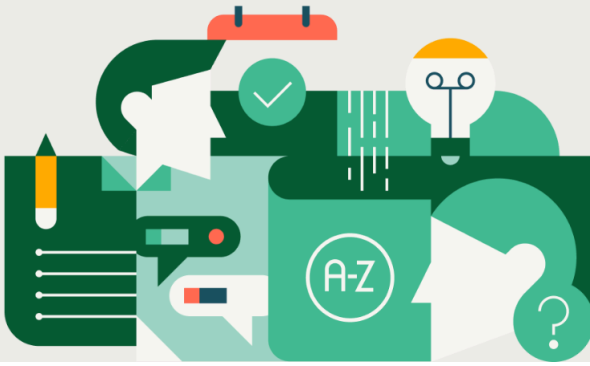
databricks

Databricks 試験ガイド

Databricks Certified 機械学習 Associate



試験ガイドへのフィードバックを提供する

情報を更新する

更新	日付
試験公開日	2025年3月1日
試験ガイド更新日	2025年3月1日

試験日の2週間前に、当社のウェブページで試験ガイドを確認し、現在のバージョンの試験に向けて準備していることをご確認ください。

この試験ガイドの目的

この試験ガイドの目的は、試験の概要と出題範囲をお伝えし、受験準備が整っているかどうかを判断するお手伝いをすることです。このドキュメントは、試験に変更があった場合に更新され（変更の発効日も記載されます）、自信を持って準備を進めていただけるようになっています。このガイドは、以上に記載された発効日時点で有効な試験バージョンを対象としています。

対象者の説明

Databricks Certified Machine Learning Associate 認定試験は、Databricks を使用して基本的な機械学習タスクを実行する個人の能力を評価します。これには、Databricks およびその機械学習機能（AutoML、Unity Catalog、MLflow の一部の機能など）を理解し活用する能力が含まれます。また、データを探索し、特徴量エンジニアリングを実行する能力も評価されます。さらに、トレーニング、チューニング、評価および選択を通じたモデル構築の能力も評価されます。最後に、機械学習モデルをデプロイする能力が評価されます。この認定試験に合格した方は、Databricks およびその関連ツールを使用して基本的な機械学習タスクを完了できると期待されます。

試験について

項目	詳細
採点対象の問題数	採点対象の多肢選択式および複数選択式の問題 48 問
採点対象外の問題	採点対象外の問題が最大 10 問含まれる場合があります（後述の「採点対象外のコンテンツ」を参照）
出題される問題の総数	採点対象の問題 48 問＋採点対象外の問題
制限時間	90 分（採点対象外の問題に対応するための追加時間を含みます）
受験料	200 米ドル（現地の法律に基づき、適用される税金が別途かかります）
受験方法	オンライン監督付き試験またはテストセンター
提供言語	英語、日本語、ブラジルポルトガル語（pt-BR）、韓国語
補助資料	使用不可
前提条件	なし（ただし、関連トレーニングの受講を強く推奨します）
推奨される経験	Databricks Platform での実務経験を 6 か月以上、試験の出題範囲に沿って有することを強く推奨します
有効期間	2年

再認定：認定ステータスを維持するには、**[2]** 年ごとに再認定が必要です。再認定を受けるには、現行の本番試験を受験する必要があります。

採点対象外のコンテンツ：試験には、将来の使用に向けて統計的な情報を収集するために、採点対象外の問題が含まれる場合があります。これらの問題は試験中に識別されず、スコアにも影響しません。また、これらの問題に対応するための追加時間が制限時間に考慮されています。

推奨される準備

トレーニング

- インストラクター主導**: Machine Learning with Databricks**
- 自己学習型（**Databricks Academy** にて利用可能）：
    - 機械学習のためのデータ準備
    - 機械学習モデルのデプロイメント

- 機械学習モデル開発
- 機械学習操作

AI Prep ガイド - すべての Databricks 試験

Databricks AI Prep Guide ガイドでは、普段お使いのAIチャットボットを任意のDatabricks認定試験向けのパーソナライズされたチューターに変える方法をご紹介します。公式試験ガイドを活用してAIに適切な前提知識を与え、AIが提示しがちな古い回答を見極め、自分の弱点を発見し、あらゆるDatabricks試験で求められる実践的な演習と組み合わせる方法を解説します。

ハンズオン演習

何を練習すればよいかわからない場合は、AI Prep ガイド（以上のAI Prep ガイドセクションを参照）をご確認ください。このガイドには「ステップ 5 — ハンズオンの最低要件」というプロンプトが含まれており、AI チャットボットがこの試験の目標にマッピングされた6〜10個のハンズオンタスクの個人向けチェックリストを生成します。各タスクは Databricks Free Edition で完了できる小さな規模のものです。

Databricksドキュメント

準備を進める際は、この試験でカバーされるプラットフォーム領域について、公式のDatabricksドキュメントを学習してください。それは最も新しく、権威あるソースです。各トピックをどの程度深く学習するかを判断し、古い情報ではなく最新の非推奨でない機能を学んでいることを確認するために活用してください。公式ドキュメント：Databricksドキュメント

試験の概要

セクション 1: Databricks Machine Learning (38%)

- MLOps 戦略のベストプラクティスを特定する
- 機械学習ランタイムを使用する利点を特定する
- AutoML がモデル/機能の選択をどのように促進するかを特定します。
- AutoML がモデル開発プロセスにもたらす利点を特定する
- Databricks の Unity Catalog においてアカウントレベルで Feature Store テーブルを作成することと、ワークスペースレベルで作成することの利点を特定してください。
- Unity Catalog に feature store テーブルを作成する
- Feature Store テーブルにデータを書き込む
- Feature Store テーブルの機能を使用してモデルをトレーニングする。
- feature store テーブルの機能を使用してモデルをスコアリングします。
- オンラインとオフラインの機能テーブルの違いを説明してください。
- MLflow Client API を使用して最適な実行を特定します。

- MLflow の実行において、メトリクス、アーティファクト、およびモデルを手動でログします。
- MLflow UIで利用可能な情報を特定する
- Unity Catalog レジストリに MLflow Client API を使用してモデルを登録する
- ワークスペースレジストリと比較して、Unity Catalog レジストリにモデルを登録する利点を特定する
- コードのプロモーションがモデルのプロモーションより望ましいシナリオ、およびその逆のシナリオを特定する
- モデルのタグを設定または削除する
- エイリアスを使用してチャレンジャーモデルをチャンピオンモデルに昇格させる

セクション2: データ処理 (19%)

- `.summary()` または dbutils のデータサマリーを使用して、Spark DataFrame のサマリー統計をコンピュートする。
- 標準偏差またはIQRに基づいてSpark DataFrameから外れ値を除去する
- カテゴリ特徴量または連続特徴量のビジュアライゼーションを作成する
- 適切な方法を使用して、2つのカテゴリカル特徴量または2つの連続特徴量を比較する
- 平均値、中央値、最頻値による欠損値の補完を比較・対比する
- 最頻値、平均値、または中央値で欠損値を補完する
- カテゴリカル特徴量にワンホットエンコーディングを使用する
- one-hot encoding が適切である、または適切でないモデルの種類やデータセットを特定し、説明してください。
- ログスケール変換が適切なシナリオを特定する

セクション 3: モデル開発 (31%)

- 機械学習の基礎を活用して、特定のモデルシナリオに適したアルゴリズムを選択する。
- トレーニングデータのデータ不均衡を軽減する方法を特定する
- エスティメーターとトランスフォーマーを比較する
- トレーニングパイプラインを開発する
- Hyperopt の fmin オペレーションを使用してモデルのハイパーパラメーターをチューニングする
- ハイパーパラメータのチューニング手法として、ランダムサーチ、グリッドサーチ、またはベイジアンサーチを実行する。
- シングルノードモデルを並列化してハイパーパラメータチューニングを行う
- クロスバリデーションをトレーニング・バリデーション分割と比較した場合の利点と欠点を説明してください。
- モデルのフィッティングの一部として交差検証を実行します。
- グリッドサーチとクロスバリデーションのプロセスと組み合わせてトレーニングされるモデルの数を特定してください。
- 一般的な分類メトリクスを使用します: F1、Log Loss、ROC/AUC など
- 一般的な回帰メトリクスを使用します（RMSE、MAE、R-squared など）。
- 与えられたシナリオの目標に対して最も適切なメトリクスを選択する
- 評価メトリクスの計算や予測の解釈を行う前に、対数変換された変数を指数変換する必要性を特定する

- モデルの複雑さとバイアス・バリアンストレードオフがモデルのパフォーマンスに与える影響を評価する

セクション4: モデルのデプロイ (12%)

- バッチ、リアルタイム、ストリーミングという各 model serving アプローチの違いと利点を特定する。
- カスタムモデルをモデル endpoint にデプロイする
- Pandas を使用してバッチ推論を実行する
- Delta Live Tables を使用したストリーミング推論の実行方法を特定する
- リアルタイム推論のためにモデルをデプロイしてクエリーする
- リアルタイム推論のためにエンドポイント間でデータを分割する

サンプル問題

サンプル問題の目的は、試験ガイドに記載されている目標を示し、各目標に対応するサンプル問題を提供することです。試験ガイドには試験で出題される可能性のある目標が記載されています。最善の準備方法は、上記の試験日程に対応した試験の全概要を確認することです。

問題 1

目的：***Unity Catalog*** に ***Feature Store*** テーブルを作成する。

あるデータサイエンティストが、モデルで使用するための特徴量テーブルを作成したいと考えています。このデータサイエンティストは Unity Catalog が有効なワークスペースで作業しており、この特徴量テーブルを Unity Catalog に保存して管理させたいと思っています。この特徴量テーブルを作成する正しい方法は何ですか？

A. 通常通りデータを含む Delta テーブルを作成し、Python の FeatureStoreClient の `register_table` メソッドを使用して、Unity Catalog に特徴量テーブルとして登録します。
B. SQL を使用して `AS FEATURE STORE` 句を指定し、Unity Catalog に空の Delta テーブルを作成してから、データを書き込みます。
C. Python の FeatureEngineeringClient の create_table メソッドを使用してテーブルを作成し、そこにデータを書き込みます。
D. Unity Catalog にデータを含む Delta テーブルを作成し、SQL の ALTER TABLE コマンドを使用して、SET AS FEATURE STORE 句でそれを特徴量テーブルとして設定します。

質問 2

目的：最頻値、平均値、または中央値を使用して欠損値を補完する。

データサイエンティストが、連続した機能の欠損値を補完する必要があります。最小限の労力で正確な結果を得たいと考えています。これを実現するのはどの戦略ですか？

A. 機能の分布に基づいて最適な手法を自動的に選択する sklearn の SimpleImputer を使用する。
B. 値の分布を調べ、レビューの結果に基づいて適切な補完方法を選択します。

C. 連続値の列に対して最も適切な補完方法である .mean() を使用する。
D. 連続値の列に対して最も適切な補完である `.mode()` を使用する。

質問 3

目標: トレーニングデータにおけるデータの不均衡を軽減する方法を特定する。

データサイエンティストが、顧客がサブスクリプションサービスを解約するかどうかを予測するモデルを開発しています。データセットは高度に不均衡であり、インスタンスのわずか10%が解約する顧客を表しています。モデルが多数クラスに偏ることなく、少数クラスを効果的に識別できるようにしたいと考えています。クラスの不均衡による非解約顧客へのモデルのバイアスを直接軽減する戦略はどれですか？

A. 特徴量を正規化して同じスケールに揃え、モデルのパフォーマンスを向上させます。
B. モデルのトレーニング中に少数クラスに対してより高い誤分類コストを割り当てることで、コスト考慮型学習を使用します。
C. 解約しない顧客に関するデータをさらに収集して、トレーニングデータセットのサイズを増やします。
D. より単純なモデルを使用して過学習を軽減し、少数クラスへの汎化性能を向上させます。

質問 4

目的：グリッドサーチとクロスバリデーションのプロセスと組み合わせてトレーニングされるモデルの数を特定する。

データサイエンティストが、scikit-learn の 5 分割交差検証と GridSearchCV を使用してサポートベクターマシン（SVM）モデルのチューニングを行っています。パラメーターグリッドには最適化すべき 3 つのハイパーパラメーターが含まれています。C の値は [0.1, 1, 10]、kernel の選択肢は ['linear', 'rbf']、gamma の値は [0.01, 0.1, 1] です。合計で何種類のモデルが学習されますか？

A. 90
B. 18
C. 1
D. 上記のいずれでもない。

質問 5

目標: ***Delta Live Tables*** を使用したストリーミング推論の実行方法を特定する。

ある企業は、数千人のユーザーを持つポッドキャストプラットフォームを運営しています。この企業は、リスニング、一時停止、ポッドキャストの終了といったユーザーイベントの10分間のランニングウィンドウに基づいて、ポッドキャストのエンゲージメントの低下を検出する異常検出アルゴリズムを実装しています。ある機械学習エンジニアは、このモデルを、1秒あたり数万件のイベントを処理する必要がある本番運用のデータパイプラインにデプロイしたいと考えています。イベントの量が1日を通じて変動するため、エンジニアはパイプラインのコンピュートを動的にリサイズできる必要があります。これらの要件を満たすパイプライン設計のアプローチはどれですか？

A. アルゴリズムをSpark UDFとして適用するDelta Live Tablesパイプラインを作成する。

B. アルゴリズムをSpark UDFとして適用する構造化ストリーミングジョブを作成する。
C. model serving endpointを作成し、endpointを呼び出すカスタムUDFを使用するDelta Live Tables パイプラインを作成します。
D. model serving endpointを作成し、endpointを呼び出すカスタムUDFを使用する構造化ストリーミングジョブを作成します。

質問 6

目的：エイリアスを使用して、チャレンジャーモデルをチャンピオンモデルに昇格させる。

チームは Unity Catalog のモデルレジストリにモデルの連続したバージョンを登録しています。バリデーションにより、新たに登録されたバージョンが現在本番運用のトラフィックを処理しているバージョンを上回ることが示された後、推奨されるアプローチに従って、本番運用が読み取る対象として新しいバージョンをマークしたいと考えています。何をすべきでしょうか？

A. `MlflowClient().transition_model_version_stage()` を使用して、新しいバージョンを「Production」ステージに移行します。
B. `MlflowClient().set_registered_model_alias()` を使用して、新しいモデルバージョンに "champion" エイリアスを割り当てます。
C. 現在のチャンピオンモデルバージョンを削除して、新しいバージョンのみがサービング可能な状態にします。
D. 新しいモデルバージョンに "champion" = "true" タグを設定するには、`MlflowClient().set_model_version_tag()` を使用します。

質問 7

目的: ***MLflow Client API*** を使用して最良の実行を特定する。

あるデータサイエンティストが、単一のエクスペリメントに対して数十件のMLflow実行をログに記録しており、それぞれに `validation_rmse` メトリクスが記録されています。MLflow Client API を使用して、`validation_rmse` が最も低い単一の実行をプログラムで取得したいと考えています。これを実現するアプローチはどれですか？

A. エクスペリメント ID を指定して `search_runs` を呼び出し、`order_by=["metrics.validation_rmse ASC"]` を設定して、最初に返された実行を取得します。
B. MLflow UI でエクスペリメントを開き、runs テーブルを validation_rmse で並べ替えて、先頭の行を確認します。
C. クライアントで実行の一覧を取得し、最新の開始時刻を持つ実行を選択します。
D. エクスペリメント名を指定して `get_run` を呼び出し、そのエクスペリメントの最良の実行を直接返します。

質問 8

目的：ログスケール変換が適切なシナリオを特定する。

データサイエンティストが住宅価格の回帰モデル向けに機能を準備しています。ある数値機能と価格ターゲットはいずれも強く右に歪んでおり、数桁にわたる非常に大きな値が少数存在します。対数スケール変換を適用するのが最も適切な状況はどれですか？

A. カテゴリカルな機能が多数の異なるレベルを持ち、そのカーディナリティを削減する必要がある場合。
B. 機能がすでにほぼ正規分布しており、ゼロ付近を中心としている場合。
C. 機能またはターゲットが右に歪んでいて数桁にわたる場合に、大きな値を圧縮して分布をより対称に近づけるため。
D. 機能に負の値が含まれており、それを正の範囲にシフトする必要がある場合。

質問 9

目標：モデルの複雑さとバイアス・バリアンストレードオフがモデルのパフォーマンスに与える影響を評価する。

評価中に、データサイエンティストはモデルがトレーニングセットでは非常に低いエラーを達成しているが、検証セットでは大幅に高いエラーを示していることを観察しました。バイアスと分散のトレードオフに基づいて、何が起きており、適切な対応は何ですか？

A. モデルは高いバイアス（アンダーフィッティング）を持っています。データサイエンティストはモデルの複雑さを増やすべきです。
B. モデルは特徴量が少なすぎることによる高いバイアスを持っているため、データサイエンティストは追加の特徴量をエンジニアリングする必要があります。
C. トレーニングエラーと検証エラーの両方が高く、高い分散を示しています。データサイエンティストはモデルをより長くトレーニングする必要があります。
D. モデルは高い分散（過学習）を示しています。データサイエンティストはモデルの複雑さを下げるか、正則化を追加する必要があります。

質問 10

目標: バッチ、リアルタイム、ストリーミングという ***model serving*** アプローチの違いと利点を特定する。

ある小売業者は、翌日のマーケティングセグメントを作成するために、毎晩1回すべての顧客ベースにスコアリングを行っています。低レイテンシのリクエストごとの予測は必要ありません。このワークロードに最も適切でコスト効率の高い model serving アプローチはどれですか？

A. 毎晩のスケジュールですべてのレコードをスコアリングするバッチ推論。
B. REST model serving endpoint を通じてリアルタイム推論を行い、顧客ごとに1回クエリを実行します。
C. 継続的に実行される構造化ストリーミングパイプラインを使用したストリーミング推論。
D. 常時稼働のGPU model serving endpointで、可能な限り低いレイテンシーに最適化されたサイズのもの。

解答キーと解説

#	解答	説明
1	C	Unity Catalog が有効なワークスペースでは、**`FeatureEngineeringClient.create_table`** メソッドを使用して特徴量テーブル（feature table）を作成し、そこにデータを書き込みます。**`AS FEATURE STORE`** という SQL 句や **`ALTER TABLE ... SET AS FEATURE STORE`** は存在せず、**`register_table`** は現在の作成パスではありません。
2	B	正しい結果を得るには、特徴量の分布を調べ、適切な補完（imputation）手法を選択する必要があります。**`SimpleImputer`** は分布から戦略を自動選択することはなく、特定の連続値特徴量に対して盲目的に平均値（または最頻値）を適用すると不適切になる場合があります。
3	B	コスト考慮型学習（Cost-sensitive learning）は、少数派（解約／churn）クラスに対して高い誤分類コストを割り当てることで、多数派クラスへのモデルのバイアスに直接対抗します。特徴量の正規化、多数派データの追加収集、またはより単純なモデルの使用では、クラスの不均衡（class imbalance）に対処できません。
4	A	グリッドには **`3 × 2 × 3 = 18`** 通りのハイパーパラメータの組み合わせがあり、5 分割交差検証（5-fold cross-validation）では各組み合わせに対してフォールドごとにモデルを 1 つトレーニングするため、合計で **`18 × 5 = 90`** 個のモデルがトレーニングされます。
5	A	アルゴリズムを Spark UDF として適用する Delta Live Tables パイプラインは、高いイベントスループットを処理し、パイプラインのコンピュートを動的にリサイズします。モデルサービング（model serving）エンドポイントを経由してルーティングするとオーバーヘッドが増加し、Structured Streaming ジョブでは同様のマネージドな動的自動スケーリングが提供されません。
6	B	Unity Catalog のモデルレジストリ（model registry）では、昇格ワークフローにレガシーなステージ遷移ではなくエイリアスを使用します。**`set_registered_model_alias`** を使用して新しいバージョンに "**`champion`**" エイリアスを割り当てることで、本番環境が読み取るバージョンになります。ステージ遷移は非推奨のワークスペースレジストリの機能であり、以前のバージョンを削除することは破壊的で不要です。また、タグは説明的なメタデータにすぎず、提供されるバージョンを変更することはありません。
7	A	MLflow Client の **`search_runs`**（**`mlflow.search_runs`** と同様）はサーバー側の並べ替えをサポートしているため、**`metrics.validation_rmse`** の昇順で並べ替えて最初の結果を取得することで、プログラムから最適な実行（run）を取得できます。MLflow UI は Client API ではありません。直近に開始された実行が必ずしも最良とは限らず、**`get_run`** は特定の **`run_id`** を必要とし、実行のランキングは行いません。
8	C	対数変換（log transformation）は大きな値を圧縮して右側の長い裾（tail）を引き込むため、複数の桁にまたがる右に歪んだ（right-skewed）特徴量やターゲットをより対称的にします。これはカーディナリティ削減手法ではなく、すでに正規分布している特徴量には何のメリットもありません。また、0 または負の値に対しては未定義であるため、負の値を正の範囲にシフトするために使用されることもありません。
9	D	低いトレーニング誤差と大幅に高い検証誤差の組み合わせは、高い分散（high variance）、すなわち過学習（overfitting）の兆候です。適切な対処法は、モデルの複雑さを低減するか、正則化（regularization）を追加すること（またはより多くのトレーニングデータを収集すること）です。高いバイアス／学習不足（underfitting）の場合は、トレーニングセットと検証セットの両方で高い誤差が示されま

		す。
10	A	低レイテンシの要件がなく、データセット全体に対して固定スケジュールで予測を生成する場合、バッチ推論（batch inference）が最もシンプルでコスト効率の高いアプローチです。リアルタイムのエンドポイントや常時稼働の GPU エンドポイントは、ワークロードが必要としないレイテンシのためにインフラとコストを増加させます。また、ストリーミング推論は夜間の大規模な一括スコアリングジョブではなく、継続的に到着するデータを対象としています。