

Handwritten Digit Recognition Model using KNN Algorithm - Statistical Tests

Accuracy

Definition: we will define the accuracy of the program to be the percentage of correct predictions over all the possible inputs, since testing the model over every possible input is not feasible.

We used for training and for testing the dataset called mnist_784 witch The dataset comprised pairs of 28×28-pixel images and corresponding digit labels, we will estimate the Accuracy by the empirical accuracy over a test subset of the dataset.

But first, let's analyze what would you expect from a completely random predictor:

If we used completely random predictor, we expect an accuracy of 10%.
Explanation:


Given a random pair, we denote by X_i the random variable that receives 1 if we succeeded in pairing and receives 0 if we made a mistake in pairing. Hence $\forall i \in \{1, \dots, 1000\}: X_i \sim \text{Ber}(0.1)$. hence:

$$\Rightarrow accuracy = \frac{1}{1000} \sum_{i=1}^{1000} X_i \sim \frac{1}{1000} \text{Bin}(1000, 0.1)$$

$$\Rightarrow \mathbb{E}[accuracy] = \frac{1}{1000} \cdot (1000 \cdot 0.1) = 0.1 = 10\%$$

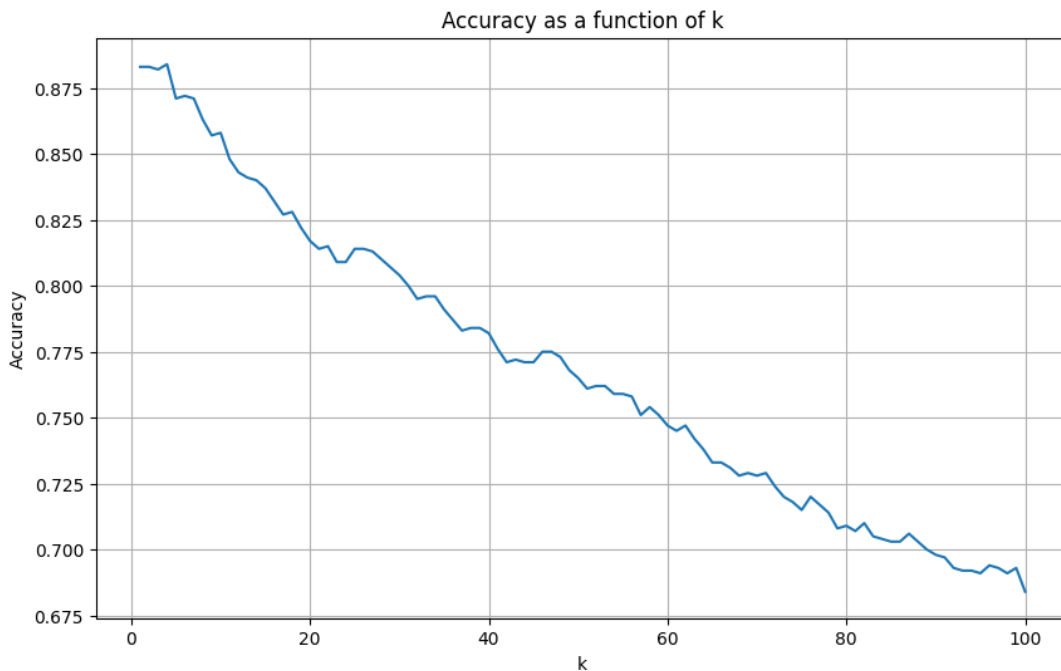
Now let's run the algorithm using the first $n = 1000$ training images, on each of the test images, using $k = 10$. And calculate the accuracy of the prediction:

Running the function called `run_knn()` we get an accuracy of 85.8%

 Accuracy: 0.858
0.858

Now let's examine the relation between k and model accuracy (over $k \in \{1, 2, \dots, 100\}$). Plotting the accuracy as a function of k we get the following result:

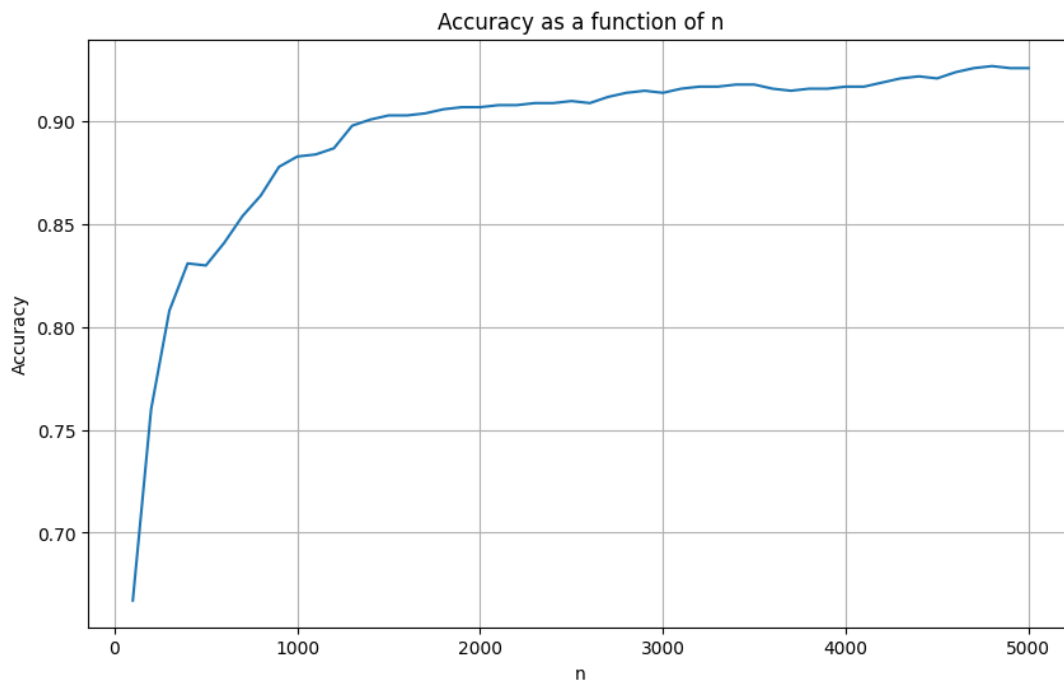
```
# Plot predictions accuracy as function of k
k_values = range(1, 101)
accuracies = [run_knn(k=k) for k in k_values]
plt.figure(figsize=(10, 6))
plt.plot(k_values, accuracies, marker='')
plt.xlabel('k')
plt.ylabel('Accuracy')
plt.title('Accuracy as a function of k')
plt.grid(True)
plt.show()
```



We can see that to get maximum accuracy you have to choose $k=1$. That is, the image is simply matched to the *label* of the closest image.

Now we want to examine the relationship between the size of the training dataset and the model accuracy:

```
# Plot the prediction accuracy as a function of n=100,200,...,5000 (with k = 1)
n_values = range(100, 5001, 100)
accuracies = [run_knn(n=n,k=1) for n in n_values]
plt.figure(figsize=(10, 6))
plt.plot(n_values, accuracies, marker='')
plt.xlabel('n')
plt.ylabel('Accuracy')
plt.title('Accuracy as a function of n')
plt.grid(True)
plt.show()
```



In the drawing we can see that when you increase n you get better accuracy, this is indeed an expected result because when a model is trained on more drawings it is observed that its ability to pair images will improve.

Also, we see that the relationship is not linear, so when we reach a finite number of images, the ability of the model **will not improve** that much. And it shows a **limitation to this pairing method** for the given problem.
