ml_assignment1

March 26, 2023

1 assignment_1

1.1 import

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal
from mpl_toolkits.mplot3d import Axes3D
```

1.2 Setup m, v

[ ]:
```
N = 1000

m1 = np.array([10, 5])
v1 = np.array([[25, 0], [0, 9]])
m2 = np.array([10, 5])
v2 = np.array([[5, -3], [-3, 10]])
m3 = np.array([2, 5])
v3 = np.array([[5, -3], [-3, 10]])
```

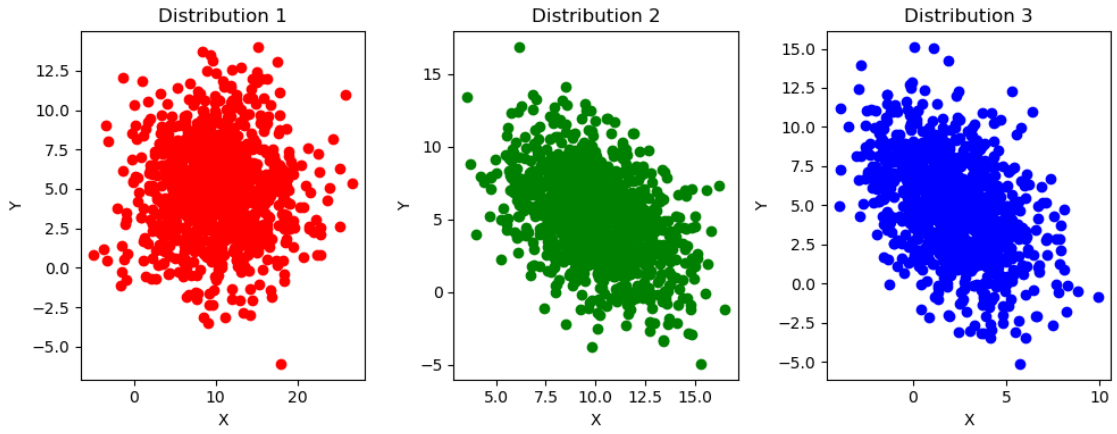
1.3 Generate 1000 samples from each Gaussian distribution

[ ]:
```
x1 = np.random.multivariate_normal(m1, v1, N)
x2 = np.random.multivariate_normal(m2, v2, N)
x3 = np.random.multivariate_normal(m3, v3, N)
```

1.4 Visualizing each data scatter plot

[ ]:
```
# Visualizing each data scatter plot
fig, axs = plt.subplots(1, 3, figsize=(10, 4))
axs[0].scatter(x1[:, 0], x1[:, 1], color='r')
axs[0].set_xlabel('X')
axs[0].set_ylabel('Y')
axs[0].set_title('Distribution 1')

axs[1].scatter(x2[:, 0], x2[:, 1], color='g')
axs[1].set_xlabel('X')
```

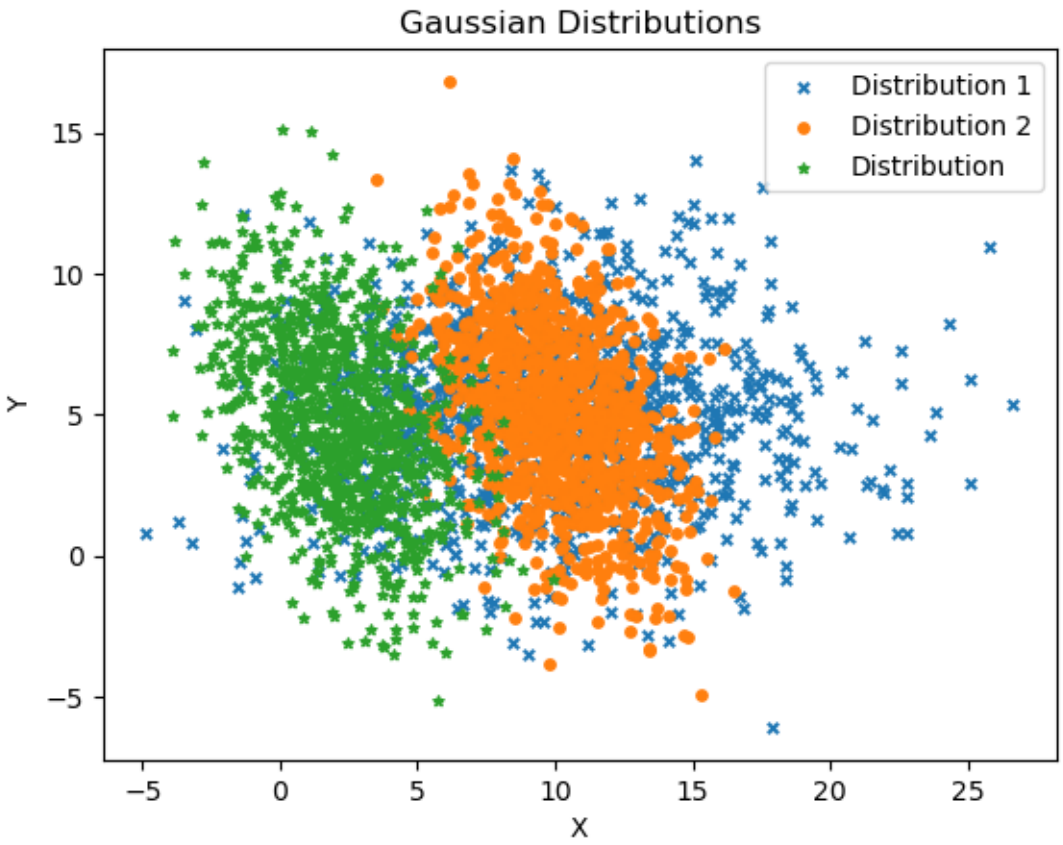
```
axs[1].set_ylabel('Y')
axs[1].set_title('Distribution 2')

axs[2].scatter(x3[:, 0], x3[:, 1], color='b')
axs[2].set_xlabel('X')
axs[2].set_ylabel('Y')
axs[2].set_title('Distribution 3')

plt.tight_layout()
```



1.5 Visualizing distribution in one figure

[ ]:
```
fig, ax = plt.subplots()
ax.scatter(x1[:, 0], x1[:, 1], s=15, marker='x', label='Distribution 1')
ax.scatter(x2[:, 0], x2[:, 1], s=15, marker='o', label='Distribution 2')
ax.scatter(x3[:, 0], x3[:, 1], s=15, marker='*', label='Distribution 3')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_title('Gaussian Distributions')
ax.legend(['Distribution 1', 'Distribution 2', 'Distribution'])
```

[ ]:
```
<matplotlib.legend.Legend at 0x2c60c33a0>
```



1.6 Define x and y ranges for plotting the contours

[ ]:
```
mesh_x1 = np.linspace(-5, 25, 1000)
mesh_x2 = np.linspace(-5, 15, 1000)
X1, X2 = np.meshgrid(mesh_x1, mesh_x2)
X = np.column_stack((X1.ravel(), X2.ravel()))
```

1.7 Calculate the pdf values for each distribution at each point in the meshgrid

[ ]:
```
y1 = multivariate_normal.pdf(X, mean=m1, cov=v1)
y2 = multivariate_normal.pdf(X, mean=m2, cov=v2)
y3 = multivariate_normal.pdf(X, mean=m3, cov=v3)
```

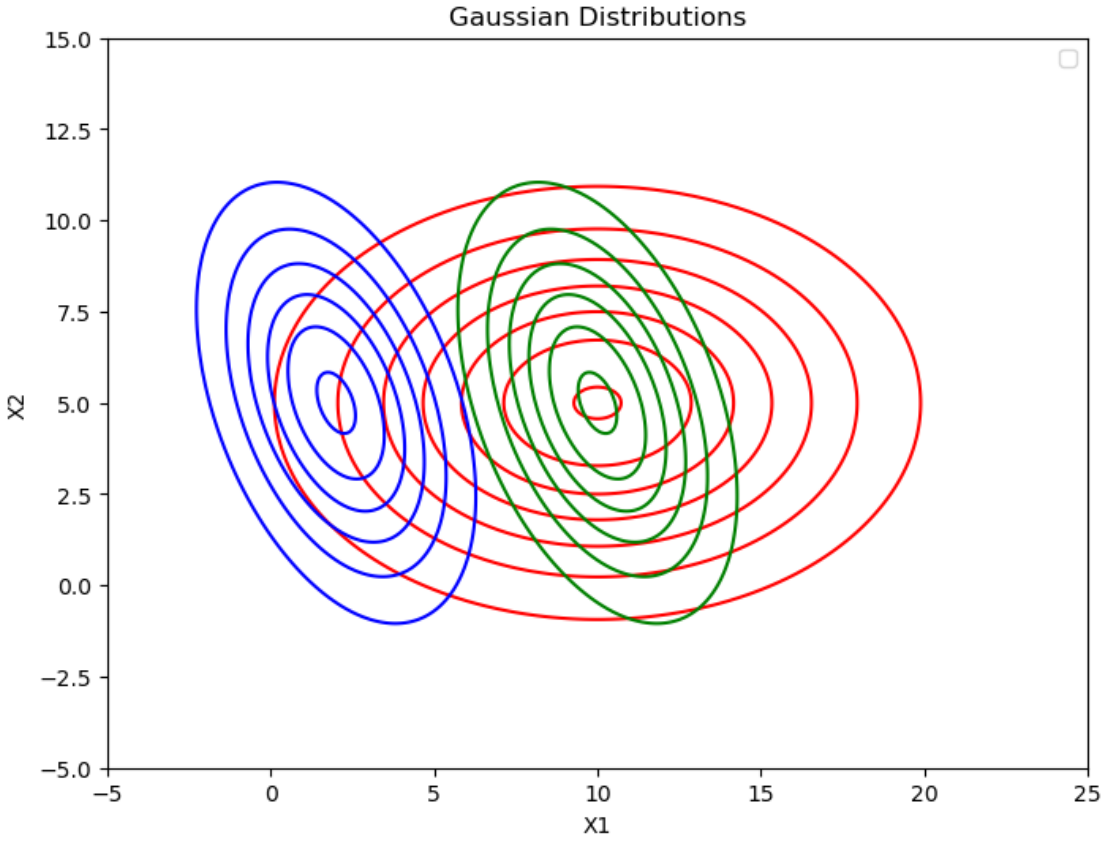
1.8 Reshape the pdf values to match the dimensions of X and Y for contour plotting

[ ]:
```
y1 = y1.reshape(X1.shape)
y2 = y2.reshape(X1.shape)
y3 = y3.reshape(X1.shape)
```

1.9 Plot the pdf contours

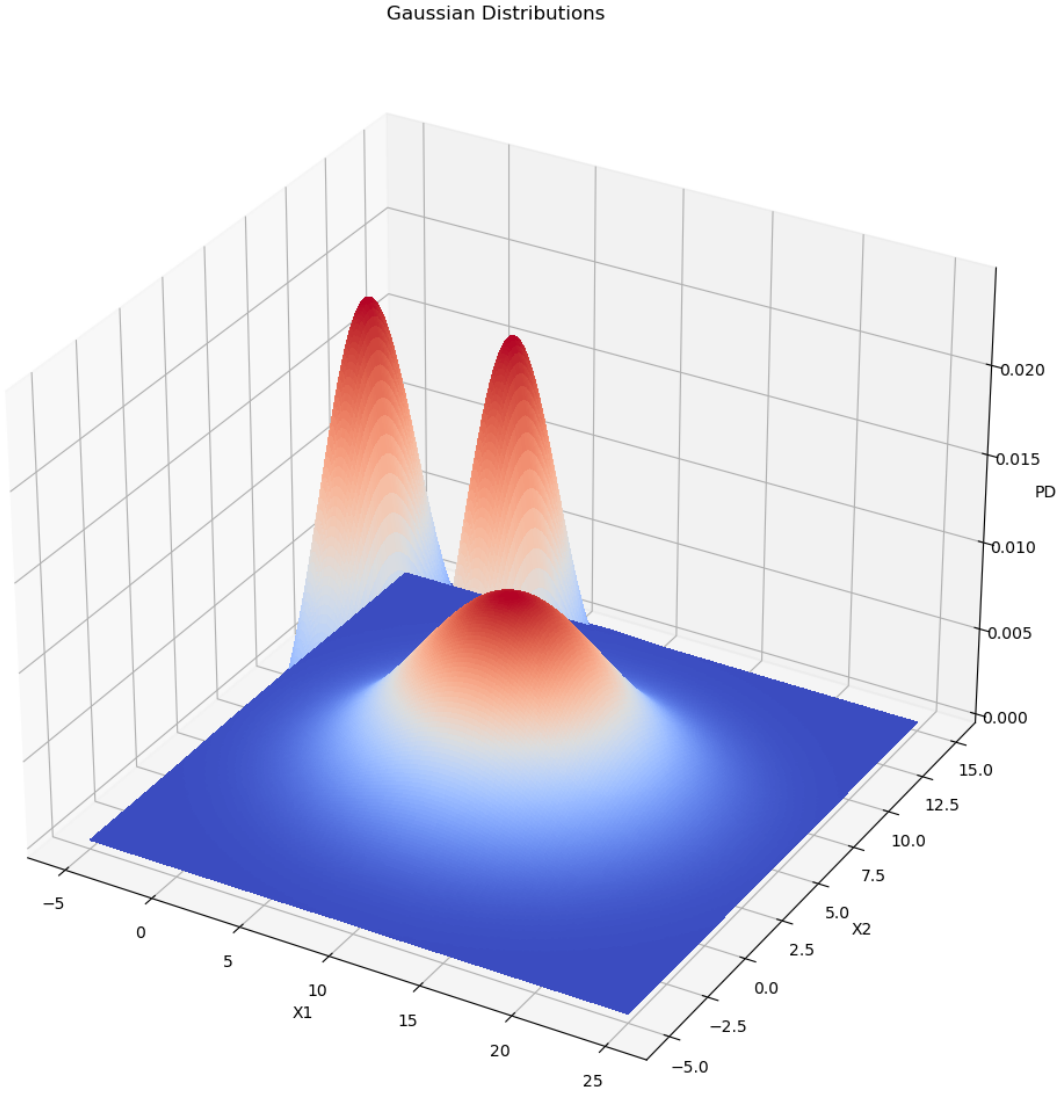
[ ]:
```
fig = plt.figure(figsize=(8, 6))
plt.contour(X1, X2, y1, colors='r')
plt.contour(X1, X2, y2, colors='g')
plt.contour(X1, X2, y3, colors='b')
plt.xlabel('X1')
plt.ylabel('X2')
plt.title('Gaussian Distributions')
plt.legend(['Distribution 1', 'Distribution 2', 'Distribution'])
```

[ ]:
```
<matplotlib.legend.Legend at 0x1444425b0>
```



1.10 Visualizing pdf to surface plot

[ ]:
```
fig = plt.figure(figsize=(10, 10))
ax = fig.add_subplot(111, projection='3d')
surf = ax.plot_surface(X1, X2, y1, cmap = 'coolwarm',rstride=5, cstride=5,␣
 ↪lw=0,antialiased=False)
ax.plot_surface(X1, X2, y2, cmap = 'coolwarm', rstride=5, cstride=5,␣
 ↪lw=0,antialiased=False)
ax.plot_surface(X1, X2, y3, cmap = 'coolwarm', rstride=5, cstride=5,␣
 ↪lw=0,antialiased=False)
ax.set_xlabel('X1')
ax.set_ylabel('X2')
ax.set_zlabel('PDF')
ax.set_title('Gaussian Distributions')
plt.tight_layout()
plt.show()
```


Gaussian Distributions
0.020
0.015
PD
0.010
0.005
0.000
15.0
12.5
10.0
7.5
5.0
X2
2.5
0.0
−2.5
−5.0
−5
0
5
10
X1
15
20
25