**Data Science Post Block Assignment 3: Task B**

Difedile Rasenyalo [28294882] & Isabel de Waal [20805055]

1. **Introduction** 
   1. **Background**

Missing values are a common occurrence in datasets. It's not unusual to encounter datasets where a significant portion, sometimes up to half, of the entries are missing. Such datasets pose challenges for data analysis methods that require complete data. To address this issue, imputation methods are often employed to fill in the missing values. This paper investigates how different imputation methods impact the performance of classifiers when applied to imputed data. The study examines the effect of missing data imputation using two imputation methods, mean and K-NN imputation. The evaluation employed two popular classifiers, K-nearest-neighbour, and decision tree classifiers.

* 1. **Objectives**

**Task A:** The objective was to evaluate the effectiveness of a baseline imputation versus that of a Naïve Bayes imputation on the performance of classification models. For this study we also decided to look at the influence of the proportion of missing values on the effectiveness of the imputation method.

**Task B:** Building upon the groundwork of Task A, this task aims to use the K Nearest neighbours’ classifier to address the missing numerical value. Additionally, the effectiveness of this approach will be compared against a baseline numerical value imputation method using two distinct machine learning models.

1. **Methodology**

In this study, we leveraged Python as the primary tool for imputation, modelling, and evaluation. The methodology involved several key steps:

* 1. **Dataset Selection**

The first step entails selecting a publicly available dataset with documented research on handling missing numerical feature values. This dataset served as the cornerstone for assessing the effectiveness of the proposed imputation methods. For this study, we selected the ‘Adult’ dataset utilized in a study conducted by A. Farhangfara, L. Kurganb, and J. Dyc.

* 1. **Data Cleaning and pre-processing:**

The ‘Adult’ dataset was thoroughly evaluated for any possible data quality issues that may have an influence on model performance. The original dataset contained a mixture of numerical and categorical values. As this portion of the project is only concerned with numerical value imputation, only the numerical features were selected moving forward. The data was also in very different ranges; and because the distributions were non-Gaussian a min-max scaling approach was used to normalise them. After cleaning dataset was randomly divided into a 30:70 split forming the testing and training sets respectively. The testing set was kept constant throughout the experiment, serving as the testing set for all the training subsets. The training set was then copied six times, to form the basis to introduce the missing values.

* 1. **Introducing Missing Values:**

Missing data were randomly introduced into each of the copied datasets using the Missing Completely at Random (MCAR) mechanism. The missing values were added to instances across all features (including the target) at three different rates: 10%, 40%, and 70%. There were two copies of each missing value level.

* 1. **Imputation Approach**
* **Baseline Imputation:** Mean imputation approach was implemented to establish a reference for comparison.
* **K Nearest Neighbours Imputation:** The K Nearest Neighbour was employed to impute missing numerical feature values. The optimal number for k was selected via a cross validation step and evaluating silhouette scores.
  1. **Model Comparison:**

A Decision Tree classification algorithm and the K- Nearest Neighbours were employed for this study. The original training set and then each of the imputed training sets were separately trained and then tested with these two models. The optimal tree depth and the optimal number for k was determined using cross validation, and choosing the parameter that delivered the best results. The models' performance metrics, including accuracy, precision, recall, and F1-score, will be evaluated and compared.

1. **Results and Discussion**

***A***

Looking at the results left (Figure 1A&B) and comparing the differences in performance of the Classification Tree(A) and that of the k-Nearest neighbour. Overall, the same pattern is repeated. At a low level of missing values(10%), the mean imputation and the k-NN imputation perform very similarly. What is very interesting is how drastically the performance of the k-NN imputation method decreases as the proportion of missing values increase. Even more interesting is how ‘reliable’ the basic mean imputation was throughout the different proportions of missing values. This could purely be due to the structure of the data. When looking at the standard deviation of the different features, it was noted that the standard deviation was quite low [CHECK THIS], meaning that in this specific case, the mean approximation was best.

According to [*A comparison of* imputation methods using machine learning models ] the type of dataset is an important factor when deciding on a imputation method. They found that k-NN worked better for some datasets than others. Conversely [Comparison of Performance of Data Imputation Methods for Numeric Dataset] found that the type of dataset did not have an influence on imputation performance. They did however find that the k-NN imputation generally outperformed the mean imputation. These and other studies [ref numbers], have found that the proportion of missing data also does not influence the imputation performance. This is in contrast to what we found. This could be because…

***B***

1. **Flowchart Development**
2. **Conclusion**

In conclusion, the analysis of the

1. **References**

*Figure 1A&B:* Bar charts visualising the performance metrics (Accuracy and F1 scores) per level of imputation (10%, 40% and 70%) per imputation method (Control, Mean and k-NN) for the Classification Tree(A) and *k*-Nearest Neighbour(B) algorithms. Accuracy scores are indicated by the purple bars and the F1 scores are the blue bars.