Datasta Tietoon, Autumn 2007

SOLUTIONS TO EXERCISES 5

H5 / Problem 1.

Simulate the levelwise algorithm. In the first phase the candidates are all sets of one variable $\{a\}$, $\{b\}$, $\{c\}$ ja $\{d\}$. To be more convenient, we will omit all { and } from now on, and write all sets simply a, b, c, and d. The frequencies of these

a	b	c	d
7	6	7	7.

Frequencies of all sets are equal or more than the threshold, so all sets are frequent. Now the following level candidates are all sets of two variables (again $ab = \{a, b\}$ and so on):

ab	ac	ad	bc	bd	cd
3	5	4	4	6	5.

All sets except ab are frequent. The candidates of 3-size sets are

acd	bcd
3	4.

Here only bcd is frequent. Therefore any larger set (in this case $abcd$) cannot be frequent and algorithm stops.

The frequent itemsets are a, b, c, d, ac, ad, bc, bd, cd, and bcd. (Often the empty set is also considered to be a frequent set.)

You can consider, e.g., observations as bags (customers), and variables as products in a supermarket, for example, a is for apples, b is for bread, c is for cheese, and d is for soda. In the 0-1-matrix each 1 means that the particular item is found in the shopping bag. The first customer has bought bread and soda, the last tenth customer all four products.

H5 / Problem 2.

When computing time complexities of algorithms it is interesting to see the asymptotic behavior of algorithms, that is, when the size of input grows to infinity. In this case time complexity is examined as a function of both input size and number of candidates. The latter connection is more difficult to explain. If the number of candidates were not taken into account, the worst case would be trivially that where the data contains only 1s. In that case all possible variables sets would become candidates, that is exponential case.

The levelwise algorithm shown in the lectures is written with pseudocode below. Let us call the size of data (number of observations) with m, and the number of all processed candidate sets with n. Candidate sets with k size candidate is marked C_k. Let t be the biggest value of k, i.e., the maximum size of candidates. Clearly, $n = \sum_{k=1}^{t} |C_k|$ and $k \leq t = O(\ln n)$

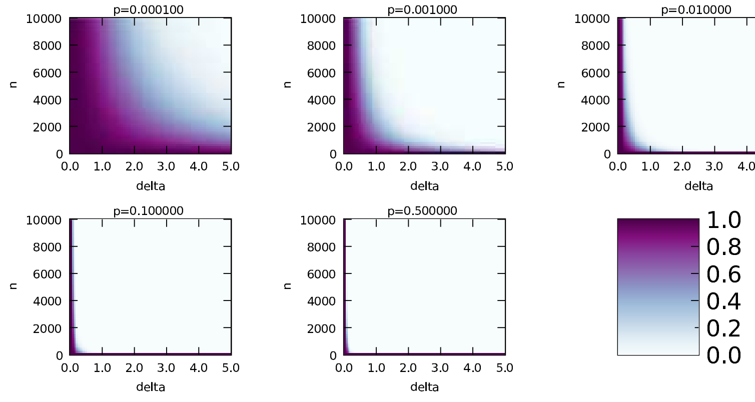
While-loop in row 3 is executed t times. At one execution for-loop in row 5 is executed m times, and at one execution step of that for-loop in row 6 is executed $|C_k|$ times. Totally, this for-loop is executed mn times. At one execution the for-loop in row 8 is computed k times, and those operations can be considered as taking a constant time. As well the if-statement in row 11 takes a constant time. Hence, the time complexity of the for-loop in row 5 is $O(mn \ln n)$.

```
 1: $k \leftarrow 1$
 2: $C_k \leftarrow \{\, \{a\} \mid a \in \text{variables} \,\}$
 3: while $C_k \neq \emptyset$ do
 4:     counter$[X] \leftarrow 0$ for all $X$
 5:     for observation in data do                          ▷ Count frequencies of candidates
 6:         for $X$ in $C_k$ do                              ▷ Check if all variables in $X$ are present
 7:             good ← True
 8:             for var in $X$ do
 9:                 if observation[var] $= 0$ then
10:                     good ← False
11:             if good then
12:                 counter$[X] \leftarrow$ counter$[X] + 1$
13:     $F_k \leftarrow \emptyset$
14:     for $X$ in $C_k$ do                                  ▷ Select frequent candidates
15:         if counter$[X] \geq N$ then
16:             $F_k \leftarrow F_k \cup \{X\}$
17:     $C_{k+1} \leftarrow \emptyset$
18:     for $A$ in $F_k$ do                                  ▷ Generate next candidates
19:         for $B$ in $F_k$ do
20:             $X \leftarrow A \cup B$
21:             if $|X| = k + 1$ then
22:                 good ← True
23:                 for var in $X$ do
24:                     if $X \setminus \{\text{var}\}$ not in $F_k$ then
25:                         good ← False
26:                 if good then
27:                     $C_{k+1} \leftarrow C_{k+1} \cup \{X\}$
28:     $k \leftarrow k + 1$
```

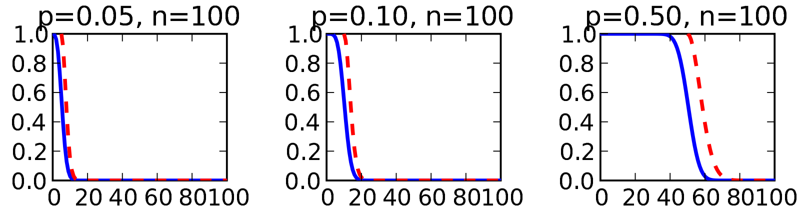
The for-loop in row 14 is executed n times and the lines inside it have constant times. The time complexity for rows 13–17 is $O(n)$, and becausedd $n = O(mn \ln n)$, it has not asymptotical meaning.

For-loops in rows 18 and 19 are executed totally $t|F_k|^2 \leq t|C_k|^2 = O(n^2 \ln n)$ times. The statement in row 20 takes at most $O(2k) = O(\ln n)$. The for-loop in row 23 is executed $k + 1 = O(\ln n)$ times, and the lines inside it as constants (F_k can be implemented with hash tables where testing is practically constant-time). The for-loop in row 18 is therefore $O(n^2(\ln n)^2)$. Because $mn \ln n$ and $n^2(\ln n)^2$ are not asymptotically comparable, the whole time complexity of the algorithm is $O(mn \ln n + n^2(\ln n)^2)$.

H5 / Problem 3.
The following figures show the behavior of the border.



The distance of the border and true probability can be examined with small enough values of n. In the following the solid curve depicts the probability $\Pr(X \geq x)$ and dashed line Tšernov border:



Tšernov border can be shown quite easily. Consider *Markov's inequality*, which deals non-negative random variables X and for which

$$\Pr(X \geq a) \leq \frac{E[X]}{a}.$$

In this case it is needed for discrete random variables. Let possible values of X be $x_1 < x_2 < \cdots$. Let x_j be the biggest of those values, which is smaller than a. Then

$$E[x] = \sum_{i=1}^{\infty} x_i \Pr(X = x_i) \geq \sum_{i=j+1}^{\infty} x_i \Pr(X = x_i)$$

$$\geq a \sum_{i=j+1}^{\infty} \Pr(X = x_i) = a \Pr(X \geq x_{j+1}) = a \Pr(X \geq a).$$

The other needed result is the inequality $1 + x \leq e^x$, which hold for all real values x and can be seen true by examining the function $f(x) = e^x - x - 1$: because $f''(x) = e^x > 0$ and $f'(0) = 0$, $f(x) \geq f(0) = 0$ for all x.

Let $Y_1, Y_2, \ldots, Y_n$ be independent random variables, where each gets the value 1 with probability p and the value 0 with probability $1 - p$, and let $X = \sum_{i=1}^{n} Y_i$. Let $t \geq 0$. Now using Markov's inequality

$$\Pr(X \geq (1+\delta)np) \leq \Pr(e^{tX} \geq e^{t(1+\delta)np}) \leq e^{-t(1+\delta)np} E[e^{tX}].$$

For the expectation of variable e^{tX} holds (independence)

$$E[e^{tX}] = E[e^{t(Y_1 + \cdots + Y_n)}] = E[e^{tY_1} \cdots e^{tY_n}] = E[e^{tY_1}]^n.$$

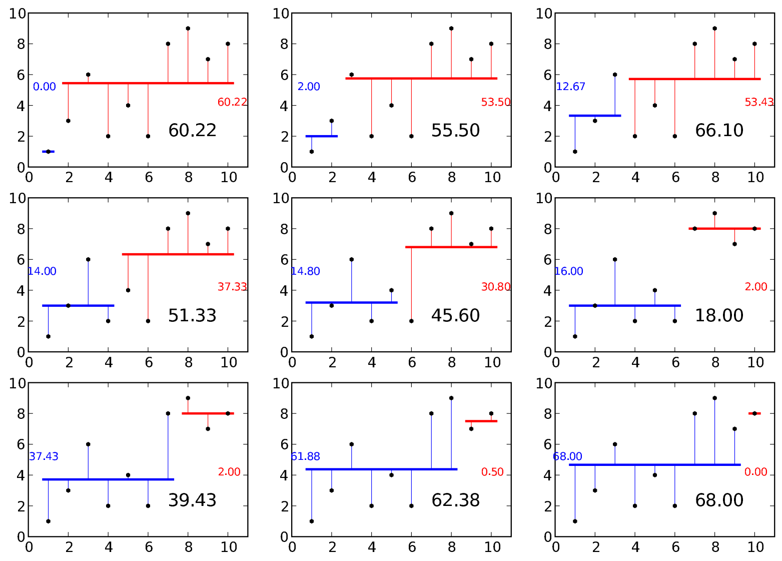
Using the definition of expectation and the inequality proved above, we get

$$E[e^{tY_1}] = pe^t + (1-p) = 1 + p(e^t - 1) \leq e^{p(e^t - 1)}.$$

By choosing $t = \ln(1+\delta)$ we get

$$\Pr(X \geq (1+\delta)np) \leq e^{-\ln(1+\delta)(1+\delta)np} e^{p(1+\delta-1)n} = (1+\delta)^{-(1+\delta)np} e^{\delta np} = \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{np}.$$

H5 / Problem 4.
By trying all possible choices it can be seen that the smallest error is reached when the border of segments is between sixth and seventh point:



H5 / Problem 5.
Let us write $f(x) = \sum_{i=1}^{n} (y_i - x)^2$ and derivate f:

$$f'(x) = 2nx - 2\sum_{i=1}^{n} y_i.$$

The derivate is zero, if $x = (1/n)\sum_i y_i$, that is, the mean of items, it is negative if x is smaller, and it is positive, if x is bigger. This the minimum point of the function f.

Let us write $g(x) = \sum_{i=1}^{n} |y_i - x|$. It can be assumed that numbers y_i are sorted: $y_1 \leq y_2 \leq \cdots \leq y_n$. For any indices $i < j$ it holds

$$|y_i - x| + |y_j - x| = \begin{cases} y_j - y_i, & y_i \leq x \leq y_j, \\ y_j - y_i + 2(y_i - x), & x < y_i, \\ y_j - y_i + 2(x - y_j), & y_j < x. \end{cases}$$

Hence, if $n = 2$, whichever point between items y_1 and y_2 minimizes the error. If n is even, it can be written

$$g(x) = \sum_{i=1}^{n/2} h_i(x),$$

where each of the functions $h_i(x) = |y_i - x| + |y_{n+1-i} - x|$ is minimized, when $y_{n/2} \leq x \leq y_{n/2+1}$. If h is even odd, the function

$$g(x) = \sum_{i=1}^{(n-1)/2} h_i(x) + |y_{(n+1)/2} - x|,$$

is minimized, when $x = y_{(n+1)/2}$. In any case, we can choose the median of points y_i.