Tutorial 3: Simple recommender system and clustering

1. (a) Euclidean distances:

	Guardian	Times	Telegraph	Independent	Steve
Guardian					$\sqrt{23}$
Times	$\sqrt{13}$				$\sqrt{6}$
Telegraph	$\sqrt{41}$	$\sqrt{8}$			$\sqrt{10}$
Independent	$\sqrt{12}$	$\sqrt{5}$	$\sqrt{17}$		$\sqrt{7}$

Closest pair: Independent/Times

Furthest pair: Guardian/Telegraph

Closest to Steve: Times

(b) We can use the following to convert the Euclidean distance (a measure of dissimilarity) to a measure of similarity:

$$\mathrm{sim}(\mathbf{x}, \mathbf{y}) = \frac{1}{1 + r_2(\mathbf{x}, \mathbf{y})} .$$

This ad hoc measure of similarity is just one possible choice. The good points are that distance of 0 has similarity 1, and distance of infinity has similarity 0. Bad points are that it is does not normalize for mean or variance (i.e., does not take account of a critic who gives consistently higher ratings). Another possible measure, that has been used in practice, is the Pearson correlation.

We can use the similarity to estimate the score $\mathrm{sc}_u(z)$ for item z for a new user u, by summing over the set of C critics:

$$\mathrm{sc}_u(z) = \frac{1}{\sum_{c=1}^{C} \mathrm{sim}(\mathbf{x}_u, \mathbf{x}_c)} \sum_{c=1}^{C} \mathrm{sim}(\mathbf{x}_u, \mathbf{x}_c) \cdot \mathrm{sc}_c(z) .$$

Putting all the things we need to compute in a table:

		Mary Goes First		*Well*		*Three Women*	
	Similarity	Score	Sim.Score	Score	Sim.Score	Score	Sim.Score
Guardian	0.17	6	1.02	2	0.34	4	0.68
Times	0.29	6	1.74	6	1.74	8	2.32
Telegraph	0.24	6	1.44	2	0.48	9	2.16
Independent	0.27	3	0.81	3	0.81	6	1.62
Sum	0.97		5.01		3.37		6.78
Est. Score			5.16		3.47		6.99

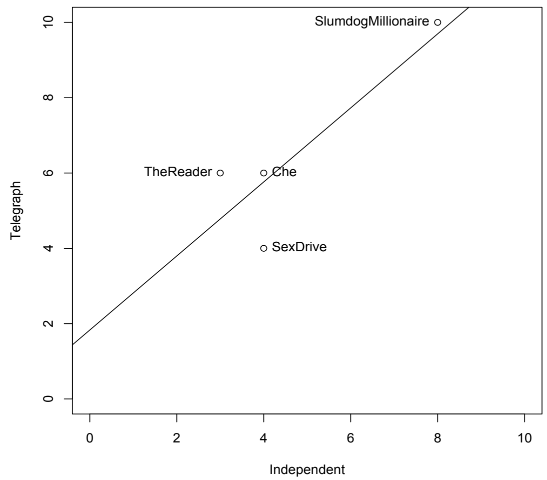
So the recommendation would be *Three Women* with an estimated rating of about 7.

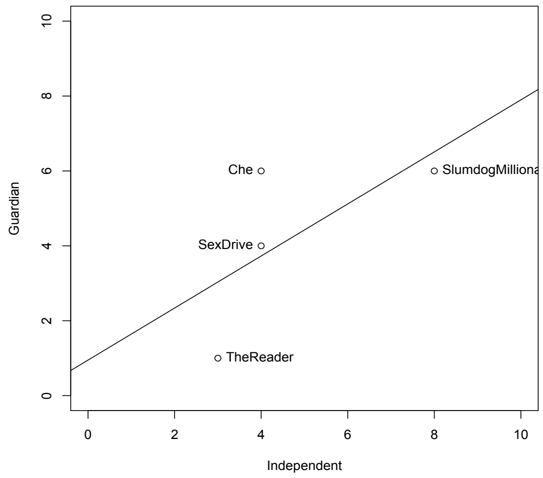
(c) Distance from *Che*:

Slumdog Millionaire	$\sqrt{36} = 6$
Sex Drive	$\sqrt{12} \simeq 3.5$
The Reader	$\sqrt{30} \simeq 5.5$

So on this limited recommender system, *Sex Drive* would be recommended as the closest to *Che*. Is this a good recommendation? You might like to discuss the limitations of the system in the light of this recommendation: limited number of movies; limited number of raters; only taking into account ratings (not genre, etc.)

(d) To get a feel for correlations plot a couple on the board: try Independent vs Telegraph and Independent vs Guardian. Although Independent has a smaller Euclidean distance to Guardian than to Telegraph, it is better correlated with Telegraph than Guardian. One reason for this is that Telegraph has a much higher mean score (6.5) than Independent (4.75).





To compute the Pearson correlation coefficient:

$$\rho_{xy} = \frac{1}{N-1}\sum_{n=1}^{N}\frac{(x_n - m_x)}{s_x}\cdot\frac{(y_n - m_y)}{s_y},$$

where m_x and m_y are the sample means and s_x and s_y are the sample standard deviations:

$$m_x = \frac{1}{N}\sum_{n=1}^{N} x_n \quad (1)$$

$$s_x = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(x_n - m_x)^2}\,. \quad (2)$$

It's a little bit tedious to do this by hand, better to write a small program to do it (or to use a library call in Matlab or R). There are ways to compute the sd more efficiently, you might like to discuss better ways to do it.

Tutorial 3 / Learning and Data Tutorial – Solutions Informatics 2B (HS v1.1)

```
# Simple python function to compute Pearson correlation
def corr(x,y):
  nx = len(x)
  ny = len(y)
  if nx != ny:
      return 0
  if nx == 0:
      return 0
  N = float(nx)

  # compute mean of each vector
  meanx = sum(x) / N
  meany = sum(y) / N

  # compute standard deviation of each vector
  sdx = math.sqrt(sum([(a-meanx)*(a-meanx) for a in x])/(N-1) )
  sdy = math.sqrt(sum([(a-meany)*(a-meany) for a in y])/(N-1) )

  # normalise vector elements to zero mean and unit variance
  normx = [(a-meanx)/sdx for a in x]
  normy = [(a-meany)/sdy for a in y]

  # return the Pearson correlation coefficient
  return sum([normx[i]*normy[i] for i in range(nx)])/(N-1)
```

The computed correlations are given below:

	Guardian	Times	Telegraph	Independent
Guardian	1	0.77	0.42	0.65
Times	0.77	1	0.90	0.90
Telegraph	0.42	0.90	1	0.87
Independent	0.65	0.90	0.87	1

The largest correlations (similarities) are between Telegraph and Times, and between Independent and Times.

Optional discussion:

In the above expression for the sample correlation coefficient, we use an unbiased estimator for the variance (which is still biased for the standard deviation): divide by $(N-1)$ rather than by N:

$$s_{N-1}^2 = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - m)^2\,.$$

You might like to discuss this expression, informally. The following, taken from David MacKay's book *Information Theory, Inference, and Learning Algorithms* (see `http://www.inference.phy.cam.ac.uk/mackay/itila/book.html`), gives an intuitive explanation for why s_N^2 gives an under-estimate of the true variance. Let the true mean be represented by μ and the true variance be represented by σ^2:

i. The data points that we observe come from a distribution centred on the true mean μ, with dispersion σ^2.

ii. The sample mean m is in unlikely to equal the true mean (particularly if the sample size is small).

iii. The sample mean is that point m which minimizes the sum of squared deviations of the data points from m.

iv. Any other value for the sample mean (including μ) will have a larger value of the sum-squared deviation than m.

v. Since the sample variance is estimated as the average sum-squared deviation from the sample mean, s_N^2 will be smaller than the average sum-squared deviation from the true mean.

2. (a) For simplicity's sake, we can assume that the samples are normalised in advance so that $m_x = m_y = 0$.

$$s_x = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N} x_n^2}, \quad s_y = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N} y_n^2}$$

$$r = \frac{1}{N-1}\sum_{n=1}^{N}\frac{x_n}{s_x}\frac{y_n}{s_y} = \frac{1}{N-1}\frac{1}{s_x s_y}\sum_{n=1}^{N} x_n y_n = \frac{1}{\sqrt{\sum_{n=1}^{N} x_n^2}\sqrt{\sum_{n=1}^{N} y_n^2}}\sum_{n=1}^{N} x_n y_n$$

$$= \frac{\mathbf{x}\cdot\mathbf{y}}{||\mathbf{x}||\,||\mathbf{y}||}$$

where $\mathbf{x}\cdot\mathbf{y}$ is the dot product between $\mathbf{x}$ and $\mathbf{y}$.

Since $\mathbf{x}\cdot\mathbf{y} = ||\mathbf{x}||\,||\mathbf{y}||\cos(\theta)$, where θ is the angle between the two vectors, $-1 \le r \le 1$.

(b) Good examples can be found in the Wikipedia's page: `http://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient`

3. Best to do this by plotting points on a graph.

Iter 1:

- $(1, 1)$: $(1, 1)$ $(4, 4)$ $(5, 1)$ $(7, 1)$
- $(7, 10)$: $(7, 4)$ $(7, 10)$

Cluster centres re-estimated to $(17/4, 7/4)$ and $(7, 7)$

Iter 2:

- $(17/4, 7/4)$: $(1, 1)$ $(4, 4)$ $(5, 1)$ $(7, 1)$
- $(7, 7)$: $(7, 4)$ $(7, 10)$

Iter 3 does not change the centres. Converged.

4. Boundary between clusters 1 and 2 is midline between $(0, 0)$ and $(0, 4)$ which is $y = 2$

Boundary between clusters 1 and 3 is midline between $(0, 0)$ and $(4, 2)$ which is $y = -2x + 5$ (points on line e.g. $(2, 1)$ $(3, -1)$ $(1, 3)$)

Boundary between clusters 2 and 3 is midline between $(0, 4)$ and $(4, 2)$ which is $y = 2x - 1$ (points on line e.g. $(2, 3)$ $(3, 5)$ $(1, 1)$)

These intersect at $(3/2, 2)$ and the boundaries are given by:

Tutorial 3 / Learning and Data Tutorial – Solutions Informatics 2B (HS v1.1)

- $y = 2$ when $x < 3/2$
- $y = -2x + 5$ when $x > 3/2$
- $y = 2x - 1$ when $x > 3/2$

Sketch the Voronoi tessellation — key point is intersection at $(3/2, 2)$ and division into 3 regions