HW1: K-Nearest Neighbor Classifiers Report

Hsin-Ping Hsu (G01167652)

* Name registered on miner website: dmjp
* Rank and accuracy score: 10/ 0.86
* Approach:

1. Read training document: read document from training folder in sorted order then put all reviews and corresponding classes (truthful: 0 or deceptive: 1) into a “trainReviews” list, e.g. [[“we stayed at the chicago Hilton….\n”, 1], [“we booked this hotel using….\n”, 0]].

Function: training\_c(trainingPath,folder), training(trainingPath)

1. Read testing document: read document from testing folder in sorted order then put all reviews into a “testReviews” list, and initialize their classes as -1, e.g. [[“a recent stay at the james hotel-chicago…”, -1]].

Function: testing(testingPath)

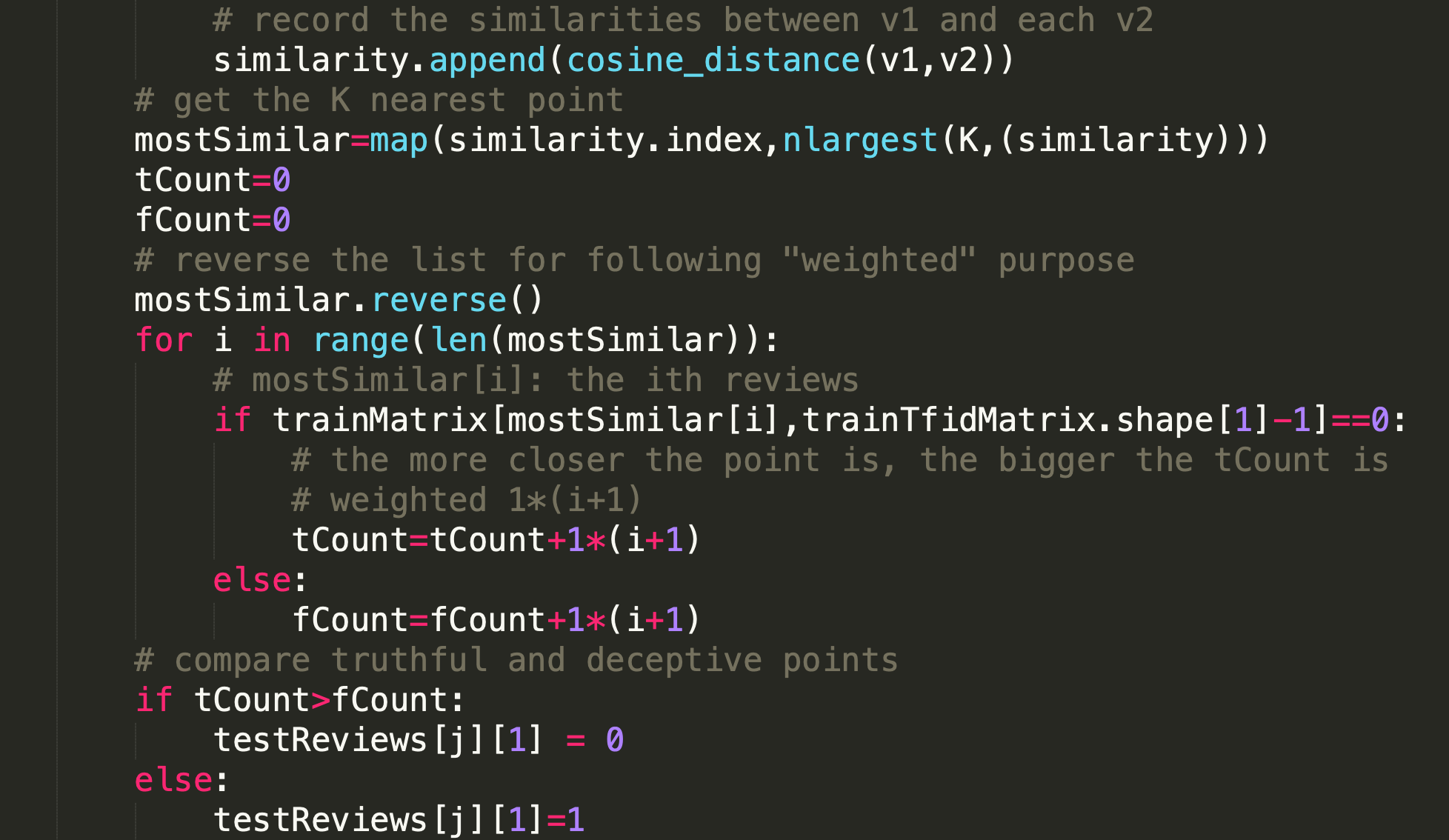
1. Convert training document to a matrix of TF-IDF features by using [sklearn.feature\_extraction.text.TfidfVectorizer](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html). To improve the accuracy, I set the parameter: max\_features = 500 to only consider top 500 features ordered by term frequency across the corpus; furthermore, stop words can also be removed by setting the parameter stop\_words=’english’ for TfidfVectorizer(). The fit\_transform(trainReviews) can learn vocabulary and idf from training document and return a term-document matrix. And the transform(testReviews) allows to transform the testing document to term-document matrix that was generated previously.

Function: tfidf()

1. Cosine similarity: evaluate the similarity between each testing document and training document by computing the cosine distance of vectors.

Function: consine\_distance(v1, v2)

1. knn\_Classifier: I use “nlargest” function to choose k nearest points to determine the unknown point by majority vote. To improve the accuracy, I consider the “weight” between each point. The more similar point has a higher weight.



Function: knn\_Classifier(K)

1. Cross validation: use cross validation to find the best K. Construct the classifier with every folder except Folder i. Use the Folder i as the testing document to compute the accuracy (the number of errors) of the classifier. (Repeatedly from i=1 to 10)

Function: cross\_validation(), get\_accuracy()

* Determine K:

According to [1, 2], a larger k suppresses the effect of noise, but makes the classification boundaries less distinct, and a higher value of k has lesser chance of error. Furthermore, the odd value of k is selected to avoid the confusion between two classes of data

The total number of errors and accuracies of each K are shown as Figure 1. When k = 413, it has a relatively higher accuracy.

Figure 1

|  |  |  |
| --- | --- | --- |
| K | # errors | Avg\_accuracy |
| 33 | 309 | 0.785608238 |
| : | : | : |
| 93 | 281 | 0.805019157 |
| : | : | : |
| 293 | 261 | 0.818908046 |
| : | : | : |
| 413 | 258 | 0.820996169 |

* Conclusion:

The k nearest neighbor classification was introduced in this assignment, which provided me a better view and concept on classification and data mining. I have learned how to deal with text document and create tf-idf; in addition, the cross validation allows me to test my algorithms and tune the parameter, which is very helpful.

* References

1. <https://cs.gmu.edu/~dbarbara/CS584/CS584L1.pdf>
2. <https://scikit-learn.org/stable/modules/neighbors.html>
3. <https://www.youtube.com/watch?v=4HKqjENq9OU&feature=youtu.be>
4. <https://www.analyticsvidhya.com/blog/2018/02/the-different-methods-deal-text-data-predictive-python/>