L5 -- min-hash
[Jeff Phillips - Utah - Data Mining]

Jaccard Similarity
  A = {0,1,2,5,6}
  B = {0,2,3,5,7,9}
How similar are A,B?

JS(A,B) = |A cap B| / |A cup B|

        = |{0,2,5}|/|{0,1,2,3,5,6,7,9}|
        = 3/8

Matrix Representation:
S1 = {1, 2, 5}
S2 = {3}
S3 = {2, 3, 4, 6}
S4 = {1, 4, 6}

Jac(S1, S3) = |S1 cap S3| / |S1 cup S3|
            = |{2}| / |{1,2,3,4,5,6}|
            = 1/6

Element	S1	S2	S3	S4
1	1	0	0	1
2	1	0	1	0
3	0	1	1	0
4	0	0	1	1
5	1	0	0	0
6	0	0	1	1

Mostly sparse == mostly 0s.
  - 90% 0s
  - size n^2, then n^{.9} are 0s.
very wasteful representation (but convenient to think about).

Idea 1 Hash-Clustering

random function hash h:{1,2,3,4,5,6} -> {A,B,C}
  example [1,2,3,4,5,6] -> [A,B,B,C,A,A]

Element | S1 | S2 | S3 | S4

A	1	0	1	1
B	1	1	1	0
C	0	0	1	1

Jac(S1,S2) = 0   Jac(h(S1),h(S2)) = 1/2
Jac(S1,S3) = 2/6 Jac(h(S1),h(S3)) = 2/3
Jac(S1,S4) = 1/5 Jac(h(S1),h(S4)) = 1/3
Jac(S2,S3) = 1/4 Jac(h(S2),h(S3)) = 1/3
Jac(S2,S4) = 0   Jac(h(S2),h(S4)) = 0
Jac(S3,S4) = 2/5 Jac(h(S3),h(S4)) = 2/3

similarity generally increases.
if intersect -> still intersect
OK when want to study frequent items and have many infrequent items (see more in Summaries)

Idea 2 Min-Hashing

Step 1. Randomly permute items:

Element	S1	S2	S3	S4
2	1	0	1	0
5	1	0	0	0
6	0	0	1	1
1	1	0	0	1
4	0	0	1	1
3	0	1	1	0

Step 2. record first 1 in each column

m(S1) : 2
m(S2) : 3
m(S3) : 2
m(S4) : 6

Step 3. Pr[m(Si) = m(Sj)] = Jac(Si,Sj)

Proof: 3 types of rows
X : 1 in both column   --> count x
Y : 1 in one column, 0 in other --> count y
Z : 0 in both columns --> count z
Jac(Si,Sj) = x/(x+y)
and z >> x,y (mostly empty)

ignore type Z.

Let row r is the min of {m(Si),m(Sj)}
 it is either type X or Y.
 it is X w.p. x/(x+y)
Which is the only case that m(Si) = m(Sj) otherwise either Si or Sj has 1, but not both.
 QED

Only gives 1 or 0. But has right expectation.

Lets consider k different random permutations
 {m_1, m_2, ..., m_k}
And consider k random variables
 {X_1, X_2, ..., X_k}
 {Y_1, Y_2, ..., Y_k}
where
   X_l = 1 if m_l(Si) = m_l(Sj)
   X_l = 0 otherwise
and Y_l = X_l - Jac(Si,Sj)

Let M = (1/k) sum_{l=1}^k Y_l
Let A = (1/k) sum_{l=1}^k X_l
Note -1 < X_l < 1 and E[M] = 0

With k = (2/eps^2) ln (2/delta)
then
 Pr[|Jac(Si,Sj) - A| < eps] > 1-delta

Chernoff-Hoeffding Inequality

Too slow:
 - Still construct full matrix.
 - permute k times!

Fast Minhash algorithm.

Make 1 pass on data. Maintain k hash functions:
 h_i : [N] -> [N] (at random)

Set k counters {c_j} set to infty.

```
for each i in [N]
 if (S(i) = 1)
  for each j in [k]
```

```
   if (h_j(i) < c_j)
    c_j := h_j(i)
```

Now m_j(S) = c_j

Space now is O(k*N) where there are N documents .