Project Part 1: Density Estimation and Classification using Fashion-MNIST.

Objective:

The project focuses on Estimating parameters using MLE, implementing Naive Bayes and Logistic regression classification models. The tasks involved are:

1. Extract features average and standard deviation for both training and testing set.
2. Perform parameter estimation using Gaussian distribution.
3. Implement naive bayes classifier, logistic regression using gradient ascent and produce predicted labels for each testing sample.
4. Compute classification accuracies for both the Naive Bayes Classifier and Logistic Regression.

Dataset:

The dataset used for this project is from Fashion-MNIST dataset. It contains 70,000 images of articles of which 60,000 are the training data and 10,000 are for testing. We are using the images for "T shirt/Top" class and "Trouser" class. We have a total of 12000 images for training data and 2000 images for testing dataset. i.e.:

- Number of samples for training data: "T shirt": 6000, "Trouser": 6000
- Number of samples for testing data: "T shirt": 1000, "Trouser": 1000

Features:

The size of each image is 28 × 28 and there are in total of 784 pixels for an image. The first step is feature extraction. We are extracting 2 features for each image:

1. The average of all pixel values in the image.
2. The standard deviation of all pixel values in the image.

The size of the training dataset is 12000 × 2 and size of the testing dataset is 2000 × 2

Parameter Estimation:

Each class is following a 2D Gaussian Distribution. Here we are assuming that the features are independent to reduce the complexity. The parameters are estimated using a 1D Gaussian Distribution. Each feature can be represented by 1D Gaussian distribution since the features are independent of each other. We have in total two 1D Gaussian Distributions for a class, i.e:

Class	Feature 1	Feature 2
0	Avg (represented as 1D distribution)	Std (represented as 1D distribution
1	Avg (represented as 1D distribution)	Std (represented as 1D distribution)

The equations used for parameter estimation are,

$$\mu = \sum_{i=1}^{n} x_i / n$$

$$\sigma^2 = \sum_{i=1}^{n} (x_i - \mu)^2 \,/\, n - 1$$

the parameters were estimated using Maximum likelihood estimation for 1D Gaussian Distribution (eq (1)):

The estimated parameters are:

The Average value for class 0 feature 1 is 0.325607766439909.
The average value for class 0 feature 2 is 0.3200360871033629.
The Standard Deviation for class 0 feature 1 is 0.11337491460878085.
The Standard Deviation for class 0 feature 2 is 0.08798281005982794.
The Average value for class 1 feature 1 is 0.22290531462585023.
The average value for class 1 feature 2 is 0.333941712027219.
Standard Deviation for class 1 feature 1 is 0.05695100874843002.
Standard Deviation for class 1 feature 2 is 0.05703228654279648.

Naive Bayes Classifier:

We need to classify the given dataset using naïve bayes algorithm.
Here we are assuming that the samples are drawn from a gaussian distribution. We can estimate the probability of a given value using 1D Gaussian distribution. Equation for probability distribution function is:

- $\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma}} \; e^{\frac{-(x_i - \mu)^2}{2\sigma^2}}$ EQ (1)
- Where σ is standard deviation, μ is the mean for the samples.

According to Naïve Bayes, Probability that data belongs to a class,

$$p(y/x) = p(x/y) \times \mathrm{p}(y)$$

where $\mathrm{p}(y)$ – prior probability of a class
Here $\mathrm{p}(y)$ = 0.5 since the number of samples belonging to class 0 and class 1 are equal.
Prior probability p(y=0) = No of train samples belongs to class 0/ Total number of samples
Prior probability p(y=1) = No of train samples belonging to class 1/Total number of samples

Since we assumed that the features are independent, $p(x/y)$ becomes,

$$p(x/y) = p(x1/y) \times p(x2/y) \times 0.5$$

$$p(y/x) = p(x1/y) \times p(x2/y) \times 0.5$$

So, we compute probability of data belonging to class 0$\left(p\left(y=0/_{x}\right)\right)$ and probability of data belonging to class 1$\left(p\left(y=1/_{x}\right)\right)$ and assign data to the class which has maximum probability. $p\left(y=0/_{x}\right)$ can be computing by plugging the values of mean and standard deviation for the two features of class 0 (i.e. avg of pixel values, standard deviation of pixel values) in the 1D Gaussian distribution function. Similarly, we compute $p\left(y=1/_{x}\right)$. Then to assign the samples to their classes we compare their class probabilities.

i.e. If $p\left(y=0/_{x}\right) > p\left(y=1/_{x}\right)$ class = 0 else class = 1. In this way the test dataset has been classified and the accuracy for the model has been calculated by comparing the predicted labels with the original labels. The following accuracies have been observed.
The accuracy was computed using the equation no of samples correctly predicted / total samples.

Results

The accuracy for naïve bayes achieved is 83.15%
The accuracy for class 0 is 78.4%
The accuracy for class 1 is 87.9%

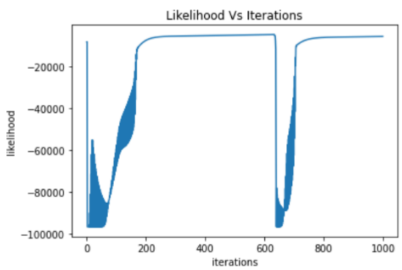
Logistic Regression Algorithm:

Logistic regression is implemented using gradient ascent algorithm. In logistic regression we learn $p(y/_{x})$ directly. It uses log-likelihood cost function. Sigmoid function is used to output predictions between o and 1.The steps involved are:

- Randomly initializing the weights.
    - For the given datasets 3 weights are initialized (bias is taken as w0) since there are two features.
- The predictions are done using sigmoid function. The equation for sigmoid function is:
    - $z = \sigma(w^T x)\ ,\ \sigma(t) = \frac{1}{1+e^{-t}}$ where $w - weights, x = samples$
- The cost function is calculated using log likelihood equation given by:
    - $\mathrm{L}(w) = \sum_{i=1}^{n}\left(y^i \log z^i + \left(1-y^i\right)\log\left(1-z^i\right)\right)$
- The gradient ascent algorithm is used for calculating the weights, it is used for finding the maxima, we need to find the weights which maximizes the cost function. The weights are updated in gradient ascent using the below equation:
    - $w_{t+1} = w_t + \eta\frac{dL(w)}{dw}$ where t – iteration number, $\eta - iteration\ number$
    - Applying the partial derivate on cost function using chain rule we get: $\frac{dL(w)}{dw} = (y-z)x^T$ where y actual value, z predicted value, x sample.

- In the gradient ascent algorithm, we choose a learning rate, and number of iterations, then compute the sigmoid function, log likelihood function, for each iteration and update the weights and bias till the iterations are completed by keeping track of log likelihood and weights at each iteration.

- For the given dataset a learning rate of 0.01 was chosen and numbers of iterations computed were 1000. Drawing a graph between iterations and log-likelihood function the maximum was occurring around the iterations 200-250 and it remained flat after some time. Hence the weights obtained at the iteration were chosen to compute the predicted values.

- 

- After calculating the predicted value, we choose a threshold to classify values into two classes. The threshold value chosen here is 0.5.
    - If predicted value > 0.5 It is determined as class 1 otherwise it is taken as class 0.
- The accuracy is computed by comparing the predicted values with the actual values and the following accuracies were observed. The accuracy was computed using the equation no of samples correctly predicted / total samples.

  Results
- The overall accuracy observed for logistic regression is 92.3%
- The accuracy for class 0 is 93.2%
- The accuracy for class 1 is 91.4%

Conclusion:

The features average and standard deviation were computed for the dataset and the parameters were estimated. The classification has been done using Naïve Bayes and Logistic Regression. Logistic Regression has given more accuracy than that of Naïve Bayes and among the classes Naïve Bayes has given more accuracy for classifying T shirt class than that of Trouser class whereas Logistic regression has given more accuracy for Trouser class than T shirt class.