Low-Precision Machine Learning

Objective:

In this assignment, you will explore the effects of low-precision data representations on the performance of traditional machine learning models. You will implement quantized versions of the input data and evaluate the models using cross-validation for a more reliable comparison against full-precision models.

Assignment Tasks:

Task 1: Quantization of Input Data

1. **Objective**: Quantize the input dataset to simulate low-precision machine learning.
2. **Steps**:
    - Choose a dataset (e.g., Iris, Wine, or any other complex dataset).
    - Apply **uniform quantization** to the features, experimenting with 8-bit, 4-bit, and 2-bit precision.
    - Use binning techniques or NumPy functions to achieve quantization.

Code Hints:

```python
import numpy as np
 def quantize_data(data, bits):
     levels = 2  bits
     min_val = np.min(data, axis=0)
     max_val = np.max(data, axis=0)
     quantized_data = np.round(((data - min_val) / (max_val - min_val)) *
(levels - 1)) * (max_val - min_val) / (levels - 1) + min_val
     return quantized_data
```

Task 2: Training and Evaluation with Quantized Data (Cross-Validation)

1. **Objective**: Train and evaluate traditional machine learning models using quantized data.
2. **Steps**:
    - Use models like **Decision Trees**, **k-Nearest Neighbors (k-NN)**, and **Support Vector Machines (SVM)**.
    - Perform **cross-validation** with 5-fold stratified splits for more robust results.
    - Train each model using quantized data (8-bit, 4-bit, 2-bit) and compare their performance.
    - Use metrics such as **accuracy** and **standard deviation** of accuracy across the cross-validation folds.

Task 3: Comparison with Full-Precision Models

1. **Objective**: Compare the performance of low-precision models with full-precision models.
2. **Steps**:
    - Train the same models using **full-precision** data without quantization.
    - Compare the results (mean accuracy, standard deviation) for full-precision and quantized data (8-bit, 4-bit, and 2-bit).
    - Analyze the effect of precision reduction on model performance, specifically which models are more sensitive to lower precision.

Task 4: Impact Analysis

1. **Objective**: Analyze how precision reduction affects model performance.
2. **Steps**:
    - Plot performance metrics (accuracy) for each model at different precision levels (full-precision, 8-bit, 4-bit, 2-bit).
    - Discuss how different models (Decision Trees, k-NN, SVM) are impacted by lower precision and hypothesize why certain models are more robust than others to quantization.

Code Hints:

```python
import matplotlib.pyplot as plt
# Example plot for accuracy vs. precision level
precision_levels = [8, 4, 2]
accuracy_scores = [accuracy_8bit, accuracy_4bit, accuracy_2bit]

plt.plot(precision_levels, accuracy_scores, marker='o')
plt.xlabel('Precision Level (bits)')
plt.ylabel('Accuracy')
plt.title('Accuracy vs Precision Level')
plt.show()
```

Task 5: Logistic regression training using quantized gradient

1. Explore quantized gradient updates in stochastic gradient descent (SGD) for training a logistic regression model with quantized inputs.
2. Modify the logistic regression training algorithm to use quantized gradient updates and analyze its impact on model convergence and performance.
3. Compare the quantized gradient model with full precision logistic regression.

Deliverables:

1. Code: Submit all code as `.py` or `.ipynb` files, ensuring that each task is clearly labelled.
2. Report: Include a brief report (2-3 pages) summarizing your findings:
3. Discuss the impact of low-precision training on different machine learning models.
4. Include comparisons of model performance at different precision levels (full-precision, 8-bit, 4-bit, and 2-bit).
5. Analyze which models are most and least affected by precision reduction.