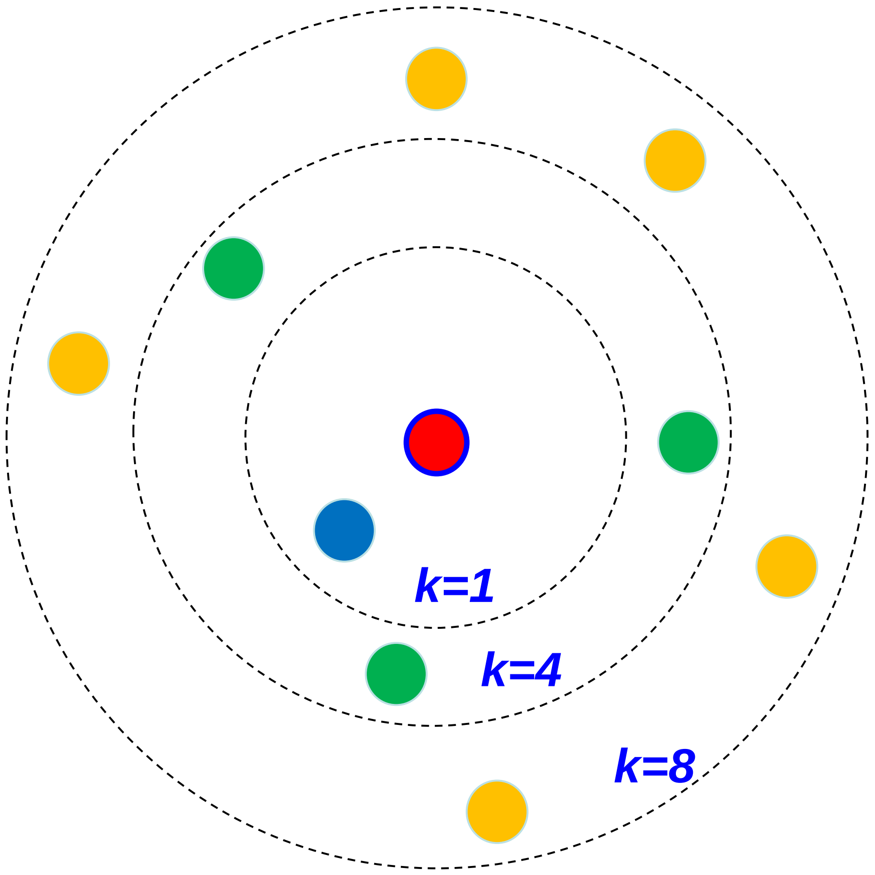
Scientific Computing with Python

Mini-Project:
k Nearest Neighbor of images



Background

- **As our first approach, we will develop what we call a Nearest Neighbor Classifier to classify hand written digits dataset MNIST. This method will allow us to get an idea about the basic approach to an image classification problem.**

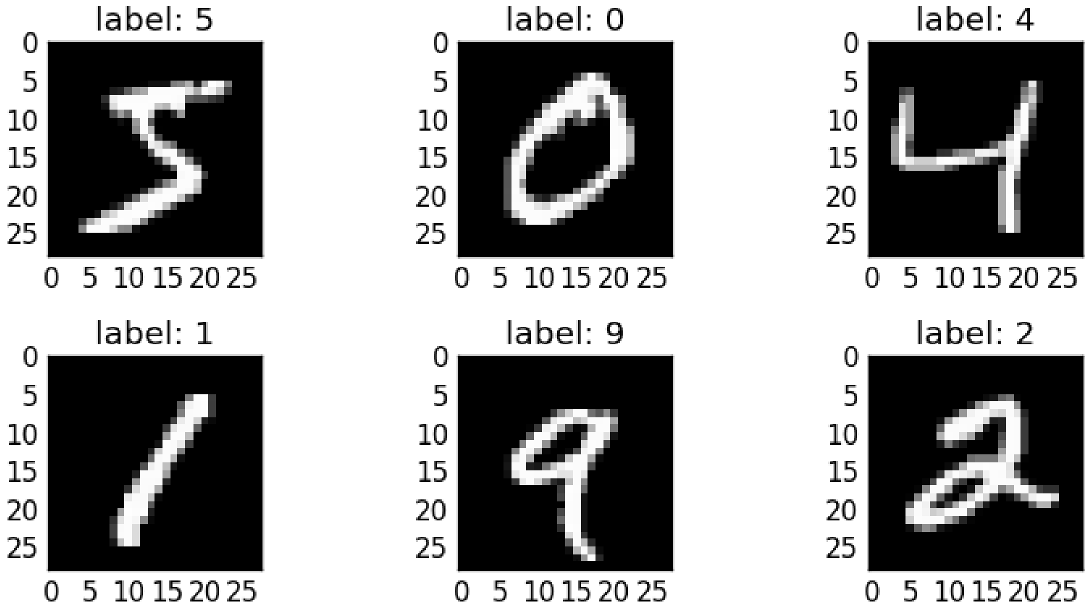
- **Example image classification dataset: MNIST.**
  - One popular toy image classification dataset is the MNIST dataset. This dataset consists of 60,000 tiny images that are 28 pixels high and wide.

- **Each image is labeled with one of 10 classes (0-9). These 60,000 images are partitioned into a training set of 50,000 images and a test set of 10,000 images.**

- **In the image below you can see 10 random example images from each one of the 10 classes:**

Introducing the MNIST problem

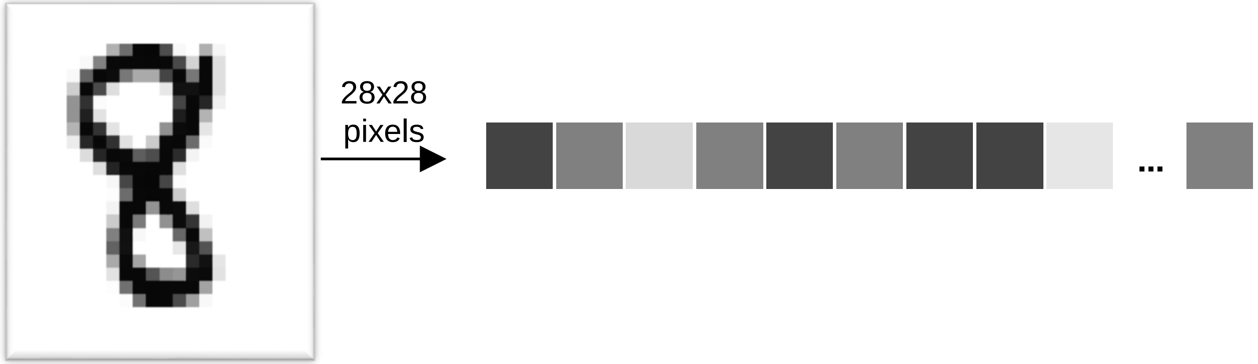
- **MNIST (Mixed National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems.**
- **It consists of images of handwritten digits like these:**



- **The MNIST database contains 50,000 training images, 10,000 validation images and 10,000 testing images.**

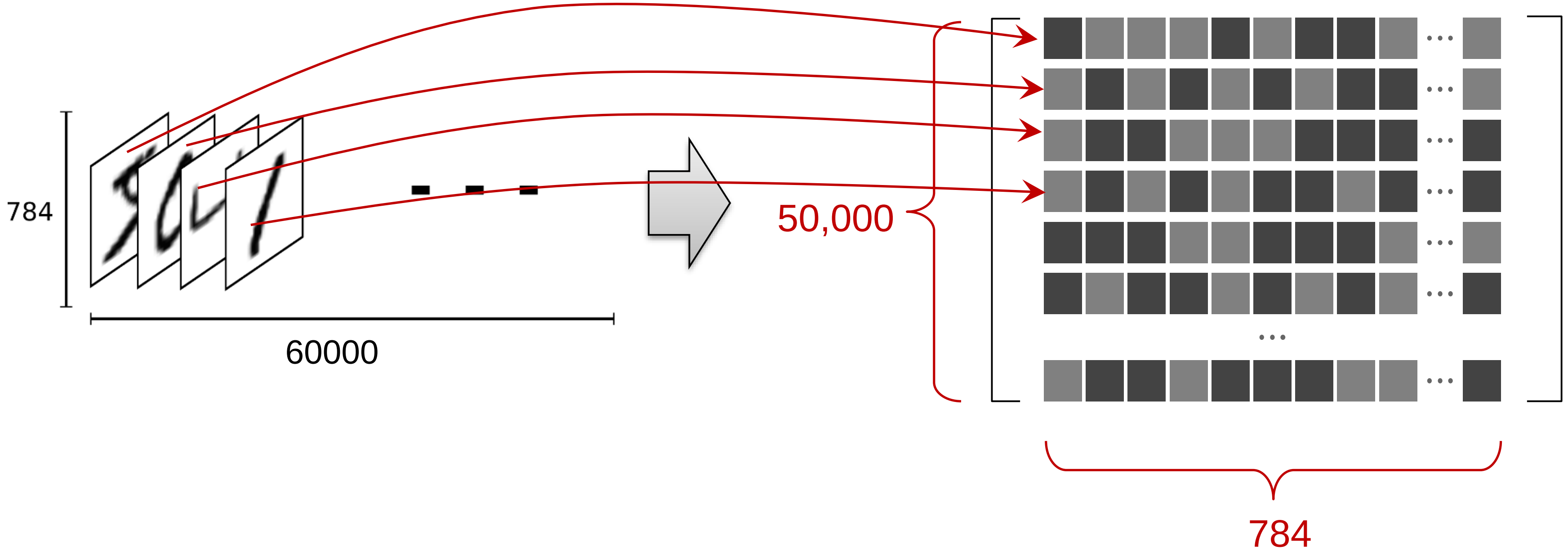
Flatten the 2D image into 1D vector

- **We first flatten each image into a vector of 28x28 = 784 numbers. It doesn't matter how we flatten the array, as long as we're consistent between images.**
- **From this perspective, the MNIST images are just a bunch of points in a 784-dimensional vector space.**



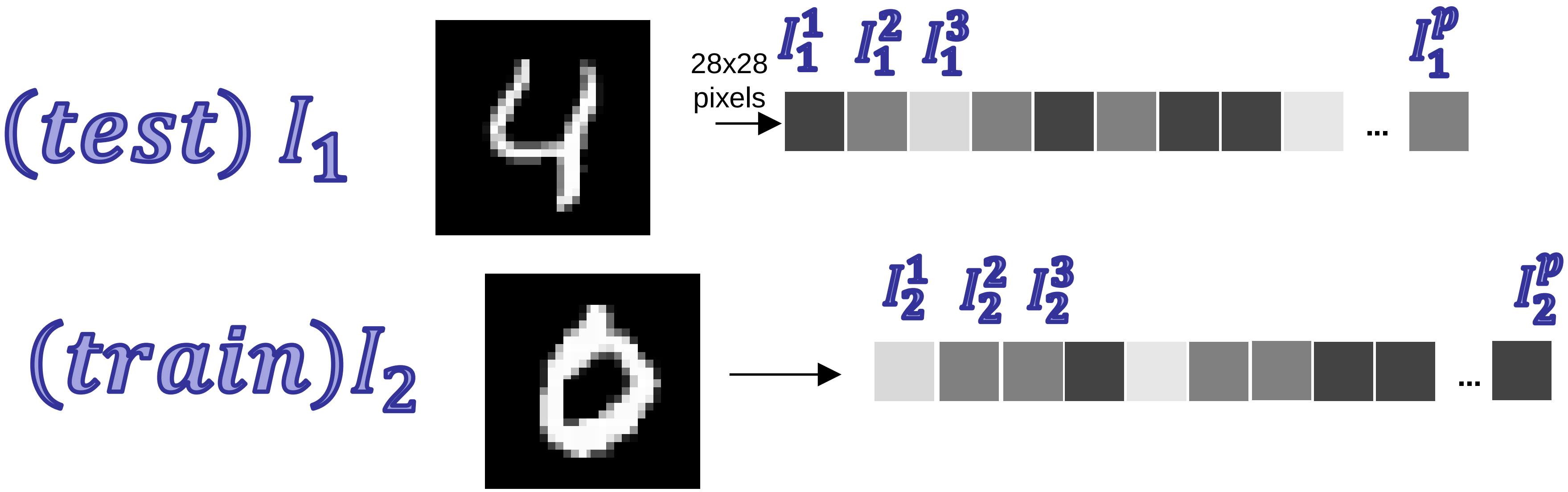
Result of the Flatten Operation

- **The result is that the training images is a matrix (tensor) with a shape of `[50000, 784]`.**
- **The first dimension is an index into the list of images and the second dimension is the index for each pixel in each image.**
- **Each entry in the tensor is a pixel intensity between 0 and 255, for a particular pixel in a particular image.**

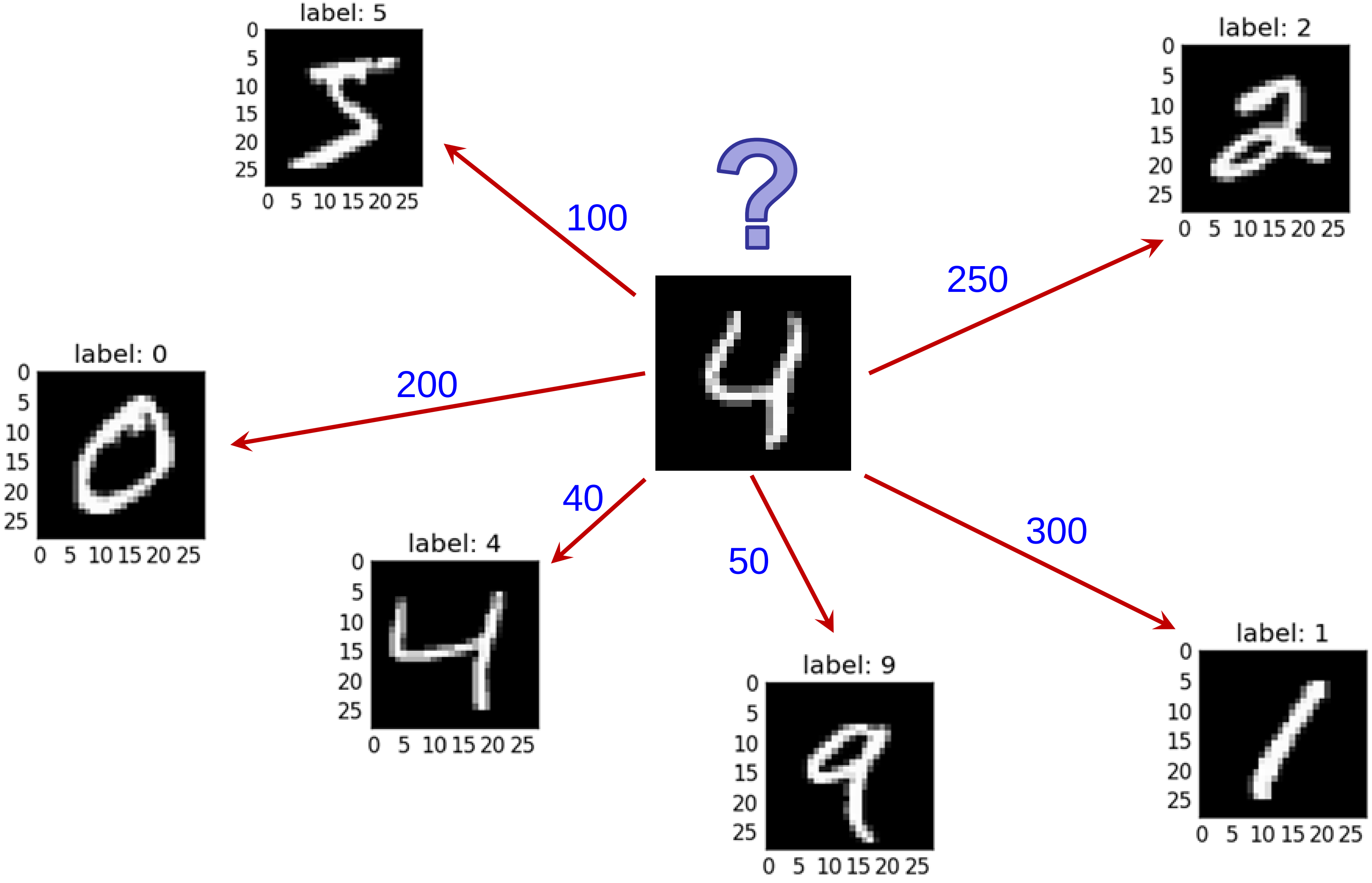


How kNN work for image classification?

- **For each test image, we "*compare*" the image with all the training set images, then we will find *k* nearest neighbor images and let the k images vote for the test image and determine the label of the test image.**
  - How do we compare image?
    - Use the L2 distance (Euclidean distance between two vectors):
    - $d_2(I_1, I_2) = \sqrt{(I_1^1 - I_2^1)^2 + (I_1^2 - I_2^2)^2 + \cdots + \left(I_1^p - I_2^p\right)^2} = \sqrt{\sum_{i=1}^{p}\left(I_1^i - I_2^i\right)^2}$

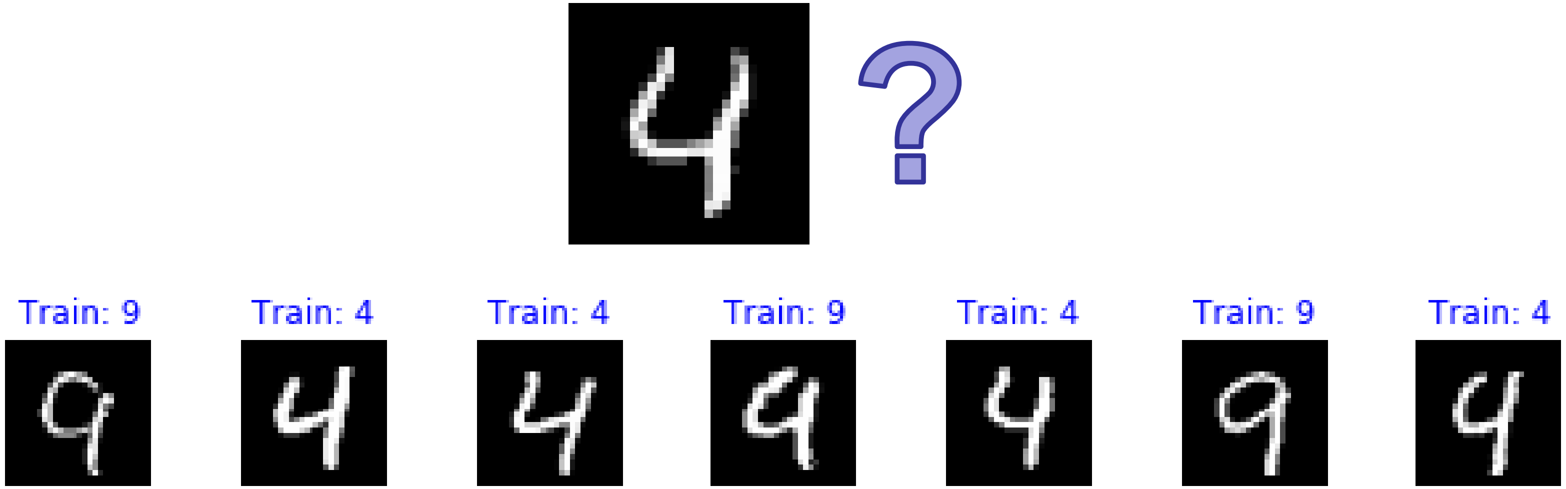


Visualization of K Nearest Neighbor



How do the k images vote the test image?

- For example, given the below image that we need to label, we have found 7 nearest neighbors for this image:



- So based on the 7 neighbors, 4 votes “4”, 3 votes “9”, this image will be labeled as 4