Rapport UE Introduction à l'Apprentissage Automatique Projet Classification d'Images

BENKORTBI Abdelhak (L3 informatique, Site de Luminy)
BELHARET Anis (L3 informatique, Site de Luminy)
OUKASSOU Anas (L3 informatique, Site de Luminy)
HADDADI Mazigh (L3 informatique, Site de Luminy)

"The Data Wizards"

<Meilleur score estimé par validation croisée: 0.79>

<Meilleur score obtenu sur images tests de mi-parcours: 0.73>

1.Introduction:

L'intelligence artificielle est un domaine en cours de développement et d'évolution: grâce à cette unité d'enseignement nous avons eu la chance de découvrir une branche de l'intelligence artificielle nommée l'apprentissage automatique, qui est devenu un sujet clé de l'intelligence artificielle et de la science de données. Dans le cadre de cet UE, nous avons abordé un projet de classification d'images prises par un drone. Notre objectif dans ce projet est de déterminer automatiquement si une image donnée contient une zone maritime ou non.

Pour cela, nous avons utilisé des techniques existantes et une étude des modèles et des algorithmes afin de mieux comprendre comment ceci fonctionne.

Cependant, durant la réalisation de ce projet, nous avons confronté certains défis que nous avons pu gérer, notamment, la documentation et la recherche de l'information qui peuvent être utile à l'avancement, ainsi que la sélection du modèle et les hyperparamètres des algorithmes de classification ainsi que la représentation des données.

En ce qui concerne l'organisation, nous avons commencé par répartir les tâches de sorte que chacun d'entre nous ait codé deux fonctions. Durant la réalisation de ce projet y'en avait certaines fonctions ou nous avons entamé la méthode de l'intelligence collective et une personne entre nous prendra le clavier afin de combiner l'idée ou nous sommes d'accord. Vers la fin du projet cette méthode nous a beaucoup aidé à décortiquer les sous problèmes rencontrés. On avait tous le même rôle qui était de programmer et de réaliser ce projet, on s'entraide de manière mutuelle que chacun de nous a recours aux idées des autres afin d'avancer sur ce projet.

2. Pré-traitement des données:

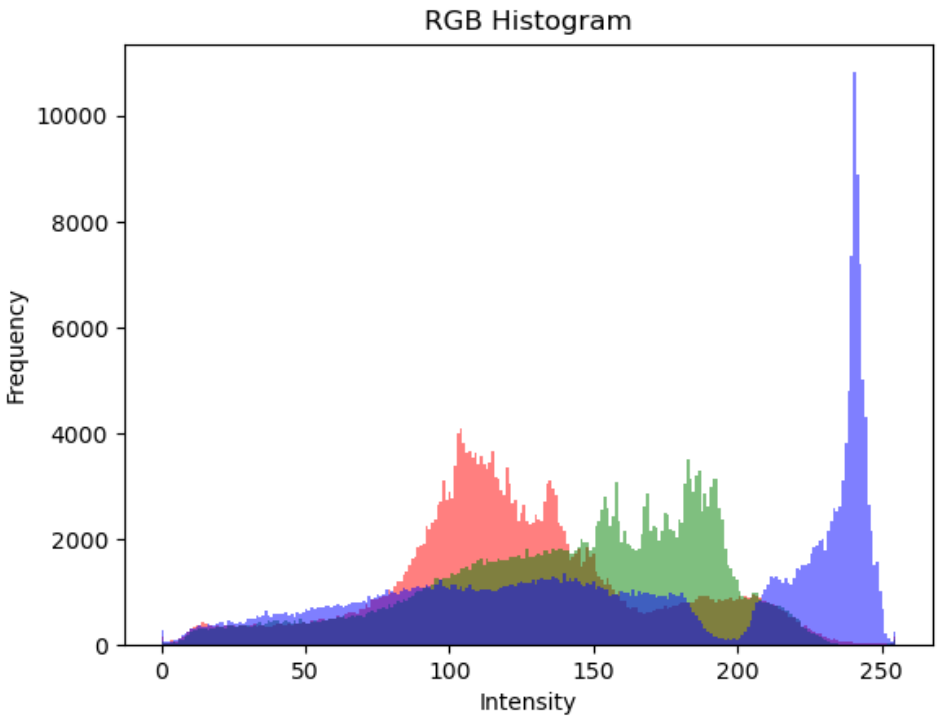
2.1. Représentation des données:

Dans la représentation des données, nous avons utilisé la bibliothèque Pillow de python qui facilite l'utilisation et le traitement de l'image.

Après avoir lu la documentation de cette bibliothèque nous avons représenté les images d'entraînement et de test par des tableaux numpy de pixels de taille 256x256, chaque pixel est représenté par une valeur entre 0 et 255. Cette représentation prend un temps remarquable pour la classification d'image et l'apprentissage de notre modèle

Le deuxième type de représentation est celui de mettre les images en noir et blanc (niveau de gris). En effet, cette méthode donne des résultats de prédiction plus rapide que les autres representations vu que la matrice des pixels contient du noir et du blanc.

Le troisième type de représentation est la représentation en histogramme de couleurs. Ceci nous stocke toutes les valeurs des pixels comptés de notre image dans un tableau Numpy.



Histogramme de couleurs

Image Pixel

Image en Gris

2.2. Augmentation des données:

En termes d'amélioration des données, nous retournons l'image verticalement, pour permettre au modèle de faire la distinction entre la mer et le ciel car ces deux éléments partagent des caractéristiques telle que la couleur. Nous gagnons ainsi en taille des données d'apprentissage.

Le fait qu'il y a du bleu dans la partie inférieure de l'image ne veut pas dire que cette dernière contient une zone maritime.

2.3. Eventuels commentaires:

Il y a la possibilité d'appliquer d'autres augmentations comme, mettre des filtres de bleu, ainsi que changer la saturation et le contraste des images pour augmenter la taille des données mais aussi pour permettre au modèles de bien classifier les images sous différentes conditions.

3. Algorithme(s) d'apprentissage considéré(s):

3.1. Algorithme retenu:

L'algorithme d'apprentissage utilisé pour calculer le score affiché est le Support Vector Machine (SVM).

Nous l'avons choisi après avoir examiné différentes options, telles que les arbres de décision, Les K Plus proches voisins (KNN), Random Forest…

3.2. Explication de(s) algorithme(s) retenu(s):

La machine à vecteurs de support (SVM) est une méthode d'apprentissage supervisé qui peut être utilisée pour classer des données en deux classes sur la base d'un hyperplan.

Le SVM est un algorithme de classification linéaire qui peut également être utilisé pour des problèmes de classification non linéaire en utilisant des fonctions.
Une recherche en grille a été effectuée en utilisant Grid Search CV pour trouver la valeur optimale des trois hyperparamètres C, Kernel et Degree. Il s'est avéré que les

paramètres par défaut donnent un meilleur score que toute autre combinaison de paramètres que nous avons essayée. Nous avons donc gardé le classifieur svm.SVC avec ses paramètres par défaut.

Les raisons pour lesquelles nous avons choisi cet algorithme sont les suivantes:

- l'efficacité de classifier les données, en particulier lorsque ces dernières sont bien séparables linéairement.
- performances élevées en termes de précision et vitesse de calcul.
- il ne nécessite pas beaucoup de pré-traitement

3.3. Alternatives:

D'autres algorithmes de classification ont été envisagés (Les K Plus proches voisins (KNN), Random Forest).

Toutefois, ces algorithmes n'ont pas donné d'aussi bons résultats que l'algorithme Support Vector Machine pour notre ensemble de données.

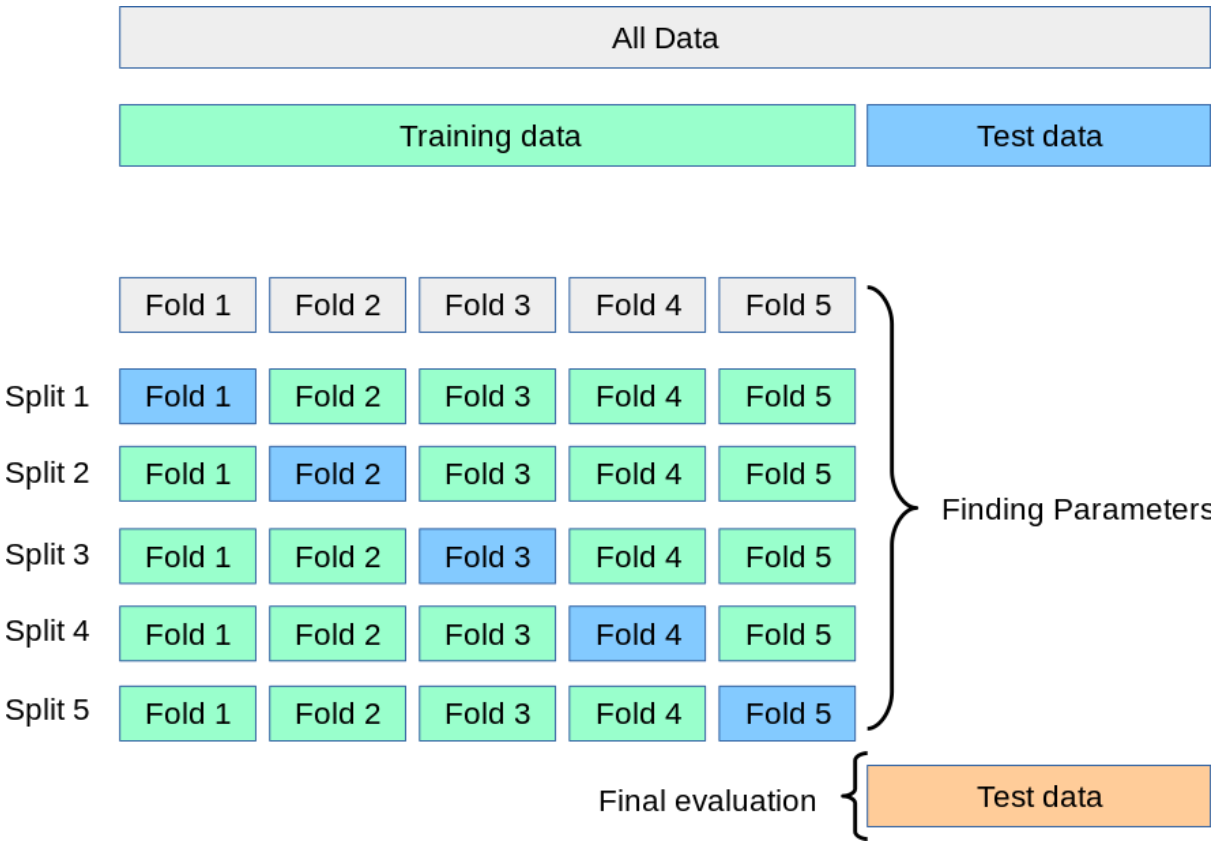
4. Evaluation des performances des classifieurs:

4.1. Protocole d'estimation des performances:

Dans notre étude, nous avons choisi comme critère de performance l'accuracy, qui mesure la proportion de prédictions correctes par rapport à l'ensemble des prédictions.

Nous avons opté pour ce critère car il est facile à interpréter et à comprendre. Nous avons utilisé la méthode de validation croisée k-fold (k=5), où les données sont divisées en 5 sous-ensembles égaux, et chaque sous-ensemble est utilisé une fois comme ensemble de validation tandis que les 4 autres sont utilisés pour l'entraînement.

Cette méthode permet d'estimer la performance du modèle de manière robuste en évitant un surapprentissage (overfitting) aux données d'entraînement.



4.2. <u>Performances obtenues:</u>

La meilleure approche trouvée a été le modèle SVM avec les hyper paramètres par défaut.

Sur les données d'entraînement, nous avons obtenu un score moyen de 0.79. Tandis que sur les données de test données à mi-parcours, nous avons obtenu un score de 0.73.

L'écart entre l'estimation et la mesure sur les données de test peut être dû à plusieurs raisons.

Tout d'abord, les données de test peuvent contenir des exemples qui diffèrent considérablement des exemples d'entraînement, et par conséquent, le modèle peut avoir plus de mal à les classifier.

En outre, le nombre de données de test peut être relativement faible par rapport à celui des données d'entraînement, ce qui peut entraîner des estimations moins précises.

4.3. Éventuelles courbes de performance:

Nous n'avons pas établi de courbes de performances pour ce projet, car nous n'avons pas modifié incrémentalement la taille des images ou des hyperparamètres. Cependant, si des courbes de performances avaient été établies, nous les aurions présentées ici et commentées en détail.

Nous avons utilisé la validation croisée k-fold pour estimer les performances de manière robuste.

Cependant, l'écart entre l'estimation et la mesure sur les données de test peut être dû à plusieurs raisons, telles que la différence de distribution entre les données d'entraînement et les données de test, ou la faible taille de l'ensemble de données de d'entraînement.

5. Résultats obtenus:

5.1. Pipeline complet:

Le pipeline complet d'évaluation est le suivant :

- Chargement des données :

  Charger les données d'entraînement et redimensionner les images.

- Transformation des images :

  Inverser les images verticalement et transformer les images en représentation tenseur de pixel.

- Apprentissage du modèle
- Estimation des performances du modèle sur l'ensemble d'apprentissage à l'aide d'une validation croisée
- Évaluation des performances du modèle sur l'ensemble de test
- Écriture les prédictions sur les données de test

Les modèles de classification utilisés sont SVM, KNN, RandomForest, DecisionTree, et NaiveBayes. Le modèle utilisé pour les résultats finaux est SVM.

5.3. Résultats et commentaires:

Les résultats finaux obtenus sont les suivants :

- Score de validation croisée : 0.79
- Score de mi-parcours : 0.73

On peut expliquer ces résultats par le fait que le modèle est capable de classifier correctement 79% des images en moyenne lorsqu'il est testé sur différentes sous-ensemble de données d'entraînement en utilisant la validation croisée. On peut considérer que ce résultat est assez bon parce que le modèle est capable d'être généraliser sur des nouvelles données.

De plus, les résultats du test de mi-parcours est 0.73 ce qui indique que le modèle a été capable de classifier correctement 73% les nouvelles images fournies. Bien que ce score soit inférieur à celui de la validation croisée, il est encore considéré comme un score de performance stable.

On doit noter que ces résultats peuvent être améliorés vu que la plupart des erreurs du modèle sont dues à des images contenant du ciel. Etant donné que la mer et le ciel partagent des caractéristiques similaires telles que la couleur, le modèle souvent ne distingue pas entre les deux.

Alors, il existe des moyens pour améliorer la performance tels que:

- L'utilisation d'autres algorithmes plus avancés comme les réseaux neurones convolutifs (CNN).
- L'augmentation du jeu de données parce que la taille des données d'entraînement reste relativement petite et cela n'aide pas le modèle à être mieux généralisé et à mieux connaître les caractéristiques de l'image.
- La possibilité d'extraire les caractéristiques de la texture d'images. C'est une méthode utilisée pour identifier les motifs dans l'image. En effet, la mer a souvent une texture granuleuse, alors que le ciel a une texture plus lisse et uniforme.

6. Conclusion:

L'une des principales difficultés rencontrées au cours de ce projet a été le processus de transformation des images, qui a nécessité plusieurs techniques différentes pour améliorer le score du modèle comme l'inversion de couleurs et la rotation d'image. Ainsi que la distinction entre mer et ciel puisque les deux partages les mêmes caractéristiques comme la couleur donc souvent le modèle se trompe si l'image contient ciel. Mais malgré ces défis, nous avons pu obtenir des résultats assez satisfaisants.

En conclusion, ce projet nous a permis d'avoir une idée générale sur l'apprentissage automatique et le fonctionnement des algorithmes. D'ailleurs, ce projet a été notre première expérience en apprentissage automatique donc il reste encore beaucoup d'améliorations à réaliser.

Références:

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

Documentation GridSearchCV:
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Documentation Numpy:
https://numpy.org/doc/1.24/

Documentation Matplotlib:
https://matplotlib.org/stable/api/index.html

Documentation Pillow:
https://pillow.readthedocs.io/en/stable/reference/index.html

Documentation Scikit-Image:
https://scikit-image.org/docs/stable/api/api.html