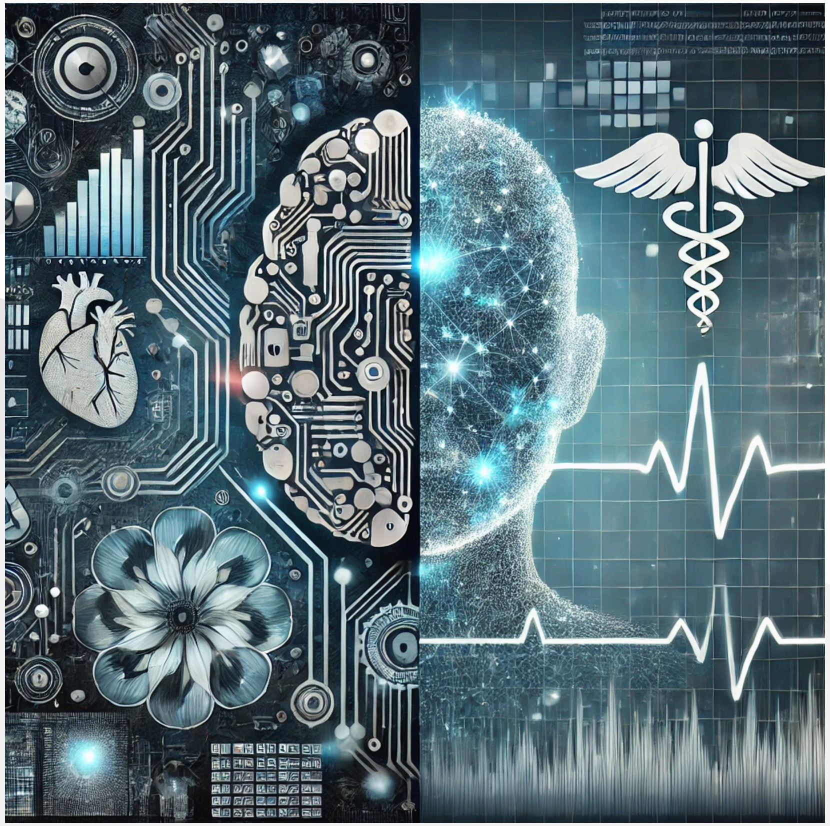

Feature Engineering

Basic Feature Engineering

Introduction

Effective features not only capture the key characteristics of the data but also align with the assumptions of the model. This often requires transforming the raw numeric data. Mastering these basic techniques is crucial, as they form the foundation of feature engineering across all data types.

Before diving into more complex data types such as text or images, let's start with the simplest type: **numeric data**.

Basic Feature Engineering Techniques for Numeric Data

To design features that improve model performance and generalization, it is important to understand three key elements:

- Magnitude
- Scale
- Distribution

Magnitude

The magnitude of a feature represents its absolute value without considering the sign (positive or negative). The first check for numeric data is to assess whether the magnitude of the values is important.

Do you need to know the exact magnitude, or is it sufficient to distinguish between positive and negative values? Sometimes, coarse granularity is all that’s necessary. This is especially true for automatically generated counts, such as daily website visits or the number of reviews for a restaurant.

Binning (quantization)

Binning is a technique where continuous numeric values are divided into discrete bins, which can be more manageable and provide insights by grouping similar data points together. This method helps in reducing the complexity of the model and often improves performance.

Fixed-width Binning

With fixed-width binning, each bin represents a specific numeric range. These ranges can either be custom-designed or automatically segmented, depending on the data and the modeling needs. The bins can be linearly scaled (equal-sized intervals) or exponentially scaled (wider bins for larger values).

Quantile Binning

In case of uneven data distribution, we can use an adaptive technique: quantile binning. Quantile binning positions bins based on the distribution of the data, ensuring that each bin contains roughly the same number of data points. This prevents empty bins and maintains a balanced representation of the data.

Log Transformation

Another powerful technique for handling large ranges in counts is the logarithmic transform. The log function compresses the range of large numbers and expands smaller ones. As values grow larger, the rate of increase of their logarithm slows down, making it a useful way to tame extreme values. This can be particularly effective when raw counts span several orders of magnitude, as the log transform creates a more manageable range for the model to work with.

Power Transforms: Generalization of the Log Transform

The log transform is a specific example of a family of transformations known as power transforms.
In statistical terms, these are variance-stabilizing transformations.
Power transforms change the distribution of the variable so that the variance is no longer dependent on the mean.

Power Transforms: Generalization of the Log Transform

A simple generalization of both the square root transform and the log transform is known as the **Box-Cox transform**:

$$X^{(\lambda)} = \begin{cases} \frac{X^\lambda - 1}{\lambda} & \text{for } \lambda \neq 0, \\ \log X & \text{for } \lambda = 0. \end{cases}$$

The formula $ { {(x ^ \lambda - 1) } / \lambda } $ is chosen so that $ x ^ {(\lambda) } $ is continuous as $ \lambda $ tends to zero and monotone increasing with respect to $ x $ for any $ \lambda $.

Power Transforms: Generalization of the Log Transform

The power parameter $ \lambda $ is estimated by a graphical technique or by the *Maximum-likelihood method*.

Scaling

Feature scaling and normalization are crucial preprocessing steps in machine learning. Many algorithms, especially those based on distances or gradients, perform better when the input features are on comparable scales. Without proper scaling, features with larger ranges can disproportionately influence the model.

Min-Max Scaling

Min-max scaling transforms the features to a fixed range, usually between 0 and 1. This is particularly useful for algorithms like k-nearest neighbors (KNN) and k-means clustering, which rely on distance metrics. However, it's sensitive to outliers because extreme values can distort the scaling.

Variance Scaling (Standardization)

Variance scaling (or standardization) transforms features so that they have a mean of 0 and a standard deviation of 1. This is useful for models like linear regression, logistic regression, and neural networks, where features are assumed to be normally distributed.

L_2 Normalization

L2 normalization divides each feature vector by its Euclidean norm, so that the resulting vectors all have a length of 1. This is particularly useful in models like SVM or KNN, where the direction of vectors is

Feature Selection

Feature selection techniques prune away nonuseful features in order to reduce the complexity of the resulting model.

> “ Given a set of d features, select a subset of size m that leads to the smallest classification error. ”

The end goal is a parsimonious model that is quicker to compute, with little or no degradation in predictive accuracy.

> “ Feature selection is not about reducing training time but about reducing model scoring time. ”

Feature Selection

Once m has been decided (rule of thumb $N_{class}/m > 10$) choose the m most informative features keeping:

- Large distances between classes
- Small distances within class