PROJECT - 3

UNSUPERVISED LEARNING – KMEANS & GMM

Mohammed Mahaboob Khan
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
mkhan32@buffalo.edu
PERSON #: 50318613

Abstract

There were three tasks to be performed in this project. The problem given to was a ten-cluster problem. The task was to cluster images and identify an image as belonging to one of the clusters by using Fashion-MNIST dataset. In the first task I use KMeans algorithm to cluster original data space of Fashion-MNIST dataset using Sklearns library. In the second task I built an Auto-Encoder based K-Means clustering model to cluster the condensed representation of the unlabelled fashion MNIST dataset using Keras and Sklearns library. Finally, the third task was to Build an Auto-Encoder based Gaussian Mixture Model clustering model to cluster the condensed representation of the unlabelled fashion MNIST dataset using Keras and Sklearns library.

1 Introduction

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labelled responses.

“Clustering” is the process of grouping similar entities together. The goal of this unsupervised machine learning technique is to find similarities in the data point and group similar data points together. Grouping similar entities together help profile the attributes of different groups. In other words, this will give us insight into underlying patterns of different groups. There are many applications of grouping unlabelled data, for example, you can identify different groups/segments of customers and market each group in a different way to maximize the revenue.

2 Dataset

The dataset in use is the Fashion - MNIST dataset. The Fashion-MNIST is a dataset of Zalando’s article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255.

The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

- **Training data**:

The training data is a set of 60,000 examples from the Fashion – MNIST dataset. Thus, it is a 2D array of the dimensions (60000,785)

- **Testing data**:

The testing data is a set of 10,000 examples from the Fashion – MNIST dataset. Thus, it is a 2D array of the dimensions (10000,785)

3 Pre-processing

Before feeding the data to the model, we separate the label values from the training and testing data. Therefore, after pre-processing we have the following:

- Training data is split into xtrain and ytrain, having the dimensions (60000,784) and (60000,1) respectively.
- Testing data is split into xtest and ytest, having the dimensions (10000,784) and (10000,1) respectively.

4 Architecture

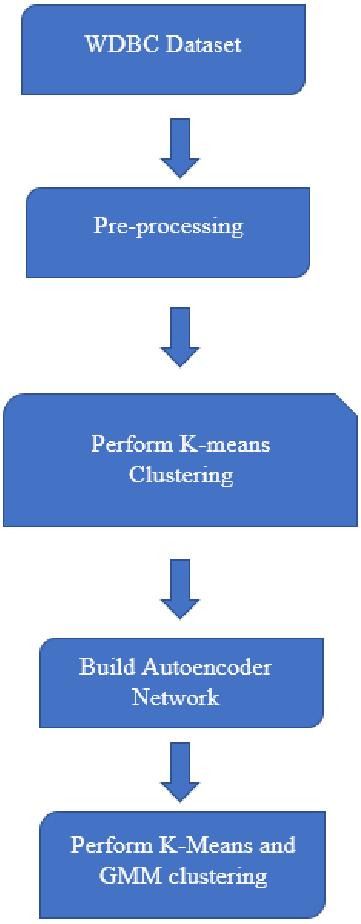

WDBC Dataset
Pre-processing
Perform K-means Clustering
Build Autoencoder Network
Perform K-Means and GMM clustering

Figure 1 – System Architecture

- **Clustering:**

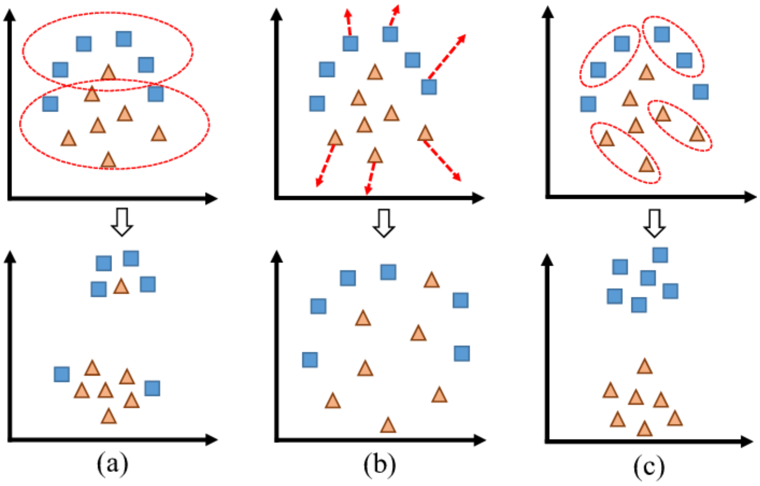
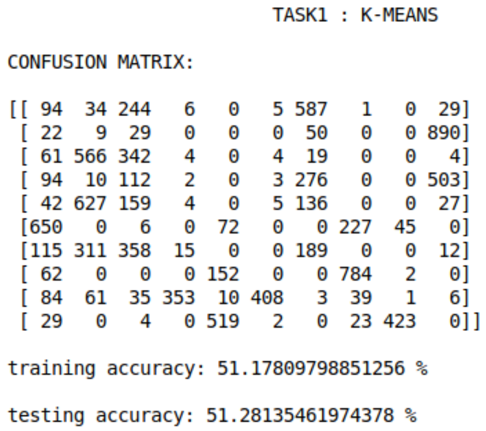


Figure 2 – Clustering

5 Results and Performance

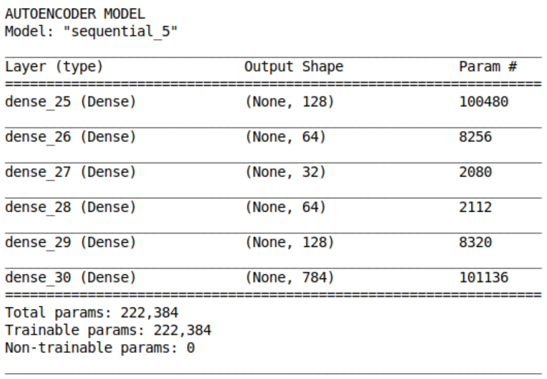
5.1 Task 1



```
                    TASK1 : K-MEANS

CONFUSION MATRIX:

[[ 94  34 244   6   0   5 587   1   0  29]
 [ 22   9  29   0   0   0  50   0   0 890]
 [ 61 566 342   4   0   4  19   0   0   4]
 [ 94  10 112   2   0   3 276   0   0 503]
 [ 42 627 159   4   0   5 136   0   0  27]
 [650   0   6   0  72   0   0 227  45   0]
 [115 311 358  15   0   0 189   0   0  12]
 [ 62   0   0   0 152   0   0 784   2   0]
 [ 84  61  35 353  10 408   3  39   1   6]
 [ 29   0   4   0 519   2   0  23 423   0]]

training accuracy: 51.17809798851256 %

testing accuracy: 51.28135461974378 %
```

Figure 3 –KMeans

5.2 Autoencoder



```
AUTOENCODER MODEL
Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_25 (Dense)             (None, 128)               100480
_________________________________________________________________
dense_26 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_27 (Dense)             (None, 32)                2080
_________________________________________________________________
dense_28 (Dense)             (None, 64)                2112
_________________________________________________________________
dense_29 (Dense)             (None, 128)               8320
_________________________________________________________________
dense_30 (Dense)             (None, 784)               101136
=================================================================
Total params: 222,384
Trainable params: 222,384
Non-trainable params: 0
_________________________________________________________________
```

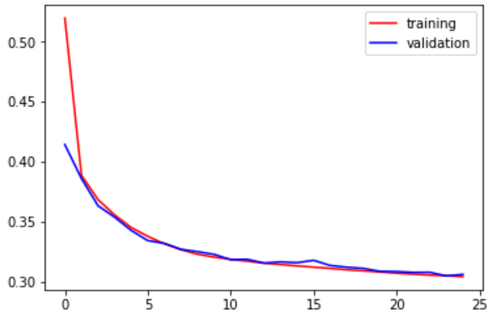
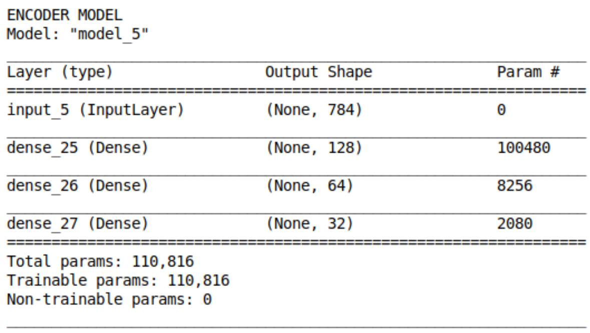


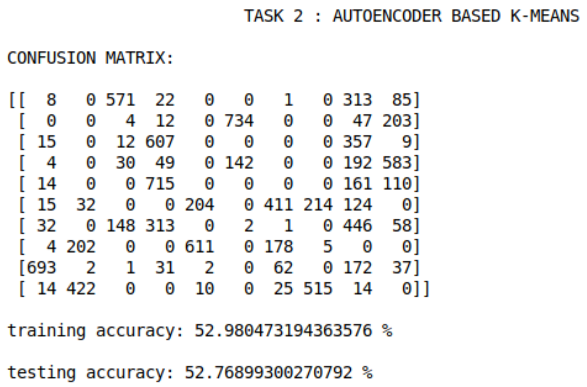
Figure 4 – Autoencoder and graph for training and validation loss



```
ENCODER MODEL
Model: "model_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         (None, 784)               0
_________________________________________________________________
dense_25 (Dense)             (None, 128)               100480
_________________________________________________________________
dense_26 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_27 (Dense)             (None, 32)                2080
=================================================================
Total params: 110,816
Trainable params: 110,816
Non-trainable params: 0
_________________________________________________________________
```

Figure 5 – Encoder model extracted from the trained autoencoder

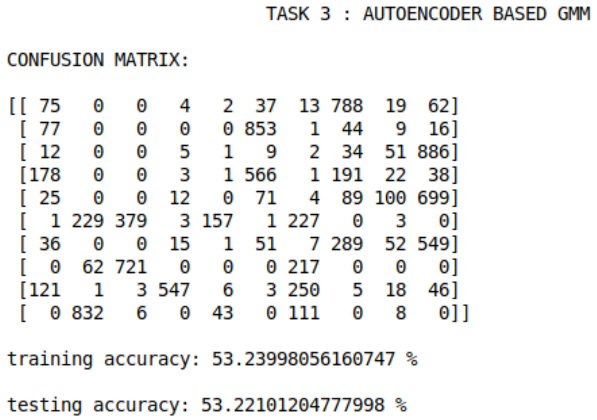
5.3 Task 2



```
                        TASK 2 : AUTOENCODER BASED K-MEANS

CONFUSION MATRIX:

[[  8   0 571  22   0   0   1   0 313  85]
 [  0   0   4  12   0 734   0   0  47 203]
 [ 15   0  12 607   0   0   0   0 357   9]
 [  4   0  30  49   0 142   0   0 192 583]
 [ 14   0   0 715   0   0   0   0 161 110]
 [ 15  32   0   0 204   0 411 214 124   0]
 [ 32   0 148 313   0   2   1   0 446  58]
 [  4 202   0   0 611   0 178   5   0   0]
 [693   2   1  31   2   0  62   0 172  37]
 [ 14 422   0   0  10   0  25 515  14   0]]

training accuracy: 52.980473194363576 %

testing accuracy: 52.76899300270792 %
```

Figure 6 – Autoencoder based KMeans

5.4 Task 3



```
                        TASK 3 : AUTOENCODER BASED GMM

CONFUSION MATRIX:

[[ 75   0   0   4   2  37  13 788  19  62]
 [ 77   0   0   0   0 853   1  44   9  16]
 [ 12   0   0   5   1   9   2  34  51 886]
 [178   0   0   3   1 566   1 191  22  38]
 [ 25   0   0  12   0  71   4  89 100 699]
 [  1 229 379   3 157   1 227   0   3   0]
 [ 36   0   0  15   1  51   7 289  52 549]
 [  0  62 721   0   0   0 217   0   0   0]
 [121   1   3 547   6   3 250   5  18  46]
 [  0 832   6   0  43   0 111   0   8   0]]

training accuracy: 53.23998056160747 %

testing accuracy: 53.22101204777998 %
```

Figure 8 – Autoencoder based KMeans

6 Conclusion

The end product of this project is a set clustering models. The first model is a K-Means clustering model built using Sklearns that has an accuracy of 51.28%. The second model is an autoencoder based K-Means model that has an accuracy of 52.76%. Finally, the third model is an autoencoder based Gaussian Mixture Model that has an accuracy of 53.22%.

This project has helped me apply my existing knowledge of clustering to practical use and gain deeper insights about unsupervised learning problems. It has bolstered my understanding of the different clustering models that I have implemented. I hope to further test my models on other datasets in the future.

Acknowledgement

I am extremely grateful to Professor Sargur Srihari for explaining all the necessary concepts of Unsupervised Learning and Clustering required to complete this project. I would also like to thank Mihir Hemant Chauhan, the senior Teaching Assistant for clarifying my doubts and making concepts clear in his recitation.