Congratulations! You passed!

Grade received 100%

Latest Submission Grade 100%

To pass 80% or higher

Go to next item Retake the assignment in **7h 57m**

1. What are the three main components of MLflow? **1 / 1 point**

- 1. Deployment
  2. Models
  3. Packaging

- 1. Experiments
  2. Packaging
  3. Deployment

- 1. Tracking
  2. Projects
  3. Models

Correct
Correct!

2. What are three useful items that MLflow can track with the Tracking UI? **1 / 1 point**

- 1. Parameters
  2. Metrics
  3. Artifacts

- ○ 1. Performance
  2. Accuracy
  3. Parameters

- ○ 1. Parameters
  2. Performance
  3. Artifacts

✓ **Correct**
Correct!

3. Where do MLflow runs get recorded? **1 / 1 point**

- ◉ They get recorded either locally in files or remotely to a tracking server.
- ○ They get recorded remotely to a tracking server which could also be Azure ML Studio.
- ○ They get recorded locally in files in an **`mlruns`** directory wherever you ran your program.

✓ **Correct**
Correct!

4. What are three properties that are essential in MLflow projects? **1 / 1 point**

- ○ 1. Git Project
  2. Dependencies
  3. Entry points

- ○ 1. Name
  2. Parameters
  3. Entry points

- ◉ 1. Name
  2. Dependencies
  3. Entry points

✓ **Correct**

Correct!

5. What are three main ways to interact with MLflow, including runs? **1 / 1 point**

- ○ 1. Command-line interface
  2. AWS Sagemaker
  3. Azure ML Studio
- ○ 1. Scikit-learn
  2. Projects
  3. Tracking UI
- ◉ 1. Command-line interface
  2. Python API
  3. REST API

✓ **Correct**
Correct!

6. What is one benefit of using MLflow projects? **1 / 1 point**

- ○ It is a solid foundation for running distributed ML tasks
- ○ That it provides a way to deploy a run on AWS Sagemaker
- ◉ That it provides a repeatable way to install and run ML tasks

✓ **Correct**
Correct!

7. What are three common components of an MLflow project? **1 / 1 point**

- ◉ 1. `MLproject` file
  2. A dependencies or env YAML file
  3. One or more scripts to run tasks

- ○ 1. A **`pipeline.py`** file
  2. A notebook (**`ipynb`**) file
  3. One or more scripts to run tasks

- ○ 1. An **`MLflow`** file
  2. A **`pipeline.py`** file
  3. A dependencies or env YAML file

✓ **Correct**
Correct!

8. What is the right Python method to use to register an existing model with **`mlflow`** ? The choices below use an **`onnx-gpt2`** example model **1 / 1 point**

- ○
```
from mlflow import create_registered_model

create_registered_model("onnx-gpt2")
```

- ○
```
from mlflow import MlflowRegisterModel

client = MlflowClient()
MlflwoRegisterModel(client, "onnx-gpt2")
```

- ◉
```
from mlflow import MlflowClient

client = MlflowClient()
client.create_registered_model("onnx-gpt2")
```

✓ **Correct**
Correct!

9. What are the three stages supported for models in MLflow? **1 / 1 point**

- ◉ 1. Staging
  2. Production
  3. Archived

- 1. Testing
  2. Production
  3. Deleted

- 1. Testing
  2. Production
  3. Archived

Correct
Correct!

10. What happens with the MLflow Python library if you try to update a model version using **`client.update_model_version()`** if a previous version wasn't created? **1 / 1 point**

- It fails with an exception because it is a requirement to have a previous version created
- The API calls succeeds but no actual version is updated. Internally MLflow sets the version to **0**
- It creates a version **1** if no version exists. It isn't a requirements to have a previous version created.

Correct
Correct!