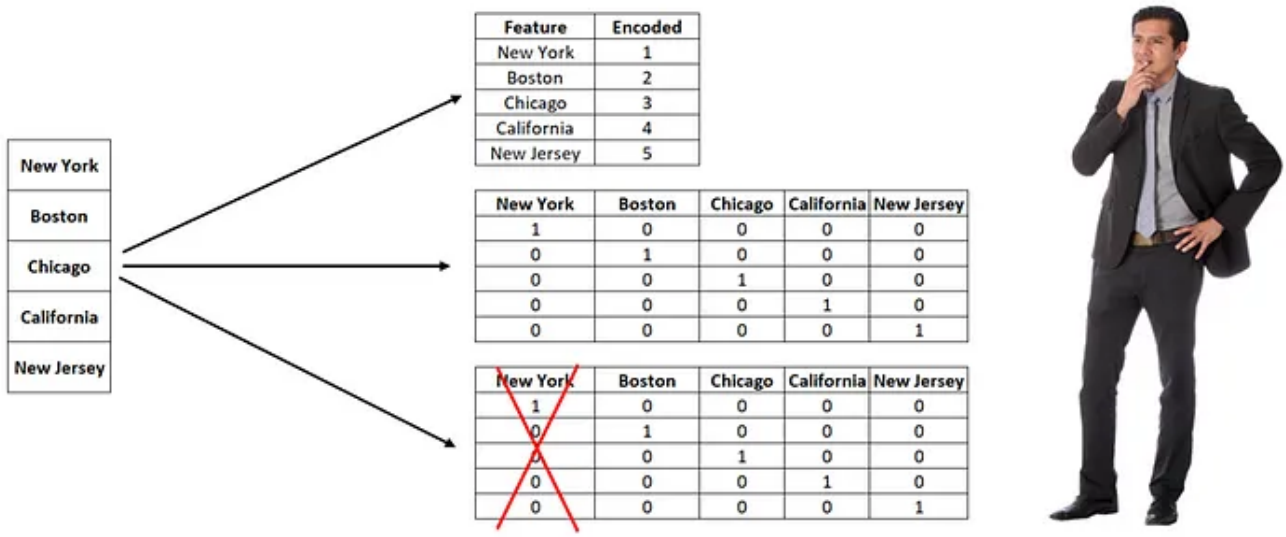
Categorical Data Encoding Techniques

Krishnakanth Naik Jarapala · Follow
Published in AI Skunks
7 min read · Mar 14



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.

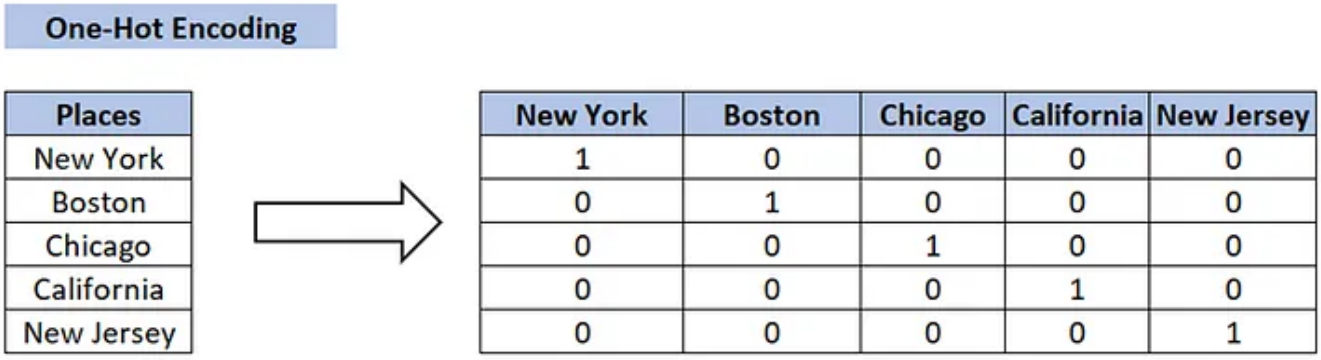
One-Hot Encoding:

• 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:
# 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)
```

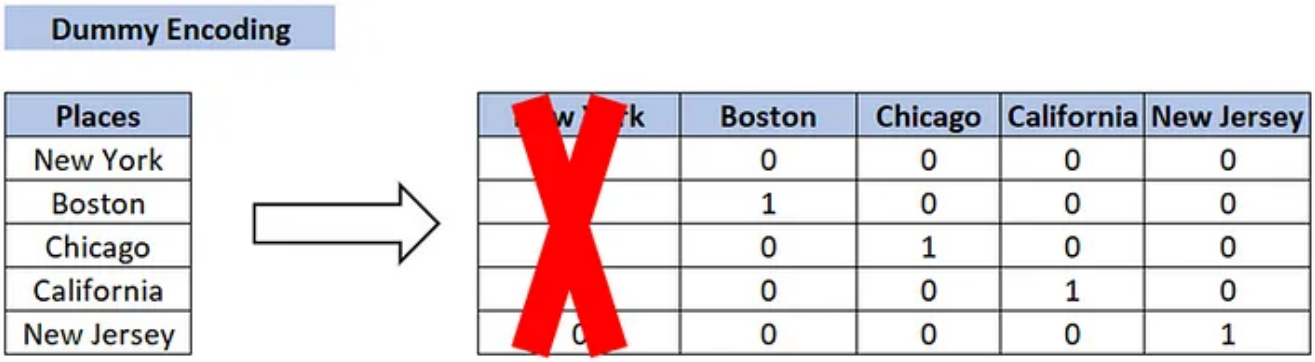
Dummy Encoding

• 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.

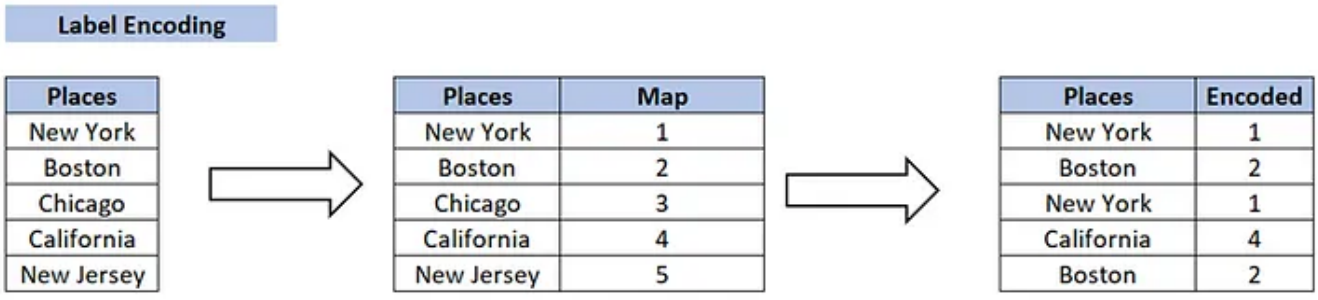


> *One-Hot Encoding vs Dummy Encoding:*
>
> ***One-Hot Encoding** — N categories in a variable, **N** binary variables.*
>
> ***Dummy encoding** — N categories in a variable, **N-1** binary variables.*

```
# 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.**



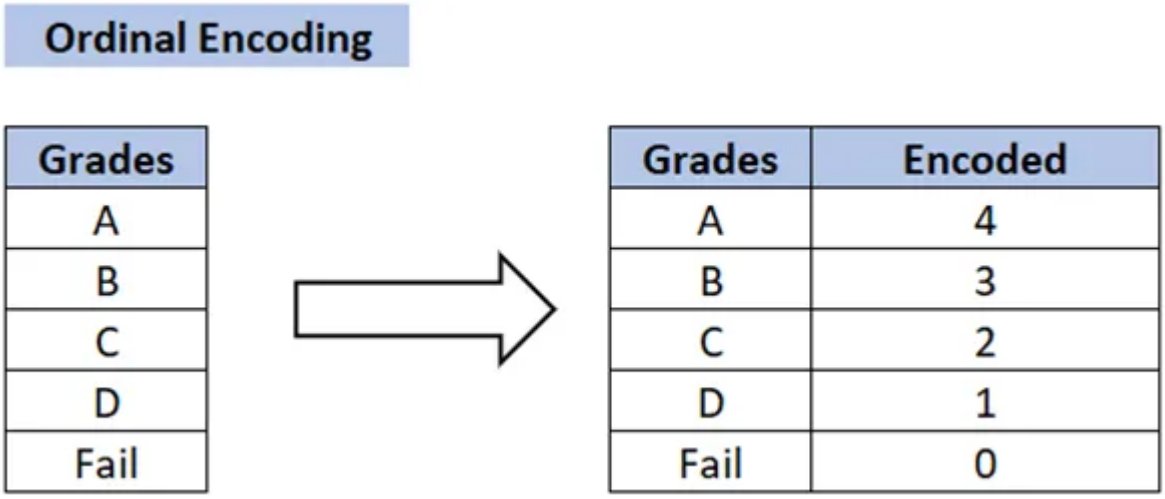
```
from sklearn.preprocessing import LabelEncoder

# 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:
# 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**.

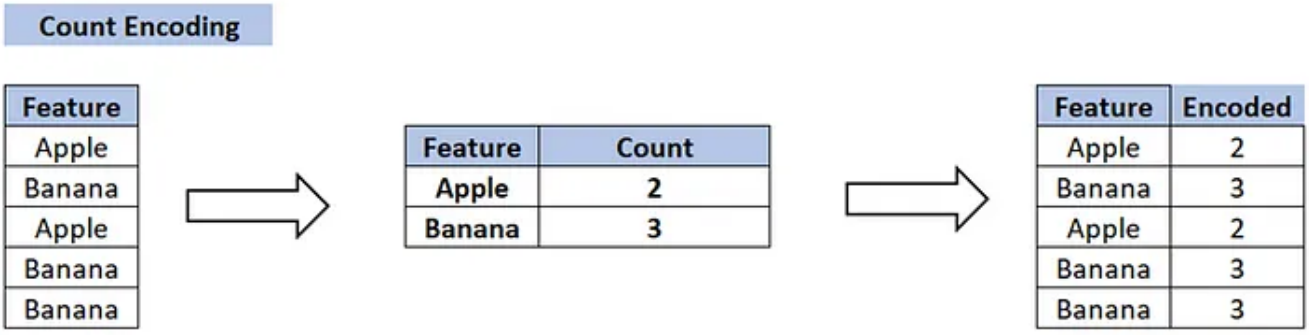
- For example, if a variable has four categories ‘A’, ‘B’, ‘C’ and ‘D’, they can be represented as 0001, 0010, 0100 and 1000, respectively.


```
# Binary Encoding:

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.



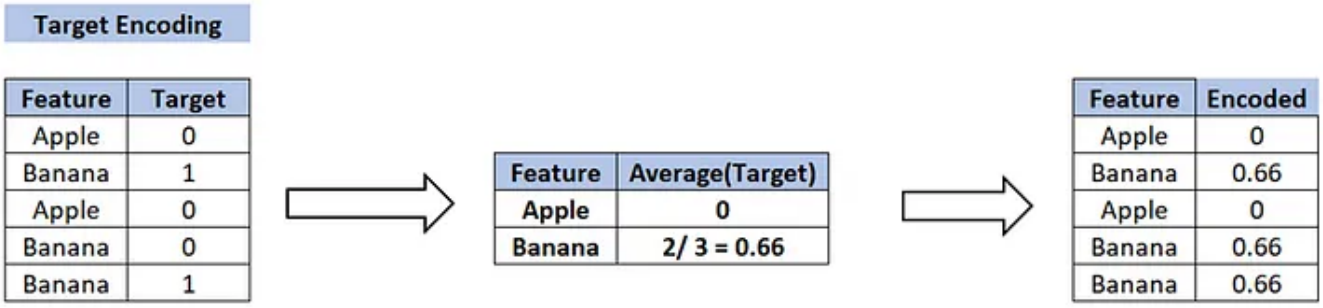

```
# Count Encoding:
# create a sample dataframe with a categorical variable
df = pd.DataFrame({'fruit': ['apple', 'banana', 'apple', 'banana']})
print(f"Before Encoding the Data:\n\n{df}\n")

# 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.




```
# 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.

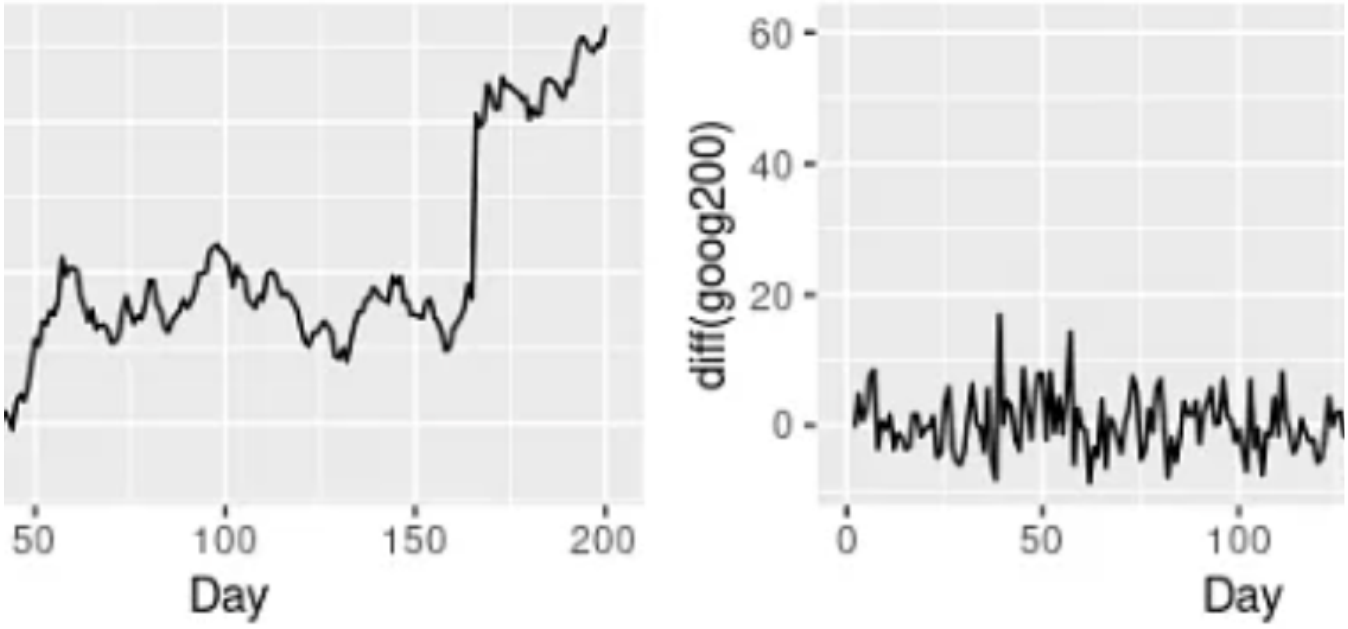
Data Encoding | Machine Learning | Categorical Encoding | Categorical Data | Encoding

Follow

Written by Krishnakanth Naik Jarapala

27 Followers · Writer for AI Skunks

More from Krishnakanth Naik Jarapala and AI Skunks



Krishnakanth Naik Jarapala

Crash Course in Time Series Analysis and Forecasting

by Krishnakanth Naik Jarapala, Venkata Bhargavi Sikhakolli

13 min read · Apr 10

63

0xdevshah in AI Skunks

Knowledge Graphs: A Comprehensive Analysis

Knowledge graphs are becoming increasingly important in NLP due to their ability to model the relationships between entities and concepts...

8 min read · Apr 9

63

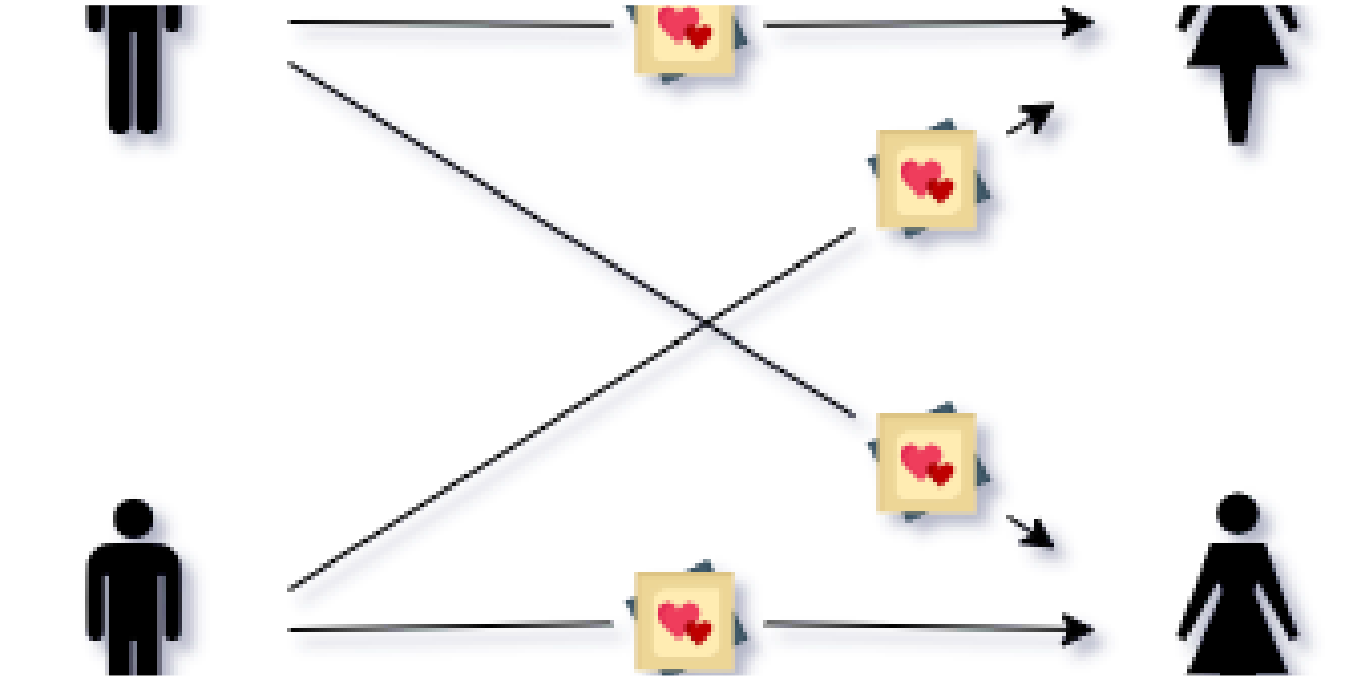
Cibaca Khandelwal in AI Skunks

Crash course in Time Series Forecasting — Techniques, Tools, and Best Practices

The article “Crash course in Time Series Forecasting — Techniques, Tools, and Best Practices” provides a comprehensive guide to time...

13 min read · Apr 10

1



Krishnakanth Naik Jarapala in AI Skunks

Understanding Gale-Shapley (Stable Matching) Algorithm and its Time Complexity

Hey Readers,

3 min read · Feb 1

3

See all from Krishnakanth Naik Jarapala

See all from AI Skunks

Recommended from Medium

...on ...rital Status	Middle School	High School	Bachelor's	Master's	Ph.D
...ried	18	36	21	9	6
...d	12	36	45	36	21
...d	6	9	9	3	3
...d	3	9	9	6	3
	39	90	84	54	33

Maninder Singh

Understanding Categorical Correlations with Chi-Square Test and Cramer's V

In this, I explained about correlation(code) between categorical features which I Learned when wanted to find the same in one of my...

9 min read · Jun 18

203

Aicha Bokbot in Towards Data Science

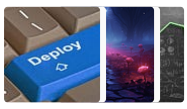
4 ways to encode categorical features with high cardinality

We explore 4 methods to encode categorical variables with high cardinality: target encoding, count encoding, feature hashing and embedding.

· 9 min read · Jun 26

46

Lists



Predictive Modeling w/ Python
20 stories · 440 saves

Practical Guides to Machine Learning
10 stories · 508 saves

Natural Language Processing
666 stories · 271 saves

The New Chatbots: ChatGPT, Bard, and Beyond
13 stories · 131 saves

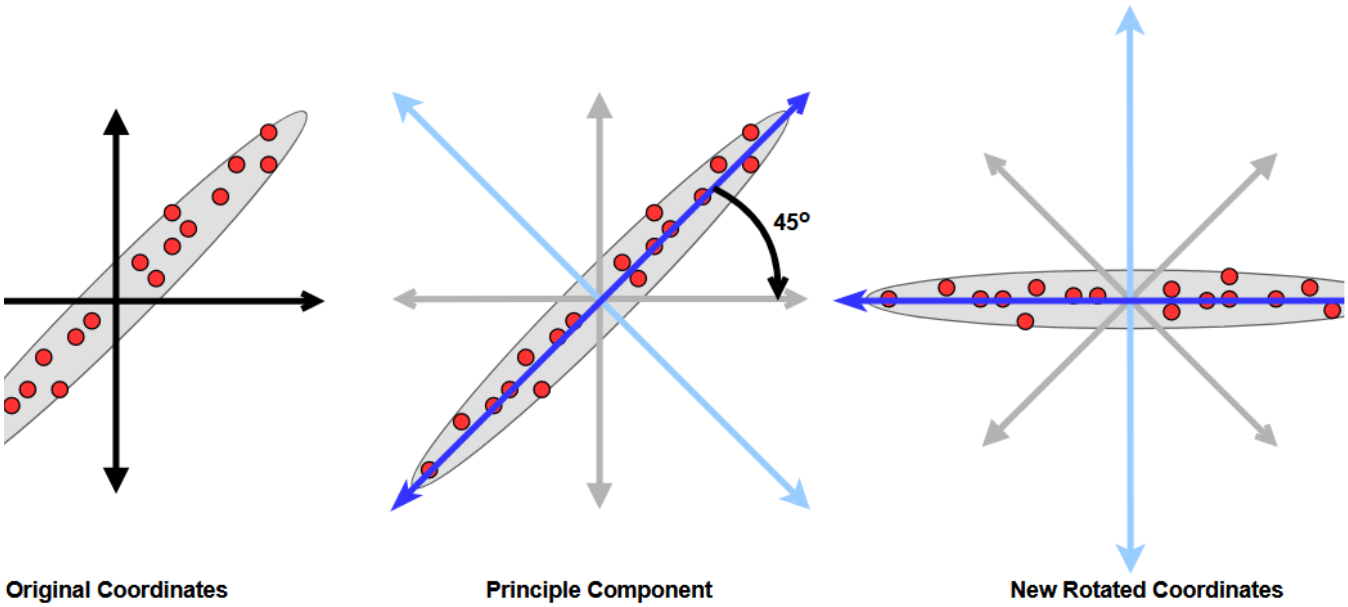
Huda Saleh

Categorical Variables Encoding

Categorical variables are a common type of data encountered in machine learning tasks. These variables represent categories or labels and...

6 min read · Sep 9

3



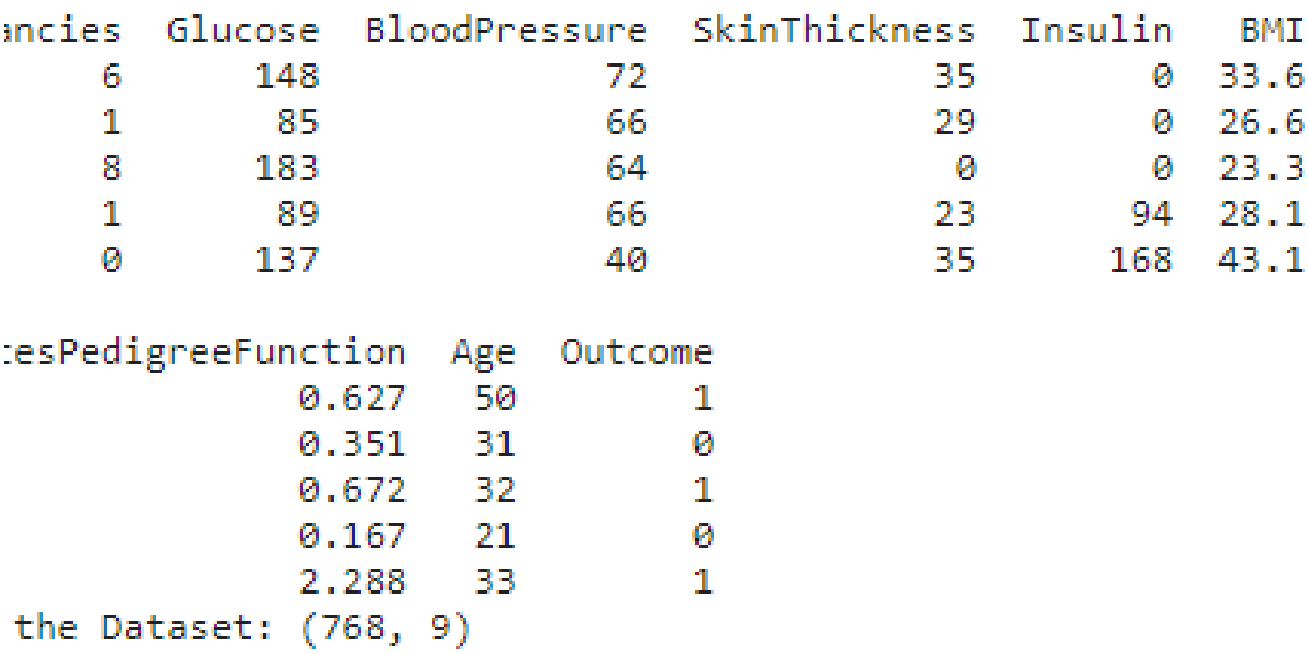
Deniz Gunay

Principal Component Analysis (PCA)

In the ever-expanding landscape of data analysis and machine learning, managing high-dimensional datasets is a formidable challenge...

19 min read · Sep 21

30 1



```
ıncies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
     6      148             72             35        0  33.6
     1       85             66             29        0  26.6
     8      183             64              0        0  23.3
     1       89             66             23       94  28.1
     0      137             40             35      168  43.1

:esPedigreeFunction  Age  Outcome
              0.627   50        1
              0.351   31        0
              0.672   32        1
              0.167   21        0
              2.288   33        1
 the Dataset: (768, 9)
```

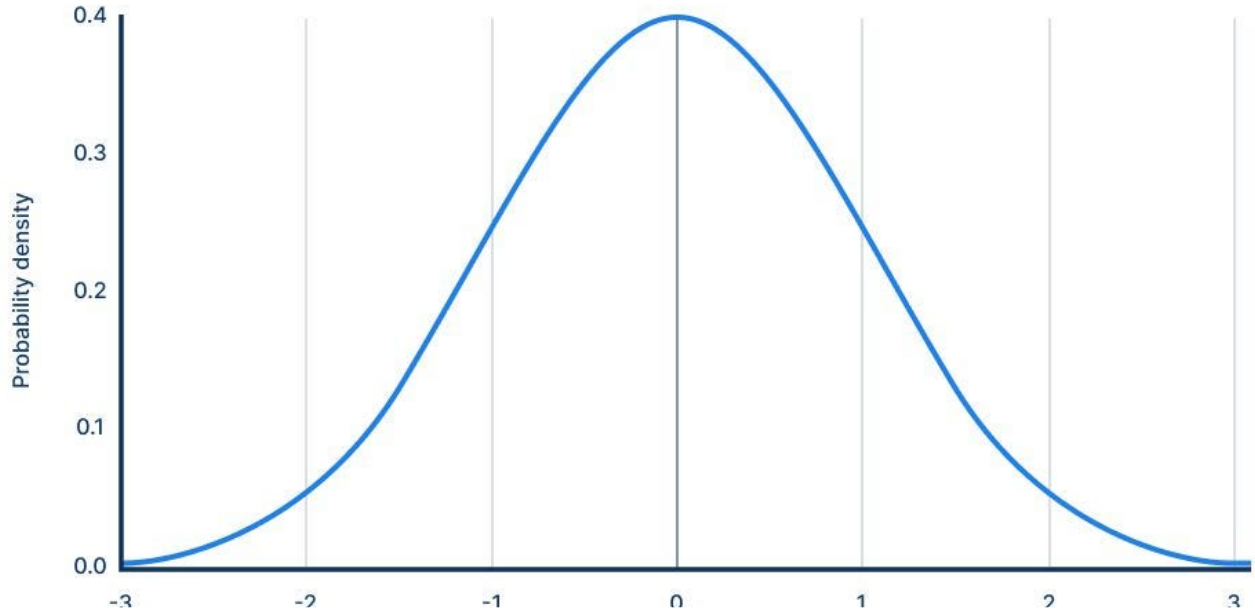
Saurav Agrawal

Feature Selection Using Lasso Regression

Lasso Regression is a regularized linear regression that includes a L1 penalty. Lasso Regression can also be used for feature selection...

3 min read · Jun 6

9



Sruthy Nath

Probability Distributions Demystified: Normal Distribution

Probability distributions play a pivotal role in various fields, including statistics, machine learning, and data science. Among the...

3 min read · Aug 4

6