**Machine Learning Project Report**

GSU Department of Computer Science

Ethan Bang, Brandon Chung, Mark Denina

14 November 2024

1. **Introduction**

In this project, we aimed to determine the most effective method by considering several key factors. The train and test data files presented varied formats, requiring tailored processing approaches. Additionally, the datasets were large, requiring classification algorithms that efficiently handle and analyze the data. By comparing multiple methods, we identified the most appropriate approach being the Random Forest classifier to deliver the most consistent and reliable results.

1. **Methodology**
   1. **Data Processing**

* Data Loading
* For each dataset, we loaded the training data, training labels, and test data from the given text files.
* Datasets were formatted to handle the missing value markers (represented by 1.00000000e+99).
* Missing Value Estimation
* Missing values were replaced using KNN imputation (a machine learning technique that estimates missing values by averaging values of its neighbors).
* We chose a neighbor n-value of 5, so missing values were estimated based on the average values of the nearest 5 data points.
  1. **Classification**

After data processing, we used the Random Forest classifier to predict test class labels. Random Forest classifier was chosen because of its efficiency with complex, non-linear relationships.

The classification steps were as follows:

1. Train the Random Forest classifier on the imputed training data.
2. Predict labels for the test data.
3. Output predicted labels.
   1. **Cross-Validation**

* StratifiedKFold was used to cross-validate the data results. This method is useful when working with imbalanced datasets, for it maintains the proportion of classes in each fold. This allows for a more reliable and representative performance metric.

1. **Results**

For each dataset, the predicted test labels were outputted as and saved in separate text files. Below is a summary of each output file.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dataset | Features | Samples  (Train/Test) | Classes | Classifier | Imputer | Output Files |
| Dataset 1 | 3312 | 150/53 | 5 | Random Forest | KNN |  |
| Dataset 2 | 9182 | 100/74 | 11 | Random Forest | KNN |  |
| Dataset 3 | 13 | 6300/2693 | 9 | Random Forest | KNN |  |
| Dataset 4 | 112 | 2547/1092 | 9 | Random Forest | KNN |  |
| Dataset 5 | 11 | 1119/480 | 6 | Random Forest | KNN |  |

The output files contain the predicted labels for each dataset, which provides insights into each classification task.

1. **Potential Improvements**

* Automated Data Processing: Instead of processing each dataset individually, consider implementing a loop to streamline the process.
* Evaluation Metrics: To assess the methods we use, consider implementing some metrics such as accuracy and precision to provide a comprehensive evaluation.
* Alternative Classifiers: Other classifiers could perform better for the task presented, so testing other classifiers such as SVM or Random Forests could improve performance. \*We have decided to move forward with Random Forest instead of Naïve Bayes\*

1. **Conclusion**

In this project, we implemented K-Nearest Neighbors (KNN) and Random Forest —one method introduced in class that we felt confident using. To ensure the accuracy of our results, we applied the Stratified K-Fold Cross-Validation, a variation of the K-Fold Cross Validation methods. Leveraging VSCode, Python, and the scikit-learn library, we developed code that efficiently processed each dataset, accurately generated test labels, and addressed missing values. These tools enabled us to build a reliable solution that effectively handled the data and met project requirements.