Introduction

This assignment is written by Christian Jensen (chrj@itu.dk)

The dataset used is:”Data_Mining_Student_DataSet_Spring_2013_Fixed.csv”.

The program structure is: Program holds 3 separate main methods, one for each algorithm. Analysis contains the actual calls to the algorithm, as well as code for printing. APriori, KMeans, kNN contains the algorithms and Field and CleanDataPoint contains the data used.

Questions:

A. What correlations exist between favored Programming languages, Operating systems and SQL servers?
B. Is it predictable whether people know about both Neural Networks and Support Vector Machines based on their age, skill in programming, years of university study and English skill?
C. Is it possible to identify particular groups of students, based on their age, skill in programming, years of university study and English skill?

Data Pre-Processing

Normalization

Min-max normalization is used twice in the final report, in the Analysis class, as part of preprocessing for both K-means and k-NN.

Missing Value Replacement

Missing Value Replacement is used primarily in initialization of CleanDataPoint. One example is that one person had not filled out an age, but had filled out a birthdate. Based on when the questionnaire was filled out, a correct birthdate could be calculated. Most other cases was solved with an Unknown variable, such as programming language.

Algorithms

A Priori

The A Priori algorithm is designed to identify correlations between sets of items.

This happens by simply looking at the number of occurrences of these items, which is turned into association rules, in the form A => B, read as in case that A exists in a given set, B will also likely occur. To identify the most significant of these association rules, the measure of Support (how many data points of the total includes this correlation) and Confidence (how many percent of sets with A also includes B).

The implementation allows for customizing minimum support and confidence, the values are initially set to 15% support and 70% confidence. The values are set this way because of the relatively small dataset and uneven distribution of data (combined, it means that you risk losing otherwise interesting correlations because the sample of these are too small)

The question answered with A Priori is question A: What correlations exist between favored Programming languages, Operating systems and SQL servers?

The answer can be found by running the analysis method RunAPriori. While absolute certainties are not present, a number of trends emerge:

- Strong correlations between Java, CSharp, MS-SQL and Windows (referred to as the Big Four). These connections make up most of the association rules. It is interesting to note that while MS-SQL is strongly correlated to Windows (100% confidence), Windows is not strongly correlated to MS-SQL. None of the correlations found does not include one of the Big Four, which also underline their dominance in the answers given.
- The only correlation including OSX is that OSX is strongly correlated to Java, which is understandable, given Javas support of all major OS platforms.
- MySQL are only seen in combinations of the Big Four, replacing MS-SQL, but with relatively low confidence margins
- C is correlated to Windows and C++ is correlated to both Windows and Java while FSharp is stronger correlated with CSharp

K-Means

k-Means is a clustering algorithm based around iterative improvements. Each cycle, all sample data is assigned to the nearest cluster (defined by their centers), and then these clusters are re-calculated to find the new center, based on the included data points.

The question answered with K-Means is Question C: Is it possible to identify particular groups of students, based on their age, skill in programming, years of university study and English skill?

The results are found in the Analysis section method RunKMeans.

The identified groups seems be solidly separated (here named humorously): “Decent students”, “Old-timers”, “Other Decent Students”, “Not that good at English” and “Over-achievers”.

The main problem here is that it is not really possible to properly evaluate the result, because that the data on each person is limited, but it would be very likely that these clusters would correspond to different groups in the class, such as exchange students, single courses takers, people with a family or work or bachelor or master students.

K-NN

K-Nearest-Neighbors is a deceptively simple classification method, in which a test tuple is evaluated against the Euclidian distance to find its K nearest neighbors and identify the majority class label of these neighbors.

The question answered with K-NN is question B: Is it predictable whether people know about both Neural Networks and Support Vector Machines based on their age, skill in programming, years of university study and English skill?

The results are found in the Analysis section method RunKnn.

The results seems interesting: 6 True Negatives, 1 True Positive and only one False Negative. However, it is worth noting that earlier tests suggests that the margin of error for Positive might be quite high.

The main problem is that the distribution of Knows to Knows Not. Since the classifier given to the tested data point is that of the *majority* of the nearest neighbors, any skew in this distribution can easily upset the balance in the favor of the majority element, in this case, Knows Not.

Implementing a system of weighting or defining a new balance point (which is the real distribution of classifiers) is a way to rectify this problem.