Dhirubhai Ambani Institute of Information and Communication Technology

Exploratory Data Analysis, 2024

MCAR Test

Group Number : 22

Group Members :
Bhavya Boda (202203067)
Krishil Jayswal (202203040)
Aniket Pandey (202411001)

Course Instructor:
Gopinath Panda

MCAR Test

1. Introduction

MCAR (Missing Completely at Random) refers to a situation in data analysis where the missing data is unrelated to any observed or unobserved values in the dataset. In other words, the probability that a data point is missing is independent of both the variable itself and other variables in the dataset.

For example, a weighing scale that ran out of batteries or a weather station that is randomly unable to record temperatures due to occasional sensor malfunctions. In both the cases, failure of the weighing scale or the sensor has nothing to do with the weight of the person or the temperature of the place or any other environmental factors.The missing data occurs completely at random and is unrelated to the data itself, hence they are examples of MCAR.

To check for MCAR in our dataset, we can employ **Little's MCAR Test**. It is a statistical test used to assess whether the missing data in a dataset are missing completely at random or if there is a systematic pattern to the missingness. The test is named after **Donald R. Little** , who introduced it.

2. Why Little's MCAR Test?

Little's test helps assess whether applying simpler methods likes imputation techniques or list-wise deletion would lead to biased or unbiased results. By checking the randomness of missingness, it allows analysts to make informed decisions about whether these methods are appropriate, or if more advanced techniques like multiple imputation is required for handling data that isn't MCAR. This makes Little's MCAR test a critical step before choosing how to handle missing data.

3. How Little's MCAR test works:

Little's test assesses whether the pattern of missing data follows the MCAR assumption. It does this by comparing the distribution of observed data in groups based on missing values across variables. Then, it uses a **chi-squared** test statistic to compare the observed data to the expected values under the assumption of MCAR. A p-value of **less than α (generally 0.05)** is usually interpreted as being that the missing data is not MCAR.

3.1 Steps of Little's MCAR Test

1. **Formulate Hypotheses**
    - **Null Hypothesis (H_o)**: The missing data is MCAR, i.e., there is no systematic pattern in the missingness.
    - **Alternative Hypothesis (H_1)**: The missing data is not MCAR, i.e., the missingness follows a systematic pattern related to the data.
2. **Group Missing Data Patterns**
    - **Identify missing data patterns**: For each variable with missing data, group the dataset based on which values are missing. This creates "patterns" of missingness that will be analyzed in the test.
    - **Missingness Indicator:**: Create an indicator matrix R, where each element R_{ij} is 1 if the value in the dataset is missing or 0 if the value is observed.
3. **Calculate Group Statistics**
    - Compute the matrix product $R^T * R$, where R^T is the transpose of the R matrix.
    - This results in a square matrix where each cell represents the count of observations sharing specific combinations of missingness patterns. The diagonal elements of this matrix represent the number of observations with each individual missingness pattern.

4. **Compute the Test Statistic:**

   - **Chi-Squared Statistic**: Little's MCAR test uses a chi-squared test statistic to compare the observed data to the expected values under the assumption of MCAR.

   - The test statistic is computed as:

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected} \tag{1}$$

     where the differences between observed and expected values for each group are squared, normalized by the expected value, and summed across groups.

5. **Degrees of Freedom**

   - **Calculate Degrees of Freedom (df)**: The degrees of freedom for the test depend on the number of variables and the patterns of missingness. Specifically:

$$df = (n-1) \cdot m \tag{2}$$

     where n is the number of groups and m is the number of variables with missing data.

6. **P-Value Calculation**

   - **Determine the p-value**: The p-value is computed based on the chi-squared statistic and the degrees of freedom. This value indicates the likelihood of observing the test statistic if the null hypothesis (MCAR) were true.

7. **Interpret the Results**

   - **If the p-value is high(typically > 0.05)**: This means there is no evidence to suggest that the data is not MCAR. In this case, the data can be treated as MCAR, and simpler methods like list-wise deletion or mean imputation can be used without introducing bias.

   - **If the p-value is low(typically < 0.05)**: This suggests that the missingness is not completely random, and the data is likely Missing at Random (MAR) or Missing Not at Random (MNAR). In this case, more sophisticated methods, like multiple imputation or maximum likelihood, are needed to handle the missing data appropriately.

4. Observed Values

The observed values from the Little's MCAR test are as follows:

- **P-value:** 1.0
- **Chi-square statistic:** 12
- **Degrees of freedom:** 558,836

These observed values indicate that the data conforms to the assumption of Missing Completely At Random (MCAR). The high p-value suggests that we fail to reject the null hypothesis, which posits that the missing data are random and not dependent on any other variables in the dataset. As a result, there is no evidence of systematic bias in the missing data patterns.

5. Conclusion

Based on the results of the Little's MCAR test, with a p-value of 1.0, a chi-square statistic of 12, and 558,836 degrees of freedom, we conclude that the missing data in the dataset follows a pattern consistent with Missing Completely At Random (MCAR). This implies that the occurrence of missing values is independent of both observed and unobserved data, and there is no systematic relationship between the missingness and the dataset itself. Consequently, any analyses performed on this dataset would not be biased due to the missing data, and standard missing data handling techniques such as listwise deletion or imputation can be safely applied.

6. Implementation code for Little MCAR Test

```
# Algorithmn of Little MCAR test.

def little_mcar_test(data):
 """
 Perform Little's MCAR test on a pandas dataframe.
 Parameters:
 data (DataFrame): The dataset to test for MCAR.
 Returns:
 dict: A dictionary containing the chi-square statistic,
        degrees of freedom, and p-value.
 """

 n = len(data)

 groups = []

 for col in data.columns:
  mask = data[col].isnull()

 if mask.any():
  groups.append(mask.astype(int).values.reshape(-1, 1))

 if len(groups) == 0:
  raise ValueError("No missing data found.")

 r = np.concatenate(groups, axis=1)

 group_stats = r.T @ r

 m = len(groups)

 df = (n - 1) * m

 chi2_stat = group_stats.trace()

 p_value = chi2.sf(chi2_stat, df)

 return {"chi2_stat": chi2_stat, "degrees_of_freedom": df, "p_value": p_value}

# Function for datermining whether missing values are MCAR or not based on p value.

def is_mcar(p, alpha=0.05):
 if(p > alpha):
  return 'Your missing values are of MCAR type.'
 else:
  return 'Your missing values are not of MCAR type'
```

In the next section, we explore the missingpy package of python and see some treatment methods for the missing data.

missingpy Package

1 Introduction to `missingpy`

`missingpy` is a Python library designed for handling missing data, focusing on imputation techniques that work effectively with datasets containing both numerical and categorical features. It provides robust alternatives to traditional imputation methods, which can struggle with complex data patterns. By implementing advanced algorithms like K-Nearest Neighbors (KNN) and MissForest, `missingpy` allows for more accurate data recovery, enhancing the quality of statistical analysis, machine learning models, and overall data-driven insights.

While some standard imputation techniques (like filling missing values with the mean or median) are simple to apply, they often fail to preserve the relationships within the data. `missingpy` addresses this issue by using more sophisticated algorithms that respect these relationships, making it an ideal choice when working with datasets of various complexities.

2 Key Features of `missingpy`

- **Support for Numerical and Categorical Data**: `missingpy` supports datasets containing both numerical and categorical data, making it versatile for real-world applications.
- **Scikit-learn Compatible**: The library integrates easily with `scikit-learn` pipelines, making it an excellent choice for anyone working within the broader Python data science ecosystem.
- **Two Main Imputation Techniques**: `missingpy` provides two powerful methods for handling missing data: KNN imputation and MissForest.
- **Non-parametric Methods**: Both KNN and MissForest are non-parametric, meaning they do not make strict assumptions about the underlying distribution of the data.
- **Efficiency in Handling Large Datasets**: `missingpy` works efficiently with large datasets, suitable for real-world machine learning problems.

3 KNN Imputation

K-Nearest Neighbors (KNN) imputation is one of the popular techniques provided by `missingpy`. This algorithm works by identifying the 'k' nearest neighbors of a sample with missing data and imputing the missing values using the values from these neighbors.

3.1 How KNN Works

For each missing value, KNN computes the distance between the incomplete sample and other samples based on the available features. The 'k' closest samples are then averaged (for numerical data) or the most frequent category is used (for categorical data) to fill in the missing values.

3.2 Advantages

- Can handle both categorical and numerical data.
- Effective for small datasets with moderate missingness.

3.3 Disadvantages

- KNN imputation can become computationally expensive for larger datasets.
- Sensitive to the choice of 'k' and the distance metric.

4 MissForest Imputation

MissForest is another powerful imputation method in the `missingpy` library. It uses a Random Forest model to predict missing values.

4.1 How MissForest Works

1. The algorithm initializes missing values with the mean (for numerical) or mode (for categorical).
2. Iteratively updates these values by training a Random Forest model on observed data and imputing the missing values based on model predictions.
3. Continues this process until the difference between two iterations becomes negligible.

4.2 Advantages

- Can capture complex interactions in the data through Random Forest models.
- Works for both numerical and categorical data.
- Suitable for datasets with mixed-type data.

4.3 Disadvantages

- Computationally intensive for very large datasets.
- Requires careful tuning of hyperparameters.

5 Conclusion

The `missingpy` library provides essential tools for dealing with missing data, particularly through its KNN and MissForest imputation techniques. Both methods offer flexible, non-parametric solutions to imputation, enabling users to recover missing values while preserving relationships within the data. Whether dealing with a small or large dataset, `missingpy` is an effective solution for improving data quality and ensuring more accurate machine learning outcomes.

Github Link

Click the below link button to access the github link.

Access Link