Download Free Databricks-Machine-Learning-Associate Exam PDF | PrepBolt

Don't miss out! Download the latest free Databricks Certified Machine Learning Associate Exam PDF questions. Access real Databricks-Machine-Learning-Associate dumps with verified answers and boost your chances to pass your certification on the first try with PrepBolt Databricks-Machine-Learning-Associate exam pdf questions and answers.

Thank you for Downloading Databricks-Machine-Learning-Associate exam PDF Demo

https://prepbolt.com/Databricks-Machine-Learning-Associate.html

Question 1

Question Type: MultipleChoice

A data scientist has produced three new models for a single machine learning problem. In the past, the solution used just one model. All four models have nearly the same prediction latency, but a machine learning engineer suggests that the new solution will be less time efficient during inference.

In which situation will the machine learning engineer be correct?

Options:

A- When the new solution requires if-else logic determining which model to use to compute each prediction
B- When the new solution's models have an average latency that is larger than the size of the original model
C- When the new solution requires the use of fewer feature variables than the original model
D- When the new solution requires that each model computes a prediction for every record
E- When the new solution's models have an average size that is larger than the size of the original model

Answer:

D

Explanation:

If the new solution requires that each of the three models computes a prediction for every record, the time efficiency during inference will be reduced. This is because the inference process now involves running multiple models instead of a single model, thereby increasing the overall computation time for each record.

In scenarios where inference must be done by multiple models for each record, the latency accumulates, making the process less time efficient compared to using a single model.

Model Ensemble Techniques

Question 2

Question Type: MultipleChoice

A data scientist wants to parallelize the training of trees in a gradient boosted tree to speed up the training process. A colleague suggests that parallelizing a boosted tree algorithm can be difficult.

Which of the following describes why?

Options:

A- Gradient boosting is not a linear algebra-based algorithm which is required for parallelization
B- Gradient boosting requires access to all data at once which cannot happen during parallelization.
C- Gradient boosting calculates gradients in evaluation metrics using all cores which prevents parallelization.
D- Gradient boosting is an iterative algorithm that requires information from the previous iteration to perform the next step.

Answer:

D

Explanation:

Gradient boosting is fundamentally an iterative algorithm where each new tree is built based on the errors of the previous ones. This sequential dependency makes it difficult to parallelize the training of trees in gradient boosting, as each step relies on the results from the preceding step. Parallelization in this context would undermine the core methodology of the algorithm, which depends on sequentially improving the model's performance with each iteration. Reference:

Machine Learning Algorithms (Challenges with Parallelizing Gradient Boosting).

Gradient boosting is an ensemble learning technique that builds models in a sequential manner. Each new model corrects the errors made by the previous ones. This sequential dependency means that each iteration requires the results of the previous iteration to make corrections. Here is a step-by-step explanation of why this makes parallelization challenging:

Sequential Nature: Gradient boosting builds one tree at a time. Each tree is trained to correct the residual errors of the previous trees. This requires the model to complete one iteration before starting the next.

Dependence on Previous Iterations: The gradient calculation at each step depends on the predictions made by the previous models. Therefore, the model must wait until the previous tree has been fully trained and evaluated before starting to train the next tree.

Difficulty in Parallelization: Because of this dependency, it is challenging to parallelize the training process. Unlike algorithms that process data independently in each step (e.g., random forests), gradient boosting cannot easily distribute the work across multiple processors or cores for simultaneous execution.

This iterative and dependent nature of the gradient boosting process makes it difficult to parallelize effectively.

Reference

Gradient Boosting Machine Learning Algorithm

Understanding Gradient Boosting Machines

Question 3

Question Type: MultipleChoice

Which of the Spark operations can be used to randomly split a Spark DataFrame into a training DataFrame and a test DataFrame for downstream use?

Options:

A- TrainValidationSplit
B- DataFrame.where
C- CrossValidator
D- TrainValidationSplitModel
E- DataFrame.randomSplit

Answer:

E

Explanation:

The correct method to randomly split a Spark DataFrame into training and test sets is by using the randomSplit method. This method allows you to specify the proportions for the split as a list of weights and returns multiple DataFrames according to those weights. This is directly intended for splitting DataFrames randomly and is the appropriate choice for preparing data for training and testing in machine learning workflows. Reference:

Apache Spark DataFrame API documentation (DataFrame Operations: randomSplit).

Question 4

Question Type: MultipleChoice

A data scientist is attempting to tune a logistic regression model logistic using scikit-learn. They want to specify a search space for two hyperparameters and let the tuning process randomly select values for each evaluation.

They attempt to run the following code block, but it does not accomplish the desired task:

```
distributions = dict(C=uniform(loc=0, scale=4), penalty=['l2', 'l1'])
clf = GridSearchCV(logistic, distributions, random_state=0)
search = clf.fit(feature_data, target_data)
```

Which of the following changes can the data scientist make to accomplish the task?

Options:

A- Replace the GridSearchCV operation with RandomizedSearchCV
B- Replace the GridSearchCV operation with cross_validate
C- Replace the GridSearchCV operation with ParameterGrid
D- Replace the random_state=0 argument with random_state=1
E- Replace the penalty= ['12', '11'] argument with penalty=uniform ('12', '11')

Answer:

A

Explanation:

The user wants to specify a search space for hyperparameters and let the tuning process randomly select values. GridSearchCV systematically tries every combination of the provided hyperparameter values, which can be computationally expensive and time-consuming. RandomizedSearchCV, on the other hand, samples hyperparameters from a distribution for a fixed number of iterations. This approach is usually faster and still can find very good parameters, especially when the search space is large or includes distributions.

Reference

Scikit-Learn documentation on hyperparameter tuning:
https://scikit-learn.org/stable/modules/grid_search.html#randomized-parameter-optimization

Question 5

Question Type: MultipleChoice

A data scientist is developing a single-node machine learning model. They have a large number of model configurations to test as a part of their experiment. As a result, the model tuning process takes too long to complete. Which of the following approaches can be used to speed up the model tuning process?

Options:

A- Implement MLflow Experiment Tracking
B- Scale up with Spark ML
C- Enable autoscaling clusters
D- Parallelize with Hyperopt

Answer:

D

Explanation:

To speed up the model tuning process when dealing with a large number of model configurations, parallelizing the hyperparameter search using Hyperopt is an effective approach. Hyperopt provides tools like SparkTrials which can run hyperparameter optimization in parallel across a Spark cluster.

Example:

from hyperopt import fmin, tpe, hp, SparkTrials search_space = { 'x': hp.uniform('x', 0, 1), 'y': hp.uniform('y', 0, 1) } def objective(params): return params['x'] ** 2 + params['y'] ** 2 spark_trials = SparkTrials(parallelism=4) best = fmin(fn=objective, space=search_space, algo=tpe.suggest, max_evals=100, trials=spark_trials)

Hyperopt Documentation

Question 6

Question Type: MultipleChoice

A machine learning engineer has created a Feature Table new_table using Feature Store Client fs. When creating the table, they specified a metadata description with key information about the Feature Table. They now want to retrieve that metadata programmatically.

Which of the following lines of code will return the metadata description?

Options:

A- There is no way to return the metadata description programmatically.
B- fs.create_training_set('new_table')
C- fs.get_table('new_table').description
D- fs.get_table('new_table').load_df()
E- fs.get_table('new_table')

Answer:

C

Explanation:

To retrieve the metadata description of a feature table created using the Feature Store Client (referred here as fs), the correct method involves calling get_table on the fs client with the table name as an argument, followed by accessing the description attribute of the returned object. The code snippet fs.get_table('new_table').description correctly achieves this by fetching the table object for 'new_table' and then accessing its description attribute, where the metadata is stored. The other options do not correctly focus on retrieving the metadata description. Reference:

Databricks Feature Store documentation (Accessing Feature Table Metadata).

Thank You for trying Databricks-Machine-Learning-Associate PDF Demo

To try our Databricks-Machine-Learning-Associate practice exam software visit link below

https://prepbolt.com/Databricks-Machine-Learning-Associate.html

Start Your Databricks-Machine-Learning-Associate Preparation

Use Coupon “SAVE50” for extra 50% discount on the purchase of Practice Test Software. Test your Databricks-Machine-Learning-Associate preparation with actual exam questions.