Cross_validation_type

Vinoth Aryan Nagabosshanam

March 23, 2017

here now we are going to implement the cross valudation techniques

```
library(ISLR)
```

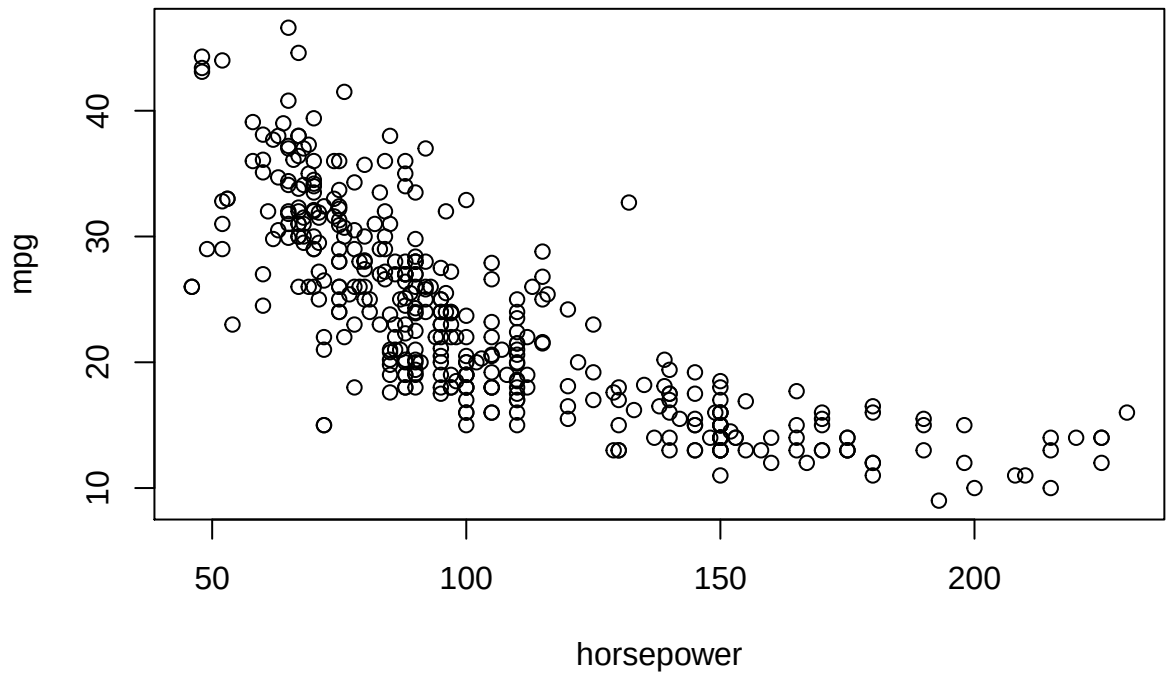
```
## Warning: package 'ISLR' was built under R version 3.3.3
```

```
library(boot)
```

```
## Warning: package 'boot' was built under R version 3.3.3
```

```
# boot is package which we need process cross validation process

#View(Auto)
# we are using Auto dataset
plot(mpg~horsepower,data=Auto)
```




```
# we simple building simple linear regression model
mod_fit<-glm(mpg~horsepower,data=Auto)
summary(mod_fit)
```

```
##
## Call:
```

```
## glm(formula = mpg ~ horsepower, data = Auto)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -13.5710   -3.2592   -0.3435    2.7630   16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 24.06645)
##
##     Null deviance: 23819.0  on 391  degrees of freedom
## Residual deviance:  9385.9  on 390  degrees of freedom
## AIC: 2363.3
##
## Number of Fisher Scoring iterations: 2
```

```
#now we fitting cross validation function to model
cv.glm(Auto,mod_fit)$delta
```

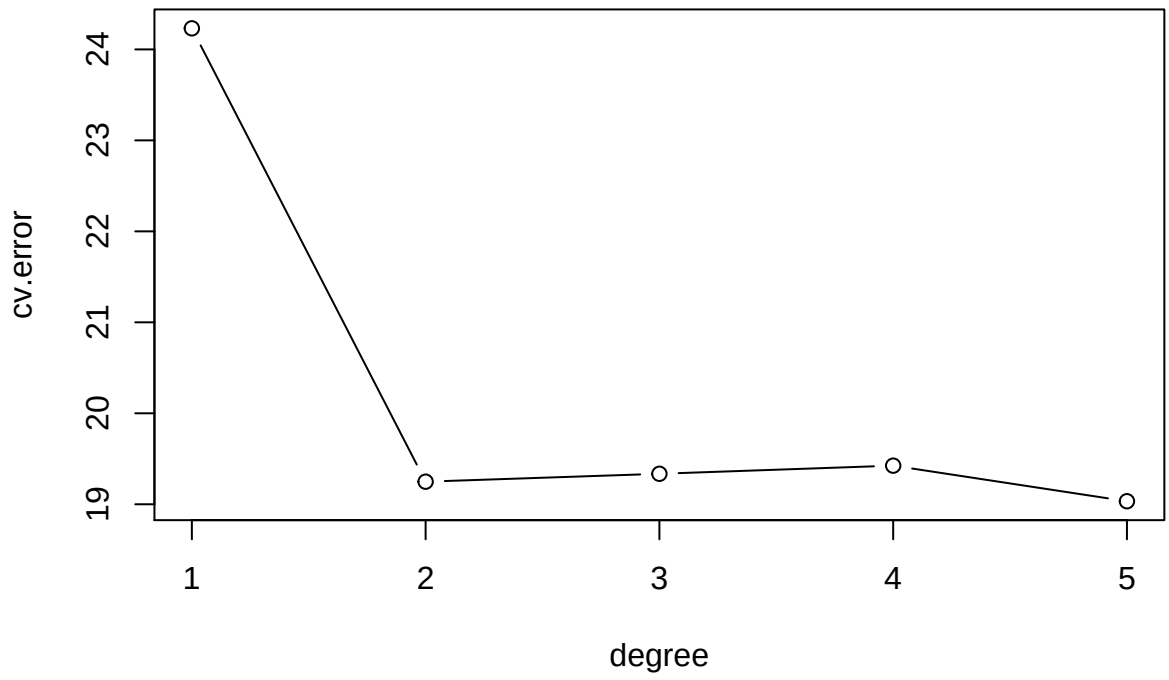
```
## [1] 24.23151 24.23114
```

```
##Lets write a simple function to use formula (5.2)
loocv=function(fit){
  h=lm.influence(fit)$h
  mean((residuals(fit)/(1-h))^2)
}

## Now we try it out
loocv(mod_fit)
```

```
## [1] 24.23151
```

```
cv.error=rep(0,5)
degree=1:5
for(d in degree){
  mod_fit=glm(mpg~poly(horsepower,d), data=Auto)
  cv.error[d]=loocv(mod_fit)
}
plot(degree,cv.error,type="b")
```




```
da <- get(load('B:/5.R.Rdata'))
#plot(y~X1+X2,data=da)
# we simple building simple linear regression model
modfit<-glm(y~.,data=da)
summary(modfit)
```

```
##
## Call:
## glm(formula = y ~ ., data = da)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.44171  -0.25468  -0.01736   0.33081   1.45860
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.26583    0.01988  13.372  < 2e-16 ***
## X1           0.14533    0.02593   5.604 2.71e-08 ***
## X2           0.31337    0.02923  10.722  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2971417)
##
##     Null deviance: 335.56  on 999  degrees of freedom
## Residual deviance: 296.25  on 997  degrees of freedom
```

```
## AIC: 1629.3
##
## Number of Fisher Scoring iterations: 2
```

```
#now we fitting cross validation function to model
cv.glm(da,modfit)$delta
```

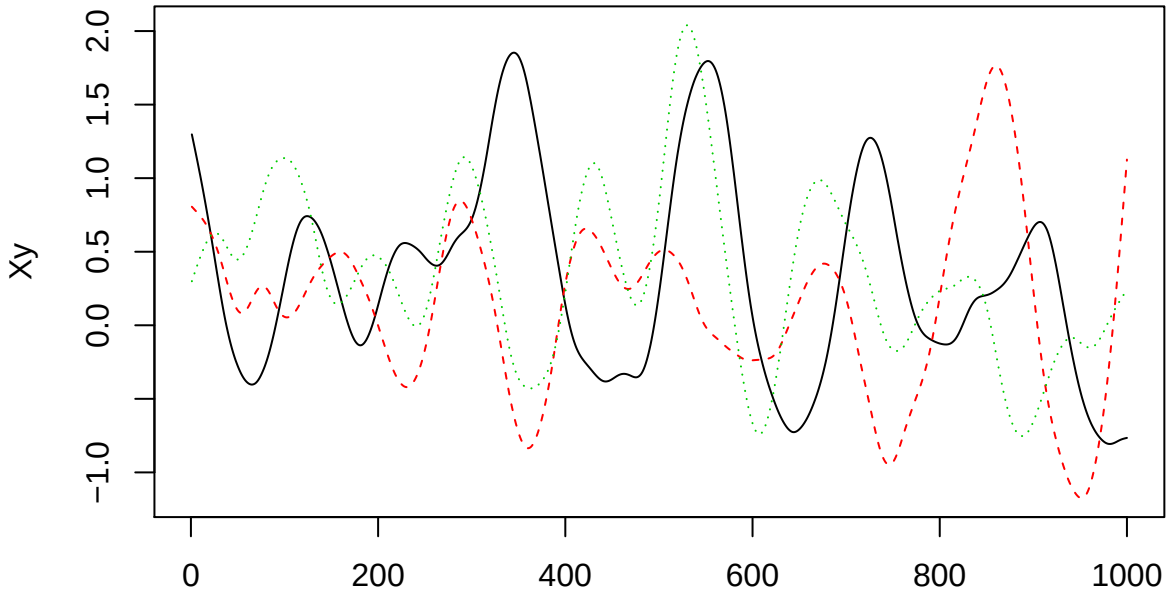
```
## [1] 0.2984826 0.2984815
```

```
##Lets write a simple function to use formula (5.2)
loocv=function(fit){
  h=lm.influence(fit)$h
  mean((residuals(fit)/(1-h))^2)
}

## Now we try it out
loocv(modfit)
```

```
## [1] 0.2984826
```

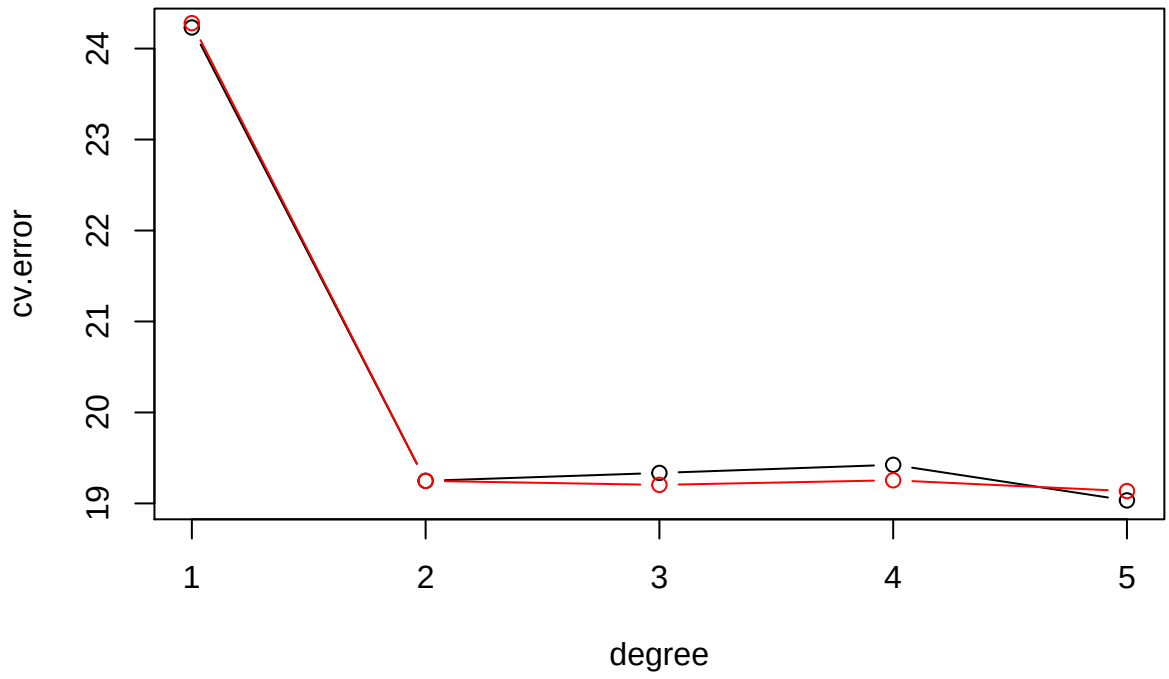
```
matplot(Xy,type="l")
```



10-fold CV

```
cv.error10=rep(0,5)
for(d in degree){
  mod_fit=glm(mpg~poly(horsepower,d), data=Auto)
```

```
  cv.error10[d]=cv.glm(Auto,mod_fit,K=10)$delta[1]
}
plot(degree,cv.error,type="b")
lines(degree,cv.error10,type="b",col="red")
```



Bootstrap

Minimum risk investment

```
alpha=function(x,y){
  vx=var(x)
  vy=var(y)
  cxy=cov(x,y)
  (vy-cxy)/(vx+vy-2*cxy)
}
alpha(Portfolio$X,Portfolio$Y)
```

```
## [1] 0.5758321
```

```
## What is the standard error of alpha?

alpha.fn=function(data, index){
  with(data[index,],alpha(X,Y))
}
```

```
alpha.fn(Portfolio,1:100)
```

```
## [1] 0.5758321
```

```
set.seed(1)
alpha.fn (Portfolio,sample(1:100,100,replace=TRUE))
```

```
## [1] 0.5963833
```

```
boot.out=boot(Portfolio,alpha.fn,R=1000)
boot.out
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Portfolio, statistic = alpha.fn, R = 1000)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 0.5758321 -7.315422e-05  0.08861826
```

```
plot(boot.out)
```