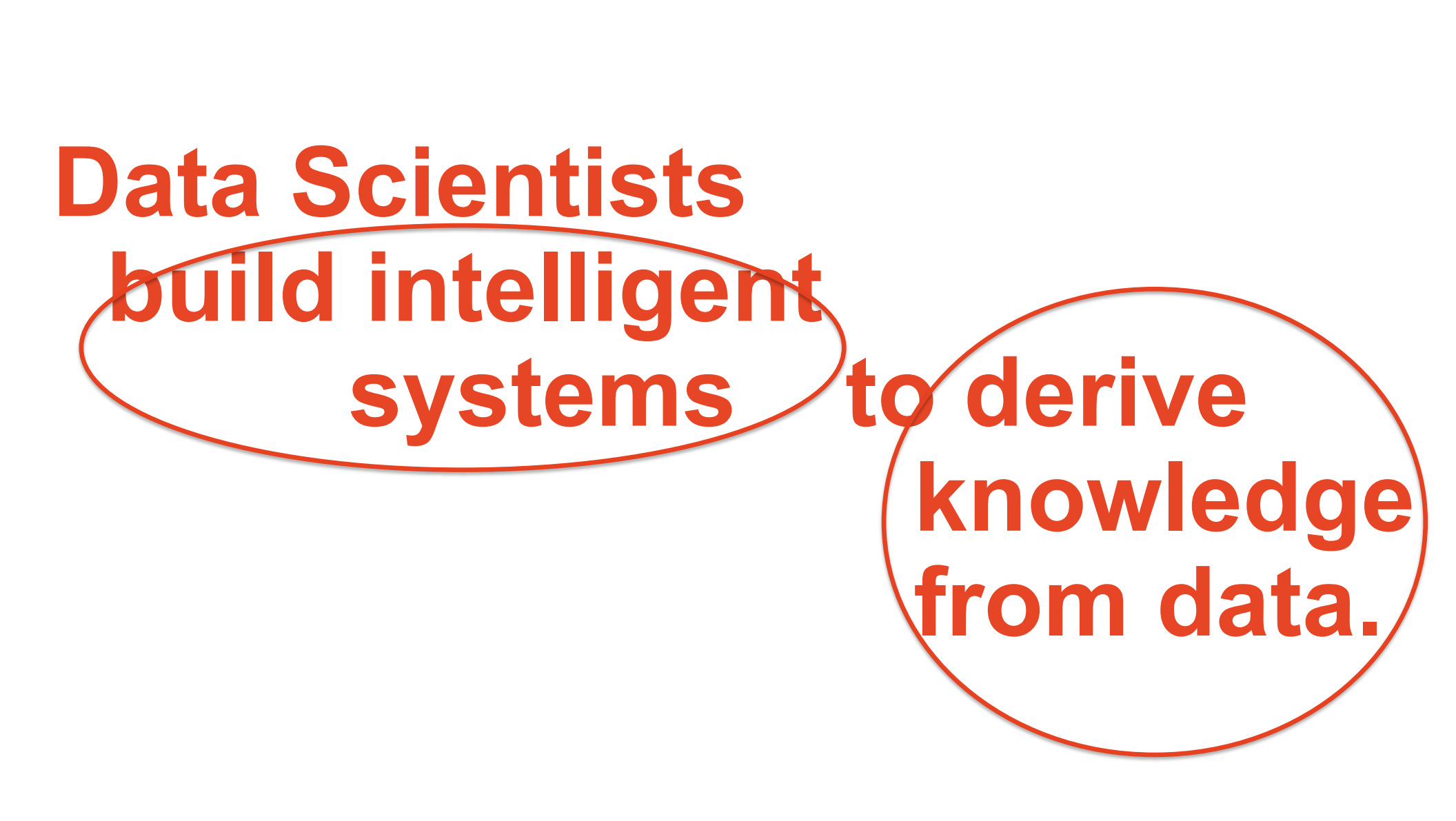
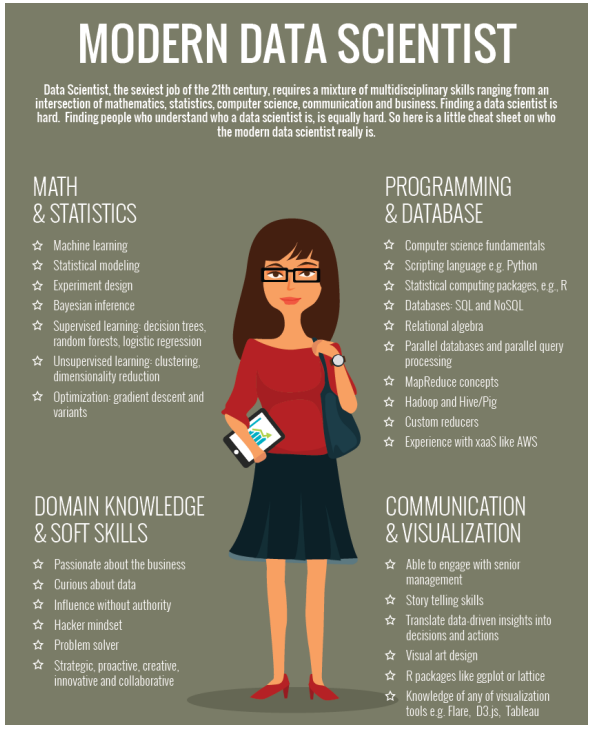
COMP5310 - Week 1

WHAT IS DATA SCIENCE?


Data Scientists
build intelligent
systems
to derive
knowledge
from data.

Data Science Skills



http://www.marketingdistillery.com/2014/11/29/is-data-science-a-buzzword-modern-data-scientist-defined/

Data scientists help organisations:

- understand their data,
- ask meaningful questions,
- derive transformative insights,
- lead empirically grounded decision making.

Example: Reducing Costs by Route Optimisation

http://www.bloomberg.com/news/articles/2013-10-30/ups-uses-big-data-to-make-routes-more-efficient-save-gas

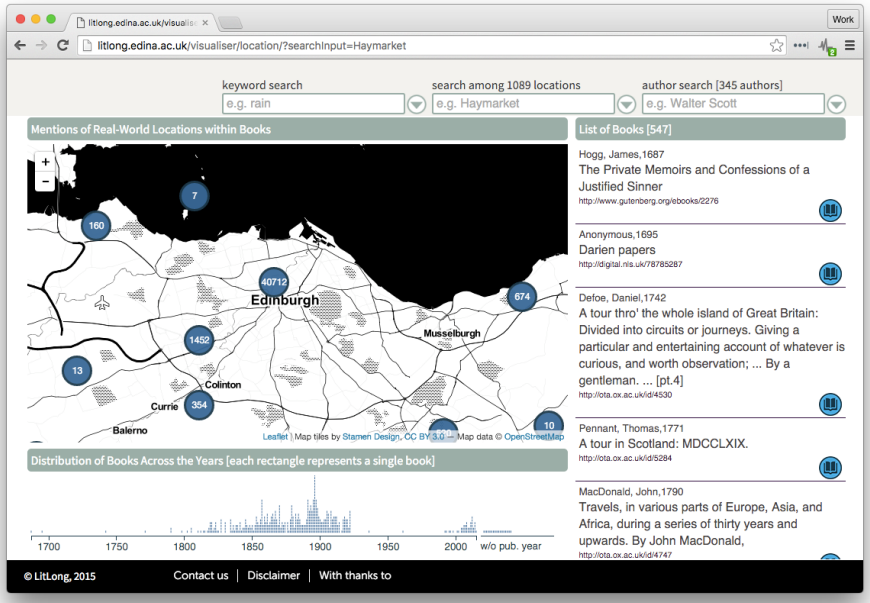
- Use customer, vehicle and delivery data
- 1 mile less per day for every driver saves $50 million p.a. in fuel, maintenance and time
- Less idling, e.g., by avoiding left turns, saved 6 million liters of fuel in 2012

Example: Urban&Transport Planning, Public Health

http://www.walkscore.com/research/

- Integration of data about road and public transport network with data about population, services, restaurants, amenities etc.
- Summarising *Walkability Score* overlayed on map visualisation
- Prediction of impact of new developments
- API for use in 3rd party apps, eg. supporting real estate agents

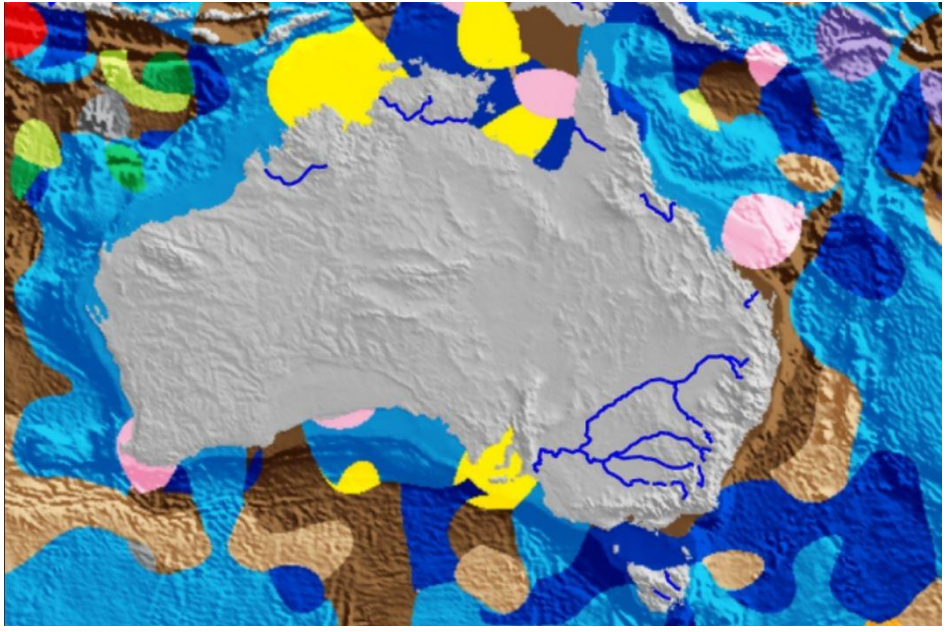
Example: Mapping Literary References



http://litlong.org/

- Identify and resolve location mentions in literature
- Overlay references on map visualisation
- Keyword, location and author search

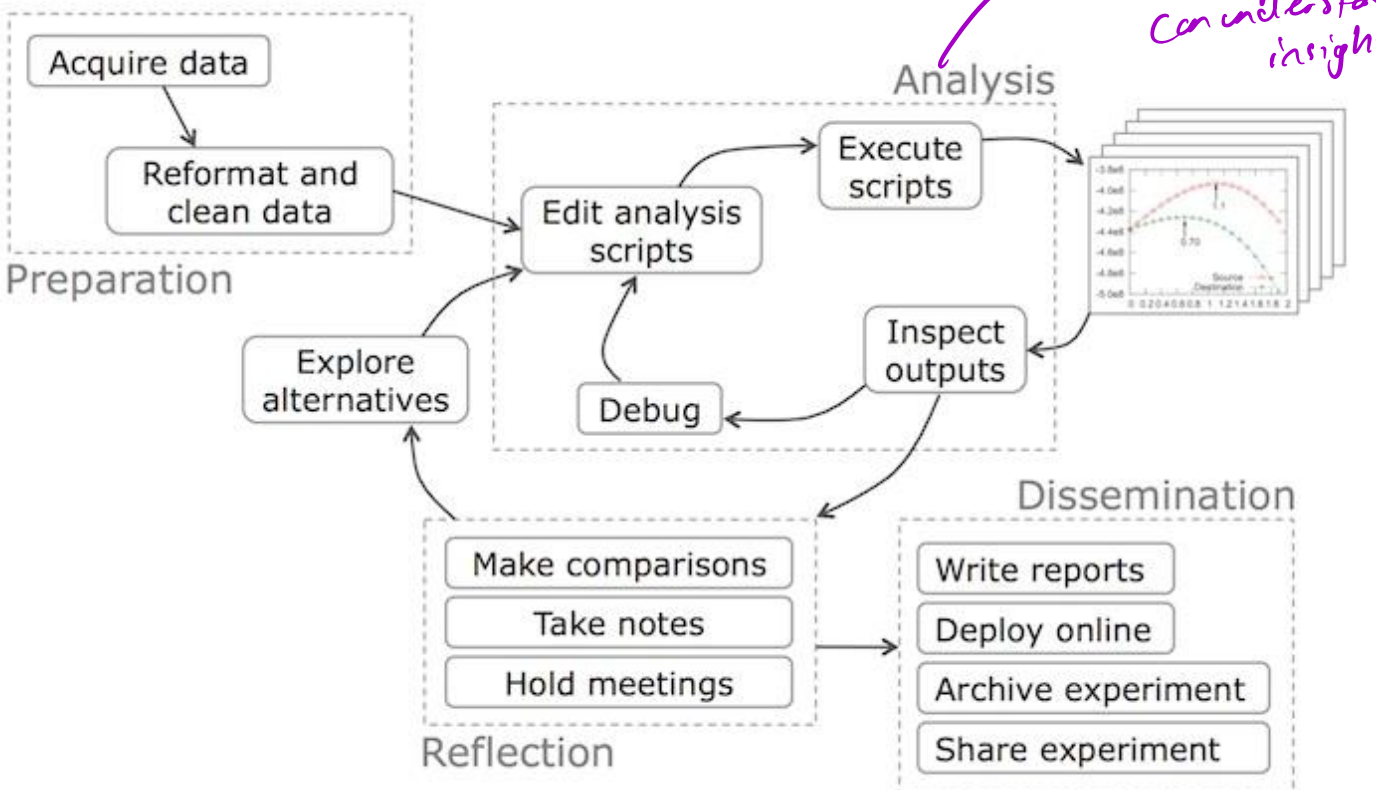
Example: Mapping Seafloor Geology with SVM



http://portal.gplates.org/#SEAFLOOR

- Use descriptions from 14,500 samples collected from 1950-present
- Predict sediment in unobserved regions using support vector machine

Data Science Workflow



[Source: https://cacm.acm.org/blogs/blog-cacm/169199-data-science-workflow-overview-and-challenges/]


THE UNIVERSITY OF
SYDNEY

quantitative, categorical, unstructured...

Types of Data and Levels of Measurement

Level of Measurements and Type of Data

- **Categorical**
  - Nominal
    - Dichotomous
  - Ordinal
- **Quantitative**
  - Interval
  - Ratio

Other types of data:

- **Text**
- **Images, Video**

} → NLP

Categorical Data

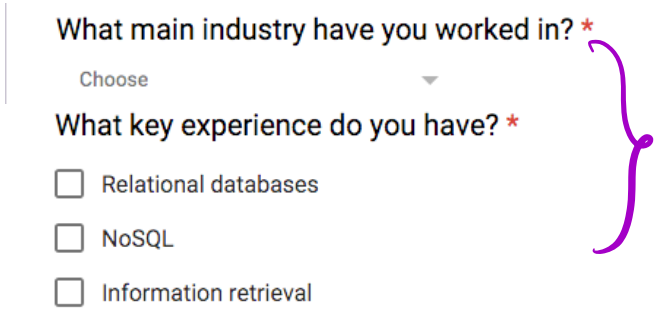
- A categorical variable is also known as a discrete or qualitative variable and can have two or more categories.

categories which are not ordered, usually represented by names

- It is further divided into two variants, **nominal** and **ordinal**.
  - These variables are sometimes coded as numerical values, or as strings.

Nominal Data

- This is an unordered category data. This type of variable may be “label-coded” in numeric form but these numerical values have no mathematical interpretation and are just labeling to denote categories.
- For example, colours: black, red and white can be coded as 1, 2 and 3.



- Values are names
- No ordering is implied
- Eg jersey numbers

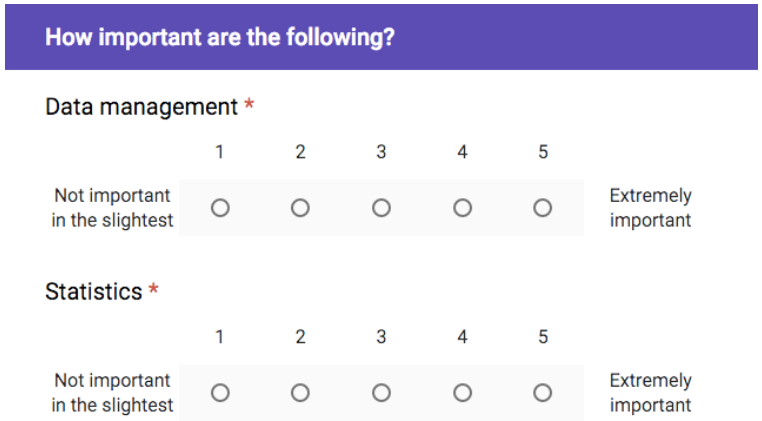
Dichotomous Data

- A dichotomous is a type of nominal data that can only have two possible values, e.g. true or false, or presence or absence. These are also sometimes referred as binary or Boolean variables.

  - True (1) or false (0)
  - Correct / Incorrect
  - Student / Academic

Ordinal Data

- This is ordered categorical data in which there is strict order for comparing the values, so a labelling as numbers is not completely arbitrary.
- For example, human height (small, medium and high) can be coded into numbers small = 1, medium = 2, high = 3.



- Values are ordered
- No distance is implied
- Eg rank, agreement

Interval Data

- It is a variable in which the interval between values has meaning and there is no true zero value.

Height as well!

- Values encode differences
- Equal intervals between values
- No true zero
- Addition is defined
- e.g Celsius temperature scale
  - can't express "no temperature"
- e.g. "What year were you born?"

Ratio Data

- It is variable that might have a true value of zero and represents the total absence of the variable being measured.
- For example, it makes sense to say a Kelvin temperature of 100 is twice as hot as a Kelvin temperature of 50 because it represents twice as much the thermal energy (unlike Fahrenheit temperatures of 100 and 50).

How long have you been in your current job?
2 years
10 years
8 years
4 years
35 years

- Values encode differences
- Zero is defined
- Multiplication defined
- Ratio is meaningful
- Eg length, weight, pressure, income

Levels of Measurement

	Nominal	Ordinal	Interval	Ratio
Countable	✓	✓	✓	✓
Order defined		✓	✓	✓
Difference defined (addition, subtraction)			✓	✓
Zero defined (multiplication, division)				✓

Measures of Central Tendency

	Nominal	Ordinal	Interval	Ratio
Mode	✓	✓	✓	✓
Median		✓	✓	✓
Mean			✓	✓

no mode, median or mean on categorical data.

Measures of Dispersion

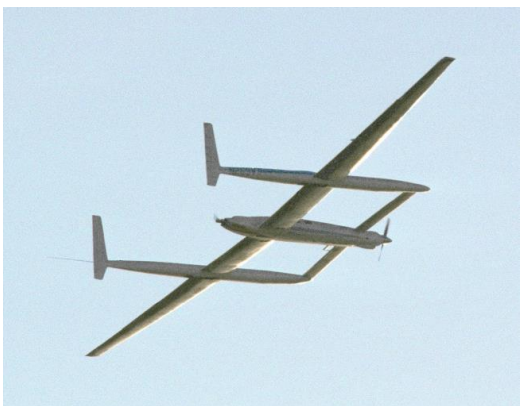
	Nominal	Ordinal	Interval	Ratio
Counts / Distribution	✓	✓	✓	✓
Minimum, Maximum		✓	✓	✓
Range		✓	✓	✓
Percentiles		✓	✓	✓
Standard deviation, Variance			✓	✓

Borderline Cases

- Mean values for rankings…
- Time without dates and timezones
  - Is 07:00 before or after 21:00?
  - What time difference between these two points in time?
  - What about a flight leaving 21:00 and arriving 07:00?

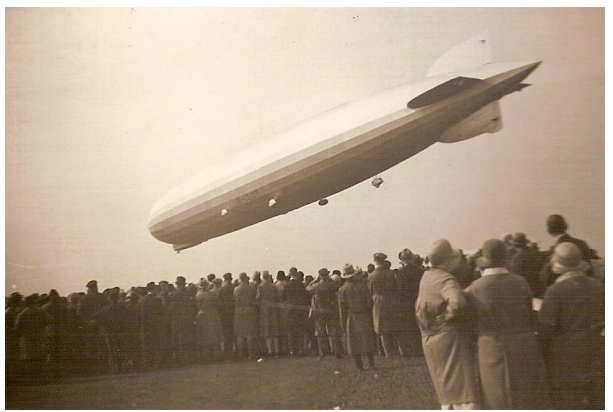
Airbus A380

Source: Wikipedia, CC BY-SA 3.0



Rutan Model 76 Voyager

Source: NASA - http://www.dfrc.nasa.gov/Gallery/Photo/Voyager/HTML/EC87-0029-02.html

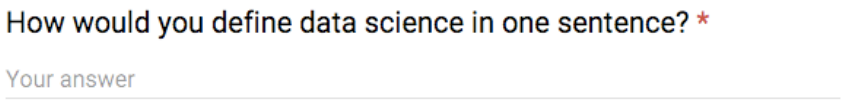


Zeppelin LZ 127

Source: Wikipedia, CC BY-SA 3.0

What about Text Data?

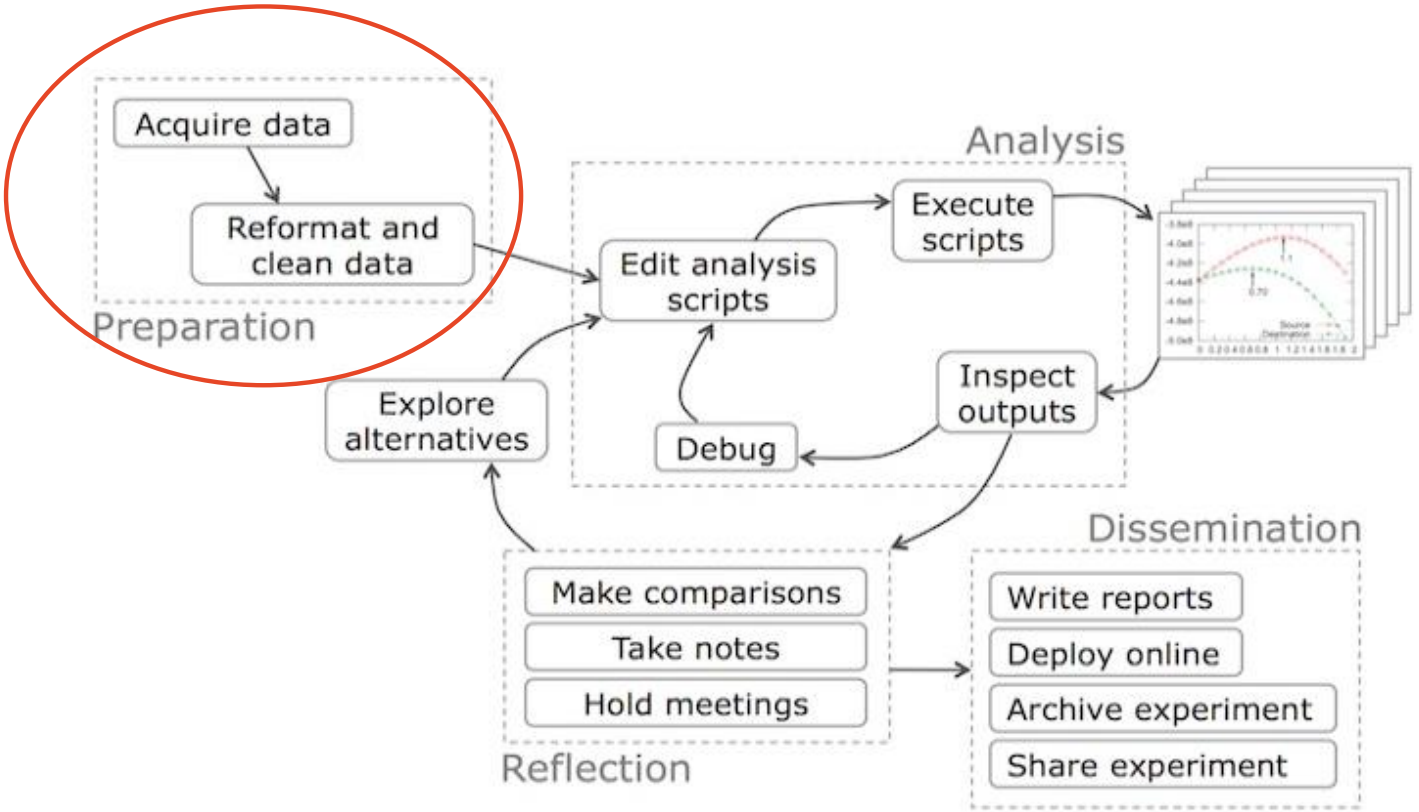
- Not defined as traditional data type in statistics
- Requires interpretation, coding or conversion
- More later in this course…




THE UNIVERSITY OF
SYDNEY

Data Acquisition and Data Cleaning

Exploratory Data Analysis Workflow



Data Acquisition – Where does Data come from?

- File Access
  - You or your organisation might already have a data set, or a colleagues provides you access to data.
  - Or: Web Download from an online data server (e.g. data.gov.au)
  - Typical exchange formats: CSV, Excel, sometimes also XML
- Programmatically
  - Scrapping the web (HTML)
  - or using APIs of Web Services (XML/JSON)
- Database Access
- Collect data yourself, eg. via a survey

or even under the cloud (Azure, AWS)

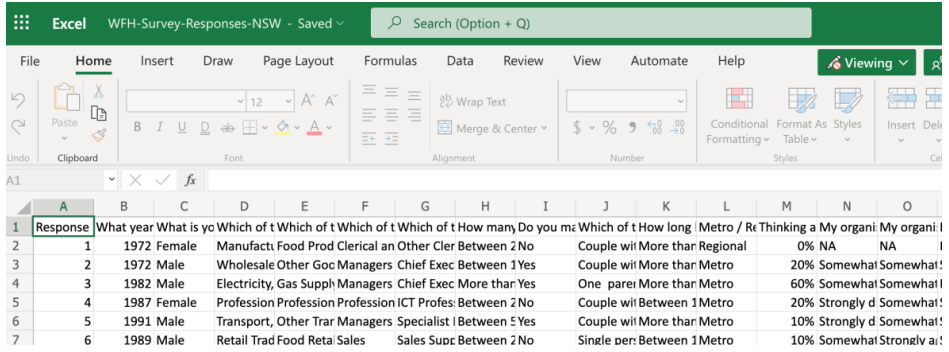
Cleaning and Transforming Data

- Real data is often '*dirty*' (Metadata)
- Important to do some data cleaning and transforming first
- Typical steps involved:
  - type and name conversion
  - filtering of missing or inconsistent data
  - unifying semantic data representations
  - matching of entries from different sources
- Later also:
  - **Scaling** and optional dimensionality reduction

Data quality and naming conventions based on business ventures

Approach 1: Spreadsheet Software

- In this first week, we will use some example data from data.gov.au
  - Available as simple CSV (comma-separated values) format
- Open, e.g., in Google spreadsheet
  - https://docs.google.com/spreadsheets
  - File > Import - Click on Upload
- Manually Inspection
  - Missing values? Placeholders?
  - Any spelling mistakes or inconsistent data?
  - Any problems with columns in spreadsheet?
  - Text vs. numeric values
  - …



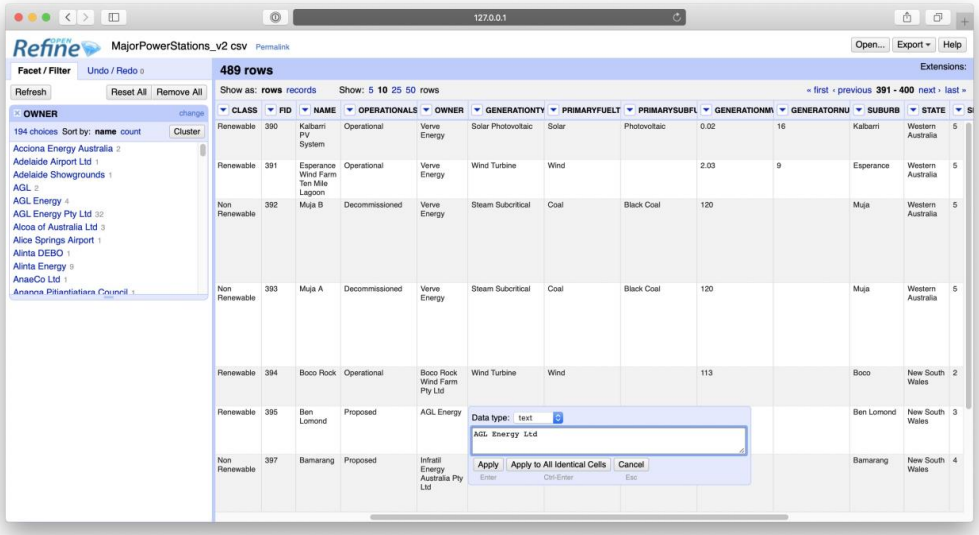
Limitation of being too manual and prone to mistakes...

Approach 2: Specific Data Cleaning Tools

OpenRefine
A free, open source, powerful tool for working with messy data

- Open Source Example: **Open Refine**
  - Originally developed by Google, but also other commercial tools available
  - Allows to visually inspect and clean data with interactive user-interface
  - More advanced: Reconcile and match different data sets
  - Export to CSV, Excel, HTML, …
- Very helpful,
  especially for smaller data sets
  - But manual interaction required

much more repeatable than spreadsheets - certain actions are more repeatable.



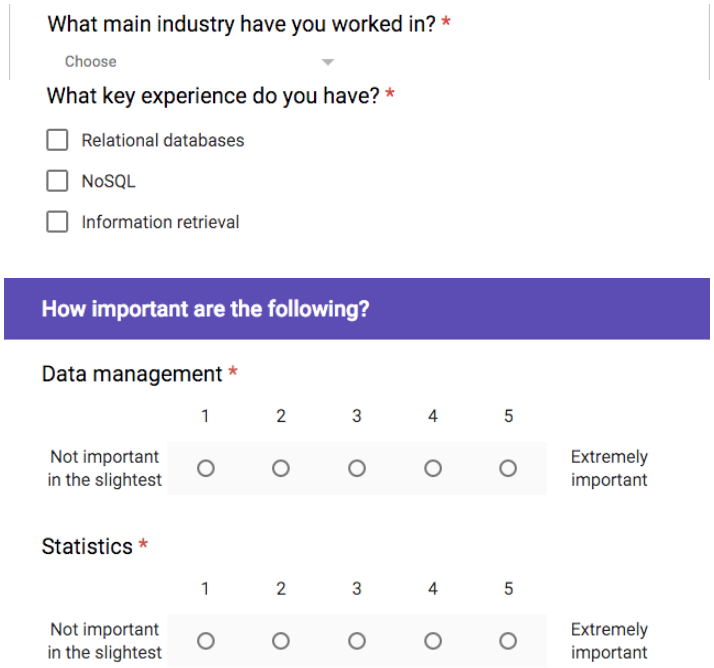

THE UNIVERSITY OF
SYDNEY

Exploratory Data Analysis (with Spreadsheets)

Use Case: Survey Data

- Data from a user survey
- Answers collected in spreadsheet
- Representing various types of data
  - Nominal data
  - Ordinal data
  - Ratio data
  - …

- Goal:
  Explorative Survey Data Analysis



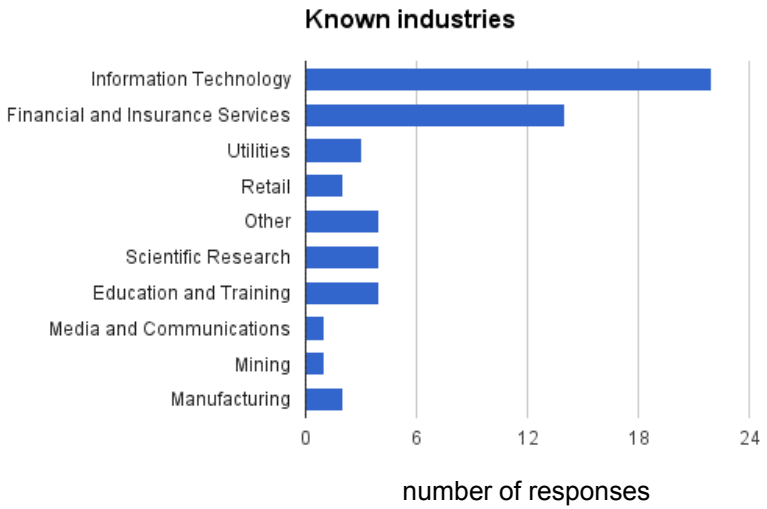
Summarising Nominal Data

Survey Example:

- *What industries have participants worked in?*
- *What industries would participants like to go into?*

Which of the following describes your industry best?
Manufacturing
Wholesale Trade
Electricity, Gas, Water and Waste Services
Professional, Scientific and Technical Services
Transport, Postal and Warehousing

Summarising Nominal Data with Bar Charts



Measures of central tendency:

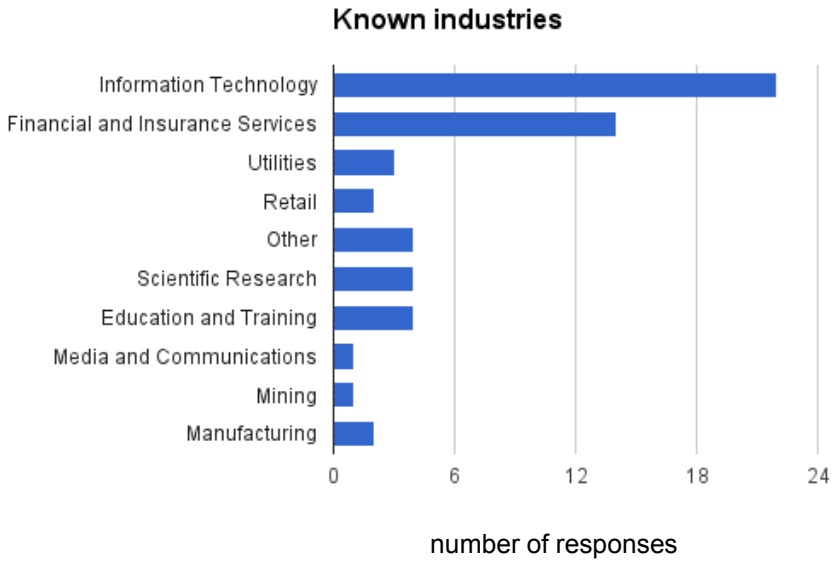
- mode
  - The most frequent value
  - Defined for nominal data, but spreadsheets might not compute
  - Can read from a bar chart

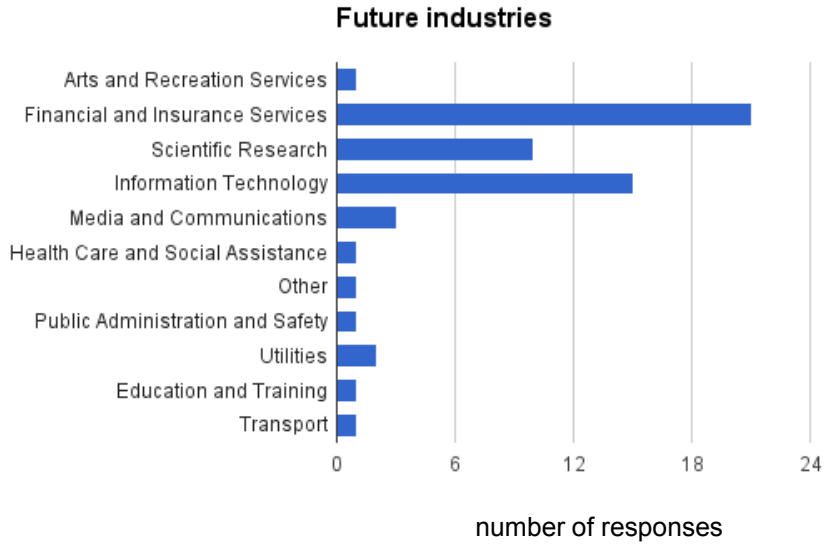
Measures of dispersion:

- counts / distribution
  - of count frequency of each category

Visualisation: Bar Chart

Chart Comparing Known and Future Industries





Summarising Ordinal Data

Likert scale.

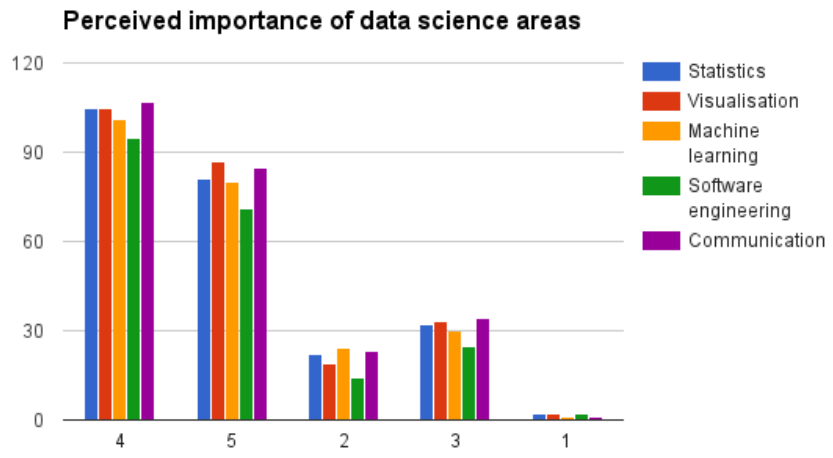
Survey Example:

- *What areas of data science are considered important?*

How Important do you see the following areas of Data Science?					
Data Management	Statistics	Visualisation	ML & Data Mining	Software Engineering	Communication
3	4	5	5	2	5
5	5	5	5	4	5
4	4	3	5	4	4
5	4	4	5	4	3
5	3	5	3	2	4
4	5	5	5	3	5
5	4	5	3	3	5
5	3	4	3	2	4
4	4	5	4	4	4

Histogram ↔ ordinal data.

Summarising Ordinal Data: Histograms, median, percentiles



- note- something is wrong with above's chart – can you see what?

1. x-axis is not ordered, so difficult to interpret
2. The total sum of ranks is shown which is not meaningful for analysis.

Measures of central tendency:

- median, mode

Measures of dispersion:

- counts/distribution
- min/max/range
- percentiles

Visualisation: Histogram Chart

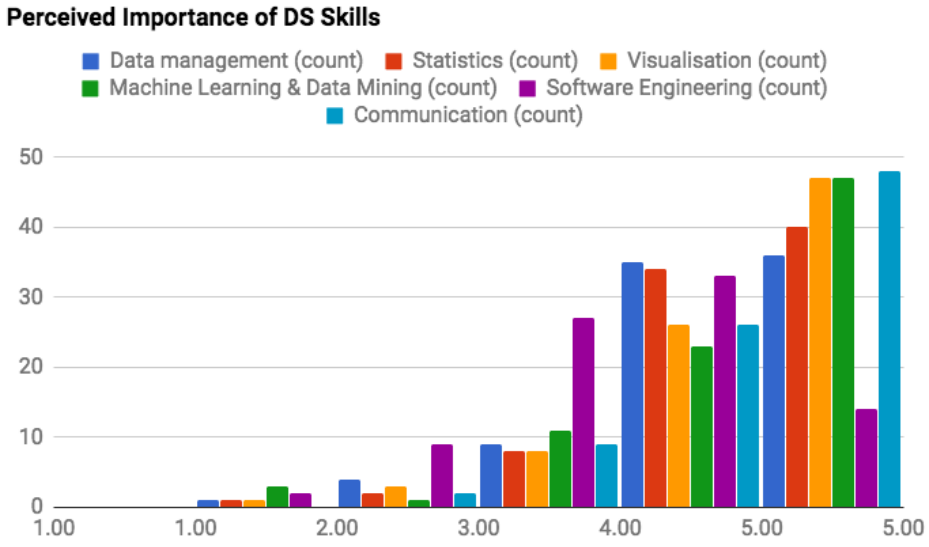
Calculating Descriptive Statistics

- First sort values, then:
  - ***Median*** is the middle value (or average of two middle values)
  - ***Minimum*** is the first value
  - ***Maximum*** is the last value
  - ***10th percentile*** is item at index 0.1*N
  - ***90th percentile*** is item at index 0.9*N
  - ***Range*** is Maximum minus Minimum

Creating a Histogram Chart

- Count frequency, e.g., of ordinal values within each category
- Display on histogram chart with one variable grouped inside
- In Google Sheets
  - Simply select data range, click “insert chart” icon and select “Histogram Chart”
- In Excel
  - Needs a column of responses and a column of counts
    - Either via an aggregation table with e.g. an array formula using COUNTIF(…)
    - or via a Pivot Table which allows to select one column as the value to be aggregated via COUNT
  - Then: Insert > Column Chart

Histogram comparing Areas of Data Science



better color scheme would also improve visual hints...

Good:

- Illustrates tendency
- Areas differentiated

Bad:

- buckets on x-axis not clear and no clear separation
- no axis titles (add manually)

some of those issues obviously depend on the used SW…

Summarising Ratio Data

Survey Example:

- *How do years of professional/programming experience compare?*

How many years professional experience do you have?	How many years programming experience do you have?
2	0
2	6
20	18
20	0
6	0
13	13
6	3
3	1
5	3

scatterplot ⟷ rational interval data.

Summarising Ratio (and Interval) Data

Measures of central tendency:

- mean, median, mode

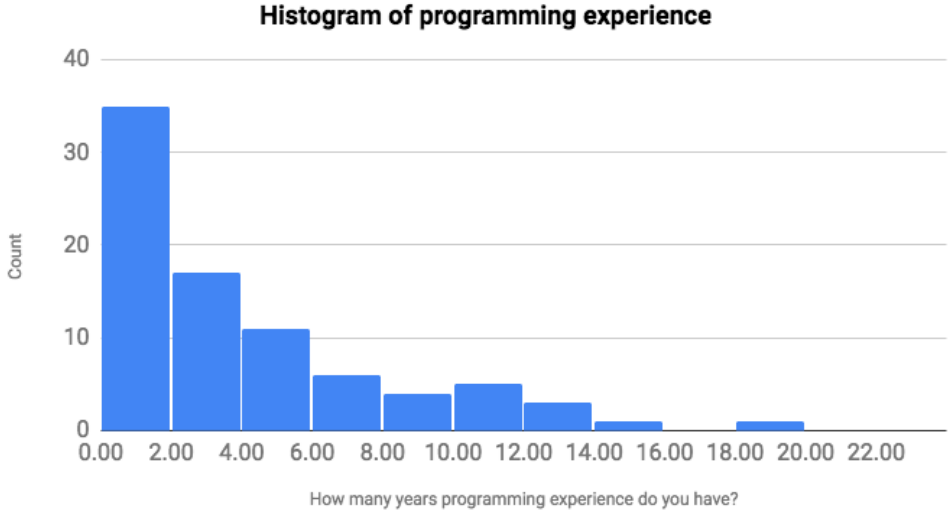
Measures of dispersion:

- counts/distribution
- min/max/range
- percentiles
- stdev/variance

Visualisation: Scatter Plot

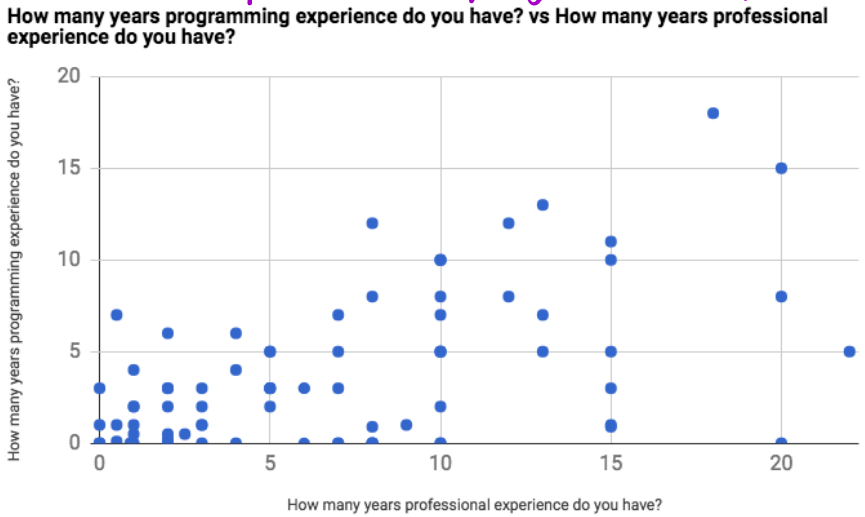
Binned Histograms for Experience

Can also use ratio and interval data on histograms

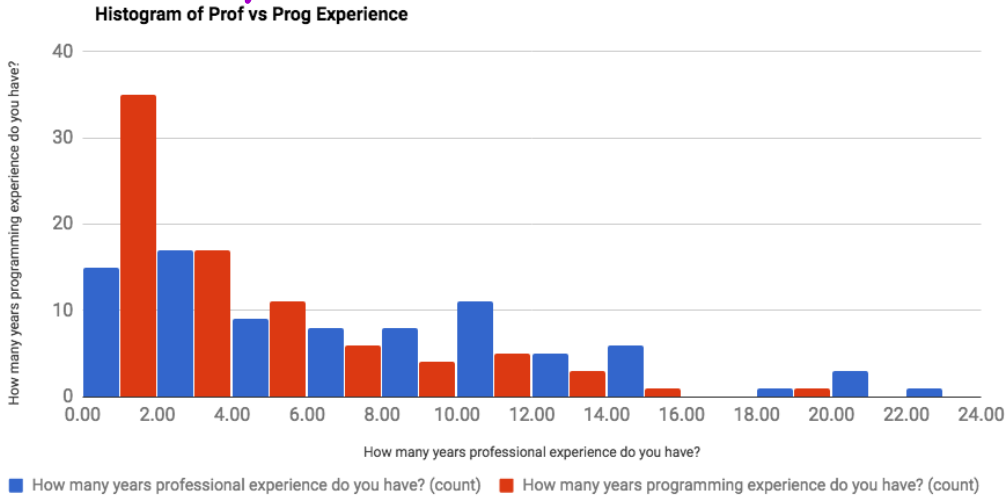


Comparison with Scatterplot and Histogram Overlays

Correlation of professional experience vs programming experience



Value distribution of professional experience vs programming experience.



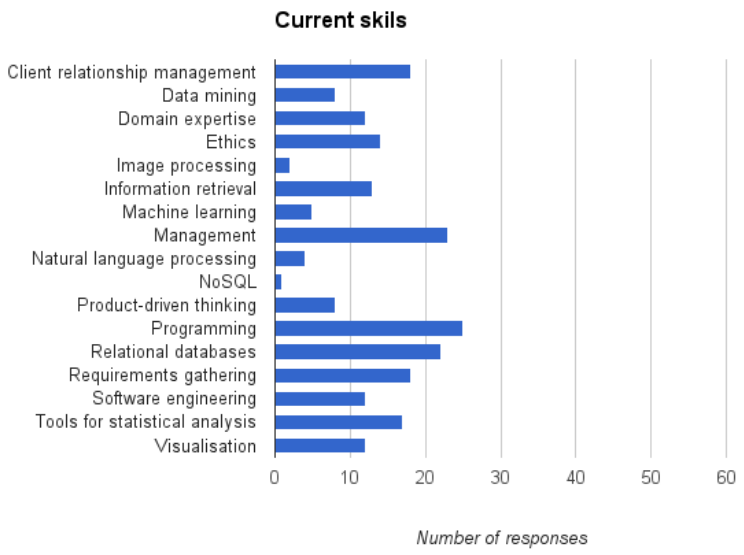
Complex Counting - How create a Histogram of skills?

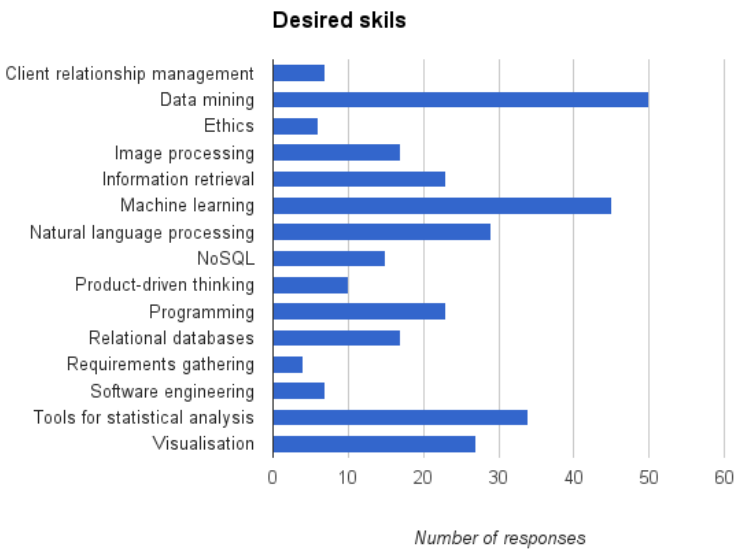
- Some datasets include columns with list of values
- For example, a survey mught have multiple values in cells of the skills column:
  “Software engineering, Requirements gathering, Product-driven thinking”
- Need to split possible values:
  `=sort(unique(transpose(split(join(“;”, Data!N2:N86, “;”, FALSE))))`
- Then count:
  `=countif(Data!N2:N86, concat(concat(“*”, A2), “*”))`

do we need to know how to do this on spreadsheet?

- Could use similar to get word counts
- Better to use programming language (clarity, reusability, etc)

Histograms of current and desired skills





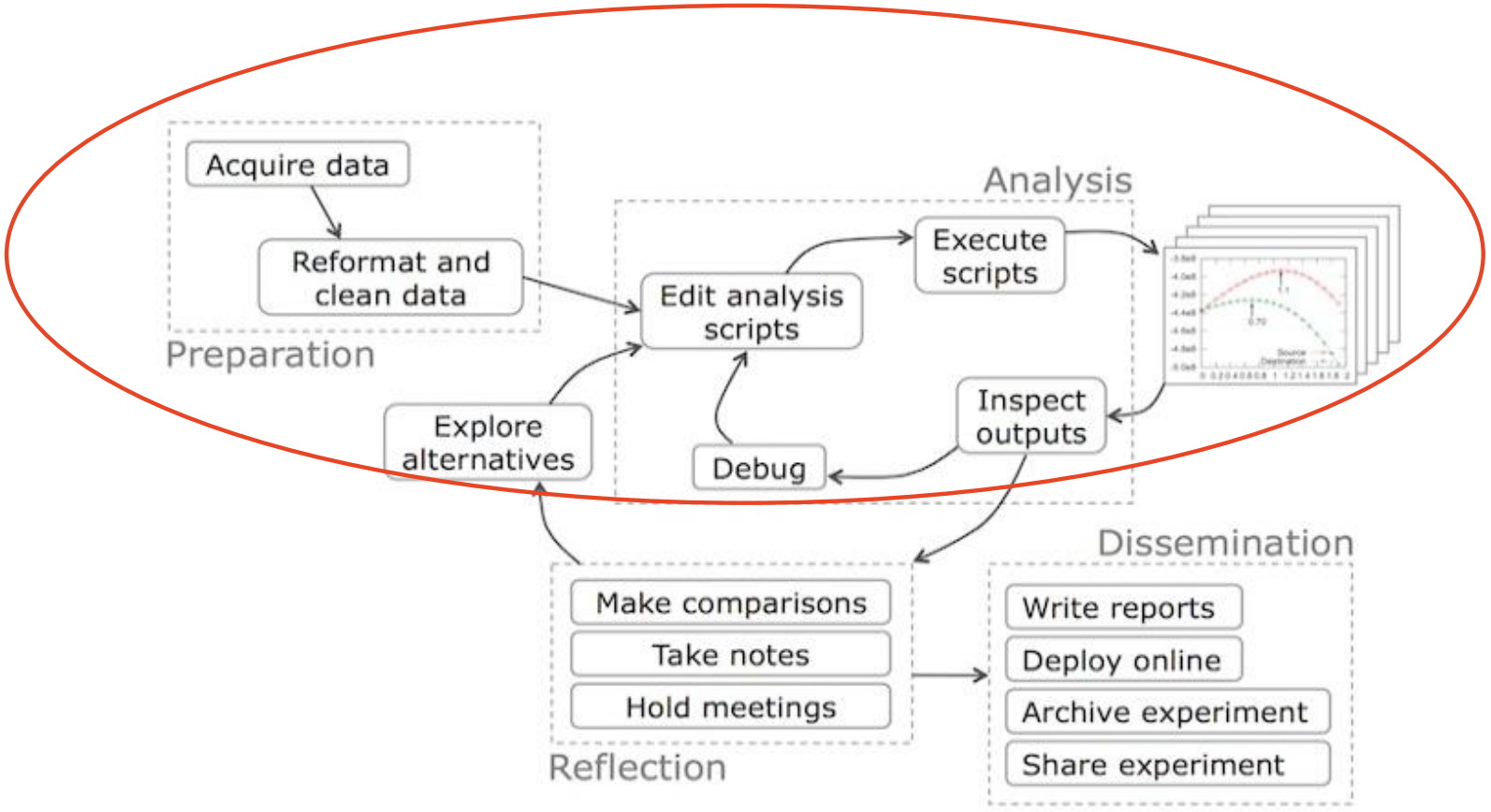
Tips and Tricks

- Data cleaning important for any meaningful analysis
- Spreadsheet software is good for quick interactive analysis
  Need programmatic analysis for bigger/complex data
  - Careful about which types of data allow what kind of measures & viz.
- Measures of central tendancy (e.g., mean) are not sufficient
  Always explore and communicate spread as well (e.g., stdev)
- Good visualisations help convey distributions and relationships
  - Label all plots and diagrams with readable and visible fonts
  - Use same axis bounds when comparing plots
  - Use meaningful axis bounds to convey effect size
    (50-55 on a 100 point scale over-sells small differences)
  - Design so comparison/effect is clear, include description of axes


THE UNIVERSITY OF
SYDNEY

Exploratory Data Analysis with Python

Exploratory Analysis Workflow

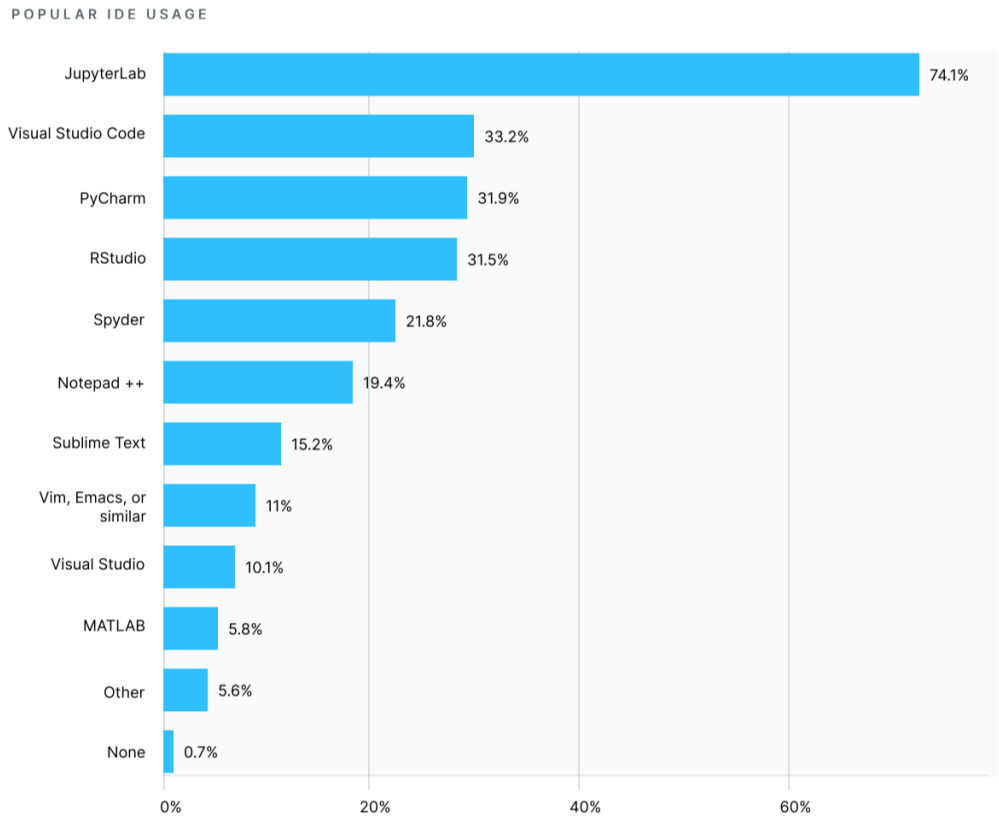


Python is Great for Prototyping

- **Interpreted**: direct execution without compilation
- **Dynamically-typed**: don't have to declare a static type
- **Readable**: easy-to-understand syntax
- **Deployable**: easy to incorporate in applications

- **Productivity**: facilitates rapid, interactive prototyping

Kaggle "State of Data Science" Survey 2020



[Source: https://www.kaggle.com/kaggle-survey-2020]

Python Overview

- general program syntax
- variables and types
  - integer and float numbers, string types, type conversion
  - list of values (list, array)
- condition statements (if/elif/else)
- for loops, ranges
- functions
  - input(), print(), len(), lower(), upper(), …
  - nesting of functions; example: print(len(*str*.upper()))

Python Import System

- Python has many *built-in* functions
- additional functionality available via **import** statement
  - gives access to classes and functions from various 3rd party modules
  - Example: **csv:** comma-separated file format support

```
import csv
for row in csv.reader( ['one,apple,green', 'two,tomato,red'] ):
    print(row[1])
```

alternatively read_csv()

  - alternative usages to introduce shortcuts or import only certain functions:

```
import csv as X
from csv import DictReader
```

also can use DataLoader

Python has excellent open-source data libraries

- **scipy**: libraries for scientific and technical computing
  - **numpy**: support for large multidimensional arrays and matrices
  - **matplotlib**: port of matlab plotting functionality
- **scikit-learn**: machine learning library
- **nltk**: natural language toolkit

seaborn

- **pandas**: R-like data frame and associated manipulations

Installing Python and Jupyter using Anaconda

- We make some Jupyter servers available
  - but this is a shared resource
- You can also install Python and Jupyter privately using the Anaconda Distribution, which includes Python, the Jupyter Notebook, and other commonly used packages for scientific computing and data science.

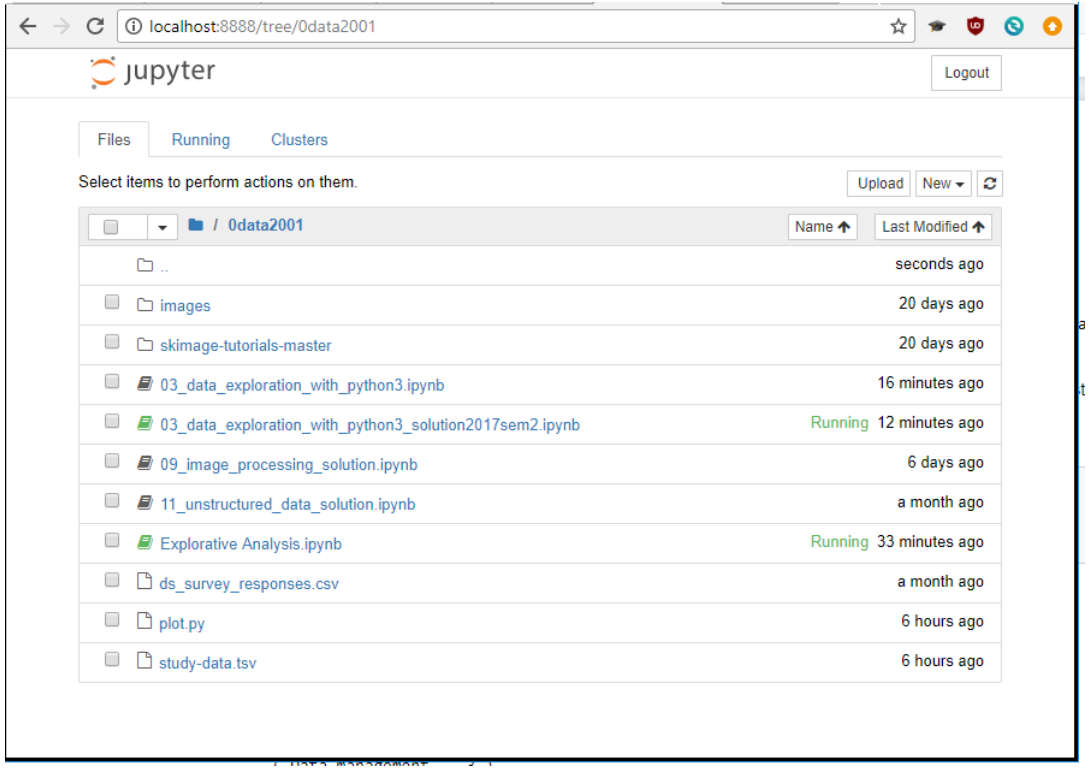
- Alternatively, try Google Colab (colab.research.google.com)

Jupyter Notebooks support interactive Data Science

- IPython interactive command shell offers:
  - Introspection
  - Tab completion
  - Command history
- Jupyter runs in a browser and supports:
  - Sharing and documenting of live code
  - Data cleaning, visualisation, machine learning, …
  - Jupyter's gallery of interesting notebooks: https://github.com/ipython/ipython/wiki/A-gallery-of-interesting-IPython-Notebooks

DEMO

- Jypyter Notebooks




THE UNIVERSITY OF
SYDNEY

Data Acquisition and Cleaning with Python

Example: Analysis of Major Power Stations in Australia

- dataset from data.gov.au

- How can we load this data into Python? → csv
- Which data preparation steps are needed? → data preprocessing

Source: https://data.gov.au/dataset/ds-ga-04661f51-82ee-144e-e054-00144fdd4fa6/details?q=power%20stations

Read Data into Python using csv

- Python **csv** module
  - Reads/writes comma-separated values with escaping to handle cases where comma occurs within a field
    - csv.reader reads rows into arrays
    - csv.DictReader reads rows into ***dictionaries***
- However: Not much support for further data handling or output…
  - E.g. convoluted syntax and type conversions needed
  - E.g. **pprint** module needed for pretty print complex data structure
    - pprint formats a dictionary read by CSV so it's easier to read

```
import csv
import pprint
data = list(csv.DictReader(open('MajorPowerStations.csv')))
pprint.pprint(data[0])
```

Pandas – Python Data Analysis Library

- Open source library providing data import and analysis functionality to Python
- https://pandas.pydata.org/
  - optimised data structures for data analysis
    - Tabular data (DataFrame)
    - Time series data (Series)
    - Matrixes
  - configurable input/output file
  - support for handling missing data, cleaning data, descriptive stats
- API documentation: https://pandas.pydata.org/pandas-docs/stable/reference/index.html

already pre-installed.

Pandas – Reading Data from a CSV file

- Pandas provides several reader functions for various file formats such as, for example, CSV
  - configurable with many options
    (cf. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html)

```
import pandas as pd
data = pd.read_csv('MajorPowerStations.csv')
data.head()
```

typical before starting data analysis.

Pandas: Fix Missing Values During Import

- Some datasets contain placeholders for missing values
  - such as ‘n/a’, ‘--’ or ‘null’
- Best to replace during import to avoid later problems

```
import pandas as pd

missing_values = ["--","<Null>"]
data = pd.read_csv('MajorPowerStations.csv', na_values = missing_values)
data.head()
```

Pandas – Missing Data Handling

- Pandas provides various functions for handling missing/wrong data
  - Part of this already included in the input functions (cf. **csv_read()**) where missing values are automatically replaced with NA/NaN
  - Other examples:
    - DataFrame.dropna() remove rows with any missing values
    - DataFrame.fillna() fill NA/NaN values using a specified method
    - DataFrame.replace() replace values

```
data2 = data['numGen'].dropna()

data['numGen'].fillna(0, inplace=True)

data['numGen'].replace(to_replace='<Null>', value=0, inplace=True)
```

Cleaning Data: Convert to Correct Types

- The standard Python **csv** module reads everything as string types
- **Pandas** is a bit better, but still will fallback to string if it can't deduce the type from all values in a column
- This will give problems sooner or later with stat functions or plots
- Need to convert as appropriate (e.g., int, float, timestamp)
  - int() creates integer objects, e.g., -1, 101
  - float() creates floating point object, e.g., 3.14, 2.71
  - datetime.strptime() creates datetime objects from strings

numpy.array as well

plots will look strange

Pandas Typing

- astype() function on Data series to convert to new types
  - Careful: fails if any entry in the series violates the new type
  - For example: ints do not support NaN
  - Also: does not support special value handling

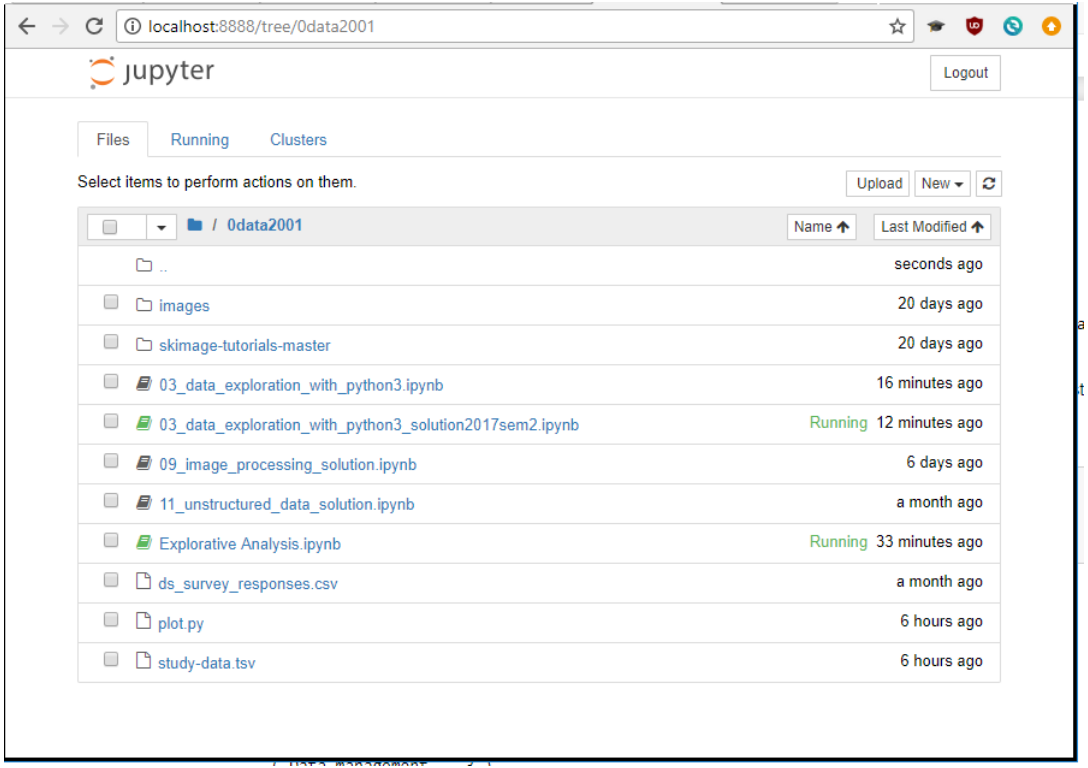
```
import pandas as pd
data = pd.read_csv('MajorPowerStations.csv')

data['numGenerator'] = data['numGenerator'].astype(int)
data['powerOutput']  = data['powerOutput'].astype(float)
```

always a good idea to return a copy of your work by doing:

```
new_data = old_data.copy()
```

DEMO

- Data Cleaning with Python and Jupyter




THE UNIVERSITY OF
SYDNEY

Descriptive Statistics with Python/Pandas

Pandas - Python Data Analysis Library

- Open source library providing data import and analysis functionality to Python
- https://pandas.pydata.org/
  - optimised data structures for data analysis
    - Tabular data → (DataFrame)
    - Time series data (Series)
    - Matrixes
  - configurable input/output file
  - support for handling missing data, cleaning data, descriptive stats
- API documentation: https://pandas.pydata.org/pandas-docs/stable/reference/index.html

Pandas – Data Structures

- Two main data structures:
  - **Series** (1-dimensional, labeled, homogenous typed)
  - **DataFrame** (2-dimensional, labeled, (potentially) heterogeneous columns)

- CSV reader imports a dataset as a **DataFrame**
  - Most Pandas functions also produce a DataFrame as output, hence multiple functions can be applied in sequence easily
  - http://pandas.pydata.org/pandas-docs/stable/reference/frame.html

```
data.axes
data.columns
data.dtypes
data['name'].count()
```

Descriptive Statistics with Pandas

https://www.tutorialspoint.com/python_pandas/python_pandas_descriptive_statistics.htm

- **DataFrame** supports a wide variety of data analysis functions

Let us now understand the functions under Descriptive Statistics in Python Pandas. The following table list down the important functions −

Sr.No.	Function	Description
1	count()	Number of non-null observations
2	sum()	Sum of values
3	mean()	Mean of Values
4	median()	Median of Values
5	mode()	Mode of values
6	std()	Standard Deviation of the Values
7	min()	Minimum Value
8	max()	Maximum Value
9	abs()	Absolute Value
10	prod()	Product of Values
11	cumsum()	Cumulative Sum
12	cumprod()	Cumulative Product

- Function application & GroupBy: groupby(), apply(), applymap()
- http://pandas.pydata.org/pandas-docs/stable/reference/frame.html

Examples of Descriptive Statistics on Numerical Data

```
import pandas as pd
data = pd.read_csv('MajorPowerStations.csv')

print( data['power'].min() )

print( data['power'].max() )

print( data['power'].mean() )

print( data['power'].median() )

print( data['power'].std() )
```

Examples of Descriptive Statistics: Mode

- Recall that the **mode** is the most frequent value
- Useful for categorial data where mean etc. are not defined

```
import pandas as pd
data = pd.read_csv('MajorPowerStations.csv')

# find most frequent class of power station
print( data['class'].mode() )

# find most frequent owner
print( data['owner'].mode() )
```

difference between loc[] and iloc[]:
loc is typically used for label indexing and can access multiple columns, while iloc is used for integer indexing

Filtering

- You can filter entries in a DataFrame using **loc[]**
  - Allows to specify a Boolean predicate where only those entries are selected in the DataFrame for which the predicate is True
  - Optionally also allows to select specific columns to keep in result

```
import pandas as pd
data = pd.read_csv('MajorPowerStations.csv')

# list of all photovoltaic solar power stations
solarStations = data.loc[ data[`type']=='Solar Photovoltaic' ]

# total power capacity of thermal solar stations
data.loc[ data[`type']=='Solar Thermal', 'power' ].sum()

# list of all large (>100 MW) wind power stations
largeWindParks = data.loc[(data[`type']=='Wind Turbine') & (data[`power']>100)]
```

Frequency Distributions using groupby() and size()

- Entries in a Pandas DataFrame can be grouped by a column

```
import pandas as pd

data = pd.read_csv('MajorPowerStations.csv')

classDistr = data.groupby('class').size()
print(classDistr)
```

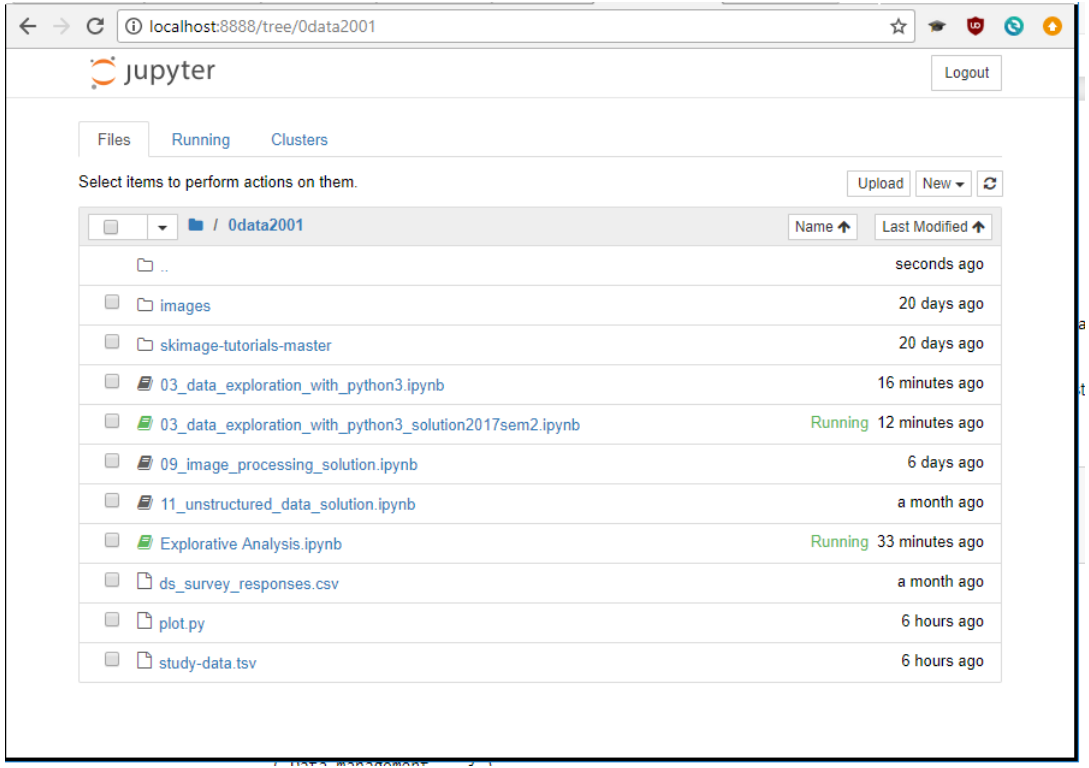
More Descriptive Statistics with numpy

- Another useful Python library is **numpy** ('Numerical Python')
- **Numpy** provides various statistics for numeric data
  - essential tool for multi-dimensional, array-oriented computing
- Median, percentiles, mean, standard deviation, etc
- nan* versions calculate same statistics, ignoring NaN values

- Reference page for numpy statistics:
  http://docs.scipy.org/doc/numpy/reference/routines.statistics.html

DEMO

- Data Exploration of Australian Power Stations Dataset




THE UNIVERSITY OF
SYDNEY

Data Visualisation with Python

Bar Chart, Histogram, Scatter Plot

Visualising data with Pandas and matplotlib

- Matplotlib provides functionality for creating various plots
- Bar charts, line charts, scatter plots, etc
- Pandas offers some easy-to-use shortcut functions

Definitely need matplotlib if we need some kind of viz.

- Reference page for Pandas plotting: https://pandas.pydata.org/pandas-docs/stable/reference/plotting.html
- Reference page for pyplot: http://matplotlib.org/api/pyplot_api.html
- Matplotlib Documentation: http://matplotlib.org/contents.html

- Pandas
- pyplot
- matplotlib

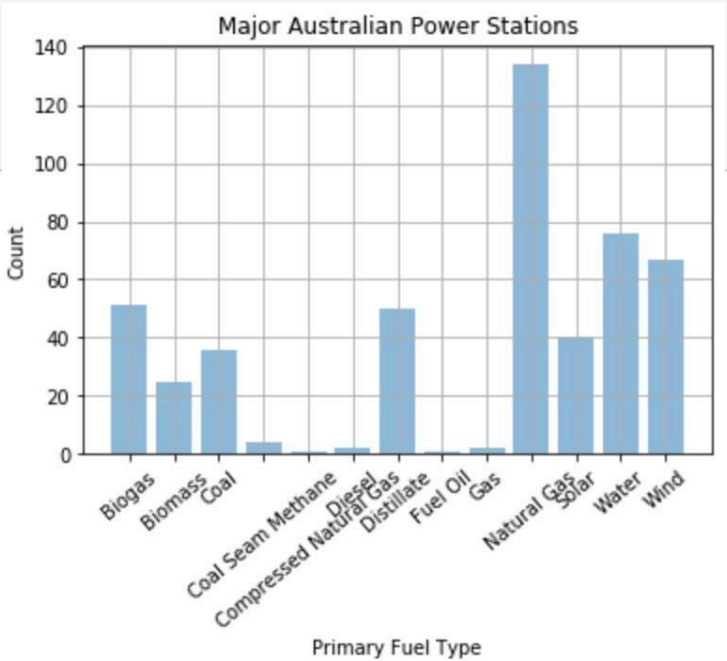
Creating a Bar Chart
for nominal / categorial data

Use groupby() to get frequency distribution
Rename resulting counts as 'numStations'

```
%matplotlib inline
fuelTypeDistr = wrkData.groupby('fueltype').size().reset_index(name='numStations')

# Plot
plt.bar(fuelTypeDistr['fueltype'], fuelTypeDistr['numStations'], alpha=0.5, align='center')
plt.xticks(rotation=40)
plt.title('Major Australian Power Stations')
plt.xlabel('Primary Fuel Type')
plt.ylabel('Count')
plt.grid()
```

Configure plot using matplotlib

Resulting plot ->



Plotting a Histogram

for continuous variables

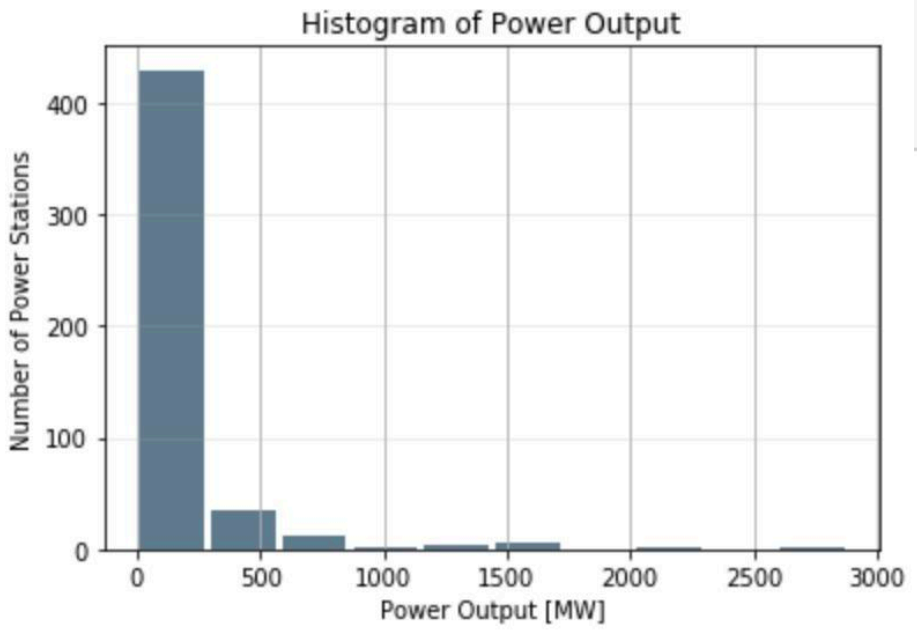
Create histogram plot with 10 bins of 'power' values

```
pyExpFreq = wrkData['power'].hist(bins=10, rwidth=0.9, color='#607c8e')
plt.title('Histogram of Power Output')
plt.xlabel('Power Output [MW]')
plt.ylabel('Number of Power Stations')
plt.grid(axis='y', alpha=0.25)
```

Configure plot

adjusts size of the grids

Resulting histogram ->



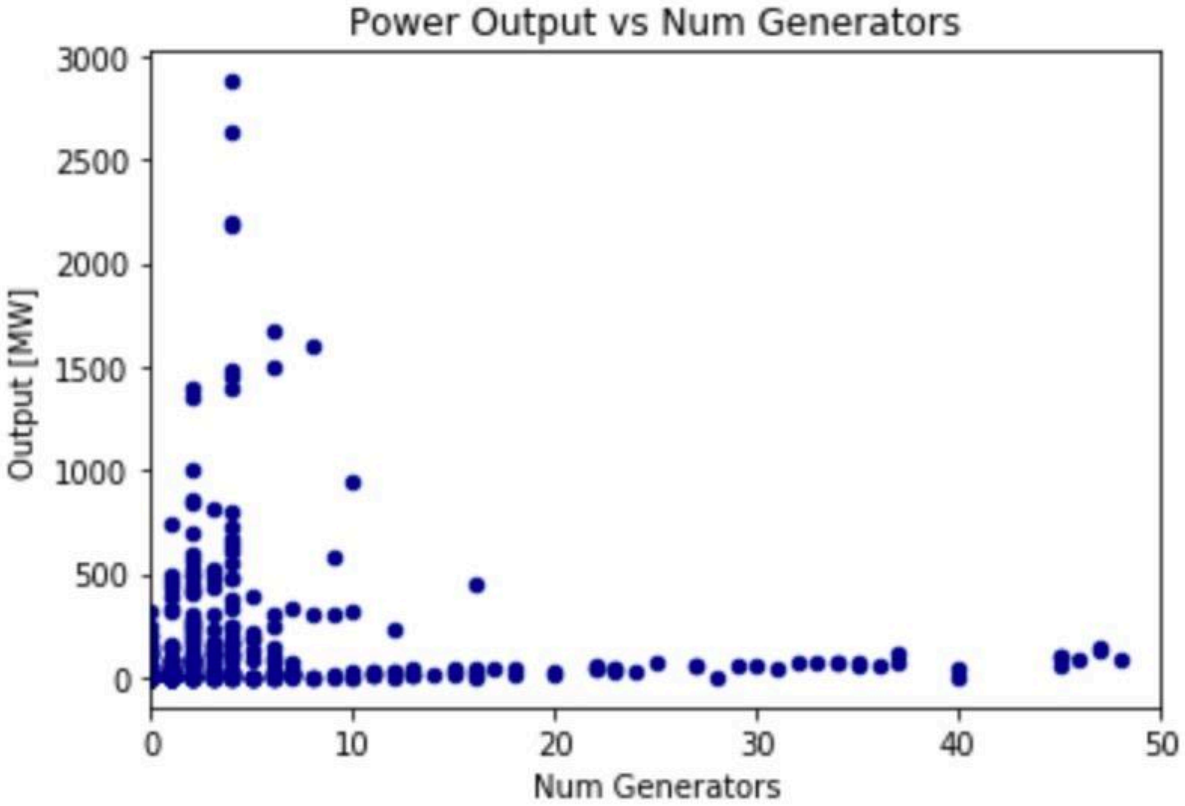
Creating a Scatter Plot

Quantitative variables

```
%matplotlib inline
import matplotlib.pyplot as plt

fig = plt.figure()
sub = plt.subplot()
wrkData.plot.scatter(x='numGen', y='power', c='DarkBlue', ax=sub)
sub.set_xlim(0,50)
plt.title('Power Output vs Num Generators')
plt.xlabel('Num Generators')
plt.ylabel('Output [MW]')
```

Create scatter plot

Scatter plot comparing Power Output vs. Generator Size



Creating a Scatter Plot with variable coloring/grayscale

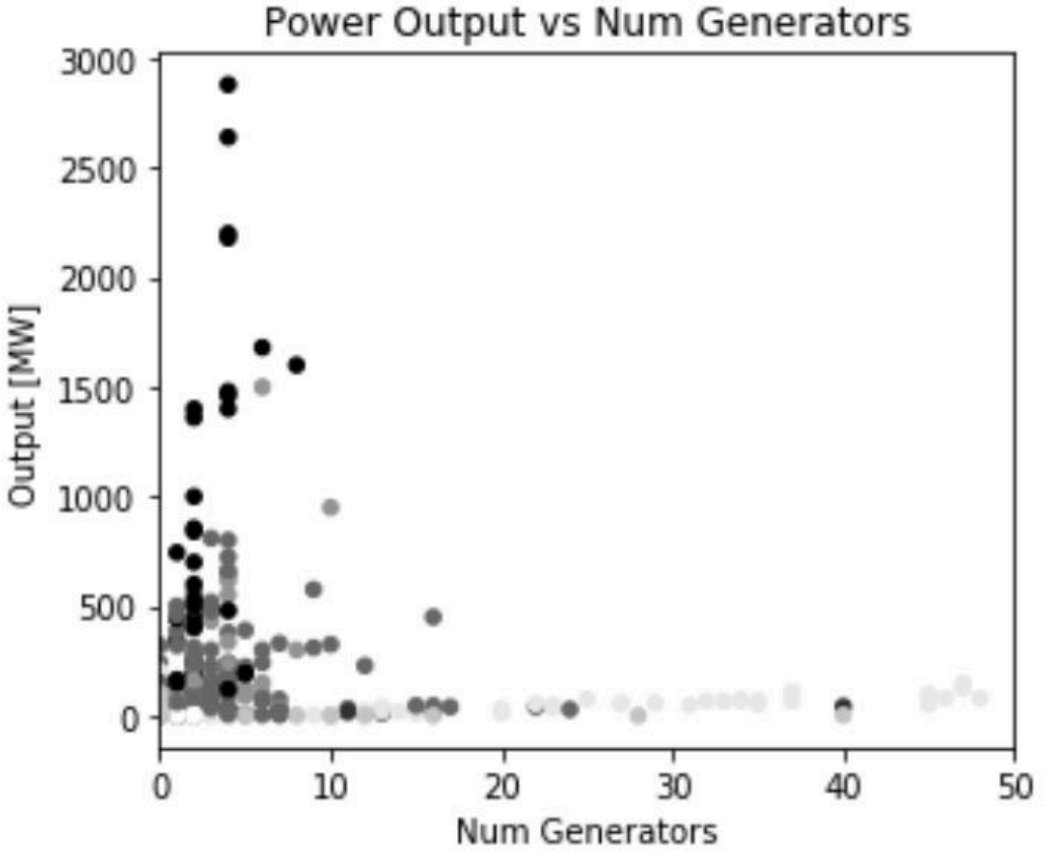
```
# assign colors to some selected fuel types
# the numbers and the order chosen are up-to you.
# we have chosen an order that works well with the color schemes used in the subsequent plots
wrkData['fuelEncoding'] = wrkData['fueltype'].map({
    'Biogas': 1,
    'Wind': 2,
    'Solar': 3,
    'Water': 4,
    'Natural Gas': 5,
    'Coal Seam Methane': 6,
    'Coal': 7
})
```

Encode fuel type into numerical values [1..7]

```
# Now we can use this encoding column to color our plot
%matplotlib inline

fig = plt.figure()
sub = plt.subplot()
wrkData.plot.scatter(x='numGen', y='power', c='fuelEncoding', ax=sub)
sub.set_xlim(0,50)
plt.title('Power Output vs Num Generators')
plt.xlabel('Num Generators')
plt.ylabel('Output [MW]')
```

Color by encoding values

Scatter Plot2 comparing Power Output vs. Generator Size



Coal

Coal Seam

Natural Gas

Water

Solar

Wind

Biogas

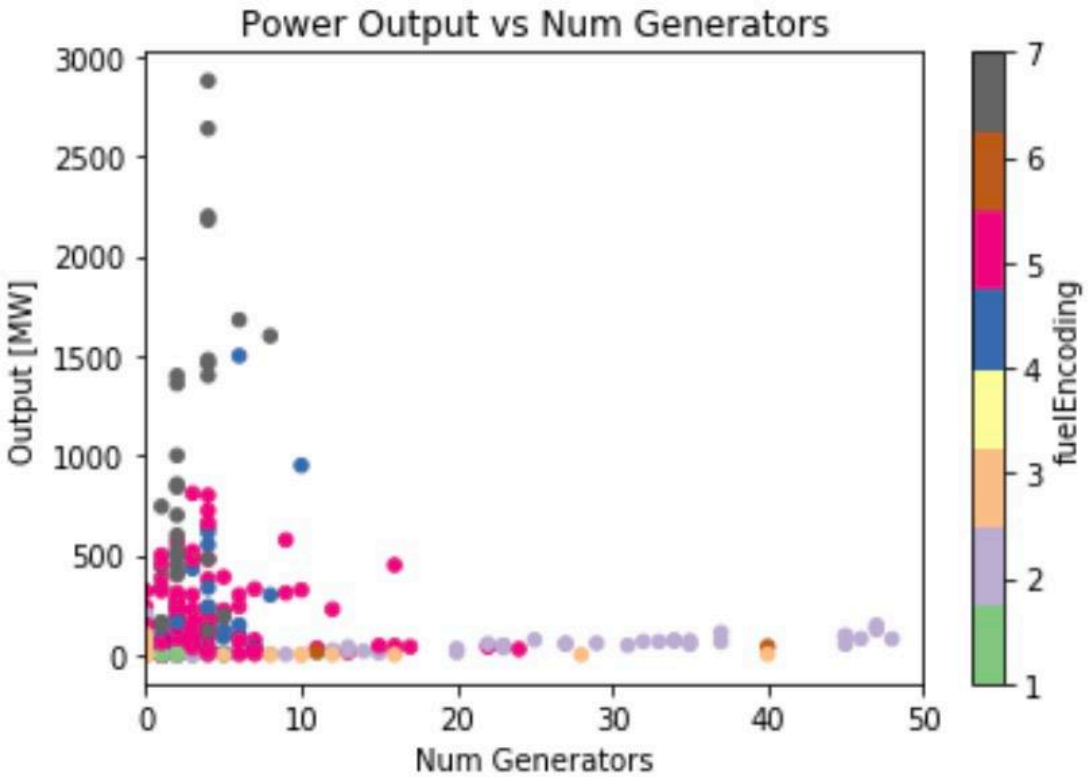
Creating a Scatter Plot with specific colormap

Using same encoding than before…

```
1  # the same plot as before, but using a more vivid color scheme (colormap='Accent')
2  # (for available colormaps from matplotlib, see https://matplotlib.org/3.1.0/tutorials/colors/colormap
3  %matplotlib inline
4
5  fig = plt.figure()
6  sub = plt.subplot()
7  wrkData.plot.scatter(x='numGen',y='power',c='fuelEncoding',colormap='Accent',ax=sub)
8  sub.set_xlim(0,50)
9  plt.title('Power Output vs Num Generators')
10 plt.xlabel('Num Generators')
11 plt.ylabel('Output [MW]')
```

Color using matplotlib's 'Accent' colormap

Scatter Plot2 comparing Power Output vs. Generator Size



Coal

Coal Seam

Natural Gas

Water

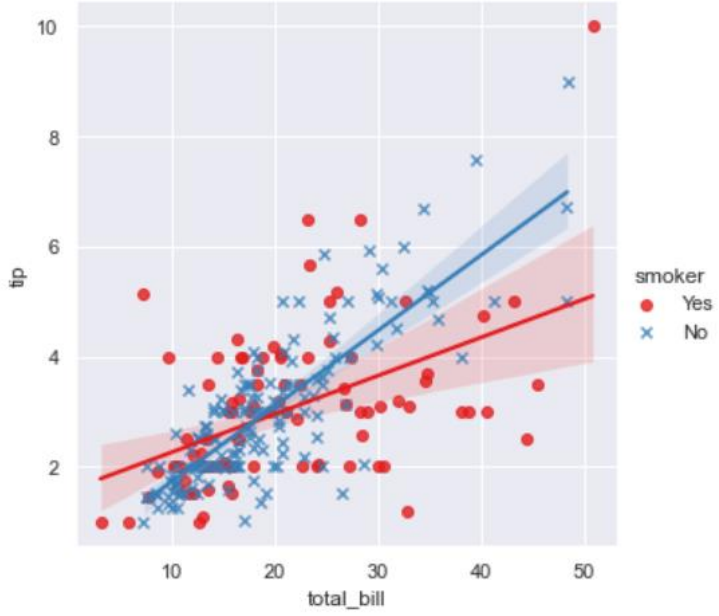
Solar

Wind

Biogas

Improving Visualisations: Seaborn library

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

sns.set_theme(color_codes=True)
tips = sns.load_dataset("tips")
sns.lmplot(x="total_bill", y="tip", hue="smoker", data=tips, markers=["o", "x"], palette="Set1");
```



always recommended for you to take some time to adjust the appearance of your visualisations if the results are meaningful.

Lessons Learned

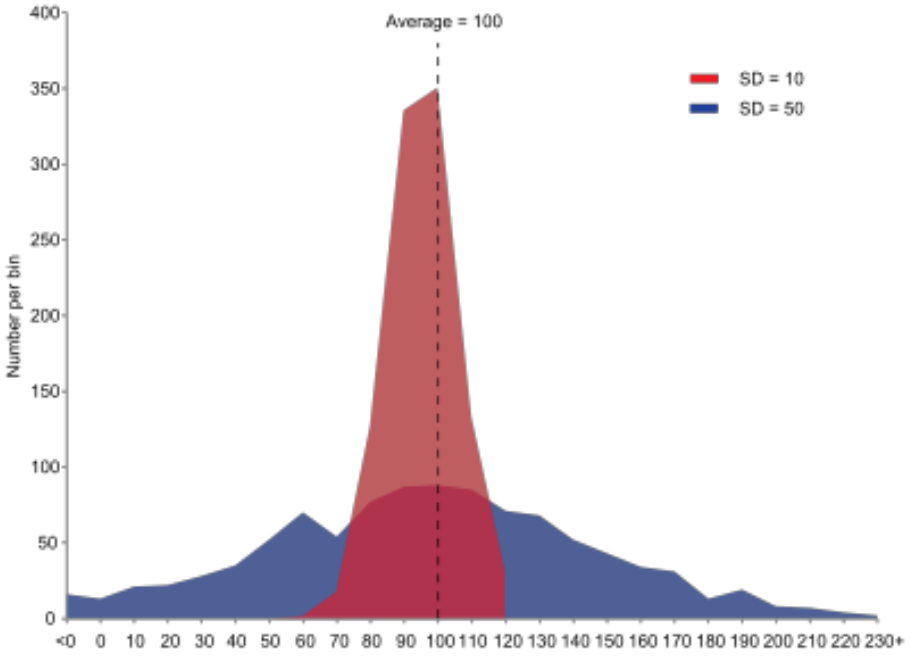
- **Bar Charts** for Categorial/Nominal Dara
- **Histograms** for Numerical Data
- **Scatter Plots** for comparing two continuing variables
  - Colouring (and/or shapes) to overlay categorial data

- Python libraries are your friend
  - Matplotlib,
  - many more, e.g.
  - many configuration options to adjust data visualisations to your needs


THE UNIVERSITY OF
SYDNEY

Box Plots

How to Analyse Data Distributions?

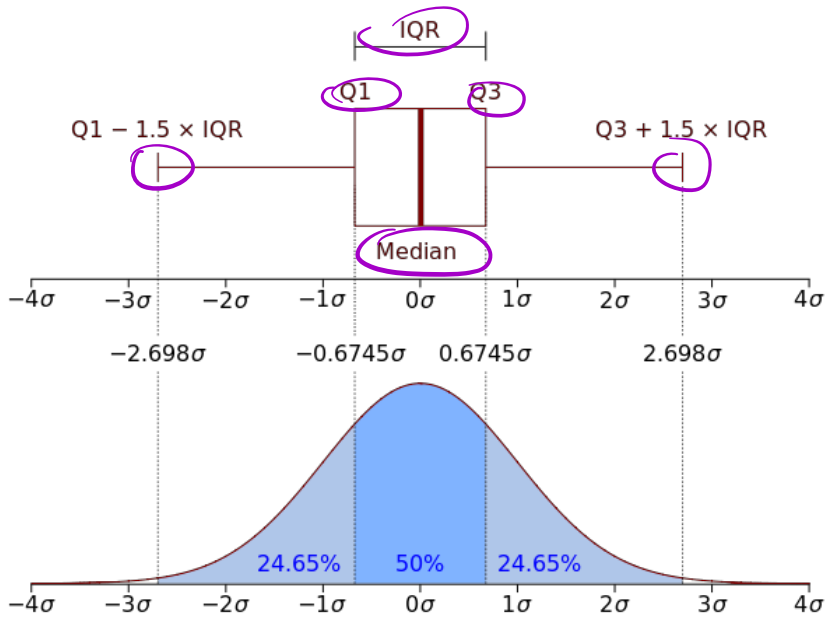


https://en.wikipedia.org/wiki/Variance

- Mean and stdev are not informative when data is skewed
- Left Figure: Samples from two populations with the same mean but different variances.
  - The red population has mean 100 and variance 100 (SD=10) while the blue population has mean 100 and variance 2500 (SD=50).

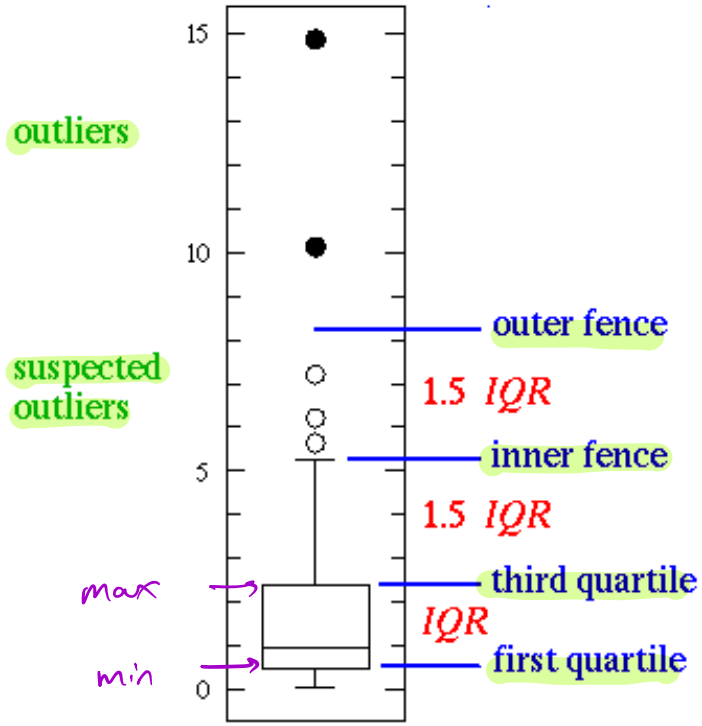
Using Boxplots to Compare Distributions

- A **boxplot** graphically summarises quantitative data by means of the following five numbers:
  - the *median* value;
  - the first and the third *quartiles* (*Q1* and *Q3*);
  - and the *minimum* and the *maximum* values as lower and upper fence.
  - Alternatively, we can express the value range by means of 1.5 IQR (*inter-quartile range*(*IQR*)) around the first and third quartiles.
    This is shown in the figure on the right:
  - Values outside fences are outliers



Boxplot of a normal distribution (after Tukey) [Source: Wikipedia (CC-BY-SA-2.5)]

Box Plot illustrated (incl. Outliers)

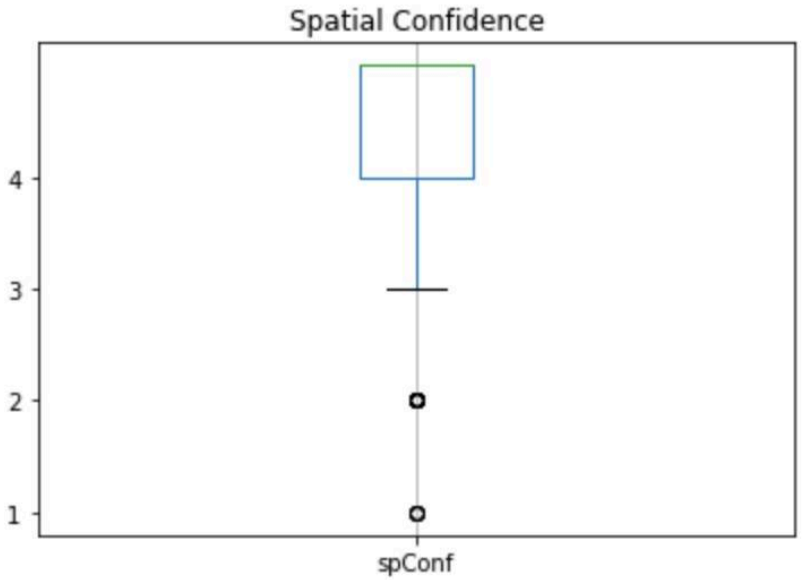


For a normal distribution the median line would be in the middle, otherwise they are skewed.

Boxplots using Pandas

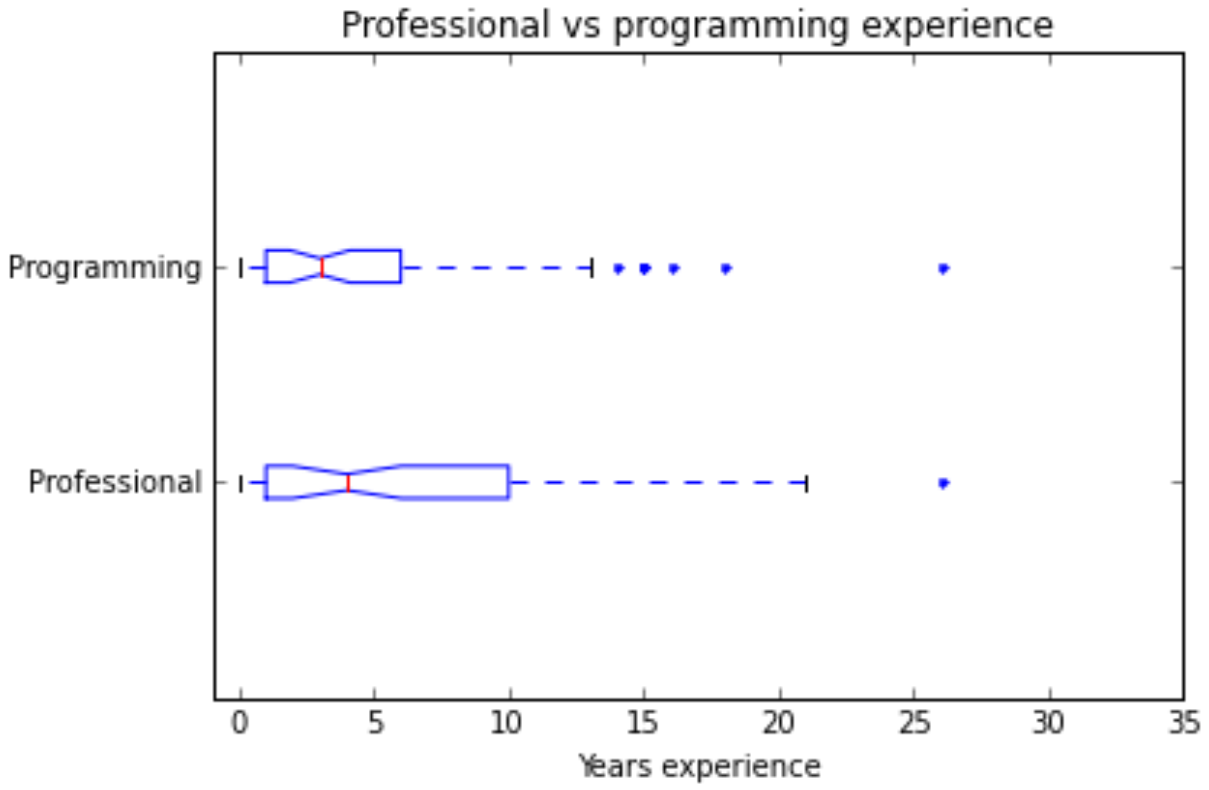
Example: box plot of the ‘spatial confidence’ values

```
%matplotlib inline

plt.yticks(np.arange(1, 5, 1.0))
fig = wrkData.boxplot(['spConf']).set_title('Spatial Confidence')
plt.grid(axis='y', alpha=0) # disable grid lines
```



spatial confidence on a likert scale of 1 to 5; 5 representing highest confidence in location data of power station

A box plot comparing experience distributions




THE UNIVERSITY OF
SYDNEY