华南理工大学

South China University of Technology

The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Wu kun

Supervisor:
Mingkui Tan or Qingyao Wu

Student ID：201721045480

Grade:
Undergraduate or Graduate

December 20.2017

Face classification based on AdaBoost algorithm

Abstract—AdaBoost is best used to boost the performance of decision trees on binary classification problems. This report is to record the process of experiment and the outcome.

I. INTRODUCTION

Boosting is an ensemble technique that attempts to create a strong classifier from a number of weak classifiers. AdaBoost was the first really successful boosting algorithm developed for binary classification. It is the best starting point for understanding boosting.
This is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added.
I programed a Adaboost classifier constituted with 5 weak classifier to distinguish human face and background, the accuracy of my classifier is up to 92 percents.

II. METHODS AND THEORY

A weak classifier (decision stump) is prepared on the training data using the weighted samples. Only binary (two-class) classification problems are supported, so each decision stump makes one decision on one input variable and outputs a +1.0 or -1.0 value for the first or second class value.
The misclassification rate is calculated for the trained model. Traditionally, this is calculated as:

$$\epsilon_m = p(h_m(\mathbf{x}_i) \neq y_i) = \sum_{i=1}^{n} w_m(i)\mathbb{I}(h_m(\mathbf{x}_i) \neq y_i)$$

Where error is the misclassification rate, correct are the number of training instance predicted correctly by the model and N is the total number of training instances.
Which is the weighted sum of the misclassification rate, where w is the weight for training instance i and terror is the prediction error for training instance i which is 1 if misclassified and 0 if correctly classified.
A stage value is calculated for the trained model which provides a weighting for any predictions that the model makes. The stage value for a trained model is calculated as follows:

$$\alpha_m = \frac{1}{2}\log\frac{1-\epsilon_m}{\epsilon_m}$$

Where stage is the stage value used to weight predictions from the model, ln() is the natural logarithm and error is the misclassification error for the model. The effect of the stage weight is that more accurate models have more weight or contribution to the final prediction.

The training weights are updated giving more weight to incorrectly predicted instances, and less weight to correctly predicted instances.
For example, the weight of one training instance (w) is updated using:

$$w_{m+1}(i) = \frac{w_m(i)}{z_m}e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$$

Where w is the weight for a specific training instance,

$$z_m = \sum_{i=1}^{n} w_m(i)e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$$

exp() is the numerical constant e or Euler's number raised to a power, stage is the misclassification rate for the weak classifier and terror is the error the weak classifier made predicting the output variable for the training instance, evaluated as:
terror = 0 if(y == p), otherwise 1
Where y is the output variable for the training instance and p is the prediction from the weak learner.
This has the effect of not changing the weight if the training instance was classified correctly and making the weight slightly larger if the weak learner misclassified the instance.

III. EXPERIMENT

To train the classifier I segmented 1000 face-images and 1000 noface-images into train dataset and test dataset.
Every base classifier is a decision tree, to prevent it become overfitting, these decision tree's max depth is restricted to 3.
After series attempt, I found 5 base classifier is good enough and add other classifier will make no sence.

	precision	recall	fl-score	support
non-face	0.85	0.99	0.92	88
face	0.99	0.87	0.92	112
avg/total	0.93	0.92	0.92	200

IV. CONCLUSION

In these experiment we can see that use cost function and gradient descent approach we can train module which is able to classify different kind of data, And use optimized gradient dexcent approach will speed up the training process.