MAKING BIG DATA COME ALIVE

THINKBIG

A TERADATA COMPANY

Self-Organising Maps

Applied R Munich

Dominik Koch, Senior Data Scientist

14.11.2016

About Me

Dominik Koch

Senior Data Scientist, Think Big

After graduating my statistics studies with distinction, I was looking for new challenges in the world of big data. Now I'm working as a Senior Data Scientist at Think Big in Munich.

As a data driven problem solver I have contributed to the success of many different Data Science projects. Through these projects I also acquired extensive knowledge of the automotive industry and in the banking sector. My main focus is on applying algorithms from machine learning and statistics to large data sets.

dominik.koch@thinkbiganalytics.com

https://de.linkedin.com/in/dominik-koch-341a39100

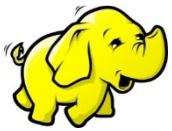
https://www.xing.com/profile/Dominik_Koch20

My Job

THINKBIG
A TERADATA COMPANY

1st Big Data provider 100% focused around open source Apache Hadoop ecosystem and cloud

- Founded in 2010 - industry thought leader
- 130 Employees in International
- 150+ successful projects & 100+ clients
- Vendor-neutral open source integration expertise
- Agile development methodology
- Global delivery model (on-site, near-shore, off-shore)



THINKBIG
A TERADATA COMPANY

Agenda

Theory / Algorithm

Kohonen Package

Live Demo

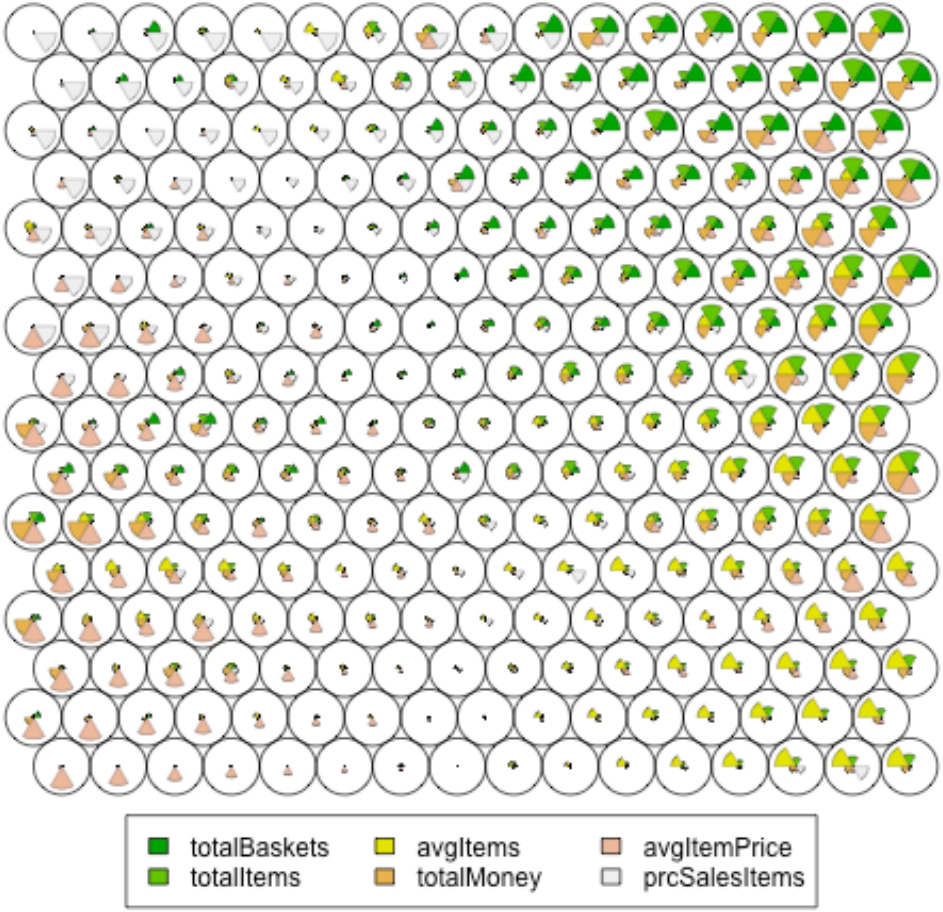
Theory / Algorithm

Self-Organising Maps

- Invented by Teuvo Kohonen (1982)
- Most cited Finnish scientist
- Sometimes called Kohonen map/network
- Unsupervised neural network
- “self organising” because no supervision is required
- Topology preserving map
- 2 dimensional visualisation
- Numerous variants of the basic SOM

Self-Organising Maps

- Visualisation consists of nodes
  - Hexagonal layout
  - Rectangular layout
- Competitive learning
- Dimensionless axes
- Node characteristics
  - Fixed position
  - Weight vector
  - Associated input samples



Algorithm

1. **Initialisation**: all node weights are initialised randomly.
2. **Competition**: Choose a random observation from the training sample and find the "Best Matching Unit" (BMU) by calculating the Euclidean distance to every node in the grid.
3. **Cooperation**: Determine all nodes that lie within the neighbourhood of the BMU. The size of the neighbourhood decreases over time. Exponential decay: $\sigma(t) = \sigma_0 \exp(-t / \lambda)$
4. **Adaption**: Update the nodes within the neighbourhood of the BMU by adjusting the weights towards the chosen observation.

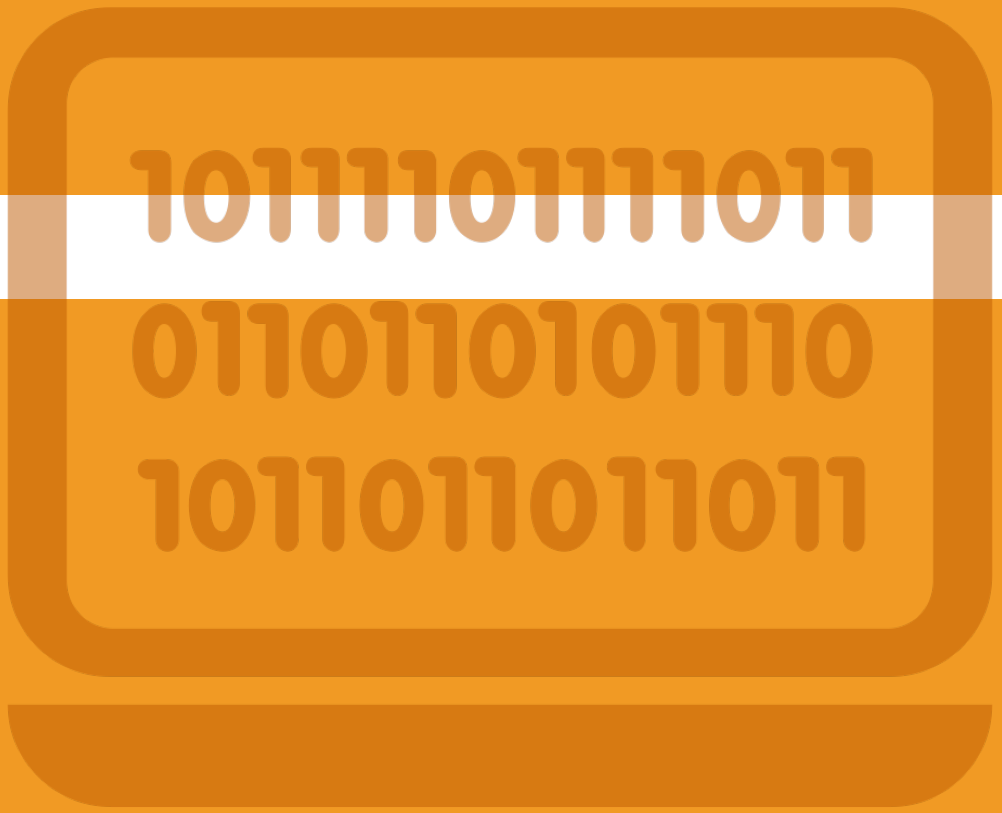
$$W(t+1) = W(t) + \Theta(t) \cdot L(t) \cdot (I(t) - W(t))$$

$$L(t) = L_0 \exp(-t / \lambda)$$

$$\Theta(t) = \exp(-distBMU^2 / (2\sigma^2(t)))$$

5. **Repetition**: Repeat from step 2 till maximum number of iterations reached.

Kohonen Package



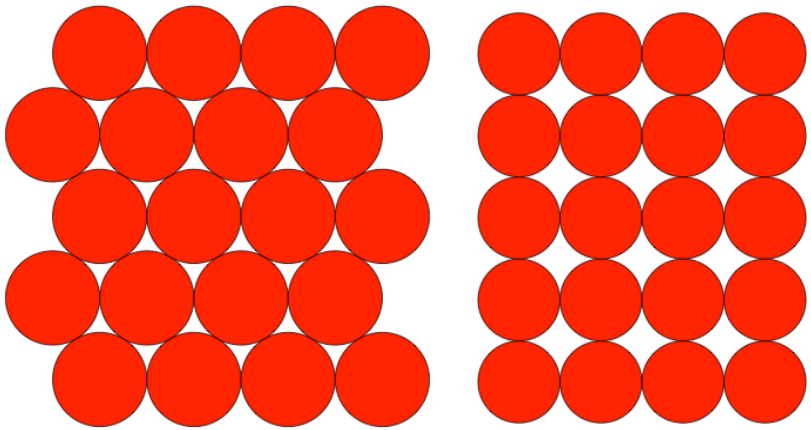
Ta-Feng Grocery Data

- Grocery shopping dataset
  - 817.741 transactions
  - 32.266 customers
  - 23.812 products
  - 4 months (11.2000 – 02.2001)
- Features:
  - Customer demographics
    - Age, residence
  - Product information
    - Amount, price
  - Feature engineering
- Goal: customer segmentation

SOM Grid

```
somgrid(xdim = 16, ydim = 16,
          topo = "hexagonal")
```

- Dimensions of the grid
  - Aim for at least 5-10 observations per node
  - Even number of rows needed if toroidal = TRUE
- Topology:
  - Hexagonal vs Rectangular
  - Influences the number of immediate neighbours
- Toroidal:
  - Edges of the map are joined



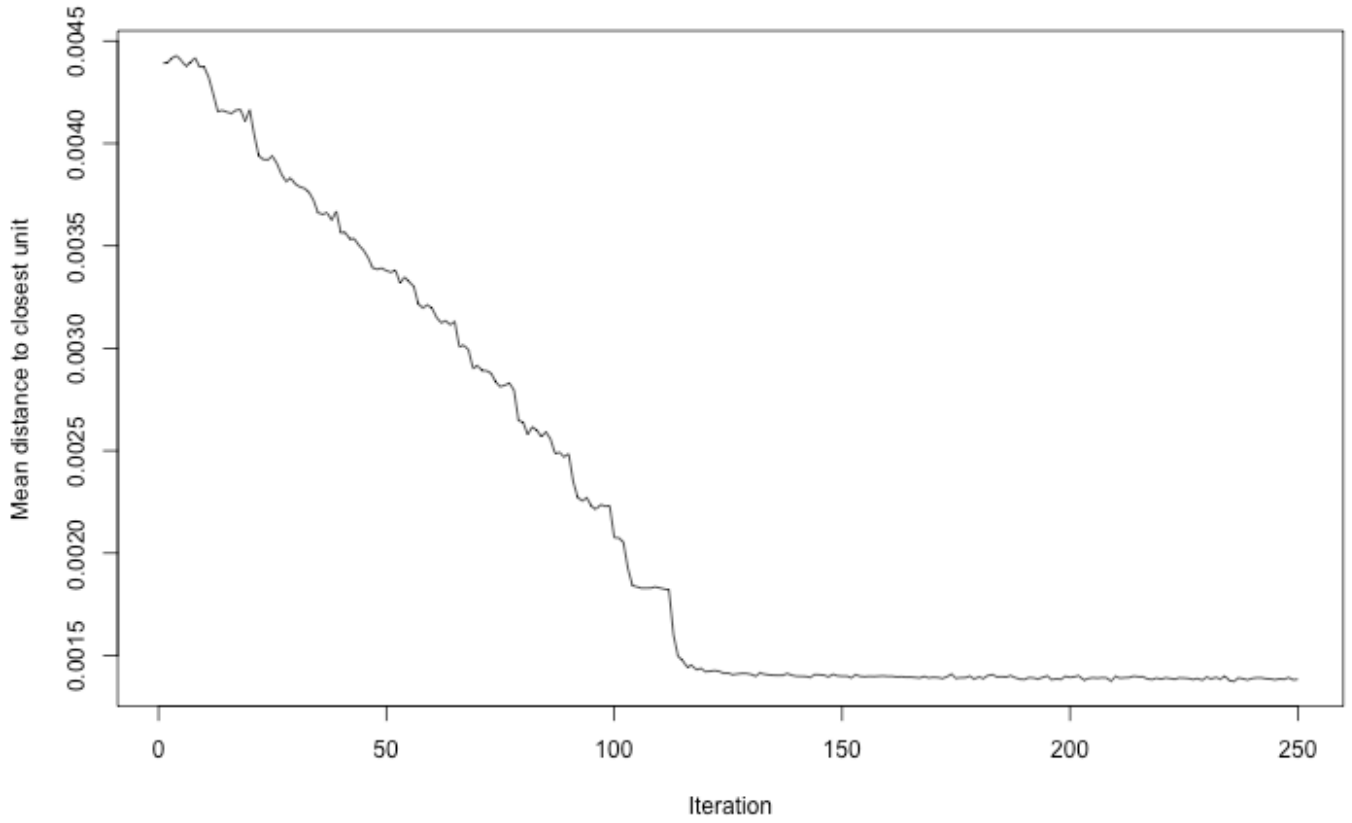
SOM Model

```
taFeng.som <- som(data = as.matrix(scale(taFeng)),
                  grid = taFeng.grid,
                  rlen = 250,
                  alpha = c(0.05, 0.01),
                  toroidal = FALSE,
                  n.hood = "circular")
```

- data: clean, normalised data as a matrix
- grid: previous defined grid of nodes
- rlen: number of times the complete data is presented to the som
- alpha: learning rate
- toroidal: topology of a torus
- n.hood: neighbourhood shape

Training Progress

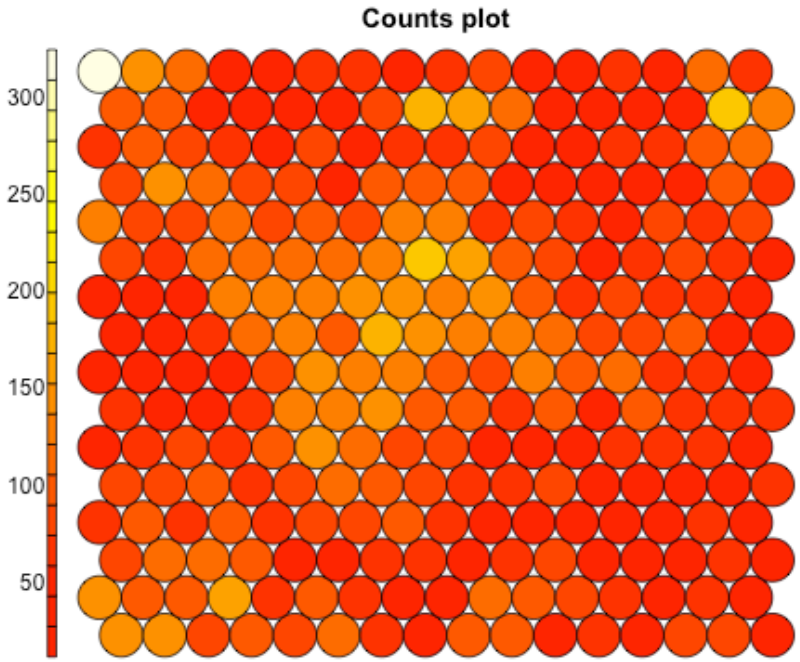
```
plot(taFeng.som, type = "changes")
```



Node counts

```
plot(taFeng.som, type = ”counts")
```

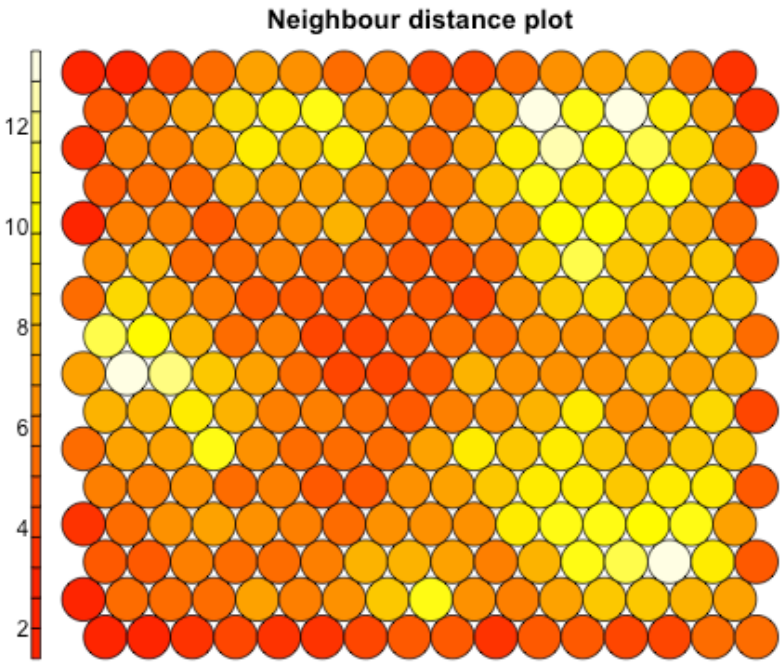
- Number of samples mapped to each nodes
- Empty nodes are depicted in gray
- Aim for at least 5-10 observations per node



Unified distance matrix

```
plot(taFeng.som, type = ” dist.neighbours")
```

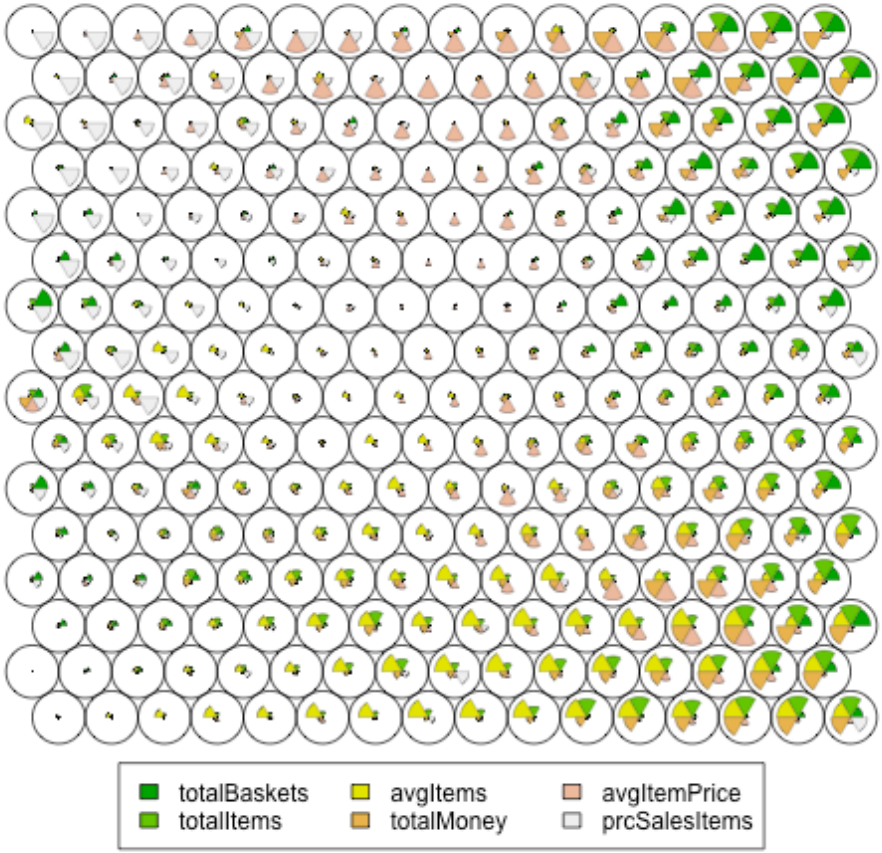
- Sometimes referred to as “U-Matrix”
- Visualises the distance between node weights of neighbouring nodes
- Large distances indicate higher dissimilarities
- Can be used to identify cluster



Fan Diagram

```
plot(taFeng.som, type = "codes")
```

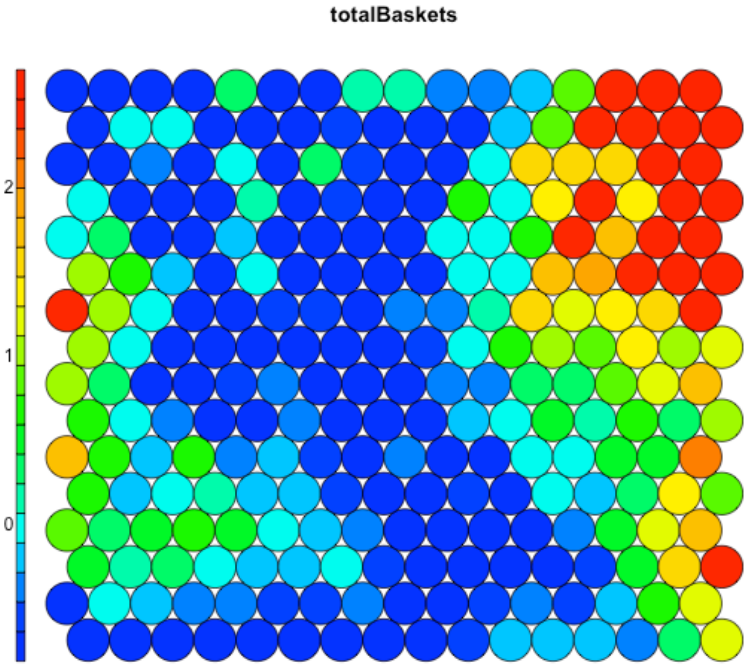
- Visualisation of all the weight vector for every node
- Distribution of all variables in one plot
- Can be used to identify patterns
- Only useful for a small number of variables



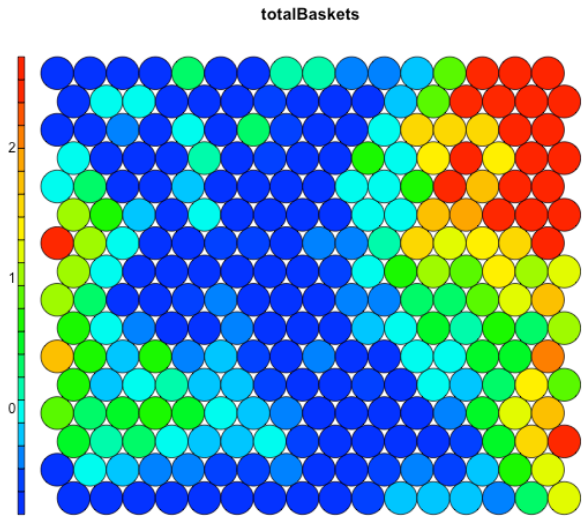
Heatmap

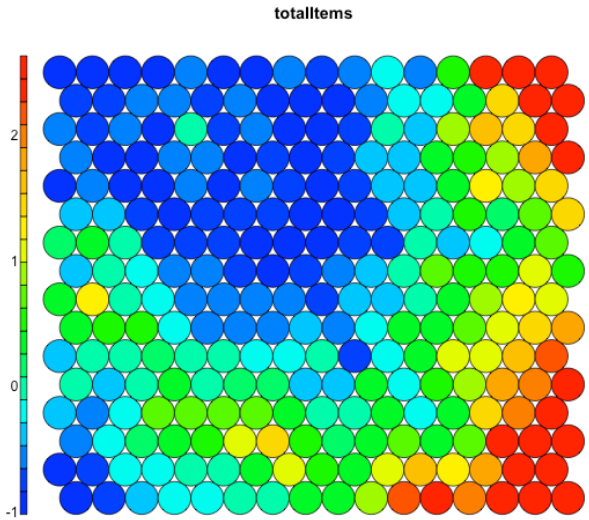
```
plot(taFeng.som, type = "property", property = taFeng.som$codes[,1],
     main = colnames(taFeng.som$codes)[1])
```

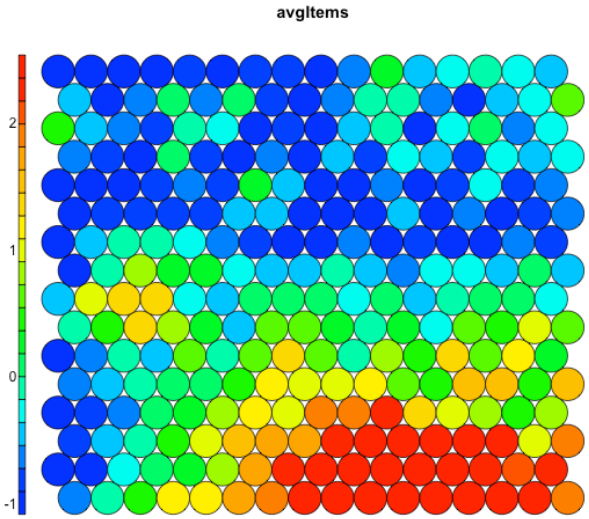
- Discover patterns between variables
- Represents the distribution of one variable across the SOM
- Same insights as fan diagram
- One heatmap for every variable
- Scale is normalised
- Suitable if more than 5 variables

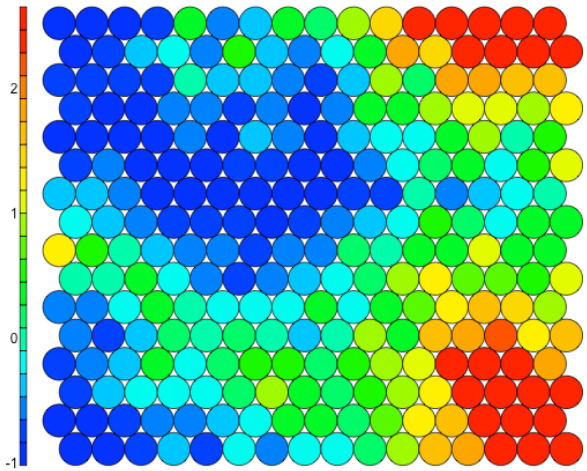


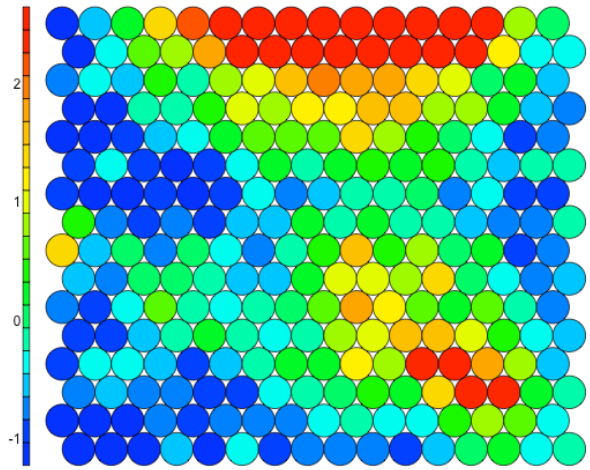
Gallery of Heatmaps

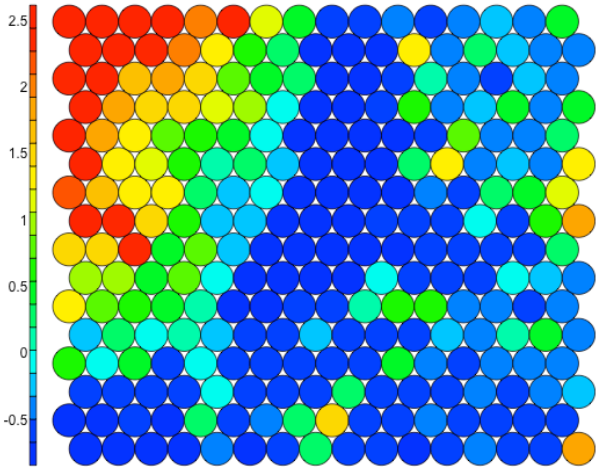






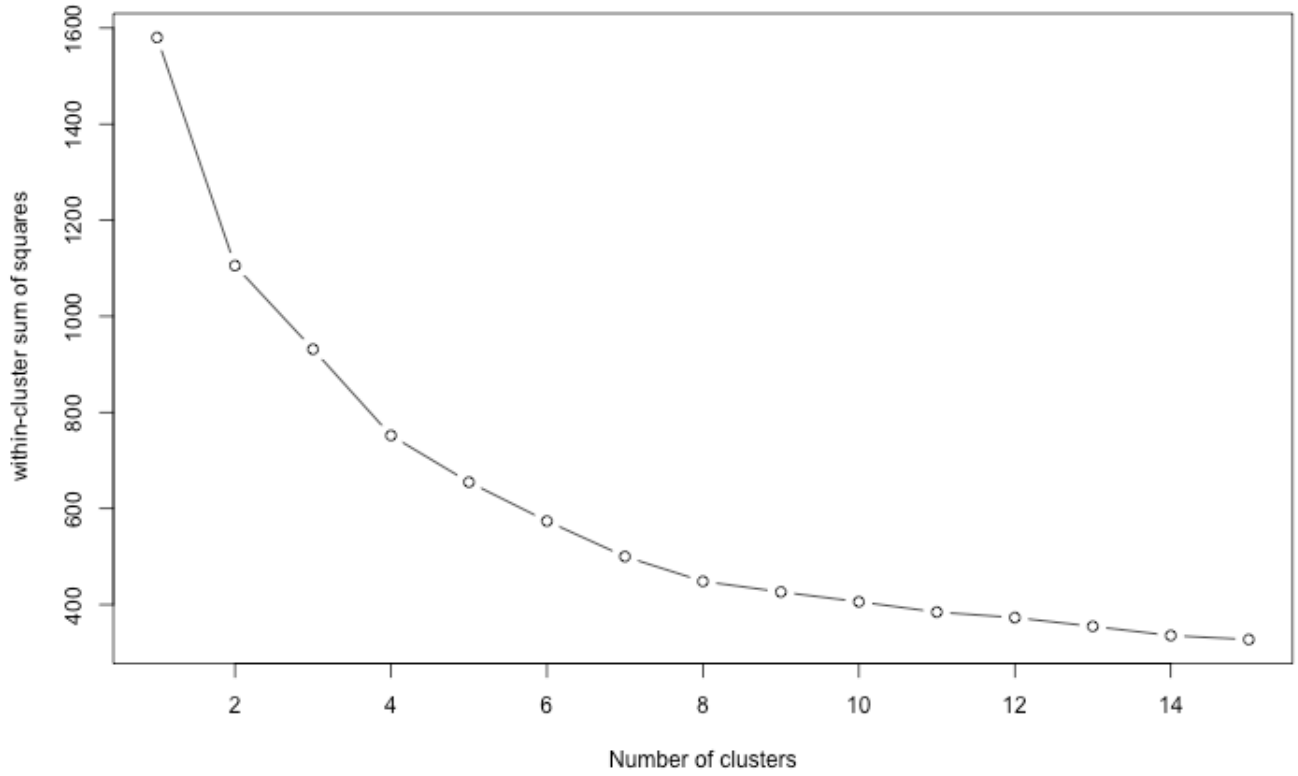






Number of Cluster

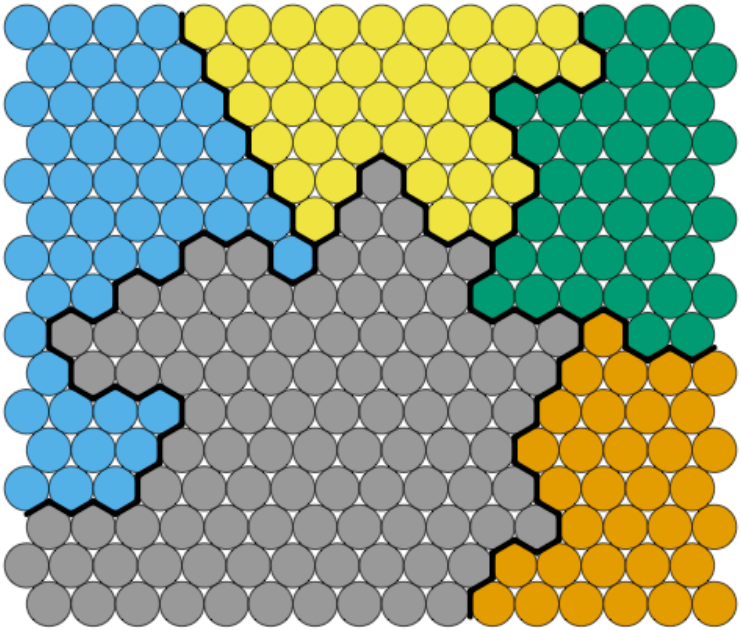
```
wss <- (nrow(taFeng.som$codes)-1)*sum(apply(taFeng.som$codes,2,var))
for (i in 2:15)  wss[i] <- sum(kmeans(taFeng.som$codes, centers=i)$withinss)
plot(1:15,wss, type="b")
```



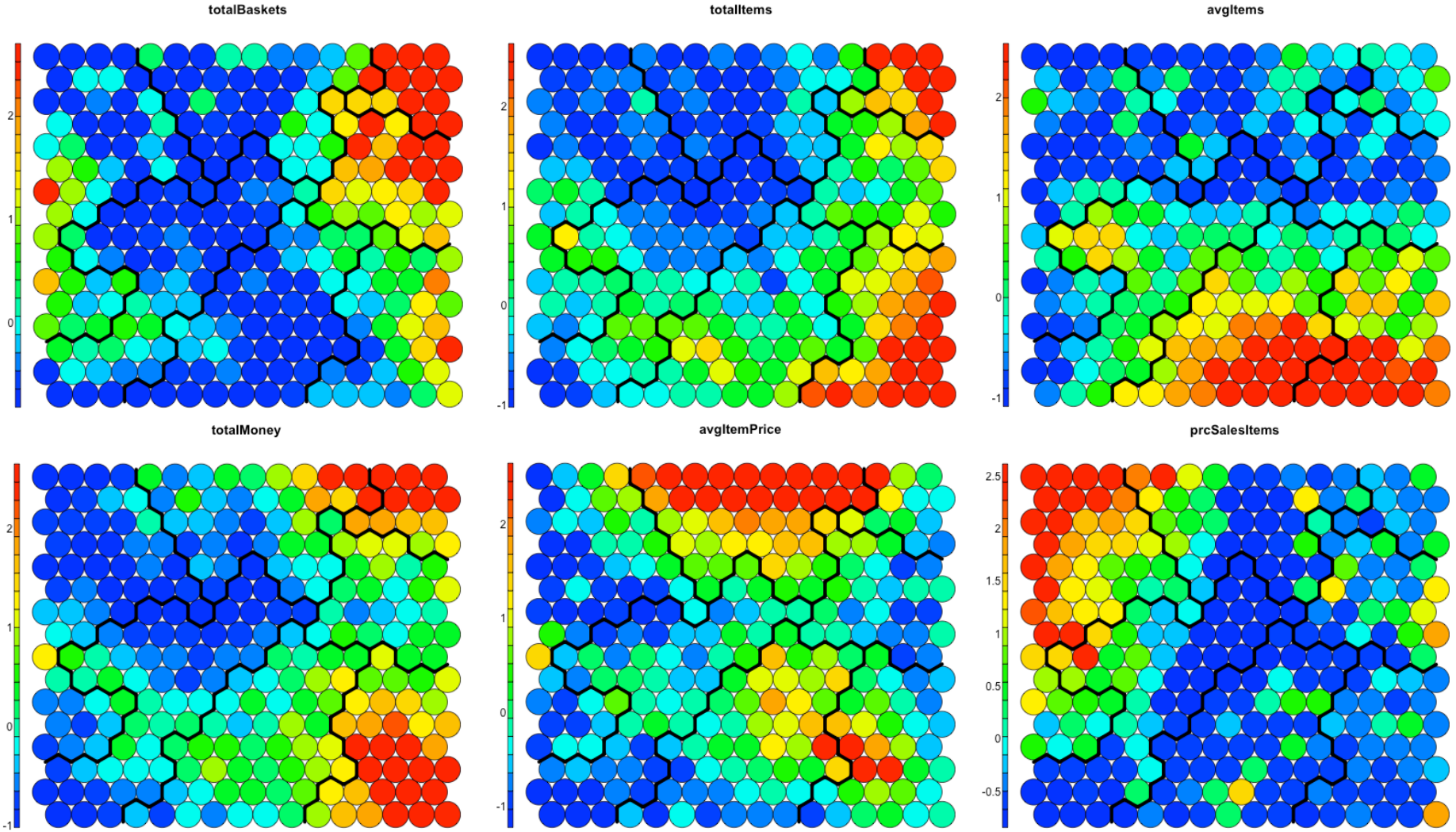
Cluster

```
pretty_palette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#F0E442”,
                    "#0072B2", "#D55E00", "#CC79A7")
taFeng.cluster <- adjacentWard(taFeng.som, toroidal = FALSE)
plot(taFeng.som, type="mapping", bgcol = pretty_palette[taFeng.cluster[[5]]])
add.cluster.boundaries(taFeng.som, taFeng.cluster[[5]])
```

- Hierarchical clustering
  - Initialised with every node as its own cluster
  - Most similar nodes are combined
  - Sometimes non-adjacent nodes are grouped together
- AdjacentWard
  - Initialised with every node as its own cluster
  - Most similar nodes are only combined if they are neighbours



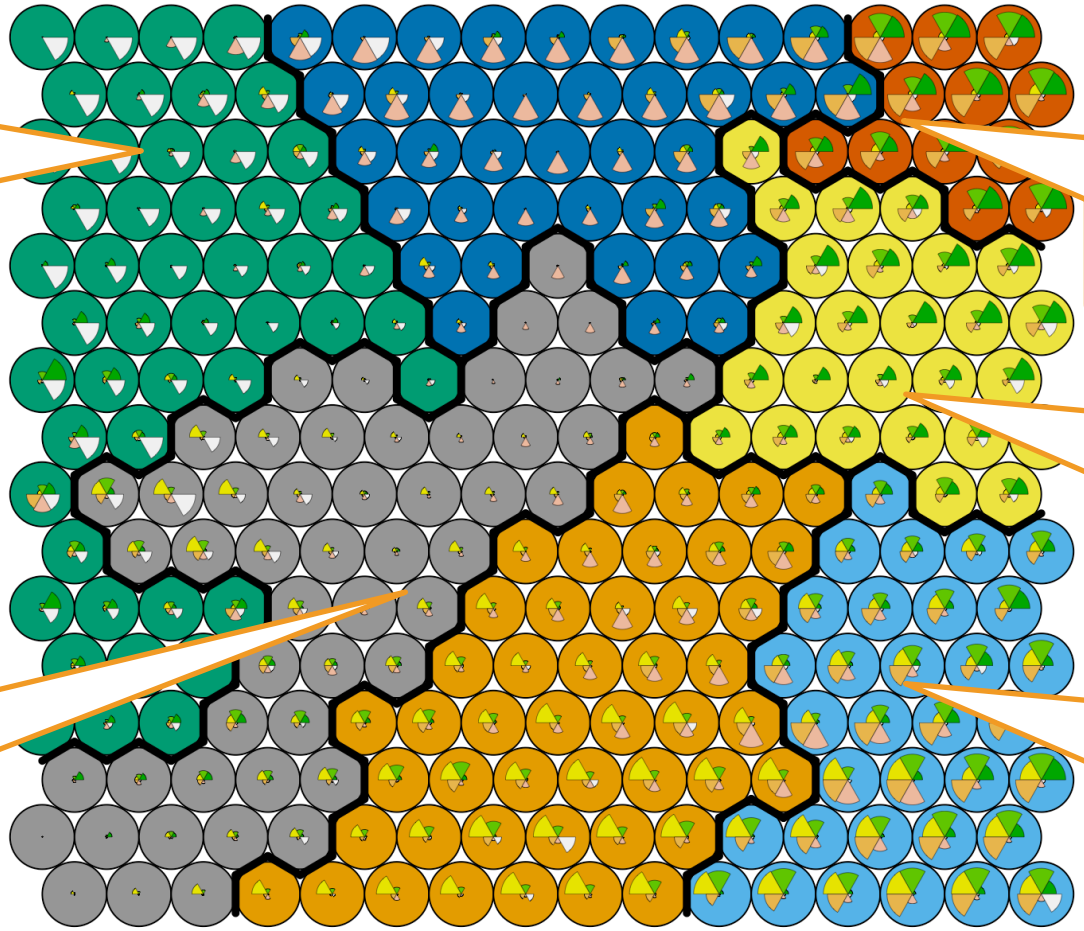
Gallery of Heatmaps with Cluster



Customer Segmentation

Bargain Hunter
- Few visits
- Focus on sale items

Walk-in customer
- Few visits
- Small profit



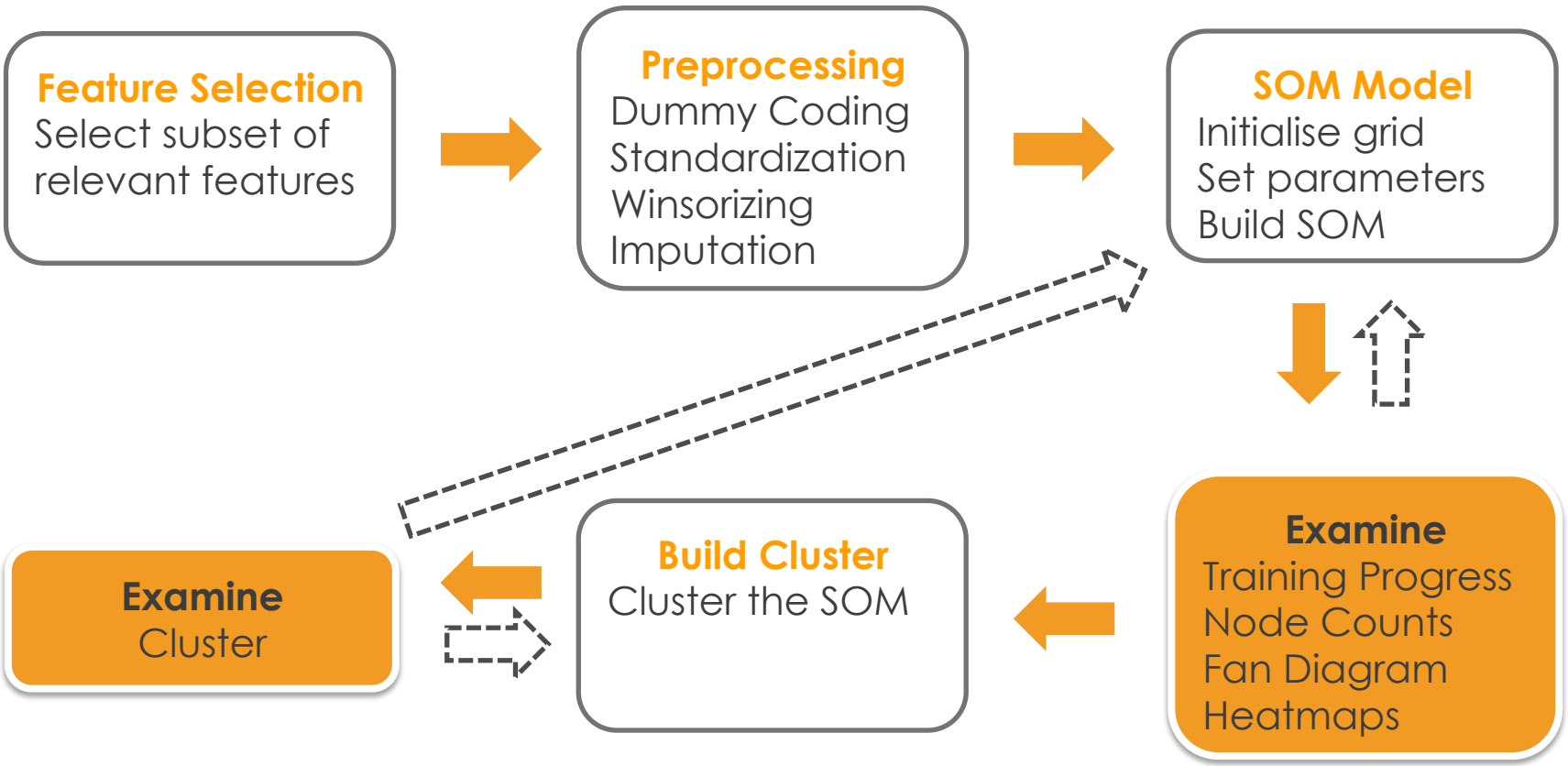
Regular Customer #1
- Many visits
- Huge Profit

Regular Customer #2
- Many visits
- Small profit

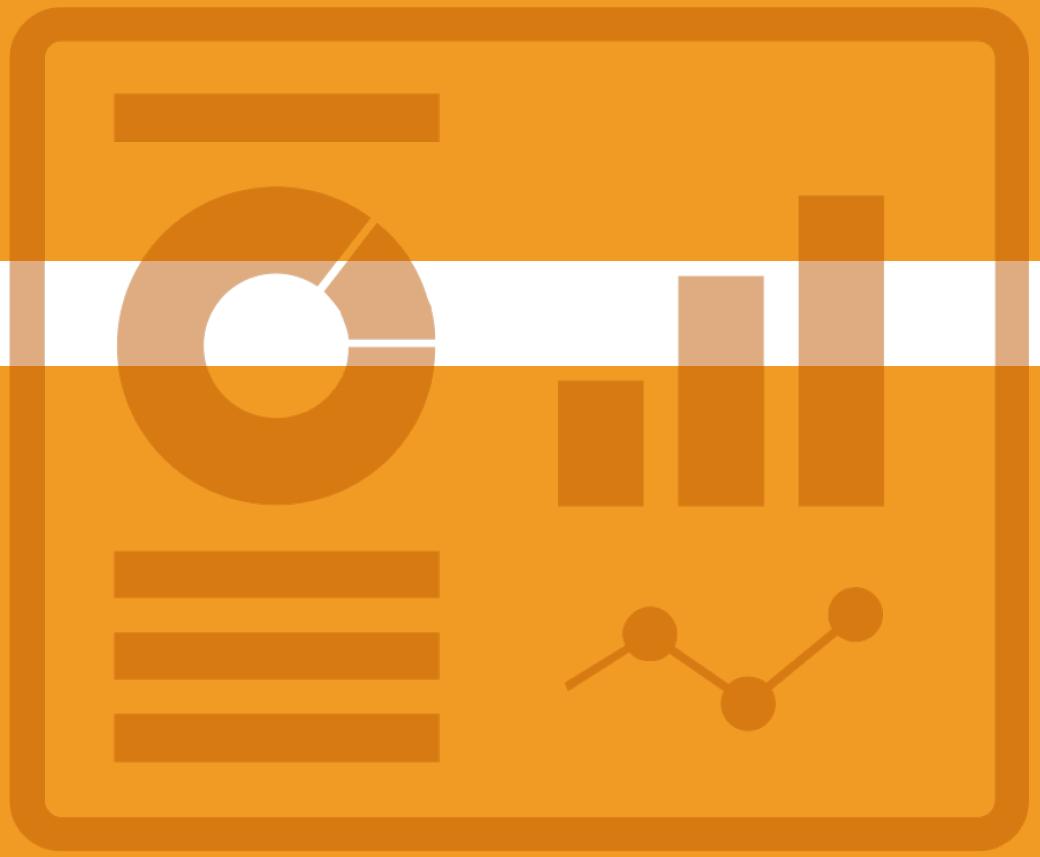
Bulk Buyer
- Huge Basket
- Profitable

totalBaskets | avgItems | avgItemPrice
totalItems | totalMoney | prcSalesItems

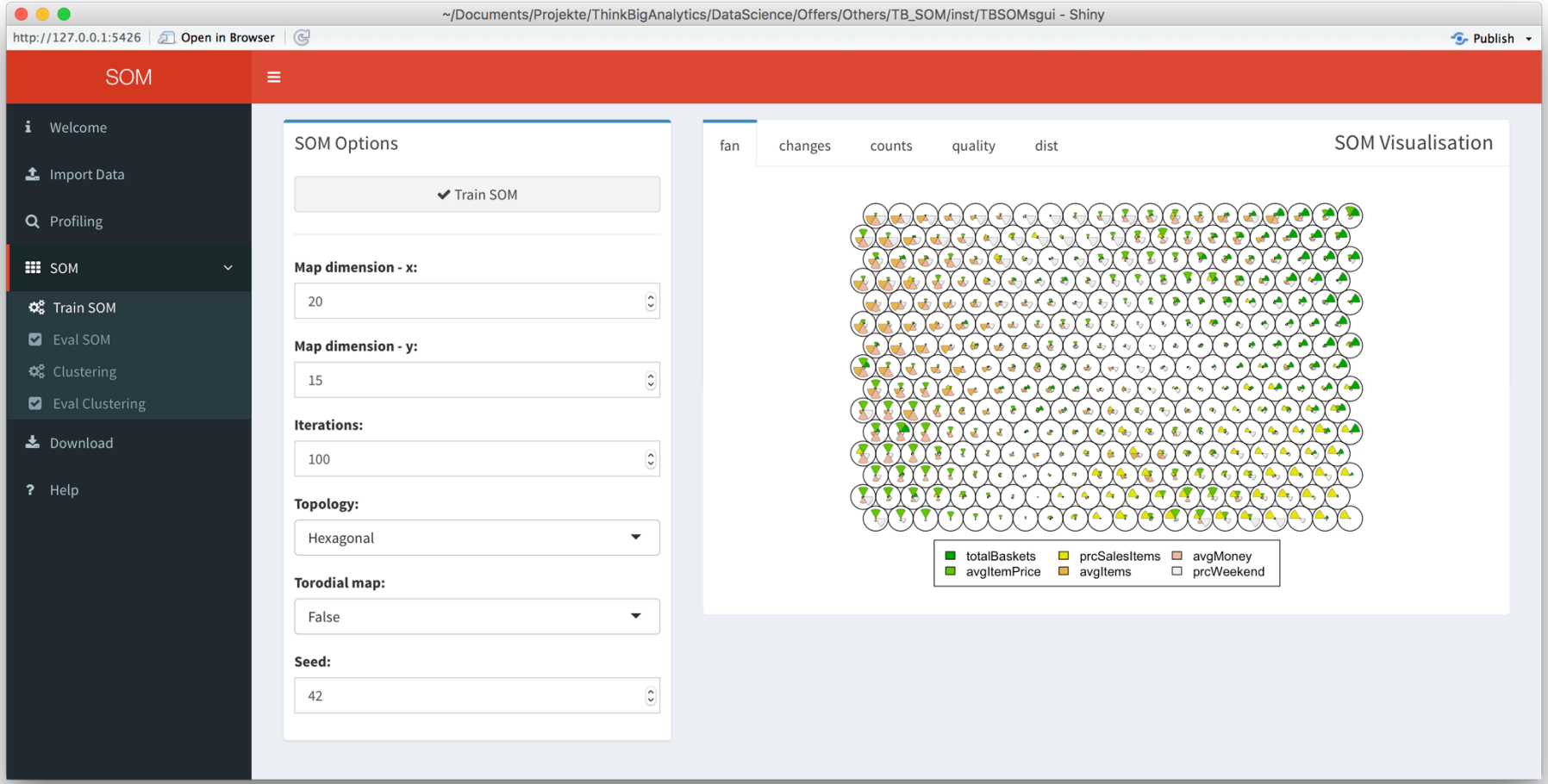
Workflow



Live Demo



R Shiny Dashboard



Conclusion

Conclusion

- Powerful algorithm
- Several versions of the original algorithm available

Advantages:

- Simple algorithm / easy to understand and interpret
- Visualisation of multi-dimensional data
- Capable of clustering large and complex data sets

Disadvantages:

- Lack of parallelisation
- Requires numeric data without any outliers or missing data
- Difficult to represent too many features in two dimensions
- Construct lots of maps to find a final good one

Be Part of a Great Team

THINKBIG
A TERADATA COMPANY

THINK YOU
THANK BIG