Project 3: Classification

Project 3

Pratik Appaso Vagyani

Graduate Student

Department of Computer Science & Engineering

SUNY Buffalo

pratikap@buffalo.edu

Overview

Goal of this project is to implement machine learning methods for the task of classification. We will first implement an ensemble of four classifiers for a given task. Then the results of the individual classifiers are combined to make a final decision. The classification task will be that of recognizing a 2828 grayscale handwritten digit image and identify it as a digit among 0, 1, 2, ... , 9. We are required to train the following four classifiers using MNIST digit images.

We will implement machine learning algorithm as below:

1. Multiclass Logistic regression
2. multilayer perceptron neural network
3. Random Forest
4. SVM

Machine Learning Methods

Multiclass Logistic Regression

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes

In Multiclass logistic regression, Instead of y=0,1 we will expand our definition so that y=0,1...n. Basically we re-run binary classification multiple times, once for each class. We will take weights of size number of feature * number of classes (in our case it's 784*10).

I used gradient descent to construct weights throughout training and softmax as activation function.

Steps

1. Divide the problem into n+1 binary classification problems (+1 because the index starts at 0?).
2. For each class...
3. Predict the probability the observations are in that single class.
4. prediction = <math>max(probability of the classes)

Gradient descent

It is an algorithm that minimizes functions. If a function consist of multiple parameters, gradient descent starts with an initial set of parameter values and iteratively moves toward a set of parameter values that minimize the function. This iterative minimization is achieved using calculus, taking steps in the negative direction of the function gradient.

Types of Gradient Descent Algorithms

1. Stochastic Gradient Descent
2. Batch Gradient Descent

Stochastic Gradient Descent

One training sample at each iteration is used rather than using whole dataset to sum all for every steps. SGD is commonly used for larger dataset trainings and computationally faster and can be trained in parallel. Training examples are needed to randomly shuffled before calculating it

In Stochastic Gradient descent, weights are fixed and error is reduced in every iteration.

Batch Gradient Descent

In the batch gradient descent, to calculate the gradient of the cost function, we need to sum all training examples for each steps. If there are 1 millions samples then the gradient descent algorithm sums 1 millions samples for every epoch. To move a single step, we have to calculate each with 1 million iteration. It is not good fit for large datasets.

Softmax Activation:

The sigmoid function can only handle two classes, which is not what we expect. The softmax function squashes the outputs of each unit to be between 0 and 1, just like a sigmoid function. But it also divides each output such that the total sum of the outputs is equal to 1.

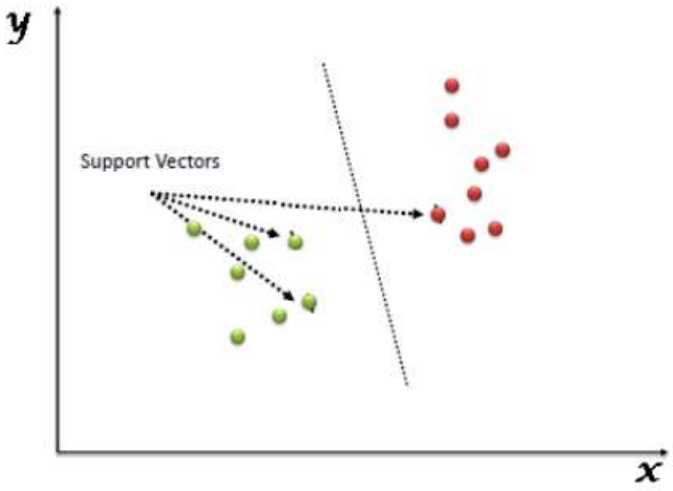
The output of the softmax function is equivalent to a categorical probability distribution, it tells you the probability that any of the classes are true.

Mathematically the softmax function is shown below, where z is a vector of the inputs to the output layer.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

SVM

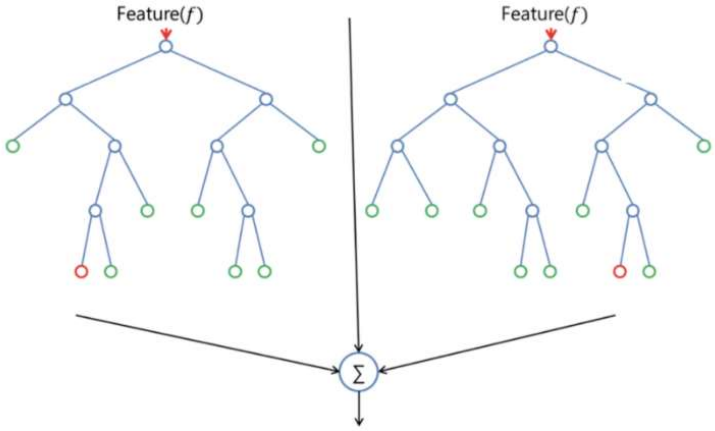
Support Vector Machine (SVM) is a supervised machine learning algorithm. It can be used for classification as well as regression problems. However, It is mostly used in classification problems. In SVM, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.



Random-Forest

Random Forest is a supervised learning algorithm. As it's name, it creates a forest and makes it somehow random. The forest it builds, is an ensemble of Decision Trees, most of the time trained with the bagging method. The general idea of the bagging method is that a combination of learning models increases the overall result

Advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. Below you can see how a random forest would look like with two trees:

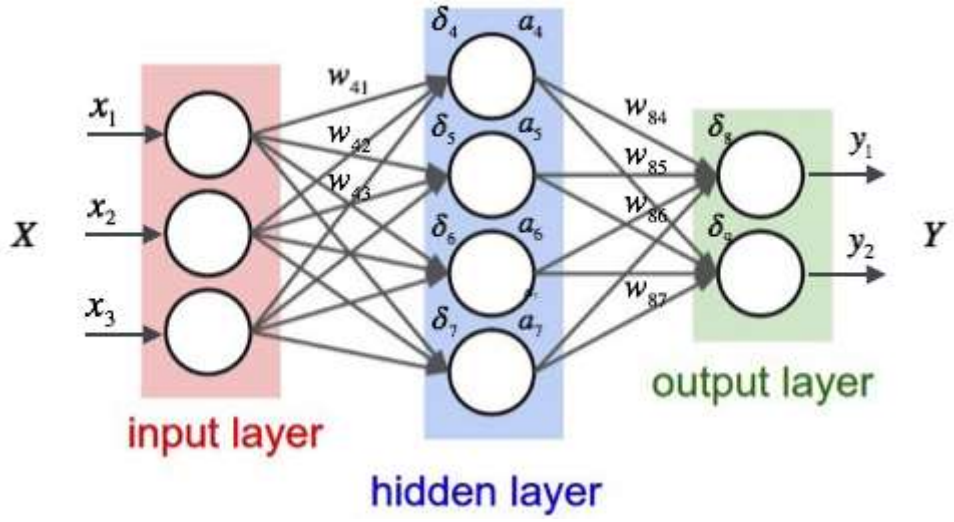


Neural Network

Neural networks are a class of machine learning algorithms used to model complex patterns in datasets using multiple hidden layers and non-linear activation functions. A neural network takes an input, passes it through multiple layers of hidden neurons (mini-functions with unique coefficients that must be learned), and outputs a prediction representing the combined input of all the neurons. Neural networks are trained iteratively using optimization techniques like gradient descent. After each cycle of training, an error metric is calculated based on the difference between prediction and target. The derivatives of this error metric are calculated and propagated back through the network using a technique called backpropagation. Each neuron's coefficients (weights) are then adjusted relative to how much they contributed to the total error. This process is repeated iteratively until the network error drops below an acceptable threshold.

Input Layer

Holds the data your model will train on. Each neuron in the input layer represents a unique attribute in your dataset.



Hidden Layer

It is between the input and output layers. We can have multiple hidden layer as well. They apply an activation function before passing on the results to next layer.

Output Layer

It receives input from last hidden layer and returns an output prediction.

Neuron

A neuron takes a group of weighted inputs, applies an activation function, and returns an output.

Activation function

They are inside neural network and modify input they receive and send further as output. It enables neural network to build complex relation between feature.

I used **relu** activation function in my neural network model.

Ensemble

I combined prediction result of each algorithm and implemented majority voting.

Observation:

Multiclass Logistic Regression:

MNIST accuracy: 84.2

Confusion matrix:

[973 0 0 0 0 1 3 1 2 0]

[44 1076 2 1 0 0 4 0 8 0]

[126 4 831 9 4 0 12 11 31 4]

[108 0 14 844 0 22 3 8 8 3]

[126 1 2 0 821 0 6 1 2 23]

[229 2 1 28 2 602 7 1 15 5]

[83 2 5 1 7 9 851 0 0 0]

[106 10 15 0 5 0 0 872 0 20]

[172 5 4 11 4 14 7 8 745 4]

[149 3 3 6 25 4 0 13 1 805]

USPS accuracy: 27.16

Confusion matrix:

[1427 3 179 13 116 28 42 11 34 147]

[1407 210 17 162 47 6 7 60 79 5]

[1019 5 796 34 19 16 37 27 41 5]

[864 0 42 894 5 121 9 21 24 20]

[988 47 8 7 665 40 5 59 130 51]

[1077 9 89 81 8 625 51 28 25 7]

[1312 5 136 28 38 60 395 2 14 10]

[1337 135 93 125 14 10 3 175 98 10]

[1351 15 37 89 39 216 38 14 185 16]

[994 105 83 277 57 13 4 217 190 60]

SVM

RandomForest model is build using Sklearn library. from sklearn.svm import SVC

I tried 3 kernal “linear”, “sigmoid”,”rbf”

MNIST accuracy : 92.57

Confusion matrix:

[951 0 4 3 0 11 8 1 1 1]

[0 1118 3 3 1 2 1 1 6 0]

[8 7 952 16 4 3 9 12 17 4]

[4 4 21 924 1 20 0 11 20 5]

[0 0 8 2 931 1 9 6 2 23]

[14 5 6 40 7 781 10 2 24 3]

[9 3 14 0 7 13 911 1 0 0]

[2 9 22 13 17 0 0 942 0 23]

[9 7 16 31 10 34 8 6 845 8]

[11 8 2 14 35 4 1 26 6 902]

USPS accuracy : 32.27

Confusion Matrix

[487 1 292 121 201 332 58 168 14 326]

[47 305 340 249 253 218 19 510 33 26]

[144 55 1121 147 49 330 58 54 33 8]

[59 38 235 950 20 565 5 43 46 39]

[24 27 109 87 913 203 19 443 31 144]

[85 31 325 247 28 1147 34 40 35 28]

[159 9 678 82 77 356 580 20 4 35]

[31 87 181 605 56 329 5 562 99 45]

[193 27 304 398 107 632 82 54 154 49]

[18 64 148 505 187 101 11 586 145 235]

Random-Forest

RandomForest model is build using Sklearn library. `from sklearn.ensemble import RandomForestClassifier`

MNIST accuracy: 94.87

Confusion Matrix

[967 1 1 2 1 3 2 1 2 0]

[0 1115 9 2 0 1 2 2 4 0]

[9 1 982 9 3 0 6 10 9 3]

[1 0 15 960 0 13 1 10 8 2]

[2 1 8 0 933 0 5 2 5 26]

[10 1 1 19 2 833 6 4 11 5]

[11 2 3 1 8 12 918 0 3 0]

[3 7 27 6 11 0 0 963 1 10]

[6 1 17 16 10 16 9 4 883 12]

[8 1 2 11 27 12 0 4 11 933]

USPS accuracy: 30.21

Confusion Matrix

[576 52 292 90 378 156 132 139 4 181]

[96 666 165 185 86 77 36 665 13 11]

[203 160 889 210 74 176 66 190 22 9]

[125 99 220 995 63 336 18 95 14 35]

[51 301 128 108 827 105 40 341 31 68]

[253 92 204 254 71 912 48 100 28 38]

[387 107 295 117 167 274 479 127 22 25]

[85 534 330 292 45 130 52 488 16 28]

[118 141 312 302 161 619 94 80 123 50]

[56 317 290 346 253 147 26 414 63 88]

Neural Network

I used tensorflow to build deep neural network. I tweaked model with 400 neuron in hidden layer, 2 hidden layers, learning rate 0.015 and 100 epocs.

MNIST Accuracy: 97.55

Confusion Matrix

```
[ 967    0    1    2    1    2    3    2    1    1]
[   0 1123    3    0    0    1    3    1    4    0]
[   4    3 1008    1    2    0    4    6    4    0]
[   0    0    4  987    1    5    0    7    4    2]
[   1    0    4    0  958    0    2    2    2   13]
[   4    1    0    9    1  858    7    1    6    5]
[   6    3    2    1    3    3  938    0    2    0]
[   2    9   10    2    0    0    0  999    0    6]
[   3    0    2    6    5    5    7    3  941    2]
[   3    5    0    7   11    2    1    4    0  976]]
```

USPS Accuracy: 45.31

Confusion Matrix

```
[ 515    1  116   91  191  119   41  107  154  665]
[ 100  413  334  114  207   71   43  534  113   71]
[  73    6 1501  117   32   89   67   41   65    8]
[  35    3  120 1412    5  295    6   34   79   11]
[  13   27   44   18 1040  103   11  365  255  124]
[  75    2  181  125   24 1312   30   45  185   21]
[ 143    7  397   78   49  153 1030   10   75   58]
[  75  108  241  483   38   76    8  798  137   36]
[ 172    5  171  338   91  320   68  101  677   57]
[  16   46   76  403  129   41    4  517  403  365]]
```

Ensemble

I used majority voting concept to ensemble 4 models. When there is clear majority, prediction will be answer of majority. Incase of conflict, Neural networks prediction is used as answer.

MNIST accuracy: 96.76

Confusion matrix

[969 0 1 2 0 2 4 1 1 0]

[0 1123 3 0 0 2 3 1 3 0]

[9 2 1002 1 2 0 4 9 3 0]

[2 0 6 977 0 8 0 8 9 0]

[1 0 5 0 956 0 4 0 2 14]

[8 2 0 15 1 845 8 1 8 4]

[9 3 1 1 3 8 931 0 2 0]

[3 9 16 2 2 0 0 987 0 9]

[6 0 6 9 5 8 7 4 925 4]

[8 4 0 11 16 3 0 5 1 961]

USPS accuracy: 42.30

Confusion matrix

[657 2 195 71 222 127 40 91 95 500]

[144 429 302 171 169 80 34 550 93 28]

[174 13 1413 100 34 113 50 49 46 7]

[84 6 125 1355 7 314 5 31 59 14]

[35 43 55 24 1042 119 10 353 205 114]

[154 10 191 145 18 1282 27 47 111 15]

[314 7 431 67 54 190 833 8 42 54]

[121 136 232 500 38 110 8 701 123 31]

[266 8 188 332 83 462 63 78 478 42]

[51 82 104 444 137 49 5 534 323 271]]

Inference:

Models	Accuracy MNIST	Accuracy USPS
Multiclass Logistic Regression	84.2	27.16
SVM	92.57	32.57
RandomForest	94.87	30.21
Neural Network	97.55	45.31

1. Neural network performs better than Logistic,SVM, Randomforest.
2. All models performs better on mnist data and less accurate on USPS data. Means we need to include part of USPS data in training set to predict better with USPS data.

Reference:

1. https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html
2. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/