COMS 4771 HW 5 (Spring 2020)

Due: May 05, 2020 at 11:59pm

This homework is to be done **alone**. No late homeworks are allowed. To receive credit, a type-setted copy of the homework pdf must be uploaded to Gradescope by the due date. You must show your work to receive full credit. Discussing possible solutions for homework questions is encouraged on piazza and with your peers, but you must write their own individual solutions. You should cite all resources (including online material, books, articles, help taken from specific individuals, etc.) you used to complete your work.

1 Improving the confidence

Given a collection of models $\mathcal{F}$, suppose you were able to develop an algorithm $\mathcal{A} : (x_i, y_i)_{i=1}^n \mapsto f_n^{\mathcal{A}}$ (that is, given n labeled training samples, $\mathcal{A}$ returns a model $f_n^{\mathcal{A}} \in \mathcal{F}$) that has the following property: for all $\epsilon > 0$, with probability 0.55 (over the draw of $n = O(\frac{1}{\epsilon^2})$ samples,

$$\text{err}(f_n^{\mathcal{A}}) - \inf_{f \in \mathcal{F}} \text{err}(f) \leq \epsilon,$$

where $\text{err}(f) := P_{(x,y)}[f(x) \neq y]$.

Show that one can construct an algorithm $\mathcal{B} : (x_i, y_i)_{i=1}^{n'} \mapsto f_{n'}^{\mathcal{B}}$ with the property: for all $\epsilon > 0$ and all $\delta > 0$, with probability at least $1 - \delta$ over a draw of n' samples:

$$\text{err}(f_{n'}^{\mathcal{B}}) - \inf_{f \in \mathcal{F}} \text{err}(f) \leq \epsilon.$$

Moreover show that $n' = \text{poly}(\frac{1}{\epsilon}, \frac{1}{\delta})$ samples are enough for the algorithm $\mathcal{B}$ to return such a model $f_{n'}^{\mathcal{B}}$. Hence, the model class $\mathcal{F}$ is *efficiently* PAC-learnable.

Hint: Algorithm $\mathcal{B}$ can make multiple calls to the algorithm $\mathcal{A}$.)

2 Non-linear Dimensionality Reduction

Here is a simple way to accomplish non-linear dimensionality reduction:

Input: High-dimensional dataset $X = x_1, \ldots, x_n \in \mathbb{R}^D$, target dimension d
Output: $y_1, \ldots, y_n \in \mathbb{R}^d$ as the low-dimensional mapping of the given dataset

- Construct a k-nearest neighbor graph[1] G on X

[1] A k-nearest neighbor graph is simply a graph where the nodes correspond to the datapoints, and edges correspond to the Euclidean distance between the corresponding datapoints. Important: For each node/datapoint, only the k closest nodes are connected, with edge weight being the Euclidean distance between the nodes.

- Let π_{ij} denote the shortest path between datapoints x_i and x_j according to G.
- Select $y_1, \ldots, y_n \in \mathbb{R}^d$ according to the following minimization problem

$$\text{minimize}_{y_1,\ldots,y_n} \sum_{i,j} \big(\|y_i - y_j\| - \pi_{ij} \big)^2$$

(i) What is the derivative of the optimization function above with respect to a fixed y_i?

(ii) Is the optimization above convex with respect a fixed y_i? Why or why not.

(iii) Write a program in your preferred language to find a low-dimension embedding of any given input dataset. You must submit your code to Courseworks to receive full credit.

(iv) For the two datasets provided, compute your two dimensional embedding. Plot the original 3D data, its 2D PCA projection and the results obtained from your 2D embedding.

Analyze the quality of results you obtain. Under what circumstances would

- this non-linear embedding fail/succeed?
- PCA will perform better/worse than this non-linear embedding?