Report: Assignment 1:
Big Data Analytics Programming

Andreas Hinderyckx

December 2021

1 Bugs

2 Learning Curves

The learning curves for the Perceptron (PC) and Very Fast Decision Tree (VFDT) run on the `clean` datasets are shown in figures 1 and 2 respectively. What's remarkable is that the PC achieves almost perfect accuracy while the VFDT only achieves an accuracy of approximately 75% on the clean data. The accuracy of the PC shows, however, much more oscillation whereas the VFDT's accuracy follows a more stable course. After around 100.000.000 examples have been trained with, we can clearly see the change of model used to generate the data in both graphs: both the PC's and VFDT's accuracy take a steep drop to 55% accuracy. The PC quickly recovers back up to its original state, wheras the VFDT doesn't reach the same accuracy it had before: this could be due to the VFDT being overfit to the previously used model to generate the data, whereas the PC's weights can be changed entirely as needed.

Tested on the `noisy` data set, we acquire the learning curves shown in figures 3 and 4. Now the oscillations of the PC's accuracy become clearly visible: it struggles to learn the model due to the added noise. The VFDT's graph is highly similar to its graph for the `clean` data, with a reduction of achieved accuracy of 10% and some extra oscillation in the accuracy.

3 Experiments

To study the effect of different parameters present in the programs, we pose the following questions:

1. What's the effect of varying η (eta) in the PC implementation?

2. What's the effect of varying δ (delta) in the VFDT implementation?

3. What's the effect of varying τ (tau) in the VFDT implementation?

4. What's the effect of varying n_{min} (how often split function is recalculated) in the VFDT implementation?

The learning curves for the PC with varying values for η on the `clean` data set are shown in figure 5. We notice that smaller learning learning rates result in less oscillating accuracies throughout the learning process.

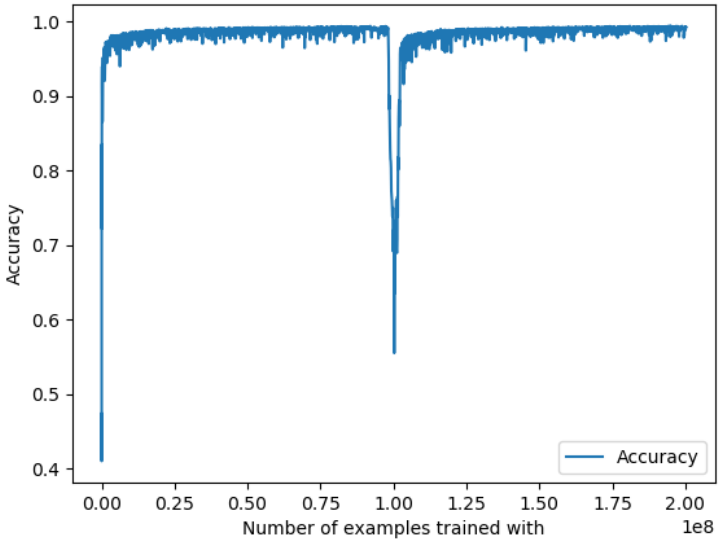

Figure 1: Learning curve of PC on the `clean` dataset

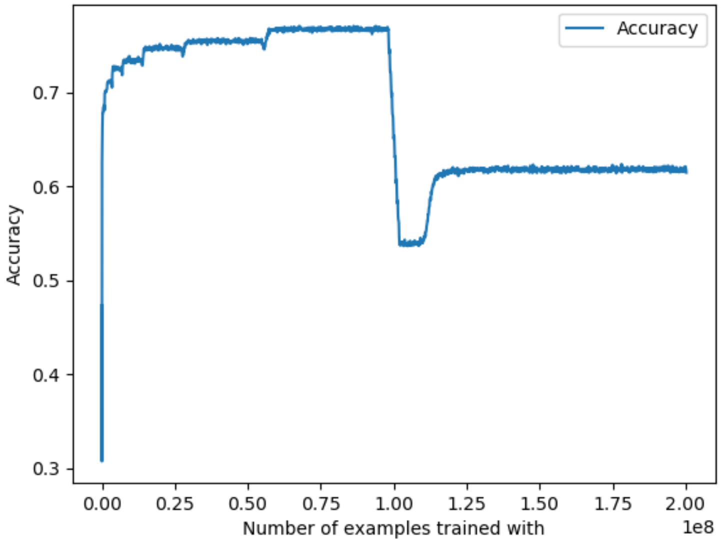

Figure 2: Learning curve of VFDT on the `clean` dataset

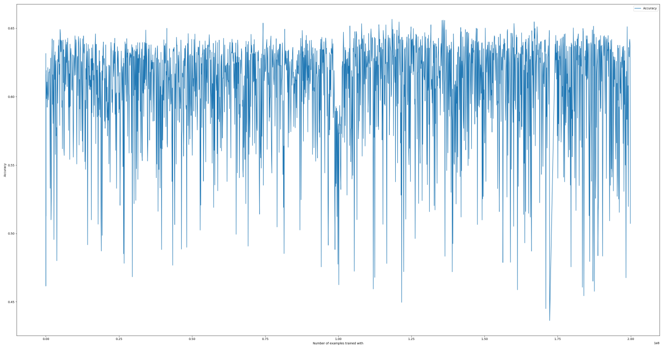

Figure 3: Learning curve of PC on the `noisy` dataset

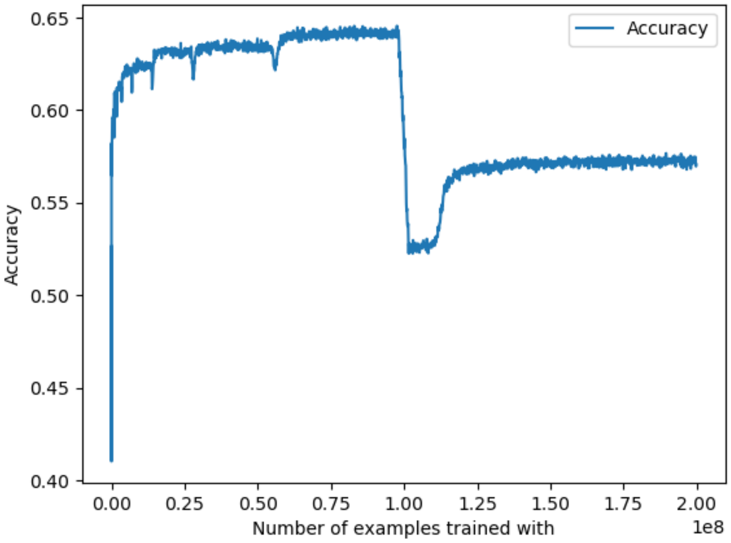

Figure 4: Learning curve of VFDT on the `noisy` dataset

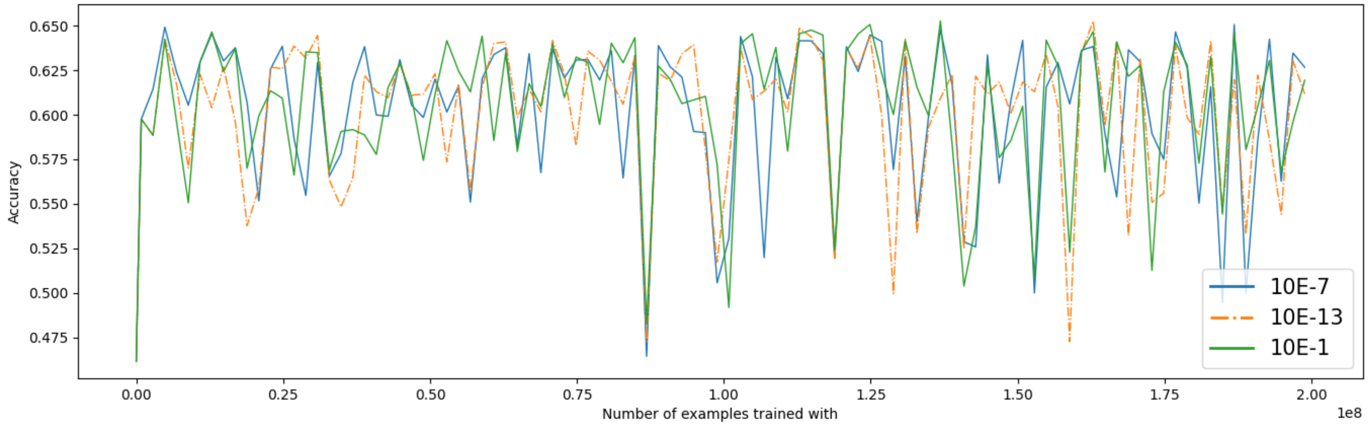

Figure 5: Effect of varying η

For studying the parameters of the VFDT, we use the `clean` dataset. In figure 6 we can see that varying δ affects the speed at which accuracy is gained. In the initial learning phase $\delta := 10\text{e}{-9}$ seems to gain accuracy the quickest, while after the concept drift $\delta := 10\text{e}{-7}$ is more succesful (see fig. 7). In general, small values for δ lead to higher accuracy: this is what we expect, as increasing δ increases the Hoeffding bound and thus will wait for a higher difference between the two highest split evaluation values before splitting. From figure 8 we can deduce that smaller values of τ increase the learning efficiency: this makes sense as the algorithm spends less time on deciding between minor differences in split evaluations and will be able to split sooner on these cases. We also note that for larger values of τ, the experiments didn't finish due to excessive memory usage, as the algorithm will then split more often and thus consume more memory as τ increases. Finally, increasing n_{min} implies the VFDT will re-compute the split evaluation function less frequently, which could increase the computation times, but should decrease the rate at which is learned, as valid splits are discovered with a delay. These assumptions are confirmed by figure 9, of which an enlarged view can be seen in figure 10.

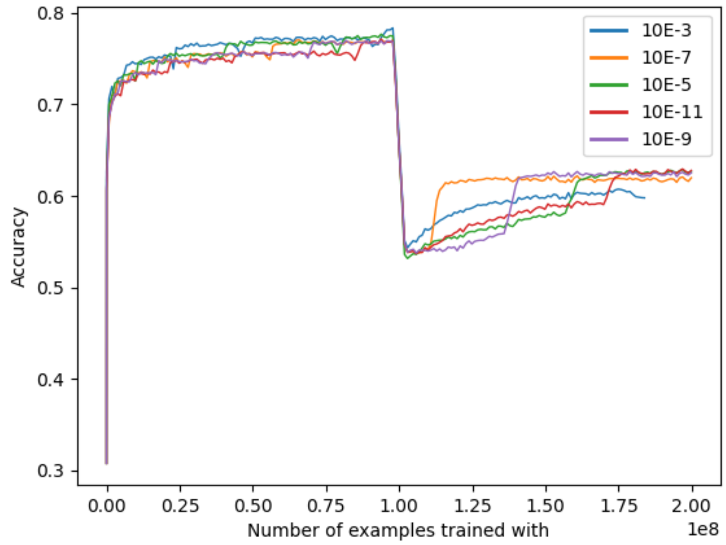

Figure 6: Effect of varying δ

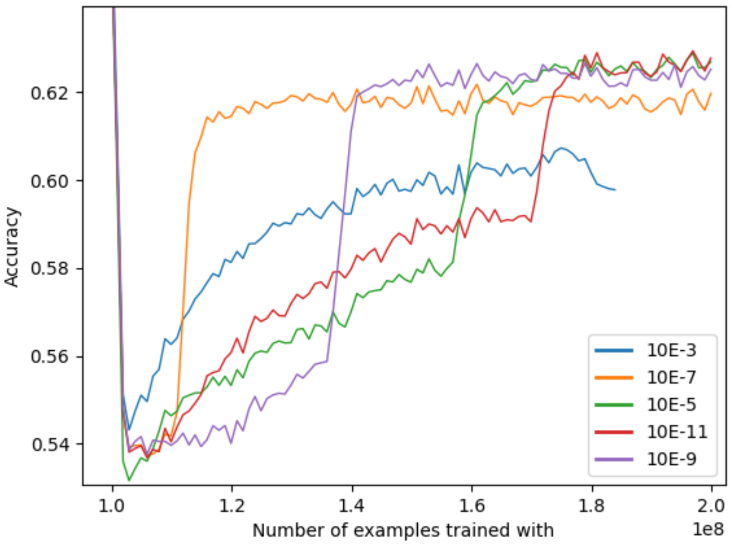

Figure 7: Enlarged view of figure 6

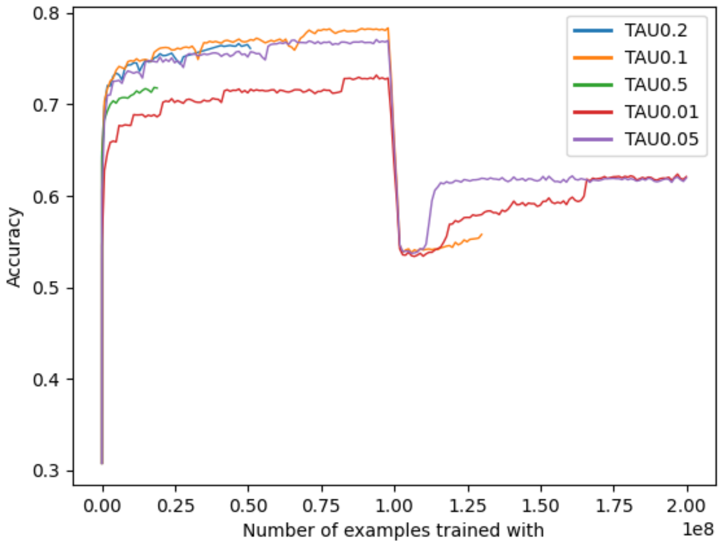

Figure 8: Effect of varying τ

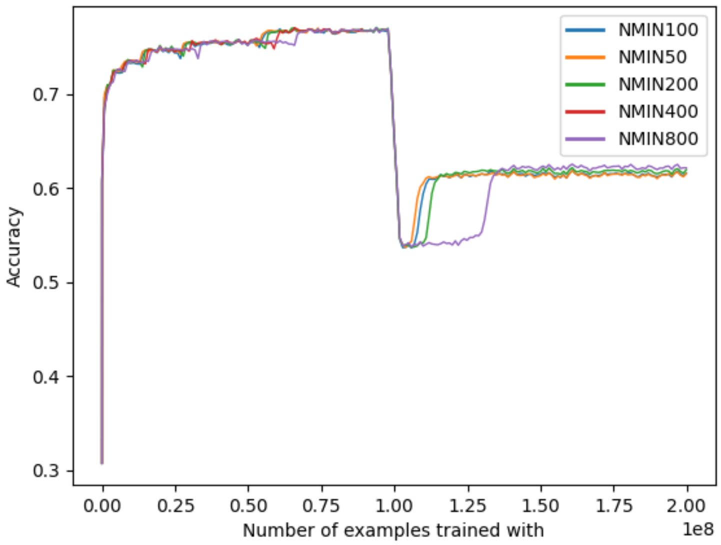

Figure 9: Effect of varying n_{min}

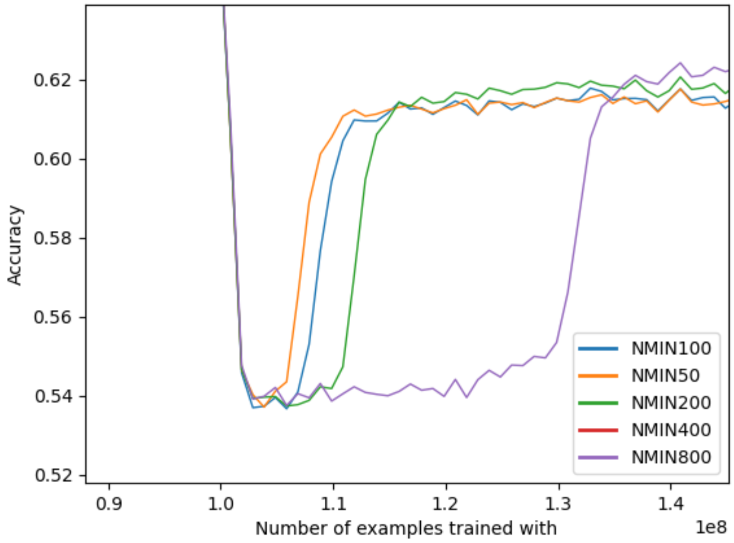

Figure 10: Enlarged view of figure 9