Statistics 202: Statistical Aspects of Data Mining

Professor David Mease

Tuesday, Thursday 9:00-10:15 AM Terman 156

Lecture 13 = Finish Chapter 5 and Chapter 8

Agenda:

1) Reminder about 5th Homework (due Tues 8/14 at 9AM)
2) Discuss Final Exam
3) Lecture over rest of Chapter 5 (Section 5.6)
4) Lecture over Chapter 8 (Sections 8.1 and 8.2)

Homework Assignment:

Chapter 5 Homework Part 2 and Chapter 8 Homework is due Tuesday 8/14 at 9AM.

Either email to me (dmease@stanford.edu), bring it to class, or put it under my office door.

SCPD students may use email or fax or mail.

The assignment is posted at

http://www.stats202.com/homework.html

Important: If using email, please submit only a single file (word or pdf) with your name and chapters in the file name. Also, include your name on the first page. Finally, please put your name and the homework # in the subject of the email.

Final Exam

I have obtained permission to have the final exam from 9 AM to 12 noon on Thursday 8/16 in the classroom (Terman 156)

I will assume the same people will take it off campus as with the midterm so please let me know if

1) You are SCPD and took the midterm on campus but need to take the final off campus

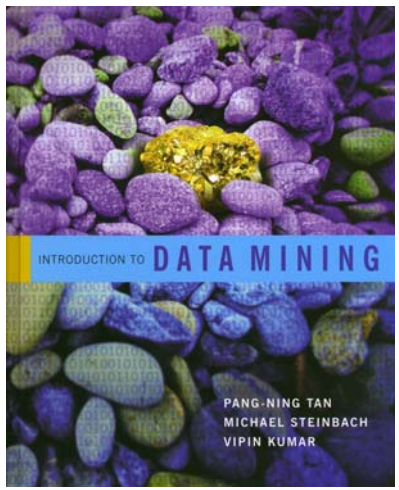
or

2) You are SCPD and took the midterm off campus but want to take the final on campus

More details to come...

Introduction to Data Mining

by
Tan, Steinbach, Kumar



Chapter 5: Classification: Alternative Techniques

Ensemble Methods (Section 5.6, page 276)

- **Ensemble methods aim at "improving classification accuracy by aggregating the predictions from multiple classifiers" (page 276)**

- **One of the most obvious ways of doing this is simply by averaging classifiers which make errors somewhat independently of each other**

In class exercise #45:

Suppose I have 5 classifiers which each classify a point correctly 70% of the time. If these 5 classifiers are completely independent and I take the majority vote, how often is the majority vote correct for that point?

In class exercise #45:

Suppose I have 5 classifiers which each classify a point correctly 70% of the time. If these 5 classifiers are completely independent and I take the majority vote, how often is the majority vote correct for that point?

Solution (continued):

```
10*.7^3*.3^2 + 5*.7^4*.3^1 + .7^5
```

or

```
1-pbinom(2, 5, .7)
```

In class exercise #46:

Suppose I have 101 classifiers which each classify a point correctly 70% of the time. If these 101 classifiers are completely independent and I take the majority vote, how often is the majority vote correct for that point?

In class exercise #46:

Suppose I have 101 classifiers which each classify a point correctly 70% of the time. If these 101 classifiers are completely independent and I take the majority vote, how often is the majority vote correct for that point?

Solution (continued):

```
1-pbinom(50, 101, .7)
```

Ensemble Methods (Section 5.6, page 276)

- **Ensemble methods include**

-Bagging (page 283)

-Random Forests (page 290)

-Boosting (page 285)

- **Bagging builds different classifiers by training on repeated samples (with replacement) from the data**

- **Random Forests averages many trees which are constructed with some amount of randomness**

- **Boosting combines simple base classifiers by upweighting data points which are classified incorrectly**

Random Forests (Section 5.6.6, page 290)

- One way to create random forests is to grow decision trees top down but at each terminal node consider only a random subset of attributes for splitting instead of all the attributes

- Random Forests are a very effective technique

- They are based on the paper

L. Breiman. Random forests. Machine Learning, 45:5-32, 2001

- They can be fit in R using the function randomForest() in the library randomForest

In class exercise #47:

Use randomForest() in R to fit the default Random Forest to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Compute the misclassification error for the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv

In class exercise #47:

Use randomForest() in R to fit the default Random Forest to the last column of the sonar training data at http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Compute the misclassification error for the test data at http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv

Solution:

```
install.packages("randomForest")
library(randomForest)
train<-read.csv("sonar_train.csv",header=FALSE)
test<-read.csv("sonar_test.csv",header=FALSE)
y<-as.factor(train[,61])
x<-train[,1:60]
y_test<-as.factor(test[,61])
x_test<-test[,1:60]
fit<-randomForest(x,y)
1-sum(y_test==predict(fit,x_test))/length(y_test)
```

Boosting (Section 5.6.5, page 285)

- **Boosting has been called the "best off-the-shelf classifier in the world"**

- **There are a number of explanations for boosting, but it is not completely understood why it works so well**

- **The most popular algorithm is AdaBoost from**

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, 1996.

Boosting (Section 5.6.5, page 285)

- **Boosting can use any classifier as its *weak learner* (base classifier) but decision trees are by far the most popular**

- **Boosting usually gives zero training error, but rarely overfits which is very curious**

Boosting (Section 5.6.5, page 285)

- **Boosting works by upweighing points at each iteration which are misclassified**

- **On paper, boosting looks like an optimization (similar to maximum likelihood estimation), but in practice it seems to benefit a lot from averaging like Random Forests does**

- **There exist R libraries for boosting, but these are written by statisticians who have their own views of boosting, so I would not encourage you to use them**

- **The best thing to do is to write code yourself since the algorithms are very basic**

AdaBoost

- **Here is a version of the AdaBoost algorithm**

- Fit the classifier g_m to the training data using weights w_i where g_m maps each x_i to -1 or 1.
- Compute the weighted error rate $\epsilon_m \equiv \sum_{i=1}^{n} w_i \mathrm{I}[y_i \neq g_m(x_i)]$ and half its log-odds, $\alpha_m \equiv \frac{1}{2} \log \frac{1-\epsilon_m}{\epsilon_m}$.
- Let $F_m = F_{m-1} + \alpha_m g_m$.
- Replace the weights w_i with $w_i \equiv w_i e^{-\alpha_m g_m(x_i) y_i}$ and then renormalize by replacing each w_i by $w_i / (\sum w_i)$.

- **The algorithm repeats until a chosen stopping time**

- **The final classifier is based on the sign of F_m**

In class exercise #48:

Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

Solution:

```
train<-read.csv("sonar_train.csv",header=FALSE)
test<-read.csv("sonar_test.csv",header=FALSE)
y<-train[,61]
x<-train[,1:60]
y_test<-test[,61]
x_test<-test[,1:60]
```

In class exercise #48:

Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

Solution (continued):

```
train_error<-rep(0,500)
test_error<-rep(0,500)
f<-rep(0,130)
f_test<-rep(0,78)
i<-1
library(rpart)
```

In class exercise #48:

Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

Solution (continued):

```
while(i<=500){
  w<-exp(-y*f)
  w<-w/sum(w)
  fit<-rpart(y~.,x,w,method="class")
  g<--1+2*(predict(fit,x)[,2]>.5)
  g_test<--1+2*(predict(fit,x_test)[,2]>.5)
  e<-sum(w*(y*g<0))
```

In class exercise #48:

Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

Solution (continued):

```
  alpha<-.5*log ( (1-e) / e )
  f<-f+alpha*g
  f_test<-f_test+alpha*g_test
  train_error[i]<-sum(1*f*y<0)/130
  test_error[i]<-sum(1*f_test*y_test<0)/78
  i<-i+1
}
```

In class exercise #48:

Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

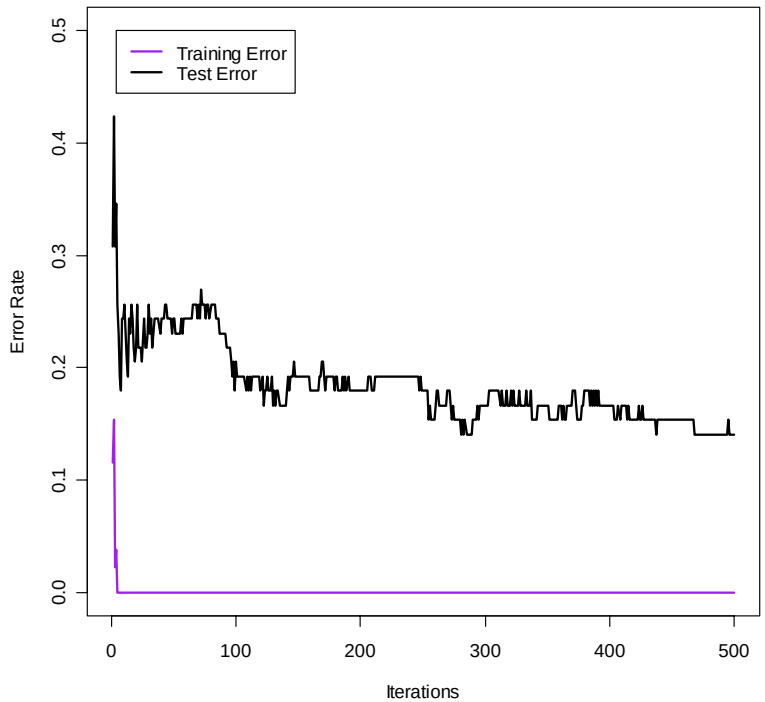
Solution (continued):

```
plot(seq(1,500),test_error,type="l",
   ylim=c(0,.5),
   ylab="Error Rate",xlab="Iterations",lwd=2)
lines(train_error,lwd=2,col="purple")
legend(4,.5,c("Training Error","Test Error"),
    col=c("purple","black"),lwd=2)
```

In class exercise #48:

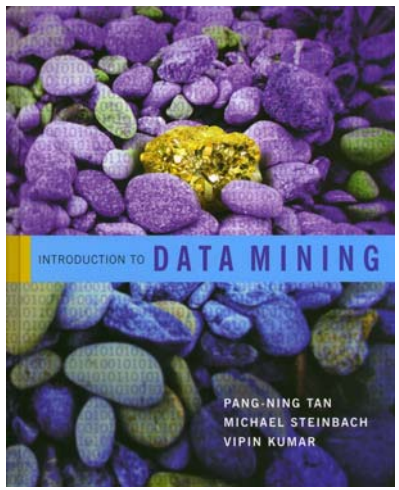
Use R to fit the AdaBoost classifier to the last column of the sonar training data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_train.csv
Plot the misclassification error for the training data and the test data at
http://www-stat.wharton.upenn.edu/~dmease/sonar_test.csv
as a function of the iterations. Run the algorithm for 500 iterations. Use default rpart() as the base learner.

Solution (continued):



Introduction to Data Mining

by
Tan, Steinbach, Kumar



Chapter 8: Cluster Analysis

What is Cluster Analysis?

- **"Cluster analysis divides data into groups (clusters) that are meaningful, useful, or both" (page 487)**

- **It is similar to classification, only now we don't know the "answer" (we don't have the labels)**

- **For this reason, clustering is often called *unsupervised learning* while classification is often called *supervised learning* (page 491 – but the book says "classification" instead of "learning")**

- **Note that there also exists *semi-supervised learning* which is a combination of both and is a hot research area right now**

What is Cluster Analysis?

- **Because there is no right answer, your book characterizes clustering as an exercise in descriptive statistics rather than prediction**

- **“Cluster analysis groups data objects based only on information found in the data that describes the objects and their similarities” (page 490)**

- **“The goal is that objects within a group be similar (or related) to one another and different from (or unrelated to) the objects in other groups” (page 490)**

Examples of Clustering (P. 488)

- **Biology: kingdom, phylum, class, order, family, genus, and species**
- **Information Retrieval: search engine query = movie, clusters = reviews, trailers, stars, theaters**
- **Climate: Clusters = regions of similar climate**
- **Psychology and Medicine: patterns in spatial or temporal distribution of a disease**
- **Business: Segment customers into groups for marketing activities**

Two Reasons for Clustering (P. 488)

- **Clustering for Understanding**

  (see examples from previous slide)

- **Clustering for Utility**

  -Summarizing: different algorithms can run faster on a data set summarized by clustering

  -Compression: storing cluster information is more efficient that storing the entire data - example: quantization

  -Finding Nearest Neighbors

How Many Clusters is Tricky/Subjective

How many clusters?

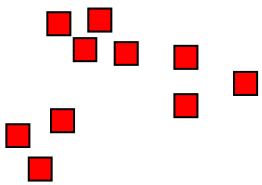
How Many Clusters is Tricky/Subjective

How many clusters?

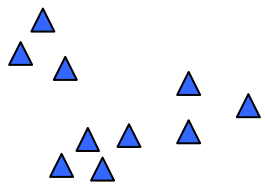
Two Clusters

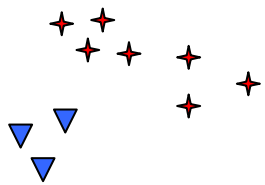
How Many Clusters is Tricky/Subjective

How many clusters?

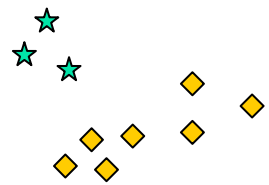


Two Clusters





Four Clusters



How Many Clusters is Tricky/Subjective

How many clusters?

Six Clusters

Two Clusters

Four Clusters

K-Means Clustering

- **K-means clustering is one of the most common/popular techniques**

- **Each cluster is associated with a centroid (center point) – this is often the mean – it is the cluster *prototype***

- **Each point is assigned to the cluster with the closest centroid**

- **The number of clusters, K, must be specified ahead of time**

K-Means Clustering

- **The most common version of k-means minimizes the sum of the squared distances of each point from its cluster center (page 500)**

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(c_i, x)$$

- **For a given set of cluster centers, (obviously) each point should be matched to the nearest center**

- **For a given cluster, the best center is the mean**

- **The basic algorithm is to iterate over these two relationships**

K-Means Clustering Algorithms

- **This is Algorithm 8.1 on page 497 of your text**

Algorithm 1 Basic K-means Algorithm.

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

- **Other algorithms also exist**

- **In R, the function kmeans() does k means clustering – no special package or library is needed**

In class exercise #49:

Use kmeans() in R with all the default values to find the k=2 solution for the 2-dimensional data at
http://www-stat.wharton.upenn.edu/~dmease/cluster.csv
Plot the data. Also plot the fitted cluster centers using a different color. Finally, use the knn() function to assign the cluster membership for the points to the nearest cluster center. Color the points according to their cluster membership.

In class exercise #49:

Use kmeans() in R with all the default values to find the k=2 solution for the 2-dimensional data at http://www-stat.wharton.upenn.edu/~dmease/cluster.csv
Plot the data. Also plot the fitted cluster centers using a different color. Finally, use the knn() function to assign the cluster membership for the points to the nearest cluster center. Color the points according to their cluster membership.

Solution (continued):

```
x<-read.csv("cluster.csv",header=F)

plot(x,pch=19,xlab=expression(x[1]),
                ylab=expression(x[2]))
fit<-kmeans(x, 2)

points(fit$centers,pch=19,col="blue",cex=2)
```

In class exercise #49:

Use kmeans() in R with all the default values to find the k=2 solution for the 2-dimensional data at http://www-stat.wharton.upenn.edu/~dmease/cluster.csv
Plot the data. Also plot the fitted cluster centers using a different color. Finally, use the knn() function to assign the cluster membership for the points to the nearest cluster center. Color the points according to their cluster membership.

Solution (continued):

```
library(class)

knnfit<-knn(fit$centers,x,as.factor(c(-1,1)))

points(x,col=1+1*as.numeric(knnfit),pch=19)
```