EE769: Introduction to Machine Learning; End-Semester Examination Solutions; IIT Bombay, 2018.05.04

1. Answer the following short and easy questions:
   a. What is the underlying dimension of a helix (like the thread of a screw or a spring) as a manifold in 3-D space? [1]

      ONE. If you stretch a helix, you can form a 1-d thread.

   b. What is the number of parameters of an n-layer neural network with L_0 inputs, and $L_1, L_2, \dots, L_n$ neurons in the respective subsequent layers? Each neuron has a bias. [2]

      Solution: $\sum_{i=1}^{n} L_{i-1}L_i + \sum_{i=1}^{n} L_i$

   c. Is $\text{sign}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n + b)$ a good activation function? Give reason. [2]

      No. The function is not continuous and has zero gradient almost everywhere [1 for one reason, 1.5 for both reasons]. The sub-gradient doesn't exist at 0 [2 for this reason].

   d. Is $\sin(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n + b)$ a good activation function? Give reason. [2]

      No. The function is not monotonic. For high weights, the function behaves unpredictably. It has infinite VC dimension. [2 marks for any two of these reasons. 0.5 for "No" without (correct) reason. 1.5 for one reason]

   e. In DBSCAN, if there are too many clusters, then what all can you adjust and how? [2]

      Lower minPts or increase ε. One mark for making a silly mistake in interpreting the question.

   f. In DBSCAN, to discover the clusters in the figure to the right, suggest a good hyper-parameter setting. [2]

      MinPts has to be at least 3, and at most 18 or so, so that there are no cross-links, and only links within each arm of the spiral. And ε has to be based on inter-point distance which is around 0.5 or 1 but definitely less than 3 or 4.

2. Examine the factor graph given in the figure to the right.
   a. What is the message from f_1 to A in terms of other messages? [2]

      Solution: $\sum_B \sum_C f_1(A,B,C)\mu_{B\to f_1}\mu_{C\to f_1}$

   b. What is the Markov blanket of B? [2]

      {A,C,D,E}

   c. Write an expression for $p(E|B{=}1)$ assuming binary variables. [2]

      Solution: $\frac{f_3(B=1,E)}{f_3(B=0,E)+f_3(B=1,E)}$. ***One mark for missing the denominator.***

3. Design a neural network for binary classification such that the area inside the triangle given by the following points $\{(-5,0),(5,0),(0,10)\}$ in $\mathbb{R}^2$ is class 0, and outside is class 1. Assume that it has a single hidden layer with step function activations, and a single output neuron with step function activation also. [4]

   There will be three hidden neurons with weights to separate the three half-spaces and the final layer will have one neuron whose binary output can directly give the class. The weights of the hidden layer will be {-α/5, α/10,bias=-α},{β/5, β/10,bias=-β},{0,γ,bias=0}. The weights of the output layer will be {sign(α),sign(β),sign(γ),bias=-(0,1)}, {α,β,γ}>0. Or some other such combination. Two marks for understanding the NN structure (two inputs, three hidden neurons, one (or two) output neurons.

4. Find the derivative of the output of the computation graph shown in the figure with respect to y using backpropagation, assuming $x = 1, y = 2, z = 0$, and $*$ is multiply. Hint: for multiplication $a * b$, derivative with respect to a gets multiplied by b, that is, it is $a'b$. If this is too hard to solve then replace $*$ with a – (minus) and solve the question with one mark penalty. [3]

   Given that f=(x+y)*max(y,z). Let a=x+y, b=max(y,z). Therefore by chain rule (or back propagation), f=a*b. So, ∂f/∂y = ∂f/∂(a*b) ∂(a*b)/∂y = 1 (a ∂b/∂y + b ∂a/∂y) = 1 (a $1_{y>z}$ + b 1). Substituting a=3, b=2, $1_{y>z}$=1, we get ∂f/∂y = 1 (3*1+2*1) = 5.

5. Cross-entropy loss is given by the expression $-\{y \log f(\boldsymbol{x}) + (1-y)\log\big(1-f(\boldsymbol{x})\big)\}$ where desired output $y \in \{0,1\}$ and estimated probability that $y = 1$ is $f(x) \in (0,1)$. Generalize this expression to 3-classes. Show that it has the lowest value for correct classification. What is its highest value and for what kind of output? Hint: you cannot use a scalar valued function f and nor can the desired output be a scalar y. You will need to define the

desired output as a one-hot-bit vector, the estimated output also as a vector representing a probability mass function. [3]

Solution: $-\sum_{d=1}^{3} y_d \log\big(f_d(x)\big)$, ***where y_d is one-hot bit encoding of the desired output, and $f_d(x)$ is the vector valued probability mass function. This assumes minimum value 0, if the prediction is correct. Max value is +∞ when $\log(f_d(x))$ has the least value, which when the correct class has zero predicted probability.***

6. (a) Assume that for a contiguous shape in $\mathbb{R}^2$ you are given an indicator function $inside(x, y)$ which when called in a program gives the value 1 if the point (x, y) lies inside the shape, and 0 if they lie outside the shape. You are asked to compute the approximate area of the shape using Metropolis algorithm. Write the pseudo code for computing the area of the shape using a proposal distribution around the current point that is a bivariate Gaussian with a fixed covariance matrix. Assume that the shape is a rectangle with height *h* and width *w.* [3]

t ← 0
Randomly initialize (x_0,y_0)
While not inside(x_0,y_0)
    Randomly initialize (x_0,y_0)
Reject_count=0
For t=1 to T iterations [1 mark for looping]
    t ← t+1
    Sample q((x_t,y_t)\|(x_{t-1},y_{t-1}) # where q is the bi-variate Gaussian with mean (x_{t-1},y_{t-1}) and Σ=[σ,0;0,σ]
    # [1 mark for understanding that the mean of the Gaussian is at the previous sample]
    If not inside(x_t,y_t)
        (x_t,y_t) ← (x_{t-1},y_{t-1}) #[1 mark for understanding that probability of rejection is binary, not continuous]
        Reject_count++
Area = some_constant * σ^2 * (1 – Reject_count / T)

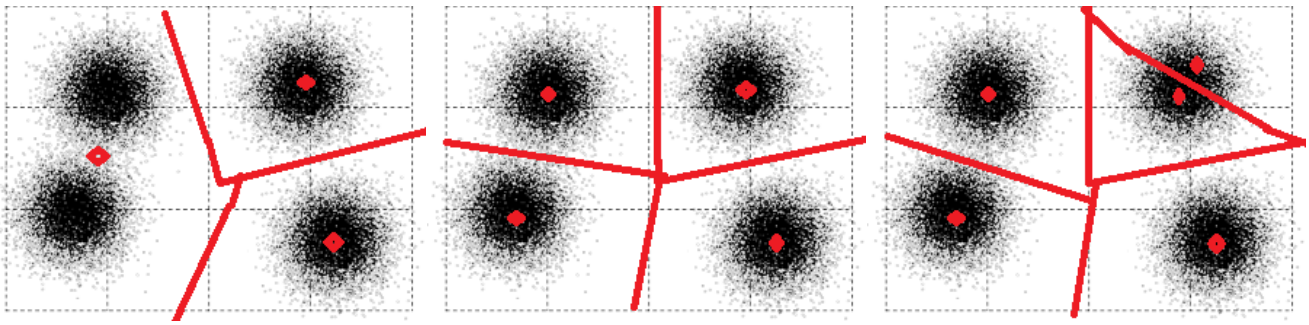
(b) For the previous part, how you will choose the hyper-parameters of the proposal distribution. Would you need any prior information about the shape? [2] ***Some idea of how big the shape is and how narrow it generally is will be necessary. Otherwise, if σ is too large, then most of the samples will get rejected, and if it is too small, we may not have enough iterations to cover the shape.***

7. Convert a ridge regression objective function $J(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n - y_n\}^2 + \frac{\lambda}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$ (where $\boldsymbol{w}$ is weight vector, $\boldsymbol{x}_n$ is a given input data vector, y_n is the desired scalar output, and $\lambda > 0$ is the weight for the L2 penalty) into its dual such that the solution is in terms of ${\boldsymbol{x}_n}^{\mathrm{T}}\boldsymbol{x}_n$ or in terms of kernel matrix $\boldsymbol{K}\in\mathbb{R}^{N\times N} = \boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$, where $\boldsymbol{X}$ is the data matrix whose each row is an input instance ${\boldsymbol{x}_n}^{\mathrm{T}}$. Hint: try to express $\boldsymbol{w}$ as $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{a}$ and get $J(\boldsymbol{a})$ instead. [3]

Solving for w we get $\boldsymbol{w} = -\frac{1}{\lambda}\sum_{n=1}^{N}\{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n - y_n\}\boldsymbol{x}_n = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{a}$ ***, where*** $a_n = -\frac{1}{\lambda}\{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_n - y_n\}$.

This gives $J(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{a} - \boldsymbol{a}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\lambda}{2}\boldsymbol{a}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{a}$***. [–1 mark for not showing expression for a]***

8. In the figure the dashed lines represent a Cartesian grid, and not any data points.
    a. Sketch the approximate cluster *boundaries* for k-means with k={3,4,5}. [3] ***Note: Various solutions for k=5***



    b. Sketch the graph of Davies-Bouldin index given by $\frac{1}{k}\sum_{i=1}^{k}\max_{i\neq j}\left(\frac{\sigma_i+\sigma_j}{\|\mu_i-\mu_j\|}\right)$, for k={3,4,5}, where σ_i is the average distance of points in cluster i to its centroid, μ_i is the centroid of cluster i, and $\|.\|$ represents Euclidean distance. Exact values are not important, only the shape of the graph is needed. [2] ***A dip at 4.***