Steps followed and common definitions:

→ Calculate zscores for all the emails in the spam dataset.
→ Divide emails ending with 1 as testing fold and rest all as training fold
→ Run linear and logistic regression for stochastic and batch gradient descent
→ Convergence criteria is taken as 0.001
→ Calculate the ROC with weights obtained at the point convergence
→ Calculate AUC for the obtained ROC.

Linear Regression:

→ In a linear regression model, the dependent variable is considered continuous. The general linear equation $Y=b0+\sum(biXi)+\epsilon$ where Y is a continuous dependent variable and independent variables X_i are usually continuous.

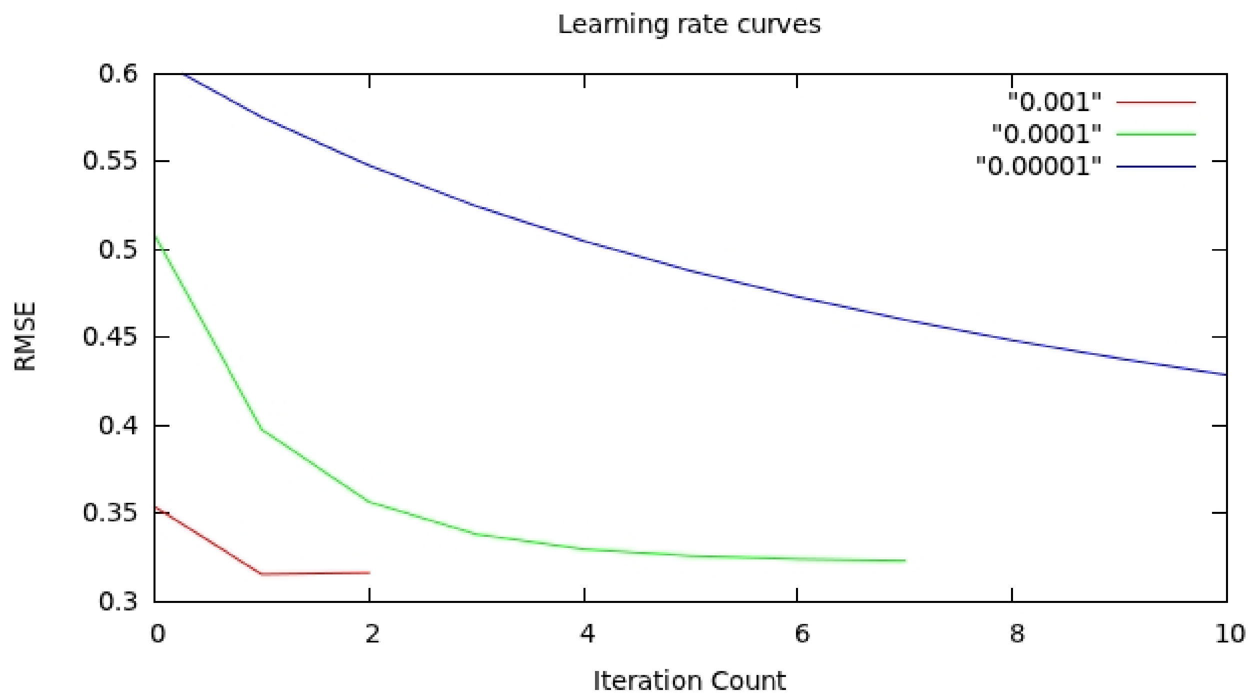
→ **Stochastic gradient descent:**

→ In this regression technique, we run through the training set, and at each data point, we update the parameters according to the gradient of the error with respect to that single training example only. Stochastic gradient descent can start making progress right away, and continue to make progress with each example it looks at. The stochastic gradient descent values for linear regression on the training data looked like below:

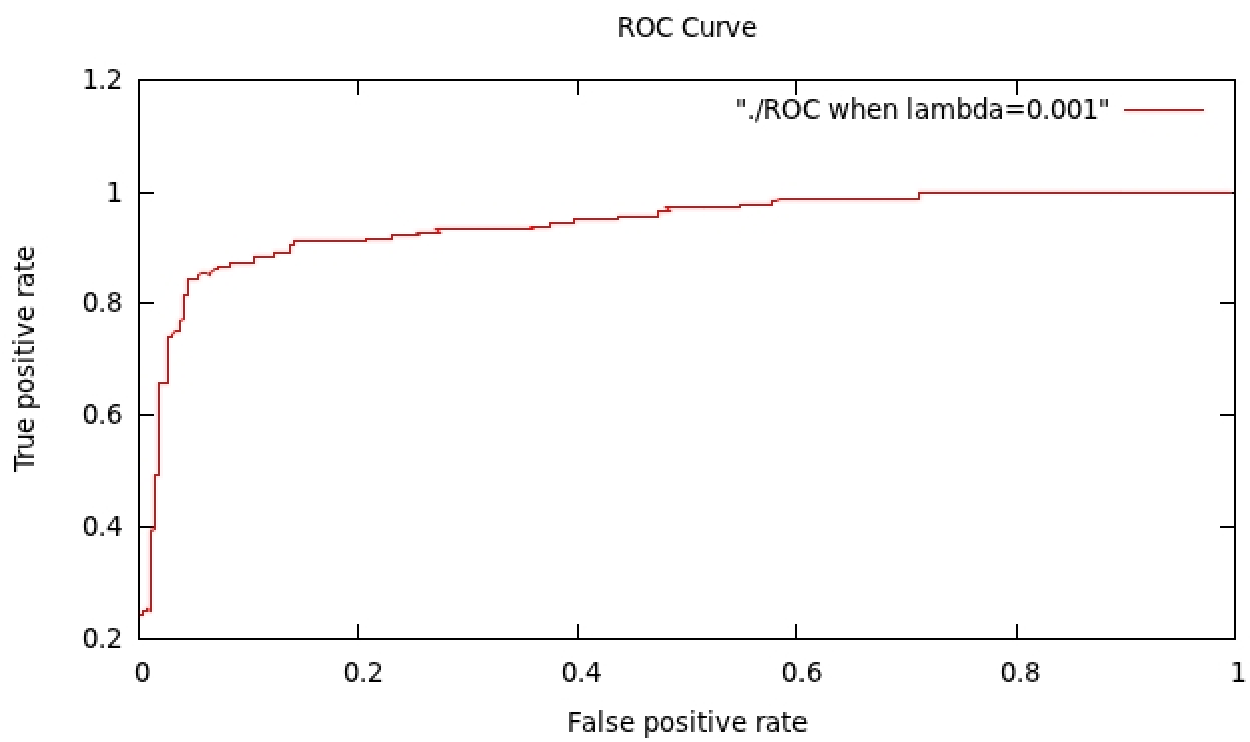
Learning rate parameter (lambda)	RMSE	Convergence Iteration
0.001	0.31627743131	3
0.0001	0.32325085143	8
0.00001	0.335121196711	39

As observed with a slower learning rate parameter, the number of iterations required to reach to convergent criterion (minimum) point is more when compared to a tad bit higher learning rate parameters.

The learning rate parameter curves for the above three values look like below:



The ROC curve for the learning rate parameter 0.001 is as below:



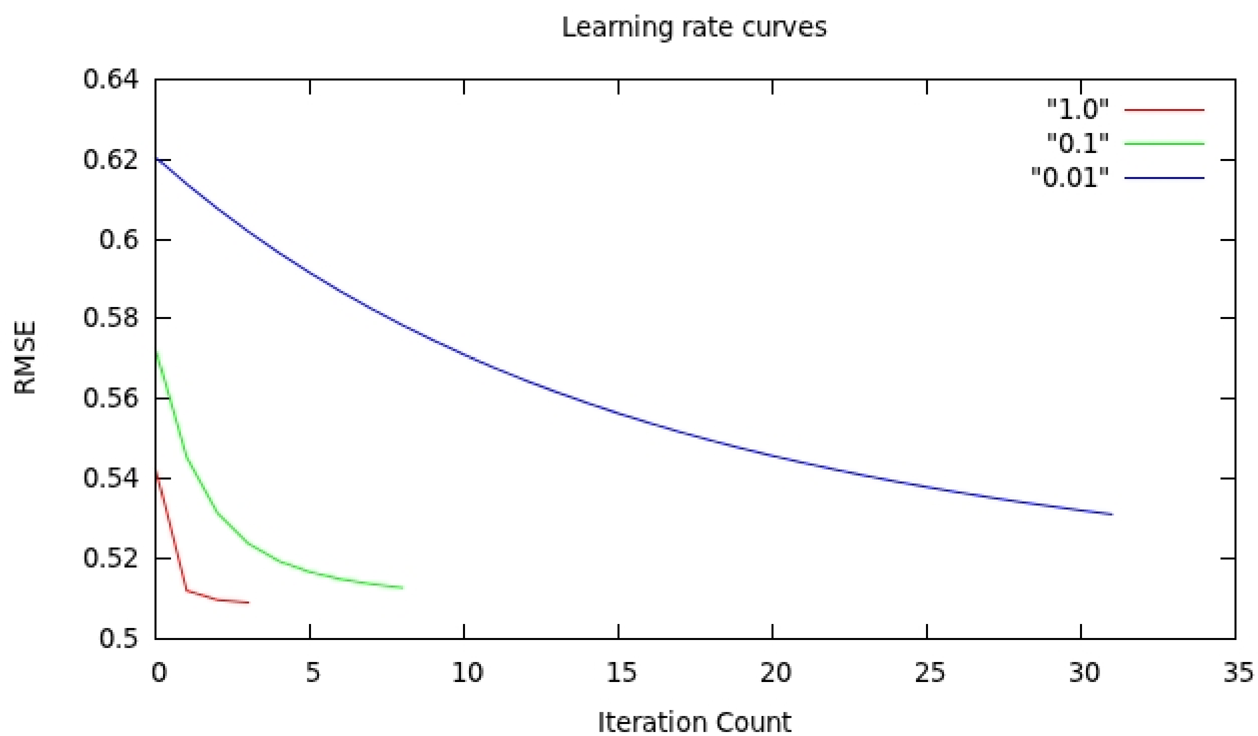
The AUC value for the above ROC is : 0.938542879623

→ **Batch gradient descent:**

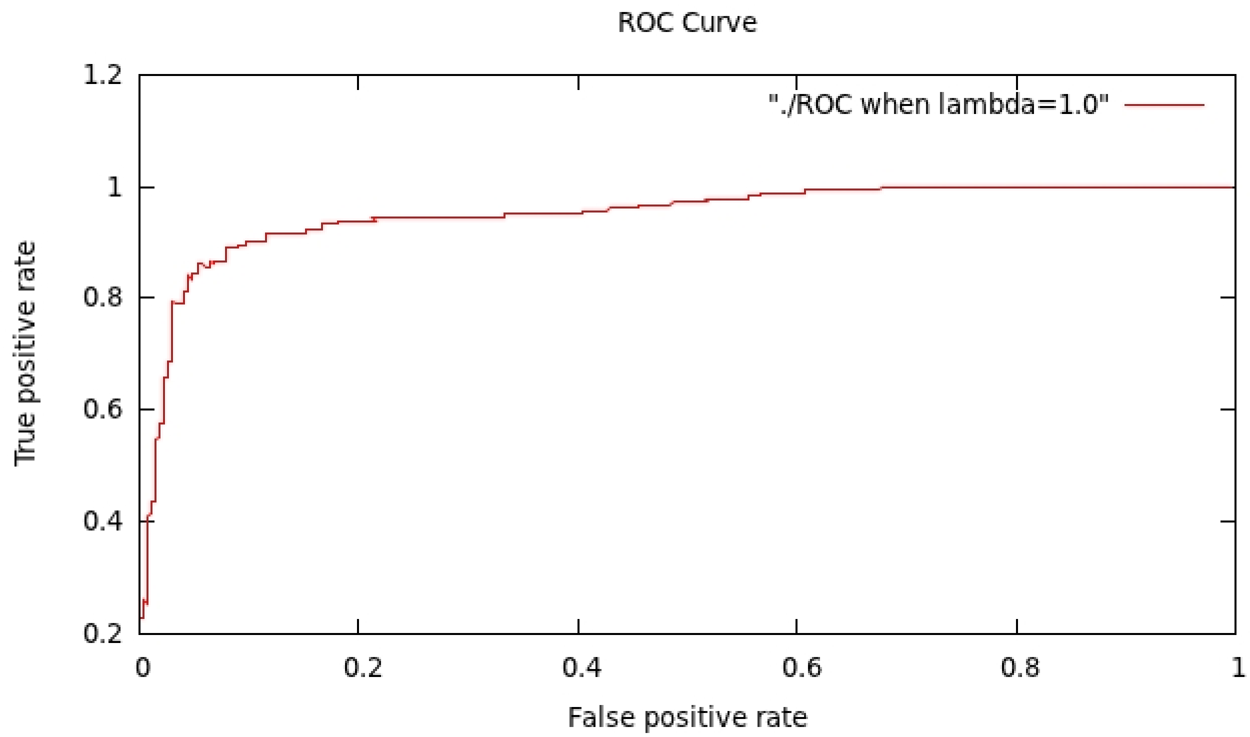
In this algorithm, we scan through the entire training set before taking a single step. This can be a costly operation if m is large. The batch gradient descent values for linear regression looked like below:

Learning rate parameter (lambda)	RMSE	Convergence Iteration
1.0	0.508932869743	4
0.1	0.512644370997	9
0.01	0.531081568263	32

The learning rate parameter curves for the above three values look like below:



The ROC curve for the learning rate parameter 1.0 is as below:



The AUC value for the above ROC is 0.946502734772

Logistic Regression:

→ This is a type of regression analysis used for predicting the outcome of a categorical variable based on other predictor variables. There are two types of logistic regression. One is binomial or binary (0 or 1) and the other is multinomial logistic regression.

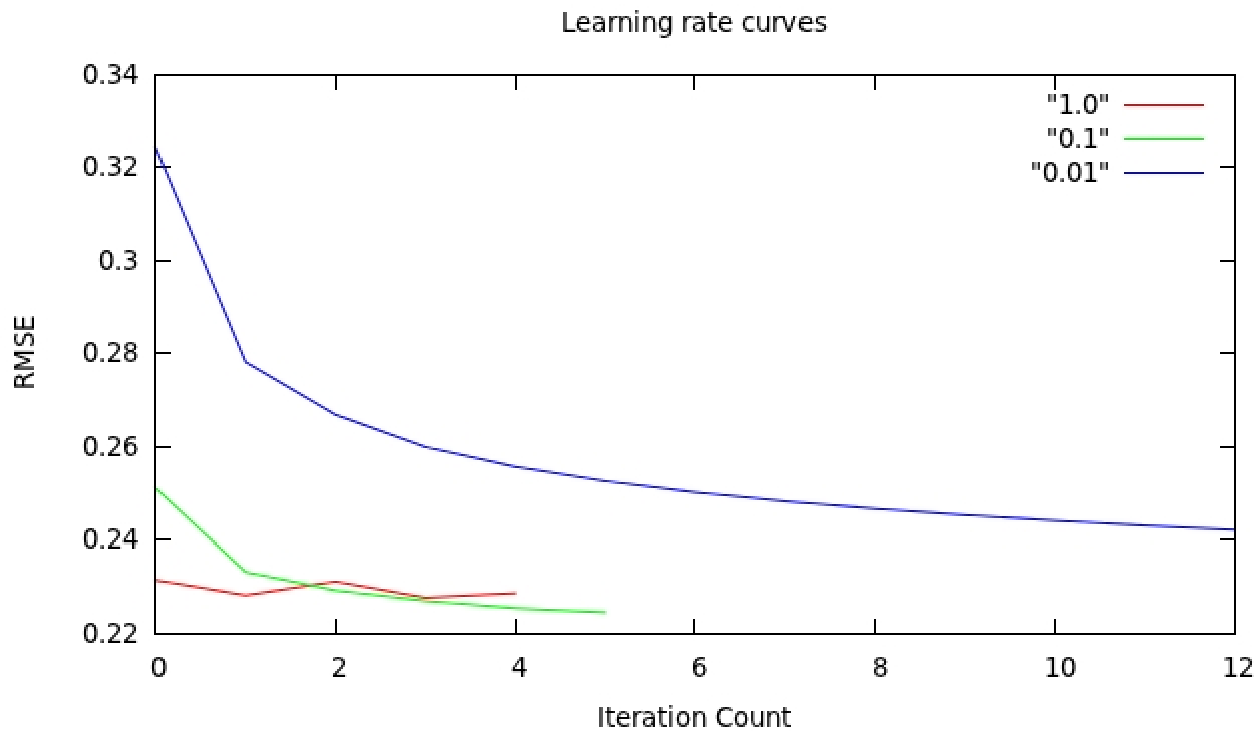
→ **Stochastic gradient descent:**

→ The stochastic gradient descent values for linear regression looked like below:

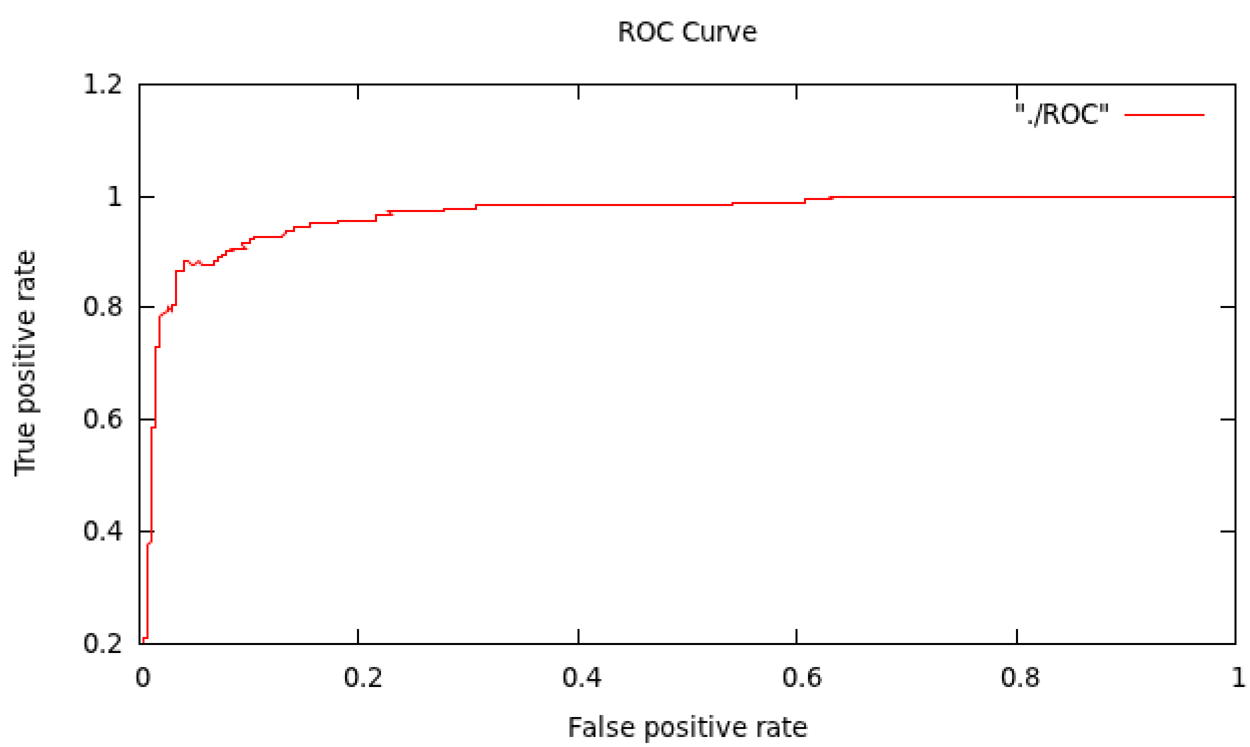
Learning rate parameter (lambda)	RMSE	Convergence Iteration
0.01	0.242206625517	12
0.1	0.22448538998	5
1.0	0.228658640091	4

→ As seen, the error rate value is lesser when learning rate parameter is at 0.1

→The learning rate parameter curves for the above three values look like below:



→The ROC curve for the learning rate parameter 0.1 is as below:



→**The AUC value for the above ROC is 0.961672381892**

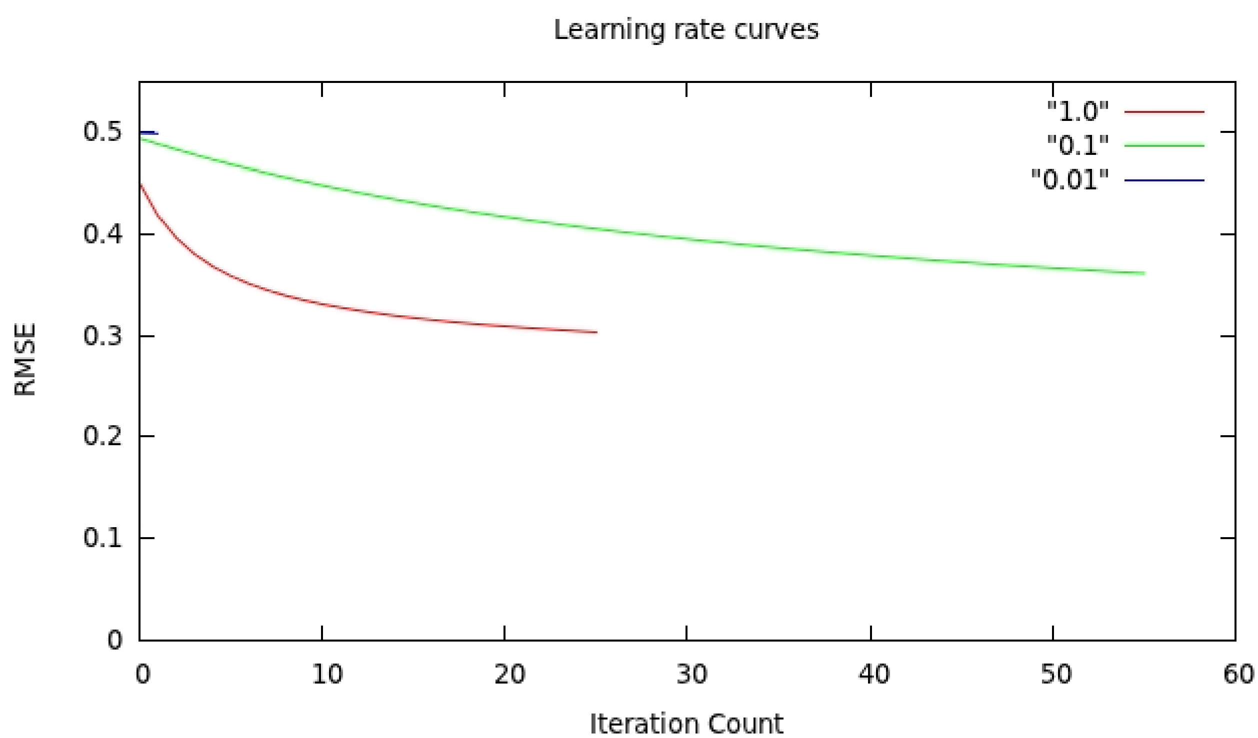
Batch gradient descent:

→ The batch gradient descent values for linear regression looked like below:

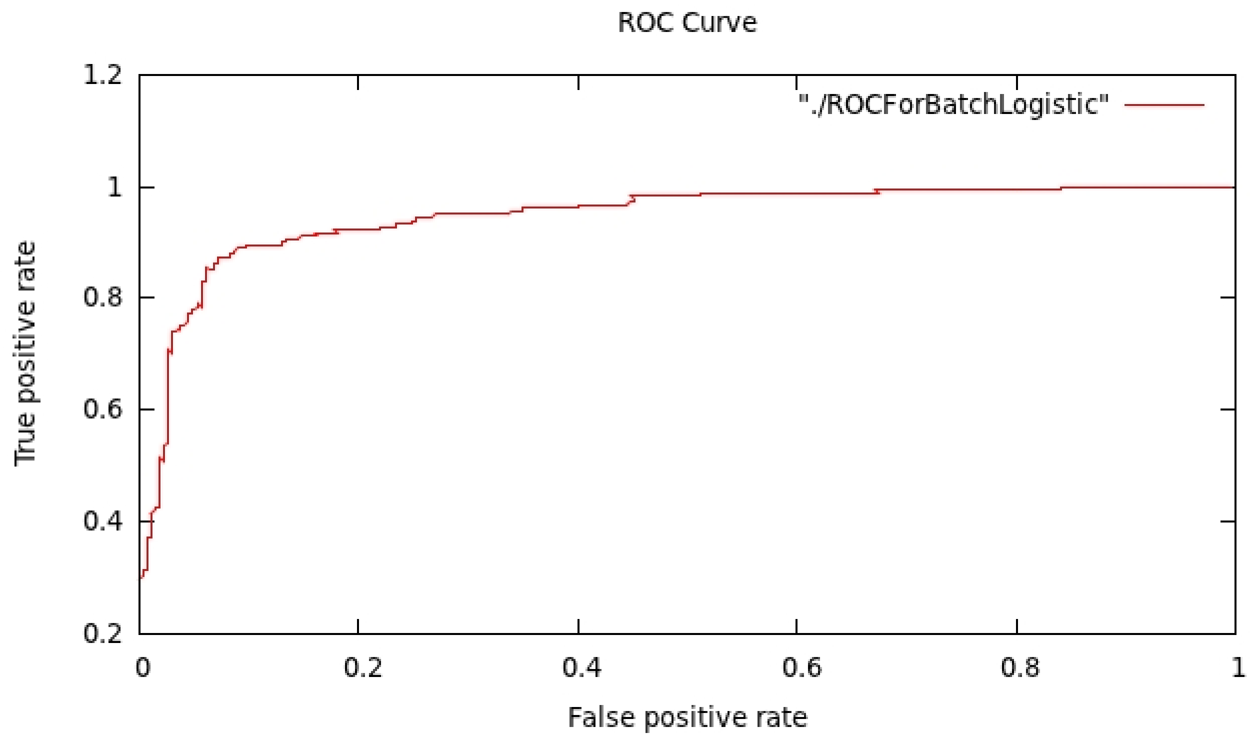
Learning rate parameter (lambda)	RMSE	Convergence Iteration
1.0	0.303422659702	25
0.1	0.361474183233	55
0.01	0.498830500921	2

→ As seen, the error rate value is lesser when learning rate parameter is at 0.1

→The learning rate parameter curves for the above three values look like below:



→The ROC curve for the learning rate parameter 1.0 is as below:



→The AUC value for the above ROC is 0.943733095344

Comparison between stochastic and batch gradient descents:

The value of how many passes needed to obtain a good predictor completely depends on the learning rate parameter. Lesser values of learning rate will lead to higher number of required iterations and might not converge close to the minimum as can be done with lesser learning rate values. Higher learning rate values will lead to lesser number of iterations required. So, the below comparisons are for learning rate that give minimum error rate values.

Linear Stochastic:

→ The convergence rate for a good RMSE value in stochastic is obtained at learning rate 0.001 and require 3 iterations. So, if the learning rate parameter is around 0.01, three iterations would suffice to obtain a good RMSE value

Linear Batch:

→ The convergence rate for a good RMSE value in batch is obtained at learning rate 1.0 and require 4 iterations. So, have to pass through the data four times for obtaining good RMSE value. If learning rate parameter is decreased, the number of required iterations might be more.

Logistic Stochastic:

→ The convergence rate for a good RMSE value in logistic stochastic at learning rate value 0.1 is 0.224 and it require a total of 5 iterations.

Logistic Batch:

→ The convergence rate for a good RMSE value in logistic batch at learning rate value 1.0 is 0.30 and requires 25 times passing through the data.

The lowest convergence RMSE value is obtained for logistic stochastic and the value is 0.224.

Model	Learning rate	Iterations required	Convergence rate (RMSE)
Linear Stochastic	0.001	3	0.31627743131
Linear Batch	1.0	4	0.508932869743
Logistic Stochastic	**0.1**	**5**	**0.22448538998**
Logistic Batch	1.0	25	0.303422659702

So, logistic stochastic gives the less error rate and hence higher performance at learning rate 0.1 with passing through data for 5 times.

Perceptron Algorithm:

→ Perceptron algorithm is an online algorithm for linear threshold function. In this homework, the threshold of 0.

w0 x 0 + w1 x1 + w2 x2 + ... + wn xn > 0.

→ A perceptron learning algorithm is created where the data point label is checked and all the features are flipped if the label is -1. And then hypothesis value is calculated and compared against a threshold (0).

→ A mistake is said to happen if the hypothesis value is less than the threshold.

→ The below are the total mistakes (cumulative) after each iteration until there are no mistakes and corresponding classifier weights and normalized weights.

Iteration: 0 total mistakes: 136
Iteration: 1 total mistakes: 204
Iteration: 2 total mistakes: 254
Iteration: 3 total mistakes: 276
Iteration: 4 total mistakes: 297
Iteration: 5 total mistakes: 331
Iteration: 6 total mistakes: 356
Iteration: 7 total mistakes: 356

Classifier weights: [-14.0, 2.5287325879008797, 5.707170513923717, 8.522314571678628, 11.32560723224928]

Normalized with threshold: [0.18062375627863428, 0.40765503670883696, 0.608736755119902, 0.8089719451606628]