Cairo University
Faculty of Computers and Artificial Intelligence

Supervised Learning

Assignment_2

Prepared By:

Christina Montasser Saad 20190382

3AI_S1 (Section 1)

Content

1. About data:

Data with the title of "**Minst Dataset**".

1.1 Description

Minst is a handwriting image dataset with 60,000 training points and 10,000 test points. Each image has a size of 28x28. They're RGB images which means their values lay between 0 -255.
Each image has a label number which is an integer from 0 -9.
I downloaded it from Keras library.

1.2 Sub-Conclusion

- It's a supervised classification problem.
- Must use a feature reduction technique as it's too large to have a feature vector of 784x1.
- It's recommended to invert it from RGB to gray-scale images.

2. Dealing with Features

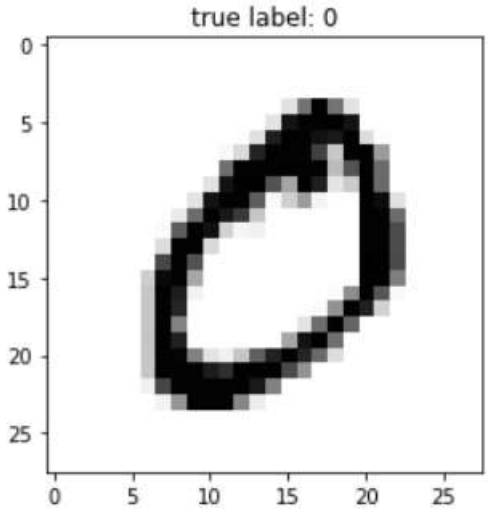
2.1 Binarization

Binarization[1] seems to be an important step especially in cases of document images due to the problems of shadows, non-uniform illumination and blur. Binarization helps to decrease computational load which leads to increase efficiency. Besides, binarization helps to update Optimizely KNN classifier by calculating dis-similarities instead of Euclidean distance.

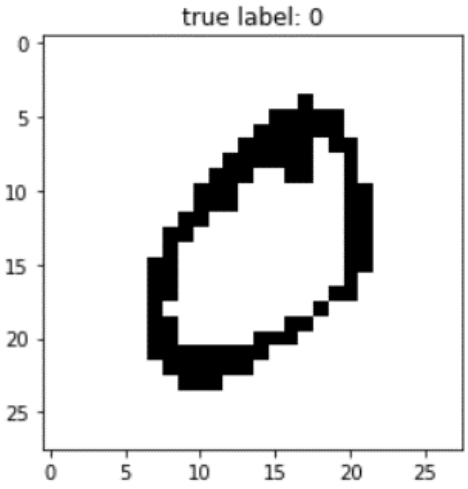
I faced some of the binarization challenges, like noise and varying in gray levels of the written number. **Global binarization method**[2] is used with Minst data set. Dependently on how the numbers are written, I choose different threshold 200 and 150. Here are plotting samples after and before binarization, with the two chosen different threshold

ds.

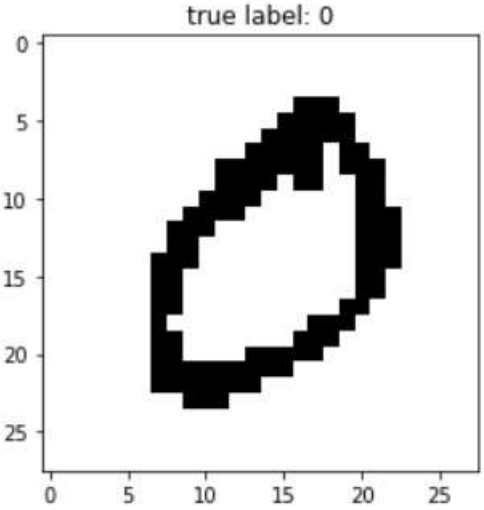
[1] Binarization computes the threshold value that differentiate object and background pixels.
[2] Choosing one threshold value over the whole image.



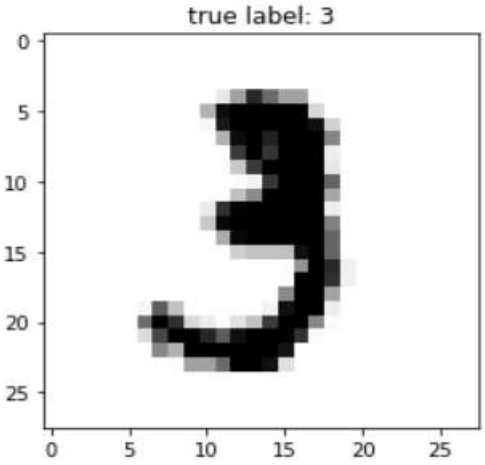
Without binarization



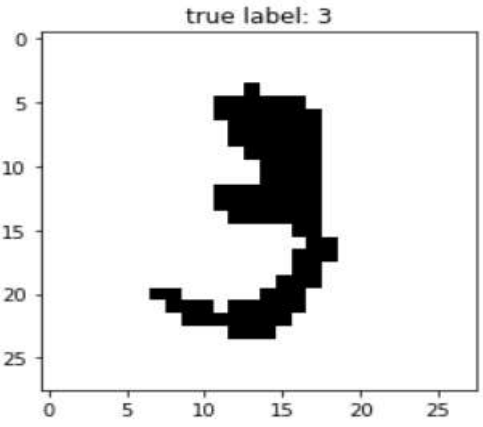
With binarization threshold 200



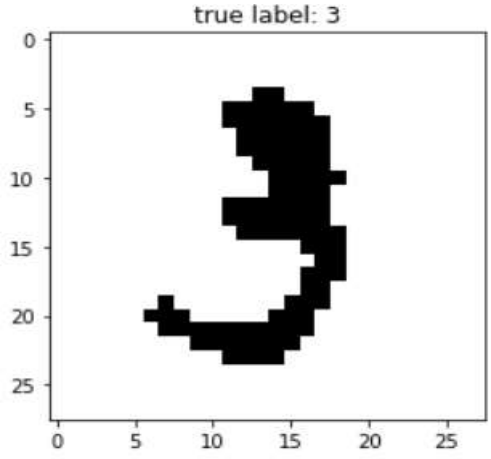
With binarization threshold 150



Without binarization



With binarization threshold 200



With binarization threshold 150

Both works fine, but somehow, I see "200" as a threshold fits more to not let noise and blur being considered as important bits. Because it may lead to matched a number can be written in small on-bits with another written in large on-bit, just because the first is noisier and blurrier. Especially I used here the "Centroid"[3] technique.

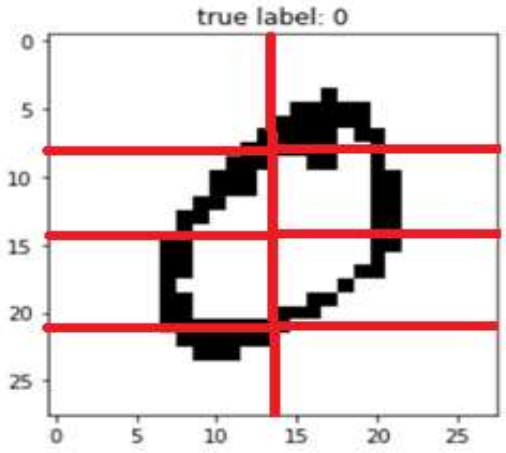
[3] the weighted average of all the pixels of the number.

Let this discussion open till the classifier says what it sees…

2.2 Feature reduction

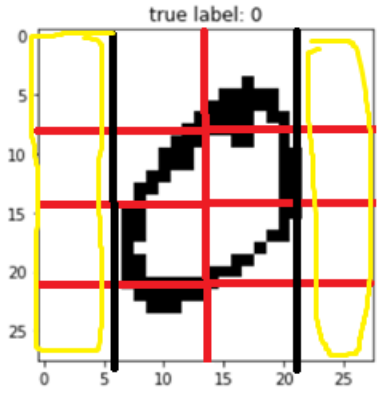
2.2.1 Chosen Size

I split all images to 8 parts, 2x4.



This chosen size depends on where most of the objects lay. Most numbers in Minst dataset lay in the middle, so it would be redundant to split a part of the most-left or right x axis of any image because those are totally white background parts so their x, y moments will be zeros.

Those yellow colored spaces are meaningless. Even if they contained pixels of the written numbers, ignoring them doesn't affect anything. Ignoring them can be described as different binarization threshold and those bits were noise.



2.2.2 Moment algorithm

Centroid is the weighted average of all pixels. "Weighted" means there are pixels contribute more than others so it's multiplied by a factor of its importance.

$C_x = \frac{\sum_{x=1}^{m} x\, f(x,y)}{f(x,y)}$ $C_y = \frac{\sum_{y=1}^{m} y\, f(x,y)}{f(x,y)}$

$C_x = \frac{M10}{M00}$ $C_y = \frac{M01}{M00}$

$$M_{ij} = \sum_{x=1}^{m} \sum_{y=1}^{n} x^i y^j I(x,y)$$

Note: M_{00} which refers to the image function.

I applied this algorithm on each block of pixels of the split parts. It's 2x4, the returning was an ordered pair (x, y) -size 2-. It ended with each image has *16 features*.

Progress: from 728 features to only 16.

Now, our data with a size of ***60,000x16***

3. KNN Classifier

3.1 Algorithm

3.1.1 Dis-similarities (Euclidean distance)

$d(x_1, x_2) = \sum_{m=1}^{D} I(x1 \neq x2)$

Because features' value is binary. Calculate dis-similarities instead of Euclidean distance is better.

3.1.2 Neighbors

Ordered them ascendingly and choose the first k neighbors.

3.1.3Voting

Find the dominant class over the k returned labels.

4. Different tests

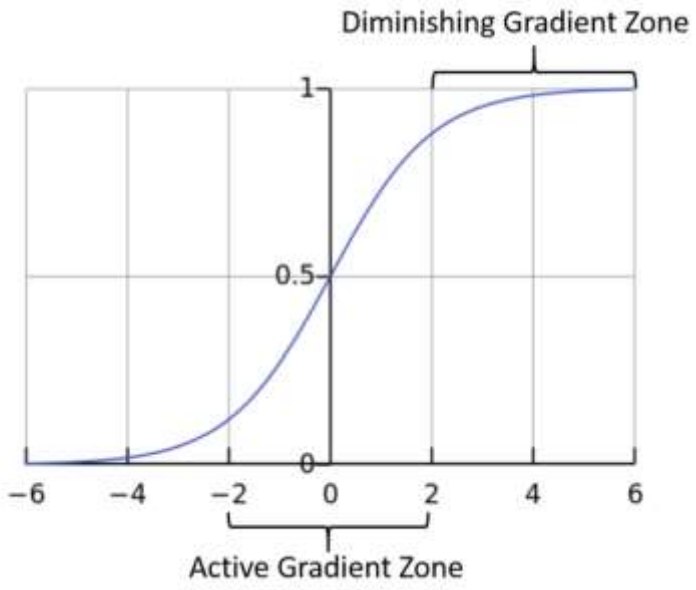
Test#	Binarization	Feature size	Accuracy	KNN distance
Test1	Without	2x4	92.74	Euclidean distance high computational function
Test2	With Threshold = 150	2x4	**92.53**	Dis-similarity
Test3	With Threshold = 200	2x4	91.74	Dis-similarity

Conclusion: It seems with threshold 150 is the best overall.

5. Neural Network

5.1. Activation function

$$A = \frac{1}{1+e^{-x}}$$



First, I got bad accuracy, within 10-15%! After checking sigmoid I found that the initialized weights lead to lay data in the saturated region which caused no variation between class and another one. Studying sigmoid clearly ended with I should guarantee laying data within the variation region, I achieved that trough

1. Initializing weights in the region of (-2, 2).
2. Normalizing inputs data so when multiplying them with wrights doesn't affect so much what I try to achieve.

From that I'm sure that each data point has its identity and affect the weights differently from each other.

5.2. Different tests with comment

- Gray-scale normalized data with the whole test data

Neuron	Weight range	Train size	eta	epoic	Train accuracy	Test accuracy	Comment
6, 12	-4, 2 integers	10000	0.05	100	72.729	70.74	Range of weight ups accuracy from 10 to 70, because that guarantee the identity of each data point where lies. • **Specify Region**
8, 12, 6	-4 ,2 integer	10000	0.03	200	73	71.509	More iterations & more neurons but less learning rate increases accuracy.
6, 15, 8	-2, 2 integer	30000	0.05	100	78	77.08	**More training** data is suggested because model learns more.
6, 15, 8	-2, 2 uniform	30000	0.05	100	80	79.759	*IMPORTANT*, integers from (-2, 2) like a set of {-2, -1 ,0 ,1 ,2}, finite discrete values, it leads to have multiple neurons with same initialized weight and it's not preferred as those same-neurons become meaningless. I found initializing them within a period [-2,2], infinite possibilities of numbers increase accuracy.
12	-2, 2 uniform	60000	0.05	100	84.09	84.66	Used from above conclusions →uniform, more training data. But one layer, and 100 iteration.
20, 18	-2, 2 uniform	30000	0.05	200	90.456	89.72	Increased layer from and iterations, less data. It gives higher accuracy from the previous one because the previous one needs more layers tat can learn from this big size.
35, 20	-2, 2 uniform	60000	0.05	300	92.708	91.97	Used from above conclusions →uniform, more training data, increasing neurons of each layer and iterations

6Conclusion

Without binarization takes much time that with and the difference accuracy is so small, we can achieve that difference and more from the architecture of our neural network.

Without binarization

Last two cases without binarization

20, 18	-2, 2 uniform	30000	0.05	200	91.5	90.68	Increased layer from and iterations, less data. It gives higher accuracy from the previous one because the previous one needs more layers tat can learn from this big size.
35, 20	-2, 2 uniform	60000	0.05	300	92.995	92.46	Used from above conclusions →uniform, more training data, increasing neurons of each layer and iterations

After half the iterations the accuracy increases slowly, like 1-3%. The dropout technique may increase the increasing rate.

Not preferred having many neurons running on small amount of data.