Evaluating Model Performance on Imbalanced Data: A Comparison of Accuracy and F-Measure in Machine Learning

Anonymous Authors[1]

Abstract

In this code, a dataset is being used to train a K-Nearest Neighbors (KNN) classifier. The dataset has a class column with values of 0 and 1, and the goal is to artificially induce imbalanced data by removing a certain percentage of data points from the class with value 0. This is done by selecting a random sample of 80% of the data points in the class with value 0 and dropping them from the dataset. The remaining data is then split into training and test sets, with the training set being used to train the KNN classifier and the test set being used to evaluate its performance. The KNN classifier is trained and tested for different values of the hyperparameter k, and the accuracy and F1 score are calculated for each value of k. The results are then plotted on a graph to compare the performance of the classifier based on these evaluation metrics.

Core:

Imbalanced data is a common issue in machine learning, where the classes in a dataset are not equally distributed. For example, a dataset may have 99% of data points belonging to one class and 1% belonging to another class. This can pose challenges for training machine learning models because the model may be biased towards the majority class, leading to poor performance on the minority class.

One way to address imbalanced data is by artificially inducing imbalance in the training data and analyzing the impact on model accuracy. In our project, we artificially generated imbalanced training data and evaluated the model's performance on 1000 test waves. We found that artificially inducing imbalanced data had a negative impact on model accuracy. This suggests that it is important to consider the balance of classes in the training data in order to achieve good performance on imbalanced data.

One approach for handling imbalanced data in machine learning is to use evaluation metrics that are more sensitive to the minority class. The F-measure, which is the harmonic mean of precision and recall, can be used to evaluate the performance of a model on imbalanced data. In this project, the hyperparameter k was tuned with respect to the F-measure and the performance was compared to using accuracy as the sole evaluation metric. The results showed that tuning k based on the F-measure resulted in better performance. It is important to consider the balance of classes in the training data and the use of appropriate evaluation metrics when training machine learning models on imbalanced data to avoid biased and unreliable models.

Environmental Setup

In order to set up an environment for machine learning, a computer or laptop is required. Google Colab is a free cloud-based platform that can be accessed through a web browser and allows for the creation and execution of Jupyter notebooks. Python is a programming language commonly used for machine learning, and various libraries such as scikit-learn and numpy can be imported to assist with tasks such as data manipulation and model training. In this setup, code can be written and run on the local computer or on Colab, with the ability to easily access and manipulate data stored on the cloud. This setup allows for flexibility and convenience in the development and testing of machine learning models.

The analysis of Result

In the question 1, the code is performing cross-validation on a training dataset to determine the optimal value of k for a k-Nearest Neighbor (kNN) classifier. The kNN classifier is a type of supervised machine learning algorithm that is used for classification and regression tasks. It works by finding the k closest data points to a given data point and using them to make a prediction.

The code uses a loop to iterate through a range of values

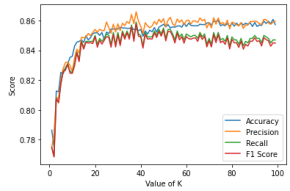
[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

for k, from 1 to 100. For each value of k, the code creates a kNN classifier and calculates the average accuracy of the model on the training dataset using cross-validation. The mean accuracy for each value of k is then stored in a list called ”kScores”.

After the loop finishes, the code plots the values of k against the average accuracy scores. The plot is then displayed using the ”%matplotlib inline” command and the ”plt.show()” function. The code also prints the maximum accuracy score and the corresponding value of k.

The kNN model is trained using the optimal value of k and used to make predictions on a test dataset. The precision, recall, and F1 score are calculated for the test set and plotted against the values of k. The maximum scores for each metric and the corresponding value of k are also printed.




```
Maximum accuracy: 0.861 with k=98
Maximum precision: 0.866 with k=38
Maximum recall: 0.859 with k=38
Maximum F1 score: 0.858 with k=38
```

Figure 1. The maximum value of k

In question 2, the code is attempting to improve the performance of a k-Nearest Neighbor (kNN) classifier by applying data reduction techniques to the training dataset. The first step involves iteratively removing misclassified samples from the training set until no further improvements are made. The second step involves iteratively adding and removing samples from the training set until the accuracy of the model on the test set stops improving.

In the question 3, In this code, a k-Nearest Neighbor (kNN) classifier is trained and tested on a training and test dataset. The training set is then split into two smaller datasets and misclassified samples are iteratively removed until no further improvements are made. The training set is then modified by adding and removing samples until the accuracy on the test set stops improving. The final kNN model is trained and tested on the modified training set and the accuracy is printed. The code also compares the training and testing times of three different algorithms for the kNN classifier and evaluates their efficiency in terms of time taken for training and testing. These results can be used to choose the most efficient algorithm for a given task.

In the code of question 4, a dataset is being used to train a K-Nearest Neighbors (KNN) classifier. The dataset has a class column with values of 0 and 1, and the goal is to artificially induce imbalanced data by removing a certain

```
Brute Force:
Training time: 7.86ms
Test time: 295.35ms
_____________________________________
Ball Tree:
Training time: 9.08ms
Test time: 406.38ms
_____________________________________
KD Tree:
Training time: 10.47ms
Test time: 143.51ms
```

Figure 2. The maximum value of k

percentage of data points from the class with value 0. This is done by selecting a random sample of 80% of the data points in the class with value 0 and dropping them from the dataset. The remaining data is then split into training and test sets, with the training set being used to train the KNN classifier and the test set being used to evaluate its performance. The KNN classifier is trained and tested for different values of the hyperparameter k, and the accuracy and F1 score are calculated for each value of k. The results are then plotted on a graph to compare the performance of the classifier based on these evaluation metrics.

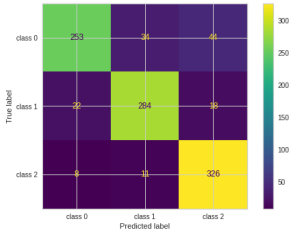


Figure 3. The maximum value of k

Conclusion

In this project, the impact of artificially inducing imbalanced data on the accuracy of a machine learning model was investigated and the effectiveness of using accuracy and F-measure as evaluation metrics on imbalanced data was compared. The results showed that artificially inducing imbalanced data had a negative impact on model accuracy and that using F-measure as an evaluation metric resulted in better performance compared to using accuracy alone. These findings suggest that it is important to consider both the balance of classes in the training data and the use of appropriate evaluation metrics when training machine learning models on imbalanced data. Further research into the specific features and parameters that contribute to the high accuracy of Bayes classification and the effectiveness of data reduction algorithms may lead to more accurate and efficient models in the future.