Machine Learning Engineer Nanodegree

Capstone Proposal

Tom Martin
3rd 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. 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. This paper in particular by Hartwick[5] is particularly relevant for this projet as he has provide 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.recall_score` function, 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 themselved contains 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.

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.

Solution Statement

I propose using a SVM classifier to train a solution that, with a reasonable level of accuracy, correctly map 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, mentioned above, that have had success with SVM classifiers. 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[7] of the datasets 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 metric for this model will be the error rate, this 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 and false positives, these comprise quadrants of a confusion matrix. At greater length, true positives represent the number of correctly identified labels, whereas fasle negatives represent the number of incorrectly rejected labels.The recall rate is the ratio to true positives (tp) divided by the sum of true positives and false negatives (fn),

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

Intuitively this is the rate at which the classifier correctly classifies the samples. 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)
```

This is also known as the false negative rate. The error rate will be calculated using the SciKit-Learn `metrics.recall_score` method.

Project Design

This project will follow a typical machine workflow, starting from the dataset acquisition, then the model generation, and then an evaluation and optimisation process. There will be no preprocessing applied to the dataset for the two reasons. The first is that this dataset is already in a form understandable to SciKit-Learn, as it will be downloaded using SciKit-Learn `fetch_mldata` method. Secondly, to form a meaningful comparison with the benchmark study, we want a comparative dataset to begin with and in this case the particular study performed no preprocessing either. Feature engineering will not be used either as the dataset is not permitting of additional feature analysis - the features are just a pixel-by-pixel greyscale score.

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 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

[2] https://people.eecs.berkeley.edu/~malik/cs294/decoste-scholkopf.pdf

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

[4] https://www.microsoft.com/en-us/research/publication/best-practices-for-convolutional-neural-networks-applied-to-visual-document-analysis/

[5] https://cseweb.ucsd.edu/~jmcauley/cse190/reports/fa15/025.pdf

[6] http://mldata.org/repository/data/viewslug/mnist-original/

[7] http://yann.lecun.com/exdb/mnist/