CIDA Assessment 1 - Edward Alvin Koesmanto (2501963141/22081335)

Number 1

Number of samples in dataset before preprocessing = 26215
Number of samples in dataset after preprocessing = 22718
Number of features before preprocessing = 10
Number of features after preprocessing = 42
Data types before preprocessing = Numerical, Categorical
Data types after preprocessing = Numerical, Categorical

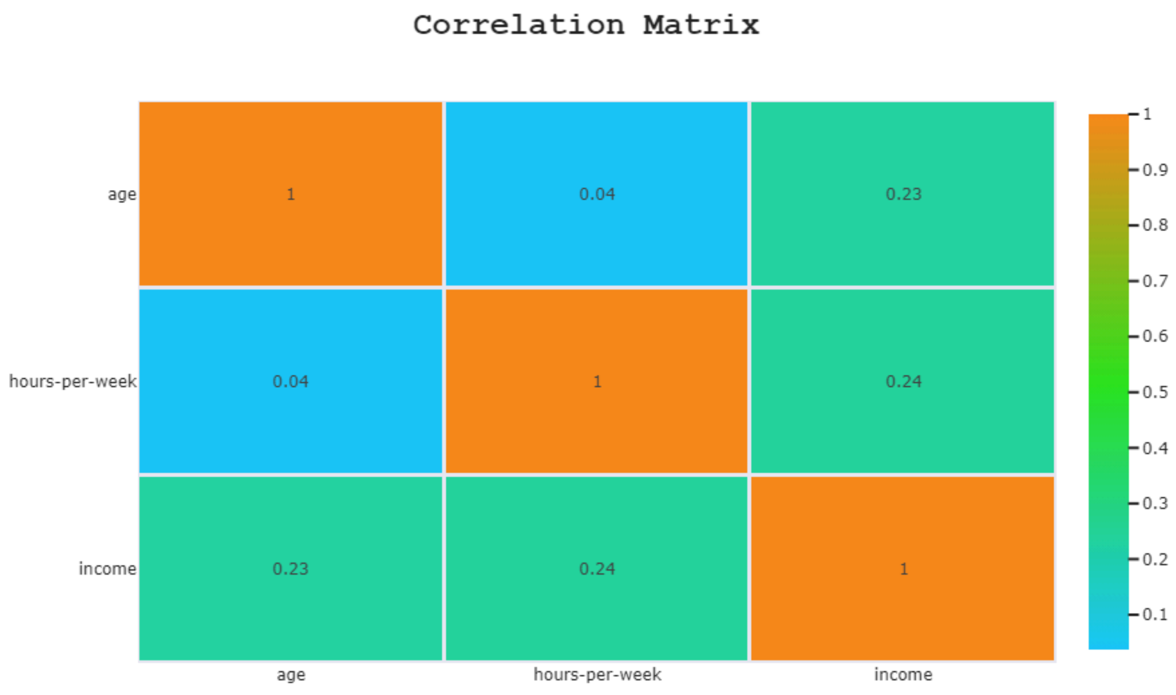
Statistical Analysis

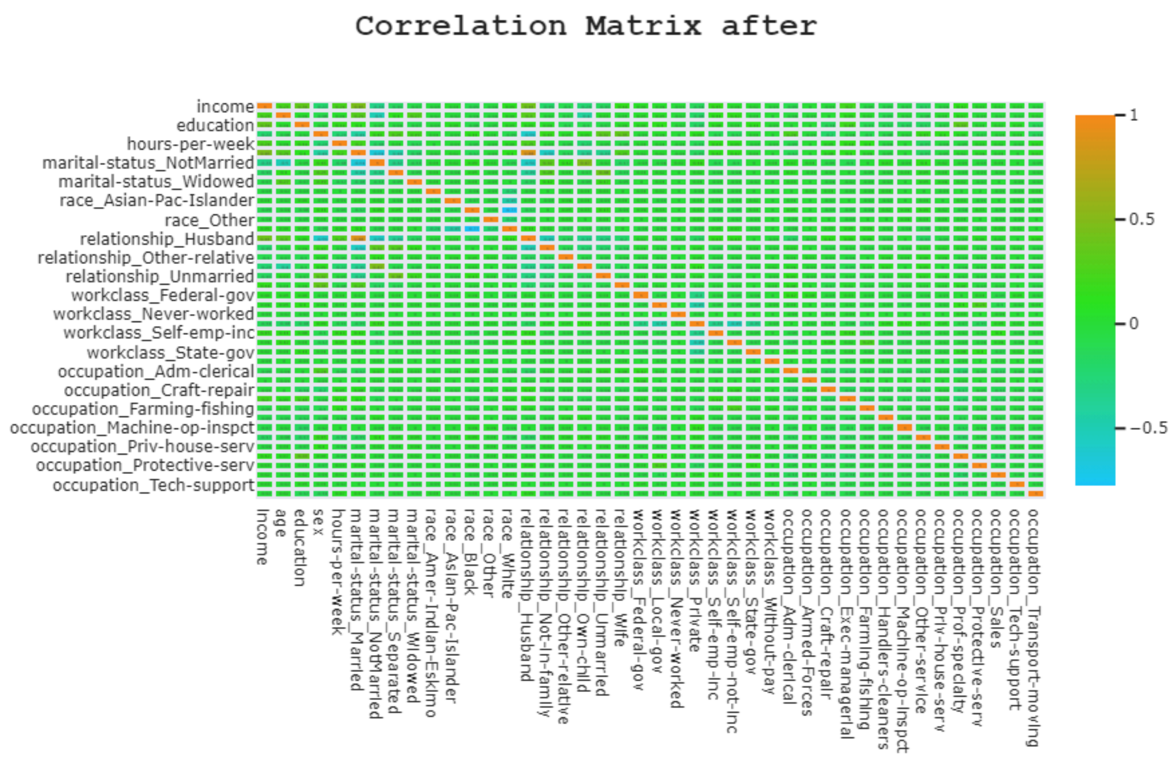
	Age	Hours-Per-Week
Mean	39.05	40.81
Standard Deviation	13.56	12.35
Min	17.00	1.00
Q1	28.00	40.00
Median	38.00	40.00
Q3	48.00	45.00
Max	90.00	99.00

Missing Values

Workclass = 1396
Occupation = 1401

Correlation Analysis





Dealing with Missing Values

As the missing values are all categorical, I used a mode function (getting the most frequent value for that feature). By doing so, we eliminate the missing values and replace them with the mode of that feature.

Dropping Duplicates

After dropping the duplicates, the number of samples in the dataset changed from 26215 to 22718 meaning there were 3497 duplicates in the dataset. After dropping, we can get a more consistent dataset.

Handling Categorical Variables

For the variables of education and sex, I used mapping so that we get the desired variables such as Preschool, Male, and Female. For the rest of the variables, I used get_dummies() so Pandas automatically changes the features according to the values. After doing so, the number of features of the dataset changed from 10 to 42.

Splitting Training and Testing data

The dataset was split into training and testing set using the train_test_split() function provided by the sklearn library with the parameters test_size=0.1 and random_state=123 which means 10% of the data will be used for testing and randomization to ensure unbiasedness. X is the input data consisting of all features without income and y is the true label (target) consisting of only the income feature.

Applying Normalization

I used MinMaxScaler to normalize both training and testing data of X (input data).

Number 2

Logistic Regression

Logistic regression is a statistical approach employed to analyze the association between a binary dependent variable and one or more independent variables. It is frequently applied in classification scenarios where the outcome falls into two distinct categories, like "yes" or "no," "spam" or "not spam," or "pass" or "fail." (LaValley, 2008).

The logistic regression model assesses the probability of an observation belonging to a particular category. Unlike linear regression, which handles continuous dependent variables ranging from negative to positive infinity, logistic regression computes the probability of a binary outcome using a logistic function, often known as the sigmoid function. This function maps any real number input to a range bounded between 0 and 1 (Nick & Campbell, 2007).

In logistic regression, the coefficients illustrate the change in log-odds of the dependent variable corresponding to a one-unit adjustment in the independent variable. These coefficients are estimated using maximum likelihood estimation, and the model's performance is assessed using metrics such as precision, recall, accuracy, and the area under the receiver operating characteristic curve (ROC AUC). (Kleinbaum et al., 2002).

Logistic regression has hyperparameters which are settings or configurations that we can alter to control the behavior and performance of the model such as (Ahmed Arafa et al., 2022):

1. Penalty: Logistic regression models can incorporate a penalty to the loss function to prevent overfitting. The penalty can be either L1 (Lasso) or L2 (Ridge) regularization. L1 regularization promotes sparsity in the coefficient estimates by incorporating the absolute values of the coefficients into the loss function, whereas L2 regularization penalizes the square of those coefficients.
2. C: This hyperparameter is inversely proportional to regularization strength. A smaller value of C indicates stronger regularization, while a larger value indicates weaker regularization. Tuning this hyperparameter helps in balancing the balance between effectively fitting the training data and preventing overfitting..
3. Solver: The solver is the algorithm used for optimizing the loss function. Common choices include 'liblinear', 'lbfgs', 'sag', and 'saga'. The choice of solver will affect the training time and performance of the model, especially for large datasets.
4. Max Iterations: This hyperparameter denotes the maximum number of iterations allowed for the model to converge. If the solver fails to converge within the set number of iterations, it might trigger a warning or an error.
5. Multi-class setting: In logistic regression, we can extend the model to manage classification tasks in multi-class problems using strategies like One-vs-Rest (OvR) or One-vs-One (OvO). The choice between these strategies may also be considered as a hyperparameter.
6. Class Weight: This hyperparameter is used to manage imbalance in class by assigning weights to the available classes. It aids in enhancing the model's sensitivity to the minority class by imposing heavier penalties on misclassifications of the minority class.
7. Tolerance: Tolerance is the stopping criterion for the optimization algorithm. It determines the convergence threshold for the change in the objective function value between iterations. Once the alteration in the objective function stoop under this threshold, the optimization process stops.

SVM

Support Vector Machine or SVM is a powerful supervised machine learning algorithm utilized in classification and regression tasks. It operates by identifying the optimal hyperplane within a high-dimensional space that efficiently segregates data points belonging to distinct classes. (Yu & Kim, 2012).

In classification tasks, SVM strives to discover a decision boundary that enhances the gap between classes. The data points closest to this boundary are known as support vectors. The hyperplane is chosen such that it optimizes the separation between the support vectors and the decision boundary, which helps in achieving better generalization to unseen data (Yu & Kim, 2012).

SVM can manage both linearly distinguishable and non-linearly distinguishable data by employing various kernels, including polynomial, linear, sigmoid, and radial basis function (RBF) kernels. These kernel functions alter the input data into higher-dimensional spaces, potentially rendering the data linearly distinguishable, thereby enabling SVM to identify an appropriate hyperplane (Yu & Kim, 2012).

In regression tasks, SVM works by fitting a hyperplane to predict continuous outcomes. It seeks to reduce the discrepancy between the predicted values and the actual values while simultaneously maximizing the margin around the hyperplane (Yu & Kim, 2012).

SVM offers numerous advantages, such as its efficacy in high-dimensional spaces, resilience against overfitting, and adaptability in managing diverse data types. Nonetheless, it can be computationally demanding, particularly with large datasets, and the selection of the kernel and its parameters may necessitate meticulous tuning (Patle & Chouhan, 2013).

Hyperparameters for SVM:

1. Kernel: SVM can use various kernels to modify the input data into higher-dimensional spaces. Common choices include:
    a. Linear Kernel: $K(x, y) = x^T * y$
    b. Polynomial Kernel: $K(x, y) = (x^T * y + c)^d$
    c. Radial Basis Function (RBF) Kernel:
       $K(x, y) = EXP(-\gamma||x - y||^2)$
    d. Sigmoid Kernel: $K(x, y) = tanh(\alpha x^T * y + c)$
2. C: The regularization parameter C trades off between expanding the margin and minimizing the classification error. A smaller value of C leads to a simpler decision boundary with more margin, while a larger value allows the model to be more flexible and potentially fit the training data better.
3. Gamma (for RBF kernel): Gamma (γ) is a parameter for the RBF kernel. It defines how far the influence of a single training example reaches, with low values meaning far and high values meaning close. A smaller value of gamma leads to a smoother decision boundary, while a larger value makes the decision boundary more irregular.
4. Degree (for polynomial kernel): The degree (d) is a parameter for the polynomial kernel. It defines the degree of the polynomial used by the kernel function. Higher degrees allow the model to fit the training data more precisely, potentially leading to overfitting.

5. Coefficient (for polynomial and sigmoid kernels): The coefficient (c) is a parameter for the polynomial and sigmoid kernels. It is the constant term in the polynomial and sigmoid functions. Adjusting this parameter can impact the shape of the decision boundary.
6. Class Weights: SVM is able to address class imbalance by mapping varying scores to different classes. This hyperparameter allows you to specify the importance of each class, helping the model to focus more on the minority class.
7. Tolerance: Tolerance is the stopping criterion for the optimization algorithm. It determines the convergence threshold for the change in the objective function value between iterations.
8. Max Iterations: This hyperparameter dictates the maximum number of iterations allowed for the optimization algorithm to reach convergence. Failure to converge within the specified iterations may trigger a warning or an error.

Test Results

	Non Cross Validated	Cross Validated
Logistic Regression	0.809	0.809
SVM	0.806	0.798

Fine Tuning

For the fine tuning of the models, I used GridSearchCV to automatically run different parameters.
Logistic Regression Parameters: 'solver': ['lbfgs', 'liblinear', 'sag','saga'], 'max_iter': [200, 300, 400, 500], 'C': [5, 10, 15, 100]
SVM Parameters: 'kernel': ['linear', 'rbf'], 'gamma': [0.1, 0.5, 1], 'C': [0.1, 0.5, 1]

The logistic regression has the optimum score when presented with the parameters solver='lbfgs', C=5, max_iter=300 and the SVM with parameters kernel='linear', C=1, gamma=0.1.

After Fine Tuning

After fine tuning, we get the results as follows:

	Results
Logistic Regression	0.809
SVM	0.808

	Results on Test set
Logistic Regression	0.808
SVM	0.803

Both logistic regression and SVM achieved relatively high accuracy scores on the test set. Logistic regression slightly outperformed SVM, with an accuracy of 0.808 compared to 0.803 for SVM. This suggests that logistic regression may have made slightly more accurate predictions on this particular dataset.

Number 3

K-means

K-means, a widely-used unsupervised machine learning algorithm, are utilized for clustering data points into groups or clusters according to their similarity. The objective of the K-means algorithm is to divide the data into k clusters, with each data sample assigned to the cluster whose average is closest, acting as the prototype of the cluster (Ahmed et al., 2020).

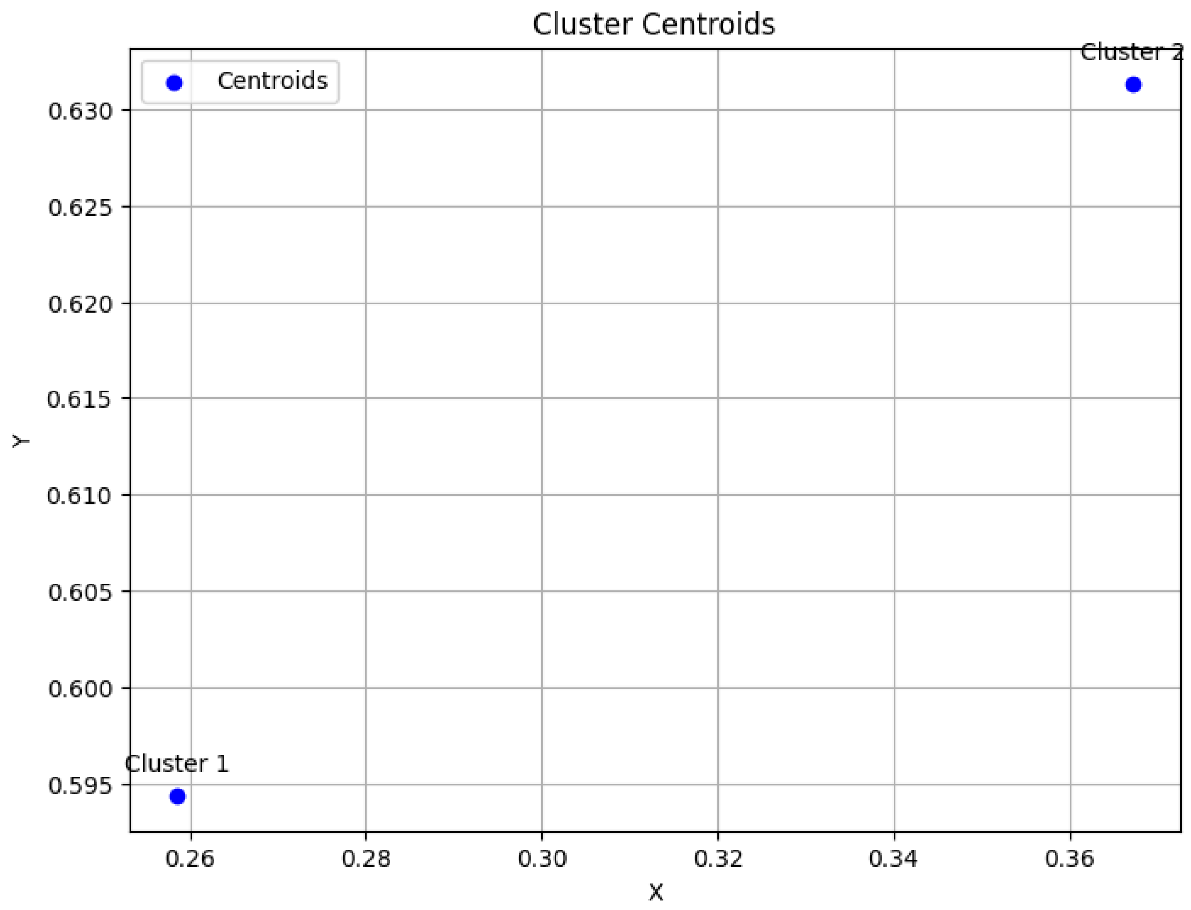
Steps for K-means

1. Initialization: The algorithm commences by randomly initializing k cluster centroids. These centroids act as the initial approximations for the centers of the clusters.
2. Assignment: Every data sample is allocated to the closest centroid from a calculation of a distance measure, typically the Euclidean distance. The data sample is assigned to the cluster whose centroid is nearest to it.
3. Update: Following the assignment of each data point to a cluster, the centroids are then updated by calculating the average of all the data points assigned to each cluster. The recalculated mean becomes the new centroid for that cluster.
4. Iteration: The second and third steps are iteratively repeated until all data samples are able to converge. Convergence occurs when all centroids no longer change significantly or when a predetermined number of iterations is completed.

5. Final Result: Once convergence is reached, the algorithm delivers the final cluster assignments, with each data point assigned to one of the k clusters.

Clusters for this dataset

For this data, I chose the number of clusters based on the number of unique values in the y_train set which resulted in 2 clusters. 1 cluster has 10797 data samples and the other has 9649 with a distance of 1.677 from the prototype of cluster 1 and cluster 2 centroids respectively. The Euclidean distance between the prototypes of clusters in K-means provides a measure of dissimilarity or separation between the centroids. Specifically, a Euclidean distance of 1.677 suggests that the prototypes of the clusters are moderately separated in the feature space.



	Cluster 1	Cluster 2
Mean	0.165	0.160
Standard Deviation	0.145	0.130

Accuracy with Testing Set

In this case, we observe varying performances among three machine learning algorithms applied to the income dataset. K-means clustering attains an accuracy of 0.282 for the testing set with a silhouette score of 0.209, while logistic regression and SVM possess higher accuracies of 0.808 and 0.803 respectively. The difference in accuracy suggests that K-means may not be as effective in accurately categorizing data points compared to the supervised learning models. While K-means is unsupervised and primarily focuses on clustering data points based on similarity, logistic regression and SVM are supervised algorithms trained to predict outcomes based on labeled data. Despite its lower accuracy, K-means may still provide valuable insights into the structure of the data and potential groupings. However, the higher accuracies achieved by logistic regression and SVM indicate their superiority in predictive performance for this dataset. The decision between these algorithms ultimately relies on the particular requirements or objectives of the analysis and the nature of the dataset, with logistic regression and SVM being preferable for prediction tasks where labeled data is available, while K-means may be more suitable for exploratory analysis and clustering tasks.

Supplementary Code

https://github.com/fl-sll/CIDA/blob/master/Assessment/Assessment1/main.ipynb

References

Ahmed Arafa, A. H., Radad, M., Badawy, M. M., & El-Fishawy, N. (2022). Logistic regression hyperparameter of optimization for cancer classification. *Menoufia Journal of Electronic Engineering Research, 31*(1), 1-8.

Ahmed, M., Seraj, R., & Shamsul Islam, S. M. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics, 9*(8), 1295.

Kleinbaum, D. G., Deitz, K., Gail, M., Klein, M., & Klein, M. (2002). *Logistic Regression*. New York: Springer-Verlag.

LaValley, M. P. (2008). Logistic Regression. *Circulation, 117*(18), 2395-2399.

Nick, T. G., & Campbell, K. M. (2007). Logistic Regression. *Topics in biostatistics*, 273-301.

Patle, A., & Chouhan, D. S. (2013). *SVM Kernel functions for classification* [Conference session]. 2013 International Conference on Advances in Technology and Engineering (ICATE). IEEE.

Sinaga, K. P., & Yang, M. S. (2020). Unsupervised K-means clustering algorithm. *IEEE access, 8*, 80716-80727.

Yu, H., & Kim, S. (2012). SVM Tutorial-Classification, Regression and Ranking. *Handbook of Natural computing, 1*, 479-506.