Seaborn

点击一键运行代码→ | 获取更多速查表→ | 使用说明书 [视频 & 图文]→ bilibili ShowMeAI

Seaborn 是一种基于 matplotlib 的 python 图形可视化工具库。

它是在 matplotlib 的基础上进行了更高级的 API 封装，从而使得作图更加容易，能做出很具有吸引力的图，更简单地对数据分析过程做出更美观的可视化图表结果呈现。

作者 | 韩信子 @ShowMeAI
设计 | 南 乔 @ShowMeAI
参考 | datacamp cheatsheet

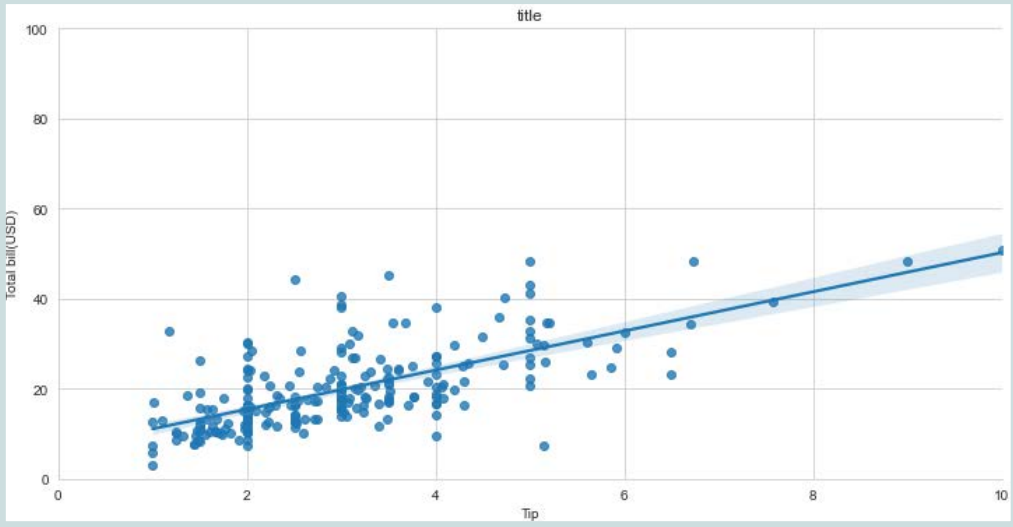
使用下列别名导入该库

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

Seaborn 绘图基本步骤与示例

Step 1 准备数据　Step 2 设定画布外观　Step 3 使用 Seaborn 绘图　Step 4 自定义图形　Step 5 展示图形

```
> tips = sns.load_dataset("tips") #Step 1
> sns.set_style("whitegrid") #Step 2
> g = sns.lmplot(x="tip", y="total_bill", data=tips, aspect=2) #Step 3
> g = (g.set_axis_labels("Tip", "Total bill(USD)").set(xlim=(0, 10), ylim=(0, 100)))
> plt.title("title") #Step 4
> plt.show(g) #Step 5
```



1. 数据准备

可以是 numpy 数组和 Dataframe 等数据格式

```
> import pandas as pd
> import numpy as np
> uniform_data = np.random.rand(10, 12)
> data = pd.DataFrame({'x':np.arange(1, 101),
                       'y':np.random.normal(0, 4, 100)})
```

Seaborn 提供了内置数据集：

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


2. 画布外观

```
> f, ax = plt.subplots(figsize=(5, 6)) # 创建画布与子图
```

Seaborn 样式

```
> sns.set()                                  # 设置或重置 Seaborn 默认值
> sns.set_style("whitegrid")                 # 设置 matplotlib 参数
> sns.set_style("ticks", {"xtick.major.size":8, "ytick.major.size":8})
> sns.axes_style("whitegrid")                # 返回参数字典或用 with 设置临时样式
```

上下文函数

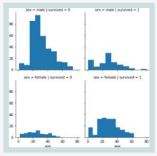
```
> sns.set_context("talk")                    # 将上下文设置为 "talk"
# 上下文设为 "notebook"，缩放字体，覆盖参数映射
> sns.set_context("notebook", font_scale=1.5, rc={"lines.linewidth":2.5})
```

调色板

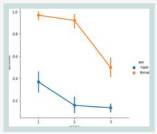
```
> sns.set_palette("husl", 3)                 # 定义调色版
> sns.color_palette("husl")                  # 使用 with 临时设置调色板
> flatui = ["#9b59b6", "#3498db", "#95a5a6", "#e74c3c", "#34495e", "#2ecc71"]
> sns.set_palette(flatui)                    # 设置调色板
```

3. 使用 Seaborn 绘图

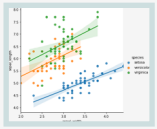
坐标轴栅格



```
# 绘制条件关系的子图栅格
> g = sns.FacetGrid(titanic, col="survived", row="sex")
> g = g.map(plt.hist, "age")
```

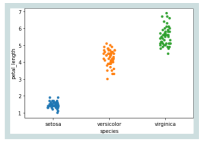


```
# 在分面栅格上绘制分类图
> sns.factorplot(x="pclass", y="survived", hue="sex", data=titanic)
```

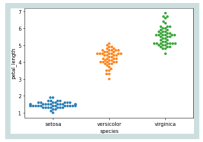


```
# 绘制适配分面栅格的数据与回归模型
> sns.lmplot(x="sepal_width", y="sepal_length", hue="species", data=iris)
```

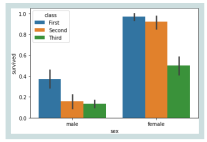
```
> h = sns.PairGrid(iris) # 绘制配对关系的子图栅格
> h = h.map(plt.scatter) # 绘制配对的双变量分布
> sns.pairplot(iris) # 绘制双变量图的边际单变量图栅格
> i = sns.JointGrid(x="x", y="y", data=data)
> i = i.plot(sns.regplot, sns.distplot)
# 绘制双变量分布
> sns.jointplot("sepal_length",  "sepal_width", data=iris, kind='kde')
```



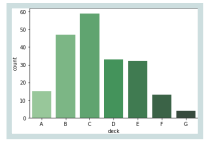
散点图·抖动图



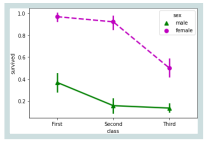
散点图·蜂群图



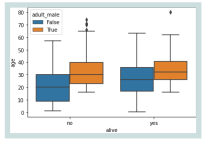
条形图



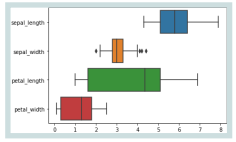
计数图



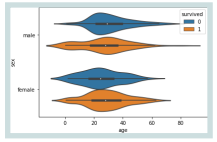
点图



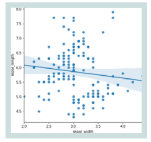
箱形图



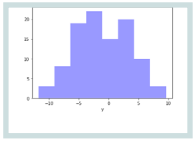
宽表数据箱形图



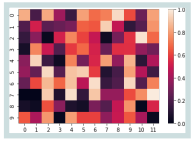
小提琴图



回归拟合图



变量分布图



热力图

常用绘图类型

散点图

```
# 含分类变量的抖动图
> sns.stripplot(x="species", y="petal_length", data=iris)

# 不重叠分类蜂群图
> sns.swarmplot(x="species", y="petal_length", data=iris)
```

条形图

```
# 用散点图示符显示点估计值和置信区间
> sns.barplot(x="sex", y="survived", hue="class", data=titanic)
```

计数图

```
# 显示观测数量
> sns.countplot(x="deck", data=titanic, palette="Greens_d")
```

点图

```
> sns.pointplot(x="class", y="survived", hue="sex", data=titanic,
                palette={"male":"g", "female":"m"},
                markers=["^", "o"],
                linestyles=["-", "--"]) # 显示点估计和置信区间
```

箱形图

```
> sns.boxplot(x="alive", y="age", hue="adult_male", data=titanic)
> sns.boxplot(data=iris, orient="h") # 使用宽表数据的箱形图
```

小提琴图

```
> sns.violinplot(x="age", y="sex", hue="survived", data=titanic)
```

回归拟合图

```
# 绘制与线性回归模型拟合的数据
> sns.regplot(x="sepal_width", y="sepal_length", data=iris, ax=ax)
```

变量分布图

```
> plot = sns.distplot(data.y, kde=False, color="b") # 绘制单变量分布
```

热力图

```
> sns.heatmap(uniform_data, vmin=0, vmax=1) # 热力图
```

4. 深度自定义

Axisgrid 对象

```
> g.despine(left=True) # 移除左框

> g.set_ylabels("Survived") # 设置 Y 轴标签
> g.set_xticklabels(rotation=45) # 设置 X 轴刻度标签

> g.set_axis_labels("Survived", "Sex") # 设置坐标轴标签

# 设置 X 与 Y 轴的幅度区间和刻度
> h.set(xlim=(0, 5), ylim=(0, 5), xticks=[0, 2.5, 5], yticks=[0, 2.5, 5])
```

图形

```
> plt.title("A Title") # 添加图形标题

> plt.ylabel("Survived") # 调整 Y 轴标签
> plt.xlabel("Sex") # 调整 X 轴标签

> plt.ylim(0, 100) # 调整 Y 轴幅度区间
> plt.xlim(0, 10) # 调整 X 轴幅度区间

> plt.setp(ax, yticks=[0, 5]) # 调整图形属性
> plt.tight_layout() # 调整子图参数
```


5. 显示或保存图形

show 与 savefig 函数

```
> plt.show() # 显示图形
> plt.savefig("foo.png") # 将画布保存为图形
> plt.savefig("foo.png", transparent=True) # 保存透明画布
```

6. 关闭与清除

绘图关闭与清除操作

```
> plt.cla() # 清除坐标轴
> plt.clf() # 清除画布
> plt.close() # 关闭窗口
```