**Fitting Data to Probability Distributions Using Maximum Likelihood Estimate (MLE)**

**Abstract**

DistFit is a Python package design to streamline the process of fitting data to a diverse array of probability distributions through the application of the Maximum Likelihood Estimate (MLE) method followed by Chi-Square goodness of fit. The package is adept at handling both discrete and continuous distributions, offering users a flexible and powerful tool for conducting in-depth statistical analyses. The implemented distributions include Bernoulli, Binomial, Geometric, Poisson, Uniform, Exponential, Normal, Weibull, and Gamma. This paper serves as a comprehensive guide to DistFit, delving into details about each distribution type it accommodates the underlying parameters utilized in the fitting process. The MLE approach is employed to estimate the parameters of these distributions, enabling users to gain insights into the underlying nature of their datasets.

1. **Introduction**

At the core of DistFit lies the MLE methodology, a robust statistical technique employed to optimize the parameters of probability distributions by maximizing the likelihood of the observed data. This principled approach allows users to obtain precise estimates for distribution parameters, facilitating a profound understanding of the intrinsic characteristics of their datasets.

The package caters to a broad spectrum of statistical needs, providing users with the capability to model discrete phenomena, such as binary outcomes, counts, and event occurrences, as well as continuous variables, including time intervals and measurements. DistFit not only equips users with the means to fit their data accurately but also empowers them with insightful visualizations, enabling the exploration of how well the chosen distribution aligns with the observed data.

This paper endeavors to guide users through the functionalities of DistFit, offering an in-depth exploration of each supported distribution type. From the Bernoulli distribution, modeling binary events, to the intricate parameters governing the Weibull and Gamma distributions for continuous data, DistFit is designed to cater to a diverse range of statistical scenarios. This paper aims to empower users to make informed decisions about the most suitable distribution for their specific datasets.

In essence, DistFit emerges as an indispensable tool for researchers, data scientists, and statisticians seeking a comprehensive and user-friendly solution for probability distribution fitting in Python. Through the intricate details provided in this paper, users can navigate the package with confidence, harnessing the power of DistFit to uncover hidden patterns and gain valuable insights into the underlying nature of their data.

1. **Methodology**

DistFit is organized into three distinct modules: datagen, distfit, and gof. These modules collectively form DistFit, providing a comprehensive suite for data generation, distribution fitting, and goodness-of-fit analysis. Additionally, an examples file complements DistFit's functionality, demonstrating real-world applications of the datagen, distfit, and gof modules. This file serves as a practical guide, illustrating how to use DistFit to analyze different datasets and distribution types. The examples showcase the versatility and effectiveness of DistFit in fitting distributions to data and assessing the goodness of fit. The subsequent sections will delve into the specific details of each module, elucidating their functionalities and contributions to the overarching capabilities of DistFit.

* 1. **datagen**

the datagen module within DistFit is dedicated to synthetic data generation based on specified probability distributions. This module provides a convenient way for users to create artificial datasets, mimicking the characteristics of various probability distributions. The primary purpose of the datagen module is to facilitate testing and experimentation with DistFit's distribution fitting capabilities.

* + 1. **Datagen**

The Datagen class initializes with parameters specifying the type of distribution, the desired number of rows (data points), and distribution-specific parameters. It supports a variety of distribution types, including normal, geometric, binomial, Poisson, exponential, gamma, Weibull, uniform, and Bernoulli.

Initialization Parameters are:

* dist\_type: Type of probability distribution (e.g., 'normal', 'geometric', 'binomial').
* row\_count: Number of rows (data points) to be generated.
* par: Distribution-specific parameters. The format of parameters varies based on the distribution type.

Method: data\_generation

The data\_generation method generates synthetic data based on the specified distribution type and parameters. It utilizes NumPy's random number generation functions for each distribution type.

* 1. **distfit**

The methodology implemented in the Fitting class of the DistFit Python package revolves around fitting data to various probability distributions using the Maximum Likelihood Estimate (MLE) method. The class encompasses both discrete and continuous distributions, offering a comprehensive toolkit for statistical analysis. The key components and steps of the methodology are Initialization, Guessing Distributions and multiple different types of distributions and the plot method to visualize the fitted distributions against the data histogram.

* + 1. **Initialization**

The class is initialized with a given dataset (data), which can be provided as a pandas DataFrame or a pandas Series. If the input is not a DataFrame, it is converted to the correct format.

Essential statistics about the dataset are computed during initialization, such as mean (mu), standard deviation (sigma), size (size), minimum value (data\_min), maximum value (data\_max), and the original data itself.

* + 1. **Guessing Distributions**

The ‘guess\_distributions’ method determines the possible distributions based on the data type (discrete or continuous) and characteristics of the dataset.

For discrete distributions, it checks for specific conditions to identify Bernoulli distribution or selects from a predefined list of discrete distributions.

For continuous distributions, it considers all specified continuous distributions, adjusting the list based on the range of the dataset.

The methodology provides a systematic and versatile approach to fitting data to a range of distributions, empowering users with tools for exploratory data analysis and statistical modeling. The visualizations aid in understanding how well the fitted distributions align with the observed data.

* 1. **Gof**

The gof (Goodness of Fit) module in DistFit is crucial for assessing how well a chosen probability distribution fits observed data. It employs chi-squared test to evaluate the statistical significance of the difference between observed and expected frequencies. This module aids in validating the appropriateness of the selected distribution for a given dataset. To streamline this process, the "Gof" class, when used in conjunction with the "Distfit" class, offers a powerful tool for evaluating the fit of data to a specified distribution. This article provides a comprehensive guide on leveraging these classes for effective goodness-of-fit analysis.

The "gof" module is designed to assess the goodness of fit for a given dataset to a specified distribution. Following are the key components of this module:

* Initialization: Initializes the Gof class with the distribution type (dist\_type) and its parameters (par). The class automatically determines the number of estimated parameters (s) based on the distribution type.
* Frequency Calculation: Calculates the frequency of the observed data within specified bin edges. This step is crucial for subsequent goodness-of-fit analysis.
* Binning Optimization: Optimizes bins by combining adjacent bins if the expected frequency is less than 5. It ensures a minimum of 3 bins even when frequencies are low.
* Goodness of Fit Test: Conducts a goodness-of-fit test using the chi-squared statistic. It generates expected data based on the specified distribution, calculates observed and expected frequencies, optimizes bins, and performs the chi-squared test.

The effectiveness of the goodness-of-fit analysis is amplified when used in conjunction with the "distfit" class. The "distfit" class, as previously discussed, efficiently fits data to a variety of probability distributions using the Maximum Likelihood Estimate (MLE) method.

* 1. **Supported Distribution Types**

DistFit is a useful Python package designed for efficiently fitting data to a variety of probability distributions using the Maximum Likelihood Estimate (MLE) method. The package supports both discrete and continuous distributions, providing users with a versatile tool for statistical analysis. Different datasets may exhibit varying characteristics, influencing the choice of an appropriate distribution.

In this section, we introduce the key distribution types supported by DistFit, each serving a unique purpose in modeling and understanding different types of datasets.

**Will Add codes, results, and plots for each distribution types when example file is finalized**

* + 1. **Continuous Distributions**
       1. **Uniform Distribution**

The Uniform distribution models outcomes with equal likelihood over a specified range. DistFit estimates the lower ('a') and upper ('b') bounds through the MLE method. This type of distribution is useful when each outcome within a range is equally likely, as seen in scenarios like random number generation.

* + 1. **Exponential Distribution**

The Exponential distribution models the time between events in a Poisson process. DistFit estimates the rate parameter, '1/rate,' through the MLE method. It commonly applied in scenarios where the focus is on the time between events, such as the time between arrivals in a queue.

* + 1. **Normal Distribution**

The Normal distribution is a versatile distribution modeling a wide range of phenomena. DistFit estimates the mean ('mu') and standard deviation ('sigma') through the MLE method. It widely used in scenarios where data distribution is symmetric and follows a bell-shaped curve, like human height.

* + 1. **Weibull Distribution**

The Weibull distribution models reliability and life data. DistFit estimates the scale ('alpha') and shape ('beta') parameters through the MLE method. Its applicable can be in reliability engineering to model time until failure, with the ability to represent different failure patterns.

* + 1. **Gamma Distribution**

The Gamma distribution is a versatile distribution that generalizes exponential, chi-squared, and Erlang distributions. The lookup method is a utility for linear interpolation based on a look-up table. It is used in the gamma\_fit method to find the scale parameter (alfa) from a precomputed table based on the sample mean. DistFit estimates the scale ('alpha') and shape ('beta') parameters through the MLE method. It is useful in scenarios requiring the modeling of waiting times, sums of exponential random variables, and various other statistical applications.

* 1. **Discrete Distributions**
     1. **Bernoulli Distribution**

The Bernoulli distribution models binary outcomes, such as success or failure in a single trial. DistFit estimates the probability of success, denoted as 'p,' through the MLE method. This distribution is particularly useful for scenarios where there are only two possible outcomes. It is commonly employed in scenarios like coin flips or determining success/failure in a single trial.

* + 1. **Binomial Distribution**

The Binomial distribution extends the Bernoulli distribution to multiple independent trials. It models the number of successes in a fixed number of trials, denoted by 'n'. DistFit estimates the probability of success, 'p', through MLE, and 'n' is user-specified. It is Applicable in scenarios where there are a fixed number of independent trials, such as the number of successful attempts in a series of experiments.

* + 1. **Geometric Distribution**

The Geometric distribution models the number of Bernoulli trials needed to achieve the first success. It is suitable for scenarios where the interest lies in the number of trials required for a specific outcome. DistFit estimates the probability of success, 'p', through MLE. It is Useful in situations where the focus is on the number of trials until a particular event occurs for the first time.

* + 1. **Poisson Distribution**

The Poisson distribution models the number of events occurring within fixed intervals. DistFit estimates the rate parameter, '\_lambda', through the MLE method. It is Applicable when dealing with count data in scenarios like the number of emails received in an hour.

1. **Challenges and Future Updates**

While the DistFit Python package offers a robust methodology for fitting data to various probability distributions, like any software project, it may encounter challenges that is worth to state in this article. Some potential future updates can help mitigate the challenges associated with the DistFit project. Here are some challenges that might be addressed through future enhancements to the project:

* **Distribution Selection**: Determining the appropriate probability distribution for a given dataset can be subjective and complex. Providing additional guidance or automated methods for users to assist in selecting suitable distributions can be helpful.
* **Handling Large Datasets**: Processing large datasets may lead to performance issues or memory constraints. For future updates, we can plan to optimize algorithms and explore the integration of parallel processing techniques. These enhancements aim to significantly improve the package's performance, particularly when dealing with larger datasets. The optimization efforts will focus on refining computational efficiency, and parallel processing strategies will be investigated to leverage the capabilities of multi-core systems, ensuring faster and more scalable data fitting processes.
* **Model Interpretability**: Interpreting the results of fitted models, especially for complex distributions, can be challenging for non-expert users. Adding more examples, and visualizations can aid users in interpreting the results of the fitted models.
* **User Interface Design**: If the package is intended for a broad audience, creating an intuitive and user-friendly interface might be challenging. We can plan to develop a user-friendly interface, possibly through a graphical user interface (GUI) or interactive visualization tools, to make the package accessible to a wider audience.
* **Maintenance and Updates**: Keeping the package up-to-date with the latest versions of dependencies and addressing issues reported by users can be challenging. We will Establish a maintenance plan, encourage user feedback, and actively address reported issues to ensure the package remains reliable and relevant.
* **Documentation and Educational Resources**: Users may face challenges understanding the concepts of probability distributions and how to use the package effectively. We will improve our work with provide more documentation, tutorials, and educational resources to assist users in understanding both the statistical concepts and the practical usage of the package.

1. **Conclusion**

The methodology provides a systematic and versatile approach to fitting data to a range of distributions, empowering users with tools for exploratory data analysis and statistical modeling. The visualizations aid in understanding how well the fitted distributions align with the observed data. DistFit emerges not only as a distribution-fitting tool but as a comprehensive guide to the intricate world of probability distributions. By navigating through each distribution type and their parameters, users can harness the full potential of DistFit, transforming data analysis into a seamless and insightful journey. Armed with this knowledge, users can confidently employ DistFit to decipher the hidden stories within their datasets, fostering a deeper understanding of their data's inherent characteristics.