Project 1: Density Estimation and Classification

CSE 575: Statistical Machine Learning

Arizona State University – Summer 2022

Course Instructor: Masudul Quraishi

Submitted By:

Aishwarya Baalaji Rao

ASU ID: 1222423228

Date of Submission: 29th May 2022

ABSTRACT: This document serves as the report to the first part of the project. In this part, we implement density estimation and image classification on the popular handwritten digit dataset - MNIST, using the Naïve Bayes and Logistic Regression classifiers. The Naïve Bayes classifier utilizes two features – mean and standard deviation to train the model. The logistic regression model uses Maximum Likelihood Estimation and gradient ascent method to train the model. Finally, we compute the testing accuracy for both classifiers and compare the results.

KEYWORDS: Maximum Likelihood Estimation, Naïve Bayes Classifier, Logistic Regression, Gaussian distribution, Gradient Ascent.

1. Introduction

Image classification is one of the most important features of statistical machine learning with numerous and vast amounts of practical applications. The aim of density estimation in this project phase is to infer the probability density function from observations of random independent variable(s). Knowing density estimation is important since we can know the extent of unlikeliness of that event from happening, so we can appropriately handle it by choosing an optimal learning method.

2. Problem Definition and Algorithm

2.1 Task Definition

In this part, image classification is performed on the MNIST (handwritten digits) dataset, which consists of 14,118 images in total. The breakdown of training and testing samples are as follows:

Number of samples in the training set: "7": 6265; "8": 5851.

Number of samples in the testing set: "7": 1028; "8": 974

The experiment will involve using Naïve Bayes and Logistic Regression classifiers to predict the results and will end with the comparison of accuracies between the two models.

2.2 Algorithm Definition

A naïve Bayes classifier is an algorithm that classifies using Bayes' theorem. Naïve Bayes classifiers assume substantial independence, or naive independence, between data point characteristics. These classifiers are commonly used in machine learning due to their ease of implementation. Naive Bayes is often referred to as straightforward Bayes or independence Bayes.

When the variable is categorical (binary), logistic regression is often the preferred algorithm for image classification. Logistic regression is a predictive analysis and is used to describe data and explain the connection between a binary dependent variable and one or more nominal, ordinal, or interval independent variables. Often logistic regressions are difficult to comprehend or interpret.

3. Experimental Evaluation

3.1 Methodology

The features extracted for the given normal distribution dataset, for each image (array) in the training dataset, are - mean and standard deviation. Estimated values for both the features for the combined dataset are as follows:

```
Mean is:  0.13234183982239325
STD is:  0.15839592414507134
```

3.1.1 Naïve Bayes Classifier

Naïve Bayes is a quite sought-after supervised machine learning model which is based off the Bayes' theorem. It also computes fast since it solely works on probabilities. Bayes' Theorem equation is as follows: **P(A|B) = P(B|A) * P(A) / P(B)**, where P(A|B) is the posterior probability of A *given* B, P(B) is the probability of event B, P(B|A) is the likelihood of event B happening *given* A and P(A) is the prior probability. The result of prediction is the class with the greatest posterior probability.

The ***Gaussian Density Function*** with mu as mean, and sigma as standard deviation is as follows:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

The ***predicted label***, the maximum of the posterior values, is computed as follows:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(\mathrm{y}) \prod_{i=1}^{d} p(\mathrm{x}_i|y)$$

The next step after calculating the prior and conditional probabilities is to calculate the posterior as follows: ***posterior = prior probability + conditional probability.*** Finally, we use all the above helper functions to write the predict function and calculate the accuracy of the model.

→***Accuracy Result***: The Naïve Bayes model gave an accuracy of **83.62%** on the MNIST test dataset for digit 7 and 8 images.

3.1.2 Logistic Regression Classifier

Logistic Regression is the classification approach for machine learning that is utilized in predictive analysis. The primary distinction between logistic regression and linear regression is the cost function. Logistic Regression employs the log-likelihood Cost function (equivalent to finding the Maximum Likelihood Estimation, MLE), which is much more sophisticated than the mean squared error (MSE) cost function. This function is based on probability theory and for a single training input (x, y), the function assumes the following:

$$P(Y = 1 \mid \mathbf{X} = \mathbf{x}) = \sigma(\theta^T \mathbf{x})$$
$$P(Y = 0 \mid \mathbf{X} = \mathbf{x}) = 1 - \sigma(\theta^T \mathbf{x})$$

The hypothesis, which is Sigmoid function between 0 and 1, is then used to map the probabilities obtained from the function using the above assumptions.

a. **Sigmoid Function**: The sigmoid function in machine learning, is a "mathematical logistic function". The formula used to compute the sigmoid function in the code is as follows:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

This function accepts a numpy array as input and returns a numpy array with mapped probabilities between 0 and 1. The sigmoid function is performed on the numpy array that is the weighted average of the samples, using numpy dot function.

b. **Cost Function**: In gradient ascent, we must maximize the cost function, which in linear regression is essentially the loss function. Cost functions are helpful to know the extent of deviation between the predicted and actual value. In gradient ascent, the difference between them is maximized, aka, Log-Likelihood Estimation or Maximum Likelihood Estimation (MLE). Therefore, we pass the actual value and the predicted values as inputs to the cost function. *Log Likelihood* is computed with the following formula:

$$\text{Log likelihood} = \sum[y_i ln\hat{y}_i + (1 - y_i)\ln(1 - \hat{y}_i)]$$

In the code, it is possible to run into log (0) with undefined or NaN value. To avoid this, a very small minimum value is assigned to the standard deviation value to avoid the undefined condition.

c. **Gradient ascent**: The algorithm to combine the sigmoid function and the log-likelihood function to give the desired result is the gradient ascent algorithm. Gradient Ascent is an iterative approach for locating local maxima of a differentiable function. The method advances in the direction of the gradient computed at every point of the cost function curve until the halting criterion is met. The formula used to maximize the log-likelihood results, on each parameter, is as follows:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta \nabla_{\mathbf{w}^{(k)}} l(\mathbf{w})$$

→ ***Accuracy Result***: The Logistic Regression classifier is giving an *accuracy* of **96.80%** on the test dataset, outperforming the Naïve Bayes classifier. Learning rate = **0.01**.

4. Results and Conclusion

4.1 Naïve Bayes Classifier Results

The Naïve Bayes classifier gave the least among the two models in prediction accuracy on the MNIST dataset with a testing accuracy of **83.62%**. One other thing to note was that the Bayes classifier computed significantly faster than the Logistic Regression model, which gave it an advantage in this aspect.

```
(base) Aishwaryas-MacBook-Pro:Statistical Machine Learning aishwaryabaalajirao$ python main.py
********** Results on MNIST Dataset (for Digits 7 and 8) **********
Data Specs:
Training Data Size: 12116
Testing Data Size: 2002
****************************************
Training NB Model ....
Prior [0.51708485 0.48291515]
Mean is:  0.13234183982239325
STD is:  0.15839592414507134
Naive Bayes Classifier Training Accuracy:  85.87817761637504
Naive Bayes Classifier Testing Accuracy:  83.61638361638362
****************************************
```

4.2 Logistic Regression Classifier Results

Logistic Regression model gave a much better accuracy than the Naïve Bayes model on the test dataset with – **96.80%**. But the computation speed was lower than the Naïve Bayes. Learning rate was taken as **0.01** by trial-and-error method.

```
****************************************
Training LR Model ....
Accuracy of LR:  96.8031968031968
```

5. Software Specifications

1. Python version: 3.9.12 **2.** Python IDE used: Sublime Text Editor **3.** Run environment: Terminal

References

1. https://scikit-learn.org/stable/modules/naive_bayes.html#gaussian-naive-bayes
2. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
3. https://en.wikipedia.org/wiki/MNIST_database