# **■** NetApp

## アプリケーションの展開 NetApp Solutions

NetApp September 10, 2024

This PDF was generated from https://docs.netapp.com/ja-jp/netapp-solutions/ai/mlrun\_get\_code\_from\_github.html on September 10, 2024. Always check docs.netapp.com for the latest.

## 目次

)'	'プリケーションの展開 · · · · · · · · · · · · · · · · · · ·
	GitHub からコードを取得します
	作業環境を構成します2
	Grafana ダッシュボードを導入します 13

## アプリケーションの展開

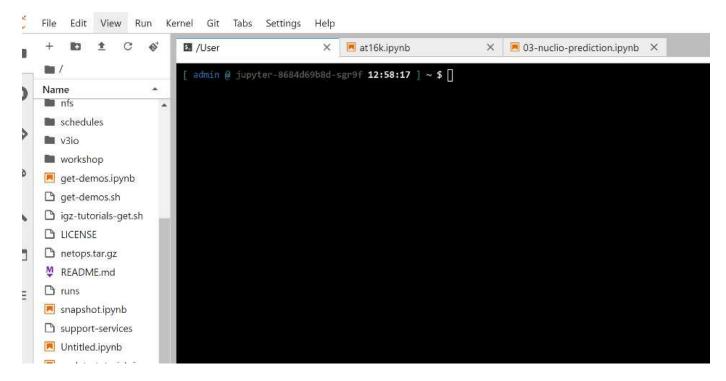
次のセクションでは、アプリケーションのインストールと展開の方法について説明します。

## GitHub からコードを取得します

これで、 Iguazio クラスタおよび開発者環境で NetApp Cloud Volume または NetApp Trident ボリュームを使用できるようになりました。アプリケーションの確認を開始できます。

ユーザは独自のワークスペース(ディレクトリ)を持ちます。すべてのノートブックで ' ユーザー・ディレクトリへのパスは '/User' ですIguazio プラットフォームは、ディレクトリを管理します。上記の手順に従った場合、 NetApp Cloud ボリュームは「 /NetApp 」ディレクトリにあります。

Jupyter 端子を使用して GitHub からコードを取得します。



Jupyter ターミナルのプロンプトで、プロジェクトのクローンを作成します。

```
cd /User
git clone .
```

Jupyter ワークスペースのファイルツリーには、「 NetOps - NetApp 」フォルダが表示されます。

## 作業環境を構成します

「Notebook ````s\_env-example.ipynb を 'et\_env.ipynb` としてコピーします。「et\_env.ipynb」を開き、編集します。このノートブックでは、資格情報、ファイルの場所、および実行ドライバの変数を設定します。

上記の手順を実行すると、次の手順だけが変更されます。

1. この値は、 Iguazio サービスダッシュボード「 dOcker registry 」から取得します

例:「ocker-registry.default-tenant.app.clusterq.iguaziodev.com:80`」

2. 「admin」を Iguazio のユーザ名に変更します。

'IGZ container path='/users/admin'

ONTAP システムの接続の詳細を次に示します。Trident のインストール時に生成されたボリューム名も指定します。オンプレミスの ONTAP クラスタの場合、次の設定が適用されます。

```
ontapClusterMgmtHostname = '0.0.0.0'
ontapClusterAdminUsername = 'USER'
ontapClusterAdminPassword = 'PASSWORD'
sourceVolumeName = 'SOURCE VOLUME'
```

Cloud Volumes ONTAP の設定は次のとおりです。

```
MANAGER=ontapClusterMgmtHostname
svm='svm'
email='email'
password=ontapClusterAdminPassword
weid="weid"
volume=sourceVolumeName
```

#### ベースとなる Docker イメージを作成

ML パイプラインの構築に必要なものはすべて、 Iguazio プラットフォームに含まれています。開発者は、パイプラインの実行に必要な Docker イメージの仕様を定義し、 Jupyter Notebook からイメージの作成を実行できます。ノートブック 'create-images .ipynb' を開き、すべてのセルを実行します。

このノートブックでは、パイプラインで使用する2つのイメージが作成されます。

• 「iguazio/NetApp. 」を参照してください ML タスクの処理に使用されます。

#### Create image for training pipeline

- [4]: fn.build\_config(image=docker\_registry+'/iguazio/netapp', commands=['pip install \
   v3io\_frames fsspec>=0.3.3 PyYAML==5.1.2 pyarrow==0.15.1 pandas==0.25.3 matplotlib seaborn yellowb
   fn.deploy()
  - 「NetApp/pipeline.」。NetApp Snapshot コピーを処理するユーティリティが含まれています。

#### Create image for Ontap utilitites

[8]: fn.build\_config(image=docker\_registry + '/netapp/pipeline:latest',commands=['apt -y update','pip install vilo\_frames netapp\_ontap' fn.deploy()

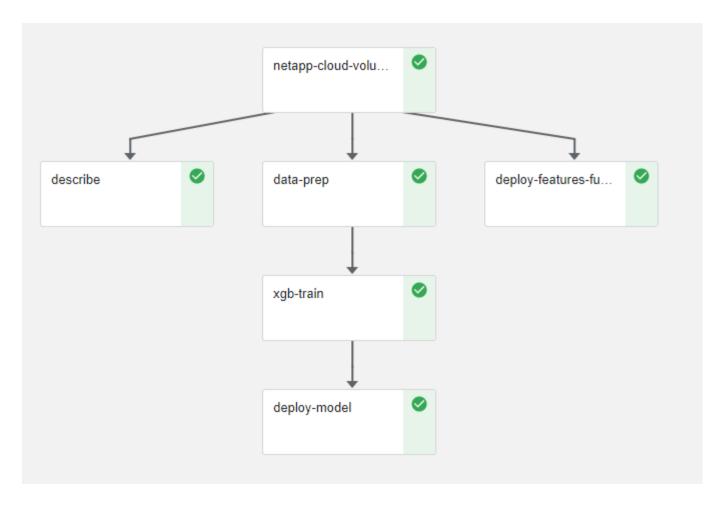
### Jupyter ノートブックを個別に確認します

次の表に、このタスクの構築に使用し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、 Iguazio のロールベース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 <i>I</i> フレームワーク	説明
MLRun ( MLRun )	Iguazio によって管理され、 ML / Al パイプラインの アセンブリ、実行、および監視を可能にします。
Nuclio	Iguazio と統合されたサーバーレス機能フレームワーク。Iguazio が管理するオープンソースプロジェクトとしても利用できます。
クビフロー	パイプラインを導入するための Kubernetes ベースのフレームワーク。これは、イグアスが寄与するオープンソースのプロジェクトでもあります。 Iguazio と統合されているため、他のインフラストラクチャとのセキュリティおよび統合が強化されています。
Docker です	Iguazio プラットフォームでは、 Docker レジストリ がサービスとして実行されます。レジストリに接続す るように変更することもできます。
NetApp Cloud Volume の略	AWS で Cloud Volume を実行すると、大量のデータにアクセスでき、トレーニングに使用するデータセットのバージョンに Snapshot コピーを作成することもできます。
Trident	Trident は、ネットアップが管理するオープンソース プロジェクトです。Kubernetes でのストレージリソ ースやコンピューティングリソースとの統合を簡易化 します。

複数のノートブックを使用して ML パイプラインを構築しました。各ノートブックを個別にテストしてから、パイプラインにまとめてテストすることができます。このデモアプリケーションの導入フローに従って、各ノートブックについて個別に説明します。

望ましい結果は、データの Snapshot コピーに基づいてモデルをトレーニングし、推論のためにモデルを導入 するパイプラインです。完成した MLRun パイプラインのブロック図を次の図に示します。



#### データ生成機能を導入します

このセクションでは、ネットワークデバイスデータの生成に Nuclio サーバーレス関数を使用する方法について説明します。この使用例は、パイプラインを展開し、イグアスのサービスを使用してネットワークデバイスの障害を監視および予測する Iquazio クライアントに適しています。

ネットワークデバイスからのデータをシミュレートしました。Jupyter ノートブック「 d ata-generator.ipynb」を実行すると、 10 分ごとに実行されるサーバーレス関数が作成され、新しいデータが保存された寄木細工のファイルが生成されます。この機能を配備するには、このノートブックのすべてのセルを実行します。を参照してください "Nuclio の Web サイト" このノートブックの構成部品を確認します。

関数の生成時に、次のコメントを持つセルは無視されます。ノートブック内のすべてのセルは、機能の一部であると見なされます。Nuclio モジュールをインポートして '%nuclio magic を有効にします

# nuclio: ignore
import nuclio

関数の仕様では、関数が実行される環境、関数がどのようにトリガされるか、および関数が消費するリソース を定義しました。

「init context」関数は、関数の初期化時に Nuclio フレームワークによって呼び出されます。

```
def init_context(context):
    ....
```

関数内にないコードは、関数が初期化されるときに呼び出されます。この関数を呼び出すと、ハンドラ関数が 実行されます。ハンドラの名前を変更し、関数仕様で指定できます。

```
def handler(context, event):
    ...
```

この機能は、導入前にノートブックからテストできます。

```
%%time
# nuclio: ignore
init_context(context)
event = nuclio.Event(body='')
output = handler(context, event)
output
```

この機能は、ノートブックから導入することも、 CI / CD パイプラインから導入することもできます(このコードを使用)。

```
addr = nuclio.deploy_file(name='generator',project='netops',spec=spec,
tag='v1.1')
```

ノートブックをパイプライン化します

これらのノートブックは、このセットアップで個別に実行することを意図したものではありません。これは、各ノートブックを確認するためのものです。ネットアップは、このような案件をパイプラインの一部として呼び出しました。個別に実行するには、 MLRun のドキュメントを参照して、これらを Kubernetes ジョブとして実行します。

#### snap\_CV.ipynb

このノートブックでは、パイプラインの最初にあるクラウドボリュームの Snapshot コピーを処理します。ボ

リュームの名前をパイプラインコンテキストに渡します。このノートブックは、スナップショットコピーを処理するシェルスクリプトを呼び出します。パイプラインでの実行中、実行コンテキストには、実行に必要なすべてのファイルを見つけるのに役立つ変数が含まれています。このコードを記述する際、開発者は、このコードを実行するコンテナ内のファイルの場所を気にする必要はありません。後で説明したように、このアプリケーションはすべての依存関係とともに配置され、実行コンテキストを提供するパイプラインパラメータの定義です。

```
command = os.path.join(context.get_param('APP_DIR'), "snap_cv.sh")
```

作成された Snapshot コピーの場所は、 MLRun コンテキストに配置され、パイプラインの各ステップで使用されます。

```
context.log_result('snapVolumeDetails',snap_path)
```

次の3つのノートブックは並行して実行されます。

#### データの前処理 ipynb

モデルのトレーニングを有効にするには、生の指標を機能に変換する必要があります。このノートでは、 Snapshot ディレクトリから生の指標を読み取り、モデルトレーニングの機能をネットアップボリュームに書 き込みます。

パイプラインのコンテキストで実行する場合、「 Data ATA\_DIR 」という入力には Snapshot コピーの場所が含まれます。

#### .ipynb を説明する

受信メトリックを視覚化するために、 Kubeflow UI と MLRun UI で使用できるプロットとグラフを提供するパイプラインステップを導入します。各実行には、この表示ツールの独自のバージョンがあります。

```
ax.set_title("features correlation")
plt.savefig(os.path.join(base_path, "plots/corr.png"))
context.log_artifact(PlotArtifact("correlation", body=plt.gcf()),
local_path="plots/corr.html")
```

#### deploy-feature-function.ipynb

ネットアップでは、異常を検出している指標を継続的に監視してこのノートブックは、受信メトリックの予測を実行するために必要な機能を生成するサーバーレス機能を作成します。このノートブックは関数の作成を呼び出します。ファンクションコードはノートブック「 ata-prep . ipynb 」にあります。この目的のために、パ

イプラインのステップとして同じノートブックを使用していることに注意してください。

#### train.ipynb

フィーチャーを作成した後、モデルトレーニングを開始します。このステップの出力は、推論に使用するモデルです。また、統計を収集して各実行を追跡します(実験)。

たとえば、次のコマンドは、その測定条件のコンテキストに精度スコアを入力します。この値は Kubeflow および MLRun で確認できます。

context.log result('accuracy', score)

#### deploy-inion-function.ipynb を展開します

パイプラインの最後のステップは、継続的な推論のためのサーバーレス機能としてモデルを導入することです。このノートブックでは、「 nuclio-increation-function.ipynb 」で定義されたサーバーレス関数の作成を呼び出します。

#### パイプラインのレビューと構築

パイプラインですべてのノートブックを実行するという組み合わせにより ' テストを継続的に実行して ' モデルの精度を新しいメトリックと比較して再評価することができますまず 'pipeline.ipynb' ノートブックを開きますネットアップと Iguazio が ML パイプラインの導入をどのように簡易化しているかを詳しく説明します。

MLRun を使用して、パイプラインの各ステップにコンテキストを提供し、リソースの割り当てを処理します。MLRun API サービスは、 Iguazio プラットフォームで動作し、 Kubernetes リソースとのやり取りのポイントです。各開発者はリソースを直接要求できません。 API は要求を処理し、アクセス制御を有効にします。

# MLRun API connection definition
mlconf.dbpath = 'http://mlrun-api:8080'

パイプラインは、 NetApp Cloud Volume やオンプレミスのボリュームと連携できます。このデモでは Cloud Volume を使用するように設計しましたが、オンプレミスで実行できるオプションをコードに示しています。

```
# Initialize the NetApp snap fucntion once for all functions in a notebook
if [ NETAPP CLOUD VOLUME ]:
    snapfn =
code to function('snap',project='NetApp',kind='job',filename="snap cv.ipyn
b").apply(mount v3io())
    snap params = {
    "metrics table" : metrics table,
    "NETAPP MOUNT PATH" : NETAPP MOUNT PATH,
    'MANAGER' : MANAGER,
    'svm' : svm,
    'email': email,
    'password': password ,
    'weid': weid,
    'volume': volume,
    "APP DIR" : APP DIR
else:
    snapfn =
code to function('snap',project='NetApp',kind='job',filename="snapshot.ipy
nb").apply(mount v3io())
snapfn.spec.image = docker registry + '/netapp/pipeline:latest'
snapfn.spec.volume mounts =
[snapfn.spec.volume mounts[0], netapp volume mounts]
      snapfn.spec.volumes = [ snapfn.spec.volumes[0],netapp volumes]
```

Jupyter ノートブックを Kubeflow ステップにするために必要な最初のアクションは、コードを関数に変換することです。関数には、ノートブックを実行するために必要なすべての仕様が含まれています。ノートブックを下にスクロールすると、パイプラインのすべてのステップに対応する関数が定義されていることがわかります。

ノートブックの一部	説明		
<code_to _function=""> ( MLRun モジュールの一部)</code_to>	関数の名前:プロジェクト名。すべてのプロジェクトアーティファクトの編成に使用されます。これはMLRun UI に表示されます。種類:この場合はKubernetes ジョブ。これには、Dask 、MPI、spark8s などがあります。詳細については、MLRunのマニュアルを参照してください。ファイル。ノートブックの名前。これは Git (HTTP)の場所にすることもできます。		
イメージ (Image)	この手順で使用する Docker イメージの名前。先ほど 'create-image.ipynb ノートブックを作成しました		
volume_mounts と volumes	実行時に NetApp Cloud Volume をマウントするための詳細情報。		

また、ステップのパラメーターも定義します。

```
"FEATURES TABLE": FEATURES TABLE,
params={
           "SAVE TO" : SAVE TO,
           "metrics_table" : metrics_table,
           'FROM TSDB': 0,
           'PREDICTIONS TABLE': PREDICTIONS TABLE,
           'TRAIN ON LAST': '1d',
           'TRAIN SIZE':0.7,
           'NUMBER OF SHARDS' : 4,
           'MODEL FILENAME' : 'netops.v3.model.pickle',
           'APP DIR' : APP DIR,
           'FUNCTION NAME' : 'netops-inference',
           'PROJECT NAME' : 'netops',
           'NETAPP SIM' : NETAPP SIM,
           'NETAPP MOUNT PATH': NETAPP MOUNT PATH,
           'NETAPP PVC CLAIM' : NETAPP PVC CLAIM,
           'IGZ CONTAINER PATH' : IGZ CONTAINER PATH,
           'IGZ MOUNT PATH' : IGZ MOUNT PATH
```

すべてのステップの関数定義が完了したら、パイプラインを構築できます。この定義には 'kfp' モジュールを使用しますMLRun を使用することと、独自に構築することの違いは、コーディングの簡素化と短縮です。

定義した関数は、 MLRun の「 As \_ step 」関数を使用してステップコンポーネントになります。

スナップショットステップの定義

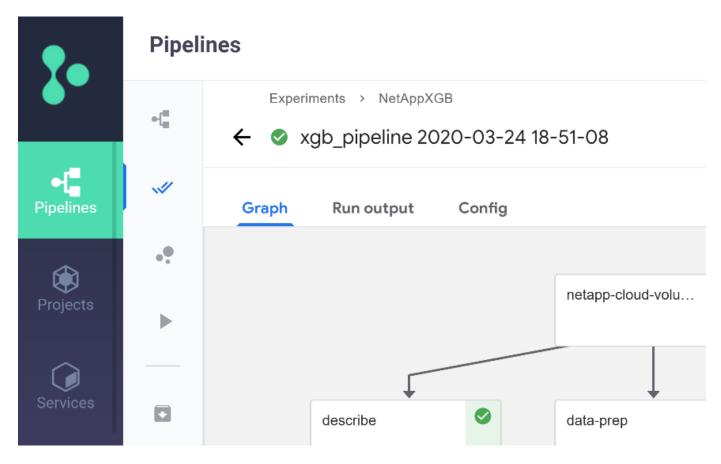
Snapshot 機能を開始し、 v3io をソースとしてマウントします。

```
snap = snapfn.as_step(NewTask(handler='handler',params=snap_params),
name='NetApp_Cloud_Volume_Snapshot',outputs=['snapVolumeDetails','training
_parquet_file']).apply(mount_v3io())
```

パラメータ	詳細		
新しいタスクです	newtask は、実行される関数の定義です。		
(MLRun モジュール)	ハンドラ。呼び出す Python 関数の名前。ノートブックでは name ハンドラーを使用しましたが、必須ではありません。パラメータ実行に渡されたパラメータ。このコードでは、 context.get_param (「パラメータ」)を使用して値を取得します。		
ステップとして(_STEP.)	名前Kubeflow パイプラインステップの名前。出力: これらは、完了時にステップが辞書に追加する値で す。SNAP_CV.ipynb ノートブックを参照してくださ い。mount_v3io()これにより、パイプラインを実行し ているユーザーの /User をマウントするステップが構 成されます。		

パラメータ	詳細		
入力	前の手順の出力に渡すことができます。この場合、 snap.outputs['napVolumeDetails'] は、スナップステッ プで作成した Snapshot コピーの名前です。		
out_path	MLRun モジュール LOG_Artifacts を使用して生成するアーティファクトを配置する場所。		

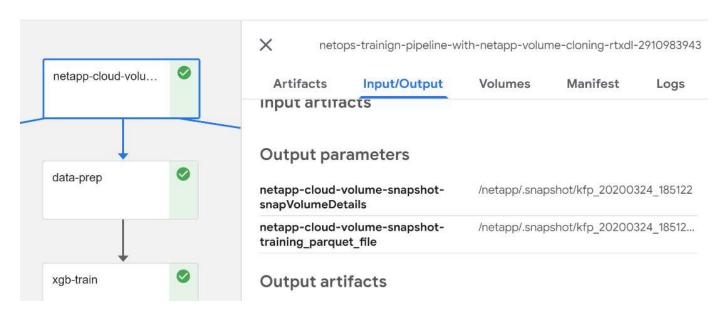
上から下に 'pipeline.ipynb' を実行できます次に、 Iguazio ダッシュボードの Pipelines タブに移動して、 Iguazio ダッシュボードの Pipelines タブに示すように、進捗状況を監視できます。



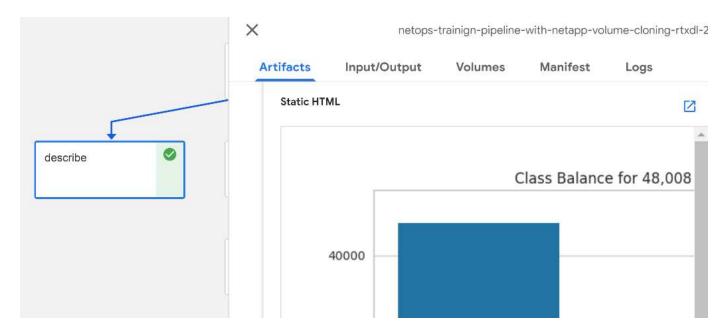
トレーニングステップの精度はすべての実行で記録されているため、トレーニングの正確性の記録に示されているように、各テストの精度の記録があります。

Run name	Status	Duration	Pipeline Version	Recurring	Start time	accuracy
xgb_pipeline 2020-03-24 18-51	0	0:08:43	[View pipeline]	180	3/24/2020, 2:51:09 PM	0.985
xgb_pipeline 2020-03-19 13-31	9	0:08:14	[View pipeline]		3/19/2020, 9:31:19 AM	0.980
xgb_pipeline 2020-03-18 12-56	0	0:08:11	[View pipeline]	9.61	3/18/2020, 8:56:08 AM	0.990
xgb_pipeline 2020-03-17 19-49	0	0:08:03	[View pipeline]		3/17/2020, 3:49:31 PM	0.985
xgb_pipeline 2020-03-17 18-34	0	0:05:54	[View pipeline]		3/17/2020, 2:34:56 PM	0.980
xgb_pipeline 2020-03-17 17-34	0	0:04:48	[View pipeline]		3/17/2020, 1:34:16 PM	0.982
xgb_pipeline 2020-03-17 17-01	0	0:05:25	[View pipeline]	100	3/17/2020, 1:01:58 PM	0.987
xgb_pipeline 2020-03-16 16-47	0	0:06:08	[View pipeline]		3/16/2020, 12:47:19	0.983
xgb_pipeline 2020-03-16 13-57	0	0:05:18	[View pipeline]	*	3/16/2020, 9:57:03 AM	0.980

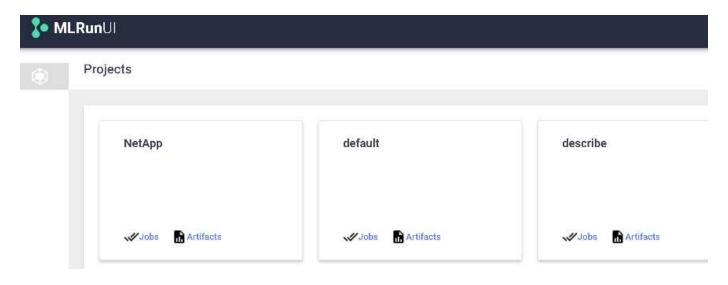
Snapshot ステップを選択すると、この実験を実行するために使用された Snapshot コピーの名前が表示されます。



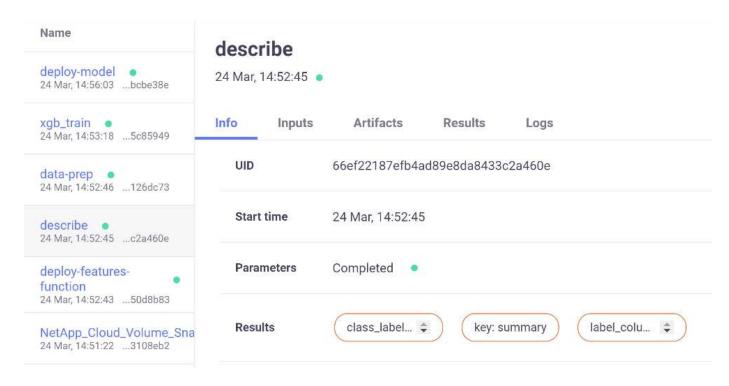
ここで説明する手順には、使用した指標を確認するための視覚的なアーティファクトがあります。を展開すると、次の図のように全プロットを表示できます。



MLRun API データベースは、プロジェクトごとに編成された各ランの入力、出力、およびアーティファクトも追跡します。各ランの入力、出力、およびアーティファクトの例を次の図に示します。



各ジョブについて、追加の詳細情報が保存されます。



MLRun の詳細については、このドキュメントで説明している内容を参照してください。ステップと関数の定義を含むアルアーティファクトは、API データベースに保存したり、バージョン管理したり、個別に呼び出すことも、完全なプロジェクトとして呼び出すこともできます。プロジェクトを保存して Git にプッシュし、後で使用することもできます。詳細については、を参照してください "MLRun GitHub サイト"。

## Grafana ダッシュボードを導入します

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、ネットワークデバイス機器の障害を予測します。予測の結果は、 Iguazio 時系列テーブルに格納されます。Iguazio のセキュリティおよびデータアクセスポリシーと統合されたプラットフォームで、 Grafana を使用して結果を表示できます。

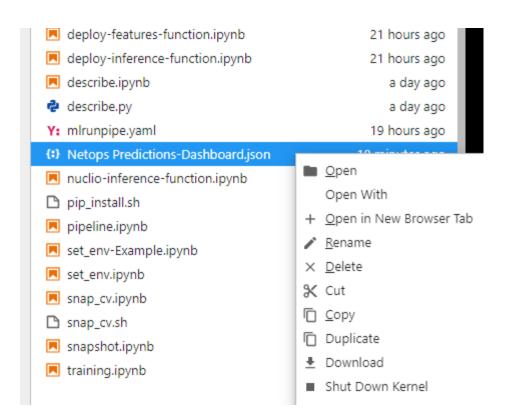
ダッシュボードを導入するには、指定した JSON ファイルをクラスタ内の Grafana インターフェイスにインポートします。

1. Grafana サービスが実行されていることを確認するには、 Services の下を参照してください。

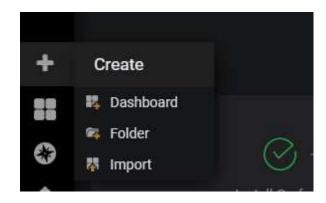
#### Services



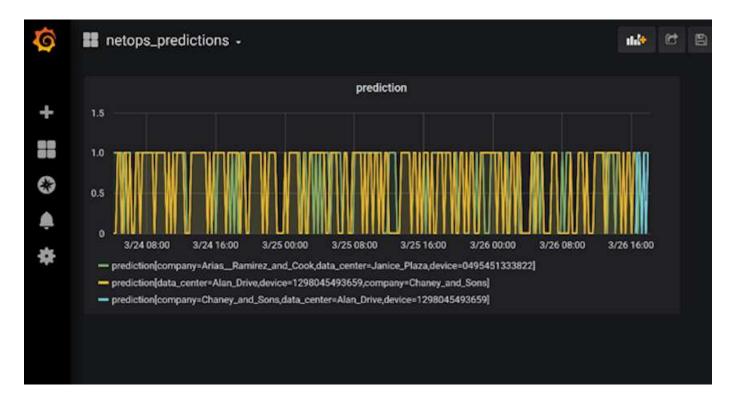
- 2. インスタンスが存在しない場合は、[サービス]セクションからインスタンスを展開します。
  - a. [新しいサービス]をクリックします。
  - b. リストから Grafana を選択します。
  - C. デフォルトを受け入れます。
  - d. 次のステップをクリックします。
  - e. ユーザ ID を入力します。
  - f. [ サービスの保存 ] をクリックします
  - 9. 上部の [Apply Changes] をクリックします。
- 3. ダッシュボードを展開するには、 Jupyter インターフェイスから「 NetopsPredictions Dashboard.json 」ファイルをダウンロードします。



4. Services セクションで Grafana を開き、ダッシュボードをインポートします。



5. Upload `\*.json ' File をクリックして、以前にダウンロードしたファイル (`NetopsPredictions - Dashboard.json ') を選択します。アップロードが完了すると、ダッシュボードが表示されます。



#### デプロイクリーンアップ機能

大量のデータを生成する場合は、すべてをクリーンで整理することが重要です。これを行うには 'cleanup.ipynb' ノートブックを使用してクリーンアップ機能を配備します

#### 利点

ネットアップと Iguazio は、 Kubeflow 、 Apache Spark 、 TensorFlow などの必須フレームワークや、 Docker や Kubernetes などのオーケストレーションツールを構築することで、 AI アプリケーションや ML アプリケーションの導入を迅速化し、簡易化します。ネットアップと Iguazio は、エンドツーエンドのデータパイプラインを統合することで、多くの高度なコンピューティングワークロードに固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に対してクエリを実行し、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権限のあるユーザと安全に共有できます。コンテナ化されたモデルを本番環境で使用できるようになったら、開発環境から運用環境に簡単に移行できます。

#### 著作権に関する情報

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S.このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為(過失またはそうでない場合を含む)にかかわらず、一切の責任を負いません。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。 ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じ る責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップ の特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1つ以上の米国特許、その他の国の特許、および出願中の特許によって保護されている場合があります。

権利の制限について:政府による使用、複製、開示は、DFARS 252.227-7013(2014年2月)およびFAR 5252.227-19(2007年12月)のRights in Technical Data -Noncommercial Items(技術データ - 非商用品目に関する諸権利)条項の(b)(3)項、に規定された制限が適用されます。

本書に含まれるデータは商用製品および / または商用サービス(FAR 2.101の定義に基づく)に関係し、データの所有権はNetApp, Inc.にあります。本契約に基づき提供されるすべてのネットアップの技術データおよびコンピュータ ソフトウェアは、商用目的であり、私費のみで開発されたものです。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用権を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用権については、DFARS 252.227-7015(b)項(2014年2月)で定められた権利のみが認められます。

#### 商標に関する情報

NetApp、NetAppのロゴ、http://www.netapp.com/TMに記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。