Final Project

Shiyu Tu

999603754

1 Question 1

1.1

· This is my code to classification, it breaks the training data from 0,1,2,3,4,5,6,7,8,9, total 10 classes. Code computes the SVD of each class and makes a prediction.

```
import classification_digits.m

load azip.mat; %load training matrix 256*1707
load dzip.mat; %load the correct answer of the training digits 1*1707
load dtest.mat; %load the correct answer of the test digits 1*2007
load testzip.mat %load test digits data 256*2007

num = zeros(1,10); %get how many numbers in 0-9 classes

for i = 1:1707
    if dzip(i) == 0
        num(10) = num(10) + 1;
    elseif dzip(i) == 1
        num(1) = num(1) + 1;
    elseif dzip(i) == 2
        num(2) = num(2) + 1;
    elseif dzip(i) == 3
        num(3) = num(3) + 1;
    elseif dzip(i) == 4
        num(4) = num(4) + 1;
    elseif dzip(i) == 5
        num(5) = num(5) + 1;
    elseif dzip(i) == 6
        num(6) = num(6) + 1;
    elseif dzip(i) == 7
```

```
        num(7) = num(7) + 1;
    elseif dzip(i) == 8
        num(8) = num(8) + 1;
    elseif dzip(i) == 9
        num(9) = num(9) + 1;
    end
end

for i = 1:10                        % get variables started
    Matrix{i} = zeros(256,num(i));
    Upper{i} = zeros(256,num(i));
    Sig{i} = zeros(256,num(i));
    Vig{i} = zeros(256,num(i));
    X_5{i} = zeros(256,num(i));
    X_10{i} = zeros(256,num(i));
    X_20{i} = zeros(256,num(i));
    Bk{i} = zeros(256,num(i));
end

numcount = zeros(1,10);

for i = 1:1707              % classificate numbers according to dzip
    if dzip(i) == 0
        numcount(10) = numcount(10) + 1;
        Matrix{10}(:,numcount(10)) = azip(:,i);
    elseif dzip(i) == 1
        numcount(1) = numcount(1) + 1;
        Matrix{1}(:,numcount(1)) = azip(:,i);
    elseif dzip(i) == 2
        numcount(2) = numcount(2) + 1;
        Matrix{2}(:,numcount(2)) = azip(:,i);
    elseif dzip(i) == 3
        numcount(3) = numcount(3) + 1;
        Matrix{3}(:,numcount(3)) = azip(:,i);
    elseif dzip(i) == 4
        numcount(4) = numcount(4) + 1;
        Matrix{4}(:,numcount(4)) = azip(:,i);
    elseif dzip(i) == 5
        numcount(5) = numcount(5) + 1;
        Matrix{5}(:,numcount(5)) = azip(:,i);
    elseif dzip(i) == 6
        numcount(6) = numcount(6) + 1;
        Matrix{6}(:,numcount(6)) = azip(:,i);
    elseif dzip(i) == 7
        numcount(7) = numcount(7) + 1;
        Matrix{7}(:,numcount(7)) = azip(:,i);
```

```
    elseif dzip(i) == 8
        numcount(8) = numcount(8) + 1;
        Matrix{8}(:,numcount(8)) = azip(:,i);
    elseif dzip(i) == 9
        numcount(9) = numcount(9) + 1;
        Matrix{9}(:,numcount(9)) = azip(:,i);
    end
end

for i = 1:10                    %get the SVD values and other
    varibalesstarted
    [Upper{i},Sig{i},Vig{i}] = svd(Matrix{i});
    Matrix_SVD5{i} = zeros(256,1);
    Matrix_SVD10{i} = zeros(256,1);
    Matrix_SVD20{i} = zeros(256,1);
     U5{i} = zeros(256,5);
     U10{i} = zeros(256,10);
     U20{i} = zeros(256,20);
     findvalue1{i} = zeros(1,10);
     findvalue2{i} = zeros(1,10);
     findvalue3{i} = zeros(1,10);
end

for i = 1:10                            % get the U for different numbers,
    and get the first 5,10,20 SVD
    for j = 1:20
        U20{i}(:,j) = Upper{i}(:,j);
        Matrix_SVD20{i} = Matrix_SVD20{i} + Upper{i}(:,j) * Sig{i}(j,j)
    * Vig{i}(:,j)';
        if j <= 10
            U10{i}(:,j) = Upper{i}(:,j);
            Matrix_SVD10{i} = Matrix_SVD20{i} + Upper{i}(:,j) * Sig{i}(
    j,j) * Vig{i}(:,j)';
             if j <= 5
                U5{i}(:,j) = Upper{i}(:,j);
                Matrix_SVD5{i} = Matrix_SVD20{i} + Upper{i}(:,j) * Sig{
    i}(j,j) * Vig{i}(:,j)';
             end
        end
    end
end

for i = 1:2007              %get started to calculate norm
     for j = 1:10
         ResidualE1{i}{j} = zeros(1,1);
         ResidualE2{i}{j} = zeros(1,1);
```

```
113         ResidualE3{i}{j} = zeros(1,1);
114     end
115 end
116
117 %get the testzip value
118 Testvalue_5 = zeros(1,2007);
119 Testvalue_10 = zeros(1,2007);
120 Testvalue_20 = zeros(1,2007);
121 A = 0;
122 B = 0;
123 C = 0;
124
125 %get the first 5,10,20 singular vectors and get the norm value
126 for i = 1:2007
127    for j = 1:10
128           A = (eye(256) - U5{j}* U5{j}')*testzip(:,i);
129           ResidualE1{i}{j} = norm(A);
130           findvalue1{i}(1,j) = ResidualE1{i}{j};
131
132           B = (eye(256) - U10{j} * U10{j}')*testzip(:,i);
133           ResidualE2{i}{j} = norm(B);
134           findvalue2{i}(1,j) = ResidualE2{i}{j};
135
136           C = (eye(256) - U20{j} * U20{j}')*testzip(:,i);
137           ResidualE3{i}{j} = norm(C);
138           findvalue3{i}(1,j) = ResidualE3{i}{j};
139    end
140
141    Minivalue1 = min(findvalue1{i}(:));
142    Minivalue2 = min(findvalue2{i}(:));
143    Minivalue3 = min(findvalue3{i}(:));
144
145    for k = 1:10                        % find for the number with min
       value
146        if Minivalue1 == findvalue1{i}(1,k);
147            Testvalue_5(1,i) = k;
148            if k == 10;
149                Testvalue_5(1,i) = 0;
150            end
151        end
152    end
153    for n = 1:10
154        if Minivalue2 == findvalue2{i}(1,n);
155            Testvalue_10(1,i) = n;
156            if n == 10;
157                Testvalue_10(1,i) = 0;
```

```
            end
        end
    end
    for m = 1:10
        if Minivalue3 == findvalue3{i}(1,m);
            Testvalue_20(1,i) = m;
            if m == 10;
                Testvalue_20(1,i) = 0;
            end
        end
    end
end

correction_5 = 0;
correction_10 = 0;
correction_20 = 0;

%get the percentage of correction of U5,U10,U20
for i = 1:2007
    if Testvalue_5(i) == dtest(i)
        correction_5 = correction_5 + 1;
    end
    if Testvalue_10(i) == dtest(i)
        correction_10 = correction_10 + 1;
    end
    if Testvalue_20(i) == dtest(i)
        correction_20 = correction_20 + 1;
    end
end

percentage_firstfew = zeros(1,3);

percentage_firstfew(1) = correction_5 / 2007 * 100;
percentage_firstfew(2) = correction_10 / 2007 * 100;
percentage_firstfew(3) = correction_20 / 2007 * 100;

%draw the table for perentage of correctly basis
BasisImages = {'First5';'First10';'First20'};
CorrectPercentage = [percentage_firstfew(1);percentage_firstfew(2);
    percentage_firstfew(3)];
T = table(BasisImages,CorrectPercentage)

countCnum5 = zeros(1,10);
countCnum10 = zeros(1,10);
countCnum20 = zeros(1,10);

```

```
testnum = zeros(1,10);
for i = 1:2007
    if dtest(i) == 0
        testnum(10) = testnum(10) + 1;
    elseif dtest(i) == 1
        testnum(1) = testnum(1) + 1;
    elseif dtest(i) == 2
        testnum(2) = testnum(2) + 1;
    elseif dtest(i) == 3
        testnum(3) = testnum(3) + 1;
    elseif dtest(i) == 4
        testnum(4) = testnum(4) + 1;
    elseif dtest(i) == 5
        testnum(5) = testnum(5) + 1;
    elseif dtest(i) == 6
        testnum(6) = testnum(6) + 1;
    elseif dtest(i) == 7
        testnum(7) = testnum(7) + 1;
    elseif dtest(i) == 8
        testnum(8) = testnum(8) + 1;
    elseif dtest(i) == 9
        testnum(9) = testnum(9) + 1;
    end
end

for i = 1:2007
    if  dtest(i) == Testvalue_5(i)
        if dtest(i) == 0
            countCnum5(1,10) = countCnum5(1,10) + 1;
        else
            countCnum5(1,dtest(i)) = countCnum5(1,dtest(i)) + 1;
        end
    end
    if  dtest(i) == Testvalue_10(i)
        if dtest(i) == 0
            countCnum10(1,10) = countCnum10(1,10) + 1;
        else
            countCnum10(1,dtest(i)) = countCnum10(1,dtest(i)) + 1;
        end
    end
    if  dtest(i) == Testvalue_20(i)
        if dtest(i) == 0
            countCnum20(1,10) = countCnum20(1,10) + 1;
        else
            countCnum20(1,dtest(i)) = countCnum20(1,dtest(i)) + 1;
        end
```

```
    end
end

percentageMatrix = zeros(3,10);

for i = 1:10
   percentageMatrix(1,i) = countCnum5(i)/testnum(i)*100;
   percentageMatrix(2,i) = countCnum10(i)/testnum(i)*100;
   percentageMatrix(3,i) = countCnum20(i)/testnum(i)*100;
end
percentageMatrix = percentageMatrix';


%draw the table for percentage of correctly classified digits as a
%function of basis vectors
rowNames = {'1','2','3','4','5','6','7','8','9','0'};
colNames = {'First5','First10','First20'};
TTable = array2table(percentageMatrix,'RowNames',rowNames,'
    VariableNames',colNames)

%Not all digits are equally easy or difficult to classify, o is the
    easiest and 8 is the hardiest.
%From the table of the correction, I found that number 8 is most
    difficult digit to read for the computers.
%It does help to increase the number of singular vectors I used, it
    changed from 79.518% to 89.759%
```

1.2

· For the accuracy of classification, this is my table of the percentage of correctly classified digits as a function of the basis vector.

BasisImages	CorrectPercentage
'First5'	90.284
'First10'	93.174
'First20'	93.971

!! height

Figure 1: Percentage of correction with 5,10,20

This table below is the percentage correction table for different classes in different basis vectors. From the Table we can see that the more basis vectors, the larger percentage correction.

	First5	First10	First20
1	96.212	98.864	98.106
2	88.384	90.404	90.404
3	86.747	87.952	89.157
4	83.5	91.5	93.5
5	81.875	86.875	87.5
6	94.118	94.118	96.471
7	93.197	93.878	94.558
8	79.518	87.349	89.759
9	90.96	93.785	93.785
0	97.772	98.329	98.886

!! height

Figure 2: Percentage of correction with different digits in 5,10,20 basis vectors

1.3

· It is not reasonable to use different numbers of basis vectors for different classes.

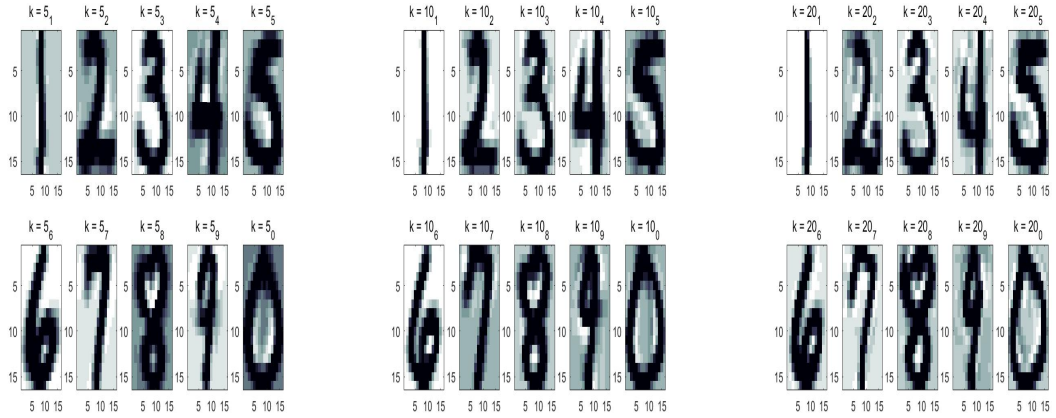
```
ans =

  1 至 9 列

  234.0415         0         0         0         0         0         0         0         0
         0   41.9309         0         0         0         0         0         0         0
         0         0   24.8492         0         0         0         0         0         0
         0         0         0   16.2948         0         0         0         0         0
         0         0         0         0   13.6110         0         0         0         0
         0         0         0         0         0   12.5218         0         0         0
         0         0         0         0         0         0   11.2288         0         0
         0         0         0         0         0         0         0   10.6685         0
         0         0         0         0         0         0         0         0    8.4134
         0         0         0         0         0         0         0         0         0

  10 列

         0
         0
         0
         0
         0
         0
         0
         0
         0
    8.3213
```

(a) singular value number1 1-10

```
ans =

  1 至 9 列

  138.2837         0         0         0         0         0         0         0         0
         0   57.3974         0         0         0         0         0         0         0
         0         0   46.3957         0         0         0         0         0         0
         0         0         0   40.9600         0         0         0         0         0
         0         0         0         0   37.5053         0         0         0         0
         0         0         0         0         0   33.5316         0         0         0
         0         0         0         0         0         0   32.2380         0         0
         0         0         0         0         0         0         0   28.4061         0
         0         0         0         0         0         0         0         0   27.1201
         0         0         0         0         0         0         0         0         0

  10 列

         0
         0
         0
         0
         0
         0
         0
         0
         0
   26.7350
```

(b) singular value number2 1-10

From the singular value we can find that the most first vectors contain the most important information. From Figure 5 - 7, we can find that for different number though the singular value is different, the more basis vectors, the more information the basis will offer.



(a) 5 basis vectors (b) 10 basis vectors (c) 20 basis vectors

Therefore, I get the same outcome for different numbers with different numbers.

1.4

· Digits are not equally easy or difficult to classify. From the table above, I find that the easiest class to identify is the 0 since it has 97.77% in first 5 basis vectors, 98.33% in first 10 basis vectors and 98.89% in the first 20 basis vectors.

The hardest number to identify is the 8, from the table, we find that it only has 79.5% correction in the first 5 basis vectors and improved a lot with 87.37% correction rate and at last it has 89.76% correction rate. Therefore, we can see very obviously that with increase of basis vector, the correcting rate increased.

2 Question 2

2.1

(a) Since we can rewrite this function as what I have showed in the picture below.

$$\min_{R^{11}} \frac{1}{n} \sum_{i=1}^{n} Z_i$$

subto

$$\mid y_i - x_1 a_1 - x_2 a_2 - ... - x_{11} a_{11} - b_1 \mid$$

$\leq \mathbf{Z}_i$

where i belongs to [1,...,n] transform this function into

$$\min_{R^{11}} \frac{1}{n} \sum_{i=1}^{n} Z_i$$

subto

$$-x_1 a_1 - x_2 a_2 - ... - x_{11} a_{11} - b_1 - Z_i$$

$\leq$ -y_1

$$x_1 a_1 + x_2 a_2 + ... + x_{11} a_{11} + b_1 - Z_i$$

$\leq$ y_1

so write this in a matrix form is

$$\min_{R^{11}} \frac{1}{n} \sum_{i=1}^{n} Z_i$$

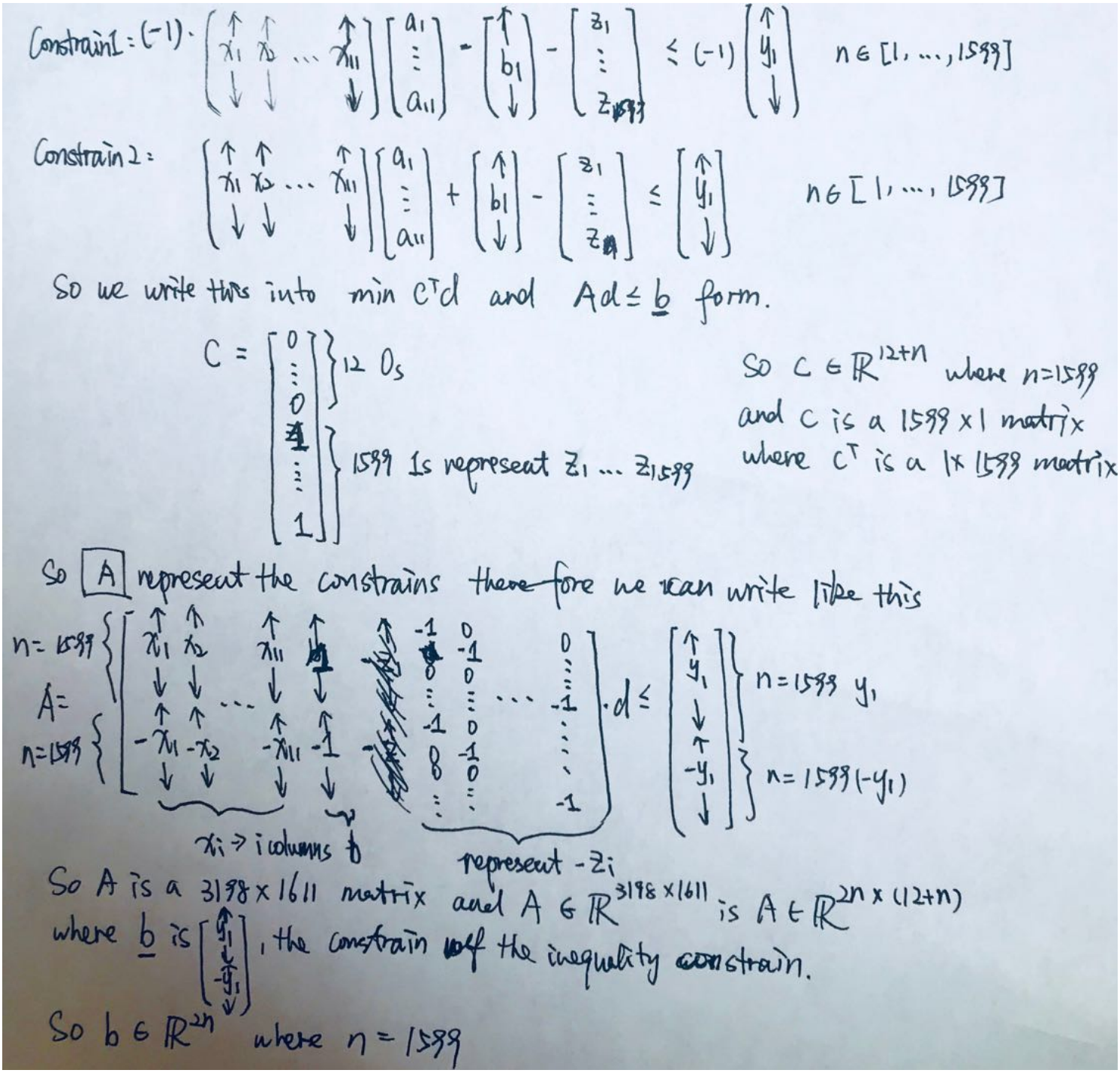


Figure 5: Matrix of transform

2.2

(b) This is my code for question 2 part1(b)

```
% Shiyu Tu
% This is a code for Question 2 part1 (b) in Final Project

GetData = readtable('winesinfo.csv');


X_i = zeros(1599,11);
Y_i = zeros(1599,1);

for i = 1:1599
    for j = 1:11
        X_i(i,j) = table2array(GetData(i,j));
    end
    Y_i(i,1) = table2array(GetData(i,12));
end

%define Matrix and Variables
a_needcol = zeros(1,11); % we use aT
b_col = zeros(3198,1);
z_col = zeros(1599,1);
b = zeros(1,1);

% This is to find the minization of (z1 + ... + z1599)
% d = [a1 ... a11 b z1 ... z1599]'

% This problem is to find cT*d
% cT = [0 * (11+1=12) ... 1(z1) ... 1(z1599)] so c is 1611*1

% A is belong to 3198 * 1611
A = zeros(3198,1611);
c_needcol = zeros(1611,1); % use cT
d = zeros(1611,1);

% putting value into A
for i = 1:11
    A(1,i) = (-1)*X_i(1,i);
end

b_col(1,1) = (-1)*Y_i(1,1);
for i = 3:2:3198
    for j = 1:11
        A(i,j) = (-1)*X_i((i-1)/2+1,j);
        b_col(i,1) = (-1)*Y_i((i-1)/2+1,1);
```

```
    end
end
for i = 2:2:3198
    for j = 1:11
        A(i,j) = X_i(i/2,j);
        b_col(i,1) = Y_i(i/2,1);
    end
end

for i = 1:1599
   for j = 13:1611
       A((j-12)*2,j) = -1;
       A((j-12)*2-1,j) = -1;
   end
end

for i = 1:3198
   if mod(i,2) == 0 %even row
       A(i,12) = 1;
   else
       A(i,12) = -1;
   end
end

%putting value into c
for i = 13:1611
   c_needcol(i,1) = 1;
end

% min         cTd
% subto       Ad <= b_col
f = c_needcol'/1599;

[d,fval] = linprog(f,A,b_col);

for i = 1:1611
    if i <= 11
        a_needcol(1,i)' == d(i,1);
    elseif i == 12
        b = d(12,1);
    elseif i > 12
        z_col(i-12,1) = d(i,1);
    end
end

d
```

```
fval
```

Since our opt value is 0.4937, which is within 1, so it is within an acceptable range

2.3

part2 (a)

```
% Shiyu Tu
% This is a code for Question 2 part2 (a) in Final Project

GetData = readtable('winesinfo.csv');

X_i = zeros(1599,11);
Y_i = zeros(1599,1);

for i = 1:1599
    for j = 1:11
        X_i(i,j) = table2array(GetData(i,j));
    end
    Y_i(i,1) = table2array(GetData(i,12));
end


n = 1599;
m = 11;
cvx_begin
    variable a(m)
    variable b(1)
    minimize(norm(Y_i - X_i * a - b))
cvx_end
```

This is my optimal value of RSS is 25.8149 and the value of b = 21.9793, and a =

$$\begin{bmatrix} 0.0250 \\ -1.0836 \\ -0.1826 \\ 0.0163 \\ -1.8742 \\ 0.0044 \\ -0.0033 \\ -17.8955 \\ -0.4136 \\ 0.9164 \\ 0.2762 \end{bmatrix}$$

2.4

part2 (b) This is my code to get the LASSO model

```
% Shiyu Tu
% This is a code for Question 2 part2 (b) in Final Project

GetData = readtable('winesinfo.csv');

X_i = zeros(1599,11);
Y_i = zeros(1599,1);

for i = 1:1599
    for j = 1:11
        X_i(i,j) = table2array(GetData(i,j));
    end
    Y_i(i,1) = table2array(GetData(i,12));
end

m = 11;
gamma_1 = 1;
gamma_2 = 0.01;
gamma_3 = 0.001;
gamma_4 = 5;

% when gamma is 1
cvx_begin
    variable a(m)
    variable b(1)
    minimize(norm(Y_i - (X_i * a + b)) + gamma_1 * norm(a,1))
cvx_end
disp(a)

% when gamma is 0.01
cvx_begin
    variable a(m)
    variable b(1)
    minimize(norm(Y_i - (X_i * a + b)) + gamma_2 * norm(a,1))
cvx_end

% when gamma is 0.001
cvx_begin
    variable a(m)
    variable b(1)
    minimize(norm(Y_i - (X_i * a + b)) + gamma_3 * norm(a,1))
cvx_end

```

```
% find the four a_value that is not 0, which are the most important
    feature

cvx_begin
    variable a(m)
    variable b(1)
    minimize(norm(Y_i - (X_i * a + b)) + gamma_4 * norm(a,1))
cvx_end
```

After running this model, I found that the only four feartures that were not 0 is the 1st, 6th, 7th and 11th feature. These non-zero features are the most important features. These features are: 1.fixedAcidity 6.freeSulfurDioxide 7.totalSulfurDioxide 11.alcohol