Assignment for SDC Biostatistics

Written by:

Yuanyi ZHANG

Introduction

This project will check tumors' response to certain drugs has great applications in practice, including screening for certain genes that are the most predictive ones of drug response. Typically, this analysis will be based on the drug response and gene expression data. Here we take the drug (Doxorubicin_IC_50) to analyze the response of tumor cells and screen some predictive genes.

Data check and Exploratory analysis

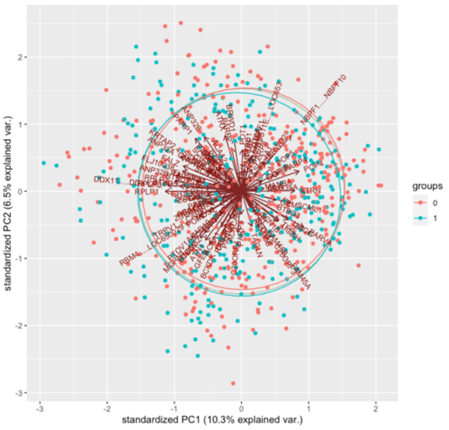
These are the given data descriptions: In data1, the rows represent tumors, while the columns represent the genes (or the explanatory variables). The entries in the matrix represent gene expression level in corresponding tumors. There are 641 tumors and 100 genes in total. In data2, the rows are also the tumors and the first column is the response of the tumors to the drug. The 641 tumors have been divided into two categories according to their response to the drug (Doxorubicin_IC_50). The second column is the class label (0 represents class 1, 1 represents class 2).

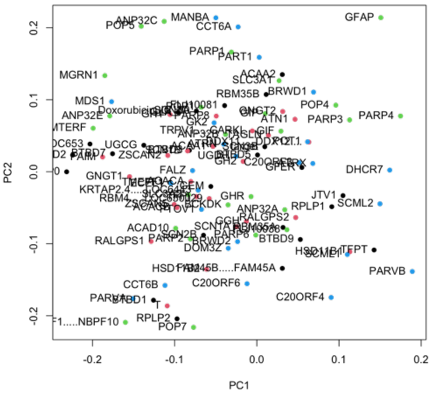
Data preprocessing

In order to viewing the data distribution, first we generated data1 and data2 and obtained a combined dataset(*data*) with all information. By calculating the mean and sd of each column, we can find that the dataset is already been normalized.

Exploratory analysis

We also want to check the relationship of the data, so we carry out a PCA analysis of the data table. Plot the loadings to verify whether the genes can be separated and use colors for better distinction of the target.





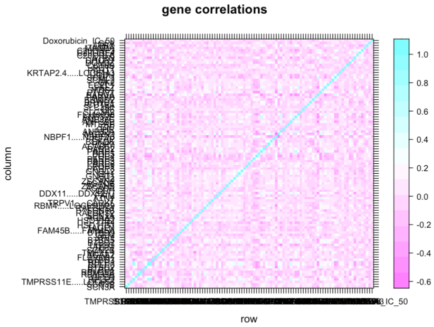
As we can see here, the genes can be separate quite well, but it’s hard to distinguish different target from PC1 and PC2.

Genes selection

This section we will try different methods to find the most explanatory genes to Doxorubicin_IC_50 response with some detailed description and figures.

Pearson's correlations

First, we calculate Pearson's correlations (*cor* function) between all gene and plot all of them at once with the *levelplot* function (library lattice). Then, we select top 10 genes which has the highest correlation rate with column 102(**Doxorubicin_IC_50**). This method is not so mathematically rigorous, but I think it's a reasonable way to get the general picture of the relationship between these genes and target.



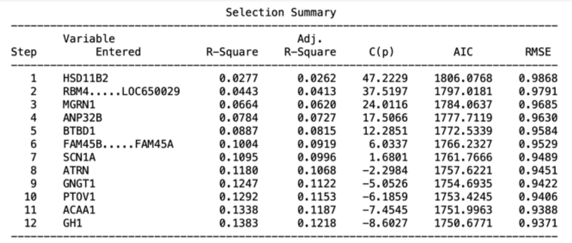
Linear regression

By using the *lm* function, we build a linear model for whole dataset and get the coefficient messages from the summary.

Forward selection procedure with P-value

This method builds regression model from a set of candidate predictor variables by entering and removing predictors based on p values, in a stepwise manner until there is no variable left to enter or remove any more.

Here, we use the library "olsrr" with *ols_step_forward_p()*.



```
                              Selection Summary
--------------------------------------------------------------------------------------
        Variable                           Adj.
Step        Entered          R-Square    R-Square      C(p)         AIC         RMSE
--------------------------------------------------------------------------------------
   1    HSD11B2                0.0277      0.0262    47.2229    1806.0768    0.9868
   2    RBM4.....LOC650029     0.0443      0.0413    37.5197    1797.0181    0.9791
   3    MGRN1                  0.0664      0.0620    24.0116    1784.0637    0.9685
   4    ANP32B                 0.0784      0.0727    17.5066    1777.7119    0.9630
   5    BTBD1                  0.0887      0.0815    12.2851    1772.5339    0.9584
   6    FAM45B.....FAM45A      0.1004      0.0919     6.0337    1766.2327    0.9529
   7    SCN1A                  0.1095      0.0996     1.6801    1761.7666    0.9489
   8    ATRN                   0.1180      0.1068    -2.2984    1757.6221    0.9451
   9    GNGT1                  0.1247      0.1122    -5.0526    1754.6935    0.9422
  10    PTOV1                  0.1292      0.1153    -6.1859    1753.4245    0.9406
  11    ACAA1                  0.1338      0.1187    -7.4545    1751.9963    0.9388
  12    GH1                    0.1383      0.1218    -8.6027    1750.6771    0.9371
--------------------------------------------------------------------------------------
```

Forward selection procedure by AIC

This method builds regression model from a set of candidate predictor variables by removing predictors based on Akaike Information Criteria, in a stepwise manner until there is no variable left to remove any more.

Here, we use the library "olsrr" with *ols_step_forward_aic()*.In this method, the outcome of top 10 genes is as same as the last one.

Lasso regression

The Lasso regression model is a commonly used linear regression model. When the model dimension is high, the Lasso algorithm selects the variables of the model by solving the sparse solution.

library(lars)

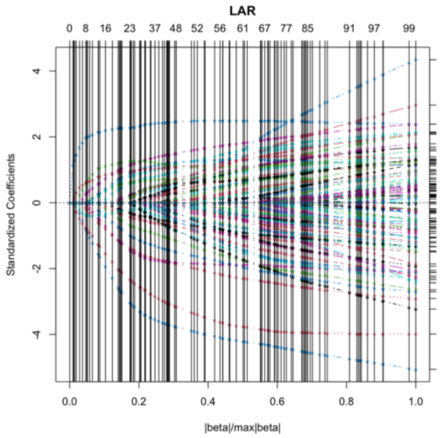
The Lars algorithm provides a fast way to solve the model.

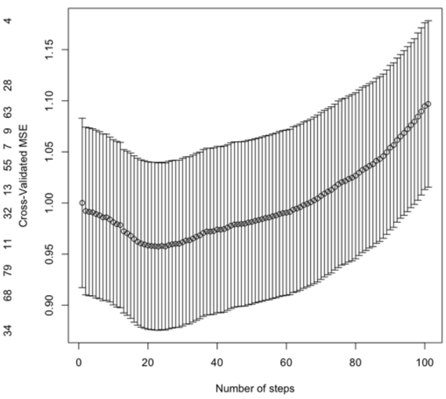
Fit the Lasso Regression Model:

lar<-lars(x,y,type="lar")

The return parameter is a list, which contains return values such as the regression coefficient beta and lambda obtained for each iteration.

We can use *plot()* to draw the image of its solution path for the return parameters respectively.





Choose an Optimal Value for Step:

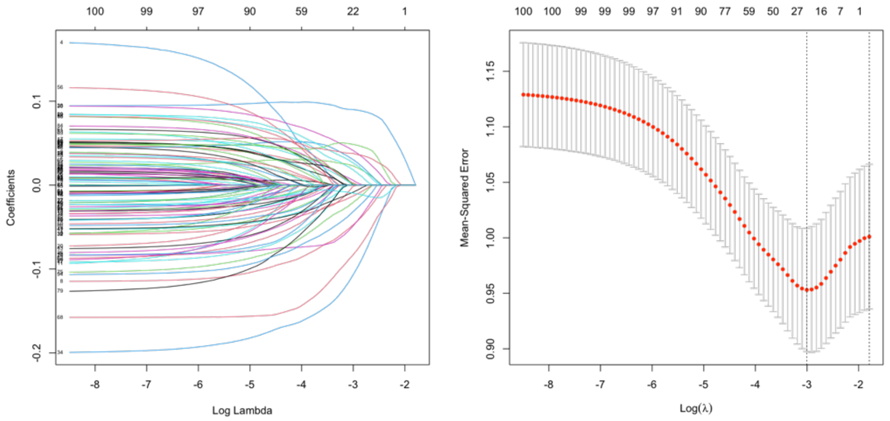
In the previous step we can see that lars has given all the solutions on its solution path at once, we need to determine which of them is the one we really want to use. In the lasso model, the constraint term is controlled by the parameter lambda, when given lambda, the model can be determined. A good regression model needs to be given a suitable lambda, but the range of lamda is often large. Note that the number of solutions on the solution path given by the lars algorithm is limited, and different solutions, i.e., different betas, correspond to different lambdas. As can be seen from the diagram of the solution path, we can select the step of the algorithm Number or select beta saturation |beta|/max|beta| (where || represents a norm, saturation also represents the learned sparsity) to select the parameters of the model.

Here, we use mode = “step” and the cross-validated mean square error MSE analysis results are as follows(step=24):

library(glmnet)

Fit the Lasso Regression Model:

We use this package to plot the changing trend of variable coefficients lambda increasing in lasso and this is the result:

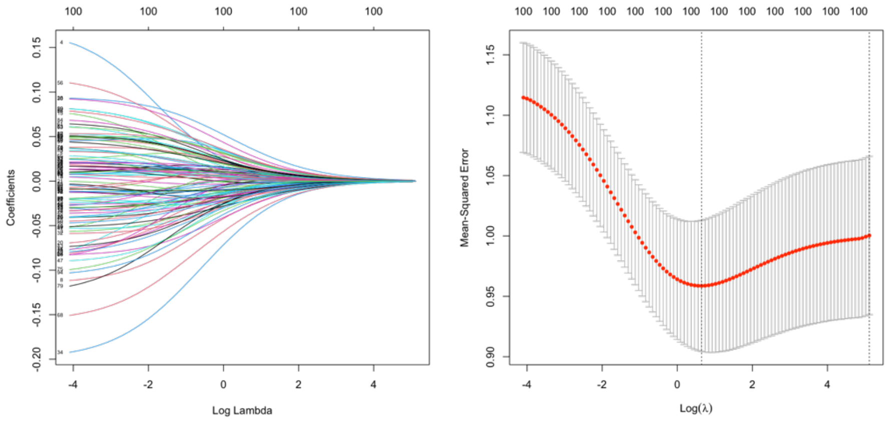


Choose an Optimal Value for Lambda:

“glmnet” has the function *cv.glmnet()* that automatically performs k-fold cross validation using k = 10 folds. So, after using *cv.glmnet()* to fit the model, we extract the minimum lambda and 1se one. The minimum is 0.0496127148062766 and 1se is 0.166282106464989. Then apply them in new models and check the coef again. It’s easy to find the min one fit better, so we extract the top 10 genes from this solution.

Ridge regression

Similarly, we use “glmnet” to fit the ridge regression and find optimal lambda value that minimizes test MSE. Here, the best lambda is 1.91.



Later, we apply the lambda to find the gene’s coefficients of model.

Selected genes conclusion

These are the gene we select through different methods (the blue ones are the common genes):

correlations	**Linear Regression**	**Lasso Regression**
'BRWD2'	'HSD11B2'	'RBM4.....LOC650029'
'ANP32B'	'RBM4.....LOC650029'	'ANP32B'
'PARP8'	'MGRN1'	'FAM45B.....FAM45A'

'NBPF1.....NBPF10'	'ANP32B'	'BTBD1'
'BTBD1'	'BTBD1'	'ATRN'
'RBM4.....LOC650029'	'FAM45B.....FAM45A'	'GNGT1'
'FAM45B.....FAM45A'	'SCN1A'	'PTOV1'
'CCT6A'	'ATRN'	'SCN1A'
'ATRN'	'GNGT1'	'MGRN1'
'POP5'	'PTOV1'	'HSD11B2'

SVM

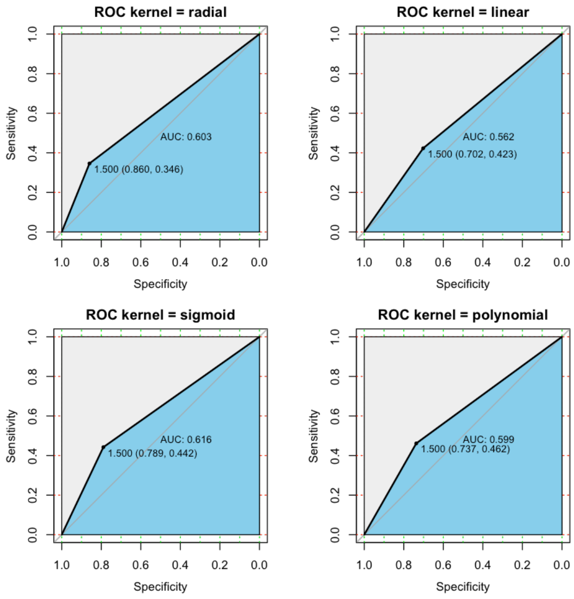
At SVM we can apply different kernels or parameters, and these the kernels used in training and predicting. We might consider changing some of the following parameters, depending on the kernel type.

Kernels	**Formula**	**Parameters**
Linear	$u'*v$	no need to set parameters
Polynomial	$(gamma*u'*v+coef0)$ ^$degree$	degree, gamma, coef0
Radial basis	$exp(-gamma*\vert u-v\vert 2)$	gamma
Sigmoid	$tanh(gamma*u'*v+coef0)$	gamma, coef0

coef0: The default value of coef0 is 0, which is the constant term of the kernel function of poly and sigmoid. It is used to solve the problem of measuring the difference between different values when the <x, y> values in the poly function are approaching and there is no obvious difference. The default value is 0. It reflects the influence of high-order polynomials on the model relative to low-order polynomials. If overfitting occurs, you can reduce coef0; if underfitting occurs, you can try to increase coef0.

gamma and C parameters: For linear kernels, we only need to optimize the c parameter. However, if the RBF kernel function is to be used, both the c parameter and the gamma parameter need to be optimized simultaneously. If gamma is large, the effect of c is negligible. If gamma is small, c affects the model as it does a linear model. Typical values of c and gamma are as follows. However, depending on the application, there may be specific optimum values: $0.0001 < gamma < 10$, $0.1 < c < 100$.

In order to train a classification model, we randomly divide the 641 tumor samples into 2 parts (with ratio is 5:1) as training data and testing data respectively. Then, we try to apply the 4 methods above with different parameters. These are the result with the best performance parameters:

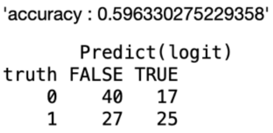


We also use a function *tune.svm()* which is said could provide the best parameters of the data, but the model performance will not become better after changing the "best" ones. It might because the model is approximately useless, as we can see the predict probs is just over 0.6 or even worse.

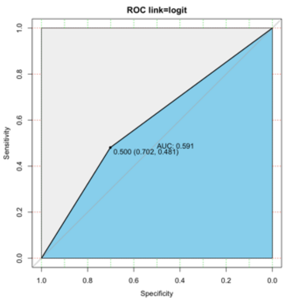
Logistic Regression

The Logistic Regression is a regression model in which the response variable (dependent variable) has categorical values such as True/False or 0/1. It actually measures the probability of a binary response as the value of response variable based on the mathematical equation relating it with the predictor variables.

The algorithm does not converge, you can increase the number of iterations through control=list(maxit=100). "glm" is default binomial model the default predictions are of log-odds (probabilities on logit scale) and type = "response" gives the predicted probabilities. if the prediction > 0, it is 1 class(0), if < 0 , it is in the 2 class(1). With the link='logit', we get the following results:

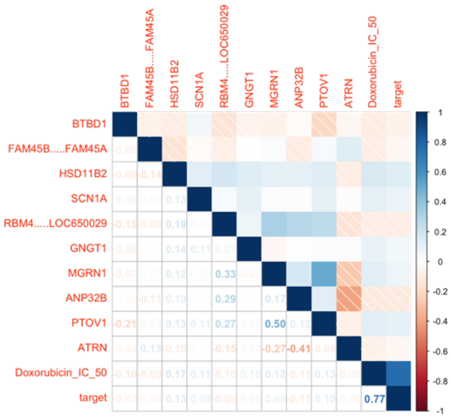


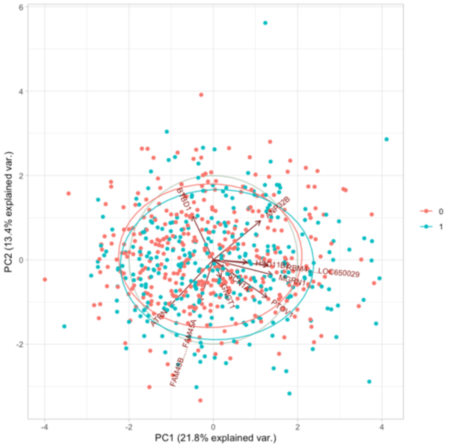
```
'accuracy : 0.596330275229358'

       Predict(logit)
truth FALSE TRUE
    0    40   17
    1    27   25
```



TOP 10 genes

Now, we focus on the 10 genes we select and use them to predict the drug response. First step is extracting these columns and generate a new genedata. Then, we test correlation between 10 genes, IC50 and target class (library corrplot) and review PCA analysis.

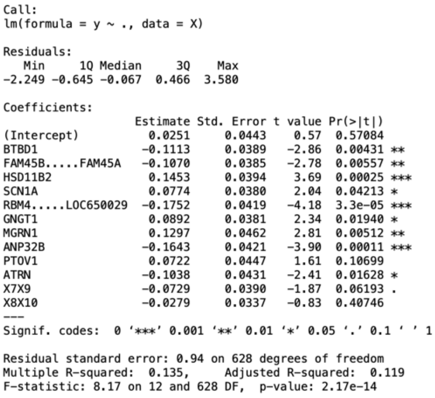




Next, we apply linear regression again with these 10 genes and use Forward selection procedure with P-value (<0.01).

```
                                 Selection Summary
----------------------------------------------------------------------------------
        Variable                          Adj.
Step        Entered            R-Square   R-Square     C(p)        AIC        RMSE
----------------------------------------------------------------------------------
   1    HSD11B2                  0.0277     0.0262    66.4114    1806.0768    0.9868
   2    RBM4.....LOC650029       0.0443     0.0413    56.3800    1797.0181    0.9791
   3    MGRN1                    0.0664     0.0620    42.4370    1784.0637    0.9685
   4    ANP32B                   0.0784     0.0727    35.6935    1777.7119    0.9630
   5    BTBD1                    0.0887     0.0815    30.2695    1772.5339    0.9584
   6    FAM45B.....FAM45A        0.1004     0.0919    23.7867    1766.2327    0.9529
----------------------------------------------------------------------------------
```

So far, the variable is just 6 and we use PCA again but it still not ideal. Then, we consider the cross influence between genes which we can get from the cor plot. So, we consider the influence between ATRN/ ANP32B and MGRN1/ PTOV1.

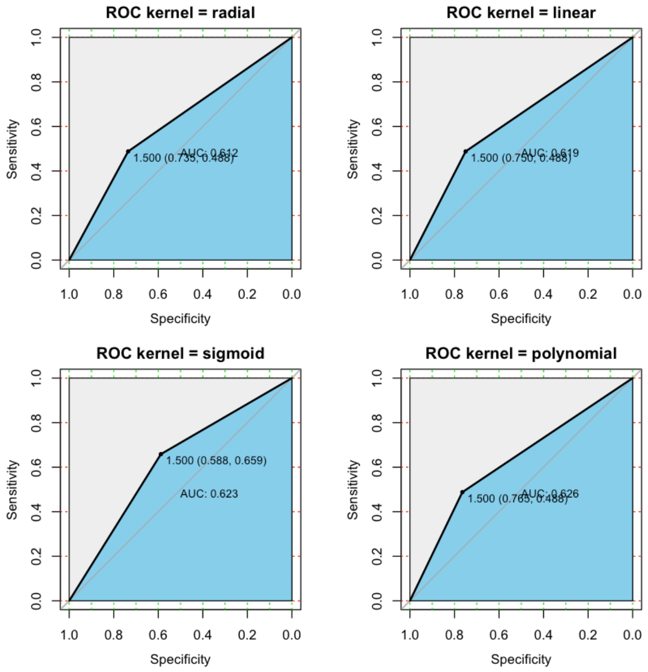


```
Call:
lm(formula = y ~ ., data = X)

Residuals:
   Min     1Q Median     3Q    Max 
-2.249 -0.645 -0.067  0.466  3.580 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)          0.0251     0.0443    0.57  0.57084    
BTBD1               -0.1113     0.0389   -2.86  0.00431 ** 
FAM45B.....FAM45A   -0.1070     0.0385   -2.78  0.00557 ** 
HSD11B2              0.1453     0.0394    3.69  0.00025 ***
SCN1A                0.0774     0.0380    2.04  0.04213 *  
RBM4.....LOC650029  -0.1752     0.0419   -4.18  3.3e-05 ***
GNGT1                0.0892     0.0381    2.34  0.01940 *  
MGRN1                0.1297     0.0462    2.81  0.00512 ** 
ANP32B              -0.1643     0.0421   -3.90  0.00011 ***
PTOV1                0.0722     0.0447    1.61  0.10699    
ATRN                -0.1038     0.0431   -2.41  0.01628 *  
X7X9                -0.0729     0.0390   -1.87  0.06193 .  
X8X10               -0.0279     0.0337   -0.83  0.40746    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.94 on 628 degrees of freedom
Multiple R-squared:  0.135,	Adjusted R-squared:  0.119 
F-statistic: 8.17 on 12 and 628 DF,  p-value: 2.17e-14
```

According to the summary the variable interaction did exist but it is not significant.

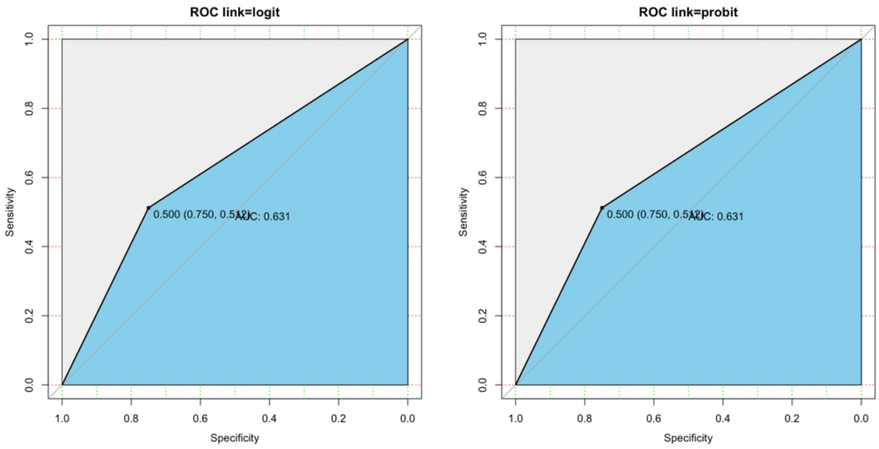
SVM (top 10 genes)

Similarly, we did the SVM again with different kernels and parameters, the outcomes did become better with the accuracy higher than 0.66.



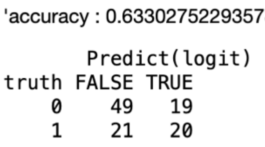
Logistic Regression (top 10 genes)

Here, we try different link function logit/probit to fit the logistic model. But, the result is the same for this situation.



Logistic Regression (top 6 genes)

Lastly, we try the top 6 genes to fit logistic model and the result is still not ideal.



```
'accuracy : 0.63302752293578'

           Predict(logit)
truth FALSE TRUE
    0    49   19
    1    21   20
```

Actually, I also try to change other parameters like type, control etc. or change gene selection but the model cannot improve much.