Final report CSC529

The Harmony Analysis of Bach

Luke

1349324

Descriptions:

Beside by the melody, the Western tonal Music is strongly determined by its chords and chord progressions, key, and bassline. For example, a minor chord often evokes a more melancholic mood, while a major chord may sound more positive. Furthermore, various genres will make use of different types of chord progressions of different complexities (e.g Jazz).

Why?

Whenever we talk to somebody that we used to learn some sorts of instrument, we basically mean, we can play 'something'. But that doesn't impress anybody if you play not that extraordinary. What impressed others is that you make them or make yourself 'sounds' better. Chords can support that opinion. When your children singing, and you can play chords to make them interested in music itself.

Many of us obtained the basic knowledge of music from our previous education, few of us would really master in music (since this is a report on the machine learning class). Machine learning can be used explained to folk as discover patterns in data, labeled by 'experts', which means teach computer think as the expert do. The report is on the Harmony analysis of Bach. How many chords are there? What are all the chords are? What are those chords subtle combination? That knowledge has to be memorized by music student, but not for most of the people.

The data is collected from: http://archive.ics.uci.edu/ml/datasets/Bach+Choral+Harmony. There are 17 attributes there in the dataset, with a number of instances of 5665. The associated task is classification. There are total 60 chorales.

The attributes Information:

1: Choral ID: corresponding to the file names from (Bach Central)

2: Event number: index(starting from1) of the event inside the chorale.

3-14: Pitch classes: YES/NO depending on whether a given pitch is present. Pitch classes/attribute correspondence is as follows:

C->3	C#/Db->4	D->5	D#/Eb->6	E->7	F->8
F3/Gb->9	G->10	G#/Ab->11	A->12	A#/Bb->13	B->14

15: Bass: Pitch class of the bass note

16: Meter: integers from 1 to 5. Lower numbers denote less accented events, higher numbers denote more accented events.

17: Chord label: Chord resonating during the given event.

As the datasets conclude many chords, here can be seen as the labels, we will selected some chords to examine classifiers. For the basic chords or most popular chords, we here select C_M, G_M, E_m, D_M. Many of hit songs are composed based on those four chords. We assume that if we can build classifiers to recognized those four chords, we can know more about the hit songs.

Related Papers:

A Study on Music Genre Recognition and Classification Techniques (Aziz Nasridinov and Young-Ho Park)

Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music (Takuya FUJISHIMA CCRMA, Stanford University Stanford CA 94305, USA)

An End-to-End Machine Learning System for Harmonic Analysis of Music (Yizhao Ni, Matt McVicar, Raul Santos-Rodriguez, and Tijl DeBie)

Using Machine-Learning Methods for Musical Style Modeling (Shlomo Dubnov Ben-Gurion University; Gerard Assayag Olivier Lartillot, Institut de Recherche et Coordination Acoustique/Musique,; Gill Bejerano, Hebrew University

Data Preprocessing:

There are total 17 attributes; we firstly delete the ID type attribute because they don't contribute for classifier. Then the Bass note in the attribute 13 is so correlated with the labels so that we would delete the bass note. Then we have 12 attributes indicating 12 different sounds from C to B.

Then we replace all the data 'Yes' to 1 indicating the button is pressed, 'No' as 0, indicating that the button is not pressed. Then we train the classifier with different note combinations.

The major chords people used to compose songs are basically C major, D major, E minor, G major. So we choose the data from those four classes. While selecting different classes, we do the balancing at the same time. There are only E minor in the whole dataset, so we will chose other three chords based on the amount of E minor. Then we finally chose __ for each chord.

For the classifier we will use later, the SVM is basically used to classify binary label. Before we fit the SVM both linear and non-linear, we have to transforming the multi-label dataset into binary dataset. When evaluating different chords we will define each specific chord into class 1, other classes as 0. Let's say that we are training classifier on predict chord C major, we select label C_M as class 1, D major, E minor, G major as class 0.

Logically, we train classifiers from easy to complicate. We orderly train the linear discriminate classifier, K-nearest neighbors, Simple decision tree; then we switch to ensemble learnings. The ensemble learners we choose are random forest, discriminant using subspace method and decision trees with AdaBoostM2 method. Last, we come to SVM in both linear and non-linear methods with both soft and hard margin. Then we would compare different classifiers to see each classifier with their specific characteristics.

The whole procedure of the report is go through each step into details in order to get familiar with different classifiers.

As all of the data is in binary type, we will not use the Naïve Bayes as there is no Gaussian distribution of the data. The data set based on its domain is different from normal data because all the data is binary with many kinds of labels.

Classifier Training:

Linear Discriminant:

Linear Discriminant analysis is a generalization of Fisher's linear discriminant are methods used to find linear combination of features which characterizes of separates two or more classes of objects. We first run the LDA to check the result on the dataset.

Linear Discriminant	Training	Testing
	0.91	0.87

As we can see that the accuracy on the discriminant analysis shows that the data is not very hard for classify. For training set, we get an accuracy of 0.91 and 0.87 on the testing dataset.

KNN:

K-nearest neighbors algorithm is pattern recognition method used for classification. KNN is a type of instance-based learning or lazy learning, it is the simplest machine learning algorithms. Here we want to have an intuitive impression of the data so with different values of K, we check the accuracy of each parameters.

	# OF NEIHBOURS	Training	Testing
KNN	1	0.62	0.58
	2	0.61	0.61
	3	0.61	0.58
	4	0.76	0.69
	5	0.87	0.85
	6	0.93	0.86
	7	0.90	0.86
	8	0.89	0.86
	9	0.89	0.86

	10	0.89	0.85

My assumption of the KNN was that the accuracy should be at the peak when the K equals to 4 as the original numbers of labels. When focus on the K, we can see that the average accuracy on Training and Testing become the highest when K=6, which means that there are some hard cases for the algorithms to classify.

Decision Tree:

The decision tree means a classifier with each internal node represents test to classify cases. The decision tree is simple to understand and interpret and also can be used combined with other decision techniques.

	Parent and Children nods	Training	Testing
Decision Tree	80 40	0.86	0.83
	60 30	0.88	0.83
	50 25	0.88	0.83
	40 20	0.89	0.83
	20 10	0.89	0.84

Ensemble Learnings:

Finishing Simple classifiers above, we are not satisfied with the final result. Then we switch the gear to the ensemble learning part to develop more powerful classifiers for selecting. Here we decided to select ensembles such as the Subspaces Discriminant, Random Forest, the similarly, trees with AdaBoostM2 as there are multiple class cases. Besides bagging, AdaBoost is used for training the classifiers because some of the labels can be hard to classify. So we would like to reweight different cases to get a better prediction result.

Subspaces Discriminant:

The Subspaces Discriminant basically used algorithm to separate the spaces into different parts and then do the linear separate. Here we use different numbers of separation to compare the accuracy.

	# of Subspaces	Training	Testing
Ensemble Discriminant Subspace	5	0.72	0.82
	10	0.72	0.68
	50	0.73	0.70
	100	0.73	0.70
	1000	0.72	0.70

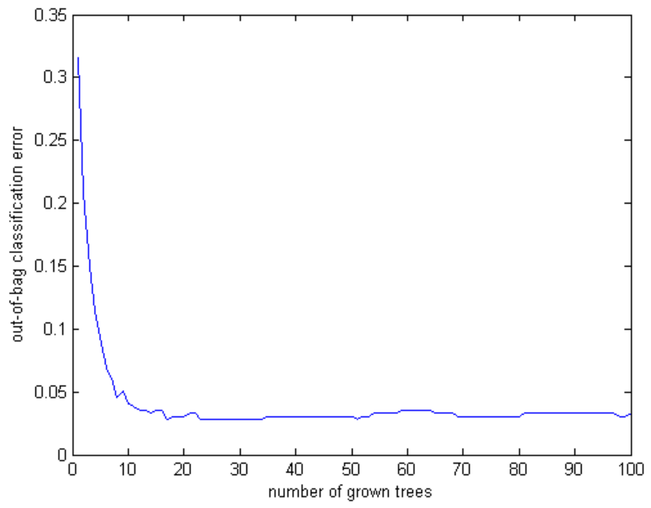
The accuracy is relatively low both for the training and testing dataset compare with other classifiers. As the Number of subspaces increasing, the accuracy of training do not increase at the same time and so does the testing dataset.

Random Forest:

For random forest, it is included in the ensemble learning methods with Randomization Injection method. The trees are grown with replacement of training set. Each node obtains m variables random out of Total input variables. The trees here are grown to largest extent possible. There are no pruning. With very good accuracy among current algorithms, the random forest also perform very well on large dataset. Thus, we would evaluate the algorithm in different aspect beside accuracy only.

	# of Trees	Training	Testing
Random Forrest	5	0.91	0.89
	10	0.92	0.89
	15	0.92	0.89
	20	0.92	0.89
	50	0.92	0.89

	100	0.92	0.89
	1000	0.92	0.89



Too many trees may become computational expensive we would like to look at the cost, or error with the growing number of trees.

There is no need for cross-validation or a separate test set to get an unbiased estimate of test set error. Each tree here is constructed using a different bootstrap samle from the original data. The Out-of-bag estimate is as accurate as using a test set of the same size as the training set. Therefore, using the out-of-bag error estimate removes the need for a set aside test set.

AdaBoostM2 with trees:

There are also an question bear in my mind that when I want to train the random forest, an ensemble learner combined with many unpruned trees, how does this different between I use AdaBoostM2 to combine different trees together. Though I know there are many difference between those two methods on parameter and method aspects, I still think this might be good for me to train an ensemble learning method based on trees.

	# of Trees	Training	Testing
Ensemble AdaBoostM2 Tree	10	0.86	0.67
	20	0.88	0.81
	50	0.87	0.82
	100	0.89	0.80
	5000	0.89	0.81

SVM:

For the Support vector machine, we have to first do the transformation of the data. As we know that the Support Vector Machine mostly used for binary classification purpose, however we here have 4 labels for classification. We have to make our labels into binary. In the following part, we will first test different margin for linear SVM model to see the best fit margin (margin with highest accuracy). Then we get to the non-linear model to get hopefully better result in the RBF method.

SVM Linear:

We assume that different labels is linear separable.

	Margins	C_Mtr	C_Mte	D_Mtr	D_Mte	E_mtr	E_mte	G_Mtr	G_Mte
SVM Linear	0.001	0.50	0.50	0.95	0.50	0.50	0.50	0.81	0.80
	0.01	0.94	0.91	0.94	0.94	0.94	0.96	0.85	0.79
	0.1	0.96	0.92	0.96	0.95	0.94	0.96	0.92	0.90
	1	0.95	0.92	0.95	0.96	0.94	0.96	0.91	0.91
	10	0.95	0.92	0.97	0.97	0.94	0.97	0.91	0.93
	100	0.96	0.92	0.97	0.97	0.94	0.98	0.91	0.93
	1000	0.96	0.92	0.97	0.97	0.94	0.98	0.91	0.92
	10000	0.96	0.92	0.97	0.97	0.94	0.98	0.91	0.92
	100000	0.97	0.92	0.95	0.97	0.94	0.98	0.93	0.93
	1000000000	0.91	0.87	0.23	0.96	0.53	0.98	0.65	0.40

With different margins from 0.001 to 10^9 we range the margin from relatively soft to hard the accuracy is increasing gradually and present a sharp decrease at the very last in a relative hard margin. By averaging four different major chords accuracy, we finally choose margin in 100 and 100000 to get a step further to do the RBF kernel SVM. The linear SVM's accuracy is better than the random forest right now.

SVM RBF:

	sigma	C_Mtr	C_Mte	D_Mtr	D_Mte	E_mtr	E_mte	G_Mtr	G_Mte
SVM RBF 100	0.001	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	0.01	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	0.1	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	1	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	10	0.977	0.917	0.961	0.962	0.944	0.964	0.920	0.911
	100	0.939	0.907	0.961	0.941	0.941	0.952	0.920	0.897
	1000	0.501	0.502	0.947	0.503	0.502	0.503	0.811	0.804
	10000	0.501	0.502	0.947	0.503	0.502	0.503	0.811	0.804
	100000	0.501	0.502	0.947	0.503	0.502	0.503	0.811	0.804
	1000000000	0.501	0.502	0.500	0.503	0.502	0.503	0.500	0.500
SVM RBF 100000	0.001	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	0.01	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	0.1	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	1	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	10	0.977	0.961	0.976	0.978	0.954	0.982	0.946	0.939
	100	0.977	0.922	0.969	0.968	0.944	0.982	0.920	0.925
	1000	0.957	0.922	0.957	0.946	0.941	0.976	0.917	0.902
	10000	0.942	0.502	0.947	0.503	0.502	0.503	0.811	0.804
	100000	0.501	0.502	0.947	0.503	0.502	0.503	0.811	0.804
	1000000000	0.501	0.502	0.500	0.503	0.502	0.503	0.500	0.500

Basically we can see different sigma doesn’t bother much, so we would like to select the sigma with 0.001 and the margin with 100.

Comparison and Conclusion:

Different matrices are compared within the parameter.csv file. As different parameters would come out with different recall, specificity and sensitivity. We would simply look at the accuracy while checking with other parameters.

RF	# of trees	train	test
	100	0.9200	0.8938
SVM RBF	Sigma	train	test
	0.001	0.963	0.934

SVM RBF performs better both on the training set and testing set.

But there are many cost on other aspect, such as the data transformation, and normalization if needed. Sometime, many data preprocessing is expensive.

The random forest, as contrary with the SVM, do not need to do the data. With the relatively similar accuracy and high speed, the random forest is better than SVM in multi-class classification.

The decision tree method is a case-sensitive classifier so that small changes in each cases would case different output. The SVM based on probability classifier, is not relatively sensitive in this case. According the assumption, I would like to test the fluctuation of changing the dataset with unbalanced dataset and balanced dataset. However the result seems not that exciting as the accuracies don't changing dramatically.

	Random Forest	SVM
Inbalanced	0.909	0.959
Balanced	0.894	0.934

Future Study:

At the very first of the study regard my topic, I would like to use the sound signal as my data input to classifier the chords. One of my colleagues majoring in Electronic Engineering graduated from UIC would like to cooperate with me for my project. He is the keyboard player in our band and also has domain knowledge in MIDI instrument. We read many papers to see the relative report. The sound signal can be captured and transform into spectrum similar to the hard writing picture or facial recognition pictures. Using Fourier transformation, the signal can be decomposed into different waves.

Using the information transformed into the spectrum, we can use Hidden Markov classifiers or Bayes networks to both classify different chords and even the music style.