情報工学3　最終課題

氏名　山村武史
クラス　学際科学科
学生証番号 08-142025

○プログラムリスト

```
# -*- coding: utf-8 -*-

import csv
import codecs
import numpy as np
import MeCab

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans, MiniBatchKMeans
from sklearn.decomposition import TruncatedSVD
from sklearn.preprocessing import Normalizer

FILENAME = 'lessons_view.csv'
NUM_CLUSTERS = 10
LSA_DIM = 500
MAX_DF = 0.8
MAX_FEATURES = 100000
MINIBATCH = True

def get_tweets_from_csv(filename):
    ret = csv.reader(open(filename))
    print ret
    # tweets = []
    # for r in ret:
    #     if len(r[2]) != 0:
    #         tweets.append(r[2].decode('utf-8'))
    tweets = [r[2].decode('utf-8') for r in ret]
    # print tweets

    for tweet in tweets[:]:
        if u'@' in tweet:
            tweets.remove(tweet)
        if len(tweet) <= 3:
            tweets.remove(tweet)
    return tweets


def analyzer(text):
    ret = []
    tagger = MeCab.Tagger('-Ochasen')
    node = tagger.parseToNode(text.encode('utf-8'))
    node = node.next
    while node.next:
        ret.append(node.feature.split(',')[-3].decode('utf-8'))
        node = node.next

    return ret


def main(filename):
    # load tweets
    tweets = get_tweets_from_csv(filename)
    # print tweets

    # feature extraction
    vectorizer = TfidfVectorizer(analyzer=analyzer, max_df=MAX_DF)
```

```
    vectorizer.max_features = MAX_FEATURES
    X = vectorizer.fit_transform(tweets)
    # dimensionality reduction by LSA
    lsa = TruncatedSVD(LSA_DIM)
    X = lsa.fit_transform(X)
    X = Normalizer(copy=False).fit_transform(X)

    # clustering by KMeans
    if MINIBATCH:
        km = MiniBatchKMeans(n_clusters=NUM_CLUSTERS, init='k-means++', batch_size=1000, n_init=10, max_no_improvement=10, verbose=True)
    else:
        km = KMeans(n_clusters=NUM_CLUSTERS, init='k-means++', n_init=1, verbose=True)
    km.fit(X)
    labels = km.labels_

    transformed = km.transform(X)
    dists = np.zeros(labels.shape)
    for i in range(len(labels)):
        dists[i] = transformed[i, labels[i]]

    # sort by distance
    clusters = []
    for i in range(NUM_CLUSTERS):
        cluster = []
        ii = np.where(labels==i)[0]
        dd = dists[ii]
        di = np.vstack([dd,ii]).transpose().tolist()
        di.sort()
        for d, j in di:
            cluster.append(tweets[int(j)])
        clusters.append(cluster)

    return clusters


if __name__ == '__main__':
    clusters = main(FILENAME)
    f = codecs.open('%s.txt' % FILENAME, 'w', 'utf-8')
    for i,tweets in enumerate(clusters):
        for tweet in tweets:
            f.write('%d: %s\n' % (i, tweet.replace('/n', '')))
    f.close()
```

○実行コマンド

```
$python tweets_clustering.py
```

○説明

実行結果ファイルはtweets.csv.txtです。 このプログラムでは自分のtwitter上でのtweet約３０００件をクラスタリングしてみました。まず自分のtweet情報のcsvファイルを読み込みこれをtf-idf法を適用して書くツイートをBag of words表現に変換してます。
またここでのMAX_FEATURES = 100000　はあまりにも出現頻度の高い単語は無視するようこの数より多く出る単語は無視しています。
次にbag of wordsを潜在意味解析で次元削減し、各文書をより本質的意味を考慮した洗剤意味空間へ投影してます。後は洗剤意味空間に位置するツイートを近い物同士でk-means法を使い１０種類にクラスタリングしています。
実際にクラスタリングしたものを見てみると

0: それにしても本当クラコン楽しかった、というかカラオケオール楽しかったな(^q^)
0: 久しぶりのクラコン楽しかった！笑
0: 今日楽しかったわ＼(^o^)／満足〜！
0: 今日はいろんな人と話せて楽しかったー!
0: 五月祭超楽しかった＼(^o^)／

上のように単純に楽しいというワードでまとめられたクラスから、

1: バス代はらえない(((・・;)汗
1: あれっ誰もいない(・ω・)
1: なんとか渋谷駅で電車に乗った！そろそろ肝臓がしにそう。・゜・(ノД`)・゜・。
1: コタツで寝てしまった体痛い⊂((・x・))⊃

このように共通するワードはそれほどないにもかかわらず、焦りや後悔といったネガティブなツイートでうまくクラスタリングされている場合もあります。