

Raport

Jakub Kosterna, Marcin Łukaszyk, Mikołaj Malec

15/04/2020

1. Ogólnie

German credit data to bardzo ładny zbiór danych pod naukę uczenia maszynowego. Jest on względnie nieduży, gdyż zawiera 1000-czną próbkę osób ubiegających się o kredyt, jednak jest przy tym wydaje się reprezentatywny (dane zdają się dobrze odzwierciedlać populację) i zawiera dużo informacji na temat każdego klienta.

W ciągu ostatnich tygodni pierwszorzędnie dobrze zapoznaliśmy się z daną ramką danych, następnie przygotowaliśmy ją pod odpalania algorytmów machine learning, żeby na końcu wybrać ten jeden fajny model i go przetestować.

2. Czystczenie

Pierwotna wersja data frame nie była najweselsza na świecie - zamiast ludzkich liczb czy jasnych wartości typu faktor, mieliśmy do czynienia z chaosem w postaci **dziwnych oznaczeń** takich jak widać na załączonym obrazku:

checking_account_status	duration	credit_history	purpose	credit_amount	savings	present_employment	installment_rate	personal	other_debtors	present_residence	property	age	
A12	30	A34	A40	4249	A61	A71		4	A94	A101	2	A123	28
A11	18	A32	A42	1131	A61	A71		4	A92	A101	2	A123	33
A12	24	A32	A43	1967	A61	A75		4	A92	A101	4	A123	20
A11	12	A32	A42	1657	A61	A73		2	A93	A101	2	A121	27
A11	18	A32	A43	1882	A61	A73		4	A92	A101	4	A123	25
A14	9	A32	A49	1449	A61	A74		3	A92	A101	2	A123	27
A14	9	A32	A42	1313	A61	A75		1	A93	A101	4	A123	20
A12	42	A34	A49	5954	A61	A74		2	A92	A101	1	A121	41
A14	30	A32	A43	1867	A65	A75		4	A93	A101	4	A123	58
A12	12	A32	A40	1223	A61	A75		1	A91	A101	1	A121	46

Z pomocą przyszła **dokumentacja**, która rozwiązała wszelkie możliwości. W celu dalszej pracy z naszymi danymi, podmieniliśmy skrótowe identyfikatory na ciągi znaków przyjazne użytkownikowi.

2. Czyszczenie danych

W celu ludzkiego przedstawienia danych zmodyfikujemy je tak, żeby wszystko stało się jasne.

```
# przekształcanie na dane numeryczne i z faktoremami
levels(data[,1]) <- c("low", "fair", "high", "not_have") #DM Low<0<fair<200<high
levels(data[,3]) <- c("all_paid", "all_paid_here", "paid_till_now", "delay", "critical")
levels(data[,4]) <- c("new_car", "used_car", "furniture/equipment", "radio/television", "domestic", "repairs", "education",
"retraining", "business", "other") #note: 0 for vacation
levels(data[,6]) <- c("low", "normal", "high", "very_high", "not_have/unknown") #DM Low<100<normal<500<high<1000<very_high
levels(data[,7]) <- c("unemployed", "less_than_year", "1-3_years", "4-6_yeras", "7+_years")
levels(data[,9]) <- c("male_d/s", "female_d/s/m", "male_single", "male_m/w") #d = divorced, s = seperated, m = married, w =
widowed ,#note: 0 female single
levels(data[,10]) <- c("none", "co-applicant", "guarantor")
levels(data[,12]) <- c("real_estate", "building_savings", "car", "not_have/unknown")
levels(data[,14]) <- c("bank", "stores", "none")
levels(data[,15]) <- c("rent", "own", "for_free")
levels(data[,17]) <- c("unskilled_non_resident", "unskilled_resident", "skilled_employee", "highly_qualified_employee") # al
so management, self-employed, officer
levels(data[,19]) <- c("no", "yes")
levels(data[,20]) <- c("yes", "no")
data[,21] <- as.factor(as.character(data[,21]))
levels(data[,21]) <- c("Good", "Bad")
```

Jak teraz wygląda nasze losowe 20 wierszy?

Końcowy efekt zaprezentował się następująco:

checking_account_status	duration	credit_history	purpose	credit_amount	savings	present_employment	installment_rate	personal	other_debtors	present_residence
low	15	paid_till_now	new_car	3959	low	1-3_years	3	female_d/s/m	none	2
fair	15	paid_till_now	new_car	2631	normal	1-3_years	2	female_d/s/m	none	4
low	18	critical	education	1190	low	unemployed	2	female_d/s/m	none	4
fair	18	delay	radio/television	4297	low	7+_years	4	male_d/s	none	3
not_have	24	paid_till_now	domestic	1311	normal	4-6_yeras	4	male_m/w	none	3
high	15	paid_till_now	retraining	1905	low	7+_years	4	male_single	none	4
fair	10	all_paid_here	domestic	1048	low	1-3_years	4	male_single	none	4
fair	15	paid_till_now	education	1308	low	7+_years	4	male_single	none	4
fair	27	critical	domestic	2520	high	1-3_years	4	male_single	none	2
not_have	24	critical	radio/television	1585	low	4-6_yeras	4	male_single	none	3
low	24	critical	domestic	1231	very_high	7+_years	4	female_d/s/m	none	4
fair	6	paid_till_now	education	454	low	less_than_year	3	male_m/w	none	1

Wielkim szczęściem okazał się za to fakt, że nasza ramka danych **nie zawierała braków ani niepokojących outlierów**.

3. Eksploracja

... dla tak przyjemnych i życiowych danych była czystą przyjemnością.

Bardzo pomocny okazało się narzędzie **DataExplorer**, które pokazało wiele ciekawych zależności w tabeli automatycznie.

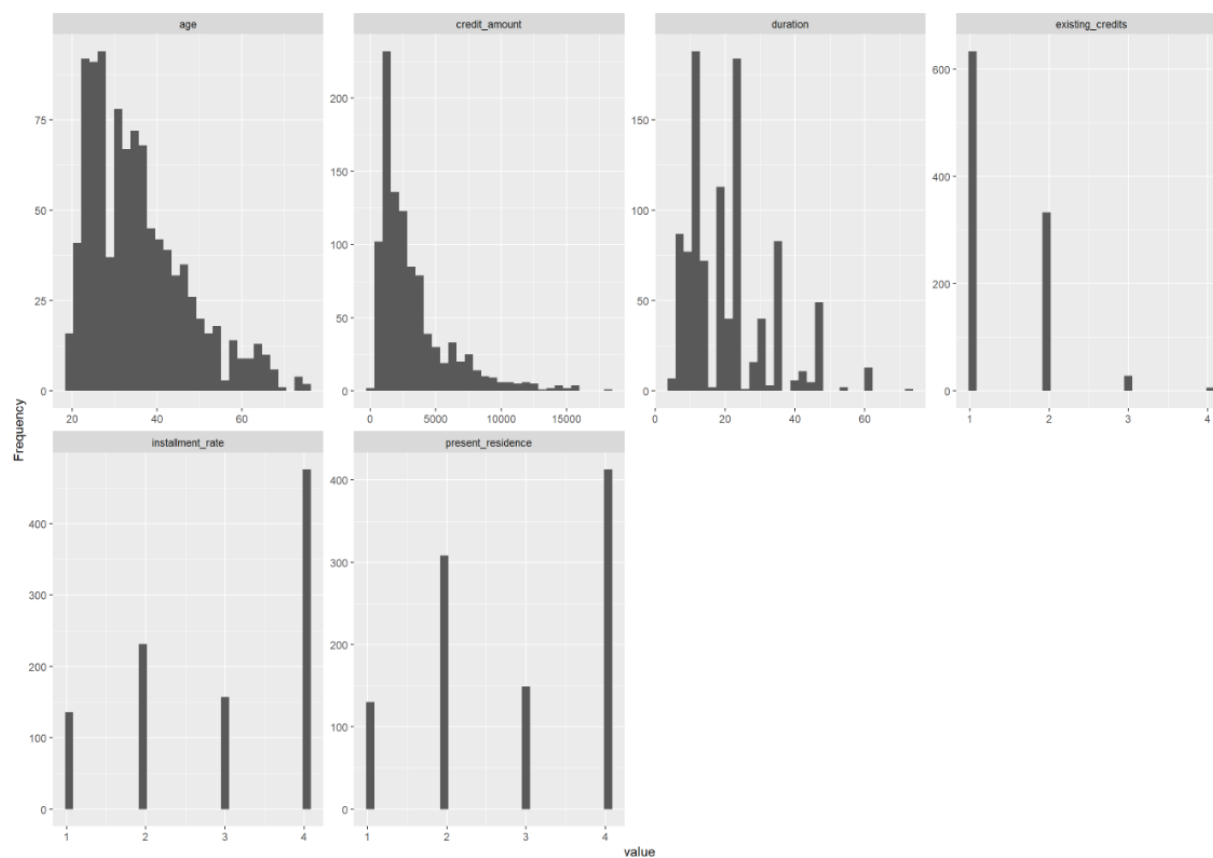
Elegancki ogląd naszych danych otrzymaliśmy dzięki zaoferowanemu przez funkcję drzewko typów.



Wykresy kolumnowe gęstości występowania danych utwierdziły nas w przekonaniu, że z naszą ramką danych wszystko w porządku.

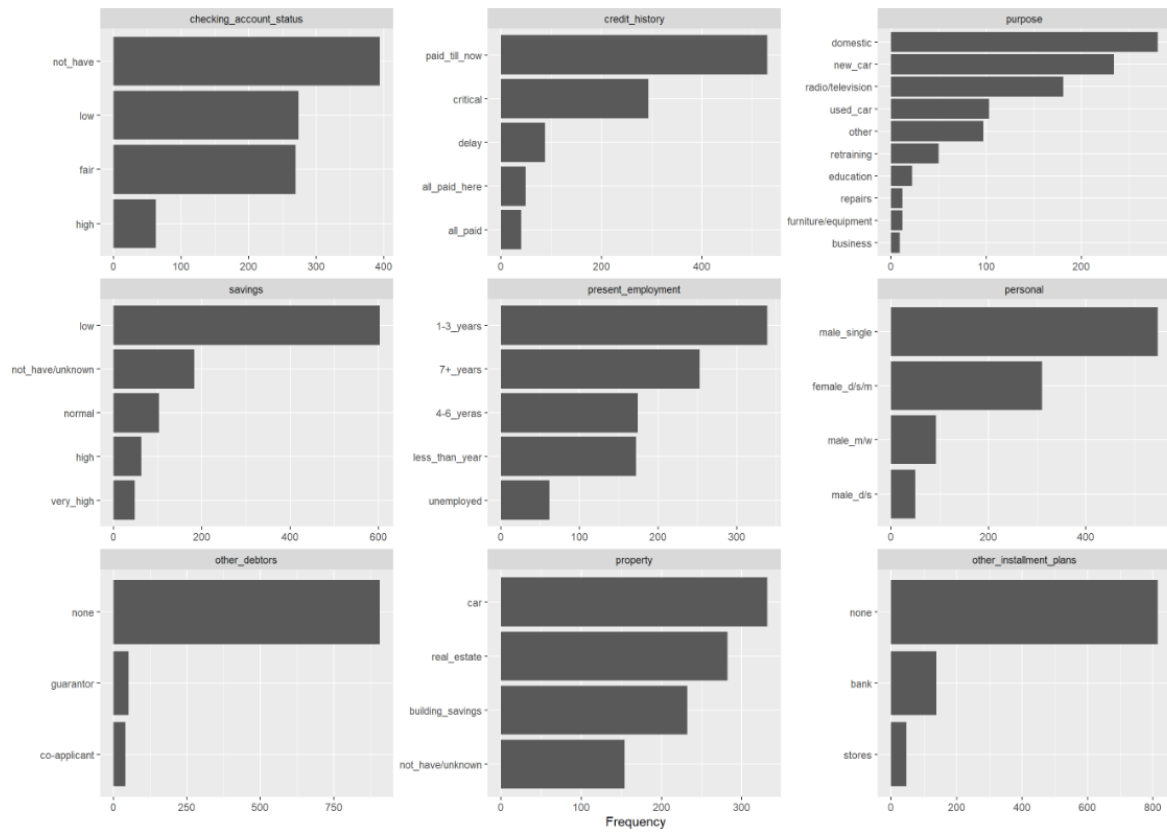
Univariate Distribution

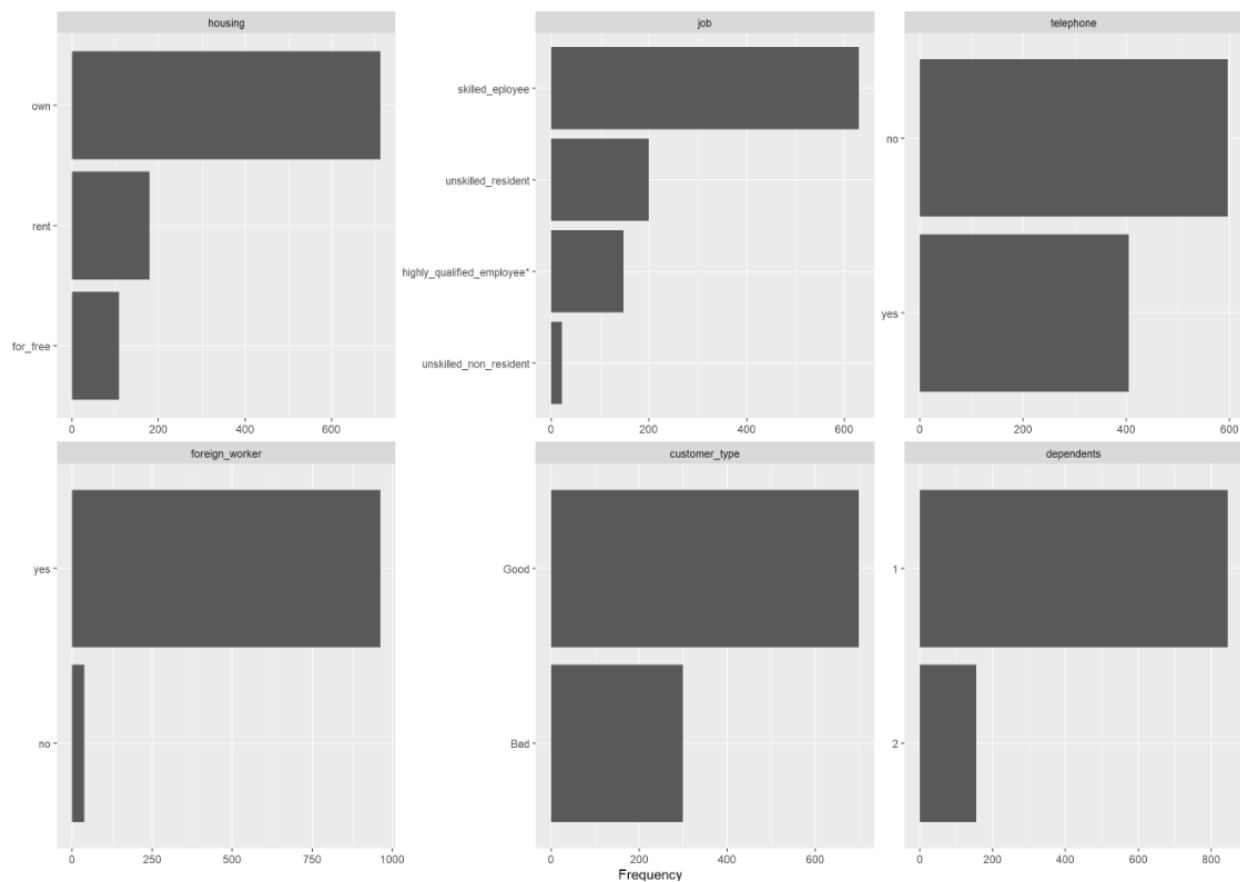
Histogram



Ładny ogłd balansu wartoŃci dały nam **barcharty zmiennych numerycznych**.

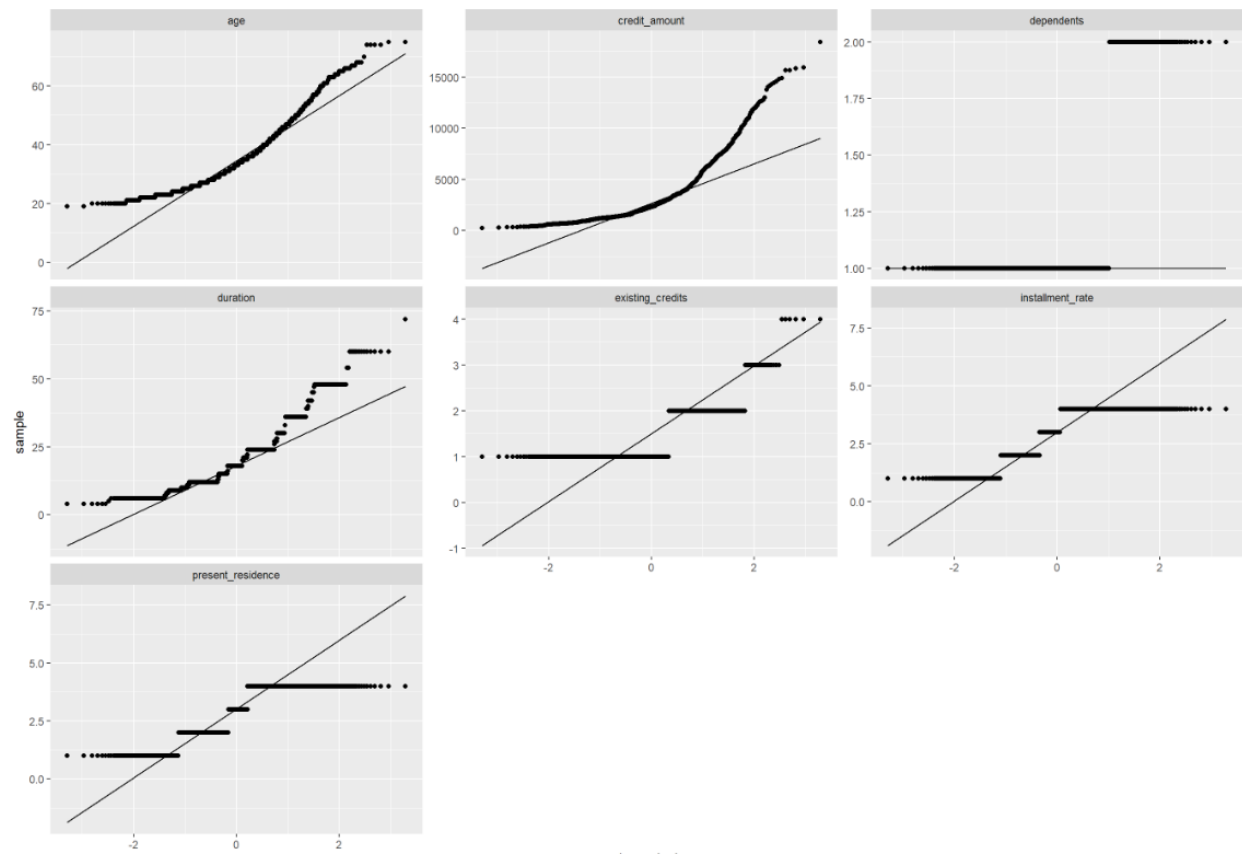
Bar Chart (by frequency)





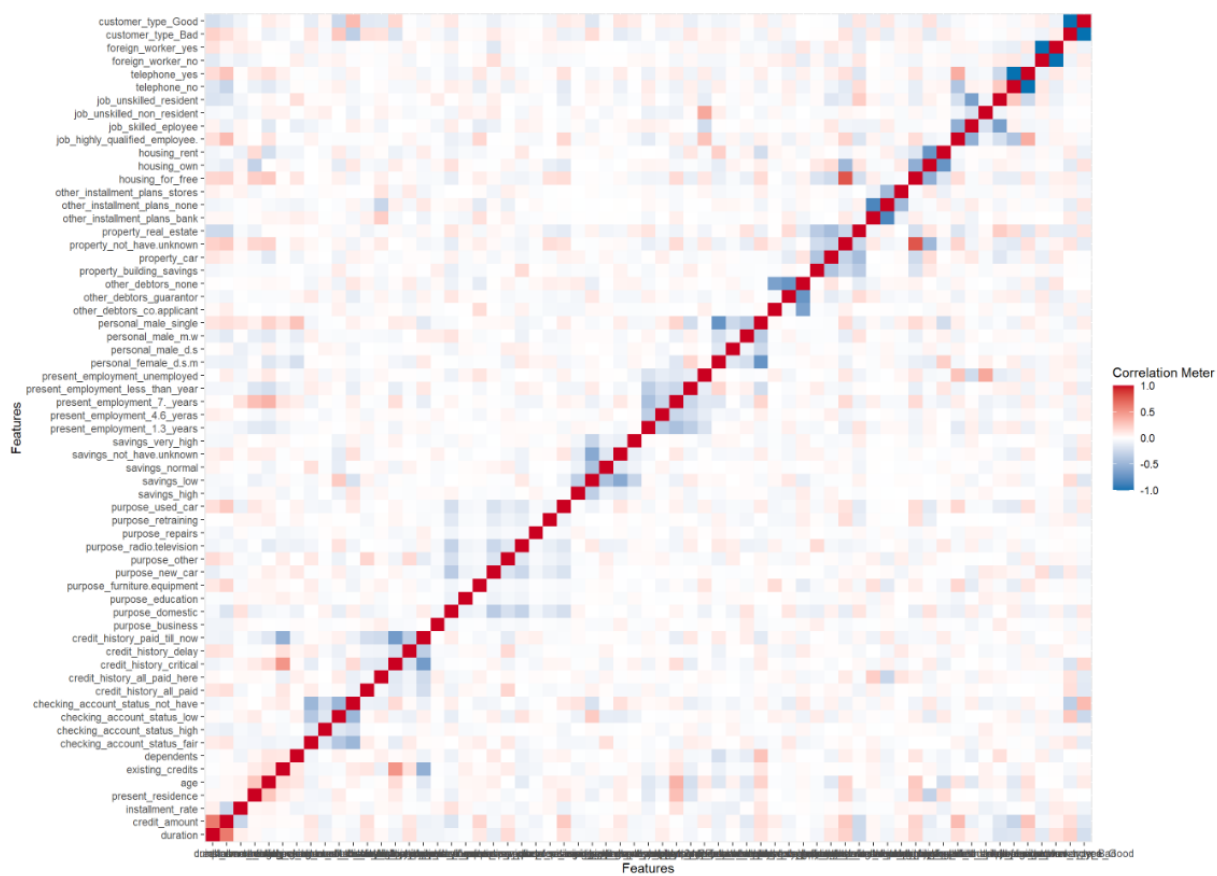
Bardzo przydatna okazała się także wizualizacja **QQplot** - dzięki niej dostaliśmy przystępny obraz wartości liczbowych w naszej dataframe oraz ich rozkład.

QQ Plot



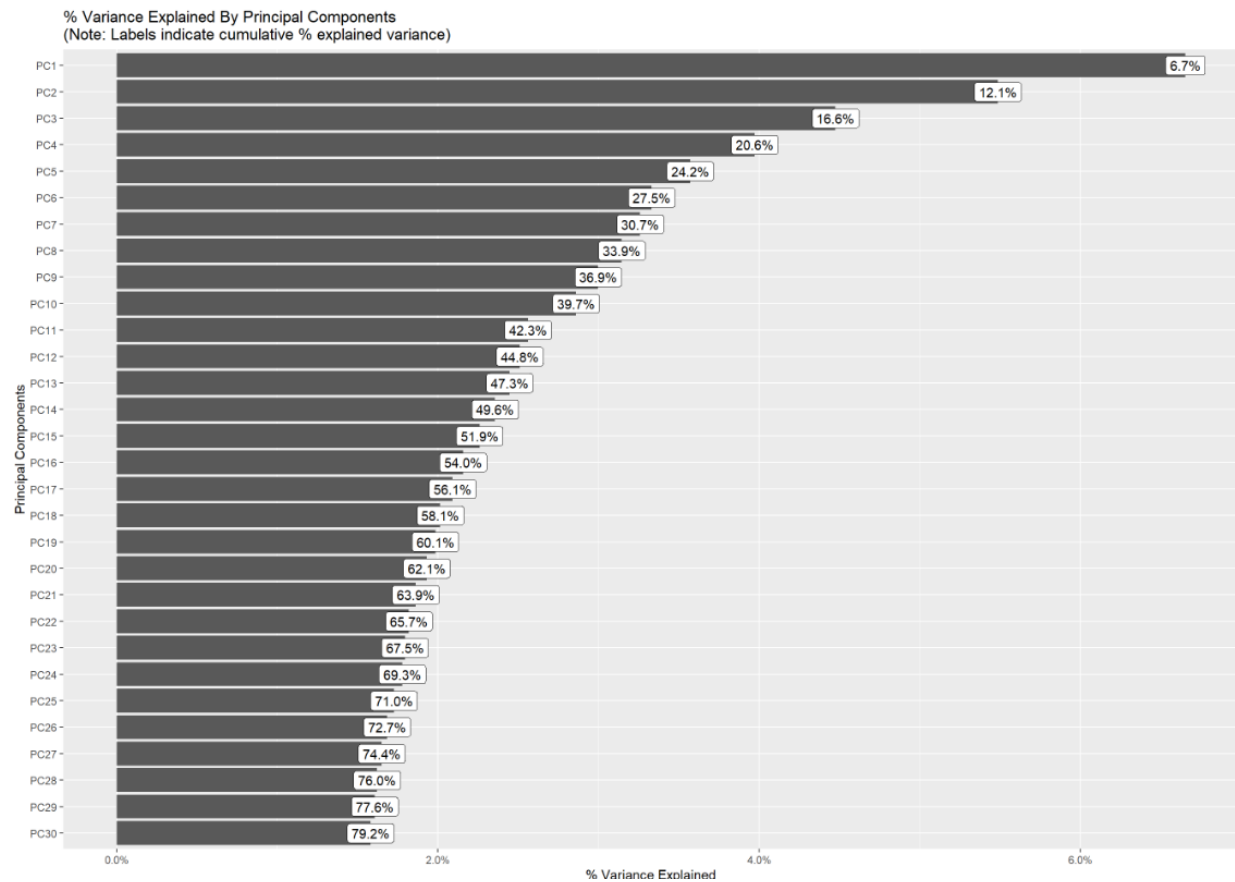
Także i **macierz korelacji** tym bardziej zbliżyła nas do pełnego pojęcia pełnego *german credit data* i wiele wartości pokryło się z naszą intuicją.

Correlation Analysis



Na koniec przyjrzelismy się jeszcze wykresowi analizy głównych składowych.

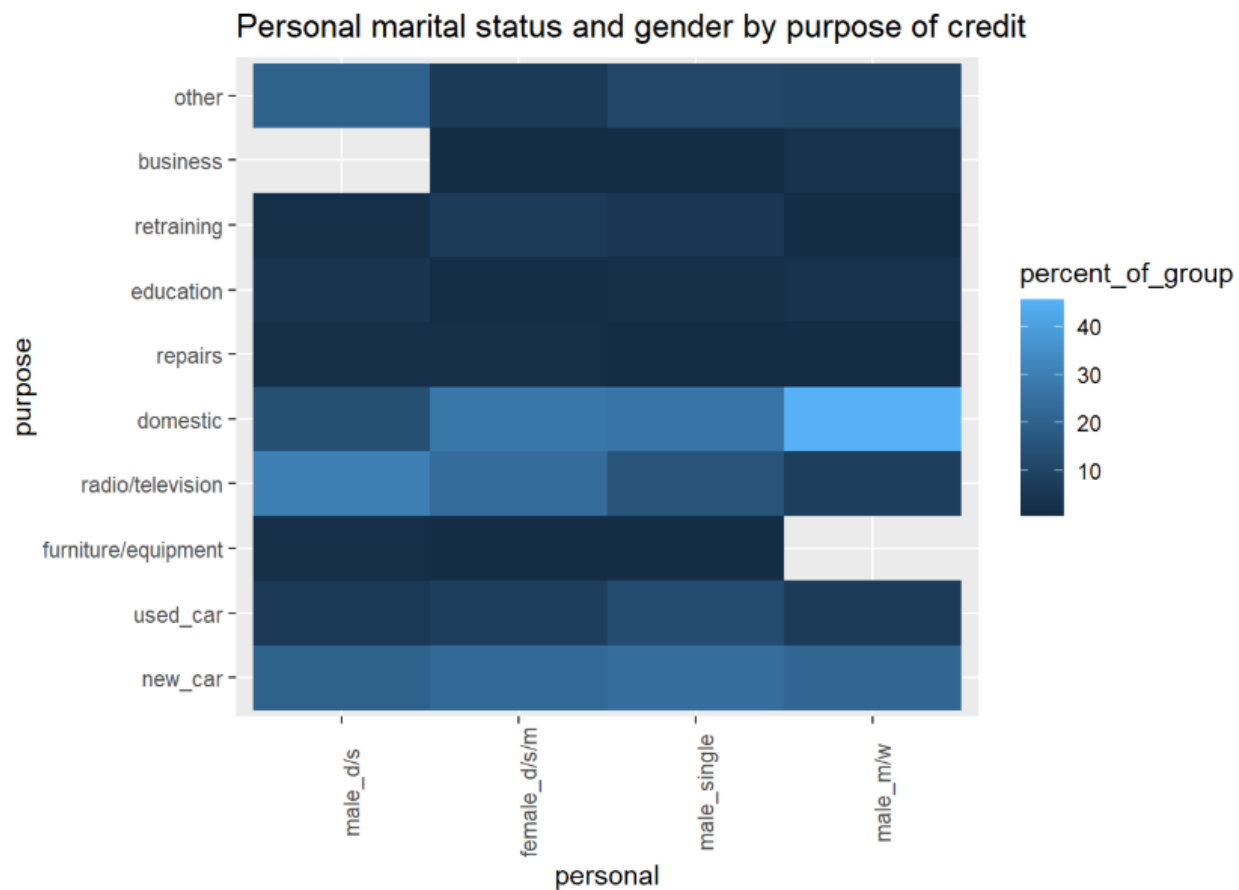
Principal Component Analysis



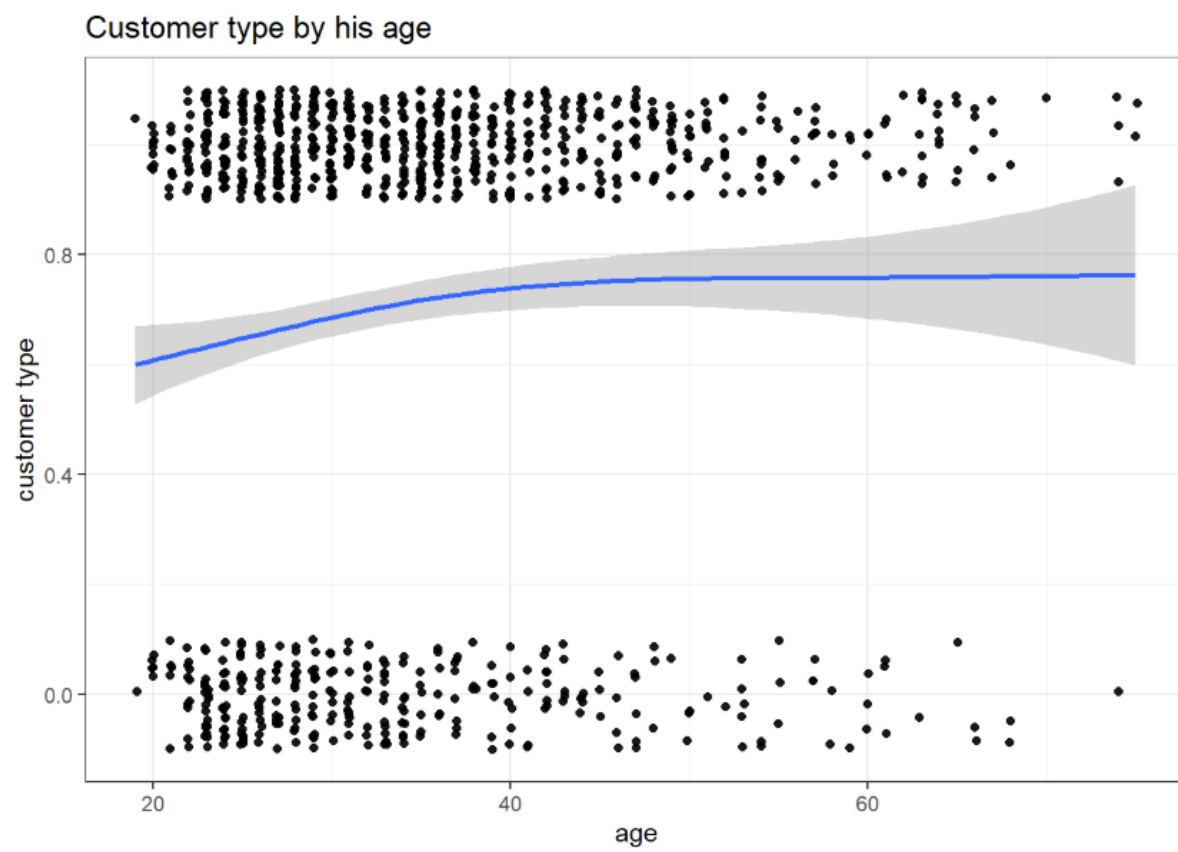
Oprócz tego postanowiliśmy sami przyrzeć się wybranym cechom.

Okazało się między innymi, że stereotypy można wyrzucić do kosza - mężczyźni o wiele częściej biorą kredyt ze względu na potrzebę funduszy na gospodarstwo domowe i nie widać znaczącej przewagi w stosunku do kobiet jeśli idzie o chęć postawienia pieniędzy na auto.

Bez zaskoczeń o wiele częściej na dom stawiają mężczyźni po ślubie niż ci samotni czy rozwodnicy. Co ciekawe ci sami ani razu nie wzięli pożyczki na wyposażenie / meble [przynajmniej na te 1000 osobników], zaś rozwodnicy i separatyści... przeciwnie do pozostałych grup nie myślą tu wręcz wcale o dodatkowej mamonie na biznes.

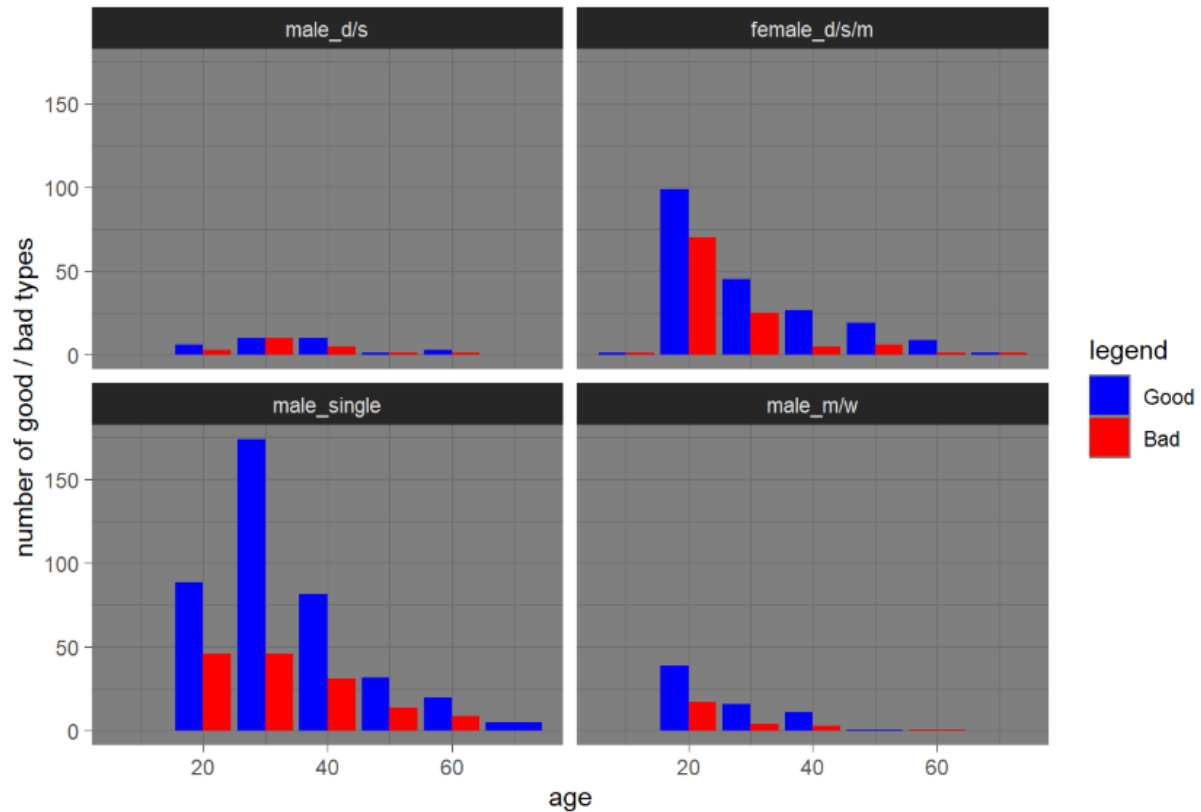


Wychodzi również na to, że generalnie większym zaufaniem firma daży osoby starsze:



Wyciągnęliśmy także wnioski na podstawie płci, wieku i stanu cywilnego.

Customer type by sex, marital status and age

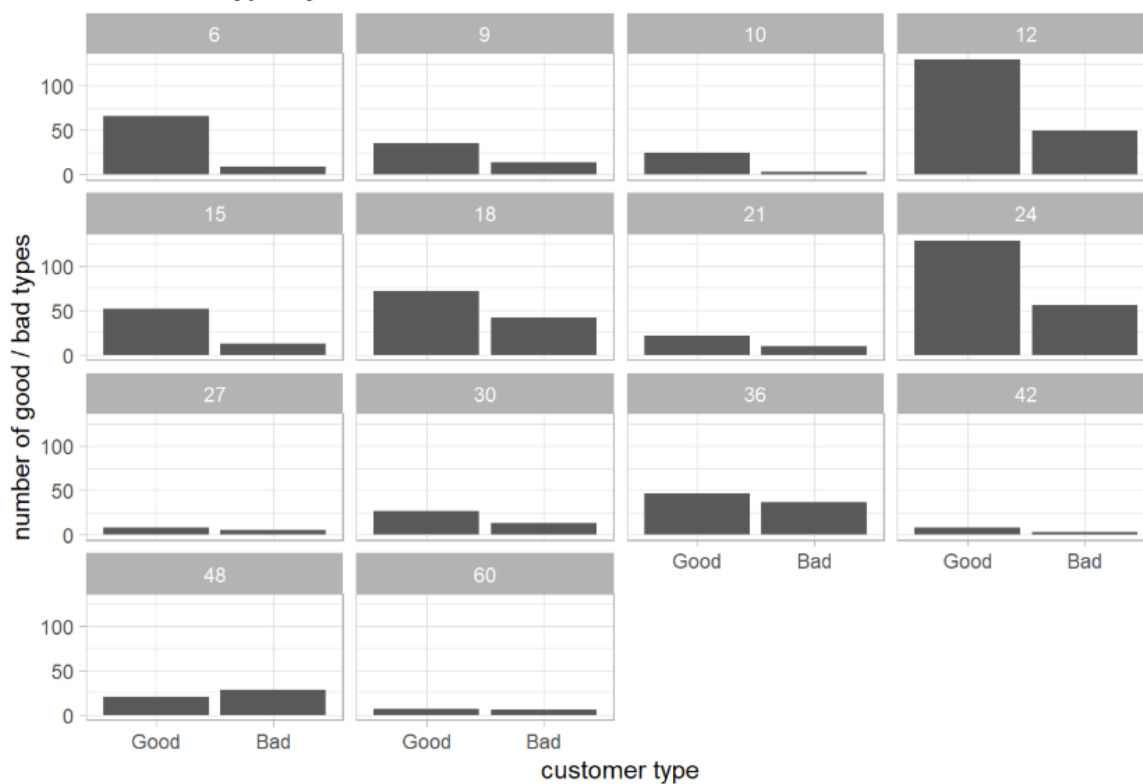


Dane mówią, że:

1. Najmniej ufamy rozwiedzionym facetom - zwłaszcza tym po 30, im zwykle nie dajemy.
2. Najbezpieczniejsi za to są też faceci po 30... ale single.
3. Żonaci to też dobre ziomki.
4. Kobiety są gorsze od mężczyzn, ale tylko przed 40. Potem raczej spokój, za wyjątkiem 70-tki psującej obraz.

Jak można się było spodziewać, pożyczka chętniej jest także udzielana na krótszy okres czasu.

Customer type by duration



4. Kodowanie

W ramach drugiego kamienia milowego dokonaliśmy szczegółowej analizy pod względem sensownego encodingu i **każdej kolumnie przyjrzelśmy się pod lupą**.

Spośród 21 kolumn, aż 14 okazało się być tekstowymi.

Do czynienia mieliśmy z problemami:

1. Prostych zmiennych binarnych
2. Kolumn nominalnych
3. Cech uporządkowanych

3.3.1. Zarobki w skali 0-4

... oczywiście jednoznacznie można uporządkować.

Do jednej kupy został wsadzony brak zarobków i brak informacji na ich temat - rozsądnie będzie obu przydzielić 0, gdyż można się spodziewać, że tak istotna informacja raczej nie byłoby zatajana przez osobę ubiegającą się o pożyczkę i realistycznie jest ona raczej w grupie *low* (przynajmniej według mnie - przypowiedź Kuba).

Wobec tego kolejne numery będą miały takie dopasowanie:

- not_have/unknown → 0
- low → 1
- normal → 2
- high → 3
- very_high → 4

```
num_data$savings <- as.numeric(data$savings)
num_data$savings[num_data$savings == 5] <- 0 # not_have/unknown
```

3.3.2. Staż pracy w przybliżonych liczbach

Kolumna *present_employment* daje nam ładne pogrupowanie długości pracy w formie grup *unemployed*, *less_than_year*, *1-3_years*, *4-6_years* i *more*. Zamienimy dane katagoryczne na liczbę oznaczającą oczekiwany staż pracy jak niżej:

- unemployed → 0
- less_than_year → 1
- 1-3_years → 2
- 4-6_yeras → 5
- more → 7

```
num_data$present_employment <- if_else(data$present_employment %in% c("unemployed"), 0, if_else(data$present_employment %in% c("less_than_year"), 1, if_else(data$present_employment %in% c("1-3_years"), 2, if_else(data$present_employment %in% c("4-6_yeras"), 5, 7))))
```

4. Zmiennych mieszanych - zawierających w sobie po parę ciekawych informacji

3.4. Zmienne “mieszane”

3.4.1. *personal* czyli płeć i stan cywilny na raz

Tu zrobimy dwie kolumny numeryczne - pierwszą binarną *is_woman* naturalnie odpowiadającą za płeć, dodatkową za pytanie o bycie singlem. W tym wypadku tracimy małą informację w stosunku do oryginalnego zbioru danych o odróżnieniu singli i rozwodników, ale jest to bardzo mała grupa, a podział na płeć powinien przynieść bardziej porządkany efekt.

```
num_data$is_woman <- if_else(data$personal == "female_d/s/m", 1, 0)
num_data$is_single <- if_else(data$personal == "male_single", 1, 0) # nie ma kobiet singli
```

Końcowy efekt wyszedł encodingu wyszedł następujący:

duration	credit_amount	installment_rate	present_residence	age	existing_credits	dependents	has_telephone	is_foreign_worker	is_good_customer_type	I
48	3914	4	2	38	1	1	0	1	0	
12	1922	4	2	37	1	1	0	1	0	
18	1345	4	3	26	1	1	0	1	0	
12	983	1	4	19	1	1	0	1	1	
21	3652	2	3	27	2	1	0	1	1	
18	1453	3	1	26	1	1	0	1	1	
24	3069	4	4	30	1	1	0	1	1	
8	1164	3	4	51	2	2	1	1	1	
18	2462	2	2	22	1	1	0	1	0	
36	7980	4	4	27	2	1	1	1	0	
18	2169	4	2	28	1	1	1	1	0	
48	7629	4	2	46	2	2	0	1	1	
48	6416	4	3	59	1	1	0	1	0	

Tutaj kolory odpowiadające miarom:

5. Poszukiwanie najlepszego modelu - dyskusja

W celu wybrania tego jednego właściwego modelu, wpierw postanowiliśmy podzielić się popularnymi znanymi już przez nas metodami i indywidualne zajęcie się nimi.

Rozpatrzyliśmy i przetestowaliśmy dla różnych hiperparametrów cztery algorytmy uczenia maszynowego:

1. Drzewo klasyfikacyjne i lasy losowe
2. Regresja liniowa
3. K-najbliższych sąsiadów
4. Naiwny klasyfikator bayesowski

... przy czym dla pierwszych trzech przygotowaliśmy dłuższe skrypty i raporty w formacie .Rmd - można je znaleźć pod [kolejno] *KM3Drzewa.Rmd*, *KM3Regresja.Rmd* i *KM3knn.Rmd*.

Efekty testowania mniej godne zraportowania znajdują się także w pojedynczych skryptach: *lm.r*, *nb.r* i *knn.r*.

Ku porównaniu efektów modeli postanowiliśmy porównywać cztery chyba najbardziej podstawowe w tej kwestii, ale i przy tym dające ogrom informacji miary: **accuracy**, **precision**, **recall** i **f1**.

2. Miary oceny klasyfikatora

W ocenie kolejnych modeli posłużę się czterema najbardziej klasycznymi miarami:

- $accuracy = \frac{TP+TN}{TP+FP+FN+TN}$
- $precision = \frac{TP}{TP+FP}$
- $recall = \frac{TP}{TP+FN}$
- $f1 = 2 * \frac{Recall * Precision}{Recall + Precision}$

```
confusion_matrix_values <- function(confusion_matrix){
  TP <- confusion_matrix[2,2]
  TN <- confusion_matrix[1,1]
  FP <- confusion_matrix[1,2]
  FN <- confusion_matrix[2,1]
  return (c(TP, TN, FP, FN))
}

accuracy <- function(confusion_matrix){
  conf_matrix <- confusion_matrix_values(confusion_matrix)
  return((conf_matrix[1] + conf_matrix[2]) / (conf_matrix[1] + conf_matrix[2] + conf_matrix[3] + conf_matrix[4]))
}

precision <- function(confusion_matrix){
  conf_matrix <- confusion_matrix_values(confusion_matrix)
  return(conf_matrix[1]/ (conf_matrix[1] + conf_matrix[3]))
}

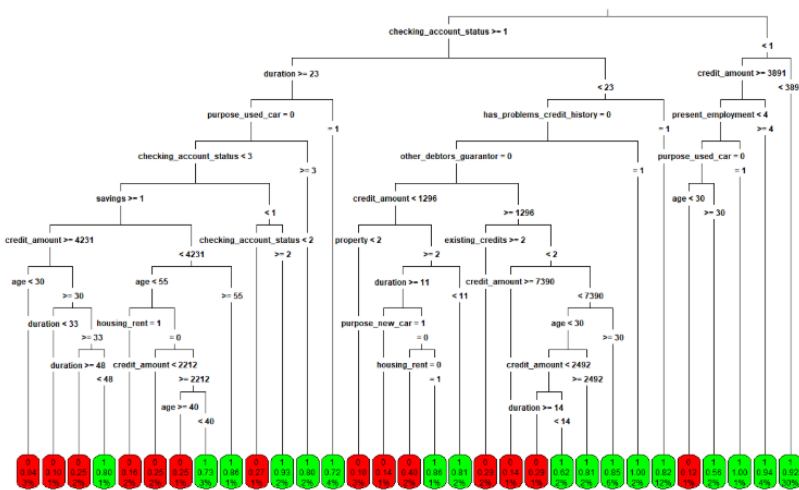
recall <- function(confusion_matrix){
  conf_matrix <- confusion_matrix_values(confusion_matrix)
  return(conf_matrix[1] / (conf_matrix[1] + conf_matrix[4]))
}

f1 <- function(confusion_matrix){
  conf_matrix <- confusion_matrix_values(confusion_matrix)
  rec <- recall(confusion_matrix)
  prec <- precision(confusion_matrix)
  return(2 * (rec * prec) / (rec + prec))
}
```

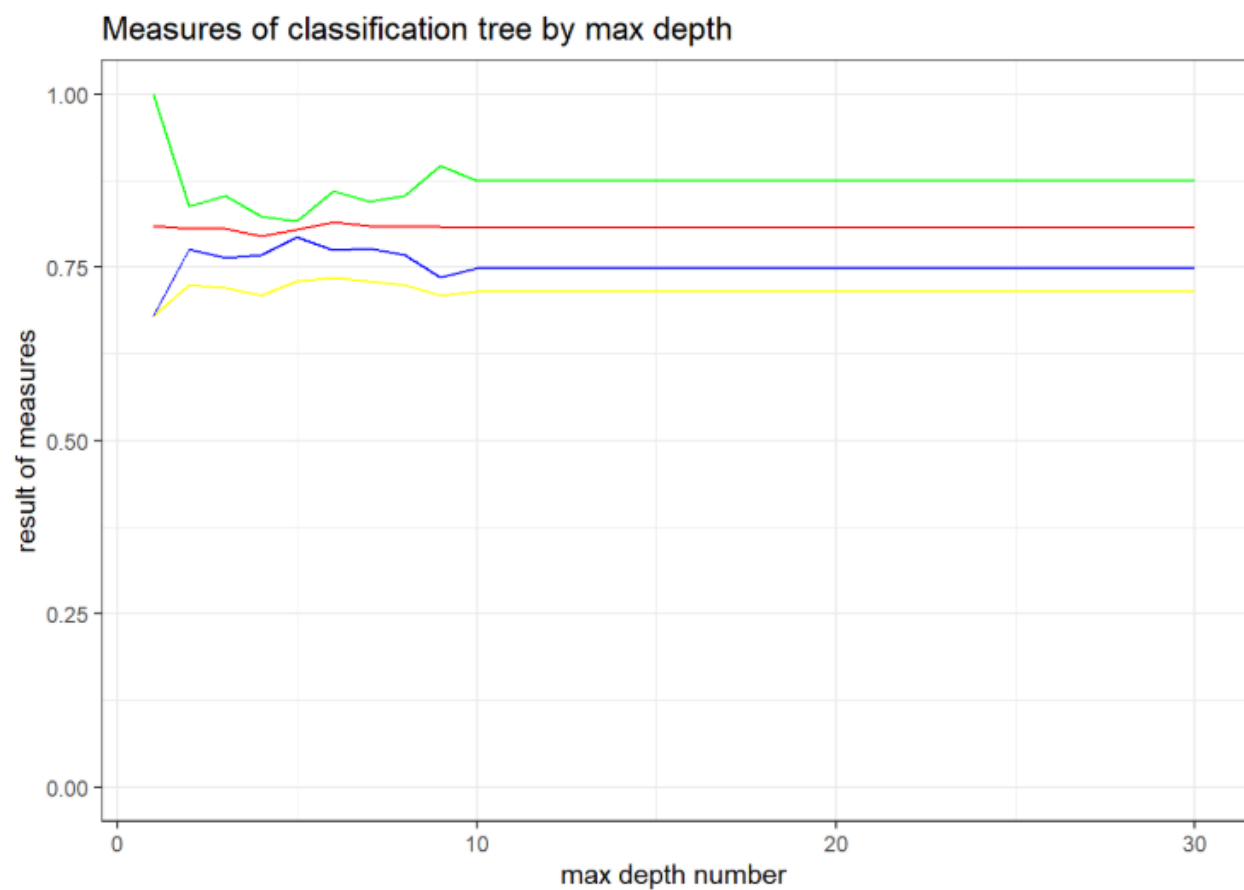
5.1. Drzewo klasyfikacyjny i las losowy

Uruchomienie algorytmu z pakietu *rpart* dało mało satysfakcjonujący wynik w myśli o logice biznesowej.


```
rpart.plot(primitive_model, type = 3, box.palette = c("red", "green"), fallen.leaves = TRUE)
```



W celu znalezienia najlepszych hiperparametrów, porównywaliśmy między innymi miary dla kolejnych maksymalnie narzuconych głębokości drzewa.



Tutaj:

- żółty - *accuracy*
- niebieski - *precision*
- zielony - *recall*
- czerwony - *f1*

Biorąc pod uwagę ideę naszego problemu, zdecydowaliśmy się na to z głębokością 6.

Prezentuje się ono tak:



Zajęliśmy się także **lasami losowymi** i koniec końców porównaliśmy otrzymane miary.

W efekcie otrzymaliśmy taką oto tabelkę:

##	model	accuracy	precision	recall	f1
## 1	primitive	0.715	0.7484277	0.8750000	0.8067797
## 2	max depth: 6	0.735	0.7748344	0.8602941	0.8153310
## 3	random forest	0.720	0.7325581	0.9264706	0.8181818

Ze względu na koncepcję naszego zadania, stwierdziliśmy, że **sumarycznie najlepiej wypada drzewo losowe o głębokości 6**.

5.2. k najbliższych sąsiadów

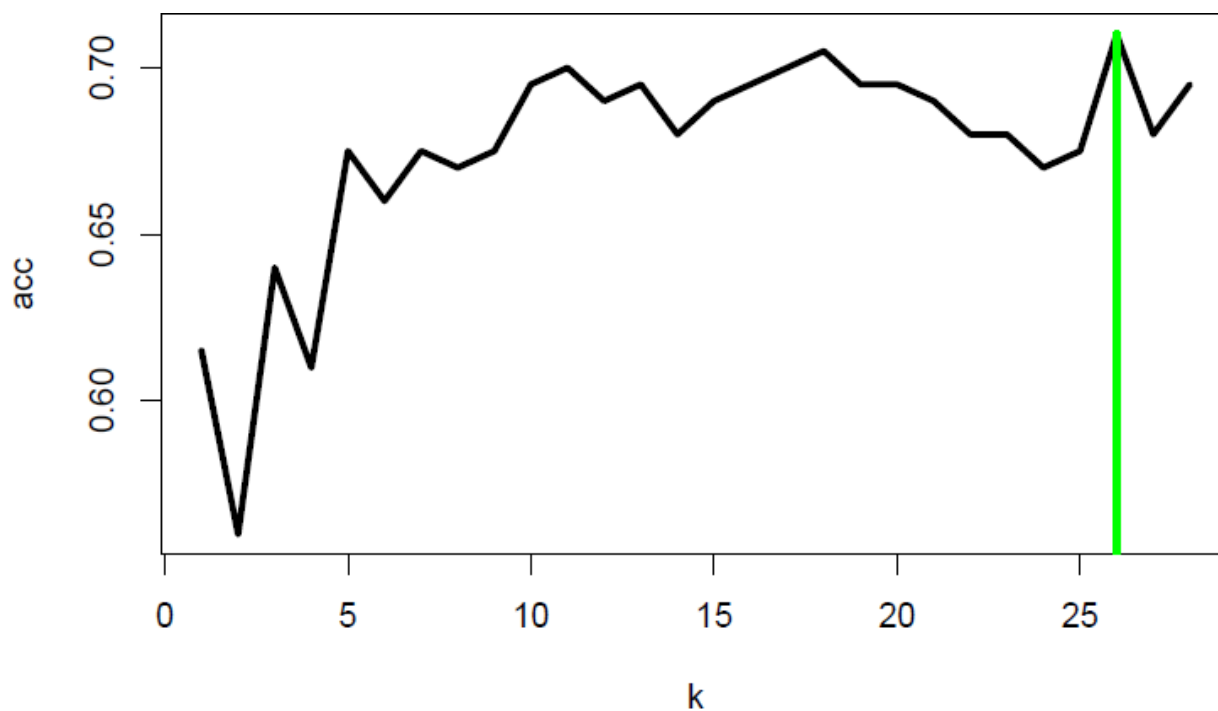
... napisaliśmy oczywiście ze wskazaną wcześniejszą **normalizacją**.

```
#normaliacion to ~(0,1)
for( col1 in 1:dim(x_train)[2]){
  #x and y have o be noemalized in the same way
  c_min <- min( c(x_train[,col1], x_test[,col1]))
  c_max <- max( c(x_train[,col1], x_test[,col1]))

  x_train[,col1] <- (x_train[,col1] - c_min) / (c_max - c_min)
  x_test[,col1] <- (x_test[,col1] - c_min) / (c_max - c_min)
}
```

Również i tutaj przeszukaliśmy różne hyperparameters w celu znalezienia tego najlepszego.

Najbardziej satysfakcjonujący wydał się efekt dla $k = 26$.



Finalnie otrzymaliśmy *accuracy* na poziomie 70%, a *confusion matrix* zaprezentował się tak:

	0	1
0	11	53
1	7	129

5.3. Regresja liniowa

Tutaj z pomocą przyszła nam funkcja *glm()*. Po dopasowaniu dobry obraz dała nam także metoda *summary*, ładnie podsumowująca co automatyczne narzędzie utworzyło.

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.4722	-0.8899	0.4654	0.7781	2.3977

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	3.384690	1.241297	2.727	0.00640	**
duration	-0.042163	0.007819	-5.392	6.97e-08	***
age	0.012578	0.009399	1.338	0.18084	
existing_credits	-0.394848	0.188578	-2.094	0.03628	*
dependents	-0.486423	0.265525	-1.832	0.06696	.
has_telephone	0.456194	0.207464	2.199	0.02788	*
is_foreign_worker	-1.482214	0.768042	-1.930	0.05362	.
has_problems_credit_history	0.782436	0.226453	3.455	0.00055	***
purpose_domestic	0.302664	0.334697	0.904	0.36584	
purpose_retraining	-0.772425	0.448215	-1.723	0.08483	.
purpose_radio_television	-0.042468	0.352021	-0.121	0.90398	
purpose_new_car	-0.671123	0.335964	-1.998	0.04576	*
purpose_used_car	1.109216	0.451582	2.456	0.01404	*
purpose_business	1.009181	1.142224	0.884	0.37695	
purpose_repairs	-0.334217	0.843957	-0.396	0.69210	
purpose_education	-0.543683	0.601829	-0.903	0.36632	
purpose_furniture_equipment	0.439982	0.828232	0.531	0.59526	
other_debtors_guarantor	0.634619	0.451470	1.406	0.15982	
other_debtors_co_applicant	-0.811476	0.444269	-1.827	0.06777	.
other_installment_plans_bank	-0.367892	0.248429	-1.481	0.13864	
other_installment_plans_stores	-0.702364	0.387586	-1.812	0.06996	.
housing_rent	-0.143903	0.389959	-0.369	0.71211	
housing_own	0.270075	0.336478	0.803	0.42218	

Tak zaimplementowana regresja liniowa nie daje binarnego dopasowania tak / nie, lecz prawdopodobieństwa na owe dwa stany. Postanowiliśmy najprościej - i chyba na tym poziomie najlepiej - przyjąć, że **jeśli p-stwo na "tak" >50%** - klient został uznany za zdolnego finansowo. Jeśli nie - przeciwnie.

```
glm.probs <- ifelse(predict(glm.fit,newdata = Reg_test,type = "response") > 0.5,1,0)
mean(glm.probs == select(Reg_test,is_good_customer_type))
```

```
## [1] 0.7
```

Jak widzimy model średnio dobrze przewiduje 70 % odpowiedzi.

Macierz pomyłek i wcześniej zaplanowane badane miary zaprezentowały się następująco:

	0	1
0	16	45
1	15	124

accuracy	precision	recall	f1
0.7	0.7337278	0.8920863	0.8051948

W celu uzyskania potencjalnie lepszych wyników postanowiliśmy także pozbyć się tych kolumn, które według *summary()* najmniej wpływają na model.

accuracy	precision	recall	f1
0.665	0.7045455	0.8920863	0.7873016

Miary nie okazały się być jednak lepsze.

Bardziej satysfakcjonujący wynik dało za to usunięcie zmiennych:

- *age*
- *is_foreign_worker*
- *present_employment*
- *property*
- *checking_account_status*
- *is_single*

Tutaj już radość została osiągnięta.

accuracy	precision	recall	f1
0.69	0.7015707	0.9640288	0.8121212

Ostatecznie postanowiliśmy jeszcze wziąć tylko pięć kolumn o największym znaczeniu według *summary()*.

- *purpose_radio_television* : **0.90**
- *housing_rent* : **0.71**
- *purpose_repairs* : **0.69**
- *purpose_furniture_equipment* : **0.59**
- *housing_own* : **0.42**

... taki model okazał się jednak bardzo zły, gdyż wszystkie obserwacje zostały zaklasyfikowane jako prawdziwe.

Podsumowując najlepszy okazał się model pierwszy - chyba, że najważniejsze znaczenie miałyby mieć *recall* - wtedy przy nieco gorszym *precision* i *accuracy* wygrywa model nieuwzględniający wspomnianych wcześniej sześciu parametrów.

6. Wybór najlepszego algorytmu uczenia maszynowego i implementacja

BAYES??

TODO

7. Zakończenie

To by było na tyle.

Mamy nadzieję, że się podobało; ja myślę, że fajna robota (dopowiedź: Kuba).

```
sessionInfo()
```

```

## R version 3.6.1 (2019-07-05)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18362)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Polish_Poland.1250 LC_CTYPE=Polish_Poland.1250
## [3] LC_MONETARY=Polish_Poland.1250 LC_NUMERIC=C
## [5] LC_TIME=Polish_Poland.1250
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods    base
##
## loaded via a namespace (and not attached):
## [1] compiler_3.6.1  magrittr_1.5    tools_3.6.1    htmltools_0.3.6
## [5] yaml_2.2.0      Rcpp_1.0.1      stringi_1.4.3  rmarkdown_2.1
## [9] knitr_1.22      stringr_1.4.0   xfun_0.6       digest_0.6.18
## [13] evaluate_0.14

```