

Seaborn - 1 feature

```
_, axes = plt.subplots(    put a sns plot in a pyplot figure
    nrows=1, ncols=2,
    figsize=(12, 4))
axis[0] = sns.xxx
```

```
sns.distplot(            histogram + density of a numeric
    df['feat']);          feature's repartition
```

```
sns.boxplot(            simple boxplot
    data=df['feat']);
```

```
sns.violinplot(        simple violin plot
    data=df['feat']);
```

```
sns.countplot(         repartition of a
    x='feat', data=df);  categorical feature
```

Seaborn - 2 features

```
sns.heatmap(matrix)    heatmap
```

```
sns.jointplot(         joint plot
    x=feat1,
    y=feat2,
    data=df,
    kind=['scatter', 'kde'])
```

```
sns.pairplot(df[feats]) scatterplots matrix
```

```
sns.boxplot(x=feat1, y=feat2,    boxplot for disjoint groups
    data=df, ax=ax)              (x categorical)
```

```
sns.violinplot(x=feat1,         violin plot for disjoint
    y=feat2, data=df, ax=ax)     groups
```

Seaborn - 2 features (cont)

```
sns.countplot(         repartition of a
    x='feat1',            categorical feature according to another one
    hue='feat2',
    data=df);
```

```
sns.lmplot(            scatterplot with color according to category
    feat1,
    feat2,
    data=df,
    hue=feat3,
    fit_reg=False)
```

```
sns.factorplot(        analyze a quantitative variable in 2 categorical
    x=cat1,              dimensions at once
    y=numeric,
```

```
    col=cat2,
    data=df,
    kind="box",
    col_wrap=4,
    size=,
    aspect=);
```

Dimensionality reduction - t-SNE

```
from sklearn.manifold import TSNE
from sklearn.preprocessing
import StandardScaler

# Standardize data
scaler = StandardScaler()
tab_scaled = scaler.fit_transform(tab)

# Run t-SNE
tsne = TSNE(random_state=17)
tsne_repr = tsne.fit_transform(tab_scaled)

# Show results
plt.scatter(tsne_repr[:, 0], tsne_repr[:, 1],
    c=df[feat].map({False: 'green', True: 'red'}));
```



Plotly

```
from plotly.offline import download_plotlyjs,
init_notebook_mode, plot, iplot
import plotly
import plotly.graph_objs as go
```

```
trace0 = go.xxx                                general syntax
```

```
data = [trace0, trace1, ...]
layout = {'title': title, ..}
fig = go.Figure(data=data,
layout=layout)
iplot(fig, show_link=False)
```

```
trace = go.Scatter(x=feat1,                    line
y=feat2, name=)
```

```
trace = go.Bar(x=feat1, y=feat2 ,            barchart
name=)
```

```
trace = go.Box(y=feat, name=genre)          boxplot
```

```
plotly.offline.plot(                          Save figure as an
fig,                                           html file
filename='file.html',
show_link=False)
```



By **Anoikis**
cheatography.com/anoikis/

Not published yet.
Last updated 16th February, 2019.
Page 2 of 2.

Sponsored by **Readability-Score.com**
Measure your website readability!
<https://readability-score.com>