IOT MOA LAB 1 - Davide Gallitelli - A.Y. 2017/18

The *Forest Covertype* dataset contains the forest cover type for 30 x 30 meter cells obtained from US Forest Service (USFS) Region 2 Resource Information System (RIS) data. It contains 581.012 instances and 54 attributes.

Such a huge dataset can be considered as a data stream by reading a certain number of elements thanks to the MOA software. This allows the analyst to compute accuracy and other metrics by building the model incrementally.

Quoting from the MOA Tutorial, when considering what procedure to use in the data stream setting,one of the unique concerns is how to build a picture of accuracy over time. Two main approaches arise:

- **(Periodic) Holdout**: When traditional batch learning reaches a scale where crossvalidation is too time consuming, it is often accepted to instead measure performance on a single holdout set.This is most useful when the division between train and test sets have been pre-defined, so that results from different studies can be directly compared.
- **Interleaved Test-Then-Train or Prequential**: Each individual example can be used to test the model before it is used for training, and from this the accuracy can be incrementally updated. When intentionally performed in this order, the model is always being tested on examples it has not seen.

The examples and tests following all share the **Interleaved test-and-train** approach, on **10 million instances** (instanceLimit = 10,000,000), with a **sample frequency equal to 10,000**. All other parameters are left to their default values.

A few different classifier have been tested. Classifier running longer have been stopped after 2 minutes. The results are as follows:

Classifier	Mean Accuracy	Run Time
Perceptron	83.48%	6.40s
KNN (k=10)	88.45%	2m
KNN+PAW+ADWIN	84.80%	2m
AdaptiveRandomForest Of		

HoeffdingTrees	90.7%	1m58s
HoeffdingTree	81.57%	16.62s
RandomHoeffdingTree	67.76%	7.45s
WEKAClassifier	66.83%	32.45s

The **AdaptiveRandomForest of HoeffdingTrees** proves to be the best classifier for this dataset, moreover it runs in a finite time (just under 2 minutes). However, if time constraints are a thing, than the *Perceptron* proves to be very good in terms of accuracy and run time.

An interesting experiment, given the great performances of the *Perceptron*, would be to test a *Multi-Layered Perceptron*, such as the one offered by the *Scikit-Learn* library, or a *Neural Network*.