kNN Classifier

Machine learning course : Technical report no.3

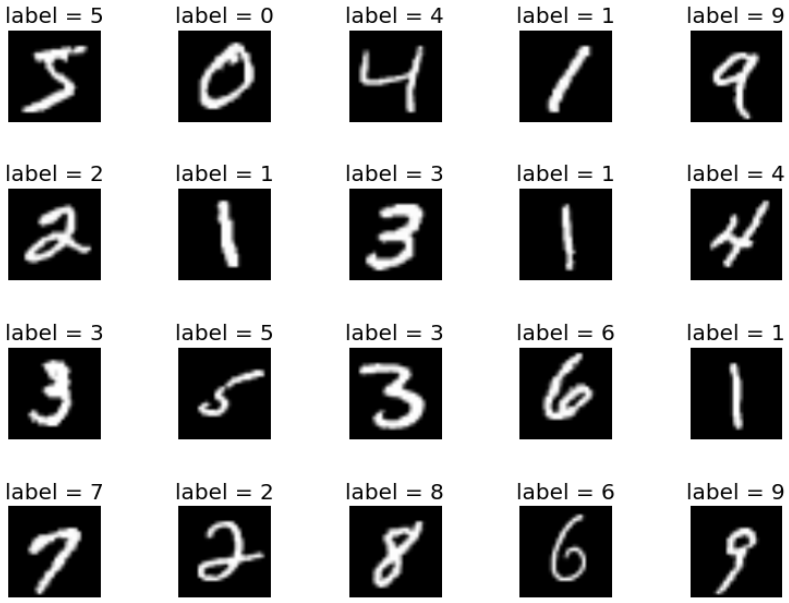

FIGURE 1 : DATA SET WITH THEIR LABELS

Théo DEPALLE

November 24, 2019

1. Abstract

In some cases you may want to your robots or your application to recognize something in an image like an object, a number, an activity … but how can a computer recognize an image? Image recognition is a big field of Machine Learning. It already exist many algorithms, and the best one often depends of the considered data. In this report we will use kNN (k-Nearest Neighbours) classifier in order to classify handwritten images.

2. Introduction

kNN Classifier belongs to supervised learning. It is mostly used for patterns recognition. It is a non-parametric model thus it is very suitable when we have no assumptions about the distributions probability of our data. It can be used on many domains. In our case we will used it on images recognition. Particularly we will try to find correlation between an image that we want to label and a training set images already labelled.

3. kNN Classifier

3.1.Theory

Let's consider a training set X of size n :

$$X = \{x_1 \quad \cdots \quad x_n\}$$

With the corresponding labels classes T also of size n :

$$T = \{t_{x_1} \quad \cdots \quad t_{x_n}\}$$

And k an integer ≥ 1.

Now we want to label a query point $\bar{x}$ with the label $\bar{t}$.

kNN Classifier select the k-data in the training set which are closest to $\bar{x}$, analyse their labels, and select the majority to predict $\bar{t}$.

for $i = 1 \; to \; n$

$$\{x_l \quad \cdots \quad x_k\} = top\text{-}k(d_i(\bar{x}, x_i))$$
$$\bar{t} = mode\{t_{x_l} \quad \cdots \quad t_{x_k}\} \qquad (1)$$

with "mode" the most frequent item in a sample.

In order to choose the closet k-data, it compute the Euclidean distance between $\bar{x}$ and all x_i in the training set :

$$d_i(\bar{x}, x_i) = \sqrt{(\bar{x} - x_i)^2} \qquad (2)$$

3.2.Comments

As you can imagine, the kNN Classifier has a strong dependence of k. How can we choose the best k in order to label our query point? Let's think about the different possible cases.

For that let's consider un 2D set of data with 2 different classes with the label A and B.

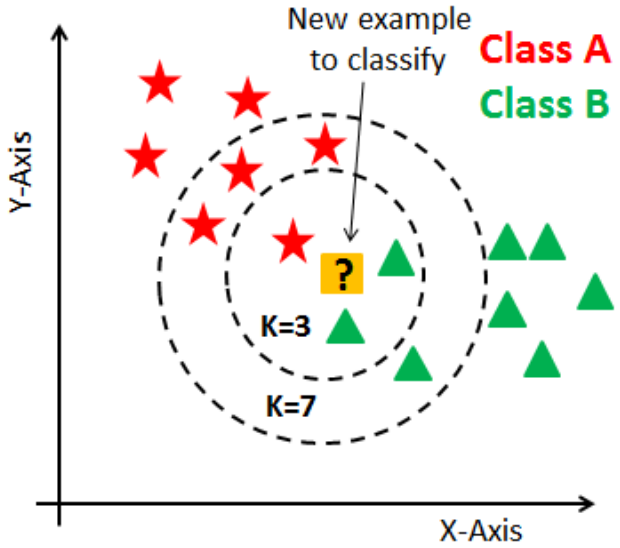

FIGURE 2 : KNN CLASSIFICATION ON 2D DATA SET

As you can see on this figure there is different result of $\bar{t}$ in function of k :

- k = 1, $\bar{t}$ = Class B
- k = 2, $\bar{t}$ = rand between A and B
- k = 3, $\bar{t}$ = Class B
- …
- k = 6, $\bar{t}$ = rand between A and B
- k = 7, $\bar{t}$ = Class A

What is the best k ? First we can notice a dependence of the parity. For instance, when k = 2, there is no majority. This dependence can be explain by the number of classes. Indeed such there are two different classes, if k is divisible by the number of classes, there are some odds that there will be no majority. Thus the best is to choose a k which is not a divisible by the number of classes.

Moreover as we decrease the value of k to 1, our predictions become less stable. Imagine k = 1 and our query point is surrounded by several A and one B but the B is the nearest neighbour. We would think the query point is most likely A, but because k = 1, kNN incorrectly predicts that the query point is B. Inversely, as we increase the value of k, our predictions become more stable due to majority voting, up to reach a step where we took to many point and the error become to increase.

Finally in order to select the best k we should run our kNN classifier several time with several k not divisible by the number of classes, and take the k that reduce the number of error.

3.3.Experiment

3.3.1. Description of the data set

Our data set is called MNIST. It is a set of handwritten digits (0 to 9). These digits have been normalized and centered in a black and white image of 28x28 pixels. Each images has been put in a vector of length 784. We have four different files :

- **train-images** : training set images (60 000 x 784)
- **train-labels** : training set labels (60 000 x 1)
- **t10k-images** : test set images (10 000 x 784)
- **t10k-labels** : test set labels (10 0000 x 1)

Thus we have approximately 15% data for testing and 85 % for training

3.3.2. Description of the experiment

The objective it is to label each test images such as the **Figure 1**.

I use Python in order to compute the kNN Classifier. Thirst we need to load each data in matrix. Train-images become X and train-labels T , with $n = 60\,000$. We consider t10k-images such as

$$\bar{X} = \{ \overline{x_1} \quad \cdots \quad \overline{x_m} \}$$

and t10k-labels :

$$\bar{T} = \{ \overline{t_{x_1}} \quad \cdots \quad \overline{t_{x_m}} \}$$

avec $m = 10\,000$.

and $\bar{x_i}, x_i$ vector of length $r = 784$.

Then for each test images $\bar{x_i}$ we compute the kNN classifier.

(2) become : $d_i(\bar{x}, x_i) = \sqrt{\sum_{r=0}^{784} (\bar{x} - x_i)^2}$

Thus, for one test image we obtain a $(1 \times n)$ matrix. And after applying the kNN classifier on each test images we obtain a $(m \times n)$ matrix M.

After, we have to sort by ascending order each rows, take the indices of the k first train images, look at their labels in the train-labels matrix and take the majority in order to labels each test-images.

3.3.3. Results and Explanations

M took a lot of time to obtain as we have to run two big loop (60k and 10k). Once this matrix is computed, I saved it and try different value of k from 1 to 50. Then I compare each results obtained and the real result (test-labels) and finally compute the error by dividing the error number by m.

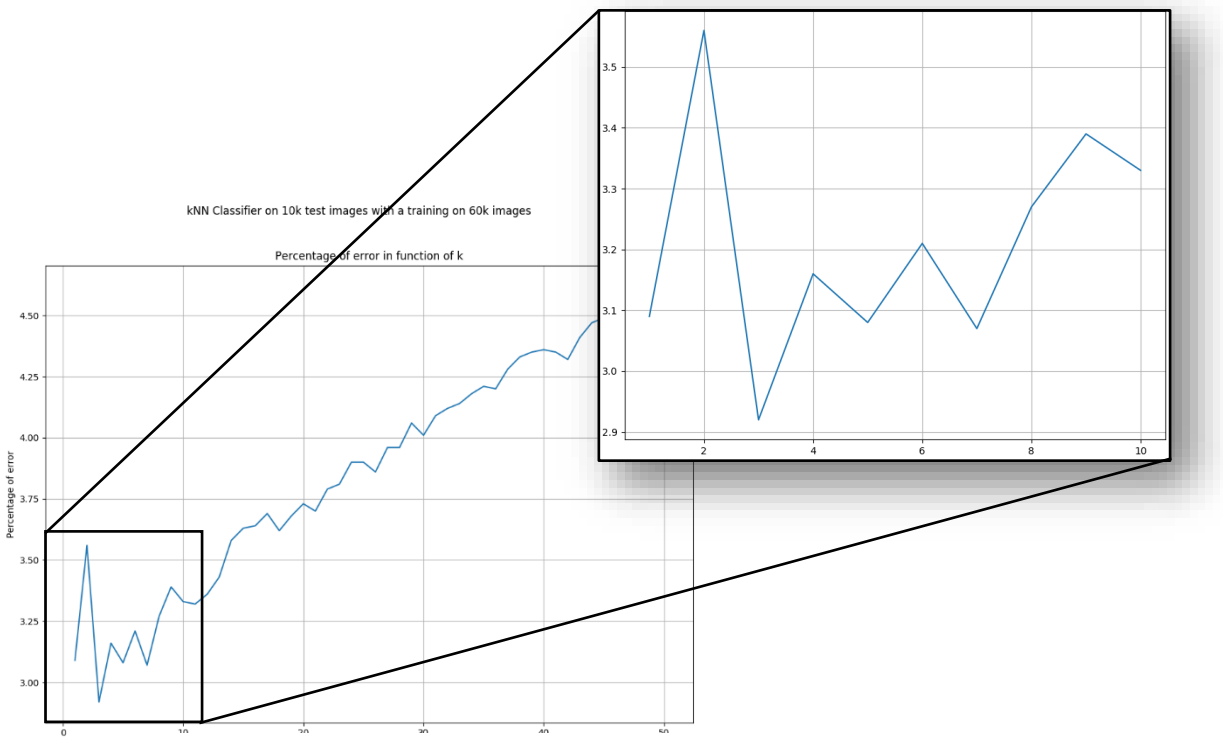

FIGURE 3 : IMPACT OF K IN THE ACCURACY OF THE CLASSIFIER

As previously said in **3.2**, if we increase to much k we reach a point (approximately 10) where the error increase significantly. So, the best choice seems to choose a k between 3 and 10.

We can easily see on the zoom an influence of the parity. Indeed with even number of k the error is bigger. The explanation can be that the classifier doubt often between two different labels (for instance a 3 with a 3 or a 7) and thus it is better to take k as an odd number to avoid random choose for majority.

The two best values of k seems to be 3 and 7 with respectively an accuracy of 97.08 % and 96.93 %.

In the mean time I compute the error for each digit in function of k. For that I make an error vector of length 10. Each time my result is different from the test-labels, I add one at the corresponding digit in the error vector. At the end I divide each values of my vector with the number of each digits in the test-labels in order to have the percentage error. I obtain the figure below.

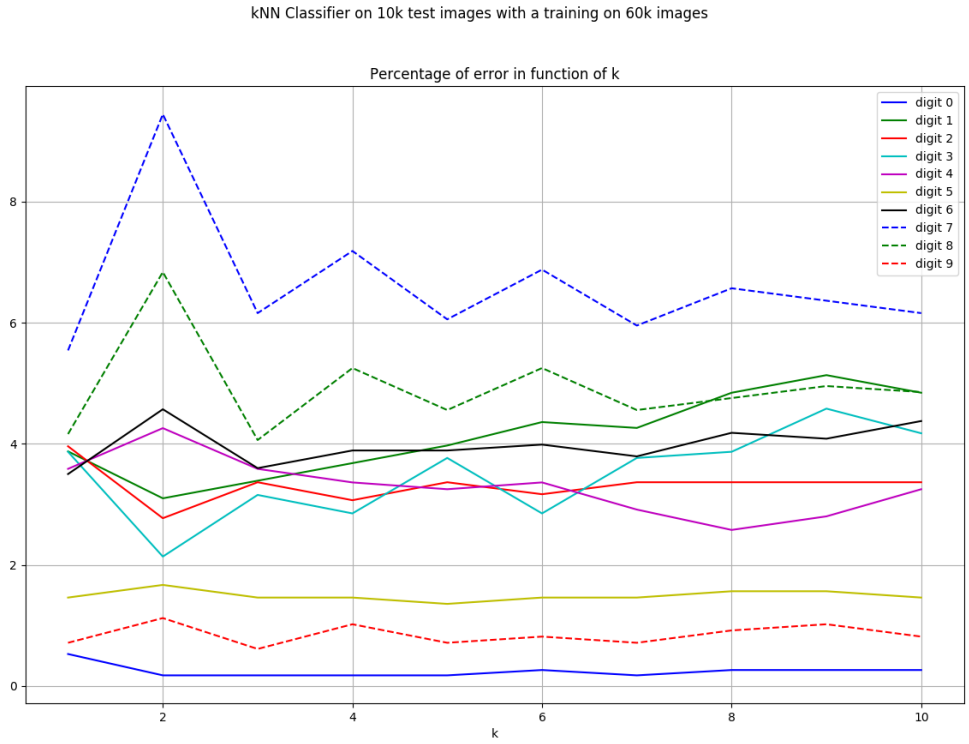

FIGURE 4 : IMPACT OF K FOR EACH DIGITS

Here again we can observe a dependences between the parity of k and the accuracy. But with difference between the digit. For instance for the digits 9, 8, 7 and 6, the accuracy is better with an odd k. Inversely for 3, 2 the accuracy is better with an even k. Once again it is because of the majority.

Now I want to have a clearest view of each error for each digit with $k = 3$ and $k = 7$. I obtain the figures below.

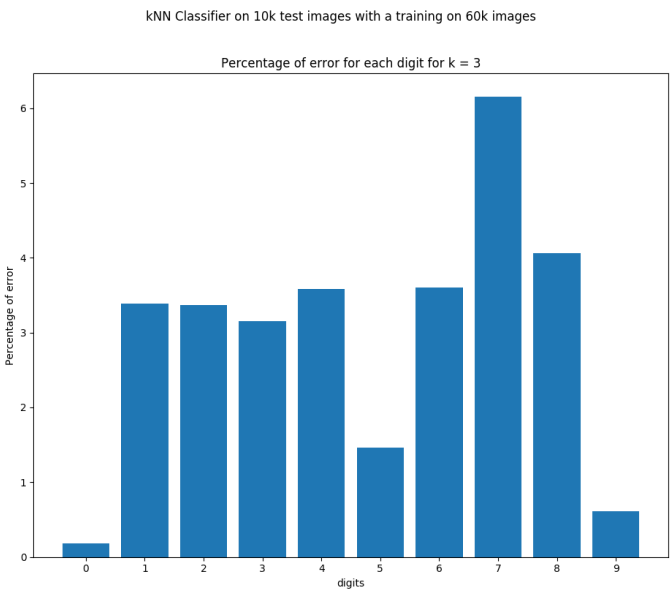

FIGURE 5 : ERROR FOR EACH DIGITS WITH K = 3

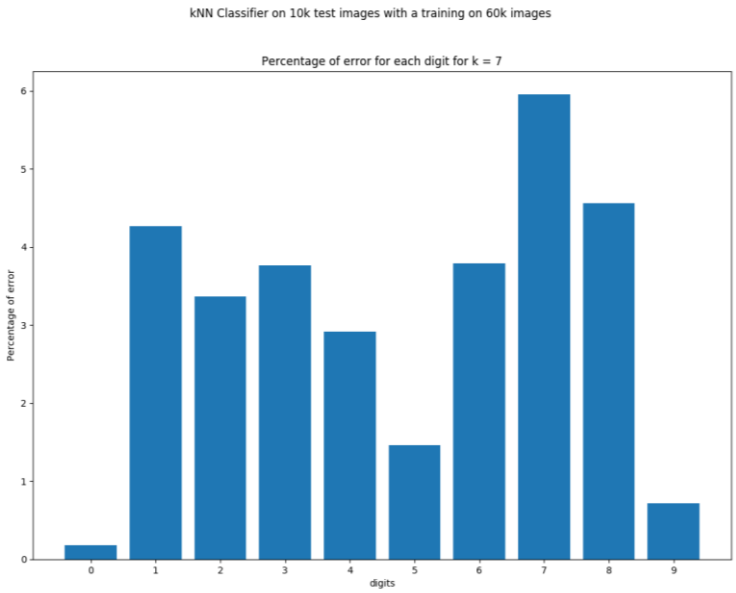

FIGURE 6 : ERROR FOR EACH DIGITS WITH K = 7

On this two figure, we can see that the digits 0 and 9 are very well recognized, with an accuracy over 99.5%. The first explanation is because everyone write a zero with a circle or an oval, and the 9 with a circle and a line, sometime bend at the bottom, without a lot of difference between each person.

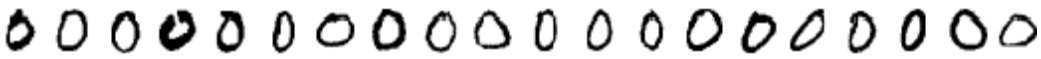

FIGURE 6 : DIFFERENT WRITEN ZEROS

At the contrary the least well recognized number is 7, first it is because we can write it with two different way : with or without the middle bar. Moreover, the 7 is very close to the 1 or some times the 3 (as the figure below). As well, the others digits can be easily be confused with the others one, like 8 with 4, 5 close to 6 etc..

FIGURE7 : 7 (CLOSE TO 3)

FIGURE 8 : 8 (CLOSE TO 4)

If we want to recognize a particular digit, it will be better to choose k in accordance with the **figure 4.** For example if we want to detect with more accuracy the digit 4, it will be better to chose $k = 8$.

4. Conclusion

To conclude thanks to the kNN Classifier we can predict the label of a handwritten digit in function of a big data set. This classifier is very well suitable for data without known probability. In order to have a good accuracy it is important to test different value of k. On the other hand, this classifier take a lot of time to compute different labels with a big set of training data (about one hour for pictures with 784 pixels with my computer).

Now it will be good to test the dependence of the number of training data on this classifier.