Visualizing Data

Lecture 3

Centre for Data Science, ITER
Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India.

Contents

1. Introduction
2. Matplotlib
3. Bar Charts
4. Line Charts
5. Scatter Plots

Introduction

- A fundamental part of the data scientist’s toolkit is data visualization.
- Data visualization is the graphical representation of information and data.
- By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.
- There are two primary uses for data visualization:
    - To explore data
    - To communicate data

Matplotlib

- For simple bar charts, line charts, and scatterplots Matplotlib is useful.
- Module **matplotlib.pyplot** of matplotlib is used for visualization of data.
- **pyplot** maintains an internal state in which you build up a visualization **step by step.**
- Once you're done, you can save it with **savefig** or display it with **show**.
- A simple line plot is showen in figure(1).

Example

- Code for a simple line plot of year v/s gdp

```
from matplotlib import pyplot as plt
years = [1950, 1960, 1970, 1980, 1990, 2000, 2010]
gdp = [300.2, 543.3, 1075.9, 2862.5, 5979.6, 10289.7, 14958.3]
’’’create a line chart, years on x-axis, gdp on y-axis’’’
plt.plot(years, gdp, color=’green’, marker=’o’, linestyle=’solid’)
’’’add a title’’’
plt.title("Nominal GDP")
’’’add a label to the y-axis’’’
plt.ylabel("Billions of $")
plt.xlabel("Year")
plt.show()
plt.savefig()
```

Simple Line Plot

- Output of above code.

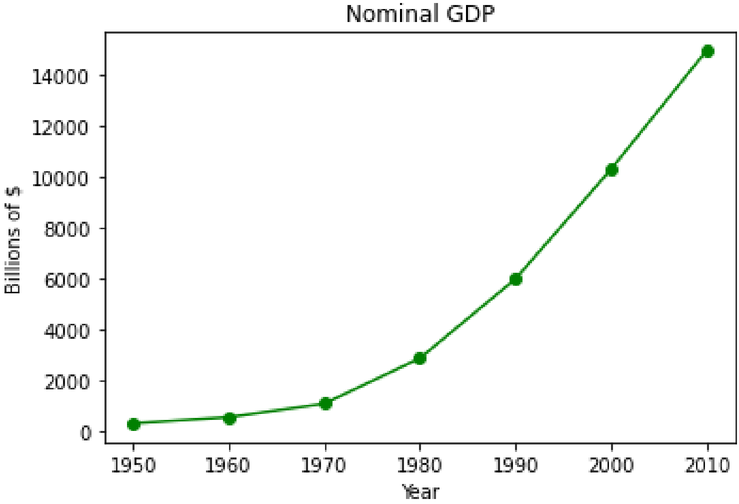

Figure 1: A Simple Line Plot of year v/s gdp

Bar Charts

- **A bar chart** is a good choice when you want to show how some quantity varies among some discrete set of items.
- Academy Awards were won by each of a variety of movies is shown in figure(2) and code is given below.

```
from matplotlib import pyplot as plt
movies = ["Annie Hall", "Ben-Hur", "Casablanca", "Gandhi",
    "West Side Story"]
num_oscars = [5, 11, 3, 8, 10]
''' plot bars with left x-coordinates [0, 1, 2, 3, 4],
    heights [num_oscars] '''
plt.bar(range(len(movies)), num_oscars)
plt.title("My Favorite Movies")
plt.ylabel("# of Academy Awards")
''' label x-axis with movie names at bar centers '''
plt.xticks(range(len(movies)), movies)
plt.show()
```

Bar Charts Cont...

- Output of above code.

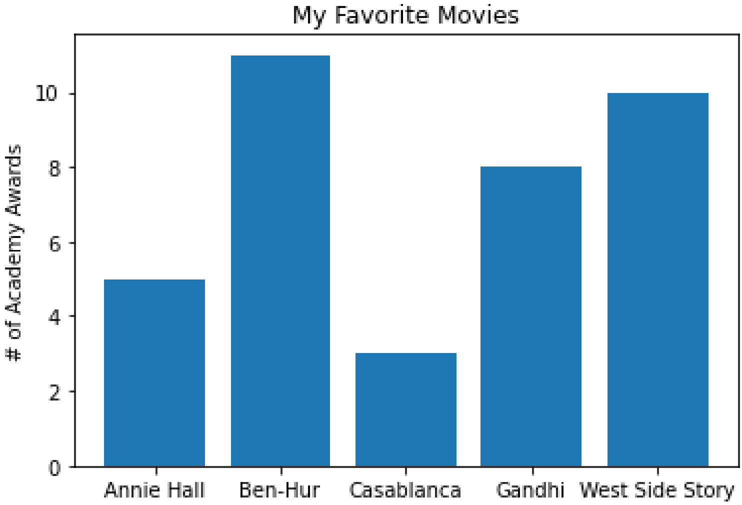

Figure 2: A simple bar chart

Bar Charts Cont...

- A bar chart can also be a good choice for plotting **histograms** of bucketed numeric values as shown in figure(3) and code is given below.
- The third argument to **plt.bar** specifies the bar width.
- The call to **plt.axis** indicates that we want the x-axis to range from –5 to 105, and that the y-axis should range from 0 to 5.
- And the call to **plt.xticks** puts x-axis labels at 0, 10, 20, . . . , 100.

Bar Charts Cont...

- Code for histogram using **plt.bar** .

```python
from collections import Counter
from matplotlib import pyplot as plt
grades=[83, 95, 91, 87, 70, 0, 85, 82, 100, 67, 73, 77, 0]
'''Bucket grades by decile, but put 100 in with the 90s'''
histogram = Counter(min(grade // 10 * 10, 90) for grade in grades)
plt.bar([x + 5 for x in histogram.keys()], '''Shift bars right by 5'''
histogram.values(), '''Give each bar its correct height '''
10, '''Give each bar a width of 10 '''
edgecolor=(0, 0, 0)) '''Black edges for each bar '''
plt.axis([-5, 105, 0, 5]) '''x-axis from -5 to 105,'''
plt.xticks([10 * i for i in range(11)]) '''x-axis labels at 0, 10,...,100'''
plt.xlabel("Decile")
plt.ylabel("# of Students")
plt.title("Distribution of Exam 1 Grades")
plt.show()
```

Bar Charts Cont...

- Output of above code.

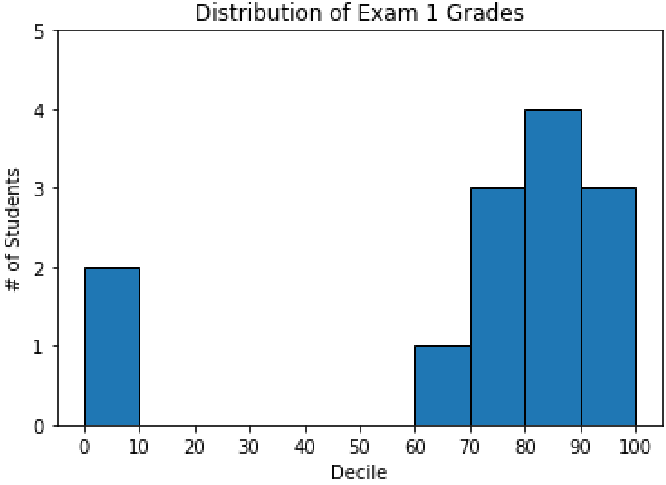

Figure 3: Using a bar chart for a histogram

Line Charts

- **Line charts** can be obtained using **plt.plot**.
- These are a good choice for showing trends.

Line Charts Cont...

- Several line charts with a legend.

```
from matplotlib import pyplot as plt
variance = [1, 2, 4, 8, 16, 32, 64, 128, 256]
bias_squared = [256, 128, 64, 32, 16, 8, 4, 2, 1]
total_error=[x + y for x, y in zip(variance, bias_squared)]
xs = [i for i, _ in enumerate(variance)]
’’’ We can make multiple calls to plt.plot to show multiple series on the same chart ’’’
plt.plot(xs, variance, ’g-’, label=’variance’) ’’’ green solid line ’’’
plt.plot(xs, bias_squared, ’r-.’, label=’bias^2’) ’’’ red dot-dashed line ’’’
plt.plot(xs, total_error, ’b:’, label=’total error’) ’’’ blue dotted line ’’’
’’’ Because we have assigned labels to each series, we can get a legend for free (loc=9 means "top center") ’’’
plt.legend(loc=9)
plt.xlabel("model complexity")
plt.xticks([])
plt.title("The Bias-Variance Tradeoff")
plt.show()
```

Line Charts Cont...

- Output of above code.

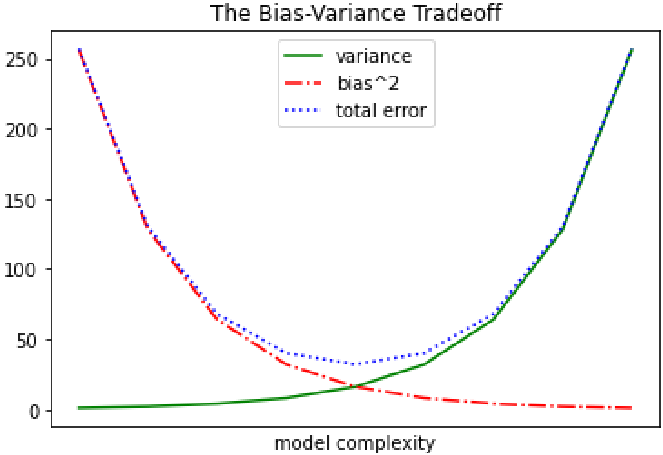

Figure 4: Several line charts with a legend

Scatter Plots

- **A scatterplot** is the right choice for visualizing the relationship between two paired sets of data.
- The relationship between the number of friends your users have and the number of minutes they spend on the site every day can be obtained using given code and its visualization is shown in figure(5).

Scatter Plots Cont...

- Code

```
from matplotlib import pyplot as plt
friends = [ 70, 65, 72, 63, 71, 64, 60, 64, 67]
minutes = [175, 170, 205, 120, 220, 130, 105, 145, 190]
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
plt.scatter(friends, minutes)
for label, friend_count, minute_count in zip(labels,
    friends, minutes):
    plt.annotate(label,
    xy=(friend_count, minute_count), ''' Put the label with
     its point '''
    xytext=(5, -5), ''' but slightly offset '''
    textcoords='offset points')
plt.title("Daily Minutes vs. Number of Friends")
plt.xlabel("# of friends")
plt.ylabel("daily minutes spent on the site")
plt.show()
```

Scatter Plot Cont...

- Output of above code.

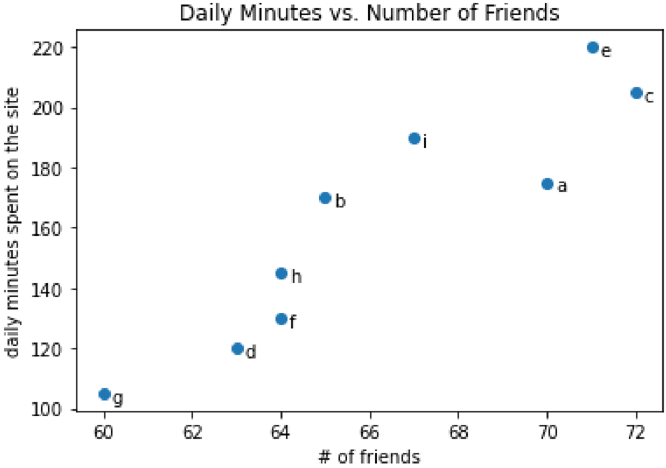

Figure 5: A scatterplot of friends and time on the site

References

[1] Data Science from Scratch Joel Grus, Shroff/O'reilly, Second Edition

Thank You
Any Questions?