Raymond Yung
Assignment 2

Custom Pipeline (Sections 1 - 10)

This section was used to understand the data. Normally, for grayscale images, the pixels range from values of 0 - 255. However, the data provided to us had values greater than 255. It might have been unnecessary, but I developed a custom pipeline, "*convert_grayscale_transform*" as a means to scale our data so that it would go back into that grayscale range. Now that the max values of my dataset are consistent, I normalized those values by dividing them all by 255 (the max value). If I had more time, I would have used the StandardScalar() instead and tried those results.

Furthermore, I had discovered that our data was already flattened into a 1-D array, and would have to be reshaped back to 28x28 if I wanted to view these images.

Classifiers:

I tested four different models before applying my ensemble method; SGD, KNN, SVC(Poly), and SVC(RGF).

Stochastic Gradient Descent

When I applied the SGD model first, I achieved around 80% accuracy for both my training and testing set. These scores were mediocre.

K - Nearest Neighbors

I then tested my results using KNN. I achieved scores around 86-88%. Upon further reading, I discovered a git repo provided from the textbook where their model achieved an accuracy score of 97% on an unmodified MNIST dataset. This is the link provided (https://github.com/ageron/handson-ml2/blob/master/03_classification.ipynb). Basing my new KNN model on their code, I used grid search to find the best parameters for KNN. Unlike the git repo, I began getting results as my K-neighbor parameter increased to 14. My new parameter led to an improved score of around 89.5%

Support Vector Machines (RBF & Poly)

I was looking for another model to try, but other classifiers led to the same score as my SGD classifier. This led me to testing out SVM models. I knew I wanted a non-linear classifier, as I had more than 2 classes. This led me to choose both the Polynomial SVM and the RBF SVM. When I ran both these models, I achieved great results: 88 - 90%, with my RBF doing the best. Furthermore, I chose a polynomial of 3 because this was defaulted.

I chose not to run a grid search on my SVM models because of the run time. It was already taking forever to run one iteration, let alone 30. Furthermore, I had to turn the probability parameter on, which would extend my run time even longer if I wanted to use a soft voting classifier as my ensemble.

Soft Voting Classifier

As stated previously, my final ensemble was the soft voting classifier. I chose my 3 best models, the two SVM's and my KNN. I did not run a grid search on these models for the same reason above. As a result, my final ensemble method led to accuracy scores of 91% or greater.

Results Condensed:

```
SVC 0.9019444444444444
Pipeline 0.8976666666666666
KNeighborsClassifier 0.8893888888888889
Pipeline 0.9129444444444444
```

```
[[4002    1   18   13    8   30   37    7   26    4]
 [   0 4611   32   12    4   13    5    7   31    4]
 [  46   93 3686   45   67   12   53   74   77   18]
 [  19   52   78 3749    3  170   17   58  104   42]
 [   4   36   25    1 3693   11   43   20   20  236]
 [  38   40   15  164   40 3288   74   18   68   50]
 [  43   35   36    1   27   51 3925    1   23    1]
 [  11   79   44   12   51   10    2 4007   13  156]
 [  21  123   50  136   21  114   43   23 3457  108]
 [  19   44   16   60  157   24    3  185   33 3623]]
```