

# IFT 603-712 : Devoir 2

## Travail par équipe de 3

Remettez votre solution aux numéros 1, 2 et 3 en format pdf ou manuscrit (et scanné) via [turninweb](#).

**1- [1.5 point]** Prouvez que la régression de Ridge ayant pour objectif de minimiser la fonction d'énergie suivante (voir les notes de cours portant sur la régression linéaire):

$$E(\vec{x}) = \frac{1}{2} \sum_{n=1}^N (t_n - \vec{w}^T \vec{\phi}(\vec{x}_n))^2 + \frac{\lambda}{2} \vec{w}^T \vec{w}$$

a pour solution l'équation suivante :

$$\vec{w} = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T t$$

**2- [2.5 points]** Nous avons vu que l'objectif de la régression logistique est de trouver un vecteur de paramètres  $\vec{w}$  pouvant reproduire la probabilité conditionnelle suivante

$$p(C_1 | \vec{\phi}(\vec{x})) = \sigma(\vec{w}^T \vec{\phi}(\vec{x})).$$

Prouvez que lorsque la fonction de perte est une entropie croisée (*cross-entropy*), le gradient utilisé pour effectuer la descente de gradient est déterminé par la fonction suivante :

$$\nabla E(\vec{w}) = \sum_{n=1}^N (y(\vec{x}_n) - t_n) \vec{\phi}(\vec{x}_n).$$

**3- [1.0 point]** Soit une variable aléatoire X pouvant prendre 3 valeurs possibles {1, 2, 3} avec probabilités  $p(X = 1) = p_1$ ,  $p(X = 2) = p_2$  et  $p(X = 3) = p_3$ . Démontrez que la loi de probabilité ayant l'entropie la plus élevée et satisfaisant la contrainte  $p_1 = 2p_2$  a les probabilités suivantes :

$$p_1 = \frac{2}{2^{2/3} + 3}, \quad p_2 = \frac{1}{2^{2/3} + 3}, \quad p_3 = \frac{2^{2/3}}{2^{2/3} + 3}$$

Pour ce faire, utilisez des multiplicateurs de Lagrange afin de tenir compte de la contrainte de sommation à 1 et de la contrainte  $p_1 = 2p_2$ . Pour le calcul de l'entropie, prenez un logarithme en base 2.

**4-[1.0 points]** Soit un problème de classification binaire dont le modèle est entraîné à l'aide d'une entropie croisée

$$EntCroisée(\vec{w}) = \frac{-1}{N} \sum_{n=1}^N \sum_{k=1}^K t_{nk} \ln(y_k(\vec{x}_n))$$

où k est l'indice de classe. (a) Expliquez en quelques lignes et/ou à l'aide de schéma ce qui arriverait si, par erreur, on utilisait une simple entropie pour entraîner le réseau soit

$$Ent(\vec{w}) = \frac{-1}{N} \sum_{n=1}^N \sum_{k=1}^K y_k(\vec{x}_n) \ln(y_k(\vec{x}_n))$$

(b) En quelques ligne, donnez un effet bénéfique à utiliser une fonction de coût qui combine les deux types d'entropies soit :

$$E(\vec{w}) = EntCroisée(\vec{w}) - \lambda Ent(\vec{w})$$

où  $\lambda$  est un terme multiplicatif (hyper-paramètre) entre 0 et 1.

**5-[4.0 points]** Programmez des algorithmes de classification linéaire fonctionnant en 2D. Pour ce faire, vous devez télécharger le fichier **devoir2.zip** du site web du cours.

Les algorithmes doivent être implémentés à l'intérieur de la classe **ClassifieurLineaire**, dans le fichier **solution\_classification\_lineaire.py** qui contient déjà une ébauche. Le premier algorithme (**2.25 points**) est l'approche générative vue en classe et présentée à la section 4.2.2 du livre de Bishop. La deuxième méthode est celle du Perceptron (**2.25 points**) également vue en classe et présentée à la section 4.1.7 du livre de Bishop. Pour ce faire, il vous faudra coder la descente de gradient stochastique vue en classe. Vous devez également implémenter la méthode du Perceptron proposée par la bibliothèque Sklearn<sup>1</sup> (**0.5 point**). Il est d'ailleurs recommandé de commencer par cette méthode qui est la plus facile à coder (à peine 4 lignes dans la solution du prof!).

Veillez vous référer aux commentaires sous la signature de chaque méthode de la classe **ClassifieurLineaire** afin d'avoir plus de détails quant à leur implantation. À noter que votre code doit être efficace et utiliser les fonctionnalités de la bibliothèque Numpy.

**Note 1** : bien que vide, le code de la classe **classificationLineaire** s'exécute déjà. Pour vous en convaincre, vous n'avez qu'à taper la commande suivante dans un terminal :

```
python classifieur_lineaire.py 1 280 280 0.001 1 1
```

**Note 2** : le code des devoirs (ainsi que des notebooks) a été testé avec python 3.5.2. Pour faire fonctionner votre code sur les ordinateurs du département, vous devez

1. démarrer une session linux (ubuntu)
2. démarrer un terminal
3. normalement, si vous démarrez une session ipython (tappez ipython dans le terminal) vous verrez « Python 3.5.2 » (ou plus récent).

**Note 3** : il est recommandé de rédiger votre code dans un ide tel *Spyder* ou *Pycharm*.

**Note 4** : pour exécuter le code des notebooks disponibles sur le site web du cours, vous devez taper la commande « jupyter notebook » dans un terminal.

**Note 5** : en plus de votre code, veuillez soumettre un fichier "**gitlab.txt**" dans lequel vous donnez le lien vers votre dépôt gitlab. Il est obligatoire d'utiliser le service gitLab de l'université (gitHub, Bitbucket, etc. sont interdits.)

**Note 6** : en plus de vos documents de travail, vous devez remplir, signer et joindre à votre travail le **formulaire d'intégrité** disponible sur le site web du cours.

**Note 7** : en plus de vos documents de travail, vous devez remplir et joindre à votre travail votre **agenda d'équipe** disponible sur le site web du cours.

---

<sup>1</sup><http://scikit-learn.org/>