Marc Jurchak

ECE 3522: Stochastic Processes in Signals and Systems

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 1912

# Problem Statement

In this assignment we will take data from MATLAB containing 329 cities and 9 categories, with a score for each category for each city. We will Do eigenvalue and eigenvector analysis on the covariance matrix of the data, and then transform the data to decorrelate it. We analyze this data to see which eigenvalues carry the most weight, and which categories are most impactful. Finally we find the most similar cities using a Euclidian distance from each other. We repeat this for the transformed data, and a third time only using the first the columns of the transformed data.

# Approach and Results

We begin by loading the “cities” data and compute the covariance matrix for the “ratings” data using the MATLAB “cov” function, shown below.

1.0e+07 \*

0.0015 0.0111 0.0026 0.0008 0.0014 0.0003 0.0127 0.0021 -0.0013

0.0111 0.5689 0.1084 0.0114 0.0941 0.0151 0.4967 0.0814 0.0697

0.0026 0.1084 0.1006 0.0109 0.0685 0.0158 0.4031 0.0264 0.0075

0.0008 0.0114 0.0109 0.0128 0.0149 0.0009 0.0646 0.0099 0.0101

0.0014 0.0941 0.0685 0.0149 0.2106 0.0156 0.3131 0.0428 0.0093

0.0003 0.0151 0.0158 0.0009 0.0156 0.0103 0.0556 0.0020 0.0042

0.0127 0.4967 0.4031 0.0646 0.3131 0.0556 2.1551 0.1420 0.0381

0.0021 0.0814 0.0264 0.0099 0.0428 0.0020 0.1420 0.0653 0.0152

-0.0013 0.0697 0.0075 0.0101 0.0093 0.0042 0.0381 0.0152 0.1176

Next we find the eigenvectors and eigenvalues using the “eig” function.

Eigenvalues:

1.0e+07 \*

0.0011

0.0067

0.0093

0.0241

0.0478

0.1076

0.1638

0.4408

2.4414

Eigenvectors (columns):

-0.9951 0.0421 -0.0814 -0.0012 0.0163 -0.0263 0.0067 -0.0155 0.0064

0.0229 0.0121 -0.0267 -0.0486 -0.0838 -0.1778 0.0826 -0.9372 0.2691

-0.0014 -0.2414 -0.1371 0.9295 -0.1591 -0.0266 -0.0278 0.0205 0.1783

0.0877 0.2668 -0.9448 -0.0540 0.1160 0.0990 -0.0376 0.0109 0.0281

-0.0094 -0.0415 0.0135 -0.0922 -0.1466 -0.0384 -0.9715 -0.0188 0.1493

0.0169 0.9292 0.2412 0.2532 -0.1063 0.0216 -0.0415 0.0014 0.0252

-0.0006 0.0159 0.0430 -0.1676 0.0087 0.0278 0.1510 0.2823 0.9309

0.0050 0.0188 0.1271 0.1733 0.9543 0.0690 -0.1496 -0.1038 0.0698

-0.0327 -0.0544 0.0702 0.0052 -0.1022 0.9745 -0.0127 -0.1734 0.0251

Next we compute the whitening transform, which is defined as follows:

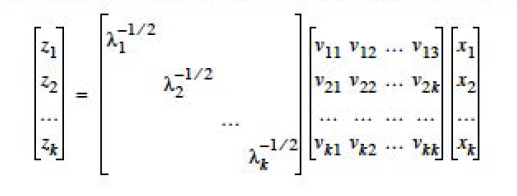


Figure : Whitening transform

Where the λ is the eigenvalues, v is the eigenvectors, and x is the input and z is the output. We also transform the “ratings” data to matrix “Y” of equal size. We take the covariance of this Y matrix and confirm that it is the identity matrix, shown below.

covar =

1.0000 -0.0000 0.0000 0.0000 -0.0000 -0.0000 0.0000 0.0000 -0.0000

-0.0000 1.0000 -0.0000 -0.0000 -0.0000 -0.0000 0.0000 0.0000 0.0000

0.0000 -0.0000 1.0000 -0.0000 0.0000 0.0000 0.0000 -0.0000 -0.0000

0.0000 -0.0000 -0.0000 1.0000 0.0000 -0.0000 -0.0000 -0.0000 -0.0000

-0.0000 -0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000 0.0000

-0.0000 -0.0000 0.0000 -0.0000 0.0000 1.0000 -0.0000 0.0000 -0.0000

0.0000 0.0000 0.0000 -0.0000 0.0000 -0.0000 1.0000 -0.0000 -0.0000

0.0000 0.0000 -0.0000 -0.0000 0.0000 0.0000 -0.0000 1.0000 0.0000

-0.0000 0.0000 -0.0000 -0.0000 0.0000 -0.0000 -0.0000 0.0000 1.0000

Next we plot the percent of the variance accounted for by n eigenvalues for n from 1 to 9. Result shown below.

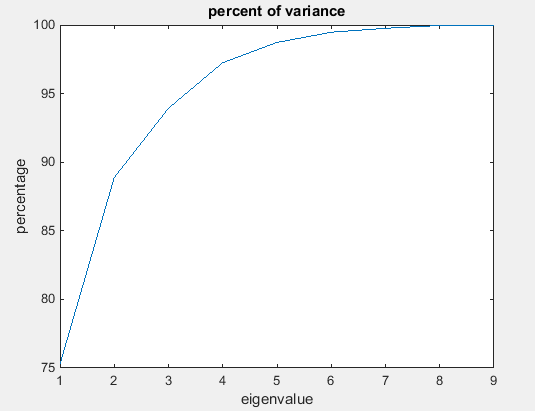


Figure : Percent of variance plot

Next we analyze the eigenvectors corresponding to the two largest eigenvalues, and see what variables in the ratings matrix are most influential in explaining the variance of the data. Result shown below.

|  |  |  |
| --- | --- | --- |
| climate | -0.0155 | 0.0064 |
| housing | -0.9372 | 0.2691 |
| health | 0.0205 | 0.1783 |
| crime | 0.0109 | 0.0281 |
| transportation | -0.0188 | 0.1493 |
| education | 0.0014 | 0.0252 |
| arts | 0.2823 | 0.9309 |
| recreation | -0.1038 | 0.0698 |
| economics | -0.1734 | 0.0251 |

Table : Eigenvalues and their corresponding categories. Values that carry the most eight are highlighted

Next we find the two cities that are the closest together using a Euclidean distance between their ratings, and found a third city that was close to the average of those cities rankings. We repeat this for the transformed data, and do this a third time only using the first three features from the transformed feature matrix. Results are shown below.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Raw data** |  |  |  |  |  |  |  |  |  |
| Johnstown, PA | 547 | 6524 | 731 | 353 | 4343 | 2691 | 666 | 903 | 4181 |
| Elkhart-Goshen, IN | 521 | 6573 | 596 | 524 | 4168 | 2537 | 353 | 1023 | 4214 |
| Altoona, PA | 561 | 6191 | 432 | 399 | 4246 | 2778 | 256 | 1210 | 4230 |
|  |  |  |  |  |  |  |  |  |  |
| **Transformed data** |  |  |  |  |  |  |  |  |  |
| Hamilton-Middletown, OH | -3.8543 | 8.9489 | 0.2399 | 1.8491 | -0.666 | 3.0731 | -2.1 | -3.838 | 1.0922 |
| Grand Rapids, MI | -3.7079 | 8.7178 | 0.3227 | 1.8962 | -0.254 | 3.4089 | -2.148 | -3.5455 | 1.2589 |
| Lewiston-Auburn, ME | -3.6557 | 8.9266 | 0.7196 | 1.9578 | 0.003 | 3.1277 | -2.086 | -3.4457 | 0.697 |
|  |  |  |  |  |  |  |  |  |  |
| **First three rows of Y** |  |  |  |  |  |  |  |  |  |
| St. Joseph, MO | -4.0686 | 10.093 | 0.7461 |  |  |  |  |  |  |
| Champaign-Urbana-Rantoul, IL | -4.061 | 10.125 | 0.7376 |  |  |  |  |  |  |
| Medford, OR | -4.1895 | 9.9515 | 0.7478 |  |  |  |  |  |  |

# MATLAB Code

**Whitening Transform**

function [ Y,covar ] = white\_transform( r )

% compute the covariance matrix

covariance = cov(r);

% covariance matrix

e = eig(covariance);

% V is the eigenvectors as columns

[V,D,W] = eig(covariance);

% create eigenvalue diagonal matrix for the transform

Da = e.^(-.5);

Da = diag(Da);

% Transform matrix, eigenvectors as rows with previous matrix

T = Da\*(V');

% Create output vector

dim = size(r);

Y = zeros(dim(1),dim(2));

% Compute Y = T\*X one row at a time

for i = 1:dim(1)

x = r(i,:)';

tran = T\*x;

Y(i,:) = tran';

end

% check if the covariance of the transformed data is I

covar = cov(Y);

end

**Three closest cities (Avg of first two)**

function [ top\_cities,city1\_data,city2\_data,city3\_data ] = three\_closest\_cities( ratings\_data )

load cities

% find the two cities with the closest data

[city1,city2,city1\_data,city2\_data,index1,index2]=closest\_cities(ratings\_data);

% compute the average of the two closest cities data

two\_cities = zeros(2,length(city1\_data));

two\_cities(1,:) = city1\_data;

two\_cities(2,:) = city2\_data;

avg\_city = mean(two\_cities,1);

% find a third city that is close to the average of the first two

[ city3,city3\_data ] = closest\_cities\_avg( ratings\_data,avg\_city,index1,index2 );

top\_cities = [city1 city2 city3];

end

**Two closest cities**

function [ city1,city2,city1\_data,city2\_data,index1,index2 ] = closest\_cities( ratings\_data )

load cities;

%euclidean distance of all ratings

d = pdist(ratings\_data);

%smallest value

small\_d = min(d);

%size of the ratings matrix

sizes = size(ratings\_data);

rows=sizes(1);

columns=sizes(2);

%will contain distance data

out = zeros(1,rows^2);

count = 1;

%nothing = 0;

% computes euclidean distance of all rows of ratings. Saves data for

% predetermined smallest distance

for m = 1:rows

for n = 1:rows

A = ratings\_data(m,:);

B = ratings\_data(n,:);

out(count) = pdist2(A,B);

if((out(count) ~= 0) && (out(count) <= small\_d))

m1 = m;

n1 = n;

city1\_data = A;

city2\_data = B;

end

count = count + 1;

end

end

%strings containing cities that are closest

city1 = names(m1,:);

city2 = names(n1,:);

index1 = m1;

index2 = n1;

end

**Finds third city from average of first two**

function [ name\_of\_city,city\_data ] = closest\_cities\_avg( data,city,index1,index2 )

load cities;

%size of the ratings matrix

sizes = size(data);

rows=sizes(1);

columns=sizes(2);

%will contain distance data

out = zeros(1,rows);

count = 1;

%nothing = 0;

% computes euclidean distance of all rows of ratings. Saves data for

% predetermined smallest distance

lowest = pdist2(data(1,:),city);

B = city;

for n = 1:rows

A = data(n,:);

out(count) = pdist2(A,B);

if((out(count) <= lowest)&&(n~=index1)&&(n~=index2))

n1 = n;

city\_data = A;

lowest = out(count);

end

count = count + 1;

end

%strings containing cities that are closest

name\_of\_city = names(n1,:);

end

**Top level, finding three closest cities**

clear;

clc;

load cities

% compute transformed data

[Y,covar]=white\_transform(ratings);

% find three cities that have the closest euclidean distance. Once for raw

% data, once for transformed data, and once for the first three columns of

% transformed data.

[cities\_transformed,city1\_data,city2\_data,city3\_data] = three\_closest\_cities(Y)

[cities\_not\_transformed,city1\_data,city2\_data,city3\_data] = three\_closest\_cities(ratings)

%cities\_transformed = three\_closest\_cities(Y(:,1:3))

**Parts 5 and 6**

clear;

clc;

load cities

% covariance matrix

cover = cov(ratings);

% sorted eigenvalues

e = eig(cover);

eSorted = sort(e,'descend');

% eigenvectors

[V,D,W] = eig(cover);

% Find and plot percent of total variance accounted for

per = zeros(1,length(eSorted));

total = 0;

for m = 1:length(eSorted)

total = total + eSorted(m);

per(m) = total/sum(eSorted);

end

%part 5

plot(100\*per)

title('percent of variance')

ylabel('percentage')

xlabel('eigenvalue')

% for part 6

%V(:,8:9);

# Conclusions

Since the diagonal of the covariance matrix is the variance of the data, we can see that the whitening transform forces the variance to 1 and scales all other data appropriately for each “dimension” of data. This scaling decorrelates the data so that all categories carry equal weight. For example in table 1, “housing” and “arts” carry lots of influence when explaining the variance of the data. Transforming the data helps us understand how all the data influences the variance. We can also see from figure 2 that the earlier eigenvalues carry much more weight than later ones. We can see that the data shifts as we transform the data. We can see from the top three cities that as the data is transformed, the data is pulled closer together, especially when only the first three rows of Y are used.