Task 7

for Advanced Methods for Regression and Classification

Teodor Chakarov

21.12.2022

Contents


```
library(mgcv)
```

```
## Loading required package: nlme

## This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

```
library(ISLR)
```

Task 1

Load data and split for training

Im going to load the data and inspect it.

```
data(OJ ,package="ISLR")

df <- OJ

df <- na.omit(df)

str(df)
```

```
## 'data.frame':    1070 obs. of  18 variables:
##  $ Purchase      : Factor w/ 2 levels "CH","MM": 1 1 1 2 1 1 1 1 1 1 ...
##  $ WeekofPurchase: num  237 239 245 227 228 230 232 234 235 238 ...
##  $ StoreID       : num  1 1 1 1 7 7 7 7 7 7 ...
##  $ PriceCH       : num  1.75 1.75 1.86 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
```

```
##  $ PriceMM       : num  1.99 1.99 2.09 1.69 1.69 1.99 1.99 1.99 1.99 1.99 ...
##  $ DiscCH        : num  0 0 0.17 0 0 0 0 0 0 0 ...
##  $ DiscMM        : num  0 0.3 0 0 0 0 0.4 0.4 0.4 0.4 ...
##  $ SpecialCH     : num  0 0 0 0 0 0 1 1 0 0 ...
##  $ SpecialMM     : num  0 1 0 0 0 1 1 0 0 0 ...
##  $ LoyalCH       : num  0.5 0.6 0.68 0.4 0.957 ...
##  $ SalePriceMM   : num  1.99 1.69 2.09 1.69 1.69 1.99 1.59 1.59 1.59 1.59 ...
##  $ SalePriceCH   : num  1.75 1.75 1.69 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
##  $ PriceDiff     : num  0.24 -0.06 0.4 0 0 0.3 -0.1 -0.16 -0.16 -0.16 ...
##  $ Store7        : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 2 2 2 ...
##  $ PctDiscMM     : num  0 0.151 0 0 0 ...
##  $ PctDiscCH     : num  0 0 0.0914 0 0 ...
##  $ ListPriceDiff : num  0.24 0.24 0.23 0 0 0.3 0.3 0.24 0.24 0.24 ...
##  $ STORE         : num  1 1 1 1 0 0 0 0 0 0 ...
```

Split the data to train/test

```
set.seed(1234555)

train_ind = sample(1:nrow(OJ), 0.66 * nrow(OJ))
train <- OJ[train_ind,]
test <- OJ[-train_ind,]

# Setting the y to be "Apps"
#y_train = train[ , which(names(train) %in% c("Apps"))]
#y_test = test[ , which(names(test) %in% c("Apps"))]

# Removing the predictive variable from the training and testing sets.
#x_train = train[ , -which(names(train) %in% c("Apps"))]
#x_test = test[ , -which(names(test) %in% c("Apps"))]
```

Fit the data to GAM model

I will fit all of the parameters but not going to assign smoothing function for the categorical attributes, they will remain as a factor.

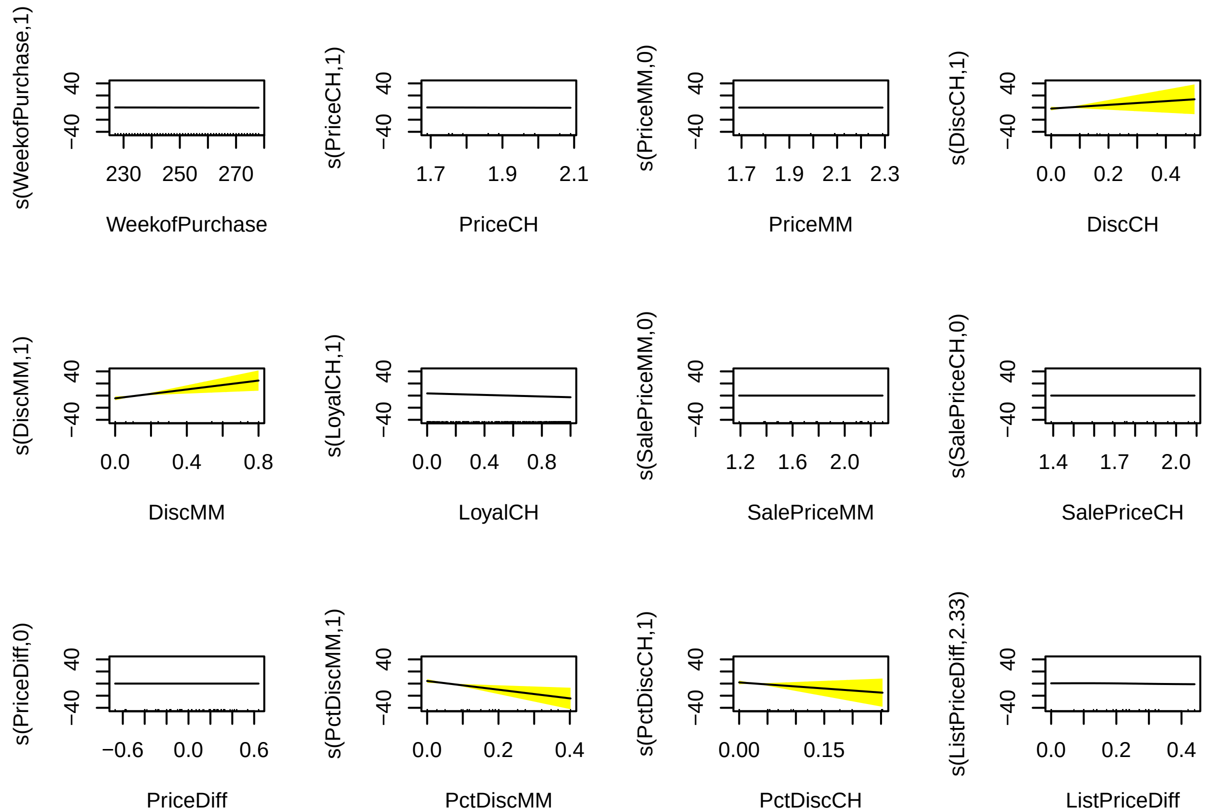
```
p <- 5

gam_mod <- gam(Purchase ~ s(WeekofPurchase, k=p) + factor(StoreID) + s(PriceCH, k=p) + s(PriceMM, k=p) +

summary(gam_mod)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## Purchase ~ s(WeekofPurchase, k = p) + factor(StoreID) + s(PriceCH,
##     k = p) + s(PriceMM, k = p) + s(DiscCH, k = p) + s(DiscMM,
##     k = p) + factor(SpecialCH) + factor(SpecialMM) + s(LoyalCH,
##     k = p) + s(SalePriceMM, k = p) + s(SalePriceCH, k = p) +
##     s(PriceDiff, k = p) + Store7 + s(PctDiscMM, k = p) + s(PctDiscCH,
```

```
##     k = p) + s(ListPriceDiff, k = p) + factor(STORE)
##
## Parametric coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -1.1404     0.4249  -2.684  0.00728 **
## factor(StoreID)2     0.0000     0.0000     NaN      NaN
## factor(StoreID)3     0.0000     0.0000     NaN      NaN
## factor(StoreID)4     0.0000     0.0000     NaN      NaN
## factor(StoreID)7     0.0000     0.0000     NaN      NaN
## factor(SpecialCH)1   0.6055     0.4240   1.428  0.15322
## factor(SpecialMM)1   0.2262     0.3464   0.653  0.51377
## Store7Yes           -0.3511     0.5221  -0.672  0.50135
## factor(STORE)1       0.4600     0.5765   0.798  0.42491
## factor(STORE)2       0.3643     0.5180   0.703  0.48185
## factor(STORE)3       0.5361     0.4522   1.185  0.23585
## factor(STORE)4       0.0000     0.0000     NaN      NaN
## ---
## 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(WeekofPurchase) 1.000e+00 1.000e+00   0.275 0.59991
## s(PriceCH)        1.000e+00 1.000e+00   0.264 0.60738
## s(PriceMM)        2.365e-05 4.486e-05   0.000 0.99837
## s(DiscCH)         1.000e+00 1.000e+00   1.243 0.26485
## s(DiscMM)         1.000e+00 1.000e+00   8.784 0.00304 **
## s(LoyalCH)        1.000e+00 1.000e+00 151.026 < 2e-16 ***
## s(SalePriceMM)    7.250e-06 1.434e-05   0.000 0.50000
## s(SalePriceCH)    1.343e-05 2.683e-05   0.000 0.99925
## s(PriceDiff)      9.407e-06 1.866e-05   0.000 0.99889
## s(PctDiscMM)      1.000e+00 1.000e+00   7.679 0.00559 **
## s(PctDiscCH)      1.000e+00 1.000e+00   1.658 0.19789
## s(ListPriceDiff)  2.326e+00 2.882e+00  10.718 0.01410 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 51/60
## R-sq.(adj) =  0.518   Deviance explained = 45.7%
## UBRE = -0.22831  Scale est. = 1         n = 706
```

Based on the model I constructed we can see that the majority of the coefficients are not significant for the model at all. For the parametric coefficients we dont have significance. For the approximated smooth therms we have significance only on LoyalCH, DiscMM, ListPriceDiff. We can also see with the help of **edf** column that for LoyalCH, DiscMM, DiscCH, PriceCH, PctDiscMM, PctDiscCH we have 1, which meand the fit is straight line. We can see that ont the next plot.

```
plot(gam_mod,page=1,shade=TRUE,shade.col="yellow")
```



We have a lot of attributes fitted with a straight line. For the attribute LoyalCH and ListPriceDiff we can see the who the line describes the data itself.

```
#x_test = test[ , -which(names(test) %in% c("Purchase"))]

gam.res <- predict(gam_mod, test)>0.5
gam.TAB <- table(test$Purchase,as.numeric(gam.res))
gam.TAB
```

```
##
##        0   1
##   CH 202  19
##   MM  55  88
```

```
print('Misclassification error:')
```

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

```
print(1-sum(diag(gam.TAB))/sum(gam.TAB))
```

```
## [1] 0.2032967
```

And with the miss-Classifications error we can see that our model produces not bad results. We can also see from the confusion matrix that we have more False Positives (the class MM is miss-classified).

Model Optimization

For Model optimization I will pick by hand the attributes based on the significance from the previous model. I will get rid of the categorical variables and also reduce the degrees of freedom to 2.

```
p <- 2

gam_mod <- gam(Purchase ~ s(DiscMM, k=p) + factor(SpecialCH) + factor(SpecialMM) + s(LoyalCH, k=p) + s(I
```

```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased
```

```
#summary(gam_mod)

gam.res <- predict(gam_mod, test)>0.5
gam.TAB <- table(test$Purchase,as.numeric(gam.res))
gam.TAB
```

```
##
##        0   1
##   CH 205  16
##   MM  52  91
```

```
print('Misclassification error:')
```

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

```
print(1-sum(diag(gam.TAB))/sum(gam.TAB))
```

```
## [1] 0.1868132
```

At the end we reduced our Misclassification error and we also have model with less parameters than the previous.