Machine Learning II

Kernel methods on cannabis microsatellite data

Marc Vernet Sancho & Jordi Puig Rabat

Professor: Lluís Belanche Muñoz

January 2021

1 Introduction

Cannabis is one of humanity's oldest cultivated plant and a very wide spread crop across the world. It has a high genetic variety, mainly due to human selection [1]. The most interesting characteristic in a plant is the THC:CDB ratio, as the proportion of this two molecular components will determine if the plant has any kind of psychoactive properties and thus it's cultivation is illegal or highly restricted. The non-psychoactive varieties are widely used for industrial purposes. This generates a need for a classification between the two main varieties: *hemp* (low index) and *marijuana* or *drug* (high index). There are many approaches to do this classification, in this project we will take the one involving genetic data from different microsatellites.

Microsatellites –or Short Term Repeats (STR)– could be described as repetitive secuencies of DNA in which certain DNA motifs are repeated; a certain number of repetitions is an allele and in this case a microsatellite is formed by two alleles. One of the most interesting characteristics from microsatellites is their high mutation rates, this makes them perfect to represent genetic diversity. In order to track the genetic difference among same specie samples, it's necessary to have a constant marker loci. In population genetics, microsatellites have been a popular choice, as they seem to represent important information of relations within populations, it's genetic variation, structure and spatial and temporal characteristics.

The source dataset was originally used in a study for forensic applications where non-kernel methods were used [2]. The dataset contains 13 microsatellite locis genotyped from a collection of 1324 cannabis samples from different locations and of different kinds (24 *hemp* varieties and 15 *drug* varieties). This dataset is considered "the most comprehensive genetic resource for Cannabis forensics worldwide".

In this project the main method we will be using and developing is the kernelized Support Vector Machine Classifier (SVM). This method allows us to classify in two classes, the kernelized version we can use this method in a non-linear separable dataset.

1.1 Objectives

Our goal is to study in which way kernel functions could better prediction results, and discover ways to fine-tune custom functions in order to to maximize the performance. To do so we will need to compare different kernel functions between them and to non-kernel methods that will work as a baseline. On the one hand we will use Kernelized Support Vector Machine (SVM) with different kernel functions that will be discussed later. On the other hand we will use non-kernel methods like Random Forest to check whether our approach with kernel methods have some advantage over other methods.

2 Dataset encoding

The original data consists of 1324 samples of cannabis plants, described in 13 microsatellites pairs. The way to encode the data is to store the number of repetitions of each allele. This characteristic should be understood as categorical. The samples are labeled as either *drug* or *hemp*.

In order to try to approach the problem with different alternatives, the following recodifications will be used:

- **Original encoding** Data will be considered in the original format, as categorical microsatelites.
- **Expanded encoding** For each possible microsatelite value, it will be encoded for each plant as 0 if the microsatelite doesn't appear, as 1 if it appears in one of the two alleles and as 2 if it appears in both alleles. Using this technique, the data can be considered non-categorical (because for example a plant with a 2 in a certain microsatelite is more similar to an another plant with 1 in the same microsatelite that it is in a plant with 0). The main problem with this configuration is that it is very sparse, and any modification of the initial data inevitably adds noise.
- **One-Hot encoding** The dataset encoded with a regular one-hot encoding of 0 and 1. The benefit of this configuration is that each plant is described as a binary vector, making it possible to implement categorical kernels and faster comparisons. This dataset is very sparse, with a lot of zero values.

The first approach was to use the Expanded and One-Hot encoding; some kernels were designed for these encodings and good results were obtained. The second approach didn't involve to change the data encoding although some modifications in the implementation of the kernel functions were needed. With kernels defined over the Original encoding we obtained even better results.

3 Kernels

When working with kernelized methods like SVM it is usual to use different kernel functions in order to see which one gives us the best results. Here we are going to discuss and define what kernel functions had been used. Among the next kernels, we can consider three categories of kernels based on the use and origin of the kernels; the differentiation of the categories serve us to explain why the kernels are used. The baseline kernels serves in this project to set a baseline for performance; the Categorical kernels are common kernels for categorical variables and the expectation of performance is that it will be better than the baseline ones; the last category is the custom kernel we have designed especially for the dataset hoping it classifies better than the categorical ones.

3.1 Baseline Kernels

Linear Kernel

This is the most basic kernel function, it is defined as the inner product between two vectors. It's one of the kernels predefined in the SciKit Learn [3] library and it is equivalent to use the SVM without kernel. It is defined as follows:

$$k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$$

This will work very well as a baseline for performance.

Radial Basis Function (RBF)

The Radial Basis Function is one of the most popular kernel. It is used for continuous data and normally works very well; in this case with categorical data we consider the input binary vectors. The kernel is defined as follows:

$$k(\mathbf{x}, \mathbf{y}) = \exp(\gamma||\mathbf{x} - \mathbf{y}||^2)$$

We know a priori that this kernel is not a good choice for our type of data, we are interested only in the baseline for performance that this kernel can give us. Later we will see how to adapt this kernel function to a categorical case.

3.2 Categorical Kernels

Overlap or Simple Matching Coefficient kernel

One of the simplest categorical kernels is to compute the proportion of matches from all the variables. It can be defined as follows:

$$k(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{n=1}^{n} [x_i = y_i]$$

$$\text{where } [x_i = y_i] = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$$

In this case we consider x and y to be a vector in a discrete space where each position is a variable and the value is discrete (categorical). Later we will see that it can be applied to the microsatellite as a whole (i.e. both alleles need to be equal) or it can be applied to each position independently.

Jaccard kernel

This kernel is similar to the SMC but instead of normalizing by the total number of variables the sum of matches is divided by the number of traits present in the union of x and y.

$$k(\mathbf{x}, \mathbf{y}) = \frac{\sum_{n=1}^{n}[x_i = y_i]}{x \cup y}$$

Categorical kernel k_0

The k_0 family is a step beyond the overlap kernel, it allows us to apply a function that preserves positive semi-definiteness of the Gram matrix [4]. Here is presented a possible way to extend the RBF kernel to categorical variables if f_p is the exponential function $exp(-\gamma x)$ and f_a is the identity function. It can be defined in general like this:

$$k_0(\mathbf{x}, \mathbf{y}) = f_p(\frac{1}{n}\sum_{n=1}^{n} f_a([x_i = y_i]))$$

There are many possible approaches to take whith this kernel taking different functions f_a and f_p. In our study we will only test the previous commented extension of the RBF kernel also known as k_0':

$$k_0'(\mathbf{x}, \mathbf{y}) = exp\big(-\gamma\frac{1}{n}\sum_{n=1}^{n}[x_i = y_i]\big)$$

3.3 Custom kernel

When comparing microsatellites formed by a pair of alleles we can try to add extra information by comparing not only position by position and comparing crossing positions. This is possible adding a new kernel which we will call crossed SMC to the original SMC kernel. This can be represented as a kernel over one microsatellite and then the kernel between two plants is the sumatory of each kernel for each allele. Since the sum of kernels is also a kernel –meaning it preserves the positive semidefiniteness of the Gram matrix– this operations are possible and likely to give us good results.

To define this kernel let's consider that a plant is represented by a vector with 13 elements where each element is a microsatellite (i.e. a pair of alleles). Let's call the kernel over two microsatellites m_1 and m_2 the cross and stright kernel (CAS kernel).

$$k_{CAS}(m, m') = [m_{1,1} = m_{2,1}] + [m_{1,2} = m_{2,2}] + [m_{1,1} = m_{2,2}] + [m_{1,2} = m_{2,1}]$$

$$\text{where} \quad [\mathrm{x} = \mathrm{y}] \begin{cases} 1 \text{ if} x = y \\ 0 \text{ otherwise} \end{cases}$$

With this definition we define next the kernel over two observations as:

$$k_{custom}(\mathbf{x}, \mathbf{y}) = \frac{1}{2n}\sum_{i=1}^{n} k_{CAS}(x_i, y_i)$$

This is a kernel function for microsatellite data where n is the number of microsatellites in this case $n = 13$. This takes in account more information combining

4 Kernel comparison with Expanded and One-Hot encoding

As a first way of analyzing the data and understanding how kernels work it's useful to compare the different families similarities in a matrix. All varieties that appear in the dataset come from a group of 48 plant families with different genetic characteristics. Calculating the centroid of each family (using the expanded encoding and assuming that the variables are quantitative) we can have an approximate idea of the constitution of each family, described as an average plant of each family. This averages can first be processed with a kernel function in order to estimate the similarities among them. It can be seen in Figure 1, where yellow represents high similarity and dark red means little similarity. There are families that have more tendency to be similar to the rest of plants, it can be suposed that have smaller genetic variety or that have already been exposed to the other families, generally it will be non-psychoactive families. There are some specific families that are very different to all the other plants, this families are geographically isolated or are man engineered to have special characteristics, there's high probability that they are psychoactive varieties.

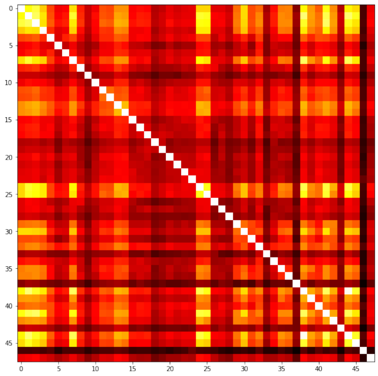
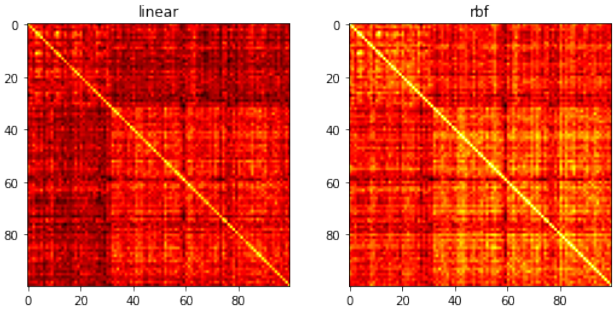


Figure 1: K0 Kernel Matrix of families average

A first glimpse of the proposed kernels studied looking at the generated kernel matrix, visualized as a heat map. The results are shown for the first 100 varieties of cannabis, because the total number of varieties resulted in an enormous heatmap that was difficult to interpret. In Figure 2, the linear and RBF kernels have been plotted for the expanded encoding dataset, as it's non-categorical data. By just looking it's difficult to notice differences, both seem to give a similar result.

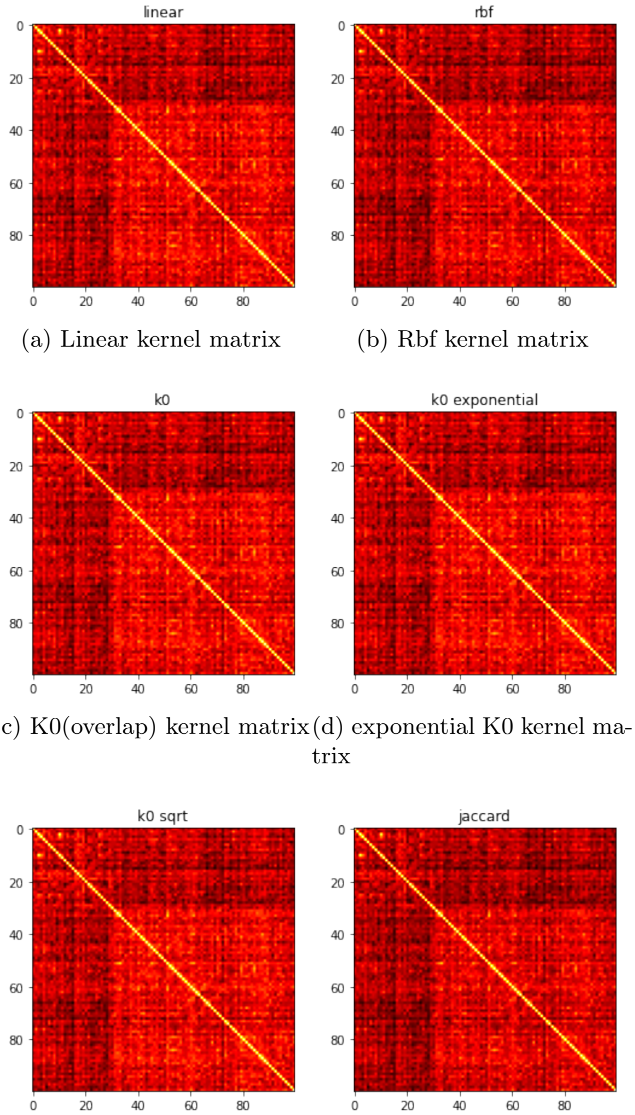


(a) Linear kernel matrix

(b) Rbf kernel matrix

Figure 2: Kernel Matrix with original encoded dataset (100 values)

When comparing the kernel outputs with the categorical data in Figure 3 using the one-hot encoding, the results are even more similar, which means that the kernel method used for prediction won't change the accuracy in any remarkable way.



(a) Linear kernel matrix (b) Rbf kernel matrix

(c) K0(overlap) kernel matrix (d) exponential K0 kernel matrix

(e) sqrt k0 kernel matrix (f) Jaccard kernel matrix

Figure 3: Kernel Matrix with one hot encoding dataset (100 values)

A SVM is trained with each kernel function and different values for the C parameter (in order to find the best performance and avoid overfitting). In the results showed in Table 1, the supositions are confirmed. All the different methods give very similar results, even the C parameter doesn't seem to be affecting that much. The ones with better performance are RBF for the expanded encoding and Jaccard for the one-hot encoding. Both matrix are a little different that the rest, what explains the different results.

Finally, as a baseline a Random Forest has been trained with the data, reaching a 0.9214 accuracy. This means that the kernel methods prediction are rather bad, as the use of kernels doesn't seem to improve performance.

Kernel	C = 1	C = 50	C = 100
Linear	0.9025	0.9025	0.9025
RBF	0.9335	0.9093	0.9093
Jaccard	0.9267	0.9086	0.9071
k0 (overlap)	0.9025	0.9025	0.9025
k0 exponential	0.9048	0.9048	0.9048
k0 square root	0.9025	0.9025	0.9025
Linear (Original)	0.9025	0.9025	0.9025
RBF(Original)	0.9335	0.9093	0.9093

Table 1: Results of validation obtained with 5 fold-CV

Kernel	C = 1	C = 50	C = 100
Linear	0.8964	0.8964	0.8964
RBF	0.7953	0.9093	0.9101
SMC	0.9472	0.9509	0.9509
k'_0	0.9434	0.9494	0.9494
custom	0.9472	0.9509	0.9509

Table 2: Performances over Original encoding with different kernels and different regularization parameter C.

5 Kernel performance over original encoding

After seeing the results with Expanded and One-Hot encoding we wanted to try how the models with kernels over the original encoding would work. Kernel methods usually have the best performance when it works directly on the source data, where it can absorb in a better way what differentiates the different elements. To this end a new implementation of the kernels was needed. The kernels used with this encoding are SMC, k'_0 and a new kernel designed specifically for this project. Also we set a baseline with the linear kernel and the Radial Basis Function kernel and with a non-kernel method like before: Random Forest. Again, the results of the validation process has been obtained with 5 fold Cross Validation as it presents a better aproximation of the error than the train-test split. It has to be noted that the custom kernel, SMC and k'_0 considered the whole pair of alleles when comparing microsatellites. The Linear and RBF kernel considered the vector of pairs as a flat vector of 26 alleles instead.

The 2 shows the results obtained with the different kernels and different values of the regularization parameter.

With the Random Forest Classifier we obtained an accuracy of 0.9365 which is pretty similar to the ones obtained with the kernelized methods. With all this information we can see that using categorical kernels over the original encoding give better results than the expanded and one-hot encoding. The custom kernel we designed give slightly better results as we can see in the table but this improvement is not significant.

6 Conclusion

The classification of Cannabis plants seemed to be a perfect study case to apply kernel methods, that are specially useful in cases of genetic data, where the samples have lots of variables that are easy to compare, in this case a string of values. As it is already known in the world of kernels, the main challenge is designing a kernel that is able to consistently extract the relevant information of each sample to provide a meaningful comparison; here we have seen that with the help of previously studied categorical kernels this challenge has been satisfactorily accomplished. In this short project, we have been able to see first-hand the strength of kernels; with default kernels the result wasn't that good, but with categorical kernels and the custom fine-tuned kernel the results were improved compared to the baseline methods including kernel and non-kernel ones.

So, the conclusion learned is that developing a good kernel can be a key element in data prediction, as it can boost performance in extraordinary ways.

7 Annex

In the annex of this project can be found a Jupyter Notebook file with all the tests and made and the results, the original data and the files to process it, and other auxiliary files with implementations of kernels.

The files can be also found in the GitHub repository

References

[1] Ernest Small and Arthur Cronquist. "A Practical and Natural Taxonomy for Cannabis". In: *Taxon* 25.4 (1976). DOI: `10.2307/1220524`.

[2] Christophe Dufresnes et al. "Broad-Scale Genetic Diversity of Cannabis for Forensic Applications". In: *PLOS ONE* 12.1 (Jan. 2017), pp. 1–13. DOI: `10.1371/journal.pone.0170522`. URL: `https://doi.org/10.1371/journal.pone.0170522`.

[3] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[4] Carlos Garcia Marquez. "Multivariate Kernel Functions for Categorical Variables". In: (2014). URL: `http://hdl.handle.net/2099.1/24508`.