**Q1. What is data encoding? How is it useful in data science?** Data encoding is the process of converting categorical data (like text labels or non-numeric values) into numerical format so that machine learning algorithms can process it effectively. Since most algorithms work only with numbers, encoding helps to translate categorical variables into a form suitable for analysis. This is crucial because categorical variables often contain important information that needs to be represented numerically for predictive models.

**Q2. What is nominal encoding? Provide an example of how you would use it in a real-world scenario.** Nominal encoding refers to the transformation of categorical variables into a numerical format without any inherent ordering or ranking between categories. Each unique category is assigned a distinct number. This encoding technique is used for nominal (categorical) variables that do not have an ordinal relationship.

**Example:** Suppose you have a dataset containing the “Country” column with three values: “USA,” “Canada,” and “Mexico.” Using nominal encoding, we can assign a unique number to each category:

* USA → 0
* Canada → 1
* Mexico → 2

This transformation allows machine learning models to process the data.

**Q3. In what situations is nominal encoding preferred over one-hot encoding? Provide a practical example.** Nominal encoding is preferred when the categorical variable has a large number of unique values, and one-hot encoding would create an unmanageably large number of columns (leading to high memory usage and sparse data). It is also useful when the machine learning algorithm can naturally handle the encoded numbers without misinterpreting them.

**Example:** Consider a dataset where the “City” column has 1000 unique cities. One-hot encoding would create 1000 new columns, which could be inefficient. In this case, nominal encoding, which assigns a unique integer to each city, would be more efficient.

**Q4. Suppose you have a dataset containing categorical data with 5 unique values. Which encoding technique would you use to transform this data into a format suitable for machine learning algorithms? Explain why you made this choice.** For a dataset with 5 unique categorical values, the choice of encoding depends on the problem and the model:

* If the categories have no ordinal relationship and the machine learning model can interpret them correctly, **nominal encoding** is appropriate.
* If the categories are independent and the model may benefit from distinguishing between them more clearly, **one-hot encoding** would be a better choice.

**Explanation:** If you use nominal encoding, it will map each category to a unique integer, whereas one-hot encoding will create 5 new columns (one for each category). If the categories have no ordinal meaning and there is no risk of the model misinterpreting the numeric labels as ordered, nominal encoding is more efficient. However, if the model could benefit from each category being treated separately, one-hot encoding should be used.

**Q5. In a machine learning project, you have a dataset with 1000 rows and 5 columns. Two of the columns are categorical, and the remaining three columns are numerical. If you were to use nominal encoding to transform the categorical data, how many new columns would be created? Show your calculations.** Nominal encoding assigns a unique integer to each category in a column. It does not create multiple columns like one-hot encoding does. Therefore, for each categorical column, only one new column will be created.

**Calculation:**

* If there are 2 categorical columns, nominal encoding will create 1 new column for each categorical feature.
* **Total new columns** = 2 (one for each categorical column)

Thus, **2 new columns** would be created through nominal encoding.

**Q6. You are working with a dataset containing information about different types of animals, including their species, habitat, and diet. Which encoding technique would you use to transform the categorical data into a format suitable for machine learning algorithms? Justify your answer.** For a dataset containing categorical features such as species, habitat, and diet, **one-hot encoding** is generally preferred. This is because these variables are likely to be non-ordinal, meaning that the categories do not have a meaningful rank or order. One-hot encoding allows each category to be treated independently, ensuring that the machine learning algorithm doesn’t assume any unintended relationships between the categories.

**Justification:**

* **Species**: Different species of animals would have no inherent ranking.
* **Habitat**: Different habitats also have no natural order.
* **Diet**: Diet types (e.g., herbivore, carnivore, omnivore) also don’t have a specific order.

One-hot encoding would prevent the algorithm from misinterpreting the categories as ordinal values while creating separate binary features for each category, which is ideal for these types of data.

**Q7. You are working on a project that involves predicting customer churn for a telecommunications company. You have a dataset with 5 features, including the customer's gender, age, contract type, monthly charges, and tenure. Which encoding technique(s) would you use to transform the categorical data into numerical data? Provide a step-by-step explanation of how you would implement the encoding.** To transform the categorical data into numerical data for this project:

1. **Gender**: This is a binary categorical feature with two categories (e.g., “Male” and “Female”). You can use **label encoding** (nominal encoding) to assign a numerical value (0 or 1) to each gender.
   * Male → 0
   * Female → 1
2. **Contract Type**: This is a categorical variable with multiple values, such as “Month-to-month,” “One-year,” and “Two-year.” For this, you can use **one-hot encoding** to create separate binary columns for each contract type.
   * Contract Type: "Month-to-month", "One-year", "Two-year"
   * One-hot encoding would generate 3 columns (one for each contract type).

**Steps to implement the encoding**:

1. For the “Gender” column, apply **Label Encoding** using a library like sklearn.preprocessing.LabelEncoder.
   * Gender: Male (0), Female (1).
2. For the “Contract Type” column, apply **One-Hot Encoding** using pandas.get\_dummies() or sklearn.preprocessing.OneHotEncoder.
   * Create new columns like “Contract\_Type\_Month\_to\_month,” “Contract\_Type\_One\_year,” and “Contract\_Type\_Two\_year.”

**Resulting data:**

* **Gender**: 0 or 1.
* **Contract Type**: 3 new columns (one for each contract type).

This approach converts categorical features into numerical values while preserving the relationships for machine learning models.