

Modelling Competition

Regression Models

Javier Esteban Aragoneses
Mauricio Marcos Fajgenbaun
Paerhati Piluolan
Danyu Zhang
Daniel Alonso

Master in Statistics for Data Science
Universidad Carlos III de Madrid



Academic Year 2020/2021

Modelling Competition

The Regressors

January 18th, 2021

Basic exploratory analysis

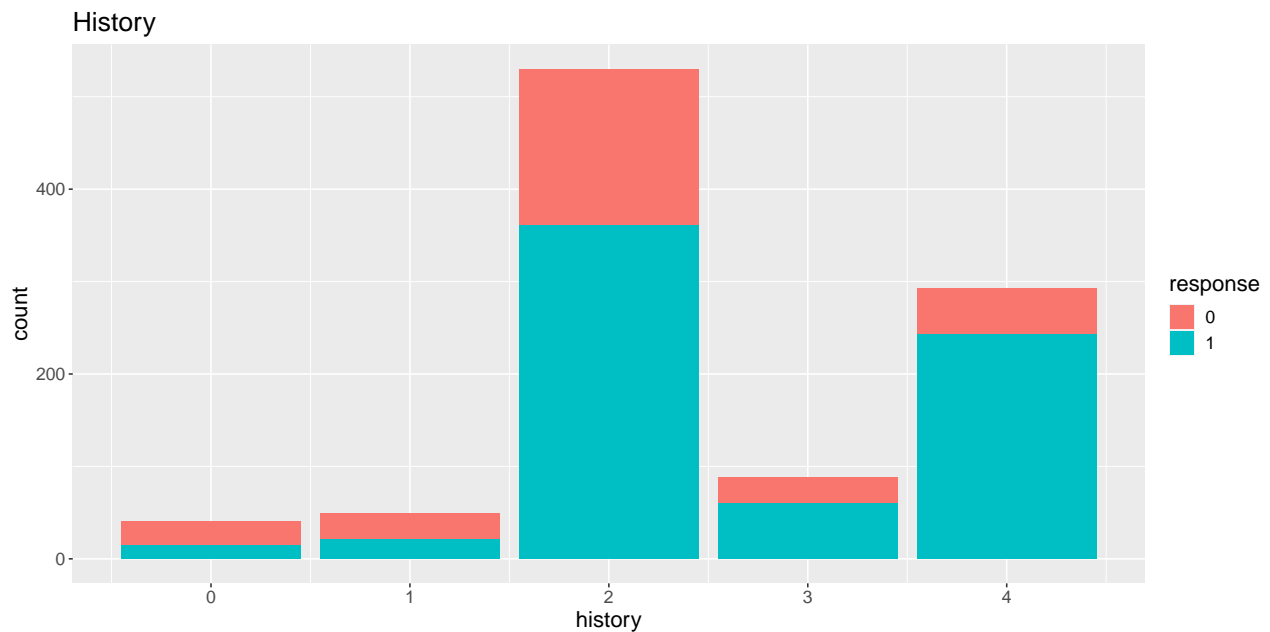
Importing and manipulating the data:

```
# import data
credit <- read.csv('./data/credit.csv')

# change names to lowercase
names(credit) <- tolower(names(credit))

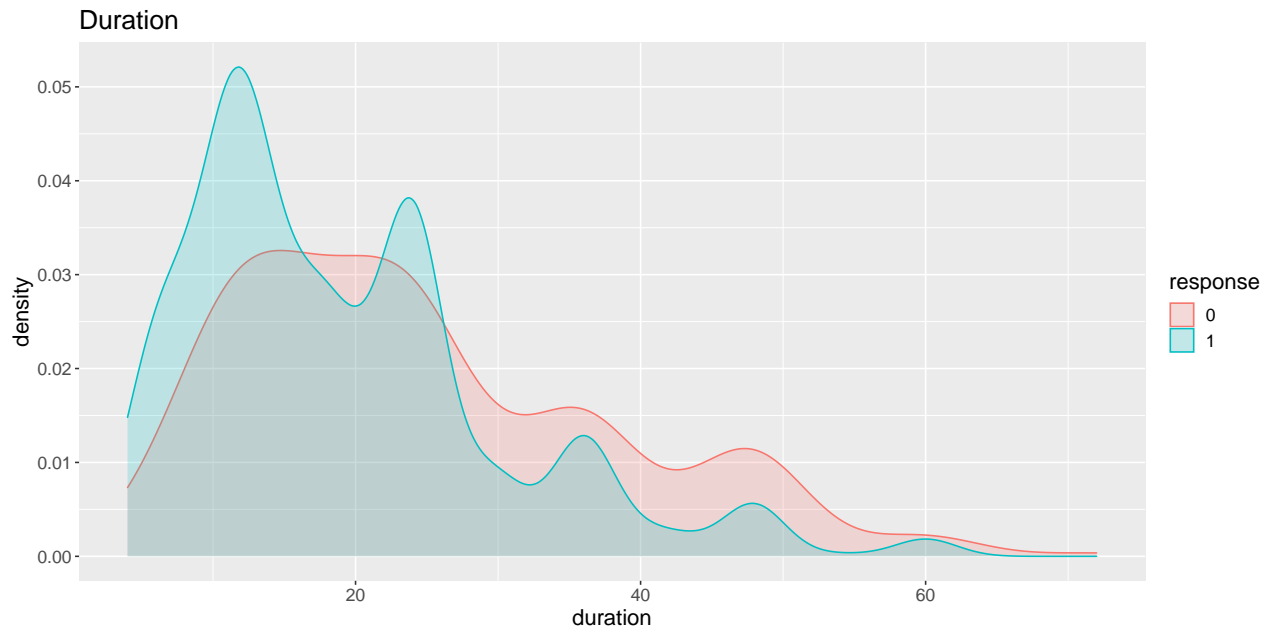
# convert response into a factor
credit$response <- as.factor(credit$response)
```

Visualizing our most important variables



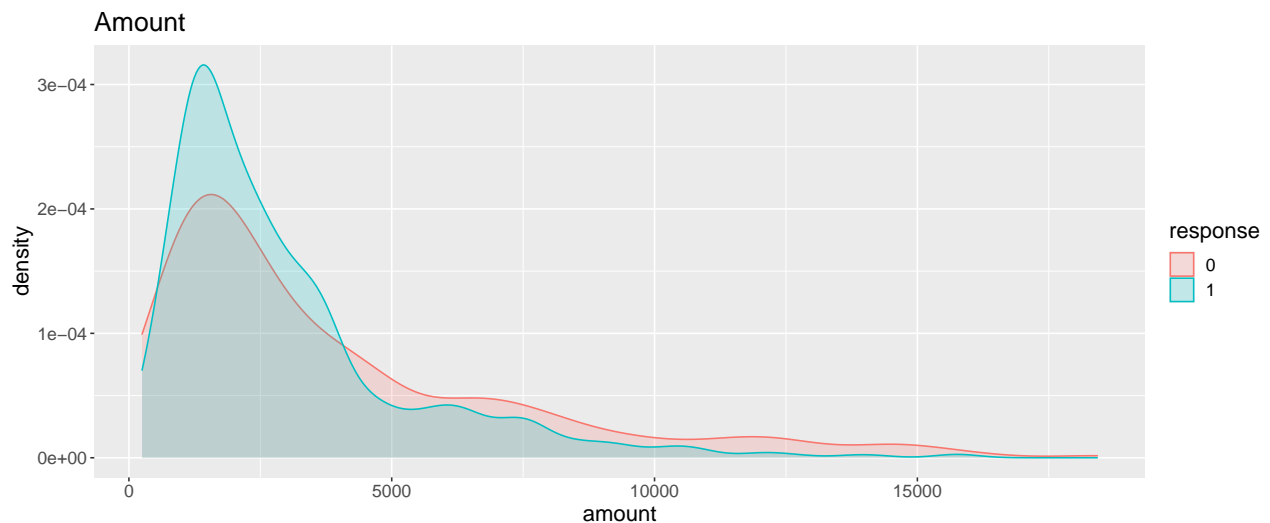
We can see the distribution of clients without good credit is not quite discernible for credit history. We can see that clients that tend to have good credit history are more likely to have a good credit rating. And we see the higher the amount of credits the client has paid duly, the more likely the client will have good credit rating.

However, delaying paying a past credit doesn't necessarily mean that the client will not have a good credit rating. In fact, the category with the most clients (credit history category 2) has a higher proportion of accounts without good credit rating than the general proportion of clients without good credit rating (30%).



In this plot we can see that clients with good credit rating tend to pay credits sooner on average. Clients without a good credit rating follow a similar trend, where the majority of them will take credits for less time, however, they will also, on average, take longer to pay their credits due.

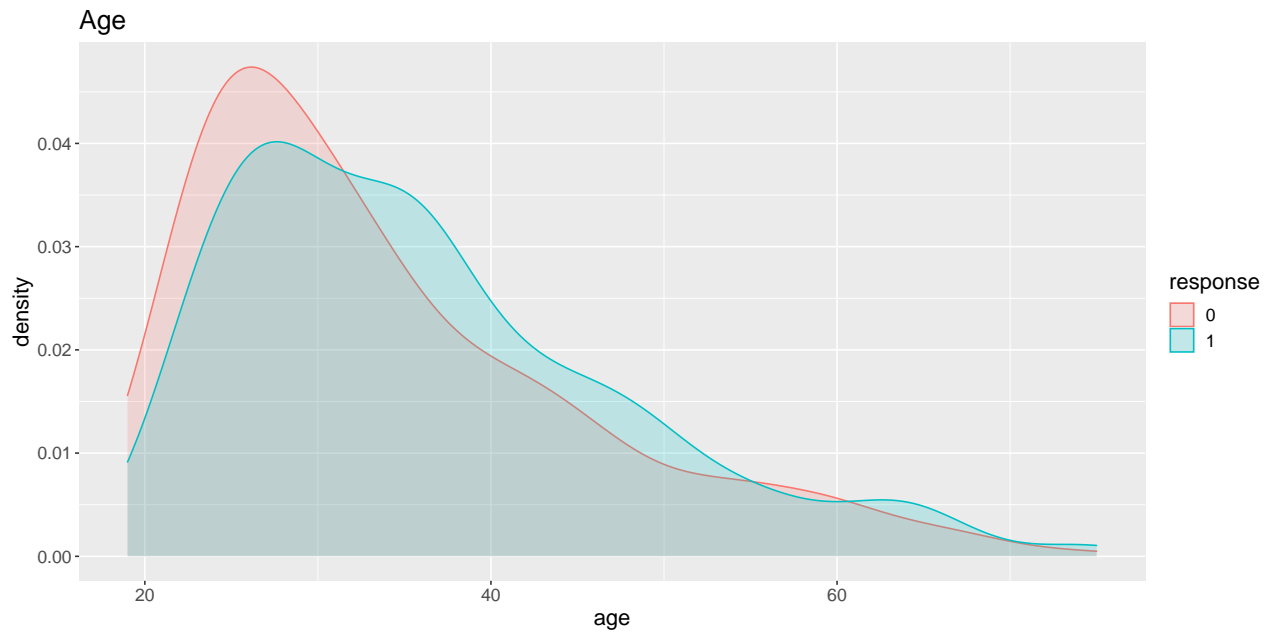
There's about 5 peaks which diminish in size the larger the duration gets, where we see large accumulation of clients which both have and don't have a good credit rating. This highlights that the institution handing out these credits might have fixed timeframes for credit payments which clients are probably offered by default.



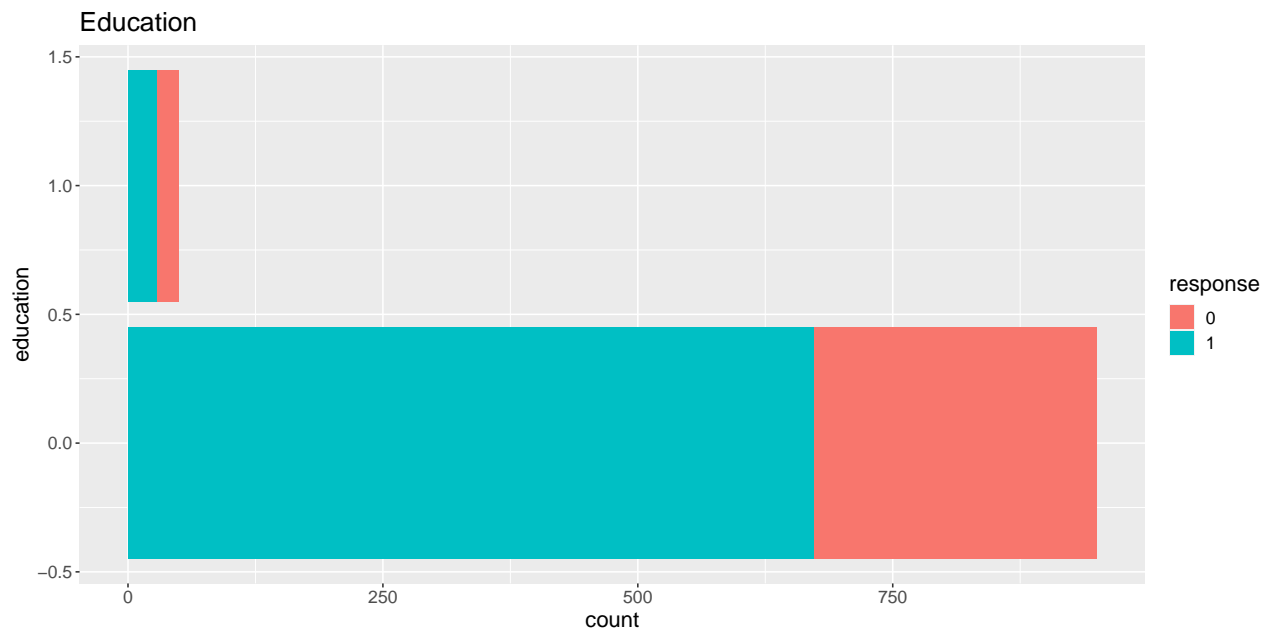
For amounts we notice a trend, where clients with better credit rating also tend to ask for smaller amounts. The plot is strongly right-skewed for both good credit rating clients and the opposite.

It seems likely that clients with worse credit rating tend to ask for larger sums which they might, in the future, have more difficulty paying back. Although we might think that the opposite should be true, this ties well with the duration of credits, where clients with worse credit rating tended to ask for longer timeframes to pay their credits due. And it makes sense that larger amounts should take longer to pay (on average).

It also seems likely that a good way to build a good credit rating could be to ask for smaller amounts, which should theoretically be easier to pay.



In the following plot, we can see that the distributions for both groups (clients with and without good credit rating) are very similar, but in general, clients with good credit rating tend to be older than the clients with lesser credit rating. We also notice the plot is significantly right-skewed towards younger ages, perhaps because clients of younger age are incentivized to use financial services and probably might need more credits than those older clients (which tend to have less financial needs as their salaries might be higher).



In the plot we can observe that, for both groups of education level, the proportion of being a good or bad client is very similar.

Testing 1-variable models (with GLM)

We test 1-variable models to assess which variables (by themselves) have the highest predictive power:

Table 1: best models

| | vars | accuracy |
|----|----------------|----------|
| 3 | history | 0.71375 |
| 2 | duration | 0.70125 |
| 10 | amount | 0.70125 |
| 22 | age | 0.70000 |
| 17 | co.applicant | 0.69625 |
| 8 | education | 0.69500 |
| 14 | male_div | 0.69375 |
| 12 | employment | 0.68875 |
| 1 | chk_acct | 0.68250 |
| 21 | prop_unkn_none | 0.68000 |

We can see that (individually), the best variables that we obtain are *history*, *duration*, *amount*, *age*, *co.applicant*, *education*, *male_div*, *employment*, *chk_acct* and *prop_unkn_none*.

Logistic regression (generalized linear model)

Basic variable selection

We remove *obs*, given that it's just an index and we remove *real_estate* and *own_res* as they are essentially equivalent to *prop_unkn_none* and *rent*.

```
#> [1] "chk_acct"      "duration"      "history"       "new_car"
#> [5] "used_car"      "furniture"     "radio.tv"      "education"
#> [9] "retraining"    "amount"        "sav_acct"      "employment"
#> [13] "install_rate"  "male_div"      "male_single"   "male_mar_or_wid"
#> [17] "co.applicant"  "guarantor"     "present_resident" "prop_unkn_none"
#> [21] "age"           "other_install" "rent"          "num_credits"
#> [25] "job"           "num_dependents" "telephone"     "foreign"
#> [29] "response"
```

Normalizing variables

Given that *amount*, *age* and *duration* are NOT normally distributed, we take their log (we also sum 1 to avoid errors with zeros):

```
credit$amount <- log(credit$amount + 1)
credit$age <- log(credit$age + 1)
credit$duration <- log(credit$duration + 1)
```

Train and test split

We use a randomly selected seed (12) and perform a train and test split with 80% train and 20% test values. We also convert the response variable of each subset of data into a factor.

```
set.seed(12)
spl = caret::createDataPartition(credit$response, p = 0.8, list = FALSE)
Train = credit[spl,]
Test = credit[-spl,]
Train$response <- as.factor(Train$response)
Test$response <- as.factor(Test$response)
```

Functions to aid the modelling and variable selection process

Code for the functions is omitted due to how long they are, to see the code, view Rmd file

Formula creation function

We defined a function to obtain the formula of all models with and without a chosen amount of interactions (2-way, 3-way, etc). The function generates all combinations of variables of a selected size (2 vars, 3 vars, etc) and puts them into a formula neatly, including or excluding their interactions.

Modelling function

We create a function that runs all the models using a chosen dataset, utilizing a list of formulas created by the previously defined function.

Function to testing models with and without interactions

We define a function to perform LRT for models with and without interactions and we check the significance of such interactions. If the p-value displayed in the dataframe is above a certain threshold we deem those interactions less useful.

We also calculate the AIC of each model (with and without interactions) to check if the interaction is not just significant but also improves the model to an extent.

Scoring function

We define a function that will score the models using several different metrics:

- ROC Area under curve (using training set)
- ROC sensitivity
- ROC specificity
- Accuracy from a confusion matrix

Model selection process

Running a model with all the variables

We run a model with all the variables and no interactions, therefore we can roughly see how the model is standing before any significant variable selection.

```
#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0  40  24
#>      1  20 116
#>
#>      Accuracy : 0.78
#>      95% CI : (0.7161, 0.8354)
#>      No Information Rate : 0.7
#>      P-Value [Acc > NIR] : 0.007151
#>
#>      Kappa : 0.486
#>
#>      Mcnemar's Test P-Value : 0.651077
#>
#>      Sensitivity : 0.6667
#>      Specificity : 0.8286
#>      Pos Pred Value : 0.6250
#>      Neg Pred Value : 0.8529
#>      Prevalence : 0.3000
#>      Detection Rate : 0.2000
#>      Detection Prevalence : 0.3200
#>      Balanced Accuracy : 0.7476
#>
#>      'Positive' Class : 0
#>
```

Using all our variables we reach an accuracy of 0.78, which as a starting point is decent, but this can improve.

Testing models with and without variables to check their significance

Here we perform LRT to check the significance of individual variables. We assess whether the p-value of the test shows significance of each model compared to the one without a specific variable (looping through them):

Table 2: best models

| | vars | scores | aic |
|----|------------------|-----------|----------|
| 16 | male_mar_or_wid | 0.8842872 | 792.4139 |
| 6 | furniture | 0.8118166 | 792.4494 |
| 19 | present_resident | 0.5815776 | 792.6964 |
| 9 | retraining | 0.5454180 | 792.7583 |
| 10 | amount | 0.5426324 | 792.7634 |
| 17 | co.applicant | 0.4271850 | 793.0232 |
| 7 | radio.tv | 0.3475789 | 793.2750 |
| 26 | num_dependents | 0.3434658 | 793.2902 |
| 8 | education | 0.2769097 | 793.5749 |
| 25 | job | 0.2525847 | 793.7017 |

This allows us to remove *furniture* and *male_mar_or_wid* which seem to not be significant given their extremely high p-value, we keep the rest of the variables regardless of their result as they are not as notoriously high as *furniture* and *male_mar_or_wid* and their interactions with other variables might make them useful.

Testing 2-variable models with and without interactions

We create models with all the combinations of 2 variables and then we perform LRT for models with and without interactions between their variables. Then we select models with an LRT p-value under 0.005, in order to keep the most important interactions.

We run the tests:

```
two_var_combs <- test(formulas_with, formulas_without, models_with, models_without)
```

We remove NAs, given that these interactions' product is 0 for all values, the LRT returns a p-value of 1 (meaning there's no difference between the models).

We present the table showing the model formulas and the p-values:

Table 3: best models

| | formulas_with | formulas_without | pvals | aic_with | aic_without |
|-----|--|------------------------------------|-----------|----------|-------------|
| 212 | response~(employment+prop_unkn_none)^2 | response~employment+prop_unkn_none | 0.0039463 | 950.6366 | 956.9450 |
| 120 | response~(radio.tv+employment)^2 | response~radio.tv+employment | 0.0034811 | 956.4306 | 962.9670 |
| 66 | response~(history+other_install)^2 | response~history+other_install | 0.0032266 | 927.7634 | 934.4381 |
| 33 | response~(duration+sav_acct)^2 | response~duration+sav_acct | 0.0030212 | 903.7032 | 910.4979 |
| 259 | response~(male_single+foreign)^2 | response~male_single+foreign | 0.0025497 | 968.1847 | 975.2893 |
| 322 | response~(job+foreign)^2 | response~job+foreign | 0.0008878 | 969.2991 | 978.3472 |
| 188 | response~(amount+telephone)^2 | response~amount+telephone | 0.0008727 | 958.6856 | 967.7654 |
| 165 | response~(retraining+age)^2 | response~retraining+age | 0.0007533 | 967.0877 | 976.4408 |
| 98 | response~(used_car+amount)^2 | response~used_car+amount | 0.0006894 | 948.6639 | 958.1815 |

Testing a model using the best 2-way interactions

We test a model using all the previously ran variables (all the variables in the dataset) but we include the interactions selected as most meaningful:

```
#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0  39  19
#>      1  21 121
#>
#>      Accuracy : 0.8
```

```
#>          95% CI : (0.7378, 0.8531)
#>    No Information Rate : 0.7
#>    P-Value [Acc > NIR] : 0.0009283
#>
#>          Kappa : 0.5192
#>
#> Mcnemar's Test P-Value : 0.8743671
#>
#>          Sensitivity : 0.6500
#>          Specificity : 0.8643
#>    Pos Pred Value : 0.6724
#>    Neg Pred Value : 0.8521
#>    Prevalence : 0.3000
#>    Detection Rate : 0.1950
#>    Detection Prevalence : 0.2900
#>    Balanced Accuracy : 0.7571
#>
#>
#>    'Positive' Class : 0
#>
```

Looking at the confusion matrix the model didn't improve much given that we have essentially the same accuracy, but we have higher sensitivity, therefore we continue the process.

Testing 3-variable models with and without interactions

We create models with all the combinations of 3 variables and then we perform LRT for models with and without interactions. Then we select models with an LRT p-value under 0.0005, in order to keep the most important interactions.

We run the tests:

```
three_var_combs <- test(formulas_with, formulas_without, models_with, models_without)
```

We remove NAs, given that these interactions' product is 0 for all values, therefore, the LRT returns a p-value of 1 (meaning there's no difference between the models).

```
three_var_combs <- na.omit(three_var_combs[order(-three_var_combs$pvals),])
best_3_vars <- three_var_combs[three_var_combs$pvals < 0.0005,]
```

We present the table showing the model formulas and the p-values:

Table 4: best models

| | formulas_with | formulas_without | pvals | aic_with | aic_without |
|------|--|--|-----------|----------|-------------|
| 1132 | response~(used_car+amount+num_dependents)^3 | response~used_car+amount+num_dependents | 0.0004517 | 947.8746 | 960.0953 |
| 1300 | response~(radio.tv+retraining+age)^3 | response~radio.tv+retraining+age | 0.0003828 | 955.7608 | 967.4969 |
| 1133 | response~(used_car+amount+telephone)^3 | response~used_car+amount+telephone | 0.0002417 | 942.4771 | 956.0685 |
| 141 | response~(chk_acct+retraining+other_install)^3 | response~chk_acct+retraining+other_install | 0.0002268 | 871.4970 | 885.2273 |
| 1082 | response~(used_car+education+amount)^3 | response~used_car+education+amount | 0.0002112 | 943.4790 | 956.4047 |
| 745 | response~(history+male_div+other_install)^3 | response~history+male_div+other_install | 0.0001857 | 918.7066 | 932.8728 |
| 439 | response~(duration+amount+telephone)^3 | response~duration+amount+telephone | 0.0000153 | 920.9474 | 940.5122 |

We also see that, as with 2-variable interactions, the fact that an interaction is meaningful also improves the AIC of a model.

Testing a model using the best 3-way and 2-way interactions

We also use all the variables as we did previously, however, we include the interactions selected:

```
#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0  38  16
#>      1  22 124
#>
#>
#>          Accuracy : 0.81
#>          95% CI : (0.7487, 0.8619)
#>    No Information Rate : 0.7
#>    P-Value [Acc > NIR] : 0.0002813
#>
#>          Kappa : 0.5343
#>
#> Mcnemar's Test P-Value : 0.4173042
#>
#>          Sensitivity : 0.6333
```



```

#>      Specificity : 0.8857
#>      Pos Pred Value : 0.7037
#>      Neg Pred Value : 0.8493
#>      Prevalence : 0.3000
#>      Detection Rate : 0.1900
#>      Detection Prevalence : 0.2700
#>      Balanced Accuracy : 0.7595
#>
#>      'Positive' Class : 0
#>

```

We see the model improves its accuracy by about 1%, this is an acceptable increase, however, we can keep trying to improve it.

We remove the variables *radio.tv:retraining:age* and *used_car:amount:education* given that they have NA coefficients:

```

#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0  40  17
#>      1  20 123
#>
#>      Accuracy : 0.815
#>      95% CI : (0.7541, 0.8663)
#>      No Information Rate : 0.7
#>      P-Value [Acc > NIR] : 0.0001479
#>
#>      Kappa : 0.5531
#>
#>      Mcnemar's Test P-Value : 0.7423084
#>
#>      Sensitivity : 0.6667
#>      Specificity : 0.8786
#>      Pos Pred Value : 0.7018
#>      Neg Pred Value : 0.8601
#>      Prevalence : 0.3000
#>      Detection Rate : 0.2000
#>      Detection Prevalence : 0.2850
#>      Balanced Accuracy : 0.7726
#>
#>      'Positive' Class : 0
#>

```

We see our accuracy has increased to about 81.5% after optimizing using stepAIC and removing the variables that have a very high standard error (*used_car:education:amount*, *radio.tv:retraining:age*, *foreign*, *male_single:foreign*, *job:foreign*)

The final model we have chosen for our GLM model section is the model that uses the following formula:

```

response ~ used_car + radio.tv + chk_acct + history + duration + new_car + retraining + amount
+ sav_acct + employment + guarantor + prop_unkn_none + age + other_install + rent + telephone
+ employment:prop_unkn_none + radio.tv:employment + history:other_install + duration:sav_acct
+ amount:telephone + retraining:age + used_car:amount + chk_acct:retraining:other_install + dura-
tion:amount:telephone

```

Summary of our final GLM

1. Duration: As the exponential of the coefficient for the variable is less than 1, it means that the higher the value of the duration, the higher the probability of being a bad client.
2. History: As the coefficient for the variable history is positive, we can see the odds ratio will be greater than 1. Which means that the more the value of history, the higher the probability of being a good client.
3. Age: The coefficient is positive, so the odds ratio will be greater than 1, which means that the older the client is, the better the client is, and that corresponds to our exploratory analysis.

Generalized additive model penalized splines

We import and manipulate the data, create dummy variables for the categorical variables and we perform a train-test split for 80% train and 20% test.

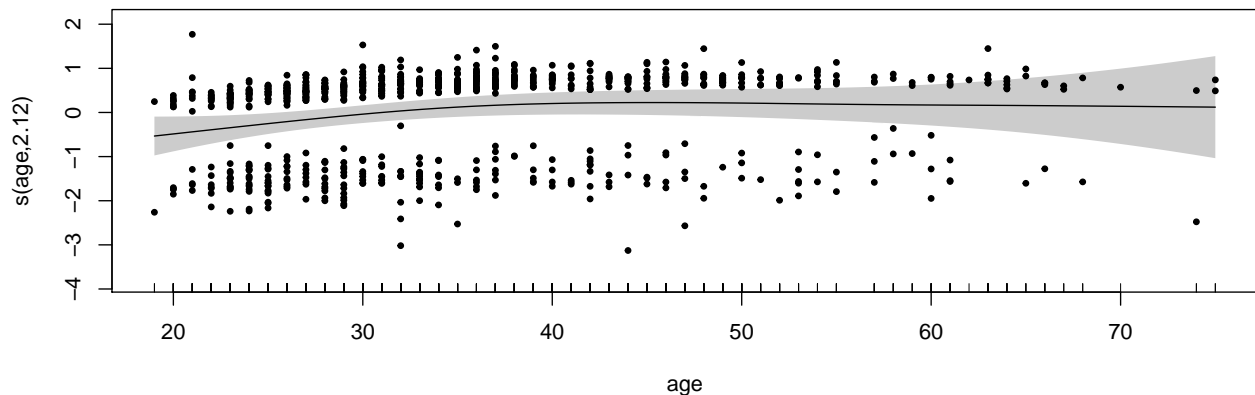
In this part of the work, we will use a generalized additive model (with penalized splines) in order to estimate, and predict a classification for the costumers of the bank. We believe this is a good idea, as by doing this we

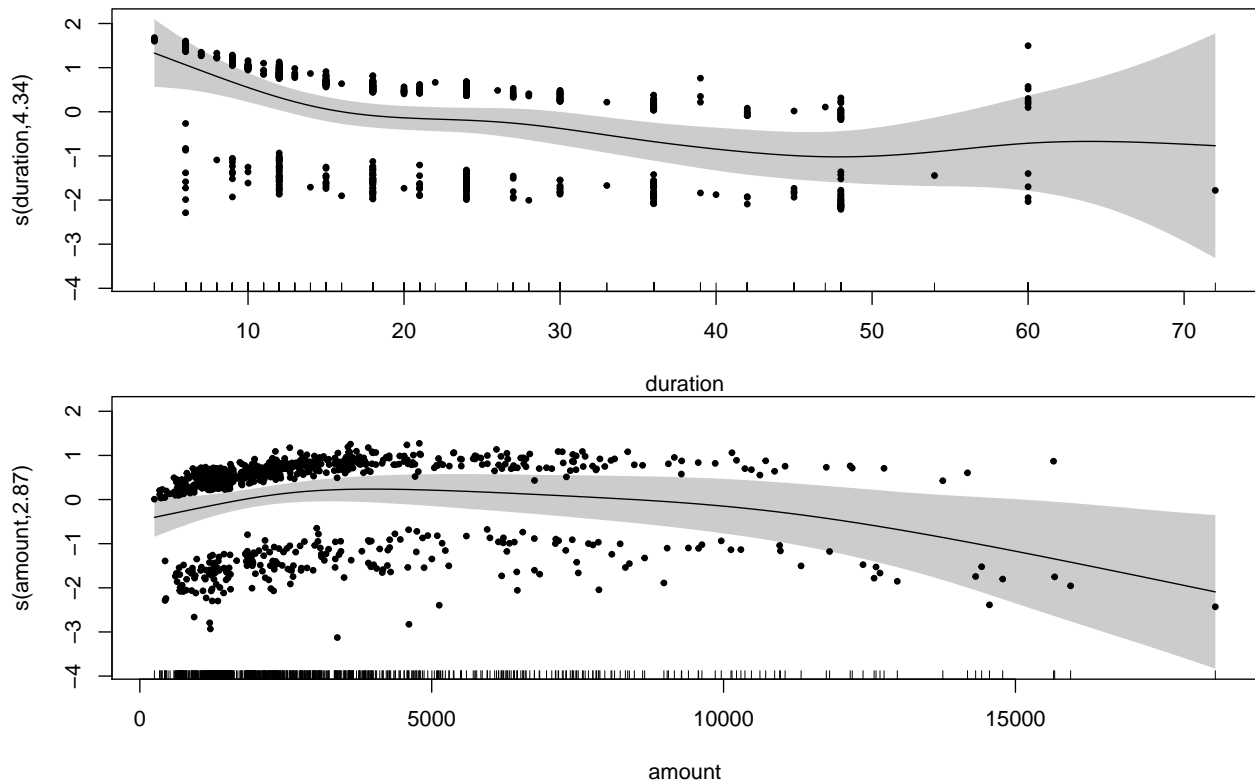
can relaxate the linearity assumption between the response variable and some of our predictors. We first started by fitting a gam model with only three variables (the three continuous) and checked different criteria. As most of our categories are categorical, we first explore a model with the continuous one to check their relevance in the model.

In general, our procedure is based on the following criteria. First, we check the coefficients estimation in the model and see if it is significant (and to what level of confidence it is).

Then we start adding variables and interactions and compare the following: Akaike value (we keep the one with a lower value of akaike, as it penalizes by degrees of freedom), we plot a ROC curve and check the AUC (that is the predictive power that our model has, and we keep on working with the model that has a higher level of AUC: that is, the ROC is closer to the sides of the square of area equal to 1), we build a confusion matrix with both training and testing set and check how many of each of the errors our model produces when predicting and what percentage of errors taking into account all the predictions, and finally we test it with Anova (LRT), using a Chisq test to see if the extra term is significant or we can omit it. We also check the explained deviance of each model. When evaluating the change in all this criteria, we decide if we continue trying with the same model or change it for another one a bit more complex. And so on.

```
fit_gam=logit1 <- gam(response~s(age)+s(duration)+s(amount),family=binomial,select=TRUE,data=creditTrain)
summary(fit_gam)
#>
#> Family: binomial
#> Link function: logit
#>
#> Formula:
#> response ~ s(age) + s(duration) + s(amount)
#>
#> Parametric coefficients:
#>             Estimate Std. Error z value Pr(>|z|)
#> (Intercept)  0.90748    0.08175   11.1   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>             edf Ref.df Chi.sq  p-value
#> s(age)         2.122     9  8.244  0.0112 *
#> s(duration)    4.342     9 27.488 3.65e-07 ***
#> s(amount)     2.867     9  8.376  0.0219 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj)  =  0.0671   Deviance explained = 6.49%
#> UBRE = 0.16826   Scale est. = 1          n = 800
plot(fit_gam,shade=TRUE,seWithMean=TRUE,pch=19,cex=0.55)
```





By looking at the first model, we can tell the both “duration” and “amount” have very small p-values and thus, are significant to a 99,9% confidence level. The three of them do not seem to be linear. We can see very well in the plots that the dots are creating two very differentiate groups. Nevertheless, our three continuous variable model is not being able to capture this effect. We obviously have to check interactions between some continuous variable and some categorical. To do so, we will study the relevance of our categorical variables in a model, and asses each of their relevance to then study possible interactions. This way, our model starts becoming more complex, but also much more accurate.

Second, we run a model with all the variables of the dataset. In the results, we see that variable “age” becomes an irrelevant variable (even when using psplines and hiperparameter tuning for “K”), and subseive analysis confirms that we can take it out from our models. After losing this variable and using hiperparameter tuning in the other two non linear predictors, we check the categorical variables and we got rid of some more variables that are not useful to explain the credit response. We use only the p-values in order to have a broad or provisional understanding of the variables we for sure want to have in the model. This does not mean (by ane means) that we will forget about the other variables, as we will check their inclusion later on the assignment.

```
fit_gam2<- gam(response+education+chk_acct+history+new_car+used_car+furniture+radio.tv+
retraining+sav_acct+employment+install_rate+male_div+male_single+male_mar_or_wid+
co.applicant+guarantor+present_resident+real_estate+prop_unkn_none+other_install+
rent+own_res+job+num_credits+job+num_dependents+telephone+foreign+s(duration,k=40,bs="ps")+
s(amount,k=40,bs="ps"),family=binomial,select=TRUE,data=creditTrain)
summary(fit_gam2)
#>
#> Family: binomial
#> Link function: logit
#>
#> Formula:
#> response ~ education + chk_acct + history + new_car + used_car +
#>   furniture + radio.tv + retraining + sav_acct + employment +
#>   install_rate + male_div + male_single + male_mar_or_wid +
#>   co.applicant + guarantor + present_resident + real_estate +
#>   prop_unkn_none + other_install + rent + own_res + job + num_credits +
#>   job + num_dependents + telephone + foreign + s(duration,
#>   k = 40, bs = "ps") + s(amount, k = 40, bs = "ps")
#>
#> Parametric coefficients:
#>              Estimate Std. Error z value Pr(>|z|)
#> (Intercept)   1.64714    1.19154   1.382 0.166862
#> education1    -0.68195    0.60626  -1.125 0.260656
#> chk_acct1     0.28471    0.25461   1.118 0.263474
#> chk_acct2     0.47130    0.43305   1.088 0.276452
#> chk_acct3     1.74082    0.27663   6.293 3.12e-10 ***
```

```

#> history1      -0.33963    0.66826   -0.509  0.611080
#> history2      0.76789    0.53731    1.429  0.152965
#> history3      1.28754    0.57301    2.247  0.024642 *
#> history4      1.62655    0.53131    3.061  0.002203 **
#> new_car1      -0.47686    0.49122   -0.971  0.331671
#> used_car1      1.43811    0.66237    2.171  0.029919 *
#> furniture1     0.05149    0.52367    0.098  0.921670
#> radio.tv1     0.47016    0.50031    0.940  0.347359
#> retraining1    0.33692    0.56871    0.592  0.553565
#> sav_acct1      0.47283    0.34039    1.389  0.164805
#> sav_acct2      0.99946    0.52944    1.888  0.059057 .
#> sav_acct3      0.82108    0.56378    1.456  0.145291
#> sav_acct4      1.16871    0.32715    3.572  0.000354 ***
#> employment1    0.57958    0.54624    1.061  0.288679
#> employment2    0.68805    0.52990    1.298  0.194132
#> employment3    1.42912    0.56740    2.519  0.011779 *
#> employment4    0.83442    0.52146    1.600  0.109561
#> install_rate2  -0.40445    0.38168   -1.060  0.289292
#> install_rate3  -0.32736    0.42797   -0.765  0.444324
#> install_rate4  -0.78248    0.37256   -2.100  0.035706 *
#> male_div1      -0.49566    0.43873   -1.130  0.258575
#> male_single1   0.45562    0.25363    1.796  0.072429 .
#> male_mar_or_wid1 0.15989    0.37560    0.426  0.670332
#> co.applicant1  -0.66958    0.47702   -1.404  0.160417
#> guarantor1     0.90139    0.48606    1.854  0.063669 .
#> present_resident2 -1.02774    0.36757   -2.796  0.005174 **
#> present_resident3 -0.57658    0.40248   -1.433  0.151987
#> present_resident4 -0.56338    0.35882   -1.570  0.116396
#> real_estate1    0.13709    0.25104    0.546  0.585019
#> prop_unkn_none1 -0.89560    0.48964   -1.829  0.067384 .
#> other_install1 -0.58277    0.25477   -2.287  0.022169 *
#> rent1          -1.35945    0.58733   -2.315  0.020634 *
#> own_res1       -0.69371    0.56218   -1.234  0.217216
#> job1           -1.56745    0.88462   -1.772  0.076413 .
#> job2           -1.65623    0.86530   -1.914  0.055612 .
#> job3           -1.55480    0.87091   -1.785  0.074220 .
#> num_credits2    -0.29339    0.28760   -1.020  0.307671
#> num_credits3    -0.88634    0.65877   -1.345  0.178484
#> num_credits4    -0.28006    1.36178   -0.206  0.837058
#> num_dependents2 -0.29929    0.29617   -1.011  0.312230
#> telephone1     0.40395    0.23903    1.690  0.091036 .
#> foreign1       1.12861    0.70839    1.593  0.111117
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>      edf Ref.df Chi.sq p-value
#> s(duration)  1.172   29  20.12 5.6e-06 ***
#> s(amount)   26.280   35   40.78  0.0286 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.313   Deviance explained = 32.3%
#> UBRE = 0.012836   Scale est. = 1          n = 800
plot(fit_gam2,shade=TRUE,seWithMean=TRUE,pch=19,l,cex=0.55)

```

We took out the following variables to start trying models: *education*, *used_car*, *furniture*, *radio.tv*, *retraining*, *employment*, *male_div*, *male_mar_or_wid*, *job*, *num_credits*, *num_dependents*, *telephone* and *foreign*.

Now, we can see that there are some variables that contain some factors that are of special interest in our analysis, while there are others that don't seem to give us a lot of information. In these cases, we produced dummy variables, transforming categorical (ordinal) variables into binary ones for each of the levels. For example: the variable "history" accounts for the credit history ("0" for not having taken any credits before, "1" for having paid all credit duly at the bank, "2" for existing credits paid duly till now, "3" for having had a delay in the payback in the past and "4" account for a critical account). There seems to be no significance for the level 1 of the categorical variable, but it is very important to know if the account is critical or not (level 4). In order to capture this effect and make right use of the provided information, we produce a binary variable for each level (for example, a variable will take value "0" if it is not a critical account and "1" if it is a critical account), and so on. Then we test for significance.

After doing this, we try and fit a model only with the levels that looked significant. Once this is done, we start trying interactions, first between continuous variables. When doing this, we see that the interaction between amount and duration is very significant, but we adjust the k parameter to (10,7) because taking a higher k we only slow down the process (increase the computational cost) but our model does not fit better. We take this model as a benchmark. We get with it 24.6% explained deviance, and Akaike value equal to 661.49. When predicting, our Area under the Curve (in the ROC curve) is 0.8104762 and when building our own confusion matrix, we see that 77% of the predictions are right, while 23% are wrong classifications. Please note, that we find in each case, the optimal threshold that optimizes our confusion matrix. As our anove function tells us that an extra interaction is significant (comparing a reduced and full model) then we keep the interaction.

So, from now on, we do this analysis every time: comparing every criteria, and only if they coincide we keep the new term or do not do it. In case the criteria does not tell us a unique answer (for example, with an extra

interaction the prediction gets more accuracy but the AIC gets bigger), we discuss and take into account the meaning of the variables we are treating. And then decide as a team if we keep it, or not.

In order to improve this model (always optimizing the parameters), we start adding interactions between our most significant categorical variables and our continuous variables. In other words, we start checking how our model behaves when we add an interaction between “amount” and “prop_unkn_noun” (meaning that the applicant owns no property). And we do this with all the combinations, until we find the one that best fits our data. Then, we also try adding interactions between categorical variables (first the ones that seem to be very significant to explain our estimation) and then the ones that have a higher p-value.

Once we compared each model doing that, we give another step and start trying with other interactions between categorical variables. To do this, we used the interactions that were more significant in the logistic regression analysis. In other words, we took the interactions that looked significant or interesting in the linear model, and brought them to this gam model to check if they were relevant here too.

At some point, there are some variables we are not including that call our attention. For example, the test told us that the variable “employment” is not significant in any model. As we consider that it should be relevant the current employability situation of the credit consumer, we try and include it as a dummy variable: employed or unemployed. Even when doing this, this variable seems not to be relevant in our estimation.

When we conclude doing all this process, we finally get to two models with gam. One is fit_gam22 and another one called fit_gam27. Let’s discuss the different criteria to chose between one of them. The difference between these two models, is an interaction (that fit_gam27 has and the other one does not have) between “amount” and “history”.

```
fit_gam22= gam(response~new_car+used_car+install_rate+foreign+telephone+radio.tv+other_install:history_4+
  guarantor:history_4+ rent+other_install+prop_unkn_none+present_resident_2+guarantor+
  co.applicant+male_single+sav_acct_1+sav_acct_3+sav_acct_4+used_car+history_4+chk_acct_2+
  chk_acct_3+prop_unkn_none:chk_acct_3+prop_unkn_none:history_4+education:own_res+
  co.applicant:chk_acct+ s(duration,k=40,bs="ps",by=rent)+s(amount,k=40,bs="ps",by=used_car)+
  s(amount,k=40,bs="ps",by=telephone) +s(amount,k=40,bs="ps",by=prop_unkn_none)+
  te(amount,age,k=c(10,10),bs=c("ps","ps"),m=2)+te(amount,duration,k=c(10,7),bs=c("ps","ps"),m=2),
  family=binomial,select=TRUE,data=creditTrain)
summary(fit_gam22)
plot(fit_gam22,shade=TRUE,seWithMean=TRUE,pch=19,1,cex=0.55)

fit_gam27= gam(response~new_car+used_car+install_rate+foreign+telephone+radio.tv+other_install:history_4+
  guarantor:history_4+ rent+other_install+prop_unkn_none+present_resident_2+guarantor+
  co.applicant+male_single+sav_acct_1+sav_acct_3+sav_acct_4+used_car+history_4+chk_acct_2+
  chk_acct_3+prop_unkn_none:chk_acct_3+prop_unkn_none:history_4+education:own_res+
  s(amount,k=40,bs="ps",by=history)+co.applicant:chk_acct+s(duration,k=40,bs="ps",by=rent)+
  s(amount,k=40,bs="ps",by=used_car)+s(amount,k=40,bs="ps",by=telephone)+
  s(amount,k=40,bs="ps",by=prop_unkn_none)+te(amount,age,k=c(10,10),bs=c("ps","ps"),m=2)+
  te(amount,duration,k=c(10,7),bs=c("ps","ps"),m=2),family=binomial,select=TRUE,data=creditTrain)
summary(fit_gam27)
plot(fit_gam27,shade=TRUE,seWithMean=TRUE,pch=19,1,cex=0.55)
```

First, we check with the anova function, and we see that the interaction present in fit_gam27 is significant at a confidence level of 99,9%.

```
#> Analysis of Deviance Table
#>
#> Model 1: response ~ new_car + used_car + install_rate + foreign + telephone +
#>   radio.tv + other_install:history_4 + guarantor:history_4 +
#>   rent + other_install + prop_unkn_none + present_resident_2 +
#>   guarantor + co.applicant + male_single + sav_acct_1 + sav_acct_3 +
#>   sav_acct_4 + used_car + history_4 + chk_acct_2 + chk_acct_3 +
#>   prop_unkn_none:chk_acct_3 + prop_unkn_none:history_4 + education:own_res +
#>   co.applicant:chk_acct + s(duration, k = 40, bs = "ps", by = rent) +
#>   s(amount, k = 40, bs = "ps", by = used_car) + s(amount, k = 40,
#>   bs = "ps", by = telephone) + s(amount, k = 40, bs = "ps",
#>   by = prop_unkn_none) + te(amount, age, k = c(10, 10), bs = c("ps",
#>   "ps"), m = 2) + te(amount, duration, k = c(10, 7), bs = c("ps",
#>   "ps"), m = 2)
#> Model 2: response ~ new_car + used_car + install_rate + foreign + telephone +
#>   radio.tv + other_install:history_4 + guarantor:history_4 +
#>   rent + other_install + prop_unkn_none + present_resident_2 +
#>   guarantor + co.applicant + male_single + sav_acct_1 + sav_acct_3 +
#>   sav_acct_4 + used_car + history_4 + chk_acct_2 + chk_acct_3 +
#>   prop_unkn_none:chk_acct_3 + prop_unkn_none:history_4 + education:own_res +
#>   s(amount, k = 40, bs = "ps", by = history) + co.applicant:chk_acct +
#>   s(duration, k = 40, bs = "ps", by = rent) + s(amount, k = 40,
#>   bs = "ps", by = used_car) + s(amount, k = 40, bs = "ps",
#>   by = telephone) + s(amount, k = 40, bs = "ps", by = prop_unkn_none) +
#>   te(amount, age, k = c(10, 10), bs = c("ps", "ps"), m = 2) +
#>   te(amount, duration, k = c(10, 7), bs = c("ps", "ps"), m = 2)
#>   Resid. Df Resid. Dev      Df Deviance Pr(>Chi)
#> 1      693.44      571.30
#> 2      664.26      491.51 29.188    79.79 1.376e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Second, let’s check both Akaike values.

```
#> [1] 765.4875
#> [1] 742.5635
```

As we can see our `fit_gam27` has a lower AIC value, which makes it more attractive.

Now, let's check our ROC curve and the AUC. This is to determine the accuracy of predictions. We will do this, both in our training and testing set.

First, let's check the ROC for our `fit_gam22` (first training set and then in the testing set).

As we can see, our `fit_gam22` has an AUC of 0.91 when predicting on our training set, and then an AUC of 0.804 when predicting in our testing set. Let's now, check these measures for the `fit_gam27`.

```
#>      [,1]
#> 0 vs. 1 0.9280729
#>      [,1]
#> 0 vs. 1 0.8281548
```

As we can see, the AUC for `fit_gam27` when predicting over the training set is equal to 0.923, and a more realistic AUC of 0.83 when predicting over the testing set. Between both models, our `fit_gam27` has more accuracy power than `fit_gam22`.

Let's now build four confusion matrixes for both training and testing set, and `fit_gam22` and `fit_gam27`. With this, we can see the distribution of our errors, not only the error rate.

First, for `fit_gam22`:

```
#>
#>      0  1
#> FALSE 153 40
#> TRUE   87 520
#> [1] 84.125
#> [1] 15.875
#> [1] 0.9285714
#> [1] 0.6375
#>
#>      0  1
#> FALSE 34 14
#> TRUE  26 126
#> [1] 80
#> [1] 20
#> [1] 0.9
#> [1] 0.5666667
```

Now, for our `fit_gam27`:

```
#>
#>      0  1
#> FALSE 177 38
#> TRUE   63 522
#> [1] 87.375
#> [1] 12.625
#> [1] 0.9321429
#> [1] 0.7375
#>
#>      0  1
#> FALSE 36 14
#> TRUE  24 126
#> [1] 81
#> [1] 19
#> [1] 0.9
#> [1] 0.6
```

As we can see, when predicting over the training set, our `fit_gam27` predicts right almost 86.5% of times (either good or bad credit score). In other words, it predicts making mistakes almost 13.5% of the costumers. In this model, our sensitivity is 0.926 and specificity is 0.717. While, our `fit_gam22` predicts well almost 85.5% of the cases (that is makes mistakes in predictions almost 14.5% of the cases). Here, our sensitivity is 0.92 and specificity is 0.7.

When looking at the testing set, we get that our `fit_gam22` predicts right on 83% of times (while makes mistakes on 17% of the costumers). Its sensitivity is 0.92 and specificity 0.6. In our `fit_gam27` we predict 81% of the cases right (19% making mistakes). Our sensitivity is 0.89 and specificity is at 0.62.

All in all, taking into account every consideration, we will use our `fit_gam27` to do predict if a future costumer will be good or bad costumer.

Variable Importance

```
#>      new_car      used_car      install_rate
#> 0.000000000 0.000000000 -0.869685455
#>      foreign      telephone      radio.tv
```

```

#>      0.000000000      -0.058026206      0.645424268
#>      rent      other_install      prop_unkn_none
#>      0.000000000      0.000000000      0.000000000
#>      present_resident_2      guarantor      co.applicant
#>      0.000000000      0.000000000      0.000000000
#>      male_single      sav_acct_1      sav_acct_3
#>      0.448490129      0.000000000      0.000000000
#>      sav_acct_4      history_4      chk_acct_2
#>      1.500239160      1.413031593      0.000000000
#>      chk_acct_3      other_install:history_4      guarantor:history_4
#>      0.000000000      0.000000000      0.000000000
#>      prop_unkn_none:chk_acct_3      prop_unkn_none:history_4      education:own_res
#>      0.000000000      0.000000000      1.028640586
#>      co.applicant:chk_acct      s(amount):history0      s(amount):history1
#>      0.000000000      0.000000000      0.000000000
#>      s(amount):history2      s(amount):history3      s(amount):history4
#>      0.000000000      0.000000000      0.306138128
#>      s(duration):rent0      s(duration):rent1      s(amount):used_car0
#>      1.039874201      0.000000000      -0.005261844
#>      s(amount):used_car1      s(amount):telephone0      s(amount):telephone1
#>      0.000000000      0.000000000      1.173471547
#>      s(amount):prop_unkn_none0      s(amount):prop_unkn_none1      te(amount,age)
#>      -0.211918974      0.000000000      1.508488928
#>      te(amount,duration)
#>      -0.729349159

```

In order to asses what are the most relevant variables in our model, we will predict 20 points from our testing set, and check the influence af all the terms in each prediction. This way, we can see to what extent each variable contributes to predict a new costumer (either good or bad). We do this in terms of probabilities, so if a variable adds 0.5 of probability to a prediction, it means it does not really add any useful information to that point of prediction. In general, we can see that there are some variables that always (or almost always) take different values to 0.5 in the predictions. This way, we can say that the most important variables in our model to predict whether a costumer will be good or bad, will be: inst_rate (accounts for the % of disposable income), male_single, sav_acct (accounts for the average balance in savings account), duration (duration of the credit in months) and amount (amount of the credit).

Others, that are important, but not as important as the ones previously mentiones are: radio.tv (if the credit is destined to buying a tv or a radio), telephone (the purpose of the credit is to buy a phone), new_car (if the costumer wants to use the credit for a new car), present_residence (amount of years in the present residence), chk_acct (accounts for the checking account status, and _3 is for the one who does not have a checking account).

To sum up, in general a good costumer would be someone who asks a credit for to buy a radio or a tv, or a phone, who has a guarantor, who is single with a high avarage balance in savings account, single and male. For a bad costumer, we could be facing someone who is asking a credit to buy a new car, who rents (residence), who has a high installment rate as % of disposable income, who owns no property, who has not been a resident for more than 2 years and with no achecking account.

Final model

We have chosen a mix of GAM and GLM for our model. We have chosen a final model where we mix both GAM and GLM. We choose based on the following table:

Table 5: best models

| gam_result | glm_result | final_choice |
|------------|------------|--------------|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

And we get the following confusion matrix for it (in sample prediction):

```

#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0 137  18

```

```

#>      1 103 542
#>
#>      Accuracy : 0.8488
#>      95% CI : (0.822, 0.8729)
#>      No Information Rate : 0.7
#>      P-Value [Acc > NIR] : < 2.2e-16
#>
#>      Kappa : 0.5993
#>
#>      Mcnemar's Test P-Value : 2.234e-14
#>
#>      Sensitivity : 0.5708
#>      Specificity : 0.9679
#>      Pos Pred Value : 0.8839
#>      Neg Pred Value : 0.8403
#>      Prevalence : 0.3000
#>      Detection Rate : 0.1713
#>      Detection Prevalence : 0.1938
#>      Balanced Accuracy : 0.7693
#>
#>      'Positive' Class : 0
#>

```

This returns an accuracy of 84.88%, sensitivity of 57.08% and specificity of 96.79%. This voting composition between both models tends to have a significantly harder time classifying clients without good credit, however, clients with good credit rating are very accurately classified.

And we get the following confusion matrix for it (out of sample prediction):

```

#> Confusion Matrix and Statistics
#>
#>      Reference
#> Prediction  0   1
#>      0    32   3
#>      1    28 137
#>
#>      Accuracy : 0.845
#>      95% CI : (0.7873, 0.8922)
#>      No Information Rate : 0.7
#>      P-Value [Acc > NIR] : 1.572e-06
#>
#>      Kappa : 0.5811
#>
#>      Mcnemar's Test P-Value : 1.629e-05
#>
#>      Sensitivity : 0.5333
#>      Specificity : 0.9786
#>      Pos Pred Value : 0.9143
#>      Neg Pred Value : 0.8303
#>      Prevalence : 0.3000
#>      Detection Rate : 0.1600
#>      Detection Prevalence : 0.1750
#>      Balanced Accuracy : 0.7560
#>
#>      'Positive' Class : 0
#>

```

The model has a final accuracy of 84.5% with a sensitivity of 53.33% and specificity of 97.86%. The model strongly predicts clients with a good credit rating and has a hard time predicting clients without a good credit rating.