

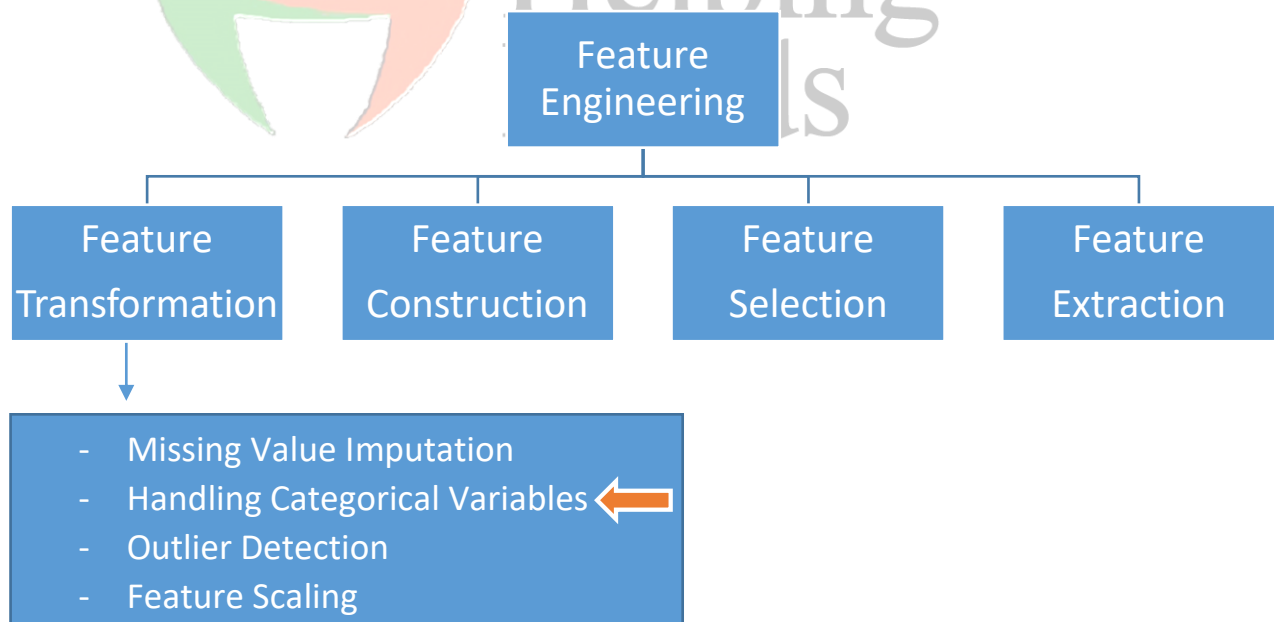
# Data Science | 30 Days of Machine Learning | Day - 9

Educator Name: Nishant Dhote  
Support Team: **+91-7880-113-112**

## ----Today Topics | Day 09----

- **How to Encode “Categorical Variables”?**
- What is categorical data?
- **Type of categorical data?**
- What is Ordinal Data?
- Ordinal Encoding
- Label Encoding
- What is Nominal Data?
- One Hot Encoding

Dataset Link GitHub: [https://github.com/TheiScale/30\\_Days\\_Machine\\_Learning/](https://github.com/TheiScale/30_Days_Machine_Learning/)



## - What is categorical data?

Categorical data refers to a form of information that can be stored and identified based on their names or labels. It is a type of qualitative data that can be grouped into categories instead of being measured numerically.

This data type is made up of categorical variables that show things like a person's gender, hometown, and so on. Categorical measurements are not given in numbers but rather in natural language descriptions.

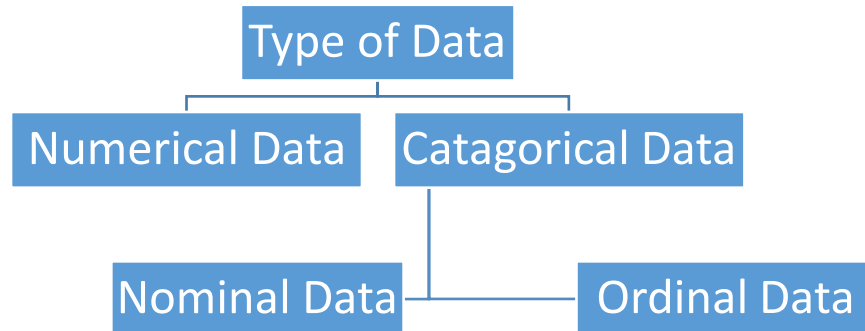
Numbers can sometimes represent it, but those numbers don't mean anything mathematically. The following are some examples of this data type:

- Birthdate
- Favourite sport
- Hair Colour
- Height

In the above example, both the birthdate and the postcode are made up of numbers. It is regarded as categorical data even though it includes numbers. Calculating the average is a simple way to determine if the provided data is categorical or numerical.

If you can figure out the average, it is considered numerical data. If you can't figure out the average, then it's considered categorical data.

## - Type of categorical data?



### - Nominal Data

Nominal data is a type of data that consists of categories that can't be ordered or ranked. It is also called a nominal scale. Nominal data can't be ranked or measured in any way. Still, nominal data can be both qualitative and quantitative at times.

Some examples of nominal data are symbols, words, letters, and the gender of a person, state and your engineering branch.

### - Ordinal Data

Ordinal data is a category of data that has a natural order. It is often used in surveys, questionnaires, and the fields of finance and economics. Ordinal data stands out since it is impossible to differentiate between data values.

Clothing sizes are one example of this type of data (small, medium, and large are not measurable differences, but they are clearly ordered to show size comparisons). Exam Grade or Division.

## - What is Encoding?

Data encoding involves converting a sequence of text characters into binary code so computers, which operate using binary numbers, can process, store, or transmit that textual information. Decoding occurs when the information is then translated from binary form into a readable version.

### “Ordinal Encoding”

When we have a feature where variables have some order/rank.


Original Encoding	Ordinal Encoding
Poor	1
Good	2
Very Good	3
Excellent	4

For example: Student's performance, Customer's review, Education of person, etc...

In the above example, we have orders/ranks/sequences. We can assign ranks based on student's performance, based on feedback given by customers, based on the highest education of the person. Those features are called Ordinal features.

### “Nominal Encoding”

When we have a feature where variables are just names and there is no order or rank to this variable's feature.



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1
0	1	0

For example: City of person lives in, Gender of person, Marital Status, etc...

In the above example, we do not have any order or rank, or sequence. All the variables in the respective feature are equal. We can't give them any orders or ranks. Those features are called Nominal features.

### Why ordinal encoding is referred as a **Label Encoding**?

Ordinal encoding is a technique that is used to transform categorical variables into a numerical format by assigning a unique value to each of its categories. It is also referred to as Label Encoding.

For example, we have customer feedback data based on a survey or online feedback mechanism. It contains categories - very dissatisfied, dissatisfied, neutral, satisfied, and very satisfied. To encode this variable using ordinal encoding, we can assign numerical values as mentioned below –

**very dissatisfied - 1**

**dissatisfied - 2**

**neutral - 3**

**satisfied - 4**

**very satisfied – 5**

Ordinal encoding assumes that categories in categorical variables have clear, natural, and intrinsic ordering to their categories. It does not work

for nominal categorical variables as no relationship exists between categories of a nominal variable. In our previous example, we encoded the categorical variable by assigning the lowest numerical value of 1 to the very dissatisfied category and the highest value of 5 to the very satisfied category.

.

This way, we were able to preserve the natural ordering of the categories - very dissatisfied < dissatisfied < neutral < satisfied < very satisfied was retained in  $1 < 2 < 3 < 4 < 5$ . Suppose we have another categorical variable, which contains red, blue, and green categories. We can encode this variable using ordinal encoding by assigning 1 to red, 2 to blue, and 3 to green, but it may lead to incorrect results. As encoded values have a natural ordering between them -  $1 < 2 < 3$  will be there, but red < blue < green does not exist.

---Start Coding---

**#Import Library | pandas | numpy**

```
import numpy as np
import pandas as pd
```

**#Import Datasets**

```
df = pd.read_csv('customer.csv')
```

**#Review Sample Data**

```
df.sample(5)
```

**#Eliminate Unnecessary Columns**

```
df = df.iloc[:,2:]
---
df
df.head()
```

## #Train\_Test\_Split

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =
train_test_split(df.iloc[:,0:2],df.iloc[:,-
1:],test_size=0.2)
```

---

## #Import\_OrdinalEncoder

```
from sklearn.preprocessing import OrdinalEncoder
```

---

```
x_train
```

## #Define Categories

```
oe =
OrdinalEncoder(categories=[['Poor','Average','Good'], ['Sch
ool','UG','PG']])
```

---

```
oe.fit(x_train)
```

## #Transform Ordinal Encoding

```
x_train = oe.transform(x_train)
x_test = oe.transform(x_test)
```

---

```
x_train
```

---

```
oe.categories_
```

## #Label Encoding

```
from sklearn.preprocessing import LabelEncoder
```

---

```
le = LabelEncoder()
```

```

---
le.fit(y_train)

---
le.classes_

---
y_train = le.transform(y_train)
y_test = le.transform(y_test)

---
y_train

```

## - One Hot Encoding for Nominal Data

	brand	km_driven	fuel	owner	selling_price
0	Maruti	145500	Diesel	First Owner	450000
1	Skoda	120000	Diesel	Second Owner	370000
2	Honda	140000	Petrol	Third Owner	158000
3	Hyundai	127000	Diesel	First Owner	225000
4	Maruti	120000	Petrol	First Owner	130000

## Difference Between Label Encoding VS OHE

Label Encoding			One Hot Encoding			
Food Name	Categorical #	Calories				
Apple	1	95	Apple	Chicken	Broccoli	Calories
Chicken	2	231	1	0	0	95
Broccoli	3	50	0	1	0	231
			0	0	1	50



**---Start Coding---****#Import Library | pandas | numpy**

```
import numpy as np
import pandas as pd
```

**#Import Datasets**

```
df = pd.read_csv('cars.csv')
```

**#Review Sample Data**

```
df.sample(5)
```

```
---
df['owner'].value_counts()
```

```
---
df['brand'].value_counts()
df['brand'].nunique()
```

```
---
df['fuel'].value_counts()
df['fuel'].nunique()
```

**#1. One Hot Encoding Using Pandas**

```
pd.get_dummies(df, columns=['fuel', 'owner'])
```

**#2. K-1 One Hot Encoding**

```
pd.get_dummies(df, columns=['fuel', 'owner'], drop_first=True)
```

### #3. One hot encoding Using Sklearn

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test =
train_test_split(df.iloc[:,0:4],df.iloc[:,-
1],test_size=0.2,random_state=2)

---
df.head()
---
X_train.head()
---
X_test.head()
```

#### # Import OHE: SK Learn

```
from sklearn.preprocessing import OneHotEncoder
---
ohe = OneHotEncoder
---
X_train_new =
ohe.fit_transform(X_train[['fuel','owner']]).toarray()
---
X_test_new =
ohe.transform(X_test[['fuel','owner']]).toarray()
---
X_train_new
---
X_train_new.shape
---

np.hstack((X_train[['brand','km_driven']].values,X_train_new))
```

### MCQ Question:

<https://www.aionlinecourse.com/ai-quiz-questions/machine-learning/data-pre-processing>

### Data Story Telling (Day 9): Curious Data Minds

**Suggest Topic in Comment Box?**



Industries  
Helping  
Hands