Capping (Floor & Ceiling)

Encoding Types

Day 19
17 nov,2022

Outlier Treatment

The treatment of the outlier values/cases is called Outlier Treatment. Typically outlier treatment is done by capping/flooring.

Capping : is replacing all higher side values exceeding a certain theoretical maximum or upper control limit (UCL) by the UCL value.

Statistical formula for UCL is UCL = Q3 + 1.5 * IQR

Flooring : is replacing all values falling below a certain theoretical minimum or lower control limit (UCL) by the LCL value.

Statistical formula for LCL is LCL = Q1 – 1.5 * IQR

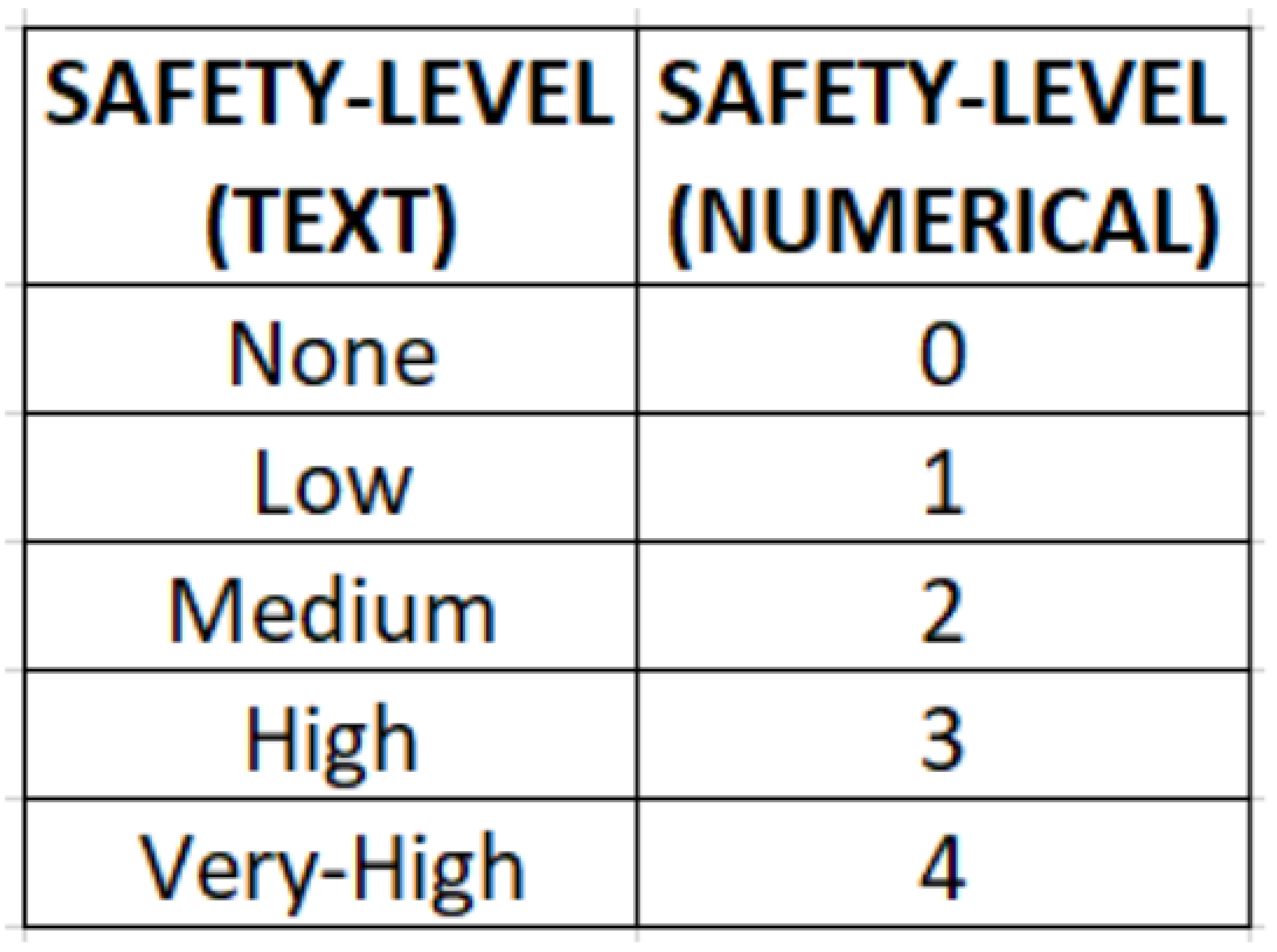
Encoding types :

Label Encoding

Difintion	the categorical data is converted into numerical data. Each category is assigned a numerical value.
Assumptions	• The number of categories is quite large as one-hot encoding can lead to high memory consumption. • When the order does not matter in categorical feature.
cases for not use it	can't be using LabelEncoder for categorical features
Example	in the next slide

Label Encoding

Example :



SAFETY-LEVEL (TEXT)	SAFETY-LEVEL (NUMERICAL)
None	0
Low	1
Medium	2
High	3
Very-High	4

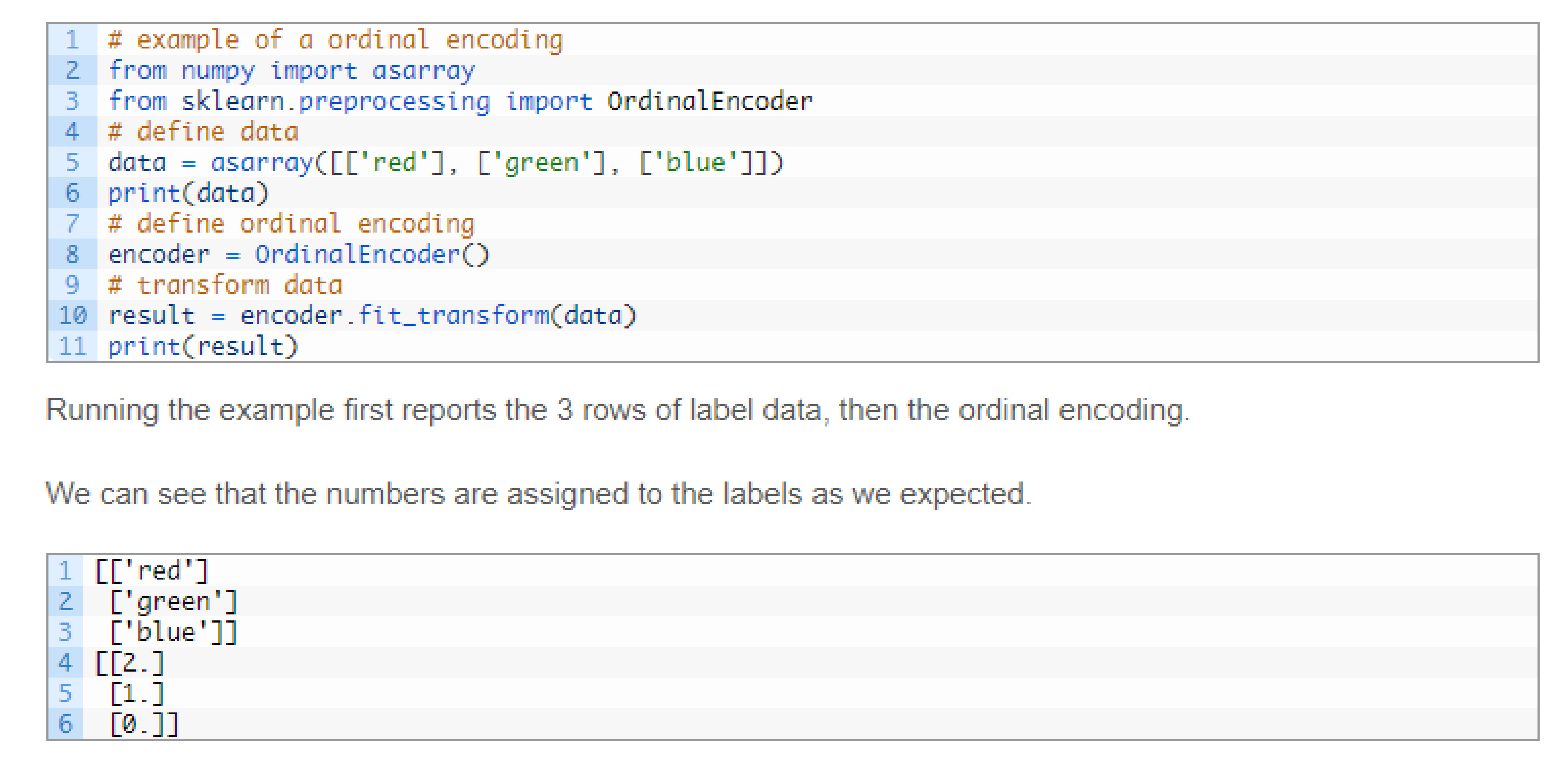
Encoding types :

Ordinal Encoding

Difintion	is used to encode categorical features into an ordinal numerical value (ordered set). This approach transforms categorical value into numerical value in ordered sets.
Assumptions	• when the variables in the data are ordinal
cases not use it	• For categorical variables, it imposes an ordinal relationship where no such relationship may exist.
Example	in the next slide

Ordinal Encoding

Example :



```
# example of a ordinal encoding
from numpy import asarray
from sklearn.preprocessing import OrdinalEncoder
# define data
data = asarray([['red'], ['green'], ['blue']])
print(data)
# define ordinal encoding
encoder = OrdinalEncoder()
# transform data
result = encoder.fit_transform(data)
print(result)
```

Running the example first reports the 3 rows of label data, then the ordinal encoding.

We can see that the numbers are assigned to the labels as we expected.

```
[['red']
 ['green']
 ['blue']]
[[2.]
 [1.]
 [0.]]
```

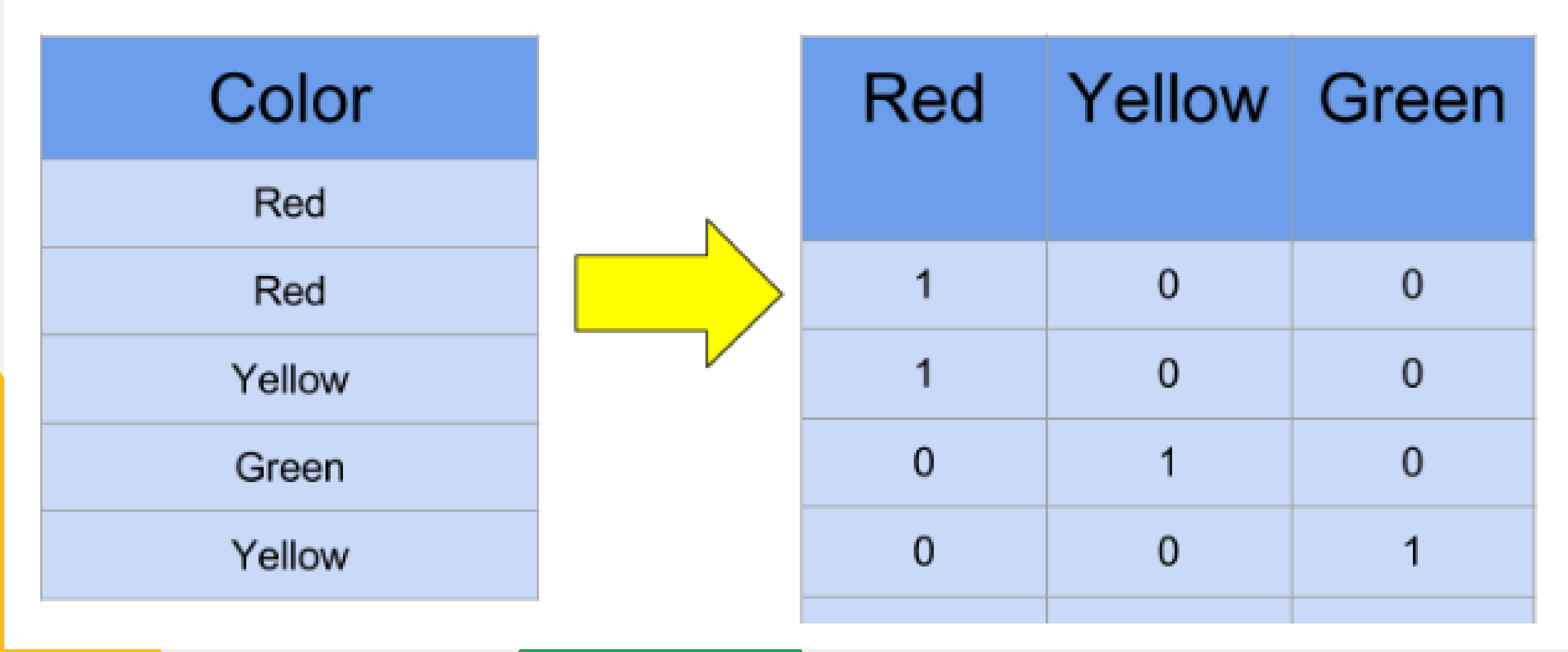
Encoding types :

One hot Encoding

Difintion	using a separate dummy variable for each category, and setting the value of the dummy variable to 1 if the observation belongs to that category and 0 otherwise.
Assumptions	• When the order does not matter in categorical features
cases not use it	• when the categories in the featuers is large
Example	in the next slide

One hot Encoding

Example :



Color
Red
Red
Yellow
Green
Yellow

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

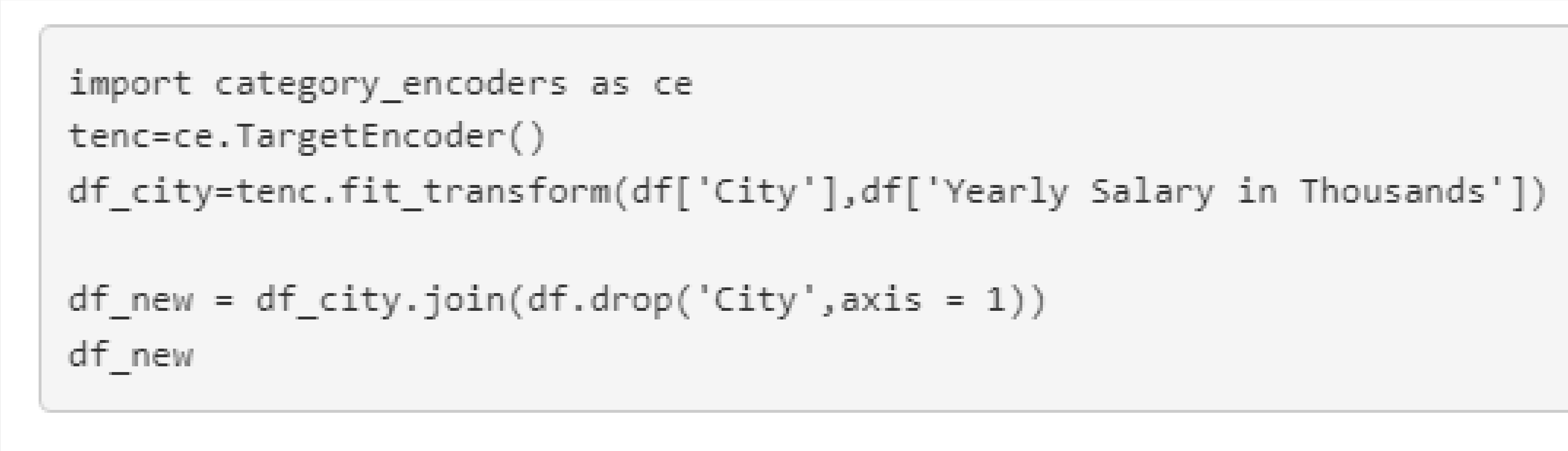
Encoding types :

Target Encoding

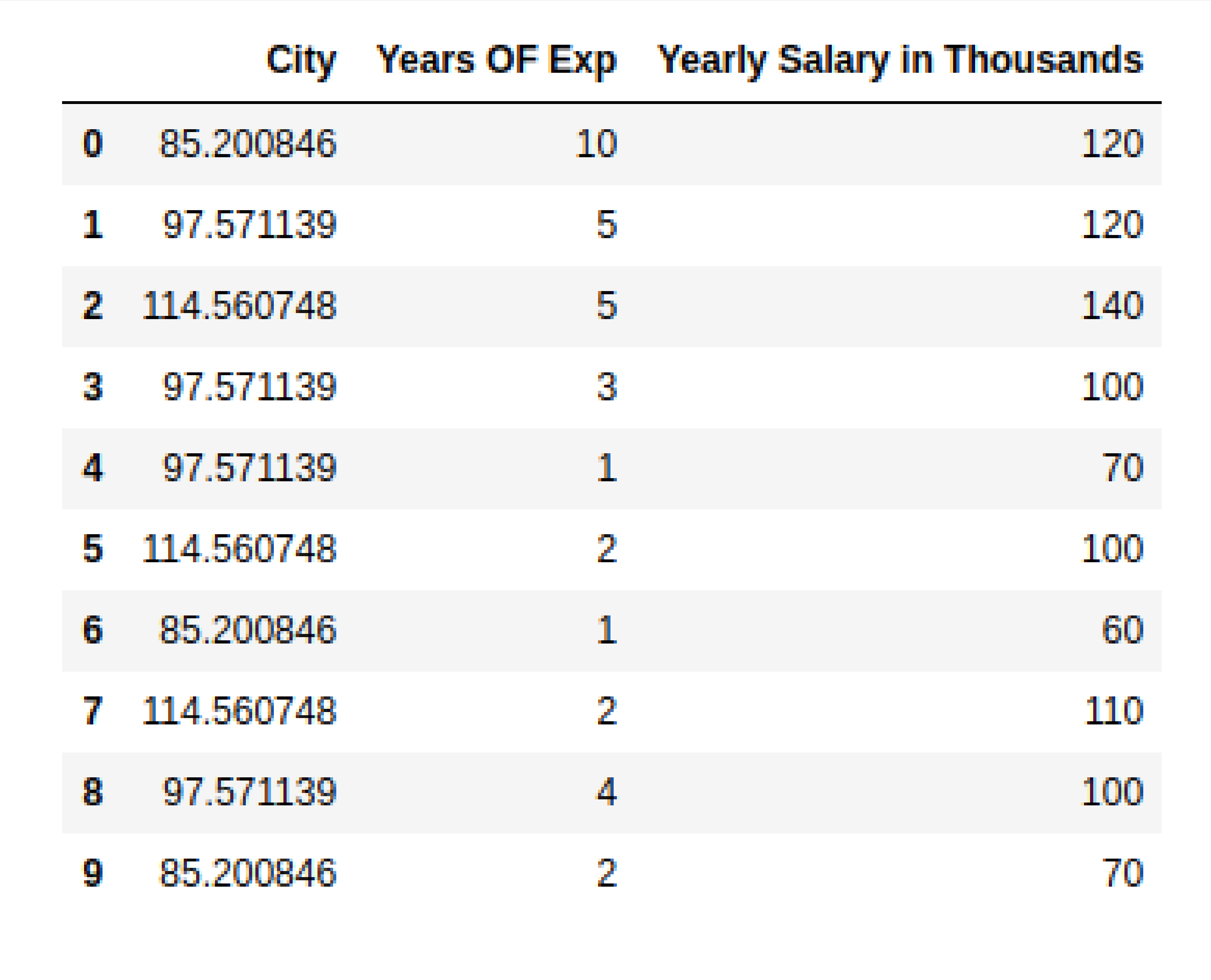
Difintion	encoding the categorical values of the features by using the target value. The idea behind this technique is that if the feature is a good predictor of the target, then its values should be closer to the target.
Assumptions	- When you have many categories, good to use target encoding over one-hot
cases not use it	- when there is a dependencies between diffrenet categorical
Exmaple	in the next slide

Target Encoding

Example :



```
import category_encoders as ce
tenc=ce.TargetEncoder()
df_city=tenc.fit_transform(df['City'],df['Yearly Salary in Thousands'])

df_new = df_city.join(df.drop('City',axis = 1))
df_new
```



	City	Years OF Exp	Yearly Salary in Thousands
0	85.200846	10	120
1	97.571139	5	120
2	114.560748	5	140
3	97.571139	3	100
4	97.571139	1	70
5	114.560748	2	100
6	85.200846	1	60
7	114.560748	2	110
8	97.571139	4	100
9	85.200846	2	70

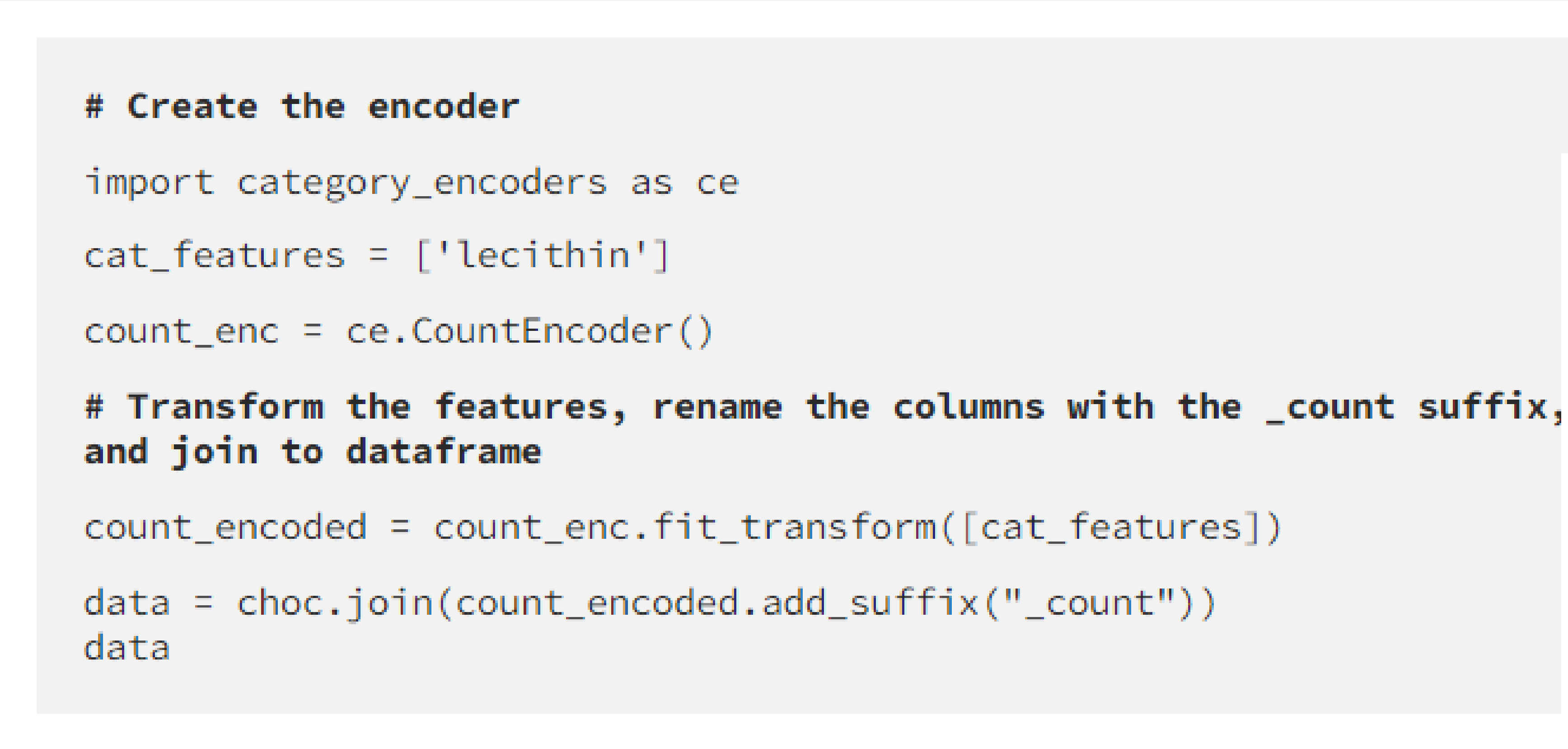
Encoding types :

Frequency / Count Encoding

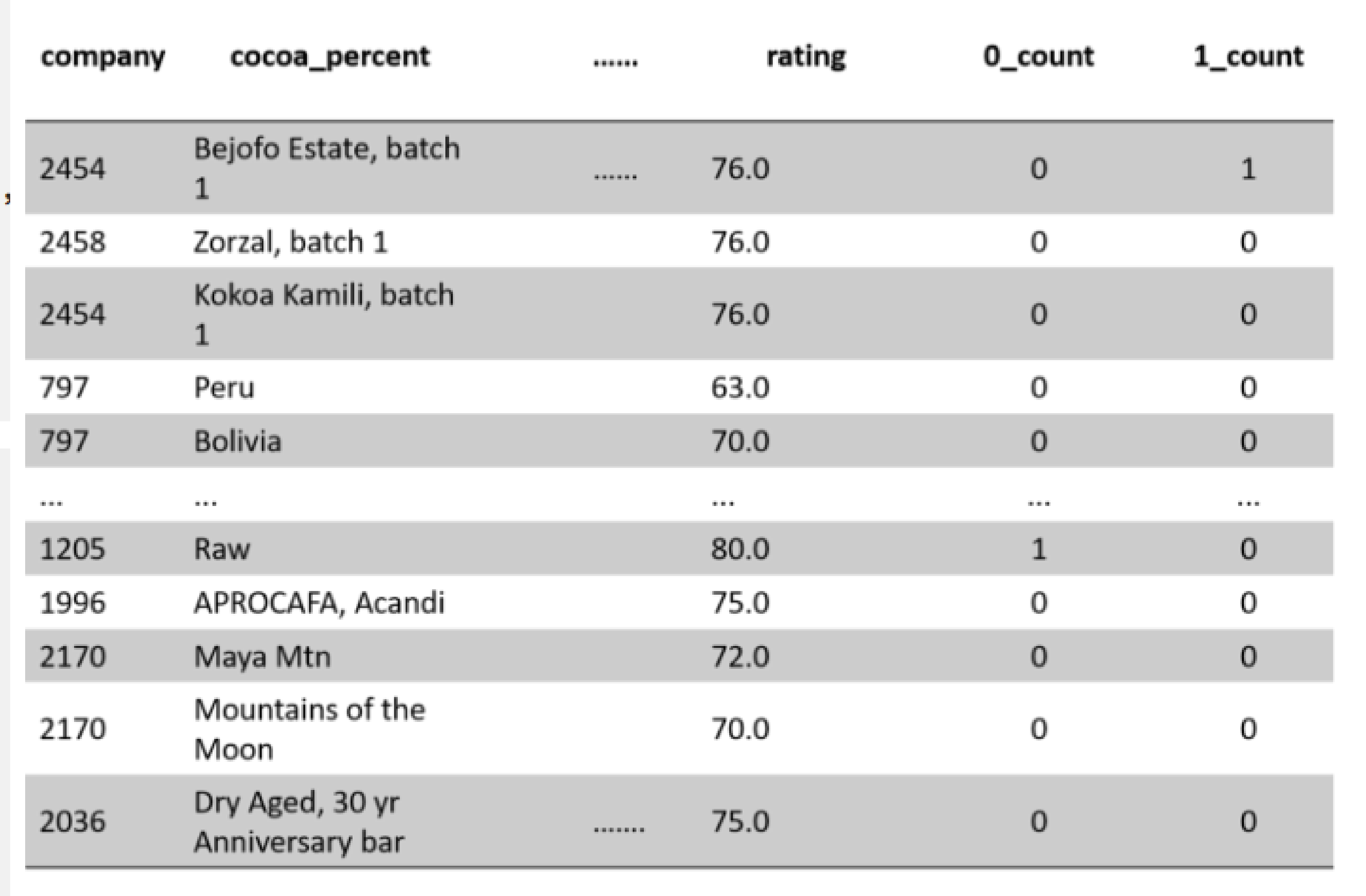
Difintion	is a way of representing categorical data using the count of the categories. Frequency encoding is simply a normalized version of count encoding.
Assumptions	Straightforward to implement.
cases not use it	if there are two different categories with the same amount of observations count
Example	in the next slide

Frequency / Count Encoding

Example :



```
# Create the encoder

import category_encoders as ce

cat_features = ['lecithin']

count_enc = ce.CountEncoder()

# Transform the features, rename the columns with the _count suffix,
and join to dataframe

count_encoded = count_enc.fit_transform([cat_features])

data = choc.join(count_encoded.add_suffix("_count"))
data
```



company	cocoa_percent	......	rating	0_count	1_count
2454	Bejofo Estate, batch 1	......	76.0	0	1
2458	Zorzal, batch 1		76.0	0	0
2454	Kokoa Kamili, batch 1		76.0	0	0
797	Peru		63.0	0	0
797	Bolivia		70.0	0	0
...	...		...	...	...
1205	Raw		80.0	1	0
1996	APROCAFA, Acandi		75.0	0	0
2170	Maya Mtn		72.0	0	0
2170	Mountains of the Moon		70.0	0	0
2036	Dry Aged, 30 yr Anniversary bar	....	75.0	0	0

Thank you :)

Day 19
17 nov,2022