Démonstration sur Azure Machine Learning SDKs

Cette démonstration a pour but de tester les APIs et de démontrer les fonctionnalités offertes par les nouveaux Azure Machine Learning SDKs.

Présentation des Azure Machine Learning SDKs

Azure Machine Learning SDK for Python

À propos de Workbench …

L'application Workbench et certaines autres fonctionnalités antérieures ont été remplacées dans la version de septembre 2018 pour améliorer l'architecture de Azure Machine Learning service. La nouvelle version contient de nombreuses mises à jour importantes. La fonctionnalité principale des expériences au déploiement du modèle n'a pas changé, mais on peut maintenant utiliser le SDK et la CLI pour accomplir nos tâches de machine learning et nos pipelines.

Qu'est-ce qu'Azure Machine Learning SDK for Python ?

Azure Machine Learning SDK for Python est utilisé par les data scientists et les développeurs d'intelligence artificielle pour créer et exécuter des flux de travail de machine learning sur l'Azure Machine Learning service.

Azure Machine Learning Data Prep SDK

Data Prep SDK permet de charger, de transformer et d'écrire des données pour les flux de travail de machine leanring. On peut interagir avec ce SDK dans n'importe quel environnement Python, y compris Jupyter notebook ou d'autres IDEs de Python.

Data Prep SDK présente certains avantages par rapport aux d'autres librairies populaires de la préparation des données, par exemple Pandas :

- **Lecture intelligente**. Le SDK peut détecter automatiquement tous les types de fichiers pris en charge. On n'a donc pas besoin d'utiliser de lecteurs de fichiers spéciaux pour CSV, texte, Excel, etc., ni de spécifier des paramètres de délimiteur, d'en-tête ou de codage.
- **Échelle en streaming**. Plutôt que de charger toutes les données en mémoire, le moteur SDK fournit les données en continu, ce qui lui permet de s'adapter et de mieux fonctionner sur les grands volumes de données.
- **Fonctionnalité multi-plateforme avec un script de code unique**. Écrivez dans un seul SDK et exécutez-le sous Windows, macOS, Linux ou Spark de manière « scale-up » ou « scale-out ». Lors de l'exécution en mode « scale-up », le moteur tente d'utiliser tous les threads matériels disponibles. Lorsqu'il exécute en mode « scale-out », il permet au planificateur distribué d'optimiser l'exécution.

Présentation de la démonstration

Le projet qu'on réalise dans les parties suivantes pour la démonstration provient de la compétition « **New York City Taxi Trip Duration »** organisé par Kaggle (Lien de compétition : https://www.kaggle.com/c/nyc-taxi-trip-duration).

Le but du projet est de prévoir la durée de chaque trajet en fonction de ses attributs.

Description des données

Le jeu de données a été publiées à l'origine par la NYC Taxi and Limousine Commission (TLC). Le jeu de données est composé de 1458644 lignes et 11 colonnes. Chaque ligne représente un enregistrement de trajet.

Un enregistrement contient 11 informations concernant un trajet donné :

- **id** – Identifiant du trajet
- **vendor_id** – Identifiant de la compagnie de Taxi
- **pickup_datetime** - Date et heure auxquelles le compteur a été engagé
- **dropoff_datetime** - Date et heure auxquelles le compteur a été désengagé
- **passenger_count** – Nombre de voyageur dans le taxi
- **pickup_longitude** - Longitude où le compteur a été engagé
- **pickup_latitude** - Latitude où le compteur a été engagé
- **dropoff_longitude** - Longitude où le compteur a été désengagé
- **dropoff_latitude** - Latitude où le compteur a été dégagé
- **store_and_fwd_flag** - Drapeau pour indiquer si cet enregistrement a déjà été enregistré dans le mémoire du taxi avant d'envoyer à la compagnie de Taxi : Y = oui, N = non
- **trip_duration** – Durée du trajet

Déroulement de la démonstration

Actions pré-requises

Pour pouvoir effectuer à bien la démonstration, il faut, au préalable, avoir effectué quelques actions requises :

- **Créer un compte Microsoft Azure**
  Lien pour créer un compte Microsoft Azure : https://azure.microsoft.com/fr-fr/free/?WT.mc_id=A261C142F

- **Créer un Azure Machine Learning service workspace**
  Lien pour créer un workspace : https://docs.microsoft.com/en-gb/azure/machine-learning/service/quickstart-get-started#create-a-workspace

- **Configurer d'un environnement de développement & Installer Azure Machine Learning SDK**

  Il existe deux environnements de développement recommandées :

  - **Utiliser Azure Notebooks**
    Azure Notebooks est un service Jupyter Notebook dans le nuage Azure. Azure Notebooks est préconfiguré pour fonctionner avec l'Azure Machine Learning service. Les composants requis, tels que le Azure Machine Learning SDK, sont préinstallés sur cet environnement. On peut donc utiliser directement Azure Notebooks comme l'environnement de développement sans avoir besoin de la configuration.

  - **Configurer Jupyter Notebook sur notre propre ordinateur**
    Si on veut utiliser Jupyter Notebook sur notre propre ordinateur, il est recommandé d'utiliser Anaconda pour isoler notre environnement de travail afin d'éviter les conflits de dépendance entre les packages. (Lien pour télécharger Anaconda : https://www.anaconda.com/download)

Voici, un tutoriel pour configurer Jupyter Notebook sur notre propre ordinateur : https://docs.microsoft.com/en-gb/azure/machine-learning/service/how-to-configure-environment#configure-jupyter-notebooks-on-your-own-computer
Note : Si la version de conda <= 4.4, il faut utiliser activate <nom_env> pour activer le nouveau environnement et deactivate <nom_env> pour le désactiver.

- **Installer Azure Machine Learning Data Prep SDK**
  Pour installer ce SDK, utilisez la commande suivante :

```
pip install --upgrade azureml-dataprep
```

1ère partie : Préparation des données

Le notebook Data preparation.ipynb contient tous les pré-traitements et les « feature engineerings » qu'on réalise avec les données « brutes ». Principalement, on utilise Data Prep SDK pour charger et transformer les données. Cependant, pour le côté de visualisation, on utilise plutôt d'autres librairies, par exemple Matplotlib, Seaborn et Plotly, puisque Data Prep SDK ne fournit pas la fonctionnalité de visualisation de données pour le moment.

2ème partie : Entraînement et déploiement du modèle

Pour entraîner le modèle avec Azure Machine Learning service, il faut, dans un premier temps, créer un fichier de configuration de workspace avec le notebook Create workspace config file.ipynb et récupérer les données à partir de Azure Storage avec le notebook Download Data from Azure Storage.ipynb.

Note : Si le volume des données n'est pas très grand, on n'a pas vraiment besoin d'utiliser Azure Storage. On peut directement les charger à partir de notre propre ordinateur.

Note : Azure Storage est un service fournissant un stockage en cloud qui est hautement sécurisé, durable, et évolutif. Azure Storage inclut Azure Blobs (objets), Azure Data Lake Storage Gen2, Fichiers Azure etc. Dans notre cas, on utilise Azure Blobs pour stocker les données. L'Azure Blob est la solution de stockage d'objets pour le cloud. Le stockage d'objets blob est optimisé pour stocker des quantités énormes de données non structurées.

Ensuite, on entraîne le modèle avec deux types de machines virtuelles :

- **Entraînement avec DSVM**
  Notebook : Train models with DSVM.ipynb
  La Data Science Virtual Machine (DSVM) est une image de machine virtuelle préconfigurée pour les travaux de la Data Science. La machine virtuelle comprend des outils, des IDEs et des packages populaires tels que Jupyter Notebook, PyCharm et Tensorflow. Les composants requis pour la démonstration, tels que le SDK Azure Machine Learning, sont également préinstallés sur cet environnement.

- **Entraînement avec un « cluster » de machines virtuelles à priorité basse**
  Notebook : Train models with Batch.ipynb
  Azure Batch propose des machines virtuelles à faible priorité pour réduire le coût des charges de travail par lots. Les machines virtuelles à faible priorité permettent de nouveaux types de charges de travail par lots en utilisant une grande quantité de puissance de calcul à un coût très bas. Les machines virtuelles à faible priorité tirent parti de la surcapacité dans Azure.

Le « trade-off » pour utiliser des machines virtuelles à faible priorité est que ces machines peuvent ne pas être disponibles pour être alloués ou peuvent être préemptés à tout moment, en fonction de la capacité disponible.

Finalement, on utilise le notebook Deploy models.ipynb pour déployer le model sous forme de web service. Un web service est une image, dans notre cas une image Docker, qui encapsule la logique de « scoring » et le modèle lui-même.

GitHub repository

Lien de repo : https://github.com/xiangzhemeng/Demo-Azure-Machine-Learning-SDKs