# Statistical Machine Learning – Project Part 1

**Density Estimation and Classification**

**Dataset Used:**

Dataset used for training and testing data: MNIST dataset for handwritten images of digits.

The digits considered for this section were all training and testing images for the digits ‘7’ and ‘8’. There are 12116 images for training (6265: ‘7’; 5851: ‘8’) and 2002 images for testing (1028: ‘7’; 974: ‘8’).

**Using Naïve Bayes Theorem:**

|  |  |  |
| --- | --- | --- |
| **Accuracy for number 7** | **Accuracy for number 8** | **Overall accuracy** |
| **96.70 %** | **89.83 %** | **93.36 %** |

Naïve Bayes uses Bayes theorem to calculate the probability in the testing case. It simplifies computation by considering all features to be independent of each other. This drastically reduces the number of calculations required for computing the probability. However, this assumption is risky as it is very rare to find cases where features are independent.

The training data was fit into a multivariate Gaussian distribution. The resulting model is used on the testing data set to identify the tag that should be given to the figure. Each tag is given a probability with respect to the image and the tag with the highest value is attached to the image. On prediction and cross checking with the actual labels, it was noted that the model accurately identified the tag 93.36% of the time. ‘7’ was tagged with 97% accuracy [994 / 1028] (the training set had a greater number of images of ’7’), while ‘8’ was tagged with 90% accuracy [875 / 974].

The method used to get the training model involved taking the mean and standard deviation of each pixel of a set of images. These were done using the standard numpy functions available in Python. These values were put into the multivariate Gaussian function shown below:

The final probability will consider how each pixel from the testing set compares with those of the training set, with a given label. The label with the maximum probability attached to it is considered as the label for the testing image.

**Using Logistic Regression:**

|  |  |  |
| --- | --- | --- |
| **Accuracy for number 7** | **Accuracy for number 8** | **Overall accuracy** |
| **98.63 %** | **97.95 %** | **98.30 %** |

Logistic regression addresses a common problem with logistic regression by using a sigmoid function to separate and classify the data. In this case, the given image must be classified as either an image of ‘7’ or an image of ‘8’.

The sigmoid function is used to map the predicted values to probabilities of the image based on the input value. It can be written as:

Weights are initial values given to the probabilities for each image. Using the formula for gradient descent, the values are converged to their global minima. The learning rate helps improve the rate of convergence of the weights. These weights are finally fed into the sigmoid function to get the probabilities of the images. The gradient descent function is defined as:

(Note: I was unable to correctly implement gradient ascent. This is the reason I used gradient descent)

On prediction and cross referencing with the original testing labels, it was noted that the model accurately identified the tag 98.30% of the time. ‘7’ was tagged with 98.63% accuracy [1014 / 1028], while ‘8’ was tagged with 97.95% accuracy [954 / 974]. One interesting difference to not in this experiment is how the differences in the accuracy for both was not as big in the case of logistic regression as it was for Naïve Bayes Classification. This could suggest a lower dependency on the quantity of the training dataset in the case of Logistic regression.

**Conclusion:**

Two classification methods, Naïve Bayes and Logistic Regression were used to train a model using a provided data set and then correctly classify the testing dataset based on this model. Through testing, it was found that the Logistic Regression had a higher accuracy when compared to Naïve Bayes Classification (98.30% to 93.36%). This shows that Logistic Regression is a better model for classification than Naïve Bayes Classification.