Sammy Pardes  
IST707  
Homework 7  
3/2/2020

**Introduction**

Online-only banking is more prevalent than ever before. The convenience is unparalleled, and the downsides are few. According to Business Insider, as of late 2019, 30% of the US population had an account with an online-only bank or planned to open one.

As fewer and fewer bank-goers find themselves tied to a physical location, the use of machine learning has become a significant part of the banking experience. Users who wish to remotely deposit checks, banks and their clientele must rely on algorithms to discern the correct amount and the validity of their checks. It is imperative that these formulas are accurate and that they can determine numeric values across many different handwriting styles. This can be done through handwriting text recognition and optical character recognition.

To compete with the ubiquity of online banks, UW Credit Union has decided to add a mobile deposit feature to their app. This will allow users to remotely submit images of their checks to be deposited. SLP Consulting has been contracted to construct a predictive model to accurately determine the numeric amounts of checks.

**Analysis and Models**

**About the Data**

UW Credit Union has collected some information to be used to conduct the analysis. They’ve gathered handwritten checks from over 2,000 over their clients and converted them into two excel files to conduct the analysis. The first file was a training data set used to create the model. The second file was used to test the model. The training file contained 1,400 observations of 785 variables, while the testing file contained 1,000 rows with identical columns.

In each data set, the first variable was called “label”. This denoted a classifying numerical value from zero to nine which users had drawn. All additional variables were pixels with decimal values ranging from 0 to 255. 0 represented the lowest brightness, while 255 represented the darkest possible color. Together, each of the 785 pixels constructed an image of the label number. In the training file, the label value was known, but in the test file the label values were replaced with question marks. There were no NAs in either file. The label variable was converted from an integer to a factor with 10 levels (0-9). All other variables remained as integers.

In order to test the accuracy of the models, a subset of the training set was separated out. This was done by determining the number of rows in 2/3 of the training set, randomizing the training set indexes, and reserving the remaining 1/3 as the training subset. This subset contained 467 observations. 933 rows remained in the training set.

For the training set, training subset, and testing set, the labels were removed and isolated in their own variables.

**Models**

With the knn() function in the class package, a K Nearest Neighbor model was created. The training set was used as the input and the training subset was used as the test data. The training set labels were used as the factor classifications. The subset labels were excluded from the KNN model.

Several values for the k parameter were tried. It was determined that k=10 yielded the most accurate results. Comparing the known training and subset digits, the KNN model generated the below confusion matrix with an accuracy of 86.7%, leaving only about 13% of the subset digits misclassified.

knnDigits 0 1 2 3 4 5 6 7 8 9

0 48 0 1 0 0 0 1 0 0 0

1 0 59 3 1 1 2 2 4 3 3

2 0 0 34 1 0 0 0 0 0 0

3 0 0 1 38 0 2 0 0 3 0

4 0 0 0 0 34 2 0 1 0 0

5 0 0 0 1 0 29 0 0 1 0

6 3 0 0 0 0 1 47 0 0 0

7 0 1 2 1 0 0 0 42 0 2

8 0 0 1 1 0 1 0 0 31 0

9 2 0 0 1 7 0 0 0 3 46

Replacing the training subset with the test data, the KNN model predicted that the test set contained more ones than any other number. The least likely digits to occur according this model were two and seven.

knnDigitsTestPred

0 1 2 3 4 5 6 7 8 9

95 148 57 105 86 78 98 100 71 95

Next, the svm() function was used to set up a support vector machine model. The label was set as the variable to be predicted and the kernel was set to polynomial. Plugging in the SVM model into the predict() function, a confusion matrix with approximately 90% accuracy was outputted. Just under 10% of the training subset data was misclassified.

digitsTrainSubsetLabels

svmDigitsPred 0 1 2 3 4 5 6 7 8 9

0 50 0 0 0 0 0 1 0 0 0

1 1 59 3 2 0 3 1 3 2 2

2 0 0 37 2 0 0 0 0 0 0

3 0 0 1 36 0 0 0 0 2 0

4 0 0 0 0 40 1 0 3 1 3

5 1 0 0 1 0 33 2 0 0 0

6 1 0 0 1 0 0 46 0 0 0

7 0 1 1 0 0 0 0 40 0 0

8 0 0 0 1 0 0 0 0 36 0

9 0 0 0 1 2 0 0 1 0 46

Using the SVM model on the test set, a similar table to the KNN model was produced. Again, the most frequent predicted digit was one and the least predicted was two.

svmDigitsTestPred

0 1 2 3 4 5 6 7 8 9

98 145 75 112 102 93 101 100 81 92

The last model generated was Random Forest. The randomForest() function took the label as the class and the training set with labels as the data. This created a model with 500 trees. The approximate error rate was 10.5% and the confusion matrix with the training subset demonstrated 90.6% accuracy.

In a similar fashion, the randomForest model predicted that the test set contained more ones and fewer twos than all other digits.

rfDigitsTestPred

0 1 2 3 4 5 6 7 8 9

106 123 76 114 104 87 105 104 91 89

**Results**

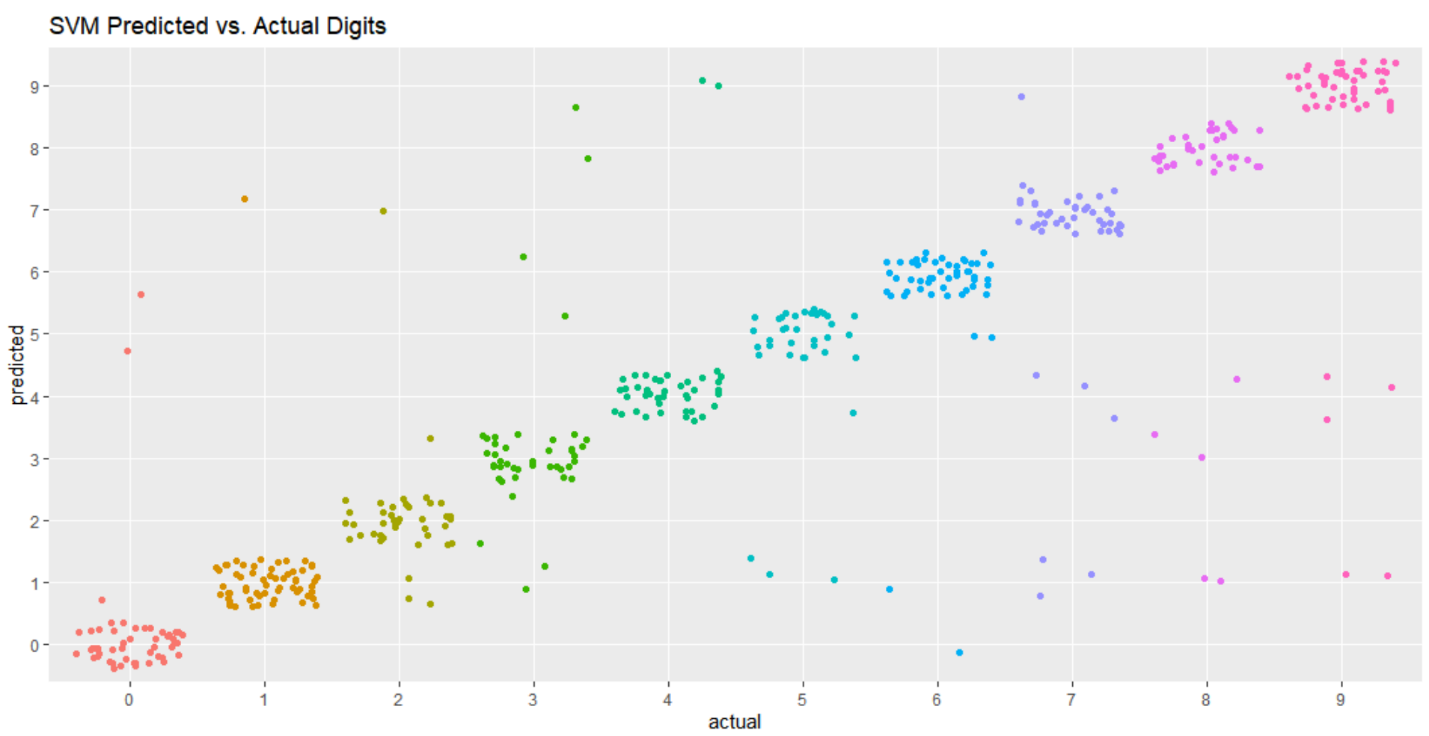
While the KNN, SVM, and Random Forest models yielded highly accurate results, there were slight differences between each of the models. The scatter plots of each model looked alike, but there are some notable distinctions.

Comparing the training set and training subset, the KNN model pretty much nailed the one, and two digit classes. The model was the most off when determining the nine and four classes. It sometimes confused nines as sevens, and fours as nines.

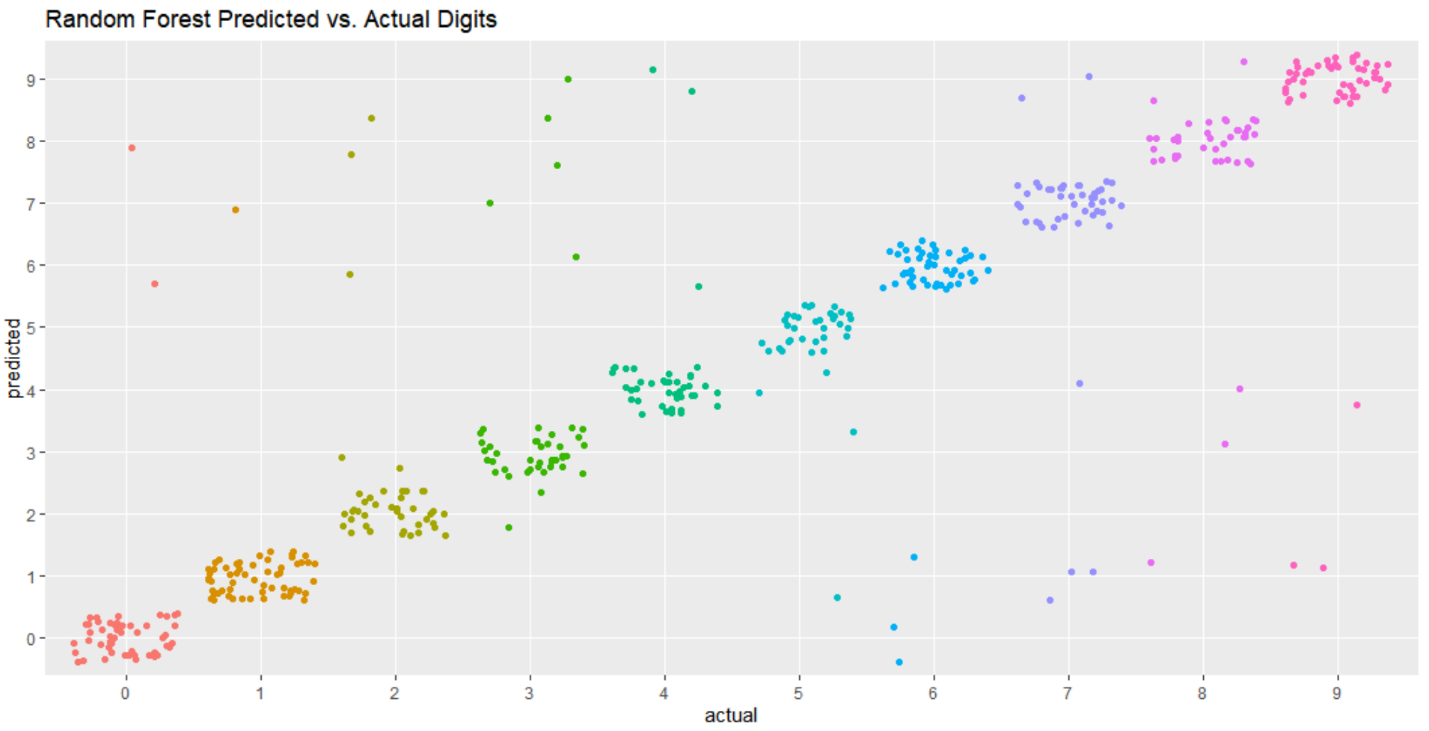
A picture containing sky

Description automatically generated

While the SVM scatter plot now looks familiar, it correctly classified more zeros than the KNN model. The SVM model was more likely than the KNN model to misidentify a nine as a four.



The plot generated by the Random Forest model was the most accurate of all three functions. Again, only about 10% of the digits from the training subset were misidentified.



Doing a side-by-side comparison of all three models, it’s easy to see that they came up with results that supported one another’s claim. Although we can’t know for certain, by viewing these models together, it can be confidently determined that in this test set of 1,000 digits, the most populous number was one. The least frequent number was two. There were approximately 100 zeros, at least 120 ones, less than 120 threes, even fewer fours and fives, a similar number of sixes and sevens, and slightly more nines than eights.

A screenshot of a cell phone

Description automatically generated

**Conclusion**

Now that SLP Consulting has done more research, they feel much more confident that UW Credit Union can use the combined models to predict the digits written on electronically deposited checks. Since the world of banking is so precise, with little to no room for error, UW Credit Union should use all three models to be more certain the predictions are correct. Additionally, it is still advised that these numbers are reviewed by a banker since these models can never be 100% exact and it’s imperative these numbers are correct.

With more fine tuning, SLP Consulting hopes to get the prediction accuracy to over 95%. In the meantime, SLP Consulting now feels that they have produced a reliable system for determining written digits using pixels. Going forward, perhaps UW Credit Union could provide higher resolution images to improve the models.

UW Credit Union will now be able to roll-out their mobile deposit feature to their users and become a strong contender against online-only banks!