CS 412: Spring'21
Introduction To Data Mining

Take-Home Final

(Due Saturday, May 8, 10:00 am)

General Instructions

- You will have to answer the questions yourself, you cannot consult with other students in class. It is an open book exam, so you can use the textbook and the material shared in class, e.g., slides, lectures, etc.
- The take-home final will be due at 10 am, Saturday, May 8. We will be using gradescope for the submissions. Please submit your answers via gradescope, and please contact the TAs if you are having technical difficulties in submitting the assignment. We will NOT accept late submissions.
- Your answers should be typeset and submitted as a pdf. You cannot submit a hand-written and scanned version of your answers.
- You DO NOT have to submit code for any of the questions.
- For the questions, you will not get full credit if you only give out a final result. Please show the necessary details, calculation steps, and explanations as appropriate.
- If you have clarification questions, you can use slack or campuswire. However, since the midterm needs to be submitted within 24 hours, please try to do your best in answering the questions based on your own understanding, in case responses are delayed.

1. (25 points) Consider the following dataset for 2-class classification (Figure 1), where the blue points belong to one class and the orange points belong to another class. Each data point has two features $\mathbf{x} = (x_1, x_2)$. We will consider learning support vector machine (SVM) classifiers on the dataset.

Figure 1: 2-class classification dataset.

   (a) (7 points) Can we train a hard margin linear SVM on the dataset? Clearly explain your answer.

   (b) (8 points) Can we train a soft margin linear SVM on the dataset? If yes, briefly describe how you will do it. If no, briefly explain why not.

   (c) (10 points) Professor Quadratic Kernel claims that mapping each feature vector $\mathbf{x}^i = (x_1^i, x_2^i)$ to a 6-dimensional space given by

$$\phi(\mathbf{x}^i) = [1 \quad x_1^i \quad x_2^i \quad x_1^i x_2^i \quad (x_1^i)^2 \quad (x_2^i)^2]^T$$

   and training a linear SVM in that mapped space would give a highly accurate predictor. Do you agree with Professor Kernel's claim? Clearly explain your answer.

2. (25 points) We consider comparing the performance of two classification algorithms A_1 and B_1 based on k-fold cross-validation. The comparison will be based on a t-test to assess statistical significance with significance level $\alpha = 5\%$.[1]

   (a) (5 points) We will assess the results for $k = 10$-fold cross-validation. What should be the degrees of freedom for the test? Briefly explain your answer.

   (b) (10 points) The accuracies for $k = 10$-fold cross-validation from algorithms A_1 and B_1 are given in Table 1.

   Is the performance of one of the algorithms significantly different than the other based a t-test at significance level $\alpha = 5\%$? Clearly explain your answer by showing details of (a) the computation of the t-statistic, and (b) the computation of the p-value.

[1] The relevant material for testing statistical significance is discussed in Chapter 7 of the text book. We will assume that the conditions needed for the validity of the test are satisfied.

	1	2	3	4	5	6	7	8	9	10
A_1	0.908	0.962	0.878	0.956	0.939	0.955	0.944	0.933	0.881	0.949
B_1	0.449	0.585	0.381	0.433	0.475	0.430	0.520	0.590	0.565	0.443

Table 1: Accuracies on 10-folds for Algorithms A_1 and B_1.

Given the t-statistic `t_stat` and degrees of freedom `df`, you should be able to compute the p-value using the following:[2]

```
from scipy.stats import t
p_val = (1-t.cdf(abs(t_stat), df)) * 2
```

(c) (10 points) Suppose we have a better version of algorithm B_1 called B_2. The accuracies for $k = 10$-fold cross-validation from algorithms A_1 and B_2 are given in Table 2.

	1	2	3	4	5	6	7	8	9	10
A_1	0.908	0.962	0.878	0.956	0.939	0.955	0.944	0.933	0.881	0.949
B_2	0.968	1.000	0.950	0.994	0.989	0.989	1.000	0.994	0.966	0.966

Table 2: Accuracies on 10-folds for Algorithms A_1 and B_2.

Is one of the algorithms significantly better than the other based a t-test at significance level $\alpha = 5\%$? Clearly explain your answer by showing details of (a) computation of the t-statistic, and (b) computation of the p-value.

3. (25 points) Let $\mathcal{Z} = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^n, y^n)\}, \mathbf{x}^i \in \mathbb{R}^d, y^i \in \{0, 1\}, i = 1, \ldots, n$ be a set of n training samples. The input $\mathbf{x}^i, i = 1, \ldots, n$ are d-dimensional features, and x^i_j denotes the j-th feature of the i-th data point $\mathbf{x}^i$. The output $y^i \in \{0, 1\}, i = 1, \ldots, n$, are the class labels. We consider training a single layer perceptron where for any input $\mathbf{x}^i$, the output is given by

$$\hat{y}^i = g(a^i) = g(\mathbf{w}^T \mathbf{x}^i) = g\left(\sum_{j=1}^{d} w_j x^i_j\right) \ ,$$

where $g(a) = \max(a, 0)$, i.e., the ReLU transfer function, and $a^i = \mathbf{w}^T \mathbf{x}^i$ is the input activation. Note that the parameters $\mathbf{w} = [w_1 \cdots w_d]^T$ are the unknown parameters of the model. Consider a learning algorithm which focuses on minimizing squared loss between the true and predicted outputs:

$$L(\mathbf{w}|\mathcal{Z}) = \frac{1}{2}\sum_{i=1}^{n}(y^i - \hat{y}^i)^2 = \frac{1}{2}\sum_{i=1}^{n}(y^i - g(\mathbf{w}^T \mathbf{x}^i))^2 \ .$$

[2] Alternatively, you can look a table for p-values for t-statistic, similar to how you had done it for the χ^2-statistic earlier in the semester.

(a) (15 points) The stochastic gradient descent (SGD) algorithm updates the parameters based on a random chosen point $(\mathbf{x}^i, y^i)$ in each step. Show that the SGD update for parameter w_j with step size η is of the form

$$w_j^{\text{new}} = w_j + \eta g'(a^i)(y^i - \hat{y}^i)x_j^i \tag{1}$$

where $a^i = \mathbf{w}^T\mathbf{x}^i$, and the gradient of the ReLU function is

$$g'(a^i) = \begin{cases} 1\ , & \text{if } a^i \geq 0\ , \\ 0\ , & \text{otherwise}\ . \end{cases} \tag{2}$$

(b) (10 points) Instead of using the ReLU transfer function $g(a) = \max(a, 0)$, consider using the linear transfer function $g(a) = a$. How will you modify (1) and/or (2) above to get the SGD algorithm for the linear transfer function $g(a) = a$. Clearly explain your answer.

4. (25 points) This question considers partitioning based clustering methods. In particular, parts (a) and (b) consider the k-medoids algorithm, and (c) considers the k-means and k-medians algorithms.

(a) (10 points) Clearly describe the k-medoids clustering algorithm using pseudocode and a brief description of each of the steps.

(b) (5 points) What is the computational complexity of the k-medoids clustering algorithm? Briefly justify your answer.

(c) (10 points) What is the motivation behind using the k-medians algorithm instead of the k-means algorithm in certain situations? Is the k-medians algorithm more computationally demanding than k-means? Briefly explain your answer.