Report for finance factoring prediction

Reinhard Palaver

13 4 2020

Introduction

In following report we are talking about predictions for finance factoring based on a dataset from Kaggle.

The dataset we are using is a set of invoice data, with additional information about the customers and the situation for each single invoice, like how long it lasts to settle the invoice and whether the invoice was disputed or not, etc.

This dataset was choosen, because it is quite good related to my working experience as a CFO and furthermore consultant in the area of accounting and controlling.

The main goals in this project are two.

First we try to predict, whether the invoices will be settled late or not. Let us say the payment or settling of the open invoices will be later than due date. With such information an accouting office is able to clarify and check beforehand the open and maybe critical invoices, just to avoid late settlements.

And on the other hand we try to figure out the expected payments for the open invoices on a timeline. With such information we are able to check our cash flow situation for every single day in the future.

Here we are documenting the analysis and presents the findings, with supporting statistics and figures.

Dataset

The original dataset we are using is called "Factoring data from IBM" and are downloaded from Kaggle and 1:1 stored at my GitHub-Repository:

https://raw.githubusercontent.com/rpalaver/FinanceFactoring/master/WA_Fn-UseC_-Accounts-Receivable.csv

In our dataset we have 2.466 invoices from the year 2012 and 2013 and the whole number of customers are 100.

After loading the data we have to convert the data into different formats for better reading and working on it.

```r
# convert invoice date from character to format date
df$PaperlessDate <- as.Date(df$PaperlessDate, "%m/%d/%Y")
df$InvoiceDate <- as.Date(df$InvoiceDate, "%m/%d/%Y")
df$DueDate <- as.Date(df$DueDate, "%m/%d/%Y")
df$SettledDate <- as.Date(df$SettledDate, "%m/%d/%Y")

# convert some variables to factors
df$countryCode <- as.factor(df$countryCode)
df$customerID <- as.factor(df$customerID)
```

```r
df$Disputed <- as.factor(df$Disputed)
df$PaperlessBill <- as.factor(df$PaperlessBill)
```

For better reading we split each row into two parts.The first part of data looks like this:

invoiceNumber	customerID	countryCode	InvoiceAmount	InvoiceDate	DueDate	PaperlessBill
611365	0379-NEVHP	391	55.94	2013-01-02	2013-02-01	Paper
7900770	8976-AMJEO	406	61.74	2013-01-26	2013-02-25	Electronic
9231909	2820-XGXSB	391	65.88	2013-07-03	2013-08-02	Electronic
9888306	9322-YCTQO	406	105.92	2013-02-10	2013-03-12	Electronic
15752855	6627-ELFBK	818	72.27	2012-10-25	2012-11-24	Paper
18104516	5148-SYKLB	818	94.00	2012-01-27	2012-02-26	Paper
23864272	8690-EEBEO	897	74.69	2013-08-13	2013-09-12	Electronic
27545037	4460-ZXNDN	770	75.06	2012-12-16	2013-01-15	Paper
28049695	3831-FXWYK	770	80.07	2012-05-14	2012-06-13	Paper
32277701	7654-DOLHO	897	48.33	2013-07-01	2013-07-31	Electronic

And the rest of informations for the same invoices looks like this:

invoiceNumber	Disputed	SettledDate	DaysToSettle	DaysLate	PaperlessDate
611365	No	2013-01-15	13	0	2013-04-06
7900770	Yes	2013-03-03	36	6	2012-03-03
9231909	No	2013-07-08	5	0	2012-01-26
9888306	No	2013-03-17	35	5	2012-04-06
15752855	Yes	2012-11-28	34	4	2012-11-26
18104516	Yes	2012-02-22	26	0	2013-08-28
23864272	No	2013-09-09	27	0	2012-12-05
27545037	No	2013-01-12	27	0	2013-06-27
28049695	Yes	2012-07-01	48	18	2013-03-08
32277701	No	2013-07-26	25	0	2012-04-04

Next step will be to split the whole dataset into a training and a validation dataset.

```r
# Validation set will be 20% of df
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = df$DaysLate, times = 1, p = 0.2, list = FALSE)
train_set <- df[-test_index,]
temp <- df[test_index,]

# Make sure customerID in validation set are also in  set
test_set <- temp %>%
  semi_join(train_set, by = "customerID")

# Add rows removed from validation set back into edx set
removed <- anti_join(temp, test_set)
train_set <- rbind(train_set, removed)

rm(df, test_index, temp, removed)
```

Analysis

Before starting the analysis and a better understanding of all further figures we add a simple variable named **Late**. This variable determines, whether the settlement of invoice was too late or not.

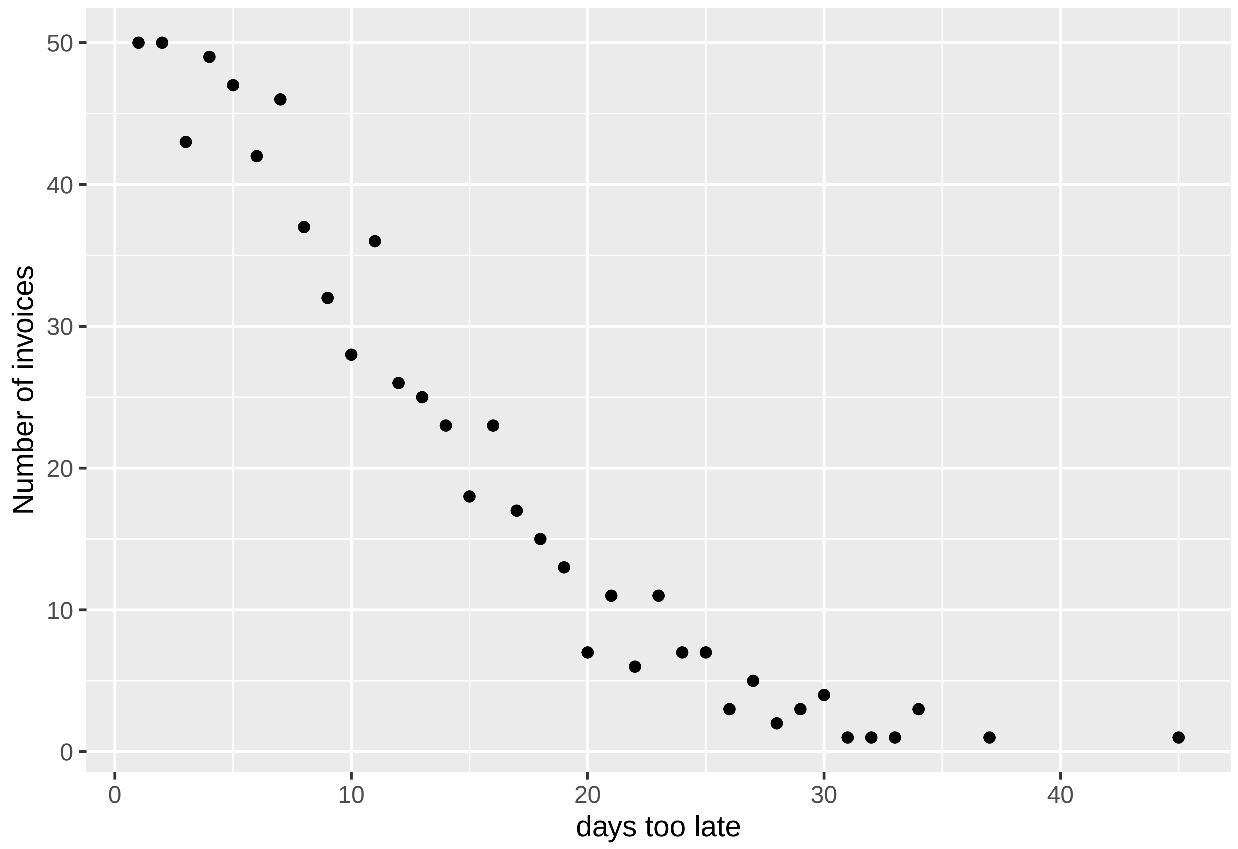
```
train_set <- train_set %>% mutate(Late = ifelse(DaysLate > 0, 1,0))
test_set  <- test_set  %>% mutate(Late = ifelse(DaysLate > 0, 1,0))
```

The first thing we compute is the proportion of late settlements:

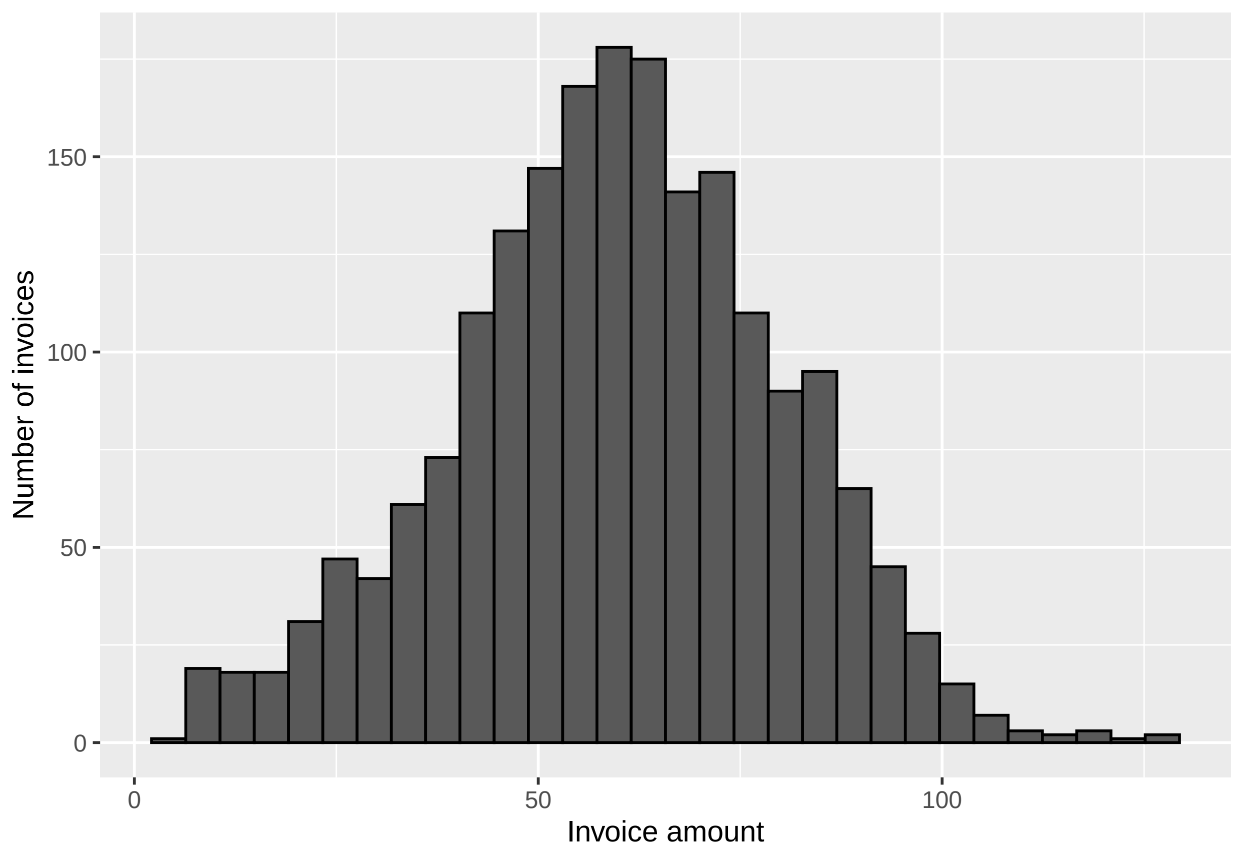
```
## 0.351927
```

So to say about one-third of all inovices are late.

And for all other invoices which are late, we are having decreasing number of invoices all the more **DaysLate** is increasing.

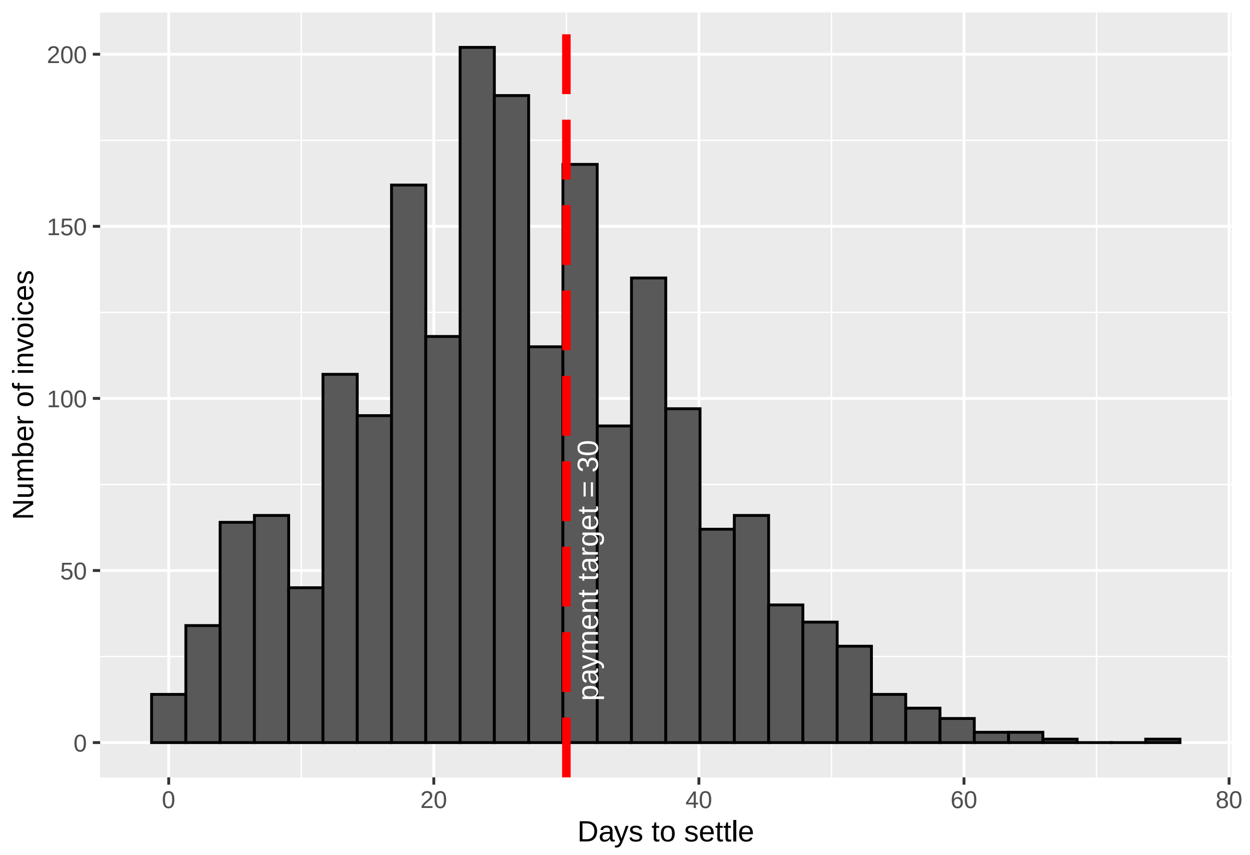


Next we have a look for the distribution of the variable **InvoiceAmount**.



The variable **InvoiceAmpount** shows us a quite perfect normally distribution.

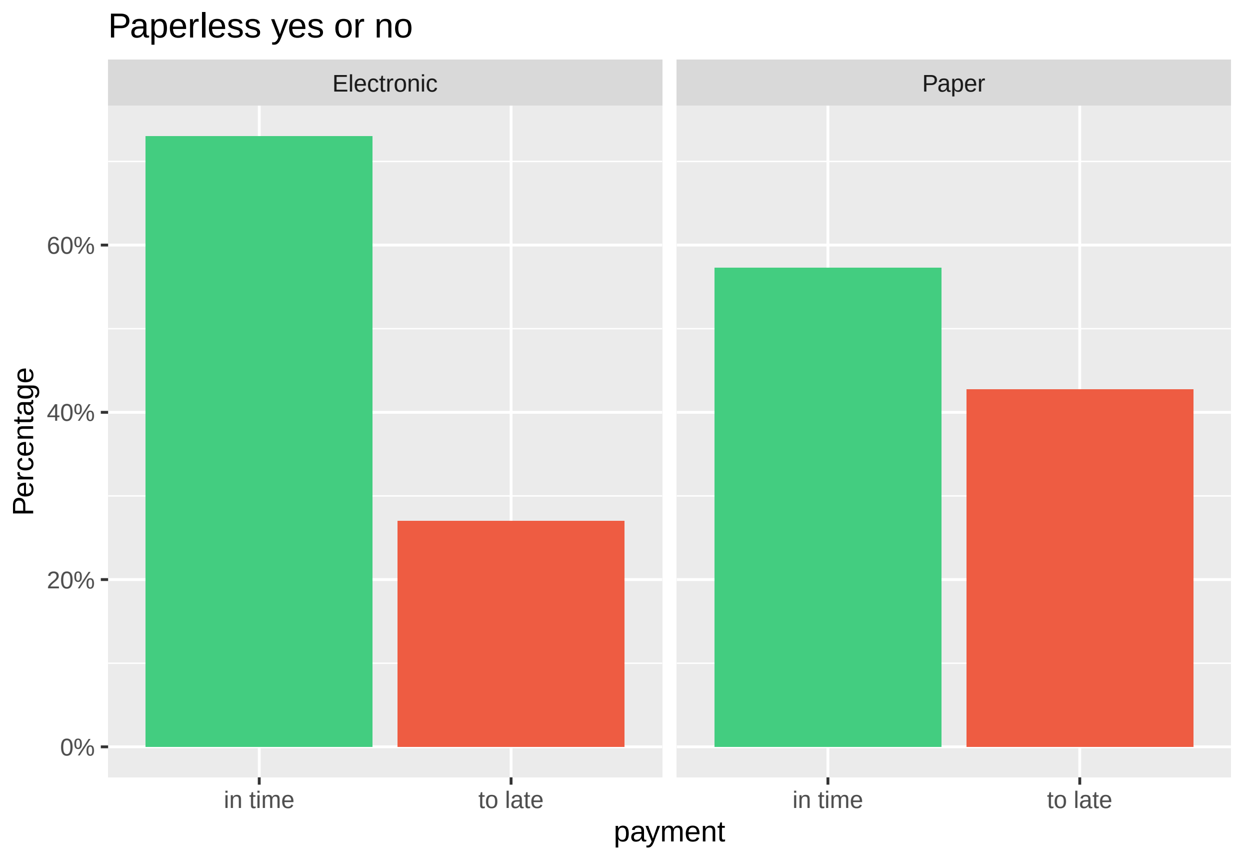
Now we have a look for the variable **DaysToSettle**.



We see a left squewed distribution lefthand from the general payment target of 30 days.

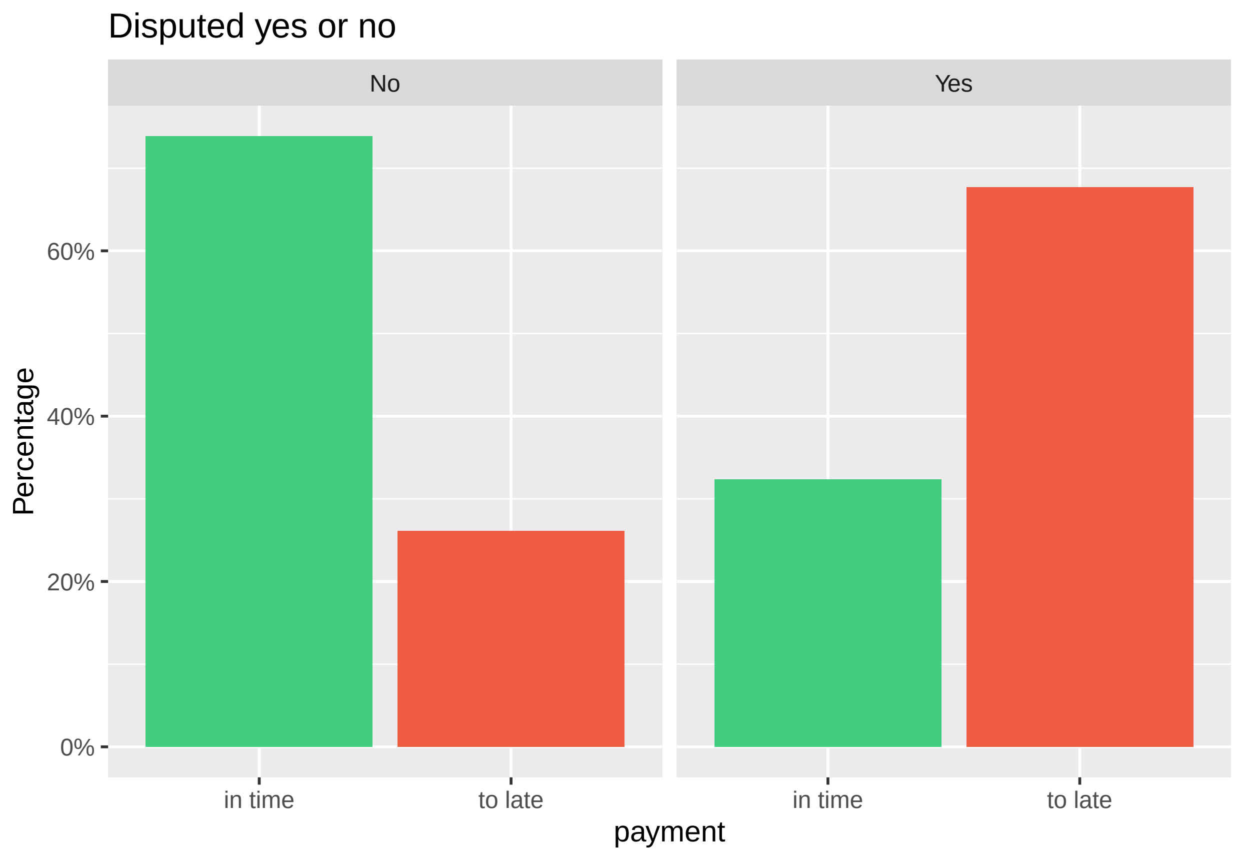
As a very interesting variable we are analysing **PaperlessBill**. This information tells us, whether this invoice was sent by paper or on electronic way (e.g. eMail, EDI, . . .).

In all following figures we work with a percentage of *"in time"* and *"to late"* and distinguish between different categories, like here *"Electronic"* and *"Paper"*.



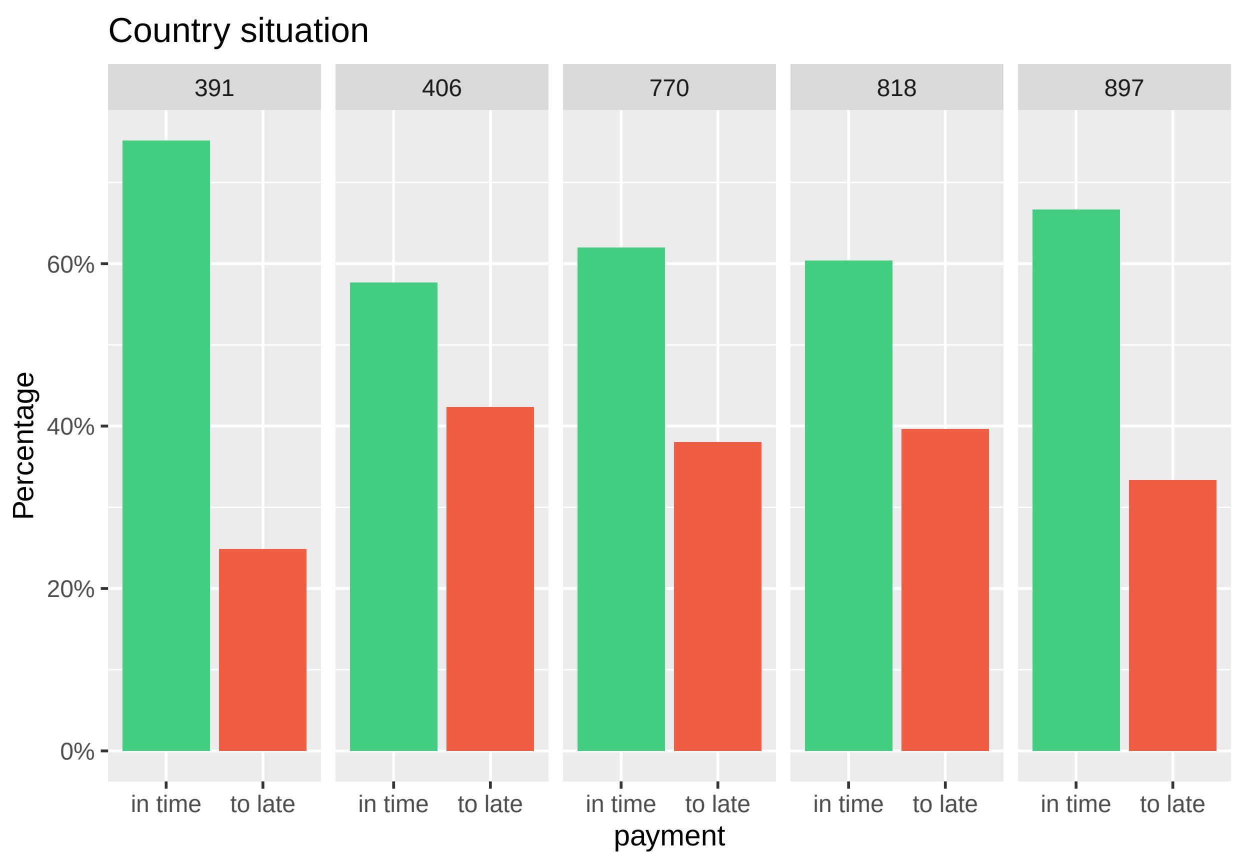
We see that **PaperlessBill** is significant regarding late payments!

Next we'll have a look for **Disputed** yes or no.



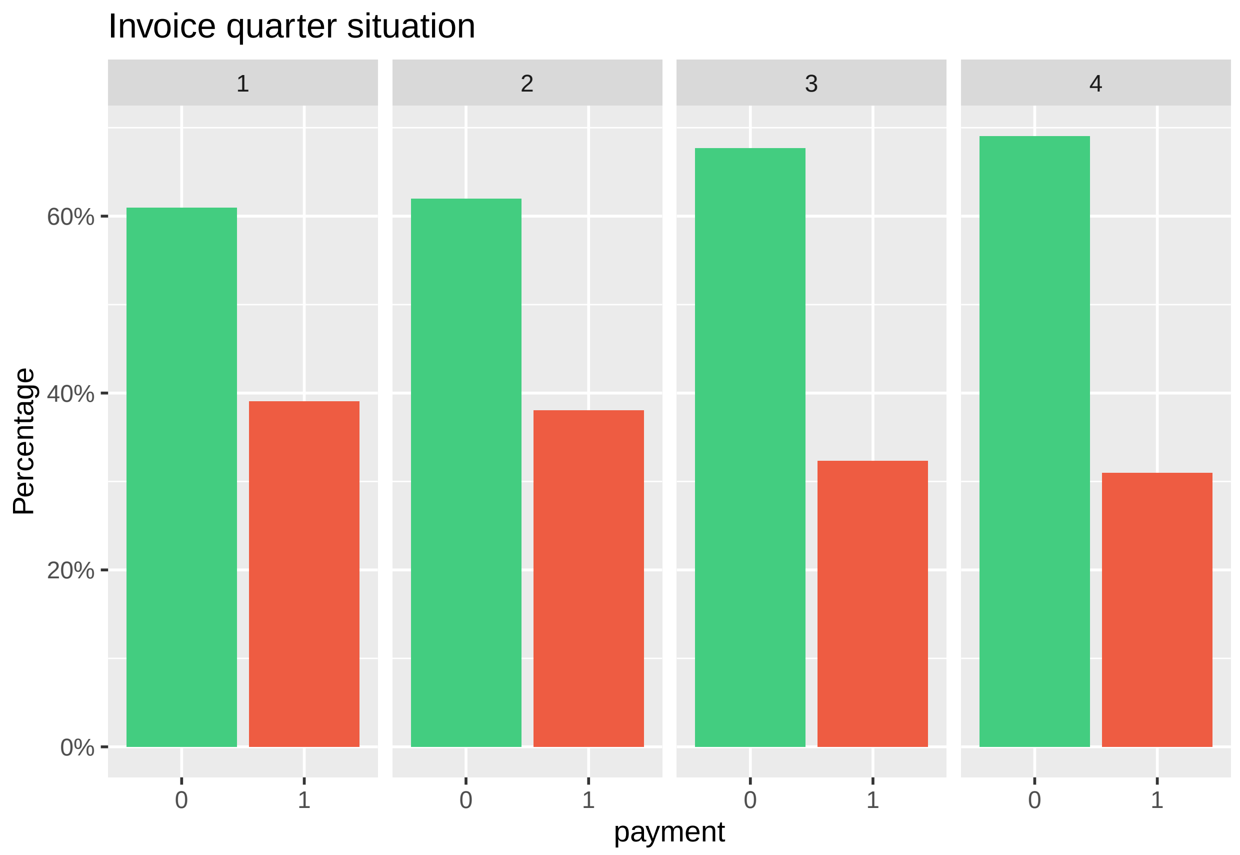
Here we do have a higher signifaction and this variable could be important to our following estimations.

Let us have a look at the impact of the different **countryCode**.



The main impact is coming from the **countryCode** 391. All others are more or less similiar.

After all we take the variable **InvoiceDate** and group these quarterly.



If we have a look at this figure we could assume that the customers are paying less invoices at the end of the year. During the year the willingness to pay is decreasing.

Summary

These variables are important for our estimation.

countryCode

InvoiceAmount

PaperlessBill

Disputed

DaysToSettle

DaysLate

quarter

Data wrangling

As a additional predictor we take the training set and group the invoices by **customerID** and compute the percentage of **Late**.

Because we have a lot of different values at **InvoiceAmount** and **DaysLate** we stratify the amount into *small*, *medium* and *high* and **DaysToSettle** by 5.

Now we add all these information to the training and testing set by customerID with the function **“left_join”**.

```
# add variable for quarter
train_set <- train_set %>% mutate(quarter = quarter(InvoiceDate, with_year = FALSE))
test_set <- test_set %>% mutate(quarter = quarter(InvoiceDate, with_year = FALSE))

# create a table grouped by customer to compute variables for mean of DaysToSettle, Late und DaysLate
custTable <- train_set %>% group_by(customerID) %>% summarize(mu_DaysToSettle = mean(DaysToSettle),
                                                              mu_Late = mean(Late),
                                                              mu_DaysLate = mean(DaysLate))

# stratify DaysToSettle
custTable <- custTable %>% mutate(strat_DaysToSettle = round(mu_DaysToSettle/5)*5)

# add values to train and test set
train_set <- left_join(train_set, custTable, by = "customerID")
test_set <- left_join(test_set, custTable, by = "customerID")
rm(custTable)

# stratify InvoiceAmount
train_set <- train_set %>% mutate(Amount_bin = ifelse(InvoiceAmount < 20,"small", ifelse(InvoiceAmount 
test_set <- test_set %>% mutate(Amount_bin = ifelse(InvoiceAmount < 20,"small", ifelse(InvoiceAmount > 
```

Correlation matrix

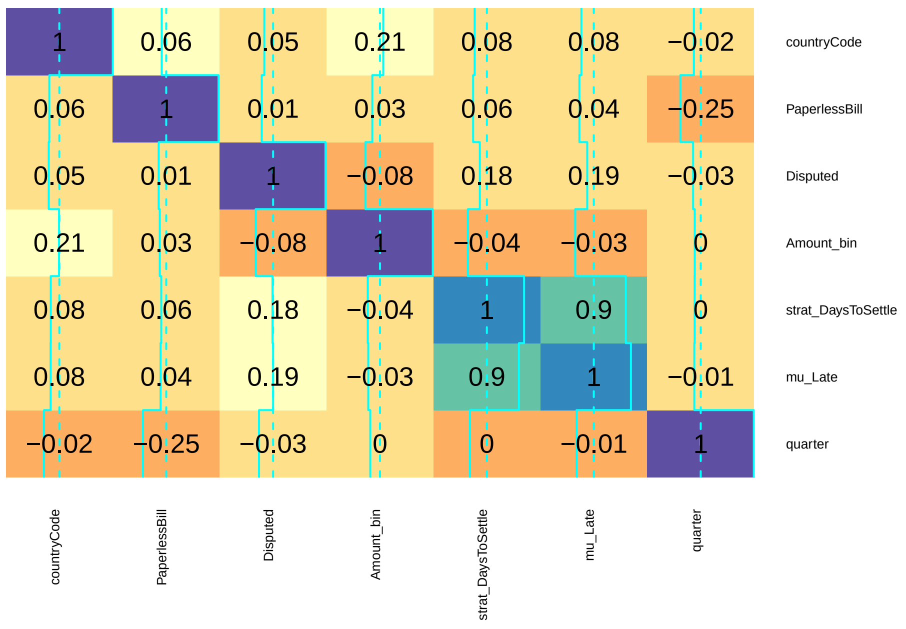
Before we start with estimation we will have a look at the correlation of all variables:

```
# create dataset from train set to plot correlation matrix
corr_set <- train_set %>%
  mutate(countryCode = as.numeric(countryCode),
  PaperlessBill = as.numeric(PaperlessBill),
  Disputed = as.numeric(Disputed),
  Amount_bin = as.numeric(factor(Amount_bin))) %>%
  select(countryCode, PaperlessBill, Disputed, Amount_bin, strat_DaysToSettle, mu_Late, quarter)

# compute correlations
res <- cor(corr_set)

# show correlation matrix
heatmap.2(res
          ,cellnote=round(res,2)
          ,notecol = "black"
          ,cexRow = 0.5
          ,cexCol = 0.5
          ,scale = "column"
          ,col = RColorBrewer::brewer.pal(11, "Spectral")
          ,main = "Correlation matrix"
          ,Colv = NA
          ,Rowv = NA
          ,key = FALSE)
```

Correlation matrix



The correlations are not very significant except DaysToSettle and DaysLate, because DaysLate is a part of DaysToSettle. Therefore it will be a challenge getting high accuracy.

Train models to predict if a payment will be late

First of all we extract the relevant variables from both sets and copy it to two new sets called **train** and **test**.

```
# prepare datasets for modeling and select predictors
test <- test_set %>%  select(countryCode, PaperlessBill, Disputed, Amount_bin, strat_DaysToSettle, mu_La
train <- train_set %>%  select(Late, countryCode, PaperlessBill, Disputed, Amount_bin, strat_DaysToSettl
                              quarter)
train <- train %>% mutate(Late = as.factor(Late))
```

Modeling

We are using six different models for classification of **Late**:

- linear discriminant analysis (lda)
- generalized linear model (glm)
- recursive partitioning and regression rrees (rpart)
- support vector machines with linear kernel (svmLinear)
- k-nearest neighbors (knn)
- generalized additive model using LOESS (gamLoess)

We are not using the model *"random forest"* in this project because of runtime behavior. Outside this project I figured out that this model shows more or less the same result as the other models.

```
# train different models
models <- c("lda", "glm", "rpart","svmLinear", "knn", "gamLoess")
set.seed(1, sample.kind = "Rounding")
fits <- lapply(models, function(model){
  print(model)
  train(Late ~ ., method = model, data = train)
})
```

```
## [1] "lda"
## [1] "glm"
## [1] "rpart"
## [1] "svmLinear"
## [1] "knn"
## [1] "gamLoess"
```

```
# predictions of different models
pred <- sapply(fits, function(object)
  predict(object, newdata = test))
colnames(pred) <- models
```

After running the different models we are creating an ensemble and add this estimate to our table of results.

```
# create ensemble
ensemble <- pred == 1
ensemble_preds <- ifelse(rowMeans(ensemble) > 0.5, 1, 0)
pred_new <- cbind.data.frame(pred, ensemble_preds)

# compute accuracy of different models
all_models <- c(models, "ensemble")
n <- seq(1:length(all_models))

# compute accuracy, sensitivity, specificity and prevalence
accuracy <- sapply(n, function(object)
  mean(pred_new[,object] == test_set$Late))

sensitivity <- sapply(n, function(object)
  confusionMatrix(factor(pred_new[,object]), reference = factor(test_set$Late))$byClass["Sensitivity"])

specificity <- sapply(n,function(object)
  confusionMatrix(factor(pred_new[,object]), reference = factor(test_set$Late))$byClass["Specificity"])

data.frame(Model = all_models, Accuracy = accuracy, Sensitivity = sensitivity,
           Specificity = specificity) %>% knitr::kable()
```

Model	Accuracy	Sensitivity	Specificity
lda	0.8502024	0.8810289	0.7978142
glm	0.8522267	0.8842444	0.7978142
rpart	0.8481781	0.8360129	0.8688525
svmLinear	0.8663968	0.8810289	0.8415301

Model	Accuracy	Sensitivity	Specificity
knn	0.8198381	0.8842444	0.7103825
gamLoess	0.8603239	0.8874598	0.8142077
ensemble	0.8502024	0.8745981	0.8087432

```
## Prevalence
##  0.6295547
```

The best result we receive is coming from **svmLinear**, but they are all very close. The key figures **Sensitivity** and **Specificity** are pretty good and the **Prevalence** seems to be good.

For further information we may have a look at the importance of variables from some selected models:

```r
# extract important variables from different models
glm <- varImp(fits[[2]])$importance
glm <- rownames_to_column(glm)
rpart <- varImp(fits[[3]])$importance
rpart <- rownames_to_column(rpart)
gam <- varImp(fits[[6]])$importance
gam <- rownames_to_column(gam)

# create table to compare different importance
variables <- data.frame(rowname = glm[,1])
variables <- left_join(variables, glm, by = "rowname")
variables <- left_join(variables, rpart, by = "rowname")
variables <- left_join(variables, gam, by = "rowname")
colnames(variables) <- c("variables","glm", "rpart", "gamLoess")
options(digits = 2)
variables %>% knitr::kable()
```

variables	glm	rpart	gamLoess
countryCode406	20.1	1.29	1.03
countryCode770	0.0	0.00	0.60
countryCode818	24.5	0.00	0.00
countryCode897	6.0	0.44	1.12
PaperlessBillPaper	58.9	8.78	1.95
DisputedYes	100.0	38.78	10.98
Amount__binmedium	1.1	0.11	NA
Amount__binsmall	8.4	0.00	NA
strat__DaysToSettle	27.6	93.97	100.00
mu__Late	49.3	100.00	7.16
quarter	16.9	0.28	0.32

Now we know, that the variables **Disputed**, **mu__Late** and **stratified days to settle** depending on the model do have the most impact on our estimations.

Conclusion

Therefore all models are very close and our origin dataset is quite "small", we do not achieve a higher accuracy. A second reason for that is, that we don't have a high number of invoices per customer. At the end I suggest to work with the ensemble and check time by time the behavior of the different models for ajdusting variables and models.

Train models to predict expected payments

In this chapter we try to figure out the expected payments for the open invoices at a timeline. The prediction is based on the variable **DaysToSettle**.

With such information we are able to check our cash flow situation for every single day in the future.

But here we will do a regression instead of a classification at the previous chapter.

Once more we copy the variables from both sets and copy it to two new sets called **train** and **test**. But in this case we are taking **all** variables.

```
test <- test_set %>%  select(countryCode, customerID, InvoiceAmount, PaperlessBill, Disputed, Amount_bi
                             mu_DaysToSettle, mu_DaysLate, strat_DaysToSettle, mu_Late, quarter)
train <- train_set %>%  select(DaysToSettle, countryCode, customerID, InvoiceAmount, PaperlessBill, Dis
                               Amount_bin, mu_DaysToSettle, mu_DaysLate, strat_DaysToSettle, mu_Late, q
```

Modeling

We are using five different models for regression of **DayToSettle**:

- linear models (lm)
- recursive partitioning and regression rrees (rpart)
- support vector machines with linear kernel (svmLinear)
- k-nearest neighbors (knn)
- generalized additive model using LOESS (gamLoess)

And once more we are not using the model *"random forest"* in this project because of runtime behavior. Outside this project I have done it and the result of this model has been more or less the same as the other models.

```
# train different models
models <- c("lm", "rpart","svmLinear", "knn", "gamLoess")
set.seed(1, sample.kind = "Rounding")
fits <- lapply(models, function(model){
  print(model)
  train(DaysToSettle ~ ., method = model, data = train)
})
```

```
## [1] "lm"
## [1] "rpart"
## [1] "svmLinear"
## [1] "knn"
## [1] "gamLoess"
```

```
# predictions of different models
pred <- sapply(fits, function(object)
  predict(object, newdata = test))
```

For comparison reason we add two dates computed by **InvoiceDate** plus *payment target* and **InvoiceDate** plus overall average of **DaysToSettle**.

```r
# add columns for mean of DaysToSettle and payment target for comparison reason
target <- rep(c(30), times = nrow(test_set))
mean_DaysToSettle <- rep(c(mean(train_set$DaysToSettle)), times = nrow(test_set))
comparisons <- c(models, "target", "mean")
pred_new <- cbind.data.frame(pred, target, mean_DaysToSettle)
colnames(pred_new) <- comparisons
```

To find out which model fits best, we are working with the keyfigure **RMSE** (root mean square deviation):

```r
# compute RMSE for all col's
n <- seq(1:length(comparisons))
rmse <- sapply(n, function(num){
  RMSE(test_set$DaysToSettle, pred_new[,num])
})

data.frame(comparisons, rmse)[order(rmse),] %>% knitr::kable()
```

	comparisons	rmse
1	lm	5.2
3	svmLinear	5.3
5	gamLoess	5.8
4	knn	7.9
2	rpart	8.8
7	mean	12.1
6	target	12.6

The **RMSE** from model **"lm"** is the best comparing to all others, specially if we compare it to the payment target or the average of days for settlement of an invoice.

To visualize the differences we are building a table based on the payments coming from the different models.

```r
# create table with payment dates for different estimates comparing to true payment day
# round predicted days
pred_new <- round(pred_new)

# convert days to dates and add true settledDate
pred_date <- cbind.data.frame(true = as.numeric(test_set$SettledDate), as.numeric(test_set$InvoiceDate)

# convert dates to periods
n <- seq(1:ncol(pred_date))
periods <- sapply(n, function(num)
  format(as.Date(pred_date[,num], origin="1970-01-01"), "%Y-%m"))
colnames(periods) <- names(pred_date)

# add amount to table
periods <- cbind.data.frame(Amount = test_set$InvoiceAmount,periods)

# create timeline
timeline <- gather(periods, key = "Amount") %>% group_by(value) %>% summarize()
```

```r
colnames(timeline) <- "period"

# create datasets for 4 estimations
true_pay <- periods[,1:2] %>% group_by(true) %>% summarize(x = sum(Amount))
colnames(true_pay) <- c("period", "true_payment")
lm_pay <- periods[,c(1,3)] %>% group_by(lm) %>% summarize(x = sum(Amount))
colnames(lm_pay) <- c("period", "lm_payment")
target_pay <- periods[,c(1,8)] %>% group_by(target) %>% summarize(x = sum(Amount))
colnames(target_pay) <- c("period", "target_payment")
mean_pay <- periods[,c(1,9)] %>% group_by(mean) %>% summarize(x = sum(Amount))
colnames(mean_pay) <- c("period", "mean_payment")

# create table by left_join
cash_Table <- left_join(timeline, true_pay, by = "period")
cash_Table <- left_join(cash_Table, lm_pay, by = "period")
cash_Table <- left_join(cash_Table, target_pay, by = "period")
cash_Table <- left_join(cash_Table, mean_pay, by = "period")

# set 0 for na's
cash_Table[is.na(cash_Table)] <- 0

# compute differences
cash_Table <- cash_Table %>% mutate(diff_lm = (lm_payment - true_payment),
                                    diff_target = (target_payment - true_payment),
                                    diff_mean = (mean_payment - true_payment))
# show table
cash_Table %>% knitr::kable()
```

period	true_payment	lm_payment	target_payment	mean_payment	diff_lm	diff_target	diff_mean
2012-01	161	190	0.0	0	28.8	-161	-161
2012-02	835	806	813.5	895	-28.8	-21	60
2012-03	1065	986	1338.2	1431	-78.7	274	367
2012-04	1553	1632	1388.6	1285	78.7	-164	-268
2012-05	1174	1245	1137.1	1383	71.2	-37	209
2012-06	1351	1171	1286.8	1116	-180.4	-64	-235
2012-07	1138	1206	1225.3	1274	68.5	88	137
2012-08	1199	1075	1414.3	1374	-123.3	216	175
2012-09	1811	1788	1402.1	1604	-23.3	-409	-207
2012-10	1469	1606	1522.3	1490	136.9	54	21
2012-11	1061	1035	1114.9	873	-25.9	54	-188
2012-12	1031	1216	1366.2	1522	185.6	335	491
2013-01	1463	1422	969.2	1052	-41.7	-494	-411
2013-02	1294	1249	1437.0	1327	-44.9	143	33
2013-03	1244	1375	1490.0	1458	130.9	246	214
2013-04	1298	1065	825.7	917	-232.9	-472	-380
2013-05	1375	1553	1512.0	1392	178.1	137	17
2013-06	1597	1488	1983.5	1963	-109.7	386	366
2013-07	1839	1886	1523.5	1544	47.2	-315	-295
2013-08	1049	1052	1225.5	1303	2.6	176	254
2013-09	1211	1172	754.9	681	-39.0	-456	-530
2013-10	1047	1106	1399.6	1383	59.2	352	336
2013-11	1341	1303	1188.9	1169	-37.8	-152	-172
2013-12	856	834	1359.1	1251	-21.4	503	395

period	true_payment	lm_payment	target_payment	mean_payment	diff_lm	diff_target	diff_mean
2014-01	227	227	8.4	0	0.0	-218	-227

Compare overview:

```r
# compare min and max differences
max <- apply(X = cash_Table[,6:8], MARGIN = 2, FUN = max, na.rm = TRUE)
min <- apply(X = cash_Table[,6:8], MARGIN = 2, FUN = min, na.rm = TRUE)
overview <- rbind.data.frame(max, min)
colnames(overview) <- c("diff true to lm", "diff true to target", "diff true to mean")
rownames(overview) <- c("highest positive difference in one month", "highest negative difference in one

# show overview
overview %>% knitr::kable()
```

	diff true to lm	diff true to target	diff true to mean
highest positive difference in one month	186	503	491
highest negative difference in one month	-233	-494	-530

Ratio of differences:

```r
options(digits = 4)
deviation <- overview/mean(cash_Table$true_payment)*100
deviation %>% knitr::kable()
```

	diff true to lm	diff true to target	diff true to mean
highest positive difference in one month	15.63	42.40	41.39
highest negative difference in one month	-19.62	-41.61	-44.62

The ratio shows us, that our differences per month are more than the halve of the other estimations.

Conclusion

Our predictions are not so bad comparing to simple estimations like payment target. But for use in practive and better reliability we need more data and estimations.