**CSC 4850 Machine Learning Project**

**Classification**

1. Data Exploration: The process of loading the datasets involved reading each file using pandas.read\_table with whitespace as the delimiter. The training data was then split into training and validation sets using an 80-20 ratio via train\_test\_split to ensure robust model evaluation. During data exploration, missing values were identified, represented by placeholders like 1.00000000000000e+99. These missing values, particularly prominent in some datasets, could impact model performance if not appropriately handled, emphasizing the need for effective preprocessing.
2. Data Preprocessing: Data preprocessing involved two primary steps to prepare the datasets for modeling. First, missing values, identified as 1.00000000000000e+99, were replaced with NaN and subsequently handled using mean imputation. This ensured that each feature’s missing values in the training and validation sets were replaced with the respective dataset’s mean, maintaining consistency and minimizing bias introduced by missing data. Second, for datasets with high dimensionality (more than 100 features), standardization was applied using StandardScaler to normalize feature ranges. This scaling step ensures that features contribute equally to model learning and prevents dominance by features with larger ranges.
3. Model Training: The choice of models was tailored to the characteristics of each dataset. For Dataset 1, K-Nearest Neighbors (KNN) was selected due to its effectiveness in lower-dimensional spaces and its simplicity for datasets without high feature counts. For Datasets 2, 4, and 5, Random Forest was chosen as it excels in handling high-dimensional data by reducing overfitting through ensemble learning and feature importance mechanisms. For Dataset 3, an intermediate-sized dataset, a Support Vector Machine (SVM) with an RBF kernel was used, leveraging its ability to handle non-linear relationships and maintain robust performance in moderately sized feature spaces. The training process involved splitting the datasets into training and validation sets, ensuring robust model evaluation during development. Models were trained using their respective training datasets and labels, after which hyperparameter tuning was performed to optimize performance.
4. Store Predictions: Predictions were generated for each validation dataset by loading the corresponding pre-trained models. For each dataset, the validation data was passed through the loaded model to obtain predictions. These predictions, representing the model’s classification results, were stored in separate files (Classification{i}.txt), ensuring each dataset’s results were saved distinctly.
5. Model Evaluation: The evaluation of model performance for each dataset was conducted using standard metrics, including accuracy, precision, recall, and F1 score. Predictions from the models were compared against the true labels from the validation datasets. Dataset 1 achieved high performance with an accuracy of 93.33% and an F1 score of 92.81%, while Dataset 2 performed even better with an accuracy of 95% and an F1 score of 96.67%. Similarly, Dataset 4 exhibited strong results, with an accuracy of 94.51% and an F1 score of 94.54%. In contrast, Dataset 3 displayed a lower accuracy of 35.32% and an F1 score of 30.64%, reflecting potential challenges in the dataset or model fit. Dataset 5 achieved moderate results, with an accuracy of 70.09% and an F1 score of 67.95%.

**Missing Data**

1. Data Exploration: To explore the datasets containing missing values, each file was loaded and inspected for the presence of placeholders indicating missing data, specifically 1.00000000000000e+99, which was replaced with NaN for analysis. Dataset 1, with dimensions of 242 rows by 14 columns, had a total of 3,388 values, of which 118 were missing (3.48%). Dataset 2, larger at 758 rows and 50 columns, contained 37,900 total values with 3,762 missing entries (9.93%). Dataset 3 exhibited a high level of sparsity, with 273 rows and 79 columns (21,567 values), and a striking 17,752 missing values, constituting 82.31% of the data. These statistics highlight the varying degrees of missingness across the datasets.
2. Data Visualization: To gain insights into the distribution of missing values across the datasets, several visualizations were generated. For each dataset, a heatmap was used to show the spatial distribution of missing values (NaN). The heatmaps revealed clusters or patterns of missing data, aiding in identifying whether missingness is random or systematic. Bar plots highlighted the count of missing values per column, providing a clearer understanding of which features were most affected by missingness. This is particularly useful for prioritizing imputation strategies on columns with significant data loss. Finally, an overall summary plot contrasted the count of missing values with observed data for each dataset. This provided a high-level view of the extent of missingness, emphasizing the sparsity in datasets such as Dataset 3, where missing values overwhelmingly dominate.
3. KNN Imputation: K-Nearest Neighbors (KNN) imputation was applied to address missing values in the datasets. Using a KNNImputer with n\_neighbors=5 and uniform weighting, missing values were estimated based on the mean of the five nearest samples in the feature space. This method leverages the similarity between observations, ensuring that imputations are informed by data patterns. Each dataset underwent imputation, transforming missing data into plausible values while preserving the dataset structure.

**Conclusion**

In the classification process, several models were trained and evaluated on five datasets, with varying results. The K-Nearest Neighbors (KNN) classifier performed well on the first dataset, achieving high accuracy and precision. Random Forests yielded strong performance across datasets with high-dimensional data, such as the second, fourth, and fifth datasets. Meanwhile, the Support Vector Machine (SVM) showed lower performance on the third dataset. Model evaluation metrics, including accuracy, precision, recall, and F1 score, revealed promising results, particularly for high-dimensional datasets where Random Forests excelled.

For the missing data analysis, we identified varying degrees of missingness across three datasets. While one dataset had a manageable 3.48% missing values, another dataset had a substantial 82.31% missingness. KNN imputation was successfully applied to fill missing values, producing plausible estimates based on the nearest neighbors. The imputed datasets were saved for further analysis, offering a more complete set of data for subsequent modeling tasks.