EE 559 **Homework 2** Posted: Sat.. 1/28/2023
Jenkins Due: Sat., 2/4/2023, 11:59 AM (noon)

Note: for a dataset, **classification accuracy** is defined as number of correctly classified data points divided by the total number of data points.

1. For a 2-class nearest-means classifier (NMC) based on *D* features, and for given mean vectors $\mu_1,\ \mu_2$:
   (a) Give an algebraic expression for the decision boundary and for the decision rule. Simplify as much as possible. Is the classifier linear?
   (b) If the classifier is linear, starting from the general expression for a 2-class discriminant function *g(x)* for a linear classifier, find an expression for the weights for the NMC in terms of the mean vectors $\mu_1,\ \mu_2$.
   (c) If the classifier is not linear, then write an expression for a nonlinear function *g(x)* with weight for coefficients of each term. Then find expressions for the weights of the NMC in terms of $\mu_1,\ \mu_2$.

   For parts (d)-(f), consider a *C*-class NMC based on *D* features, with *C* > 2.

   (d) Consider the following decision rule:
   $$\mathrm{x} \in \Gamma_k \ \text{ iff } \ k = \mathrm{argmax_m}\{g_m(x)\}$$
   Can you find expressions for the $g_m(x),\ i = 1,2,\cdots,C$, such that this is the decision rule for a NMC? If so, give your expression for $g_m(x)$, and simplify it as much as possible.
   Hint: when comparing $g_m(x)$ only to each other (e.g., $g_i(x)$ to $g_j(x)$), any additive term that doesn't depend on *m*, and that is common to $g_m(x)\ \forall m$, can be dropped from all $g_m(x)$.
   (e) Is $g_m(x)$ linear? Justify your answer. If yes, give expressions for the weights for the NMC in terms of the mean vectors μ_k.
   (f) Is multiclass NMC an example of the MVM multiclass method? Justify your answer.

2. Code up a *C*-class nearest-means classifier (NMC), for *C* classes and *D* features. Homework 1 ground rules on libraries you can use apply here also.

   Tip: for plots in this problem, you may use plotDecBoundaries_2.py.

   (a) Run it on the given dataset. There are C = 5 classes and D = 7 features. Report the classification accuracy on the training set and test set.
   (b) For visualization, run it again using only the following 2 features: X1 and X2. Plot in 2D feature space: the training data, decision boundaries, and decision regions for all the classes. Report the classification accuracy on the training set and test set, using only the 2 features. As a check, do the decision boundaries look consistent with the class means, given it's a NMC?
   (c) Repeat (b) using only the following features: X3 and X4.

(d) Repeat (b) using only the following features: X3 and X7.

(e) Of (b), (c), (d): which gives the best training accuracy? the best test accuracy? Does the use of all 7 features perform better than the pairs tried in (b), (c), (d)?

(f) In the plots of (b), (c), (d): Do you see any indeterminate regions? Are all the decision regions convex?

=> Feel free to optionally explore other pairs of features to see the variety of decision boundaries, regions, and results obtainable. (No need to report on results.)

3. This problem uses the notation we used in Lecture 5, and m, m_0, and m_1 are positive integers. For the following computational complexity:

$$p(m) = 10m - 50$$

(a) Is $p(m) = O(m)$?

If yes, prove your answer by letting $a = 1$, and solve for what m_0 we have $m \geq p(m) \quad \forall m \geq m_0$. If you need a larger *a*, then state what value of *a* will work. Find the smallest positive integer m_0 for the value of *a* you used.

If no, justify why not.

(b) Is $p(m) = \Omega(m)$?

If yes, prove your answer by letting $b = 1$, and solve for what m_1 we have $m \leq p(m) \quad \forall m \geq m_1$. If you need a smaller *b*, then state what value of *b* will work. Find the smallest positive integer m_1 for the value of *b* you used.

If no, justify why not.

(c) Is $p(m) = \Theta(m)$?

Justify your answer.

4. This problem also uses the notation of Lecture 5, and here also all *m* are positive integers.

(a) Suppose we have a function $p(m)$ that can be expressed as:

$$p(m) = p_1(m) + p_2(m) + p_3(m)$$

and we have:

$$p_k(m) = O(q_k(m)), \quad k = 1,2,3 \qquad \text{(i)}$$

Prove that:

$$p(m) = O(q_1(m) + q_2(m) + q_3(m)) \qquad \text{(ii)}$$

Hints:

(i) If you find the problem statement unclear or confusing, try looking at the example in the appendix below.

(ii) You can start the proof by applying the definition of big-O to (i) to get 3 sets of inequalities, then summing them. Can you get your new inequality to look like the definition of big-O?

(b) Is a similar statement to (a) true for $\Omega(..)$? (That is, if you replace each $O(..)$ in part (a) with $\Omega(..)$, would the last equation be true?) Justify your answer.

5. Consider the following computational complexity, in which *m* is a positive integer:

$$\mathrm{p}(m) = m^2 \log_2 \mathrm{m} + 10\left(\frac{2^m}{\log_2 m}\right) + 0.1\left(2^{(m-5)}\right)$$

(a) Find the asymptotic upper bound for *p(m)*, in simplest form (no unnecessary constants) but no looser than necessary.

Hint: $p(m)$ is the sum of 3 terms. You can use the result of Problem 4.

(b) Find the asymptotic lower bound for *p(m)*, in simplest form (no unnecessary constants) but no looser than necessary.

Appendix - Example (relates to Problem 4)

Suppose we want to find and prove the (tightest) asymptotic upper bound of $p(m)$, with:

$$p(m) = 3m^3 + 100m^2 \log_2 m + 0.1\left(2^m\right)$$

Applying the definition directly to $p(m)$ (especially to prove your bound, including finding m_0) might be difficult. Instead, you could use the result of Problem 4a, to apply the big-O bound to each term independently:

$$3m^3 = O\left(m^3\right)$$

$$100m^2 \log_2 m = O\left(m^2 \log m\right)$$

$$0.1\left(2^m\right) = O\left(2^m\right)$$

Then using Problem 4a equation (ii), we can conclude:

$$\begin{aligned} 3m^3 + 100m^2 \log_2 m + 0.1\left(2^m\right) &= O\left(m^3 + m^2 \log m + 2^m\right) \\ &= O\left(2^m\right) \end{aligned}$$