**Summary**

This article purpose is to take a look and review some machine learning algorithms that is usually be used in machine learning to create ML model project then focus on why and how those ML algorithms are used in image classification in MNIST database.

Keywords: KNN (K-Nearest Neighbors) algorithm, SVM (Support Vector Machine), Image Classification, MNIST Data, Euclidean distance, Cross Validation.

**Introduction**

Nowadays, technology especially computers sector has a great development speed or rather call really advance stage. People can just use their camera on their phone to translate non-mother language to their mother language instantly, if a student who study foreign language sometimes they don’t know a foreign word or characters, they’ll just need to open their phone and write the characters or word in translate application, the application will automatically detect their handwritten then BOOM, translation and how it’s pronounced appears. Trash is everywhere. Left uncollected, it causes a major concern for environment. To clean it up efficiently, we need to focus on the places that are most affected. That is when image classification comes in handy. One last example is, Taobao online shopping platform. Users can upload a photo of goods that users desire into Taobao. Taobao will detect given image to classify what goods that contain in the provided photo, then Taobao will give a list of goods that matched user’s photo, and many other cool stuff. The given example that appears to be a really cool and magical technology is one of so-called class image classification.

The MNIST database (Modified [National Institute of Standards and Technology](https://en.wikipedia.org/wiki/National_Institute_of_Standards_and_Technology) database) is a large [database](https://en.wikipedia.org/wiki/Database) of handwritten digits that is commonly used for [training](https://en.wikipedia.org/wiki/Training_set) various [image processing](https://en.wikipedia.org/wiki/Image_processing) systems (Wikipedia, 2020).  It has various of classifiers include KNN (K-Nearest Neighbors) algorithm, SVM (Support Vector Machine), CNN (Convolutional Neural Network), Liner Classifier, Boosted Stumps, Non-Liner Classifier, Random Forest, DNN (Deep Neural Network), RMDL (Random Multimodel Deep Learning). In this article, we’ll brieflygo through every algorithm and how they implemented in MNIST database.

**KNN (K-Nearest Neighbors) algorithm**

K-nearest neighbor algorithm is one the simplest and easy to implement for classification and regression. For instant, we have two sets of data, represented by 6 squares and 5 triangles like given figure 1.1. Says we want to find out about new data green circle which class the new data belongs to square squad or triangle squad. That’s when K-NN algorithm imerge. K is value of closest datas to the new data. Let say, k=3, hence closest square=1 and triangle=3, voted by majority so the new data belongs to class triangle. But when K=5, hence closest square=3 and triangle=2 so the new data belongs to class square. How KNN work?. By the definition of KNN, test knows closest neighbors by calculate the distance between itself to all other data on dataset. Hence, the shorter distant between a train data and test data, the more likely that test data is classified as the train data. To calculate the distance between two points (between train datas and test datas), there are many formulas, including Euclidean, Mahalanobis, Manhattan, Minkowski, Chebychev, Cosine, Correlation, Hamming, Jaccard, Standardized Euclidean and Spearman distances. Using different distance measurement, produced different accuracy (V. B. Surya Prasatha, 2019). Empathy nearest neighbors k value also has impact on result accuracy. So how do we choose k value to make prediction more accurate. There is no clear definition to choose the best choice of value k to get more accuracy result due to best choice of value k depends upon a given data but it has some basic principle to choose the suited k value:

- larger values of k reduces effect of the noise on the classification, but make boundaries between classes less distinct. Due to result base on majority votes, hence larger value of k likely to get more accuracy prediction.

- choose k to be an odd number as this avoids tied votes.

KNN is a type of [instance-based learning](https://en.wikipedia.org/wiki/Instance-based_learning), or [lazy learning](https://en.wikipedia.org/wiki/Lazy_learning) because it doesn’t learn a discriminative function from the training data but “memorizes” the training dataset instead. Pros and cons of using KNN:

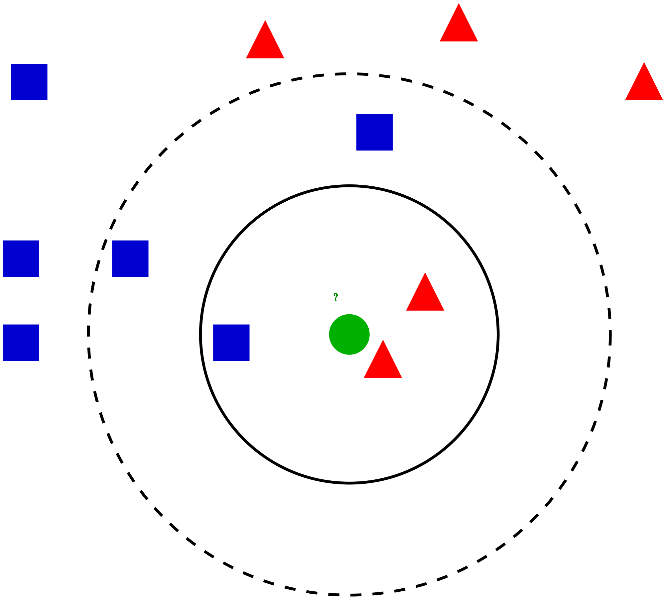
Pros:

- the algorithm is really simple and easy to implement

- no need to build model

Cons:

- the algorithm gets slower as amount of data increasing because such algorithm needs to calculate distant from test data to all data in sets

*figure 1.1*

**MNIST database:**

This term was mentioned and gave a short definition in introduction. MNIST stands for Modified [National Institute of Standards and Technology](https://en.wikipedia.org/wiki/National_Institute_of_Standards_and_Technology) database like mentioned in previous article section is a large [database](https://en.wikipedia.org/wiki/Database) of handwritten digits that is commonly used for [training](https://en.wikipedia.org/wiki/Training_set) various [image processing](https://en.wikipedia.org/wiki/Image_processing) systems. MNIST was a recreate from sample of NIST’s (national institute of standard and technology) training dataset. The original black and white images from NIST was not well suited for machine learning. The images were compressed into 28x28 pixel images by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field which each image contains a single grayscale. MNIST consists of 60,000 handwritten digit images of all numbers from 0 to 9. Each image has its corresponding label number representing the number in image. For example, in figure 1. 2, first image has a label of 4, second has labels of 1, third has label of 8, so on so forth. MNIST database contained 60000 training image and other 10000 images for testing. Haft of the training set and half of the testing set were obtained from NIST’s training set, and the other halfs were obtained from NIST’s testing set.

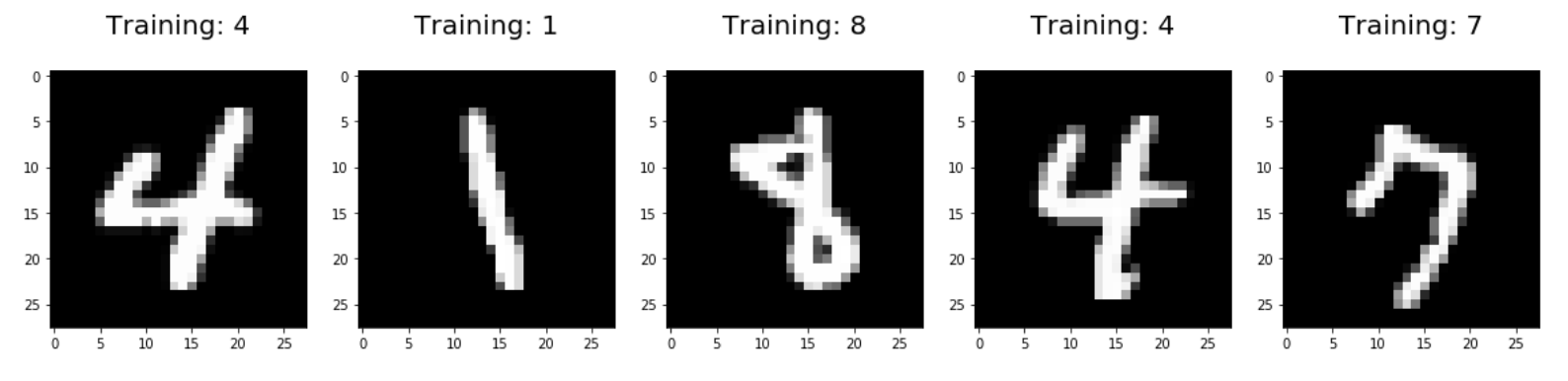


Figure 1.2

There are many algorithms to classify MNIST handwritten dataset, include Linear Classifier, Non-Linear Classifier, K-Nearest Neighbors, Boosted Stumps, SVMs, Neural Nets, And Convolutional Nets. Different algorithm has different accuracy value. Here is a chart from official website of MNIST database that dedicated each and every classification method that use to classify MNIST and their performance and in figure 1.3:

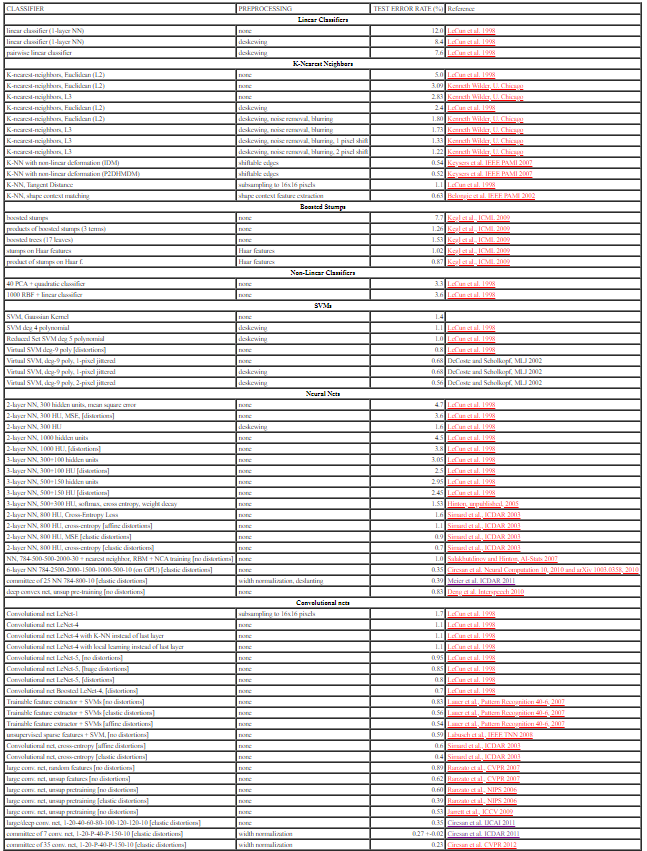


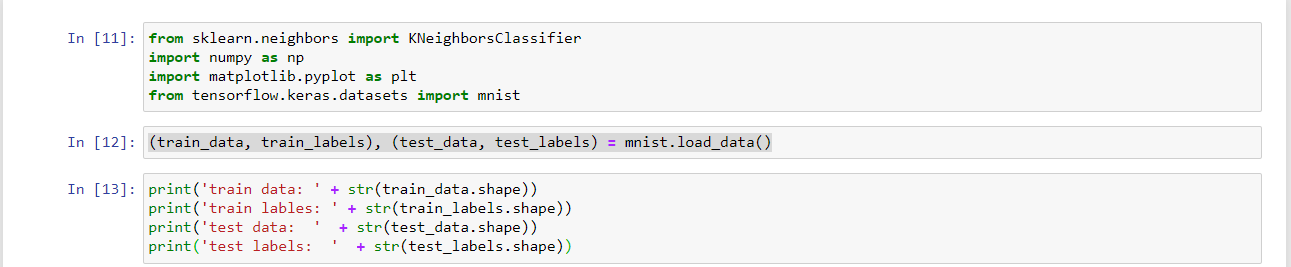
Figure 1.3

In this article will be covered in only two algorithms, KNN algorithm and SVMs (Support Vector Machine) algorithm.

**Using KNN algorithm classify MNIST database handwritten digits**

Although KNN algorithm is the simplest one among the two chosen algorithms, KNN’s classification accuracy is not bad for such a simple classification algorithm. Like mentioned, there are various ways to calculate distance between points and data, but in this section, Euclidean distance will be used. We’ll use Anaconda navigator to help use jump in to Jupyter notebook, and python programming language will be used to code.

First let’s import necessary libraries, load MNIST data then print them out. skikit



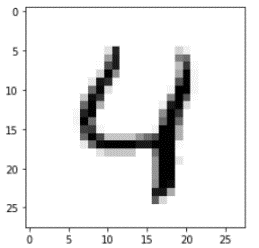
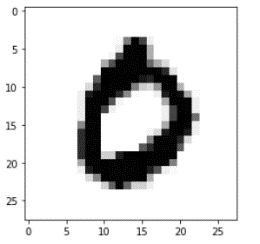
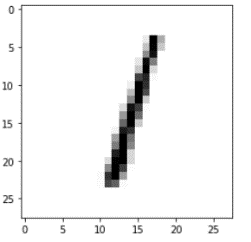
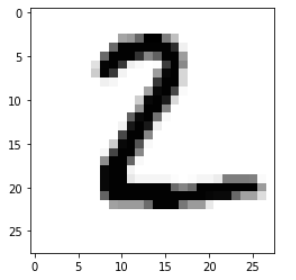
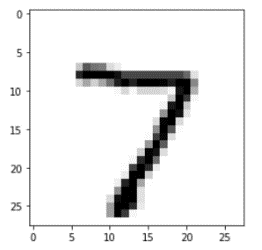
Output :



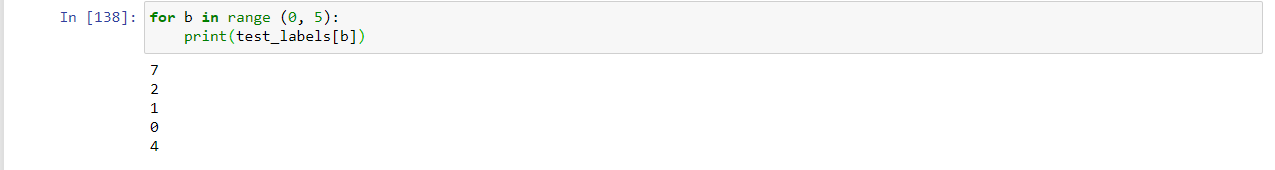
Display some MNIST handwritten images:



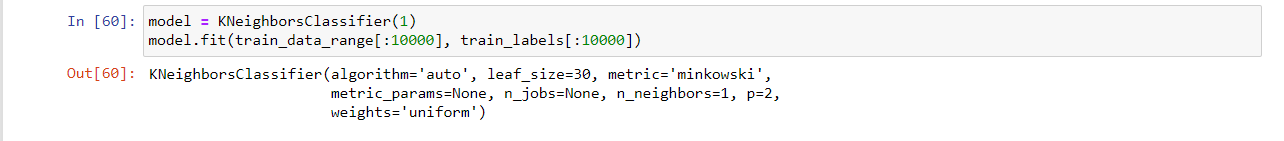
Output :



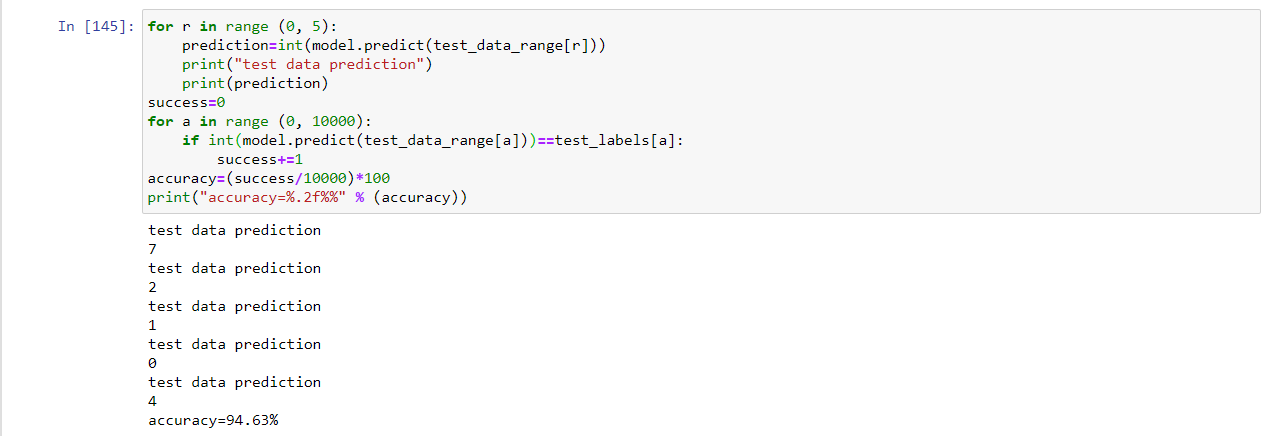
and their label :



Train our KNN model by assign train labels to corresponse train data. To shorten the amount of time to run, by using only train first 10000 of train data :



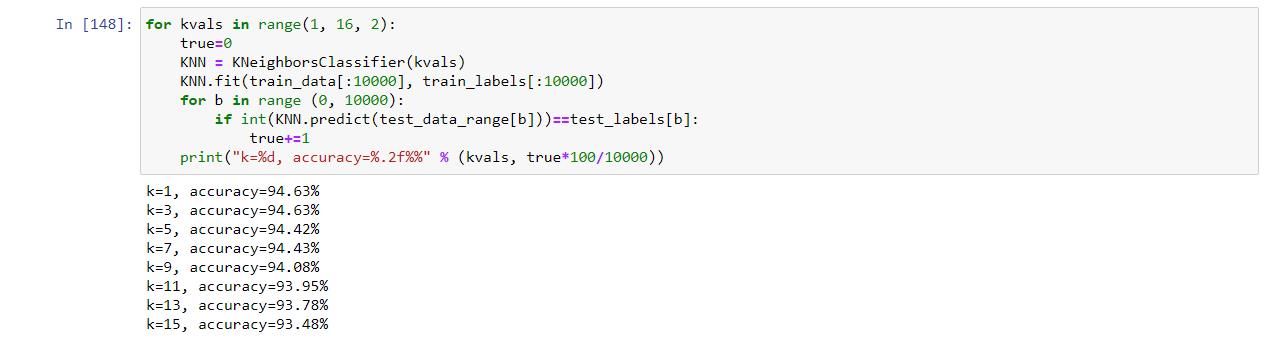
We’ll make prediction on k=3 (In [140]) the first 5 of test data and see it it match the above result or not and display how accurate our KNN model is :



Now that we see that our prediction results are matched with the result in In [4] and In [137].

To testing 10000 of test data set of 10000 training set takes aproximately 2 minutes and gives nearly 95% accurate results. Keep in mind thay accuracy and time may varied depends on the amount of trained set and testing set.

Now let see how k value impacts model’s accuracy :



We can see that k=1 and k=3 provides the best result

**Using SVMs classify MNIST database handwritten digits**

SVM (Support Vector Machine)

SVM instance of support vector machines are a set of supervised learning method used for classification, regression and outliers detection in machine learning field. SVMs goal is to classify new data to known data classes. Support vector machine view data as n-dimensional vector, then SVMs creates a (n-1)-dimensional hyperplane to separates data into classes. The distance between boundary hyperplane that separate data into classes and nearest data of any class called margin. For example, in figure 2.1, give a visualization of support vector machine

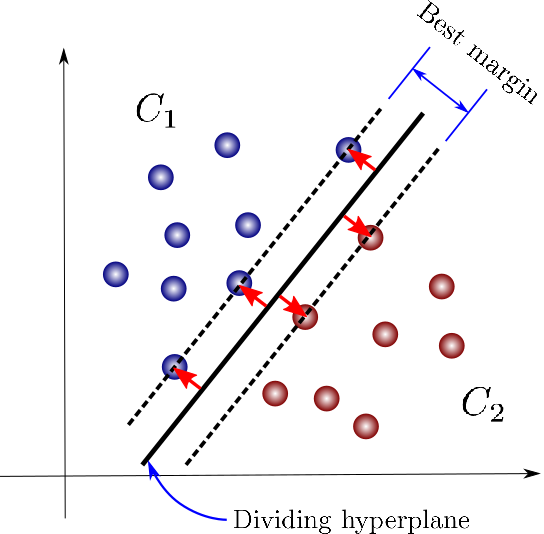


Figure 2.1

The hyperplane that makes the largest separation called maximum margin classifier. In general maximum margin classifier is the best dividing hyperplane that provides best result for data classification, hence the larger margin the better data classification. The data in figure 2.1 are called linear separable data. Is case what if data is look like figure 2.2, that is called non-linear separable data. By using formulas so-called Kernel functions, data in two dimensional space in figure 2.1 transforms into three dimensional space data so that separating hyperplane becomes easier to be constructed. So application of kernel trick is, when data cannot be separated into classes by linear separation, it maps the data into a higher dimensional space, find boundary hyperplane then performs classification. Kernel trick has couple of functions including polynomial kernel and radial basis functions kernel

- polynomial kernel

- radial basis functions kernel

** 

Figure 2.2 Figure 2.3