Machine Learning — Assignment 2 – SVM

Sébastien Bouquet

November 2, 2018

Abstract

This assignment will present the functioning of Support Vector Machines (SVM's).

1 Theoretical Probability (5 points)

1.1 Question 1

For P(X) to be a density function, it is necessary that

$$\int_{-\infty}^{\infty} P(X = x)\, dx = 1$$

Since this function returns 0 for any $x \in X < 0$ because of the identity function component, we need to define k st.

$$\int_{0}^{\infty} 0.3e^{-x} + ke^{-2x}\, dx = 1$$

$$\big(-0.3e^{-x}\big)\Big|_{0}^{\infty} + \Big(-\frac{1}{2}ke^{-2x}\Big)\Big|_{0}^{\infty} = 1$$

$$\big(-0.3e^{-\infty} + 0.3 * e^{0}\big) + \Big(-\frac{1}{2}ke^{-2\infty} + \frac{1}{2}ke^{0}\Big) = 1$$

As $(\lim_{b\to\infty} e^{-b}) = 0$ we can remove the two terms containing it, which leaves

$$0.3 * 1 + \frac{1}{2}K * 1 = 1 \Rightarrow \mathbf{k} = 2 * (1 - 0.3) = \mathbf{0.4}$$

1.2 Question 2

The expectation of a density function is given by $E[X = x] = \int_a^b x * f(x)\, dx$ where $[a, b] = Range(f)$.

In our case the expectation is defined by

$$E[X] = \int_0^\infty x(0.3e^{-x} + ke^{-2x})\,dx = \int_0^\infty 0.3e^{-x}x\,dx + \int_0^\infty ke^{-2x}x\,dx$$

For $\int e^{-x}x$ and $\int e^{-2x}x$, we use the integration by parts ($\int f\,dg = fg - \int g\,df$), which gives $\int e^{-x}x = x(-e^{-x}) - \int -e^{-x}dx$ and $\int e^{-2x}x = (\frac{-1}{2}e^{-2x})x - \int \frac{-1}{2}e^{-2x}\,dx$:

$$= -0.3e^{-x}x\Big|_0^\infty + 0.3\int_0^\infty e^{-x}\,dx - \frac{1}{2}ke^{-2x}x\Big|_0^\infty + \frac{1}{2}k\int_0^\infty e^{-2x}\,dx$$

As $\big(-0.3(-e^{-x})x\big)\Big|_0^\infty = \big(\lim_{b\to\infty} -0.3e^{-b}b\big) - (-0.3e^0 * 0)$ resolves to 0, as well as $(-\frac{1}{2}ke^{-2x}x)\Big|_0^\infty$, we get $E[X]$

$$= -0.3\int_0^\infty e^{-x}\,dx + \frac{k}{2}\int_0^\infty e^{-2x}\,dx = 0.3 + \left(\frac{-k}{4}e^{-2x}\right)\Big|_0^\infty = \mathbf{0.3} + \frac{\mathbf{k}}{\mathbf{4}}$$

$$= \mathbf{0.4} \text{ where } \mathbf{k} = \mathbf{0.4}$$

2 Theory of SVMs (50 points)

2.1 Question 1

An SVM constructs an hyper-plane that linearly separates two classes of data, for data points of dimension n. To find the best class separation, the SVM maximizes its margin (the distance from the boundary to the closest support vector.) Doing this gives a higher probability that an added data point will still be classified correctly. It might happen however, that a data set is not linearly separable. To solve this problem, SVM's use kernel tricks to plot the data in a higher dimensional space than it originally used to be, in the hope that there exists a higher dimensional space in which the data will be linearly separable.

This approach of only separating classes linearly has the drawback that it can rapidly overfit the training data. This problem can be solved (or reduced) by allowing "soft margins", meaning that some data points can be inside the margin or even on the wrong side of the frontier but the distances between these points and the boundary will be summed up as a penalty.

Advantages of SVM's are:

- they work well when there is a clear margin of separation,
- they are effective in high dimensional spaces and they are memory efficient since only a subset of the training data (the support vectors) is used to compute the decision function,
- the function they try to optimize is a Quadratic programming problem which means that there are no local minima.

Disadvantages are:

- they are not computationally efficient for large data sets,
- they can't separate classes well when the classes are overlapping,
- the results they produce can be hard to interpret if the separation happened in a higher dimension.

2.2 Question 2

In the case of hard-margin, the objective function is to minimize $||\mathbf{W}||$ as the size of the margin is $M = \frac{2}{||\mathbf{W}||}$. Constraint: $Y_i(f(X_i)) > 0 \, \forall i = 1, ..., N$; this constraint can be rescaled to: $Y_i(f(X_i)) \geq 1 \, \forall \, i$.

For soft-margin, we need to factor in penalty variables, which are defined by:

$$\xi_i \text{ (or } \epsilon_i) = \begin{cases} 0 & \text{if } X_i \text{ is inside or on the correct margin boundary} \\ |Y_i - f(X_i)| & \text{otherwise} \end{cases} \tag{1}$$

The objective function is therefore:

$$\min_{\mathbf{W},b} \left(\frac{1}{2}||\mathbf{W}||^2 + C \sum_{n=1}^{N} \xi_n \right) \text{, where C is a constant}$$

with constraints: $Y_i(\mathbf{W}^T \Phi(X_i) + b) \geq 1 - \xi_i \, , \, i = 1, ..., N$ and $\xi_i \geq 0 \, , \, i = 1, ..., N$.

2.3 Question 3

The kernel trick is the use of a kernel function to map non linearly separable data sets to a higher dimensional space with the hope that the data will become linearly separable. The kernel trick stays efficient because it does not need to compute the actual positions of the data in a higher dimension, but only their inner products.

A function $\mathcal{K}(X, Y)$ is a valid kernel if it satisfies Mercer's condition which states that a function $\mathcal{K}(x, y)$ satisfies the condition if one has:

$$\iint g(x)\mathcal{K}(x, y)g(y)\,dx\,dy \geq 0$$

for all square-integrable functions $g(x)$.

This means that $\mathcal{K}$ can be expressed as a product of Φ functions such that $\mathcal{K}(x, y) = \Phi(x)\Phi(y) = \Phi(y)\Phi(x) = \mathcal{K}(y, x)$.

2.4 Question 4

That problem is equivalent to:

$$\max_{\alpha} \left(\sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m Y_n Y_m \Big(\Phi(X_n)\Phi(X_M) \Big) \right)$$

where $(\Phi(X_n)\Phi(X_M))$ is a kernel $\mathcal{K}(X_n, X_m)$

2.5 Question 5

The decision function would be

$$f_d(x) = \mathbf{W}^T \Phi(x) + b = \sum_{i=1}^{N} \alpha_i y_i \mathcal{K}(x_i, x) + b$$

where $\mathcal{K}(x_i, x) = \Phi(x_i)^T \Phi(x)$ is a kernel.
A point x is classified in the first class if $f_d(x) > 0$ and in the second class if $f_d(x) < 0$.

2.6 Question 6

In the case of hard-margin SVM's, the only hyper parameter is the choice of a kernel function. The kernel changes the probability of finding a linear separation of the data in a higher dimension. Some kernels also have hyper-parameters on their own; for example, an RBF kernel has a γ parameter that will have to be tuned as well. The effect of these parameters depends on the chosen kernel.

In the case of soft-margin SVM's, the kernel function is still an hyper-parameter. Another one is the choice of the constant C that modifies the sum of the penalties. A lower C allows for more generalization in the decision boundary by allowing more data points inside the margin or in the wrong side of the boundary. A higher C increases the total penalty which reduces the chances of miss-classified data points but increases the chances of overfitting.

3 Kernels I (15 points)

3.1 $K(x, y) = x^T y + (x^T y)^2$

This is a valid kernel because it can be expressed (in $\mathbb{R}^2$) as:

$$K(x, y) = x^T y + (x^T y)^2 = x_1 y_1 + x_2 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2 + 2x_1 x_2 y_1 y_2$$

$$= (x_1, x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)(y_1, y_2, y_1^2, y_2^2, \sqrt{2}y_1 y_2) = \Phi(x)\Phi(y)$$

This construction can be generalized for $d \geq 1$.

3.2 $K(x, y) = x^2 e^{-y}$, $d = 1$

This is not a valid kernel because it cannot be expressed as a product of $\Phi(x)\Phi(y)$.

3.3 $K(x, y) = c{*}k_1(x, y){+}k_2(x, y)$, where k_1, k_2 are valid kernels in $\mathbb{R}^d$

We know that $K(x, y) = k_1(x, y) + k_2(x, y)$ is a valid kernel if $k_1(x, y), k_2(x, y)$ are valid kernels since $\Phi_1(x)\Phi_1(y)+\Phi_2(x)\Phi_2(y) = (\Phi_1(x), \Phi_2(x))(\Phi_1(y), \Phi_2(y))$. We also know that $K(x, y) = c * k_1(x, y)$ is a valid kernel if k_1 is a valid kernel.

Therefore $\mathbf{K}(\mathbf{x}, \mathbf{y}) = c*k_1(x, y)+k_2(x, y)$ **is a valid kernel** if k_1, k_2 are valid kernels since $K(x, y) = c*\Phi_1(x)\Phi_1(y)+\Phi_2(x)\Phi_2(y) = (c\Phi_1(x), \Phi_2(x))(c\Phi_1(y), \Phi_2(y))$.

4 Kernels II (16 points)

4.1 Point a

For this data set, which is not linearly separable, I would choose an RBF kernel. First because it maps the data to a higher dimensional space which means it can handle non linear classification.

A second reason, is that an RBF kernel is less complex than other kernels like a polynomial kernel because it has less hyper-parameters. It has also few numerical difficulties: an RBF kernel will always return a value superior to 0 and smaller or equal to 1, contrary to a polynomial kernel which can return infinite values or zero values.
The sigmoid kernel could also be a choice but it is also more complex to implement, as it is not always a valid kernel, depending on the parameters.

4.2 Point b

For this data set, we can see that the data is not linearly separable but a simple transformation to $\mathbb{R}^3$ would make it linearly separable. However I would still choose an RBF kernel unless number of features of the data is very high which should not be the case since the given data has only 2 dimensions. The reasons for this choice are the same as the ones given in Point a.

4.3 Point c

Since this data is visually linearly separable, I would choose a linear kernel to classify this data. The main argument is that, although a non-linear kernel may give similar results, the computation would be slower, the tuning of the hyper-parameter would take more time and non-linear kernels create a higher chance of over-fitting to the training data. That is why, following Occam's Razor, I would rather choose the simplest working solution.

4.4 Point d

Since this data set is clearly non linearly separable, I would choose an RBF kernel. The reasons are the same than the ones given in Point A. If, after testing the results are unsatisfying with an RBF Kernel, then only would I try another non-linear kernel like a sigmoid or a polynomial.

5 SVMs (14 points)

5.1 Question 1

Yes, the decision boundary would be an hyper-plane of dimension $2 - 1 = 1$ and an hyper-plane of dimension 1 is essentially a straight line.

5.2 Question 2

No, the C parameter can have an influence on the W returned even in the case of a linearly separable data set because the α_n, corresponding to a support vector, has an upper bounded set by C.

5.3 Question 3

The decision boundary would be a $3 - 1 = 2$ dimensional hyper-plane, ie. a two-dimensional plane.

5.4 Question 4

The computational effort increases for SVM's using a non-linear kernel function. This happens because the kernel function computes the inner products of data points in a higher dimensional space and the computation of the dot product depends on the number of features (ie. dimensions) of the data. Hence a higher dimension means a greater computational effort.

5.5 Question 5

This is false because $(\tilde{x}_4, y_4)$ may not be a support vector but (x_4, y_4) might have been a support vector had it not been replaced by wrong values. Then, after

the training the missing support vector would have been replaced by another one.

References

[1] S. Raschka *How to Select Support Vector Machine Kernels.* Michigan State University.
https://www.kdnuggets.com/2016/06/select-support-vector-machine-kernels.html

[2] H. Bhavsar, M. H. Panchal *A Review on Support Vector Machine for Data Classification.* International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
http://ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-10-185-189.pdf

[3] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin *A Practical Guide to Support Vector Classification.* National Taiwan University
https://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf