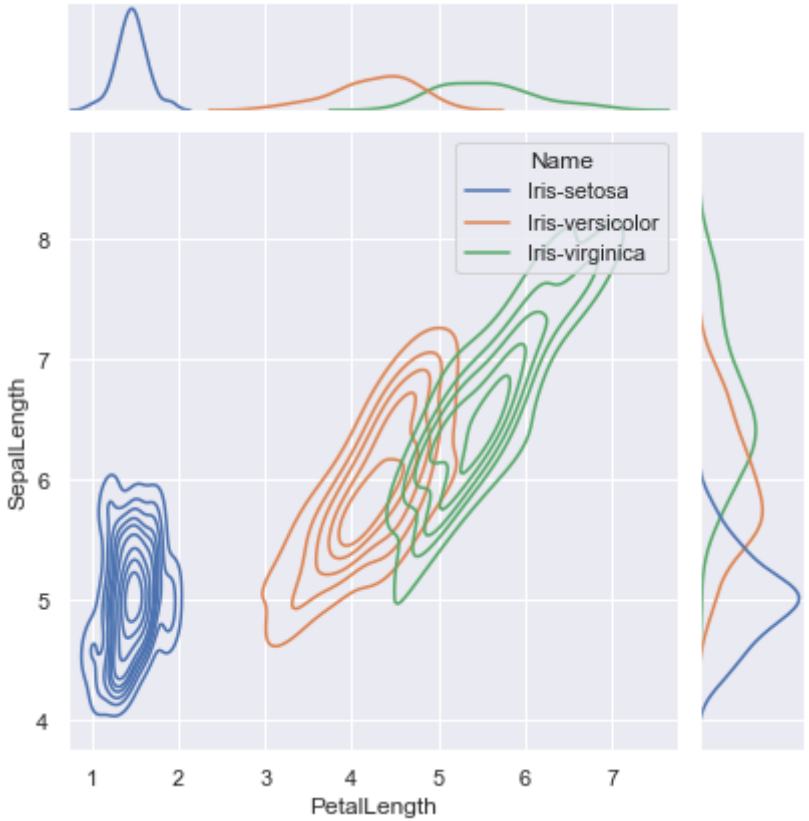
```
sns.jointplot(data=iris, x="PetalLength", y="SepalLength")
sns.jointplot(data=iris, x="PetalLength", y="SepalLength",hue="Name",)
```





Bivariate plot : sns.jointplot

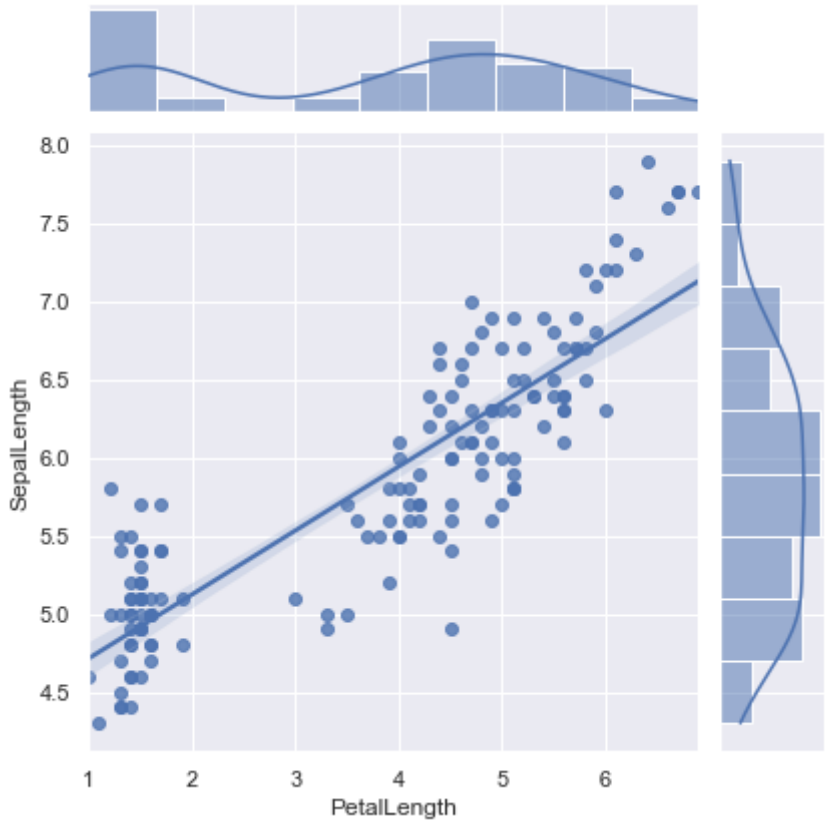
```
sns.jointplot(data=iris, x="PetalLength", y="SepalLength",hue="Name",kind="kde")
```



Bivariate plot: jointplot

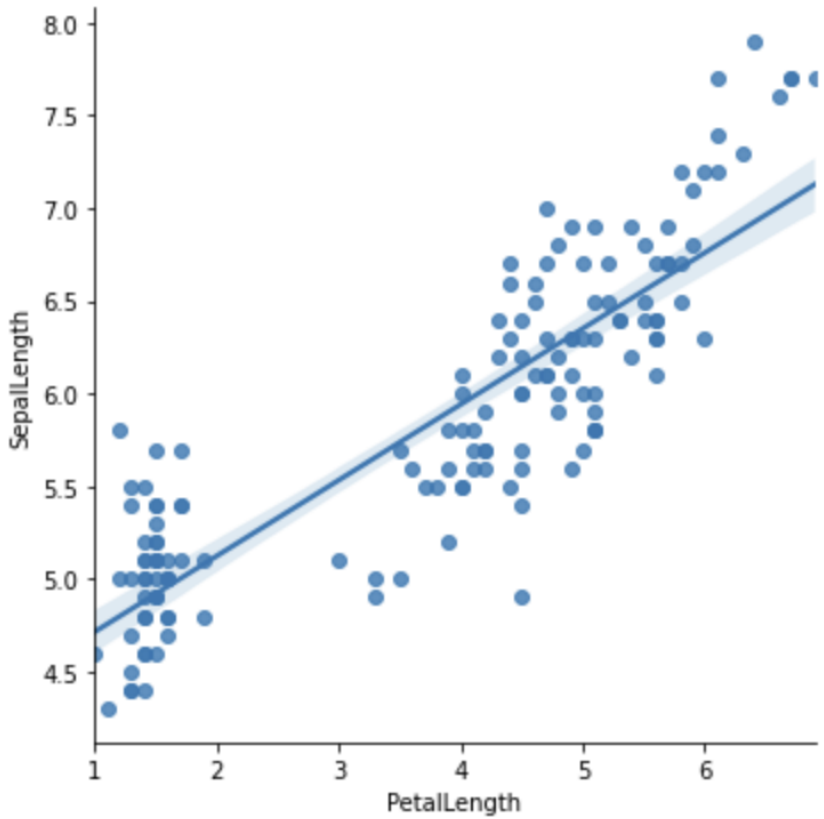
```
#Set kind="reg" to add a linear regression fit and univariate KDE curves:
```

```
sns.jointplot(data=iris, x="PetalLength", y="SepalLength",kind="reg")
```



Bivariate plot: lmplot

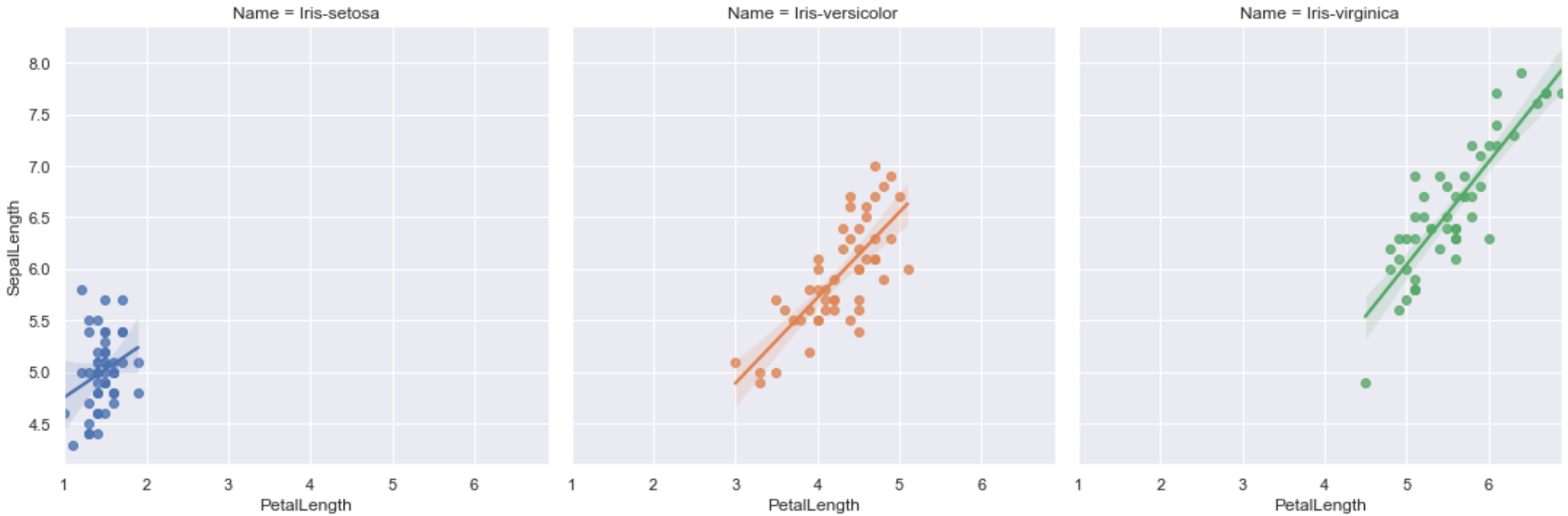
```
## fit regression models using lmplot
sns.lmplot(x = 'PetalLength', y = 'SepalLength', data = iris)
```



Bivariate plot: lmplot with conditioning

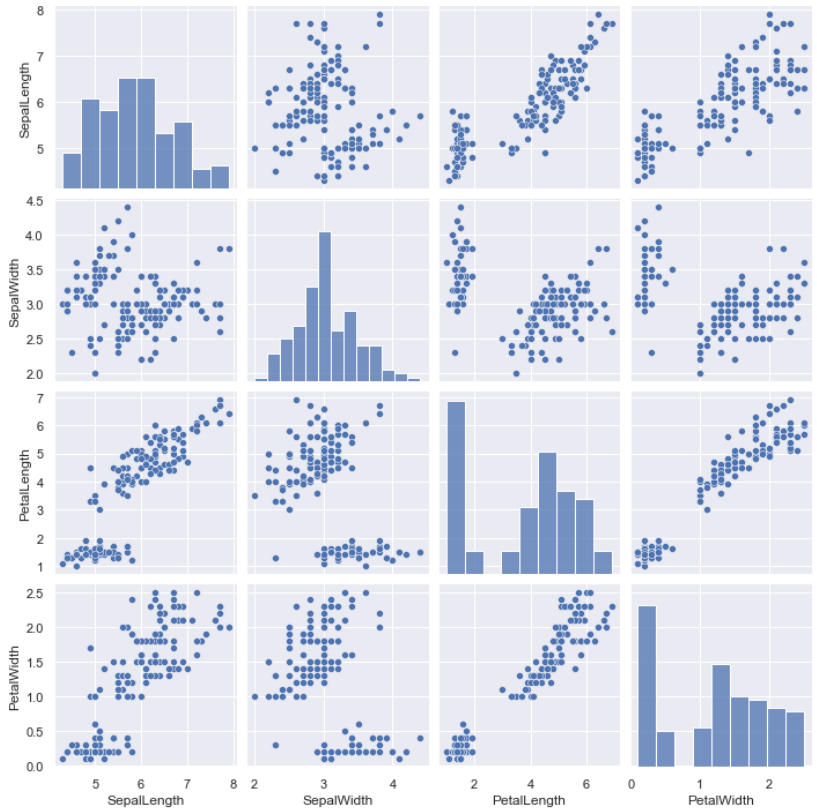
```
## fit regression models using lmplot across conditional subsets of a dataset
```

```
sns.lmplot(x = 'PetalLength', y = 'SepalLength', data = iris, hue = 'Name', col = 'Name')
```



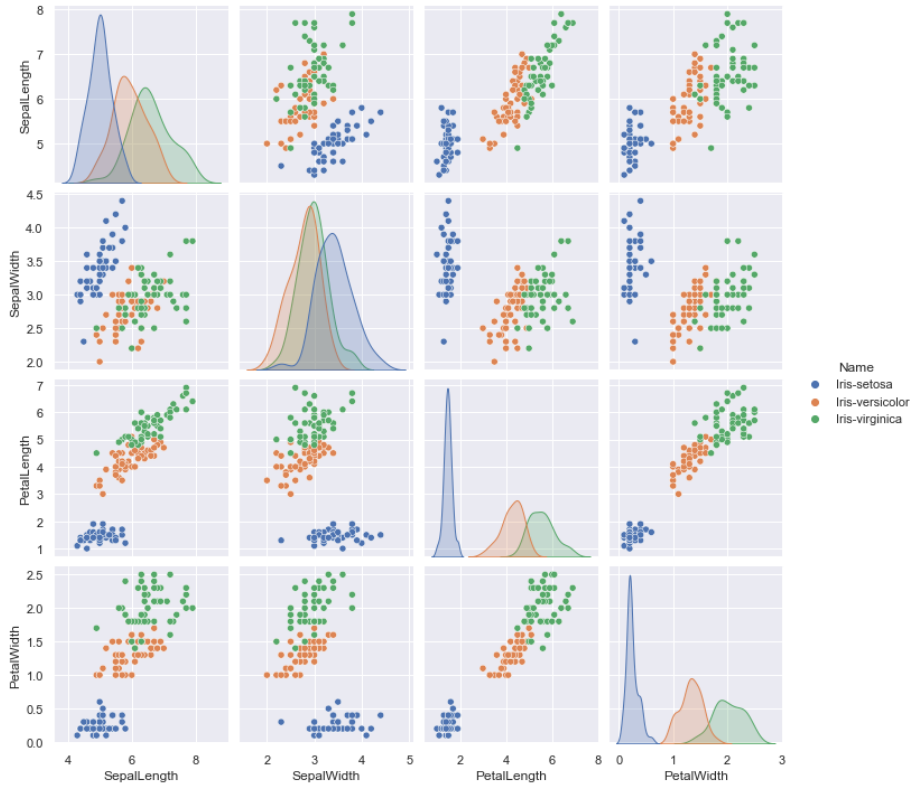
Bivariate plot : pairplot

```
sns.pairplot(data=iris)
```



Bivariate plot : pairplot

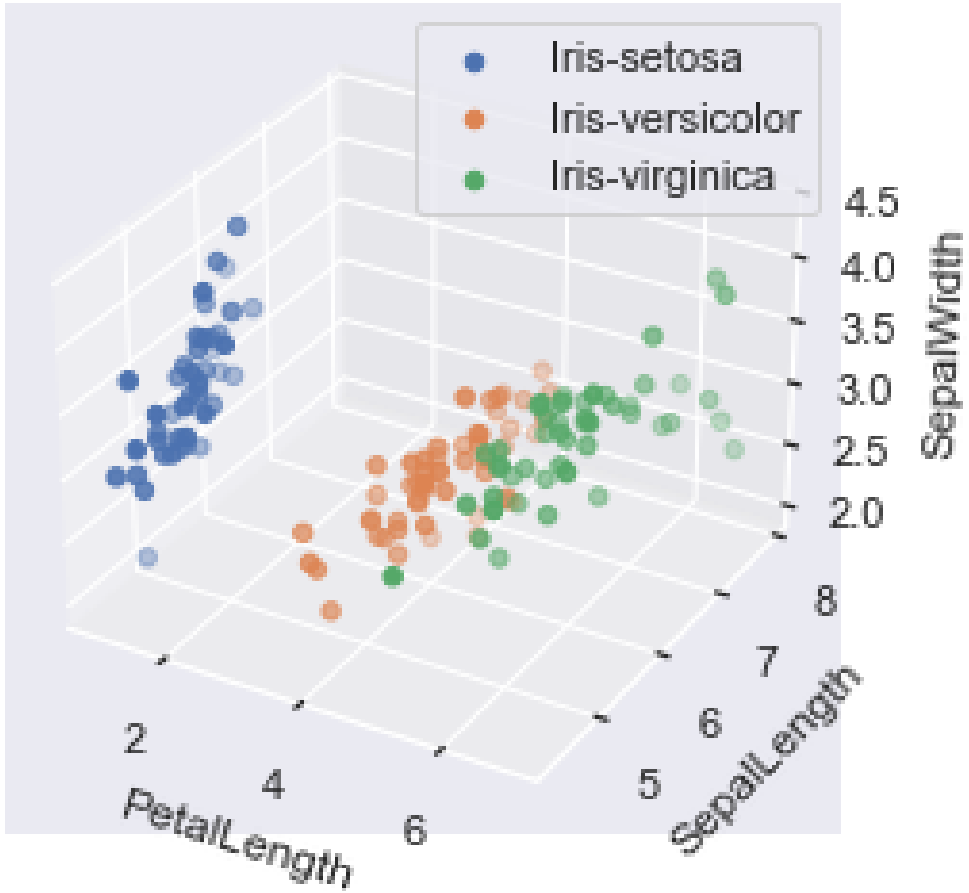
```
sns.pairplot(data=iris,hue="Name")
```



Trivariate Plot: 3D Plot

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
name =iris["Name"].unique()#['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
z1= iris["Name"]
for i in range(3):
    # print(i)
    flag =z1==name[i]
    # print(flag)
    x = iris['PetalLength'][flag]
    y = iris['SepalLength'][flag]
    z = iris['SepalWidth'][flag]
    ax.scatter(x, y, z,label=name[i])
ax.legend()
ax.set_xlabel("PetalLength")
ax.set_ylabel("SepalLength")
ax.set_zlabel("SepalWidth")
plt.show()
```

Trivariate Plot: 3D Plot



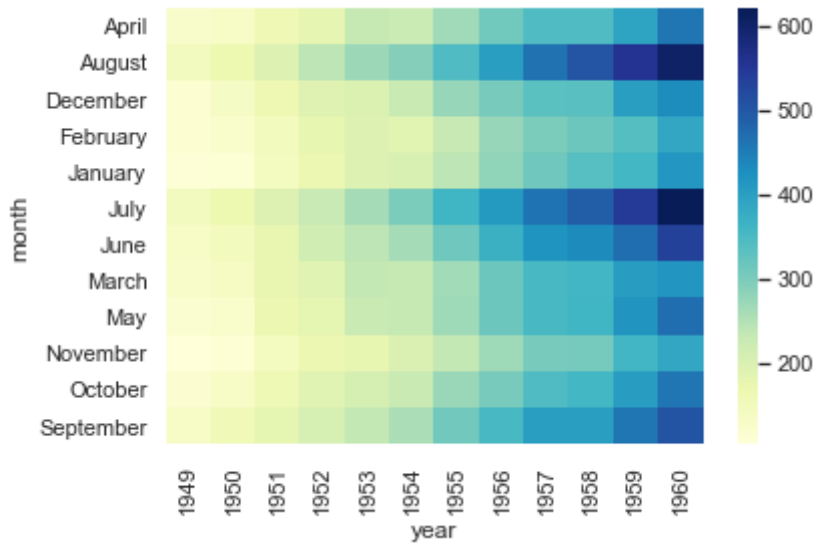
Trivariate Plot: Heatmap

- Plot a heatmap with meaningful row and column labels

```
flights = pd.read_csv("flights.csv")
#flights.head()
```

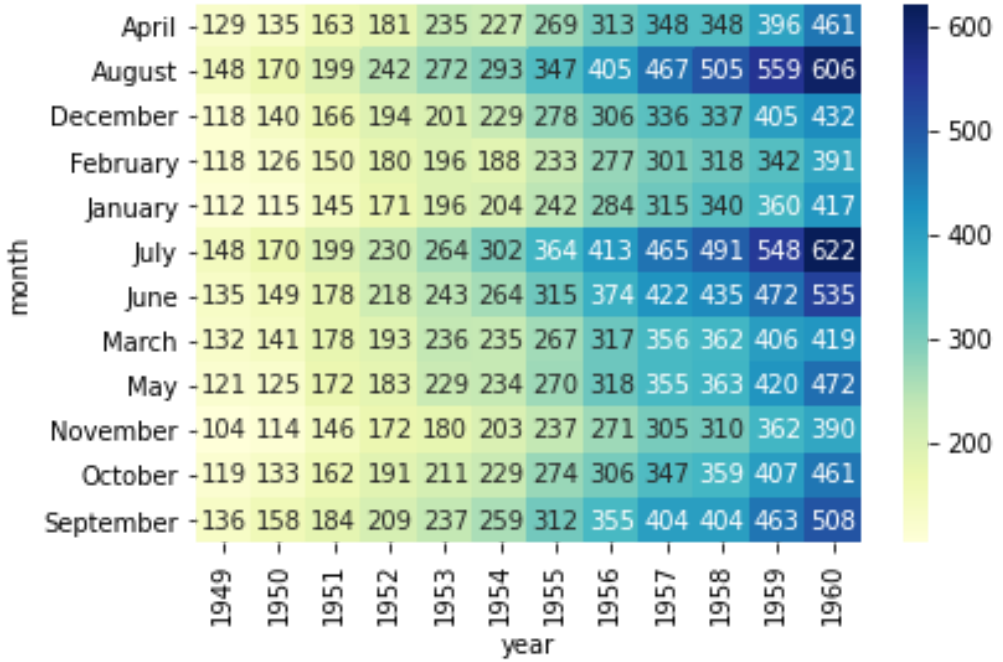
```
flights = flights.pivot("month", "year", "passengers")
```

```
ax = sns.heatmap(flights,cmap="YlGnBu")
```



Trivariate Plot: Heatmap

```
ax = sns.heatmap(flights,,cmap="YlGnBu", annot=True, fmt="d")
```



Thank you!

Q&A or Email wbdu@hku.hk