IFT 603-712 : Devoir 4

Travail par équipe de 2 ou 3

Remettez votre solution aux numéros 1 et 2 en format pdf ou manuscript (et scanné) via **turninweb**. Même chose pour le code.

1- [1.5 points] Nous avons vu dans le cours que l'opération Softmax est représentée par l'équation mathématique suivante :

$$y_{\vec{w}_i}(\vec{x}) = \frac{e^{a_i}}{\sum_c e^{a_i}}$$

où a_i est la sortie du i-ème neurone de sortie et que la fonction de perte entropie croisée (*cross-entropy*) est

$$E_D(W) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{kn} \ln y_{\vec{w}_i}(\vec{x})$$

où t_{kn} est la cible du n-ième élément de la base de données d'entraînement. Donner l'équation du gradient de la fonction de perte par rapport à a_i ainsi que toutes les étapes mathématiques pour arriver à ce résultat.

2- [1.5 points] On vous demande de classifier les quatre (4) modèles de véhicules suivantes : (1) voitures sports, (2) voitures familiales, (3) camionnettes, (4) camions lourds. Pour ce faire, vous disposez pour chaque véhicule des cinq caractéristiques suivantes : (i) longueur, (ii) poids, (iii) accélération 0-100 km/h, (iv) prix, (v) consommation d'essence. Dans ce contexte et considérant qu'on vous fournit un ensemble d'entraînement, expliquez (a) ce qu'est une distribution de vraisemblance et comment vous pourriez la calculer, (b) ce qu'est la distribution a priori et comment vous pourriez la calculer et (c) dites s'il est raisonnable ou non d'émettre l'hypothèse que la distribution

p(consommation d'essence|voitures sports)

est gaussienne. Justifiez bien vos réponses.

3- [1 point] Démontrer le lien qu'il y a entre la descente de gradient de type *momentum*, et les formules pour calculer la position, la vitesse et l'accélération d'un objet en mouvement.

4- [6 points] Programmez des réseaux de neurones à 1 et 2 couches afin d'avoir un classifieur linéaire et un classifieur non linéaire telle qu'illustré au chapitre 7 du cours. Pour ce faire, vous devez implanter un entropie croisée (*cross-entropy*) avec une couche *Softmax* à la fin du réseau. Le classifieur non linéaire exige l'implantation d'un réseau avec une couche d'entrée, une couche cachée et une couche de sortie. Le code est contenu dans le fichier **devoir4.zip** via le site web du cours.

Les algorithmes doivent être implémentés à l'intérieur des fichiers pythons **`linear_classifier.py`** et **`two_layer_classifier.py`** qui contiennent déjà une ébauche et des mentions **TODO** aux endroits où vous devez ajouter du code. L'exécution des fonctions et des classes associées à ces fichiers se fait via le notebook **`Devoir4.ipynb.`** Comme vous le verrez, ce notebook contient bon nombre de « sanity checks » mentionnés tout au long du cours.

Note 1 : bien que vide, le code du notebook fonctionne déjà. Pour vous en convaincre, vous n'avez qu'à taper la commande suivante dans un terminal :

```
jupyter notebook
```

Et de sélectionner le fichier **`Devoir4.ipynb.`**

Note 2 : le code des devoirs (ainsi que des notebooks) a été testé avec python 3.5 sous Linux.

Note 3 : il est recommandé de rédiger son code dans un ide tel `spyder` ou `pycharm`.

Note 4 : voici de quoi devrait avoir l'air votre solution pour le modèle non linéaire appliqué aux données « Ncircles » 4 classes :