NYU Paris **Spring 2019**

CSCI-UA 9473 Introduction to Machine Learning

Problem Set 3: Linear classification + Kernel methods (Part I)

The exercises vary in difficulty. More advanced exercises are marked with a star (*).

1. Classification

Exercise 1 (source: F. Lauer, LORIA)**.** *A dataset* $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}$ *with* $y_i \in \{+1, -1\}$ *is linearly separable if* $\exists(\boldsymbol{\beta}_1, \beta_0) \in \mathbb{R}^{N+1}$ *such that* $\text{sign}(\langle\boldsymbol{\beta}_1, \boldsymbol{x}_i\rangle + \beta_0) = y_i$, $i = 1, \ldots, M$

- *A notorious example of a non linearly separable dataset is the XOR dataset shown in table 1 below. Prove that this dataset is not linearly separable.*
- *Show that for binary classification (i.e.* $y_i = \{0, 1\}$*), a dataset is linearly separable if the set of linear inequalities* (1) *is feasible*

$$y_i(\langle\boldsymbol{\beta}_1, \boldsymbol{x}_i\rangle + \beta_0) > 0, \quad \textit{for all } i \tag{1}$$

Exercise 2 (source: Bishop)**.** *Given a set of points* $\{\boldsymbol{x}_j\}_{j=1}^N$*, one can define their convex hull to be the set of points* $\boldsymbol{x}$ *that can be written as*

$$\boldsymbol{x} = \sum_k \alpha_k \boldsymbol{x}_k \tag{2}$$

for some non negative coefficients $\alpha_k \geq 0$ *that satisfy* $\sum_k \alpha_k = 1$*. Now consider a second set of points* $\left\{\boldsymbol{y}_j\right\}_{j=1}^N$*. The two sets of points are linearly separable if there exists a vector* $\boldsymbol{w}$ *and a scalar* w_0 *such that* $\boldsymbol{w}^T\boldsymbol{x}_j + w_0 > 0$ *for all* $\boldsymbol{x}_j$ *and* $\boldsymbol{w}^T\boldsymbol{y}_j + w_0 < 0$ *for all* $\boldsymbol{y}_j$*. Show that if their convex hulls intersect, the two sets of points cannot be linearly separable, and conversely, if they are linearly separable, their convex hulls do not intersect.*

Exercise 3 (source: EE236A, Vandenberghe)**.** *Fig. 1 below shows a block diagram of a linear classifier. The classifier has n inputs* x_i*. These inputs are multiplied by coefficients* a_i *and then added. The result* $\boldsymbol{a}^T\boldsymbol{x} = \sum_{i=1}^N a_i x_i$ *is then compared to a threshold b (i.e passed into the step function). If* $\boldsymbol{a}^T\boldsymbol{x} \geq b$ *then the output is* $y = 1$*. If* $\boldsymbol{a}^T\boldsymbol{x} < b$*, the output is* $y = -1$*. As we saw, the geometric interpretation of such simple classifier is the following: The equation* $\boldsymbol{a}^T\boldsymbol{x} = b$ *defines a hyperplane with normal vector* $\boldsymbol{a}$*. The hyperplane divides the space* $\mathbb{R}^n$ *into two open halfspaces. One in which* $\boldsymbol{a}^T\boldsymbol{x} > b$*, the other where* $\boldsymbol{a}^T\boldsymbol{x} < b$*. The output of the classifier is* $+1$ *or* -1 *depending on the halfspace in which* $\boldsymbol{x}$ *lies. If* $\boldsymbol{a}^T\boldsymbol{x} = b$*, we arbitrarily assign* $+1$ *to the output (see Fig. 2). By combining simple linear classifiers, one can build a classifier that divides* $\mathbb{R}^n$ *in more complicated regions than halfspaces. As an illustration of this, consider Fig. 3 which combines* 4 *linear classifiers. The first three take a unique vector* $\boldsymbol{x}$ *as their input. The fourth one takes as input the*

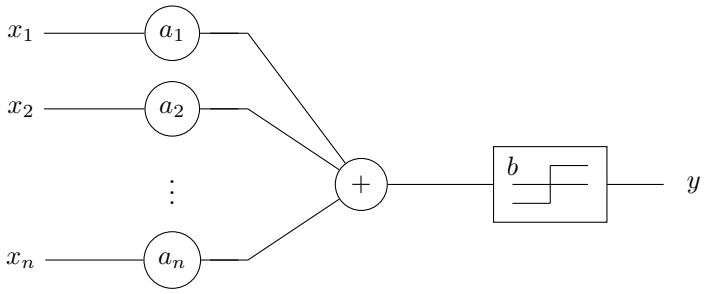

FIGURE 1. Linear classifier (block diagram).

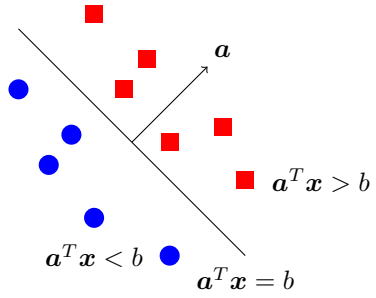

FIGURE 2. Separating hyperplane and normal vector

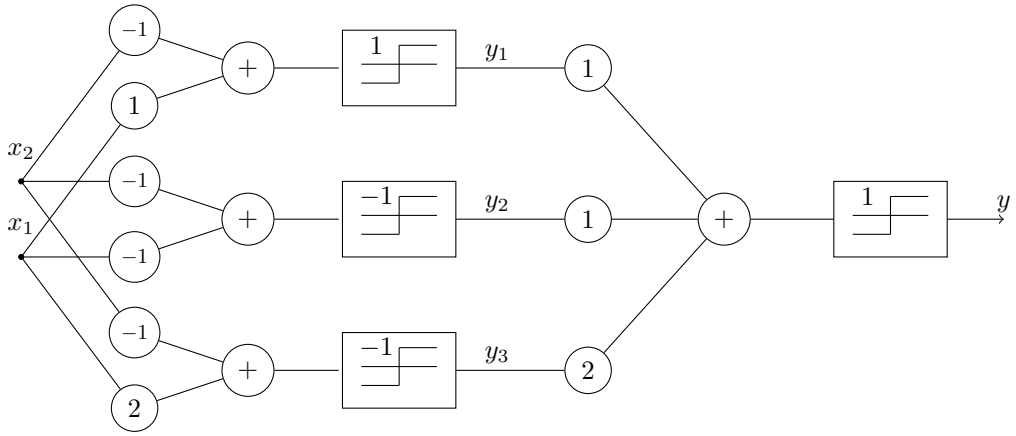

FIGURE 3. Combining linear classifiers

output of the first three. Sketch the region of $\mathbb{R}^2$ *for which the final output* y *is equal to* 1.

Exercise 4 (Bishop)**.** *Just as we used it in regression, we can use the least squares criterion for classification. When considering a classification problem with* K *classes, the common approach is to consider* K *dimensional binary vectors* $t_j = (0, 1, 0, \ldots, 0)$ *which encode the class of each prototype* $\boldsymbol{x}_j$*.* $\boldsymbol{t}_j$ *is thus a binary vector whose* k^{th}

$\boldsymbol{x}$	y
$(0,0)$	0
$(1,0)$	1
$(0,1)$	1
$(1,1)$	0

TABLE 1. XOR dataset

entry is 1 *if the prototype* $\boldsymbol{x}_j$ *belongs to class* $\mathcal{C}_k$*. We want to learn a prediction model*

$$y_j(\boldsymbol{x}) = \boldsymbol{W}^T \boldsymbol{x}_j \tag{3}$$

Where $\boldsymbol{W}$ *(i.e its columns) encode* K *separating planes. If we group the prototypes in a matrix* $\boldsymbol{X}$ *whose* i^{th} *row is given by* $[1, \boldsymbol{x}_i]$*, and put the binary vectors* $\boldsymbol{t}_i$ *as the rows of a matrix* $\boldsymbol{T}$*, the weights in the model* (3) *have to satisfy*

$$\min_{\boldsymbol{W}} \frac{1}{2} \mathrm{Tr}\left\{(\boldsymbol{X}\boldsymbol{W} - \boldsymbol{T})^T (\boldsymbol{X}\boldsymbol{W} - \boldsymbol{T})\right\} \tag{4}$$

Solving the problem by setting the derivatives to zero as in regression, we have

$$\boldsymbol{W} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{T} \tag{5}$$

and hence

$$\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{W}^T \boldsymbol{x} = \boldsymbol{T}^T \left[(\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T\right]^T \boldsymbol{x} \tag{6}$$

Show that if every target vector $\boldsymbol{t}_n$ *satisfies a linear constraint of the form* $\boldsymbol{a}^T \boldsymbol{t}_n + b = 0$ *then so does the prediction model,*

$$\boldsymbol{a}^T \boldsymbol{y}(\boldsymbol{x}) + b = 0 \tag{7}$$

Exercise 5 (source: Bishop)**.** *Logistic regression is a slightly improved version of linear regression in which we define classes through probabilities. We introduce* $K - 1$ *models for each of the classes* $\mathcal{C}_k$*,* $k = 1, \dots K_1$

$$P(\mathcal{C}_k|\boldsymbol{x}) = \frac{\exp(\beta_{0,k} + \boldsymbol{\beta}_{1,k}^T \boldsymbol{x})}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{0,\ell} + \boldsymbol{\beta}_{1,\ell}^T \boldsymbol{x})}, \quad k = 1, \dots, K-1, \tag{8}$$

$$P(\mathcal{C}_K|\boldsymbol{x}) = \frac{1}{1 + \sum_{\ell=1}^{K-1} \exp(\beta_{0,\ell} + \boldsymbol{\beta}_{1,\ell}^T \boldsymbol{x})} \tag{9}$$

The last probability is defined from the $K - 1$ *previous probabilities as* $P(\mathcal{C}_K|\boldsymbol{x}) = 1 - \sum_{\ell=1}^{K-1} P(\mathcal{C}_\ell|\boldsymbol{x})$*. Despite its elegance, unlike the more simple linear regression model, logistic regression does not make it easy to derive a closed form solution. The way one typically proceeds is by viewing the samples as independent and writing down the log likelihood function.*

In the two classes case, for instance, we consider a binary label $y \in \{0, 1\}$ *which takes the value* 0 *for class* $\mathcal{C}_0$ *and* 1 *for class* $\mathcal{C}_1$*. Given this, the probability to*

observe the class distribution of a particular dataset[1] *reads as*

$$P = \prod_{i=1}^{N} P(y_i|\boldsymbol{x}_i;\boldsymbol{\beta}) = \prod_{i=1}^{N} P(y_i = 1|\boldsymbol{x}_i;\boldsymbol{\beta})^{y_i} P(y_i = 0|\boldsymbol{x}_i;\boldsymbol{\beta})^{1-y_i} \tag{10}$$

The log likelihood then reads as

$$\log(P) = \sum_{i=1}^{N} \{y_i \log(P(y_i = 1|\boldsymbol{x}_i;\beta)) + (1 - y_i) \log(P(y_i = 0|\boldsymbol{x}_i;\beta))\} \tag{11}$$

- *Show that the derivative of the logistic sigmoid function,*

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{12}$$

  is given by $\frac{d\sigma}{dx} = \sigma(1 - \sigma)$
- *Now show that the derivative of the log likelihood* (11) *is given by*

$$\nabla\ell(\boldsymbol{\beta}) = \frac{\partial\ell(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \sum_{i=1}^{N}(P(y_i = 1|\boldsymbol{x}_i;\beta) - y_i)\boldsymbol{x}_i \tag{13}$$

- *Finally, show that for a linearly separable dataset, the maximum likelihood solution for the logistic regression model is obtained by finding a vector* $\boldsymbol{w}$ *whose decision boundary* $\boldsymbol{w}^T\phi(\boldsymbol{x})$ *separates the classes and then taking the magnitude of* $\boldsymbol{w}$ *to infinity.*

Exercise 6 (LDA*)**.** *An alternative to logistic regression, Linear Discriminant Analysis (LDA) starts from the more general model*

$$P(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_{\ell=1}^{K} p(\boldsymbol{x}|\mathcal{C}_\ell)p(\mathcal{C}_\ell)} = \frac{f_k(\boldsymbol{x})\pi_k}{\sum_{\ell=1}^{K} f_\ell(\boldsymbol{x})\pi_\ell(\boldsymbol{x})} \tag{14}$$

We then choose the density of each class to be given by a multi (i.e D-)dimensional Gaussian

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp(-\frac{1}{2}((\boldsymbol{x} - \boldsymbol{\mu}_k)\boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k))) \tag{15}$$

- *As in logistic regression, the log odd ratios (i.e the logs of the ratios of the class posterior probabilities),*

$$\log\left(\frac{P(\mathcal{C}_k|\boldsymbol{x})}{P(\mathcal{C}_\ell|\boldsymbol{x})}\right) \tag{16}$$

  define hyperplanes. Show that those logg odd ratio are indeed linear in $\boldsymbol{x}$ *for the Gaussian model* (15)*.*
- *We consider two classes* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. Following exercise 5, write down the expression for the likelihood and then the log-likelihood for this model.*
- *Set the derivatives with respect to the class probabilities* π_ℓ *to zero and solve. How are these probabilities set by the model ?*
- *Now set the derivatives with respect to* $\boldsymbol{\mu}$ *to* 0 *and get the expressions for* $\boldsymbol{\mu}_1$ *and* $\boldsymbol{\mu}_2$
- *Finally, derive the expression of the shared covariance matrix* $\boldsymbol{\Sigma}$*.*

[1]Note that the likelihood function here is a little bit different from what we were used to. It can in fact be understood as a traditional likelihood, i.e $\prod_{i=1}^{N} p(x_i, y_i|\beta)$ for which one would assume uniform distribution of the x_i

- *After learning the LDA model (we still consider two classes), a new point* $\boldsymbol{z}$ *is classified by comparing the probabilities* $P(\mathcal{C}_1|\boldsymbol{x})$ *and* $P(\mathcal{C}_2|\boldsymbol{x})$ *and labelling* $\boldsymbol{x}$ *as belonging to the class which gives the highest posterior probability. Using the expression for the log odds ratios, show that in the two classes framework, this is equivalent to the rule*

$$\begin{cases} \boldsymbol{z} \in \mathcal{C}_2 & \text{if} \quad \boldsymbol{z}^T \boldsymbol{\Sigma}^{-1}(\hat{\boldsymbol{\mu}}_2 - \hat{\boldsymbol{\mu}}_1) > \frac{1}{2}(\hat{\boldsymbol{\mu}}_2 + \hat{\boldsymbol{\mu}}_1)^T \hat{\boldsymbol{\Sigma}}^{-1}(\hat{\boldsymbol{\mu}}_2 - \hat{\boldsymbol{\mu}}_1) - \log(N_2/N_1) \\ \boldsymbol{z} \in \mathcal{C}_1 & \text{otherwise} \end{cases} \tag{17}$$

Exercise 7 (Naive Bayes*, Murphy, Jaakkola)**.** *Given pairs* $(\boldsymbol{x}_i, y_i)$*, the naive Bayes classifier learns a model for the class conditional probability density by assuming that the features are independent*

$$p(\boldsymbol{x}|y = \mathcal{C}_\ell, \boldsymbol{\theta}) = \prod_{j=1}^{D} p(x_j|y = \mathcal{C}_\ell, \boldsymbol{\theta}_{j\ell}) \tag{18}$$

Where $\boldsymbol{\theta}_{j\ell}$ *indicates the parameter of the distribution encoding the probability of observing feature* j *from class* $\mathcal{C}_\ell$*. In this exercise, we want to develop a model for the number of occurence of a given word in a document. If we let* $\theta_{j\ell}$ *to denote the probability of observing word* j *in document* ℓ*, and if we assume that the probability of observing each word is independent from the probability of observing the others, then we can write the probability of observing a document* i *(belonging to class* $\mathcal{C}_\ell$*) as*

$$p(\boldsymbol{x}_i|\mathcal{C}_\ell, \boldsymbol{\theta}) = \prod_{w=1}^{W} \theta_{\ell w}^{x_{iw}} (1 - \theta_{\ell w})^{1 - x_{iw}} \tag{19}$$

$\theta_{\ell w}$ *is the estimate of the probability that word* w *occurs in a document of class* $\mathcal{C}_\ell$*.* $x_{iw} = 1$ *if word* w *occurs in document* i *and* 0 *otherwise.*

- *Is classifier* (18) *generative or discriminative?*
- *Write down the expression for the log likelihood.*
- *This log likelihood can read compactly as*

$$\log(p(\boldsymbol{x}_i|C = c, \theta)) = \phi(\boldsymbol{x})^T \boldsymbol{\beta}_c \tag{20}$$

  where β_c *encodes the parameters of the model. Give the expressions for* $\boldsymbol{\beta}_c$ *and* $\phi(\boldsymbol{x})$[2]
- *We now assume that there are two classes* 1 *and* 2*. If we further assume* $p(C = 1) = p(C = 2) = 1/2$*, write down the expression for the log-odd ratios,*

$$\log_2 \frac{p(C = 1|\boldsymbol{x}_i)}{p(C = 2|\boldsymbol{x}_i)} \tag{21}$$

  in terms of the features $\phi(\boldsymbol{x}_i)$ *and vectors of parameters* $\boldsymbol{\beta}_1$ *and* $\boldsymbol{\beta}_2$*.*
- *Words that occur with a similar frequency in documents from two different classes are not very meaningful. Using the log-odd ratio that you derived above, derive the conditions on* $\boldsymbol{\beta}_{1w}$ *and* $\boldsymbol{\beta}_{2w}$ *(or* θ_{1w}/θ_{2w}*) under which the*

[2]From this expression, you see that the classifier can be considered linear (i.e the log of the class conditional density, which will give us an indication on the class of each point is linear in the parameters β)

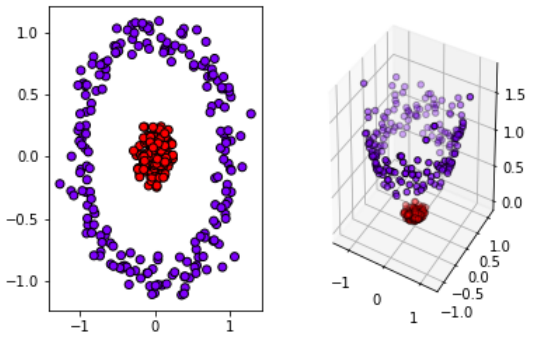

FIGURE 4. Concentric dataset 1

presence or absence of a word w will have no effect on the classification of a document.

Exercise 8 (back to XOR)**.** *Consider the dataset of Fig. 4.*

- *Is this dataset linearly separable?*
- *In order to learn a classification model for such dataset, one approach is to bring the data into a higher dimensional space (rightmost figure) and then learn a separating plane. Here the transformation is given by* $(x, y) \mapsto (x, y, z = x^2 + y^2)$*. We will do the same with the XOR dataset of exercise 1. For this dataset, we will use a general model, known as Radial Basis Function (RBF) Network. An RBF network is similar to a linear regression/classification model except that we now replace the original data by data after transformation,*

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} \beta_i \phi_i(\|\boldsymbol{x} - \boldsymbol{\mu}_i\|) \tag{22}$$

A popular choice for the functions ϕ_i *is to use Gaussian kernels,* $\phi_i = \exp(-\frac{\|\boldsymbol{x}-\mu_i\|^2}{\sigma_i^2})$*. We decide to use two kernels. Choose the parameters* μ_1, μ_0*,* σ_1, σ_0 *and* β_1, β_0 *to learn a model that separates the* $y = 1$ *points from the* $y = 0$ *points in the XOR dataset (no differentiation needed here just basic thinking). How do you set the boundary ?*