DESCRIPTION

were are currently updating our project and you will be regularly updated and for more infomation notes,pdf,code,full png ans svg map you can follow us on following and DM on social media platforms.

`instagram` https://www.instagram.com/ahanger.asif?igsh=MXZiczBreWgwd25iMA==

`facebook` https://www.facebook.com/ahangerasifahanger.asif

📘 Machine Learning Mathematics Notebook

Author: Asif Ahanger
Purpose: Core Mathematics for Machine Learning

This notebook covers essential mathematical concepts required for Machine Learning with **definitions, explanations, and Python examples.**

🔢 1. Equations

An **equation** is a mathematical statement that shows the equality of two expressions.

In ML, equations are used to:

- Define models
- Express loss functions
- Update parameters

In [1]:
```
# Simple linear equation: ax + b = 0
a = 2
b = -4

x = -b / a
x
```

Out[1]:
```
2.0
```

📐 2. Linear Algebra

Linear Algebra is the **backbone of ML.**

It deals with:

- Vectors
- Matrices

- Matrix operations

Used heavily in **neural networks, regression, PCA, embeddings**.

In [2]:
```
import numpy as np

# Vectors
asif = np.array([1, 2, 3])
ahanger = np.array([4, 5, 6])

dot_product = np.dot(asif, ahanger)
dot_product
```

Out[2]:
```
np.int64(32)
```

🔲 Matrix Multiplication

Matrix multiplication is used in **forward propagation** of neural networks.

In [3]:
```
A_mat = np.array([[2, 3], [4, 5]])
B_mat = np.array([[4, 6], [6, 8]])

A_mat @ B_mat
```

Out[3]:
```
array([[26, 36],
       [46, 64]])
```

🔁 Transpose of Matrix

Transpose flips rows into columns.

Used in **gradient calculations and optimization**.

In [4]:
```
Danish = np.array([[1, 2], [3, 4]])
Danish.T
```

Out[4]:
```
array([[1, 3],
       [2, 4]])
```

📉 3. Differential Calculus

Differential calculus deals with **rates of change**.

In ML it is used for:

- Gradient Descent
- Loss minimization
- Backpropagation

In [5]:
```
# Gradient Descent Example

w = 5          # initial weight
lr = 0.1       # learning rate
```

```
# derivative of loss = w^2 is 2w
gradient = 2 * w
w = w - lr * gradient

w
```

Out[5]: 4.0

📊 4. Probability

Probability measures the **likelihood of events.**

Used in ML for:

- Classification
- Uncertainty modeling
- Naive Bayes

In [6]:
```
# Mean, Variance, Standard Deviation

Tawqeer = np.array([10, 20, 30, 40])

mean = np.mean(Tawqeer)
variance = np.var(Tawqeer)
std_dev = np.std(Tawqeer)

mean, variance, std_dev
```

Out[6]: (np.float64(25.0), np.float64(125.0), np.float64(11.180339887498949))

🎲 Random Variables & Normal Distribution

Many ML algorithms assume data follows a **normal distribution.**

In [7]:
```
samples = np.random.normal(loc=0, scale=1, size=100)
samples[:10]
```

Out[7]:
```
array([ 0.21422157,  1.54292053,  1.64483243, -1.13610975, -0.88939877,
        0.42912518,  0.01466276,  2.0462194 ,  0.71457549,  0.17234994])
```

📐 5. Bayes Theorem

Bayes Theorem helps update probability using new evidence.

Formula:

$P(A|B) = (P(B|A) \times P(A)) / P(B)$

Used in **Naive Bayes classifiers**.

In [8]:
```
P_A = 0.3
P_B_given_A = 0.8
P_B = 0.5
```

```
P_A_given_B = (P_B_given_A * P_A) / P_B
P_A_given_B
```

Out[8]: 0.48

📈 6. Correlation

Correlation measures the **relationship between variables**.

Important for **feature selection**.

In [9]:
```
x = np.array([1, 2, 3, 4])
y = np.array([10, 20, 30, 40])

np.corrcoef(x, y)
```

Out[9]:
```
array([[1., 1.],
       [1., 1.]])
```

📉 7. Loss Function (MSE)

Loss functions measure **how wrong a model is**.

Mean Squared Error (MSE) is widely used in regression.

In [10]:
```
y_true = np.array([3, 5, 7])
y_pred = np.array([2, 5, 8])

loss = np.mean((y_true - y_pred) ** 2)
loss
```

Out[10]: np.float64(0.6666666666666666)