Data-Mining Coursework

Hugh Rawlinson
Reg No. 33219875
mu202hr@gold.ac.uk

2015/12/18

1 Introduction

This document contains my submission for the coursework to IS53023B Data Mining. It should be accompanied by a selection of supporting files, sorted by question number.

2 Question One

Please find source code solution enclosed in the zip file. Also available at github.com/hughrawlinson/data-mining.

3 Question Two

In this section, I will illustrate the whole process of the application of the Apriori algorithm to the provided dataset.

3.1 Setup

The first step of the Apriori algorithm is to calculate the number of rows required for a rule to be considered 'supported'. In this case, with 15 rows and a minimum support threshold of 25%, the threshold is calculated by $0.25 \times 15 = 3.75$. Therefore, a supported rule will have greater than 3.75 four instances

3.2 One-Frequent Item Set

After calculating the minimum support threshold number, one can proceed to find the *1-frequent item set*. This is achieved by counting the number of instances of each classification on each of the fields in a dataset. Please see the following frequent item set constructed from the provided data:

Sex=Male	8	candidate
Sex=Female	7	candidate
LifeInsuranceProm=Yes	9	candidate
LifeInsuranceProm=No	6	candidate
IncomeRange=20−30K	4	candidate
IncomeRange=30−40K	5	candidate
IncomeRange=40−50K	4	candidate
IncomeRange=50−60K	2	non−candidate

As you can see, there are more than 3.75 instances of each attribute/class pair except for where the *IncomeRange* attribute is equal to *50-60K*. Therefore, all rules but that are considered to be members of the frequent item set for this dataset. Now that we have calculated the one-frequent item set, we can use it to build our two-frequent item set

3.3 Two-Frequent Item Set

The two-frequent item set is constructed by joining two items from the one-frequent item set to form *candidates*. The following is a listing of the candidates I constructed from the one-frequent set, the counts of the number of instances fitting those rules, and my classifications as to whether or not those candidates were members of the two-frequent item set.

Sex=Male&LifeInsuranceProm=Yes	3	non−candidate
Sex=Female&LifeInsuranceProm=Yes	6	candidate
Sex=Male&LifeInsuranceProm=No	5	candidate
Sex=Female&LifeInsuranceProm=No	1	non−candidate
Sex=Male&IncomeRange=20−30K	2	non−candidate
Sex=Female&IncomeRange=20−30K	2	non−candidate
Sex=Male&IncomeRange=30−40K	3	non−candidate
Sex=Female&IncomeRange=30−40K	2	non−candidate
Sex=Male&IncomeRange=40−50K	3	non−candidate
Sex=Female&IncomeRange=40−50K	1	non−candidate
IncomeRange=20−30K&LifeInsuranceProm=Yes	2	non−candidate
IncomeRange=20−30K&LifeInsuranceProm=No	2	non−candidate
IncomeRange=30−40K&LifeInsuranceProm=Yes	4	candidate
IncomeRange=30−40K&LifeInsuranceProm=No	1	non−candidate
IncomeRange=40−50K&LifeInsuranceProm=Yes	1	non−candidate
IncomeRange=40−50K&LifeInsuranceProm=No	3	non−candidate

I determined that there were three supported candidates in this set: where sex is female, the customer was more likely to take part in the life insurance promotion; where sex is male, the customer was less likely to take part in the life insurance promotion; and where the income range is 30-40K, the customer is more likely to take part in the promotion. With these three rules, I can now proceed to calculate the candidacy of any potential three-frequent rules.

3.4 Three-frequent item set

Using the same method that I used to calculate the two-frequent item set, I joined the valid candidates from the previous step to find candidates for this item set. They are listed below:

Sex=Female&LifeInsuranceProm=Yes&IncomeRange=30−40K	2	non−candidate
Sex=Male&LifeInsuranceProm=No&IncomeRange=30−40K	1	non−candidate

Neither of the constructed rules had support high enough to reach the minimum support threshold; therefore using this support threshold on this dataset does not yield any L3 or greater rules.

3.5 Calculation of confidences

We are left with three rules left on which to calculate confidences. They are as follows:

Sex=Female&LifeInsuranceProm=Yes	6	candidate
Sex=Male&LifeInsuranceProm=No	5	candidate
IncomeRange=30−40K&LifeInsuranceProm=Yes	4	candidate

To calculate confidences, we take the number of rows for which all of the constituent rules are true, divided by the number of rows for which the last-column rule is true.

	|A,B|	|B|	P(A|B)
Sex=Female&LifeInsuranceProm=Yes	6	9	66.67%
Sex=Male&LifeInsuranceProm=No	5	6	83.32%
IncomeRange=30−40K&LifeInsuranceProm=Yes	4	9	44.45%

The only rule that passes our given threshold of 75% confidence is that if sex is male, the customer will not partake in the life insurance promotion.

3.6 Summary

In summary, the Apriori application to this data indicates that the life insurance promotion will be least successful among men, and should therefore likely be primarily marketed towards people who do not identify as male, who despite not passing the confidence threshold, have a greater than 65% chance of taking part in the promotion. Further techniques that could yield some more useful information include adding more data to the dataset (both in added dimensions and in a larger instance count), re-quantising existing data (such as income ranges), and adjusting the support and confidence thresholds.

4 Question Three

4.1 The K-Means Algorithm

The k-means algorithm can be described in five steps:

1. Choose a value for k, the number of clusters to be used in the algorithm
2. Generate k random points to be used as cluster-centres
3. Assign each point in the data-set to their nearest cluster centre
4. Move the cluster centre to the average position of all points assigned to it
5. Repeat steps 3-4 until the clusters converge

Convergence is defined as the members of each cluster not changing for one iteration (or epoch) of training.

4.2 K-Means in Weka

Figures 1 and 2 are screenshots of Weka's window showing the result of the clustering for each window. Both are available at full resolution in the accompanying materials distributed with this report. The quality of the two clusterings is approximately equal. Having run many different applications of clustering with different seed values, I was not able to find any significant difference in relative cluster size, nor in mean error. Between the two applications that I submitted, figure 1 converged after 12 epochs, whereas figure 2 converged after 10. This suggests that the randomly chosen initial points were better in figure 2, which had an impact on training time (0.18 seconds vs 0.25 seconds).

Figure 1: First clustering

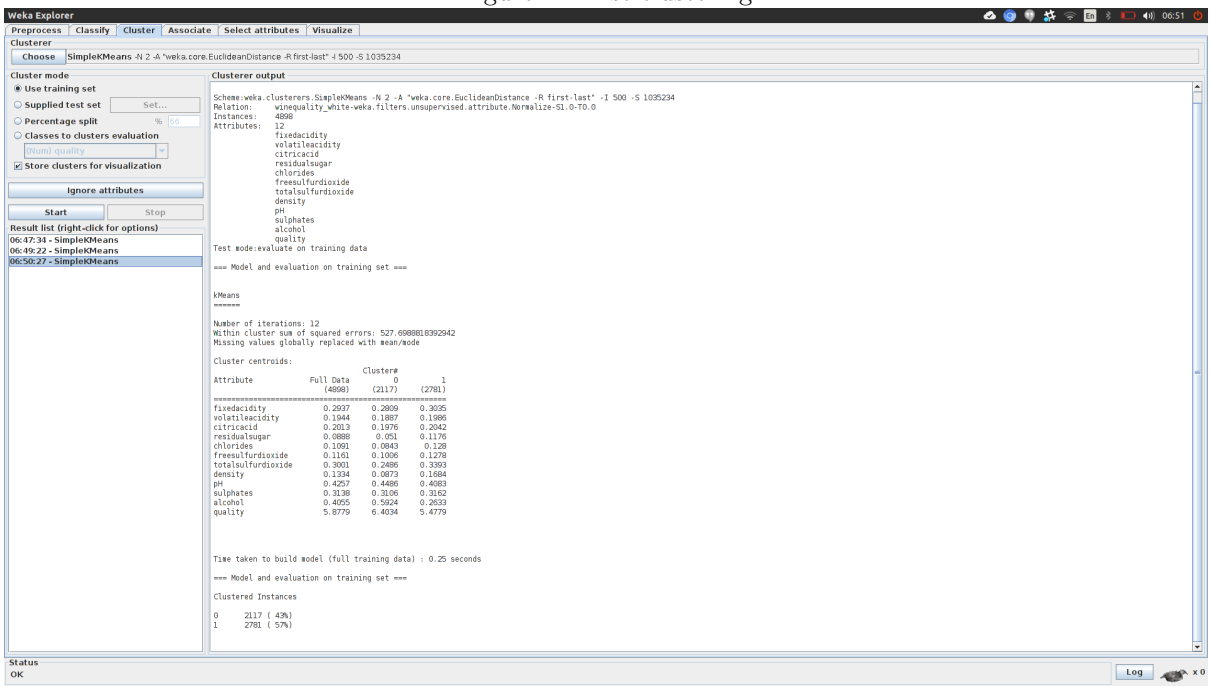
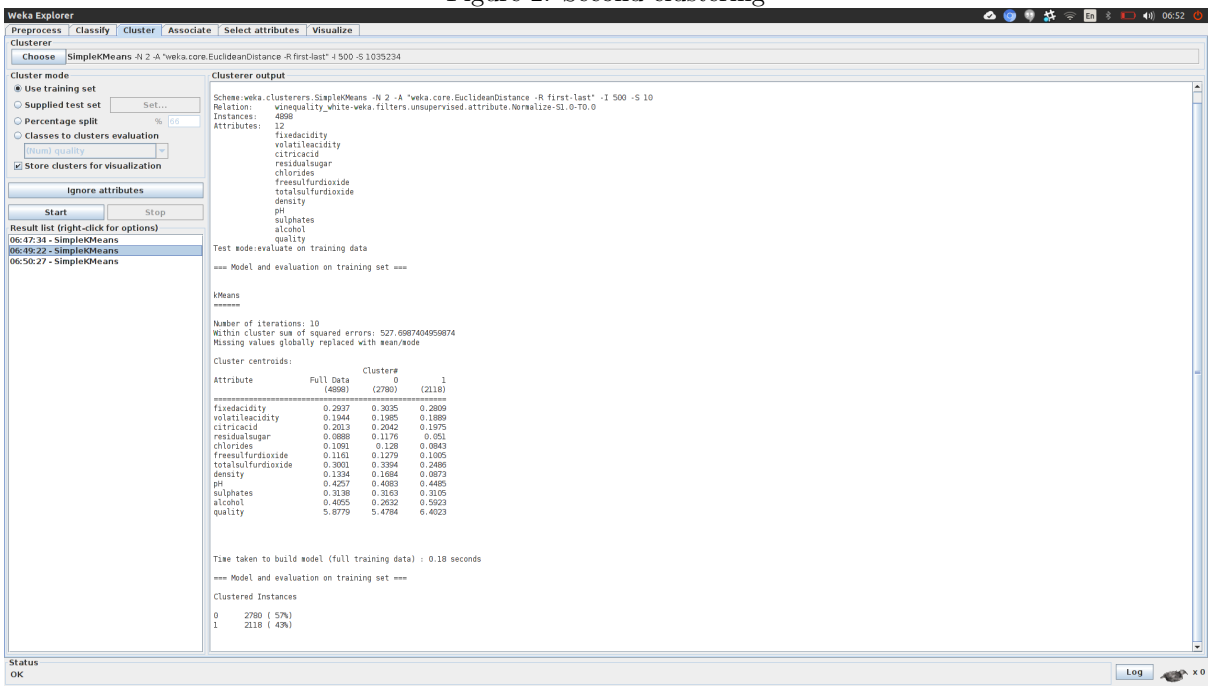


Figure 2: Second clustering



5 Question Four

Please find screenshots accompanying this section in the submitted zipfile. There is a) a screenshot showing selected attributes in Weka, and b) screenshots showing the confusion matrix and model evaluation for each of 5 models.

I used Weka's Select Attributes functionality to choose the best attributes to use for classification. These were:

0.09272242852567214	1	BalanceCurrentAccount
0.04932247273892688	3	PaymentPreviousCredits
0.03863998862142704	2	DurationMonths
0.03128437108814919	6	ValueSavingsOrStocks
0.01968464771721124	12	MostValuableAssets
0.01758697265140674	15	TypeApartment
0.01596202504161803	13	Age
0.01471565605847447	7	InCurrentEmployment
0.01447046066378477	9	GenderMaritalStatus

My 'best' model was selected based on least error. The following is a list of the models, and their accuracy.

Random Forest	74.35%
Decision Tree	72.87%
Decision Table	70.76%
ZeroR	70%
KNN	67.88%

The Random Forest performed the best.