Uppsala University
Department of Information Technology
Kjell Orsborn

DATA MINING I - 1DL360

Assignment 1 - Classification using kNN

1 Classification using a k-Nearest Neighbours Algorithm

This assignment is taken from the field of forensic science. The practical goal is to classify a number of glass samples (building window glass, vehicle window glass, vehicle headlight glass, etc) using a k-Nearest Neighbors (kNN) algorithm. The academic goal is to:

1. Understand how to analyze data and experiment with data in an interactive and iterative manner.
2. Understand data preprocessing and normalisation.
3. Understanding kNN classification model generation and analysis.
4. Understand tuning of kNN classification.

You will use, analyse and experiment with a kNN-algorithm that is implemented in the AmosMiner (AmoxMiner for Mac OS and Linux) application using the Amos II database management system. AmosMiner extends the Amosql query language of Amos II with data mining functionality. This means that the complete data analysis process can be carried out by interactively posing query expressions to a database supporting an incremental and iterative experimentation with the data sets. Thus, since AmosMiner is based on a DBMS and provides a query-language-based analysis, there is no need for writing, compiling and excecuting conventional programs. Furthermore, there is no need move data from where it is stored or to copy and transform data to specialised formats and analysis tools.

You will find further instructions on how to install Amos II and AmosMiner according to the instructions on the assignments home page. It can be used either in the computer labs at Uppsala University or on your own laptop or desktop computer (Windows, Mac OS and Linux supported).

The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence – if it is correctly identified. The data in this assignment is taken from [ES87], from UK Home Office Forensic Science Service, and describes 6 types of glass defined in terms of the their refractive index and their oxide content (i.e. Na, Mg, Al, Si, K, Ca, Ba and Fe). The training data set also include a class attribute, i.e. the known glass type for the samples in the training data set.

Attribute information:

1. RI: refractive index
2. Na: Sodium
3. Mg: Magnesium
4. Al: Aluminum
5. Si: Silicon
6. K: Potassium
7. Ca: Calcium
8. Ba: Barium
9. Fe: Iron
10. Type of glass: (class attribute)
    (a) building_windows_float_processed (1)
    (b) building_windows_non_float_processed (2)
    (c) vehicle_windows_float_processed (3)
    (d) headlamps (4)
    (e) containers (5)
    (f) tableware (6)

Attributes 2-9 are expressed as unit measurement, i.e. weight percent in corresponding oxide.

2 Preparation

You should prepare for this assignment by reading about the kNN-algorithm presented in Chapter 4 in Tan et al. [Tan06].

It is also advisible to read Chapters 2.3 and 2.4 that include sections on *sampling*, *normalization* and *proximity measures.*

You can find and download the AmosMiner and Amos II system from the assignments home page and you will also get a chance to familiarize yourself with those systems in the introductory tutorials. **We also strongly recommend you to run the tutorials for Amos II and AmosMiner before you start with the actual assignment.**

3 Assignment

You will find data for this assignment in the `data` directory under AmosMiner. In the data, there are no missing values and you may assume that there is no noise.

Data resides in the following files:

`glassdata.nt`: This file contain 163 data points of 10 dimensions. Each data point is one observation. Dimension 1. . . 9 corresponds to a certain measurement of a glass feature, i.e.: (refractive index, sodium concentration, magnesium concentration, etc.) for a data point. The 10th dimension is the known class of each data point represented as an integer between 1 and 6. This means that each record of values in the data file is a vector of values that consists of 10 values and have the following structure:

```
{RI, Na, Mg, Al, Si, K, Ca, Ba, Fe, class}
```

, where

`testdata.nt`: This file consists of 30 observations of unclassified glass samples, i.e. it includes values for the first 9 dimensions but the class value is unknown.

In the assignment, you should classify the 30 unclassified glass samples using the kNN algorithm in AmosMiner.

Under the section for Assignment 1 on the assignment course page, you will find the script file for the this assignment named 'a1.osql'. This file include predefined functions for importing data, preprocessing and normalization of data, tuning of kNN analysis parameters, and for the kNN analysis itself. You should explore the various phases of the kNN analysis by interactively investigate and adapt the predefined query language expressions found in the script file. This is done most easily by opening the script file in a text editor (e.g. Notepad in Windows and TextEdit in Mac OS), copy a query expression from the editor and paste it into the AmosMiner terminal window were it will be executed.

That is, you should do the following:

START-UP

Once you have started the AmosMiner system (follow the instructions on the assignments home page) and opened your `a1.osql` script file, you can start explore the various aspects of kNN analysis by the copy-and-paste approach described above.

You should first go through step 1 to step 4. In step 5 you should repeat steps 2 through 4 using different combinations of parameters and functions.

1. DATA DEFINITIONS

   (a) Definition of stored functions (data tables)

2. DATA PREPROCESSING

   (a) Defining parameters for importing data

(b) Import data

(c) Analyze and normalize data

Note that normalization of data can be an optional step. One alternative is to perform analysis using unnormalized data.

Question: Why would that be an alternative?

3. kNN MODEL GENERATION

(a) LEAVE-ONE-OUT cross validation

You are supposed to repeat this step in the script file with different values of k to find the best k.

Hint:

- It can be run automatically by implementing a loop.
- Amos II has a built-in function `iota(1, N)` which returns all incremental values from `1` to `N`.
- You should store value of k in the Amos II variable `:k`.

Questions:

- What is the best k?
- Will normalizing the data improve the classifier result?
- Can you explain how Leave-One-Out cross validation work?
- What can be the advantage or disadvantage with this approach?
- Can you think of an alternative method for partitioning your data into training data and test data?

4. kNN CLASSIFICATION

(a) CLASSIFY THE UNKNOWN DATA

Classify the unknown data with your current settings from previous steps (see Section 5 for additional alternatives).

The `classified_glass_data` stored function should contain all classified data.

5. kNN TUNING

   You should repeat step 2 through 4 and experimenting with different normalization methods, distance functions and attribute weighting (optional) and investigate how the kNN analysis is influenced.

   (a) Normalization:

   You should experiment with different normalization methods, such as Gaussian, MinMax and MaxNorm.

   Normalization of the dataset D in mapping each vector $\vec{x}$ to a normalized vector $norm(\vec{x})$, so that the norm of all normalized vectors falls into certain range, typically $[0, 1]$ or $[-1, 1]$. We are going to experiment with linear normalization functions, i.e. those that can be expressed as

$$norm(\vec{x}) = \left\{ \frac{x_i - subtract_i}{divide_i} \right\}_{i=1}^{d}$$

   where d is the number of dimensions, and components of vector parameters *subtract* and *divide* are computed as aggregate values of the dataset in respective dimensions.

   Try, for example, the following normalization methods

	subtract	divide
Gaussian	$mean_{x \in D}(x_i)$	$stdev_{x \in D}(x_i)$
Minmax	$min_{x \in D}(x_i)$	$max_{x \in D}(x_i) - min_{x \in D}(x_i)$
Maxnorm	0	$max_{x \in D}(\lvert x_i \rvert)$

   In all classification and clustering algorithms used here, we apply a metric (distance) function to determine how different two data points are from each other. A distance function expressed as a function of power r is computed as

$$dist_r(\vec{x}, \vec{y}) = \sqrt[r]{\sum_{i=1}^{d} |x_i - y_i|^r}$$

Different normalization methods have different effects on the distance metric $dist_r$, assigning certain weights to the dimensions:

$$dist_r(norm(\vec{x}), norm(\vec{y})) = \sqrt[r]{\sum_{i=1}^{d} \left| \frac{x_i - subtract_i}{divide_i} - \frac{y_i - subtract_i}{divide_i} \right|^r}$$

Simplification results in:

$$dist_r(norm(\vec{x}), norm(\vec{y})) = \sqrt[r]{\sum_{i=1}^{d} |divide_i|^{-r} \cdot |x_i - y_i|^r}$$

Hint: It is useful to use the built-in aggregate functions: mean, standard deviation and maximum, minimum or maxnorm number in a bag of numbers respectively (see Amos II manual).

```
avg(Bag of Number x) -> Real
stdev(Bag of Number x) -> Real
maxagg(Bag of Number x -> Number
minagg(Bag of Number x -> Number
maxnormagg(Bag of Number x) -> Number
```

Questions:

- Which normalization gives the best classifier accuracy (a number of points which have been classified correctly)?
- Why?

(b) Distance function:

Also try different distance functions such as Manhattan, Euclidian and MaxNorm as defined an AmosMiner as:

```
manhattan(Vector of Number x,
          Vector of Number y) -> Number
euclid(Vector of Number x,
          Vector of Number y) -> Number
maxnorm(Vector of Number x,
          Vector of Number y) -> Number
```

Questions:

- Which distance measure gives the best classifier accuracy?
- Why?

(c) Attribute weights (optional):

In kNN, all attributes of an object equally contribute to the distance between the object and another.

However, in reality, some attributes might be less important than others in order to measure the difference (distance) between two objects. That means, that the less important attributes should have less influence on the distance function compared to important ones.

This problem can be treated by weighting the contribution of each attribute so that some attributes will have a higher weight than others, when calculating the distance.

Hint:

- Defining a vector weight:
  ```
  create function weights() -> vector of number;
  ```
- Assigning your own weights:
  ```
  set weights() = {1,1,1,1,1,1,1,1,1};
  ```
- Creating your own weighted_euclid function:
  ```
  create function weighted_euclid(Vector of Number x,
                                  Vector of Number y)
     -> Number as
        sqrt(sum(select weights()[i] *
                        power(x[i]-y[i], 2)
                 from Integer i));
  ```
- Passing your own `#'weighted_euclid'` function instead of the built-in one `#'euclid'`
- You can experiment with some weighted form of the previous distance functions.

Questions:

- If all the values of a dimension are the same, how valuable is that dimension as a classifier?
- Discuss what weights give the best classification?

You can also find these instructions and hints in the TODO list of the `a1.osql` script file.

You are also referred to the Amos II manual for further reading, which is available through the lab course home page. We suggest that you read section 2.6 about Collections. You might find the following sections especially useful: `avg`, `stdev`, and `count` in 2.6.1, `groupby` in 2.6.2, and `aggv`.

4 Examination

At the examination, you should be prepared to show your implementation and to execute it in AmosMiner by executing the `a1.osql` script file.

Be prepared to answer questions regarding topics such as:

- A brief description of your model tuning decisions in the kNN implementation. For example, how the data was preprocessed when it comes to normalization, distance messure, etc.
- The reasoning behind your choice of k.
- If it is known a priori (beforehand) that each feature has different importance, how can that knowledge be used in the pre-processing to improve the classication performance?
- How did you choose your distance measure? Why did you do it that way?
- What would be the complexity of the current algorithm for chosing k? What is the complexity for the algorithm to classify one data point?

Acknowledgements

Contributions to the material for this assignment has been supplied by several of the former and current colleagues at the department, in alphabetical order, including Andrej Andrejev, Per Gustavsson, Tobias Lindahl, Kjell Orsborn, Tore Risch, Thanh Troung and Erik Zeitler.

References

[ES87] Ian W. Evett and Ernest J. Spiehler. Rule Induction in Forensic Science. Central Research Establishment. Home Office Forensic Science Service. Aldermaston, Reading, Berkshire RG7 4PN.

[Tan06] Tan, P-N, Steinbach, M. and Kumar, V.: Introduction to Data Mining, Addison-Wesley, 2006.