Azure での分散トレーニング - クリックスルー率予測

NetApp Solutions

NetApp
September 10, 2024

目次

Azure での分散トレーニング - クリックスルー率予測

TR-4904 ：『 Distributed Training in Azure - Click Through Rate Prediction 』

ネットアップ、 Verron Martina 、 Muneer Ahmad 、 Rick Huang 氏

データサイエンティストの仕事は、機械学習（ ML ）モデルと人工知能（ AI ）モデルのトレーニングと調整に集中する必要があります。しかし、 Google の調査によると、データサイエンティストは、モデルをエンタープライズアプリケーションと連携させ、大規模に運用する方法を検討する時間の約 80% を費やしています。

エンドツーエンドの AI / ML プロジェクトを管理するには、エンタープライズコンポーネントについてより広範な理解が必要です。DevOps がその定義、統合、導入を引き継ぎましたが、 ML の運用では、 AI や ML プロジェクトを含む同様のフローがターゲットとなります。エンドツーエンドの AI / ML パイプラインが企業内でどのように影響するかを知るには、次の必要なコンポーネントのリストを参照してください。

- ストレージ
- ネットワーキング
- データベース
- ファイルシステム
- コンテナ
- 継続的統合 / 継続的導入（ CI / CD ）パイプライン
- 統合開発環境（ IDE ）
- セキュリティ
- データアクセスポリシー
- ハードウェア
- クラウド
- 仮想化
- データサイエンスのツールセットとライブラリ

対象読者

データサイエンスの世界は、 IT とビジネスのさまざまな分野に影響をもたらしています。

- データサイエンティストは、選択したツールとライブラリを柔軟に使用できる必要があります。
- データエンジニアは、データの流れと配置場所を把握する必要があります。
- DevOps エンジニアは、新しい AI / ML アプリケーションを CI / CD パイプラインに統合するためのツールを必要としています。
- クラウド管理者とアーキテクトは、 Azure リソースをセットアップおよび管理できる必要があります。

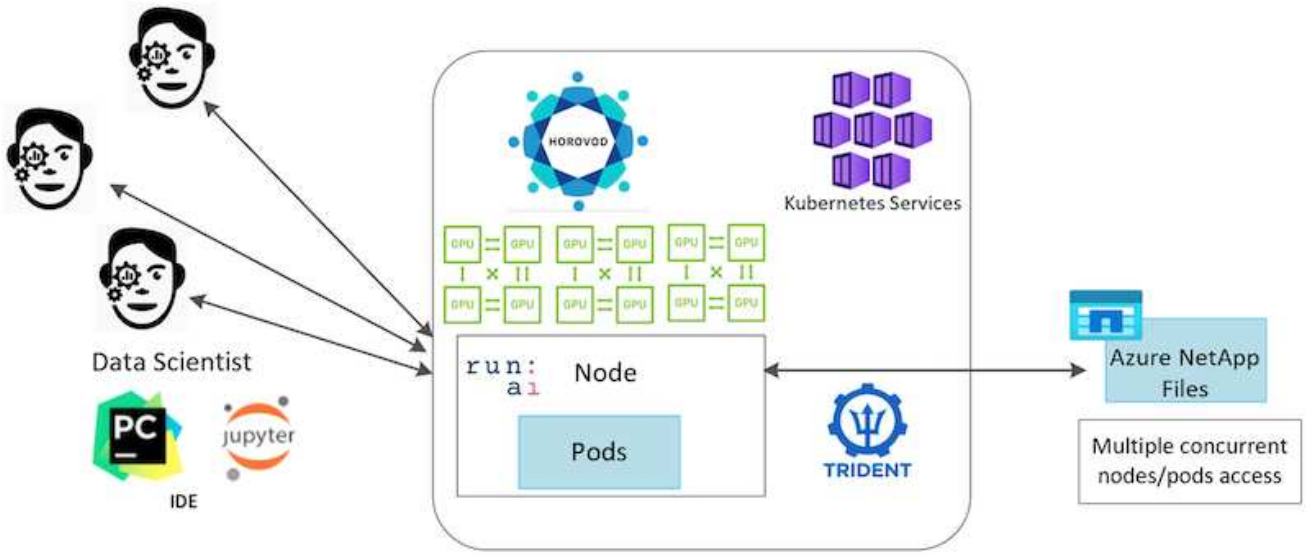
- ビジネスユーザは、 AI / ML アプリケーションにアクセスしたいと考えています。

このテクニカルレポートでは、 Azure NetApp Files 、 Rapids AI 、 Dask 、 Azure が、これらの各役割がビジネスにもたらす価値について説明します。

解決策の概要

この解決策は、 AI / ML アプリケーションのライフサイクルに従います。まず、データサイエンティストの仕事から始めて、データの準備やモデルのトレーニングに必要なさまざまなステップを定義します。Dask のラピッズを活用することで、 Azure Kubernetes Service （ AKS ）クラスタ全体で分散トレーニングを実施し、従来の Python の坐骨神経痛手法に比べてトレーニング時間を大幅に短縮しました。完全なサイクルを完了するには、パイプラインと Azure NetApp Files を統合します。

Azure NetApp Files は、さまざまなパフォーマンス階層を提供します。お客様はまず Standard 階層から始めて、データを移動することなく、スケールアウトしてハイパフォーマンス階層まで無停止でスケールアップできます。この機能により、データサイエンティストは、次の図に示すように、パフォーマンスの問題を発生させることなく、大規模なモデルのトレーニングを実施できます。クラスタ全体にデータサイロが発生することはありません。



テクノロジの概要

このページでは、この解決策で使用されているテクノロジの概要を説明します。

Microsoft とネットアップ

2019 年 5 月より、 Microsoft は Azure ネイティブのファーストパーティポータルサービスを提供し、 NetApp ONTAP テクノロジをベースとしたエンタープライズ NFS および SMB ファイルサービスを提供しています。この開発は、 Microsoft とネットアップの戦略的パートナーシップによって推進されており、ワールドクラスの ONTAP データサービスの Azure への対応範囲がさらに拡大しています。

Azure NetApp Files の特長

Azure NetApp Files サービスは、エンタープライズクラスの高パフォーマンスな従量課金制のファイルストレ

ージサービスです。Azure NetApp Files は、あらゆる種類のワークロードに対応し、デフォルトで高可用性を実現します。サービスレベルとパフォーマンスレベルを選択し、サービスを使用して Snapshot コピーをセットアップできます。Azure NetApp Files は Azure ファーストパーティサービスで、コードを変更することなく、データベース、 SAP 、ハイパフォーマンスコンピューティングアプリケーションなど、クラウドで最も要件の厳しいエンタープライズファイルワークロードを移行して実行します。

このリファレンスアーキテクチャには、 IT 組織に次のようなメリットがあります。

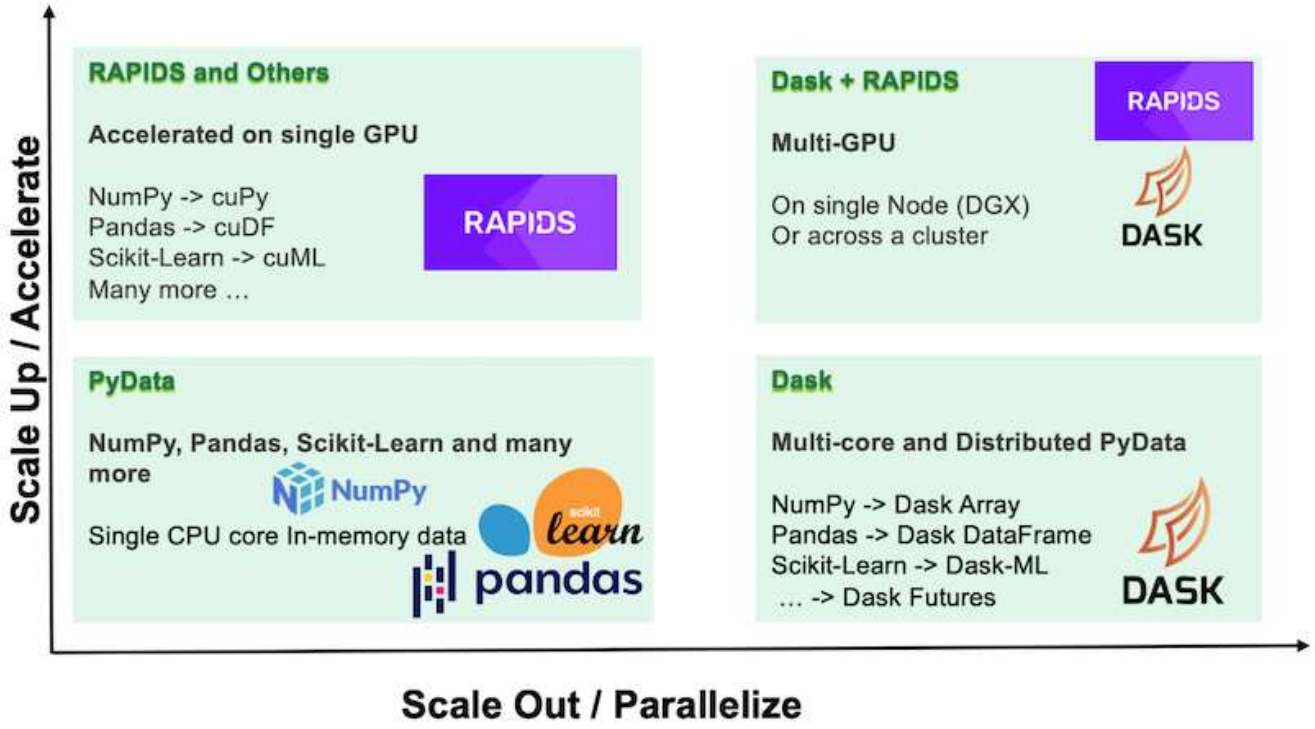
- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- さまざまなパフォーマンスとコストを考慮して、幅広いストレージ階層を提供します

Dask と NVIDIA Rapids の概要

Dask は、 Python ライブラリを複数のマシン上で拡張し、大量のデータを高速処理する、オープンソースの並列コンピューティングツールです。これは、 Pandas 、 numpy 、 scikit learn などのシングルスレッド従来の Python ライブラリに類似した API を提供します。その結果、ネイティブの Python ユーザは、クラスタ全体でリソースを使用するために既存のコードを大幅に変更する必要がなくなります。

NVIDIA Rapids はオープンソースライブラリのスイートで、 GPU 上でエンドツーエンドの ML ワークフローとデータ分析ワークフローを完全に実行できます。Dask と組み合わせることで、 GPU ワークステーション（スケールアップ）からマルチノードのマルチ GPU クラスタ（スケールアウト）へ簡単に拡張できます。

クラスタに Dask を導入するには、 Kubernetes を使用してリソースのオーケストレーションを行います。次の図に示すように、ワーカーノードをプロセス要件に従ってスケールアップまたはスケールダウンすることもできます。これは、クラスタのリソース消費を最適化するのに役立ちます。



ソフトウェア要件

次の表に、この解決策に必要なソフトウェア要件を示します。

ソフトウェア	バージョン
Azure Kubernetes Service の略	1.18.14
Rapids および Dask の容器のイメージ	リポジトリ : "rapidsai/rapidsai" タグ :0.17-cudda11.1-runtime-ubuntu18.04
NetApp Trident	20.01.1
Helm	3.0.0

クラウドリソースの要件

このページでは、 Azure NetApp Files のクラウドリソースの設定について説明します。

Azure NetApp Files を設定します

の説明に従って、 Azure NetApp Files を設定します "クイックスタート： Azure NetApp Files をセットアップし、 NFS ボリュームを作成します"。

「 Azure NetApp Files 用 NFS ボリュームの作成」のセクションを過ぎても、 Trident を使用してボリュームを作成できます。続行する前に、次の手順を実行します。

1. Azure NetApp Files とネットアップのリソースプロバイダに（ Azure Shell を使用）登録します "リンク"）。
2. Azure NetApp Files でアカウントを作成します（ "リンク"）。
3. 容量プール（必要に応じて、 4TB 以上の Standard または Premium ）をセットアップします（ "リンク"）。次の表に、クラウドでのセットアップに必要なネットワーク構成を示します。Dask クラスタと Azure NetApp Files は同じ Azure Virtual Network （ VNet ）またはピア関係にある VNet に配置されている必要があります。

リソース	「 **/version** 」と入力します
Azure Kubernetes Service の略	1.18.14
エージェントノード	3x Standard_DS2_v2
GPU ノード	3x Standard_NC6s_v3
Azure NetApp Files の特長	標準的な容量のプールがある
容量（ TB ）	4.

クリックスルー率予測ユースケースの概要

このユースケースは、一般に公開されているに基づいています "[ログ をクリックします"^] データセットの作成元 "Crito AI Lab の略"。ML プラットフォームとアプリケーショ

ンの最近の進歩により、現在は大規模な学習が注目されています。クリックスルー率（CTR）は、オンライン広告インプレッション数 100 件あたりの平均クリックスルー数（パーセンテージ）と定義されています。デジタルマーケティング、小売、 E コマース、サービスプロバイダなど、さまざまな業界やユースケースで重要な指標として広く採用されています。CTR を潜在的な顧客トラフィックの重要な指標として使用する例を以下に示します。

- * デジタルマーケティング :* インチ "Google アナリティクス"、 CTR は、広告主または販売主のキーワード、広告、および無料リストがどの程度効果を発揮しているかを測定するために使用できます。クリック率が高いと、ユーザーは広告やリストを便利で関連性の高いものとして見つけることができます。CTR はまたあなたのキーワードの予想される CTR に貢献する、の構成要素である "広告ランク"。
- * e- コマース： * 活用に加えて "Google アナリティクス"E コマースバックエンドには、少なくともいくつかの訪問者統計情報があります。これらの統計情報は一目見ると有用ではないように見えますが、通常は読みやすく、他の情報よりも正確な情報になる可能性があります。このような統計で構成されるファーストパーティデータセットは独占的なものであり、 E コマースの販売者、購買担当者、プラットフォームに最も関連性があります。これらのデータセットは、ベンチマークの設定に使用でき、過去 1 年と過去 1 日の間に結果を比較するために、さらに詳細な分析を行うための時系列を作成します。
- * 小売： * 実店舗の小売業者は、訪問者数と顧客数を CTR に関連付けることができます。お客様の数は、販売時点の履歴から確認できます。小売業者のウェブサイトや広告トラフィックの CTR が、前述の売上につながる可能性があります。ロイヤルティプログラムは、オンライン広告や他の Web サイトからリダイレクトされたお客様が報奨を獲得するために参加する可能性があるため、別のユースケースです。小売業者は、ロイヤルティプログラムを通じて顧客を獲得し、販売履歴から行動を記録することで、さまざまなカテゴリーで消費者の購買行動を予測するだけでなく、クーポンをパーソナライズし、チャーンを減らす推奨システムを構築できます。
- * 通信事業者とインターネット・サービス・プロバイダーは、豊富なデータを提供するファーストパーティのユーザー・テレメトリ・データを使用して、洞察に富んだ AI 、 ML 、分析のユースケースを実現しています。たとえば、携帯電話会社の Web 閲覧のトップレベルのドメイン履歴ログを毎日活用して、既存のモデルを微調整して最新のオーディエンスセグメンテーションを作成したり、顧客の行動を予測したり、リアルタイム広告を配置してオンライン体験を向上させることができます。このようなデータ主導のマーケティングワークフローでは、 CTR はコンバージョンを反映する重要な指標です。

デジタルマーケティングの文脈では、 "Crito Terabyte のログをクリックします" 現在は、 ML プラットフォームとアルゴリズムのスケーラビリティを評価する際の参考データセットとなっています。広告主は、クリックスルーレートを予測することで、広告に対応する可能性が最も高い訪問者を選択し、閲覧履歴を分析し、ユーザーの関心に基づいて最も関連性の高い広告を表示できます。

このテクニカルレポートで紹介する解決策には、次のようなメリットがあります。

- 分散型または大規模なトレーニングにおける Azure NetApp Files の利点
- CUDA 対応のデータ処理（ cDF 、 cuPy など）と ML アルゴリズム（ cuML ）をラピッズで表示
- 分散型トレーニング用 Dask 並列コンピューティングフレームワーク

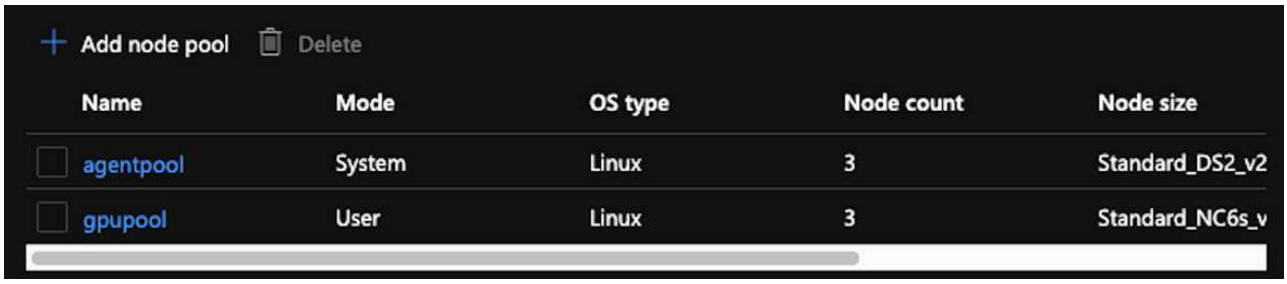
Rapids AI と Azure NetApp Files をベースに構築されたエンドツーエンドのワークフローでは、ランダムフォレストモデルのトレーニング時間が 2 桁単位で大幅に短縮されたことが示されています。この点は、構造化された表形式データが 45 GB （平均）の実世界のクリックログを毎日処理する場合の従来の Pandas アプローチと比べて大幅に改善されています。これは、約 20 億行を含む DataFrame に相当します。このテクニカルレポートでは、クラスタ環境のセットアップ、フレームワークとライブラリのインストール、データのロードと処理、従来型のトレーニングと分散型のトレーニング、可視化と監視について説明し、重要なエンドツーエンドのランタイム結果を比較します。

セットアップ（ Setup ）

AKS クラスタをインストールしてセットアップします

AKS クラスタをインストールしてセットアップする方法については、 Web ページを参照してください "AKS クラスタを作成します" 次に、次の手順を実行します。

1. ノードのタイプ（ system [CPU] ノードまたは worker[GPU] ノード）を選択するときは、次のいずれかを選択します。
   a. プライマリ・システム・ノードは ' 標準 DS2v2 （デフォルトでは 3 ノード）である必要があります
   b. 次に 'gpupool' という名前のユーザ・グループ（ GPU ノードの場合）のワーカー・ノード Standard_NC6s_v3 プール（最小 3 ノード）を追加します



2. 導入には 5 〜 10 分かかります。完了したら、 Connect to Cluster （クラスタへの接続）をクリックします。
3. 新しく作成した AKS クラスタに接続するには、ローカル環境（ラップトップ / PC ）から次のものをインストールします。
   a. を使用した Kubernetes コマンドラインツール "使用している OS に応じた手順が表示されます"
   b. 本ドキュメントに記載されている Azure CLI を使用して、 "Azure CLI をインストールします"
4. 端末から AKS クラスタにアクセスするには、「 AZ login 」と入力し、クレデンシャルを入力します。
5. 次の 2 つのコマンドを実行します。

```
az account set --subscription xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
aks get-credentials --resource-group resourcegroup --name aksclustername
```

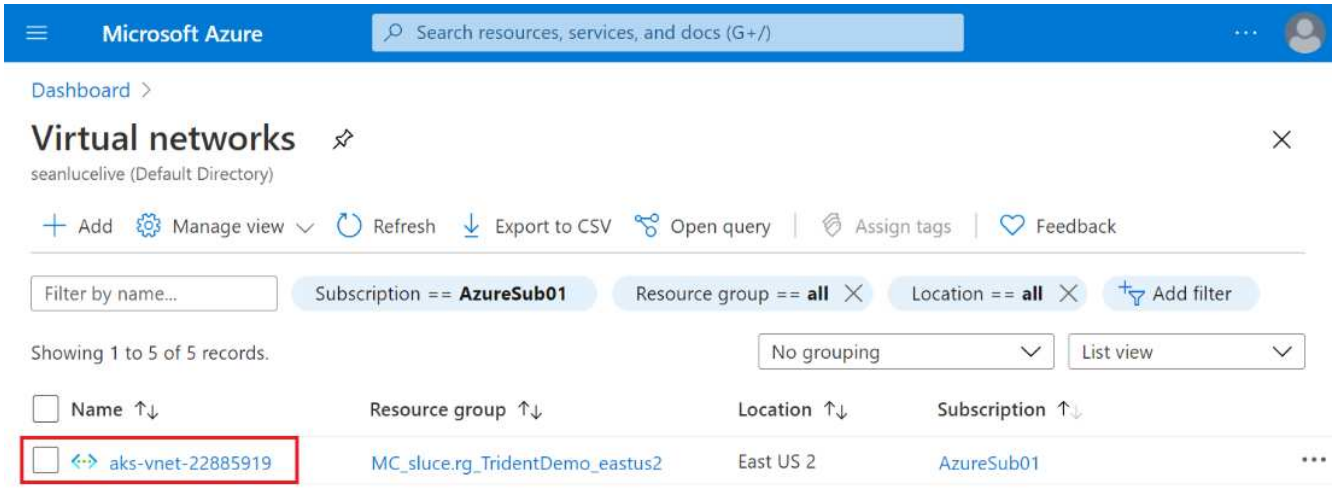
6. 「 Azure CLI ： kubectl get nodes 」と入力します。
7. 次の例に示すように、 6 つのノードがすべて稼働していれば、 AKS クラスタをローカル環境に接続することができます

```
verronmartina@verron-mac-0 ~ % kubectl get nodes
NAME                                STATUS   ROLES   AGE   VERSION
aks-agentpool-34613062-vmss000000   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000001   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000002   Ready    agent   22m   v1.18.14
aks-gpupool-34613062-vmss000000     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000001     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000002     Ready    agent   20m   v1.18.14
verronmartina@verron-mac-0 ~ %
```

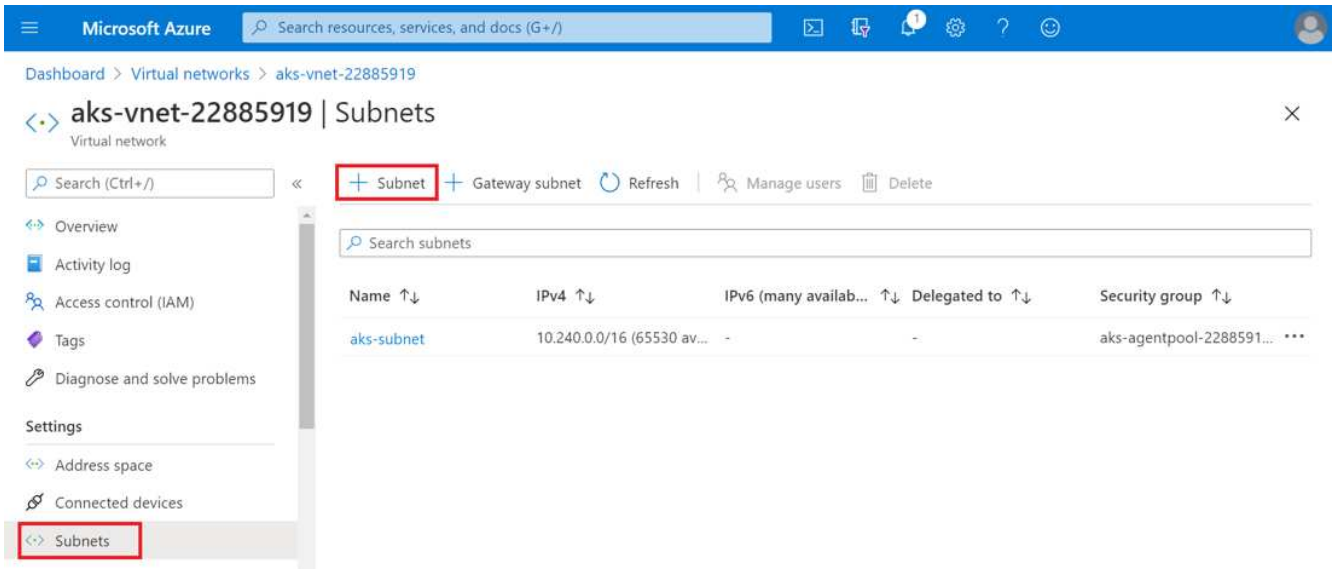
Azure NetApp Files の委譲されたサブネットを作成します

Azure NetApp Files の委任されたサブネットを作成するには、次の手順を実行します。

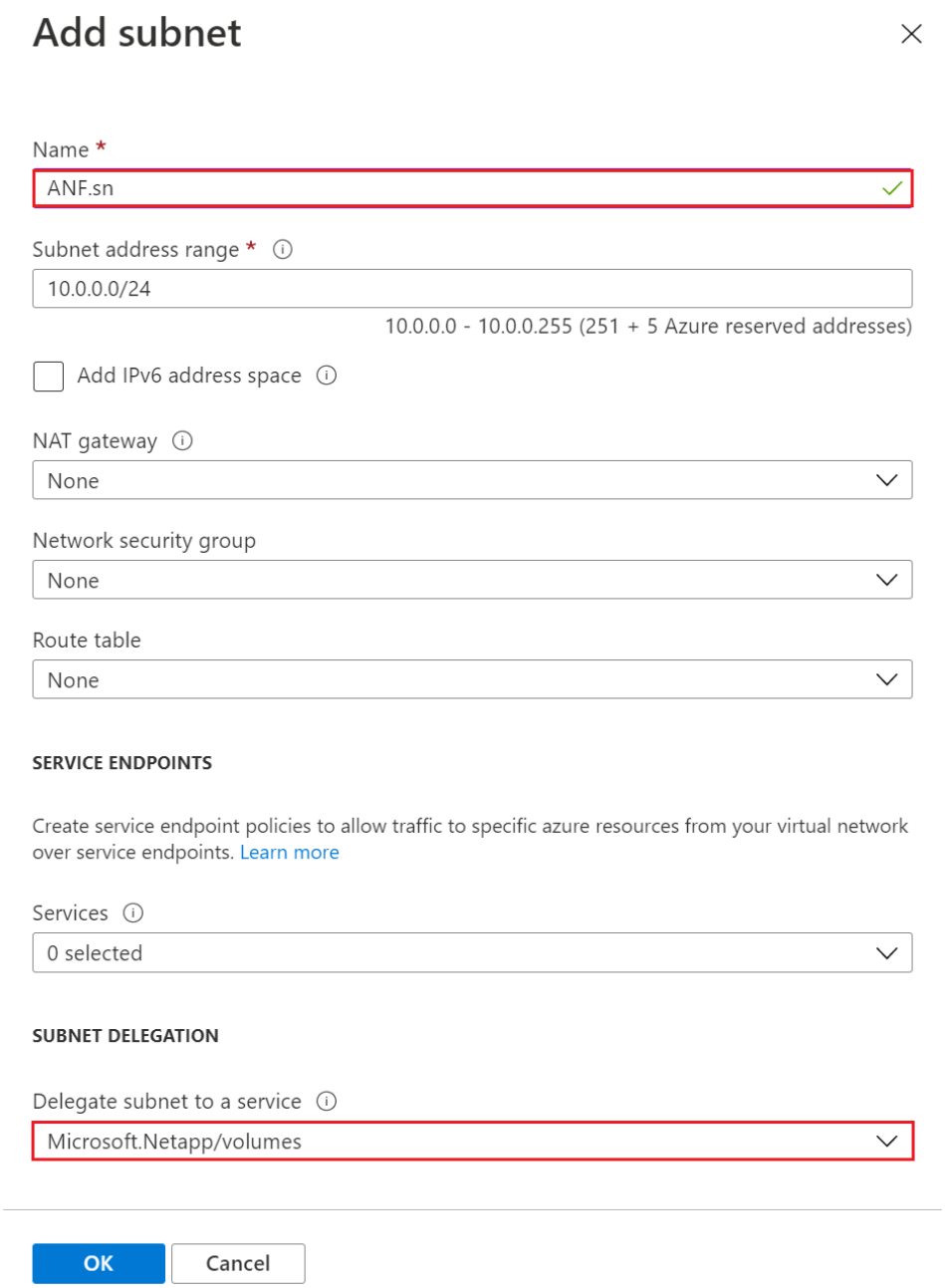
1. Azure ポータル内の仮想ネットワークに移動します。新しく作成した仮想ネットワークを検索します。「AKs-vnet 」などのプレフィックスが必要です。
2. VNet の名前をクリックします。



3. [サブネット] をクリックし、上部のツールバーの [サブネット] をクリックします。



4. サブネットに「 ANF 」などの名前を付け、「サブネットの委任」見出しの下にある「 M icrosoft.Netapp/volumes` 」を選択します。他のものは変更しないでください。[OK] をクリックします。



Azure NetApp Files ボリュームはアプリケーションクラスタに割り当てられ、 Kubernetes で永続ボリューム要求（ PVC ）として使用されます。その結果、 Jupyter ノートブック、サーバーレス関数などのさまざまなサービスに柔軟にマップできます。

サービスのユーザは、プラットフォームのストレージをさまざまな方法で消費できます。このテクニカルレポートでは NFS について説明しているため、 Azure NetApp Files の主なメリットは次のとおりです。

- ユーザに Snapshot コピーを使用できるようにする。
- ユーザが Azure NetApp Files ボリュームに大量のデータを格納できるようにする。
- 大容量のファイルセットでモデルを実行する場合、 Azure NetApp Files のパフォーマンスが向上します。

ピア AKS の VNet と Azure NetApp Files VNet

AKS VNet を Azure NetApp Files VNet にピアリングするには、次の手順を実行します。

1. 検索フィールドに Virtual Networks と入力します。
2. 「 vnet AK - vnet-name 」を選択します クリックして、検索フィールドに peerings と入力します。
3. + Add をクリックします。
4. 次の記述子を入力します。
   a. ピアリングリンク名は 'AKs-vnet-name_-to-anf' です
   b. VNet ピアリングパートナーとしての SubscriptionID および Azure NetApp Files VNet
   c. アスタリスク以外のすべてのセクションは、デフォルト値のままにします。
5. 追加をクリックします。

詳細については、を参照してください "仮想ネットワークピアリングを作成、変更、削除します"。

Trident をインストール

Helm を使用して Trident をインストールするには、次の手順を実行します。

1. Install Helm （インストール手順については、を参照してください） "ソース"）。
2. Trident 20.01.1 インストーラをダウンロードして展開します。

```
$wget
$tar -xf trident-installer-21.01.1.tar.gz
```

3. ディレクトリを 'trident-installer' に変更します

```
$cd trident-installer
```

4. tridentctl' をシステム「 $PATH 」のディレクトリにコピーします。

```
$sudo cp ./tridentctl /usr/local/bin
```

5. Kubernetes （ Kubernetes ）クラスタに Trident をインストールし、 Helm （を参照 "ソース"）：
   a. ディレクトリを 'helm' ディレクトリに変更します

   ```
   $cd helm
   ```

   b. Trident をインストール

```
$helm install trident trident-operator-21.01.1.tgz --namespace
trident --create-namespace
```

c. Trident ポッドのステータスを確認

```
$kubectl -n trident get pods
```

すべてのポッドが稼働中の場合は、 Trident がインストールされてから次のポッドに移動できます。

6. AKS の Azure NetApp Files バックエンドとストレージクラスをセットアップします。

a. Azure サービスプリンシパルを作成します。

サービスプリンシパルは、 Trident が Azure と通信して Azure NetApp Files リソースを操作する方法を示します。

```
$az ad sp create-for-rbac --name ""
```

出力は次の例のようになります。

```
{
"appId": "xxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
"displayName": "netapptrident",
"name": "",
"password": "xxxxxxxxxxxxxxx.xxxxxxxxxxxxxx",
"tenant": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx"
}
```

7. Trident バックエンド JSON ファイルを作成します。例：「 anf-backend.json 」
8. 任意のテキストエディタを使用して 'anf-backend.json ファイル内の次のフィールドに値を入力します

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "fakec765-4774-fake-ae98-a721add4fake",
    "tenantID": "fakef836-edc1-fake-bff9-b2d865eefake",
    "clientID": "fake0f63-bf8e-fake-8076-8de91e57fake",
    "clientSecret": "SECRET",
    "location": "westeurope",
    "serviceLevel": "Standard",
    "virtualNetwork": "anf-vnet",
    "subnet": "default",
    "nfsMountOptions": "vers=3,proto=tcp",
    "limitVolumeSize": "500Gi",
    "defaults": {
    "exportRule": "0.0.0.0/0",
    "size": "200Gi"
}
```

9. 次のフィールドを置き換えます。
   - ' スクリプト ID' 。お客様の Azure サブスクリプション ID
   - 「 tenantID 」。前の手順で「 AZ AD SP 」の出力から取得した Azure テナント ID 。
   - 「 clientID 」。前のステップで 'AZ ad sp' の出力からのあなたの appID 。
   - 「 clientSecret 」を入力します。前の手順で「 AZ ad sp 」の出力から得たパスワード。
10. 構成ファイルとして 'anf-backend.json を使用して 'trident'namespace に Azure NetApp Files バックエンドを作成するように Trident に指示します

```
$tridentctl create backend -f anf-backend.json -n trident
```

```
+------------------------+-------------------+--------------------------------------+--------+---------+
|          NAME          |   STORAGE DRIVER  |                 UUID                 | STATE  | VOLUMES |
+------------------------+-------------------+--------------------------------------+--------+---------+
| azurenetappfiles_86181 | azure-netapp-files | 2ca85462-59ac-4946-be05-c03f5575a2ad | online |       0 |
+------------------------+-------------------+--------------------------------------+--------+---------+
```

11. ストレージクラスを作成する。Kubernetes ユーザは、名前でストレージクラスを指定する PVC を使用してボリュームをプロビジョニングします。前の手順で作成した Trident バックエンドを参照するストレージクラス「 azurenetappfiles 」を作成するよう、 Kubernetes に指示します。
12. ストレージクラスおよびコピー用の YAML （ 'anf-storage-class.yaml ）ファイルを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: azurenetappfiles
provisioner: netapp.io/trident
parameters:
backendType: "azure-netapp-files"
$kubectl create -f anf-storage-class.yaml
```

13. ストレージクラスが作成されたことを確認します。

```
kubectl get sc azurenetappfiles
```

```
NAME              PROVISIONER            RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
azurenetappfiles  csi.trident.netapp.io  Delete          Immediate           false                  98s
```

Helm を使用して、 AKS で Rapids デプロイメントを使用して Dask をセットアップします

Helm を使用して AKS で Rapids を使用して Dask をセットアップするには、次の手順を実行します。

1. Dask with Rapids をインストールするための名前空間を作成します。

```
kubectl create namespace rapids-dask
```

2. クリックスルーレートデータセットを保存する PVC を作成します。
   a. 次の YAML コンテンツをファイルに保存して PVC を作成します。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-criteo-data
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1000Gi
  storageClassName: azurenetappfiles
```

   b. YAML ファイルを Kubernetes クラスタに適用します。

```
kubectl -n rapids-dask apply -f <your yaml file>
```

3. rapidsai git リポジトリを複製します ("https://github.com/rapidsai/helm-chart"）。

```
git clone https://github.com/rapidsai/helm-chart helm-chart
```

4. 値 .yaml を変更し、作業者および Jupyter ワークスペース用に前に作成した PVC を含めます。
   a. リポジトリの 'rapidsai' ディレクトリに移動します

```
cd helm-chart/rapidsai
```

   b. 「 values] .yaml ファイルを更新し、 PVC を使用してボリュームをマウントします。

```
dask:
  …
  worker:
    name: worker
    …
    mounts:
      volumes:
        - name: data
          persistentVolumeClaim:
            claimName: pvc-criteo-data
      volumeMounts:
        - name: data
          mountPath: /data
    …
  jupyter:
    name: jupyter
    …
    mounts:
      volumes:
        - name: data
          persistentVolumeClaim:
            claimName: pvc-criteo-data
      volumeMounts:
        - name: data
          mountPath: /data
    …
```

5. リポジトリのホーム・ディレクトリに移動し 'Helm を使用して AKS 上に 3 つのワーカー・ノードを持つ Dask を展開します

```
cd ..
helm dep update rapidsai
helm install rapids-dask --namespace rapids-dask rapidsai
```

Azure NetApp Files のパフォーマンス階層

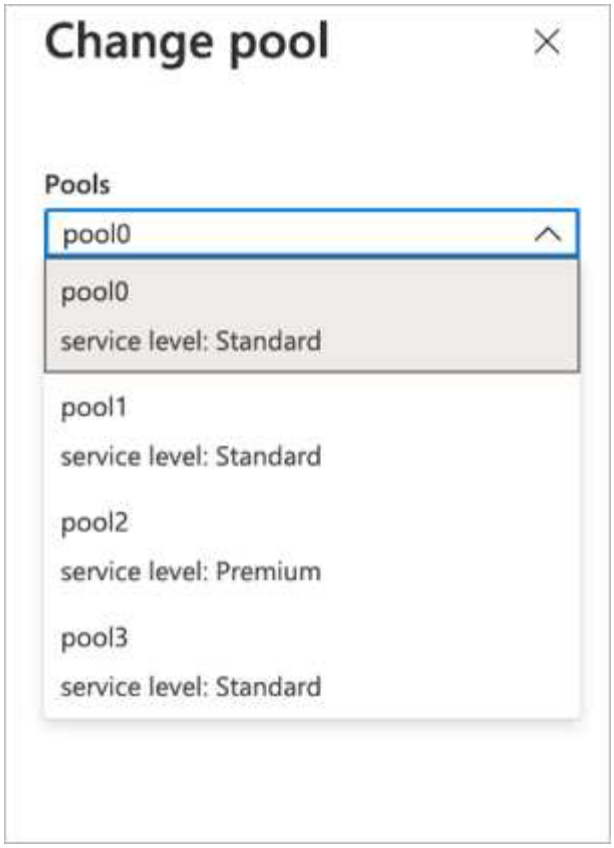
既存のボリュームのサービスレベルを変更するには、そのボリュームに必要なサービスレベルを使用する別の容量プールにボリュームを移動します。この解決策を使用することで、お客様は、まず小規模なデータセットと少数の GPU を標準階層に配置し、データ量と GPU の増加に合わせてスケールアウトまたは Premium Tier へのスケールアップを行うことができます。Premium Tier は、 Standard 階層のテラバイトあたりスループットの 4 倍を提供し、ボリュームのサービスレベルを変更するためにデータを移動することなくスケールアップを実行できます。

ボリュームのサービスレベルを動的に変更する

ボリュームのサービスレベルを動的に変更するには、次の手順を実行します。

1. Volumes （ボリューム）ページで、サービスレベルを変更するボリュームを右クリックします。［ プールの変更 ］ を選択します

2. プールの変更ウィンドウで、ボリュームの移動先となる容量プールを選択します。



3. [OK] をクリックします。

パフォーマンス階層の変更を自動化

パフォーマンス階層の変更を自動化するには、次のオプションを使用できます。

- 現在も動的サービスレベルの変更はパブリックプレビューで有効になっており、デフォルトでは有効になっていません。Azure サブスクリプションでこの機能を有効にする方法については、このドキュメントを参照してください "ボリュームのサービスレベルを動的に変更する"。
- Azure CLI の volume pool change コマンドについては、を参照してください "ボリュームプールの変更に関するドキュメント" 次に例を示します。

```
az netappfiles volume pool-change -g mygroup --account-name myaccname
--pool-name mypoolname --name myvolname --new-pool-resource-id
mynewresourceid
```

- PowerShell "set-AzNetAppFilesVolumePool コマンドレット" Azure NetApp Files ボリュームのプールを変更し、次の例に示すようにします。

```
Set-AzNetAppFilesVolumePool
-ResourceGroupName "MyRG"
-AccountName "MyAnfAccount"
-PoolName "MyAnfPool"
-Name "MyAnfVolume"
-NewPoolResourceId 7d6e4069-6c78-6c61-7bf6-c60968e45fbf
```

クリックスルー率予測データの処理とモデルトレーニング

データ処理およびモデルトレーニング用のライブラリ

次の表に、このタスクの構築に使用されたライブラリとフレームワークを示します。これらのコンポーネントはすべて、 Azure の役割ベースのアクセスおよびセキュリティ制御と完全に統合されています。

ライブラリ / フレームワーク	説明
Dask cuML	ML を GPU で動作させるには、を使用します "cuML ライブラリ" Dask を使用して Rapids cuML パッケージにアクセスできます。Rapids cuML は、クラスタリング、寸法縮小、回帰アプローチなどの一般的な ML アルゴリズムを高性能 GPU ベースの実装で実装し、 CPU ベースのアプローチで最大 100 倍のスピードアップを実現します。
Dask cuDF	cuDF には、データのサブ設定、変換、ワンホットエンコーディングなど、 GPU アクセラレーションによる抽出、変換、読み込み（ ETL ）をサポートするその他のさまざまな機能があります。Rapids チームはを維持する "dask -cudf ライブラリ" これには、 Dask および cuDF を使用するためのヘルパーメソッドが含まれています。
Scikit learn	Scikit-Learn には、数十の機械学習アルゴリズムとモデルが組み込まれています。これらは、試算ツールと呼ばれます。各 "エスティメータ" は、を使用して一部のデータに装着できます "フィット" メソッド

2 つのノートブックを使用して、比較のための ML パイプラインを構築しました。 1 つは従来の Pandas の坐骨坐骨学習アプローチで、もう 1 つは Rapids および Dask との分散トレーニングです。各ノートブックを個別にテストして、パフォーマンスを時間と規模の観点から確認できます。各ノートブックについて個別に説明し、 Rapids および Dask を使用した分散型トレーニングの利点を示します。

Pandas で **Logs Day 15** をクリックして、 **scikit** に学習したランダムフォレストモデルをトレーニングします

このセクションでは、 Pandas と Dask DataFrames を使用して、 Criteo Terabyte データセットから Click Logs データをロードする方法について説明します。このユースケー

スは、広告交換のためのデジタル広告において、広告がクリックされるかどうかを予測したり、交換品が自動化されたパイプラインで正確なモデルを使用していないかどうかを予測したりすることで、ユーザーのプロファイルを作成する場合に適しています。

Click Logs データセットから 15 日目のデータをロードし、合計 45GB にしました。Jupyter ノートブックの ctr-pandasrf-colated で次のセルを実行すると、最初の 5,000 万行を含む Pandas DataFrame が作成され、scikit 学習ランダムフォレストモデルが生成されます。

```
%%time
import pandas as pd
import numpy as np
header = ['col'+str(i) for i in range (1,41)] #note that according to
criteo, the first column in the dataset is Click Through (CT). Consist of
40 columns
first_row_taken = 50_000_000 # use this in pd.read_csv() if your compute
resource is limited.
# total number of rows in day15 is 20B
# take 50M rows
"""
Read data & display the following metrics:
1. Total number of rows per day
2. df loading time in the cluster
3. Train a random forest model
"""
df = pd.read_csv(file, nrows=first_row_taken, delimiter='\t',
names=header)
# take numerical columns
df_sliced = df.iloc[:, 0:14]
# split data into training and Y
Y = df_sliced.pop('col1') # first column is binary (click or not)
# change df_sliced data types & fillna
df_sliced = df_sliced.astype(np.float32).fillna(0)
from sklearn.ensemble import RandomForestClassifier
# Random Forest building parameters
# n_streams = 8 # optimization
max_depth = 10
n_bins = 16
n_trees = 10
rf_model = RandomForestClassifier(max_depth=max_depth,
n_estimators=n_trees)
rf_model.fit(df_sliced, Y)
```

トレーニングされたランダムフォレストモデルを使用して予測を実行するには、このノートブックで次の段落を実行します。重複を避けるために、 15 日目から最後の 100 万行をテストセットとして使用しました。セルはまた、モデルの発生率として定義された予測精度を計算し、ユーザーが広告をクリックするかどうかを正確に予測します。このノートブックの構成部品を確認するには、を参照してください "公式の坐骨神経痛 - 学習

文書"。

```
# testing data, last 1M rows in day15
test_file = '/data/day_15_test'
with open(test_file) as g:
    print(g.readline())

# dataFrame processing for test data
test_df = pd.read_csv(test_file, delimiter='\t', names=header)
test_df_sliced = test_df.iloc[:, 0:14]
test_Y = test_df_sliced.pop('col1')
test_df_sliced = test_df_sliced.astype(np.float32).fillna(0)
# prediction & calculating error
pred_df = rf_model.predict(test_df_sliced)
from sklearn import metrics
# Model Accuracy
print("Accuracy:",metrics.accuracy_score(test_Y, pred_df))
```

Dask の 15 日目をロードし、 Dask cuML ランダムフォレストモデルをトレーニングします

前のセクションと同様に、 Pandas の Logs Day 15 をロードして、 scikit に学習したランダムフォレストモデルをトレーニングします。この例では、 Dask cuDF を使用して DataFrame のロードを実行し、 Dask cuML でランダムなフォレストモデルのトレーニングを行いました。セクションのトレーニング時間と規模の違いを比較しました "「トレーニング時間の比較」"

Crito_dASK_RF.ipynb

このノートブックは、次の例に示すように、「 numpy`, cuml` 」、および必要な「 Ask 」ライブラリをインポートします。

```
import cuml
from dask.distributed import Client, progress, wait
import dask_cudf
import numpy as np
import cudf
from cuml.dask.ensemble import RandomForestClassifier as cumlDaskRF
from cuml.dask.common import utils as dask_utils
```

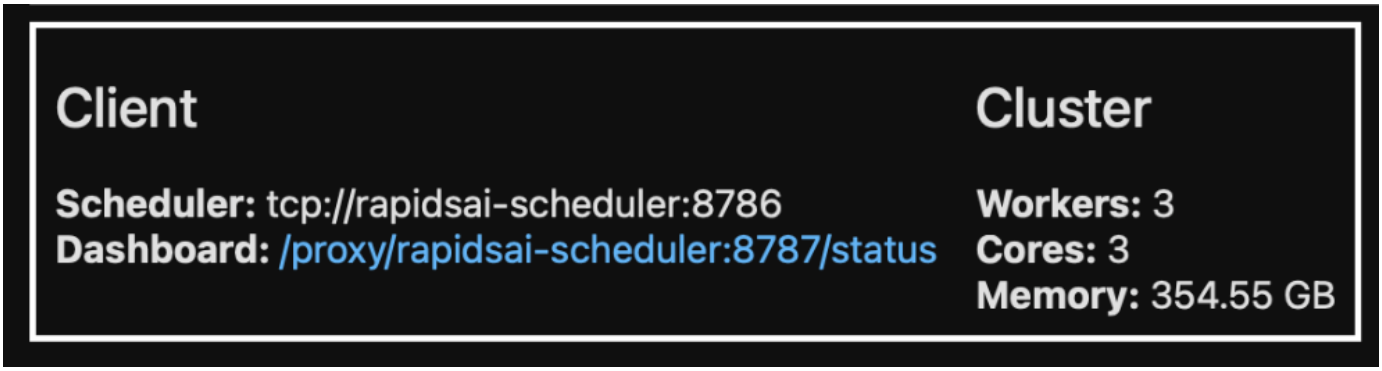
Dask Client() を開始します。

```
client = Client()
```

クラスタが正しく設定されていれば、ワーカーノードのステータスを確認できます。

```
client
workers = client.has_what().keys()
n_workers = len(workers)
n_streams = 8 # Performance optimization
```

AKS クラスタでは、次のステータスが表示されます。



Dask は遅延実行パラダイムを採用しています。処理コードを瞬時に実行するのではなく、 Dask は Directed Acyclic Graph （ DAG ）を実行します。DAG には、各ワーカーが実行する必要のある一連のタスクとそのやり取りが含まれています。このレイアウトは、ユーザーが Dask に一方の方法または別の方法で実行するように指示するまで、タスクが実行されないことを意味します。Dask を使用すると、 3 つの主なオプションがあります。

- * DataFrame 上のコールコンピュート () 。 * このコールはすべてのパーティションを処理し、結果をスケジューラに返して最終的な集約を行い、 cuDF DataFrame に変換します。このオプションは慎重に使用してください。スケジューラノードのメモリが不足しないかぎり、結果が大幅に低下する場合にのみ使用してください。
- * DataFrame 上で Persist() を呼び出します。 * この呼び出しはグラフを実行しますが、スケジューラノードに結果を返すのではなく、クラスタ内で結果をメモリに保持するため、ユーザは同じ処理を再実行することなくパイプライン内で中間結果を再利用できます。
- * DataFrame 上のコールヘッド () 。 * cuDF と同様に、この呼び出しは 10 件のレコードをスケジューラノードに返します。このオプションを使用すると、 DataFrame に目的の出力形式が含まれているかどうか、または処理と計算に応じてレコード自体が適切かどうかをすばやく確認できます。

したがって、ユーザがこれらのアクションのいずれかをコールしない限り、スケジューラが処理を開始するのを待機するアイドル状態になります。この遅延実行パラダイムは、 Apache Spark などの最新の並列および分散コンピューティングフレームワークで一般的です。

次の段落では、 Dask cuML を使用して、 GPU アクセラレーションによる分散コンピューティングを行い、モデル予測精度を計算することにより、ランダムなフォレストモデルのトレーニングを行います。

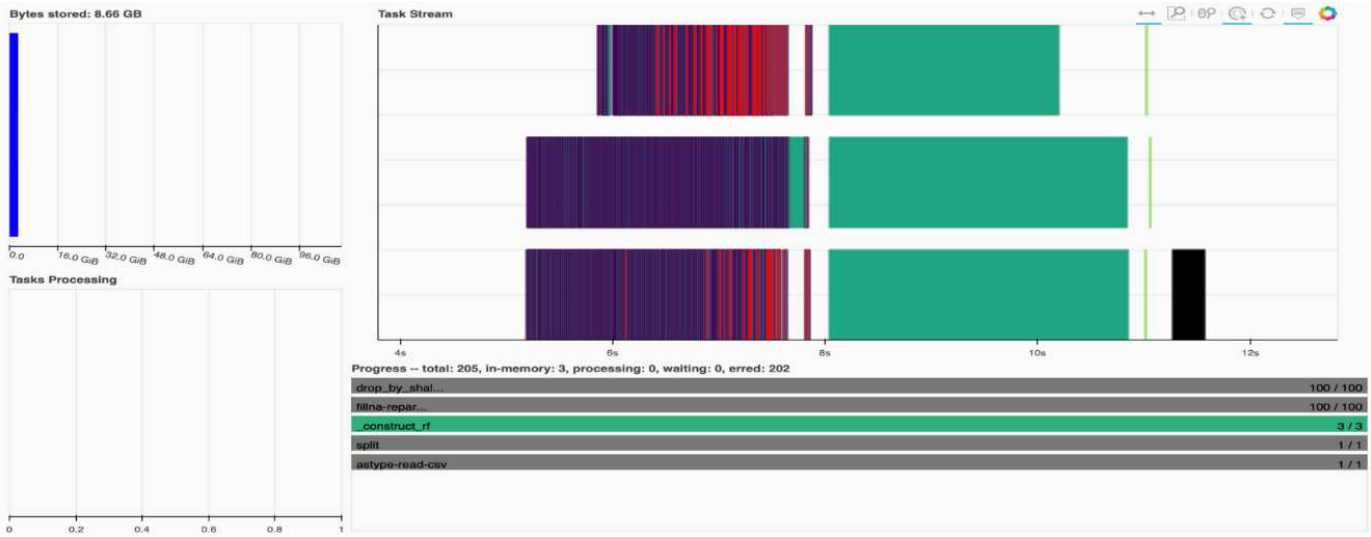
```
Adsf
# Random Forest building parameters
n_streams = 8 # optimization
max_depth = 10
n_bins = 16
n_trees = 10
cuml_model = cumlDaskRF(max_depth=max_depth, n_estimators=n_trees,
n_bins=n_bins, n_streams=n_streams, verbose=True, client=client)
cuml_model.fit(gdf_sliced_small, Y)
# Model prediction
pred_df = cuml_model.predict(gdf_test)
# calculate accuracy
cu_score = cuml.metrics.accuracy_score( test_y, pred_df )
```

ネイティブタスクストリームダッシュボードを使用して **Dask** を監視します

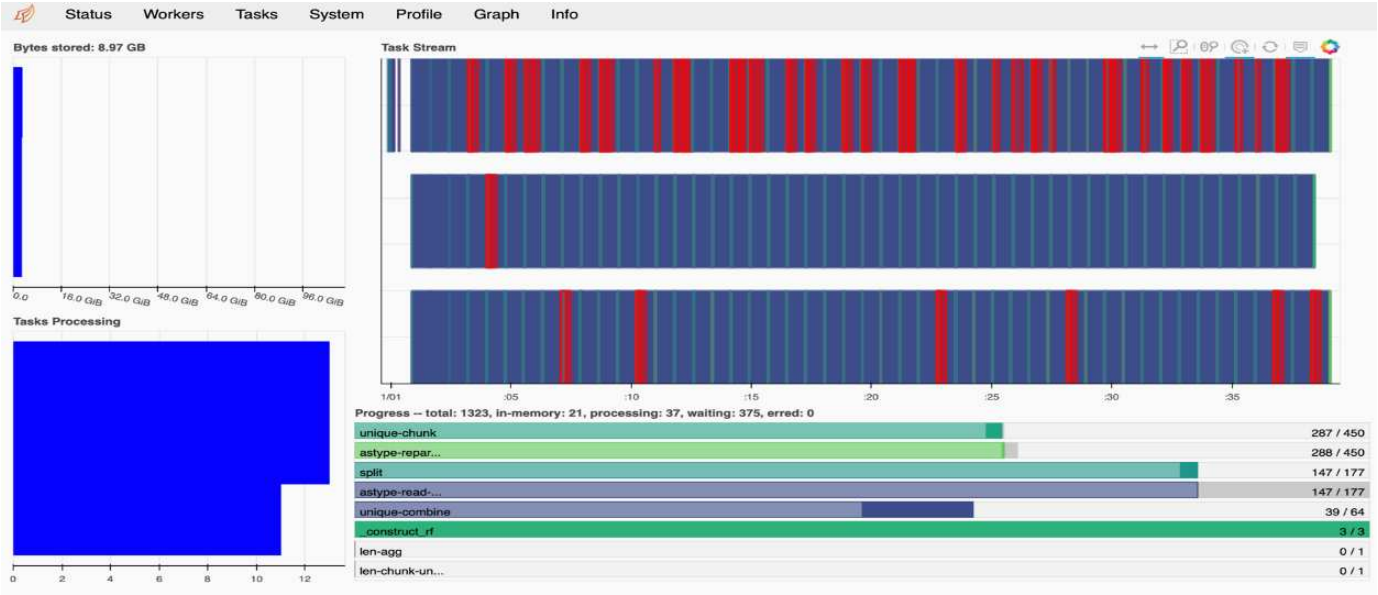
。 "Dask 分散スケジューラ" ライブフィードバックは、次の 2 つの形式で提供します。

- ライブ情報を含む多数のプロットやテーブルを含むインタラクティブなダッシュボード
- コンソールやノートブックでの対話型の使用に適したプログレスバーです

この場合、次の図は、保存されたバイト数、ストリーム数の詳細な内訳を示すタスクストリーム、実行された関連機能を持つタスク名ごとの進捗状況を監視する方法を示しています。この例では、ワーカーノードが 3 つあるため、ストリームには 3 つの主要なチャンクがあり、各ストリーム内で異なるタスクを示すカラーコードがあります。



個々のタスクを分析し、実行時間をミリ秒単位で調査するか、障害や障害を特定することができます。たとえば、次の図は、ランダムフォレストモデルフィッティングステージのタスクストリームを示しています。実行される関数は、 DataFrame 処理用の一意のチャンク、ランダムフォレストをフィッティングするための _construct_RF など、はるかに多くあります。Criteo のクリックログに含まれる 1 日分のデータのサイズ（ 45GB ）が大きいため、 DataFrame の処理にほとんどの時間が費やされていました。



トレーニング時間の比較

このセクションでは、従来の Pandas を使用したモデルのトレーニング時間を Dask と比較します。Pandas では、メモリオーバーフローを回避するために、処理時間が遅くなるため、より少量のデータをロードしました。そのため、結果を補間して公平な比較を行いました。

次の表は、 Pandas ランダムフォレストモデルに使用されるデータが大幅に少ない場合の、生のトレーニング時間の比較を示しています (データセットの 1 日あたりの 2000 億行のうち、 5,000 万行) 。このサンプルでは、使用可能なすべてのデータの 0.25% 未満しか使用されていません。DASK cuML の場合は '20 億行すべての使用可能なローについてランダムフォレストモデルをトレーニングしましたこの 2 つのアプローチでは、同等のトレーニング時間が得られました

アプローチ	トレーニング時間
Scikit - Learn ：トレーニングデータとして day15 の 50 M 行のみを使用します	47 分 21 秒
Rapids-DASK ：トレーニングデータとして、 Day15 のすべての 20B 行を使用します	1 時間 12 分 11 秒

次の表に示すように、トレーニング時間の結果を直線的に補間する場合、 Dask を使用した分散型トレーニングを使用すると大きな利点があります。従来の Pandas の scikit 学習アプローチでは、クリックログ 1 日あたり 45 GB のデータを処理してトレーニングするのに 13 日かかりますが、 Rapids-Dask アプローチでは同じ量のデータを処理するのにかかる時間は 262.39 倍になります。

アプローチ	トレーニング時間
Scikit - Learn ：トレーニングデータとして day15 のすべての 20B 行を使用します	13 日、 3 時間、 40 分、 11 秒
Rapids-DASK ：トレーニングデータとして、 Day15 のすべての 20B 行を使用します	1 時間 12 分 11 秒

前の表では、 Dask と Rapids を使用してデータ処理とモデルトレーニングを複数の GPU インスタンスに分

散することで、従来の Pandas DataFrame 処理と比較して、 scikit 学習モデルトレーニングでの実行時間が大幅に短縮されたことを確認できます。このフレームワークを使用すると、マルチノードのマルチ GPU クラスタ内だけでなく、クラウド内でもオンプレミスでのスケールアップとスケールアウトが可能です。

Prometheus と Grafana で Dask と Rapids を監視します

すべてのデータを導入したら、新しいデータに対して推論を実行します。このモデルは、ユーザーが閲覧アクティビティに基づいて広告をクリックするかどうかを予測します。予測の結果は Dask cuDF に格納されます。Prometheus で結果を監視し、 Grafana ダッシュボードで視覚化できます。

詳細については、を参照してください "Rapids AI 培地ポスト"。

NetApp DataOps ツールキットを使用したデータセットとモデルのバージョン管理

NetApp DataOps Toolkit for Kubernetes は、ストレージリソースと Kubernetes ワークロードをデータサイエンスのワークスペースレベルまで抽象化します。これらの機能は、データサイエンティストとデータエンジニア向けに設計された、使いやすいシンプルなインターフェイスにパッケージ化されています。使い慣れた Python プログラムを使用しており、データサイエンティストやエンジニアは JupyterLab ワークスペースをわずか数秒でプロビジョニングおよび削除できます。これらのワークスペースには、テラバイト、あるいはペタバイト規模のストレージ容量が含まれることがあり、データサイエンティストは、すべてのトレーニングデータセットをプロジェクトのワークスペースに直接格納できます。ワークスペースとデータボリュームを個別に管理する時代は終わりました。

詳細については、ツールキットを参照してください "GitHub リポジトリ"。

Jupyter ノートブックを参考にしてください

このテクニカルレポートには、 Jupyter ノートブックが 2 つ関連付けられています。

- "*CTR - PandasRF - 照合済み。 ipynb. *" このノートブックは Crito Terabyte Logs データセットから 15 日目を読み込み、データを Pandas DataFrame に処理してフォーマットし、 Scikit-learn ランダムフォレストモデルのトレーニングを行い、予測を実行し、精度を計算します。
- "* Crito_dAsk _RF.ipynb.*" このノートブックは Crito Terabyte Logs データセットから 15 日目をロードし、データを Dask cuDF に処理してフォーマットし、 Dask cuML ランダムフォレストモデルのトレーニングを行い、予測を実行し、精度を計算します。GPU を搭載した複数のワーカーノードを活用することで、この分散データとモデルの処理とトレーニングのアプローチを非常に効率的に行うことができます。処理するデータが多いほど、従来の ML アプローチに比べて時間を大幅に節約できます。このノートブックは、ネットワークセットアップによってデータやモデルの配布が自由に移動できる限り、クラウド、オンプレミス、または Kubernetes クラスタにコンピューティングとストレージが異なる場所に配置されているハイブリッド環境に導入できます。

まとめ

Azure NetApp Files 、 Rapids 、 Dask は、 Docker や Kubernetes などのオーケストレ

ーションツールと統合することで、大規模な ML 処理とトレーニングの導入を高速化し、簡易化します。エンドツーエンドのデータパイプラインを統合する解決策ことで、多くの高度なコンピューティングワークロードに特有のレイテンシと複雑さを軽減し、開発と運用のギャップを効果的に解消します。データサイエンティストは、大規模なデータセットでクエリを実行し、トレーニングフェーズ中にデータやアルゴリズムのモデルを他のユーザーと安全に共有できます。

独自の AI / ML パイプラインを構築する場合、アーキテクチャ内のコンポーネントの統合、管理、セキュリティ、およびアクセス性の設定は困難な作業です。開発者に環境へのアクセスと管理を許可することには、もう 1 つの課題があります。

クラウドにエンドツーエンドの分散トレーニングモデルとデータパイプラインを構築することで、 GPU によって高速化されたデータ処理フレームワークやコンピューティングフレームワークを活用していない従来のオープンソースアプローチと比較して、ワークフロー全体の完了時間が 2 桁向上することを実証しました。

ネットアップ、 Microsoft 、オープンソースのオーケストレーションフレームワーク、 NVIDIA を組み合わせることで、最新テクノロジをマネージドサービスとして統合し、優れた柔軟性を実現してテクノロジの採用を促進し、新しい AI / ML アプリケーションの市場投入期間を短縮できます。これらの高度なサービスはクラウドネイティブ環境で提供され、オンプレミス環境やハイブリッド導入アーキテクチャで簡単に移行できます。

追加情報の参照先

このドキュメントに記載されている情報の詳細については、次のリソースを参照してください。

- Azure NetApp Files の特長
  - Azure NetApp Files のソリューションアーキテクチャのページです

    "https://docs.microsoft.com/azure/azure-netapp-files/azure-netapp-files-solution-architectures"

- コンテナ向けの Trident 永続的ストレージ：
  - Azure NetApp Files と Trident

    "https://netapptrident.readthedocs.io/en/stablev20.07/kubernetes/operations/tasks/backends/anf.html"

- Dask および Rapids ：
  - Dask

    "https://docs.dask.org/en/latest/"

  - Dask をインストールします

    "https://docs.dask.org/en/latest/install.html"

  - Dask API

    "https://docs.dask.org/en/latest/api.html"

  - Dask Machine Learning の略

"https://examples.dask.org/machine-learning.html"

- Dask Distributed Diagnostics の実行

  "https://docs.dask.org/en/latest/diagnostics-distributed.html"

- ML フレームワークとツール：
  - TensorFlow ：あらゆる環境に対応するオープンソースの機械学習フレームワーク

    "https://www.tensorflow.org/"

  - Docker です

    "https://docs.docker.com"

  - Kubernetes

    "https://kubernetes.io/docs/home/"

  - クビフロー

    "http://www.kubeflow.org/"

  - Jupyter Notebook Server の 2 つのツールを使用

    "http://www.jupyter.org/"