IME692A: Assignment 1

Arvind Singh Yadav, Sunil Dhaka

September 7, 2021

Question-2

Part-a

Note: For random variable we have used X, Y symbols and for specific value of random variables in sample space($\chi, \mathcal{Y}$) we have used x, y symbols. All the other symbols and formulas have their usual meaning taken from class notes.
We have been given bi-variate joint mass function of (X, Y). Marginal p.m.f. for X:

$$p(X = 0) = \frac{2}{3}$$
$$p(X = 1) = \frac{1}{3}$$

Marginal p.m.f. for Y:

$$p(Y = 0) = \frac{1}{3}$$
$$p(Y = 1) = \frac{2}{3}$$

As we know conditional Entropy is defined by,

$$\mathbf{H}(Y|X) = \sum_{x \in \chi} p(X)\mathbf{H}(Y|X = x)$$

Using joint density, marginal densities in conditional entropy formula, we get $\mathbf{H}(Y|X) = 0.77$.

Part-b

As we know cross Entropy is defined by,

$$\mathbf{H}(X, Y) = -\sum_{x \in \chi, y \in \mathcal{Y}} p(X) log_2(p(Y))$$

Using joint density, marginal densities in cross entropy formula, we get $\mathbf{H}(X, Y) = 1.255$.

Part-c

As we know mutual information is defined by,

$$\mathbf{I}(X, Y) = \sum_{x \in \chi} \sum_{y \in \mathcal{Y}} p(X, Y) log_2 \frac{p(X, Y)}{p(X)p(Y)}$$

Using joint density, marginal densities in mutual information formula, we get $\mathbf{I}(X, Y) = 1.51$.

ime692_assignment1_group8

September 7, 2021

1 Q1_Asgn1

[1]:
```
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from scipy.stats import chi2
import matplotlib.pyplot as plt
from matplotlib.patches import Ellipse
import math
```

[2]:
```
data=pd.read_csv("train3.csv",header=None)
dataset=data.iloc[0:130,]
```

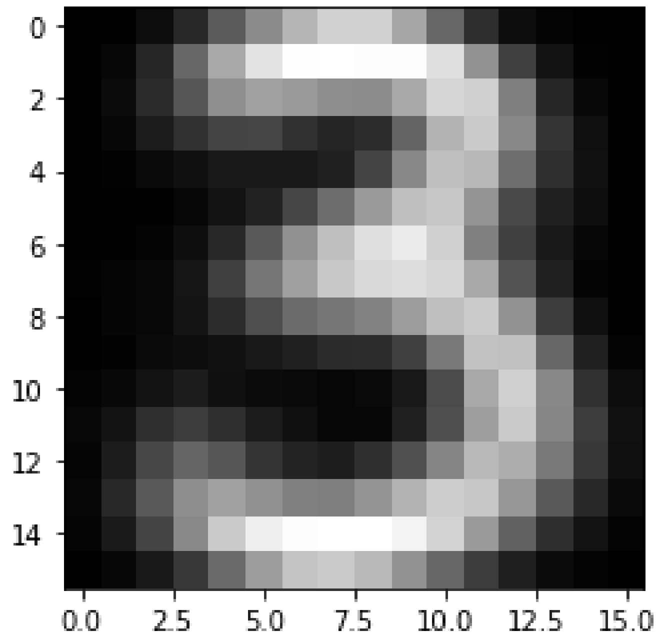
[3]:
```
pca=PCA(n_components=2,svd_solver="full")
pca_model=pca.fit(dataset)
```

[4]:
```
mean_image=np.array(dataset.mean())
```

Mean Image

[5]:
```
mean_image=mean_image.reshape([16,16])
plt.gray()
plt.imshow(mean_image)
```

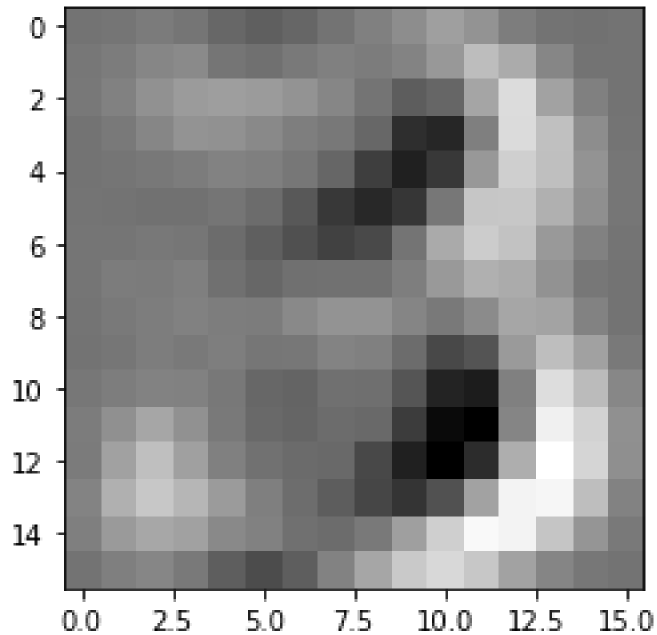
[5]:
```
<matplotlib.image.AxesImage at 0x7f00c81085e0>
```



1.0.1 PCA1 Image

[6]:
```
PCA1=pca_model.components_[0].reshape([16,16])
plt.gray()
plt.imshow(PCA1)
```

[6]: <matplotlib.image.AxesImage at 0x7f00c80766a0>



1.0.2 First eigen vector

[7]:
```python
print(pca_model.components_[0])
```

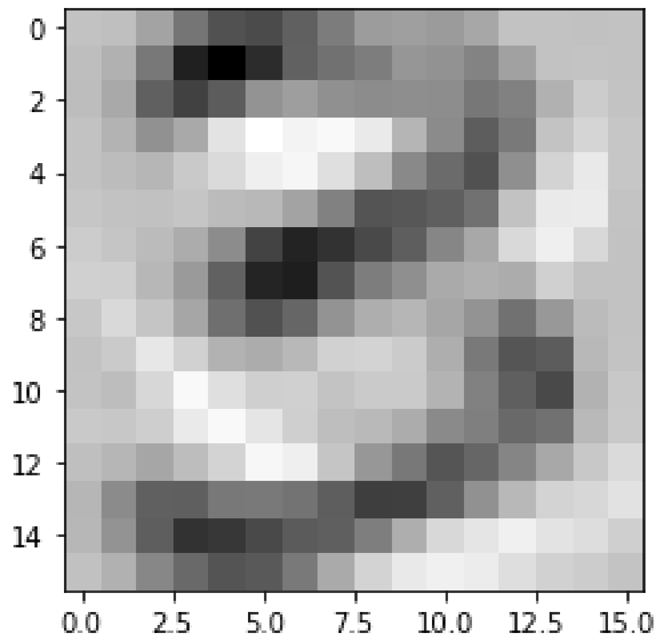
```
[-1.11022302e-16  2.63296430e-03  1.10331761e-02  3.24234924e-03
 -1.62900321e-02 -2.76929218e-02 -1.80589162e-02  7.26675093e-04
  1.85696264e-02  3.68620478e-02  6.14789921e-02  4.52702250e-02
  1.40720751e-02 -4.01882808e-05 -5.32740846e-04  0.00000000e+00
  5.69257949e-03  1.25963395e-02  2.64100915e-02  3.19981402e-02
  3.21198085e-03 -4.30254242e-03  8.21720789e-03  1.76447164e-02
  1.24052033e-02  2.24713750e-02  5.12084158e-02  1.03434389e-01
  7.83667109e-02  2.69401845e-02  5.79870468e-04  5.60519386e-45
  6.54848506e-03  1.90444014e-02  4.21173038e-02  5.57069596e-02
  6.00917770e-02  5.63997154e-02  4.66889956e-02  2.69181146e-02
  1.87214477e-03 -3.10483054e-02 -1.85735234e-02  6.98440643e-02
  1.47207857e-01  6.58534255e-02  1.82834299e-02  3.61438111e-04
  5.24847755e-04  1.03947200e-02  2.68136010e-02  4.45883472e-02
  4.21057925e-02  3.10329996e-02  1.55688404e-02  6.57454399e-03
 -1.70347387e-02 -9.60508036e-02 -1.07919344e-01  1.66801412e-02
  1.45032155e-01  1.07086556e-01  3.71564991e-02  1.81213640e-03
  0.00000000e+00  2.84843572e-03  7.69135779e-03  1.32110575e-02
  2.11969230e-02  1.67860038e-02  6.41894290e-03 -1.78705940e-02
 -7.41334166e-02 -1.15957167e-01 -8.12485087e-02  5.18901826e-02
  1.28972765e-01  1.05719671e-01  4.53337014e-02  4.82334176e-03
  1.17636260e-03  1.76631022e-06 -6.97242303e-04 -1.99375284e-03
  2.31014825e-03 -9.53245648e-03 -3.81510742e-02 -8.02303902e-02
```

```
-1.04381301e-01 -8.42318307e-02  5.92289805e-03  1.15987358e-01
 1.17409584e-01  8.47701052e-02  3.85720927e-02  3.94508445e-03
 2.66889474e-03  3.05729503e-03  6.71090894e-03  3.88376456e-03
-9.82178393e-03 -2.76331420e-02 -4.87953849e-02 -6.80024764e-02
-5.76214801e-02  1.47285501e-03  7.73982811e-02  1.25053215e-01
 1.10646070e-01  5.28841975e-02  2.00029420e-02  1.09421886e-03
 3.33910650e-03  1.23648494e-02  1.18374692e-02  1.73326370e-02
-4.35008492e-03 -1.62307232e-02 -3.86655803e-03 -2.79539892e-03
-2.19983205e-03  1.61222078e-02  5.25534386e-02  8.47057801e-02
 7.80528909e-02  4.31020325e-02  6.14965458e-03  0.00000000e+00
 1.13581375e-03  9.84371715e-03  1.42932389e-02  1.90440107e-02
 1.36283570e-02  1.28074688e-02  3.06433998e-02  4.43959412e-02
 4.43671712e-02  2.54311199e-02  1.01319633e-02  3.41700991e-02
 7.15952754e-02  6.77177930e-02  2.14900547e-02  2.57749285e-04
 0.00000000e+00  3.81225254e-03  1.38263807e-02  7.93304616e-03
 1.53077105e-02  4.52152242e-03  5.30384921e-03  2.40675350e-02
 1.86195343e-02 -1.01675540e-02 -5.98705031e-02 -4.33261169e-02
 5.41528730e-02  1.04672784e-01  6.39547475e-02  8.43155286e-03
 5.09366256e-03  1.44143824e-02  2.14662898e-02  1.96917619e-02
 7.49413771e-03 -1.67306682e-02 -2.02180380e-02 -3.73084528e-03
-5.24368084e-03 -4.21275546e-02 -1.11666981e-01 -1.21609844e-01
 1.80021444e-02  1.47448279e-01  1.00433180e-01  2.86853516e-02
 1.25713723e-02  4.00954924e-02  7.07887194e-02  4.20170643e-02
 8.84487214e-03 -1.43731903e-02 -1.90479876e-02 -9.85977570e-03
-1.21000787e-02 -7.46103011e-02 -1.45754567e-01 -1.59628509e-01
 2.48862673e-02  1.74443325e-01  1.32651440e-01  4.08846157e-02
 1.11843418e-02  6.42917207e-02  1.06623726e-01  6.33558805e-02
 1.83966595e-02 -3.08662872e-03 -1.12921507e-02 -1.37343549e-02
-5.92251496e-02 -1.14471208e-01 -1.60945752e-01 -9.90313642e-02
 8.30080484e-02  1.96214282e-01  1.36520601e-01  3.88011629e-02
 2.23643987e-02  8.56276639e-02  1.18070488e-01  9.41216486e-02
 5.64546998e-02  1.72659837e-02 -8.24425821e-03 -3.11608960e-02
-6.25209310e-02 -8.83022900e-02 -4.51913026e-02  6.56433340e-02
 1.80222617e-01  1.83105458e-01  1.05257502e-01  2.14363871e-02
 1.72149748e-02  5.47515371e-02  7.28214489e-02  6.41965994e-02
 3.04534703e-02  1.95709080e-02 -3.33233296e-03 -1.00192469e-02
 7.99633324e-03  6.21651732e-02  1.28562402e-01  1.86561942e-01
 1.80845051e-01  1.14959831e-01  4.73032130e-02  8.17976842e-03
 6.43448812e-04  1.74390476e-02  2.69283107e-02  1.04423895e-02
-2.95671463e-02 -5.38176568e-02 -2.96738798e-02  2.14192328e-02
 7.15860620e-02  1.20370337e-01  1.41416102e-01  1.17199845e-01
 6.53648443e-02  2.51885792e-02  6.46130784e-03  1.67793333e-04]
```

1.0.3 PCA2 Image

[8]:
```
PCA2=pca_model.components_[1].reshape([16,16])
plt.gray()
plt.imshow(PCA2)
```

[8]: <matplotlib.image.AxesImage at 0x7f00bbfd7c70>



1.0.4 Second eigen vector

[9]:
```
print(pca_model.components_[1])
```

```
[-1.21430643e-17 -3.13437877e-03 -3.20995451e-02 -7.92679638e-02
 -1.15901435e-01 -1.22090475e-01 -1.00489915e-01 -7.20617566e-02
 -3.95265448e-02 -3.64548032e-02 -4.02716719e-02 -2.79367583e-02
 -4.15763955e-04  3.55357591e-06 -1.21747734e-03 -1.27054942e-21
 -4.16956830e-03 -1.78054959e-02 -7.61567203e-02 -1.65890352e-01
 -1.99012413e-01 -1.52321457e-01 -9.98256683e-02 -8.30466853e-02
 -7.11837044e-02 -4.50002727e-02 -4.89767471e-02 -6.41175686e-02
 -3.42771016e-02 -1.67048841e-04  1.49051285e-03 -1.46936794e-39
 -4.79648211e-03 -2.54620254e-02 -1.00859543e-01 -1.31714972e-01
 -1.04408239e-01 -4.67707279e-02 -3.69462454e-02 -5.15205759e-02
 -5.50227307e-02 -5.34809312e-02 -5.41537893e-02 -7.66791216e-02
 -6.70007818e-02 -1.76418330e-02  1.00923314e-02  1.68898722e-03
 -3.84428283e-04 -1.44367648e-02 -4.99876851e-02 -2.59900015e-02
  3.42301081e-02  6.26917457e-02  5.05572778e-02  5.60510139e-02
```

4.05480542e-02	-1.38318379e-02	-5.62400716e-02	-1.02939770e-01
-7.40385950e-02	1.02862934e-03	1.92519859e-02	3.86583082e-03
-0.00000000e+00	-5.87091766e-03	-1.22630501e-02	7.45900064e-03
2.40299800e-02	4.59353026e-02	5.34807220e-02	2.94361422e-02
-4.29307699e-03	-5.83904158e-02	-8.87414323e-02	-1.15499823e-01
-5.20019883e-02	1.71848481e-02	3.92736662e-02	3.46841306e-03
4.38300983e-03	6.58109584e-06	-7.14526284e-04	2.79051084e-03
-7.82689021e-03	-1.05109346e-02	-3.17465250e-02	-6.71065294e-02
-1.12663484e-01	-1.09128832e-01	-1.01021233e-01	-8.29833875e-02
-3.46117209e-04	4.06223953e-02	4.21460159e-02	1.87055207e-03
9.94403582e-03	3.23520405e-03	-7.82187898e-03	-2.28122305e-02
-5.49902456e-02	-1.29875811e-01	-1.62275635e-01	-1.47064945e-01
-1.22641811e-01	-1.01959358e-01	-6.19920662e-02	-2.72463252e-02
2.38252202e-02	4.56394968e-02	2.27564212e-02	3.09716479e-04
1.43438995e-02	1.32545564e-02	-1.14524818e-02	-4.07250416e-02
-9.97676947e-02	-1.60271405e-01	-1.67353042e-01	-1.13245279e-01
-7.05173947e-02	-5.27505823e-02	-2.50812388e-02	-1.90784842e-02
-2.21607831e-02	1.38956302e-02	1.34963151e-04	-0.00000000e+00
5.25575954e-03	2.34021039e-02	2.85947152e-03	-2.84460337e-02
-8.53359168e-02	-1.15908037e-01	-9.45433280e-02	-4.67244254e-02
-2.06157594e-02	-1.19526096e-02	-2.92515825e-02	-4.72891826e-02
-8.20651638e-02	-4.30088145e-02	-7.07065196e-03	-1.42589798e-04
-0.00000000e+00	7.55501475e-03	3.72575076e-02	1.54103444e-02
-1.66131850e-02	-2.25926553e-02	-1.05829772e-02	1.50370476e-02
1.72255534e-02	9.24601167e-03	-2.03103618e-02	-7.41191019e-02
-1.11102514e-01	-1.04050339e-01	-1.01518169e-02	9.55226562e-04
2.02162703e-03	-5.47813388e-03	2.17851300e-02	5.61818174e-02
2.99015868e-02	1.36109881e-02	1.37410791e-02	1.65529173e-03
8.42515819e-03	7.80268254e-03	-1.50755326e-02	-6.62558628e-02
-1.01203338e-01	-1.22604491e-01	-1.63272313e-02	5.91780463e-03
7.27697678e-03	4.80643470e-03	1.25457029e-02	4.11401027e-02
5.58070667e-02	3.53578697e-02	1.35640924e-02	-3.84967807e-03
-9.17860755e-03	-2.28776246e-02	-5.74699555e-02	-6.88267058e-02
-9.03505361e-02	-8.31428883e-02	-9.25127300e-03	7.47968765e-03
-2.83491133e-03	-1.22972154e-02	-2.86408320e-02	-5.56690126e-03
1.78870018e-02	5.38250169e-02	4.61334251e-02	2.90271441e-03
-4.39681514e-02	-7.55998101e-02	-1.11686063e-01	-9.39075917e-02
-6.22759951e-02	-2.67493695e-02	5.72131451e-03	2.45983689e-02
-1.27533398e-02	-5.62411392e-02	-1.00260013e-01	-1.00917180e-01
-7.54827740e-02	-7.42745404e-02	-8.07423296e-02	-1.02387775e-01
-1.35171554e-01	-1.34231465e-01	-1.00145504e-01	-5.03590913e-02
-1.00714478e-02	1.70368623e-02	2.08458402e-02	3.28254888e-02
-1.17745958e-02	-4.80885050e-02	-1.02016807e-01	-1.47102400e-01
-1.42411206e-01	-1.23861547e-01	-1.05550281e-01	-1.01493663e-01
-7.00395824e-02	-2.06330076e-02	2.21349986e-02	3.56246106e-02
4.68044498e-02	3.46305372e-02	2.73298195e-02	1.34529065e-02
-8.35269429e-04	-1.75166492e-02	-6.01393256e-02	-8.97769309e-02
-1.12631767e-01	-1.06469147e-01	-7.33241602e-02	-2.42271460e-02

```
 1.77342905e-02  3.93739802e-02  4.69406924e-02  4.29224263e-02
 2.87741107e-02  1.53432607e-02  1.01034833e-02  3.33748328e-04]
```

1.0.5 Eigen values

[10]:
```
round(pca_model.explained_variance_ratio_[0],3)
```

[10]:
```
0.144
```

[11]:
```
round(pca_model.explained_variance_ratio_[1],3)
```

[11]:
```
0.098
```

- **Observation: First two eigen vectors explain around 25% of variance.**

2 Q3_Asgn1

[12]:
```
mu=np.array([2,1])
cov=np.array([[2 ,0.4],[0.4,2]])
eigen_values,eigen_vector_matrix=np.linalg.eig(cov)
```

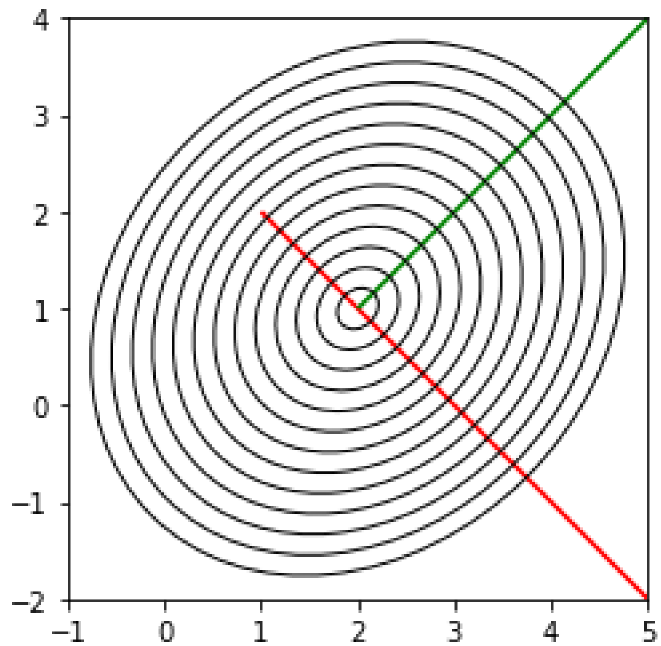
[13]:
```
eigen_vector_matrix
```

[13]:
```
array([[ 0.70710678, -0.70710678],
       [ 0.70710678,  0.70710678]])
```

2.0.1 Contour Plot for Bi-variate Normal Distribution

[14]:
```
step_size = 0.3
cont_levels = np.arange(0, 4, step_size)
axis_angle=np.arccos(np.dot(eigen_vector_matrix[:,0]/np.linalg.
 ↪norm(eigen_vector_matrix[:,0]),[1,0]))*180/np.pi
# to get angle in degrees between x-axis and the contour ellipses major axis
sub_plot = plt.subplot(aspect='equal')
for level in cont_levels:
    ellipse = Ellipse((2,1), level*math.sqrt(eigen_values[0]), level*math.
 ↪sqrt(eigen_values[1]), angle=axis_angle, fill=False, edgecolor='black')
    sub_plot.add_artist(ellipse)

plt.xlim(-1, 5)
plt.ylim(-2, 4)
plt.quiver([1,2],[2,1],eigen_vector_matrix[:,0],eigen_vector_matrix[:
 ↪,1],scale=0.5,color=['r','g'])
plt.show()
```



- Red one is minor and green is major. Both vectors are stored below.

[15]:
```
major_axis=eigen_vector_matrix[:,0]
minor_axis=eigen_vector_matrix[:,1]
```

2.0.2 Axis Lengths

- Here predict_level indicates the $(1-\alpha)$ prediction area of above ellipse.

[16]:
```
predict_level=0.95 # that means alpha = 0.05
critical_level=chi2.ppf(predict_level,2)
```

a). Major Axis Length with $\alpha = 0.05$

[17]:
```
round(np.sqrt(critical_level*eigen_values[0]),2)
```

[17]: 3.79

b). MInor Axis Length with $\alpha = 0.05$

[18]:
```
round(np.sqrt(critical_level*eigen_values[1]),2)
```

[18]: 3.1