Name – Vikram Kumar

Entry no. 2016CE10237

Scores Tables and confusion matrix for 3 different cases(partA)

Rows for true labels and column for predicted labels

Please don't see title of plots (mistakenly tiltes of plots remained same) please see headings

1. **learning rate – 0.01(fixed) , iteration number – 4000**
    a. **Confusion Matrix and scores class wise**

	not_recom	recommend	very_recom	priority	spec_prior	precesion	recall	F1_scores
not_recom	1997.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
recommend	0.0	0.0	0.0	1.0	0.0	0.0	nan	nan
very_recom	0.0	0.0	0.0	145.0	0.0	0.0	nan	nan
priority	0.0	0.0	0.0	1843.0	185.0	0.9088	0.8477	0.88
spec_prior	0.0	0.0	0.0	185.0	1644.0	0.8989	0.8989	0.9

    b. **F1 Scoring – mirco and macro averages scores**

	Micro Scores	Macro Scores
F1	0.914	0.555
average precall	0.914	0.562
average recall	0.549	0.914

2. **Adaptive learning rate =1 , iteration - 4000**

Confusion matrix and scoring for each class separately

	not_recom	recommend	very_recom	priority	spec_prior	precesion	recall	F1_scores
not_recom	1997.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
recommend	0.0	0.0	0.0	1.0	0.0	nan	0.0	nan
very_recom	0.0	0.0	0.0	145.0	0.0	nan	0.0	nan
priority	0.0	0.0	0.0	1841.0	187.0	0.848	0.9078	0.88
spec_prior	0.0	0.0	0.0	184.0	1645.0	0.8979	0.8994	0.9

a. F1 scoring

	Micro Scores	Macro Scores
F1	0.914	0.555
average precall	0.914	0.549
average recall	0.561	0.914

3. Backtracking , learning rate= 1.0 , alpha = 0.1, Beta = 0.7, iteration - 4000

a. Confusion matrix

	not_recom	recommend	very_recom	priority	spec_prior	precesion	recall	F1_scores
not_recom	1997.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
recommend	0.0	0.0	1.0	0.0	0.0	nan	0.0	nan
very_recom	0.0	0.0	104.0	41.0	0.0	0.8889	0.7172	0.79
priority	0.0	0.0	12.0	1825.0	191.0	0.8829	0.8999	0.89
spec_prior	0.0	0.0	0.0	201.0	1628.0	0.895	0.8901	0.89

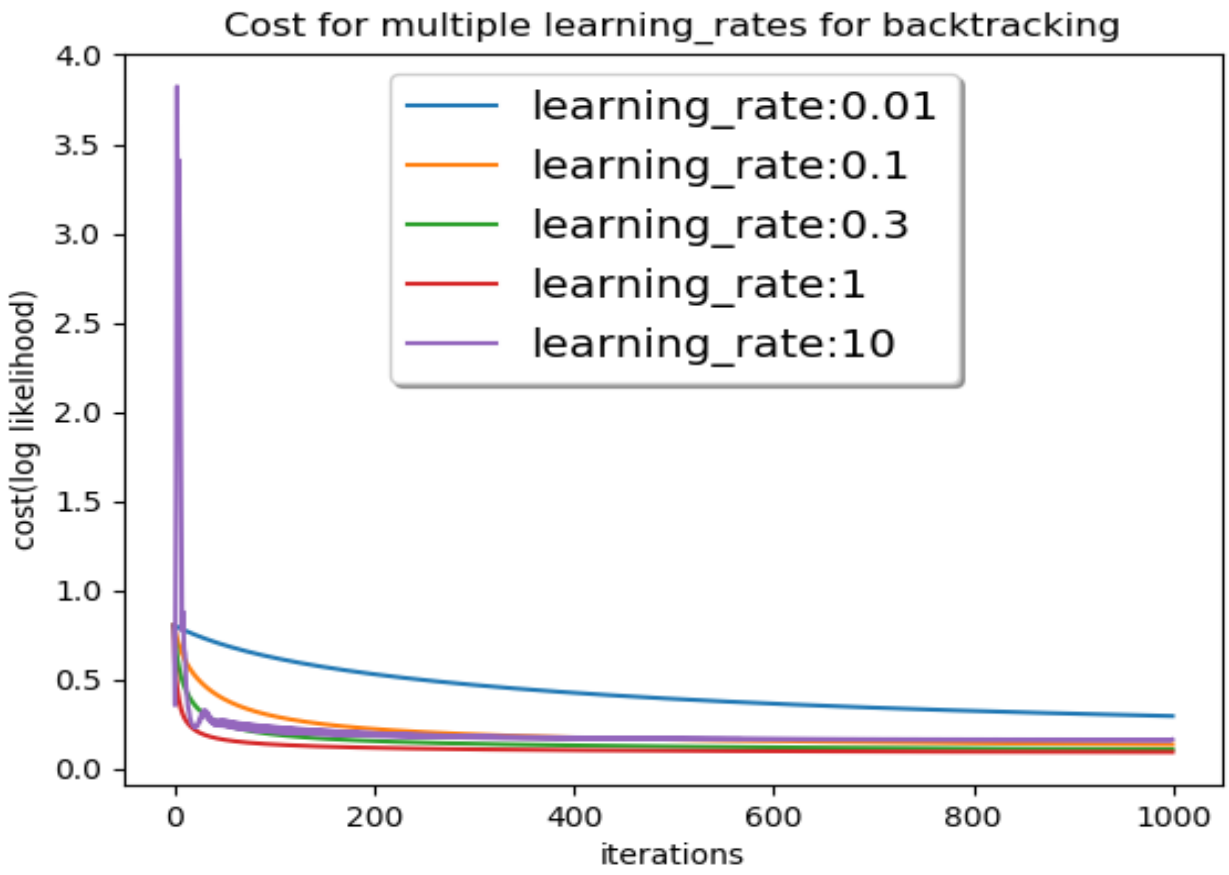
b. F1 Scoring

	Micro Scores	Macro Scores
F1	0.926	0.717
average precall	0.926	0.733
average recall	0.701	0.926

It can be seen that accuracy score remain approximately constant(there is increase of small amount(around 1%) in backracking algorithm) and also backtracking does slightly good job on imbalanced classes comparing to other too algorithms, there are too classes namely **recommend** and **very_recom** which does not predicted by other too ways of gradient descent, backtracking does good job on predicting of **very_recom** and precesion and recall values boost up for this algo, still single training example of **recommend** does not recognised by any of these gradient descent methods(which is actually rare to predict because of not enough information of this class).

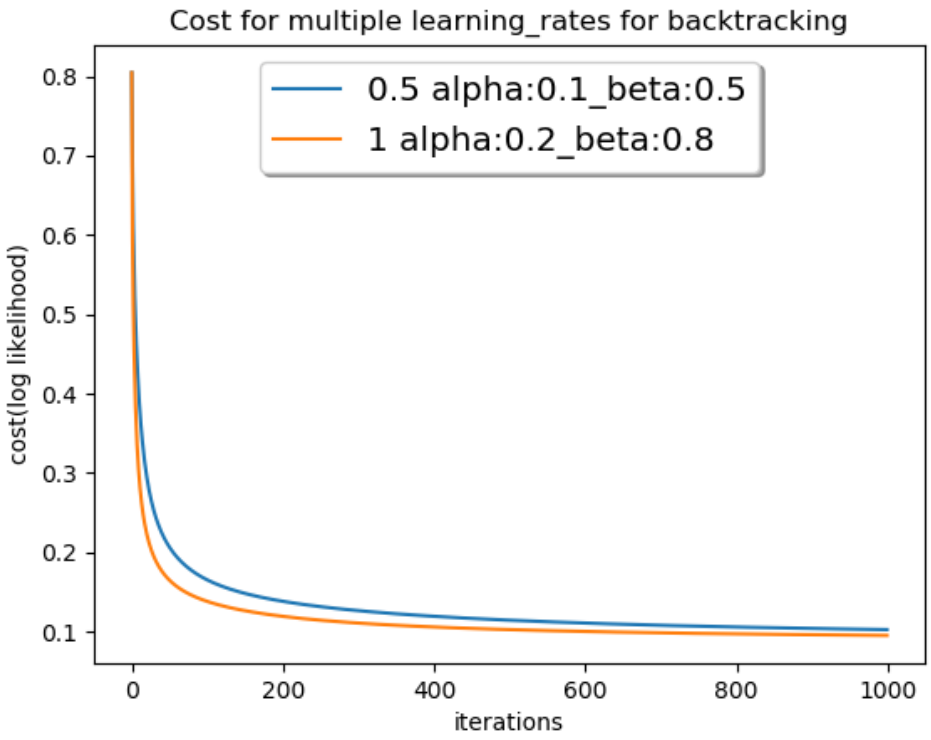
Loss Function for different Learning Rates-

Case1. Fixed learning rate

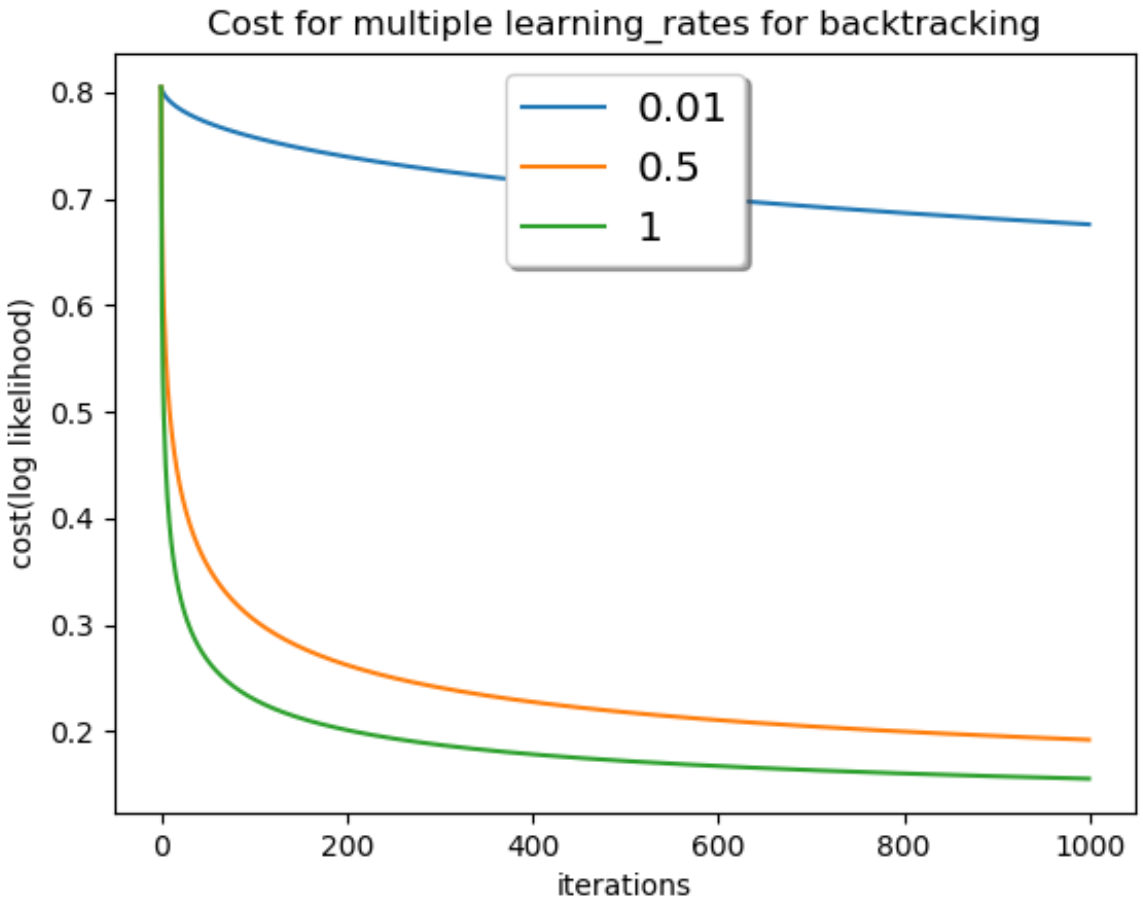


Random fluctuation can be seen due to over shooting in case of learning rate equal to 10(high) and slow convergence of learning rate of 0.01(not very low) compared to others

Case2 . Backtracking algorithm:



Case 3 . Adaptive gradient descent



Loss function for different batch size and Fixel learning rate(0.1):

Different batch sizes = [50,100,200]

Since I am dividing calculated cost in 1 epochs by number of mini batches smoothens curve