----------------------- Page 1-----------------------

Machine Learning

Minor Project Report

Ayush Choudhary(2K11/CO/032),
Aviral Takkar(2K11/CO/031),
Amanjeet Singh Bhatia(2K11/CO/020,
Ayush Ravi Rai(2K11/CO/033)

----------------------- Page 2-----------------------

Introduction

Machine learning is a branch of artificial intelligence, concerned with the construction and study of systems that can learn from data, also known as 'experience'. For example, a machine learning system could be trained on email messages to distinguish between spam and non-spam messages. After ‘learning’, it can be used to classify new email messages into spam and non-spam folders.

The core of machine learning deals with representation and generalization. Representation of data instances and functions evaluated on these instances are part of all machine learning systems. Generalization is the property that the system will perform well on unseen data instances; the conditions under which this can be guaranteed are a key object of study in the subfield of computational learning theory.

There is a wide variety of machine learning tasks and successful applications. Optical character recognition, in which printed characters are recognized automatically based on previous examples, is a classic example of machine learning.

Objectives and Scope

Machine learning algorithms can be organized into a taxonomy based on the desired outcome of the algorithm or the type of input available during training the machine. We have implemented the following class of algorithms.

Supervised learning algorithms are trained on labelled examples i.e. for the given input, the desired output is known. The supervised learning algorithm attempts to generalise a function or mapping from inputs to outputs which can then be used to speculatively generate an output for previously unseen inputs.

For our project, we have studied various machine learning algorithms, such as Classification and Regression, and applied them to build a basic machine learning application, an adblocker for websites.

Ads, which are static images have been considered.We obtained a dataset containing parameters which can be used to classify different images/pictures as ads and nonads from the Internet[1] and used it to train our machine learning algorithm to recognise and classify images other than those present in the dataset as ads or nonads. Next we set up a proxy server (implemented in python on Ubuntu/MacOS). All webpages being downloaded on the system are first screened by this proxy, and all images are extracted and passed to our machine learning program, which then classifies these images. Once classified, the proxy only forwards those images to the browser which have been recognized as non ads.

----------------------- Page 3-----------------------

MAIN CONTENT

We have followed some basic steps in creating our adblocker
There are 2 main parts to it

- A Machine Learning part
- Running a Proxy server to apply machine learning

PART 1: MACHINE LEARNING

The Machine Learning Algorithm is applied on a previously obtained data set containing various parameters (the input) of images which are used to train the algorithm to recognize images as ads and nonads (the output).

Classification : A classification problem is one in which result of the problem is a simple yes/no. This yes/no classification is also called binary binomial or binary classification.

We have applied a machine learning technique called Logistic Regression which is used in problems of classification as described above. Since an adblocker is essentially giving an output of yes (its an ad) or no (its not an ad), it is a Classification Problem.

Algorithm for logistic regression

The three main components of the algorithm are

- Hypothesis
- Cost function
- Gradient descent

Hypothesis

The Hypothesis is a equation involving certain variables and constants, which represent the parameters being used for classification, and based on these a “hypothesis” is found.

H(hypothesis) = a + a1*x1 + a2*x2 + a3*x3 + ...

H is the result found upon evaluation.
x1,x2,x3 . . . . are the variables and in our case they are the features which take
different values for different images.

Some features that we have used are
height/width of the image
aspect ratio of image
whether the image contains “ad” in its definition or not

----------------------- Page 4-----------------------

a,a1,a2,a2 . . . . are constants which are initialised to 0 and are determined using gradient descent to “best fit” the data set.

To implement this formulation, we use matrix representation.

Two matrices of the hypothesis equation are

- X: a variable matrix of dimension MxN (the features matrix)
- THETA: a constants matrix of dimension Nx1(the parameters matrix)

M = number of images used to train the algorithm
N = number of features used to describe an image

now above matrix is only for single image , for multiple image that above matrix will have
columns equal to the number of images in data set.

The Hypothesis Equation can now be formulated as

H(hypothesis) = X.THETA

The Hypothesis H is a Mx1 matrix. These M values represent the prediction of the algorithm about each of the M images in the training data set in each iteration.

In Logistic Regression, we expect the output to be binary, ie either a ‘1’ or a ‘0’. Hence, we modify the above hypothesis to obtain a binary result by taking a “sigmoid” of the above hypothesis. A sigmoid function is defined as

$$G(z) = 1/(1+e^{-z}) \text{ where } 0<G(z)<1$$

Hence our Hypothesis function can now be stated as

Y(THETA) = G(H)

----------------------- Page 5-----------------------

Cost Function
cost function can be defined as the summation of the difference between the actual
value (ad or nonad) of the images in the training data set and value predicted, as obtained in the hypothesis.

Mathematically, the formulation is

$$J(THETA) = (1/m)*(Y - y)^2$$

m = number images in our data set
Y = the hypothesis (ie the predicted values)
y = actual value

Gradient Descent

To obtain the “best fit” parameters, we apply a technique called gradient descent to “minimize” the cost function. As the cost function represents the deviation of the prediction from the actual result, we aim to minimize it.

Minimization can be achieved by setting up some initial value of the parameter THETA, say 0, and constantly updating these values till we obtain convergence in the cost function J(THETA), ie the rate of change of J(THETA) wrt THETA is at a desirable minimum.

In each iteration of gradient descent, we obtain a new set of parameters such that the cost function is approaching its global minimum.

The Gradient Descent Algorithm is as follows

```
Repeat Until Convergence{

      $THETA_j = THETA_j - alpha * \partial/\partial(THETA_j)\{J(THETA)\}$ ; for all j
simultaneously, where 0 <= j <= m

}
```

The parameter alpha, is called the **learning rate**, controls the rate of convergence of the cost function. A value for alpha has to be carefully chosen.

After 400 to 500 iterations, a desirable amount of convergence is observed, and the parameter THETA is thus obtained.

The machine learning algorithm has been implemented in **Octave**, an open source software for mathematical purposes. We chose this software as matrix implementation is easy. After running the program on the obtained dataset, we next exported the parameters THETA to a file, THETA.txt.

PART 2: PROXY SERVER