International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 14, Issue 2 [February. 2025] PP: 109-114

Some applications of Normal distribution in Machine Learning algorithms

Tran Thi Bich Hoa
The University of Danang - Vietnam-Korea University of Information and Communication Technology, Danang, Vietnam.

Abstract:
Based on the theory of normal distribution and machine learning algorithms, the article highlights the applications of normal distribution in machine learning algorithms, especially providing a method to normalize input data using the Python programming language.
Keywords: *data normalization, machine learning, normal distribution, Python language*

Date of Submission: 14-02-2025 Date of acceptance: 28-02-2025

I. INTRODUCTION

In recent years, artificial intelligence (AI) has emerged as evidence of the 4th industrial revolution. Artificial intelligence has become a core component in high-tech systems. It has penetrated into almost all areas of life. Machine learning is a subset of artificial intelligence. It is a field of computer science, capable of self-learning based on input data without having to be specifically programmed. The developments of artificial intelligence have led to a high demand for human resources in data science, technology and related industries worldwide. Among them, there is a strong development in the applications of Natural Sciences, especially Mathematics in the field of technology. Probability and statistics is a branch of Mathematics, studying random phenomena. It can be said that probability theory is one of the most important theories of modern science and especially Machine Learning because most of Machine Learning algorithms are based on probability.

As AI continues to revolutionize industries across the globe, Python's role in data science and machine learning is becoming increasingly important. Its salient features such as rich library support, versatility in handling various tasks, strong community support, and readability make it an ideal choice for developers and researchers working on AI projects.

The research objective of the paper is to study the applications of normal distribution in machine learning algorithms and use Python to check the normality of data to make data processing decisions before putting it into the Machine Learning model.

II. CONTENTS

2.1. Theoretical basis

2.1.1. Definition of one-dimensional normal distribution

A continuous random variable X is said to have a normal distribution with two parameters μ, σ, denoted $X \sim N(\mu, \sigma)$ or $N(\mu, \sigma^2)$ if its probability density function has the form:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}; \forall x \in \mathbb{R}$$

The probability distribution function has the form:

$$F(x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt$$

In particular, if $\mu=0$ and $\sigma=1$ then X is said to be a normal distribution. Then the distribution function is denoted and defined as:

$$\mathrm{F(x)} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt$$

The graph of the probability density function of a one-dimensional random variable has the form:

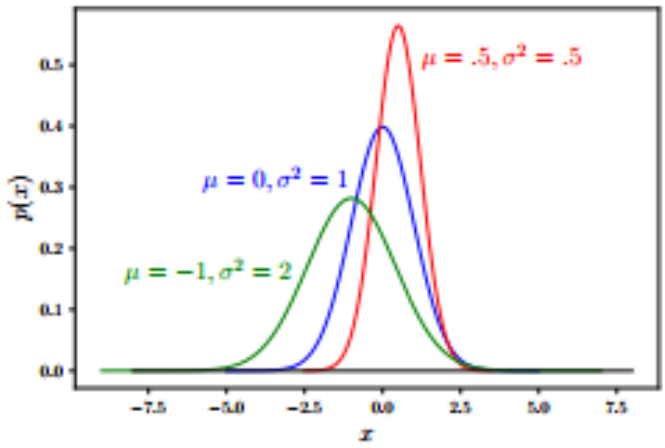


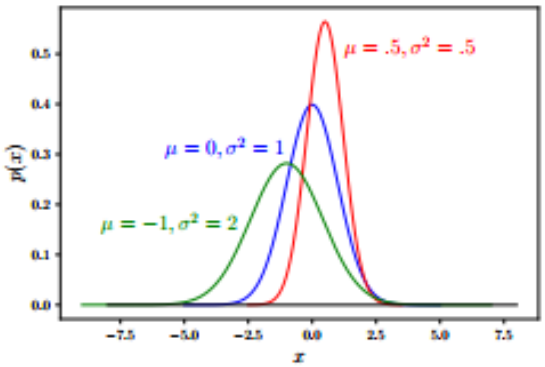
Figure 1

Thus, the graph of the density function has a symmetrical bell shape with the highest frequency located in the middle and the lower frequencies located on both sides. This is a special feature of the normal distribution and based on this feature, people can recognize and standardize data by relying on the histogram.

2.1.2. Definition of multivariate normal distribution

This distribution is a general case of an n-dimensional random variable. There are two parameters describing this distribution, the expectation vector $\mu \in \mathbb{R}^n$ and the covariance matrix sai $\Sigma \in \mathbb{S}^n$,, which is a positive definite symmetric matrix.

Let X be an n-dimensional continuous random variable, X has a normal distribution with parameters μ, Σ, denoted as X~ N(μ, Σ) if the probability density function has the form:



$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \quad ; \forall x \in \mathbb{R}^n.$$

Where, |Σ| is the determinant of the covariance matrix Σ.

The graph of the probability density function of a two-dimensional random variable is a curved surface of the following form:

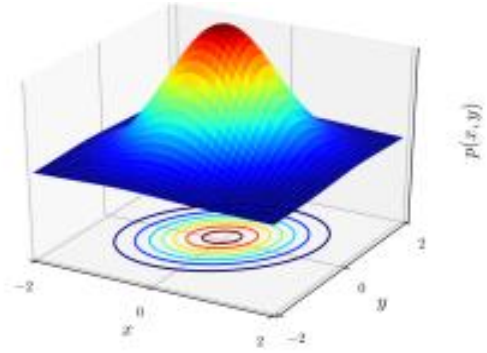


Figure 2

If we cut this surface along planes parallel to the base surface, we get concentric ellipses.

2.1.3. Some properties of normal distribution

- The Gaussian distribution must be symmetric around its mean with same probability density on both sides of mean.
- The sum of many independent, identically distributed random variables converges to a Gaussian distribution.
- When you estimate the mean and variance of a Gaussian distribution from a set of data, the maximum likelihood estimators provide the most accurate estimates compared to other distributions.
- In linear transformations, if X follows a Gaussian distribution, then aX+b also follows a Gaussian distribution for constants a and b. This property makes the Gaussian distribution robust and convenient for modeling various real-world phenomena that involve linear transformations.
- In multiple dimensions, the Gaussian distribution extends naturally. It describes how multiple variables can be jointly Gaussian, meaning that any linear combination of these variables also follows a Gaussian distribution. This property is valuable for modeling complex systems.
- Empirical rule in normal distribution.

The empirical rule is a convenient and quick estimate of the data range based on the mean and standard deviation of a data set that follows a normal distribution.

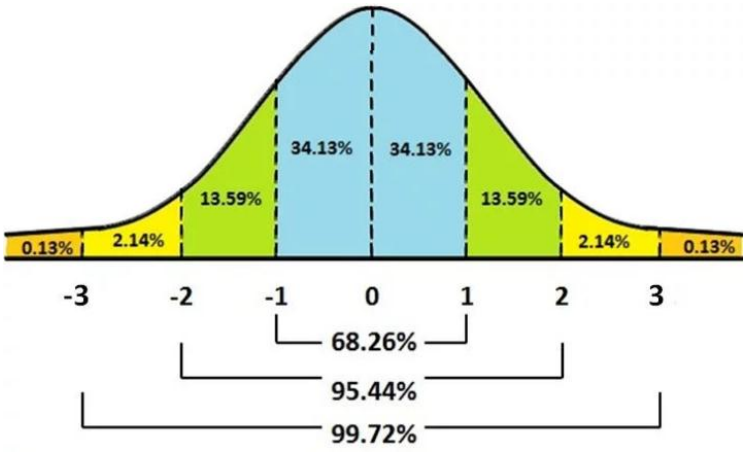


Figure 3

It states that:
68.26% of the data will fall within 1 standard deviation of the mean ($\mu \pm 1\sigma$)
95.44% of the data will fall within 2 standard deviations of the mean ($\mu \pm 2\sigma$)
99.7% of the data will fall within 3 standard deviations of the mean ($\mu \pm 3\sigma$)
This rule helps check for outliers and is useful when determining the normality of any distribution.

2.2. Some Machine Learning Algorithms

Machine learning is a field of artificial intelligence that focuses on developing algorithms and models to help computers automatically learn from data without being directly programmed. Instead of specifying each step to solve a specific problem, machine learning allows computers to automatically learn from input data and improve their performance over time. Machine learning has many applications in life and industry, from data analysis to decision automation and manufacturing process optimization. Businesses are increasingly recognizing the importance of machine learning. Because it allows them to gain deeper insights into customer trends and behavior, as well as build business models and develop new products. Many large corporations, such as Facebook, Google and Uber, have made machine learning an integral part of their business strategy. In fact, machine learning is becoming an important factor helping companies outperform their competitors.

There are many ways to classify machine learning, usually divided into 4 types: Supervised learning, Unsupervised learning, Semi-supervised learning and Reinforcement learning. Depending on the type of data scientists want to predict, they will choose to use the appropriate type of algorithm.

2.2.1. Linear Regression Algorithm

This is a popular machine learning algorithm for predicting continuous values. This algorithm determines the linear relationship between input and output variables by finding the best straight line to fit the data. This algorithm can be used to predict stock prices, house prices, or any other continuous value.

2.2.2. Logistic Regression Algorithm

This is a classification algorithm used to predict the probability of an event occurring or not occurring. This algorithm uses a logistic function to calculate the probability and then generate classification predictions. Logistic Regression is used in classification problems such as classifying emails as spam or not, or classifying potential customers.

2.2.3. Decision Tree Algorithm

Decision Tree is a machine learning algorithm used to predict outputs based on decisions made on a decision tree. This algorithm separates data into branches based on the questions and decisions made. Decision Tree is used to classify potential customers, predict the success of a new product, or detect diseases.

2.2.4. Random Forest Algorithm

This is a classification and prediction machine learning algorithm used for large data sets. This algorithm generates many random decision trees and combines them to create a final prediction. Random Forest is often used to classify customers, predict stock prices, or predict the success of a new product.

2.2.5. Naive Bayes Algorithm (Naïve Bayes)

Naive Bayes is a classification algorithm based on probability theory. This algorithm calculates the probability of a sample belonging to a certain class and then chooses the class with the highest probability as the prediction. Naive Bayes is used in text classification, image classification, or news classification problems.

2.2.6. Support Vector Machine (SVM) Algorithm

This is a classification and regression machine learning algorithm used to find the best boundary between data classes. This algorithm finds the boundary so that the distance from the data points to the boundary is the largest. SVM is often used in text classification, image classification, or customer classification problems.

2.2.7. K-Nearest Neighbors (KNN) Algorithm

This is a classification and prediction algorithm that uses Euclidean distance to find the k data points that are closest to a new data point. The algorithm then uses the labels of the nearest points to predict the label of the new point. KNN is used in customer classification, product classification, or image classification problems.

2.2.8. Artificial Neural Network (ANN) Algorithm

ANN is an artificial neural network used to learn nonlinear and nonlinear models. This algorithm is structured as layers of neurons connected together to create a machine learning model. ANN is often used in speech recognition, handwriting recognition, or face recognition problems.

2.2.9. Gradient Boosting Algorithm

This is a machine learning optimization method used to create a predictive model by combining many weak models together. This algorithm creates a predictive model by iterating the optimization process on the weak models. Gradient Boosting is often used in problems such as predicting stock prices, or predicting the success of a new product.

2.3. Applications of normal distribution in machine learning algorithms

- Likelihood Modeling: In algorithms, such as linear regression, logistic regression, and Gaussian mixture models, it is often assumed that the observed data is generated from a Gaussian distribution. It simplifies the model and allows for efficient parameter estimation.
- Bayesian Inference: In Bayesian machine learning, the Gaussian distribution is commonly used as a prior distribution over model parameters. This prior distribution reflects about the parameters before observing any data and is updated to a posterior distribution using Bayes' theorem.
- Clustering: Gaussian mixture models (GMMs) can model complex data distributions and are often used in image segmentation and data compression.
- Anomaly Detection: Gaussian distribution is often used in anomaly detection algorithms, where the goal is to identify rare events or outliers in the data. Anomalies are detected based on the likelihood of the data under the Gaussian distribution.
- Dimensionality Reduction: Principal Component Analysis (PCA), it finds the directions of maximum variance in the data, which correspond to the principal components.
- Kernel Methods: Gaussian kernel is commonly used in kernelized machine learning algorithms, such as Support Vector Machines (SVMs) and Gaussian Processes (GPs), to define the similarity between data points.

2.4. Using Python to normalize input data

There are many ways to normalize data in Machine Learning, here we are interested in the "whitening" technique. This is a technique to normalize data, bringing the data set to a normal distribution with a mean of 0 and a standard deviation of 1. This technique is often used in algorithms such as linear regression, logistic regression,… when the input data values have different value domains.

The normalization formula is as follows:

$$x' = \frac{x - \bar{x}}{\delta}$$

Where $\bar{x}$ and σ are the expectation and standard deviation of that component over the entire training data, respectively.

Standardization assumes that the observations have a Gaussian (bell-shaped) distribution. If the data distribution is not normally distributed, then standardization will not be effective.

When standardize the data, we need to calculate the mean and standard deviation based on the observations.

The standardization formula:

```
PLAINTEXT
y = (x - mean) / standard_deviation
```

In which mean is calculated as follows:

```
PLAINTEXT
mean = sum(x) / count(x)
```

Calculate standard deviation:

```
standard_deviation = sqrt( sum( (x - mean)^2 ) / count(x))
```

Assuming the mean is 10, the standard deviation is 5, the value 20.7 will be standardized as follows:

```
PLAINTEXT
y = (x - mean) / standard_deviation
y = (20.7 - 10) / 5
y = (10.7) / 5
y = 2.14
```

We can standardize the data using the scikit-learn library with StandardScaler:

```
# demonstrate data standardization with sklearn
from sklearn.preprocessing import StandardScaler
# load data
data = ...
# create scaler
scaler = StandardScaler()
# fit scaler on data
scaler.fit(data)
# apply transform
standardized = scaler.transform(data)
# inverse transform
inverse = scaler.inverse_transform(standardized)
```

Or use the fit_transform function as follows:

```
# demonstrate data standardization with sklearn
from sklearn.preprocessing import StandardScaler
# load data
data = ...
# create scaler
scaler = StandardScaler()
# fit and transform in one step
standardized = scaler.fit_transform(data)
# inverse transform
inverse = scaler.inverse_transform(standardized)
```

III. CONCLUSION

The normal distribution or Gaussian distribution is an important concept in statistics and is the foundation of Machine Learning. A data scientist needs to understand this distribution when working with linear models as well as how to use Python to normalize data. In the scope of this article, I cannot give more illustrative examples of the application of the normal distribution, I hope in the next research we will provide normalization models for more specific problems.

REFERENCES

[1]. Amin Jullanvari (2023), Machine Learning with Python Theory and Implementation, Springer Publisher.
[2]. Charu C.Aggawal (2024), Probability and Statistics for Machine Learning, Springer Publisher.
[3]. Dang, T.H. (1997), Introduction to Probability Theory and Applications, Educational Publisher, Vietnam.
[4]. Dao, H.H (2010), Probability and statistics , Hanoi National University Publisher, VietNam.
[5]. Le Cun, Y., Kanter, I., and Sola, S. A.(1991), Eigenvalues of covariance matrices: Application to neural network learning, Physical Review Letters, 66(18): 2396-2399.
[6]. Nield.T (2022), Essential Math For Data Science, Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472
[7]. Nilsson, P., Li, J. (2015), Teaching and Learning of Probability. In: Cho, S. (eds) The Proceedings of the 12th International Congress on Mathematical Education. Springer, Cham. https://doi.org/10.1007/978-3-319-12688-3_36.