INF8225 – Intelligence Artificielle: Techniques Probabilistes et d'Apprentissage

TP2

Éric Morissette

1631103

19 Février 2016

École Polytechnique de Montréal

Partie I – Approche par Batch vs Mini-Batch

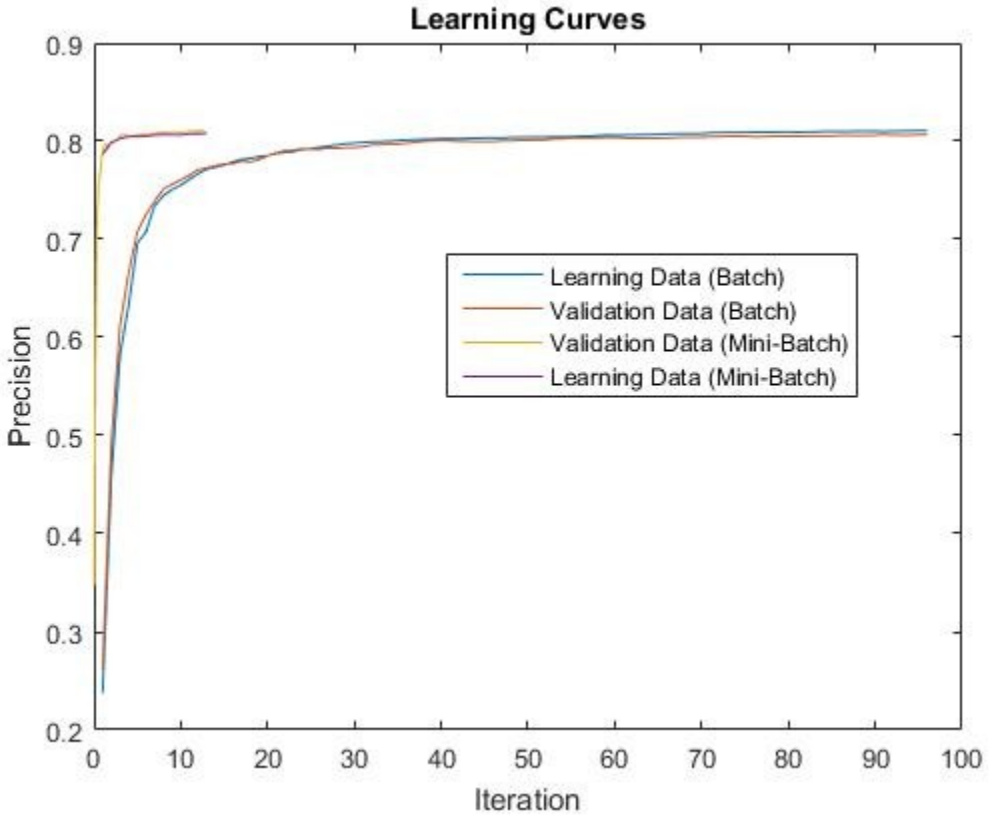
Pour la première partie du laboratoire, nous avions à utiliser deux algorithmes de gradient. Le premier était par batch entière, comprenant toutes les données, tandis que le deuxième utilisait des mini-batchs, qui représentent une fraction de l'ensemble de valeurs d'apprentissage possible.

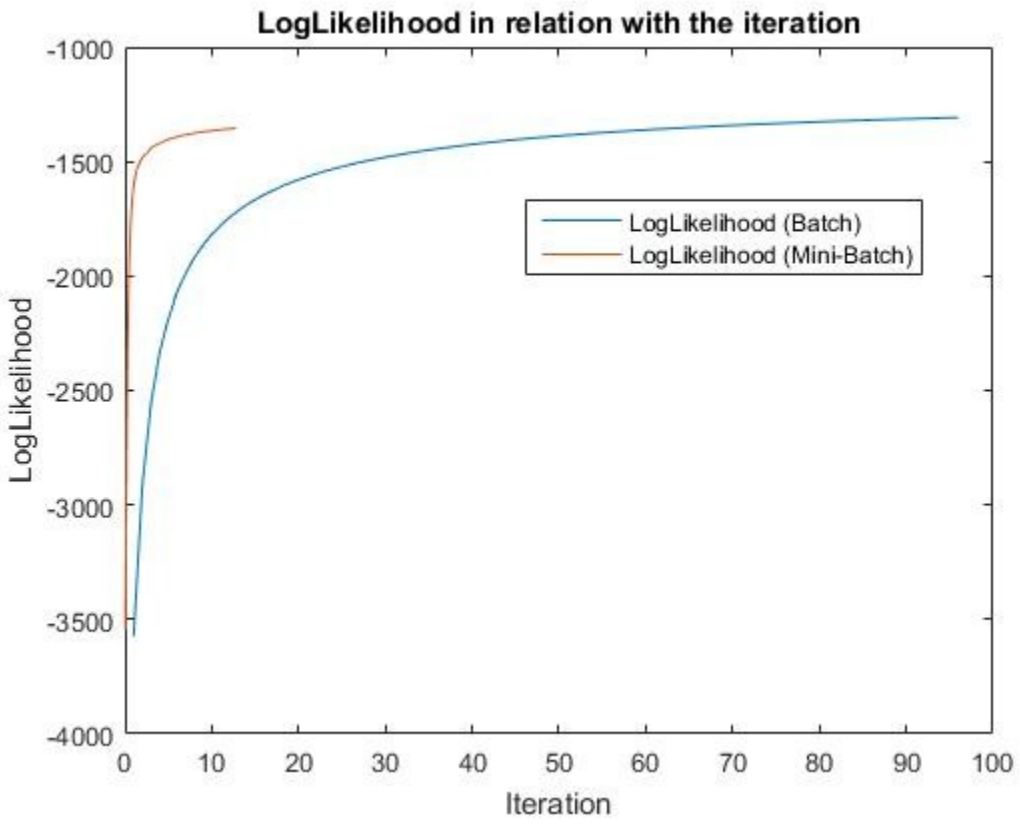
Voici ce que l'on obtien après avoir exécuté les deux algorithmes l'un à la suite de l'autre, en reinitialisant nos variables et en relisant les données du fichier mat.

```
Precision of the batch descent with test data = 0.804187
Precision of the mini-batch descent with test data = 0.807471
```

On peut voir que la précision tourne aux alentours de 80% pour les deux approches, cependant, l'approche par mini-batchs est beaucoup plus rapide. Afin de comparer, l'algorithme par batch prend environ une centaine d'itérations tandis que les mini-batchs en prennent une trentaine. Ces résultats sont sensiblement constant, tant pour le pourcentage de prédiction que pour le nombre d'itérations nécessaires.

Ici, nous pouvons voir les graphiques des courbes d'apprentissages et de log vraisemblance en fonction du nombre d'itérations des algorithmes. Les deux algorithmes de gradient sont placés sur le même graphique afin de faciliter la comparaison.





Sur les graphiques, il est facile de voir que l'approche par mini-batch converge beaucoup plus rapidement que l'utilisation d'une seule batch. Une explication de ce phénomène est le fait que si l'on fait une itération de mini-batch, on fait en fait 20 petites itérations internes, ce qui nous donne une convergeance accrue.

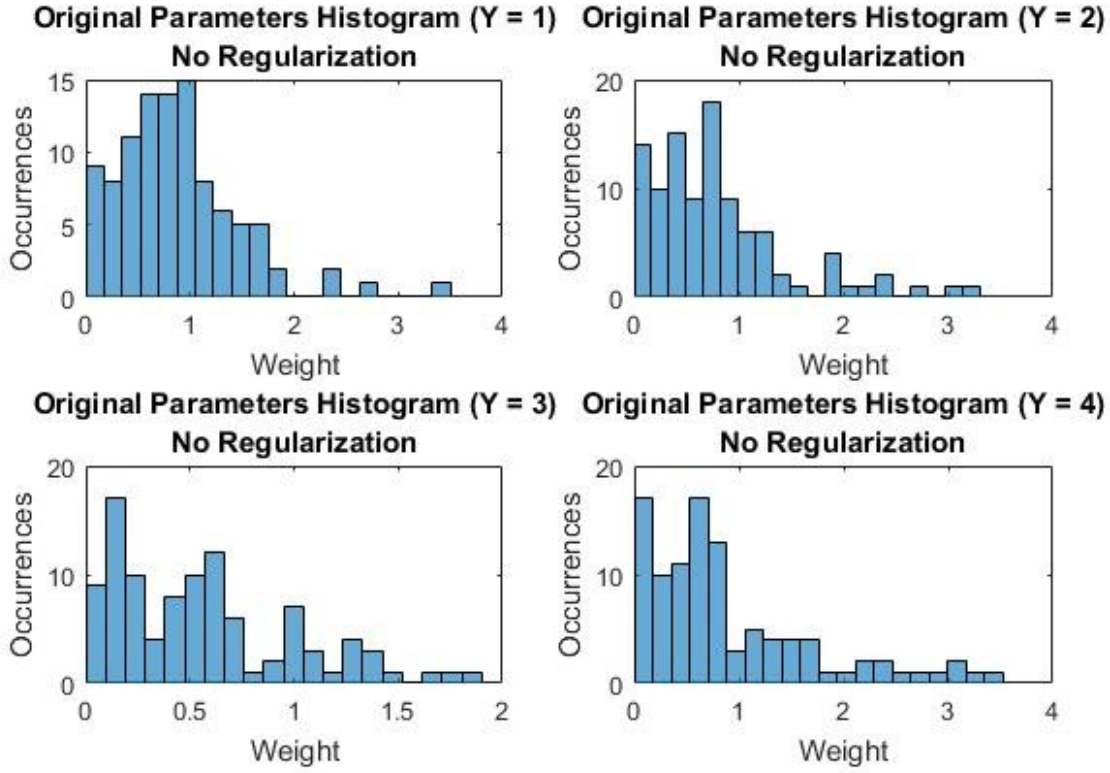
Pour l'approche par mini-batchs, une augmentation de la grosseur des batch va entrainer une réduction de vitesse, jusqu'au point où la taille de la “mini-batch” sera de l'ensemble des données, ce qui nous revient à faire une descente par batch.

Partie II – Elastic Net

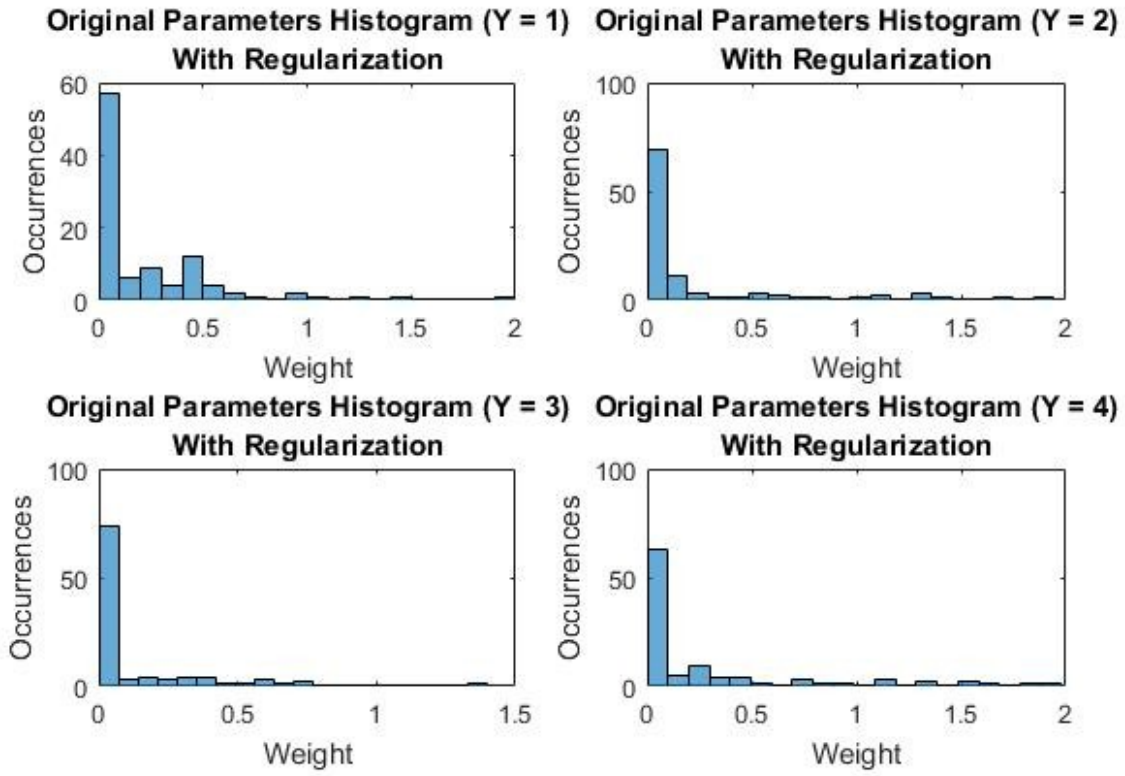
Pour la deuxième partie du laboratoire, il fallait utiliser un algorithme d'Elastic Net, qui permet de ne pas surentrainer notre algorithme avec les valeurs d'apprentissage. Afin de vraiment prouver l'utilité de l'algorithme, nous allons ajouter une centaine de données aléatoires aux échantillons d'apprentissage, de validation et de test.

L'algorithme d'Elastic Net est appliqué sur notre approche par mini-batch en utilisant 20 petites batchs. En gros, l'algorithme d'EN est là pour réduire le poids des points aléatoires afin de les ignorer lors de l'apprentissage des données. Cet algorithme utilise deux variables, nommées lambda1 et lambda2, afin de calculer le gradient de chaque mini-batch.

Voyons ce que l'algorithme donne sous forme de graphiques.



Ici nous pouvons voir que le poids des valeurs est vraiment étalé, ce qui veut dire que les valeurs aléatoires “poubelles” ajoutées au débeut seront prises en compte dans le calcul du gradient.



Avec l'utilisation d'un Elastic Net, on voit que nos valeurs ayant un poids non-négligable sont beaucoup plus concentrées.

Le fait de concentrer nos données sur certaines valeurs permet d'accélérer grandement le calcul tout en gardant un pourcentage d'environ 80% de précision.