STA 567 HW5

Lina Lee

10/25/2019

```
setwd("C:/Users/linal/Desktop/Miami2019/STA567/Homework/Homework5")
powerplant <- read.csv("powerplant.csv")
```

Validation choice

For all the models, I used 5-fold cross validation to evaluate models' predictive performance using train function in Caret package. I used RMSE as the validated Test MSE values for the five models.

Polynomial regression

```
# The first order polynomial
set.seed(123)
mlr_mod1 <- train(PE ~ V,
                   data=powerplant,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

mean(mlr_mod1$resample$RMSE)

## [1] 8.421648

# The second order polynomial
set.seed(123)
mlr_mod2 <- train(PE ~ V + I(V^2),
                   data=powerplant,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

mean(mlr_mod2$resample$RMSE)

## [1] 8.100268

# The third order polynomial
set.seed(123)
mlr_mod3<- train(PE ~ V + I(V^2) + I(V^3),
                   data=powerplant,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))
```

```
mlr_mod3$results$RMSE

## [1] 8.100743

# The fourth order polynomial
set.seed(123)
mlr_mod4<- train(PE ~ V + I(V^2) + I(V^3) +I(V^4),
                  data=powerplant,
                  method="lm",
                  trControl=trainControl(method="cv",number = 5),
                  preProcess = c("center", "scale"))

mlr_mod4$results$RMSE

## [1] 8.015729
```

Knot choice.

I refer to the scatter plot to make four sets of three knots.I chose three sets of knots around 45,55,and 70 since the pattern of the data spread changes at those points points on the scatterplot. knots1: (25.36, **40, 65, 70**, 81.56) ,knots2: (25.36, **45, 55, 70**, 81.56), knots3: (25.36, **45, 58, 65**, 81.56), knots4: (25.36, **50, 60, 75**, 81.56)

```
#create 4 sets of knots
v_knots1 <- c(min(powerplant$V),40,65,70,max(powerplant$V))
v_knots2 <- c(min(powerplant$V),45,55,70,max(powerplant$V))
v_knots3 <- c(min(powerplant$V),45,58,65,max(powerplant$V))
v_knots4 <- c(min(powerplant$V),50,60,75,max(powerplant$V))

# create a knots list inluding four sets of knots.
knot_list<-list(v_knots1,v_knots2,v_knots3,v_knots4)

#create a function to make a bin for each knots set
knots<-function(v_knots){
bin_v <- cut(powerplant$V, breaks=v_knots,
                          right=FALSE, include.lowest=TRUE)
return(bin_v)
}
#apply the function to create bin to knots list
bin_list<-lapply(knot_list,function(x) knots(x))


# transform bin list into dataframe
bin_df<-as.data.frame(bin_list)

# Change variable name for the each bin
names(bin_df)<-c("bin1", "bin2", "bin3", "bin4")

# combine bin columns with original data
data_bins<-cbind(powerplant,bin_df)
```

piecewise regression using 3 knots

```
# Try the first knots
set.seed(123)
  pwr_mod <- train(PE ~ bin1*V +bin1*I(V^2),
                   #use the combined data including bin
                   data=data_bins,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

  pwr_mod$results$RMSE

## [1] 7.777393

# Try the second knots
set.seed(123)
  pwr_mod2 <- train(PE ~ bin2*V +bin2*I(V^2),
                   data=data_bins,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

  pwr_mod2$results$RMSE

## [1] 7.644582

# Try the third knots
set.seed(123)
  pwr_mod3 <- train(PE ~ bin3*V +bin3*I(V^2),
                   data=data_bins,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

  pwr_mod3$results$RMSE

## [1] 7.734705

# Try the fourth knots
set.seed(123)
  pwr_mod4 <- train(PE ~ bin4*V +bin4*I(V^2),
                   data=data_bins,
                   method="lm",
                   trControl=trainControl(method="cv",number = 5),
                   preProcess = c("center", "scale"))

  pwr_mod4$results$RMSE

## [1] 7.689187
```

Cubic regression splines

```
# create a bs matrix
bs<-bs(powerplant$V,knots=v_knots1,degree=3)
bs_mat<-as.data.frame(bs)
names(bs_mat)<-c("x","x2","x3","x_kn1","x_kn2","x_kn3","x_kn4","x_kn5")
# combine the bs matrix with original powerplant data
powerplant2<-cbind(powerplant,bs_mat)

# Try the first knots
set.seed(123)
power_spline1 <- train(PE~x+x2+x3+x_kn1+x_kn2+x_kn3+x_kn4+x_kn5,
                  data=powerplant2,
                  method="lm",
                  trControl=trainControl(method="cv",number = 5),
                  preProcess = c("center", "scale"))

power_spline1$results$RMSE

## [1] 7.928718

# create a bs matrix
bs<-bs(powerplant$V,knots=v_knots2,degree=3)
bs_mat<-as.data.frame(bs)
names(bs_mat)<-c("x","x2","x3","x_kn1","x_kn2","x_kn3","x_kn4","x_kn5")
# combine the bs matrix with original powerplant data
powerplant2<-cbind(powerplant,bs_mat)

# Try the second knots
set.seed(123)
power_spline2 <- train(PE~x+x2+x3+x_kn1+x_kn2+x_kn3+x_kn4+x_kn5,
                  data=powerplant2,
                  method="lm",
                  trControl=trainControl(method="cv",number = 5),
                  preProcess = c("center", "scale"))

power_spline2$results$RMSE

## [1] 7.860859

# create a bs matrix
bs<-bs(powerplant$V,knots=v_knots3,degree=3)
bs_mat<-as.data.frame(bs)
names(bs_mat)<-c("x","x2","x3","x_kn1","x_kn2","x_kn3","x_kn4","x_kn5")
# combine the bs matrix with original powerplant data
powerplant2<-cbind(powerplant,bs_mat)

# Try the third knots
set.seed(123)
power_spline3 <- train(PE~x+x2+x3+x_kn1+x_kn2+x_kn3+x_kn4+x_kn5,
                  data=powerplant2,
                  method="lm",
```

```
                 trControl=trainControl(method="cv",number = 5),
                 preProcess = c("center", "scale"))

power_spline3$results$RMSE

## [1] 7.837153

# create a bs matrix
bs<-bs(powerplant$V,knots=v_knots4,degree=3)
bs_mat<-as.data.frame(bs)
names(bs_mat)<-c("x","x2","x3","x_kn1","x_kn2","x_kn3","x_kn4","x_kn5")
# combine the bs matrix with original powerplant data
powerplant2<-cbind(powerplant,bs_mat)

# Try the fourth knots
set.seed(123)
power_spline4 <- train(PE~x+x2+x3+x_kn1+x_kn2+x_kn3+x_kn4+x_kn5,
                 data=powerplant2,
                 method="lm",
                 trControl=trainControl(method="cv",number = 5),
                 preProcess = c("center", "scale"))

power_spline4$results$RMSE

## [1] 7.786256
```

Smoothing splines

```
# The first df trial: df=seq(0,1000,by=20)
set.seed(123)
smooth_spline1<- train(PE ~ V,
                  method="gamSpline",
                  data=powerplant,
                  trControl=trainControl(method="cv",number = 5),
                  tuneGrid=data.frame(df=seq(0,1000,by=20)))

## Loading required package: gam

## Loading required package: foreach

##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when

## Loaded gam 1.16.1

smooth_spline1$finalModel$tuneValue

##    df
## 8 140
```

```r
# The second df trial: df=seq(120,160,by=5)
set.seed(123)
smooth_spline2<- train(PE ~ V,
                  method="gamSpline",
                  data=powerplant,
                  trControl=trainControl(method="cv",number = 5),
                  tuneGrid=data.frame(df=seq(120,160,by=5)))

smooth_spline2$finalModel$tuneValue

##    df
## 2 125

# The last df trial: df=seq(120,130,by=1)
set.seed(123)
smooth_spline3<- train(PE ~ V,
                  method="gamSpline",
                  data=powerplant,
                  trControl=trainControl(method="cv",number = 5),
                  tuneGrid=data.frame(df=seq(120,130,by=1)))

smooth_spline3$finalModel$tuneValue

##    df
## 5 124

mean(smooth_spline3$resample$RMSE)

## [1] 7.263602
```

The degrees of freedom choice for smooth splines

First, I tried sequence from 0 to 1000 by 20 for degrees of freedom to investigate best tune. The result gave me the best df of 140. Next, I applied the sequence from 120 to 160 by 5 for degrees of freedom. The result says that df 125 is the best tune.I investigate further by using the sequence from 120 to 130 by 1 for the degrees of freedom. From the result, the degrees of freedom 124 is the best.

LOESS using a standard weight

```r
# The first tune grid
tune_grid <- expand.grid(span = seq(0.1, 0.9, by=0.2), degree = 1)

set.seed(123)
loess_mod<- train(PE ~ V,
                  method="gamLoess",
                  data=powerplant,
                  trControl=trainControl(method="cv",number = 5),
                  tuneGrid=tune_grid)

loess_mod$finalModel$tuneValue
```

```
##   span degree
## 1  0.1      1

# The second tune grid
tune_grid2 <- expand.grid(span = seq(0.05, 0.2, by=0.1), degree = 1)

set.seed(123)
loess_mod2<- train(PE ~ V,
                      method="gamLoess",
                      data=powerplant,
                      trControl=trainControl(method="cv",number = 5),
                      tuneGrid=tune_grid2)

##   span degree
## 1 0.05      1

# The last tune grid
tune_grid3 <- expand.grid(span = seq(0.01, 0.1, by=0.005), degree = 1)

set.seed(123)
loess_mod3<- train(PE ~ V,
                      method="gamLoess",
                      data=powerplant,
                      trControl=trainControl(method="cv",number = 5),
                      tuneGrid=tune_grid2)

loess_mod3$finalModel$tuneValue

##   span degree
## 1 0.05      1

mean(loess_mod3$resample$RMSE)

## [1] 7.508623
```

span and polynomial degree choice for LOESS

First, I tried sequence from 0.1 to 0.9 by 0.2, and degree 1. The degree 2 broke all the operations, so I only applied degree 1. The result said that span 0.1 is the best tune.since span 0.1 is edge of the sequence, I investigate further below the span 0.1. secondly, I tried the sequence from 0.05 to 0.2 by 0.1. The result said that span 0.05 is the best. Finally, I applied the sequence from 0.01 to 0.1 by 0.005. The result said that span 0.05 is the best tune. 0.05 is not a edge in the sequence, I didn't investigate furthur, and I concluded that span 0.05 is the best tune.

Table 1 Selected parameters and RMSE

Model Types	**Parameters to Tune**	**RMSE**
Polynomial regression	Tried polynomial: From the first to fourth polynomial. Best polynomial order: The fourth order polynomial	First: 8.4216 Second: 8.1002 Third: 8.1007 Fourth: 8.0157 The lowest RMSE: 8.0157
Piecewise linear regression using 3 knots	Tried knot locations knots1: (25.36**, 40, 65, 70**, 81.56) knots2: (25.36, **45, 55, 70**, 81.56) knots3: (25.36, **45, 58, 65**, 81.56) knots4: (25.36, **50, 60, 75**, 81.56) Best tune: knots2 (25.36, **45, 55, 70**, 81.56)	Knots1: 7.7773 Knots2: 7.6445 Knots3: 7.7347 Knots4: 7.6891 The lowest RMSE: 7.6445
Cubic regression splines using 3 knots	Tried knot locations knots1: (25.36, **40, 65, 70**, 81.56) knots2: (25.36, **45, 55, 70**, 81.56) knots3: (25.36, **45, 58, 65**, 81.56) knots4: (25.36, **50, 60, 75**, 81.56) Best tune: knots2 knots4: (25.36, **50, 60, 75**, 81.56)	Knots1: 7.9287 Knots2: 7.8608 Knots3: 7.8371 Knots4: 7.7862 The lowest RMSE: 7.7862
Smoothing splines	Effective degrees of freedom (df): 124	7.2636
LOESS using a standard weight	span 0.05, polynomial degree: 1	7.5086

Conclusion: Among the all the models Smoothing splines with degrees of freedom 124 has the lowest RMSE as 7.2636, therefore the model provides the strongest predictions for the energy output values.