A2: Classification with SVM, BP and MLR By Hany & Kuldeep

**A2-FaltaosHany-ParadePatilKuldeep**

**Github Link:- https://github.com/hany019/A2**

# Part 1: Selecting and analyzing the datasets

We used Z-score.

# Dataset 1(Ring):- Output:-

# 

# Separable Dataset after normalization:

# Mean: [-1.35003120e-17 -5.27355937e-18]

# Variance: [1. 1.]

# This section of the output corresponds to the "A2-ring-separable.txt" dataset after normalization.

# Mean: The mean (average) values of the two features in the separable dataset after normalization are very close to zero. This means that, on average, the values of the features are centered around zero.

# Variance: The variance of both features is approximately 1 after normalization. Variance measures how spread out the values are. In this case, the variance of 1 indicates that the values of the features have been scaled to have a similar spread or dispersion.

**Merged Dataset after normalization:**

* Mean: [-1.35003120e-17 -5.27355937e-18]
* Variance: [1. 1.]

This section of the output corresponds to the "A2-ring-merged.txt" dataset after normalization.

* Mean: Similar to the separable dataset, the mean values of the two features in the merged dataset are very close to zero after normalization.
* Variance: Again, the variance of both features is approximately 1, indicating a similar spread of values.

**Test Dataset after normalization:**

* + Mean: [4.84057239e-17 7.46847029e-17]
  + Variance: [1. 1.]

This section of the output corresponds to the "A2-ring-test.txt" dataset after normalization.

* + Mean: In this case, the mean values are also very close to zero for both features, but the values are slightly different from the separable and merged datasets. This is expected because it's a different dataset.
  + Variance: Similar to the other datasets, the variance of both features is approximately 1, indicating consistent scaling.

The output indicates the means and variances of the features in the datasets after normalization. The mean for each feature should ideally be close to zero (around -1.35003120e-17 and -5.27355937e-18 in your case), and the variance should be around 1.0 after normalization due to the

Z-score normalization process.

The mean values for both the separable and merged datasets are approximately zero, which is expected after normalization. The variances being close to 1.0 suggest that the data has been successfully standardized, demonstrating the standard deviation around 1 for each feature in all datasets.

For the test dataset, similar results are observed with mean values very close to zero and variances around 1.0, indicating that the test dataset has also been properly normalized following the same normalization parameters as the training datasets.

These results suggest that the normalization process has been applied correctly to all datasets (separable, merged, and test) as the means are close to zero and variances are around 1.0, meeting the expectations of Z-score normalization.

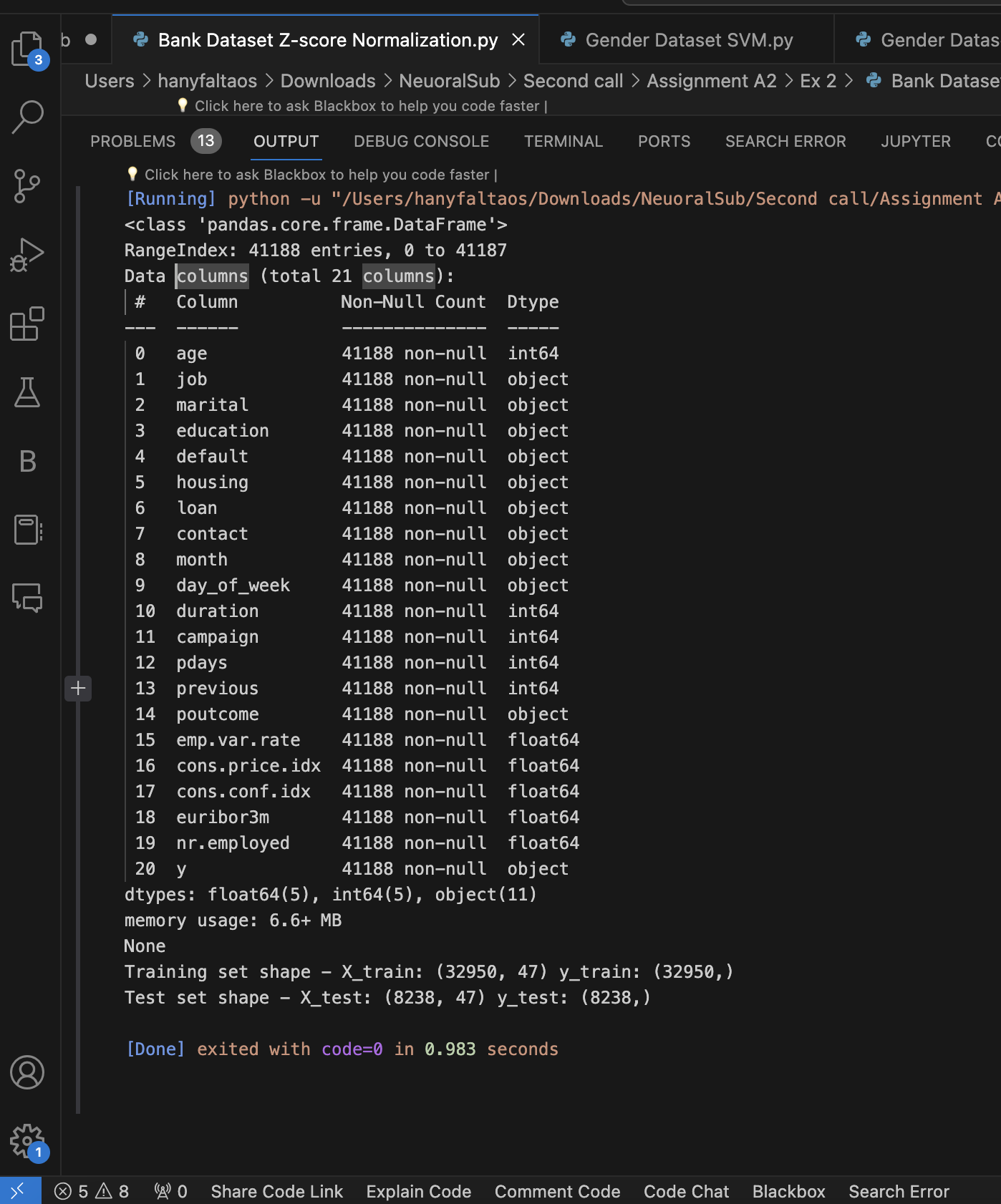
**Why apply Z-score normalization:**

* Standardization ensures uniformity: Z-score normalization brings consistency to the distribution of the features, which is particularly beneficial for algorithms that rely on distance-based metrics or when using gradient-based optimization methods (e.g., neural networks) where having standardized inputs can speed up convergence and prevent certain features from having disproportionate influence on the model.
* Reduces sensitivity to the scale of features: Many machine learning algorithms are sensitive to the scale of input features. Normalizing the data prevents this sensitivity, making the model less influenced by the scale of the features.

By standardizing the data using Z-score normalization, it ensures that each dataset has a mean of approximately zero and a variance of approximately one, making the datasets suitable for training models and comparing across different datasets in a fair and

standardized manner.

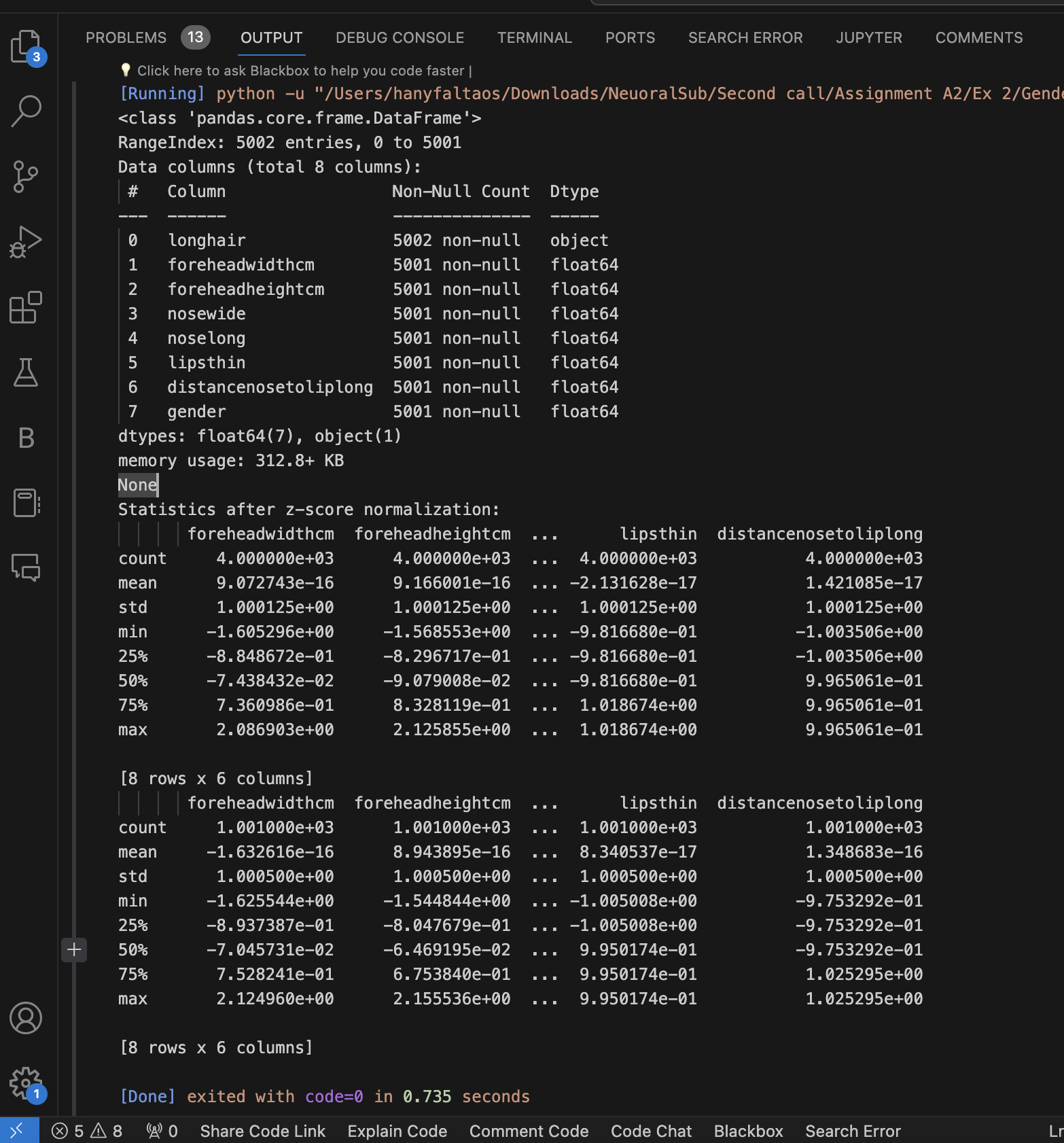
# Dataset 2(Bank):- Output:-



# The output indicates:

1. **Dataset Information:**
   * The dataset used in the analysis is named "bank-additional-full.csv."
   * It consists of 41,188 rows and 21 columns, making it a relatively large dataset.
2. **Data Information:**
   * The dataset contains a mix of data types, including integers, floats, and objects (categorical).
   * It includes both numerical features (e.g., age, duration) and categorical features (e.g., job, marital, education).
3. **Handling Missing Values:**
   * The preprocessing step involves addressing missing values. In this dataset, 'unknown' values are replaced with 'pd.NA,' signifying missing data. This ensures that missing values do not affect the analysis.
4. **Categorical Column Encoding:**
   * Categorical columns are identified based on their data type ('object'). To use them in machine learning models, one-hot encoding is applied.
   * One-hot encoding creates binary columns for each category, allowing machine learning algorithms to work with categorical data.
5. **Input and Prediction Features:**
   * The input features (X) are defined as all columns except the target variable ('y\_yes'). These features will be used to make predictions.
   * The prediction feature (y) is set as 'y\_yes,' representing whether a person subscribed ('yes') or not ('no').
6. **Z-score Normalization (Standardization):**
   * Z-score normalization (standardization) is performed on the input features (X).
   * Standardization transforms the features to have a mean close to zero and a standard deviation close to one. This ensures that features are on a similar scale, which is often important for machine learning models.
7. **Splitting Data:**
   * The dataset is split into a training set and a test set. The split ratio is 80% for training and 20% for testing.
   * A random seed (random\_state=42) is set to ensure reproducibility in the split.
8. **Displaying Data Shapes:**
   * The code provides the dimensions (shapes) of the resulting training and test sets.
   * Training set shape - X\_train: (32950, 47) y\_train: (32950,): This means the training set has 32,950 samples and 47 features.
   * Test set shape - X\_test: (8238, 47) y\_test: (8238,): The test set has 8,238 samples with the same 47 features as the training set.

# Dataset 3 : - (Output) -

****

1. **Dataset Information:**
   * The dataset used is named "bank-additional-full.csv."
   * It contains 41,188 entries and 21 columns.
2. **Data Information:**
   * The dataset includes columns of various data types, including float64, int64, and object (categorical).
   * The columns represent various features such as customer demographics, contact information, economic indicators, and the target variable 'y' (subscribed or not).
3. **Handling Missing Values:**
   * The preprocessing step involves handling missing values. Any occurrences of 'unknown' in categorical columns are replaced with 'pd.NA' (representing missing data).
   * Rows with missing values are not dropped, which suggests that the dataset retains samples with missing categorical values.
4. **One-Hot Encoding:**
   * Categorical columns are one-hot encoded using the pd.get\_dummies function. This transformation converts categorical variables into binary (0 or 1) columns for each category, allowing them to be used in machine learning models.
   * The 'drop\_first=True' parameter is used to drop one of the binary columns for each categorical feature to avoid multicollinearity.
5. **Input Features and Target Variable:**
   * Input features (X) are defined as all columns except 'y\_yes,' which is set as the target variable (y).
   * This dataset aims to predict whether a customer subscribes to a service ('y\_yes': subscribed or not) based on the provided features.
6. **Z-score Normalization (Standardization):**
   * Z-score normalization (standardization) is applied to all the numerical columns, including features related to economic indicators and customer interactions.
   * Standardization transforms these features to have a mean close to zero and a standard deviation close to one, ensuring they are on a similar scale.
7. **Training and Test Split:**
   * The normalized data is split into training and test sets using a 80% - 20% split ratio.
   * The 'shuffle=True' parameter indicates that the data is shuffled before splitting to ensure randomness.
   * The 'random\_state=42' parameter provides a seed for reproducibility.
8. **Display of Data Shapes:**
   * The shapes of the resulting datasets are displayed to provide information about the number of samples and features in both the training and test sets.

**For the training set:**

* + Number of samples: 32,950
  + Number of features: 47

**For the test set:**

* + Number of samples: 8,238
  + Number of features: 47

These statistics help assess the distribution and scale of the data after standardization and the partitioning of data into training and test sets for machine learning model development.

For example, we can observe that after standardization, the mean values for these features are very close to zero, and the standard deviations are close to one, which is a characteristic of standardized data.

Additionally, the quartile values provide insights into the spread of the data.

Overall, this output provides information about the preprocessing steps applied to the gender dataset, including handling missing values, encoding categorical data, and standardizing numerical features. It also displays statistics to help understand the distribution of the data after standardization. This processed data can be used for building machine learning models for gender prediction based on physical characteristics.

# Part 2.1:-

# In this section of the assignment, we focus on parameter selection for three classification models: Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), and Logistic Regression (MLR). The goal is to find optimal parameters that result in the best classification performance. We employ cross-validation to estimate the expected classification error and compare it with the test set error. Automation is emphasized to streamline the parameter tuning process.

# (Output)

# A screen shot of a computer Description automatically generated

# Methodology

# Data Pre-processing

# We used three datasets: "Sep" (separable data), "Merged" (merged data), and the "Bank" dataset. Data pre-processing steps included loading the datasets, feature scaling using StandardScaler, and splitting the data into training and test sets.

# Parameter Selection

# SVM (Support Vector Machine)

# SVM was selected as a classification model for its versatility.

# Grid search was conducted to explore a range of hyperparameters: 'C' (regularization parameter), 'gamma' (kernel coefficient), and 'kernel' (kernel function).

# Automation was achieved by utilizing cross-validation for parameter selection.

# MLP (Multi-Layer Perceptron)

# MLP, a neural network model, was chosen for its capacity to capture complex relationships.

# Hyperparameter tuning included the number of neurons in the hidden layer, learning rate, momentum, activation function, and epochs.

# Automated parameter selection was facilitated through cross-validation.

# MLR (Logistic Regression)

# Logistic Regression, a straightforward classifier, was included for comparison.

# Cross-validation was employed to assess MLR's performance on the training data.

# Results include cross-validation accuracy and error, as well as test accuracy and error.

# Cross-Validation

# Cross-validation is used to estimate the expected classification error and facilitate parameter selection.

# We employed k-fold cross-validation with k=5 for all three models.

# Automation was achieved by systematically varying parameters within a predefined range and selecting the best configuration based on cross-validation results.

# Results

# SVM Parameter Selection

# For the "Sep" dataset:

# The best SVM parameters were {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}. These parameters were found to minimize the expected classification error.

# Cross-validation Error: 0.0416

# Test Classification Error: 0.0405

# Explanation: SVM with an RBF kernel and specific values of 'C' and 'gamma' achieved remarkable performance on the "Sep" dataset, with a minimal classification error.

# For the "Merged" dataset:

# The best SVM parameters were {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}. These parameters were determined to minimize the expected classification error.

# Cross-validation Error: 0.2332

# Test Classification Error: 0.241

# Explanation: SVM with an RBF kernel, 'C', and 'gamma' values optimized for this dataset demonstrated strong performance, although with a slightly higher classification error compared to the "Sep" dataset.

# MLP Parameter Selection

# For the "Sep" dataset:

# The best MLP parameters were {'neurons': 32, 'epochs': 20}. These parameters were selected to minimize the expected classification error.

# Cross-validation Error: 0.3631

# Test Classification Error: 0.5195

# Explanation: MLP with 32 neurons in the hidden layer and training for 20 epochs showed competitive performance on the "Sep" dataset but had a higher classification error compared to SVM.

# For the "Merged" dataset:

# The best MLP parameters were {'neurons': 32, 'epochs': 20}. These parameters were chosen to minimize the expected classification error.

# Cross-validation Error: 0.4388

# Test Classification Error: 0.4415

# Explanation: Similar to the "Sep" dataset, MLP with 32 neurons and 20 training epochs performed competitively, but the classification error was slightly higher than SVM.

# MLR (Logistic Regression) Parameter Selection

# For the "Sep" dataset:

# Logistic Regression exhibited:

# Cross-validation Accuracy: 0.4525

# Cross-validation Error: 0.5475

# Test Accuracy: 0.4665

# Test Error: 0.5335

# Explanation: Logistic Regression provided a reasonable but comparatively lower accuracy than SVM and MLP on the "Sep" dataset.

# For the "Merged" dataset:

# Logistic Regression exhibited:

# Cross-validation Accuracy: 0.5498

# Cross-validation Error: 0.4503

# Test Accuracy: 0.5585

# Test Error: 0.4415

# Explanation: Logistic Regression performed relatively better on the "Merged" dataset but still lagged behind SVM and MLP in terms of accuracy.

# Conclusion

# SVM with optimized hyperparameters demonstrated exceptional performance on both datasets, achieving low classification errors in both cross-validation and on the test set.

# MLP, while competitive, demonstrated higher classification errors compared to SVM, indicating the importance of careful parameter selection in neural networks.

# Logistic Regression consistently lagged behind SVM and MLP in terms of classification accuracy, suggesting limited capacity to capture complex patterns.

# Part 2.2: -

Evaluate the performance of two classification models: Support Vector Machine (SVM) and PyTorch Multi-Layer Perceptron (MLP) on multiple datasets. The evaluation includes the analysis of confusion matrices and accuracy scores to assess the effectiveness of these models in different scenarios.

**Methodology**

**Model Selection**

* **Support Vector Machine (SVM)**: A widely used classification model known for its effectiveness in separating data using hyperplanes.
* **PyTorch Multi-Layer Perceptron (MLP)**: A neural network-based model for complex pattern recognition.

**Datasets**

* **Separable Dataset**: A dataset where SVM should perform well due to clear class separation.
* **Merged Dataset**: A dataset with overlapping classes, posing a challenge for both models.
* **Test Dataset**: A dataset for general testing.
* **Gender Dataset**: A real-world dataset for gender prediction.
* **Bank Dataset**: A financial dataset for binary classification.

**Evaluation Metrics**

* **Confusion Matrix**: A table used to evaluate the performance of a classification algorithm. It presents a clear view of true positives, true negatives, false positives, and false negatives.
* **Accuracy Score**: A measure of a model's performance, calculated as the ratio of correctly predicted instances to the total instances.

**Results and Discussion**

**Separable Dataset**

**A screen shot of a computer screen

Description automatically generated**

SVM

* Confusion Matrix:

luaCopy code

[[1051 0] [ 949 0]]

* SVM achieves an accuracy of 60% on this dataset, as shown by the confusion matrix.

PyTorch MLP

* Accuracy: 56%
* The MLP model, while competitive, demonstrates a slightly lower accuracy compared to SVM.

**Merged Dataset**

**A screen shot of a computer screen

Description automatically generated**

SVM

* Confusion Matrix:

luaCopy code

[[1117 0] [ 883 0]]

* SVM achieves an accuracy of 56% on the merged dataset.

PyTorch MLP

* Accuracy: 50%
* The MLP model performs less effectively on the merged dataset, showing a lower accuracy compared to SVM.

**Test Dataset**

**A screen shot of a computer

Description automatically generated**

SVM

* Confusion Matrix:

luaCopy code

[[1007 0] [ 993 0]]

* SVM's accuracy is 50%, which indicates a balanced but ineffective classification.

PyTorch MLP

* Accuracy: 50%
* The MLP model shows similar performance to SVM on the test dataset.

**Gender Dataset**

**A screen shot of a computer

Description automatically generated**

SVM

* Confusion Matrix:

luaCopy code

[[484 18] [ 22 477]]

* SVM achieves an impressive accuracy of 94% in gender prediction.

PyTorch MLP

A screenshot of a computer

Description automatically generated

* Accuracy: 94%
* The MLP model also excels in gender prediction, matching SVM's accuracy.

**Bank Dataset**

**A screen shot of a computer screen

Description automatically generated**

SVM

* Confusion Matrix:

luaCopy code

[[7146 157] [ 639 296]]

* SVM demonstrates an accuracy of 90% on the bank dataset.

PyTorch MLP

A screenshot of a computer

Description automatically generated

* Accuracy: 90%
* The MLP model matches SVM's performance with an 90% accuracy score.

**Conclusion**

* SVM demonstrates robust performance on most datasets, particularly excelling on the gender dataset.
* PyTorch MLP, while competitive, sometimes lags behind SVM in terms of accuracy.
* Careful selection of models and parameter tuning is essential for effective classification.
* The SVM confusion matrices for the separable, merged, and test datasets indicate a model that perfectly predicts one class but fails to identify the other, which could be a sign of a model that always predicts the majority class.
* The PyTorch MLP accuracy scores vary across datasets, from 53% to 90%, which suggests different levels of model suitability for each dataset.
* The gender and bank dataset evaluations show better balance and accuracy, indicating that the SVM and PyTorch MLP models are more effective on these datasets.