Algorithm 1. Naive Bayes' Classifier

The Naive Bayes classifier is a popular machine learning algorithm used for classification tasks. It is based on the principles of Bayes' theorem, which provides a probabilistic framework for making decisions and predictions. At its core, the Naive Bayes classifier is designed to estimate the probability of a particular outcome or class given a set of input features.

Bayes' Theorem:

Bayes' theorem allows us to calculate the probability of an event based on prior knowledge of conditions that might be related to that event. It is expressed as:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

Where:

- P(A|B) is the probability of event A occurring given that event B has occurred.
- P(B|A) is the probability of event B occurring given that event A has occurred.
- P(A) is the prior probability of event A.
- P(B) is the prior probability of event B.

In the Naive Bayes classifier, we use this theorem to estimate the probability of a particular class (A) based on the observed features or attributes (B).

Working of Naive Bayes' Classifier:

The Naive Bayes classifier works by making a "naive" assumption that the features used for classification are conditionally independent, given the class. In other words, it assumes that the presence or absence of one feature does not affect the presence or absence of another feature. While this assumption is not always true in practice, Naive Bayes can still be surprisingly effective in many real-world scenarios.

"Naive" assumption made by the classifier:

$$P(x_1,\ldots,x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

The key components of the Naive Bayes algorithm include:

- Prior Probabilities: These are the initial probabilities of each class occurring before observing any data. The prior probabilities are estimated based on the frequency of each class in the training data.

- Likelihood Probabilities: These are the probabilities of observing the features given a particular class. In the context of Bayes' theorem, this is represented as P(B|A), which signifies the likelihood of seeing the features B when the class is A. These likelihood probabilities are estimated from the training data.

- Posterior Probabilities: The Naive Bayes classifier calculates the posterior probabilities P(A|B) and P(B|A) for each class based on the prior probabilities and likelihood probabilities. It then assigns the class with the highest posterior probability as the predicted class for the given set of features.

Dataset used:

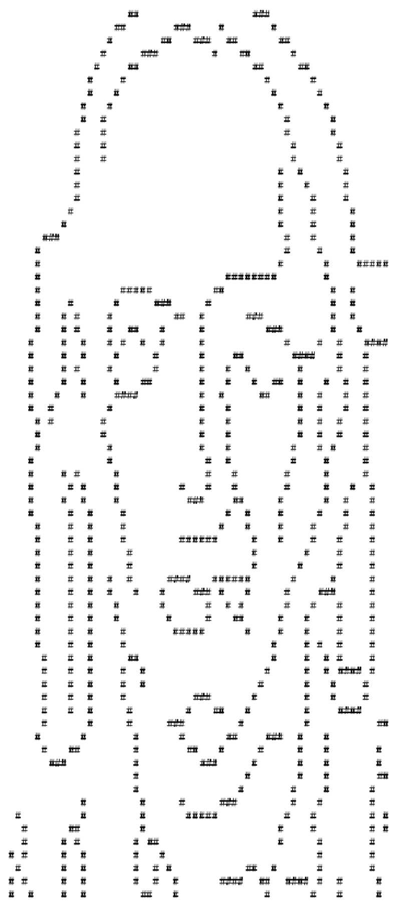
In order to illustrate the working of the Naive Bayes' classifier, we have used a dataset of images which are constructed in the form of a 70x60 grid of # and 'blank' characters. In other words, each of the 420 positions in our image grid holds either a # character or a blank character. Each of the image is classified as a face, or a non-face image in our training dataset. The labels are 0(for non-face) and 1(for face) for each image. Our training dataset is composed of 150 training examples.

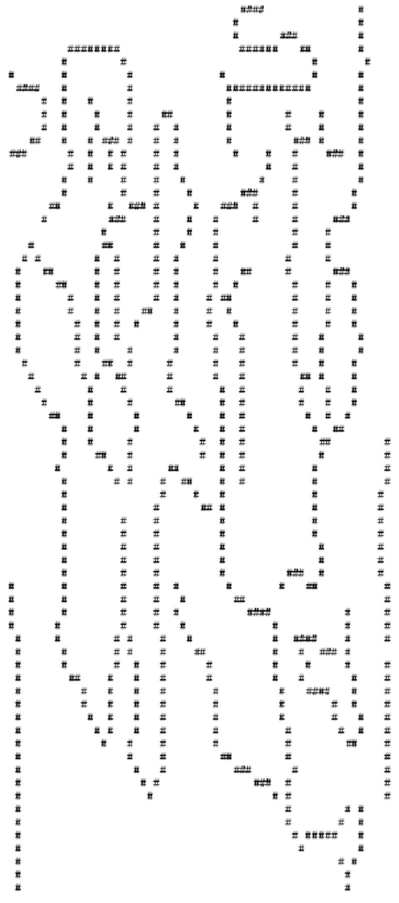
In the dataset, the number of face images is 73 and the number of non-face images is 78. Thus, we have the following prior probabilities :-

P(face) = 73 / 150 = 0.487

P(non-face) = 78 / 150 = 0.513

Example of face image **Example of non-face**





Pooling:

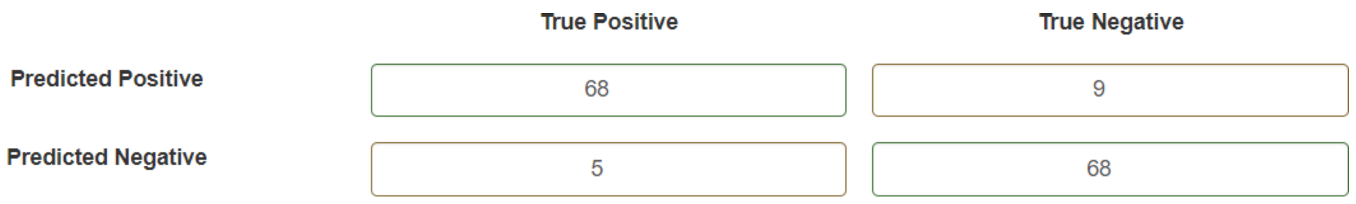
We have used a pseudo variable with a value of 1.9 to perform pooling. By adding the pseudo variable and dividing by the total count plus the pseudo value, we avoid division by zero and ensure that even events with zero counts still have a non-zero probability estimate. This helps in preventing the Naive Bayes classifier from giving zero probabilities, which can cause problems during classification.

For example, for a particular pixel out of the 420 pixels available, let's say we need to find the probability P(hash | face). To find this :-

P(hash | face) = (Total number of face images with # character in that pixel + pseudo*0.5)/
(Total number of face images + pseudo)

In this way, we have eliminated the chance of division by zero as well as a zero numerator problem. Thus, none of the intermediate probabilities are 0, which can result in the entire posterior probability to be zero. The value of 1.9 was arrived at after trial and error of different values between 1 and 5.

Results:
We tested on a separate testing set which had 150 entries. The final confusion matrix is as follows:-



	True Positive	True Negative
Predicted Positive	68	9
Predicted Negative	5	68

Based on the above confusion matrix, following are the accuracy, precision and recall :-

Accuracy	**:**	**90.6667 %**
Precision	:	88.3117 %
Recall	:	93.1507 %