Determinants of Mortality

Huang Fang, Wenyu Li, Meng Li

Abstract

The objective of the project is to construct a regression model to relate total age-adjusted population mortality to precipitation, school years, nonwhite population percentage, low-income percentage of households, relative pollution potential of oxides of nitrogen and sulphur dioxide. Methods of stepwise regression, ridge regression, lasso and partial least squares regression are applied to fit different models which are compared by cross-validation method to select the best model. It is concluded that ridge regression model is the best in terms of capacity of prediciton, which displays linear association between mortality and precipatation, school years, nonwhite population and pollution. It is worth noting that pollution plays an important role in prediction of mortality.

1 Introduction

The mortalities of different factors differ dramatically. Reducing mortality is a core challenge for researchers and what effects take place grasps a huge amount of attentions. In this thesis, our goal is to relate the mortality to a large scale of factors, including PRECIP, EDUC and another five factors. We are also interested in the relation between the pollution and the mortality. The results can help researchers understand the factors that cause mortality and have the maximum possible opportunity to reduce the mortality in practical experience.

2 Data Description

The dataset we study here has eight variables, namely, MORTALITY, PRECIP, EDUC, NONWHITE, POOR, NOX, SO2 and CITY. The variable MORTALITY here is the response variable, all other variables are the predictable variables. Below is a more specific description of each variable.

- PRECIP: Mean annual precipitation (in inches)
- EDUC: Median number of school years completed by persons of age 25 or over
- NONWHITE: Percentage of population in 1960 that is nonwhite
- POOR: Percentage of households with annual income under $3000 in 1960
- NOX: Relative pollution potential of oxides of nitrogen
- SO2: Relative pollution potential of sulphur dioxide
- MORTALITY : Total age-adjusted mortality from all causes, in deaths per 100,000 population
- CITY: Different cities

3 Preliminary investigation

After observing the dataset, we found the variable "CITY" is relatively useless for our analysis, it has a different level for each observation. So it is reasonable to drop it from our dataset. Now all the remaining variables "MORTALITY", "PRECIP", "EDUC", "NONWHITE", "POOR", "NOX" and "SO2" can be considered as continuous variables.

The histograms (Figure 1) show us the distributions of each variable. From the figure, we can see the variables "MORTALITY", "NONWHITE", "POOR", "NOX" and "SO2" are right skewed. So we consider applying logarithm transformation to these variables. After applying transformation, we got the histograms (Figure 2). Now we can see the distributions of these variables tend to be normal, which means our transformation is appropriate.

Since we are going to apply Ridge and Lasso methods, it is necessary for us to standardize all the variables. From the matrix plot and the correlation matrix(Figure 3), we found that variables "NOX" and "SO2" seem to have linear relationship, which means the effect of multicollinearity possibly presents. Furthermore, we found that the largest VIF is 3.2069 by applying first order linear regression. So the effect of muticollinearity of our data is moderate.

In order to proceed to the next step, it's also important to check if second-order terms and interaction terms are negligible. Applying the first-order regression, from the residuals vs. interaction terms plot(Figure 4), we found each regression line is basically horizontal line coinciding with x axis, which proves that the quadratic terms and interaction terms are not significant.

4 Methods and Results

4.1 Stepwise Regression

Firstly, we want to fit our full model which includes all the first-order terms, then drop the terms stepwisely according to the AIC criterion.

Through this method, we dropped the variable "POOR". According to the summary table of final model, all the variables in our final model are significant now.

Variable	Estimate	Std.Error	t value	P-value
PRECIP	0.35771	0.09479	3.774	0.00039
EDUC	-0.20252	0.09430	-2.148	0.03608
NONWHITE	0.41840	0.07932	5.275	2.22e-06
SO2	0.38146	0.08307	4.592	2.53e-05

Our final model is:

$$(\ln MOTALITY)^* = 0.358 PRECIP^* - 0.203 EDUC^* + 0.418(\ln NONWHITE)^* + 0.381(\ln SO2)^*$$

Note that $(\ln MOTALITY)^*$ stands for the standardized $(\ln MOTALITY)$, etc.

After observing the figures(Figure 5), we can find that there is no obvious nonlinear pattern in the residuals vs. fitted values plot, which implies the model is appropriate. Meanwhile, the distributions of residuals under different fitted values are approximately

the same, which supports the equal variance assumption. From the Q-Q plot, we can see the distribution of residuals is sightly heavy-tailed. Since the effect of heavy-tail is not severe, we can not reject the assumption of normality.

4.2 Ridge

First, we used the GCV criterion to choose the best penalty parameter k(Figure 6). The estimate is $\hat{k}_{optimal} = 7.69$. After plugging in the optimal penalty parameter, we got the estimation of $hat\beta(k)$. Then we set k = 1 (a value quite greater than 0 and smaller than $\hat{k}_{optimal}$) and calculated the value of $\hat{\sigma}^2$, it is easy to get $\hat{\sigma}^2 = 0.3486719$. Now we can calculate the relate estimation of each β_i and their corresponding standard error and t-value.

Variable	Estimate	Std.Error	t value
PRECIP	0.3563	0.0796	4.4646
EDUC	-0.2105	0.0777	-2.7071
NONWHITE	0.3392	0.0759	4.4578
POOR	0.02430	0.0811	0.3075
NOX	0.1540	0.0833	1.8569
SO2	0.2440	0.0823	2.9633

According to the summary table, the variable "POOR" seems to be insignificant, so we decide to drop it from our model.

Our final ridge regression function is:

$$(\ln MOTALITY)^* = 0.3563PRECIP^* - 0.2105EDUC^* + 0.3392(\ln NONWHITE)^* + 0.154(\ln NOX)^* + 0.244(\ln SO2)^*$$

From the observed vs. fitted values plot, residuals vs. fitted values plot, histogram and Q-Q plot of residuals(Figure 7), we can see now our model fits the data well. Meanwhile, there is no obvious nonlinear pattern in the residuals vs. fitted values plot, we can say our model is adequate. At the same time, the histogram and Q-Q plot of residuals further support the normality assumption.

Comparing the value of VIFs with and without ridge regression.

	PRECIP	EDUC	NONWHITE	POOR	NOX	SO2
original VIF	1.9488	1.9257	1.6551	2.2741	3.2069	3.1948
VIF_ridge	1.3880	1.3210	1.2577	1.4466	1.5450	1.5057

Ridge Regression significantly decreased the value of VIF, and reduced the effect of multicollinearity.

4.3 Lasso

At first, we applied 10-fold cross-validation, which led to $\hat{\lambda} = 0.03651799$. And the CV criterion achieved its minimum(Figure 8). Further implementation of Lasso method with $\hat{\lambda}$ fitted the model to be:

$$\ln MORTALITY^* = 0.1869PRECIP^* - 0.1373EDUC^* + 0.2632\ln NONWHITE^* + 0.1712\ln SO_2{}^*$$

Noticed that the estimated beta coefficients for predictor POOR and NOX are zero, which helps to drop those variables and select other predictor variables to be regressed to MORTALITY.

Further diagnoses include observed response values against fitted values, residuals against fitted values, histogram and Q-Q plot of residuals(Figure 9). From the residuals, the fit of model is reasonable and no obvious nonlinear pattern exists. It suggests the adequacy of the model. Meanwhile, the residuals are roughly distributed evenly and consistently around the x-axis, we can conclude that the variances are consistent and the assumption of equality of variances is valid. Histogram and Q-Q plot exhibit that the residuals are distributed slightly heavy-tailed, which is not that significant. Therefore, the assumption of normality could be accepted.

4.4 Partial Least Square

First we considered fitting a PLS model with 6 components.

We can get the value of CV from Cross-validation table and calculate the R^2 and F_k for each k.

k	1	2	3	4	5	6
CV	0.6571	0.6541	0.6455	0.6577	0.6833	0.6875
R^2	0.6389	0.6664	0.6838	0.6843	0.6847	0.6847
F_k	102.6094	4.7032	3.0865	0.08542	0.06209	0.0075
$qf(0.95, 1, n-k-1)$	4.0069	4.0099	4.0130	4.0162	4.0195	4.0230

According to the Cross-validation method, we should choose k = 3 as the number of components. However, the F-tests indicate that k = 2 components are enough. After carefully comparing these two methods, we decided to set k = 3 because adding the third component significantly decreased the value of CV. And the value of F_3 is still close to $qf(0.95, 1, n-3-1)$, which means the third component is acceptable.

We found that about 75% of total variance can be explained by plotting the scores of the first 3 components(Figure 10), We also got the first 3 loadings:

	Comp1	Comp2	Comp3
PRECIP	0.4355	-0.2847	0.6957
EDUC	-0.5000	0.1988	0.1097
NONWHITE	0.4673	-0.0004	0.1472
POOR	0.4823	-0.6119	-0.1950
NOX	0.2110	0.6589	-0.5949
SO2	0.2998	0.7388	-0.3251

Under Partial Least Squares method, our final model is:

$$(\ln MOTALITY)^* = 0.4209 PRECIP^* - 0.2015 EDUC^* + 0.3725(\ln NONWHITE)^* - 0.0056(\ln POOR)^* + 0.1488(\ln NOX)^* + 0.2909(\ln SO2)^*$$

The observed vs. fitted values plot(Figure 11) indicates that our model fits the data well and is appropriate since there is no nonlinear pattern in the plot. Furthermore, the distribution of residuals is approximately normal, so the normality assumption holds.

5 Model Selection

Under Stepwise Regression, all these final models including Ridge, Lasso and Partial Least Square seem to be approperiate. However we need to choose the best model for our data. There are many criterion can be taken, like AIC, BIC, etc. Here we consider using Cross-validation to find the best model with the best predictive ability.

We applied 10-fold cross-validation to each method, and calculated the average prediction error under each method. (The algorithm and code are attached in the appendix)

$$CV^{(10)} = \frac{1}{n}\sum_{j=1}^{10}\sum_{i\in I_j}(Y_i - x_i^T\hat{\beta}^{(j)})^2$$

I_j denotes the indices of the jth group.

	Stepwise Regression	Ridge	Lasso	PLS
CV	0.4815050	0.2535218	0.5697742	0.3105891

The model we got under Ridge Regression has the strongest predictive ability. As a result, our final best model is the model fitted under ridge regression.

$$(\ln MOTALITY)^* = 0.3563PRECIP^* - 0.2105EDUC^* + 0.3392(\ln NONWHITE)^* + 0.154(\ln NOX)^* + 0.244(\ln SO2)^*$$

6 Conclusions and Discussion

The ridge regression model is the optimal among all the fitted models since it provides best prediction ability. Mortality of population is associated to precipitation, education years, nonwhite population percentage and pollution, where education years is negative related to mortality while others have positive relation with mortality. These foregoing results reveal population with higher education level tends to have lower mortality while higher precipitation, nonwhite populaiton percentage and severer pollution lead to higher mortality. It is noteworthy to mention that the influence of pollution on mortality is observed in the selected model, where SO2 is inclined to have more impact than NOX.

Although the ridge regression model is best in terms of prediciton, the limitations are also not negligible due to the existence of bias. Another limitation is the heavy-tailed distribution of residuals displayed by Q-Q plot.

7 Appendices

7.1 Figures

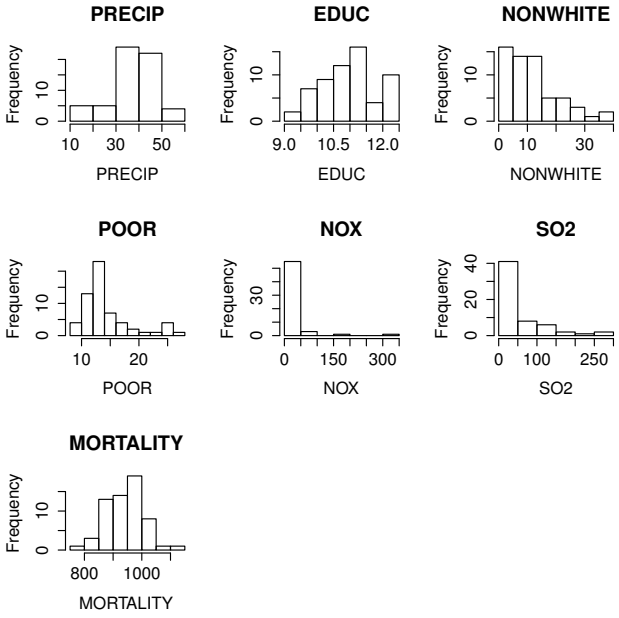

Figure 1

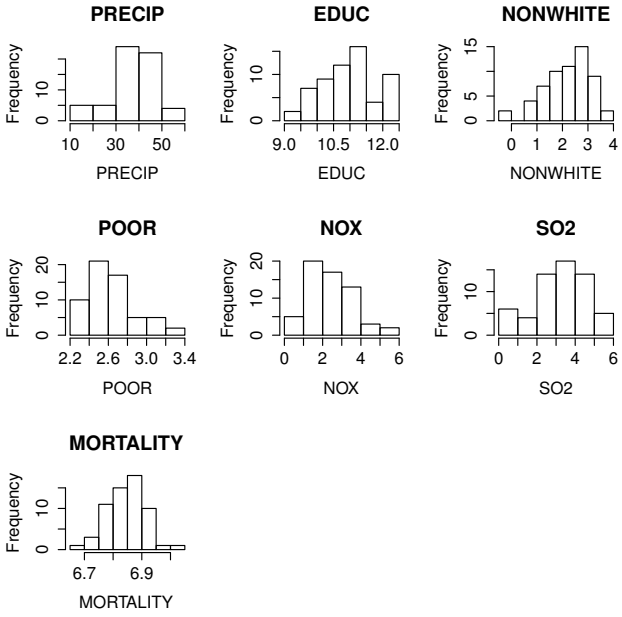

Figure 2

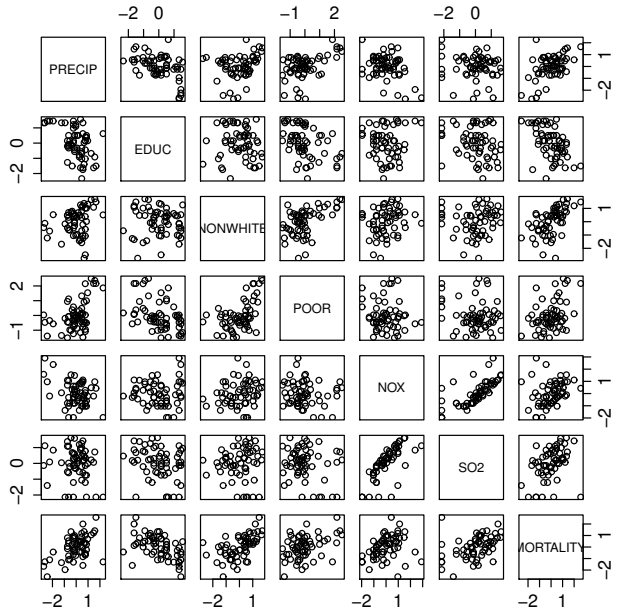

Figure 3

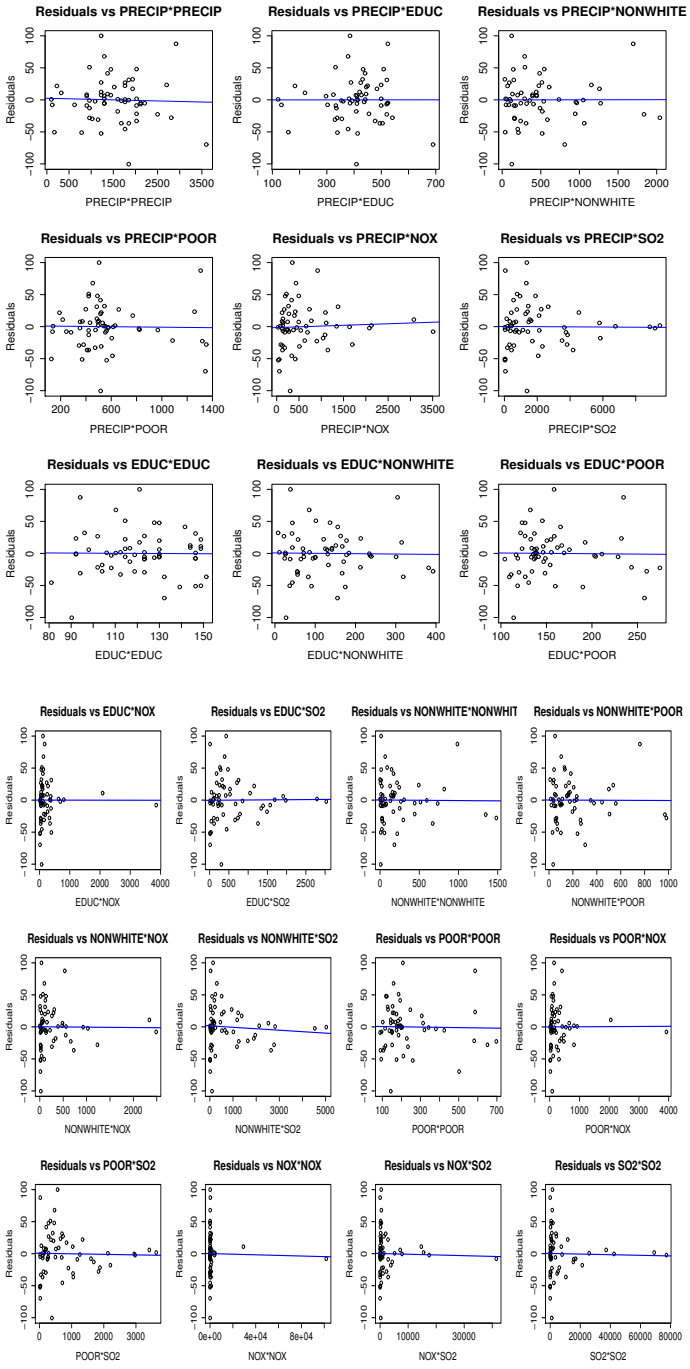
Residuals vs PRECIP*PRECIP
Residuals vs PRECIP*EDUC
Residuals vs PRECIP*NONWHITE
Residuals vs PRECIP*POOR
Residuals vs PRECIP*NOX
Residuals vs PRECIP*SO2
Residuals vs EDUC*EDUC
Residuals vs EDUC*NONWHITE
Residuals vs EDUC*POOR
Residuals vs EDUC*NOX
Residuals vs EDUC*SO2
Residuals vs NONWHITE*NONWHIT
Residuals vs NONWHITE*POOR
Residuals vs NONWHITE*NOX
Residuals vs NONWHITE*SO2
Residuals vs POOR*POOR
Residuals vs POOR*NOX
Residuals vs POOR*SO2
Residuals vs NOX*NOX
Residuals vs NOX*SO2
Residuals vs SO2*SO2
Residuals
PRECIP*PRECIP
PRECIP*EDUC
PRECIP*NONWHITE
PRECIP*POOR
PRECIP*NOX
PRECIP*SO2
EDUC*EDUC
EDUC*NONWHITE
EDUC*POOR
EDUC*NOX
EDUC*SO2
NONWHITE*NONWHITE
NONWHITE*POOR
NONWHITE*NOX
NONWHITE*SO2
POOR*POOR
POOR*NOX
POOR*SO2
NOX*NOX
NOX*SO2
SO2*SO2

Figure 4

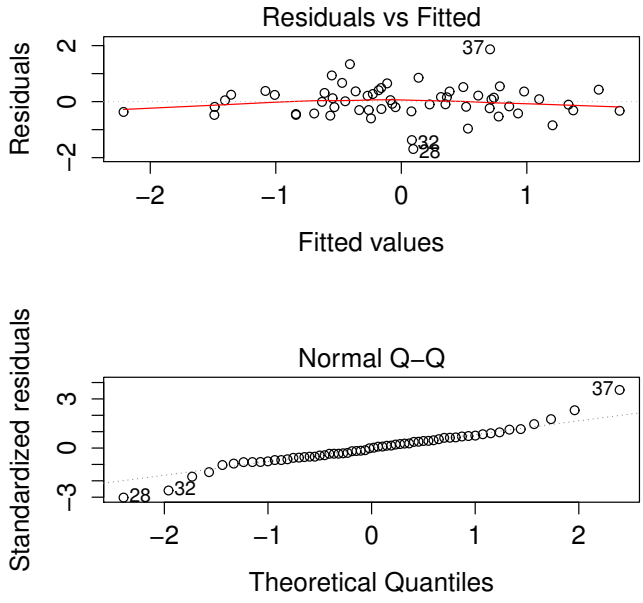

Figure 5

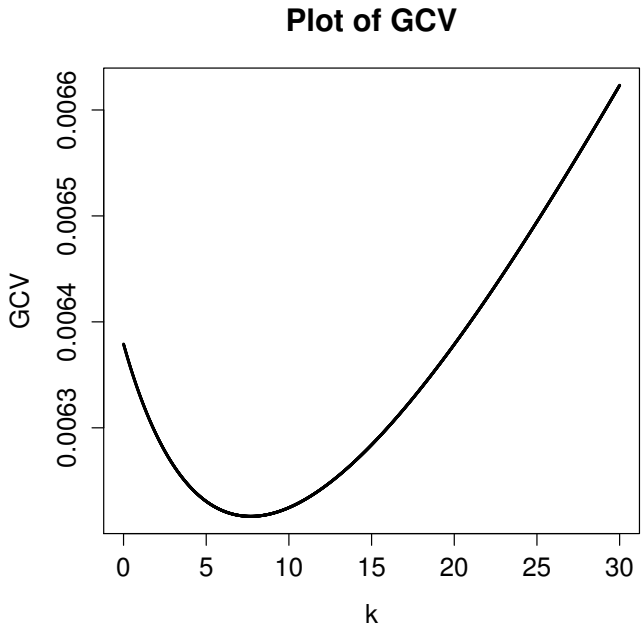

Figure 6

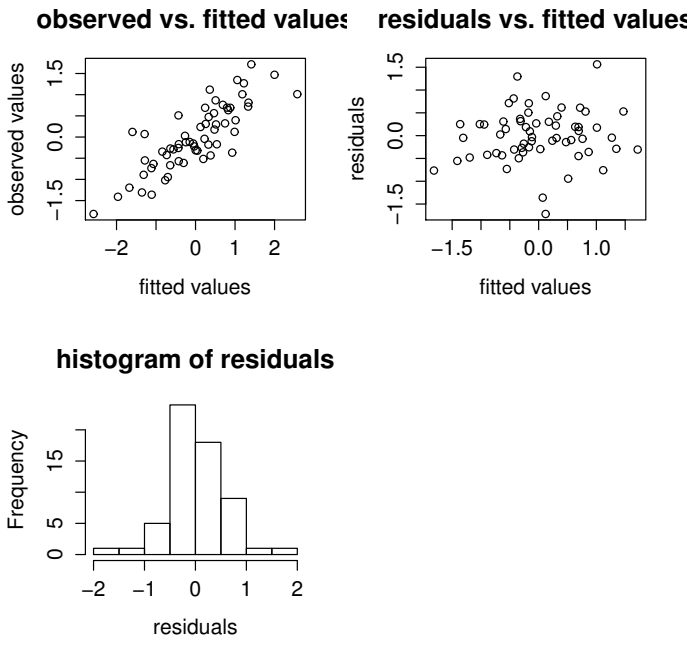

Figure 7

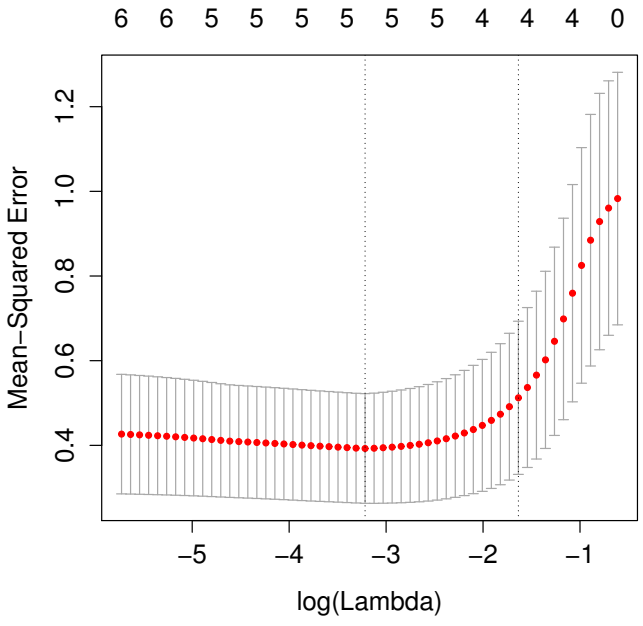

Figure 8

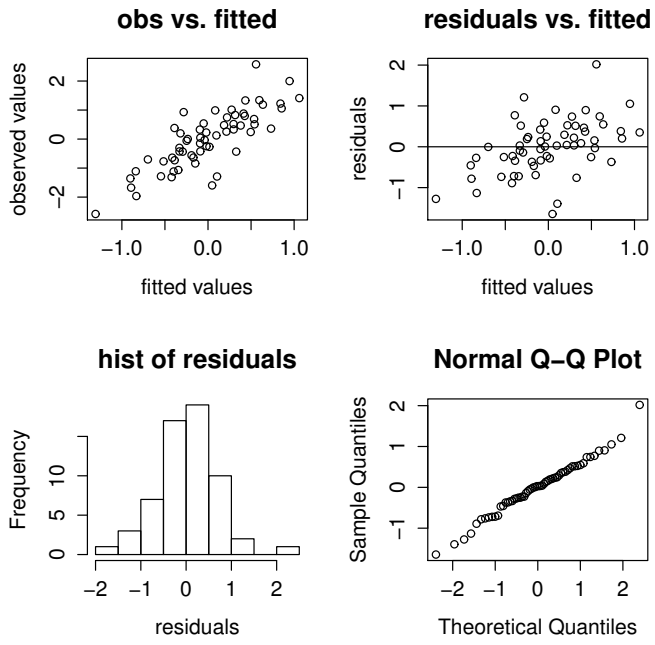

Figure 9

7.2 Codes and outputs

```
#Project
library(MASS)
library(DAAG)
library(glmnet)
library(pls)
#Preliminary Investigation
data = read.csv("~/academic/Sta207/project/mortality.csv", header = TRUE)
data = data[, 1:7]
sapply(data, class)
var = names(data)
par(mfrow = c(3, 3))
for(i in 1:length(var)){
  hist(data[, i], main = var[i], xlab = var[i])
}
#The distribution of MORTALITY, NONWHITE POOR NOX and SO2  are left skewed, so
we consider logrithm transformation to them.
data$NONWHITE = log(data$NONWHITE)
data$POOR = log(data$POOR)
data$NOX = log(data$NOX)
data$SO2 = log(data$SO2)
data$MORTALITY = log(data$MORTALITY)
#After logrithm transformation, their distributions are more normal

#Standerlization
```

```
data = as.data.frame(scale(data))

pairs(data)
cor(data)
#The effect of multicollinearility
X = as.matrix(data[,-7])
Y = as.matrix(data[,7])
eigen(t(X) %*% X)

#Calculate VIF
par(mfrow = c(1, 1))
fit = lm(MORTALITY ~ . -1, data = data)
summary(fit)
plot(fit, which = 1)
plot(fit, which = 2)
#There are nonlinear pattern in the residuals vs. fitted values plot, and the
distribution of residuals is heavy-tailed.

vif(fit)
#D = t(X) %*% X
#diag(D) * diag(solve(D))

#Stepwise regression according to AIC criterion.
stepfit = stepAIC(fit, scope = list(upper = fit, lower = ~1), direction =
"backward", k = 2)

fit_step_reg = lm(MORTALITY ~ PRECIP + EDUC + NONWHITE + SO2 - 1, data = data)
plot(fit_step_reg, which = 1)
plot(fit_step_reg, which = 2)

#ridge
par(mfrow = c(2, 2))

select(lm.ridge(MORTALITY ~ . -1, data = data, lambda = seq(0, 30, 0.01)))
result = lm.ridge(MORTALITY ~ . -1, data = data, lambda = seq(0, 30, 0.01))
#GCV plot
plot(seq(0, 30, 0.01), result$GCV, main = "Plot of GCV", xlab = "k", ylab =
"GCV", cex = 0.2)
k_opt = 7.69
fit_ridge = lm.ridge(MORTALITY ~ . -1, data = data, lambda = k_opt)
coef(fit_ridge)
beta = matrix(coef(fit_ridge))
fv = X %*% beta
res = data$MORTALITY - fv
plot(data$MORTALITY, fv, main = "observed vs. fitted values", xlab = "fitted
values", ylab = "observed values")
plot(fv, res, main = "residuals vs. fitted values", xlab = "fitted values",
```

```
ylab = "residuals")
hist(res, main = "histogram of residuals", xlab = "residuals")
qqnorm(res)


D = t(X) %*% X
H = X%*%solve(D + k_opt * diag(ncol(X)))%*%t(X)
beta_e = solve(D + k_opt*diag(ncol(X)))%*%t(X)%*%Y
residual = Y - X%*%beta_e
de_matrix = diag(ncol(H)) - H
sigma2 = sum(residual^2)/sum(diag(de_matrix %*% de_matrix))

cov_beta = sigma2*solve(D + k_opt*diag(ncol(X))) %*% D %*% solve(D +
k_opt*diag(ncol(X)))
std_error = sqrt(diag(cov_beta))
estimation = beta_e
data.frame(estimation, std_error)

#VIF values
VIF_ridge = rep(0, 6)
names(VIF_ridge) = names(vif(fit))
VIF_cov = solve(D + k_opt*diag(ncol(X))) %*% D %*% solve(D +
k_opt*diag(ncol(X)))
VIF_ridge = diag(D) * diag(VIF_cov)
VIF_ridge
vif(fit)

#lasso
fit_lasso = cv.glmnet(X, Y, intercept = FALSE)
plot(fit_lasso)
fit_lasso$lambda.min
coef(fit_lasso)


yfit = predict(fit_lasso, newx = X)
par(mfrow = c(2, 2))
plot(yfit, Y, main = "obs vs. fitted", xlab = "fitted values", ylab =
"observed values")
plot(yfit, Y - yfit, main = "residuals vs. fitted", xlab = "fitted values",
ylab = "residuals")
abline(0, 0)
hist(Y - yfit, main = "hist of residuals", xlab = "residuals")
qqnorm(Y - yfit)


#Partial Least Square
  set.seed(100)
```

```
  fit_pls = plsr(MORTALITY ~ ., 6, data = data, validation = "CV")
  summary(fit_pls)

  sse = rep(0, 6)
  for(i in 1:6){
    sse[i] = sum((residuals(fit_pls)[(60*i-59):(60*i)])^2)
  }

  ssto = sum((data$MORTALITY)^2)
  R_square = rep(0, 6)
  for(i in 1:6){
    R_square[i] = 1 - sse[i]/ssto
  }
  k = c(1:6)
  data.frame(k, sse, R_square)


  F_k = rep(0 ,6)
  F_k[1] = (nrow(data) - 1 - 1)*R_square[1]/(1 - R_square[1])
  for(i in 2:6){
    F_k[i] =  (nrow(data) - i - 1)*(R_square[i] - R_square[i-1])/(1 - R_square[i])
  }
  qF = rep(0, 6)
  for(i in 1:6){
    qF[i] = qf(0.95, 1, nrow(data) - i -1)
  }
  data.frame(k, sse, R_square, F_k, qF)

  plot(fit_pls, plottype = "scores", comps = 1:3)
  loadings(fit_pls)[, 1:3]


#Model Selection
CV = function(train_data, test_data, method){
  if(method == "stepreg"){
    fit = lm(MORTALITY ~ . - 1, data = train_data)
    stepfit = stepAIC(fit)
    res = test_data$MORTALITY - predict(stepfit, test_data)
    return( sum(res^2)/nrow(test_data) )
  }
  else if(method == "ridge"){
    result = lm.ridge(MORTALITY ~ . - 1, data = data_train, lambda = seq(0,
    30, 0.01))
    k_opt = 0.01 * which.min(result$GCV)
    fit = lm.ridge(MORTALITY ~ . - 1, data = data_train, lambda = k_opt)
    beta = matrix(coef(fit))
    X = as.matrix(test_data[, -7])
```

```
    fv = X %*% beta
    res = test_data$MORTALITY - fv
    return( sum(res^2)/nrow(test_data) )
  }
  else if(method == "lasso"){
    X = as.matrix(train_data[, -7])
    Y = as.matrix(train_data[, 7])
    fit = cv.glmnet(X, Y, intercept = FALSE)
    #l_opt = fit$lambda.min
    #fit = cv.glmnet(X, Y, lambda = l_opt, intercept = FALSE)
    X_pred = as.matrix(test_data[, -7])
    Y_pred = as.matrix(test_data[, 7])
    yfit = predict(fit, X_pred)
    res = Y_pred - yfit
    return( sum(res^2)/nrow(test_data) )
  }
  else if(method == "pls"){
    set.seed(100)
    fit_pls = plsr(MORTALITY ~ ., 6, data = data_train, validation = "CV")
    k_opt = which.min(fit_pls$validation$PRESS)
    fit = plsr(MORTALITY ~ ., k_opt, data = data_train, validation = "CV")
    #beta = as.matrix(coef(fit))
    #X_pred = as.matrix(test_data[, -7])
    #Y_pred = as.matrix(test_data[, 7])
    n = nrow(test_data)
    fv = predict(fit, newdata = test_data)[((k_opt - 1)*n + 1):(k_opt*n)]
    res = test_data$MORTALITY - fv
    return( sum(res^2)/nrow(test_data) )
  }
}

#Cross validation - 10 folders
set.seed(123)
data_new = sample(data)

getCV = function(method, data_new, nfolder){
  cv_value = rep(0, nfolder)
  n = floor(nrow(data_new)/nfolder)
  for(i in 1:nfolder){
    test_data = data_new[((i-1)*n +1):(i*n), ]
    train_data = data_new[-c(((i-1)*n +1):(i*n)), ]
    cv_value[i] = CV(train_data, test_data, method)
  }
  print(method)
  return(sum(cv_value)/nfolder)
}
```

```
method = c("stepreg", "ridge", "lasso", "pls")
CV_result = sapply(method, function(x) getCV(x, data_new, 10))
CV_result

#Ridge is the best
```