Names

Arthur David Redfern	jsf7un
Gabriel Hanson	gh8sj
Ajay Sanjeevan	zwz3wu
Keshara Weerasinghe	cjh9fw
Kabir Menghrajani	km5qte

Problem 1

Part 1
B

Part 2
A

Part 3
B

Part 4
A
(use one hot encoding)

Part 5
A

Part 6
A

Part 7
B

Part 8

B

Part 9

B

Part 10

B, assumes a gaussian shape

Problem 2

Part 1

D

Part 2

E.

Part 3

A.

Part 4

B

Part 5

D

Part 6

B, C, E

Part 7

A, B, D

Part 8

A

Part 9

C, D

Part 10

D

Problem 3

Part A

U0 will move to the left and u1 will move to the right

Part B

It will increase since each iteration increases the likelihood of the data assuming that we are not exactly at a local optimum.

Problem 4

```python
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
```

```python
X = np.array([[1.90, 0.97], [1.76, 0.84], [2.32, 1.63],
              [2.31, 2.09], [1.14, 2.11], [5.02, 3.02],
              [5.74, 3.84], [2.25, 3.47], [4.71, 3.60], [3.17, 4.96]])
```

```python
init1 = np.array([[1.90, 0.97], [3.17, 4.96]])
```

```python
kmeans1 = KMeans(n_clusters=2, random_state=0, init = init1, max_iter = 1).fit(X)
```

/Users/gabrielhanson/anaconda3/envs/metabolism/lib/python3.7/site-packages/sklearn/cluster/_kmeans.py:984: RuntimeWarning: Explicit initial center position passed: performing only one init in KMeans instead of n_init=10.
  self._check_params(X)

```python
y1 = kmeans1.labels_
y1c = kmeans1.cluster_centers_
print('cluster centers after 1 iteration: ', y1c)
```

cluster centers after 1 iteration: [[1.886 1.528]
 [4.178 3.778]]

```python
x1 = [1.90, 1.76, 2.32, 2.31, 1.14, 5.02, 5.74, 2.25, 4.71, 3.17]
x2 = [0.97, 0.84, 1.63, 2.09, 2.11, 3.02, 3.84, 3.47, 3.60, 4.96]
```

```python
plt.scatter(x1, x2, c = y1)
plt.title('Labels after 1 iterations')
```

Text(0.5, 1.0, 'Labels after 1 iterations')


```python
kmeans = KMeans(n_clusters=2, random_state=0, max_iter = 300).fit(X)
```

```python
yf = kmeans.labels_
yfc = kmeans.cluster_centers_
print('cluster centers after convergence :' ,yfc)
```

cluster centers after convergence : [[4.66   3.855   ]
 [1.94666667 1.85166667]]

```python
plt.scatter(x1, x2, c = yf)
plt.title('Labels after Convergence ')
```

Text(0.5, 1.0, 'Labels after Convergence ')

Data #	Cluster Assignment after 1 iteration	Cluster Assignment after convergence
1	1	1
2	1	1
3	1	1
4	1	1
5	1	1
6	2	2
7	2	2
8	2	1

9	2	2
10	2	2

Problem 5

1. Autoencoders use weights with variable initialization to perform dimensionality reduction. As a result they will have variable performance on the same data, versus PCA which when given the same input data and data explanation % target will give the same results.
2. Autoencoders use an iterative training process using backpropagation, as a result it takes significantly longer to use than PCA.
3. The user has more control over PCA in that they can choose a target percentage they want the data to explain, and create a variable number of eigenvalues from that. Autoencoders have no real equivalent to this.