Solution Sketches

Midterm2 Exam

COSC 4335 *Data Mining*

April 7, 2015

Your Name:

Your student id:

Problem 1 --- Computing GINI in R [8]

Problem 2 --- R-Code to Analyze Clustering Results [5]

Problem 3 --- Tree Models and Classification in General [14]

Problem 4 --- K-NN, SVM and Ensembles [17]

Problem 5 --- R-Programming with Data Frames [11]

:

**Grade:**



The exam is “open books” and use of computers (but not e-mail) is allowed and you have 75 minutes to complete the exam. The exam will count approx. 14-19% towards the course grade.

1. **Computing GINI in R [8]**

Write a function *gini* in R, whose input is a vector of class proportions of arbitrary length[[1]](#footnote-1) called v (v contains O and positive numbers whose sum is exactly one) and returns the gini coefficient for v; e.g.

v<-c(0.5, 0.25, 0.25)

gini(v)

*would return:* 1- (0.5)\*\*2-(0.25)\*\*2-(0.25)\*= 1-1/4-1/8=5/8

gini <- function(v) {

  tempSum<- 0;

// use length function to determine positive number v contains

       for( int i in **1**:length(v)){

       tempSum<-  tempSum + v[i]\*v[i];

     }

// return gini co-efficient for v

       return (1-tempSum);

}

2) R-Code to Summarize Clustering Results [5]

Write some R-code that applies K-means for k=4 to the iris-flower dataset and then provides a summary, how many setosas, virginicas, and versicolor are in clusters 1, 2, 3, and 4, respectively (in the obtained clustering result)!

**z<-kmeans(iris[1:4],4)**

**z**

**#analyze clusters**

**t<-table(iris[,5], z$cluster)**

**t**

**One error, even a minor one: 0-2.5 points…**

**3) Tree Models and Classification in General [14]**

1. Compute the information-gain[[2]](#footnote-2) for the following decision tree split[5] (compute the exact value; just giving the formula will only obtain partial credit)

(2,2,2) (0,1,1)

(2, 1, 1)

Hbefore= **(1/3 \* (log3) )\*3**) = log 3 = 1.58 [log base 2 is considered]

Hafter= 1.58 - [ 2/6 Entropy\_Upper+ 4/6 Entropy\_Lower ] = 1.333

Entropy\_Upper= 0+ (1/2 log2) \*2**=1**

Entropy\_Lower = 1/2 log 2 + (1/4 log 4)\*2= **1.5**

**Hgain=Hbefore-Hafter = 1.58 -1.33 = 0.252**

b) Are decision trees capable to learns disjunctive concepts; for example.

IF A>3 OR B>7 THEN class1 ELSE class2? Give a reason for your answer! [2]

Yes. Reason: Depict the decision tree with tests A>3 and B>7 that computes the above function

1. What are the characteristics of overfitting? What can be done to deal with overfitting? [3]

Model is too complex, training error is low, but testing error is not optimal; …pruning…

1. How do regression trees predict the value of an example? What are their most significant differences from decision trees? [4]

1. Regression trees are used for prediction and not classification problems; consequently, regression tree leafs contain numbers and not class labels.

2. Test are chosen based on variance/MSE and not based on entropy/GINI/purity

3. Numbers associated with leaf nodes are chosen using averaging

**4) kNN, SVM, & Ensembles [17]**

a) The soft margin support vector machine solves the following optimization problem:



What does the second term minimize (be precise)? How many examples have non-zero ξi in the figure below! What purpose/role does C play? [3]



It minimizes the sum of the errors (which is 0, if an example is on the “correct” side of it’s class’ hyperplane or the distance to the hyperplane, otherwise. [1.5] Two[0.5]. C determines the importance of minimizing the error in relationship to maximizing the width of the margin.

b) SVMs are frequently successfully used in conjunction with kernels, such as polynomial and other kernels. What is the idea of the approach? Why do you believe it is successful? [4]

Examples are mapped into higher dimensional spaces, leading to more opportunities—as there are more attributes there are many more hyperplans that might successfully separate the examples belonging to two classes—to successfully split the examples and/or leading to solutions with wider magins/lower errors in the case that the soft margin SVM is used.

c) k-NN employ a lazy approach to learning models from training examples. What does this mean? What disadvantages you see with k-NN’s lazy learning approach? [3+1 extra point}

No model is learnt and consequently no training is performed; the data themselves are the model. [1.5]; slow; major computations have to be performed at runtime when determining the class label of an example. [1.5]

d) What can be said about the number and shape of decision boundaries of a k-NN classifier? [3]

The shape consist of lines formed of edges of convex polygons of the Vonoroi tessellation[1]; in general, decision boundaries can be non-convex and can take many forms [1]; there can be many decision boundaries[[3]](#footnote-3).

Problem 4 continued

e) What are the characteristics of a “good” ensemble of base classifiers in ensemble learning? [2]

The classifiers are diverse and make different kind of errors, and have an “okay” but not necessarily high accuracy (above 50%)

1. Why does AdaBoost increase the weights of misclassified examples? [2]

To enhance the chance that they are classified correctly next time, leading to a classifier that classifies those examples correctly and which is therefore different from the previously learnt classifier, leading to a more diverse ensemble.

5) R-Programming with Data Frames [11]

A dataframe df is given which only contains numerical data. Write an R function *myfilter* to filter the dataframe: *myfilter(df, th)*, as follows[[4]](#footnote-4): If any value in a row is greater or equal to the threshold *th*, then keep the row; otherwise this row is removed. Explain the code you wrote briefly!

Hint: a grouping function can be applied on vector/list: any(), all(); if you are not familiar with this function, write a nested loop that checks for the removal condition and goes through the rows sequentially instead!

myfilter <- function(df, threshold) {

 df$flag <- rep(FALSE, nrow(df))

 for (i in 1: nrow(df)) {

 if (any(df[i,] >= threshold))

 df$flag[i] <- TRUE

 }

 ret <- subset(df, flag)

 ret$flag <- NULL

 return (ret)

}

myfilter\_loop <- function(df, th) {

 df$flag <- rep(FALSE, nrow(df))

 for (i in 1: nrow(df)) {

 isFiltered = TRUE

 for (v in df[i,]) {

 if (v > th) {

 isFiltered <- FALSE

 break

 }

 }

 df$flag[i] <- isFiltered

 }

 ret <- subset(df, !flag)

 ret$flag <- NULL

 return (ret)}

1. You can use the length function to determine how many numbers v contains. [↑](#footnote-ref-1)
2. Entropy before (computed using the H-function) and after the split; moreover, log2(3)=1.58 and log2(6)=2.58 [↑](#footnote-ref-2)
3. Its number is only bounded by the number of examples in the training set. [↑](#footnote-ref-3)
4. If my-filter(df, 3) is called and df has 3 rows (3, 3, 3), (3,2,1), (2, 2, 1) then a modified dataframe df will be returned with the third row (2,2,1) removed. [↑](#footnote-ref-4)