Sadržaj

[Uvod 2](#_Toc58777180)

[Učitavanje i analiza podataka 3](#_Toc58777181)

[Učitavanje podataka iz fajla app.csv 3](#_Toc58777182)

[Duplikati 3](#_Toc58777183)

[NA i NULL vrednosti 4](#_Toc58777184)

[Eksplorativna analiza 4](#_Toc58777185)

[Učitavanje podataka iz fajla cb.csv 8](#_Toc58777186)

[Duplikati 8](#_Toc58777187)

[NA i NULL vrednosti 9](#_Toc58777188)

[Eksplorativna analiza 9](#_Toc58777189)

[Spajanje podataka 9](#_Toc58777190)

[Deljenje uzorka po godinama 11](#_Toc58777191)

[Logistička regresija – razvojni uzorak 11](#_Toc58777192)

[Prag za logističku regresiju 13](#_Toc58777193)

[Logistička regresija – test uzorak 14](#_Toc58777194)

[Random forest – razvojni podaci 15](#_Toc58777195)

[Random forest – test podaci 16](#_Toc58777196)

[Deljenje uzorka po nausmičnom odabiru podataka 16](#_Toc58777197)

# Uvod

Ovaj document…

# Učitavanje i analiza podataka

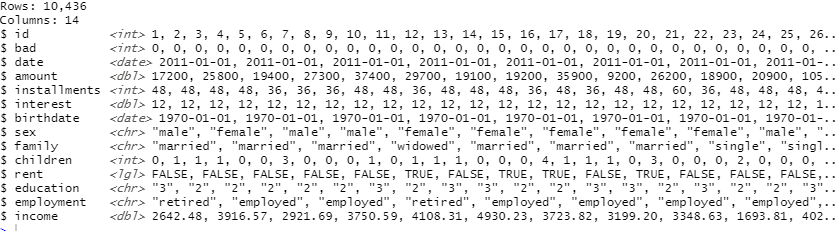
Podaci su učitani iz dva izvora, app.csv i cb.csv (opis podataka je u sekciji 2.3 u dokumentu zadaci.pdf).

## Učitavanje podataka iz fajla app.csv

Prvo smo učitali podatke iz fajla app.csv.

app<-read\_csv("app.csv",col\_types = cols (col\_integer(), col\_integer(), col\_date(), col\_double(), col\_integer(), col\_double(),col\_date(), col\_character(), col\_character(), col\_integer(),col\_logical(), col\_character(), col\_character(), col\_double()))

Pomoću funkcije *glimpse,* vidimo strukturu podataka i da je broj redova 10436.



### Duplikati

Kako bi *id* trebao da ima jedinstvene vrednosti, sledeći korak je da proverimo da li postoje duplikati.

dupl\_id<-app$id[duplicated(app$id)]

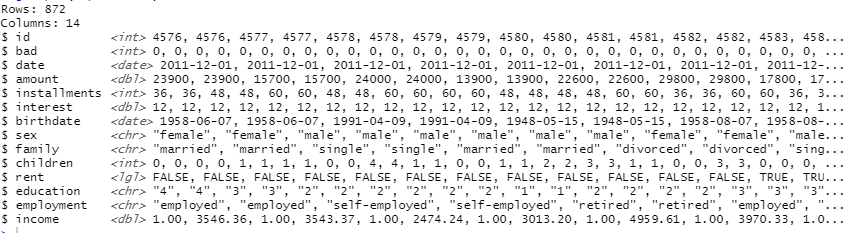
glimpse(dupl\_id)

int [1:436] 4576 4577 4578 4579 4580 4581 4582 4583 4584 4585 ...

Vidimo da ima 436 *id*-eva koji nisu jedinstveni. Sada ćemo filtrirati app samo po tim vrednostima.

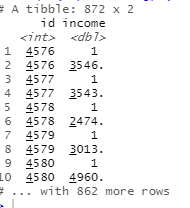
dupl\_values<-filter(app, id %in% dupl\_id)

glimpse(dupl\_values)



Čini se da za svaki *id* postoje dve vrednosti u koloni *income,* jedna od tih vrednosti je 1, kako bismo potvrdili sumnju pogledaćemo koliko ima jedinstvenih vrednosti za kombinaciju *id* i *income*.

unique(dupl\_values[,c('id','income')])



Vidimo zaista da za svaki *id,* *income* ima vrednost 1 i drugu više logičniju vrednost, te ćemo izbaciti vrednosti za ove *id*-eve gde je *income* jednak 1.

app<-filter(app, (id %in% dupl\_id & income !=1) | !(id %in% dupl\_id))

(dupl\_id<-app$id[duplicated(app$id)])

### NA i NULL vrednosti

U ovoj sekciji proverićemo koliko ima *missing values*.

(na\_count <- sapply(app, function(x) sum(is.na(x))))

id bad date amount installments interest birthdate sex family

0 0 0 0 0 0 0 0 0

children rent education employment income

93 0 0 0 0

(null\_count <- sapply(app, function(x) sum(is.null(x))))

id bad date amount installments interest birthdate sex family

0 0 0 0 0 0 0 0 0

children rent education employment income

0 0 0 0 0

Kolona *children* ima mali broj *na* vrednosti. Proverićemo koliko među njima ima klijenata koji su kasnili.

table(filter(app, is.na(children))$bad)

0 1

91 2

Kako ih nema mnogo, možemo te vrednosti da zamenimo sa nulom.

### Eksplorativna analiza

U ovoj sekciji posmatraćemo raspodele promenljivih, kao i njihov odnos sa brojem default-a.

table(app$sex)

female male

4937 5063

aggregate(bad ~ sex, data = app, mean)

sex bad

1 female 0.02167308

2 male 0.02923168

app<-mutate(app, sexMale = as.factor(ifelse(sex == 'male' ,1,0)))

Vidimo da je udeo žena i muškaraca sličan, takođe srednja vrednost za default je slična.

Verovatno ova varijabla neće igrati značajnu ulogu u predviđanju verovatnoće default-a.

Uveli smo novu promenljivu sexMale koja ima vrednost 1 ako je klijent musko, u suprotnom 0.

Kada posmatramo promenljivu *family,* možemo da uočimo određene pravilnosti.

>aggregate(bad ~ family, data = app, mean)

family bad

1 divorced 0.04086957

2 married 0.02075556

3 single 0.03271538

4 widowed 0.02821317

Razvedeni i samci imaju veću stopu default-a. Stoga, možemo da transformišemo ovu promenljivu tako da dobija vrednost 1 kada je klijent razveden ili samac, a u suprotnom 0.

app<-mutate(app, divorced\_single = as.factor(ifelse(family == 'divorced' |family == 'single' ,1,0)))

Klijenti koji iznajmljuju nekretninu u kojoj žive imaju veću stopu default-a.

aggregate(bad ~ rent, data = app, mean)

rent bad

1 FALSE 0.01891313

2 TRUE 0.05074915

Transformisaćemo ovu promenljivu tako da dobija vrednost 1 kada klijent iznajmljuje nekretninu, a 0 u suprotnom slučaju.

app$rent\_1 <-as.factor(ifelse(app$rent, 1, 0))

Kada posmatramo promenljivu koja pokazuje vrstu zaposlenja, vidimo da klijenti koji imaju spopstveni biznis imaju stopu default-a.

aggregate(bad ~ employment, data = app, mean)

1 employed 0.02678324

2 retired 0.01340616

3 self-employed 0.04487179

Stoga, transformisaćemo *employement* tako da dobija vrednost 1 kada klijent ima sopstveni biznis, a 0 u suprotnom.

app<-mutate(app, self\_empl = as.factor(ifelse(employment == 'self-employed',1,0)))

Kada posmatramo obrazovanje u odnosu na stopu default-a, ne možemo da uočimo neku jasnu vezu, te verovatno ova promenljiva neće igrati ulogu u modelu.

> aggregate(bad ~ education, data = app, mean)

education bad

1 1 0.01570681

2 2 0.02131742

3 3 0.03027806

4 4 0.01901141

app$education <-as.factor(app$education)

Posmatrajmo sada promenljive *income* i *amount*.

par(mfrow = c(2, 2))

hist(app$income)

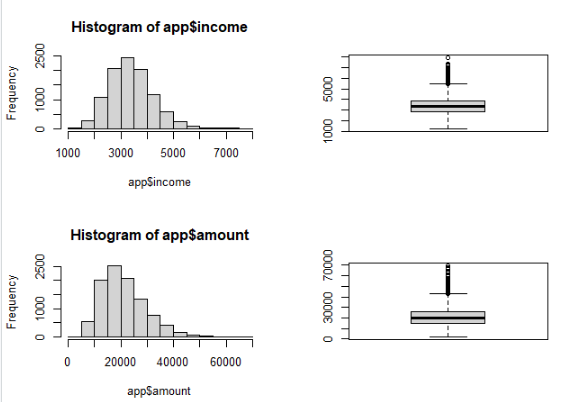
boxplot(app$income)

hist(app$amount)

boxplot(app$amount)

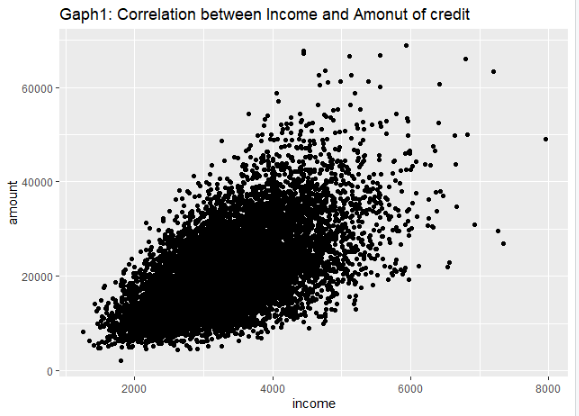
par(mfrow = c(1, 1))

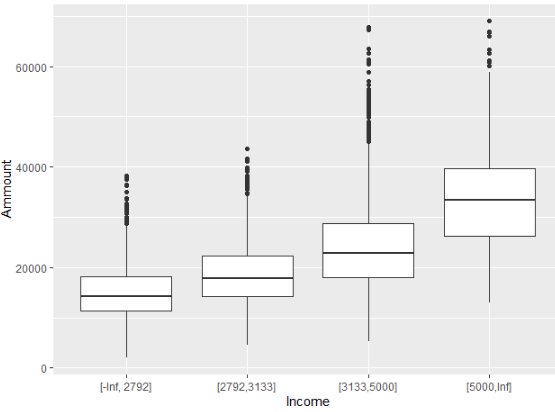
Vidimo da obe promenljive imaju smislene raspodele. Imaju i outilier-e, ali ne previše nelogočne vrednosti.



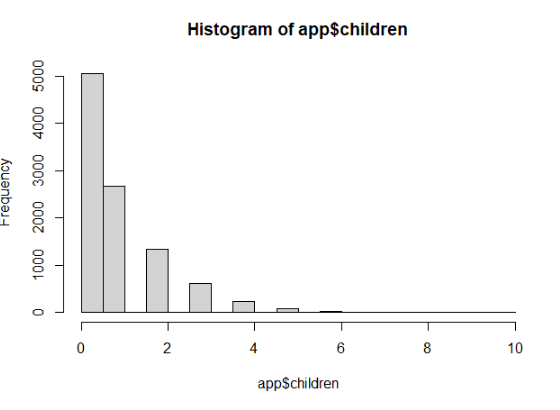
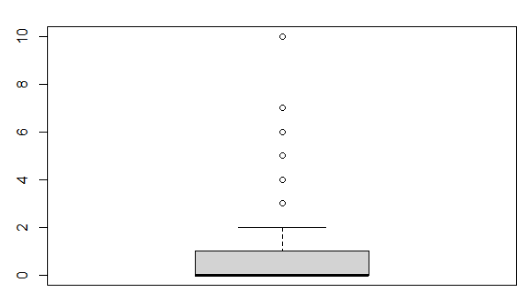
Vrlo je verovatno da su ove dve promenljive povezane. Pogledajmo naredna dva graifka.

Možemo primetiti korelaciju izmedju ove dve promenljive, odnosno da veća primanja dovode do većeg iznosa kredita. Outlier-i nisu toliko nerazumni, npr klijenti sa primanjima u opsegu [3133,5000] podgli su kredit od oko 60 000.





Kad posmatramo broj dece vidimo da je zabeleženo da nekoliko klijenata ima preko sedmoro dece.



aggregate(bad ~ children, data = app, mean)

children bad

1 0 0.01838671

2 1 0.02436282

3 2 0.04282494

4 3 0.03618421

5 4 0.05752212

6 5 0.05263158

7 6 0.03846154

8 7 0.00000000

9 10 0.00000000

table(app$children)

0 1 2 3 4 5 6 7 10

5058 2668 1331 608 226 76 26 6 1

Vidimo da veći broj dece indikuje veću stopu default-a, osim kada klijent ima preko sedmoro dece, što je motiv da pomislimo da ti podaci nisu tačni.

Kako je i boxplot označio ove vrednosti kao outlier-e, sklonićemo ih iz uzorka.

app1<-filter(app, children <=6)

Broj rata izleda da je igra veliku ulogu u proceni stope default-a.

installments bad

1 36 0.02689487

2 48 0.02574686

3 60 0.02366381

Proverili smo, za svaki slučaj da li podaci zaista spadaju u dati vremenski okvir, tj da su podaci iz 2011. i 2012 godine.

summary(app1$date)

Min. 1st Qu. Median Mean 3rd Qu. Max.

"2011-01-01" "2011-06-29" "2011-12-31" "2011-12-30" "2012-06-30" "2012-12-31"

Proverili smo da li godina rođenja ima smislene podatke.

summary(app1$birthdate)

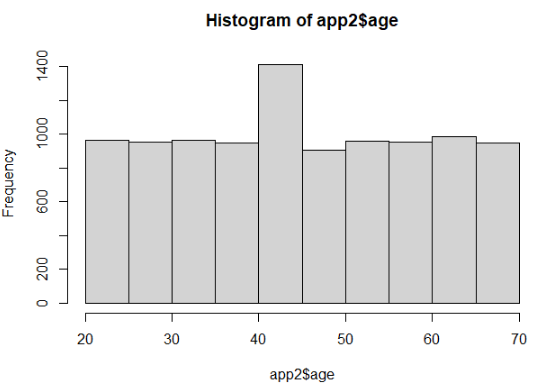
Min. 1st Qu. Median Mean 3rd Qu. Max.

"1941-04-19" "1954-11-29" "1968-05-19" "1967-03-08" "1979-01-13" "1992-11-23"

Uveli smo novu promenljivu *age –* broj godina klijenta u trenutku apliciranja za kredit.

app2<-(mutate(app1, age = as.double(difftime(date,birthdate, units="days")/365)))

hist(app2$age)



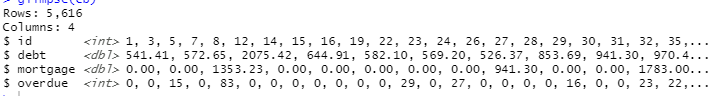
Deluje da su u približnom broju prisutne starosne grupe od 20-70 godina.

## Učitavanje podataka iz fajla cb.csv

U ovoj sekciji učitaćemo podatke za druge kreditne obaveze klijenta i analiziraćemo te podatke.

cb<-read\_csv("cb.csv",col\_types = cols (col\_integer(), col\_double(), col\_double(), col\_integer()))

glimpse(cb)



### Duplikati

Kao i za prvi fajl, proveravamo duplikate za kolonu *id*. U ovom fajlu nema duplikata.

#get duplicated id, if any

(dupl\_cbid<-cb$id[duplicated(cb$id)]) # -> no duplicates

integer(0)

### NA i NULL vrednosti

U ovom fajlu nema NA i NULL vrednosti.

(na\_count <- sapply(cb, function(x) sum(is.na(x))))

id debt mortgage overdue

0 0 0 0

(null\_count <- sapply(cb, function(x) sum(is.null(x))))

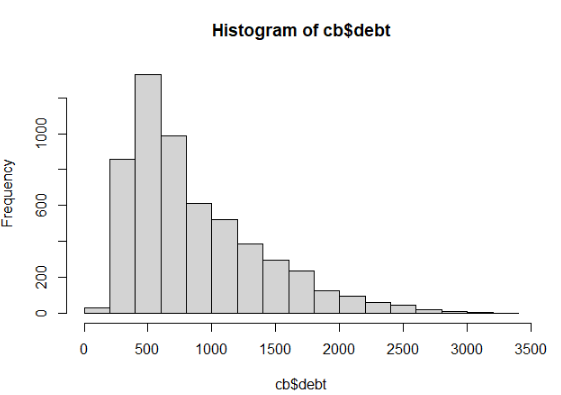
id debt mortgage overdue

0 0 0 0

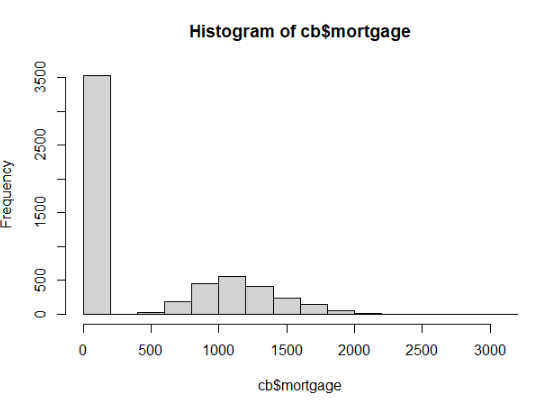
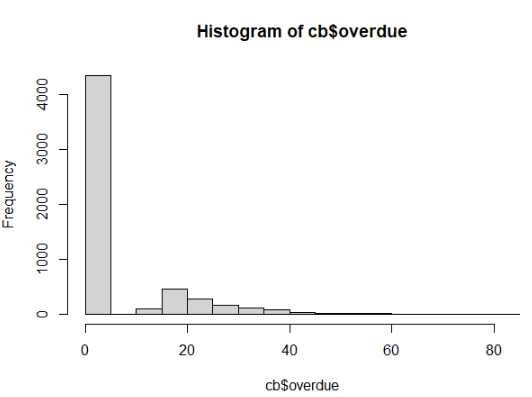
### Eksplorativna analiza

U ovoj sekciji posmatraćemo raspodele promenljivih, kao i njihov odnos sa brojem default-a.

Kada posmatramo promenljivu *debt,* možemo videti da je raspodela smislena.



Promenljive *mortgage* i *overude* nemaju izbalasnirane rapspodele, odnosno medijana i minimum je 0 za obe kolone, ali za sada nećemo ništa isključivati iz uzorka.

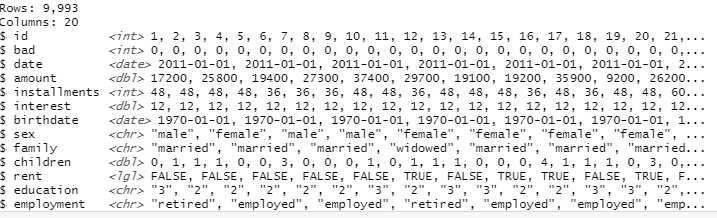
 

## Spajanje podataka

U ovoj sekciji spojićemo podatke iz app.csv i cb.csv i posmatrati podatke.

df<-left\_join(app2, cb, by = "id")

glimpse(df)



Vidimo da ima 4382 reda gde su debt, mortgage i overdue NA, tj da za te klijente nema podataka iz kreditnog biroa.

(na\_count <- sapply(df, function(x) sum(is.na(x))))

id bad date amount installments interest birthdate

0 0 0 0 0 0 0

sex family children rent education employment income

0 0 0 0 0 0 0

rent\_1 self\_empl age debt mortgage overdue

0 0 0 4382 4382 4382

na\_values<-filter(df, is.na(debt))

table(na\_values$bad)

0 1

4311 71

Medju NA vrednostima nema mnogo njih gde je bad=1, tako da ćemo te vrednosti zameniti nulom i kolonu *overdue* ćemo podeliti na 3 klase prema danima kasnjenja.

df <-mutate(df, overdue = ifelse(is.na(df$overdue),0,df$overdue))

df <- mutate(df, overdue\_1 = cut(df$overdue, c(-Inf, 0,40,Inf)))

df <-mutate(df, debt\_1 = ifelse(is.na(df$debt),0,df$debt))

df <-mutate(df, mortgage\_1 = ifelse(is.na(df$mortgage),0,df$mortgage))

#### Kolinearnost

U ovoj sekciji ispitaćemo da li postoji kolienarnost među podacima.

Posmatraćemo prvo podatke iz kreditnog biroa.

cor2pcor(cov(cb))

[,1] [,2] [,3] [,4]

[1,] 1.000000000 -0.009609347 0.007050046 0.003280599

[2,] -0.009609347 1.000000000 0.861929608 -0.007880941

[3,] 0.007050046 0.861929608 1.000000000 0.006088821

[4,] 0.003280599 -0.007880941 0.006088821 1.000000000

Postoji povezanost između *mortgage* i *debt*, pa ćemo isključiti *mortgage* iz dalje analize.

df1<-select(df, -mortgage)

Posmatramo kolinearnost na celom skupu podataka.

test1<-select(df1,amount, income,age,installments,debt\_1, children, overdue)

cor2pcor(cov(test1))

[,1] [,2] [,3] [,4] [,5] [,6] [,7]

[1,] 1.000000000 0.60234302 0.001119749 0.4270728761 0.014580924 -0.004930511 0.0065278098

[2,] 0.602343024 1.00000000 -0.197335847 -0.2489143319 0.148955551 0.064620327 -0.0503289520

[3,] 0.001119749 -0.19733585 1.000000000 -0.0091864260 -0.157463343 0.107009404 0.0237845612

[4,] 0.427072876 -0.24891433 -0.009186426 1.0000000000 -0.016039553 -0.004768982 -0.0002629879

[5,] 0.014580924 0.14895555 -0.157463343 -0.0160395528 1.000000000 0.008354677 0.2366865330

[6,] -0.004930511 0.06462033 0.107009404 -0.0047689823 0.008354677 1.000000000 0.0067923212

[7,] 0.006527810 -0.05032895 0.023784561 -0.0002629879 0.236686533 0.006792321 1.0000000000

Vidimo da su *income* i *installments* povezani sa amount varijablom, te ćemo iz dalje analize isljučiti *income* i *installments*.

df2<-select(df1, c(-income,-installments))

## Deljenje uzorka po godinama

Kada smo završili trasnformaciju/odstranjivanje promenljivih, sada možemo da podelimo uzorak na razvojni i testni kako bismo krenuli sa pravljenjem modela.

train <-filter(df2, year(date)==2011)

test <-filter(df2, year(date)==2012)

glimpse(train) #5009

glimpse(test) #4984

## Logistička regresija – razvojni uzorak

U ovoj sekciji razmatraćemo logističku regresiju na razvojnom uzorku.

Call:

glm(formula = bad ~ sexMale + divorced\_single + children + self\_empl +

rent\_1 + debt\_1 + overdue\_1 + amount + education + age, family = binomial,

data = train)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.3101 -0.2349 -0.1625 -0.1160 3.2670

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -1.979e+01 3.705e+02 -0.053 0.95741

sexMale1 4.094e-01 1.952e-01 2.097 0.03596 \*

divorced\_single1 8.250e-01 2.111e-01 3.908 9.30e-05 \*\*\*

children 3.469e-01 7.376e-02 4.704 2.56e-06 \*\*\*

self\_empl1 6.775e-01 2.626e-01 2.580 0.00987 \*\*

rent\_11 1.431e+00 2.381e-01 6.011 1.85e-09 \*\*\*

debt\_1 9.542e-04 1.632e-04 5.848 4.98e-09 \*\*\*

overdue\_1(0,40] 5.349e-01 2.411e-01 2.219 0.02652 \*

overdue\_1(40, Inf] 1.276e+00 6.173e-01 2.067 0.03876 \*

amount 2.799e-06 1.127e-05 0.248 0.80384

education2 1.361e+01 3.705e+02 0.037 0.97069

education3 1.378e+01 3.705e+02 0.037 0.97034

education4 1.390e+01 3.705e+02 0.038 0.97009

age 4.647e-03 8.690e-03 0.535 0.59283

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1102.87 on 5008 degrees of freedom

Residual deviance: 986.77 on 4995 degrees of freedom

AIC: 1014.8

Vidimo da *amount*, *age* i *education* nisu značajne promenljive za ovaj model, te ćemo ih isključiti.

Call:

glm(formula = bad ~ divorced\_single + children + self\_empl +

rent\_1 + debt\_1 + overdue\_1, family = binomial, data = train)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.1356 -0.2432 -0.1695 -0.1219 3.3459

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -5.5936736 0.2483217 -22.526 < 2e-16 \*\*\*

divorced\_single1 0.8093235 0.2075301 3.900 9.63e-05 \*\*\*

children 0.3477808 0.0719739 4.832 1.35e-06 \*\*\*

self\_empl1 0.6824060 0.2602410 2.622 0.00874 \*\*

rent\_11 1.3917400 0.2230593 6.239 4.39e-10 \*\*\*

debt\_1 0.0009331 0.0001520 6.141 8.23e-10 \*\*\*

overdue\_1(0,40] 0.5491749 0.2395179 2.293 0.02186 \*

overdue\_1(40, Inf] 1.2037486 0.6169165 1.951 0.05103 .

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1102.87 on 5008 degrees of freedom

Residual deviance: 996.67 on 5001 degrees of freedom

AIC: 1012.7

Number of Fisher Scoring iterations: 7

Vidimo da su sve promenljive značajne za model.

Sada ćemo da izračunamo verovatnoće default-a.

train\_lr<- mutate(train, p = predict(train\_model1, type = "response"))

mean(train\_lr $p)

0.02315832

summary(train$p)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.003707 0.007405 0.014746 0.023158 0.030293 0.475248

table(train\_lr$bad)

0 1

4893 116 #neizbalansiran broj onih koji su kasnili sa kreditom i onih koji nisu

Kada smo razdvojili verovatnoće na klase vidimo da svaku klasu model preceni verovatnoću default-a.

train\_lr$p\_class <-cut(train\_lr$p, 5)

table(p\_class)

merge(aggregate(bad ~ p\_class, data = train\_lr,function(x) c(length = length(x), sum = sum(x)))

,aggregate(p ~ p\_class, data = train\_lr, mean))

p\_class bad.length bad.sum p

1 (0.00324,0.098] 4901 101 0.02037069

2 (0.098,0.192] 93 13 0.12849074

3 (0.192,0.287] 10 1 0.22052203

4 (0.287,0.381] 2 0 0.30522150

5 (0.381,0.476] 3 1 0.46598609

### Prag za logističku regresiju

Kako bismo našli prag, odnosno koja verdnost verovatnoće nam govori da će klijent doći do defaulta-a, posmatrajmo matricu konfuzije za različite pragove.

Ukoliko kao prag stavimo 0.04, odnosno da odlučimo da odbijemo klijente koji imaju veću ili jednaku verovatnoću, a prihvatimo klijente sa manjom verovatnoćom, pogledajmo da li bismo bili u pravu.

train\_lr <- mutate(train\_lr, bad\_predicted = ifelse(p<=0.04,0,1))

# Calculate the model's accuracy

mean(train\_lr $bad == train\_lr $bad\_predicted)

[1] 0.8520663

Vidimo da je preciznost modela 85.52%, što može da nas navede da pomislimo da je model dobar, ali ova informacija nam ne govori koliko smo pogodili da je klijent imao default.

Pogledajmo matircu konfuzije

table(ifelse(train\_lr$bad==1,'Default True','Default False'), train\_lr$bad\_predicted)

0 1

Default False 4214 679

Default True 62 54

Vidimo da dobro pogađamo oni koji zaista neće imati default, ali da od klijenata koji će imati default, sa ovim modelom odbili bi samo 46% njih. Što je u ovom slučaju veliki procenat, ne bi bilo dobro za banku da odobri toliko veliki procenat loših kredita.

**# Sensitivity 0.4655172 - TruePositive / (TruePositive + FalseNegative)**

**54/(62+54)**

**#Specificity 0.8612303 - TrueNegative / (FalsePositive + TrueNegative)**

**4214/(4214+679)**

Pokušajmo da promenimo prag.

ROCRpred = prediction(train\_lr$p, train\_lr$bad)

ROCRperf = performance(ROCRpred, "tpr", "fpr")

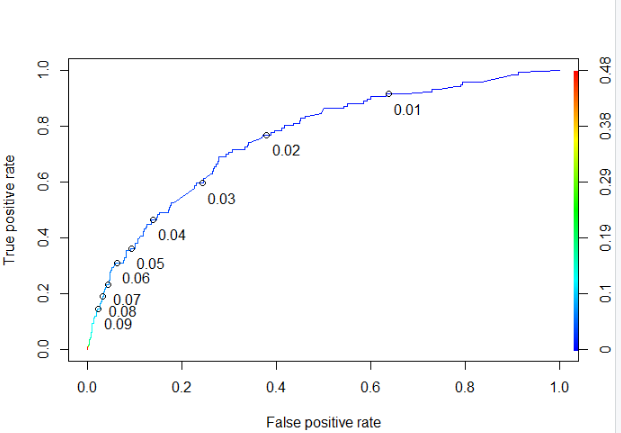
plot(ROCRperf)

# Add colors

plot(ROCRperf, colorize=TRUE)

# Add threshold labels

plot(ROCRperf, colorize=TRUE, print.cutoffs.at=seq(0.01,0.09,by=0.01), text.adj=c(-0.2,1.7))



Kako u ovom slučaju želimo da imamo više onih koji su True Positive, jer je bolje da odbijemo neke klijente koji bi otplatili kredit nego da damo kredit klijentima koji ga neće vratiti, probajmo da kao prag uzmemo 0.02, jer se sa slike čini da bi taj prag bio odgovarajući.

train\_lr1 <- mutate(train\_lr, bad\_predicted = ifelse(p<=0.02,0,1))

mean(train\_lr1$bad == train\_lr1$bad\_predicted) # 0.6013857

table(ifelse(train\_lr1$bad==1,'Default True','Default False'), train1$bad\_predicted)

0 1

Default False 3038 1855

Default True 27 89

**# Sensitivity 0.7672414 - TruePositive / (TruePositive + FalseNegative)**

**89/(27 +89)**

**#Specificity 0.620887 - TrueNegative / (FalsePositive + TrueNegative)**

**3038 /(3038 +1855)**

Vidimo da, iako ovaj model ima manju preciznost, bolji je što se tiče pogađanja klijenata koji ne bi otplatili kredit.

## Logistička regresija – test uzorak

Sada ćemo videti kako se naš model ponaša na podacima koje nije video do sada.

test\_lr <- mutate(test\_lr, p = predict(train\_model1, type = "response",newdata = test))

test\_lr $p <- predict(train\_model\_lr1, type = "response", newdata = test\_lr)

p\_class\_test\_lr <-cut(test\_lr$p, 5)

merge(aggregate(bad ~ p\_class\_test\_lr, data = test,function(x) c(length = length(x), sum = sum(x)))

,aggregate(p ~ p\_class\_test, data = test\_lr, mean))

p\_class\_test bad.length bad.sum p

1 (0.0034,0.0648] 4743 109 0.01799492

2 (0.0648,0.126] 182 17 0.08839180

3 (0.126,0.187] 43 12 0.15138369

4 (0.187,0.248] 11 0 0.20553463

5 (0.248,0.31] 5 1 0.28150375

summary(test\_lr$p)

summary(train\_lr$p)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.003708 0.007405 0.014547 0.022395 0.029131 0.309284

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.003707 0.007405 0.014746 0.023158 0.030293 0.475248

Vidimo da je verovatnoća malo veća sa razvojnim podacima.

Kada koristimo isti prag 0.02, rezultati su sledeći.

test\_lr <- mutate(test\_lr, bad\_predicted = ifelse(p<=0.02,0,1))

# Calculate the model's accuracy

mean(test\_lr $bad == test\_lr $bad\_predicted) #0.635634

table(ifelse(test\_lr $bad==1,'Default True','Default False'), test\_lr$bad\_predicted)

0 1

Default False 3060 1785

Default True 31 108

# Sensitivity 0.7769784 - TruePositive / (TruePositive + FalseNegative)

108/(31+108)

#Specificity 0.6315789 - TrueNegative / (FalsePositive + TrueNegative)

3060 /(3060 +1785)

## Random forest – razvojni podaci

U ovoj sekciji posmatraćemo model napravljen pomoću algoritma random forest.

train\_rf<-mutate(train, bad=as.factor(bad))

test\_rf<-mutate(test, bad=as.factor(bad))

model\_rf <- randomForest(bad ~ sexMale + divorced\_single + children + self\_empl + rent\_1 + debt\_1 + overdue\_1 + amount + education +age,data=train\_rf, importance = TRUE)

Uključili smo one promenljive koje smo isključili iz logističke regresije jer bi random forest trebao sam da proceni koje promenljive nisu značajne.

table(ifelse(train\_rf$bad==1,'Default True','Default False'), train\_rf$p1)

mean(train\_rf$bad == train\_rf$p1) #0.9960072

0 1

Default False 4893 0

Default True 20 96

# Sensitivity 0.7769784 - TruePositive / (TruePositive + FalseNegative)

96/(20 +96) #0.8275862

#Specificity 0.6315789 - TrueNegative / (FalsePositive + TrueNegative)

4893/(4893)

Vidimo da je tačnost modela velika oko 99%, kao i da je veliki postotak True Positive vrednosti.

## Random forest – test podaci

Pogledajmo kako se random forest ponaša na podacima koje nije video pre.

test\_rf<-mutate(test\_rf, p1 = predict(model\_rf, test\_rf,type = "class"))

mean(test\_rf$bad==test\_rf$p1) #0.9721108

table(ifelse(test\_rf$bad==1,'Default True','Default False'), test\_rf$p1)

0 1

Default False 4845 0

Default True 139 0

Vidimo da se random forest jako lose ponaša sa klijentima koju su imali default. Iako je tačnost modela velik, oko 97%, nikako ne bismo smeli da prihvatimo ovo kao model jer bi to značilo da bi banka prihvatila klijente koji bi imali defualt.

## Deljenje uzorka po nausmičnom odabiru podataka

U falju DataScienceRandomSampleSplit\_BSANDO.R smo podeili uzorak slučajnim odabirom u odnosu 65% - 35%. Ponovili smo isti proces kao i kod podele uzorka prema godinama. Rezultati se nisu značajno razlikovali.