Efficient Digit Recognition Using Machine Learning Algorithms: K-Means Clustering and K-Nearest Neighbors

Luke Caprio

University of Miami

May 7th, 2024

Introduction

In today's day and age technology is one of the most valuable and powerful industries in the world, and the recent surge of AI machine learning makes it extremely important to be educated on these topics. Machine learning is a subsection of artificial intelligence that allows computers to learn from data sets and subsequently make predictions and decisions based on its analysis (IBM, 2024a). Machine learning is a broad field, filled with many different algorithms and models, each one uniquely tailored for different data sets and situations. Multiple algorithms can be combined to complete specialized tasks such as facial recognition or image compression.

One important aspect of machine learning is classification. Classification is the process of putting items or data points into categories based on their characteristics or features such as height, weight, or color. The focus of my report is to develop a machine-learning model that efficiently and accurately recognizes and classifies handwritten digits 0-9.

In the realm of computer science, the speed or efficiency of an algorithm or model is arguably the most important characteristic of a model. Although there could be a model that classifies objects with a 99% testing accuracy if it takes an extremely long time to complete the task it becomes impractical. Models with high processing speeds enhance the efficiency and scalability of the model by taking up fewer computing resources and can easily be scaled to fit large data sets. Models that are efficient and scalable are very practical in the real world where time and speed are crucial factors.

To ensure my model was both efficient and accurate at classifying, I utilized two foundational machine learning algorithms in my model: k-means clustering and k-nearest neighbor classifier. This report covers the functionality and applications of the two machine learning algorithms I employed, outlines the methodology and steps taken to create my model, and discusses the significance and implications of my results.

Literature Review

K-Means Clustering

The first algorithm implemented in my model is the k-means clustering algorithm. K-means clustering is classified as an unsupervised machine learning algorithm. An unsupervised machine learning algorithm can discover patterns or data groupings autonomously without the need of human intervention (IBM, 2024a). The goal of k-means clustering is to simplify and reduce the complexity of a data set by clustering similar data points together resulting in a set that is much more manageable.

The process of k-means clustering partitions an unlabeled data set into distinct groups called clusters, where each data point in a cluster is similar to the others (IBM, 2024). The process begins by choosing the number of cluster centers for the model, and repeatedly assigning data points to these centers based on similarity. Similarity is measured by the Euclidean Distance, a standard geometric distance formula, to see how close points are to one another. All of the data points inside a given cluster can be represented by the centroid or cluster center which is the average position of all the data points. The cluster center creates an approximate image or point based on all of the characteristics of the cluster (IBM, 2024). This means that the cluster centers are an approximate representation of all of the data in the cluster. The clusters are then updated by taking the mean of all of the data points assigned to each cluster. This process will repeat continuously until the cluster centers converge and no longer change significantly (IBM, 2024).

K-means clustering simplifies and organizes your data into distinct clusters making the data much more accessible and applicable. K-means clustering can be seen in common applications such as organizing large document libraries or image compression. In all, k-means is an essential and foundational algorithm of machine learning as it provides a way to simplify large data sets without specific labels and organize them into distinct clusters or groups based on similarity which makes the data much easier to interpret and analyze.

K-Nearest Neighbor Classifier (KNN)

The second machine learning algorithm I utilized in my model is the k-nearest neighbor classifier. Contrary to k-means clustering, KNN is a supervised machine learning algorithm that is "defined by its use of labeled datasets to train algorithms to classify data or predict outcomes accurately (IBM, 2024a)." In the case of KNN, it stores and memorizes an entire labeled data set and makes classifications by comparing new data points to existing ones in the stored data set based on proximity, which is also measured using the Euclidean distance formula. (IBM, n.d).

When given a data point, KNN looks at a predetermined number *k* amount of data points that are closest in proximity to the original data points. These data points are referred to as the nearest neighbors. The number of nearest neighbors that the model views is an important parameter that is decided before the model runs and can be optimized based on the characteristics of the data set (GeeksforGeeks, 2024). The attributes of these nearest neighbors are then used to classify new data points.

The process starts by giving the algorithm the labeled training set, a subsection of the main data set, and letting it store the data and their labels. Once the training data set has been stored, the algorithm can be used to classify new data points in a testing set. The testing set is a different data set not included in the training set that is used to test the accuracy of the algorithm's classification.

The goal of KNN is to classify a new data point based on the attributes of its nearest neighbors. Common applications of KNN include recommendation systems, like the show recommendation feature on Netflix, or medical diagnoses. KNN is a simple and adaptable algorithm that allows for the simple classification or predictions of data points (GeeksforGeeks, 2024).

Methodology

Introduction to Methodology

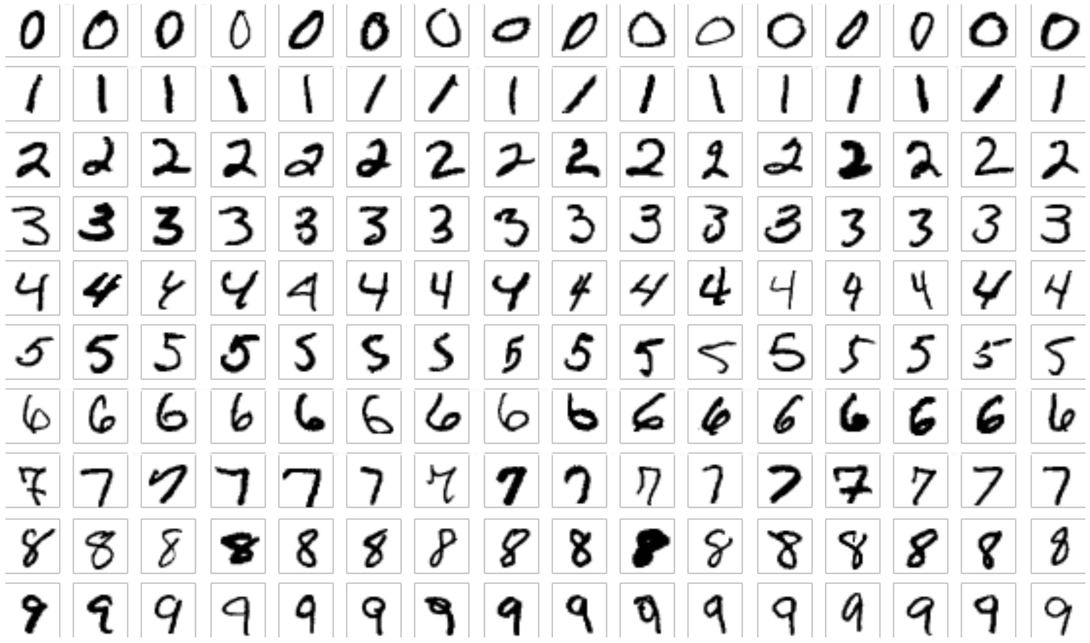
This section covers the step-by-step process taken to reach my goal of creating a model that classifies handwritten digits efficiently and effectively. To achieve this goal, I utilized both the k-means clustering and KNN machine learning algorithms. These two algorithms were chosen to make quick and accurate classifications of digits by reducing the complexity of the training set and the usage of computational power. Both algorithms were implemented in my model using public libraries Scikit-learn and TensorFlow.

Data Collection

The first component needed is a data set. For my model, I decided to use one of the most common databases in machine learning, the MNIST data set of handwritten digits. The MNIST data set contains a training set of 60,000 grayscale images of digits zero through nine along with a testing set of 10,000 grayscale images of digits of the same range. The first 60,000 images were used to train my model and the remaining 10,000 were used to test the accuracy of the classifications and evaluate the results of my model.

Figure 1

MNIST Database



[Image displays examples of the images of handwritten digits in the MNIST database] (https://upload.wikimedia.org/wikipedia/commons/f/f7/MnistExamplesModified.png)

Data Preprocessing with K-Means

Before the training data set can be inputted into the machine learning algorithms and classify the digits, it must first be preprocessed to make sure that the data set and the machine learning algorithms are compatible together. The three steps taken to preprocess the data were normalizing, reshaping, and clustering using k-means clustering.

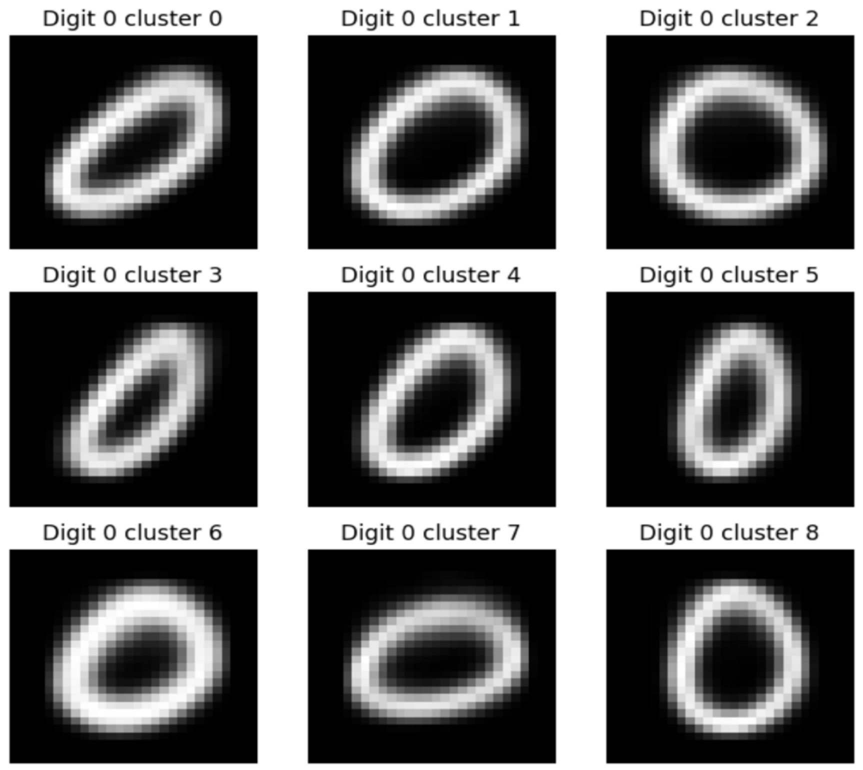
Normalization and reshaping are the simplest of the three. Since data sets can dramatically vary in size and format, the data must be in a uniform format so that it is compatible with the machine learning algorithms. Normalization transforms the data values to a standard 0-1 scale so that large data values don't dominate and possibly skew results. Reshaping the data, which involves changing the structure or format of the data without changing the data itself, transforms the data into a format that both k-Means and KNN can process.

Technically, after being both normalized and rescaled, the data set could be put into the KNN algorithm in its current state and classifications could be made, but that would cause a major issue to arise. The issue is the performance of the algorithm. Since the MNIST data set contains over 60,000 images, a KNN classifier would need to compare each of the 10,000 images in the testing set against each of the 60,000 images in the training set to find the k nearest neighbors and make classifications. This would take over 600,000,000 comparisons which would take up an unbelievably large amount of time and computing resources. The k-means clustering algorithm can be implemented to fix this problem and speed up its performance exponentially. It achieves this by simplifying the training data set.

First, the MNIST training set of 60,000-digit images is separated into ten groups based off of its label digit “0-9”, so that the ones are grouped with the ones and so on totaling up to ten groups. Now that there are ten groups, one for each digit, each group is placed through the k-means classifier algorithm. The k-means algorithm goes through all of the digits in each group and creates nine clusters based off of the features and variations of the digits. The centroids, or the cluster centers, of those clusters can serve as an approximate representation of all the common features of the images in that cluster. This effectively lowers the number of images in the group to nine approximate images for a single digit. This process is repeated for the remaining nine-digit groups resulting in 90 total images, nine approximate images for each digit. These 90 images are essentially an approximate representation of the 60,000-image training set. Therefore, k-means has simplified the training set from over 60,000 images to just 90 images.

Figure 2

Approximate Image Set Zero



[The 9 approximate images created for the digit 0after using k-means clustering was applied]

Training KNN

Now that the training data set has been reduced to 90 approximate images from the original 60,000, KNN can now be used to make classifications on the test data set. I used a 1-nearest neighbor model meaning that the model looks at the one nearest neighbor in the training set. The algorithm classifies a data point based on the attributes of the point's nearest neighbor. Since the training set was reduced, the KNN model will only have to compare each of the 10,000 images in the testing set against the 90 approximate training images to find the closest one for classification. The model stores the approximate training set and classifies all of the 10,000 images in the testing set. While these images are being classified by KNN, the elapsed time is being recorded to measure the speed of the model.

Results

Model Performance

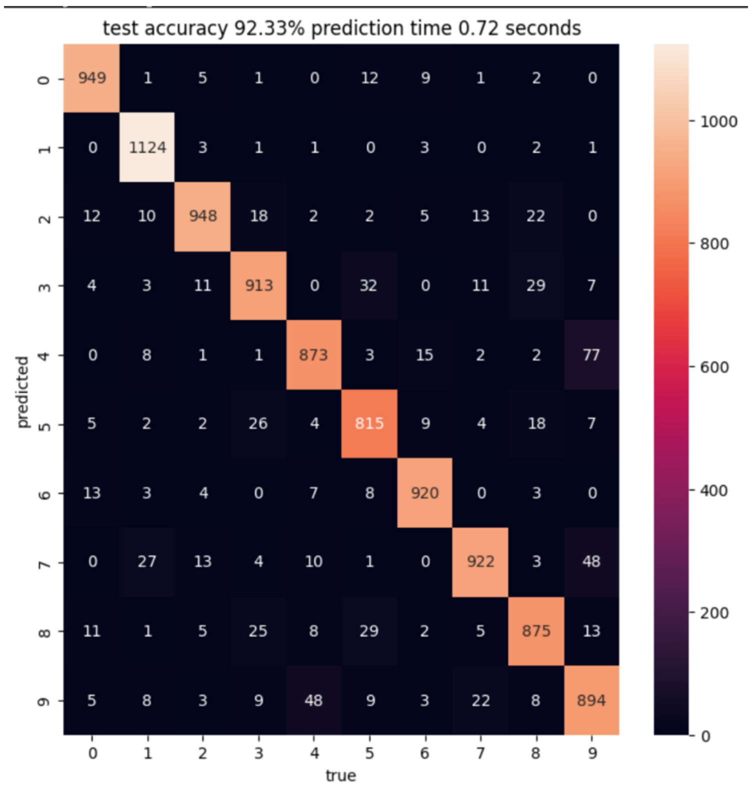
As stated previously, the goal of my research was to create an efficient and accurate model to classify handwritten digits. The key metrics used to evaluate the model were the prediction time and classification accuracy. The prediction time is the total time it took for the KNN model to classify all 10,000 images in the testing data set. The classification accuracy is the percentage of digits in the testing image set that were predicted correctly.

Since k-means reduces the complexity of an image or data set, the creation of the 90 approximate training images results in very specific details and variations from the original images being lost. This causes a small decrease in the classification accuracy when making classifications with KNN, but the trade-off is a significant gain in speed. This gain is apparent when comparing the speed of my model with a model that does not create an approximate training set with k-means and instead uses the 60,000-image training set. The prediction time for my model was 0.72 seconds. Meanwhile, the prediction time for a standard model that does not use k-means clustering to create an approximate training set is 38.27 seconds. With the help of k-means clustering my model is 98.12% faster than the standard model without k-means clustering. This metric shows that k-means clustering is a very important tool that exponentially increases the speed by reducing the complexity of the training set.

Although the model is extremely fast, it is still important that it classifies the digits with high accuracy, or else it would be impractical. The classification accuracy of my model was 92.33% meaning that out of the 10,000 images in the testing data set, my model classified 9,233 of them as the correct digit. Meanwhile, the test accuracy of a model that does not use k-means clustering was 96.91%. Although there was a 4.58% decrease in classification accuracy in my model, trading about 4% of accuracy for a model that is 98% faster is a tradeoff that is a beneficial compromise. Therefore, based on the metrics of prediction time and classification accuracy, it can be seen that my model met the goal of being extremely efficient while still being accurate.

Figure 3

Confusion Matrix for Classification Model Performance



predicted \ true	0	1	2	3	4	5	6	7	8	9
0	949	1	5	1	0	12	9	1	2	0
1	0	1124	3	1	1	0	3	0	2	1
2	12	10	948	18	2	2	5	13	22	0
3	4	3	11	913	0	32	0	11	29	7
4	0	8	1	1	873	3	15	2	2	77
5	5	2	2	26	4	815	9	4	18	7
6	13	3	4	0	7	8	920	0	3	0
7	0	27	13	4	10	1	0	922	3	48
8	11	1	5	25	8	29	2	5	875	13
9	5	8	3	9	48	9	3	22	8	894

[This confusion matrix displays the prediction accuracy and misclassifications of a classification model. Each cell shows the number of predictions made for each true class (labeled 0-9) versus the predicted class. The model achieved a test accuracy of 92.33% with a prediction time of 0.72 seconds.]

Discussion

Although I was able to achieve my main goal and create a model that can efficiently and accurately classify handwritten digits, some limitations and improvements can be made for future research.

Limitations

One limitation in my model as mentioned before is the loss of accuracy. Although the use of k-means clustering creates an approximate training image set that significantly sped up the processing speed of the model, it came with a loss of detail in the images which can explain the decrease in classification accuracy.

Improvements and Future Research

Some improvements can be made to my model for future research to enhance it. One improvement that can be made is the optimization of the hyperparameters in both of the machine-learning algorithms. "Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning (Nyuytiymbiy, 2022)." The hyperparameters used in my model were n_clusters and n_neighbors which were the number of clusters and neighbors used in their respective algorithms.

Although my model that used a 9-cluster k-means clustering algorithm produced good results, by increasing the number of clusters it can capture more variability in the data set which could potentially increase the accuracy of the KNN classification without a significant decrease in speed. The same procedure can be done for the hyperparameter n_neighbors in the KNN algorithm. Ultimately experimentation with the hyperparameters I used or even adding additional ones are improvements that could be useful for future research looking to further improve upon my model.

Future research can either take my model and improve upon the accuracy while still maintaining or even improving upon a fast processing speed, or it can apply the fundamentals or methodology of my model to other image recognition or classification tasks such as facial recognition.

Conclusion

Implications

The goal of my research was to create a model that could efficiently, and accurately classify handwritten digits using the machine learning algorithms k-means clustering and K-nearest neighbors. My model utilizing both algorithms was 98.12% faster than a basic KNN model while only sacrificing a 4.58% decrease in testing accuracy which is manageable. Therefore, based on the results, my model succeeds in being both an accurate and efficient way to classify handwritten digits. The enhanced speed of the model along with its maintained 92% accuracy allows it to be applicable in real-world and practical applications where processing speed is paramount.

In conclusion, this research provides a basic demonstration of the two foundational machine learning algorithms k-means clustering and k-nearest neighbor classifier showing their potential to enhance the processing speeds of classification opening the door for them to be applied to more complex real-world applications.

References

Caprio, L. (2024). *Source code for Efficient Digit Recognition Using Machine Learning Algorithms: K-Means Clustering and K-Nearest Neighbors*. GitHub repository. Available at: https://github.com/lukecaprio/efficienthandwrittendigitclassification.git

K-means clustering. k-means clustering. (n.d.). https://www.ibm.com/docs/en/db2oc?topic=procedures-k-means-clustering

K-Nearest Neighbor(KNN) algorithm. GeeksforGeeks. (2024, January 25). https://www.geeksforgeeks.org/k-nearest-neighbours/#

Nyuytiymbiy, K. (2022, March 28). *Parameters and hyperparameters in machine learning and Deep Learning*. Medium. https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac

What is the K-nearest neighbors algorithm?. IBM. (2024, April 22). https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20(KNN)%20al

What is machine learning (ML)?. IBM. (2024a, April 5). https://www.ibm.com/topics/machine-learning