

# Life\_expectancy project code

LitianZhou

04/07/2019

```
library(boot)
library(pls)
library(leaps)
library(gam)
library(glmnet)

life_exp=read.csv("Life Expectancy Data.csv")
life_exp=na.omit(life_exp)
life_exp=life_exp[2:22] # remove the "country" predictor
attach(life_exp)

set.seed(1)
trainid <- sample(1:nrow(life_exp), nrow(life_exp)/2)
train <- life_exp[trainid,]
test <- life_exp[-trainid,]

require(boot)
glm.fit = glm(Life.expectancy~., data=life_exp)
loocv.err.OLS=cv.glm(life_exp, glm.fit)$delta[1]
loocv.err.OLS

## [1] 12.84627

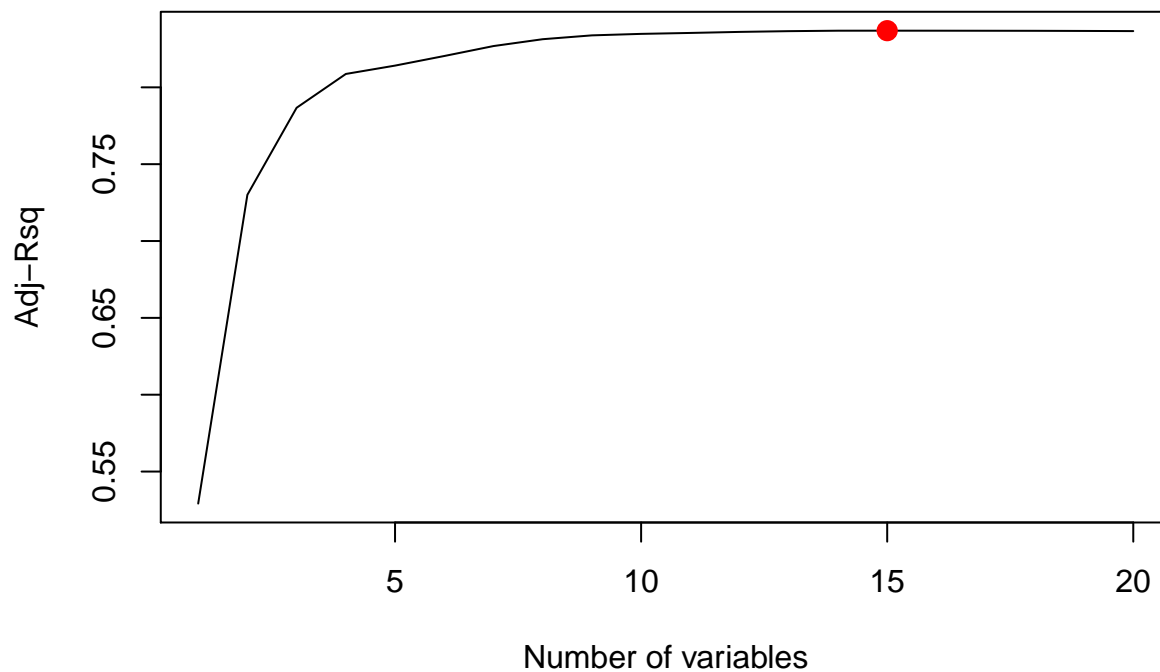
require(leaps)

set.seed(7)
regfit.full=regsubsets(Life.expectancy~., life_exp,nvmax=20)
reg.summary=summary(regfit.full)

plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adj-Rsq",type ="l")
which.max(reg.summary$adjr2) # give 15

## [1] 15

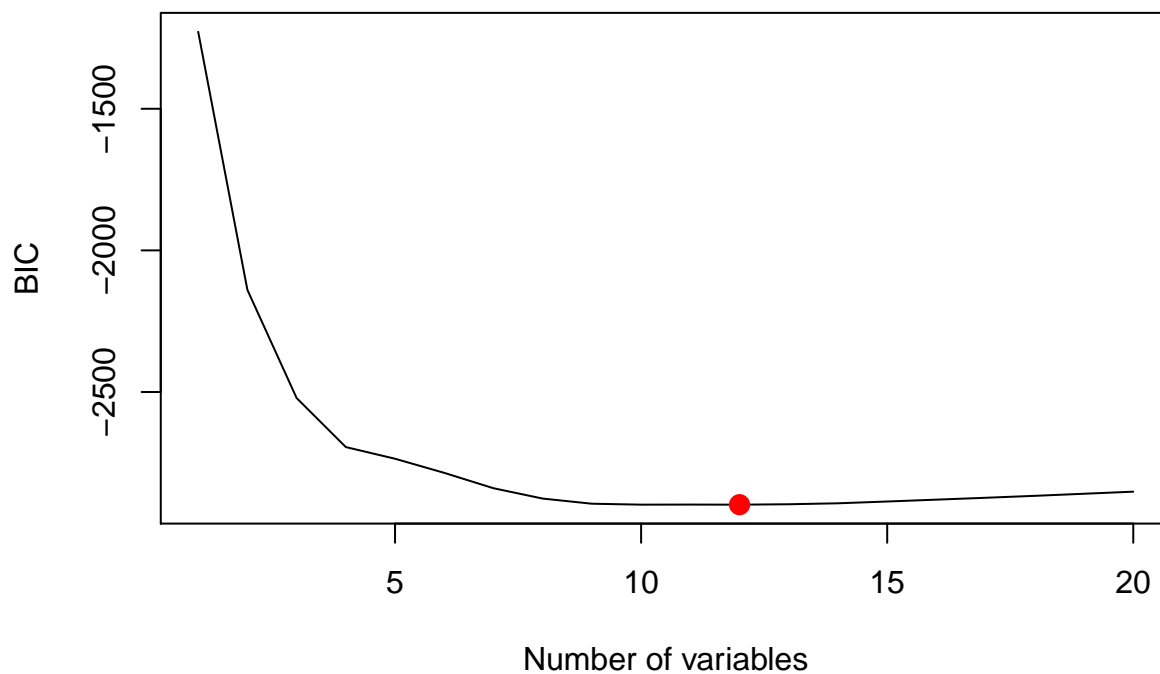
points(15,reg.summary$adjr2[15], col="red",cex=2,pch=20)
```



```
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(reg.summary$bic) # give 12
```

```
## [1] 12
```

```
points(12, reg.summary$bic[12], col = "red", cex = 2, pch = 20)
```



```
coef(regfit.full, 12)
```

```
##              (Intercept)              Year
##          3.048802e+02         -1.251579e-01
##          StatusDeveloping         Adult.Mortality
```

```
##          -9.240120e-01          -1.646155e-02
##          infant.deaths          Alcohol
##          8.361896e-02          -1.172219e-01
##          percentage.expenditure          BMI
##          4.638867e-04          3.697864e-02
##          under.five.deaths          Diphtheria
##          -6.395703e-02          1.541662e-02
##          HIV.AIDS Income.composition.of.resources
##          -4.457164e-01          1.056436e+01
##          Schooling
##          9.154314e-01
```

```
regfit.fwd=regsubsets(Life.expectancy~., life_exp, nvmax=20, method = "forward")
coef(regfit.fwd,12)
```

```
##          (Intercept)          Year
##          3.048802e+02          -1.251579e-01
##          StatusDeveloping          Adult.Mortality
##          -9.240120e-01          -1.646155e-02
##          infant.deaths          Alcohol
##          8.361896e-02          -1.172219e-01
##          percentage.expenditure          BMI
##          4.638867e-04          3.697864e-02
##          under.five.deaths          Diphtheria
##          -6.395703e-02          1.541662e-02
##          HIV.AIDS Income.composition.of.resources
##          -4.457164e-01          1.056436e+01
##          Schooling
##          9.154314e-01
```

```
regfit.bwd=regsubsets(Life.expectancy~., life_exp, nvmax=20, method = "backward")
coef(regfit.bwd,12)
```

```
##          (Intercept)          Year
##          3.048802e+02          -1.251579e-01
##          StatusDeveloping          Adult.Mortality
##          -9.240120e-01          -1.646155e-02
##          infant.deaths          Alcohol
##          8.361896e-02          -1.172219e-01
##          percentage.expenditure          BMI
##          4.638867e-04          3.697864e-02
##          under.five.deaths          Diphtheria
##          -6.395703e-02          1.541662e-02
##          HIV.AIDS Income.composition.of.resources
##          -4.457164e-01          1.056436e+01
##          Schooling
##          9.154314e-01
```

*# Using 12 predictors, BSS, forward selection and backward selection all give the identical model*

```
require(leaps)
```

*# predict function from chapter 6 labs*

```
predict.regsubsets <- function(object, newdata, id, ...){
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
```

```

coefi <- coef(object, id=id)
xvars <- names(coefi)
mat[,xvars] %*% coefi
}

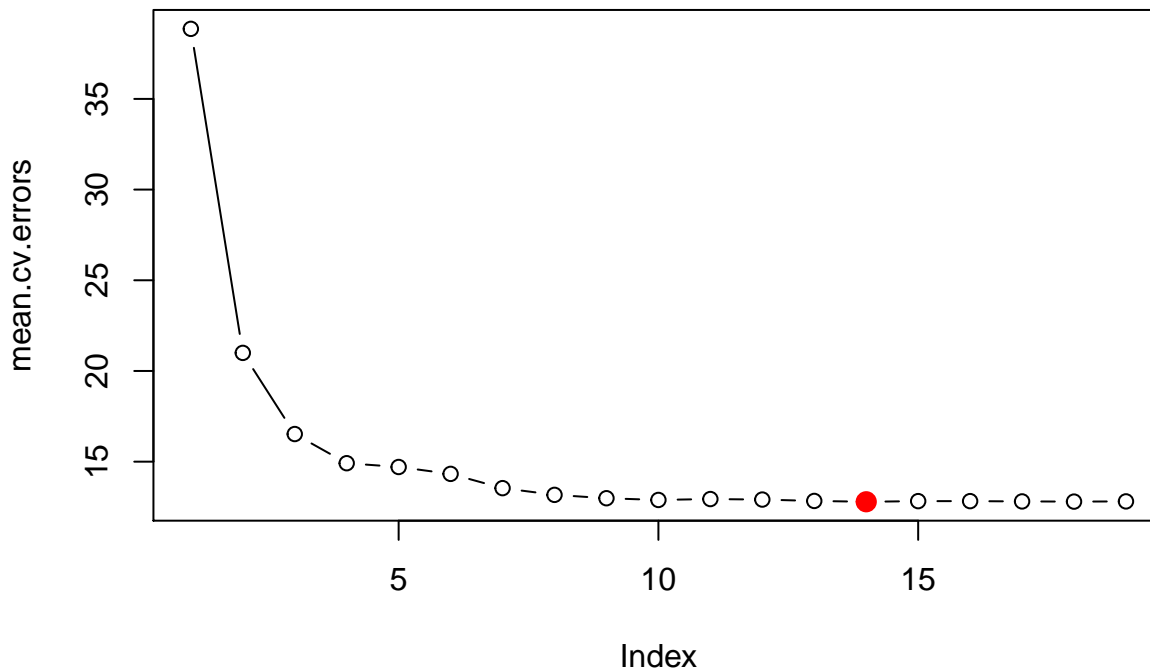
k=10
set.seed(5)
folds=sample(1:k, nrow(life_exp),replace = TRUE)
cv.errors=matrix(NA,k,19,dimnames=list(NULL,paste(1:19)))

for(j in 1:k){
  best.fit=regsubsets(Life.expectancy~., data=life_exp[folds!=j,], nvmax = 20)
  for(i in 1:19){
    pred=predict(best.fit, life_exp[folds==j,],id=i)
    cv.errors[j,i]=mean((life_exp$Life.expectancy[folds==j]-pred)^2)
  }
}
mean.cv.errors=apply(cv.errors,2,mean)
plot(mean.cv.errors,type="b")
which.min(mean.cv.errors) # give 14, depend on the random seed

## 14
## 14

points(14,mean.cv.errors[14], col="red",cex=2,pch=20)

```



```

mean.cv.errors[14]

##      14
## 12.78913

#use 14 predictors to fit the model
regfit.cv.select=regsubsets(Life.expectancy~.,data=life_exp, nvmax = 14)
coef(regfit.cv.select, 14)

```

```
##              (Intercept)                      Year
##          310.086873211                -0.127717305
##      StatusDeveloping      Adult.Mortality
##          -0.897508949                -0.016262157
##      infant.deaths          Alcohol
##          0.086080629                -0.129897924
##      percentage.expenditure      BMI
##          0.000452335                0.031993887
##      under.five.deaths      Total.expenditure
##          -0.065159011                0.092009080
##      Diphtheria            HIV.AIDS
##          0.015089120                -0.447801053
##      thinness.5.9.years Income.composition.of.resources
##          -0.052118455                10.523194825
##      Schooling
##          0.904719271
```

```
fit.lm <- lm(Life.expectancy~., data=train)
pred.lm <- predict(fit.lm, test)
(err.lm <- mean((test$Life.expectancy - pred.lm)^2))
```

```
## [1] 13.77854
```

*#err.lm may inflate the error since it uses validation set approach, which does not use whole data to b*

```
require(glmnet)
xmat.train <- model.matrix(Life.expectancy~., data=train)[,-1]
xmat.test <- model.matrix(Life.expectancy~., data=test)[,-1]
fit.ridge <- cv.glmnet(xmat.train, train$Life.expectancy, alpha=0)
(lambda <- fit.ridge$lambda.min) # optimal lambda
```

```
## [1] 0.7133576
```

```
pred.ridge <- predict(fit.ridge, s=lambda, newx=xmat.test)
(err.ridge <- mean((test$Life.expectancy - pred.ridge)^2)) # test error
```

```
## [1] 14.39299
```

```
lifetest.avg <- mean(life_exp$Life.expectancy)
#ridge.r2 <- 1 - mean((pred.ridge - xmat.test$Life.expectancy)^2) / mean((lifetest.avg - xmat.test$Life
```

```
xmat.train <- model.matrix(Life.expectancy~., data=train)[,-1]
xmat.test <- model.matrix(Life.expectancy~., data=test)[,-1]
fit.lasso <- cv.glmnet(xmat.train, train$Life.expectancy, alpha=1)
(lambda <- fit.lasso$lambda.min) # optimal lambda
```

```
## [1] 0.002879837
```

```
pred.lasso <- predict(fit.lasso, s=lambda, newx=xmat.test)
(err.lasso <- mean((test$Life.expectancy - pred.lasso)^2)) # test error
```

```
## [1] 13.78092
```

```
coef.lasso <- predict(fit.lasso, type="coefficients", s=lambda)[1:ncol(life_exp),]
coef.lasso[coef.lasso != 0]
```

```
##              (Intercept)                      Year
##          3.744972e+02                -1.592786e-01
##      StatusDeveloping      Adult.Mortality
```

```
##          -1.382550e+00          -1.667109e-02
##          infant.deaths          Alcohol
##          7.489124e-02          -1.564712e-01
##          percentage.expenditure          Hepatitis.B
##          3.947122e-04          -4.639169e-03
##          Measles          BMI
##          -9.974977e-06          3.011031e-02
##          under.five.deaths          Polio
##          -5.695558e-02          8.483315e-04
##          Total.expenditure          Diphtheria
##          -1.158607e-02          1.553190e-02
##          HIV.AIDS          GDP
##          -4.171617e-01          7.686975e-06
##          Population          thinness..1.19.years
##          -9.191059e-10          -2.193032e-02
##          thinness.5.9.years Income.composition.of.resources
##          -2.439081e-02          1.055896e+01
##          Schooling
##          9.344881e-01
```

```
length(coef.lasso[coef.lasso != 0])
```

```
## [1] 21
```

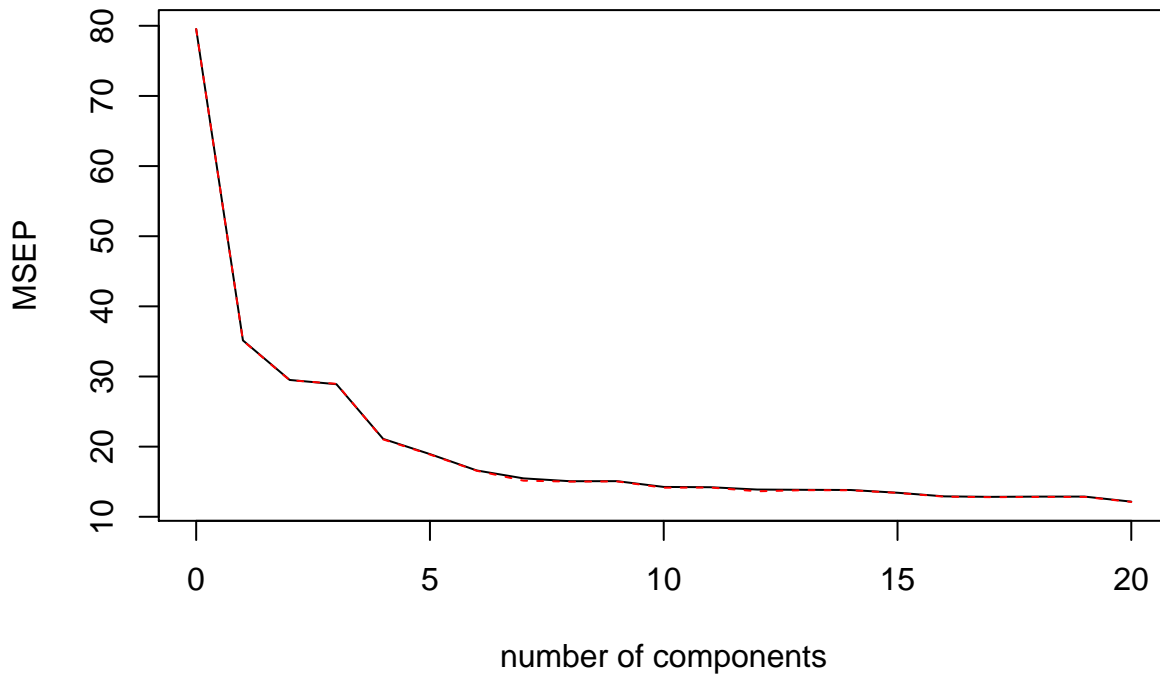
```
require(pls)
```

```
set.seed(1)
```

```
fit.pcr <- pcr(Life.expectancy~., data=train, scale=TRUE, validation="CV")
```

```
validationplot(fit.pcr, val.type="MSEP") # I choose M = 7
```

## Life.expectancy



```
summary(fit.pcr)
```

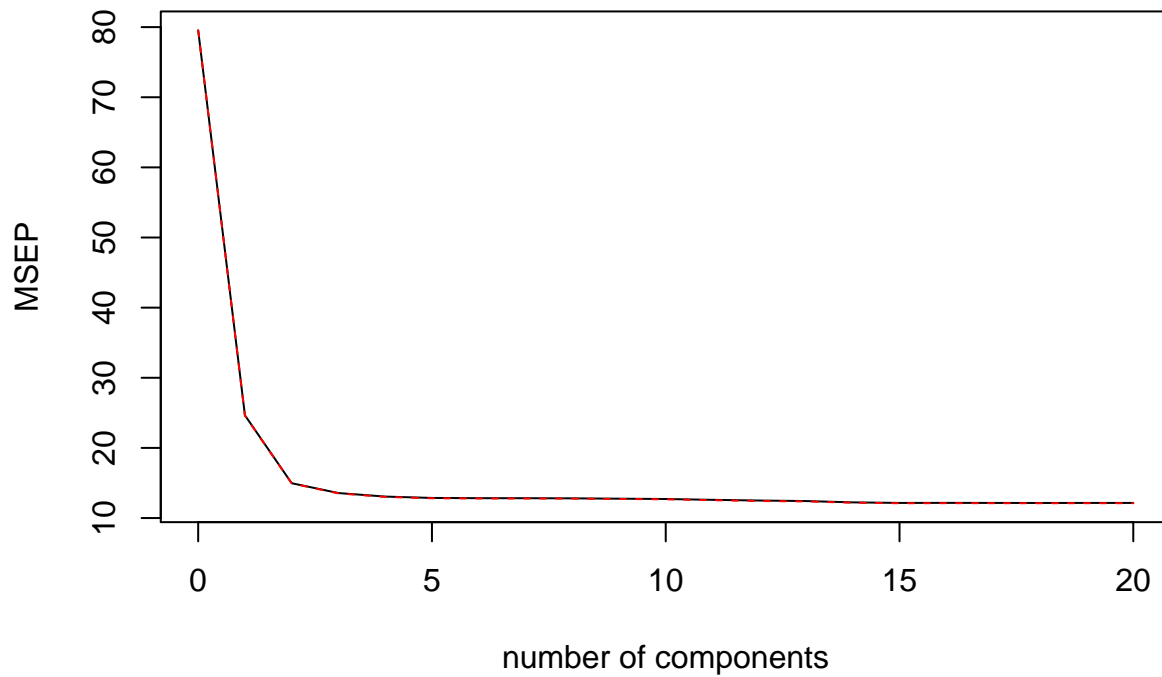
```
## Data:      X dimension: 824 20
## Y dimension: 824 1
## Fit method: svdpc
## Number of components considered: 20
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##      (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV              8.919   5.928   5.432   5.376   4.594   4.352   4.075
## adjCV           8.919   5.925   5.429   5.381   4.586   4.345   4.070
##      7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV          3.934   3.882   3.883   3.776   3.770   3.726   3.721
## adjCV       3.892   3.874   3.878   3.761   3.764   3.693   3.714
##      14 comps 15 comps 16 comps 17 comps 18 comps 19 comps
## CV          3.718   3.664   3.593   3.583   3.588   3.589
## adjCV       3.712   3.661   3.587   3.577   3.582   3.582
##      20 comps
## CV          3.487
## adjCV       3.481
##
## TRAINING: % variance explained
##      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X          29.45   43.08   51.87   59.66   66.19   71.25
## Life.expectancy 56.59   63.56   64.48   74.17   76.93   79.94
##      7 comps  8 comps  9 comps 10 comps 11 comps 12 comps
## X          75.65   79.78   83.46   86.40   89.13   91.29
## Life.expectancy 81.91   81.91   81.96   83.01   83.01   83.61
##      13 comps 14 comps 15 comps 16 comps 17 comps
## X          93.43   95.37   96.99   98.49   99.43
## Life.expectancy 83.62   83.74   84.14   84.68   84.79
##      18 comps 19 comps 20 comps
## X          99.79   99.98  100.00
## Life.expectancy 84.80   84.81   85.52
```

```
pred.pcr <- predict(fit.pcr, test, ncomp=10)
(err.pcr <- mean((test$Life.expectancy - pred.pcr)^2)) # test error
```

```
## [1] 15.37257
```

```
set.seed(1)
fit.pls <- plsrf(Life.expectancy~., data=train, scale=TRUE, validation="CV")
validationplot(fit.pls, val.type="MSEP")
```

## Life.expectancy



```
summary(fit.pls)
```

```
## Data:      X dimension: 824 20
## Y dimension: 824 1
## Fit method: kernelpls
## Number of components considered: 20
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##      (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV              8.919   4.965   3.869   3.683   3.614   3.587   3.582
## adjCV           8.919   4.960   3.866   3.677   3.607   3.581   3.576
##      7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV          3.582   3.579   3.572   3.566   3.549   3.535   3.525
## adjCV        3.576   3.573   3.566   3.560   3.542   3.526   3.517
##      14 comps 15 comps 16 comps 17 comps 18 comps 19 comps
## CV          3.498   3.487   3.488   3.487   3.486   3.487
## adjCV        3.490   3.480   3.482   3.481   3.480   3.481
##      20 comps
## CV          3.487
## adjCV        3.481
##
## TRAINING: % variance explained
##      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X          28.66   39.26   49.10   54.46   58.75   64.63
## Life.expectancy 69.91   81.89   83.91   84.58   84.77   84.81
##      7 comps  8 comps  9 comps 10 comps 11 comps 12 comps
## X          69.49   74.42   77.38   80.49   82.20   83.61
## Life.expectancy 84.83   84.85   84.90   84.97   85.15   85.31
##      13 comps 14 comps 15 comps 16 comps 17 comps
```



```
## X            86.71      87.60      89.26      92.80      94.02
## Life.expectancy 85.35      85.47      85.51      85.51      85.52
##              18 comps  19 comps  20 comps
## X            95.14      97.84     100.00
## Life.expectancy 85.52      85.52      85.52
```

```
pred.pls <- predict(fit.pls, test, ncomp=5) # min Cv at M=5
(err.pls <- mean((test$Life.expectancy - pred.pls)^2)) # test error
```

```
## [1] 14.24667
```

fit the CV\_chosen model into poly, regression spline and GAM:

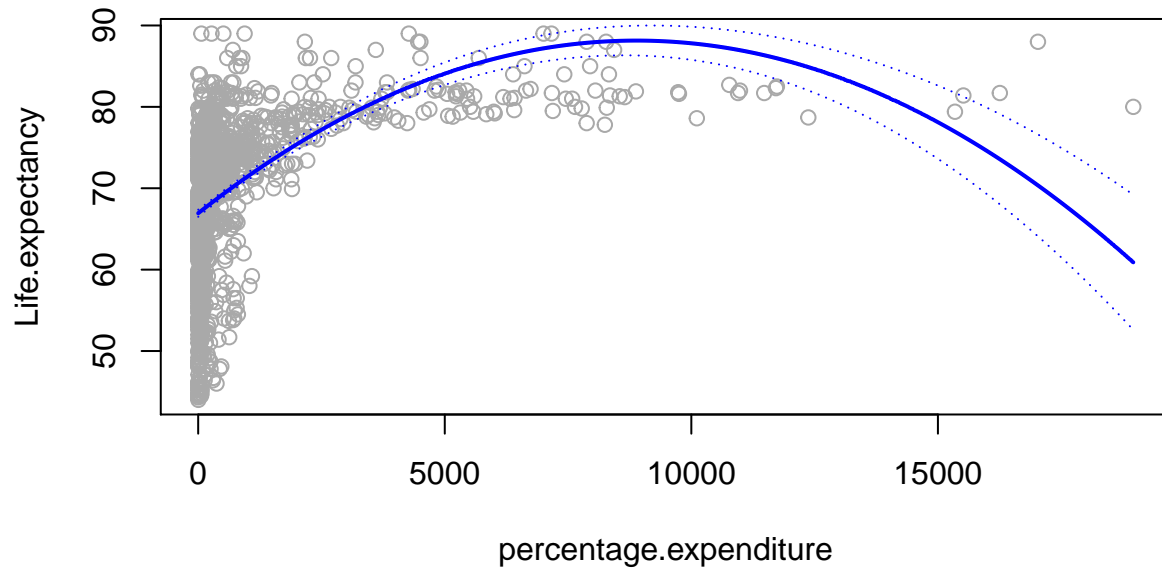
```
poly.fit=lm(Life.expectancy~poly(percentage.expenditure,2))
summary(poly.fit)
```

```
##
## Call:
## lm(formula = Life.expectancy ~ poly(percentage.expenditure, 2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.946  -4.079   1.315   5.288  21.765
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      69.3023     0.1889   366.83 <2e-16 ***
## poly(percentage.expenditure, 2)1 146.2843     7.6718   19.07 <2e-16 ***
## poly(percentage.expenditure, 2)2 -96.1937     7.6718  -12.54 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.672 on 1646 degrees of freedom
## Multiple R-squared:  0.2404, Adjusted R-squared:  0.2394
## F-statistic: 260.4 on 2 and 1646 DF,  p-value: < 2.2e-16
```

```
percentage.expenditurelims = range(percentage.expenditure)
percentage.expenditure.grid=seq(from=percentage.expenditurelims[1], to=percentage.expenditurelims[2])
preds=predict(poly.fit,newdata = list(percentage.expenditure=percentage.expenditure.grid), se=TRUE)
se.bands=cbind(preds$fit+2*preds$se.fit, preds$fit-2*preds$se.fit)
```

```
par(oma=c(0,0,3,0))
plot(percentage.expenditure,Life.expectancy, xlim=percentage.expenditurelims, col="darkgrey")
title("Degree-2 Polynomial",outer=T)
lines(percentage.expenditure.grid, preds$fit, lwd="2",col="blue")
matlines(percentage.expenditure.grid, se.bands, lwd=1, col="blue",lty=3)
```

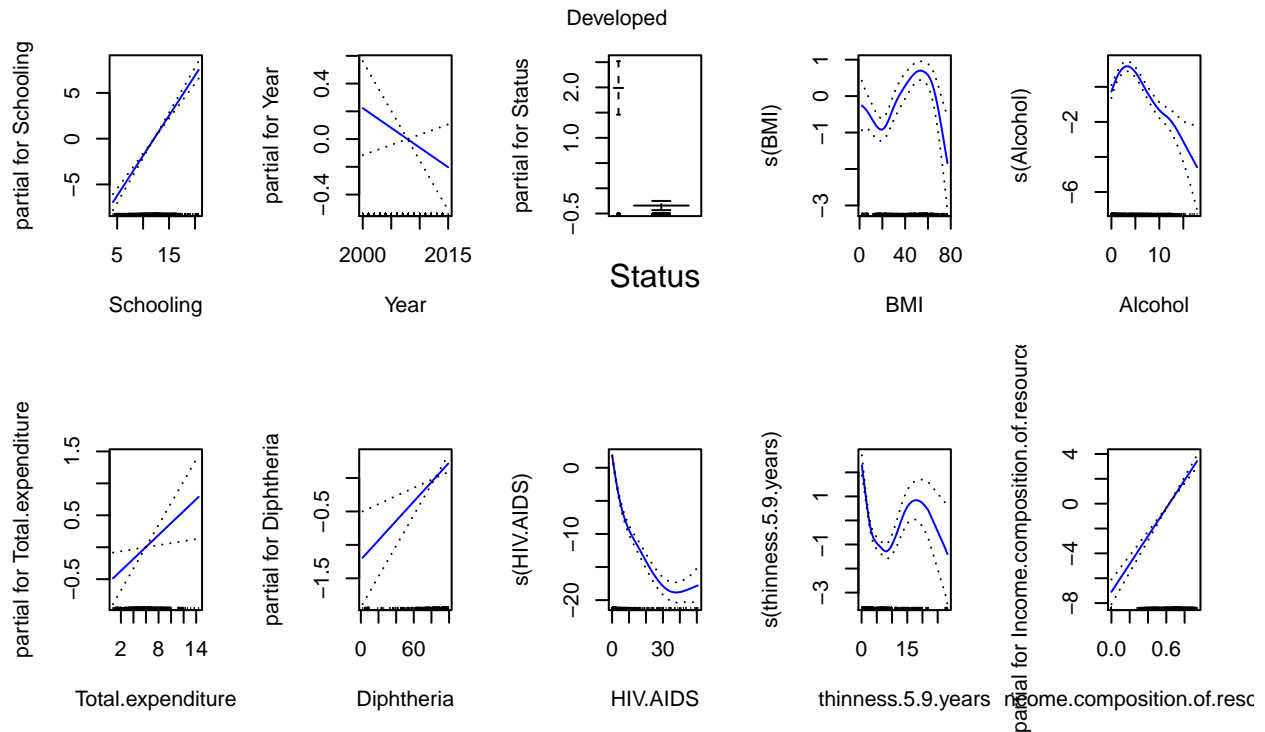
## Degree-2 Polynomial



The graph shows that as people increase their expenditure on health, the life expectancy first increases, then decreases (or unchanged).

```
require(gam)
# Choose from best subset model of 14 predictors. Then remove "Adult.Mortality", "infant.deaths", "under-5.mortality"
gam10.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol)+Total.expenditure+Diphtheria+s(HIV))
par(mfrow=c(2,5))
par(oma=c(0,0,3,0))
plot.Gam(gam10.fit, se=TRUE, col="blue")
title("GAM: linear + smoothing spline fit", outer=TRUE)
```

## GAM: linear + smoothing spline fit



```
gam1.fit=gam(Life.expectancy~Schooling)
gam2.fit=gam(Life.expectancy~Schooling+Year)
gam3.fit=gam(Life.expectancy~Schooling+Year+Status)
gam4.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI))
gam5.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol))
gam6.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol)+Total.expenditure)
gam7.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol)+Total.expenditure+Diphtheria)
gam8.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol)+Total.expenditure+Diphtheria+s(HIV
gam9.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+s(Alcohol)+Total.expenditure+Diphtheria+s(HIV

anova(gam1.fit, gam2.fit, gam3.fit, gam4.fit, gam5.fit, gam6.fit,gam7.fit,gam8.fit,gam9.fit,gam10.fit)
```

```
## Analysis of Deviance Table
```

```
##
```

```
## Model 1: Life.expectancy ~ Schooling
```

```
## Model 2: Life.expectancy ~ Schooling + Year
```

```
## Model 3: Life.expectancy ~ Schooling + Year + Status
```

```
## Model 4: Life.expectancy ~ Schooling + Year + Status + s(BMI)
```

```
## Model 5: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol)
```

```
## Model 6: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol) +
## Total.expenditure
```

```
## Model 7: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol) +
## Total.expenditure + Diphtheria
```

```
## Model 8: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol) +
## Total.expenditure + Diphtheria + s(HIV.AIDS)
```

```
## Model 9: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol) +
## Total.expenditure + Diphtheria + s(HIV.AIDS) + s(thinness.5.9.years)
```

```
## Model 10: Life.expectancy ~ Schooling + Year + Status + s(BMI) + s(Alcohol) +
##      Total.expenditure + Diphtheria + s(HIV.AIDS) + s(thinness.5.9.years) +
##      Income.composition.of.resources
##      Resid. Df Resid. Dev      Df Deviance  Pr(>Chi)
## 1      1647      60009
## 2      1646      59985 1.0000      24.5  0.149053
## 3      1645      59160 1.0000     824.7 < 2.2e-16 ***
## 4      1641      51395 4.0001    7765.8 < 2.2e-16 ***
## 5      1637      48722 4.0001    2672.9 < 2.2e-16 ***
## 6      1636      48597 1.0000     125.1  0.001098 **
## 7      1635      47673 1.0000     923.1 < 2.2e-16 ***
## 8      1631      22994 4.0002   24679.9 < 2.2e-16 ***
## 9      1627      21220 3.9998    1773.8 < 2.2e-16 ***
## 10     1626      19099 1.0000    2120.8 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
set.seed(1)
trainid <- sample(1:nrow(life_exp), nrow(life_exp)/2)
train <- life_exp[trainid,]
test <- life_exp[-trainid,]
```

*#according to the ANOVA test, we find the gam6 has a low p-value, to keep the model easier to grasp, we*

*# get the test error by VS approach*

```
gam.fit.train=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+ s(Alcohol)+Diphtheria+s(HIV.AIDS)+ s(thinness.5.9.years)+
Income.composition.of.resources)
gam.final.pred = predict(gam.fit.train, test)
(gam.error = mean((test$Life.expectancy - gam.final.pred)^2))
```

```
## [1] 12.7334
```

*# fit the model with 9 predictors:*

```
gam.final.fit=gam(Life.expectancy~Schooling+Year+Status+s(BMI)+ s(Alcohol)+Diphtheria+s(HIV.AIDS)+ s(thinness.5.9.years)+
Income.composition.of.resources)
summary(gam.final.fit)
```

```
##
## Call: gam(formula = Life.expectancy ~ Schooling + Year + Status + s(BMI) +
##      s(Alcohol) + Diphtheria + s(HIV.AIDS) + s(thinness.5.9.years) +
##      Income.composition.of.resources)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -14.707  -1.973   0.102   1.948  12.569
##
## (Dispersion Parameter for gaussian family taken to be 11.7727)
##
##      Null Deviance: 127529.3 on 1648 degrees of freedom
## Residual Deviance: 19154.26 on 1627 degrees of freedom
## AIC: 8769.594
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##
##      Df Sum Sq Mean Sq F value    Pr(>F)
## Schooling      1  50254    50254 4268.698 < 2.2e-16 ***
## Year            1    259      259   22.028 2.912e-06 ***
```

```
## Status          1   1186   1186  100.731 < 2.2e-16 ***
## s(BMI)          1   2424   2424  205.879 < 2.2e-16 ***
## s(Alcohol)      1    903    903   76.690 < 2.2e-16 ***
## Diphtheria      1    715    715   60.697 1.178e-14 ***
## s(HIV.AIDS)     1  23362  23362 1984.455 < 2.2e-16 ***
## s(thinness.5.9.years) 1    364    364   30.961 3.073e-08 ***
## Income.composition.of.resources 1   2399   2399  203.785 < 2.2e-16 ***
## Residuals      1627  19154    12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##
##              Npar Df  Npar F      Pr(F)
## (Intercept)
## Schooling
## Year
## Status
## s(BMI)          3   15.117 1.048e-09 ***
## s(Alcohol)      3   26.929 < 2.2e-16 ***
## Diphtheria
## s(HIV.AIDS)     3  107.825 < 2.2e-16 ***
## s(thinness.5.9.years) 3   55.714 < 2.2e-16 ***
## Income.composition.of.resources
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The MSE of those methods are pretty close, while GAM has the best performance.

```
err.all <- c(err.lm, loocv.err.OLS, err.ridge, err.lasso, err.pcr, err.pls, gam.error)
names(err.all) <- c("OLS-VS", "OLS-L00CV", "ridge", "lasso", "pcr", "pls", "gam-VS")
barplot(err.all)
```

