Stat 849 Final Project

Xiang Zhong

Main Findings:

For linear regression, transformation is needed on response “daystolastfollowup” to satisfy modeling assumption. High multicollinearity is observed for all the data sets. To deal with multicollinearity, AIC/BIC and lasso regression are used for variable selection. Results show that models with smaller number of genes selected have smaller Adjusted-R square, but the multicollinearity is better mitigated. Combining two data sets will improve the model in respect to a larger adjusted R-square and the smaller mean prediction error when doing cross-validation. The final gene sets selected using combine data sets is slightly different from those selected using each set separately. Using Lasso regression, the final gene set seems more consistence (More detail analysis in following sections). For generalized linear model, GMC is applied to test the variance of response explained by the data set. Five functions are choose and the quadratic function can best explain the variance with the highest GMC value produced. In general the ability of X*beta (fitted) to explain Y (response) is not good, however more investigation might be needed. For logistic regression, Pearson residuals are plotted, and the Pearson Statistic and G^2 are good for all the models. Check the estimated coefficient for those genes which are significant in the model, none of the selected genes behave like having a strong impact on the response (survival above certain percentile), the coefficients of these estimates are nested around (-1, 1) for most of the cases. AUC are calculate to evaluate the prediction ability. The AUC of those logistic regression models are ranged from 0.66 to 0.78, in general the prediction accuracy is not too bad but still not that satisfied.

Data Description:

There are 593 observations for this data set, only 558 observations are non-identical and with available daystolastfollowup response. Of those 558 patients, 293 of them with vitalstatus being 1. After variable matching, 426 genes are selected and 213 (non-overlapping) for (A, B) correspondingly. The name of genes are coded as index (X1, X2,…, X213) for each set. The index mapping is sequentially as in original file.

Linear Regression Models:

Set A:

1) Fitting the model:

First, fit the multiple linear regression model with all genes in set A. The diagnose plots show that the homogeneity and the normality are not good. There is obvious pattern of the residuals plot. (See Appendix Figure 1.a.) Thus transformation is needed. Several transformations on Y have been tested and sqrt(y) is chosen.

2) Diagnostic:

From the residuals plot it could be found that homogeneity is improved. (See Figure 1.b.) Using outlierTest() in R, the result shows that no suspicious outlier is found. There are several influential points judging by Cook’s distance, but it is not necessary to remove them. Calculate the VIF value of each covariate, high collinearity is presented (See Figure 1.c.). So remedies are needed to deal with multicollinearity with large number of variables. Variable selection or some shrinkage methods can be applied.

3) Variable Selection:

Subset selection using AIC/BIC is performed. First, conduct backward, forward and stepwise selection using AIC, three candidate models are generated. Second, use cross-validation to choose one among the three using mean prediction error as the standard. Similar procedures are applied to subset selection using BIC. Besides, shrinkage method is tested. Here to reduce number of variables, Lasso regression is chosen. Noted that y is not transformed for Lasso regression.

4) Final Models:

Comparing the cross validation results, the forward selection model using AIC is selected as the first model for set A (model a1), stepwise selection model using BIC is selected (model a2). (The diagnostic plots are shown as Figure1.d. and Figure 1.e., the homogeneity is good, the normality is not too bad.) The multicollinearity is mitigated (see plotted VIF value Figure 1.f.) and the number of covariates are reduced significantly. (model a1 has 78 genes and model a2 has 20 genes included.) For Lasso regression (model a3), the optimal lambda is selected using cross validation as a standard and 14 genes are retained.

5) Comments:

For model a1, more genes are selected thus the corresponding fitting is good. R-square is 0.6446, adjusted R-squared is 0.5151. For model a2, fewer genes are retained thus R-values (0.3777, 0.3319) are sacrificed. Lasso regression keeps the least number of variables and the corresponding R-square (or the deviance explained) is only 0.132. So all three models have their pros and cons. But in general, the multicollinearity is mitigated, and we can expect a stable estimation of all the coefficients in those models.

Set B:

1) Fitting the model:

Similar problems are observed for directly regress response on all genes. Transformation sqrt(y) is chosen.

2) Diagnostic and Variable Selection:

Diagnostic plots are shown in Figure 2.a. and no suspicious outlier is found. Influential points are found but still kept in the model. Besides, high collinearity is presented (See Figure 2.b.). So remedies are needed to cope with multicollinearity. Same variable selection procedures are used as in set A.

3) Final Models and Comments:

Comparing the cross validation results, the forward selection model using AIC is picked (model b1), backward selection model using BIC is selected (model b2). (The diagnostic plots are shown as Figure2.c. and Figure 2.d., the homogeneity and normality are satisfied.) The multicollinearity is mitigated (see plotted VIF value Figure 2.e.) and the number of covariates are reduced. (model b1 has 78 genes and model b2 has 27 genes included.) For model b1, R-square is 0.668, adjusted R-squared is 0.547. For model b2, R-values (0.4434, 0.3866). For Lasso regression (model b3), 39 genes are retained and the R-square (or the deviance explained) is 0.335. The properties of these models are similar to those achieved in set A.

Combine Set A and B:

1) Data sets:

If we keep all the 426 genes, there won't be enough observations to fit a model. So only the genes of kept in model a1 and b1 are selected (156 in total). The reason for doing this is to keep more genes for further variable selection.

2) Fitting, Diagnostic and Variable Selection:

Similar transformation sqrt(y) is applied. Diagnostic plots are shown in Figure 3.a., multicollinearity is even severe (See Figure 3.b.). Same procedures as used in set A.

3) Final Models and Comments:

Comparing the cross validation results, the stepwise selection model using AIC is picked (model c1), backward selection model using BIC is selected (model c2). (The diagnostic plots are shown as Figure 3.c. and Figure 3.d., the homogeneity and normality are satisfied.) The multicollinearity is not entirely mitigated (see plotted VIF value Figure 3.e.). Model c1 has 98 genes and model c2 has 52 genes included.) R-square and adjusted R-squared are (0.818, 0.717) and (0.668, 0.621), correspondingly. For

Lasso regression (model c3), 34 genes are retained and the R-square (or the deviance explained) is 0.35. In general, combing two sets generate larger R-square, and have less mean prediction errors when doing cross validation. However, the multicollinearity is not entirely eliminated due to large number of covariates retained in the model.

Summary for linear regression model:
The selected genes for each model are summarized in appendix. For each set, there are overlapping among genes from different models reported but none of the models contain exactly the same genes. It could be found that when combining two sets, gene selection using AIC or BIC is not consistence. Those genes which are retained in models using A, B set data only might no longer significant. However, when using Lasso regression, the results is much more consistence. The meaningful genes in the original model 1c and 2c are mostly still important in model 3c, as all of them are fitted using Lasso regression. This is nature because combine A and B might introduce more collinearity while Lasso regression could better handle this in this study.

Generalized linear models:
To test correlation between the response and the fitted value, GMC is applied for generalized linear models. Since GMC (Generalized Measures of Correlation) is motivated to test asymmetries in explained variances, and has the advantage that does not require the linearity in the relation of two random variables, it is very suitable for generalized linear model analysis. To further analyze the correlation, five g-functions are defined. I also test some other common link functions like "logit" and "probit" used in generalized linear regression, but the result turns out to be worse. A list of g-function are shown below.

1) G1(x) =x (this is to emulate multiple linear regression)
2) G2(x) =x^2 (this is to emulate the sqrt transformation in response)
3) G3(x) =exp (x) (this is to emulate possion regression)
4) G4(x) =sin (x) (this is test the effect of periodic/non monotone function)
5) G5(x) =x^3 (this is to emulate the cubic transformation in response)

Procedures:

0) Choose an initial X, and keep the same number of variables as previous fitting.
1) Choose an initial beta. One way is to fit the generalized linear model to estimate the initial beta (e.g, if choose g=G1(x), then fit the linear regression model without intercept (using lm in R), if choose g=G3(x), then fit the Poisson regression (using glm in R), using the corresponding estimates as the initial beta for optimization).
2) Normalize the data, both response Y and the X*beta, and calculate the GMC(Y|g(X*beta)), use optim in R to choose the best beta to get the maximized GMC value.
3) Change X and redo step 1)-2)

R-codes for these procedures are provided in appendix.

Results:
Among Set A, the best GMC is achieved when g is G2(x), the quadratic function, with GMC=0.35. Among Set B, the best GMC is achieved when G5(x) the cubic function is taken, GMC=0.35. For the combine set, optimal GMC= 0.59 and the optimal function is G2(x) the quadratic form. The changing trends are in consistence with the reported Adjusted-R square in multiple linear regression, but all the GMC values in general are smaller than Adjusted-R square. The optimal combination of genes is the same as the results from linear regression model.

Limitation and Analysis:

Due to time limit, only 200 random draw samples are tested for each data set, so the optimal might not be the global optimal since genes are randomly picked, however, it still can reflect some properties of the data. First, in general, those selected genes have limit power to explain the variations in response, the data set is not informative and shows small GMC in all case. Secondly, the g function is important to reflect the real relationship. For example, a periodic function generally cannot give a larger GMC value in all case, so does the Probit link function and the logit link function (although I didn't list them as g, I did several tests on them). However, quadratic and cubic perform well, this indicates a natural relationship between response and fitted value might be quadratic or cubic, which is consistence as the linear regression with transformation on response y. It proves that GMC has the ability to test the correlations in nonlinear form. However, previously I run multiple simulations with response Y and g(X*beta) not being standardized, the GMC is always close to 1 when Y and g(X*beta) are not in the same scale. This indicates that in order to get accurate correlation, the variables should be standardized, and otherwise the high GMC value is not meaningful.

Remedies and Future Work:
Firstly, the gene selection procedure could be optimized, for here it is just randomly choose certain number of gene from the corresponding set. However, if using some domain knowledge from previous linear model, an advance procedure targeted at picking those more meaningful genes could be proposed. Otherwise, more iteration is needed to generate a more confident results (e.g. have 5000 random draw). Secondly, the g function tested here are limited to a small range of functions and most of them are monotone function, it would be meaningful to test other non-monotone functions or functions with more than one peak to test the ability of GMC. Third, in respect to gene selection, I choose the model with less number of genes selected to test GMC for each data set. Due to time limit I haven't test the effect of difference in k (number of gene picked). The improvability of including more genes is under investigation.

Logistic regression models:

Data and Models:
To get the response, find the quantiles (0.25, 0.5, 0.75) for those patients with vitalstatus =1, and the corresponding survival date is (493, 982, 1451). Next, for the 558 patients, convert the daystolastfollowup to binary variable (0, 1) using the 25, 50, 75 percentiles as the threshold. Then we have the data for logistic regression. Using the variables selected by BIC and crossvalisation, three sets of genes are used to build logistic models for set A, B and the combination, separately.

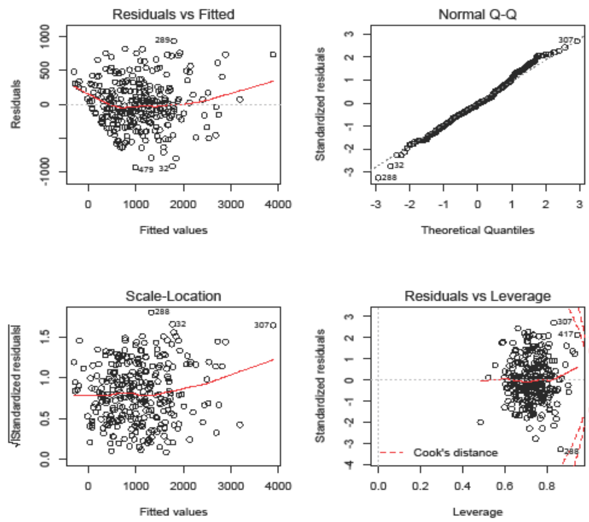
Diagnostics:
The model fitting results are summarized in appendix. The diagnostics for logistic regression are different from those for OLS regression. A customize written R function examine.resid(model.fit) is provided in Hosmer, D. & Lemeshow, S. (2000)'s book and is used to test the models here. In general the Pearson statistic and G^2 are good for all models, the p-value for those models are close to zero in all case. From the residual plots, it could be found there are several points with pearson residuals larger than 2 but the entire fitting is acceptable.

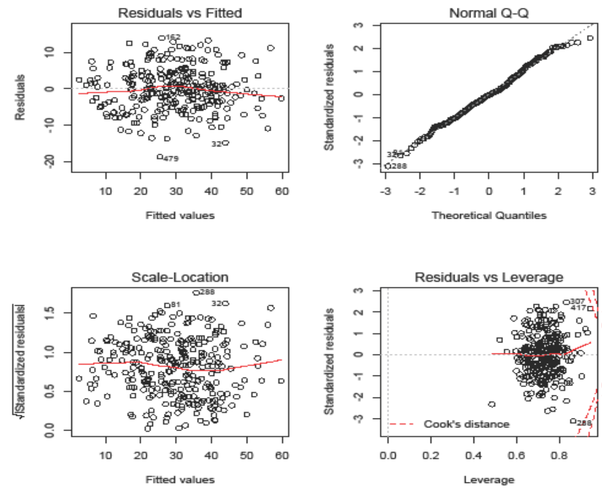
Results and Analysis:
For models in each data set, the significant genes are different when using different responses. In set A, only X80, X127 are significant in all three models. In set B, X4, X31, X41, X124, X161, X163, X180 are significant in all three models. In combine set, X99, X204, X56 and X111 from set A are significant in all three models and X131 form set B is the only one keeping significant in all three models. Compare the

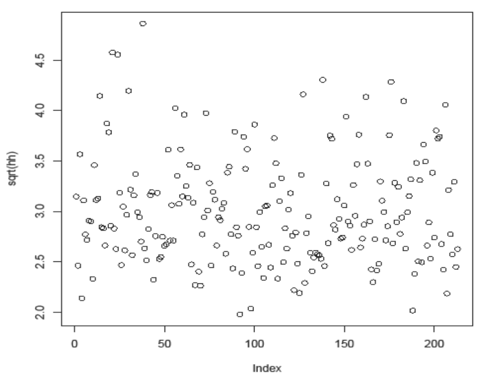
fitted coefficient values, there is no obvious trends with the estimated value. i.e., some of the estimated value increase while some of them decrease, for fitting to different responses. However, the sign of the estimates are consistence, i.e., those genes who have positive effects will keep positive in all models and the negative effects will keep negative. Besides, it could be noticed that almost all of the coefficients are within (-1,1) range, there are no dominate genes that have stronger impact than the others.

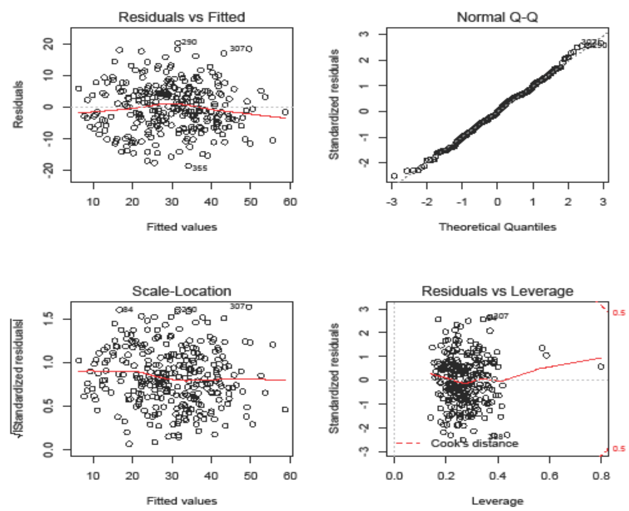
Prediction:
To test whether the models are good for prediction, ROC curves are plotted and the AUC values are summarized. For set A, the AUCs are (0.657, 0.689, 0.706) for three logistic regression models, for set B, the AUCs are (0.709, 0.725, 0.705); for combination set, the AUCs are (0.739, 0.757, 0.778). The results indicate that the AUCs are not very high, somehow the prediction won't be very accurate. When combining two sets of genes, the AUC or the prediction ability is improved. The ROC curves are shown in Figure 4.a, b, c, accordingly.

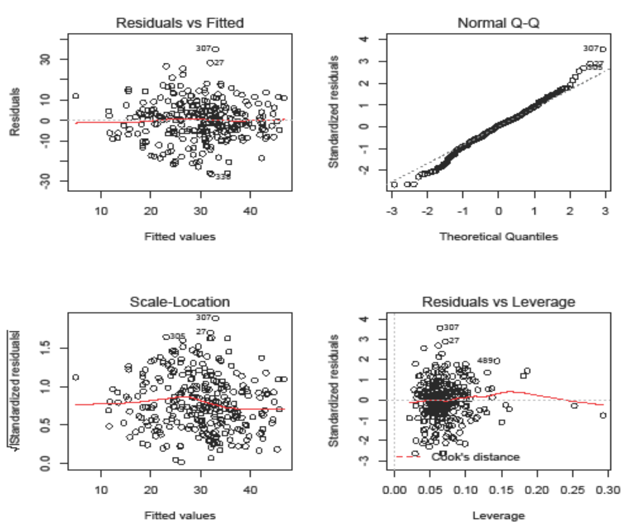
Figure 1.a, 1.b, 1.c, 1.d, 1.e, 1.f for data set A are shown in page 6.
Figure 2.a, 2.b, 2.c, 2.d, 2.e for data set B are shown in page 7.
Figure 3.a, 3.b, 3.c, 3.d, 3.e for combine data set are shown in page 8.
Figure 4.a, 4.b, 4.c are shown in page 9.
Selected genes as indexed are summarized in page 9-10. Outputs for GMC and sample code are shown in page 10-11. Results for logistic regression are summarized in page 11-15.
An excel file is attached to mapping the index of genes.

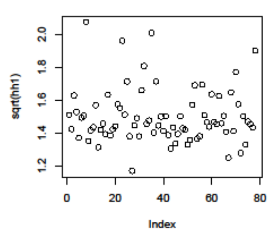

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

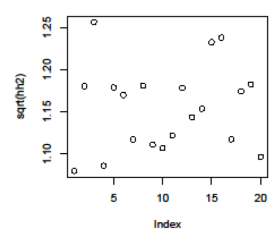

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

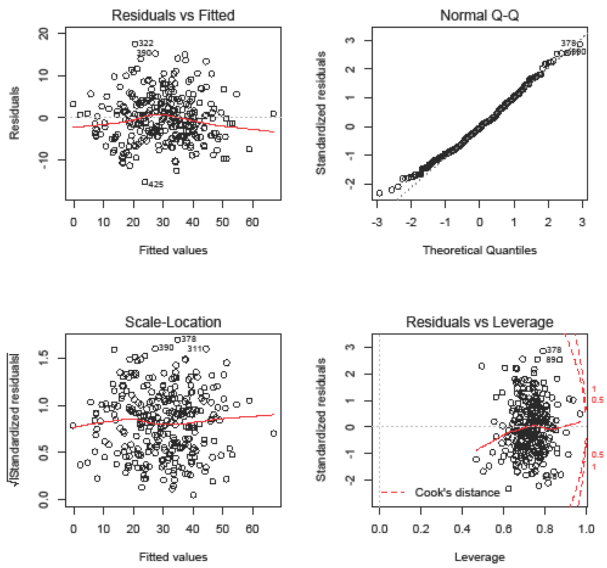

sqrt(hh)
Index

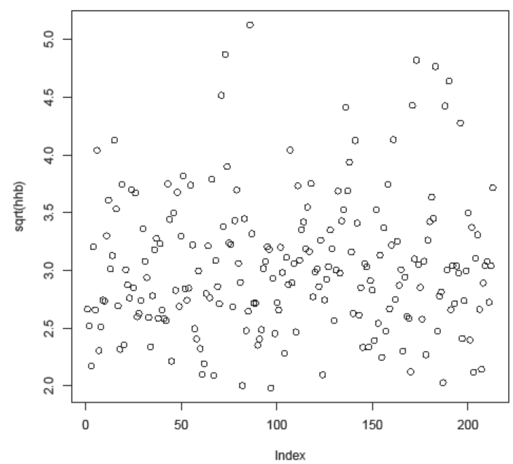

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

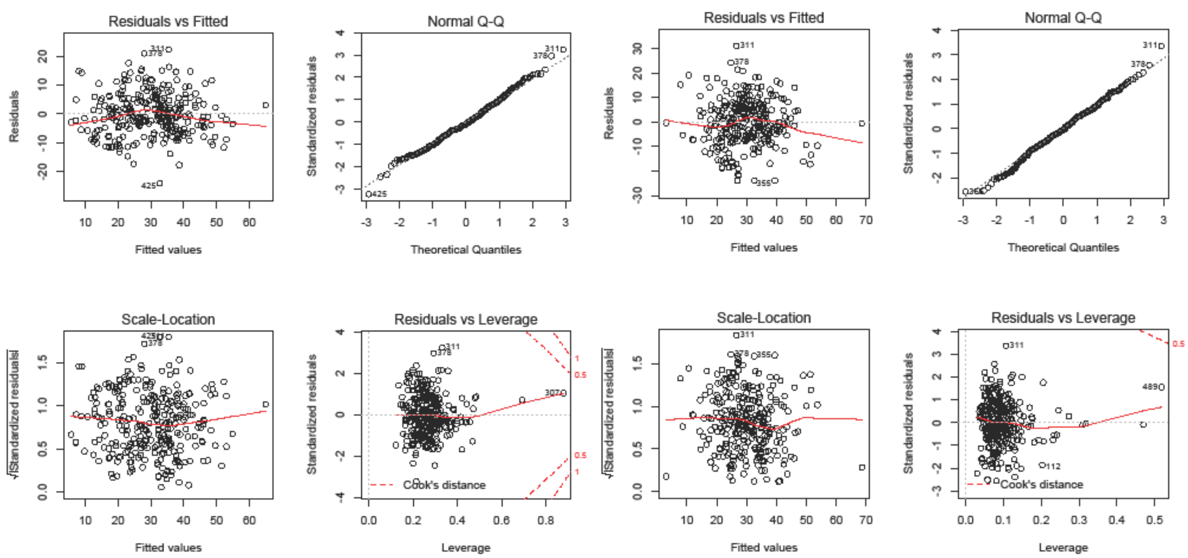

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

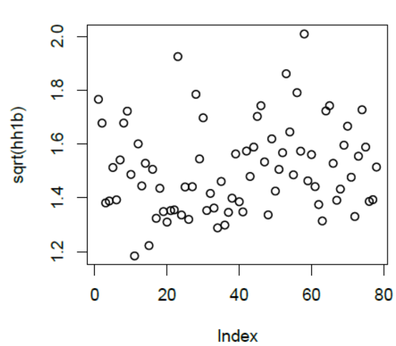

sqrt(hh1)
Index

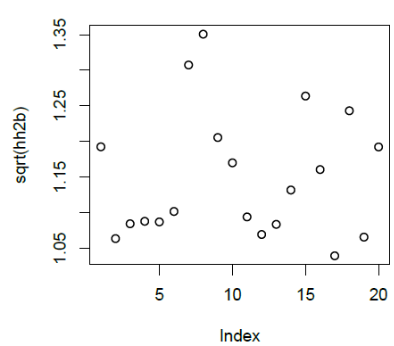

sqrt(hh2)
Index

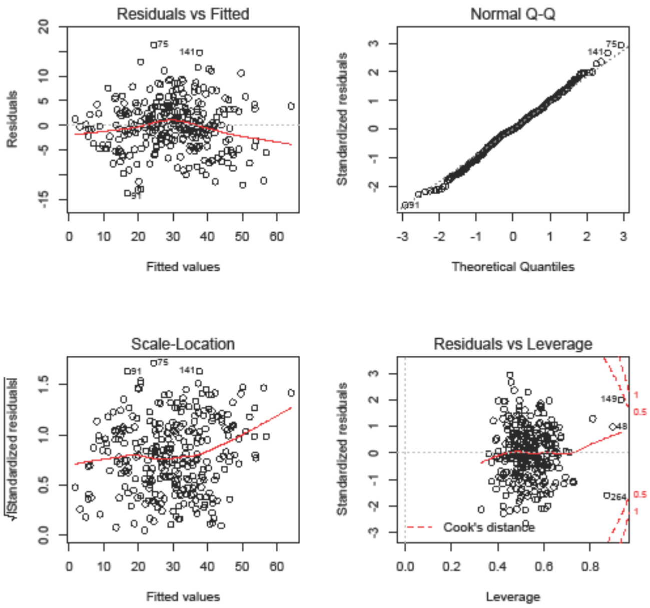

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
√|Standardized residuals|
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

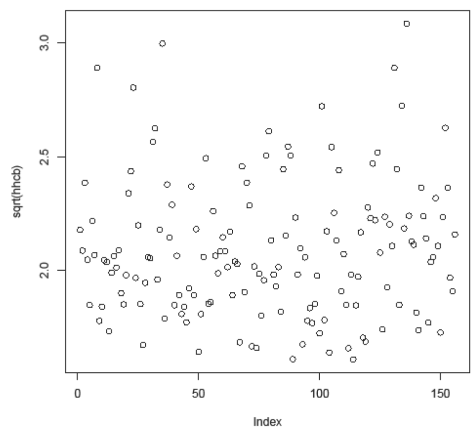

sqrt(hhb)
Index

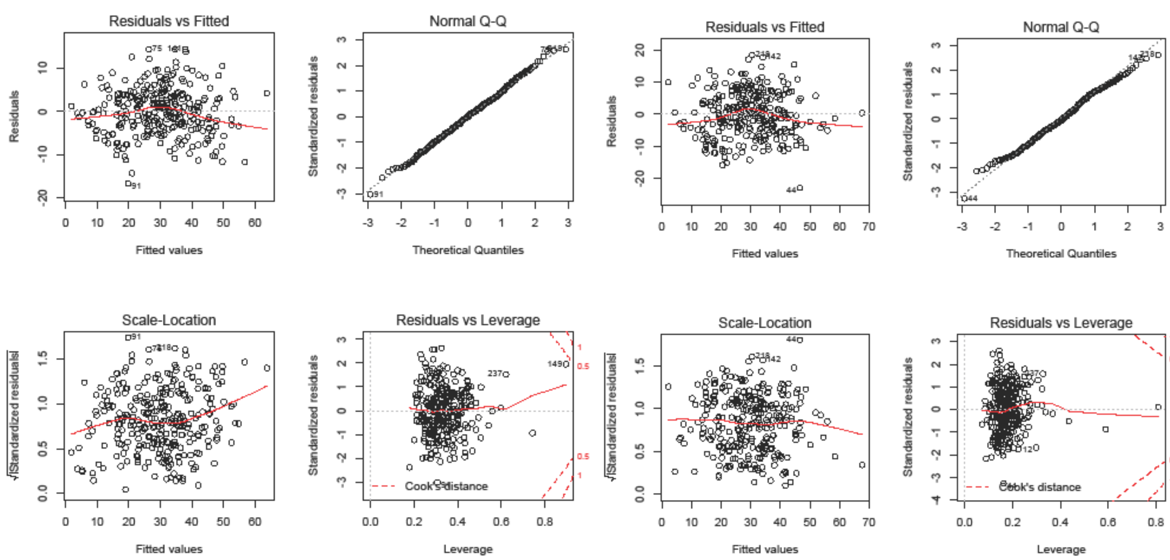

Residuals vs Fitted
Normal Q-Q
Scale-Location
Residuals vs Leverage
Residuals
Standardized residuals
√|Standardized residuals|
Fitted values
Theoretical Quantiles
Leverage
Cook's distance

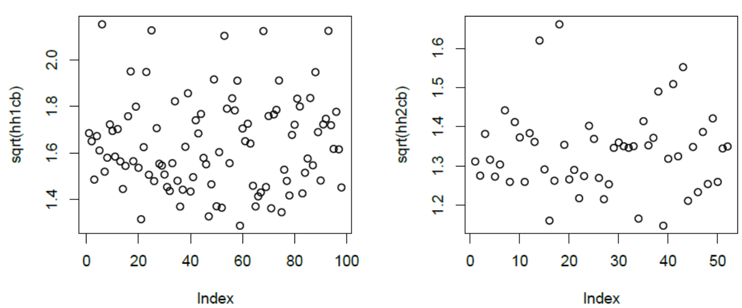

sqrt(hh1b)
Index

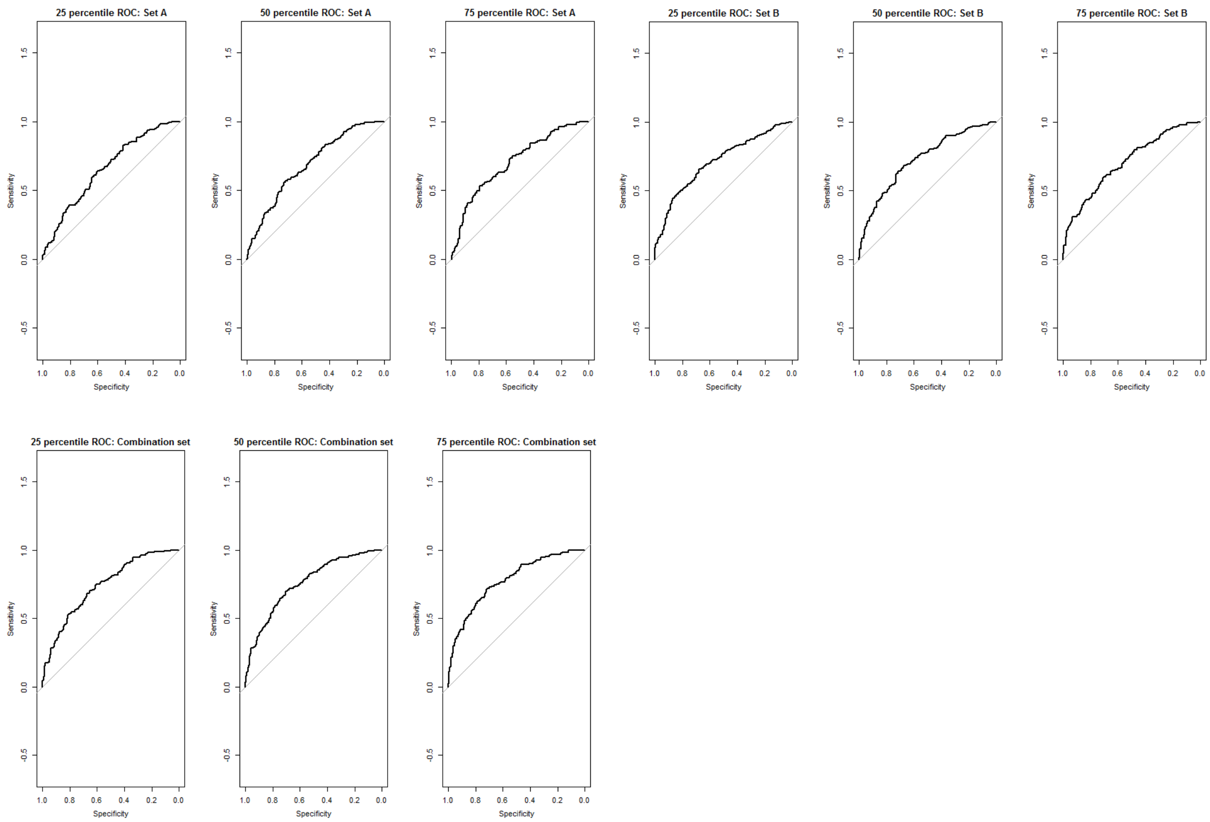

sqrt(hh2b)
Index


Residuals vs Fitted
Residuals
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage


sqrt(hhcb)
Index


Residuals vs Fitted
Residuals
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Residuals vs Fitted
Residuals
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage


sqrt(hh1cb)
Index
sqrt(hh2cb)
Index



Outputs for linear regressions:

```
> summary(model a1 (fortest1))
lm(formula = sqrt(y) ~ X190 + X114 + X126 + X34 + X99 + X77 +
    X106 + X30 + X189 + X204 + X131 + X53 + X207 + X81 + X197 +
    X90 + X78 + X154 + X41 + X134 + X80 + X56 + X151 + X37 +
    X66 + X171 + X49 + X67 + X155 + X165 + X60 + X127 + X112 +
    X55 + X143 + X50 + X62 + X135 + X209 + X109 + X150 + X36 +
    X17 + X145 + X76 + X213 + X68 + X23 + X46 + X122 + X167 +
    X22 + X111 + X27 + X44 + X157 + X1 + X210 + X200 + X83 +
    X153 + X174 + X13 + X184 + X69 + X123 + X125 + X3 + X132 +
    X142 + X211 + X193 + X9 + X47 + X7 + X88 + X116 + X75, data = XA)

> summary (model a2 (steptest2))
lm(formula = sqrt(y) ~ X17 + X22 + X30 + X34 + X37 + X55 + X76 +
    X77 + X80 + X90 + X99 + X106 + X114 + X126 + X127 + X151 +
    X190 + X197 + X204 + X213, data = XA)

> summary(model a3 (lasso.fit$beta))
213 x 1 sparse Matrix of class "dgCMatrix", with 14 entries
1   X30 X114  X56 X189  X97  X33 X126  X99 X199  X210
11  X34 X171  X190  X106

> summary(model b1 (fortest1b))
lm(formula = sqrt(y) ~ X196 + X102 + X148 + X168 + X117 + X2 +
    X161 + X4 + X55 + X180 + X207 + X122 + X149 + X146 + X67 +
    X126 + X45 + X40 + X18 + X157 + X143 + X42 + X107 + X155 +
    X48 + X34 + X31 + X113 + X147 + X154 + X197 + X195 + X208 +
    X124 + X8 + X140 + X84 + X17 + X68 + X5 + X27 + X127 + X163 +
    X87 + X134 + X37 + X77 + X75 + X152 + X150 + X131 + X193 +
```

```
    X183 + X78 + X119 + X51 + X194 + X71 + X153 + X109 + X205 +
    X130 + X61 + X121 + X118 + X94 + X120 + X110 + X167 + X123 +
    X175 + X38 + X98 + X43 + X16 + X191 + X101 + X70, data = XB)
```

```
> summary(model b2 (backtest2b))
lm(formula = sqrt(y) ~ X4 + X17 + X18 + X27 + X31 + X38 + X40 +
    X48 + X51 + X71 + X87 + X107 + X110 + X124 + X126 + X127 +
    X134 + X143 + X155 + X161 + X163 + X168 + X173 + X180 + X183 +
    X195 + X207, data = XB)
```

```
> summary(model b3 (lasso.fit.b$beta))
213 x 1 sparse Matrix of class "dgCMatrix", with 39 entries
1    X2 X4 X18 X27 X31 X33 X34 X40 X48 X49
11  X52 X55 X60 X65 X67 X71 X88 X89 X102 X107
21 X116 X117 X119 X126 X131 X143 X148 X149 X150 X154
31 X155 X161 X163 X168 X180 X196 X207 X212 X213
```

```
> summary(model c1 (steptest1cb))
lm(formula = sqrt(y) ~ X114 + X34 + X99 + X77 + X106 + X30 +
    X204 + X131 + X53 + X197 + X90 + X78 + X154 + X41 + X80 +
    X56 + X151 + X37 + X66 + X171 + X49 + X67 + X60 + X112 +
    X143 + X50 + X135 + X109 + X150 + X36 + X17 + X76 + X213 +
    X68 + X46 + X122 + X167 + X22 + X111 + X27 + X44 + X157 +
    X83 + X13 + X184 + X123 + X125 + X132 + X142 + X193 + X9 +
    X47 + X196b + X102b + X168b + X161b + X4b + X55b + X207b +
    X122b + X149b + X146b + X126b + X40b + X18b + X157b + X42b +
    X107b + X155b + X48b + X34b + X113b + X147b + X154b + X124b +
    X8b + X17b + X27b + X127b + X163b + X87b + X134b + X75b +
    X150b + X131b + X78b + X119b + X51b + X194b + X120b + X167b +
    X123b + X43b + X16b + X70b + X98b + X209 + X195b, data = XCb)
```

```
> summary(model c2 (backtest2cb))
lm(formula = sqrt(y) ~ X114 + X99 + X77 + X106 + X204 + X53 +
    X197 + X90 + X78 + X154 + X41 + X80 + X56 + X151 + X37 +
    X49 + X112 + X143 + X135 + X109 + X150 + X17 + X76 + X213 +
    X68 + X46 + X122 + X167 + X22 + X111 + X83 + X13 + X184 +
    X47 + X102b + X168b + X4b + X55b + X207b + X149b + X146b +
    X40b + X107b + X155b + X163b + X150b + X131b + X119b + X194b +
    X120b + X16b + X70b, data = XCb)
```

```
> summary(model c3 (lasso.fit.cb1$beta))
156 x 1 sparse Matrix of class "dgCMatrix", with 34 entries
A: X190 X114 X126 X34 X99 X77 X106 X189 X53 X207 X56 X171 X165 X213 X23 X210
B: X196 X102 X148 X168 X2 X161 X4 X55 X180 X207
X149 X67 X18 X155 X31
X163 X131 X119
```

Outputs for GMC:

Function	Set A	Set B	Combine Set
G1	0.346	0.340	0.571
G2	0.343	0.332	0.591
G3	0.324	0.338	0.581
G4	0.098	0.111	0.219
G5	0.328	0.345	0.337

R code for GMC:

```
optGMC <- function(beta) {
  data<-cbind(stdy, (g(x,beta)-mean(g(x,beta)))/sd(g(x,beta)))
  return(-GMC(data)[1])
}
GMC_g1_setA<-rep(0,200)
```

```
GMC_g1_setA_par<- matrix(data= NA, nrow=200, ncol=20)
for (i in 1:200)
{
    GMC_g1_setA_par[i,]<-sample(2:214, 20, replace=F)
    NEWXA<-cbind(y,XA[, GMC_g1_setA_par[i,]])
    test.lm<-lm(y~.-1,data=NEWXA)
    beta<-as.numeric(test.lm$coef)
    x<-as.matrix(NEWXA[,-1])
    res <- optim(beta, optGMC, method = "Nelder-Mead",control= list(maxit
=5000))
    GMC_g1_setA[i]<- -res$value
}
```

Outputs for Logistic regression:

```
> summary(logit.fit25<-glm(y25~., data=LogXA, family=binomial(link=logit)))
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 11.09519    6.94061   1.599  0.10991
x17         -0.44981    0.31377  -1.434  0.15169
x22          0.36326    0.37638   0.965  0.33447
x30         -0.10867    0.10277  -1.057  0.29035
x34         -0.23866    0.20306  -1.175  0.23987
x37          2.19882    0.83936   2.620  0.00880 **
x55          1.44179    0.71420   2.019  0.04351 *
x76          0.10610    0.09018   1.177  0.23939
x77          0.28129    0.14178   1.984  0.04726 *
x80          0.34942    0.17284   2.022  0.04321 *
x90         -0.06510    0.10265  -0.634  0.52597
x99         -0.93560    0.37561  -2.491  0.01274 *
x106        -0.01647    0.20689  -0.080  0.93656
x114        -0.30173    0.33834  -0.892  0.37250
x126        -0.21596    0.20989  -1.029  0.30352
x127        -0.92162    0.45123  -2.042  0.04111 *
x151        -0.07669    0.23067  -0.332  0.73955
x190        -1.59675    0.57093  -2.797  0.00516 **
x197         0.15332    0.17668   0.868  0.38550
x204        -1.05374    0.51735  -2.037  0.04167 *
x213         0.24363    0.15680   1.554  0.12023
```

```
> summary(logit.fit50<-glm(y50~., data=LogXA, family=binomial(link=logit)))
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 14.32412    6.64769   2.155 0.031181 *
x17         -0.08800    0.31982  -0.275 0.783198
x22          0.83777    0.36510   2.295 0.021754 *
x30         -0.21485    0.09999  -2.149 0.031662 *
x34         -0.25586    0.20250  -1.264 0.206404
x37          1.76258    0.80851   2.180 0.029255 *
x55          1.30121    0.68256   1.906 0.056603 .
x76          0.12477    0.08722   1.430 0.152590
x77          0.03941    0.13224   0.298 0.765699
x80          0.44433    0.16306   2.725 0.006432 **
x90         -0.26736    0.10187  -2.624 0.008679 **
x99         -1.45612    0.38798  -3.753 0.000175 ***
x106        -0.24288    0.19896  -1.221 0.222191
x114        -0.57917    0.33155  -1.747 0.080668 .
x126        -0.32647    0.20594  -1.585 0.112906
x127        -0.84859    0.43039  -1.972 0.048647 *
x151        -0.14894    0.21927  -0.679 0.496991
x190        -1.20676    0.58019  -2.080 0.037533 *
x197         0.24554    0.17065   1.439 0.150176
x204        -0.86051    0.50140  -1.716 0.086124 .
x213         0.18948    0.14819   1.279 0.201026
```

```
> summary(logit.fit75<-glm(y75~., data=LogXA, family=binomial(link=logit)))
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 23.56979    7.72907   3.049  0.00229 **
x17         -0.08749    0.37954  -0.231  0.81769
x22          1.05557    0.42178   2.503  0.01233 *
x30         -0.25823    0.11598  -2.227  0.02598 *
```

```
x34          -0.27316    0.23966  -1.140  0.25437
x37           0.17217    0.93625   0.184  0.85410
x55           1.16610    0.78398   1.487  0.13690
x76           0.24466    0.09976   2.453  0.01418 *
x77          -0.02081    0.15370  -0.135  0.89228
x80           0.34908    0.18250   1.913  0.05577 .
x90          -0.34876    0.11968  -2.914  0.00357 **
x99          -1.22101    0.45350  -2.692  0.00709 **
x106         -0.30398    0.22710  -1.339  0.18073
x114         -0.69950    0.39755  -1.759  0.07849 .
x126         -0.53065    0.24173  -2.195  0.02815 *
x127         -1.05243    0.48974  -2.149  0.03164 *
x151         -0.08693    0.25032  -0.347  0.72838
x190         -0.79130    0.68067  -1.163  0.24502
x197          0.19837    0.19659   1.009  0.31295
x204         -1.39381    0.58510  -2.382  0.01721 *
x213          0.21325    0.16945   1.258  0.20821
```

```
> summary(logit.fit25.b<-glm(y25~., data=LogXB, family=binomial(link=logit)))
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 12.083544   5.571383   2.169  0.03009 *
x4           0.413695   0.174981   2.364  0.01807 *
x17          0.154100   0.183734   0.839  0.40163
x18         -0.888785   0.706938  -1.257  0.20867
x27          0.618965   0.279484   2.215  0.02678 *
x31          0.617271   0.212135   2.910  0.00362 **
x38         -0.014399   0.183796  -0.078  0.93756
x40         -0.141043   0.156676  -0.900  0.36800
x48         -0.269163   0.217668  -1.237  0.21624
x51         -0.484565   0.274739  -1.764  0.07778 .
x71         -0.104560   0.159486  -0.656  0.51208
x87          0.005533   0.139866   0.040  0.96845
x107         0.109978   0.130367   0.844  0.39889
x110        -0.613121   0.186491  -3.288  0.00101 **
x124         0.708057   0.268268   2.639  0.00831 **
x126         0.138418   0.247040   0.560  0.57527
x127         0.168950   0.225499   0.749  0.45372
x134        -0.103013   0.109148  -0.944  0.34528
x143        -0.327935   0.717143  -0.457  0.64747
x155        -0.050808   0.126474  -0.402  0.68789
x161        -0.532075   0.166445  -3.197  0.00139 **
x163        -0.397393   0.165819  -2.397  0.01655 *
x168        -0.342434   0.455338  -0.752  0.45203
x173         0.365376   0.200145   1.826  0.06792 .
x180        -0.570247   0.182744  -3.120  0.00181 **
x183         0.256413   0.209853   1.222  0.22176
x195        -0.167576   0.147286  -1.138  0.25522
x207        -0.933034   0.414387  -2.252  0.02435 *
```

```
> summary(logit.fit50.b<-glm(y50~., data=LogXB, family=binomial(link=logit)))
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 15.36980    5.38164   2.856 0.004291 **
x4           0.29631    0.13030   2.274 0.022963 *
x17          0.32589    0.17831   1.828 0.067593 .
x18         -2.05768    0.73292  -2.808 0.004993 **
x27          0.40463    0.27675   1.462 0.143717
x31          0.68732    0.20506   3.352 0.000803 ***
x38          0.15098    0.17577   0.859 0.390337
x40          0.09068    0.15249   0.595 0.552039
x48         -0.29543    0.20822  -1.419 0.155937
x51         -0.66187    0.28063  -2.359 0.018348 *
x71         -0.35518    0.15254  -2.328 0.019891 *
x87          0.01629    0.13403   0.122 0.903258
x107        -0.20938    0.12635  -1.657 0.097485 .
x110        -0.35771    0.17920  -1.996 0.045913 *
x124         0.78561    0.22502   3.491 0.000481 ***
x126         0.17104    0.24423   0.700 0.483743
x127         0.22153    0.21574   1.027 0.304480
x134        -0.04845    0.10432  -0.464 0.642348
x143        -0.48029    0.67540  -0.711 0.477008
x155         0.23414    0.12121   1.932 0.053396 .
```

```
x161        -0.65123    0.16591  -3.925 8.67e-05 ***
x163        -0.37271    0.17104  -2.179 0.029325 *
x168        -0.65451    0.43296  -1.512 0.130606
x173         0.08601    0.18999   0.453 0.650748
x180        -0.57281    0.17902  -3.200 0.001376 **
x183         0.48605    0.20421   2.380 0.017306 *
x195        -0.12285    0.14673  -0.837 0.402446
x207        -0.66818    0.42373  -1.577 0.114815
```

```
> summary(logit.fit50.b<-glm(y75~., data=LogXB, family=binomial(link=logit)))
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 10.98750    5.97687   1.838  0.06601 .
x4           0.31071    0.12227   2.541  0.01105 *
x17          0.11279    0.19657   0.574  0.56613
x18         -1.20690    0.81923  -1.473  0.14069
x27          0.44830    0.31254   1.434  0.15147
x31          0.61190    0.23030   2.657  0.00789 **
x38          0.25395    0.19864   1.278  0.20109
x40          0.02067    0.17854   0.116  0.90783
x48         -0.28820    0.23373  -1.233  0.21755
x51         -0.68848    0.31777  -2.167  0.03026 *
x71         -0.49134    0.17368  -2.829  0.00467 **
x87          0.09126    0.15050   0.606  0.54429
x107        -0.40761    0.14345  -2.841  0.00449 **
x110        -0.21495    0.20386  -1.054  0.29170
x124         0.61850    0.21401   2.890  0.00385 **
x126         0.20378    0.27028   0.754  0.45088
x127         0.18733    0.24239   0.773  0.43962
x134        -0.05599    0.11750  -0.476  0.63372
x143         0.13852    0.76045   0.182  0.85546
x155         0.03356    0.13681   0.245  0.80621
x161        -0.31075    0.18164  -1.711  0.08712 .
x163        -0.56755    0.21251  -2.671  0.00757 **
x168        -1.21806    0.50787  -2.398  0.01647 *
x173         0.44149    0.21596   2.044  0.04092 *
x180        -0.36742    0.20083  -1.829  0.06733 .
x183         0.40065    0.22917   1.748  0.08042 .
x195        -0.09060    0.16523  -0.548  0.58346
x207        -0.59876    0.48597  -1.232  0.21791
```

```
> summary(logit.fit25.cb<-glm(y25~., data=LogXCb))
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -7.7182248  9.3855541  -0.822 0.410877
x114        -0.1472588  0.3926630  -0.375 0.707641
x99         -1.1385902  0.4219029  -2.699 0.006961 **
x77          0.4871557  0.1781640   2.734 0.006251 **
x106        -0.1392836  0.2390771  -0.583 0.560170
x204        -1.4906756  0.5928056  -2.515 0.011916 *
x53          0.2089735  0.2603327   0.803 0.422138
x197         0.2844337  0.2151252   1.322 0.186109
x90         -0.0397225  0.1200194  -0.331 0.740669
x78          1.0690501  0.6934303   1.542 0.123151
x154        -0.3405017  0.7234141  -0.471 0.637864
x41          1.1060358  0.7465822   1.481 0.138483
x80          0.5904570  0.2198858   2.685 0.007247 **
x56         -0.9093293  0.2669300  -3.407 0.000658 ***
x151        -0.0411083  0.3019066  -0.136 0.891693
x37          2.3590104  0.9687734   2.435 0.014890 *
x49         -0.5017117  0.2948159  -1.702 0.088797 .
x112         0.5631549  0.2252460   2.500 0.012413 *
x143        -0.3849255  0.3083947  -1.248 0.211973
x135         0.1768287  0.3047011   0.580 0.561689
x109         0.0734110  0.0902050   0.814 0.415746
x150        -0.1336732  0.2612946  -0.512 0.608945
x17         -0.4944716  0.3701826  -1.336 0.181631
x76          0.1249525  0.1046627   1.194 0.232533
x213         0.2454417  0.1960145   1.252 0.210511
x68          0.4536447  0.1992614   2.277 0.022808 *
x46          0.3024123  0.2325107   1.301 0.193382
x122         0.0455726  0.8203608   0.056 0.955699
x167         0.1637085  0.4354850   0.376 0.706975
```

```
x22           0.6412852  0.4333486   1.480 0.138917  
x111         -0.2092243  0.1121337  -1.866 0.062063 .
x83          -0.5187195  0.2203148  -2.354 0.018550 *
x13           0.0344168  0.1744969   0.197 0.843644  
x184          0.2730038  0.1459327   1.871 0.061380 .
x47           1.2184326  0.6635737   1.836 0.066333 .
x102b         0.7255022  0.5569038   1.303 0.192663  
x168b         0.1722948  0.5183099   0.332 0.739575  
x4b           0.2889276  0.1924825   1.501 0.133340  
x55b          0.2345263  0.2108982   1.112 0.266123  
x207b        -0.8591172  0.4592828  -1.871 0.061406 .
x149b        -0.1785369  0.2203905  -0.810 0.417886  
x146b        -0.4282535  0.1973124  -2.170 0.029974 *
x40b         -0.0268770  0.1698806  -0.158 0.874291  
x107b         0.1778607  0.1494650   1.190 0.234053  
x155b        -0.0008356  0.1329139  -0.006 0.994984  
x163b        -0.4559230  0.1939669  -2.351 0.018747 *
x150b        -0.2638964  0.1360115  -1.940 0.052349 .
x131b        -0.5631086  0.2366598  -2.379 0.017341 *
x119b         0.1105001  0.1370121   0.806 0.419955  
x194b        -0.4234301  0.2665794  -1.588 0.112200  
x120b         0.1186390  0.1722634   0.689 0.491008  
x16b         -0.2001892  0.5160691  -0.388 0.698081  
x70b         -0.1876789  0.4731771  -0.397 0.691636  
```

```
> summary(logit.fit50.cb<-glm(y50~., data=LogXCb, family=binomial(link=logit)))
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  3.96728    9.08971   0.436 0.662504    
x114        -0.40311    0.38376  -1.050 0.293531    
x99         -1.66113    0.44030  -3.773 0.000161 ***
x77          0.21343    0.15890   1.343 0.179213    
x106        -0.38884    0.22881  -1.699 0.089248 .  
x204        -1.33418    0.56972  -2.342 0.019190 *  
x53          0.23079    0.25288   0.913 0.361426    
x197         0.32615    0.20643   1.580 0.114110    
x90         -0.26716    0.12028  -2.221 0.026346 *  
x78          1.06171    0.66635   1.593 0.111087    
x154        -0.75480    0.68474  -1.102 0.270326    
x41          0.13439    0.70373   0.191 0.848551    
x80          0.69594    0.20756   3.353 0.000800 ***
x56         -0.86265    0.26812  -3.217 0.001294 ** 
x151         0.17263    0.28535   0.605 0.545194    
x37          2.15636    0.93096   2.316 0.020542 *  
x49         -0.83520    0.32268  -2.588 0.009646 ** 
x112         0.15728    0.21592   0.728 0.466358    
x143        -0.84009    0.29518  -2.846 0.004427 ** 
x135         0.09458    0.29751   0.318 0.750569    
x109         0.09143    0.08792   1.040 0.298344    
x150        -0.24643    0.25082  -0.982 0.325869    
x17          0.31532    0.37831   0.833 0.404566    
x76          0.13337    0.10151   1.314 0.188887    
x213         0.07190    0.18252   0.394 0.693627    
x68          0.12274    0.19269   0.637 0.524151    
x46          0.42581    0.22014   1.934 0.053085 .  
x122         0.37314    0.75367   0.495 0.620531    
x167         0.70130    0.42978   1.632 0.102733    
x22          0.81354    0.41181   1.976 0.048208 *  
x111        -0.22303    0.10683  -2.088 0.036832 *  
x83         -0.32381    0.21685  -1.493 0.135369    
x13          0.03014    0.16465   0.183 0.854744    
x184         0.26295    0.13831   1.901 0.057289 .  
x47          0.82957    0.61578   1.347 0.177925    
x102b        0.47342    0.39466   1.200 0.230309    
x168b       -0.08889    0.49472  -0.180 0.857408    
x4b          0.12935    0.12944   0.999 0.317629    
x55b         0.07060    0.21201   0.333 0.739144    
x207b       -0.26850    0.45046  -0.596 0.551138    
x149b       -0.38890    0.21280  -1.828 0.067619 .  
x146b       -0.05677    0.18859  -0.301 0.763383    
x40b         0.09467    0.16840   0.562 0.573995    
x107b       -0.03391    0.14396  -0.236 0.813771    
```

```
x155b         0.20177    0.12861   1.569 0.116691
x163b        -0.24208    0.19162  -1.263 0.206468
x150b        -0.21101    0.12923  -1.633 0.102511
x131b        -0.64451    0.26216  -2.458 0.013954 *
x119b         0.07388    0.12753   0.579 0.562402
x194b         0.13368    0.24844   0.538 0.590509
x120b        -0.02640    0.16606  -0.159 0.873688
x16b         -0.56286    0.49436  -1.139 0.254885
x70b          0.21326    0.45614   0.468 0.640127
```

```
> summary(logit.fit75.cb<-glm(y75~., data=LogXCb, family=binomial(link=logit)))
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 21.498296 10.899143   1.972  0.04856 *
x114        -0.668319  0.472051  -1.416  0.15684
x99         -1.042161  0.521331  -1.999  0.04560 *
x77          0.238517  0.184896   1.290  0.19705
x106        -0.462612  0.265739  -1.741  0.08171 .
x204        -1.920520  0.662302  -2.900  0.00373 **
x53          0.472626  0.303300   1.558  0.11917
x197         0.287523  0.242334   1.186  0.23544
x90         -0.338415  0.147298  -2.297  0.02159 *
x78          0.264231  0.786212   0.336  0.73681
x154        -0.441054  0.805049  -0.548  0.58379
x41         -0.064487  0.815531  -0.079  0.93697
x80          0.520593  0.233666   2.228  0.02588 *
x56         -0.829006  0.322178  -2.573  0.01008 *
x151         0.148993  0.322768   0.462  0.64436
x37          0.237150  1.078353   0.220  0.82593
x49         -0.665447  0.372072  -1.788  0.07370 .
x112         0.146464  0.254075   0.576  0.56430
x143        -0.762573  0.342059  -2.229  0.02579 *
x135         0.464839  0.348507   1.334  0.18227
x109         0.050947  0.102498   0.497  0.61916
x150        -0.229591  0.294935  -0.778  0.43631
x17          0.271570  0.451380   0.602  0.54741
x76          0.290467  0.119268   2.435  0.01487 *
x213         0.038501  0.212566   0.181  0.85627
x68          0.253647  0.231489   1.096  0.27320
x46         -0.035290  0.259139  -0.136  0.89168
x122         0.557256  0.836225   0.666  0.50516
x167         0.589913  0.508938   1.159  0.24641
x22          0.866700  0.479430   1.808  0.07064 .
x111        -0.265885  0.126868  -2.096  0.03610 *
x83         -0.463894  0.255242  -1.817  0.06915 .
x13         -0.073034  0.190302  -0.384  0.70114
x184         0.213961  0.158082   1.353  0.17590
x47          0.823159  0.699869   1.176  0.23953
x102b        0.725780  0.366829   1.979  0.04787 *
x168b       -0.927983  0.612846  -1.514  0.12997
x4b          0.100091  0.132680   0.754  0.45062
x55b        -0.041580  0.246942  -0.168  0.86629
x207b       -0.365045  0.536695  -0.680  0.49640
x149b       -0.484608  0.249208  -1.945  0.05182 .
x146b        0.242718  0.221083   1.098  0.27227
x40b         0.112522  0.202578   0.555  0.57859
x107b       -0.406611  0.172847  -2.352  0.01865 *
x155b       -0.010885  0.151221  -0.072  0.94262
x163b       -0.560937  0.247090  -2.270  0.02320 *
x150b       -0.365597  0.155773  -2.347  0.01893 *
x131b       -0.729776  0.337275  -2.164  0.03048 *
x119b        0.186285  0.141927   1.313  0.18934
x194b       -0.104422  0.296493  -0.352  0.72469
x120b        0.294225  0.198450   1.483  0.13818
x16b         0.096834  0.580414   0.167  0.86750
x70b        -0.006546  0.533672  -0.012  0.99021
```