COMPTE RENDU

Participants: Leonardo Nassabain

1. Préparation des données

1.) Combien d'attributs ces données comportent-elles?
Elles comportent 2 attributs: x1, x2.

2.) En combien de classes ces données sont-elles séparées?
Elles sont séparées en 3 classes: class-0, class-1 et class-2.

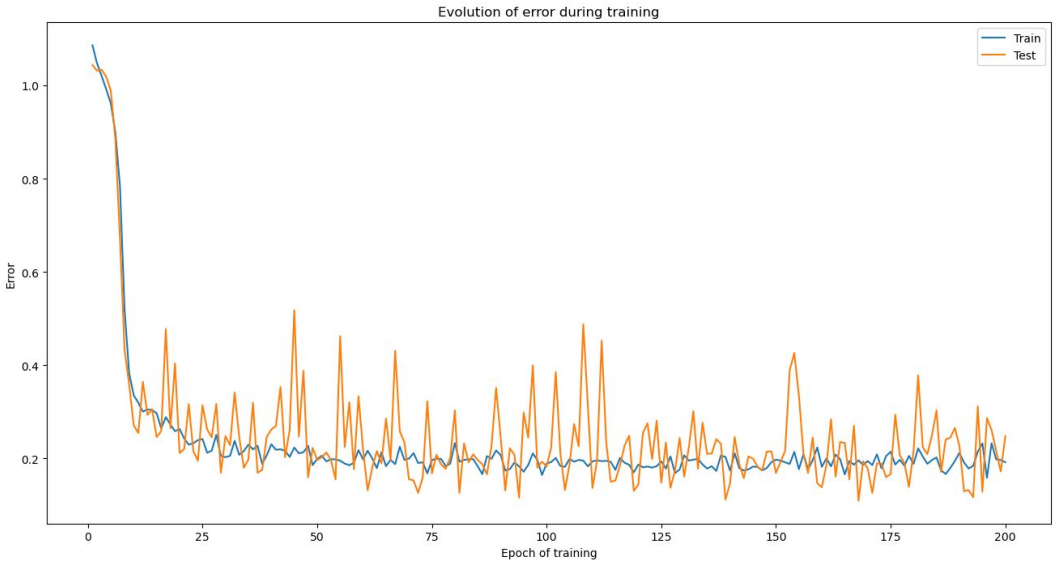
3.) Ces données sont-elles linéairement séparables?
Ces données ne sont pas linéairement séparables car il n'existe aucun classifieur linéaire qui pourrait correctement séparer les classes les unes des autres. Cela est dû au fait que les données d'une classe prennent une forme circulaire dans un repère.

2.)b)
La variable test_final_df contiendra un échantillon aléatoire de données du dataframe gaussian_df. Cet échantillon représente 20% des données du gaussian_df. La variable test_final_df est reproduisible car on a défini un seed. (random_state = 42)
La variable gaussian_df contiendra toutes ses valeurs précédentes en excluant les valeurs qui sont maintenant contenues dans la variable test_final_df.

2. Construction du modèle

2.3
Ce modèle comporte 2*4 + 4*3 + 3*2 + 2*3 poids différents et 4+3+2+3 biais. Donc en tout 44 paramètres.
Si on augmente le pas d'apprentissage à 0.1, les pas pris pendant la descente de gradient pourraient être trop importants, le comportement de l'entraînement pourrait devenir instable et en faisant des pas trop larges, on pourrait dépasser le minimum de notre fonction de coût.

<u>Affichage de l'évolution de l'erreur de mon modèle sur le jeu d'entraînement et sur le jeu de test :</u>



3. Analyse du modèle

3.1

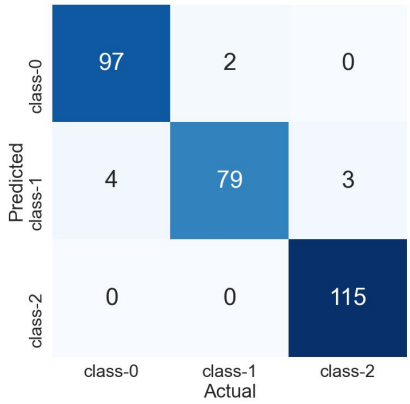
Pour l'instance i=0, la classe prédite est la classe-2. Cette prédiction est correcte car dans y_actual[0] la 3ème colonne est à 1, signifiant que la classe correcte est effectivement la classe-2.
Sans entraînement, le pourcentage de prédictions correctes de mon modèle est autour de 27%, en entraînant le modèle pendant les 200 époques, le pourcentage s'élève à entre 86% et 97%.

3.2

L'erreur la plus fréquente de mon modèle survient quand il faut faire le classement d'un point appartenant à la classe-1 ou bien quand quand le modèle prédit qu'un point appartenant à la classe-0 ou classe-2, appartient à la classe-1. Autrement dit, la majorité des erreurs se trouve sur la ligne 2 et sur la colonne 2 de la matrice de confusion. Cela arrive parce que les points appartenant à la classe-1 se trouvent entre les points de la classe-0 et de la classe-2, par conséquent la moindre erreur peut mener à une prédiction fausse.

Cette analyse complémente le pourcentage de prédictions correctes car, connaissant le taux de prédictions correctes nous connaissons aussi le taux de prédictions fausses. Et avoir une matrice de confusion nous permet d'avoir une idée plus claire de l'origine de ces erreurs.

Une matrice de confusion obtenue après l'entraînement du modèle :



Remarques sur le projet :
Le projet était vraiment intéressant. Au début, les notions évoqués me paraissaient floues mais aboutissant à la fin de ce projet, la construction des réseaux de neurones me semble limpide. Le seul problème récurrent qui s'est présenté au cours du projet est lié aux dimensions des matrices et des tableaux Numpy. Par conséquent, il se peut que dans certaines fonctions, les solutions que je vous propose auraient pu être plus simples. Certaines fonctions que j'ai faites n'étaient pas requises (ex. evolution_plot()), mais elles m'ont facilité la mise en place des test du réseau.
Ne sachant pas exactement si je dois rendre uniquement le code 'source' ou l'intégralité de mon code, à la fin du fichier **neuron.py** (lignes 266-284), vous trouverez une partie délimitée et mise en commentaires qui vous permettra de tester le modèle que j'ai crée. Pour l'utiliser il suffit d'enlever le '#' qui se trouve au début de chaque ligne. Le test proposé consiste à créer un modèle, l'entraîner pendant 200 époques, afficher l'évolution de l'erreur sur le jeu d'entraînement et sur le jeu de test, puis faire des prédictions sur le jeu de données X_test_final et afficher le pourcentage de prédictions correctes et enfin afficher la matrice de confusion faite à partir de ces prédictions.