Programming Assignment: Logistic Regression and k-Nearest Neighbors (kNN)

Submission Deadline: 13th November, 2024, 2:59 PM

Objective

The goal of this assignment is to build and evaluate classifiers for binary classification. You will implement both Logistic Regression and k-Nearest Neighbors (kNN) classifiers from scratch, using a given dataset.

Dataset

Use the Breast Cancer Wisconsin (Diagnostic) dataset for binary classification:

- **Download Link**: Breast Cancer Dataset
- **Features**: This dataset contains 30 numerical features for each instance.
- **Target**: A binary label indicating if a case is malignant (1) or benign (0).

Note: Download the data and store it in a CSV file format named `breast_cancer.csv` with headers: `id, diagnosis, feature1, feature2, ..., feature30`.

Tasks

1. Data Preprocessing

- **Load the Dataset**: Use `pandas` to load the dataset.
- **Encoding**: Convert the `diagnosis` column to binary labels (1 for malignant, 0 for benign). In the dataset, M is for malignant and B is for benign.
- **Train-Test Split**: Split the data into training and testing sets (e.g., 80% training and 20% testing).

2. Logistic Regression Implementation

Implement the following functions:

- `sigmoid(z)`: Compute the sigmoid of z, where z is a linear combination of weights and features.
- `initialize_weights(n_features)`: Initialize the weights and bias term to zero for n_features.
- `compute_cost(X, y, weights, bias)`: Compute the binary cross-entropy cost function for logistic regression.
- `optimize_weights(X, y, weights, bias, learning_rate, num_iterations)`: Perform gradient descent to optimize weights and bias.
- `train_logistic_regression(X_train, y_train, learning_rate, num_iterations)`: Use the above helper functions to train logistic regression on the training set.
- `predict_logistic_regression(X_test, weights, bias)`: Predict binary labels for the test set using the trained model.

Evaluation Metric:

- **Accuracy**: Print the accuracy of the model on both train and test sets.

3. k-Nearest Neighbors (kNN) Implementation

Implement the following functions:

- `euclidean_distance(x1, x2)`: Calculate the Euclidean distance between two feature vectors x1 and x2.
- `get_neighbors(X_train, X_test_instance, k)`: Retrieve the k nearest neighbors to a given test instance from the training set.
- `predict_kNN(X_train, y_train, X_test, k)`: For each instance in `X_test`, use `get_neighbors` to predict the label by majority vote from k neighbors.

Evaluation Metric:

- **Accuracy**: Measure and print the accuracy of the kNN model on both train and test sets for various values of k.

4. Comparison and Analysis

- Compare the performance of Logistic Regression and kNN on the test set and discuss:
  - Which model performs better and why?
  - How does the choice of k affect the performance of kNN?
  - What are the strengths and limitations of Logistic Regression and kNN for this classification problem?

Requirements

- Implement all functions from scratch, without using any pre-built machine learning libraries (e.g., scikit-learn).
- Use `Numpy` for numerical operations and `pandas` for data handling.
- For optional visualization, you may use `matplotlib`.

Submission

Submit a Jupyter Notebook (`.ipynb`)

- Your implemented code for all functions.
- Accuracy results for both Logistic Regression and kNN.
- Analysis as discussed in Task 4.
- Experiment with different values of k for kNN (e.g., k=1,3,5,7).