Rapport Projet Zoidberg

Sommaire

Introduction

Le projet Zoidberg:

Le projet Zoidberg a pour objectif de détecter les pneumonies aux travers de radio de patients. Pour réaliser cela, nous bénéficions d'un jeu de données(dataset) contenant des radiographies labellisées de personnes saines et malades.

Le rendu final du projet est un input permettant d'uploader une radiographie et de déterminer si le patient a une pneumonie ou non.

Méthodologie de travail

Introduction:

Pour réaliser ce projet, nous sommes une équipe de 3 personnes, composée de M. Mahamat ABAKAR-HASSAN, M. Florian TORCHY et M. Hamdi NASSRI.

Au vu du temps restreint alloué à ce projet nous avons fait certains choix pour mener à bien ce projet:

- Nous limiter à 1 algorithme par personne
- Développer chacun notre manière de transformer les données et échanger pour trouver la meilleure manière de traiter les données
- Favoriser les échanges au sein et à l'extérieur du groupe

Pour le choix des algorithmes, nous nous sommes basé sur les différentes documentations existantes, notamment la map de scikit learn (cf:https://scikit-learn.org/stable/tutorial/machine_learning_map/), qui fût d'une grande aide pour délimiter le périmètre des possibles choix algorithmiques.

Transformation des données:

Le cœur du sujet est la transformation des données du dataset. Nous avons fait le choix de développer cette partie là chacun de notre côté pour ensuite rassembler nos trouvailles. On a tout de même chercher à respecter la même structure de transformation des données en effectuant les actions suivantes:

- Préprocesser les images(redimensionner, extraire les labels
- Standardiser les images
- Réduire les dimensions

1 - Preprocessing des images

Cette partie là consiste dans la transformation d'images brutes en appliquant divers filtres pour essayer d'obtenir un format exploitable et intéressant à analyser. Cette étape est nécessaire avant la standardisation. Les transformations que l'on a principalement effectué sont : le redimensionnement de l'image, le passage en gris, l'extraction des labels des titres de l'image.

2 - Standardisation

La standardisation transforme les valeurs d'une variable pour qu'elles aient une moyenne de 0 et un écart-type de 1. Contrairement à la normalisation, la standardisation ne fixe pas de plage spécifique pour les valeurs transformées. La standardisation est utile lorsque les variables ont des échelles très différentes, et elle permet de centrer les données autour de zéro et de les mettre à l'échelle par rapport à l'écart-type, ce qui peut faciliter l'interprétation des coefficients dans certains modèles.

3 - Réduction de dimensions

La réduction de dimensionnalité est une technique utilisée pour simplifier les données complexes avant qu'elles ne soient traitées par l'apprentissage automatique.

Il s'agit du processus consistant à mapper les variables ou les caractéristiques présentes dans un jeu de données à un espace à plus faible dimension.

Cette technique permet aux algorithmes d'IA de calculer plus rapidement et d'obtenir des résultats plus précis en supprimant les variables corrélées sans compromettre la précision.

Pourquoi utiliser la réduction de dimension?

Dans un jeu de données volumineux, il peut y avoir une corrélation entre certaines variables qui peuvent être supprimées sans compromettre la précision et la qualité des résultats obtenus par l'apprentissage automatique. La réduction de dimensionnalité permet aux algorithmes d'IA d'effectuer plus rapidement leurs calculs et d'obtenir des résultats plus précis.

Les modèles utilisés

1 - SGDClassifier

SGDClassifier(Stochastic Gradient Descent Classifier) est un modèle de machine learning qui utilise divers modèle de classification linéaire tel que la régression logistique ou SVM avec comme méthode d'apprentissage la descente de gradient stochastique. Pour obtenir de meilleurs résultats, il faut standardiser les données.

Cet algorithme bénéficie d'un large éventail d'hyperparamètres. Nous avons fait le choix de concentrer nos recherches sur 3 d'entre eux :

- loss: c'est le paramètre qui permet de définir quel type de classification nous souhaitons, par exemple si nous prenons la valeur par défaut 'hinge', le modèle va utiliser SVM comme type de modèle
- alpha: c'est une constante qui multiplie la régularisation, elle a un impact sur la learning rate si celle ci est set sur 'optimal'(ce qui est notre cas)
- penalty: c'est la régularisation, celle ci même qui est impacté par alpha, son but est d'éviter l'overfitting ou "sur apprentissage" en français

Ce choix d'hyperparamètres nous as permis de maximiser nos potentiels résultats tout en limitant le nombre d'itérations

Lien de la documentation:
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

2 - SVC

Le Support Vector Classifier (SVC) est un algorithme de classification puissant basé sur les Support Vector Machines (SVM). Il est particulièrement efficace pour les problèmes de classification binaire.

Avantages de SVC

1. Efficacité en haute dimension:
2. Flexibilité grâce aux noyaux:
    - Les SVC peuvent utiliser différentes fonctions noyau (linéaire, RBF, polynomiale) pour trouver une frontière de décision optimale dans des espaces de dimensions supérieures.
3. Généralisation:
    - En maximisant la marge entre les classes, les SVC tendent à mieux généraliser sur les données non vues, réduisant ainsi le risque de surapprentissage.

Explication des hyperparamètres:

1. `C`

- **Description**: Contrôle le compromis entre une marge de séparation large et une erreur de classification minimale.
- **Valeurs**:
    - **Valeur faible**: Marge plus large, tolère plus d'erreurs (moins de surapprentissage).
    - **Valeur élevée**: Moins d'erreurs, marge plus étroite (risque de surapprentissage).

2. `kernel`

- **Description**: Détermine la fonction noyau utilisée pour transformer les données dans un espace de dimensions supérieures.
- **Types courants**:
    - `linear`: Noyau linéaire, pour données linéairement séparables.
    - `rbf` (Radial Basis Function): Noyau gaussien, pour capturer des relations complexes et non linéaires.

3. `gamma`

- **Description**: Définit l'influence d'un seul exemple d'entraînement
- **Valeurs**:
    - **Valeur faible**: Influence plus large, frontière de décision plus lisse.
    - **Valeur élevée**: Influence restreinte, frontière plus complexe (risque de surapprentissage).

Documentation:

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

3 - Régression logistique

Définition : La régression logistique est un modèle de classification binaire qui utilise une fonction sigmoïde pour estimer la probabilité qu'un échantillon appartiennent à une classe donnée.

Comment on a procédé :

- Préparation des Données :
  Nous avons défini les chemins des répertoires contenant les images et les labels associés. Chaque image a été prétraitée en niveaux de gris, redimensionnée à une taille de 100x100 pixels.. Cette étape est hyper importante pour l'extraction et la standardisation qui permettra d'extraire les caractéristiques d'une image.

- Extraction et Standardisation des Caractéristiques :
  Après avoir extrait et standardisé les images, nous avons utilisé le PCA pour réduire la dimensionnalité des données. Cela permet de simplifier les données tout en conservant l'essentiel de l'information, rendant ainsi les calculs plus efficaces et plus faciles pour l'algorithme.
- Entraînement du Modèle :
  Nous avons ensuite divisé les données en ensembles d'entraînement et de test et appliqué la régression logistique avec une recherche sur grille pour optimiser les hyperparamètres. Ce processus implique de tester différentes combinaisons de paramètres pour trouver les meilleurs pour notre modèle.

- Résultats et Évaluation :
  En utilisant la classe classification_report ça nous a permis d'évaluer plus facilement la presicion et l'évaluation de notre algorithme.