## **1. Linear Algebra**

### ****Vectors****

* A vector is an ordered collection of numbers (scalars), represented as a column or row.
* **Operations:** Addition, scalar multiplication, dot product, cross product (for 3D vectors).
* **Norms:**
  + **L1 Norm**:
  + **L2 Norm (Euclidean Norm)**:
  + **Module(numpy,scipy,pandas**: dot,rank,svd)

### ****Matrices****

* A matrix is a rectangular array of numbers.
* **Types of Matrices:** Square, diagonal, identity, symmetric, orthogonal.
* **Operations:** Addition, multiplication, transpose, determinant, inverse.
* **Inverse of a Matrix**: A matrix has an inverse if **. At**

### ****Eigenvalues & Eigenvectors****

* **Eigenvector equation:**
* **Computation:** Solve for eigenvalues () and then find eigenvectors.
* **Significance:** Eigenvectors indicate principal directions of transformation; used in PCA.

## **2. Probability & Statistics**

### ****Probability Distributions****

* **Discrete:** Bernoulli, Binomial, Poisson.
* **Continuous:** Normal (Gaussian), Exponential, Uniform.

### ****Bayes' Theorem****

* Formula:
* **Use case:** Updating beliefs given new data (e.g., Naive Bayes classifier).

### ****Expectation & Variance****

* **Expectation (Mean):** for discrete, for continuous.
* **Variance:** .
* **Standard Deviation:** .

## **3. Calculus**

### ****Gradients & Partial Derivatives****

* **Gradient** of : .
* Used in optimization to find direction of steepest ascent/descent.

### ****Chain Rule****

* If , then:
* Crucial for backpropagation in neural networks.

## **4. Optimization Techniques**

### ****Gradient Descent****

* **Formula:** .
* **Learning Rate ()**: Too high causes divergence, too low slows convergence.
* **Variants:** Batch GD (full dataset), Mini-batch GD.

### ****Stochastic Gradient Descent (SGD)****

* Updates weights using one random sample at a time instead of the full dataset.
* **Advantages:** Faster convergence, avoids local minima.
* **Disadvantages:** More variance, requires tuning learning rate.