EMMA ding

emmading.com

info@datainterviewpro.com

Encoding Categorical Data

Lesson Structure

📌 Interview Questions

- How to deal with categorical variables?
- What's the difference when treating a variable as a dummy variable vs none dummy?
- How to deal with categorical features when the number of levels is very large, i.e. high cardinality?

Categorical Data

Categorical data indicates types of data which may be divided into groups and the groups may or may not have a specific order. e.g. gender, ethnicity, age group, or a choice of preference.

To make sure the learning algorithm interprets categorical features correctly, we need to **convert categorical string values into integers**.

📌 Very few algorithms, e.g. decision trees, LightGBM, CatGBM, can take categorical features as it is.

▼ **Ordinal and nominal features**

Ordinal features: categorical values that can be ordered or sorted. e.g. t-shirt size: L > M > S.

Nominal features: don't imply any order. e.g. color: green, red, blue.

▼ **Class labels**

Dependent variable of a classification problem.

Class labels are not ordinal.

It's a good practice to provide class labels as integer arrays to avoid technical glitches.

▼ Ordinal features & Class labels

- Define a mapping to map strings to numbers. Typically, we start with integer 1.

User	Rating
1	Poor
2	Fair
3	Good
4	Fair

User	Rating
1	1
2	2
3	3
4	2

- **Pros**: doesn't increase dimensionality of data.
- **Cons**:
  - Imposes an artificial order. Only works for ordinal features or class labels.

Nominal features

▼ One-hot encoding

Transform a categorical feature into several binary features with each level in a category turns into a new feature. e.g. if we have 3 categories → create 3 new features.

For the new feature vector, only the one that gets chosen will have a value of 1 and the rest will be set to 0.

User	Preference
1	Red
2	Green
3	Blue
4	Red

User	Red	Green	Blue
1	1	0	0
2	0	1	0
3	0	0	1
4	1	0	0

Number of new features = cardinality of the original feature

- **Pros**: Easy to examine how each level of a category contributes to predictions.
- **Cons**:
  - Increase the dimensionality of feature vectors.
  - Introduces **multicollinearity** (one feature is perfectly correlated with one or more features), which can be an issue for certain algorithms, e.g. algorithms that require matrix inversion. if features are highly correlated, matrices are computationally difficult to invert, which can lead to numerically unstable estimates.

> ❗ One-hot encoding introduces multicollinearity which may cause interference and parameter estimates being inaccurate.

▼ Dummy encoding

Remove one feature column from the one-hot encoded array. We do not lose any important information by removing a feature column.

User	Preference
1	Red
2	Green
3	Blue
4	Red

User	Red	Green
1	1	0
2	0	1
3	0	0
4	1	0

Number of new features = cardinality of the original feature - 1

- **Pros:** Avoids collinearity of the features.
- **Cons**: Increase the dimensionality of feature vectors.

▼ Problems of one-hot and dummy encoding

> ✏️ Both one-hot encoding and dummy encoding require us to know the vocabulary of the categorical feature beforehand.

What if the vocabulary from the training data is **incomplete**?

What if there are **new categories** get added to the data (i.e. cold-start problems)? The model will not be able to make predictions for such new data.

What if some categorical features have **high cardinality**, e.g. thousands to millions? The model will take up a large amount of storage space and grow in size as the training set grows.

▼ Feature hashing (Hashing trick)

Widely used to encode large-scale categorical features in practice, especially in continual learning settings where the model learns from incoming data in production.

> Feature hashing - Wikipedia
>
> In machine learning, feature hashing, also known as the hashing trick (by analogy to the kernel trick), is a fast and space-efficient way of vectorizing features, i.e. turning arbitrary features into indices in a vector or matrix.
>
> 𝕎 https://en.wikipedia.org/wiki/Feature_hashing#Feature_hashing_(Weinberger_et_al._2009)

Use a **deterministic** (no random seeds) and **portable** (the same algorithm can be using for both training and serving) **hash function** to generate a hashed value of each category. The hashed value will become the index of that category.

	departure_airport	hash3	hash10	hash1000
0	SJU	2	1	81
1	PVD	1	7	667
2	MSY	1	1	991
3	MHT	2	7	507
4	CHS	1	2	432
5	CPR	2	6	306
6	PHX	0	3	743
7	SFO	0	4	714
8	SNA	2	7	587
9	SAN	0	7	447

Source: Lakshmanan, V., Robinson, S., & Munn, M. (2021). *Machine Learning Design Patterns: Solutions to common challenges in data preparation, model building, and MLOps*. O'Reilly.

Pros:

- Can choose the number of encoded values for a feature in advance, without having to know how many categories there will be.
- Does not increase dimensionality of the data - practical for categorical features have **high cardinality.**

Potential problems:

- **Collision** - two distinct categories being assigned the same hash code. Model accuracy will be compromised.

📍 Don’t use the hashing trick if you know the vocabulary beforehand, if the vocabulary size is relatively small, and if cold start is not a concern.