Report on the Implementation of An Improved KNN Algorithm Based on Minority Class Distribution for Imbalanced Dataset

Raghda Al taei

2024

1 Introduction

In the realm of machine learning, classification tasks are pivotal for predicting categorical labels based on training data. This report presents an implementation of the improved K-Nearest Neighbors (KNN) algorithm proposed in the paper titled ”An Improved KNN Algorithm Based on Minority Class Distribution for Imbalanced Dataset.” The aim is to assess the performance of this modified algorithm on various datasets, focusing on its effectiveness in handling imbalanced classes.

2 Problem Statement

The primary challenge addressed in this report is the classification of instances into two categories: positive (1) and negative (0) based on their features. The datasets provided consist of numerical values, with the last column indicating the class labels. The goal is to accurately predict the labels of unseen instances using the improved KNN algorithm and evaluate its performance across different datasets.

3 Methodology

3.1 Weighted K-Nearest Neighbors (WKNN)

The WKNN algorithm extends the traditional KNN approach by incorporating weights based on the distribution of minority class instances. This enables the algorithm to focus more on underrepresented classes during classification.

3.1.1 Steps of the WKNN Algorithm:

1. **Data Preparation:** Load each dataset, separate the features from the labels, and split the data into training (80%) and testing (20%) sets.

2. **Class Weights:** Calculate class weights based on the inverse frequency of each class in the training set to address any class imbalance.

3. **K-Nearest Neighbors:** For each test instance, compute the Euclidean distance to all training instances, identify the k nearest neighbors, and calculate weights for each neighbor.

4. **Weighted Voting:** Perform a weighted voting mechanism based on calculated weights to determine the predicted class label for each test instance.

5. **Accuracy Evaluation:** Compare the predicted labels with the true labels of the test set to calculate the accuracy of the classifier.

4 Implementation

The WKNN algorithm was implemented in MATLAB, following the aforementioned methodology. The following key functions were utilized in the implementation:

- **Distance Calculation:** Computed Euclidean distances between test points and training samples.

- **Sorting and Indexing:** Identified neighbors by sorting distances and selecting the nearest k neighbors.

- **Weighted Voting:** Determined classification based on weighted votes from neighbors.

4.1 Code Summary

The core implementation consists of iterating through multiple datasets, performing data loading, separation, training/testing splits, and accuracy calculations.

```
% MATLAB code snippet illustrating implementation steps
for i = 1:num_datasets
    % Load dataset
    % Separate features and labels
    % Split data into training and testing sets
    % Calculate class weights
    % Perform WKNN classification
    % Evaluate accuracy
end
```

5 Results

The accuracy results obtained from the WKNN classification across the seven datasets are as follows:

Dataset	**Accuracy (%)**
Dataset 1	97.73
Dataset 2	69.05
Dataset 3	97.62
Dataset 4	100.00
Dataset 5	71.24
Dataset 6	97.06
Dataset 7	78.04

Table 1: Accuracies for all datasets

5.1 Observations

- **High Accuracy:** Dataset 4 achieved a perfect accuracy of 100%, indicating effective classification.
- **Variable Performance:** Datasets 1, 3, and 6 showed high accuracy, exceeding 97%, suggesting suitable features for classification.
- **Moderate Accuracy:** Datasets 2, 5, and 7 exhibited lower accuracy, indicating challenges that may stem from class imbalance or data complexity.

6 Conclusion

The implementation of the improved KNN algorithm based on minority class distribution demonstrated effective classification performance across several datasets. While high accuracies were achieved for most datasets, others revealed potential areas for further investigation, such as class imbalance handling and feature engineering.

Future work may involve fine-tuning the k value and exploring additional techniques to enhance performance on more challenging datasets.