Assignment 4 Written Up

Graded

21 Hours, 2 Minutes Late

Student

Richard Desatnik

Total Points

138 / 140 pts

Question 1

1.a **5** / 5 pts

✔ **– 0 pts** Correct

– 3 pts Data is not normalized

– 1 pt Incorrect normalisation

– 1 pt Incorrect SD

Question 2

1.b **5** / 5 pts

✔ **– 0 pts** Correct

– 1 pt Incorrect eigen values

– 2 pts Eigen values not reported

Question 3

1.c **10** / 10 pts

✔ **– 0 pts** Correct

– 2 pts Choice of K not explained

– 3 pts Variance not reported

– 1 pt Incorrect explaination

– 2 pts Incorrect variance

Question 4

1.d **10** / 10 pts

✔ **– 0 pts** Correct

– 2 pts Graph is different

Question 5

1.e **5** / 5 pts

✔ **– 0 pts** Correct

– 2.5 pts One part unanswered

– 2 pts Incorrect covariance dimensions

– 2 pts Plot not included

– 5 pts Answer not uploaded

Question 6

2.a **5** / 5 pts

✔ **– 0 pts** Correct

– 1 pt Train/test split should be 70%/30%

– 5 pts Missing Submission

– 5 pts Missing result

Question 7

2.b **8** / 10 pts

– 0 pts Correct

✔ **– 2 pts** Missing plot of data points colored by class

– 1 pt K-Means Decision boundary plot should include centroids as well as data points

– 3 pts Missing decision boundary plot

– 10 pts Missing submission

– 1 pt Using same colours for all clusters

– 2 pts Incorrect Clusters

– 2 pts Incorrect decision boundary

– 3 pts Missing Cluster plot

– 1 pt K-means should be fit on the training set, not the entire data set

Question 8

2.c **10** / 10 pts

✔ **– 0 pts** Correct

– 3 pts Missing plot of accuracy on testing set vs number of estimators

– 10 pts Missing submission

– 2 pts Decision boundary should have data points also plotted for comparison

Question 9

2.d **5** / 5 pts

✔ **– 0 pts** Correct

– 5 pts Missing submission

Question 10

3 **40** / 40 pts

✔ **– 0 pts** Correct

Question 11

4. **35** / 35 pts

✔ **– 0 pts** Correct

– 3 pts Incorrect 1(b)

– 3 pts Incorrect 1(c)

– 3 pts Incorrect 1(d)

– 3 pts Incorrect 1(f)

– 3 pts Incorrect 2(a)

– 3 pts Incorrect 2(b)

– 3 pts Incorrect 2(c)

– 3 pts Incorrect 2(d)

– 3 pts Incorrect 2(e)

– 35 pts Missing Question

Question assigned to the following page: 1

Principal Component Analysis

The goal of this question is to build a conceptual understanding of dimensionality reduction using PCA and implement it on a toy dataset. You'll only have to use numpy and matplotlib for this question.

In [1]:
```
####This
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import time
import random
from numpy import linalg as LA
```

Question assigned to the following page: 1

In [2]:

```
####This
# (a) Load data (features)
#mean0
#std0
def load_data(npyfile):
    dataFromFile = np.load(npyfile)
    normal = np.linalg.norm(dataFromFile)
    samples, feature = dataFromFile.shape
    normlist = []
    Meanlist = []
    for feat in range(feature):
        mean = np.mean(dataFromFile[:,feat])
        std = np.std(dataFromFile[:,feat])
        featureVector = dataFromFile[:,feat]
        NormMean0Std1 = (featureVector-mean)*(1/std)
        NormMeanDataV1 = (featureVector-mean)
        print(round(np.mean(NormMean0Std1),10))
        print(round(np.std(NormMean0Std1),10))
        NormList = NormMean0Std1.tolist()
        NormMeanDataV2 = NormMeanDataV1.tolist()
        Meanlist.append(NormMeanDataV2)
        normlist.append(NormList)

    NormMeanData = np.transpose(np.array(Meanlist))
    NormMatrixTranspose = np.array(normlist)
    NormFinal = np.transpose(NormMatrixTranspose)
    print(NormFinal.shape)
    data = NormFinal
    NonNormData = dataFromFile
    return data, NormMeanData, NonNormData

feat, NormMeanData, NonNormData = load_data("features.npy")
```

```
-0.0
1.0
-0.0
1.0
0.0
1.0
-0.0
1.0
0.0
1.0
0.0
1.0
-0.0
1.0
-0.0
1.0
(150, 8)
```

Question assigned to the following page: 2

In [3]:

```
###new this
# (b) Perform eigen decomposition and return eigen pairs in desecending order of
def eigendecomp(feat):
    samples, features = feat.shape
    featT = np.transpose(feat)
    Co = np.cov(featT)
    eiganValues, eiganVectors = LA.eig(Co)
    sorted_eig_vals = eiganValues
    sorted_eig_vecs = eiganVectors
    CoMatrix = np.diag(sorted_eig_vals)
    print(sorted_eig_vals)
    print(sorted_eig_vecs)
    print(Co)
    print(CoMatrix)
    return (sorted_eig_vals, sorted_eig_vecs)

EiganValues,EiganVectors = eigendecomp(feat)
```

```
[ 4.74298961e+00  2.29585309e+00  7.76910512e-01  2.04172901e-01
  3.37651661e-02 -1.75243456e-16  2.86910531e-16 -5.44809950e-17]
[[-0.39124937  0.13884872 -0.46160937  0.58034539  0.24934936 -0.40069986
  -0.02256515  0.18043453]
 [ 0.11687696 -0.4391715  -0.78711289 -0.2905579  -0.12725786  0.01138973
   0.13157531  0.17592915]
 [-0.40655289  0.29080021 -0.13961871 -0.12636707 -0.54994554 -0.0053785
   0.38948762 -0.26576102]
 [-0.39944906  0.26454833 -0.16206048 -0.54404218  0.49904279  0.35157176
  -0.37925083 -0.36099212]
 [-0.3778555  -0.35426671  0.07790627  0.42060984  0.12822569  0.6296306
   0.03545723 -0.28352169]
 [-0.09816172 -0.64299795  0.11941452 -0.04972667 -0.0795516  -0.03130771
  -0.36166983 -0.48358818]
 [-0.45509399 -0.03231459  0.12200908 -0.08034689 -0.51935676  0.00592791
  -0.42927385  0.29290856]
 [-0.38587285 -0.30545597  0.29393481 -0.28457653  0.27864817 -0.56411724
   0.6085299   0.57923275]]
[[ 1.00671141 -0.11010327  0.87760486  0.82344326  0.61123001 -0.07220205
   0.776564    0.48190166]
 [-0.11010327  1.00671141 -0.42333835 -0.358937    0.07459068  0.52416995
  -0.28731011 -0.06998508]
 [ 0.87760486 -0.42333835  1.00671141  0.96921855  0.4704054  -0.25019762
   0.85445501  0.51042104]
 [ 0.82344326 -0.358937    0.96921855  1.00671141  0.44634059 -0.21541033
   0.8273983   0.54484244]
 [ 0.61123001  0.07459068  0.4704054   0.44634059  1.00671141  0.70151359
   0.84012249  0.93454714]
 [-0.07220205  0.52416995 -0.25019762 -0.21541033  0.70151359  1.00671141
   0.2731165   0.659988  ]
 [ 0.776564   -0.28731011  0.85445501  0.8273983   0.84012249  0.2731165
   1.00671141  0.88321462]
 [ 0.48190166 -0.06998508  0.51042104  0.54484244  0.93454714  0.659988
   0.88321462  1.00671141]]
[[ 4.74298961e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  2.29585309e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
```

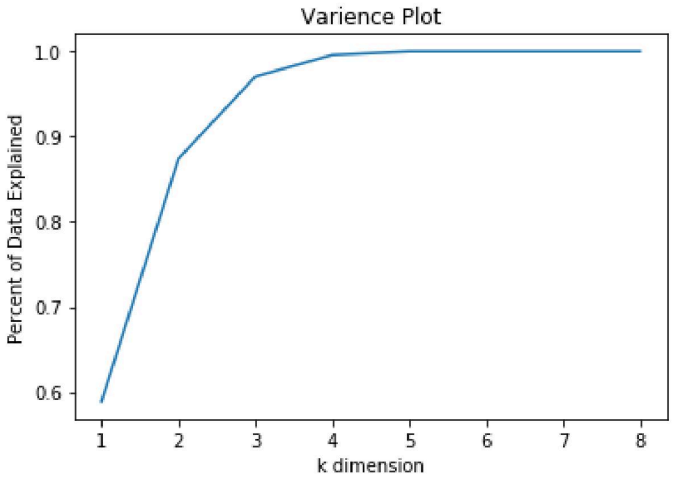
Question assigned to the following page: 2

```
[ 0.00000000e+00  0.00000000e+00  7.76910512e-01  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
[ 0.00000000e+00  0.00000000e+00  0.00000000e+00  2.04172901e-01
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
[ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  3.37651661e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00]
[ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00 -1.75243456e-16  0.00000000e+00  0.00000000e+00]
[ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  2.86910531e-16  0.00000000e+00]
[ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00 -5.44809950e-17]]
```

Question assigned to the following page: 3

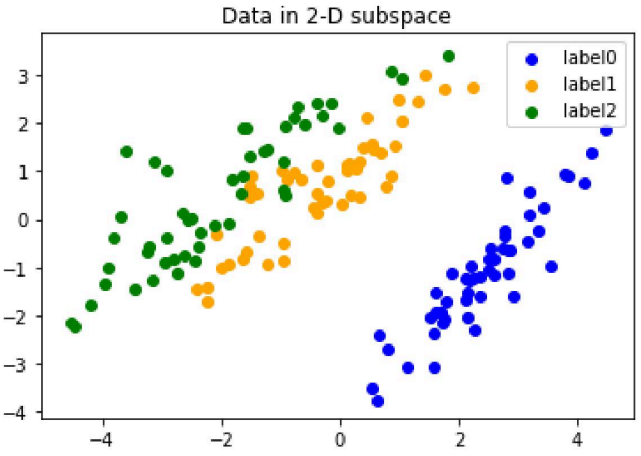
In [4]:
```
# (c) Evaluate using variance_explained as the metric
def eval(EiganVectors, EiganValues, feat):
    print(EiganVectors.shape)
    print(EiganValues)
    totvar = np.sum(EiganValues)
    samples,features = feat.shape
    dimensionlist = []
    varlist = []
    klist = []
    for feats in range(features):
        klist.append(feats+1)
        dimensionlist.append(EiganValues[feats])
        print("k = "+str(feats+1)+" Ei: "+str(EiganValues[feats])+" Variance: "+
        varlist.append((np.sum(np.array(dimensionlist)))/totvar)

    #print(varlist)
    plt.plot(klist,varlist)
    plt.xlabel("k dimension")
    plt.ylabel("Percent of Data Explained")
    plt.title("Varience Plot")
    plt.show()

eval(EiganVectors, EiganValues, feat)

#I would use k = 2 as it explains 87.4% of the data
```

```
(8, 8)
[ 4.74298961e+00  2.29585309e+00  7.76910512e-01  2.04172901e-01
  3.37651661e-02 -1.75243456e-16  2.86910531e-16 -5.44809950e-17]
k = 1 Ei: 4.742989609365717 Variance: 0.5889212098295771
k = 2 Ei: 2.2958530862227136 Variance: 0.873989634702231
k = 3 Ei: 0.7769105123670551 Variance: 0.9704560233211404
k = 4 Ei: 0.2041729011616391 Variance: 0.9958074918820439
k = 5 Ei: 0.033765166050653564 Variance: 1.0
k = 6 Ei: -1.7524345641154058e-16 Variance: 1.0
k = 7 Ei: 2.869105305177416e-16 Variance: 1.0
k = 8 Ei: -5.448099503364864e-17 Variance: 1.0
```



Question assigned to the following page: 4

In [5]:
```
# (d) Visualize after projecting to 2-D space

Matrix = feat@EiganVectors
def viz(Matrix):
    D2feat = Matrix
    label0 = D2feat[1:50,0]
    label1 = D2feat[51:100:,0]
    label2 = D2feat[101:150:,0]
    y0 = D2feat[1:50,1]
    y1 = D2feat[51:100:,1]
    y2 = D2feat[101:150:,1]

    plt.scatter(label0,y0,c="blue",label="label0")
    plt.scatter(label1,y1,c="orange",label="label1")
    plt.scatter(label2,y2,c="green",label="label2")
    plt.title("Data in 2-D subspace")
    plt.legend()
    plt.show()

viz(Matrix = Matrix)
```



Question assigned to the following page: 5

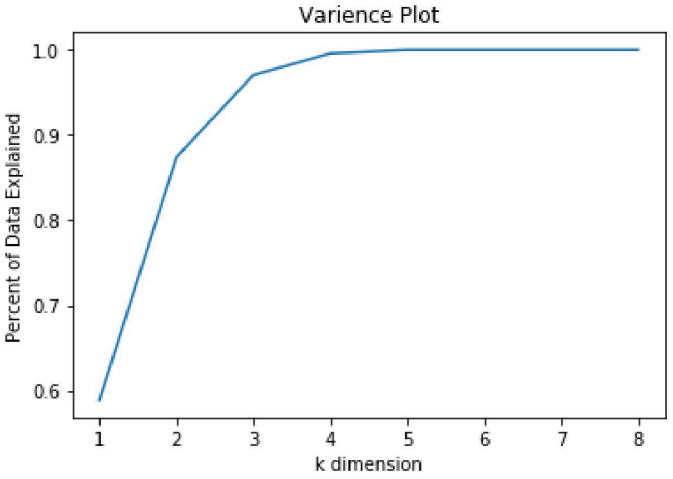
In [6]:

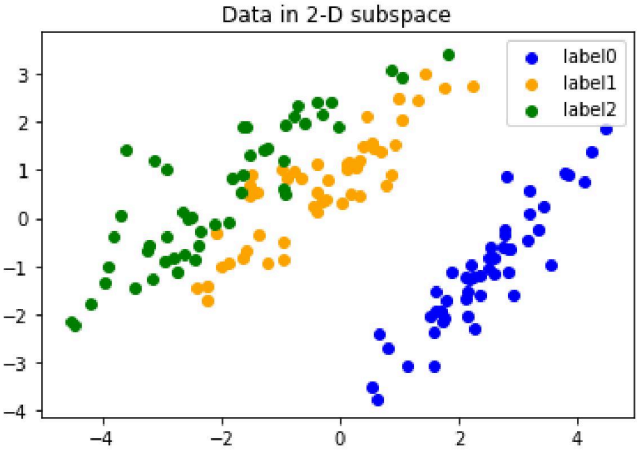
```python
def main(feat):
    EiganValues,EiganVectors = eigendecomp(feat)
    eval(EiganVectors, EiganValues, feat)
    viz(Matrix=feat@EiganVectors)

if __name__ == "__main__":
    main(feat)
```

```
[ 4.74298961e+00  2.29585309e+00  7.76910512e-01  2.04172901e-01
  3.37651661e-02 -1.75243456e-16  2.86910531e-16 -5.44809950e-17]
[[-0.39124937  0.13884872 -0.46160937  0.58034539  0.24934936 -0.40069986
  -0.02256515  0.18043453]
 [ 0.11687696 -0.4391715  -0.78711289 -0.2905579  -0.12725786  0.01138973
   0.13157531  0.17592915]
 [-0.40655289  0.29080021 -0.13961871 -0.12636707 -0.54994554 -0.0053785
   0.38948762 -0.26576102]
 [-0.39944906  0.26454833 -0.16206048 -0.54404218  0.49904279  0.35157176
  -0.37925083 -0.36099212]
 [-0.3778555  -0.35426671  0.07790627  0.42060984  0.12822569  0.6296306
   0.03545723 -0.28352169]
 [-0.09816172 -0.64299795  0.11941452 -0.04972667 -0.0795516  -0.03130771
  -0.36166983 -0.48358818]
 [-0.45509399 -0.03231459  0.12200908 -0.08034689 -0.51935676  0.00592791
  -0.42927385  0.29290856]
 [-0.38587285 -0.30545597  0.29393481 -0.28457653  0.27864817 -0.56411724
   0.6085299   0.57923275]]
[[ 1.00671141 -0.11010327  0.87760486  0.82344326  0.61123001 -0.07220205
   0.776564    0.48190166]
 [-0.11010327  1.00671141 -0.42333835 -0.358937    0.07459068  0.52416995
  -0.28731011 -0.06998508]
 [ 0.87760486 -0.42333835  1.00671141  0.96921855  0.4704054  -0.25019762
   0.85445501  0.51042104]
 [ 0.82344326 -0.358937    0.96921855  1.00671141  0.44634059 -0.21541033
   0.8273983   0.54484244]
 [ 0.61123001  0.07459068  0.4704054   0.44634059  1.00671141  0.70151359
   0.84012249  0.93454714]
 [-0.07220205  0.52416995 -0.25019762 -0.21541033  0.70151359  1.00671141
   0.2731165   0.659988  ]
 [ 0.776564   -0.28731011  0.85445501  0.8273983   0.84012249  0.2731165
   1.00671141  0.88321462]
 [ 0.48190166 -0.06998508  0.51042104  0.54484244  0.93454714  0.659988
   0.88321462  1.00671141]]
[[ 4.74298961e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  2.29585309e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  7.76910512e-01  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  2.04172901e-01
   0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   3.37651661e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00 -1.75243456e-16  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  2.86910531e-16  0.00000000e+00]
```

Question assigned to the following page: 5

```
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00  0.00000000e+00  0.00000000e+00 -5.44809950e-17]]
(8, 8)
[ 4.74298961e+00  2.29585309e+00  7.76910512e-01  2.04172901e-01
  3.37651661e-02 -1.75243456e-16  2.86910531e-16 -5.44809950e-17]
k = 1 Ei: 4.742989609365717 Variance: 0.5889212098295771
k = 2 Ei: 2.2958530862227136 Variance: 0.873989634702231
k = 3 Ei: 0.7769105123670551 Variance: 0.9704560233211404
k = 4 Ei: 0.2041729011616391 Variance: 0.9958074918820439
k = 5 Ei: 0.033765166050653564 Variance: 1.0
k = 6 Ei: -1.7524345641154058e-16 Variance: 1.0
k = 7 Ei: 2.869105305177416e-16 Variance: 1.0
k = 8 Ei: -5.448099503364864e-17 Variance: 1.0
```





(e1): If the number of features is 1000 and the number of data points is 10, what will be the dimension of your covariance matrix? Can you suggest what can be changed to improve the performance?

```
#The covariance matrix would be 1000 by 1000
```

Question assigned to the following page: 5

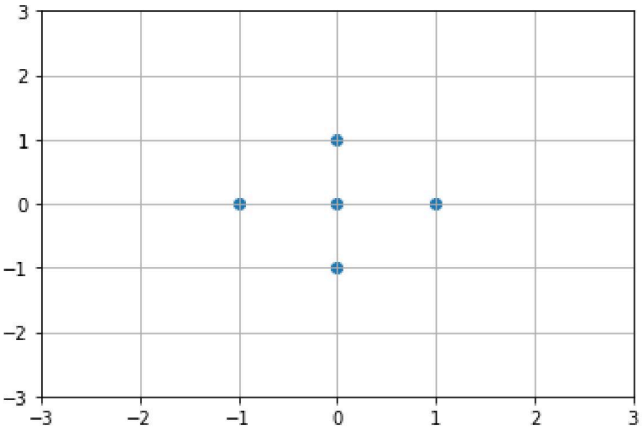
```
#To improve performance reduce number of features by using the eiganvectors
with the highest eiganvalue
```

```
(e2): Assume you have a dataset with the original dimensionality as 2 and you
have to reduce it to 1. Provide a sample scatter plot of the original data
(less than 10 datapoints) where PCA might produce misleading results. You can
plot it by hand and then take a picture. In the next cell, switch to Markdown
mode and use the command: ![title](<your_plot_file_path>)
```

```
PCA breaks down in conditions of high kurtosis as seen when there is no
correlation between points and there is just a circular cluster
badData = np.array([[1,2],[4,5],[7,8],[10,11]])
x = badData[:,0]
y = badData[:,1]
print(badData.shape)
```

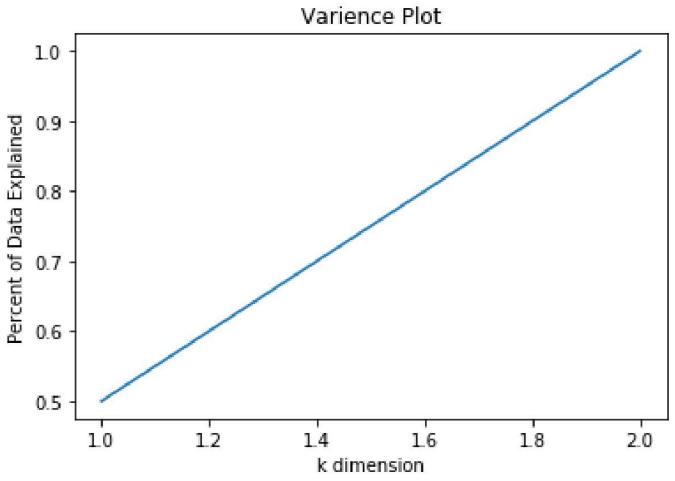
Question assigned to the following page: 5

In [7]:
```
badData = np.array([[1,0],[0,-1],[0,1],[-1,0],[0,0]])
x = badData[:,0]
y = badData[:,1]
plt.scatter(x,y)
plt.xlim([-3,3])
plt.ylim([-3,3])
plt.grid()
plt.show()
print(badData.shape)
EiganValues_bad,EiganVectors_bad = eigendecomp(badData)
eval(EiganVectors_bad, EiganValues_bad, badData)
k=1
dimMatrixT = []

for dim in range(k):
    dimMatrixT.append(EiganVectors_bad[dim])

dimMatrix = np.transpose(dimMatrixT)
D1feat = badData@dimMatrix
print(D1feat.shape)
label0 = D1feat[:,0]
y0 = np.array([0,0,0,0,0])
print(label0)
print(y0)
plt.scatter(label0,y0,c="blue",label="label0")
plt.title("Data in 1-D subspace")
plt.show()
print(badData.shape)
```

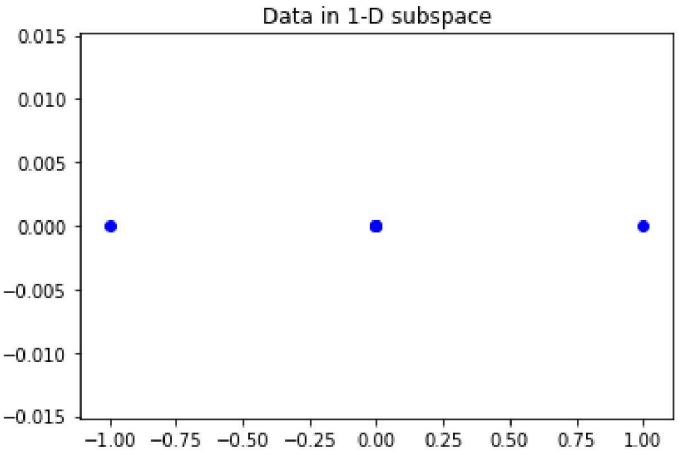



```
(5, 2)
[0.5 0.5]
[[1. 0.]
 [0. 1.]]
[[0.5 0. ]
 [0.  0.5]]
[[0.5 0. ]
 [0.  0.5]]
(2, 2)
[0.5 0.5]
k = 1 Ei: 0.5 Variance: 0.5
k = 2 Ei: 0.5 Variance: 1.0
```

Question assigned to the following page: 5




```
(5, 1)
[ 1.  0.  0. -1.  0.]
[0 0 0 0 0]
```




```
(5, 2)
```

In [ ]:

In [ ]:

Question assigned to the following page: 5

This problem was adapted from Professor Farimani's paper. If you are interested in learning more, you can read it here (https://www.nature.com/articles/s41699-018-0060-8.pdf).

Question assigned to the following page: 6

In [1]:
```
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import time
import random
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:30: DeprecationWarning: `np.float` is a deprecated alias for the builtin `f
loat`. To silence this warning, use `float` by itself. Doing this will not mo
dify any behavior and is safe. If you specifically wanted the numpy scalar ty
pe, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  method='lar', copy_X=True, eps=np.finfo(np.float).eps,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:167: DeprecationWarning: `np.float` is a deprecated alias for the builtin `
float`. To silence this warning, use `float` by itself. Doing this will not m
odify any behavior and is safe. If you specifically wanted the numpy scalar t
ype, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  method='lar', copy_X=True, eps=np.finfo(np.float).eps,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:284: DeprecationWarning: `np.float` is a deprecated alias for the builtin `
float`. To silence this warning, use `float` by itself. Doing this will not m
odify any behavior and is safe. If you specifically wanted the numpy scalar t
ype, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_Gram=True, verbose=0,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:862: DeprecationWarning: `np.float` is a deprecated alias for the builtin `
float`. To silence this warning, use `float` by itself. Doing this will not m
odify any behavior and is safe. If you specifically wanted the numpy scalar t
ype, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1101: DeprecationWarning: `np.float` is a deprecated alias for the builtin
 `float`. To silence this warning, use `float` by itself. Doing this will not
modify any behavior and is safe. If you specifically wanted the numpy scalar
 type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
```

Question assigned to the following page: 6

```
y:1127: DeprecationWarning: `np.float` is a deprecated alias for the builtin
 `float`. To silence this warning, use `float` by itself. Doing this will not
modify any behavior and is safe. If you specifically wanted the numpy scalar
 type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, positive=False):
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1362: DeprecationWarning: `np.float` is a deprecated alias for the builtin
 `float`. To silence this warning, use `float` by itself. Doing this will not
modify any behavior and is safe. If you specifically wanted the numpy scalar
 type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1602: DeprecationWarning: `np.float` is a deprecated alias for the builtin
 `float`. To silence this warning, use `float` by itself. Doing this will not
modify any behavior and is safe. If you specifically wanted the numpy scalar
 type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1738: DeprecationWarning: `np.float` is a deprecated alias for the builtin
 `float`. To silence this warning, use `float` by itself. Doing this will not
modify any behavior and is safe. If you specifically wanted the numpy scalar
 type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, positive=False):
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\decomposition\online_lda.p
y:29: DeprecationWarning: `np.float` is a deprecated alias for the builtin `f
loat`. To silence this warning, use `float` by itself. Doing this will not mo
dify any behavior and is safe. If you specifically wanted the numpy scalar ty
pe, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  EPS = np.finfo(np.float).eps
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\ensemble\gradient_boostin
g.py:32: DeprecationWarning: `np.bool` is a deprecated alias for the builtin
 `bool`. To silence this warning, use `bool` by itself. Doing this will not m
odify any behavior and is safe. If you specifically wanted the numpy scalar t
ype, use `np.bool_` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  from ._gradient_boosting import predict_stages
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\ensemble\gradient_boostin
g.py:32: DeprecationWarning: `np.bool` is a deprecated alias for the builtin
 `bool`. To silence this warning, use `bool` by itself. Doing this will not m
odify any behavior and is safe. If you specifically wanted the numpy scalar t
```

Question assigned to the following page: 6

```
ype, use `np.bool_` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  from ._gradient_boosting import predict_stages
```

In [2]:
```
# (a)
#x in the shape of (2000,2)
#y in the shape of 2000
# data preprocessing
DNAData = pd.read_csv("data.csv")
DNANames = DNAData.columns.values
DNADataNp = DNAData.to_numpy()
samples, labels = DNADataNp.shape
x1 = []
x2 = []
labellist = []
for label in range(0,labels,2):
    for sample in range(samples):
        x1.append(DNADataNp[:,label][sample])
        x2.append(DNADataNp[:,label+1][sample])
        labellist.append(DNANames[label+1])
DNADataReshapeT = np.array([x1,x2])
DNADataReshape = np.transpose(DNADataReshapeT)
labelnp = np.transpose(np.array(labellist))
print(DNADataReshape.shape)
print(labelnp.shape)
```

```
(2000, 2)
(2000,)
```

In [3]:
```
#Splitting data
DNADataReshape.shape
labelnp.shape
X_train, X_test, y_train, y_test = train_test_split(DNADataReshape, labelnp, tes
```

Question assigned to the following page: 7

In [4]:

```
# (b)
# k-means
X_train
X_test
y_train
y_test
clustering = KMeans(n_clusters=20, random_state=24787)
clustering.fit(X_train,y_train)


h = 1

x_min, x_max = X_train[:, 0].min() - 1, X_train[:, 0].max() + 1
y_min, y_max = X_train[:, 1].min() - 1, X_train[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
```

Question assigned to the following page: 7

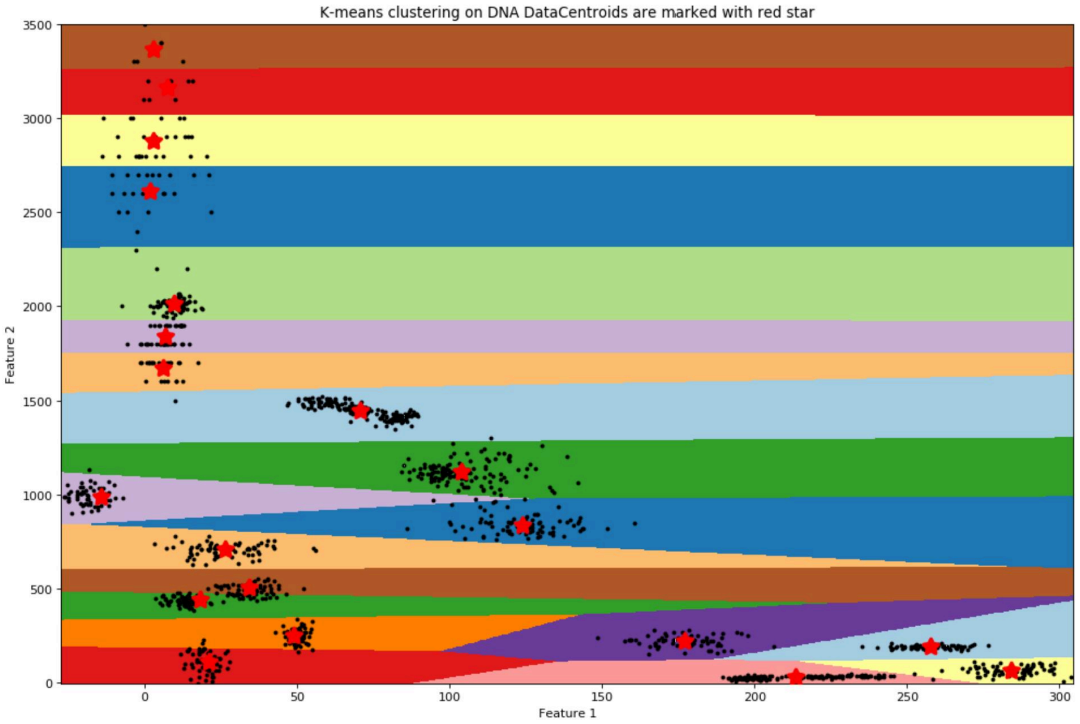
In [5]:

```
from matplotlib.pyplot import figure
figure(figsize=(15, 10), dpi=80)
Z = clustering.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
plt.figure(1)
plt.clf()
plt.imshow(
    Z,
    interpolation="nearest",
    extent=(xx.min(), xx.max(), yy.min(), yy.max()),
    cmap=plt.cm.Paired,
    aspect="auto",
    origin="lower",
)

plt.plot(X_train[:, 0], X_train[:, 1], "k.", markersize=5)
centroids = clustering.cluster_centers_
plt.scatter(
    centroids[:, 0],
    centroids[:, 1],
    marker=(5,1),
    s=169,
    linewidths=3,
    color="red",
    zorder=10,
)
plt.title(
    "K-means clustering on DNA Data"
    "Centroids are marked with red star"
)

plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.show()
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
```

Question assigned to the following page: 7

```
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
```



Question assigned to the following page: 7

In [6]:

```
OneHot = y_train
length, = OneHot.shape
OneHotLab = []

for index in range(length):
    #print(OneHot[index])
    if OneHot[index] == "CYS":
        OneHotLab.append(0)
    elif OneHot[index] == "LEU":
        OneHotLab.append(1)
    elif OneHot[index] == "HIS":
        OneHotLab.append(2)
    elif OneHot[index] == "THR":
        OneHotLab.append(3)
    elif OneHot[index] == "GLN":
        OneHotLab.append(4)
    elif OneHot[index] == "MET":
        OneHotLab.append(5)
    elif OneHot[index] == "ARG":
        OneHotLab.append(6)
    elif OneHot[index] == "VAL":
        OneHotLab.append(7)
    elif OneHot[index] == "LYS":
        OneHotLab.append(8)
    elif OneHot[index] == "ALA":
        OneHotLab.append(9)
    elif OneHot[index] == "SER":
        OneHotLab.append(10)
    elif OneHot[index] == "GLU":
        OneHotLab.append(11)
    elif OneHot[index] == "GLY":
        OneHotLab.append(12)
    elif OneHot[index] == "ISO":
        OneHotLab.append(13)
    elif OneHot[index] == "TYR":
        OneHotLab.append(14)
    elif OneHot[index] == "ASN":
        OneHotLab.append(15)
    elif OneHot[index] == "PRO":
        OneHotLab.append(16)
    elif OneHot[index] == "TRP":
        OneHotLab.append(17)
    elif OneHot[index] == "PHE":
        OneHotLab.append(18)
    elif OneHot[index] == "ASP":
        OneHotLab.append(19)
    else:
        OneHotLab.append(20)
        print(OneHot[index])

Y_train = np.array(OneHotLab)
```

Question assigned to the following page: 7

In [7]:

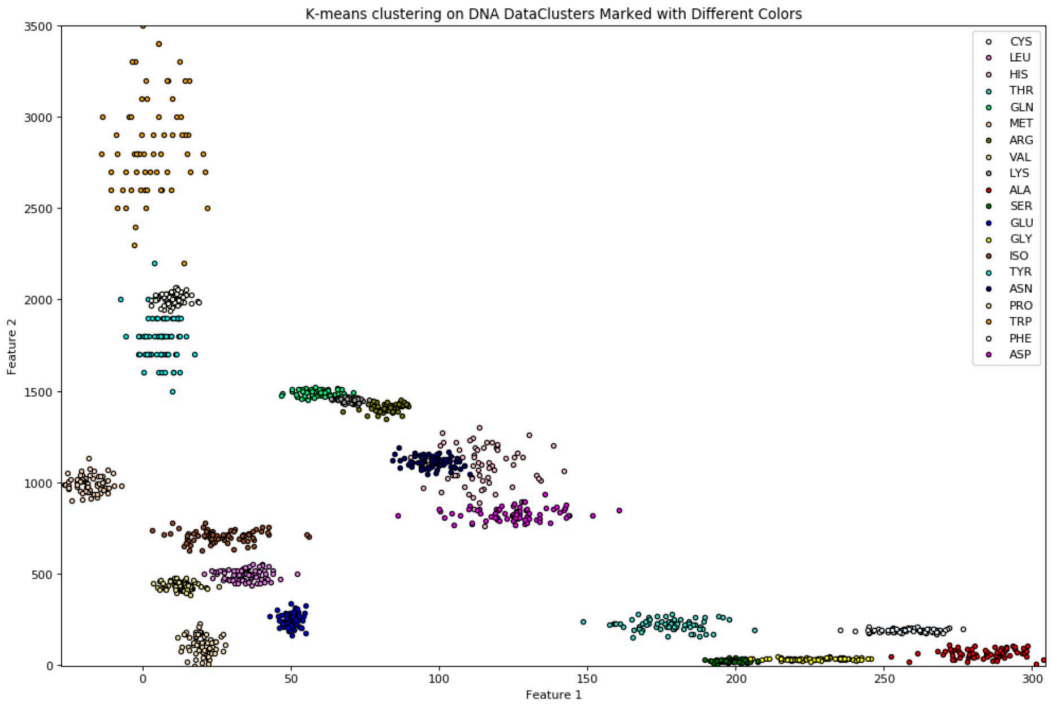
```
from matplotlib.pyplot import figure
figure(figsize=(15, 10), dpi=80)
Z = clustering.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
plt.figure(1)
plt.clf()
X = X_train
y = Y_train
n_classes = 20
plot_colors = ["aliceblue","violet","pink","turquoise","springgreen","peachpuff"
labelsnp=np.array(["CYS","LEU","HIS","THR","GLN","MET","ARG","VAL","LYS","ALA","S
for i, color in zip(range(n_classes), plot_colors):
    print(i)
    print(color)
    idx = np.where(y == i)
    plt.scatter(
        X[idx, 0],
        X[idx, 1],
        c=color,
        label=labelsnp[i],
        edgecolor="black",
        s=15,
    )

plt.title(
    "K-means clustering on DNA Data"
    "Clusters Marked with Different Colors"
)
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.legend()
plt.show()
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\metrics\pairwise.py:56: De
precationWarning: `np.float` is a deprecated alias for the builtin `float`. T
```

Question assigned to the following page: 7

```
o silence this warning, use `float` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use `
np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  dtype = np.float

0
aliceblue
1
violet
2
pink
3
turquoise
4
springgreen
5
peachpuff
6
olive
7
khaki
8
darkgrey
9
red
10
green
11
blue
12
yellow
13
sienna
14
aqua
15
navy
16
moccasin
17
orange
18
ivory
19
magenta
```

Questions assigned to the following page: 7 and 8



In [8]:
```
# (c)
# random forest
clf = RandomForestClassifier(n_estimators=100)
clf.fit(X_train, y_train)
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:489: De
precationWarning: `np.int` is a deprecated alias for the builtin `int`. To si
lence this warning, use `int` by itself. Doing this will not modify any behav
ior and is safe. When replacing `np.int`, you may wish to use e.g. `np.int64`
or `np.int32` to specify the precision. If you wish to review your current us
e, check the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_store_unique_indices = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
```

Question assigned to the following page: 8

In [9]:
```
#Random Forest iteration
Est_num = 24
scorelist = []
N_estimator_list = []
for Num_est in range(Est_num):
    print(Num_est)
    tree = RandomForestClassifier(n_estimators=(Num_est+1))
    tree.fit(X_train, y_train)
    score = tree.score(X_test, y_test)
    scorelist.append(score)
    N_estimator_list.append(Num_est+1)
    print(score)
N_estimator = np.array(N_estimator_list)
score = np.array(scorelist)

plt.plot(N_estimator, score)
plt.xlabel("N_estimator in RandomForest")
plt.ylabel("Accuracy")
plt.title("Random Forest Accuracy 0 to 25")
plt.show()
```

```
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
```

Question assigned to the following page: 8

In [10]:
```
clf = RandomForestClassifier(n_estimators=100)
clf.fit(X_train, y_train)
print(clf.score(X_train,y_train))
score = clf.score(X_test, y_test)
print(score)
Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
```

In [11]:
```
print(clf.score(X_train,y_train))
print(score)
```

```
1.0
0.9783333333333334

C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\ensemble\base.py:158: Deprec
ationWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
this warning, use `int` by itself. Doing this will not modify any behavior and
is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.in
t32` to specify the precision. If you wish to review your current use, check th
e release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devd
ocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/release/
1.20.0-notes.html#deprecations)
  dtype=np.int)
```

Question assigned to the following page: 8

In [12]:

```
OneHot = y_train
length, = OneHot.shape
OneHotLab = []

for index in range(length):
    #print(OneHot[index])
    if OneHot[index] == "CYS":
        OneHotLab.append(0)
    elif OneHot[index] == "LEU":
        OneHotLab.append(1)
    elif OneHot[index] == "HIS":
        OneHotLab.append(2)
    elif OneHot[index] == "THR":
        OneHotLab.append(3)
    elif OneHot[index] == "GLN":
        OneHotLab.append(4)
    elif OneHot[index] == "MET":
        OneHotLab.append(5)
    elif OneHot[index] == "ARG":
        OneHotLab.append(6)
    elif OneHot[index] == "VAL":
        OneHotLab.append(7)
    elif OneHot[index] == "LYS":
        OneHotLab.append(8)
    elif OneHot[index] == "ALA":
        OneHotLab.append(9)
    elif OneHot[index] == "SER":
        OneHotLab.append(10)
    elif OneHot[index] == "GLU":
        OneHotLab.append(11)
    elif OneHot[index] == "GLY":
        OneHotLab.append(12)
    elif OneHot[index] == "ISO":
        OneHotLab.append(13)
    elif OneHot[index] == "TYR":
        OneHotLab.append(14)
    elif OneHot[index] == "ASN":
        OneHotLab.append(15)
    elif OneHot[index] == "PRO":
        OneHotLab.append(16)
    elif OneHot[index] == "TRP":
        OneHotLab.append(17)
    elif OneHot[index] == "PHE":
        OneHotLab.append(18)
    elif OneHot[index] == "ASP":
        OneHotLab.append(19)
    else:
        OneHotLab.append(20)
        print(OneHot[index])

Y_train = np.array(OneHotLab)
```

Question assigned to the following page: 8

In [13]:
```
import matplotlib.pyplot as plt
import matplotlib.cm as mplcm
import matplotlib.colors as colors
import numpy as np
import matplotlib.colors as mcolors
```

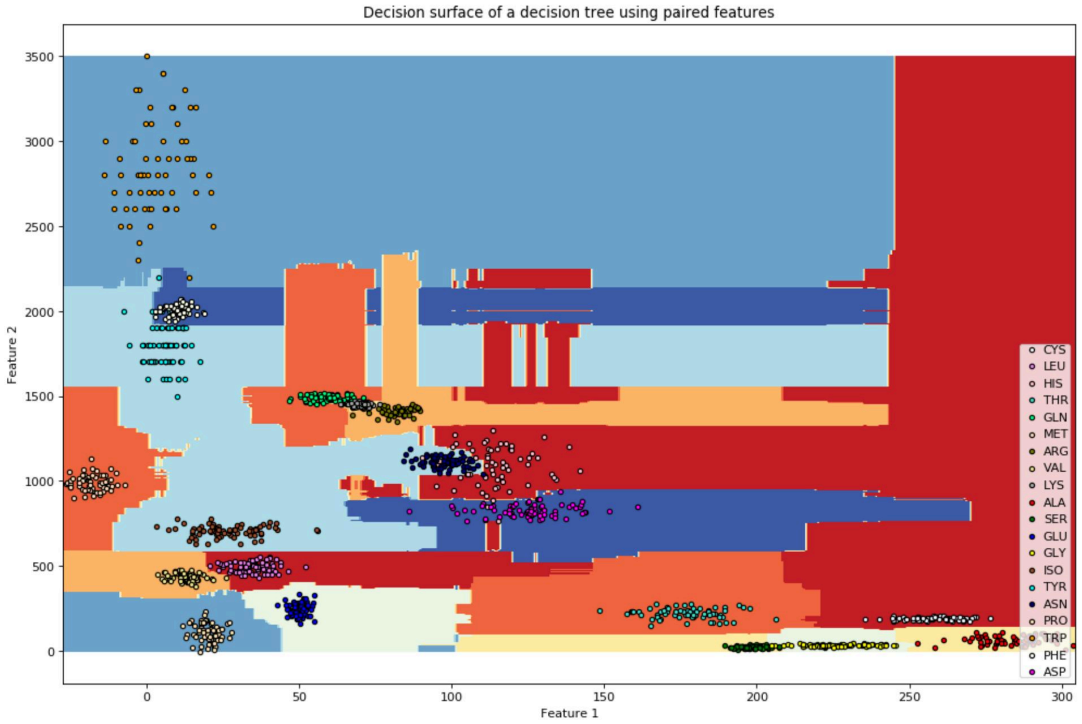
In [14]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors


n_classes = 20
plot_colors = ["aliceblue","violet","pink","turquoise","springgreen","peachpuff"
plot_step = 1

X = X_train
y = Y_train

# Train
clf = RandomForestClassifier(n_estimators=100)
clf.fit(X_train, Y_train)

x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, plot_step), np.arange(y_min, y_max,

Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
```

```
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
 this warning, use `int` by itself. Doing this will not modify any behavior a
nd is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `
np.int32` to specify the precision. If you wish to review your current use, c
heck the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/de
vdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations)
  y_encoded = np.zeros(y.shape, dtype=np.int)
C:\Users\rdesa\Anaconda3\lib\site-packages\sklearn\tree\tree.py:163: Deprecat
ionWarning: `np.int` is a deprecated alias for the builtin `int`. To silence
```

Question assigned to the following page: 8

In [15]:

```
figure(figsize=(15, 10), dpi=80)
cs = plt.contourf(xx, yy, Z, cmap=plt.cm.RdYlBu)

plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
labelsnp = np.array(["CYS","LEU","HIS","THR","GLN","MET","ARG","VAL","LYS","ALA"
# Plot the training points
for i, color in zip(range(n_classes), plot_colors):
    #print(i)
    idx = np.where(y == i)
    plt.scatter(
        X[idx, 0],
        X[idx, 1],
        c=color,
        label=labelsnp[i],
        cmap=plt.cm.RdYlBu,
        edgecolor="black",
        s=15,
    )
plt.title("Decision surface of a decision tree using paired features")
plt.legend(loc="lower right", borderpad=0, handletextpad=0)
plt.show()
```



Question assigned to the following page: 9

In [16]:
```
# (d)
# Analysis
#The K means algorithm produces long straight lines for
#decision boundaries that are nearly perpendicular to the
#to a line made between two centroids. This makes sense
#as the decision boundary is seperating the groups based
# on the centroids of the custer
#K means also produces smoother decision boundaries
#The K random forest can only produce straight line
#decision boundaries parallel to the axes
#The accuracy of the k random forest is high even for the
#test data being = to 97%
```

In [ ]:

In [ ]:

No questions assigned to the following page.

Question assigned to the following page: 10

In [1]:

```
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import time
import random
from sklearn.model_selection import train_test_split
from tensorflow import keras
import tensorflow as tf
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import InputLayer, Conv2D, MaxPooling2D
from keras import backend as K
import matplotlib.pyplot as plt
```

```
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:523: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:524: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:525: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:526: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorflow\python\framework\dtype
s.py:532: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is
deprecated; in a future version of numpy, it will be understood as (type,
(1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
```

Question assigned to the following page: 10

```
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:543: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:544: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:545: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\rdesa\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub
\dtypes.py:550: FutureWarning: Passing (type, 1) or '1type' as a synonym of t
ype is deprecated; in a future version of numpy, it will be understood as (ty
pe, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
Using TensorFlow backend.
```

In [2]:
```
# data preprocessing
DNAData = pd.read_csv("data.csv")
DNANames = DNAData.columns.values
DNADataNp = DNAData.to_numpy()
samples, labels = DNADataNp.shape
x1 = []
x2 = []
labellist = []
for label in range(0,labels,2):
    for sample in range(samples):
        x1.append(DNADataNp[:,label][sample])
        x2.append(DNADataNp[:,label+1][sample])
        labellist.append(DNANames[label+1])
DNADataReshapeT = np.array([x1,x2])
DNADataReshape = np.transpose(DNADataReshapeT)
labelnp = np.transpose(np.array(labellist))
print(DNADataReshape.shape)
print(labelnp.shape)
```

```
(2000, 2)
(2000,)
```

In [3]:
```
#Splitting data
DNADataReshape.shape
labelnp.shape
X_train, X_test, y_train, y_test = train_test_split(DNADataReshape, labelnp, tes
```

Question assigned to the following page: 10

In [4]:

```
OneHot = y_train
length, = OneHot.shape
OneHotLab = []

for index in range(length):
    #print(OneHot[index])
    if OneHot[index] == "CYS":
        OneHotLab.append(0)
    elif OneHot[index] == "LEU":
        OneHotLab.append(1)
    elif OneHot[index] == "HIS":
        OneHotLab.append(2)
    elif OneHot[index] == "THR":
        OneHotLab.append(3)
    elif OneHot[index] == "GLN":
        OneHotLab.append(4)
    elif OneHot[index] == "MET":
        OneHotLab.append(5)
    elif OneHot[index] == "ARG":
        OneHotLab.append(6)
    elif OneHot[index] == "VAL":
        OneHotLab.append(7)
    elif OneHot[index] == "LYS":
        OneHotLab.append(8)
    elif OneHot[index] == "ALA":
        OneHotLab.append(9)
    elif OneHot[index] == "SER":
        OneHotLab.append(10)
    elif OneHot[index] == "GLU":
        OneHotLab.append(11)
    elif OneHot[index] == "GLY":
        OneHotLab.append(12)
    elif OneHot[index] == "ISO":
        OneHotLab.append(13)
    elif OneHot[index] == "TYR":
        OneHotLab.append(14)
    elif OneHot[index] == "ASN":
        OneHotLab.append(15)
    elif OneHot[index] == "PRO":
        OneHotLab.append(16)
    elif OneHot[index] == "TRP":
        OneHotLab.append(17)
    elif OneHot[index] == "PHE":
        OneHotLab.append(18)
    elif OneHot[index] == "ASP":
        OneHotLab.append(19)
    else:
        OneHotLab.append(20)
        print(OneHot[index])

Y_train = np.array(OneHotLab)
```

Question assigned to the following page: 10

In [5]:

```
OneHot = y_test
length, = OneHot.shape
OneHotLab = []

for index in range(length):
    #print(OneHot[index])
    if OneHot[index] == "CYS":
        OneHotLab.append(0)
    elif OneHot[index] == "LEU":
        OneHotLab.append(1)
    elif OneHot[index] == "HIS":
        OneHotLab.append(2)
    elif OneHot[index] == "THR":
        OneHotLab.append(3)
    elif OneHot[index] == "GLN":
        OneHotLab.append(4)
    elif OneHot[index] == "MET":
        OneHotLab.append(5)
    elif OneHot[index] == "ARG":
        OneHotLab.append(6)
    elif OneHot[index] == "VAL":
        OneHotLab.append(7)
    elif OneHot[index] == "LYS":
        OneHotLab.append(8)
    elif OneHot[index] == "ALA":
        OneHotLab.append(9)
    elif OneHot[index] == "SER":
        OneHotLab.append(10)
    elif OneHot[index] == "GLU":
        OneHotLab.append(11)
    elif OneHot[index] == "GLY":
        OneHotLab.append(12)
    elif OneHot[index] == "ISO":
        OneHotLab.append(13)
    elif OneHot[index] == "TYR":
        OneHotLab.append(14)
    elif OneHot[index] == "ASN":
        OneHotLab.append(15)
    elif OneHot[index] == "PRO":
        OneHotLab.append(16)
    elif OneHot[index] == "TRP":
        OneHotLab.append(17)
    elif OneHot[index] == "PHE":
        OneHotLab.append(18)
    elif OneHot[index] == "ASP":
        OneHotLab.append(19)
    else:
        OneHotLab.append(20)
        print(OneHot[index])

Y_test = np.array(OneHotLab)
```

Question assigned to the following page: 10

In [6]:

```
X_train = X_train.reshape(X_train.shape[0], 2, 1)
print(X_train.shape)
num_classes = 20
Y_train = keras.utils.to_categorical(Y_train, num_classes)
print(Y_train.shape)
X_test = X_test.reshape(X_test.shape[0], 2, 1)
print(X_test.shape)
num_classes = 20
Y_test = keras.utils.to_categorical(Y_test, num_classes)
print(Y_test.shape)
input_shape = (2,1)
```

```
(1400, 2, 1)
(1400, 20)
(600, 2, 1)
(600, 20)
```

Question assigned to the following page: 10

In [7]:
```
# Sequential model is a linear stack of layers
input_shape = (2,1)
num_classes= 20
model = Sequential()

# Our input is a 28 by 28 image/matrix, in implementation (28x28x1)
model.add(InputLayer(input_shape=input_shape))
model.add(Flatten())
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])
print(model.summary())
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten_1 (Flatten)          (None, 2)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 200)               600
_________________________________________________________________
dense_2 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_3 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_4 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_5 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_6 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_7 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_8 (Dense)              (None, 200)               40200
_________________________________________________________________
dense_9 (Dense)              (None, 200)               40200
_________________________________________________________________
```

Question assigned to the following page: 10

```
dense_10 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_11 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_12 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_13 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_14 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_15 (Dense)             (None, 200)               40200
_________________________________________________________________
dense_16 (Dense)             (None, 200)               40200
_________________________________________________________________
dropout_1 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_17 (Dense)             (None, 20)                4020
=================================================================
Total params: 607,620
Trainable params: 607,620
Non-trainable params: 0
_________________________________________________________________
None
```

Question assigned to the following page: 10

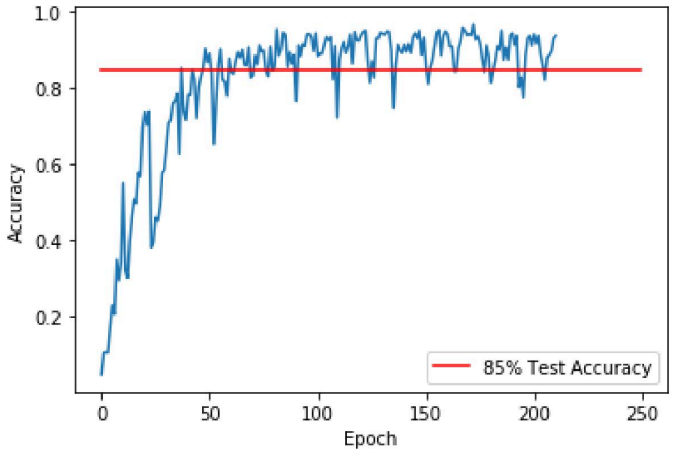
In [8]:

```
callback = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=50)
batch_size = 200
epochs = 400
# train network by calling fit function
history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=epochs,callbacks=[callback],
          verbose=1,
          validation_data=(X_test, Y_test))

# evaluate the accuracy of trained model using the testing data
score = model.evaluate(X_test, Y_test, verbose=1)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
curacy: 0.8586 - val_loss: 0.2473 - val_accuracy: 0.9317
Epoch 199/400
1400/1400 [==============================] - 0s 44us/step - loss: 0.3107 - ac
curacy: 0.8843 - val_loss: 0.2296 - val_accuracy: 0.9383
Epoch 200/400
1400/1400 [==============================] - 0s 44us/step - loss: 0.2638 - ac
curacy: 0.8957 - val_loss: 0.2587 - val_accuracy: 0.9133
Epoch 201/400
1400/1400 [==============================] - 0s 45us/step - loss: 0.2281 - ac
curacy: 0.9064 - val_loss: 0.1870 - val_accuracy: 0.9417
Epoch 202/400
1400/1400 [==============================] - 0s 44us/step - loss: 0.2145 - ac
curacy: 0.9171 - val_loss: 0.1960 - val_accuracy: 0.9183
Epoch 203/400
1400/1400 [==============================] - 0s 44us/step - loss: 0.2006 - ac
curacy: 0.9250 - val_loss: 0.1788 - val_accuracy: 0.9383
Epoch 204/400
1400/1400 [==============================] - 0s 44us/step - loss: 0.2540 - ac
curacy: 0.9021 - val_loss: 0.2856 - val_accuracy: 0.8900
Epoch 205/400
```

Question assigned to the following page: 10

In [9]:
```
plt.plot(history.history['val_accuracy'])
x = np.array(range(0,250))
y = np.ones(250)*0.85
plt.plot(x,y,c="red",label="85% Test Accuracy")
plt.ylabel("Accuracy")
plt.xlabel("Epoch")
plt.legend()
plt.show()
```



In [10]:
```
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Test loss: 0.1773594323794047
Test accuracy: 0.9383333325386047
```

In [ ]:
```
```

Question assigned to the following page: 11

Problem 4

Forward Prop

$$\underset{4\times 6}{\begin{bmatrix} 1 & 2 & -3 & 0 & 1 & -3 \\ 3 & 1 & 2 & 1 & 0 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 0 & 2 & 1 & -2 & 2 \end{bmatrix}} \underset{6\times 1}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}} = \begin{bmatrix} (1)(1)+(1)(2)+0(-3)+0(0)+(1)(1)+(-3)(1) \\ (1)(3)+(1)(1)+0(2)+0(1)+1(0)+1(2) \\ (1)(2)+(1)(2)+0(2)+0(2)+1(2)+1(1) \\ (1)(1)+(1)(0)+0(2)+0(1)+1(-2)+1(2) \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} 1 \\ 6 \\ 7 \\ 1 \end{bmatrix}}_{+} \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}}_{\text{bias}} = \underbrace{\begin{bmatrix} 2 \\ 7 \\ 8 \\ 2 \end{bmatrix}}_{a\text{ vector}} \rightarrow \underbrace{\begin{bmatrix} \frac{1}{1+\exp(-2)} \\ \frac{1}{1+\exp(-7)} \\ \frac{1}{1+\exp(-8)} \\ \frac{1}{1+\exp(-2)} \end{bmatrix}}_{z\text{ vector}} = \underbrace{\begin{bmatrix} 0.8808 \\ 0.9991 \\ 0.9997 \\ 0.8808 \end{bmatrix}}_{z\text{ vector}}$$

1a. $a_1 = 2$ $\quad z_1 = 0.8808$

1b. $a_3 = 8$ $\quad z_3 = 0.9997$

$$\begin{bmatrix} 1 & 2 & -2 & 1 \\ 1 & -1 & 1 & 2 \\ 3 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0.8808 \\ 0.9991 \\ 0.9997 \\ 0.8808 \end{bmatrix} = \begin{bmatrix} (.8808)(1)+(0.9991)(2)+(.9997)(-2)+(.8808)(1) \\ (.8808)(1)+(0.9991)(-1)+(.9997)(1)+(.8808)(2) \\ (.8808)(3)+(0.9991)(1)+(.9997)(-1)+(.8808)(1) \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} 1.7604 \\ 2.643 \\ 3.5226 \end{bmatrix}}_{+} \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}_{\text{bias}} = \underbrace{\begin{bmatrix} 2.7604 \\ 3.6430 \\ 4.5226 \end{bmatrix}}_{b\text{ vector}} \rightarrow \underbrace{\begin{bmatrix} \frac{\exp(2.7604)}{146.1} \\ \frac{\exp(3.6430)}{146.1} \\ \frac{\exp(4.5226)}{146.1} \end{bmatrix}}_{y\text{ vector}} = \underbrace{\begin{bmatrix} 0.1082 \\ 0.2615 \\ 0.6302 \end{bmatrix}}_{y\text{ vector}}$$

$\exp(2.7604)+\exp(3.6430)+\exp(4.5226) = 146.1$

1c. $b_2 = 3.6430$

1d. $\hat{y}_2 = 0.6302$

1e. Class Predicted: Class 3

Question assigned to the following page: 11

Problem 4 Weight Gradients

$\frac{\partial C}{\partial y} = (\hat{y} - y)^2$

$\frac{\partial C}{\partial w} = \frac{\partial C}{\partial y_o} \frac{\partial y_o}{\partial y} \frac{\partial y}{\partial w}$ $\quad \frac{\partial C}{\partial b} = \frac{\partial C}{\partial y_o} \frac{\partial y_o}{\partial y} \frac{\partial y}{\partial b} \rightarrow 1$

$\frac{\partial C}{\partial w_{11}} = [0.1082 - 0][0.1082]$ $\quad \frac{\partial C}{\partial w_{12}} = [0.1082 - 0][0.1082]$

$.0117$ $\quad .0117$

$\frac{\partial C}{\partial w_{13}} = [0.1082 - 0][0.1082]$ $\quad \frac{\partial C}{\partial w_{14}} = [0.1082 - 0][0.1082]$

$.0117$ $\quad .0117$

$\frac{\partial C}{\partial w_{21}} = [0.2615 - 1][0.2615]$ $\quad \frac{\partial C}{\partial w_{22}} = [0.2615 - 1][0.2615]$

$-.1931$ $\quad -.1931$

$\frac{\partial C}{\partial w_{23}} = [0.2615 - 1][0.2615]$ $\quad \frac{\partial C}{\partial w_{24}} = [0.2615 - 1][0.2615]$

$-.1931$ $\quad -.1931$

$\frac{\partial C}{\partial w_{31}} = [0.6302 - 0][0.6302]$ $\quad \frac{\partial C}{\partial w_{32}} = [0.6302 - 0][0.6302]$

$.3972$ $\quad .3972$

$\frac{\partial C}{\partial w_{33}} = [0.6302 - 0][0.6302]$ $\quad \frac{\partial C}{\partial w_{34}} = [0.6302 - 0][0.6302]$

$.3972$ $\quad .3972$

$\frac{\partial C}{\partial b_1} = [0.1082 - 0] = 0.1082$

$\frac{\partial C}{\partial b_2} = [0.2615 - 1] = -.7385$

$\frac{\partial C}{\partial b_3} = [0.6302 - 0] = 0.6302$

Question assigned to the following page: 11

Problem 4 Weight Gradients in a 3

$$\frac{\partial C}{\partial w} = \frac{\partial C}{\partial z_0}\frac{\partial z_0}{\partial z}\frac{\partial z}{\partial w}$$

$$\frac{\partial C}{\partial z_0} = \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} + \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} + \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0}$$

$$\frac{\partial C}{\partial w} = \begin{bmatrix} \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} + \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} + \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} \end{bmatrix} \begin{bmatrix} \frac{\partial z_0}{\partial z} \end{bmatrix} \begin{bmatrix} \frac{\partial z}{\partial w} \end{bmatrix}$$

$$\frac{\partial C}{\partial w_{11}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad .19$$

$$\frac{\partial C}{\partial w_{12}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad .19$$

$$\frac{\partial C}{\partial w_{13}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{14}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{15}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad .19$$

$$\frac{\partial C}{\partial w_{16}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .8808 \\ (1-.8808) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad .19$$

$$\frac{\partial C}{\partial w_{21}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 1.94\times10^{-4}$$

$$\frac{\partial C}{\partial w_{22}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 1.94\times10^{-4}$$

$$\frac{\partial C}{\partial w_{23}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{24}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{25}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 1.94\times10^{-4}$$

$$\frac{\partial C}{\partial w_{26}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9991 \\ (1-.9991) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 1.94\times10^{-4}$$

$$\frac{\partial C}{\partial w_{31}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 6.472\times10^{-5}$$

$$\frac{\partial C}{\partial w_{32}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 6.472\times10^{-5}$$

$$\frac{\partial C}{\partial w_{33}} = \begin{bmatrix} .0117 + -.1931 \\ +.3972 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{34}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 0 \end{bmatrix} \quad 0$$

$$\frac{\partial C}{\partial w_{35}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 6.472\times10^{-5}$$

$$\frac{\partial C}{\partial w_{36}} = \begin{bmatrix} .2158 \end{bmatrix} \begin{bmatrix} .9997 \\ (1-.9997) \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \quad 6.472\times10^{-5}$$

Question assigned to the following page: 11

Problem 4 Weight Gradients in b 4

$$\frac{\partial C}{\partial w_{41}} = [.2158][.8808(1-.8808)][1] = .19$$

$$\frac{\partial C}{\partial w_{42}} = [.2158][.8808(1-.8808)][1] = .19$$

$$\frac{\partial C}{\partial w_{43}} = [.2158][.8808(1-.8808)][0] = 0$$

$$\frac{\partial C}{\partial w_{44}} = [.2158][.8808(1-.8808)][0] = 0$$

$$\frac{\partial C}{\partial w_{45}} = [.2158][.8808(1-.8808)][1] = .19$$

$$\frac{\partial C}{\partial w_{46}} = [.2158][.8808(1-.8808)][1] = .19$$

$$\frac{\partial C}{\partial b} = \left[\frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0} + \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial z_0}\right]\left[\frac{\partial z_0}{\partial z}\right]\left[\frac{\partial z}{\partial b}\right] \rightarrow 1$$

$$\frac{\partial C}{\partial b_1} = [.2158][.8808(1-.8808)] = .19$$

$$\frac{\partial C}{\partial b_2} = [.2158][.9991(1-.9991)] = 1.94\times10^{-4}$$

$$\frac{\partial C}{\partial b_3} = [.2158][.9997(1-.9997)] = 6.472\times10^{-5}$$

$$\frac{\partial C}{\partial b_4} = [.2158][.8808(1-.8808)] = .19$$

Question assigned to the following page: 11

Problem 4 $w_{k+1} = w_k + \eta \frac{\partial C}{\partial w}$ $b_{k+1} = b_k + 1 \frac{\partial C}{\partial b}$

a new

$$\begin{bmatrix} 1-1(.19) & 2-1(.19) & -3-1(0) & 0-1(0) & 1-1(.19) & -3-1(.19) \\ 3-1(.0002) & 1-1(.0002) & 2-1(0) & 1-1(0) & 0-1(.0002) & 2-1(.0002) \\ 2-1(.0001) & 2-1(.0001) & 2-1(0) & 2-1(0) & 2-1(.0001) & 1-1(.0001) \\ 1-1(.19) & 0-1(.19) & 2-1(0) & 1-1(0) & -2-1(.19) & 2-1(.19) \end{bmatrix}$$

anew

$$\begin{bmatrix} .81 & 1.81 & -3 & 0 & .81 & -3.19 \\ 2.9998 & .9998 & 2 & 1 & -.0002 & 1.9998 \\ 1.9999 & 1.999 & 2 & 2 & 1.999 & .9999 \\ .81 & -.19 & 2 & 1 & -2.19 & 1.81 \end{bmatrix}$$

bias

$$\begin{bmatrix} 1-1(.19) \\ 1-1(.0002) \\ 1-1(.0001) \\ 1-1(.19) \end{bmatrix}$$

= bias new

$$\begin{bmatrix} .81 \\ .9998 \\ .9999 \\ .81 \end{bmatrix}$$

b new

$$\begin{bmatrix} 1-1(.0117) & 2-1(.0117) & -2-1(.0117) & 1-1(.0117) \\ 1-1(-.1931) & -1-1(-.1931) & 1-1(-.1931) & 2-1(-.1931) \\ 3-1(.3972) & 1-1(.3972) & -1-1(.3972) & 1-1(.3972) \end{bmatrix}$$

bnew

$$\begin{bmatrix} .9883 & 1.9883 & -2.0117 & .9883 \\ 1.1931 & -.8069 & 1.1931 & 2.1931 \\ 2.6028 & 0.6028 & -1.3972 & 0.6028 \end{bmatrix}$$

bias

$$\begin{bmatrix} 1-1(.1082) \\ 1-1(-.7382) \\ 1-1(.6302) \end{bmatrix}$$

= bias new

$$\begin{bmatrix} .8918 \\ 1.7382 \\ .3698 \end{bmatrix}$$

Question assigned to the following page: 11

Problem 4

a new (6×1)

$$\begin{bmatrix} .81 & 1.81 & -3 & 0 & .81 & -3.19 \\ 3 & 1 & 2 & 1 & 0 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 \\ .81 & -.19 & 2 & 1 & -2.19 & 1.81 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} (1)(.81)+(1)(1.81)+0(-3)+0(0)+1(.81)+1(-3.19) \\ (1)(3)+(1)(1)+0(2)+0(1)+1(0)+1(2) \\ (1)(2)+(1)(2)+0(2)+0(2)+1(2)+1(1) \\ (1)(.81)+(1)(-.19)+0(2)+0(-2.19)+1(-2.19)+1(1.81) \end{bmatrix}$$

bias a vector → z vector = z vector

$$\begin{bmatrix} .24 \\ 6 \\ 7 \\ .24 \end{bmatrix} + \begin{bmatrix} .81 \\ .9998 \\ .9999 \\ .81 \end{bmatrix} = \begin{bmatrix} 1.05 \\ 7 \\ 8 \\ 1.05 \end{bmatrix} \longrightarrow \begin{bmatrix} \frac{1}{1+\exp(-1.05)} \\ \frac{1}{1+\exp(-7)} \\ \frac{1}{1+\exp(-8)} \\ \frac{1}{1+\exp(-1.05)} \end{bmatrix} = \begin{bmatrix} .7408 \\ .9991 \\ .9997 \\ .7408 \end{bmatrix}$$

b new

$$\begin{bmatrix} .9883 & 1.9883 & -2.0117 & .9883 \\ 1.1931 & -.8069 & 1.1931 & 2.1931 \\ 2.6028 & 0.6028 & -1.3972 & .6028 \end{bmatrix} \begin{bmatrix} .7408 \\ .9991 \\ .9997 \\ .7408 \end{bmatrix} = \begin{bmatrix} .9883(.7408)+1.9883(.9991)+-2.0117(.9997)+.9883(.7408) \\ 1.1931(.7408)+-.8069(.9991)+1.1931(.9997)+2.1931(.7408) \\ 2.6028(.7408)+0.6028(.9991)+-1.3972(.9997)+.6028(.7408) \end{bmatrix}$$

bias b vector → y vector

$$\begin{bmatrix} 1.4397 \\ 2.895 \\ 1.5846 \end{bmatrix} + \begin{bmatrix} .8918 \\ 1.7382 \\ .3698 \end{bmatrix} = \begin{bmatrix} 2.3315 \\ 4.6333 \\ 1.9545 \end{bmatrix} \rightarrow \begin{bmatrix} \frac{\exp(2.3315)}{120.2} \\ \frac{\exp(4.6333)}{120.2} \\ \frac{\exp(1.9545)}{120.2} \end{bmatrix} = \begin{bmatrix} .0856 \\ .8556 \\ .0587 \end{bmatrix}$$

$\exp(2.3315)+\exp(4.6333)+\exp(1.9545)$

2a: 1.1931 2b: .8918

2c: 2 2d: .81

2e: Predicted Class: Class 2

Question assigned to the following page: 11

Problem 4

$$l(\hat{y}, y) = -\sum_{i=1}^{3} y_i \times \log(\hat{y}_i)$$

$$l(\hat{y}, y) = -\sum_{i=1}^{3} \underset{1\times 3}{[0 \;\; 1 \;\; 0]} \underset{3\times 1}{\begin{bmatrix} \log(0.1082) \\ \log(0.2615) \\ \log(0.6302) \end{bmatrix}} =$$

1.f. $l(\hat{y}, y) = 1.3413$

$$l(\hat{y}, y) = -\sum_{i=1}^{3} [0 \;\; 1 \;\; 0] \begin{bmatrix} \log(.0856) \\ \log(.8556) \\ \log(.0587) \end{bmatrix}$$

$$l(\hat{y}, y) = 0.156$$

1.a. $a_1 = 2$
$z_1 = 0.8808$

1b. $a_3 = 8$
$z_3 = 0.9997$

1c. $b_2 = 3.6430$

1d. $\hat{y}_2 = 0.6302$

1e. Class predicted : Class 3

1f. loss : 1.3413

2a. $\beta_{2,1} = 1.1931$

2b. $\beta_{1,0} = .8918$

2c. $\alpha_{3,4} = 2$

2d. $z_2 \alpha_{2,0} = .9999$

2e. Class predicted : Class 2