Tracking classification Nomodel training with MLflow

Juan Sebastián Peñaloza Quintana*

November 24, 2024

1 Introduction

You can use MLflow in a notebook to track any models you train. As you'll run this notebook with an Azure Machine Learning compute instance, you don't need to set up MLflow: it's already installed and integrated.

You'll prepare some data and train a model to predict diabetes. You'll use autologging, and custom logging to explore how you can use MLflow in notebooks.

1.1 Before you start

You'll need the latest version of the **azure-ai-ml** package to run the code in this notebook. Run the cell below to verify that it is installed.

> **Note**: If the **azure-ai-ml** package is not installed, run `pip install azure-ai-ml` to install it.

[1]:
```
pip show azure-ai-ml
```

```
Name: azure-ai-ml
Version: 1.22.4
Summary: Microsoft Azure Machine Learning Client Library for Python
Home-page: https://github.com/Azure/azure-sdk-for-python
Author: Microsoft Corporation
Author-email: azuresdkengsysadmins@microsoft.com
License: MIT License
Location: /anaconda/envs/azureml_py38/lib/python3.10/site-packages
Requires: azure-common, azure-core, azure-mgmt-core, azure-storage-blob, azure-
storage-file-datalake, azure-storage-file-share, colorama, isodate, jsonschema,
marshmallow, msrest, opencensus-ext-azure, opencensus-ext-logging, pydash,
pyjwt, pyyaml, strictyaml, tqdm, typing-extensions
Required-by:
Note: you may need to restart the kernel to use updated packages.
```

*jsebastianDS@proton.me

1.2 Connect to your workspace

With the required SDK packages installed, now you're ready to connect to your workspace.

To connect to a workspace, we need identifier parameters - a subscription ID, resource group name, and workspace name. Since you're working with a compute instance, managed by Azure Machine Learning, you can use the default values to connect to the workspace.

[2]:
```
from azure.identity import DefaultAzureCredential, InteractiveBrowserCredential
from azure.ai.ml import MLClient

try:
    credential = DefaultAzureCredential()
    # Check if given credential can get token successfully.
    credential.get_token("https://management.azure.com/.default")
except Exception as ex:
    # Fall back to InteractiveBrowserCredential in case DefaultAzureCredential␣
 ↪not work
    credential = InteractiveBrowserCredential()
```

[3]:
```
# Get a handle to workspace
ml_client = MLClient.from_config(credential=credential)
```

```
Found the config file in: /config.json
```

1.3 Configure MLflow

As you're running this notebook on a compute instance in the Azure Machine Learning studio, you don't need to configure MLflow.

Still, it's good to verify that the necessary library is indeed installed.

> **Note**: If the **mlflow** library is not installed, run `pip install mlflow` to install it.

[4]:
```
pip show mlflow
```

```
Name: mlflow
Version: 2.18.0
Summary: MLflow is an open source platform for the complete machine learning
License: Copyright 2018 Databricks, Inc.  All rights reserved.

                                 Apache License
                           Version 2.0, January 2004
                        http://www.apache.org/licenses/

   TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
```

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the

purposes
of this License, Derivative Works shall not include works that
remain
separable from, or merely link (or bind by name) to the interfaces
of,
the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including
the original version of the Work and any modifications or
additions
to that Work or Derivative Works thereof, that is intentionally
submitted to Licensor for inclusion in the Work by the copyright
owner
or by an individual or Legal Entity authorized to submit on behalf
of
the copyright owner. For the purposes of this definition,
"submitted"
means any form of electronic, verbal, or written communication
sent
to the Licensor or its representatives, including but not limited
to
communication on electronic mailing lists, source code control
systems,
and issue tracking systems that are managed by, or on behalf of,
the
Licensor for the purpose of discussing and improving the Work, but
excluding communication that is conspicuously marked or otherwise
designated in writing by the copyright owner as "Not a
Contribution."

"Contributor" shall mean Licensor and any individual or Legal
Entity
on behalf of whom a Contribution has been received by Licensor and
subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
this License, each Contributor hereby grants to You a perpetual,
worldwide, non-exclusive, no-charge, royalty-free, irrevocable
copyright license to reproduce, prepare Derivative Works of,
publicly display, publicly perform, sublicense, and distribute the
Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
this License, each Contributor hereby grants to You a perpetual,
worldwide, non-exclusive, no-charge, royalty-free, irrevocable
(except as stated in this section) patent license to make, have
made,
use, offer to sell, sell, import, and otherwise transfer the Work,

where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor

has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by
reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS
APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!) The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
See the License for the specific language governing permissions and
limitations under the License.

Location: /anaconda/envs/azureml_py38/lib/python3.10/site-packages
Requires: alembic, docker, Flask, graphene, gunicorn, Jinja2, markdown,
matplotlib, mlflow-skinny, numpy, pandas, pyarrow, scikit-learn, scipy,
sqlalchemy
Required-by:
Note: you may need to restart the kernel to use updated packages.

1.4 Prepare the data

You'll train a diabetes classification model. The training data is stored in the **data** folder as **diabetes.csv**.

First, let's read the data:

[5]:
```
import pandas as pd

print("Reading data...")
df = pd.read_csv('./data/diabetes.csv')
df.head()
```

```
Reading data...
```

[5]:

	PatientID	Pregnancies	PlasmaGlucose	DiastolicBloodPressure	\
0	1354778	0	171	80	
1	1147438	8	92	93	
2	1640031	7	115	47	
3	1883350	9	103	78	
4	1424119	1	85	59	

	TricepsThickness	SerumInsulin	BMI	DiabetesPedigree	Age	Diabetic
0	34	23	43.509726	1.213191	21	0
1	47	36	21.240576	0.158365	23	0
2	52	35	41.511523	0.079019	23	0
3	25	304	29.582192	1.282870	43	1
4	27	35	42.604536	0.549542	22	0

Next, you'll split the data into features and the label (Diabetes):

[6]:
```
print("Splitting data...")
X, y =␣
 ↪df[['Pregnancies','PlasmaGlucose','DiastolicBloodPressure','TricepsThickness','SerumInsulin',
 ↪values, df['Diabetic'].values
```

```
Splitting data...
```

[7]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,␣
 ↪random_state=0)
```

You now have four dataframes:

- `X_train`: The training dataset containing the features.
- `X_test`: The test dataset containing the features.
- `y_train`: The label for the training dataset.
- `y_test`: The label for the test dataset.

You'll use these to train and evaluate the models you'll train.

1.5 Create an MLflow experiment

Now that you're ready to train machine learning models, you'll first create an MLflow experiment. By creating the experiment, you can group all runs within one experiment and make it easier to find the runs in the studio.

[8]:
```
import mlflow
experiment_name = "mlflow-experiment-diabetes"
mlflow.set_experiment(experiment_name)
```

```
2024/11/23 21:01:14 INFO mlflow.tracking.fluent: Experiment with name 'mlflow-
experiment-diabetes' does not exist. Creating a new experiment.
```

[8]:
```
<Experiment: artifact_location='', creation_time=1732395674641,
experiment_id='6115d7a9-818a-473a-aa63-39618382a63e', last_update_time=None,
lifecycle_stage='active', name='mlflow-experiment-diabetes', tags={}>
```

1.6 Train and track models

To track a model you train, you can use MLflow and enable autologging. The following cell will train a classification model using logistic regression. You'll notice that you don't need to calculate any evaluation metrics because they're automatically created and logged by MLflow.

[9]:
```
from sklearn.linear_model import LogisticRegression

with mlflow.start_run():
    mlflow.sklearn.autolog()

    model = LogisticRegression(C=1/0.1, solver="liblinear").fit(X_train, y_train)
```

```
View run joyful_boot_y5vsbbt0 at: https://eastus.api.azureml.ms/mlflow/v2.0/su
bscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01
dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l7
8ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e/runs/0a9c72
7e-3bd5-422a-afcc-80b780558938
View experiment at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0
108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/p
roviders/Microsoft.MachineLearningServices/workspaces/mlw-
dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e
```

You can also use custom logging with MLflow. You can add custom logging to autologging, or you can use only custom logging.

Let's train two more models with scikit-learn. Since you ran the `mlflow.sklearn.autolog()` command before, MLflow will now automatically log any model trained with scikit-learn. To disable the autologging, run the following cell:

[10]:
```
mlflow.sklearn.autolog(disable=True)
```

Now, you can train and track models using only custom logging.

When you run the following cell, you'll only log one parameter and one metric.

[11]:
```
from sklearn.linear_model import LogisticRegression
import numpy as np

with mlflow.start_run():
    model = LogisticRegression(C=1/0.1, solver="liblinear").fit(X_train, y_train)

    y_hat = model.predict(X_test)
    acc = np.average(y_hat == y_test)

    mlflow.log_param("regularization_rate", 0.1)
    mlflow.log_metric("Accuracy", acc)
```

```
 View run joyful_glove_ljbctt5h at: https://eastus.api.azureml.ms/mlflow/v2.0/s
ubscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee0
1dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l
78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e/runs/4200f
c78-0728-467b-84fb-68e94870ad51
View experiment at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0
108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/p
roviders/Microsoft.MachineLearningServices/workspaces/mlw-
dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e
```

The reason why you'd want to track models, could be to compare the results of models you train with different hyperparameter values.

For example, you just trained a logistic regression model with a regularization rate of 0.1. Now, train another model, but this time with a regularization rate of 0.01. Since you're also tracking the accuracy, you can compare and decide which rate results in a better performing model.

[12]:
```
from sklearn.linear_model import LogisticRegression
import numpy as np

with mlflow.start_run():
    model = LogisticRegression(C=1/0.01, solver="liblinear").fit(X_train,␣
 ↪y_train)

    y_hat = model.predict(X_test)
    acc = np.average(y_hat == y_test)

    mlflow.log_param("regularization_rate", 0.01)
    mlflow.log_metric("Accuracy", acc)
```

```
View run careful_milk_ybs76lpk at: https://eastus.api.azureml.ms/mlflow/v2.0/s
ubscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee0
1dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l
78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e/runs/8ac9c
1fc-e363-42fa-89f2-b39743eabde1
View experiment at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0
108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/p
```

roviders/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e

Another reason to track your model's results is when you're testing another estimator. All models you've trained so far used the logistic regression estimator.

Run the following cell to train a model with the decision tree classifier estimator and review whether the accuracy is higher compared to the other runs.

[13]:
```
from sklearn.tree import DecisionTreeClassifier
import numpy as np

with mlflow.start_run():
    model = DecisionTreeClassifier().fit(X_train, y_train)

    y_hat = model.predict(X_test)
    acc = np.average(y_hat == y_test)

    mlflow.log_param("estimator", "DecisionTreeClassifier")
    mlflow.log_metric("Accuracy", acc)
```

```
View run busy_roof_319349xq at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e/runs/b32e85f2-8f74-4445-9a70-97ee26219c94
View experiment at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e
```

Finally, let's try to log an artifact. An artifact can be any file. For example, you can plot the ROC curve and store the plot as an image. The image can be logged as an artifact.

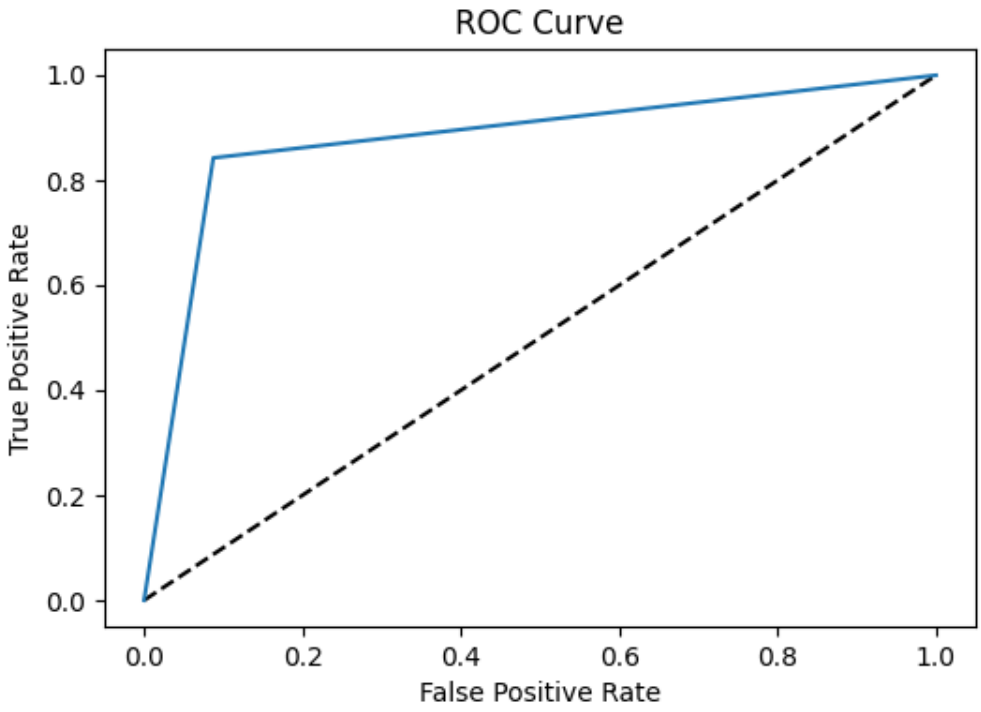
Run the following cell to log a parameter, metric, and an artifact.

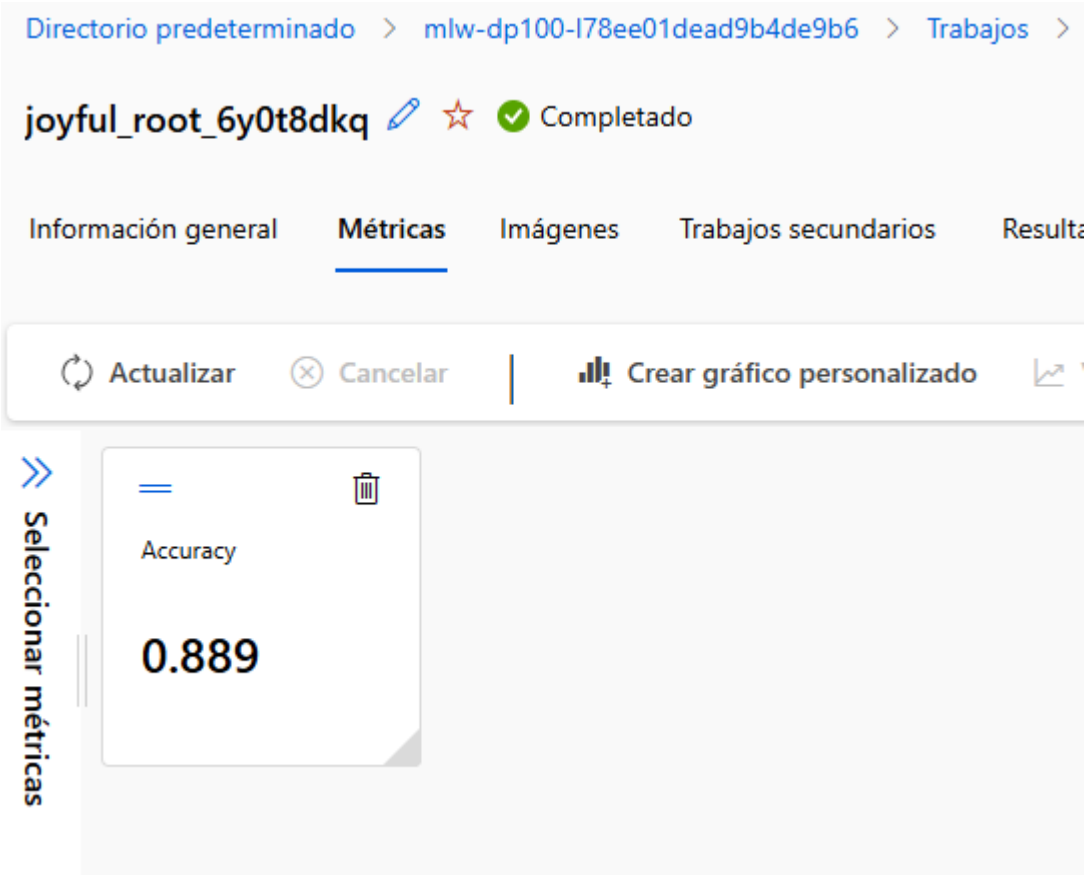
[14]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import roc_curve
import matplotlib.pyplot as plt
import numpy as np

with mlflow.start_run():
    model = DecisionTreeClassifier().fit(X_train, y_train)

    y_hat = model.predict(X_test)
    acc = np.average(y_hat == y_test)

    # plot ROC curve
    y_scores = model.predict_proba(X_test)

    fpr, tpr, thresholds = roc_curve(y_test, y_scores[:,1])
    fig = plt.figure(figsize=(6, 4))
    # Plot the diagonal 50% line
    plt.plot([0, 1], [0, 1], 'k--')
    # Plot the FPR and TPR achieved by our model
    plt.plot(fpr, tpr)
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC Curve')
    plt.savefig("ROC-Curve.png")

    mlflow.log_param("estimator", "DecisionTreeClassifier")
    mlflow.log_metric("Accuracy", acc)
    mlflow.log_artifact("ROC-Curve.png")
```

```
View run joyful_root_6y0t8dkq at: https://eastus.api.azureml.ms/mlflow/v2.0/su
bscriptions/b0108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01
dead9b4de9b6/providers/Microsoft.MachineLearningServices/workspaces/mlw-dp100-l7
8ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e/runs/99f3c4
56-d7ec-4c0f-80bf-d8ba6cc16a05
View experiment at: https://eastus.api.azureml.ms/mlflow/v2.0/subscriptions/b0
108ced-f6be-4641-917c-8e9e1cf2c8d9/resourceGroups/rg-dp100-l78ee01dead9b4de9b6/p
roviders/Microsoft.MachineLearningServices/workspaces/mlw-
dp100-l78ee01dead9b4de9b6/#/experiments/6115d7a9-818a-473a-aa63-39618382a63e
```

Review the model's results on the Jobs page of the Azure Machine Learning studio.

- You'll find the parameters under **Params** in the **Overview** tab.
- You'll find the metrics under **Metrics** in the **Overview** tab, and in the **Metrics** tab.
- You'll find the artifacts in the **Outputs + logs** tab.


ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


Directorio predeterminado
mlw-dp100-l78ee01dead9b4de9b6
Trabajos
joyful_root_6y0t8dkq
Completado
Información general
Métricas
Imágenes
Trabajos secundarios
Actualizar
Cancelar
Crear gráfico personalizado
Seleccionar métricas
Accuracy
0.889