Machine Learning

KNNC Olivetti Faces Data – Dimensionality Reduction using Random Projections

This is a classification task using KNNC.
(a) Download the Olivetti faces dataset. There are 40 classes (corresponding to 40 people), each class having 10 faces of the individual; so there are a total of 400 images. Here each face is viewed as an imgae of size 64 × 64 (= 4096) pixels; each pixel having values 0 to 255 which are ultimateley converted into floating numbers in the range [0,1]. Visit https://scikit-learn.org/0.19/datasets/olivetti_faces.html for more details.

Task 2: Here, you need to use **bootstrapping** to generate 10 more training patterns from each class (person), as follows:

(a) Let $\mathcal{X}$ be the training dataset of 400 face images.

(b) Let the set $RESAMPLES$ be empty.

(c) For each of the training patterns $X_i \in \mathcal{X}$ (for $i = 1, .., 400$) do the following:

i. Let X_i be the training pattern.

ii. Let $X_i^1, X_i^2, \cdots, X_i^P$ be the P nearest neighbors of X_i from the **remaining patterns of the same class as that of** X_i.

iii. Let

$$X_i' = \frac{1}{P+1} \sum_{j=0}^{P} X_i^j,$$

where $X_i^0 = X_i$ itself.

iv. Add X_i' to set $RESAMPLES$.

(d) Note that there are 400 patterns in $\mathcal{X}$. Obtain 400 more in $RESAMPLES$ using $P = 3$. Now update $\mathcal{X}$ as

$$\mathcal{X} = \mathcal{X} \cup RESAMPLES.$$

1.
Task 4: In this task you are supposed to reduce the dimensionality using l random projections with the values of l = 500; 1000; 1500; 2000; 2500.Use KNNC with values of K = 1; 3; 5; 10; 20; 100 on the 800×4096 data matrix obtained in problem 2 and step (d). For each value of K, use KNNC based on Minkowski distance with r = 1; 2; 1. Also consider fractional norms with r = 0:8; 0:5; 0:3. Compute the percentage accuracy using Leave-one-out-strategy and report results

CODE:

Please find the code for RANDOM PROJECTIONS committed as
KNNC_OlivettiFaceData_Task4_Dimentionality_Reduction_RandomProjections_impl.py

- X resampled data is first obtained using the bootstrapping method mentioned in TASK2.
- **SparseRandomProjection** is used to reduce the dimensionality of the data from 4096 to given values of l=500 to 2500.
- With reduced dimensionality, KNNC is applied for different values of K and R and LOO accuracy is computed.

RESULT:

- **The result is tabulated for all the L random projection values listed in the problem.**
- **Accuracy and LOO accuracy values are also tabulated and corresponding plots are generated.**

For L= 2500 – Below Table shows the Accuracy and Loo Accuracy values.

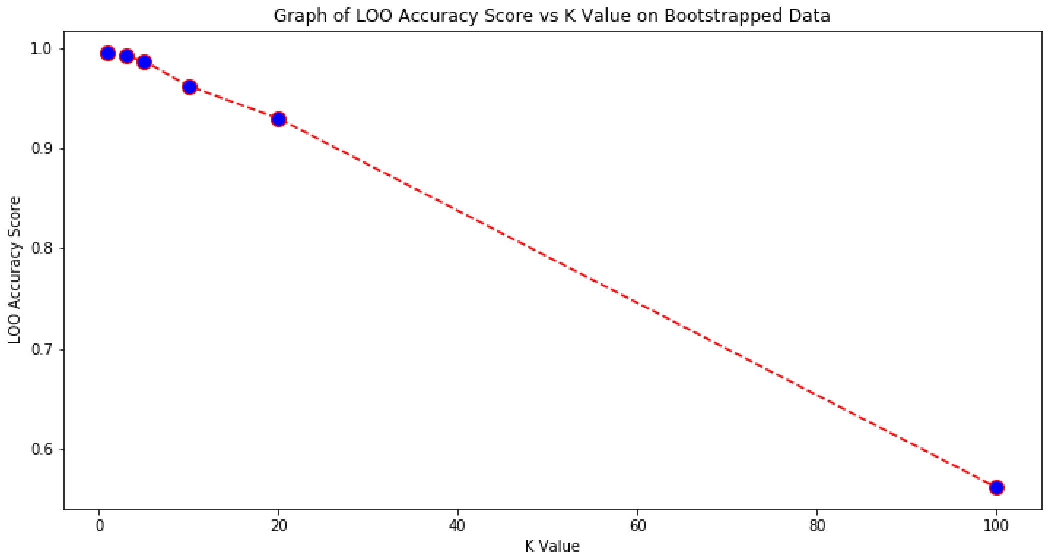
L	r - (exp value in Minkowski distance)	K	Accuracy	Leave on out Accuracy (Resamples, n=800 samples)
2500	1	1	1	0.995
		3	0.991666667	0.991
		5	0.983333333	0.988
		10	0.966666667	0.963
		20	0.908333333	0.917
		100	0.525	0.556
	2	1	1	0.995
		3	0.991666667	0.991
		5	0.9875	0.988
		10	0.970833333	0.965
		20	0.908333333	0.916
		100	0.520833333	0.554
	Inf	1	0.991666667	0.995
		3	0.983333333	0.989
		5	0.970833333	0.983
		10	0.954166667	0.949
		20	0.845833333	0.916
		100	0.583333333	0.58

Output_result_table.xlsx

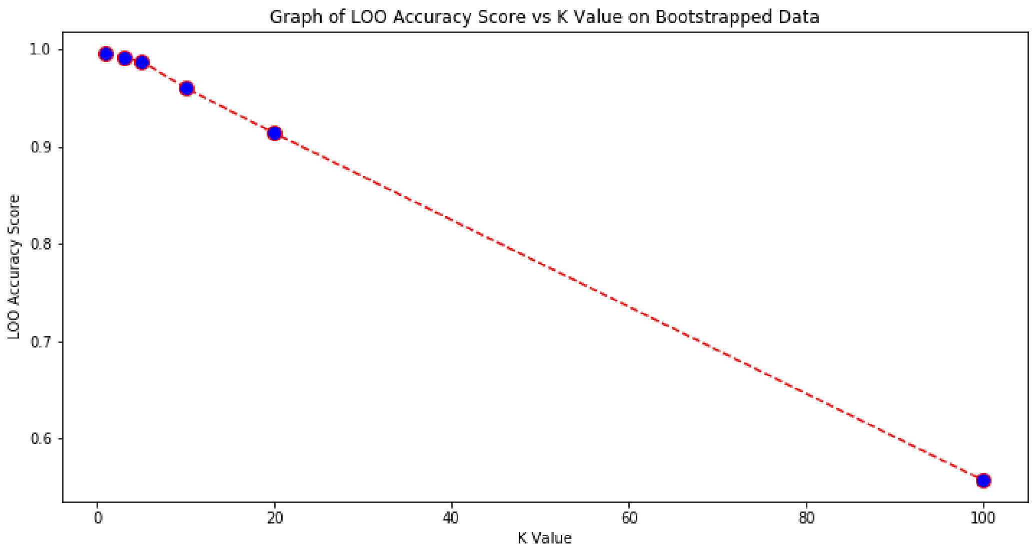
Detailed result sheet is attached for remaining values of L

PLOTS: Few examples are shown here. All plots are attached.

Random Projections: L=500, Euclidean Dist



Random Projections: L=2000, Euclidean Dist



INFERENCE/ANALYSIS:

- **The task here required to reduce the image dimensionality from 4096 to various L values like 500,1000,1500,2000 and 2500.**
- Random projection was implemented using Sparse Random matrix which is more memory efficient and optimized than Gaussian Random matrix.
- The result table shows very clearly that for K<=20, the accuracies and LOO accuracy is high even for reduced dimensions and the result can be obtained.
- Reducing dimensionality helps in optimization.

RESOURCES USED FOR THE ASSIGNMENT:

• **Environment:** Anaconda, Jupyter notebook
• **Software :** Python **Python libraries/modules:** Pandas, Numpy, SkLearn etc