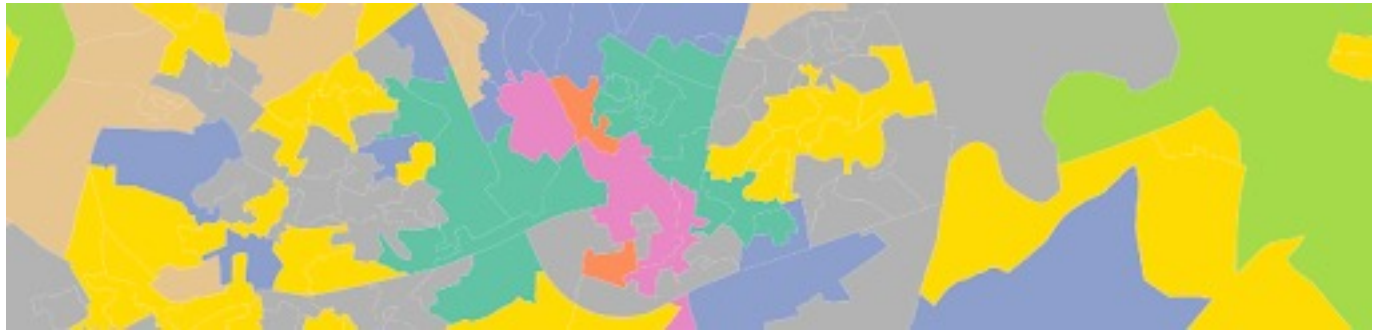
Principles of Spatial Analysis

WEEK 09: GEODEMOGRAPHICS



UCL

This week

- Geodemographic classification
- k-means clustering
- Bonus section on speeding up code in R

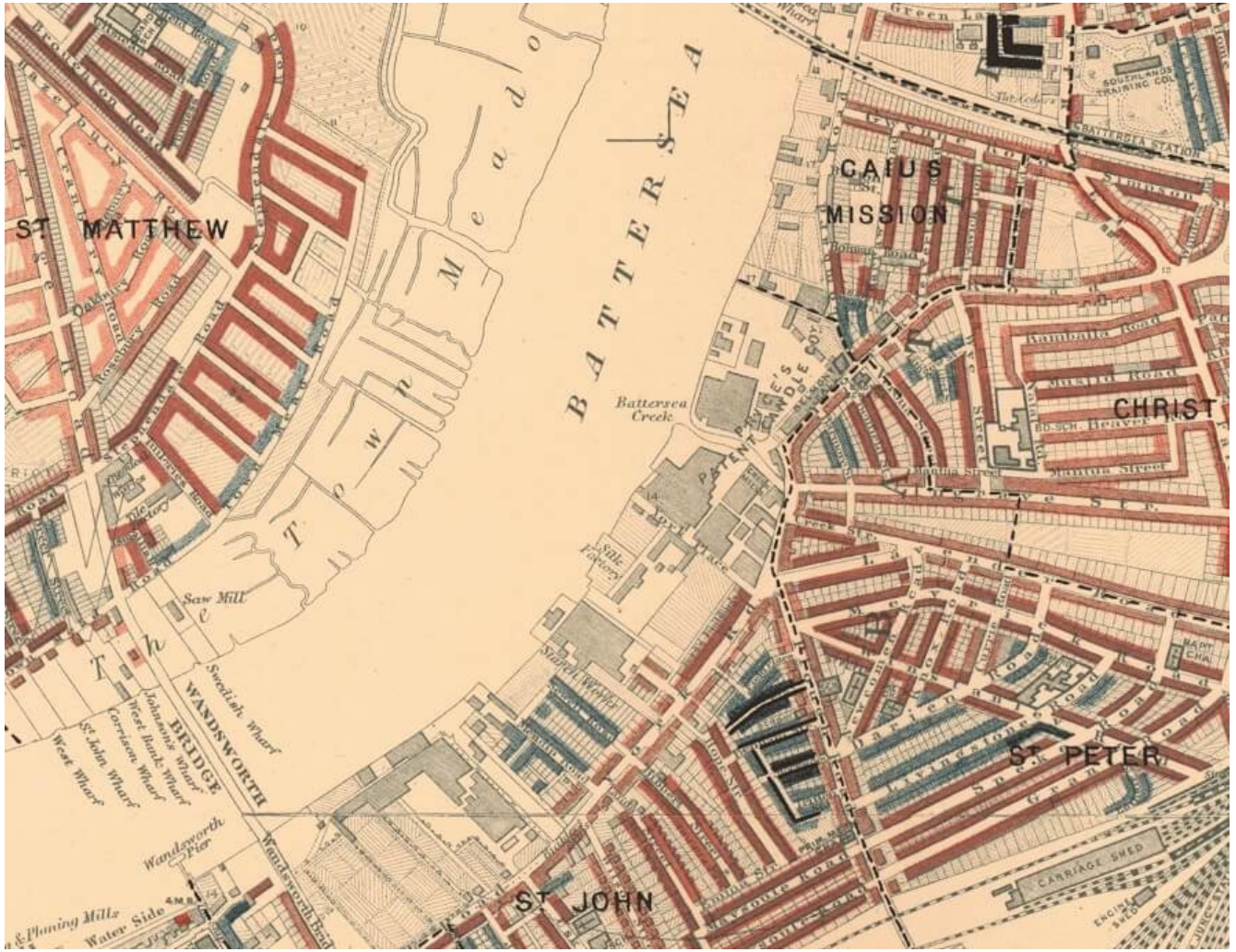
Geodemographics

- Analysis of people (*demographics*) by where they live (*geo*).
- Used to identify similar neighbourhoods or administrative areas.
- Means of multivariate data reduction for the differentiation of areas.
- Been around for many many years, dating back to the late 1800s.

Charles Booth I

- Created the first geodemographic-style classification.
- Shipping business owner and philanthropist.
- Survey: *Life and Labour of People in London.*
- Mostly qualitative analysis by walking through areas.
- Books and books and books with notes.
- He noticed that there is a geographical pattern in the distribution of different social categories: people who live in a particular neighbourhood and share similar living conditions, are of similar characteristics and social status

Charles Booth II



Maps Descriptive of London Poverty 1889. Charles Booth's *Inquiry into Life and Labour in London (1886-1903).*

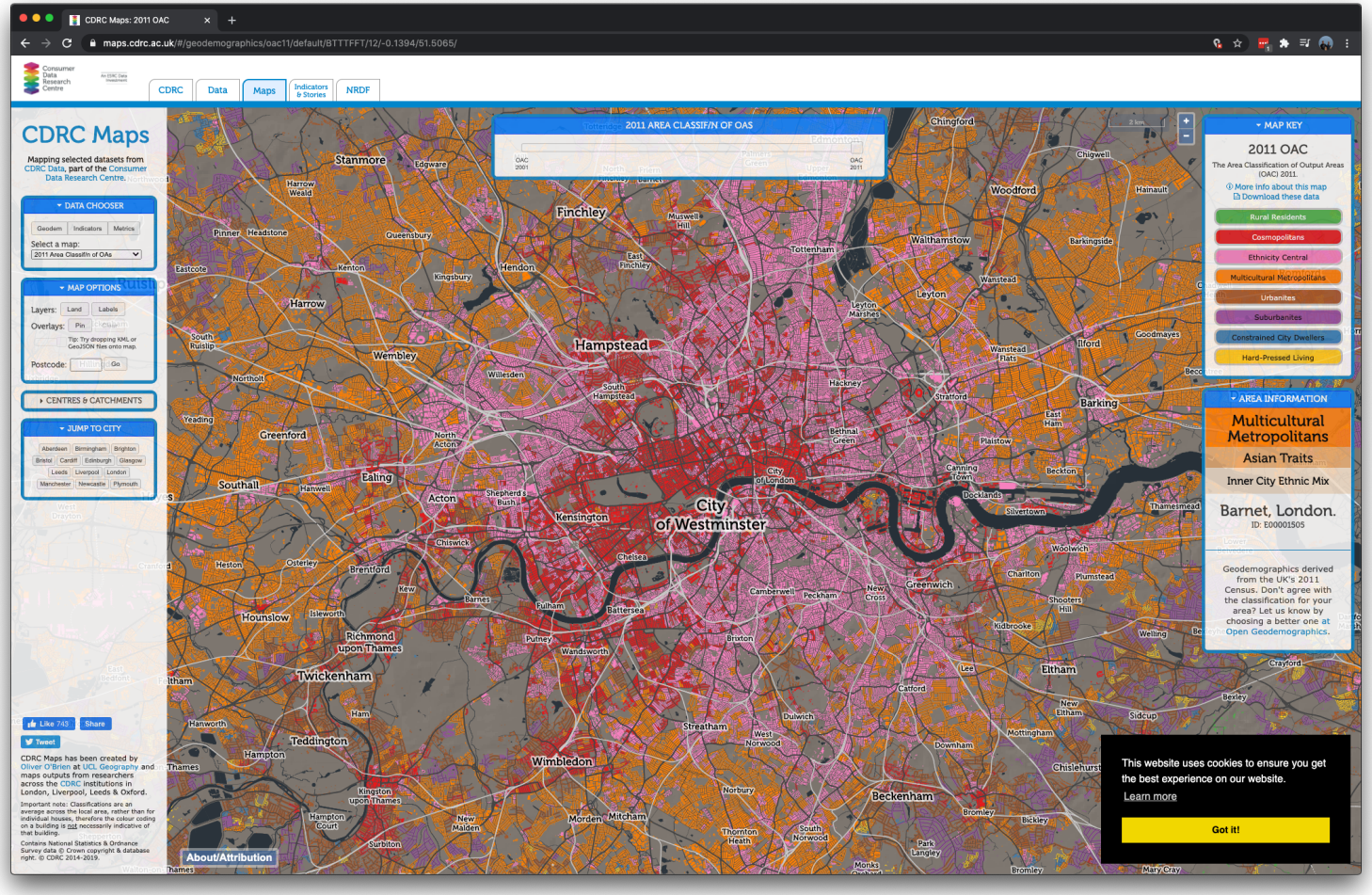
Charles Booth III

Classification	Colour	
Lowest class. Vicious, semi-criminal.	Black	
Very poor, casual. Chronic want.	Dark blue	
Poor. 18s. to 21s. a week for a moderate family.	Light blue	
Mixed. Some comfortable others poor.	Purple	
Fairly comfortable. Good ordinary earnings.	Pink	
Middle class. Well-to-do.	Red	
Upper-middle and upper classes. Wealthy.	Yellow	

Geodemographics I

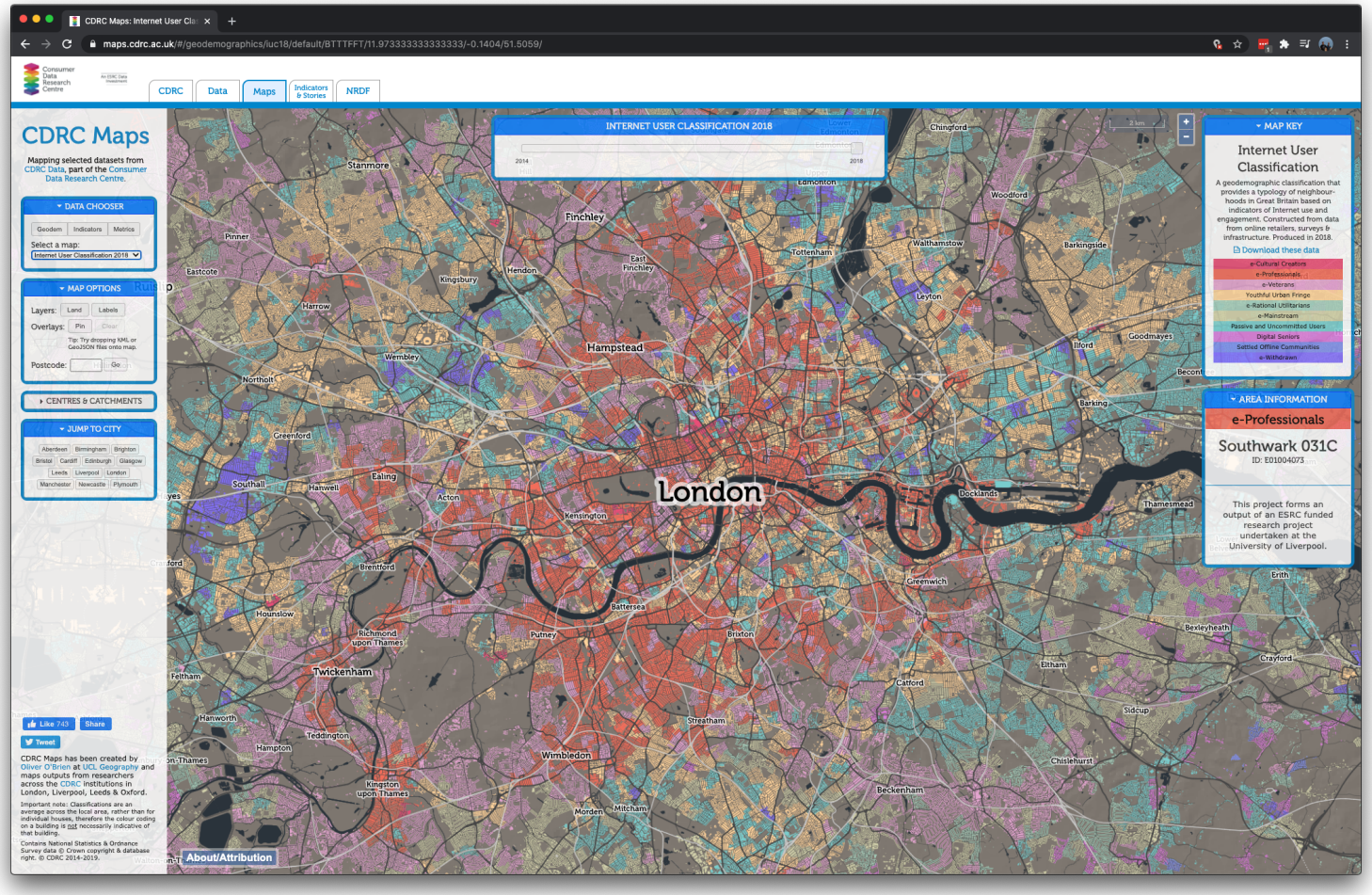
- Further developed in 1970s to target urban deprivation funding.
- Commercial sector also got involved (CACI ACORN / Experia MOSAIC).
- Office for National Statistics' Output Area Classification (2001, 2011, 2021).
- ONS' Output Area Classification completely open using Census data.
- All classification have a similar structure, typically hierarchical.

Geodemographics II



2011 OAC on maps.cdrc.ac.uk

Geodemographics III



Internet User Classification on maps.cdrc.ac.uk

Geodemographics IV

Singleton *et al.* 2020:

“A geodemographic classification is created by assembling a wide range of measures that describe the characteristics of areas and/or those people living within them, and then, through the implementation of unsupervised learning (clustering), identifies groups of areas that share common characteristics. Emerging clusters may be divided or aggregated to create a hierarchy, and it is typical that these be accompanied by labels, descriptions, photographs, diagrams and graphs.”

Internet user classification I

Singleton *et al.* 2020:

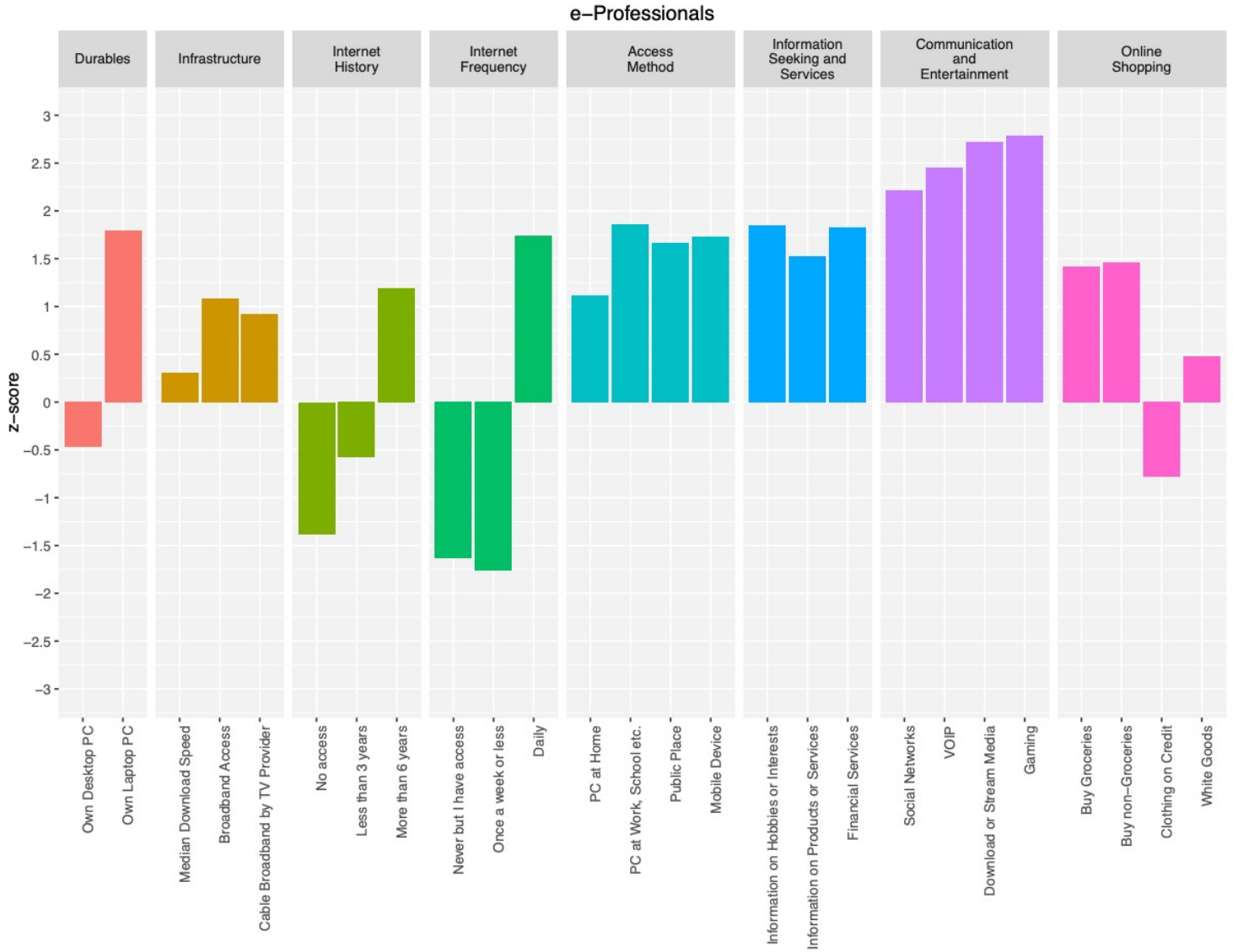
- bespoke classification created by CDRC researchers
- how do populations interact with the internet
- ‘profiles of internet use and engagement’
- built from a range of consumer data, survey data, and open data
- classification is available as open data available

Internet user classification II

Several noteworthy variables:

- British Population Survey: internet access, frequency of internet usage, access to PC, type of internet use
- transactional (consumer) data on online shopping
- average broadband speed
- census variables such as age, ethnicity
- National Statistics Socio-economic classification (NS-SEC)

Internet user classification III



Internet User Classification mean attributes of the *e-Professionals*

Internet user classification IV

e-Professionals:

“This Group has high levels of Internet engagement, particularly regarding social networks, communication, streaming and gaming, but relatively low levels of online shopping, besides groceries. They are new but very active users, with a very high proportion of the population engaging on a daily basis. (…) Geographically, this Group is mainly located close to the city centre or within the proximity of Higher Education Institutes, where infrastructure accessibility, such as cable broadband, is sufficient”

Internet user classification V

- Measures of access to and use of internet.
- Identification of areas to target potential interventions.
- Analysis of areas where people are likely to work from home.

Limitations

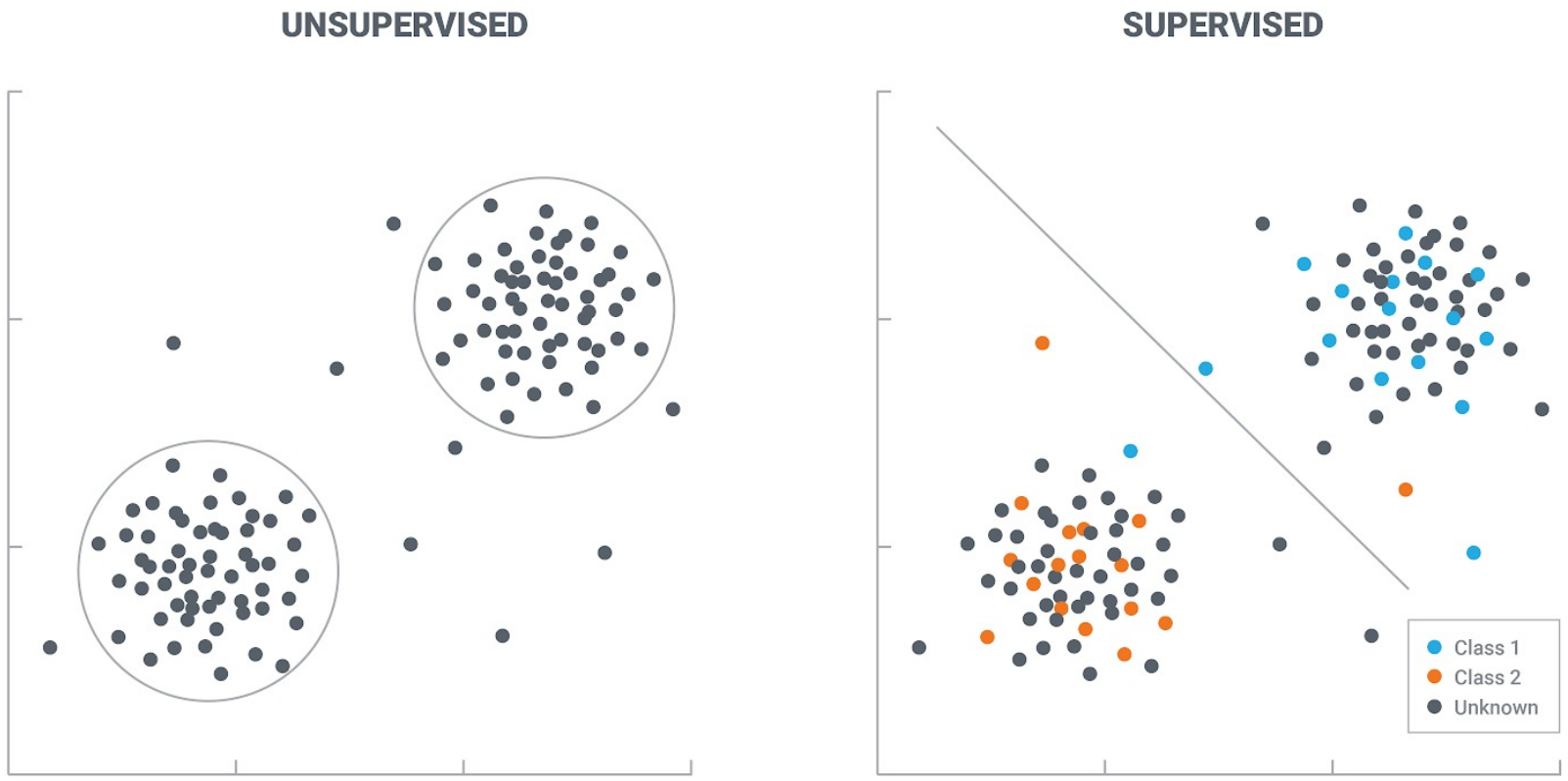
- Highly dependent on the input data (complete data necessary!).
- Input data can get old very quickly (depending on the topic).
- Inherent biases within the input data – see also Data, Politics and Society article on Geodemographics by Dalton and Thatcher (2015).

Further applications

Using the geodemographic classification as input for further analysis:

- Harris *et al.* 2007: differences in school choice between social groups
- Brundson *et al.* 2011: participation in higher education
- Martin *et al.* 2018: analysis of travel-to-work flows
- Goodman *et al.* 2011: socio-economic inequalities in exposure to air pollution

Unsupervised versus supervised



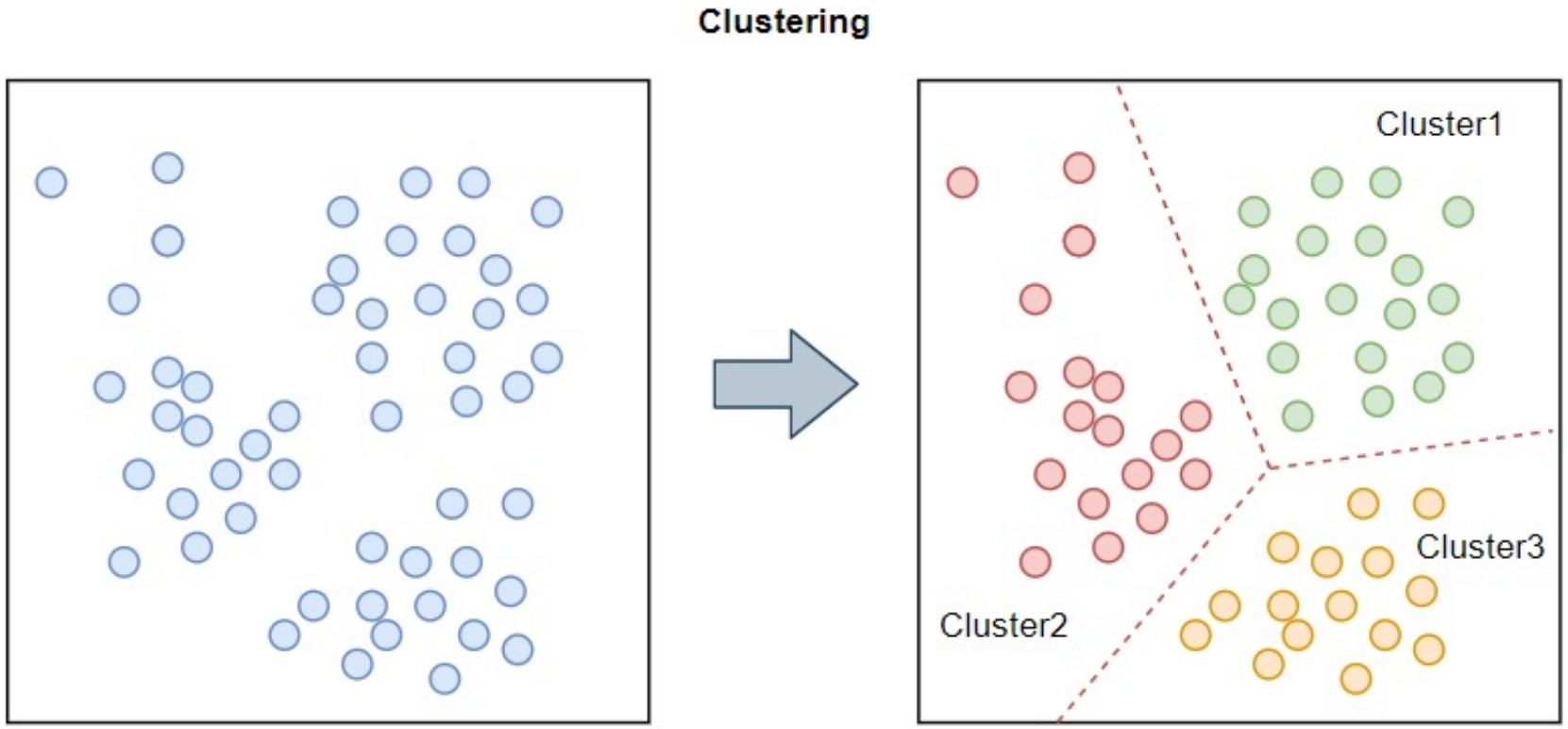
k-means I

- Assign geographic areas with common underlying attributes to similar classification groups.
- Used in: Internet User Classification, ONS' Output Area Classification.

k-means II

- `k` clusters (pre-defined) of `n` individual observations.
- Each observation can have any number of attribute data.
- Choice of data is a balancing act: theory, available data, statistical considerations.

k-means IIII



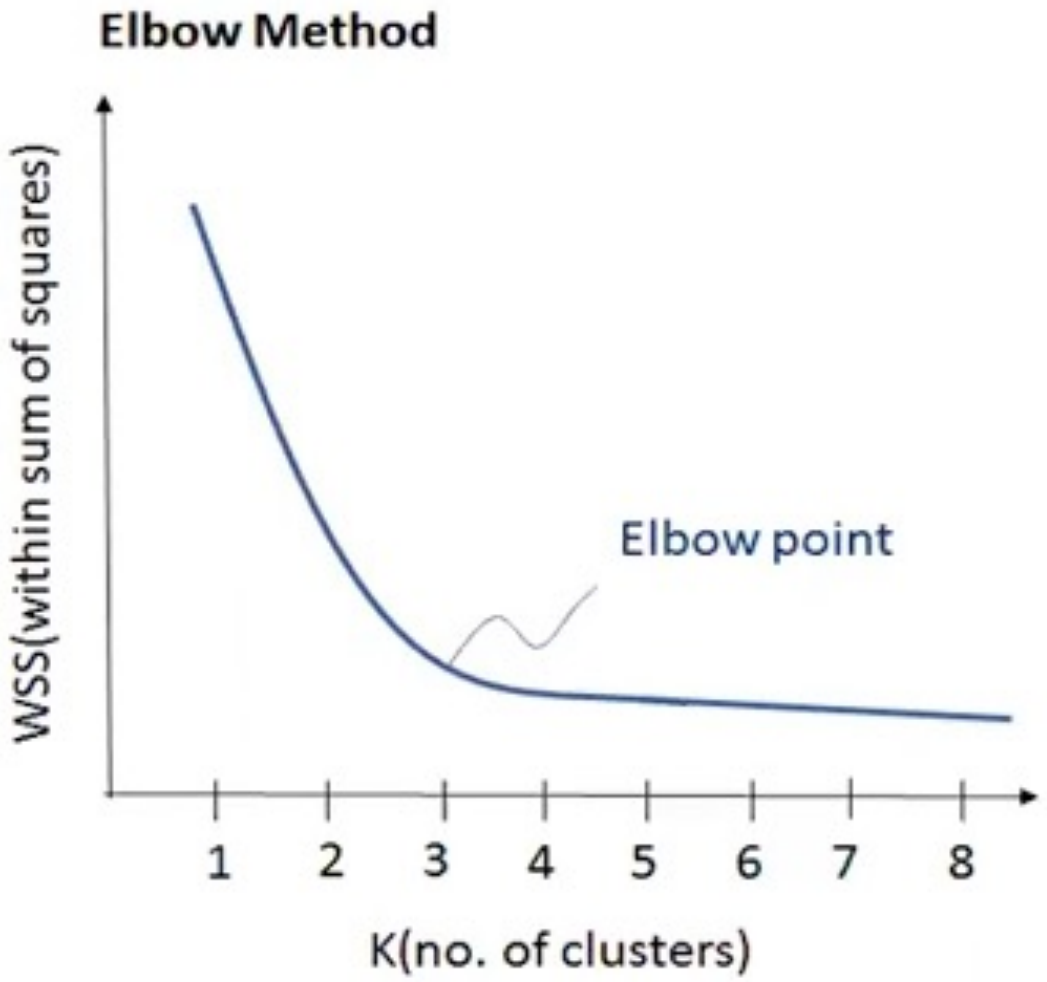
k-means III

- Minimising the distance between an observations input variables to the means of the respective cluster groups.
- Maximising the distance between cluster groups.
- Number of clusters defined a priori.

Number of clusters I

- Too few: too much variation within the groups.
- Too many: overfitting and splitting similar observations.
- Iterate through the model multiple times and try to minimise variation within cluster groups.

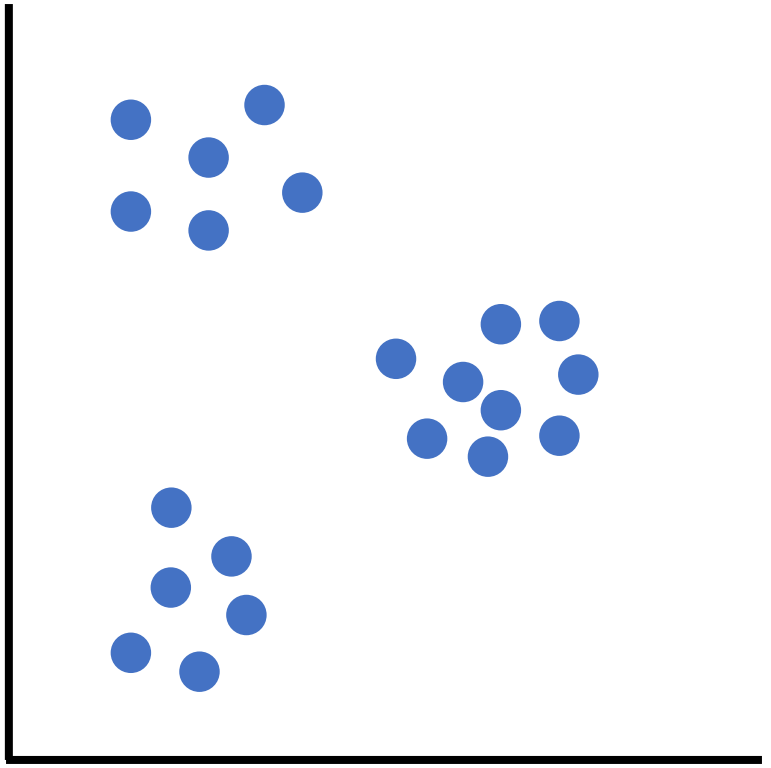
Number of clusters II



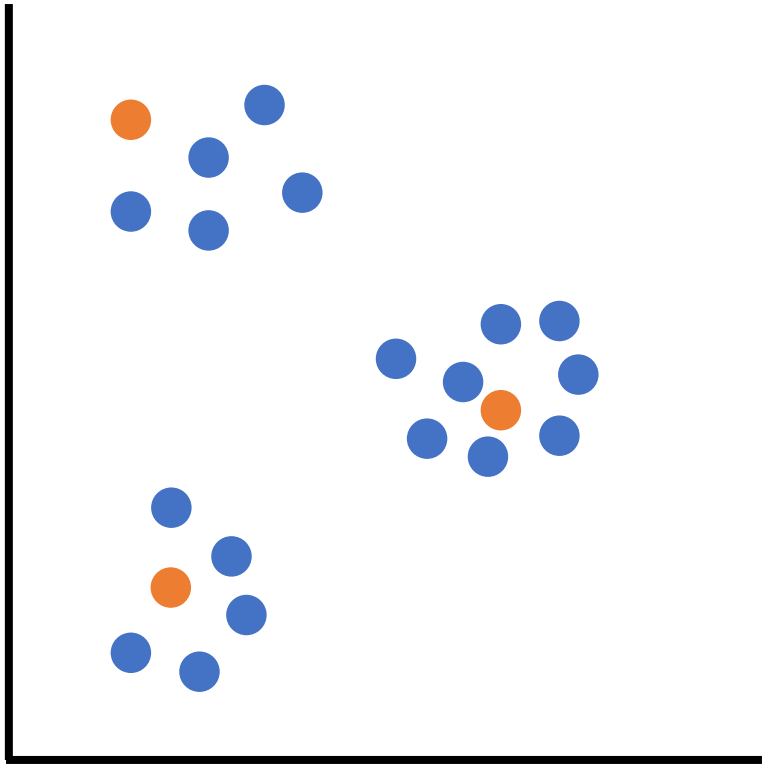
Process

- Step 1: identify your `k`
- Step 2: randomly identify `k` distinct data points as initial cluster centre
- Step 3: assign each observations to the nearest cluster
- Step 4: calculate the mean of each cluster
- Step 5: repeat with mean value becoming new cluster centre until no change

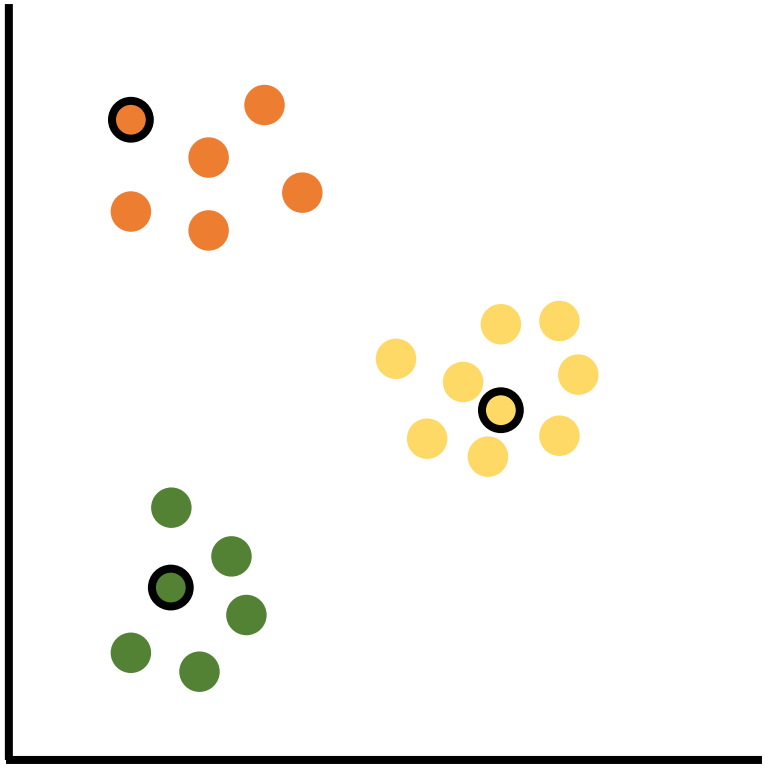
Step 1



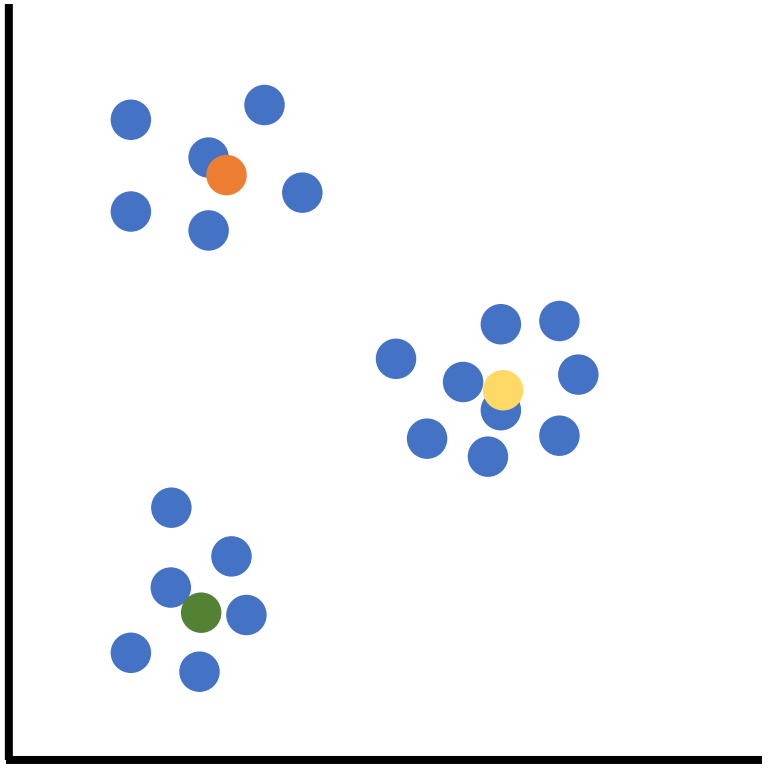
Step 2



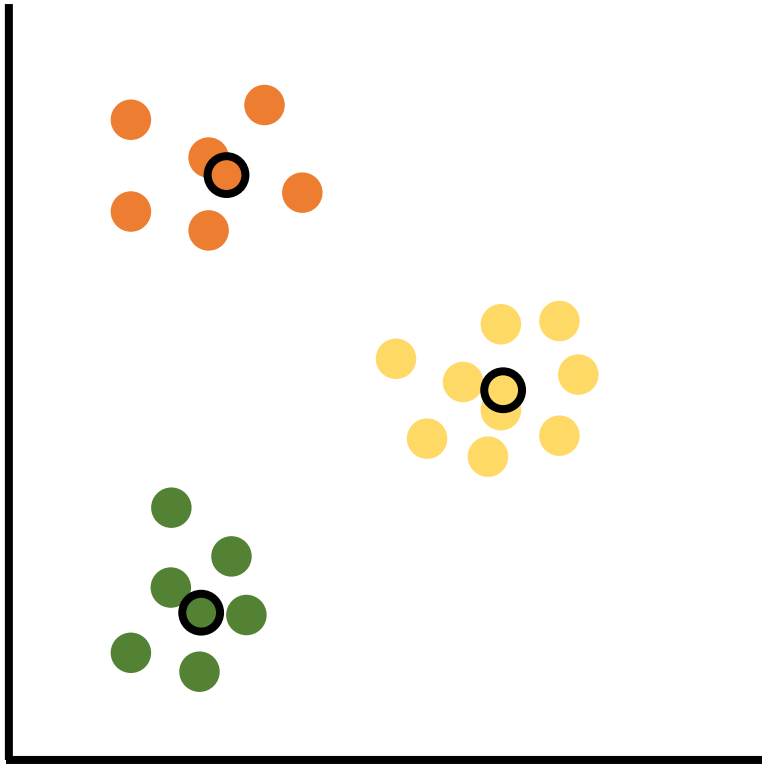
Step 3



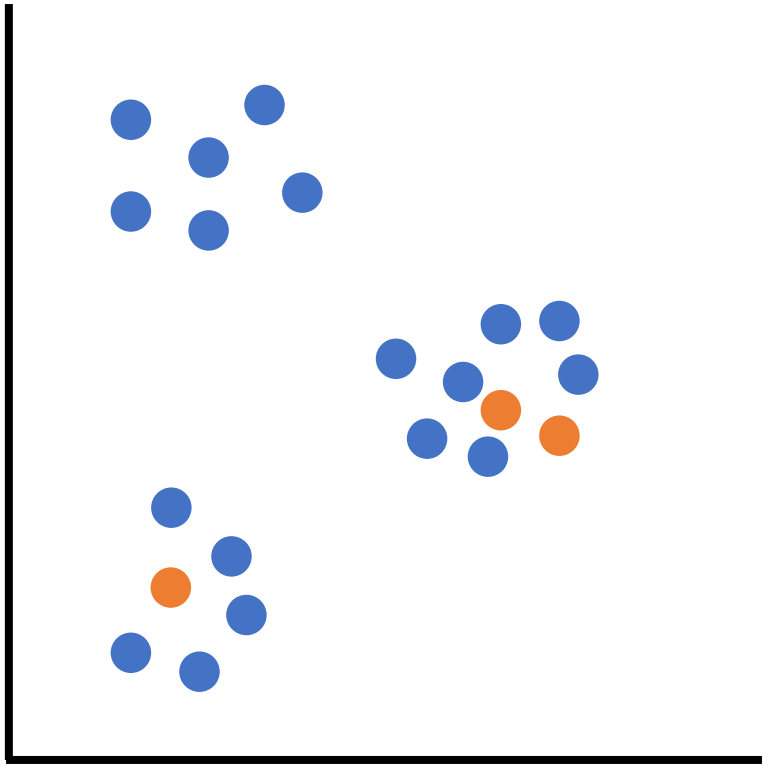
Step 4



Step 5



Multiple iterations



Interpretation of clusters

- Look at the means of the input data for each cluster (‘signature’)
- Based on the underlying (mean) signature pen portraits can be developed
- Not spatial in nature (different to the DBSCAN)
- Important to consider collinearity issues (PCA?)

Conclusion

- Geodemographics as the analysis of people by where they live.
- Typically, a form of unsupervised machine learning is used; k-means.
- Profiling of areas has several research applications.


BONUS
BONUS
BONUS
BONUS

Automation

99% of what you have done / will do is using existing functions

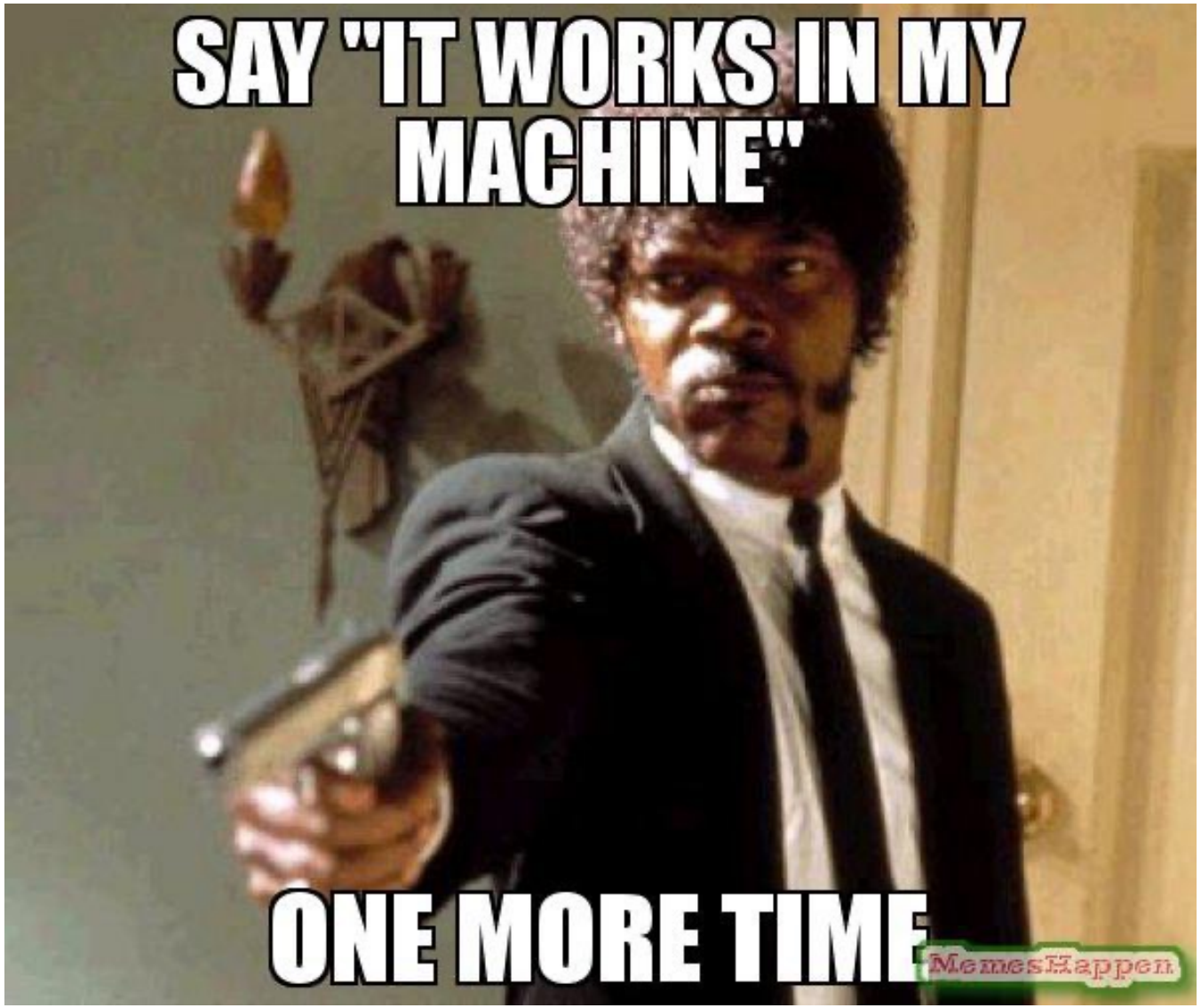
… but you can combine these into functions and use your own arguments and parameters.

… but if you use some workflows repeatedly and across projects you could think of create a library or package?

No matter what:

R is optimised in certain ways: design of your code is very important.


My code doesn't always work
But when it does, it works on my machine
quickmeme.com


SAY "IT WORKS IN MY MACHINE"
ONE MORE TIME
MemesHappen

```r
dayloop2 <- function(temp){
    for (i in 1:nrow(temp)){
        temp[i,10] <- i
        if (i > 1) {
            if ((temp[i,6] == temp[i-1,6]) & (temp[i,3] == temp[i-1,3])) {
                temp[i,10] <- temp[i,9] + temp[i-1,10]
            } else {
                temp[i,10] <- temp[i,9]
            }
        } else {
            temp[i,10] <- temp[i,9]
        }
    }
    names(temp)[names(temp) == "V10"] <- "Kumm."
    return(temp)
}
```

https://stackoverflow.com/questions/2908822/speed-up-the-loop-operation-in-r

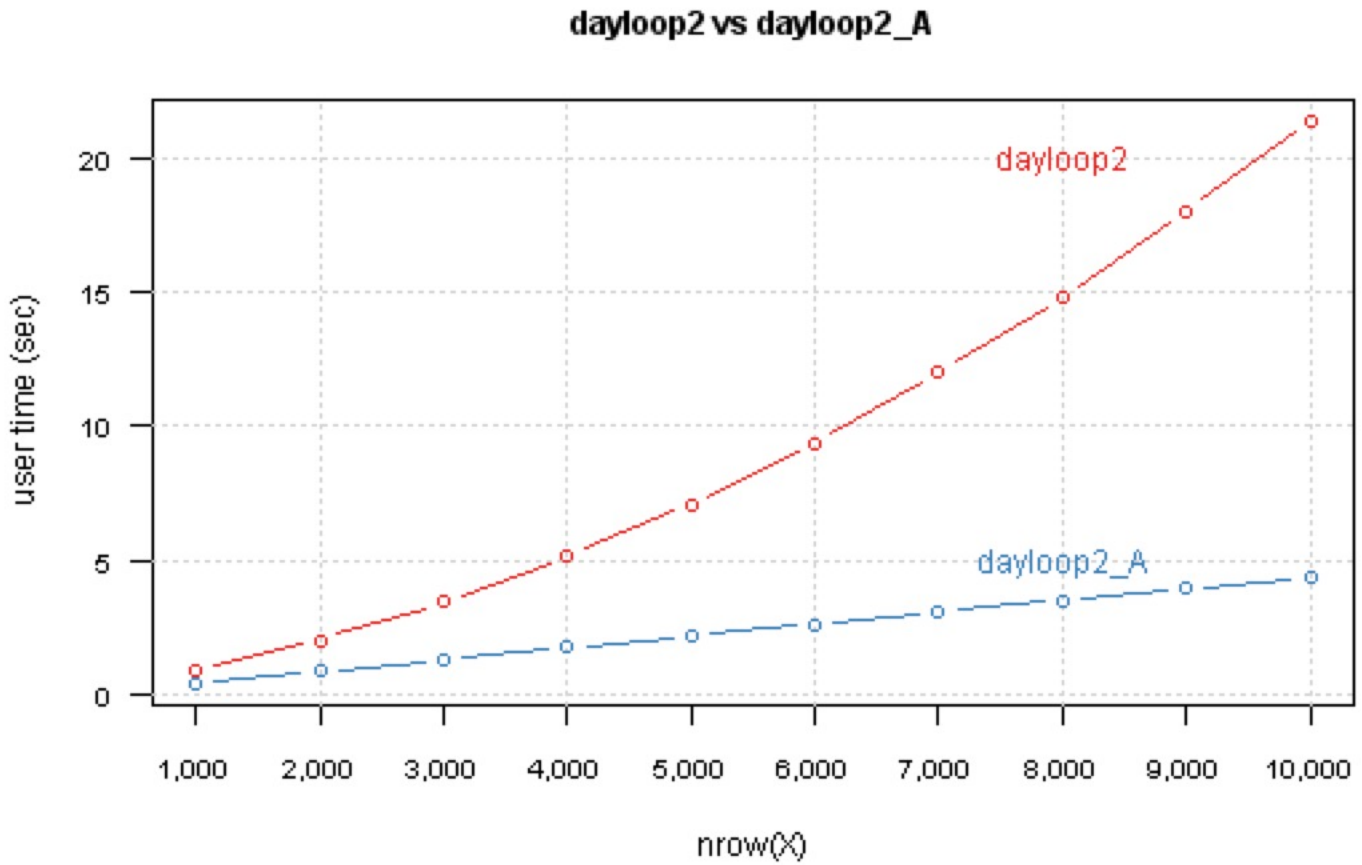
~ 850k rows

```
dayloop2 <- function(temp){
    for (i in 1:nrow(temp)){
        temp[i,10] <- i
        if (i > 1) {
            if ((temp[i,2] == temp[i-1,2]) & (temp[i,3] == temp[i-1,3])) {
                temp[i,10] <- temp[i,9] + temp[i-1,10]
            } else {
                temp[i,10] <- temp[i,9]
            }
        } else
            temp[i,10] <- temp[i,9]
        }
    }
    names(temp)[names(temp) == "V10"] <- "Kumm."
    return(temp)
}
```

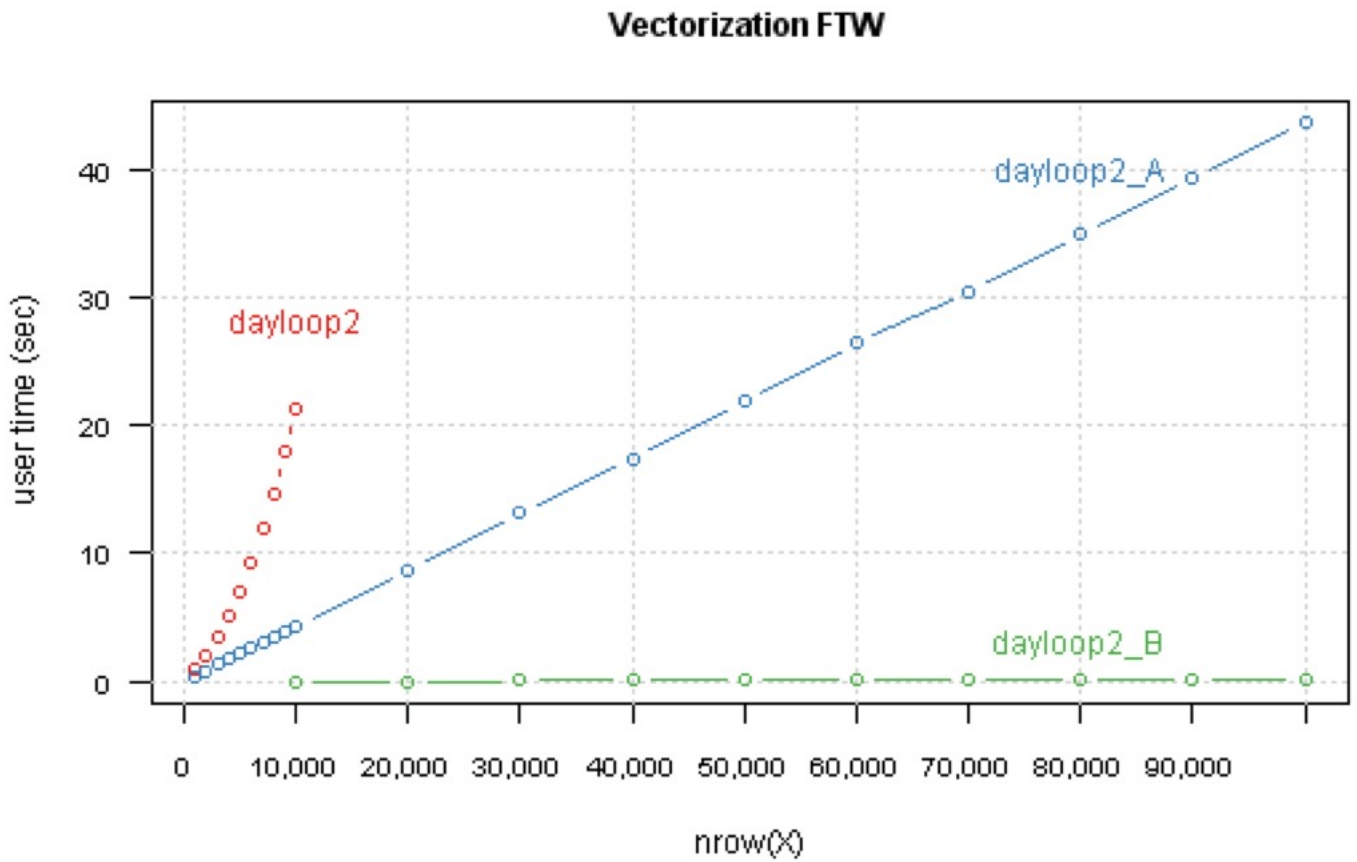
> 10 hours

https://stackoverflow.com/questions/2908822/speed-up-the-loop-operation-in-r

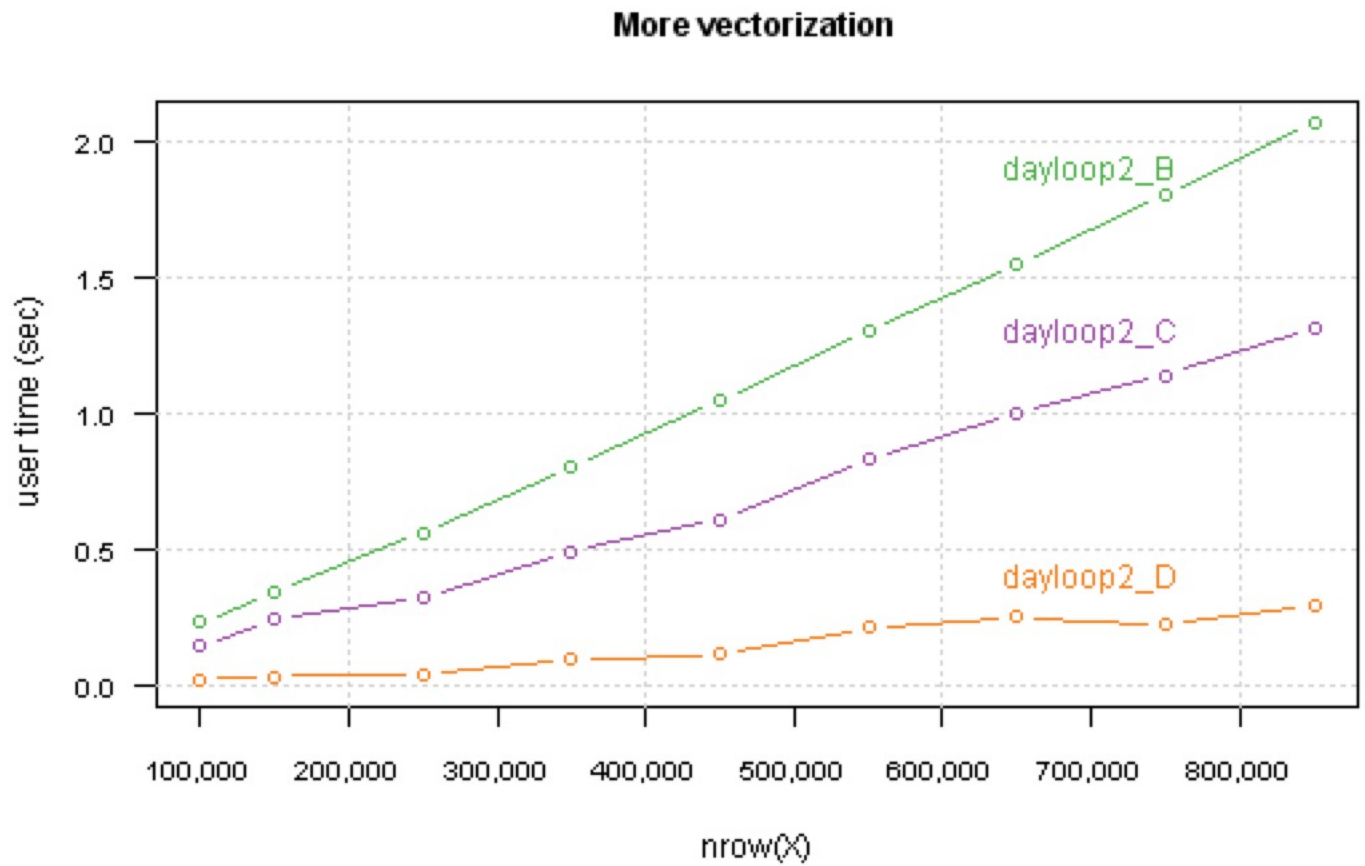
~ 850k rows



reduce indexing



vectorisation



more vectorisation

Speeding up

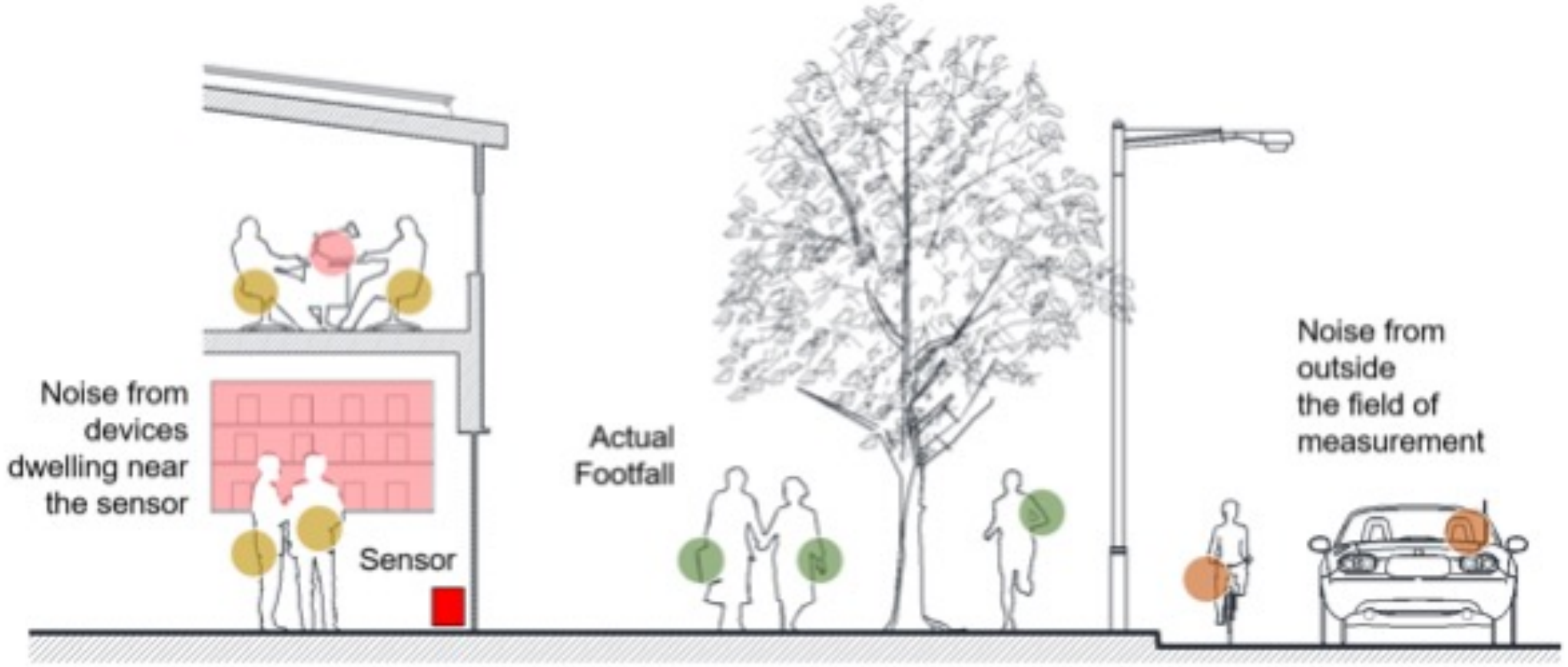
First things to check:

- Store the results and access them rather than repeatedly re-calculating
- Take non-loop-dependent calculations out of loops
- Avoid necessary calculations (e.g. `regex` or fixed search?)

Not always enough, more advanced strategies:

- Parallel processing
- GPU-accelerated analytics
- Different tools?

Smart street sensor project CDRC I



Smart street sensor project CDRC II

Start of the project:

- 40 locations equipped with sensors
- 1 million measurements / day
- 100 MB / day
- Download from servers: 20 minutes
- Processing: 30 minutes

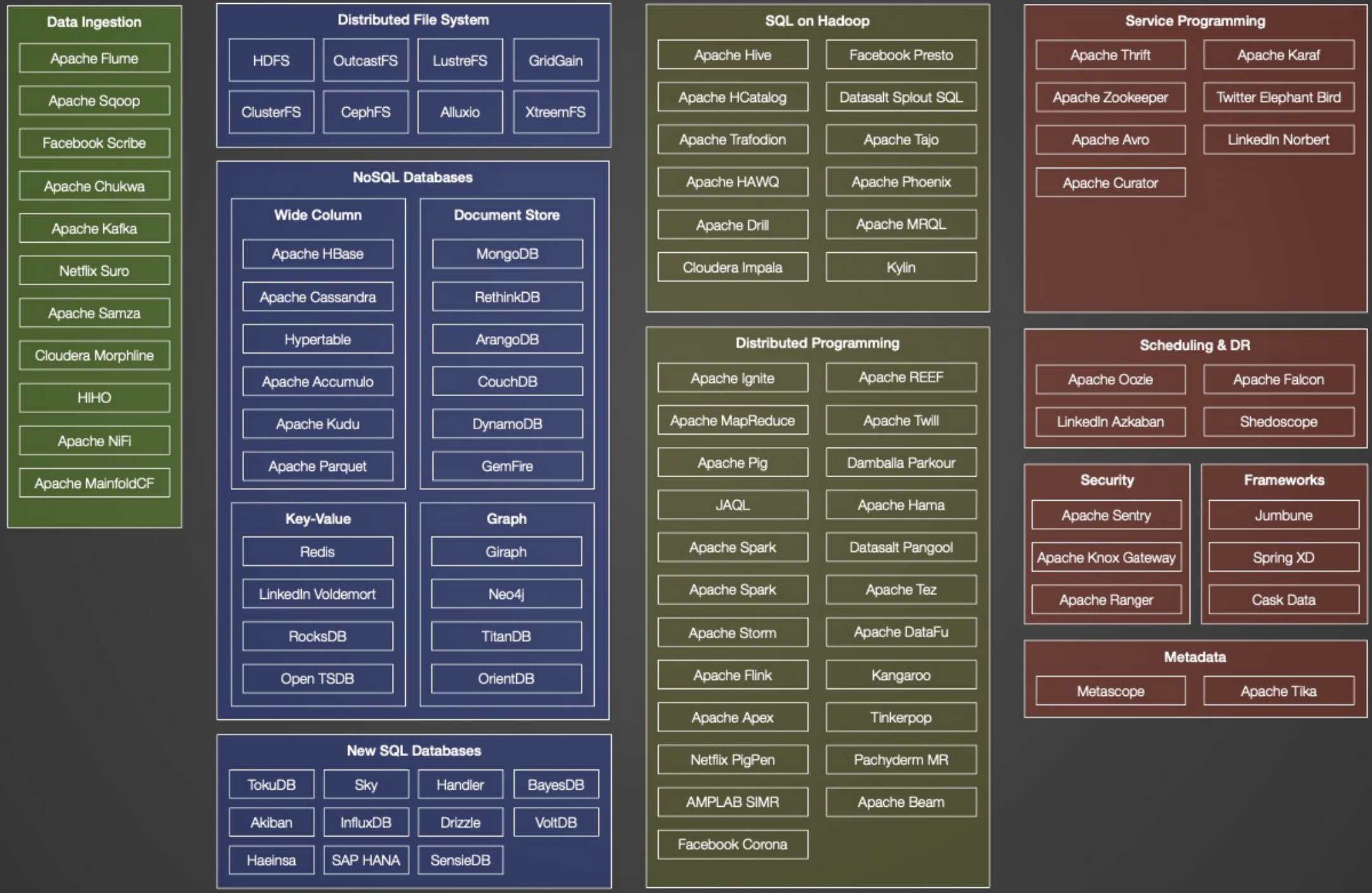
Smart street sensor project CDRC III

Growth of the project:

- 675 locations equipped with sensors
- 31 million measurements / day
- 2 GB / day
- Download from servers: 2 hours
- Processing: 5 hours

Smart street sensor project CDRC IV

- Solution: set-up a Big Data infrastructure?
- Can be very confusing, costly


Data Ingestion
Apache Flume
Apache Sqoop
Facebook Scribe
Apache Chukwa
Apache Kafka
Netflix Suro
Apache Samza
Cloudera Morphline
HIHO
Apache NiFi
Apache MainfoldCF
Distributed File System
HDFS
OutcastFS
LustreFS
GridGain
ClusterFS
CephFS
Alluxio
XtreemFS
NoSQL Databases
Wide Column
Apache HBase
Apache Cassandra
Hypertable
Apache Accumulo
Apache Kudu
Apache Parquet
Document Store
MongoDB
RethinkDB
ArangoDB
CouchDB
DynamoDB
GemFire
Key-Value
Redis
LinkedIn Voldemort
RocksDB
Open TSDB
Graph
Giraph
Neo4j
TitanDB
OrientDB
New SQL Databases
TokuDB
Sky
Handler
BayesDB
Akiban
InfluxDB
Drizzle
VoltDB
Haeinsa
SAP HANA
SensieDB
SQL on Hadoop
Apache Hive
Facebook Presto
Apache HCatalog
Datasalt Splout SQL
Apache Trafodion
Apache Tajo
Apache HAWQ
Apache Phoenix
Apache Drill
Apache MRQL
Cloudera Impala
Kylin
Distributed Programming
Apache Ignite
Apache REEF
Apache MapReduce
Apache Twill
Apache Pig
Damballa Parkour
JAQL
Apache Hama
Apache Spark
Datasalt Pangool
Apache Spark
Apache Tez
Apache Storm
Apache DataFu
Apache Flink
Kangaroo
Apache Apex
Tinkerpop
Netflix PigPen
Pachyderm MR
AMPLAB SIMR
Apache Beam
Facebook Corona
Service Programming
Apache Thrift
Apache Karaf
Apache Zookeeper
Twitter Elephant Bird
Apache Avro
LinkedIn Norbert
Apache Curator
Scheduling & DR
Apache Oozie
Apache Falcon
LinkedIn Azkaban
Shedoscope
Security
Apache Sentry
Apache Knox Gateway
Apache Ranger
Frameworks
Jumbune
Spring XD
Cask Data
Metadata
Metascope
Apache Tika

Medium data toolkit

Soundararaj 2019:

- Unix philosophy: pipes and text streams
- Command-line tools: `cat`, `find`, `sort`, `uniq`, `sed`, `grep`, `awk`
- Parallelisation: using `xargs` to utilise all processing cores

Unix philosophy

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

Pipelines compared I

Benchmarked:

- Full Pipeline in R: 20 minutes, 14 seconds
- Full pipeline in Unix tools: 18 seconds
- Full pipeline in Unix tools (parallelised): 3 seconds

2 hours download, 5 hours of processing got done in 15 minutes.

Pipelines compared II

- Lots of data != Big Data
- In some cases: evaluate the data for ‘bigness’ in each dimension (volume, velocity, variety, etc.) and then decide if there are appropriate tools available that do one thing very well.

Questions

Justin van Dijk

j.t.vandijk@ucl.ac.uk