B656 DATA MINING

Homework 3

9 April 2018

Chuck Jia

Problem 1

First, we prove the following statement[1].

Proposition 1. Let $\{x_i\}_{i=1}^n$ be a set of n data points in the d-dimensional Euclidean space. Let $x_i^{(j)}$ denote the j^{th} coordinate of x_i, i.e. $x_i = \left(x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(d)}\right) \in \mathbb{R}^d$ for all i. Let c be their centroid, i.e. $c = \frac{1}{n}\sum x_i$. Then

$$\text{SSE} \stackrel{\text{def}}{=} \sum_{i=1}^{n} \|x_i - c\|^2 = \frac{1}{n} \sum_{1 \le i < j \le n} \|x_i - x_j\|^2 \tag{E1}$$

where $\|\cdot\|$ is the Euclidean norm, i.e.

$$\|x_i\| = \left\| \left(x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(d)}\right) \right\| \stackrel{\text{def}}{=} \left[\sum_{k=1}^{d} \left(x_i^{(k)}\right)^2 \right]^{1/2}$$

Proof. We start from the left hand side of (E1).

$$\begin{aligned}
\sum_{i=1}^{n} \|x_i - c\|^2 &= \sum_{i=1}^{n} \left\| x_i - \frac{1}{n}\sum_{j=1}^{n} x_j \right\|^2 \\
&= \sum_{i=1}^{n} \left\| \frac{1}{n}\left(n x_i - \sum_{j=1}^{n} x_j \right) \right\|^2 \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \left\| n x_i - \sum_{j=1}^{n} x_j \right\|^2 \qquad \rightsquigarrow \text{Since } \|ax\| = a\|x\| \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \left\| \sum_{j=1}^{n} (x_i - x_j) \right\|^2 \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \left\| \left(\textstyle\sum_{j=1}^{n} (x_i^{(1)} - x_j^{(1)}), \cdots, \sum_{j=1}^{n} (x_i^{(d)} - x_j^{(d)}) \right) \right\|^2 \qquad \text{(E2)} \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{k=1}^{d} \left(\sum_{j=1}^{n} \left(x_i^{(k)} - x_j^{(k)} \right) \right)^2 \qquad \rightsquigarrow \text{By definition of the Euclidean norm} \\
&= \frac{1}{n^2} \sum_{k=1}^{d} \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \left(x_i^{(k)} - x_j^{(k)} \right) \right)^2 \qquad \rightsquigarrow \text{Changed order of summation on } i \text{ and } k \\
&= \frac{1}{n^2} \sum_{k=1}^{d} \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \left(x_i^{(k)} - x_j^{(k)} \right)^2 + \sum_{j_1 \neq j_2} \left(x_i^{(k)} - x_{j_1}^{(k)} \right) \cdot \left(x_i^{(k)} - x_{j_2}^{(k)} \right) \right) \\
&= \frac{1}{n^2} \sum_{k=1}^{d} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(x_i^{(k)} - x_j^{(k)} \right)^2 + \frac{1}{n^2} \sum_{k=1}^{d} \sum_{i=1}^{n} \sum_{j_1 \neq j_2} \left(x_i^{(k)} - x_{j_1}^{(k)} \right) \cdot \left(x_i^{(k)} - x_{j_2}^{(k)} \right)
\end{aligned}$$

[1] In this problem, we assume that we do not know $\{x_i\}$ or $\{c_j\}$.

Note that in the second term above, the inner two summations on i, j_1, j_2 can be simplified as

$$
\begin{aligned}
&\sum_{i=1}^{n}\sum_{j_1\neq j_2}\left(x_i^{(k)}-x_{j_1}^{(k)}\right)\cdot\left(x_i^{(k)}-x_{j_2}^{(k)}\right)\\
=&\sum_{i=1}^{n}\sum_{j_1\neq j_2}\left(\left[x_i^{(k)}\right]^2-x_i^{(k)}\cdot x_{j_1}^{(k)}-x_i^{(k)}\cdot x_{j_2}^{(k)}+x_{j_1}^{(k)}\cdot x_{j_2}^{(k)}\right)\\
=&\sum_{i=1}^{n}\sum_{j_1\neq j_2}\left[x_i^{(k)}\right]^2-\sum_{i=1}^{n}\sum_{j_1\neq j_2}x_i^{(k)}\cdot x_{j_1}^{(k)}-\sum_{i=1}^{n}\sum_{j_1\neq j_2}x_i^{(k)}\cdot x_{j_2}^{(k)}+\sum_{i=1}^{n}\sum_{j_1\neq j_2}x_{j_1}^{(k)}\cdot x_{j_2}^{(k)}\\
=&n(n-1)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-(n-1)\sum_{i=1}^{n}\sum_{j_1=1}^{n}x_i^{(k)}\cdot x_{j_1}^{(k)}-(n-1)\sum_{i=1}^{n}\sum_{j_2=1}^{n}x_i^{(k)}\cdot x_{j_2}^{(k)}+n\sum_{j_1\neq j_2}x_{j_1}^{(k)}\cdot x_{j_2}^{(k)}
\end{aligned}
$$

$\rightsquigarrow$ Step above: simplification of summation, using facts that $x_i^{(k)}$ is independent from j_1 and j_2, $x_i^{(k)}x_{j_1}^{(k)}$ is independent from j_2, $x_i^{(k)}x_{j_2}^{(k)}$ is independent from j_1, and $x_{j_1}^{(k)}x_{j_2}^{(k)}$ is independent from i

$$
\begin{aligned}
=&n(n-1)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-2(n-1)\sum_{i=1}^{n}\sum_{j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}+n\sum_{j_1\neq j_2}x_{j_1}^{(k)}\cdot x_{j_2}^{(k)} \qquad \rightsquigarrow \text{Combining middle 2 terms}\\
=&n(n-2)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2+n\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-2(n-1)\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}+n\sum_{j_1\neq j_2}x_{j_1}^{(k)}\cdot x_{j_2}^{(k)}\\
=&n(n-2)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-2(n-1)\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}+\left(n\sum_{j_1\neq j_2}x_{j_1}^{(k)}\cdot x_{j_2}^{(k)}+n\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2\right)
\end{aligned}
$$

$\rightsquigarrow$ Step above: combining the 2nd and the 4th terms

$$
\begin{aligned}
=&n(n-2)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-2(n-1)\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}+n\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}\\
=&n(n-2)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-(n-2)\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}\\
=&(n-2)\left(n\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-\sum_{i,j=1}^{n}x_i^{(k)}\cdot x_j^{(k)}\right)\\
=&(n-2)\left((n-1)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-\sum_{i\neq j}x_i^{(k)}\cdot x_j^{(k)}\right)\\
=&\frac{(n-2)}{2}\left(2(n-1)\sum_{i=1}^{n}\left[x_i^{(k)}\right]^2-\sum_{i\neq j}2x_i^{(k)}\cdot x_j^{(k)}\right)\\
=&\frac{(n-2)}{2}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2 \qquad \rightsquigarrow \text{Completing the square}
\end{aligned}
$$

Using this in (E2), we have

$$\begin{aligned}
\sum_{i=1}^{n}\|x_i-c\|^2 =& \frac{1}{n^2}\sum_{k=1}^{d}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i^{(k)}-x_j^{(k)}\right)^2+\frac{1}{n^2}\sum_{k=1}^{d}\sum_{i=1}^{n}\sum_{j_1\neq j_2}\left(x_i^{(k)}-x_{j_1}^{(k)}\right)\cdot\left(x_i^{(k)}-x_{j_2}^{(k)}\right)\\
=& \frac{1}{n^2}\sum_{k=1}^{d}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i^{(k)}-x_j^{(k)}\right)^2+\frac{1}{n^2}\frac{(n-2)}{2}\sum_{k=1}^{d}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2\\
=& \frac{1}{n^2}\sum_{k=1}^{d}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2+\frac{1}{n^2}\frac{(n-2)}{2}\sum_{k=1}^{d}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2\\
&\rightsquigarrow \text{Step above: since } x_i^{(k)}-x_j^{(k)}=0 \text{ when } i=j\\
=& \left(\frac{1}{n^2}+\frac{1}{n^2}\frac{(n-2)}{2}\right)\sum_{k=1}^{d}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2\\
=& \frac{1}{2n}\sum_{k=1}^{d}\sum_{i\neq j}\left(x_i^{(k)}-x_j^{(k)}\right)^2\\
=& \frac{1}{n}\sum_{k=1}^{d}\sum_{1\leq i<j\leq n}\left(x_i^{(k)}-x_j^{(k)}\right)^2 \quad \rightsquigarrow \text{By symmetry on } i \text{ and } j
\end{aligned}$$

Since our distance $d(\cdot,\cdot)$ is the Euclidean distance, we have

$$d(x,y)=\|x-y\|$$

Therefore, by Proposition 1, we can calculate the SSE of cluster k: $C_k=\{x_i\}_{i=1}^{n_k}$, by

$$\mathrm{SSE}_k=\frac{1}{n_k}\sum_{1\leq i<j\leq n_k}d(x_i,x_j)^2 \tag{E3}$$

where n_k is the number of data points in cluster k. Note that this calculation only involves the distances between data points and we do not need the data points themselves.

Therefore, we can find the best cluster pair to merge by the following procedures:

1. For any pair of clusters, cluster i and cluster j ($i<j$), calculate the following SSEs by using (E3):
   - the SSE of cluster i: SSE_i,
   - the SSE of cluster j: SSE_j, and
   - the SSE of the union of cluster i and j as a single cluster: $\mathrm{SSE}_{(i,j)}$

   Then we calculate the change in SSE by merging cluster i and j:

$$\Delta\mathrm{SSE}_{(i,j)}=\mathrm{SSE}_{(i,j)}-(\mathrm{SSE}_i+\mathrm{SSE}_j)$$

   Note that the change is always non-negative[2].

[2] $\mathrm{SSE}_{X\cup Y}=\sum_{z\in X\cup Y}\|z-c_{X\cup Y}\|^2=\sum_{z\in X}\|z-c_{X\cup Y}\|^2+\sum_{z\in Y}\|z-c_{X\cup Y}\|^2\geq\sum_{z\in X}\|z-c_X\|^2+\sum_{z\in Y}\|z-c_Y\|^2=\mathrm{SSE}_X+\mathrm{SSE}_Y$. The inequality holds, since c_X and c_Y are the minimizers for the problems $\min_c\sum_{z\in X}\|z-c\|^2$ and $\min_c\sum_{z\in Y}\|z-c\|^2$.

2. Select the pair (i_*, j_*) such that $\Delta\text{SSE}_{(i_*,j_*)}$ is the smallest among all $\Delta\text{SSE}_{(i,j)}$'s. Then cluster i_* and cluster j_* are the two clusters to merge.

Problem 2

(a) The frequent item set generation, with both $\mathbf{F}_{k-1} \times \mathbf{F}_1$ and $\mathbf{F}_{k-1} \times \mathbf{F}_{k-1}$, is implemented in the MATLAB function `apriori`, which completes the association rule generation algorithm, including both frequent item set generation and rule generation.

(b)

Car Set

1728 Examples, 25 Items

Min Support Threshold	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_1$**	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_{k\text{-}1}$**	**Savings**
0.1	393	350	43
0.15	273	253	20
0.2	261	253	8

Nursery Set

12960 Examples, 32 Items

Min Support Threshold	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_1$**	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_{k\text{-}1}$**	**Savings**
0.1	1251	937	314
0.12	784	531	253
0.16	642	497	145

Mushroom Set

8124 Examples, 39 Items

Min Support Threshold	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_1$**	**# of Candidates $\mathbf{F}_{k\text{-}1} \times \mathbf{F}_{k\text{-}1}$**	**Savings**
0.3	165	115	50
0.4	64	46	18
0.45	30	24	6

Observations: In our experiments with the 3 data sets, we can see the method $\mathbf{F}_{k-1} \times \mathbf{F}_{k-1}$ always generates fewer candidates than the method $\mathbf{F}_{k-1} \times \mathbf{F}_1$. Therefore, the method $\mathbf{F}_{k-1} \times \mathbf{F}_{k-1}$ is more efficient in frequent itemset generations.

(c)

Car Set

1728 Examples, 25 Items

Min Support Threshold	# of Frequent Itemsets	# of Closed Frequent Itemsets	# of Maximal Frequent Itemsets
0.1	86	73	43
0.15	43	41	22
0.2	31	29	22

Nursery Set

12960 Examples, 32 Items

Min Support Threshold	# of Frequent Itemsets	# of Closed Frequent Itemsets	# of Maximal Frequent Itemsets
0.1	179	155	123
0.12	88	82	58
0.16	63	57	41

Mushroom Set

8124 Examples, 39 Items

Min Support Threshold	# of Frequent Itemsets	# of Closed Frequent Itemsets	# of Maximal Frequent Itemsets
0.3	69	57	10
0.4	31	29	10
0.45	18	18	5

Comparison. In the above 3 data sets, there are much fewer maximal frequent itemsets then the total

number of frequent itemsets. The number of closed frequent sets are mostly less than the total number of frequent itemsets, but the difference is less significant.

(d)

Car Set

1728 Examples, 25 Items

(Min Support, Min Confidence)	# of Rules Conf. Pruning	# of Rules Brute Force	Savings
(0.1, 0.8)	15	170	155
(0.15, 0.7)	8	40	32
(0.2, 0.7)	5	16	11

Nursery Set

12960 Examples, 32 Items

(Min Support, Min Confidence)	# of Rules Conf. Pruning	# of Rules Brute Force	Savings
(0.12, 0.55)	10	124	114
(0.15, 0.6)	7	78	71
(0.17, 0.8)	2	8	6

Mushroom Set

8124 Examples, 39 Items

(Min Support, Min Confidence)	# of Rules Conf. Pruning	# of Rules Brute Force	Savings
(0.4, 0.97)	8	68	60
(0.45, 0.9)	13	34	21
(0.5, 0.9)	4	14	10

(e) To select the top 5 rules, for each combination of support and confidence thresholds, we rank the results

by the values of confidence and then by support values. Then we select the top 5 rules, keeping all the ties, which might result in selecting more than 5 rules.

Car Set

1. `min_support` $= 0.1$, `min_confidence` $= 0.9$

```
[ persons_2, ] -> [ evalulation_unacc, ]
[ safety_low, ] -> [ evalulation_unacc, ]
[ persons_2, lug_boot_small, ] -> [ evalulation_unacc, ]
[ persons_2, lug_boot_med, ] -> [ evalulation_unacc, ]
[ persons_2, lug_boot_big, ] -> [ evalulation_unacc, ]
[ persons_2, safety_low, ] -> [ evalulation_unacc, ]
[ persons_2, safety_med, ] -> [ evalulation_unacc, ]
[ persons_2, safety_high, ] -> [ evalulation_unacc, ]
[ persons_4, safety_low, ] -> [ evalulation_unacc, ]
[ persons_more, safety_low, ] -> [ evalulation_unacc, ]
[ lug_boot_small, safety_low, ] -> [ evalulation_unacc, ]
[ lug_boot_med, safety_low, ] -> [ evalulation_unacc, ]
[ lug_boot_big, safety_low, ] -> [ evalulation_unacc, ]
```

Comment. 13 rules generated in total. Many ties exist. All of the rules have confidence 1. The first 2 rules tie in support (0.33), and the last 11 rules tie in support (0.11). All the rules are of very decent quality. For instance, the first 2 rules indicate that in the car evaluation, cars with capacity 2 or with low safety standard are often marked as unacceptable. Note that later rules are in some sense "derived" from the first 2 rules. All rules with a `unacc` consequent include `persons_2` or `safety_low` in their antecendent. Therefore, those rules are also reasonable rules.

2. `min_support` $= 0.15$, `min_confidence` $= 0.7$

```
[ buying_vhigh, ] -> [ evalulation_unacc, ]
[ maint_vhigh, ] -> [ evalulation_unacc, ]
[ persons_2, ] -> [ evalulation_unacc, ]
[ lug_boot_small, ] -> [ evalulation_unacc, ]
[ safety_low, ] -> [ evalulation_unacc, ]
```

Comment. With this combination of support and confidence thresholds, the rules generated show very high quality. The 5 rules indicates that high purchase price, high maintenance cost, 2-person-only capacity, small trunk size, and low safety measures often occur with an unacceptable evaluation, which all seem very plausible compare to real world situations.

3. `min_support` $= 0.2$, `min_confidence` $= 0.7$

```
[ buying_vhigh, ] -> [ evalulation_unacc, ]
[ maint_vhigh, ] -> [ evalulation_unacc, ]
[ persons_2, ] -> [ evalulation_unacc, ]
[ lug_boot_small, ] -> [ evalulation_unacc, ]
[ safety_low, ] -> [ evalulation_unacc, ]
```

Comment. This level of support and confidence yielded the same result as with before. This indicates that all the rules have good support.

Also, as we can observe, many of the rules have confidence 1 or close to 1. Therefore, lowering the support than the levels in combination 1 would yield the same top 5 results. Increasing confidence in combination 2 or 3 would yield less than 5 rules. Therefore, we have only 2 different sets of top 5 rules in the experiments above.

In general, if we have rules with very high support and very high confidence, they will appear as top rules in all rule generation results with different levels of support and confidence. That is the reason we sometimes have same set of top rules for different levels of support and confidence.

Nursery Set

In this set, many of the rules generated have very high confidence, with confidence = 1. Therefore, using our ranking method, defined as above, there exists only 1 set of 6 rules using different levels of confidence and support.

1. `min_support` $= 0.14$, `min_confidence` $= 0.8$

```
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
[ finance_convenient, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_convenient, health_not_recom, ] -> [ admission_not_recom, ]
[ finance_inconv, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_inconv, health_not_recom, ] -> [ admission_not_recom, ]
```

Comment. The first 2 rules tie in support (0.33) and the other 4 rules tie in support (0.167). All rules have confidence 1. The first rules point out that applicants with bad health often are not admitted and applicants that are not admitted often have bad health. These seem to be reasonable rules. The other rules also have `admission_not_recom` in the antecedent with `health_not_recom` in the consequent or the other way around, with `admission_not_recom` in the consequent with `health_not_recom` in the antecedent. Therefore, they are also plausible rules.

2. `min_support` $= 0.15$, `min_confidence` $= 0.9$

```
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
```

```
[ finance_convenient, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_convenient, health_not_recom, ] -> [ admission_not_recom, ]
[ finance_inconv, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_inconv, health_not_recom, ] -> [ admission_not_recom, ]
```

Comment. Same as the other support and confidence levels. See explanation above.

3. `min_support` $= 0.16$, `min_confidence` $= 0.99$

```
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
[ finance_convenient, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_convenient, health_not_recom, ] -> [ admission_not_recom, ]
[ finance_inconv, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_inconv, health_not_recom, ] -> [ admission_not_recom, ]
```

Comment. Same as the other support and confidence levels. See explanation above.

Mushroom Set

1. `min_support` $= 0.4$, `min_confidence` $= 0.9$

```
[ edible_p, ] -> [ gill_attachment_f, ]
[ bruises_t, ] -> [ gill_attachment_f, ]
[ edible_p, bruises_f, ] -> [ gill_attachment_f, ]
[ edible_p, gill_spacing_c, ] -> [ gill_attachment_f, ]
[ bruises_t, gill_spacing_c, ] -> [ gill_attachment_f, ]
```

Comment. All five rules here have confidence level above 0.99. They are also very reasonable rules. For instance, many poisonous mushrooms (such as the pretty but deadly members of the Amanita family, which are responsible for approximately 95% of the fatalities resulting from mushroom poisoning[3]) have free gill attachments. Thus, the rule `[edible_p] -> [ gill_attachment_f]` is consistent with this fact.

2. `min_support` $= 0.3$, `min_confidence` $= 1$

```
[ edible_p, ] -> [ gill_attachment_f, ]
[ bruises_t, ] -> [ gill_attachment_f, ]
[ edible_p, bruises_f, ] -> [ gill_attachment_f, ]
[ edible_p, gill_spacing_c, ] -> [ gill_attachment_f, ]
[ bruises_t, gill_spacing_c, ] -> [ gill_attachment_f, ]
```

[3] https://en.wikipedia.org/wiki/Amanita

Comment. This level of confidence and support generates the same top 5 rules as in the last level. This is reasonable, as all the rules have very high confidence (≥ 0.99).

3. `min_support` $= 0.5$, `min_confidence` $= 0.8$

```
[ bruises_f, ] -> [ gill_attachment_f, ]
[ gill_spacing_c, ] -> [ gill_attachment_f, ]
[ gill_attachment_f, ] -> [ gill_spacing_c, ]
[ gill_size_b, ] -> [ gill_attachment_f, ]
[ gill_spacing_c, gill_size_b, ] -> [ gill_attachment_f, ]
```

Comment. With higher support but lower confidence, these rules are less indicative of causal relationships. For instance, bruises and gill attachment are usually independent factors. But it might also indicate some underlying causal links.

(f) Same as before, to select the top 5 rules, for each combination of support and lift thresholds, we rank the results by the values of lift and then by support values. Then we select the top 5 rules, keeping all the ties, which might result in selecting more than 5 rules.

Lift vs Confidence. For a rule $A \to B$, lift is defined to be

$$\text{lift}(A \to B) \stackrel{\text{def}}{=} \frac{\text{confidence}(A \to B)}{\text{support}(B)}$$

Therefore, in the calculation of lift, we not only consider the confidence, which reflects how often items in B appear in transactions that contain A, but also consider the support of B. The lift of the rule becomes smaller if the support of B is larger, which indicates that the rule might just be happenstance, since B is very prevalent. Therefore, using lift can help reduce rules that occur by coincidence.

Comparison on the Results. In the following results, most of the top rules are similar with those generated from confidence, containing similar items as in the confidence rules. These rules are also mostly of high quality. However, the lift rules below sometimes contain larger itemsets. This might be the effect of lift generating fewer coincidental rules. The antecedents and consequents are often different in order compared with the rules from the confidence generation.

Car Set

1. `min_support` $= 0.1$, `min_lift` $= 1.7$

```
[ safety_high, evalulation_unacc, ] -> [ persons_2, ]
[ persons_2, ] -> [ safety_high, evalulation_unacc, ]
[ persons_4, evalulation_unacc, ] -> [ safety_low, ]
[ safety_low, ] -> [ persons_4, evalulation_unacc, ]
[ persons_more, evalulation_unacc, ] -> [ safety_low, ]
[ safety_low, ] -> [ persons_more, evalulation_unacc, ]
```

Comment. All the six rules here have high lift level but not very high support level (support = 0.11). The rules here are of high quality. For example, the first rule indicates that when a highly safe car is marked as unacceptable, it often happens that the car has a small capacity of 2 persons. Other rules are reflective of usual relationships as well.

2. `min_support` $= 0.2$, `min_lift` $= 1.16$

```
[ evalulation_unacc, ] -> [ buying_vhigh, ]
[ buying_vhigh, ] -> [ evalulation_unacc, ]
[ evalulation_unacc, ] -> [ maint_vhigh, ]
[ maint_vhigh, ] -> [ evalulation_unacc, ]
[ evalulation_unacc, ] -> [ persons_2, ]
[ persons_2, ] -> [ evalulation_unacc, ]
[ evalulation_unacc, ] -> [ safety_low, ]
[ safety_low, ] -> [ evalulation_unacc, ]
```

Comment. All the rules are very reasonable rules as well. For instance, the first rule indicates that an unacceptable evaluation often happens when the buying price is very high.

3. `min_support` $= 0.05$, `min_lift` $= 2.1$

```
[ persons_4, safety_med, ] -> [ evalulation_acc, ]
[ evalulation_acc, ] -> [ persons_4, safety_med, ]
[ persons_4, safety_high, ] -> [ evalulation_acc, ],
[ evalulation_acc, ] -> [ persons_4, safety_high, ]
[ persons_more, safety_med, ] -> [ evalulation_acc, ]
[ evalulation_acc, ] -> [ persons_more, safety_med, ]
[ persons_more, safety_high, ] -> [ evalulation_acc, ]
[ evalulation_acc, ] -> [ persons_more, safety_high, ]
```

Comment. This lower support but higher confidence level produces reasonable rules as well. For instance, the first rule says that a 4-person car with medium safety level is often found acceptable.

Nursery Set

1. `min_support` $= 0.1$, `min_lift` $= 1.7$

```
[ admission_priority, ] -> [ health_recommended, ]
[ health_recommended, ] -> [ admission_priority, ]
[ admission_spec_prior, ] -> [ health_priority, ]
[ health_priority, ] -> [ admission_spec_prior, ]
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
```

2. min_support = 0.15, min_lift = 2

```
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
[ finance_convenient, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_convenient, health_not_recom, ] -> [ admission_not_recom, ]
[ admission_not_recom, ] -> [ finance_convenient, health_not_recom, ]
[ health_not_recom, ] -> [ finance_convenient, admission_not_recom, ]
[ finance_inconv, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_inconv, health_not_recom, ] -> [ admission_not_recom, ]
[ admission_not_recom, ] -> [ finance_inconv, health_not_recom, ]
[ health_not_recom, ] -> [ finance_inconv, admission_not_recom, ]
```

3. min_support = 0.15, min_lift = 3

```
[ admission_not_recom, ] -> [ health_not_recom, ]
[ health_not_recom, ] -> [ admission_not_recom, ]
[ finance_convenient, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_convenient, health_not_recom, ] -> [ admission_not_recom, ]
[ admission_not_recom, ] -> [ finance_convenient, health_not_recom, ]
[ health_not_recom, ] -> [ finance_convenient, admission_not_recom, ]
[ finance_inconv, admission_not_recom, ] -> [ health_not_recom, ]
[ finance_inconv, health_not_recom, ] -> [ admission_not_recom, ]
[ admission_not_recom, ] -> [ finance_inconv, health_not_recom, ]
[ health_not_recom, ] -> [ finance_inconv, admission_not_recom, ]
```

Mushroom Set

1. min_support = 0.5, min_lift = 0.9

```
[ gill_attachment_f, ] -> [ bruises_f, ]
[ bruises_f, ] -> [ gill_attachment_f, ]
[ gill_spacing_c, ] -> [ gill_attachment_f, ]
[ gill_attachment_f, ] -> [ gill_spacing_c, ]
[ gill_size_b, ] -> [ gill_attachment_f, ]
[ gill_attachment_f, ] -> [ gill_size_b, ]
```

2. min_support = 0.3, min_lift = 2

```
[ bruises_t, gill_size_b, ] -> [ edible_e, gill_spacing_c, ]
[ edible_e, gill_spacing_c, ] -> [ bruises_t, gill_size_b, ]
[ edible_e, gill_attachment_f, gill_spacing_c, gill_size_b, ] -> [ bruises_t, ]
[ bruises_t, gill_attachment_f, gill_size_b, ] -> [ edible_e, gill_spacing_c, ]
[ edible_e, gill_attachment_f, gill_spacing_c, ] -> [ bruises_t, gill_size_b, ]
[ bruises_t, gill_size_b, ] -> [ edible_e, gill_attachment_f, gill_spacing_c, ]
[ edible_e, gill_spacing_c, ] -> [ bruises_t, gill_attachment_f, gill_size_b, ]
[ bruises_t, ] -> [ edible_e, gill_attachment_f, gill_spacing_c, gill_size_b, ]
```

3. `min_support` $= 0.2$, `min_lift` $= 2.5$

```
[ odor_n, gill_attachment_f, gill_spacing_c, gill_size_b, ] -> [ edible_e, bruises_t, ]
[ edible_e, gill_attachment_f, gill_spacing_c, gill_size_b, ] -> [ bruises_t, odor_n, ]
[ bruises_t, odor_n, gill_size_b, ] -> [ edible_e, gill_attachment_f, gill_spacing_c, ]
[ edible_e, gill_attachment_f, gill_spacing_c, ] -> [ bruises_t, odor_n, gill_size_b, ]
[ bruises_t, odor_n, ] -> [ edible_e, gill_attachment_f, gill_spacing_c, gill_size_b, ]
[ edible_e, bruises_t, ] -> [ odor_n, gill_attachment_f, gill_spacing_c, gill_size_b, ]
```

Problem 3

The program for this problem is coded in C++. To run the program, compile and run `p3.cpp`. The execution of `p3` uses the following format

```
p3 [Method] [DistFcn] [IfUseUserDemo] [DataSet] [TrainFile] [TestFile]
[UserDemoFile] [Min#Neighbors] [Max#Neighbors]
```

where

- `[Method] = naive, nnb`
- `[DistFcn] = euclidean, mink1, mink2, mink3, mink7, minkInf, cosine`
- `[IfUseUserDemo] = use, no_use`
- `[DataSet] = 100k, 10m`

(a) First, we give the results from our calculation using nearest neighbors. The best k's and their MAD scores are written in bold. Note that although only 5 results are listed in the table for each distance function, they are the best 5 results among all k's less than 100. We tested each distance function with 1 - 100 neighbors and selected the best 5 results.

Mean Absolute Difference

100K Dataset

Euclidean Distance		**Distance Function 1**		**Distance Function 2**	
# of Neighbors	MAD	# of Neighbors	MAD	# of Neighbors	MAD
96	0.8206432	20	0.771841	15	0.7773154
97	0.8205710	21	0.771737	16	0.7770438
98	0.8205184	**22**	**0.771580**	17	0.7771542
99	0.8204244	23	0.771951	**18**	**0.7770396**
100	**0.8203950**	24	0.772110	19	0.7775290

Table 3.1. *Mean absolute difference values for each choice of distance function and number of neighbors. All MAD scores are averages of scores from 5 validation sets. In the table above, the best choice of number of neighbors are emboldened.*

Let M be the number of movies in total in our data set. In our calculation of user dissimilarities, each user i is represented by a vector u_i of length M (number of movies in total). The j^{th} element of u_i, denoted by $u_i^{(j)}$, is the rating of movie j by user i. If user i did not rate movie j, then $u_i^{(j)}$ is set to be 0. Our Euclidean distance function is thus defined as usual,

$$d_{\text{Eucl}}(u_{i_1}, u_{i_2}) \stackrel{\text{def}}{=} \|u_{i_1} - u_{i_2}\|_2 = \left(\sum_{j=1}^{M} \left(u_{i_1}^{(j)} - u_{i_2}^{(j)} \right)^2 \right)^{1/2}$$

Our distance function 1 is defined as

$$d_1(u_{i_1}, u_{i_2}) \stackrel{\text{def}}{=} \frac{\sum_{j=1}^{M} \left| u_{i_1}^{(j)} - u_{i_2}^{(j)} \right| \cdot I\left(u_{i_1}^{(j)} \neq 0,\ u_{i_2}^{(j)} \neq 0 \right)}{\text{card}\left(\left\{ j : u_{i_1}^{(j)} \neq 0,\ u_{i_2}^{(j)} \neq 0 \right\} \right)}$$

where $I(\cdot)$ is the indicator function, i.e. $I(\texttt{true}) = 1$ and $I(\texttt{false}) = 1$, and $\text{card}(\cdot)$ is the cardinality of a set, i.e. for any set E, $\text{card}(E)$ = the number of elements in set E. d_1 is in fact the city block distance (l^1 distance), ignoring all movies that are not rated by at least 1 user, and normalized by the number of movies rated by both users.

Our distance function 2 is defined as

$$d_2(u_{i_1}, u_{i_2}) \stackrel{\text{def}}{=} \frac{\sum_{j=1}^{M} \left(u_{i_1}^{(j)} - u_{i_2}^{(j)} \right)^2 \cdot I\left(u_{i_1}^{(j)} \neq 0,\ u_{i_2}^{(j)} \neq 0 \right)}{\text{card}\left(\left\{ j : u_{i_1}^{(j)} \neq 0,\ u_{i_2}^{(j)} \neq 0 \right\} \right)}$$

d_1 is the Euclidean distance (l^2 distance), ignoring all movies that are not rated by at least 1 user, and normalized by the number of movies rated by both users.

Comparison with Results From the Naive Algorithm

Using the naive algorithm, i.e. the algorithm that gives each user-movie pair a rating that equals the average score over all users who rated the movie, we have the following result:

$$\text{MAD of naive algorithm} = 0.816903$$

This result is the average of the 5 MAD scores calculated using 5 folds of validation sets.

Note that compared to the naive algorithm, our distance function 1 d_1 with 22 neighbors performs much better in terms of the MAD score.

(b) The results using both movie ratings and user information is given below. The best results are written in bold.

Mean Absolute Difference

100K Dataset with User Information

Cosine Distance Function		**Distance Function 1**		**Distance Function 2**	
# of Neighbors	MAD	# of Neighbors	MAD	# of Neighbors	MAD
50	**0.8049342**	**20**	**0.7710190**	15	0.7773470
51	0.8049728	21	0.7710472	**16**	**0.7769888**
52	0.8049872	22	0.7710350	17	0.7771540
53	0.8050124	23	0.7710248	18	0.7771010
54	0.8050592	24	0.7714130	19	0.7775010

Table 3.2. *Mean absolute difference values for each choice of distance function and number of neighbors. All MAD scores are averages of scores from 5 validation sets. In the table above, the best choice of number of neighbors are emboldened.*

Here, the cosine distance between two users is defined as usual:

$$d_{\cos}(u_{i_1}, u_{i_2}) \stackrel{\text{def}}{=} 1 - \frac{u_{i_1} \cdot u_{i_1}}{\|u_{i_1}\|_2 \|u_{i_2}\|_2}$$

How We Incorporate User Information in All Distance Calculations

We modify our distance by adding user information. The actual distance we use in this section is of form

$$d^{\text{modified}}(u_{i_1}, u_{i_2}) = d(u_{i_1}, u_{i_2}) + a_1[\text{Age}(u_{i_1}) - \text{Age}(u_{i_2})] + a_2[\text{Gender}(u_{i_1}) - \text{Gender}(u_{i_2})] \tag{E4}$$

where d is one of $d_{\cos}, d_1, d_2$ we defined above; $\text{Age}(u_i)$ is the age of the user i; $\text{Gender}(u_i)$ is the gender of the user i, with 0 being male and 1 being female; a_1 and a_2 are constants, depending only on the choice of d. For d_1, we choose $a_1 = 0.05$ and $a_2 = 0.03$, for d_2 and d_3, we choose $a_1 = 1 \times 10^{-6}$ and $a_2 = 1 \times 10^{-6}$.

Justification of the Modified Distance Functions

It is reasonable to assume that two users of different age might have different rating tendencies, as they grew up in different years and cultural differences are often large across generations. It is also reasonable

to assume that two users from different genders might hold different views towards movies. For instance, more male audience might prefer action movies, which female audience might prefer less of the genre. Therefore, we add the difference in age and gender to our distance calculation (E4). Also, the effect of age and gender might be very different. Therefore, we used a_1 and a_2, depending on the specific distance function, to weight the effect of the two factors.

Comments on the Results

As we can see, adding user information generally had positive effects. In particular, our modified d_1 function gives better results on all numbers of neighbors tested.

(c) Note that in order to save on memory usage, for the 10M data set, our program stores ratings as 10 integers $\{1,2,\cdots,10\}$, instead of the original $\{1.0,1.5,2.0,2.5,\cdots,5.5,5.0\}$. Therefore, the output MAD score by the program should be divided by 2 when calculating the correct MAD score.

Our results on the 10M data set is given as follows.

Mean Absolute Difference

10M Dataset

Cosine Distance Function		**Distance Function 1**		**Distance Function 2**	
# of Neighbors	MAD	# of Neighbors	MAD	# of Neighbors	MAD
50	0.8000224	22	0.7613710	16	0.7693572

Table 3.3. *Mean absolute difference values for each choice of distance function and number of neighbors.*

Naive algorithm gives the result

$$\text{MAD of naive algorithm} = 0.7380836$$

Comments on the Results. Note that using the nearest neighbor algorithms, the MAD scores on the 10M datasets are similar with those on the 100K datasets. Their performances are thus similar as on the smaller data sets.

However, the naive algorithm performs much better on the larger data set. Its MAD score is now lower than the score by the nearest neighbor algorithms. This is also reasonable, since if we have more data on a movie, then the average rating using all available data is going to be more accurate than using a small number of neighbors. When having large data sets, the effect of a few “bad" or inaccurate ratings is going to be very small. The average of all data is therefore closer to the true prediction.

(d) Normalizing user ratings might be a good next step. Normalization might help eliminate the bias on users' rating habits. For instance, user 1 might be more critical, only rating his favorite movies as 4 and most of his movies as 1, while another user, user 2, might be very easy on rating, ranking most of his movies as 5. Normalizing each user's ratings using their means and standard deviations might help with this problem and bring more accurate predictions.