

Python For Data Science Cheat Sheet

Seaborn

Learn Data Science Interactively at www.datacamp.com



Statistical Data Visualization With Seaborn

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

Make use of the following aliases to import the libraries:

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

The basic steps to creating plots with Seaborn are:

- 1. Prepare some data
- 2. Control figure aesthetics
- 3. Plot with Seaborn
- 4. Further customize your plot

```
>>> import matplotlib.pyplot as plt
>>> import seaborn as sns
>>> tips = sns.load_dataset("tips")
>>> sns.set_style("whitegrid")
>>> g = sns.lmplot(x="time", y="total_bill", data=tips, aspect=2)
>>> g = (g.set_axis_labels("Time", "Total bill (USD)"))
>>> plt.title("Title")
>>> plt.show()
```

1 Data

Also see [Lists](#), [NumPy](#) & [Pandas](#)

```
>>> 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 also offers built-in data sets:

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

2 Figure Aesthetics

```
>>> f, ax = plt.subplots(figsize=(5, 6))
```

```
>>> sns.set()
>>> sns.set_style("whitegrid")
>>> sns.set_style("ticks",
>>>               {'xtick.major.size': 8,
>>>                'ytick.major.size': 8})
>>> sns.axes_style("whitegrid")
```

(Re)set the seaborn default
Set the matplotlib parameters
Set the matplotlib parameters
Return a dict of params or use with
with to temporarily set the style

3 Plotting With Seaborn

Axis Grids

```
>>> g = sns.FacetGrid(titanic,
>>>                   col="survived",
>>>                   row="sex")
>>> g = g.map(plt.hist, "age")
>>> sns.factorplot(x="pclass",
>>>                y="survived",
>>>                hue="sex",
>>>                data=titanic)
>>> sns.lmplot(x="sepal_width",
>>>             y="sepal_length",
>>>             hue="species",
>>>             data=iris)
```

Subplot grid for plotting conditional relationships
Draw a categorical plot onto a FacetGrid
Plot data and regression model fits across a FacetGrid

```
>>> h = sns.PairGrid(iris)
>>> h = h.map(plt.scatter)
>>> sns.pairplot(iris)
>>> i = sns.JointGrid(x="x",
>>>                  y="y",
>>>                  data=data)
>>> i = i.plot(sns.regplot,
>>>            sns.distplot)
>>> sns.jointplot("sepal_length",
>>>               "sepal_width",
>>>               data=iris,
>>>               kind='kde')
```

Subplot grid for plotting pairwise relationships
Plot pairwise bivariate distributions
Grid for bivariate plot with marginal univariate plots
Plot bivariate distribution

Categorical Plots

```
>>> sns.stripplot(x="species",
>>>                y="petal_length",
>>>                data=iris)
>>> sns.swarmplot(x="species",
>>>                y="petal_length",
>>>                data=iris)
>>> sns.barplot(x="sex",
>>>              y="survived",
>>>              hue="class",
>>>              data=titanic)
>>> sns.countplot(x="deck",
>>>                data=titanic,
>>>                palette="Greens_d")
>>> sns.pointplot(x="class",
>>>                y="survived",
>>>                hue="sex",
>>>                data=titanic,
>>>                palette={"male": "g",
>>>                          "female": "m"},
>>>                markers={"^", "o"},
>>>                linestyle=["-", "--"])
>>> sns.boxplot(x="alive",
>>>              y="age",
>>>              hue="adult_male",
>>>              data=titanic)
>>> sns.boxplot(data=iris, orient="h")
>>> sns.violinplot(x="age",
>>>                 y="sex",
>>>                 hue="survived",
>>>                 data=titanic)
```

Scatterplot with one categorical variable
Categorical scatterplot with non-overlapping points
Show point estimates and confidence intervals with scatterplot glyphs
Show count of observations
Show point estimates and confidence intervals as rectangular bars
Boxplot
Boxplot with wide-form data
Violin plot

Regression Plots

```
>>> sns.regplot(x="sepal_width",
>>>              y="sepal_length",
>>>              data=iris,
>>>              ax=ax)
```

Plot data and a linear regression model fit

Distribution Plots

```
>>> plot = sns.distplot(data.y,
>>>                      kde=False,
>>>                      color="b")
```

Plot univariate distribution

Matrix Plots

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

Heatmap

4 Further Customizations

Also see [Matplotlib](#)

Axisgrid Objects

```
>>> g.despine(left=True)
>>> g.set_ylabels("Survived")
>>> g.set_xticklabels(rotation=45)
>>> g.set_axis_labels("Survived",
>>>                   "Sex")
>>> h.set(xlim=(0, 5),
>>>        ylim=(0, 5),
>>>        xticks=[0, 2.5, 5],
>>>        yticks=[0, 2.5, 5])
```

Remove left spine
Set the labels of the y-axis
Set the tick labels for x
Set the axis labels
Set the limit and ticks of the x-and y-axis

Plot

```
>>> plt.title("A Title")
>>> plt.ylabel("Survived")
>>> plt.xlabel("Sex")
>>> plt.ylim(0, 100)
>>> plt.xlim(0, 10)
>>> plt.setp(ax, yticks=[0, 5])
>>> plt.tight_layout()
```

Add plot title
Adjust the label of the y-axis
Adjust the label of the x-axis
Adjust the limits of the y-axis
Adjust the limits of the x-axis
Adjust a plot property
Adjust subplot params

5 Show or Save Plot

Also see [Matplotlib](#)

```
>>> plt.show()
>>> plt.savefig("foo.png")
>>> plt.savefig("foo.png",
>>>             transparent=True)
```

Show the plot
Save the plot as a figure
Save transparent figure

Close & Clear

Also see [Matplotlib](#)

```
>>> plt.cla()
>>> plt.clf()
>>> plt.close()
```

Clear an axis
Clear an entire figure
Close a window

