Introduction:

Classifiers are often used to undergird some of the most useful features or lucrative factors in many companies today. The canonical example is filtering emails into the categories of spam or not spam. As with most things, the principals are few, but the methods are many. Machine learning classifiers are no different. The model selection must be curated to the use-case problem for the best results.

The goal of this exercise is to partake on a lesser scale, in the Kaggle Digit Recognizer competition. The classification problem at hand is attempting to successfully recognize hand-drawn digits by a user from 0-9. The original data set was too cumbersome, as such, a prorated version will be used for this exercise. The sampled data provided was bifurcated into training and testing sets and will be used within the Naïve Bayes and Decision Tree algorithms. The parameters will be tuned, and efficacy will be measured by a form of cross validation. A final decision will be made about which algorithm is superior in this instance.

Analysis and Models:

About the Data:

The data set is comprised of two csv files. The first file is “train.csv” and the second is “test.csv”. Both files contain hand drawn gray-scales images of numeric values beginning with 0 and ending at 9. Within in both files are images. Each image holds the dimensions 28x28 pixels yielding a pixel total of 784 for each image. Every pixel has a numeric association which denotes how light or dark that specific pixel is. The increasing numerical value indicates a concomitate increase in darkness. As such, lower value numbers will be lighter and higher numbered values will be darker. The pixel value range is from 0 to 255 inclusively.

The training data set (train.csv) and the testing data set (test.csv) are nearly identical. The only difference is the initial column in train.csv, “label.” Train.csv has 785 columns. Column one is the “label” column which is the digit drawn by the user. Since test.csv excludes this column, it has one less row at 784. The rest of the column in both data sets are the pixel values of the images. The pixel column names increment linearly from pixel1, pixel2, pixel3,…,pixel783. Being that the images are a 28x28 matrix the position and darkness of the pixel is correlated with the numerical image it represents. There are 42,000 observations in total.

Regarding the extent of data preprocessing, the “label” column is converted to a factor or nominal variable. The total size of the dataset is reduced to 25% of its previous size. There are no missing values to remove.

Model:

The first algorithm used in R is NaïveBayes. The algorithm is based on Bayes theorem which is used to calculate the prior and conditional probabilities of outcomes. This package is an extension of the Naïve Bayes classifier in R. The function detects the class of each feature in the data set and assumes potential different distributions for each feature. Predictors are assumed to be independent within each class label or between every pair of features. This classifier tends to be well suited for cases of document classification or the filtering of spam. A small amount of training data is comparatively needed to estimate parameters which is why 25% of the data is adequate. This result is a faster executing classifier than other options.

The second R model of choice for this analysis is the rpart model. This model is used for classification and regression decision trees. In this instance, the rpart model will be used for classification. The data is separated in, a testing set and a training set. The selected root node and internal nodes comprise the splitting attributes. A root node splits the data in half. If the root nodes lead to unanimous decision, they are regarded as pure. If the internal node subsets give inconsistent answers, they will be continually split until they attain a full decision. Multiple trees can fit the same data. There are two methods of splitting data. A two-way split creating a narrow and deep decision tree or, a multi-way split that creates a broad but shallow decision tree. The methodology is dictated by the problem and the type of variables contained in the data set. Pruning is essential for simplifying the tree by keeping only the most important splits. Accuracy is intentionally reduced to keep model usability ubiquitous and prevent overfitting.

Rpart classification accuracy may be validated in several ways. Method 1, the trained model based on the training data is used to predict the testing data. Predicted values are compared against actual values in the test data to quantify model accuracy. Method 2, a cross-validation comparing the r-square against the number of splits. This tells us which split offers the most information. Method 3, the final check is to run another cross-validation, this time relative error against number of splits. This tells how the tree should be pruned split wise.

Results:

The first classifier used is NaïveBayes. The objective is to predict the number drawn by the user. A principal component analysis (PCA) is conducted to examine if there are any variable columns that can be eliminated due to having no pixel value. Next, the dataset is converted to a data frame that only contains the variables found valid in the PCA. Below, are graphs of the PCA which show the frequencies of values for certain pixels within individual observations.

Chart

Description automatically generatedChart

Description automatically generated

For the NaïveBayes classifier, the data set is split randomly into training and testing for 25% of the overall data, approximately 10,000 observations. Below is one of a series of density plots generated that display the predictions of how many of each digit exists in a particular outcome.

Chart, histogram

Description automatically generated

The accuracy of the above results can be compared with the below confusion matrix and accuracy measures. Zeroing in on a data point from the plot, the number 7 was predicted to appear approximately 140 times. According to the confusion matrix below, it only appeared 85 times and was incorrectly predicted 41 times. This is not surprising as the as the accuracy was only around 64%. The p-value is extremely close to 0 as well.

Text

Description automatically generated with medium confidence

Next, many of these steps are repeated to create the circumstances for a decision tree model. A training and testing set of sampled data are created. Subsequently, further subsets of both those sets are created. When the model is plotted a rather unsightly diagram, shown below, is created due to the sheer number of attributes represented by each box. The spits determine whether each pixel has ink in it of a certain density.

Chart

Description automatically generated with medium confidence

The resulting efficacy of the model can be displayed below by an examination of the confusion matrix and validation measures. The accuracy of the decision tree appears to be 86% with an exactly equivalent p-value as the NaiveBayes algorithm. The kappa score is 86% as well. These statistics in their totality would imply that the decision tree was a more accurate model than the Naïve Bayes.

Graphical user interface, text

Description automatically generated

Conclusion:

The results of the amended Kaggle Digit Recognizer competition were that the decision tree classifier was a significantly more accurate classifier than Naïve Bayes. Though the p-values were the same, the accuracy of the decision tree was 86% whereas the Naïve bayes was a paltry 60%. Altering different parameters could potentially shift the balance of power amongst the algorithms. Additionally, the Naïve Bayes classifier could have shone brighter with a different problem. With clear handwriting the decision tree classifier can discern most digits however, more sloppily written numbers are easily confused with others of similar shape. Numbers like 4 and 9 or 2 and 3 were with some regularity, confused.