Kernel Methods - Kaggle Challenge

Team Name: VB
Vivien Brandt

Public Score: 0.702 ($12^{\text{th}}/37$) **- Private Score: 0.714** ($6^{\text{th}}/37$)

In this report, I present my work done on the Kaggle challenge "Data Challenge Kernel Methods (2024-2025)".

1 Presentation of the Challenge

The goal of this challenge is to predict whether a DNA sequence region is binding site to a specific transcription factor. The data consists of 3 datsets, each corresponding to a different transcription factor with 2000 training sequences and 1000 testing sequences per dataset. This is a binary classification problem, where each sequence is labeled as either bound (1) or unbound (0). One important fact is that the challenge must be addressed using Kernel Methods.

2 Methodology

The Kernels I decided to work with where the **Substring**, **Spectrum** and **Mismatch** Kernels and the chosen classifier was the **Logistic Regression**. These Kernels seems well adapted for DNA sequences as they capture similarity by counting shared substrings between DNA sequences. The Logistic Regression, which is not too hard to implement can then perform the final classification.

2.1 Substring Kernel

We denote as $\mathcal{I}(k,m)$ the sequences $i = (i_1, \dots, i_k)$ such that $1 \leq i_1 < \cdots < i_k \leq m$. For x a string of length m and $i \in \mathcal{I}(k,m)$, we define $x(i) = x_{i_1} \dots x_{i_k}$ and $l(i) = i_k - i_1 + 1$.

We now fix $k \in \mathbb{N}$ and $\lambda \in \mathbb{R}_+$. For all u and x we define $\Phi_u(x) = \sum_{i \in \mathcal{I}(k,|x|), x(i)=u} \lambda^{l(i)}$.

Given sequences x, x', the Substring Kernel is defined as: $K_{k,\lambda}(x,x') = \sum_{u \in \mathcal{A}^k} \Phi_u(x)\Phi_u(x')$

The parameter k determines the length of substrings considered in the similarity computation and the parameter λ controls the contribution of substrings based on their length in the sequence.

I couldn't use this Kernel as it took more than 7 hours to compute a 2000×2000 Kernel matrix for a single training dataset with my computing power. Thus, this Kernel won't be used in the experiments.

2.2 Spectrum Kernel

This Kernel will measure how well x and x' have subsequences in common.

The Spectrum Kernel computes $K_k(x,x') = \sum_{u \in \mathcal{A}^k} \Phi_u(x)\Phi_u(x')$ where $\Phi_u(x)$ is the number of occurences of the substring u of length k in x.

2.3 Mismatch Kernel

The Mismatch Kernel is very similar to the Spectrum Kernel, but it allows m mismatches.

2.4 Logistic Regression

Using these kernels, we will perform our Kernel Logistic Regression.

Our classifier $\hat{f}$ is the solution of the following optimization problem:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ln(1 + e^{-y_i f(x_i)}) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2$$

By the representer theorem, $\hat{f}(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$, with $n = 2000$ during training.

We therefore in practice will be solving $\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \ln(1 + e^{-y_i [K\alpha]_i}) + \frac{\lambda}{2} \alpha^T K \alpha$

3 Implementation - Code and Reproducibility

To efficiently compute the kernel matrix, I used the **joblib** library for parallelization, significantly speeding up the process, and during the hyper-parameter tuning phase I saved the kernel matrices to avoid recomputation.

The code used for the challenge is available on `GitHub`. The README provides instructions on how to use the code.

4 Results

First, I performed a train/validation split with a 0.8-0.2 ratio in order to test out the different Kernels and tune the hyper-parameters k, m and λ. Then a 5-fold cross-validation was performed. The Kernels and parameters that where finally chosen are:

Table 1: Chosen Parameters

	Dataset 0	**Dataset 1**	**Dataset 2**
Kernel	Mismatch	Spectrum	Mismatch
k	10	8	11
m	1	None	1
λ	10^{-6}	10^{-6}	10^{-6}

With the above parameters, the obtained results of the 5-fold cross-validation are:

Table 2: Mean Training Accuracy

	Dataset 0	**Dataset 1**	**Dataset 2**
Accuracy	1	1	1

Table 3: Mean Validation Accuracy

	Dataset 0	**Dataset 1**	**Dataset 2**
Accuracy	0.66	0.77	0.67

Therefore, the final prediction is made by training a Mismatch Kernel -Logistic Regression for datasets 0 and 2 and a Spectrum Kernel - Logistic regression for dataset 1.

The results are a score of **0.702** on the Public Leaderboard (rank 12^{th}/37), and a score of **0.714** on the Private Leaderboard (rank 6^{th}/37).

5 Conclusion

For this Kaggle challenge, I implemented a Kernel Logistic Regression model from scratch for DNA sequence classification. The main challenge was the computational cost of kernel matrix computation, which sometimes took a lot of time. Despite this, the final results were good, achieving a competitive final score and final rank.