Competition - Music classification

Student: Gaiceanu Theodora

Investigating the data

- the shape and some samples of the data were analyzed
- a summary of the training data (using function *info()* from pandas) was printed in order to get the names of the features and their type
- the values of the features (I plotted a histogram for the training data) were analyzed
- the histograms for the liked and disliked songs were plotted using different colours
- by analyzing the histograms, it can be observed that the data from all features is clearly delimited for the liked and disliked songs
- therefore, I considered all features for the classification

Preprocessing Data

- the data was splitted into features and labels
- the training and testing features were scaled (because scaling is necessary for certain models, like KNN Neighbours. for example)
- the chosen scaler was StandardScaler (the results were better than the ones when using MinMaxScaler)
- StandardScaler implies normalizing the features so that mean = 0 and variance = 1
- the inputs were considered as qualitative
- the data was splitted into training and testing because splitting the data reduces overfitting and gives better performances
- 33% of the data was used for testing (this ratio seemed to give the best results; the other considered ratios were 20%, 30%)

Considered methods

My idea was to consider multiple methods that are suitable for classification and to compare their performances in the end.

The methods are:

- ★ KNN Neighbours
- ★ Decision Tree
- ★ Random Forest
- ★ Logistic Regression
- ★ Bagging
- ★ Boosting
- ★ SVM

KNN Neighbours

- Short description
  - normalization of the input data is very important [1]
  - it is a distance method
  - the choice of k (numbers of neighbours) is very important - small k leads to overfitting, large k leads to underfitting
- Steps in a KNN algorithm [1]:
  - calculate the euclidean distance for all training data points
  - define the set N* = {i : xi being one of the k data points closest to the test input x*}
  - compute the prediction as the majority vote from {yj: j is from N*}
- Tuning:
  - k = 5 in this case (I was experimentally testing values and checking the cross-validation score; I noticed that the cross-validation score decreased when I used values bigger than 5)
  - this parameters was chosen after trying several values and comparing the accuracy using cross-validation

Decision Tree

- Short description
  - it is a rule-based method
  - the input space (the features) are divided into several disjoint regions [1]
  - a threshold is used in every region in order to predict the output [1]
  - the problem is that an unconstrained decision tree overfits (the variance is high) [2]
  - so one should put a constraint over the max depth (large max depth - overfitting; small max depth - underfitting)
- Tuning:
  - decrease the max_depth (the maximal depth of the tree) so the model doesn't overfit
  - I noticed that the score using cross-validation didn't improve for max_depth > 2
  - consequently, the chosen max_depth was 2
  - this parameters was chosen after trying several values and comparing the accuracy using cross-validation

Random Forest

- Short description
  - a way to reduce the correlation between the members of an ensemble (a model that uses several base models) [1]
  - consists of multiple decision trees
  - some randomness is introduced in each decision tree
  - in this way, the correlation between each tree is reduced
  - when splitting nodes in a tree, only a certain part of the input data points are considered (usually, for classification the number is q = sqrt(p), where p is the number of inputs) [2]
- Tuning:
  - increase the number of estimators (number of trees) in order to improve the accuracy - best results with 1000 estimators
  - reduce the depth of each tree in order to avoid overfitting - best result for depth 5
  - these parameters were chosen after trying several values and comparing the accuracy using cross-validation

Random Forest

- As it is known, the Random Forest model has also the advantage that one can get the feature importance
- After observing the feature importance, I tried to select for training only the features that had the importance above 0.05 (danceability, duration, instrumentalness, mode, tempo), but the accuracy of all models decreased

	Features	Feature Importance
0	danceability	0.142228
1	duration	0.097256
2	energy	0.051950
3	instrumentalness	0.133178
4	key	0.027901
5	liveness	0.017621
6	loudness	0.030809
7	mode	0.171398
8	speechiness	0.003237
9	tempo	0.245762
10	time_signature	0.034497
11	valence	0.004093
12	label	0.040069

Fig. 1 Feature importance

Logistic Regression

- Short description
  - a linear regression model modified in order to be suitable for classification problems [1]
  - construct the linear regression model, then map it in order to output values only within the interval [0,1]; use sigmoid function for the mapping
- Tuning:
  - increase the max_iterations in order to avoid underfitting
  - decrease the tolerance (an error of 1e-3 is also acceptable and it may reduce computational effort)
  - the values of parameters above were chosen after considering several values and analysing the model accuracy with cross-validation

Bagging (using a Decision Tree Classifier)

- Short description
  - eliminates the downside of Decision Trees, by reducing the variance in the model [1], [2]
  - uses several base models which are learned from distinct training sets and averages over the models [2]
  - learn the models in parallel
  - the distinct data sets are formed using the bootstrap method (implies sampling with replacement)
- Tuning:
  - increase the number of estimators n_estimators in order to obtain a more flexible model (and a better accuracy) - best results with 55 estimators
  - increase the number of features used for drawing from X to train each base estimator (may increase the accuracy) - best results with 13 features (all the features, actually)
  - the values of parameters above were chosen after considering several values and analysing the model accuracy with cross-validation

Boosting (Gradient Boosting)

- Short description
  - uses models that are individually weak and combines them, the result being a better model, able to make good predictions [1]
  - reduces bias
  - learns the models sequentially, so that each model improves the previous one
  - the learning method is very similar to gradient descent method [2]
- Tuning:
  - increase the number of features (max_features) considered when looking for the best split - best value was 6
  - limit the number of nodes in each three by decreasing the max_depth parameter - best value was 3
  - increase the number of boosting stages - n_estimators- (as the gradient boosting doesn't overfit easily and a bigger number of estimators would lead to a better accuracy) - best value was 50
  - the values of parameters above were chosen after considering several values and analysing the model accuracy with cross-validation

SVM (SVC)

- Short description
  - a version of logistic regression that uses the Hinge Loss [2]
  - comparing to logistic regression, the Hinge Loss function equals to 0 when the margin is >= 1 [1]
  - the prediction is dependent only on a few data points (the support vectors) [1]
  - suitable for binary classification, but not for multi classification problems
- Tuning:
  - the model actually performed better without any tuning

Performances - model accuracy

Model	Cross-validation score
KNN Neighbours	0.754
Decision Tree	0.790
Random Forest	0.834
Logistic Regression	0.795
Bagging	0.850
Boosting	0.838
SVM	0.775

Conclusions

- In general, the ensemble models worked better than the individual models. But this was expected, as the ensemble models eliminate the downsides of the individual models (high variance and bias).
- The best accuracy was obtained with the Bagging model (0.85%). But the Boosting and the Random Forest Classifier models also gave similar accuracy.
- Choosing the minimal amount of training features is very important and a small number of features decreases the accuracy of the model.
- The train/test ratio also influences the accuracy of the model.

References

[1] Lindholm A., Wahlstrom N., Lindsten F., Schon T. B. (2021), Machine Learning - A First Course for Engineers and Scientists

[2] Bernhardsson B. (2021), Lecture notes - Modeling and Learning from Data