CMLS HW1

April 2021

1 Introduction

Our group worked on **Assignment 3**: implement a classifier system able to predict the audio effect used in recordings of electric guitar and bass. We divided our work in steps as presented in the sections of this document, from the choice of the feature through the selection of a classification method, for each phase we present the decisions we made. In general, we used as a reference the second laboratory on Classification, mainly because we implemented the Support Vector Machine with majority voting for multiclass classification.

2 Feature choice

3 Dataset selection

4 multi-class Support Vector Machine

5 Cross-validation for SVM parameters

In order to select appropriate parameters for the SVM, we tested our classifier through Cross-validation using the function GridSearchCV from scikit-learns, in doing so we were able to fine-tune the machine learning algorithm and to avoid overfitting the model. We decided also to run the grid-search only on the classifier between Tremolo and NoFX, because those were the classes that would presents more problems during the classification. The code snippet and the result is shown below, the score measures are precision and recall, while the C parameters were chosen arbitrarily within an order of magnitude:

```
tuned_parameters = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'kernel': ['rbf']},
 ]

scores = ['precision', 'recall']
```

```
for score in tqdm(scores):
    print("# Tuning hyper-parameters for %s" % score)
    print()

    clf = GridSearchCV(
        SVC(), tuned_parameters, scoring='%s_micro' % score
    )
    clf.fit(X_train12, y_train_12)

    print("Best parameters set found on development set:")
    print()
    print(clf.best_params_)
    [...]
```

Output:

```
# Tuning hyper-parameters for precision

Best parameters set found on development set:

{'C': 1000, 'kernel': 'rbf'}

Grid scores on development set:

0.759 (+/-0.121) for {'C': 1, 'kernel': 'linear'}
0.735 (+/-0.073) for {'C': 10, 'kernel': 'linear'}
0.710 (+/-0.103) for {'C': 100, 'kernel': 'linear'}
0.689 (+/-0.085) for {'C': 1000, 'kernel': 'linear'}
0.916 (+/-0.044) for {'C': 1, 'kernel': 'rbf'}
0.984 (+/-0.035) for {'C': 10, 'kernel': 'rbf'}
0.993 (+/-0.019) for {'C': 100, 'kernel': 'rbf'}
0.993 (+/-0.016) for {'C': 1000, 'kernel': 'rbf'}
[...]

# Tuning hyper-parameters for recall

Best parameters set found on development set:

{'C': 1000, 'kernel': 'rbf'}

Grid scores on development set:

0.759 (+/-0.121) for {'C': 1, 'kernel': 'linear'}
0.735 (+/-0.073) for {'C': 10, 'kernel': 'linear'}
0.710 (+/-0.103) for {'C': 100, 'kernel': 'linear'}
```

```
0.689 (+/-0.085) for {'C': 1000, 'kernel': 'linear'}
0.916 (+/-0.044) for {'C': 1, 'kernel': 'rbf'}
0.984 (+/-0.035) for {'C': 10, 'kernel': 'rbf'}
0.993 (+/-0.019) for {'C': 100, 'kernel': 'rbf'}
0.993 (+/-0.016) for {'C': 1000, 'kernel': 'rbf'}
[...]
```

6 Results