Project 2 — Face and Digit Classification

Fernando Gonzalez, Pranav Prakash, and Wanyun Liu

{*fdg17, pp618, wl432*}*@scarletmail.rutgers.edu*

August 9, 2020

1 Classification Algorithms

1.1 Naive Bayes

The Naive Bayes algorithm classifies images by keeping track of two sets of data. First are the *prior probabilities*, which are determined by:

$$\text{Prior}(y) = \Pr(Y = y) = \frac{\text{number of images with label = y}}{\text{total number of images}} \tag{1}$$

Our goal in using the Naive Bayes classifier is to compute the probability that a given image has a certain label, given that a set of features is observed. To compute these conditional probabilities, we introduce Bayes' Rule:

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)} \tag{2}$$

Here, A should refer to a *class*. The image classes are the set of all labels that can be given to an image. When classifying digits, these are the actual digits $\{0, 1, \ldots, 9\}$. For faces, the classes are "*not-face*" and "*face*", where "*not-face*" is represented by 0 or False, and "*face*" is represented by 1 or True. We let Y, y refer to classes, and X, x refer to features, where capitals are random variables. N is the total number of features in an image. From (1), we have:

$$\Pr(Y = y \mid \bigcap_i^N X_i = x_i) = \frac{\Pr(\bigcap_i^N X_i = x_i \mid Y = y)\Pr(Y = y)}{\Pr(\bigcap_i^N X_i = x_i)} \tag{3}$$

Observing from (1) that $\Pr(Y = y)$ is the prior probability $\text{Prior}(y)$, and assuming the feature probabilities are conditionally independent, we have:

$$\Pr(Y = y \mid \bigcap_i^N X_i = x_i) = \frac{\Pi_i^N \Pr(X_i = x_i \mid Y = y) \cdot \text{Prior}(y)}{\Pr(\bigcap_i^N X_i = x_i)} \tag{4}$$

By looking for the class which maximizes this value, we can simplify this further to get our final equation:

$$\begin{aligned}
\operatorname*{argmax}_y(\Pr(Y = y \mid \bigcap_i^N X_i = x_i)) &= \operatorname*{argmax}_y(\Pi_i^N \Pr(X_i = x_i \mid Y = y) \cdot \text{Prior}(y)) &(5)\\
&= \operatorname*{argmax}_y \log(\text{Prior}(y) \cdot \Pi_i^N \Pr(X_i = x_i \mid Y = y)) &(6)\\
&= \operatorname*{argmax}_y(\log \text{Prior}(y) + \log \sum_i^N \text{Cond}(i, x_i, y)) &(7)
\end{aligned}$$

As stated previously, $\text{Prior}(y)$ is computed by finding the proportion of images with label y compared to the total number of images. $\text{Cond}(i, x_i, y)$, the probability that the feature at index i has value x_i, given the label y, is given by:

$$\text{Cond}(i, x_i, y) = \frac{\text{number of times feature } i = x_i \text{ when label} = y}{\text{number of images where label} = y} \tag{8}$$

The Naive Bayes classifier also has a smoothing parameter k, so that $\text{Cond}(i, x_i, y) \neq 0$, and to avoid direct usage of raw estimates. Our final equation for Cond, which we use to compute the *conditional probabilities*, is:

$$\text{Cond}(i, x_i, y, k) = \frac{k + \text{number of times feature } i = x_i \text{when label} = y}{(\text{number of classes} \cdot k) + \text{number of images where label} = y} \tag{9}$$

The training period of the Naive Bayes classifier consists of computing $\text{Prior}(y)$ and $\text{Cond}(i, x_i, y, k)$ for all classes y, all pixel indices i, and all possible feature values x_i. This state is then used to classify images. The result of classification is the output of (7).

1.2 Perceptron

2 Implementation

2.1 Features

Pixels were extracted to ternary features: 0 for an empty pixel, 1 for a "grey" pixel (represented by '+'), and 2 for a "black" pixel (represented by "#"). These were directly used in the case of the Perceptron algorithm, but for the Naive Bayes algorithm, the features were mapped to indicator triples $[f(X_i), g(X_i), h(X_i)]$, where $f(X_i) = 1$ if $X_i = 0$, $g(X_i) = 1$ if $X_i = 1$, $h(x_i) = 1$ if $X_i = 2$, and zero otherwise.

2.2 Results

2.2.1 Digit Classification

2.2.2 Face Classification

2.3 Obstacles