Machine Learning for Applications in Computer Vision: Week 1

George Mesesan, IMAT: 03649684

1. Introduction

The object of this week's exercise is to classify handwritten digits using different classifiers: Support Vector Machines, Decision Trees and Random Forests. The dataset is MNIST Digit Recognition Dataset as provided at http://yann.lecunn.com/exdb/mnist.

I haven't modified the data in any way because all features have the same value range (0-255), so no normalisation was deemed necessary. As an implementation I used, as suggested, the scikit-learn library for Python.

2. SVM (Support Vector Machine) results

The SVM trainings were executed with smaller training set sizes than the full 60000 images, because training with the full set hasn't completed even after running for 22 hours on a 2.6 GHz Intel Core i5 machine. The best result is marked with orange.

Kernel type	Kernel parameters	Training set size	Training set correct classification	Test set correct classification
rbf	gamma=0.0	10000	1.0	0.1135
rbf	gamma=0.00001	1000	1.0	0.1028
rbf	gamma=0.01	1000	1.0	0.1028
rbf	gamma=0.1	1000	1.0	0.1028
rbf	gamma=1	1000	1.0	0.1028
rbf	gamma=10	1000	1.0	0.1028
rbf	gamma=30	1000	1.0	0.1028
rbf	gamma=3000	1000	1.0	0.1028
polynomial	degree=3	1000	1.0	0.1135
polynomial	degree=3	2000	1.0	0.1135
polynomial	degree=5	1000	1.0	0.1135
linear		1000	1.0	0.2051
linear		2000	1.0	0.2199
LinearSVC		1000	0.979	0.1697
LinearSVC		2000	0.629	0.2127
LinearSVC		10000	0.2799	0.2123

The Support Vector Machine seems to perform very poorly, but I cannot exclude a programming error. The results for the rbf kernel are no better than chance (the dataset has 10 classes, random guessing would give a correct classification of 0.1). The linear kernel implementation seems to perform best.

3. Decision Tree results

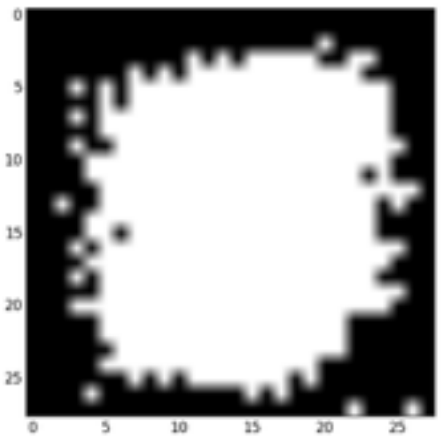
All Decision Tree trainings were performed with the full training set of 60000 images.

criterion	max_depth	Training set correct classification	Test set correct classification
gini	-	1.0	0.8559
gini	7	0.7472	0.7502
gini	12	0.9165	0.847
gini	15	0.9675	0.8509
gini	20	0.9898	0.8491
gini	30	0.9964	0.8466
gini	60	0.9998	0.8432
entropy	-	1.0	0.8525
entropy	12	0.9314	0.8556
entropy	15	0.9813	0.8595
entropy	20	0.998	0.8532

The best result was achieved with the criterion=entropy and the max_depth=15. Changing the max_features default setting brought no improvement, it actually made the classification worse.

The visualised constructed tree is in the source code as file “tree.pdf” (including it here would’ve made the report unwieldy).

The pixels used by the tree for classification are shown below:



4. Random Forest results

All Random Forest trainings are performed with the full training set of 60000 images. .

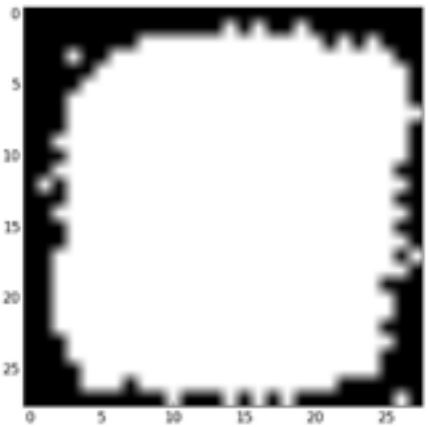
criterion	max_depth	Training set correct classification	Test set correct classification
gini	-	0.9989	0.9271
gini	7	0.8288	0.8423
gini	15	0.986	0.9254
gini	30	0.9987	0.9247
gini	60	0.9991	0.9249
entropy	-	0.9987	0.9272
entropy	7	0.8398	0.8466
entropy	15	0.9904	0.9249
entropy	30	0.9989	0.925

The best classifier uses criterion=entropy and no limit for the maximum depth.

The default number of classifiers is 10, increasing it improves classification performance, but at the cost of execution time, as seen in the table below.

criterion	n_estimators	Training set correct classification	Test set correct classification
entropy	5	0.9915	0.89
entropy	10	0.9987	0.9272
entropy	15	0.9997	0.9406
entropy	20	0.9998	0.9436
entropy	50	0.9999	0.9535

The pixels used by the random forest for classification are shown below



5. Cross Validation

A 5-fold cross-validation was performed on the DecisionTree with criterion=entropy and max_depth=15 with a result of 0.843 (+/- 0.015). For RandomForest with criterion=entropy, max_depth=15 and n_estimators=20 the result was 0.936 (+/- 0.009).

6. Conclusion

Overall, the RandomForest solution seems to give the best results with the mention that the SVM training results are suspiciously low.