Machine Learning Competition

Stephen Carmody, Stefano Costantini, Monika Krzyzanowska

27 March 2015

Introduction

This technical report summarises the analysis that our team (**'Thy Nearest Neighbours'**) has carried out for the Machine Learning competition. The objective is to provide an overview of our approach, showing the techniques we have considered and the results that have led us make our final submission to the competition.

The report is structured as follows:

- First we describe the approach we have used to test the chosen techniques and to produce the results.
- Then we discuss our results relative to the application of the shortlisted techniques. For each technique we show the results of the testing that we used to choose the method underpinning our final submission.
- We conclude by showing our chosen technique and its results

Initial assessment and approach

Our first step to address this classification problem was to identify a short-list of tecniques that could be used. We review the existing literature and identified a longlist of possible candidates. For each of these techniques we carried out some preliminary testing to check whether they warrated further assessement.

During this initial testing phase, we decided to discard some approach that were not delivering acceptable results. For example, we tested the **Adaboost.M1** algorithm (using the `R` package `adabag`) and found the error to be too high. Specifically, the test error on a 10% holdout sample was **0.282** and the 10-fold cross validation error was **0.276.** We decided not to pursue this approach further.

[STEPHEN: COULD WE ADD SOMETHING ON NEURAL NETS]

At the end of this initial phase we identified a shortlist of techniques to explore further. These are:

- k-Nearest Neighbour rule
- Support Vector Machines [**NOT SURE WE SHOULD INCLUDE THIS, ERRORS ARE QUITE HIGH**]
- Bagging
- Random Forests

For each technique, we implemented the following testing procedure:

1. First we identified the set of parameters that we wished to change. On this basis we defined a "testing grid" to set out all possible combination of parameters.
2. We implemented the testing, parallelising the calculations over the "testing grid". We did this by means of the `foreach`, `snow` and `doSNOW` `R` packages. We trained each model on 40,000 observations and calculated the test error on a hold-out sample of 10,000 observation,

3. We summarised the results of the multiple testing in a data table and selected the specification yielding the lowest test error.

k-NN

SVM

[NOT SURE WE SHOULD INCLUDE THIS, IT DOES NOT LOOK TOO PROMISING]

Bagging

Random Forest

We have tested a Random Forest approach by considering these ranges of parameters:

- **Number of trees in each iteration:** from 100 to 1,500, with an increment of 100.
- **Number of features considered in each iteration:** from 3 to 30, with an increment of 3.

The following table provides a summary of the best results:

# of trees	# of features	Test error
1400	27	0.1153
1000	30	0.1160
1000	30	0.1161
1500	30	0.1161

The code to reproduce these results is provided in our repository on GitHub in the file `Random_Forest-ThyNearestNeighbours.`

On this basis, for this technique we selected a Random Forest approach with these parameters:

- Number of trees: 1400
- Number of features in each iteration: 27

Conclusions

The following table summarises the best result that we have achieved with each of the techiques we have tested.

Technique	Test Error
kNN	0.000
SVM	0.000
Bagging	0.000
RF	0.000

On the basis of the results above, we have selected **[ADD SELECTED APPROACH]**, using the following parameters:

- Parameter 1
- Parameter 2
- Parameter 3

We have used this approach to make our final submission to Kaggle which yielded a score of **[REPORT**

KAGGLE SCORE OF OUR PREFERRED SUBMISSION]