CS 189/289A Introduction to Machine Learning
Fall 2024 Jennifer Listgarten, Saeed Saremi **DIS6**

1 Motivation: Dimensionality reduction

In this problem sheet we explore the motivation for general dimensionality reduction in machine learning and derive from first principles why projection on the first eigenvectors of the covariance matrix of the data has some favorable properties. A deeper understanding on the advantages of PCA and other dimensionality reduction methods is conveyed in the homework.

In general, we assume the following scenario: Suppose we are given n points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in $\mathbb{R}^d$ and the dimension of the feature vectors is d (very big, like 10^3). By dimensionality reduction, we refer to a mapping $\psi : \mathbb{R}^d \mapsto \mathbb{R}^k$ that maps vectors from $\mathbb{R}^d$ to $\mathbb{R}^k$ with $k \ll d$.

(a) (Motivation) Given n feature vectors of d dimensions, in which regimes of n, d and why would you want to reduce the dimensionality in practical machine learning applications? Think about the concept of regularization studied extensively in the past few weeks.

Solution:

In general:

There are two big motivations for dimensionality reduction. First, there is the simple one coming from the bias/variance tradeoff. You have seen that every feature essentially brings its own variance to the problem that scales like $\frac{1}{n}$. So, when there are a lot of potential features that we could use, the optimal number of features that we do use might be fewer. Hence, dimensionality reduction. The second is computational. Processing the data is costly. If a system of linear equations has to be solved, the complexity is super-linear in the number of variables. Cutting down the number of features cuts down the number of variables that we need to solve for.

Reducing dimension here seems to be win-win. Though for computing the reduction, we need to be smart, else there are no computational savings.

In slightly more detail: There are three basic regimes $n \ll d$ (high dimensional data regime = underdetermined), $n >\sim d$ and $n \gg d$ (overdetermined):

- When $n \ll d$ (underdetermined system), we use regularization (or model selection) assuming low rank structure to avoid the overfitting that would naturally occur.
- When $n \gg d$ there might appear to be enough data to get good estimates of the variables. However, even here, we can potentially get some advantages from dimensionality reduction (and other forms or regularization) if there is lower-dimensional models have good approximation error. (You saw this with polynomials fitting the exponential function.)

Another idea here (not explored that much in 189) is to do "sketching" to reduce the n samples to something that is fewer since there is plenty of data.

- Dimensionality reduction is in general most helpful in cases when $n \ll d$ or $n >\sim d$ and the problem has lower effective dimension (otherwise $n \ll d$ obviously wouldn't give you a reasonable estimator in the first place).

(b) (Computational aspect) Revisit this in the context of linear regression. What is the computational complexity of performing a linear regression of n data points in d dimensions with $n > d$ (say by solving the normal equations when $\mathbf{X}^\top\mathbf{X}$ is invertible)? If the projection was given to you for free, approximately how many operations would you save if you reduced the dimension from $d = 10^3$ to $d = 10$?

Solution:

Solving linear regression requires $O(nd^2)$ for $n > d$ (and $O(n^2d)$ for $n < d$). We discuss the case $n > d$ for which the computational complexity can be seen by considering the solution via normal equations, which is

$$\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}. \tag{1}$$

Forming the matrix $\mathbf{X}^\top\mathbf{X}$ costs $O(nd^2)$ and inverting it costs $O(d^3)$. Forming $\mathbf{X}^\top\mathbf{y}$ costs $O(nd)$ and the final matrix multiplication of the two $d \times d$ matrices costs $O(d^3)$. The total cost is therefore $O(d^2(n + d))$ which in the case $d < n$ is equal to $O(nd^2)$. An alternative approach to compute the computational complexity of L.R. is via SVD computation.

Reducing the dimension by a factor of 100 therefore gives a 10,000 fold speedup for solving the linear regression, which is quite substantial.

2 The Minimizing Reconstruction Error Perspective

One perspective on PCA is minimizing the perpendicular distance between the principle component subspace and the data points. Let's say we want to find the best 1D space that minimizes the reconstruction error. This is very closely linked to the interpretation of maximizing variance along a vector, which is covered in your homework 4.

(a) Show the (vector) projection of the feature vector x onto the subspace spanned by a unit vector w is

$$P_w(x) = w\left(x^\top w\right). \tag{2}$$

Solution: The general vector projection formula $P_w(x) = \frac{x^T w}{w^T w} w$. Because w is a unit vector $w^T w = 1$.

(b) Now, we want to choose w to minimize the reconstruction error. Show that taking w as the minimizer for the corresponding problem below gives us the same result as before.

$$\min_{w:|w|=1} \sum_{i=1}^{n} \|x_i - P_w(x_i)\|_2^2 \tag{3}$$

150

Solution: We have

$$\min_{w:|w|=1} \sum_{i=1}^{n} \|x_i - P_w(x_i)\|_2^2 \tag{4}$$

$$= \min_{w:|w|=1} \sum_{i=1}^{n} \left(\|x_i\|^2 - 2x_i^\top P_w(x_i) + \|P_w(x_i)\|^2\right) \tag{5}$$

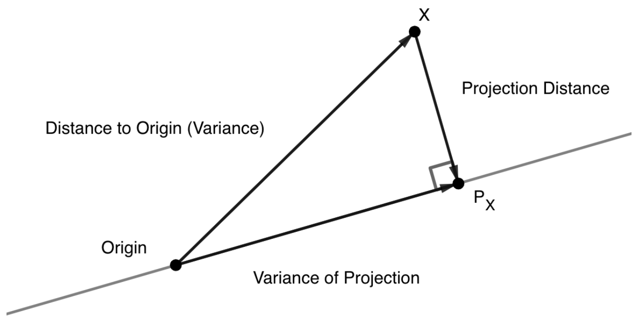
$$= \min_{w:|w|=1} \sum_{i=1}^{n} \left(\|x_i\|^2 - 2(x_i - P_w(x_i))^\top P_w(x_i) - 2P_w(x_i)^\top P_w(x_i) + \|P_w(x_i)\|^2\right) \tag{6}$$

$$= \min_{w:|w|=1} \sum_{i=1}^{n} \left(\|x_i\|^2 - \|P_w(x_i)\|^2\right) \tag{7}$$

$$= \min_{w:|w|=1} \sum_{i=1}^{n} \|x_i\|^2 - \sum_{i=1}^{n} \|(x_i^\top w)w\|^2 \tag{8}$$

$$= \min_{w:|w|=1} \sum_{i=1}^{n} \|x_i\|^2 - \underbrace{\sum_{i=1}^{n} (x_i^\top w)^2}_{\text{Variance Term}}. \tag{9}$$

where the third equality follows from the fact that $P_w(x_i)$ is an orthogonal projection onto the subspace spanned by w (and thus the error is orthogonal to any vector in the subspace including $P_w(x_i)$. Thus we see that minimizing reconstruction error is as same as maximizing variance as what we do in the previous question. Note that this problem can also be shown in alternative ways which you can find in the notes.



The above image serves as a useful visualization. Consider mean centered data. A data point has some fixed distance from the origin. We may consider finding a lower dimensional representation as either maximizing the variance of the projecting or minimizing the projection distance. The squared quantities must sum to a constant (the distance to the origin or original variance) thus minimizing one is equivalent to maximizing the other.

3 t-sne? Never heard of her

In this question we'll explore t-sne, which stands for t-distributed stochastic neighborhood embeddings. This is a **nonlinear** dimensionality reduction technique (as opposed to PCA), and is great when dealing with high dimensional data that isn't linear in fashion.

For the purposes of this problem, assume that we have a dataset $X = \{x_1, x_2, \dots, x_n\}$ where each x_i is d-dimensional. We'll project this down into $Y = \{y_1, y_2, \dots, y_n\}$, where each y is 2 or 3 dimensions. Y is initially generated randomly (either through a gaussian, or another process, and then iteratively modified through gradient descent).

We'll walk through the process outlined in the original t-sne paper, linked here.

(a) The classical stochastic neighborhood embedding algorithm generates probabilities

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|_2^2/(2\sigma_i^2)\right)}{\sum_{k\neq i}\exp\left(-\|x_i - x_k\|_2^2/(2\sigma_i^2)\right)}.$$

Why did we choose to model probabilities this way? What does the σ_i term represent?

Solution: This is equivalent to viewing under the following formulation: Suppose we had a gaussian centered on x_i with variance σ_i. Given that we choose a point proportional to its pdf, the probability we choose point j is given by $p_{i|j}$.

(b) p is not symmetric i.e. $p_{i|j} \neq p_{j|i}$ generally. When does this occur, and what is one way to fix the p matrix so that it is symmetric (assuming we don't change σ_i, σ_j?)

Solution: If x_j is an outlier then $p_{j|i}$ will be approximately $\frac{1}{n}$, but $p_{i|j}$ will be extremely close to 0. One way to solve this, which is what the paper does, is set $p_{ij} = \frac{p_{i|j}+p_{j|i}}{2}$. This ensures that $p_{ij} \geq \frac{1}{2n}$ which helps resolve numerical instability issues.

(c) Given lower dimensional projections y_i, t-SNE defines the following:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|_2^2)^{-1}}{\sum_{k\neq i}(1 + \|y_i - y_k\|_2^2)^{-1}}.$$

Contrast this to above where we modeled similarities using gaussians. Why might using a t-distribution be better and what problems can it potentially solve?

Solution: Gaussian distributions have very light tails at their ends, which are appealing for higher dimensional distributions. However, t-distributions have fatter tails which are appealing when looking at similarity metrics for lower dimensional distributions. This allows a moderate distance in the high-dimensional space to be faithfully modeled by a much larger distance in the map and, as a result, it eliminates the unwanted attractive forces between map points that represent moderately dissimilar datapoints.

The t-distribution with one degree of freedom is nice because it has the property that $(1 + \|y_i - y_j\|_2^2)^{-1}$ approaches an inverse square law for very large distances $\|y_i - y_j\|_2$. This makes the map's representation of joint probabilities (almost) invariant to changes in the scale of the map for map points that are far apart. It also means that large clusters of points that are far apart interact in just the same way as individual points, so the optimization operates in the same way at all but the finest scale

(d) y_i is then optimized through gradient descent. In order to do this, we need to define a cost function to optimize. t-SNE uses the cost function

$$C = KL(P\|Q) = \sum_i \sum_j p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right).$$

Why is the KL-divergence used here? And what is $\frac{\partial C}{\partial y_i}$?

Solution: The KL-divergence is a natural choice when trying to see the difference between two distributions. In this case we're projecting X into a smaller subspace Y so we want to make sure pairwise distance calculations are approximately maintained (which are p and q respectively). Optimizing the KL-divergence is one way to help do this.

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|_2^2)^{-1}$$

is the gradient with respect to y_i.