# Use Case 1

Oleg Deev & Štefan Lyócsa

Masaryk University

# Goal

Predict profitability of loans - 'building credit scoring model'

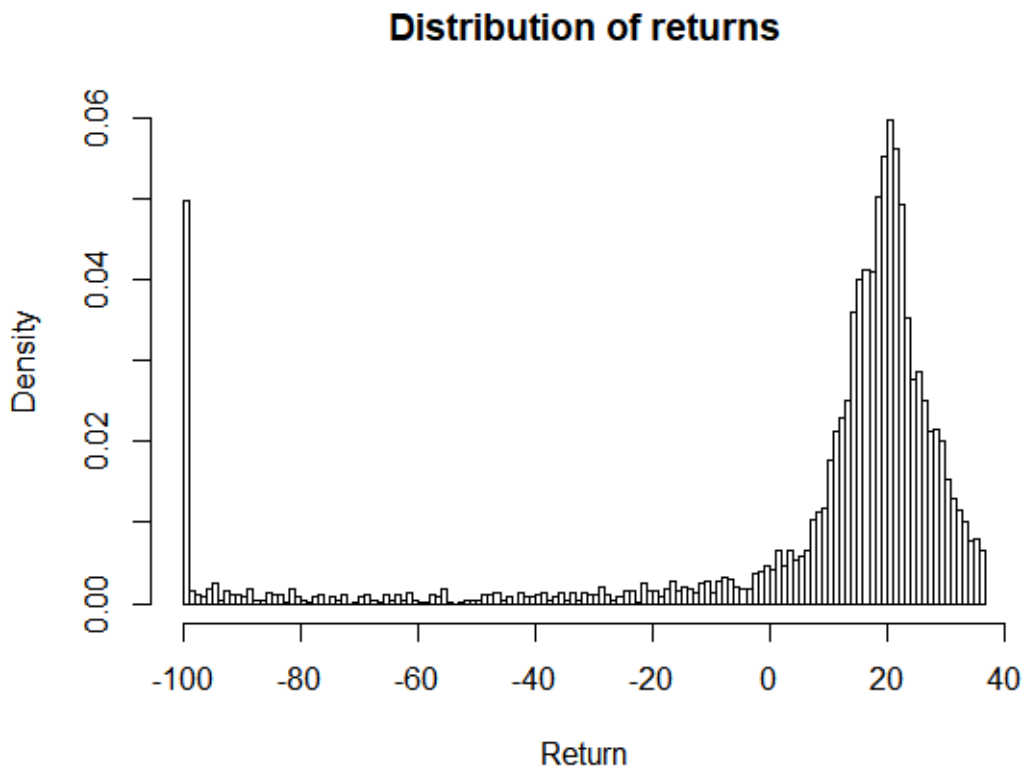
- Better credit models $->$ less defaults $->$ higher financial stability.
- Innovative financial services use (require) new approaches.

*'Starting September 8, 2017, we are rolling out the fifth-generation credit model, which further leverages machine learning and 10 years of LendingClub data to better assess and price credit risk.'*

https://blog.lendingclub.com/lendingclubs-next-generation-credit-model/

- We have data on $4157$ loans: loan amount, loan duration, ...
- We model profitability using different models:
  - OLS (benchmark model)
  - RIDGE model
  - LASSO model
  - Elastic Net model
- Using data from $4057$ loans we build a model and forecast profitability of the last $100$ loans.
- We evaluate models based on the mean squared error.

- We have data on $4157$ loans: loan amount, loan duration, ...



- We model profitability using different models:
  - OLS (benchmark model), $\alpha = 0, \lambda = 0$
  - RIDGE model, $\alpha = 0.0$
  - LASSO model, $\alpha = 1.0$
  - Elastic Net model, $0.0 < \alpha < 1.0$

$$\underset{\hat{\beta}_0,\ldots\hat{\beta}_p}{min} \rightarrow \frac{1}{2n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \lambda(\frac{1-\alpha}{2}\sum_{j=1}^{p}\beta_j^2 + \alpha\sum_{j=1}^{p}|\beta_j|)$$

- Using data from $4057$ loans we build a model and forecast profitability of the last $100$ loans.

- We evaluate models based on the **m**ean **s**quared **e**rror.

$$MSE = n^{-1} \sum_{i=1}^{n} (y - \hat{y})^2$$

- The idea is to **extract new** information from data, using **network analysis**, and include them in credit models.
- Network analysis **meets** econometric modeling

We hope, that new models augmented with network variables will improve credit models, i.e. the prediction of loan's profitability. To be able to compare models, we need to re-estimate models from previous session.

# Import data

Opening the file UseCase1 in RStudio. Importing data.
We split the sample:

- `NF = 100`
- `N = dim(DT)[1]`
- `S1 = DT[1:(N-NF),]`
- `S2 = DT[(N-NF+1):N,]`

Now we estimate models:...

# OLS model

Estimate model:

- `m1 = lm(RR2 new+ver3+ver4+lfi+lee+luk+lrs+lsk+age+undG+ female+lamt+int+durm+educprim+educbasic+ educvocat+educse espem+esfue+essem+esent+esret+dures+exper+ linctot+noliab lamntplr+lamteprl+nopearlyrep,data=S1)`

Forecast loan returns:

- `yhat = predict(m1,new=S2)`

Calculate **m**eans **s**quared **e**rror:

- `ytrue = S2$RR2`
- `OLS = mean((yhat-ytrue)^2)`

# LASSO model - estimation

Load program library:

- `library(glmnet)`

Define the matrix of input and output variables:

- `indep = as.matrix(S1[,c('new','ver3','ver4','lfi','lee'`
`'undG','female','lamt','int','durm','educprim','educbasic`
`'educvocat','educsec','msmar','msco','mssi','msdi','nrode`
`'espem','esfue','essem','esent','esret','dures','exper',`
`'linctot','noliab','lliatot','norli','noplo','lamountplo'`
`'lamntplr','lamteprl','nopearlyrep')])`
- `dep = S1$RR2`

Estimate model:

- `m2 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=1)`

# LASSO model - prediction

Define the matrix of input variables from predicted loans:

- `pred = as.matrix(S2[,c('new','ver3','ver4','lfi','lee',`
`'undG','female','lamt','int','durm','educprim','educbasic`
`'educvocat','educsec','msmar','msco','mssi','msdi','nrode`
`'espem','esfue','essem','esent','esret','dures','exper',`
`'linctot','noliab','lliatot','norli','noplo','lamountplo'`
`'lamntplr','lamteprl','nopearlyrep')])`

Forecast loan returns:

- `yhat = predict(m2,newx=pred,s=m2$lambda.1se)`

Calculate **m**ean **s**quared **e**rror:

- `ytrue = S2$RR2`
- `LASSO = mean((yhat-ytrue)^2)`

# RIDGE model

Estimate model:

- `m3 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0)`

Forecast loan returns:

- `yhat = predict(m3,newx=pred,s=m3$lambda.1se)`

Calculate **m**ean **s**quared **e**rror:

- `ytrue = S2$RR2`
- `RIDGE = mean((yhat-ytrue)^2)`

# Elastic Net model(s)

Estimate model:

- `m4_25 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.25)`
- `m4_50 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.50)`
- `m4_75 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.75)`

Forecast loan returns:

- `yhat = predict(m4_25 ,newx=pred,s=m4_25$lambda.1se)`
- `yhat = predict(m4_50 ,newx=pred,s=m4_50$lambda.1se)`
- `yhat = predict(m4_75 ,newx=pred,s=m4_75$lambda.1se)`

Calculate **m**ean **s**quared **e**rror:

- `EN25 = mean((yhat-ytrue)^2)`
- `EN50 = mean((yhat-ytrue)^2)`
- `EN75 = mean((yhat-ytrue)^2)`

# Elastic Net model(s)

| | MSEs |
|---|---|
| EN50 | 870.3685 |
| LASSO | 871.5219 |
| EN75 | 874.5317 |
| EN25 | 874.8249 |
| RIDGE | 929.6943 |
| OLS | 995.6180 |

How similar are loans?

- Perhaps, more risky loans share multiple features (e.g. higher loan, longer duration, past loans,...)
- How to compare loans across many characteristics?

We use a distance metric:

- Let $\boldsymbol{x}_i$ be a column vector of $i^{th}$ loan attributes.
- Let $\boldsymbol{\Delta}$ be a diagonal matrix with standard deviation of variables on the diagonal.

Distance metric:

$$d_{i,j} = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \Delta^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_j)$$

**Dissimilarity matrix** is defined as: $d_{i,j} \in \boldsymbol{D}$

**Dissimilarity matrix** is symmetric.

We (expert opinion) select variables that we think should be indicative of a bad loan:

- `DMV = DT[,c('int','durm','linctot','noliab')]`

Using Euclidean distance metric - we create the dissimilarity matrix $\boldsymbol{D}$.

- `DM = as.matrix(dist(scale(DMV)))`

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.000000 | 2.676694 | 1.074275 | 3.190169 | 1.238050 |
| 2 | 2.676694 | 0.000000 | 2.805000 | 5.334486 | 3.636588 |
| 3 | 1.074275 | 2.805000 | 0.000000 | 3.322493 | 1.264062 |
| 4 | 3.190169 | 5.334486 | 3.322493 | 0.000000 | 2.590860 |
| 5 | 1.238050 | 3.636588 | 1.264062 | 2.590860 | 0.000000 |

- Dissimilarity matrix is an adjacency matrix with weights. Larger distance between loan $i$ and loan $j$, the less similar are loans $i$ and $j$, i.e. the edge between the two loans is longer.
- Dissimilarity matrix leads to a **complete graph**. Everybody is connected to everybody. Way too complex.

A common strategy is to select a 'suitable' **sub-graph**. From complete graph we can extract the **M**inimum **S**panning **T**ree. Minimum spanning tree is a subset of a **connected graph** which satisfies the following:

- connects all vertices,
- there are no cycles,
- with a minimum possible total distance.

Click here for the Animated Kruskal algorithm

en.wikipedia.org see Kruskal Algorithm

- `library(igraph)` We create a graph object in R:
- `g = graph_from_adjacency_matrix(DM, mode = ündirected", weighted = TRUE)`

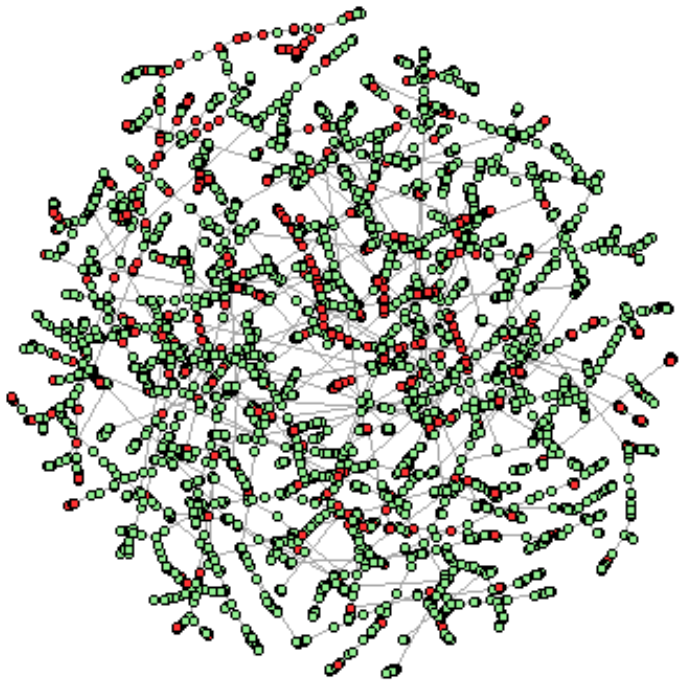
We create a minimum spanning tree:

- `g_mst = mst(g)`

We visualize the 'MST':

- `status = (S1$RR2<0)*1`
- `V(g_mst)$status = status`
- `V(g_mst)[status == 1]$color = "firebrick1"`
- `V(g_mst)[status == 0]$color = "lightgreen"`
- `plot(g_mst, graph = "MST",vertex.label=NA,vertex.size = 3,main = "MST of the P2P applicants networks")`

**MST of the P2P applicants networks**



- vertex degree
- vertex strength
- closeness, $c(x) = \frac{1}{\sum_y d(y,x)}$
- betweenness, $b(x) = \sum_{s \neq v \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$
- Community detection - Louvain method

Louvain method creates communities while increasing the modularity of the network:

$$Q = \frac{1}{2m} \sum_{ij} \left[ d_{i,j} - \frac{s_i s_j}{2m} \right] \delta(c_i, c_j)$$

Here, $2m$ is the sum of all edge weights in the graph, $s_i$ is the sum of all weights of vertex $i$, and $\delta(c_i, c_j)$ is a delta function such that it returns $1$ if $c_i = c_j$ and $0$ otherwise (i.e. if the two vertices belong to the same community it returns $1$.

# Estimating vertex level attributes in R

Degree centrality:

- `DT$Deg = degree(g_mst)`

Vertex strength:

- `DT$Str = strength(g_mst)`

**Closeness** centrality:

- `DT$Clos = closeness(g_mst)`$*10^4$

**Betweenness** centrality:

- `DT$Bet = betweenness(g_mst)`

Community detection via **Louvain method**:

- `com = cluster_louvain(g_mst)`
- `length(unique(com$membership))`
- `CD=dummy(com$membership)`
- `DT = data.frame(DT,CD)`

# Preparing data

Define the matrix of input and output variables:

- `indep = as.matrix(S1[,c('new','ver3','ver4','lfi','lee'`
`'undG','female','lamt','int','durm','educprim','educbasic`
`'educvocat','educsec','msmar','msco','mssi','msdi','nrode`
`'espem','esfue','essem','esent','esret','dures','exper',`
`'linctot','noliab','lliatot','norli','noplo','lamountplo'`
`'lamntplr','lamteprl','nopearlyrep','Deg','Str','Clos','B`
`paste('membership',1:124,sep=''))])`

We have additional $124$ communities - clusters! Suitable for shrinkage methods.

- `dep = S1$RR2`

# Preparing data

- pred = as.matrix(S2[,c('new','ver3','ver4','lfi','lee',
'undG','female','lamt','int','durm','educprim','educbasic
'educvocat','educsec','msmar','msco','mssi','msdi','nrode
'espem','esfue','essem','esent','esret','dures','exper',
'linctot','noliab','lliatot','norli','noplo','lamountplo'
'lamntplr','lamteprl','nopearlyrep','Deg','Str','Clos','B
paste('membership',1:124,sep='')),])
- ytrue = S2$RR2

# LASSO model

Model estimation:

- `m5_L = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=1)`
- `coef(m5_L,s='lambda.1se')`

Forecast loan returns:

- `yhat = predict(m5_L,newx=pred,s=m5_L$lambda.1se)`

Calculate Mean squared error:

- `LASSO_N = mean((yhat-ytrue)^2)`

# RIDGE model

Model estimation:

- `m5_R = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0)`
- `coef(m5_R,s='lambda.1se')`

Forecast loan returns:

- `yhat = predict(m5_R,newx=pred,s=m5_R$lambda.1se)`

Calculate Mean squared error:

- `RIDGE_N = mean((yhat-ytrue)^2)`

# Elastic net model

- `m5_E25 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.25)`
- `m5_E50 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.50)`
- `m5_E75 = cv.glmnet(x=indep,y=dep,nfolds=30,alpha=0.75)`

- `yhat = predict(m5_E25 ,newx=pred,s=m5_E25$lambda.1se)`
- `yhat = predict(m5_E50 ,newx=pred,s=m5_E50$lambda.1se)`
- `yhat = predict(m5_E75 ,newx=pred,s=m5_E75$lambda.1se)`

- `EN25N = mean((yhat-ytrue)^2)`
- `EN50N = mean((yhat-ytrue)^2)`
- `EN75N = mean((yhat-ytrue)^2)`

# Comparing forecast accuracy

Is network approach worth the strugle?

- `MSEs = c(OLS, LASSO, RIDGE, EN25, EN50, EN75, LASSO_N, RIDGE_N, EN25N, EN50N, EN75N)`
- `names(MSEs) = c('OLS', 'LASSO', 'RIDGE', 'EN25', 'EN50', 'EN75', 'LASSO_N', 'RIDGE_N', 'EN25N', 'EN50N', 'EN75N')`
- `MSEs = sort(MSEs)`
- `cbind(MSEs)`

# Comparing forecast accuracy

Is the network approach worth the struggle?

| | MSEs |
|---|---|
| LASSO_N | 856.3854 |
| EN25N | 858.0379 |
| RIDGE_N | 860.3076 |
| EN50N | 862.0838 |
| EN75N | 864.3380 |
| EN50 | 870.3685 |
| LASSO | 871.5219 |
| EN75 | 874.5317 |
| EN25 | 874.8249 |
| RIDGE | 929.6943 |
| OLS | 995.6180 |

# Use Case 1

Oleg Deev & Štefan Lyócsa

Masaryk University