Question 1

Domain: Manage Azure resources for machine learning

For your ML experiments, you need to process CSV data files. Size of your files is about 10GB each. Your training script loads the ingested data to a pandas dataframe object. During the runs, you get an “Out of memory” error. You decide to convert the files to Parquet format and process it partially, i.e. loading only the columns relevant from the modelling point of view.

Does it solve the problem?

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because the data loaded from a CSV file can expand significantly when loaded into a dataframe in memory. Converting it to the columnar Parquet format is a viable solution because it enables loading selected columns which are necessary for the training process.
- Option B is incorrect because using the columnar Parquet format instead of CSV can be used to optimize memory consumption, therefore it is a good solution.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-create-register-datasets
- https://docs.microsoft.com/en-us/azure/machine-learning/concept-optimize-data-processing

Question 2

Domain: Implement responsible machine learning

After successfully training your ML model and after selecting the best run, you are about to deploy it as a web service to the production environment. Because you anticipate a massive amount of requests to be handled by the service, you choose AKS as a compute target. You want to use the following script to deploy your model:

```
# deploy model
from azureml.core.model import Model

service = Model.deploy(workspace=ws,
                      name = 'my-inference-service',
                      models = [classification_model],
                      inference_config = my_inference_config,
                      deployment_config = my_deploy_config,
                      deployment_target = my_production_cluster)
service.wait_for_deployment(show_output = True)
```

After running the deployment script, you receive an error. After a short investigation you find that an important setting is missing from the inference_config definition:

```
# inference config
from azureml.core.model import InferenceConfig

inference_config = InferenceConfig(runtime= "python",
                                   source_directory = 'my_files',
                                   <insert code here>,
                                   conda_file="environment.yml")
```

You decide to add <entry_script="my_scoring.py">

Does this solve the problem??

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because the InferenceConfig object is used to combine two important things: the entry script and the environment definition. The entry_script defines the path to the file that contains the ML code to execute, therefore it must be set.
- Option B is incorrect because the entry_script parameter is actually missing from the InferenceConfig definition. Adding it does solve the problem.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-deploy-and-where?tabs=python

Question 3

Domain: Manage Azure resources for machine learning

You are tasked to set up a ML environment for running experiments. While configuring the compute environment for your experiments, your priority is to provision the necessary capacity but keeping costs as low as possible.

Which is not the way of optimizing costs?

- A. Develop code in local, low-cost environment

- B. Use Azure Kubernetes Service
- C. Use managed computes that start automatically on-demand and stop when not needed

- D. Set automatic scaling for computes, based on the workload

Explanation:

Answer: B

- Option A is incorrect because using your own local environment while developing code generates no extra cost, not even for long execution times.
- Option B is CORRECT because AKS is recommended for high-scale production environments. Not the best way to keep costs low.
- Option C is incorrect because in a cloud environment, utilizing the pay-as-you-go model, automatically launching and stopping computes is one of the best ways of preventing costs from “exploding”.
- Option D is incorrect because in the cloud environment, you can set up compute clusters which provide the necessary compute power while scaling up and down automatically, according to the actual demand.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/concept-compute-target

Question 4

Domain: Implement responsible machine learning

You have a real-time inference web service which you have just deployed to Azure Kubernetes Service. During its run, some unexpected errors occur. You need to troubleshoot it quickly and cost-effectively.

Which is the quickest and cheapest option you should use?

- A. Deploy it as a local web service and debug locally
- B. Deploy it to ACI
- C. Use a compute instance as deployment target for debugging
- D. Deploy it to AKS and set the maximum number of replicas to one; debug it in the production environment

Explanation:

Answer: A

- Option A is CORRECT because using a local web service makes it easier to troubleshoot and debug problems. If you have problems with your model deployed to ACI or AKS, try deploying it as a local web service. You can then troubleshoot runtime issues by making changes to the scoring file that is referenced in the inference configuration, and reloading the service without redeploying it. This can be done only with local services.
- Option B is incorrect because Azure Container Instances is designed to be used for low-scale production deployments. It is highly recommended to debug locally before deploying the web service.
- Option C is incorrect because while an Azure compute instance might be a good target platform for debugging, using a local containerized environment is even better in this case.
- Option D is incorrect because Azure Kubernetes Service is to be used for high-scale production deployments. Being a powerful runtime environment, it is far too expensive for testing and debugging.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-troubleshoot-deployment
- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-troubleshoot-deployment#debug-locally

Question 5

Domain: Run experiments and train models

You are developing a ML pipeline using Python SDK, and you have to separate your data for training the model as well as for testing the trained model. You got a code snippet from your teammate, which is a great help, if it works. You have to check if it does the job.

By its description, the script loads data from the default datastore and separates the 70% of the observations for training and the rest of them for testing, by using the scikit-learn package, in a reproducible way.

```
from sklearn.model_selection import train_test_split

# Get the experiment run context
run = Run.get_context()

# load data
print("Loading Data...")
diabetes_data =
run.input_datasets['diabetes_train'].to_pandas_dataframe()

# Separate features and labels
X, y = diabetes_data[['Pregnancies','PlasmaGlucose',
       'DiastolicBloodPressure','BMI','Age']].values,
       diabetes_data['Diabetic'].values

# Split data into training set and test set
X_train, X_test, y_train, y_test =
       train_test_split(X, y, test_size=0.30, random_state=0)
```

After reviewing the code, do you think it does its job as described?

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because the code is correct. It does its job exactly as it is described.
- Option B is incorrect because the code is correct.

Reference:

- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html?highlight=split#sklearn.model_selection.train_test_split

Question 6

Domain: Implement responsible machine learning

After successfully training your ML model and after selecting the best run, you are about to deploy it as a web service to the production environment. Because you anticipate a massive amount of requests to be handled by the service, you choose AKS as a compute target. You want to use the following script to deploy your model:

```
# deploy model
from azureml.core.model import Model

service = Model.deploy(workspace=ws,
                          name = 'my-inference-service',
                          models = [classification_model],
                          inference_config = my_inference_config,
                          deployment_config = my_deploy_config,
                          deployment_target = my_production_cluster)
service.wait_for_deployment(show_output = True)
```

After running the deployment script, you receive an error. After a short investigation you find that an important setting is missing from the inference_config definition:

```
# inference config
from azureml.core.model import InferenceConfig

inference_config = InferenceConfig(runtime= "python",
                                   source_directory = 'my_files',
                                   <insert code here>,
                                   conda_file="environment.yml")
```

You decide to add <cluster_name = ‘aks-cluster’>

Does this solve the problem??

- A. Yes
- B. No

Explanation:

Answer: B

- Option A is incorrect because the InferenceConfig object is used to combine two important things: the entry script and the environment definition. The entry_script defines the path to the file that contains the ML code to execute, therefore it is missing and it must be set: entry_script="my_scoring.py".
- Option B is CORRECT because the cluster_name parameter is actually important for the deployment, but it is part of the ComputeTarget configuration (which is, in your case, the my_production_cluster), i.e. set elsewhere in your code.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-deploy-and-where?tabs=python

Question 7

Domain: Deploy and operationalize machine learning solutions

While setting up your machine learning experiments, you need to ensure that the trained models will be appropriately scored with validation data. Azure ML SDK provides several methods to specify in your scripts how to determine the data to be used for validation.

Which is not a valid set of parameters for specifying validation method? Select two!

- A. Primary metric; training data
- B. Primary metric; training data; validation data

- C. Primary metric; validation data; number of cross-validations
- D. Primary metric; training data; validation set size

- E. Primary metric; training data; validation data; validation set size

Explanation:

Answers: C and E

- Option A is incorrect because the primary metric must be given in all cases. When only training data is provided, the default splitting rules will be applied.
- Option B is incorrect because the primary metric is always required. When both training and validation data are provided, these will be used, since no automatic splitting is needed.
- Option C is CORRECT because primary metric and training data are always required. Validation data either can be provided explicitly, or can be generated automatically (by default or explicit splitting rules).
- Option D is incorrect because when only training data is provided, with validation set size set manually, this value will be used to split data into training and validation subsets.
- Option E is CORRECT because it is a kind of redundancy because either validation data or training data with validation set size can be set. Setting both validation data and validation set size is wrong.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-configure-cross-validation-data-splits#provide-validation-data
- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-configure-cross-validation-data-splits#set-the-number-of-cross-validations

Question 8

Domain: Deploy and operationalize machine learning solutions

You are building an ML model, for which you want to find the optimal parameter setting which results in the best performing model. You decide to use the hyperparameter-tuning feature of Azure ML, i.e. use Hyperdrive in your experiments. Using Hyperdrive requires some specific conditions your script must fulfil.

Which components/settings are specific only for Hyperdrive experiments?

- A. Define ScriptConfig; create ScriptRunConfig
- B. Add script argument for hyperparameters; create an Estimator
- C. Add script argument for hyperparameters; Log primary metricright
- D. Define training dataset; validation dataset

Explanation:

Answer: C

- Option A is incorrect because ScriptConfig and ScriptRunConfig are common configuration objects used for any ML experiment. They are not specific for Hyperdrive experiments.
- Option B is incorrect because adding a script argument for hyperparameters to be adjusted is specific for Hyperdrive indeed, estimators are commonly used in any experiment, as “wrappers” for ScriptConfig and ScriptRunConfig.
- Option C is CORRECT because if you want to tune model parameters using Hyperdrive, you must include a script argument for each parameter to be adjusted, as well as the primary performance metric (e.g. Accuracy) must be logged, so that Hyperdrive can evaluate the runs and it can select the best performer combination.
- Option D is incorrect because training and test/validation datasets are fundamental components of any ML experiment.

Reference:

- https://github.com/MicrosoftLearning/DP100/blob/master/08A%20-%20Tuning%20Hyperparameters.ipynb
- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-tune-hyperparameters

Question 9

Domain: Deploy and operationalize machine learning solutions

You are running experiments in Azure ML and you want to configure the model's hyperparameters. You need to define the sampling space for two parameters and you want Azure ML to use a sampling method which picks samples based on the result of previous runs, in order to improve the performance in the primary metric.

```
# define parameter space
from azureml.train.hyperdrive import BayesianParameterSampling, choice,
uniform
my_parameter_space = {
        'batch_size': choice(8, 16, 32, 64),
        "keep_probability": uniform(0.05, 0.1)
    }
param_sampling = BayesianParameterSampling(my_parameter_space)
...
```

Does the script above fulfills your requirement?

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because the Bayesian parameter sampling is the right choice, and this sampling method supports choice, uniform, and quniform distributions. Therefore, the code is correct.
- Option B is incorrect because the code defines the parameter search space for two parameters, batch_size and keep_probability, with the Bayesian sampling method. Bayesian sampling improves sampling based on the result of previous runs, and it can be used with either choice or uniform methods.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-tune-hyperparameters

Question 10

Domain: Run experiments and train models

As part of a ML team, your task is to build a workflow to chain several steps of the ML process in order to automate the series of tasks and to allow automation and reuse. You are going to use the Pipeline feature of the Azure ML stack. You have planned the main steps in the pipeline, and now you are about to optimize how to organize your files around your pipeline.

Which two of the following options are true and should be used as best practice in pipeline design?

- A. Store scripts and dependencies of the whole pipeline in a single source directory in order to take the advantage of data reuse

- B. Store scripts and dependencies for each step in separate source directories in order to take advantage of data reuseright
- C. Store scripts and dependencies of the whole pipeline in a single source directory in order to reduce the size of the snapshots for given steps

- D. Store scripts and dependencies for each step in separate source directories in order to reduce the size of snapshot for given stepsright
- E. Force output regeneration for steps in a run by setting the allow_reuse to False

Explanation:

Answers: B and D

- Option A is incorrect because if all the scripts belonging to the pipeline are kept in a single directory, every time the content of the directory changes, it will force all steps to rerun, i.e. no data reuse.
- Option B is CORRECT because Steps in a pipeline can be configured to reuse results from their previous runs if the step's scripts, dependencies, inputs etc. are unchanged. Keeping the files for each step in separate folders ensures that if only files of any step changes, only the output data of the given step is regenerated, while all the others' remain unchanged and reusable.
- Option C is incorrect because if the scripts and dependencies of the whole pipeline are kept in a single directory, snapshotting of the steps takes an unnecessary high amount of time and storage.

- Option D is CORRECT because keeping the scripts and dependencies of each step in separate directories helps minimize the resources necessary for snapshotting the steps.
- Option E is incorrect because the default setting for allow_reuse is True which means that the results of the previous step run is reused until the content of the source_directory is unchanged. In order to save time and resources, changing this default behavior is only recommended if there is a special reason why the results must be re-generated.

Reference:

- https://docs.microsoft.com/en-us/python/api/azureml-pipeline-core/azureml.pipeline.core.builder.pipelinestep?view=azure-ml-py&preserve-view=true#remarks

Question 11

Domain: Implement responsible machine learning

You have deployed your real-time inference web service on the Azure Container Instances (ACI) environment. You need to ensure that only consumer services having the appropriate authentication credentials can have access to it.

Which authentication method can you use?

- A. User/Password
- B. Key
- C. SAS

- D. Token

Explanation:

Answer: B

- Option A is incorrect because Azure ML provides two ways to control access to web services: Kea and Token. User/Password is not applicable here.
- Option B is CORRECT because using a Key is the only way to authenticate consumers of a real-time inference service on ACI. The Key has to be included in the Authorization header of the request.
- Option C is incorrect because the SQL engine
- Option D is incorrect because while using time limited tokens for authentication can be an option for inference models, it is not supported for ACI; it is only available for AKS.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-consume-web-service?tabs=python#authentication-for-services

Question 12

Domain: Deploy and operationalize machine learning solutions

You are working for a company which is operating a webshop. All the transactions flowing through the site are directed to a real-time inferencing web service to identify potentially risky transactions. One of the transactions is classified by the model as “suspicious” and, before taking actions, you are tasked to investigate which features made the model “think” so.

You decide to use Mimic Explainer to help you understand why this specific transaction has been classified as “suspicious”.

Does it serve your purpose?

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because Azure offers a selection of model explainers: Tabular, Mimic and Permutation Feature Importance. All of them can be used for explaining global importance of features, but only two of them (Tabular, Mimic) are applicable if you need to interpret local importance. Mimic is a good choice for your task.
- Option B is incorrect because either Mimic or Tabular explainer can be used for interpreting the local importance of features, i.e. Mimic is a good choice.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-machine-learning-interpretability

Question 13

Domain: Run experiments and train models

For running your ML experiments, you want to create a separate Python script for configuring and running the experiment, and store it in a folder for future use. While writing the script, there is a list of key steps you have to include in a specific order.

Which of the following options reflects the right order of the required steps within the script?

- A. Workspace() -> Compute target() -> RunConfiguration() -> Experiment.submit()
- B. Run.get_context() -> Experiment() -> ScriptRunConfiguration() -> Workspace()

- C. Workspace() -> Run.get_context() -> ScriptRunConfig() -> Experiment.submit()
- D. Workspace() -> ScriptRunConfig() -> Run.get_context() -> Experiment()

Explanation:

Answer: C

- Option A is incorrect because defining compute targets is not part of the experiment script; instead of RunConfiguration() the Run.get_context() has to be used.
- Option B is incorrect because connecting to a Machine Learning workspace must be the very first step.
- Option C is CORRECT because the very first step is connection to an ML workspace, then the run context for running the script has to be retrieved, then a ScriptRunConfig is needed to define the script to be run. Finally, you have to submit the experiment by the submit() method.
- Option D is incorrect because the Experiment.submit() method must be used as the last step, in order to run the experiment.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/tutorial-1st-experiment-sdk-train

Question 14

Domain: Run experiments and train models

For your machine learning experiments, you are going to use the Scikit-Learn framework. You want to keep your Python code defining the run configuration as simple and compact as possible.

Which is the best way to achieve this goal?

- A. Use CondaDependencies.create(conda_packages=['scikit-learn']...) to define the environment and use it as the environment_definition parameter of an Estimator

- B. Import the SKLearn package and use the SKLearn pre-configured estimator to define the run configuration
- C. Import the Estimator package and use Estimator with parameter conda_packages=['scikit-learn']

- D. You don't need to set anything special because the azure ML environments are pre-configured for the Scikit-Learn framework

Explanation:

Answer: B

- Option A is incorrect because while this solution can be used to set the run configuration, in the case of Scikit-Learn framework, using the pre-configured SKLearn estimator is the best solution.
- Option B is CORRECT because the simplest way to define the run configuration for the learning script built on a given ML framework (like Scikit-Learn) is to use the framework-specific estimators
- Option C is incorrect because while this can be used to set the run configuration, in the case of Scikit-Learn framework, using the pre-configured SKLearn estimator is the best solution.
- Option D is incorrect because the specific ML packages (like ScikitLearn, PyTorch etc.) are not contained in the base configuration. If you need Scikit-Learn, you have to add it to your run configuration (either via ScriptRunConfig or via an estimator).

Reference:

- https://docs.microsoft.com/en-us/python/api/azureml-train-core/azureml.train.estimator?view=azure-ml-py&preserve-view=true

Question 15

Domain: Manage Azure resources for machine learning

For your ML experiments, you need to process CSV data files. Size of your files is about 2GB each. Your training script loads the ingested data to a pandas dataframe object. During the first run, you get an “Out of memory” error. You decide to double the size of the compute’s memory (which is 16GB currently).

Does it solve the problem?

- A. Yes
- B. No

Explanation:

Answer: A

- Option A is CORRECT because the data loaded from a CSV file can expand even as much as 10 times when loaded into a dataframe in memory. It is recommended to set the size of the memory at least two times the size of the input data.
- Option B is incorrect because a typical reason for “Out of memory” errors during this process is that data loaded from a CSV file expands significantly when loaded to a dataframe. Extending the compute memory is one possible solution.

Reference:

- https://docs.microsoft.com/en-us/azure/machine-learning/how-to-create-register-datasets