1. Introduction

The main goal of this project is to classify short text.

The technique I use here is GloVe (Global Vectors for Word Representation), which is a NLP technique developed at Stanford University. GloVe is an unsupervised learning algorithm for obtaining vector representations for words, which can be downloaded freely on the official website. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, basically relies on conditional probability. The pre-trained word vectors has different versions (capacity of words) and dimension of the word vectors (50, 100, 200, etc.). The aim is to propose an algorithm to process the short texts, so I only adpot the smallest corpus and the lowest dimension (50) to train my dataset. In order to improve the accuracy of classification, larger corpus and higher dimension of word vectors should be utilised, at the cost of spending much longer time.

2. Dataset

The training set, "HS_CODE_Raw.xlsx", has 14281 rows and two columns. The first column is the "code", the real classification of a certain product, and the second column shows the basic description of corresponding "code". The code consists of 4, 8 or 10 digits (in which 6 digits are used as testing set). 4 digits class is the most generous classfication and can be divided into multiple sub-classes, which at most can be distinguished in 10 digits.

The testing set, "effective.xlsx", which cannot be post for some confidential reasons, consists of codes only in 6 digits and their descriptions. Actually you can make up all kinds of descriptions you want and the algorithm will classify these words into a trained class.

3. Algorithm

3.1 Text Processing

Firstly, we need to transfer the original dataset in `.xlsx` and the corpus in `.txt` into readable python form. We create every possible 6 digits code and their description by merging upper-level information (4 digits) and lower-level information (8 and 10 digits). The whole process basically covers all skills in text processing. Mainly the first part of the code is designed to quantify the words into matrices by means of NLP so that we can run machine learning algorithm.

3.2 Kmeans

After obtaining the words in decent data matrices form, we run Kmeans algorithm to cluster the descriptions into K groups, each group has a center location and a percentage of words that has been classfied into, we denote the ratio of word numbers in each cluster as weights. Technically, instead of only using Kmeans, we can run multiple machine learning algorithms and ensemble them in order to acquire the best result.

Then, when we have a new description, we also vectorize the words by means of GloVe, calculate weighted sum of euclidean distance on K center locations as a measurement of proximity, figure out the minimal distance and choose the corresponding cluster as our final classfication.