INFS4203 2012 Theory

**Data mining** is the extraction of interesting (i.e. non-trivial, previously unknown, and potentially useful) patterns or knowledge hidden in large amounts of data. It is needed as the process needs to be automated due to the enormous size of today’s datasets.

# Classification Algorithms

Given a collection of records (training set), find a model for a class attribute as a function of the values of other attributes. The goal is to accurately assign previously unseen records to a class.

Two types: lazy and eager. **Eager** learners generate the model explicitly (decision tree), **lazy** do not (kNN).

## A Two-Step Process

Model construction: describes a set of predetermined classes

Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute

The set of tuples used for model construction is a **training set**

The model is represented as:

* classification rules,
* decision trees,
* mathematical formulae, or

Model usage: for classifying future or unknown objects

Estimate **accuracy** of the model

* The known labels of test sample is compared against the classified result from the model
* Accuracy rate is the percentage of **testing set** samples that are correctly classified by the model

If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

## Performance Evaluation

**Confusion matrix:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Prediction** |  |
|  |  | **Positive** | **Negative** |
| **Actual** | **Positive** | A | B |
|  | **Negative** | C | D |

A: True positive

B: False negative

C: False positive

D: True Negative

**Accuracy** = (TP + TN) / (TP + TN + FP + FN)

**Precision** (“exactness”) = TP / (TP + FP)

**Recall** (“completeness”) = TP / (TP + FN)

To evaluate models, can use holdout or cross validation.

**Holdout**

Randomly take 70% of the examples as training and the remaining 30% as testing

Repeat for several times (e.g. 10)

used for data set with large number of samples

**Cross validation**

Randomly partition the data into k mutually exclusive subsets (D1,D2, .., Dk), each approximately equal size

At i-th iteration, use Di as test set and others as training set

used for data set with moderate size

## Nearest Neighbour Classifiers

“If it walks like a duck, quacks like a duck, then it’s *probably* a duck.”

Requires three things:

1. The set of stored records
2. A distance metric to compute distance between records
3. The value of k, the number of nearest neighbours to retrieve

To classify an unknown record:

1. Compute distance to other training records (using the given distance metric)
2. Identify k nearest neighbours
3. Use class labels of nearest neighbours to determine the class label of unknown record (e.g. by taking majority vote or by weighing the vote according to distance, i.e. w = 1/d^2)

If k is *too small*, it is sensitive to noise. If it is *too large*, it could include points from other classes.

**K-nearest neighbours** of a record *x* are data points that have the k smallest distance to x.

kNN is robust to noisy data as it can average the k-nearest neighbours. It does, however, suffer from the “curse of dimensionality”. This is where the distance between neighbours could be dominated by irrelevant attributes. It can be overcome by eliminating the least relevant attributes.

**Similarity**

* Numerical measure of how alike two data objects are
* Is higher when objects are more alike
* Often falls in the range [0,1]

**Dissimilarity**

* Numerical measure of how different two data objects are
* Lower when objects are more alike
* Minimum dissimilarity is often 0

**Scaling issues**

Attributes may have to be scaled:

to prevent distance measures from being dominated by one of the attributes

## Bayes Classifier

The **Bayes classifier** is a probabilistic model for the classification problem.

Bayes’ Theorem allows conditional probabilities to be flipped, i.e. to find P(spam | keywords) we can use P(keywords | spam).

If we assume independence, P(A1, A2, …, An) | C) = P(A1 | C) P(A2 | C) … P(An | C)

Thus, given a bunch of attributes, calculate P(A | C) for each one. Do this for all possible classes. Choose the class that maximises P(A1, A2, …, An) | C) P(C)

Naive Bayesian prediction requires each conditional probability be non-zero, otherwise the predicted probability is zero. To correct this we could add one to each case (**Laplacian correction**).

Advantages of this classifier are that it is easy to implement and obtains good results in most cases. Some disadvantages are that the independence assumption does not always hold - in practical situations, dependence exists among variables.

## Decision Tree

To generate tree,split the records based on an **attribute test** that optimizes certain criterion. Problems are how to split the records and how to decide when to stop splitting.

Splitting attributes depends on the type (nominal, ordinal, continuous) and can be split multiple ways (2-way, or multi-way).

Advantages of decision trees is that they are inexpensive to construct, fast, easy to interpret for small sizes, and the accuracy is comparable to other classification techniques.

**Impurity Measures**

Can use **GINI index** to decide which attribute gives the best split.

**GINI(t)** = 1 - sum(P(j | t)^2 for j=1 to number of classes at a node t)

e.g. = 1 - (1/4)^2 - (3/4)^2

Then, can determine how good a split is by weighting each of the partitions in a split:

**GINI(split)** = sum(ni/n GINI(i)) where ni is the number of records at child i and n is the number of records at parent node p

Can also use classification error at a node:

**Error(t)** = 1 - max P(i|t)

e.g. = 1 - max(1/4, 3/4) = 1 - 3/4 = 1/4

GINI index gives a smoother curve than classification error (which is linear).

**Stop Expanding**

Stop expanding when all records belong to same class, or all records have similar attribute values. Sometimes need early termination to avoid overfitting. Thus could stop when the number of instances at a node are less than some value, or if expanding the current node does not improve the impurity measures.

**Overfitting** is where the model starts to reflect noise and outliers.

**Notes on Overfitting**

An induced tree may overfit the training data

* Too many branches (complex tree)

Some may reflect noise or outliers

* Poor accuracy for unseen samples

Training error no longer provides a good estimate

**How to Address Overfitting**

Stop the algorithm before creating fully-grown tree!

* Typical stopping conditions for a node:

1. Stop if all instances belong to the same class

2. Stop if all the attribute values are the same

* More restrictive conditions:

1. Stop if number of instances is less than some user-specified threshold

2. Stop if expanding the current node does not improve impurity measures (e.g., Gini).

**Advantages:**

* Inexpensive to construct
* Extremely fast at classifying unknown records
* Easy to interpret for small-sized trees
* Accuracy is comparable to other classification techniques for many simple data sets

# Clustering

**Clustering** simply clusters objects ­­­that are similar together. It is different to classification as it is unsupervised and does not need to know the actual classifications.

A good clustering has high intra-class similarity and low inter-class similarity.

Two types: partitional and hierarchical.

**Partitional clustering** is a division of data objects into *non-overlapping* subsets such that each data object is in exactly one subset.

**Hierarchical clustering** is a set of nested clusters organised as a hierarchical tree.

## K-means

Each cluster associated with a centroid, each point assigned to nearest centroid. Number of clusters (k) must be specified.

Algorithm: select k points as the initial centroids. Form k clusters by assigning points to nearest cluster, then recompute centroid. Repeat until centroids do not change.

Stopping condition is often “until relatively few points change clusters” rather than until no centroids change.

K-means converges pretty fast (usually in the first few iterations)< but this does not mean the clusters are optimal.

**Sum of Squared Error (SSE)** can be used to evaluate K-means clusters.

**SSE** = sum of SSE for each cluster, ie. sum of the (distance for each point to the centroid squared)

Choice in initial picks is important. To reduce error some post processing is often done: small clusters (outliers) are eliminated, loose clusters (high SSE) are split, and close cluster (low SSE) are merged.

K-means has problems when clusters are of different sizes, densities, or non-spherical shapes, and with outliers.

## Density-Based Clustering

**Eps** is the maximum radius of the neighbourhood, a neighbourhood with a radius of eps is defined as the eps-neighbourhood.

**MinPts** is the minimum number of points in an eps-neighbourhood of a point

**Neps(q)** is the set of points in the neighbourhood of point q

A point p is **directly density-reachable** from point q if it belongs to Neps(q) and q is a core point (Neps(q) >= MinPts).

A point p is **density-reachable** from a point q if there is a chain of points such that each one is directly density-reachable from the last.

A point p is **density-connected** to a point q if there is a point o such that both p and q are **density-reachable** from o.

**DBSCAN**: select a point p, if Neps(p) < MinPts, mark as noise. Otherwise, retrieve all points density-reachable from p and if p is a core point, a cluster if formed. If p is a border point, go to the next point.

## Hierarchical Clustering

Two types: agglomerative and divisive.

**Agglomerative clustering** starts with each point as an individual cluster and each step merges the closest pair of clusters until only one cluster is left.

**Divisive clustering** starts with one all-inclusive cluster and at each step splits a cluster until each cluster contains a point (or there are k clusters)>

These algorithms use a similarity of distance matrix, merging or splitting only one cluster at a time.

Different distance approaches: min (single linkage), max (complete linkage), group average, and distance between centroids.

Min is sensitive to noise and has the chaining phenomenon where clusters may be forced together due to a single element being close, even though many of the elements in each cluster are far apart.

Max tends to break large clusters.

Group average is a compromise between single and complete linkage.

# Distances

Many types of distance metrics: Manhattan (L1 norm), Euclidean (L2 norm), max of any dimension (Lmax norm). Also Jaccard (used for binary), cosine (difference in angles), and edit (number of operations needed to change one string into another)

**Jaccard** = number of terms in intersection of two sets / number of terms in their union

**Cosine** = dot product of sets / magnitude of each set multiplied together