What's in the network?

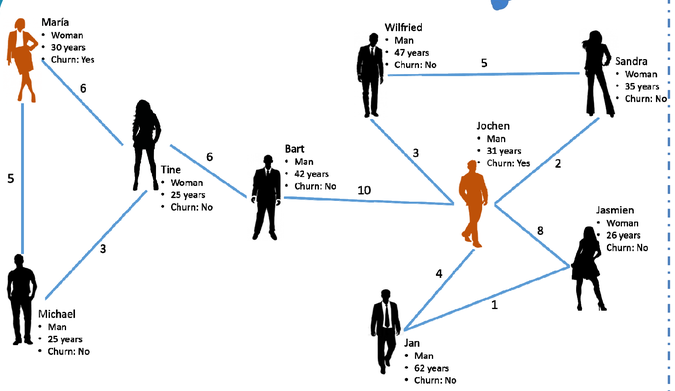
A stepwise overview of working with networked data in *R*

Tine Van Calster, Michael Reusens, María Óskarsdóttir, Sandra Mitrović, Jasmien Lismont, Jochen De Weerdt, Wilfried Lemahieu, Bart Baesens, and Jan Vanthienen

KU Leuven, Dept. of Decision Sciences and Information Management
Naamsestraat 69, B-3000, Leuven, Belgium; {GivenName.LastName@kuleuven.be}

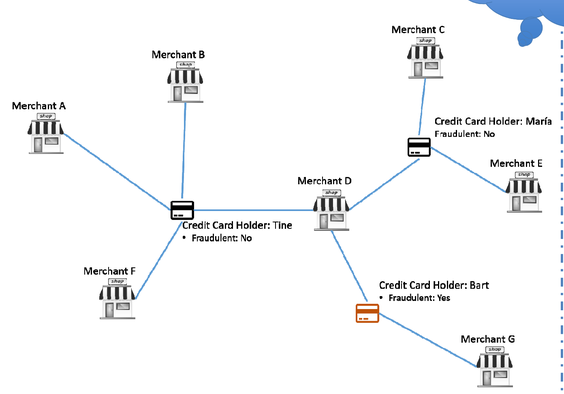
Business cases

Weighted unipartite network



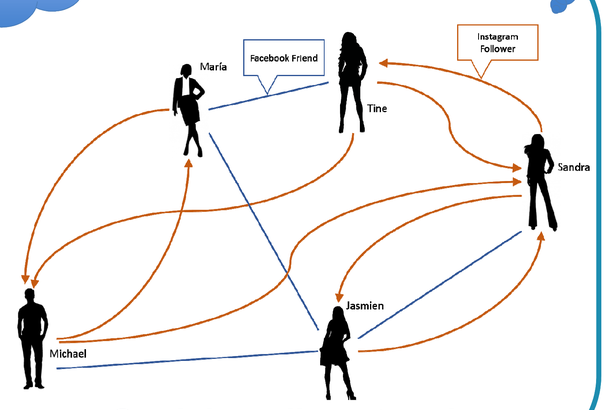
Call network

Bipartite network



Credit card fraud

Directed, multi-edges network



Social media network

- Networks are everywhere!
- They represent any type of connection between persons or objects
- Many applications in marketing, fraud, transportation, retail, biology, research citation, etc.
- Abundance of data leads to new challenges

Data structure

library("igraph")

```
#R code:
#graph g with nodes V(g) and edges E(g)
V(g)$name <- c("María", "Michael",etc.)
V(g)$churn <- c(1,0,0,0,1,etc.)
E(g)$weight <- c(5,6,3,etc.)
```

library("igraph")

```
#R code:
#transform bipartite into unipartite graph
gUnipartite <- bipartite_projection(gBipartite,
multiplicity = T, which = "false", remove.type= T)
```

Adjacency matrix

	María	Michael	Tine	Bart	Jochen	…
María	-	5	6	0	0	…
Michael	5	-	3	0	0	…
Tine	6	3	-	6	0	…
Bart	0	0	6	-	10	…
Jochen	0	0	0	10	-	…
…	…	…	…	…	…	…

library("igraph")

```
#R code:
g <- graph_from_adjacency_matrix(adjmatrix, mode =
"undirected", weighted = (1 | TRUE))
```

Edge list

Source	Target	Weight
María	Michael	5
Michael	Tine	3
Tine	Bart	6
Bart	Jochen	10
…	…	…

library("igraph")

```
#R code:
edges <- as.data.frame(get.edgelist(g))
# create edgelist from graph g
colnames(edges) <-
    c("Source", "Target", "Weight")
#Source = from; Target = to
```

Sparse matrix

	María	Michael	Tine	Bart	…
María	-	-	-	2	…
Michael	-	-	3	-	…
…	…	…	…	…	…

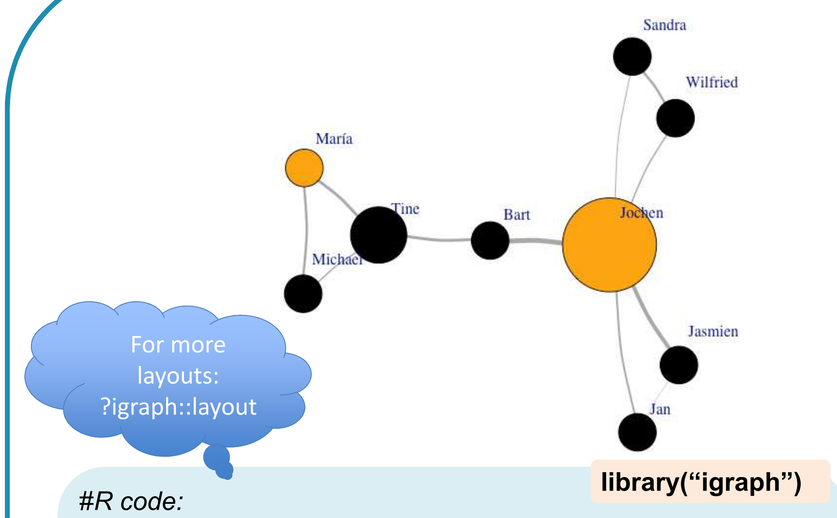
Row	Column	Value
María	Bart	2
Michael	Tine	3
…	…	…

Sparse matrices are useful when you have a lot of missing values ➔ take advantage of white space

library("Matrix")
library("slam")

```
#R code:
#create sparse adjacency matrix from edgelist using Matrix
sparseA <- sparseMatrix(dims=c(length(unique(edges$Source)),
                               length(unique(edges$Target))),
                        i = as.numeric(factor(edges$Source)),
                        j=as.numeric(factor(edges$Target)),
                        x = rep(1, length(as.numeric(edges$Source))) )
#alternatively use simple_triplet_matrix() from slam
```

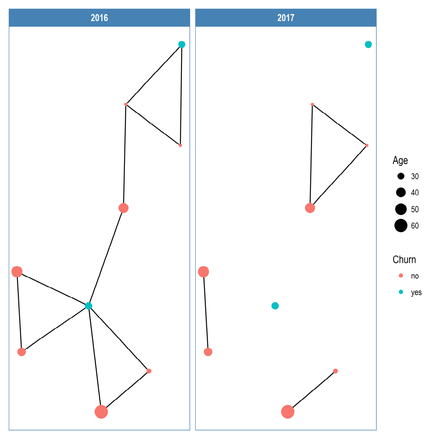
Network visualization



For more layouts: ?igraph::layout

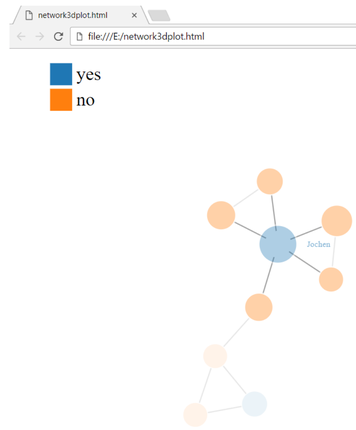
library("igraph")

```
#R code:
# adjust node size and edge width based on degree and weight
respectively; color churners in orange and the other in black
deg <- degree(g, mode="all");
V(g)$size <- deg*10; E(g)$width <- E(g)$weight/2;
V(g)$color <- ifelse(V(g)$churn == 1, "orange", "black");
plot(g, layout = layout.fruchterman.reingold, vertex.label.dist =
1.2, edge.arrow.size = 0.4, edge.curved = 0.1)
```



library("ggraph")

```
#R code:
ggraph(g, layout = "kk") +
  geom_edge_link(show.legend = FALSE) +
  geom_node_point(aes(size = age, color = churn)) +
  facet_edges(~year) +
  theme_graph(foreground = "steelblue", fg_text_colour =
  "white") + geom_node_text(aes(label = name), size = 2)
```



Embed graph visualizations in web-pages, markdown documents, and shiny apps

library("networkd3")

```
#R code:
#Works with an existing igraph object
d3g = igraph_to_networkD3(g,group = data)
# Create force directed network plot
forceNetwork(Links = d3g$links, Nodes = d3g$nodes,
        Source = "Source", Target = "Target", NodeID = "name",
        Group = "churn", zoom = TRUE, legend=TRUE,
        Nodesize = "age", colourScale =
        JS("d3.scaleOrdinal(d3.schemeCategory10);"),
        height= 400, width = 400) %>%
```

Analysis & Modeling

Featurization

library("igraph")

```
#R code:
#centralization measures:
V(g)$degree <- degree(g, mode = "all", loops = F, normalized = T);
V(g)$betweenness <- betweenness(g, directed = F); #requires a lot of computation power
#calculate PageRank for each node, can be personalized e.g. with churn scores
V(g)$page_rank <- page_rank(g, algo = "prpack", vids = V(g), directed = F, damping = 0.85,
    weights = NA, personalized = c(1,0,0,0,1,etc.) )$vector
```

Network learning

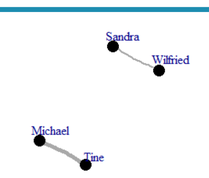
library("igraph")

```
#R code:
#edge prediction:
predict_edges(g, hrg = NULL, start = FALSE,
    num.samples = 10, num.bins = 25)
```

```
#Result
#Most likely edges
Jan – Sandra: 32%
Wilfried – Jasmien: 27%
```

Graph sampling

For more on sampling methods: ?igraph::sample_*



library("igraph")

```
#R code:
#random node sampling:
induced_subgraph(g, v=sample(1:length(V(g)),
    4, replace=FALSE))
```