



Categorical Data Encoding Techniques



Krishnakanth Naik Jarapala · Follow

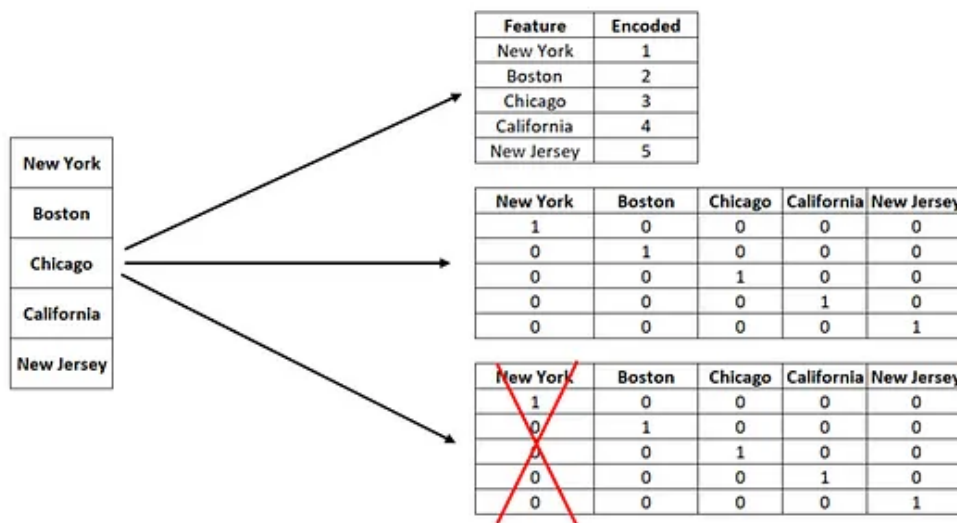
Published in AI Skunks · 7 min read · Mar 13, 2023



47



2



Different Encoding Techniques

Introduction:

Data Encoding is an important pre-processing step in Machine Learning. It refers to the process of converting categorical or textual data into numerical format, so that it can be used as input for algorithms to process. The reason

for encoding is that most machine learning algorithms work with numbers and not with text or categorical variables.

The Main focus of this Notebook is to understand

- What is Categorical Data and Why to Encode Data
- Different Data Encoding Techniques
- How to Implement it.

What is Categorical Data?

When we collect data, we often encounter different types of variables. One such type is categorical variables. Categorical variables are usually represented as 'strings' or 'categories' and are finite in number.

There are two types of categorical data -

- *Ordinal Data*
- *Nominal Data*

Here are a few examples of categorical variables:

- **Places:** Delhi, Mumbai, Ahmedabad, Bangalore, etc.
- **Departments:** Finance, Human resources, IT, Production.
- **Grades:** A, A-, B+, B, B- etc.

Ordinal Data:

The categories of ordinal data have an **Inherent Order**. This means that the categories can be **Ranked** or ordered from highest to lowest or vice versa.

For example, the variable “highest degree a person has” is an ordinal variable. The categories (High school, Diploma, Bachelors, Masters, PhD) can be ranked in order of the level of education attained.

Nominal Data:

The categories of nominal data **do not have an Inherent Order**. This means that the categories cannot be ranked or ordered.

For example, the variable “city where a person lives” is a nominal variable. The categories (Delhi, Mumbai, Ahmedabad, Bangalore, etc.) cannot be ranked or ordered.

What is Data Encoding?

Data Encoding is an important pre-processing step in Machine Learning. It refers to the process of converting categorical or textual data into numerical format, so that it can be used as input for algorithms to process. The reason for encoding is that most machine learning algorithms work with numbers and not with text or categorical variables.

Why it is Important?

- Most machine learning algorithms work only with numerical data, so categorical variables (such as text labels) must be transformed into numerical values.
- This allows the model to identify patterns in the data and make predictions based on those patterns.

- Encoding also helps to prevent bias in the model by ensuring that all features are equally weighted.
- The choice of encoding method can have a significant impact on model performance, so it is important to choose an appropriate encoding technique based on the nature of the data and the specific requirements of the model.

There are several methods for encoding categorical variables, including

1. One-Hot Encoding

2. Dummy Encoding

3. Ordinal Encoding

4. Binary Encoding

5. Count Encoding

6. Target Encoding

Let's take a closer look at each of these methods.

One-Hot Encoding:

- One-Hot Encoding is the **Most Common** method for encoding **Categorical** variables.
- a **Binary Column** is created for each **Unique Category** in the variable.

- If a category is present in a sample, the corresponding column is set to 1, and all other columns are set to 0.
- For example, if a variable has three categories 'A', 'B' and 'C', three columns will be created and a sample with category 'B' will have the value [0,1,0].

One-Hot Encoding

Places		New York	Boston	Chicago	California	New Jersey
New York	→	1	0	0	0	0
Boston		0	1	0	0	0
Chicago		0	0	1	0	0
California		0	0	0	1	0
New Jersey		0	0	0	0	1

```
# One-Hot Encoding:
# create a sample dataframe with a categorical variable
df = pd.DataFrame({'color': ['red', 'green', 'blue', 'red']})

# perform one-hot encoding on the 'color' column
one_hot = pd.get_dummies(df['color'])

# concatenate the one-hot encoding with the original dataframe
df1 = pd.concat([df, one_hot], axis=1)

# drop the original 'color' column
df1 = df1.drop('color', axis=1)
```

Dummy Encoding

- Dummy coding scheme is **similar to one-hot encoding**.
- This categorical data encoding method transforms the categorical variable into a set of binary variables [0/1].

- In the case of **one-hot encoding**, for N categories in a variable, it uses N binary variables.
- The dummy encoding is a small improvement over one-hot-encoding. Dummy encoding uses N-1 features to represent N labels/categories.

Dummy Encoding

Places		New York	Boston	Chicago	California	New Jersey
New York		0	0	0	0	0
Boston		1	0	0	0	0
Chicago		0	1	0	0	0
California		0	0	1	0	0
New Jersey		0	0	0	1	0

One-Hot Encoding vs Dummy Encoding:

One-Hot Encoding — N categories in a variable, N binary variables.

Dummy encoding — N categories in a variable, N-1 binary variables.

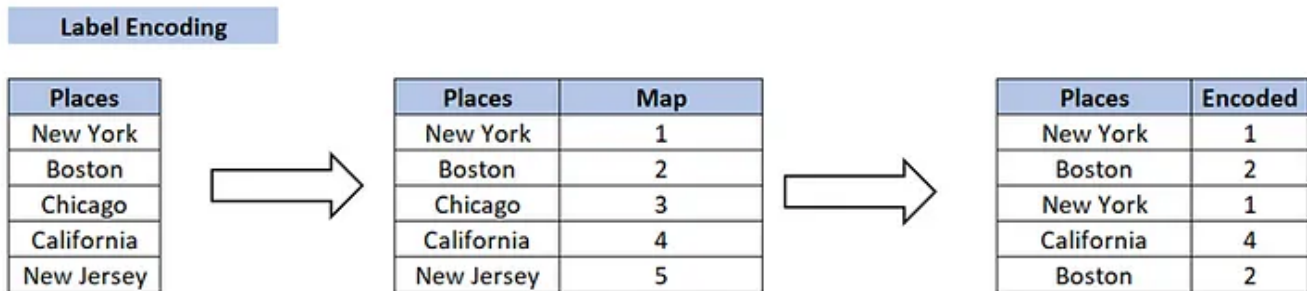
```
# Create a sample dataframe with categorical variable
data = {'Color': ['Red', 'Green', 'Blue', 'Red', 'Blue']}
df = pd.DataFrame(data)

# Use get_dummies() function for dummy encoding
dummy_df = pd.get_dummies(df['Color'], drop_first=True, prefix='Color')

# Concatenate the dummy dataframe with the original dataframe
df = pd.concat([df, dummy_df], axis=1)
```

Label Encoding:

- Each unique category is assigned a **Unique Integer** value.
- This is a simpler encoding method, but it has a **Drawback** in that the assigned integers may be **misinterpreted** by the machine learning algorithm as **having an Ordered Relationship** when in fact they **do not**.



```
from sklearn.preprocessing import LabelEncoder

# Create a sample dataframe with categorical data
df = pd.DataFrame({'color': ['red', 'green', 'blue', 'red', 'green']})

print(f"Before Encoding the Data:\n\n{df}\n")

# Create a LabelEncoder object
le = LabelEncoder()

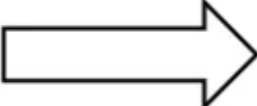
# Fit and transform the categorical data
df['color_label'] = le.fit_transform(df['color'])
```

Ordinal Encoding:

- Ordinal Encoding is used when the **categories** in a variable have a **Natural Ordering**.
- In this method, the **categories** are assigned a **numerical value** based on their order, such as 1, 2, 3, etc.

For example, if a variable has categories 'Low', 'Medium' and 'High', they can be assigned the values 1, 2, and 3, respectively.

Ordinal Encoding

Grades		Grades	Encoded
A		A	4
B		B	3
C		C	2
D		D	1
Fail		Fail	0

```
# Ordinal Encoding:
# create a sample dataframe with a categorical variable
df = pd.DataFrame({'quality': ['low', 'medium', 'high', 'medium']})
print(f"Before Encoding the Data:\n\n{df}\n")

# specify the order of the categories
quality_map = {'low': 0, 'medium': 1, 'high': 2}

# perform ordinal encoding on the 'quality' column
df['quality_map'] = df['quality'].map(quality_map)
```

Binary Encoding:

- Binary Encoding is similar to One-Hot Encoding, but instead of creating a separate column for each category, the categories are represented as binary digits.

- For example, if a variable has four categories 'A', 'B', 'C' and 'D', they can be represented as 0001, 0010, 0100 and 1000, respectively.

Binary Encoding



```
# Binary Encoding:

import pandas as pd

# create a sample dataframe with a categorical variable
df = pd.DataFrame({'animal': ['cat', 'dog', 'bird', 'cat']})
print(f"Before Encoding the Data:\n\n{df}\n")

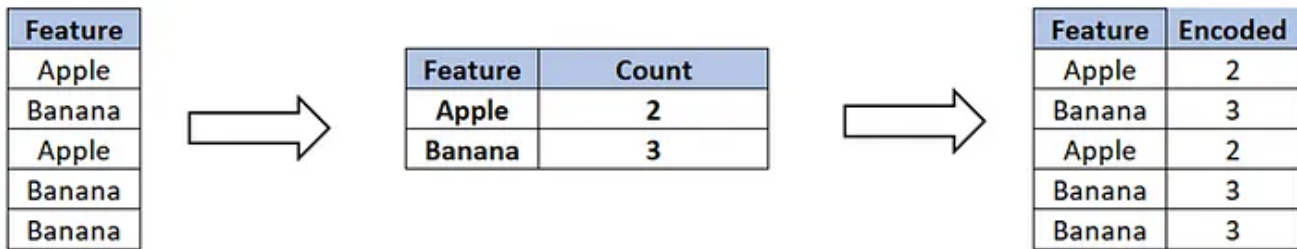
# perform binary encoding on the 'animal' column
animal_map = {'cat': 0, 'dog': 1, 'bird': 2}
df['animal'] = df['animal'].map(animal_map)
df['animal'] = df['animal'].apply(lambda x: format(x, 'b'))

# print the resulting dataframe
print(f"After Encoding the Data:\n\n{df}\n")
```

Count Encoding:

- Count Encoding is a method for encoding categorical variables by **counting the number of times a category appears** in the dataset.
- For example, if a variable has categories 'A', 'B' and 'C' and category 'A' appears 10 times in the dataset, it will be assigned a value of 10.

Count Encoding



```
# Count Encoding:
# create a sample dataframe with a categorical variable
df = pd.DataFrame({'fruit': ['apple', 'banana', 'apple', 'banana']})
print(f"Before Encoding the Data:\n\n{df}\n")

# perform count encoding on the 'fruit' column
counts = df['fruit'].value_counts()
df['fruit'] = df['fruit'].map(counts)

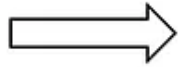
# print the resulting dataframe
print(f"After Encoding the Data:\n\n{df}\n")
```

Target Encoding:

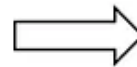
- This is a more **advanced encoding technique** used for dealing with **high cardinality categorical features**, i.e., features with many unique categories.
- The average target value for each category is calculated and this average value is used to replace the categorical feature.
- This has the **advantage of considering the relationship between the target and the categorical feature**, but it can also lead to **overfitting** if not used with caution.

Target Encoding

Feature	Target
Apple	0
Banana	1
Apple	0
Banana	0
Banana	1



Feature	Average(Target)
Apple	0
Banana	2 / 3 = 0.66



Feature	Encoded
Apple	0
Banana	0.66
Apple	0
Banana	0.66
Banana	0.66

```
# Create a sample dataframe with categorical data and target
df = pd.DataFrame({'color': ['red', 'green', 'blue', 'red', 'green'],
                    'target': [0, 1, 0, 1, 0]})
print(f"Before Encoding the Data:\n\n{df}\n")

# Calculate the mean target value for each category
target_mean = df.groupby('color')['target'].mean()

# Replace the categorical data with the mean target value
df['color_label'] = df['color'].map(target_mean)

print(f"After Encoding the Data:\n\n{df}")
```

Conclusion:

Data Encoding is an important step in the pre-processing of data for machine learning algorithms. The choice of encoding method depends on the type of data and the problem being solved. One-Hot Encoding is the most commonly used method, but other methods like Ordinal Encoding, Binary Encoding, and Count Encoding may also be used in certain situations.

Data Encoding

Machine Learning

Categorical Encoding

Categorical Data

Encoding

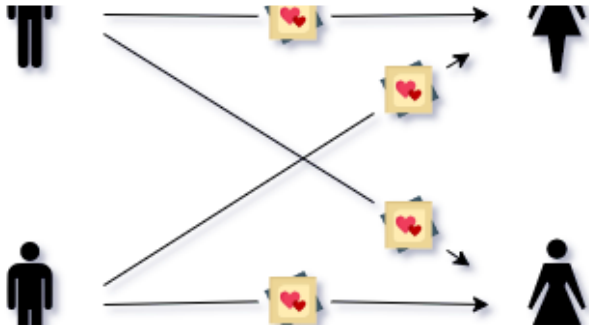


Written by Krishnakanth Naik Jarapala

42 Followers · Writer for AI Skunks

Follow

More from Krishnakanth Naik Jarapala and AI Skunks



K Krishnakanth Naik Jarapala in AI Skunks

Understanding Gale-Shapley (Stable Matching) Algorithm and...

Hey Readers,

3 min read · Jan 31, 2023



5



122

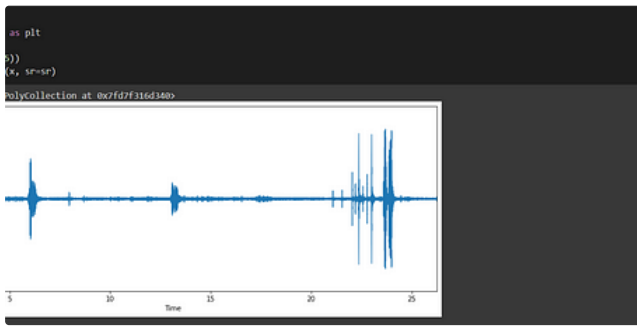



Shivani Shekhawat in AI Skunks

The Significance of Distance and Similarity Measures in Clustering

Abstract

18 min read · Mar 13, 2023



 Aditi Yadav in AI Skunks

Pre-Processing of Audio Data

INTRODUCTION:

6 min read · Mar 27, 2023



2



1



5

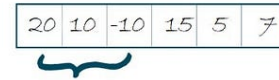


See all from Krishnakanth Naik Jarapala

See all from AI Skunks

Sliding Window Algorithm

Sliding Window



Slide one element
forward



 Krishnakanth Naik Jarapala in AI Skunks

Introduction to Sliding Window Algorithm

by Krishnakanth Naik Jarapala,

8 min read · Apr 20, 2023

Recommended from Medium



 Huda Saleh

Categorical Variables Encoding


Categorical variables are a common type of data encountered in machine learning tasks...

6 min read · Sep 9, 2023



4



 Indraneel Dutta Baruah in ANOLYTICS

All you need to know about encoding techniques!

How to use label encoding, one hot encoding, catboost encoding, etc. along with its Pytho...

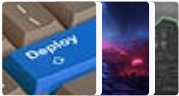
14 min read · Sep 29, 2023



109



Lists



Predictive Modeling w/ Python

20 stories · 894 saves



Natural Language Processing

1184 stories · 659 saves



Practical Guides to Machine Learning

10 stories · 1035 saves



The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 302 saves





Ridwan Abdurahman

Exploring Categorical Data Encoding: Label Encoding vs. One...

Photo by Firmbee.com on Unsplash

6 min read · Sep 26, 2023



13



Federico Viscioletti

How to Choose the Best Categorical Encoding Method

aka do not one-hot encode everything.



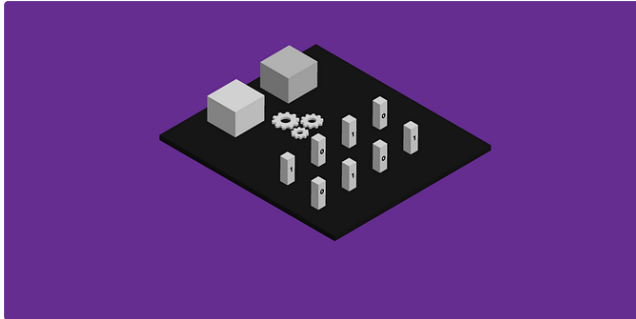
4 min read · Oct 14, 2023



24



1



$$E = \ln \left(\frac{\text{Distribution of Good (Events) in the Category}}{\text{Distribution of Bad (Non-Events) in the Category}} \right)$$

E:

E is the Weight of Evidence for the category.

ln presents the natural logarithm.

"Distribution of Good" refers to the proportion of positive (1) target variable instances within the category.

"Distribution of Bad" refers to the proportion of negative (0) target variable instances within the category.



Paresh Patil

The ultimate guide to Encoding Numerical Features in Machine...

Table of Contents:

5 min read · Aug 17, 2023



5



Ishan | Virginia Tech & IIT Delhi | 🧠 ML/AI

Weight of Evidence (WoE) Encoding — with Python Code

Weight of Evidence (WOE) quantifies the strength of the relationship between a...

4 min read · Oct 3, 2023



58



See more recommendations