১) **Categorical Data Encoding** কী এবং কেন দরকার?

মেশিন লার্নিং মডেল মূলত ম্যাথ করে। ম্যাথ করার জন্য ইনপুট/আউটপুট শেষ পর্যন্ত সংখ্যায় থাকতে হয়। কিন্তু বাস্তবে ডেটাতে থাকে:

- শহর: ঢাকা/চট্টগ্রাম
- রিভিউ: Poor/Average/Good
- Purchased: Yes/No

এগুলোকে সরাসরি মডেলে দিলে মডেল "অর্থ" বুঝতে পারে না। তাই আমরা ক্যাটেগরি **(**টেক্সট/লেবেল**)** কে সংখ্যায় রূপান্তর করি—এটাই Encoding।

কিন্তু গুরুত্বপূর্ণ সতর্কতা:

- যেকোন সংখ্যা দিলেই হবে না
- ভুলভাবে সংখ্যা দিলে মডেল ভুল সম্পর্ক ধরে ফেলতে পারে (যেমন ঢাকা=2, চট্টগ্রাম=1 দিলে মডেল ভাবতে পারে ঢাকা "বড়" বা "উচ্চ")

২) প্রথমে বুঝতে হবে: **Nominal** না **Ordinal?**

Encoding করার আগে সবচেয়ে জরুরি প্রশ্ন:
এই ক্যাটেগরিগুলোর মধ্যে কোনো স্বাভাবিক ক্রম আছে কি নেই?

(A) **Nominal (Order** নেই**)**

এখানে কোনো "বড়-ছোট" বা "উচ্চ-নিম্ন" নেই। উদাহরণ:

- City: ঢাকা, চট্টগ্রাম, খুলনা
- Gender: Male, Female
- Color: Red, Green, Blue

এগুলোর মধ্যে ranking নেই।
সুতরাং এমন encoding লাগবে, যেটা কৃত্রিম অর্ডার তৈরি করবে না।

(B) **Ordinal (Order** আছে**)**

এখানে স্বাভাবিক ক্রম/লেভেল আছে। উদাহরণ:

- Review: Poor < Average < Good
- Education: High School < UG < PG
- Size: Small < Medium < Large

এখানে “Good” যে “Poor” থেকে বেশি—এটা সত্য।
তাই encoding এমন হতে হবে, যাতে এই ক্রম মডেল বুঝতে পারে।

৩) নোটে বলা তিনটি **encoding:** কে কোথায় ব্যবহার হবে?

আপনার নোটে তিনটি টেকনিক এসেছে:

1. Ordinal Encoding
2. One-Hot Encoding (পরের নোটে)
3. Label Encoding (বিশেষ করে target/y)

এখন এগুলোকে পরিষ্কারভাবে আলাদা করি।

৪) **Ordinal Encoding (**ইনপুট **X-**এর **Ordinal** কলামের জন্য)

কখন ব্যবহার করবেন?

যখন feature (X)-এর মধ্যে ক্রমযুক্ত ক্যাটেগরি থাকে।

উদাহরণ:

- Review: Poor, Average, Good
- Education: High School, UG, PG

কীভাবে কাজ করে?

আপনি নিজে অর্ডার নির্ধারণ করে দেবেন:

- Poor = 0, Average = 1, Good = 2
- High School = 0, UG = 1, PG = 2

এতে মডেল বুঝতে পারে:

- 2 > 1 > 0
  অর্থাৎ “Good” সত্যিই “Poor” থেকে বেশি লেভেল।

কেন **categories parameter** গুরুত্বপূর্ণ?

যদি আপনি অর্ডার না বলে দেন, অনেক টুল/লাইব্রেরি ডিফল্টভাবে

- অ্যালফাবেটিকাল অর্ডার
- বা অভ্যন্তরীণ কোনো অর্ডার
  ব্যবহার করতে পারে।

তখন উদাহরণস্বরূপ:

- “Average” কে 2
- “Good” কে 0
  দেওয়া হয়ে গেলে মডেল ভুল শেখে।

সুতরাং ordinal encoding-এর মূল দায়িত্ব আপনার:

- বাস্তব অর্ডারটি সঠিকভাবে নির্ধারণ করা

৫) Label Encoding (Target y-এর জন্য)

সঠিক ব্যবহার কোথায়?

Label Encoding মূলত **target / output (y)** কে numeric করতে ব্যবহৃত হয়।

উদাহরণ:

- Purchased: Yes/No
  এখানে মডেলকে শিখতে হবে “কিনবে কি কিনবে না”—এটা আউটপুট।

Label Encoder সাধারণত:

- No → 0
- Yes → 1
  (বা উল্টোও হতে পারে; তবে mapping একবার fit হয়ে গেলে consistent থাকে)

কেন **input feature (X)-এ Label Encoding** ভুল বলা হয়?

কারণ আপনি যদি nominal feature-এ label encoding ব্যবহার করেন, যেমন:

- City: Dhaka=0, Chittagong=1, Khulna=2
  তাহলে মডেল ভাবতে পারে:
  Khulna (2) > Chittagong (1) > Dhaka (0)

কিন্তু শহরের ক্ষেত্রে “বড়-ছোট” কোনো অর্থই নেই।
এটা মডেলকে মিথ্যা ordinal relation শেখায়।

বিশেষ করে linear/logistic regression, KNN, SVM—এখানে ভুলটা বেশি ক্ষতি করে।

ব্যতিক্রম হিসেবে কিছু tree-based model (Decision Tree, Random Forest) অনেক সময় nominal-এ label encoding দিয়েও “প্র্যাক্টিক্যালি” কাজ করতে পারে, কারণ তারা threshold split দিয়ে অর্ডারের অর্থ ঠিকভাবে ব্যবহার না-ও করতে পারে। কিন্তু নিয়মগতভাবে safe practice হলো:

- Nominal X → One-Hot
- Ordinal X → Ordinal Encoding
- Target y → Label Encoding (classification হলে)

৬) One-Hot Encoding (Nominal X-এর জন্য)

এটা পরের, কিন্তু concept টা এখানে পরিষ্কার করে রাখলে ভুল হবে না:

Nominal feature-এর জন্য one-hot encoding এমনভাবে কাজ করে:

- প্রতিটা ক্যাটেগরির জন্য আলাদা binary কলাম (0/1)
- কোনো অর্ডার তৈরি করে না

উদাহরণ:
City = Dhaka/Chittagong/Khulna হলে:

- City_Dhaka
- City_Chittagong
- City_Khulna

এখানে তিনটা কলামের একটায় 1 হবে, বাকিগুলো 0।
মডেল বুঝবে “কোনটা” — কিন্তু “বড়-ছোট” বানাবে না।

৭) Fit-Transform নিয়ম (Train-Test leakage এড়াতে)

Encoding-এর ক্ষেত্রেও scaling-এর মতো একই নীতি:

1. আগে Train-Test split
2. তারপর encoder-কে শুধু train data তে fit
3. তারপর train এবং test—দুটোতেই transform

কারণ:

- encoder train থেকে category mapping শিখবে
- test set থেকে আগে শিখে ফেললে leakage হয়

৮) Practical Pitfalls (খুব কমন ভুল)

(A) Ordinal encoding-এ ভুল order

আপনি যদি order উল্টো দেন, মডেল পুরো সম্পর্ক উল্টো শিখবে।

(B) Test set এ নতুন category (Unseen category)

Real-world data তে test/production এ নতুন category আসতে পারে।
তখন encoder error দিতে পারে, বা ভুল mapping হতে পারে।
এ জন্য pipeline design এ “unknown handle” কৌশল লাগে (যেমন one-hot এ handle_unknown)।

(C) Target encoding নিয়ে বিভ্রান্তি

Label encoding target-এর জন্য, কিন্তু multi-class classification এও হয়:

- Class A=0, Class B=1, Class C=2
  এখানেও “2 বড়” এমন অর্থ নেই, কিন্তু target label হিসেবে এটা সমস্যা নয়, কারণ model internally class id হিসেবেই ধরে (বিশেষ করে sklearn classifiers)।

৯) সংক্ষিপ্ত সিদ্ধান্ত

1. Feature X যদি Ordinal হয় → Ordinal Encoding (order আপনাকে দিতে হবে)
2. Feature X যদি Nominal হয় → One-Hot Encoding
3. Target y যদি classification label হয় → Label Encoding (বা model-specific label mapping)