Handling Imbalanced Data with SVM and Resampling Techniques

Raghda Altaei

2024

1 Introduction

Imbalanced datasets pose significant challenges in machine learning, particularly in classification tasks. In such datasets, one class (the majority class) significantly outnumbers the other class (the minority class), which can lead to poor model performance. This document outlines a process to handle imbalanced datasets using an example code that employs Support Vector Machines (SVM) along with resampling techniques.

2 Code Explanation

2.1 Step 1: Install Required Libraries

The code begins by ensuring that necessary libraries are installed. The libraries used include:

- `scikit-learn` for machine learning algorithms.
- `pandas` for data manipulation.
- `imbalanced-learn` for handling imbalanced datasets.
- `matplotlib` for plotting.

2.2 Step 2: Import Necessary Libraries

The required libraries are imported, including:

- `pandas`, `numpy` for data manipulation.
- `train_test_split` for splitting the dataset.
- `SVC` for the SVM model.
- Metrics like `roc_auc_score`, `roc_curve`, and `accuracy_score` for evaluating the model.
- `SMOTE` for oversampling and `RandomUnderSampler` for undersampling.

2.3 Step 3: Load the Dataset

The dataset used is the Shuttle dataset, loaded from a specified URL. The columns are defined for clarity, with `Class` being the target variable.

2.4 Step 4: Preprocess the Data

In this step, the features and the target variable are separated. The features X are all columns except `Class`, while y contains the `Class` labels.

2.5 Step 5: Split the Data into Training and Test Sets

The dataset is split into training and testing sets using `train_test_split`, with stratification to maintain the class distribution in both sets.

2.6 Step 6: Resample the Data using Random Under-Sampling

Random Under-Sampling is performed to reduce the size of the majority class. This helps balance the dataset by randomly selecting samples from the majority class.

2.7 Step 7: Resample the Data using SMOTE (Oversampling)

Next, the Synthetic Minority Over-sampling Technique (SMOTE) is applied to increase the number of instances in the minority class by creating synthetic samples.

2.8 Step 8: Train an SVM Model

An SVM model is trained using the resampled dataset. The `probability=True` parameter is set to enable probability estimates.

2.9 Step 9: Make Predictions and Calculate AUC

Predictions are made, and the Area Under the Curve (AUC) score is calculated for each class. AUC is a useful metric for evaluating classifier performance, especially in multi-class settings.

2.10 Step 10: Plot the ROC Curves for Each Class

The code concludes by plotting the Receiver Operating Characteristic (ROC) curves for each class, providing a visual representation of the model's performance.

3 Imbalanced Data Issue

Imbalanced datasets can lead to several issues, including:

- **Bias Toward Majority Class:** Models may predict the majority class more frequently, leading to high accuracy but poor recall for the minority class.
- **Poor Generalization:** Models trained on imbalanced data may not generalize well to unseen data, especially for the minority class.
- **Difficulty in Learning:** Learning algorithms may struggle to identify the characteristics of the minority class due to a lack of sufficient training examples.

To address these issues, resampling techniques like undersampling and oversampling (e.g., SMOTE) are effective strategies that help balance the dataset, leading to improved model performance.

4 Conclusion

Handling imbalanced datasets is crucial for building effective classification models. The provided code demonstrates a practical approach using SVM combined with resampling techniques, allowing for better handling of such datasets. This methodology can be applied to various imbalanced datasets, such as Shuttle, Abalone, Yeast, and Churn datasets, to improve classification outcomes.