Python For Data Science Cheat Sheet

Seaborn

Learn Data Science Interactively at www.DataCamp.com



With Seaborn Statistical Data Visualization

matplotlib and provides a high-level interface for drawing The Python visualization library Seaborn is based on 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:

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

```
>>> g = (q.set axis labels("Tip","Total bill(USD)").
set(xlim=[0,10),ylim=[0,100)))
                                                                                                                                                                                                                                          Step 4
                                                        Step 1
>>> import matplotlib.pyplot as plt
                                                     >>> tips = sns.load dataset("tips")
                                                                                                       >>> g = sns.lmplot(x="tip",
y="total_bill",
                                                                               >>> sns.set_style("whitegrid")
                                                                                                                                                   data=tips,
                                                                                                                                                                             aspect=2)
                              >>> import seaborn as sns
                                                                                                                                                                                                                                                    >>> plt.title("title")
                                                                                                                                                                                                                                                                             >>> plt.show(g)
```

so see Lists, NumPy & Panda:

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

Seaborn also offers built-in data sets:

>>> titanic = sns.load dataset("titanic")
>>> tris = sns.load dataset("tris")

>>> f, ax = plt.subplots(figsize=(5,6)) | Create a figure and one subplot

Seaborn styles

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

(Re)set the seaborn default Set the matplotlib parameters Set the matplotlib parameters

Color Palette

Return a dict of params or use with with to temporarily set the style

3 Plotting With Seaborn

Axis Grids

Subplot grid for plotting conditional relationships Plot data and regression model fits Draw a categorical plot onto a across a FacetGrid >>> g = sns.FacetGrid(titanic, col="survived", hue="sex", data=titanic) row="sex") sns.lmplot(x="sepal_width", y="sepal_length", y="survived", g = g.map(plt.hist,"age")
sns.factorplot(x="pclass", hue="species", data=iris) **^** Ŷ Ŷ

relationships Plot pairwise bivariate distributions Grid for bivariate plot with marginal

univariate plots

data=data)

Plot bivariate distribution

>>> sns.jointplot("sepal_length", "sepal_width",

sns.distplot)

>>> i = i.plot(sns.regplot,

kind='kde')

data=iris,

Subplot grid for plotting pairwise

>>> h = sns.PairGrid(iris) >>> h = h.map(plt.scatter) >>> sns.pairplot(iris) >>> 1 = sns.JointGrid(x="x",

Categorical Plots

y="petal_length", data=iris) sns.swarmplot(x="species", y="petal_length", >> sns.stripplot(x="species", data=iris) y="survived", data=titanic) hue="class", >> sns.barplot(x="sex", Scatterplot Count Plot Bar Chart ŝ

data=titanic, >> sns.countplot(x="deck",

palette="Greens d") >> sns.pointplot(x="class", Point Plot

"female":"m"), linestyles=["-","--"]) palette=("male":"g", markers=["^","o"], y="survived", data=titanic, hue="sex",

>> sns.boxplot(x="alive", "age"-V Boxplot

hue="adult male",

>>> sns.boxplot(data=iris,orient="h") data=titanic) ->> sns.violinplot(x="age", Violinplot

hue="survived", data=titanic) y="sex",

Scatterplot with one categorical variable

Plot data and a linear regression

model fit

y="sepal length",

data=iris,

ax=ax)

Distribution Plots

>>> sns.regplot(x="sepal_width",

Regression Plots

Plot univariate distribution

kde=False, color="b")

>>> plot = sns.distplot(data.y,

Categorical scatterplot with non-overlapping points

Show point estimates and confidence intervals with scatterplot glyphs

Matrix Plots

>>> sns.heatmap(uniform data,vmin=0,vmax=1) | Heatmap

Further Customizations

Show count of observations

Show point estimates and confidence intervals as rectangular bars

soxplot

Boxplot with wide-form data

Violin plot

Set the limit and ticks of the x-and y-axis

Set the labels of the y-axis Set the tick labels for x

>>> g.set xticklabels(rotation=45)

>>> g.set_ylabels("Survived")

>>> g.despine(left=True)

>>> g.set_axis labels("Survived",

ylim=(0,5), xticks=[0,2.5,5],

>> h.set(xlim=(0,5),

Set the axis labels

Remove left spine

yticks=(0,2.5,5)) >>> plt.ylabel("Survived") >>> plt.xlabel("Sex") >>> plt.title("A Title")

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 >>> plt.setp(ax,yticks=[0,5]) >>> plt.tight layout() >>> plt.ylim(0,100) >>> plt.xlim(0,10)

5 Show or Save Plot

Also see Matolotlib

Show the plot Save the plot as a figure Save transparent figure transparent=True) >>> plt.show()
>>> plt.savefig("foo.png")
>>> plt.savefig("foo.png",

Close & Clea

Set context to "talk"
Set context to "notebook"
Scale font elements and
override param mapping

>>> sns.set_context("talk")
>>> sns.set_context("notebook",
font_scale=1.5,
ro={"lines.linewidth":2.5})

Context Functions

Also see Matplotlib

>>> plt.cla() >>> plt.clf() >>> plt.clf()

Use with with to temporarily set palette

>>> sns.set_palette("husl",3)
>>> sns.color_palette("husl")
>>> flatul = ["#9b59b6","#3498db","#9581

Define the color palette

,"#2ecc71"]

se", "#e74e3e", "#34495e", Set your own color palette

Clear an axis Clear an entire figure Close a window

DataCamp Leam Python for Data Science

