Workshop 2 - Selecting Explanatory Variables

JPS

22/12/2021

Introduction

Let's practice the content of today's lecture with a series of exercises using the Crown Court Sentencing Survey. First, we are going to see how useful stepwise selection can be at building models when the research goal is to *predict*. We will then contrast such unsupervised model selection process with the more careful approach that we should follow when modelling to *explain*. Lastly, we will see how we can use the VIF (Variance Inflation Factor) to detect problems of multicollinearity. Hence, after taking this workshop with two important new techniques in your expanding toolbox, stepwise regression and VIF.

Each of the exercises presented is linked to relevant research questions in the field of sentencing. Two of them replicate and take forward two articles published in the British Journal of Criminology. Specifically, the second exercise suggests an important piece of empirical research that could be easily explored with the techniques that you will acquire today. It deals with an ongoing debate in the sentencing literature, which rather surprisingly, remains unexplored empirically.

Exercise 1: Imagine we are a consultant for a criminal law firm that wants to use data analytics to inform what type of clients they should take in. We suggest focusing on cases where the most severe penalty (immediate custody) can be avoided. To do so we decide to build a predictive model based on the case characteristics, but also on the characteristics of the court where the sentence is going to be imposed.

Exercise 2: According to the 2009 Criminal Justice Act, judges and magistrates must follow sentencing guidelines in deciding the appropriate sentence. Critics have argued that forcing sentencers to comply with the guidelines has made sentencing in England and Wales more punitive. Specifically Hutton (2013) and Reitz (2013) argue that the guidelines have relegated the relevance of mitigating factors. We will explore this claim by testing whether the effect of mitigating factors on the sentence outcome is weaker than the effect derived from the use of aggravating factors.

Exercise 3: Maslen and Roberts (2015) hypothesised that sentence reductions cannot be offered uniformly to all offenders who show remorse (e.g. 10% shorter custodial sentences to all offenders who show remorse), instead the mitigating factor of remorse will be applied differently in different circumstances. Lightowlers and Pina-Sanchez (2017) showed that the aggravating factor 'alcohol intoxication' interacts with the mitigating factor 'good character', i.e. 'alcohol intoxication' did not have an aggravating effect with the offender was deemed to be of 'good character'. Belton (2018) found multiple additional interactions between mitigating factors. Pointing at the need to control for such effects in models seeking to reflect the sentencing decision process accurately. Yet, none of these studies consider the problem of multicollinearity, therefore their findings are questionable. We are going to assess this problem by exploring whether mitigating factors are applied differently when in combination with the offender expressing genuine remorse, while checking potential problems of multicollinearity.

Accessing the Data

We are going to use a simplified version of the Crown Court Sentencing Survey (CCSS) including the court where each of the sentences was imposed. We are going to match that with data from the Census capturing the demographic characteristics of the Lower Super Output Area where the court is located, and with a range of other court characteristics obtained from various official reports. This combined dataset was used

in: Pina-Sanchez, J. & Grech, D. (2017) Location and Sentencing: To What Extent Do Contextual Factors Explain between Court Disparities? *British Journal of Criminology*.

Let's now access each of those three datasets and proceed to merge them ourselves:

```
ccss = read.csv("CCSS2011_trimmed.csv", stringsAsFactors = TRUE)
#We use 'stringsAsFactors = TRUE' so the nominal variables are read as factors rather than as character
##Also, make sure you provide the direction to the folder where you saved the dataset.
```

This is a clean(er) version of the 2011 CCSS. I dropped cases where the sentence outcome was missing, dropped variables that are not relevant and others measured inconsistently (e.g. previous convictions was severely affected by problems of item-nonresponse), and created dummy variables for the principal offence so they are ready to be used in a model. I have also tried to provide meaningful variable / value labels, if something is not clear you can always check the metadata file 'CCSS_metadata.xls', available on Minerva.

Checking the main features of this dataset at the top-right menu in Rstudio you can see that it is composed of 49 variables, which is probably bigger than what you are used to. To have a quick look at the content of the dataset we can use...

```
names(ccss)
```

```
##  [1] "Sentencing.court" "Step1HigherCulpA" "Step1HigherCulpC" "Step1HigherCulpD"
##  [5] "Step1HigherCulpF" "Step1HigherCulpG" "Step1HigherCulpI" "Step1HigherCulpJ"
##  [9] "Step1HigherHarmA" "Step1HigherHarmB" "Step1HigherHarmC" "Step1LowerCulpA"
## [13] "Step1LowerCulpB"  "Step1LowerCulpC"  "Step1LowerCulpD"  "Step1LowerCulpE"
## [17] "Step1LowerHarmA"  "Aggravating.3"    "Aggravating.5"    "Aggravating.9"
## [21] "Aggravating.19"   "Aggravating.25"   "Aggravating.26"   "Aggravating.28"
## [25] "Aggravating.37"   "Aggravating.44"   "Aggravating.45"   "Aggravating.54"
## [29] "Aggravating.55"   "Aggravating.70"   "Aggravating.72"   "Mitigating.3"
## [33] "Mitigating.6"     "Mitigating.16"    "Mitigating.20"    "Mitigating.22"
## [37] "Mitigating.24"    "Mitigating.25"    "Mitigating.29"    "Mitigating.33"
## [41] "Mitigating.36"    "Mitigating.39"    "pleafirst"        "disorder"
## [45] "affray"           "GBH"              "intent"           "common"
## [49] "custody"
```

Ok, so we have a variable capturing the court where the sentence was imposed (this has been anonymised so it is just an id rather than the original name), 16 harm and culpability factors, 14 aggravating factors, 11 mitigating factors, 'pleafirst' indicating whether a guilty plea was entered at first opportunity, 5 dummy variables capturing the specific case of assault being sentenced (with assault with bodily harm as the reference category), and 'custody' indicating whether the offender received an immediate custodial sentence. With the exception of 'Sentencing.court' all of the variables are binary. To assess their relative prevalence we can simply use...

```
summary(ccss)
```

Does it all look ok? I would say so. Our outcome variable seems evenly distributed, no missing cases except 291 cases in 'Sentencing.court' and big enough frequencies for the case characteristics.

Let's now import the census data and the court characteristics datasets.

```
census = read.csv("Census.csv")
summary(census)
```

We can see again a court id (we will use this to facilitate the data merge below) and 15 variables capturing the proportion of households found in the same LSOA where the court is located, featuring different sociodemographic characteristics, such as the percentage of Muslims living in the area, whites, teenagers, unemployed, etc.

```
courtchars = read.csv("CourtChars.csv")
summary(courtchars)
```

Here we find the same court id, six binary variables indicating whether certain facilities are present in the court, such as interview rooms, wifi (important to facilitate judges to use the guidelines in their tablets), etc., and a continuous variable 'Volume' capturing the average number of cases processed in each court per month.

Now, notice how most variables are either binary, or proportions (confined to the 0-1 range), but then we have 'Volume', which takes a 217-4226 range. This could create different types problems (e.g. lack of identifiability) and make the computational effort more demanding, especially in complex models like the ones we are going to estimate (with >60 explanatory variables), and in computationally intensive models like stepwise regression. We can eliminate those problems by transforming 'Volume' so it takes a narrower range, e.g. we can simply divide it by 1000. There are no downsides to this and it is something you should always consider when using variables with vast ranges. All we have to remember is, when interpreting the regression coefficient for this variable, to bring it back to its original scale.

```
courtchars$Vol_thous = courtchars$Volume / 1000
summary(courtchars$Vol_thous)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2170  0.9875  1.3510  1.7407  2.3260  4.2260
```

```
courtchars$Volume = NULL
```

Good, so we have imported all the three datasets and checked they are in good shape. Let's merge them now. When putting together different datasets it is important to consider what we want to do. Is it to add new cases or new variables? In our case it will be the latter. Notice however that the three datasets do not have the same number of observations. This is what is known as *one to many* merge, as opposed to a *one to one*. In this case we will need an id variable that can be used to tell R where to match each case. Fortunately for us we can do that using the courts id. If you see yourself in a different *merging* scenario I recommend that you consult what procedure to follow here Quick-R.

Before running the *merge* command we need to make sure that the variables to be used to identify unique ids are given the same name in each dataset.

```
names(ccss)[names(ccss) == 'Sentencing.court'] = 'court'
names(census)[names(census) == 'court_name'] = 'court'
names(courtchars)[names(courtchars) == 'Court'] = 'court'
```

We can now proceed with the merging, which we will undertake in two stages since the command *merge* only allow us to merge two datasets at a time.

```
merged1 = merge(ccss, courtchars, by = 'court')
merged2 = merge(merged1, census, by = 'court')
```

Notice that we have lost 291 observations from the ccss dataset. This is because the three datasets did not have exactly the same courts (courts have been closing down systematically over the last decade so depending when the data was captured the list of Crown Court locations varies). Given that the lost cases represent <10% original sample we can simply report this as a caveat and move on with the analysis. In Workshop 5 we will see how to deal with missing data more proficiently.

Once the merging process is successfully undertaken we can drop the variable 'court' from the final dataset since we are not going to use this in our models, instead we will use the variables describing their characteristics. If we were to include both court characteristics and dummy variables for each court location, we would have perfect collinearity. This means that with one of the explanatory variables you can predict perfectly the values of another explanatory variable. Hence, the model is unidentifiable.

```
merged2$court = NULL
```

Exercise 1. Modelling to Predict

Our research goal, as explained in the instructions of Exercise 1, is to predict custodial sentences as accurately as possible. To do so we will try to build the best possible predictive model. But how do we do that? First, we want a logistic model since the outcome to be predicted is binary (custody or not). Then we need to consider the next challenge, how to select the set of explanatory variables. We have seen in the lecture that throwing all of the variables we have in a model is a poor strategy since some of them might only be adding noise. Given the size of our dataset we cannot possibly explore all the different combinations of explanatory variables (70 potential explanatory variables. If we were to choose only 20 of them we would have to compare 115,631,859,759,041,340 models). Instead we will use stepwise regression.

One more thing, to be able to assess the predictive accuracy of our models we should first split the dataset so we have a training sample, which will be used to build the model, and a test sample, to assess how well our model performs on a different dataset. We can undertake such a dataset split using the command *createDataPartition* (from the machine learning package *caret*), and the *%>%* operator.

```
set.seed(10)
library(caret) #this is to use createDataPartition
library(dplyr) #this is to be able to use %>%
training.sample = merged2$custody %>% createDataPartition(p = 0.8, list = FALSE)
train.data = merged2[training.sample, ]
test.data = merged2[-training.sample, ]
```

First we have used *set.seed*. This is because we are going to ask R to draw pseudo-random numbers. By providing a figure to *set.seed*, we will ensure that we get the same random numbers every time we replicate our code. Next, we have used *createDataPartition* to create a vector representing the 80% of cases that have been picked from our original dataset ('merged2') to conform the training.sample. The choice of 80% is rather arbitrary, we normally see researchers using 50%, however since our sample size is not massive and we have a good number of explanatory variables I did not want to risk ending with a sample too small to undertake the model selection effectively. The *%>%* operator facilitates using loops in R. Specifically, our command can be translated as: for every case in a given variable in the dataset ('custody' in our case), give it an 80% chance of being picked up, and report those that were selected. At this point we can use the vector we have created ('training.sample') to create our training and test samples.

We are now going to use the 'training.sample', and *stepAIC* from the package *MASS*, to look for the best predictive model. This command will employ a stepwise selection algorithm (based on a combination of backward and forward selection processes) to find the set of explanatory variables that minimises the model's AIC. To illustrate the benefits of using stepwise selection to build a predictive model we will use the full model as a benchmark. This is the model where all the explanatory variables are included, which can be specified in R easily using '.' on the right hand side of the equation.

```
full.model = glm(custody ~., data=train.data, family=binomial)
summary(full.model)
```

We can see there are several non-significant variables in the model. Most of the court and area specific characteristics are not significant, but neither are many of the harm, culpability, aggravating and mitigating factors, which by definition are supposed to be *substantively* significant. Let's see now whether we can improve that model by trimming it down using *stepAIC*. This, however, can take a while (about 4 minutes in my university laptop).

```
library(MASS)
step.model = full.model %>% stepAIC
summary(step.model)
```

You can see how R is running multiple iterations with different sets of explanatory variables following backward and forward selections, in each of them a list of AIC associated with the addition/removal of variables is provided. You can also see how the models' AIC go down following each iteration, until a model is achieved for which neither adding nor eliminating more variables improve the AIC.

Ok, let's now assess which model seems to perform better, our initial full model, or this new model obtained through stepwise selection. To compare model performance we will use different metrics, the AIC, the models' size (number of coefficients included, considering both the total number of coefficients included and those found significant), and their predictive accuracy.

```
full.model$aic
```

```
## [1] 2987.901
```

```
step.model$aic
```

```
## [1] 2955.585
```

Hands down the AIC is considerably lower in the 'step.model'. Also, as to be expected, the 'step.model' uses far less explanatory variables.

```
length(coef(full.model))
```

```
## [1] 70
```

```
length(coef(step.model))
```

```
## [1] 49
```

And it even finds more significant variables!

```
pvalues_full = coef(summary(full.model))[,4]
significant_full = ifelse(pvalues_full>0.05, 0, 1)
sum(significant_full)
```

```
## [1] 35
```

```
pvalues_step = coef(summary(step.model))[,4]
significant_step = ifelse(pvalues_step>0.05, 0, 1)
sum(significant_step)
```

```
## [1] 39
```

While still managing to provide higher predictive accuracy! All of the above combined illustrates quite well some of the advantages of using stepwise procedures for exploratory and predictive purposes.

```
probs_full = full.model %>% predict(test.data, type = "response")
predicted.clas_full = ifelse(probs_full > 0.5, "immediate custody", "")
observed.clas_full = test.data$custody
mean(predicted.clas_full == observed.clas_full)
```

```
## [1] 0.7579909
```

```
probs_step = step.model %>% predict(test.data, type = "response")
predicted.clas_step = ifelse(probs_step > 0.5, "immediate custody", "")
observed.clas_step = test.data$custody
mean(predicted.clas_step == observed.clas_step)
```

```
## [1] 0.7591324
```

Exercise 2. Modelling to Explain

Let's now compare the above with the model selection process that we should follow when exploring a clearly identified deductive research question. In the introduction we suggested testing the hypothesis that aggravating factors are more relevant than mitigating factors in deciding the sentence outcome (Hutton 2013; and Reitz 2013).

To opt for a full model (throwing all available explanatory variables in the model) would not be an adequate strategy. It would result in an overfitted model that will likely lead to problems of multicollinearity, which then will bias the regression coefficients for the aggravating and mitigating factors and their measures of uncertainty. Let's check that using the 'full.model' that we have already estimated

```
library(regclass) #this is to use VIF
VIF(full.model)
```

The VIF exploration only shows a problem of multicollinearity ($VIF>5$) in some of the variables capturing court and area characteristics, but not in any of the sentencing factors. Most of those court and area characteristics were dropped following the stepwise selection process that we undertook. However, a stepwise procedure would not offer an adequate strategy either. The hypothesis to be tested is very clearly defined, it requires a comparison of the effect of aggravating and mitigating factors, but our previous 'step.model' only kept 9 out of the 14 aggravating factors available in the dataset, and 10 out of the 11 mitigating factors.

Since we want to test a specific hypothesis the choice of the variables to be included in the model should be theoretically driven. This way we will make sure that we retain all aggravating and mitigating factors, even if they are not significant. Notice how the fact that some of them are significant or not is already a relevant result in deciding whether to refute or corroborate the hypothesis that mitigating factors have been relegated in the sentencing guidelines. Let's explore this hypothesis in more detail.

We should select variables that, based on the - sentencing - theory, can be considered to have an effect on the - sentence - outcome. This is a subjective choice and should always involve careful judgement. For the particular case we are considering this choice is perhaps easier than for most other questions explored by social scientists, since the sentencing guidelines provide a list of factors deemed to be relevant in deciding the sentence outcome. So we can simply include all of them. To do so we will go back to use the 'ccss' data. We will first proceed to remove court location since, based on the guidelines, sentencing should be consistent across courts (we will see how to explore this hypothesis in Workshop 7 using multilevel models).

```
ccss$court = NULL
theo.model = glm(custody ~., data=ccss, family=binomial)
VIF(theo.model)
summary(theo.model)
```

We do not find problems of multicollinearity so we can proceed to interpret the model's regression coefficients. We want to compare the importance of aggravating and mitigating factors. Since they are all measured in the same scale (binary) we can proceed to add up the coefficients for all the aggravating factors found significant (if they are not significant we can think of them as not having an influence on the outcome, or at least not having an influence that we can detect) and compare that against the sum of all significant mitigating factors. In the code below, notice how we have identified each of the aggravating and mitigating factors included in our model.

```
#Overall effect associated to the 10 aggravating factors found significant
sum(coef(theo.model)[19], coef(theo.model)[20], coef(theo.model)[23], coef(theo.model)[25], coef(theo.m
```

```
## [1] 7.477075
```

```
#Overall effect associated to the 10 mitigating factors found significant
sum(coef(theo.model)[32], coef(theo.model)[33], coef(theo.model)[34], coef(theo.model)[35], coef(theo.m
```

```
## [1] -9.428598
```

Out of the 14 aggravating factors included in the model only 10 are significant, when they are all present they can push the sentence up by 7.48 log-odds; whereas 10 mitigating factors are found significant, even though there were 11 identified in the sentencing guidelines, when they are all present in a given case they can push the sentence down by -9.43 log-odds. In conclusion, it seems that Hutton (2013) and Reitz (2013) are wrong, mitigating factors are more consequential than aggravating factors, at least for the case of offences of assault processed in the Crown Court. Believe it or not, no single sentencing scholar has bothered to test this hypothesis empirically, and the myth of mitigating factors having a residual effect lives on.

Exercise 3. Multicollinearity

In this last exercise we are going to test whether different sentencing factors are applied differently when other offence characteristics are present - as suggested by multiple sentencing scholars. To do so we are going to use interaction effects. Specifically we are going to focus on assessing whether mitigating factors are applied differently when present in combination with the offender expressing genuine remorse. Interaction effects, however, represent the factorial combination of two explanatory variables already in the model. By definition they are prone to generate problems of multicollinearity. So we will proceed to explore the interaction effect of each mitigating factor with 'remorse' while controlling for multicollinearity.

Go ahead and give this a try. Can you identify robustly whether the effect of the different mitigating factors vary when 'remorse' is present? I recommend you that you do this exploring interaction effects in our last 'theo.model' one by one. Since we need to make sure the regression coefficients are robustly estimated, I would not provide a substantive interpretation of those interaction effects that are not significant or with *VIF>5*. Also, to avoid having to specify the whole model in R, rather than specifying interaction effects using the ':' operator (as you saw last last year), you could instead create a new variable that represents that interaction effect in your datasets, as shown below.

```
ccss$rem_miti3 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.3)-1)
#The above is the interaction effect between remorse (Mitigating.36) and addressing addiction (Mitigati
int1.model = glm(custody ~., data=ccss, family=binomial)
summary(int1.model)
VIF(int1.model)
ccss$rem_miti3 = NULL
```

Notice how I used *as.numeric()-1* to turn the variables from factors to (0,1) variables so we can multiply them to create the interaction effect, which, as we can see in the model output, is not significant. Its VIF is <5 but that is not too relevant since the interaction effect is not significant anyway. Go ahead and undertake a similar analysis. If you wanted to replicate this procedure for the second mitigating factor available in our dataset ('Mitigating.6', 'lack of maturity') you could do as follows:

```
ccss$rem_miti6 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.6)-1)
int2.model = glm(custody ~., data=ccss, family=binomial)
summary(int2.model)
VIF(int2.model)
ccss$rem_miti6 = NULL
```

As before, the interaction effect between 'remorse' and 'lack of maturity' is not affected by multicollinearity but nor is it statistically significant. Hence, we can conclude the effect of 'lack of maturity' is not influenced by the presence of 'remorse'.

Carry on testing interactions with the rest of mitigating factors. Can you identify whether the effect of the mitigating factors vary when 'remorse' is present? Is there ground to consider that sentencing factors included in the guidelines interact with each other as suggested by multiple sentencing scholars?

```
ccss$rem_miti16 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.16)-1)
int3.model = glm(custody ~., data=ccss, family=binomial)
summary(int3.model)
VIF(int3.model)
ccss$rem_miti16 = NULL
```

```
ccss$rem_miti20 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.20)-1)
int4.model = glm(custody ~., data=ccss, family=binomial)
summary(int4.model)
VIF(int4.model)
ccss$rem_miti20 = NULL
```

```
ccss$rem_miti22 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.22)-1)
int5.model = glm(custody ~., data=ccss, family=binomial)
summary(int5.model)
VIF(int5.model)
ccss$rem_miti22 = NULL
```

Mitigating.22 (lapse of time since the offence) is significant and its VIF<5. This is therefore an interaction worth reporting, providing some support to Maslen and Roberts (2015) view on the context-specific effect of remorse. The sign of the interaction term is positive, meaning that when both remorse and lapse of time are present, their independent effects are not simply added on top of each other, but somehow their individual mitigating effect becomes weaker than if those two mitigating factors were not present simultaneously.

```
ccss$rem_miti24 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.24)-1)
int6.model = glm(custody ~., data=ccss, family=binomial)
summary(int6.model)
VIF(int6.model)
ccss$rem_miti24 = NULL
```

```
ccss$rem_miti25 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.25)-1)
int7.model = glm(custody ~., data=ccss, family=binomial)
summary(int7.model)
VIF(int7.model)
ccss$rem_miti25 = NULL
```

```
ccss$rem_miti29 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.29)-1)
int8.model = glm(custody ~., data=ccss, family=binomial)
summary(int8.model)
VIF(int8.model)
ccss$rem_miti29 = NULL
```

```
ccss$rem_miti33 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.33)-1)
int9.model = glm(custody ~., data=ccss, family=binomial)
summary(int9.model)
VIF(int9.model)
ccss$rem_miti33 = NULL
```

```
ccss$rem_miti39 = (as.numeric(ccss$Mitigating.36)-1) * (as.numeric(ccss$Mitigating.39)-1)
int10.model = glm(custody ~., data=ccss, family=binomial)
summary(int10.model)
VIF(int10.model)
ccss$rem_miti39 = NULL
```

Only one of the ten interactions explored was significant. Clearly, there are many other interactions that we could explore between remorse and aggravating factors, or harm and culpability factors, however, based on our results I would be hesitant to support Maslen and Roberts (2015) view that remorse is a factor applied differently depending on the context. The evidence generated here tells us that the mitigating effect of remorse seems pretty stable, applied uniformly regardless the characteristic of the case.