Large-Scale Machine Learning with Apache Spark - Comprehensive Question Bank

Question 1: Linear Regression Fundamentals and Gradient Descent (12 marks)

Part A (4 marks)

Theoretical Foundation

Given a dataset with 4 samples and 2 features, explain the mathematical foundation of linear regression. Consider the following dataset:

Sample	x_1	x_2	y
1	2	3	7
2	1	4	6
3	3	2	8
4	2	1	5

a) Write the linear regression equation in vector form (1 mark)
b) Calculate the Mean Squared Error (MSE) for initial weights w = [1, 1] and bias b = 0 (show all steps) (3 marks)

Part B (4 marks)

Manual Gradient Calculation

Using the same dataset from Part A:
a) Derive the gradient formula for MSE with respect to weights w (2 marks)
b) Calculate the gradient values for the first iteration with w = [1, 1] (show detailed calculations) (2 marks)

Part C (4 marks)

Algorithm Implementation

Write a Python function from scratch to perform one iteration of gradient descent for linear regression:

```python
```

```python
def gradient_descent_step(X, y, w, learning_rate):
    """
    Perform one step of gradient descent for linear regression

    Parameters:
    X: numpy array of shape (m, d) - feature matrix
    y: numpy array of shape (m,) - target values
    w: numpy array of shape (d,) - current weights
    learning_rate: float - learning rate

    Returns:
    new_w: updated weights
    mse: current mean squared error
    """
    # Your implementation here
```

Question 2: Distributed Computing and Stochastic Gradient Descent (10 marks)

Part A (3 marks)

Distributed Computing Challenges

A company has a dataset with 1 million samples and wants to train a linear regression model using 4 worker nodes.

a) Explain the concept of data parallelism in this context (1 mark)
b) Identify and explain the major bottleneck in distributed gradient descent (1 mark)
c) Describe how communication overhead scales with the number of workers and model parameters (1 mark)

Part B (4 marks)

SGD vs Batch Gradient Descent

Consider a dataset with m = 1000 samples. Compare batch gradient descent and stochastic gradient descent:

a) Write the weight update formula for batch gradient descent (1 mark)
b) Write the weight update formula for SGD with a mini-batch size of 32 (1 mark)
c) Calculate how many parameter updates occur in one epoch for both methods (1 mark)
d) Explain three advantages of SGD over batch gradient descent in large-scale settings (1 mark)

Part C (3 marks)

Practical Implementation

Given the following scenario: You have 10,000 samples distributed across 5 workers (2,000 samples each). Explain step-by-step how one iteration of distributed gradient descent would work, including:

- Data distribution
- Computation at each worker
- Communication between workers and parameter server
- Parameter update process

Question 3: Spark MLlib Core Concepts and Architecture (8 marks)

Part A (3 marks)

MLlib Components

Explain the four core components of Spark MLlib with specific examples: a) Transformers - provide definition and two examples (1 mark) b) Estimators - explain their relationship with transformers (1 mark)
c) Evaluators and Pipelines - describe their roles in the ML workflow (1 mark)

Part B (2 marks)

Data Requirements

MLlib has specific data type requirements for machine learning algorithms.
a) What data types are required for labels and features? (1 mark)
b) Why are these specific types necessary in a distributed computing environment? (1 mark)

Part C (3 marks)

RFormula Operations

Given a dataset with columns: `age`, `income`, `education`, `target`, explain what each RFormula expression does:

a) `"target ~ ."` (1 mark) b) `"target ~ age + income - education"` (1 mark) c) `"target ~ . + age:income + education:age"` (1 mark)

Question 4: Feature Engineering and Data Preprocessing (9 marks)

Part A (4 marks)

RFormula Implementation

You have a dataset for predicting customer satisfaction with the following structure:

```
+--------+------+----------+------------+
|service |rating|experience|satisfaction|
+--------+------+----------+------------+
|premium |  4.2 |       3 |       high|
|basic   | 3.1 |       1 |        low|
|premium |  4.8 |       5 |       high|
|standard|  3.5 |       2 |    medium |
+--------+------+----------+------------+
```

Write the PySpark code to: a) Create an RFormula transformer for the formula `"satisfaction ~ . + service:rating"` (2 marks) b) Apply the transformation and show the expected output structure (2 marks)

Part B (3 marks)

Manual Feature Engineering

For the dataset above, manually calculate the feature vector for the first row after applying the RFormula transformation. Show:
a) One-hot encoding for categorical variables (1 mark)
b) Interaction term calculation (1 mark)
c) Final feature vector structure (1 mark)

Part C (2 marks)

Data Splitting Strategy

Explain the difference between using `preparedDF.randomSplit([0.7, 0.3])` versus splitting the original DataFrame before applying transformations. What are the implications for model evaluation?

Question 5: Pipeline Implementation and Model Training (11 marks)

Part A (5 marks)

Complete Pipeline Construction

Write a complete PySpark MLlib pipeline for a binary classification problem using the following requirements:

- Use RFormula for feature engineering with the formula `"label ~ . + feature1:feature2"`
- Use DecisionTreeClassifier as the estimator
- Include proper column specifications

```python
```

```python
from pyspark.ml import Pipeline
from pyspark.ml.feature import RFormula
from pyspark.ml.classification import DecisionTreeClassifier

# Your complete implementation here
```

Part B (3 marks)

Hyperparameter Tuning Setup

Create a parameter grid for the pipeline above that includes:
a) Two different RFormula expressions (1 mark)
b) Three different values for DecisionTree maxDepth (2, 5, 10) (1 mark)
c) Two different values for DecisionTree maxBins (32, 64) (1 mark)

Part C (3 marks)

Model Evaluation and Validation

Set up a complete evaluation framework including:
a) BinaryClassificationEvaluator with areaUnderROC metric (1 mark)
b) TrainValidationSplit with 80% training ratio (1 mark)
c) Code to train the model and evaluate on test set (1 mark)

Question 6: Advanced Topics and Real-World Application (10 marks)

Part A (4 marks)

Flight Delay Prediction Scenario

You're tasked with building a model to predict flight delays using Spark MLlib. The dataset contains:

- `airline` (categorical: AA, UA, DL)
- `departure_hour` (numerical: 0-23)
- `distance` (numerical: miles)
- `weather_score` (numerical: 1-10)
- `is_delayed` (target: 0/1)

Design a complete MLlib solution including:
a) Appropriate RFormula with interaction terms (1 mark)
b) Pipeline with at least two different algorithms to compare (2 marks)
c) Evaluation strategy with cross-validation (1 mark)

Part B (3 marks)

Model Persistence and Deployment

a) Write code to save the best model from your pipeline (1 mark)
b) Write code to load and use the saved model for new predictions (1 mark)
c) Explain the advantages of MLlib's model persistence in production environments (1 mark)

Part C (3 marks)

Scalability Analysis

Consider scaling your flight delay prediction model:
a) How would you handle a dataset with 100 million flight records? Discuss data partitioning strategies (1 mark)
b) What considerations would you have for feature engineering at this scale? (1 mark)
c) How would you monitor model performance in a streaming environment? (1 mark)

Additional Practice Questions

Question 7: Mathematical Derivations (8 marks)

Part A: Derive the gradient of MSE for linear regression step-by-step, starting from the basic MSE formula. (4 marks)

Part B: Prove that minimizing MSE is equivalent to minimizing RMSE for optimization purposes. (2 marks)

Part C: Calculate the computational complexity of batch gradient descent versus SGD for m samples and d features. (2 marks)

Question 8: Comparative Analysis (7 marks)

Compare Spark MLlib with Scikit-Learn across the following dimensions:

- Data handling capabilities (2 marks)
- Scalability features (2 marks)
- Algorithm availability (1 mark)
- Ease of use and learning curve (2 marks)

Question 9: Error Analysis and Debugging (6 marks)

Given the following error scenarios in Spark MLlib, identify the problem and provide solutions:

Part A: `IllegalArgumentException: Feature column must be of type Vector` (2 marks)

Part B: `AnalysisException: Column label does not exist` (2 marks)

Part C: Poor model performance despite good training accuracy (2 marks)

Question 10: Implementation from Scratch (12 marks)

Implement a complete mini-batch gradient descent algorithm for linear regression without using any ML libraries. Your implementation should include:

a) Data preprocessing functions (3 marks)
b) Cost function calculation (3 marks)
c) Gradient computation (3 marks)
d) Training loop with convergence criteria (3 marks)

Answer Guidelines and Marking Rubrics

Theoretical Questions:

- **Full marks:** Complete explanation with correct terminology and clear understanding
- **Partial marks:** Correct concept but incomplete explanation or minor errors
- **Minimal marks:** Basic understanding shown but significant gaps

Coding Questions:

- **Full marks:** Complete, syntactically correct code with proper structure
- **Partial marks:** Mostly correct with minor syntax errors or missing components
- **Minimal marks:** Shows understanding but significant implementation issues

Mathematical Calculations:

- **Full marks:** All steps shown clearly with correct final answer
- **Partial marks:** Correct method but computational errors or missing steps
- **Minimal marks:** Attempted calculation but major errors in approach

Application Questions:

- **Full marks:** Comprehensive solution addressing all practical considerations
- **Partial marks:** Good solution but missing some practical aspects
- **Minimal marks:** Basic solution with limited practical awareness

---

Exam Preparation Tips

1. **Practice Manual Calculations:** Be prepared to show detailed steps for gradient descent, MSE calculations, and feature transformations.
2. **Understand MLlib Architecture:** Know the relationships between Transformers, Estimators, Evaluators, and Pipelines.

3. **Code Implementation:** Practice writing both from-scratch implementations and MLlib-specific code.
4. **Real-world Applications:** Study how to apply concepts to practical scenarios like the flight delay prediction example.
5. **Theoretical Foundations:** Understand the mathematical principles behind gradient descent and linear regression.
6. **Distributed Computing Concepts:** Be clear on the challenges and solutions in large-scale machine learning.