

- **Vendor: Microsoft**
- **Exam Code: DP-100**
- **Exam Name: Designing and Implementing a Data Science Solution on Azure**
- **New Updated Questions from [Braindump2go](#) (Updated in [May/2020](#))**

Visit Braindump2go and Download Full Version DP-100 Exam Dumps

QUESTION 152

You plan to use automated machine learning to train a regression model. You have data that has features which have missing values, and categorical features with few distinct values.

You need to configure automated machine learning to automatically impute missing values and encode categorical features as part of the training task.

Which parameter and value pair should you use in the AutoMLConfig class?

- A. featurization = 'auto'
- B. enable_voting_ensemble = True
- C. task = 'classification'
- D. exclude_nan_labels = True
- E. enable_tf = True

Answer: A

Explanation:

Featurization str or FeaturizationConfig

Values: 'auto' / 'off' / FeaturizationConfig

Indicator for whether featurization step should be done automatically or not, or whether customized featurization should be used.

Column type is automatically detected. Based on the detected column type preprocessing/featurization is done as follows:

Categorical: Target encoding, one hot encoding, drop high cardinality categories, impute missing values.

Numeric: Impute missing values, cluster distance, weight of evidence.

DateTime: Several features such as day, seconds, minutes, hours etc.

Text: Bag of words, pre-trained Word embedding, text target encoding.

Reference:

<https://docs.microsoft.com/en-us/python/api/azureml-train-automl-client/azureml.train.automl.automlconfig.automlconfig>

QUESTION 153

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You create a model to forecast weather conditions based on historical data.

You need to create a pipeline that runs a processing script to load data from a datastore and pass the processed data to a machine learning model training script.

Solution: Run the following code:

```
datastore = ws.get_default_datastore()
data_output = pd.read_csv("traindata.csv")
process_step = PythonScriptStep(script_name="process.py",
    arguments=["--data_for_train", data_output],
    outputs=[data_output], compute_target=aml_compute,
    source_directory=process_directory)
train_step = PythonScriptStep(script_name="train.py",
    arguments=["--data_for_train", data_output],
    inputs=[data_output], compute_target=aml_compute,
    source_directory=train_directory)
pipeline = Pipeline(workspace=ws, steps=[process_step, train_step])
```

Does the solution meet the goal?

- A. Yes
- B. No

Answer: A

Explanation:

The two steps are present: process_step and train_step

Note:

Data used in pipeline can be produced by one step and consumed in another step by providing a PipelineData object as an output of one step and an input of one or more subsequent steps.

PipelineData objects are also used when constructing Pipelines to describe step dependencies. To specify that a step requires the output of another step as input, use a PipelineData object in the constructor of both steps.

For example, the pipeline train step depends on the process_step_output output of the pipeline process step:

from azureml.pipeline.core import Pipeline, PipelineData from azureml.pipeline.steps import PythonScriptStep

```
datastore = ws.get_default_datastore()
```

```
process_step_output = PipelineData("processed_data", datastore=datastore) process_step =
```

```
PythonScriptStep(script_name="process.py", arguments=["--data_for_train", process_step_output],
```

```
outputs=[process_step_output],
```

```
compute_target=aml_compute,
```

```
source_directory=process_directory)
```

```
train_step = PythonScriptStep(script_name="train.py",
```

```
arguments=["--data_for_train", process_step_output],
```

```
inputs=[process_step_output],
```

```
compute_target=aml_compute,
```

```
source_directory=train_directory)
```

```
pipeline = Pipeline(workspace=ws, steps=[process_step, train_step])
```

Reference:

<https://docs.microsoft.com/en-us/python/api/azureml-pipeline-core/azureml.pipeline.core.pipelinedata?view=azure-ml-py>

QUESTION 154

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You create a model to forecast weather conditions based on historical data.

You need to create a pipeline that runs a processing script to load data from a datastore and pass the processed data to a machine learning model training script.

Solution: Run the following code:

```
datastore = ws.get_default_datastore()
data_output = PipelineData("processed_data", datastore=datastore)
process_step = PythonScriptStep(script_name="process.py",
    arguments=["--data_for_train", data_output],
    outputs=[data_output], compute_target=aml_compute,
    source_directory=process_directory)
pipeline = Pipeline(workspace=ws, steps=[process_step])
```

Does the solution meet the goal?

- A. Yes
- B. No

Answer: B

Explanation:

train_step is missing.

Reference:

<https://docs.microsoft.com/en-us/python/api/azureml-pipeline-core/azureml.pipeline.core.pipelinedata?view=azure-ml-py>

QUESTION 155

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You create a model to forecast weather conditions based on historical data.

You need to create a pipeline that runs a processing script to load data from a datastore and pass the processed data to a machine learning model training script.

Solution: Run the following code:

```
datastore = ws.get_default_datastore()
data_input = PipelineData("raw_data", datastore=rawdatastore)
data_output = PipelineData("processed_data", datastore=datastore)
process_step = PythonScriptStep(script_name="process.py",
    arguments=["--data_for_train", data_input],
    outputs=[data_output], compute_target=aml_compute,
    source_directory=process_directory)
train_step = PythonScriptStep(script_name="train.py",
    arguments=["--data_for_train", data_input], inputs=[data_output],
    compute_target=aml_compute, source_directory=train_directory)
pipeline = Pipeline(workspace=ws, steps=[process_step, train_step])
```

Does the solution meet the goal?

- A. Yes
- B. No

Answer: B

Explanation:

Note: Data used in pipeline can be produced by one step and consumed in another step by providing a PipelineData object as an output of one step and an input of one or more subsequent steps.

Compare with this example, the pipeline train step depends on the process_step_output output of the pipeline process step:

```
from azureml.pipeline.core import Pipeline, PipelineData
from azureml.pipeline.steps import PythonScriptStep
datastore = ws.get_default_datastore()
process_step_output = PipelineData("processed_data", datastore=datastore)
process_step = PythonScriptStep(script_name="process.py",
                                arguments=["--data_for_train", process_step_output],
                                outputs=[process_step_output],
                                compute_target=aml_compute,
                                source_directory=process_directory)
train_step = PythonScriptStep(script_name="train.py",
                                arguments=["--data_for_train", process_step_output],
                                inputs=[process_step_output],
                                compute_target=aml_compute,
                                source_directory=train_directory)
pipeline = Pipeline(workspace=ws, steps=[process_step, train_step])
```

QUESTION 156

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a Python script named train.py in a local folder named scripts. The script trains a regression model by using scikit-learn. The script includes code to load a training data file which is also located in the scripts folder.

You must run the script as an Azure ML experiment on a compute cluster named aml-compute.

You need to configure the run to ensure that the environment includes the required packages for model training. You have instantiated a variable named aml-compute that references the target compute cluster.

Solution: Run the following code:

```
from azureml.train.sklearn import SKLearn
sk_est = SKLearn(source_directory='./scripts',
                  compute_target=aml_compute,
                  entry_script='train.py')
```

Does the solution meet the goal?

- A. Yes
- B. No

Answer: A

Explanation:

The scikit-learn estimator provides a simple way of launching a scikit-learn training job on a compute target. It is implemented through the SKLearn class, which can be used to support single-node CPU training.

Example:

```
from azureml.train.sklearn import SKLearn
}
estimator = SKLearn(source_directory=project_folder,
                    compute_target=compute_target,
                    entry_script='train_iris.py'
                    )
```

Reference:

<https://docs.microsoft.com/en-us/azure/machine-learning/how-to-train-scikit-learn>

QUESTION 157

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one

[DP-100 Exam Dumps](#) [DP-100 Exam Questions](#) [DP-100 PDF Dumps](#) [DP-100 VCE Dumps](#)

<https://www.braindump2go.com/dp-100.html>

correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a Python script named train.py in a local folder named scripts. The script trains a regression model by using scikit-learn. The script includes code to load a training data file which is also located in the scripts folder.

You must run the script as an Azure ML experiment on a compute cluster named aml-compute.

You need to configure the run to ensure that the environment includes the required packages for model training. You have instantiated a variable named aml-compute that references the target compute cluster.

Solution: Run the following code:

```
from azureml.train.dnn import TensorFlow
sk_est = TensorFlow(source_directory='./scripts',
    compute_target=aml-compute,
    entry_script='train.py')
```

Does the solution meet the goal?

- A. Yes
- B. No

Answer: B

Explanation:

The scikit-learn estimator provides a simple way of launching a scikit-learn training job on a compute target. It is implemented through the SKLearn class, which can be used to support single-node CPU training.

Example:

```
from azureml.train.sklearn import SKLearn
}
estimator = SKLearn(source_directory=project_folder,
    compute_target=compute_target,
    entry_script='train_iris.py'
)
```

Reference:

<https://docs.microsoft.com/en-us/azure/machine-learning/how-to-train-scikit-learn>

QUESTION 158

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a Python script named train.py in a local folder named scripts. The script trains a regression model by using scikit-learn. The script includes code to load a training data file which is also located in the scripts folder.

You must run the script as an Azure ML experiment on a compute cluster named aml-compute.

You need to configure the run to ensure that the environment includes the required packages for model training. You have instantiated a variable named aml-compute that references the target compute cluster.

Solution: Run the following code:

```
from azureml.train.estimator import Estimator
sk_est = Estimator(source_directory='./scripts',
    compute_target=aml-compute,
    entry_script='train.py',
    conda_packages=['scikit-learn'])
```

Does the solution meet the goal?

- A. Yes

B. No

Answer: B

Explanation:

The scikit-learn estimator provides a simple way of launching a scikit-learn training job on a compute target. It is implemented through the SKLearn class, which can be used to support single-node CPU training.

Example:

```
from azureml.train.sklearn import SKLearn
}
estimator = SKLearn(source_directory=project_folder,
                    compute_target=compute_target,
                    entry_script='train_iris.py'
                    )
```

Reference:

<https://docs.microsoft.com/en-us/azure/machine-learning/how-to-train-scikit-learn>

QUESTION 159

You create a multi-class image classification deep learning model that uses a set of labeled images. You create a script file named train.py that uses the PyTorch 1.3 framework to train the model.

You must run the script by using an estimator. The code must not require any additional Python libraries to be installed in the environment for the estimator. The time required for model training must be minimized.

You need to define the estimator that will be used to run the script.

Which estimator type should you use?

- A. TensorFlow
- B. PyTorch
- C. SKLearn
- D. Estimator

Answer: B

Explanation:

For PyTorch, TensorFlow and Chainer tasks, Azure Machine Learning provides respective PyTorch, TensorFlow, and Chainer estimators to simplify using these frameworks.

Reference:

<https://docs.microsoft.com/en-us/azure/machine-learning/how-to-train-ml-models>

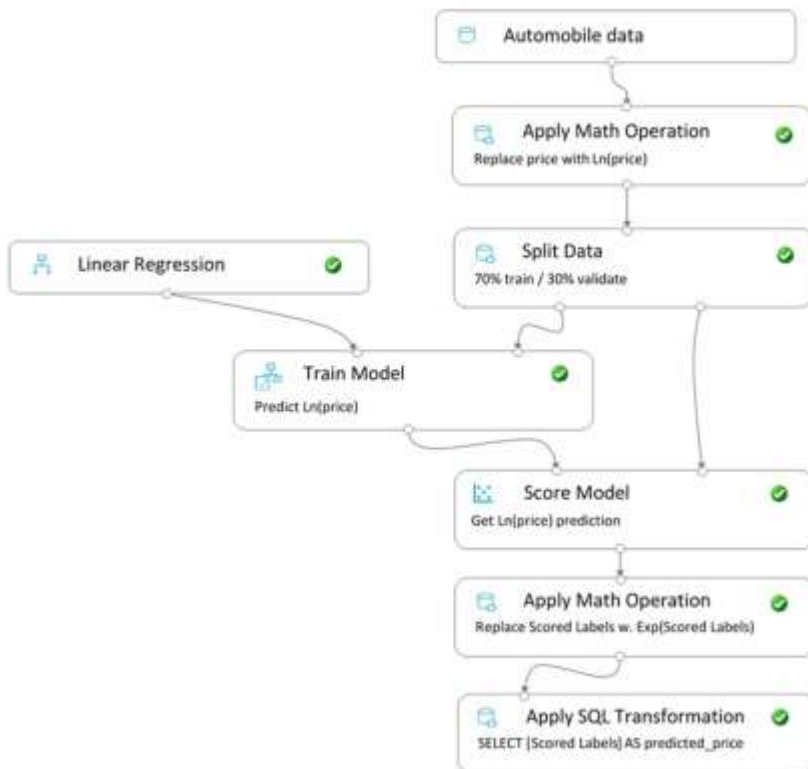
QUESTION 160

You create a pipeline in designer to train a model that predicts automobile prices.

Because of non-linear relationships in the data, the pipeline calculates the natural log (Ln) of the prices in the training data, trains a model to predict this natural log of price value, and then calculates the exponential of the scored label to get the predicted price.

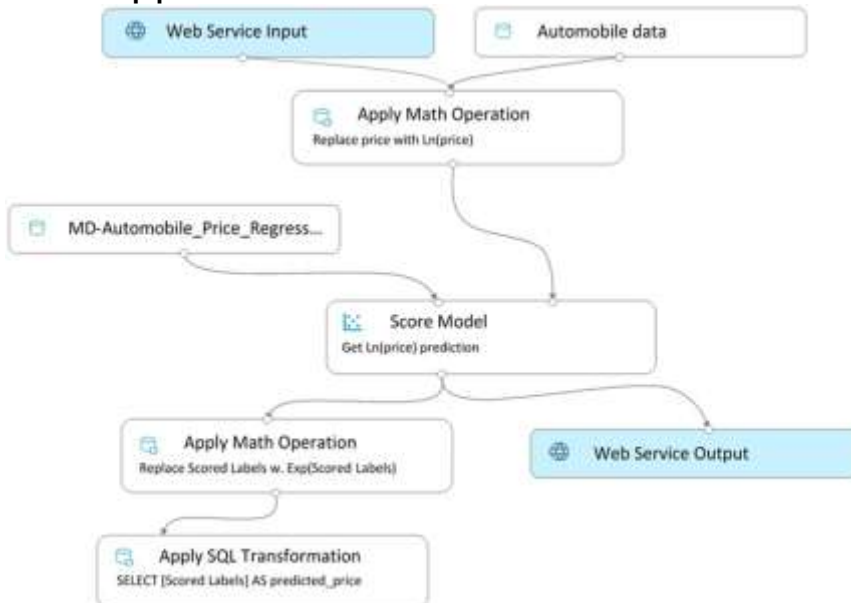
The training pipeline is shown in the exhibit. (Click the **Training pipeline** tab.)

Training pipeline



You create a real-time inference pipeline from the training pipeline, as shown in the exhibit. (Click the **Real-time pipeline** tab.)

Real-time pipeline



You need to modify the inference pipeline to ensure that the web service returns the exponential of the scored label as the predicted automobile price and that client applications are not required to include a price value in the input values. Which three modifications must you make to the inference pipeline? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

- A. Connect the output of the Apply SQL Transformation to the Web Service Output module.
- B. Replace the Web Service Input module with a data input that does not include the price column.
- C. Add a Select Columns module before the Score Model module to select all columns other than price.
- D. Replace the training dataset module with a data input that does not include the price column.

- E. Remove the Apply Math Operation module that replaces price with its natural log from the data flow.
- F. Remove the Apply SQL Transformation module from the data flow.

Answer: ACE

QUESTION 161

You train a model and register it in your Azure Machine Learning workspace. You are ready to deploy the model as a real-time web service.

You deploy the model to an Azure Kubernetes Service (AKS) inference cluster, but the deployment fails because an error occurs when the service runs the entry script that is associated with the model deployment.

You need to debug the error by iteratively modifying the code and reloading the service, without requiring a re-deployment of the service for each code update.

What should you do?

- A. Modify the AKS service deployment configuration to enable application insights and re-deploy to AKS.
- B. Create an Azure Container Instances (ACI) web service deployment configuration and deploy the model on ACI.
- C. Add a breakpoint to the first line of the entry script and redeploy the service to AKS.
- D. Create a local web service deployment configuration and deploy the model to a local Docker container.
- E. Register a new version of the model and update the entry script to load the new version of the model from its registered path.

Answer: B

Explanation:

How to work around or solve common Docker deployment errors with Azure Container Instances (ACI) and Azure Kubernetes Service (AKS) using Azure Machine Learning.

The recommended and the most up to date approach for model deployment is via the `Model.deploy()` API using an `Environment` object as an input parameter. In this case our service will create a base docker image for you during deployment stage and mount the required models all in one call. The basic deployment tasks are:

1. Register the model in the workspace model registry.

2. Define Inference Configuration:

F. Create an `Environment` object based on the dependencies you specify in the environment yaml file or use one of our procured environments.

G. Create an inference configuration (`InferenceConfig` object) based on the environment and the scoring script.

3. Deploy the model to Azure Container Instance (ACI) service or to Azure Kubernetes Service (AKS).

QUESTION 162

You are creating a classification model for a banking company to identify possible instances of credit card fraud. You plan to create the model in Azure Machine Learning by using automated machine learning.

The training dataset that you are using is highly unbalanced.

You need to evaluate the classification model.

Which primary metric should you use?

- A. `normalized_mean_absolute_error`
- B. `AUC_weighted`
- C. `accuracy`
- D. `normalized_root_mean_squared_error`
- E. `spearman_correlation`

Answer: B

Explanation:

`AUC_weighted` is a Classification metric.

Note: AUC is the Area under the Receiver Operating Characteristic Curve. Weighted is the arithmetic mean of the score for each class, weighted by the number of true instances in each class.

Incorrect Answers:

A: normalized_mean_absolute_error is a regression metric, not a classification metric.

C: When comparing approaches to imbalanced classification problems, consider using metrics beyond accuracy such as recall, precision, and AUROC. It may be that switching the metric you optimize for during parameter selection or model selection is enough to provide desirable performance detecting the minority class.

D: normalized_root_mean_squared_error is a regression metric, not a classification metric.

Reference:

<https://docs.microsoft.com/en-us/azure/machine-learning/how-to-understand-automated-ml>