Big Data Analytics Programming
Assignment 4: *Product Quantization for k-Nearest Neighbor Search*

Feiyang Tang (r0728168)
feiyang.tang@student.kuleuven.be

1 Performance comparison

	Memory	Prediction Time	Accuracy
Java (PQkNN)	239.5MB	47.95s (+2724.48s compressing)	94.41%
Python (sklearn)	1.4GB	734.29s (all prediction)	94.48%

Here we compare performance of our PQkNN written in Java with traditional implementation in Scikit-learning (Python). The performance metrics we used are memory, time used and accuracy.

We noticed that our PQkNN implementation widely saved time and memory use - computational costs and still kept an excellent accuracy.

2 Discussion

Noted the detailed explanation of code structure is included as comments in `PQkNN.java` and `Main.java`.

Compared with the standard k-NN algorithm, the compression acceleration algorithm has additional preprocessing time, but reduces the size of the data that needs to be saved. When the actual prediction is made, the calculation amount is hugely reduced, and the prediction efficiency is much higher.

Therefore, when comparing the compression algorithm with the standard algorithm, it is more appropriate to compare the time on **prediction** separately. The Java code is able to output the preprocessing time and actual prediction time separately while the outputted time of the python code is the entire time spent on making predictions.

We only compared the accuracy because this is a balanced classification scene, each sample from 0-9 accounts for about the same proportion, the following is the number of each sample in the training data.

Sample	Sample size
0	5923
1	6742
2	5958
3	6131
4	5842
5	5421
6	5918
7	6265
8	5851
9	5949

2.1 Problems with existing algorithm

When performing K-Means clustering, the existing algorithm does NOT set an **upper limit** on the number of iterations, it continues to iterate till the centre point does not move position, the actual number of iterations will be uncertain and produce contingency, sometimes clustering may need to run for a long time.