UNIVERSITY OF TORONTO MISSISSAUGA STA314 KAGGLE PREDICTION COMPETITION 2021

Code

Hide

```
# Set working directory
setwd("C:/Users/amrin/OneDrive/Desktop/sta314 kaggle competition")
```

```
Warning: The working directory was changed to C:/Users/amrin/OneDrive/Desktop/sta314 kaggle comp
etition inside a notebook chunk. The working directory will be reset when the chunk is finished
running. Use the knitr root.dir option in the setup chunk to change the working directory for no
tebook chunks.
```

Hide

```
# Read in the data
# d.train is the training set
d.train = read.csv('trainingdata.csv')
# d.test are the predictors in the test set
d.test = read.csv('test_predictors.csv')
```

```
Warning: stack imbalance in '=', 2 then 4
```

Hide

```
# Load Libraries
#install.packages('glmnet')
#install.packages('gbm')
#install.packages('splines')
#install.packages('gam')
library(glmnet)
library(gbm)
library('splines')
library('gam')

# Response variable
y = d.train$y

# Explanatory variable
x = model.matrix(d.train$y ~ . ,d.train)[,-1]

# Set random seed to 1
set.seed(1)

# Randomly sample half of given data set for training model
train = sample(1:nrow(x),nrow(x)/2)

# Separate other half of given data set for testing model
test = (-train)

# Response variable of test data set
ytest = y[test]

################################ LASSO Regression ################################
# Create grid for lambda
grid = 10^seq(10,-2,length = 100)

#Base lasso model
lasso.mod = glmnet(x[train,],y[train],alpha =1, lambda = grid)

# Set Random Seed
set.seed(1)

# Run cross validation for lasso to choose optimal lambda value
cv.la = cv.glmnet(x[train,],y[train],alpha =1, lambda = grid)

# Optimal Lambda value for lasso model
lambda_best <- cv.la$lambda.min

# Optimal lasso model
la = glmnet(x[train,],y[train],alpha =1, lambda = lambda_best)

# Training set prediction
predictions_train <- predict(la, s = lambda_best, newx = x[train,])

# Rooted mean squared error
sqrt(mean((predictions_train - y[train])^2))
```

```
[1] 3.769483
```

Hide

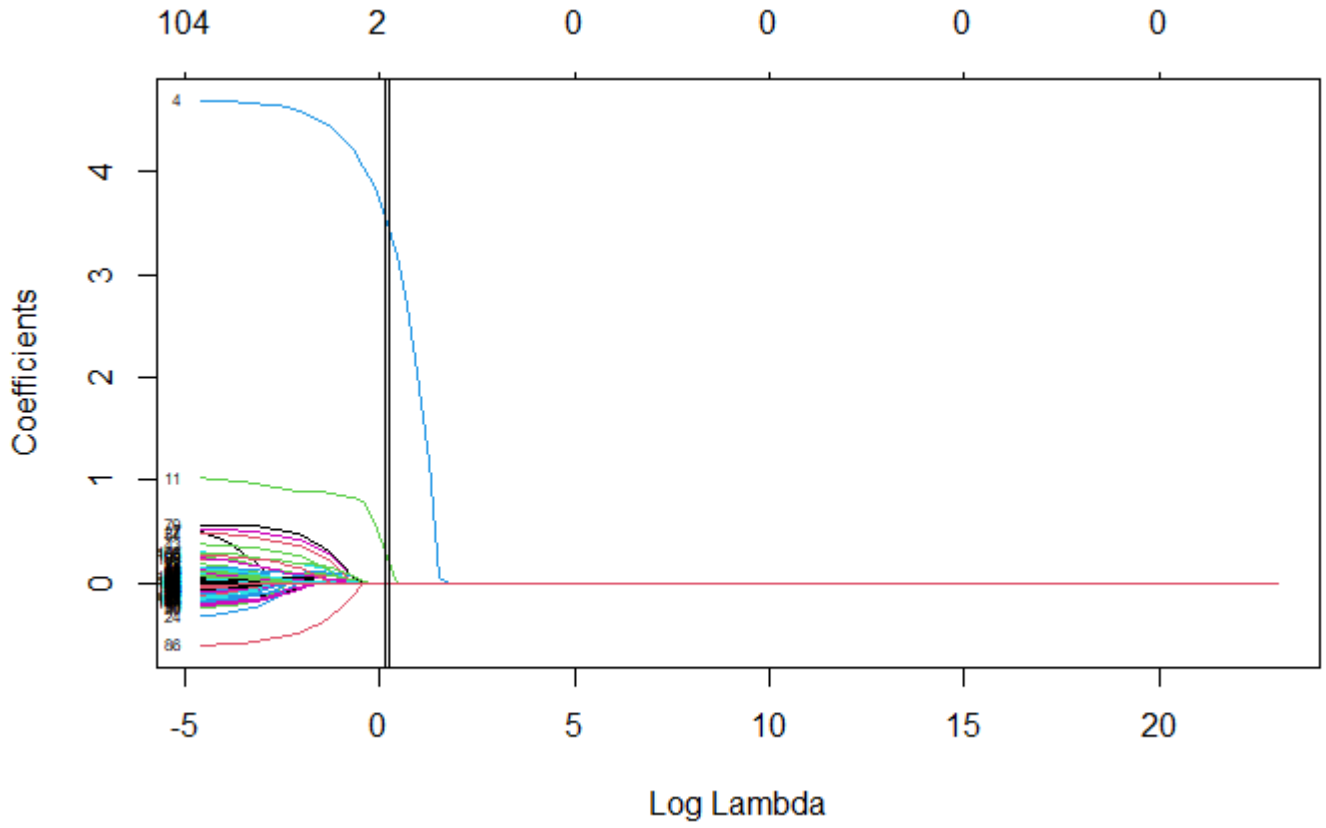
```
# Plot lasso model
plot(lasso.mod, label = TRUE,xvar = 'lambda', cex=0.5)

# Plot optimal lambda via cross validation
abline(v=cv.la$lambda.min, add=T)
```

```
Warning in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...) :
  "add" is not a graphical parameter
```

Hide

```
# Plot optimal lambda via one standard error rule
abline(v=cv.la$lambda.1se, add=T)
```

```
Warning in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...) :
  "add" is not a graphical parameter
```



Hide

```
# Show top variable selected by lasso regression to predict response variable
which(coef(lasso.mod)[,100]>0)
```

```
(Intercept)          X1          X2          X3          X4          X6          X7          X9
X11
          1           2           3           4           5           7           8          10
12
        X14         X16         X17         X19         X22         X23         X25         X27
X33
         15          17          18          20          23          24          26          28
34
        X35         X37         X39         X41         X42         X43         X45         X46
X47
         36          38          40          42          43          44          46          47
48
        X51         X54         X57         X59         X60         X61         X64         X65
X67
         52          55          58          60          61          62          65          66
68
        X68         X71         X74         X75         X76         X79         X80         X81
X82
         69          72          75          76          77          80          81          82
83
        X84         X85         X87         X89         X92         X94         X96        X100
X103
         85          86          88          90          93          95          97         101
104
       X104        X105        X106        X111
       105         106         107         112
```

Hide

```
# Test set prediction
pred_test <- predict(la, s = lambda_best, newx = x[test,])

# Rooted mean squared error
sqrt(mean((pred_test - ytest)^2))
```

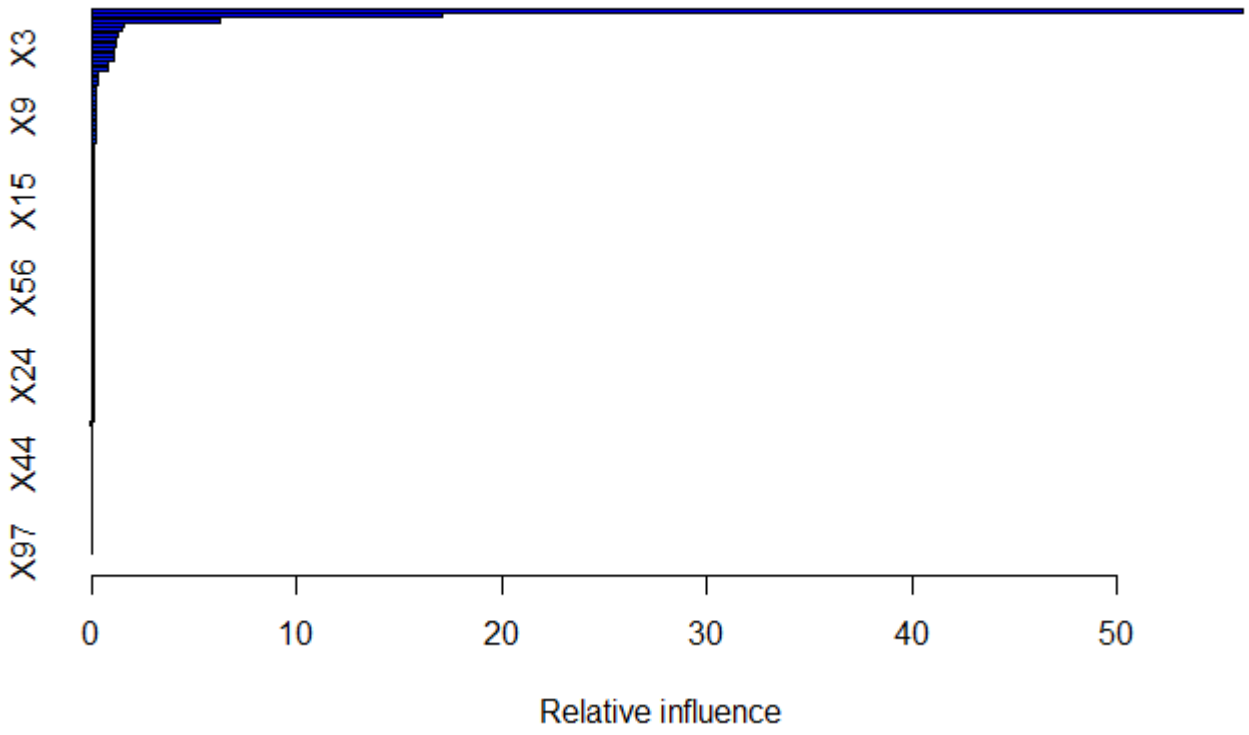
```
[1] 3.955602
```

Hide

```
# Generate prediction for competition test set for lasso model
x_test = model.matrix(d.test$id~., d.test)[,-1]
pred_lasso = predict(la, s = lambda_best, newx = x_test)


######################### Boosting With Regression Tress ##########################
y.test = d.train[-train,'y']

# Base boosting model with regression model
boost = gbm(y~.,data=d.train[train,], distribution='gaussian',n.trees = 5000,
            interaction.depth = 6, shrinkage = 0.01, cv.folds = 5)

# Model summary with order list of relevant predictor variables
summary(boost)
```

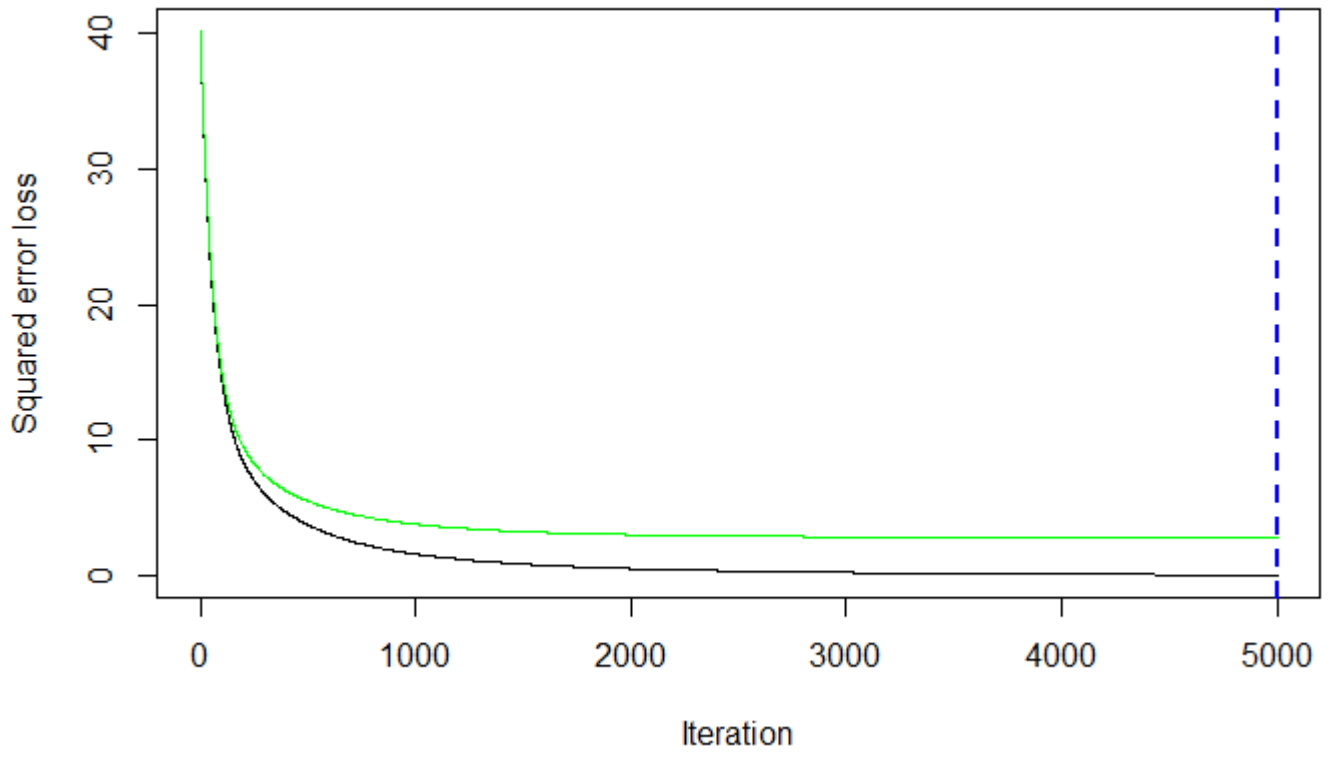
	var <chr>	**rel.inf** <dbl>
X4	X4	5.612926e+01
X86	X86	1.707047e+01
X11	X11	6.214578e+00
X14	X14	1.564293e+00
X1	X1	1.420125e+00
X55	X55	1.296003e+00
X54	X54	1.213356e+00
X79	X79	1.206617e+00
X3	X3	1.112777e+00
X43	X43	1.093313e+00

1-10 of 112 rows Previous **1** 2 3 4 5 6 … 12 Next



Hide

```
# Optimal number of tress for model chosen by cross validation
bi = gbm.perf(boost,method="cv")
```



Hide

```
# Test set prediction with optimal model
pr.boo = predict(boost,newdata=d.train[-train,],n.trees=bi)

# Rooted mean squared error
sqrt(mean((pr.boo - y.test)^2))
```

```
[1] 1.484779
```

Hide

```
# Generate prediction for competition test set for boosting model
pred_boosting = predict(boost,newdata=d.test,n.trees=bi)


# ######################## Generalized Additive Model ##############################

# Base GAM model with all variables
gam = gam(y~.,data=d.train[train,])

# Model Summary
summary(gam)
```

```
Call: gam(formula = y ~ ., data = d.train[train, ])
Deviance Residuals:
      Min        1Q    Median        3Q       Max
-12.61471  -2.46415  -0.03789   2.45628  12.17921

(Dispersion Parameter for gaussian family taken to be 14.4893)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 20096.67 on 1387 degrees of freedom
AIC: 8377.449

Number of Local Scoring Iterations: 2

Anova for Parametric Effects
            Df Sum Sq Mean Sq   F value    Pr(>F)
X1           1      1       1    0.0771  0.781316
X2           1      4       4    0.2565  0.612641
X3           1    298     298   20.5532 6.299e-06 ***
X4           1  33230   33230 2293.4436 < 2.2e-16 ***
X5           1    276     276   19.0740 1.351e-05 ***
X6           1   1658    1658  114.4345 < 2.2e-16 ***
X7           1    459     459   31.6539 2.226e-08 ***
X8           1    270     270   18.6391 1.692e-05 ***
X9           1    272     272   18.7894 1.565e-05 ***
X10          1      0       0    0.0322  0.857515
X11          1   1170    1170   80.7559 < 2.2e-16 ***
X12          1     33      33    2.2903  0.130413
X13          1      1       1    0.0735  0.786416
X14          1      7       7    0.4596  0.497915
X15          1      1       1    0.0522  0.819297
X16          1      5       5    0.3189  0.572389
X17          1      4       4    0.2644  0.607191
X18          1     13      13    0.8722  0.350514
X19          1     20      20    1.3825  0.239876
X20          1      3       3    0.1846  0.667504
X21          1     45      45    3.0955  0.078731 .
X22          1     14      14    1.0000  0.317475
X23          1     32      32    2.2217  0.136314
X24          1     31      31    2.1051  0.147037
X25          1     30      30    2.0409  0.153343
X26          1      6       6    0.4193  0.517388
X27          1     11      11    0.7305  0.392866
X28          1      2       2    0.1392  0.709153
X29          1      0       0    0.0006  0.979722
X30          1     39      39    2.7028  0.100399
X31          1      0       0    0.0105  0.918335
X32          1      5       5    0.3131  0.575859
X33          1     32      32    2.1865  0.139451
X34          1      0       0    0.0103  0.919200
X35          1     13      13    0.9316  0.334625
X36          1      1       1    0.0719  0.788581
```

```
X37          1     51      51    3.5061  0.061354 .
X38          1      5       5    0.3614  0.547841
X39          1     23      23    1.5749  0.209703
X40          1      3       3    0.2204  0.638773
X41          1      5       5    0.3263  0.567960
X42          1      2       2    0.1087  0.741657
X43          1    226     226   15.6017 8.211e-05 ***
X44          1      0       0    0.0084  0.927069
X45          1     22      22    1.5480  0.213641
X46          1     17      17    1.1795  0.277638
X47          1      1       1    0.0560  0.812970
X48          1      1       1    0.0473  0.827942
X49          1     12      12    0.8157  0.366587
X50          1     13      13    0.9086  0.340653
X51          1     10      10    0.7114  0.399131
X52          1      4       4    0.2850  0.593507
X53          1     11      11    0.7527  0.385771
X54          1    376     376   25.9237 4.040e-07 ***
X55          1      5       5    0.3373  0.561495
X56          1      9       9    0.5929  0.441420
X57          1      3       3    0.2340  0.628634
X58          1     16      16    1.0982  0.294839
X59          1    283     283   19.5281 1.068e-05 ***
X60          1     15      15    1.0185  0.313058
X61          1     19      19    1.3116  0.252301
X62          1      9       9    0.5975  0.439669
X63          1     24      24    1.6395  0.200602
X64          1      0       0    0.0029  0.957143
X65          1     20      20    1.4061  0.235914
X66          1      6       6    0.4476  0.503599
X67          1      7       7    0.4846  0.486444
X68          1      0       0    0.0228  0.879885
X69          1      0       0    0.0057  0.939789
X70          1      2       2    0.1488  0.699705
X71          1      1       1    0.0719  0.788635
X72          1      1       1    0.0784  0.779572
X73          1      0       0    0.0195  0.888903
X74          1      3       3    0.2333  0.629197
X75          1      1       1    0.0910  0.762990
X76          1      6       6    0.3971  0.528688
X77          1      2       2    0.1144  0.735268
X78          1     11      11    0.7605  0.383317
X79          1    496     496   34.2033 6.183e-09 ***
X80          1     49      49    3.3874  0.065910 .
X81          1      2       2    0.1587  0.690433
X82          1      0       0    0.0276  0.868007
X83          1     18      18    1.2122  0.271083
X84          1     29      29    1.9710  0.160561
X85          1      0       0    0.0000  0.996192
X86          1    538     538   37.1340 1.426e-09 ***
X87          1     20      20    1.4135  0.234672
X88          1     17      17    1.1911  0.275293
```

```
X89          1      9       9    0.6363  0.425188
X90          1     74      74    5.0900  0.024219 *
X91          1      3       3    0.1919  0.661416
X92          1      7       7    0.4599  0.497766
X93          1      0       0    0.0254  0.873370
X94          1     55      55    3.8100  0.051148 .
X95          1     35      35    2.4300  0.119263
X96          1      3       3    0.1940  0.659670
X97          1     16      16    1.0935  0.295875
X98          1      2       2    0.1266  0.722064
X99          1      7       7    0.5069  0.476590
X100         1     71      71    4.9241  0.026646 *
X101         1     24      24    1.6595  0.197891
X102         1      2       2    0.1284  0.720139
X103         1     37      37    2.5453  0.110849
X104         1      1       1    0.0559  0.813193
X105         1    116     116    7.9913  0.004768 **
X106         1      8       8    0.5204  0.470783
X107         1     27      27    1.8708  0.171604
X108         1     20      20    1.3978  0.237292
X109         1      0       0    0.0290  0.864806
X110         1     17      17    1.1850  0.276536
X111         1      1       1    0.0491  0.824735
X112         1      2       2    0.1340  0.714354
Residuals 1387  20097      14
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 3.660298
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 4.011662
```

Hide

```
# Used relevant variables used by lasso and boosting model above to construct
# GAM model

## Variable 1 ##
# Plot variable X4
plot(d.train[train,]$X4, d.train[train,]$y)

# Run cross validation to choose optimal degrees of freedom for smoothing spline
fit1 = smooth.spline(d.train[train,]$X4,d.train[train,]$y,cv=TRUE)
```

```
Warning in smooth.spline(d.train[train, ]$X4, d.train[train, ]$y, cv = TRUE) :
  cross-validation with non-unique 'x' values seems doubtful
```

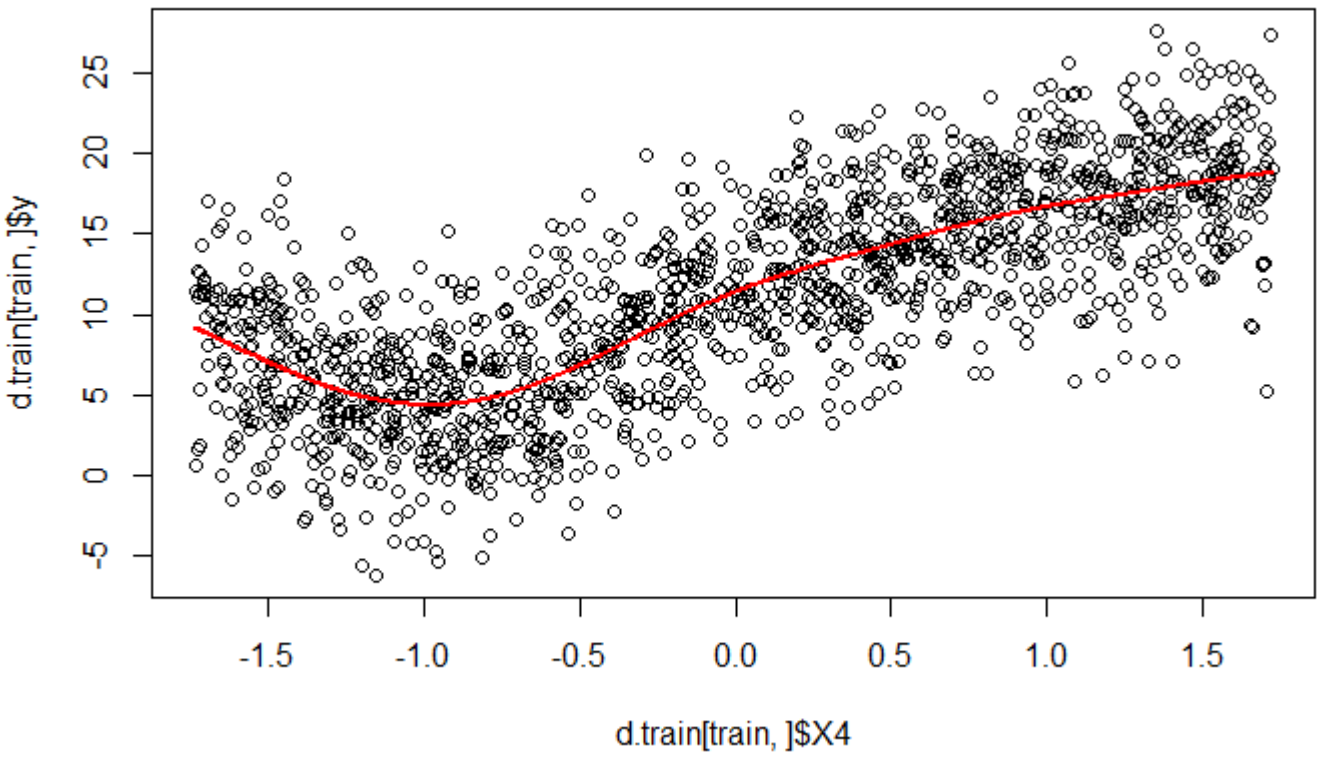
Hide

```
fit1
```

```
Call:
smooth.spline(x = d.train[train, ]$X4, y = d.train[train, ]$y,
    cv = TRUE)

Smoothing Parameter  spar= 0.9572574  lambda= 0.01271415 (16 iterations)
Equivalent Degrees of Freedom (Df): 7.538678
Penalized Criterion (RSS): 23220.49
PRESS(l.o.o. CV): 15.64342
```

Hide

```
lines(fit1 ,col ="red ",lwd =2)
```



Hide

```
# Add variable X4 to GAM model
gam = gam(y~s(X4,8),data=d.train[train,])

# Model Summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8), data = d.train[train, ])
Deviance Residuals:
      Min        1Q    Median        3Q       Max
-13.46493  -2.57616   0.09705   2.68150  11.84226

(Dispersion Parameter for gaussian family taken to be 15.5536)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 23190.34 on 1490.999 degrees of freedom
AIC: 8384.223

Number of Local Scoring Iterations: NA

Anova for Parametric Effects
            Df Sum Sq Mean Sq F value    Pr(>F)
s(X4, 8)     1  33000   33000  2121.7 < 2.2e-16 ***
Residuals 1491  23190      16
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
            Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)          7 44.292 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 3.931948
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 4.023214
```

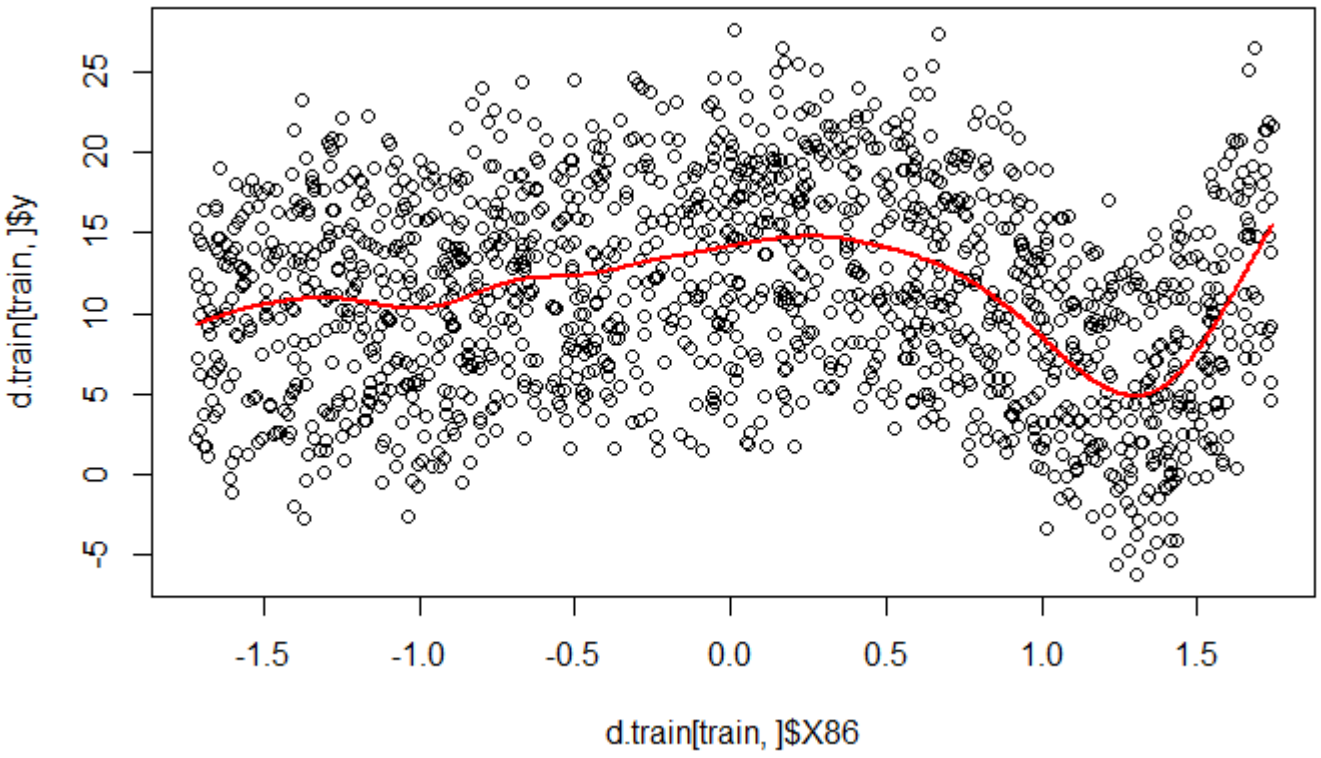
Hide

```
## Variable 2 ##
# Plot variable X86
plot(d.train[train,]$X86, d.train[train,]$y)

# Run cross validation to choose optimal degrees of freedom for smoothing spline
fit1 = smooth.spline(d.train[train,]$X86,d.train[train,]$y,cv=TRUE)
fit1
```

```
Call:
smooth.spline(x = d.train[train, ]$X86, y = d.train[train, ]$y,
    cv = TRUE)

Smoothing Parameter  spar= 0.8329657  lambda= 0.001490344 (16 iterations)
Equivalent Degrees of Freedom (Df): 12.18795
Penalized Criterion (RSS): 48902.88
PRESS(l.o.o. CV): 49.52124
```

Hide

```
lines(fit1 ,col ="red ",lwd =2)
```



Hide

```
# Add variable X86 to GAM model
gam = gam(y~s(X4,8)+s(X86,11),data=d.train[train,])

# Model Summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11), data = d.train[train,
    ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-7.87759 -2.15956  0.03121  1.75601 10.01525

(Dispersion Parameter for gaussian family taken to be 8.5471)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 12649.66 on 1479.999 degrees of freedom
AIC: 7497.065

Number of Local Scoring Iterations: NA

Anova for Parametric Effects
             Df Sum Sq Mean Sq F value    Pr(>F)
s(X4, 8)      1  31977   31977 3741.23 < 2.2e-16 ***
s(X86, 11)    1    486     486   56.83 8.239e-14 ***
Residuals  1480  12650       9
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
            Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)          7  79.228 < 2.2e-16 ***
s(X86, 11)       10 117.986 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.903987
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 3.014233
```

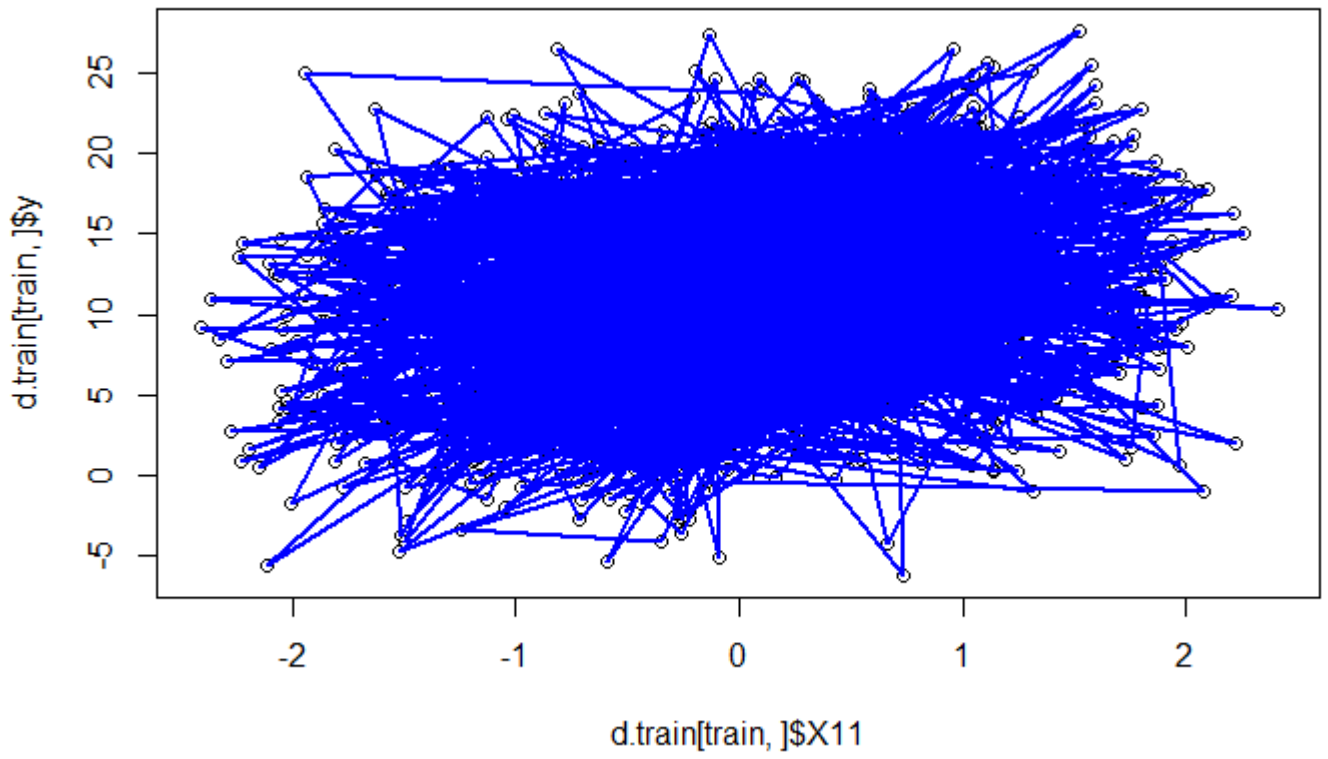
Hide

```
## Variable 3 ##
# Plot variable X11
plot(d.train[train,]$X11, d.train[train,]$y)

# Base local regression model
fit = loess(d.train[train,]$y ~ d.train[train,]$X11, data=d.train,span = 0.1)
fit
```

```
Call:
loess(formula = d.train[train, ]$y ~ d.train[train, ]$X11, data = d.train,
    span = 0.1)

Number of Observations: 1500
Equivalent Number of Parameters: 29.68
Residual Standard Error: 6.11
```

Hide

```
lines(fit ,col ="blue ",lwd =2)
```



Hide

```
# Choosing optimal span for model
span.seq <- seq(from = 0.1, to = 0.9, by = 0.1)
span = 0.1
testerror = 50000000000
for(i in 1:length(span.seq)) {
  gam = gam(y~s(X4,8)+s(X86,11)+lo(X11, span = span.seq[i]),data=d.train[train,])
  preds <- predict(gam, newdata = d.train[-train,],type="response")
  testerror_i = sqrt(mean((preds - y.test)^2))
  if (testerror_i<testerror){
    testerror = testerror_i
    span = span.seq[i]
  }
}
#span 0.1 selected
span
```

```
[1] 0.2
```

Hide

```
# Add variable X11 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1),
    data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-6.7971 -1.5388 -0.2961  1.3619  8.6102

(Dispersion Parameter for gaussian family taken to be 5.7535)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 8408.078 on 1461.374 degrees of freedom
AIC: 6921.659

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31484.3 31484.3 5472.158 < 2.2e-16 ***
s(X86, 11)            1.0   450.5   450.5   78.296 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3210.1  3210.1  557.931 < 2.2e-16 ***
Residuals          1461.4  8408.1     5.8
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 116.331 < 2.2e-16 ***
s(X86, 11)            10.0 176.732 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  10.544 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.36757
```

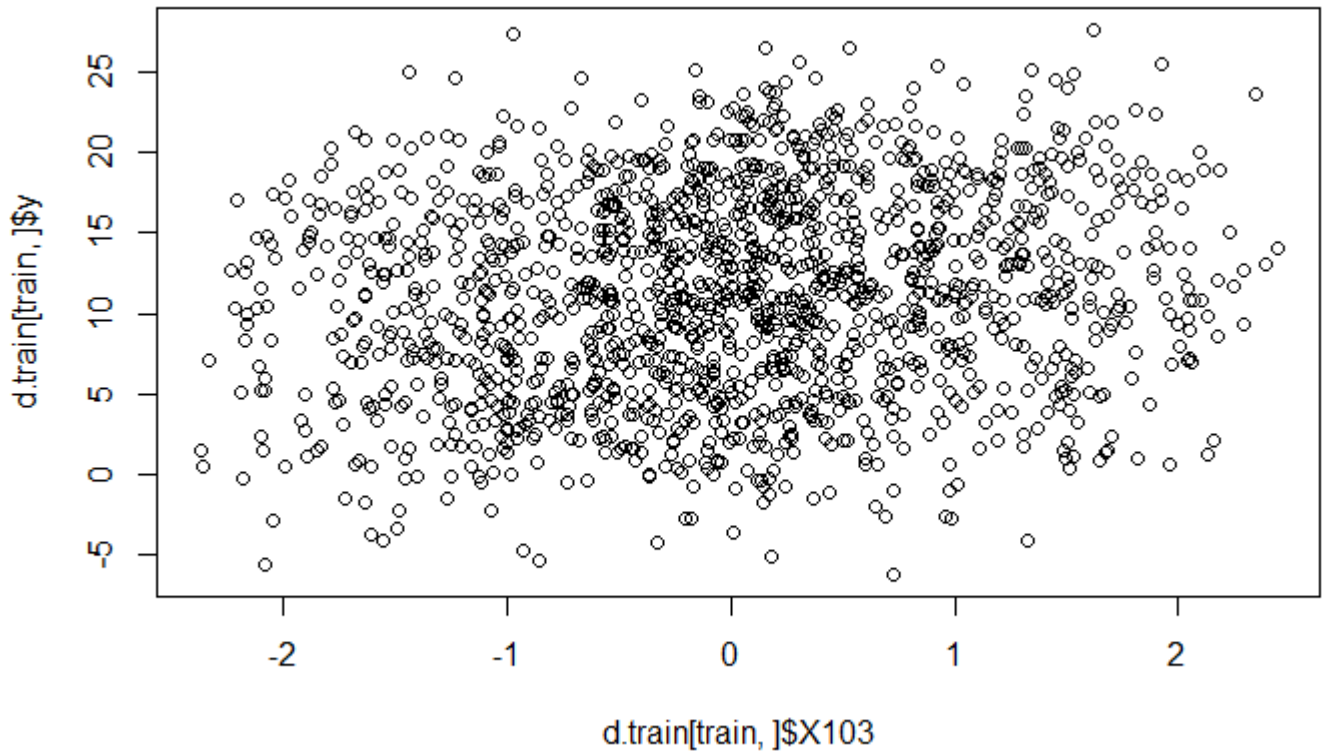
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.483988
```

Hide

```
## Variable 4 ##
# Plot variable X103
plot(d.train[train,]$X103, d.train[train,]$y)
```



Hide

```
# Add varibale X103 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103, data = d.train[train, ])
Deviance Residuals:
   Min     1Q Median     3Q    Max
-7.091 -1.501 -0.340  1.353  8.558

(Dispersion Parameter for gaussian family taken to be 5.6868)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 8304.884 on 1460.374 degrees of freedom
AIC: 6905.135

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31481.8 31481.8 5535.930 < 2.2e-16 ***
s(X86, 11)            1.0   450.5   450.5   79.224 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3206.0  3206.0  563.767 < 2.2e-16 ***
X103                  1.0   106.8   106.8   18.787 1.561e-05 ***
Residuals          1460.4  8304.9     5.7
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 116.926 < 2.2e-16 ***
s(X86, 11)            10.0 178.244 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  10.659 < 2.2e-16 ***
X103
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.352996
```

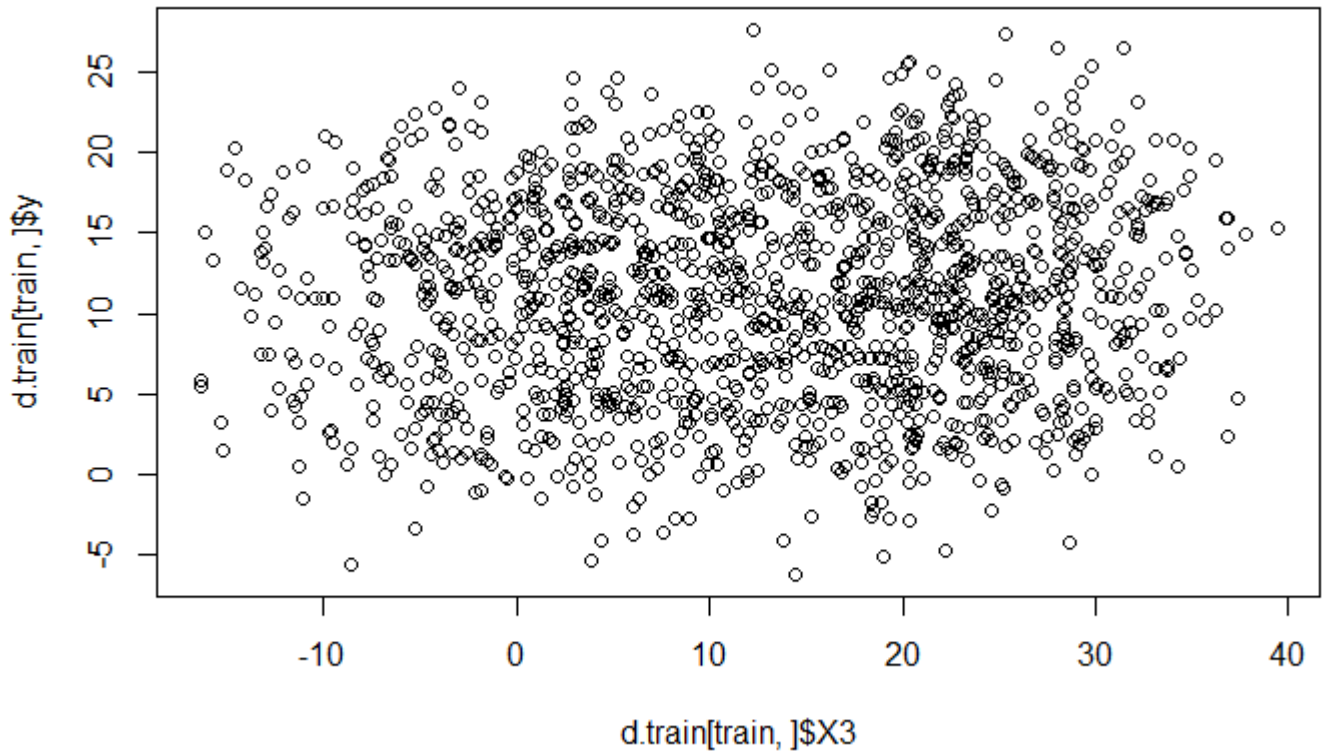
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.463704
```

Hide

```
## Variable 5 ##
# Plot variable X3
plot(d.train[train,]$X3, d.train[train,]$y)
```



Hide

```
# Add variable X3 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3, data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-6.4480 -1.4272 -0.2766  1.3080  7.9956

(Dispersion Parameter for gaussian family taken to be 5.3934)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 7870.979 on 1459.374 degrees of freedom
AIC: 6826.643

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31514.3 31514.3 5843.124 < 2.2e-16 ***
s(X86, 11)            1.0   453.5   453.5   84.091 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3204.1  3204.1  594.073 < 2.2e-16 ***
X103                  1.0   105.4   105.4   19.541 1.057e-05 ***
X3                    1.0   441.7   441.7   81.898 < 2.2e-16 ***
Residuals          1459.4  7871.0     5.4
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 122.232 < 2.2e-16 ***
s(X86, 11)            10.0 187.305 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  11.297 < 2.2e-16 ***
X103
X3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.290703
```

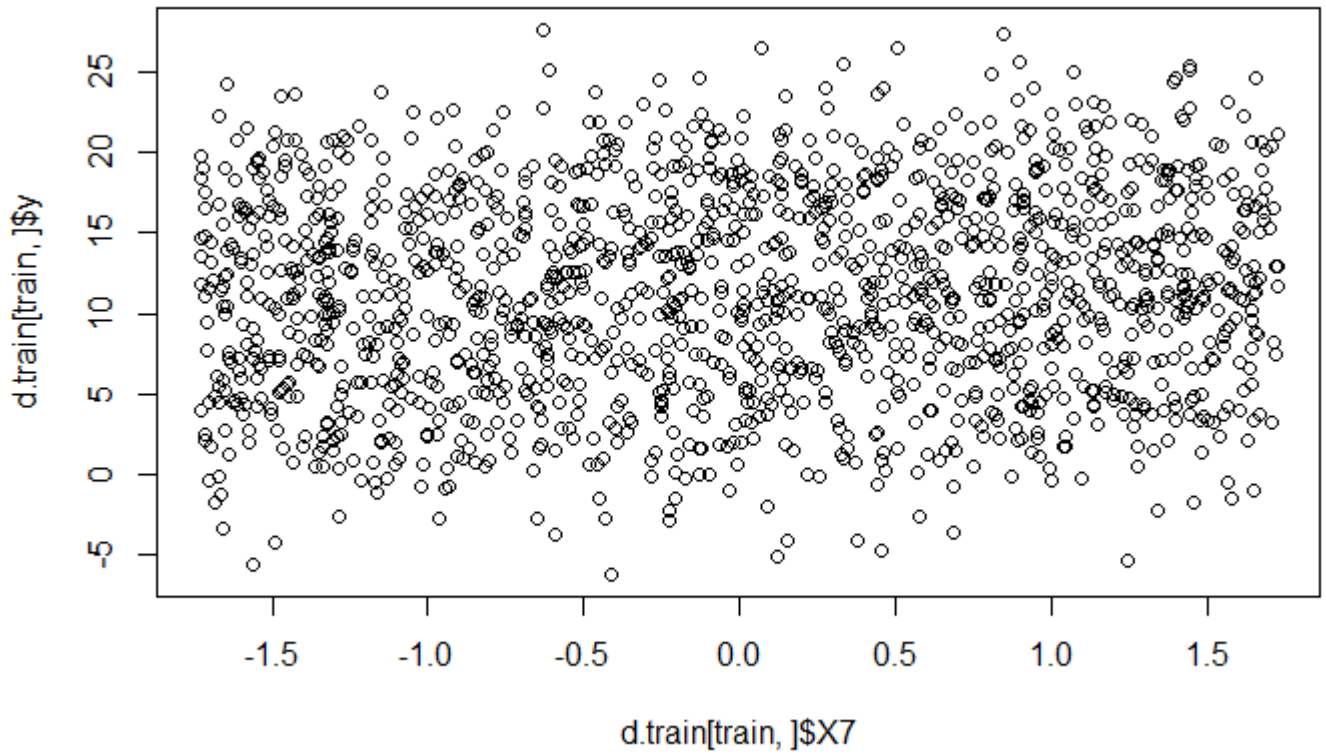
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.426869
```

Hide

```
## Variable 6 ##
# Plot variable X7
plot(d.train[train,]$X7, d.train[train,]$y)
```



Hide

```
# Add variable X7 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7, data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-6.7616 -1.4381 -0.2855  1.2527  7.7207

(Dispersion Parameter for gaussian family taken to be 5.1429)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 7500.24 on 1458.374 degrees of freedom
AIC: 6756.272

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31520.3 31520.3 6128.921 < 2.2e-16 ***
s(X86, 11)            1.0   451.3   451.3   87.753 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3200.5  3200.5  622.311 < 2.2e-16 ***
X103                  1.0   106.7   106.7   20.745 5.683e-06 ***
X3                    1.0   439.2   439.2   85.393 < 2.2e-16 ***
X7                    1.0   397.8   397.8   77.341 < 2.2e-16 ***
Residuals          1458.4  7500.2     5.1
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 127.791 < 2.2e-16 ***
s(X86, 11)            10.0 196.245 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  11.087 < 2.2e-16 ***
X103
X3
X7
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.236104
```

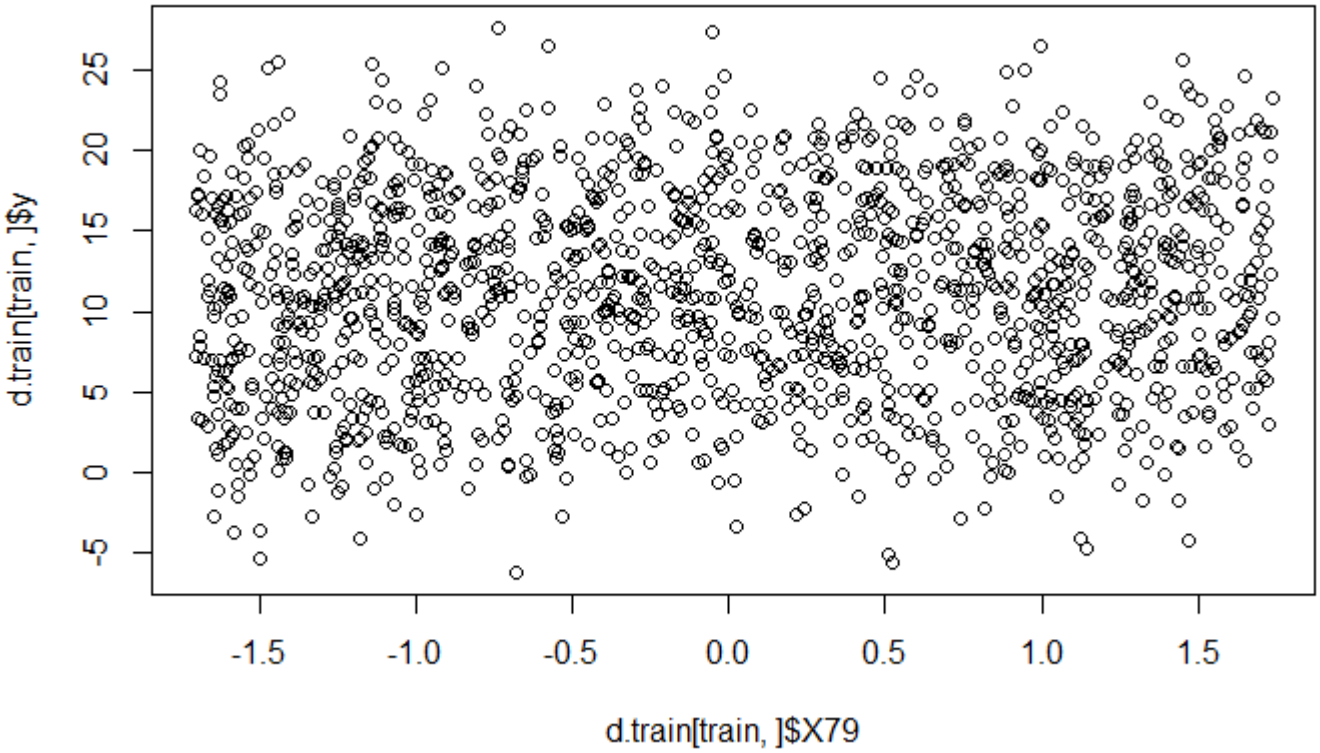
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.371482
```

Hide

```
## Variable 7 ##
# Plot variable X79
plot(d.train[train,]$X79, d.train[train,]$y)
```



Hide

```
# Add variable X79 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79, data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.9673 -1.4199 -0.2666  1.1973  8.4511

(Dispersion Parameter for gaussian family taken to be 4.8182)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 7021.988 on 1457.374 degrees of freedom
AIC: 6659.439

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31509.1 31509.1 6539.533 < 2.2e-16 ***
s(X86, 11)            1.0   449.9   449.9   93.364 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3198.2  3198.2  663.767 < 2.2e-16 ***
X103                  1.0   105.4   105.4   21.884 3.166e-06 ***
X3                    1.0   440.2   440.2   91.354 < 2.2e-16 ***
X7                    1.0   396.3   396.3   82.253 < 2.2e-16 ***
X79                   1.0   501.9   501.9  104.164 < 2.2e-16 ***
Residuals          1457.4  7022.0     4.8
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                    Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)                7.0 136.150 < 2.2e-16 ***
s(X86, 11)             10.0 207.684 < 2.2e-16 ***
lo(X11, span = 0.1)    17.6  12.248 < 2.2e-16 ***
X103
X3
X7
X79
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.163638
```

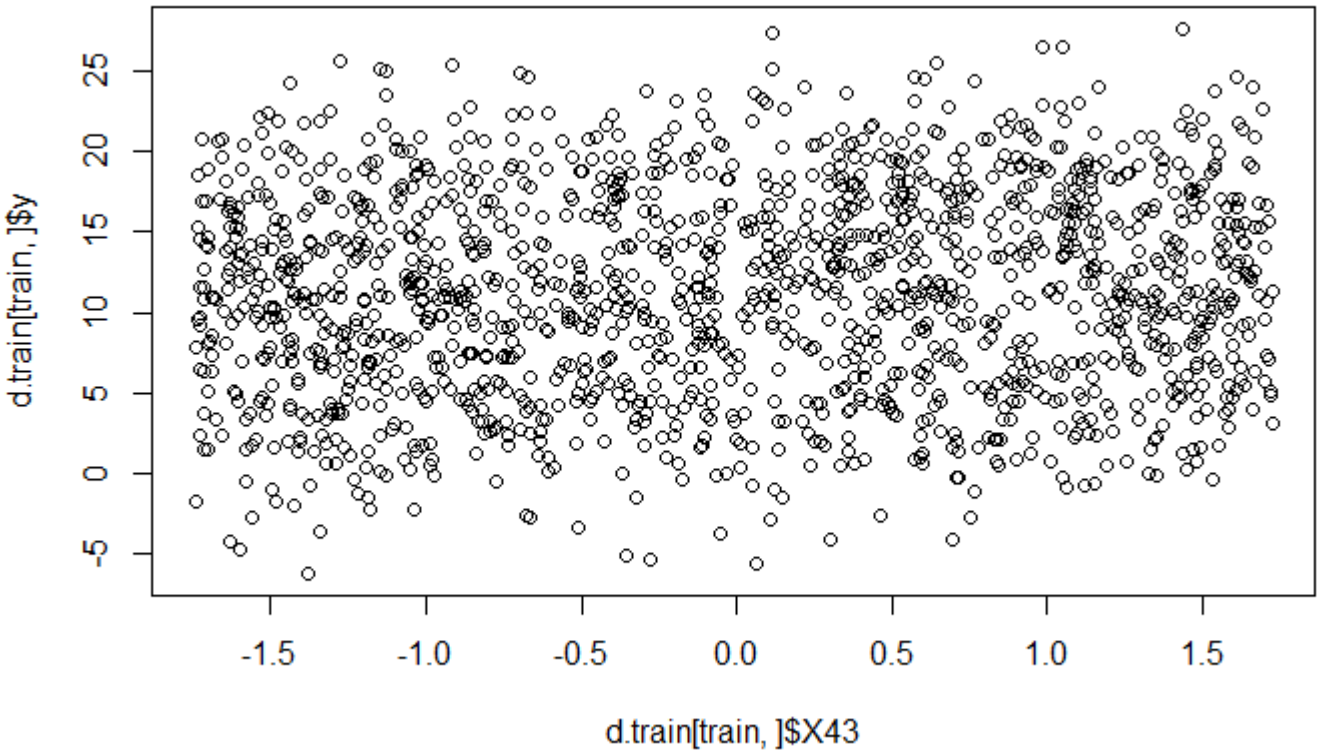
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.319451
```

Hide

```
## Variable 8 ##
# Plot variable X43
plot(d.train[train,]$X43, d.train[train,]$y)
```



Hide

```
# Add variable X43 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43, data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.5056 -1.4512 -0.3016  1.1106  8.7006

(Dispersion Parameter for gaussian family taken to be 4.5468)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 6621.845 on 1456.374 degrees of freedom
AIC: 6573.431

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq  F value    Pr(>F)
s(X4, 8)              1.0 31506.9 31506.9 6929.471 < 2.2e-16 ***
s(X86, 11)            1.0   446.9   446.9   98.299 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3198.0  3198.0  703.350 < 2.2e-16 ***
X103                  1.0   106.9   106.9   23.511 1.375e-06 ***
X3                    1.0   437.2   437.2   96.148 < 2.2e-16 ***
X7                    1.0   399.6   399.6   87.893 < 2.2e-16 ***
X79                   1.0   498.5   498.5  109.647 < 2.2e-16 ***
X43                   1.0   408.0   408.0   89.743 < 2.2e-16 ***
Residuals          1456.4  6621.8     4.5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                    Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)                7.0 146.333 < 2.2e-16 ***
s(X86, 11)             10.0 221.745 < 2.2e-16 ***
lo(X11, span = 0.1)    17.6  13.457 < 2.2e-16 ***
X103
X3
X7
X79
X43
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.101087
```

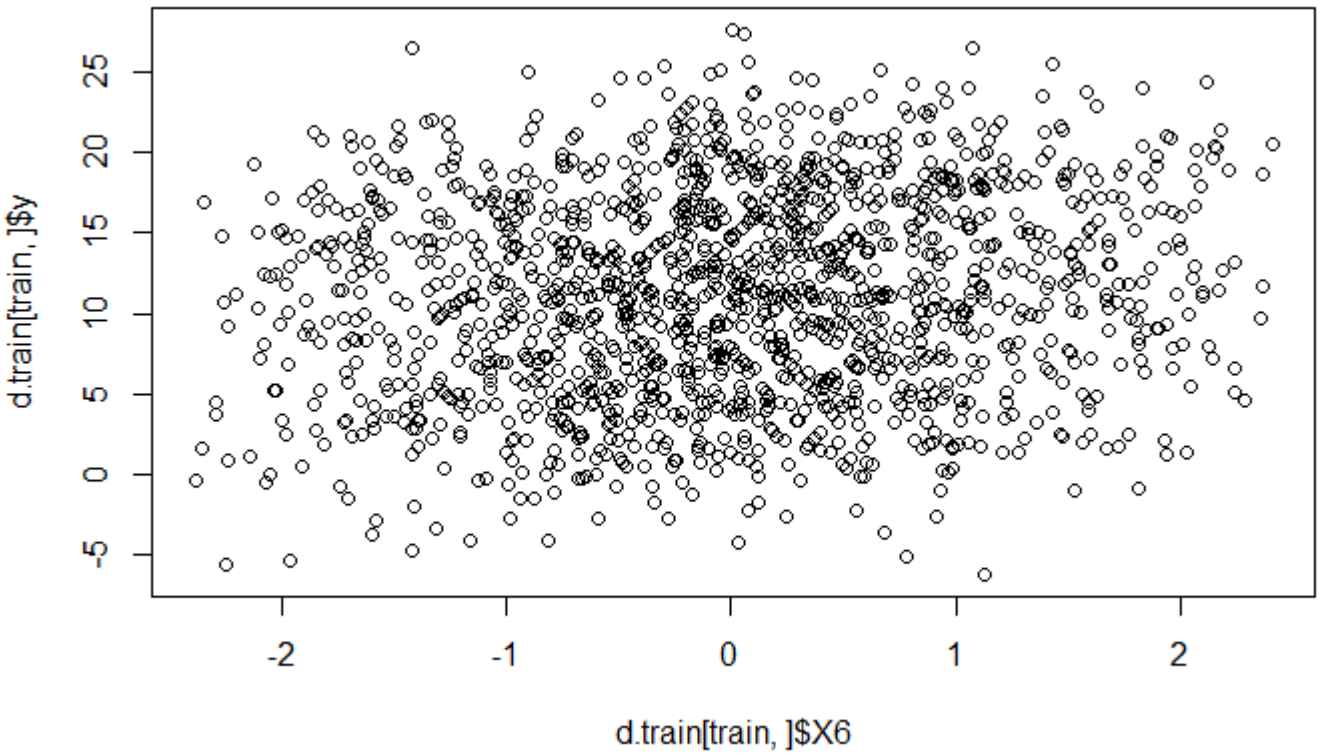
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.267492
```

Hide

```
## Variable 9 ##
# Plot variable X6
plot(d.train[train,]$X6, d.train[train,]$y)
```



Hide

```
# Add variable X6 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6, data = d.train[train, ])
Deviance Residuals:
   Min     1Q Median     3Q    Max
-5.458 -1.420 -0.303  1.101  8.488

(Dispersion Parameter for gaussian family taken to be 4.5265)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 6587.802 on 1455.374 degrees of freedom
AIC: 6567.699

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0 31508.4 31508.4 6960.8283 < 2.2e-16 ***
s(X86, 11)            1.0   448.4   448.4   99.0706 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3197.1  3197.1  706.2925 < 2.2e-16 ***
X103                  1.0   107.0   107.0   23.6299 1.294e-06 ***
X3                    1.0   437.2   437.2   96.5960 < 2.2e-16 ***
X7                    1.0   400.3   400.3   88.4451 < 2.2e-16 ***
X79                   1.0   499.2   499.2  110.2775 < 2.2e-16 ***
X43                   1.0   407.6   407.6   90.0374 < 2.2e-16 ***
X6                    1.0    40.0    40.0    8.8461  0.002986 **
Residuals          1455.4  6587.8     4.5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 146.32 < 2.2e-16 ***
s(X86, 11)            10.0 220.38 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  13.61 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.095679
```

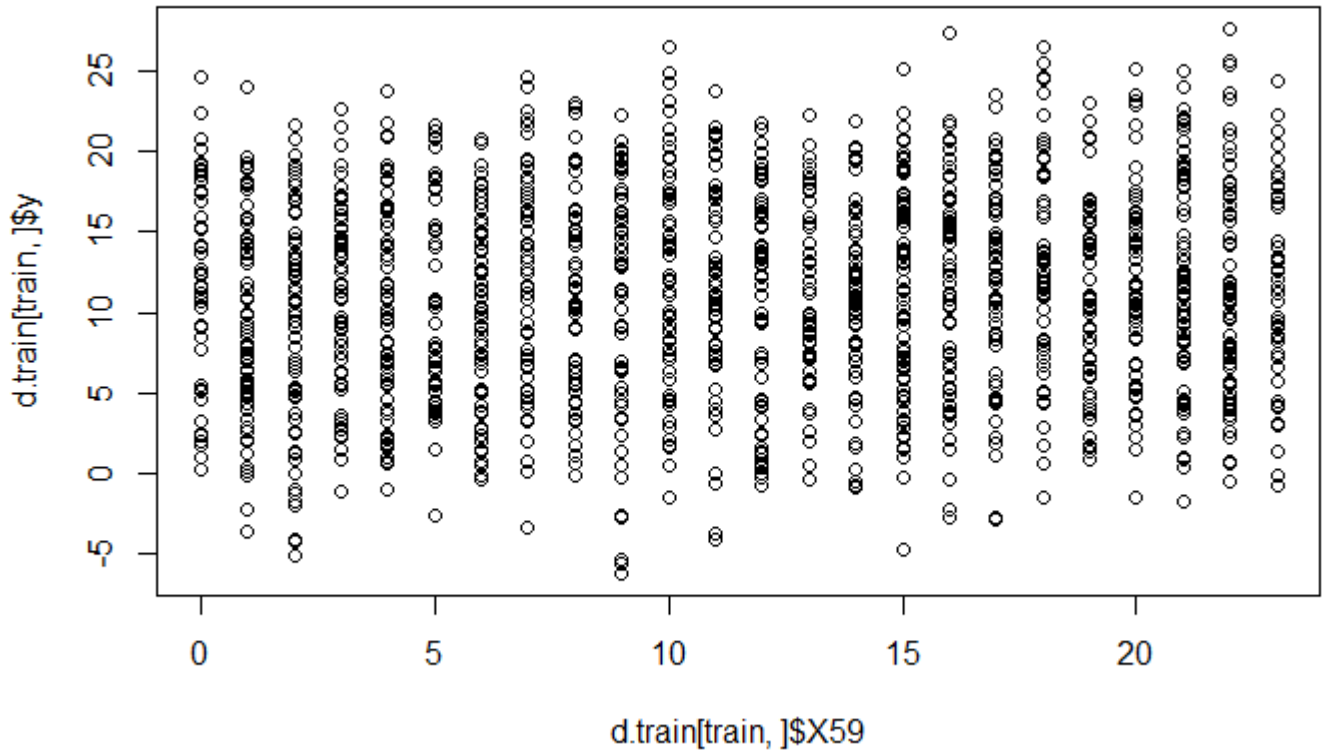
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.258397
```

Hide

```
## Variable 10 ##
# Plot variable X59
plot(d.train[train,]$X59, d.train[train,]$y)
```



Hide

```
# Add variable X59 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59,data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59, data = d.train[train,
    ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.4230 -1.3858 -0.2768  1.0349  8.9795

(Dispersion Parameter for gaussian family taken to be 4.3405)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 6312.678 on 1454.374 degrees of freedom
AIC: 6505.71

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0 31498.5 31498.5 7256.9283 < 2.2e-16 ***
s(X86, 11)            1.0   452.5   452.5  104.2399 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3194.7  3194.7  736.0175 < 2.2e-16 ***
X103                  1.0   107.4   107.4   24.7411 7.337e-07 ***
X3                    1.0   436.3   436.3  100.5178 < 2.2e-16 ***
X7                    1.0   400.2   400.2   92.2067 < 2.2e-16 ***
X79                   1.0   495.8   495.8  114.2222 < 2.2e-16 ***
X43                   1.0   403.9   403.9   93.0631 < 2.2e-16 ***
X6                    1.0    39.8    39.8    9.1583  0.002519 **
X59                   1.0   284.1   284.1   65.4437 1.249e-15 ***
Residuals          1454.4  6312.7     4.3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 149.850 < 2.2e-16 ***
s(X86, 11)            10.0 229.656 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  13.377 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.051452
```

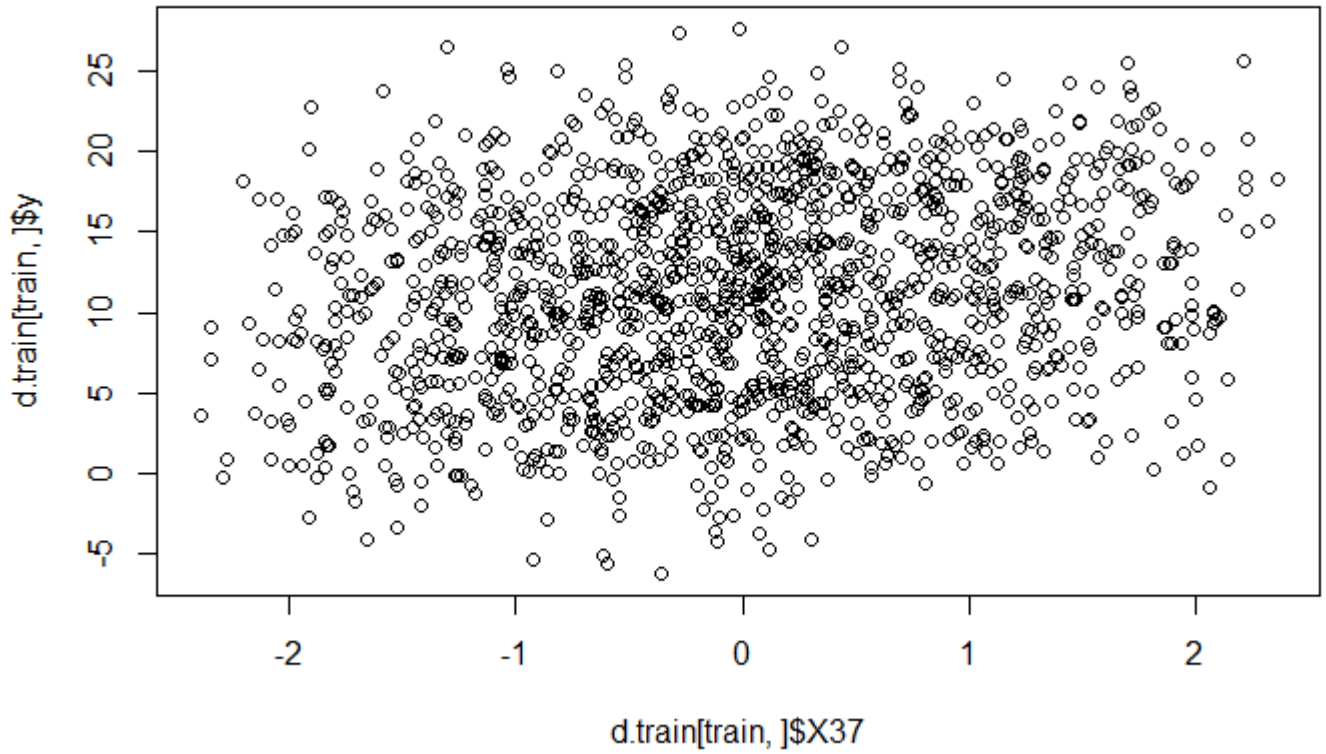
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.233451
```

Hide

```
## Variable 11 ##
# Plot variable X37
plot(d.train[train,]$X37, d.train[train,]$y)
```



Hide

```
# Add variable X37 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37,data=d.train[trai
n,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37, data = d.train[train,
    ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.2110 -1.3642 -0.2641  1.0276  8.8832

(Dispersion Parameter for gaussian family taken to be 4.3193)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 6277.53 on 1453.374 degrees of freedom
AIC: 6499.335

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0 31499.5 31499.5 7292.7591 < 2.2e-16 ***
s(X86, 11)            1.0   452.8   452.8  104.8382 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3195.7  3195.7  739.8642 < 2.2e-16 ***
X103                  1.0   106.9   106.9   24.7539 7.290e-07 ***
X3                    1.0   437.3   437.3  101.2384 < 2.2e-16 ***
X7                    1.0   400.5   400.5   92.7210 < 2.2e-16 ***
X79                   1.0   495.1   495.1  114.6174 < 2.2e-16 ***
X43                   1.0   403.8   403.8   93.4935 < 2.2e-16 ***
X6                    1.0    40.1    40.1    9.2861  0.002351 **
X59                   1.0   284.9   284.9   65.9488 9.781e-16 ***
X37                   1.0    34.2    34.2    7.9242  0.004943 **
Residuals          1453.4  6277.5     4.3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 150.225 < 2.2e-16 ***
s(X86, 11)            10.0 230.863 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  13.365 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
X37
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.045733
```

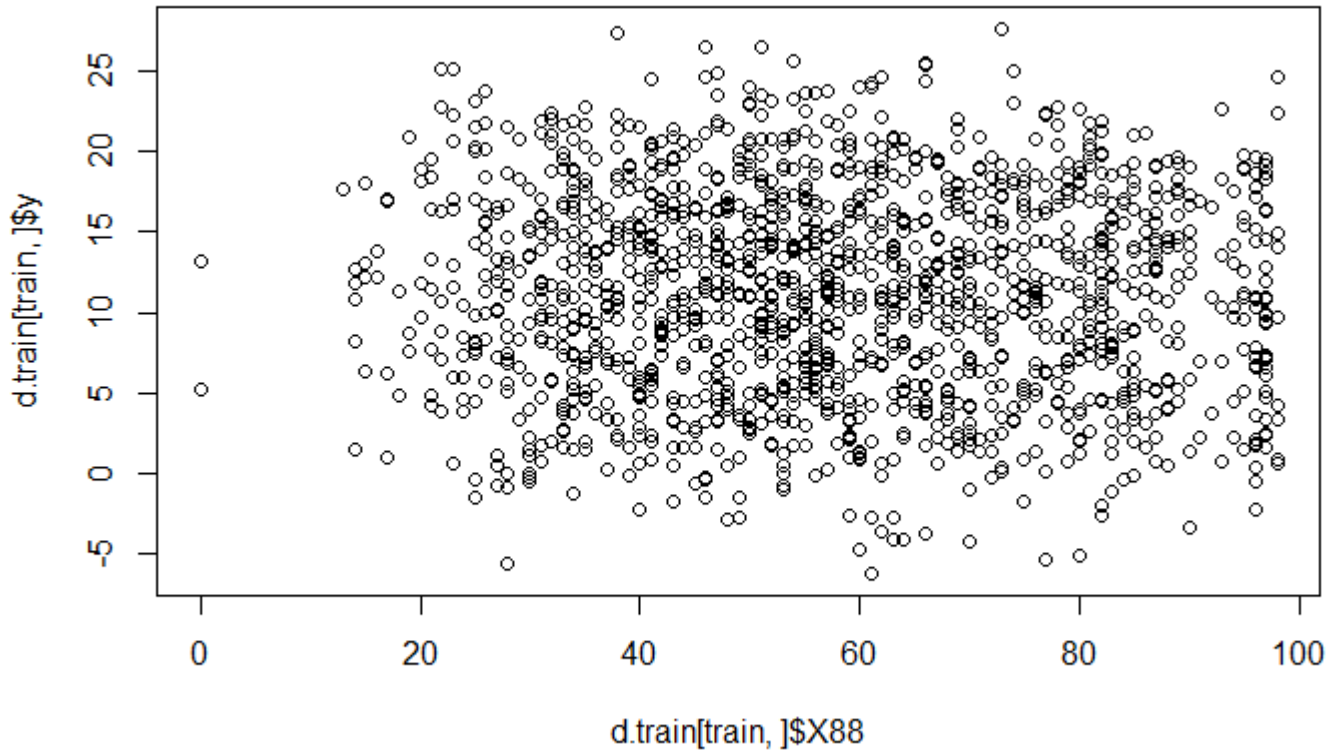
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.229146
```

Hide

```
## Variable 12 ##
# Plot variable X88
plot(d.train[train,]$X88, d.train[train,]$y)
```



Hide

```
# Choose optimal span for local regression
span.seq <- seq(from = 0.1, to = 0.9, by = 0.1)
span = 0.1
testerror = 50000000000
for(i in 1:length(span.seq)) {
  gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
            +lo(X88,span = span.seq[i]),data=d.train[train,])
  preds <- predict(gam, newdata = d.train[-train,],type="response")
  testerror_i = sqrt(mean((preds - y.test)^2))
  if (testerror_i<testerror){
    testerror = testerror_i
    span = span.seq[i]
  }
}
#span 0.4 selected
span
```

```
[1] 0.4
```

Hide

```
# Add variable X88 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4),
    data = d.train[train, ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.1707 -1.3352 -0.2824  0.9968  8.6869

(Dispersion Parameter for gaussian family taken to be 4.2252)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 6121.831 on 1448.873 degrees of freedom
AIC: 6470.663

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df  Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0 31534.2 31534.2 7463.2917 < 2.2e-16 ***
s(X86, 11)            1.0   446.7   446.7  105.7217 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0  3208.9  3208.9  759.4677 < 2.2e-16 ***
X103                  1.0   104.4   104.4   24.7172 7.430e-07 ***
X3                    1.0   424.4   424.4  100.4474 < 2.2e-16 ***
X7                    1.0   393.6   393.6   93.1538 < 2.2e-16 ***
X79                   1.0   491.5   491.5  116.3352 < 2.2e-16 ***
X43                   1.0   416.3   416.3   98.5370 < 2.2e-16 ***
X6                    1.0    38.3    38.3    9.0538  0.002667 **
X59                   1.0   295.4   295.4   69.9219 < 2.2e-16 ***
X37                   1.0    34.0    34.0    8.0368  0.004647 **
lo(X88, span = 0.4)   1.0    96.4    96.4   22.8226 1.957e-06 ***
Residuals          1448.9  6121.8     4.2
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 151.411 < 2.2e-16 ***
s(X86, 11)            10.0 237.077 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  13.638 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
X37
lo(X88, span = 0.4)    3.5   4.006  0.004797 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 2.020203
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 2.199507
```

Hide

```
# ############## Add Interaction Variables ##################

# GAM model with relevant predictors and interaction
gam = gam(y~(X1+X14+X55+X103+X4+X86+X11+X88+X103+X3+X7+X43+X59)^2,data=d.train[train,])

# Model summary to choose relevant interaction for predictions
summary(gam)
```

```
Call: gam(formula = y ~ (X1 + X14 + X55 + X103 + X4 + X86 + X11 + X88 +
    X103 + X3 + X7 + X43 + X59)^2, data = d.train[train, ])
Deviance Residuals:
      Min        1Q    Median        3Q       Max
-12.11869  -2.31501   0.04578   2.48134  10.99197

(Dispersion Parameter for gaussian family taken to be 12.801)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 18190.2 on 1421 degrees of freedom
AIC: 8159.942

Number of Local Scoring Iterations: 2

Anova for Parametric Effects
            Df Sum Sq Mean Sq   F value    Pr(>F)
X1           1      1       1    0.0873 0.7677300
X14          1     44      44    3.4485 0.0635160 .
X55          1     72      72    5.6054 0.0180386 *
X103         1   1398    1398  109.2469 < 2.2e-16 ***
X4           1  32987   32987 2576.9032 < 2.2e-16 ***
X86          1    547     547   42.7131 8.811e-11 ***
X11          1   1825    1825  142.5381 < 2.2e-16 ***
X88          1    172     172   13.4256 0.0002573 ***
X3           1    604     604   47.1940 9.588e-12 ***
X7           1    429     429   33.5409 8.578e-09 ***
X43          1    198     198   15.4748 8.765e-05 ***
X59          1    282     282   22.0221 2.955e-06 ***
X1:X14       1   1224    1224   95.6012 < 2.2e-16 ***
X1:X55       1    995     995   77.6993 < 2.2e-16 ***
X1:X103      1      2       2    0.1693 0.6807720
X1:X4        1      3       3    0.2407 0.6237689
X1:X86       1      8       8    0.6536 0.4189746
X1:X11       1      3       3    0.2724 0.6018410
X1:X88       1      7       7    0.5846 0.4446534
X1:X3        1     82      82    6.4362 0.0112878 *
X1:X7        1      2       2    0.1195 0.7296586
X1:X43       1     16      16    1.2835 0.2574343
X1:X59       1     33      33    2.5696 0.1091597
X14:X55      1   1031    1031   80.5466 < 2.2e-16 ***
X14:X103     1      0       0    0.0006 0.9801197
X14:X4       1      0       0    0.0252 0.8738237
X14:X86      1      0       0    0.0323 0.8574528
X14:X11      1     16      16    1.2768 0.2586768
X14:X88      1      2       2    0.1725 0.6779640
X14:X3       1      1       1    0.0708 0.7902048
X14:X7       1      2       2    0.1644 0.6852058
X14:X43      1     20      20    1.5294 0.2164012
X14:X59      1     37      37    2.9039 0.0885857 .
X55:X103     1     59      59    4.6023 0.0320984 *
X55:X4       1     28      28    2.1879 0.1393162
```

```
X55:X86       1      5       5    0.4168 0.5186268
X55:X11       1      5       5    0.3877 0.5336134
X55:X88       1      2       2    0.1281 0.7204681
X55:X3        1     15      15    1.2064 0.2722242
X55:X7        1      0       0    0.0078 0.9297073
X55:X43       1      1       1    0.0392 0.8431444
X55:X59       1      1       1    0.0829 0.7734362
X103:X4       1      7       7    0.5838 0.4449429
X103:X86      1      8       8    0.6595 0.4168755
X103:X11      1     13      13    1.0182 0.3131275
X103:X88      1     13      13    0.9769 0.3231295
X103:X3       1     14      14    1.0702 0.3010853
X103:X7       1     41      41    3.2370 0.0722062 .
X103:X43      1     24      24    1.8905 0.1693636
X103:X59      1     45      45    3.5213 0.0607905 .
X4:X86        1      8       8    0.6003 0.4385803
X4:X11        1     31      31    2.3943 0.1220009
X4:X88        1     22      22    1.7161 0.1904071
X4:X3         1     13      13    0.9928 0.3192179
X4:X7         1      0       0    0.0048 0.9450037
X4:X43        1      8       8    0.6459 0.4217006
X4:X59        1      4       4    0.2849 0.5935811
X86:X11       1     36      36    2.8294 0.0927731 .
X86:X88       1     12      12    0.9191 0.3378755
X86:X3        1     49      49    3.8217 0.0507883 .
X86:X7        1      5       5    0.4081 0.5230469
X86:X43       1     10      10    0.8020 0.3706564
X86:X59       1      2       2    0.1550 0.6938936
X11:X88       1     20      20    1.5724 0.2100657
X11:X3        1      8       8    0.6631 0.4156089
X11:X7        1      0       0    0.0000 0.9990135
X11:X43       1      4       4    0.3267 0.5677070
X11:X59       1     25      25    1.9495 0.1628557
X88:X3        1     67      67    5.2450 0.0221555 *
X88:X7        1      9       9    0.6778 0.4104952
X88:X43       1      3       3    0.2656 0.6063569
X88:X59       1     22      22    1.6841 0.1945910
X3:X7         1     24      24    1.8618 0.1726293
X3:X43        1      3       3    0.2594 0.6106343
X3:X59        1     68      68    5.2885 0.0216116 *
X7:X43        1     20      20    1.5996 0.2061611
X7:X59        1     19      19    1.4972 0.2212982
X43:X59       1      3       3    0.2644 0.6071975
Residuals  1421  18190      13
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
## Variable 13 ##
# Plot interaction between variables X14, X1
plot(d.train[train,]$X14*d.train[train,]$X1, d.train[train,]$y)

# Run cross validation to choose optimal degrees of freedom for smoothing spline
fit1 = smooth.spline(d.train[train,]$X14*d.train[train,]$X1,d.train[train,]$y,cv=TRUE)
```

```
Warning in smooth.spline(d.train[train, ]$X14 * d.train[train, ]$X1, d.train[train,  :
  cross-validation with non-unique 'x' values seems doubtful
```

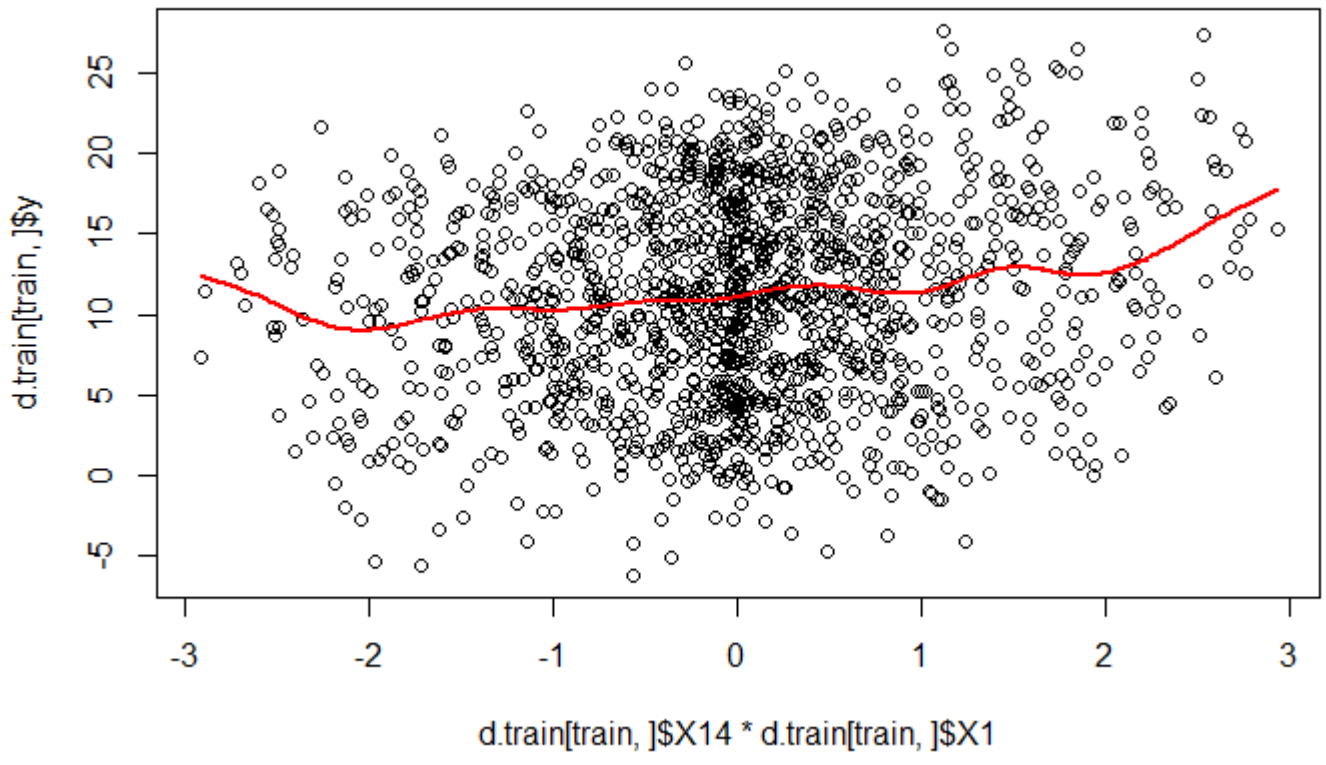
Hide

```
fit1
```

```
Call:
smooth.spline(x = d.train[train, ]$X14 * d.train[train, ]$X1,
    y = d.train[train, ]$y, cv = TRUE)

Smoothing Parameter  spar= 1.008908  lambda= 0.001436792 (12 iterations)
Equivalent Degrees of Freedom (Df): 11.2561
Penalized Criterion (RSS): 59165.14
PRESS(l.o.o. CV): 40.01125
```

Hide

```
lines(fit1 ,col ="red ",lwd =2)
```



Hide

```
# Add interaction between variables X14, X1 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4)+s(X14*X1,11),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4) +
    s(X14 * X1, 11), data = d.train[train, ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-5.87574 -1.19093 -0.08674  1.09384  6.20347

(Dispersion Parameter for gaussian family taken to be 3.3108)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 4760.531 on 1437.875 degrees of freedom
AIC: 6115.407

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0  31765   31765 9594.2227 < 2.2e-16 ***
s(X86, 11)            1.0    447     447  135.1249 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0   3236    3236  977.5421 < 2.2e-16 ***
X103                  1.0    104     104   31.4485 2.452e-08 ***
X3                    1.0    429     429  129.7035 < 2.2e-16 ***
X7                    1.0    395     395  119.1833 < 2.2e-16 ***
X79                   1.0    486     486  146.7721 < 2.2e-16 ***
X43                   1.0    420     420  126.8018 < 2.2e-16 ***
X6                    1.0     46      46   13.8960 0.0002007 ***
X59                   1.0    299     299   90.1877 < 2.2e-16 ***
X37                   1.0     31      31    9.4305 0.0021740 **
lo(X88, span = 0.4)   1.0     85      85   25.7191 4.463e-07 ***
s(X14 * X1, 11)       1.0   1110    1110  335.1973 < 2.2e-16 ***
Residuals          1437.9   4761       3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df  Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 198.056 < 2.2e-16 ***
s(X86, 11)            10.0 290.232 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  18.760 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
X37
lo(X88, span = 0.4)    3.5   3.427   0.01183 *
s(X14 * X1, 11)       10.0   8.877 2.798e-14 ***
```

```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 1.781485
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.944958
```

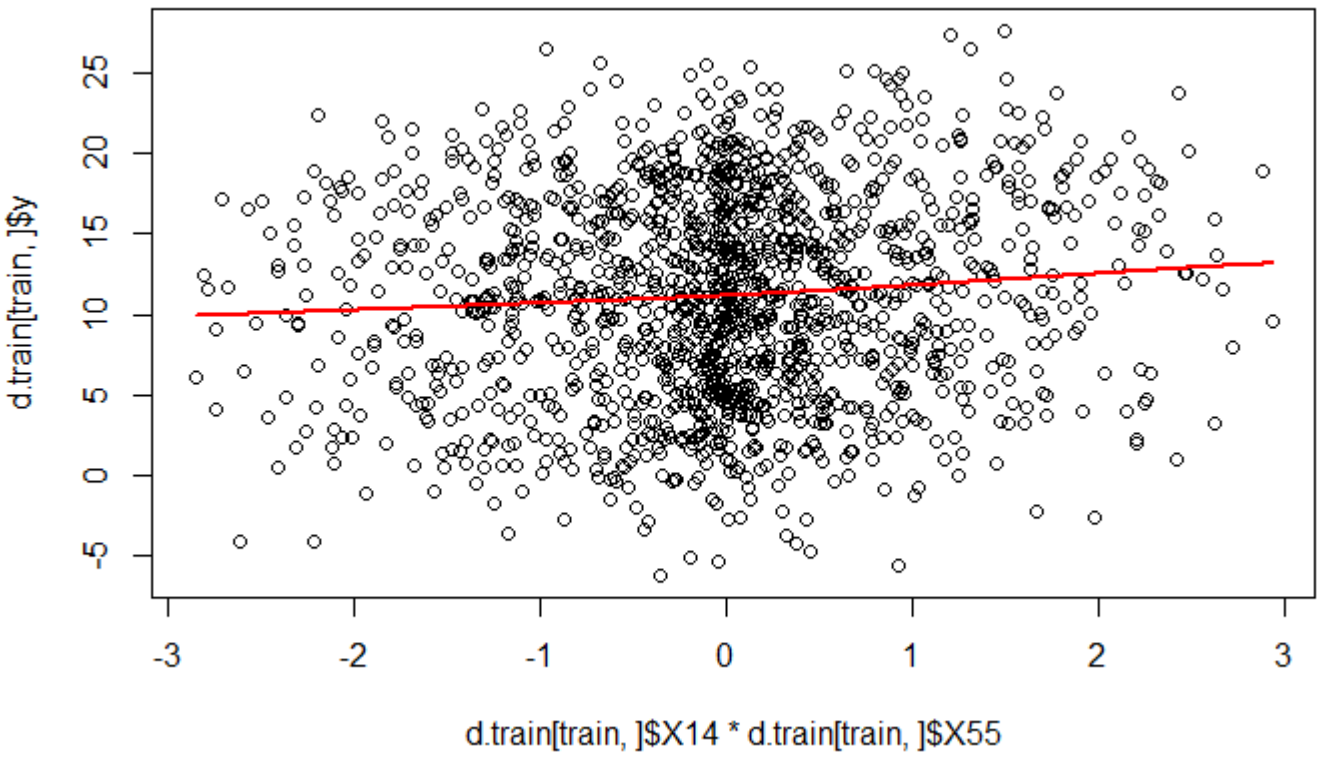
Hide

```
## Variable 14 ##
# Plot interaction between variables X14, X55
plot(d.train[train,]$X14*d.train[train,]$X55, d.train[train,]$y)

# Run cross validation to choose optimal degrees of freedom for smoothing spline
fit1 = smooth.spline(d.train[train,]$X14*d.train[train,]$X55,d.train[train,]$y,cv=TRUE)
fit1
```

```
Call:
smooth.spline(x = d.train[train, ]$X14 * d.train[train, ]$X55,
    y = d.train[train, ]$y, cv = TRUE)

Smoothing Parameter  spar= 1.457796  lambda= 2.168069 (15 iterations)
Equivalent Degrees of Freedom (Df): 2.495034
Penalized Criterion (RSS): 60516.96
PRESS(l.o.o. CV): 40.70243
```

Hide

```
lines(fit1 ,col ="red ",lwd =2)
```



Hide

```
# Add interaction between variables X14, X55 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4) +
    s(X14 * X1, 11) + s(X14 * X55, 2), data = d.train[train,
    ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-5.0713 -1.0128 -0.0879  0.9628  6.2167

(Dispersion Parameter for gaussian family taken to be 2.6082)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 3745.097 on 1435.875 degrees of freedom
AIC: 5759.539

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0  31825   31825 12201.763 < 2.2e-16 ***
s(X86, 11)            1.0    440     440   168.789 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0   3192    3192  1223.708 < 2.2e-16 ***
X103                  1.0    103     103    39.619 4.092e-10 ***
X3                    1.0    427     427   163.534 < 2.2e-16 ***
X7                    1.0    406     406   155.608 < 2.2e-16 ***
X79                   1.0    479     479   183.823 < 2.2e-16 ***
X43                   1.0    413     413   158.504 < 2.2e-16 ***
X6                    1.0     46      46    17.518 3.019e-05 ***
X59                   1.0    310     310   118.673 < 2.2e-16 ***
X37                   1.0     32      32    12.222 0.0004867 ***
lo(X88, span = 0.4)   1.0     82      82    31.471 2.425e-08 ***
s(X14 * X1, 11)       1.0   1090    1090   418.002 < 2.2e-16 ***
s(X14 * X55, 2)       1.0   1026    1026   393.401 < 2.2e-16 ***
Residuals          1435.9   3745       3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                    Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)                7.0 244.55 < 2.2e-16 ***
s(X86, 11)             10.0 371.58 < 2.2e-16 ***
lo(X11, span = 0.1)    17.6  23.34 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
X37
lo(X88, span = 0.4)     3.5   3.99  0.004953 **
```

```
s(X14 * X1, 11)        10.0  10.63 < 2.2e-16 ***
s(X14 * X55, 2)         1.0  17.59 2.918e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 1.580105
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.683563
```

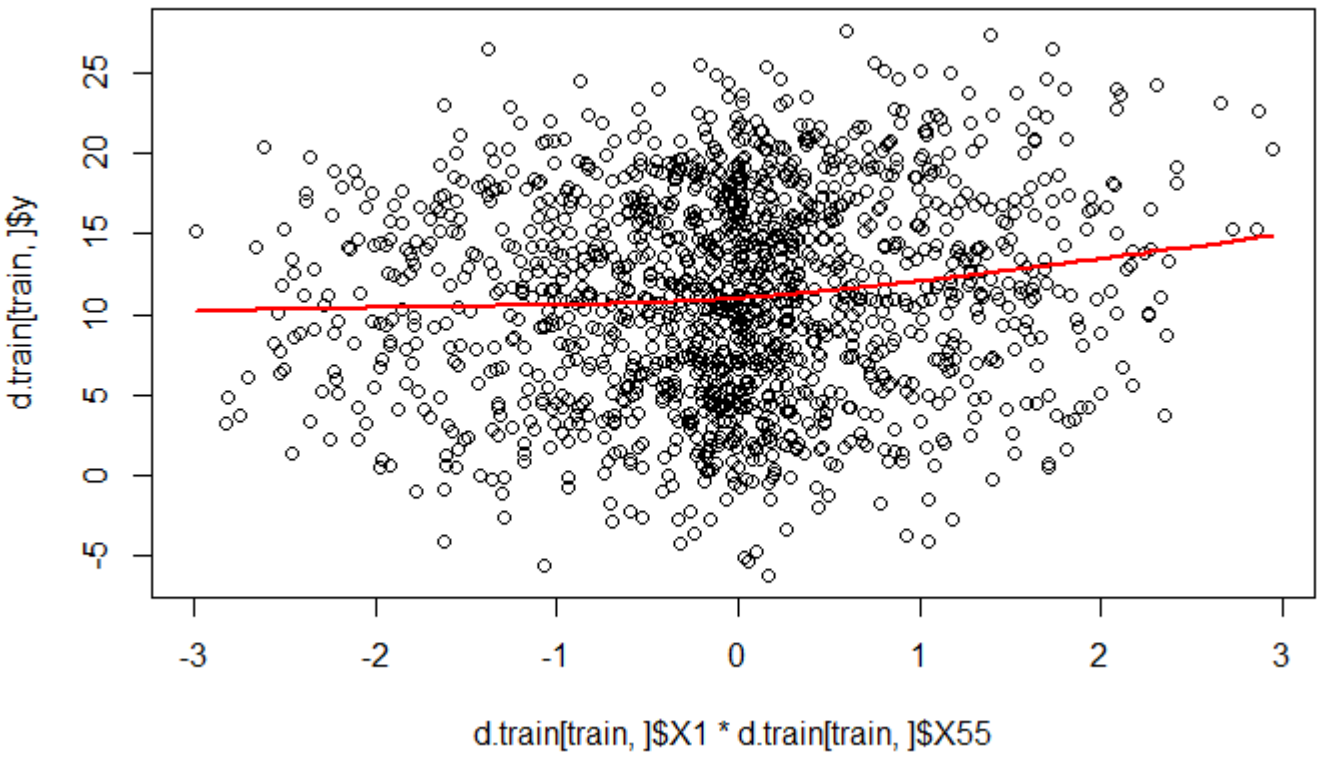
Hide

```
#Variable 15
# Plot interaction between variables X1, X55
plot(d.train[train,]$X1*d.train[train,]$X55, d.train[train,]$y)

# Run cross validation to choose optimal degrees of freedom for smoothing spline
fit1 = smooth.spline(d.train[train,]$X1*d.train[train,]$X55,d.train[train,]$y,cv=TRUE)
fit1
```

```
Call:
smooth.spline(x = d.train[train, ]$X1 * d.train[train, ]$X55,
    y = d.train[train, ]$y, cv = TRUE)

Smoothing Parameter  spar= 1.356127  lambda= 0.4005487 (15 iterations)
Equivalent Degrees of Freedom (Df): 3.267869
Penalized Criterion (RSS): 59980.88
PRESS(l.o.o. CV): 40.90707
```

Hide

```
lines(fit1 ,col ="red ",lwd =2)
```



Hide

```
# Add interaction between variables X1, X55 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2)+s(X1*X55,2),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4) +
    s(X14 * X1, 11) + s(X14 * X55, 2) + s(X1 * X55, 2), data = d.train[train,
    ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-4.35134 -0.84018 -0.02136  0.78014  5.32010

(Dispersion Parameter for gaussian family taken to be 1.8111)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 2596.957 on 1433.875 degrees of freedom
AIC: 5214.379

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                       Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)              1.0  31941   31941 17635.897 < 2.2e-16 ***
s(X86, 11)            1.0    435     435   240.232 < 2.2e-16 ***
lo(X11, span = 0.1)   1.0   3167    3167  1748.519 < 2.2e-16 ***
X103                  1.0    102     102    56.552 9.605e-14 ***
X3                    1.0    433     433   238.807 < 2.2e-16 ***
X7                    1.0    418     418   231.062 < 2.2e-16 ***
X79                   1.0    486     486   268.258 < 2.2e-16 ***
X43                   1.0    416     416   229.735 < 2.2e-16 ***
X6                    1.0     46      46    25.570 4.817e-07 ***
X59                   1.0    311     311   171.528 < 2.2e-16 ***
X37                   1.0     28      28    15.446 8.894e-05 ***
lo(X88, span = 0.4)   1.0     78      78    43.007 7.594e-11 ***
s(X14 * X1, 11)       1.0   1089    1089   601.078 < 2.2e-16 ***
s(X14 * X55, 2)       1.0   1034    1034   570.690 < 2.2e-16 ***
s(X1 * X55, 2)        1.0   1095    1095   604.582 < 2.2e-16 ***
Residuals          1433.9   2597       2
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                   Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)               7.0 358.44 < 2.2e-16 ***
s(X86, 11)            10.0 527.82 < 2.2e-16 ***
lo(X11, span = 0.1)   17.6  34.70 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
X37
```

```
lo(X88, span = 0.4)     3.5   5.00 0.0009942 ***
s(X14 * X1, 11)        10.0  14.08 < 2.2e-16 ***
s(X14 * X55, 2)         1.0  32.83 1.230e-08 ***
s(X1 * X55, 2)          1.0  49.74 2.695e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 1.315791
```

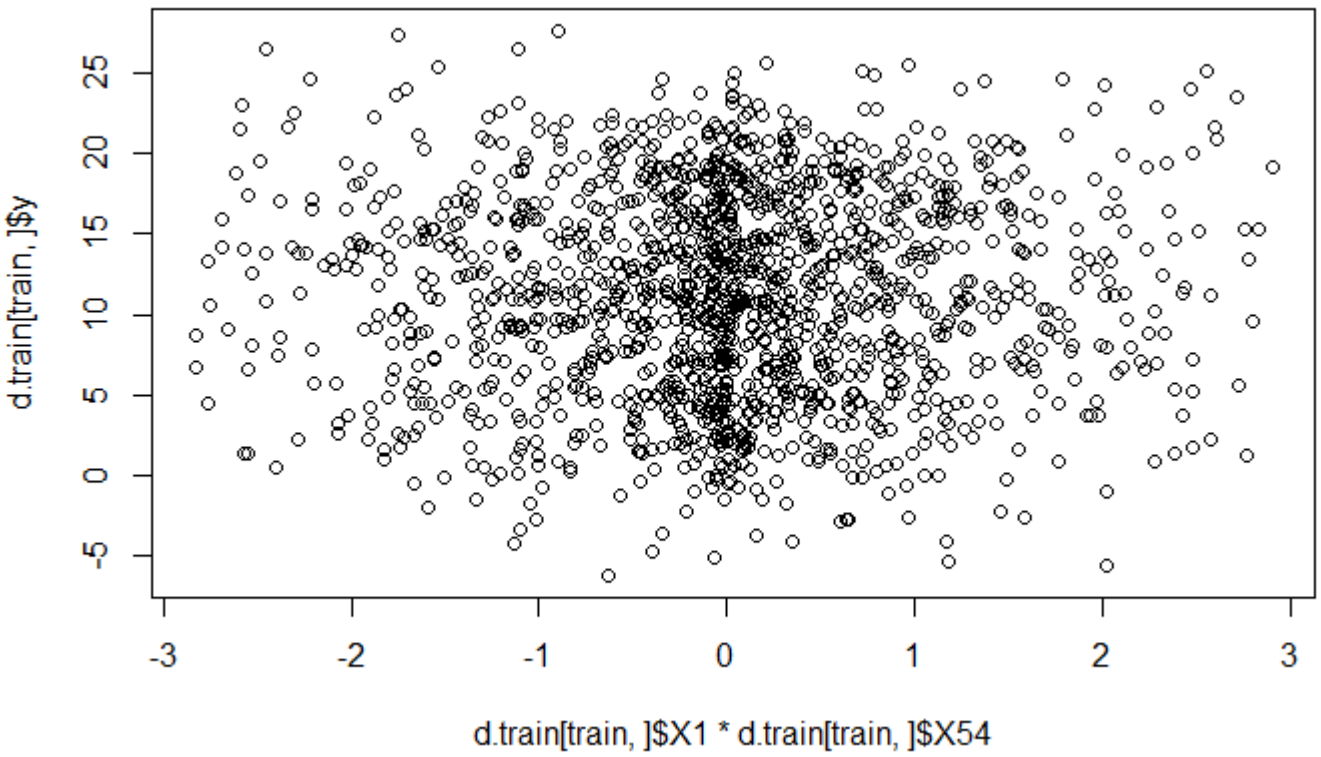
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.380376
```

Hide

```
## Variable 16 ##
# Plot interaction between variables X1, X54
plot(d.train[train,]$X1*d.train[train,]$X54, d.train[train,]$y)
```



Hide

```
# Choose optimal span for local regression
span.seq <- seq(from = 0.1, to = 0.9, by = 0.1)
span = 0.1
testerror = 50000000000
for(i in 1:length(span.seq)) {
  gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
            +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2)+s(X1*X55,2)
            +lo(X54,X1,span=span.seq[i]),data=d.train[train,])
  preds <- predict(gam, newdata = d.train[-train,],type="response")
  testerror_i = sqrt(mean((preds - y.test)^2))
  if (testerror_i<testerror){
    testerror = testerror_i
    span = span.seq[i]
  }
}
#span 0.1 selected
span
```

```
[1] 0.7
```

Hide

```
# Add interaction between variables X1, X54 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2)+s(X1*X55,2)
          +lo(X54,X1,span=0.1),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4) +
    s(X14 * X1, 11) + s(X14 * X55, 2) + s(X1 * X55, 2) + lo(X54,
    X1, span = 0.1), data = d.train[train, ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-3.79218 -0.76650 -0.04983  0.73802  4.57103

(Dispersion Parameter for gaussian family taken to be 1.5134)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 2114.411 on 1397.167 degrees of freedom
AIC: 4979.448

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                           Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)                  1.0  32003   32003 21147.307 < 2.2e-16 ***
s(X86, 11)                1.0    445     445   294.001 < 2.2e-16 ***
lo(X11, span = 0.1)       1.0   3175    3175  2097.701 < 2.2e-16 ***
X103                      1.0    107     107    70.603 < 2.2e-16 ***
X3                        1.0    440     440   290.543 < 2.2e-16 ***
X7                        1.0    430     430   283.811 < 2.2e-16 ***
X79                       1.0    498     498   328.950 < 2.2e-16 ***
X43                       1.0    416     416   274.971 < 2.2e-16 ***
X6                        1.0     49      49    32.542 1.422e-08 ***
X59                       1.0    295     295   195.065 < 2.2e-16 ***
X37                       1.0     29      29    18.851 1.515e-05 ***
lo(X88, span = 0.4)       1.0     78      78    51.815 9.926e-13 ***
s(X14 * X1, 11)           1.0   1087    1087   718.401 < 2.2e-16 ***
s(X14 * X55, 2)           1.0   1010    1010   667.326 < 2.2e-16 ***
s(X1 * X55, 2)            1.0   1117    1117   737.900 < 2.2e-16 ***
lo(X54, X1, span = 0.1)   2.0    395     197   130.402 < 2.2e-16 ***
Residuals              1397.2   2114       2
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                       Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)                   7.0 440.08 < 2.2e-16 ***
s(X86, 11)                10.0 624.59 < 2.2e-16 ***
lo(X11, span = 0.1)       17.6  42.72 < 2.2e-16 ***
X103
X3
X7
X79
X43
X6
X59
```

```
X37
lo(X88, span = 0.4)         3.5   4.66  0.001695 **
s(X14 * X1, 11)            10.0  10.14 < 2.2e-16 ***
s(X14 * X55, 2)             1.0  59.34 2.531e-14 ***
s(X1 * X55, 2)              1.0  32.50 1.440e-08 ***
lo(X54, X1, span = 0.1)    34.7   2.38 1.317e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 1.18727
```

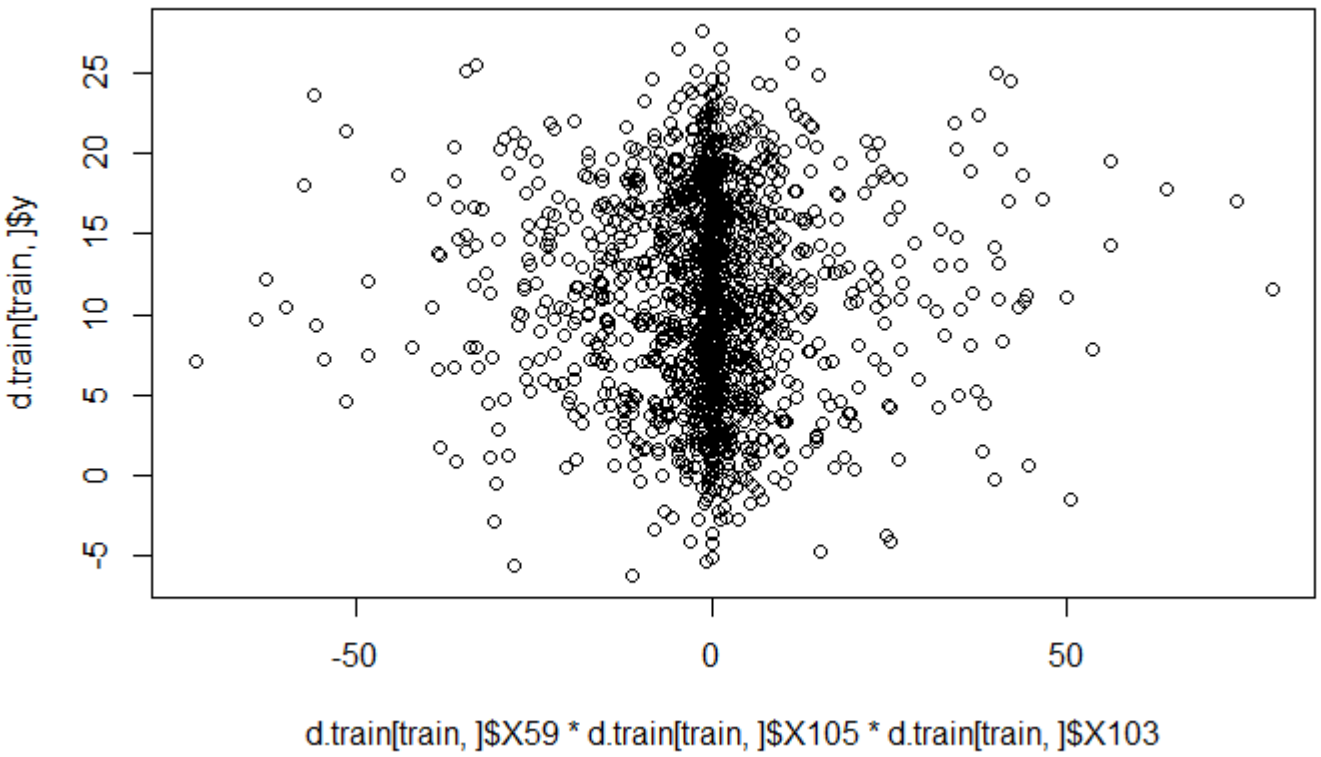
Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")

# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.2566
```

Hide

```
#Variable 17
# Plot interaction between variables X59, X105, X103
plot(d.train[train,]$X59*d.train[train,]$X105*d.train[train,]$X103, d.train[train,]$y)
```



Hide

```
# Choose optimal span for local regression
span.seq <- seq(from = 0.1, to = 0.9, by = 0.1)
span = 0.1
testerror = 50000000000
for(i in 1:length(span.seq)) {
  gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
            +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2)+s(X1*X55,2)
            +lo(X54,X1,span=0.1)+lo(X59,X105,X103,span =span.seq[i]),data=d.train[train,])
  preds <- predict(gam, newdata = d.train[-train,],type="response")
  testerror_i = sqrt(mean((preds - y.test)^2))
  if (testerror_i<testerror){
    testerror = testerror_i
    span = span.seq[i]
  }
}
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
```

Hide

```
#span 0.1 selected
span
```

```
[1] 0.1
```

Hide

```
# Add interaction between variables X59, X105, X103 to GAM model
gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+X103+X3+X7+X79+X43+X6+X59+X37
          +lo(X88,span=0.4)+s(X14*X1,11)+s(X14*X55,2)+s(X1*X55,2)
          +lo(X54,X1,span=0.1)+lo(X59,X105,X103,span =0.1),data=d.train[train,])

# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    X103 + X3 + X7 + X79 + X43 + X6 + X59 + X37 + lo(X88, span = 0.4) +
    s(X14 * X1, 11) + s(X14 * X55, 2) + s(X1 * X55, 2) + lo(X54,
    X1, span = 0.1) + lo(X59, X105, X103, span = 0.1), data = d.train[train,
    ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-3.02278 -0.64492 -0.02644  0.59975  4.15715

(Dispersion Parameter for gaussian family taken to be 1.1124)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 1485.91 on 1335.745 degrees of freedom
AIC: 4573.169

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                                  Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)                         1.0  32029   32029 28792.324 < 2.2e-16 ***
s(X86, 11)                       1.0    428     428   385.030 < 2.2e-16 ***
lo(X11, span = 0.1)              1.0   3188    3188  2865.942 < 2.2e-16 ***
X103                             1.0    108     108    97.004 < 2.2e-16 ***
X3                               1.0    449     449   403.710 < 2.2e-16 ***
X7                               1.0    448     448   402.458 < 2.2e-16 ***
X79                              1.0    530     530   476.619 < 2.2e-16 ***
X43                              1.0    411     411   369.169 < 2.2e-16 ***
X6                               1.0     47      47    42.668 9.198e-11 ***
X59                              1.0    297     297   266.716 < 2.2e-16 ***
X37                              1.0     30      30    26.939 2.425e-07 ***
lo(X88, span = 0.4)              1.0    104     104    93.510 < 2.2e-16 ***
s(X14 * X1, 11)                  1.0   1083    1083   973.364 < 2.2e-16 ***
s(X14 * X55, 2)                  1.0   1039    1039   933.761 < 2.2e-16 ***
s(X1 * X55, 2)                   1.0   1132    1132  1017.920 < 2.2e-16 ***
lo(X54, X1, span = 0.1)          2.0    377     188   169.397 < 2.2e-16 ***
lo(X59, X105, X103, span = 0.1)  1.0    428     428   385.080 < 2.2e-16 ***
Residuals                       1335.7   1486       1
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                               Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)                           7.0 600.83 < 2.2e-16 ***
s(X86, 11)                        10.0 873.89 < 2.2e-16 ***
lo(X11, span = 0.1)               17.6  59.51 < 2.2e-16 ***
X103
X3
X7
X79
X43
```

```
X6
X59
X37
lo(X88, span = 0.4)                 3.5   7.10 3.376e-05 ***
s(X14 * X1, 11)                    10.0  15.30 < 2.2e-16 ***
s(X14 * X55, 2)                     1.0  89.53 < 2.2e-16 ***
s(X1 * X55, 2)                      1.0  45.27 2.517e-11 ***
lo(X54, X1, span = 0.1)            34.7   2.45 7.037e-06 ***
lo(X59, X105, X103, span = 0.1)    60.4   3.40 4.441e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
```

Hide

```
# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 0.9952925
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
```

Hide

```
# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.098009
```

Hide

```
# ########### Final GAM Model ##############

gam = gam(y~s(X4,8)+s(X86,11)+lo(X11,span=0.1)+s(X14*X1,11)+X103
          +s(X14*X55,2)+s(X1*X55,2)+X3+X7+X79+X43+lo(X88,span=0.4)
          +X6+X59+X37+lo(X54,X1,span=0.1)+lo(X59,X105,X103,span =0.1), data=d.train[train,])
# Model summary
summary(gam)
```

```
Call: gam(formula = y ~ s(X4, 8) + s(X86, 11) + lo(X11, span = 0.1) +
    s(X14 * X1, 11) + X103 + s(X14 * X55, 2) + s(X1 * X55, 2) +
    X3 + X7 + X79 + X43 + lo(X88, span = 0.4) + X6 + X59 + X37 +
    lo(X54, X1, span = 0.1) + lo(X59, X105, X103, span = 0.1),
    data = d.train[train, ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-3.02278 -0.64492 -0.02644  0.59975  4.15715

(Dispersion Parameter for gaussian family taken to be 1.1124)

    Null Deviance: 61013.31 on 1499 degrees of freedom
Residual Deviance: 1485.91 on 1335.745 degrees of freedom
AIC: 4573.169

Number of Local Scoring Iterations: 1

Anova for Parametric Effects
                                  Df Sum Sq Mean Sq   F value    Pr(>F)
s(X4, 8)                         1.0  32029   32029 28792.324 < 2.2e-16 ***
s(X86, 11)                       1.0    428     428   385.030 < 2.2e-16 ***
lo(X11, span = 0.1)              1.0   3188    3188  2865.942 < 2.2e-16 ***
s(X14 * X1, 11)                  1.0   1164    1164  1046.105 < 2.2e-16 ***
X103                             1.0    105     105    94.763 < 2.2e-16 ***
s(X14 * X55, 2)                  1.0   1111    1111   998.909 < 2.2e-16 ***
s(X1 * X55, 2)                   1.0   1166    1166  1048.254 < 2.2e-16 ***
X3                               1.0    418     418   375.405 < 2.2e-16 ***
X7                               1.0    463     463   416.423 < 2.2e-16 ***
X79                              1.0    424     424   381.297 < 2.2e-16 ***
X43                              1.0    429     429   385.479 < 2.2e-16 ***
lo(X88, span = 0.4)              1.0    159     159   142.690 < 2.2e-16 ***
X6                               1.0     31      31    28.173 1.298e-07 ***
X59                              1.0    163     163   146.082 < 2.2e-16 ***
X37                              1.0     45      45    40.258 3.040e-10 ***
lo(X54, X1, span = 0.1)          2.0    377     188   169.397 < 2.2e-16 ***
lo(X59, X105, X103, span = 0.1)  1.0    428     428   385.080 < 2.2e-16 ***
Residuals                       1335.7   1486       1
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                                Npar Df Npar F     Pr(F)
(Intercept)
s(X4, 8)                            7.0 600.83 < 2.2e-16 ***
s(X86, 11)                         10.0 873.89 < 2.2e-16 ***
lo(X11, span = 0.1)                17.6  59.51 < 2.2e-16 ***
s(X14 * X1, 11)                    10.0  15.30 < 2.2e-16 ***
X103
s(X14 * X55, 2)                     1.0  89.53 < 2.2e-16 ***
s(X1 * X55, 2)                      1.0  45.27 2.517e-11 ***
X3
```

```
X7
X79
X43
lo(X88, span = 0.4)                 3.5   7.10 3.376e-05 ***
X6
X59
X37
lo(X54, X1, span = 0.1)            34.7   2.45 7.037e-06 ***
lo(X59, X105, X103, span = 0.1)    60.4   3.40 4.441e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
# Training set prediction
yhat_train = predict(gam, d.train[train,],type="response")
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
```

Hide

```
# Rooted mean squared error
sqrt(mean((yhat_train - d.train[train,'y'])^2))
```

```
[1] 0.9952925
```

Hide

```
# Test set prediction
yhat_test = predict(gam, d.train[-train,],type="response")
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
```

Hide

```
# Rooted mean squared error
sqrt(mean((yhat_test - y.test)^2))
```

```
[1] 1.098009
```

Hide

```
######## Final Prediction for Kaggle competition using GAM model ############
pred_gam = predict(gam,d.test[-1],type="response")
```

```
Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==  :
  prediction from a rank-deficient fit may be misleading
Warning in gam.lo(data[["lo(X59, X105, X103, span = 0.1)"]], z, w, span = 0.1,  :
  eval  22 -1.0415 -2.4097
Warning in gam.lo(data[["lo(X59, X105, X103, span = 0.1)"]], z, w, span = 0.1,  :
  lowerlimit  22 -1.762 -2.3965
Warning in gam.lo(data[["lo(X59, X105, X103, span = 0.1)"]], z, w, span = 0.1,  :
  extrapolation not allowed with blending
```