DATA SCALING & TRANSFORMATION

11 October 2023 21:03

scaling is a data preprocessing technique used to transform the values of features or variables in a dataset to a similar scale. The purpose is to ensure that all features contribute equally to the model and to avoid the domination of features with larger values.

Feature scaling becomes necessary when dealing with datasets containing features that have different ranges, units of measurement, or orders of magnitude. In such cases, the variation in feature values can lead to biased model performance or difficulties during the learning process.

There are several common techniques for feature scaling, including standardization, normalization, and min-max scaling. These methods adjust the feature values while preserving their relative relationships and distributions.

By applying feature scaling, the dataset's features can be transformed to a more consistent scale, making it easier to build accurate and effective machine learning models. Scaling facilitates meaningful comparisons between features, improves model convergence, and prevents certain features from overshadowing others based solely on their magnitude.

Scaling:

- Scaling involves transforming the values of features to a specific range. It maintains the shape of the original distribution while changing the scale.
- Scaling is particularly useful when features have different scales, and certain algorithms are sensitive to the magnitude of the features.
- Common scaling methods include Min-Max scaling and Standardization (Z-score scaling).

Min-Max Scaling:

- Min-Max scaling rescales the values to a specified range, typically between 0 and 1.
- It preserves the original distribution and ensures that the minimum value maps to 0 and the maximum value maps to 1.

m = (x -xmin) / (xmax -xmin)

- Python3

```
from sklearn.preprocessing import MinMaxScaler

# initialising the MinMaxScaler
```

```
scaler = MinMaxScaler(feature_range=(0, 1))

# Numerical columns
num_col_ = [col for col in X.columns if X[col].dtype != 'object']
x1 = X
# learning the statistical parameters for each of the data and transforming
x1[num_col_] = scaler.fit_transform(x1[num_col_])
x1.head()
```

Standardization (Z-score scaling):

- Standardization transforms the values to have a mean of 0 and a standard deviation of 1.
- It centers the data around the mean and scales it based on the standard deviation.
- Standardization makes the data more suitable for algorithms that assume a Gaussian distribution or require features to have zero mean and unit variance.
  `Z = (X - μ) / σ`
  Where,
- X = Data
- μ = Mean value of X
- σ = Standard deviation of X

Advantages of Data Cleaning in Machine Learning:

1. Improved model performance: Data cleaning helps improve the performance of the ML model by removing errors, inconsistencies, and irrelevant data, which can help the model to better learn from the data.
2. Increased accuracy: Data cleaning helps ensure that the data is accurate, consistent, and free of errors, which can help improve the accuracy of the ML model.
3. Better representation of the data: Data cleaning allows the data to be transformed into a format that better represents the underlying relationships and patterns in the data, making it easier for the ML model to learn from the data.
4. Improved data quality: Data cleaning helps to improve the quality of the data, making it more reliable and accurate. This ensures that the machine learning models are trained on high-quality data, which can lead to better predictions and outcomes.
5. Improved data security: Data cleaning can help to identify and remove sensitive or confidential information that could compromise data security. By eliminating this information, data cleaning can help to ensure that only the necessary and relevant data is used for machine learning.

Disadvantages of Data Cleaning in Machine Learning:

6. Time-consuming: Data cleaning can be a time-consuming task, especially for large and complex datasets.
7. Error-prone: Data cleaning can be error-prone, as it involves transforming and cleaning the data, which can result in the loss of important information or the introduction of new errors.
8. Limited understanding of the data: Data cleaning can lead to a limited understanding of the data, as the transformed data may not be representative of the underlying relationships and patterns in the data.
9. Data loss: Data cleaning can result in the loss of important information that may be

valuable for machine learning analysis. In some cases, data cleaning may result in the removal of data that appears to be irrelevant or inconsistent, but which may contain valuable insights or patterns.

10. Cost and resource-intensive: Data cleaning can be a resource-intensive process that requires significant time, effort, and expertise. It can also require the use of specialized software tools, which can add to the cost and complexity of data cleaning.
11. Overfitting: Overfitting occurs when a machine learning model is trained too closely on a particular dataset, resulting in poor performance when applied to new or different data. Data cleaning can inadvertently contribute to overfitting by removing too much data, leading to a loss of information that could be important for model training and performance.

DATA TRANSFORMATION

Robust Scaler
Robust Scaler are robust to outliers.It is used to scale the feature to median and quantiles Scaling using median and quantiles consists of substracting the median to all the observations, and then dividing by the interquantile difference. The interquantile difference is the difference between the 75th and 25th quantile:

IQR = 75th quantile - 25th quantile

X_scaled = (X - X.median) / IQR

Guassian Transformation
Some machine learning algorithms like linear and logistic assume that the features are normally distributed -Accuracy -Performance

- logarithmic transformation
- square root transformation
- exponential transformation (more general, you can use any exponent)
- boxcox transformation

logarithmic transformation

The log transformation is, arguably, the most popular among the different types of transformations used to transform skewed data to approximately conform to normality.

If the original data follows a log-normal distribution or approximately so, then the log-transformed data follows a normal or near normal distribution.

- Square root transformation

square root transformation is a mathematical operation applied to data to make it more suitable for analysis or modelling. It's a type of data transformation used to make the data more closely resemble a normal distribution or to stabilize the variance of the data. Here's how it

works in simple terms:

Square Root Transformation: For each data point in your dataset, you take the square root of that value. So, if you have a number like 25, the square root of 25 is 5, and that becomes the new value for that data point.
Example:
Suppose you have a dataset of the ages of a group of people: [9, 16, 25, 36, 49]. If you apply a square root transformation to this data, it would become: [3, 4, 5, 6, 7].

- Why Use Square Root Transformation:

Stabilizing Variance: In some statistical models, it's assumed that the variance of the data is constant. If your data has increasing variance with increasing values, a square root transformation can help stabilize it.

Normalization: It can help make the data more normally distributed, which is often an assumption in statistical tests and models. Normal distribution is a bell-shaped curve, and square root transformation can make your data more bell-shaped.

Interpretability: In some cases, taking the square root of values can make them more interpretable. For example, if you're dealing with areas, like square feet, the square root can transform them into lengths, which might be easier to understand.

- **exponential transformation**

An exponential transformation in data science involves raising data values to an exponent. This transformation is useful for handling data that grows or decays at a consistent rate. In simple terms, it makes the data grow or shrink much faster.
Exponential Transformation Example:

Imagine you have a dataset representing the population of a town over several years:

- Year 1: 1,000
- Year 2: 1,100
- Year 3: 1,210
- Year 4: 1,331
- Year 5: 1,464

If you apply an exponential transformation with an exponent of 2, it would look like this:

- Year 1: 1,000^2 = 1,000,000
- Year 2: 1,100^2 = 1,210,000
- Year 3: 1,210^2 = 1,464,100
- Year 4: 1,331^2 = 1,772,161
- Year 5: 1,464^2 = 2,144,896

In this example, the exponential transformation squared the values, making the growth more pronounced.

Why Use Exponential Transformation:

Highlighting Trends: Exponential transformations can help emphasize trends or patterns that might not be as apparent in the original data. For example, in finance, exponential growth is often used to model compound interest or investment returns.

Stabilizing Variance: It can help stabilize the variance in data, just like other transformations. If your data has a variance that increases or decreases with the magnitude of values, an exponential transformation can mitigate this effect.

Applying Linear Models: In some cases, you might want to use linear models to analyse data that exhibits exponential growth. Applying an exponential transformation can make the data more suitable for linear regression analysis.

boxcox transformation:

The Box-Cox transformation is a method developed by George Box and Sir David Cox to transform non-normal or skewed data into a more normally distributed form.
It involves applying a power function to the original data, which helps address issues of heteroscedasticity and improve the assumptions of normality and constant variance in statistical models.

Certainly! The Box-Cox transformation formula is Y(lambda) = (X^lambda - 1) / lambda.

In this formula, Y(lambda) represents the transformed data, X is the original data, and lambda is the transformation parameter.

By varying the value of lambda, different types of transformations can be applied to the data. The optimal lambda value is determined using statistical techniques like maximum likelihood estimation.

BENEFITS:
The Box-Cox transformation is valuable because it helps normalize skewed data and improves the assumptions required by many statistical techniques, such as linear regression. It can address issues of non-normality and heteroscedasticity, allowing for more accurate and reliable statistical modeling. By transforming the data, it helps meet the assumptions of normality and constant variance, enhancing the validity of the results.

EXAMPLE:
Let's say we're analyzing a dataset where the target variable is right-skewed, meaning it has a long tail on the right side. By applying the Box-Cox transformation to the target variable, we can bring it closer to a normal distribution, which can improve the performance of statistical models that assume normality.

This transformation can be particularly beneficial in fields like finance, economics, and social sciences, where data often deviates from normality.