Handle different types of data for encoding:

We all know that Machine learning cannot understand string type of data, it only understands numerical type of data.so in order to make it under stand the string type of data we need to convert it into numerical type of data.

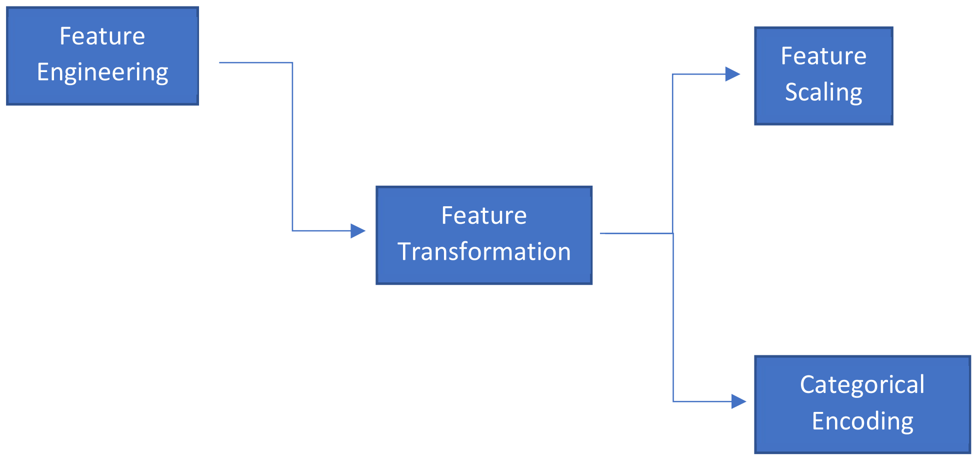
Note: This article doesn't focus on encoding of singular variable typed of data, rather its main focus is on **mixed variable** typed data. but before getting into multi variable typed data we will understand the types of data that is used in our day-to-day life for encoding and also understand encoding.

What is mixed variable typed data?

Hold on let us first understand singular variable typed of data first.

Encoding:

In machine learning, encoding refers to the process of converting categorical or textual data into a numerical format that can be used by machine learning algorithms for analysis and modelling. This is necessary because most machine learning algorithms can only work with numerical data. Also said Encoding is a part of Feature Engineering.



Categorical Encoding:

1. Categorical data refers to data that consists of a limited number of categories or levels, such as colours, gender, and country names.it may further be categorised as Nominal and Ordinal.

    1.1 Nominal data: Nominal data refers to data that does not have any inherent order or ranking. Examples of nominal data include colours, gender, and country names. To encode nominal data, we can use a technique called one-hot encoding. In one-hot encoding, we create a binary vector for each category in the nominal variable, where each vector has a 1 in the position corresponding to the category and 0s in all other positions.

    1.2 Ordinal data: Ordinal data refers to categorical data where the categories have a natural ordering or ranking. Examples of ordinal data include ratings, education levels, and income brackets. To encode ordinal data, we can use a technique called ordinal encoding. In ordinal encoding, we assign a unique numerical value to each category in the ordinal variable based on their order.

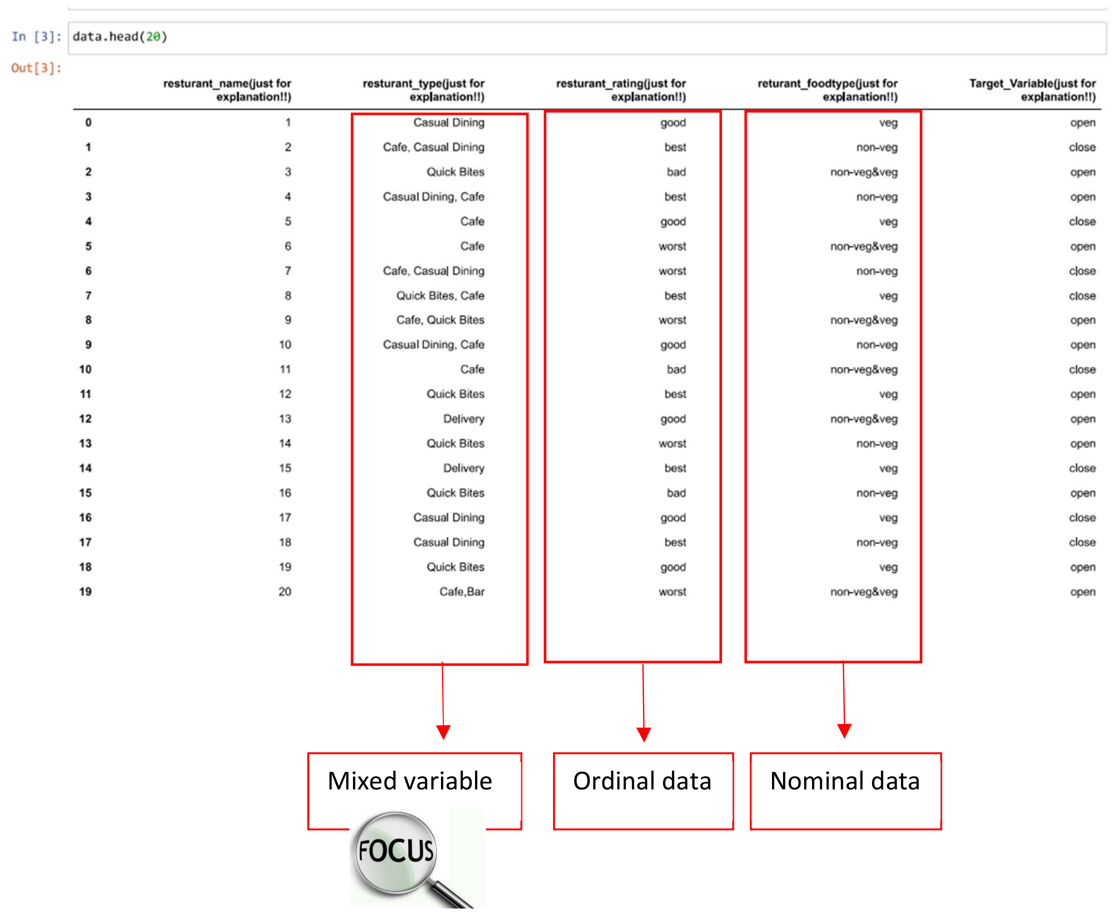
2. Numerical data: Numerical data refers to continuous or discrete numeric values. Examples of numerical data include age, height, and temperature. To encode numerical data, we can use a technique called normalization. In normalization, we rescale the values of the numerical variable to a common scale to avoid differences in magnitude affecting the results of the analysis. We can use methods such as Min-Max scaling, Z-score scaling, or Decimal scaling for normalization.

3. Text data: Text data refers to textual information such as documents, sentences, and paragraphs. To encode text data, we can use techniques such as bag-of-words, TF-IDF (term frequency-inverse document frequency), and word embeddings. Bag-of-words is a simple technique that counts the frequency of words in the text, while TF-IDF assigns weights to words based on their frequency and importance. Word embeddings use neural networks to represent words as high-dimensional vectors.

4. Date and time data: Date and time data refer to information about dates, times, and durations. To encode date and time data, we can use techniques such as date-partitioning, timestamp encoding, and duration encoding. Date-partitioning splits the date and time information into separate variables such as year, month, day, and time. Timestamp encoding assigns a unique numerical value to each point in time, while duration encoding encodes the duration between two points in time as a numerical value.

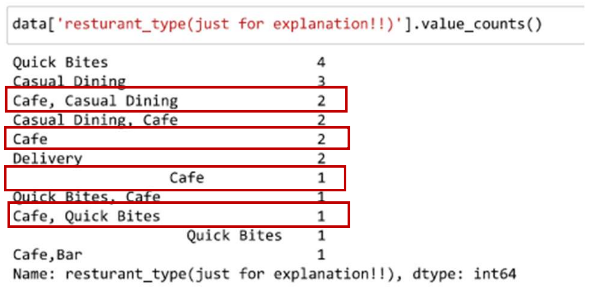
Answering to your question mixed variable is a type of variable which is combined with a string or a numerical with comma separated

Let us consider an example to understand more in depth:



```
In [3]: data.head(20)
Out[3]:
```

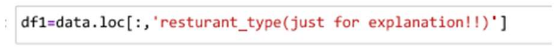
	resturant_name(just for explanation!!)	resturant_type(just for explanation!!)	resturant_rating(just for explanation!!)	returant_foodtype(just for explanation!!)	Target_Variable(just for explanation!!)
0	1	Casual Dining	good	veg	open
1	2	Cafe, Casual Dining	best	non-veg	close
2	3	Quick Bites	bad	non-veg&veg	open
3	4	Casual Dining, Cafe	best	non-veg	open
4	5	Cafe	good	veg	close
5	6	Cafe	worst	non-veg&veg	open
6	7	Cafe, Casual Dining	worst	non-veg	close
7	8	Quick Bites, Cafe	best	veg	close
8	9	Cafe, Quick Bites	worst	non-veg&veg	open
9	10	Casual Dining, Cafe	good	non-veg	open
10	11	Cafe	bad	non-veg&veg	close
11	12	Quick Bites	best	veg	open
12	13	Delivery	good	non-veg&veg	open
13	14	Quick Bites	worst	non-veg	open
14	15	Delivery	best	veg	close
15	16	Quick Bites	bad	non-veg	open
16	17	Casual Dining	good	veg	close
17	18	Casual Dining	best	non-veg	close
18	19	Quick Bites	good	veg	open
19	20	Cafe,Bar	worst	non-veg&veg	open



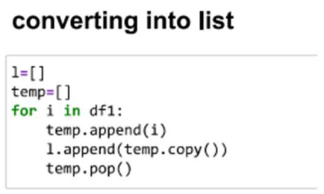
```
data['resturant_type(just for explanation!!)'].value_counts()
```

```
Quick Bites                          4
Casual Dining                        3
Cafe, Casual Dining                  2
Casual Dining, Cafe                  2
Cafe                                 2
Delivery                             2
                  Cafe               1
Quick Bites, Cafe                    1
Cafe, Quick Bites                    1
                    Quick Bites      1
Cafe,Bar                             1
Name: resturant_type(just for explanation!!), dtype: int64
```

we can see that café is mixed with wide space, 'casual Dining' and 'Quick Bites' hence it becomes very difficult to convert it into numerical as per the category type. The best way to separate he variables by making the separated variables as a new coloumn and then encoding it.let us create a script that will separate out the variable and create dummies.

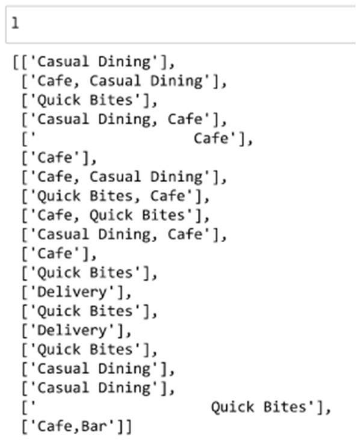


```
df1=data.loc[:,'resturant_type(just for explanation!!)']
```



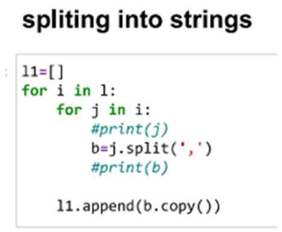
converting into list

```
l=[]
temp=[]
for i in df1:
    temp.append(i)
    l.append(temp.copy())
    temp.pop()
```



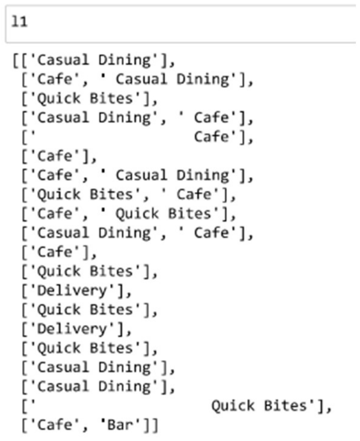
```
l
```

```
[['Casual Dining'],
 ['Cafe, Casual Dining'],
 ['Quick Bites'],
 ['Casual Dining, Cafe'],
 ['                   Cafe'],
 ['Cafe'],
 ['Cafe, Casual Dining'],
 ['Quick Bites, Cafe'],
 ['Cafe, Quick Bites'],
 ['Casual Dining, Cafe'],
 ['Cafe'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Casual Dining'],
 ['Casual Dining'],
 ['                     Quick Bites'],
 ['Cafe,Bar']]
```



spliting into strings

```
l1=[]
for i in l:
    for j in i:
        #print(j)
        b=j.split(',')
        #print(b)

    l1.append(b.copy())
```



```
l1
```

```
[['Casual Dining'],
 ['Cafe', ' Casual Dining'],
 ['Quick Bites'],
 ['Casual Dining', ' Cafe'],
 ['                   Cafe'],
 ['Cafe'],
 ['Cafe', ' Casual Dining'],
 ['Quick Bites', ' Cafe'],
 ['Cafe', ' Quick Bites'],
 ['Casual Dining', ' Cafe'],
 ['Cafe'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Casual Dining'],
 ['Casual Dining'],
 ['                     Quick Bites'],
 ['Cafe', 'Bar']]
```

removing space

```
new=[]
for i in l1:
    temp=[]
    for j in i:
        temp.append(j.strip())
    new.append(temp.copy())
    del(temp)
```

```
new
```

```
[['Casual Dining'],
 ['Cafe', 'Casual Dining'],
 ['Quick Bites'],
 ['Casual Dining', 'Cafe'],
 ['Cafe'],
 ['Cafe'],
 ['Cafe', 'Casual Dining'],
 ['Quick Bites', 'Cafe'],
 ['Cafe', 'Quick Bites'],
 ['Casual Dining', 'Cafe'],
 ['Cafe'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Delivery'],
 ['Quick Bites'],
 ['Casual Dining'],
 ['Casual Dining'],
 ['Quick Bites'],
 ['Cafe', 'Bar']]
```

unique values

```
unique=[]
for i in new:
    for j in i:
        unique.append(j)
unique_set=set(unique)
unique_list=list(unique_set)
```

```
unique_list
```

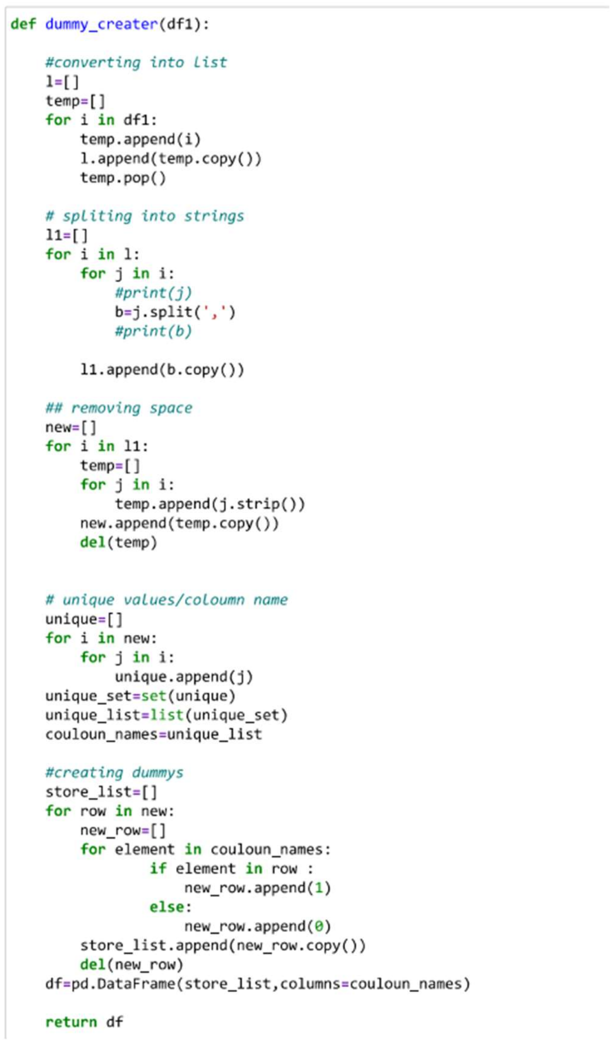
```
['Cafe', 'Casual Dining', 'Bar', 'Delivery', 'Quick Bites']
```

creating columns in a dataframe

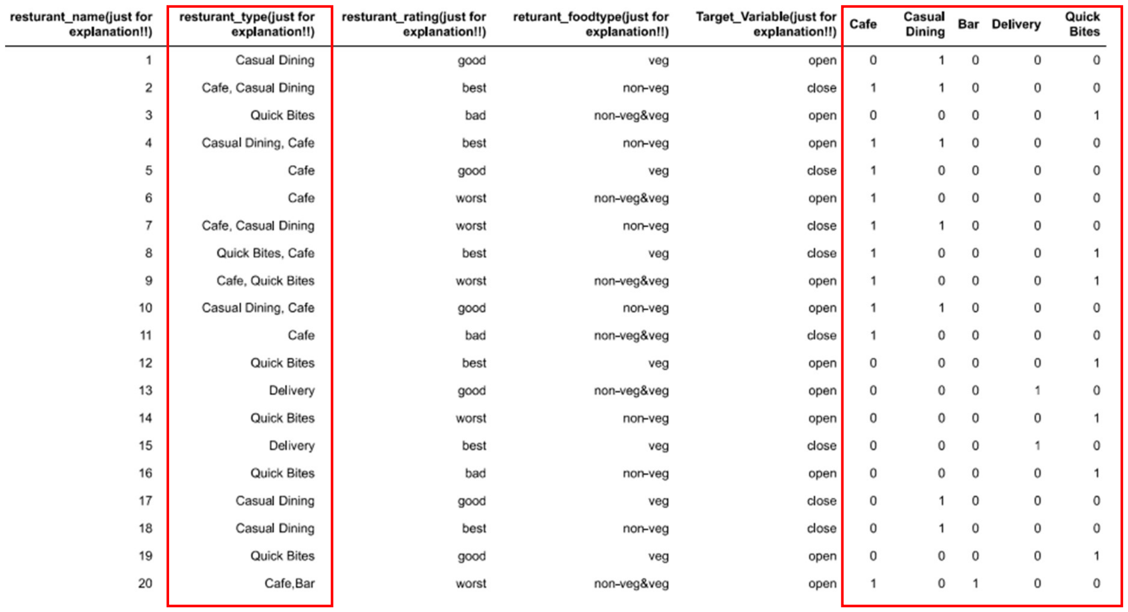
```
store_list=[]
for row in new:
    new_row=[]
    for element in unique_list:
            if element in row :
                new_row.append(1)
            else:
                new_row.append(0)
    store_list.append(new_row.copy())
    del(new_row)
df=pd.DataFrame(store_list,columns=unique_list)
```

```
df
```

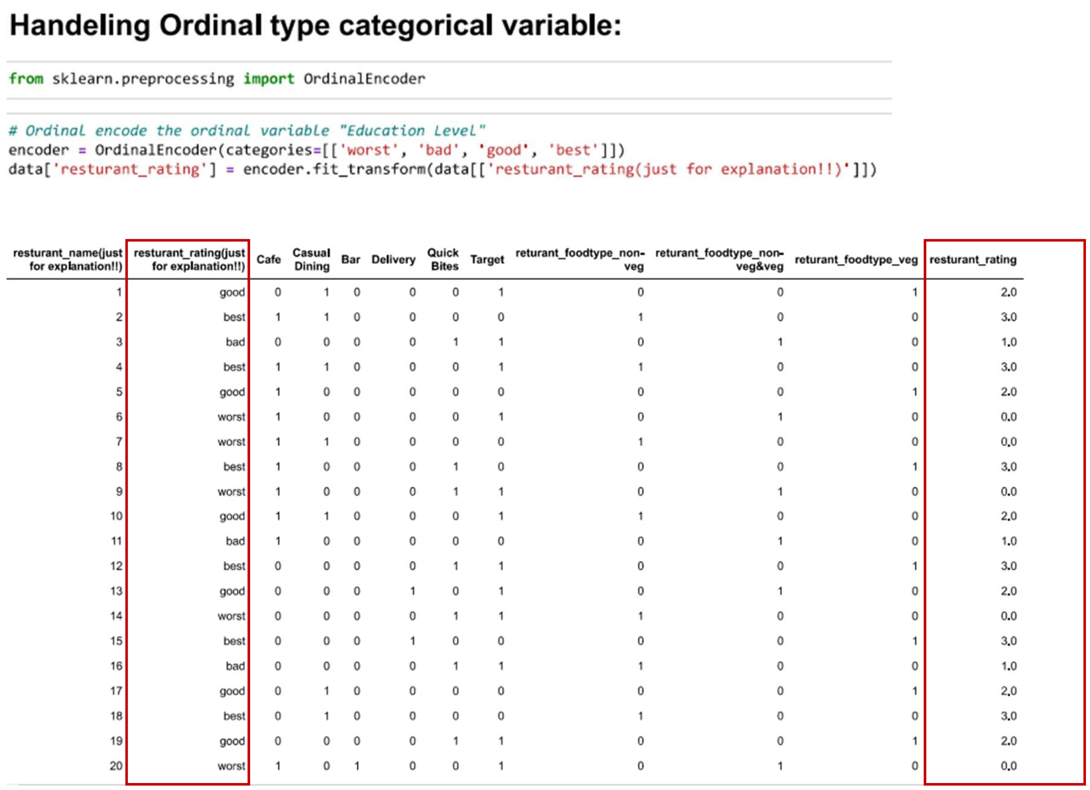
	Cafe	Casual Dining	Bar	Delivery	Quick Bites
0	0	1	0	0	0
1	1	1	0	0	0
2	0	0	0	0	1
3	1	1	0	0	0
4	1	0	0	0	0
5	1	0	0	0	0
6	1	1	0	0	0
7	1	0	0	0	1
8	1	0	0	0	1
9	1	1	0	0	0
10	1	0	0	0	0
11	0	0	0	0	1
12	0	0	0	1	0
13	0	0	0	0	1
14	0	0	0	1	0
15	0	0	0	0	1
16	0	1	0	0	0
17	0	1	0	0	0
18	0	0	0	0	1
19	1	0	1	0	0



```python
def dummy_creater(df1):

    #converting into List
    l=[]
    temp=[]
    for i in df1:
        temp.append(i)
        l.append(temp.copy())
        temp.pop()

    # spliting into strings
    l1=[]
    for i in l:
        for j in i:
            #print(j)
            b=j.split(',')
            #print(b)

        l1.append(b.copy())

    ## removing space
    new=[]
    for i in l1:
        temp=[]
        for j in i:
            temp.append(j.strip())
        new.append(temp.copy())
        del(temp)


    # unique values/coloumn name
    unique=[]
    for i in new:
        for j in i:
            unique.append(j)
    unique_set=set(unique)
    unique_list=list(unique_set)
    couloun_names=unique_list

    #creating dummys
    store_list=[]
    for row in new:
        new_row=[]
        for element in couloun_names:
                if element in row :
                    new_row.append(1)
                else:
                    new_row.append(0)
        store_list.append(new_row.copy())
        del(new_row)
    df=pd.DataFrame(store_list,columns=couloun_names)

    return df
```



resturant_name(just for explanation!!)	resturant_type(just for explanation!!)	resturant_rating(just for explanation!!)	returant_foodtype(just for explanation!!)	Target_Variable(just for explanation!!)	Cafe	Casual Dining	Bar	Delivery	Quick Bites
1	Casual Dining	good	veg	open	0	1	0	0	0
2	Cafe, Casual Dining	best	non-veg	close	1	1	0	0	0
3	Quick Bites	bad	non-veg&veg	open	0	0	0	0	1
4	Casual Dining, Cafe	best	non-veg	open	1	1	0	0	0
5	Cafe	good	veg	close	1	0	0	0	0
6	Cafe	worst	non-veg&veg	open	1	0	0	0	0
7	Cafe, Casual Dining	worst	non-veg	close	1	1	0	0	0
8	Quick Bites, Cafe	best	veg	close	1	0	0	0	1
9	Cafe, Quick Bites	worst	non-veg&veg	open	1	0	0	0	1
10	Casual Dining, Cafe	good	non-veg	open	1	1	0	0	0
11	Cafe	bad	non-veg&veg	close	1	0	0	0	0
12	Quick Bites	best	veg	open	0	0	0	0	1
13	Delivery	good	non-veg&veg	open	0	0	0	1	0
14	Quick Bites	worst	non-veg	open	0	0	0	0	1
15	Delivery	best	veg	close	0	0	0	1	0
16	Quick Bites	bad	non-veg	open	0	0	0	0	1
17	Casual Dining	good	veg	close	0	1	0	0	0
18	Casual Dining	best	non-veg	close	0	1	0	0	0
19	Quick Bites	good	veg	open	0	0	0	0	1
20	Cafe,Bar	worst	non-veg&veg	open	1	0	1	0	0

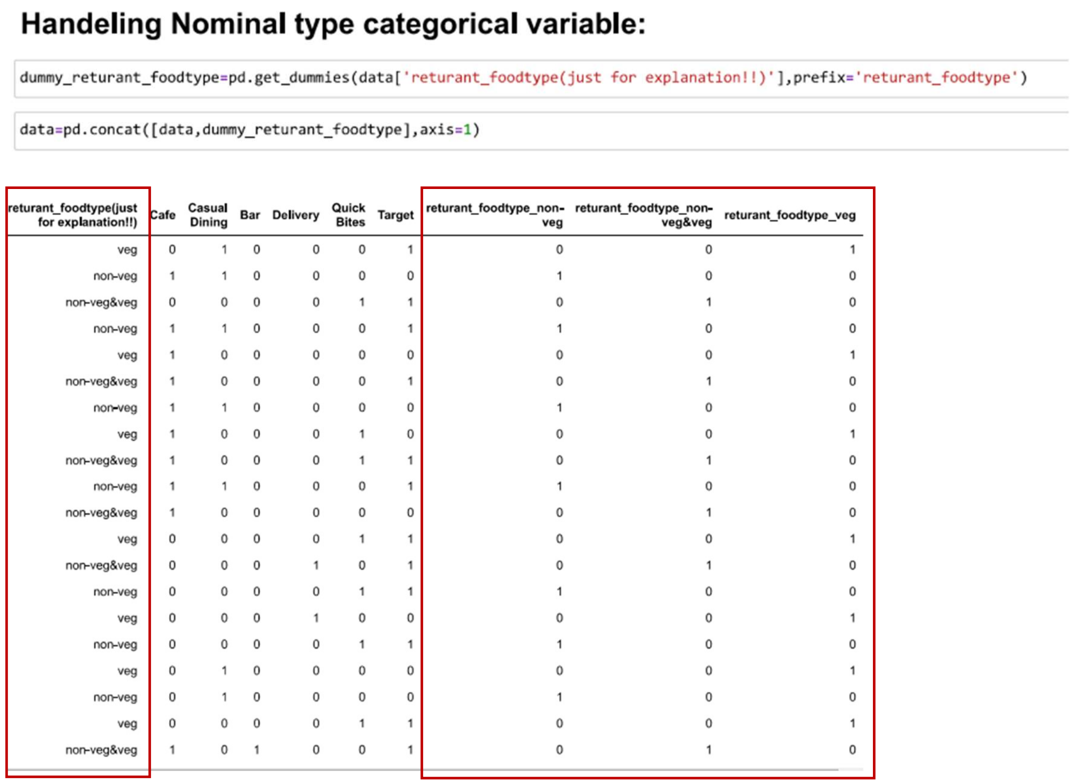


Handeling Ordinal type categorical variable:

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

```python
# Ordinal encode the ordinal variable "Education Level"
encoder = OrdinalEncoder(categories=[['worst', 'bad', 'good', 'best']])
data['resturant_rating'] = encoder.fit_transform(data[['resturant_rating(just for explanation!!)']])
```

resturant_name(just for explanation!!)	resturant_rating(just for explanation!!)	Cafe	Casual Dining	Bar	Delivery	Quick Bites	Target	returant_foodtype_non-veg	returant_foodtype_non-veg&veg	returant_foodtype_veg	resturant_rating
1	good	0	1	0	0	0	1	0	0	1	2.0
2	best	1	1	0	0	0	0	1	0	0	3.0
3	bad	0	0	0	0	1	1	0	1	0	1.0
4	best	1	1	0	0	0	1	1	0	0	3.0
5	good	1	0	0	0	0	0	0	0	1	2.0
6	worst	1	0	0	0	0	1	0	1	0	0.0
7	worst	1	1	0	0	0	0	1	0	0	0.0
8	best	1	0	0	0	1	0	0	0	1	3.0
9	worst	1	0	0	0	1	1	0	1	0	0.0
10	good	1	1	0	0	0	1	1	0	0	2.0
11	bad	1	0	0	0	0	0	0	1	0	1.0
12	best	0	0	0	0	1	1	0	0	1	3.0
13	good	0	0	0	1	0	1	0	1	0	2.0
14	worst	0	0	0	0	1	1	1	0	0	0.0
15	best	0	0	0	1	0	0	0	0	1	3.0
16	bad	0	0	0	0	1	1	1	0	0	1.0
17	good	0	1	0	0	0	0	0	0	1	2.0
18	best	0	1	0	0	0	0	1	0	0	3.0
19	good	0	0	0	0	1	1	0	0	1	2.0
20	worst	1	0	1	0	0	1	0	1	0	0.0



Handeling Nominal type categorical variable:

```python
dummy_returant_foodtype=pd.get_dummies(data['returant_foodtype(just for explanation!!)'],prefix='returant_foodtype')
```

```python
data=pd.concat([data,dummy_returant_foodtype],axis=1)
```

returant_foodtype(just for explanation!!)	Cafe	Casual Dining	Bar	Delivery	Quick Bites	Target	returant_foodtype_non-veg	returant_foodtype_non-veg&veg	returant_foodtype_veg
veg	0	1	0	0	0	1	0	0	1
non-veg	1	1	0	0	0	0	1	0	0
non-veg&veg	0	0	0	0	1	1	0	1	0
non-veg	1	1	0	0	0	1	1	0	0
veg	1	0	0	0	0	0	0	0	1
non-veg&veg	1	0	0	0	0	1	0	1	0
non-veg	1	1	0	0	0	0	1	0	0
veg	1	0	0	0	1	0	0	0	1
non-veg&veg	1	0	0	0	1	1	0	1	0
non-veg	1	1	0	0	0	1	1	0	0
non-veg&veg	1	0	0	0	0	0	0	1	0
veg	0	0	0	0	1	1	0	0	1
non-veg&veg	0	0	0	1	0	1	0	1	0
non-veg	0	0	0	0	1	1	1	0	0
veg	0	0	0	1	0	0	0	0	1
non-veg	0	0	0	0	1	1	1	0	0
veg	0	1	0	0	0	0	0	0	1
non-veg	0	1	0	0	0	0	1	0	0
veg	0	0	0	0	1	1	0	0	1
non-veg&veg	1	0	1	0	0	1	0	1	0