Introduction to Machine Learning **Fall Semester**

Homework 4: March 6, 2024

Due: March 20, 2024

Theory Questions

1. **(25 points) SVM with multiple classes.** One limitation of the standard SVM is that it can only handle binary classification. Here is one extension to handle multiple classes. Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and now let $y_1, \ldots, y_n \in [K]$, where $[K] = \{1, 2, \ldots, K\}$. We will find a separate classifier $\mathbf{w}_j$ for each one of the classes $j \in [K]$, and we will focus on the case of no bias ($b = 0$). Define the following loss function (known as the *multiclass hinge-loss*):

$$\ell(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{x}_i, y_i) = \max_{j \in [K]} (\mathbf{w}_j \cdot \mathbf{x}_i - \mathbf{w}_{y_i} \cdot \mathbf{x}_i + \mathbb{1}(j \neq y_i)),$$

   where $\mathbb{1}(\cdot)$ denotes the indicator function. Define the following multiclass SVM problem:

$$f(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{x}_i, y_i)$$

   After learning all the $\mathbf{w}_j$, $j \in [K]$, classification of a new point $\mathbf{x}$ is done by $\arg\max_{j \in [K]} \mathbf{w}_j \cdot \mathbf{x}$. The rationale of the loss function is that we want the "score" of the true label, $\mathbf{w}_{y_i} \cdot \mathbf{x}_i$, to be larger by at least 1 than the "score" of each other label, $\mathbf{w}_j \cdot \mathbf{x}_i$. Therefore, we pay a loss if $\mathbf{w}_{y_i} \cdot \mathbf{x}_i - \mathbf{w}_j \cdot \mathbf{x}_i \leq 1$, for $j \neq y_i$.

   Consider the case where the data is linearly separable. Namely, there exists $\mathbf{w}_1^*, ..., \mathbf{w}_K^*$ such that $y_i = \text{argmax}_y \, \mathbf{w}_y^* \cdot \mathbf{x}_i$ for all i. Show that any minimizer of $f(\mathbf{w}_1, ..., \mathbf{w}_K)$ will have zero classification error.

2. **(10 points) Suboptimality of ID3.** Solve exercise 2 in chapter 18 in the course book: Understanding Machine Learning: From Theory to Algorithms.

3. **(25 points) Step-size Perceptron.** Consider the modification of Perceptron algorithm with the following update rule:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \eta_t y_t \mathbf{x}_t$$

   whenever $\hat{y}_t \neq y_t$ ($\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$ otherwise). Assume that data is separable with margin $\gamma > 0$ and that $\|\mathbf{x}_t\| = 1$ for all t. For simplicity assume that the algorithm makes M mistakes at the first M rounds, after which it makes no mistakes. For $\eta_t = \frac{1}{\sqrt{t}}$, show that the number of mistakes step-size Perceptron makes is at most $\frac{4}{\gamma^2} \log(\frac{1}{\gamma})$. (Hint: use the fact that if $x \leq a \log(x)$ then $x \leq 2a \log(a)$). It's okay if you obtain a bound with slightly different constants, but the asymptotic dependence on γ should be tight.

4. **(40 Points) Kernel PCA.** In the PCA algorithm, we are given a sample $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$. We would like to extend it to Kernel PCA, as follows. We are given a mapping function $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$. We would like to perform PCA on the mapped points, $\phi(\mathbf{x}_i)$. For the Kernel PCA algorithm, we will use the matrix $\bar{K}$ defined as

$$\bar{K}_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j),$$

where as usual $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$. The algorithm should only use K and not $\phi(\mathbf{x})$. Throughout this question you can assume that $\bar{K}$ is invertible.

(a) Recall that in PCA, we require the sample to be mean-centered. Namely: $\frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i = 0$. In regular PCA, we can achieve this by subtracting the mean. For kernel PCA, we would like to achieve this by using the kernel function alone. Denote the following, mean-centered version of $\phi(\mathbf{x})$:

$$\mathbf{v}_i = \phi(\mathbf{x}_i) - \frac{1}{n}\sum_{t=1}^{n} \phi(\mathbf{x}_t).$$

We would like to calculate the kernel matrix for the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$. Namely, define the matrix $\bar{K}' \in \mathbb{R}^{m \times m}$:

$$\bar{K}'_{i,j} = \langle \mathbf{v}_i, \mathbf{v}_j \rangle.$$

Show how $\bar{K}'$ can be calculated using only the original kernel matrix $\bar{K}$.

(b) For the rest of the question, you may assume that $\sum_{i=1}^{n} \phi(\mathbf{x}_i) = 0$. We would like to apply PCA to the vectors $\phi(\mathbf{x}_i)$. Denote by $\mathbf{u}_1, \ldots, \mathbf{u}_k$ the first k principal components in $\mathbb{R}^{d'}$, corresponding to the sample $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)$. Assuming $k \leq n$, show that $\mathbf{u}_j$ (for $j = 1, \ldots, k$) is a linear combination of $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)$. That is, show that $\mathbf{u}_j = \sum_{i=1}^{n} \alpha_{j,i} \phi(\mathbf{x}_i)$. (Hint: use the fact that $\mathbf{w}\mathbf{w}^T\mathbf{v} = (\mathbf{w}^T\mathbf{v})\mathbf{w}$ for any vectors $\mathbf{v}$ and $\mathbf{w}$).

(c) Use the above to show that the coefficients $\alpha_{j,i}$ can be calculated efficiently (without dependence in d')? (Hint: show that $\alpha_j = \frac{1}{\lambda_j}\Phi\mathbf{u}_j$ where Φ is the matrix whose rows are the $\phi(\mathbf{x}_i)$'s and λ_j is the eigenvalue of $\Phi^T\Phi$ corresponding to the principal component $\mathbf{u}_j$. Conclude that each vector of coefficients α_j is a an eigenvector of $\bar{K}$)

(d) Since d' can be very large (perhaps even infinite), we will not look for the principal components themselves, but instead will be satisfied with the ability to perform a dot product of each principal component with the mapping $\phi(\mathbf{x})$ of a new point, $\mathbf{x}$. More explicitly, let $\mathbf{x}$ be a new point. Show how we can calculate

$$\langle \mathbf{u}_j, \phi(\mathbf{x}) \rangle$$

for $j = 1, \ldots, k$. What is the complexity of the solution?