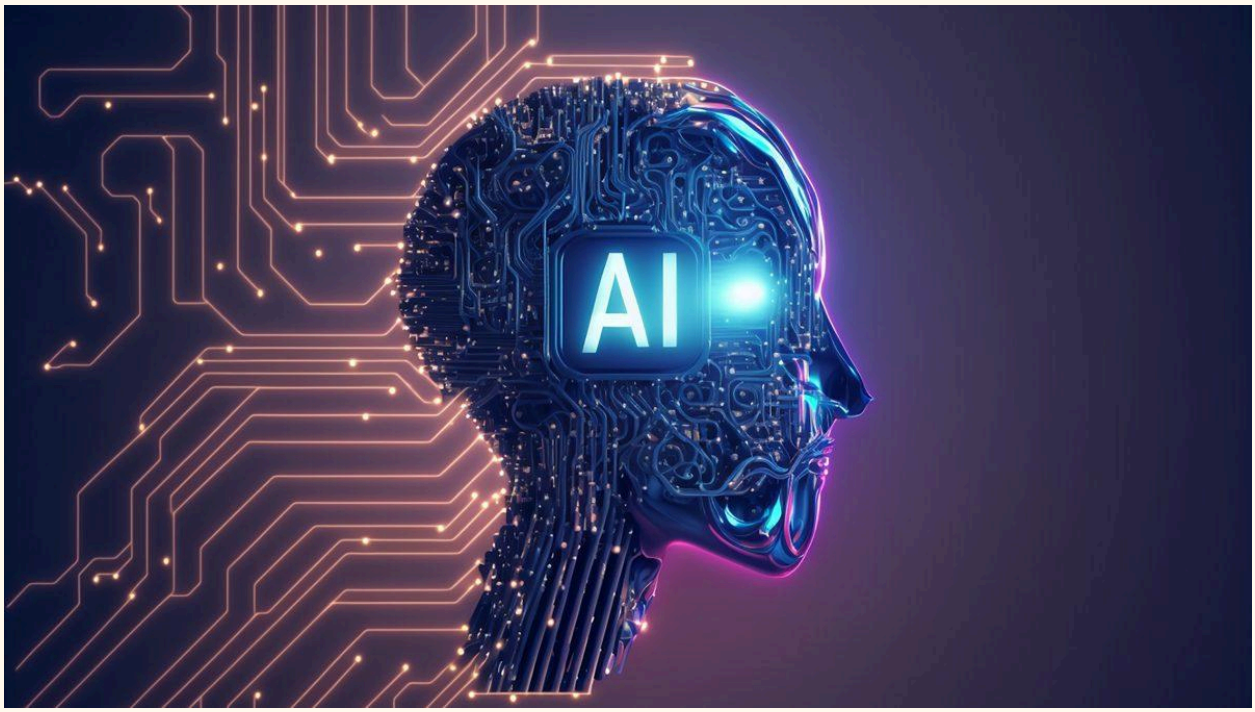
Rapport de TP3
Apprentissage supervisé

Université d'Avignon - Janvier 2025



Réalisé Par :

- Mounir Mouterfi
- Nadjim Ait Meddour

Proposé par : Juan-Manuel Torres

Partie 1 :

Le programme a pour objectif d'apprendre à un perceptron à classer des données en utilisant l'algorithme de recuit simulé (simulated annealing) pour ajuster les poids du modèle. L'objectif est de minimiser l'erreur de classification et de calculer la stabilité des points de test.

Structure du programme :

Cette classe représente le perceptron et gère l'apprentissage et la prédiction.

Méthodes principales de la classe :

heaviside: Fonction d'activation utilisée pour prédire les classes. Elle renvoie 1 si l'entrée est positive et -1 sinon.

tanh_derivative: Dérivée de la fonction hyperbolique tangente, utilisée pour la mise à jour des poids.

V: Fonction associée à la stabilité qui dépend de la tangente hyperbolique.

predict: Prédit la classe d'un exemple donné en utilisant la fonction d'activation de Heaviside.

fit: Entraîne le perceptron avec l'algorithme de recuit simulé, en mettant à jour les poids de manière itérative tout en réduisant la température pour éviter les minima locaux.

calculate_errors_and_stabilities: Calcule les erreurs d'apprentissage et de généralisation (sur les données de test) et la stabilité des points de test.

plot_stabilities: Affiche un graphique des stabilités des exemples de test.

Les données d'exemple (train_X, train_Y, test_X, test_Y) sont utilisées pour entraîner le perceptron et tester la performance du modèle.

À la fin de l'entraînement, les erreurs (d'apprentissage et de généralisation) et les poids du perceptron sont affichés, et un graphique des stabilités est tracé.

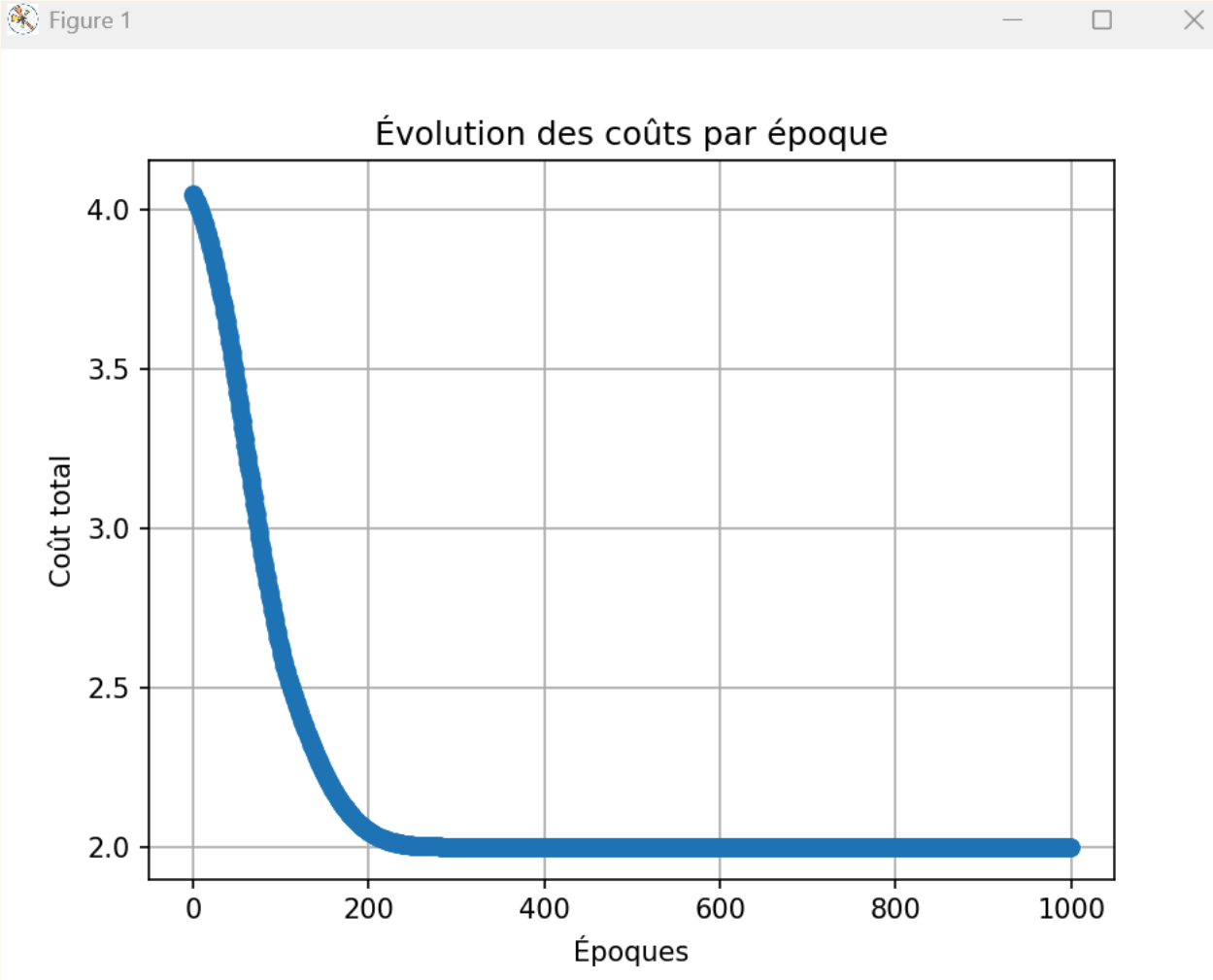
Résultat :

À la fin de l'exécution, le programme affiche :

- Les erreurs d'apprentissage et de généralisation.

```
[Running] python -u "m:\master_1_IA\TPMinimerro\minimerror\minimerrorPartie1.py"
Erreur d'apprentissage Ea: 25.00%
Erreur de g□n□ralisation Eg: 0.00%
```

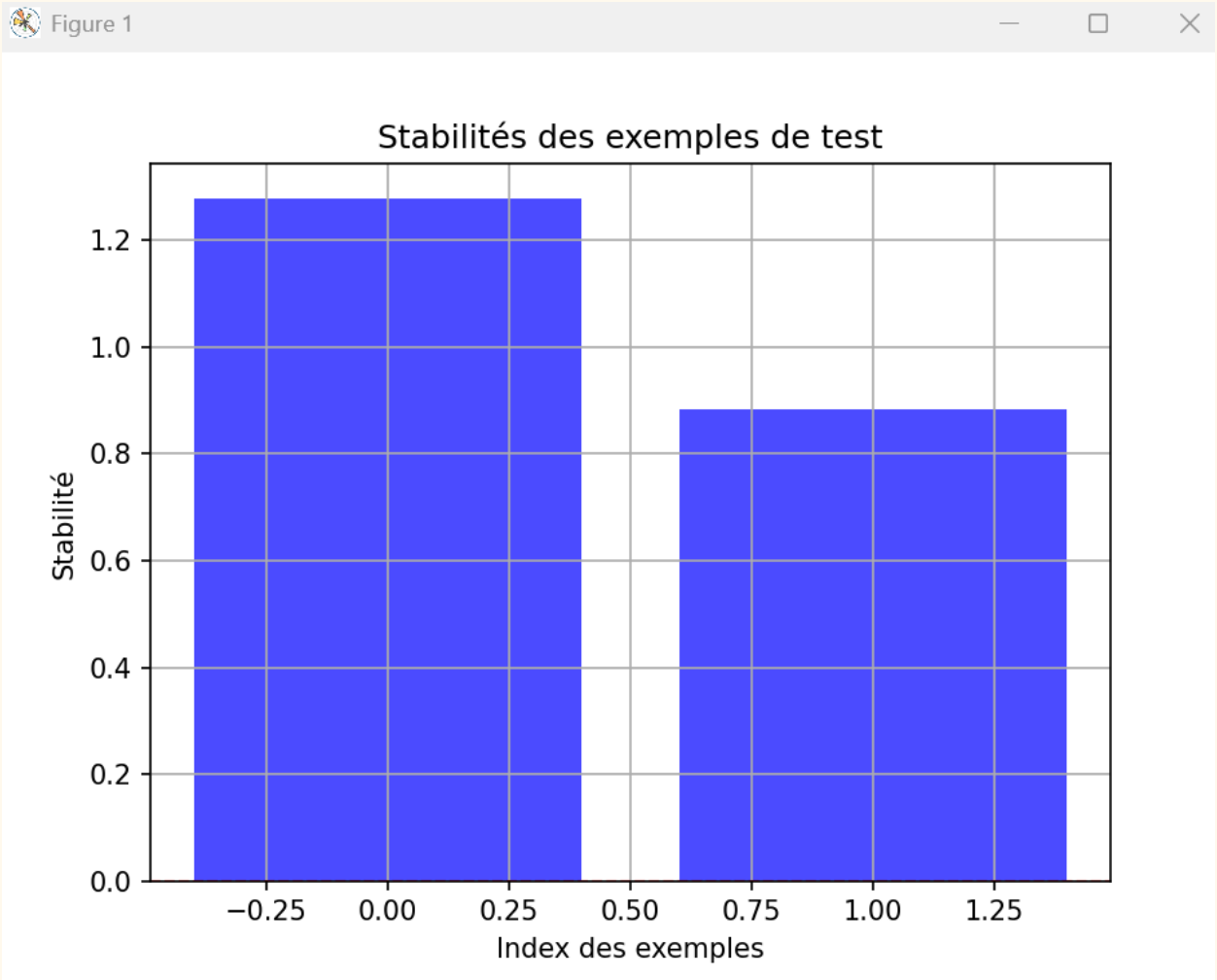
- L'évolution des coûts :



- Les poids finaux du perceptron.

```
Poids du perceptron (W):
[-0.53972936 -1.09696335 -0.71088931]
```

- Un graphique des stabilités des exemples de test.



Partie 2 :

Le programme implémente un perceptron pour la classification binaire avec un apprentissage basé sur le recuit simulé, utilisant deux températures distinctes (`beta_plus` et `beta_minus`). Le programme cherche à minimiser l'erreur de classification et à analyser l'impact de ces températures sur la stabilité du modèle.

Structure du programme :

Classe Minimerror :

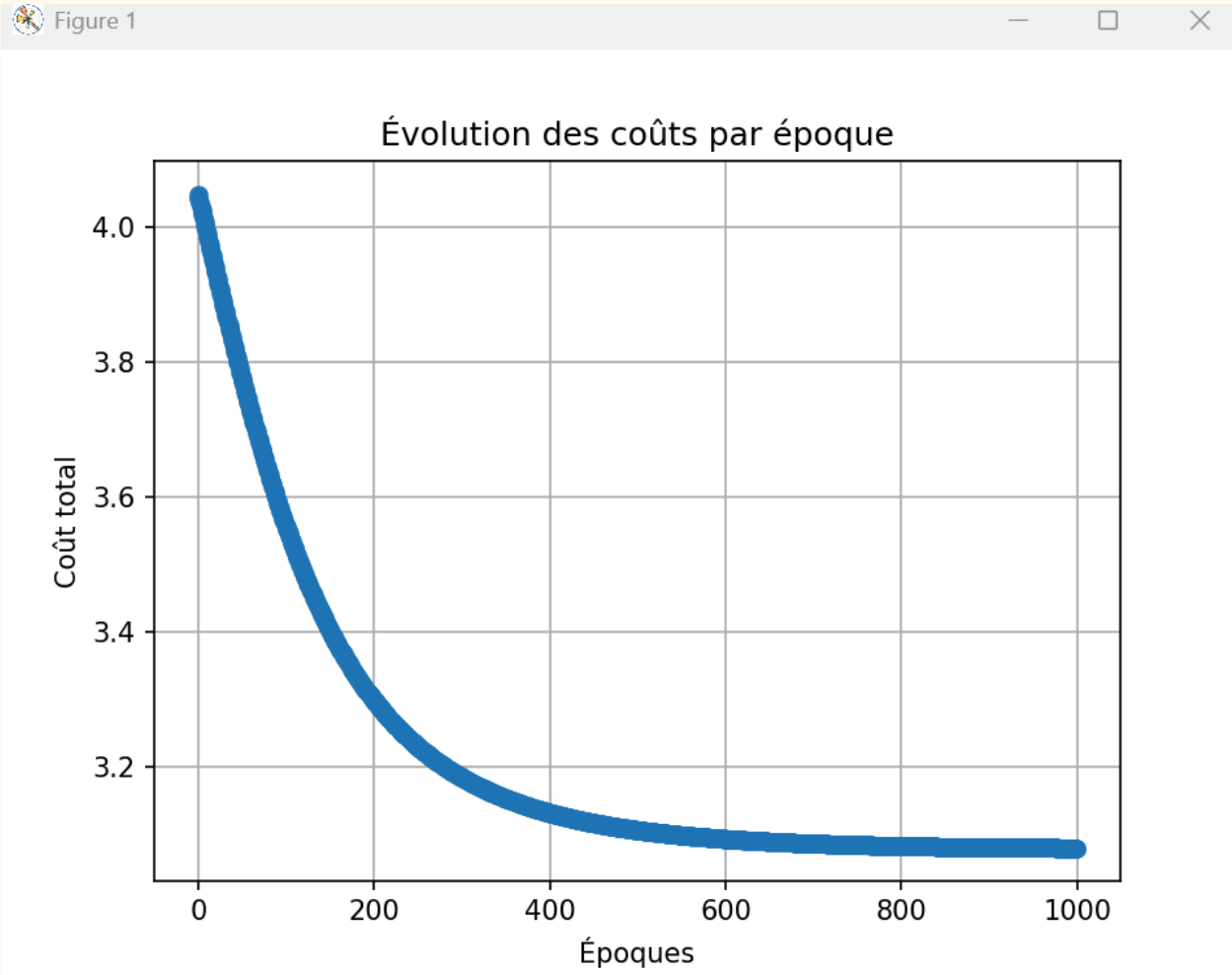
- Cette classe représente un perceptron et gère l'apprentissage et la prédiction.
- Les paramètres importants incluent les poids du perceptron, les températures **`beta_plus`** et **`beta_minus`**, ainsi que le taux d'apprentissage (`recuit`).

Méthodes principales de la classe :

- **`heaviside`**: Fonction d'activation utilisée pour la prédiction. Elle renvoie `1` si l'entrée est positive, sinon `-1`.
- **`tanh_derivative`**: Dérivée de la fonction tangente hyperbolique, utilisée dans le calcul du gradient de la fonction de coût.
- **`V`**: Fonction associée à la stabilité qui dépend de la tangente hyperbolique.
- **`predict`**: Prédit la classe d'un exemple donné, en appliquant la fonction d'activation.
- **`fit`**: Entraîne le perceptron avec l'algorithme de recuit simulé en utilisant les températures **`beta_plus`** et **`beta_minus`**. Les poids sont mis à jour à chaque époque en fonction de la stabilité calculée.
- **`save_weights`**: Sauvegarde les poids du perceptron dans un fichier CSV pour une utilisation ultérieure.

Résultat :

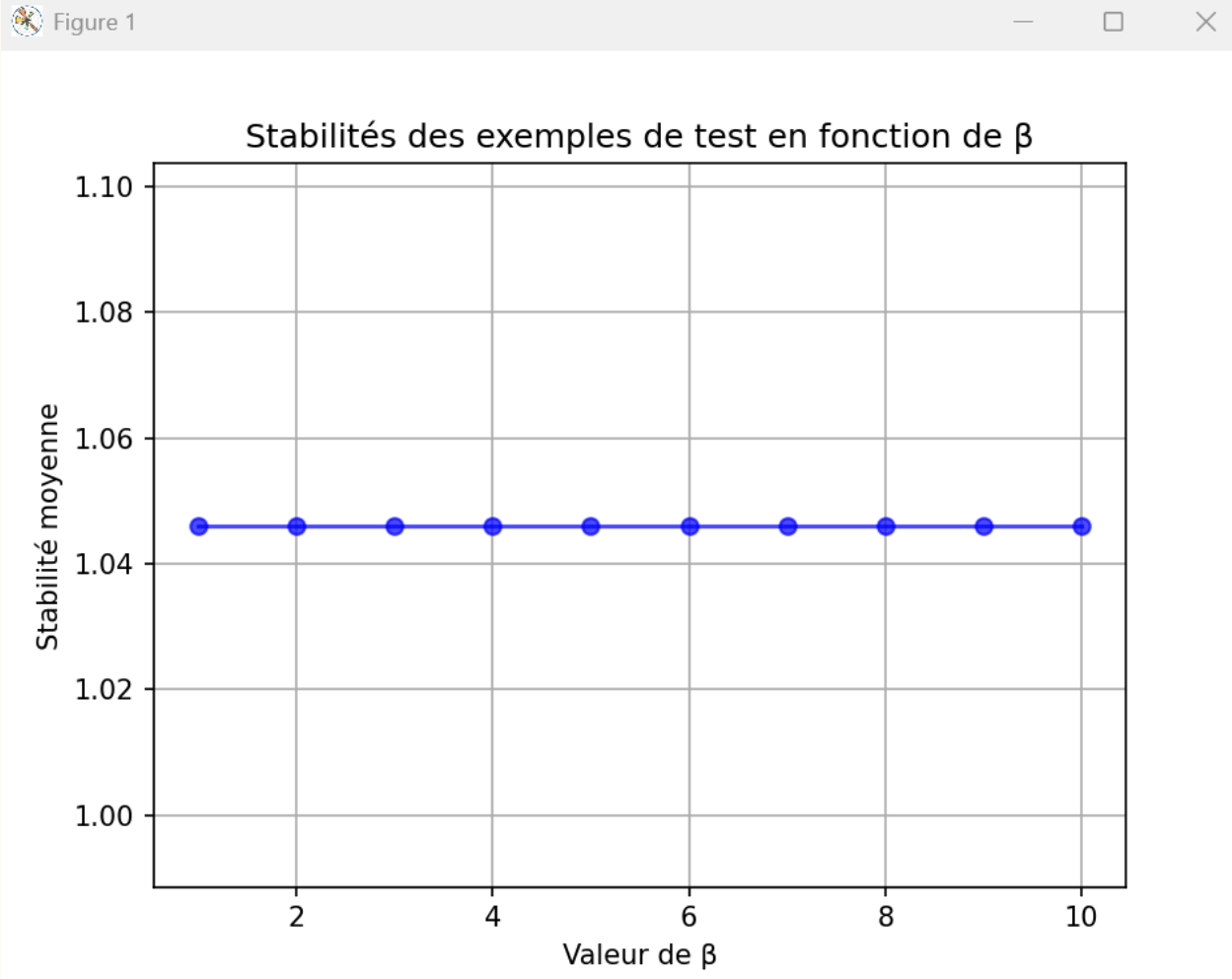
Evolution des coûts :



affichage des erreurs Ea et Eg :et les poids finaux (N+1) :

```
[Running] python -u "m:\master_1_IA\TPMinimerro\minimerror\minimerrorPartie2.py"
Erreur d'apprentissage Ea: 25.00%
Erreur de g�n�ralisation Eg: 0.00%
Poids du perceptron (W):
[-0.06356628 -0.07355506 -1.41086817]
```

les stabilités de test en fonction de beta :



et en fin, le stockage des valeurs de N+1 poids dans un fichier csv :