Chapter 14
Genomic Prediction via LASSO and Support Vector Machine

1. Least absolute shrinkage and selection operator (LASSO)

Lasso was developed by Tibshirani (1996) for both variable selection and prediction. It can handle a model with a very large number of independent variables (predictors). In fact, the number of predictors (*m*) can be many times larger than the sample size (*n*). However, Lasso can only return < *n* non-zero regression coefficients. The linear model is

$$y=\beta_0+\sum_{k=1}^{m}X_k\beta_k+\varepsilon \qquad (1)$$

where β_0 is the intercept, X_k is the *k*th predictor, β_k is the *k*th regression coefficient and ε is a vector of residual errors with an unknown variance σ^2. The Lasso estimates of the regression coefficient is obtained by

$$\beta_{LASSO}=\arg\min_{\beta}\left\{\| y-\beta_0-\sum_{k=1}^{m}X_k\beta_k \|^2+\lambda\sum_{k=1}^{m}|\beta_k|\right\} \qquad (2)$$

This is called L$_1$ penalty. An L$_2$ penalty is

$$\beta_{Ridge}=\arg\min_{\beta}\left\{\| y-\beta_0-\sum_{k=1}^{m}X_k\beta_k \|^2+\lambda\sum_{k=1}^{m}\beta_k^2\right\} \qquad (3)$$

Another version of the L$_2$ penalty is similar to BayesA,

$$\beta_{BayesA}=\arg\min_{\beta}\left\{\| y-\beta_0-\sum_{k=1}^{m}X_k\beta_k \|^2+\sum_{k=1}^{m}\lambda_k\beta_k^2\right\} \qquad (4)$$

In the original Lasso, the shrinkage parameter λ is chosen so that the prediction error is minimum. The prediction error is measured by the mean squared error and is obtained via a K-fold cross-validation, where K = 10 is often used. The software package for LASSO implementation is the R package GlmNet/R (Friedman et al. 2010).

For the same IMF2 data of rice, the LASSO code and the result are given below.

```
dir<-"C:\\Users\\Lecture Notes\\PLS"
setwd(dir)

imf2<-read.csv(file="imf2.csv",header=T)
library(pls)

foldid<-imf2$foldid
y<-imf2$kgw
x<-as.matrix(imf2[,-c(1:9)])
n<-length(y)

library(glmnet)

fit<-cv.glmnet(x=x,y=y,foldid=foldid)
plot(fit)
theta<-coef(fit,s="lambda.min")
beta<-theta[1]
gamma<-theta[-1]
yhat<-predict(fit,newx=x,s="lambda.min")
lambda<-fit$lambda.min
index<-match(lambda,fit$lambda)
mse.cv<-fit$cvm[index]
nonzero<-fit$nzero[index]
sigma2<-sum((y-yhat)^2)/(n-1)
vp<-sum((y-mean(y))^2)/(n-1)
goodness<-drop(cor(y,yhat)^2)
pred_mse<-1-mse.cv/vp
parm<-data.frame(beta,sigma2,vp,nonzero,lambda,mse.cv,
      goodness,pred_mse)
parm

plot(x=1:1619,y=gamma,type="h",xlab="Genome(bin)",ylab="Effect")
```

Table 1. Result of LASSO analysis for the KGW trait of the IMF2 rice population

β_0	σ_e^2	σ_p^2	m_e	λ	MSE.CV	R_{FIT}^2	R_{CV}^2
24.58053	0.342932	3.69547	136	0.016592	1.17458	0.910272	0.682157

The predictability from a 10-fold cross validation for KGW of the IMF2 population is

$$R_{\text{CV}}^2 = 1 - \frac{\text{MSE.CV}}{\sigma_p^2} = 1 - \frac{1.17458}{3.69547} = 0.682157$$

This predictability is slightly higher than that of the gBLUP method (0.662243). The effects of the 1619 bins are plotted in Figure 1 below,

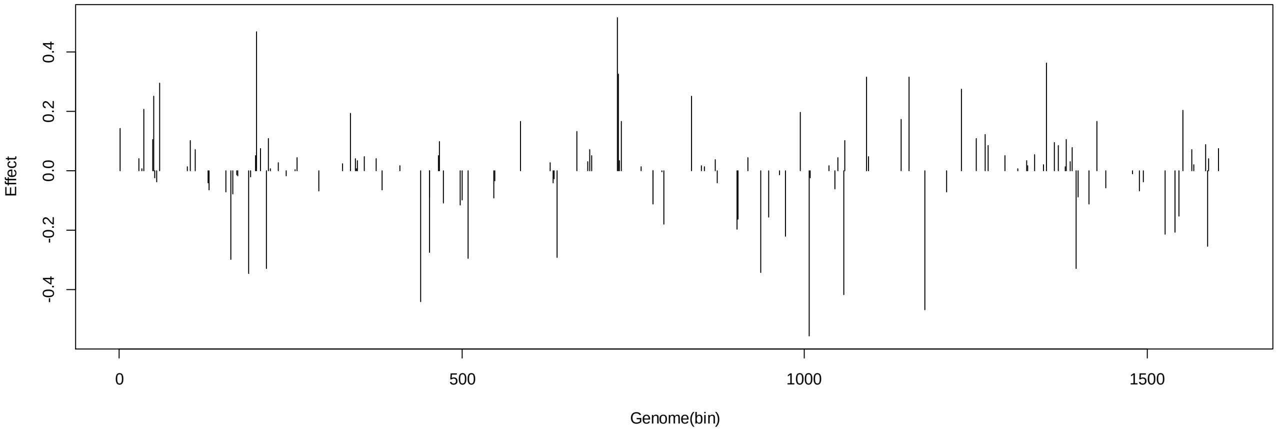

Figure 1. The effects of the 1619 bins estimated from the LASSO method,

2. Support vector machine (SVM)

Support vector machine (Vapnik 2000) is another method for prediction. It is supposed to capture non-linear information of the predictors. In genomic prediction, this method is supposed to be better than any linear predictor if there are epistatic effects between markers (Gianola and van Kaam 2008; Morota et al. 2014). The model is

$$\begin{aligned} y &= \beta_0 + f_X(X \mid \beta) + e \\ &= \beta_0 + K(x, x^T)\beta + e \end{aligned} \tag{5}$$

where

$$\begin{aligned} &K(x, x^T) \text{ is an } n \times n \text{ kernel matrix} \\ &\beta \text{ is an } n \times 1 \text{ vector (unknown)} \end{aligned} \tag{6}$$

There are many different kernels you can choose, but the most popular kernels are the Gaussian kernel (radial basis function) and the polynomial kernel function, which are defined as

$$\begin{aligned} &\text{Gaussian Kernel (Radial Basis Function)} \\ &K_{ij}(x_i, x_j^T) = \exp\left[-\sigma(x_i - x_j)(x_i - x_j)^T\right] \end{aligned} \tag{7}$$

and

$$\begin{aligned} &\text{Polynomial Kernel Function} \\ &K_{ij}(x_i, x_j^T) = (\text{scale } x_i x_j^T + \text{offset})^{\text{degree}} \end{aligned} \tag{8}$$

The σ parameter in the SVM-RBF method is often determined from the data. The scale, offset and degree parameters for the SVM-POLY are preset by the investigators. The SVM methods are also called kernel-based methods. These methods can only be used for prediction, not for variable selection. The “kernlab” package in R provides seven kernels for users to choose (Table 1).

Table 2. Definitions of kernels available in the “kernlab” package of R, where x_j (a row vector of *m* elements) is the marker genotype indicators for individual *j* for $j = 1, \cdots, n$.

Kernel	Formula	Code
Linear kernel	$k(x_i, x_j) = x_i x_j^T$	`kernel = “vanilladot”`
Gaussian radial basis function	$k(x_i, x_j) = \exp(-\sigma \lVert x_i - x_j \rVert^2)$	`kernel = “rbfdot”`
Polynomial kernel	$k(x_i, x_j) = (\text{scale } x_i x_j^T + \text{offset})^{\text{degree}}$	`kernel = “polydot”`
Hyperbolic tangent kernel	$k(x_i, x_j) = \tanh(\text{scale } x_i x_j^T + \text{offset})$	`kernel = “tanhdot”`
Bessel function kernel	$k(x_i, x_j) = \dfrac{\text{Bessel}_{(v+1)}^{n}(\sigma \lVert x_i - x_j \rVert)}{\lVert x_i - x_j \rVert^{-n(v+1)}}$	`kernel = “besseldot”`
Laplace kernel	$k(x_i, x_j) = \exp(-\sigma \lVert x_i - x_j \rVert)$	`kernel = “laplacedot”`
ANOVA kernel	$k(x_i, x_j) = \sum_{k=1}^{n} \exp\left(-\sigma(x_{ik} - x_{jk})^2\right)^d$	`kernel = “anovadot”`

The “kernlab” software uses a sequential minimization optimization (SMO) algorithm to minimize a loss function with two parameters (β_i and β_j) at a time. For two parameters, there is an explicit solution. However, since there are n (sample size) β’s, we must optimize $n(n-1)/2$ times to complete one sweep (iteration). Once all pairs of beta have been updated, the next round of iterations starts and the iteration continues until the sequence converges. Gianola et al (2008) proposed to use the ridge regression (BLUP) method to estimate β. A Bayesian MCMC approach was used to estimate β in the BGLR package of R (de los Campos et al 2013).

The following R codes reads the imf2 data and perform kernel-based prediction.

```
***********************************************************
dir<-"C:\\Users\\SHXU\\Dropbox\\My UCR Teaching\\GEN
234\\Lecture Notes\\SVM"
setwd(dir)
library(kernlab)
imf2<-read.csv(file="imf2.csv",header=T)
foldid<-imf2$foldid
y<-as.matrix(imf2$kgw)
x<-as.matrix(imf2[,-c(1:9)])
n<-length(y)

kern<- ksvm(x=x,y=y,type='eps-svr',kernel = "rbfdot",
            kpar = "automatic", cross=10)
yhat<-predict(kern,x,type='decision')
rbf.fit<-cor(y,yhat)^2
mse<-cross(kern)
vp<-sum((y-mean(y))^2)/(n-1)
rbf.mse<-1-mse/vp
rbf.mse
***********************************************************
```

The predictability obtained from a 10-fold cross validation is

$$R^2_{\text{MSE}} = 1 - \frac{\text{MSE.CV}}{\sigma^2_P} = 1 - \frac{1.458444}{3.69547} = 0.6053428$$

We performed independent 10-fold cross validation to calculate the predictability using squared correlation between observed and predicted phenotypic values. The code and result are shown in next page.

```
***************************************************************
yhat<-NULL
yobs<-NULL
obs<-NULL
for(k in 1:max(foldid)){
     id1<-which(foldid!=k)
     id2<-which(foldid==k)
     x1<-x[id1,]
     x2<-x[id2,]
     y1<-y[id1]
     y2<-y[id2]
     kern.k<- ksvm(x=x1,y=y1,type='eps-svr',kernel = "rbfdot",
                   kpar = "automatic", cross=0)
     ypred<-predict(kern.k,x2,type='decision')
     yhat<-c(yhat,ypred)
     yobs<-c(yobs,y2)
     obs<-c(obs,id2)
}
rbf.pred<-cor(yobs,yhat)^2
R2<-c(rbf.fit,rbf.pred,rbf.mse)
pred<-data.frame(obs,yobs,yhat)
R2
write.csv(x=pred,file="PRED-rbf.csv",row.names=F)
write.csv(x=R2,file="R2-rbf.csv",row.names=F)
****************************************************************
```

The predictability measured by the squared correlation between predicted and observed phenotypes via the 10-fold cross validation is

$$R_{\text{CV}}^2 = \frac{\text{cov}^2(y,\hat{y})}{\text{var}(y)\,\text{var}(\hat{y})} = 0.6248414$$

The Laplace kernel produced a predictability (measured by the squared correlation between predicted and observed phenotypes via the 10-fold cross validation) of 0.3185574, much lower than the Gaussian kernel.

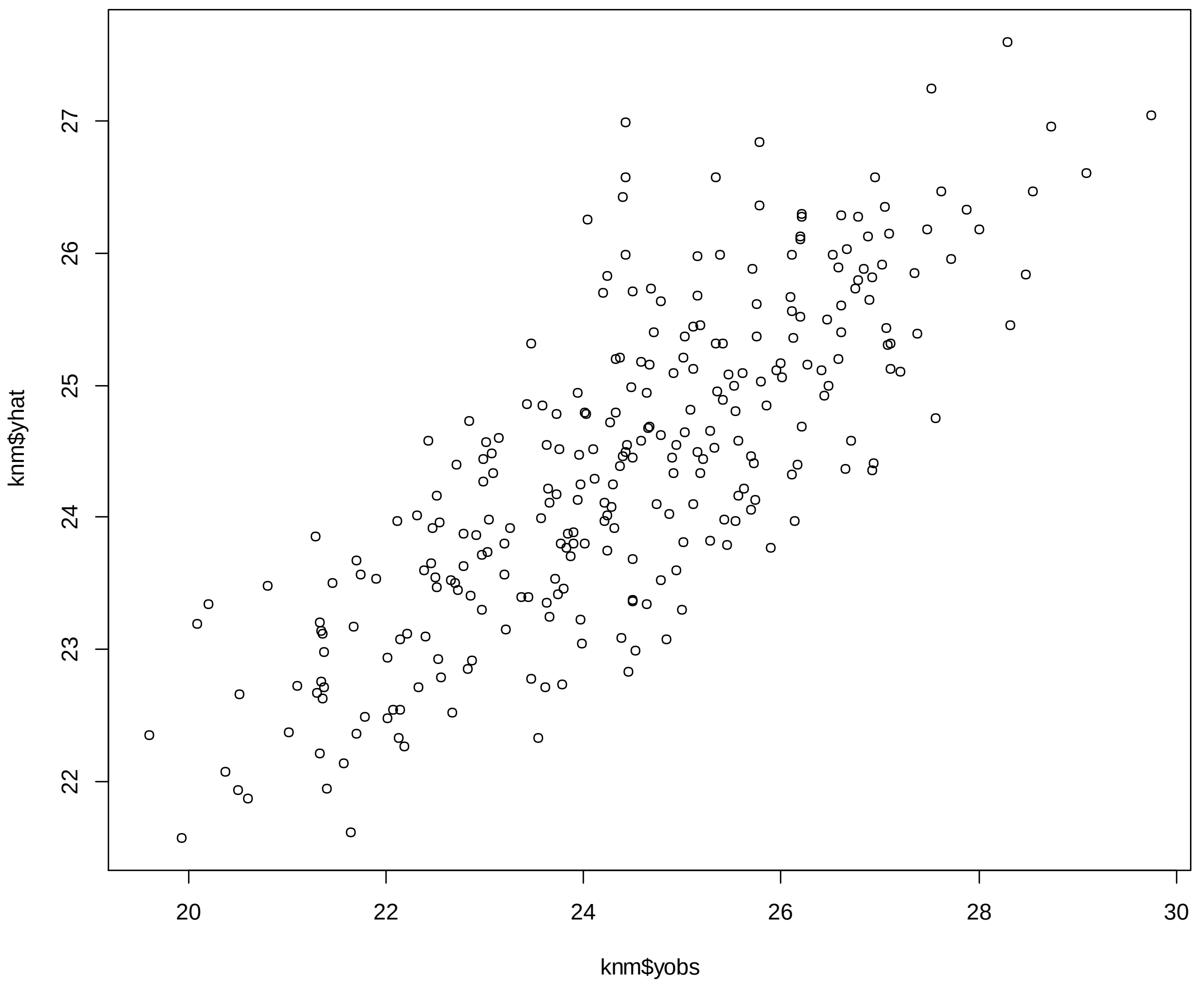

Figure 2. Predicted KGW from 10-fold cross validation of the IMF2 rice against the observed KGW value using the Gaussian radial basis function kernel.