Machine Learning Engineer Nanodegree

Capstone Proposal

Tom Martin
21st December 2016

Proposal

Domain Background

My project will examine the MNIST database of handwritten digits. This is a very well known dataset having attracted a great deal of academic attention since its inception. More broadly, the analyis of automated handwriting recognition has applications for fields where it is important to quickly and securely process handwritten documents at scale. For instance, this can be useful in processing historical documents, input to handheld devices via a stylus or pen, or determining authorship of incriminating documents.

Over the years, it has proved fruitful territory for examining a range of machine learning classifiers, such as linear classifiers[1], svm[2], k-nearest neighbours[3], and a range of neural network implementations[4]. There have therefore been a number of different approaches shown to be suitable to classify the dataset correctly. A paper by Hartwick[5] is particularly relevant for this projet as he has provided a clear analysis of the dataset without any further preprocessing with an SVM classifier. For these reasons, I will focus on this paper later on in this proposal.

Problem Statement

The capstone should attempt to train and tune a classifier that is able to correctly determine the number intended from the supplied image of a handwritten sample. The model produced will be trained, tested and validated against the supplied dataset. The success of the classifier will be measured using the Scikit-Learn metric's module `metrics.classification_report` and `metrics.confusion_matrix` functions, in particular I will focus on the recall ratio, which gives the error rate of the classifier: In this the case the proportion of wrongly classified images.

Datasets and Inputs

The MNIST dataset contatins 70000 samples of handwritten digits, labelled from 0 to 9. These are split into subsamples of 60000 and 10000 for training and testing respectively. The samples themselve have been centred and normalised to a grid size of 28-by-28 pixels, with each training entry composed of 784 features, corresponding the greyscale level for each pixel. The MNIST dataset in this case will be the MNIST original[6] dataset obtained via the mldata repository using SciKit-Learn's `datasets.fetch_mldata` method. A sample of the dataset is given below.

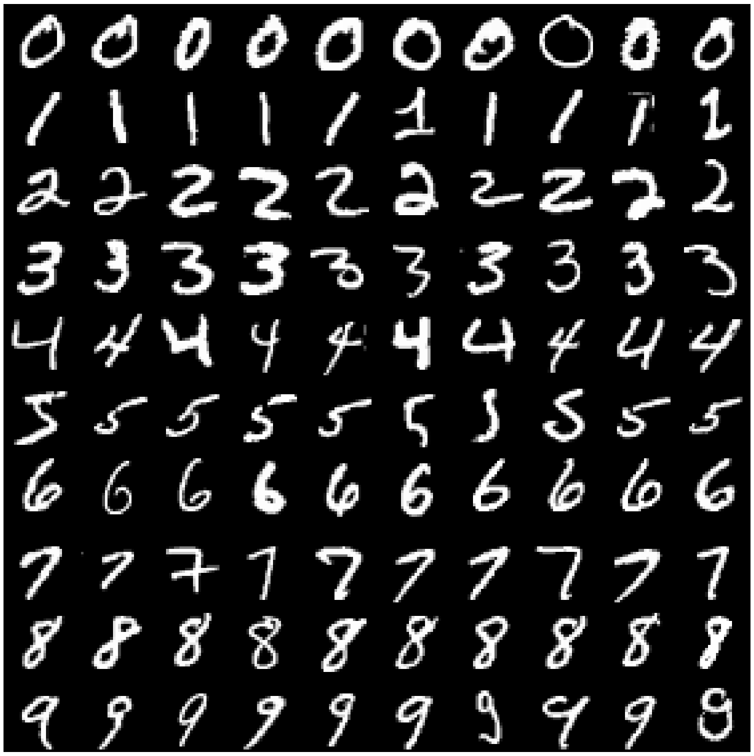

Figure 1: Sample of MNIST Dataset

The class labels in the testing set are roughly uniformly distributed, with the number of occurences of each label ranging from around 6300 and 7900. The distribution is shown graphically below. This distribution of labels means that no special sampling needs to take place to train and test correctly.

On a historical note, this dataset is the result of subsampling the original NIST dataset so that is was overall more consistent, and more suitable for machine learning: mixing together the original training and testing sets. The samples were collected from a combination of American Census Bureau employees and American high school students. This dataset contains both training and testing samples, so no further data is needed to evaluate the classifier.

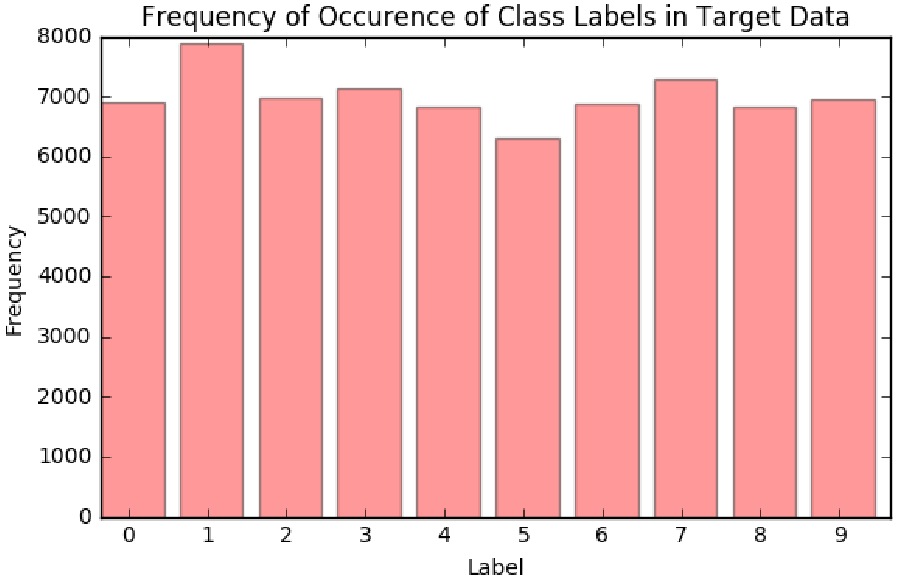

Figure 2: Frequency of Occurence of Class Labels

Solution Statement

I propose using a SVM classifier to train a solution that, with a reasonable level of accuracy, correctly maps a handwritten sample to the correct digit. A supervised classifier should be an appropriate solution to the problem as we have training data. There are also a number of academic studies that have had success with SVM classifiers[2]. Before building the model, I will use principal component analyis (PCA) with dimension reduction. PCA is used to reduce the feature space of the training data to reduce the overall training and testing time, as well as making it easier to graphically illustrate - can produce two-dimensional plot of classifying boundaries. This is especially useful for a dataset both as large and feature-rich as the MNIST dataset. It has been shown that in general PCA does not negatively impact the accuracy of a classifier, and has even been shown to boost the accuracy of the SVM classfier[7].

The trained classifier can be evaluated using a confusion matrix, and derived metrics such as f1 score, to determine its degree of success. To evaluate the trained model thoroughly, k-fold cross validation will be used to get a representative performance score of the model. Furthermore, we can consider a number of previous models[8] of the dataset using a SVM classifier, which have accuracies around 99%.

Benchmark Model

This dataset is very well studied and as such, there are many comparable studies to check against. For the project, I will make direct comparison to the paper referenced above by Hartwick[5]. This paper produces results for a SVM classifier with Guassian kernel, with parameters,

```
C = 10^6
gamma = (1/len(features)) * 10^-3.5 (approx. 4 * 10^-7)
```

The model in this paper achieves a error of around 1.4% against the MNIST testing set. Given all these results and the availability of the identical testing set, a direct comparison with this paper's results is possible.

Evaluation Metrics

The evaluation metrics for this model will be the confusion matrix, precison, recall, and error rate. The latter is to enable a direct comparison with the benchmark, which calculates this value in the paper. We noted above that the error rate can be derived from the recall rate. A supervised classfier such as SVM has known labelled data, so we can determine the number of true positives, false positives, and false negatives these are ultimately derived from the confusion matrix. To be clear, these terms are defined as follows:

- True positives: Entries that are correctly labelled
- False positives: Entries that a wrongly identified with a given label
- False negatives: Entries for a given label that are wrongly identified with other labels

In the general case, a confusion matrix is simply a matrix illustrating the mapping from the true labels to the predicted labels. Elements along the diagonal represent a correct classification, whereas the off-diagonal represent a misclassification. A confusion matrix can be a useful check to see what digits in particular are most likely confused for one another. From here we can derive[9] both the recall and precision.

Precision is the result of the number of true positives divided by the sum of true positives and false negatives. This can be given by the following equation,

```
precision = tp / (tp + fp)
```

where tp and fp stand for true positive and false positive respectively.

Recall is the result of dividing the true positives by the sum of true positives with false negatives. This can be given as follows,

```
recall = tp / (tp + fn)
```

where tp and fn stand for true positive and false negative respectively. From this, the error rate or rate at which the classifier wrongly classifies the samples can be derived,

```
error rate = 1 - (tp / (tp + fn)) = fn / (tp + fn)
```

which is is also known as the false negative rate. The error rate per digit could be derived similarly, but on a digit-by-digit basis, as given by the confusion matrix above.

These metrics altogether will give us a means to determine how well the classifier correctly labels the digits as well the error rate per digit. The error rate as well as all the other metrics discussed in this section will be calculated using the SciKit-Learn `metrics.classification_report` and `metrics.confusion_matrix` methods.

Project Design

This project will follow a typical machine workflow, starting from the dataset acquisition, then preprocessing, model generation, and then an evaluation and optimisation process.Preprocessing will be done using SciKit-Learn's `decomposition.PCA` method. Following the study by Lei and Govindaraju[7] I will choose to model with several different numbers of principal components between 10 and 100, as this is where they found a boosted classifier performance.

Due to the previous success of such classifiers and the wealth of related former studies, this project will use a SVM classifier. To both optimise and evaluate the classifier a test-training split will be done on both the training lables and test labels. I would like to recover some of the same results as the benchmark study i.e. SVM with Guassian kernel, as well as another kernel for comparison. Using the training and testing data, I will use grid search cross-validation during the optimisation step. Discussed at greater length above, I will derive recall, precision and the confusion matrix to determine the accuracy of the classifier. I will also use the error rate or false negative rate to both determine the accuracy of the derived models and make a direct comparison with the benchmark study.

References

[1] http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf "Gradient-Based Learning Applied to Document Recognition"

[2] https://people.eecs.berkeley.edu/~malik/cs294/decoste-scholkopf.pdf "Training Invariant Support Vector Machines"

[3] http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=E0F3BDC7642FBA1D8E2811526BD0E596?doi=10.1.1.106.3963&rep=rep1&type=pdf "Deformation Models for Image Recognition"

[4] https://www.microsoft.com/en-us/research/publication/best-practices-for-convolutional-neural-networks-applied-to-visual-document-analysis/ “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”

[5] https://cseweb.ucsd.edu/~jmcauley/cse190/reports/fa15/025.pdf “Reproducing Results of Guassian Kernel SVM classifers on the MNIST Dataset”

[6] http://mldata.org/repository/data/viewslug/mnist-original/ “MNIST (original)”

[7] https://www.researchgate.net/profile/Giovanni_Felici/publication/226795010_Feature_Selection_for_Data_Mining/links/53e413d70cf25d674e94b475.pdf#page=78 “Speeding Up Multi-class SVM Evaluation by PCA and Feature Selection”

[8] http://yann.lecun.com/exdb/mnist/ “The MNIST Database of Handwritten Digits”

[9] http://softclassval.r-forge.r-project.org/2013/2013-01-03-ChemomIntellLabSystTheorypaper.html “Validation of Soft Classification Models using Partial Class Memberships: An Extended Concept of Sensitivity & Co. applied to Grading of Astrocytoma Tissues”