Homework Assignment 1

Instructions

In this homework, you are required to complete all of the problems outlined below. Each problem is worth 1 point. You must submit your solution to each problem in a separate Google Colab notebook.

Deadline: Lab Class in the week of 4th to 8th of November.

Problem 1: Finding the Optimal k and Bootstrap Iterations

In this task, you will use the K-Nearest Neighbors (KNN) classifier to classify handwritten digits from the MNIST dataset. The goal is to determine:

1. The optimal number of neighbors (k) for the KNN classifier.
2. The optimal number of bootstrap iterations to obtain a confident estimate of the model's accuracy.

You will:

1. Download and preprocess the MNIST dataset (use 10% of the dataset for faster experimentation).
2. Implement bootstrap resampling to evaluate the KNN classifier for various values of k.
3. Experiment with different numbers of bootstrap iterations.
4. Determine the optimal values of k and the minimum number of bootstrap iterations required for a confident result.

Steps to Follow:

1. **Preprocess the MNIST dataset:** Normalize the pixel values (e.g., divide by 255 to scale between 0 and 1) and select a random 10% subset of the dataset.
2. **KNN Classifier:** Use the `KNeighborsClassifier` from `sklearn.neighbors`. Iterate over different values of k, specifically $k = 1, 2, \ldots, 10$.
3. **Bootstrap Resampling:**
   - For each value of k, perform several bootstrap iterations:
     - Resample the entire dataset with replacement to create a training set.
     - Train the KNN model on the resampled data.
     - Test the model on the remaining data points (out-of-bag data).
     - Compute and store the accuracy on the out-of-bag data for each iteration.
4. **Determine Optimal k and Bootstrap Iterations:**
   - Experiment with different numbers of bootstrap iterations (e.g., try 1, 2, 11, 22, 44, 88, etc.).
   - Calculate the mean accuracy across bootstrap iterations for each k.
   - Determine the optimal k and the number of iterations required for a stable estimate.

5. **Plot:**
    - Plot the mean accuracy for each value of k.
    - Include another plot showing how the mean accuracy stabilizes with increasing bootstrap iterations.

Problem 2: Finding the Optimal k Using Leave-One-Out Cross-Validation

In this task, you will use the K-Nearest Neighbors (KNN) classifier to classify handwritten digits from the MNIST dataset. The goal is to determine:

1. The optimal number of neighbors (k) for the KNN classifier.
2. Evaluate the classifier's performance using leave-one-out cross-validation (LOO CV).

You will:

1. Download and preprocess the MNIST dataset (use 10% of the dataset for faster experimentation).
2. Implement leave-one-out cross-validation to evaluate the KNN classifier for various values of k.
3. Determine the optimal value of k based on LOO CV results.

Steps to Follow:

1. **Preprocess the MNIST dataset:** Normalize the pixel values (e.g., divide by 255 to scale between 0 and 1) and select a random 10% subset of the dataset.
2. **KNN Classifier:** Use the `KNeighborsClassifier` from `sklearn.neighbors`. Iterate over different values of k, specifically $k = 1, 2, \ldots, 10$.
3. **Leave-One-Out Cross-Validation:**
    - For each value of k, perform LOO CV:
        - Train the KNN model on $n - 1$ data points, where n is the total number of samples.
        - Test the model on the single data point left out.
        - Repeat this for all data points and compute the overall accuracy.
4. **Determine the Optimal k:**
    - Calculate the accuracy for each value of k.
    - Determine the value of k with the highest accuracy.
5. **Plot:**
    - Plot the accuracy for each value of k based on the LOO CV results.

Problem 3: K-Means Clustering and Centroid Norm Averaging

In this task, you will generate $n = 10,000$ samples from a 2D Gaussian distribution and apply the k-means algorithm to cluster the data into $k = 3$ clusters. You will calculate the average of the centroid norms and analyze how the cumulative average converges over multiple iterations. The goal is to empirically estimate the number of iterations required to achieve a stable result with 9-digit accuracy.
Do this all in Python:

1. Generate $n = 10,000$ samples (x_1^i, x_2^i) from a 2D Gaussian distribution.

2. Apply the k-means clustering algorithm with $k = 3$.
3. For each centroid (C_1^j, C_2^j), $j = 1, 2, 3$, calculate its norm: $\|C^j\| = \sqrt{(C_1^j)^2 + (C_2^j)^2}$.
4. Average the norms of the three centroids and call this value R_1.
5. Repeat this process 88 times (or more) to get the values $R_1, R_2, \ldots, R_{88}$.
6. Calculate the cumulative average of the R_i values as $\frac{1}{n}\sum_{i=1}^{n} R_i$ for $n = 1, 2, \ldots, 88$ (or more).
7. Plot the cumulative averages and analyze how the value converges.
8. **Task:** Empirically estimate the number of iterations needed instead of 88 to achieve 9-digit accuracy in the cumulative averages.