✅ 30-Day Machine Learning Roadmap (1 algorithm per day)

Each day includes:
✔ What it is
✔ When to use
✔ Small math intuition
✔ Python code
✔ Mini-task for practice

📌 WEEK 1 — Basic ML Foundations + Simple Models

Day 1: Linear Regression

– Prediction of continuous values
– Fit a line
– Code: `sklearn.linear_model.LinearRegression`

Day 2: Logistic Regression

– Classification (Yes/No)
– Sigmoid function
– Code: `LogisticRegression`

Day 3: K-Nearest Neighbors (KNN)

– Instance-based learning
– Distance calculation
– Code: `KNeighborsClassifier`

Day 4: Naive Bayes

– Probability-based
– Good for text classification
– Code: `GaussianNB`, `MultinomialNB`

Day 5: Decision Tree

– Tree representation
– Gini/Entropy
– Code: `DecisionTreeClassifier`

Day 6: Random Forest

– Multiple trees → strong model
– Code: `RandomForestClassifier`

Day 7: Gradient Boosting (GBM)

– Boosting weak learners
– Code: `GradientBoostingClassifier`

📌 WEEK 2 — Intermediate Algorithms

Day 8: Support Vector Machine (SVM)

– Hyperplane, margin
– Linear, RBF kernel
– Code: `SVC`

Day 9: XGBoost

– State-of-the-art boosting
– High accuracy
– Code: `xgboost.XGBClassifier`

Day 10: LightGBM

– Fast high-performance boosting
– Good for large datasets

Day 11: CatBoost

– Works well with categorical data
– Reduces preprocessing

Day 12: K-Means Clustering

– Unsupervised
– Clusters data
– Code: `KMeans`

Day 13: Hierarchical Clustering

– Dendrogram
– Bottom-up or top-down
– Code: `AgglomerativeClustering`

Day 14: PCA (Dimensionality Reduction)

– Reduce feature space
– Convert to components
– Code: `PCA`

📌 WEEK 3 — Advanced Techniques

Day 15: t-SNE / UMAP

– Visualization of high-dim data
– Use case: cluster visualization

Day 16: DBSCAN

– Density-based clustering
– Detect noise/outliers
– Code: `DBSCAN`

Day 17: Linear Discriminant Analysis (LDA)

– Supervised dimensionality reduction

Day 18: Polynomial Regression

– Non-linear regression
– Code: `PolynomialFeatures`

Day 19: Regularization (L1, L2)

– Ridge, Lasso, ElasticNet
– Prevent overfitting

Day 20: Ensemble Techniques

– Bagging
– Boosting
– Stacking

📌 WEEK 4 — Deep Learning + Real Project

Day 21: Neural Networks (Basics)

– Perceptron
– Activation functions
– Code: `Keras Sequential`

Day 22: MLP Classifier

– Multi-layer neural network
– Code: `MLPClassifier`

Day 23: CNN (Convolutional Neural Network)

– Image classification
– Code: `Conv2D`, `Flatten`, etc.

Day 24: RNN (Recurrent Neural Networks)

– For sequential data
– LSTM, GRU

Day 25: Autoencoders

– Data compression
– Anomaly detection

Day 26: Transfer Learning (VGG, ResNet)

– Pre-trained models
– Faster training

Day 27: NLP (Text Classification)

– Tokenization
– Word2Vec
– TF-IDF

Day 28: Feature Engineering

– Handling missing values
– Scaling (Standard, MinMax)
– Encoding

Day 29: Model Evaluation

– Confusion matrix
– Accuracy, Precision, Recall
– ROC curve

Day 30: Build a Full ML Project (End-to-End)

Choose one:
✔ Predict house prices
✔ Customer churn prediction
✔ Sentiment analysis
✔ Fake news detection

হ্যাঁ — **এই 30-day plan ফলো করলে তুমি Machine Learning-এর 80–90% Core Knowledge পুরোপুরি শিখে ফেলতে পারবে।**

✅ ৩০ দিনে ঠিক কী কী শিখে ফেলতে পারবে?

✔ 1. Machine Learning-এর সব ধরনের Algorithms

Supervised, Unsupervised, Ensemble, Dimensionality Reduction, Deep Learning basics— সবই থাকবে।

✔ 2. Mathematical intuition

– Linear Algebra basics
– Probability basics
– Gradient Descent concept
– Cost function
— এগুলা তুমি Example + Visualization সহ শিখবে।

✔ 3. Python Implementation (sklearn + tensorflow/keras)

প্রতিদিন একটি অ্যালগরিদমের কোড, ব্যাখ্যা, ডেটাসেট, এবং ছোট প্র্যাকটিস টাস্ক — সবই দেয়া হবে।

✔ 4. Data preprocessing & feature engineering

– Missing value
– Outliers
– Scaling
– Encoding
– Feature selection
এগুলা সবই শিখবে।

✔ 5. Model Evaluation

– Accuracy
– Precision
– Recall
– F1-score
– ROC-AUC

– Confusion Matrix
— সব evaluation techniques clear হবে।

✔ 6. Deep Learning Basics

– Neural Networks
– CNN
– RNN
– Transfer Learning
— এগুলার বেসিক খুব solidভাবে বুঝে যাবে।

✔ 7. একটি Real ML Project build করবে

Day-30 এর সময় একটি Full end-to-end ML Project বানাবা:
✔ EDA
✔ Algorithm selection
✔ Training
✔ Hyperparameter tuning
✔ Model saving
✔ Deployment file structure

😍 ৩০ দিনের শেষে তুমি কী করতে পারবে?

⭐ Industry-level ML job-ready basics clear

⭐ ML interviews-এর 60–70% প্রশ্ন confidently answer করতে পারবে

⭐ নিজের Django ML website-এ Models integrate করতে পারবে

(তুমি আগেই বলেছো Django দিয়ে ML apply করতে চাও — এই course setভাবেই তোমার জন্য perfect)

⭐ Kaggle beginner–intermediate সমস্যা solve করতে পারবে

⭐ নিজে নিজে ML/DL advance topic continue করতে পারবে

🔥 তুমি যদি চাও — আমি এখনই শুরু করতে পারি:

Say “Start Day 1” — then I provide full Linear Regression (A to Z)

👉 Concept
👉 Math
👉 Formula

👉 Python code
👉 Visualization
👉 Practice task

✅ DAY 1 – Linear Regression (A to Z Complete Guide)

আজ আমরা Machine Learning-এর সবচেয়ে বেসিক কিন্তু সবচেয়ে Important algorithm শিখবো:
Linear Regression

এটা খুব সহজ, কিন্তু 80% ML project-এ এর concept ব্যবহার হয়—Gradient Descent, Cost Function, Weights—all start from here.

🎯 1. What is Linear Regression? (Simple Explanation)

Linear Regression একটি **prediction algorithm** যা **continuous value** predict করে:

👉 বাড়ির দাম
👉 গাড়ির price
👉 salary prediction
👉 temperature prediction

Formula:

[
y = mx + c
]

যেখানে
m = slope (weight)
c = intercept (bias)
x = input
y = predicted output

🎯 2. When to Use Linear Regression?

যখন output **number** হয় এবং relationship **straight line** এর মতো হয়।

✔ Price prediction
✔ Time prediction
✔ Growth prediction
✔ Simple forecasting

🎯 3. Mathematical Intuition (Easy Version)

✔ Goal:

Best fitting line খুঁজে বের করা যা errors minimize করে।

Error = Difference between actual vs predicted

[
\text{Error} = y*{actual} - y*{pred}
]

Cost Function:

[
J(m,c) = \frac{1}{2n} \sum (y*{pred} - y*{actual})^2
]

সবচেয়ে small error যেই line দেয়, সেটাই Best Fit Line।

🎯 4. Types of Linear Regression

1. **Simple Linear Regression** — one feature
2. **Multiple Linear Regression** — many features

🎯 5. Linear Regression Using Python (Hands-on)

✔ Import library

```python
from sklearn.linear_model import LinearRegression
import numpy as np
import pandas as pd
```

✔ Sample Data

```python
df = pd.DataFrame({
    "study_hours": [1, 2, 3, 4, 5],
    "marks": [40, 50, 60, 70, 80]
})
```

✔ Split X, y

```
X = df[['study_hours']]  # input
y = df['marks']          # output
```

✔ Train Model

```
model = LinearRegression()
model.fit(X, y)
```

✔ Predict

```
print(model.predict([[6]]))   # 6 hour study -> ?
```

🎯 6. Visualization (Optional but useful)

```
import matplotlib.pyplot as plt

plt.scatter(df.study_hours, df.marks)
plt.plot(df.study_hours, model.predict(X))
plt.xlabel("Study Hours")
plt.ylabel("Marks")
plt.show()
```

🎯 7. Advantages

✔ Very easy
✔ Fast
✔ Interpretable
✔ Good baseline model

🎯 8. Disadvantages

❌ Only works for linear relation
❌ Sensitive to outliers
❌ Not good for complex data

📝 DAY 1 Practice Task (You Must Do This)

একটা dataset বানাও:

experience	salary
1	20k
2	30k
3	45k
4	55k
5	65k

✔ Task:

1. LinearRegression use করে model train করবে
2. Predict করতে হবে:
   "6 years experience → salary কত হবে?"
3. Graph draw করতে হবে

যদি চাও আমি তোমার code check করবো।

🔥 Ready for Day 2 (Logistic Regression)?

বল: **Start Day 2**

✅ DAY 2 – Logistic Regression (A to Z Complete Guide)

Linear Regression দিয়ে **number** predict করেছিলে।
আজ Logistic Regression দিয়ে শিখবে **Yes/No, 0/1** টাইপ classification।

এটা Machine Learning-এর সবচেয়ে জনপ্রিয় classifier.

🎯 1. What is Logistic Regression?

Logistic Regression একটি **classification algorithm**।

👉 Email spam / not-spam
👉 Disease: yes or no
👉 Admission: accept or reject

👉 Credit card fraud detection
👉 Pass / Fail prediction

এটা regression না — **classification**!

🎯 2. Why is it called “Logistic Regression”?

কারণ এটি Linear Regression এর output কে **0–1 range** এ convert করে
Sigmoid function ব্যবহার করে।

🎯 3. Sigmoid Function (Main Concept)

[
\sigma(z) = \frac{1}{1 + e^{-z}}
]

Prediction rule:

```
if probability ≥ 0.5 → class = 1
else → class = 0
```

🎯 4. Logistic Regression Formula

[
z = w_1x_1 + w_2x_2 + b
]

[
y = \sigma(z)
]

🎯 5. Where to Use Logistic Regression?

✔ Binary classification
✔ Medical decision
✔ Fraud detection
✔ Customer churn

✔ Loan approval
✔ Sentiment analysis (positive/negative)

🎯 6. Python Implementation (Step-by-step)

✔ Import library

```
from sklearn.linear_model import LogisticRegression
import pandas as pd
```

✔ Example Dataset

(Exam pass or fail based on study hours)

```
df = pd.DataFrame({
    "study_hours": [1, 2, 3, 4, 5, 6],
    "pass_fail": [0, 0, 0, 1, 1, 1]
})
```

✔ Split X, y

```
X = df[['study_hours']]
y = df['pass_fail']
```

✔ Train Model

```
model = LogisticRegression()
model.fit(X, y)
```

✔ Predict

```
print(model.predict([[2]]))  # 2 hours -> 0 (fail)
print(model.predict([[5]]))  # 5 hours -> 1 (pass)
```

✔ Predict Probability

```
print(model.predict_proba([[4]]))
```

Output like:

```
[[0.30 0.70]]
```

Meaning → 70% chance to pass.

🎯 7. Visualization (Sigmoid curve)

```
import numpy as np
import matplotlib.pyplot as plt

X_new = np.linspace(0, 6, 100).reshape(-1, 1)
y_prob = model.predict_proba(X_new)[:, 1]

plt.plot(X_new, y_prob)
plt.xlabel("Study Hours")
plt.ylabel("Probability to Pass")
plt.show()
```

🎯 8. Advantages of Logistic Regression

✔ Simple & fast
✔ Works well for small datasets
✔ Gives probability
✔ Easy to interpret

🎯 9. Disadvantages

❌ Works only for linear boundary
❌ Not good for large features
❌ Poor performance on complex data

📝 DAY 2 Practice Task (You Must Do This)

Dataset:

age	diabetes
25	0

age	diabetes
30	0
35	0
40	1
45	1
50	1

Task:

1. Logistic Regression train করো
2. Predict করো:
   👉 Age = 37 → Diabetes probability কত?
3. Sigmoid curve graph draw করো

চাইলে তোমার কোড এখানে পাঠাও—আমি ভুল থাকলে ঠিক করে দেবো।

🔥 Ready for Day 3 (KNN – K Nearest Neighbors)?

বল: **Start Day 3**

✅ DAY 3 – K-Nearest Neighbors (KNN) (A to Z Complete Guide)

আজ তুমি শিখবে **KNN**, যেটা সবচেয়ে সহজ কিন্তু খুব শক্তিশালী ML algorithm—classification + regression দুই কাজেই ব্যবহার করা যায়।

🎯 1. What is KNN? (Simplest Explanation)

KNN হলো:

👉 **"Find the nearest neighbors and take a decision based on them."**

যদি তুমি কোনো নতুন ডেটাকে classify করতে চাও →
তাহলে Algorithm দেখে:
তার সবচেয়ে কাছের K সংখ্যক data point কোন class এর?

Example:
নতুন student-এর result predict করতে গেলে
→ কাকে বেশি resemble করে? তাদের classify করে।

🎯 2. KNN কীভাবে কাজ করে? (Step by Step)

1️⃣ নতুন data point আসলো
2️⃣ Dataset এর সব point এর সাথে distance measure করা হলো
3️⃣ সব distances ছোট → বড় sort করা হলো
4️⃣ সবচেয়ে কাছের **K number** neighbors choose করা হলো
5️⃣ Majority vote = prediction

✔ If classification → majority class
✔ If regression → average value

🎯 3. Distance Calculation

Most used: **Euclidean Distance**

[
d = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}
]

🎯 4. When to Use KNN?

✔ Small dataset
✔ Clear boundaries
✔ Image recognition
✔ Recommendation system
✔ Anomaly detection

🎯 5. K Value (Super Important)

👉 ছোট K → model noise-sensitive
👉 বড় K → model over-generalize

Usually:
Best K = sqrt(total samples) (rule of thumb)

🎯 6. KNN Using Python (sklearn)

✔ Import library

```
from sklearn.neighbors import KNeighborsClassifier
import pandas as pd
```

✔ Dataset (simple)

```
df = pd.DataFrame({
    "height":  [150, 160, 170, 180, 190],
    "weight":  [50, 60, 65, 80, 90],
    "gender":  ["F", "M", "M", "M", "M"]   # M or F
})
```

✔ Split X, y

```
X = df[['height', 'weight']]
y = df['gender']
```

✔ Train KNN model

```
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X, y)
```

✔ Predict

```
print(model.predict([[165, 60]]))  # height=165, weight=60
```

🎯 7. Visualization (Optional)

Scatter plot ব্যবহার করে decision boundary দেখা যায়।

```
import matplotlib.pyplot as plt

plt.scatter(df.height, df.weight, c=(y=="M"))
plt.xlabel("Height")
plt.ylabel("Weight")
plt.show()
```

🎯 8. Advantages of KNN

✔ Very simple
✔ No training required
✔ Works for classification + regression
✔ Good accuracy for small datasets

🎯 9. Disadvantages

❌ Slow for large datasets
❌ Needs scaling (StandardScaler)
❌ Sensitive to noise
❌ Memory expensive (stores entire dataset)

📝 DAY 3 Practice Task

Dataset:

age	salary	buy_car
22	20k	0
25	30k	0
26	35k	1
30	45k	1
35	60k	1

Your Tasks:

1️⃣ Use KNN (k = 3)
2️⃣ Train model
3️⃣ Predict:
✔ age = 28, salary = 40k → buy_car = ?
4️⃣ Try different k values: 1, 3, 5
দেখো prediction change হয় কি না।

👉 চাইলে কোড পাঠাও — আমি review করে ঠিক করে দেবো।

🔥 Ready For Day 4 – Naive Bayes?

বল: **Start Day 4**

✅ DAY 4 – Naive Bayes (A to Z Complete Guide)

আজ আমরা শিখবো **Naive Bayes**, যেটা সবচেয়ে দ্রুত, সহজ এবং NLP (text classification) এ সবচেয়ে বেশি ব্যবহার হয়।

Spam detection → Naive Bayes
Sentiment analysis → Naive Bayes
Document classification → Naive Bayes
Email filtering → Naive Bayes

🎯 1. What is Naive Bayes? (Simple Explanation)

Naive Bayes হলো একটি **probability-based classification algorithm**।

এটা Bayes' Theorem ব্যবহার করে বলে:
👉 "কোনো class হওয়ার probability কত?"

Formula:

[
P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}
]

"Naive" কেন?

কারণ algorithm ধরে নেয় **features একে অপরের থেকে independent**
যা আসলে সত্য না, কিন্তু তারপরও model খুব ভালো কাজ করে।

🎯 2. Where is Naive Bayes Used?

✔ Email spam filter
✔ Sentiment analysis (positive/negative)
✔ Document classification
✔ News categorization
✔ Medical diagnosis

🎯 3. Types of Naive Bayes

1. **Gaussian Naive Bayes** → Continuous data
2. **Multinomial Naive Bayes** → Text data
3. **Bernoulli Naive Bayes** → Binary data

🎯 4. How Naive Bayes Works? (Simple)

Example: "This is good" → positive review?

Algorithm calculates:

✔ Probability(sentence | positive class)
✔ Probability(sentence | negative class)

Then chooses highest probability class.

🎯 5. Naive Bayes Using Python

✔ Import

```
from sklearn.naive_bayes import GaussianNB
import pandas as pd
```

✔ Simple Dataset

```
df = pd.DataFrame({
    "age": [25, 30, 35, 40, 45],
    "diabetes": [0, 0, 1, 1, 1]
})
```

✔ Split

```
X = df[['age']]
y = df['diabetes']
```

✔ Train model

```
model = GaussianNB()
model.fit(X, y)
```

✔ Predict

```
print(model.predict([[33]]))
```

✔ Probability

```
print(model.predict_proba([[33]]))
```

🎯 6. Naive Bayes for Text Classification (Most Important)

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

texts = ["I love this movie", "I hate this movie"]
labels = [1, 0]

cv = CountVectorizer()
X = cv.fit_transform(texts)

model = MultinomialNB()
model.fit(X, labels)

print(model.predict(cv.transform(["I love it"])))
```

🎯 7. Advantages

✔ Very fast
✔ Works great for text
✔ Works well with small data
✔ Simple and accurate
✔ Less training time

🎯 8. Disadvantages

❌ Assumes independence (not true)
❌ Not good for complex datasets
❌ Zero-frequency problem (some words never appear)

📝 DAY 4 Practice Task

Dataset (Weather → Play cricket)

weather	temperature	play
sunny	hot	no
sunny	mild	no
overcast	hot	yes
rainy	mild	yes
rainy	cool	yes

Your Tasks:

1️⃣ Convert categorical → label encode
2️⃣ Train **GaussianNB**
3️⃣ Predict:
✔ weather = sunny, temperature = cool → play?
4️⃣ Predict probability

👉 চাইলে তোমার কোড পাঠালে আমি review করে দেবো।

🔥 Ready for Day 5 – Support Vector Machine (SVM)?

বল: **Start Day 5**

✅ DAY 5 – Support Vector Machine (SVM) (A to Z Complete Guide)

আজ তুমি শিখবে Machine Learning-এর সবচেয়ে powerful এবং interview-friendly algorithm:

⭐ Support Vector Machine (SVM)

Classification + regression দুইটাই পারে, কিন্তু classification-এ অসাধারণ।

🎯 1. What is SVM? (Simple Explanation)

SVM এমন একটি algorithm যা দুই class এর মাঝে **best possible boundary (hyperplane)** খুঁজে বের করে।

👉 Boundary টি এমন হয় যা দুই class থেকে **maximum distance** এ থাকে।
👉 এই distance কে বলে **Margin**।

Goal of SVM:
➡️ Best hyperplane খুঁজে বের করা যা দুই class কে most separated করে।

🎯 2. Important Terms

✔ Hyperplane:

Decision boundary যা দুই class আলাদা করে।

✔ Support Vectors:

Boundary-এর সবচেয়ে কাছের যেসব data point →
এই পয়েন্টগুলিই boundary তৈরি করতে সাহায্য করে।

✔ Margin:

Two class-এর boundary থেকে distance।

SVM tries to **maximize margin** → তাই এটি noise-resistant।

🎯 3. Where is SVM Used?

✔ Image classification
✔ Medical diagnosis
✔ Face detection
✔ Spam filtering
✔ Fraud detection
✔ Linearly separable AND non-linear data দুইটাতেই কাজ করে

🎯 4. Linear vs Non-Linear SVM

✔ Linear SVM:

Straight line boundary

✔ Non-linear SVM:

Curved boundary
এটা সম্ভব হয় **Kernel Trick** এর মাধ্যমে।

🎯 5. Kernel Trick (Very Important)

Kernel SVM কে non-linear boundary draw করতে সাহায্য করে।

Popular Kernels:

Kernel	Use Case
linear	simple data
rbf	complex boundary
polynomial	curved patterns
sigmoid	neural network style

🎯 6. SVM Using Python (Practical)

✔ Import library

```python
from sklearn.svm import SVC
import pandas as pd
```

✔ Example dataset

```python
df = pd.DataFrame({
    "age": [22, 25, 30, 35, 40, 45],
    "buy": [0, 0, 1, 1, 1, 1]
})
```

✔ Split X and y

```python
X = df[['age']]
y = df['buy']
```

✔ Train SVM model

```python
model = SVC(kernel='linear', probability=True)
model.fit(X, y)
```

✔ Predict

```python
print(model.predict([[28]]))
```

✔ Predict probability

```python
print(model.predict_proba([[28]]))
```

🎯 7. SVM with Non-linear Kernel (RBF)

```python
rbf_model = SVC(kernel='rbf', probability=True)
rbf_model.fit(X, y)
print(rbf_model.predict([[28]]))
```

🎯 8. Visualization (Optional)

```python
import numpy as np
import matplotlib.pyplot as plt

X_new = np.linspace(20, 50, 100).reshape(-1, 1)
y_plot = model.predict(X_new)

plt.scatter(df.age, df.buy)
plt.plot(X_new, y_plot)
plt.xlabel("Age")
plt.ylabel("Buy or Not")
plt.show()
```

🎯 9. Advantages of SVM

✔ Works great for complex boundary
✔ High accuracy
✔ Good for small datasets
✔ Memory efficient
✔ Outlier resistant

🎯 10. Disadvantages

❌ Slow for large datasets
❌ Parameter tuning required (C, gamma)
❌ No direct probability output (need probability=True)

📝 DAY 5 Practice Task (You Must Do This)

Dataset:

salary	experience	promoted
30k	1	0
40k	3	0
50k	5	1
60k	7	1
70k	10	1

Your Task:

1️⃣ Train SVM with:
✔ kernel = linear
✔ kernel = rbf

2️⃣ Predict promotion for:
salary = 55k, experience = 4

3️⃣ Compare two kernels
→ কোনটি ভালো predict করলো?

👉 তোমার কোড আমাকে পাঠালে আমি ঠিক করে দেবো।

🔥 Ready for Day 6 – Decision Tree?

বল: **Start Day 6**

✅ DAY 6 – Decision Tree (A to Z Complete Guide)

আজ তুমি শিখবে Machine Learning এর সবচেয়ে intuitive ও powerful algorithm:

🌳 Decision Tree

মানুষ যেভাবে decision নেয় →
"If condition → then decision"
Decision Tree ঠিক সেভাবেই কাজ করে।

🎯 1. What is Decision Tree? (Simple Explanation)

Decision Tree হলো tree-এর মতো একটি structure যেখানে:

👉 Root node → প্রথম decision
👉 Branch → condition
👉 Leaf node → final output / class

Example:
Loan approve?

```
IF income > 40k:
       IF credit_score > 700:
              Approve
       ELSE:
              Reject
ELSE:
       Reject
```

Manush jemon reasoning kore → Model o temon.

🎯 2. When to Use Decision Tree?

✔ Classification
✔ Regression
✔ Loan approval
✔ Medical diagnosis
✔ Customer prediction
✔ Rule-based ML system

🎯 3. Why Decision Tree is Important?

✔ Easy to understand
✔ No scaling required
✔ Works with categorical + numerical data
✔ Non-linear relationships handle করতে পারে

🎯 4. Key Terms

✔ Gini Impurity

Used in classification
Measures impurity (lower is better)

[
G = 1 - \sum p^2
]

✔ Entropy

Used in ID3 algorithm
Measures uncertainty

[
E = - \sum p \log_2 p
]

✔ Information Gain

Entropy কমানোর মাধ্যমে best split determine করা হয়।

🎯 5. Decision Tree Using Python

✔ Import

```python
from sklearn.tree import DecisionTreeClassifier
import pandas as pd
```

✔ Example Dataset

```python
df = pd.DataFrame({
    "age": [22, 25, 28, 35, 40, 50],
    "buy": [0, 0, 1, 1, 1, 1]
})
```

✔ Split

```python
X = df[['age']]
y = df['buy']
```

✔ Train model

```python
model = DecisionTreeClassifier()
model.fit(X, y)
```

✔ Predict

```
print(model.predict([[30]]))
```

🎯 6. Visualize the Tree (Most Important)

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))
plot_tree(model, filled=True)
plt.show()
```

🎯 7. Advantages

✔ Highly interpretable
✔ Works with categorical data
✔ Handles non-linear data
✔ No need for feature scaling
✔ Fast training

🎯 8. Disadvantages

❌ Overfitting tendency
❌ Small changes → big structure change
❌ Less accurate alone (best when combined with ensemble)

🎯 9. Hyperparameters to Know

Parameter	Meaning
max_depth	tree এর depth limit
min_samples_split	split হওয়ার জন্য minimum sample
min_samples_leaf	leaf এ minimum sample
criterion	gini বা entropy

Example:

```
model = DecisionTreeClassifier(max_depth=3, criterion="entropy")
```

📝 DAY 6 Practice Task

Dataset:

age	income	loan_approved
25	30k	0
28	40k	0
32	50k	1
40	60k	1
45	80k	1

Your Tasks:

1️⃣ Train a Decision Tree classifier
2️⃣ Predict:
age = 33, income = 52k → loan_approved?
3️⃣ Visualize the tree
4️⃣ Try max_depth = 2 vs max_depth = 3

👉 তোমার কোড পাঠালে আমি review করবো।

🔥 Ready for Day 7 – Random Forest (The powerful ensemble)?

বল: **Start Day 7**