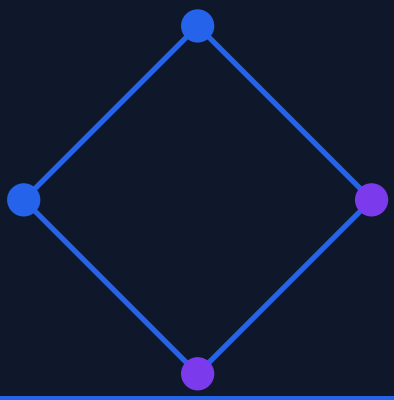
PMAI LAB

Mathematics • Python • Artificial Intelligence



LINEAR ALGEBRA FOR MACHINE LEARNING

Lecture 01

Introduction and Linear Systems for Machine Learning

Lecture focus Representing data with vectors and matrices, expressing linear systems as $A\mathbf{x} = \mathbf{b}$, identifying solution types, and analysing systems through determinant, rank, row reduction, and null space.

Dr. Hira Benish

Associate Professor of Mathematics

Student Learning Handout | Includes NumPy practice and review exercises

1 Lecture Overview

Linear algebra is the mathematical language used to represent data and perform computations in machine learning. Datasets, images, model parameters, and neural-network operations can all be expressed using vectors and matrices.

This lecture introduces basic mathematical objects, systems of linear equations, matrix form, solution types, determinants, rank, row reduction, and null space. Each idea is connected with its computational use in machine learning.

Learning Outcomes

After completing this lecture, students should be able to:

- explain why linear algebra is important in machine learning;
- distinguish among scalars, vectors, matrices, and tensors;
- represent a system of equations in the form $A\mathbf{x} = \mathbf{b}$;
- identify systems having a unique solution, infinitely many solutions, or no solution;
- explain singular and non-singular matrices;
- calculate a determinant and matrix rank using Python; and
- perform a basic neural-network matrix operation using NumPy.

1.1 Why Linear Algebra Matters in Machine Learning

Machine-learning systems use linear algebra to organize and process large amounts of data efficiently.

ML component	Linear-algebra representation
A single data record	Vector
Complete dataset	Matrix
Grayscale image	Matrix of pixel values
Colour image or image batch	Multidimensional tensor
Model parameters	Weight vectors or weight matrices
Neural-network layer	Matrix transformation followed by an activation
Dimensionality reduction	Eigenvectors or singular vectors

Machine-Learning Connection

If a dataset contains m observations and n features, it is commonly stored as a matrix $X \in \mathbb{R}^{m \times n}$. Rows represent observations and columns represent features. Understanding this shape convention is essential when writing machine-learning code.

2 Basic Mathematical Objects

2.1 Scalar

A **scalar** is a single numerical value, for example $a = 5$. In machine learning, a learning rate, loss value, model accuracy, or regularization parameter is a scalar.

2.2 Vector

A **vector** is an ordered collection of numbers. A student's age, GPA, and attendance may be written as

$$\mathbf{x} = \begin{bmatrix} 20 \\ 3.2 \\ 85 \end{bmatrix} \in \mathbb{R}^3.$$

The vector contains three features, so its dimension is 3.

2.3 Matrix

A **matrix** is a rectangular arrangement of numbers. Three students described by the same three features can form the data matrix

$$X = \begin{bmatrix} 20 & 3.2 & 85 \\ 21 & 3.5 & 90 \\ 19 & 2.9 & 78 \end{bmatrix} \in \mathbb{R}^{3\times 3}.$$

Each row is one student; each column is one feature. The **shape** of X is 3×3.

2.4 Tensor

A **tensor** is a multidimensional collection of numbers. A colour image can have shape $h \times w \times 3$, where h and w are its height and width and the last dimension contains red, green, and blue channels.

Notation Check

Symbol	Object	Example use
a	Scalar	Learning rate
$\mathbf{x}$	Vector	One observation or feature vector
A **or** X	Matrix	Dataset or weight matrix
$\mathcal{T}$	Tensor	Batch of images

2.5 Shape Compatibility

For matrix multiplication $A\mathbf{x}$, the number of columns of A must equal the number of entries in $\mathbf{x}$. If $A \in \mathbb{R}^{m\times n}$ and $\mathbf{x} \in \mathbb{R}^n$, then

$$A\mathbf{x} \in \mathbb{R}^m.$$

Important coding distinction: in NumPy, `A @ x` performs matrix multiplication, whereas `A * x` performs element-wise multiplication using broadcasting rules.

3 Systems of Linear Equations

Consider the system

$$x + y = 10,$$
$$x + 2y = 12.$$

The objective is to find values of x and y that satisfy both equations simultaneously.

3.1 Matrix Representation

The system can be written compactly as

$$A\mathbf{x} = \mathbf{b}, \qquad \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} 10 \\ 12 \end{bmatrix}}_{\mathbf{b}}.$$

A	Coefficient matrix; in ML it can represent features or a linear transformation.
$\mathbf{x}$	Unknown vector; in ML it can represent parameters or weights.
$\mathbf{b}$	Output or target vector.

Subtracting the first equation from the second gives $y = 2$. Substituting it into $x + y = 10$ gives $x = 8$. Therefore,

$$\mathbf{x} = \begin{bmatrix} 8 \\ 2 \end{bmatrix}.$$

3.2 Geometric Interpretation

In two variables, each equation represents a line. The solution of the system is the point shared by both lines.

- Lines intersect once $\Rightarrow$ one unique solution.
- Lines coincide $\Rightarrow$ infinitely many solutions.
- Lines are parallel and distinct $\Rightarrow$ no solution.

Machine-Learning Connection

The form $A\mathbf{x} = \mathbf{b}$ appears in regression, parameter estimation, optimization, signal processing, recommendation systems, and scientific computing. Even when an ML problem is not solved directly as a square system, the same matrix language remains central.

4 Types of Solutions

Let $[A \mid \mathbf{b}]$ denote the augmented matrix and let n be the number of unknowns.

Solution type	Rank condition	Meaning
Unique solution	$\text{rank}(A) = \text{rank}([A \mid \mathbf{b}]) = n$	Exactly one vector satisfies the system.
Infinitely many	$\text{rank}(A) = \text{rank}([A \mid \mathbf{b}]) < n$	At least one variable is free; some information is redundant.
No solution	$\text{rank}(A) < \text{rank}([A \mid \mathbf{b}])$	The equations are inconsistent.

4.1 Singular and Non-Singular Matrices

For a square matrix A:

Non-Singular Matrix

If $\det(A) \neq 0$, then A is non-singular. It has full rank, its inverse exists, and $A\mathbf{x} = \mathbf{b}$ has a unique solution for every compatible $\mathbf{b}$.

Singular Matrix

If $\det(A) = 0$, then A is singular. Its rows or columns are linearly dependent and its inverse does not exist. Depending on $\mathbf{b}$, the system may have infinitely many solutions or no solution.

Common misconception: a zero determinant does not automatically mean "no solution." It means a unique solution cannot be guaranteed. The ranks of A and $[A \mid \mathbf{b}]$ determine whether the system has infinitely many solutions or no solution.

4.2 Quick Examples

1. $x + y = 10$ and $2x + 2y = 20$: the equations describe the same line, so there are infinitely many solutions.
2. $x + y = 10$ and $2x + 2y = 25$: the equations contradict each other, so there is no solution.

In both examples, the coefficient matrix is singular. The different target vectors produce different solution types.

5 Row Reduction and Gaussian Elimination

Row reduction transforms a system into a simpler but equivalent system. The permitted elementary row operations are:

1. interchange two rows;
2. multiply a row by a non-zero scalar; and
3. add a multiple of one row to another row.

For the system in Section 3, begin with the augmented matrix

$$\left[\begin{array}{cc|c} 1 & 1 & 10 \\ 1 & 2 & 12 \end{array}\right].$$

Apply $R_2 \leftarrow R_2 - R_1$:

$$\left[\begin{array}{cc|c} 1 & 1 & 10 \\ 0 & 1 & 2 \end{array}\right].$$

The second row gives $y = 2$. Substituting it into the first row gives $x = 8$.

5.1 Echelon Forms

Row echelon form (REF) has all zero rows at the bottom and each leading entry appears to the right of the leading entry above it. **Reduced row echelon form (RREF)** additionally requires every pivot to be 1 and the only non-zero entry in its column.

The RREF of the augmented matrix is

$$\left[\begin{array}{cc|c} 1 & 0 & 8 \\ 0 & 1 & 2 \end{array}\right].$$

Machine-Learning Connection

Gaussian elimination is a fundamental computational procedure, but software libraries use optimized numerical algorithms. In practice, use a solver such as `np.linalg.solve` instead of explicitly computing $A^{-1}\mathbf{b}$. The solver is typically faster and numerically more reliable.

Numerical note: a determinant extremely close to zero may indicate that a matrix is nearly singular. In real datasets, floating-point tolerance matters; exact comparison with zero can be misleading.

6 Rank and Null Space

6.1 Rank

The **rank** of a matrix is the number of linearly independent rows or columns. It measures the amount of independent information carried by the matrix.

For $A \in \mathbb{R}^{m \times n}$,

$$\text{rank}(A) \leq \min(m, n).$$

A square matrix has full rank when $\text{rank}(A) = n$. A square full-rank matrix is non-singular.

Rank in Data Analysis

If columns of a data matrix repeat the same information, the matrix loses rank. This can indicate redundant or perfectly correlated features. Such redundancy may make parameter estimation unstable or non-unique.

6.2 Null Space

The **null space** of A is the set of all vectors satisfying

$$\mathcal{N}(A) = \{\mathbf{x} : A\mathbf{x} = \mathbf{0}\}.$$

The zero vector always belongs to the null space. If a non-zero vector also belongs to it, then A maps at least one non-zero direction to zero and therefore loses information in that direction.

For example,

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} -2 \\ 1 \end{bmatrix}, \qquad A\mathbf{v} = \mathbf{0}.$$

Thus $\mathbf{v}$ is a non-zero null-space vector, and A is singular.

6.3 Rank-Nullity Preview

For a matrix having n columns,

$$\text{rank}(A) + \text{nullity}(A) = n.$$

Rank counts independent output directions; nullity counts input directions that are lost.

Machine-Learning Connection

Rank and null space help us understand redundant features, model identifiability, compression, and dimensionality reduction. Later lectures will connect these concepts with basis, projections, least squares, eigenvectors, and SVD.

7 Python and NumPy Practice

7.1 Representing Vectors and Matrices

```python
import numpy as np

x = np.array([2, 4, 6])

A = np.array([
    [1, 2],
    [3, 4]
])

print("Vector:", x)
print("Matrix:\n", A)
print("Matrix shape:", A.shape)
```

7.2 Solving a Linear System

```python
A = np.array([
    [1, 1],
    [1, 2]
], dtype=float)

b = np.array([10, 12], dtype=float)

solution = np.linalg.solve(A, b)

print("Solution:", solution)
print("Verification:", A @ solution)
```

Expected output:

```
Solution: [8. 2.]
Verification: [10. 12.]
```

7.3 Determinant and Rank

```python
determinant = np.linalg.det(A)
rank = np.linalg.matrix_rank(A)

print("Determinant:", determinant)
print("Rank:", rank)
```

> Use `np.allclose(A @ solution, b)` when verifying floating-point results. It tests approximate equality and is safer than requiring every decimal value to match exactly.

8 Diagnosing Systems with Python

The following function compares the rank of the coefficient matrix with the rank of the augmented matrix.

```python
def classify_system(A, b):
    rank_A = np.linalg.matrix_rank(A)
    augmented = np.column_stack((A, b))
    rank_augmented = np.linalg.matrix_rank(augmented)
    variables = A.shape[1]

    if rank_A < rank_augmented:
        return "No solution"
    elif rank_A < variables:
        return "Infinitely many solutions"
    else:
        return "Unique solution"


print(classify_system(A, b))
```

8.1 Neural-Network Connection

A basic neural-network layer calculates

$$\mathbf{z} = W\mathbf{x} + \mathbf{b},$$

where **x** is the input vector, W is the weight matrix, **b** is the bias vector, and **z** is the output before activation.

```python
x = np.array([0.8, 0.3, 0.5])

W = np.array([
    [0.4, 0.7, 0.2],
    [0.6, 0.1, 0.9]
])

bias = np.array([0.1, 0.2])

z = W @ x + bias
relu_output = np.maximum(0, z)

print("Linear output:", z)
print("After ReLU:", relu_output)
```

Machine-Learning Connection

Every neuron forms a weighted combination of its inputs. A complete layer performs many such calculations simultaneously through matrix multiplication. This is why efficient linear-algebra libraries are central to modern deep learning.

9 Review and Independent Practice

Practice Tasks

1. Represent the system $2x + y = 7$ and $x - y = 2$ in the form $A\mathbf{x} = \mathbf{b}$.
2. Solve the system manually using row reduction, then verify the result with `np.linalg.solve`.
3. Find the determinant and rank of $A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$. Is it singular?
4. For the same matrix, compare $\mathbf{b}_1 = \begin{bmatrix} 3 \\ 6 \end{bmatrix}$ and $\mathbf{b}_2 = \begin{bmatrix} 3 \\ 7 \end{bmatrix}$. Classify both systems and explain why their results differ.
5. Change the values in the neural-layer weight matrix W. Record how the output changes and explain the role of the weights.
6. Create a 4×3 NumPy matrix. State how many observations and features it could represent under the standard ML convention.

9.1 Key Takeaways

- Machine-learning data and parameters are commonly represented using vectors, matrices, and tensors.
- A linear system is written compactly as $A\mathbf{x} = \mathbf{b}$.
- Rank measures independent information and helps determine the number of solutions.
- A non-singular square matrix has a unique solution for every compatible target vector.
- The null space describes input directions mapped to zero.
- NumPy provides efficient tools for solving and analysing linear systems.
- A neural-network layer is built from the matrix transformation $W\mathbf{x} + \mathbf{b}$.

9.2 Key Terms

Scalar	Vector	Matrix
Tensor	Linear system	Augmented matrix
Determinant	Rank	Null space
Singular matrix	Gaussian elimination	Neural layer

Next lecture: Vectors, linear combinations, span, and feature spaces in machine learning.