LIU Yingjie
LI Mengxiao
YU Zhengdao
DING Zhengtian

L3 INFORMATIQUE 2022-2023

Rapport - l'Apprentissage Statistique

Classification des étoiles en fonction de leurs caractéristiques spectrales

Table des matières

Introduction 1
Contenu 2
1. Préparation de données 2
1.1 Analyse du dataset et Pré-processing des données 2
2. Application avec différentes méthodes 5
2.1 logic regression 5
2.2 SVM 5
2.3 Arbre de décision 5
2.4 Random Forest 6
3 Comparasion des résultats 6
Conclusion 7

Introduction

Le sujet de notre projet est sur la classification des étoiles en fonction de leurs caractéristiques spectrales.

En astronomie, la classification stellaire est la classification des étoiles basée sur leurs caractéristiques spectrales. Le schéma de classification des galaxies, des quasars et des étoiles

est l'un des plus fondamentaux en astronomie. La catégorisation précoce des étoiles et leur répartition dans le ciel a conduit à la compréhension qu'elles composent notre propre galaxie et, après avoir distingué qu'Andromède était une galaxie distincte de la nôtre, de nombreuses galaxies ont commencé à être étudiées à mesure que des télescopes plus puissants étaient construits.

Donc notre projet vise à classer les étoiles, les galaxies et les quasars en fonction de leurs signatures spectrales.

L'ensemble des données que nous utilisons pendant le projet se trouve sur le lien suivant : https://www.kaggle.com/datasets/fedesoriano/stellar-classification-dataset-sdss17?datasetId=1866141&sortBy=voteCount

Contenu

1. Préparation de données

1.1 Analyse du dataset et Pré-processing des données

Avant de commencer à manipuler notre dataset, il est important de analyser l'ensemble du dataset.
Tout d'abord, nous visualisons directement l'ensemble de données et observons directement toutes les caractéristiques des données. Les données présente les caractéristiques suivantes :

obj_ID = Object Identifier, the unique value that identifies the object in the image catalog used by the CAS

alpha = Right Ascension angle (at J2000 epoch)

delta = Declination angle (at J2000 epoch)

u = Ultraviolet filter in the photometric system

g = Green filter in the photometric system

r = Red filter in the photometric system

i = Near Infrared filter in the photometric system

z = Infrared filter in the photometric system

run_ID = Run Number used to identify the specific scan

rereun_ID = Rerun Number to specify how the image was processed

cam_col = Camera column to identify the scanline within the run

field_ID = Field number to identify each field

spec_obj_ID = Unique ID used for optical spectroscopic objects (this means that 2 different observations with the same spec_obj_ID must share the output class)

class = object class (galaxy, star or quasar object)

redshift = redshift value based on the increase in wavelength

plate = plate ID, identifies each plate in SDSS

MJD = Modified Julian Date, used to indicate when a given piece of SDSS data was taken

fiber_ID = fiber ID that identifies the fiber that pointed the light at the focal plane in each observation

Après l'analyse, nous avons trouvé les problèmes suivants avec cet ensemble de données

1. Fonctionnalités redondantes(redundant feature) : lors de la formation de modèles d'apprentissage automatique, vous pouvez envisager de supprimer les fonctionnalités d'identification obj_ID, run_ID, rereun_ID et field_ID, car elles ne fournissent aucune information sur les étoiles. De plus, si un algorithme d'apprentissage supervisé doit être utilisé pour prédire la catégorie cible, la fonctionnalité spec_obj_ID peut être supprimée, car elle contient des informations sur la catégorie cible, ce qui peut entraîner un surajustement du modèle.

2. La quantité de données dans les trois catégories est déséquilibrée. Lorsqu'il existe des différences quantitatives significatives entre différentes classes dans un ensemble de données, cela entraînera une distribution déséquilibrée des classes de l'ensemble de données. Cela peut avoir un impact négatif sur les performances des modèles d'apprentissage automatique, notamment :

- Biais vers un nombre plus élevé de classes : les modèles d'apprentissage automatique ont tendance à prédire les classes avec un nombre plus élevé de classes, car cela maximise la précision de la prédiction. Cela entraînera des performances médiocres du modèle sur un plus petit nombre de classes.

- Précision insuffisante : en raison du petit nombre d'échantillons dans les données d'entraînement pour une petite classe, le modèle d'apprentissage automatique peut ne pas être en mesure

d'apprendre les caractéristiques de cette classe et peut classer la classe par erreur comme d'autres classes.

- Performances instables du modèle : lors de l'utilisation de la validation croisée sur un jeu de données, différents plis dans le jeu de données peuvent avoir des distributions de classes différentes, ce qui peut entraîner de grandes fluctuations dans les performances du modèle.

Pour résoudre ce problème, nous avons adopté deux approches pour nos données :

1. Sous-échantillonnage (undersampling) : sélectionnez au hasard une partie des échantillons parmi un grand nombre de catégories, de sorte que le nombre d'échantillons dans les trois catégories soit égal ou proche. Cette approche peut entraîner une perte d'informations, car des échantillons sélectionnés à partir d'un grand nombre de classes peuvent être représentatifs.

2. Suréchantillonnage (oversampling) : augmenter le nombre d'échantillons en copiant les catégories minoritaires ou en synthétisant artificiellement de nouveaux échantillons de catégories minoritaires, de sorte que le nombre d'échantillons dans les trois catégories soit égal ou proche. Cette approche peut entraîner des problèmes de surajustement, car le même échantillon dans l'ensemble de données apparaît plusieurs fois lors de la formation et des tests.

3. Une plage de données excessive pour chaque fonctionnalité peut entraîner les problèmes suivants :

-Apprentissage du modèle plus lent : une plage de données excessive signifie que les valeurs propres sont très différentes, ce qui peut entraîner une convergence plus lente de l'apprentissage du modèle.

- Surajustement : si certaines caractéristiques ont une plage de valeurs étendue et d'autres caractéristiques ont une plage de valeurs réduite, alors dans le modèle, ces caractéristiques avec une plage de valeurs étendue peuvent être plus importantes, ce qui entraîne un surajustement du modèle.

- Calcul inexact des poids des caractéristiques : dans certains algorithmes d'apprentissage automatique, les poids des caractéristiques sont calculés en fonction de la plage de variation des valeurs des caractéristiques. Si la plage de valeurs d'une caractéristique est importante, le poids de cette caractéristique peut être suramplifié.

Afin de résoudre ces problèmes, on pout envisager de normaliser ou de standardiser les fonctionnalités pour mettre à l'échelle les valeurs des fonctionnalités dans une plage plus petite. Cela peut rendre les pondérations entre les caractéristiques plus équilibrées, améliorant ainsi les performances du modèle.

4.Après l'analyse de la matrice de coefficients de corrélation trop grande, nous avons constaté qu'il existe des caractéristiques de corrélation élevées. En utilisant l'ACP pour réduire la

dimension, selon la relation entre le nombre de composantes principales et la variance, nous avons trouvé qu'il est suffisant et plus approprié de retenir six composantes principales.

2. Application avec différentes méthodes

2.1 logic regression

Dans cette partie, nous avons utilisé et optimisé un modèle de régression logistique. Nous avons généré un ensemble de données, créé un modèle de régression logistique et évalué sa performance. Ensuite, nous avons tracé la courbe d'apprentissage pour analyser les performances du modèle.

Nous avons également utilisé la régularisation L2 et le prétraitement des données avec StandardScaler. Nous avons expérimenté avec différentes valeurs de paramètre de régularisation C et évalué les performances du modèle. Finalement, nous avons calculé la matrice de confusion et d'autres métriques d'évaluation, puis tracé une carte thermique de la matrice de confusion.

Notre code montre comment utiliser la régression logistique pour résoudre des problèmes de classification et comment évaluer, ajuster et optimiser le modèle.

2.2 SVM

SVM est permet de réaliser des tâches de classification et de régression en trouvant la meilleure séparation entre les différentes classes. Il utilise une fonction de noyau pour transformer les données en un espace de grande dimension, où la séparation des classes est plus facile à trouver.

Dans cette partie, nous avons utilisé un modèle de classification à l'aide de la méthode des SVM. Tout d'abord, nous avons créé un modèle SVM en utilisant un noyau linéaire pour notre modèle et l'avons ajusté sur les données d'apprentissage.

Après avoir ajusté le modèle, nous avons évalué ses performances sur l'ensemble de test en calculant la matrice de confusion et la précision.

De plus, nous avons visualisé la matrice de confusion et tracé la courbe d'apprentissage pour analyser l'évolution de la précision du modèle en fonction de la taille de l'ensemble d'apprentissage. La courbe d'apprentissage montre comment la précision du modèle évolue à mesure que la taille de l'échantillon d'apprentissage augmente, ce qui peut nous aider à identifier les problèmes éventuels de surapprentissage ou de sous-apprentissage.

2.3 Arbre de décision

L'arbre de décision est un modèle d'apprentissage automatique qui prédit la classe d'un objet en fonction de ses attributs en suivant un ensemble de règles de décision.

Dans cette partie, nous avons créé un modèle d'arbre de décision et l'avons ajusté à l'aide des données d'apprentissage. Une fois le modèle entraîné, nous avons effectué des prédictions sur l'ensemble de test et évalué les performances du modèle à l'aide de différentes métriques, telles que la matrice de confusion, l'exactitude, la précision, le rappel et etc.

Pour analyser plus en détail les performances de notre modèle, nous avons également tracé une courbe d'apprentissage, qui montre comment les performances du modèle évoluent en fonction de la taille de l'ensemble d'apprentissage. Cette courbe nous aide à déterminer si notre modèle souffre de surajustement ou de sous-ajustement et à identifier d'éventuelles améliorations.

Enfin, nous avons visualisé l'arbre de décision lui-même pour mieux comprendre la structure du modèle et les règles de décision qu'il utilise pour effectuer des prédictions. La visualisation de l'arbre de décision peut également aider à expliquer le modèle aux parties prenantes et à déterminer si les règles de décision utilisées par l'arbre sont conformes à l'expertise du domaine.

2.4 Random Forest

Random Forest est une méthode d'apprentissage automatique utilisée pour la classification et la régression. Il s'agit d'un algorithme d'ensemble, qui combine plusieurs arbres de décision pour produire une prédiction finale.

Lorsque nous avons utilisé cette méthode dans le code, nous avons créé un objet de type RandomForestClassifier avec 100 arbres de décision, et nous avons ensuite entraîné ce modèle avec des données d'entraînement. Nous avons ensuite utilisé ce modèle pour faire des prédictions sur des données de test et avons mesuré la précision de ces prédictions à l'aide d'une fonction d'évaluation.

Enfin, nous avons tracé une courbe d'apprentissage pour évaluer les performances du modèle. Cette courbe nous montre comment les performances du modèle évoluent en fonction de la taille de l'ensemble d'apprentissage, ce qui peut nous aider à détecter les problèmes de sur-apprentissage ou de sous-apprentissage.

3 Comparasion des résultats

Over Sample:

Comparasion des modeles	
Donnees	Regression logistique
Résultats de la validation croisée：	76.7%
Résultats de l’ensemble test:	76.8%
Matrice de confusion：	accuracy： 74.5% precision： 86.1% recall:76.5% F1-score:81%
ROC Courbe Test:	76.7%

Under Sample:

Comparasion des modeles				
Donnees	Regression logistique	SVM	Decision Tree	Random Forest
Résultats de la validation croisée:	73.1%			
Résultats de l'ensemble test:	73.3%			
Matrice de confusion:	accuracy: 74.5% precision: 86.1% recall:76.5% F1-score:81%	accuracy:74.1%	accuracy：88.7% precision：88.7% recall:88.7% F1-score:88.7%	accuracy：92.3%
ROC Courbe Test:	73.3%			

Dans notre projet, nous avons essayé deux méthodes d'échantillonnage: la sur-échantillonnage(Over sample) et la sous-échantillonnage(Under Sample). Au cours de ce processus, nous avons constaté que les deux méthodes d'échantillonnage avaient peu d'impact sur les résultats des données. Nous avons donc décidé d'utiliser la sous-échantillonnage comme méthode d'échantillonnage principale et la sur-échantillonnage comme méthode d'observation auxiliaire.

Nous avons donc utilisé un seul modèle pour la sur-échantillonnage : la régression logistique, et quatre modèles pour la sous-échantillonnage: la régression logistique, SVM, l'arbre de décision et la forêt aléatoire. Après comparaison, nous avons constaté que Random Forest avait le taux de précision le plus élevé, soit 92,3 %. Bien que le taux de précision de l'arbre de décision ne soit pas aussi bon que celui de Random Forest, il est également excellent et il possède un modèle de données intuitif.

Conclusion

En conclusion, pour mieux analyser le jeu de données et comparer les avantages et les inconvénients de différents modèles, nous avons utilisé de nombreuses techniques différentes pour la comparaison :

nous avons utilisé deux méthodes d'échantillonnage, la sur-échantillonnage(Over sample) et la sous-échantillonnage(Under Sample). Bien que nous ayons finalement constaté que les résultats des données étaient similaires, nous avons également vu certains avantages et inconvénients de ces deux méthodes d'échantillonnage.

- La sur-échantillonnage a une précision plus élevée, mais peut causer un sur-ajustement.
- La sous-échantillonnage nécessite moins de temps d'entraînement et convient à de nombreux modèles d'entraînement, tels que la régression logistique, SVM, les arbres de décision, mais peut perdre des informations utiles, en particulier lorsque le nombre d'échantillons dans le jeu de données est élevé.

Après avoir utilisé différents modèles, nous avons également une compréhension plus approfondie des différences entre ces modèles.

- La régression logistique est rapide, mais elle est sensible aux données aberrantes, et il est préférable de les éliminer à l'avance.

- SVM est adapté à la classification de données à haute dimension, mais prend plus de temps d'entraînement, et n'est pas adapté aux jeux de données avec un grand nombre d'échantillons.
- L'arbre de décision a un modèle de données concis et peut expliquer le processus de décision, mais ne fonctionne pas aussi bien pour les données à haute dimension comme la forêt aléatoire. Cependant, la forêt aléatoire est moins pratique pour expliquer le processus de décision et prend plus de temps d'entraînement.

Après ce projet, nous avons non seulement une compréhension plus approfondie du cours de l'apprentissage statistique, mais cela a également enrichi notre expérience. Nous avons une meilleure compréhension de la façon de filtrer des bases de données, de traiter des données déséquilibrées et de choisir le meilleur modèle, etc. Enfin, nous vous remercions de votre attention portée à notre projet.