Atelier Python pour Data Scientist

Séance 2: Application des modèles de machine learning avec Sklearn et Tensorflow

Ali ZAINOUL
ali.zainoul.az@gmail.com

21 décembre 2023

Introduction à Scikit-Learn

- Scikit-Learn (également connu sous le nom de Sklearn) est une bibliothèque open-source populaire pour l'apprentissage automatique en Python.
- Offrant des outils simples et efficaces pour l'analyse de données et la modélisation statistique, Scikit-Learn est utilisé par de nombreux praticiens et chercheurs dans le domaine de l'intelligence artificielle, de l'apprentissage automatique et de la science des données.

Techniques Classiques

- Régression Logistique : Classification binaire, probabilités. (Régression Logistique)
- Forêts d'Arbres de Décision : Classification, régression. (Forêts d'Arbres de Décision)
- Machines à Vecteurs de Support (SVM) : Classification, régression. (Machines à Vecteurs de Support)

Techniques de Base

- Régression Linéaire : Modélisation des relations linéaires. (Régression Linéaire)
- K-Plus Proches Voisins (KNN) : Classification, régression. (K-Plus Proches Voisins)
- Réseaux de Neurones : Modèle inspiré du cerveau, utilisé pour diverses tâches. (Réseaux de Neurones)

Techniques Avancées

- Régression Ridge : Régression linéaire avec régularisation L2. (Régression Ridge)
- Régression Lasso : Régression linéaire avec régularisation L1. (Régression Lasso)
- Machines à Vecteurs de Support à Noyau (SVM à Noyau) : SVM avec des noyaux non linéaires. (SVM à Noyau)

La liste est bien évidemment non exhaustive ...

Objectif de l'activité

- L'objectif de cette activité est de vous initier à l'utilisation de Scikit-Learn pour appliquer des modèles d'apprentissage automatique à des jeux de données réels.
- Nous explorerons les principales étapes du processus, de la préparation des données à l'évaluation des modèles, en utilisant un exemple pratique de régression logistique appliquée au jeu de données Iris.
- Inspiré par l'exemple qui va suivre, nous illustrerons comment charger les données, les préparer, entraîner un modèle, effectuer des prédictions et évaluer la performance, tout en expliquant chaque étape avec des détails pertinents.

Introduction à la Régression Logistique en Apprentissage Automatique

- L'apprentissage automatique, une branche de l'intelligence artificielle (IA), propose des solutions puissantes pour extraire des informations significatives à partir de données.
- Parmi les techniques d'apprentissage automatique, la régression logistique se distingue en tant que méthode populaire pour résoudre des problèmes de classification.

Comprendre la Régression Logistique

- La régression logistique est une technique supervisée pour prédire la probabilité d'appartenance à une classe spécifique.
- Souvent utilisée dans des problèmes de classification binaire.
- Elle utilise la fonction sigmoïde pour transformer une combinaison linéaire de variables en une probabilité entre 0 et 1. La fonction sigmoïde est définie comme suit :

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad \underline{\text{(source)}}$$

But et Applications de la Régression Logistique

- Applications diverses : finance, médecine, marketing, et apprentissage automatique.
- Objectifs courants : prédiction de probabilité, classification binaire, évaluation de performance.
- Exemple : prédire la probabilité d'achat en ligne, détection de spam, etc.

Conclusion

- La régression logistique est une méthode polyvalente pour résoudre des problèmes de classification.
- Scikit-Learn offre des outils puissants pour créer, entraîner et évaluer des modèles avec facilité.
- La compréhension de ces concepts ouvre la voie à un voyage captivant au cœur de l'intelligence artificielle et de la science des données.
- Explorez les documents officiels de Scikit-Learn pour approfondir vos connaissances et découvrez d'autres exemples en ligne.

Documentation Régression Logistique

- Semantic Scholar: The iris data set
  Cette ressource explore l'origine du jeu de données Iris et son importance dans le domaine de l'apprentissage automatique.
- UCI Machine Learning Repository: Iris Dataset
  L'UCI Machine Learning Repository est une source fiable pour le jeu de données Iris, fournissant des détails sur ses caractéristiques et son utilisation.
- UCI: Iris Dataset CSV File
  Lien direct vers le fichier CSV du jeu de données Iris, utile pour l'importation dans des environnements Python.

Introduction à TensorFlow

- TensorFlow est une bibliothèque open source développée par Google pour l'apprentissage automatique et le calcul numérique.
- Elle offre une architecture flexible et extensible, adaptée à la création de modèles de machine learning, en particulier les réseaux de neurones.
- TensorFlow est utilisé dans divers domaines, y compris la vision par ordinateur, le traitement du langage naturel, la reconnaissance vocale et d'autres applications liées à l'intelligence artificielle.
- Sa popularité est attribuée à sa communauté active, ses outils puissants et sa compatibilité avec une variété de plates-formes.

Utilisation dans la Classification d'Images

- TensorFlow est largement utilisé pour la classification d'images à l'aide de réseaux de neurones, en particulier les CNN (Convolutional Neural Networks).
- Des modèles pré-entraînés comme Inception, MobileNet et EfficientNet exploitent la puissance de TensorFlow pour la reconnaissance d'objets dans des images.
- Ces applications incluent la détection d'objets, la classification d'images médicales et la reconnaissance faciale.
- TensorFlow propose des outils tels que TensorFlow Lite pour déployer des modèles sur des appareils mobiles et intégrés.

Traitement du Langage Naturel (NLP)

- Dans le domaine du NLP, TensorFlow est utilisé pour construire des modèles de traitement du langage naturel, y compris les réseaux de neurones récurrents (RNN) et les transformers.
- Des applications telles que la traduction automatique, la génération de texte et l'analyse de sentiment bénéficient des fonctionnalités avancées de TensorFlow.
- TensorFlow propose des outils comme TensorFlow NLP pour faciliter le prétraitement des données et la construction de modèles NLP performants.
- Des modèles tels que BERT (Bidirectional Encoder Representations from Transformers) ont été implémentés avec succès à l'aide de TensorFlow.

Développement de Modèles de Régression

- TensorFlow est également utilisé pour le développement de modèles de régression, adaptés à la prédiction de valeurs continues.
- Des exemples incluent la régression linéaire, les modèles de régression logistique, et même des réseaux de neurones pour des tâches de régression complexes.
- Les applications couvrent des domaines tels que la prédiction de prix, la modélisation financière, et d'autres scénarios où la prédiction de valeurs numériques est cruciale.
- Les fonctionnalités de TensorFlow, comme le calcul distribué et l'accélération matérielle, améliorent les performances des modèles de régression.

Recherche et Expérimentation

- TensorFlow est un choix privilégié pour la recherche et l'expérimentation en raison de sa flexibilité et de ses fonctionnalités avancées.
- Les chercheurs peuvent créer rapidement des prototypes de modèles, tester différentes architectures, et utiliser des techniques de pointe grâce aux fonctionnalités de TensorFlow.
- TensorFlow Extended (TFX) offre une plate-forme complète pour le déploiement et la gestion des modèles dans des environnements de production.
- La documentation étendue, les tutoriels et les forums en ligne font de TensorFlow une ressource précieuse pour les chercheurs et les passionnés d'apprentissage automatique.

Objectif - Exemple 1

Objectif de l'exemple : Construction, entraînement et évaluation d'un réseau neuronal dense pour la classification des données Iris.

Importation des bibliothèques (Exemple 1)

Bibliothèques utilisées :

- TensorFlow (TF) : Bibliothèque open source pour l'apprentissage automatique.
- scikit-learn : Bibliothèque pour l'apprentissage automatique en Python.
- NumPy : Bibliothèque pour le calcul numérique en Python.

Chargement des données (Exemple 1)

Chargement du jeu de données Iris :

- Données : Chargées à partir du jeu de données Iris.
- Division : 80% pour l'entraînement, 20% pour les tests.
- Random_state : 42 pour la reproductibilité.

Normalisation des données (Exemple 1)

Normalisation des données :

- StandardScaler : Paramètres appris sur l'ensemble d'entraînement.

Construction du modèle (Exemple 1)

Construction du modèle TensorFlow :

- Modèle séquentiel avec une couche dense.
- Dense : 3 neurones, activation softmax, input_shape selon le nombre de features.

Compilation du modèle (Exemple 1)

Compilation du modèle :

- Optimiseur : Adam.
- Loss : sparse_categorical_crossentropy.
- Metrics : accuracy.

Entraînement du modèle (Exemple 1)

Entraînement du modèle :

- Données normalisées, 50 époques, verbose=2.

Évaluation du modèle (Exemple 1)

Évaluation du modèle :

- Prédictions : Probabilités prédites.
- Précision : Calculée avec accuracy_score.

Objectif - Exemple 2

Objectif de l'exemple : Construction, entraînement et évaluation d'un réseau neuronal dense pour la classification d'images du jeu de données MNIST.

Importation des bibliothèques (Exemple 2)

Bibliothèques utilisées :

- TensorFlow (TF) : Bibliothèque open source pour l'apprentissage automatique.
- TensorFlow.keras.layers : Composants de couche pour les modèles Keras.
- TensorFlow.keras.models : Construction de modèles séquentiels en Keras.
- TensorFlow.keras.datasets : Fonction pour charger des jeux de données.
- TensorFlow.keras.utils : Utilitaires Keras, utilisé pour to_categorical.
- scikit-learn : Bibliothèque pour l'apprentissage automatique en Python.

Chargement des données (Exemple 2)

Chargement du jeu de données MNIST :

- Chargement : Fonction mnist.load_data().

Prétraitement des données (Exemple 2)

Prétraitement des images :

- Normalisation : Les valeurs des pixels sont divisées par 255.
- Conversion des étiquettes : to_categorical pour les étiquettes.

Construction du modèle (Exemple 2)

Construction du modèle TensorFlow :

- Modèle séquentiel avec une couche flatten et deux couches dense.

Compilation du modèle (Exemple 2)

Compilation du modèle :

- Optimiseur : Adam.
- Loss : categorical_crossentropy.
- Metrics : accuracy.

Entraînement du modèle (Exemple 2)

Entraînement du modèle :

- 10 époques, validation sur un ensemble de validation, verbose=2.

Évaluation du modèle (Exemple 2)

Évaluation du modèle :

- Évaluation sur l'ensemble de test.

Récapitulatif et questions

- Révision des concepts abordés.
- Répondre aux questions des participants.

Lectures recommandées

- Documentation Scikit-Learn : `https://scikit-learn.org/stable/`
- Documentation Pandas : `https://pandas.pydata.org`
- Documentation NumPy : `https://numpy.org`
- Documentation TensorFlow : `https://www.tensorflow.org`
- Teachable Machine avec Google : `https://teachablemachine.withgoogle.com`

Plateformes conseillées

- 360 Learning : `https://www.360learning.com`
- Lien avec exemples : `https://github.com/AliZainoul/DSEPSIBDX`