## HW1: K-Nearest Neighbor Classification Report

Date: 09/15/2020

Binhui Xu(G01172196)

Username on miner2: WhitneyXu

LeadBoard Score: 0.69

# Approach:

1. **Reading train\_data.txt,test\_data.txt and doing some text normalization**: read train\_data.txt, test\_data.txt files and put all reviews and corresponding classes(1,-1) if have into “train1”, “test1” list respectively, and make all letters in lowercase, remove all punctuations, and filter all specific symbols. e.g.[[1,“ One of the other reviewers has mentioned that after watching just 1 Oz episode you'll be hooked….\n”], [-1,” Basically there's a family where a little boy (Jake) thinks there's a zombie in his closet & his parents are fighting all the time…\n”]].

Function: processTrain(trainPath), processTest(testPath)

1. Convert training data 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). I set the parameters: max\_features=2000 to build a vocabulary build a vocabulary that only consider the top 2000 max\_features ordered by term frequency across the corpus. Stop\_words=’english’ to remove all stop words. The fit\_transform(train1), fit\_transform(test1) Learn vocabulary and idf, return document-term matrix trTfidf, teTfidf.

Function: tfidf()

1. **osine similarity:** measuring the similarity between each term in teating document and training document by computing the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction.

Function: cosine\_distance(v1,v2)

1. **K-NN classifier:** I used nlargest(k, similarity) to return the first k rows with the largest values in columns of similarity, in descending order. To improve the accuracy, use “weight” to determine the more similar point, which have higher weight.

Function: knn(k)

1. **Cross\_validation:** use cross validation to find the best k. I used the sklearn.model\_selection.KFold to split data in train(tr\_x,tr\_y)/test(te\_x,te\_y) sets. Split dataset into 5 consecutive folds. Each fold is then used once as a validation while the k - 1 remaining folds form the training set. And compute the accuracy of the classifier.

Function: cross\_validation(), get\_acc(x,y)

## Find K:

A precise k value is important in the process of K-NN classification. If k is too small, then the nearest neighbor classifier may be susceptible to overfitting due to noise, i.e., mislabeled examples in the training data. On the other hand, if k is too large, the nearest neighbor classifier may misclassify the test instance because its list of nearest neighbors includes training examples that are located far away from its neighborhood[1]. In my program, when k=423, it has a relatively high accuracy.

## Conclusion:

This assignment gives me a better understand on k nearest neighbor classification and data mining and python programing. And I also obtained a complete concept on a data mining process. Although this time my program’s accuracy and running time did not achieve the standard status, I have learned how to preprocess the raw text dataset and build ifidf. Besides, I have learned one of the model selection method cross validation, which allows me to improve my algorithm and tune the parameter. I believe all of these could help me in the feature data mining study.

## Reference:

1. [Pang-Ning Tan](http://www.cse.msu.edu/~ptan/), [Michael Steinbach](https://www-users.cs.umn.edu/~stei0062/) , [Anuj Karpatne](https://www-users.cs.umn.edu/~karpa009/), [Vipin Kumar](https://www-users.cs.umn.edu/~kumar001/), *Introduction of Data Mining,* 2019
2. <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text>. TfidfVectorizer. html
3. <https://www.sciencedirect.com/topics/computer-science/cosine-similarity>