Darrell Nelson

Big Data Analytics (IST – 736)

09/07/2019

Lab 03

Research Question

Research question: Can algorithms identify clothing items based on pixel intensity?

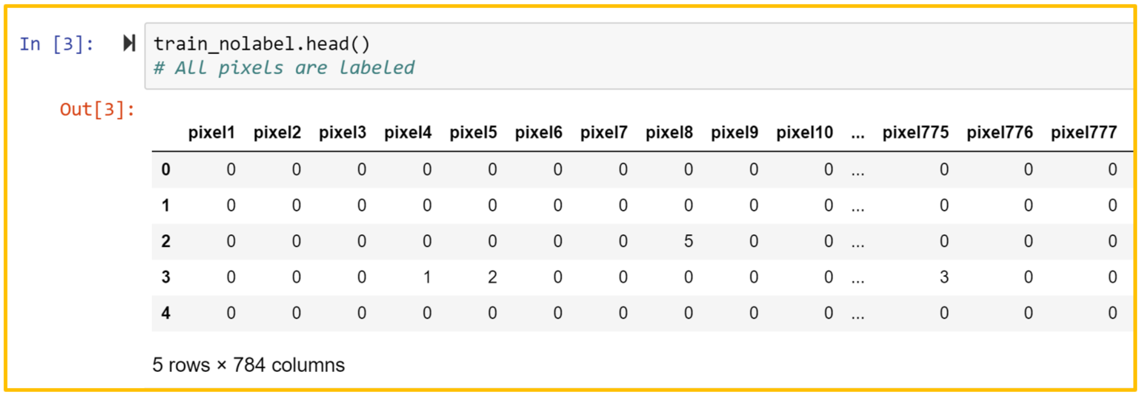
About the Data

Fashion-MNIST is a dataset of article images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. This dataset is intended to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms (Zalando, n.d.). It shares the same image size and structure of training and testing splits. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel. A higher number meaning darker intensity. The pixel-value is an integer between 0 and 255. Training and test data sets have 785 columns. The first column consists of the class label, which represents the article of clothing. The rest of the columns contain the pixel-values of the associated image. Each training and test example is assigned to one of the following labels:

Label	Clothing Item
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Data Preparation

The training and testing datasets were imported into a Python IDE (Jupyter Notebook) and their labels were removed and stored as separate variables. The shape of the training and testing set were verified, and the datasets inspected. (*Table 1.)*



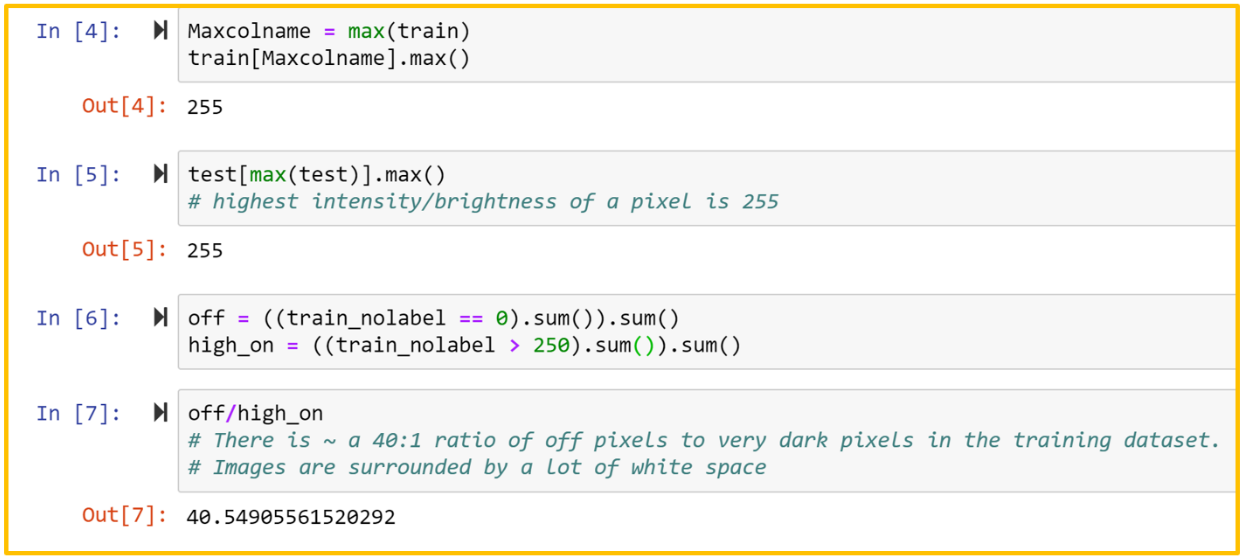
```
In [3]:  train_nolabel.head()
         # All pixels are labeled

Out[3]:
```

	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	pixel10	...	pixel775	pixel776	pixel777
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0
2	0	0	0	0	0	0	0	5	0	0	...	0	0	0
3	0	0	0	1	2	0	0	0	0	0	...	3	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0

5 rows × 784 columns

Table 1. Training Data without Label

A value of 0 means the pixel is turned off (pure white). There is ~ a 40:1 ratio of "off" pixels to very dark pixels in the training dataset (pixel intensity > 250). Meaning the images are surrounded by a lot of white space. The highest intensity of a pixel was verified to be 255.



```
In [4]:  Maxcolname = max(train)
         train[Maxcolname].max()

Out[4]:  255

In [5]:  test[max(test)].max()
         # highest intensity/brightness of a pixel is 255

Out[5]:  255

In [6]:  off = ((train_nolabel == 0).sum()).sum()
         high_on = ((train_nolabel > 250).sum()).sum()

In [7]:  off/high_on
         # There is ~ a 40:1 ratio of off pixels to very dark pixels in the training dataset.
         # Images are surrounded by a lot of white space

Out[7]:  40.54905561520292
```

Code Snippet 1. Pixel Intensity and Ratio

The dataset is perfectly balanced with all class labels having the same number of samples in the training and testing set.

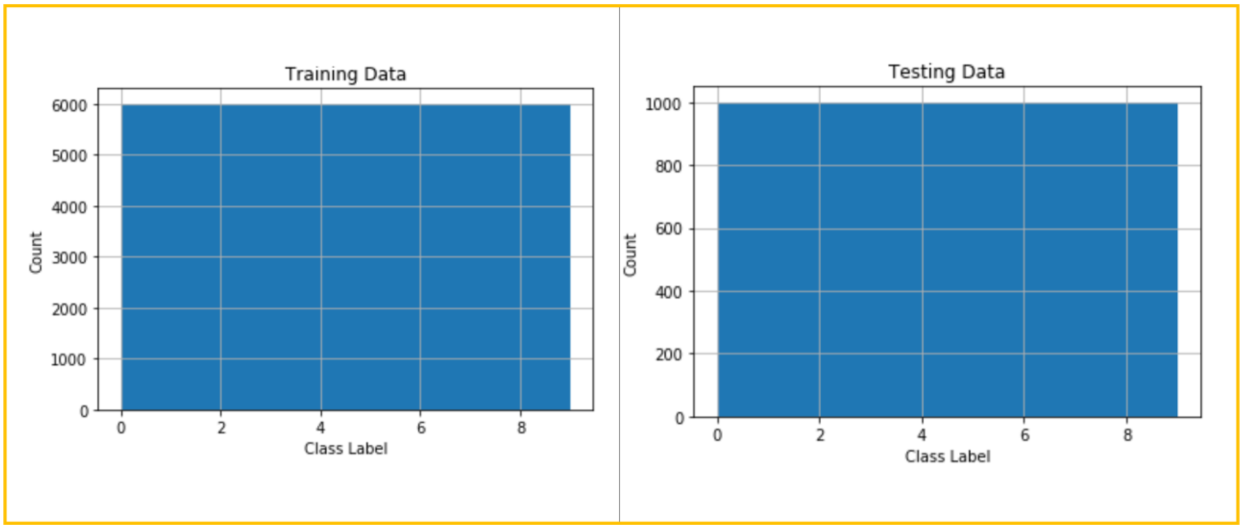


Figure 1. Sample Counts vs Class Label

Results

Linear Support Vector Classifier

A Linear Support Vector Classifier (SVC) is a linear boundary support vector machine (SVM). SVMs are a classification technique that is used to run supervised machine learning. It has its roots in statistical learning theory and has shown promising empirical results in many practical applications, including handwritten digit recognition and sentiment analysis. SVMs also work well with high-dimensional data and avoid the curse of [the] dimensionality problem. Another unique aspect of this approach is that it represents the decision boundary using a subset of the training examples, known as the support vectors.[1] The decision boundary is chosen by maximizing the distance between datapoints within a cluster and the boundary itself without sacrificing accuracy. This increases the chance of properly labeling unknown test data that may reside near the boundaries.

Model optimization was achieved by changing the parameter "dual" from True to False which is prescribed if n_samples > n_features. The "max_iter" parameter were also increased to 3,000 due to failed convergence issues at lower iterations. The overall training accuracy of the SVC ~87%:

[1] (Tan, Steinbach, & Kumar, 2006)

```
Overall training accuracy of Linear SVC:  0.8727833333333334
```

Confusion Matrix

```
[[814  10  21  48   2   3  84   0  18   0]
 [  4 970   5  12   2   2   4   1   0   0]
 [ 19   7 768  12 120   0  63   0  11   0]
 [ 26  26  15 873  27   2  24   3   4   0]
 [  1   3  69  29 812   0  83   0   3   0]
 [  2   3   0   1   1 894   0  58  10  31]
 [167  12 111  46  89   0 556   0  19   0]
 [  0   1   0   0   0  30   0 923   1  45]
 [  6   4   6  13   2  14  20  10 920   5]
 [  0   0   0   0   1  19   0  41   1 938]]
```

```
              precision    recall  f1-score   support

           0       0.78      0.81      0.80      1000
           1       0.94      0.97      0.95      1000
           2       0.77      0.77      0.77      1000
           3       0.84      0.87      0.86      1000
           4       0.77      0.81      0.79      1000
           5       0.93      0.89      0.91      1000
           6       0.67      0.56      0.61      1000
           7       0.89      0.92      0.91      1000
           8       0.93      0.92      0.93      1000
           9       0.92      0.94      0.93      1000

   micro avg       0.85      0.85      0.85     10000
   macro avg       0.84      0.85      0.84     10000
weighted avg       0.84      0.85      0.84     10000
```

Class Report 1. SVM (Precision, Recall, F1-score)

The model did very well in terms of class precision. Precision calculates what percentage of a given predicted class is actually correct. With many classes having 77% - 96% of all the predicted values in each class being correct; class 6 (shirt) was the outlier with a precision of 67%. Concluding that there must be less distinct patterns in the class 6 samples. Shirts generally consist of rectangular shapes; a long one for the torso and two tilted at a 45 degree angle for each sleeve. Such a generic identity leads to issues when trying to find distinct patterns from pixel to pixel. Recall appears to be the best metric to perceive model performance in this experiment. Recall calculates what percentage of the actual class was correctly predicted. This metric penalizes the model for guessing and gives an accurate description of how confident the model is in choosing a distinct class in a line-up. F1-Score is a type of average that equally weights the importance of precision and recall and outputs it as a single number.

Micro- and macro-averages compute slightly different averages, and thus their interpretations differ. Macro-average computes the average independently for each class and then takes the average (treating all classes equally), whereas a micro-average will aggregate the contributions of all classes to compute the average metric. In a balanced multi-class classification such as this, macro-average works well. Therefore, the main grading criteria for model comparison is the macro-average recall metric which is at 85% for the SVC model.

Neural Network

Neural Networks (NN) are choice models that use logistic regression to represent nonlinear (multi-dimensional) behavior. They are named “neural” because they mimic the way the human brain operates. NNs can adapt to changing input; the network generates the best possible result without needing to redesign the output criterial. A neural network contains layers of interconnected nodes. Each node is a perceptron and resembles a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. Hidden layers fine-tune the input weightings until the neural network’s margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis.

The model was initially run with 2 hidden layers with 400 nodes each, but the structure was too complex for this dataset and yielded an overall accuracy of 10%. The baseline parameters set by the python library sci-kit learn, were then used to tune the model and the accuracy jumped up to ~90%.

```
Overall training accuracy of Neural Network:  0.9017
```

Confusion Matrix

```
[[736   6  17  40   0   2 192   1   6   0]
 [  4 980   0  11   1   0   4   0   0   0]
 [  3   3 739  12 131   0 109   0   3   0]
 [ 24  17   8 896  21   0  32   0   2   0]
 [  0   2  49  35 833   0  81   0   0   0]
 [  0   0   0   2   0 919   2  50   1  26]
 [113   9  69  34  84   0 681   0  10   0]
 [  0   0   0   1   0  12   0 930   1  56]
 [  3   0   4   5   5   4  43   3 932   1]
 [  0   0   0   1   0   2   0  30   1 966]]
```

```
              precision    recall  f1-score   support

           0       0.83      0.74      0.78      1000
           1       0.96      0.98      0.97      1000
           2       0.83      0.74      0.78      1000
           3       0.86      0.90      0.88      1000
           4       0.77      0.83      0.80      1000
           5       0.98      0.92      0.95      1000
           6       0.60      0.68      0.64      1000
           7       0.92      0.93      0.92      1000
           8       0.97      0.93      0.95      1000
           9       0.92      0.97      0.94      1000

   micro avg       0.86      0.86      0.86     10000
   macro avg       0.87      0.86      0.86     10000
weighted avg       0.87      0.86      0.86     10000
```

Class Report 2. NN (Precision, Recall, F1-score)

Class 6 (shirt) topic was again the hardest to tune with a precision of 60%. The rest of the classes ranged from 77% - 98%. The recall macro-average was 86% which is a 1 % increase from the previous model.

Questions

1. What is the accuracy of each method?
    a. SVC overall accuracy: 87%
    b. NN overall accuracy: 90%
2. What are the trade-offs of each approach?
    a. The Linear SVC is more computationally intense and the parameters must be tuned to account for multi-output classification. The Neural Network is faster, but fine tuning the hidden nodes appears to be a lot harder than expected. With a [400 x 2] hidden layer structure only producing a 10% accuracy while a [100 x 1] hidden layer structure resulted in a 90% accuracy.
3. What is the compute performance of each approach?
    a. Linear SVC ~15 minutes
    b. NN ~ 3 minutes