Course: Statistical methods for machine learning

March 10, 2010

Case 3: Unsupervised learning

This assignment is based on parts of chapters 2, 9, and 12 from C. M. Bishop *Pattern Recognition and Machine Learning* as well as parts of chapter 14 from Hastie et al. *The Elements of Statistical Learning*. The goal is for you to get familiar with the unsupervised learning task and to get practical experience using a number of methods such that after this assignment you will be able to use and interpret the results of the methods in other contexts.

You have to pass this and the following mandatory assignments in order to be eligible for the exam of this course. There are in total 3 mandatory pass/fail assignments on this course, which can be solved individually or in groups of no more than 3 participants. The course will end with a larger exam assignment which must be solved individually and is graded (7-point scale).

The deadline for this assignment is Tuesday 23/3 at 23:55. You must submit your solution electronically via the Absalon home page. Go to the assignments list and choose this assignment and upload your solution prior to the deadline. If you choose to work in groups on this assignment you should only upload one solution, but remember to include the names of all participants both in the solution as well as in Absalon when you submit the solution. If you do not pass the assignment (after having made a serious attempt), you will get a second chance of submitting a new solution.

A solution consist of:

- Your solution code (Matlab / R / Python scripts) with comments about the major steps involved in each Question (see below).
- Notes detailing your answer to the non-programming questions, which may include graphs and tables if needed (Max 2 page text).

1 Working with High Dimensional Data

In many practical problems, we are given high dimensional observations, i.e. a set of vectors in $\mathbf{R}^D$ where D is a large number. Once the dimensionality of our observations exceed three, we are faced with the problem of how to visualise the observations. We also have the problem that many machine learning algorithms do not provide satisfactory results when working on high dimensional data. This is a symptom of *the Curse of Dimensionality* as described in section 1.4 in Bishops book.

One practical approach to both these problems is to reduce the dimensionality of the observations. The classical approach to dimensionality reduction is PCA. This approach projects the data onto a linear subspace of $\mathbf{R}^D$, so as much as possible of the variance of the data is kept. The basis vectors of the subspace are called the *Principal Components* of the data. PCA explicitly provides us with information about how much variance the individual components captures. This information helps us select how many components we wish to utilise. A common choice in machine learning is to include as many components as necessary to capture 95% of the total variation in the observations. The total variation of PCA is simply defined as the sum of all the eigenvalues λ_i of the PCA, i.e. the sum of the variances along all principal components, $\sum_{i=1}^{D} \lambda_i$.

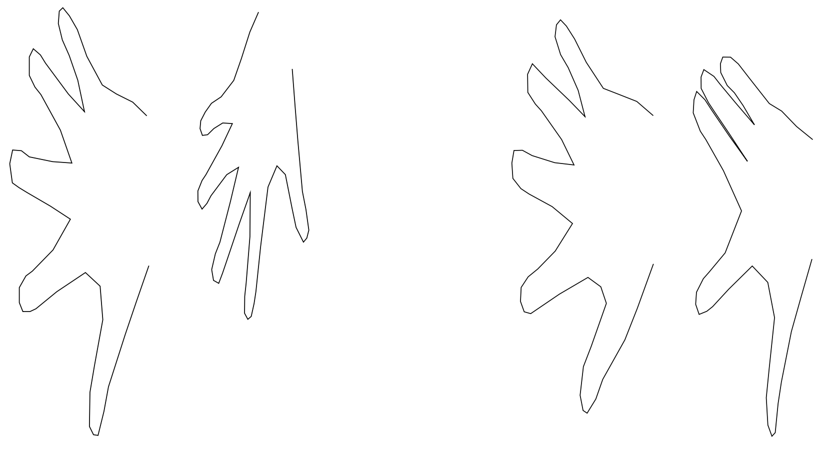

Figure 1: Outline of hands. (a) Two hands with the same shape. One is a translated, scaled and rotated copy of the other. (b) Two hands with different shapes.

1.1 Shapes

Shape Analysis is not surprisingly the act of analysing shapes. For this definition to make sense, we need to define the concept of *shape*. One common definition is the one given by D. G. Kendall [1]:

> "*Shape* is all the geometrical information that remains when, *location*, *scale* and *rotational effects* are filtered out from an object."

Fig. 1a shows the outline of two hands that have the same shape, and Fig. 1b shows two hands with different shapes.

Mathematically, a two dimensional shape is represented as a curve:

$$f(t) : \mathbf{R} \rightarrow \mathbf{R}^2, \quad f(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}. \tag{1}$$

From a practical point of view, this representation is not of much use as it is not directly usable on a computer. Several solutions to the problem exists. Here, we will use a set of n points on the curve to represent shapes. This simply means that we represent a shape as a vector $\mathbf{s}$ containing the x and y coordinates of the points:

$$\mathbf{s} = (x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)^T, \tag{2}$$

where n is the number of points. We see that one shape corresponds to a point in $\mathbf{R}^{2n}$. Fig. 2 shows the point representation of one of the hands from Fig. 1a.

1.2 Analysis of Shapes

In the previous section, we noted that one shape corresponds to one point in $\mathbf{R}^{2n}$. So, to analyse a set of shapes, we have to analyse a set of points in $\mathbf{R}^{2n}$. Since n is usually quite large, we have to analyse high dimensional data. In this exercise $n = 56$, which means we will be analysing data in a 112 dimensional space.

According to the definition of *shape* we should, however, remove any information about translation, rotation and scale from the data before doing any analysis. From a practical point of view, this means we have to translate, rotate and scale our set of shapes, so they are as similar as possible. This process is known as *alignment*. For two dimensional shapes, this process removes

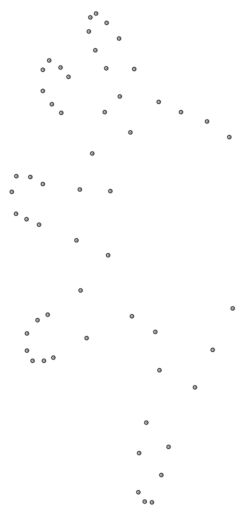

Figure 2: The point representation of the shape of one of the hands shown in Fig. 1a.

$2 + 1 + 1 = 4$ degrees of freedom from our data set. Hence, even though our observations are in $\mathbf{R}^{2n}$, we know that aligned data can have no more than $2n - 4$ degrees of freedom. However, since we are studying shapes of a specific type of object (in our case a hand) the actual number of degrees of freedom are much lower than this. *Shape analysis* is the process of finding these degrees of freedom and quantifying their relative importance. If you would like to know more about shape analysis, Stegmann and Gomoz have written a good introduction [1] to the topic. A good understanding of shape analysis is, however, not needed to solve this part of the assignment. Applications of PCA is found in other areas than shape analysis and is especially useful whenever the data have fewer degrees of freedom than the dimensionality of the representation.

1.3 Shape Variations

In this assignment you are given a data set containing the x and y coordinates of 56 points placed along the outline of 40 hands. This corresponds to 40 points in $\mathbf{R}^{112}$. The shapes have already been aligned, so you do not need to do this. Your objective is instead to compute and visualise the variations of these hands.

The data are available in the file `hands.txt`. To load this data in Matlab you can write

```
data = load ('hands.txt');
```

and in R you can write

```
data <- matrix (t (scan ("hands.txt")), ncol=40);
```

To plot the outline of a hand, you simply plot the individual x and y coordinates. In Matlab this is done by writing

```
plot (data (1:56, k), data (57:end, k))
```

and in R

```
plot (data [1:56, k], data [57:112, k], "l")
```

Here you should substitute `k` with the index of the hand you wish to plot.

Question 3.1: Visualise the Variations of Individual Points

Each shape consists of 56 two dimensional points. As a first step in visualising the variations of the shapes, we will visualise how much these points varies in the set of shapes.

To do this we first focus on the first point (x_1, y_1) on all the hands. Compute the mean vector and covariance matrix of these points. Use these to visualise the variation of the first point by drawing iso-probability density curves. Use the `plot_normal_dist` function for this. Repeat this procedure for all 56 points on the shapes. For clarity, you should plot all mean values and covariances on the same figure.

Question 3.2: Principal Component Analysis (PCA)

Next, we analyse the variance of all points in the shape vectors using PCA. To do this, you have to compute the mean vector and covariance matrix of the 40 vectors in $\mathbf{R}^{112}$. Then, compute the eigenvectors and -values of the covariance matrix (hint: You can use the function `eig` in Matlab or `eigen` in R. Alternatively you can use the Singular Value Decomposition (SVD) approach as discussed in the lecture).

From this information, you must answer the following questions:

- What does the mean vector represent? Try to visualise the mean vector by plotting the 2D points that it represents (hint: See Sec. 1.3).
- How many components are needed to capture 95% of the total variations in the data set? How many are needed to capture 99%? In general we can consider components with large eigenvalues (variance) as being important (carrying relevant information about the data) and components with small eigenvalues (variance) as being noise. In our hand data set how many principal components are needed to capture the most relevant parts of the data? How many components can then be considered to be noise?
- Lets try to visualize the first principal components, that is the component with the largest eigenvalue. If μ is the mean vector and κ is the component vector normalized such that $\|\kappa\| = 1$, try plotting μ along with $\mu + \alpha\sqrt{\lambda}\kappa$ for $\alpha \approx \pm 2$. Here λ denotes the eigenvalue of the component κ, i.e. the variance of the component, and $\sqrt{\lambda}$ is the standard deviation or units of the component. Remember that, just as above, both μ and $\mu + \alpha\sqrt{\lambda}\kappa$ represents hand shapes and can be plotted by plotting the individual 2D points (hint: See Sec. 1.3). Which kind of shape variation does the first component seem to capture?
- How does this analysis compare to the work you did in question 3.1? What did you gain by doing the principal component analysis?

Question 3.3: Outlier Detection

The last step is to find "odd looking" hands. That is, find the shapes that look the least like the rest of the shapes. This is also known as *outlier detection.* To do this, plot the data in the coordinate system given by the two principal components with the largest eigenvalues (largest variance). Let us call these components κ_1 and κ_2 and lets assume that they are column vectors $\kappa_1, \kappa_2 \in \mathbf{R}^{112}$, normalized to unit length $\|\kappa_1\| = \|\kappa_2\| = 1$ and with corresponding eigenvalues $\lambda_1, \lambda_2 \in \mathbf{R}$. The projection of a shape data point $\mathbf{s} \in \mathbf{R}^{112}$ into the coordinate system formed by κ_1 and κ_2 is given by

$$u = \frac{1}{\sqrt{\lambda_1}}\kappa_1^T(\mathbf{s} - \mu) \tag{3}$$

$$v = \frac{1}{\sqrt{\lambda_2}}\kappa_2^T(\mathbf{s} - \mu) \quad , \tag{4}$$

where $\mu \in \mathbf{R}^{112}$ denotes the mean vector and (u, v) denotes the resulting coordinate in the κ_1 and κ_2 coordinate system.

In this coordinate system, try to locate (visually or by computing the distance from the origin of the coordinate system) the point (shape) that are the farthest from the rest of the points. Which shape does this point correspond to? To answer this simply plot the 2D point representation of the outlier shape (hint: See Sec. 1.3).

2 Analysis of human tumor micro-array data

We will explore the ability of the unsupervised techniques to give us a better understanding of a gene expression dataset. No prior knowledge about gene expression data is required. It is enough to understand that we have a matrix of normalized mRNA expression levels[1] with each column representing an experiment. There are therefore several thousand rows representing individual genes, and tens of columns representing samples: in the particular example in `nci_60.txt` there are 6830 genes (rows) and 59 samples (columns). We call the row vectors expression profiles. It is the convention in the field to show the data as a heat map, ranging from green (negative) to red (positive). The samples are 59 cancer samples from different patients. The somewhat inaccurate classification[2] of the tumors is given as the first row in `nci_60.txt`. We will not use these labels in the learning process but only as a post hoc test of our findings. The questions below will take you through the standard data analysis steps for micro-arrays. The different analysis steps are aimed at answering questions like:

1. What samples are similar? Hopefully similar/same conditions in different patients share characteristics in their expression profile. But it could also be that different but related cancers share characteristics or one condition divide into two or more subtypes.
2. What genes are similar? Co-variation between genes indicate that they take part in the same cellular process.
3. Are there any genes that are predictive of the condition? This could be interesting for tumor classification and for finding potential drug targets.

Question 3.4: PCA for visualization

Read the data from the text file provided (`nci_60.txt`) and compute SVD. Note in the code (`case34to37`), that for each gene the mean expression value is subtracted from individual values (centering values around 0). Plot the samples using only the 1st and 2nd components. PCA is often used as a fist glance explanatory tool for the, usually, very big and high dimensional gene expression data. What can you say about the cancer/cancer groups after looking at the PCA plot? Note that (some) cancers can be very heterogeneous with variance originating from different sources and that some types are more heterogeneous than others. It is possible that some samples are wrongly annotated, see footnote 2.

Question 3.5: Heatmap and two-way hierarchical clustering

Compute the distance matrix from the data loaded in the previous exercise and use it to run hierarchical clustering. Plot the clustering tree (dendrogram). Observe how the various cancers are clustered, do you see any relationship between results of PCA visualization and hierarchical clustering? To be able to visualize gene expression using a heatmap, we reduce first the number of genes. One way of doing this is to select genes with high variance, so basically we assume that genes, whose expression vary more, convey more information while the changes in expression of low variance genes, can be attributed to technical/experimental noise and not to real biological signal/changes. Can you see any disadvantages of applying such a variance filter? Look at the

[1]There exists a large body of literature on preprocessing and normalization of expression data. Many of these methods are available in the Bioconductor package in R. Normalization and filtering (removal of uninformative genes) have already been performed on this dataset.

[2]see Hastie, Tibshirani and Friedman, The Elements of Statistical Learning 2nd edition, Springer, page 514.

heatmap produced, can you see any advantages of the two-way clustering for analysis of gene expression? Look at the hierarchical clustering of the cancers on top of the heatmap, do you see considerable changes from the hierarchical clustering based on expression of all genes? Would you expect this result?

Question 3.6: K-means for unsupervised classification

Perform K-means clustering to the data, label the samples according to the clusters found and show them on the PCA plot from question 3.4. Try different values of K, how would you choose the "optimal" K. For a chosen K, what is the agreement between the K-means clusters and the cancer type labels? K-means clustering final results depend on initial random clusters. Try to re-run the clustering several times, can you see (considerable) changes in the outcome? You have now run both PCA, K-means and hierarchical clustering on the data set. Have you noticed any reoccurring patterns: do you think some of the cancers might have been wrongly annotated?

Question 3.7: Visualizing low dimensional distributions

Lets consider one-dimensional data first. Use a histogram to estimate the density of the second principal component from question 3.4. Try for different settings of the number of bins and make your best guess in terms of the smoothness of the plot. What happens if the number of bins is too small/large? Now use a kernel density estimator for the same purpose. Try for different settings of the kernel bandwidth and again make your best guess. What happens if the bandwidth is too small/large? Compare your bandwidth guess with the one obtained by automatic selection (see the code for details), are they similar? Which one do you think is the best? Why? In what cases do you think one must prefer kernel density estimators over histograms? what about the opposite?

Lets try now two-dimensional data. We are not going to use histograms in this case because we only have 59 observations available. Use a kernel density estimator on the two principal components from question 3.4 and plot it along with the data. Try again different values of the bandwidths and compare them with the automatic selector. Finally, try a mixture of Gaussians for different number of components. Compare your choice with the automatic selector provided with the function. In what cases do you think one must prefer kernel density estimators over a mixture of Gaussians? what about the opposite? Think about advantages/disadvantages of both methods.

Søren Hauberg, Ricardo Henao, Bogumil Kaczkowski, Kim Steenstrup Pedersen and Ole Winther, March 2010.

References

[1] M. B. Stegmann and D. D. Gomez. A brief introduction to statistical shape analysis, mar 2002. Available at `http://www2.imm.dtu.dk/pubdb/p.php?403`.