UL04. Feature Transformation

Scan me

What is Feature Transformation?

- The problem of pre-processing a set of features to create a new (more compact) feature set, while retaining as much (relevant/useful) information as possible.
- Feature Selection is a subset of Feature Transformation, where the pre-processing is literally extracting a subset of the features.
- In Feature Transformation, we apply a "linear transformation operator". The goal is to find a matrix P such that we can project the examples into a newer subspace (that is typically smaller than the original subspace) to get new features that are **linear combinations** of the old features.
- Why? We combine features together hoping to eliminate false positives/negatives.

Principal Components Analysis:

- Eigenproblems and Eigenvectors: A (non-zero) vector v of dimension N is an eigenvector of a square $N \times N$ matrix A if and only if it satisfies the linear equation $AV = \lambda V$ where λ is a scalar value. Matrices can be decomposed as follows:
  - Taking a 2×2 real matrix A as an example to be decomposed into a diagonal matrix through multiplication of a non-singular matrix B:
$$A = \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in \mathbb{R}^{2\times 2}$$
  - Then, for some real diagonal matrix a, d:
$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix} B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in \mathbb{R}^{2\times 2}$$
  - Such that:
$$\begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix}$$
$$\begin{cases} \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} a \\ c \end{bmatrix} = \begin{bmatrix} ax \\ cx \end{bmatrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} b \\ d \end{bmatrix} = \begin{bmatrix} by \\ dy \end{bmatrix} \end{cases} \qquad \begin{cases} \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} a \\ c \end{bmatrix} = x \begin{bmatrix} a \\ c \end{bmatrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} b \\ d \end{bmatrix} = y \begin{bmatrix} b \\ d \end{bmatrix} \end{cases}$$
  - Letting:
$$\vec{a} = \begin{bmatrix} a \\ c \end{bmatrix}, \qquad \vec{b} = \begin{bmatrix} b \\ d \end{bmatrix}$$
  - This gives us the vector equations:
$$\begin{cases} A\vec{a} = x\vec{a} \\ A\vec{b} = y\vec{b} \end{cases}$$
  - Where λ represents the two eigenvalues x, y.

- Principal Components Analysis is an example of an eigenproblem which will transform the features set by:
  - Finding the direction (vector) that maximizes variance. The is called the Principal Component.
  - Finding directions that are orthogonal to the Principal Component.
- Each Principal Component has a prescribed eigen value. We can throw away the components with the least eigenvalues as they correspond to the features that matter less in the reconstruction.
- PCA gives the ability to do reconstruction, because it's a linear rotation of the original space that minimizes L2 error by moving N to M dimensions. So, we don't lose information.
- We can center the problem around the origin by subtracting the mean of the data.
- In effect, PCA produces a set of orthogonal Gaussians.
- PCA is a global algorithm that is very fast.

Independent Components Analysis:

- ICA attempts to maximize independence. It tries to find a linear transformation of the feature space, such that each of the individual new features are mutually statistically independent.
  - The mutual information between any two random features equals zero:

$$I\left(y_i; y_j\right) = 0$$

  - The mutual information between the new features set and the old features set is as high as possible:

$$I(Y; X) = \uparrow\uparrow$$

Random Components Analysis:

- Similar to Principal Components Analysis, but instead of generating directions that maximize variance, it generates random directions.
- It captures some of the correlations that works well with classification settings.
- It's faster than PCA and ICA.

Linear Discriminant Analysis:

- Linear Discriminant Analysis finds a projection that discriminates based on the label. That is, it finds projections of features that ultimately align best with the desired output (wrapping function instead of filtering).