exam

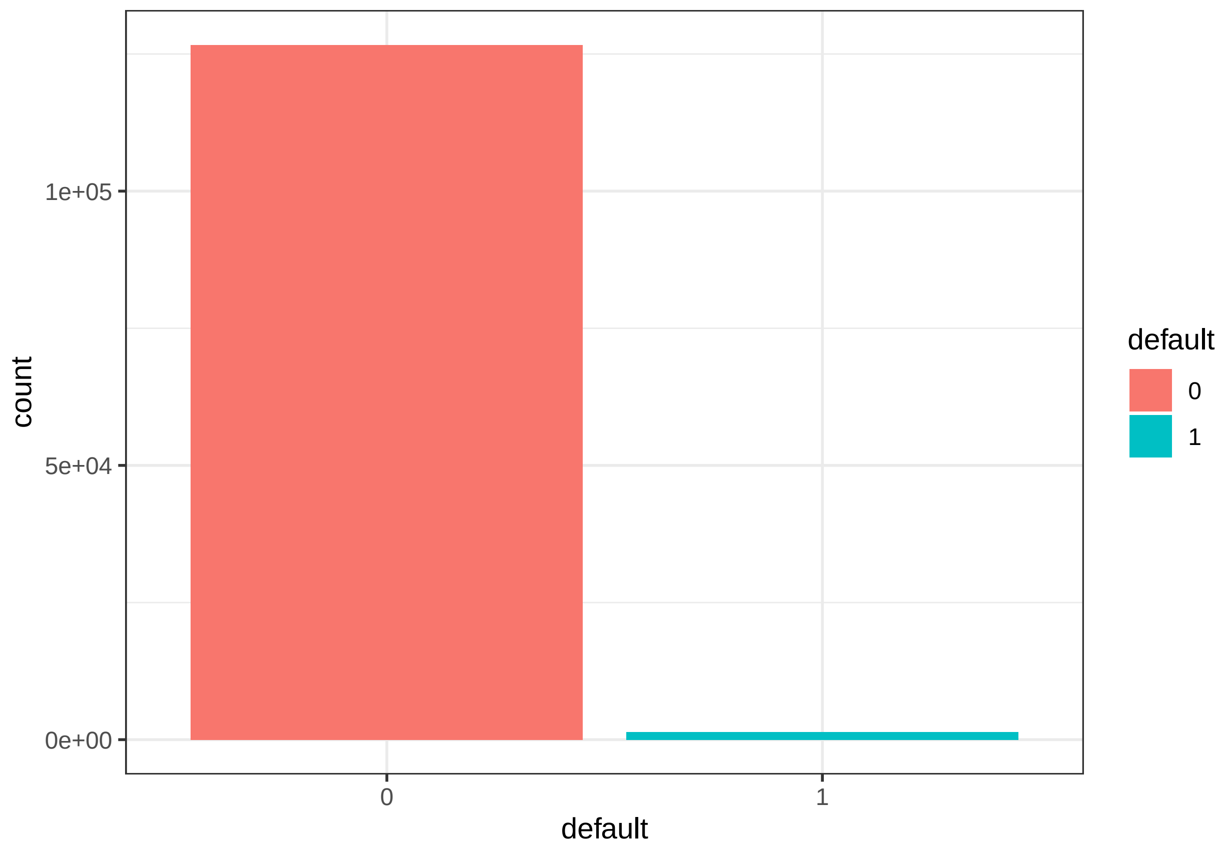
R Markdown

1. Setting up the data

An overview of the data shows us tha we have to change the structure of some of the variabels. Moreover, there are many extreme values in the data set and observations of defaulting companies are higly under-represented in the data set. We will deal with all these problems prior to implementing our prediction models.

Table 1:

Statistic	Min	Max
default	0	1
profit_margin	−10,000,000,000,000,000,000.000	25,553.330
gross_operating_inc_perc	0.000	1.000
operating_margin	−10,000,000,000,000,000,000.000	586.175
EBITDA_margin	−10,000,000,000,000,000,000.000	586.171
interest_coverage_ratio	−10,000,000,000,000,000,000.000	10,000,000,000,000,000,000.000
cost_of_debt	−8,131.722	165,944.600
interest_bearing_debt	−10,000,000,000,000,000,000.000	7,523.000
revenue_stability	−10,000,000,000,000,000,000.000	4,473.242
equity_ratio	−10,000,000,000,000,000,000.000	168.999
equity_ratio_stability	−10,000,000,000,000,000,000.000	168.000
liquidity_ratio_1	−2,473.828	10,000,000,000,000,000,000.000
liquidity_ratio_2	−2,473.790	10,000,000,000,000,000,000.000
liquidity_ratio_3	−2,473.809	10,000,000,000,000,000,000.000
equity	−771,200	182,466,000
total_assets	−9,685	544,267,000
revenue	−2,883,588	588,422,000
age_of_company	2	22
unpaid_debt_collection	−5.000	10,000,000,000,000,000,000.000
paid_debt_collection	−5.500	10,000,000,000,000,000,000.000
adverse_audit_opinion	0	6
industry	0	11
amount_unpaid_debt	−121,226	10,000,000,000,000,000,000
payment_reminders	0	3



Factor variables: Then we recategorize som of the factor variables. Adverse audit is coded as a dummy, where 1 indicates that there has been an adverse audit opinion, and 0 indicates no adverse audit.

The tables below show the distribution of the companies along the factor variables, depending on wether they have defaulted or not.

	0	1	2	3	4	5	6
0	86937	1029	87	12435	3305	22340	536
1	293	22	1	147	59	825	54

	0	1
0	86937	39732
1	293	1108

	0	1	2	3	4	5	7	8	9	10	11
0	50541	916	842	12627	13255	639	37287	4391	3286	2279	606
1	369	10	7	178	171	13	501	82	48	7	15

Let's have a look of the distribution of the variables in the data set. The figure below contains density plots for all the numeric variables. Due to the presence of outliers, these figures do not provide much information.

#Handling missing observations and outliers

We observe that the number xx appears throughout the data set, and assume that these are missing observations. In total, these extreme values account for xx percent of our observations.

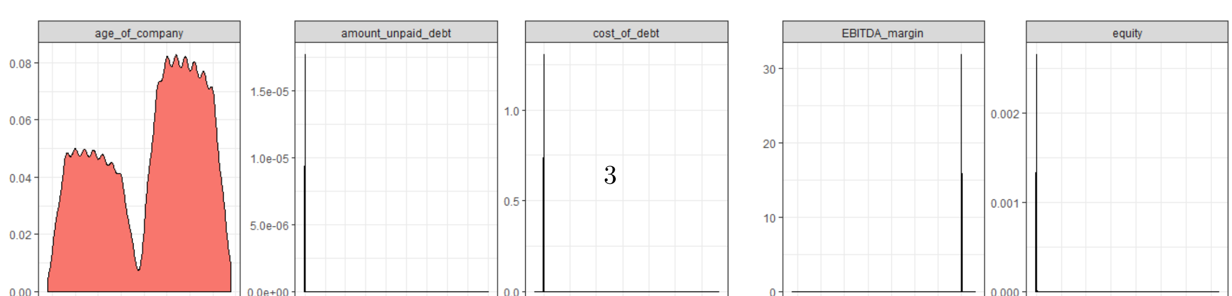
We choose to replace these values with NA to begin with.

	0	1	2	3
0	80182	26677	13775	6035
1	328	229	292	552

	key	value
1	profit__margin	0.10
2	gross__operating__inc__perc	0.00
3	operating__margin	0.10
4	EBITDA__margin	0.10
5	interest__coverage__ratio	0.17
6	cost__of__debt	0.00
7	interest__bearing__debt	0.00
8	revenue__stability	0.23
9	equity__ratio	0.01
10	equity__ratio__stability	0.14
11	liquidity__ratio__1	0.02
12	liquidity__ratio__2	0.02
13	liquidity__ratio__3	0.02
14	equity	0.00
15	total__assets	0.00
16	revenue	0.00
17	age__of__company	0.00
18	unpaid__debt__collection	0.01
19	paid__debt__collection	0.01
20	amount__unpaid__debt	0.01

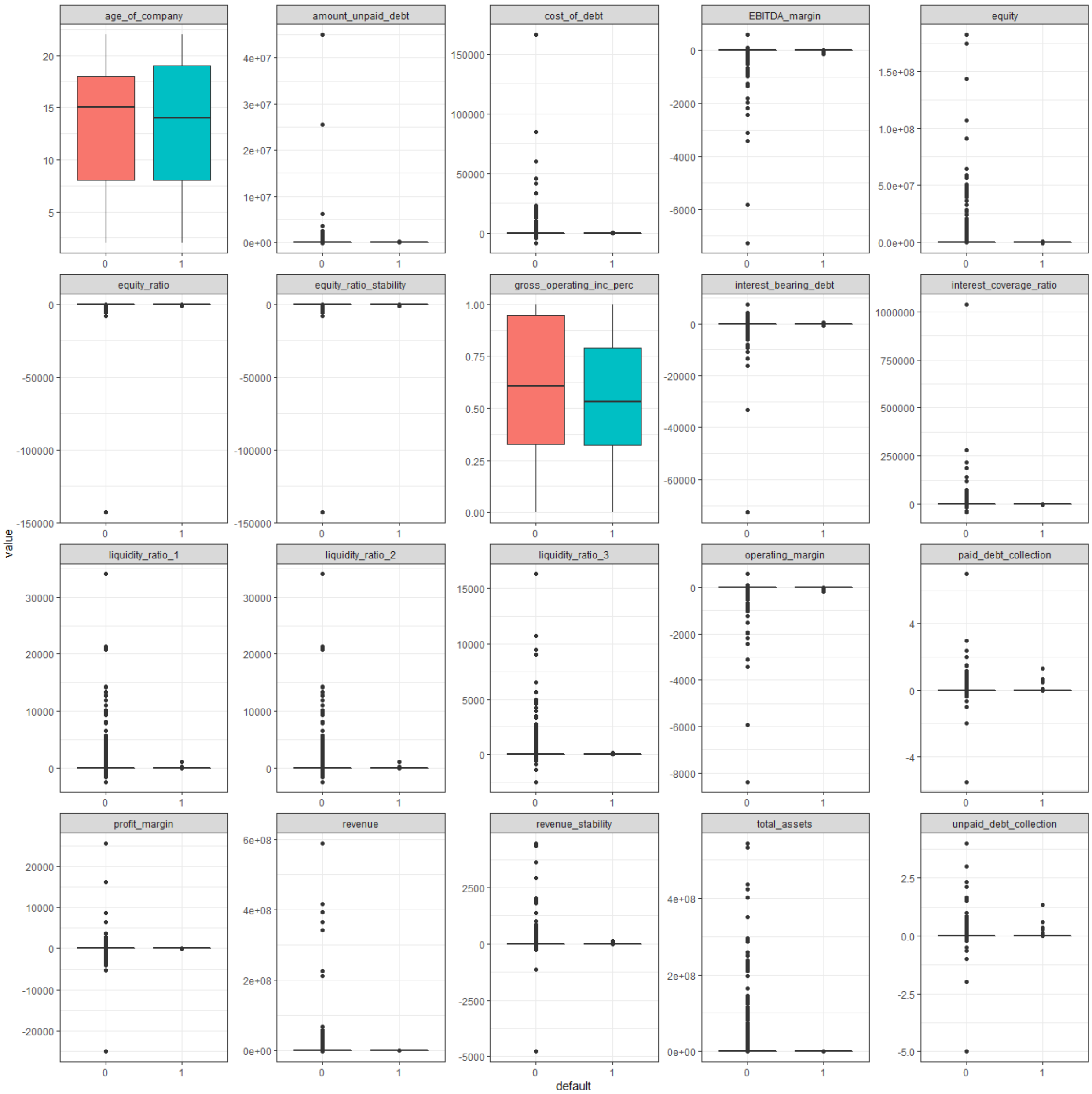
Table 2:

Statistic	Min	Max	Mean	St. Dev.
profit__margin	−25,001.000	25,553.330	0.382	128.950
gross__operating__inc__perc	0.000	1.000	0.588	0.339
operating__margin	−8,391.346	586.175	−0.641	37.493
EBITDA__margin	−7,259.521	586.171	−0.536	34.862
interest__coverage__ratio	−41,488.830	1,037,285.000	85.094	3,551.132
cost__of__debt	−8,131.722	165,944.600	13.098	616.227
interest__bearing__debt	−72,702.900	7,523.000	−1.827	247.353
revenue__stability	−4,775.143	4,473.242	0.682	35.810
equity__ratio	−143,015.000	168.999	−2.899	405.043
equity__ratio__stability	−143,016.000	168.000	−4.151	435.700
liquidity__ratio__1	−2,473.828	34,126.680	8.238	201.297
liquidity__ratio__2	−2,473.790	34,126.590	7.901	201.280
liquidity__ratio__3	−2,473.809	16,330.310	3.378	87.779
equity	−771,200	182,466,000	32,016.920	1,090,415.000
total__assets	−9,685	544,267,000	122,994.800	4,352,257.000
revenue	−2,883,588	588,422,000	60,941.350	2,890,611.000
age__of__company	2	22	13.167	5.455
unpaid__debt__collection	−5.000	4.000	0.001	0.030
paid__debt__collection	−5.500	7.000	0.001	0.035
amount__unpaid__debt	−121,225.600	45,000,000.000	957.351	147,363.800



The figures below show the distribution after replacing xxx with NA. As shown, we still have an issue with outliers.

(Show summary table here??)



We assume that many of these values are error measurements. Applying a treshold of 2,5 percent at each end of the variables' distribution, we replace values exceeding this treshold with NAs.

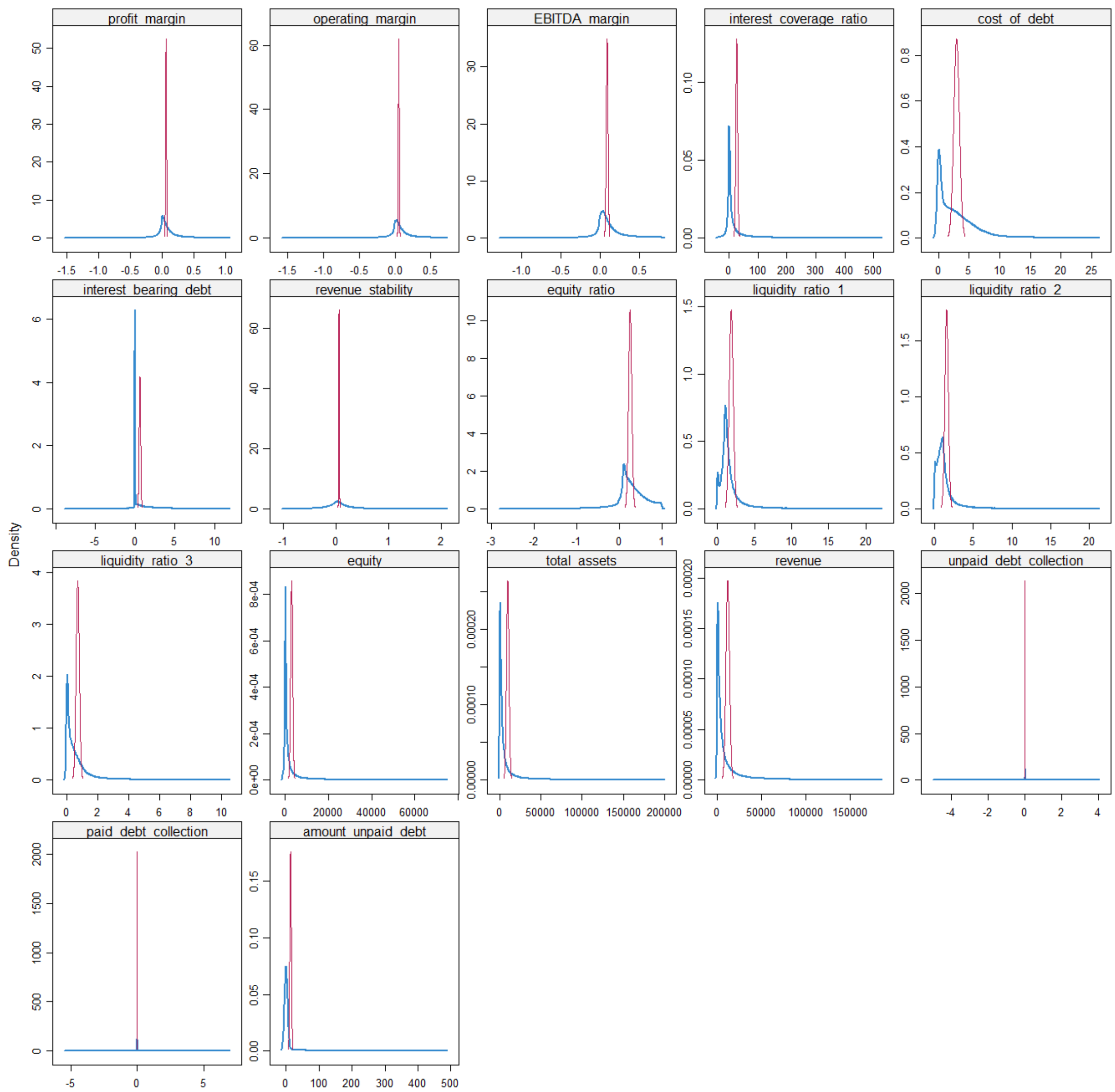
Mia: Might mention that equity ratio stability seems to have exactly same distribution as equity ratio. We test for correlation etc etc and end up removing this variable moving forward. Saves some computation time for r when imputing.

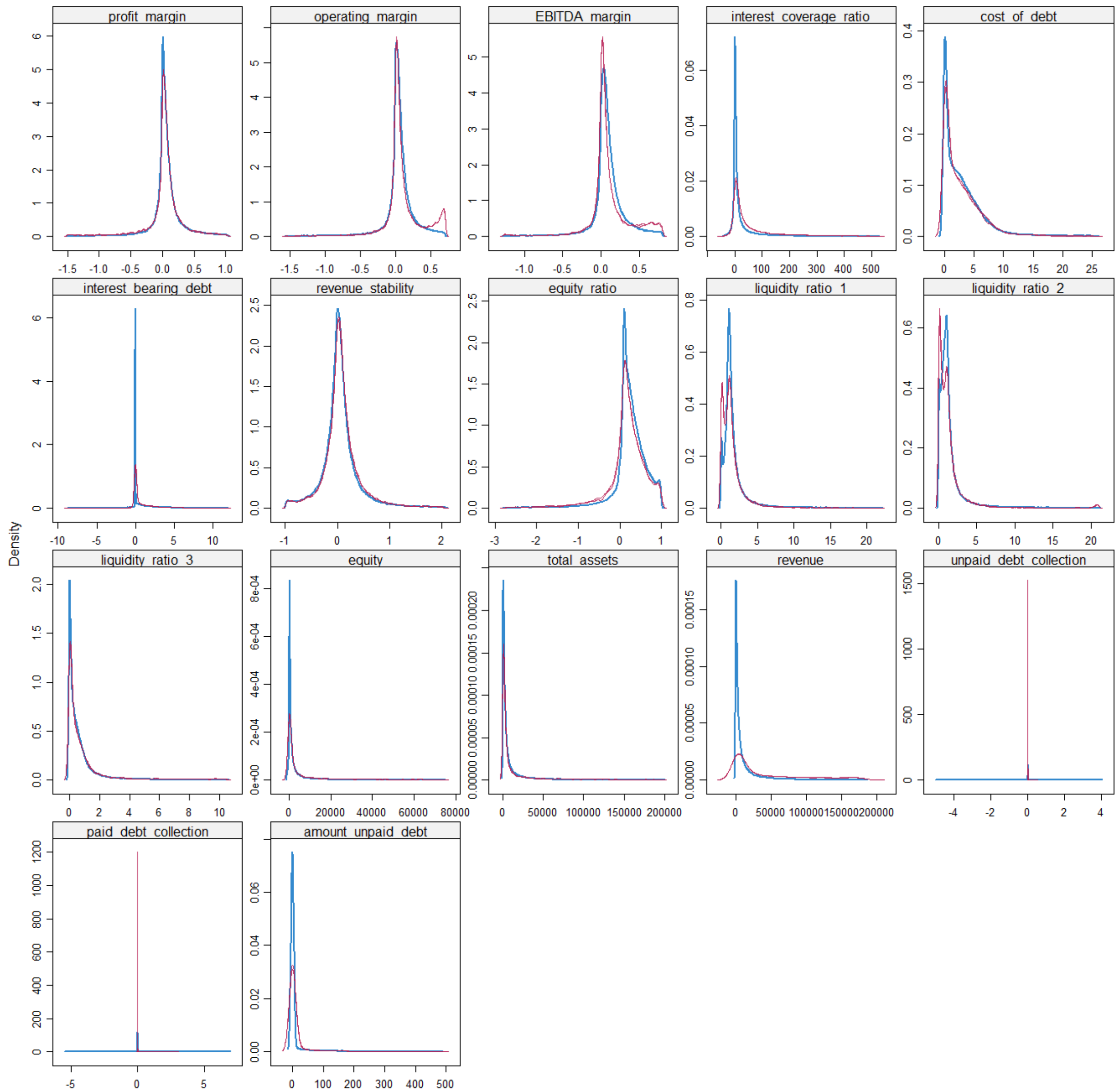
Imputation

As we have as much as xxx NA's, we choose not to delete these values, but rather impute them using the MICE package.

We deploy two methods for imputation: mean and ppm. Explain: - ppm and why it is good - why imputation can introduce challenges -

The tables below show how the distributions change when we apply different imputation methods. The original data is shown in the blue line, the mean imoutation in xxx line, and the pmm method in the xxx line.





	1	2
Orginal	128070	23
Imputed	128070	23

As shown in the distribtion plots, there is not much variation in the variables measuring paid and unpaid debt collection. We generate two new dummy variables that provide two binary measures of paid and unpaid debt. Moreover,

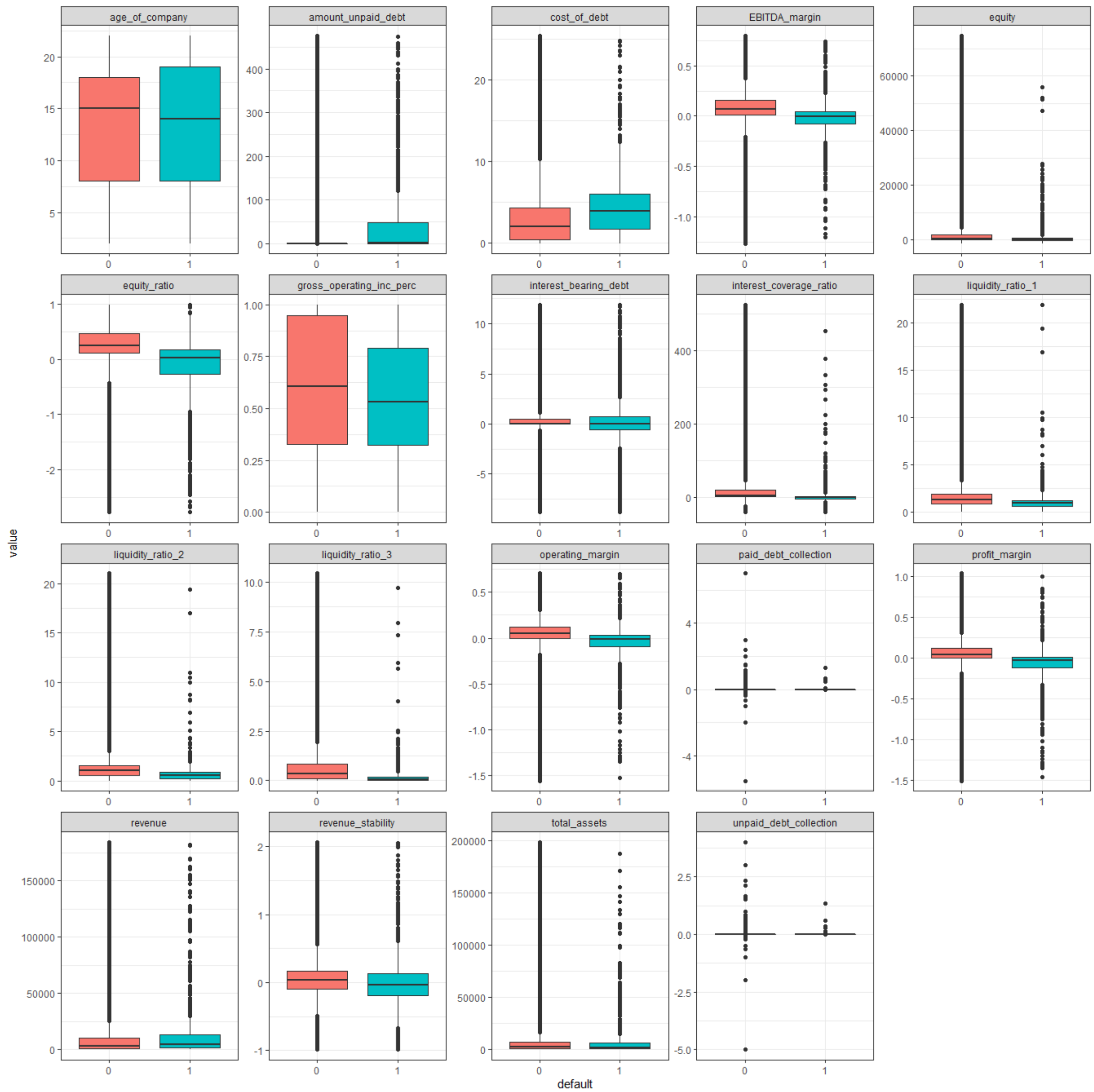
The table below shows how this variable is distributed. As shown, defaulting firms are more frequently represented among those with debt collection. Moreover, firms who have reported paying down previous debt are less frequently represented among defaulters.

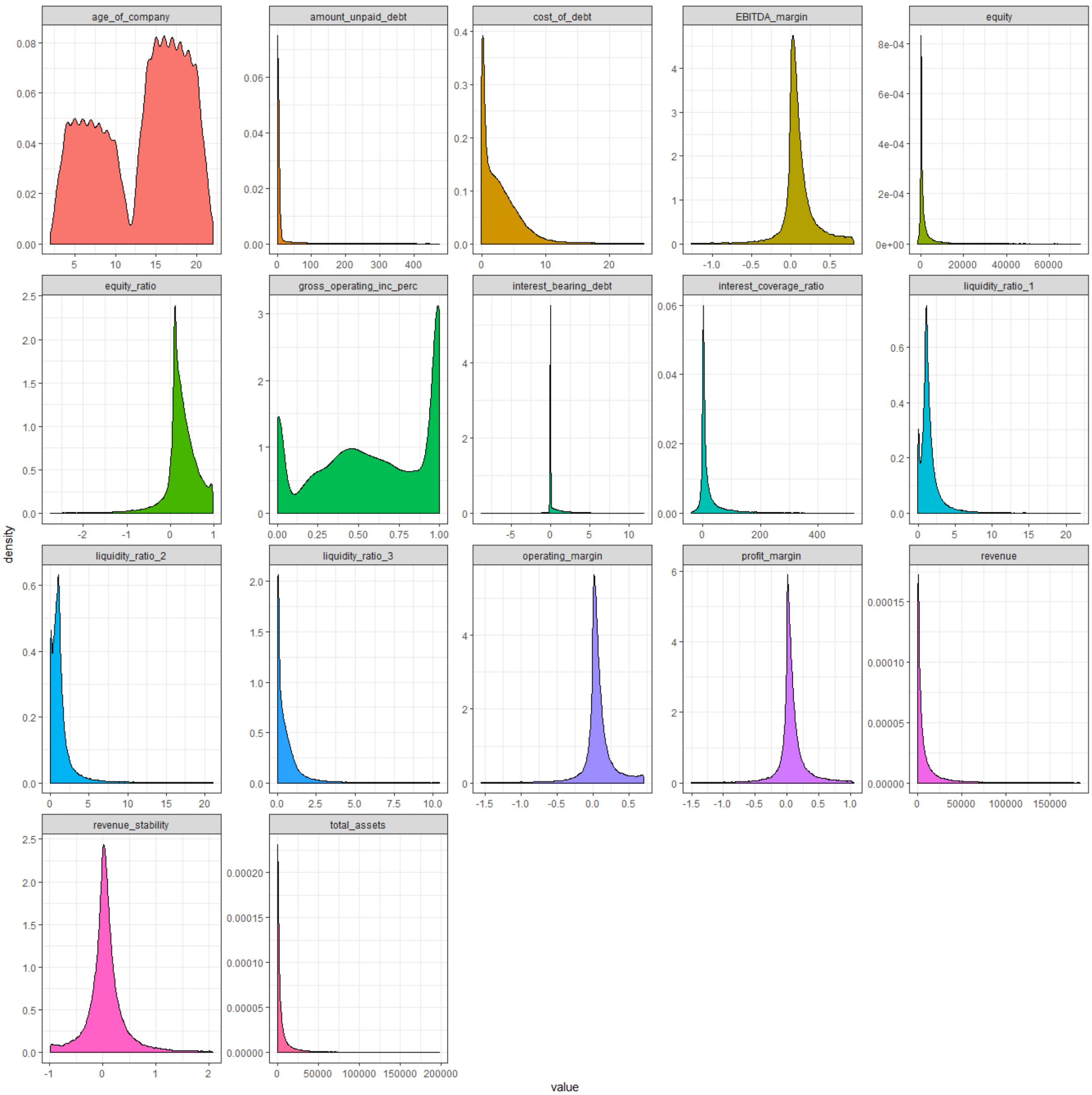
Mia: Might need to work on the reasoning behind generating this dummy a bit more. Could we do without it?

	0	1
0	97469	29200
1	449	952

	0	1
0	109170	17499
1	922	479

After cleaning, imputing and restructuring our data set is better suited for prediciton modelling, see density plots below.





#Modelling preparations A few more steps before we are ready to start modelling:

We split the data frame into a training and test set. The variables total_assets, revenue, industry and paid_debt_collection are removed as they correlate with other independent variables.

[1] TRUE

	1
Train defaults	0.01
Test defaults	0.01

Dealing with data imbalance

As mentioned, defaulting firms are highly underrepresentated in the data set. We deal with this by implementing the oversamlpling technique “smote”. - Writ some sentences about smote.

#Model 1: GLM

Our first prediction model is a logistic regression model. - Some words about variable selection

Summary statistics are presented below.

Comments:

- What variables are significant?
- Do they make economically sense?

The plot below shows the variable importance of the independent variables in the glm model. - Comments: what variables perform well

Call: NULL

Deviance Residuals: Min 1Q Median 3Q Max
-3.2735 -0.5719 -0.2062 0.5334 3.5894

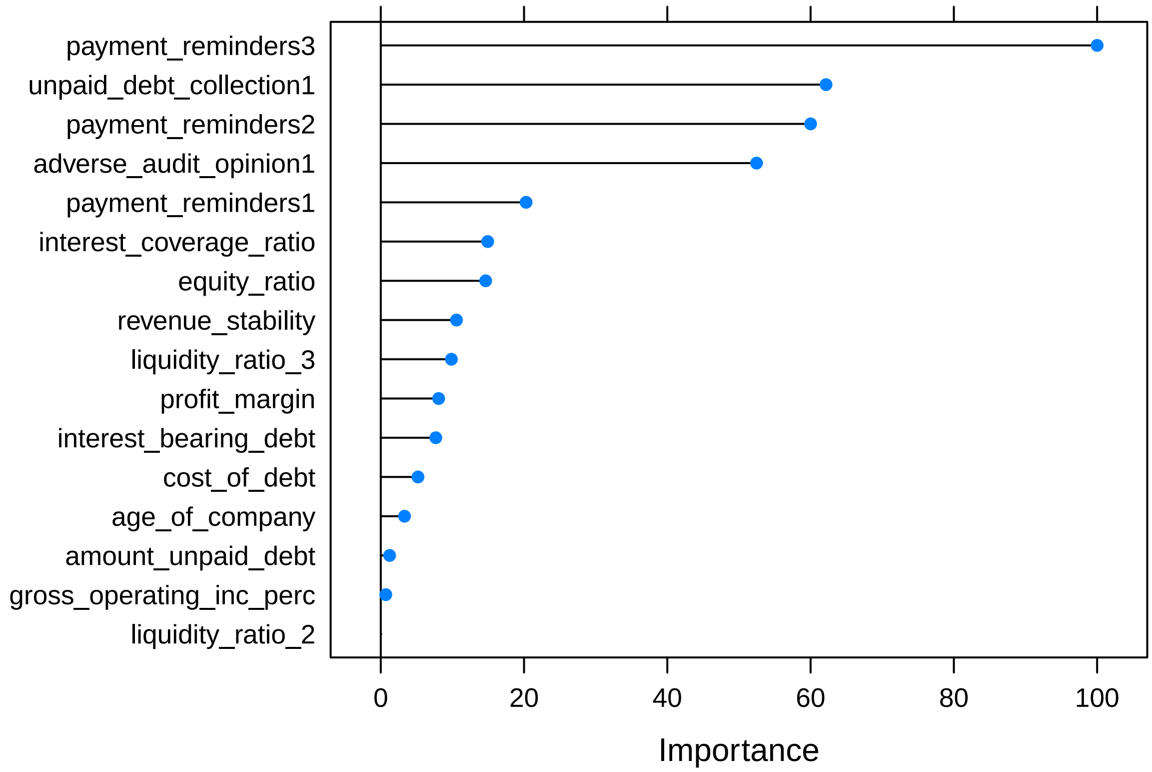
Coefficients: Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.4972174 0.1449764 -17.225 < 2e-16 ***profit_margin -0.7085471 0.1700292 -4.167 3.08e-05*** gross_operating_inc_perc -0.2625578 0.1182611 -2.220 0.026408 *
interest_coverage_ratio -0.0071384 0.0011950 -5.973 2.32e-09 ***cost_of_debt 0.0375007 0.0110134 3.405 0.000662*** interest_bearing_debt -0.0601575 0.0148104 -4.062 4.87e-05 ***revenue_stability -0.4813783 0.0997921 -4.824 1.41e-06*** equity_ratio -0.5196414 0.0880969 -5.899 3.67e-09 ***liquidity_ratio_2 -0.0874081 0.0429613 -2.035 0.041893***
liquidity_ratio_3 -0.4104520 0.0885584 -4.635 3.57e-06 *age_of_company 0.0193731 0.0066584 2.910 0.003619* ***unpaid_debt_collection1 1.4323496 0.0776961 18.435 < 2e-16*** adverse_audit_opinion1 1.2293424 0.0774344 15.876 < 2e-16 ***amount_unpaid_debt 0.0012354 0.0005232 2.361 0.018213***
payment_reminders1 0.6614652 0.0895456 7.387 1.50e-13 *payment_reminders2 1.7178734 0.0961480 17.867 < 2e-16* **payment_reminders3 3.1267094 0.1100188 28.420 < 2e-16** * — Signif. codes: 0 ‘***’ 0.001 ’’*** *0.01* ” 0.05 ‘.’ 0.1 ’ ’ 1

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 9379.1  on 6866  degrees of freedom
```

Residual deviance: 5241.7 on 6850 degrees of freedom AIC: 5275.7

Number of Fisher Scoring iterations: 6



Confusion Matrix and Statistics

```
      Reference
```

Prediction 0 1 0 32849 93 1 5151 327

```
           Accuracy : 0.8635
             95% CI : (0.86, 0.8669)
No Information Rate : 0.9891
P-Value [Acc > NIR] : 1

              Kappa : 0.0925
```

Mcnemar's Test P-Value : <2e-16

```
        Sensitivity : 0.86445
        Specificity : 0.77857
     Pos Pred Value : 0.99718
     Neg Pred Value : 0.05969
         Prevalence : 0.98907
     Detection Rate : 0.85500
```

Detection Prevalence : 0.85742
Balanced Accuracy : 0.82151

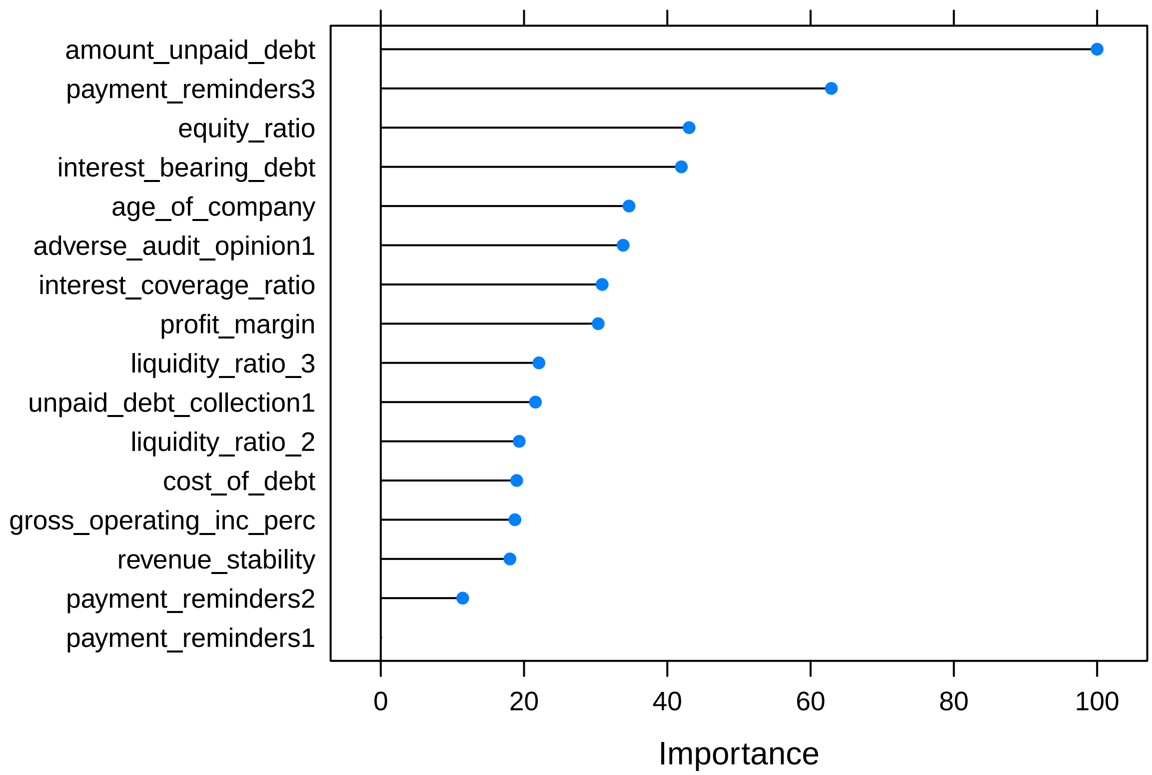
```
'Positive' Class : 0
```

Comments:

Discuss the confusion matrix.

Random forest

Our second predicion model is a random forest model.



Confusion Matrix and Statistics

```
      Reference
```

Prediction 0 1 0 34738 139 1 3262 281

```
               Accuracy : 0.9115
                 95% CI : (0.9086, 0.9143)
    No Information Rate : 0.9891
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1247
```

Mcnemar's Test P-Value : <2e-16

```
        Sensitivity : 0.91416
        Specificity : 0.66905
     Pos Pred Value : 0.99601
     Neg Pred Value : 0.07931
         Prevalence : 0.98907
     Detection Rate : 0.90416
```

Detection Prevalence : 0.90778
Balanced Accuracy : 0.79160

```
   'Positive' Class : 0
```

Comments: About the confustion matrix How well the model performs relative to glm Accuracy, specificity and sensitivity etc.

model_xgb <- readRDS("xgb.Rdata")

model_xgb

plot(varImp(model_xgb))

xgb_pred <- data.frame(actual = test_data$default, predict(model_xgb, newdata = test_data, type = "prob"))

rf_pred$predict < -ifelse(xgb_pred$X1 > 0.5, 1, 0) xgb_pred$predict < -as.factor(xgb_pred$predict)

cm_xgb <- confusionMatrix(xgb_pred$predict, test_data$default) cm_xgb

ROC curve xgb

result.predicted.prob <- predict(model_xgb, test_data, type="prob") # Prediction

result.roc <- roc(test_data$default, result.predicted.prob$1) # Draw ROC curve.

plot(result.roc, print.thres="best", print.thres.best.method="closest.topleft")

result.coords <- coords(result.roc, "best", best.method="closest.topleft", ret=c("threshold", "accuracy")) print(result.coords)#to get threshold and accuracy

Look at at difference all together

Look at the performance

models <- list(glm = model_glm, rf = model_rf, xgb = model_xgb)

resampling <- resamples(models)

bwplot(resampling)

density plots of accuracy

scales <- list(x=list(relation="free"), y=list(relation="free")) densityplot(resampling, scales=scales, pch = "|", allow.multiple = TRUE)

Other snacks for comparing

splom(resampling)

xyplot(resampling, models=c("rf", "xgb"))

summary(resampling)