Exercises04-Revised-Solutions

February 21, 2015

```
In [1]: import os
        import sys
        import glob
        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        %matplotlib inline
        %precision 4
        plt.style.use('ggplot')
```

```
In [2]: import scipy.linalg as la
        import scipy.stats as st
```

```
In [3]: import cPickle
```

```
In [4]: np.set_printoptions(formatter={'float': '{: 0.3f}'.format})
```

Latent Semantic Analysis (LSA) is a method for reducing the dimnesionality of documents treated as a bag of words. It is used for document classification, clustering and retrieval. For example, LSA can be used to search for prior art given a new patent application. In this homework, we will implement a small library for simple latent semantic analysis as a practical example of the application of SVD. The ideas are very similar to PCA.

We will implement a toy example of LSA to get familiar with the ideas. If you want to use LSA or similar methods for statiscal language analyis, the most efficient Python library is probably gensim - this also provides an online algorithm - i.e. the training information can be continuously updated. Other useful functions for processing natural language can be found in the Natural Lnaguage Toolkit.

Note: The SVD from scipy.linalg performs a full decomposition, which is inefficient since we only need to decompose until we get the first k singluar values. If the SVD from `scipy.linalg` is too slow, please use the `sparsesvd` function from the sparsesvd package to perform SVD instead. You can install in the usual way with

```
!pip install sparsesvd
```

Then import the following

```
from sparsesvd import sparsesvd
from scipy.sparse import csc_matrix
```

and use as follows

```
sparsesvd(csc_matrix(M), k=10)
```

Exercise 1 (10 points). Calculating pairwise distance matrices.

Suppose we want to construct a distance matrix between the rows of a matrix. For example, given the matrix

```
M = np.array([[1,2,3],[4,5,6]])
```

the distance matrix using Euclidean distance as the measure would be

```
[[ 0.000  1.414  2.828]
 [ 1.414  0.000  1.414]
 [ 2.828  1.414  0.000]]
```

if M was a collection of column vectors.

Write a function to calculate the pairwise-distance matrix given the matrix M and some arbitrary distance function. Your functions should have the following signature:

```
def func_name(M, distance_func):
    pass
```

0. Write a distance function for the Euclidean, squared Euclidean and cosine measures.
1. Write the function using looping for M as a collection of row vectors.
2. Write the function using looping for M as a collection of column vectors.
3. Wrtie the function using broadcasting for M as a colleciton of row vectors.
4. Write the function using broadcasting for M as a colleciton of column vectors.

For 3 and 4, try to avoid using transposition (but if you get stuck, there will be no penalty for using transpoition). Check that all four functions give the same result when applied to the given matrix M.

```
In [5]: # Questions 1.1 to 1.5

        def squared_euclidean_norm(u, axis=-1):
            return (u**2).sum(axis)

        def euclidean_norm(u, axis=-1):
            return np.sqrt(squared_euclidean_norm(u, axis))

        def squared_euclidean_dist(u, v, axis=-1):
            """Returns squared Euclidean distance between two vectors."""
            return squared_euclidean_norm(u-v, axis)

        def euclidean_dist(u, v, axis=-1):
            """Return Euclidean distacne between two vectors."""
            return np.sqrt(squared_euclidean_dist(u, v, axis))

        def cosine_dist(u, v, axis=-1):
            """Returns cosine of angle betwwen two vectors."""
            # return 1 - np.dot(u, v)/(la.norm(u)*la.norm(v))
            return 1 - (u * v).sum(axis)/(euclidean_norm(u, axis) * euclidean_norm(v, axis))

        def loop_row_pdist(M, f):
            """REturns pairwise-distance matrix assuming M consists of row vectors.."""
            nrows, ncols = M.shape
            return np.array([[f(M[u,:], M[v,:]) for u in range(nrows)]
                                                for v in range(nrows)])

        def loop_col_pdist(M, f):
            """REturns pairwise-distance matrix assuming M consists of column vectors.."""
            nrows, ncols = M.shape
            return np.array([[f(M[:,u], M[:,v]) for u in range(ncols)]
```

```
                                                for v in range(ncols)])

        def broadcast_row_pdist(M, f):
            """REturns pairwise-distance matrix assuming M consists of row vectors.."""
            return f(M[None,:,:], M[:,None,:])

        def broadcast_col_pdist(M, f):
            """REturns pairwise-distance matrix assuming M consists of column vectors.."""
            return f(M[:,None,:], M[:,:,None], axis=0)
```

In [8]:
```
        # Q1 checking reuslts

        M = np.array([[1,2,3],[4,5,6]])

        # dist = euclidean_dist
        for dist in (cosine_dist, euclidean_dist, squared_euclidean_dist):
            print loop_row_pdist(M, dist), '\n'
            print broadcast_row_pdist(M, dist), '\n'
            print loop_col_pdist(M, dist), '\n'
            print broadcast_col_pdist(M, dist)
            print
```

```
[[ 0.000  0.025]
 [ 0.025  0.000]]

[[ 0.000  0.025]
 [ 0.025  0.000]]

[[ 0.000  0.009  0.024]
 [ 0.009 -0.000  0.003]
 [ 0.024  0.003  0.000]]

[[ 0.000  0.009  0.024]
 [ 0.009 -0.000  0.003]
 [ 0.024  0.003  0.000]]

[[ 0.000  5.196]
 [ 5.196  0.000]]

[[ 0.000  5.196]
 [ 5.196  0.000]]

[[ 0.000  1.414  2.828]
 [ 1.414  0.000  1.414]
 [ 2.828  1.414  0.000]]

[[ 0.000  1.414  2.828]
 [ 1.414  0.000  1.414]
 [ 2.828  1.414  0.000]]

[[ 0 27]
 [27  0]]

[[ 0 27]
 [27  0]]
```

```
[[0 2 8]
 [2 0 2]
 [8 2 0]]

[[0 2 8]
 [2 0 2]
 [8 2 0]]
```

Exercise 2 (10 points). Write 3 functions to calculate the term frequency (tf), the inverse document frequency (idf) and the product (tf-idf). Each function should take a single argument `docs`, which is a dictionary of (key=identifier, value=dcoument text) pairs, and return an appropriately sized array. Convert '-' to '' (space), remove punctuation, convert text to lowercase and split on whitespace to generate a collection of terms from the dcoument text.

- tf = the number of occurrences of term i in document j
- idf = $\log \frac{n}{1+\text{df}_i}$ where n is the total number of documents and df_i is the number of documents in which term i occurs.

Print the table of tf-idf values for the following document collection

```
s1 = "The quick brown fox"
s2 = "Brown fox jumps over the jumps jumps jumps"
s3 = "The the the lazy dog elephant."
s4 = "The the the the the dog peacock lion tiger elephant"

docs = {'s1': s1, 's2': s2, 's3': s3, 's4': s4}
```

Note: You can use either a numpy array or pandas dataframe to store the matrix. However, we suggest using a Pnadas dataframe since that will allow you to keep track of the row (term) and column (document) names in a single object. Of course, you could also maintain a numpy matrix, a list of terms, and a list of documents separately if you prefer.

```
In [9]: # The tf() function is optional - it can also be coded directly into tfs()

        # Questino 2.1
        def tf(doc):
            """Returns the number of times each term occurs in a dcoument.
            We preprocess the document to strip punctuation and convert to lowercase.
            Terms are found by splitting on whitespace."""
            from collections import Counter
            from string import punctuation

            terms = doc.lower().replace('-', ' ').translate(None, punctuation).split()
            return Counter(terms)

        def tfs(docs):
            """Create a term freqeuncy dataframe from a dictionary of documents."""
            from operator import add

            df = pd.DataFrame({k: tf(v) for k, v in docs.iteritems()}).fillna(0)
            return df

        # Question 2.2
        def idf(docs):
```

```
    """Find inverse document frequecny series from a dictionry of doucmnets."""
    term_freq = tfs(docs)
    num_docs = len(docs)
    doc_freq = (term_freq > 0).sum(axis=1)
    return np.log(num_docs/(1 + doc_freq))

# Question 2.3
def tf_idf(docs):
    """Return the product of the term-frequency and inverse document freqeucny."""
    return tfs(docs).mul(idf(docs), axis=0)
```

In [10]:
```
# Question 2.4

s1 = "The quick brown fox"
s2 = "Brown fox jumps over the jumps jumps jumps"
s3 = "The the the lazy dog elephant."
s4 = "The the the the the dog peacock lion tiger elephant"

docs = {'s1': s1, 's2': s2, 's3': s3, 's4': s4}

tf_idf(docs)
```

Out[10]:

	s1	s2	s3	s4
brown	0.287682	0.287682	0.000000	0.000000
dog	0.000000	0.000000	0.287682	0.287682
elephant	0.000000	0.000000	0.287682	0.287682
fox	0.287682	0.287682	0.000000	0.000000
jumps	0.000000	2.772589	0.000000	0.000000
lazy	0.000000	0.000000	0.693147	0.000000
lion	0.000000	0.000000	0.000000	0.693147
over	0.000000	0.693147	0.000000	0.000000
peacock	0.000000	0.000000	0.000000	0.693147
quick	0.693147	0.000000	0.000000	0.000000
the	-0.223144	-0.223144	-0.669431	-1.115718
tiger	0.000000	0.000000	0.000000	0.693147

Exercise 3 (10 points).

1. Write a function that takes a matrix M and an integer k as arguments, and reconstructs a reduced matrix using only the k largest singular values. Use the `scipy.linagl.svd` function to perform the decomposition. This is the least squares approximation to the matrix M in k dimensions.

2. Apply the function you just wrote to the following term-frequency matrix for a set of 9 documents using $k = 2$ and print the reconstructed matrix M'.

```
M = np.array([[1, 0, 0, 1, 0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0, 0, 0, 0, 0],
       [1, 1, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 1, 0, 1, 0, 0, 0, 0],
       [0, 1, 1, 2, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 1, 0, 0, 0, 0],
       [0, 1, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 1, 1, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 0, 0, 0, 1],
       [0, 0, 0, 0, 0, 1, 1, 1, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 1],
       [0, 0, 0, 0, 0, 0, 0, 1, 1]])
```

3. Calculate the pairwise correlation matrix for the original matrix M and the reconstructed matrix using $k = 2$ singular values (you may use scipy.stats.spearmanr to do the calculations). Consider the fist 5 sets of documents as one group $G1$ and the last 4 as another group $G2$ (i.e. first 5 and last 4 columns). What is the average within group correlation for $G1$, $G2$ and the average cross-group correlation for G1-G2 using either M or M'. (Do not include self-correlation in the within-group calculations.).

In [12]:
```
# Question 3.1

def svd_projection(M, k):
    """Returns the matrix M reconstructed using only k singluar values"""
    U, s, V = la.svd(M, full_matrices=False)
    s[k:] = 0
    M_ = U.dot(np.diag(s).dot(V))

    try:
        return pd.DataFrame(M_, index=M.index, columns=M.columns)
    except AttributeError:
        return M_
```

In [13]:
```
# Qeustion 3.2

M = np.array([[1, 0, 0, 1, 0, 0, 0, 0, 0],
    [1, 0, 1, 0, 0, 0, 0, 0, 0],
    [1, 1, 0, 0, 0, 0, 0, 0, 0],
    [0, 1, 1, 0, 1, 0, 0, 0, 0],
    [0, 1, 1, 2, 0, 0, 0, 0, 0],
    [0, 1, 0, 0, 1, 0, 0, 0, 0],
    [0, 1, 0, 0, 1, 0, 0, 0, 0],
    [0, 0, 1, 1, 0, 0, 0, 0, 0],
    [0, 1, 0, 0, 0, 0, 0, 0, 1],
    [0, 0, 0, 0, 0, 1, 1, 1, 0],
    [0, 0, 0, 0, 0, 0, 1, 1, 1],
    [0, 0, 0, 0, 0, 0, 0, 1, 1]])

Md = svd_projection(M, 2)
Md
```

Out[13]:
```
array([[ 0.162,  0.400,  0.379,  0.468,  0.176, -0.053, -0.115, -0.159,
        -0.092],
       [ 0.141,  0.370,  0.329,  0.400,  0.165, -0.033, -0.071, -0.097,
        -0.043],
       [ 0.152,  0.505,  0.358,  0.410,  0.236,  0.024,  0.060,  0.087,
         0.124],
       [ 0.258,  0.841,  0.606,  0.697,  0.392,  0.033,  0.083,  0.122,
         0.187],
       [ 0.449,  1.234,  1.051,  1.266,  0.556, -0.074, -0.155, -0.210,
        -0.049],
       [ 0.160,  0.582,  0.375,  0.417,  0.277,  0.056,  0.132,  0.189,
         0.217],
       [ 0.160,  0.582,  0.375,  0.417,  0.277,  0.056,  0.132,  0.189,
         0.217],
       [ 0.218,  0.550,  0.511,  0.628,  0.243, -0.065, -0.143, -0.197,
        -0.108],
       [ 0.097,  0.532,  0.230,  0.212,  0.267,  0.137,  0.315,  0.444,
         0.425],
```

```
       [-0.061,  0.232, -0.139, -0.266,  0.145,  0.240,  0.546,  0.767,
         0.664],
       [-0.065,  0.335, -0.146, -0.301,  0.203,  0.306,  0.695,  0.977,
         0.849],
       [-0.043,  0.254, -0.097, -0.208,  0.152,  0.221,  0.503,  0.707,
         0.616]])
```

In [16]:
```
# Question 3.3

# Results for full-rank matrix (not graded - just here for comparison)
rho, pval = st.spearmanr(M)
np.mean(rho[:5, :5][np.tril_indices_from(rho[:5, :5], 1)]), \
np.mean(rho[5:, 5:][np.tril_indices_from(rho[5:, 5:], 1)]), \
rho[5:, :5].mean()
```

Out[16]:
```
(0.2643, 0.6627, -0.3076)
```

In [15]:
```
# Results after LSA (graded)
# G1/G1, G2/G2 and G1/G2 average correlation

rho, pval = st.spearmanr(Md)
np.mean(rho[:5, :5][np.tril_indices_from(rho[:5, :5], 1)]), \
np.mean(rho[5:, 5:][np.tril_indices_from(rho[5:, 5:], 1)]), \
rho[5:, :5].mean()
```

Out[15]:
```
(0.8988, 0.9930, -0.6780)
```

Exercise 4 (20 points). Clustering with LSA

1. Begin by loading a pubmed database of selected article titles using 'cPickle'. With the following: `import cPickle docs = cPickle.load(open('pubmed.pic'))`

   Create a tf-idf matrix for every term that appears at least once in any of the documents. What is the shape of the tf-idf matrix?

2. Perform SVD on the tf-idf matrix to obtain $U\Sigma V^T$ (often written as $T\Sigma D^T$ in this context with T representing the terms and D representing the documents). If we set all but the top k singular values to 0, the reconstructed matrix is essentially $U_k\Sigma_k V_k^T$, where U_k is $m \times k$, Σ_k is $k \times k$ and V_k^T is $k \times n$. Terms in this reduced space are represented by $U_k\Sigma_k$ and documents by $\Sigma_k V_k^T$. Reconstruct the matrix using the first $k = 10$ singular values.

3. Use agglomerative hierachical clustering with complete linkage to plot a dendrogram and comment on the likely number of document clusters with $k = 100$. Use the dendrogram function from SciPy.

4. Determine how similar each of the original documents is to the new document `mystery.txt`. Since $A = U\Sigma V^T$, we also have $V = A^T U S^{-1}$ using orthogonality and the rule for transposing matrix products. This suggests that in order to map the new document to the same concept space, first find the tf-idf vector v for the new document - this must contain all (and only) the terms present in the existing tf-idx matrix. Then the query vector q is given by $v^T U_k \Sigma_k^{-1}$. Find the 10 documents most similar to the new document and the 10 most dissimilar.

5. Many documents often have some boilerplate material such as organization information, Copyright, etc. at the front or back of the document. Does it matter that the front and back matter of each document is essentially identical for either LSA-based clustering (part 3) or information retrieval (part 4)? Why or why not?

```
In [58]: # Quesiton 4.1

         import cPickle

         docs = cPickle.load(open('pubmed.pic'))
         df = tf_idf(docs)
         df.shape

Out[58]: (6488, 178)

In [109]: # Question 4.2

          k = 10
          T, s, D = la.svd(df)

          print T.shape, s.shape, D.shape, '\n'

          df_10 = T[:,:k].dot(np.diag(s[:k]).dot(D[:k,:]))
          assert(df.shape == df_10.shape)
          print df_10

(6488, 6488) (178,) (178, 178)

[[ 0.044 -0.051  0.177 ...,  0.023  0.127 -0.210]
 [ 0.003  0.043  0.008 ...,  0.035  0.029  0.092]
 [ 0.002 -0.076  0.066 ..., -0.041 -0.009 -0.180]
 ...,
 [ 0.008  0.009  0.016 ...,  0.006  0.026  0.012]
 [ 0.031  0.122  0.070 ...,  0.093  0.128  0.198]
 [ 0.008  0.016  0.017 ...,  0.011  0.031  0.026]]

In [113]: # Question 4.2 (alternative solution 1 setting unwanted singluar values to zero)

          T, s, D = la.svd(df, full_matrices=False)
          print T.shape, s.shape, D.shape, '\n'

          s[10:] = 0
          df_10 = T.dot(np.diag(s).dot(D))
          assert(df.shape == df_10.shape)
          print df_10

(6488, 178) (178,) (178, 178)

[[ 0.044 -0.051  0.177 ...,  0.023  0.127 -0.210]
 [ 0.003  0.043  0.008 ...,  0.035  0.029  0.092]
 [ 0.002 -0.076  0.066 ..., -0.041 -0.009 -0.180]
 ...,
 [ 0.008  0.009  0.016 ...,  0.006  0.026  0.012]
 [ 0.031  0.122  0.070 ...,  0.093  0.128  0.198]
 [ 0.008  0.016  0.017 ...,  0.011  0.031  0.026]]

In [103]: # Question 4.2 (alternative solution 2 using sparsesvd)

          from scipy.sparse import csc_matrix
          from sparsesvd import sparsesvd
```

```
k = 10
T, s, D = sparsesvd(csc_matrix(df), k=k)

print T.shape, s.shape, D.shape, '\n'
df_10 = T.T.dot(np.diag(s).dot(D))
assert(df.shape == df_10.shape)
print df_10
```

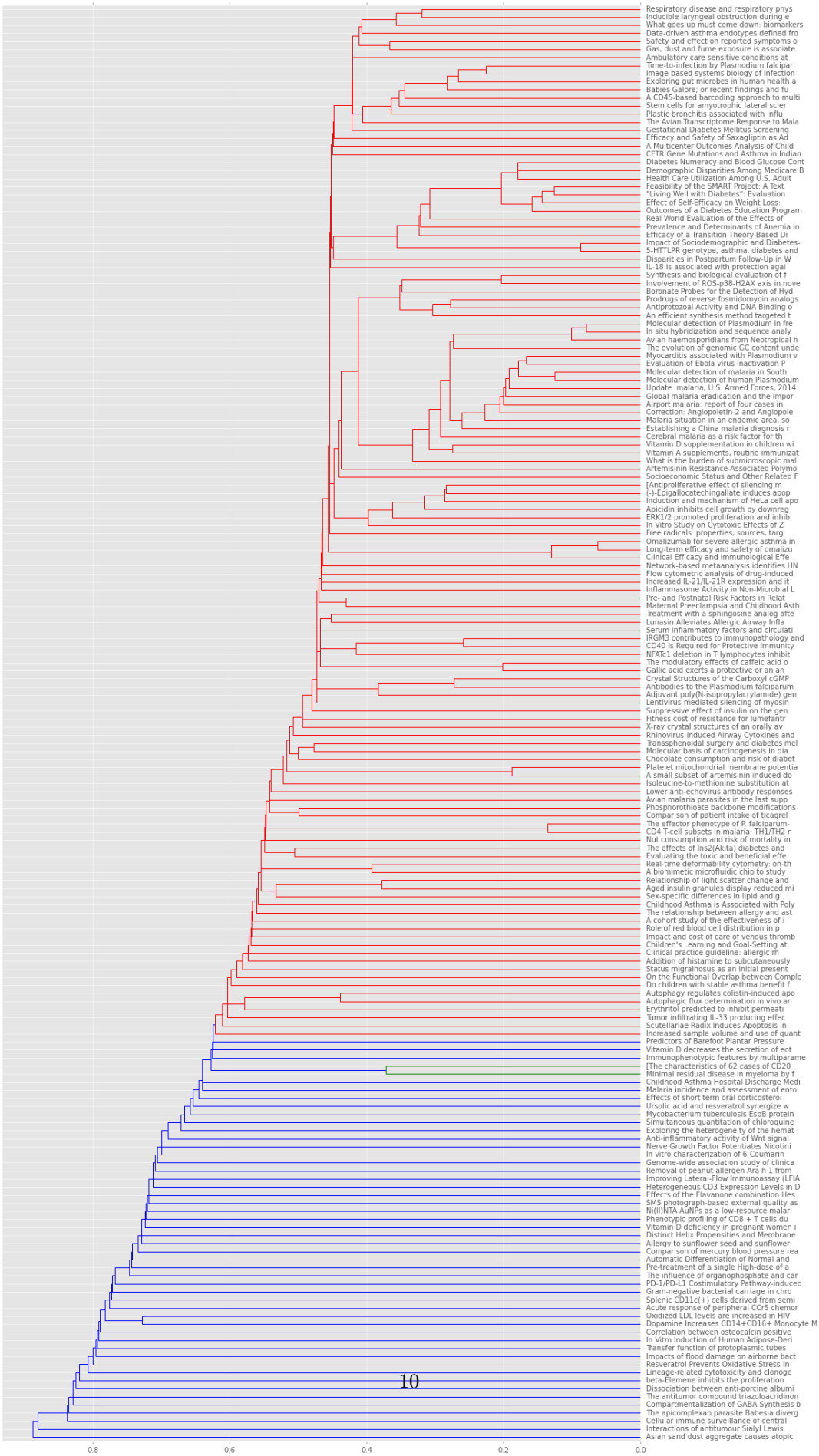
```
(10, 6488) (10,) (10, 178)

[[ 0.044 -0.051  0.177 ...,  0.023  0.127 -0.210]
 [ 0.003  0.043  0.008 ...,  0.035  0.029  0.092]
 [ 0.002 -0.076  0.066 ..., -0.041 -0.009 -0.180]
 ...,
 [ 0.008  0.009  0.016 ...,  0.006  0.026  0.012]
 [ 0.031  0.122  0.070 ...,  0.093  0.128  0.198]
 [ 0.008  0.016  0.017 ...,  0.011  0.031  0.026]]
```

In [52]:
```
# Question 4.3

from scipy.cluster.hierarchy import linkage, dendrogram
from scipy.spatial.distance import pdist, squareform

plt.figure(figsize=(16,36))
T, s, D = sparsesvd(csc_matrix(df), k=100)
x = np.diag(s).dot(D).T
data_dist = pdist(x, metric='cosine') # computing the distance
data_link = linkage(data_dist) # computing the linkage
labels = [c[:40] for c in df.columns[:]]
dendrogram(data_link, orientation='right', labels=labels);
```


Respiratory disease and respiratory phys
Inducible laryngeal obstruction during e
What goes up must come down: biomarkers
Data-driven asthma endotypes defined fro
Safety and effect on reported symptoms o
Gas, dust and fume exposure is associate
Ambulatory care sensitive conditions at
Time-to-infection by Plasmodium falcipar
Image-based systems biology of infection
Exploring gut microbes in human health a
Babies Galore, or recent findings and fu
A CD45-based barcoding approach to multi
Stem cells for amyotrophic lateral scler
Plastic bronchitis associated with influ
The Avian Transcriptome Response to Mala
Gestational Diabetes Mellitus Screening
Efficacy and Safety of Saxagliptin as Ad
A Multicenter Outcomes Analysis of Child
CFTR Gene Mutations and Asthma in Indian
Diabetes Numeracy and Blood Glucose Cont
Demographic Disparities Among Medicare B
Health Care Utilization Among U.S. Adult
Feasibility of the SMART Project: A Text
"Living Well with Diabetes": Evaluation
Effect of Self-Efficacy on Weight Loss:
Outcomes of a Diabetes Education Program
Real-World Evaluation of the Effects of
Prevalence and Determinants of Anemia in
Efficacy of a Transition Theory-Based Di
Impact of Sociodemographic and Diabetes-
5-HTTLPR genotype, asthma, diabetes and
Disparities in Postpartum Follow-Up in W
IL-18 is associated with protection agai
Synthesis and biological evaluation of f
Involvement of ROS-p38-H2AX axis in nove
Boronate Probes for the Detection of Hyd
Prodrugs of reverse fosmidomycin analogs
Antiprotozoal Activity and DNA Binding o
An efficient synthesis method targeted t
Molecular detection of Plasmodium in fre
In situ hybridization and sequence analy
Avian haemosporidians from Neotropical h
The evolution of genomic GC content unde
Myocarditis associated with Plasmodium v
Evaluation of Ebola virus Inactivation P
Molecular detection of malaria in South
Molecular detection of human Plasmodium
Update: malaria, U.S. Armed Forces, 2014
Global malaria eradication and the impor
Airport malaria: report of four cases in
Correction: Angiopoietin-2 and Angiopoie
Malaria situation in an endemic area, so
Establishing a China malaria diagnosis r
Cerebral malaria as a risk factor for th
Vitamin D supplementation in children wi
Vitamin A supplements, routine immunizat
What is the burden of submicroscopic mal
Artemisinin Resistance-Associated Polymo
Socioeconomic Status and Other Related F
[Antiproliferative effect of silencing m
(-)-Epigallocatechingallate induces apop
Induction and mechanism of HeLa cell apo
Apicidin inhibits cell growth by downreg
ERK1/2 promoted proliferation and inhibi
In Vitro Study on Cytotoxic Effects of Z
Free radicals: properties, sources, targ
Omalizumab for severe allergic asthma in
Long-term efficacy and safety of omalizu
Clinical Efficacy and Immunological Effe
Network-based metaanalysis identifies HN
Flow cytometric analysis of drug-induced
Increased IL-21/IL-21R expression and it
Inflammasome Activity in Non-Microbial L
Pre- and Postnatal Risk Factors in Relat
Maternal Preeclampsia and Childhood Asth
Treatment with a sphingosine analog afte
Lunasin Alleviates Allergic Airway Infla
Serum inflammatory factors and circulati
IRGM3 contributes to immunopathology and
CD40 is Required for Protective Immunity
NFATc1 deletion in T lymphocytes inhibit
The modulatory effects of caffeic acid o
Gallic acid exerts a protective or an an
Crystal Structures of the Carboxyl cGMP
Antibodies to the Plasmodium falciparum
Adjuvant poly(N-isopropylacrylamide) gen
Lentivirus-mediated silencing of myosin
Suppressive effect of insulin on the gen
Fitness cost of resistance for lumefantr
X-ray crystal structures of an orally av
Rhinovirus-induced Airway Cytokines and
Transsphenoidal surgery and diabetes mel
Molecular basis of carcinogenesis in dia
Chocolate consumption and risk of diabet
Platelet mitochondrial membrane potentia
A small subset of artemisinin induced do
Isoleucine-to-methionine substitution at
Lower anti-echovirus antibody responses
Avian malaria parasites in the last supp
Phosphorothioate backbone modifications
Comparison of patient intake of ticagrel
The effector phenotype of P. falciparum-
CD4 T-cell subsets in malaria: TH1/TH2 r
Nut consumption and risk of mortality in
The effects of Ins2(Akita) diabetes and
Evaluating the toxic and beneficial effe
Real-time deformability cytometry: on-th
A biomimetic microfluidic chip to study
Relationship of light scatter change and
Aged insulin granules display reduced mi
Sex-specific differences in lipid and gl
Childhood Asthma is Associated with Poly
The relationship between allergy and ast
A cohort study of the effectiveness of i
Role of red blood cell distribution in p
Impact and cost of care of venous thromb
Children's Learning and Goal-Setting at
Clinical practice guideline: allergic rh
Addition of histamine to subcutaneously
Status migrainosus as an initial present
On the Functional Overlap between Comple
Do children with stable asthma benefit f
Autophagy regulates colistin-induced apo
Autophagic flux determination in vivo an
Erythritol predicted to inhibit permeati
Tumor infiltrating IL-33 producing effec
Scutellariae Radix Induces Apoptosis in
Increased sample volume and use of quant
Predictors of Barefoot Plantar Pressure
Vitamin D decreases the secretion of eot
Immunophenotypic features by multiparame
[The characteristics of 62 cases of CD20
Minimal residual disease in myeloma by f
Childhood Asthma Hospital Discharge Medi
Malaria incidence and assessment of ento
Effects of short term oral corticosteroi
Ursolic acid and resveratrol synergize w
Mycobacterium tuberculosis EspB protein
Simultaneous quantitation of chloroquine
Exploring the heterogeneity of the hemat
Anti-inflammatory activity of Wnt signal
Nerve Growth Factor Potentiates Nicotini
In vitro characterization of 6-Coumarin
Genome-wide association study of clinica
Removal of peanut allergen Ara h 1 from
Improving Lateral-Flow Immunoassay (LFIA
Heterogeneous CD3 Expression Levels in D
Effects of the Flavanone combination Hes
SMS photograph-based external quality as
Ni(II)NTA AuNPs as a low-resource malari
Phenotypic profiling of CD8 + T cells du
Vitamin D deficiency in pregnant women i
Distinct Helix Propensities and Membrane
Allergy to sunflower seed and sunflower
Comparison of mercury blood pressure rea
Automatic Differentiation of Normal and
Pre-treatment of a single High-dose of a
The influence of organophosphate and car
PD-1/PD-L1 Costimulatory Pathway-induced
Gram-negative bacterial carriage in chro
Splenic CD11c(+) cells derived from semi
Acute response of peripheral CCr5 chemor
Oxidized LDL levels are increased in HIV
Dopamine Increases CD14+CD16+ Monocyte M
Correlation between osteocalcin positive
In Vitro Induction of Human Adipose-Deri
Transfer function of protoplasmic tubes
Impacts of flood damage on airborne bact
Resveratrol Prevents Oxidative Stress-In
Lineage-related cytotoxicity and clonoge
beta-Elemene inhibits the proliferation
Dissociation between anti-porcine albumi
The antitumor compound triazoloacridinon
Compartmentalization of GABA Synthesis b
The apicomplexan parasite Babesia diverg
Cellular immune surveillance of central
Interactions of antitumour Sialyl Lewis
Asian sand dust aggregate causes atopic
0.8
0.6
0.4
0.2
0.0

In [55]:
```
# Quesiton 4.4

        k = 10
        T, s, D = sparsesvd(csc_matrix(df), k=100)

        doc = {'mystery': open('mystery.txt').read()}
        terms = tf_idf(doc)
        query_terms = df.join(terms).fillna(0)['mystery']
        q = query_terms.T.dot(T.T.dot(np.diag(1.0/s)))

        ranked_docs = df.columns[np.argsort(cosine_dist(q, x))][::-1]
        print "Query article:",
        print ' '.join(line.strip() for line in doc['mystery'].splitlines()[:2])
        print
        print "Most similar"
        print '='*80
        for i, title in enumerate(ranked_docs[:10]):
            print '%03d' % i, title

        print
        print "Most dissimilar"
        print '='*80
        for i, title in enumerate(ranked_docs[-10:]):
            print '%03d' % (len(docs) - i), title
```

```
Query article: Intensive blood-glucose control with sulphonylureas or insulin compared with conventional

Most similar
================================================================================
000 Diabetes Numeracy and Blood Glucose Control: Association With Type of Diabetes and Source of Care.
001 Feasibility of the SMART Project: A Text Message Program for Adolescents With Type 1 Diabetes.
002 Health Care Utilization Among U.S. Adults With Diagnosed Diabetes, 2013.
003 Demographic Disparities Among Medicare Beneficiaries with Type 2 Diabetes Mellitus in 2011: Diabetes
004 Disparities in Postpartum Follow-Up in Women With Gestational Diabetes Mellitus.
005 Prevalence and Determinants of Anemia in Older People With Diabetes Attending an Outpatient Clinic:
006 Outcomes of a Diabetes Education Program for Registered Nurses Caring for Individuals With Diabetes
007 Gestational Diabetes Mellitus Screening Using the One-Step Versus Two-Step Method in a High-Risk Pra
008 Evaluating the toxic and beneficial effects of lichen extracts in normal and diabetic rats.
009 Efficacy and Safety of Saxagliptin as Add-On Therapy in Type 2 Diabetes.

Most dissimilar
================================================================================
178 Phenotypic profiling of CD8 + T cells during Plasmodium vivax blood-stage infection.
177 ERK1/2 promoted proliferation and inhibited apoptosis of human cervical cancer cells and regulated t
176 Avian haemosporidians from Neotropical highlands: Evidence from morphological and molecular data.
175 Nerve Growth Factor Potentiates Nicotinic Synaptic Transmission in Mouse Airway Parasympathetic Neur
174 Dopamine Increases CD14+CD16+ Monocyte Migration and Adhesion in the Context of Substance Abuse and
173 Crystal Structures of the Carboxyl cGMP Binding Domain of the Plasmodium falciparum cGMP-dependent P
172 Antibodies to the Plasmodium falciparum proteins MSPDBL1 and MSPDBL2 opsonise merozoites, inhibit pa
171 CD4 T-cell subsets in malaria: TH1/TH2 revisited.
170 CD40 Is Required for Protective Immunity against Liver Stage Plasmodium Infection.
169 IRGM3 contributes to immunopathology and is required for differentiation of antigen-specific effecto
```

1 Question 4.5

The invere-document-frequency cacluated by idf() penalizes terms that are common across many doucments by giving them a small weight. In particular, terms that appear in every document will have an idf weight of 0. Hence, there is gnerally no need to filter out the boilerplate text - these terms will not play any role in clustering or seaching in low dimensions.

One potential issue is when the terms in the boilerplate mask possibly meaningful terms in the text - for example, if you are clustering legal documents, the term "copyright" in the boilerplate may mask useful information for clustering IP-related documents that also use "copyright" in the actual text. In such cases, we would need to filter out boilerplate.

So either answer is fine, so long as it is justified.

1.1 Notes on the Pubmed articles

These were downloaded with the following script.

```
from Bio import Entrez, Medline
Entrez.email = "YOUR EMAIL HERE"
import cPickle

try:
    docs = cPickle.load(open('pubmed.pic'))
except Exception, e:
    print e

    docs = {}
    for term in ['plasmodium', 'diabetes', 'asthma', 'cytometry']:
        handle = Entrez.esearch(db="pubmed", term=term, retmax=50)
        result = Entrez.read(handle)
        handle.close()
        idlist = result["IdList"]
        handle2 = Entrez.efetch(db="pubmed", id=idlist, rettype="medline", retmode="text")
        result2 = Medline.parse(handle2)
        for record in result2:
            title = record.get("TI", None)
            abstract = record.get("AB", None)
            if title is None or abstract is None:
                continue
            docs[title] = '\n'.join([title, abstract])
            print title
        handle2.close()
    cPickle.dump(docs, open('pubmed.pic', 'w'))
docs.values()
```