# HW7

ASG

2023-04-03

## 1

The very first thing one should look into when selecting the model terms , is the domain knowledge of the data , the hypothesis problem and the dataset itself. It is very important to first weed out the terms based on the domain knowledge of the data and the hypothesis one is working on. Rest steps can follow after this necessary first step.

## 3.2

### a

```
set.seed(1) ; n <- 500 ; error <- rnorm(n,0,0.5)
x1 <- runif(n) ; x2 <- runif(n) ; x3 <- runif(n)
x4 <- runif(n) ; x5 <- runif(n) ; x6 <- runif(n)
x7 <- runif(n) ; x8 <- runif(n) ; x9 <- runif(n)
f4 <- function(x) return(x + dnorm(x,0.5,0.25))
f5 <- function(x) return(x + 0.5*dnorm(x,0.5,0.25))
f6 <- function(x) return(x + 0.1*dnorm(x,0.5,0.25))
y <- x1 + x2 + x3 + f4(x4) + f5(x5) + f6(x6) + error
```

### b

```
library(gam)
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.22-2
```

```
gamObj <- gam(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9)
stepFit <- step.Gam(gamObj,scope =list("x1" = ~ 1 + x1 + s(x1,df = 2),"x2" = ~ 1 + x2 + s(x2,df = 2),"x3
```

```
## Start:  y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9; AIC= 1125.171
## Step:1 y ~ x1 + x2 + x3 + s(x4, df = 2) + x5 + x6 + x7 + x8 + x9 ; AIC= 890.2083
## Step:2 y ~ x1 + x2 + x3 + s(x4, df = 2) + s(x5, df = 2) + x6 + x7 +      x8 + x9 ; AIC= 817.6723
## Step:3 y ~ x1 + x2 + x3 + s(x4, df = 2) + s(x5, df = 2) + s(x6, df = 2) +      x7 + x8 + x9 ; AIC= 8
## Step:4 y ~ x1 + x2 + x3 + s(x4, df = 2) + s(x5, df = 2) + s(x6, df = 2) +      x8 + x9 ; AIC= 812.772
## Step:5 y ~ x1 + x2 + x3 + s(x4, df = 2) + s(x5, df = 2) + s(x6, df = 2) +      x9 ; AIC= 811.1671
## Step:6 y ~ x1 + x2 + x3 + s(x4, df = 2) + s(x5, df = 2) + s(x6, df = 2) +      s(x9, df = 2) ; AIC= 8
```

### c

```
print(names(stepFit$"model")[-1])
```

```
## [1] "x1"             "x2"             "x3"             "s(x4, df = 2)"
## [5] "s(x5, df = 2)" "s(x6, df = 2)" "s(x9, df = 2)"
```

```
correct_fxn<-function(fnames){
  correct<-c(
  "x1" %in% fnames,
  "x2" %in% fnames,
  "x3" %in% fnames,
  "s(x4, df = 2)" %in% fnames,
  "s(x5, df = 2)" %in% fnames,
  "s(x6, df = 2)" %in% fnames,
  !("s7" %in% fnames | "s(x7, df = 2)" %in% fnames),
  !("s8" %in% fnames | "s(x8, df = 2)" %in% fnames),
  !("s9" %in% fnames | "s(x9, df = 2)" %in% fnames)
  )
  as.numeric(correct)
}
ans<-correct_fxn(names(stepFit$"model")[-1])
print(ans)
```

```
## [1] 1 1 1 1 1 1 1 1 0
```

### d

```
library(gam)
prop<-rep(0,9)
for (i in 1:100){
  set.seed(i) ; n <- 500 ; error <- rnorm(n,0,0.5)
  x1 <- runif(n) ; x2 <- runif(n) ; x3 <- runif(n)
  x4 <- runif(n) ; x5 <- runif(n) ; x6 <- runif(n)
  x7 <- runif(n) ; x8 <- runif(n) ; x9 <- runif(n)
  f4 <- function(x) return(x + dnorm(x,0.5,0.25))
  f5 <- function(x) return(x + 0.5*dnorm(x,0.5,0.25))
  f6 <- function(x) return(x + 0.1*dnorm(x,0.5,0.25))
  y <- x1 + x2 + x3 + f4(x4) + f5(x5) + f6(x6) + error
  gamObj <- gam(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9)
  stepFit <- step.Gam(gamObj,trace=FALSE,scope =list("x1" = ~ 1 + x1 + s(x1,df = 2),"x2" = ~ 1 + x2 + s
  vars<-correct_fxn(names(stepFit$"model")[-1])
  prop=prop+vars
}
print(prop/100)
```

```
## [1] 0.73 0.78 0.77 1.00 1.00 0.78 0.88 0.90 0.84
```

```
detach(package:gam)
```

### e

```
library(gam)
```

```
## Loaded gam 1.22-2
```

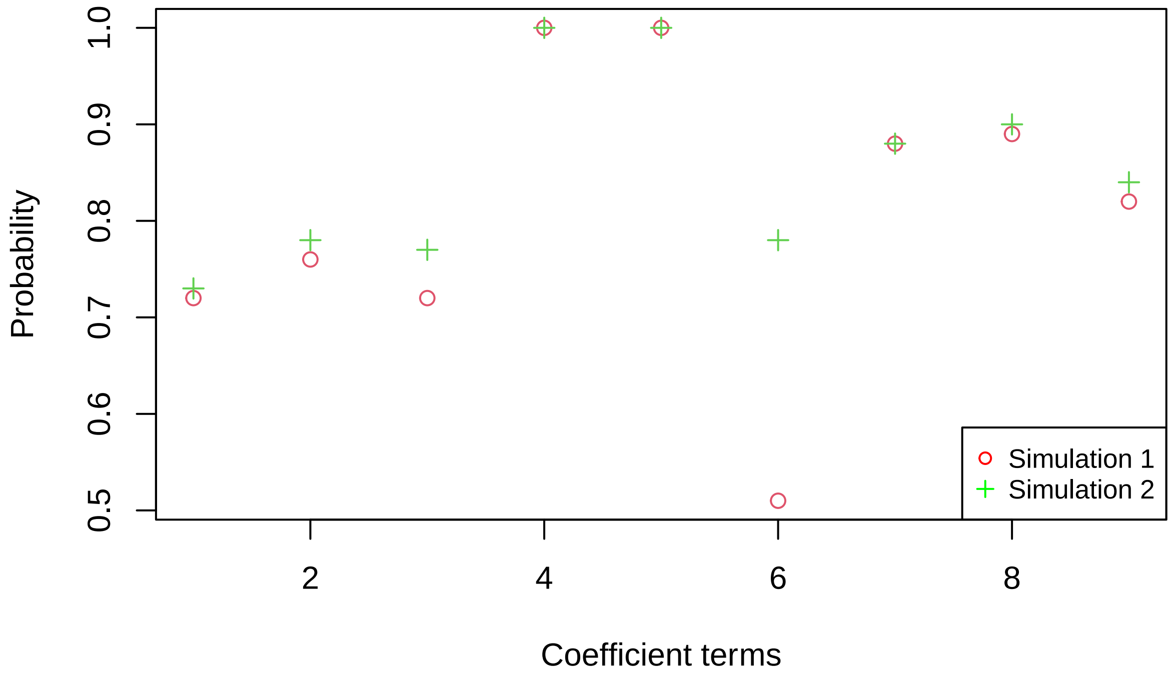
```
prop2<-rep(0,9)
for (i in 1:100){
  set.seed(i) ; n <- 500 ; error <- rnorm(n,0,1)
  x1 <- runif(n) ; x2 <- runif(n) ; x3 <- runif(n)
  x4 <- runif(n) ; x5 <- runif(n) ; x6 <- runif(n)
  x7 <- runif(n) ; x8 <- runif(n) ; x9 <- runif(n)
  f4 <- function(x) return(x + dnorm(x,0.5,0.25))
  f5 <- function(x) return(x + 0.5*dnorm(x,0.5,0.25))
  f6 <- function(x) return(x + 0.1*dnorm(x,0.5,0.25))
  y <- x1 + x2 + x3 + f4(x4) + f5(x5) + f6(x6) + error
  gamObj <- gam(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9)
  stepFit <- step.Gam(gamObj,trace=FALSE,scope =list("x1" = ~ 1 + x1 + s(x1,df = 2),"x2" = ~ 1 + x2 + s
  vars<-correct_fxn(names(stepFit$"model")[-1])
  prop2=prop2+vars
}
print(prop2/100)
```

```
## [1] 0.72 0.76 0.72 1.00 1.00 0.51 0.88 0.89 0.82
```

```
detach(package:gam)
```

### f

```
y2<-prop2/100
y1<-prop/100
x<-seq(1,9)
plot(y2,col=2,xlab="Coefficient terms",ylab="Probability")
points(y1,col=3,pch=3)
legend("bottomright", legend=c("Simulation 1", "Simulation 2"),
       col=c("red", "green"), pch=c(1,3), cex=0.8)
```



### 3.4

#### a

```
library(aplore3) ; data(icu) ; help(icu)
str(icu)
```

```
## 'data.frame':    200 obs. of  21 variables:
##  $ id    : int  4 8 12 14 27 28 32 38 40 41 ...
##  $ sta   : Factor w/ 2 levels "Lived","Died": 2 1 1 1 2 1 1 1 1 1 ...
##  $ age   : int  87 27 59 77 76 54 87 69 63 30 ...
##  $ gender: Factor w/ 2 levels "Male","Female": 2 2 1 1 2 1 2 1 1 2 ...
##  $ race  : Factor w/ 3 levels "White","Black",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ ser   : Factor w/ 2 levels "Medical","Surgical": 2 1 1 2 2 1 2 1 2 1 ...
##  $ can   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ crn   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ inf   : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 2 2 2 1 1 ...
##  $ cpr   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sys   : int  80 142 112 100 128 142 110 110 104 144 ...
##  $ hra   : int  96 88 80 70 90 103 154 132 66 110 ...
##  $ pre   : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 1 2 1 1 1 ...
##  $ type  : Factor w/ 2 levels "Elective","Emergency": 2 2 2 1 2 2 2 2 1 2 ...
##  $ fra   : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 2 1 1 1 1 ...
##  $ po2   : Factor w/ 2 levels "> 60","<= 60": 2 1 1 1 1 1 1 2 1 1 ...
##  $ ph    : Factor w/ 2 levels ">= 7.25","< 7.25": 2 1 1 1 1 1 1 1 1 1 ...
##  $ pco   : Factor w/ 2 levels "<= 45","> 45": 2 1 1 1 1 1 1 1 1 1 ...
##  $ bic   : Factor w/ 2 levels ">= 18","< 18": 1 1 1 1 1 1 1 2 1 1 ...
##  $ cre   : Factor w/ 2 levels "<= 2.0","> 2.0": 1 1 1 1 1 1 1 1 1 1 ...
##  $ loc   : Factor w/ 3 levels "Nothing","Stupor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

**b**

```
library(gam)
```

```
## Loaded gam 1.22-2
```

```
fitInitial <- gam::gam(sta ~age+gender+race+ser+can+crn+inf+cpr+sys+hra+pre+type+fra+po2+ph+pco+bic+cr
stepFit <- step.Gam(fitInitial, scope =
                      list(
                      "age" = ~ 1 + age + s(age,2),
                      "gender" = ~ 1 + gender,
                      "race" = ~ 1 + race ,
                      "ser" = ~ 1 + ser,
                      "can" = ~ 1 + can,
                      "crn" = ~ 1 + crn,
                      "inf" = ~ 1+ inf,
                      "cpr" = ~ 1 + cpr,
                      "sys" = ~ 1 + sys + s(sys,2),
                      "hra" = ~ 1+ hra + s(hra,2),
                      "pre" = ~ 1 + pre,
                      "type" = ~ 1 + type ,
                      "fra" = ~ 1 + fra,
                      "po2" = ~ 1 + po2,
                      "ph" = ~ 1 + ph,
                      "pco" = ~ 1 + pco,
                      "bic" = ~ 1 + bic,
                      "cre" = ~ 1 + cre ,
                      "loc" = ~ 1 + loc
                      ))
```

```
## Start:  sta ~ age + gender + race + ser + can + crn + inf + cpr + sys +      hra + pre + type + fra
## Step:1 sta ~ age + gender + race + ser + can + crn + inf + cpr + sys +      hra + pre + type + fra +
## Step:2 sta ~ age + gender + race + ser + can + inf + cpr + sys + hra +      pre + type + fra + po2 +
## Step:3 sta ~ age + gender + race + ser + can + cpr + sys + hra + pre +      type + fra + po2 + ph +
## Step:4 sta ~ age + gender + race + ser + can + cpr + sys + hra + pre +      type + fra + po2 + ph +
## Step:5 sta ~ age + gender + race + ser + can + cpr + sys + pre + type +      fra + po2 + ph + pco +
## Step:6 sta ~ age + gender + race + ser + can + cpr + sys + pre + type +      fra + ph + pco + loc ;
## Step:7 sta ~ age + gender + race + can + cpr + sys + pre + type + fra +      ph + pco + loc ; AIC= 1
## Step:8 sta ~ age + gender + can + cpr + sys + pre + type + fra + ph +      pco + loc ; AIC= 143.2064
## Step:9 sta ~ age + gender + can + cpr + sys + pre + type + ph + pco +      loc ; AIC= 142.4638
## Step:10 sta ~ s(age, 2) + gender + can + cpr + sys + pre + type + ph +      pco + loc ; AIC= 141.744
```

```
print(names(stepFit$"model")[-1])
```

```
##  [1] "s(age, 2)" "gender"    "can"       "cpr"       "sys"       "pre"
##  [7] "type"      "ph"        "pco"       "loc"
```

```
detach(package:gam)
```

### c

```
library(mgcv)
```

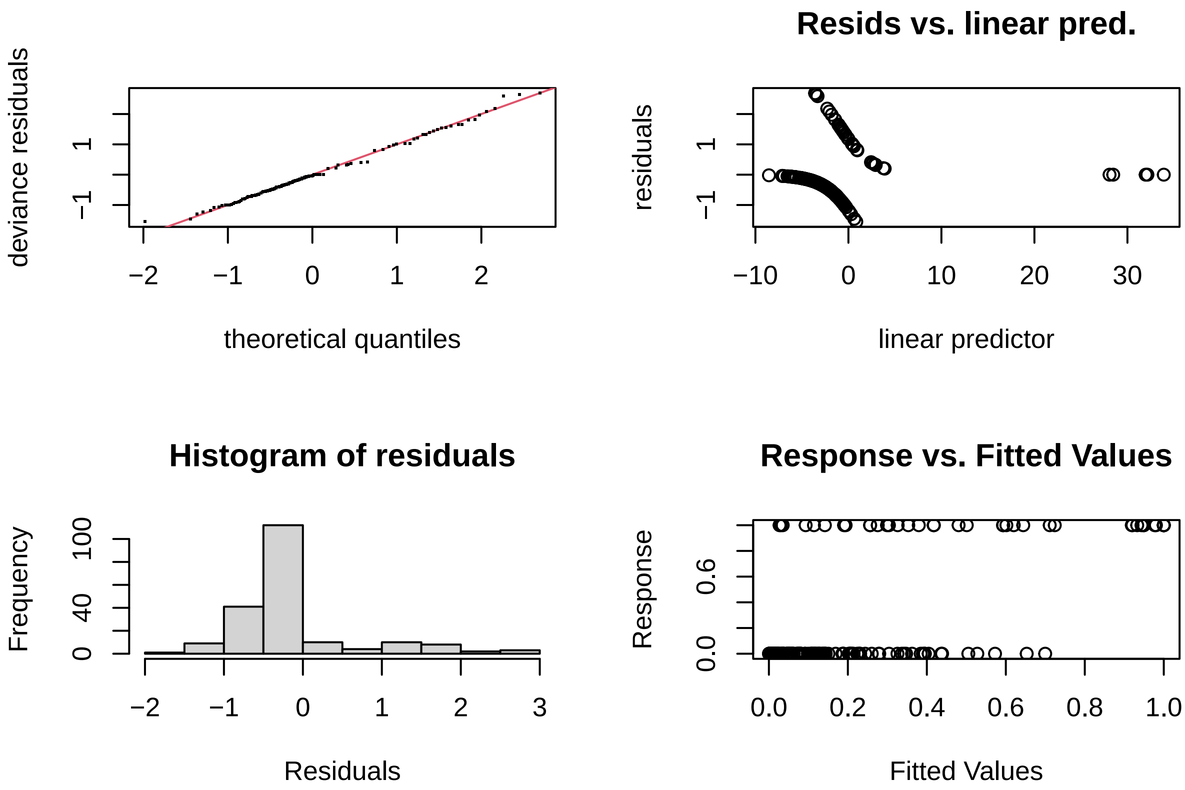
```
## Loading required package: nlme
```

```
## This is mgcv 1.8-42. For overview type 'help("mgcv-package")'.
```

```
fitgam<-mgcv:::gam(sta~s(age)+as.factor(gender)+as.factor(can)+as.factor(cpr)+sys+as.factor(pre)+as.fac
summary(fitgam)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## sta ~ s(age) + as.factor(gender) + as.factor(can) + as.factor(cpr) +
##     sys + as.factor(pre) + as.factor(type) + as.factor(ph) +
##     as.factor(pco) + as.factor(loc)
##
## Parametric coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -3.181e+00  1.562e+00  -2.037  0.04165 *
## as.factor(gender)Female  -7.100e-01  5.201e-01  -1.365  0.17219
## as.factor(can)Yes         3.316e+00  1.045e+00   3.174  0.00151 **
## as.factor(cpr)Yes         1.207e+00  8.596e-01   1.404  0.16041
## sys                      -1.970e-02  8.178e-03  -2.408  0.01602 *
## as.factor(pre)Yes         1.044e+00  6.408e-01   1.630  0.10315
## as.factor(type)Emergency  3.948e+00  1.290e+00   3.060  0.00221 **
## as.factor(ph)< 7.25       1.835e+00  9.716e-01   1.889  0.05888 .
## as.factor(pco)> 45       -2.092e+00  9.967e-01  -2.099  0.03582 *
## as.factor(loc)Stupor      3.405e+01  5.860e+05   0.000  0.99995
## as.factor(loc)Coma        3.286e+00  1.129e+00   2.911  0.00360 **
## ---
## 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) 1.715  2.144  12.94  0.0019 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## R-sq.(adj) =  0.418   Deviance explained =   42%
## UBRE = -0.2921  Scale est. = 1         n = 200
```

```
gam.check(fitgam)
```



```
##
## Method: UBRE   Optimizer: outer newton
## full convergence after 3 iterations.
## Gradient range [-1.28899e-07,-1.28899e-07]
## (score -0.2921034 & scale 1).
## Hessian positive definite, eigenvalue range [0.002457308,0.002457308].
## Model rank =  20 / 20
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##           k'  edf k-index p-value
## s(age) 9.00 1.71    1.07    0.93
```

### d

```
newdatadf<-data.frame(
  gender="Female",
  age=79,
  race="White",
  ser="Medical",
  can="No",
  crn="No",
  inf="No",
  cpr="No",
  sys=228,
  hra=94,
  pre="No",
  type="Emergency",
  fra="No",
  po2="> 45",
  ph=">= 7.25",
  pco="> 45",
  bic="< 18",
  cre="> 2.0",
  loc="Coma"
)
predObjdir <- predict(fitgam,newdata = newdatadf,
                      type = "response",se.fit = TRUE)
print(predObjdir)
```

```
## $fit
##         1
## 0.1088643
##
## $se.fit
##         1
## 0.1431123
```

### 3.7

#### a

```
library(kernlab) ; data(spam) ; help(spam)
print(names(spam))
```

```
##  [1] "make"              "address"           "all"
##  [4] "num3d"             "our"               "over"
##  [7] "remove"            "internet"          "order"
## [10] "mail"              "receive"           "will"
## [13] "people"            "report"            "addresses"
## [16] "free"              "business"          "email"
## [19] "you"               "credit"            "your"
## [22] "font"              "num000"            "money"
## [25] "hp"                "hpl"               "george"
## [28] "num650"            "lab"               "labs"
## [31] "telnet"            "num857"            "data"
## [34] "num415"            "num85"             "technology"
## [37] "num1999"           "parts"             "pm"
## [40] "direct"            "cs"                "meeting"
## [43] "original"          "project"           "re"
## [46] "edu"               "table"             "conference"
## [49] "charSemicolon"     "charRoundbracket"  "charSquarebracket"
## [52] "charExclamation"   "charDollar"        "charHash"
## [55] "capitalAve"        "capitalLong"       "capitalTotal"
## [58] "type"
```

### b

```
set.seed(1) ; nTest <- 1000
indsTest <- sample(1:nrow(spam),nTest,replace = FALSE)
indsTrain <- setdiff(1:nrow(spam),indsTest)
spamTest <- spam[indsTest,]
spamTrain <- spam[indsTrain,]
```

### c

```
fitTrainFullGLM <- glm(type ~ .,family = binomial,data = spamTrain)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
print(summary(fitTrainFullGLM))
```

```
##
## Call:
## glm(formula = type ~ ., family = binomial, data = spamTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8117  -0.1976   0.0000   0.1195   5.5509
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.628e+00  1.621e-01 -10.044  < 2e-16 ***
## make              -3.735e-01  2.530e-01  -1.476 0.139896
## address           -1.227e-01  7.299e-02  -1.681 0.092696 .
## all                1.743e-01  1.216e-01   1.433 0.151749
## num3d              2.625e+00  1.724e+00   1.522 0.127893
## our                7.189e-01  1.193e-01   6.024 1.70e-09 ***
## over               1.063e+00  2.987e-01   3.559 0.000372 ***
## remove             2.016e+00  3.318e-01   6.076 1.23e-09 ***
## internet           4.799e-01  1.634e-01   2.938 0.003308 **
## order              5.627e-01  3.111e-01   1.808 0.070530 .
## mail               9.162e-02  7.411e-02   1.236 0.216382
## receive           -2.438e-01  3.258e-01  -0.749 0.454116
## will              -1.448e-01  8.517e-02  -1.700 0.089186 .
## people            -7.797e-03  2.673e-01  -0.029 0.976728
## report             4.348e-02  1.552e-01   0.280 0.779401
## addresses          1.159e+00  7.084e-01   1.636 0.101773
## free               1.081e+00  1.613e-01   6.701 2.06e-11 ***
## business           7.959e-01  2.330e-01   3.416 0.000635 ***
## email              1.852e-01  1.260e-01   1.470 0.141525
## you                7.217e-02  3.914e-02   1.844 0.065191 .
## credit             1.009e+00  6.118e-01   1.650 0.099038 .
## your               2.170e-01  5.759e-02   3.767 0.000165 ***
## font               3.092e-01  2.245e-01   1.377 0.168568
## num000             2.336e+00  5.286e-01   4.420 9.86e-06 ***
## money              4.503e-01  1.676e-01   2.687 0.007206 **
## hp                -2.008e+00  3.624e-01  -5.540 3.03e-08 ***
## hpl               -7.920e-01  4.435e-01  -1.786 0.074096 .
## george            -1.225e+01  2.509e+00  -4.883 1.05e-06 ***
## num650             4.961e-01  3.478e-01   1.426 0.153778
## lab               -2.065e+00  1.411e+00  -1.464 0.143189
## labs              -3.934e-01  3.795e-01  -1.037 0.299902
## telnet            -1.171e-01  3.728e-01  -0.314 0.753565
## num857            -7.662e+01  4.207e+03  -0.018 0.985470
## data              -1.073e+00  3.941e-01  -2.722 0.006483 **
## num415             1.203e+00  1.762e+00   0.683 0.494604
## num85             -2.035e+00  8.535e-01  -2.385 0.017096 *
## technology         6.414e-01  3.555e-01   1.805 0.071139 .
## num1999            1.051e-01  1.923e-01   0.546 0.584808
## parts              1.680e+00  9.658e-01   1.739 0.082046 .
## pm                -6.455e-01  4.491e-01  -1.437 0.150654
## direct            -3.202e-01  3.796e-01  -0.844 0.398907
## cs                -4.643e+01  2.658e+01  -1.747 0.080673 .
## meeting           -3.196e+00  1.115e+00  -2.867 0.004144 **
## original          -7.022e-01  7.233e-01  -0.971 0.331654
## project           -1.827e+00  6.341e-01  -2.880 0.003971 **
## re                -7.023e-01  1.527e-01  -4.601 4.21e-06 ***
## edu               -1.329e+00  2.841e-01  -4.678 2.89e-06 ***
## table             -1.444e+00  1.859e+00  -0.777 0.437339
## conference        -4.568e+00  1.947e+00  -2.347 0.018938 *
## charSemicolon     -1.703e+00  6.358e-01  -2.679 0.007394 **
## charRoundbracket  -1.464e-01  3.081e-01  -0.475 0.634607
## charSquarebracket -6.028e-01  1.060e+00  -0.569 0.569422
## charExclamation    2.419e-01  6.792e-02   3.561 0.000369 ***
## charDollar         4.373e+00  7.422e-01   5.891 3.84e-09 ***
## charHash           2.843e+00  1.162e+00   2.446 0.014434 *
## capitalAve         9.987e-03  2.087e-02   0.479 0.632243
## capitalLong        9.190e-03  2.875e-03   3.196 0.001392 **
## capitalTotal       1.178e-03  2.520e-04   4.675 2.94e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4806.0  on 3600  degrees of freedom
## Residual deviance: 1413.2  on 3543  degrees of freedom
## AIC: 1529.2
##
## Number of Fisher Scoring iterations: 22
```

### d

```
library(gam)
```

```
## Loaded gam 1.22-2

##
## Attaching package: 'gam'

## The following objects are masked from 'package:mgcv':
##
##     gam, gam.control, gam.fit, s
```

```
fitgam3<-gam::gam(type ~ .,family = binomial,data = spamTrain)
spam_scope<-gam.scope(spamTrain,arg="df=2",response=58)
stepfit3<-step.Gam(fitgam3,scope=spam_scope)
```

```
## Start:  type ~ .; AIC= 1529.234
## Step:1 type ~ make + address + all + num3d + our + over + remove + internet +      order + mail + rec
## Step:2 type ~ make + address + all + num3d + our + over + remove + internet +      order + mail + rec
## Step:3 type ~ make + address + all + num3d + our + over + remove + internet +      order + mail + rec
## Step:4 type ~ make + address + all + num3d + s(our, df = 2) + over +      remove + internet + order +
## Step:5 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + interne
## Step:6 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + interne
## Step:7 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + interne
## Step:8 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + interne
## Step:9 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + interne
## Step:10 type ~ make + address + all + num3d + s(our, df = 2) + over +      s(remove, df = 2) + intern
```

```
detach(package:gam)
```

### d

```
func<-names(stepfit3$"model")[-1]
print(func)
```

```
##  [1] "make"                       "address"
##  [3] "all"                        "num3d"
##  [5] "s(our, df = 2)"             "over"
##  [7] "s(remove, df = 2)"          "internet"
##  [9] "order"                      "mail"
## [11] "receive"                    "will"
## [13] "people"                     "report"
## [15] "addresses"                  "free"
## [17] "business"                   "email"
## [19] "you"                        "credit"
## [21] "your"                       "font"
## [23] "num000"                     "s(money, df = 2)"
## [25] "s(hp, df = 2)"              "hpl"
## [27] "s(george, df = 2)"          "s(num650, df = 2)"
## [29] "lab"                        "labs"
## [31] "num857"                     "data"
## [33] "num415"                     "num85"
## [35] "technology"                 "num1999"
## [37] "parts"                      "pm"
## [39] "direct"                     "cs"
## [41] "meeting"                    "original"
## [43] "project"                    "re"
## [45] "edu"                        "table"
## [47] "conference"                 "charSemicolon"
## [49] "charRoundbracket"           "charSquarebracket"
## [51] "s(charExclamation, df = 2)" "s(charDollar, df = 2)"
## [53] "charHash"                   "capitalAve"
## [55] "capitalLong"                "s(capitalTotal, df = 2)"
```

```
library(mgcv)
fitmgcvgam3<-mgcv:::gam(type~make+address+all+num3d+s(our)+over+s(remove)+internet+order+mail+receive+wi
```

```
## Warning in newton(lsp = lsp, X = G$X, y = G$y, Eb = G$Eb, UrS = G$UrS, L =
## G$L, : Iteration limit reached without full convergence - check carefully
```

```
summary(fitmgcvgam3)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## type ~ make + address + all + num3d + s(our) + over + s(remove) +
##     internet + order + mail + receive + will + people + report +
##     addresses + free + business + email + you + credit + your +
##     font + num000 + s(money) + s(hp) + s(george) + hpl + s(num650) +
##     lab + labs + num857 + data + num415 + num85 + technology +
##     num1999 + parts + pm + direct + cs + meeting + original +
##     project + re + edu + table + conference + charSemicolon +
##     charRoundbracket + charSquarebracket + s(charExclamation) +
##     s(charDollar) + charHash + capitalAve + capitalLong + s(capitalTotal)
##
## Parametric coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -9.887e+01  4.014e+02  -0.246 0.805442
## make              -4.111e-01  3.153e-01  -1.304 0.192239
## address           -7.157e-02  9.150e-02  -0.782 0.434094
## all               -2.896e-01  1.732e-01  -1.672 0.094511 .
## num3d              1.147e+00  2.158e+00   0.531 0.595095
## over               6.163e-01  2.787e-01   2.211 0.027012 *
## internet           5.243e-01  1.514e-01   3.464 0.000533 ***
## order              2.242e-01  3.020e-01   0.742 0.457819
## mail               5.617e-02  8.421e-02   0.667 0.504754
## receive            1.549e-02  4.057e-01   0.038 0.969548
## will              -2.090e-01  1.010e-01  -2.069 0.038523 *
## people            -4.951e-01  3.494e-01  -1.417 0.156531
## report             1.493e-01  1.674e-01   0.892 0.372469
## addresses          8.597e-01  7.469e-01   1.151 0.249767
## free               6.958e-01  1.460e-01   4.766 1.88e-06 ***
## business           6.641e-01  2.494e-01   2.663 0.007749 **
## email              1.556e-01  1.421e-01   1.095 0.273665
## you                1.090e-01  4.842e-02   2.250 0.024417 *
## credit             4.523e-01  4.989e-01   0.907 0.364625
## your               1.326e-01  6.916e-02   1.917 0.055291 .
## font               2.695e-01  2.128e-01   1.266 0.205348
## num000             1.495e+00  4.707e-01   3.176 0.001492 **
## hpl               -1.007e-02  2.223e-01  -0.045 0.963850
## lab               -1.722e+00  1.582e+00  -1.088 0.276503
## labs              -3.610e-01  4.613e-01  -0.783 0.433882
## num857            -4.555e+02  3.105e+06   0.000 0.999883
## data              -6.048e-01  3.556e-01  -1.701 0.088990 .
## num415             7.571e-01  3.285e+00   0.230 0.817712
## num85             -1.760e+00  9.593e-01  -1.834 0.066618 .
## technology         9.265e-01  4.628e-01   2.002 0.045285 *
## num1999            2.093e-03  2.249e-01   0.009 0.992575
## parts              1.023e+00  1.538e+00   0.665 0.505978
## pm                -7.814e-01  5.717e-01  -1.367 0.171635
## direct            -2.785e-01  5.923e-01  -0.470 0.638187
## cs                -7.056e+01  2.678e+01  -2.635 0.008414 **
## meeting           -3.630e+00  1.488e+00  -2.439 0.014727 *
## original          -4.639e-01  8.065e-01  -0.575 0.565113
## project           -1.969e+00  8.587e-01  -2.293 0.021838 *
## re                -5.422e-01  1.797e-01  -3.018 0.002547 **
## edu               -1.238e+00  3.000e-01  -4.127 3.68e-05 ***
## table             -2.247e+00  2.790e+00  -0.805 0.420572
## conference        -5.301e+00  2.781e+00  -1.906 0.056688 .
## charSemicolon     -1.728e+00  6.326e-01  -2.732 0.006292 **
## charRoundbracket  -6.753e-01  3.696e-01  -1.827 0.067706 .
## charSquarebracket -1.185e-01  9.750e-01  -0.122 0.903241
## charHash           1.252e+00  1.562e+00   0.802 0.422734
## capitalAve         2.930e-02  2.342e-02   1.251 0.210803
## capitalLong        1.219e-03  3.209e-03   0.380 0.704066
## ---
## 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(our)             6.510  7.497  65.54  < 2e-16 ***
## s(remove)          6.939  7.410  41.40  < 2e-16 ***
## s(money)           3.483  4.137  28.48 1.29e-05 ***
## s(hp)              5.104  5.252  60.63  < 2e-16 ***
## s(george)          1.000  1.000  12.44  0.00042 ***
## s(num650)          3.807  4.095  27.11 2.29e-05 ***
## s(charExclamation) 4.172  5.045 107.01  < 2e-16 ***
## s(charDollar)      3.959  4.753  44.85  < 2e-16 ***
## s(capitalTotal)    8.899  8.990  69.73  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
```

```
## R-sq.(adj) =  0.833   Deviance explained =   79%
## UBRE = -0.66827  Scale est. = 1         n = 3601
```

**e**

```r
library(caret)
```

```
## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:kernlab':
##
##     alpha

## Loading required package: lattice
```

```r
pdata <- predict(fitmgcvgam3, newdata = spamTest, type = "response")
SpamTestpred = rep("nonspam", dim(spamTest)[1])
SpamTestpred[pdata>0.5]="spam"
p<-confusionMatrix(data = as.factor(SpamTestpred), reference = spamTest$type)
print(p)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction nonspam spam
##    nonspam     557   43
##    spam         23  377
##
##                Accuracy : 0.934
##                  95% CI : (0.9168, 0.9486)
##     No Information Rate : 0.58
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8636
##
##  Mcnemar's Test P-Value : 0.01935
##
##             Sensitivity : 0.9603
##             Specificity : 0.8976
##          Pos Pred Value : 0.9283
##          Neg Pred Value : 0.9425
##              Prevalence : 0.5800
##          Detection Rate : 0.5570
##    Detection Prevalence : 0.6000
##       Balanced Accuracy : 0.9290
##
##        'Positive' Class : nonspam
##
```

```
print("classification error:")
```

```
## [1] "classification error:"
```

```
print(1-p$overall[1])
```

```
## Accuracy
##    0.066
```

### f

```
library(mgcv)
fitmgcvgam4<-mgcv::gam(type~s(make)+s(address)+s(all)+s(num3d)+s(our)+s(over)+s(remove)+s(internet)+s(
```

```
library(caret)
pdata <- predict(fitmgcvgam4, newdata = spamTest, type = "response")
SpamTestpred = rep("nonspam", dim(spamTest)[1])
SpamTestpred[pdata>0.5]="spam"
p<-confusionMatrix(data = as.factor(SpamTestpred), reference = spamTest$type)
print(p)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction nonspam spam
##    nonspam     553   36
##    spam         27  384
##
##                Accuracy : 0.937
##                  95% CI : (0.9201, 0.9513)
##     No Information Rate : 0.58
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8703
##
##  Mcnemar's Test P-Value : 0.3135
##
##             Sensitivity : 0.9534
##             Specificity : 0.9143
##          Pos Pred Value : 0.9389
##          Neg Pred Value : 0.9343
##              Prevalence : 0.5800
##          Detection Rate : 0.5530
##    Detection Prevalence : 0.5890
##       Balanced Accuracy : 0.9339
##
##        'Positive' Class : nonspam
##
```

```
print("classification error:")
```

```
## [1] "classification error:"
```

```
print(1-p$overall[1])
```

```
## Accuracy
##    0.063
```

### g

For this scenario, One rule is not significantly better than other. This is seen because the accuracy/classification error seems to be very much similar. So adding spline coefficients for all the terms didnt significantly improve the model but it surely increased the computational time. One more reason maybe because of the overly complicated model with lot of features, doesnt really affect when all the terms are added for spline coefficients