# 21st Mar Assignment

March 23, 2023

## 1 Assignment 45

**Q1. What is the difference between Ordinal Encoding and Label Encoding? Provide an example of when you might choose one over the other.**

**Ans. Ordinal Encoding and Label Encoding are two methods of encoding categorical data into numerical values. The main difference between them is that Ordinal Encoding assigns a unique numerical value to each category based on their rank or order, while Label Encoding assigns a unique numerical value to each category arbitrarily.**

**For example, suppose you have a categorical feature called "size" with three categories: small, medium, and large. Ordinal Encoding would assign a value of 1 to small, 2 to medium, and 3 to large based on their order. Label Encoding, on the other hand, could assign a value of 1 to large, 2 to small, and 3 to medium.**

**One situation where you might choose Ordinal Encoding over Label Encoding is when there is an inherent order or hierarchy to the categories. For example, in a feature like education level with categories of "high school," "college," and "graduate," there is a clear order from lowest to highest level of education, and Ordinal Encoding would preserve this order in the numerical values.**

**In contrast, Label Encoding might be more appropriate when there is no natural ordering to the categories, such as in a feature like "color" with categories of "red," "blue," and "green." Here, arbitrary numerical values can be assigned to each category without affecting the meaning of the data.**

**Q2. Explain how Target Guided Ordinal Encoding works and provide an example of when you might use it in a machine learning project.**

**Ans. Target Guided Ordinal Encoding is a type of ordinal encoding technique that is used when encoding categorical variables with a large number of categories or levels. This technique takes into account the relationship between the categorical variable and the target variable, which is the variable we want to predict in our machine learning model.**

**The process involves calculating the mean of the target variable for each category or level of the categorical variable and then encoding the categories in ascending order of their mean target value. This ensures that the encoded values of the categorical variable are monotonically increasing with the mean target value.**

**For example, let's consider a dataset where we have a categorical variable called "city" with several categories, and we want to predict the salary of an employee based on this categorical variable. We can use Target Guided Ordinal Encoding to encode this variable by calculating the mean salary for each category of the city variable, and then encoding the categories in ascending order of their mean salary.**

**One of the benefits of using Target Guided Ordinal Encoding is that it can capture the relationship between the categorical variable and the target variable, which can improve the performance of our machine learning model. Another benefit is that it can reduce the dimensionality of the data by replacing the original categorical variable with a single numerical variable.**

**We might use Target Guided Ordinal Encoding in a machine learning project when dealing with high cardinality categorical variables, where One-Hot Encoding or Label Encoding can lead to the curse of dimensionality, and when the relationship between the categorical variable and the target variable is important for the task at hand. However, it's important to note that this technique should be used with caution, as it can introduce bias into the model if the relationship between the categorical variable and the target variable is spurious.**

**Q3. Define covariance and explain why it is important in statistical analysis. How is covariance calculated?**

**Ans. Covariance is a statistical measure that describes the relationship between two variables. It indicates how much two variables change together, and in what direction. Specifically, covariance measures the extent to which two variables are linearly related. Covariance is important in statistical analysis because it provides information about the degree of dependence between two variables. If the covariance between two variables is positive, it means that they tend to increase or decrease together. If the covariance is negative, it means that when one variable increases, the other tends to decrease. A covariance of zero indicates that there is no linear relationship between the two variables.**

**Covariance is calculated using the following formula:**

- cov(X, Y) = (1/n) * sum[(xi - mean(X)) * (yi - mean(Y))]

**where X and Y are the two variables being compared, xi and yi are individual data points, mean(X) and mean(Y) are the means of the two variables, and n is the total number of data points.**

**In essence, the formula computes the average of the product of the deviations of the two variables from their respective means. A positive covariance indicates a positive relationship between the variables, a negative covariance indicates a negative relationship, and a covariance of zero indicates no relationship.**

**Covariance is important in statistical analysis because it is a key component in calculating other measures, such as correlation coefficients and regression models. By examining the covariance between two variables, we can determine the direction and strength of their relationship, which can help us better understand the data and make informed decisions based on that understanding.**

**Q4. For a dataset with the following categorical variables: Color (red, green, blue), Size (small, medium, large), and Material (wood, metal, plastic), perform label encoding using Python's scikit-learn library. Show your code and explain the output.**

**Ans.**

[1]:
```
import pandas as pd
```

[11]:
```
#creating a Dataframe
data = {
    'Color': ['red', 'green', 'blue'],
    'Size': ['small', 'medium', 'large'],
    'Material': ['wood', 'metal', 'plastic']
}
df = pd.DataFrame(data)
```

[12]:
```
df
```

[12]:

| | Color | Size | Material |
|---|---|---|---|
| 0 | red | small | wood |
| 1 | green | medium | metal |
| 2 | blue | large | plastic |

[13]:
```
from sklearn.preprocessing import LabelEncoder
```

[14]:
```
encoder=LabelEncoder()
```

[15]:
```
df_encoded=df.apply(encoder.fit_transform)
```

[16]:
```
df_encoded
```

[16]:

| | Color | Size | Material |
|---|---|---|---|
| 0 | 2 | 2 | 2 |
| 1 | 1 | 1 | 0 |
| 2 | 0 | 0 | 1 |

**Q5. Calculate the covariance matrix for the following variables in a dataset: Age, Income, and Education level. Interpret the results.**

**Ans.**

[17]:
```
import numpy as np

# Create a sample dataset with Age, Income, and Education level
data = np.array([[35, 50000, 16],
                 [45, 70000, 18],
                 [30, 30000, 12],
                 [40, 60000, 15],
                 [50, 80000, 20]])
```

[18]:
```
data
```

[18]:
```
array([[   35, 50000,    16],
       [   45, 70000,    18],
       [   30, 30000,    12],
       [   40, 60000,    15],
       [   50, 80000,    20]])
```

[19]:
```
# Calculate the covariance matrix
covariance_matrix = np.cov(data.T)
```

[20]:
```
print(covariance_matrix)
```

```
[[6.25e+01 1.50e+05 2.25e+01]
 [1.50e+05 3.70e+08 5.55e+04]
 [2.25e+01 5.55e+04 9.20e+00]]
```

**Q6. You are working on a machine learning project with a dataset containing several categorical variables, including "Gender" (Male/Female), "Education Level" (High School/Bachelor's/Master's/PhD), and "Employment Status" (Unemployed/Part-Time/Full-Time). Which encoding method would you use for each variable, and why?**

**Ans. For the given categorical variables, here is my recommendation on which encoding method to use:**

1. Gender: Binary Encoding or One-Hot Encoding can be used. Since there are only two categories (Male and Female), Binary Encoding would be sufficient to encode this variable. In Binary Encoding, Male can be encoded as 0 and Female can be encoded as 1. Alternatively, One-Hot Encoding can also be used, where a new binary column is created for each category, indicating whether or not that category is present in the observation. However, in this case, One-Hot Encoding would be less efficient than Binary Encoding, as only one column is needed to represent the Gender variable.

2. Education Level: Ordinal Encoding can be used. Since there is a natural order to the categories (High School < Bachelor's < Master's < PhD), Ordinal Encoding would be a good

choice. In Ordinal Encoding, each category is assigned a numerical value based on its rank or order. For example, High School can be encoded as 1, Bachelor's as 2, Master's as 3, and PhD as 4. This encoding preserves the ranking of the categories while also reducing the number of columns needed to represent the variable.

3. Employment Status: One-Hot Encoding can be used. Since there are three categories (Unemployed, Part-Time, Full-Time), One-Hot Encoding would be a good choice. In One-Hot Encoding, a new binary column is created for each category, indicating whether or not that category is present in the observation. For example, if an observation is Full-Time, then the Full-Time column would be set to 1, and the Unemployed and Part-Time columns would be set to 0. This encoding allows for each category to be treated independently and avoids any issues with ranking or order.

**Q7. You are analyzing a dataset with two continuous variables, "Temperature" and "Humidity", and two categorical variables, "Weather Condition" (Sunny/Cloudy/Rainy) and "Wind Direction" (North/South/East/West). Calculate the covariance between each pair of variables and interpret the results.**

**Ans.**

[21]:
```
import pandas as pd
import numpy as np

# Create a sample dataset
data = pd.DataFrame({
    'Temperature': [25, 28, 30, 22, 24],
    'Humidity': [60, 65, 70, 55, 50],
    'Weather Condition': ['Sunny', 'Cloudy', 'Rainy', 'Sunny', 'Cloudy'],
    'Wind Direction': ['North', 'South', 'East', 'West', 'South']
})
```

[22]:
```
data
```

[22]:

| | Temperature | Humidity | Weather Condition | Wind Direction |
|---|---|---|---|---|
| 0 | 25 | 60 | Sunny | North |
| 1 | 28 | 65 | Cloudy | South |
| 2 | 30 | 70 | Rainy | East |
| 3 | 22 | 55 | Sunny | West |
| 4 | 24 | 50 | Cloudy | South |

[25]:
```
# Calculate covariance between Temperature and Humidity
cov1 = np.cov(data['Temperature'], data['Humidity'])[0][1]
print(f"Covariance between Temperature and Humidity: {cov1}")
```

```
Covariance between Temperature and Humidity: 22.5
```

[29]:
```
# Calculate covariance between Temperature and Weather Condition
temp_by_wc = pd.pivot_table(data, index='Temperature', columns='Weather␣
 ↪Condition', aggfunc=len, fill_value=0)
```

```
cov2 = temp_by_wc.cov().iloc[0][1]
print(f"Covariance between Temperature and Weather Condition: {cov2}")
```

```
Covariance between Temperature and Weather Condition: -0.1
```

[30]:
```
# Calculate covariance between Temperature and Wind Direction
temp_by_wd = pd.pivot_table(data, index='Temperature', columns='Wind␣
 ↪Direction', aggfunc=len, fill_value=0)
cov3 = temp_by_wd.cov().iloc[0][1]
print(f"Covariance between Temperature and Wind Direction: {cov3}")
```

```
Covariance between Temperature and Wind Direction: -0.05
```

[31]:
```
# Calculate covariance between Humidity and Weather Condition
hum_by_wc = pd.pivot_table(data, index='Humidity', columns='Weather Condition',␣
 ↪aggfunc=len, fill_value=0)
cov4 = hum_by_wc.cov().iloc[0][1]
print(f"Covariance between Humidity and Weather Condition: {cov4}")
```

```
Covariance between Humidity and Weather Condition: -0.1
```

[32]:
```
# Calculate covariance between Humidity and Wind Direction
hum_by_wd = pd.pivot_table(data, index='Humidity', columns='Wind Direction',␣
 ↪aggfunc=len, fill_value=0)
cov5 = hum_by_wd.cov().iloc[0][1]
print(f"Covariance between Humidity and Wind Direction: {cov5}")
```

```
Covariance between Humidity and Wind Direction: -0.05
```