Project 2 - Classification

Jacob Walsh

November 5, 2016

Introduction

Gene Expression Data and Physical Interaction Network (Brem and Kruglyak, 2005): This report classifies protein-protein gene interactions (PPI) in response to Rapamycin. There are 231 active genes, 93 individuals and 6 time points measured for each individual. Predictions for the interacted genes will be done by assessing 5 predictors. X1 - mean of gene i, X2- mean of gene j, X3 - covariance between gene i and gene j, X4- variance of gene i, and X5 - variance of gene j. The data has been split into a training set of 7381 observations and a validation set with 1378 observations. Five models will be looked at: logistic regression model, linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), K-nearest neighbors, and logistic regression generalized additive model (GAM). Sensitivity, specificity and error rate will be calculated for both the training sets and the validation sets.

Analysis

Logistic Regression

A logistic model yields the following coefficients:

```
  (Intercept)            X1            X2            X3            X4 
-14.123839459   0.494346366   0.403707849   1.435135035   0.001905672 
           X5 
 -0.083625432 
```

The predictors X4 and X5 should be noted as not significant based on their resulting p-values (0.984 and 0.387 respectively). Results may differ if we do not include those predictors. The following table compares the predicted outcomes to the actual outcomes. The first table shows the results for the training data set and the second table shows the results when predictions are made on the validation set.
An observation in class 1 indicates that their posterior probability is greater than the cutoff probability, which in this case is 0.172. This cut off probability corresponds to the 201st highest gene pair posterior probability. Hence, in row 1 for the training data, we have the 200 observations that are the most likely interacted gene pairs.

```
                  Y.train
Ghat.train.logistic    0    1
                  0 7062  119
                  1  122   78
```

```
                  Y.valid
Ghat.valid.logistic    0    1
                  0 1301   19
                  1   34   24
```

```
   Classification ErrorRate Sensitivity Specificity
1   Training Data     3.27%       39.6%       98.3%
2 Validation Data     3.85%       55.8%       97.4%
```

Linear Discriminant Analysis

The LDA model doesn't perform as well as the logistic regression. The model sensitivity has decreased from 55.8% to 51.2%. The cut off posterior probability used was 0.143.

```
               Y.train
Ghat.train.lda    0    1
             0 7051  130
             1  133   67

               Y.valid
Ghat.valid.lda    0    1
             0 1311   21
             1   24   22

   Classification ErrorRate Sensitivity Specificity
1   Training Data     3.56%         34%       98.1%
2 Validation Data     3.26%       51.2%       98.2%
```

Quadratic Discriminant Analysis

The accuracy in the predictions from the QDA are very similar to the results from the logistic regression. Here the cut-off posterior probability for the top 200 is 0.35.

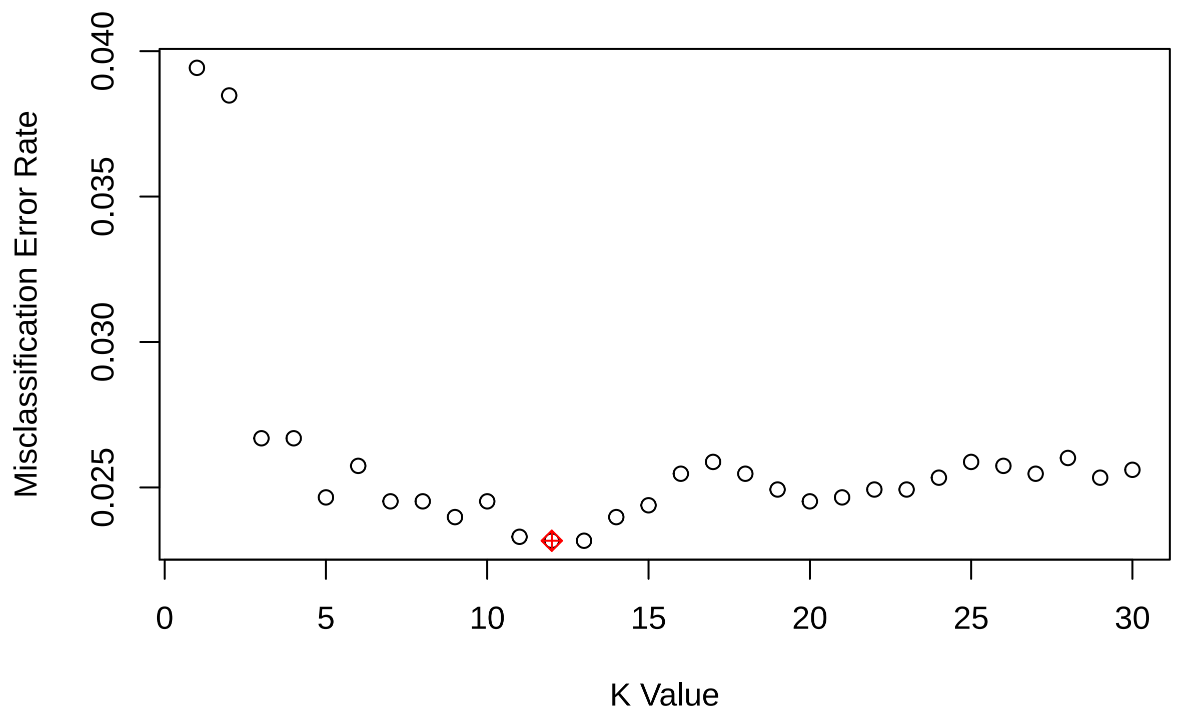
```
               Y.train
Ghat.train.qda    0    1
             0 7055  126
             1  129   71

               Y.valid
Ghat.valid.qda    0    1
             0 1307   19
             1   28   24

   Classification ErrorRate Sensitivity Specificity
1   Training Data     3.45%         36%       98.2%
2 Validation Data     3.41%       55.8%       97.9%
```

KNN

The K-th nearest neighbor analysis was done by using cross-validation to choose the optimal k value. The misclassification error is minimized when choosing k = 12, so each model is constructed using 12 as the index.



A cut-off posterior probability for the most likely interacted pairs in the KNN is 0.25.

```
              Y.train
Ghat.train.knn    0    1
             0 7108  105
             1   76   92

              Y.valid
Ghat.valid.knn    0    1
             0 1301   23
             1   34   20

   Classification ErrorRate Sensitivity Specificity
1   Training Data     2.45%       46.7%       98.9%
2 Validation Data     4.13%       46.5%       97.4%
```

For the validation set, the sensitivity of the KNN model is lower than 50% and the error rate is much higher at over 4%.

Logistic Regression Generalized Additive Model (GAM)

The GAM was constructed using a smoothing spline with df=5 for each of the 5 predictors.

```
              Y.train
Ghat.train.gam    0    1
```

```
             0 7071  110
             1  113   87

              Y.valid
Ghat.valid.gam    0    1
             0 1254   19
             1   81   24

   Classification ErrorRate Sensitivity Specificity
1   Training Data      3.0%       44.1%       98.4%
2 Validation Data     7.26%       55.8%       93.9%
```

Although the sensitivity of the GAM is comparable to that of the logistic and qda models, the error rate is much higher. The specificity is nearly 5% lower in this model.

Results

The following table shows the final validation set results for all 5 models.

```
  ModelType ErrorRate Sensitivity Specificity
1  Logistic     3.85%       55.8%       97.4%
2       LDA     3.26%       51.2%       98.2%
3       QDA     3.41%       55.8%       97.9%
4       KNN     4.13%       46.5%       97.4%
5       GAM     7.26%       55.8%       93.9%
```

Conclusion

In summary, the sensitivity of the logistic, QDA, and GAM model are identical but the QDA has the highest specificity. The QDA model is then concluded to be the “best” when predicting the interacting protein-protein gene pairs. Using the QDA model the following observations would be selected with the cut off posterior probability criteria used:

```
 [1] "174"  "180"  "600"  "602"  "608"  "617"  "621"  "623"  "625"  "627" 
[11] "636"  "637"  "677"  "683"  "692"  "696"  "698"  "700"  "702"  "711" 
[21] "712"  "754"  "761"  "763"  "767"  "771"  "773"  "782"  "783"  "952" 
[31] "956"  "960"  "962"  "971"  "972"  "1131" "1135" "1172" "1176" "1178"
[41] "1187" "1188" "1246" "1257" "1258" "1286" "1302" "1311" "1312" "1332"
[51] "1333" "1378"
```

APPENDIX 1 - R CODE

Logistic Regression

```
model.logistic <- glm(Y ~., data.train, family=binomial("logit"))
summary(model.logistic)
post.train.logistic <- model.logistic$fitted.values # n.train posterior probabilities of Y=1

Ghat.train.logistic <-ifelse(post.train.logistic > sort(post.train.logistic, decreasing=T)[201], 1, 0)
table(Ghat.train.logistic,Y.train) # classification table

sum(abs(Ghat.train.logistic-Y.train))/n.train # training data classification error rate
sum(Ghat.train.logistic==1&Y.train==1)/sum(Y.train==1) # sensitivity
sum(Ghat.train.logistic==0&Y.train==0)/sum(Y.train==0) # specificity

post.valid.logistic <- predict(model.logistic, data.valid, type="response") # n.valid post probs
Ghat.valid.logistic <- ifelse(post.valid.logistic > sort(post.train.logistic, decreasing=T)[201], 1, 0)
table(Ghat.valid.logistic,Y.valid)
sum(abs(Ghat.valid.logistic-Y.valid))/n.valid # classification error rate
sum(Ghat.valid.logistic==1&Y.valid==1)/sum(Y.valid==1) # sensitivity
sum(Ghat.valid.logistic==0&Y.valid==0)/sum(Y.valid==0) # specificity
```

Linear Discriminant Analysis

```
library(MASS)
model.lda <- lda(Y ~ ., data.train)
model.lda
plot(model.lda)
post.train.lda <- predict(model.lda)$posterior[,2] # n.train posterior probabilities of Y=1

sort(post.train.lda, decreasing=T)[201]
Ghat.train.lda <- ifelse(post.train.lda > sort(post.train.lda, decreasing=T)[201], 1, 0)  # classificat

table(Ghat.train.lda,Y.train)
sum(abs(Ghat.train.lda-Y.train))/n.train # training data classification error rate
sum(Ghat.train.lda==1&Y.train==1)/sum(Y.train==1) # sensitivity

sum(Ghat.train.lda==0&Y.train==0)/sum(Y.train==0) # specificity

post.valid.lda <- predict(model.lda, data.valid)$posterior[,2] # n.valid posterior probabilities of Y=1
Ghat.valid.lda <- ifelse(post.valid.lda > sort(post.train.lda, decreasing=T)[201], 1, 0)
table(Ghat.valid.lda,Y.valid) # classification table

sum(abs(Ghat.valid.lda-Y.valid))/n.valid # classification error rate

sum(Ghat.valid.lda==1&Y.valid==1)/sum(Y.valid==1) # sensitivity

sum(Ghat.valid.lda==0&Y.valid==0)/sum(Y.valid==0) # specificity
```

Quadratic Discriminant Analysis

```
model.qda <- qda(Y ~ ., data.train)
post.train.qda <- predict(model.qda)$posterior[,2]
Ghat.train.qda <- ifelse(post.train.qda > sort(post.train.qda, decreasing=T)[201], 1, 0)  # classificat

table(Ghat.train.qda,Y.train) # classification table

sum(abs(Ghat.train.qda-Y.train))/n.train # training data classification error rate

sum(Ghat.train.qda==1&Y.train==1)/sum(Y.train==1) # sensitivity

sum(Ghat.train.qda==0&Y.train==0)/sum(Y.train==0) # specificity

post.valid.qda <- predict(model.qda, data.valid)$posterior[,2] # n.valid posterior probabilities of Y=1
Ghat.valid.qda <- ifelse(post.valid.qda > sort(post.train.qda, decreasing=T)[201], 1, 0)
table(Ghat.valid.qda,Y.valid) # classification table

sum(abs(Ghat.valid.qda-Y.valid))/n.valid # classification error rate

sum(Ghat.valid.qda==1&Y.valid==1)/sum(Y.valid==1) # sensitivity

sum(Ghat.valid.qda==0&Y.valid==0)/sum(Y.valid==0) # specificity
```

KNN

```
library(class)
mer <- rep(NA, 30) # misclassification error rates based on leave-one-out cross-validation

set.seed(2014)

for (i in 1:30) mer[i] <- sum((Y.train-(c(knn.cv(train=X.train, cl=Y.train, k=i))-1))^2)/n.train
plot(mer, ylab="Misclassification Error Rate", xlab="K Value")
points(which.min(mer), mer[which.min(mer)], pch=9, col="red")
which.min(mer) # minimum occurs at k=12
set.seed(2014)
trainmodel.knn <- knn(train=scale(X.train), test=scale(X.train), cl=scale(Y.train), k=12, prob=T)

trainpredclass.knn <- c(trainmodel.knn)-1 # convert factor to numeric classes

trainpredprob.knn <- attr(trainmodel.knn, "prob") # proportion of votes for winning class

post.train.knn <- trainpredclass.knn*trainpredprob.knn+(1-trainpredclass.knn)*(1-trainpredprob.knn)

Ghat.train.knn <- ifelse(post.train.knn > sort(post.train.knn, decreasing=T)[201], 1, 0) # classificati

table(Ghat.train.knn,Y.train)
sum(abs(Ghat.train.knn-Y.train))/n.train # training data classification error rate
sum(Ghat.train.knn==1&Y.train==1)/sum(Y.train==1) # sensitivity
sum(Ghat.train.knn==0&Y.train==0)/sum(Y.train==0) # specificity
```

```r
set.seed(2014)
model.knn <- knn(train=scale(X.train), test=scale(X.valid), cl=scale(Y.train), k=12, prob=T)
predclass.knn <- c(model.knn)-1 # convert factor to numeric classes

predprob.knn <- attr(model.knn, "prob")

post.valid.knn <- predclass.knn*predprob.knn+(1-predclass.knn)*(1-predprob.knn)

Ghat.valid.knn <-  ifelse(post.valid.knn > sort(post.train.knn, decreasing=T)[201], 1, 0)

table(Ghat.valid.knn,Y.valid) # classification table

sum(abs(Ghat.valid.knn-Y.valid))/n.valid # classification error rate

sum(Ghat.valid.knn==1&Y.valid==1)/sum(Y.valid==1) # sensitivity

sum(Ghat.valid.knn==0&Y.valid==0)/sum(Y.valid==0) # specificity
```

Logistic Regression Generalized Additive Model (GAM)

```r
library(gam)
model.gam <- gam(Y ~ s(X1,df=5) + s(X2,df=5)+ s(X3,df=5)+ s(X4,df=5)+ s(X5,df=5), data.train, family=bi
summary(model.gam)
post.train.gam <- model.gam$fitted.values # n.train posterior probabilities of Y=1

Ghat.train.gam <- ifelse(post.train.gam > sort(post.train.gam, decreasing=T)[201], 1, 0) # classificati

table(Ghat.train.gam,Y.train) # classification table

sum(abs(Ghat.train.gam-Y.train))/n.train # training data classification error rate

sum(Ghat.train.gam==1&Y.train==1)/sum(Y.train==1) # sensitivity

sum(Ghat.train.gam==0&Y.train==0)/sum(Y.train==0) # specificity

post.valid.gam <- predict(model.gam, data.valid, type="response") # n.valid post probs

Ghat.valid.gam <- ifelse(post.valid.gam > sort(post.train.gam, decreasing=T)[201], 1, 0)
table(Ghat.valid.gam,Y.valid) # classification table
sum(abs(Ghat.valid.gam-Y.valid))/n.valid # classification error rate

sum(Ghat.valid.gam==1&Y.valid==1)/sum(Y.valid==1) # sensitivity

sum(Ghat.valid.gam==0&Y.valid==0)/sum(Y.valid==0) # specificity
```