Data Visualization

Prof. Dr. Noman Islam

Introduction

- *matplotlib is a desktop plotting package designed for creating (mostly twodimensional) publication-quality plots.*
- *Over time, matplotlib has spawned a number of add-on toolkits for data visualization that use matplotlib for their underlying plotting. One of these is seaborn*

Preparing for visualization

- *import matplotlib as mpl*
- *Import matplotlib.pyplot as plt*
- *plt.style.use('default')*
- *%matplotlib inline*

Plotting sin function

- x = np.arange(0, math.pi*2, 0.05)
- y = np.sin(x)
- plt.xlabel("angle")
- plt.ylabel("sine")
- plt.title('sine wave')
- plt.grid(**True**, which='both')
- plt.plot(x,y)

Plotting lines

- x = np.linspace(0, 20)
- plt.plot(x, .5 + x)
- plt.plot(x, 1 + 2 * x, '--‘)

Customizing plots

- *plt.plot(x, .5 + x, color='blue') # specify color by name*
- *plt.plot(x, 1 + 2 * x, '--', color='#FFDD44') # Hex code (RRGGBB from 00 to FF)*
- *plt.plot(x, np.cos(x), (1.0,0.2,0.3)) # RGB tuple, values 0 and 1*

Customizing plots

- *x = np.linspace(0, 10, 1000)*
- *plt.plot(x, np.sin(x), linestyle='solid')*
- *plt.plot(x, np.cos(x), linestyle='dotted')*

Specifying limits

- *plt.plot(x, np.tan(x))*
- *plt.xlim(-1, 11)*
- *plt.ylim(-1.5, 1.5);*

Labeling axis

- *plt.title("titlte")*
- *plt.xlabel("x-axis")*
- *plt.ylabel("y-axis")*

Plotting legends

- *plt.plot(x, np.sin(x), '-g', label='sin(x)')*
- *plt.plot(x, np.cos(x), ':b', label='cos(x)')*
- *plt.legend()*

Scatter plot

- from sklearn.datasets import load_iris
- iris = load_iris()
- features = iris.data.T
- plt.scatter(features[0], features[1], alpha=0.2, s=100*features[3], c=iris.target, cmap='viridis')
- plt.xlabel(iris.feature_names[0])
- plt.ylabel(iris.feature_names[1])

Bar plot

- import matplotlib.pyplot as plt
- %matplotlib inline
- plt.style.use('ggplot')
- x = ['Viettel', 'VNPT', 'Mobiphone']
- revenue = (37600000, 6445000, 6045000)
- x_pos = [i for i, _ in enumerate(x)]
- plt.bar(x_pos, revenue, color='green')
- plt.xlabel("Telco")
- plt.ylabel("VND")
- plt.title("Telecom service revenues in Vietnam")
- plt.xticks(x_pos, x)
- plt.show()

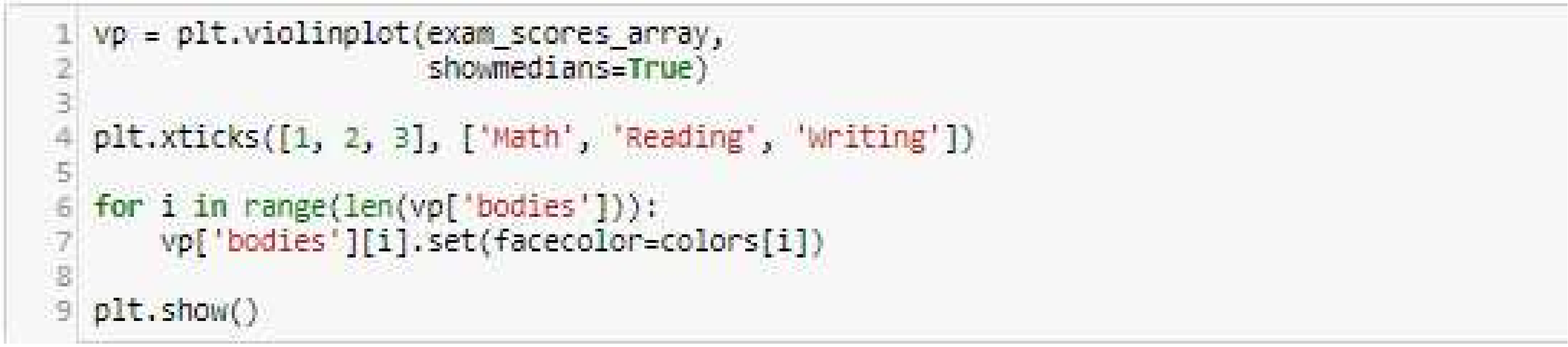
Plotting histogram

- *import numpy as np*
- *import matplotlib.pyplot as plt*
- *X = np.random.randn(1000)*
- *plt.hist(X, bins = 20)*
- *plt.show()*

Box plot

```
import numpy as np
import matplotlib.pyplot as plt
data = np.random.randn(100)
plt.boxplot(data)
plt.show()
```

Violin plot

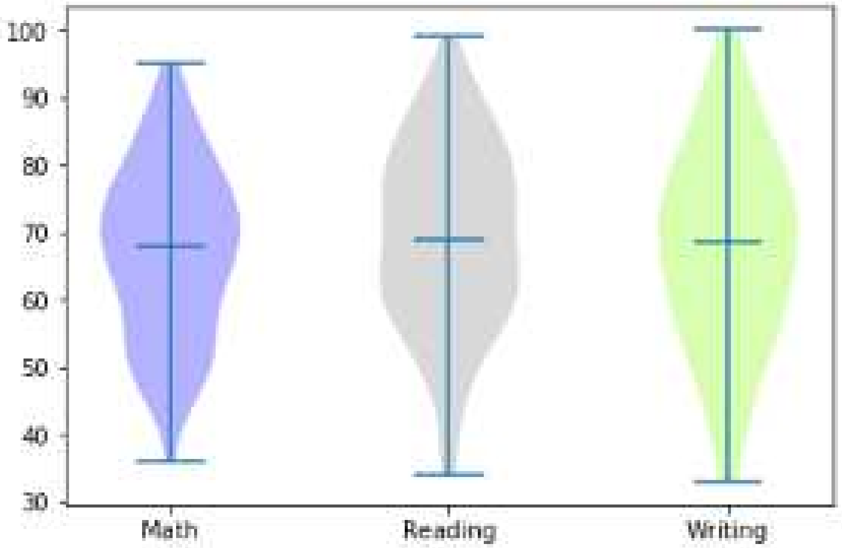


```
vp = plt.violinplot(exam_scores_array,
                    showmedians=True)

plt.xticks([1, 2, 3], ['Math', 'Reading', 'Writing'])

for i in range(len(vp['bodies'])):
    vp['bodies'][i].set(facecolor=colors[i])

plt.show()
```



Plotting pie chart

- *import matplotlib.pyplot as plt*
- *telcos = 'Viettel', 'VNPT', 'Mobiphone', 'Vietnammobile', 'Gtel'*
- *data = [46.7, 22.2, 26.1, 2.9, 2.1]*
- *fig1, ax1 = plt.subplots()*
- *ax1.pie(data, labels=telcos, autopct='%1.1f%%', shadow=True, startangle=90)*
- *ax1.axis('equal')*
- *plt.show()*

Saving figure

- *fig1.savefig('my_figure.png')*
- *fig1.savefig('my_figure.png', transparent=True)*

Figures and subplot

- *Plots in matplotlib reside within a Figure object*
  - *fig = plt.figure()*
- *You can’t make a plot with a blank figure. You have to create one or more subplots using add_subplot:*
  - *ax1 = fig.add_subplot(2, 2, 1)*
  - *plt.plot(np.random.randn(50).cumsum(), 'k--')*

- *The 'k--' is a style option instructing matplotlib to plot a black dashed line.*
- *The objects returned by fig.add_subplot here are AxesSubplot objects, on which you can directly plot on the other empty subplots by calling each one’s instance method*

- x = np.linspace(0, 2*np.pi, 400)
- y = np.sin(x**2)
- f, (ax1, ax2) = plt.subplots(1, 2)
- ax1.plot(x, y)
- ax1.set_title('Sharing Y axis')
- ax2.scatter(x, y)

- *fig, axes = plt.subplots(2, 3)*
- *ax.plot(x, y, 'g--')*
- *ax.plot(x, y, linestyle='--', color='g')*
- *plt.plot(randn(30).cumsum(), 'ko--')*

Plotting with pandas

- *s = pd.Series(np.random.randn(10).cumsum(), index=np.arange(0, 100, 10))*
- *s.plot()*

- *df = pd.DataFrame(np.random.randn(10, 4).cumsum(0), columns=['A', 'B', 'C', 'D'], index=np.arange(0, 100, 10))*
- *df.plot()*
- *df.plot.bar()*

Seaborn

- *import seaborn as sns*
- *sns.set()*
- *plt.plot(x, y)*
- *plt.legend('ABCDEF', ncol=2, loc='upper left')*

Plotting histogram

- *data = np.random.multivariate_normal([0, 0], [[5, 2], [2, 2]], size=2000)*
- *data = pd.DataFrame(data, columns=['x', 'y'])*
- *for col in 'xy':*
  - *plt.hist(data[col], normed=True, alpha=0.5)*

Kernel density plot

- *for col in 'xy':*
  - *sns.kdeplot(data[col], shade=True)*

Distplot

- *sns.distplot(data['x'])*
- *sns.distplot(data['y']);*

Pair plot

- *iris = sns.load_dataset("iris")*
- *iris.head()*
- *sns.pairplot(iris, hue='species', size=2.5);*

Factor plot

- *tips = sns.load_dataset('tips')*
- *tips.head()*
- *with sns.axes_style(style='ticks'):*
  - *g = sns.factorplot("day", "total_bill", "sex", data=tips, kind="box")*
  - *g.set_axis_labels("Day", "Total Bill");*

Scatter plot

- *sns.relplot(x="Views", y="Upvotes", data = df)*
- *sns.relplot(x="Views", y="Upvotes", hue = "Tag", data = df)*
- *sns.relplot(x="Views", y="Upvotes", hue = "Answers", data = df);*
-

Box plot

- *sns.catplot(x="education", y="avg_training_score", kind = "box", data=df2)*

Violin plot

- *sns.catplot(x="education", y="avg_training_score", hue = "is_promoted", kind = "violin", data=df2)*

Heatmaps

- *corrmat = df2.corr()*
- *f, ax = plt.subplots(figsize=(9, 6))*
- *sns.heatmap(corrmat, vmax=.8, square=True)*

Other tools

- *As is common with open source, there are a plethora of options for creating graphics in Python*
- *With tools like Bokeh and Plotly, it’s now possible to specify dynamic, interactive graphics in Python that are destined for a web browser*