AMATH 582 Personal project

Hyesu Lee

March 19, 2020

Abstract

There are many factors that could influence the accent: speaker's culture, region, family, and the native language. The first language becomes habit and often stays when the person speaks in other languages. This experiment is conducted to examine the difference in accent with speech data of different native languages. I will build classification model that determines the speaker's native language. This experiment has two parts: comparison of accent with languages from different continents and same continent.

1 Introduction and Overview

Particular accent reflects a person's linguistic background. This experiment is based on the assumption that the speakers with same native language contains similar accent. I used the data set created by Steven H. Weinberger. [1] The data set includes recorded file of individuals speaking same English sentence. There are 2140 speakers with 214 different native language. In part1, I will examine the speaking sample which native languages are from two different continents(Polish/Romanian vs Japanese/Cantonese). Part2 will examine speaking sample which native languages are in same continent(Polish/Romanian vs. German/Italian).

2 Theoretical Background

2.1 Machine learning

Machine learning could be define as learning from data. With large data, machine learning(ML) algorithms find some pattern or structure to cluster and classify. There are three areas of machine learning algorithm: unsupervised learning, supervised learning, and reinforcement learning. For the purpose of the report, I will only elaborate on unsupervised and supervised learning.

2.2 Unsupervised learning

Unsupervised learning assumes that the given data set is unknown. The unsupervised learning algorithm finds some patterns within the given data set and make clusters of the data such that any data point within the cluster is similar to each other. The main idea is that the algorithm is to find pattern and the labels for each data set are not needed. Followings are some of the unsupervised learning algorithm: K-means, Mixture model.

2.3 Supervised learning

Supervised learning assume that given data is known. In other words, the data set has been already clustered and labels for each data set are given . Supervised learning algorithms "train" the machine with the given data set such that when a new data is given, the machine could classify the new data. Therefore, supervised learning algorithm contains testing set, training set, and label set. The label set represents the cluster each data point belongs. Lastly, it is important to cross-validate to prevent one specific discrimination based of one specific training set. Cross-validate could be done by repeating the training and testing process with the newly selected data points.

2.3.1 K-nearest-neighbors

K nearest neighbors (KNN) algorithm classify the given data point based on the K nearest values. Suppose there are 3 nearest values. If the two nearest values have label of dog and the other nearest value has label of cat, then the given data point is classified as dog. Also, KNN is capable of generating non-linear classification line.

2.3.2 Naive Bayes

Naive Bayes algorithm calculates the probability of each classes given some data. Equation 1 is representation of Naive Bayes algorithm with two classes. The right side of equation 1 is simpler to compute than the left side. It is how the Naive Bayes algorithm classify a new given data point.

$$\frac{P(1|x)}{P(0|x)} = \frac{P(x|1) * P(1)}{P(x|0) * P(0)} \tag{1}$$

2.3.3 Linear discriminant analysis

Linear discriminant analysis draws a line that separates the clusters. In other words, it projects every data points onto some basis and find a linear line for the classification. This algorithm is powerful to use with PCA, because PCA allows to find new bases such that when the data points projected onto those new bases, the clusters get separated nicely. Then, one could use linear discriminant analysis to find a linear classification line. Once the line is defined, then the any new data point could be classified easily.

2.3.4 Support vector machine

Support vector machine(SVM) finds the maximal separation line of given training set. Similar to linear discriminate analysis, SVM finds a line for separation, but SVM optimize the thickness of the line such that it has the maximal separation. The points on the edge of the cluster are called support vectors and SVM finds a line based on those support vectors. Any points on the wrong side would be a penalized factor. SVM allows to have curved surfaces in the higher spaces. Lastly, SVM does not require huge data set, because the only important points are the support vectors.

2.3.5 Classification and regression tree(CART)

Classification and regression tree(CART) generates a binary tree that breaks down the data set with some sorting based on the pattern of each cluster. By some binary separation, the new data could be classified as one of the cluster.

3 Algorithm Implementation and Development

3.1 Part 1

Because the given speech samples were spatially and computationally expensive, I cropped the sample to be first 5 seconds of the total speech sample length. Then, cropped frequency range of the spectrogram to only capture the significant frequency features. Due to different speaking rate, getting the first part of the speech was crucial. The part one compares accent of native language in Pacific Asian countries,Cantonese and Japanese, and Eastern European countries, Polish and Romanian. Part two compares accents of native language in Western European countries,German and Italian, and Eastern European countries, Polish and Romanian. For the supervised learning algorithm, I used linear discriminant analysis with PCA. For each group, there are 50 sample clips and I portioned 40 sample clips as training set and 10 sample clips as testing sets. Following implementation is for both part one and part two.

1. Load and parse the speech samples to same time length.

2. Perform Fourier transform and create spectrogram on each sample.

3. Reshape each sample to be a vector and make a matrix for each cluster.
4. Randomly choose training set and test set from each cluster.
5. Perform SVD on training set and find the significant singular values.
6. Project the cluster matrices onto the reduced principal components.
7. By using the projected features, use linear discriminant analysis to train and test.
8. Find the success rate of the classification.
9. Repeat procedure 6 8 with newly selected train set and test set for cross validation.
10. Find success rate distribution of all the cross validations.

4 Computational Results

4.1 Part 1

I used linear discriminant analysis with PCA, but the result was 50% success rate after 1000 time cross validation, which shows that classification did not succeeded. After examining the algorithm, I concluded that the data set of Japanese and Cantonese was not consistent enough to capture all the accent difference. Some data sample contain some pause or repeating sentences, which could cause noisy data set. Figure 1 is a singular value decomposition of the Eastern Europe native speaker accent and Pacific Asian native speaker accent. Compare to the part 2 examination which considers Western Europe vs Eastern Europe,figure 5, more singular values in figure 1 have energy. This feature consisted in different execution with one of the data sets, when I created extra examination just on Japanese vs. Cantonese native speaker accent. Figure 2 also shows that more singular values have energy compare to figure 5. Japanese vs. Cantonese native speaker accent experiment also resulted success rate of 50%.

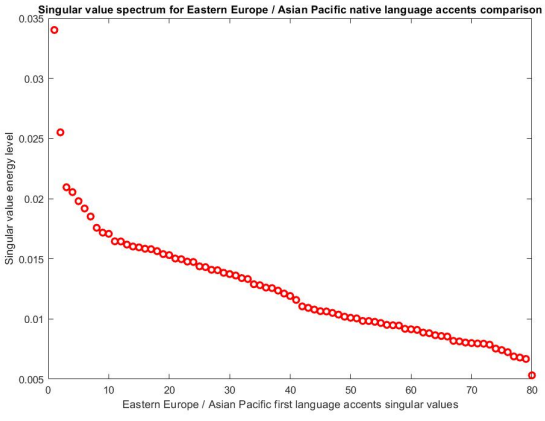

Figure 1: Europe vs Asia native speaker accent: singular value energy spectrum

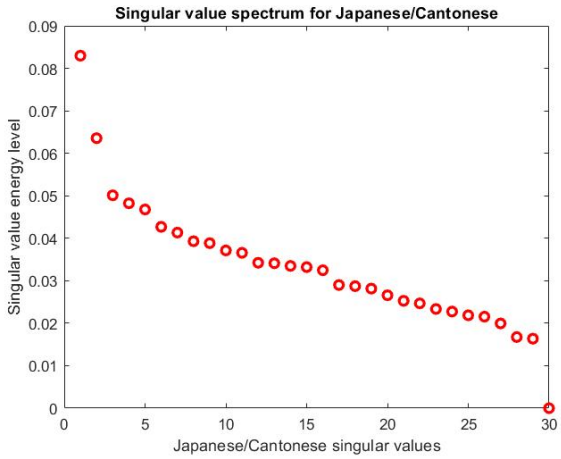

Figure 2: Japanese vs Cantonese native speaker accent:singular value energy spectrum

4.2 Part 2

I used linear discriminant analysis for the classification algorithm. I used speech sample whose native language is Polish, Romanian, German, and Italian, to classify the accent different with Western and Eastern European native language. I cross validate the 1000 times and plot the distribution. As shown in Figure 5,

30 singular values captures 79.66% of total energy. The distribution of success rate, 6, has mean of 61.05% with standard deviation of 6.41%. I repeated part 2 with larger frequency range in each spectrogram and

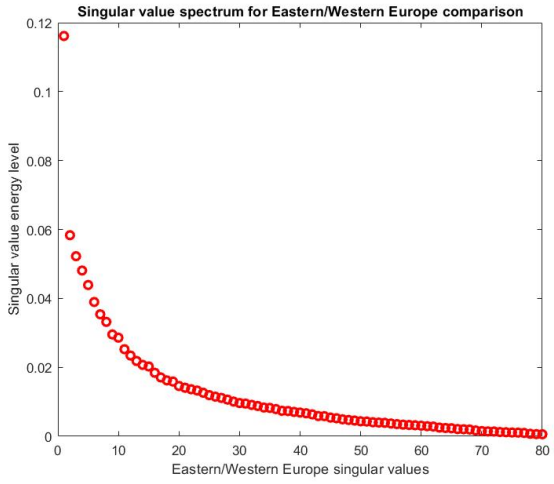

Figure 3: test1:singular value energy spectrum

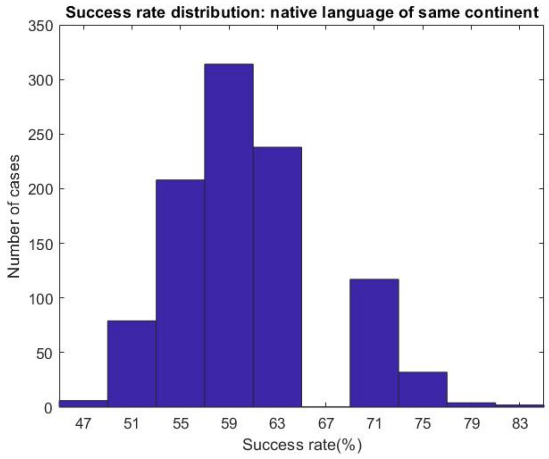

Figure 4: test1:success rate distribution

got better accuracy rate than previous test. The success rate was 71% with around 2% standard deviation.

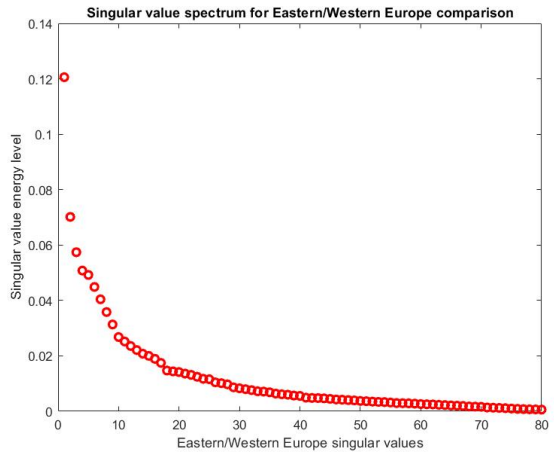

Figure 5: test1:singular value energy spectrum

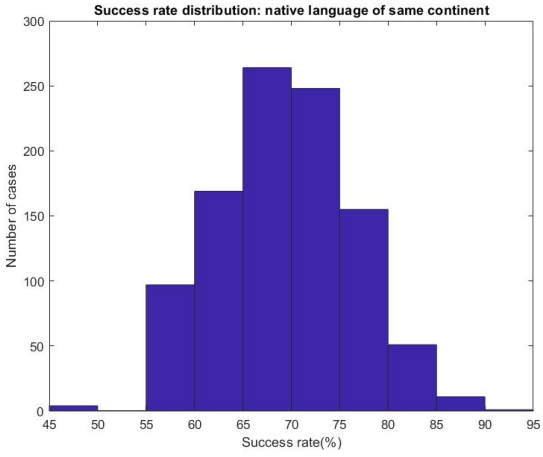

Figure 6: test1:success rate distribution

5 Summary and Conclusions

I did not expected to observe excellent separation especially on the Eastern European and Western European countries' language accent. However, it has success rate around 70% if the frequency range is wide enough. From the failed experiments, I understood the importance of data cleaning and pre-processing. If I continue to develop the accent recognition algorithm, I would pre-process such that data set has enough correlation for analysis or more efficient algorithm for noisy data.

References

[1] S.(George Mason University) Weinberger. *Speech accent archive*. This datasets is distributed under a CC BY-NC-SA 2.0 license. 2013. URL: `https://www.kaggle.com/rtatman/speech-accent-archive`.

Appendix A MATLAB Functions

- `B = reshape(A, sz)` return reshaped matrix `A`, using the size vector `sz` for size of matrix `B`.
- `imshow(A)` display the image A.
- `[U,S,V] = svd(X, 'econ')` produces a diagonal matrix S and unitary matrices U and V so that $X = U * S * V'$. With 'econ' parameter, executes the reduced SVD.
- `lambda = diag(S)` returns the diagonal entries of matrix S as vector.
- `P = randperm(N)` returns a vector containing a random permutation of the integers 1:N.
- `Classified = classify(sample, training, group)` classifies each row of the data in sample into one of the groups in training.
- `num = nnz(S)` returns the number of nonzero elements in S.
- `T = num2str(X)` converts the matrix X into its character representation T.
- `PD = fitdist(X,distname)` fits the probability distribution distname to the data in the column vector X, and returns an object PD representing the fitted distribution.
- `N = floor(X)` rounds the elements of X to the nearest integers towards minus infinity.
- `y = fftshift(x)` return the rearranged Fourier transform x which contains zero-frequency component at the center of the array.
- `y = fft(x)` returns the discrete Fourier transform of vector X.

Appendix B Github page

https://github.com/HyesLee99/Data-analysis/tree/master/Accent$_A nalysis$

Appendix C MATLAB Code

```
clc; close all; clear all; second = 220500;
%% native speakers of pacific asian vs. western european language

path_str = ['recordings/recordings/'];
Asian_pacific_str = [path_str,'continent/Asia_pacific'];
Eastern_europe_str = [path_str,'continent/Eastern Europe'];
Western_europe_str = [path_str, 'continent/Western Europe'];
diff_nations_str = [path_str, 'Nationality'];
Asian_pacific_dir = dir(Asian_pacific_str);
Asian_pacific_dir = Asian_pacific_dir(3:end);
song_title = [Asian_pacific_str, '/', Asian_pacific_dir(1).name];
[y,Fs]=audioread(song_title);
time  = 1:length(y);
seconds = time(end)/Fs;
y_sample = y(1:220500);
```

```
value = [y_sample];
%playblocking(audioplayer(y,Fs));
%%
for i = 1:length(Asian_pacific_dir)
    song_title = [Asian_pacific_str, '/', Asian_pacific_dir(i).name];
    [y,Fs]=audioread(song_title);
    y_sample = y(1:220500);
    value = [value, y_sample];
end
save Asian_pacific.dat value -ascii;
value = zeros(220500,1);
Eastern_europe_dir = dir(Eastern_europe_str);
Eastern_europe_dir = Eastern_europe_dir(3:end);
for i = 1:length(Eastern_europe_dir)
    song_title = [Eastern_europe_str, '/', Eastern_europe_dir(i).name];
    [y,Fs]=audioread(song_title);
    y_sample = y(1:220500);
    value = [value, y_sample];
end
value = value(:, 2:end);
save Eastern_europe.dat value -ascii;
value = zeros(220500,1);
Western_europe_dir = dir(Western_europe_str);
Western_europe_dir = Western_europe_dir(3:end);
for i = 1:length(Western_europe_dir)
    song_title = [Western_europe_str, '/', Western_europe_dir(i).name];
    [y,Fs]=audioread(song_title);
    y_sample = y(1:220500);
    value = [value, y_sample];
end
value = value(:, 2:end);
save Western_europe.dat value -ascii;
%%
value = zeros(220500,1);
diff_nations_dir = dir(diff_nations_str);
diff_nations_dir = diff_nations_dir(3:end);
for i = 1:length(diff_nations_dir)
    str = [diff_nations_str, '/',diff_nations_dir(i).name, '/'];
    nations = dir(str);
    test = [diff_nations_str,'/',diff_nations_dir(i).name, '/',nations(i).name];
    samples = dir(test);
    samples = samples(3:end);
    for j = 1:length(samples)
        song_title = [diff_nations_str,'/',diff_nations_dir(i).name, '/',nations(i).name,'/', samples(j
        [y,Fs]=audioread(song_title);
        y_sample = y(1:220500);
        value = [value, y_sample];
    end
end
value = value(:, 2:end);
save diff_nations.dat value -ascii;
%%
%1. Make a frequency and turn into spectrogram using gaussian
%2. build a matrix and call the svd
```

```
%3. find the singular value energy
%4. use the singular value to only plot specific things
%5. Project the value onto those vectors
%6. Then use the classify & find the statistical value

%% native language of estern european vs. Eastern european languages
clear all; close all; clc;
load Eastern_europe.dat;
load Western_europe.dat;
%%
v = Eastern_europe(:,1);
t2 = linspace(0,5,length(v));
t = t2(1:length(v));
tslide=0:0.1:5
vgt_spec=[];
n = 220500;
k = (2 * pi / 5) * [0 : n/2-1 -n/2:-1];
ks = fftshift(k);
ks = ks(:,1:end-1);
eastern_europe_spectro = [];
for i = 1:length(Eastern_europe(1,:))
    v = Eastern_europe(:,i);
    v = v';
    vgt_spec=[];
    for j=1:length(tslide)
        g=exp(-10*(t-tslide(j)).^2); % Gabor
        vg=g.*v;
        vgt=fft(vg);
        vgt_spec=[vgt_spec;
        abs(fftshift(vgt))];
        %subplot(3,1,1), plot(t,v,'k',t,g,'r')
        %subplot(3,1,2), plot(t,vg,'k')
        %temp = abs(fftshift(vgt));
        %temp = temp(:,1:end-1);
        %subplot(3,1,3), plot(ks,temp/max(abs(vgt)))
        %axis([-5 5 0 1])
        %drawnow
        %pause(0.05)
    end
    vgt_spec = vgt_spec';
    temp = vgt_spec(1:end-1,:);
    check = temp(end/2:end,:);
    temp = reshape(check, 110250* 51, 1);
    eastern_europe_spectro = [eastern_europe_spectro temp];
end
save Ee_parsed.dat eastern_europe_spectro -ascii;

%%
v = Western_europe(:,1);
t2 = linspace(0,5,length(v));
t = t2(1:length(v));
tslide=0:0.1:5
vgt_spec=[];
n = 220500;
```

```
k = (2 * pi / 5) * [0 : n/2-1 -n/2:-1];
ks = fftshift(k);
ks = ks(:,1:end-1);
western_europe_spectro = [];
for i = 1:length(Western_europe(1,:))
    v = Western_europe(:,i);
    v = v';
    vgt_spec=[];
    for j=1:length(tslide)
        g=exp(-10*(t-tslide(j)).^2); % Gabor
        vg=g.*v;
        vgt=fft(vg);
        vgt_spec=[vgt_spec;
        abs(fftshift(vgt))];
        %subplot(3,1,1), plot(t,v,'k',t,g,'r')
        %subplot(3,1,2), plot(t,vg,'k')
        %temp = abs(fftshift(vgt));
        %temp = temp(:,1:end-1);
        %subplot(3,1,3), plot(ks,temp/max(abs(vgt)))
        %axis([-5 5 0 1])
        %drawnow
        %pause(0.05)
    end
    vgt_spec = vgt_spec';
    temp = vgt_spec(1:end-1,:);
    check = temp(end/2:end,:);
    temp = reshape(check, 110250* 51, 1);
    western_europe_spectro = [western_europe_spectro temp];
end
save We_parsed.dat western_europe_spectro -ascii;
%%
western_europe_spectro = western_europe_spectro(:, 1:50);
eastern_europe_spectro = eastern_europe_spectro(:, 1:50);
%%
load Ee_parsed.dat;
save We_parsed.dat western_europe_spectro -ascii;
%%
figure(2)
vgt_spec = vgt_spec';
%%
figure(3)
temp = vgt_spec(1:15000,:);

%% Making spectrogram with asian countries accent %%%%%%%%%%%%%%%%%%%%%%%%%5555555
load Asian_pacific.dat;
%%
v = Asian_pacific(:,1);
t2 = linspace(0,5,length(v));
t = t2(1:length(v));
tslide=0:0.1:5
vgt_spec=[];
n = 220500;
k = (2 * pi / 5) * [0 : n/2-1 -n/2:-1];
ks = fftshift(k);
```

```
ks = ks(:,1:end-1);
Asian_pacific_spectro = [];
for i = 1:length(Asian_pacific(1,:))
    v = Asian_pacific(:,i);
    v = v';
    vgt_spec=[];
    for j=1:length(tslide)
        g=exp(-10*(t-tslide(j)).^2); % Gabor
        vg=g.*v;
        vgt=fft(vg);
        vgt_spec=[vgt_spec;
        abs(fftshift(vgt))];
        %subplot(3,1,1), plot(t,v,'k',t,g,'r')
        %subplot(3,1,2), plot(t,vg,'k')
        %temp = abs(fftshift(vgt));
        %temp = temp(:,1:end-1);
        %subplot(3,1,3), plot(ks,temp/max(abs(vgt)))
        %axis([-5 5 0 1])
        %drawnow
        %pause(0.05)
    end
    vgt_spec = vgt_spec';
    temp = vgt_spec(1:end-1,:);
    check = temp(end/2:end,:);
    temp = reshape(check, 110250* 51, 1);
    Asian_pacific_spectro = [Asian_pacific_spectro temp];
end
save AP_parsed.dat Asian_pacific_spectro -ascii;
%% graphing spectrogram
check = temp;%(end/2:end,:);
ks_check = ks(:,(end/2):(end/2)+14999);
pcolor(tslide,ks_check,check),
shading interp
set(gca,'Ylim',[0 2.5*10^4],'Fontsize',[14])
colormap(hot)
title('spectrogram: one eastern1')
xlabel('Time location (sec)')
ylabel('Frequency location (Hz)')
%%
figure(3)
pcolor(tslide,ks,temp),
shading interp
set(gca,'Ylim',[0 10^4],'Fontsize',[14])
colormap(hot)
title('spectrogram: one eastern2')
xlabel('Time location (sec)')
ylabel('Frequency location (Hz)')
%%
q1 = randperm(length(eastern_europe_spectro(1,:)));
q2 = randperm(length(western_europe_spectro(1,:)));

x1 = eastern_europe_spectro(:, q1(1:40));
x2 = western_europe_spectro(:, q2(1:40));
%%
```

```
x1 = x1(1:500000, :);
x2 = x2(1:500000, :);
western_europe_spectro = western_europe_spectro(1:500000, :);
eastern_europe_spectro = eastern_europe_spectro(1:500000, :);
%%
X = [x1, x2];
[u s v] = svd(X, 'econ');
S = diag(s)
figure(1)
plot(S/sum(S), 'ro', 'LineWidth', [2])
xlabel('Eastern/Western Europe singular values')
ylabel('Singular value energy level')
title('Singular value spectrum for Eastern/Western Europe comparison')
%%
%Eastern = Ee_parsed(:,1:50);
%Western = western_europe_spectro(:, 1:50);
% find the good energy level
energy = sum(S(1:30))/sum(S);
singular_values = 30;% 87.71%
projected_value = u(:, 1:singular_values)' * [eastern_europe_spectro western_europe_spectro];
xtrain = [projected_value(:, q1(1:40)) projected_value(:, q1(1:40))+50];

c_train = [ones(40,1) ; 2 * ones(40,1)];
true_value = [ones(10,1); 2* ones(10,1)];
cross_validate = [];
past_success = 0;
times = 1000;
for j = 1: times
    q1 = randperm(50);
    q2 = randperm(50);
    past_success = 0;
    xtest = [projected_value(:,q1(41:50)) projected_value(:, q2(41:50)+50)];
    result=classify(xtest', xtrain', c_train);
    indeces = result == true_value;
    success =nnz(indeces) / 20 * 100;
    if past_success < success
        figure(2)
        bar(classify(xtest', xtrain', c_train))
        past_success = success;
    end
    cross_validate = [cross_validate, success];
end
average_success = sum(cross_validate) / 1000;
figure(3)
hist(cross_validate)
xlabel('Success rate(%)');
ylabel('Number of cases');
title('Success rate distribution: native language of same continent')
pd = fitdist(cross_validate','Normal');
printing = ['The mean of ', num2str(times), 'cross-validation is ', num2str(pd.mu)];
printing2 = ['The std is ', num2str(pd.sigma)];
disp(printing);
disp(printing2);
```

```
%% compare japanese / German
c

%% compare japanese / cantonese
cantonese = Asian_pacific_spectro(:, 1:20);
japanese = Asian_pacific_spectro(:,25:44);

q1 = randperm(length(cantonese(1,:)));
q2 = randperm(length(japanese(1,:)));
x1 = cantonese(:, q1(1:15));
x2 = japanese(:, q2(1:15));
%%
X = [x1, x2];
[u s v] = svd(X, 'econ');
S = diag(s)
figure(1)
plot(S/sum(S), 'ro', 'LineWidth', [2])
xlabel('Japanese/Cantonese singular values')
ylabel('Singular value energy level')
title('Singular value spectrum for Japanese/Cantonese')

energy = sum(S(1:20))/sum(S);
singular_values = 20;% 80.75%
projected_value = u(:, 1:singular_values)' * [cantonese japanese];
xtrain = [projected_value(:, q1(1:15)) projected_value(:, q1(1:15))+20];

c_train = [ones(15,1) ; 2 * ones(15,1)];
true_value = [ones(5,1); 2* ones(5,1)];
cross_validate = [];
past_success = 0;
times = 1000;
%%
xtrain = abs(xtrain(:,:));
for j = 1: times
    q1 = randperm(20);
    q2 = randperm(20);
    past_success = 0;
    xtest = [projected_value(:,q1(16:20)) projected_value(:, q2(16:20)+20)];
    result=classify(xtest', xtrain', c_train);
    indeces = result == true_value;
    success =nnz(indeces) / 10 * 100;
    if past_success < success
        figure(2)
        bar(classify(xtest', xtrain', c_train))
        past_success = success;
    end
    cross_validate = [cross_validate, success];
end
average_success = sum(cross_validate) / 1000;
figure(3)
hist(cross_validate)
xlabel('Success rate(%)');
ylabel('Number of cases');
title('Success rate distribution: native language of same continent')
```

```
pd = fitdist(cross_validate','Normal');
printing = ['The mean of ', num2str(times), 'cross-validation is ', num2str(pd.mu)];
printing2 = ['The std is ', num2str(pd.sigma)];
disp(printing);
disp(printing2);
```