|  |  |  |  |
| --- | --- | --- | --- |
| **Machine Learning (ML)**  **Examen Final**  **1h30** |  | * Documents autorisés * Il faut rendre :   + page 1 de l’énoncé (cette feuille) ;   + une réponse sur papier (TP) ;   + un fichier Jupyter notebook avec votre nom. | |
| Nom et Prénom : | | | 2CSSIQ3  2CSSIL2 |

**Questions (6pts)**

Q1) Il faut transformer les caractéristiques nominales en numériques avant d’appliquer (1pt)

Régression logistique  Naïve Bayes  Arbre de décision  SVM

Q2) Quels sont les algorithmes d'apprentissage qui peuvent classer le dataset suivant parfaitement (1pt)

|  |  |
| --- | --- |
| y  x | Régression linéaire  Régression logistique  Arbre de décision  SVM |

Q3) L’erreur de classification tend vers 0 lorsque le nombre des échantillons d’entrainement tend vers l’infini. Quand est-ce que cette déclaration est fausse ? (1pt)

|  |
| --- |
|  |

Q4) Dessiner les graphes de performance d’un algorithme de classement non complexe d’entrainement et de test selon le nombre des itérations dans les deux cas : normal et sur-apprentissage (1pt)

|  |  |
| --- | --- |
| **performance**  **# itérations itérations**  **entraînement**  **test** | **performance**  **# itérations itérations** |
| Apprentissage normal | Apprentissage avec sur-apprentissage |

Q5) Dans les SVM, les classes sont +1 et -1. Que signifie 0 ? (0.5pt) Quel est l’intérêt ? (0.5pt)

|  |
| --- |
|  |

Q6) Vrai ou Faux (1pt)

a) Un classifieur entrainé sur moins de données est moins probable de sur-apprendre  Vrai  Faux

b) La régulation L1 peut être utilisée pour la sélection d’attributs  Vrai  Faux

c) Les arbres de décision n’ont pas besoin de sélection d’attribut  Vrai  Faux

d) On peut utiliser la régression logistique simple pour le classement multinomial  Vrai  Faux

**TP (14pts)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| name | eggs | aquatic | backbone | covering | class |
| dolphin | 0 | 1 | 1 | skin | Mammal |
| elephant | 0 | 0 | 1 | hair | Mammal |
| platypus | 1 | 1 | 1 | hair | Mammal |
| elephant | 0 | 0 | 1 | hair | Mammal |
| falcon | 1 | 0 | 1 | feathers | Bird |
| penguin | 1 | 1 | 1 | feathers | Bird |
| sole | 1 | 1 | 1 | skin | Fish |
| tortoise | 1 | 0 | 1 | skin | Reptile |
| seasnake | 0 | 1 | 1 | skin | Reptile |
| tuna | 1 | 1 | 1 | skin | Fish |
| mink | 0 | 1 | 1 | hair | Mammal |
| chicken | 1 | 0 | 1 | feathers | Bird |

Voici un dataset pour classer les animaux en Mammal, Bird, Fish et Reptile. Nous avons choisi un animal représentant pour chaque vecteur de caractéristiques. Par exemple, le lion a les mêmes caractéristiques que l’éléphant.

Application sur Jupyter notebook (questions « a ») + Réponse sur feuille d’examen (questions  « b »)

**Mot de passe : ML\_20@esi.dz**

**1- Préparation des données**

1.a- Lire le fichier, afficher son contenu et prétraiter les données sur le notebook **(2pts)**

1.b- Quels traitements avez-vous utilisé et pourquoi ? **(1pt)**

**2- Encodage des attributs**

2.a- Encoder les attributs afin d’utiliser les données par naïve Bayes multinomial, arbre de décision et la régression logistique **(1pt)**

2.b- Justifier le choix d’encodage **(1pt)**

**3- Échantillonnage et division des données**

3.a- Diviser le dataset en données d’entraînement et de test en gardant les 4 derniers échantillons pour le test. **(1.5pt)**

3.b- En se basant sur cette division :

* Est-ce que le dataset d’entraînement est déséquilibré ? Justifier la réponse en indiquant la classe majoritaire en cas de déséquilibre **(0.5pt)**
* Mentionner une méthode pour régler le problème de déséquilibre **(0.5pt)**
* Est-ce que cette division est raisonnable et pourquoi ? **(0.5pt)**

**4- Entraînement**

4.a- Entraîner trois modèles : naïve Bayes multinomial (NB), arbre de décision (CART) et la régression logistique (REG) en utilisant leurs implémentations dans scikit-learn **(0.75pt)**

4.b- Après l’entraînement :

* Le F1 score sur l’entraînement *F1(NB, CART, REG) = (0.714, 1.0, 0.857)*. Qu’est-ce que chaque algorithme cherche à faire ? c-à-d, comment une classe est vue par rapport aux caractéristiques ? **(1.5pt)** Pourquoi CART a pu atteindre 1.0 ? **(0.25pt)**
* Le temps d’entraînement *T(NB, CART, REG) = (0.0017, 0.0006, 0.0073)*. Expliquer comment chaque algorithme fonctionne en motivant ces durées **(1.5pt)**
* Nous avons utilisé la régularisation L2 dans la régression logistique. Expliquer l’intérêt de cette régularisation en indiquant son équation **(0.5)**

***Tourner la page ; la torture n’est pas encore finie !***

**5- Test**

5.a- Prédire les classes de test pour chaque modèle et afficher le rapport de classement. **(0.5pt)**

5.b- Que signifient les résultats suivants :

* F1-score de la classe « Bird » est 1.0 dans les trois modèles **(0.25pt)**
* F1-score de la classe « Reptile » est 0.0 dans les trois modèles **(0.25pt)**
* La précision d’une classe dans un des modèles est 0.33 et le rappel est 1.0 **(0.25pt)**
* La précision dans un modèle en terme de micro-avg est 0.75 et en terme de macro-avg est 0.62 **(0.25pt)**

***Bonne chance (luck is overrated; hard work is the key)***

**ANNEXE**

|  |  |  |
| --- | --- | --- |
| **Fonction/Classe** | **Parametres/Méthodes** | **Description** |
| pandas.DataFrame.copy [Fn] | … | Make a copy of this object’s indices and data. |
| pandas.get\_dummies [Fn] | Data, columns=[…], … | Convert categorical variable into dummy/indicator variables. |
| sklearn.preprocessing.OrdinalEncoder [Cls] | Method: fit\_transform(X[, y]) | Encode categorical features as an integer array. |
| sklearn.preprocessing.OneHotEncoder [Cls] | Method: fit\_transform(X[, y]) | Encode categorical features as a one-hot numeric array. |
| sklearn.linear\_model.LogisticRegression [Cls] | Parametres: penalty='l2|l1|none', solver='lbfgs| liblinear|saga', multi\_class='auto|ovr|multinomial', …  Méthodes: fit(X, y[, sample\_weight])  predict(X) | Logistic Regression (aka logit, MaxEnt) classifier. |
| sklearn.naive\_bayes.MultinomialNB [Cls] | Parametres: alpha=1.0, fit\_prior=True, …  Méthodes: fit(X, y[, sample\_weight])  predict(X) | Naive Bayes classifier for multinomial models |
| sklearn.naive\_bayes.GaussianNB [Cls] | Parametres: priors=None, var\_smoothing=1e-09, …  Méthodes: fit(X, y[, sample\_weight])  predict(X) | Gaussian Naive Bayes (GaussianNB) |
| sklearn.metrics.f1\_score [Fn] | y\_true, y\_pred, \*[, …] | Compute the F1 score, also known as balanced F-score or F-measure |