Investigating classifiers choices for Ensemble methods in audio classification

Isabelle Rodrigues Vaz de Melo
Programa de Engenharia Elétrica
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
isabelle.melo@coppe.ufrj.br

João Baptista de Oliveira e Souza Filho
Programa de Engenharia Elétrica
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
jbfilho@poli.ufrj.br

***Abstract*—In Machine Learning, Ensemble methods can be described as methods that encapsulate several classification and (or) regression models and build the final result from a combination of the predictors. A question usually raised in the classification problem is: among a set of N predictors, is there any way to select certain of these to be part of the Ensemble without having to use all the N models? The current work investigates three possible classifier selection methods for Ensemble Stacking applied to the classification problem of musical genres: F_κ-score, $F_{(1-\kappa)}$-score and the structured orthogonal matching method. pursuit (SOMP) [3] from a set of ten isolated classifiers: k-nearest neighbors, Multinomial Logistic Regression, Support Vector Machine, Random Forest, and Gradient Boosting, Multiple Layer Perceptron, Linear Discriminant Analysis and Quadratic Discriminant Analysis.**

***Index Terms*—ensemble, music genre classification, feature selection, machine learning**

I. Introduction

In several Machine learning works, in the context of classification, it is common to find a methodology in which the author tests several predictors and finds isolated performances - such as, for example, a metric that best fits the problem. Once this is done, the predictors are used to build an Ensemble [8] in search of better performances and better generalization in the problem. Ensemble methods are known as models that use several predictors to generate a final result. The three most used today are Boostrap Aggregation (Bagging) [4], Boosting [5] and Stacking [6].

Boostrap aggregation works in such a way that, from a sample of a training dataset, it is resampled by Boostrap generating a number of sub-samples. For each sub-sample, a single estimator is applied and its outputs are analyzed in each case. The voting criterion - in the case of classification - is used to decide the final winning estimation among the different models. This strategy is positive in reducing overfitting, since training the same estimator on several similar samples helps in an unbiased result.

Boosting, on the other hand, works similarly to Bagging, however with a sequential rather than parallel process. Each base classifier is sequentially trained on a sample of data. The base algorithm first assigns equal weight in the vote to each sample observation, and then the false predictions made by the previous estimator are identified. These false predictions are then assigned to the next learner with a higher weight on correct predictions.

The method that is based on the fusion of different classifiers is Stacking. Unlike Bagging, Stacking uses a combination of these different models - called weak-learners - to generate a final output as a function of a meta-model. For a sample of data, labels are generated from the weak-learners, without matching, and then these labels are used as entries in a matrix that becomes a data matrix - thus applying the meta-model to this set. data and generating a final prediction.

The current work is based on two previous researches for classifying music genres [1] [2] and aims to investigate methods that select a number n of classifiers from a total N (such that $n < N$) to be used in Stacking compared to ten classifiers already used in these works. Previously, k-nearest neighbors, Multinomial Logistic Regression, Support Vector Machine, Random Forest, Gradient Boosting, Multiple Layer Perceptron, Linear Discriminant Analysis and Quadratic Discriminant Analysis and several methods of feature selection and reduction were tested, such that the structured orthogonal matching pursuit (SOMP) [3] provided the best results for median accuracy for the classifiers. In addition, an exploratory analysis is carried out to compare the results obtained by the Stackings with those of the isolated classifiers.

This paper is organized as follows. Section II describes the classical approach to audio signal processing, section III presents a brief coverage of the Machine learning algorithms used, section IV describes the methods investigated for selecting classifiers. V discusses the system evaluation, addressing the median accuracy metric, VI performs the experiments carried out and finally the conclusion is made in section VII.

II. Audio Signal Processing

Fig. 1. Audio Signal Processing Schematic.

For audio processing in this work, the methodology presented in [1] is used, in which the raw audio signal is processed to obtain a set of features that summarize temporal information. Fig. 1 represents the approach used in the problem. First,

the raw data goes through a process of Standardisation, the signal is divided into frames in order to form a matrix of features, and then separations occur in the data that encompass segments that exhibit similar characteristics over time. These three steps are better explained as follows:

- Standardisation: The raw audio signal $s[n]$ (kHz) is standardised using the z-score (1), such that $\bar{s}$ represents the sample mean and σ_s is the standard deviation. This leads to a zero-mean and unit-variance signal.

$$\hat{s}[n] = \frac{s[n] - \bar{s}}{\sigma_s} \tag{1}$$

- Feature Extractor: The standardised signal is divided into M frames [1]. There will be N_f frame-level features, such that a frame-level feature vector $\mathbf{f_m} \in \mathbb{R}^{N_f}$ $(1 \leq m \leq M)$ is extracted from the frame m. This results in the feature matrix $\mathbf{F}_s \in \mathbb{R}^{M \times N_f}$.
- Texture Generator: In this step, it's possible to construct texture-level feature vectors. L Consecutive $\mathbf{f}_n$ are aggregated over whose dimensions are produced N_s descriptive statistics resulting in a texture matrix $\mathbf{T}_s \in \mathbb{R}^{Q \times N_t}$, wherein $N_t = N_s \cdot N_f$ and it represents the number of texture-level features. Texture-level feature vectors are represented by $\mathbf{t}_q \in \mathbb{R}^{N_t}$ such that $1 \leq q \leq Q$.

More details about frame-level features (handcrafted and derivative features) and texture-level features (high and low-order statistics) used in this work can be found at [1].

III. Machine learning classifiers

Machine learning algorithms used in this work are the same from [2] except for the Stacking learner. They can be described as follows:

- Multinomial Logistic Regression (MLR): Different from the traditional logistic regression algorithm, it predicts a multinomial probability for each input through the optimisation of the cross-entropy loss function. In this case L_2 regularization was used and parameter C (tuned in cross-validation procedure) controls regularization strength.
- k-nearest neighbours (kNN): In kNN, a test point is assigned to the most abundant class in the radius of the k nearest training neighbors that are already labeled with a certain class. In this case, distance from neighbours has been chosen to be Euclidian distance.
- Support Vector Machine (SVM): The working principle of SVM is to map data to a high-dimensional feature space so that there is categorization of data, regardless of whether the data is linearly separable or not. A separator between the classes is defined in this process and the data is transformed so that the separator can be drawn as a hyperplane. Once this is done, the new data can be used to predict which class a new test point belongs to. One-vs-one scheme was adopted and radial and linear basis function kernels were assumed. Here, C is inversely proportional to regularization strength and controlls the amount of margin violation admissible (also tuned in cross-validation).
- Gradient Boosting (GB): Gradient Boosting is a Boosting Ensemble model. Boosting Ensembles work in a sequential method so GB adds at most n decision trees sequentially applying a gradient descent procedure to minimise the logaritmic loss function. In Boosting, unlike the Bagging method, the results of each tree receive a weighting based on their classification errors, and the final result of the model takes this weighting into account.
- Random Forest (RF): Ensemble method of decision trees built over boostrap samples [11] . Each node considers a random subster of input features. Each tree model from each boostrap sample will predict a certain result, and the model outcome will be defined by the most predicted class. n maximum-depth trees were considered in this work such as Gini impurity measure.
- Linear Discriminant Analysis (LDA): This model assumes the data is distributed normally, and each of the classes has identical covariance matrices, using Baye's rules. The model outcome is defined by a linear surface decision (hyperplane) that separates classes. LDA can also be used in data pre-processing to reduce the number of features, which reduces the computing cost significantly. Hyperparameters tuned in cross-validation were solver and shrinkage parameter.
- Quadratic Discriminant Analysis (QDA): QDA is a generative model. It assumes that each class follow a Gaussian distribution, just like LDA, except it's not assumed each of the classes has identical covariances matrices. It's also based in Baye's rules. The model outcome is define by a quadratic surface decision that separates classes. Hyperparameter tuned in cross-validation was reg-param, such that it regularizes the covariance estimate.
- Gaussian Naive Bayes (GNB): Naive Bayes is also a generative model. It assumes that each class follow a Gaussian distribution. The difference between QDA and Gaussian Naive Bayes is that Naive Bayes assumes independence of the features. In this case, decision boundary is not linear because the class distributions are circular Gaussian distributions. The tuned parameter was var-smoorthing, representing the portion of the largest variance of all features that is added to variances for calculation stability.
- Multiple Layer Perceptron (MLP): It is a neural network where the mapping between inputs and output is non-linear. MLP has input and output layers, and one or more hidden layers with many neurons stacked together and it can use any arbitrary activation function. Parameters set in this case were activation function, hidden layer sizes and maximium of iterations (max-iter).
- Stacking Ensemble (Stacking): Corresponds to an algorithm that merges different classifiers. It uses base models called weak-learners in which each one, from the data sample, generates its respective outputs, and with the generated labels, the equivalent of a new dataset is formed

in which a classifier called meta-learner leads to a final result of classification. The model output depends on which model is defined to be the meta-learner.

More details about used frameworks can be found in [1] and [2].

IV. Investigated Methods

As previously discussed, the objective of this work is to investigate possible selection methods of classifiers to Ensembles, more specifically used in the work, the Stacking Ensemble. There are ten total classifiers that were used mentioned in the previous section. Here, three proposals are presented for choosing Ensemble candidates: F_κ-score, $F_{(1-\kappa)}$-score and the SOMP method, which will be discussed in sequence.

- F_κ-score: This measure proposes to evaluate together (from a harmonic mean) the accuracy measure for the multiclass scenario together with the Cohen kappa score [12], given that the kappa evaluates agreement between raters and eliminates the effect of chance coincidence in classification. Therefore, the proposal is that this could help in choosing classifiers that have better accuracy and greater kappa agreement.

$$F_\kappa = \frac{2a\kappa}{(a+\kappa)} \tag{2}$$

  Wherein a represents accuracy and κ represents Cohen Kappa score.
- $F_{(1-\kappa)}$-score: This measure proposes to evaluate together (from a harmonic mean) the accuracy measure for the multiclass scenario together with the complement of the Cohen kappa score (1-kappa), given that the kappa evaluates agreement between raters and eliminates the coincidence effect. luckily in ranking. Therefore, the proposal is that this could help to choose more diverse classifiers, given that we would be evaluating the mismatch between accuracy and kappa complement.

$$F_{(1-\kappa)} = \frac{2a(1-\kappa)}{(a+(1-\kappa))} \tag{3}$$

  Wherein a represents accuracy and κ represents Cohen kappa score.
- structured orthogonal matching pursuit (SOMP): The SOMP method is considered a feature selection method. It works based on a modified sparse linear regression that does a linear combination of the output of the dataset to predict the output. Since SOMP acts as a feature selector, it was proposed that this method be used to choose the most relevant classifiers for the problem. For that, the outputs of each classifier were generated for each song - that is, which class it belongs to - and a Label-Encoder encoding was used. This encoding allows labeling each class as an integer. Thus, it is possible to form a new data matrix in which the features are the classifiers, the elements of the rows of each column are their outputs, and the targets matrix is a binary matrix by One-Hot-encoding [9]. A possible problem with this encoding is that the regression performed by SOMP will not consider that the labels of the Label-encoder encoding classes are categorical and without numerical distinction. Therefore, the encoding for this method is still subject to future changes.

For each method, the classifiers achieving the highest results will be selected as Ensemble members.

V. System evaluation

Consider tuples of audio singal s_i with true labels given by g_i. Thus, a evaluation set of musics S can be defined as $S = \{(s_1, g_1), ..., (s_N, g_N)\}$, with $1 \leq i \leq N$. In addition, consider $\tilde{g}_i(\mathbf{t}_j)$ the predicted label for each texture-level feature vector $\mathbf{t}_j$ extracted from the audio signal, such that $1 \leq j \leq Q$. Given that, the predicted label for the audio sinal can be defined by majority voting:

$$\tilde{g}_i = mode(\tilde{g}_i(\mathbf{t}_1), \tilde{g}_i(\mathbf{t}_2), ..., \tilde{g}_i(\mathbf{t}_Q)) \tag{4}$$

Since we are working in a multiclass scenario, the best metric to evaluate for the current problem is accuracy:

$$ACC_a = \frac{1}{\mathbf{card}(S)} \sum_{s_i \in S} \mathbb{I}(\tilde{g}_i = g_i) \tag{5}$$

Where $\mathbb{I}(\cdot)$ denotes an indicator function, $\mathbf{card}(S)$ denotes the cardinality of S and ACC_a is audio-level accuracy.

VI. Experiments

The stages of the experiments performed in this work follow the same methodology as the experiments in [2], and the same base classifiers were used (k-nearest neighbors, Multinomial Logistic Regression, Support Vector Machine, Random Forest, and Gradient Boosting, Multiple Layer Perceptron, Linear Discriminant Analysis and Quadratic Discriminant Analysis).

A. Database

The Dataset used to perform the experiments was GTZAN [10]. It is a public database collected between 2000 and 2001 from recordings, radio and CDs, consisting of 1000 audio tracks of 30 seconds long equally distributed between 10 music genres (classical, country, blues, disco jazz, metal, hip-hop, pop, reggae, rock). The audio tracks are stored in WAVE format at a sampling rate of 22050Hz, 16-bits of quantisation and single-channel. Thus, the classes are balanced.

To find the best hyperparameters, a nested cross-validation of 10 folds [11] was done. In this process, 100 different combinations of training, validation and test sets are generated. Each test set represents 10% of the total songs. Therefore, the training and validation process is repeated 100 times for the different combinations of these sets, generating the accuracies and making it possible to analyze the best hyperparameters.

B. Hyperparameters Tuning

Table I shows the range of hyperparameters used for each classifier, as done in [2]. In the validation procedure, the ACC_a's were generated for each parameter and those with the best performance were chosen as the best parameters to be used.

TABLE I
RANGE OF HYPERPARAMETERS INVESTIGATED PER METHOD

Method	Solver/lib	Hyperparameters	Values
LDA	'lsqr'	shrinkage	$\{0, 0.001, 0.01, 0.1, 1\}$
QDA	'svd'	reg-param	$\{0, 0.005, 0.01, 0.05, 0.1\}$
kNN	-	n-neighbors	$\{1, 3, 5, 7, 9\}$
MLR	'lbfgs'	C	$\{0.001, 0.01, 0.1, 1\}$
SVM_L	LIBLINEAR	C	$\{0.01, 0.1, 1, 10, 100\}$
SVM_R	LIBSVM	C, kernel, gamma	$\{1, 10, 100, 1000\}$,$\{'rbf'\}$, $\{'auto'\}$
RF	-	n-estimators	$\{10, 50, 100, 200\}$
GB	LightGBM	n-estimators	$\{50, 100, 200, 400\}$

C. Evaluating F_κ, $F_{(1-\kappa)}$ and SOMP

The methodology used to compute the methods is shown below.

- F_κ-score: The pre-processed dataset undergoes a boostrap re-sampling, generating 10 sub-samples, one for each classifier. For each classifier applied to the sub-samples, the accuracy and kappa metrics are generated, using the equation (2) and thus the F_κ are generated. In the experiment, stackings with 3,5 and 8 classifiers were tested, so the 8 winners were taken into account.
- $F_{(1-\kappa)}$-score: The pre-processed dataset undergoes a boostrap re-sampling, generating 10 sub-samples, one for each classifier. For each classifier applied to the sub-samples, the accuracy and kappa metrics are generated, using the equation (2) and thus the $F_{(1-\kappa)}$ are generated. In the experiment, stackings with 3,5 and 8 classifiers were tested, so the 8 winners were taken into account.
- SOMP: According to reference [3], the SOMP method depends on a weight parameter η. In the experiment, four weights were used: $\{0.01, 0.1, 1, 10\}$. The weight chosen depended on the accuracy performance of the final stacking model. In the end, the chosen one was $\eta = 10$

Having the winners results, it's possible to start constructing the stacking models.

D. Results

In this work, the data pre-processing used for feature selection and texture selection adopted will be based on the winning results found in [2]. The texture selector chosen was LIN-20 and the feature selector was SOMP with the weight $\eta = 1$. After cross-validation procedure, the best hyperparameters and respective parameters were found as shown in Table II.

TABLE II
BEST HYPERPARAMETERS AND PARAMETERS FOUND

Method	Solver/lib	Hyperparameters	Values
LDA	'lsqr'	shrinkage	0.1
QDA	'svd'	reg-param	0.1
kNN	-	n-neighbors	5
MLR	'lbfgs'	C, max-iter	0.01,10000
SVM_L	LIBLINEAR	C, max-iter	0.01, 1000
SVM_R	LIBSVM	C, kernel, gamma	1,$'rbf'$, $'auto'$
RF	-	n-estimators	50
GB	LightGBM	n-estimators, learning-rate	200, 0.1
GNB	-	var-smoothing	0.00000001
MLP	-	max-iter, hidden-layer-sizes, activation	1000, [25], 'tanh'

Fig. 2 shows boxplot results of accuracy median from test sets (isolate classifiers). Numerical results found for median accuracies, F_κ and $F_{(1-\kappa)}$ are shown in Table III. As it can be seen, Gaussian Naive Bayes has the lowest value for accuracy (around 64.0%) and SVM_R has the highest one (86.0%). The 8 winners from F_κ-score are (from best to worst): SVM_R, GB, MLR, QDA, MLP, LDA, SVM_L, KNN. The 8 winners from $F_{(1-\kappa)}$-score are (from best to worst): GNB, KNN, SVM_L RF, MLR, QDA, LDA, MLP.

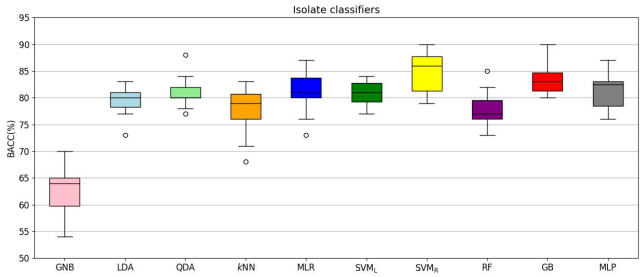

Fig. 2. Boxplot median accuracies for isolate classifiers.

TABLE III
RESULTS ACC_a, κ, F_κ, $F_{(1-\kappa)}$

Method	ACC_a	κ	F_κ	$F_{(1-\kappa)}$
LDA	80.0%	76.7%	78.3%	36.1%
QDA	80.0%	76.7%	78.3%	36.1%
kNN	79.0%	73.6%	76.2%	39.6%
MLR	81.0%	75.5%	78.9%	37.6%
SVM_L	81.0%	73.9%	77.3%	39.5%
SVM_R	86.0%	82.4%	84.2%	29.2%
RF	77.0%	74.4%	75.7%	38.4%
GB	83.0%	76.7%	78.9%	26.5%
GNB	64.0%	59.0%	61.4%	49.9%
MLP	82.5%	78.8%	79.9%	33.7%

For the SOMP method, as mentioned in subsection C, with the parameter $\eta = 10$, the following were found as the best classifiers (from best to worst): RF, GB, SVM_R, KNN , QDA, MLP, MLR and SVM_L.

It is worth to mention that even Stackings using three classifiers were taking more than twice as long as isolate classifiers.

Once that was done, it was finally possible to start building the Stacks. As mentioned before, for each method stacks of 3, 5 and 8 classifiers were tested, with each stacking assuming a meta-learner, so there are 48 Stackings in total to evaluate. Each meta-learner was one of the winning classifiers in the set of estimators. The bar plots of median accuracy results for the F_κ, $F_{(1-\kappa)}$ and SOMP methods, as a function of each meta-learner, can be seen, respectively, in Fig. 3, Fig. 4 and Fig. 5, and its respective ACC_a mean values are labeled in Table IV.

TABLE IV
MEAN VALUES ACCURACY (3, 5 AND 8 CLASSIFIERS) F_κ, $F_{(1-\kappa)}$, SOMP

	3 clf	5 clf	8 clf
F_κ	81.8%	83.4%	82.8%
$F_{(1-\kappa)}$	77.2%	81.5%	81.8%
SOMP	81.2%	81.9%	82.6%

A brief analysis of Table 4 and Figs 3, 4, and 5 shows us that the values, in general, between the use of 3, 5 or 8 classifiers

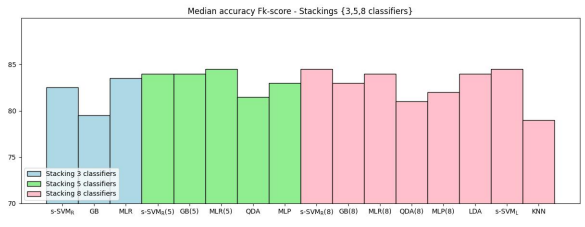

Fig. 3. Barplot median accuracy F_κ-score by meta-learner.

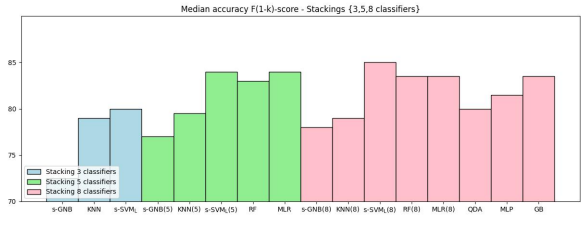

Fig. 4. Barplot median accuracy $F_{(1-\kappa)}$-score by meta-learner.

do not differ much from each other. However, for each of the tested sets, we can choose a winner based on the average of the medians of accuracy for comparative purposes at the end and delimit the best method. For the F_κ method, the winner was Stacking with 5 classifiers. For $F_{(1-k)}$ and SOMP were the Stackings of 8 classifiers.

Comparing these three results, we can see that F_κ and SOMP provide the best results, while F_{1-k} is the worse one, result that contradicts the idea that a harmonic average between accuracy and $1-k$ would generate a set of more diverse classifiers and therefore more expressive results than the average F_k.

Thus, F_k-score and SOMP are the best candidates, in this context, for possible classifier selectors to enter the Ensemble method. The absolute winner is F_k-score. Figs. 6 and 7 show, respectively, the results of mean accuracy for the winner - showing $F_{(1-k)}$ has the worse result - and results of accuracy by number of classifiers used. It can be noted that $F_{(1-k)}$ achieves high results when 8 classifiers are used, although low values can be found for 3 classifiers. SOMP and F_k are simillar and get the best values.

Fig. 8 shows a comparsion between the absolute winner (F_k) and isolate classifiers. Figs. 9, 10, 11 show accuracy

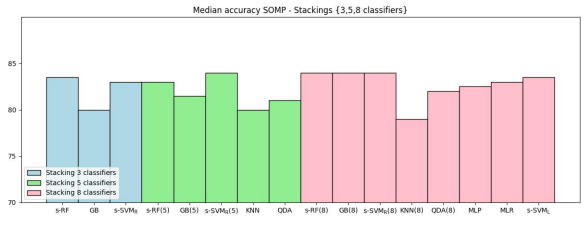

Fig. 5. Barplot median accuracy SOMP by meta-learner.

TABLE V
META-LEARNERS BY WINNER METHOD

	F_k	$F_{(1-k)}$	SOMP
LDA			
QDA	x	x	x
KNN		x	x
MLR	x	x	x
SVM_L			x
SVM_R	x	x	x
RF		x	x
GB	x	x	x
GNB		x	
MLP	x	x	x

results for each method by meta-learner.

Table V shows the meta-learners used by the winning method.

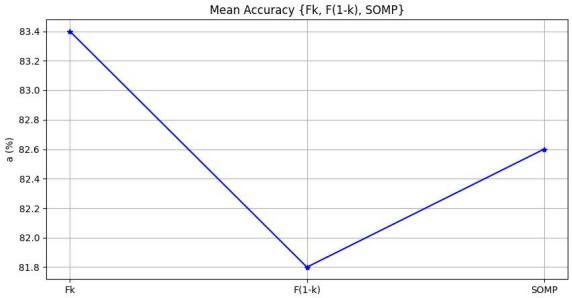

Fig. 6. Lineplot mean accuracy winner stackings.

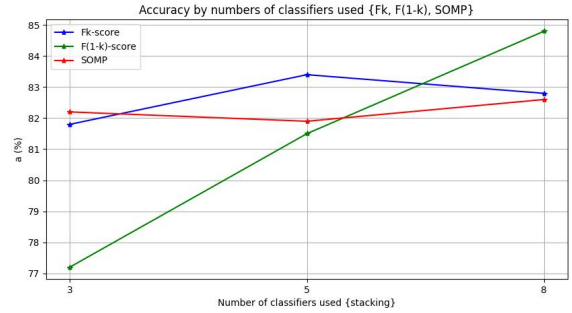

Fig. 7. Lineplot accuracy by numbers of classifiers used.

VII. CONCLUSION

Experiments with the GTZAN database made it possible to make a comparison between the investigated methods. It is possible to conclude that using the strategy of a classifier selector for Ensemble methods can be beneficial in that choosing certain classifiers based on a well-established criterion provides results as good as, or even better than those found for most of the isolated estimators tested. , paying a relatively lower computational price.

In this case the winners were the F_k method and the SOMP. It is important to note that even though they obtained above-average accuracy results for isolated classifiers, the Stackings

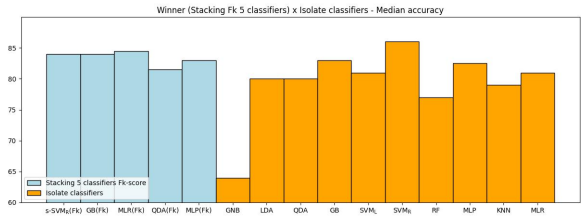

Fig. 8. Winner method (F_k) accuracies compared with isolate classifiers.

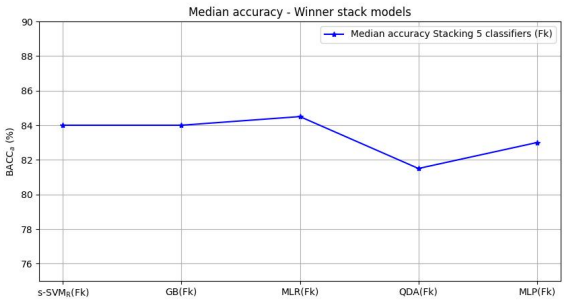

Fig. 9. Lineplot median accuracy by meta-learner (F_k).

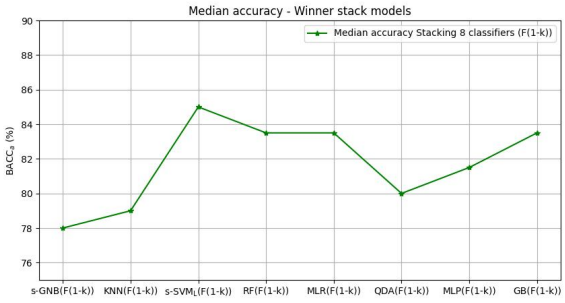

Fig. 10. Lineplot median accuracy by meta-learner ($F_{(1-k)}$).

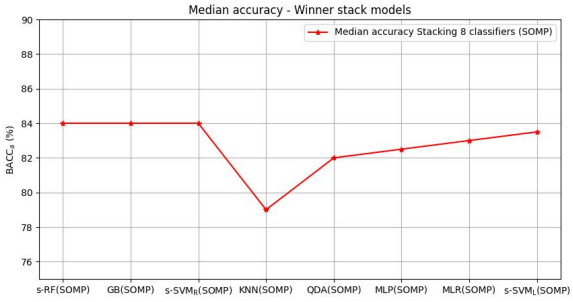

Fig. 11. Lineplot median accuracy by meta-learner (SOMP).

were still unable to reach the value found for the SVM method with Radial kernel (86%). This can happen in the Stacking method since it is not a rule that it will always perform better than a given classifier, but rather that it performs as well as a set of those tested.

An interesting analysis is that the results for $F_{(1-k)}$ were expected to obtain a better performance compared to those of F_k, since the first would represent a measure of mismatch between accuracy and complementary to the kappa score, which may result in more diverse classifiers. This did not happen, as $F_{(1-k)}$ had the worst performance. These results show that intuition is not always correct. Despite this, the values found for $F_{(1-k)}$ were not bad compared on average to the ten classifiers tested.

Another result worth mentioning is that even though there was a need to make changes to the SOMP coding, it had the second best performance and an overall average accuracy of 82.2%. The global average for the ten classifiers was 79.4%, and despite the difference not being statistically significant, it was still a result that performed above expectations. In future works, we intend to investigate possible encodings that correct the Label-encoder so that the regression presented by SOMP does not make this distinction between numerical variables that represent classes.

In Table V, it can be noted that MLR, MLP, GB and SVM$_R$ are classifiers in common for all three methods. This reflects the fact that these alone lead to the best accuracy performances.

REFERENCES

[1] V. H. Da Silva Muniz, J. B. de Oliveira and S. Filho, "Feature Vector Design for Music Genre Classification," 2021 IEEE Latin American Conference on Computational Intelligence (LA-CCI), 2021, pp. 1-6, doi: 10.1109/LA-CCI48322.2021.9769848.

[2] V. H. Da Silva Muniz, J. B. de Oliveira and S. Filho, "Robust Handcrafted Features for Music Genre Classification".

[3] Xiaoshuang Shi a , Fuyong Xing b , Zhenhua Guo d, Hai Sua , Fujun Liuc , Lin Yanga,c,, "Structured orthogonal matching pursuit for feature selection", 2019, ISSN 0925-2312 .

[4] Breiman, L. Bagging predictors. Mach Learn 24, 123–140 (1996). https://doi.org/10.1007/BF00058655

[5] Schapire, R.E. (2003). The Boosting Approach to Machine Learning: An Overview. In: Denison, D.D., Hansen, M.H., Holmes, C.C., Mallick, B., Yu, B. (eds) Nonlinear Estimation and Classification. Lecture Notes in Statistics, vol 171. Springer, New York, NY. https://doi.org/10.1007/978-0-387-21579-29.

[6] B. Pavlyshenko, "Using Stacking Approaches for Machine Learning Models," 2018 IEEE Second International Conference on Data Stream Mining Processing (DSMP), 2018, pp. 255-258, doi: 10.1109/DSMP.2018.8478522.

[7] Dietterich, T.G. (2000). Ensemble Methods in Machine Learning. In: Multiple Classifier Systems. MCS 2000. Lecture Notes in Computer Science, vol 1857. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45014-91

[8] Al-shehari, T.; Alsowail, R.A. An Insider Data Leakage Detection Using One-hot Encoding, Synthetic Minority Oversampling andMachine Learning Techniques. Entropy 2021,23, 1258.

[9] Sturm, B. L. (2012). An Analysis of the GTZAN Music Genre Dataset. In Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies (Vol. 2012, pp. 7- 12). Association for Computing Machinery. ACM Multimedia https://doi.org/10.1145/2390848.2390851.

[10] N. Japkowicz and M. Shah, Evaluating Learning Algorithms: A Classification Perspective. Cambridge University Press, 2011.

[11] J. Chen, P. He, and F. Pan, “A preferred music recommendation method for tinnitus personalized treatment based on signal processing and random forest,” in 2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP), pp. 470–473, 2021.
[12] Banerjee, Mousumi Capozzoli, Michelle Mcsweeney, Laura Sinha, Debajyoti. (2008). Beyond Kappa: A Review of Interrater Agreement Measures. Canadian Journal of Statistics. 27. 3 - 23. 10.2307/3315487.