Lecture 22: SSL demo

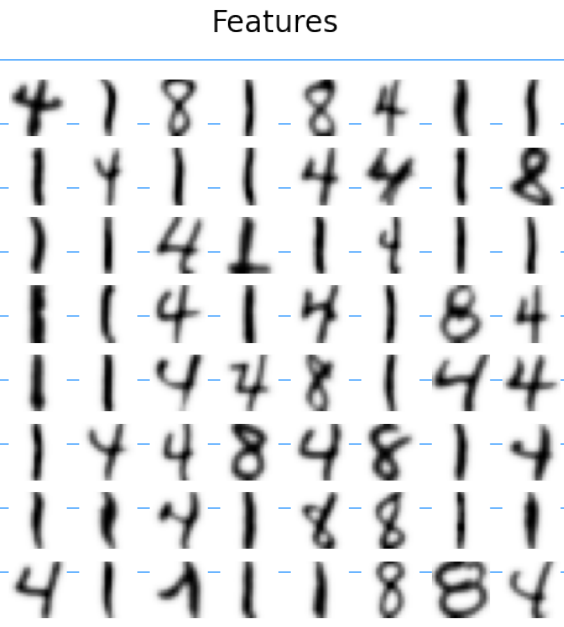
We dedicate this lecture to a demo of graphical SSL for multi-class classification of hand written digits, the MNIST dataset.

We will see:

- How to construct similarity graphs on realistic data sets.
- How to modify SSL to perform multi-class classification using one-hot encoding.

We will consider the problem of classifying images of handwritten digits from MNIST.

In particular we look at a subset of MNIST consisting of roughly 600 images of 1's, 4's & 8's.



But we wish to label the entire dataset using only 10 randomly selected labelled images!

ie, we only have 10 labelled images & the rest are unlabelled! We will solve this prob. using the graphical SSL technique of Lec 21 that was demo'd on a toy data set.

22.1 One-hot encoding

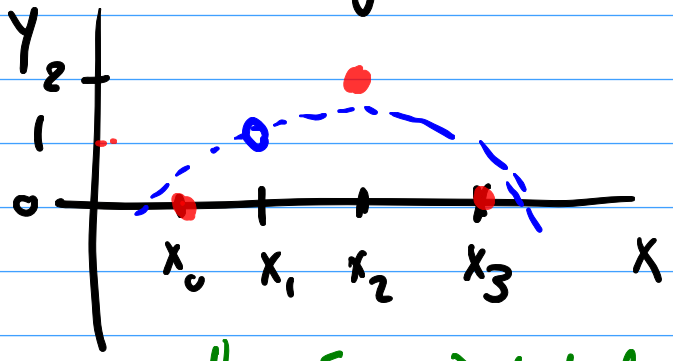
Before getting to the algorithmic details we have to deal with the issue of multi-class classification.

So far in the course we have mostly worked with binary classification where the two classes could be encoded as $\{-1, +1\}$. This worked very well & was natural in many ways.

But this encoding of binary labels does not accomodate more than two classes such as our example!

A naive approach would simply be to use the encoding $\{0, 1, 2\}$ for example. Indeed, this is how the MNIST labels are given in the dataset.

But this encoding does not work well in practice & in particular when regression is employed.



essentially if you arrange the $\{0,1,2\}$ labels in a vector $\underline{y} \in \mathbb{R}^M$ & do regression on it you are implicitly introducing an ordering of the classes where your predictor cannot go straight from class $\{0\}$ to $\{2\}$ without predicting class $\{1\}$ in between!

This ordering is arbitrary & can lead to inconsistencies.

To remedy this we will use a one-hot encoding!

Rather than labeling each point $\underline{x}_j \in X$ with a label $y_j \in \mathbb{R}$ we instead take $\underline{y}_j \in \mathbb{R}^K$ where K is # of classes in data set. More

precisely, if $\underline{x}_j$ belong to class $k \in \{0, \dots, K-1\}$

then $\underline{y}_j = e_k = (0, 0, \dots, \underbrace{1}_{k^{th}\text{ entry}}, 0, 0, \dots, 0) \in \mathbb{R}^K$

In the MNIST case, since we have only 3-digits ($K=3$) then our labels have the form

$$\underline{y}_j = \begin{cases} (1,0,0) & \leftarrow 4\text{'s} \\ (0,1,0) & \leftarrow 8\text{'s} \\ (0,0,1) & \leftarrow 1\text{'s} \end{cases}$$

Thus, the output/label data $Y \in \mathbb{R}^{M \times K}$ (M = # of labelled pts, K = # of classes) & the class of each $\underline{x}_j \in X$ is given by the index of the largest entry in the j^{th} row of Y.

This encoding does not impose any arbitrary orderings of the classes & is invariant under reordering of the classes.

22.2 SSL on MNIST

As mentioned above, we wish to perform SSL on a subset of MNIST with $N \approx 600$ images but with only $M = 10$ observed labels. There are only 3 classes in the data, the digits 1, 4, & 8.

so, $Y = \mathbb{R}^{M \times 3}$ ↙ one-hot encoding

The feature $\underline{x}_j \in \mathbb{R}^{256}$ are flattened versions of MNIST 16x16 images.

- Step 1: Construct a graph on X.

take the $\underline{x}_j$ to be the vertices of the graph with

$$w_{ij} = \exp\left(-\frac{\|\underline{x}_i - \underline{x}_j\|^2}{2\ell^2}\right)$$

- Step2: Compute eigen decomposition of graph Laplacian.

$$L = D - W \quad \text{(unnormalized Laplacian)}$$

& write

$$L = Q \Lambda Q^T, \quad Q = \left[\underline{q}_0 \middle| \underline{q}_1 \middle| \cdots \middle| \underline{q}_{N-1} \right]$$

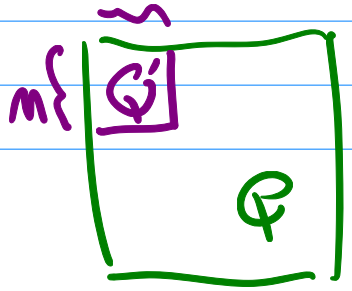
- Step3: Perform regression on labelled set.

Without loss of generality we assume $\underline{x}_0, \cdots, \underline{x}_{M-1}$ are the labelled points.

Choose $J \geq 1$ the # of eigenvectors we wish to use for regression

$Q' \in \mathbb{R}^{M \times J}$ submatrix of Q

$$Q'_{ij} = Q_{ij} \quad \text{for} \quad \begin{matrix} i = 0, \cdots, M-1 \\ j = 0, \cdots, J-1 \end{matrix}$$



$$\min \|Q\beta - Y\|_F^2 + \lambda \|\beta\|_F^2$$

finally solve regression problem

$$\beta = \left[\underline{\beta}_0 \,\middle|\, \underline{\beta}_1 \,\middle|\, \cdots \,\middle|\, \underline{\beta}_{K-1}\right] \qquad \hat{\underline{\beta}}_k = \underset{\underline{\beta}_k \in \mathbb{R}^n}{\operatorname{argmin}} \|Q'\underline{\beta}_k - \underline{y}_k\|_2^2 + \lambda \|\underline{\beta}_k\|_2^2$$

$$Y = \left[\underline{y}_0 \,\middle|\, \underline{y}_1 \,\middle|\, \cdots \,\middle|\, \underline{y}_{K-1}\right]$$

*Note $\hat{\beta} \in \mathbb{R}^{J\times 3}$ since $Y \in \mathbb{R}^{M\times 3}$!

- Step 4: Predict labels on all of X.

With $\hat{\beta}$ we can predict the one-hot vectors of all pts on the graph simply as

$$\mathbb{R}^{N\times 3} \ni \tilde{Y} = \underbrace{\left[\underline{q}_0 \,\middle|\, \underline{q}_1 \,\middle|\, \cdots \,\middle|\, \underline{q}_J\right]}_{\mathbb{R}^{N\times J}} \underbrace{\hat{\beta}}_{\mathbb{R}^{J\times 3}}$$

But we are not done yet! Rows of $\tilde{Y}$ are not 0's & 1's. So we threshold them!

i^{th} row of $\tilde{Y}$ ↳

$$\underline{\tilde{y}}^i \longrightarrow \underline{\hat{y}}^i \begin{cases} \text{set largest entry} = 1 \\ \text{\& rest to } 0. \end{cases}$$

$$\tilde{Y} = \left[\begin{array}{c} \underline{\tilde{y}}^0 \\ \hline \underline{\tilde{y}}^1 \\ \hline \vdots \\ \hline \underline{\tilde{y}}^{N-1} \end{array}\right]$$

- Comparing first 30 labels.
- Yellow boxes indicate the corresponding class

- error rate: 28% misclassified