Mathemathics and Statistics for AI

elena.loli

July 2023

Outline

Bibliography

- M.Deisenroth, A. Faisal, C.S. Ong *Mathematics for Machine Learning*, Cambridge University Press.
- Thomas Garrett, Mathematics for Machine Learning, preprint, Department of Electrical Engineering and Compuer Science, Berkley University.
- R. Johanson, Numerical Python, Apress
- A. Quarteroni, R. Sacco, F Saleri, Numerical mathematics, Springer
- G. Thomas, Mathematics for Machine Learning `http://gwthomas.github.io/docs/math4ml.pdf`

Motivation

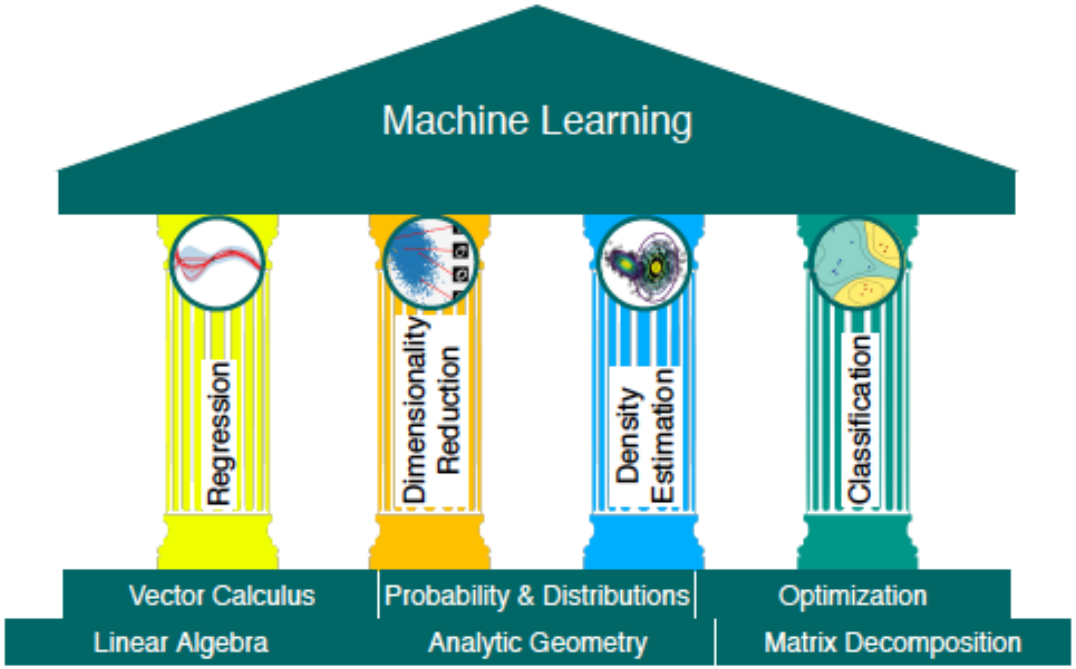
- Machine learning is about designing algorithms that automatically extract valuable information from data. The emphasis here is on "automatic", i.e., machine learning is concerned about general-purpose methodologies that can be applied to many datasets, while producing something that is meaningful.
- There are three concepts that are at the core of machine learning: data, a model, and learning.

Motivation

Why a course on mathematics and statistics in master degree on Artifical Intelligence?

- We introduce the main mathematical and statistical concepts to talk about the three main components of machine learning: data, models and learning.
- **Data** are represented as vectors: linear algebra is the setting for manipulating and using vectors.
- **Model**. A good model can be used to predict what happens in the real world without performing real-world experiments. Optimization and probability are two different perspective for building models.
- **Learning** from available data is the basis of Machine Learning. Multivariate analysis and optimization are necessary tools for learning.

Motivation


Machine Learning
Regression
Dimensionality Reduction
Density Estimation
Classification
Vector Calculus
Probability & Distributions
Optimization
Linear Algebra
Analytic Geometry
Matrix Decomposition

Data and linear algebra

- Since machine learning is inherently data driven, data is at the core data of machine learning.
- The goal of machine learning is to design general purpose methodologies to extract valuable patterns from data, ideally without much domain-specific expertise.
- We represent numerical data as vectors and represent a table of such data as a matrix. The study of vectors and matrices is called *linear algebra*

Models and probability

- 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 modeling 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. We often consider data to be noisy observations of some true underlying signal.
- 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*

Learning and optimization

- The crux of the matter is the learning component of machine learning.
- Assume we are given a dataset and a suitable model. Training the model means to use the data available to optimize some parameters of the model with respect to a utility function that evaluates how well the model predicts the training data.
- Most training methods can be thought of as an approach analogous to climbing a hill to reach its peak. In this analogy, the peak of the hill corresponds to a maximum of some performance measures.

By summarizing...

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

Outline

Prerquisites:vectors, matrices and vector spaces (ch.2 MML book)

- Vectors and Matrices operations
- Inverse and transpose matrix
- Solving linear systems of equations
- Vector spaces ans subspaces
- Linear independence of vectors: basis and rank
- Linear mappings between vector spaces and their matrix representation
- Affine spaces and affine mappings

Insight on linear mappings

Definition 2.19 (Transformation Matrix). Consider vector spaces V, W with corresponding (ordered) bases $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ and $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m)$. Moreover, we consider a linear mapping $\Phi : V \to W$. For $j \in \{1, \ldots, n\}$,

$$\Phi(\boldsymbol{b}_j) = \alpha_{1j}\boldsymbol{c}_1 + \cdots + \alpha_{mj}\boldsymbol{c}_m = \sum_{i=1}^{m} \alpha_{ij}\boldsymbol{c}_i \tag{2.92}$$

is the unique representation of $\Phi(\boldsymbol{b}_j)$ with respect to C. Then, we call the $m \times n$-matrix $\boldsymbol{A}_\Phi$, whose elements are given by

$$A_\Phi(i, j) = \alpha_{ij} \,, \tag{2.93}$$

the *transformation matrix* of Φ (with respect to the ordered bases B of V and C of W).

Geometric vectors: computing length and distances (ch3 MML book)

To compute the length of a vector (matrix) and the distance between vectors (matrices) we need a measure.

Definition 3.1 (Norm). A *norm* on a vector space V is a function

$$\|\cdot\| : V \to \mathbb{R}\,, \tag{3.1}$$

$$\boldsymbol{x} \mapsto \|\boldsymbol{x}\|\,, \tag{3.2}$$

which assigns each vector $\boldsymbol{x}$ its *length* $\|\boldsymbol{x}\| \in \mathbb{R}$, such that for all $\lambda \in \mathbb{R}$ and $\boldsymbol{x}, \boldsymbol{y} \in V$ the following hold:

- *Absolutely homogeneous:* $\|\lambda \boldsymbol{x}\| = |\lambda| \|\boldsymbol{x}\|$
- *Triangle inequality:* $\|\boldsymbol{x} + \boldsymbol{y}\| \leqslant \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$
- *Positive definite:* $\|\boldsymbol{x}\| \geqslant 0$ and $\|\boldsymbol{x}\| = 0 \iff \boldsymbol{x} = \boldsymbol{0}$

Example of norms

- 1-norm (or Manhattan):

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|, \quad x \in \mathcal{R}^n$$

- 2-norm (or Euclidean):

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}, \quad x \in \mathcal{R}^n$$

Example of norms

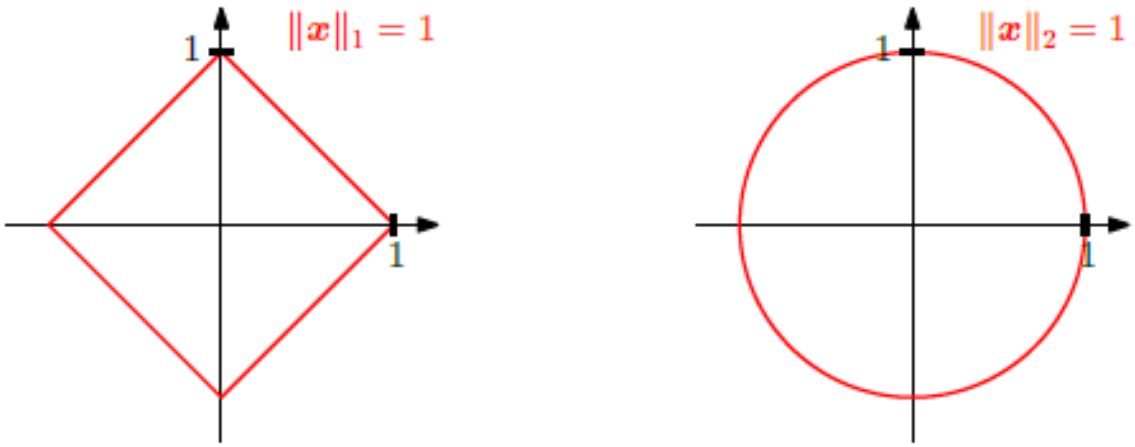


Figure 3.3 For different norms, the red lines indicate the set of vectors with norm 1. Left: Manhattan norm; Right: Euclidean distance.

Inner products

- Inner products allow for the introduction of intuitive geometrical concepts, such as the length of a vector and the angle or distance between two vectors.
- A major purpose of inner products is to determine whether vectors are orthogonal to each other.
- We may already be familiar with a particular type of inner product, the scalar product in $\mathcal{R}^n$, which is given by:

$$x^T y = \sum_{i=1}^{n} x_i y_i$$

- Inner products and norms are related:

$$\|x\|_2^2 = x^T x$$

Inner products

In addition to enabling the definition of lengths of vectors, as well as the distance between two vectors, inner products also capture the geometry of a vector space by defining the unique angle $\omega \in [0, \pi]$ between two vectors:

$$cos\omega = \frac{x^T y}{\|x\|_2 \|y\|_2}$$

Orthogonal vectors and matrices

- $x, y \in \mathcal{R}^n$ are orthogonal iff $< x, y >= x^T y = 0$
- $x, y \in \mathcal{R}^n$ are orthonormal iff x, y are orthogonal and $\|x\| = \|y\| = 1$
- A square matrix $A \in \mathcal{R}^{n \times n}$ is orthogonal iff its columns are orthonormal vectors. In this case, $A^T = A^{-1}$

Orthonormal basis (ONB)

Definition 3.9 (Orthonormal Basis)**.** Consider an n-dimensional vector space V and a basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ of V. If

$$\langle \boldsymbol{b}_i, \boldsymbol{b}_j \rangle = 0 \quad \text{for } i \neq j \tag{3.33}$$

$$\langle \boldsymbol{b}_i, \boldsymbol{b}_i \rangle = 1 \tag{3.34}$$

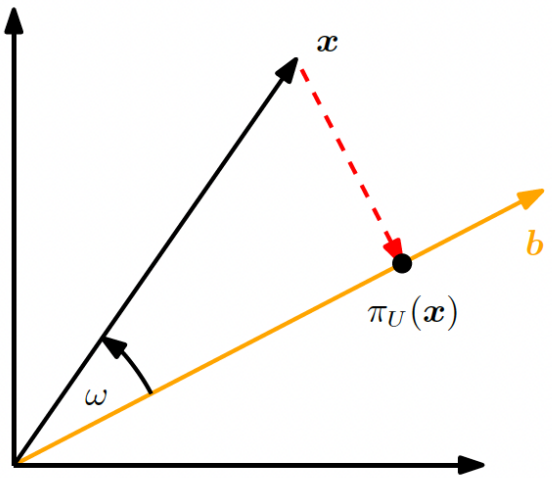
for all $i, j = 1, \ldots, n$ then the basis is called an *orthonormal basis* (*ONB*). If only (3.33) is satisfied, then the basis is called an *orthogonal basis*. Note that (3.34) implies that every basis vector has length/norm 1.

Orthogonal projections

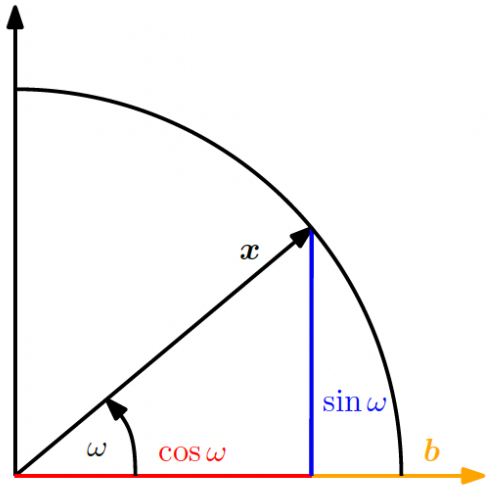
- Projections are an important class of linear transformations (besides rotations and reflections) and play an important role in graphics, coding theory, statistics and machine learning.
- In machine learning, we often deal with data that is high-dimensional. High-dimensional data is often hard to analyze or visualize.
- However, high-dimensional data quite often possesses the property that only a few dimensions contain most information, and most other dimensions are not essential to describe key properties of the data.
- When we compress or visualize high-dimensional data, we will lose information. To minimize this compression loss, we ideally find the most informative dimensions in the data.
- More specifically, we can project the original high-dimensional data onto a lower-dimensional feature space and work in this lower-dimensional space to learn more about the dataset and extract relevant patterns
- For example, machine learning algorithms, such as principal component analysis (PCA) by Pearson (1901) and Hotelling (1933) and deep neural networks (e.g., deep auto-encoders (Deng et al., 2010)), heavily exploit the idea of dimensionality reduction.

Projections onto one dimensional subspaces

Assume we are given a line (one-dimensional subspace) through the origin with basis vector $\boldsymbol{b} \in \mathbb{R}^n$. The line is a one-dimensional subspace $U \subseteq \mathbb{R}^n$ spanned by $\boldsymbol{b}$. When we project $\boldsymbol{x} \in \mathbb{R}^n$ onto U, we seek the vector $\pi_U(\boldsymbol{x}) \in U$ that is closest to $\boldsymbol{x}$. Using geometric arguments, let



(a) Projection of $\boldsymbol{x} \in \mathbb{R}^2$ onto a subspace U with basis vector $\boldsymbol{b}$.



(b) Projection of a two-dimensional vector $\boldsymbol{x}$ with $\|\boldsymbol{x}\| = 1$ onto a one-dimensional subspace spanned by $\boldsymbol{b}$.

Projections onto one dimensional subspaces

$$\pi_U(\boldsymbol{x}) = \lambda \boldsymbol{b} = \boldsymbol{b}\lambda = \boldsymbol{b}\frac{\boldsymbol{b}^\top \boldsymbol{x}}{\|\boldsymbol{b}\|^2} = \frac{\boldsymbol{b}\boldsymbol{b}^\top}{\|\boldsymbol{b}\|^2}\boldsymbol{x}\,,$$

$$\pi_U(\boldsymbol{x}) = \boldsymbol{P}_\pi \boldsymbol{x}.$$

Remark. The projection $\pi_U(\boldsymbol{x}) \in \mathbb{R}^n$ is still an n-dimensional vector and not a scalar. However, we no longer require n coordinates to represent the projection, but only a single one if we want to express it with respect to the basis vector $\boldsymbol{b}$ that spans the subspace U: λ. ♢

Projections onto a two dimensional subspace

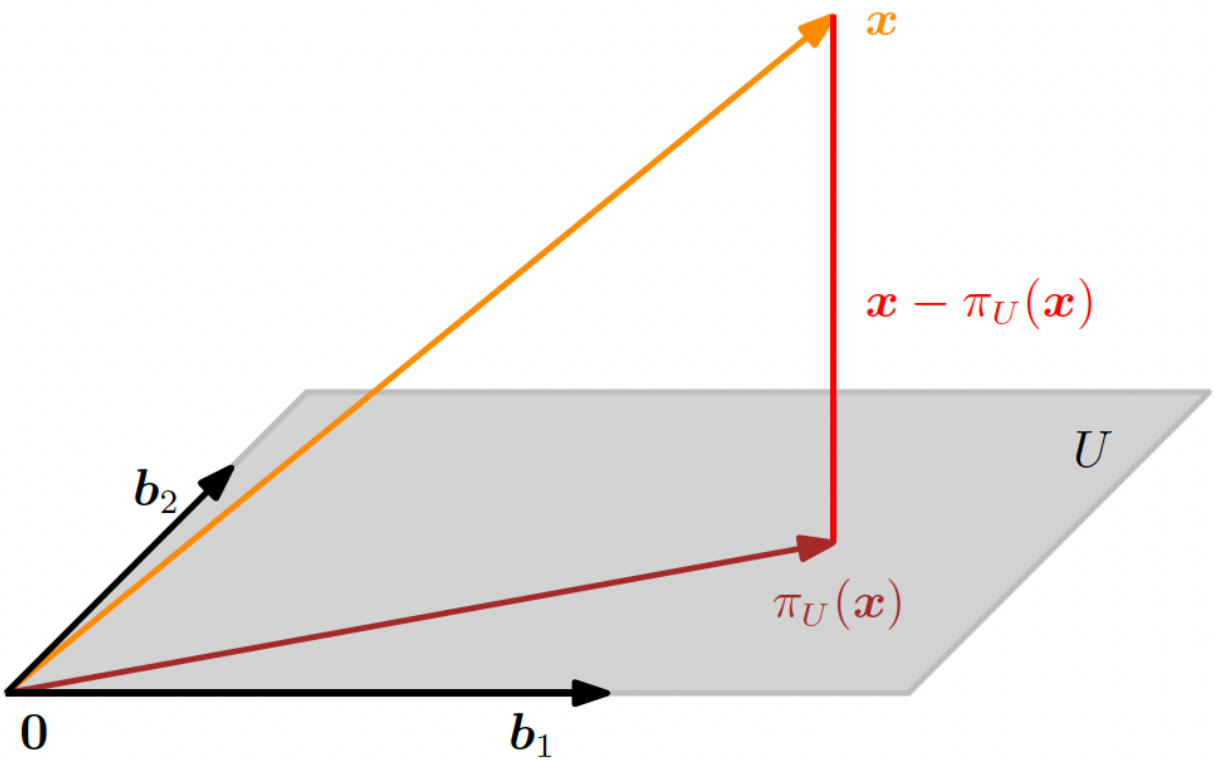


Figure 3.11 Projection onto a two-dimensional subspace U with basis $\boldsymbol{b}_1, \boldsymbol{b}_2$. The projection $\pi_U(\boldsymbol{x})$ of $\boldsymbol{x} \in \mathbb{R}^3$ onto U can be expressed as a linear combination of $\boldsymbol{b}_1, \boldsymbol{b}_2$ and the displacement vector $\boldsymbol{x} - \pi_U(\boldsymbol{x})$ is orthogonal to both $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$.

Projections onto a general subspace

- Projections allow us to look at situations where we have a linear system $Ax = b$ without a solution. Recall that this means that b does not lie in the span of A, i.e., the vector b does not lie in the subspace spanned by the columns of A.
- Given that the linear equation cannot be solved exactly, we can find an approximate solution
- The idea is to find the vector in the subspace spanned by the columns of A that is closest to b, i.e., we compute the orthogonal projection of b onto the subspace spanned by the columns of A.
- This problem arises often in practice, and the solution is called the least-squares (assuming the dot product as the inner product) of solution an overdetermined system

Matrix decompositions : how to describe a matrix with few important numbers(ch4 MML book)

- Data is often represented in matrix form as well, e.g., where the rows of the matrix represent different people and the columns describe different.
- We consider: how to summarize matrices, how matrices can be decomposed, and how these decompositions can be used for matrix approximations.

Matrix decompositions

A matrix can be characterized by few important numbers:

1. Eigenvalues (and associated eigenvectors)
2. Singular values (and associated singular vectors)

These values are computed through matrix factorizations.

Eigenvalues and eigenvectors

transformed by the linear mapping.

Definition 4.6. Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ be a square matrix. Then $\lambda \in \mathbb{R}$ is an *eigenvalue* of $\boldsymbol{A}$ and $\boldsymbol{x} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$ is the corresponding *eigenvector* of $\boldsymbol{A}$ if

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \,. \tag{4.25}$$

We call (4.25) the *eigenvalue equation*.

Eigenvalues and eigenvectors

Theorem 4.8. *$\lambda \in \mathbb{R}$ is eigenvalue of $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ if and only if λ is a root of the characteristic polynomial $p_{\boldsymbol{A}}(\lambda)$ of $\boldsymbol{A}$.*

Definition 4.9. Let a square matrix $\boldsymbol{A}$ have an eigenvalue λ_i. The *algebraic multiplicity* of λ_i is the number of times the root appears in the characteristic polynomial.

Definition 4.10 (Eigenspace and Eigenspectrum)**.** For $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, the set of all eigenvectors of $\boldsymbol{A}$ associated with an eigenvalue λ spans a subspace of $\mathbb{R}^n$, which is called the *eigenspace* of $\boldsymbol{A}$ with respect to λ and is denoted by E_λ. The set of all eigenvalues of $\boldsymbol{A}$ is called the *eigenspectrum*, or just *spectrum*, of $\boldsymbol{A}$.

Eigenvalues and eigenvectors

Definition 4.11. Let λ_i be an eigenvalue of a square matrix $\boldsymbol{A}$. Then the *geometric multiplicity* of λ_i is the number of linearly independent eigenvectors associated with λ_i. In other words, it is the dimensionality of the eigenspace spanned by the eigenvectors associated with λ_i.

Remark. A specific eigenvalue's geometric multiplicity must be at least one because every eigenvalue has at least one associated eigenvector. An eigenvalue's geometric multiplicity cannot exceed its algebraic multiplicity, but it may be lower. ♢

Example 4.6

The matrix $\boldsymbol{A} = \begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix}$ has two repeated eigenvalues $\lambda_1 = \lambda_2 = 2$ and an algebraic multiplicity of 2. The eigenvalue has, however, only one distinct unit eigenvector $\boldsymbol{x}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and, thus, geometric multiplicity 1.

Eigenvalues and eigenvectors

Theorem 4.15 (Spectral Theorem). *If $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is symmetric, there exists an orthonormal basis of the corresponding vector space V consisting of eigenvectors of $\boldsymbol{A}$, and each eigenvalue is real.*

A direct implication of the spectral theorem is that the eigendecomposition of a symmetric matrix $\boldsymbol{A}$ exists (with real eigenvalues), and that we can find an ONB of eigenvectors so that $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top$, where $\boldsymbol{D}$ is diagonal and the columns of $\boldsymbol{P}$ contain the eigenvectors.

Symmetric positive definite matrices

A square symmetric matrix $A \in \mathcal{R}^{n \times n}$ is said positive definite iff :

$$\forall x \in R^n, \quad x^T A x > 0$$

The eigenvalues of a symmetric positive definite matrix A are all positive.

Eigendecomposition and diagonalization

Theorem 4.20 (Eigendecomposition). *A square matrix* $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ *can be factored into*

$$\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1}, \tag{4.55}$$

where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ *and* $\boldsymbol{D}$ *is a diagonal matrix whose diagonal entries are the eigenvalues of* $\boldsymbol{A}$*, if and only if the eigenvectors of* $\boldsymbol{A}$ *form a basis of* $\mathbb{R}^n$.

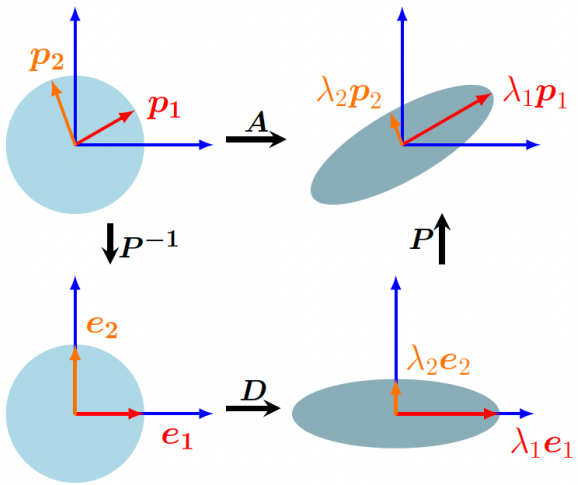
Eigendecomposition and diagonalization

Theorem 4.20 implies that only non-defective matrices can be diagonalized and that the columns of $\boldsymbol{P}$ are the n eigenvectors of $\boldsymbol{A}$. For symmetric matrices we can obtain even stronger outcomes for the eigenvalue decomposition.

Theorem 4.21. *A symmetric matrix $\boldsymbol{S} \in \mathbb{R}^{n \times n}$ can always be diagonalized.*

Theorem 4.21 follows directly from the spectral theorem 4.15. Moreover, the spectral theorem states that we can find an ONB of eigenvectors of $\mathbb{R}^n$. This makes $\boldsymbol{P}$ an orthogonal matrix so that $\boldsymbol{D} = \boldsymbol{P}^\top \boldsymbol{A} \boldsymbol{P}$.

Geometrical interpretation of eigendecomposition



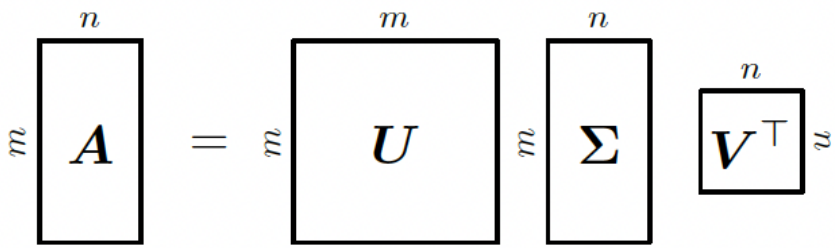
We can interpret the eigendecomposition of a matrix as follows (see also Figure 4.7): Let $\boldsymbol{A}$ be the transformation matrix of a linear mapping with respect to the standard basis. $\boldsymbol{P}^{-1}$ performs a basis change from the standard basis into the eigenbasis. This identifies the eigenvectors $\boldsymbol{p}_i$ (red and orange arrows in Figure 4.7) onto the standard basis vectors $\boldsymbol{e}_i$. Then, the diagonal $\boldsymbol{D}$ scales the vectors along these axes by the eigenvalues λ_i. Finally, $\boldsymbol{P}$ transforms these scaled vectors back into the standard/canonical coordinates yielding $\lambda_i \boldsymbol{p}_i$.

Singular Value Decomposition (SVD)

- The Singular Value Decomposition (SVD) can be applied to all the amtrices (not limited to square) and it always exists.
- It represents a linear mapping $\Phi : V \rightarrow W$ and quantifies the chenage of the geometry of the two vector spaces.

SVD

Theorem 4.22 (SVD Theorem). *Let $\boldsymbol{A}^{m\times n}$ be a rectangular matrix of rank $r \in [0, \min(m, n)]$. The SVD of $\boldsymbol{A}$ is a decomposition of the form*



(4.64)

with an orthogonal matrix $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ with column vectors $\boldsymbol{u}_i$, $i = 1, \ldots, m$, and an orthogonal matrix $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ with column vectors $\boldsymbol{v}_j$, $j = 1, \ldots, n$. Moreover, $\boldsymbol{\Sigma}$ is an $m \times n$ matrix with $\Sigma_{ii} = \sigma_i \geqslant 0$ and $\Sigma_{ij} = 0$, $i \neq j$.

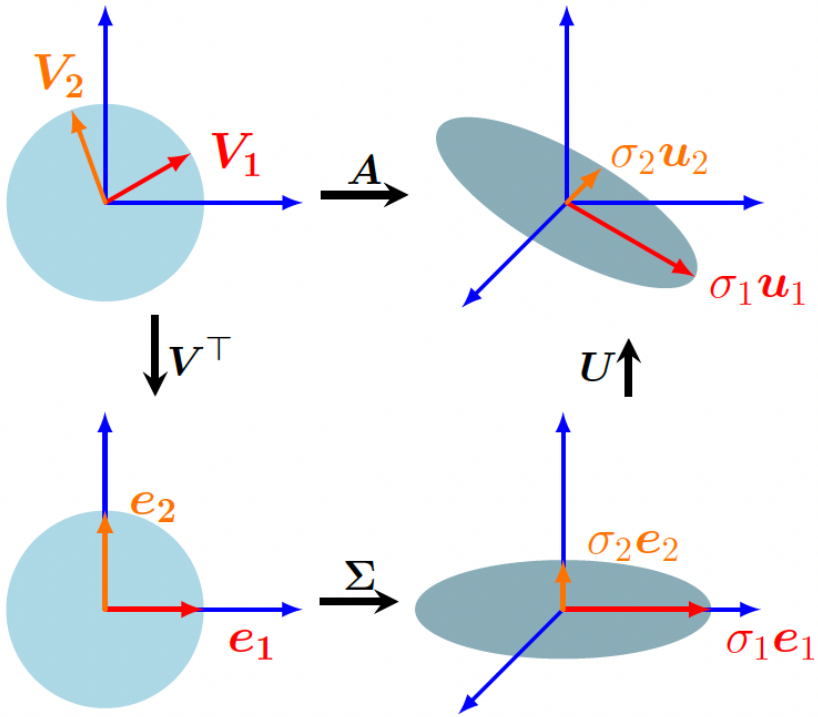
SVD

The diagonal entries σ_i, $i = 1, \ldots, r$, of $\boldsymbol{\Sigma}$ are called the *singular values*, $\boldsymbol{u}_i$ are called the *left-singular vectors*, and $\boldsymbol{v}_j$ are called the *right-singular vectors*. By convention, the singular values are ordered, i.e., $\sigma_1 \geqslant \sigma_2 \geqslant \sigma_r \geqslant 0$.

The *singular value matrix* $\boldsymbol{\Sigma}$ is unique, but it requires some attention. Observe that the $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is rectangular. In particular, $\boldsymbol{\Sigma}$ is of the same size as $\boldsymbol{A}$. This means that $\boldsymbol{\Sigma}$ has a diagonal submatrix that contains the singular values and needs additional zero padding. Specifically, if $m > n$,

Geometrical interpretation of SVD

- The SVD of a matrix can be interpreted as a decomposition of a corresponding linear mapping $\Phi : \mathcal{R}^n \rightarrow \mathcal{R}^m$ into three operations.
- It performs a change of basis through V^T, followed by a scaling through Σ and a second basis change via U.



Eigenvalues vs. Singular values

- The SVD and the eigendecomposition are closely related through their projections
  - The left-singular vectors of $\boldsymbol{A}$ are eigenvectors of $\boldsymbol{A}\boldsymbol{A}^\top$
  - The right-singular vectors of $\boldsymbol{A}$ are eigenvectors of $\boldsymbol{A}^\top\boldsymbol{A}$.
  - The nonzero singular values of $\boldsymbol{A}$ are the square roots of the nonzero eigenvalues of $\boldsymbol{A}\boldsymbol{A}^\top$ and are equal to the nonzero eigenvalues of $\boldsymbol{A}^\top\boldsymbol{A}$.
- For symmetric matrices $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, the eigenvalue decomposition and the SVD are one and the same, which follows from the spectral theorem 4.15.

SVD

The SVD is used in a variety of applications in machine learning from least-squares problems in curve fitting to solving systems of linear equations. These applications harness various important properties of the SVD, its relation to the rank of a matrix, and its ability to approximate matrices of a given rank with lower-rank matrices. Substituting a matrix with its SVD has often the advantage of making calculation more robust to numerical rounding errors. As we will explore in the next section, the SVD's ability to approximate matrices with "simpler" matrices in a principled manner opens up machine learning applications ranging from dimensionality reduction and topic modeling to data compression and clustering.

Matrix approximation by SVD

4.6 Matrix Approximation

We considered the SVD as a way to factorize $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \in \mathbb{R}^{m\times n}$ into the product of three matrices, where $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ are orthogonal and $\boldsymbol{\Sigma}$ contains the singular values on its main diagonal. Instead of doing the full SVD factorization, we will now investigate how the SVD allows us to represent a matrix $\boldsymbol{A}$ as a sum of simpler (low-rank) matrices $\boldsymbol{A}_i$, which lends itself to a matrix approximation scheme that is cheaper to compute than the full SVD.

We construct a rank-1 matrix $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$ as

$$\boldsymbol{A}_i := \boldsymbol{u}_i \boldsymbol{v}_i^\top , \tag{4.90}$$

Matrix approximation by SVD

A matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ of rank r can be written as a sum of rank-1 matrices $\boldsymbol{A}_i$ so that

$$\boldsymbol{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top = \sum_{i=1}^{r} \sigma_i \boldsymbol{A}_i \,, \tag{4.91}$$

We define the rank-k approximation of A

$$\widehat{\boldsymbol{A}}(k) := \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top = \sum_{i=1}^{k} \sigma_i \boldsymbol{A}_i$$

with rank $\hat{A}(k) = k$.

The spectral norm

Definition 4.23 (Spectral Norm of a Matrix)**.** For $\boldsymbol{x} \in \mathbb{R}^n \backslash \{\mathbf{0}\}$, the *spectral norm* of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is defined as

$$\|\boldsymbol{A}\|_2 := \max_{\boldsymbol{x}} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} . \tag{4.93}$$

We introduce the notation of a subscript in the matrix norm (left-hand side), similar to the Euclidean norm for vectors (right-hand side), which has subscript 2. The spectral norm (4.93) determines how long any vector $\boldsymbol{x}$ can at most become when multiplied by $\boldsymbol{A}$.

Theorem 4.24. *The spectral norm of $\boldsymbol{A}$ is its largest singular value σ_1.*

Matrix approximation by SVD

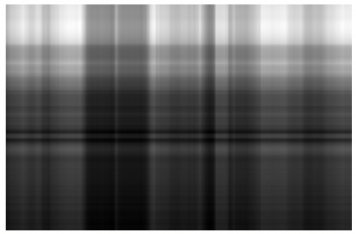
Theorem 4.25 (Eckart-Young Theorem (Eckart and Young, 1936)). *Consider a matrix* $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ *of rank* r *and let* $\boldsymbol{B} \in \mathbb{R}^{m \times n}$ *be a matrix of rank* k*. For any* $k \leqslant r$ *with* $\widehat{\boldsymbol{A}}(k) = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top$ *it holds that*

$$\widehat{\boldsymbol{A}}(k) = \operatorname{argmin}_{\operatorname{rk}(\boldsymbol{B})=k} \|\boldsymbol{A} - \boldsymbol{B}\|_2 \ , \tag{4.94}$$

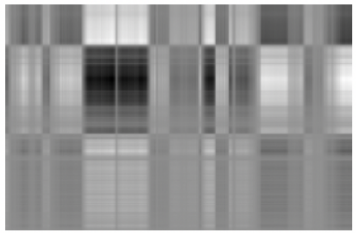
$$\left\|\boldsymbol{A} - \widehat{\boldsymbol{A}}(k)\right\|_2 = \sigma_{k+1} \ . \tag{4.95}$$

Example of image approximation by SVD

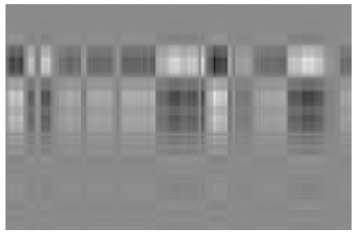
(a) Original image $\boldsymbol{A}$.



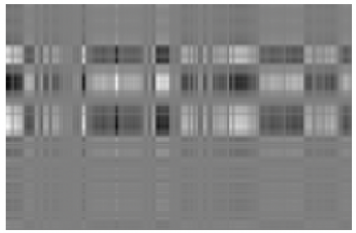
(b) $\boldsymbol{A}_1,\ \sigma_1 \approx 228,052.$



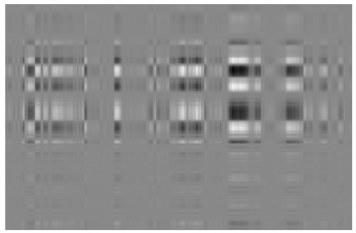
(c) $\boldsymbol{A}_2,\ \sigma_2 \approx 40,647.$



(d) $\boldsymbol{A}_3,\ \sigma_3 \approx 26,125.$

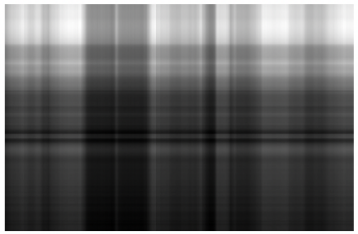


(e) $\boldsymbol{A}_4,\ \sigma_4 \approx 20,232.$

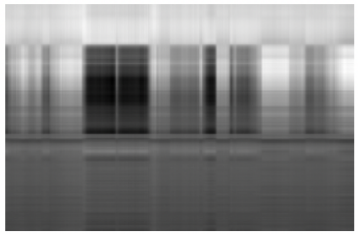


(f) $\boldsymbol{A}_5,\ \sigma_5 \approx 15,436.$

(a) Original image $\boldsymbol{A}$.



(b) Rank-1 approximation $\widehat{\boldsymbol{A}}(1)$.



(c) Rank-2 approximation $\widehat{\boldsymbol{A}}(2)$.

(d) Rank-3 approximation $\widehat{\boldsymbol{A}}(3)$.

(e) Rank-4 approximation $\widehat{\boldsymbol{A}}(4)$.

(f) Rank-5 approximation $\widehat{\boldsymbol{A}}(5)$.

Outline

Introduction to probability (Ch. 6 MML book)

- We often quantify uncertainty in the data, uncertainty in the machine learning model, and uncertainty in the predictions produced by random variable the model.
- Quantifying uncertainty requires the idea of a random variable, which is a function that maps outcomes of random experiments to a set of properties that we are interested in.
- Associated with the random variable is a function that measures the probability that a particular outcome (or set of outcomes) will occur; this is called the probability distribution.

Introduction to probability

- The theory of probability aims at defining a mathematical structure to describe random outcomes of experiments.
- For example, when tossing a single coin, we cannot determine the outcome, but by doing a large number of coin tosses, we can observe a regularity in the average outcome.
- Using this mathematical structure of probability, the goal is to perform automated reasoning, and in this sense, probability generalizes logical reasoning

Bayesian and frequentist probability interpretations

In machine learning and statistics, there are two major interpretations of probability: the Bayesian and frequentist interpretations

- The Bayesian interpretation uses probability to specify the degree of uncertainty that the user has about an event. It is sometimes referred to as "subjective probability" or "degree of belief".
- The frequentist interpretation considers the relative frequencies of events of interest to the total number of events that occurred. The probability of an event is defined as the relative frequency of the event in the limit when one has infinite data.

Sample space, events and probability

- The sample space is the set of all possible outcomes of the experiment, sample space usually denoted by Ω. For example, two successive coin tosses have a sample space of $\{hh, tt, ht, th\}$ where "h" denotes "heads" and "t" denotes "tails".
- The event space $\mathcal{A}$ is the space of potential results of the experiment. The event space $\mathcal{A}$ is obtained by considering the collection of subsets of Ω.
- With each event $A \in \mathcal{A}$, we associate a number P(A) that measures the probability or degree of belief that the event will occur. P(A) is called the probability of A.
- The probability of a single event A must lie in the interval [0; 1], and the total probability over all outcomes in the sample space must be 1, i.e., $P(\Omega) = 1$

Random variable

- However, we mostly do not work directly with this basic probability space. Instead, we work with random variables (the second idea), which transfers the probability to a more convenient (often numerical) space.
- In machine learning, we often avoid explicitly referring to the probability space, but instead refer to probabilities on quantities of interest, which we denote by T .
- We refer to T as the target space and refer to elements of T as states.
- We introduce a target space function $X : \Omega \to T$ and that takes an element of Ω (an event) and returns a particular quantity of interest x, a value in T .
- This association/mapping from to T is called a random variable.
- A random variable X is said discrete or continuous if its target space T is a discrete or continuos set.

Random variable

For example, in the case of tossing two coins and counting the number of heads, a random variable X maps to the three possible outcomes: X(hh) = 2, X(ht) = 1, X(th) = 1, and X(tt) = 0. In this particular case, $T = \{0; 1; 2\}$, and it is the probabilities on elements of T that we are interested in.

Discrete random variables

- When the target space T is discrete, we can specify the probability that a random variable X takes a particular value $x \in T$, denoted as $P(X = x)$.
- The expression P(X = x) for a discrete random variable X is known as the probability mass function.

Joint probability of two discrete random variables

- If we have two random variables X and Y with values $(x_1, \dots x_n)$ and $(y_1, \dots y_m)$, respectively.
- The target space of the joint probability is the Cartesian product of the target spaces of each of joint probability the random variables.
- The joint probability $p(x, y)$ is the probability of the intersection of both events, that is, $P(X = x_i; Y = y_j) = P(X = x_i \cap Y = y_j)$.
- The marginal probability that X takes the value x irrespective of the value marginal probability of random variable Y is (lazily) written as $p(x)$.
- If we consider only the instances where $X = x$, then the fraction of instances the conditional probability, for which $Y = y$ is written (lazily) as $p(y|x)$.

Discrete probability distributions

- In machine learning, we use discrete probability distributions to model categorical variable categorical variables, i.e., variables that take a finite set of unordered values.
- They could be categorical features, such as the degree taken at university when used for predicting the salary of a person, or categorical labels, such as letters of the alphabet when doing handwriting recognition.
- Discrete distributions are also often used to construct probabilistic models that combine a finite number of continuous distributions.

Continuous random variables

- We consider real-valued random variables, i.e., we consider target spaces that are intervals of the real line.
- The distribution function for a continuous random variable is called density function.

Observe that the probability density function is any function f that is non-negative and integrates to one. We associate a random variable X with this function f by

$$P(a \leqslant X \leqslant b) = \int_a^b f(x)\mathrm{d}x\,, \tag{6.16}$$

where $a, b \in \mathbb{R}$ and $x \in \mathbb{R}$ are outcomes of the continuous random variable X. States $\boldsymbol{x} \in \mathbb{R}^D$ are defined analogously by considering a vector of $x \in \mathbb{R}$. This association (6.16) is called the *law* or *distribution* of the random variable X.

Multivariate random variables

- We refer to univariate distribution to refer to distributions of a single random variable
- We will refer to distributions of more than one random variable as multivariate distributions, and will usually consider a vector of random variables $\in \mathcal{R}^D, D > 1$.
- All the previous definitions and results given for univariate random variables can be easily extended to multivariate random variables.

Bayes' theorem

- In machine learning and Bayesian statistics, we are often interested in making inferences of unobserved (latent) random variables given that we have observed other random variables.
- Let us assume we have some prior knowledge p(x) about an unobserved random variable x and some relationship $p(y|x)$ between x and a second random variable y, which we can observe.
- If we observe y, we can use Bayes' theorem to draw some conclusions about x given the observed values of y.

Bayes' theorem

Bayes' theorem (also Bayes' theorem Bayes' rule or Bayes' law) Bayes' rule:

$$\underbrace{p(\boldsymbol{x} \mid \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y} \mid \boldsymbol{x})}^{\text{likelihood}} \overbrace{p(\boldsymbol{x})}^{\text{prior}}}{\underbrace{p(\boldsymbol{y})}_{\text{evidence}}}$$

- $p(x)$ is the prior, which encapsulates our subjective prior prior knowledge of the unobserved (latent) variable x before observing any data. We can choose any prior that makes sense to us, but it is critical to ensure that the prior has nonzero pdf (or pmf) on all plausible x.
- The likelihood $p(y|x)$ describes how x and y are related, and in the likelihood case of discrete probability distributions, it is the probability of the data y if we were to know the latent variable x.
- The posterior $p(x|y)$ is the quantity of interest in Bayesian statistics posterior because it expresses exactly what we are interested in, i.e., what we know about x after having observed y.

Bayes' theorem

- The quantity:

$$p(\boldsymbol{y}) := \int p(\boldsymbol{y} \mid \boldsymbol{x}) p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_X[p(\boldsymbol{y} \mid \boldsymbol{x})]$$

  is the marginal likelihood/evidence.
- Bayes' theorem allows us to invert the relationship between x and y given by the likelihood. Therefore, Bayes' theorem is sometimes probabilistic inverse called the probabilistic inverse
- In Bayesian statistics, the posterior distribution is the quantity of interest as it encapsulates all available information from the prior and the data. Instead of carrying the posterior around, it is possible to focus on some statistic of the posterior, such as the maximum of the posterior,

Mean of a random variable

The mean of a random variable X (also called population mean) (also defined as the mean of the probability distribution p of X) is defined as:

$$\mu = E[x] = \sum_{x_i \in \mathcal{X}} x_i p(x_i)$$

if X is a discrete random variable ($\mathcal{X}$is the target space of X) and

$$\mu = E[x] = \int_{\mathcal{X}} x p(x) dx$$

if X is a continuous random variable.

Variance and standard deviation of a random variable

The variance of a random variable X (also called population variance) (also defined as the variance of the probability distribution p of X) is defined as:

$$\sigma^2 = V[x] = \sum_{x_i \in \mathcal{X}} (x_i - E[x])^2 p(x_i)$$

if X is a discrete random variable ($\mathcal{X}$is the target space of X) and

$$\sigma^2 = E[x] = \int_{\mathcal{X}} (x - E[x])^2 p(x) dx$$

if X is a continuous random variable.
The standard deviation of a random variable X (also called population standard deviation) (also defined as the standard deviation of the probability distribution p of X) is defined as:

$$\sigma = \sqrt{\sigma^2}$$

Covariance between two random variables

Definition 6.5 (Covariance (Univariate)). The *covariance* between two univariate random variables $X, Y \in \mathbb{R}$ is given by the expected product of their deviations from their respective means, i.e.,

$$\mathrm{Cov}_{X,Y}[x, y] := \mathbb{E}_{X,Y}\left[(x - \mathbb{E}_X[x])(y - \mathbb{E}_Y[y])\right] . \tag{6.35}$$

By using the linearity of expectations, the expression in Definition 6.5 can be rewritten as the expected value of the product minus the product of the expected values, i.e.,

$$\mathrm{Cov}[x, y] = \mathbb{E}[xy] - \mathbb{E}[x]\mathbb{E}[y] . \tag{6.36}$$

The covariance $C[x, x$ coincides with the variance $V[x]$.

Correlation between two random variables

Definition 6.8 (Correlation)**.** The *correlation* between two random variables X, Y is given by

$$\operatorname{corr}[x, y] = \frac{\operatorname{Cov}[x, y]}{\sqrt{\mathbb{V}[x]\mathbb{V}[y]}} \in [-1, 1] \,. \tag{6.40}$$

The covariance (and correlation) indicate how two random variables are related. Positive correlation *corr*$[x; y]$ means that when x grows, then y is also expected to grow. Negative correlation means that as x increases, then y decreases.

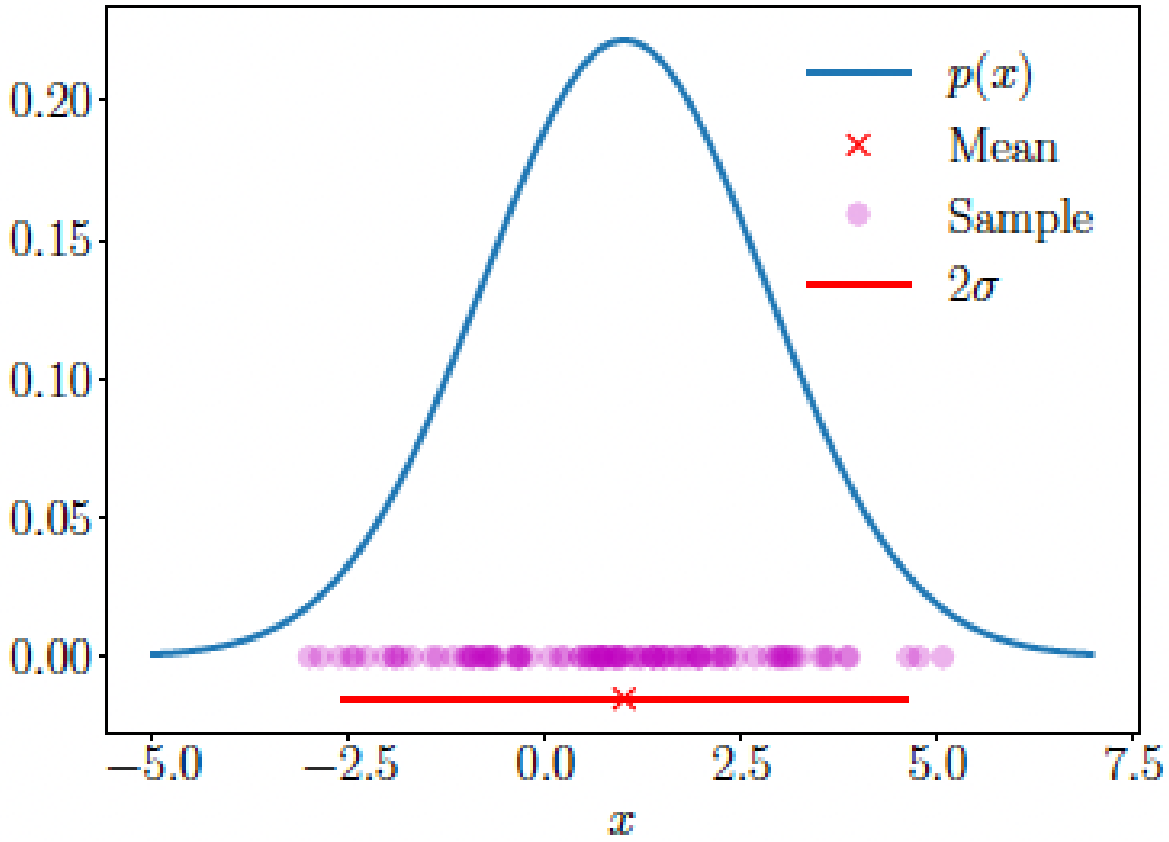
Gaussian distribution

- The Gaussian distribution is the most well-studied probability distribution for continuous-valued random variables.
- It is also referred to as the normal normal distribution distribution.
- Its importance originates from the fact that it has many computationally convenient properties.
- There are many other areas of machine learning that also benefit from using a Gaussian distribution, for example Gaussian processes, variational inference, and reinforcement learning

Gaussian distribution

For a univariate random variable, the Gaussian distribution has a density that is given by

$$p(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{6.62}$$



Statistics

- Probability theory and statistics are often presented together, but they concern different aspects of uncertainty
- One way of contrasting them is by the kinds of problems that are considered. Using probability, we can consider a model of some process, where the underlying uncertainty is captured by random variables, and we use the rules of probability to derive what happens.
- In statistics, we observe that something has happened and try to figure out the underlying process that explains the observations. In this sense, machine learning is close to statistics in its goals to construct a model that adequately represents the process that generated the data. We can use the rules of probability to obtain a "best-fitting" model for some data.

Statistics

- The previous definitions are often also called the population mean and variance, as it refers to the true statistics for the population.
- In ma- and covariance chine learning, we need to learn from empirical observations of data. Consider a random variable X. There are two conceptual steps to go from population statistics to the realization of empirical statistics.
- First, we use the fact that we have a finite dataset (of size N) to construct an empirical statistic that is a function of a finite number of identical random variables, $(X_1; \ldots; X_N)$.
- Second, we observe the data, that is, we look at the realization $x_1; \ldots x_N$ of each of the random variables and apply the empirical statistic.

Empirical mean and covariance

- In machine learning, we need to learn from empirical observations of data. Consider a random variable X.
- There are two conceptual steps to go from population statistics to the realization of empirical statistics. First, we use the fact that we have a finite dataset (of size N) to construct an empirical statistic that is a function of one (or more in the multivariate case) random variable.
- Second, we observe the data, that is, we look at the realization $x_1; \ldots; x_N$ of each of the random variables and apply the empirical statistic.

Descriptive statistics:Empirical mean and variance

Given a particular dataset we an obtain an estimate of the mean, which is called the empirical mean or sample mean. The same holds for the empirical covariance and variance.

Empirical mean The empirical mean is the average of the observations $x_1, \dots x_N$ of the random variable X dxefined as:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

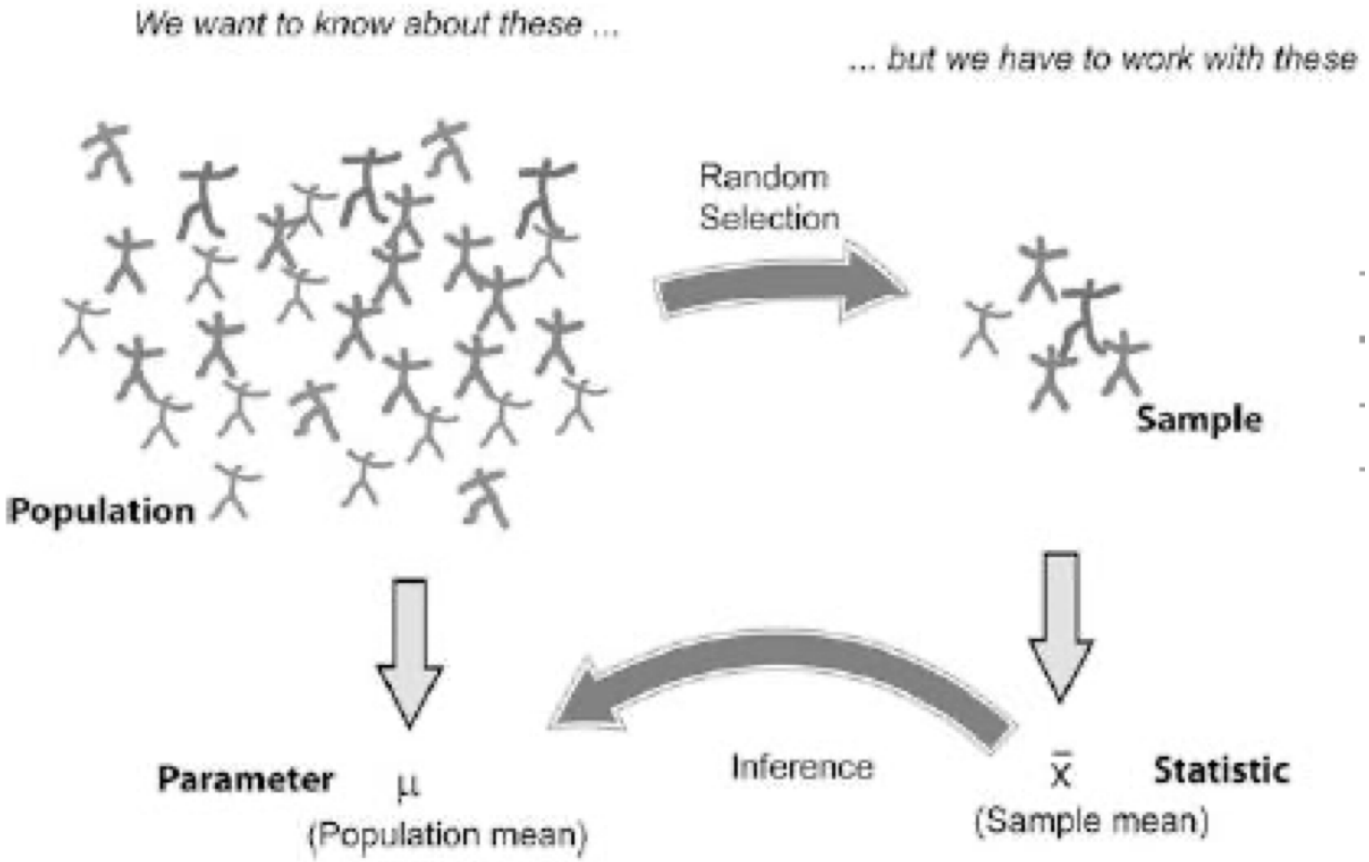
Empirical variance The empirical variance is defined as follows:

$$S^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2$$

Statistical inference

- Statistical inference is the process of using data analysis to infer properties of an underlying distribution of probability.
- Inferential statistical analysis infers properties of a population, for example by testing hypotheses and deriving estimates. It is assumed that the observed data set is sampled from a larger population.

Statistical inference



Sampling

The sample of size N chosen from the population is meaningful if:

- it is randomly chosen
- it is representative of the different features of the population
- the size N is adequate

In this case we call it a Simple Random Sample of size N (SRS(N))

Parameter estimates and confidence intervals

- The estimate of the population mean from a SRS(N) is made by the empirical mean.
- The empirical mean value depends on the particular sample considered.
- We define the confidence interval of probability *lpha* the interval I_α that contains the population mean value μ with a probability of $100(1-\alpha)\%$.

Confidence interval for the mean of a population with normal distribution

Suppose $x_1, \dots x_n$ a SRS(N) from a population with normal distribution $\mathcal{N}(\mu, \sigma)$ and $\bar{x}$ the estimated mean with the empirical mean formula.
Given $\alpha \in [0, 1]$, the confidence interval I_α is defined as:

$$I_\alpha = [\bar{x} - z_{\alpha/2}\frac{\sigma}{\sqrt{N}}, \bar{x} + z_{\alpha/2}\frac{\sigma}{\sqrt{N}}]$$

where $z_{\alpha/2}$ is the quantile of index $\alpha/2$ of the standard normal distribution, i.e. $\mathcal{N}(0, 1)$

Outline

Vector calculus (ch. 5 MML book)

In Machine Learning we deal with functions whose domain is in $\mathcal{R}^D$, with $D > 1$.

$$f : \mathbb{R}^D \to \mathbb{R}$$
$$\boldsymbol{x} \mapsto f(\boldsymbol{x})$$

Partial derivative and gradient

Definition 5.5 (Partial Derivative). For a function $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x} \mapsto f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^n$ of n variables $x_1, \ldots, x_n$ we define the *partial derivatives* as

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= \lim_{h \to 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(\boldsymbol{x})}{h} \\ &\vdots \\ \frac{\partial f}{\partial x_n} &= \lim_{h \to 0} \frac{f(x_1, \ldots, x_{n-1}, x_n + h) - f(\boldsymbol{x})}{h} \end{aligned} \tag{5.39}$$

and collect them in the row vector

$$\nabla_{\boldsymbol{x}} f = \operatorname{grad} f = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \left[\frac{\partial f(\boldsymbol{x})}{\partial x_1} \quad \frac{\partial f(\boldsymbol{x})}{\partial x_2} \quad \cdots \quad \frac{\partial f(\boldsymbol{x})}{\partial x_n} \right] \in \mathbb{R}^{1 \times n}, \tag{5.40}$$

An example

Example 5.7 (Gradient)
For $f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3 \in \mathbb{R}$, the partial derivatives (i.e., the derivatives of f with respect to x_1 and x_2) are

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 x_2 + x_2^3 \tag{5.43}$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = x_1^2 + 3x_1 x_2^2 \tag{5.44}$$

and the gradient is then

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f(x_1, x_2)}{\partial x_1} & \frac{\partial f(x_1, x_2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 x_2 + x_2^3 & x_1^2 + 3x_1 x_2^2 \end{bmatrix} \in \mathbb{R}^{1\times 2} . \tag{5.45}$$

Basic rules of partial differentiation

Product rule: $$\frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x})g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}}g(\boldsymbol{x}) + f(\boldsymbol{x})\frac{\partial g}{\partial \boldsymbol{x}}$$

Sum rule: $$\frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x}) + g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}} + \frac{\partial g}{\partial \boldsymbol{x}}$$

Chain rule: $$\frac{\partial}{\partial \boldsymbol{x}}(g \circ f)(\boldsymbol{x}) = \frac{\partial}{\partial \boldsymbol{x}}\big(g(f(\boldsymbol{x}))\big) = \frac{\partial g}{\partial f}\frac{\partial f}{\partial \boldsymbol{x}}$$

Example of chain rule

Consider a function $f : \mathbb{R}^2 \to \mathbb{R}$ of two variables x_1, x_2. Furthermore, $x_1(t)$ and $x_2(t)$ are themselves functions of t. To compute the gradient of f with respect to t, we need to apply the chain rule (5.48) for multivariate functions as

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1(t)}{\partial t} \\ \frac{\partial x_2(t)}{\partial t} \end{bmatrix} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t}, \tag{5.49}$$

where d denotes the gradient and ∂ partial derivatives.

Example 5.8

Consider $f(x_1, x_2) = x_1^2 + 2x_2$, where $x_1 = \sin t$ and $x_2 = \cos t$, then

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t} \tag{5.50a}$$

$$= 2\sin t \frac{\partial \sin t}{\partial t} + 2\frac{\partial \cos t}{\partial t} \tag{5.50b}$$

$$= 2\sin t \cos t - 2\sin t = 2\sin t(\cos t - 1) \tag{5.50c}$$

is the corresponding derivative of f with respect to t.

Backpropagation and automatic differentiation

In many machine learning applications it is necessary to compute the gradient of a learning objective function with respect to the parameters of the model and this is done by applying the chain rule.

Consider the function

$$f(x) = \sqrt{x^2 + \exp(x^2)} + \cos\left(x^2 + \exp(x^2)\right) . \tag{5.109}$$

By application of the chain rule, and noting that differentiation is linear, we compute the gradient

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}x} &= \frac{2x + 2x\exp(x^2)}{2\sqrt{x^2 + \exp(x^2)}} - \sin\left(x^2 + \exp(x^2)\right)\left(2x + 2x\exp(x^2)\right) \\ &= 2x\left(\frac{1}{2\sqrt{x^2 + \exp(x^2)}} - \sin\left(x^2 + \exp(x^2)\right)\right)\left(1 + \exp(x^2)\right) .\end{aligned}$$

Backpropagation and automatic differentiation

- Writing out the gradient in this explicit way is often impractical since it often results in a very lengthy expression for a derivative
- In practice, it means that, if we are not careful, the implementation of the gradient could be significantly more expensive than computing the function, which imposes unnecessary overhead.
- For training deep neural network models, the backpropagation algorithm is an efficient way to compute the gradient of an error function with respect to the parameters of the model.

Backpropagation and automatic differentiation

An area where the chain rule is used to an extreme is deep learning, where the function value $\boldsymbol{y}$ is computed as a many-level function composition

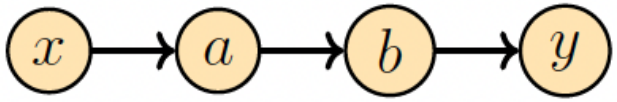
$$\boldsymbol{y} = (f_K \circ f_{K-1} \circ \cdots \circ f_1)(\boldsymbol{x}) = f_K(f_{K-1}(\cdots(f_1(\boldsymbol{x}))\cdots)), \quad (5.111)$$

where $\boldsymbol{x}$ are the inputs (e.g., images), $\boldsymbol{y}$ are the observations (e.g., class labels), and every function f_i, $i = 1, \ldots, K$, possesses its own parameters.

Backpropagation and automatic differentiation

- Backpropagation is a special case of a general technique in numerical analysis called automatic differentiation.
- We can think of automatic differentation as a set of techniques to numerically (in contrast to differentiation symbolically) evaluate the exact (up to machine precision) gradient of a function by working with intermediate variables and applying the chain rule.
- Automatic differentiation applies a series of elementary arithmetic operations, e.g., addition and multiplication and elementary functions, e.g., sin; cos; exp; log. By applying the chain rule to these operations, the gradient of quite complicated functions can be computed automatically
- Automatic differentiation applies to general computer programs and has forward and reverse modes.

Backpropagation and automatic differentiation



The Figure shows a simple graph representing the data flow from inputs x to outputs y via some intermediate variables a and b.
If we were to compute the derivative dy/dx, we would apply the chain rule and obtain:

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}y}{\mathrm{d}b}\frac{\mathrm{d}b}{\mathrm{d}a}\frac{\mathrm{d}a}{\mathrm{d}x}$$

Backpropagation and automatic differentiation

Intuitively, the forward and reverse mode differ in the order of multiplication. Due to the associativity of matrix multiplication, we can choose between

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \left(\frac{\mathrm{d}y}{\mathrm{d}b}\frac{\mathrm{d}b}{\mathrm{d}a}\right)\frac{\mathrm{d}a}{\mathrm{d}x}\,, \tag{5.120}$$

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}y}{\mathrm{d}b}\left(\frac{\mathrm{d}b}{\mathrm{d}a}\frac{\mathrm{d}a}{\mathrm{d}x}\right)\,. \tag{5.121}$$

Equation (5.120) would be the *reverse mode* because gradients are propagated backward through the graph, i.e., reverse to the data flow. Equation (5.121) would be the *forward mode*, where the gradients flow with the data from left to right through the graph.

Backpropagation and automatic differentiation

In the context of neural networks, where the input dimensionality is often much higher than the dimensionality of the labels, the reverse mode is computationally significantly cheaper than the forward mode. Let us start with an instructive example.

Backpropagation and automatic differentiation

Automatic differentiation is a formalization of Example 5.14. Let $x_1, \ldots, x_d$ be the input variables to the function, $x_{d+1}, \ldots, x_{D-1}$ be the intermediate variables, and x_D the output variable. Then the computation graph can be expressed as follows:

$$\text{For } i = d+1, \ldots, D: \quad x_i = g_i(x_{\text{Pa}(x_i)}), \tag{5.143}$$

Backpropagation and automatic differentiation

where the $g_i(\cdot)$ are elementary functions and $x_{\mathrm{Pa}(x_i)}$ are the parent nodes of the variable x_i in the graph. Given a function defined in this way, we can use the chain rule to compute the derivative of the function in a step-by-step fashion. Recall that by definition $f = x_D$ and hence

$$\frac{\partial f}{\partial x_D} = 1\,. \tag{5.144}$$

For other variables x_i, we apply the chain rule

$$\frac{\partial f}{\partial x_i} = \sum_{x_j : x_i \in \mathrm{Pa}(x_j)} \frac{\partial f}{\partial x_j}\frac{\partial x_j}{\partial x_i} = \sum_{x_j : x_i \in \mathrm{Pa}(x_j)} \frac{\partial f}{\partial x_j}\frac{\partial g_j}{\partial x_i}\,, \tag{5.145}$$

where $\mathrm{Pa}(x_j)$ is the set of parent nodes of x_j in the computation graph. Equation (5.143) is the forward propagation of a function, whereas (5.145) is the backpropagation of the gradient through the computation graph. For neural network training, we backpropagate the error of the prediction with respect to the label.

Ottimizzazione non vincolata: definizioni

Consideriamo il problema di **ottimizzazione non vincolata**:

$$\min_x f(x)$$

dove $x \in \mathcal{R}^n$ è un vettore reale di $n \geq 1$ componenti e la funzione obiettivo $f : \mathcal{R}^n \to \mathcal{R}$ è una funzione regolare.

- Un vettore x^* è un punto di **minimo locale** di $f(x)$ se esiste un $\epsilon > 0$ tale

$$f(x^*) \leq f(x) \quad \text{per ogni } x \text{ tale che } \|x - x^*\| < \epsilon.$$

- Un vettore x^* è un punto di **minimo globale** di $f(x)$ se

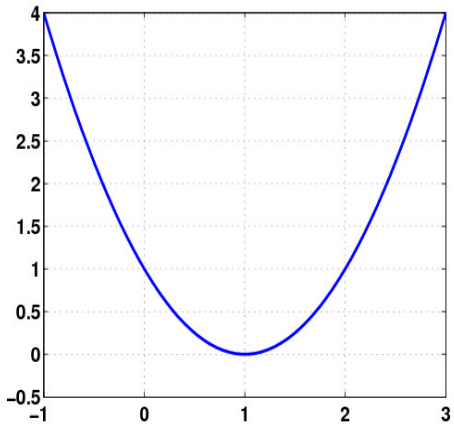
$$f(x^*) \leq f(x) \quad \text{per ogni } x \in \mathcal{R}^n.$$

- x^* è un punto di **minimo globale in senso stretto** di $f(x)$ se

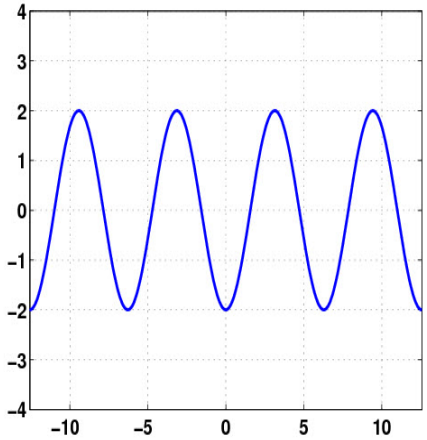
$$f(x^*) < f(x) \quad \forall x \in \mathcal{R}^n, \quad x \neq x^*.$$

Ottimizzazione non vincolata: definizioni

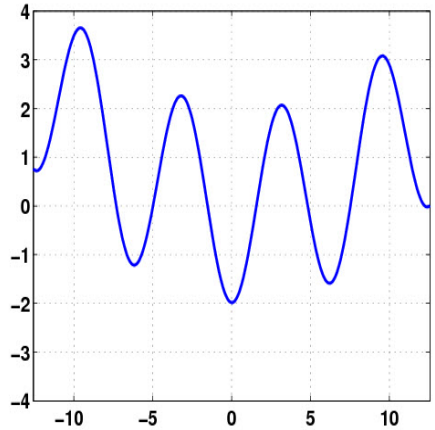
Una funzione $f(x)$ può avere un unico punto di minimo locale (quindi anche globale), oppure può non avere nè minimi locali nè globali, può avere sia minimi locali che globali...



(a) $y = (x - x^*)^2$ un unico punto di minimo



(b) $y = -2\cos(x - x^*)$ molti punti di minimo globale.



(c) $y = 0.015(x-x^*)^2 - 2\cos(x - x^*)$ un punto di minimo globale e molti punti di minimo locale.

Figure: Punti di minimo locale e globale

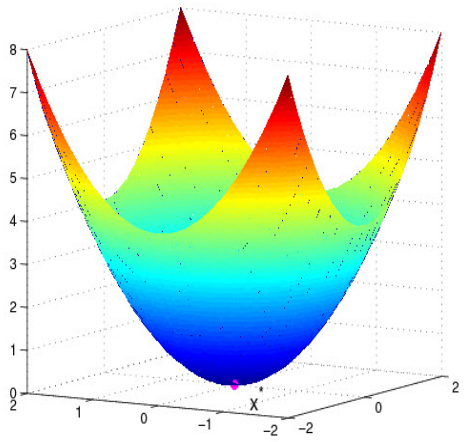
Ottimizzazione: algoritmi

- Le condizioni precedenti per i punti di minimo locale non hanno utilità pratica.
- Sarà però possibile determinare, con ipotesi aggiuntive sulla funzine obiettivo, delle condizioni di ottimalità utilizzabili in pratica.
- Gli algoritmi classici determinano minimi locali, in quanto la ricerca di minimi globali è molto complessa e costosa.

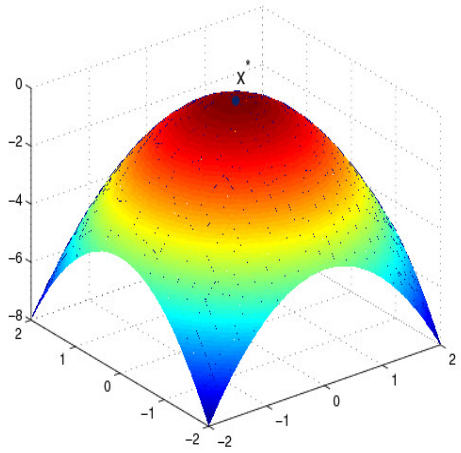
Condizioni di ottimalità

- Se x^* è un punto di minimo locale e f è differenziabile con continuità in un intorno aperto di x^*, allora $\nabla f(x^*) = 0$.
- Un punto x^* tale che $\nabla f(x^*) = 0$ è detto **punto stazionario**. Dal teorema segue che:
  x^* punto di minimo $\quad \Rightarrow \quad$ x^* punto stazionario
- La condizione $\nabla f(x^*)$ è condizione necessaria affinchè x^* sia un punto di minimo locale; tale condizione non è però sufficiente poichè un punto stazionario può essere un punto di minimo locale, un punto di massimo locale o un punto di sella.

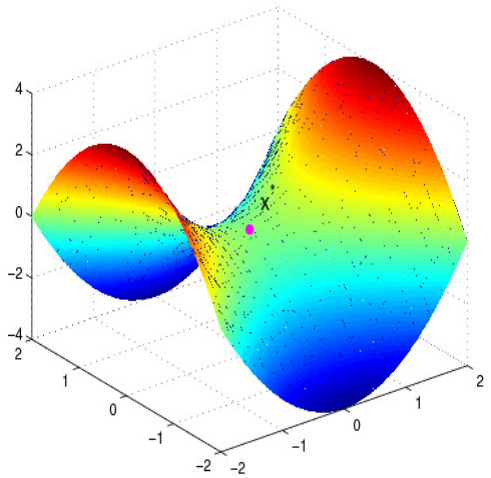
Condizioni di ottimalità



(a) Punto di minimo.



(b) Punto di massimo.



(c) Punto di sella.

Figure: Tipi di punti stazionari.

Funzioni convesse

Una funzione f è convessa in $\mathcal{R}^n$ se

$$f\big(\alpha x + (1-\alpha)y\big) \leq \alpha f(x) + (1-\alpha)f(y)$$

per ogni $0 \leq \alpha \leq 1$ e per ogni $x, y \in \mathcal{R}^n$.

- Se f è convessa, allora ogni punto di minimo locale x^* è un punto di minimo globale di f.
- Se f è strettamente convessa, allora esiste un unico punto di minimo globale.
- Se f è differenziabile e convessa, allora ogni punto stazionario x^* è un punto di minimo globale di f.

Caratteristiche degli algoritmi di ottimizzazione

Gli algoritmi di ottimizzazione sono **algoritmi iterativi**.

- Gli algoritmi iterativi calcolano una sequenza di iterati $x_1, x_2, \ldots$, a partire da un iterato iniziale x_0 assegnato, secondo una legge del tipo:

$$x_{k+1} = G(x_k).$$

- La successione $\{x_k\}$ deve avere proprietà di **convergenza** alla soluzione esatta x^*:

$$\lim_{k \to \infty} x_k = x^*$$

Generalità

- Tutti gli algoritmi per l'ottimizzazione non vincolata richiedono che l'utente fornisca un punto iniziale x_0. In generale, non esistono criteri generali per effettuare una buona scelta di x_0 e quindi si è interessati a definire algoritmi le cui proprietà di convergenza siano indipendenti dalla scelta del punto iniziale.
- A partire da x_0, gli algoritmi generano una successione di iterati $\{x_k\}_{k=0}^{\infty}$ che termina o quando non è possibile fare alcun progresso verso la soluzione o quando la soluzione è stata approssimata con sufficiente accuratezza.
- Gli algoritmi utilizzano informazioni sulla funzione obiettivo f in x_k e, a volte, informazioni dagli iterati precedenti per determinare un nuovo punto x_{k+1} con un valore più piccolo della funzione obiettivo.
- Esistono due strategie fondamentali per muoversi dall'iterato corrente x_k verso il nuovo iterato x_{k+1}: line search e trust region.

Convergenza a punti stazionari

- Gli algoritmi di ottimizzazione garantiscono, in generale, la determinazione di punti che soddisfano condizioni necessarie di ottimo.
- Tutti i metodi considerati consentono di determinare punti stazionari di f.
- Nel caso non convesso, la determinazione di punti stazionari non fornisce una soluzione globale e non è neanche possibile, in generale, garantire che sia raggiunto un punto di minimo locale.

Criteri di arresto e fallimenti

- Negli algoritmi di ottimizzazione non vincolata per criterio di arresto si intende il criterio che dovrebbe indicare il raggiungimento con successo di un punto stazionario con la tolleranza specificata dall'utente.
- Dal punto di vista teorico, il criterio di arresto di un algoritmo che genera la successione x_k dovrebbe essere la condizione

$$\|\nabla f(x_k)\| \leq \epsilon, \quad \epsilon > 0 \tag{1}$$

- Il raggiungimento di un numero massimo di iterazioni, che non garantisce la convergenza del metodo

Definizione di metodo di discesa

Sia dato il problema di minimizzare $f : \mathcal{R}^n \to \mathcal{R}$ differenziabile con continuità.
I metodi di discesa sono metodi iterativi che, a partire da un iterato iniziale $x_0 \in \mathcal{R}R^n$ e detta x^* la soluzione esatta del problema, generano una successione di vettori

$$x_0, x_1, x_2, \ldots,$$

definiti dall'iterazione

$$x_{k+1} = x_k + \alpha_k p_k \tag{2}$$

dove il vettore p_k è una direzione di ricerca e lo scalare α_k è un parametro positivo chiamato lunghezza del passo (step-length) che indica la distanza di cui ci si deve muovere lungo la direzione p_k.

Metodo di discesa

In un metodo di discesa, il vettore p_k ed il parametro α_k sono scelti in modo da garantire la decrescita di $f(x)$ ad ogni iterazione:

$$f(x_{k+1}) < f(x_k), \quad k = 0, 1, \ldots$$

- Il vettore p è una direzione di discesa di f in x se esiste un $\overline{\alpha} > 0$ tale che

$$f(x + \alpha p) < f(x) \text{ per ogni } \alpha \in (0, \overline{\alpha}]$$

- Quindi la retta $x = x_k + \alpha p_k$ deve formare un angolo ottuso con la direzione del gradiente (derivata direzionale negativa)

Direzione di discesa

Lemma.
Sia $f \in C^1$, il vettore p è una direzione di discesa di f in x se

$$p^T \nabla f(x) < 0$$

- I metodi di discesa utilizzano direzioni p_k di discesa poichè garantiscono una decrescita di f nella direzione di p_k.
- A seconda della scelta di p_k si hanno diversi metodi di discesa.

Interpretazione geometrica

Ricordando che il concetto di angolo tra due vettori $x \neq 0$ e $y \neq 0$ in $\mathbb{R}^n$ si può introdurre attraverso la definizione di coseno, ponendo

$$\cos\theta = \frac{x^T y}{\|x\|\|y\|}$$

si può dire che l'angolo tra p e $\nabla f(x)$ è

- *ottuso* se $p^T(x) < 0$
- *acuto* se $p^T(x) > 0$

I vettori p e $\nabla f(x)$ sono *ortogonali* se $p^T(x) = 0$.

Schema generale metodo di discea

Metodo di discesa

Dato x_0, per $k = 0, 1, 2, \ldots$

i) determinare una direzione di discesa p_k;

ii) determinare una lunghezza del passo α_k;

iii) porre $x_{k+1} = x_k + \alpha_k p_k$ e $k + 1 = k$;

Scelta della direzione di discesa

Direzione del gradiente.
Corrisponde alla scelta

$$p_k = -\nabla f(x), \text{ per ogni } k$$

In corrispondenza di questa scelta della direzione di discesa si ottengono i *metodi di tipo gradiente*

Scelta della lunghezza del passo

- Esistono diverse tecniche per determinare la lunghezza del passo in modo tale da garantire la convergenza del metodo di discesa verso punti stazionari di f.
- Esse sono, in generale, chiamate tecniche di *ricerca in linea* (line search) perchè la ricerca di un nuovo iterato x_{k+1} è fatta lungo la linea $y(\alpha) = x_k + \alpha p_k$.

Convergenza

- Diciamo **globalmente convergenti** gli algoritmi per cui vale la precedente condizione:

$$\lim_{k \to \infty} \|(x_k)\| = 0 \tag{3}$$

- Questo non significa che l'algoritmo converga ad un minimo globale, ma solo che converge ad un punto stazionario.
- Con le condizioni di ricerca in linea inesatte che abbiamo visto è la convergenza più forte che si può ottenere.