Nick Chapman

# Data Mining Project 2

February 23, 2016

COSC-285

Project status: Complete

Time spent: 12 hours

Relevant information not given beforehand:

1. The correlation is more easily calculated in terms of confidences than in terms of raw probabilities

**Project Design**

My association rule mining engine has 2 main parts to it: the preprocessor and the miner itself. The preprocessor does a relatively minimal amount of work. Its primary function is to load the data files and ensure that the market basket data corresponds to the specified product list. To ensure that the two are compatible the DataFileParser simply ensures that the length of the first row of sales data is the same as the length of the product list. The DataFileParser loads the two files, checks that they are compatible with each other, and then extracts the relevant information.

In the basket data file, the relevant information is only whether each transaction contains one or more of the products on the product list. The transaction IDs are stripped out when the data file is loaded since we don't have to do anything with it. Once the product list and the market basket data is generated, it is combined into item sets that are Python `set`s of my own Product objects. There is one set per transaction in the basket data and each set is simply the collection of Products that were sold in the transaction. When we convert the transaction record to a set we lose the information on how many of each product we sold, however for association rule mining this is irrelevant. It doesn't matter to us whether someone bought one avocado or whether they bought 1000, just that they bought them at all.

Once the data is loaded and in a usable format it is passed off to the AssociationRuleMiner. The basic operational flow of the AssociationRuleMiner is:

1. Generate frequent item sets
2. Use the frequent item sets to generate association rules

To generate the frequent item sets we utilize the a priori principle and begin by generating the size k=1 frequent item sets. The size k=1 sets are then joined into k=2 possible frequent item sets. Then for each possible item set we calculate whether there is appropriate support for it. I note that I chose to check whether an item set was frequent based on support rather than on stratifying it back into subsets and checking whether it contained any potentially infrequent subsets. This approach turned out to be computationally faster since we didn't necessarily have information for all of the subsets that the new set could be split into. Once the frequent item sets of size k=2 were created we repeated this process incrementing k until there was a level with 1 or fewer frequent item sets. The algorithm terminates when a level contains only one frequent item set because in this case it is impossible to join anything.

With all of the frequent item sets generated I produced all of the potential association rules from each frequent item set. To do this I generated all of the possible antecedents and their corresponding consequents and then tested each pair as a rule. Each pair was put into an AssociationRule with its corresponding support, confidence, and lift. All of these potential rules are generated in AssociationRuleMiner.get_possible_rules(frequent_item_set) which is called in

AssociationRuleMiner.get_association_rules. In get_association_rules we prune down the results returned by get_possible_rules. For every rule if its support or confidence is below the threshold then the rule is disregarded. Additionally, any rules that have a lift <= 1 are disregarded since that means they either tell us less information than we started with or the rule is independent. As an aside, it was easier to calculate the correlation as the ratio of confidence to expected confidence than as a bunch of raw probabilities.

> Performance note: To get improved performance in calculating rule statistics I used a hash map to keep track of the confidence counts for all generated antecedents. This dramatically sped up performance since anytime we encountered a repeated antecedent (which happens quite a lot when you are checking all possible pairings) all of the difficult work of scanning the entire data set is already done. This makes the cost of calculating a rule's confidence much less than the O(len(data set)) that one would expect.

Finally, once we have all of the acceptable rules I sorted them using a stable sort first on lift and then on confidence. This way the rules were sorted primarily on confidence and then when the confidences were similar the one with the better lift (ie greater correlation) was taken.

The demo was put together using the AssociationRuleMiner's full_run_with_report which is simply a helper that builds all of the frequent item sets, generates the rules, and then outputs the appropriate information to the screen.

**Report 1**

| Min-Support | Min-Confidence | Number of frequent itemsets (for each k) | Largest value of k | Number of Rules | Total Run Time (Seconds) |
|---|---|---|---|---|---|
| 0.20 | 0.75 | k=1: 71<br>k=2: 69<br>k=3: 3<br>k=4: 0 | 3 | 11 | 2.790 |
| 0.40 | 0.75 | k=1: 10<br>k-2: 1 | 2 | 1 | 0.0817 |
| 0.50 | 0.75 | k=1: 2<br>k=2: 0 | 1 | 0 | 0.0551 |
| 0.20 | 0.60 | k=1: 71<br>k=2: 69<br>k=3: 3<br>k=4: 0 | 3 | 65 | 3.32 |
| 0.40 | 0.60 | k=1: 10<br>k=2: 1 | 2 | 2 | 0.211 |
| 0.50 | 0.60 | k=1: 2<br>k=2: 0 | 1 | 0 | 0.173 |
| 0.50 | 0.50 | k=1: 2<br>k=2: 0 | 1 | 0 | 0.168 |
| 0.30 | 0.50 | k=1: 53<br>k=2: 3<br>k=3: 0 | 2 | 5 | 0.921 |

Total demo runtime: 8.19 seconds

**Report 2[1]**

1. General Mills Multi Bran Chex + Avocado -> Bounty 8-Pack White Paper Towels
    - Sup=0.245 ; Conf=0.979 ; Lift=3.912
2. General Mills Multi Bran Chex -> Bounty 8-Pack White Paper Towels
    - Sup=0.377 ; Conf=0.977 ; Lift=2.530
3. Puff's Plus Facial Tissue -> Avocado
    - Sup=0.453 ; Conf=0.938 ; Lift=1.944
4. Skippy Peanut Butter + Puff's Plus Facial Tissue -> Avocado
    - Sup=0.220 ; Conf=0.932 ; Lift=3.946
5. Dole 6-Pack Pineapple Juice + Puff's Plus Facial Tissue -> Avocado
    - Sup=0.233 ; Conf=0.932 ; Lift=3.728
6. RealLemon Pure Lemon Juice -> Dole 6-Pack Pineapple Juice
    - Sup=0.259 ; Conf=0.920 ; Lift=3.268
7. Skippy Peanut Butter + Avocado -> Puff's Plus Facial Tissue
    - Sup=0.220 ; Conf=0.873 ; Lift=3.461
8. Avocado + Bounty 8-Pack White Paper Towels -> General Mills Multi Bran Chex
    - Sup=0.245 ; Conf=0.854 ; Lift=2.980
9. Bounty 8-Pack White Paper Towels -> General Mills Multi Bran Chex
    - Sup=0.377 ; Conf=0.849 ; Lift=1.912
10. Skippy Peanut Butter -> Avocado
    - Sup=0.252 ; Conf=0.805 ; Lift=2.571
11. Skippy Peanut Butter -> Puff's Plus Facial Tissue
    - Sup=0.236 ; Conf=0.754 ; Lift=2.408
12. Ocean Spray White Grapefruit Juice -> Apple & Eve Apple Juice
    - Sup=0.215 ; Conf=0.741 ; Lift=2.556
13. Dole 6-Pack Pineapple Juice + Avocado -> Puff's Plus Facial Tissue
    - Sup=0.233 ; Conf=0.704 ; Lift=2.127
14. Skippy Peanut Butter -> Avocado + Puff's Plus Facial Tissue
    - Sup=0.220 ; Conf=0.703 ; Lift=2.245
15. Tylenol Infant Liquid -> Avocado
    - Sup=0.257 ; Conf=0.697 ; Lift=1.889

[1] Top 15 generated rules based on confidence and lift

**Analysis**

These top 15 rules that we have generated are actually quite interesting. The first thing that is interesting to note is that people purchased avocados and General Mills Multi Bran Chex so much that they are largely correlated to the purchase of many other products. Looking at the support for the rules that are generated is also quite interesting. It's pretty wild to see a rule with higher than 90% confidence that applies to almost 50% of all transactions, with numbers like that it's just so ubiquitous. Furthermore, one would expect that rules with such high scores would be simple things like paper plates -> paper napkins or forks -> knives, but in our case it's tissues -> avocados which is startling.

These top 15 rules are also interesting in the ways that they can be chained together. For example we have a rule saying that if you buy children's Tylenol then you're more likely to purchase an avocado. But we also know that if you purchase an avocado you have an increased likely hood of purchasing a number of other products. So to some extent this informs us of something along the lines of if you are at the grocery store and you purchase medicine for your child and one grocery item, then you are inclined to do more grocery shopping. Upon examination this also makes sense because the two scenarios we can envision would likely be "grocery shopping -> might as well get Tylenol -> more groceries" vs "ill child -> buy Tylenol -> go home".

With respect to the pairs of minimum supports and confidences, these also returned some interesting results. The first thing we note from looking at the relationship between item set generation and minimum support is that the two are intimately paired. This makes sense since the minimum threshold is used to generate the frequent item sets. From these item sets that we generate we see that the confidence threshold is then a slider detailing what portion of the frequent item sets we are interested in hearing about.

Points of interest in Report 1 are definitely the cases where the support is so high that we don't generate any frequent item sets, such as when the support threshold is 0.50. It's not terribly surprising that when the support gets cranked way up we lose all item set frequency because in reality, there are very few products that are purchased in one out of every two orders. Setting the support threshold to 0.50 would really only be valuable during holidays or other periods like that where people's purchasing of a specific good increases. For example, during Halloween we could potentially filter the generated rules to only be about candy sales by setting the threshold to 0.50.

As far as wiggling confidence goes, that one is actually quite interesting. From this data we generate rules that we are tremendously confident in, it's just that they don't always apply to a tremendous number of items in the data. However, when the support threshold is at 20% we're looking at rules that

apply to 20% of transactions and we still get incredibly accurate results. We also note that on some rules the lift is incredibly high as well so not only are we fairly certain of the rule but it also tells us a lot more information than simply guessing would have.