R vs Python: Data Visualization Tools Comparison

Quick Start

R

ggplot2 is an R plotting package based on the grammar of graphics. A completed ggpolot structure contains several functions, users can create complex plots by adding different geoms functions.

Python

Matplotlib and Seaborn are two comprehensive Python visualization libraries for creating publication-quality figures and providing attractive statistical graphics in a variety of environments.

The Basic Steps to Create Plots

Step 1. Load ggplot2

```
> library(ggplot2)
```

Step 2. Load data Required

```
> ggplot(data=<DATA>)+
```

Step 3. Customize plot

```
<GEOM_FUNCTION>(mapping
=aes<MAPPINGS>,
stat=<STAT>,
position=<POSITION>)+
<COORDINATE_FUNCTION>+
<SCALE_FUNCTION>+
<THEME_FUNCTION>
```

Step 1. Import libraries

```
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 2. Prepare data

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

Step 3. Customize plot

```
fig , ax = plt.subplots(1, 1,
figsize=(12, 8))
ax.<PLOT_NAME>(X, y,
color= 'red', marker='o')
plt.show()
```

Figure Aesthetics

```
t + labs(
x = "New x axis label",
y = "New y axis label",
title ="Add a title,
subtitle = "Add a subtitle
below title",
caption = "Add a caption
below plot",
alt = "Add alt text to the
plot")
```

Scale, Labels, Title

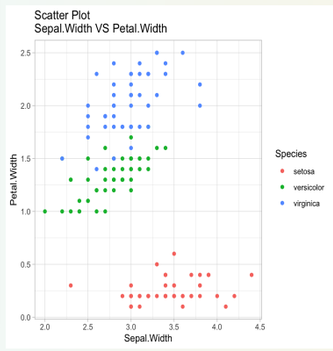
```
ax.set_xlim(1, 6)
ax.set_xlabel('x')
ax.set_title('Iris Plot')
```

Seaborn Style

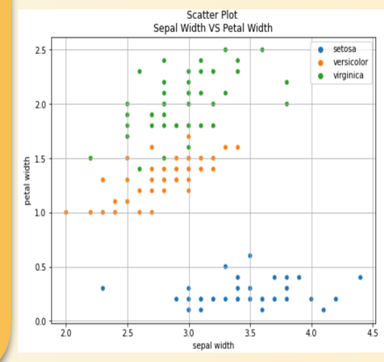
```
sns.set_style("whitegrid")
sns.set_context("talk")
sns.set_palette("husl",3)
```

1 Scatter Plot (R vs Python)

```
ggplot(iris,
aes(x=Sepal.Length, y=Petal.Length))
+
geom_point(color=
"blue")+
 labs(title="Scatter
Plot \nSepal.Width
VS Sepal.Length")+
 theme_light()
```

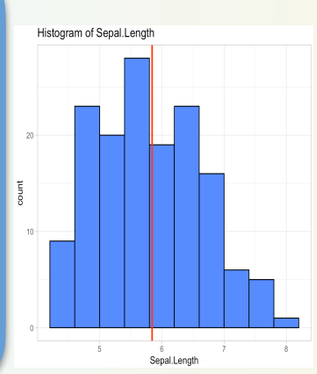



```
sns.scatterplot(x=iris["sepal_
width"], y=iris["petal_width"],
        marker="o",
hue=iris["species"]
ax.set_xlabel('sepal width')
ax.set_ylabel('petal width')
ax.set_title('Scatter Plot
\nSepal Width VS Petal
Width')
plt.legend()
plt.grid()
```

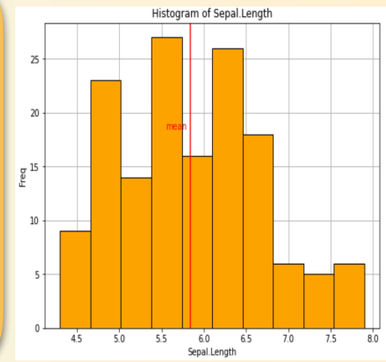


2 Histogram (R vs Python)

```
ggplot(iris,
aes(x=Sepal.Length))
+
geom_histogram(bins
= 10, color =
"black",fill="royalblue
1")+theme_light()+
geom_vline(aes(xinte
rcept =
mean(Sepal.Length)),
col='red',size=1)
```

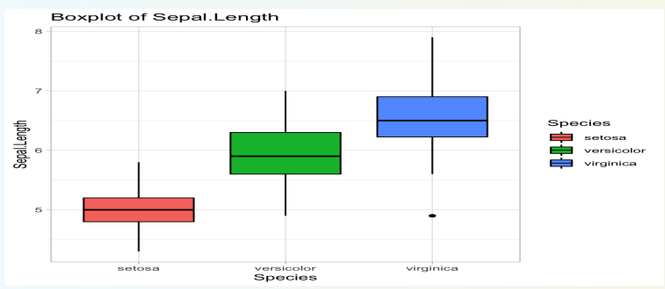



```
ax.hist(iris["sepal_length"],
bins=10, edgecolor="black",
color="orange")
ax.axvline(iris["sepal_length
"].mean(), color="red")
ax.text(iris["sepal_length"].
mean()*0.95, ax.get_ylim()
[1]*0.65, "mean",
color="red")
```

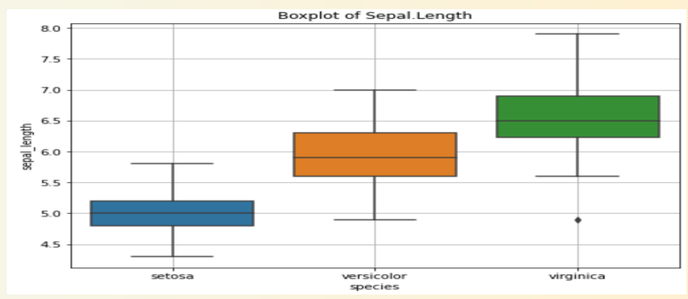


3 Boxplot (R vs Python)

```
ggplot(iris, aes(x=Sepal.Length, y=Petal.Length)) +
geom_point(color="blue")
```



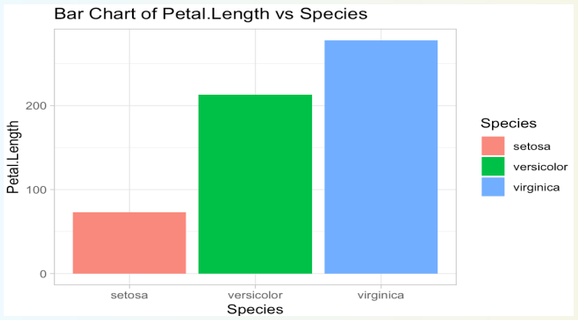

```
sns.boxplot(x=iris["species"], y=iris["sepal_length"])
```



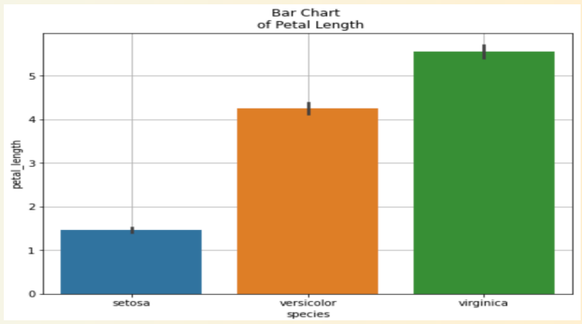
Created by **Dawei He** dh3027@columbia.edu & **Minhui Liao** ml4517@columbia.edu • Updated: 2021-10

4 Bar Chart (R vs Python)

```
ggplot(iris, aes(x=Species, y = Petal.Length,
fill=Species))+geom_bar(stat="identity")
```



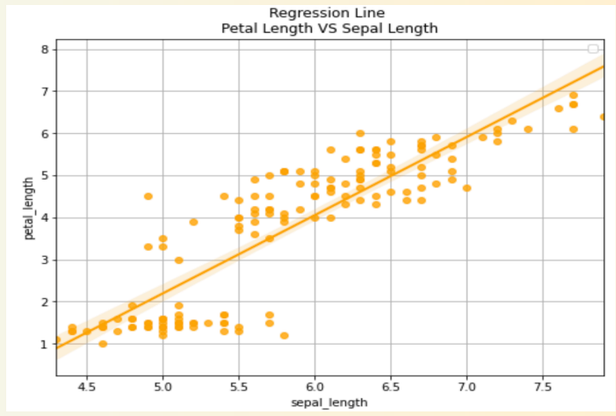

```
sns.barplot(x=iris["species"],
y=iris["petal_length"])
```



5 Regression Fitline (R vs Python)

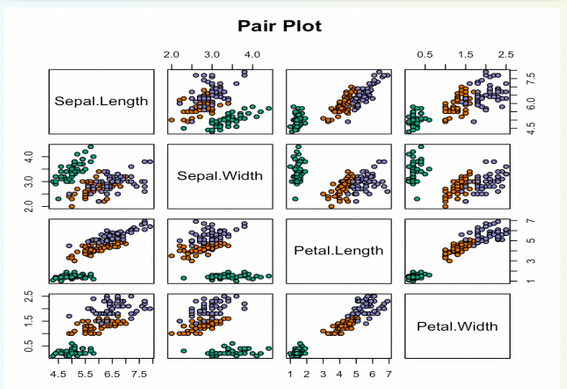
```
ggplot(iris, aes(x = Sepal.Length, y =
Petal.Length))+ geom_point()+
geom_smooth(method='lm', formula = y~x)
```


```
sns.regplot(x=iris["sepal_length"],
y=iris["petal_length"], color="orange")
```

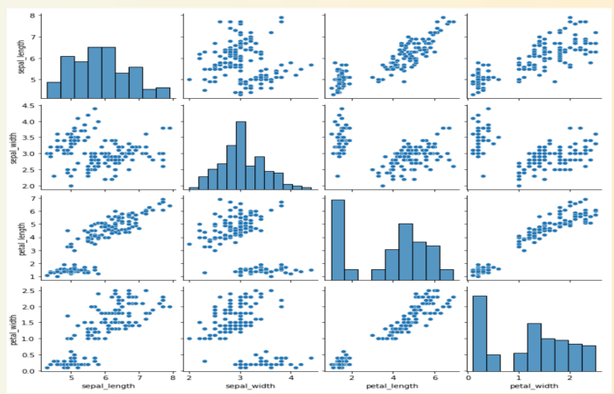


6 Pairplot (R vs Python)

```
pairs(iris[1:4], main = "Pair Plot",pch = 21,
bg = c("#1b9e77", "#d95f02", "#7570b3")
[unclass(iris$Species)])
```

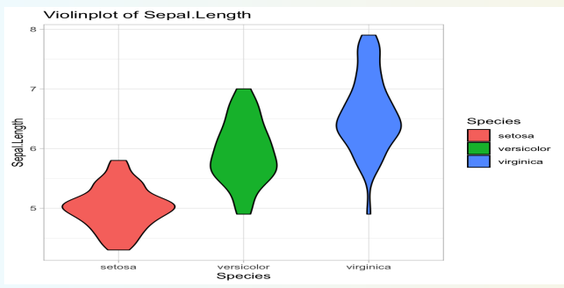



```
sns.pairplot(iris[["sepal_length",
"sepal_width", "petal_length",
"petal_width"]]);
```

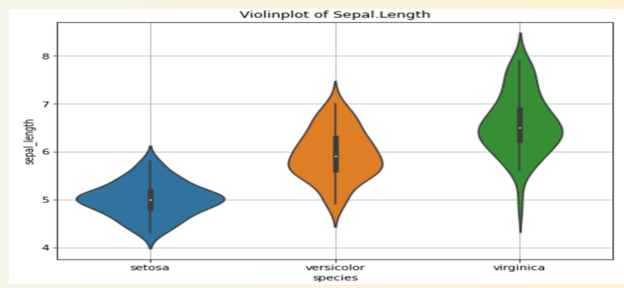


7 violinplot (R vs Python)

```
ggplot(iris, aes(x = Species, y = Petal.Length,
fill = Species)) + geom_violin(color = "black")
```

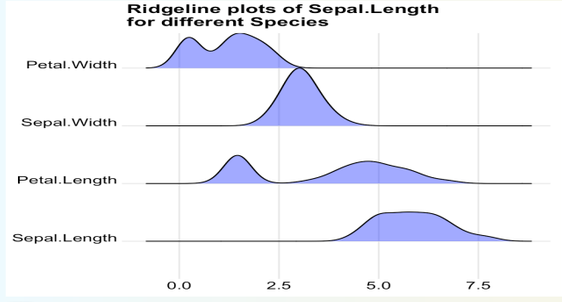



```
sns.violinplot(x=iris["species"],
y=iris["sepal_length"])
```

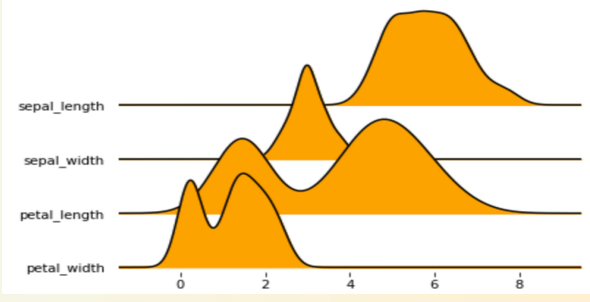


8 Ridgeline (R vs Python)

```
library(ggridges)
ggplot(iris %>% pivot_longer(!Species,
names_to = "paras", values_to="vals",
aes(vals, reorder(paras, -vals,
median)))+
geom_density_ridges(scale=1,
fill="royalblue1", alpha=.5)
```

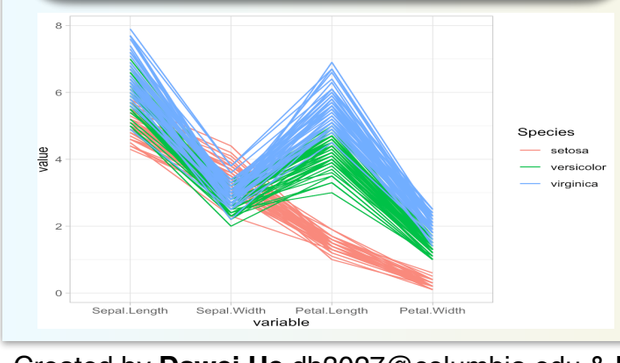



```
iris_ = pd.read_csv("iris.csv")
sample = ['t1', 't2', 't3', 't4']
fig, axes = joypy.joyplot(iris_,
ylim='own', color="orange")
for i in range(len(sample)):
    y_position = axes[i].get_ylim()[1] / 3.5
    axes[i].text(9, y_position)
```

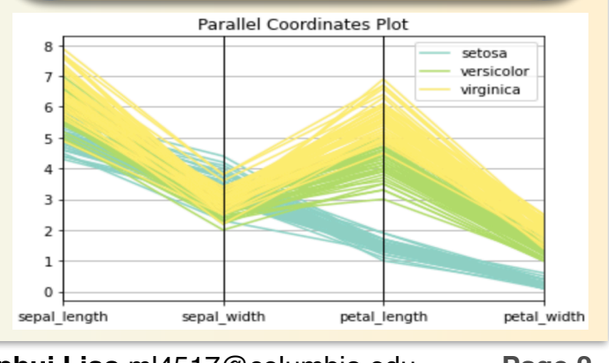


9 Parallel Coordinate Plot (R vs Python)

```
library(GGally)
ggparcoord(iris, columns = 1:4,
groupColumn = 5,scale="globalminmax",)
+ theme_light()
```




```
import pandas.plotting
from pandas.plotting import
parallel_coordinates
parallel_coordinates(iris, 'species',
colormap=plt.get_cmap("Set3"))
```



Created by **Dawei He** dh3027@columbia.edu & **Minhui Liao** ml4517@columbia.edu