Queen Mary
University of London

IOT607U Data Mining

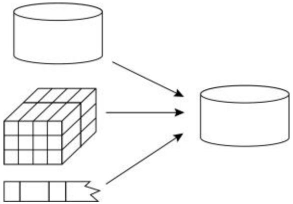
Week 3: Data exploration and visualization

Dr Lin Wang

School of EECS, Queen Mary University of London

Last week: Data

- Attributes and Objects
- Characteristics of Data
- Data Representations
- Basic Statistical Descriptions of Data
- Similarity and Distance

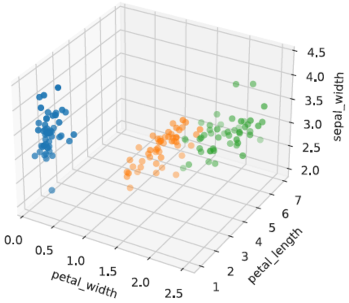


This week's contents

1. Data Exploration

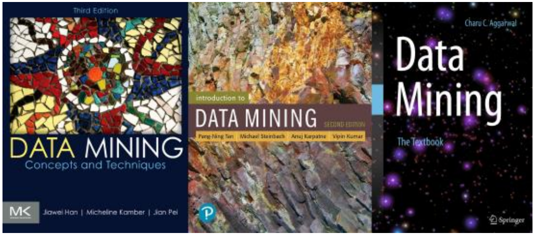
2. Data Summarization

3. Data Visualization



Reading

- **Chapter 2.3 of J. Han, M. Kamber, J. Pei, "Data Mining: Concepts and Techniques", 3rd edition, Elsevier/Morgan Kaufmann, 2012**
- Chapter "Data Exploration" of Section 2.3 of P.-N. Tan, M. Steinbach, A. Karpatne, V. Kumar, "Introduction to Data Mining", 2nd edition, Pearson, 2019



Data Exploration

Data Exploration

- Data exploration refers to a preliminary investigation of the data
- This investigation typically has the following goals:
    - Revealing the need for **data pre-processing**
    - Answering simple questions about the data
    - Identifying applicable data analysis techniques
- There are two main methods for data exploration: **data summarisation** and **data visualisation**

Data Summarization

Summarization techniques

Data Summarization

- Descriptive statistics
- Central tendency: mean, median, mode
- Dispersion: range, inter-quartile range, variance

Data Visualization

Purpose of Visualization

- **Exploratory data analysis:** what does my data look like?
- **Error detection:** anything weird in the data before you try to use it?
- **Communication:** can you explain the data? Or convince your audience about your point?
  - **e.g. data journalism:** reporting a story with charts.

Visualization Process

- Data **collection and cleansing**.
- **Research** on the data, e.g. get descriptive statistics.
- Produce **visualization that conveys those statistics**.
- Assess the **effectiveness of the visualization**.
  - Does it show what we intend to show?
  - Is it easy to interpret for the target reader?
  - Could it mislead readers?

Visualization Tools

- **Scatter plot:** exploring data
- **Time series:** time
- **Bar chart:** comparisons
- **Histogram:** binned frequency
- **Density:** continuous functions
- **Box plot:** quantiles
- **QQ plot:** comparing distributions
- **Other:** displaying multivariate

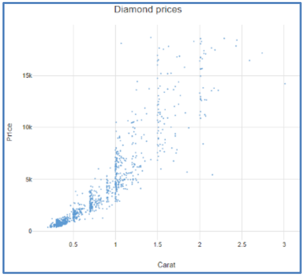
Data Visualization

Scatter Plot

Scatter Plot: Definition

- Using Cartesian coordinates to **display values for two variables** for a set of data

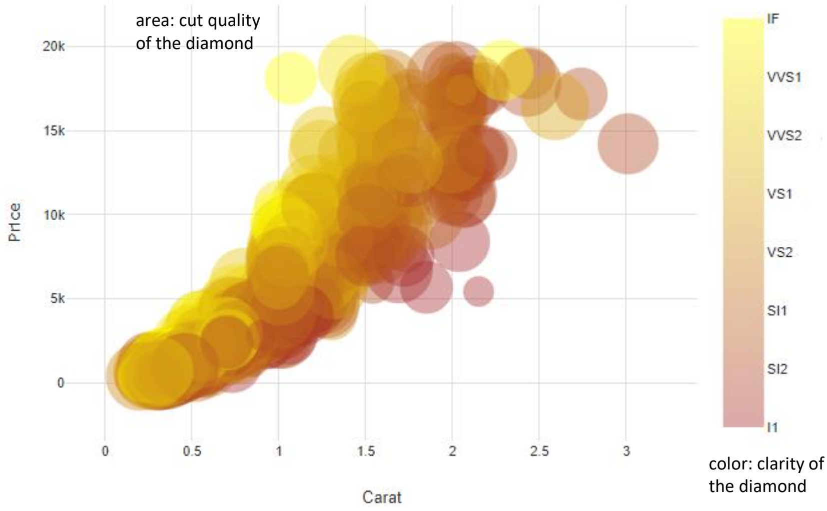
- The data is displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis.



Scatter Plot vs Bubble Plot

BUBBLE PLOT

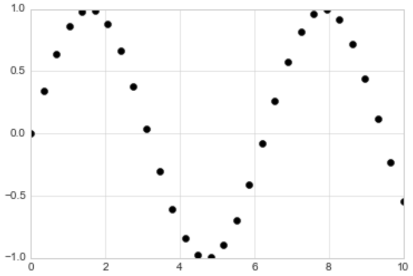
area: cut quality of the diamond



color: clarity of the diamond

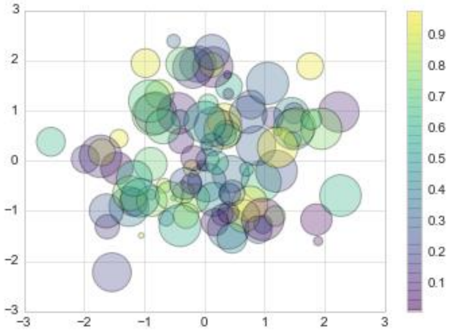
Scatter Plot in Python

```
x = np.linspace(0, 10, 30)
y = np.sin(x)
plt.plot(x, y, 'o', color='black')
```



Scatter Plot in Python

```
rng = np.random.RandomState(0)
x = rng.randn(100)
y = rng.randn(100)
colors = rng.rand(100)
sizes = 1000 * rng.rand(100)
plt.scatter(x, y, c=colors, s=sizes,
        alpha=0.3, cmap='viridis')
plt.colorbar()
```



aka bubble plot

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html

Scatter Plot: Adding Line of Fit

- **Line of best fit:**
  straight line that is the best approximation of the given set of data

- In a scatter plot, what is the linear tendency of my data?

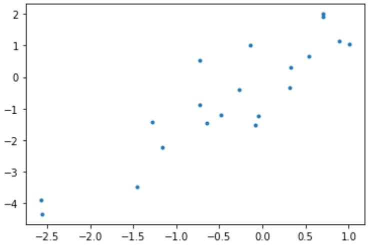
```
numpy.polyfit(x, y, deg)
```

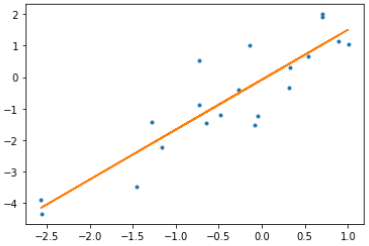
https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html

Scatter Plot: Adding Line of Fit

```
x = np.random.randn(20)
y = 2*x + np.random.randn(20)
plt.plot(x,y,'.')
```

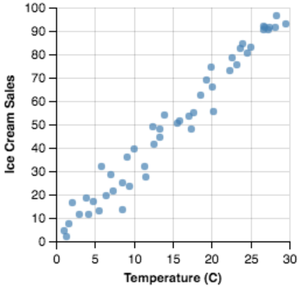
```
m,b = np.polyfit(x,y,1)
yy = m*x + b
plt.plot(x,yy)
```





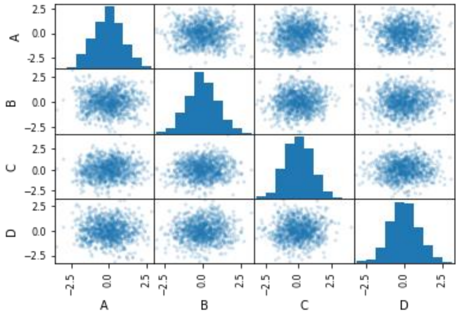
Scatter Plot: Discussion

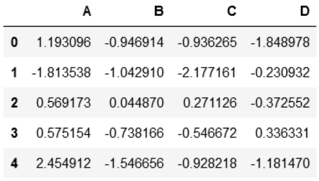
- Useful for **exploration of variables** and their **relationships**
- Shows the raw data: **outliers will show up**
- Link to descriptive statistics: provides an immediate feeling of
  - **Range**
  - **Dispersion**
  - **Correlation**



Scatter Plot Matrix

```python
import pandas as pd
df = pd.DataFrame(np.random.randn(1000, 4),
                  columns=['A','B','C','D'])
pd.plotting.scatter_matrix(df, alpha=0.2)
```





	A	B	C	D
0	1.193096	-0.946914	-0.936265	-1.848978
1	-1.813538	-1.042910	-2.177161	-0.230932
2	0.569173	0.044870	0.271126	-0.372552
3	0.575154	-0.738166	-0.546672	0.336331
4	2.454912	-1.546656	-0.928218	-1.181470

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.plotting.scatter_matrix.html

Data Visualization

Time Series

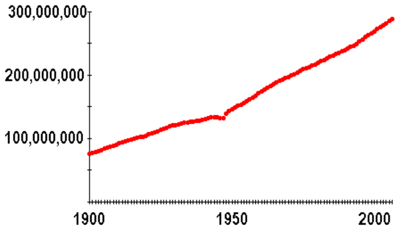
Time Series: Definition

- **Sequence of data points**, measured typically at **successive points in time** spaced at uniform time intervals

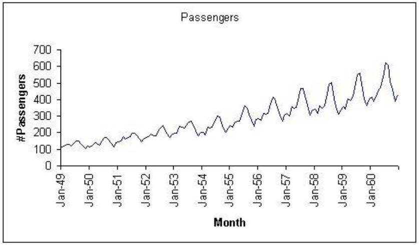
- Time series are used in statistics, signal processing, pattern recognition, econometrics, mathematical finance, weather forecasting, earthquake prediction, electroencephalography, control engineering, astronomy, communications engineering, and largely in any domain of applied science and engineering which involves temporal measurements.

Time Series: Example

```
plt.plot(x, y)
```



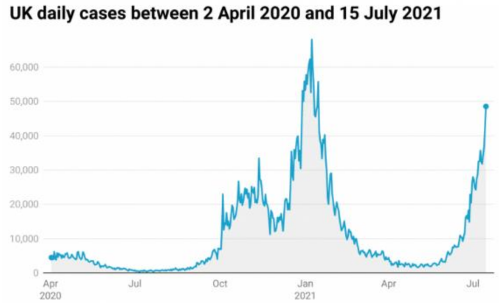
A time series graph of the population of the United States from the years 1900 to 2000



Monthly airline bookings of a flight company

Time Series: Discussion

- Trends and patterns over time
- Immediate feeling of
    - Noise vs. signal
    - Trends
    - Anomalies



Data Visualization

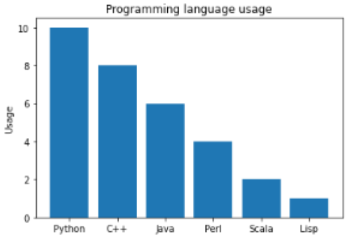
Bar Chart

Bar Chart: Definition

- **Rectangular bars** with lengths proportional to the values they represent.

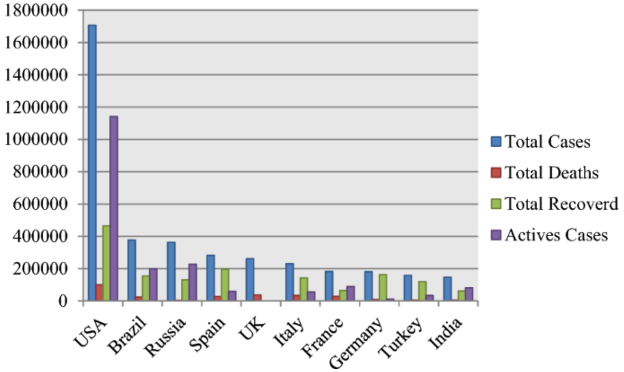
- Used to show **comparisons among categories**.

- Some bar graphs present bars clustered in groups of more than one (**grouped bar graphs**), and others show the bars divided into subparts to show cumulative effect (**stacked bar graphs**).



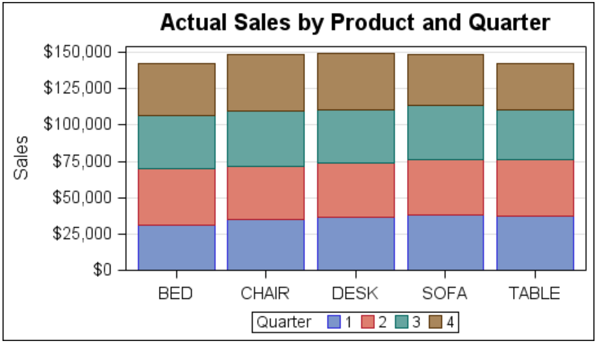
Grouped Bar Chart

Example of a grouped (clustered) bar chart



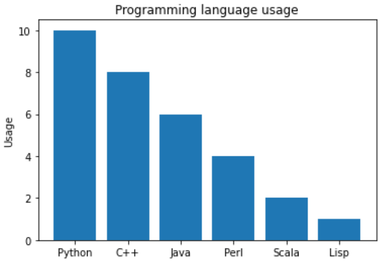
Stacked Bar Chart

Example of a stacked bar chart



Bar Chart in Python

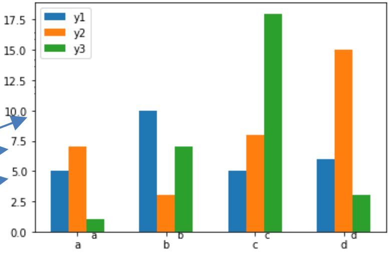
```
import matplotlib.pyplot as plt
import numpy as np
objects = ('Python', 'C++', 'Java', 'Perl', 'Scala', 'Lisp')
x_pos = np.arange(len(objects))
usage = [10, 8, 6, 4, 2, 1]
plt.bar(x_pos, usage)
plt.xticks(x_pos, objects)
plt.ylabel('Usage')
plt.title('Programming language usage')
```



https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar.html

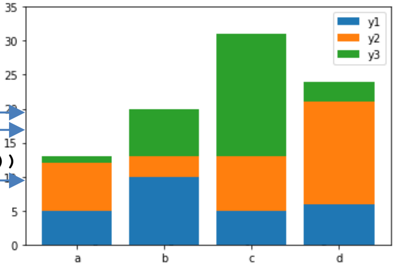
Grouped Bar Chart in Python

```
import matplotlib.pyplot as plt
import numpy as np
x = ['a', 'b', 'c', 'd']
y1 = [5, 10, 5, 6]
y2 = [7, 3, 8, 15]
y3 = [1, 7, 18, 3]

w = 0.2
x_pos = np.array(range(len(x)))
plt.bar(x_pos-0.2, y1, width=w)
plt.bar(x_pos, y2, width=w)
plt.bar(x_pos+0.2, y3, width=w)
plt.xticks(x_pos,x)
plt.legend(['y1','y2','y3’])
```



Stacked Bar Chart in Python

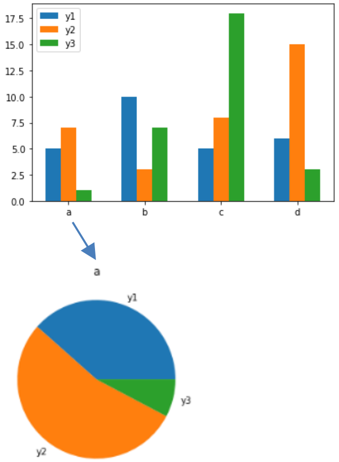
```
import matplotlib.pyplot as plt
import numpy as np
x = ['a', 'b', 'c', 'd']
y1 = [5, 10, 5, 6]
y2 = [7, 3, 8, 15]
y3 = [1, 7, 18, 3]
plt.bar(x, y1)
plt.bar(x, y2, bottom=y1)
z = list(np.array(y1) + np.array(y2))
plt.bar(x, y2, bottom=z)
```



Pie Chart in Python

PIE CHART IN PYTHON

```
import matplotlib.pyplot as plt
import numpy as np
x = ['a', 'b', 'c', 'd']
y1 = [5, 10, 5, 6]
y2 = [7, 3, 8, 15]
y3 = [1, 7, 18, 3]

za= [y1[0], y2[0], y3[0]]
plt.pie(za, labels=['y1','y2','y3'])
plt.title(x[0])
```



https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.pie.html

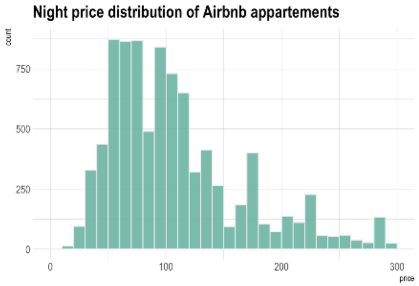
Data Visualization

Histogram

Histogram: Definition

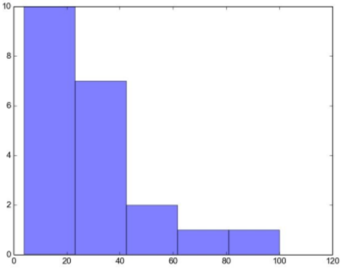
- Representation of **tabulated frequencies, shown as adjacent rectangles**, erected over discrete intervals (bins), with an area proportional to the frequency of the observations in the interval.

- The **height of a rectangle is also equal to the frequency density of the interval**, i.e., the frequency divided by the width of the interval. The total area of the histogram is equal to the number of data.



Histogram in Python

```
import matplotlib.pyplot as plt
x = [21, 22, 23, 4, 5, 6, 77, 8, 9, 10, 31, 32, 3
3, 34, 35, 36, 37, 18, 49, 50, 100]
num_bins = 5

[count, bin_boundary, patch] = plt.hist(x,
 bins=num_bins, facecolor='blue')


#bin_boundary: [4.0, 23.2, 42.4, 61.6, 80.8, 100.0]
#count: [10.0, 7.0, 2.0, 1.0, 1.0]
```



```
# can also use np.histogram + bar
[count2, bins2] = np.histogram(x, bins=5)
plt.bar((bins2[0:-
1]+bins2[1:])/2,count2,width=20)
```

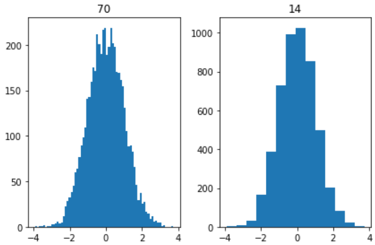
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html

Histogram: Discussion

- Can be hard to read due to the bin size choice
- No “best” number of bins: different bin sizes can reveal different features of the data
- Square root choice (used by Excel): $k = \sqrt{n}$
- Sturges (normal data): $k = [\log_2 n + 1]$

$k = \sqrt{n}$

$k = [\log_2 n + 1]$



$n = 5000$ normally distributed data points

Data Visualization

Density Plot

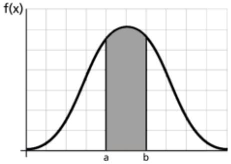
Density: Definition

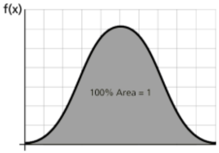
- Function that describes the **relative likelihood for this variable to take a given value**
- The probability of the random variable falling within a particular range of values is given by the integral of this variable's density over that range—that is, it is given by the **area under the density function** but above the horizontal axis and between the lowest and greatest values of the range. The probability density function is **non-negative** everywhere, and its **integral over the entire space is equal to one**.

$$p(x) \geq 0$$

$$p(a < x < b) = \int_a^b p(x)dx$$

$$\int_{-\infty}^{\infty} p(x)dx = 1$$



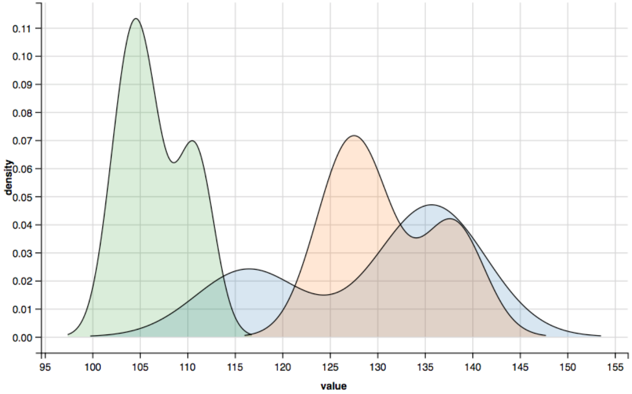


Density: Example

$$p(x) \geq 0$$

$$p(a < x < b) = \int_a^b p(x)dx$$

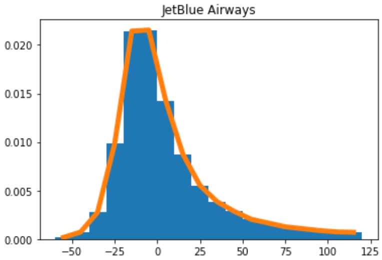
$$\int_{-\infty}^{\infty} p(x)dx = 1$$



Density in Python

`plt.hist()`

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

flights = pd.read_csv("http://www.eecs.qmul.ac.uk/~linwang/download/
ecs764/formatted_flights.csv")
JetBlue = flights[flights['name']=='JetBlue Airways']
delay = JetBlue['arr_delay’]

hist, bins, path = plt.hist(delay, bins=18, density=True)
bin_centers = (bins[1:]+bins[:-1])*0.5
plt.plot(bin_centers, hist, linewidth=5)
plt.title('JetBlue Airways')
```

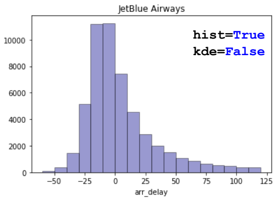


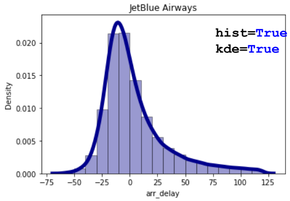
Density in Python

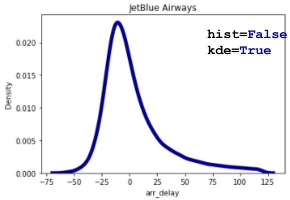
`seaborn.distplot()`

```
import seaborn as sns
sns.distplot(delay, hist=True, kde=True, bins=18,
             color = 'darkblue',
             hist_kws={'edgecolor':'black'},
             kde_kws={'linewidth': 5})
plt.title('JetBlue Airways')
```

kernal density estimation (kde)



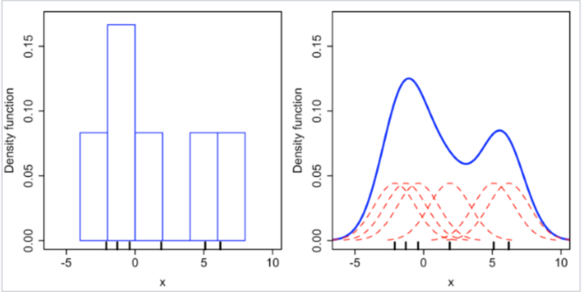




https://seaborn.pydata.org/generated/seaborn.distplot.html

Density: Kernal Density Estimation

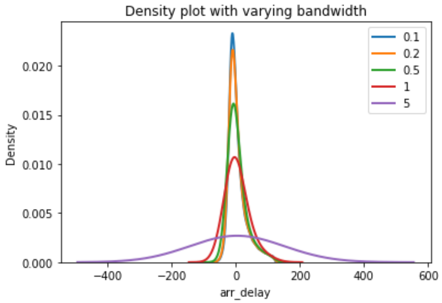
Sample	1	2	3	4	5	6
Value	-2.1	-1.3	-0.4	1.9	5.1	6.2



Comparison of the histogram (left) and kernel density estimate (right) constructed using the same data. The six individual kernels are the red dashed curves, the kernel density estimate the blue curves. The data points are the rug plot on the horizontal axis.

Density: Bandwidth Values

```
bw_list = [0.1, 0.2, 0.5, 1, 5]
for b in bw_list:
  sns.distplot(delay, bins = 18, hist = False, kde = True,
               kde_kws={'bw': b})
plt.legend(bw_list)
plt.title('Density plot with varying bandwidth')
```



Data Visualization

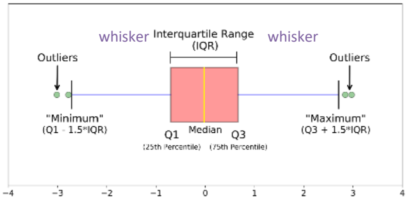
Box Plot

Box Plot: Definition

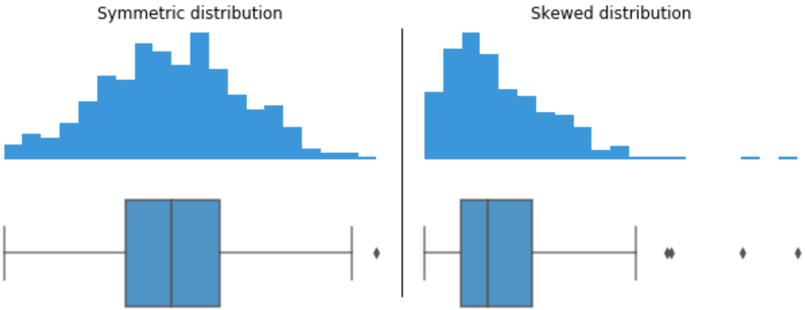
BOX PLOT: DEFINITION

- Graphically depicts **groups of numerical data through their quartiles/percentiles**.

- Box plots display five number summary: “minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”.
  - Lower whisker: Minimum - Q1
  - Upper whisker: Q3 - Maximum
  - Small/large values (**outliers**) may be plotted as individual points.



Box Plot



Box Plot in Python

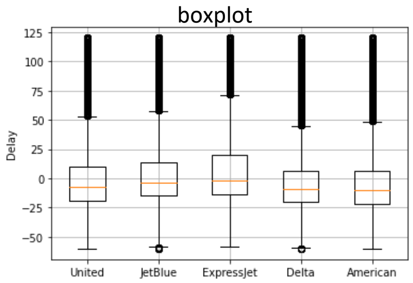
plt.boxplot()

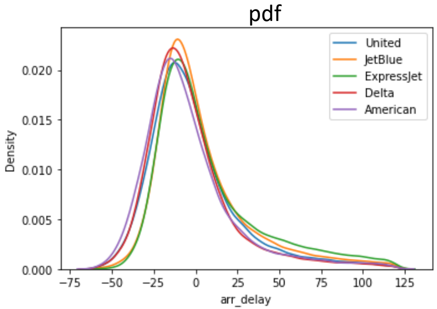
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
flights = pd.read_csv("http://www.eecs.qmul.ac.uk/~linwang/download/ecs764/formatted_flights.csv")

JetBlue = flights[flights['name']=='JetBlue Airways']['arr_delay']
Delta = flights[flights['name']=='Delta Air Lines Inc.']['arr_delay']
United = flights[flights['name']=='United Air Lines Inc.']['arr_delay']
American = flights[flights['name']=='American Airlines Inc.']['arr_delay']
ExpressJet = flights[flights['name']=='ExpressJet Airlines Inc.']['arr_delay']

delay = [United, JetBlue, ExpressJet, Delta, American]
plt.boxplot(delay)
```

https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.boxplot.html

Box Plot in Python

`plt.boxplot()`

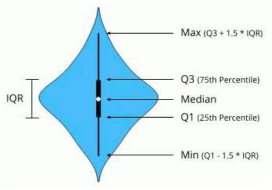


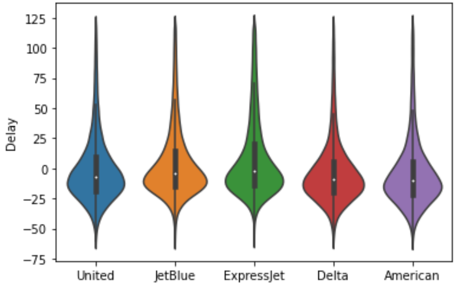


Box Plot Variant

seaborn.violinplot ()

```
import seaborn as sns
sns.violinplot(data=delay)
```





https://seaborn.pydata.org/generated/seaborn.violinplot.html

Box Plot: Discussion

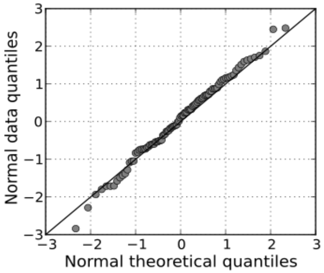
- Quick way of examining one or more sets of data visually (exploratory)
- More primitive than histogram or density
- No need for choice of bins and bandwidth
- Compact
- Histogram better for one group while boxplot better for multiple groups

Data Visualization

QQ Plot

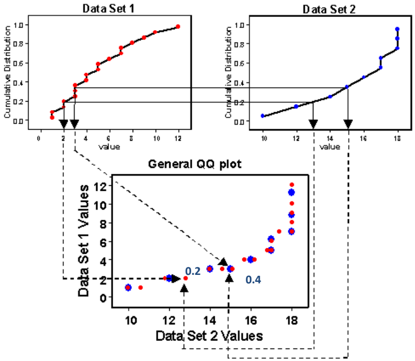
QQ Plot: Definition

- **Comparing** two probability **distributions** by plotting their **quantiles against each other**



QQ Plot: Definition

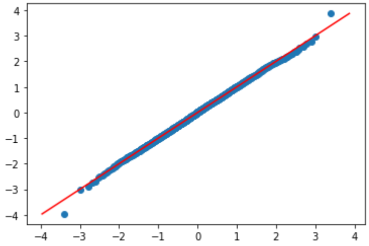
- Method:
  1) The set of intervals for the quantiles is chosen (typically k/(n + 1))
  2) A point (x, y) on the plot corresponds to one of the quantiles of the second distribution (y-coordinate) plotted against the same quantile of the first distribution (x-coordinate).



QQ Plot in Python

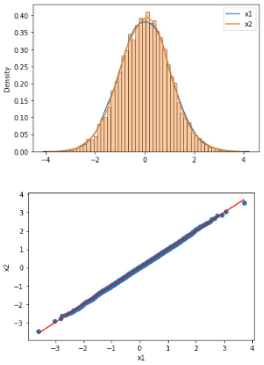
```
import matplotlib.pyplot as plt
import numpy as np

def standardize(data):
  z = (data - np.mean(data))/np.std(data)
  return z

def get_quantiles(data):
    quantiles = []
    for q in np.arange(0, 1.001, 0.001):
        quantiles.append(np.quantile(data, q))
    return quantiles

def pyqqplot(data1, data2):
    data1 = standardize(data1)
    data2 = standardize(data2)

    q1 = np.array(get_quantiles(data1))
    q2 = np.array(get_quantiles(data2))
    plt.scatter(q1, q2 )

    minim = min(data1.min(), data2.min())
    maxim = max(data1.max(), data2.max())
    plt.plot([minim, maxim], [minim, maxim], 'r-')
```

```
x1 = np.random.normal(0, 1, 10000 )
x2 = np.random.normal(0, 1, 10000 )
pyqqplot(x1, x2)
```

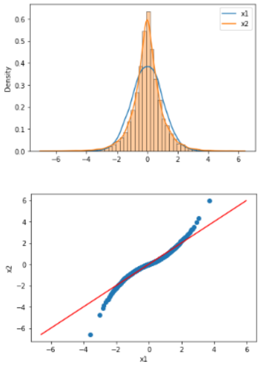


QQ Plot: Example

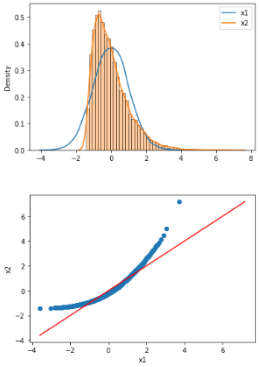
normally distributed data



data too peaked in middle



skewed data

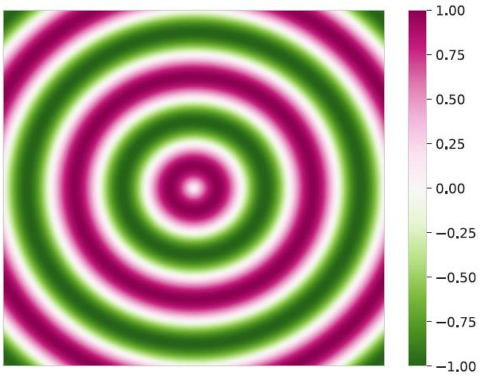


Data Visualization

Displaying Multivariate:
Heatmap, Surface, Contour, Quiver

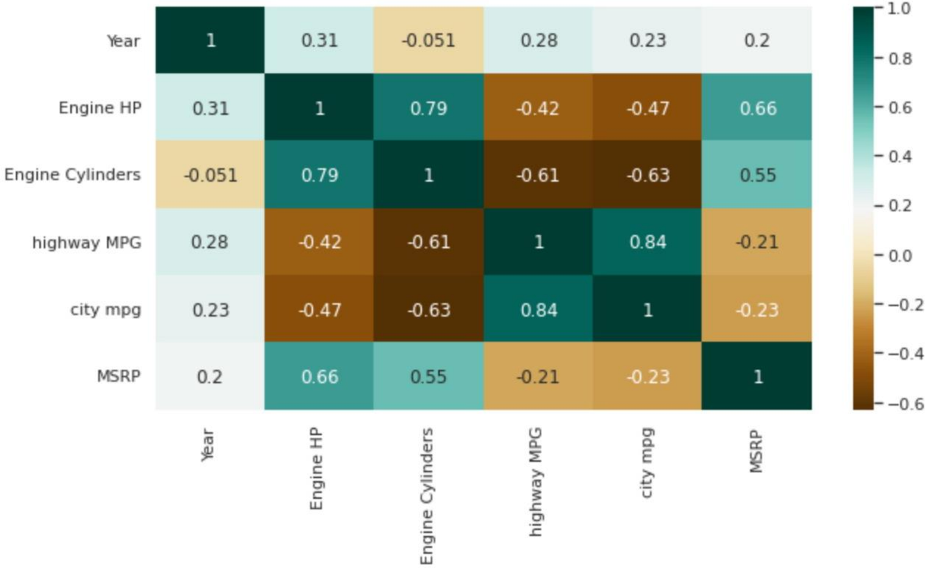
Heatmap

- A heat map uses a colour to represent the value $f(x, y)$ of each point (x, y) in a scalar field $f : \mathbb{R}^2 \to \mathbb{R}$



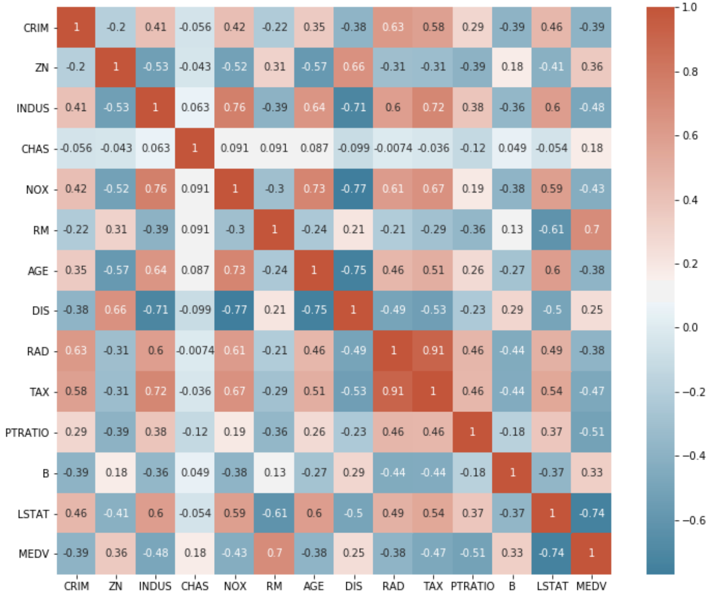
https://python-graph-gallery.com/heatmap/
https://matplotlib.org/stable/gallery/images_contours_and_fields/image_annotated_heatmap.html

Heatmap Example



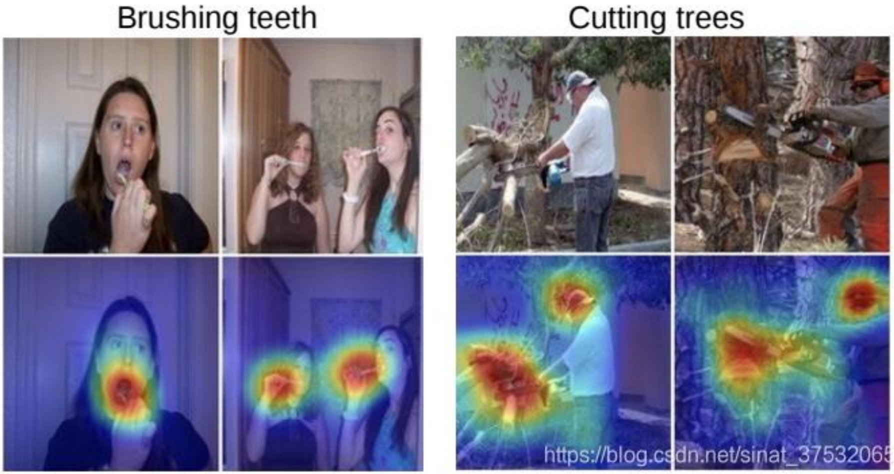
	Year	Engine HP	Engine Cylinders	highway MPG	city mpg	MSRP
Year	1	0.31	-0.051	0.28	0.23	0.2
Engine HP	0.31	1	0.79	-0.42	-0.47	0.66
Engine Cylinders	-0.051	0.79	1	-0.61	-0.63	0.55
highway MPG	0.28	-0.42	-0.61	1	0.84	-0.21
city mpg	0.23	-0.47	-0.63	0.84	1	-0.23
MSRP	0.2	0.66	0.55	-0.21	-0.23	1

Heatmap Example

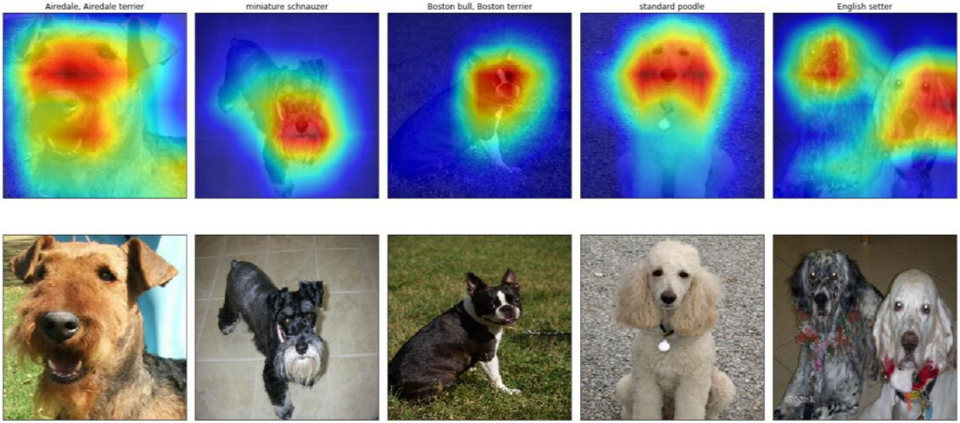


	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	1	-0.2	0.41	-0.056	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39
ZN	-0.2	1	-0.53	-0.043	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36
INDUS	0.41	-0.53	1	0.063	0.76	-0.39	0.64	-0.71	0.6	0.72	0.38	-0.36	0.6	-0.48
CHAS	-0.056	-0.043	0.063	1	0.091	0.091	0.087	-0.099	-0.0074	-0.036	-0.12	0.049	-0.054	0.18
NOX	0.42	-0.52	0.76	0.091	1	-0.3	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43
RM	-0.22	0.31	-0.39	0.091	-0.3	1	-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.7
AGE	0.35	-0.57	0.64	0.087	0.73	-0.24	1	-0.75	0.46	0.51	0.26	-0.27	0.6	-0.38
DIS	-0.38	0.66	-0.71	-0.099	-0.77	0.21	-0.75	1	-0.49	-0.53	-0.23	0.29	-0.5	0.25
RAD	0.63	-0.31	0.6	-0.0074	0.61	-0.21	0.46	-0.49	1	0.91	0.46	-0.44	0.49	-0.38
TAX	0.58	-0.31	0.72	-0.036	0.67	-0.29	0.51	-0.53	0.91	1	0.46	-0.44	0.54	-0.47
PTRATIO	0.29	-0.39	0.38	-0.12	0.19	-0.36	0.26	-0.23	0.46	0.46	1	-0.18	0.37	-0.51
B	-0.39	0.18	-0.36	0.049	-0.38	0.13	-0.27	0.29	-0.44	-0.44	-0.18	1	-0.37	0.33
LSTAT	0.46	-0.41	0.6	-0.054	0.59	-0.61	0.6	-0.5	0.49	0.54	0.37	-0.37	1	-0.74
MEDV	-0.39	0.36	-0.48	0.18	-0.43	0.7	-0.38	0.25	-0.38	-0.47	-0.51	0.33	-0.74	1

Heatmap Example

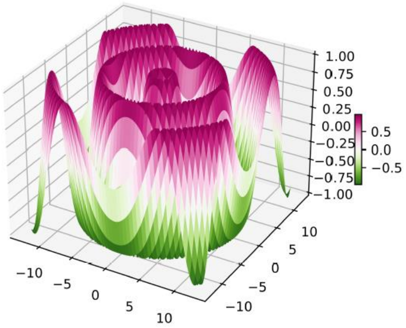


Heatmap Example



Surface Plot

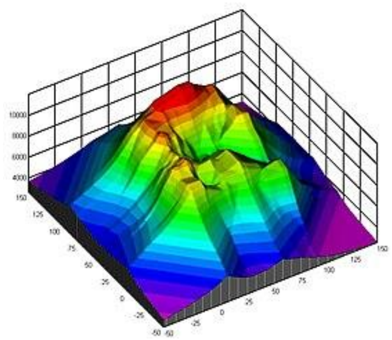
- A surface plot displace the graph of a bivariate function
- A surface is plotted to fit a set of data (X, Y, Z), where Z if obtained by the function to be plotted Z = f(X, Y).
- Optionally, the plotted values can be color-coded.

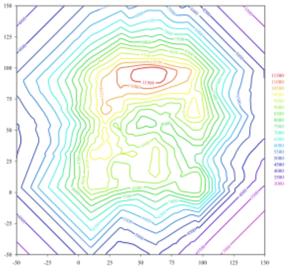


https://matplotlib.org/stable/gallery/mplot3d/surface3d.html

Contour Plot

- A contour plot represents a 3-D surface by plotting lines that connect points with common z-values along a slice.
- For example, you can use a contour plot to visualize the height of a surface in two dimensions.

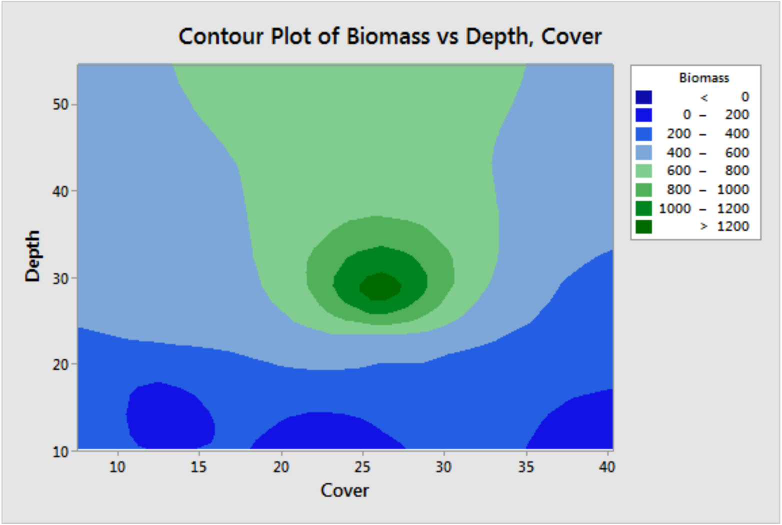




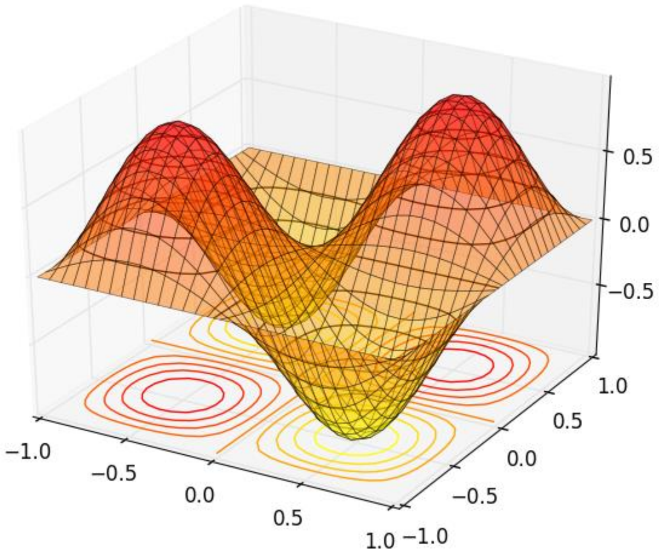
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.contour.html

Contour Plot Example

A biologist studies the effect of stream depth and canopy cover on fish biomass.

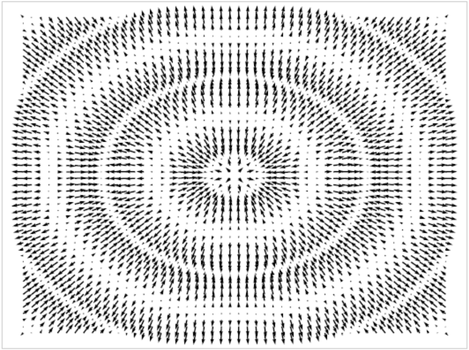


Contour and Surface Plot



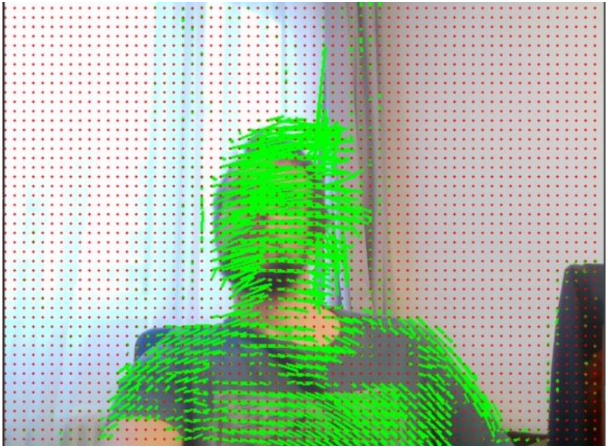
Vector Field (Quiver)

- In some data, a characteristic may have both magnitude and a direction associated with it (e.g. flow of a substance)
- Vector filed plots (or quiver plots) display both direction and magnitude
- Vector fields can model velocity, magnetic force, fluid motion, and gradients.



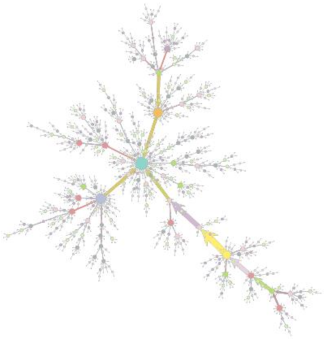
Vector Field

Optical flow in computer vision



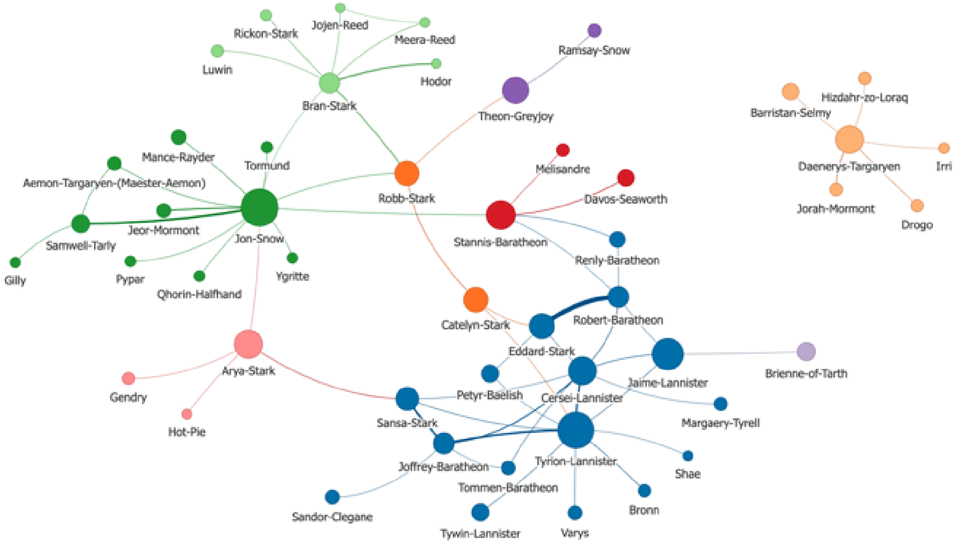
GRAPH VISUALIZATION

- Graph drawing is concerned with obtaining visual representations of graphs based on nodes and links
- There are many different graph layout algorithms that attempt to maximise different **quality measures**



https://plotly.com/python/network-graphs/

GRAPH VISUALIZATION

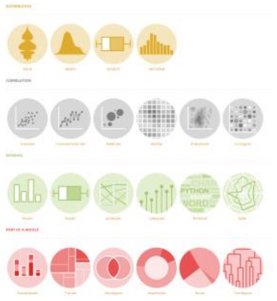


Game of Thrones characters with more than 60 interactions

MORE ON PYTHON GRAPHS

- https://python-graph-gallery.com/

THE PYTHON
GRAPH GALLERY



Data Visualization

Visualization principles

Data Visualization Tools Covered

- **Scatter plot:** exploring data
- **Time series:** time
- **Bar chart:** comparisons
- **Histogram:** binned frequency
- **Density:** continuous functions
- **Box plot:** quantiles
- **QQ plot:** comparing distributions
- **Others**: displaying multivariate

Data Visualization

- Data visualisation is an art
- Many plot types for different purposes: knowing the right tools saves a lot of time
- Important tool for data exploration, especially extracting basic relationships

Check list: Accent Principle

- Apprehension: does the visualisation maximize the understanding of the relationships in the data?
- Clarity: are the most important relationships the most visually prominent?
- Consistency: are the visual elements consistent with their use in previous visualisations?
- Efficiency: are the visual elements economically used and easy to interpret?
- Necessity: is the visualisation more useful than alternative ways to represent the data?
- Truthfulness: are the visual elements accurate and unambiguous?

Questions?

also please use the forum on QM+