Ievgeniia Dobryden

Report on SVM, Trees

Objective: recognize hand-written symbols

Dataset: MNIST database – set of 28x28 images for training and testing (60000 – training, 10000 - testing)

Methods: SVM; Decision trees, Random forests. Used scikit lib.

SVM

1. Tried RBF kernel. Without image normalization got bad results, all symbols were interpreted as 7

   numToTrain = 1000, numToTest = 1000
   Total score = 0.099

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
0.0	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*

   With image normalization ([0; 255] - > [-1; 1])

   numToTrain = 1000, numToTest = 1000

   Total score = 0.877

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
0.94	*0.99*	*0.91*	*0.77*	*0.85*	*0.87*	*0.92*	*0.85*	*0.76*	*0.87*

   Image normalization is importand because during energy minimizations data in wide range may cause numerical problems. (255 * 255 >> 1*1)

   All following tests were done on normalized images

2. Experimented with weights. In last scores we see that “3” is badly recognized, so I increased it’s weights, got:

   Total score = 0.856

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
0.93	*0.95*	*0.80*	*0.89*	*0.82*	*0.76*	*0.85*	*0.85*	*0.87*	*0.83*

   After some tuning of weights I got 0.877 score, but could not get higher
   weights = {0:1, 1:0.9, 2:5, 3:15, 4:3, 5:4, 6:4, 7:5, 8:14, 9:3}

So, with rbf kernel I got pretty nice results after training on all images and testing on all - Total score = 0.9027

3. Tried linear kernel

   Total score = 0.8758

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
0.97	*0.97*	*0.90*	*0.78*	*0.88*	*0.83*	*0.91*	*0.88*	*0.79*	*0.84*

4. Polinomial kernel:
   After tuning coef0, gamma, weights= ‘auto’, got Total score = 0.885
   Pretty same results for Sygmoid

Result: SVM with rbf kernel returned best score = 0.9027

Decision tree.

1. Tried unlimited depth:
   a) numToTrain = 2000 test all
      Total score = 0.6935

   b) numToTrain = 20000
      Total score = 0.8387

   c) train all test all
      Total score = 0.8772

   In case (a) we see that if we have unlimited depth, data is overfitted, that is why score is so low in case a. (with same training images number and depth = 10, score = 0.8217). But as soon as training images number is increased, we get better results. In case (c) tree was huge, but such nice detection rate (0,87 with full depth!!) can be explained with fact that digits in test data are pretty much similar to training data, that is why this overfitting returns ok results here. Probably in case of more variable data (e.g faces) results with be much worse with full depth.
2. Depth limit =7 -> Total score = 0.7921
3. Depth limit =10 -> Total score = 0.8586. Increasing depth did not add much improvement
4. Tuned max features number.
   #30
   Total score = 0.7918
   # 300
   Total score = 0.8452
   With increasing features score is better, because we are more likely to have important features in our final set.
   #100 features was optimal – after adding more score did not increase significally, but training time – did.
5. Criterion: Gini and Entropy returned pretty same results.

Result: Tree with depth 10, max features 100, criterion – entropy is optimal : score = 0.8687

Random forest

1. Number of trees: 10 worked best, increasing did not add significant difference in result
2. Depth: same as for decision tree, depth 10 was optimal.
3. Number of features: Since we have a lot of trees, it max number of features per each tree may be smaller, because each of trees have contribution, and even we have < 30 features, very likely that important one will be selected. After tuning max_features parameter, I found that # 30 was the best
4. Gini and Entropy criterion produced ~ same result

Result: Forest with 10 trees, max 30 features and depth 10 is optimal : score = 0.9386.