CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

MODULE 1: INTRODUCTION TO DATA SCIENCE AND PYTHON

DAT405, 2019-2020, READING PERIOD 1

Car insurance prices based on Facebook posts

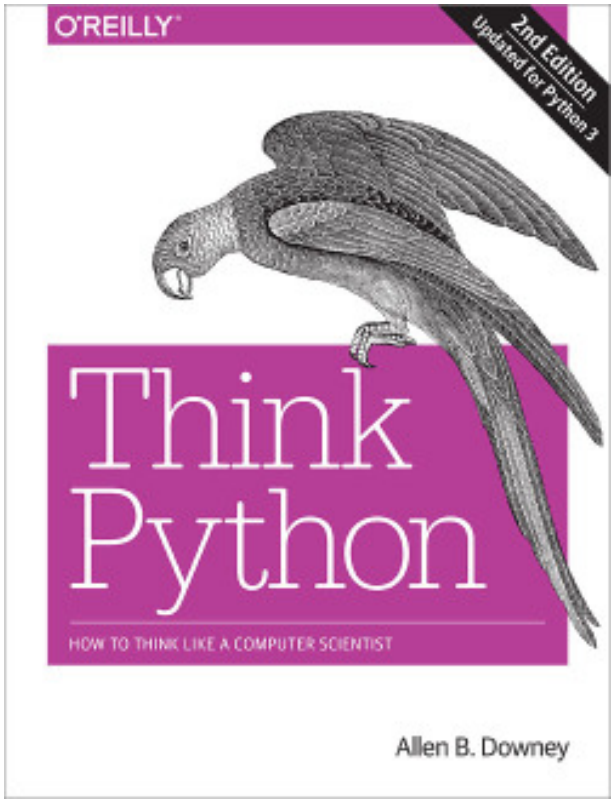
- “Admiral Insurance will analyse the Facebook accounts of first-time car owners to look for personality traits that are linked to safe driving. For example, individuals who are identified as conscientious and well-organised will score well.”
- “These [traits] include writing in short concrete sentences, using lists, and arranging to meet friends at a set time and place, rather than just “tonight”.”
- “In contrast, evidence that the Facebook user might be overconfident – such as the use of exclamation marks and the frequent use of “always” or “never” rather than “maybe” – will count against them.”
- “The scheme is voluntary, and will only offer discounts rather than price increases”

https://www.theguardian.com/technology/2016/nov/02/admiral-to-price-car-insurance-based-on-facebook-posts

Analyzing data with Python

Python programming

- Good for those with no previous programming experience

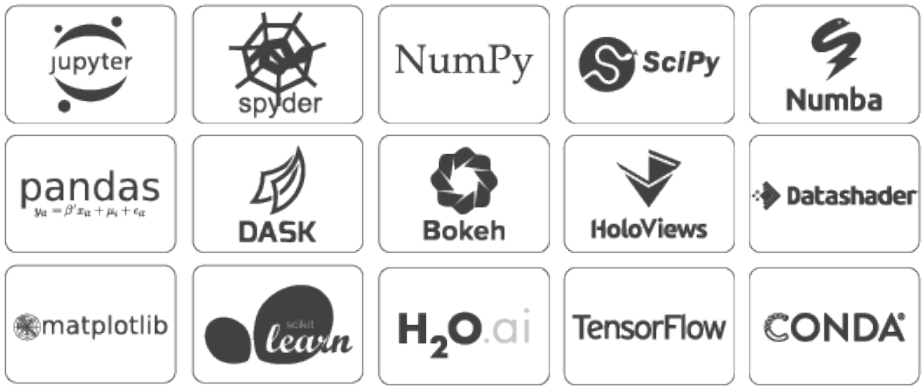


Python programming

- Faster pace than “Think Python”

ANACONDA

- Free and open source distribution of Python and R
- Over 1500 packages
- Anaconda Navigator includes:
  - Jupyter Notebook
  - Spyder – an integrated development environment (IDE) for Python



https://www.anaconda.com/

Python packages

Lots! including:

- NumPy
- SciPy
- Pandas
- Matplotlib
- Seaborn
- Scikit-learn

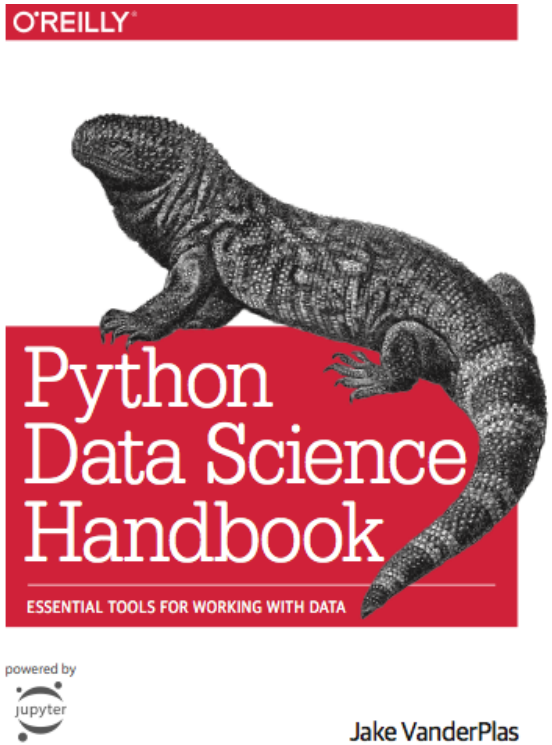
To use the functions in a module or a package, these have to be imported, e.g.

```
import pandas

import numpy as np

from sklearn.linear_model import LinearRegression
```

Python programming

- Assumes some knowledge of Python

- Focuses on using packages like NumPy, Pandas, Matplotlib, Scikit-learn.



Pandas

Provides data structures for working with data, e.g.

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

Pandas data structures

Series

- 1D labeled homogeneously-typed array

DataFrame

- General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed columns

Reading data with pandas

- Suppose GDP-2015.csv is a text file containing tabular data about gross domestic product per capita as comma-separated values.

```
Entity,Code,Year,GDP per capita
Afghanistan,AFG,2015,1928
Albania,ALB,2015,10947
Algeria,DZA,2015,13024
Angola,AGO,2015,8631
Argentina,ARG,2015,19316
...
```

- Read a comma-separated values (csv) file into DataFrame.
- Select the column, as a *Series*

```
import pandas

df = pandas.read_csv("GDP-2015.csv")
gdp = df['GDP per capita']
```

Pandas data structures

Series

- 1D labeled homogeneously-typed array

DataFrame

- General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed columns

```
>>> import pandas

>>> df = pandas.read_csv("GDP-2015.csv")
>>> gdp = df['GDP per capita’]


>>> type(df)
<class 'pandas.core.frame.DataFrame'>
>>> type(gdp)
<class 'pandas.core.series.Series'>
>>>
```

pandas.DataFrame.describe

- Generates descriptive statistics that summarize the central tendency, dispersion and shape of a dataset's distribution.
- Analyzes both numeric and object series, as well as DataFrame column sets of mixed data types. The output will vary depending on what is provided.

```
>>> df.describe()
              Year  GDP per capita
count    167.0       167.000000
mean    2015.0     18216.598802
std        0.0     19305.364946
min     2015.0       605.000000
25%     2015.0      3705.000000
50%     2015.0     11738.000000
75%     2015.0     25843.000000
max     2015.0    139542.000000
```

NumPy

- Provides support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

```
print( "Mean GDP is", numpy.mean(gdp) )
print( "Standard deviation is", numpy.std(gdp) )
print( "Standard deviation (divide by N-1) is", numpy.std(gdp, ddof=1) )
print( "Count is", numpy.count_nonzero(gdp) )
print( "Minimum GDP is", min(gdp) )
print( "Maximum GDP is", max(gdp) )
print( "25 percentile of GDP is", numpy.percentile(gdp, 25) )
print( "50 percentile of GDP is", numpy.percentile(gdp, 50) )
print( "75 percentile of GDP is", numpy.percentile(gdp, 75) )
print( "Sum is", numpy.sum(gdp) )
print( "Median is", numpy.median(gdp) )
```

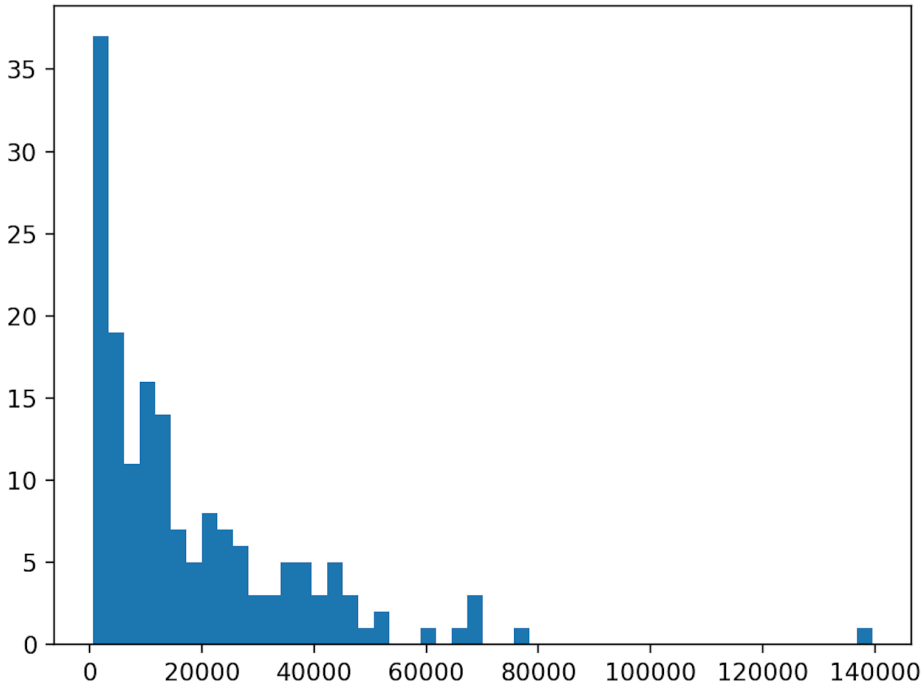
```
Mean GDP is 18216.59880239521
Standard deviation is 19247.477664769667
Standard deviation (divide by N-1) is 19305.36494634497
Count is 167
Minimum GDP is 605
Maximum GDP is 139542
25 percentile of GDP is 3705.0
50 percentile of GDP is 11738.0
75 percentile of GDP is 25843.0
Sum is 3042172
Median is 11738.0
```

Data visualisation

- Matplotlib
  - A plotting library for drawing plots, histograms, scatter plots, etc.
  - pyplot module provides a MATLAB-like interface
  - https://matplotlib.org/gallery/index.html

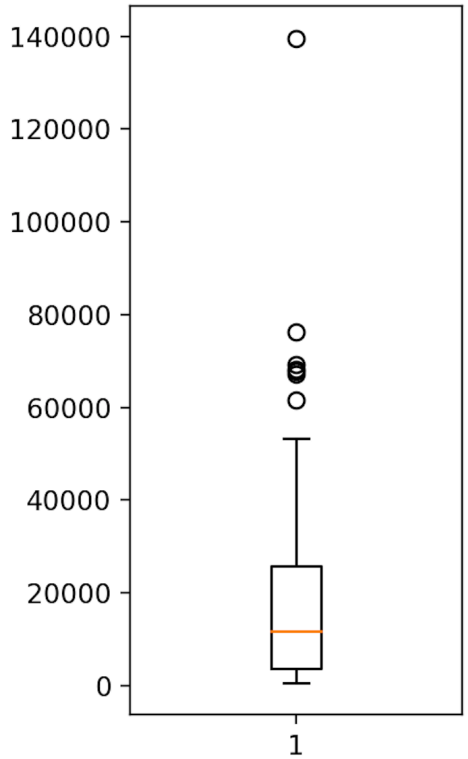
- Seaborn
  - a Python data visualisation library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics
  - https://seaborn.pydata.org/examples/index.html

Histogram in matplotlib



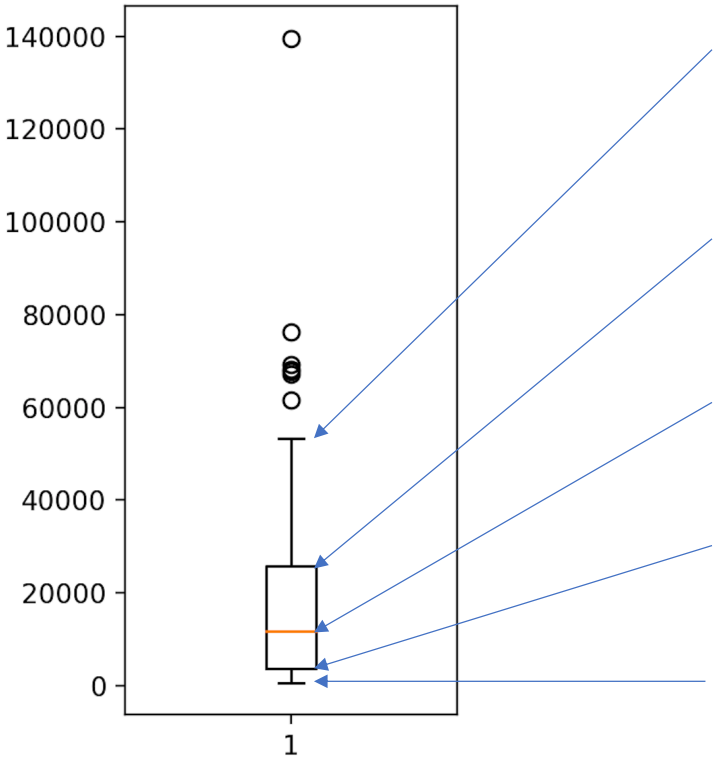

```
import pandas
import matplotlib.pyplot as plt

df = pandas.read_csv("GDP-2015.csv")
gdp = df['GDP per capita']

plt.hist(gdp, bins=50)
plt.show()
```

Boxplot in matplotlib




```
import pandas
import matplotlib.pyplot as plt

df = pandas.read_csv("GDP-2015.csv")
gdp = df['GDP per capita']

plt.boxplot(gdp)
plt.show()
```

Boxplot in matplotlib



Upper whisker, e.g. largest data value lower than Q3 + 1.5 * (interquartile range)

Q3 (3rd quartile)

median

Q1 (1st quartile)

Lower whisker

Urban Atlas 2012

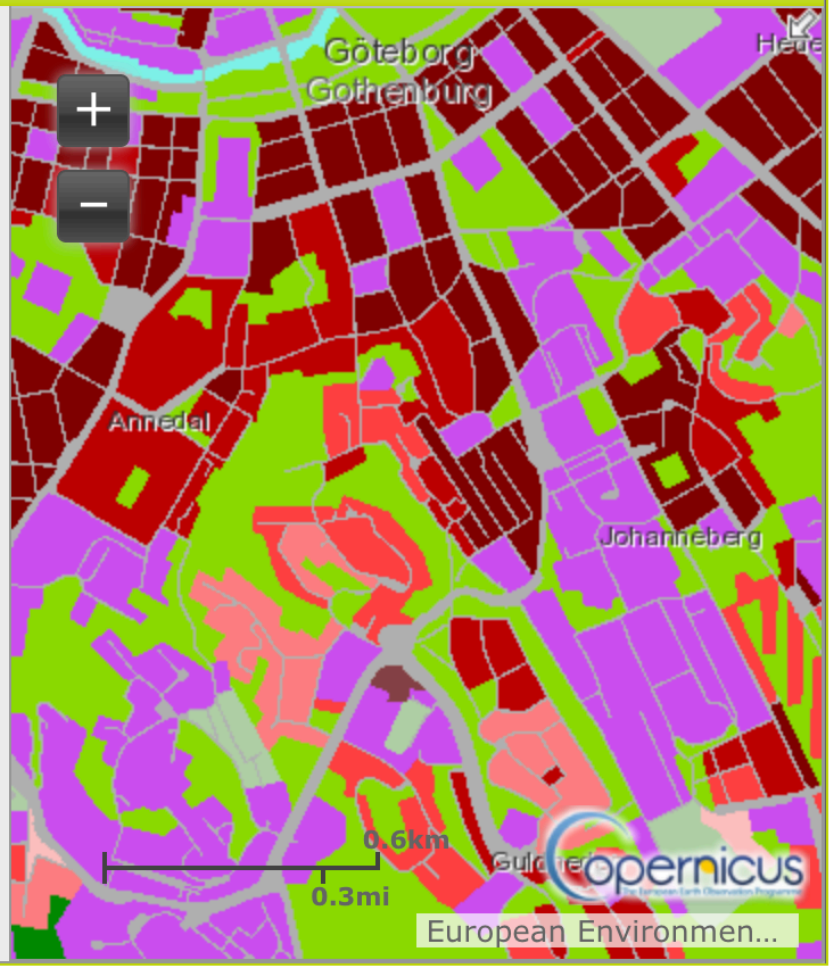
Print

Map View | **Metadata** | **Download**

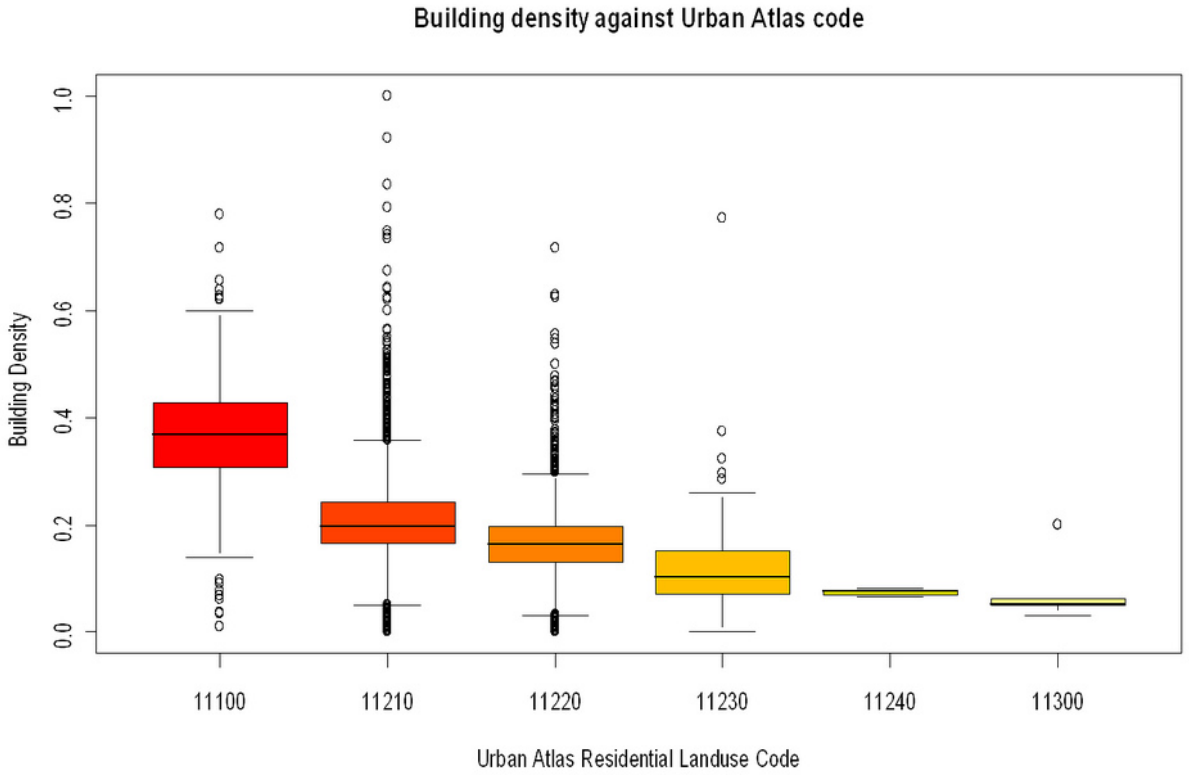
Layers | Legend | Web services

Land Use vector

- 11100: Continuous Urban fabric (S.L. > 80%)
- 11210: Discontinuous Dense Urban Fabric (S.L.: 50% - 80%)
- 11220: Discontinuous Medium Density Urban Fabric (S.L.: 30% - 50%)
- 11230: Discontinuous Low Density Urban Fabric (S.L.: 10% - 30%)
- 11240: Discontinuous very low density urban fabric (S.L. < 10%)
- 11300: Isolated Structures
- 12100: Industrial, commercial, public, military and private units
- 12210: Fast transit roads and associated land
- 12220: Other roads and associated land
- 12230: Railways and associated land



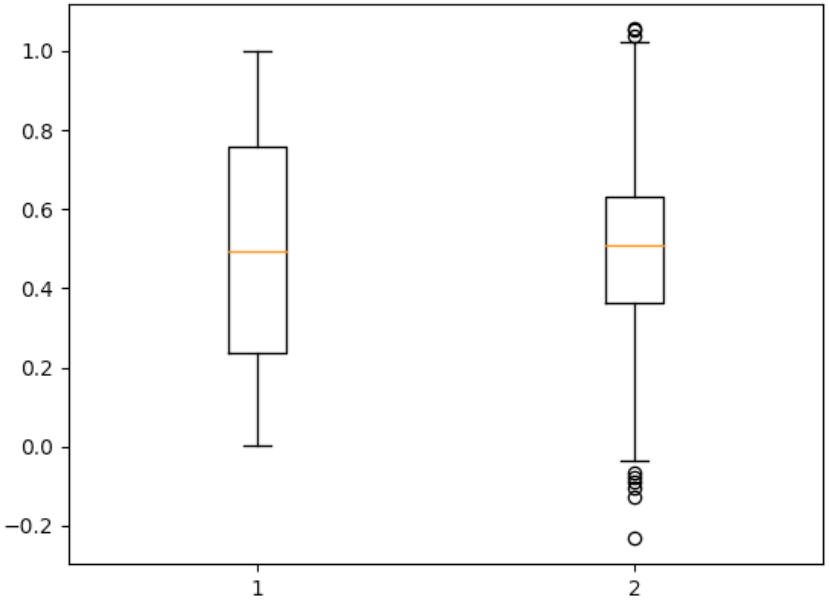
Box plots for stratified data



Data for around 7000 landuse parcels in Nottingham

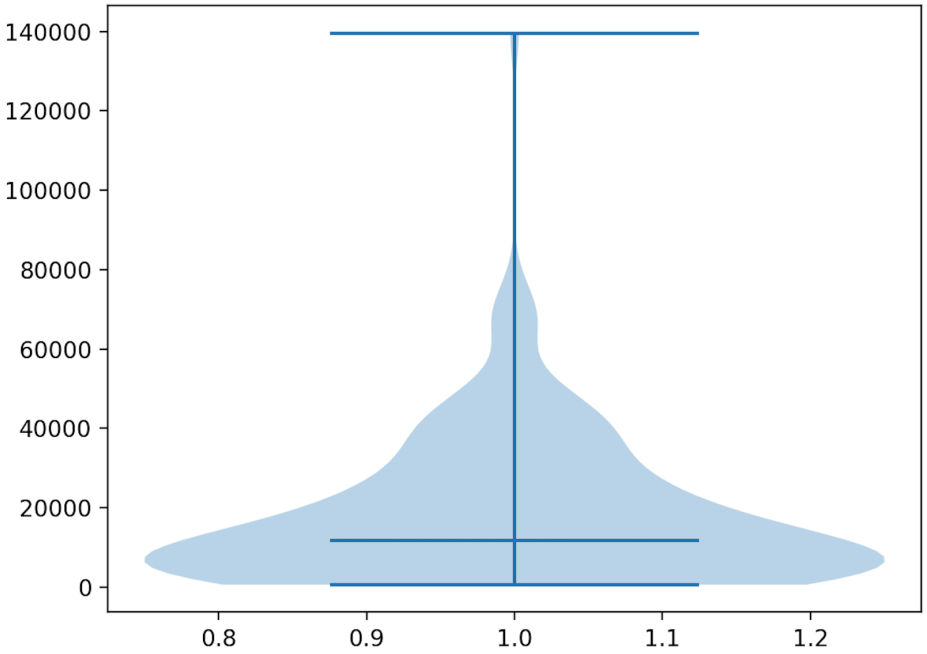
https://www.flickr.com/photos/sk53_osm/6021454611

Box plots for stratified data




```
import matplotlib.pyplot
matplotlib.pyplot.boxplot([data1,data2])
matplotlib.pyplot.show()
```

Violin plot in matplotlib




```
import pandas
import matplotlib.pyplot as plt

df = pandas.read_csv("GDP-2015.csv")
gdp = df['GDP per capita']

plt.violinplot(gdp, showmedians=True)
plt.show()
```

SciPy

- numerical routines such as routines for numerical integration, interpolation, optimization, linear algebra and statistics.

Scatter plots

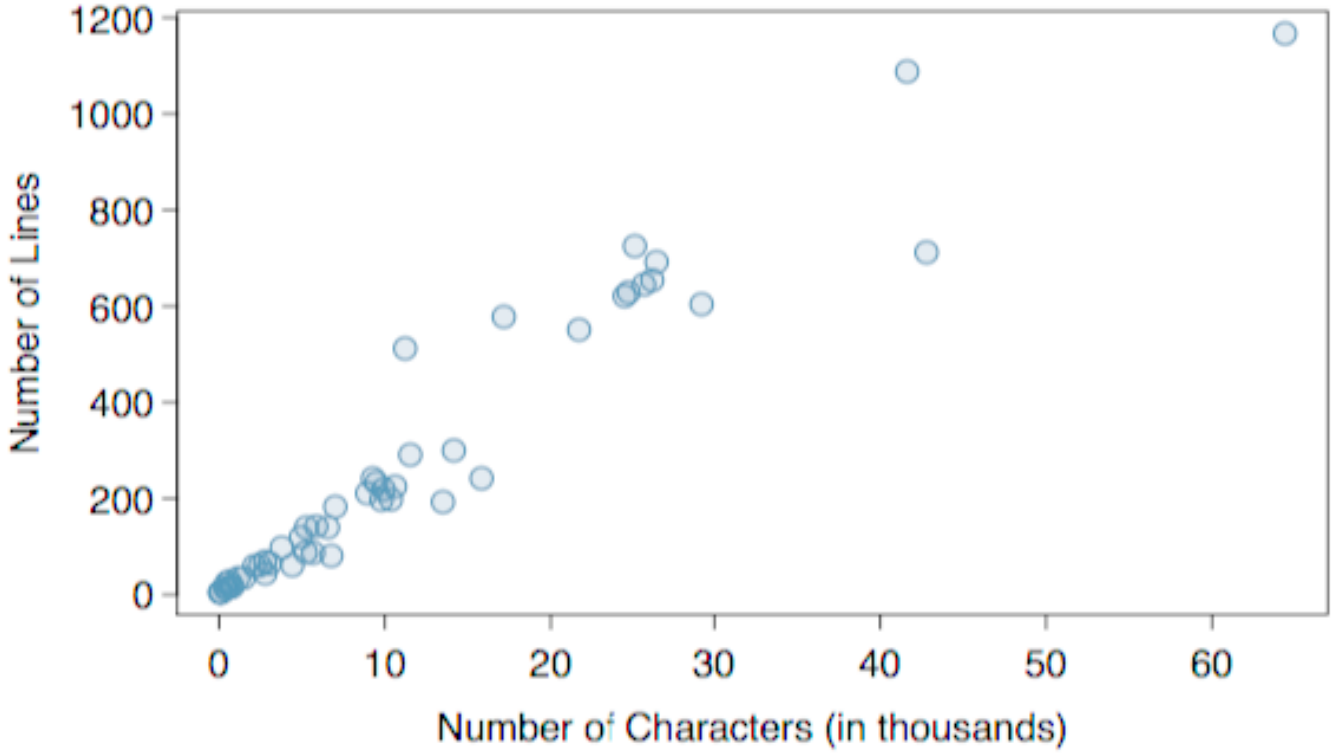


Figure 1.17: A scatterplot of `line_breaks` versus `num_char` for the `email50` data.

Figure from OpenIntro Statistics

Developing a visualisation aesthetic

- Maximise data-ink ratio
- Minimise the lie factor
- Minimise chart junk
- Use proper scales and clear labelling
- Make effective use of colour
- Exploit the power of repetition

Limits of mean and variance

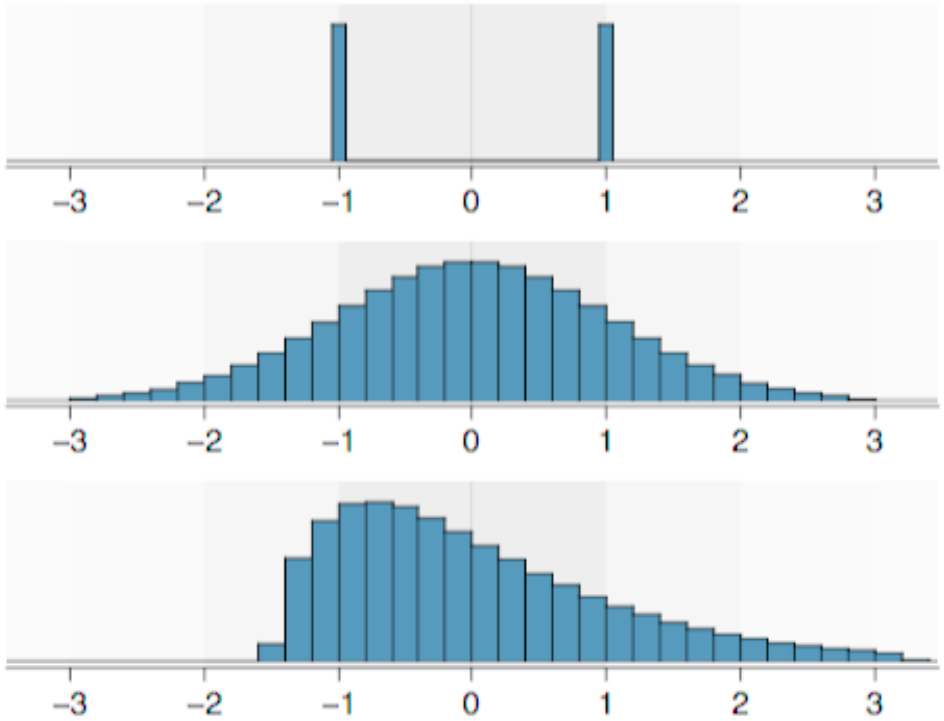
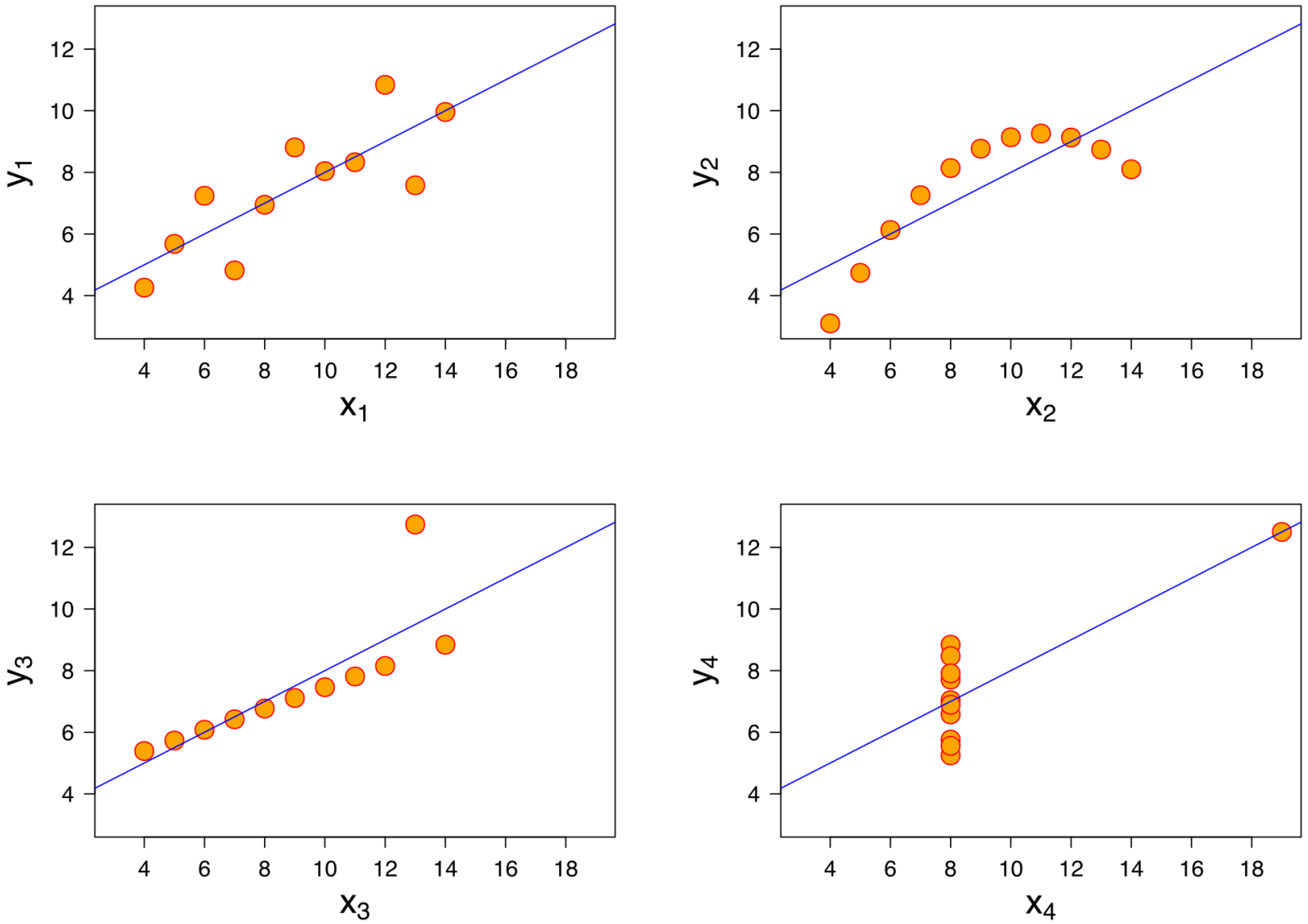


Figure 1.25: Three very different population distributions with the same mean $\mu = 0$ and standard deviation $\sigma = 1$.

Figure from OpenIntro Statistics

Consider these four data sets

x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89


y1
x1
y2
x2
y3
x3
y4
x4
4
6
8
10
12
14
16
18
4
6
8
10
12

Command line arguments

d1.txt

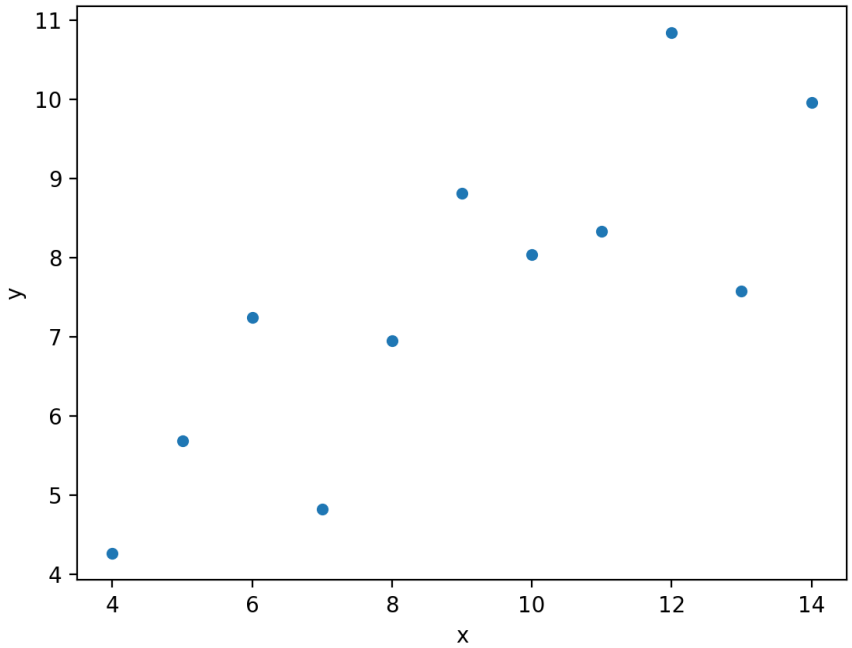
```
x y
10.0 8.04
8.0 6.95
13.0 7.58
9.0 8.81
11.0 8.33
14.0 9.96
6.0 7.24
4.0 4.26
12.0 10.84
7.0 4.82
5.0 5.68
```

```
# File "read.py"

import sys
import pandas

df = pandas.read_csv(sys.argv[1], sep=' ')
print(df)
```

```
$ python read.py d1.txt
       x      y
0   10.0   8.04
1    8.0   6.95
2   13.0   7.58
3    9.0   8.81
4   11.0   8.33
5   14.0   9.96
6    6.0   7.24
7    4.0   4.26
8   12.0  10.84
9    7.0   4.82
10   5.0   5.68
$
```

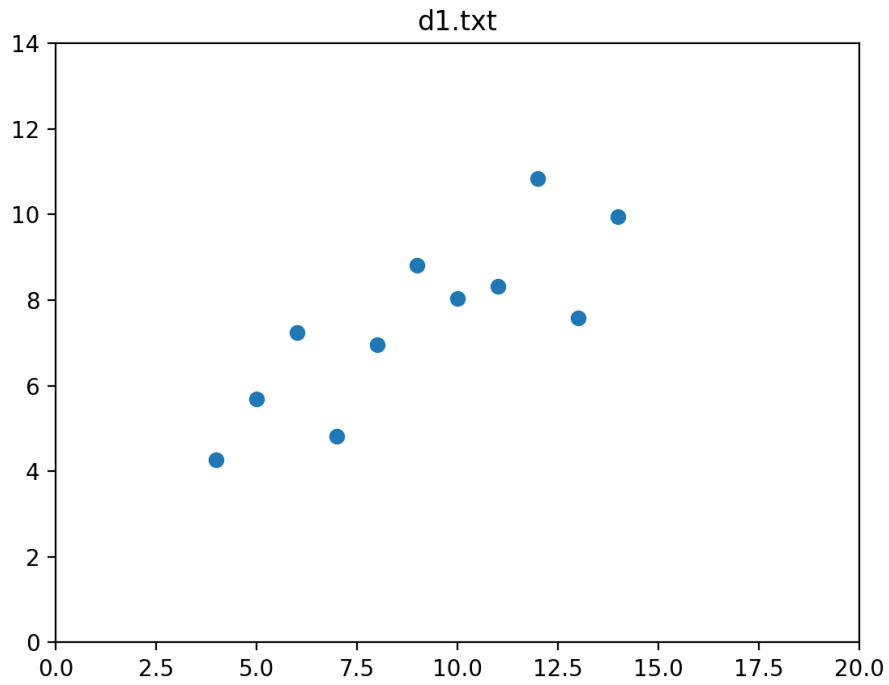
Read data from a file, then plot it

```
import sys
import pandas
import matplotlib.pyplot as plt

df = pandas.read_csv(sys.argv[1], sep=' ')

df.plot.scatter(x='x', y='y')
plt.show()
```



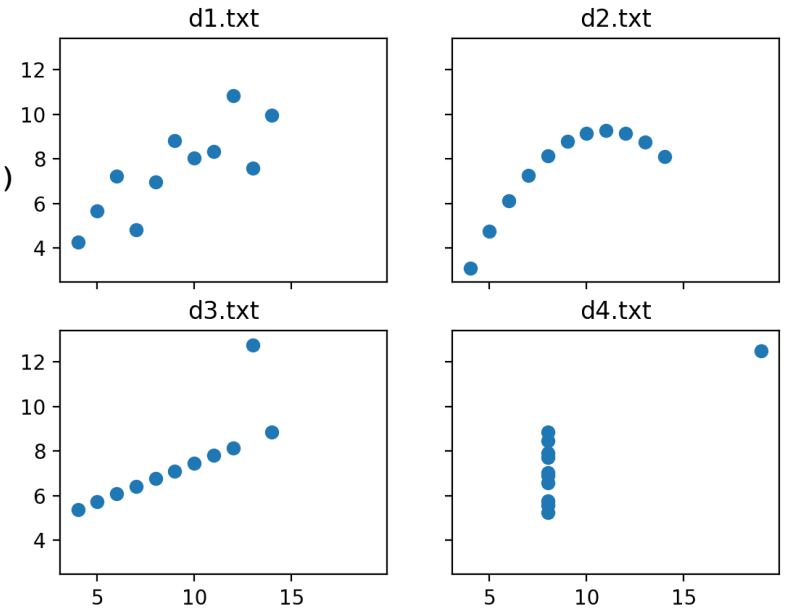
Customise plot: choose ranges and add title

```
import sys
import pandas
import matplotlib.pyplot as plt

df = pandas.read_csv(sys.argv[1], sep=' ')
print(df)

xValues = df['x']
yValues = df['y']

plt.axis([0, 20, 0, 14])
plt.scatter(xValues, yValues)
plt.title(sys.argv[1])
plt.show()
```



Subplots

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

fig, axs = plt.subplots(2, 2, sharex = 'all', sharey = 'all')

for i in range(4):
    df = pandas.read_csv(sys.argv[i+1], sep=' ')
    xValues = df['x']
    yValues = df['y']
    axs[ i // 2, i % 2 ].scatter(xValues, yValues)
    axs[ i // 2, i % 2 ].set_title(sys.argv[i+1])

# Hide x labels and tick labels for top plots
# and y ticks for right plots.
for ax in axs.flat:
    ax.label_outer()

plt.show()
```



Discussing ethical issues

- **Identify stakeholders**
- **Identify benefits and possible harm for each stakeholder**
- **Weigh benefits against possible harm**

- Get input from others

Determining user personality characteristics from social networking system communications and characteristics

- ”A social networking system obtains linguistic data from a user's text communications on the social networking system. For example, occurrences of words in various types of communications by the user in the social networking system are determined. The linguistic data and non-linguistic data associated with the user are used in a trained model to predict one or more personality characteristics for the user. The inferred personality characteristics are stored in connection with the user's profile, and may be used for targeting, ranking, selecting versions of products, and various other purposes.”

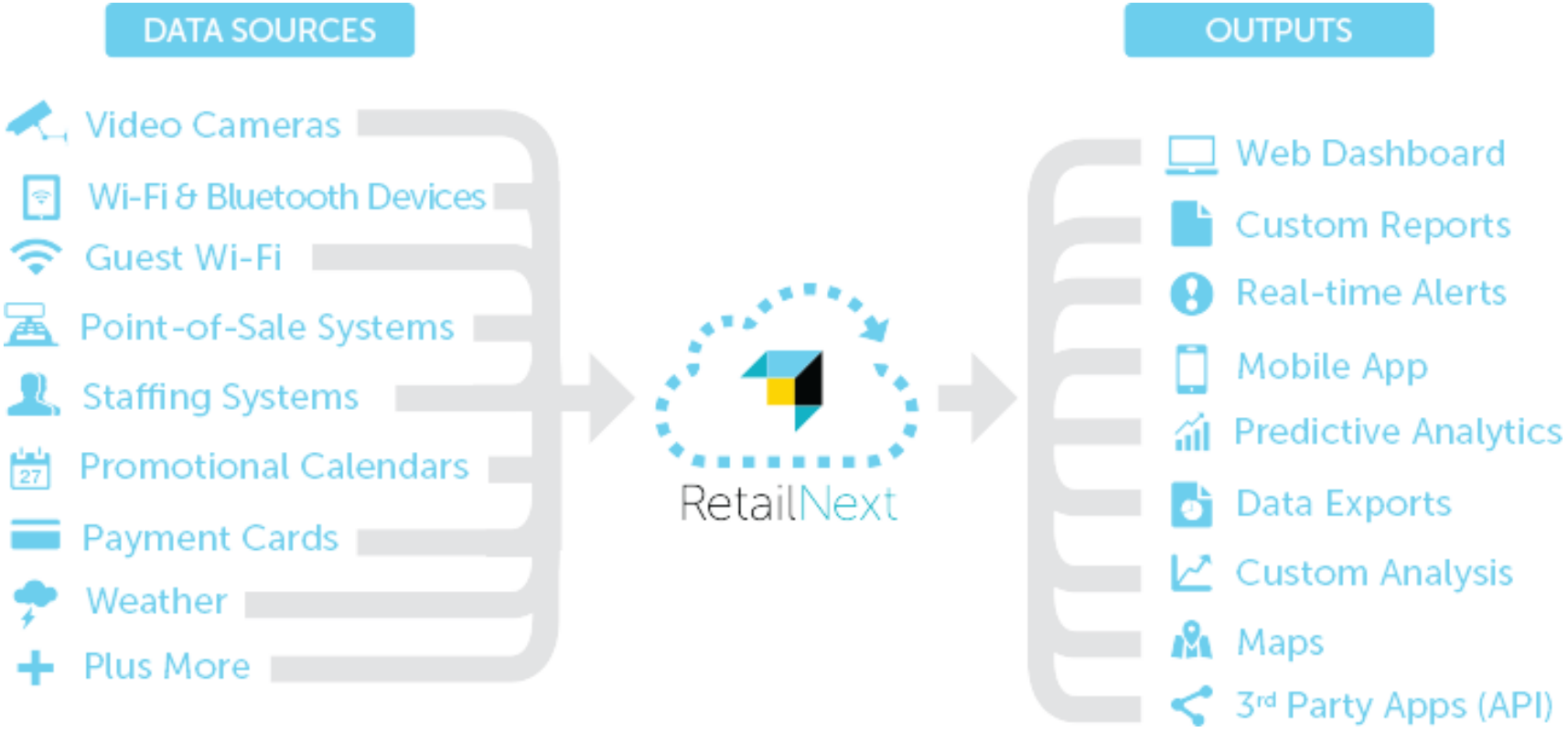
US patent US8825764B2

Emotion recognition in video conferencing

- ”… when faces or other parts of video conference participants are detected, the present technology determines an emotional status of video conference participants. This may include identification of facial expressions or changes in facial expressions over time. The emotional status can be also partly based on speech recognition or voice analysis. If it is determined that the emotional status is negative, an alert communication can be generated and transmitted to one of a video conference participant or a third party.”

US patent US9576190B2

RetailNext in-store analytics



Feet, not faces?

- ”market leading technology provides 95% footfall accuracy and is 80% accurate at identifying gender. The technology is unobtrusive and sensors can be easily installed throughout and outside a store. Cameras point to the ground, so intelligence can be gathered while preserving people's privacy. The proprietary technology can track outside traffic, in-store occupancy, dwell times, group size, demographic details and brand recognition to provide real-time data with the potential for rich analysis.”

[Hoxton Analytics web site, 2018]

Modules 2 and 3

In the next two weeks we shall look at some core data science tasks:

- Regression
  - Predicting a numerical quantity
- Classification
  - Assigning a label from a discrete set of possibilities
- Clustering
  - Grouping items by similarity