CS50 Project — Face Classification vis Principal Component Analysis & K-Nearest Neighbours

Hafsa Akbar

September 2020

Contents

1 Introduction

Let's say that we have a bunch of d-dimensional vectors $\mathbf{x}_i \in \mathbb{R}^d$, $i \in \{1, \ldots, n\}$ (throughout this report we will assume $n \geq d$, i.e. the number of vectors is at least equal to the size of the vectors). We can put all of these vectors side-by-side in the form of a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$. We call this matrix $\mathbf{X}$ as the "data matrix," and the vectors $\mathbf{x}_i$ as a "data samples" or just "samples." We can represent almost any form of data in such a way. For example images can be viewed as vectors $\mathbf{x}_i$ if we stack pixels on top of one-another. Speaking of pictures, we can now imagine a scenario where we can have pictures of dogs and cats (consider this very specific scenario of only 2 categories for now, it will later

become clear how we can extend this concept to multiple categories). We call the categories as "classes." Mathematically, we can represent them as just 1s (for cats) and 0s (for dogs). We can represent this with the symbol $y_i \in \{1, 0\}$, for $i \in \{1, \ldots, n\}$ called a "class labels" or just "labels." You can also stack these class labels together in the form of a vector $\mathbf{y} = [y_1, \ldots, y_n]^\top \in \mathbb{R}^n$ called "label vector".

1.1 Classification

Classification refer es to the process of assigning label y to a sample $\mathbf{x}$. This can be done in a number of different ways, but this project only discusses "discriminative" (or "distance-based") classifiers. Such classifiers simply assign the label y based on which training sample[1] is the closest to the sample $\mathbf{x}$ being classified. The simplest of such algorithms is the K-Nearest Neighbour (K-NN) algorithm which will be discussed in later sections.

1.2 Curse of Dimensionality

Discriminative classification faces several related issues in higher dimension, which are collectively refer ed to as the "Curse of Dimensionality." Samples from high-dimensional distributions get increasingly equi-distant as the number of dimensions increases. Therefore, for a finite number of samples, the concept of "distance" becomes meaningless as the number of dimensions approaches infinity. Distance metrics are hence ineffective for classification in higher dimensions.

1.3 Dimensionality Reduction

The "dimension" of a vector is the number of elements in it. For example, the dimension of all vectors $\mathbf{x}_i \in \mathbb{R}^d$ in the above section is d. Sometimes, we might want to reduce the dimension of a vector but still retain the "essence" of the vector. We can represent these smaller vectors as $\mathbf{z}_i \in \mathbb{R}^p$, $p < d$. And we call these smaller vectors as "representations/features" of $\mathbf{x}_i$. We will discuss later how we can get $\mathbf{z}_i$ given some $\mathbf{x}_i$ (and maybe also y_i). The definition of "essence" can pretty much be anything, but we will discuss 2 main ideas in sections 4 and **??**.

2 Eigen Value Decomposition (EVD)

For any square matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$, we have n "eigenvalues" $\lambda_i \in \mathbb{R}$ and corresponding "eigenvectors" $\mathbf{u}_i \in \mathbb{R}^n$. Eigenvalues and eigenvectors are special in the sense that for eah pair the following equation holds

$$\mathbf{X}\mathbf{u}_i = \lambda_i \mathbf{u}_i \ . \tag{1}$$

[1] Samples which are used to fit the model so that it can be used on other unseen samples, called "test samples."

To get an intuitive understanding for equation (1), watch this video. It turns out that matrix $\mathbf{X}$ can be "decomposed" into the product of (square) matrices of its eigenvalues and eigenvectors

$$\mathbf{X} = \begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_n \end{bmatrix}^{-1} \tag{2}$$

$$= \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1} \tag{3}$$

This is what we call "Eigen Value Decomposition" (EVD). Note that these eigenvalues and eigenvectors in $\mathbf{\Lambda}$ and $\mathbf{U}$ don't have any particular order.

3 Singular Value Decomposition (SVD)

It turns out that any (not necessarily square) matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ can be "decomposed" into the product of three matrices as

$$\mathbf{X} = \overbrace{\begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_d \end{bmatrix}}^{\in \mathbb{R}^{d \times d}} \overbrace{\left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & \mathbf{0}_{d \times (n-d)} \\ & & \sigma_d & \end{array}\right]}^{\in \mathbb{R}^{d \times n}} \overbrace{\begin{bmatrix} \mathbf{v}_1 & \dots & \mathbf{v}_n \end{bmatrix}^\top}^{\in \mathbb{R}^{n \times n}} \tag{4}$$

$$= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \,, \tag{5}$$

where $\mathbf{U}$ and $\mathbf{V}$ are "orthonormal" matrices, which just means that $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_d$, or in other words $\mathbf{U}^\top = \mathbf{U}^{-1}$ and $\mathbf{\Sigma}$ is a (not necessarily square) diagonal matrix. σ_i are called "singular values" and $\mathbf{u}_i \in \mathbb{R}^d$ and $\mathbf{v}_i \in \mathbb{R}^n$ are called left and right "singular vectors" respectively. And, interestingly, there is a descending order to the singular values in $\mathbf{\Sigma}$ and they are also all non-negative, i.e. $\sigma_1 \geq \cdots \geq \sigma_d \geq 0$.

3.1 SVD vs. EVD

There is a very close relationship between EVD and SVD. This can be seen by substituting the SVD of $\mathbf{X}$ from (5) in $\mathbf{X}\mathbf{X}^\top$ and $\mathbf{X}^\top\mathbf{X}$.

$$
\begin{aligned}
\mathbf{X}\mathbf{X}^\top &= \big(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\big)\big(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\big)^\top \\
&= \big(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\big)\big(\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top\big) \\
&= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top \\
&= \mathbf{U}\mathbf{\Sigma}\mathbf{I}_n\mathbf{\Sigma}\mathbf{U}^\top \\
&= \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}\mathbf{U}^\top \\
&= \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^\top \\
&= \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^{-1} \qquad (6)
\end{aligned}
$$

Here, we have used the fact that $(\mathbf{AB})^\top = \mathbf{B}^\top\mathbf{A}^\top$. Equations (6) and (3) are similar. Therefore, $\mathbf{U}$ is the matrix of eigenvectors of $\mathbf{X}\mathbf{X}^\top$ and, similarly, $\mathbf{V}$ is the matrix of eigenvectors of $\mathbf{X}^\top\mathbf{X}$. In each case, $\mathbf{\Sigma}^2$ is the eigenvalue matrix.

4 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a special (and arguably the most famous) dimensionality reduction technique. Let's first go over the steps of PCA and we shall discuss the intuition and and motivation behind it shortly after.

Given a data matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, we wish to reduce the dimension of the data from d to $k < d$ to get smaller versions $\mathbf{z}_i$ of each sample $\mathbf{x}_i$. We carry out the following steps to get the so called "principal components" and "score matrix" of the data.

1. Calculate the "mean" of the data

$$\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = \frac{1}{n}\mathbf{X}\mathbf{1}_n \ .$$

2. Calculate the "centered" data matrix[2]

$$
\begin{aligned}
\bar{\mathbf{X}} &= \mathbf{X} - \boldsymbol{\mu}\otimes\mathbf{1}_n^\top \\
&= \begin{bmatrix}(\mathbf{x}_1 - \boldsymbol{\mu}) & \ldots & (\mathbf{x}_n - \boldsymbol{\mu})\end{bmatrix} \\
&= \begin{bmatrix}\bar{\mathbf{x}}_1 & \ldots & \bar{\mathbf{x}}_n\end{bmatrix} \in \mathbb{R}^{d\times n} \ .
\end{aligned}
$$

   Beyond this point, it would be better for the reader to consider $\mathbf{X}$ and $\bar{\mathbf{X}}$ as essentially the same for ease of understanding.

3. Perform the SVD on $\bar{\mathbf{X}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$.

[2]The "Kronecker product" of matrices $\mathbf{A} = \begin{bmatrix}a_{1,1} & a_{1,2}\\ a_{2,1} & a_{2,2}\end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix}b_{1,1} & b_{1,2}\\ b_{2,1} & b_{2,2}\end{bmatrix}$ is defined as $\mathbf{A}\otimes\mathbf{B} := \begin{bmatrix}a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B}\\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B}\end{bmatrix}$.

4. Throw away all but the first k "singular vectors" (columns) from $\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_d\end{bmatrix}$ to get a smaller matrix $\mathbf{U}_k = \begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_k\end{bmatrix} \in \mathbb{R}^{d\times k}$. This matrix is called the matrix of k "principal components" of $\mathbf{X}$.

5. The matrix $\mathbf{Z} = \mathbf{U}_k^\top \bar{\mathbf{X}} = \begin{bmatrix}\mathbf{z}_1 & \dots & \mathbf{z}_n\end{bmatrix} \in \mathbb{R}^{k\times n}$ is called the "score matrix" of $\mathbf{X}$, where each "score vector" $\mathbf{z}_i \in \mathbb{R}^k$ is a low-dimensional version of the corresponding data sample $\mathbf{x}_i \in \mathbb{R}^d$.

One interesting thing to note here is that given $\mathbf{Z}$ and $\mathbf{U}_k$ that were obtained from PCA, we can "reconstruct" an "approximate" version of the original (centered) data matrix $\bar{\mathbf{X}}$ as $\widehat{\bar{\mathbf{X}}} = \mathbf{U}_k\mathbf{Z} = \mathbf{U}_k\mathbf{U}_k^\top\bar{\mathbf{X}} = \begin{bmatrix}\widehat{\bar{\mathbf{x}}}_1 & \dots & \widehat{\bar{\mathbf{x}}}_n\end{bmatrix} \in \mathbb{R}^{d\times n}$. These "approximations" $\widehat{\bar{\mathbf{x}}}_i$ of samples $\bar{\mathbf{x}}_i$ are also called the "projections" of $\bar{\mathbf{x}}_i$ onto the "space spanned by the principal components." For a new, previously unseen sample $\mathbf{x}_{\text{new}}$, we perform only steps 2 and 5 to get $\mathbf{z}_{\text{new}}$.

Now that we have discussed the precise steps of PCA, we can discuss the intuition behind it.

4.1 Minimum residual view of PCA

Imagine we have the (centered) data samples in $\mathbb{R}^d$ (e.g., $d = 2$). Our goal is to find a different version of this data that instead lies in $\mathbb{R}^k$, which is a lower-dimensional space for $k < d$ (e.g., $k = 1$). Then, PCA can be thought of as "finding" this low-dimensional "linear"[3] space such that, on average, the difference (also called "residual") between any sample $\bar{\mathbf{x}}_i$ and its projection $\widehat{\bar{\mathbf{x}}}_i$ onto the low-dimensional space is as small as possible. The unit vectors (also called "basis vectors" or just "basis") of this low-dimensional space are precisely the principal components $\mathbf{u}_i$ (therefore there are k unit vectors for a k-dimensional space).

4.2 Maximum variance view of PCA

The principal components of the data are vectors along the maximum variance directions of the data. The first principal component lies along the direction of highest variance. The second principal component lies along the direction of second highest variance, and so on. The score vectors $\mathbf{z}_i$ are then the "projections"[4] $\widehat{\bar{\mathbf{x}}}_i$ of the original (centered) samples $\bar{\mathbf{x}}_i$ onto a linear-subsapce with unit-vectors $\mathbf{u}_j$, but viewed in the sup-sapce instead of the original space.

5 K-Nearest Neighbours (K-NN)

K-NN is a classification technique where we store all the training samples $\mathbf{X}$ and their associated labels $\mathbf{y}$ in memory and every new sample $\mathbf{x}_{\text{new}}$ is classified

[3] A linear space means just what it sounds like — something that looks like a line. For example, a line drawn on a piece of paper is a 1-dimensional linear "sub-space" that lies in a higher 2-dimensional space, the paper.

[4] the orthogonal projection of some $\mathbf{a}$ onto a "linear sub-space" is basically the closest point $\hat{\mathbf{a}}$ to $\mathbf{a}$ that lies on that linear sub-space

by taking the labels of K closest samples to $\mathbf{x}_{\text{new}}$ from $\mathbf{X}$ and taking the label $\hat{y}_{\text{new}}$ which occurs most often among those K samples. This label $\hat{y}_{\text{new}}$ is then assigned to the unseen sample $\mathbf{x}_{\text{new}}$.

Because distance based classifiers are in general ineffective in higher dimension, it is useful to perform dimension-reduction techniques on $\mathbf{X}$ to get the low dimensional embeddings $\mathbf{Z}$ and then perform K-NN on the embeddings instead of the original samples so as to mitigate the curse of dimensionality. In this project, we use PCA for dimension reduction and use K-NN on the score-vectors for classificaiton.

6 Experiments

Please see the experiments notebook.

7 Conclusion

In this project we discussed the concept of discriminative classification and why it is prone to the curse of dimensionality in higher dimension. We then discuss possible mitigation strategies by using dimension reduction and introduce Principal Component Analysis and K-Nearest Neighbours as canonical dimension reduction and classification techniques respectively. Lastly, we perform several experiments on a face classification dataset to explore the effectiveness of PCA and K-NN being used in tandem.