

A thick dark blue vertical bar runs along the left edge of the page. A blue arrow-shaped banner points to the right from this bar, containing the date. Below the banner, several thin, curved lines in dark blue and light grey sweep upwards from the bottom left corner.

19/10/2024

Evaluation de Deep Learning

Professeur Djamal Seck

Samba SY

MASTER 2 BI 2023-2024

Table des matières

I	Introduction	2
II	Le modèle détection du cancer du sein	2
II.1	Objectif	2
II.2	Jeu de données.....	3
II.3	Prétraitement des données.....	3
II.4	Modèle utilisé	4
II.5	couches :.....	4
II.6	Entraînement et évaluation du modèle	4
II.7	. Résultats du modèle	4
II.8	Conclusions.....	5
II.9	Enregistrement du modèle.....	5
III	Le modèle de détection du diabète	5
III.1	Objectif	5
III.2	Jeu de données.....	5
III.3	Prétraitement des données.....	6
III.4	Modèle utilisé	6
III.5	Évaluation du modèle avec validation croisée	6
III.6	Résultats du modèle	7
III.7	Conclusions.....	7
III.8	Enregistrement du modèle.....	7
IV	Déploiement de l'Application avec Flask	8
IV.1	Objectif de l'application	8
IV.2	Environnement de développement	8
IV.3	Structure de l'application	9
IV.4	Fonctionnalités de l'application	9
IV.5	Implémentation.....	9
IV.5.1	Installation des dépendances.....	9
IV.5.2	Chargement des modèles.....	10
IV.5.3	Création des routes	10
IV.5.4	Exemple de code	10
IV.6	Interface utilisateur	12
IV.7	Déploiement.....	13
V	Conclusion.....	16

I [Introduction](#)

L'essor du Deep Learning a transformé de nombreux domaines, notamment la santé, où les modèles de réseaux de neurones offrent des solutions innovantes pour le diagnostic des maladies. Dans ce projet, nous explorons l'application des techniques de Deep Learning pour deux problématiques critiques : le diagnostic du cancer du sein et la détection du diabète, à partir des données issues de patients. Ces pathologies, qui touchent des millions de personnes à travers le monde, nécessitent des outils de diagnostic performants et accessibles, permettant aux professionnels de santé de prendre des décisions éclairées.

L'objectif de ce projet est double : d'une part, concevoir et valider des modèles de réseaux de neurones capables de prédire avec précision l'état de santé des patients à partir de jeux de données cliniques, et d'autre part, rendre ces modèles facilement accessibles via une application web conviviale. Pour cela, nous utilisons les bibliothèques Keras et TensorFlow, reconnues pour leur efficacité dans la construction de modèles de Deep Learning, ainsi que Flask ou Django pour le déploiement de l'application.

Les deux jeux de données utilisés, "Breast-cancer-Wisconsin" et "Pima-indians-diabetes", sont des standards dans la recherche sur ces maladies. Le premier vise à classifier des diagnostics entre bénins et malins, tandis que le second est conçu pour prédire l'apparition du diabète. En appliquant une démarche rigoureuse, depuis la division des données en ensembles d'apprentissage et de test jusqu'à l'évaluation des performances via des métriques clés comme l'accuracy, la précision, le rappel, et le F1-score, nous avons cherché à maximiser l'efficacité et la fiabilité des modèles.

Ce rapport présente non seulement le processus de construction et de validation des modèles, mais également les résultats obtenus et leur interprétation, ainsi que l'intégration finale dans une application web accessible et pratique. Grâce à cette approche, nous visons à offrir une solution technologique pouvant potentiellement aider à améliorer les diagnostics médicaux.

II [Le modèle détection du cancer du sein](#)

II.1 [Objectif](#)

L'objectif de ce projet est de développer un modèle d'apprentissage automatique pour prédire si une tumeur mammaire est bénigne (0) ou maligne (1) à partir des caractéristiques cliniques des tumeurs. Le modèle utilise un réseau de neurones artificiels entraîné sur le jeu de données Breast Cancer Wisconsin.

II.2 Jeu de données

Le jeu de données utilisé est le fichier CSV Breast-cancer-Wisconsin.csv. Les variables présentes dans le jeu de données sont les suivantes :

ID : Identifiant unique pour chaque échantillon.

diagnosis : Étiquette de diagnostic (B pour bénin, M pour malin).

radius_mean : Moyenne du rayon des tumeurs.

texture_mean : Moyenne de la texture des tumeurs.

perimeter_mean : Moyenne du périmètre des tumeurs.

area_mean : Moyenne de la surface des tumeurs.

smoothness_mean : Moyenne de la douceur des tumeurs.

compactness_mean : Moyenne de la compacité des tumeurs.

concavity_mean : Moyenne de la concavité des tumeurs.

concave points_mean : Moyenne des points concaves des tumeurs.

symmetry_mean : Moyenne de la symétrie des tumeurs.

fractal_dimension_mean : Moyenne de la dimension fractale des tumeurs.

(Et d'autres caractéristiques, au total 30).

II.3 Prétraitement des données

Les étapes suivantes ont été effectuées pour préparer les données avant l'entraînement du modèle :

Chargement et exploration des données : Le jeu de données a été chargé et des informations sur les données ont été affichées (aperçu, types, statistiques descriptives).

Vérification des valeurs manquantes : Aucune valeur manquante n'a été trouvée dans le jeu de données.

Étiquetage des diagnostics : Les étiquettes de la variable diagnosis ont été remplacées par des valeurs numériques (0 pour bénin, 1 pour malin).

Séparation des variables explicatives (X) et de la variable cible (y) : La variable diagnosis a été utilisée comme variable cible.

Standardisation des données : Les caractéristiques ont été normalisées à l'aide du StandardScaler pour assurer une échelle comparable entre les variables.

II.4 Modèle utilisé

Un réseau de neurones artificiels a été construit avec la bibliothèque Keras.
L'architecture du modèle comprend :

II.5 couches :

1ère couche dense avec 32 neurones et fonction d'activation ReLU.

2ème couche dense avec 16 neurones et fonction d'activation ReLU.

3ème couche de sortie avec 1 neurone et fonction d'activation sigmoïde (pour la classification binaire).

Dropout : Un taux de dropout de 0,5 a été utilisé pour éviter le surapprentissage (overfitting).

Fonction de perte : binary_crossentropy a été utilisée, car il s'agit d'une tâche de classification binaire.

Optimiseur : Adam a été utilisé pour la mise à jour des poids.

II.6 Entraînement et évaluation du modèle

Le modèle a été entraîné sur 100 époques avec un batch size de 10. Les résultats suivants ont été obtenus lors de l'évaluation du modèle sur l'échantillon de test.

II.7 Résultats du modèle

Le modèle a atteint une précision de 97 % sur l'échantillon de test.

```
Breast Cancer Wisconsin Classification Report:
              precision    recall  f1-score   support

      0       0.99         0.97         0.98         71
      1       0.95         0.98         0.97         43

   accuracy              0.97         114
  macro avg              0.97         114
weighted avg              0.97         114

Confusion Matrix:
[[69  2]
 [ 1 42]]
```

Classe 0 (Bénin) :

Précision : 99 % (sur toutes les prédictions de tumeurs bénignes, 99 % étaient correctes).

Rappel : 97 % (le modèle a détecté 97 % des cas réels de tumeurs bénignes).

F1-score : 0.98 (moyenne harmonique entre la précision et le rappel).

Classe 1 (Malin) :

Précision : 95 % (sur toutes les prédictions de tumeurs malignes, 95 % étaient correctes).

Rappel : 98 % (le modèle a détecté 98 % des cas réels de tumeurs malignes).

F1-score : 0.97.

69 tumeurs bénignes correctement classées (vrais négatifs).

2 tumeurs bénignes incorrectement classées comme malignes (faux positifs).

1 tumeur maligne incorrectement classée comme bénigne (faux négatifs).

42 tumeurs malignes correctement classées (vrais positifs).

II.8 Conclusions

Le modèle a très bien performé avec une précision globale de 97 %.

Le rappel pour les tumeurs bénignes (97 %) et malignes (98 %) est excellent, ce qui indique que le modèle identifie très bien les deux classes.

La faible quantité de faux positifs et de faux négatifs suggère que le modèle est robuste et fiable.

II.9 Enregistrement du modèle

Le modèle a été sauvegardé avec succès au format HDF5 :
model_breast_cancer.h5, permettant de le recharger pour des prédictions futures.

III Le modèle de détection du diabète

III.1 Objectif

L'objectif de ce projet est de créer un modèle d'apprentissage automatique capable de prédire si un individu est diabétique ou non, en se basant sur les caractéristiques fournies dans l'ensemble de données Pima Indians Diabetes Dataset. Le modèle utilise un réseau de neurones artificiels entraîné sur cet ensemble de données.

III.2 Jeu de données

Le jeu de données utilisé est le fichier CSV Pima-indians-diabetes.csv. Les variables présentes dans le jeu de données sont les suivantes :

Pregnancies : Nombre de grossesses.

Glucose : Concentration de glucose dans le plasma.

BloodPressure : Tension artérielle diastolique (mm Hg).

SkinThickness : Épaisseur du pli cutané du triceps (mm).

Insulin : Niveau d'insuline sérique (mu U/ml).

BMI : Indice de masse corporelle (poids en kg / (taille en m²)).

DiabetesPedigreeFunction : Hérité du diabète.

Age : Âge de l'individu.

Outcome : Résultat du diagnostic (0 = non diabétique, 1 = diabétique).

III.3 Prétraitement des données

Les étapes suivantes ont été effectuées pour préparer les données avant l'entraînement du modèle :

Séparation des caractéristiques (X) et de la variable cible (y) : Outcome a été utilisé comme variable cible.

Normalisation : Les caractéristiques ont été normalisées à l'aide du StandardScaler pour s'assurer que toutes les variables ont une échelle comparable, ce qui est important pour la performance du modèle.

III.4 Modèle utilisé

Un réseau de neurones artificiels a été construit avec la bibliothèque Keras. Le modèle comporte :

3 couches :

1ère couche dense avec 64 neurones et fonction d'activation ReLU.

2ème couche dense avec 32 neurones et fonction d'activation ReLU.

3ème couche de sortie avec 1 neurone et fonction d'activation sigmoïde (pour la classification binaire).

Dropout : Un taux de dropout de 0,3 a été utilisé pour éviter le surapprentissage (overfitting).

Fonction de perte : binary_crossentropy a été utilisée, car il s'agit d'une tâche de classification binaire.

Optimiseur : Adam a été utilisé pour la mise à jour des poids.

III.5 Évaluation du modèle avec validation croisée

Le modèle a été évalué à l'aide de la méthode de validation croisée KFold avec 5 plis, ce qui permet d'obtenir une évaluation plus robuste. Les résultats suivants ont été obtenus après l'entraînement sur 150 époques avec un batch size de 10.

III.6 Résultats du modèle

Le modèle a atteint une précision moyenne de 92 % après la validation croisée.

```
Rapport de classification :
              precision    recall  f1-score   support

     0         0.91         0.97         0.94         500
     1         0.94         0.82         0.88         268

 accuracy          0.92         0.92         0.92         768
 macro avg         0.93         0.90         0.91         768
 weighted avg      0.92         0.92         0.92         768

Matrice de confusion :
[[486  14]
 [ 48 220]]
```

Classe 0 (Non diabétique) :

Précision : 91 % (sur toutes les prédictions de non-diabétiques, 91 % étaient correctes).

Rappel : 97 % (le modèle a détecté 97 % des cas réels de non-diabétiques).

F1-score : 0.94 (moyenne harmonique entre la précision et le rappel).

Classe 1 (Diabétique) :

Précision : 94 % (sur toutes les prédictions de diabétiques, 94 % étaient correctes).

Rappel : 82 % (le modèle a détecté 82 % des cas réels de diabétiques).

F1-score : 0.88.

486 non-diabétiques correctement classés (vrais négatifs).

14 non-diabétiques incorrectement classés comme diabétiques (faux positifs).

48 diabétiques incorrectement classés comme non-diabétiques (faux négatifs).

220 diabétiques correctement classés (vrais positifs).

III.7 Conclusions

Le modèle a bien performé avec une précision globale de 92 %.

Le rappel pour les non-diabétiques (97 %) est excellent, ce qui signifie que le modèle identifie très bien cette classe.

III.8 Enregistrement du modèle

Le modèle a été sauvegardé avec succès en tant que fichier HDF5 :

mon_modele_diabete.h5, ce qui permet de le recharger pour des prédictions futures.

	Modèle 1 (Données Breast-cancer-Wisconsin)	Modèle 2 (Données Pima-indians-diabetes)
Accuracy	0.97	0.92
Précision	0.99 (classe 0), 0.95 (classe 1)	0.91 (classe 0), 0.94 (classe 1)
Rappel	0.97 (classe 0), 0.98 (classe 1)	0.97 (classe 0), 0.82 (classe 1)
F1-score	0.98 (classe 0), 0.97 (classe 1)	0.94 (classe 0), 0.88 (classe 1)

IV Déploiement de l'Application avec Flask

Rapport sur l'application Flask pour la détection du cancer du sein et la détection du diabète

IV.1 Objectif de l'application

L'application Flask a été développée pour fournir une interface utilisateur simple permettant de soumettre des données de caractéristiques pour la détection du cancer du sein et le dépistage du diabète. L'application vise à faciliter l'accès aux prédictions de santé, permettant ainsi une détection précoce et une intervention rapide.

IV.2 Environnement de développement

L'application a été développée en utilisant les technologies suivantes :

Flask : Un micro-framework Python pour le développement d'applications web.

TensorFlow/Keras : Utilisé pour charger et exécuter les modèles de détection du cancer du sein et de détection du diabète.

Pandas : Pour la manipulation et le traitement des données d'entrée.

HTML/CSS : Pour le front-end de l'application.

IV.3 Structure de l'application

L'application Flask se compose des principaux composants suivants :

app.py : Le fichier principal contenant la logique de l'application Flask.

templates/ : Dossier contenant les fichiers HTML pour l'interface utilisateur.

static/ : Dossier contenant les fichiers CSS et JavaScript pour le style et l'interaction de l'interface.

IV.4 Fonctionnalités de l'application

L'application permet aux utilisateurs de :

Entrer des données pour la détection du cancer du sein : Saisir les caractéristiques des tumeurs mammaires, telles que :

Rayon moyen

Texture moyenne

Périmètre moyen

Etc. (30 caractéristiques au total)

Obtenir des prédictions de malignité : Après avoir soumis les données, l'application prédit si la tumeur est bénigne ou maligne.

Entrer des données pour la détection du diabète : Saisir des données telles que :

Âge

IMC (Indice de Masse Corporelle)

Taux de glucose

Etc. (8 caractéristiques au total)

Obtenir des prédictions de diabète : Après avoir soumis les données, l'application prédit si l'utilisateur est susceptible de développer un diabète.

Afficher les résultats : Les résultats des prédictions sont affichés sur une nouvelle page, indiquant la malignité de la tumeur ou la probabilité de diabète.

IV.5 Implémentation

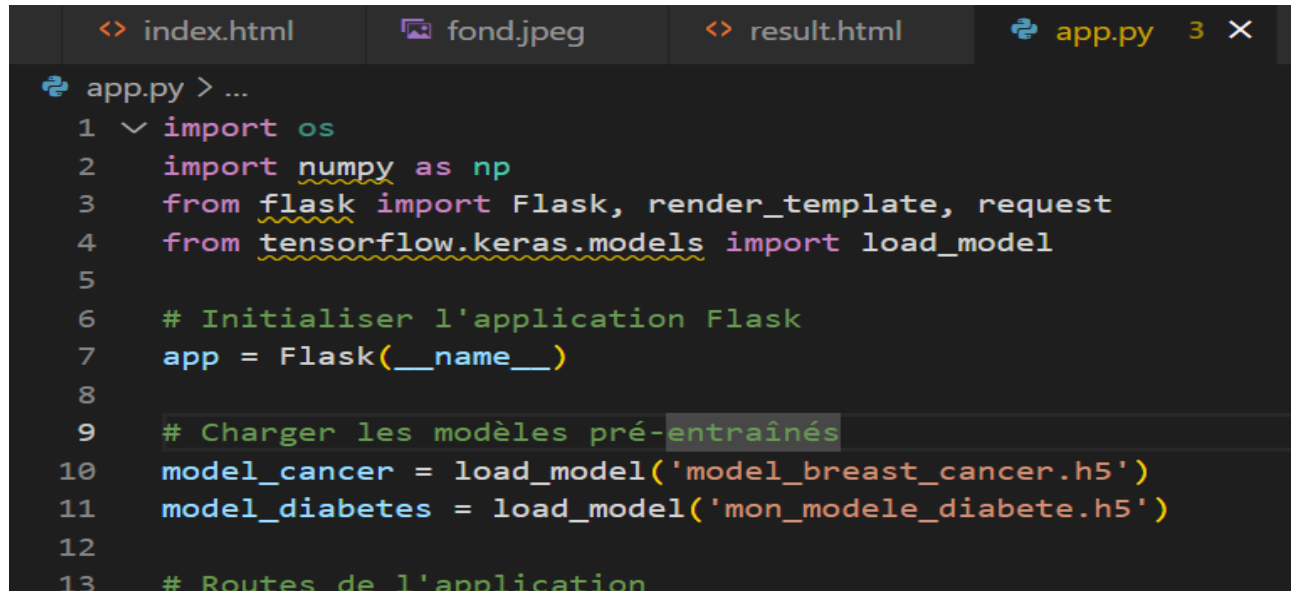
IV.5.1 Installation des dépendances

Avant de commencer, les dépendances nécessaires ont été installées :

`pip install Flask tensorflow pandas`

IV.5.2 Chargement des modèles

Les modèles pré-entraînés pour la détection du cancer du sein et la détection du diabète ont été chargés dans l'application Flask :



```
<> index.html  fond.jpeg  <> result.html  app.py 3 X
app.py > ...
1  import os
2  import numpy as np
3  from flask import Flask, render_template, request
4  from tensorflow.keras.models import load_model
5
6  # Initialiser l'application Flask
7  app = Flask(__name__)
8
9  # Charger les modèles pré-entraînés
10 model_cancer = load_model('model_breast_cancer.h5')
11 model_diabetes = load_model('mon_modele_diabete.h5')
12
13 # Routes de l'application
```

IV.5.3 Création des routes

Route principale (/) : Affiche le formulaire de saisie pour les deux types de données.

Route de prédiction pour le cancer (/predict_cancer) : Prend les données saisies pour la détection du cancer et appelle le modèle pour faire une prédiction.

Route de prédiction pour le diabète (/predict_diabetes) : Prend les données saisies pour la détection du diabète et appelle le modèle de détection du diabète.

IV.5.4 Exemple de code

Voici un extrait du code de l'application Flask :

```

# Routes de l'application
@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict_cancer', methods=['POST'])
def predict_cancer():
    try:
        # Récupérer les données du formulaire
        features = [
            float(request.form['mean_radius']),
            float(request.form['mean_texture']),
            float(request.form['mean_perimeter']),
            float(request.form['mean_area']),
            float(request.form['mean_smoothness']),
            float(request.form['mean_compactness']),
            float(request.form['mean_concavity']),
            float(request.form['mean_concave_points']),
            float(request.form['mean_symmetry']),
            float(request.form['mean_fractal_dimension']),
            float(request.form['radius_se']),
            float(request.form['texture_se']),
            float(request.form['perimeter_se']),
            float(request.form['area_se']),
            float(request.form['smoothness_se']),
            float(request.form['compactness_se']),
            float(request.form['concavity_se']),
            float(request.form['concave_points_se']),
            float(request.form['symmetry_se']),
            float(request.form['fractal_dimension_se']),
            float(request.form['radius_worst']),
            float(request.form['texture_worst']),

```

PORTS AZURE

```

    float(request.form['breast_dimension_morse']),
]

# Prédire avec le modèle
prediction = model_cancer.predict(np.array(features).reshape(1, -1))
result = "Malin" if prediction[0][0] > 0.5 else "Bénin"

return render_template('result.html', result=result, model="Cancer du Sein")

except Exception as e:
    return f"Une erreur s'est produite: {str(e)}"

.route('/predict_diabetes', methods=['POST'])
predict_diabetes():
try:
    # Récupérer les données du formulaire
    features = [
        float(request.form['pregnancies']),
        float(request.form['glucose']),
        float(request.form['blood_pressure']),
        float(request.form['skin_thickness']),
        float(request.form['insulin']),
        float(request.form['bmi']),
        float(request.form['diabetes_pedigree_function']),
        float(request.form['age']),
    ]

    # Prédire avec le modèle
    prediction = model_diabetes.predict(np.array(features).reshape(1, -1))

```

```

    # Prédire avec le modèle
    prediction = model_diabetes.predict(np.array(features).reshape(1, -1))
    result = "Diabétique" if prediction[0][0] > 0.5 else "Non Diabétique"

    return render_template('result.html', result=result, model="Diabète")

except Exception as e:
    return f"Une erreur s'est produite: {str(e)}"

if __name__ == '__main__':
    app.run(debug=True)

```

IV.6 Interface utilisateur

L'interface utilisateur a été conçue en utilisant HTML et CSS pour être intuitive et conviviale. Le formulaire permet aux utilisateurs de saisir les caractéristiques pour la détection du cancer et le dépistage du diabète. Les pages de résultats affichent les prédictions de manière claire et informative.

Prédiction du Diabète

Nombre de grossesses :

Niveau de glucose :

Pression artérielle :

Épaisseur de la peau :

Insuline :

Indice de masse corporelle (BMI) :

Fonction héréditaire du diabète :

Âge :

PRÉDIRE DIABÈTE

© 2024 Samba SY - Tous droits réservés

IV.7 Déploiement

L'application Flask a été déployée localement en utilisant :

`flask run`

Prédiction du Cancer du Sein et du Diabète

Prédiction du Cancer du Sein

Rayon moyen :

Texture moyenne :

Circonférence moyenne :

Surface moyenne :

Lissité moyenne :

Compacité moyenne :

Concavité moyenne :

Points concaves moyens :

Symétrie moyenne :

Dimension fractale moyenne :

Erreur de rayon :

Erreur de texture :

127.0.0.1:5000

Erreur de symétrie :

Erreur de dimension fractale :

Pire rayon :

Pire texture :

Pire circonférence :

Pire surface :

Pire lissité :

Pire compacité :

Pire concavité :

Pires points concaves :

Pire symétrie :

Pire dimension fractale :

PRÉDIRE CANCER DU SEIN

Formulaire de prédiction du cancer du sein

Résultat de Prédiction

Cancer du sein

Résultat : cancer du sein

[Faire une autre prédiction](#)

Resulat de la prediction du cancer du sein

Prédiction du Diabète

Nombre de grossesses :

0

Niveau de glucose :

13

Pression artérielle :

78

Épaisseur de la peau :

67

Insuline :

57

Indice de masse corporelle (BMI) :

24

Fonction héréditaire du diabète :

24

Âge :

34

PRÉDIRE DIABÈTE

Formulaire de prédiction du diabète

Résultat de Prédiction

Diabète

Résultat : Diabétique

[Faire une autre prédiction](#)

Résultat de prédiction du diabète

Prédiction du Diabète

Nombre de grossesses :

0

Niveau de glucose :

13

Pression artérielle :

78

Épaisseur de la peau :

67

Insuline :

4

Indice de masse corporelle (BMI) :

24

Fonction héréditaire du diabète :

24

Âge :

34

PRÉDIRE DIABÈTE

Résultat de Prédiction

Diabète

Résultat : Non Diabétique

[Faire une autre prédiction](#)

V Conclusion

Ce projet démontre l'impact potentiel de l'apprentissage automatique dans le domaine de la santé, offrant des solutions innovantes pour le diagnostic précoce et la prise de décision. En continuant à affiner le modèle et en explorant de nouvelles techniques, ce système pourrait jouer un rôle crucial dans l'amélioration des soins aux patients.