Machine Learning for Materials Scientists
Summer Semester 2024

Prof. Bernhard Eidel
Rahul Vishnu Narkhede
rahul-vishnu.narkhede@student.tu-freiberg

TECHNISCHE UNIVERSITÄT BERGAKADEMIE FREIBERG
The University of Resources. Since 1765.

Exercise 3 - K-Nearest Neighbors and K-Means Clustering

June 18, 2024

The learning objectives of this exercise are

1. to write code that implements K-Nearest Neighbors (KNN) and K-Means Clustering methods,
2. to apply the code to the popular MNIST digit dataset for classification,
3. to apply Cross Validation and Grid Search with KNN to determine the best hyperparameters.

In the given jupyter notebook `Ex03_KNN_KMeans.ipynb` complete the function `eulcidean`, the class `My_KNeighborsClassifier` and the class `My_KMeans`. Completing these shall enable the use of some subsequent code blocks with which the KNN and KMeans clustering methods are applied to the MNIST dataset as shown in the listing 1

```
# The datasets can be applied with prior PCA-based dimensionality reduction
# KNN (Supervised learning)
knn = My_KNeighborsClassifier(k = 5)
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
knn_test_accuracy = np.mean(y_pred_knn == y_test)

# KMeans (Unsupervised learning)
kmeans = My_KMeans(n_clusters = 3)
kmeans.fit(X_train)
clusters = kmeans.predict(X_test)
y_pred_kmeans, label_cluster_map = map_kmeans_labels_to_original(clusters, y_test)
kmeans_test_accuracy = np.mean(y_pred_kmeans == ytest)
```

Listing 1: Sample of using `My_KNeighborsClassifier` and `My_KMeans` classes.

The notebook also shows the `sklearn` based implementations of KNN and KMeans Clustering. The datasets are reduced to dimensionality of two, enabling plotting the samples in scatter plots and visualizing the decision boundaries of the two clustering approaches as shown in the figure 1.

However, the number of principal components shall also affect the performance of the clustering. Thus, there is a need to determine the best set of hyperparameters. In this case, we restrict ourselves to the following hyperparameters:

- **KNN**: `n_neighbors` and `n_components`.
- **KMeans**: `n_clusters` and `n_components`.

For understanding the grid search based approach of finding the best hyperparameters, the functions `k_fold_split` and `grid_search_cv` are provided.

Finally, your task is using the `sklearn` library to implement PCA and KNN along with grid search and cross-validation. Find the most suitable set of hyperparameters from the range given in the notebook. Further, answer the questions given at the end of the notebook regarding implementation using the `sklearn` library.

Step by step hints for finishing the code block are provided in the `ipython` notebook as well.

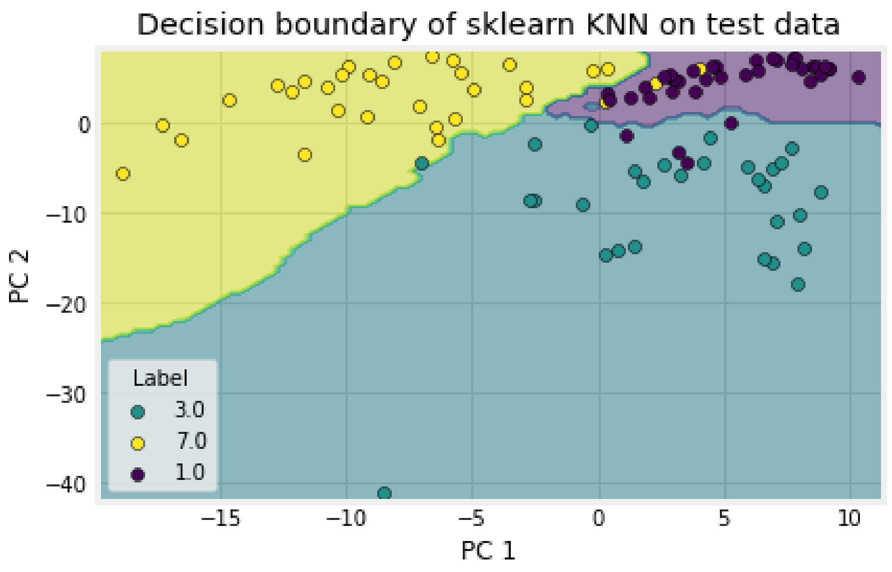

Figure 1: Visualization of decision boundary in KNN approach.

Update the code in the jupyter notebook which has been provided for the exercise. Account for the instructions and comments above.

For submission, provide a PDF document(based on LaTeX) explaining the results with:

1. The completed code block with all relevant comments (wherever required).
2. `sklearn` based implementation of KNN with PCA, grid search and cross-validation. Report the best and second-best set of parameters.
3. Visualization of the decision boundary of the final model with the best set of hyperparameters. Ensure that the plot contains a suitable legend, title and axes labels.

Deadline: 02-July-2024, 23:59h