✏️

Chapter 2: Plotting time-series

Created	@May 7, 2024 4:59 PM
Class	Introduction to Data Visualization with Matplotlib

Visualizing Time-Series Data

Time-Series Data

- Data organized by time (e.g. weather data with temperatures/precipitation by month)

Dataset

- Climate data from 1958-2016
- Columns: date, co2 (ppm), relative temp

Loading Data

```
data = pd.read_csv('climate_data.csv', parse_dates=['date'],
index_col='date')
```

- Parses date column as dates
- Sets date as index for time-series indexing

Plotting

```
fig, ax = plt.subplots()
ax.plot(data.index, data['co2'])
ax.set_xlabel('Date')
ax.set_ylabel('CO2 (ppm)')
```

- Plots CO_2 vs date
- Matplotlib auto scales x-axis ticks by years

Slicing

```
decade = data['1980-01-01':'1989-12-31']
year = data['1980-01-01':'1980-12-31']
```

- Slice by date range
- X-ticks change based on slice period

Missing Data

- Represented as breaks in line plot

Plotting Multiple Time-Series on Same Axes

Relating Time-Series

- To relate time-series that coincide in time but measure different variables

Example: Climate Data

- DataFrame with co2 levels and relative temperatures from 1958-2016
- Plot them on the same Axes

Plotting Both Variables

```
fig, ax = plt.subplots()
ax.plot(data.index, data['co2'])
ax.plot(data.index, data['relative_temp'])
ax.set_xlabel('Date')
ax.set_ylabel('CO2 (ppm)')
plt.show()
```

- Issue: Different scales, co2 line shifted up, temp line flat

Using Twin Axes

```
fig, ax = plt.subplots()
ax.plot(data.index, data['co2'])
ax.set_xlabel('Time')
ax.set_ylabel('CO2_ppm')
ax2 = ax.twinx()# Creates twin of ax
ax2.plot(data.index, data['relative_temp'])
ax2.set_ylabel('Relative temperature (Celsius)')
plt.show()
```

- Shares x-axis but separate y-axes

Color-Coding

```
ax.plot(data.index, data['co2'], color='blue')
ax.set_ylabel('CO2 (ppm)', color='blue')
ax2.plot(data.index, data['relative_temp'], color='red')
ax2.set_ylabel('Relative Temperature', color='red')
ax.tick_params(colors='blue', axis='y')
ax2.tick_params(colors='red', axis='y')
```

- Use color to distinguish variables and axes

Reusable Function

```
def plot_timeseries(axes, x, y, color, xlabel, ylabel):
    axes.plot(x, y, color=color)
    axes.set_xlabel(xlabel)
    axes.set_ylabel(ylabel, color=color)
    axes.tick_params(colors=color, axis='y')
```

- Implement as reusable function

Adding Annotations to Visualizations

Annotations

- Small text referring to particular part of visualization
- Focus attention on features of data

Example

- Climate data with co2 levels and relative temperatures over 50 years

Annotating a Data Point

```
x.annotate(">1 degree",
            xy=(pd.Timestamp('2015-10-06'), 1),
            xytext=(pd.Timestamp('2008-10-06'), -0.2))
```

- `annotate()` takes annotation text and xy coordinates
- `xytext` sets text position

Adding Arrow

```
ax.annotate(...,
            arrowprops={})
```

- `arrowprops` dict to customize arrow properties

- Empty dict gives default arrow

```
ax.annotate(...,
            arrowprops={'arrowstyle': '->', 'color': 'gray'})
```

- Customize arrow style and color

Annotation Options

- Many options for customizing annotations
- See Matplotlib documentation