Assignment #2

COMP 424 Big Data

Qiangqiang Li (Aaron)

ID: 300422249

Tutors: Prof. Mengjie Zhang

Dr. Bing Xue

1. *Regression*

(1) How many predictors are there, i.e. what is p?

Reply: There are 400 instances and 65 predictors, and p is 65.

```
> Credit = read.csv("Credit.csv",header=TRUE)
> Credit<-Credit[-c(1)]
> X = model.matrix(Balance~.*.,Credit)[,-1]
> y = Credit$Balance
> dim(X)
[1] 400  65
```

(2) How did you generate your training and test sets?

Reply: RNG random seed was 13579, split 400 instances to 2 data set,
200 training data and 200 test data.

```
> set.seed(13579)
> train=sample(1:nrow(X),nrow(X)/2)
> test=-train
```

(3) Select the tuning parameter for the ridge regression model using crossvalidation, and show the process.

Reply: RNG random seed was 13579. The tuning parameter for the ridge regression, According to lecture , K= 5 or 10 is typical , this case assume K = 10:

```
> grid = 10^seq(3,-1,length=100)
> cv1.out = cv.glmnet(X[train,],y[train],alpha=0,lambda=grid,nfolds=10,thresh=1e-10)
> cv1.out$lambda.min
[1] 0.8497534
```

The result is that set λ_{min} as the tuning parameter.

(4)Select the tuning parameter for the lasso regression model using crossvalidation, and show the process. How many features have been selected by the lasso?

Reply: RNG random seed was 13579. The tuning parameter for the lasso regression, assume K = 10:

```
> cv2.out = cv.glmnet(X[train,],y[train],alpha=1,lambda=grid,nfolds=10,thresh=1e-10)
> cv2.out$lambda.min
[1] 0.9326033
```

Set λ_{min} as tuning parameter and the number of selected features is 28.

```
> bestlam2 = cv2.out$lambda.min
> out = glmnet(X,y,alpha=1,lambda=grid,thresh=1e-10)
> result = predict(out,type="coefficients",s=bestlam2)[1:66,]
> selected_result = result[which(predict(out,type="coefficients",s=bestlam2)[1:66,]!=0)]
> names(selected_result)
 [1] "(Intercept)"                 "Income"                      "Limit"
 [4] "Cards"                       "StudentYes"                  "MarriedYes"
 [7] "Income:Rating"               "Income:Cards"                "Income:Age"
[10] "Income:Education"            "Income:GenderFemale"         "Income:StudentYes"
[13] "Limit:Rating"                "Limit:Cards"                 "Limit:Education"
[16] "Limit:GenderFemale"          "Limit:StudentYes"            "Limit:EthnicityAsian"
[19] "Cards:GenderFemale"          "Cards:StudentYes"            "Cards:EthnicityCaucasian"
[22] "Age:Education"               "Age:StudentYes"              "Age:EthnicityAsian"
[25] "Education:GenderFemale"      "Education:EthnicityCaucasian" "GenderFemale:MarriedYes"
[28] "StudentYes:MarriedYes"       "MarriedYes:EthnicityCaucasian"
> num=sum(as.numeric(result!=0))-1
> num
[1] 28
```

(5) Compare and discuss the final form of the model from the linear regression, ridge regression, and lasso regression.

Reply: According to RSS for linear regression, it is $\sum_{i=1}^{200}(y_i - \beta_0 - \sum_{j=1}^{65} x_{ij}\,\beta_j)^2$. Using the result of the question 3 and the question 4, the final form of ridge regression is $RSS + 0.8497534\sum_{j=1}^{65}\beta_j^2$, and the final form of lasso regression is $RSS + 0.9326033\sum_{j=1}^{28}|\beta_j|$.

In linear regression, we estimate parameters by minimizing the RSS in a given set of data. However, when $\hat{\beta}_j$ are unconstrained, this will lead to a very high variance. So we add a penalized method to constrain the coefficient estimates.

In this case, $0.8497534\sum_{j=1}^{65}\beta_j^2$ is the penalized method for ridge regression. $0.9326033\sum_{j=1}^{28}|\beta_j|$ is the penalized method for lasso regression.

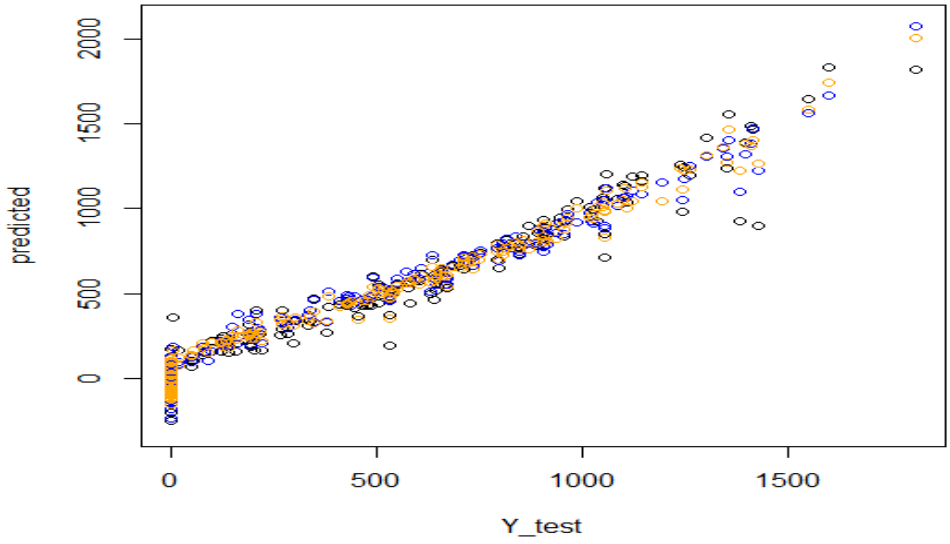
(6) Compare the test errors for the linear model, ridge regression model, and lasso model.

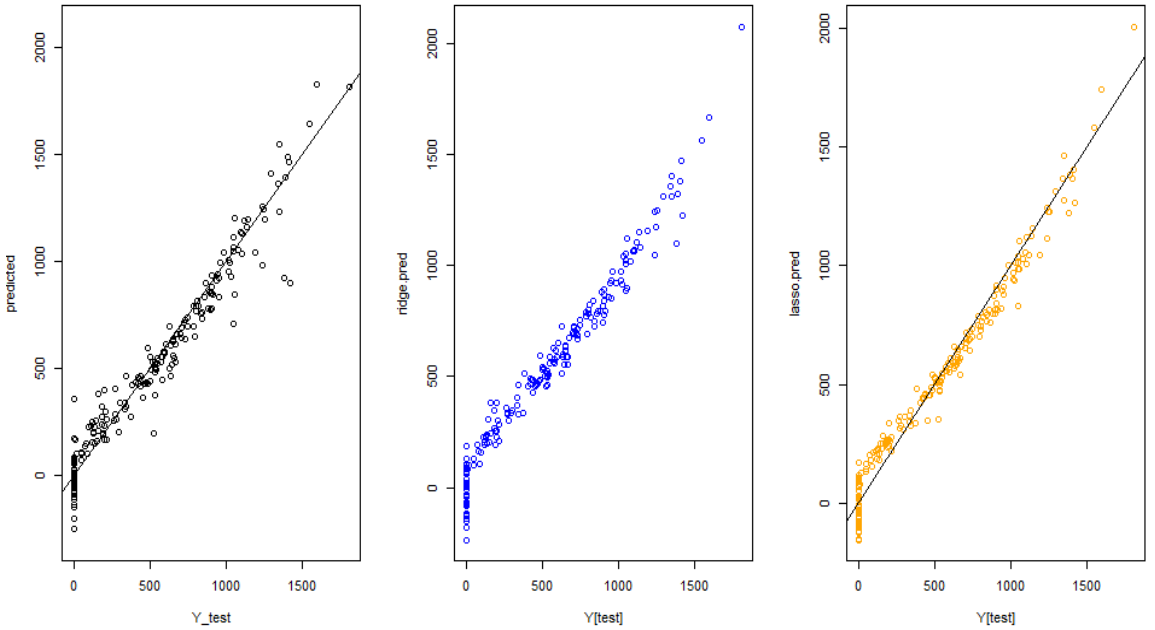
Reply: According to the result, the test error for linear model is 8348.549, which is a little higher than 5916.27 of ridge model. The lowest model is lasso model which is 4729.232. In this case, lasso model is the best.

```
> set.seed(13579)
> linear.mod = lm(y[train]~X[train,])
> linear.pred = coef(linear.mod)[1]+X[test,] %*% coef(linear.mod)[-1]
> mean((linear.pred-y[test])^2)
[1] 8348.549
> ridge.pred = predict(cv1.out,s=bestlam1,newx=X[test,])
> mean((ridge.pred-y[test])^2)
[1] 5916.27
> lasso.pred = predict(cv2.out,s=bestlam2,newx=X[test,])
> mean((lasso.pred-y[test])^2)
[1] 4729.232
```

(7) Plot a comparison of the test predictions for the three approaches

Reply: Linear model: **black point** ; Ridge regression model: **blue point**; Lasso model: **orange point**
Three models are in one plot:





2 . Generalised additive models

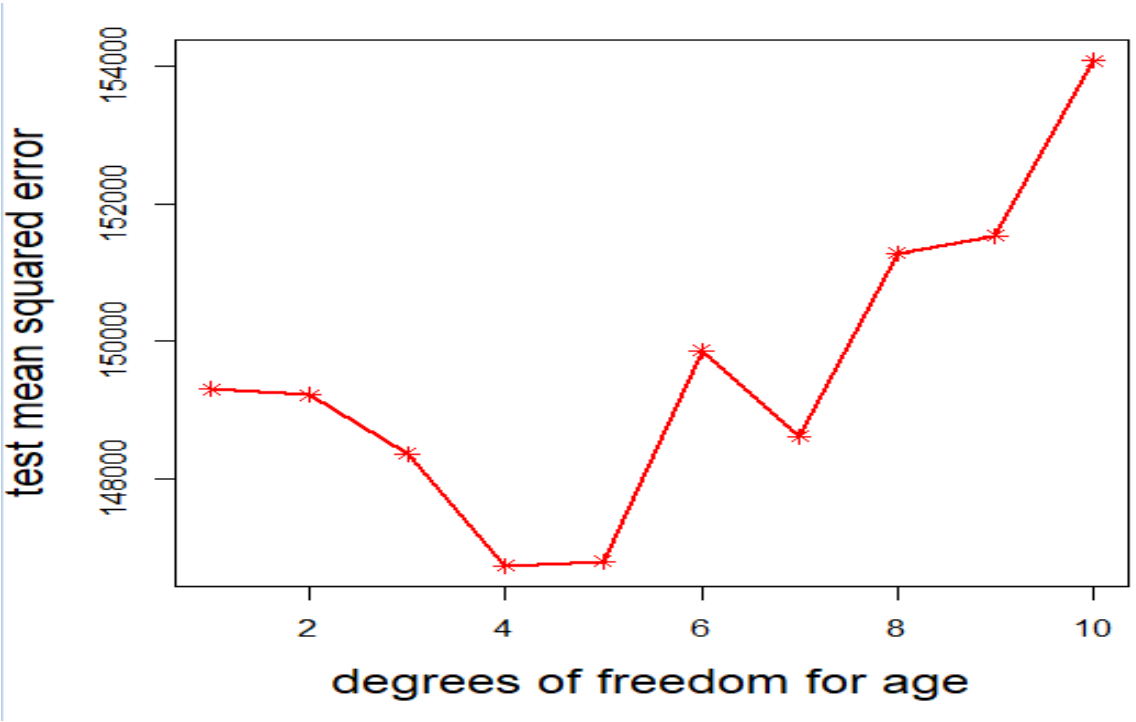
(1).What happens to the test mean squared error as the degrees of freedom of the natural spline for age is varied?

Reply: According to the model as follows:

gam(balance~ ns(income,df=n)+ age+student,data=Credit[train,])

When RNG seed: 10000, and the degrees of freedom of the natural spline for agen = 1-10:

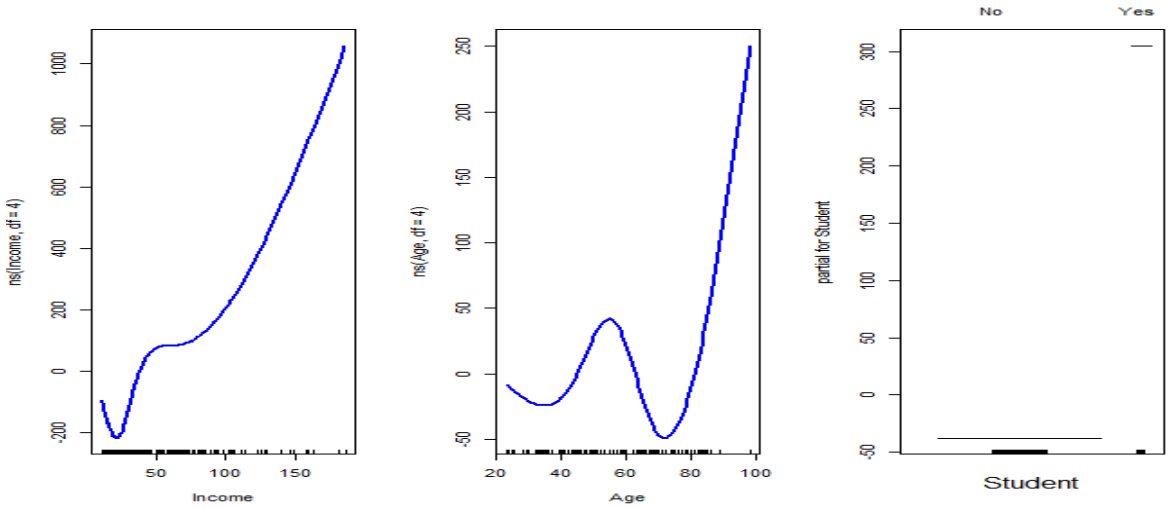
```
> train = sample(1:nrow(Credit),nrow(Credit)/2)
> test = -train
> set.seed(10000)
> library(gam)
> train = sample(1:nrow(Credit),nrow(Credit)/2)
> test = -train
> gam.mod1 = gam(Balance~ ns(Income,df=4)+ ns(Age,df=1)+Student,data=Credit[train,])
> gam.mod2 = gam(Balance~ns(Income,df=4)+ns(Age,df=2)+Student,data=Credit[train,])
> gam.mod3 = gam(Balance~ns(Income,df=4)+ns(Age,df=3)+Student,data=Credit[train,])
> gam.mod4 = gam(Balance~ ns(Income,df=4)+ ns(Age,df=4)+Student,data=Credit[train,])
> gam.mod5 = gam(Balance~ns(Income,df=4)+ns(Age,df=5)+Student,data=Credit[train,])
> gam.mod6 = gam(Balance~ns(Income,df=4)+ns(Age,df=6)+Student,data=Credit[train,])
> gam.mod7 = gam(Balance~ns(Income,df=4)+ns(Age,df=7)+Student,data=Credit[train,])
> gam.mod8 = gam(Balance~ns(Income,df=4)+ns(Age,df=8)+Student,data=Credit[train,])
> gam.mod9 = gam(Balance~ns(Income,df=4)+ns(Age,df=9)+Student,data=Credit[train,])
> gam.mod10 = gam(Balance~ns(Income,df=4)+ns(Age,df=10)+Student,data=Credit[train,])
>
> pred.mod1 = predict(gam.mod1,newdata=Credit[test,])
> pred.mod2 = predict(gam.mod2,newdata=Credit[test,])
> pred.mod3 = predict(gam.mod3,newdata=Credit[test,])
> pred.mod4 = predict(gam.mod4,newdata=Credit[test,])
> pred.mod5 = predict(gam.mod5,newdata=Credit[test,])
> pred.mod6 = predict(gam.mod6,newdata=Credit[test,])
> pred.mod7 = predict(gam.mod7,newdata=Credit[test,])
> pred.mod8 = predict(gam.mod8,newdata=Credit[test,])
> pred.mod9 = predict(gam.mod9,newdata=Credit[test,])
> pred.mod10 = predict(gam.mod10,newdata=Credit[test,])
>
> mse1 = mean((pred.mod1-Credit $Balance[test])^2)
> mse2 = mean((pred.mod2-Credit $Balance[test])^2)
> mse3 = mean((pred.mod3-Credit $Balance[test])^2)
> mse4 = mean((pred.mod4-Credit $Balance[test])^2)
> mse5 = mean((pred.mod5-Credit $Balance[test])^2)
> mse6 = mean((pred.mod6-Credit $Balance[test])^2)
> mse7 = mean((pred.mod7-Credit $Balance[test])^2)
> mse8 = mean((pred.mod8-Credit $Balance[test])^2)
> mse9 = mean((pred.mod9-Credit $Balance[test])^2)
> mse10 = mean((pred.mod10-Credit $Balance[test])^2)
>
> y = c(mse1,mse2,mse3,mse4,mse5,mse6,mse7,mse8,mse9,mse10)
> y
 [1] 149306.7 149225.3 148373.1 146751.2 146815.1 149867.1 148625.2 151277.1 151533.5 154074.5
```



According to the result, the degree of freedom is 4, the test mean squared error is lowest.
It is clear that the error is decreasing while the degrees of freedom for age is increasing. The general trend is that after reaching the degree of freedom with lowest mean squared error, the error value rises as the increase of degree of freedom.

(2). Choose the model with the lowest test error. Describe the effect of age in this model.
Reply: The model with the lowest test error is:
gam.mod4 = gam(balance~ ns(income,df=4)+ ns(age,df=4)+student,data=Credit[train,])

The effect of age in this model is fluctuant. Looking the result like the below picture, there is positive correlation between age and balance when age is from 35-55 and >70. On the other hand, the negative effect is the age which is below 35 and the range in 55-70.



(3)How does the square root of the mean square error (RMSE) compare to the typical size of balance? Does your best model seem like a good model and why ?

```
> sqrt(mse4)
[1] 383.0811
> mean(Credit$Balance)
[1] 520.015
```

Reply: RMSE is 383.0811, which is used to measure the deviation between the observed value and the predicted value.
The typical size of balance is 520.015, which demonstrates the general situation and average level of a dataset.
The coefficient of variation of this model is 73% which means the dispersion of this model is high. So this model is not good enough.

3. Regression with high-dimensional data

(1) Confirm that a linear model can fit the training data exactly and provide the evidence. Discuss this model is going to be useful or not and why?
Reply: set RNG seed: 987654321

```
set.seed(987654321)
Parkinsons= read.csv("parkinsons.csv",header=TRUE)
Parkinsons<-Parkinsons[-c(1)]
X = model.matrix(UPDRS~.,Parkinsons)[,-1]
y = Parkinsons$UPDRS
X = scale(X)
train = sample(1:nrow(X), 30)
test = -train
linear.mod = lm(y[train]~X[train,])
summary(linear.mod)


Call:
lm(formula = y[train] ~ X[train, ])

Residuals:
ALL 30 residuals are 0: no residual degrees of freedom!

Coefficients: (68 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)
(Intercept)      36.19         NA      NA       NA
X[train, ]X1   -197.75         NA      NA       NA
X[train, ]X2    206.27         NA      NA       NA
X[train, ]X3   -122.06         NA      NA       NA
X[train, ]X4    -29.20         NA      NA       NA
X[train, ]X5    280.54         NA      NA       NA
X[train, ]X6     35.22         NA      NA       NA
X[train, ]X7    220.90         NA      NA       NA
X[train, ]X8   -123.45         NA      NA       NA
X[train, ]X9     37.36         NA      NA       NA

X[train, ]X18   -53.35         NA      NA       NA
X[train, ]X19  -118.87         NA      NA       NA
X[train, ]X20    61.42         NA      NA       NA
X[train, ]X21   185.98         NA      NA       NA
X[train, ]X22   -18.66         NA      NA       NA
X[train, ]X23  -294.41         NA      NA       NA
X[train, ]X24    19.33         NA      NA       NA
X[train, ]X25 30097.20         NA      NA       NA
X[train, ]X26 10095.99         NA      NA       NA
X[train, ]X27 -3134.49         NA      NA       NA
X[train, ]X28  9345.08         NA      NA       NA
X[train, ]X29 -7946.79         NA      NA       NA
X[train, ]X30       NA         NA      NA       NA
X[train, ]X31       NA         NA      NA       NA
X[train, ]X32       NA         NA      NA       NA
X[train, ]X33       NA         NA      NA       NA
```

```
X[train, ]X91         NA          NA       NA        NA
X[train, ]X92         NA          NA       NA        NA
X[train, ]X93         NA          NA       NA        NA
X[train, ]X94         NA          NA       NA        NA
X[train, ]X95         NA          NA       NA        NA
X[train, ]X96         NA          NA       NA        NA
X[train, ]X97         NA          NA       NA        NA

Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:    NaN
F-statistic:   NaN on 29 and 0 DF,  p-value: NA
```

Reply: "Residual standard error" thrown out, in a linear model, more observations compared with the number of features is required. However, there are only 42 observations. There are 97 features, and the result present that the p-value is NA, Therefore, the model is useless.

(2)Now use the lasso to fit the training data, using leave-one-out crossvalidation to find the tuning parameter λ. (You will find it convenient to set grid = 10∧seq(3, − 1, 100) and thresh=1e-10.) What is the optimal value of λ and what is the resulting test error?

Reply: According to the result, the λ is 1.232847, and the resulting test error is 27.01453.

```
> library(glmnet)
> grid = 10^seq(3,-1,length = 100)
> cv.out = cv.glmnet(X[train,],y[train],alpha=1,lambda=grid,nfolds=30,thresh=1e-10)
Warning message:
Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold
> cv.out$lambda.min
[1] 1.232847
>
> bestlam = cv.out$lambda.min
> lasso.pred = predict(cv.out,s=bestlam,newx=X[test,])
> mean((lasso.pred-y[test])^2)
[1] 27.01453
> plot(cv.out)
```

(3)State your final model for the UPDRS. How many features have been selected? What conclusions can you draw?

Reply: According to the result, 2 features are selected, which are X83, X97.

The value of X97 is significantly higher than X83. So X97 is the most important feature in this model.

```
> out = glmnet(X,y,alpha=1,lambda=grid,thresh=1e-10)
> result = predict(out,type="coefficients",s=bestlam)[1:98,]
> selected_result = result[which(predict(out,type="coefficients",s=bestlam)[1:98,]!=0)]
> selected_result
(Intercept)         X83         X97
 26.6192659   0.1620791   8.6736408
```

(4)Repeat your analysis with a different random split into training and test sets. Have the same features been selected in your final model?

Reply: In this case ,using this split data into 33 training data and rest as test data. Only X97 which was chosen in which is selected in question 3.

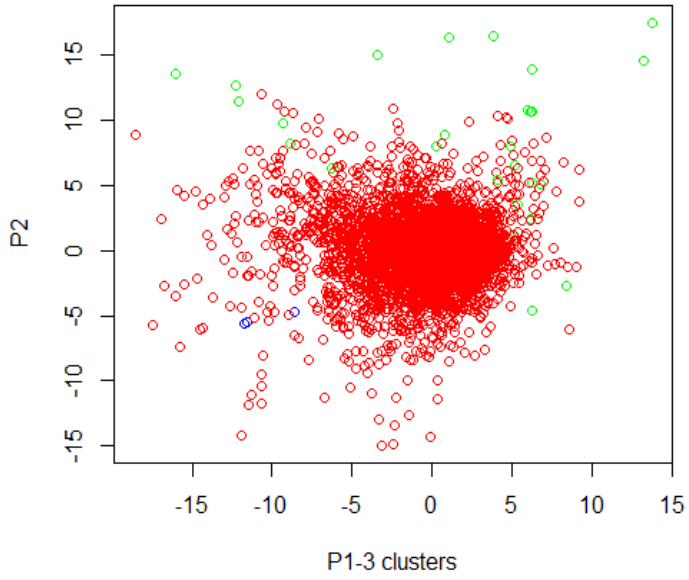
```
> train = sample(1:nrow(X), 33)
> test = -train
> cv1.out = cv.glmnet(X[train,],y[train],alpha=1,lambda=grid,nfolds=30,thresh=1e-10)
Warning message:
Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold
> cv1.out$lambda.min
[1] 2.364489
> bestlam1 = cv1.out$lambda.min
> lasso.pred = predict(cv1.out,s=bestlam1,newx=X[test,])
> mean((lasso.pred-y[test])^2)
[1] 7.913001
> out = glmnet(X,y,alpha=1,lambda=grid,thresh=1e-10)
> result = predict(out,type="coefficients",s=bestlam1)[1:98,]
> selected_result = result[which(predict(out,type="coefficients",s=bestlam1)[1:98,]!=0)]
> selected_result
(Intercept)         X97
  26.619266    7.573616
```

4.Clustering

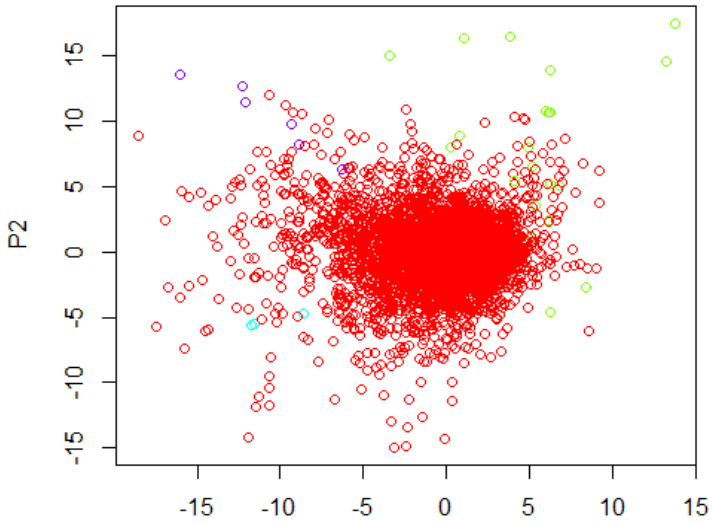
(1) Carry out hierarchical clustering with Euclidean distance and complete linkage. Describe the resulting clustering for 3-6 clusters. Hint:table() also works with only one argument. Plot the first 2 principal components against each other with the colour argument set equal to the cluster labels. What can you deduce/observe about the clustering?

Reply: Using the resulting clustering for 3-6 clusters. When increasing clusters from 3 to 6, the result is not good enough. According to the result, the majority of gene expression are clustered to one cluster. So this is far from reality. It is clear that the hierarchical clustering with Euclidean distance and complete linkage is not a good solution in this case.

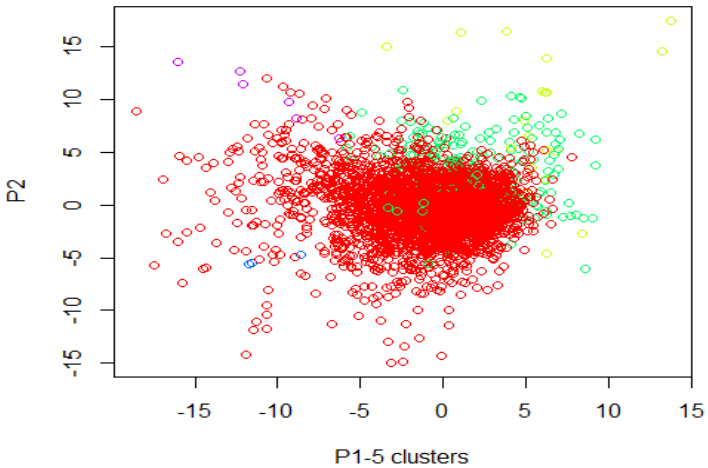
```
> table(cutree(hc,3))

   1    2    3
6792   35    3
> table(cutree(hc,4))

   1    2    3    4
6792   29    3    6
> table(cutree(hc,5))

   1    2    3    4    5
6559   29  233    3    6
> table(cutree(hc,6))

   1    2    3    4    5    6
6559    7  233   22    3    6
```



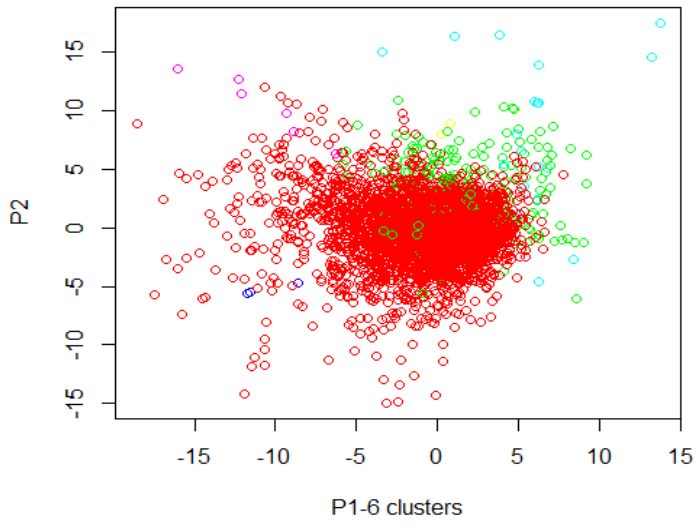
P1-3 clusters



P1-4 clusters



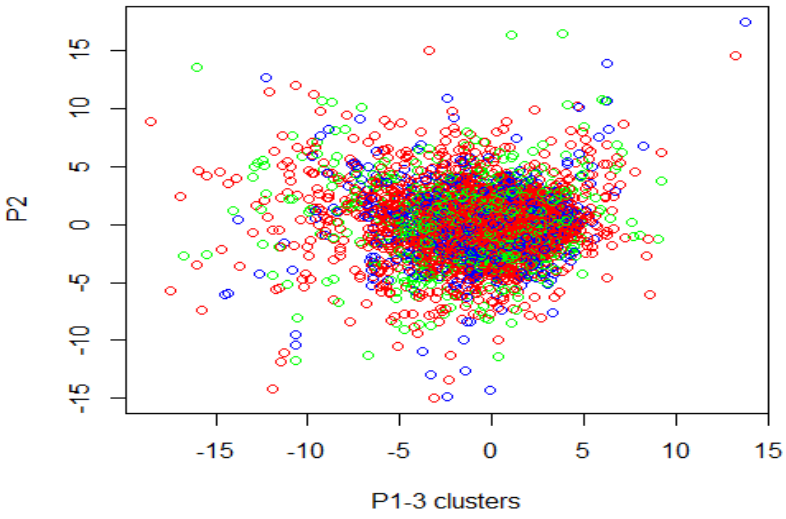
P1-5 clusters

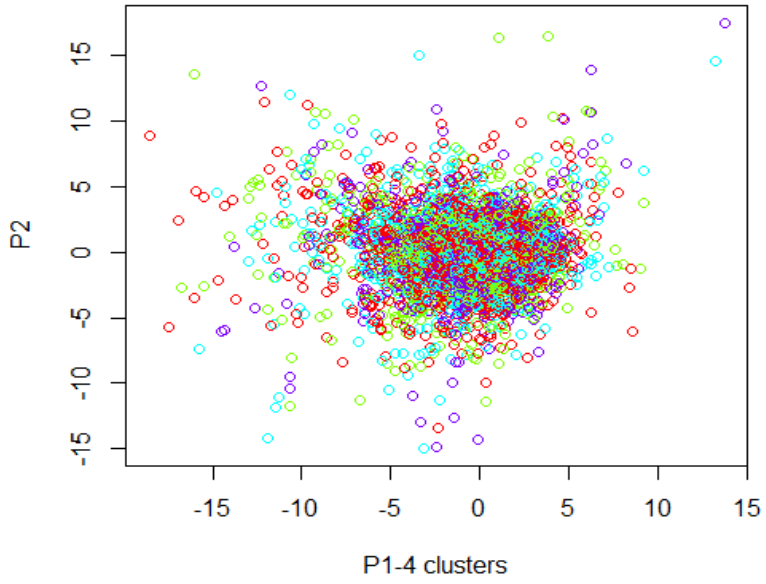


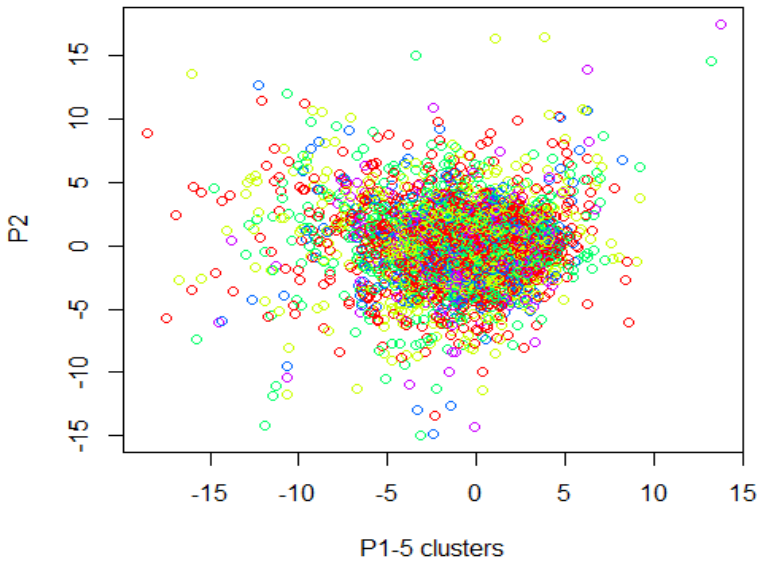
(2) Repeat the cluster analysis using correlation-based distance and complete linkage. You will need to use as.dist(), which converts a distance matrix into a form recognised by hclust(). As for the distance matrix itself, note that cor(t(X)) gives all pairwise correlations of the rows of X. Compare the clusters with those found above.

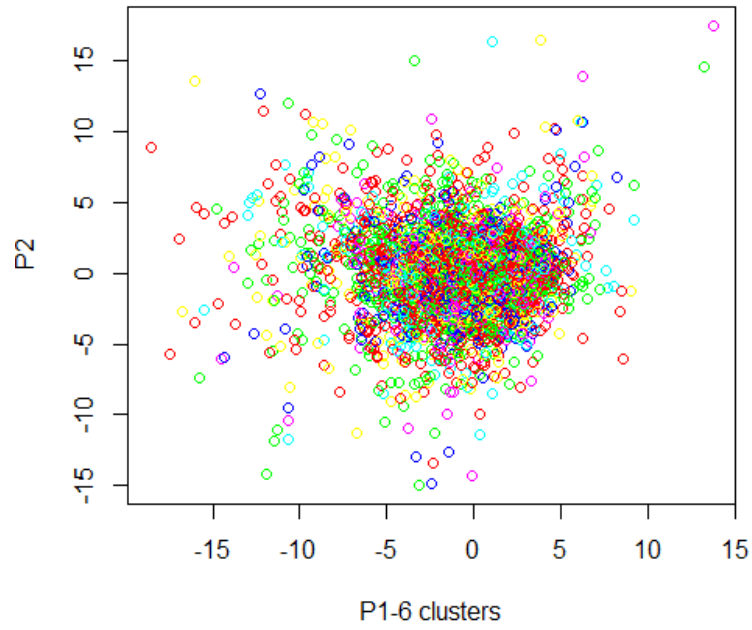
Reply: From the result of the tables, the clustering analysis using correlation-based distance and complete linkage has better performance than the hierarchical clustering with Euclidean distance. Clusters are distributed well, Looking at the result , it is unlike the hierarchical clustering with Euclidean distance that the majority of gene expression are clustered to one cluster.

```
> table(cutree(hcl,3))

   1    2    3
3855 1665 1310
> table(cutree(hcl,4))

   1    2    3    4
2215 1665 1640 1310
> table(cutree(hcl,5))

   1    2    3    4    5
2215 1665 1640  763  547
> table(cutree(hcl,6))

   1    2    3    4    5    6
2215  920 1640  745  763  547
```


P2
15
10
5
0
-5
-10
-15
-15 -10 -5 0 5 10 15
P1-3 clusters

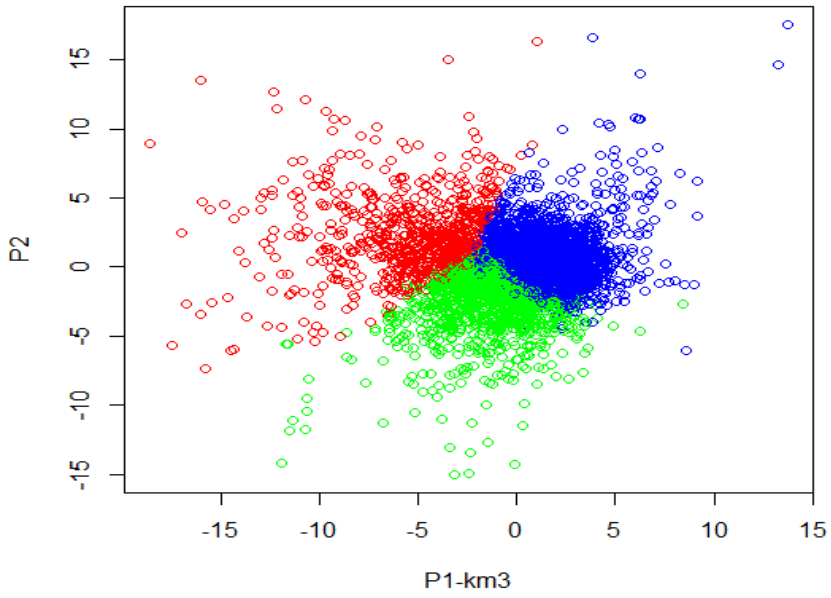

P2
15
10
5
0
-5
-10
-15
-15 -10 -5 0 5 10 15
P1-4 clusters

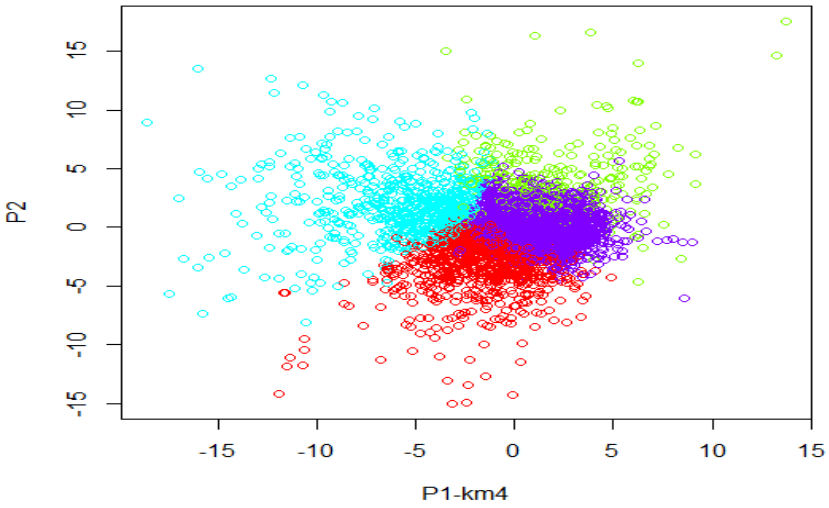

P2
15
10
5
0
-5
-10
-15
-15 -10 -5 0 5 10 15
P1-5 clusters

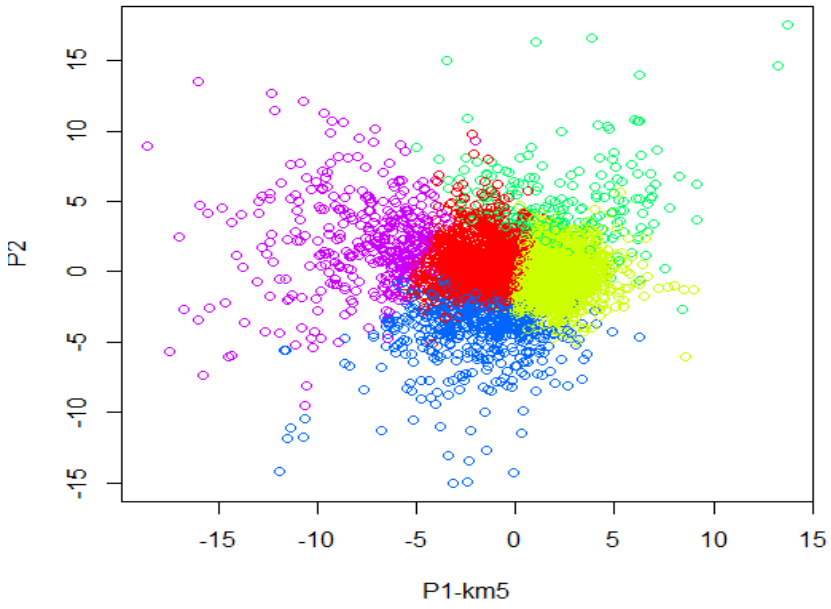


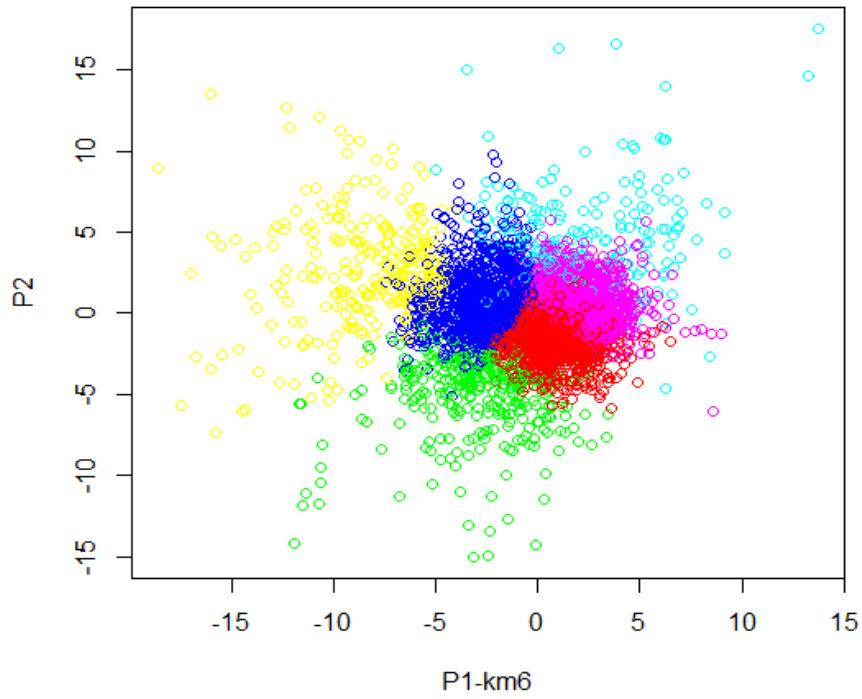
(3) Finally, carry out K-means clustering for 3 - 6 clusters. Compared the clusters of K-means with that of the above two approaches, find which of the hierarchical clusterings results is more similar to that of K-means?

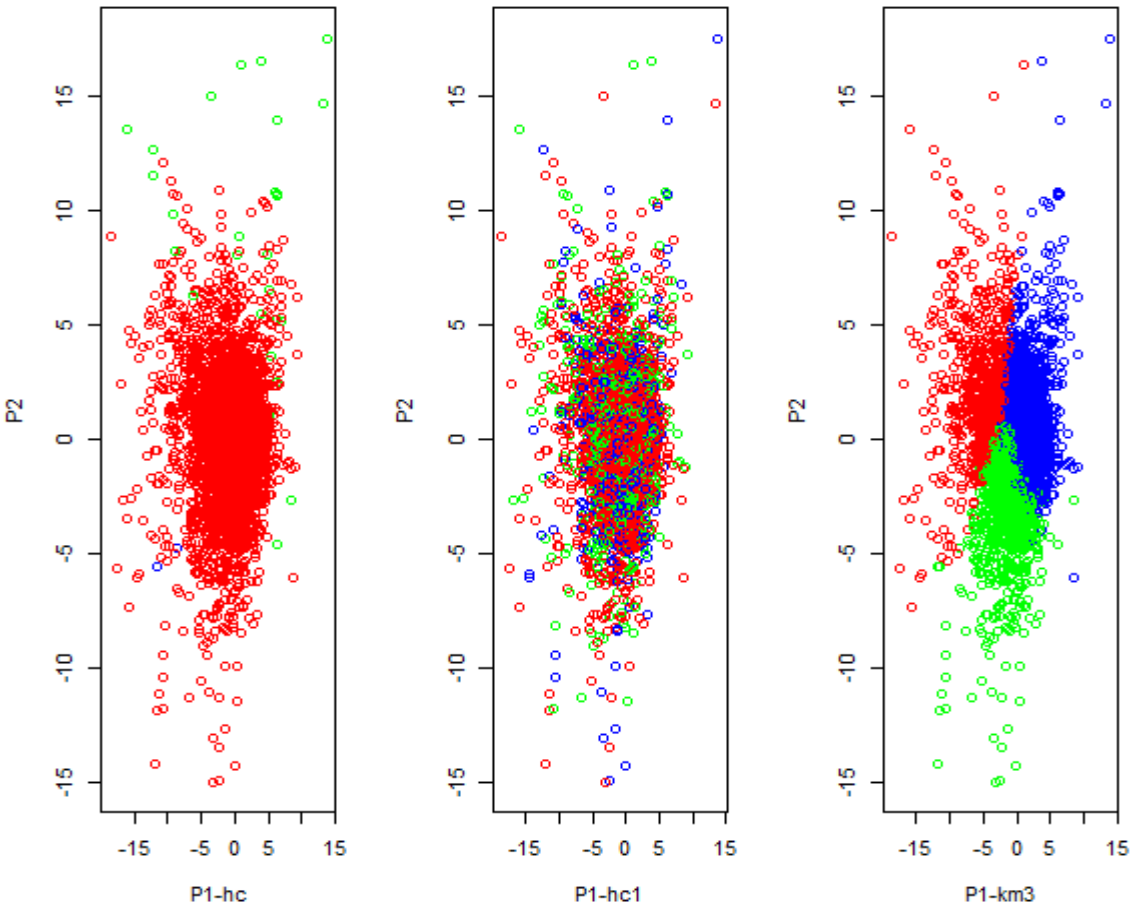
```
> km3 = kmeans(X,3,nstart=50)
> table(km3$cluster)

   1    2    3
 763 1399 4668
> km4 = kmeans(X,4,nstart=50)
Warning messages:
1: Quick-TRANSfer stage steps exceeded maximum (= 341500)
2: Quick-TRANSfer stage steps exceeded maximum (= 341500)
3: Quick-TRANSfer stage steps exceeded maximum (= 341500)
4: Quick-TRANSfer stage steps exceeded maximum (= 341500)
> table(km4$cluster)

   1    2    3    4
1034  380  711 4705
> km5 = kmeans(X,5,nstart=50)
> table(km5$cluster)

   1    2    3    4    5
1823 3786  277  553  391
> km6 = kmeans(X,6,nstart=50)
> table(km6$cluster)

   1    2    3    4    5    6
2054  280  328  255 1226 2687
```

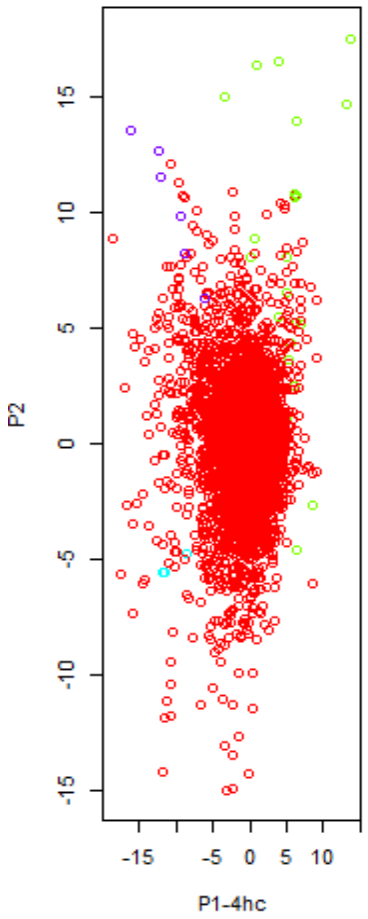

P2
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15
P1-km3

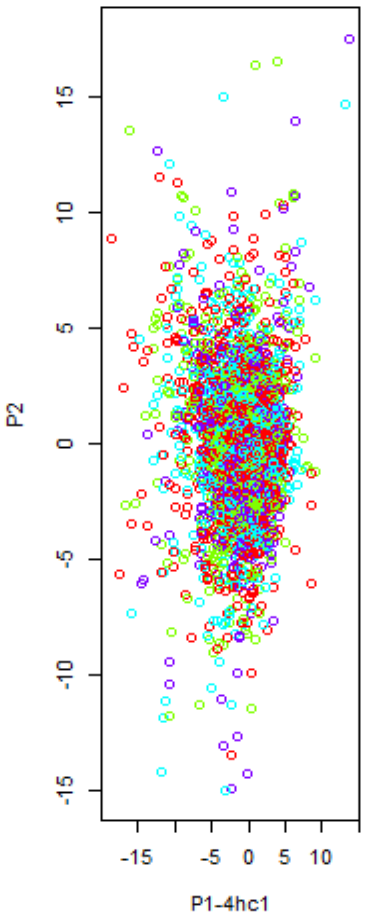

P2
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15
P1-km4

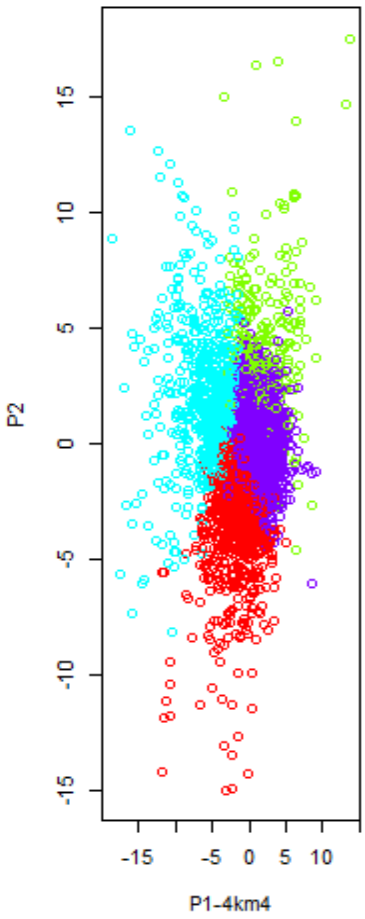

P2
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15
P1-km5

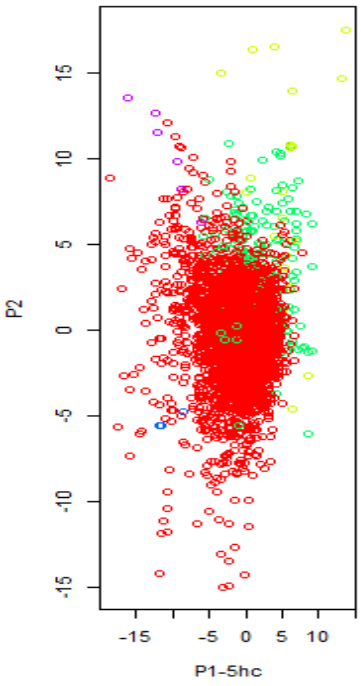

P2
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15
P1-km6

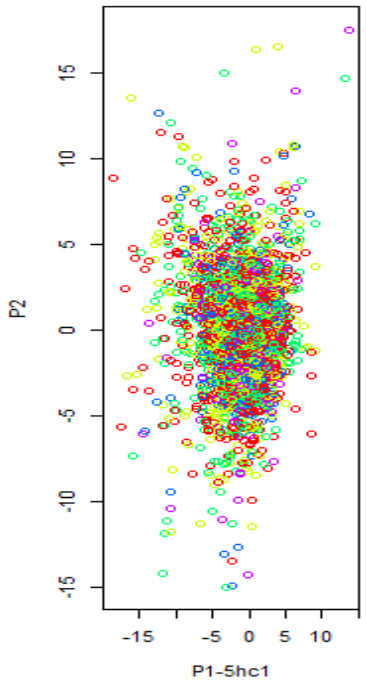

P2
P1-hc
P1-hc1
P1-km3
-15
-5
0
5
10
15

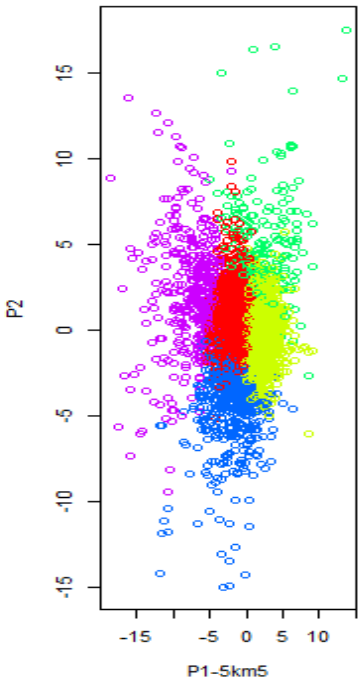

P2
P1-4hc

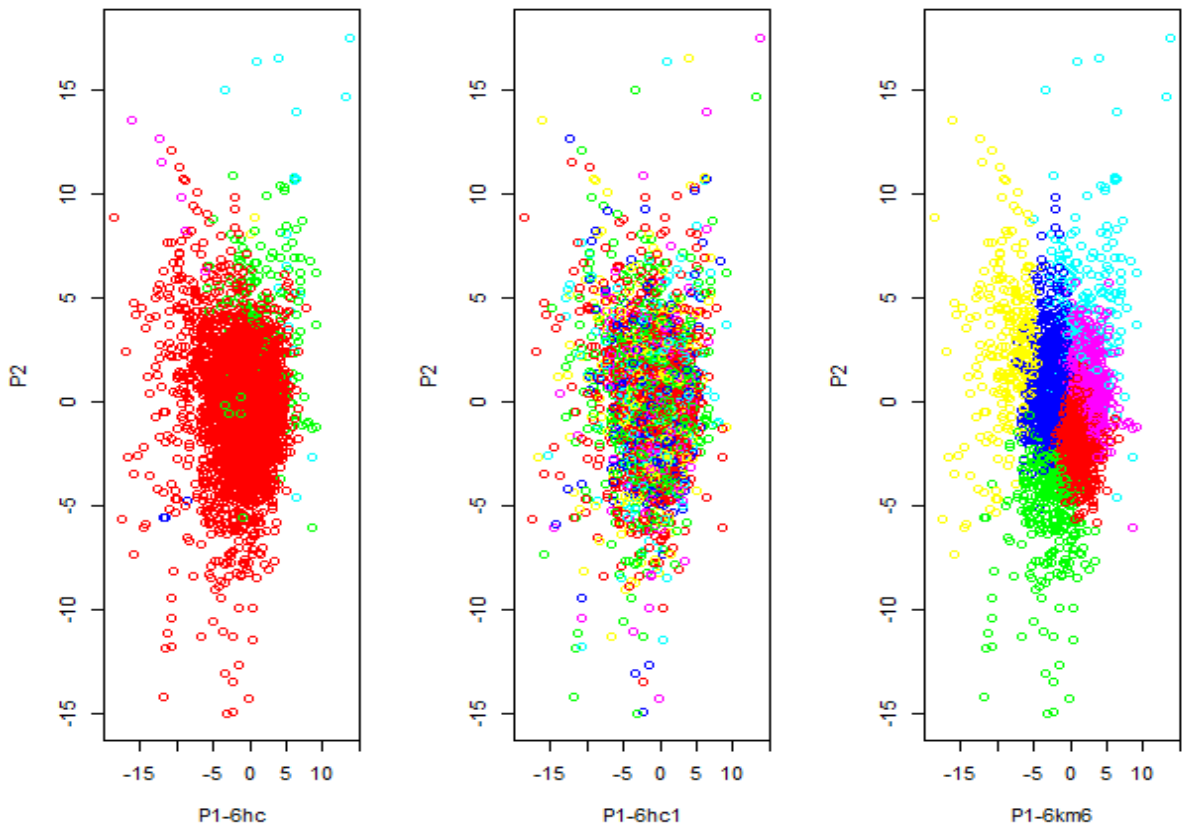

P2
P1-4hc1


P2
P1-4km4


P2
P1-5hc


P2
P1-5hc1


P2
P1-5km5



Comparing the table and clustering results, k-mean clustering is also a good method for this case. It is clear to find some similarities between the hierarchical clustering with Euclidean distance and k-mean clustering from plot images.

5. write a report on the paper "Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission" by Caruana et al

In the modern society, the application of machine learning plays an important role in healthcare. A tradeoff must be made between accuracy and intelligibility in machine learning. Many accurate models such as boosted trees, random forests, and neural nets usually are not intelligible. But more intelligible models such as logistic regression, naive-Bayes, and single decision trees often have significantly worse accuracy.

This paper presents two case studies where high-performance generalized additive models with pairwise interactions (GA^{2}Ms), which is applied to deal with tradeoff between accuracy and intelligibility in real healthcare problems.

Firstly, this paper provides a brief introduction to GAM and GA^{2}Ms. GAMs is Let $D = \{(x_i, y_i)\}_1^N$ denote a training dataset of size N, where $x_i = (x_{i1}, \ldots, x_{ip})$ is a feature vector with p features and y_i is the target (response). We use x_j to denote the jth variable in the feature space:

$$y_i = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i$$

GA²Ms is building the best GAM first and then detects and ranks all possible pairs of interactions in the residuals to improve accuracy and the pairwise interactions:

$$y = \beta_0 + \sum_{j=1}^{p} f_j(x_j) + \sum_{i \neq j}^{p} f_{ij}(x_i, x_j)$$

Secondly, GA²Ms model is accurate, intelligible, and repairable in the case study of pneumonia. GA² Ms model offers a simple approach to discover and correct risky rules like the asthma rule, chronic lung disease and history of chest pain beyond the Cost-Effective HealthCare (CEHC) study.

Thirdly, in the 30-day hospital readmission case study, predictions of only three typical patients and the top 6 terms for each of these patients are examined because the readmission dataset contains too many features. Compared to an unintelligible models as boosted trees or a complex neural net, the model is fairly transparent, and the predictions are intelligible for both whole and individual patient.

Last but not least, compared with many other learning methods, the models are intelligible enough to provide a window into the data and prediction problem, and this window allows questions that will require investigation and further data analysis to answer. For example, like the below Table1, GAM/ GA²Ms methods achieve the best accuracy in AUC on this problem.

Model	Pneumonia	Readmission
Logistic Regression	0.8432	0.7523
GAM	0.8542	0.7795
GA2M	0.8576	0.7833
Random Forests	0.8460	0.7671
LogitBoost	0.8493	0.7835

Table 1: AUC for different learning methods

Finally, this paper discusses a wide range of issues that arise when learning with intelligible model and our general lessons. The risk score for every term is set to zero in order to make the models identifiable and modular. GA²Ms model can figure out continuous attributes, which is impossible for many other methods, and obtain more accurately than experts by shaping these attributes.

In conclusion, GA²Ms model achieve state-of-the-art accuracy while remaining intelligible on two case studies on real medical data. GA²Ms represent a significant step forward in the tradeoff between model accuracy and intelligibility. The high-accuracy learned models can be easier deployed in applications where model verification and debug ability are as important as accuracy.

(word counts: 491)