Online Learning, Bandits, Reinforcement Learning

Advanced Machine Learning Lab Session

BLANCHARD Jérémie, Oussama BOULDJEDRI, Austin SCHWINN

## Online Passive-Aggressive Algorithms

### Without Noise

We implemented this algorithm in Python and we decided to use the classification generator from sickit learn to create a dataset that we are going to use to train our SVM, and another to test it. To update our weights, we had to implement three different methods to update them and compare our results.

We obtained the following results:

* With the classic update:

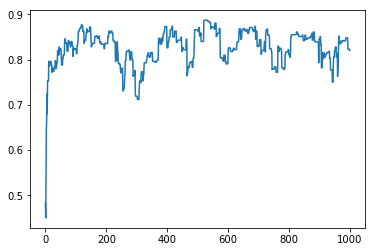


Figure 1- Accuracy in % / Number of iterations

*Final Accuracy: 82 %*

*Total Computational Time: 0.05 s*

* With a first relaxation with C=0.1:

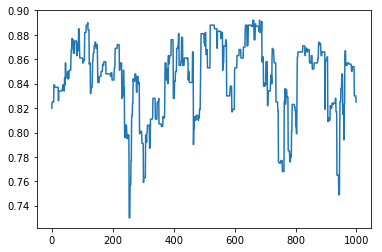


Figure 2- Accuracy in % / Number of iterations

*Final Accuracy: 82.5%*

*Total Computational Time: 0.11s*

* *With a second relaxation with C=0.1:*

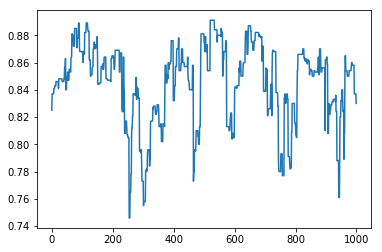
**

Figure 3-Accuracy in % / Number of iterations

*Final Accuracy: 83 %*

*Total Computational Time: 0.15s*

* With LibSVM we obtained:

*Final Accuracy: 61.4 %*

*Total Computational Time: 0.001s*

### Conclusion:

It seems that the different Tetas help to improve the accuracy but increase a lot the computational time. With LibSVM we have the same correlation and we have a bad final accuracy but a fast-computational time.

With 5% Noise  
 Now Let’s see how they behave when we add *5 percent* of noise.

We obtained the following results:

* With the classic update:

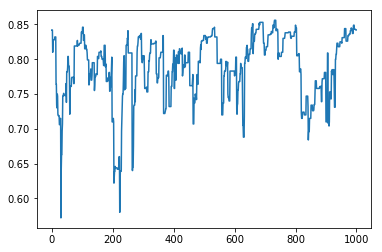


Figure 1- Accuracy in % / Number of iterations

*Final Accuracy: 84.2 %*

*Total Computational Time: 0.08 s*

* With a first relaxation with C=0.1:

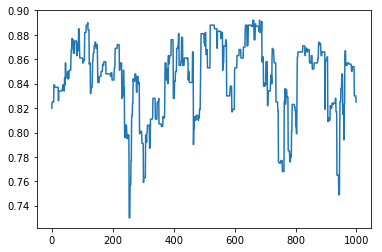


Figure 2- Accuracy in % / Number of iterations

*Final Accuracy: 85.1%*

*Total Computational Time: 0.15s*

* *With a second relaxation with C=0.1:*

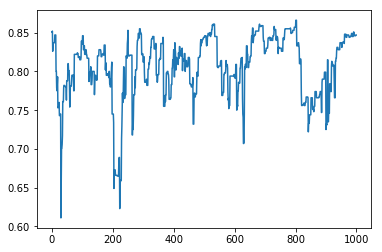
**

Figure 3-Accuracy in % / Number of iterations

*Final Accuracy: 84.7 %*

*Total Computational Time: 0.21s*

* With LibSVM we obtained:

*Final Accuracy: 55.5 %*

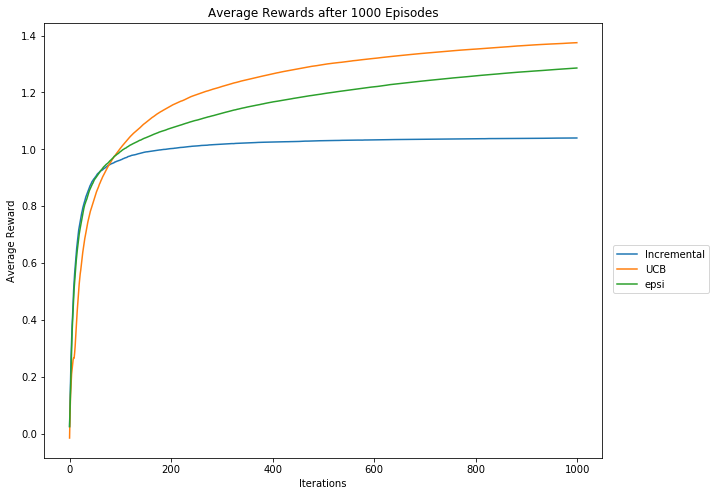
*Total Computational Time: 0.002s*

### Conclusion:

We can observe the same things as before except that this time we can clearly see that Teta 2 and 3 avoid the model to lose too many accuracies at some steps. Here again LibSVM does not perform well and has a final accuracy of only 55.5 % but it has also a small computational time.

## Bandit Algorithm

To compare these three different approaches, we decided to create a Python class where we implemented these algorithms. Then we are launching these approaches in parallel and see the results on a graph.



We can see that every algorithm has a specific shape and evolves differently. For example, it’s clear that the “Incremental Uniform” algorithm stops “learning” quickly while the UCB and Epsi algorithm keep learning even after a lot of iterations. It also seems that the average reward for the epsi algorithm is going to catch up with the UCB one after some more iterations.