Vinay Pawar TYITB098

Assignment No. 4

Problem Statement:
Implement a normal distribution in Python and visualize it for a mean of 100, standard deviation of 4, and dataset size of 100,000.

Objective:
Generate a dataset with 100,000 data points, centered around a specified mean and standard deviation, ensuring it follows a normal distribution.
Visualize the data distribution using histograms and probability density plots.
Analyze the dataset to verify that its characteristics align with the theoretical expectations of a normal distribution, particularly for the specified mean and standard deviation.

Prerequisites:

1. Basic Python Programming
2. Statistics: Understanding the normal distribution, including mean, standard deviation, and probability density function (PDF).
3. Python Libraries:
    - **NumPy**: Essential for numerical computations, especially generating random data from various distributions.
    - **Matplotlib**: A versatile library for generating static, interactive, and animated visualizations.
    - **Seaborn**: Built on Matplotlib, Seaborn provides an easy-to-use, high-level interface for creating attractive statistical plots.

Theory:

Normal Distribution:

The normal distribution is a continuous probability distribution, often represented as a symmetrical bell curve centered on the mean. Its primary characteristics are:

- **Mean (μ)**: The central value of the distribution.
- **Standard Deviation (σ)**: Indicates the spread or variability within the distribution. A larger standard deviation results in a broader distribution.

The PDF of the normal distribution is given by the following formula:

f(x∣μ,σ)=1σ2πe−(x−μ)22σ2f(x|μ,σ) = \frac{1}{σ \sqrt{2π}} e^{ - \frac{(x - μ)^2}{2σ^2} }f(x∣μ,σ)=σ2π1e−2σ2(x−μ)2

Where:

- $f(x \mid \mu,\sigma)f(x|\mu,\sigma)f(x \mid \mu,\sigma)$ is the probability density at a given value xxx.
- $\mu\mu\mu$ is the mean of the distribution.
- $\sigma\sigma\sigma$ is the standard deviation.
- eee is Euler's number (~2.718).
- $\pi\pi\pi$ is Pi (~3.14159).

68-95-99.7 Rule:
An important property of the normal distribution is that approximately:

- 68% of values lie within one standard deviation of the mean,
- 95% within two standard deviations, and
- 99.7% within three standard deviations.

Importance of Normal Distribution:

The normal distribution is foundational in statistics due to the Central Limit Theorem, which suggests that the sum or average of a large number of independent, identically distributed variables will approximate a normal distribution, regardless of their original distribution. This makes the normal distribution applicable to many scenarios, such as:

- Heights within a population
- Measurement errors
- Stock market returns

Algorithm:

1. **Set Up the Environment**: Import necessary Python libraries - NumPy, Matplotlib, and Seaborn.
2. **Generate Data**: Use the numpy.random.normal() function to generate a dataset of 100,000 points, specifying the mean and standard deviation to ensure a normal distribution.
3. **Plot Data**:
    - Use Seaborn to create a histogram with KDE (Kernel Density Estimation) overlay to visualize the PDF.
    - Use Matplotlib to enhance the clarity and aesthetics of the plot.
4. **Compute Statistics**: Calculate and display the mean and standard deviation of the generated dataset to verify that they match the specified parameters.
5. **Interpret Results**: Analyze the generated plot for its shape and spread to confirm it follows a normal distribution.

References:

1. NumPy Documentation: Official documentation for functions and data handling in NumPy.

2. Matplotlib Documentation: Official reference for plotting and visualization using Matplotlib.
3. Seaborn Documentation: Official documentation for statistical visualization in Seaborn.
4. Wikipedia on Normal Distribution: In-depth theoretical understanding of the normal distribution.

Conclusion:

In this project, Python was used to generate a dataset with a normal distribution, mean of 100, and standard deviation of 4. The dataset was visualized through a histogram and KDE plot, illustrating a bell-shaped curve typical of a normal distribution. Calculated values for the mean and standard deviation aligned closely with the specified parameters, verifying the accuracy of the simulation. This exercise demonstrates Python's capabilities in statistical data generation, analysis, and visualization, proving highly effective for understanding the characteristics of the normal distribution.