Tutorial 1: Simple recommender system and clustering

1. (a) Euclidean distances:

<table>
<thead>
<tr>
<th></th>
<th>Guardian</th>
<th>Times</th>
<th>Telegraph</th>
<th>Independent</th>
<th>Steve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guardian</td>
<td>$\sqrt{23}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Times</td>
<td>$\sqrt{13}$</td>
<td>$\sqrt{6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Telegraph</td>
<td>$\sqrt{41}$</td>
<td>$\sqrt{8}$</td>
<td>$\sqrt{10}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Independent</td>
<td>$\sqrt{12}$</td>
<td>$\sqrt{5}$</td>
<td>$\sqrt{17}$</td>
<td>$\sqrt{7}$</td>
<td></td>
</tr>
</tbody>
</table>

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:

$$\text{sim}(x, y) = \frac{1}{1 + r^2(x, y)}.$$  

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 $sc_u(z)$ for item $z$ for a new user $u$, by summing over the set of $C$ critics:

$$sc_u(z) = \frac{1}{\sum_{c=1}^{C} \text{sim}(x_u, x_c)} \sum_{c=1}^{C} \text{sim}(x_u, x_c) \cdot sc_c(z).$$

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

<table>
<thead>
<tr>
<th></th>
<th>Guardian</th>
<th>Times</th>
<th>Telegraph</th>
<th>Independent</th>
<th>Steve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary Goes First</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Similarity</td>
<td>Score</td>
<td>Sim.Score</td>
<td>Score</td>
<td>Sim.Score</td>
<td>Score</td>
</tr>
<tr>
<td>Guardian</td>
<td>0.17</td>
<td>6</td>
<td>1.02</td>
<td>2</td>
<td>0.34</td>
</tr>
<tr>
<td>Times</td>
<td>0.29</td>
<td>6</td>
<td>1.74</td>
<td>6</td>
<td>1.74</td>
</tr>
<tr>
<td>Telegraph</td>
<td>0.24</td>
<td>6</td>
<td>1.44</td>
<td>2</td>
<td>0.48</td>
</tr>
<tr>
<td>Independent</td>
<td>0.27</td>
<td>3</td>
<td>0.81</td>
<td>3</td>
<td>0.81</td>
</tr>
<tr>
<td>Sum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est. Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.97</td>
<td>5.01</td>
<td>3.37</td>
<td>6.78</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.16</td>
<td>3.47</td>
<td>6.99</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 $\sqrt{12} \approx 3.5$
The Reader $\sqrt{30} \approx 5.5$

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:

$$\rho_{xy} = \frac{1}{N - 1} \sum_{n=1}^{N} \frac{(x_n - \bar{x})}{s_x} \frac{(y_n - \bar{y})}{s_y},$$

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

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

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

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.
# 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:

<table>
<thead>
<tr>
<th></th>
<th>Guardian</th>
<th>Times</th>
<th>Telegraph</th>
<th>Independent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guardian</td>
<td>1</td>
<td>0.77</td>
<td>0.42</td>
<td>0.65</td>
</tr>
<tr>
<td>Times</td>
<td>0.77</td>
<td>1</td>
<td>0.90</td>
<td>0.90</td>
</tr>
<tr>
<td>Telegraph</td>
<td>0.42</td>
<td>0.90</td>
<td>1</td>
<td>0.87</td>
</tr>
<tr>
<td>Independent</td>
<td>0.65</td>
<td>0.90</td>
<td>0.87</td>
<td>1</td>
</tr>
</tbody>
</table>

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 \(N\):

\[
s^2_{N-1} = \frac{1}{N-1} \sum_{n=1}^{N} (x_n - m)^2.
\]

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](http://www.inference.phy.cam.ac.uk/mackay/itila/book.html)), gives an intuitive explanation for why \(s^2_{N-1}\) gives an under-estimate of the true variance. Let the true mean be represented by \(\mu\) and the true variance be represented by \(\sigma^2\):
i. The data points that we observe come from a distribution centred on the true mean $\mu$, with dispersion $\sigma^2$.

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 $\mu$) 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, $s^2$ 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 $\bar{x} = \bar{y} = 0$.

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

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

$$= \frac{x \cdot y}{\|x\| \|y\|} = \cos \theta$$

where $x \cdot y$ is the dot product between $x$ and $y$, and $x \cdot y = \|x\| \|y\| \cos \theta$, where $\theta$ is the angle between the two vectors. Thus, $-1 \leq r \leq 1$.

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

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

<table>
<thead>
<tr>
<th>Iter 1: Centroids</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 1)</td>
<td>(1, 1), (4, 4), (5, 1), (7, 1)</td>
</tr>
<tr>
<td>(7, 10)</td>
<td>(7, 4), (7, 10)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Iter 2: Centroids</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>(17/4, 7/4)</td>
<td>(1, 1), (4, 4), (5, 1), (7, 1)</td>
</tr>
<tr>
<td>(7, 7)</td>
<td>(7, 4), (7, 10)</td>
</tr>
</tbody>
</table>

Iter 3 does not change the centres. Converged.

4. (a) See Figure 4(a) below. Boundary between $x_1$ and $x_2$ is the midline (perpendicular bisector) between (0, 0) and (0, 4), which is $y = 2$.

Boundary between $x_1$ and $x_3$ is the midline between (0, 0) and (2, 2), which is $y = -x + 2$.

Boundary between $x_2$ and $x_1$ is the midline between (0, 4) and (2, 2), which is $y = x + 2$.

These intersect at (0, 2) and the boundaries are given by:

- $y = 2$ when $x < 0$
• \( y = -x + 2 \) when \( x > 0 \)
• \( y = x + 2 \) when \( x > 0 \)

The key points for the sketch in Figure 4(a) are that there is an intersection at \((0, 2)\) and the space is divided into 3 regions.

(b) See Figure 4 (b)
(c) See Figure 4 (c), where \( C_1 \) region is shown in red, \( C_2 \) region in blue, and \( C_3 \) region in green.