Foundations of Machine Learning
Non linear SVM and Kernel Methods

Lecturer: Yann Chevaleyre
Scribe: Mathilde Kretz

Lecture n°8
23/11/2023

1 Non linear SVM

1.1 Introduction

SVM are machine learning methods for classification and regression aiming to find the optimal hyperplan that best separates classes in a space maximizing the margin between them. We have seen the linear SVM problem formulation considering linearly separable (and not) problems. Given the learning data $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^n$, the linear SVM primal problem formulation is the following:

$$\begin{aligned} \min_{w,b,\{\xi_i\}_i} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^n \xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w}^\top \mathbf{x_i} + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i \in [\![1, n]\!] \end{aligned}$$

The dual problem is then defined as follows:

$$\begin{aligned} \max_{\{\alpha_i\}_i} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \mathbf{x_i}^\top \mathbf{x_j} \\ \text{subject to:} \quad & 0 \leq \alpha_i \leq C \quad \forall i \in [\![1, n]\!] \\ & \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

The decision function which is in the primal problem $f(\mathbf{x}) = \mathbf{w}^\top\mathbf{x}+b$, then becomes $f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i \mathbf{x_i}^\top \mathbf{x}+b$ as we have $\mathbf{w} = \sum_{i=1}^n \alpha_i y_i \mathbf{x_i}$.
There exists, however, situations where points are separable but non-linear, as in example 1. In such cases, a non-linear transformation is employed as a solution.

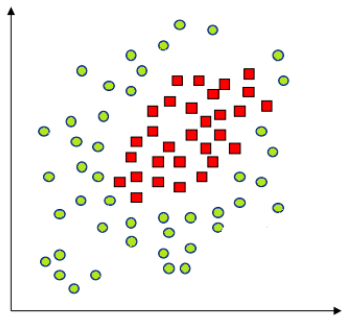


Figure 1: Separable problem but non linear.

1.2 Non linear transformation

Example - Polar coordinates

The data points are not linearly separable in the space in Cartesian coordinates but they are in polar coordinates. The non linear transformation of the original space (from Cartesian to polar) acts as a change of basis.

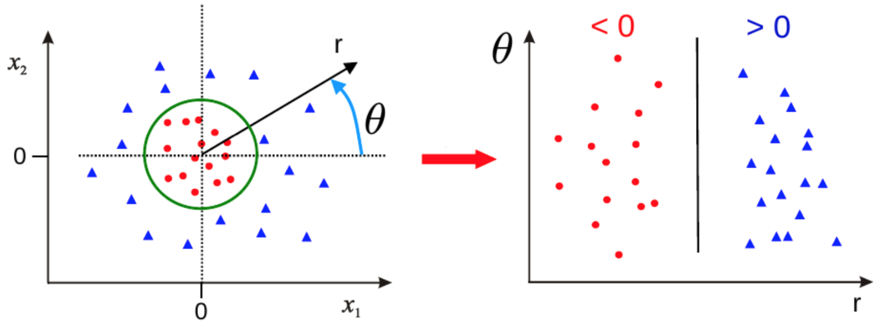


Figure 2: Transformation via polar coordinates.

It can be formalized with $\phi\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}\right) = \begin{bmatrix}r\\\theta\end{bmatrix}$.

Example - Projection in a higher dimension space

The idea remains the same, we operate a transformation ϕ which projects the input $\mathbf{x}$ in a higher dimension space in which there exist an hyperplan that linearly separates the data.

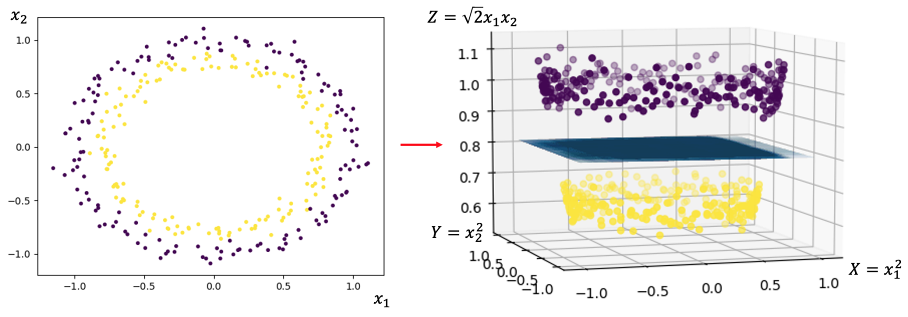


Figure 3: Transformation in higher dimensional space.

It can also be formalized with a transformation ϕ such that $\phi\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}\right) = \begin{bmatrix}x_1^2\\x_2^2\\\sqrt{2}x_1x_2\end{bmatrix}$.
With this transformation, we can learn a hyperplan in $\mathbb{R}^3$ (in blue in the example 3) such that $\mathbf{wV} + b = 0$, with $\mathbf{V} = \begin{bmatrix}X\\Y\\Z\end{bmatrix}$. This is equivalent to $w_1x_1^2 + w_2x_2^2 + \sqrt{2}w_3x_1x_2 + b = 0 \Leftrightarrow \mathbf{w}\phi(\mathbf{x}) + b = 0$. We note that $\mathbf{wV} + b = 0$ is linear in $\mathbb{R}^3$, which is the desired goal, while $w_1x_1^2 + w_2x_2^2 + \sqrt{2}w_3x_1x_2 + b = 0$ is not linear in $\mathbb{R}^2$.

Remark. In the higher dimensional space (here 3D space), all the data points are on a manifold of the original dimension (here 2).

Non linear principles

In the non linear SVM case, the goal is still to learn a linear classifier f, however, this one is located in the new space $\mathbb{R}^D$ such that $f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b$ and where ϕ is the transformation function $\phi : \mathbb{R}^d \to \mathbb{R}^D$.

The function f thus defined is the new decision function of the new primal problem:

$$\begin{aligned} \min_{w,b,\{\xi_i\}_i} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w}^\top \phi(\mathbf{x_i}) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i \in [\![1, n]\!] \end{aligned}$$

We note that now $\mathbf{w} \in \mathbb{R}^D$.

The associated dual problem thus follows:

$$\begin{aligned} \max_{\{\alpha_i\}_i} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x_i})^\top \phi(\mathbf{x_j}) \\ \text{subject to:} \quad & 0 \leq \alpha_i \leq C \quad \forall i \in [\![1, n]\!] \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned}$$

And the associated decision function becomes $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x})^\top \phi(\mathbf{x_i}) + b$. This function is indeed linear in ϕ (in $\mathbb{R}^D$ the new space) while not linear in $\mathbf{x}$ (in $\mathbb{R}^d$ the original space $\mathcal{X}$).

2 Kernel methods

2.1 Kernel trick

Intuition

When resolving the dual problem as defined just above, we have to compute the product $\phi(\mathbf{x_i})^\top \phi(\mathbf{x_j})$ which is the product between two vectors of $\mathbb{R}^D$. This can be long to compute compared to the original problem of $\mathbb{R}^d$ especially if $D \gg d$. Let's go back to the example 1.2 of the 3D transformation and calculate it.

$$\begin{aligned} \phi(\mathbf{x_i})^\top \phi(\mathbf{x_j}) &= \begin{bmatrix} x_{i,1}^2 & x_{i,2}^2 & \sqrt{2}x_{i,1}x_{i,2} \end{bmatrix} \begin{bmatrix} x_{j,1}^2 \\ x_{j,2}^2 \\ \sqrt{2}x_{j,1}x_{j,2} \end{bmatrix} \\ &= x_{i,1}^2 x_{j,1}^2 + x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} \\ &= (x_{i,1}x_{j,1} + x_{i,2}x_{j,2})^2 \\ &= (\mathbf{x_i}^\top \mathbf{x_j})^2 \end{aligned}$$

This last term is a dot product of two vectors from $\mathbb{R}^d$ (here $\mathbb{R}^2$) which is very interesting compared to the initial computation.

Kernel trick

The concept underlying the kernel trick is that the scalar product $\phi(\mathbf{x_i})^\top\phi(\mathbf{x_j})$ in $\mathbb{R}^D$ can be expressed as a scalar product in $\mathbb{R}^d$. Consequently a kernel function k, implicitly defines the transformation allowing us to represent it as $k(\mathbf{x},\mathbf{y}) = \langle\phi(\mathbf{x}),\phi(\mathbf{y})\rangle$. In the previous example, we had $k(\mathbf{x},\mathbf{y}) = (\mathbf{x}^\top\mathbf{y})^2$.

Subsequently, the dual problem and the decision function of it are re defined (again):

$$\begin{aligned} \max_{\{\alpha_i\}_i} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j k(\mathbf{x_i}\mathbf{x_j}) \\ \text{subject to:} \quad & 0 \leq \alpha_i \leq C \quad \forall i \in [\![1,n]\!] \\ & \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i k(\mathbf{x},\mathbf{x_i}) + b$$

We note that the high dimension D indeed does not appear in the dual problem, making it easier to compute.

2.2 Theoretical framework

For generalization, we adjust the notations such that the transformation $\phi : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is the feature space endowed with the dot product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$. The kernel k is then a function $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ whose idea is to compute the similarity between both of its inputs.

Definition 2.1 (Positive Definite Kernel)**.** A kernel $k(\mathbf{x},\mathbf{y})$ is said to be positive definite if:
- it is symetric: $k(\mathbf{x},\mathbf{y}) = k(\mathbf{y},\mathbf{x})$
- $\forall n \in \mathbb{N}$,

$$\forall\{\alpha_i\}_{i=1}^n \in \mathbb{R}, \forall\{\mathbf{x_i}\}_{i=1}^n \in \mathcal{X}, \quad \sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j k(\mathbf{x_i},\mathbf{x_j}) \geq 0$$

Remark. A kernel is stricly positive definite if $\sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j k(\mathbf{x_i},\mathbf{x_j}) > 0$ for $\alpha_i \neq 0$.

Remark. Let the matrix $K \in \mathbb{R}^{n\times n}$ such that $K_{i,j} = k(\mathbf{x_i},\mathbf{x_j})$. Saying that the kernel k is (strictly) positive definite is equivalent to say that the matrix K is positive semi definite.

Definition 2.2 (the Gram Matrix)**.** Given a kernel $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$, the Gram Matrix K is from $\mathbb{R}^{n\times n}$ and such that $K_{i,j} = k(\mathbf{x_i},\mathbf{x_j})$.

Proposition 1. *For any set of samples $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^n$, the associated Gram Matrix $K \in \mathbb{R}^{n\times n}$ is positive semi definite iff k is a positive definite kernel on $\mathcal{X}$.*

2.3 Kernels properties

Linear kernel

Definition 2.3 (Linear Kernel)**.** A kernel k is linear if it is of the form $k(\mathbf{x},\mathbf{y}) = \mathbf{x}^\top\mathbf{y}$. This is equivalent to the transformation $\phi(\mathbf{x}) = \mathbf{x}$.

Proof. - We have $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ - It is symetric as $\mathbf{x}^\top \mathbf{y} = \mathbf{y}^\top \mathbf{x}$ - It is positive as:

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j k(\mathbf{x_i}, \mathbf{x_j}) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j \mathbf{x_i}^\top \mathbf{x_j} \\ &= \left(\sum_{i=1}^{n}\alpha_i \mathbf{x_i}\right)^\top \left(\sum_{j=1}^{n}\alpha_j \mathbf{x_j}\right) \\ &= \|\sum_{i=1}^{n}\alpha_i \mathbf{x_i}\|^2 \geq 0\end{aligned}$$

□

Product kernel

Definition 2.4 (Product Kernel)**.** A kernel k is a product if it is of the form $k(\mathbf{x}, \mathbf{y}) = g(\mathbf{x})g(\mathbf{y})$ for some $g : \mathcal{X} \to \mathbb{R}$.

Proof. - We have $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ - It is symetric by construction - It is positive as:

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j k(\mathbf{x_i}, \mathbf{x_j}) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j g(\mathbf{x_i})g(\mathbf{x_j}) \\ &= \left(\sum_{i=1}^{n}\alpha_i g(\mathbf{x_i})\right)\left(\sum_{j=1}^{n}\alpha_j g(\mathbf{x_j})\right) \\ &= \left(\sum_{i=1}^{n}\alpha_i g(\mathbf{x_i})\right)^2 \geq 0\end{aligned}$$

□

There are several forms of kernel that exist such as quadratic, Gaussian or finite.

Aronszajn's theorem

Theorem 2.1 (Aronszajn's theorem)**.** *k is a positive definite kernel on $\mathcal{X}$ if and only if, there exists a Hilbert space $\mathcal{H}$ and a mapping $\phi : \mathcal{X} \to \mathcal{H}$ such that*

$$\forall x, y \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{\mathcal{H}}$$

Remark. Aronszajn's theorem defines the link between a kernel function k and the existence of a Hilbert space that "represents" it. Such Hilbert spaces are called *Reproducing Kernel Hilbert Spaces* and are presented in the following section.

2.4 Reproducing Kernel Hilbert Space

Intuition

Aronszajn's theorem defines the need of a particular Hilbert space that represent k and whose link is made by the transformation function ϕ. The question is then: how can we find ϕ given a kernel k?

The intuition behind the answer is, giving the kernel k we want use, build the Hilbert space as a space of functions from $\mathcal{X} \to \mathbb{R}$. In this way the mapping ϕ is defined as

$$\begin{aligned}\phi : \mathcal{X} &\to \mathbb{R}^{\mathcal{X}} \\ \mathbf{x} &\mapsto k(\cdot, \mathbf{x})\end{aligned}$$

RKHS

Definition 2.5 (RKHS)**.** Let $\mathcal{X}$ be a set and $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ be a class of function forming a real Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. The function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a reproducing kernel of $\mathcal{H}$ if:

- $\mathcal{H}$ contains all functions of the form:

$$\forall \mathbf{x} \in \mathcal{X}, \quad k(\mathbf{x}, \cdot) : \mathbf{z} \mapsto k(\mathbf{x}, \mathbf{z})$$

-for every $\mathbf{x} \in \mathcal{X}$ and $f \in \mathcal{H}$, the **reproducing property** of k holds:

$$f(\mathbf{x}) = \langle f, k(\mathbf{x}, \cdot) \rangle_{\mathcal{H}}$$

If a reproducing kernel exists, $\mathcal{H}$ is called a reproducing kernel Hilbert space (RKHS).

Method 2.1 (RKHS for Machine Learning Problem)**.** We can use the RKHS as a hypothesis space for Machine Learning problem that requires non linear models. The method is the following:

- Maps the data $\mathbf{x} \in \mathcal{X}$ to a higher dimensional RKHS $\mathcal{H}$ with the mapping $\phi : \mathcal{X} \to \mathcal{H}$ with $\phi(\mathbf{x}) = k(\cdot, \mathbf{x})$.

-In the space $\mathcal{H}$, consider a linear model $f(\mathbf{x}) = \langle f, k(\mathbf{x}, \cdot) \rangle_{\mathcal{H}}$. Use then this linear model in the learning problem. For supervised learning we would have, for example

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{n} \ell(y_i, f(\mathbf{x_i})) + \lambda \|f\|_{\mathcal{H}}^2$$

Theorem 2.2. *A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite if and only if it is a reproducing kernel.*

A COMPLÉTER ?

3 Applications

We have seen that kernel function can be used in various situations. Actually many problems can be resolved with kernel functions.

3.1 Kernelized 1-Nearest Neighbor

We still consider a data set $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^n$. The standard decision function of the 1-Nearest Neighbor problem is $f_{1-NN}(\mathbf{x}) = y_{\hat{i}}$ such that $\hat{i} = \arg\min_{i \in [\![1,n]\!]} \|\mathbf{x}_i - \mathbf{x}\|$. $\hat{i}$ is the index of the closest neighbor of $\mathbf{x}$, it takes then its label.

Assume we want to apply the 1-NN problem in another space $S = \{(\phi(\mathbf{x_i}), y_i)\}_{i=1}^n$ because in this space the 1-NN clustering is possible - just as points were linearly separable previously. We then need to compute the distance between new vectors:

$$\begin{aligned}\|\phi(\mathbf{x_i}) - \phi(\mathbf{x})\|^2 &= (\phi(\mathbf{x_i}) - \phi(\mathbf{x}))^\top (\phi(\mathbf{x_i}) - \phi(\mathbf{x})) \\ &= \phi(\mathbf{x_i})^\top \phi(\mathbf{x_i}) + \phi(\mathbf{x})^\top \phi(\mathbf{x}) - 2\phi(\mathbf{x_i})^\top \phi(\mathbf{x}) \\ &= k(\mathbf{x_i}, \mathbf{x_i}) + k(\mathbf{x}, \mathbf{x}) - 2k(\mathbf{x_i}, \mathbf{x})\end{aligned}$$

The new decision function in this space thus becomes:

$$f_{\text{kernelized 1-NN}}(\mathbf{x}) = y_{\hat{i}}, \quad \text{with } \hat{i} = \arg\min_i \left(k(\mathbf{x_i}, \mathbf{x_i}) + k(\mathbf{x}, \mathbf{x}) - 2k(\mathbf{x_i}, \mathbf{x})\right)$$

Here again, the computation of the distance in the new space does not involve the high dimension of the new space.

3.2 Kernel Ridge Regression

It may also be possible to perform a regularized regression in a new space. But first let's recall the original problem.

The original problem

We consider the same data set $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^n \in \mathcal{X} \times \mathbb{R}$ where $\mathcal{X} = \mathbb{R}^d$ and we will note $\mathbf{X} = \begin{bmatrix} \mathbf{x_1} \\ \vdots \\ \mathbf{x_n} \end{bmatrix} \in \mathbb{R}^{n \times d}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n$. The optimization problem considering a linear model $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$ is the following:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

By deriving this equation along $\mathbf{w}$, we obtain the following optimality condition and solution:

$$\begin{aligned} & -\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^*) + \lambda \mathbf{w}^* = 0 \\ \iff & -\mathbf{X}^\top \mathbf{y} + \mathbf{X}^\top \mathbf{X}\mathbf{w}^* + \lambda I_d \mathbf{w}^* = 0 \\ \iff & (\mathbf{X}^\top \mathbf{X} + \lambda I_d)\mathbf{w}^* = \mathbf{X}^\top \mathbf{y} \\ \iff & \mathbf{w}^* = (\mathbf{X}^\top \mathbf{X} + \lambda I_d)^{-1}(\mathbf{X}^\top \mathbf{y}) \quad \textit{(solution)} \\ \iff & \mathbf{w}^* = \frac{1}{\lambda}\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}^*) \quad \textit{(rewriting)} \end{aligned}$$

Then, by considering the reformulated problem with $\alpha \in \mathbb{R}^n$, the optimality condition is rewritten as it exists α^* such that $\mathbf{w}^* = \mathbf{X}^\top \alpha^*$:

$$\begin{aligned} & \mathbf{X}^\top \alpha^* = \frac{1}{\lambda}\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{X}^\top \alpha^*) \\ \iff & \mathbf{X}^\top(I_d + \frac{1}{\lambda}\mathbf{X}\mathbf{X}^\top)\alpha^* = \frac{1}{\lambda}\mathbf{X}^\top \mathbf{y} \end{aligned}$$

Consider $\hat{\alpha}$ such that $(I_d + \frac{1}{\lambda}\mathbf{X}\mathbf{X}^\top)\hat{\alpha} = \frac{1}{\lambda}\mathbf{y}$. If such a $\hat{\alpha}$ exists, then

$$\alpha^* = \hat{\alpha} = (\lambda I_d + \mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{y}$$

The decision function and the learning problem are reformulated as following:

$$f(\mathbf{x}) = \sum_{i=0}^{n} \alpha_i \mathbf{x_i}^\top \mathbf{x}$$

$$\min_{\alpha} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\mathbf{X}^\top \alpha\|_2^2 + \frac{\lambda}{2} \alpha^\top \mathbf{X}\mathbf{X}^\top \alpha$$

We note that the original problem solution depends on $\mathbf{X}^\top\mathbf{X}$ that represents a covariance matrix of $\mathbb{R}^{d\times d}$ while the reformulated problem solution depends on $\mathbf{X}\mathbf{X}^\top = \mathbf{K}$ the Gram matrix 2.2 of $\mathbb{R}^{n\times n}$.

The kernelized problem

If we apply the change of representation $\phi : \mathbf{x} \mapsto \phi(\mathbf{x})$, we then replace $\mathbf{X}$ by $\begin{bmatrix} \phi(\mathbf{x_1}) \\ \vdots \\ \phi(\mathbf{x_n}) \end{bmatrix}$. The Gram Matrix $\mathbf{K}$ which previously was $\mathbf{X}\mathbf{X}^\top$ now becomes $[k(\mathbf{x_i}, \mathbf{x_j})]_{i,j} = \left[\phi(\mathbf{x_i})^\top \phi(\mathbf{x_j})\right]_{i,j}$.

The learning problem, adapted from the reformulated version of the original one is:

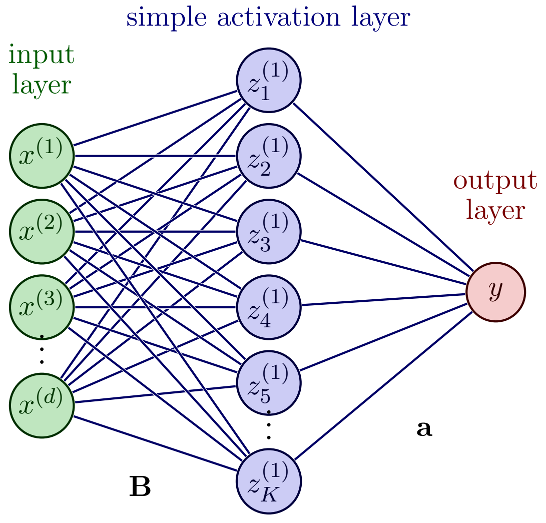
$$\begin{aligned}
&\min_{\alpha} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\mathbf{X}^\top \alpha\|_2^2 + \frac{\lambda}{2} \alpha^\top \mathbf{X}\mathbf{X}^\top \alpha \\
\iff &\min_{\alpha} \frac{1}{2} \|\mathbf{y} - \mathbf{K}\alpha\|_2^2 + \frac{\lambda}{2} \alpha^\top \mathbf{K}\alpha \\
\iff &\min_{\alpha} \sum_{i=1}^{n} \Big(y_i - \alpha^\top \left[k(\mathbf{x1}, \mathbf{x_i}) \quad \dots \quad k(\mathbf{x_n}, \mathbf{x_i})\right]\Big)^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x_i}, \mathbf{x_j})
\end{aligned}$$

Because the reformulation of the problem depends only on $\mathbf{X}\mathbf{X}^\top = \mathbf{K}$, the mapping representing the change of representation does not affect the calculation of the decision function and the whole problem that depends on the (lower dimension) kernel function k.

3.3 Applications on Neural Networks

3.3.1 Random Features Model

Considering any Neural Network sufficiently wide. For simplicity, we will consider a single layer NN in the following section.



In this neural network, the decision function is of the following form: $f(\mathbf{x}) = \frac{1}{\sqrt{K}} \sum_{j=1}^{K} a_j \sigma(\mathbf{B}_j^\top \mathbf{x})$, with $\mathbf{a} \in \mathbb{R}^K, \mathbf{B} \in \mathbb{R}^{d \times K}, \mathbf{x} \in \mathbb{R}^d$. (For simplification we didn't include any bias).

It can be shown that any 'simple' sufficiently wide Neural Network works as a kernel. Assume in the first place that each element of the weight matrix $\mathbf{B}$ is randomly distributed ($\sim \mathcal{N}(0,1)$) and the matrix is fixed - ie. not learned. We thus have $f(\mathbf{x}) = \frac{1}{\sqrt{K}} a^\top \phi(\mathbf{x})$ with $\phi(\mathbf{x}) = \begin{bmatrix} \sigma(\mathbf{B_1}^\top \mathbf{x}) \\ \vdots \\ \sigma(\mathbf{B_K}^\top \mathbf{x}) \end{bmatrix}$. Assuming the NN is infinitely wide ($K \to \infty$), the underlying kernel is then:

$$\begin{aligned} k(\mathbf{x}, \mathbf{x'}) &= \lim_{K \to +\infty} \langle \phi(\mathbf{x}), \phi(\mathbf{x'}) \rangle \\ &= \lim_{K \to +\infty} \frac{1}{K} \sigma(\mathbf{Bx})^\top \sigma(\mathbf{Bx'}) \\ &= \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(0, I_d)} \left[\sigma(\mathbf{wx})^\top \sigma(\mathbf{wx'}) \right] \\ &= \mathbb{E}_{\mathbf{u}, \mathbf{v} \sim \mathcal{N}(0, \Lambda)} \left[\sigma(\mathbf{u})^\top \sigma(\mathbf{v}) \right] \quad \text{with} \quad \Lambda = \begin{pmatrix} \|\mathbf{x}\|^2 & \mathbf{x}^\top \mathbf{x'} \\ \mathbf{x}^\top \mathbf{x'} & \|\mathbf{x'}\|^2 \end{pmatrix} \end{aligned}$$

If the activation function σ is taken as ReLu, we can show that:

$$k(\mathbf{x}, \mathbf{x'}) = \frac{1}{\pi} \Big((\mathbf{x}^\top \mathbf{x'})(\pi - \arccos(\mathbf{x}^\top \mathbf{x'})) \Big) + \sqrt{1 - (\mathbf{x}^\top \mathbf{x'})^2}$$

.

A still thriving field of research tends to prove that learning a infinitely wide Neural Network is equivalent to apply the Kernel Ridge Regression of the kernel k defined above.

3.3.2 Neural Tangent Kernels

The idea behind Neural Tangent Kernels is similar to the one of Random Features Model except that we lift the assumption on frozen weights. Considering a neural network with all its weights learned and an unlimited number of layers and nodes. It can be resumed as a function $f(\mathbf{w}, \mathbf{x}) \to \mathbb{R}$.

Let's define $f_\mathbf{x}(\mathbf{w}) = f(\mathbf{x}, \mathbf{w})$, the prediction of the neural network on $\mathbf{x}$ based on the weights $\mathbf{w}$. Let $\mathbf{w_0}$ the initial weights vector. The Taylor expansion of $f_\mathbf{x}(\mathbf{w})$ around $\mathbf{w_0}$ is:

$$f_\mathbf{x}(\mathbf{w}) \underset{\mathbf{w} \to \mathbf{w_0}}{=} f_\mathbf{x}(\mathbf{w_0}) + \langle \mathbf{w} - \mathbf{w_0} \rangle^\top \nabla_{\mathbf{w_0}} f_\mathbf{x}(\mathbf{w_0}) + \underbrace{\cdots}_{\text{higher order terms}}$$

We can show that the higher order terms vanish the more the neural network grows. That means that it becomes linear in $\mathbf{w}$ around $\mathbf{w_0}$ as the network grows wider.

Example

If we consider for simplicity that $f_\mathbf{x}(\mathbf{w_0}) = \mathbf{w_0}$, at the neighborhood of $\mathbf{w_0}$, we have

$$\begin{aligned} f_\mathbf{x}(\mathbf{w}) &\approx \langle \mathbf{w} - \mathbf{w_0} \rangle^\top \nabla_{\mathbf{w_0}} f_\mathbf{x}(\mathbf{w_0}) \\ &= \mathbf{w}^\top \phi(\mathbf{x}) + \text{cste} \\ &\qquad\qquad \text{with } \phi(\mathbf{x}) = \nabla_{\mathbf{w_0}} f_\mathbf{x}(\mathbf{w_0}) \end{aligned}$$

This function can be written as dependent on a mapping *phi* and the kernel function is then the following:

$$k(\mathbf{x}, \mathbf{x}') = \langle \nabla_{\mathbf{w_0}} f_{\mathbf{x}}(\mathbf{w_0}), \nabla_{\mathbf{w_0}} f_{\mathbf{x}'}(\mathbf{w_0}) \rangle$$

We note that this forms implies that the gradient must not change for the kernel to have the right properties. This is true if the region is linear, and validated if the network is wide.