**DATA AND INFORMATION QUALITY PROJECT REPORT**

PROJECT ID: 29

PROJECT NUMBER: 1

ASSIGNED DATASETS: abalone, users

STUDENTS:

Gabriele Ginestroni (10687747)

Tommaso Capacci (10654230)

ASSIGNED TASK: Clustering

1. **SETUP CHOICES**

**Chosen ML algorithms:**

Since the given datasets presented different characteristics we decided to use different algorithms for each of them.

In particular, for the “abalone” dataset we used:

* **K-Prototypes**: is a clustering method used to cluster mixed-type data (i.e. data that contains both categorical and numerical variables). This algorithm combines the K-Means algorithm for numerical variables and the K-Modes algorithm for categorical variables.

It works by first initializing the centroids for each cluster using either random sampling or a user-specified initialization method. Then, for each iteration, it assigns each data point to the cluster with the closest centroid. Next, it calculates the new centroid for each cluster by taking the mean of the numerical variables and the mode of the categorical variables of the data points assigned to that cluster. Finally, it repeats these steps until the centroids no longer change or a maximum number of iterations is reached.

* **DBSCAN (Density-Based Spatial Clustering of Applications with Noise):** is a density-based clustering algorithm that groups data points that are closely packed together (i.e. dense regions of data points) and separates data points that are sparsely located (i.e. less dense regions of data points).

DBSCAN works by defining a neighborhood around each data point, and then identifying clusters based on the density of data points in these neighborhoods. The algorithm takes as input two parameters: eps and minPts.: the eps parameter defines the radius of the neighborhood around each data point, and minPts defines the minimum number of data points required to form a dense region.

For the “users” one, instead, we opted for:

* **K-modes:** is a clustering algorithm that is similar to the k-means algorithm but is used for categorical data rather than numerical data. Instead of calculating the mean of a cluster, it calculates the mode, which is the most common value in a cluster.
* **Spectral clustering**: is a clustering algorithm that can be used to find clusters in non-linearly separable data. It works by first constructing a similarity matrix for the data, which encodes the similarity between each pair of data points. The algorithm then constructs the Laplacian matrix, which is a transformation of the similarity matrix. The Laplacian matrix encodes the relationships between data points in terms of their similarity and proximity to one another. Next, the algorithm finds the eigenvectors of the Laplacian matrix corresponding to its K lowest non-zero eigenvalues, where K is the number of clusters. These eigenvectors are used as the coordinates of the data points in a new, low-dimensional space. Finally, the data points in this new space are clustered using k-means algorithm.

**Chosen ML performance evaluation metrics:**

In order to analyse all the relevant aspects we decided to use 2 different types of metrics: the first one gave us general information about how good the results of the application of our ML algorithms were (in fact we also used them to select the values of their parameters), the second one instead allowed us to compare the results of the same algorithms applied on the imputed datasets and on the original ones.

Inside the first category we chose to use:

* **Silhouette score**
* **Calinski-Harabasz score**

While for the second one:

* **Adjusted Mutual Information (AMI)**
* **Completeness score**

**Imputation techniques selected:**

As for the ML algorithms, different datasets required different imputation techniques due to the nature of their features.

“Abalone” dataset:

**Standard**) Single imputation using the mode of features

**Advanced**) Multiple Imputation by Chained Equations (MICE) for the numerical values and a KNN classifier trained to infer the categorical one

“Users” dataset:

**Standard**) sklearn SimpleImputer to infer the mode

**Advanced**) Multiple Imputation by Chained Equations (MICE) with KNN classifier

1. **PIPELINE IMPLEMENTATION**

In order to optimize time we decided to split efforts and focus on different datasets. Anyhow we concurrently reached very similar results, so we simply chose some common metrics and decided to refer to these.

For the “abalone” dataset the pipeline has been the following:

1. Identify optimal number of clusters using the elbow method with respect to the Silhouette score on the original dataset for K-Protypes and manually tuned eps and minPts for DBSCAN
2. Perform clustering over the original dataset: results have been visualized with both PCA and t-sne techniques in order to provide also a graphical representation of the newly found clusters
3. Compute “absolute” scores on the clustering results
4. Create NaN-injected versions of the original dataset with the provided script
5. Infer missing values with the chosen imputation techniques: thanks to our choices we were able to perform imputation completely without needing to one-hot encode the categorical feature
6. Assess DQ Accuracy dimension of the imputed datasets with respect to the original one
7. Using the best performing clustering and imputation algorithm found in the previous steps plot Silhouette, AMI and Completeness scores at changing percentages of injected missing values to evaluate their impact on the performance of the ML algorithm
8. Compare ML algorithms and imputation techniques results varying the completeness percentage

For what concerns the “users” dataset, we performed the following main steps:

1. Creating a dirty version of dataset using the provided script to inject NaNs
2. Identify optimal number of clusters using the elbow method on the original dataset
3. Perform clustering over the original dataset to select the two ML algorithms to use in the rest of pipeline. We used Silhouette and Calinski-Harabasz indexes to make this decision. At the end, we selected KModes and Spectral Clustering. During the whole pipeline, we used t-sne technique for visualizing the computed clusters in a low-dimensional space
4. Use chosen standard and advanced imputation methods to infer missing values. For the imputation of the mode, we were able to directly impute the categorically encoded version of the dirty dataset. For the MICE KNN imputer instead, we had to impute the one-hot encoded version of the dirty dataset
5. Assess DQ Accuracy dimension of the imputed dataset with respect to the original dataset
6. Evaluate the clustering results over the imputed dataset. To compare the results of the different imputation methods in a robust way, we decided to compute, for each imputation method, an average Silhouette and Calinski-Harabasz score with respect to all the clustering algorithms. In this phase, we also computed the AMI and Completeness metrics following the same approach, to compare the closeness of the clusters obtained from the imputed dataset with the ones we got by applying the clustering on the original dataset
7. Using the best performing clustering algorithm found in the previous steps (Spectral clustering), plot Silhouette, AMI and Completeness scores at changing percentages of injected missing values to evaluate their impact on the performances of the ML algorithm, for the two different imputation methods. The same has been done for the accuracy of the imputed datasets to assess this DQ dimension

3. RESULTS

a. Description of the main results obtained

b. ML performance comparison between the imputation/outlier detection techniques you have implemented