Sadržaj

Uvod

Ovaj dokument opisuje proces razvoja modela za verovatnoću default-a.
Raspolagali smo sa podacima iz 2011. i 2012. godine.
Najpre smo odstranili duple vrednosti, zamenili NA vrednosti i odstranili outlier-e gde je imalo smisla.

Razvijali smo dva modela: logističku regresiju i random forest. Podelili smo uzorak na razvojni i test prema dva kriterijuma: godina i slučajni odabir podataka u odnosu 65%-35%. Za svaki kriterijum smo testirali oba modela. Rezultat se nije značajno razlikovao na out of sample i out of time kod logističke regresije, dok kod random forest-a jeste.

Random forest je dao veliku tačnost, 99% na razvojnom i 97% na test uzorku, međutim ispostavilo se da je greška 1. vrste na test uzorku jednka 1. Model bi dozvolio da banka daje kredit klijentima koji bi imali default. Stoga smo odbacili ovaj model.

Logistička regresija sa pragom 0.02 je dala skoro iste rezultate za razvojni i test uzorak. Tačnost logističke regresije je 64%, što možda ne deluje mnogo, ali logistička regresija sa tačnošću od 77% detektuje klijente koji će imati default. U ovom slučaju bolje je da banka odbije dobar kredit, nego da odobri mnogo loših.

Međutim kako True Negative vrednosti ima 63%, možda možemo da se zapitamo da li je ovo ipak dobar model, jer ne želimo da odbijemo 37% klijenata koji bi zaista vratili kredit.
Stoga, bilo bi poželjno ispitati i druge modele.

Propratni dokumenti su kodovi sa komentarima, odnosno R fajlovi : DataScience_BSANDO.R i DataScienceRandomSampleSplit_BSANDO.R.

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.

```
Rows: 10,436
Columns: 14
$ id           <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26..
$ bad          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..
$ date         <date> 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-..
$ amount       <dbl> 17200, 25800, 19400, 27300, 37400, 29700, 19100, 19200, 35900, 9200, 26200, 18900, 20900, 105..
$ installments <int> 48, 48, 48, 48, 36, 36, 36, 48, 48, 36, 48, 48, 48, 36, 48, 36, 48, 48, 60, 36, 48, 48, 48, 4..
$ interest     <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 1..
$ birthdate    <date> 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-..
$ sex          <chr> "male", "female", "male", "male", "female", "female", "female", "female", "female", "male", "..
$ family       <chr> "married", "married", "married", "widowed", "married", "married", "married", "single", "singl..
$ children     <int> 0, 1, 1, 1, 0, 0, 3, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 4, 1, 1, 1, 0, 3, 0, 0, 0, 2, 0, 0, 0, ..
$ rent         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, TRUE, FALSE, TRUE, FALSE, FALSE, FALSE,..
$ education    <chr> "3", "2", "2", "2", "2", "2", "3", "2", "3", "3", "2", "2", "3", "3", "2", "3", "2", "2", "3"..
$ employment   <chr> "retired", "employed", "employed", "retired", "employed", "employed", "employed", "employed",..
$ income       <dbl> 2642.48, 3916.57, 2921.69, 3750.59, 4108.31, 4930.23, 3723.82, 3199.20, 3348.63, 1693.81, 402..
```

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)
```
```
Rows: 872
Columns: 14
$ id           <int> 4576, 4576, 4577, 4577, 4578, 4578, 4579, 4579, 4580, 4580, 4581, 4581, 4582, 4582, 4583, 458...
$ bad          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ date         <date> 2011-12-01, 2011-12-01, 2011-12-01, 2011-12-01, 2011-12-01, 2011-12-01, 2011-12-01, 2011-12-...
$ amount       <dbl> 23900, 23900, 15700, 15700, 24000, 24000, 13900, 13900, 22600, 22600, 29800, 29800, 17800, 17...
$ installments <int> 36, 36, 48, 48, 60, 60, 48, 48, 60, 60, 60, 60, 48, 48, 48, 48, 60, 60, 36, 36, 60, 60, 36, 3...
$ interest     <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 1...
$ birthdate    <date> 1958-06-07, 1958-06-07, 1991-04-09, 1991-04-09, 1948-05-15, 1948-05-15, 1958-08-07, 1958-08-...
$ sex          <chr> "female", "female", "male", "male", "male", "male", "male", "male", "female", "female", "male...
$ family       <chr> "married", "married", "single", "single", "married", "married", "divorced", "divorced", "sing...
$ children     <int> 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 4, 4, 1, 1, 0, 0, 1, 1, 2, 2, 3, 3, 1, 1, 0, 0, 3, 3, 0, 0, ...
$ rent         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, TRU...
$ education    <chr> "4", "4", "3", "3", "2", "2", "2", "2", "2", "2", "1", "1", "2", "2", "2", "2", "3", "3", "3"...
$ employment   <chr> "employed", "employed", "self-employed", "self-employed", "retired", "retired", "employed", "...
$ income       <dbl> 1.00, 3546.36, 1.00, 3543.37, 1.00, 2474.24, 1.00, 3013.20, 1.00, 4959.61, 1.00, 3970.33, 1.0...
```

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')])
```
```
# A tibble: 872 x 2
      id income
   <int>  <dbl>
 1  4576      1
 2  4576   3546.
 3  4577      1
 4  4577   3543.
 5  4578      1
 6  4578   2474.
 7  4579      1
 8  4579   3013.
 9  4580      1
10  4580   4960.
# ... with 862 more rows
```

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.
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)))
```

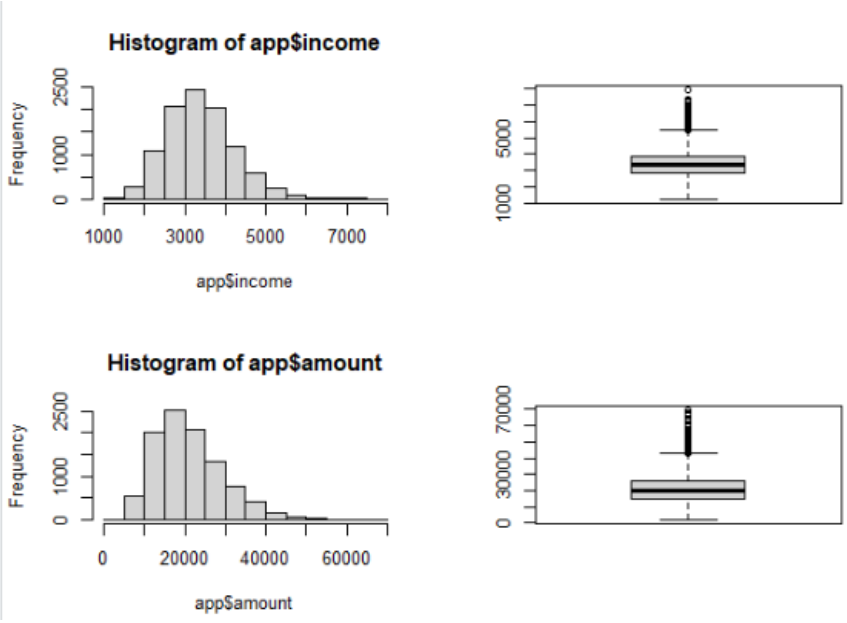
Kada posmatramo obrazovanje u odnosu na stopu default-a, ne možemo da uočimo neku jasnu vezu, te verovatno ova promenljiva neće igrati ulogu u modelu.

```
> 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*.

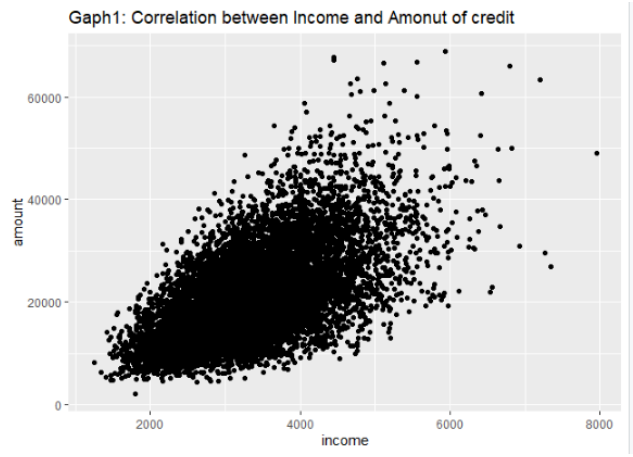
```
par(mfrow = c(2, 2))
hist(app$income)
boxplot(app$income)
hist(app$amount)
boxplot(app$amount)
par(mfrow = c(1, 1))
```

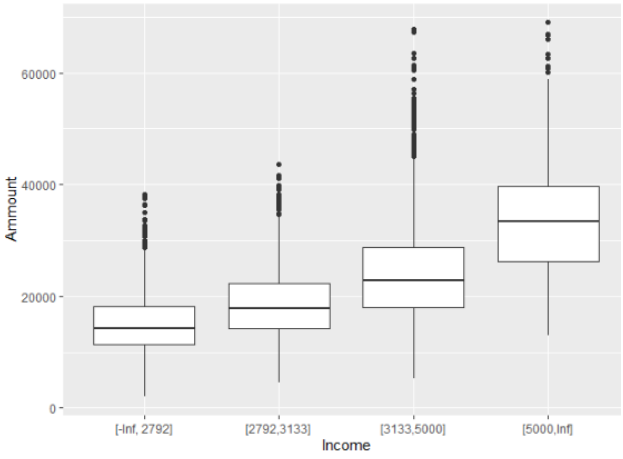
Vidimo da obe promenljive imaju smislene raspodele. Imaju i outilier-e, ali ne previše nelogične vrednosti.



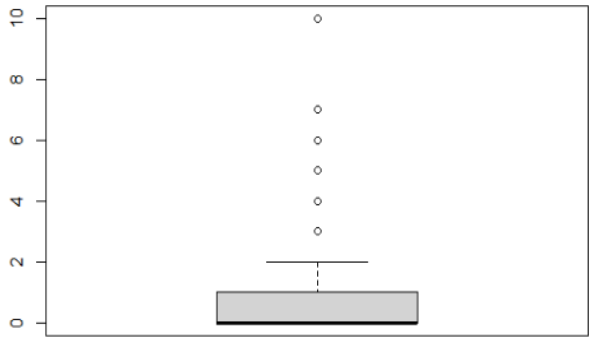
Vrlo je verovatno da su ove dve promenljive povezane. Pogledajmo naredna dva graifka.

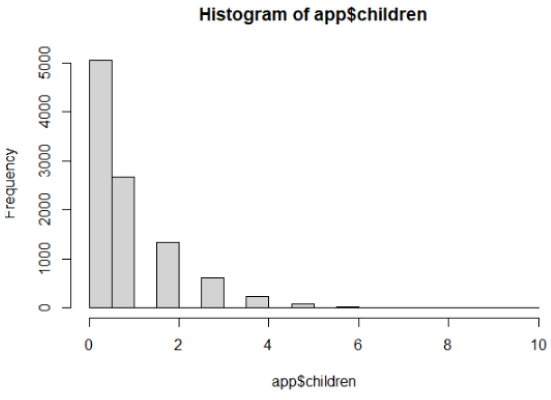
Možemo primetiti korelaciju izmedju ove dve promenljive, odnosno da veća primanja dovode do većeg iznosa kredita. Outlier-i nisu toliko nerazumni, npr klijenti sa primanjima u opsegu [3133,5000] podgli su kredit od oko 60 000.





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.

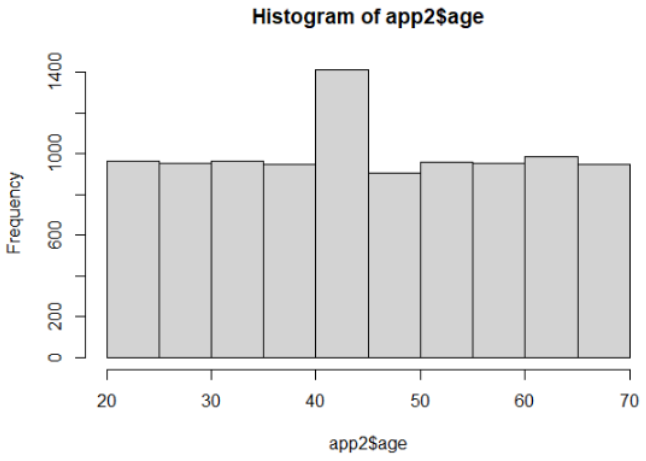
```
summary(app1$date)
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
"2011-01-01" "2011-06-29" "2011-12-31" "2011-12-30" "2012-06-30" "2012-12-31"
```

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)
Rows: 5,616
Columns: 4
$ id       <int> 1, 3, 5, 7, 8, 12, 14, 15, 16, 19, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32, 35,...
$ debt     <dbl> 541.41, 572.65, 2075.42, 644.91, 582.10, 569.20, 526.37, 853.69, 941.30, 970.4...
$ mortgage <dbl> 0.00, 0.00, 1353.23, 0.00, 0.00, 0.00, 0.00, 0.00, 941.30, 0.00, 0.00, 1783.00...
$ overdue  <int> 0, 0, 15, 0, 83, 0, 0, 0, 0, 0, 0, 0, 29, 0, 27, 0, 0, 0, 0, 16, 0, 0, 23, 22,...
```

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

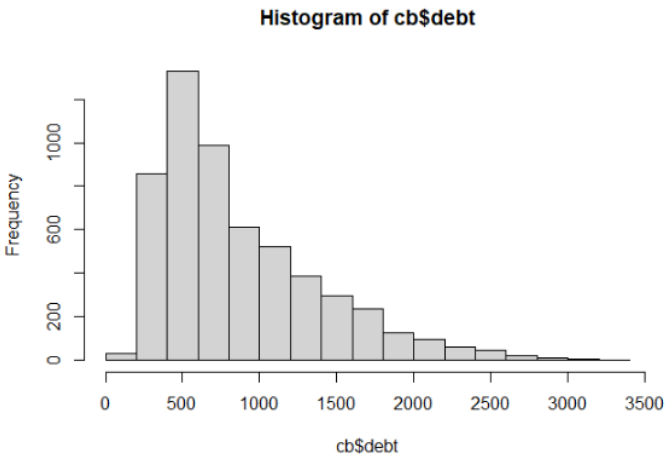
U ovom fajlu nema NA i NULL vrednosti.

```
 (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
```

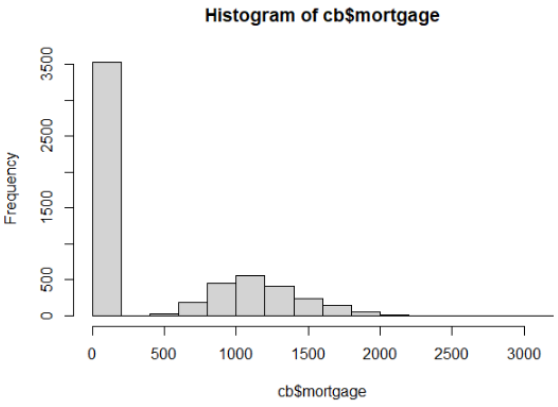
Eksplorativna analiza

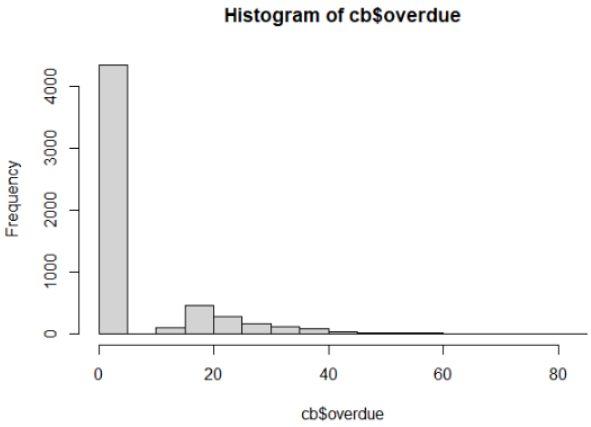
U ovoj sekciji posmatraćemo raspodele promenljivih, kao i njihov odnos sa brojem default-a.

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)
```

```
Rows: 9,993
Columns: 20
$ id           <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,...
$ bad          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ date         <date> 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2011-01-01, 2...
$ amount       <dbl> 17200, 25800, 19400, 27300, 37400, 29700, 19100, 19200, 35900, 9200, 26200...
$ installments <int> 48, 48, 48, 48, 36, 36, 36, 48, 48, 36, 48, 48, 48, 36, 48, 36, 48, 48, 60...
$ interest     <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12...
$ birthdate    <date> 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 1...
$ sex          <chr> "male", "female", "male", "male", "female", "female", "female", "female", ...
$ family       <chr> "married", "married", "married", "widowed", "married", "married", "married"...
$ children     <dbl> 0, 1, 1, 1, 0, 0, 3, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 4, 1, 1, 1, 0, 3, 0,...
$ rent         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, TRUE, FALSE, TRUE, F...
$ education    <chr> "3", "2", "2", "2", "2", "2", "3", "2", "3", "3", "2", "2", "3", "3", "2",...
$ employment   <chr> "retired", "employed", "employed", "retired", "employed", "employed", "emp...
```

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 test 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 ~ sexMale + divorced_single + children + self_empl +
    rent_1 + debt_1 + overdue_1, family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.2142  -0.2320  -0.1623  -0.1159   3.3076

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -5.8188604  0.2758760 -21.092  < 2e-16 ***
sexMale1          0.4051087  0.1948932   2.079   0.0377 *
divorced_single1  0.8194438  0.2081597   3.937 8.26e-05 ***
children          0.3530242  0.0722264   4.888 1.02e-06 ***
self_empl1        0.6654711  0.2610282   2.549   0.0108 *
rent_11           1.3954418  0.2235381   6.243 4.31e-10 ***
debt_1            0.0009253  0.0001518   6.094 1.10e-09 ***
```

```
overdue_1(0,40]     0.5503324  0.2398831   2.294   0.0218 *
overdue_1(40, Inf]  1.2434276  0.6167676   2.016   0.0438 *
---
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:  992.28  on 5000  degrees of freedom
AIC: 1010.3
```

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 za 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 0.02036444
2   (0.098,0.192]         93      13 0.13016403
3   (0.192,0.287]         10       1 0.20446418
4   (0.287,0.381]          2       0 0.30162307
5   (0.381,0.476]          3       1 0.48025296
```

Prag za logističku regresiju

Kako bismo našli prag, odnosno koja vrednost 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] 0.8562587
```

Vidimo da je preciznost modela 85.56%, 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 4236  657
 Default True    63   53
```

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.

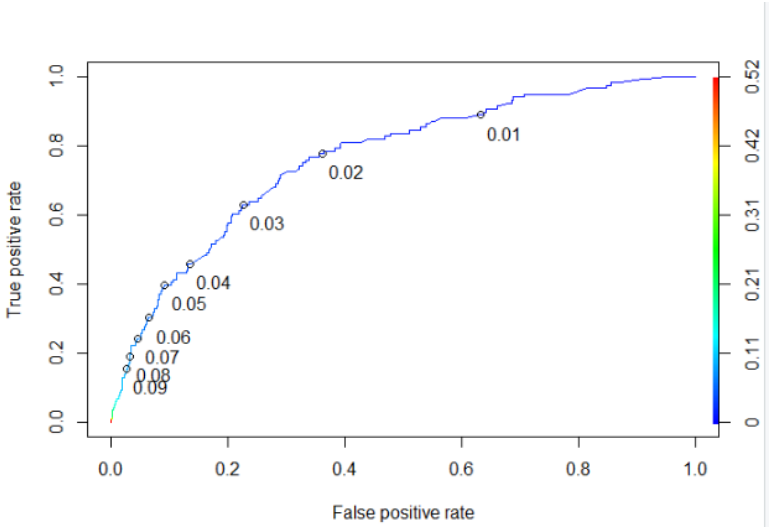
```
# Sensitivity 0.4568966 - TruePositive / (TruePositive + FalseNegative)
53/(63+53)

#Specificity 0.8657265 - TrueNegative / (FalsePositive + TrueNegative)
4236  /(4236+657)
```

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) # 0.6412458
table(ifelse(train_lr1$bad==1,'Default True','Default False'), train1$bad_predicted)
```

```
                    0    1
 Default False 3122 1771
 Default True    26   90
```

 # Sensitivity 0.7758621 - TruePositive / (TruePositive + FalseNegative)
90/(26+90)

#Specificity 0.6380544 - TrueNegative / (FalsePositive + TrueNegative)
3122 /(3122 +1771)

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 (0.00261,0.0736]      4776     112 0.01837484
2   (0.0736,0.144]       169      20 0.10034934
3    (0.144,0.215]        32       6 0.17455995
4    (0.215,0.285]         4       0 0.23694455
5    (0.285,0.356]         3       1 0.32143267

summary(test_lr$p)
summary(train_lr$p)
     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.002962 0.006697 0.013840 0.022515 0.026933 0.355943
  Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.002962 0.006842 0.013893 0.023158 0.028326 0.521510
```

Vidimo da je verovatnoća malo veća sa razvojnim podacima.
Kada koristimo isti prag 0.02, rezultati su sledeći. Tačnost i True Positive vrednosti se nisu mnogo promenile u odnosu na razvnojni model, što znači da nije došlo do overfitting-a.

```
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) # 0.6494783
table(ifelse(test_lr $bad==1,'Default True','Default False'), test_lr$bad_predicted)
                    0    1
 Default False 3133 1712
 Default True    35  104
```

 # Sensitivity 0.7482014 - TruePositive / (TruePositive + FalseNegative)
104/(35+104)

#Specificity 0.646646- TrueNegative / (FalsePositive + TrueNegative)
3133 /(3133 +1712)

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.