Università degli Studi di Milano

Anna Fusari

932882

Music Genre Classification

Project report

Introduction

The aim of "Music genre Classification" project is to analyze and classify a set of songs belonging to three different music genres, extracting audio features.

The first step consists in collecting eighteen songs equally distributed among three different music genres, then extracting the audio features, time and frequency domain, for each song.

The second step consists in divide the data into two groups, the first one represents the training set, the other one is the test set.

The kNN, k-nearest neighbor algorithm, is used for the classification.

Structure of the project

My project is based on main.m script; it contains the following functions and scripts:

Song.m :

- imports 18 songs, extracts the different audio features divided into frequency and time features, thanks to the "singnal_freq()" and "signal_time()" functions, and stores them into three different matrix.
  example:

```
classic1_freq = signal_freq('classic1.wav');
classic1_time = signal_time('classic1.wav');
classic1 = [classic1_freq; classic1_time];
```

- uses features of first three songs for each genre, to create three different training sets.
  example :

```
trainClassic = [classic1 classic2 classic3];
trainClassic_freq = [classic1_freq classic2_freq classic3_freq];
trainClassic_time = [classic1_time classic2_time classic3_time];
```

- creates three different sets of labels for each genre
  example:

```
labelClassic = ones(1,length(trainClassic));
labelClassic_freq = repmat(1, 1, length(trainClassic_freq));
labelClassic_time = repmat(1, 1, length(trainClassic_time));
```

- uses features of last three songs for each genre, to create test and label sets.
  example:

```
testClassic = dataClassic(:,floor(length(dataClassic)/2)+1:end);
testlabelClassic = repmat(1, 1,length(testClassic));
```

```
testClassic_fre=
dataClassic_freq(:,floor(length(dataClassic_freq)/2)+1:end);
testlabelClassic_freq = repmat(1, 1,length(testClassic_freq));

testClassic_time=
dataClassic_time(:,floor(length(dataClassic_time)/2)+1:end);
testlabelClassic_time = repmat(1, 1,length(testClassic_time));
```

Signal analysis functions:

- signal_freq.m: turns a stereo signal into mono through the function "stereomono.m", cuts 60 central seconds of the song, and uses them to create different frames thanks to the function "windowize.m", then extracts all frequency features from the song using the algorithms seen in class ("feature_mfccs_init.m", "getDFT.m", "feature_spectral_dentroid.m", "feature_spectral_rolloff.m", feature_mfccs.m").

- signal_time.m: turns a stereo signal into mono through the function "stereomono.m", cuts 60 central seconds of the song, and uses them to create different frames thanks to the function "windowize.m", then extracts all time features from the song using the algorithms seen in class ("feature_energy.m", "feature_zcr.m").

define_all_train_label_test.m:

- uses training, label and test sets created in "Song.m" script to make generic frequency, time and all features sets.
  example:

```
all_train = [trainClassic trainSoul trainPop];
all_label = [labelClassic labelSoul labelPop];
all_test = [testClassic testSoul testPop];
correct_label = [testlabelClassic testlabelSoul testlabelPop];
```

predict.m:

- uses the "k_nearest neighbor" algorithm to calculate the recognition rate, which indicates the percentage of success in the classification of music genres given the number of neighbors k.
- returns also the maximum value of recognition rate

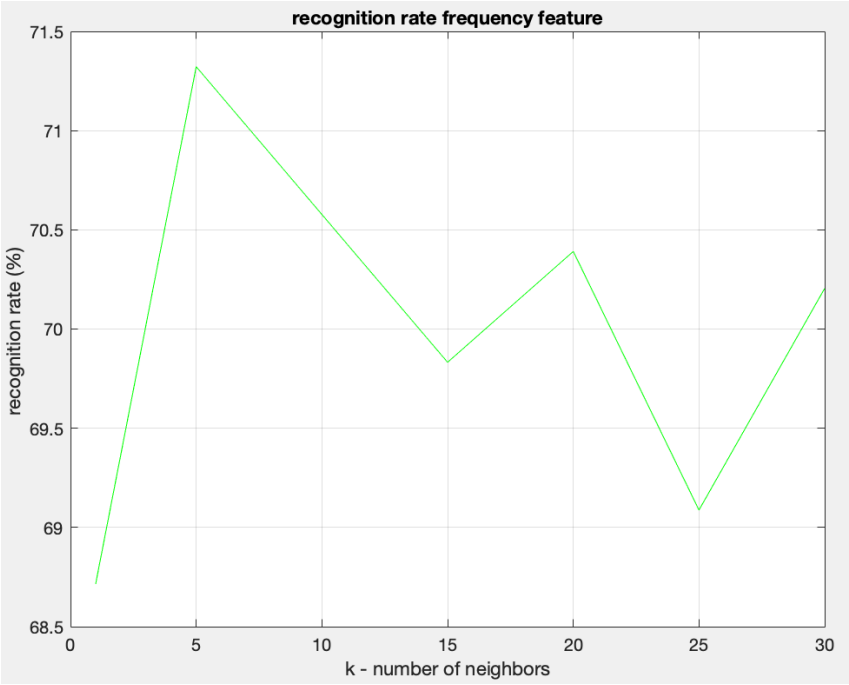
Apply the function "predict.m" to the generic sets for frequency, time and all features and plot the results.

Results

The results depend on the choice of the parameter k.

The first time I chose

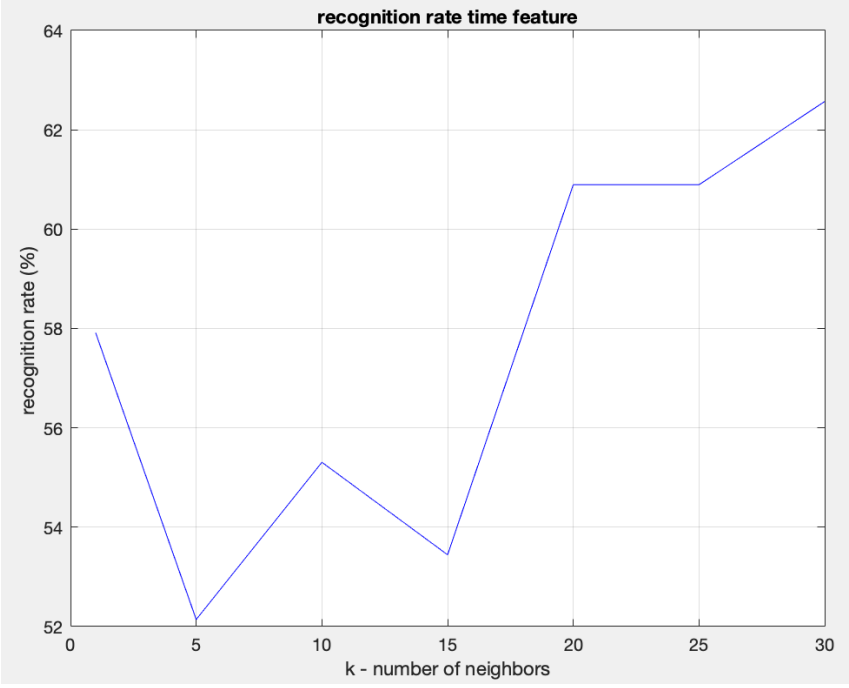
k = [1 5 10 15 20 25 30]



This is the "recognition rate frequency features" graph from k value 1 to 30.

As we can see the graph doesn't run smoothly.

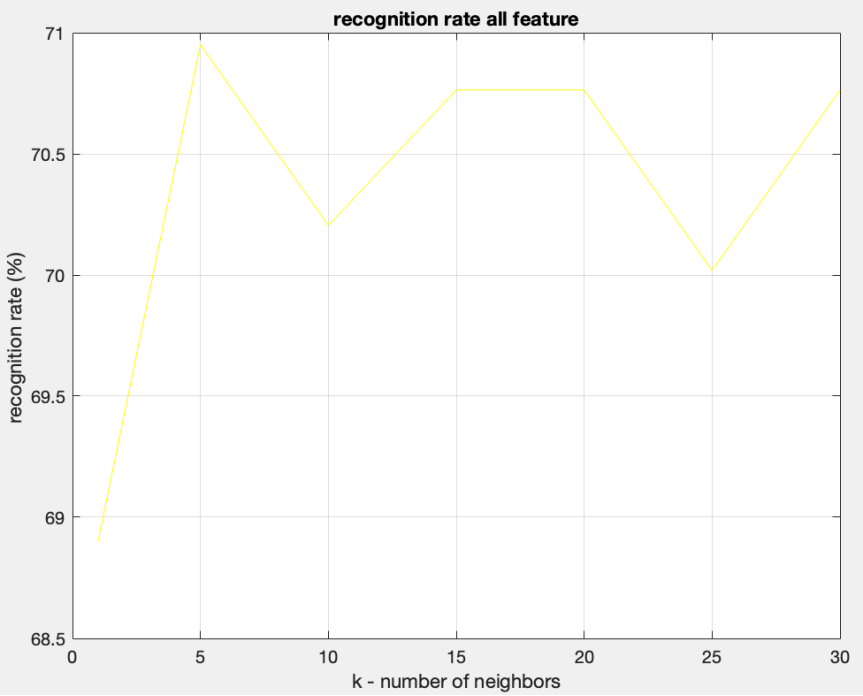
It reaches the maximum recognition rate value, round 72 %, at the k value 5.



This is the "recognition rate time features" graph from k value 1 to 30.

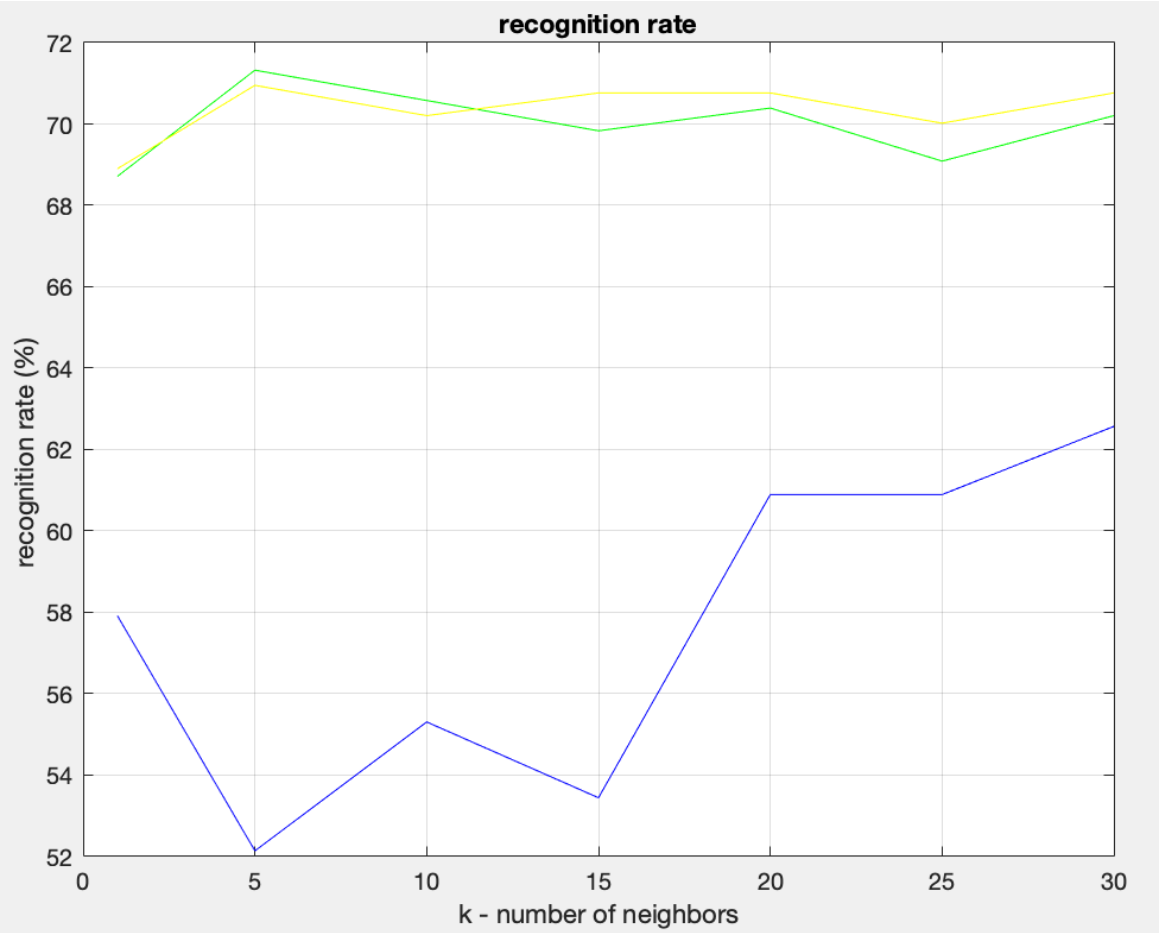
As we can see, the graph has an increasing trend from the k value 15 onwards.

In particular it must be underlined that at the k value 5 the recognition rate is the lowest, unlike the previous graph.



This is the "recognition rate all features" graph from k value 1 to 30.

As we can see, the graph is similar to that of the frequency features, in fact the maximum recognition rate value is about 71 at the k value 5.

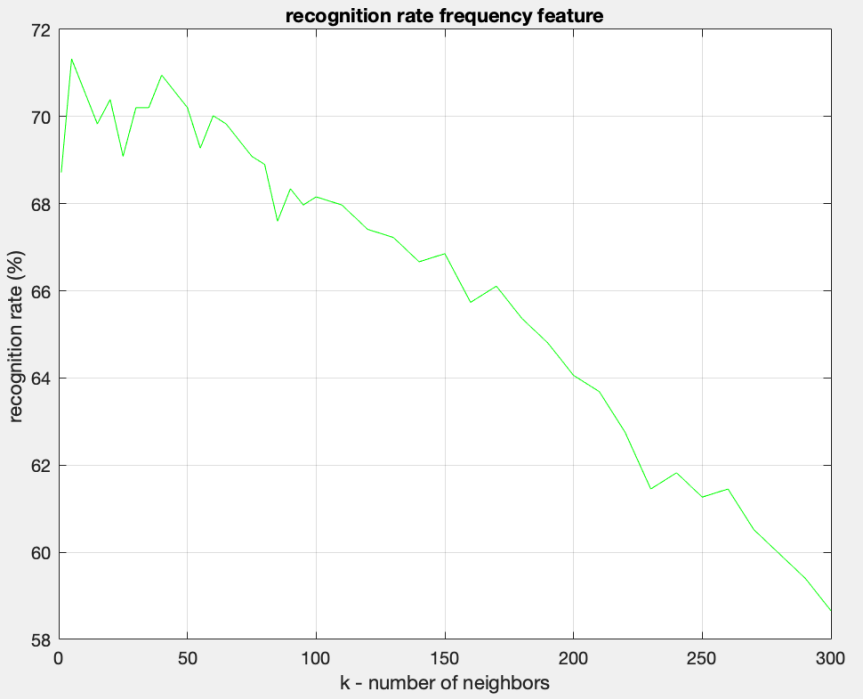


By comparing the three different graphs we can see more clearly the similarity between the frequency features recognition rate and the all features recognition rate.

Due to the growing trend of the time features graph, I decided to extend the k value range to analyze the trend in a more precise way.

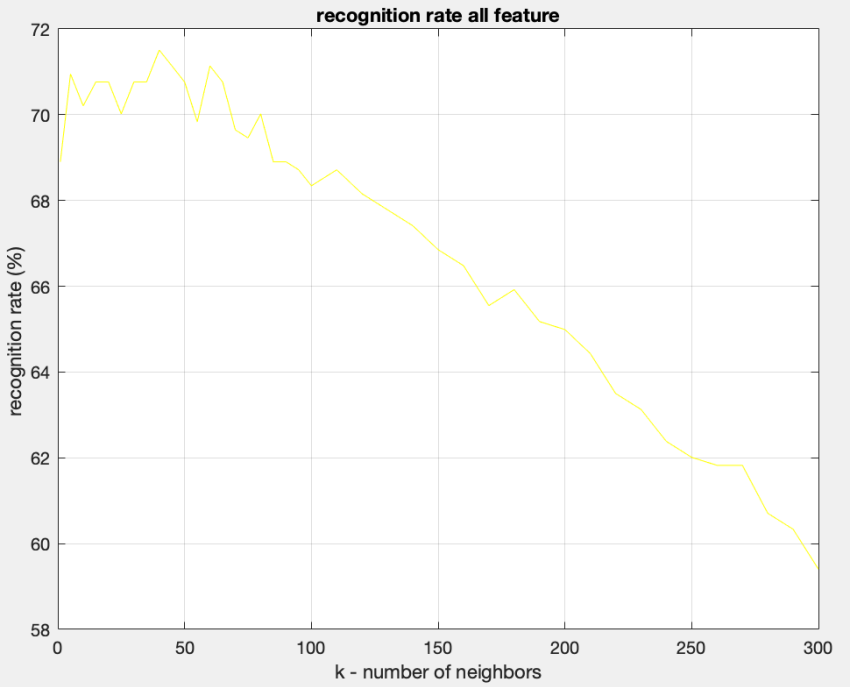
I chose

k= [1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 1350 140 1450 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300];

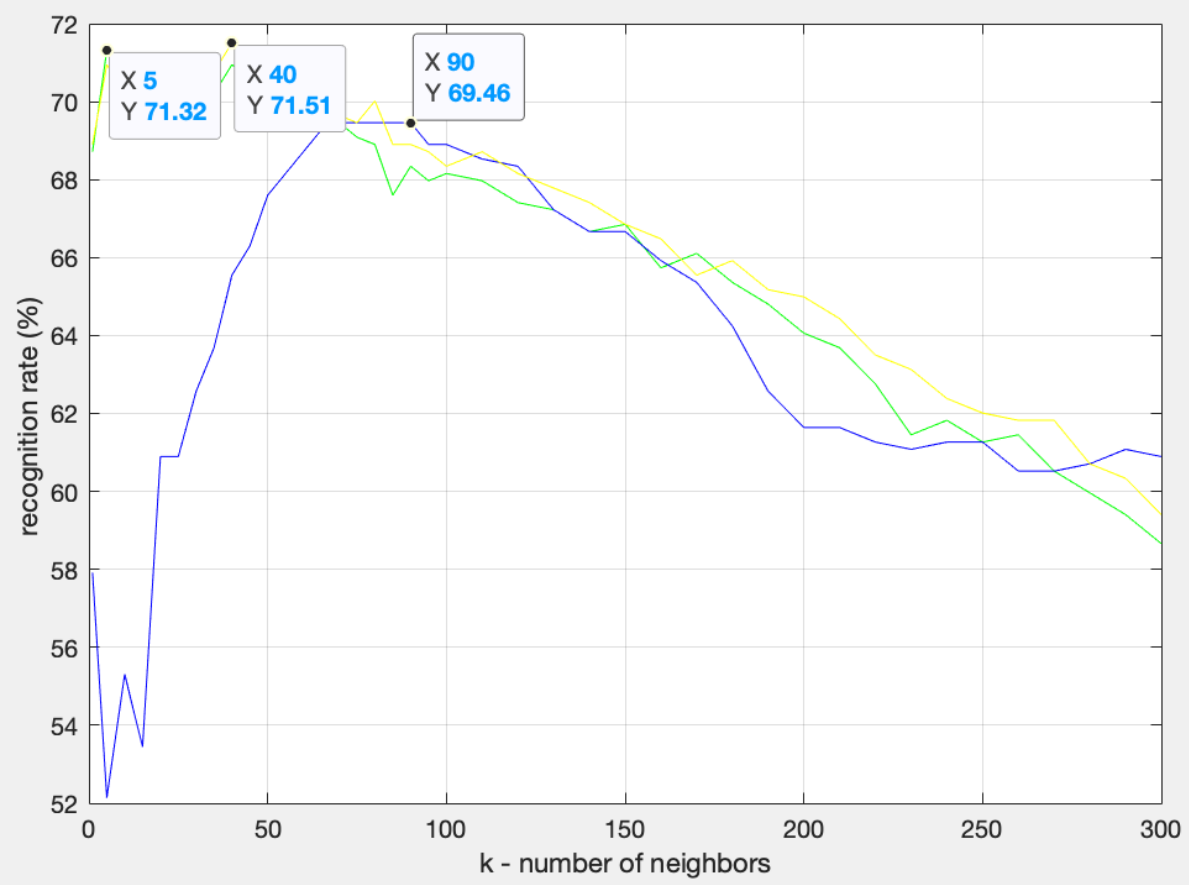


This is the "recognition rate frequency" features graph from k value 1 to 300.

This is the "recognition rate time features" graph from k value 1 to 300.



This is the "recognition rate all features" graph from k value 1 to 300.



By increasing the k value range, we can see that:

- the time features curve grows up to the value 69.46% with k value 90, after that it decreases similarly to the other two curves.

- the maximum value for the frequency features recognition rate is the same as for the first extension of the k value range, 71.32% with k value 5
- if we consider the all features recognition rate curve, we can see that the maximum value 71.51%, is reached at k value 40, which is halfway between time and frequency k maximum value, 5 and 90.

Conclusion

Based on the results obtained, exceeded the k value 100 the curve decreases considerably, so if we consider all features together, the best k value range, with the high recognition rate, is from 5 to 90.