Developing credit risk scoring

using R programming

Presenter: **Mihai DAVID**

Date: April 28th, 2020
Venue: **R ladies Meet-Up**

Content

Disclaimer

Opinions expressed are strictly and wholly of the presenter.

R and all other R product or service names are registered trademarks.

Other brand and product names are trademarks of their respective companies.

The information contained in the present document has not been independently verified and no representation or warranty expressed or implied is made as to, and no reliance should be placed on the fairness, accuracy, completeness or correctness of this information or opinions contained herein.

The presenter shall have no liability from any loss howsoever arising from any use of this document or its content or otherwise arising in connection with this document.

Abstract

- Building solid and reliable credit scoring systems is of the utmost importance for the financial institutions as one of the most important threats to a country's financial stability is delinquency and defaults of credits (*Bayraci, 2017*)
- **Credit Risk Score** is an analytical method of modeling the credit riskiness of individual borrowers. This statistical measure, transformed into a individual score, is used in the credit decision-making process, along with other business considerations
- In measuring credit risk, a **variety of statistical and machine learning models** are being used, such as: discriminant analysis, linear regression, logit analysis, decision trees, Bayesian classifiers, k-nearest neighbors, support vector machines, and artificial neural networks.
- This presentation will introduce the audience on **how to develop an in-house Credit Risk Score in R programming, using k-nearest neighbors**
    - **The k-nearest neighbor method (k-NN**) - a non-parametric statistical approach which evaluates the similarities between the pattern identified in the training set and the input pattern. It is based on choosing a metric on the space of applicants and takes k-nearest neighbor of the input pattern that is nearest in some metric sense. A new applicant will be classified in the class to which the majority of the neighbors belong (*Bayraci, 2017*)
- The scoring development flow is applied on a **sample** credit scoring data set
- **R Studio and R packages** have been used in the estimation processes

Scoring Set-up

1. Database – learning and testing
2. “Bad” definition
3. List of characteristics
4. Predictive power of characteristics
5. Scorecard
6. Descriptive statistics & Score
7. Machine learning model
8. Confusion matrix and results

Database

- **Dataset:** sample database of **4.446** loans, downloaded from GitHub

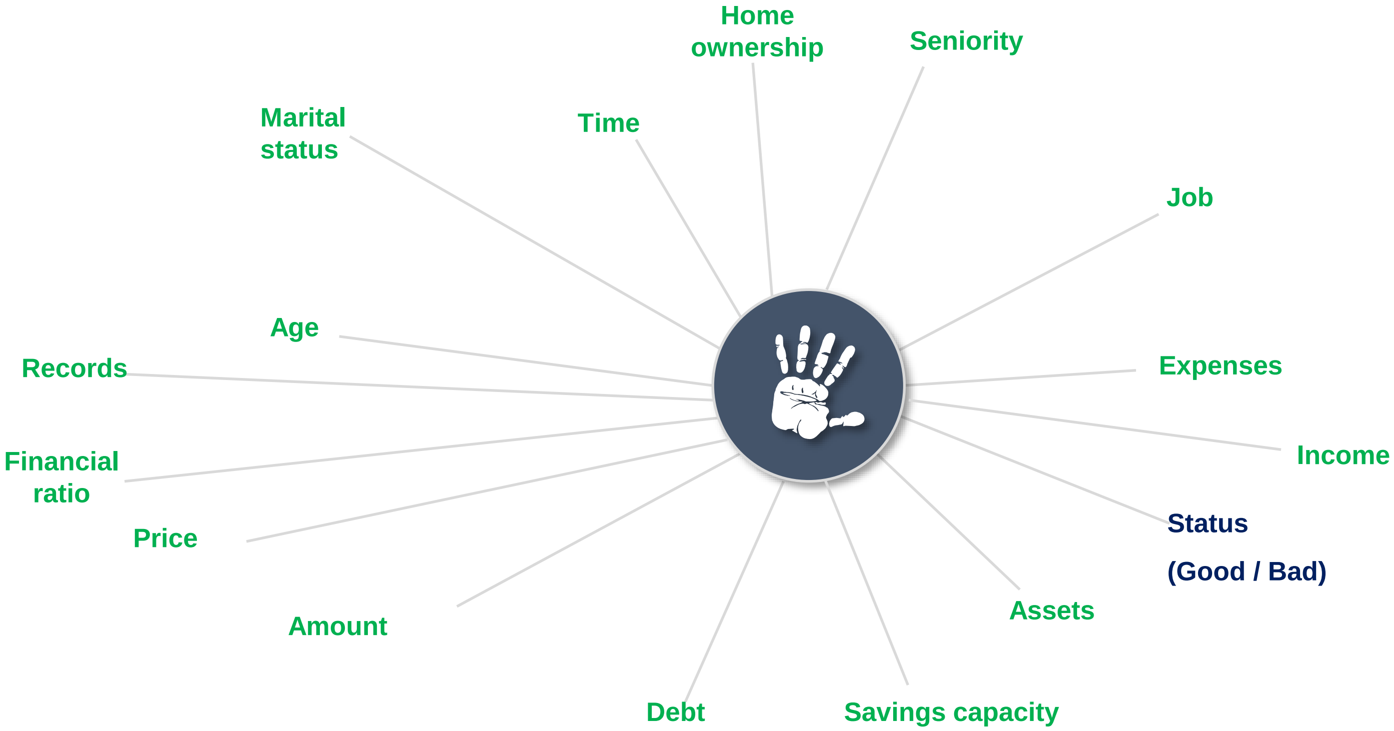
Sample portfolio data

Status	Seniority	Home	Time	Age	Marital	Records	Job	Expenses	Income	Assets	Debt	Amount	Price	Finrat	Savings
good	9	rent	60	30	married	no_rec	freelance	73	129	0	0	800	846	94.56265	4.2
good	17	rent	60	58	widow	no_rec	fixed	48	131	0	0	1000	1658	60.31363	4.98
bad	10	owner	36	46	married	yes_rec	freelance	90	200	3000	0	2000	2985	67.00168	1.98
good	0	rent	60	24	single	no_rec	fixed	63	182	2500	0	900	1325	67.92453	7.933333
good	0	rent	36	26	single	no_rec	fixed	46	107	0	0	310	910	34.06593	7.083871
good	1	owner	60	36	married	no_rec	fixed	75	214	3500	0	650	1645	39.51368	12.83077
good	29	owner	60	44	married	no_rec	fixed	75	125	10000	0	1600	1800	88.88889	1.875
good	9	parents	12	27	single	no_rec	fixed	35	80	0	0	200	1093	18.29826	2.7
good	0	owner	60	32	married	no_rec	freelance	90	107	15000	0	1200	1957	61.31834	0.85
bad	0	parents	48	41	married	no_rec	partime	90	80	0	0	1200	1468	81.74387	-0.4
good	6	owner	48	34	married	no_rec	freelance	60	125	4000	0	1150	1577	72.92327	2.713043
good	7	owner	36	29	married	no_rec	fixed	60	121	3000	0	650	915	71.03825	3.378462
good	8	owner	60	30	married	no_rec	fixed	75	199	5000	2500	1500	1650	90.90909	3.96
good	19	priv	36	37	married	no_rec	fixed	75	170	3500	260	600	940	63.82979	5.544
bad	0	other	18	21	single	yes_rec	partime	35	50	0	0	400	500	80	0.675
good	0	owner	24	68	married	no_rec	fixed	75	131	4162	0	900	1186	75.88533	1.493333
good	15	priv	24	52	single	no_rec	freelance	35	330	16500	0	1500	2201	68.15084	4.72

For illustrative purpose only

Variables

- For each customer in the dataset, **16** variables are analyzed
- Each scoring defines a specific set of variables which generally cannot be disclosed for confidentiality reasons.

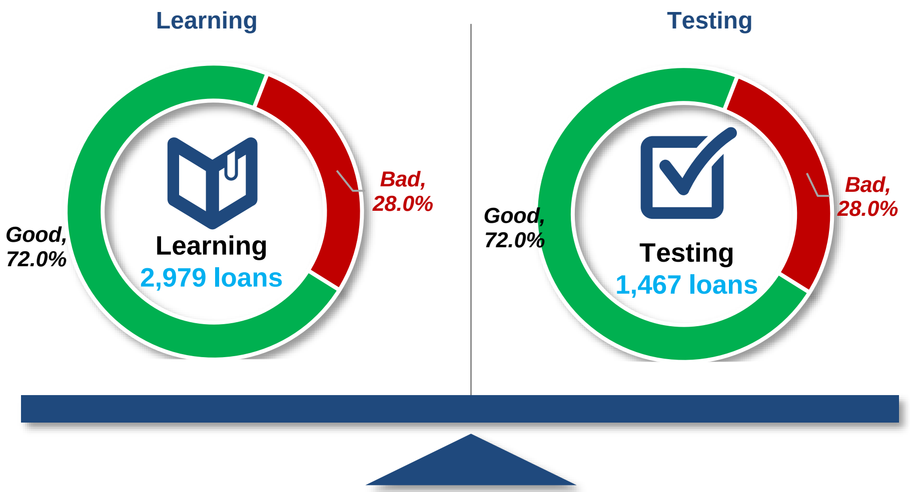


Good / Bad

1 Status	credit status
2 Seniority	job seniority (years)
3 Home	type of home ownership
4 Time	time of requested loan
5 Age	client's age
6 Marital	marital status
7 Records	existance of records
8 Job	type of job
9 Expenses	amount of expenses
10 Income	amount of income
11 Assets	amount of assets
12 Debt	amount of debt
13 Amount	amount requested of loan
14 Price	price of good
15 Financial ratio	
16 Savings capacity	

Database partition and “Bad definition”

- **Bad definition**
  - When developing a credit risk score, the definition of default ("Bad") must be clearly established via regulatory (e.g., Basel II, IFRS 9) and Risk analytics. Example of Bad: **30+, 90+**
  - **Default rate** in analyzed dataset = **28%** (1.249 loans)
- **Dataset partition:**
  - **Learning → 67%** of the dataset (2,979 observations), randomly selected from the full dataset
  - **Testing → 33%** of the dataset (1,467 observations), randomly selected from the full dataset, used for model evaluation purposes



Dataset partition into Learning and Testing in R

```
#impart dd pe trainig si test
library(caTools)
set.seed(123)
split = sample.split(dd$Status, SplitRatio = 0.67)
training_set = subset(dd, split == TRUE)
test_set = subset(dd, split == FALSE)
```

For illustrative purpose only

Binning

- **Variable Transformation** is performed through **binning**, which is widely accepted as the "Gold standard" and has good interpretability with business implications

Original portfolio data

Status	Seniority	Home	Time	Age	Marital	Records	Job	Expenses	Income	Assets	Debt	Amount	Price	Finrat	Savings
good	9	rent	60	30	married	no_rec	freelance	73	129	0	0	800	846	94.56265	4.2
good	17	rent	60	58	widow	no_rec	fixed	48	131	0	0	1000	1658	60.31363	4.98
bad	10	owner	36	46	married	yes_rec	freelance	90	200	3000	0	2000	2985	67.00168	1.98
good	0	rent	60	24	single	no_rec	fixed	63	182	2500	0	900	1325	67.92453	7.933333
good	0	rent	36	26	single	no_rec	fixed	46	107	0	0	310	910	34.06593	7.083871
good	1	owner	60	36	married	no_rec	fixed	75	214	3500	0	650	1645	39.51368	12.83077
good	29	owner	60	44	married	no_rec	fixed	75	125	10000	0	1600	1800	88.88889	1.875
good	9	parents	12	27	single	no_rec	fixed	35	80	0	0	200	1093	18.29826	2.7
good	0	owner	60	32	married	no_rec	freelance	90	107	15000	0	1200	1957	61.31834	0.85
bad	0	parents	48	41	married	no_rec	partime	90	80	0	0	1200	1468	81.74387	-0.4
good	6	owner	48	34	married	no_rec	freelance	60	125	4000	0	1150	1577	72.92327	2.713043
good	7	owner	36	29	married	no_rec	fixed	60	121	3000	0	650	915	71.03825	3.378462
good	8	owner	60	30	married	no_rec	fixed	75	199	5000	2500	1500	1650	90.90909	3.96
good	19	priv	36	37	married	no_rec	fixed	75	170	3500	260	600	940	63.82979	5.544
bad	0	other	18	21	single	yes_rec	partime	35	50	0	0	400	500		
good	0	owner	24	68	married	no_rec	fixed	75	131	4162	[illegible]	900	1186	75.885	
good	15	priv	24	52	single	no_rec	freelance	35	330	16500	0	1500	2201	68.150	

Variable transformation in R

```r
v_sen= quantile(dd$Seniority,probs = seq(0, 1, 0.25))
v_sen=as.numeric(v_sen)
seniorityR = cut(dd$Seniority, breaks=v_sen)
seniorityR[is.na(seniorityR)]<-"(0,2]"
dd$seniorityR<-seniorityR

v_sen= quantile(dd$Time,probs = seq(0, 1, 0.25))
v_sen=as.numeric(v_sen)
timeR = cut(dd$Time, breaks=v_sen)
timeR[is.na(timeR)]<- "(6,36]"
dd$timeR<-timeR
```

Transformed variables

seniorityR	timeR	ageR	expensesR	incomeR	assetsR	debtR	amountR	priceR	finratR	savingsR
(5,12]	(48,60]	(28,36]	(72,180]	(124,170]	(0,3000]	(0,30000]	(700,1000]	(105,1116]	(88.5,100]	(3.12,5.2]
(12,48]	(48,60]	(45,68]	(35,51]	(124,170]	(0,3000]	(0,30000]	(700,1000]	(1400,1691]	(60,77.1]	(3.12,5.2]
(5,12]	(6,36]	(45,68]	(72,180]	(170,959]	(0,3000]	(0,30000]	(1300,5000]	(1691,11140]	(60,77.1]	(1.62,3.12]
(0,2]	(48,60]	(18,28]	(51,72]	(170,959]	(0,3000]	(0,30000]	(700,1000]	(1116,1400]	(60,77.1]	(5.2,33.2]
(0,2]	(6,36]	(18,28]	(35,51]	(90,124]	(0,3000]	(0,30000]	(100,700]	(105,1116]	(6.7,60]	(5.2,33.2]
(0,2]	(48,60]	(28,36]	(72,180]	(170,959]	(3000,6000]	(0,30000]	(100,700]	(1400,1691]	(6.7,60]	(5.2,33.2]
(12,48]	(48,60]	(36,45]	(72,180]	(124,170]	(6000,300000]	(0,30000]	(1300,5000]	(1691,11140]	(88.5,100]	(1.62,3.12]
(5,12]	(6,36]	(18,28]	(35,51]	(1,90]	(0,3000]	(0,30000]	(100,700]	(105,1116]	(6.7,60]	(1.62,3.12]
(0,2]	(48,60]	(28,36]	(72,180]	(90,124]	(6000,300000]	(0,30000]	(1000,1300]	(1691,11140]	(60,77.1]	(-8.16,1.62]
(0,2]	(36,48]	(36,45]	(72,180]	(1,90]	(0,3000]	(0,30000]	(1000,1300]	(1400,1691]	(77.1,88.5]	(-8.16,1.62]
(5,12]	(36,48]	(28,36]	(51,72]	(124,170]	(3000,6000]	(0,30000]	(1000,1300]	(1400,1691]	(60,77.1]	(1.62,3.12]

Descriptive statistics

- **Transformed variables (11 of the 16)** after binning (**number of loans** in each category):
- This allows us to assess the dataset distribution and **customer profile**
  - For example: most of our customers are aged 30 – 40 years and have a work seniority of less than one year

Seniority

seniorityR	Trn	Tst
(0,2]	1021	475
(2,5]	553	280
(5,12]	676	356
(12,48]	729	356
Grand Total	**2979**	**1467**

Time

timeR	Trn	Tst
(6,36]	1075	537
(36,48]	591	294
(48,60]	1312	636
(60,72]	1	
Grand Total	**2979**	**1467**

Age

ageR	Trn	Tst
(18,28]	797	396
(28,36]	780	380
(36,45]	684	343
(45,68]	718	348
Grand Total	**2979**	**1467**

Expenses

expensesR	Trn	Tst
(35,51]	1485	751
(51,72]	740	362
(72,180]	754	354
Grand Total	**2979**	**1467**

Income

incomeR	Trn	Tst
(1,90]	796	376
(90,124]	714	352
(124,170]	729	370
(170,959]	740	369
Grand Total	**2979**	**1467**

Assets

assetsR	Trn	Tst
(0,3000]	1500	752
(3000,6000]	787	373
(6000,300000]	692	342
Grand Total	**2979**	**1467**

Debt

debtR	Trn	Tst
(0,30000]	2979	1467
Grand Total	**2979**	**1467**

Amount

amountR	Trn	Tst
(100,700]	786	377
(700,1000]	881	412
(1000,1300]	639	307
(1300,5000]	673	371
Grand Total	**2979**	**1467**

Price

priceR	Trn	Tst
(105,1116]	748	364
(1116,1400]	762	355
(1400,1691]	752	353
(1691,11140]	717	395
Grand Total	**2979**	**1467**

Financial ratio

finratR	Trn	Tst
(6.7,60]	753	359
(60,77.1]	752	359
(77.1,88.5]	724	387
(88.5,100]	750	362
Grand Total	**2979**	**1467**

Savings capacity

savingsR	Trn	Tst
(-8.16,1.62]	747	368
(1.62,3.12]	741	374
(3.12,5.2]	753	351
(5.2,33.2]	738	374
Grand Total	**2979**	**1467**

Home

Row Labels	Count of Home
good	**3,197.00**
ignore	11.00
other	173.00
owner	1,716.00
parents	550.00
priv	162.00
rent	585.00
bad	**1,249.00**
ignore	9.00
other	146.00
owner	390.00
parents	232.00
priv	84.00
rent	388.00
Grand Total	**4,446.00**

Job

Row Labels	Count of Job
good	**3,197.00**
fixed	2,223.00
freelance	690.00
others	103.00
partime	181.00
bad	**1,249.00**
fixed	580.00
freelance	331.00
others	68.00
partime	270.00
Grand Total	**4,446.00**

Row Labels	Average of Seniority	Average of Age	Average of Expenses	Average of Income	Average of Assets	Average of Debt	Average of Amount	Average of Price	Average of Finrat	Average of Savings
good	9.32	37.74	55.24	147.86	6,055.91	334.16	992.97	1,458.50	69.79	4.29
bad	4.59	35.41	56.53	122.11	3,560.74	362.98	1,155.97	1,472.68	79.85	2.75

Predictive power of variables – **Information Value (IV)**

- **Information value (IV)** of each variable is calculated using the formula:

$$IV = \sum_{i=1}^{n} (\text{Distr. Good} - \text{Distr. Bad}) * \ln\left(\frac{\text{Distr. Good}}{\text{Distr. Bad}}\right) * 100$$

where

- *n* is number of characteristics of each variable

Best practice regarding inferences from information value would be that:

- Less than 0.02 - Not Predictive;
- 0.02 to 0.05 – Weak;
- 0.05 to 0.3 – Medium;
- More than 0.3 – Strong

- Out of all variables in dataset, we choose those with **medium and strong IV.** This combination of variables and different smaller combinations will be considered in order to compute Gini.

Information value calculation

```
#calcul IV
library("woe")
library("InformationValue")

sen<-IV(X=as.factor(training_set$seniorityR),Y=training_set$Status)
hm<-IV(X=as.factor(training_set$Home),Y=training_set$Status)
tm<-IV(X=as.factor(training_set$timeR),Y=training_set$Status)
ag<-IV(X=as.factor(training_set$ageR),Y=training_set$Status)
mr<-IV(X=as.factor(training_set$Marital),Y=training_set$Status)
rec<-IV(X=as.factor(training_set$Records),Y=training_set$Status)
jb<-IV(X=as.factor(training_set$Job),Y=training_set$Status)
exp<-IV(X=as.factor(training_set$expensesR),Y=training_set$Status)
inc<-IV(X=as.factor(training_set$incomeR),Y=training_set$Status)
asst<-IV(X=as.factor(training_set$assetsR),Y=training_set$Status)
deb<-IV(X=as.factor(training_set$debtR),Y=training_set$Status)
amt<-IV(X=as.factor(training_set$amountR),Y=training_set$Status)
pt<-IV(X=as.factor(training_set$priceR),Y=training_set$Status)
fin<-IV(X=as.factor(training_set$finratR),Y=training_set$Status)
sav<-IV(X=as.factor(training_set$savingsR),Y=training_set$Status)
IVdata<-data.frame(sen,hm,tm,ag,mr,rec,jb,exp,inc,asst,deb,amt,pt,fin,sav)
IVdata
```

Information value of each variable

\> IVdata

	sen	hm	tm	ag	mr	rec	jb	exp	inc	asst	deb	amt	pt	fin	sav
1	0.4745959	0.2643097	0.02472338	0.05779807	0.06791812	0.3393173	0.3974399	0.02284326	0.2022236	0.1816303	0	0.1078428	0.03637394	0.2682613	0.2358478

\>

7 Variables with medium and strong IV: **seniority, home, job, income, assets, financial ratio, savings capacity**

For illustrative purpose only

Predictive power of Variables – **Weight of evidence (WOE)**

- The **weight of evidence (WOE)** is used to measure the strength of each bin in isolating “good” from “bad” accounts. It is calculated using the following formula:

$$WOE = \left[\ln\left(\frac{\text{Distr. Good}}{\text{Distr. Bad}}\right)\right] * 100$$

where,

  - Distr. Good – percentage of “good” accounts in the sample data,
  - Distr. Bad – percentage of “bad” accounts in the sample data.

Negative number of WOE would indicate that the specific variable is isolating a higher proportion of “bad” than “good”

Weight of evidence calculation

```r
woe.atr = subset(training_set, select=c("Status","seniorityR","Home","Job","assetsR","finratR","savingsR","incomeR"))

#fac tabel cu toate WOE pentru a le aduna
woesen=woe(Data=woe.atr,"seniorityR",FALSE,"Status",length(unique(woe.atr$seniorityR)),Bad=0,Good=1)
woe.atr$seniorityR=as.character(woe.atr$seniorityR)
for (i in seq(1,length(woesen$BIN),1)) {woe.atr$seniorityR[woe.atr$seniorityR==woesen$BIN[i]]<-woesen$WOE[i]}
woe.atr$seniorityR=as.numeric(woe.atr$seniorityR)
woeh=woe(Data=woe.atr,"Home",FALSE,"Status",length(unique(woe.atr$Home)),Bad=0,Good=1)
woe.atr$Home=as.character(woe.atr$Home)
for (i in seq(1,length(woeh$BIN),1)) {woe.atr$Home[woe.atr$Home==woeh$BIN[i]]<-woeh$WOE[i]}
woe.atr$Home=as.numeric(woe.atr$Home)
woej=woe(Data=woe.atr,"Job",FALSE,"Status",length(unique(woe.atr$Job)),Bad=0,Good=1)
woe.atr$Job=as.character(woe.atr$Job)
for (i in seq(1,length(woej$BIN),1)) {woe.atr$Job[woe.atr$Job==woej$BIN[i]]<-woej$WOE[i]}
woe.atr$Job=as.numeric(woe.atr$Job)
woeas=woe(Data=woe.atr,"assetsR",FALSE,"Status",length(unique(woe.atr$assetsR)),Bad=0,Good=1)
woe.atr$assetsR=as.character(woe.atr$assetsR)
for (i in seq(1,length(woeas$BIN),1)) {woe.atr$assetsR[woe.atr$assetsR==woeas$BIN[i]]<-woeas$WOE[i]}
woe.atr$assetsR=as.numeric(woe.atr$assetsR)
woefin=woe(Data=woe.atr,"finratR",FALSE,"Status",length(unique(woe.atr$finratR)),Bad=0,Good=1)
woe.atr$finratR=as.character(woe.atr$finratR)
for (i in seq(1,length(woefin$BIN),1)) {woe.atr$finratR[woe.atr$finratR==woefin$BIN[i]]<-woefin$WOE[i]}
woe.atr$finratR=as.numeric(woe.atr$finratR)
woesav=woe(Data=woe.atr,"savingsR",FALSE,"Status",length(unique(woe.atr$savingsR)),Bad=0,Good=1)
woe.atr$savingsR=as.character(woe.atr$savingsR)
for (i in seq(1,length(woesav$BIN),1)) {woe.atr$savingsR[woe.atr$savingsR==woesav$BIN[i]]<-woesav$WOE[i]}
woe.atr$savingsR=as.numeric(woe.atr$savingsR)
woeinc=woe(Data=woe.atr,"incomeR",FALSE,"Status",length(unique(woe.atr$incomeR)),Bad=0,Good=1)
woe.atr$incomeR=as.character(woe.atr$incomeR)
for (i in seq(1,length(woeinc$BIN),1)) {woe.atr$incomeR[woe.atr$incomeR==woeinc$BIN[i]]<-woeinc$WOE[i]}
woe.atr$incomeR=as.numeric(woe.atr$incomeR)
```

For illustrative purpose only

Predictive power of variables – **Weight of evidence (WOE)**

- WOE for the **7 variables** with medium and strong IV: **seniority, home, job, income, assets, financial ratio, savings capacity**
  - **Higher WOE = lower credit risk**
  - **Lower WOE = higher credit risk**

Seniority

> woesen

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	(0,2]	464	557	1021	0.554	0.260	0.343	-75.6	0.222	0.454	0.546
2	(2,5]	156	397	553	0.186	0.185	0.186	-0.5	0.000	0.282	0.718
3	(5,12]	128	548	676	0.153	0.256	0.227	51.5	0.053	0.189	0.811
4	(12,48]	89	640	729	0.106	0.299	0.245	103.7	0.200	0.122	0.878

Home ownership

> woeh

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	rent	9	6	15	0.011	0.003	0.005	-129.9	0.010	0.600	0.400
2	owner	93	105	198	0.111	0.049	0.066	-81.8	0.051	0.470	0.530
3	priv	253	1143	1396	0.302	0.534	0.469	57.0	0.132	0.181	0.819
4	ignore	165	375	540	0.197	0.175	0.181	-11.8	0.003	0.306	0.694
5	parents	54	113	167	0.065	0.053	0.056	-20.4	0.002	0.323	0.677
6	other	263	400	663	0.314	0.187	0.223	-51.8	0.066	0.397	0.603

Job

> woej

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	fixed	369	1480	1849	0.441	0.691	0.621	44.9	0.112	0.200	0.800
2	partime	226	472	698	0.270	0.220	0.234	-20.5	0.010	0.324	0.676
3	freelance	48	77	125	0.057	0.036	0.042	-46.0	0.010	0.384	0.616
4	others	194	113	307	0.232	0.053	0.103	-147.6	0.264	0.632	0.368

Assets

> woeas

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	(0,3000]	548	952	1500	0.655	0.444	0.504	-38.9	0.082	0.365	0.635
2	(3000,6000]	156	631	787	0.186	0.295	0.264	46.1	0.050	0.198	0.802
3	(6000,300000]	133	559	692	0.159	0.261	0.232	49.6	0.051	0.192	0.808

Savings capacity

> woesav

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	(-8.16,1.62]	330	417	747	0.394	0.195	0.251	-70.3	0.140	0.442	0.558
2	(1.62,3.12]	204	537	741	0.244	0.251	0.249	2.8	0.000	0.275	0.725
3	(3.12,5.2]	173	580	753	0.207	0.271	0.253	26.9	0.017	0.230	0.770
4	(5.2,33.2]	130	608	738	0.155	0.284	0.248	60.6	0.078	0.176	0.824

Income

> woeinc

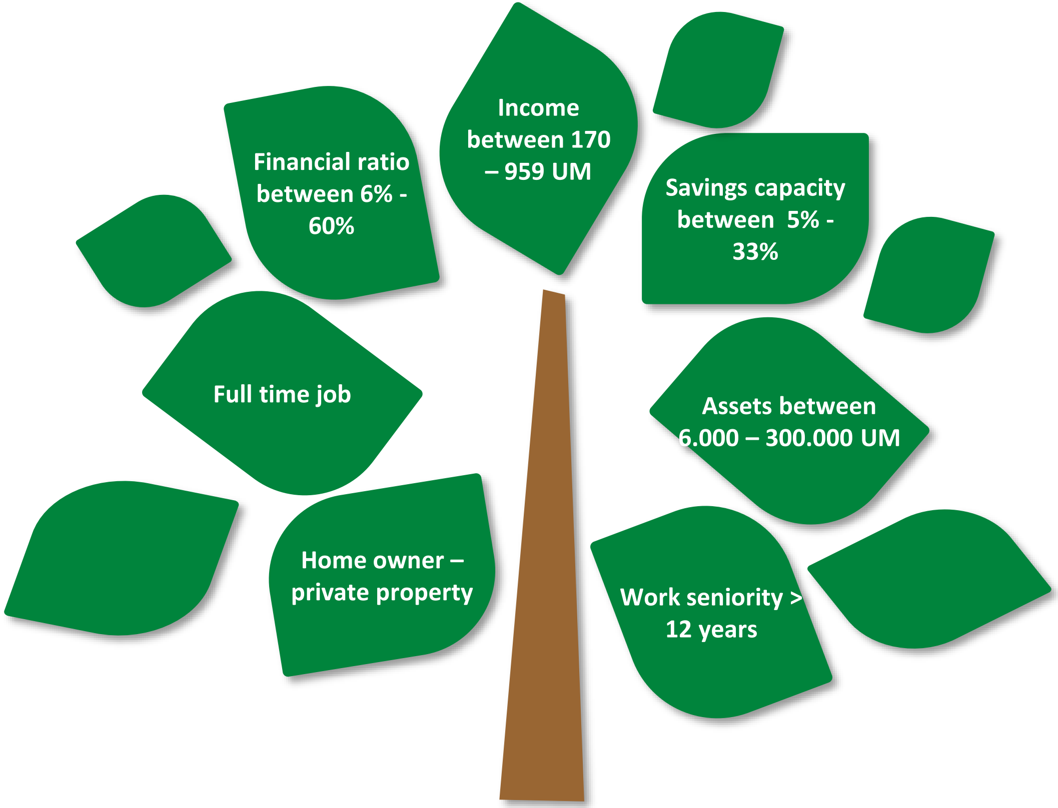
	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	(1,90]	344	452	796	0.411	0.211	0.267	-66.7	0.133	0.432	0.568
2	(90,124]	183	531	714	0.219	0.248	0.240	12.4	0.004	0.256	0.744
3	(124,170]	158	571	729	0.189	0.267	0.245	34.6	0.027	0.217	0.783
4	(170,959]	152	588	740	0.182	0.275	0.248	41.3	0.038	0.205	0.795

Financial ratio

> woefin

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE	IV	BAD_SPLIT	GOOD_SPLIT
1	(6.7,60]	104	649	753	0.124	0.303	0.253	89.3	0.160	0.138	0.862
2	(60,77.1]	183	569	752	0.219	0.266	0.252	19.4	0.009	0.243	0.757
3	(77.1,88.5]	254	470	724	0.303	0.219	0.243	-32.5	0.027	0.351	0.649
4	(88.5,100]	296	454	750	0.354	0.212	0.252	-51.3	0.073	0.395	0.605

Best client profile



ROC curve and Gini

We use the **ROC curve** as a tool to visualize and evaluate the probability for scoring classifiers, for the Learning dataset. The **area under the ROC curve** depicts the accuracy of the classification performance (**the greater the area, the better average performance of classifiers).**

```
gini = 2*N@ROCFunctions["AUROC"][aROCs] - 1
```

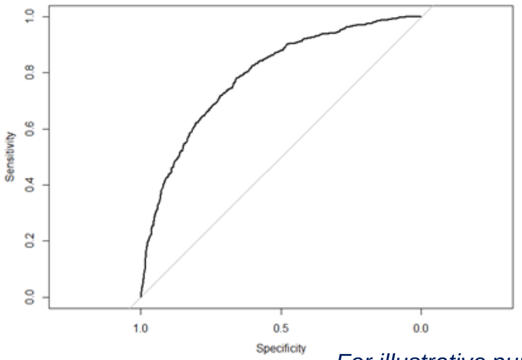
Learning

ROC determined with all **7 variables**: seniority, home, job, income, assets, financial ratio, savings capacity

ROC determination

```
library("pROC")

rocSC<-(woe.atr$seniorityR+woe.atr$Home+woe.atr$Job+woe.atr$assetsR+woe.atr$finratR+woe.atr$savingsR+woe.atr$incomeR)
roc(woe.atr$Status,rocSC,levels=c(0,1))#78.76

rocSC<-(woe.atr$seniorityR+woe.atr$Home+woe.atr$Job+woe.atr$finratR+woe.atr$savingsR)
roc(woe.atr$Status,rocSC,levels=c(0,1))#78.98
plot.roc(woe.atr$Status,rocSC,levels=c(0,1))
```

Higher ROC determined with a selection of **5 variables**, hence the scorecard will be composed out of these most discriminative variables: **seniority, home, job, financial ratio, savings capacity**



For illustrative purpose only

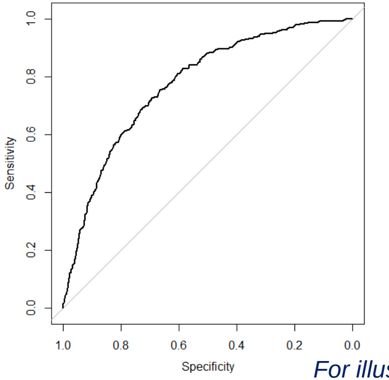
Testing

ROC determined with all **7 variables**: seniority, home, job, income, assets, financial ratio, savings capacity

ROC determination

```
#caclcul ROC values -pe test

rocSC<-(woe.atr.test$seniorityR+woe.atr.test$Home+woe.atr.test$Job+woe.atr.test$assetsR+woe.atr.test$finratR+woe.atr.test$savingsR+woe.atr.test$incomeR)
roc(woe.atr.test$Status,rocSC,levels=c(0,1))#77.15

rocSC<-(woe.atr.test$seniorityR+woe.atr.test$Home+woe.atr.test$Job+woe.atr.test$finratR+woe.atr.test$savingsR)
roc(woe.atr.test$Status,rocSC,levels=c(0,1))#77.36
plot.roc(woe.atr.test$Status,rocSC,levels=c(0,1))
```

Higher ROC determined with a selection of **5 variables**, hence the scorecard will be composed out of these most discriminative variables: **seniority, home, job, financial ratio, savings capacity**



For illustrative purpose only

Total score

- **Scorecard varibles = seniority, home, job, financial ratio, savings capacity**
- **Score = constant (1000) + the sum of WOE** for each of variable that composes the scorecard
- For example:
  - **Highest possible score** = **1000** + 103.7 + 57.0 + 44.9 + 89.3 + 60.6 = **1,355.5**
  - **Lowest possible score** = 1000 – 75.6 – 129.9 – 147.6 – 51.3 – 70.3 = **525.3**

Seniority

```
> woesen
```

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE
1	(0,2]	464	557	1021	0.554	0.260	0.343	-75.6
2	(2,5]	156	397	553	0.186	0.185	0.186	-0.5
3	(5,12]	128	548	676	0.153	0.256	0.227	51.5
4	(12,48]	89	640	729	0.106	0.299	0.245	103.7

Job

```
> woej
```

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE
1	fixed	369	1480	1849	0.441	0.691	0.621	44.9
2	partime	226	472	698	0.270	0.220	0.234	-20.5
3	freelance	48	77	125	0.057	0.036	0.042	-46.0
4	others	194	113	307	0.232	0.053	0.103	-147.6

Savings capacity

```
> woesav
```

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE
1	(-8.16,1.62]	330	417	747	0.394	0.195	0.251	-70.3
2	(1.62,3.12]	204	537	741	0.244	0.251	0.249	2.8
3	(3.12,5.2]	173	580	753	0.207	0.271	0.253	26.9
4	(5.2,33.2]	130	608	738	0.155	0.284	0.248	60.6

Home ownership

```
> woeh
```

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE
1	rent	9	6	15	0.011	0.003	0.005	-129.9
2	owner	93	105	198	0.111	0.049	0.066	-81.8
3	priv	253	1143	1396	0.302	0.534	0.469	57.0
4	ignore	165	375	540	0.197	0.175	0.181	-11.8
5	parents	54	113	167	0.065	0.053	0.056	-20.4
6	other	263	400	663	0.314	0.187	0.223	-51.8

Financial ratio

```
> woefin
```

	BIN	BAD	GOOD	TOTAL	BAD%	GOOD%	TOTAL%	WOE
1	(6.7,60]	104	649	753	0.124	0.303	0.253	89.3
2	(60,77.1]	183	569	752	0.219	0.266	0.252	19.4
3	(77.1,88.5]	254	470	724	0.303	0.219	0.243	-32.5
4	(88.5,100]	296	454	750	0.354	0.212	0.252	-51.3

Machine learning (1)

- **First method – machine learning applied on the sum of WOE** of the five variables in the scorecard

Learning

```r
rocSC<-(woe.atr$seniorityR+woe.atr$Home+woe.atr$Job+woe.atr$finratR+woe.atr$savingsR)
rocSCtest<-(woe.atr.test$seniorityR+woe.atr.test$Home+woe.atr.test$Job+woe.atr.test$finratR+woe.atr.test$savingsR)
rocSC<-rocSC+1000
rocSCtest<-rocSCtest+1000
statustest=woe.atr.test$Status
status<-woe.atr$Status
bigset<-data.frame(rocSC,status,stringsAsFactors = FALSE)
bisettest<-data.frame(rocSCtest,statustest,stringsAsFactors = FALSE)

library(class)
y_pred = knn(train = bigset[1],
             test = bigset[1],
             cl = bigset[,2],
             k = 5,
             prob = TRUE)

# Making the Confusion Matrix
cm = table(bigset[,2], y_pred)
cm
```

```r
library("pROC")
mcptroc<-data.frame(probab=y_pred,status=bigset$status,stringsAsFactors = FALSE)
mcptroc$probab<-as.numeric(as.character(mcptroc$probab))
mcptroc$status<-as.numeric(as.character(mcptroc$status))
roc(mcptroc$status,mcptroc$probab,levels=c(0,1)) #70.87
```

ROC

Testing

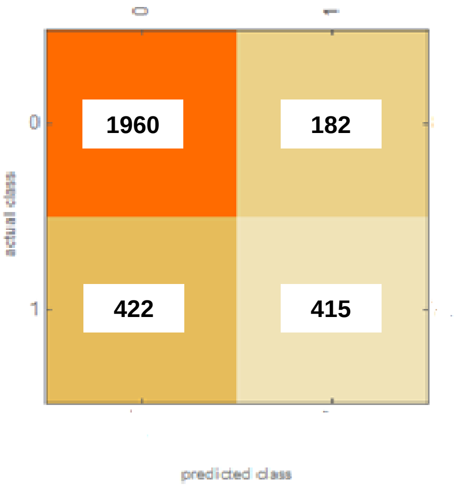
```r
rocSC<-(woe.atr$seniorityR+woe.atr$Home+woe.atr$Job+woe.atr$finratR+woe.atr$savingsR)
rocSCtest<-(woe.atr.test$seniorityR+woe.atr.test$Home+woe.atr.test$Job+woe.atr.test$finratR+woe.atr.test$savingsR)
rocSC<-rocSC+1000
rocSCtest<-rocSCtest+1000
statustest=woe.atr.test$Status
status<-woe.atr$Status
bigset<-data.frame(rocSC,status,stringsAsFactors = FALSE)
bisettest<-data.frame(rocSCtest,statustest,stringsAsFactors = FALSE)

library(class)
y_pred = knn(train = bigset[1],
             test = bisettest[1],
             cl = bigset[,2],
             k = 5,
             prob = TRUE)

# Making the Confusion Matrix
cm = table(bisettest[,2], y_pred)
cm
```

```r
library("pROC")
mcptroc<-data.frame(probab=y_pred,status=bisettest$status,stringsAsFactors = FALSE)
mcptroc$probab<-as.numeric(as.character(mcptroc$probab))
mcptroc$status<-as.numeric(as.character(mcptroc$status))
roc(mcptroc$status,mcptroc$probab,levels=c(0,1)) #63.93
```

ROC

Machine learning (1) – Confusion matrix

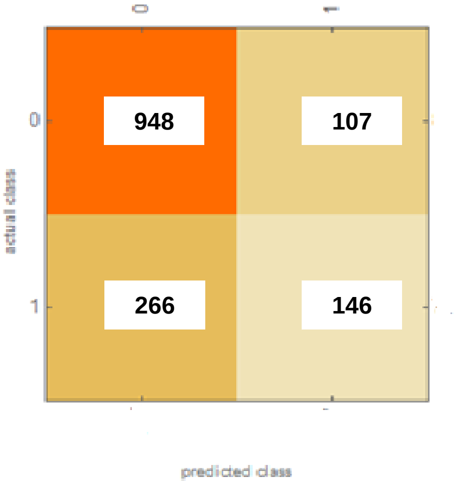
Learning



- **Accuracy=**$\frac{1960+415}{2979} * 100 = 79.73\%$

- **Loss of business potential** - loans predicted to be Bad and they are actually Good - **Type I error – 8.49%** *(182 / (1960 + 182))*
- **Risk appetite** → Loans predicted Good and they are actually Bad - **Type II error – 17.71% (***422 / (1960 + 422))*

Testing



- **Accuracy=**$\frac{948+146}{1476} * 100 = 74.12\%$

- **Loss of business potential** - loans predicted to be Bad and they are actually Good - **Type I error – 10.14%** *(107 / (948 + 107))*
- **Risk appetite** → Loans predicted Good and they are actually Bad - **Type II error – 21.91% (***266/ (948 + 266))*

Machine learning (2)

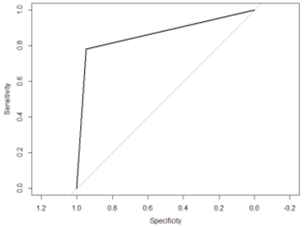
- **Second method – machine learning applied on each of the WOE of** the five variables in the scorecard

Learning

```r
#fac mc pe toate fara adunare
rocSC<-data.frame(training_set$seniorityR,training_set$Home,training_set$Job,training_set$finratR,training_set$savingsR)
status<-training_set$Status
bigset<-data.frame(rocSC,status,stringsAsFactors = FALSE)

rocSCtest<-data.frame(test_set$seniorityR,test_set$Home,test_set$Job,test_set$finratR,test_set$savingsR)
statustest<-test_set$Status
bigsettest<-data.frame(rocSCtest,statustest,stringsAsFactors = FALSE)

#sa fac machine learning pe cele adunate de mai sus cu roc maare
library(class)
y_pred = knn(train = bigset[,-1],
             test = bigset[,-1],
             cl = bigset[,6],
             k = 5,
             prob = TRUE)

# Making the Confusion Matrix
cm = table(bigset[,6], y_pred)
cm
```

```r
library("pROC")
mcptroc<-data.frame(probab=y_pred,status=bigset$status,stringsAsFactors = FALSE)
mcptroc$probab<-as.numeric(as.character(mcptroc$probab))
mcptroc$status<-as.numeric(as.character(mcptroc$status))
bigsettest$status<-as.numeric(as.character(bigsettest$status))
roc(mcptroc$status,mcptroc$probab,levels=c(0,1)) #86.25
```

ROC higher in this method



Testing

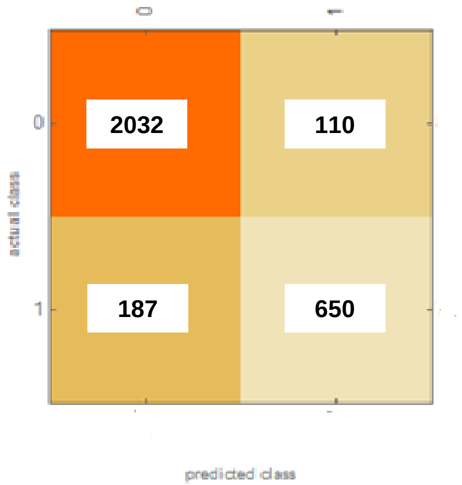
```r
rocSC<-data.frame(training_set$seniorityR,training_set$Home,training_set$Job,training_set$finratR,training_set$savingsR)
status<-training_set$Status
bigset<-data.frame(rocSC,status,stringsAsFactors = FALSE)

rocSCtest<-data.frame(test_set$seniorityR,test_set$Home,test_set$Job,test_set$finratR,test_set$savingsR)
statustest<-test_set$Status
bigsettest<-data.frame(rocSCtest,statustest,stringsAsFactors = FALSE)

#sa fac machine learning pe cele adunate de mai sus cu roc maare
library(class)
y_pred = knn(train = bigset[,-1],
             test = bigsettest[,-1],
             cl = bigset[,6],
             k = 5,
             prob = TRUE)

# Making the Confusion Matrix
cm = table(bigsettest[,6], y_pred)
cm
```

```r
library("pROC")
mcptroc<-data.frame(probab=y_pred,status=bigsettest$status,stringsAsFactors = FALSE)
mcptroc$probab<-as.numeric(as.character(mcptroc$probab))
mcptroc$status<-as.numeric(as.character(mcptroc$status))
roc(mcptroc$status,mcptroc$probab,levels=c(0,1)) #82.09
plot.roc(mcptroc$status,mcptroc$probab,levels=c(0,1))
```

ROC higher in this method

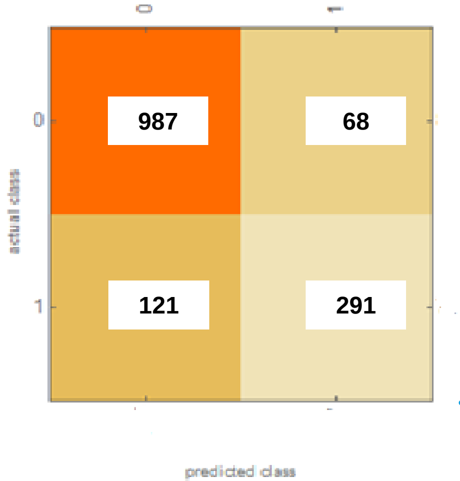
Machine learning (2) – Confusion matrix

Learning



- **Accuracy**=$\frac{2032+650}{2979} * 100 = 90.03\%$
- **Error type 2**=$\frac{187}{2032+187} * 100 = 8.43\%$
- **Error type 1**=$\frac{110}{2032+110} * 100 = 5.14\%$

Testing



- **Accuracy**=$\frac{987+291}{1467} * 100 = 87.12\%$
- **Error type 2**=$\frac{121}{987+121} * 100 = 10.92\%$
- **Error type 1**=$\frac{68}{987+68} * 100 = 6.45\%$

- **Loss of business potential** - loans predicted to be Bad and they are actually Good - **Type I error – 5.13%**
- **Risk appetite** → Loans predicted Good and they are actually Bad - **Type II error – 8.42%**
- ***!! Lower Type I and II errors in this method compared to previous one***

- **Loss of business potential** - loans predicted to be Bad and they are actually Good - **Type I error – 6.44%**
- **Risk appetite** → Loans predicted Good and they are actually Bad - **Type II error – 10.92%**

Takeaway

- Large R community worldwide with resourceful advices:
- https://github.com/gastonstat/CreditScoring
- The specific of R being developed for statistics bring to table various packages to be applied.
- Accessibility in writing code.
- Advantages of Credit scoring brings more structured data.

Contact

Mihai David
mihaidavid87@yahoo.com