Introduction: Summary and problem definition for management

Assignment 5 uses a random forest learning method for multi-class prediction of handwritten digits in the MNIST dataset. The challenge is to build a model that successfully assigns a digit that is equal to the handwritten one. As there is a slight differentiation in each handwritten digit, the challenge is for the model to accurately distinguish between each digit. The additional challenge is to build a model that runs in reasonable time, as the data set is quite large.

Assignment 5 – Random Forest Classification (MNIST)

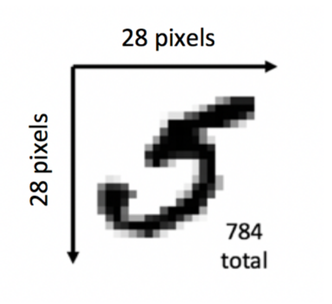
***MNIST** – Modified National Institute of Standards and Technology (example data)*

The steps for the analysis are the following:

1) Fit a random forest multi class model using each variable (784)
2) Use Principle Components Analysis (PCA) to reduce the data
3) Fit another random forest model using reduced data set
4) Compare the model timing, F1, Precision & Recall Scores

Research Design, Measurement & Statistical Methods

The data set itself is fairly straight forward. There are 70,000 handwritten digits. Each row represents one of these digits. There are 785 columns of data. 784 of them are the integer grey scale values of each pixel in a 28 x 28 pixel square. The first column is the 'response' variable, which is the actual value to test the predicted estimate against.



An example of a plotted row of data (784 pixels)

For example, the plot on the left is a binary plot showing a row of data that has a y value of '5'. The first column of data is the actual value – for training & testing. The subsequent 784 columns of data are the greyscale values for each of the 28x28 pixels representing the digit.

The next step in the process is to split the data by training & test data sets. In the instructions, we are told to use 60,000 rows for training, and 10,000 for test. I then shuffle the training data set by the index to randomize the data.

We then begin the experiment by creating a random forest model using SciKit Learn, making sure to include all 784 variables.

Model overview:

Below is a summary of the models that were created in the process of this analysis and their performance comparison. Performance variables include the reduced number of estimators as well as the F1, Precision scores of training & test sets. In the next section, I will review the build for each model and a visual comparison of each models' 'confusion' matrix.

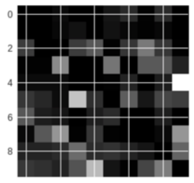
	Model 1 – all estimators	**Model 2: PCA Reduced 1 (PCA on all data)**	**Model 3: PCA Reduced 2 (PCA on train data only)**
Model Parameters & Performance	RandomForestClassifier n_estimators = 784 F1 & Precision (train/test) .96/.94	RandomForestClassifier n_estimators = 154 F1 & Precision (train/test) .94/.90	RandomForestClassifier n_estimators = 154 F1 & Precision (train/test) .94/.90
Observation:	Time to build & test: **~27 mins** High accuracy at a cost - amount of time to run the model is high.	Time to build & test: **~10 mins** Reasonable accuracy with reduced time by 33% - design flaw in PCA	Time to build & test: **~10 mins** Reasonable accuracy with reduced time by 33% - fixed design flaw in PCA.

Programming overview:

Model 1 – Random Forest using all 784 variables

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(n_estimators=784)
clf.fit(X_train, y_train)

from sklearn.model_selection import cross_val_predict
y_scores = cross_val_predict(clf, X_train, y_train)
y_test_scores = cross_val_predict(clf, X_test, y_test)
```

```
conf_mx1 = confusion_matrix(y_test, y_test_scores)
conf_mx1
```



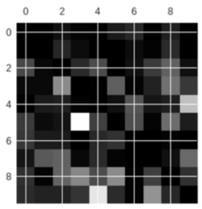
```
[ 962,    0,    2,    0,    0,    3,    6,    1,    6,    0]
[   0, 1122,    2,    3,    0,    3,    2,    1,    2,    0]
[   9,    0,  965,    9,   13,    1,    8,   18,    8,    1]
[   1,    1,   20,  940,    0,   17,    0,   13,   13,    5]
[   2,    2,    2,    0,  930,    1,    5,    0,    4,   36]
[   7, |  5,    2,   25,    6,  815,   12,    3,    9,    8]
[  12,    4,    3,    0,    4,    6,  925,    0,    4,    0]
[   2,   13,   22,    1,    8,    1,    0,  957,    2,   22]
[   5,    5,    6,   15,    6,    7,    5,    3,  908,   14]
[   8,    7,    5,   12,   27,    3,    1,   10,   18,  918]
```

Notes (Model 1): With a test accuracy of .94, the model performs very well. The reason it would not be ideal is due to the time it takes to run (~27 mins). The confusion matrix of test data shows that the most frequent mis-interpretation (highlighted) is when a 5 is misinterpreted as a 10, when a 6 is misinterpreted as a 4, or when a 8 is misinterpreted as a 3.

Model 2 – Random Forest w/ PCA reduced data

```
from sklearn.decomposition import PCA

pca = PCA()
pca.fit(mnist["data"])
cumsum = np.cumsum(pca.explained_variance_ratio_)
d = np.argmax(cumsum >= 0.95) + 1
d = 154

%%time
pca = PCA(n_components = 154)
X_reduced = pca.fit_transform(X_train)
X_test_reduced = pca.fit_transform(X_test)

clf_2 = RandomForestClassifier(n_estimators=154)
clf_2.fit(X_reduced, y_train)

y_scores_2 = cross_val_predict(clf_2, X_reduced, y_train)
y_test_scores_2 =
            cross_val_predict(clf_2, X_test_reduced, y_test)
```

```
conf_mx2 = confusion_matrix(y_test, y_test_scores_2)
conf_mx2
```



```
[ 950,    0,    4,    2,    1,    6,    9,    1,    7,    0],
[   0, 1103,   10,    4,    1,    0,    4,    2,   11,    0],
[  17,    3,  939,   10,   18,    0,    4,   14,   23,    4],
[   3,    3,   32,  900,    1,   22,    2,    8,   26,   13],
[   1,    6,    7,    2,  888,    4,   19,    5,    7,   43],
[  12,    3,    4,   52,   13,  762,   16,    2,   22,    6],
[  12,    6,    6,    0,    9,   11,  912,    0,    2,    0],
[   5,   21,   23,    3,   10,    2,    0,  935,    4,   25],
[  11,    2,   22,   30,   17,   31,    2,   14,  834,   11],
[  10,    8,    8,   12,   53,   10,    2,   34,    9,  863]])
```

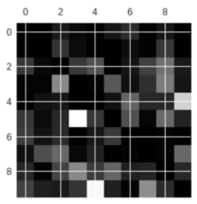
Notes (Model 2): With a test accuracy of .90, the model performs is acceptable. The model takes about 66% less time to run, so it is an improvement in terms of process. The confusion matrix of test data shows that the most frequent mis-interpretation (highlighted) is when a 5 is misinterpreted as a 10, when a 6 is misinterpreted as a 4, or a 8 is misinterpreted as a 3 or when a 10 is misinterpreted as a 5.

Model 3 – Random Forest w/ PCA reduced data

*model 2 re-run with PCA generated from 'train' data only

```
pca_2 = PCA()
pca_2.fit(X_train)
cumsum = np.cumsum(pca.explained_variance_ratio_)
d2 = np.argmax(cumsum >= 0.95) + 1

pca_3 = PCA(n_components = 154)
X_train_reduced_3 = pca_3.fit_transform(X_train)
X_test_reduced_3 = pca_3.fit_transform(X_test)

clf_3 = RandomForestClassifier(n_estimators=149)
clf_3.fit(X_train_reduced_3, y_train)

y_train_scores_3 = cross_val_predict
                    (clf_3, X_train_reduced_3, y_train)
y_test_scores_3 = cross_val_predict
                    (clf_3, X_test_reduced_3, y_test)
```

```
conf_mx3 = confusion_matrix(y_test, y_test_scores_3)
conf_mx3
```



```
[ 949,    0,    6,    3,    1,    5,   10,    1,    5,    0],
[   0, 1098,   13,    4,    0,    0,    4,    2,   14,    0],
[  13,    3,  931,   13,   19,    1,    5,   15,   26,    6],
[   2,    2,   31,  905,    2,   20,    4,   11,   28,    5],
[   1,    4,    5,    0,  882,    4,   22,    9,    9,   46],
[  11,    3,    8,   51,   11,  762,   16,    2,   21,    7],
[  13,    4,    9,    0,    5,   12,  913,    0,    2,    0],
[   4,   18,   24,    3,    8,    1,    2,  942,    2,   24],
[   9,    2,   15,   30,   13,   22,    7,    8,  859,    9],
[  15,    7,    6,   12,   56,    7,    2,   32,   10,  862]])
```

Notes (Model 3): With a test accuracy of .90, the model still performs well. The model is as fast as Model 2, and the issue of PCA being fit on 'all' data has been resolved.. The confusion matrix of test data shows Is very similar to the analysis of Model 2, so this model would be the Recommended model of the three.

Note: A shared version of the python notebook used to create this analysis can be found at Google CoLab:

https://colab.research.google.com/drive/1M7hXbshwybEKlUcf3RgrDKbCrr_NLa9U

Conclusion: Model 3 performs the best in terms of timing and precision/recall testing. It has a training score of .94 and a test score of .90, and takes 66% less time to run than Model 1. The recommended approach, therefore, is to run a PCA of the training data. This will yield a recommended number of estimators of 154 to explain 95% of the variance in the original data. We then transform the original data to only include the PCA transformation of those 154, and run another Random Forest model using the reduced data set. This will yield a model that is accurate 90% of the time, but will run in a reasonable amount of time.