Supervised Machine Learning Algorithms for Classification

In the recent times, machines started to be able to learn things from outside world. And with this improvement, many algorithms and techniques found for Machine Learning purpose. In this article, we will discuss the results of 3 different techniques(which are KnearestNeighbour, NaiveBayes, and DecisionTree) with different parameters using MNIST Dataset.

Algorithms Evaluated:

KnearestNeighbour(KNN): Takes test samples, classifies each sample by looking to the class distribution of nearest k neighbours, and assigning class to a sample which is most popular in the neighbourhood.

Parameter: How many Nearest Neighbours?(k)

NaiveBayes: For each sample, NaiveBayes calculates probabilities of likelihood to each class, and assigns the label of class with highest posterior probability to the sample.

Parameter: There is no parameter, but there are 2 types of Naive Bayes algorithms which will be compared.

DecisionTree(DT): DecisionTree divides the training set until all regions in the training set has enough purity or limit has been reached.

Parameters: Max_depth.

General Parameter for all of the algorithms: test_size

Implementation Details:

Main is taking the dataset and flattening images to be able to turn data to a classifiable. After that, the program is automatically runs 3 algorithms with different parameters sequentially. Each algorithm has a class defined. In these classes, there are 2 methods: run(splitting data(train and test), fitting and predicting with regarding data using classifier) and plotMeasure(plotting confusion matrix and returning the accuracy of the method). In main method, each algorithm have been tested with parameters.

Each algorithm has run with different train-test data distribution and parameters.

If you remove comments from the lines in plotMeasure() methods, you can see the confusion matrixes.

The main method also collects the data(as array) from each run and plots graph using them.

Parameters:

K: This parameter controls the size of the neighbour set. We have to find an optimal k value for the set. Increasing/decreasing k too much may cause overfitting/underfitting, and decrease the accuracy on the test data.

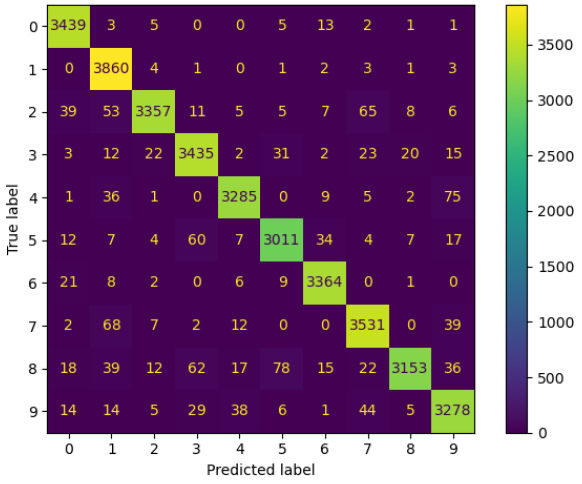
Naive Bayes Type: There are 2 types of bayes which will be used in this comparison(Multinomial Naive Bayes(MNB), and Gaussian Naive Bayes(GNB)). GNB is better in continuous values, but in discrete values(like the dataset we use) MNB makes better classification.

Test_size: Low test_size(meaning high training_size) gives better results on testing because the more data classifiers used to fit, more accurate estimations being made.

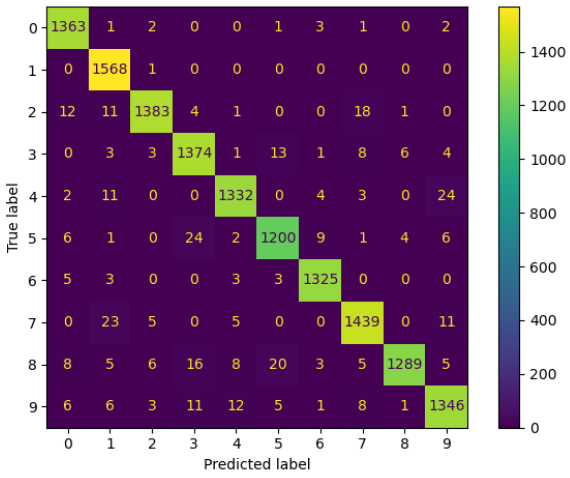
Max_depth: This parameter decides maximum split number. Taking a too small number will decrease accuracy because there is not enough split to be able to split data to gain enough information. Giving a big number may decrease accuracy a bit by dividing data based on noises after some number of iterations, also we can say it is not worth the time taken by realizing accuracy doesn't improve much after some divisions.

Results before and after parameter tuning:

KNN:

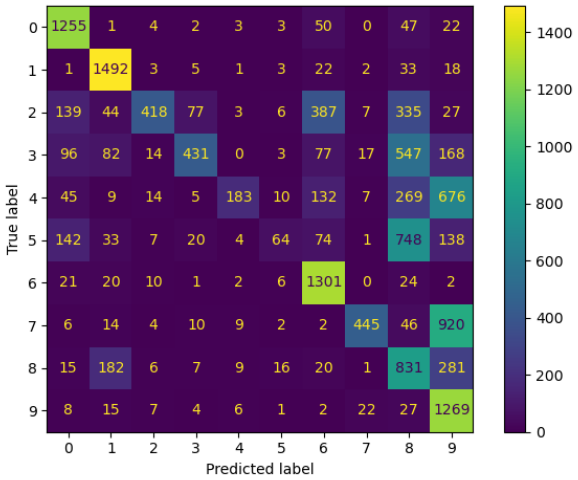


Before tuning k: Accuracy = 96.3%

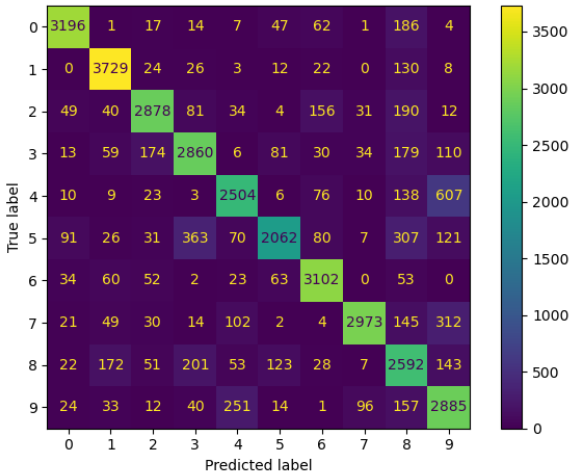


After tuning k: Accuracy = 97.3%

NB:

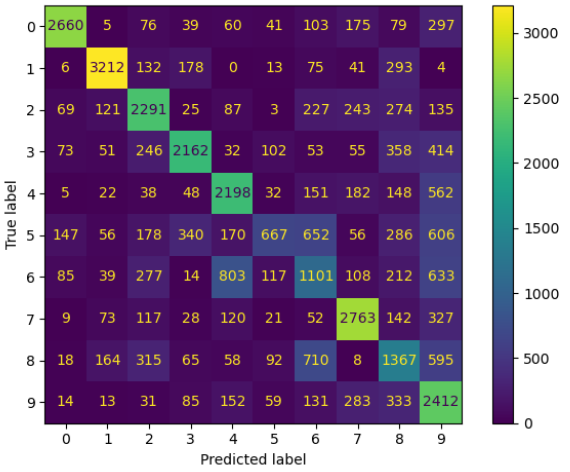


Using GaussianNB: Accuracy = 55%

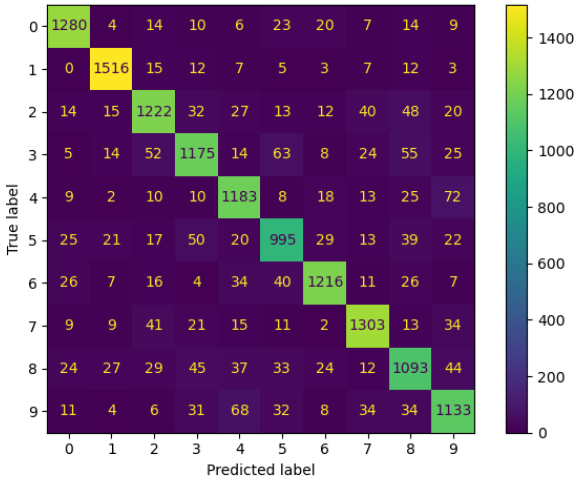


Using MultinomialNB: Accuracy = 82%

DT:



Before tuning max_depth: Accuracy = 60%



After tuning max_depth: Accuracy = 87%

Runtime(Based on dataset of 7000 digits and i5-3230m processor):

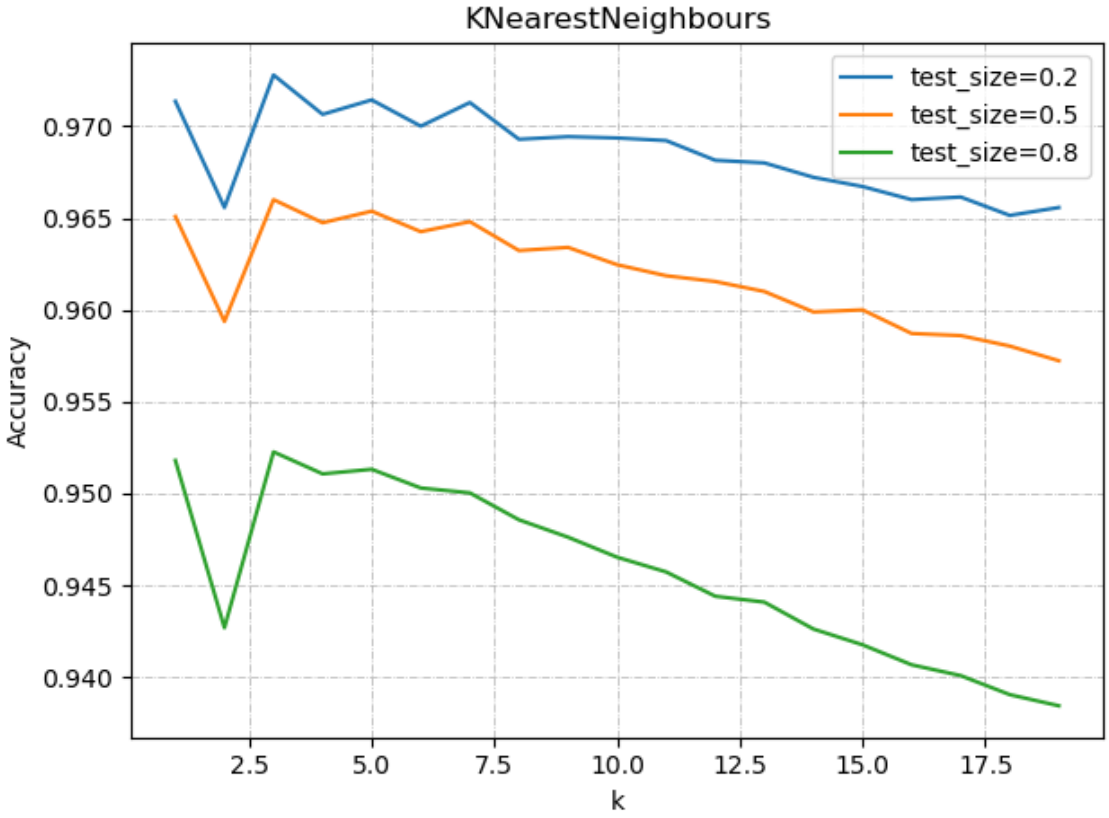
KNN is able to give precise estimation with cost of time(Average time = 19-20 minutes per prediction).

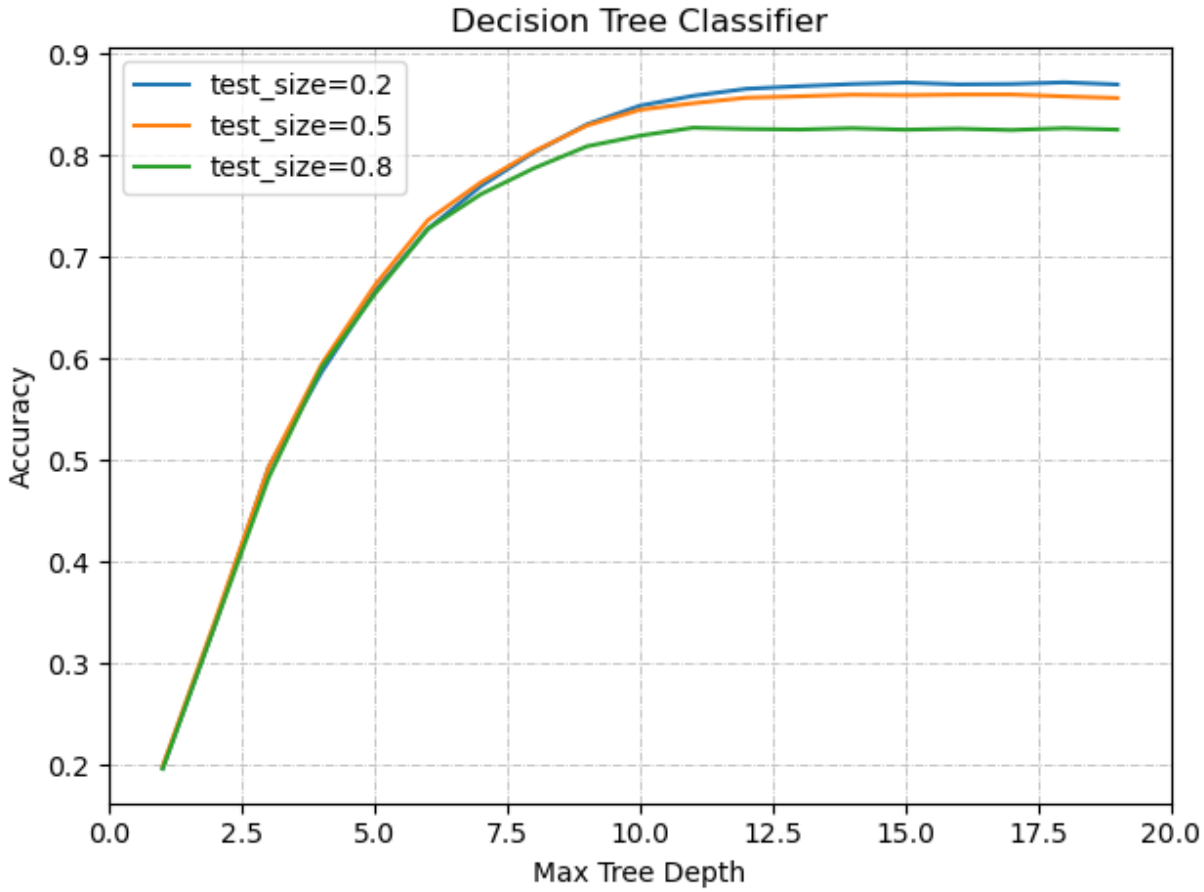
NB classifiers:

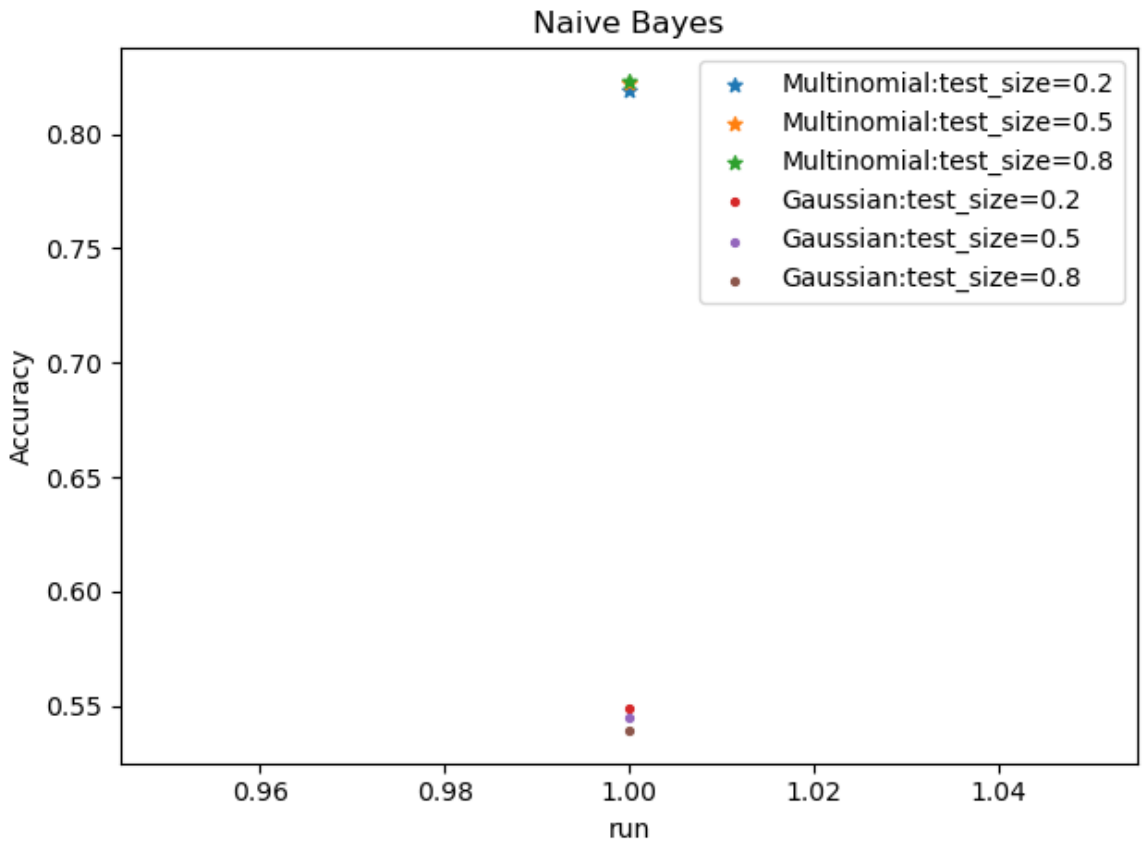
Gaussian: Takes less time(Average = ~5 seconds per prediction) but precision is too low.

Multinomial: Takes impressively less time(Average ~= 1.6 sec per prediction) and has acceptable accuracy rate.

DT is able to give nice information in this dataset by (Average~=12-13 sec. per prediction) having an accuracy rate between KNN's and NBs'


KNearestNeighbours
test_size=0.2
test_size=0.5
test_size=0.8
Accuracy
0.970
0.965
0.960
0.955
0.950
0.945
0.940
2.5
5.0
7.5
10.0
12.5
15.0
17.5
k


Decision Tree Classifier
test_size=0.2
test_size=0.5
test_size=0.8
Accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Max Tree Depth


Naive Bayes
Multinomial:test_size=0.2
Multinomial:test_size=0.5
Multinomial:test_size=0.8
Gaussian:test_size=0.2
Gaussian:test_size=0.5
Gaussian:test_size=0.8
Accuracy
0.80
0.75
0.70
0.65
0.60
0.55
0.96
0.98
1.00
1.02
1.04
run