HW1_Group14_Vidya_PujaAnkitha_PavanaAasritha_Pushpahas

September 8, 2023

HW 1 CODE

By Group 14 1. Vidya, Ramineni (1002082819) 2. Puja Ankitha, Ivaturi (1002083111) 3. Pavana Aasritha, Pendyala (1002114927) 4. Pushpahas, Kuchipudi (1002040696)

[2]:
```
#Installing and Importing the h5py library which provides a interface to the␣
 ↪HDF5 file format
#!pip install h5py
import h5py

# Load the USPS dataset from the HDF5 file
with h5py.File("usps.h5", "r") as file:
    # Get the list of keys in the HDF5 file
    keys = list(file.keys())
    # Print the keys to see the available datasets
    print(keys)
```

```
['test', 'train']
```

[3]:
```
#Extract the training and testing data and labels from the HDF5
with h5py.File("usps.h5", "r") as file:
    X_train = file["train"]["data"][:]
    y_train = file["train"]["target"][:]
    X_test = file["test"]["data"][:]
    y_test = file["test"]["target"][:]
```

[4]:
```
#Performing Pre processing on the training and testing data
#!pip install scikit-learn
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[5]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, precision_score, recall_score
from sklearn.model_selection import cross_val_score
```

```
#Initialize a list to store cross-validation results
cv_scores = []
list_scores = []



#####  TASK 4: Change different K for KNN   #####

#Test different values of k
k_values = [3, 6, 9, 12, 15]


# Performing K-fold cross-validation for different values of K using K nearest␣
 ↪neighbors(KNN) classifier
for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)

    #####   TASK 1: Cross validation and accuracy   #####
    scores = cross_val_score(knn, X_train, y_train, cv=5, scoring='accuracy')
    list_scores.append(scores)
    cv_scores.append(scores.mean())

#Find the best k value with the highest cross-validation accuracy
#print(cv_scores)
while len(cv_scores)>0:
    #Finding the value of k that corresponds to the highest cross validation␣
 ↪score
    best_k = k_values[cv_scores.index(max(cv_scores))]
    print(f"For k = {best_k}")
    print("For k = {0}, The corresponding list scores of which the mean is␣
 ↪calculated to get cv score is: {1}".format(best_k,list_scores[cv_scores.
 ↪index(max(cv_scores))]))
    knn = KNeighborsClassifier(n_neighbors=best_k)
    #Fitting the KNN classifier on the training data
    knn.fit(X_train, y_train)
    #Predicting the labels for the test data
    y_pred = knn.predict(X_test)

    #Calculate accuracy
    print()
    accuracy = (y_pred == y_test).mean()
    print(f"Test Accuracy: {accuracy:.6f}")

    #####   TASK 2: Confusion Matrix     #####
    conf_matrix = confusion_matrix(y_test, y_pred)
    precision = precision_score(y_test, y_pred, average='macro')
```

```
    recall = recall_score(y_test, y_pred, average='macro')
    print()
    print("Confusion Matrix:")
    print(conf_matrix)

    #####   TASK 3: Precision and Recall  #####
    print()
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")


    cv_scores.remove(max(cv_scores))
    k_values.remove(best_k)


    print("=======================================")
    print("+++++++++++++++++++++++++++++++++++++++")
    print("=======================================")
    print()
    print()
```

```
For k = 3
For k = 3, The corresponding list scores of which the mean is calculated to get
cv score is: [0.96435915 0.94924554 0.95404664 0.94101509 0.96021948]

Test Accuracy: 0.927753

Confusion Matrix:
[[355   0   3   0   0   0   0   0   0   1]
 [  0 257   0   0   4   0   2   1   0   0]
 [  8   1 176   3   0   0   2   4   4   0]
 [  1   0   6 147   0   7   0   2   1   2]
 [  0   2   4   0 177   1   2   3   0  11]
 [  7   0   1   5   0 137   0   4   2   4]
 [  4   0   3   0   2   2 159   0   0   0]
 [  0   1   1   0   6   0   0 138   0   1]
 [  6   0   3   3   0   2   0   1 148   3]
 [  1   0   1   0   2   0   0   5   0 168]]

Precision: 0.9256
Recall: 0.9193
=======================================
+++++++++++++++++++++++++++++++++++++++
=======================================


For k = 6
For k = 6, The corresponding list scores of which the mean is calculated to get
```

```
cv score is: [0.96435915 0.94924554 0.95404664 0.94101509 0.96021948]

Test Accuracy: 0.922770

Confusion Matrix:
[[353   0   2   1   0   0   1   1   0   1]
 [  0 258   0   0   4   0   2   0   0   0]
 [  7   0 177   3   1   0   2   4   4   0]
 [  3   0   5 152   0   3   0   1   0   2]
 [  0   3   5   0 176   1   2   3   0  10]
 [  5   0   1  10   0 135   0   4   1   4]
 [  5   0   4   0   2   2 157   0   0   0]
 [  0   2   1   0   4   0   0 139   0   1]
 [  8   3   2   7   0   3   1   2 138   2]
 [  1   0   0   0   1   0   0   7   1 167]]

Precision: 0.9211
Recall: 0.9138
=====================================
+++++++++++++++++++++++++++++++++++++
=====================================


For k = 9
For k = 9, The corresponding list scores of which the mean is calculated to get
cv score is: [0.96435915 0.94924554 0.95404664 0.94101509 0.96021948]

Test Accuracy: 0.923767

Confusion Matrix:
[[353   0   1   1   1   0   1   1   0   1]
 [  0 258   0   0   4   0   2   0   0   0]
 [  8   1 177   1   1   0   2   4   4   0]
 [  3   0   4 149   0   7   0   2   0   1]
 [  0   4   4   0 175   0   2   4   0  11]
 [  8   0   1   7   0 134   0   4   1   5]
 [  6   0   4   0   2   1 156   0   1   0]
 [  0   3   1   0   4   0   0 139   0   0]
 [  7   3   2   3   0   1   0   2 145   3]
 [  1   0   0   0   1   0   0   6   1 168]]

Precision: 0.9237
Recall: 0.9150
=====================================
+++++++++++++++++++++++++++++++++++++
=====================================
```

```
For k = 12
For k = 12, The corresponding list scores of which the mean is calculated to get
cv score is: [0.96435915 0.94924554 0.95404664 0.94101509 0.96021948]

Test Accuracy: 0.914798

Confusion Matrix:
[[353   0   1   1   1   0   1   1   0   1]
 [  0 258   0   0   3   0   3   0   0   0]
 [  8   2 173   3   1   0   2   4   5   0]
 [  4   0   4 149   0   5   0   2   0   2]
 [  1   5   4   0 172   0   2   4   0  12]
 [  9   0   0   6   1 133   0   5   2   4]
 [  7   0   4   0   2   1 155   0   1   0]
 [  0   4   1   0   4   0   0 137   0   1]
 [  8   3   2   6   0   2   1   2 140   2]
 [  1   0   0   0   1   0   0   7   2 166]]

Precision: 0.9150
Recall: 0.9048
=====================================
+++++++++++++++++++++++++++++++++++++
=====================================


For k = 15
For k = 15, The corresponding list scores of which the mean is calculated to get
cv score is: [0.96435915 0.94924554 0.95404664 0.94101509 0.96021948]

Test Accuracy: 0.914300

Confusion Matrix:
[[353   0   1   1   1   0   1   1   0   1]
 [  0 258   0   0   3   0   3   0   0   0]
 [  8   3 170   4   2   0   1   5   5   0]
 [  4   0   4 149   0   5   0   2   0   2]
 [  1   5   4   0 172   0   1   5   0  12]
 [  9   0   0   7   1 132   0   4   2   5]
 [  5   0   4   0   2   1 157   0   1   0]
 [  0   3   0   1   4   0   0 138   0   1]
 [  7   3   2   7   0   2   2   2 138   3]
 [  1   0   0   0   1   0   0   6   1 168]]

Precision: 0.9138
Recall: 0.9045
=====================================
+++++++++++++++++++++++++++++++++++++
=====================================
```

HW 1 REPORT

By Group 14 1. Vidya, Ramineni (1002082819) 2. Puja Ankitha, Ivaturi (1002083111) 3. Pavana Aasritha, Pendyala (1002114927) 4. Pushpahas, Kuchipudi (1002040696)

Building a USPS Dataset Classifier using K-Nearest Neighbors

The main aim of this report is to provide an overview of the tasks performed to build a classifier for the USPS dataset using the K-Nearest Neighbors algorithm. The goal is to achieve over 90% accuracy on the test set, and the following tasks were executed to accomplish this: Note: Here, we ran a loop over different K's and each of them execute in such a manner that the best K I.e., the one that gives the best accuracy is resulted first, and the second-best accuracy K follows it and so on… I.e., from the best to the worst.

For all the K values taken, we achieved over 90% accuracy on the test set.

Task 1: Cross-Validation

In this task, we employed K-fold cross-validation to evaluate the performance of the KNN classifier with different values of K [3, 6, 9, 12, 15]. The purpose of cross-validation is to estimate how well the model will generalize to unseen data. To accomplish this, we imported the necessary libraries, including scikit-learn, for KNN and evaluation metrics. A list was initialized to store cross-validation results, and another list was used to record individual scores for each K. Different values of K (3, 6, 9, 12, and 15) were tested. KNN classifiers with each K-value were trained and evaluated using 5-fold cross-validation. The mean accuracy scores for each K were recorded.

And here, accuracy is calculated to check which K gives the best result of all others and proceed with that at first.

Task 2: Confusion Matrix

In this task, we calculated and displayed the confusion matrix to gain insight into the KNN classifier's performance on the test data. A confusion matrix is a valuable tool for understanding the classifier's ability to correctly classify each class. To accomplish this, we used the trained KNN classifier with the best-performing K. Predictions were made on the test data. A confusion matrix was generated to show the number of true positives, true negatives, false positives, and false negatives for each class.

Task 3: Precision and Recall

In this task, we computed the precision and recall scores to further assess the classifier's performance. Precision measures the accuracy of positive predictions, while recall assesses the ability to find all positive instances. To accomplish this, Precision and recall scores were calculated using the predicted and true labels for each class. We used the 'macro' average to compute a single precision and recall score that considers all classes.

Task 4: Change Different K for KNN

This task focused on iterating through different values of K for the KNN classifier and finding the K that achieved the highest cross-validation accuracy. To accomplish this: We repeated the K-fold cross-validation procedure for each K in the list of tested values. The K with the highest cross-validation accuracy was identified as the best- performing K. The KNN classifier with the

best K was trained on the entire training dataset. Predictions were made on the test data, and accuracy, confusion matrix, precision, and recall were computed.

And thereby, the K values are used throughout the program, one at a time, to calculate confusion matrix, recall and precision of best in descending order of accuracy.

In conclusion, this report outlined the tasks performed to build a classifier for the USPS dataset using the K-Nearest Neighbors algorithm. By conducting cross-validation, evaluating confusion matrices, and calculating precision and recall scores, we were able to assess and optimize the classifier's performance. Additionally, we iteratively selected the best K for KNN to achieve high accuracy on the test set. The results and insights gained from these tasks are valuable in understanding the classifier's strengths and weaknesses and can aid in further fine-tuning the model for better performance.

END OF HW 1