# Massive Online Analysis - Lab 1

Alban de Crevoisier

The purpose of this lab is to get to grips with the MOA tool. We'll test and select the best classifier on a toy dataset.

Let's start with a wide range of classifiers. For now, we'll only test them on accuracy, not cost.

Each classifier is used with its default settings:

| Classifier | Accuracy (%) |
|---|---|
| Naive Bayes | 61 |
| Perceptron | 82 |
| SGDMultiClass | 62 |
| AWE | 81 |
| MultiLabelHoeffding | 61 |
| HoeffdingAdaptiveTree | 82 |
| HoeffdingTree | 81 |
| RandomHoeffdingTree | 65 |

With saner defaults:

| Classifier | Accuracy (%) | Param |
|---|---|---|
| Perceptron | 82 | learning rate 0.9 |
| SGD | 78 | learning rate 0.1, regularization 0.1 |
| SGDMultiClass | 89 | learning rate 0.1, regularization 0.1 |

Considering these exploration results, let's select perceptron, AWE, multi-class stochastic gradient descent, Hoeffding tree and Hoeffding adaptive tree.

# 1 percepton

| learning rate | Accuracy (%) |
|---|---|
| 0.01 | 74 |
| 0.3 | 82 |
| 0.5 | 82 |
| 1 | 82 |
| 2 | 74 |

It's doubtful that we can get better accuracy than 82.4%.
Ressources: cpu time: 10.39s, ram-cost: 8E-8.

## 2 SGDMultiClass

| learning rate | lambda regularization | Accuracy (%) |
|---|---|---|
| 0.01 | 0.01 | 80 |
| 0.01 | 0.05 | 87 |
| 0.05 | 0.01 | 80 |
| 0.05 | 0.05 | 87 |
| 0.05 | 0.1 | 89 |
| 0.1 | 0.05 | 87 |
| 0.1 | 0.1 | 89 |
| 0.1 | 0.2 | 91 |
| 0.2 | 0.1 | 89 |
| 0.2 | 0.2 | 91 |

Best : 91.4%, cpu time: 11.29s, ram-cost: 8.9E-8.

## 3 Hoeffding tree

| Split confidence | Accuracy (%) |
|---|---|
| 0.01 | 85 |
| 0.02 | 85 |
| 0.05 | 86 |
| 0.1 | 87 |
| 0.2 | 88 |
| 0.5 | 89 |
| 0.7 | 90 |
| 0.9 | 90 |
| 0.95 | 90 |

Best : 90.3%, cpu time: 36.10s, ram-cost: 1.4E-4.

## 4 Hoeffding adaptive tree

| Split confidence | Accuracy (%) |
|---|---|
| 0.8 | 90 |
| 0.9 | 90 |
| 0.95 | 90 |
| 0.99 | 90 |

Best : 90.3%, cpu time: 1min9s, ram-cost: 6.7E-6.

# 5 AWE

Let's use our best classifier so far the the AWE: SDGMultiClass.
This yields an accuracy of 57%, when the defaults (Naive Bayes) gave 80%.
Ressources hungry: 5m33 for naive Bayes and 1.3E-4 ram-cost.

# 6 Conclusion

Since the accuracy does not change all that much, I believe that ressources used is a better discriminatory criterium. Taking this into account, the multiclass stochastic gradient descent is what I would use, as it yields both the best accuracy and uses the least ressources.