**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**
   1. **Confusion Matrix and scores class wise**

A screenshot of a cell phone

Description automatically generated

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

A close up of a logo

Description automatically generated

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

**Confusion matrix and scoring for each class separately**

A screenshot of a cell phone

Description automatically generated

* 1. **F1 scoring**

**A close up of a logo

Description automatically generated**

1. **Backtracking , learning rate= 1.0 , alpha = 0.1, Beta = 0.7, iteration - 4000**
   1. **Confusion matrix**

A screenshot of a cell phone

Description automatically generated

* 1. **F1 Scoring**

A close up of a logo

Description automatically generated

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**

A screenshot of a cell phone

Description automatically generated

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: A picture containing screenshot

Description automatically generated**

**Case 3 . Adaptive gradient descent**

**A picture containing screenshot

Description automatically generated**

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

A screenshot of a cell phone

Description automatically generated