Cluster Analysis on Fashion MNIST dataset

Akash Kumar Roy
akashroy@buffalo.edu
Person Number: 50316991
University at Buffalo, State University of New York

Abstract: This is a report for K-Means Clustering Algorithm (Part 1), Auto Encoder with K-Means Clustering Algorithm (Part 2) and Auto Encoder with Gaussian Mixture Modeling (Part 3), performed on Fashion-MNIST dataset. Here in the first part we are applying K-Means clustering algorithm on the dataset and calculating the accuracy. This is implemented in python using the Sklearns library. In the second part we are building a Auto-Encoder Network with open-source neural-network library, Keras and applying K-Means algorithm on that encoded dataset. In the third part we are applying Gaussian Mixture Model on that encoded data set. At the end we are calculating Loss, Confusion Matrix, Heat Maps for each part and observing the relative strengths and weaknesses.

1. Introduction: Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses. Clustering is an unsupervised machine learning technique used to get an intuition about the structure of the data. It can be defined as the task of identifying subgroups in the data such that data points in the same subgroup (cluster) are very similar while data points in different clusters are very different. In other words, we try to find homogeneous subgroups within the data such that data points in each cluster are as similar as possible.

2. Dataset: The given Dataset for this Project is 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. and test data sets have 785 columns. The first column consists of the class labels and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image

3. Preprocessing: Preprocessing of Data is generally necessary to remove redundant data, handling Null values and remove multicollinearity. I have performed following operations for Preprocessing

For K Means Algorithm

A. I have Normalized the Data by dividing the whole matrix by 255. Because a single pixel has always a value between 0 to 255 and the Maximum value of a pixel can be only 255.

For Auto-Encoder using K-means Clustering

A. I have divided the Training set into Training Data and Validation data. Validation data in 10% of the training data.
B. Unflatten each row of the Training Data into 28*28 image matrix so we can use proper convolution to the images.

For Auto-Encoder using Gaussian Mixture Model Clustering (Preprocessing steps are same as Auto-Encoder using K-means Algorithm)

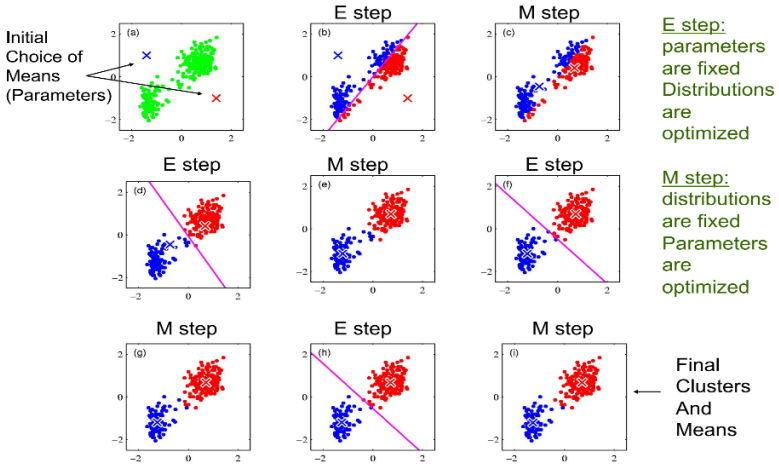
4. Architecture:

A. K-Means Algorithm

K means algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

Algorithm for K-means Clustering:

- Specify number of clusters K.
- Initialize centroids by randomly selecting K data points without replacement.
- Assign each data point to the closest cluster (centroid).
- Calculate the Loss
- Keep iterating until there is no change to the centroids.



Equations for K-Means Clustering:

- ***Objective Function for K-Means Clustering:***

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk} \,||\, \boldsymbol{x}_n - \mu_k \,||^2$$

- ***Determination of the indicator for K-means Clustering:*** As the objective function is a linear function of r_{nk} this optimization is performed easily

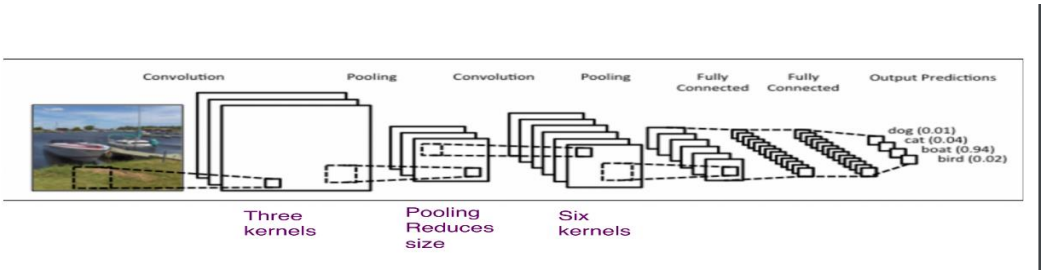
$$r_{nk} = \begin{cases} 1 \text{ if } k = \arg\min_j \,||\, \boldsymbol{x}_n - \mu_j \,||^2 \\ 0 \quad \text{otherwise} \end{cases}$$

A. Auto-Encoder with K-Means Clustering

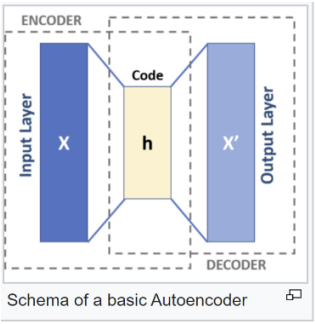
An Autoencoder neural network is an unsupervised Machine learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. They work by compressing the input into a latent-space representation and then reconstructing the output from this representation. An autoencoder is trained to attempt to copy its input to its output. Internally, it has a hidden layer that describes a code used to represent the input. Autoencoders are trained to preserve as much information as possible when an input is run through the encoder and then the decoder, but are also trained to make the new representation have various nice properties. Here I have implemented a convolutional autoencoder because the input objects are images. So, the encoding and decoding models will be convolutional neural networks instead of fully-connected networks. The Auto-Encoder Model Consists of two parts in my code:

- **Encoder Part:** This part of the network that compresses the input image into an encoded image by an Encoding Function. I have used Relu Function for this encoding.
- **Decoder Part:** This part of the network reconstructs the encoded image by using a decoding a function.

A Convolution Neural Network Architecture looks as following



An Auto-Encoder Architecture looks as following

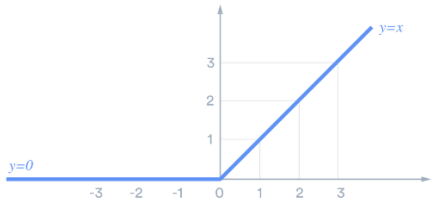


Equations and Functions for Auto Encoder Model:

- ***Loss Function: (Mean Squared Error)***: I have used Mean Squared Error as the Loss Function to train the Auto-Encoder Network

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

- ***Encoding and Decoding Functions used in CNN:***
    - **Relu Function**



    - **Sigmoid Function**

$$f(x) = \frac{1}{1 + e^{-x}}$$

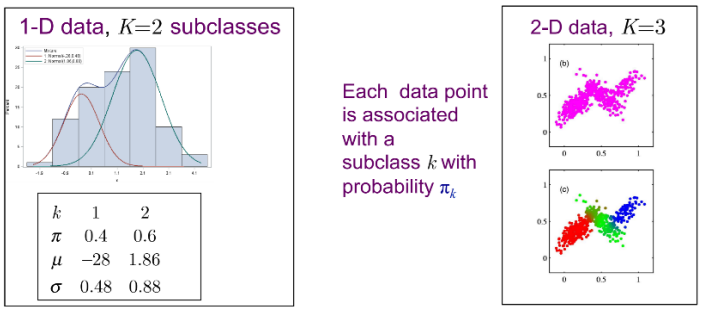
B. Auto-Encoder with Gaussian Mixture Model

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. Since subpopulation assignment is not known, this constitutes a form of unsupervised learning. A Gaussian Mixture is a function that is comprised of several Gaussians, each identified by $k \in \{1,\ldots, K\}$, where K is the number of clusters of our dataset. Each Gaussian k in the mixture is comprised of the following parameters:

- A mean μ that defines its center
- A covariance Σ that defines its width. This would be equivalent to the dimensions of an ellipsoid in a multivariate scenario.
- A mixing probability π that defines how big or small the Gaussian function will be.

The goal of the Gaussian Mixture Modelling is find the maximum likelihood parameters π, Σ, μ

Example of a Gaussian Mixture Model is as following:



Equations for Gaussian Mixture Modelling:

- Gaussian Mixture distribution is written as a linear superposition of K Gaussian Components and expressed as follows

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \Sigma_k)$$

5.Result:

- **K-Means Algorithm: (**Number of Cluster given: 10 ,Maximum Iterations: 300)

Calculated Accuracy: 51.26

To calculate accuracy, I have used the function

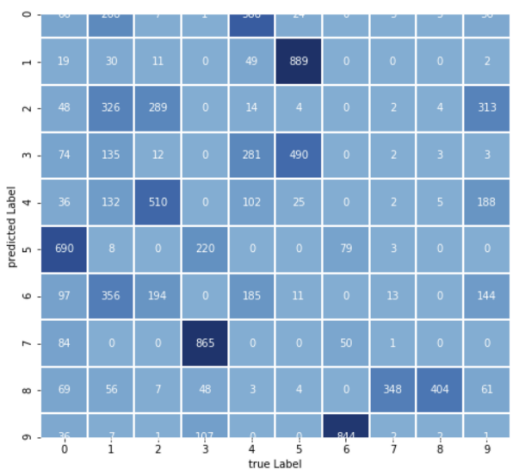
metrics.normalized_mutual_info_score(labels_true, labels_pred)

The Mutual Information is a function that measures the agreement of the two assignments, ignoring permutations. Normalized Mutual Information (NMI) is a normalization of the Mutual Information (MI) score to scale the results between 0 (no mutual information) and 1 (perfect correlation). In this function, mutual information is normalized by some generalized mean of H(labels_true) and H(labels_pred)), defined by the average method.

Confusion Matrix

```
[[ 66 268   7   1 588  24   0   5   5  36]
 [ 19  30  11   0  49 889   0   0   0   2]
 [ 48 326 289   0  14   4   0   2   4 313]
 [ 74 135  12   0 281 490   0   2   3   3]
 [ 36 132 510   0 102  25   0   2   5 188]
 [690   8   0 220   0   0  79   3   0   0]
 [ 97 356 194   0 185  11   0  13   0 144]
 [ 84   0   0 865   0   0  50   1   0   0]
 [ 69  56   7  48   3   4   0 348 404  61]
 [ 36   7   1 107   0   0 844   2   2   1]]
```

Heat Map

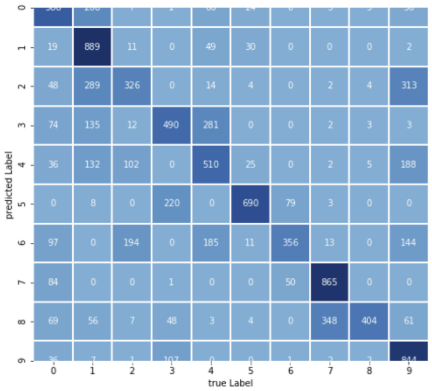


After mapping the labels to cluster Ids:

Confusion Matrix

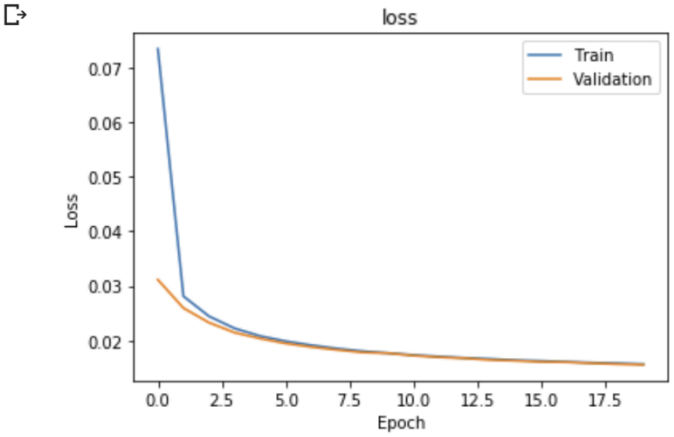
```
[[588 268   7   1  66  24   0   5   5  36]
 [ 19 889  11   0  49  30   0   0   0   2]
 [ 48 289 326   0  14   4   0   2   4 313]
 [ 74 135  12 490 281   0   0   2   3   3]
 [ 36 132 102   0 510  25   0   2   5 188]
 [  0   8   0 220   0 690  79   3   0   0]
 [ 97   0 194   0 185  11 356  13   0 144]
 [ 84   0   0   1   0   0  50 865   0   0]
 [ 69  56   7  48   3   4   0 348 404  61]
 [ 36   7   1 107   0   0   1   2   2 844]]
```

Heat Map



Auto-Encoder with K-Means Clustering: **Calculated Accuracy:** 53.48

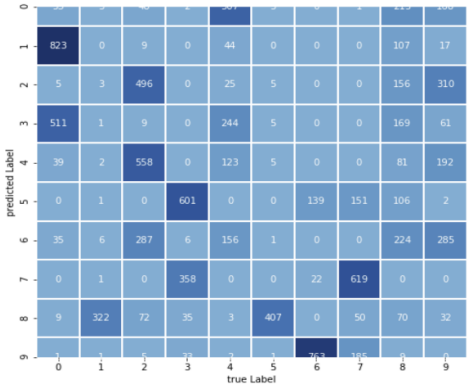
Graph of training loss and validation loss vs number of epochs while training for autoencoder



Confusion Matrix

```
[[ 53   3  48   2 507   3   0   1 215 168]
 [823   0   9   0  44   0   0   0 107  17]
 [  5   3 496   0  25   5   0   0 156 310]
 [511   1   9   0 244   5   0   0 169  61]
 [ 39   2 558   0 123   5   0   0  81 192]
 [  0   1   0 601   0   0 139 151 106   2]
 [ 35   6 287   6 156   1   0   0 224 285]
 [  0   1   0 358   0   0  22 619   0   0]
 [  9 322  72  35   3 407   0  50  70  32]
 [  1   1   5  33   2   1 763 185   9   0]]
```

Heat Map

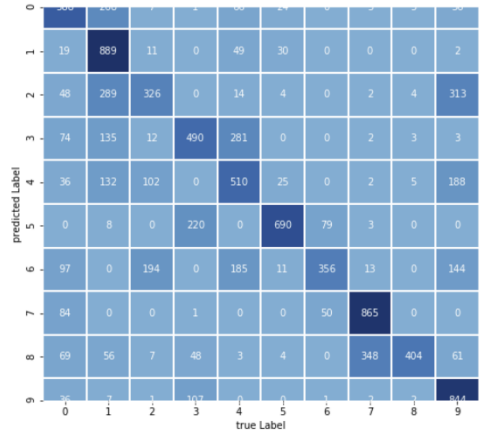


After mapping the labels to cluster Ids:

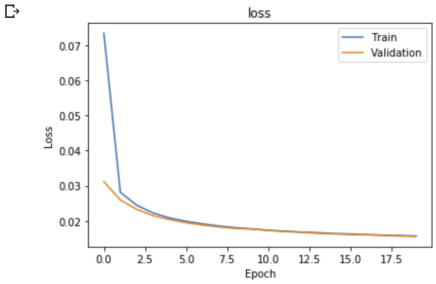
Confusion Matrix

```
[[507   3  48   2  53   3   0   1 215 168]
 [  0 823   9   0  44   0   0   0 107  17]
 [  5   3 496   0  25   5   0   0 156 310]
 [  0   1   9 511 244   5   0   0 169  61]
 [ 39   2 123   0 558   5   0   0  81 192]
 [  0   1   0   0   0 601 139 151 106   2]
 [ 35   6   0   6 156   1 287   0 224 285]
 [  0   1   0 358   0   0  22 619   0   0]
 [  9 322  72  35   3  70   0  50 407  32]
 [  1   1   5  33   2   1   0 185   9 763]]
```

Heat Map



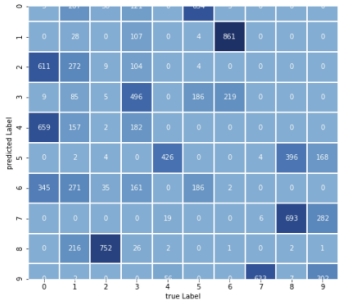
- **Auto-Encoder with Gaussian Mixture Modeling: Calculated Accuracy:** 58.26
  Graph of training loss and validation loss vs number of epochs while training for autoencoder



Confusion Matrix

```
[[  3 207  30 121   0 634   5   0   0   0]
 [  0  28   0 107   0   4 861   0   0   0]
 [611 272   9 104   0   4   0   0   0   0]
 [  9  85   5 496   0 186 219   0   0   0]
 [659 157   2 182   0   0   0   0   0   0]
 [  0   2   4   0 426   0   0   4 396 168]
 [345 271  35 161   0 186   2   0   0   0]
 [  0   0   0   0  19   0   0   6 693 282]
 [  0 216 752  26   2   0   1   0   2   1]
 [  0   2   0   0  56   0   0 633   7 302]]
```

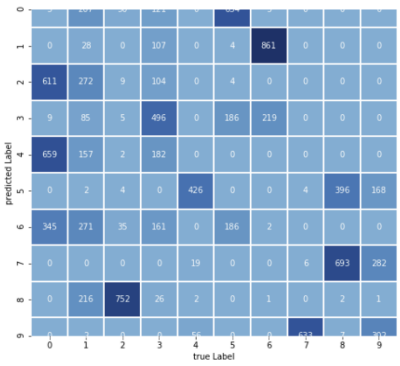
Heat Map



Confusion Matrix

```
[[634 207  30 121   0   3   5   0   0   0]
 [  0 861   0 107   0   4  28   0   0   0]
 [  9 272 611 104   0   4   0   0   0   0]
 [  9  85   5 496   0 186 219   0   0   0]
 [  0 157   2 182 659   0   0   0   0   0]
 [  0   2   4   0   0 426   0   4 396 168]
 [  2 271  35 161   0 186 345   0   0   0]
 [  0   0   0   0  19   0   0 693   6 282]
 [  0 216   2  26   2   0   1   0 752   1]
 [  0   2   0   0  56   0   0 302   7 633]]
```

Heat Map



6. Conclusion: After Observing the output of the three tasks we can see that we are getting maximum accuracy from Auto-Encoder with GMM Clustering. Also, we can observe Mixture models in general don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically

7. References

- Lecture Slides
- Bishop - Pattern Recognition And Machine Learning - Springer 2006
- Wikipedia
- Medium.com