ECOLE D'INFORMATIQUE
SUP DE VINCI

Projet Deep Learning

Prédiction des maladies à l'aide de l'apprentissage profond

Réalisé par

Lisa SMAH

Année Universitaire : 2023/2024.

Introduction

Le présent projet se situe à l'intersection de deux domaines dynamiques qui sont : le Deep Learning et la médecine, mettant en œuvre des techniques de deep learning pour la prédiction de maladies à partir de symptômes spécifiques.

L'objectif fondamental de ce projet est d'explorer la capacité des modèles de deep learning à classifier les maladies en fonction d'un ensemble de symptômes donné. Pour ce faire, nous exploitons un ensemble de données exhaustif comprenant 132 paramétres représentant divers symptômes médicaux, avec un ensemble de 42 classes correspondant à différentes maladies.

Nous entamons ce rapport en présentant les détails du jeu de données utilisé, expliquant la nature des symptômes, la structure des classes de maladies, et en décrivant notre approche méthodologique pour construire un modèle de deep learning robuste. Cette exploration méthodique comprendra la préparation des données, la construction et l'entraînement du modèle, ainsi que l'évaluation de ses performances sur un ensemble de test indépendant.

Description des Données

L'ensemble de données qui alimente notre exploration provient de : Kaggle. Plus précisément, les données proviennent du jeu de données intitulé "Disease Prediction Using Machine Learning", disponible à l'adresse lien vers Kaggle.

L'ensemble de données est composé de deux fichiers CSV : l'un dédié à l'entraînement de notre modèle et l'autre destiné à tester ses capacités. Chacun de ces fichiers contient 133 colonnes, avec 132 colonnes représentant des symptômes spécifiques qu'une personne peut présenter, et la dernière colonne dévoilant le pronostic associé. Ce tableau de données se présente sous la forme d'une matrice de dimensions 4920x133 pour l'ensemble d'entraînement, témoignant de la richesse des informations à explorer.

Les symptômes, tels qu'ils sont encapsulés dans ces colonnes, sont des signaux variés qui balisent le chemin vers le diagnostic. De la simple fièvre aux indicateurs plus nuancés comme des éruptions cutanées ou des douleurs abdominales, chaque colonne incarne un aspect précis du tableau.

Prétraitements des Données

Encodage des étiquettes de classe :

Les étiquettes de classe représentant les maladies ont été encodées pour permettre une manipulation facile dans le cadre de l'apprentissage automatique. L'encodage a été réalisé à l'aide de la méthode LabelEncoder de la bibliothéque scikit-learn. Chaque maladie a été attribuée à un identifiant unique, facilitant ainsi le traitement et l'analyse des données.

```
# Créer un objet LabelEncoder
label_encoder = LabelEncoder()

# Adapter et transformer les étiquettes de classe dans Y_train
Y_train_encoded = label_encoder.fit_transform(Y_train)
```

Modèle de Deep Learning

Le modèle de Deep Learning utilisé pour la prédiction des maladies est basé sur une architecture de réseau de neurones artificiels. Il comprend 2 couches, une couche cachée et une couche de sortie. Les données d'entrée sont constituées de 132 entrées, représentant les différents symptômes médicaux du jeu de données. La couche de sortie comporte 42 sorties, correspondant aux classes des maladies prédites.

Initialisation des poids et biais :

Les poids et biais du modèle sont initialisés de manière à favoriser une convergence rapide du modèle pendant l'apprentissage. La méthode d'initialisation utilisée est celle des poids nuls, où les poids sont initialement fixés à zéro.

```
[7]  # Initialisation des poids et biais avec 128 neurones
     def initialisation(nb_neurone, nb_classes):
         w = np.zeros((y, nb_neurone))
         b = np.zeros((1, nb_neurone))
         v = np.zeros((nb_neurone, nb_classes))
         c = np.zeros((1, nb_classes))
         return w, b, v, c

         # 'w' et 'b' représentent les poids et le biais associés à la couche cachée.
         # 'v' et 'c' représentent les poids et le biais associés à la couche de sortie.
```

Fonctions d'activation :

Deux fonctions d'activation ont été utilisées dans les couches du réseau :

1. **Sigmoid :** La fonction sigmoide est utilisée dans la couche cachée.

```python
# Fonction d'activation sigmoid
def sigmoid(z):
    return 1 / (1 + np.exp(-z))
```

2. **Softmax :** La fonction softmax est utilisée dans la couche de sortie pour transformer les scores bruts en probabilités, facilitant ainsi l'interprétation des prédictions comme une distribution de probabilités sur les différentes classes.

```python
# Fonction d'activation softmax
def softmax(z):
    exp_z = np.exp(z - np.max(z))  # Pour la stabilité numérique
    return exp_z / exp_z.sum(axis=1, keepdims=True)
```

Propagation avant (forward propagation) et rétropropagation (backpropagation) :

La propagation avant est le processus par lequel les données d'entrée sont transmises à travers le réseau pour produire des prédictions. La rétropropagation est ensuite utilisée pour calculer le gradient de la fonction de coût par rapport aux poids du réseau, permettant ainsi d'ajuster ces poids pour minimiser la perte.

```python
def propagate(w, b, v, c, X, Y):
    m = X.shape[0]

    # Forward Propagation
    Z1 = np.dot(X, w) + b
    A1 = sigmoid(Z1)
    Z2 = np.dot(A1, v) + c
    A2 = softmax(Z2)

    # Calcul du coût
    cost = compute_cost(A2, Y)

    # Backward Propagation
    dZ2 = A2 - Y
    dv = 1/m * np.dot(A1.T, dZ2)
    dc = 1/m * np.sum(dZ2, axis=0, keepdims=True)
    dZ1 = np.dot(dZ2, v.T) * A1 * (1 - A1)
    dw = 1/m * np.dot(X.T, dZ1)
    db = 1/m * np.sum(dZ1, axis=0, keepdims=True)

    return dw, db, dv, dc, cost
```

Fonction de coût :

La fonction de coût utilisée est la perte logistique pour la couche cachée et la perte de cross-entropy pour la couche de sortie. Ces fonctions mesurent l'écart entre les prédictions du modèle et les étiquettes réelles du jeu de données.

```
# Fonction de coût
def compute_cost(A, Y):
    m = Y.shape[0]
    cost = -1/m * np.sum(Y * np.log(A))
    return cost
```

Optimisation des poids et biais (descente de gradient) :

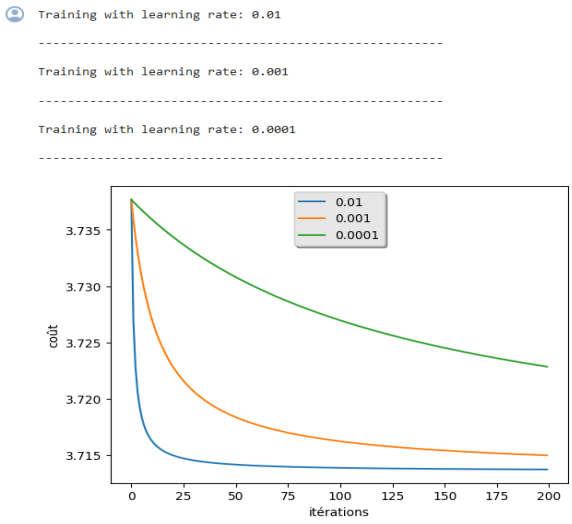
La descente de gradient est utilisée pour minimiser la fonction de coût en ajustant itérativement les poids et les biais du réseau. Le taux d'apprentissage est un paramétre crucial qui détermine la taille des pas effectués lors de la mise à jour des paramétres.

```
# Optimisation
def optimize(w, b, v, c, X, Y, num_iterations, learning_rate):
    costs = []

    for i in range(num_iterations):
        dw, db, dv, dc, cost = propagate(w, b, v, c, X, Y)

        # Mise à jour des poids et biais
        w -= learning_rate * dw
        b -= learning_rate * db
        v -= learning_rate * dv
        c -= learning_rate * dc

        # Enregistrement du coût à chaque 100 itérations
        if i % 100 == 0:
            costs.append(cost)

    return w, b, v, c, costs
```

Évaluation et Interprétation des Résultats : Analyse Approfondie des Performances du Modèle à Travers Divers Scénarios

Analyse de l'Impact des Taux d'Apprentissage sur la Convergence du Modèle :

Observation des résultats pour les trois taux d'apprentissage (0.01, 0.001, 0.0001) sur 200 itérations :



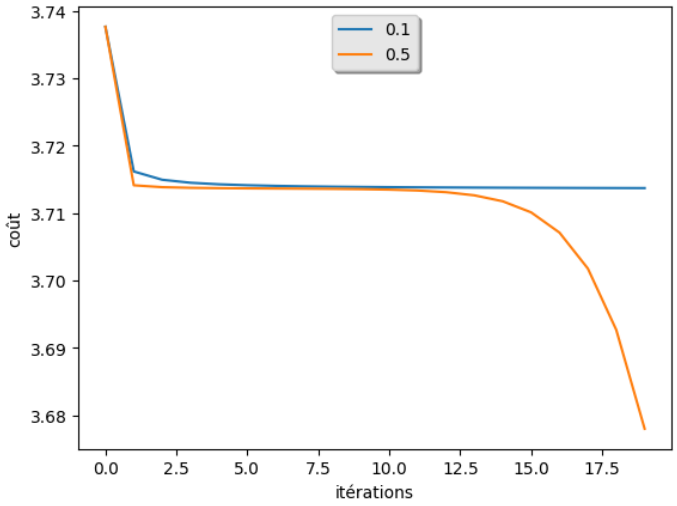
Inteprétaion :

— Un taux d'apprentissage de 0.01 semble être un compromis efficace entre une convergence rapide et une amélioration significative du modèle.

— Des taux d'apprentissage plus bas (0.001 et 0.0001) conduisent également à des améliorations, mais la convergence est plus lente.

Observation des résultats pour les deux taux d'apprentissage (0.1, 0.5) sur 2000 itérations :

```
Training with learning rate: 0.1

-------------------------------------------------------

Training with learning rate: 0.5

-------------------------------------------------------
```



Un taux d'apprentissage plus élevé (0.5 dans ce cas) semble être plus efficace, il permet au modèle de converger plus rapidement et de réduire la fonction de coût de manière plus efficace.

Analyse Comparative des Performances du Modèle : Impact du Nombre d'Itérations avec un Taux d'Apprentissage de 0.5

Nous avons évalué les performances du modèle en ajustant un paramètre clé : le nombre d'itérations. Voici les résultats pour chaque cas :

Cas 1 (30,000 itérations) :

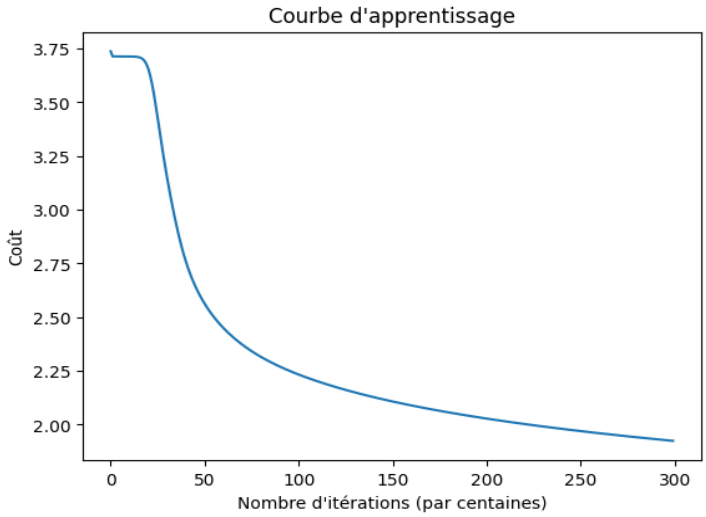
— Taux d'apprentissage : 0.5

— Précision sur l'ensemble d'entraînement : 43.66%

```
Train Accuracy: 43.66%
```

— Précision sur l'ensemble de test : 48.54%

— Coût : 2.0



Interprétation :

Le modèle a une faible précision tant sur l'ensemble d'entraînement que sur l'ensemble de test (43.66% et 48.54% respectivement). Le coût associé à ce cas est relativement élevé, indiquant que le modèle ne converge pas efficacement.

Cas 2 (60,000 itérations) :

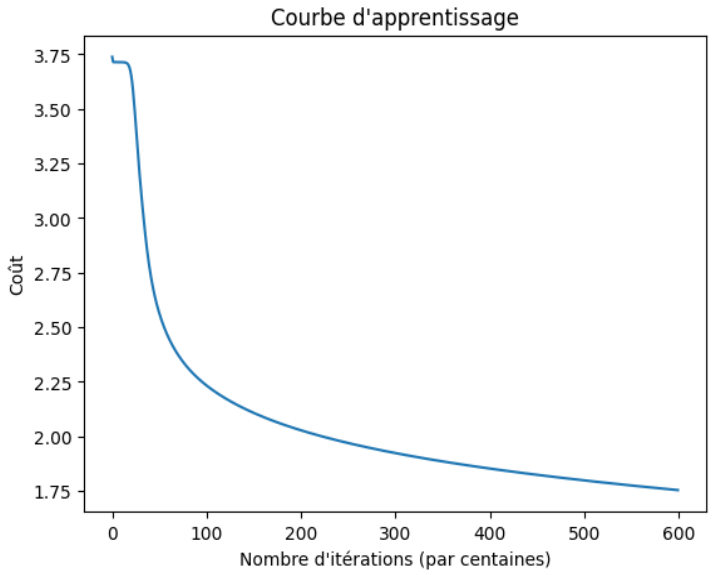
— Taux d'apprentissage : 0.5

— Précision sur l'ensemble d'entraînement : 48.54%

Précision sur l'ensemble d'entraînement : 48.54%

— Précision sur l'ensemble de test : 50.00%

— Coût : 1.75



Interprétation :

Bien que le nombre d'itérations ait augmenté, la précision reste modérée, montrant une amélioration légère par rapport au Cas 1.

Le coût diminue par rapport au Cas 1, suggérant une meilleure convergence, mais il reste relativement élevé.

Cas 3 (120,000 itérations) :

— Taux d'apprentissage : 0.5

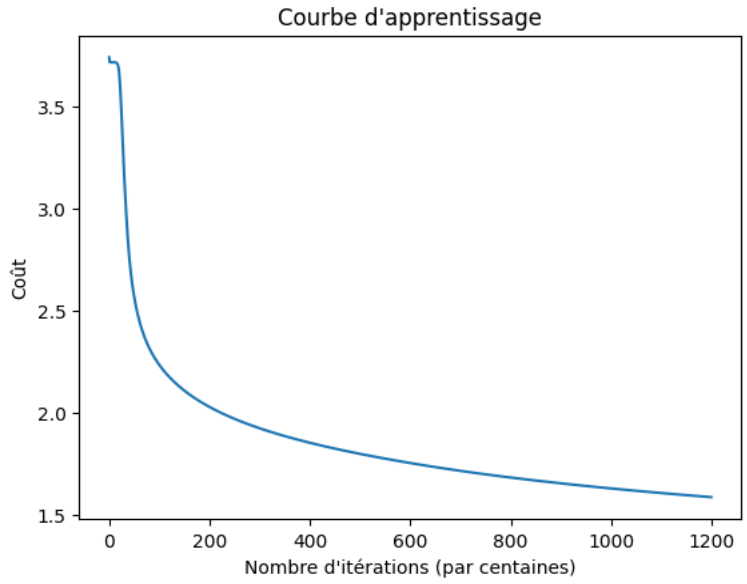
— Précision sur l'ensemble d'entraînement : 57.20%

```
Train Accuracy: 57.20%
```

— Précision sur l'ensemble de test : 61.90%

```
Précision sur l'ensemble de test : 61.90%
```

— Coût : 1.6



Interprétation :

Le modèle montre une amélioration significative avec une précision plus élevée sur les ensembles d'entraînement et de test (57.20% et 61.90% respectivement). Le coût a diminué, indiquant une convergence plus efficace et une meilleure adaptation du modèle aux données.

En résumé, augmenter le nombre d'itérations a un impact positif sur les performances du modèle, avec une amélioration notable observée dans le Cas 3.

Cas 4 (250,000 itérations) :

— Taux d'apprentissage : 0.5

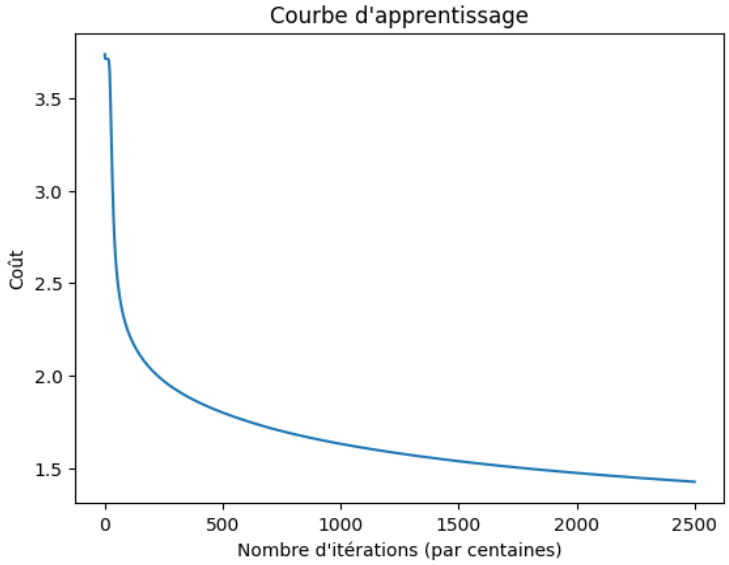
— Précision sur l'ensemble d'entraînement : 61.71%

```
Train Accuracy: 61.71%
```

— Précision sur l'ensemble de test : 66.67%

```
Précision sur l'ensemble de test : 66.67%
```

— Coût : 1.4



Interprétation :

— **Précision sur l'ensemble d'entraînement** (61.71%) **:** La précision de 61.71% sur l'ensemble d'entraînement suggère que le modèle a réussi à bien s'ajuster aux données utilisées pour son entraînement.

— **Précision sur l'ensemble de test** (66.67%) **:** Une précision de 66.67% sur l'ensemble de test indique que le modèle a une capacité acceptable à généraliser à de nouvelles données qu'il n'a pas rencontrées pendant l'entraînement.

— **Coût de 1.4 :** Un coût de 1.4 suggère un bon ajustement global du modèle aux données.

Le modèle semble avoir une performance raisonnable avec les paramètres spécifiques (taux d'apprentissage de 0.5 et 250,000 itérations) et démontre une capacité à généraliser aux données de test.

Résultats de Prédiction du Modèle sur l'Ensemble de Test

Après avoir entraîné notre modèle de Deep Learning sur un ensemble de données exhaustif comprenant 132 paramétres représentant divers symptômes médicaux, nous avons évalué ses performances sur un ensemble de test indépendant. Les résultats de prédiction, basés sur les probabilités calculées par le modèle, sont présentés ci-dessous pour deux jeux d'itérations : 120 000 itérations et 250 000 itérations

Résultats de Prédiction (120,000 Itérations)

Les résultats de prédiction ci-dessous, obtenus après 120 000 itérations d'entraînement du modèle, présentent les probabilités calculées pour chaque classe de maladie, ainsi que les maladies prédites correspondantes

```
[[3.96754045e-05 4.21128627e-03 7.92935450e-02 ... 4.76589593e-02
  8.28065270e-26 7.79282080e-10]
 [3.29594122e-02 1.60850345e-01 1.59019730e-05 ... 9.53512543e-02
  1.94692066e-19 8.86925894e-11]
 [2.60745355e-02 1.54644664e-01 3.30091885e-05 ... 1.10491823e-01
  9.31205722e-20 1.23229025e-10]
 ...
 [3.74041379e-09 2.86152774e-06 1.31117174e-01 ... 2.14681909e-04
  4.77096952e-32 2.20713789e-11]
 [1.26782657e-03 4.09491082e-02 7.03525164e-03 ... 1.48300305e-01
  5.70395754e-23 8.01133466e-10]
 [1.26441979e-02 1.25775250e-01 2.06767014e-04 ... 1.47457663e-01
  1.18874235e-20 2.66139630e-10]]
```

Les listes de probabilités fournissent des informations sur les maladies les plus probables pour chaque échantillon, tandis que les prédictions finales indiquent la

classe de maladie prédite.

```
[15  4 16 11  2 34  1 12  0  6 35 26 12 32 28 29  8 11 29 40 21 11 36
22  8 36 10 34 15 18 39 26 24 25 31 24  0 23 38 35 27 16]
```

Les noms de maladies prédites ont été extraits en utilisant un dictionnaire de correspondance entre les indices et les noms de maladies.

```
['Malaria', 'Drug Reaction', 'Chicken pox', 'Migraine', 'GERD',
'Osteoarthristis', 'Allergy', 'Cervical spondylosis', 'Fungal
infection', 'AIDS', 'Arthritis', 'Common Cold', 'Cervical spondylosis',
'Hyperthyroidism', 'Dimorphic hemmorhoids(piles)', 'Heart attack',
'Gastroenteritis', 'Migraine', 'Heart attack', 'Impetigo', 'Hepatitis
C', 'Migraine', '(vertigo) Paroymsal  Positional Vertigo', 'Hepatitis
D', 'Gastroenteritis', '(vertigo) Paroymsal  Positional Vertigo',
'Hypertension ', 'Osteoarthristis', 'Malaria', 'Typhoid', 'Psoriasis',
'Common Cold', 'Alcoholic hepatitis', 'Tuberculosis', 'Hypothyroidism',
'Alcoholic hepatitis', 'Fungal infection', 'Hepatitis E', 'Urinary
tract infection', 'Arthritis', 'Pneumonia', 'Chicken pox']
```

Interprétation

Les résultats montrent les probabilités prédites pour chaque classe de maladie, et les prédictions finales sont obtenues en choisissant la classe avec la probabilité la plus élevée.
La première maladie "Malaria" correspond à la maladie prédite pour le premier échantillon, elle a la plus haute probabilité. Pour le deuxième échantillon, la maladie prédite est "Drug Reaction" (réaction médicamenteuse), et ainsi de suite.
Ces résultats mettent en lumière les prédictions du modèle pour chaque échantillon de test, fournissant une indication des maladies les plus probables en fonction des symptômes fournis en entrée.

Résultats de Prédiction (250,000 Itérations)

Avec 250 000 itérations, les résultats de notre modèle montrent des probabilités associées à chaque classe pour un échantillon donné, suivi des prédictions associées.

Probabilités associées à chaque classe pour le premier échantillon :

```
[[1.29601275e-05 2.07145245e-03 1.58340763e-02 ... 5.97800165e-02
  2.39453899e-33 8.21638111e-13]
 [1.99714129e-02 1.63168093e-01 3.50241028e-07 ... 1.16622123e-01
  3.18976317e-26 6.95418594e-14]
 [7.84563126e-03 1.14638080e-01 4.69771383e-06 ... 1.68838630e-01
  2.12985960e-27 1.85923863e-13]
 ...
 [5.96231820e-14 2.94477845e-10 8.14947356e-02 ... 6.05870365e-07
  3.16298757e-46 3.10917093e-16]
 [4.85831395e-04 2.47829021e-02 5.77061907e-04 ... 1.72810349e-01
  2.91788865e-30 7.11773737e-13]
 [6.23466271e-03 1.03304868e-01 8.01131216e-06 ... 1.77900442e-01
  1.15365589e-27 2.23696196e-13]]
```

Les valeurs dans cette liste représentent les probabilités que l'échantillon appartienne à chaque classe. La classe avec la probabilité la plus élevée est la prédiction du modèle pour cet échantillon.

Prédictions associées :

```
[15  4 16  9 14 33 18 12  0  6 23 26  7 32 28 29  8 11 29 40 19  9 19 22
  8 36 28 34 14 18 39 26 24 25 31 24  0 23  2  7 27 16]
```

Chaque nombre dans cette liste correspond à la classe prédite. Par exemple, la première classe prédite est d'indice 15.

Noms des maladies prédites :

```
['Malaria', 'Drug Reaction', 'Chicken pox', 'Bronchial Asthma',
'Jaundice', 'Hypoglycemia', 'Typhoid', 'Cervical spondylosis', 'Fungal
infection', 'AIDS', 'Hepatitis E', 'Common Cold', 'Diabetes ',
'Hyperthyroidism', 'Dimorphic hemmorhoids(piles)', 'Heart attack',
'Gastroenteritis', 'Migraine', 'Heart attack', 'Impetigo', 'hepatitis
A', 'Bronchial Asthma', 'hepatitis A', 'Hepatitis D',
'Gastroenteritis', '(vertigo) Paroymsal  Positional Vertigo',
'Dimorphic hemmorhoids(piles)', 'Osteoarthristis', 'Jaundice',
'Typhoid', 'Psoriasis', 'Common Cold', 'Alcoholic hepatitis',
'Tuberculosis', 'Hypothyroidism', 'Alcoholic hepatitis', 'Fungal
infection', 'Hepatitis E', 'GERD', 'Diabetes ', 'Pneumonia', 'Chicken
pox']
```

En fonction de ces résultats, le modèle suggère que le premier échantillon appartient à la classe "Malaria".

On peut voir qu'une variation dans les prédictions du modèle entre les itérations de 120 000 et 250 000. Notamment, les classes prédites pour certains échantillons présentent des différences subtiles. Par exemple, à l'itération de 120 000, la classe prédite pour le 4e échantillon est "11", tandis qu'à l'itération de 250 000, la classe prédite est "9". De même, pour le 13e échantillon, la classe prédite passe de "35" à "23".

Conclusion

En conclusion, ce projet exploite la puissance du deep learning pour la prédiction de maladies à partir de symptômes spécifique. Les avancées réalisées ouvrent des possibilités passionnantes pour améliorer la précision du modèle et son utilité future dans le domaine médical. Ces progrès vont contribuer de manière significative à l'amélioration continue des applications médicales basées sur l'IA, offrant des solutions innovantes pour le diagnostic précoce et la prise en charge des patients.