IECEE Midterm2

(1) Iris Classification

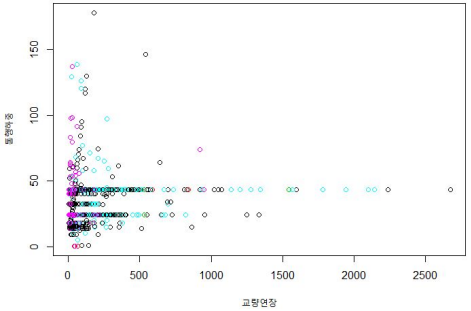
(2) Environmental Engineering

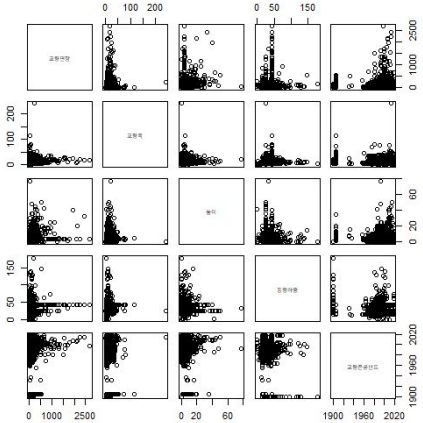
2014-13567 Daeun Kim

1.0 Introduction

Had trouble finding appropriate CEE dataset for classification.

(1) National Bridge Standard Data (공공데이터포털, 2018)
Reponse: Upper structure type(Slab, Girder, Rahmen, Truss, Cable, Arch, Suspension)
Predictors: Length, Width, Height, Weight, Year





Could not find patterns between predictors and response variable.

(2) Single Person Households
Reponse: Type of single person households
Predictors possible: Age, Income

이어서 변미리 외(2008)는 서울의 1인가구의 현황과 주거 특성에 대해 살펴보고 1인가구 밀집지역의 공간적 특성과 관련한 변수를 사용하여 '산업예비군, 골드세대, 불안한 독신자, 실버세대'의 4가지 유형으로 구분하였다. 김재형(2008)은

Only statistical datas exist. Individual raw dataset is not open to public due to privacy issue.

1.0 Introduction

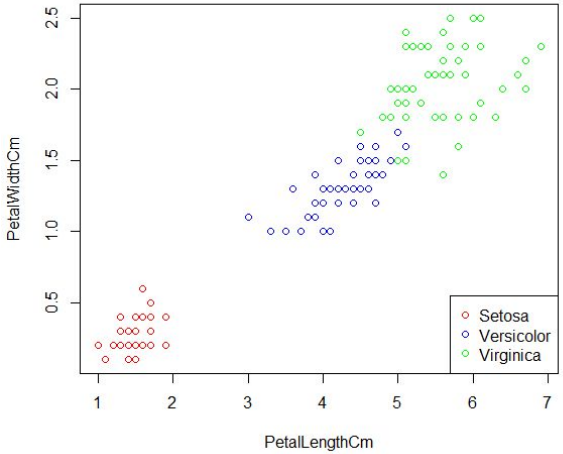
Choose famous dataset “iris” to understand and practice classification methods using R.

Purpose: Classify the species of iris using Petal size data.

Response Y (Species): 3 categories of iris (Setosa, Versicolor, Virginica), sepal and petal size measured (Ronald Fisher, 1936)

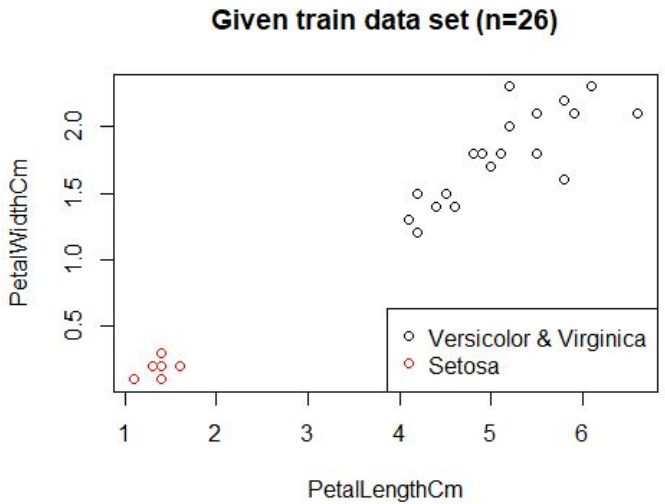
Predictor X: 2 quantitative variables, 150 datas

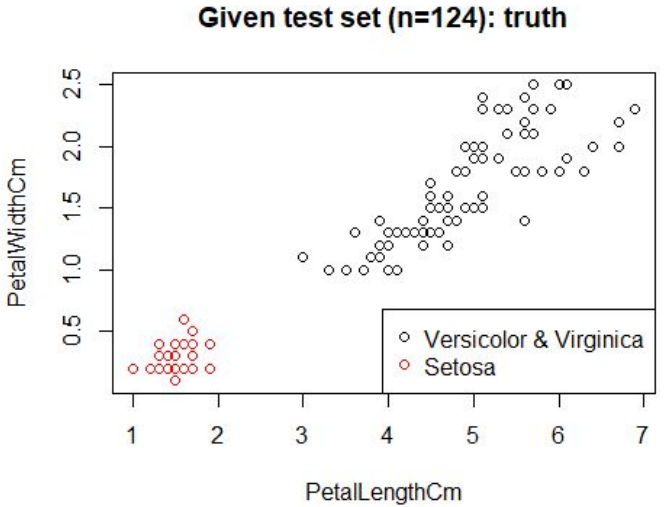
(1) Petal Width (cm)
(2) Petal Length (cm)



1.1 Logistic Regression

Consider the case which one wants to divide the vehicle into 2 groups (Setosa & Versicolor, Virginica) from given test data set:



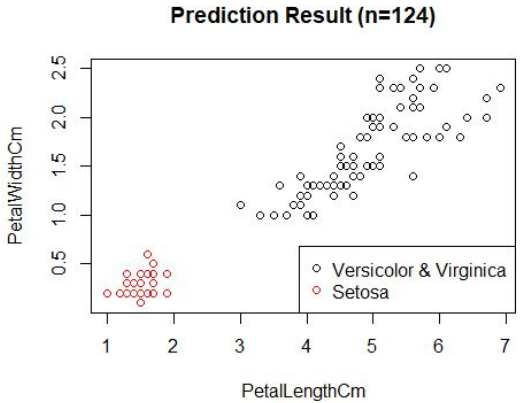


1.1 Logistic Regression

Classification result of Logistic Regression (Accuracy: 100%)

```
#1.Logistic Regression
iris$Type=0
iris$Type[iris$Species=='Iris-setosa']= 1

train=(SepalWidthCm==3)
iris.train=iris[train,]
iris.pred=iris[!train,]
Type.pred=Type[!train]
plot(iris.train$PetalLengthCm, iris.train$PetalWidthCm, main="Given train data set (n=26)",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=c("red","black","black")[iris.train$Species], pch=1)
legend("bottomright", legend=c("Versicolor & Virginica", "Setosa"), col=1:2, pch=1)
plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="Given test set (n=124): truth",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=c("red","black","black")[iris.pred$Species], pch=1)
legend("bottomright", legend=c("Versicolor & Virginica", "Setosa"), col=1:2, pch=1)

glm.fit=glm(Type~PetalLengthCm+PetalWidthCm, data=iris.train, family=binomial, control=list(maxit=50))
glm.prob=predict(glm.fit,iris.pred,type="response")
glm.pred=rep(0,length(iris.pred$Type))
glm.pred[glm.prob>0.5]=1
glm.pred=factor(glm.pred, levels = c("0","1"))

plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="Prediction Result (n=124)",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=glm.pred, pch=1)
legend("bottomright", legend=c("Versicolor & Virginica", "Setosa"), col=1:2, pch=1)

mean(glm.pred==Type.pred)
table(glm.pred,Type.pred)
```

```
          Type.pred
glm.pred  0  1
       0 80  0
       1  0 44
```



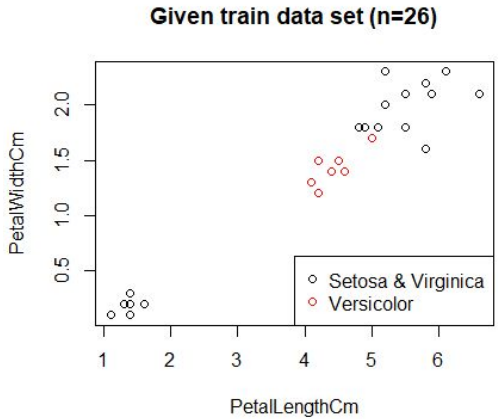

```
install.packages("ROCR")
library(ROCR)
install.packages("arulesViz")
library(arulesViz)

glm.rocr = prediction(glm.prob, Type.pred)
glm.roc = performance(glm.rocr, "tpr", "fpr")
plot(glm.roc, col=1, lty=1, lwd=2)

glm.auc=performance(glm.rocr, "auc")
glm.auc=glm.auc@y.values[[1]]
```

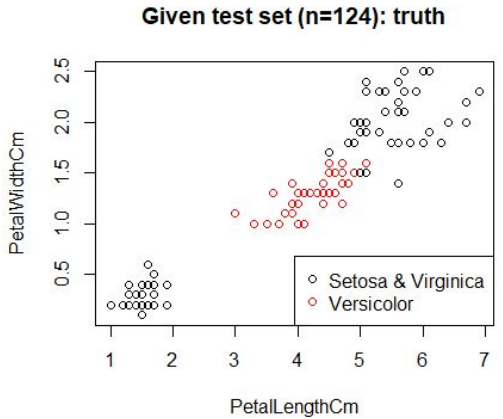
AUC = 1.0

1.2 Comparison between classification methods

Suppose we wish to separate only Versicolor, compare Logistic regression, LDA, QDA, KNN

```
#Logistic Regression
glm.fit=glm(Type~PetalLengthCm+PetalWidthCm, data=iris.train, family=binomial, control=list(maxit=50))
glm.prob=predict(glm.fit,iris.pred,type="response")
glm.pred=rep(0,length(iris.pred$Type))
glm.pred[glm.prob>0.5]=1
glm.pred=factor(glm.pred, levels = c("0","1"))

plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="Logistic Regression",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=glm.pred, pch=1)
legend("bottomright", legend=c("Setosa & Virginica", "Versicolor"), col=1:2, pch=1)
mean(glm.pred==Type.pred)
table(glm.pred,Type.pred)

#LDA and QDA
lda.fit=lda(Type~PetalLengthCm+PetalWidthCm, data=iris.train)
qda.fit=qda(Type~PetalLengthCm+PetalWidthCm, data=iris.train)
lda.class=predict(lda.fit, iris.pred)$class
qda.class=predict(qda.fit, iris.pred)$class

plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="LDA",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=lda.class, pch=1)
legend("bottomright", legend=c("Setosa & Virginica", "Versicolor"), col=1:2, pch=1)
mean(lda.class==Type.pred)

plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="QDA",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=qda.class, pch=1)
legend("bottomright", legend=c("Setosa & Virginica", "Versicolor"), col=1:2, pch=1)
mean(qda.class==Type.pred)

#KNN
library(class)
train.x=cbind(nPetalLengthCm,nPetalWidthCm)[train,]
test.x=cbind(nPetalLengthCm,nPetalWidthCm)[!train,]
Type.train=iris$Type[train]
knn.pred=knn(train.x,test.x,Type.train,k=5,prob=TRUE)
plot(iris.pred$nPetalLengthCm, iris.pred$nPetalWidthCm, main="KNN (n=5)",
     xlab="nPetalLengthCm", ylab="nPetalWidthCm",col=knn.pred, pch=1)
legend("bottomright", legend=c("Setosa & Virginica", "Versicolor"), col=1:2, pch=1)
mean(knn.pred==Type.pred)
table(knn.pred,Type.pred)
```

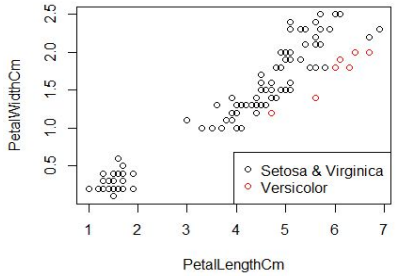




1.2 Comparison between classification methods

Suppose we wish to separate only Versicolor, compare Logistic regression, LDA, QDA, KNN

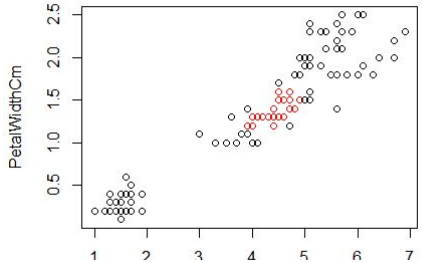
Logistic Regression



62.1%
AUC: 0.77

```
          Type.pred
glm.pred  0  1
       0 76 41
       1  6  1
```

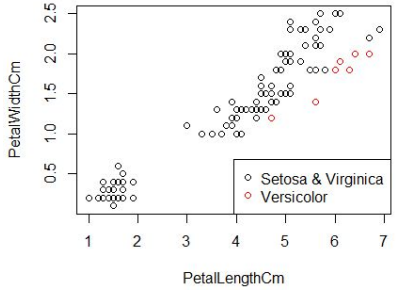
QDA



87.9%
AUC: 0.97

```
          Type.pred
qda.class  0  1
       0 82 15
       1  0 27
```

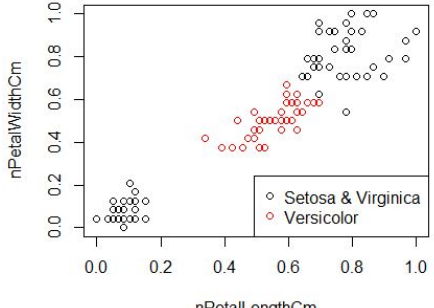
LDA



62.1%
AUC: 0.77

```
          Type.pred
lda.class  0  1
       0 76 41
       1  6  1
```

KNN (n=5)

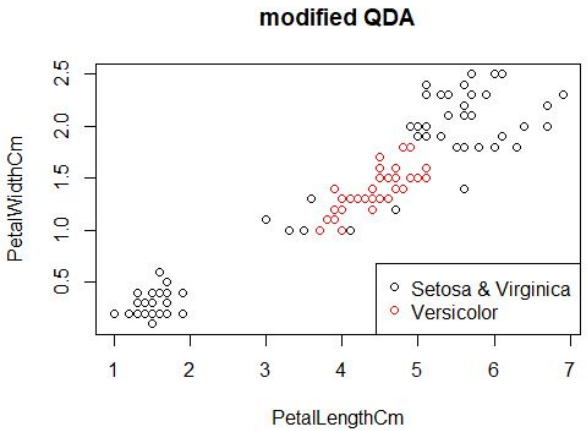


96.0%
AUC: 0.99

```
          Type.pred
knn.pred  0  1
       0 79  2
       1  3 40
```

1.2 Comparison between classification methods

```
#modified QDA
mqda.class=(predict(qda.fit, iris.pred)$posterior[,2]>0.1)+1
plot(iris.pred$PetalLengthCm, iris.pred$PetalWidthCm, main="modified QDA",
     xlab="PetalLengthCm", ylab="PetalWidthCm",col=mqda.class, pch=1)
legend("bottomright", legend=c("Setosa & Virginica", "Versicolor"), col=1:2, pch=1)
mean(mqda.class==Type.pred)
```

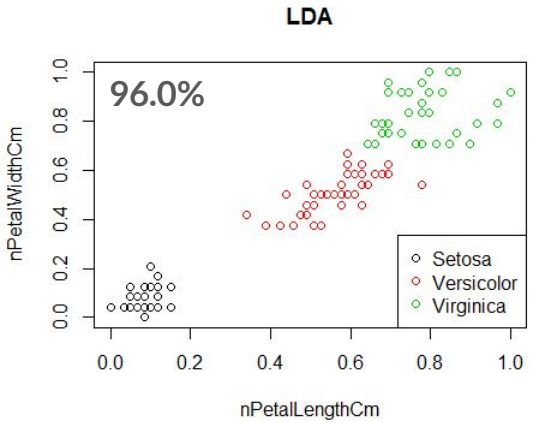


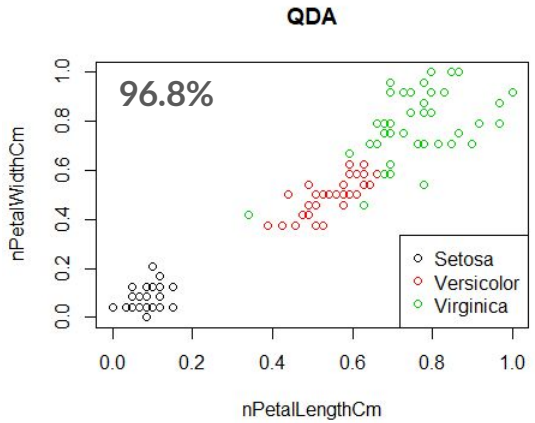
89.5%

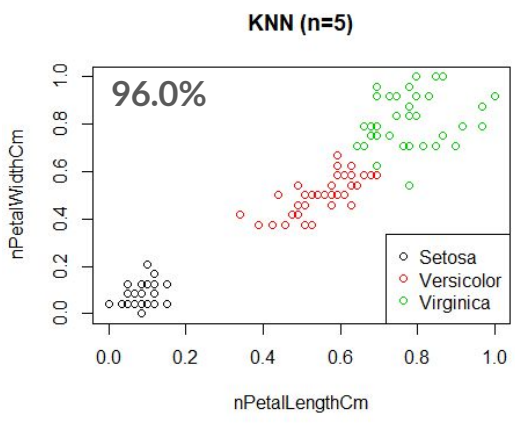
```
             Type.pred
mqda.class  0  1
     FALSE 77  8
     TRUE   5 34
```

1.3 Multi class classification (LDA, QDA, KNN)

Suppose we wish to separate all 3 types of Iris:
Total number of data: 150 / Train set: 26 / Test set: 124







lda.class	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	44	0	0
Iris-versicolor	0	41	4
Iris-virginica	0	1	34

qda.class	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	44	0	0
Iris-versicolor	0	38	0
Iris-virginica	0	4	38

knn.pred	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	44	0	0
Iris-versicolor	0	40	3
Iris-virginica	0	2	35

1.4 Resampling (10-fold CV)

Estimate accuracy of QDA model using 10-fold cross validation

```
#Resampling (k-fold CV)
install.packages("crossval")
library(crossval)

predfun.qda = function(train.x, train.y, test.x, test.y, negative)
{
        qda.fit=qda(train.x, grouping = train.y)
        ynew = predict(qda.fit, test.x)$class
        out = confusionMatrix(test.y, ynew, negative = negative)
        return(out)
}

set.seed(5)
X = as.matrix(iris[,4:5])
Y = iris[,6]

cv.out = crossval(predfun.qda, X, Y, K=10, B=1, negative='No')
cv.out$stat
diagnosticErrors(cv.out$stat)
```

```
Round # 1 of 1
CV Fit # 1 of 10
CV Fit # 2 of 10
CV Fit # 3 of 10
CV Fit # 4 of 10
CV Fit # 5 of 10
CV Fit # 6 of 10
CV Fit # 7 of 10
CV Fit # 8 of 10
CV Fit # 9 of 10
CV Fit # 10 of 10
> cv.out$stat
FP TP TN FN
 0 15  0  0
> diagnosticErrors(cv.out$stat)
 acc sens spec  ppv  npv  lor
   1    1  NaN    1  NaN  NaN
```

```
set.seed(10)
cv.error.10=rep(0,10)
for (i in 1:10){
        cv.error.10[i]=knn.cv(train.x,species.train,k=i,prob=TRUE)
}

knn.cv(train.x,species.train,k=10, prob=TRUE)
knn.cv(train.x,species.train,k=3, prob=TRUE)
```

1.5 Conclusion

- When boundary is clear, logistic regression perform well
  Accuracy: Logistic Regression(100%)

- When boundary is highly curved, only QDA or KNN can perform well.
  Accuracy: Logistic Regression (62.1%), LDA (62.1%), QDA (87.9%), KNN (96.0%)

- By modifying QDA model, the accuracy increased(87.9% -> 89.5%)

- In multi-class problem, every model show great performance.
  Accuracy: LDA (96.0%), QDA (96.8%), KNN (96.0%)

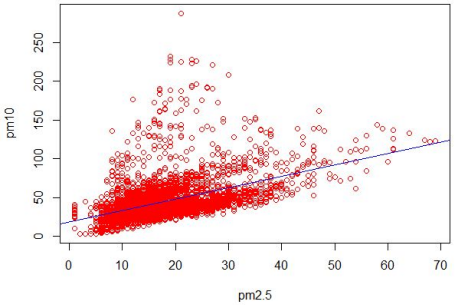
- QDA model is reliable for multi-class problem.
  Using 10-fold CV, AUC: 1.00

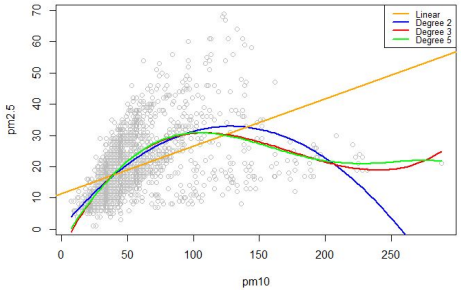
2.0 Review

Purpose: In order to control air quality, determine which variables affect the concentration of $PM_{2.5}$

Response Y (pm2.5): Hourly average concentration of $PM_{2.5}$ measured at 160 stations in Seoul during 2019-06-02 to 2019-06-09(4200 datas). *Day time data selected (07:00-18:00)

Predictor X: 8 variables below (including one qualitative variable*)






```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.028e+00  5.920e-01   5.116 3.43e-07 ***
pm10        7.510e-01  1.126e-02  66.720  < 2e-16 ***
time       -1.150e-03  1.927e-04  -5.966 2.88e-09 ***
no2        -1.364e+02  1.369e+01  -9.967  < 2e-16 ***
o3         -6.614e+01  8.296e+00  -7.973 2.62e-15 ***
co          2.951e+00  1.009e+00   2.924  0.00349 **
so2         2.479e+02  8.990e+01   2.758  0.00587 **
I(pm10^3)  -5.394e-06  9.367e-07  -5.759 9.80e-09 ***
```

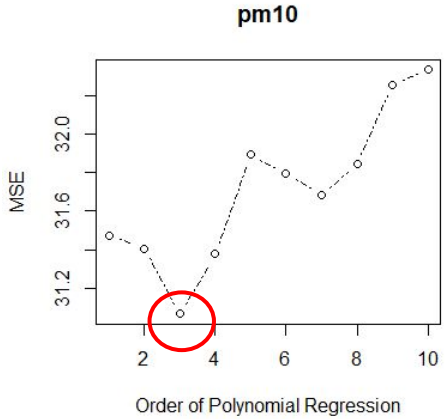
```
> vif(lm.fit)
     pm10      time       no2        o3        co
 4.128413  1.278182  2.176136  1.608818  1.732097
      so2 I(pm10^3)
 1.104499  3.344265
```

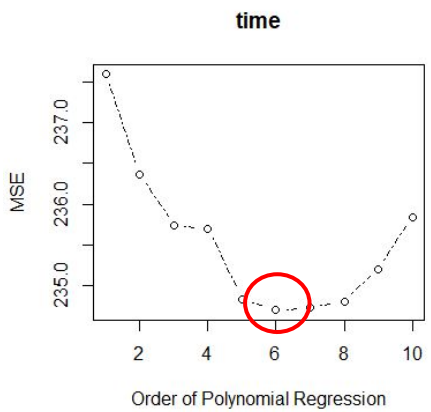
pm2.5
= 3.028 + 0.7510*__pm10__ - 0.00115*__time__- 136.4*__no2__
-66.14*__o3__ + 2.951*__co__ + 247.9*__so2__ - 0.000005394*__pm10^3__

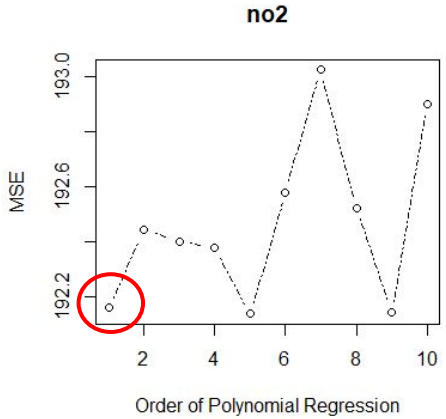
2.1 Resampling (10-fold CV)

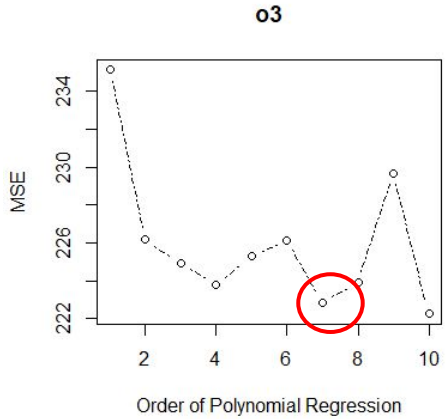
Find best degree of polynomial regression by 10-fold cross validation.

```r
#Resampling methods(10-fold CV)
set.seed(10)
cv.error_k10.10=rep(0,10)
for (i in 1:10){
  glm.fit=glm(pm2.5~poly(pm10,i),data=pro1.day)
  cv.error_k10.10[i]=cv.glm(pro1.day,glm.fit,K=10)$delta[1]
}
plot(1:10, cv.error_k10.10, type='b',lty='dotdash',pch=1,
     main='pm10', ylab='MSE', xlab='Order of Polynomial Regression')
```









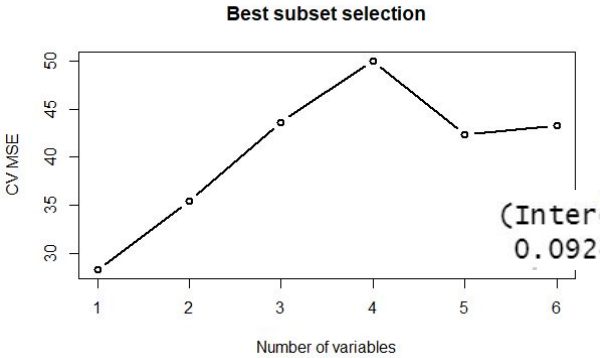
2.2 Model selection and regularization (n≅p)

Group dataset by district. ((1)Gwanak-gu: n=76, p=6 / (2)Guro-gu: n=60, p=6 / (3)full: n=1968, p=6)
Try model selection and regularization using **Best subset selection, Ridge regression, Lasso, PCR and PLS.**

Model selection for Guro-gu dataset(n=60, p=6) using Best subset selection

```
k=5
folds=sample(1:k, nrow(air2_subset),replace=TRUE)
cv.errors = matrix(NA,k,n_pa,dimnames = list(NULL,paste(1:n_pa)))

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
}

install.packages("leaps")
library(leaps)

for(j in 1:k){
  best.fit = regsubsets(pm2.5~.,data=air2_subset[folds!=j,],nvmax=n_pa)
  for(i in 1:n_pa){
    pred=predict(best.fit,air2_subset[folds==j,],id=i)
    cv.errors[j,i]=mean((air2_subset$pm2.5[folds==j]-pred)^2)
    }
  }

mean.cv.errors=apply(cv.errors,2,mean)
```

The linear model works better when the number of variable is small (1 variable is best)
-> The number of data is not much larger than the number of variables



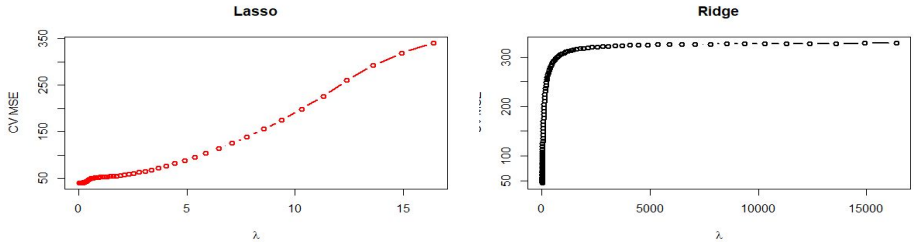

```
(Intercept)        pm10
 0.09241762  0.74987155
```

2.2 Model selection and regularization (n≅p)

Regularization for Guro-gu dataset(n=60, p=6) using Ridge, Lasso, PCR, and PLS.

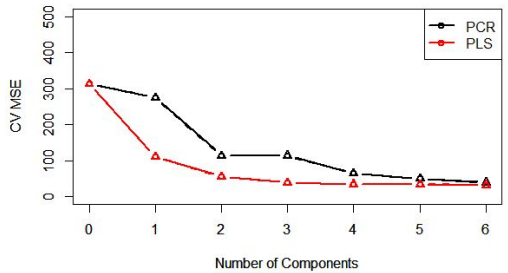
```
#Ridge regression & The Lasso
install.packages("glmnet")
library(glmnet)

#Ridge
grid=10^seq(10,-2,length=100)
ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid,thresh = 1e-12)

cv.out.ridge=cv.glmnet(x[train,],y[train],alpha=0)
bestlam=cv.out.ridge$lambda.min

out.ridge = glmnet(x,y,alpha=0)
predict(out.ridge,type="coefficients",s=bestlam)[1:(n_pa+1),]

#Lasso
grid=10^seq(10,-2,length=100)
lasso.mod=glmnet(x[train,],y[train],alpha=1,lambda=grid,thresh = 1e-12)

cv.out.lasso=cv.glmnet(x[train,],y[train],alpha=1)
bestlam=cv.out.lasso$lambda.min

out.lasso = glmnet(x,y,alpha=1)
predict(out.lasso,type="coefficients",s=bestlam)[1:(n_pa+1),]

x = cv.out.ridge$lambda
y = cv.out.ridge$cvm
xx = cv.out.lasso$lambda
yy = cv.out.lasso$cvm

plot(x,y,type='b',col='black',xlab=expression(lambda),ylab="CV MSE",lwd=2, ma
plot(xx,yy,type='b',col='red',xlab=expression(lambda),ylab="CV MSE",lwd=2, ma
```

```
#PCR & PLS
install.packages("pls")
library(pls)

x=model.matrix(pm2.5~.,air2)[,-1]
y=air2$pm2.5
set.seed(9)
train = sample(1:nrow(x),nrow(x)*2/3)
test=(-train)
y.test=y[test]

pcr.fit=pcr(air2$pm2.5~.,data=air2,subset=train,scale=TRUE,validation="CV")
pls.fit=plsr(air2$pm2.5~.,data=air2,subset=train,scale=TRUE,validation="CV")

summary(pcr.fit)
validationplot(pcr.fit,val.type = "MSEP",ylab='CV MSE',,type='b',
               ylim=c(0,500),xlab='Number of Components',col='black',lwd=2, main="")
par(new=TRUE)
summary(pls.fit)
validationplot(pls.fit,val.type = "MSEP",ylab='CV MSE',,type='b',
               ylim=c(0,500),xlab='Number of Components',col='red',lwd=2, main="")
legend("topright",legend=c("PCR","PLS"),col=c("black","red"),pch=1,lwd=2)
```



The coefficients for full data set

```
> predict(out.ridge,type="coefficients",s=bestlam)[1:(n_pa+1),]
  (Intercept)          time           no2           o3           co          so2          pm10
 8.651493e+00  1.257776e-03 -1.691225e+02 -3.877942e+01  1.333508e+01 -1.602461e+03  6.958768e-01
> predict(out.lasso,type="coefficients",s=bestlam)[1:(n_pa+1),]
(Intercept)        time          no2          o3          co          so2        pm10
  2.2252578   0.0000000 -47.4353226   0.0000000   0.0000000   0.0000000   0.7218732
```



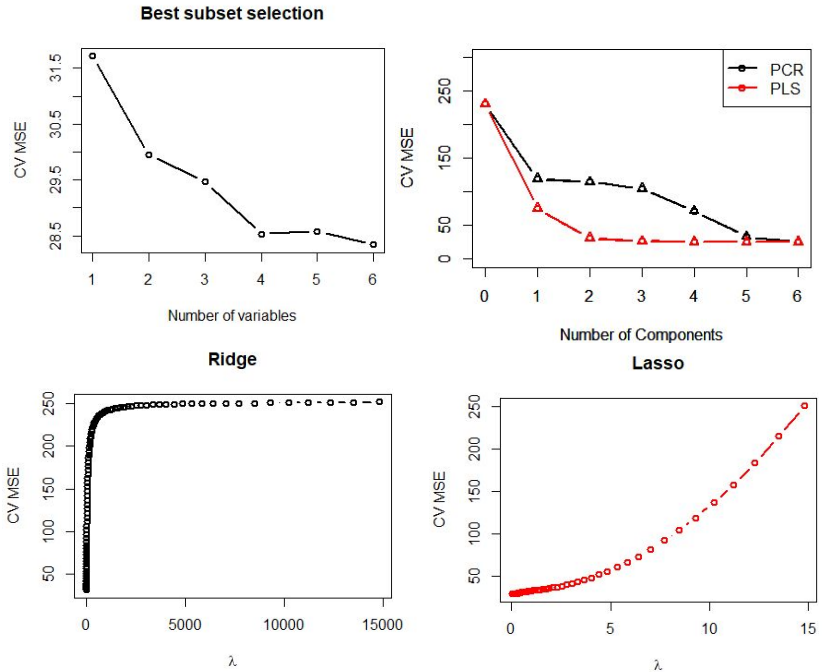
Test MSE for the linear regression methods

	Best Subset Selection	Ridge Regression	The Lasso	PCR	PLS
Model Complexity	1	6	2	6	4
Test MSE	66.06	70.45	67.38	72.93	106.64

2.2 Model selection and regularization (n>>p)

Model selection and regularization for full dataset(n= 1968, p=6).

Trained with 1312 data and tested with 656 data.



The coefficients for full data set

	Inter.	time	no2	o3	co	so2	pm10
Best Subset Selection	3.92	-1.15e-03	-1.33e+02	-6.63e+01	3.10	3.10e+02	7.00e-01
Ridge Regression	3.21	-9.42e-04	-5.56e+01	-4.38e+01	6.45	2.27e+02	6.06e-01
The Lasso	3.88	-1.13e-02	-1.24e+02	-6.25e+01	2.79	2.78e+02	6.98e-01

Test MSE for the linear regression methods

	Best Subset Selection	Ridge Regression	The Lasso	PCR	PLS
Model Complexity	6	6	6	6	3
Test MSE	28.91	28.89	28.87	36.90	33.33

2.3 Conclusion

- Find best degree of polynomial regression by 10-fold cross validation.
  Linear model shows least MSE when $D_{pm10} = 3$, $D_{time} = 6$, $D_{no2} = 1$, $D_{o3} = 7$.
- Best subset selection model shows lowest test MSE.
- For full data, where the number of data is sufficient when compared to the variables, the enhancement of prediction performance was not as prominent as district selected data.
- Even for Guro-gu data, the enhancement of prediction performance was not clear. Guess the number of data is quite large compared to the variables.