DATA MINING

TEST 2

INSTRUCTIONS:

- this test consists of 4 questions
- you may attempt all questions.
- maximum marks = 75

Question 1 (15 marks)

Learning algorithms can be classified as *supervised* or *unsupervised*.

a) Describe the difference between *supervised* and *unsupervised* learning. [2]

b) Discuss the use of training and testing subsets for supervised learning. [3]

c) Contrast *decision tree learning* with *hierarchical clustering* outlining similarities and differences. [10]

Marking schedule

a) In both cases we try to predict the value of one attribute in a dataset observation using information from the other attributes in the observation. In supervised learning, the attribute we are trying to predict is used to train the learning algorithm whist in unsupervised learning the target attribute is hidden from us.

b) In supervised learning, to prevent over-learning, the dataset is split into training and testing subsets. (split proportions vary) the algorithm is trained on one subset and evaluated on the other.

c)

decision tree learning	*hierarchical clustering*
supervised	unsupervised
top down	bottom up
entropy based algorithm	distance measure algorithm
decision point selected via entropy consideration	clusters formed via merging nearest neighbours
final representation is a tree	final representation is a tree

Question 2 (25 marks)

Consider a training set of n observations, $\{\vec{x}_i, y_i\}_{i=1\ldots n}$, where the $\vec{x}_i$ are 2 dimensional feature vectors and $y_i \in \{-1, +1\}$ is the target attribute that places each observation in one of two possible classes.

In linear classification, we seek to divide the two classes by a linear separator in the feature space.

a) Explain with the aid of a diagram, the role played by the parameters b and $\vec{w}$ for a linear separator. [5]

b) Prove that the decision boundary generated by $(\vec{w}, b)$ is identical to the decision boundary generated by $(s\vec{w}, sb)$ for any scaling parameter s. [3]

c) Assuming the observations are linearly separable, outline the *perceptron algorithm* for finding a classifier. [6]

d) Consider the following *weak* version of Novikoff's convergence theorem and explain the **three** steps in the proof marked by the † symbol. [6]

Assumptions: Let $R = \max ||\vec{x_i}||$ and suppose that the learning task is solvable via a separator that passes through the origin. i.e. there exists some vector $\vec{w^*}$ of unit length and some $\delta > 0$ such that $y_i(\vec{w^*} \cdot \vec{x_i}) > \delta$ for all i.

Theorem: Under these assumptions, the perceptron algorithm converges after at most $(\frac{R}{\delta})^2$ updates.

Proof: Let $\vec{w}_n$ be the $\vec{w}$ vector after n updates and let $\vec{w}_0 = 0$. We will argue that whenever $\vec{w}$ is updated it becomes closer to $\vec{w}^*$. Suppose $\vec{w}_{n+1}$ is an update, i.e. $\vec{w}_n$ fails to classify an $\vec{x}$ correctly and hence $\vec{w}_{n+1} = \vec{w}_n + y\vec{x}$. Consider:

$$\begin{aligned} \vec{w}_{n+1} \cdot \vec{w}^* &= (\vec{w}_n + y\vec{x}) \cdot \vec{w}^* \\ &= \vec{w}_n \cdot \vec{w}^* + y\vec{x} \cdot \vec{w}^* \\ &\geq \vec{w}_n \cdot \vec{w}^* + \delta \qquad \dagger 1 \end{aligned}$$

This tells us that the projection of $\vec{w}_{n+1}$ onto $\vec{w}^*$ has increased. We would like this to mean that $\vec{w}_{n+1}$ is closer to $\vec{w}^*$. However, what it really means is that **either** $\vec{w}_{n+1}$ is closer to $\vec{w}^*$ **or** $\vec{w}_{n+1}$ has simply grown larger.

Now, consider the Euclidean length of $\vec{w}_{n+1}$:

$$\begin{aligned} ||\vec{w}_{n+1}||^2 &= ||\vec{w}_n + y\vec{x}||^2 \\ &= ||\vec{w}_n||^2 + 2y(\vec{w}_n \cdot \vec{x}) + ||\vec{x}||^2 \\ &\leq ||\vec{w}_n||^2 + R^2 \qquad \dagger 2 \end{aligned}$$

Thus, after N actual updates we know two facts: $||\vec{w}_N||^2 \leq NR^2$ and $\vec{w}_N \cdot \vec{w}^* \geq N\delta$. Putting these together:

$$N\delta \leq \vec{w}_N \cdot \vec{w}^* \leq ||\vec{w}_N|| \leq R\sqrt{N} \qquad \dagger 3$$

and so

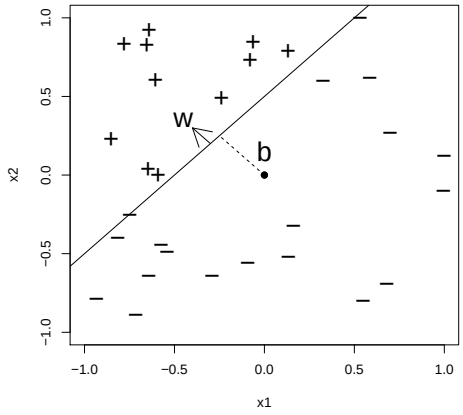
$$\sqrt{N} \leq \frac{R}{\delta}$$

which means N is bounded and updates must cease eventually.

e) Give a numerical example of a separable training set that leads to many iterations of the perceptron learning algorithm before a separator is discovered. [5]

Marking schedule

a) In the diagram feature vectors are two-dimensional. We wish to find a line that separates the positive observations form the negative ones. Any line can be specified by a unit normal vector, $\vec{w}$, and a signed perpendicular distance from the origin, b. Classification of a point $\vec{x}$ is then accomplished according to the value of $sign(\vec{x} \cdot \vec{w} - b)$. In the diagram above, a particular $\vec{w}$ and b is given that accomplishes the separation. The perceptron algorithm will discover such a separator. Note that in the above diagram b is negative with respect



to the direction of $\vec{w}$.

b) Suppose an observation, $\vec{x}$, is classified positive (or negative) by the classifier, $sign(b + \vec{x} \cdot \vec{w})$. Then the same observation will be classified positive (or negative) by the classifier, $sign(sb + \vec{x} \cdot s\vec{w})$ provided s is positive, because $sign(sb + \vec{x} \cdot s\vec{w}) = sign(s(b + \vec{x} \cdot \vec{w})) = sign(b + \vec{x} \cdot \vec{w})$. If s is negative then the classifications of all observations will be swapped yielding the same classes.

c)

```
euclidean.norm = function(x) {sqrt(sum(x * x))}

perceptron = function(x, y, learning.rate=1) {
  w = vector(length = ncol(x)) # initialize w
  b = 0 # Initialize b
  k = 0 # count updates
  R = max(apply(x, 1, euclidean.norm))
  made.mistake = TRUE # to enter the while loop
  while (made.mistake) {
    made.mistake=FALSE # hopefully
    yc <- classify.linear(x,w,b)
    for (i in 1:nrow(x)) {
      if (y[i] != yc[i]) {
        w <- w + learning.rate * y[i]*x[i,]
        b <- b + learning.rate * y[i]*R^2
        k <- k+1
        made.mistake=TRUE
      }
    } }
  s = euclidean.norm(w)
  return(list(w=w/s,b=b/s,updates=k))
}
```

d) 1) as $\vec{w}^*$ is a separator we know that $y(\vec{x} \cdot \vec{w}^*)$ is strictly greater than zero, in fact greater than δ by the assumptions listed in the theorem statement.

2) as $\vec{w}_n$ failed to classify $\vec{x}$ we know that $y(\vec{x} \cdot \vec{w}_n) < 0$ and we are also given that $||\vec{x}||^2 < R^2$ in the assumptions to the theorem.

3) The first inequality can be obtained by applying †1 N times.
The second inequality is due to the fact that $\vec{w}^*$ is a unit vector.
The final inequality is obtained by applying †3 N times and then taking square roots.

e) Just use any separable sets with most point separated by large margin but one from each set separated by a tiny margin.

Question 3 (20 marks)

Consider a *non-separable* training set of n observations, $\{\vec{x}_i, y_i\}_{i=1...n}$.

a) Outline the so-called *pocket* algorithm that allows the perceptron to find the *best* possible linear separator for the training set. [5]

b) What is *Mahalonobis* separation and what role does it play in the construction of a linear classifier when the two classes are **not** linearly separable.? [10]

c) In R, the best possible linear separation is accomplished using the `lda` function from the `lda` package. Once a linear discriminant has been trained it can be passed to R's `predict` function to classify new data points. Although the `lda` function does not return an explicit formula for the class boundaries we can *trick* R into revealing them. Explain how you would do this. [5]

Marking schedule

a) Each time you generate a separator, count how many it gets right and if it beats the current separator in your pocket then empty your pocket and place the new separator in it.

b) Once the optimal separation direction $\vec{w}$ is found, we must project the feature vectors onto $\vec{w}$ and then using the *Mahalonobis* bias b to separate the two classes:

$$b = \frac{\mu_+\sigma_- + \mu_-\sigma_+}{\sigma_+ + \sigma_-}$$

where the various μ and σ are the means and standard deviations of the two classes.

Assuming that the distribution of the projection onto $\vec{w}$ is double humped in the form of two overlapping normal distributions, this Mahalonobis bias will yield the point of overlap which provides the best offset for the separation plane.

c) What we do is generate a large testing set with data uniformly distributed in each of the variables in the original training set and then we predict the class of each observation in the new set and then we plot the observations in 2D using the best two linear discriminants and coloring the observations according to class. The 2D linear separation lines will be revealed in the plot.

Question 4 (15 marks)

The R package, `tm`, provides a suite of routines for undertaking text mining tasks.

a) In the `tm` package text for data mining is commonly stored in a *corpus*. Outline 5 procedures that are commonly used for cleaning a corpus. [5]

b) What is a *term-document matrix* and what is it used for? [4]

c) Outline two real life examples, one where you might want to cluster terms from a corpus and the other where you might want to cluster documents from a corpus? [6]

Marking schedule

a) Via the `tm_map` function, the tm package provides common comon cleaning operations:

1) `tolower`, transform all characters to lower case.
2) `stripWhitespace`, remove white space
3) `removePunctuation`, get rid of punctuation characters
4) `removeWords, myStopwords`, get rid of common words that may distort later analysis
5) `stemDocument, language = "english")`, replace similar words by their stem.

b) A term-document matrix represents the relationship between terms (or words) and documents, where each row stands for a term and each column for a document, and an entry is the number of occurrences of the term in the document. Correlations, associations, clustering and classification is usually carried out on the term document matrix and not the original corpus.

c) 1) Consider a collection of letters written over a long period, Clusters of words might indicate different topics the author covers.

2) Consider each work of Shakespeare as a corpus document and each work of Marlow as a corpus document. Combine the two corpora into one corpus and cluster the documents. Hopefully two clusters appear and any disputed work can be attributed to the author of the nearest cluster.