CSC 529 Advanced Data Mining - Assignment 4

Akhil Kumar Ramasagaram

Saturday, March 05, 2016

Problem 1

a) Define decision boundary. Explain the differences in the decision boundaries for two classifiers of your choice.

A decision boundary is the region of a problem space in which the output label of a classification is difficult to calculate. If the decision surface is a hyperplane, then the classification problem is linear and the classes are linearly seperable.

b) In Bayes Decision Theory, what does the prior probability capture?

So in bayesian decision theory prior probability distribution is the probability distribution that can express one's beliefs about certain quantity before some evidence is taken into account.

c) What is the zero-one loss function?

In a classification problem with we use a loss function for evaluating model $l : -1, 1X-1, 1- > R$ which measures the error of a given prediction. The value of the loss function l at an arbitary point$(y, \hat{y})$ is interpreted as the cost incurred by predicting $\hat{y}$ when the true label is y. In general classification task this loss function is class zero-one loss function.

d) What is the basic idea of the curse of dimensionality?

when building a model, the more features you have the better the chance of getting a higher performance on models. But it comes at a price, the cost of computing also increases, there will a cut-off where we have to choose how many features are we comfortable introducing in our model. If the features are really important then we can use methods like PCA or LDA to reduce the dimensions. This is the curse of dimensionality phenomenon.

e) Define three evaluation metrics used for probabilistic classifiers' performance evaluation and explain the advantages and disadvantages of using one versus the others.

For probabilistic classifier we can use log-loss, mean squared error or AUC curve for evaluation. The MSE makes an excellent general purpose error metric for numerical predictions. In log-loss, the use of log on the error provides extreme punishment for being both confident and wrong.In AUC, the receiver operating characteristic is a graphical plot that illustrates the performance of a binary classifications.

Problem 2

1) Browse the UCI Machine Learning Repository and select three datasets with the following characteristics / constraints

a) Provide a short introduction describing your rationale for your choice of the datasets.

I choose my dataset to represent different problem like too many observations and too less observations. The smallest one has 583 observations with 10 features while my biggest dataset has 30K observations with 24 features.

b) Provide data set description and characteristics.

My first dataset is ILPD. This data set contains 416 liver patient records and 167 non liver patient records.The data set was collected from north east of Andhra Pradesh, India. Selector is a class label used to divide into groups(liver patient or not). This data set contains 441 male patient records and 142 female patient records. Any patient whose age exceeded 89 is listed as being of age "90".

My second dataset is Teaching Assistant Evaluation. The data consist of evaluations of teaching performance over three regular semesters and two summer semesters of 151 teaching assistant (TA) assignments at the Statistics Department of the University of Wisconsin-Madison. The scores were divided into 3 roughly equal-sized categories ("low", "medium", and "high") to form the class variable.

My third dataset is Default of credit card.This research aimed at the case of customer default payments in Taiwan. There are 23 features and 30K observations.

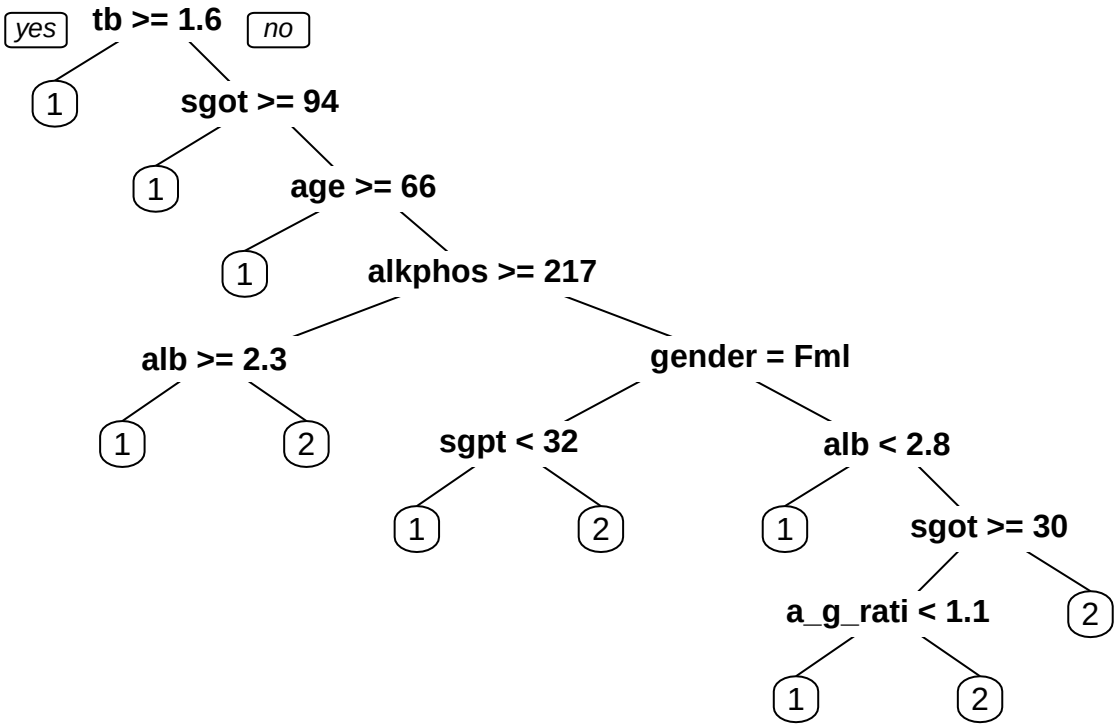
c) Discuss any other relevant data issues, for example, class distribution, data normalization / standardization, etc. Tables and plots can be included.

The class distribution for the abalone is highly skewed, there are 29 different classes and class 10 has 689 observations while some classes like class1, class2, class 29 have a single observation.

2) Apply at least three classification schemes (e.g., decision tree, logistic regression, KNN, SVM, Naïve Bayes) to each data set and evaluate / compare their performance. For SVM, you can choose from three types of kernels (linear kernel, polynomial kernel, and RBF kernel), and each one counts for one scheme. You can also try some meta classification algorithms such as Bagging and Boosting.

ILPD data

```
library(caret)
library(glmnet)
library(e1071)
library(rpart)
library(rpart.plot)
ILPD <- read.table("Indian Liver Patient Dataset (ILPD).csv", header = F,stringsAsFactors = F, sep = ",
names(ILPD) <- c("age","gender","tb","db","alkphos","sgpt","sgot","tp","alb","a_g_ratio","target")
ILPD <- na.omit(ILPD)
ind <- createDataPartition(ILPD$target, p = 0.7, times = 1, list = F)
train_data <- ILPD[ind,]
test_data <- ILPD[-ind,]
rp_model <- rpart(target ~., data = train_data,method = "class")
rpart.plot(rp_model)
```




```
rp_pred <- predict(rp_model, newdata = test_data[,-11], type = "class")
confusionMatrix(rp_pred, test_data$target)$overall[1]
```

```
##  Accuracy
## 0.7109827
```

```
# tuning
rp_tuned <- train(as.factor(target) ~ age + gender + tb + db + alkphos + sgpt + sgot + tp +
    alb + a_g_ratio, data=train_data, method = "rpart", tuneGrid = expand.grid(cp = seq(0,.25,0.05)))
rp_tuned_pred <- predict(rp_tuned, test_data)
confusionMatrix(rp_tuned_pred, test_data$target)$overall[1]
```

```
## Accuracy
## 0.716763
```

```
#glmnet
X <- train_data[,-11]
X$gender <- ifelse(X$gender == "Male",1,0)
X <- data.matrix(X)
Y <- train_data$target
glmnet_model<- cv.glmnet(x = X,y=Y, alpha = 1, family = "binomial")
X2 <- test_data[,-11]
X2$gender <- ifelse(X2$gender == "Male",1,0)
X2 <- data.matrix(X2)
glmnet_pred <- predict(glmnet_model, X2,s="lambda.min", type = "class")
confusionMatrix(glmnet_pred, test_data$target)$overall[1]
```

```
## Accuracy
## 0.699422
```

```
svm_model <- svm(target ~., data = train_data, type = "C-classification")
confusionMatrix(predict(svm_model, train_data[,-11]), train_data$target)$overall[1]
```

```
##  Accuracy
## 0.7142857
```

```
svm_pred <- predict(svm_model, test_data)
confusionMatrix(svm_pred, test_data$target)$overall[1]
```

```
## Accuracy
## 0.716763
```

```
svmTuneGrid <- data.frame( .C = 2^(-2:7))
svmFit <- train(as.factor(target) ~ .,
    data = train_data,
    method = "svmLinear",
    tuneGrid = svmTuneGrid,
    trControl = trainControl(method = "cv", number = 3))
confusionMatrix(predict(svmFit, train_data), train_data$target)$overall[1]
```

```
##  Accuracy
## 0.7142857
```

```
svm_tuned_pred <- predict(svmFit, test_data)
confusionMatrix(svm_tuned_pred, test_data$target)$overall[1]
```

```
## Accuracy
## 0.716763
```

So the best model was decision tree, the other two models had similar performance because both of them were kind of linear classfiers while the decision tree uses split to classify an object.

Credit card

```
library(randomForest)
cc <- read.csv("default of credit card clients.csv", header = T)
names(cc)[ncol(cc)] <- "target"
ind <- createDataPartition(cc$target, p = 0.7, times = 1, list = F)
train_data <- cc[ind,]
test_data <- cc[-ind,]
## RF
rf_model <- randomForest(as.factor(target) ~. - ID, train_data)
rf_pred <- predict(rf_model, test_data)
confusionMatrix(rf_pred, test_data$target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    0    1
##          0 6607 1286
##          1  406  701
##
##                Accuracy : 0.812
##                  95% CI : (0.8038, 0.82)
##     No Information Rate : 0.7792
##     P-Value [Acc > NIR] : 1.22e-14
##
##                   Kappa : 0.3505
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9421
##             Specificity : 0.3528
##          Pos Pred Value : 0.8371
##          Neg Pred Value : 0.6332
##              Prevalence : 0.7792
##          Detection Rate : 0.7341
##    Detection Prevalence : 0.8770
##       Balanced Accuracy : 0.6475
##
##        'Positive' Class : 0
##
```

```
rf_model <- randomForest(as.factor(target) ~. - ID, train_data, mtry = 5)
rf_pred <- predict(rf_model, test_data)
confusionMatrix(rf_pred, test_data$target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 6608 1281
##          1  405  706
##
##                Accuracy : 0.8127
##                  95% CI : (0.8044, 0.8207)
##     No Information Rate : 0.7792
##     P-Value [Acc > NIR] : 3.524e-15
##
##                   Kappa : 0.3534
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9423
##             Specificity : 0.3553
##          Pos Pred Value : 0.8376
##          Neg Pred Value : 0.6355
##              Prevalence : 0.7792
##          Detection Rate : 0.7342
##    Detection Prevalence : 0.8766
##       Balanced Accuracy : 0.6488
##
##        'Positive' Class : 0
##
```

```
library(class)
knn_model <- knn(train_data[,!names(train_data) %in% c("ID","target")], test = test_data[,!names(test_da
confusionMatrix(knn_model, test_data$target)$overall[1]
```

```
##  Accuracy
## 0.6954444
```

```
X <- data.matrix(train_data[,!names(train_data) %in% c("ID","target")])
Y <- as.factor(train_data$target)
glmnet_model<- cv.glmnet(x = X,y=Y, alpha = 1, family = "binomial")
X2 <- data.matrix(test_data[,!names(test_data) %in% c("ID","target")])
glmnet_pred <- predict(glmnet_model, X2,s="lambda.min", type = "class")
confusionMatrix(glmnet_pred, test_data$target)$overall[1]
```

```
##  Accuracy
## 0.8078889
```

Here the best model was a randomforest model, the svm was very close to it. The third model was a k nearest neighbour classifier and its performance was only 0.7. I think the way the data points are placed for this particular data is not feasible for a k nearest neighbour problem so it didn't do better.

TA Evaluation

This particular data is only has 151 observations so i implmented a 10 k cross-validation.

```
ta <- read.csv("tae.txt", header = F)
names(ta) <- c("eng_prof","ci","course","semester","class_size","target")
ta$eng_prof <- as.factor(ta$eng_prof)
ta$ci <- as.factor(ta$ci)
ta$course <- as.factor(ta$course)
ta$semester <- as.factor(ta$semester)
ta$target <- as.factor(ta$target)

ten_cv <- function(method){
  k= 10
  n=floor(nrow(ta)/k)
  acc.vect <- rep(NA,k)
  for(i in 1:10){
    s1 = ((i-1)*n+1)
    s2 = (i*n)
    subset = s1:s2
    train_data = ta[-subset,]
    test_data = ta[subset,]
    if(method == "rf"){
      model <- randomForest(x = train_data[,-6], y = train_data$target)
      pred <- predict(model,test_data)
      acc.vect[i] <- round(confusionMatrix(pred, test_data$target)$overall[1],3)
      print(paste("Accuracy at ",i," cv is: ",acc.vect[i],sep = ""))
    }
    if(method == "svm"){
      svm_model <- svm(as.factor(target) ~., train_data)
```

```
      svm_pred <- predict(svm_model, test_data, type = "response")
      acc.vect[i] <- round(confusionMatrix(svm_pred, test_data$target)$overall[1],3)
      print(paste("Accuracy at ",i," cv is: ",acc.vect[i],sep = ""))
    }
    if(method == "knn"){
      knn_model <- knn(train = train_data[,-6],test = test_data[,-6],cl = train_data$target, k = 3)
      print(table(test_data$target, knn_model, dnn = c("original","predicted")))
    }
  }
}

ten_cv("rf")
```

```
## [1] "Accuracy at 1 cv is: 0.867"
## [1] "Accuracy at 2 cv is: 0.933"
## [1] "Accuracy at 3 cv is: 0.933"
## [1] "Accuracy at 4 cv is: 0.867"
## [1] "Accuracy at 5 cv is: 0.933"
## [1] "Accuracy at 6 cv is: 0.4"
## [1] "Accuracy at 7 cv is: 0.267"
## [1] "Accuracy at 8 cv is: 0.333"
## [1] "Accuracy at 9 cv is: 0.133"
## [1] "Accuracy at 10 cv is: 0.467"
```

```
ten_cv("svm")
```

```
## [1] "Accuracy at 1 cv is: 0.133"
## [1] "Accuracy at 2 cv is: 0.067"
## [1] "Accuracy at 3 cv is: 0.133"
## [1] "Accuracy at 4 cv is: 0.067"
## [1] "Accuracy at 5 cv is: 0"
## [1] "Accuracy at 6 cv is: 0.133"
## [1] "Accuracy at 7 cv is: 0.067"
## [1] "Accuracy at 8 cv is: 0.067"
## [1] "Accuracy at 9 cv is: 0.2"
## [1] "Accuracy at 10 cv is: 0.067"
```

```
ten_cv("knn")
```

```
##          predicted
## original 1 2 3
##        1 0 0 0
##        2 0 1 0
##        3 4 2 8
##          predicted
## original 1 2 3
##        1 1 1 0
##        2 4 7 2
##        3 0 0 0
##          predicted
## original 1 2 3
##        1 4 3 2
```

```
##        2 0 0 0
##        3 2 1 3
##         predicted
## original 1 2 3
##        1 0 0 0
##        2 1 5 1
##        3 2 1 5
##         predicted
## original 1 2 3
##        1 3 1 4
##        2 4 3 0
##        3 0 0 0
##         predicted
## original 1 2 3
##        1 1 2 0
##        2 0 0 0
##        3 4 2 6
##         predicted
## original 1 2 3
##        1 0 0 0
##        2 3 3 4
##        3 1 2 2
##         predicted
## original 1 2 3
##        1 2 5 6
##        2 2 0 0
##        3 0 0 0
##         predicted
## original 1 2 3
##        1 1 0 0
##        2 3 1 3
##        3 2 3 2
##         predicted
## original 1 2 3
##        1 1 5 6
##        2 1 1 1
##        3 0 0 0
```

the randomforest had average accuracy of 0.62 approximately, the svm did very poor and knn classifier was the worst. The think the fact that the data was very limited and our target function had 3 classes made it difficult to classify.

Problem 3

Write a half page summary of a research paper written in the last three years that shows the applicability of one of the techniques learned in the course to a domain of interest to you. You summary should include: