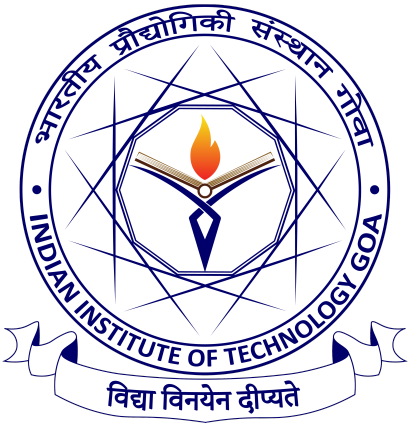



Indian Institute of Technology Goa

CS230: Probability and Statistics for Computer Science

# Handwritten Data Classifier

*Author:*
Nandni Mawane

*Roll Number:*
2003315

October 19, 2021

# Contents

# 1 Abstract

In the handwritten data classifier project, Naive Bayes was used to make predictions on handwritten letters.The code was written in Python, on Jupyter Notebook. The accuracy score was found to be 70.356%, which is quite decent for a Naive Bayes classifier.The classifier gave the most accurate predictions for the letter 'O' and then, 'S'.

# 2 Introduction

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. Bayes' theorem was named after the Reverened Thomas Bayes, who studied how to compute a distribution for the probability parameter of a binomial distribution. After Bayes' death, his friend Richard Price edited and presented this work in 1763, as An Essay towards solving a Problem in the Doctrine of Chances. So it is safe to say that Bayes classifiers have been around since the 2nd half of the 18th century.

In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naïve Bayes is not (necessarily) a Bayesian method.

$$P(H|e) = \frac{P(e|H)P(H)}{P(e)} \tag{1}$$

Likelihood determines how probable the evidence is given our hypothesis is true. Prior determines how probable our hypothesis was before observing the evidence. Posterior determines how probable our hypothesis is given the observed evidence. Marginal determines how probable the new evidence is under all possible hypotheses. $P(e) = \sum P(e|H_i)P(H_i)$

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

In this project, a dataset of 372,450 images is used. The dataset is split into training and test datasets. The training dataset has 335,205 images, whereas the test dataset has 37,245 images. Prior and class conditional densities are calculated for the training dataset, which are then used while making predictions for the test data.

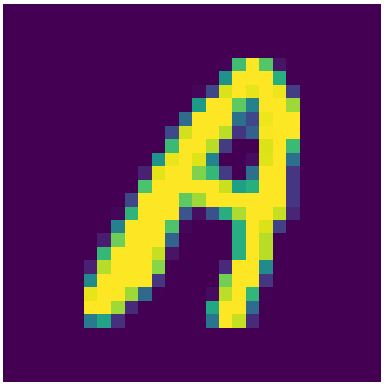


Diagram 1: An image of the dataset.

# 3 Implementation

The classifier is aimed at predicting labels for test images. The dataset is available in the form of a comma separated file (.csv extension), where each row has 784 elements carrying the pixel values of a $28 \times 28$ image. The images are labeled from 0 to 25 where, 0 stands for 'A', 1 for 'B' and so on. The first column of the csv file contains labels. The labels and features are extracted from the dataset. The dataset is then split into training and testing datatsets.

Each image is represented as an array of 784 pixel values. Each pixel value is denoted by $x_i$ where i ranges from 0 to 783.

Every pixel value greater than 127.5 is treated as 1 and those less than or equal to 127.5 are treated as 0. Prior and class conditional densities are calculated for the training dataset and stored in a file for later use. Prior is an array of 26 elements with the $k^{th}$ element containing the value of $P(Y = k)$. Class conditional density is a 2D array of dimension $26 \times 784$, where $ccd[k][i] = P(x_i = 1|Y = k)$.

When an image is to be tested, it is first preprocessed where, each pixel value greater

than 127.5 is replaced with 1 and the others with 0.

$$\hat{y} = \underset{k \in \{0,..,25\}}{\operatorname{argmax}}\, p(Y = k) \prod_{i=0}^{783} p(x_i | Y = k) \tag{2}$$

Equation of a Naive Bayes Classifier

Here, $\hat{y}$ is the predicted label for the given test image. $P(Y = k)$ is stored on the $k^{th}$ index of the prior array. The conditional probabilities are calculated from the stored class conditional densities.

# 4 Theory

Let X be the vector $(x_0, x_1, x_2, ....., x_{783})$, representing a test image.

$$p(Y = k|X) = \frac{p(Y = k)\, p(X|Y = k))}{\sum_{i=0}^{25} p(Y = i)\, p(X|Y = i)} \tag{3}$$

Formula for Bayes' theorem

$$p(X|Y = k) = \prod_{i=0}^{783} p(X_i = x_i|Y = k) \tag{4}$$

Naive Bayes' independence assumption

$$p(Y = k|X) = \frac{p(Y = k) \prod_{j=0}^{783} p(X_j = x_j|Y = k)}{\sum_{i=0}^{25} [p(Y = i) \prod_{j=0}^{783} p(X_j = x_j|Y = k)]} \tag{5}$$

Using equation 4 in equation 3.

# 5 Handling Corner Cases

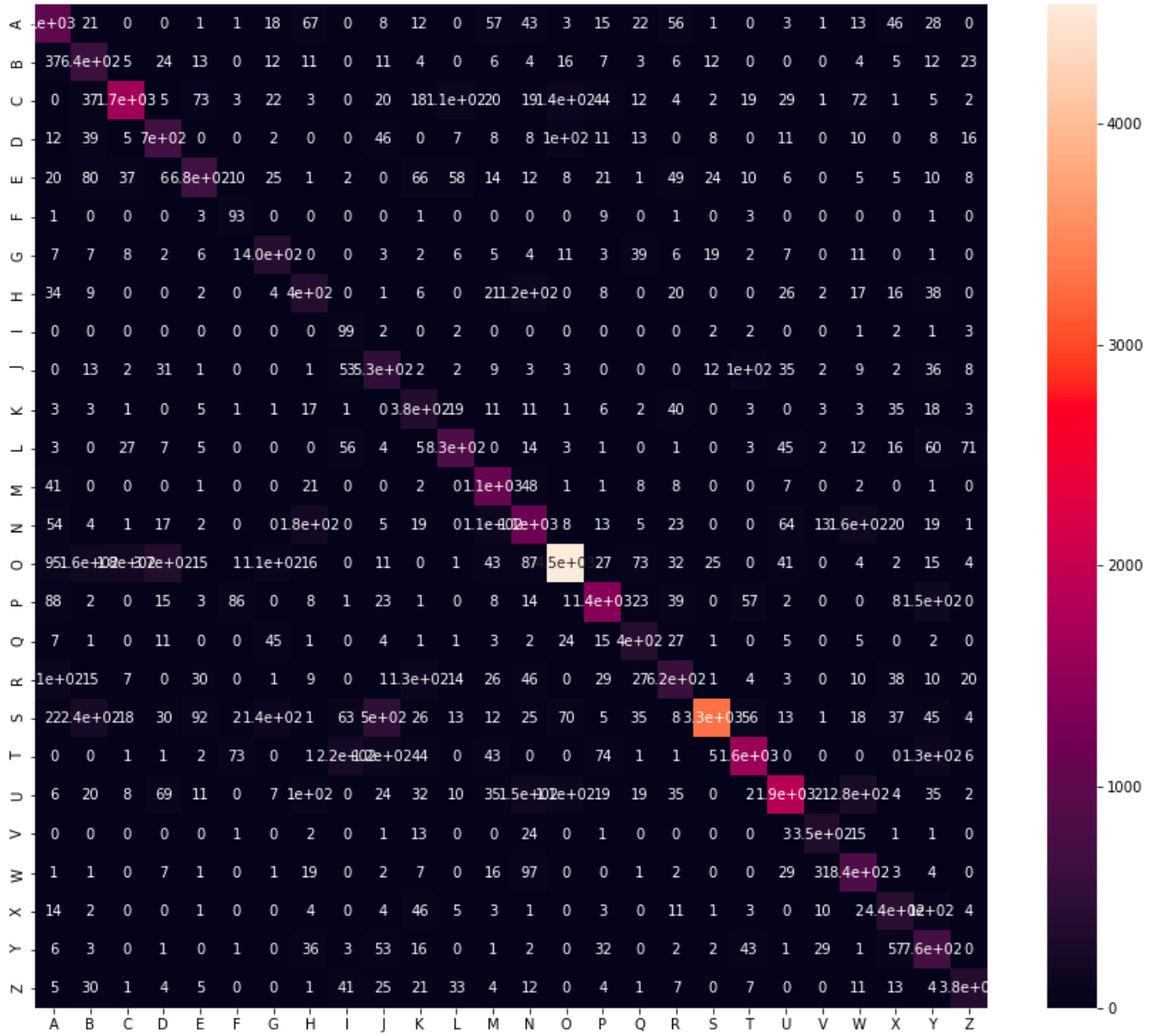
Factoring in the possibility of conditional probabilities being close to zero, calculation of their continued product would possibly throw a range error. To avoid this, natural logarithm has been used to calculate the continued product.

$$S = a.b.c.d \Rightarrow S = e^{ln(a)+ln(b)+ln(c)+ln(d)} \tag{6}$$

The second corner case arises when for the test image, the value of a particular pixel is different from the ones seen during training. This gives rise to the problem of zero probability in Naive Bayes. To tackle this Laplace Smoothing has been used.

For each $Y = k \; \forall \; k \in \{0, 1, 2, ..., 25\}$, two images in addition to the training images are considered. One of these 2 images has every pixel value 0 and the other has every pixel value 1.

# 6 Performance Evaluation



Heatmap: On the x-axis are the true labels, while the y-axis houses the predicted labels.

From the heatmap, confusion matrix can be visualised, which helps us comment on the accuracy of the Naive Bayes classifier. Higher the values on the diagonal, the better the model's accuracy. As can be seen, the letter 'B' is often confused with the letter 'S' and the letter 'W' with 'U'. The letter 'O' is most accurately predicted.

Note: The heatmap due to size restrictions looks cluttered here, refer to the code file attached herewith for a clear view of the heatmap (present under the head 'Performance Evaluation'): Code File

The model has an accuracy score of **70.356%**.

# 7 References

[1] https://stats.stackexchange.com/questions/18212/origin-of-the-na%C3%AFve-bayes-classifier: :text=A%20naive%20Bayes%20classifier%20is,parameter%20of%20a%20binomial%20distribution

[2] https://en.wikipedia.org/wiki/Naive$_{B}ayes_{c}lassifier$

[3] https://medium.com/@mark.rethana/bayesian-statistics-and-naive-bayes-classifier-33b735ad7b16

[4] https://towardsdatascience.com/laplace-smoothing-in-na%C3%AFve-bayes-algorithm-9c237a8bdece