Contents

Elijah Bernstein-Cooper, Ahmed Saif, Ben Conrad - ECE532 Project - 141204

```
clc; clear all; close all; format compact;
set(0,'defaultaxescolor',.7*[1,1,1]);
```

Load Data

```
load('labData.mat');
```

Generate figure 1 - not required for students

```
%{
x = 0:.01:1;
data = randn(1,length(x)) + .5;
fit = polyfit(x,data,1);
y = polyval(fit,x);

msz = 20; lwd = 3;
figure(); hold on;
plot(x, data, 'b.','MarkerSize',msz);
plot(x, y, 'r-','LineWidth',lwd);
xlim([0,1]); ylim([0,1]);
fs = 20;
ylabel('Normalized Salary','FontSize',fs); xlabel('Normalized Occurrences of `analyse`','FontSiz
set(gca,'FontSize',fs);
%}
```

Activity 1) - pinv() Solution

{

```
fprintf('----Activity 1----\n');
desc1.id = 1; %from the activity1 training set
desc2.id = 2; %from the activity1 training set
desc3.id = 1; %from the test set
```

```
desc1.id = 1; %from the activity1 training set
desc2.id = 2; %from the activity1 training set
desc3.id = 1; %from the test set

%count unique words
fprintf('desc1:\n');
desc1.words = strsplit(data.activity1.train.desc{desc1.id});
desc1.words = strrep(desc1.words, ',','');
desc1.words = strrep(desc1.words, '.','');
desc1.words = lower(desc1.words);
desc1.words = sort(desc1.words);
desc1.uwords = unique(desc1.words);
for i = 1:length(desc1.uwords);
    desc1.a(i) = sum( strcmp(desc1.words, desc1.uwords{i}) );
    fprintf('%d = %s\n', desc1.a(i), desc1.uwords{i});
end

fprintf('\ndesc2:\n');
desc2.words = strsplit(data.activity1.train.desc{desc2.id});
desc2.words = strrep(desc2.words, ',','');
desc2.words = strrep(desc2.words, '.','');
desc2.words = lower(desc2.words);
desc2.words = sort(desc2.words);
desc2.uwords = unique(desc2.words);
for i = 1:length(desc2.uwords);
    desc2.a(i) = sum( strcmp(desc2.words, desc2.uwords{i}) );
    fprintf('%d = %s\n', desc2.a(i), desc2.uwords{i});
end

fprintf('\ndesc3:\n');
desc3.words = strsplit(data.activity1.test.desc{desc3.id});
desc3.words = strrep(desc3.words, ',','');
desc3.words = strrep(desc3.words, '.','');
desc3.words = lower(desc3.words);
desc3.words = sort(desc3.words);
desc3.uwords = unique(desc3.words);
for i = 1:length(desc3.uwords);
    desc3.a(i) = sum( strcmp(desc3.words, desc3.uwords{i}) );
    fprintf('%d = %s\n', desc3.a(i), desc3.uwords{i});
end

%combine word lists
comb = [desc1.uwords, desc2.uwords];
comb = unique(comb);
comb = sort(comb);

i1 = 1; %per-description counters
i2 = 1;
A = zeros(2,length(comb));
for i = 1:length(comb);
    if i1 <= length(desc1.uwords) && strcmp( comb{i}, desc1.uwords{i1} )
        A(1,i) = desc1.a(i1);
        i1 = i1+1;
    end
    if i2 <= length(desc2.uwords) && strcmp( comb{i}, desc2.uwords{i2} )
        A(2,i) = desc2.a(i2);
        i2 = i2+1;
    end
end
[m,n] = size(A);
```

```
    end
end
[m,n] = size(A);

fprintf('\nA = \n'); nA = 10;
for i = 1:nA-1; fprintf('%d & ',A(1,i)); end; fprintf('%d\\cdots\\\\\n', A(1,nA));
for i = 1:nA-1; fprintf('%d & ',A(2,i)); end; fprintf('%d\\cdots\n', A(2,nA));

% Display the frequency table
fprintf('\nFreq table 1\n')
for i = 1:11;
    disp([desc1.uwords{i} ' & ' num2str(A(1,i)) ' \\'])
end
fprintf('\nFreq table 2\n')
for i = 1:11;
    disp([desc2.uwords{i} ' & ' num2str(A(2,i)) ' \\'])
end

fprintf('\nb matrix\n')
b = data.activity1.train.salary;
xhat = pinv(A)*b;
[xsrt,isrt] = sort(xhat,'descend');
for i = 1:11;
    if i < 11
        fprintf('%3.4f & ``%s" \\text{from} [%d, %d] \\\\ \n', xsrt(i), comb{isrt(i)}, A(1,isrt(
    end
end
```

```
----Activity 1----
desc1:
2 = ****k
1 = analysis
3 = analyst
1 = and
1 = are
2 = client
1 = development
3 = dorking
3 = engineering
1 = for
1 = in
1 = is
1 = keywords
1 = located
1 = looking
1 = mathematical
1 = miser
2 = modelling
1 = optimisation
2 = our
1 = pioneeer
1 = provides
1 = risk
2 = salary
1 = software
1 = specialist
3 = surrey
1 = system
```

```
1 = specialist
3 = surrey
1 = system
3 = systems

desc2:
1 = 1015
2 = a
2 = all
2 = also
3 = an
5 = and
1 = assessments
1 = at
6 = be
2 = cable
1 = cables
1 = candidate
1 = care
1 = cfd
1 = chartered
2 = company
1 = competitive
1 = developing
2 = engineer
2 = engineering
2 = experience
1 = experienced
1 = expert
1 = extremely
1 = fea
4 = for
1 = gas
1 = has
1 = have
1 = health
1 = in
1 = industries
1 = interpersonal
1 = is
1 = issues
1 = it
1 = leadership
1 = least
1 = looking
1 = membership
1 = methods
1 = modelling
4 = must
1 = need
1 = new
1 = of
1 = offers
1 = offshore
1 = oil
1 = or
1 = package
1 = plan
1 = proactive
1 = problem
1 = professional
```

```
1 = proactive
1 = problem
1 = professional
2 = providing
1 = qualification
1 = qualified
1 = related
1 = relocation
2 = responsible
1 = risk
1 = salary
1 = significant
1 = skills
1 = solving
1 = someone
1 = sponsorship
1 = staff
1 = strong
3 = subsea
1 = technical
3 = the
1 = therefore
1 = they
2 = to
1 = towards
1 = training
2 = who
4 = will
1 = with
1 = within
1 = working
1 = years
4 = you

desc3:
5 = a
1 = after
2 = an
8 = and
1 = area
2 = as
1 = based
2 = be
1 = both
1 = chain
1 = client
5 = cluster
1 = company's
1 = conference
1 = continued
1 = coordination
1 = currently
2 = customer
1 = deliver
1 = development
2 = drive
1 = experienced
2 = for
1 = four
1 = from
```

```
2 = for
1 = four
1 = from
1 = growth
1 = hertfordshire
1 = hotel
3 = hotels
2 = in
1 = influence
1 = international
1 = is
1 = job
1 = large
1 = looking
1 = management
7 = manager
1 = managing
1 = market
1 = maximise
1 = minimum
1 = mission
2 = multiple
4 = of
2 = or
1 = originally
1 = our
1 = part
1 = people
1 = posted
1 = procedures
1 = processes
1 = profit
2 = profits
1 = propertys
1 = quality
1 = regional/
1 = relating
1 = require
11 = revenue
1 = rooms
1 = service
1 = share
1 = space
1 = standards
1 = strategic
1 = that
5 = the
1 = this
1 = three
1 = through
4 = to
1 = two
1 = unit
1 = was
4 = will
1 = wwwcaterercom/jobseeking/clusterrevenuemanager_job****
2 = you

A =
2 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 1 & 1\cdots\\
A =
2 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 1 & 1\cdots\\
0 & 1 & 2 & 2 & 2 & 3 & 0 & 0 & 5 & 0\cdots

Freq table 1
****k & 2 \\
analysis & 0 \\
analyst & 0 \\
and & 0 \\
are & 0 \\
client & 0 \\
development & 1 \\
dorking & 3 \\
engineering & 1 \\
for & 1 \\
in & 0 \\

Freq table 2
1015 & 0 \\
a & 1 \\
all & 2 \\
also & 2 \\
an & 2 \\
and & 3 \\
assessments & 0 \\
at & 0 \\
be & 5 \\
cable & 0 \\
cables & 1 \\

b matrix
1819.6517 & ``be" \text{from} [0, 6] \\
1730.9558 & ``and" \text{from} [1, 5] \\
1427.6805 & ``for" \text{from} [1, 4] \\
1250.2887 & ``engineering" \text{from} [3, 2] \\
1213.1011 & ``must" \text{from} [0, 4] \\
1213.1011 & ``will" \text{from} [0, 4] \\
1213.1011 & ``you" \text{from} [0, 4] \\
909.8258 & ``an" \text{from} [0, 3] \\
909.8258 & ``subsea" \text{from} [0, 3] \\
909.8258 & ``the" \text{from} [0, 3] \\
```

Activity 2) - Estimate Salary of Third Description

```
fprintf('\n\n----Activity 2----\n');
% Now predict salary of a new description
i1 = 1; %per-description counters
A3 = zeros(1,length(comb));
for i = 1:length(comb);
    if i1 <= length(desc3.uwords) && strcmp( comb{i}, desc3.uwords{i1} )
        A3(1,i) = desc3.a(i1);
        i1 = i1+1;
    end
end

b3 = data.activity1.test.salary(desc3.id);

bhat = A3*xhat;
```

```
b3 = data.activity1.test.salary(desc3.id);

bhat = A3*xhat;
fprintf('Salary: Actual %3.2f vs Predicted %3.2f  difference = %3.2f\n', b3, bhat, norm(b3-bhat)
%}
```

```
----Activity 2----
Salary: Actual 45000.00 vs Predicted 3032.75  difference = 41967.25
```

Activity 3

Following the same method as in the warm-up, describe how the frequency matrix will change with added descriptions. If we include every word from each of the descriptions in our frequency matrix, is it likely that the number of salaries will be greater than the number of words in the frequency matrix? Describe how the computation time scales with the size of the frequency matrix. Is it practical to include every word from each description in the frequency matrix?

Activity 4

With the data in data.activity4, construct a frequency matrix of words, excluding common words you are confident will not be important to a description's salary. Derive a linear least squares solution for $\hat{\bm{x}}$ in Equation~\ref{eq:linsolve} with the first 10 descriptions in the training set. Predict the salaries of the first 10 descriptions in the test set. Report the L_2 norm of the difference between the true salaries and predicted salaries for the test set. Report the word which is best able to predict the salary of a description. State whether this problem is over determined or under determined.

```
Error updating Text. Following is the chain of causes of the error:

 String must have valid interpreter syntax:
$\hat{\bm{x}}$
```

```
% {
fprintf('\n\n----Activity 4----\n');
% Number of data points
N = 10;

% Grab every word in the description
descTrain = data.activity4.train.desc(1:N);
descTest = data.activity4.test.desc(1:N);

%get the salary
salaryTrain = data.activity4.train.salary(1:N);
salaryTest = data.activity4.test.salary(1:N);

tic
all_words = true;
if all_words
    words = {'RemoveTheInitialWordFromwords'};
    for i = 1:N
        text = strsplit(descTrain{i}, ' ');
        for j = 1:length(text)
            %Remove all non English letters characters and numbers
            text{j} = regexprep(text{j},'[^a-zA-Z0-9]','');
            % turn to all letters to lower case
            text{j} = lower(text{j});
            %Add text to words
            if length(text{j}) > 2
                words = [words, text{j}]; %#ok<AGROW>
            %Add text to words
            if length(text{j}) > 2
                words = [words, text{j}]; %#ok<AGROW>
            end
        end
    end
    words = words(2:end); %remove the initial
    keywords = unique(words);
end
%remove the ignore words that probably dont contribute anything to the data
%set.
ignore = {'be' 'at' 'you' 'we' 'the' 'and' 'it' 'them' 'a' 'these' ...
          'those' 'with' 'can' 'for' 'an' 'is' 'or' 'of' 'are' 'has' 'have' ...
          'in' 'or' 'to' 'they' 'he' 'she' 'him' 'her' 'also'...
          '', 'able','all','as','but','by','cv','every','from','get','had','if','its',...
          'not','on','only','our','put','per','so','that','this','what','will','year','years','y
keywords = setdiff(keywords, ignore);
keywords = sort(keywords);

%calculate the frequency matrix
nKeys = length(keywords);
freqMatrixTrain = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTrain,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTrain(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(desc
    end
end

% Get frequencies of keywords, or A matrix, of the train Set
tic
nKeys = length(keywords);
freqMatrixTrain = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTrain,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTrain(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(desc
    end
end

% Get frequencies of keywords, or A matrix, of the test Set
freqMatrixTest = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTest,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTest(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(descT
    end
end
toc

% Predict
xhat = pinv(freqMatrixTrain)*salaryTrain;
bhat = freqMatrixTest*xhat;


% Most useful words:
[xsrt,isrt] = sort(xhat,'descend');
for i = 1:11;
    if i < 11
        fprintf('%3.4f & ``%s"\n', xsrt(i), keywords{isrt(i)});
for i = 1:11;
    if i < 11
        fprintf('%3.4f & ``%s"\n', xsrt(i), keywords{isrt(i)});
    end
end

fprintf('\nSalary: Actual vs. Predicted\n');
fprintf(' %3.2f vs %3.2f\n',salaryTest, bhat);
fprintf('L2 norm = %3.4f\n', norm(salaryTest-bhat));
%}
```

```
----Activity 4----
Elapsed time is 0.155303 seconds.
16308705.7655 & ``product"
16043615.9143 & ``end"
15524292.9763 & ``test"
14998050.2859 & ``client"
14723609.0738 & ``one"
14529106.2897 & ``lead"
11867795.5393 & ``their"
11855061.2498 & ``work"
9499545.2452 & ``own"
9038677.3404 & ``within"

Salary: Actual vs. Predicted
 32500.00 vs 32640.00
 42500.00 vs 35000.00
 28500.00 vs 33500.00
 27500.00 vs 26500.00
 25000.00 vs 33600.00
 32945.82 vs 33800.59
 29864.47 vs 17376.76
 25709.92 vs 26564.36
 34604.42 vs 22422.68
 40718.05 vs 40284.02
L2 norm = 29774.1366
```

Activity 5 - Construct A

Number of data points

```
N = 500;

% Grab every word in the description
descTrain = data.activity4.train.desc(1:N);
descTest = data.activity4.test.desc(1:N);

%get the salary
salaryTrain = data.activity4.train.salary(1:N);
salaryTest = data.activity4.test.salary(1:N);

tic
all_words = true;
if all_words
    words = {'RemoveTheInitialWordFromwords'};
    for i = 1:N
        text = strsplit(descTrain{i}, ' ');
    words = {'RemoveTheInitialWordFromwords'};
    for i = 1:N
        text = strsplit(descTrain{i}, ' ');
        for j = 1:length(text)
            %Remove all non English letters characters and numbers
            text{j} = regexprep(text{j},'[^a-zA-Z0-9]','');
            % turn to all letters to lower case
            text{j} = lower(text{j});
            %Add text to words
            if length(text{j}) > 2
                words = [words, text{j}]; %#ok<AGROW>
            end
        end
    end
    words = words(2:end); %remove the initial
    keywords = unique(words);
end
%remove the ignore words that probably dont contribute anything to the data
%set.
ignore = {'be' 'at' 'you' 'we' 'the' 'and' 'it' 'them' 'a' 'these' ...
          'those' 'with' 'can' 'for' 'an' 'is' 'or' 'of' 'are' 'has' 'have' ...
          'in' 'or' 'to' 'they' 'he' 'she' 'him' 'her' 'also'...
          '', 'able','all','as','but','by','cv','every','from','get','had','if','its',...
          'not','on','only','our','put','per','so','that','this','what','will','year','years','y
keywords = setdiff(keywords, ignore);
keywords = sort(keywords);

%calculate the frequency matrix
nKeys = length(keywords);
freqMatrixTrain = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTrain,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTrain(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(desc
    end
end

% Get frequencies of keywords, or A matrix, of the train Set
tic
nKeys = length(keywords);
freqMatrixTrain = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTrain,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTrain(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(desc
    end
end

% Get frequencies of keywords, or A matrix, of the test Set
freqMatrixTest = zeros(N,nKeys);
for ikeys = 1:nKeys;
    a = strfind(descTest,keywords{ikeys});
    for idesc = 1:N;
        freqMatrixTest(idesc, ikeys) = length(a{idesc}) / length(keywords{ikeys}) / length(descT
    end
end
toc
```

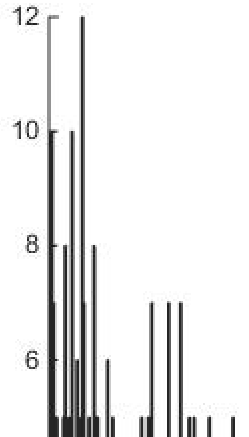
```
Elapsed time is 64.127053 seconds.
Elapsed time is 64.127053 seconds.
```

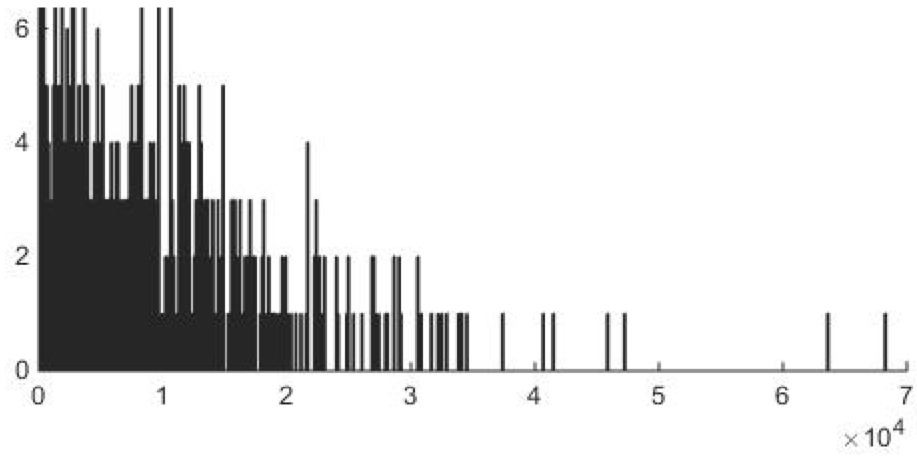
Activity 5a) - Lasso Implementation

```
fprintf('\n\n----Activity 5a----\n');
A = freqMatrixTrain; b = salaryTrain;
xhat = zeros(size(A,2),1);
[u,s,v] = svd(A'*A);
alpha = 1/s(1,1);

lambda = 100;
maxIter = 1e3;
eps = 10^-5;
delta = 10;
iter = 0;

tic
while( iter < maxIter && delta > eps)
    y = xhat + alpha*A'*(b-A*xhat);
    xnext = sign(y) .* max([abs(y) - alpha*lambda,zeros(size(A,2),1)],[],2);
    iter = iter+ 1;
    delta = norm(xnext - xhat);
    xhat = xnext;
end
toc
if iter >= maxIter;
    warning('MATLAB:LassoWarmup','Lasso exited at max iterations (%d) with delta %3.4f > eps %3.
end
xhat100 = xhat;

% Number of words
fprintf('For lambda = %d, there are %d words\n', lambda, numel(find(xhat ~= 0)));

% Display top key words
[~,ind] = sort(xhat,'descend');
for i = 1:(numel(find(xhat ~= 0))+1);
    fprintf('%d (%d) @ %3.4f = [%s]\n', i, ind(i), xhat(ind(i)), keywords{ind(i)} );
end

% Histogram
bhat = freqMatrixTest * xhat;
figure('Name','5a - Salary Residual Histogram'); hold on;
hist( abs(salaryTest - bhat), N );
```

```
----Activity 5a----
Elapsed time is 45.965321 seconds.
Warning: Lasso exited at max iterations (1000) with delta 5033.2916 > eps 0.0000
For lambda = 100, there are 46 words
1 (5971) @ 6999517.6914 = [sea]
2 (5261) @ 6481144.2753 = [project]
3 (4532) @ 5472102.3420 = [nsi]
4 (6146) @ 5449152.4198 = [sign]
5 (7347) @ 4829607.8560 = [work]
6 (2282) @ 4577990.7116 = [eng]
```

```
4 (6146) @ 5449152.4198 = [sign]
5 (7347) @ 4829607.8560 = [work]
6 (2282) @ 4577990.7116 = [eng]
7 (3568) @ 4433302.0763 = [ive]
8 (3536) @ 4160491.3373 = [ion]
9 (1940) @ 4001526.8908 = [develop]
10 (2274) @ 4000167.7348 = [end]
11 (393) @ 3947507.2041 = [ami]
12 (300) @ 3780658.5401 = [age]
13 (3154) @ 3596311.5027 = [hit]
14 (5183) @ 3494785.8482 = [pro]
15 (3815) @ 2784385.4821 = [lie]
16 (1349) @ 2669959.7646 = [com]
17 (3752) @ 2625191.4173 = [lead]
18 (2407) @ 2448339.1395 = [etc]
19 (459) @ 2284157.0591 = [app]
20 (5836) @ 2128458.1970 = [round]
21 (6447) @ 2106492.6777 = [str]
22 (3146) @ 2104062.0790 = [hire]
23 (3884) @ 1998734.8461 = [log]
24 (404) @ 1996081.7020 = [anage]
25 (4466) @ 1989995.9800 = [nic]
26 (4574) @ 1909452.3484 = [off]
27 (6182) @ 1808851.9589 = [sit]
28 (788) @ 1610148.7161 = [ben]
29 (5857) @ 1410896.2008 = [rtu]
30 (2492) @ 1259031.5263 = [exp]
31 (6003) @ 1208354.7211 = [see]
32 (3279) @ 1162297.5857 = [ill]
33 (3773) @ 1045626.6289 = [led]
34 (6180) @ 958644.7670 = [sis]
35 (5467) @ 943069.3382 = [rec]
36 (204) @ 862401.1445 = [act]
37 (2258) @ 837678.7788 = [ems]
38 (614) @ 813768.7636 = [ate]
39 (3710) @ 812442.2135 = [lan]
40 (3148) @ 724892.7685 = [his]
41 (2383) @ 714617.0539 = [ess]
42 (5014) @ 573216.7703 = [port]
43 (7201) @ 492022.0393 = [ware]
44 (7297) @ 192369.7342 = [wil]
45 (1) @ -0.0000 = [011]
46 (2) @ 0.0000 = [05m]
47 (3) @ 0.0000 = [100]
```



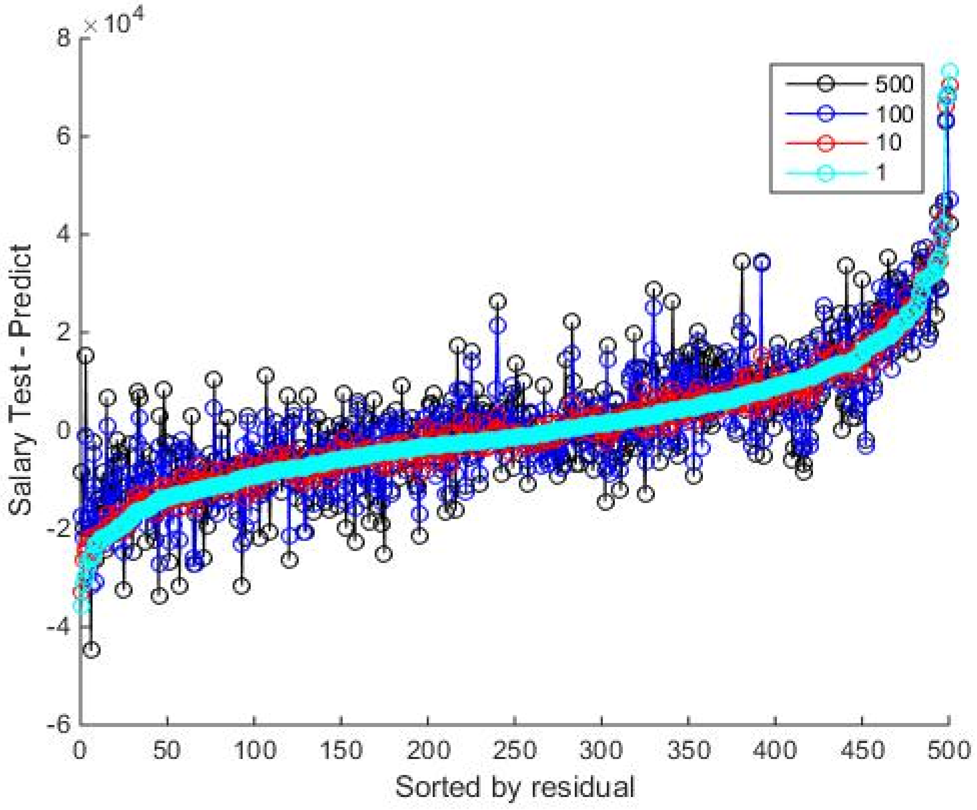


Activity 5b) - using lasso with different lambda values

```
fprintf('\n\n----Activity 5b----\n');
%N=100
% lambda = .001; maxIter = 1e4; %max its, delta 68, error 1e5
% lambda = .00001; maxIter = 1e5; %max its, delat .7 error 1e5, 4.4min
% lambda = .1; maxIter = 1e4;
%N=300:
% lambda = .01; maxIter = 1e4; % max its, delta 1513, error 2e5, 5.2 min
% lambda = .001; maxIter = 1e5; %max its, delat 20, error 2e5, 32 min
%N=500:
% lambda = .1; maxIter = 1e3; %max its, delta 32940, error 3e5, 5 min

tic
lambda = 500;
xhat = zeros(size(A,2),1); iter = 0; delta = 10;
while( iter < maxIter && delta > eps)
    y = xhat + alpha*A'*(b-A*xhat);
    xnext = sign(y) .* max([abs(y) - alpha*lambda,zeros(size(A,2),1)],[],2);
    iter = iter+ 1;
    delta = norm(xnext - xhat);
    xhat = xnext;
end
xhat500 = xhat;

lambda = 10;
xhat = zeros(size(A,2),1); iter = 0; delta = 10;
while( iter < maxIter && delta > eps)
    y = xhat + alpha*A'*(b-A*xhat);
    xnext = sign(y) .* max([abs(y) - alpha*lambda,zeros(size(A,2),1)],[],2);
    iter = iter+ 1;
    delta = norm(xnext - xhat);
    xhat = xnext;
end
xhat010 = xhat;

lambda = 1;
xhat = zeros(size(A,2),1); iter = 0; delta = 10;
while( iter < maxIter && delta > eps)
    y = xhat + alpha*A'*(b-A*xhat);
    xnext = sign(y) .* max([abs(y) - alpha*lambda,zeros(size(A,2),1)],[],2);
    iter = iter+ 1;
    delta = norm(xnext - xhat);
    xhat = xnext;
end
```

```
    delta = norm(xnext - xhat);
    xhat = xnext;
end
xhat001 = xhat;
toc

r500 = salaryTest - freqMatrixTest*xhat500;
r100 = salaryTest - freqMatrixTest*xhat100;
r010 = salaryTest - freqMatrixTest*xhat010;
r001 = salaryTest - freqMatrixTest*xhat001;
[~,ind] = sort(r001);

figure('Name','5b - Salary Residuals'); hold on;
plot( 1:N, r500(ind), 'ko-');
plot( 1:N, r100(ind), 'bo-');
plot( 1:N, r010(ind), 'ro-');
plot( 1:N, r001(ind), 'co-');
xlabel('Sorted by residual'); ylabel('Salary Test - Predict');
legend('500','100','10','1');
```

```
----Activity 5b----
Elapsed time is 137.632123 seconds.
```



Activity 5c) -

Experiment with the Lasso method using smaller values of λ. What λ value do you think keeps the number of words reasonable while still including enough words in the weight matrix to accurately predict the salary? This can be an art, there is not

Activity 6a) - Predict salaries of a Moder Major General and an EE Grad

Activity 6a) - Predict salaries of a Moder Major General and an EE Grad

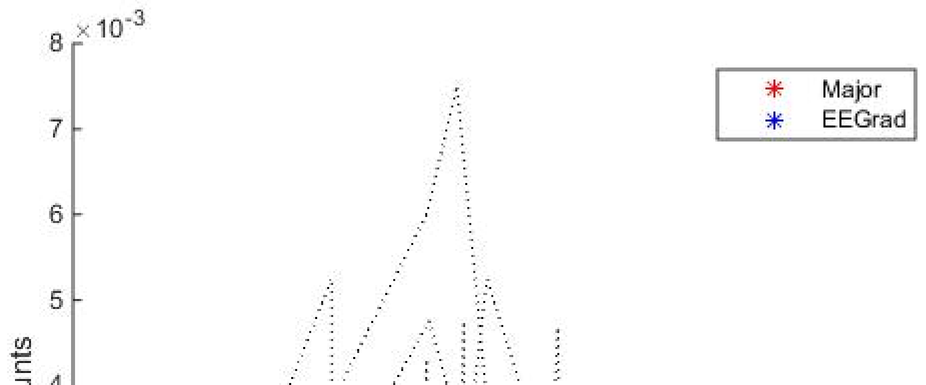
```
fprintf('\n\n----Activity 6a----\n');
Amjr = zeros(1,nKeys);
for i = 1:nKeys;
    Amjr(i) = length( strfind(data.activity6.major, keywords{i}) ) / length(keywords{i}) / lengt
end

Aeeg = zeros(1,nKeys);
for i = 1:nKeys;
    Aeeg(i) = length( strfind(data.activity6.eengr, keywords{i}) ) / length(keywords{i}) / lengt
end

[~,ind] = min([norm(r500),norm(r100),norm(r010),norm(r001)]);
xhat = [xhat500,xhat100,xhat010,xhat001]; xhat = xhat(:,ind);
predMajor = Amjr * xhat;
predEEGrad = Aeeg * xhat;

fprintf('Predicted Major General %3.4f\n', predMajor);
fprintf('Predicted Electrical Grad %3.4f\n', predEEGrad);
```

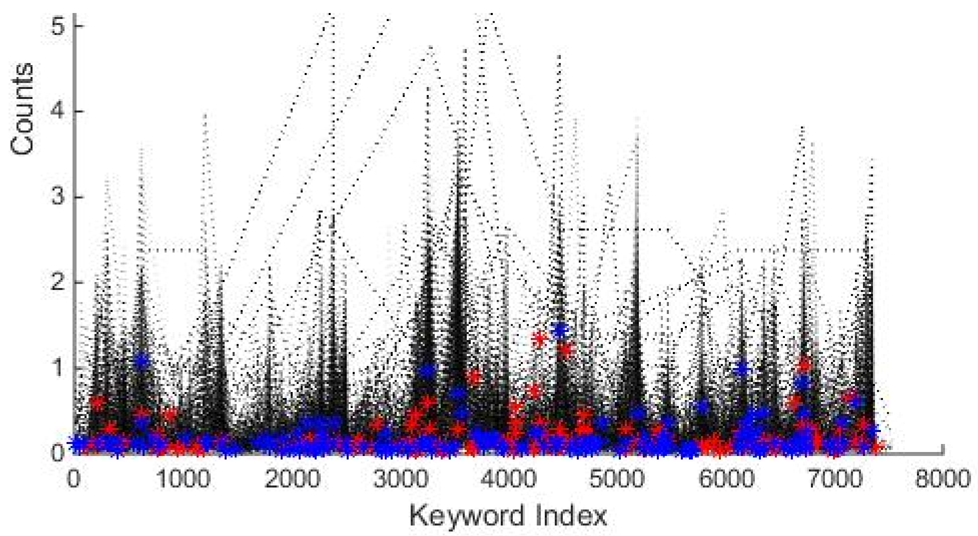
```
----Activity 6a----
Predicted Major General 7109.0585
Predicted Electrical Grad 19458.6262
```

Activity 6b) -

The salaries fall substantially below the poverty line in any country; why is this the case? Compare the most important keywords against those contained in \texttt{majorGen} and \texttt{elecGrad}; it may help to consider how the predicted salary is computed.

```
figure('Name','6b - Occurrences of Keywords'); hold on;
col = gray(N);
[~,inds] = sort(sum(freqMatrixTrain,2),'ascend');
for i = N:-1:1;
    ii = inds(i);
    ind = find(freqMatrixTrain(ii,:) > 0);
    plot( ind, freqMatrixTrain(ii,ind), ':', 'Color',col(i,:) );
end
ind = find(Amjr > 0); h1 = plot( ind, Amjr(ind), 'r*' ); set(h1,'DisplayName','Major');
ind = find(Aeeg > 0); h2 = plot( ind, Aeeg(ind), 'b*' ); set(h2,'DisplayName','EEGrad');
xlabel('Keyword Index'); ylabel('Counts');
legend([h1,h2])
```





Published with MATLAB® R2014b