SVMs, kNN, and Random Forest for handwriting recognition

Toni Hanrahan

# Introduction:

Much like fingerprints, no two people have identical handwriting. And even if two people are asked to simply write simple numbers by hand, they are not always spaced the same size, width, or have the same orientation on the page. The inherent similarity between digits such as 3 and 8, 1 and 7, 5 and 6, etc., compounded by the uniqueness and variety in the handwriting of different individuals, makes digit recognition quite difficult.

For many years, vast departments of human workers were required to review bank checks, government forms, applications, and many other types of financial documents in order to visually recognize numbers, and then enter them into a structured system that could be read by machines. A keyboard or number pad interface was typically required to enable this translation. This was laborious, time intensive and at times error prone.

However, improvements in image capture technology have enabled machine learning to reduce this human effort needed to recognize handwritten digits. Simple handwritten numbers are converted into an image, broken down by pixel, then stored as data that can be used to train machine learning models. Today, many of these models can achieve near-human performance of accuracy.

# Analysis and Models:

## *About the data*

The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits. It contains contains 60,000 training and 10,000 testing gray-scale images of hand-drawn digits, from zero through nine. Each image has a total of 784 pixels, and each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker.

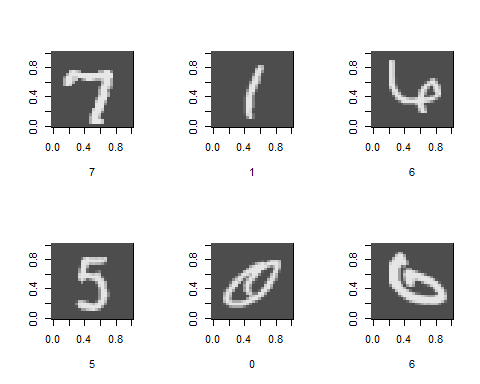
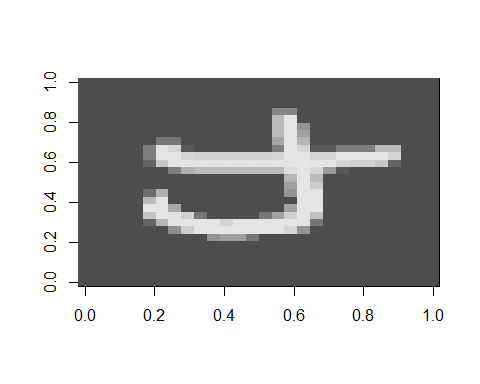
Sample of data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | pixel39 | pixel40 | pixel41 | pixel42 | pixel43 | pixel44 | pixel45 | pixel46 | pixel47 |
| 41 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 0 | 25 | 244 | 202 | 0 | 0 |
| 51 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The first column or value is the “label”, that is, the actual true digit that the handwriting is supposed to classify, such as a “7” or “9”. The remaining 784 columns represent each pixel in the image. However, the source data is extremely large, so the data used in this analysis is a smaller subset of the master database. The data was sampled by selecting every 10th record, and creating a 10% sample of 1,400 records.

## 'data.frame': 1400 obs. of 785 variables:  
## $ label : Factor w/ 10 levels "0","1","2","3",..: 8 2 7 6 1 7 2 3 10 5 ...  
## $ pixel0 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel1 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel2 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel3 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel4 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel5 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel6 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel7 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel8 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel9 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel10 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel11 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel12 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel13 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel14 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel15 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel16 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel17 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel18 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel19 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel20 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel21 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel22 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel23 : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ pixel24 : int 0 0 0 0 0 0 0 0 0 0 ...

The data can also be plotted to produce a picture of the digits.



## *About the models*

Five different models were used in this analysis in order to compare which algorithms classify the digits with the highest accuracy. And each model was run once with 3 Fold Cross Validation and again with 5 Fold Cross Validation.

Model Summary

|  |  |  |  |
| --- | --- | --- | --- |
| MODEL | DETAILS\_AND\_USAGE | X.\_CV\_FOLDS\_1st | X.\_CV\_FOLDS\_2nd |
| J48 Classifier | Creates a pruned or unpruned decision tree using the C4.5 algorithm. | 3 | 5 |
| NaÃ¯ve Bayes | Classifier based on applying Bayes’ theorem assuming independence between the factors. | 3 | 5 |
| K Nearest Neighbor (KNN) | Lazy classifier that does not have a training phase. | 3 | 5 |
| Support Vector Machine (SVM) | Supervised learning that selects the classifier to maximized margin. | 3 | 5 |
| Random Forest | Ensemble learnings that consists of a large number of individual decision trees. | 3 | 5 |

# Results

*J48 Classifier*

In this analysis, the J48 model did not use a pruned tree, the minimum number of instances per leaf was set to 2, and the confidence threshold for pruning was set to 0.5. The outcome was reasonably accurate, with 69% of the instances correctly classified.

## === 3 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 964 68.8571 %  
## Incorrectly Classified Instances 436 31.1429 %  
## Kappa statistic 0.6535  
## Mean absolute error 0.0653  
## Root mean squared error 0.2381  
## Relative absolute error 36.2896 %  
## Root relative squared error 79.3796 %  
## Total Number of Instances 1400   
##

## === 5 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1004 71.7143 %  
## Incorrectly Classified Instances 396 28.2857 %  
## Kappa statistic 0.6854  
## Mean absolute error 0.0599  
## Root mean squared error 0.2258  
## Relative absolute error 33.274 %  
## Root relative squared error 75.2964 %  
## Total Number of Instances 1400   
##

-Naïve Bayes-

In this analysis, the NB model used supervised discretization to process numeric attributes. The outcome was nearly the same level of accuracy as the J48 model, with 68% of the instances correctly classified.

## === 3 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 953 68.0714 %  
## Incorrectly Classified Instances 447 31.9286 %  
## Kappa statistic 0.6444  
## Mean absolute error 0.0636  
## Root mean squared error 0.2507  
## Relative absolute error 35.3607 %  
## Root relative squared error 83.6088 %  
## Total Number of Instances 1400   
##

## === 5 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 947 67.6429 %  
## Incorrectly Classified Instances 453 32.3571 %  
## Kappa statistic 0.6396  
## Mean absolute error 0.0649  
## Root mean squared error 0.2533  
## Relative absolute error 36.0919 %  
## Root relative squared error 84.4539 %  
## Total Number of Instances 1400   
##

-K Nearest Neighbor-

In this model, the number of nearest neighbors (k) used in classification was set to 3 and no weighting was applied. The outcome was better than the J48 and NB model, with 74% of the instances correctly classified.

## === 3 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1233 88.0714 %  
## Incorrectly Classified Instances 167 11.9286 %  
## Kappa statistic 0.8673  
## Mean absolute error 0.0315  
## Root mean squared error 0.1353  
## Relative absolute error 17.5016 %  
## Root relative squared error 45.124 %  
## Total Number of Instances 1400   
##   
##

## === 5 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1244 88.8571 %  
## Incorrectly Classified Instances 156 11.1429 %  
## Kappa statistic 0.876   
## Mean absolute error 0.0289  
## Root mean squared error 0.1304  
## Relative absolute error 16.0801 %  
## Root relative squared error 43.4954 %  
## Total Number of Instances 1400   
##   
##

-Support Vector Machine (SVM)-

In this model, the default parameters of the model were used, including leaving validation checks on. The outcome was very successful, with 88% of the instances correctly classified.

## === 3 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1235 88.2143 %  
## Incorrectly Classified Instances 165 11.7857 %  
## Kappa statistic 0.8689  
## Mean absolute error 0.1611  
## Root mean squared error 0.2737  
## Relative absolute error 89.5631 %  
## Root relative squared error 91.2597 %  
## Total Number of Instances 1400   
##   
##

## === 5 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1241 88.6429 %  
## Incorrectly Classified Instances 159 11.3571 %  
## Kappa statistic 0.8737  
## Mean absolute error 0.1611  
## Root mean squared error 0.2736  
## Relative absolute error 89.5561 %  
## Root relative squared error 91.2367 %  
## Total Number of Instances 1400   
##   
##

-Random Forest-

In this model, the number of iterations (trees) was set to 10. Too many trees could lead to overfitting the model. The outcome was very successful, with 89% of the instances correctly classified.

## === 3 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1256 89.7143 %  
## Incorrectly Classified Instances 144 10.2857 %  
## Kappa statistic 0.8856  
## Mean absolute error 0.0939  
## Root mean squared error 0.1806  
## Relative absolute error 52.226 %  
## Root relative squared error 60.2192 %  
## Total Number of Instances 1400   
##   
##

## === 5 Fold Cross Validation ===  
##   
## === Summary ===  
##   
## Correctly Classified Instances 1264 90.2857 %  
## Incorrectly Classified Instances 136 9.7143 %  
## Kappa statistic 0.892   
## Mean absolute error 0.0908  
## Root mean squared error 0.1761  
## Relative absolute error 50.4615 %  
## Root relative squared error 58.7282 %  
## Total Number of Instances 1400   
##

-Model Performance Summary-

Comparing the results of all the algorithms indicates that the Random Forest model performed with the most accuracy. And, increasing the cross validation from 3 fold to 5 fold, improved the performance of all models except Naive Bayes. The SVM model offered nearly the same level of accuracy as the RF model, so a possible approach could be to use both models together to validate the results of each.

Model Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MODEL | X3FLD\_QTY\_CORRECT | X3FLD\_PCT\_CORRECT | X5FLD\_QTY\_INCORRECT | X5FLD\_PCT\_INCORRECT |
| J48 | 964 | 68.86% | 1004 | 71.71% |
| NB | 953 | 68.07% | 947 | 67.64% |
| KNN | 1233 | 88.07% | 1244 | 88.86% |
| SVM | 1235 | 88.21% | 1241 | 88.64% |
| RF | 1256 | 89.71% | 1264 | 90.29% |

# Conclusions:

The challenge of classifying a handwritten digit based on a 28 pixel by 28 pixel black and white image is surprisingly complex. And while popular media praises machine learning as a solution that can “auto-magically” solve all of our problems, the reality is that the outcome can vary widely.

Different methods used in preparing data and varying approaches in configuring algorithms, can produce “correct” results, but lead to different outcomes. So, the challenges of handwritten digit recognition do not just result from the many different ways in which a single digit can be written, but also arise from the various different requirements to structure the data in a way that a machine can read and learn.

Because recognizing the handwritten digits of people has been an important field of progressive research for several decades, it has been studied intensely and there is a significant body of work available. This provides data scientists the opportunity to continually test and validate their own approach, driving accuracy and speed higher and higher. And as business and technology evolve, the stakes become even greater. From check processing and zip code recognition, to data entry and fraud detection, handwritten digit recognition has become a widely used technique.