Course Title Portfolio

Name
Email

***Abstract*—This document This document.**

Keywords—mean, standard deviation, variance, probability density function, classifier

I. Introduction

This document This document This document This document This document This document This document This document This document This document This document This document This document This document This document This document. [1].

This project practiced the use of density estimation through several calculations via the Naïve Bayes Classifier. The data for each equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling 'numpy.mean()digit 0 or digit 1. The test images were then classified based on the previous calculations and the accuracy of the computations were determined.

The project consisted of 4 tasks:

A. Extract features from the original training set

There were two features that needed to be extracted from the original training set for each image. The first feature was the average pixel brightness values within an image array. The second was the standard deviation of all pixel brightness values within an image array.

B. Calculate the parameters for the two-class Naïve Bayes Classifiers

Using the features extracted from task A, multiple calculations needed to be performed. For the training set involving digit 0, the mean of all the average brightness values was calculated. The variance was then calculated for the same feature, regarding digit 0. Next, the mean of the standard deviations involving digit 0 had to be computed. In addition, the variance for the same feature was determined. These four calculations had to then be repeated using the training set for digit 1.

C. Classify all unknown labels of incoming data

Using the parameters obtained in task B, every image in each testing sample had to be compared with the corresponding training set for that particular digit, 0 or 1. The probability of that image being a 0 or 1 needed to be determined so it can then be classified.

D. Calculate the accuracy of the classifications

Using the predicted classifications from task C, the accuracy of the predictions needed to be calculated for both digit 0 and digit 1, respectively.

Each equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling 'numpy.mean()of the data. These features helped formulate the probability density function when determining the classification.

II. Description of Solution

This project required a series of computations in order to successfully equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling 'numpy.mean(). Once acquiring the data, the appropriate calculations could be made.

A. Finding the mean and standard deviation

The data was provided in the form of NumPy arrays, which made it useful for performing routine mathematical operations equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was

determined by calling ‘numpy.mean()by calling ‘numpy.std()’, another useful NumPy function. These extracted features from the training set for digit 0 also had to be evaluated from the training set for digit 1. Once all the features for each image were obtained from both training sets, the next task could be completed.

$$\bar{x} = \frac{1}{n}\Sigma x_i$$

Equ. 1. Mean formula

B. Determining the parameters for the Naïve Bayes Classifiers

To equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling ‘numpy.mean() and the array of the standard deviations created for digit 1.

$$\sigma^2 = \frac{1}{N}\Sigma(X - \mu)^2$$

Equ. 2. Variance formula

This equation was used to find the probability of the mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling ‘numpy.mean()’ for each image in the set. In addition, the standard deviation of the pixel brightness values was calculated for each image by calling ‘numpy.std()’, another useful NumPy function. These extracted features from the training. This was multiplied by the prior probability, which is 0.5 in this case because the value is either a 0 or a 1.

This]. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling ‘numpy.mean()’ for each image in the set. In addition, the standard deviation of the pixel brightness values was calculated for each image by calling ‘numpy.std()’, another useful NumPy function. These extracted features from the training.

This entire procedure had to be conducted once again but utilizing the test sample for digit 1 instead. This meant finding the mean and standard deviation of each image, using the probability density function to calculate the probability of the mean and probability of the standard deviation for digit 0, and calculating the probability that the image is classified as digit 0. The same operations had to be performed again, but for the training set for digit 1. The probability of the image being classified as digit 0 had to be compared to the probability of the image being classified as digit 1. Again, the larger of the two values suggested which digit to classify as the label.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

C. Determining the accuracy of the label

The mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling ‘numpy.mean()’ for each image in the set. In addition, the standard deviation of the pixel brightness values was calculated for each image by calling ‘numpy.std()’, another useful NumPy function. These extracted features from the by the total number of images in the test sample for digit 1.

III. RESULTS

mean for. Without using a built-in function, the first feature, the mean, could be calculated using the equation in Fig. 1. The second feature, the standard deviation, could be calculated using the equation in Fig. 2. Utilizing the training set for digit 0, the mean of the pixel brightness values was determined by calling ‘numpy.mean()’ for each image in the set. In addition, the standard deviation of the pixel brightness values was calculated for each image by calling ‘numpy.std()’, another useful NumPy function. These extracted features from the also higher.

TABLE I. TRAINING SET FOR DIGIT 0

TTTTTTTT	000000
XXXXX	000000

When comparing the test images, the higher values of the means and the standard deviations typically were labeled as digit 0 and the lower ones as digit 1. However, this was not always the case because then the calculated accuracy would then be 100%.

The e. After classifying all the images in the test sample for digit 0, the total amount predicted as digit 0 was 899. This meant that the accuracy of classification was 0000%, which can be represented in Fig. 5.

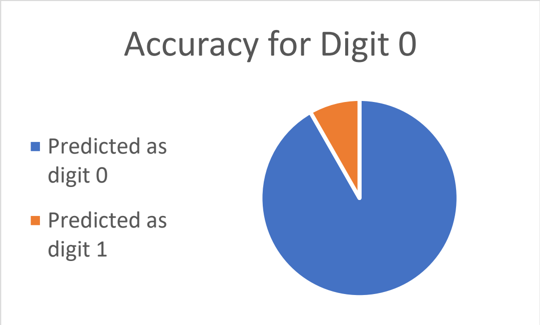

Fig. 1. Accuracy of classification for digit 0

The total amount of images in the test sample for digit 1 0000. After classifying all the images in the test sample for digit 1, the total amount predicted as digit 00000. This meant that the accuracy of classification was 00000%, which can be represented in Fig. 6.

IV. Lessons Learned

The procedures practiced in this project required skill in the Python programming language, as well as understanding concepts of statistics. It required plenty of practice to implement statistical equations, such as finding the mean, the standard deviation, and the variance. My foundational knowledge of mathematical operations helped me gain an initial understanding of how to set up classification problems. My lack of understanding of the Python language made it difficult to succeed initially. Proper syntax and built-in functions had to be learned first before continuing with solving the classification issue. For example, I had very little understanding of NumPy prior to this project. I learned that it was extremely beneficial for producing results of mathematical operations. One of the biggest challenges for me was creating and navigating through NumPy arrays rather than a Python array. Looking back, it was a simple issue that I solved after understanding how they were uniquely formed. Once I had a grasp on the language and built-in functions, I was able to create the probability density function in the code and then apply classification towards each image.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

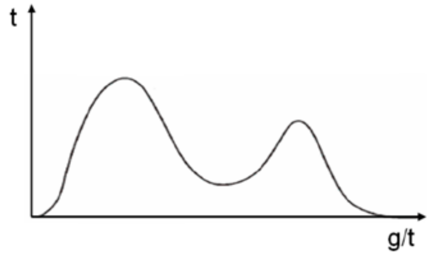

Fig. 2. Bimodal distribution example [5]

Upon completion of the project, I was able to realize that One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the bell is where the mean of the data is located [4]. A bimodal distribution is one that displays two bell-shaped distributions on the same graph. After calculating the features for both digit 0 and digit 1, the probability density function gave statistical odds of that particular image being classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

One aspect of machine learning that I understood better after completion of the project was Gaussian distribution. This normalized distribution style displays a bell-shape of data in which the peak of the the project was Gaussian distribution. This normalized distribution style the project was Gaussian distribution. This normalized distribution style bell is where the mean of the data is located [4]. A bimodal distribution is one that displays classified under a specific bell-shaped curve. An example of a bimodal distribution can be seen in Fig. 7 below.

V. References

[1] N. Kumar, *Naïve Bayes Classifiers*, GeeksforGeeks, May 15, 2020. Accessed on: Oct. 15, 2021. [Online]. Available: https://www.geeksforgeeks.org/naive-bayes-classifiers/

[2] J. Brownlee, *How to Develop a CNN for MNIST Handwritten Digit Classification*, Aug. 24, 2020. Accessed on: Oct. 15, 2021. [Online]. Available: https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/

[3] "*What is NumPy*," June 22, 2021. Accessed on: Oct. 15, 2021. [Online]. Available: https://numpy.org/doc/stable/user/whatisnumpy.html

[4] J. Chen, *Normal Distribution*, Investopedia, Sept. 27, 2021. Accessed on: Oct. 15, 2021. [Online]. Available: https://www.investopedia.com/terms/n/normaldistribution.asp

[5] "*Bimodal Distribution*," Velaction, n.d. Accessed on: Oct. 15, 2021. [Online]. Available: https://www.velaction.com/bimodal-distribution/