Speaker Recognition in non-linear signal processing and pattern recognition.

Rune A. Heick, Rasmus S. Reimer & Nicolai Glud

Aarhus University, Department of Engineering

Abstract

The Content of this paper seeks to present the knowledge gained throughout the non-linear signal processing and pattern recognition course from Aarhus University, department of engineering. The paper is split into multiple sections explaining the data used in the paper, the methods used to treat the data and the methods used for categorising the data.

I. Introduction

The idea behind the project is to recognise the speaker using the methods and categorisers learned in the course pattern recognition and machine learning (TINONS). The voices of all authors was recorded and imported to matlab. The features from the data was extracted in matlab using the Mel-frequency cepstral coefficient(Hereafter MFCC) method from the voicebox toolbox. The MFCC's are used as features for the classifiers that are tested in this paper.

II. Data Gathering

How did we get data?

III. Feature Extraction

Introtext to MFCC

Math

How we use it

Intermediate result

IV. Features

Size and number of features and stuff.

V. Dimensionality Reduction

e.g. finding projection vectors, choosing number of components, applications.

I. PCA

Introtext

Math

How we use it or why we don't use it

Intermediate result

II. Fisher

Introtext

Math

How we use it or why we don't use it

Intermediate result

VI. Classifiers

Classifiers were first know from the world of linear regression. The classifiers found in this section are featured in the non-linear signal processing and pattern recognition course. The section seeks to explain the basis of each of the classifiers along with how we have used them in our project. Intermediate results can be found in the section about the classifiers while the comparison between classifiers can be found in the Results section.

I. Linear Classifier

"e.g. cost/error function, decision boundary and training method"

The goal of linear classification is to take an input vector with multiple x values and assign it to one of multiple classes K. This can be done with one or more linear decision boundaries. The first way to classify is called the one-vs-one linear classifier. This works for 2 classes as seen in figure 1. If multiple clusters of x belonging to more than 2 classes are present we get ambiguous regions as one class might appear to be two different classes. An example of this can be seen in figure 2.

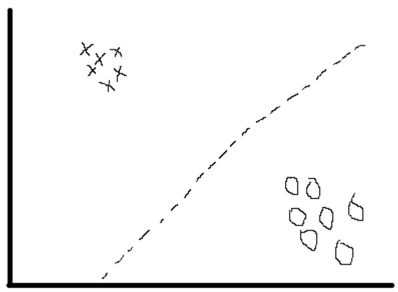

Figure 1: *One-vs-one linear classifier for 2 classes*

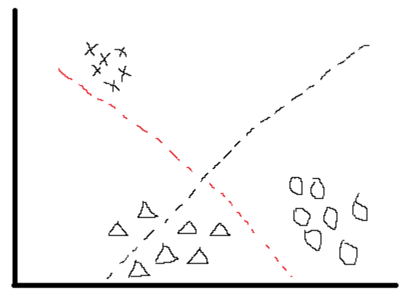

Figure 2: *One-vs-one linear classifier for 3 classes*

Another way to classify the 3 classes seen in figure 2 could be to utilise 1-of-k classification. This can be seen in figure 3. The 1-of-k classifier has no ambiguity in this case.

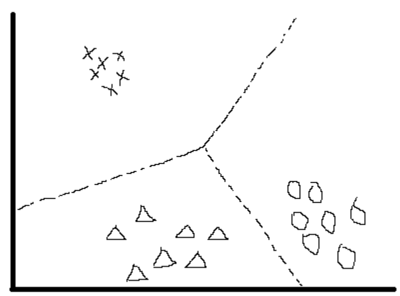

Figure 3: *1-of-k linear classifier for 3 classes*

In math terms the one-vs-one can be written as:

$$y(\mathbf{x}) = \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} \tag{1}$$

This is prone to ambiguity for more than 2 classes. We know that the ambiguity issue can be avoid by using the form:

$$y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + \omega_{k0} \tag{2}$$

and choosing the value of x to be a part of class k if $y_k(\mathbf{x}) > y_m(\mathbf{x})$ for all $m \neq k$. This leads to decision boundaries corresponding to the 1-of-k classifier.

How we use it

Intermediate result

II. Probability Classifier

e.g. maximum likelihood, training/testing and generative vs. discriminative models.

Introtext

In many cases the features follows a certain distribution. By using the information in the distribution it is possible to filter out outliers and determine how likely it is that the point is a part of our class. This can be very powerful since we not blindly put a sample in a class but also get information about the likelihood. This of course demands that we are able to give a qualified guess of the distribution to use. In this analysis it is assumed that the data is Gaussian distributed. This assumption is made by looking at the histogram of the features. this is shown on figure (fix me).

show figure 1

By using the probability for a given sample if a certain class is assumed P(x|C). Iteration over the the different classes we can get the probability of the sample given these classes. This information can be used to determine what class and how certain we are of this decision. This is illustrated in figure X. (fix me)

figure 2

Instead of asking what the probability of the sample given the class is P(x|C), the reveres probability can be used. That is the probability of a class given a sample P(C|x). This can be found using Bayes rule:

$$P(C|x) = \frac{P(x|C)P(C)}{P(x)} \tag{3}$$

This will for the Gaussian distribution something like a sigmoid function. using this model to classify it is no longer able to tell about the probability of a sample not being in any class, but on the same time also simplifies the classifier a lot. Compared to the linear classifier the this probabilistic classifier are able to create a mouth sharper decision bound.

If the sharper decision bound is the goal then a easer approach is to estimate the optimal sigmoid for separation of classes directly. This can be done by optimizing a softmax-function to separate the classes. The softmax-function is expressed as:

$$Y_k(w_k, x) = \frac{e^{w_k x}}{\sum\limits_{k=1}^{K} e^{w_k x}} \tag{4}$$

Comparing this to the previous probabilist function we can assume that:

$$P(t|w, x) = p_n^t (1 - p_n)^{1-t} =, t \in [0, 1] \tag{5}$$

Here we see how lilly it is that the class vector t, is correct given data point x, and some weights w in the soft-max. Where t is the class vector, that indicates which class the data point x is part of. The pn is given by:

$$p_n = P(C|w, x) = y(w, x) \tag{6}$$

The challenge is now to find the optimal weights wk for each class to create the best classifier. This can be done by combining the two equations 4, 5 to create a non linear optimisation problem.

$$L(w) = \log \prod_{i=1}^{N} y(w, x_i)^{t_i} (1 - y(w, x_i))^{1-t_1} = \sum_{i=1}^{N} t_i \log y(w, x_i) + (1 - t_i) \log(1 - y(w, x_i)) \tag{7}$$

This can be solved by many different optimizations strategies. By optimizing this for each class we will find the optimal weights for the softmax-class separator in equation 4.

How we use it or why we don't use it

Intermediate result

III. Artificial Neural Network Classifier

e.g. graphical network model, training method, model flexibility (expressive power)

Introtext

Math

How we use it or why we don't use it

Intermediate result

IV. EM Classifier

e.g. training method, cost functions, model order selection, initialisation of parameters.

Introtext

Math

How we use it or why we don't use it

Intermediate result

V. Sequential Models

Markov model and Hidden Markov Model.
e.g. meaning of parameters, left-to-right model, outline of training/testing method.

Introtext

Math

How we use it or why we don't use it

Intermediate result

VI. Support Vector Machines

e.g. decision function, support vectors, soft margins, kernel trick.

Introtext

Math

How we use it or why we don't use it

Intermediate result

VII. Results

Compare all the methods in a table in order to show the performance.

VIII. Discussion

I. Subsection One

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

II. Subsection Two

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus.

Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

IX. Conclusion

References

[Figueredo and Wolf, 2009] Figueredo, A. J. and Wolf, P. S. A. (2009). Assortative pairing and life history strategy - a cross-cultural study. *Human Nature*, 20:317–330.