IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

Multivariate Calculus: A Deep Dive into Machine Learning

Payel Mondal[1], Suvankar Laga[2], Liza Roy[3] and Lisa Roy[4]

Assistant Professor, Basic Science and Humanities, Narula Institute of Technology, Kolkata, India[1,2]

Student, Computer Science and Engineering, Narula Institute of Technology, Kolkata, India[3,4]

Abstract: Multivariate calculus is a fancy method of briefing the topic in calculus that requires a bit more thought and work. It is just the application of some basic multivariable principles like differentiation, integration, rate of change, etc. Vector space, linear transformation, and matrices are some important areas of multivariable calculus. It majorly deals with three-dimensional objects or higher dimensions. The typical operations involved in the multivariable calculus are: Limits and Continuity, Partial Differentiation, Multiple Integration etc. It provides us with the tools to build an accurate predictive model [1]. Moreover, multivariate calculus can explain the change in our target variable in relation to the rate of change in the input variables.

Machine learning is the latest in a long line of attempts to distil human knowledge and reasoning into a form that is suitable for constructing machines and engineering automated systems. As machine learning becomes more ubiquitous and its software packages become easier to use, it is natural and desirable that the low-level technical details are abstracted away and hidden from the practitioner [2]. However, this brings with it the danger that a practitioner becomes unaware of the design decisions and, hence, the limits of machine learning algorithms.

For historical reasons, courses in machine learning tend to be taught in the computer science department, where students are often trained in the first two areas of knowledge, but not so much in mathematics and statistics.

Machine learning builds upon the language of mathematics to express concepts that seem intuitively obvious but that are surprisingly difficult to formalize. Once formalized properly, we can gain insights into the task we want to solve. One common complaint of students of mathematics around the globe is that the topics covered seem to have little relevance to practical problems. We believe that machine learning is an obvious and direct motivation for people to learn mathematics.

Keywords: Dimensionality Reduction, Difference Quotient, Principal Component Analysis, Principal Subspace.

I. INTRODUCTION

Machine learning is the latest in a long line of attempts to distil human knowledge and reasoning into a form that is suitable for constructing machines and engineering automated systems. As machine learning becomes more ubiquitous and its software packages become easier to use, it is natural and desirable that the low-level technical details are abstracted away and hidden from the practitioner. However, this brings with it the danger that a practitioner becomes unaware of the design decisions and, hence, the limits of machine learning algorithms.

A challenge we face regularly in machine learning is that concepts and words are slippery, and a particular component of the machine learning system can be abstracted to different mathematical concepts.

While not all data is numerical, it is often useful to consider data in a number format. Here, we assume that data has already been appropriately converted into a numerical representation suitable for reading into a computer program. Therefore, we think of data as vectors. As another illustration of how subtle words are, there are (at least) three different ways to think about vectors: a vector as an array of numbers (a computer science view), a vector as an arrow with a direction and magnitude (a physics view), and a vector as an object that obeys addition and scaling (a mathematical view).

A model is typically used to describe a process for generating data, similar to the dataset at hand. Therefore, good models can also be thought of as simplified versions of the real (unknown) data-generating process, capturing aspects that are relevant for modelling the data and extracting hidden patterns from it. A good model can then be used to predict what would happen in the real world without performing real-world experiments.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

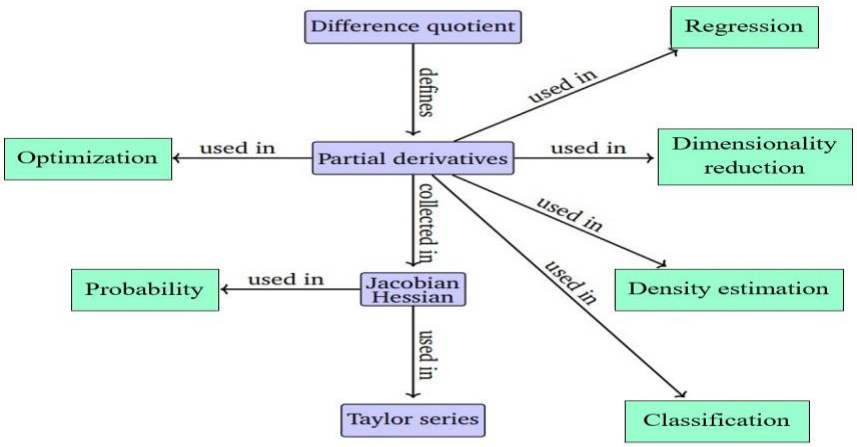
Vol. 9, Special Issue 2, September 2022

If we summarize the overall concepts of machine learning that we take under consideration:

- Represent data as vectors.
- Choose an appropriate model, either using the probabilistic or optimization view.
- Learn from available data by using numerical optimization methods with the aim that the model performs well on data not used for training.

II. A SOFT DISCUSSION TO MULTIVARIATE CALCULUS

Many algorithms in machine learning optimize an objective function with respect to a set of desired model parameters that control how well a model explains the data: Finding good parameters can be phrased as an optimization problem [3]. Examples include: (i) linear regression, where we look at curve-fitting problems and optimize linear weight parameters to maximize the likelihood; (ii) neural-network auto-encoders for dimensionality reduction and data compression, where the parameters are the weights and biases of each layer, and where we minimize a reconstruction error by repeated application of the chain rule; and (iii) Gaussian mixture models for modelling data distributions, where we optimize the location and shape parameters of each mixture component to maximize the likelihood of the model [4]. A function f is a quantity that relates two quantities to each other. In this paper, these quantities are typically inputting$x \in \mathbb{R}^D$ and targets (function values) $f(x)$, which we assume are real-valued if not stated otherwise. Here $\mathbb{R}^D$ is the domain of f, and the function values $f(x)$ are the image/codomain of f.



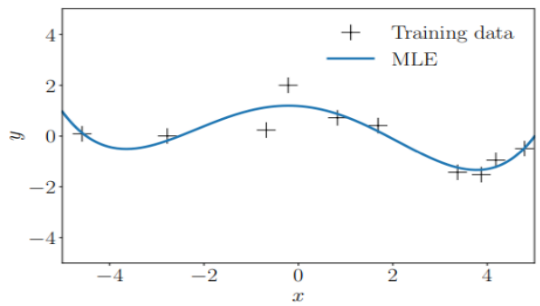


Fig. 2(a) Regression problem

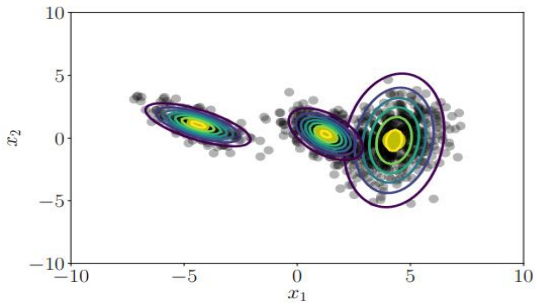


Fig. 2(b) Density estimation with a Gaussian mixture model

Let us take an example,
The dot product as a special case of an inner product. The function $f(x) = x^T x, x \in R^2$, would be specified as,
$f: \mathbb{R}^2 \rightarrow \mathbb{R}$ ..... (1)
$x \rightarrow x_1^2 + x_2^2$ ..... (2)
Here, equation (1) specifies that f is a mapping from $\mathbb{R}^2 \rightarrow$ $\mathbb{R}$and equation (2) specifies the explicit assignment of an input x to a function value $f(x)$. A function f assigns every input x exactly one function value $f(x)$.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

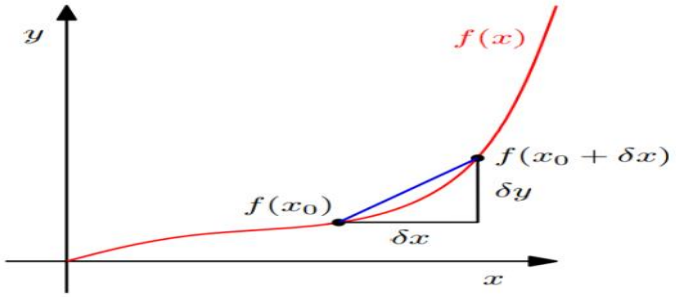


Fig. 3 Differentiation of Univariate Functions

The average incline of a function f between x_0 and $x_0 + \delta_x$ is the incline of the secant (blue) through $f(x_0)$ and $f(x_0 + \delta x)$ and given by $\delta y/\delta x$.

1. (Difference Quotient). The difference quotient

$$\delta y/\,\delta x := [f(x + \delta x) - f(x)]\,/\delta x$$

computes the slope of the secant line through two points on the graph of f. In Figure A, these are the points with x −co-ordinates x_0 and $x_0 + \delta x$.

2. (Derivative). More formally, for $h > 0$ the derivative of f at x is defined as the limit

$$df = \lim_{h \to 0} \frac{f(x + h) - f(x)}{dx},$$

and the secant in Figure 3 becomes a tangent. The derivative of f points in the direction of steepest ascent of f.

3. (Taylor Polynomial). The Taylor polynomial of degree n of $f : \mathbb{R} \to \mathbb{R}$ at x_0 is defined as

$$T_n(x) := \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k$$

where $f^k(x_0)$ is the kth derivati) $/k!$ are the coefficients of the polynomial.

4: (Taylor Series). For a smooth function $f \in C^\infty$, $f : \mathbb{R} \to \mathbb{R}$, the Taylor series of f at x_0 is defined as

$$T_\infty(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k$$

For $x_0 = 0$, we obtain the Maclaurin series as a special instance of the Taylor series. If $f(x) = T^\infty(x)$, then f is called analytic.

.

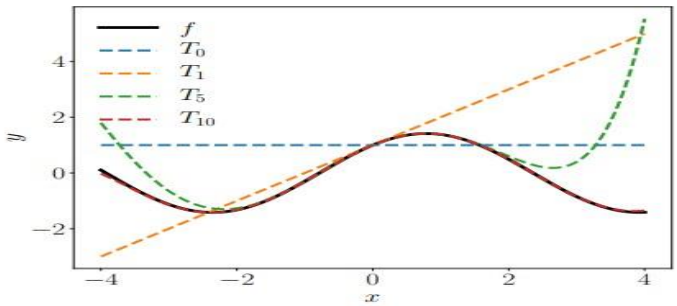


Fig. 4 Taylor polynomial

Example (Taylor Series) Consider the function in above Figure given by

$$f(x) = sin(x) + cos(x) \in C\infty.$$

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

We seek a Taylor series expansion of f at$x_0 = 0$, which is the Maclaurin series expansion of f. We obtain the following derivatives:

$$f(0) = sin(0) + cos(0) = 1$$
$$f'(0) = cos(0) - sin(0) = 1$$
$$f''(0) = -sin(0) - cos(0) = -1$$
$$f'''(0) = -cos(0) + sin(0) = -1$$
$$f^{iv}(0) = sin(0) + cos(0) = f(0) = 1 \ldots$$

We can see a pattern here: The coefficients in our Taylor series are only± 1 (since $sin(0) = 0$), each of which occurs twice before switching to the other one. Furthermore, $f^{k+4}(0) = f^k(0)$.
Therefore, the full Taylor series expansion of f at $x_0 = 0$ is given by

$$\begin{aligned} T_\infty(x) &= \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k \\ &= 1 + x - \frac{1}{2!}x^2 - \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 - \cdots \\ &= 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 \mp \cdots + x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 \mp \cdots \\ &= \sum_{k=0}^{\infty}(-1)^k \frac{1}{(2k)!}x^{2k} + \sum_{k=0}^{\infty}(-1)^k \frac{1}{(2k+1)!}x^{2k+1} \\ &= \cos(x) + \sin(x)\,, \end{aligned}$$

We often consider data to be noisy observations of some true underlying signal. We hope that by applying machine learning we can identify the signal from the noise. This requires us to have a language for quantifying what "noise" means. We often would also like to have predictors that allow us to express some sort of uncertainty, e.g., to quantify the confidence we have about the value of the prediction at a particular test data point. Quantification of uncertainty is the realm of probability theory. To train machine learning models, we typically find parameters that maximize some performance measure. Many optimization techniques require the concept of a gradient, which tells us the direction in which to search for a solution. The three components of machine learning are data, models, and parameter estimation in a mathematical fashion. In addition, we provide some guidelines for building experimental set-ups that guard against overly optimistic evaluations of machine learning systems. Recall that the goal is to build a predictor that performs well on unseen data [5-8].

we will have a close look at linear regression, where our linear regression objective is to find functions that map inputs $x \in \mathbb{R}^D$ to corresponding observed function values $y \in \mathbb{R}$, which we can interpret as the labels of their respective inputs. We will discuss classical model fitting (parameter estimation) via maximum likelihood and maximum estimation, as well as Bayesian linear regression, where we integrate the parameters out instead of optimizing them. The key objective of dimensionality reduction is to find a compact, lower-dimensional representation of high-dimensional data $x \in \mathbb{R}^D$, which is often easier to analyse than the original data. Unlike regression, dimensionality reduction is only concerned about modelling the data – there are no labels associated with a data point x. The density estimation objective of density estimation is to find a probability distribution that describes a given dataset. We will focus on Gaussian mixture models for this purpose, and we will discuss an iterative scheme to find the parameters of this model. As in dimensionality reduction, there are no labels associated with the data points$x \in \mathbb{R}^D$. However, we do not seek a low-dimensional representation of the data. Instead, we are interested in a density model that describes the data. At last, we conclude with an in-depth discussion of the fourth pillar: classification. We will discuss classification in the context of support vector machines. Similar to regression, we have inputs x and corresponding labels y. However, unlike regression, where the labels were real-valued, the labels in classification are integers, which requires special care.

III. DIMENSIONALITY REDUCTION WITH PRINCIPAL COMPONENT ANALYSIS

Working directly with high-dimensional data, such as images, comes with some difficulties: It is hard to analyse, interpretation is difficult, visualization is nearly impossible, and (from a practical point of view) storage of the data vectors can be expensive. However, high-dimensional data often has properties that we can exploit. For example, high-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions. Furthermore, dimensions in high-dimensional data are often correlated so that the data possesses an intrinsic lower-dimensional structure. Dimensionality reduction exploit structure and correlation and allows us to work with a more compact representation of the data, ideally without losing information [9]. We can think of dimensionality reduction as a compression technique, similar to jpeg or mp3, which are compression algorithms for images and music [10].

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

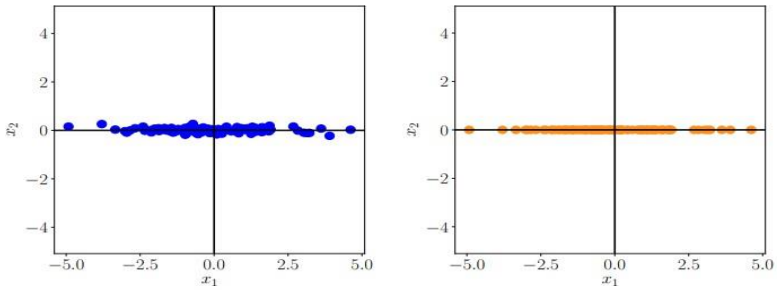
Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

Here we will discuss principal component analysis (PCA), an algorithm for linear dimensionality reduction. It is also used for the identification of simple patterns, latent factors, and structures of high-dimensional data.

- **Problem Setting**

In PCA, we are interested in finding projections $\tilde{x}_n$ of data points x_n that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality. Figure below gives an illustration of what this could look like.



More concretely, we consider adataset $X = \{x_1, x_2, \ldots, x_N\}, x_n \in \mathbb{R}^D$, with mean 0 that possesses the data covariance matrix

$$S = \frac{1}{N}\sum_{n=1}^{N} x_n x_n^T$$

Furthermore, we assume there exists a low-dimensional compressed representation (code)

$$z_n = B^T x_n \in \mathbb{R}^M$$

of x_n, where we define the projection matrix
$\boldsymbol{B} := [b_1, \ldots, b_M] \in R^{D \times M}$.
We will find low-dimensional representations that retain as much information as possible and minimize the compression loss. An alternative derivation of PCA is given where we will be looking at minimizing the squared reconstruction error $\|x_n - \tilde{x}_n\|^2$between the original data and its projection.

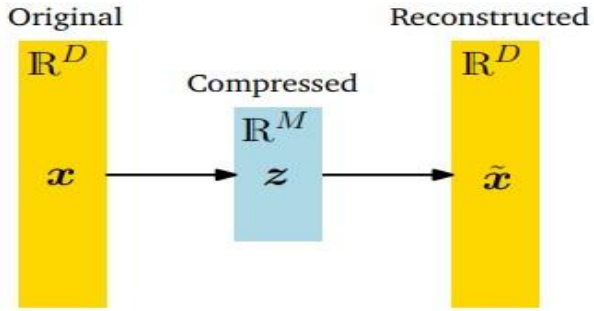


Fig. 6 Dimensionality reduction-2

Figure above illustrates the setting we consider in PCA, where z represents the lower-dimensional representation of the compressed data $\tilde{x}$ and plays the role of a bottleneck, which controls how much information can flow between $x$and $\tilde{x}$. In PCA, we consider a linear relationship between the original data x and its low-dimensional code $z \; so \; that \; z = B^T \; x \; and \; \tilde{x} = Bz$ for a suitable matrix B. Based on the motivation of thinking of PCA as a data compression technique, we can interpret the arrows in. As a pair of operations representing encoders and decoders. The linear mapping represented by B can be thought of as a decoder, which maps the low-dimensional code $z \in \mathbb{R}^M$ back into the original data space. Similarly, B^T can be thought of an encoder, which encodes the original data x as a low-dimensional (compressed) code z.

- **Projection Perspective**

In the following, we will derive PCA as an algorithm that directly minimizes the average reconstruction error. This perspective allows us to interpret PCA as implementing an optimal linear auto-encoder.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

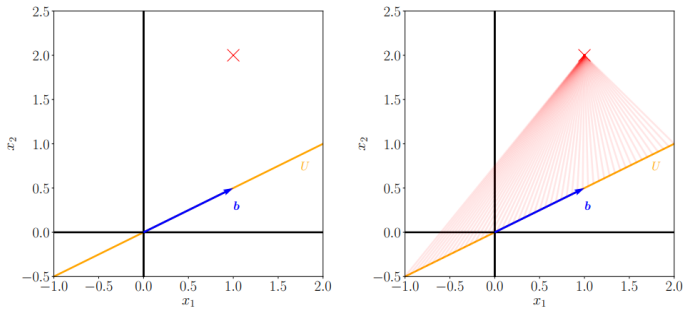


Fig. 7 PCA algorithm

We derived PCA by maximizing the variance in the projected space to retain as much information as possible. In the following, we will look at the difference vectors between the original data x_n and their reconstruction $\tilde{x}_n$ and minimize this distance so that x_n and$\tilde{x}_n$ are as close as possible.

- **Setting and Objective**

We are interested in finding vectors $\tilde{x} \in \mathbb{R}^D$, which live in lowerdimensional subspace$U \subseteq \mathbb{R}^D$, $dim(U) = M$, so that is as similar to x as possible. Note that at this point we need to assume that the coordinates $z_m$of$\tilde{x}$ and ζ_m of x are not identical.

$$\tilde{\boldsymbol{x}} = \sum_{m=1}^{M} z_m \boldsymbol{b}_m \in U \subseteq \mathbb{R}^D$$

In the following, we use exactly this kind of representation of $\tilde{x}$ to find optimal coordinates z and basis vectors $b_1, b_2, \ldots, b_M$such that $\tilde{x}$ is as similar to the original data point x as possible, i.e., we aim to minimize the (Euclidean) distance$\|x_n - \tilde{x}_n\|$. Figureillustrates this setting.

Without loss of generality, we assume that the dataset $X = \{x_1, x_2 \ldots, x_N\}, x_n \in \mathbb{R}^D$, is centered at 0, i.e., $E[X] = 0$. Without the zero-mean assumption, we would arrive at exactly the same solution, but the notation would be substantially more cluttered. We are interested in finding the best linear projection of X onto a lowerdimensional subspace U of $\mathbb{R}^D$ with $dim(U) = M$ and orthonormal basis vectors $b_1, b_2, \ldots, b_M$. We will call this subspace U the principal subspace. The projections of the data points are denoted by where $Z_n := [z_{1n}, z_{2n} \ldots, z_{Mn}] \in \mathbb{R}^M$ is the coordinate vector of $\tilde{x}_n$ with respect to the basis $(b_1, b_2, \ldots, b_M)$. More specifically, we are interested in having $\tilde{x}_n$the as similar to $x_n$as possible. The similarity measure we use in the following is the squared distance (Euclidean norm) $\|x - \tilde{x}\|^2$ between x and$\tilde{x}$. We therefore define our objective as minimizing the average squared Euclidean

$$\tilde{\boldsymbol{x}}_n := \sum_{m=1}^{M} z_{mn} \boldsymbol{b}_m = \boldsymbol{B}\boldsymbol{z}_n \in \mathbb{R}^D, \qquad J_M := \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2,$$

distance (reconstruction error) where we make it explicit that the dimension of the subspace onto which we project the data is M. In order to find this optimal linear projection, we need to find the orthonormal basis of the principal subspace and the coordinates $z_n \in \mathbb{R}^M$ of the projections with respect to this basis. To find the coordinates z_n and the ONB of the principal subspace, we follow a two-step approach. First, we optimize the coordinates z_n for a given ONB $(b_1, b_2, \ldots, b_M)$; second, we find the optimal ONB.

- **PCA Using Low-Rank Matrix Approximations**

To maximize the variance of the projected data (or minimize the average squared reconstruction error), PCA chooses the columns of U in to be the eigenvectors that are associated with the M largest eigenvalues of the data covariance matrix S so that we identify$U$ as the projection matrix B, which projects the original data onto subspace of dimension M. The Eckart-Young theorem offers a direct way to estimate the low-dimensional representation. Consider the best rank-M approximation

$$\tilde{\boldsymbol{X}}_M := \operatorname{argmin}_{\mathrm{rk}(\boldsymbol{A}) \leqslant M} \|\boldsymbol{X} - \boldsymbol{A}\|_2 \in \mathbb{R}^{D \times N}$$

of X, where $\|.\|_2$ is the spectral norm. The Eckart-Young theorem states that $\tilde{X}_M$ is given by truncating the SVD at the top-M singular value. In other words, we obtain

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

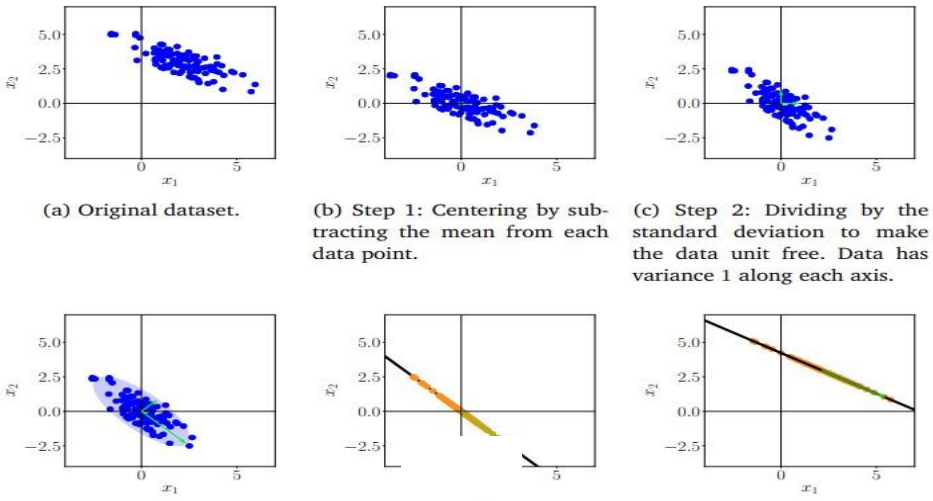
$$\tilde{\boldsymbol{X}}_M = \underbrace{\boldsymbol{U}_M}_{D\times M}\ \underbrace{\boldsymbol{\Sigma}_M}_{M\times M}\ \underbrace{\boldsymbol{V}_M^\top}_{M\times N} \in \mathbb{R}^{D\times N}$$

with orthogonal matrices $U_M := [u_1, u_2 \ldots, u_M] \in \mathbb{R}^{D\times M}$ and $V_M := [v_1, v_2 \ldots, v_M] \in \mathbb{R}^{N\times M}$and a diagonal matrix $\sum_M \in \mathbb{R}^{M\times M}$whose diagonal entries are the M largest singular values of X.

- **Key Steps of PCA in Practice**

In the following, we will go through the individual steps of PCA using a running example, which is summarized in Figure. We are given a two-dimensional dataset, and we want to use PCA to project it onto a one-dimensional subspace.

1. **Mean subtraction**: We start by centring the data by computing the mean μ of the dataset and subtracting it from every single data point. This ensures that the dataset has mean 0. Mean subtraction is not strictly necessary but reduces the risk of numerical problems.



(a) Original dataset.

(b) Step 1: Centering by subtracting the mean from each data point.

(c) Step 2: Dividing by the standard deviation to make the data unit free. Data has variance 1 along each axis.

(d) Step 3: Compute eigenvalues and eigenvectors (arrows) of the data covariance matrix (ellipse).

(e) Step 4: Project data onto the principal subspace.

(f) Undo the standardization and move projected data back into the original data space from (a).

2. **Standardization:** Divide the data points by the standard deviation σ_d of the dataset for every dimension $d = 1, \ldots, D$. Now the data is unit free, and it has variance 1 along each axis, which is indicated by the two arrows in the following figure. This step completes the standardization of the data.

3. **Eigen decompositionof the covariance matrix:** Compute the data covariance matrix and its eigenvalues and corresponding eigenvectors. Since the covariance matrix is symmetric, the spectral theorem states that we can find an ONB of eigenvectors. In the following figure, the eigenvectors are scaled by the magnitude of the corresponding eigenvalue. The longer vector spans the principal subspace, which we denote by U. The data covariance matrix is represented by the ellipse.

4. **Projection:** We can project any data point$x_* \in \mathbb{R}^D$ onto the principal subspace: To get this right, we need to standardize x_*using the mean μ_d and standard deviation σ_d of the training data in the dth dimension, respectively, so that where $x_*{}^d$ is the dth component of x_*. We obtain the projection as

with coordinates

$$\tilde{\boldsymbol{x}}_* = \boldsymbol{B}\boldsymbol{B}^\top \boldsymbol{x}_*$$

$$x_*^{(d)} \leftarrow \frac{x_*^{(d)} - \mu_d}{\sigma_d}, \quad d = 1, \ldots, D$$

with respect to the basis of the principal subspace. Here, **B** is the matrix that contains the eigenvectors that are associated with the largest eigenvalues of the data covariance matrix as columns. PCA returns the coordinates, not the projections x_*.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

5th National Conference on Science, Technology and Communication Skills – NCSTCS 2K22

Narula Institute of Technology, Agarpara, Kolkata, India

Vol. 9, Special Issue 2, September 2022

Having standardized our dataset, only yields the projections in the context of the standardized dataset. To obtain our projection in the original data space (i.e., before standardization), we need to undo the standardization and multiply by the standard deviation before adding the mean so that we obtain

$$\tilde{x}_*^{(d)} \leftarrow \tilde{x}_*^{(d)}\sigma_d + \mu_d\,, \quad d = 1, \ldots, D$$

Above figure above illustrates the projection in the original data space.

IV. CONCLUSION

Here we have discussed a close relation between an important branch in Mathematics (Multivariate Calculus) and machine learning and tried to give some guidelines for building experimental set-ups that guard against overly optimistic evaluations of machine learning systems. We have also focused on dimensionality reduction, an essential pillar of machine learning, using principal component analysis. The key objective of dimensionality reduction is to find a compact, lower-dimensional representation of high-dimensional data $x \in \mathbb{R}^D$, which is often easier to analyze than the original data. Unlike regression, dimensionality reduction is only concerned about modelling the data – there are no labels associated with a data point.

REFERENCES

[1] Bennett, Kristin P., and Bredensteiner, Erin J. 2000b. Geometry in Learning. Pages 132–145 of: Geometry at Work. Mathematical Association of America.
[2] Baydin, Atılım G., Pearlmutter, Barak A., Radul, Alexey A., and Siskind, Jeffrey M. 2018. Automatic Differentiation in Machine Learning: A Survey. Journal of Machine Learning Research, 18, 1–43
[3] Dostal, Zdenek. 2009. Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities. Springer.
[4] Brown, Lawrence D. 1986. Fundamentals of Statistical Exponential Families: With Applications in Statistical Decision Theory. Institute of Mathematics
[5] Eckart, Carl, and Young, Gale. 1936. The Approximation of One Matrix by Another of Lower Rank. Psychometrika, 1(3), 211–218.
[6] Grinstead, Charles M., and Snell, J. Laurie. 1997. Introduction to Probability. American Mathematical Society.
[7] Jimenez Rezende, Danilo, and Mohamed, Shakir. 2015. Variational Inference with Normalizing Flows. In: Proceedings of the International Conference on Machine Learning.
[8] Kalman, Dan. 1996. A Singularly Valuable Decomposition: The SVD of a Matrix. College Mathematics Journal, 27(1), 2–23.
[9] Leemis, Lawrence M., and McQueston, Jacquelyn T. 2008. Univariate Distribution Relationships. American Statistician, 62(1), 45–53.
[10] Moonen, Marc, and De Moor, Bart. 1995. SVD and Signal Processing, III: Algorithms, Architectures and Applications. Elsevier.