Chengjun Yuan

cy3yb@virginia.edu

Question #1

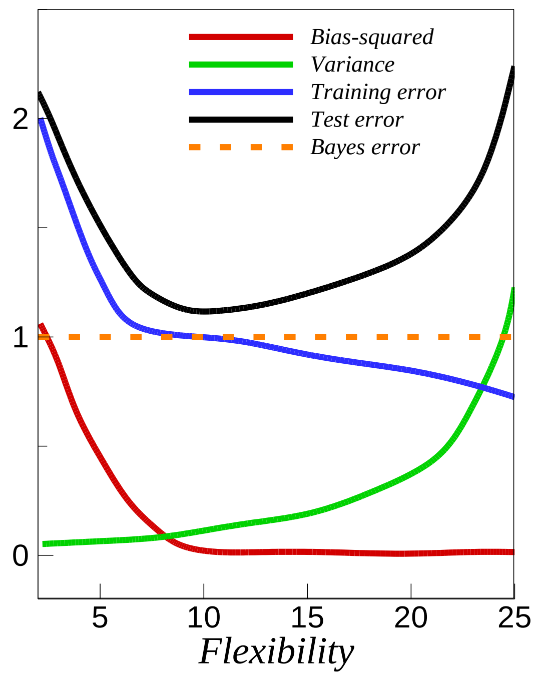

FIG. 1. Curves of the typical squared bias, variance, training error, test error and Bayes error as the function of flexibility.

a) A more flexible model includes greater number of parameters and so is able to fit more different possible functional forms for true *f*, which is the advantages. The disadvantage lies in that the more flexible models can lead to a phenomenon known as over-fitting the data, which means they follow the errors or noise too closely[1] and may leads to high test error.

b) For squared bias, it refers to the error that is introduced by approximating a real-life problem by a simpler method. So more flexibility, more accurate approximation in training data, which leads to smaller bias error.

For variance, it refers to the amount by which $\hat{f}$ would change if a different training data set is used. High flexible method has high variance since the small changes in the training data can result in large changes in $\hat{f}$ because of the negative effect of over fitting.

For training error and test error, since the more flexible method can fit the training data more accurately, the training error will decrease as the flexibility increases, but the test error will not. If the method's flexibility is apparently less than the complexity of the true function f, then increasing flexibility will decrease the test error as well as the training data. But if the flexibility has exceeded the complexity of f, which means that the method is working too hard to find patterns in the training data and may be picking up some patterns that are just caused by random chance rather than by true properties of f, the test error will grow because the patterns found in the training data don't exist in the test data[1]. That is why the test error has a U-shape curve.

c) Resampling techniques is to fit the same statistical model many times by using different subsets of the training data. And the cross-validation is often used to estimate the test error of the statistical learning model to evaluate the performance and to optimize its parameters, such as flexibility. The bootstrap is used to measure the accuracy of the parameter estimate and the performance of a given regression or classification model.

d) Compared with the standard sample set method, the resampling methods have a significant advantage. The standard sample method assumes that the sample data is a good reflection of the distribution of parent data, but the resampling methods make no assumptions about the parent distribution. Therefore, by using resampling methods, the model's performance can be evaluated more accurately as well as the selection of the proper flexibility of the model.

Question #2

a) Firstly, we display the matrix of scatterplots of the training data (using CODES. 1) in FIG. 2. The "y" is found to be positively correlated with "x3" and "x5". In addition, there is a strong positively linear relationship between "x1" and "x6" and a parentally non-linear correlation between "x3" and "x5". Therefore, there is considerable collinearity in the data set. Furthermore, the matrix of the correlation between the variables is provided in CODES. 1 using command "cor()". The correlation value of "x1" and "x6" is 0.9394, and that of "x3" and "x5" is 0.7883. Both of them are much larger than 0.5. Besides, the correlation value between "x1" & "x2", "x2" & "x6" and "x3" & "x5" also exceed 0.5. They further validate the many presence of collinearity. Finally, the variable "x2" has only two status: 0 and 1. The boxplot of y vs. "x2" in FIG. 3 shows that the difference of "y" under the two status of "x2" is negligible. Therefore, the "x2" can be removed from the variables used for the prediction model.

```
>library(ISLR)
>setwd("C:/Dropbox/DataMining/FinalWork")
>Q2.train<-read.csv("SYS6018-FinalQ2_train.csv",header=T)
>Q2.test<-read.csv("SYS6018-FinalQ2_test.csv",header=T)
>names(Q2.train)

[1] "y"  "x1" "x2" "x3" "x4" "x5" "x6"
```

```
>pairs(Q2.train)
>cor(Q2.train)

          y           x1           x2            x3
y   1.00000000  0.05523496  0.05991603  0.672460800
x1  0.05523496  1.00000000  0.66246952 -0.019048281
x2  0.05991603  0.66246952  1.00000000 -0.003959420
x3  0.67246080 -0.01904828 -0.00395942  1.000000000
x4 -0.01440622 -0.03559272 -0.00777172 -0.024008839
x5  0.85525837 -0.01328182  0.02960272  0.788329914
x6  0.08205818  0.93939476  0.69727288  0.003242477
             x4            x5            x6
y  -0.014406218  0.855258373  0.082058182
x1 -0.035592719 -0.013281825  0.939394763
x2 -0.007771720  0.029602719  0.697272876
x3 -0.024008839  0.788329914  0.003242477
x4  1.000000000 -0.008460438 -0.011543897
x5 -0.008460438  1.000000000  0.016426787
x6 -0.011543897  0.016426787  1.000000000
>plot(as.factor(x2),y,data=Q2.train,xlab="x2",ylab="y",col=
"red")
```

CODES. 1

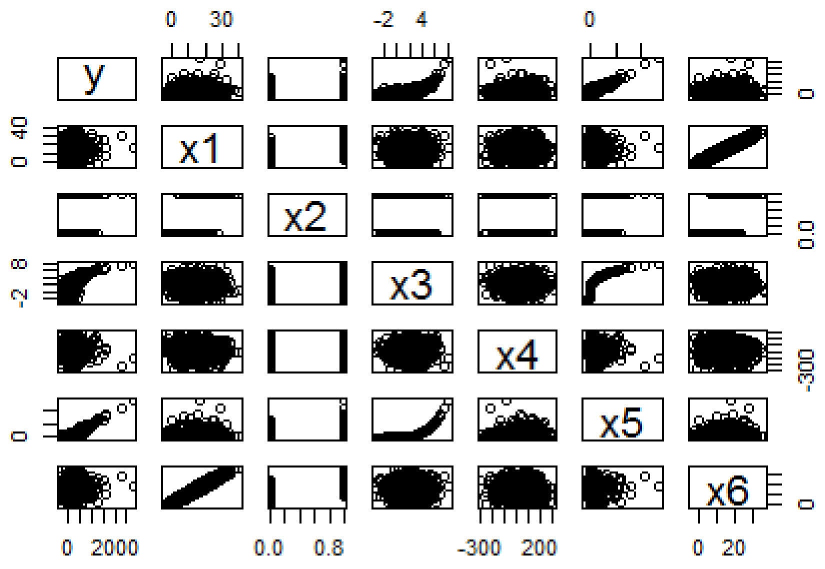

FIG. 2. The matrix of scatter plots of the training data set.

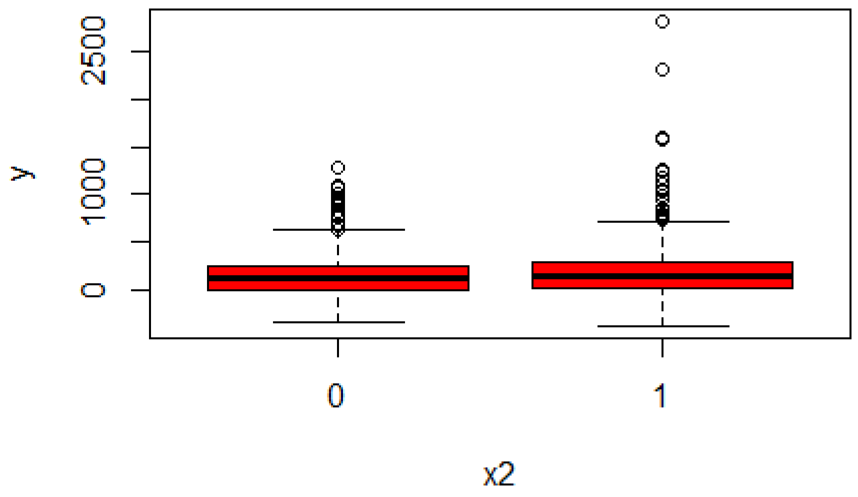

FIG. 3. The boxplot of y as a function of x2 status

b) i. we start the linear regression from y~x1+x2+x3+x4, and adjust the variables by examining their p-value. The final model is (y~x1+x3)

$$y = \beta_0 + \beta_1 x1 + \beta_3 x3 + \varepsilon \, .$$

As illustrated in CODES. 2, the p-value of x4 is 0.874. It is a very large value, which means x4 is irrelevant to y. So it is removed from the predictors. Then in the linear fit of x1, x2 and x3, the p-value of x2 is also relatively large. So x2 is also removed. For the final linear fit of x1 and x3, both of their p-values are less than 0.05, suggesting that x1 and x3 are highly correlated with the value of y.

```
# linear fit with four variables
>linear.fit=lm(y~x1+x2+x3+x4,data=Q2.train)
>summary(linear.fit)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -80.78951   16.97449  -4.759 2.23e-06 ***
x1            1.70228    1.11571   1.526    0.127    
x2           17.20176   17.26589   0.996    0.319    
x3          107.33419    3.72326  28.828  < 2e-16 ***
x4            0.01023    0.06459   0.158    0.874    
# remove x4 variable from fit
>linear.fit=lm(y~x1+x2+x3,data=Q2.train)
>summary(linear.fit)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -80.464     16.842  -4.778 2.04e-06 ***
x1             1.695      1.114   1.521    0.129    
```

```
x2             17.260     17.253   1.000    0.317
x3            107.319      3.720  28.847  < 2e-16 ***
# remove x2 variable from fit
>linear.fit=lm(y~x1+x3,data=Q2.train)
>summary(linear.fit)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -82.2794    16.7435  -4.914 1.04e-06 ***
x1            2.4335     0.8346   2.916  0.00363 **
x3          107.3625     3.7200  28.861  < 2e-16 ***
```

CODES. 2

ii. The statistical significance of the model is listed in the results part of CODES. 2 up.

iii. T-statistic is given by $T = \left(\hat{\beta}_1 - 0\right) / SE\left(\hat{\beta}_1\right)$, and it measures the size of difference in units of standard error. The greater the magnitude of T, the greater the evidence against the null hypothesis, which means the existence of the correlation between the two variables. The T-statistic of x2 equals 1, which is smaller than that of x1 and x3. So the correlation between x2 and y is very weak. In addition, it is a simple matter to compute the possibility of observing any value equal to |T| or larger. This possibility is the p-value. A small p-value indicates that there is an association between the predictor and the response[1]. The p-value of x2 is relatively large. So its correlation with y is negligible.

iv. Here we use the final linear regression model y~x1+x3 to calculate the mean RSS and R^2 for both the training and test data (using CODES. 3). The results are listed in TABLE 1 below.

TABLE 1

	Training data	Test data
Mean RSS	40641.88	41193.54
R^2	0.45684	0.47455

```
# mean RSS R^2 for train data
>fit.pred=predict(linear.fit,Q2.train)
>fit.RSS=sum((fit.pred-Q2.train$y)^2)
```

```
>fit.meanRSS=mean((fit.pred-Q2.train$y)^2)
>y.mean=mean(Q2.train$y)
>fit.TSS=sum((Q2.train$y-y.mean)^2)
>fit.R2=1.0-fit.RSS/fit.TSS
# mean RSS R^2 for test data
>fit.pred=predict(linear.fit,Q2.test)
>fit.RSS=sum((fit.pred-Q2.test$y)^2)
>fit.meanRSS=mean((fit.pred-Q2.test$y)^2)
>y.mean=mean(Q2.test$y)
>fit.TSS=sum((Q2.test$y-y.mean)^2)
>fit.R2=1.0-fit.RSS/fit.TSS
```

CODES. 3

c) i. the structure of the final model is (y~x5+x6) :

$$y = \beta_0 + \beta_5 x5 + \beta_6 x6 + \varepsilon$$

ii. We start from linear regression with y~x1+x2+x3+x4+x5+x6. The x3 has the smallest |T| and the largest p-value, so it is firstly removed. Then repeat this procedure by removing the variable that having the biggest p-value and smallest |T|-statistics. Finally the y~x5+x6 model is obtained. The changes of their respective statistical significance (T-statistic and p-value) are respectively listed in TABLE 2 and TABLE 3.

TABLE 2

T-statistic	Full variables	Remove x3	Remove x4	Remove x1	Remove x2
x1	0.490	0.490	0.517	--	--
x2	-1.119	-1.113	-1.114	-1.098	--
x3	-0.174	--	--	--	--
x4	-0.360	-0.355	--	--	--
x5	32.335	52.311	52.345	52.508	52.492
x6	1.276	1.275	1.256	3.763	4.181

TABLE 3

p-value	Full variables	Remove x3	Remove x4	Remove x1	Remove x2
x1	0.624	0.625	0.605	--	--
x2	0.263	0.266	0.266	0.2724	--
x3	0.862	--	--	--	--
x4	0.719	0.722	--	--	--
x5	<2e-16	<2e-16	<2e-16	<2e-16	<2e-16
x6	0.202	0.203	0.209	1.78e-4	3.16e-5

iii. In the model of y~x5+x6, the T-statistic of x6 is 4.181. Compared with the T-statistic of the removed variables, it is relatively large, which means that the null hypothesis is not true for the relation between x6 and y. In addition, the p-value of x6 is becoming smaller in the process of choosing variables. It suggests that there is an increase in the correlation of x6 and y when the less number of variables are involved in the model fit.

iv. Use the final model y~x5+x6 to calculate the mean RSS and R^2 for both the training and test data. The results are listed in TABLE 4 below. It can be clearly seen that this model performs better than y~x1+x3 does.

TABLE 4

	Training data	Test data
Mean RSS	19746.61	20695.37
R^2	0.73609	0.73602

d) i. After adding the interaction term of x2 & x6, the model shifts from y~x5+x6 to y~x5+x6+x2*x6.

$$y = \beta_0 + \beta_5 x5 + \beta_6 x6 + \beta_{2:6} x2 \times x6 + \varepsilon$$

$$\Leftrightarrow y = \beta_0 + \beta_5 x5 + \beta_6 x6 + \begin{cases} \beta_{2:6} x6 & \text{if } x2=1 \\ 0 & \text{if } x2=0 \end{cases}$$

$$\Leftrightarrow y = \begin{cases} \beta_0 + \beta_5 x5 + (\beta_6 + \beta_{2:6}) x6 & \text{if } x2=1 \\ \beta_0 + \beta_5 x5 + \beta_6 x6 & \text{if } x2=0 \end{cases}$$

It means that 3 kinds of variables: x5, x6, and x2*x6 are involved in the linear regression fit.

ii. After adding the new interaction, the T-statistic of x6 decreases from 4.181 to 2.604 and its p-value increases from 3.16e-5 to 9.35e-3, which means that the correlation between x6 and y is weakened. Before adding the interaction x2*x6, the model of y~x5+x6 and the training data are plotted together in FIG. 4 below. After adding the interaction term, there are two model functions to fit the data set that are respectively one with x2=0 and one with x2=1. These two functions are plotted in FIG. 5 marked by red and blue color mesh. They are very similar and nearly overlap each other. Therefore, the incremental contribution on the forecast of y due to x6 depending on the value of x2 is very small.

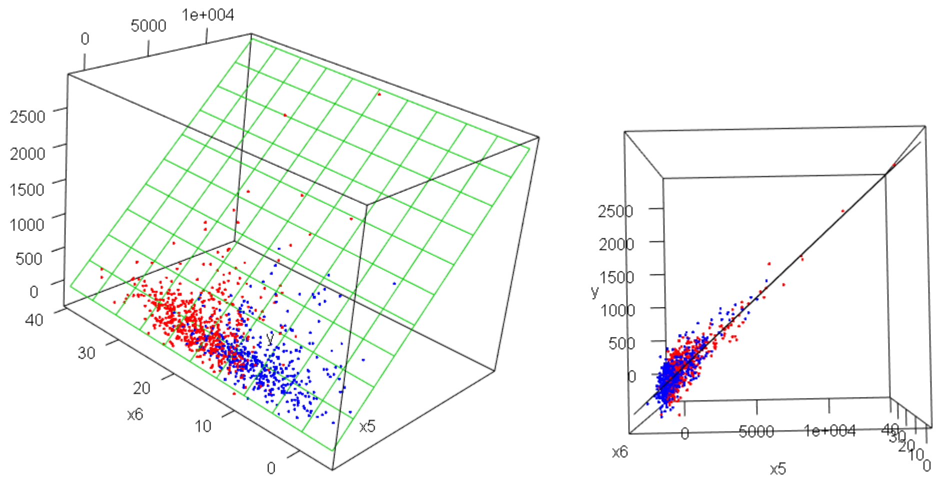

FIG. 4. Two different angles visual inspection of training data (the red points represent y with x2=1 and the blue points represent y with x2=0) and the model of y~x5+x6 (the green mesh plane).

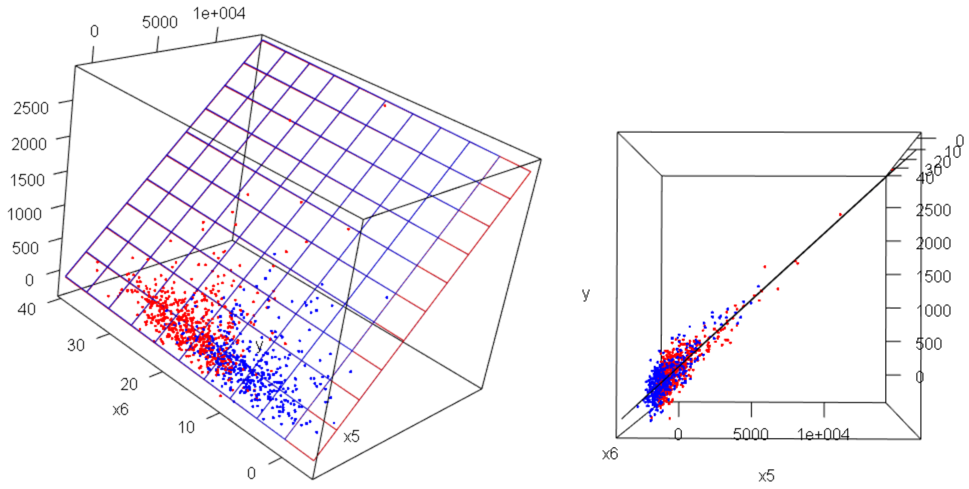

FIG. 5. Two different angles visual inspection of training data (the red points represent y with x2=1 and the blue points represent y with x2=0) and the model of y~x5+x6+x6*x2 (the red mesh plane represents the model under x2=1 and the blue mesh plane represents the model under x2=0).

iii. The mean RSS and R^2 of the new model y~x5+x6+x6*x2 for both the training and test data are listed in below.

TABLE 5

	Training data	Test data
Mean RSS	19742.82	20762.47
R^2	0.73614	0.73516

Question #3

a) The procedures are listed in CODES. 4 below.

```
>library(glmnet)
>trainSet<-read.csv("SYS6018-FinalQ2_train.csv",header=T)
>testSet<-read.csv("SYS6018-FinalQ2_test.csv",header=T)
>numLambda<-300
>x<-model.matrix(y~.,trainSet)[,-1]
>grid=10^seq(-2,6,length=numLambda)
>ridge.mod<-glmnet(x,trainSet$y,alpha=0,lambda=grid)
```

CODES. 4

b) The standardized ridge regression coefficients as a function of λ are plotted in FIG. 6 below.

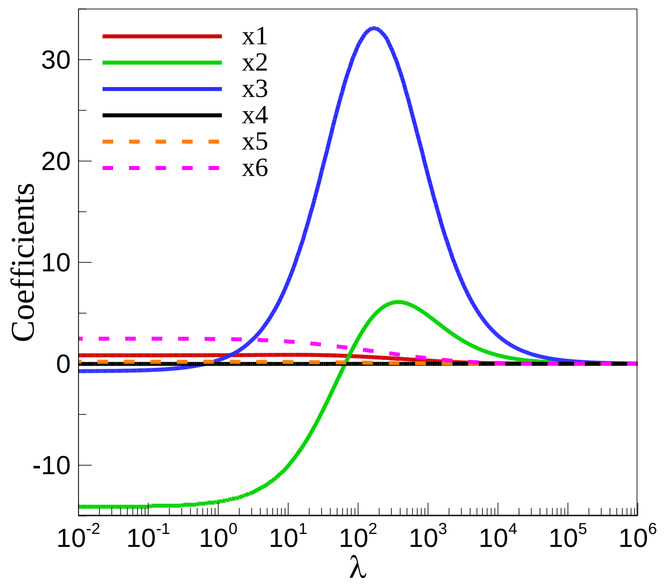

FIG. 6. The ridge regression coefficients as a function of λ.

c) The mean RSS as a function of λ for both the training data and test data sets are plotted in FIG. 7 below.

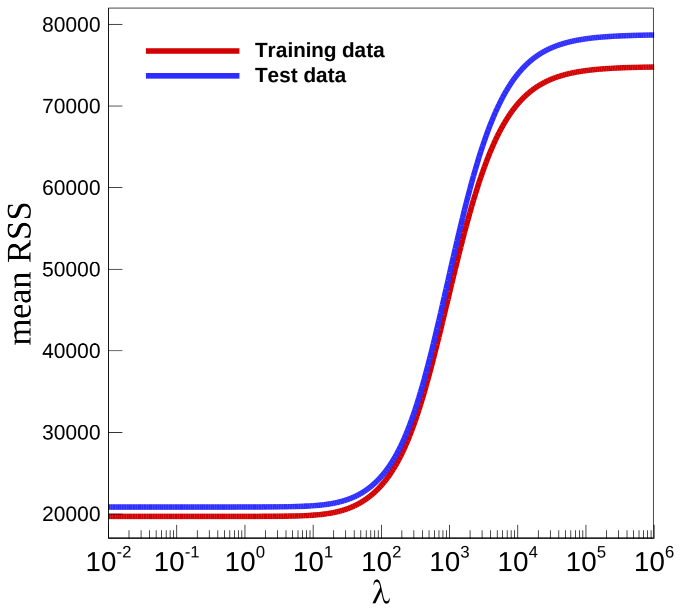

FIG. 7. The mean RSS as a function of λ for the training data (red line) and test data (the blue line).

d) As shown in FIG. 8 below, the cross validation MSE reach the minimum value at **λ=1.3818**, which is the best performing λ. The largest λ of cross validation MSE such that error is within 1 standard error of the minimum is located at λ=22.1046.

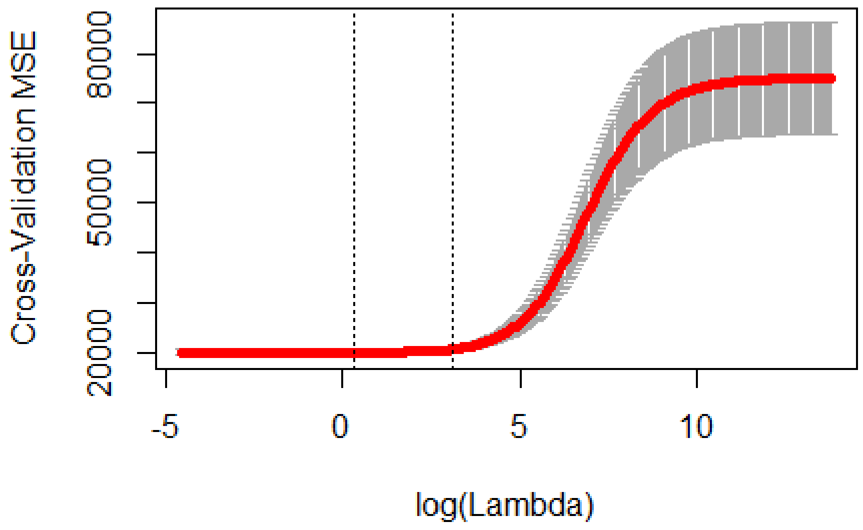

FIG. 8. The cross validation MSE as a function of *λ* for the training data.

If we use "lm.ridge" and set GCV for assessment criteria, then the smallest value of GCV is located at **λ=3.31**. The plot of GCV as a function of λ is shown in FIG. 9.

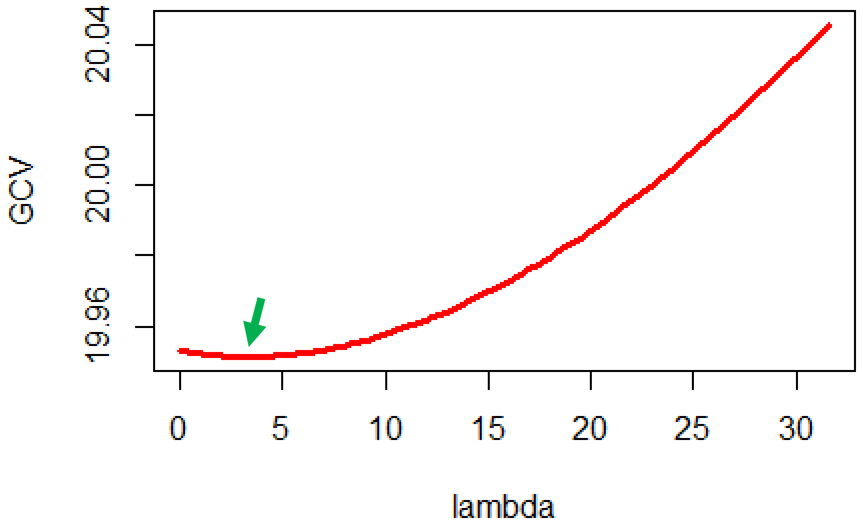

FIG. 9. The GCV (general cross validation) as a function of *λ* for the training data. The green arrow marks the minimum location.

e) For λ=1.3818 ridge regression, the coefficients for each variables are listed in TABLE 6. As expected, none of the coefficients are zero, and ridge regression does not perform variable selection. The performance of the optimized ridge regression and the selected linear regression are compared in TABLE 7 below, which suggests that their performance are very close or similar.

TABLE 6

x1	0.8525	x5	0.1980
x2	-13.4327	x6	2.4299
x3	0.7329	Intercept: 3.0387	
x4	-0.0158		

TABLE 7

	y~x5+x6	ridge regression
Mean RSS	19746.61	19718.12
R^2	0.73609	0.73647

f) In current data set, some independent variables are highly collinear, for example "x1" and "x6". The problem of the collinearity lies in that the variance of the parameter estimate is huge. Ridge regression can reduce this variance at the price of the increasing of the bias. The parameter λ in ridge regression is seen to reduce the complexity of the model. Naturally, with increasing λ, the training error grows as the residual sum of squares becomes larger. However, the test error will reach the minimum at some optimal λ before it increases with the growing λ.

Question #4

a) According to the visual inspection of the training and test data in FIG. 10, we can clearly see that the relation model between the explanatory variable ‘X’ and the target variable ‘Y’ may be **sine or nonlinear** function rather than linear function. Here we can use the polynomial or spline regression. Considering the polynomial regression must use a high degree to produce a flexible fit, the spline will be a better choice since it can increase flexibility by adding the knots with a fixed degree and so is relatively stable[1]. In addition, the statistic summary of training set (in CODES. 5) shows that the observations are distributed in the region centered at near (X=0.0, Y=0.2).

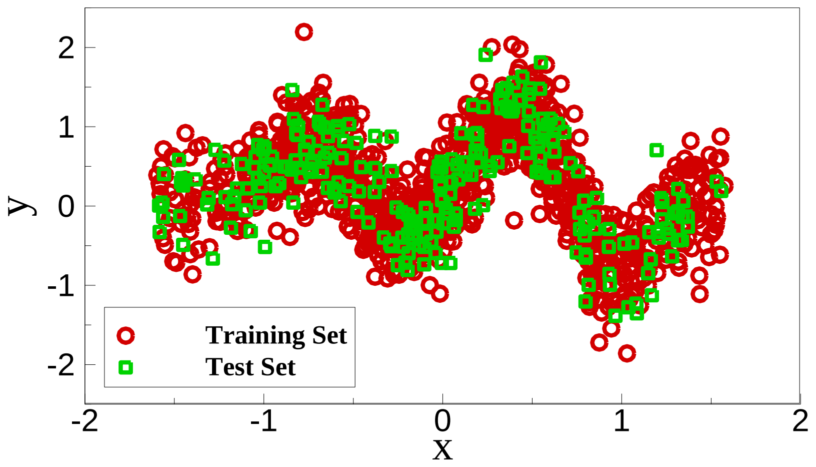

FIG. 10. Visual inspection of training (red points) and test (green points) data sets.

```
>trainset<-read.csv(“SYS6018-FinalQ4_train.csv”,header=T)
>testSet<-read.csv(“SYS6018-FinalQ4_test.csv”,header=T)
>summary(trainset)
       y                  x
 Min.   :-1.8585   Min.   :-1.59520
 1st Qu.:-0.2318   1st Qu.:-0.59956
 Median : 0.2427   Median : 0.00171
```

```
Mean    : 0.2495    Mean    : 0.02329
3rd Qu.: 0.7137    3rd Qu.: 0.61665
Max.    : 2.6564    Max.    : 1.57498
```

CODES. 5

b) i. The evolution of the MSE as a function of number of knots in cubic spline models (using the training set) is shown in FIG. 11 below.

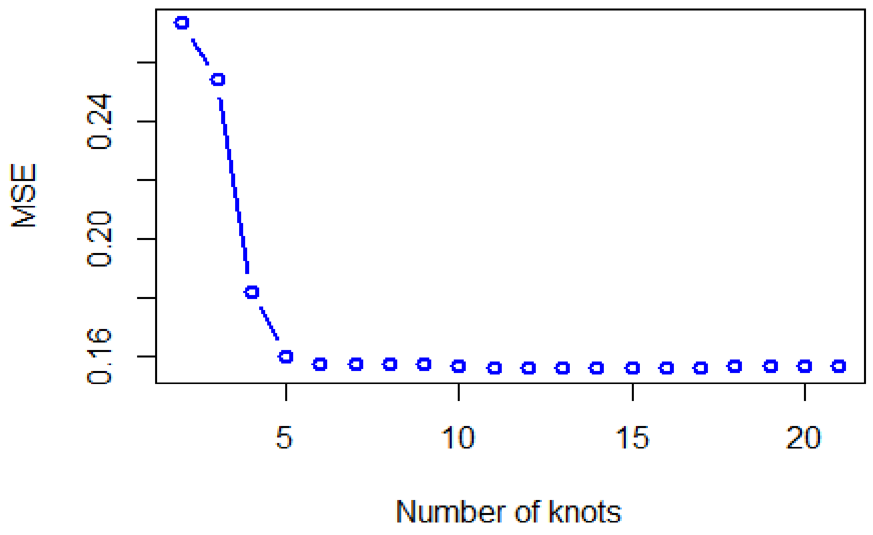

FIG. 11. The evolution of the mean square error as a function of the number of knots using the training data set.

ii. Using the test data set, the evolution of the MSE as a function of number of knots is shown in FIG. 12 below. The best model lies in the **6 knots** with the smallest mean squared error = 0.1435 as marked by the red arrow.

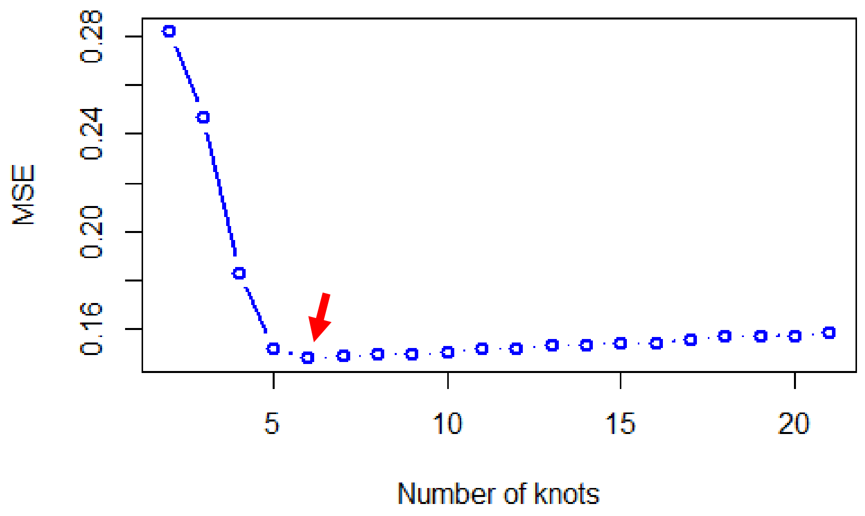

FIG. 12. The evolution of the mean square error as a function of the number of knots using the test data set.

iii. The plot of the best model is shown in FIG. 13.

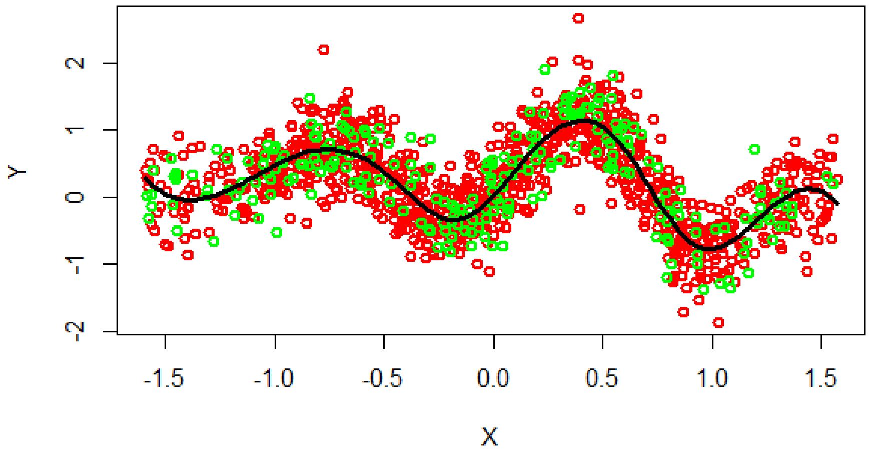

FIG. 13. The 6-knots cubic spline (black solid line) fitted using training data (red points). Green points represent the test data.

iv.

c) i. Use “anova” to compare 6-Knots cubic spline with degree-5 polynomial in CODES. 6. Their plots are compared in FIG. 14. Here F-statistic is 260.89 and p-value is near zero. The RSS of cubic spline model is 155.33 much less than that of the 5-degree polynomial model. Therefore, the 6-Knots cubic spline is superior to the 5-degree polynomial model.

```
>fit=lm(y~bs(x,df=3+6),data=trainSet)
>fit.poly5=lm(y~poly(x,5),data=trainSet)
>anova(fit.poly5,fit)

Analysis of Variance Table

Model 1: y ~ poly(x, 5)
Model 2: y ~ bs(x, df = 3 + 6)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    994 319.07
2    990 155.33  4    163.74 260.89 < 2.2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

CODES. 6

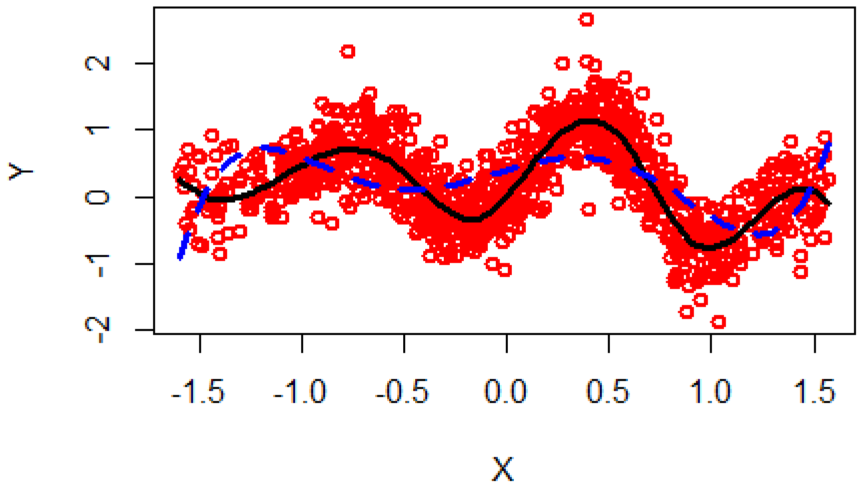

FIG. 14. The 6-knots cubic spline (black solid line), the training data set (red points) and the 5-degree polynomial regression (blue dashed line).

ii. Compare 6-Knots cubic spline with 6-degree smoothing spline in CODES. 7. The RSS of the latter model is 233.47 which is larger than the RSS=155.33 of the former model. Therefore the 6-Knots cubic spline is superior to the 5-degree smoothing spline. The same conclusion can be made by comparison of their plots in FIG. 15 below.

```
>fit.smooth=smooth.spline(trainSet$x,trainSet$y,df=6)
# RSS of the 6-degree smoothing spline
>smooth.pred=predict(fit.smooth,trainSet$x,se=T)
>sum((smooth.pred$y-trainSet$y)^2)

[1] 233.4728
```

CODES. 7

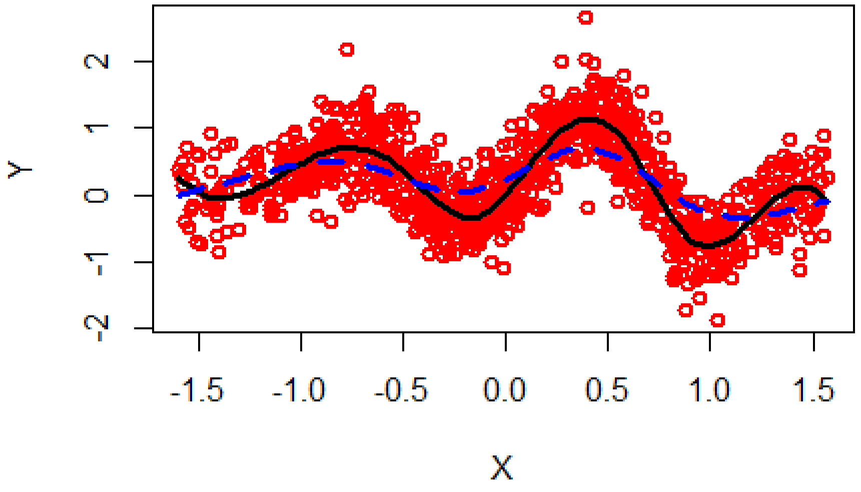

FIG. 15. The 6-knots cubic spline (black solid line), the training data set (red points) and the 6-degree smoothing spline (blue dashed line).

iii. Compare 6-Knots cubic spline with local regression (span=0.2) in CODES. 8. The training RSS of the latter model is slightly less than that of the former model. Their test RSS also are compared. The local regression has RSS=35.410, and cubic spline has RSS=35.598. Therefore, these two models have very similar performance, and the local regression (span=0.2) might be slightly better than the 6-Knots cubic spline, which is supported by their almost overlapped plots in FIG. 16 below.

```
>fit.local=loess(y~x,span=0.2,data=trainSet)
# the training RSS of local regression (span=0.2)
>local.pred=predict(fit.local,data.frame(x=trainSet$x))
>sum((local.pred-trainSet$y)^2)
```

```
[1] 153.6628
# the test RSS of local regression (span=0.2)
>local.pred=predict(fit.local,data.frame(x=testSet$x))
>sum((local.pred-testSet$y)^2)

[1] 35.41032
```

CODES. 8

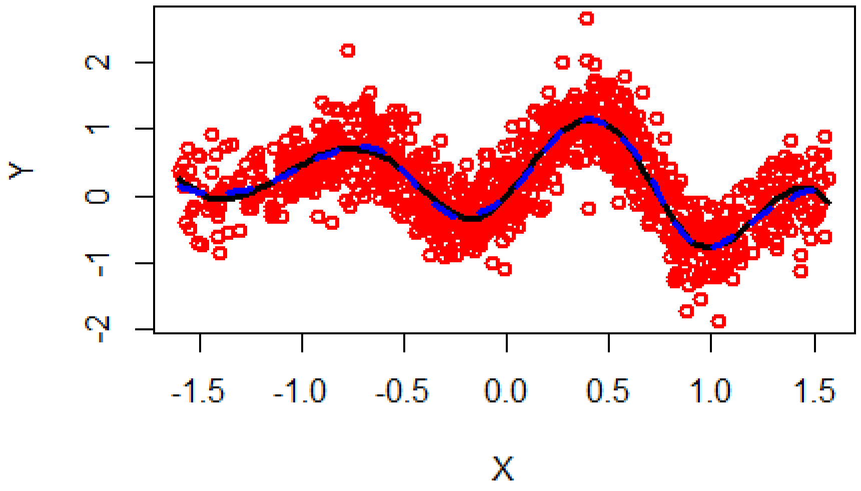

FIG. 16. The 6-knots cubic spline (black solid line), the training data set (red points) and the span=0.2 local regression (blue dashed line).

d)

Question #5

a) Compared with the bagging approach, the random tree has the advantage in reducing variance by way of a small tweak that decorrelates the trees. In the bagging approach, if one predictor has a very strong correlation with the response, most or all of the trees will use this strong predictor in the top split. So, all of the bagged trees will look quite similar to each other[1]. Averaging many similar quantities has little contribution in reducing variance. The random tree fixes this problem by forcing each split to consider only a subset m of the predictors, usually m=$\sqrt{p}$, which results in many uncorrelated trees and consequent low variance.

b) If bagging results and random have similar prediction results for the same given dataset, it suggests that there are few strong predictors that has high correlation with the responses in the dataset. In other words, most or all of the predictors have moderate influence on the prediction of the test response.

c) The importance of each predictor can be obtained by recording the total amount that the RSS is decreased due to splits over a given predictor, averaged over all B trees. A large value indicates an important predictor[1].

Question #6

a) Use ‘princomp’ to compute the principal components of the data set. The commands and results are listed in CODES. 9 below.

```
>OSdata<-read.csv("SYS6018-FinalQ6.csv",header=T)
>names(OSdata)

[1] "Price"   "Software"   "Aesthetics"  "Brand"   "OS"
# use princomp to generate PCA
>PC.OSdata=princomp(OSdata[,-5],scale=T)
>PC.OSdata$loadings

Loadings:
           Comp.1 Comp.2 Comp.3 Comp.4
Price      -0.596 -0.379  0.706       
Software   -0.109 -0.834 -0.540       
Aesthetics  0.605 -0.268  0.318 -0.679
Brand       0.517 -0.298  0.329  0.732

               Comp.1 Comp.2 Comp.3 Comp.4
SS loadings      1.00   1.00   1.00   1.00
Proportion Var   0.25   0.25   0.25   0.25
Cumulative Var   0.25   0.50   0.75   1.00
# Principle components of the dataset (16 observations)
>PC.OSdata$scores
```

```
        Comp.1       Comp.2     Comp.3      Comp.4
 [1,] -2.336166 -0.027631589  0.6113487  0.42371803
 [2,] -4.353369  2.126750466  1.4228309 -0.37065439
 [3,] -1.105703  0.240638317  1.7981464  0.49789879
 [4,] -3.684711 -0.484034838 -2.1400230  1.05860044
 [5,] -1.421810 -2.908279705  1.2020491 -0.29523406
 [6,] -3.349535  1.372624188  0.5048867  0.39157212
 [7,] -4.112641 -0.154582174 -2.4794655 -1.08455516
 [8,] -1.730865 -0.295142658  0.9292878 -0.25522494
 [9,]  2.816880 -0.589755107  0.4318368  0.73662089
[10,]  3.797577  2.165460098 -0.2402044 -1.26222463
[11,]  3.304099 -1.045399519 -0.8147982  0.76674108
[12,]  1.496939 -2.984546851  0.7536837 -0.81873847
[13,]  2.399278  1.189119099 -0.3810924  0.75561259
[14,]  1.783650  0.007208626 -0.2255448 -0.72759173
[15,]  2.261334  0.197693712 -2.4965860  0.03260905
[16,]  4.235042  1.189877936  1.1236443  0.15085038
```
CODES. 9

b) For component 3, the expression to represent the original data set (x_1 ="Price", x_2 ="Software", x_3 ="Aesthetics", x_4 ="Brand") is :

$$Z_3 = \phi_{13} X_1 + \phi_{23} X_2 + \phi_{33} X_3 + \phi_{43} X_4 \ ,$$

Where X_i represent the normalized x_i by $X_i = x_i - center_i$. $center_i$ is listed in CODES. 10 below.

```
# The centers used to normalize the original variables
>PC.OSdata$center

    Price    Software Aesthetics      Brand
   4.1875      5.0625     4.5000     4.6875
```
CODES. 10

Therefore, the expression for component 3 is

$$Z_3 = 0.706 \times (x_1 - 4.1875) - 5.4 \times (x_2 - 5.0625)$$
$$+ 0.318 \times (x_3 - 4.5) + 0.329 \times (x_4 - 4.6875)$$

c) Use the components 1 and 2 to fit QDA model as shown in CODES. 11 below. The error rate is **6.25%**, which means that QDA performs very well in prediction or classification of whether individuals purchase OS or not.

```
>library(MASS)
# combine components 1 & 2 with the OS
>OSdata.PC12=PC.OSdata$scores[,1:2]
>OSdata.PC12=data.frame(OSdata.PC12,OSdata$OS)
>names(OSdata.PC12)

[1] "Comp.1"    "Comp.2"    "OSdata.OS"
>qda.fit=qda(OSdata.OS~Comp.1+Comp.2,data=OSdata.PC12)
>qda.pred=predict(qda.fit,OSdata.PC12[,1:2])$class
>table(qda.pred,OSdata.PC12$OSdata.OS)

qda.pred 0 1
       0 7 0
       1 1 8
```

CODES. 11

d) Use components 1, 2 &3 in QDA model as shown in CODES. 12 below. The error rate remains **6.25%** as the same as that in c).

```
# combine components 1, 2 & 3 with the OS
>OSdata.PC123=PC.OSdata$scores[,1:3]
>OSdata.PC123=data.frame(OSdata.PC123,OSdata$OS)
>names(OSdata.PC123)

[1] "Comp.1"    "Comp.2"    "Comp.3"    "OSdata.OS"
>qda.fit=qda(OSdata.OS~Comp.1+Comp.2+Comp.3,OSdata.PC123)
>qda.pred=predict(qda.fit,OSdata.PC123[,1:3])$class
>table(qda.pred,OSdata.PC123$OSdata.OS)

qda.pred 0 1
       0 7 0
       1 1 8
```

CODES. 12

e) Perform QDA on the original features in CODES. 13 below. The accuracy rate is 100%, which performs better than previous two approaches.

```
>qda.fit=qda(OS~.,data=OSdata)
>qda.pred=predict(qda.fit,OSdata[,1:4])$class
>table(qda.pred,OSdata$OS)

qda.pred 0 1
       0 8 0
       1 0 8
```

CODES. 13

f) The QDA model in d) with three principle components has the identical performance with the model in c) with two components. It is because the first two principle components have contained the most information of the predictor variables. So adding the third principle component provides little improvement in the prediction. The QDA with full original features in e) performs better the previous models with PCA because the PCA reduces the dimension of the variables and thus has missed part of variables information.

PART #2 OPEN ENDED ANALYSIS

a) Here we apply vector generalized linear model (VGLM) to forecast the expected rating based on a set of observed characteristics. The codes are listed in CODES. 14. The training data is divided randomly into training set and test set. The training set is used to fit vglm. The **test error rate** is **49.83%.**

```
>library(VGAM)
>OEA.train<-read.csv("SYS6018-FinalOEA_train.csv",header=T)
>OEA.test<-read.csv("SYS6018-FinalOEA_test.csv",header=T)
>train=sample(nrow(OEA.train),0.7*nrow(OEA.train))
>EA.vglm=vglm(quality~.,data=OEA.train[train,-1],family=cumu
lative(link="logit",parallel=TRUE))
>OEA.pred=predict(OEA.vglm,newdata=OEA.train[-train,-1],type
="response")
>OEA.pred.quality=apply(OEA.pred,1,which.max)+2
>table(OEA.pred.quality,OEA.train$quality[-train])

OEA.pred.quality   3   4   5   6   7   8   9
               4   1   1   1   0   0   0   0
               5   5  29 167 110  15   4   0
               6   2   8 177 389 175  19   0
               7   0   0   2  42  44   7   1
               8   0   0   0   0   0   1   0
```

CODES. 14

b) Then we use random forest model to fit the training set and predict the test set as shown in CODES. 15. The **test error rate** is **21.42%**, which is much lower than that of the previous linear regression. It suggests that random forest classification performs much better in prediction of wine qualify at this data set than the vector generalized linear model.

```
>library(randomForest)
>OEA.train$type=ifelse(OEA.train$quality<=5,"below",ifelse(O
EA.train$quality<8,"above","exceptional"))
>OEA.type.rf=randomForest(as.factor(type)~.,data=OEA.train[t
rain,-c(1,13)],mtry=4)
>type.pred=predict(OEA.type.rf,newdata=OEA.train[-train,-c
(1,13)],type="response")
>table(type.pred,OEA.train[-train,14])

type.pred     above below exceptional
  above         673   134          31
  below          91   258           0
  exceptional     1     0          12
```

CODES. 15

If we use this model to predict the quality of one wine, the predicted quality is the most likely quality of the wine. It may belongs to other kinds of quality. In order to calculate the probability of this kind of situations, we further provide the random forest classification on the "quality" factor as shown in CODES. 16. Then the probabilities of one prediction for different kinds of quality,

$$P_{ij} = Probabibity(prediction\ quality = i\ |actual\ quality = j),$$

can be given on TABLE 8.

```
>OEA.quality.rf=randomForest(as.factor(quality)~.,data=OEA.t
rain[train,-c(1,14)],mtry=4)
>quality.pred=predict(OEA.quality.rf,newdata=OEA.train[-trai
n,-c(1,14)],type="response")
>table(quality.pred,OEA.train[-train,13])

quality.pred   3   4   5   6   7   8
           3   0   0   0   0   0   0
           4   0   8   0   0   0   0
           5   4  27 218  85   2   0
           6   3  15 115 429 100  12
```

```
       7    0    0    2   46  101   19
       8    0    0    0    1    1   12
       9    0    0    0    0    0    0
```

CODES. 16

TABLE 8

Predicted Quality	Actual Quality					
	3	4	5	6	7	8
3	100%	--	--	--	--	--
4	--	100%	--	--	--	--
5	1.19%	8.04%	64.88%	25.30%	0.60%	--
6	0.44%	2.23%	17.06%	63.65%	14.84%	1.78%
7	--	--	1.19%	27.38%	60.12%	11.31%
8	--	--	--	7.14%	7.14%	85.71%

c) We use the random forest model trained at the b) section to predict the quality of the wines on the test list for coming year purchase. The predicted quality ranges from 4 to 8. Each kind of quality has its own potential profit as well as the corresponding probability. They are presented in FIG. 17. The wine with quality=4 will definitely bring about the negative profit (=-5) for the wine merchant. The wine of quality=6 or 7 corresponds to three potential kinds of profit with respectively three different probability. It can be found that, except quality=8, all other kinds of quality wine has the possibility to result in negative profit to the wine merchant, which means losing money. Therefore, it is very clear that the merchant should try to choose the wine with predicted quality=8 for the coming year business.

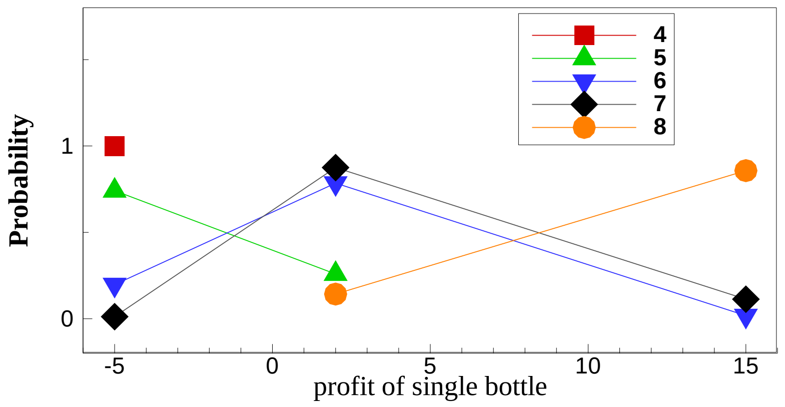

FIG. 17. The probability of each potential profit of single bottle wine for different predicted quality (from 4 to 8).

Another advantage of this purchasing strategy is attributed to that the expectation profit of the 8-quality wine is much higher than other kinds of quality wines as shown in FIG. 18, which suggests that purchasing the 8-quality wine will maximize the benefits. The expectation profit is calculated by the equation below:

$$\text{expectation profit} = \sum_i profit_i \times probability(profit_i) \ .$$

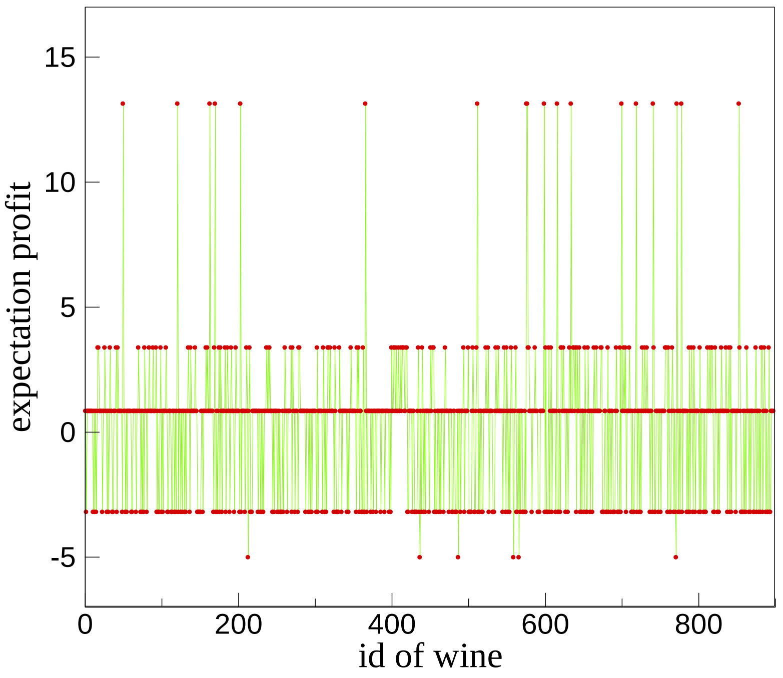

FIG. 18. The expectation profit (of single bottle) as a function of the wine id in the test data.

There are totally 18 kinds of 8-quality wine. Now another problem is coming: should the merchant purchase one or some kinds of 8-quality wine or all kinds of them? The expectation profit will not change no matter which kind or how many kinds of the 8-quality wine are chosen because each kind of 8-quality wine has the identical expectation profit, but the variance of profit will be influenced. If two eight or eighteen kinds of wine are chosen, the normalized profit-probability distributions are shown in FIG. 19. the normalized profit is given by the equation below:

$$normalized\ profit = total\ profit/total\ number\ of\ bottles\,.$$

From FIG. 19, we can see that when more kinds of wind are involved in purchase, the probability for low normalized profit will decrease. In other words, the plenty types of wine will make the potential actual profit more likely approach to the expectation value. So the risk of the low profit is eliminated.

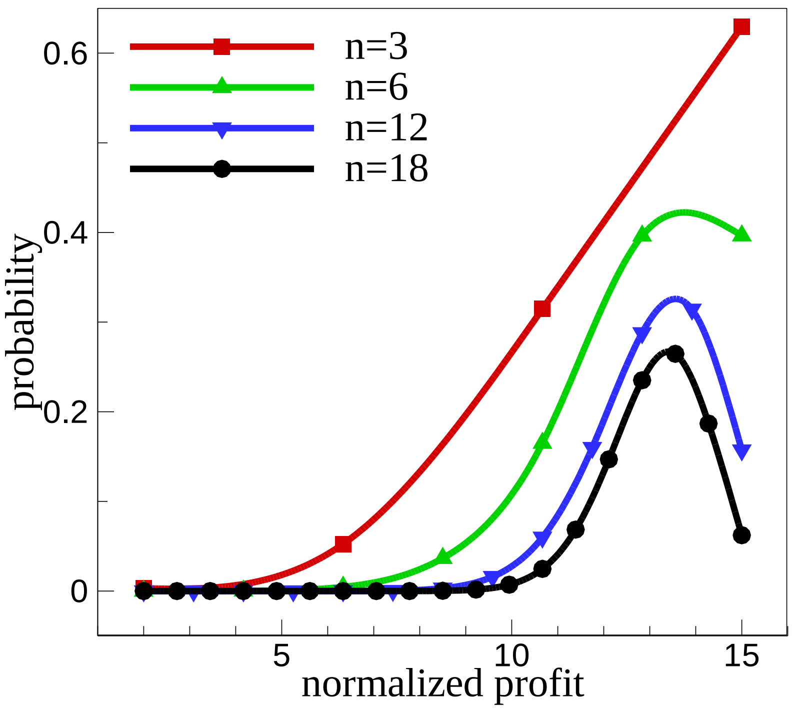

FIG. 19. The normalized profit-probability distributions with different number of 8-qualify wine being chosen.

In conclusion, the optimal strategy is to purchase the same amount of the 18 kinds of 8-quality wines. It will give the merchant a stable high expectation profit with relatively small risk. If the budget is $25,000, then the expectation profit of this strategy is $32,855.25.

1. James, G., D. Witten, and T. Hastie, *An Introduction to Statistical Learning: With Applications in R*. 2014, Taylor & Francis.