Lectures 14: Unsupervised Learning
Introduction

So far we have described supervised learning techniques where the class of each training example is known. The training data appears in pairs as

\[ \{x_1, \omega_1\}, \ldots, \{x_n, \omega_k\} \]

In this lecture we will consider the situation where the class of the training examples is unknown. Thus the training data consists of \(d\)-dimensional samples

\[ \{x_1\}, \ldots, \{x_n\} \]

We will examine two aspects of unsupervised learning.

1. **Data projection/visualisation**: can the data be represented by some small number of features, projecting to low dimensions allows the data to be visualised.

2. **Clustering**: assume that the training data was generated by a set of \(K\) distinct clusters. Clustering techniques should allow us to obtain the positions of these clusters.
Data Projection/Visualisation

In many situations it is useful to be able to view the data that we are trying to classify or model. This helps in selecting the form of distributions to use, or how separable we expect the data to be.

Unfortunately lots of practical situations a large number of features are extracted. Viewing all these features is impractical. One approach is to select pairs of features and plot this reduced data. Rather than selecting pairs of data by hand a systematic approach to selecting interesting directions is useful.

In addition to aiding visualisation, altering the feature vector can help as:

- reducing dimensionality may increase generalisation;
- may remove “noise” dimension;
- may reduce correlations between features.

Two approaches are examined in this course:

- **principal component analysis**;
- **factor analysis**.


**Principal Component Analysis**

PCA is a method of extracting the most “important” features when the class labels are unknown. It relies on the assumption that the direction in which there is most variation in the data is the most discriminatory (contains most class information). Since the variation in the data is described by the covariance matrix, we want the directions in which the variances are the largest.

PCA is closely related to the Karhunen Loeve transform in Signal Processing and the Matrix analysis technique of Singular Value Decomposition.

Let $x_1, \ldots, x_n$ be the original measurements and $z_i, i = 1, \ldots, n$ be the transformed features.

$$z_i = A'x_i$$

- Seek values of $A$ so that the first transformed feature variable $z_{i1}$ has maximum variance (i.e. projection of data $x_i$ in this direction has most variance).

- $z_{i2}$ has most variance in a direction orthogonal to $z_{i1}$, etc

- Overall the total variance retained by the transformation is maximised (corresponds to minimising least squares reconstruction error)
Finding Principal Components

From the definition, the first element of $z_i$ is given by

$$z_{i1} = a_1'x_i$$

To avoid being able to arbitrarily scale the elements of $a_1$, a constraint is introduced that $a_1'a_1 = |a_1|^2 = 1$.

The aim is to maximise the variance of $z_{i1}$.

$$\text{var}(z_{i1}) = \mathbb{E}\{z_{i1}^2\} - \mathbb{E}\{z_{i1}\}^2$$

$$= \mathbb{E}\{a_1'x_ix_i'\} - \mathbb{E}\{a_1'x_i\} \mathbb{E}\{x_i'\} a_1$$

$$= a_1' (\mathbb{E}\{x_ix_i'\} - \mathbb{E}\{x_i\} \mathbb{E}\{x_i'\}) a_1$$

$$= a_1' \Sigma_x a_1$$

where $\Sigma_x$ is the covariance matrix of $x$.

To maximise the variance, maximise $a_1' \Sigma_x a_1$ with the constraint that $a_1'a_1 = 1$ use Lagrange multipliers.

Construct the Lagrangian function $L(a_1, \lambda)$ with the Lagrange multiplier $\lambda$

$$L(a_1, \lambda) = a_1' \Sigma_x a_1 - \lambda(a_1'a_1 - 1)$$
First Principal Component (cont)

Differentiating the Langrangian wrt $a_1$ and equating to zero

$$\frac{\partial L(a_1, \lambda)}{\partial a_1} = 2(\Sigma_xa_1 - \lambda a_1) = 0$$

Hence for a non-trivial solution

$$\Sigma_xa_1 = \lambda a_1$$

i.e. $a_1$ is an eigenvector of $\Sigma_x$, $\lambda$ the corresponding eigenvalue.

Which eigenvalue to choose for $\lambda$?

The eigenvalues can be ordered such that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$$

The variance of the projected data $z_{i1}$ is

$$a_1'\Sigma_xa_1 = \lambda a_1'a_1$$

$$= \lambda$$

Since want the largest variance, choose the eigenvector with largest eigenvalue for the first principal direction.
Second Principal Component

For the second principal direction require that $a_2$ is uncorrelated with $a_1$. This implies that

$$\mathcal{E}\{z_i z_j\} - \mathcal{E}\{z_i\} \mathcal{E}\{z_j\} = 0$$

which is equivalent to

$$a_2' \Sigma_x a_1 = 0$$

and since $a_1$ is an eigenvector of $\Sigma_x$ then $a_2' a_1 = 0$

Again using Lagrange multipliers, maximise

$$\tilde{L}(a_2, \mu, \eta) = a_2' \Sigma_x a_2 - \mu(a_2' a_2 - 1) - \eta(a_2' a_1)$$

where $\mu, \eta$ are the Lagrange multipliers. Differentiating and equating to zero for the maximum:

$$\frac{\partial \tilde{L}(a_2, \mu, \eta)}{\partial a_2} = 2(\Sigma_x a_2 - \mu a_2) - \eta a_1 = 0$$

Pre-multiplying by $a_1'$ gives (noting that $a_1' a_2 = 0$)

$$2a_1' \Sigma_x a_2 - \eta = 0$$

which since $a_1$ is an eigenvector of $\Sigma_x$ then implies that $\eta = 0$. Hence

$$\frac{\partial \tilde{L}(a_2, \mu, \eta)}{\partial a_2} = 2(\Sigma_x a_2 - \mu a_2) = 0$$

Therefore the second principal is also an eigenvector of $\Sigma_x$. Since we are seeking to maximise the remaining variance the largest of the remaining eigenvalues is chosen.
Principal Component Analysis

The argument on the previous slides can be repeated for all the principal components. Hence to find the principal components perform eigenvector analysis of the covariance matrix and order the eigenvalues by size. Thus PCA involves using the standard eigenvector/eigenvalue decomposition

\[ \Sigma_x = U\Lambda U' \]

Assuming that the eigenvectors are ordered, largest first

\[ A = [u_1 \cdots u_p] \]

where \( u_i \) is the \( i^{th} \) eigenvector (column of \( U \)).

Note:

- Since eigenvalues give variance in principal directions, sum of the eigenvalues for the first \( p \) principal components show the proportion of the total variance that is accounted for.

- The principal directions are not scale invariant: the variance of data in a particular direction depends on scaling.

- Often the original measurements are standardized to have a unit variance and zero mean before PCA is applied.

- If all dimensions are retained then the feature vector will simply be decorrelated, \( (\Sigma_z = \Lambda) \).
Least Squares Reconstruction

PCA also has a least squares interpretation. Consider the squared error between the true transformed point, \( \tilde{x} \) and the zero extended point in the PCA determined space

\[
\tilde{z}_i = \begin{bmatrix} z_i \\ 0 \end{bmatrix}
\]

The squared error is (note the data is assumed to be zero mean)

\[
E = \frac{1}{n} \sum_{i=1}^{n} (\tilde{x}_i - \tilde{z}_i)'(\tilde{x}_i - \tilde{z}_i)
\]

Since the error in the \( p \)-dimensions retained must be zero, then

\[
E = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=p+1}^{d} \tilde{x}_{ij}^2 = \sum_{j=p+1}^{d} \lambda_j
\]

Thus by removing dimensions with the smallest eigenvalues the squared error is minimised.
Factor Analysis

Factor analysis may be viewed as a multivariate analysis scheme. The aim is to represent the data in terms of a small number of underlying variables, the factors.

Factor analysis may be described using the following equations

\[ z_i \sim \mathcal{N}(0, I) \]
\[ x_i = Cz_i + w; \quad w \sim \mathcal{N}(0, \Sigma^{(w)}) \]

where \( z_i \) is the \( p \)-dimensional vector of the factors.

As previously mentioned factor analysis may also be viewed as a covariance matrix model. The resultant estimated covariance matrix for \( x \) is

\[ \hat{\Sigma}_x = CC' + \Sigma^{(w)} \]

The factors for sample \( x_i \) are \( z_i \). In contrast to the PCA, where the representative variables were transforms of the observations, here the observations are a transformation (plus noise) of the representative variables.
Estimating the Factors

As the observations are now a linear transform of the factors with observation noise added, there is no unique estimate of the factors. The factor for sample $x_i$, $z$ is known to be Gaussian. From the work on EM

$$\mathcal{E}\{z|x_i\} = C'(CC' + \Sigma^{(w)})^{-1}(x_i - \mu)$$

$$\mathcal{E}\{zz'|x_i\} - \mathcal{E}\{z|x_i\}\mathcal{E}\{z|x_i\}' = I - C'(CC' + \Sigma^{(w)})^{-1}C$$

The simplest approach is to take the mean of the distribution as the estimate of the factor. So

$$z_i = \mathcal{E}\{z|x_i\} = C(CC' + \Sigma^{(w)})^{-1}(x_i - \mu)$$

Note:

- There is no unique solution. Consider any orthonormal matrix $T$. If the factors are transformed by this then

$$\Sigma = CTT'C' + \Sigma^{(w)}$$

$$= CC' + \Sigma^{(w)}$$

so $Tz_i$ is an equally good way of describing the data as $z_i$!

- The factors are not necessarily orthogonal to one another.

- Using ML to estimate the FA, it is invariant to scaling.
Iris Data

The Iris data has already been briefly mentioned. The full data consists of 4 measurements of 3 different forms of iris.

The diagram above shows the classes for 2-dimensional projections of the Iris data.
Clustering

Rather than simply visualising the data it may be useful to group observations together, so that observations in the same group are “similar”. This is the process of clustering.

Thus, without the use of any class labels, we would like to split the data into separate classes.

Consider the dimensions 3 and 4 of the iris data shown above. Without the class labels, these can be easily clustered into two or three clusters.

We would like to formalise how to determine an appropriate clustering of the data.
Clustering (cont)

Assume that a set of cluster centers, $\mu_1, \ldots, \mu_K$, are known. We need to cluster the data points $x_1, \ldots, x_n$.

1. For each point in the training set, $x_i$ compute the nearest cluster, $c_i$, using

$$c_i = \arg \min_k (d(x_i, \mu_k))$$

where $d(x_i, \mu_k)$ is the distance measure being used.

2. Recompute a new set of centers using

$$\mu_k = \frac{\sum_{i:c_i=k} x_i}{\sum_{i:c_i=k} 1}$$

3. Compute the total distortion, $D$,

$$D = \sum_{i=1}^{n} d(x_i, \mu_{c_i})$$

4. If any of the means have changed, or a minimum total distortion not reached, goto step 1.

There are the following issues to address

- How many clusters, $K$ should be used?
- How to get the initial set of cluster means $\mu_1, \ldots, \mu_K$?
- What is the appropriate distance measure to use?
Cluster Number and Initialisation

Determining the number of clusters is similar to selecting the number of levels in a decision tree. It is simple to show that increasing the number of clusters is guaranteed to decrease the total distortion. Thus overly complex numbers of clusters need to be penalised (for example using MDL).

Two standard initialisation schemes are

- **K-Means**: from the available training samples select $K$ examples as the means, then perform clustering algorithm. This is a simple initialisation scheme and (probably) puts more clusters where the points are most dense.

- **Centroid Splitting**: this is an iterative scheme.

  1. Initialise with a single cluster mean $\mu_1$.
  2. For each cluster mean perturb it to obtain two means. This doubles the number of means.
  3. Perform standard clustering as previously described.
  4. If the required number of clusters means is obtained, stop. Else goto (2).

In the form described here there are $2^l$ cluster means at level $l$. This approach is computationally more expensive than K-means initialisation as the clusters must be optimised at each stage.
Distance Measures

A variety of distance measures may be used:

- **Euclidean**: \( d(x, \mu) = \sqrt{((x - \mu)'(x - \mu))} \)
- **Weighted Euclidean**: \( d(x, \mu) = \sqrt{((x - \mu)'\Sigma^{-1}(x - \mu))} \)
- **Mahalanobis**: \( d(x, \mu) = (x - \mu)'\Sigma^{-1}(x - \mu) \)
- **Minkowski metric, order \( s \)**: \( d(x, \mu) = s \sqrt{\left(\sum_j |x_j - \mu_j|^s\right)} \)

The clustering obtained from weighted Euclidean and Mahalanobis will be the same.

Other than the Mahalanobis distance measure these may be shown to be valid metrics. A metric satisfies the following properties

1. \( d(x, y) \geq 0 \) (positivity)
2. \( d(x, y) = d(y, x) \) (symmetry)
3. \( d(x, y) + d(y, z) \geq d(x, z) \) (triangle inequality)

Note:

- The clusters generated depend on the initialisation.
- Training is similar to EM with hard assignments to clusters.