10-315 Notes
Principal Component Analysis

Carnegie Mellon University
Machine Learning Department

Contents

1 Motivation: Dimensionality Reduction

Dimensionality reduction is a key technique in machine learning and data analysis. It aims to map the original dataset to a smaller dimension, where each sample has fewer features. Dimensionality reduction can be thought of as reducing the dimensions of the data matrix $\mathbf{x}$ from $N \times M$ to $N \times K$ where $K < M$, hence the name dimensionality reduction. By reducing the number of features, we can

- Reduce computational cost and therefore increase the training speed of our models because there are fewer features to train on
- Improve model performance by excluding irrelevant or noisy features
- Reconstruct high-dimensional data in two or three dimensions for better visualization

Principal component analysis is just one dimensionality reduction strategy, which we will cover in detail below.

2 PCA Math Background

2.1 Projections

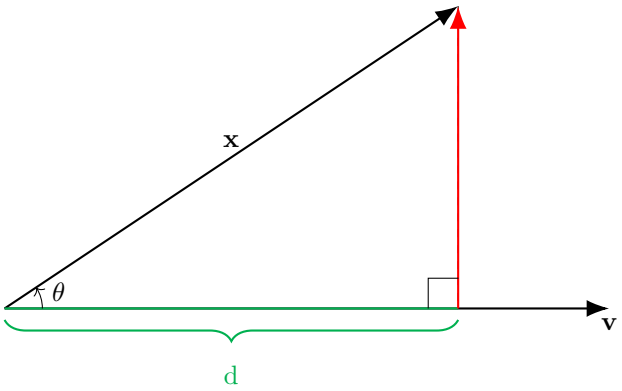
2.1.1 Scalar Projection

Given two vectors in $\mathbb{R}^N$ denoted $\mathbf{x}$ and $\mathbf{v}$, the scalar projection of $\mathbf{x}$ onto $\mathbf{v}$ is defined as:

$$d = \frac{\mathbf{v}^\top \mathbf{x}}{\|\mathbf{v}\|_2}$$

Note that if we assume that $\mathbf{v}$ is a unit vector, i.e., $\|\mathbf{v}\|_2 = 1$, the projection formula is *much* simpler:

$$d = \mathbf{v}^\top \mathbf{x}$$



The scalar projection, d, is the length of the vector $\mathbf{x}$ projected onto $\mathbf{v}$. We can prove this geometrically (recall that $\mathbf{v}^\top \mathbf{x} = \mathbf{v} \cdot \mathbf{x} = \|\mathbf{v}\|_2 \|\mathbf{x}\|_2 \cos(\theta)$):

$$\begin{aligned} \cos(\theta) &\triangleq \frac{d}{\|\mathbf{x}\|_2} \\ d &= \|\mathbf{x}\|_2 \cos(\theta) \\ d &= \frac{\mathbf{v}^\top \mathbf{x}}{\|\mathbf{v}\|_2} \end{aligned}$$

2.1.2 Vector Projection

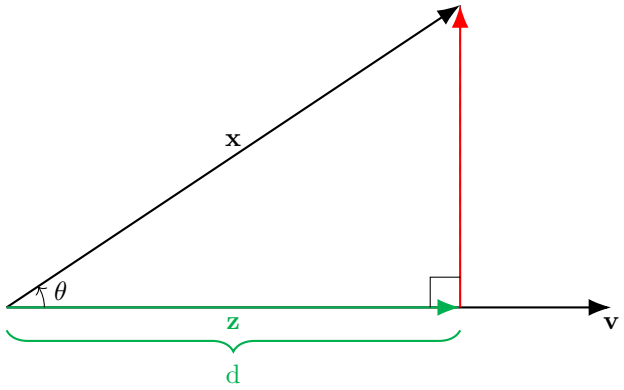
Recall from linear algebra, the definition of vector projection. Given two vectors in $\mathbb{R}^N$ denoted $\mathbf{x}$ and $\mathbf{v}$, the vector projection of $\mathbf{x}$ onto $\mathbf{v}$, or $\text{proj}_{\mathbf{v}}\mathbf{x}$, is defined as:

$$\mathbf{z} = \frac{\mathbf{v}^\top \mathbf{x}}{\|\mathbf{v}\|_2} \frac{\mathbf{v}}{\|\mathbf{v}\|_2}$$

Note that if we assume that $\mathbf{v}$ is a unit vector, i.e., $\|\mathbf{v}\|_2 = 1$, the projection formula is *much* simpler:

$$\mathbf{z} = \left(\mathbf{v}^\top \mathbf{x}\right) \mathbf{v}$$

The projected vector $\mathbf{z}$ lies in the direction of $\mathbf{v}$ and represents the component of $\mathbf{x}$ in the direction of $\mathbf{v}$. We can think of the projection $\mathbf{z}$ as a restriction of $\mathbf{x}$ to the $\mathbf{v}$-axis. The vector projection is visualized below:



Geometrically, the vector projection can be thought of as the vector **z** above, when both vectors are anchored at the origin.

To develop a better understanding of how projections work and what they mean, feel free to use this Desmos link. Moving around the u and v points in the Desmos link will change the projected vector.

2.2 Rotating Data

Now that we have established an intuition for projections, we can use this concept for rotating data. Recall that a basis for $\mathbb{R}^N$ is a set of unit vectors $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_N\}$ such that any vector $\mathbf{x} \in \mathbb{R}^N$ can be written as a linear combination of $\mathbf{v}_1, \ldots, \mathbf{v}_N$. When the basis vectors are perpendicular to each other, they can also be thought of as the axes of our data. For example, the basis vectors $\mathbf{v}_1 = [1, 0]^\top$ and $\mathbf{v}_2 = [0, 1]^\top$ represent the x-axis and y-axis in the standard Cartesian coordinate system. You should observe that vector $\mathbf{x} \in \mathbb{R}^2$ can be written as $z_1\mathbf{v}_1 + z_2\mathbf{v}_2$ where z_1 and z_2 are scalars.

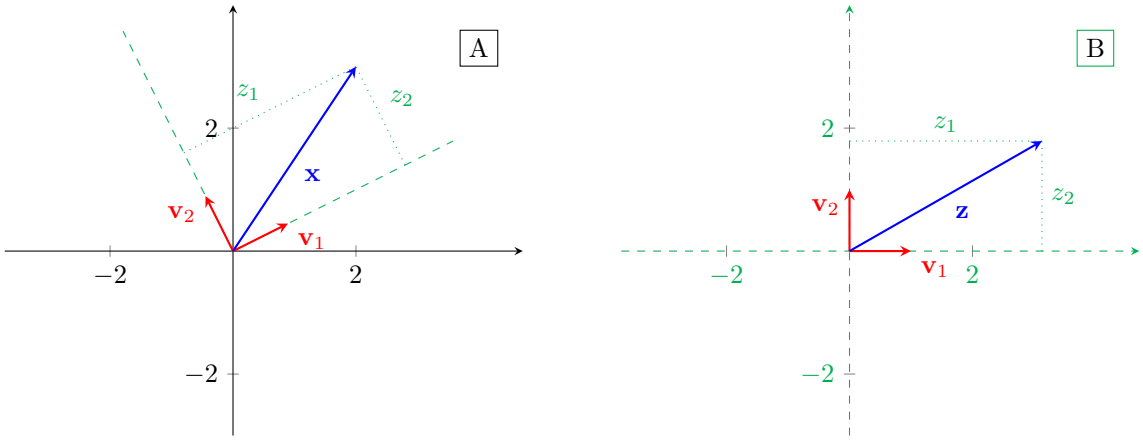
2.2.1 Rotation

We can transform the data to be aligned to any set of axes by projecting onto the corresponding set of basis vectors. Consider the $\mathbf{x} = [2, 3]^\top$. Now we will project $\mathbf{x}$ onto the basis vectors

$$\mathbf{v}_1 = \left[\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}}\right]^\top \text{ and } \mathbf{v}_2 = \left[\frac{-1}{\sqrt{5}}, \frac{2}{\sqrt{5}}\right]^\top$$

Note that $\mathbf{v}_1$ and $\mathbf{v}_2$ are once again unit vectors. Performing the projection calculation, we have that

$$z_1 = \mathbf{v}_1^\top \mathbf{x} = \frac{7}{\sqrt{5}} \approx 3.13$$
$$z_2 = \mathbf{v}_2^\top \mathbf{x} = \frac{4}{\sqrt{5}} \approx 1.79$$



2.2.2 Projection Matrix

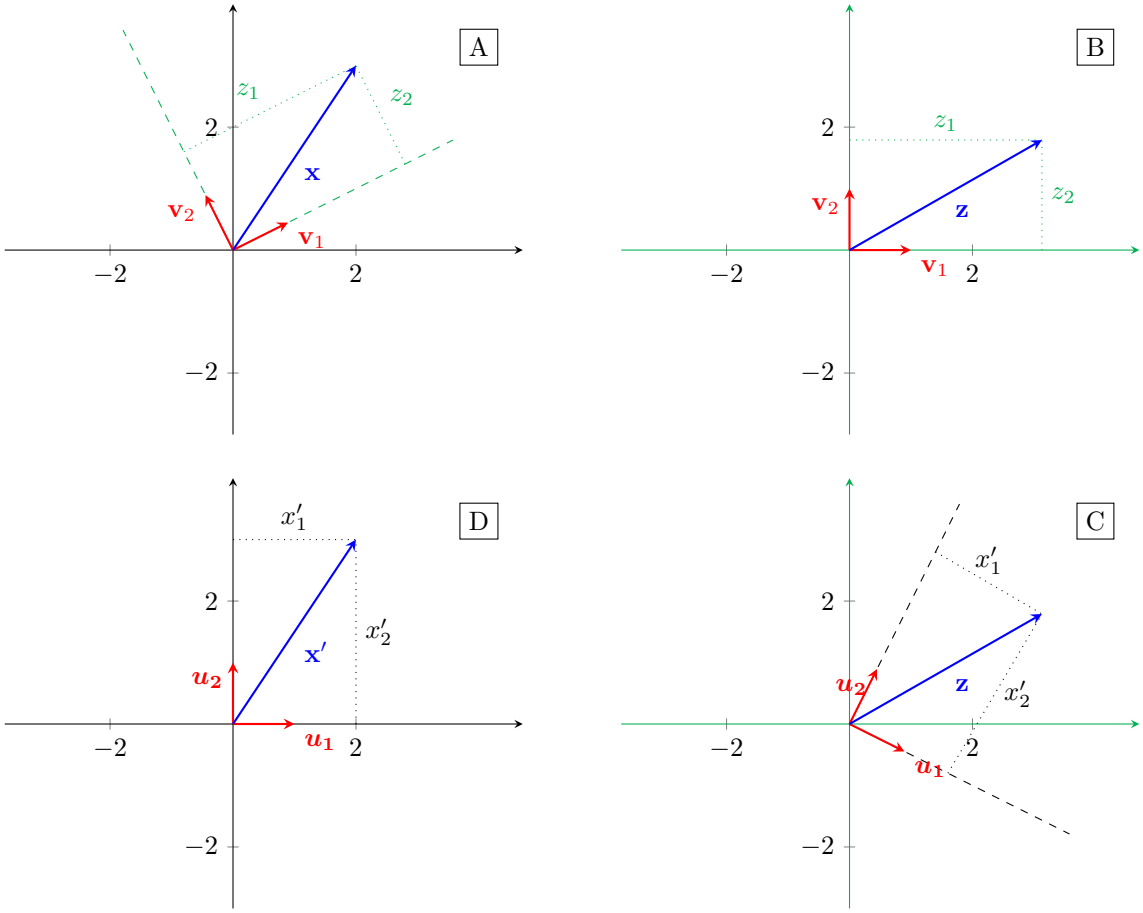
You may have noticed that when the $\mathbf{v}_i$ are unit vectors, z_i is just the dot product of $\mathbf{v}_1$ and $\mathbf{x}$. Let us now define V as:

$$V = \begin{bmatrix} - & \mathbf{v}_1^\top & - \\ & \vdots & \\ - & \mathbf{v}_N^\top & - \end{bmatrix}$$

Then it follows that $\mathbf{z} = V\mathbf{x}$ where the matrix V is called the projection matrix. Sometimes we want to reconstruct the original $\mathbf{x}$ from the projection $\mathbf{z}$. Let $\mathbf{x}' = V^\top \mathbf{z} = V^\top V\mathbf{x}$. When the projection matrix V is a square matrix, we are projecting $\mathbf{x}$ into a space with the same number of dimensions. Then $V^\top V = I \implies \mathbf{x}' = \mathbf{x}$, so we can reconstruct $\mathbf{x}$ perfectly. When we project into a lower-dimensional space, V is not a square matrix and thus we cannot reconstruct $\mathbf{x}$ perfectly.

Now, we can consider reversing that rotation through a different rotation matrix. Define $U = V^\top$. We can see that by applying the rotation U onto $\mathbf{z}$, we get $\mathbf{x}' = U\mathbf{z}$. Note that this is simply another rotation; however, because we have specifically chosen $U = V^{-1} = V^\top$, we get that $\mathbf{x}' = \mathbf{x}$.

Intuitively, if $\mathbf{x}$ represents the data in terms of the standard basis $[1, 0]^\top$ and $[0, 1]^\top$, then $\mathbf{z}$ represents the data in terms of $\mathbf{v}_1 = \left[\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}}\right]$ and $\mathbf{v}_2 = \left[\frac{-1}{\sqrt{5}}, \frac{2}{\sqrt{5}}\right]$. We can visualize what is happening below. Moving from plot A to plot B represents applying the rotation matrix V onto $\mathbf{x}$ to get $\mathbf{z}$. Then, by moving from plot C to D, we can see the reverse step by applying the rotation matrix $\boldsymbol{U}$ onto $\mathbf{z}$ to get $\mathbf{x}'$



2.3 Covariance Matrix

You may be wondering what we mean by the variance of our dataset. In machine learning, when our dataset $\mathcal{D}$ has multiple features, each sample is represented as a column vector $\mathbf{x}^{(i)} \in \mathbb{R}^M$. Then we can represent the entire dataset as a matrix $\mathbf{x} = [\mathbf{x}^{(i)} \dots \mathbf{x}^{(N)}]^\top$.

In this context, variance measures the variability of each feature in the dataset, as well as the relationships between different features. We will now introduce the covariance matrix.

The covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{M \times M}$ is a square matrix that summarizes the covariance between each pair of features in the dataset. The size of the matrix is determined by the number of features.

- **Diagonal Elements**: The diagonal elements of the covariance matrix represent the variances of each feature. In other words, they represent how spread out the values are for a specific feature. Mathematically, the variance of feature m is $\frac{1}{N}\sum_{i=1}^{N}(x_m^{(i)} - \mu_m)^2$ where $x_m^{(i)}$ is the value of feature m for the i-th data point and μ_m is the mean of feature m.
- **Off-Diagonal Elements**: The off-diagonal elements of the covariance matrix represent the covariance between different features. Covariance is a measure of how two features vary together. If the covariance is positive, it means that when one feature increases, the other tends to increase as well. If it's negative, it means that when one feature increases, the other tends to decrease. Mathematically, the covariance between feature j and feature j is $\frac{1}{N}\sum_{i=1}^{N}(x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$.

Generally, we assume that our data is centered and scaled for each feature, in other words $\mu_m = \sum_{i=1}^{N} x_m^{(i)} = 0$.

Because each feature of our dataset has a mean of zero, and $\mathbf{\Sigma}_{jk} = \frac{1}{N}\sum_{i=1}^{N}(x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$, the covariance matrix simplifies to

$$\mathbf{\Sigma} = \frac{1}{N}\mathbf{x}^\top\mathbf{x}$$