feature-engineering-4-5

August 13, 2023

[ ]:
```
# Q1. What is the difference between Ordinal Encoding and Label Encoding?
# Provide an example of when you might choose one over the other.
```

Ordinal Encoding and Label Encoding are both techniques used to convert categorical data into numerical values, but they are applied under different circumstances and for different types of categorical data.

Ordinal Encoding: Ordinal encoding is used when there is a clear and meaningful order or hierarchy among the categories. In this encoding, categories are assigned integer values based on their relative order. For example, if you have a feature like "Education Level" with categories "High School," "Bachelor's," "Master's," and "Ph.D.," you can assign ordinal values such as 1, 2, 3, and 4 respectively.

Label Encoding: Label encoding is a simpler form of encoding that is used when there is no inherent order among the categories. In this technique, each category is assigned a unique integer value. For instance, if you have a feature like "Color" with categories "Red," "Green," and "Blue," you can assign integer values like 0, 1, and 2.

Summary:

Ordinal encoding is used when there is an order or hierarchy among the categories. Label encoding is used when categories don't have an inherent order and need to be represented as distinct numerical values.

[ ]:
```
# Q2. Explain how Target Guided Ordinal Encoding works and provide an example␣
↪of when you might use it in a machine learning project.
```

Target Guided Ordinal Encoding is a technique used to encode categorical variables based on the relationship between the categories and the target variable in a supervised machine learning setting. It's particularly useful when dealing with ordinal categorical variables, where the categories have a meaningful order.

Here's how Target Guided Ordinal Encoding works:

1 Calculate the Mean Target Value for Each Category:

For each category in the categorical variable, calculate the mean of the target variable for instances belonging to that category. This means computing the average of the target variable values for all instances in each category.

2 Order the Categories:

Order the categories based on their mean target values. Assign a lower encoded value to categories with lower mean target values and a higher encoded value to categories with higher mean target values. This ranking reflects the relationship between the categories and the target variable.

3 Replace Categories with Encoded Values:

Replace the original categorical values with their corresponding encoded values based on the order established in step 2.

This encoding technique uses the information from the target variable to create an ordinal relationship among the categories that mirrors their impact on the target.

4 Example Scenario:

Imagine you are working on a credit risk prediction project, and you have a categorical feature called "Income Group" with categories "Low," "Medium," and "High." The target variable is whether a customer defaults on a loan or not.

You can use Target Guided Ordinal Encoding to encode the "Income Group" feature as follows:

Calculate the mean default rate for each income group. Order the income groups based on their mean default rates (e.g., High > Medium > Low). Encode the income groups accordingly (High = 2, Medium = 1, Low = 0).

In this case, you're using the relationship between the income groups' default rates and the target variable to create an ordinal encoding that captures the impact of income groups on loan default. This encoding can help the model learn from the patterns in the data more effectively, potentially improving prediction accuracy.

[ ]:
```
# Q3. Define covariance and explain why it is important in statistical analysis.
↪ How is covariance calculated?
```

Covariance is a statistical measure that quantifies the degree to which two variables change together. It indicates the direction of the linear relationship between two variables and whether they tend to increase or decrease together. In other words, covariance measures how changes in one variable are associated with changes in another variable.

Mathematically, the covariance between two variables, X and Y, is calculated as follows:

$\text{Cov}(X, Y) = \Sigma[(X - \bar{X})(Y - \bar{Y})] / (n - 1)$

Where:

X and Y are individual data points of X and Y. $\bar{X}$ and $\bar{Y}$ are the means of X and Y, respectively. n is the number of data points.

The sign of the covariance indicates the direction of the relationship between the variables:

Positive covariance ($\text{Cov}(X, Y) > 0$) indicates that when one variable increases, the other tends to increase as well. Negative covariance ($\text{Cov}(X, Y) < 0$) indicates that when one variable increases,

the other tends to decrease.

Importance of Covariance in Statistical Analysis: Covariance plays a crucial role in statistical analysis for several reasons:

Linear Relationship: Covariance is a measure of the linear relationship between two variables. It helps identify whether the variables move in the same direction (positive covariance) or opposite directions (negative covariance).

Regression Analysis: Covariance is a key component in calculating regression coefficients and assessing the strength and direction of relationships between predictor and response variables.

[ ]:
```
# 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.
```

[1]:
```
from sklearn.preprocessing import LabelEncoder
import pandas as pd

# Sample dataset
data = {
    'Color': ['red', 'green', 'blue', 'green', 'red'],
    'Size': ['small', 'medium', 'large', 'medium', 'small'],
    'Material': ['wood', 'metal', 'plastic', 'wood', 'metal']
}

df = pd.DataFrame(data)

# Initialize LabelEncoder
label_encoder = LabelEncoder()

# Apply label encoding to each column
for column in df.columns:
    df[column + '_encoded'] = label_encoder.fit_transform(df[column])

print(df)
```

```
   Color    Size Material  Color_encoded  Size_encoded  Material_encoded
0    red   small     wood              2             2                 2
1  green  medium    metal              1             1                 0
2   blue   large  plastic              0             0                 1
3  green  medium     wood              1             1                 2
4    red   small    metal              2             2                 0
```

In the encoded columns, the categorical values have been replaced with their corresponding numerical labels. Each unique category within a column is assigned a unique integer value. It's important

to note that label encoding should be used when there's an inherent ordinal relationship among the categories or when the categories have no ordinal relationship and the algorithm can interpret the encoded values as meaningful numerical representations.

[ ]:
```
# Q5. Calculate the covariance matrix for the following variables in a dataset:
 ↪Age, Income, and Education level.


# Interpret the results
```

To calculate the covariance matrix for the variables Age, Income, and Education Level in a dataset, you would need the data points for each variable.

However, I can provide you with a general explanation of how to interpret the covariance matrix and what the results might mean.

The covariance matrix is a square matrix where each element represents the covariance between two variables. For three variables (Age, Income, and Education Level), the covariance matrix would look like this:

```
        Age           Income      Education Level
```

Age Cov(Age, Age) Cov(Age, Income) Cov(Age, Education) Income Cov(Income, Age) Cov(Income, Income) Cov(Income, Education) Education Cov(Education, Age) Cov(Education, Income) Cov(Education, Education)

Interpretation:

Diagonal Elements: The diagonal elements (e.g., Cov(Age, Age), Cov(Income, Income), Cov(Education, Education)) represent the variances of each variable. They indicate how much the variable deviates from its mean value. A larger diagonal element suggests higher variability.

Off-Diagonal Elements: The off-diagonal elements (e.g., Cov(Age, Income), Cov(Age, Education), Cov(Income, Education)) represent the covariances between pairs of variables. Covariance is a measure of how two variables change together. A positive covariance indicates that when one variable increases, the other tends to increase as well, and vice versa. A negative covariance indicates that when one variable increases, the other tends to decrease.

[2]:
```
import numpy as np

# Generate random data for Age, Income, and Education Level
np.random.seed(0)
num_samples = 100

age = np.random.randint(20, 65, num_samples)
income = np.random.randint(20000, 150000, num_samples)
education = np.random.randint(1, 5, num_samples)  # Assume 1 to 4 represents
 ↪different education levels

# Create a matrix with Age, Income, and Education Level
data_matrix = np.vstack((age, income, education)).T
```

```
# Calculate the covariance matrix
covariance_matrix = np.cov(data_matrix, rowvar=False)

print("Covariance Matrix:")
print(covariance_matrix)
```

```
Covariance Matrix:
[[ 1.85037980e+02 -2.14782885e+04 -1.72787879e+00]
 [-2.14782885e+04  1.30363652e+09  7.53099596e+02]
 [-1.72787879e+00  7.53099596e+02  1.27383838e+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?
```

5 Gender (Binary Categorical):

Since “Gender” has only two categories (Male/Female), you can use label encoding or a simple mapping to convert it into numerical values.

6 Assuming ‘gender’ column contains ‘Male’ and ‘Female’

data[‘gender’] = data[‘gender’].map({‘Male’: 0, ‘Female’: 1})

7 Education Level (Nominal Categorical with Ordinal Relationship):

“Education Level” is ordinal in nature, as it has a clear order: High School < Bachelor’s < Master’s < PhD. In this case, you can use ordinal encoding to preserve the order of categories.

8 Assuming ‘education_level’ column contains ‘High School’, ‘Bachelor’s’, ‘Master’s’, ‘PhD’

data[‘education_encoded’] = data[‘education_level’].map({‘High School’: 1, “Bachelor’s”: 2, “Master’s”: 3, ‘PhD’: 4})

9 Employment Status (Nominal Categorical without Ordinal Relationship):

"Employment Status" is nominal and lacks inherent order. For nominal categorical variables, one-hot encoding is often suitable. Each category is transformed into a separate binary column.

10 Using pandas' get_dummies() function for one-hot encoding

employment_dummies = pd.get_dummies(data['employment_status'], prefix='employment') data = pd.concat([data, employment_dummies], axis=1) data.drop('employment_status', axis=1, inplace=True)

[ ]:
```
# 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
```

[4]:
```
import numpy as np

# Generate random data for Temperature and Humidity
np.random.seed(0)
num_samples = 100

temperature = np.random.uniform(20, 35, num_samples)  # Temperature in Celsius
humidity = np.random.uniform(30, 80, num_samples)  # Humidity as a percentage

# Generate random data for Weather Condition and Wind Direction
weather_conditions = np.random.choice(['Sunny', 'Cloudy', 'Rainy'], num_samples)
wind_directions = np.random.choice(['North', 'South', 'East', 'West'],
 ↪num_samples)

# Create a matrix with Temperature, Humidity, Weather Condition, and Wind
 ↪Direction
data_matrix = np.vstack((temperature, humidity)).T

# Calculate the covariance matrix
covariance_matrix = np.cov(data_matrix, rowvar=False)

print("Covariance Matrix:")
print(covariance_matrix)
```

```
Covariance Matrix:
[[ 18.8904126   -3.99525551]
 [ -3.99525551 193.35317896]]
```

[ ]: