**Les bases de Machine Learning**

**2024/2025**

**II-BDCC 1ere Année**

**Devoir**

**Problème de régression N°1: dataset synthètique**

Niveau de complexité du problème : simple

**Rappel sur les réseaux de neurones**

**Un réseau de neurones se compose de :**

* Inputs
* 0 ou plusieurs couches cachées
* Une seule couche de sortie
* Chaque couche contient un ou plusieurs neurones

**Le réseau de neurones le plus simple se compose de :**

* Inputs
* 0 couche cachée
* Une couche de sortie avec un seul neurone

**Un neurone se compose de :**

* Inputs : x1,x2, ..xm
* Paramètres : bias, w1, w2…wm
* weighted sum: z=bias+w1.x1+w2.x2+…+wm.xm
* Output=f(z) ou f est une fonction d’activation

**Exercice 1 : Régression basée sur les réseaux de neurones avec tensorflow & keras**

**Ouvrir le fichier devoir\_regression.ipynb**

**Répondre aux questions tout en mettant à jour le fichier.**

1. **Préparation de données – code source section 1, fourni**
   1. Générer un dataset synthètique
   2. Extraire la partie features : X = np.column\_stack((X1, X2)) et y
   3. Diviser le dataset en training set (80) et test set (20%)
   4. Expliquer l'importance de division en training set et test set

**Compléter les codes sources du fichier .ipynb**

**Là où il ya des ?, il faut la remplacer par l’instruction adéquate**

1. Modèle de réseau de neurones

2-a) Architecture de réseau de neurones :

2-a-1) Proposition d’une architecture

|  |  |
| --- | --- |
|  | Un réseau de neurones composé de :   * 2 inputs (X1,x2). Chaque input est accompagné par un weight. * Un bias * Unité de calcul 1 : Une sommation pondérée, z=w1.X1+w2.X2+bias * Unité de calcul 2 : Une activation, f(z). f peut être linear,sigmoid, relu,… |

2-a-2) à partir de la nature du dataset, établir les inputs du réseau de neurones

2-a-3) à partir de la nature du dataset, établir le nombre de neurones à mettre dans outputlayer du réseau de neurones.

2-a-4) à partir de la nature de target, établir la fonction d'activation de la couche d'output.

2-b) sans aucune couche cachée, créer un modèle basé sur un réseau de neurones « model\_nn » qui correspond à ce problème

2-c) compiler le modèle :

optimizer= adam, cost function adequate, metrics: mse

2-d) entrainer le modèle en utilisant **4100** epochs  
si les ressources de mémoire de votre machine sont insuffisantes, mettre uniquement 100 epochs

2-e) préciser ce que fait la fonction fit

2-f) quel est le nombre de paramètres du réseau

-faire un calcul à la main ET afficher le détail du réseau

2-g) afficher les paramètres du réseau de neurones

2-h) dessiner le réseau de neurones en utilisant les paramètres trouvés

2-i) dans quels cas il est important de faire des régulations (tuning)

1. **Prédiction en utilisant le modèle**

3-a) En utilisant le modèle « model\_nn », faire les prédictions en utilisant X\_test. Stocker le résultat dans yhat\_nn

3-b) En utilisant les paramètres du modèle, faire une prédiction sans utilisation de predict de

tensorflow.keras.Sequential

1. **Évaluation du modèle**

4-a) Performance du modèle sur training set

* Évaluer le modèle en utilisant mse, r2
* Calculer y\_test- yhat\_lin\_reg
* Tracer le graphique des résidus(yhat\_lin\_reg,y\_test- yhat\_lin\_reg)
* Interpréter les résultats

4-b) Performance du modèle sur test set

4-c) À quoi peut servir, évaluer le modèle en utilisant training set et test set en même temps

**Exercice 2 : Régression basée sur la régression linéaire sans utilisation de sklearn**

**(Version avec matrices)**

**Compléter les codes sources du fichier .ipynb**

**Là où il ya des ?, il faut la remplacer par l’instruction adéquate**

1. Calculs élémentaires pour une itération

Et on suppose la dataset suivant

|  |  |  |
| --- | --- | --- |
| X1 | X2 | Y |
| 1 | 2 | 10 |
| 3 | 4 | 20 |
| 5 | 6 | 30 |
| 7 | 8 | 40 |

On suppose le modèle suivant :

bias=0.1, W=[[0.2], [0.3]]

1. Calcul de la prédiction :
2. Calcul des erreurs
3. Calcul des gradients
4. Mise à jour du modèle

NB. Pour les différents calculs, écrire des bouts de code qui font les calculs

1. Compléter le code source de manière à entrainer le modèle
2. Évaluer le modèle
3. Comparer les résultats obtenus avec le modèle de régression linéaire avec sklearn

**Problème de classification N°1: dataset de diabetes**

Niveau de complexité du problème : simple

**Exercice 1 : Classification basée sur neural network avec utilisation de tensorflow**

**Ouvrir le fichier devoir\_classification.ipynb**

**Répondre aux questions tout en mettant à jour le fichier.**

**Compléter les codes sources du fichier .ipynb**

**Là où il ya des ?, il faut la remplacer par l’instruction adéquate**

1. **Préparation de données**

1-a) importer le data de diabetes.csv

1-b) Extraire la partie features X et la target y

1-c) Diviser le dataset en training set (80) et test set (20%)

1. **Modèle de réseau de neurones**

2-a) Architecture de réseau de neurones :

2-a-1) à partir de la nature du dataset, établir les inputs du réseau de neurones

2-a-2) à partir de la nature du dataset, établir le nombre de neurones à mettre dans outputlayer du réseau de neurones.

2-a-3) à partir de la nature de target, établir la fonction d'activation de la couche d'output.

2-b) sans aucune couche cachée, créer un modèle basé sur un réseau de neurones « model\_nn » qui correspond à ce problème

2-c) compiler le modèle :

optimizer= adam, cost function adequate, metrics: accuracy, precision

2-d) entrainer le modèle en utilisant **1000** epochs  
si les ressources de mémoire de votre machine sont insuffisantes, mettre uniquement 100 epochs

2-e) préciser ce que fait la fonction fit

2-f) quel est le nombre de paramètres du réseau

-faire un calcul à la main ET afficher le détail du réseau

2-g) afficher les paramètres du réseau de neurones

2-h) dessiner le réseau de neurones en utilisant les paramètres trouvés

2-i) dans quels cas il est important de faire des régulations (tuning)

1. **Prédiction en utilisant le modèle**

3-a) En utilisant le modèle « model\_nn », faire les prédictions en utilisant X\_test. Stocker le résultat dans yhat\_nn

3-b) En utilisant les paramètres du modèle, faire une prédiction sans utilisation de predict de

tensorflow.keras.Sequential mais en utilisant des matrices

1. **Évaluation du modèle**

4-a) Performance du modèle sur training set

* Évaluer le modèle en utilisant mse, r2
* Calculer y\_test- yhat\_lin\_reg
* Tracer le graphique des résidus(yhat\_lin\_reg,y\_test- yhat\_lin\_reg)
* Interpréter les résultats

4-b) Performance du modèle sur test set

4-c) À quoi peut servir, évaluer le modèle en utilisant training set et test set en même temps

**Exercice 2 : Classification basée sur la régression logistique sans utilisation de sklearn**

**(Version avec matrices)**

**Compléter les codes sources du fichier .ipynb**

**Là où il ya des ?, il faut la remplacer par l’instruction adéquate**

1. Utiliser le même dataset de l’exercice 1
2. Pourquoi développer des modèles sans utilisation de librairies ?
3. Quelles sont les principales étapes à suivre
4. Comment les dérivées de cost function par rapport aux paramètres peuvent participer à la mise à jour du modèle
5. Compléter le code source de manière à entrainer le modèle
6. Évaluer le modèle
7. Comparer les résultats obtenus avec le modèle de régression linéaire avec sklearn