# Advanced Learning Models Data Challenge Report

Predrag Pilipovic, Lucas Batier

February 18, 2020

## 1 Introduction

The aim of the data challenge is to predict whether a **DNA sequence** region is **binding** site to a specific **transcription factor** (TF). We get 3 different sets of sequences where each set corresponds to a different transcription factor. To solve this classification problem we implemented different machine learning algorithms such as Perceptron and Adaline, but also some more advanced ones like **Multinomial Naïve Bayes** (MNB) and **Kernel Support Vector Machines** (SVM). We tested our model with K-fold cross validation on the provided training set before submitting. The best results we got are with MNB, in average: 67% for the first TF, 68% for the second one and 64% for the last one. We obtained 67% in the first round of the Kaggle competition, which was compatible with our testing results. Afterwards, it turned out that the model was overfitted, something we did not expect. Alongside the report, the python notebook (used to compute all the results), python script and `README` file defining its usage are included.

## 2 DNA sequences and k-mers

A DNA sequence is composed of amino-acids `{A, C, G, T}` e.g. `AACTTTGTC` which are transcripted 3 by 3 in protein by the ribosome, e.g. `AAC|TTT|GTC` → `APV`. A classical preprocess of DNA sequence is splitting it in subsequences and counting their occurrences. This process is called k-mers counting and it is also useful for computing spectrum kernel used in SVM. The only parameter of the algorithm is $k$ which is the length of the subsequence. Example in Figure 1 for $k = 2$.

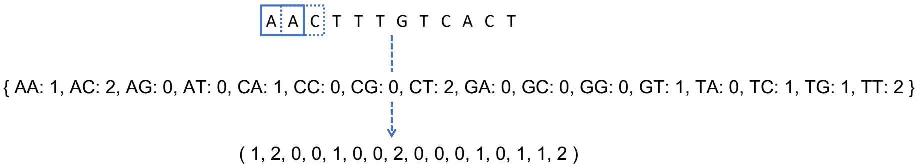

A A C T T T G T C A C T
{ AA: 1, AC: 2, AG: 0, AT: 0, CA: 1, CC: 0, CG: 0, CT: 2, GA: 0, GC: 0, GG: 0, GT: 1, TA: 0, TC: 1, TG: 1, TT: 2 }
( 1, 2, 0, 0, 1, 0, 0, 2, 0, 0, 0, 1, 0, 1, 1, 2 )

Figure 1: k-mer counting ($k = 2$)

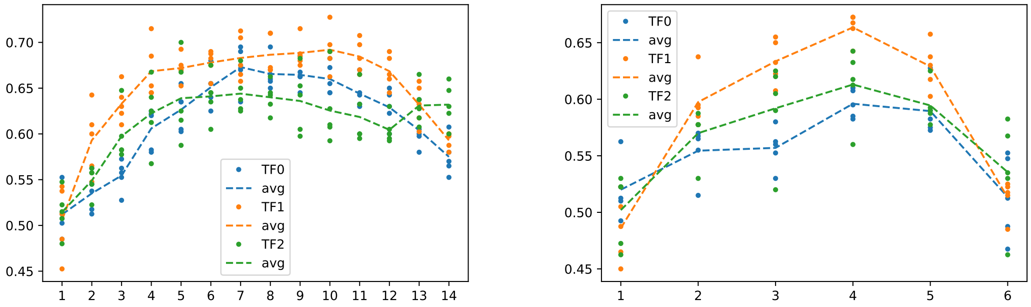
## 3 Solutions and results

The first attempt to solve the problem consisted in working with the provided numerical embedding, which was computed as a transformation of the string sequences. This did not provide good results, that is why we proceed with more sophisticated approaches.

### 3.1 Multinomial Naïve Bayes

The MNB is a basic algorithm for Natural Language Processing (NLP) or document information retrieval. First, we need to know how many times a word from a vocabulary appears in a document. By analogy, each sequence can be seen as a document containing $n$ words of length $k$. As the name implies, this classifier utilizes Bayes' rule to classify an observation as belonging to one of 2 groups.

“Naïve” in this sense means that this classifier will use strong independence assumptions that are perhaps unwarranted. Specifically, the MNB classifier assumes conditional independence among all of the predictor variables of an observation. Still, we need to find the optimal $k$. We computed the accuracy with 5-fold cross validation (5 times for each $k$) and calculated the average over each. Results are shown graphically in Figure 2a. Then we chose $k = 7$. Moreover, we see that depending on the TF $k$ can be tuned to fit better. Nevertheless, this validation is done on a really small part of the data, so a tuned $k$ can lead to overfitting.



(a) Accuracy of MNB with different $k$

(b) Accuracy of spectrum SVM with different $k$

Figure 2: Choosing $k$ for MNB and SVM with spectrum kernel

### 3.2 Kernel Support Vector Machines

Another advanced algorithm would be SVM with appropriate kernel. Since our data is character string, we need to find a good feature map $\Psi$, to map our data in Hilbert space. One of the most convenient Kernels for string sequences as commonly recognized by literature to work well with DNA sequence classification is Spectrum Kernel. Feature map is indexed by all possible k-mers from the alphabet of amino acids, i.e. feature space will be $\mathbb{R}^{4^k}$. Because of the high dimension of the feature space, our data is sparse, thus we apply $L_2$ regularization. Hence, we need to choose regularization parameter ($C$ or $\lambda$ depending on the model description). In our case, we tried different values but the results were not sufficiently different, so we kept the default value $C = 1$. Regardless, $k$ is an influent parameter, so we tuned it again by 5-fold cross validation and we ended with $k = 4$ (Figure 2b) which is feasible from a biological point of view, considering that reading errors can occur. Indeed, the TF needs 3 amino-acids in the sequence, thus by adding 1 it can absorb the reading errors. N.B. that we did not apply the kernel in the SVM but we prepossess the data.

## 4 Conclusion

Many classic methods for machine learning have a very strict performance limit when attempting to solve methods that have internal behaviors that numerical representations fail to capture. But kernel transformations can deal with it efficiently. In this data challenge we realized the importance of the "kernel trick", i.e. finding a feature space with good data representation in order to linearly separate them. We believe that the spectrum kernel is the best solution, even though we did not get the best results with it. It is more flexible and adapted to computational biology. Further work could be dealing with the possible reading errors in order to improve the results, this would include using different kind of kernel like Mismatch Kernel.