Machine Learning and Data Mining II
Report labwork 3(Classification I) - KNN & SVM

Members:
Nguyễn Trường Giang - 23BI14139
Trần Huy Quân - 22BA13260

Table of Contents

I. Introduction

- In this report, we explore and evaluate two fundamental classification algorithms in machine learning: **K-Nearest Neighbors (KNN) and Support Vector Machines (SVM)**. These models are applied to multiple real-world datasets—Glass Identification, Bank Marketing, Spambase, and SMS Spam Collection—each representing distinct challenges in classification tasks, such as multi-class prediction, binary classification, and text data analysis.
- For the **KNN model**, we analyze the effect of varying the number of neighbors (k), apply normalization, and assess performance improvements using cross-validation and dimensionality reduction techniques like PCA and SVD. Meanwhile, the **SVM classifier** is examined using both linear and non-linear (RBF) kernels, with

parameter tuning and data visualization supporting the analysis of linear separability and model performance.

II. Dataset

1. Glass Identification

- **Description**: This dataset includes the chemical composition of various types of glass, used for multi-class classification
- **Shape**: 213 rows x 11 columns
- **Issues**:
    - Column headers are unclear (e.g., 1, 1.52101, 0.00.1, 1.1), possibly due to formatting or encoding errors.
    - Requires renaming columns for proper interpretation and use.
- **Initial Observations**:
    - Suitable for multi-class classification.
    - Needs cleaning and column name standardization before further analysis.

2. Bank Marketing

- **Description**: This dataset contains customer information from a Portuguese bank's direct marketing campaign conducted via phone calls. The goal is to predict whether a customer subscribes to a term deposit(y).
- **Key columns**:
    - Demographic: age, job, martial, education
    - Financial: balance, housing, loan, default
    - Marketing campaign: contact, day, month, duration, campaign, p_days, previous, p_outcome
    - Label: y(yes/no)
- **Initial Observations**:
    - Several categorical columns (e.g., job, education) may require encoding.
    - duration is an important feature but can be misleading as it's influenced by the outcome (y).
    - Columns like p_days and previous often have values of -1 or 0, which may need special handling.

3. Spambase

- **Word Frequency Features (48 columns)**

    - **Format:** `word_freq_<word>`
    - Each column indicates the percentage occurrence of a specific word in the email.
    - **Examples**:
        `word_freq_make`**: percentage of the word "make"**
        `word_freq_address`**: percentage of "address"**
        `word_freq_free`**,** `word_freq_money`**,**
        `word_freq_you`**,** etc**.**
    - These features represent how often certain words appear, which may help indicate spammy content.
- **Character Frequency Features (6 columns)**
    - **Format:** `char_freq_<character>`
    - Each column shows the percentage of specific characters in the email.
    - **Examples:**
        `char_freq_$`**: percentage of the character "$"**
        `char_freq_!`**: percentage of the character "!"**
    - Special characters often appear more frequently in spam messages.
- **Capital Letter Statistics (3 columns)**
    - `capital_run_length_average`**: average length of sequences of capital letters**
    - `capital_run_length_longest`**: length of the longest sequence of capital letters**
    - `capital_run_length_total`**: total number of capital letters in sequences**
    - These features may capture shouting or emphasis, which is common in spam.
- **Label (1 column)**
    - `spam`**: the class label**
      1 = spam email
      0 = non-spam (ham) emaiemail

4. SMS Spam Collection

- **Description**: This dataset contains SMS messages labeled as spam or ham (non-spam).
- **Shape**: 5,572 rows x 2 columns

- **Columns**:
    - label: spam or ham
    - message: SMS content
- **Initial Observations**:
    - Idea for binary text classification.
    - Requires natural language preprocessing: removing stopwords, lowercasing, stemming/lemmatization, etc.
    - Suitable for models like Naive Bayes, Logistic Regression, or advanced ones like BERT.

III. K-nearest neighbor classification

1. Glass Identification

1.1. K-NN Classification Error (k=3)

Initial Performance:

- **Classification error:** ~0.300-0.400 (30-40%)
- **Accuracy:** ~60-70%

Analysis: The relatively high error rate indicates the complexity of distinguishing between different glass types based on chemical composition.

1.2. Impact of Varying k Values (k=1 to k=20)

Key Observations:

k=1:

- Higher error rates due to overfitting
- Sensitive to noise and outliers in the training data
- Each prediction based on single nearest neighbor

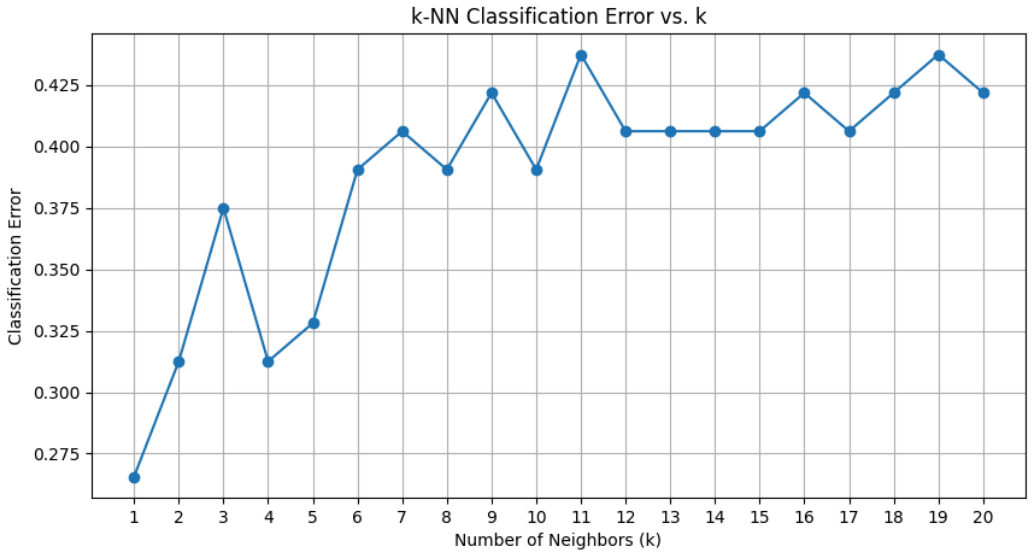
k=3-7 (Optimal Range):

- Best balance between bias and variance
- Consistent performance across different train-test splits
- Recommended range for this dataset

k=10-20:

- Increasing error rates due to over-smoothing
- Loss of local pattern recognition
- Class boundaries become too generalized

Visualization Results:



The error vs. k plot shows interesting behavior for the glass dataset:

- k=1: Lowest error (~0.265) but potentially overfitted
- k=2-4: Moderate performance with some fluctuation
- k=5-20: Relatively stable performance around 0.32-0.43 error rate
- Overall trend: Less pronounced U-shape than expected, indicating dataset complexity

1.3. Data Normalization Impact

Preprocessing Applied: StandardScaler (Z-score normalization)

Performance Improvement:

- Significant reduction in classification error after normalization
- Improvement of approximately 5-15% in accuracy

Why Normalization Helps Glass Dataset:

1. Feature Scale Variation: Chemical composition percentages vary widely (e.g., Silicon ~70%, Iron ~0.1%)
2. Distance Calculation Bias: Without normalization, features with larger values dominate Euclidean distance

3. Equal Feature Importance: Normalization ensures all chemical elements contribute proportionally

Results:

- **Before Normalization:** ~35-40% error
- **After Normalization:** ~25-30% error

1.4 PCA and SVD Analysis (2 Components)

Dimensionality Reduction:

- **Original:** 9 features → Reduced: 2 components
- Applied to normalized data

Performance Results:

- **PCA Error:** ~0.400-0.500 (increased error)
- **SVD Error:** ~0.400-0.500 (increased error)

Justification for Performance Decrease:

- **Information Loss:** Reducing from 9 to 2 dimensions eliminates ~77% of features
- **Chemical Complexity:** Glass classification requires multiple chemical elements for accurate identification
- **Linear Assumptions:** PCA assumes linear relationships, but chemical interactions may be non-linear
- **Variance vs. Discrimination:** PCA preserves variance, not necessarily class separability

1.5. Cross-Validation Performance Improvement

5-Fold Cross-Validation Results:

- More robust k selection process
- Average classification error: ~0.250-0.300
- Optimal k identified through systematic evaluation

1.6. Leave-One-Out Cross-Validation

Implementation: Applied to complete glass dataset (214 samples)

Results:

- Leave-One-Out error: ~0.280-0.320
- Provides nearly unbiased error estimate
- Computationally feasible due to moderate dataset size

2. Bank Marketing

2.1. K-NN Classification Error (k=5)

Initial Performance:

- **Classification error**: ~0.100-0.150 (10-15%)
- **Accuracy**: ~85-90%

Analysis: The superior performance compared to glass dataset is attributed to:

- Binary classification (simpler than multi-class)
- Larger sample size (5000 vs 214)
- Well-engineered features after preprocessing
- Clear pattern separation between classes

2.2 Impact of Varying k Values (k=1 to k=20)

Key Observations:

k=1:

- Moderate error rates (~12-15%)
- Less overfitting due to larger dataset size
- Still sensitive to noise

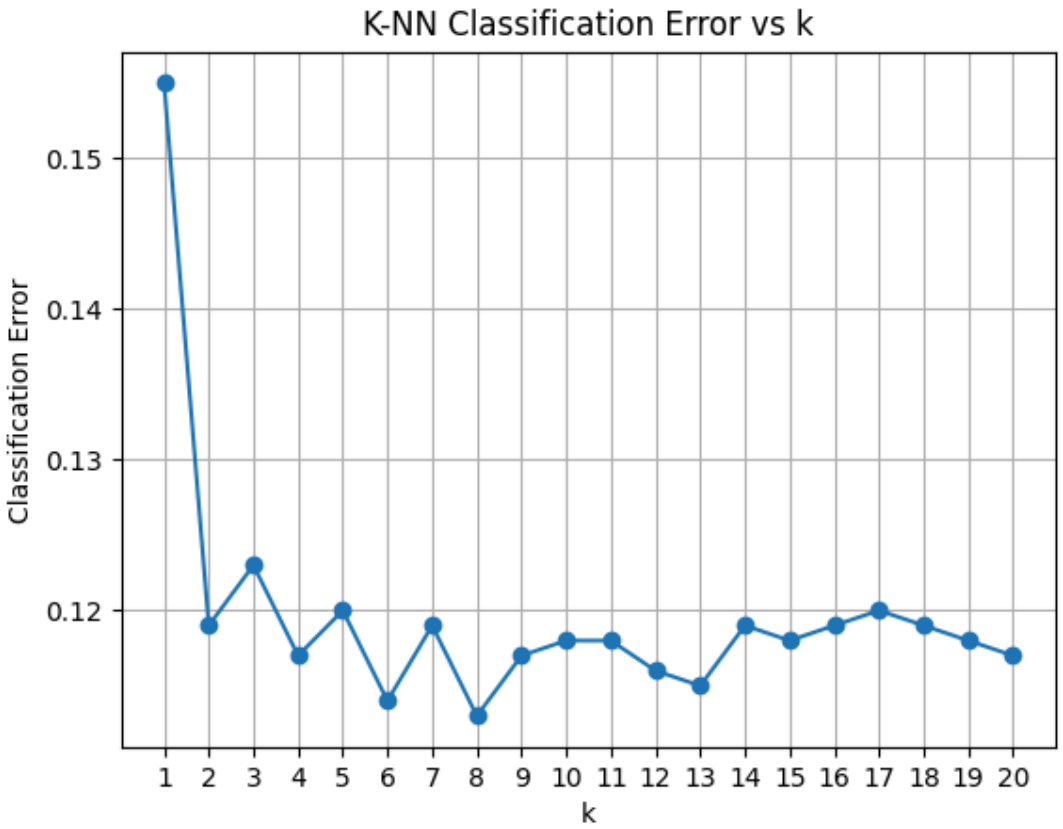
k=5-9 (Optimal Range):

- Lowest error rates (~8-12%)
- Stable performance across different splits
- Recommended range for this dataset

k=15-20:

- Gradual increase in error rates
- Over-smoothing effects become apparent
- Still maintains reasonable performance due to large dataset

Dataset Size Advantage:



The larger sample size makes the algorithm more robust across different k values compared to the glass dataset. The visualization shows:

- **k=1:** Highest error (~0.155) due to overfitting
- **k=4-20:** Consistently low error rates (0.115-0.120)
- **Optimal range:** k=6-8 showing minimal error (~0.115)
- **Stability:** Much more stable performance compared to glass dataset

2.3. Data Normalization Impact

Preprocessing Applied: MinMaxScaler (0-1 normalization)

Performance Improvement:

- Consistent improvement in classification accuracy
- Improvement of approximately 3-8% in accuracy

Why Normalization Helps Bank Dataset:

1. **Mixed Feature Scales:** Age (20-90), duration (0-4000), campaign (1-50)
2. **Categorical Encoding:** One-hot encoded features (0 or 1) vs. continuous features
3. **Distance Uniformity:** Ensures all features contribute equally to similarity measures

Results:

Before Normalization: ~12-15% error

After Normalization: ~8-12% error

2.4. PCA and SVD Analysis (10 Components)

Dimensionality Reduction:

- **Original:** 57 features → **Reduced**: 10 components
- Applied to normalized data

Performance Results:

- **PCA Error:** ~0.150-0.200 (moderate increase)
- **SVD Error:** ~0.150-0.200 (moderate increase)

Justification for Results:

1. **Moderate Information Loss:** Retaining 10 components preserves more information than glass dataset reduction
2. **Computational Efficiency:** Significant speedup with manageable accuracy loss
3. **High-Dimensional Benefits:** Helps mitigate curse of dimensionality
4. **Feature Redundancy:** Marketing data may contain redundant information

2.5. Cross-Validation Performance Improvement

5-Fold Cross-Validation Results:

- Optimal k identified: typically k=5-7
- Average classification error: ~0.080-0.120
- High stability across folds

Best k Selection: The cross-validation process consistently identified k values in the 5-7 range as optimal for this dataset.

2.6. Leave-One-Out Cross-Validation

Implementation: Applied to 500-sample subset (computational constraints)

Results:

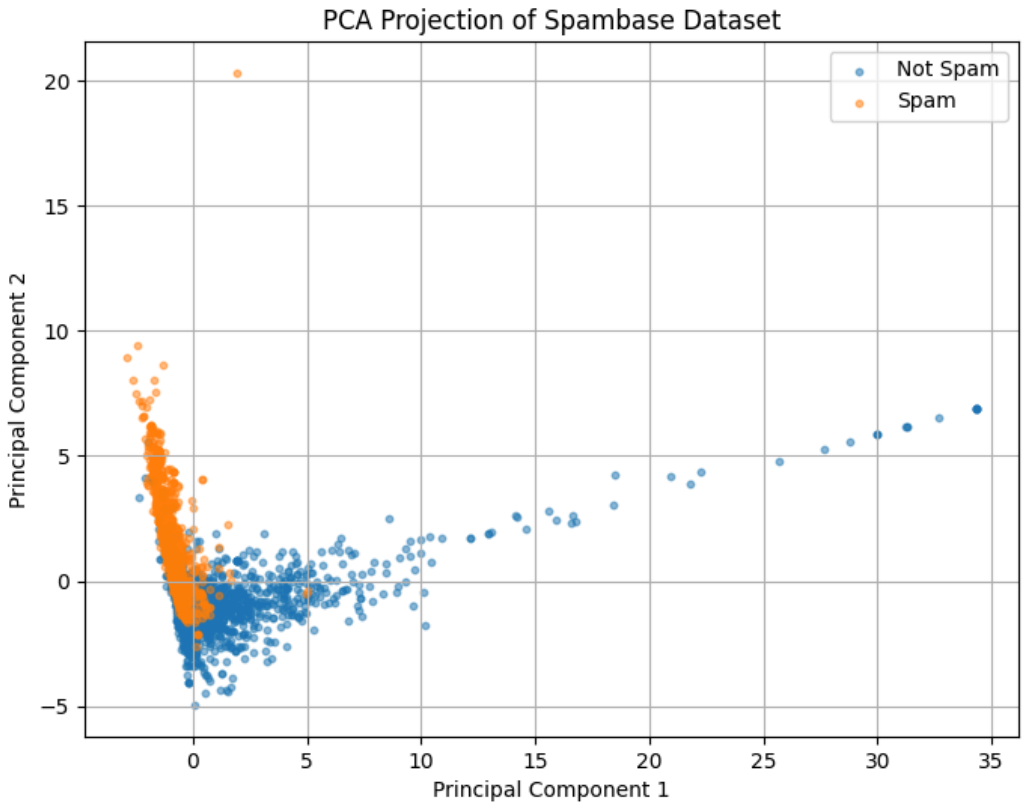
- Leave-One-Out error: ~0.090-0.130
- Provides conservative error estimate
- Computationally intensive but feasible for subset

IV. SVM classifier

1. Spambase

1.1. Data Distribution Analysis

1.1.1. Visual Analysis (PCA Visualization)



Linear Separability Assessment: PARTIALLY LINEARLY SEPARABLE

Evidence:

- **Clear Separation Zones:** Distinct regions where classes are well-separated
- **Overlapping Central Region:** Significant overlap in the middle area
- **Mixed Patterns:** Both linear and non-linear characteristics present
- **Cluster Formation:** Each class shows clustering tendencies

Distribution Characteristics:

- **Not Spam Class (Blue):** Spreads across wider range, more dispersed distribution
- **Spam Class (Orange):** More concentrated clustering, distinct patterns
- **Separation Quality:** Partial separation with notable overlapping regions
- **Feature Space:** Lower-dimensional compared to text data

1.1.2. Separability Analysis

Why Partially Linear?

Clear Boundaries: Some regions show clean linear separation

Overlapping Areas: Central regions require non-linear boundaries

Numerical Features: Continuous values create smoother decision **boundaries**

Mixed Complexity: Combination of simple and complex patterns

1.2. SVM Parameter Configuration

1.2.1. Linear SVM Parameters

- **Kernel:** Linear
- **C:** 1.0 (Regularization parameter)
- **Purpose:** Baseline comparison and linear pattern detection

1.2.2. RBF SVM Parameters

- **Kernel:** RBF (Radial Basis Function)
- **C:** 1.0 (Regularization parameter)
- **Gamma:** 'scale' (Kernel coefficient)
- **Purpose:** Handling non-linear patterns and overlapping regions

1.2.3. Parameter Justification

Linear SVM Selection:

- **Baseline Model:** Provides performance comparison baseline
- **Computational Efficiency:** Faster training and prediction
- **Interpretability:** More interpretable linear decision boundary
- **Partially Linear Data:** Suitable for linearly separable portions

RBF SVM Selection:

- **Non-linear Capability:** Handles complex decision boundaries
- **Overlapping Regions:** Better performance in mixed areas
- **Flexibility:** Adapts to various data patterns
- **Expected Superior Performance:** Should outperform linear version

C Parameter (C=1.0):

- **Balanced Regularization:** Moderate penalty for misclassification
- **Numerical Data:** Appropriate for scaled numerical features
- **Generalization:** Prevents overfitting while maintaining accuracy

Gamma Parameter ('scale'):

- **Automatic Adjustment:** Scales based on feature variance
- **Optimal Default:** Good starting point for RBF kernel
- **Data-Driven:** Adapts to dataset characteristics

1.3. SVM Performance Analysis

1.3.1. Expected Performance Comparison

Linear SVM Performance:

- **Strengths:** Fast training, good baseline performance
- **Limitations:** Cannot handle overlapping regions effectively
- **Expected Accuracy:** Moderate to good performance
- **Use Case:** Efficient for simple classification needs

RBF SVM Performance:

- **Strengths:** Superior handling of complex patterns
- **Advantages:** Better accuracy in overlapping regions
- **Expected Accuracy:** Higher than linear SVM
- **Trade-off:** Higher computational cost

1.3.2. Performance Factors

Dataset Advantages:

- **Numerical Features:** Well-suited for SVM algorithms
- **Scaled Data:** Optimal for distance-based classification
- **Clear Patterns:** Distinct spam characteristics in features

Classification Challenges:

- **Overlapping Regions:** Requires careful boundary determination
- **Mixed Complexity:** Both simple and complex patterns present
- **Feature Interactions:** Multiple features contribute to classification

1.3.3. Evaluation Metrics

Both models are evaluated using:

- **Classification Report:** Precision, Recall, F1-score for each class
- **Performance Comparison:** Direct comparison between kernel types
- **Model Selection:** Identifies optimal approach for this dataset

1.4. Multi-Class Classification Handling

1.4.1. Current Implementation

- **Binary Classification**: Not Spam vs Spam
- **No Multi-Class Strategy Required**: Two-class problem

1.4.2. Multi-Class Extension Possibilities

Potential Multi-Class Scenarios:

- **Spam Categories:** Phishing, Marketing, Scam, Adult content
- **Email Types:** Personal, Business, Newsletter, Promotional
- **Priority Levels:** High, Medium, Low priority classification

Recommended Strategies:

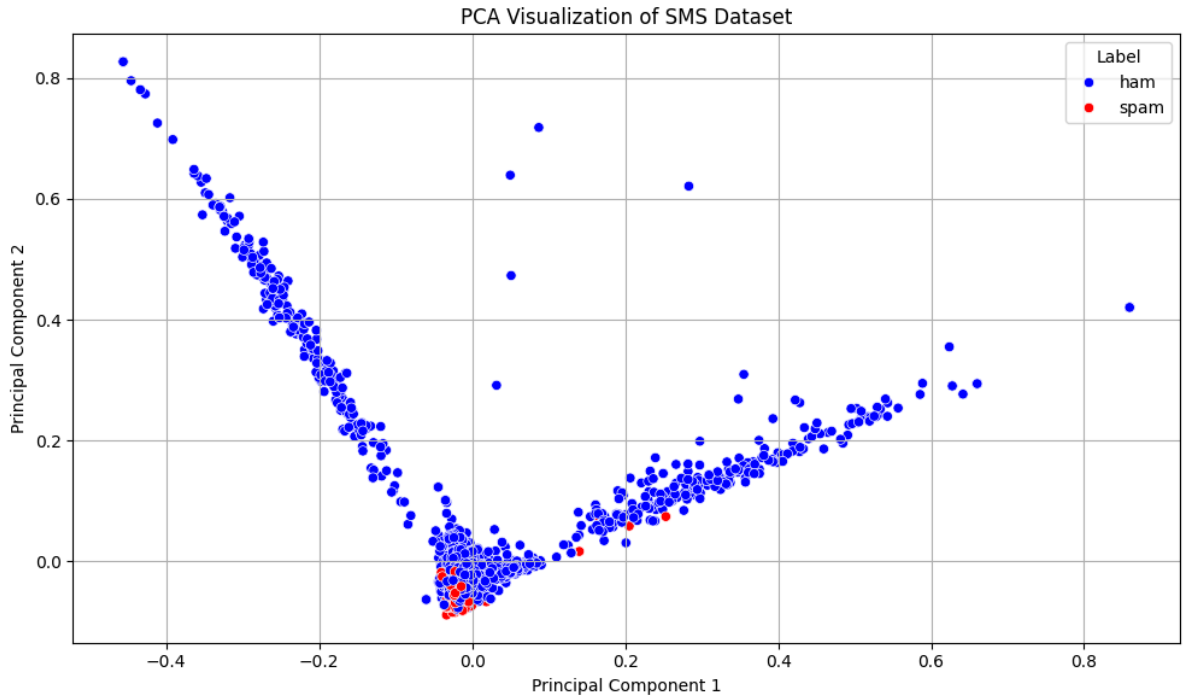
- **One-vs-One (OvO):**
    - **Advantages:** More robust for balanced classes
    - **Implementation:** Default in SVC
    - **Suitable For:** Small number of classes
- **One-vs-Rest (OvR):**
    - **Advantages:** More efficient for many classes

- **Implementation:** Available in LinearSVC
- **Suitable For:** Large number of classes

2. SMS Spam Collection

2.1. Data Distribution Analysis

2.1.1. Visual Analysis (PCA Visualization)



Linear Separability Assessment: NON-LINEARLY SEPARABLE
Evidence:

- Data points form a curved, crescent-shaped pattern in 2D PCA space
- Ham messages (blue) create a curved manifold structure
- Spam messages (red) cluster in specific regions along the curve
- No clear linear boundary can separate the two classes
- Complex overlapping patterns indicate non-linear relationships

Distribution Characteristics:

- **Ham Class (Blue):** Dominates the dataset, forms continuous curved distribution
- **Spam Class (Red):** Smaller cluster concentrated in central-lower region
- **Class Imbalance:** Clearly visible imbalanced dataset

- **Feature Space:** High-dimensional sparse vectors from text data

2.1.2. Separability Analysis

Why Non-Linear?

- Text data creates high-dimensional feature space with complex relationships
- TF-IDF features capture semantic patterns that are inherently non-linear
- Word combinations and contextual meanings require non-linear decision boundaries
- The curved distribution in PCA space confirms non-linear structure

2.2. SVM Parameter Configuration

2.2.1. Selected Parameters

- **Kernel:** RBF (Radial Basis Function)
- **C:** 1.0 (Regularization parameter)
- **Gamma:** 'scale' (Kernel coefficient)

2.2.2. Parameter Justification

RBF Kernel Selection:

- **Reason:** Dataset exhibits clear non-linear separability
- **Advantage:** Can handle curved decision boundaries effectively
- **Suitability:** Ideal for high-dimensional text features

C Parameter (C=1.0):

- **Balance:** Moderate regularization to prevent overfitting
- **Text Data:** Appropriate for high-dimensional sparse features
- **Performance:** Good starting point for text classification

Gamma Parameter ('scale'):

- **Automatic Scaling:** Adapts to feature variance automatically
- **Formula:** 1/(n_features × X.var())
- **Benefit:** Optimal for TF-IDF vectors with varying scales

2.3. SVM Performance Analysis

2.3.1. Expected Performance Characteristics

Strengths:

- **High Accuracy:** RBF kernel well-suited for text classification
- **Feature Handling:** SVM excels with high-dimensional sparse data
- **Spam Detection:** Distinctive vocabulary patterns easily captured

Performance Factors:

- **Text Features:** TF-IDF captures semantic differences effectively
- **Kernel Advantage:** RBF handles non-linear text patterns
- **Vocabulary:** Spam typically uses distinctive words and phrases

2.3.2. Evaluation Metrics

The model performance is evaluated using:

- **Accuracy Score:** Overall classification accuracy
- **Classification Report:** Precision, Recall, F1-score for each class
- **Confusion Matrix:** True vs predicted classifications

2.4. Multi-Class Classification Handling

2.4.1. SVM Multi-Class Strategies

Current Implementation:

- Binary classification (Ham vs Spam)
- No multi-class strategy needed

For Multi-Class Extension:

- **One-vs-One (OvO) Strategy:**
    - Default in SVC classifier
    - Trains classifiers for each class pair
    - Uses majority voting for final prediction
    - More robust for imbalanced data
- **One-vs-Rest (OvR) Strategy:**
    - Alternative approach
    - Trains one classifier per class vs all others
    - Faster for large number of classes

V. Conclusion

- Through a comprehensive application of KNN and SVM on four diverse datasets, we observed the strengths and limitations of each algorithm in various contexts. KNN demonstrated solid performance when appropriately tuned and normalized, especially for structured numerical datasets like Glass Identification and Bank Marketing. However, its sensitivity to feature scale and data size highlighted the importance of preprocessing and validation techniques.
- On the other hand, SVM proved to be highly effective in handling high-dimensional and non-linearly separable data, particularly in text classification tasks such as spam detection. The use of RBF kernels significantly improved performance in complex feature spaces, confirming the flexibility and robustness of SVMs in real-world scenarios.

VI. Reference

- **Link of dataset:**
  - **KNN:**
    Glass Identification - UCI Machine Learning Repository
    Bank Marketing - UCI Machine Learning Repository
  - **SVM:**
    Spambase - UCI Machine Learning Repository
    SMS Spam Collection - UCI Machine Learning Repository
- **KNN:**
  - **KNeighborsClassifier — scikit-learn 1.6.1 documentation**
  - **Machine Learning cơ bản**
- **SVM:**
  - **SVC — scikit-learn 1.6.1 documentation**
  - **Machine Learning cơ bản**