Exploratory Data Analysis
Assignment 2 : MCAR TEST

Group Number : Group 12

Group Members : Preet Shah 202411053
Neerav Sharma 202312028

https://github.com/Preet28/EDA-SEM1/tree/main

Missing Data and Its Challenges

Missing data is a prevalent problem in datasets that can arise from a number of sources, including insufficient data collection or inaccessible information. Missing data can distort results, lower the precision of statistical models, and provide biased conclusions, thus it's critical to handle it correctly. Choosing the appropriate imputation technique requires an understanding of the various forms of missing data processes, including Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR).

- **MCAR**: Missing data is entirely random and unrelated to both observed and unobserved values. Little's MCAR test helps check this mechanism, where a p-value above 0.05 suggests randomness.
- **MAR**: Missing data depends on observed values but not on the missing values themselves.
- **MNAR**: Missing data depends on the missing values, creating a more complex challenge.

The missingpy Library

The `missingpy` library is specifically designed for machine learning tasks where data imputation is necessary. It provides several imputation techniques, including:

- **MissForest Imputation**: An iterative method using Random Forests to predict and impute missing values for both continuous and categorical data.
- **KNN Imputation**: Uses the similarity between data points to fill missing values by considering their nearest neighbors.
- **Iterative Imputer**: A multivariate approach where each feature with missing values is modeled iteratively using other observed features.

If `missingpy` is not feasible due to installation or environment constraints, alternatives like the `fancyimpute` package can be useful.

Missing Completely at Random (MCAR)

MCAR is the most desirable type of missing data because it does not introduce bias into the dataset. When data is MCAR, it implies that the missing values occur randomly, and any statistical analysis using available data remains valid. To determine whether data is MCAR, **Little's MCAR Test** is widely used.

Little’s MCAR Test

Little’s MCAR Test is a statistical tool to assess if missing data is truly random. It compares the patterns of missing data with what would be expected if the data were MCAR.

- **Hypotheses**:
    - Null Hypothesis (H0): The data is MCAR.
    - Alternative Hypothesis (H1): The data is not MCAR (i.e., MAR or MNAR).
- **Test Statistic**: The test calculates a statistic by comparing the observed and expected patterns of missing data.
- **P-Value Interpretation**:
    - p-value > 0.05: Data is likely MCAR.
    - p-value ≤ 0.05: Data is not MCAR, indicating the possibility of MAR or MNAR.

KNN Imputation

K-Nearest Neighbors (KNN) Imputation is an effective technique to address missing data by identifying the most similar data points (neighbors) to estimate missing values. The algorithm calculates distances between data points based on observed values, and the missing data is filled in using the nearest neighbors' values.

```
from fancyimpute import KNN
knn_filled = KNN(k=2).fit_transform(df)
```

- **K=2**: In this case, the missing values are imputed based on the 2 nearest neighbors.
- **fit_transform()**: The method fits the KNN model to the data and imputes the missing values.

KNN is useful for both continuous and categorical data, but it may struggle with high-dimensional data or very large datasets.

Iterative Imputation

Iterative Imputation is a more advanced method where missing values in each feature are predicted using other features in the dataset. The process is iterative, improving the accuracy of imputations with each pass through the data.

```
from sklearn.experimental import IterativeImputer
iterative_filled = IterativeImputer().fit_transform(df)
```

- **IterativeImputer()**: This creates an imputer that fills in missing values iteratively. It models missing data by predicting each feature based on other observed variables.
- **fit_transform()**: Applies the imputer to the data, resulting in a fully imputed dataset.

Iterative Imputation is particularly useful when the features are highly correlated.

Missing Forest Imputation Method

MissForest is a robust imputation method that uses Random Forests to predict and fill missing data. It works iteratively and can handle both numerical and categorical data, making it a versatile tool for complex datasets.

```
from missingpy import MissForest
missing_forest_filled = MissForest().fit_transform(df)
```

- **MissForest**: Trains Random Forest models on observed data to predict missing values, refining the imputation iteratively.
- **fit_transform()**: Fills in the missing data using predictions from the Random Forest model.

This method works well when there are complex interactions between variables, and it does not assume any specific data distribution.

Also,

The ***chi2 object in scipy.stats represents the chi-squared distribution***, which is a continuous probability distribution. It arises in statistics when testing the goodness of fit or comparing categorical data.