Abstract

In statistics, logistic regression, or logit regression, or logit model is a regression model where the dependent variable is categorical. In this assignment, we will focus on the binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as fake/original. In this assignment, I will design the logistic regression algorithm to classify the bank notes as genuine or fake using the dataset provided by the UCI Machine Learning repository.

The outcome of this assignment shows that a higher learning rate generally achieves lower error rate. However, they are more prone to large fluctuations because of the use of Stochastic Gradient Descent.

Section 1 Logistic Regression Equation

$$Z = X * W$$

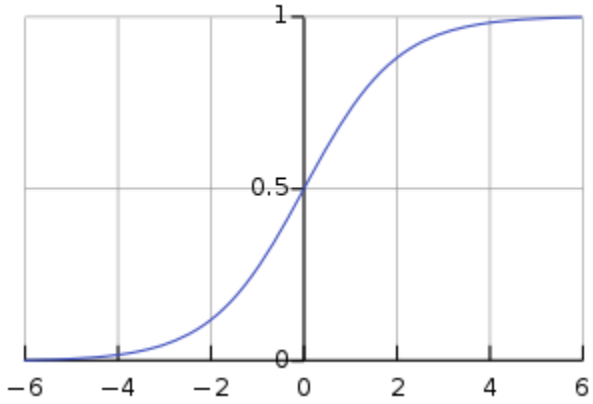
$$\hat{Y} = \frac{1}{1 + e^{-Z}}$$

$$\hat{Y} = \hat{Y} \ > 0.5$$

Let n be the number of input features. Assuming X is a (1xn+1) vector and W is a (n+1x1) vector.

X is the input feature vector with a dummy feature X0 = 1. W is the weights vector.

The predicted output $\hat{Y}$ should be a number between [0,1], this is the probability that a bank note is genuine.
However, $Z = X * W$ will not get us a number between [0,1]. Hence we use the sigmoid function,



This function enables us to have an output in [0, 1].

If the output $\hat{Y}$ is larger than 0.5, we predict that we have a genuine bank note, otherwise fake. Thus,

$$\hat{Y} = \hat{Y} \ > 0.5$$

Section 2 Error Function

Let m be the number of data examples.

$$J = -\frac{1}{m}\sum(Y * \log(\hat{Y}) + (1 - Y) * \log(1 - \hat{Y}))$$

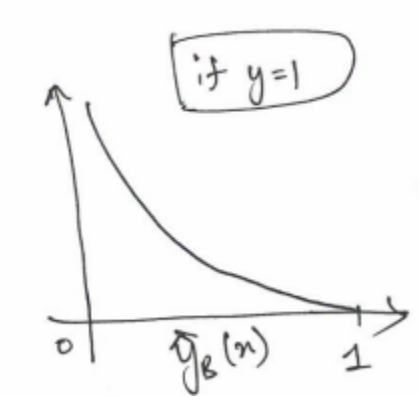
The error for each example,

$$J = -Y * \log(\hat{Y}) - (1 - Y) * \log(1 - \hat{Y})$$

When the expected output for an example is 1,

$$J = -\log(\hat{Y})$$

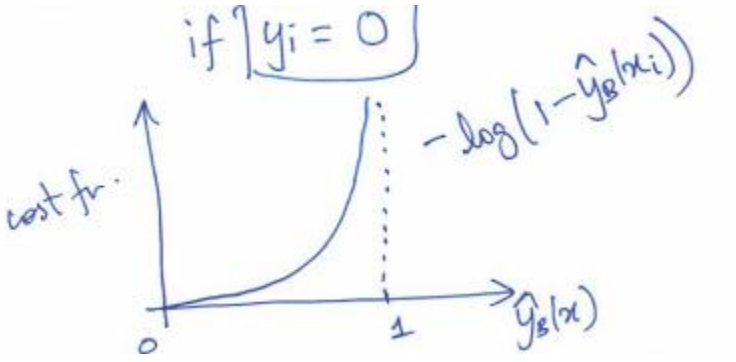
This is the graph for the negative log function, $-\log(\hat{Y})$ in [0, 1],



As show in the picture above, when the example has an output Y = 1 and the predicted output has an output of 1, the error J = 0. Otherwise, if the predicted output is 0, the error J approaches infinity.

When the expected output for an example is 0,

$$J = -\log(1 - \hat{Y})$$



As show in the picture above, when the example has an output Y = 0 and the predicted output has an output of 0, the error J = 0. Otherwise, if the predicted output is 1, the error J approaches infinity.

Section 3 SGD Algorithm

Batch Gradient Descent requires the algorithm to look at all of the training samples before updating of the weights, W. Batch Gradient Descent will be slow to converge if the number of training samples is large.

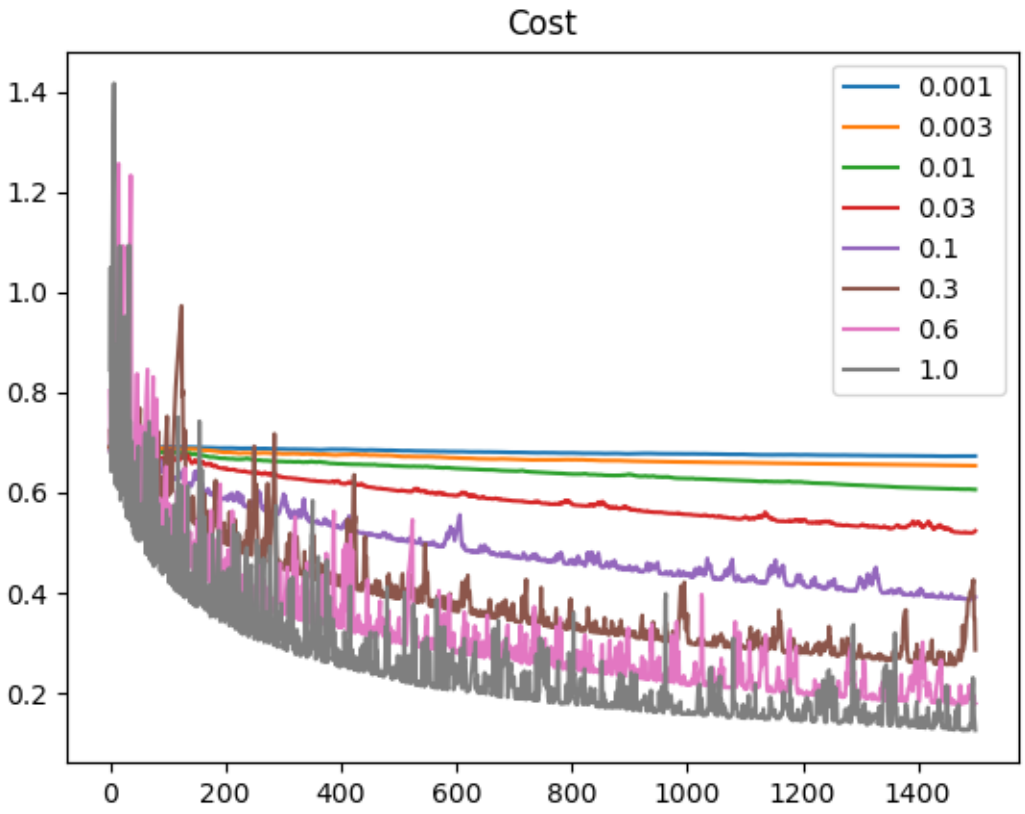
Stochastic Gradient Descent requires us to only look at one sample at a time before updating the weights. However, SGD doesn't converge. Instead it brings down the error close to a global minimum. Often times, this is good enough for practical applications.

In order for the algorithm to converge faster, we shuffle the dataset before running SGD.

To reiterate,

1. Shuffle the dataset
2. Loop over the number of iterations.
3. For each iteration, loop over the training examples.
4. Predict $\hat{Y}$, for an example.
5. Update the weights immediately.

Section 4 Learning Rate

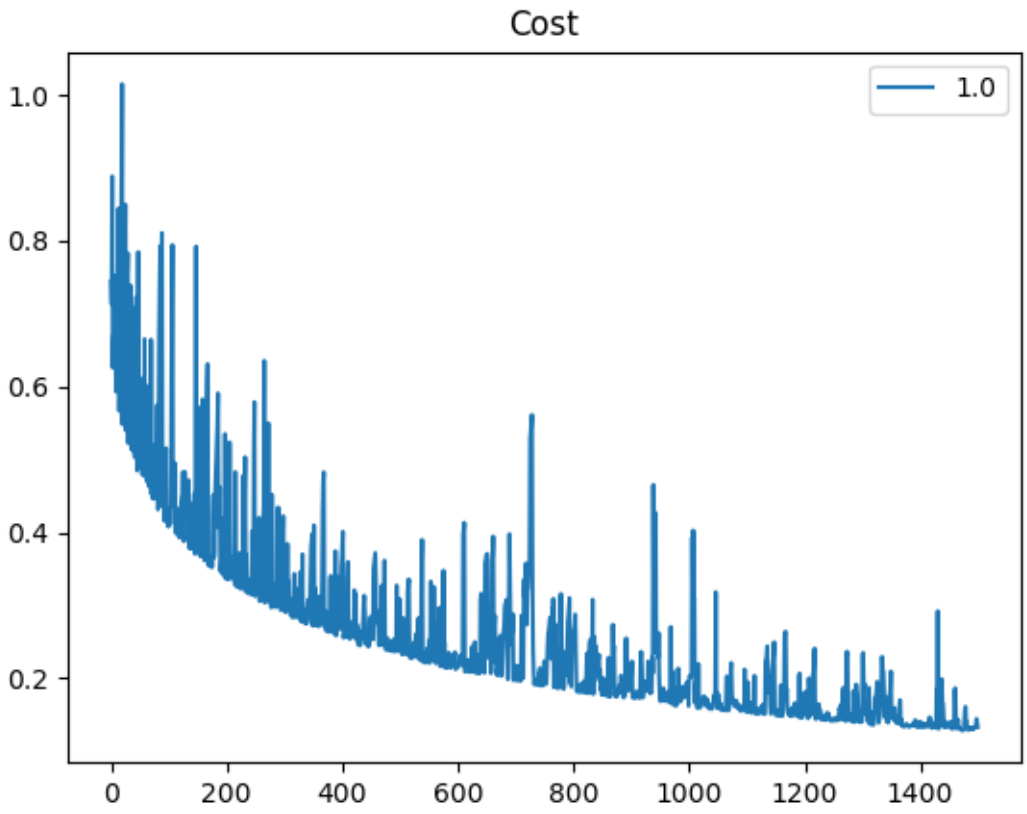


As Stochastic Gradient Descent doesn't converge, we can hope for a best estimate that will get us close enough to a global minimum. A high learning rate of 1.0 seems to clearly decrease the error faster when compared to a learning rate of 0.1 or 0.001, etc. But a higher learning rate also produces much noisier results. We can see that for a number of iterations, a learning rate of 1.0 seems to have higher error than 0.6. All in all, the error rate for 1.0 seems to be on a down trend and towards the end have the lowest error. Thus the optimal learning rate seems to be 1.0 for this experiment

Section 5 Experimental Result

	W1	W2	W3	W4	W5	Accuracy
Set 1	38.7356	-29.1487	-28.7373	-32.153	2.052645	98.542
Set 2	38.88186	-29.1933	-29.2167	-32.0123	2.01845	98.542
Set 3	38.11006	-29.5498	-27.4621	-30.3574	2.511901	98.36
Set 4	37.73228	-30.6121	-26.6836	-30.4398	3.613396	98.17
Set 5	38.604	-28.7735	-29.3958	-31.0128	0.979573	99.27

The table above is generated with an Alpha of 1.0. The accuracy of the each model generated with an alpha of 1.0 manage to consistently perform well. Accuracy > 98%.



When using Stochastic Gradient Descent, the curve will fluctuate throughout, even as it nears minimum. Compared to Batch gradient descent, which will have a smooth curve towards convergence.