Maximum Likelihood Estimation and Gradient Descent Optimization

1 Overview

The objective of this assignment is to apply Maximum Likelihood Estimation (MLE) for logistic regression. To achieve this, we first performed Exploratory Data Analysis (EDA) on the customer churn dataset to understand its structure and key patterns. We then implemented three variants of Gradient Descent for optimization Stochastic gradient descent (SGD), Gradient descent (GD) and Batch Gradient descent (BGD).

After implementing these approaches, we trained and tested the logistic regression model, evaluated its performance using classification metrics, and analyzed the impact of different optimization techniques.

2 Data Preprocessing

The preprocessing pipeline consists of several essential steps to ensure data quality and model effectiveness.

2.1 Exploratory Data Analysis (EDA)

- Conducted exploratory data analysis (EDA) using statistical plots like histagram, boxplot and countplot etc. and graphical representations.
- Identified patterns, trends, and distributions to guide preprocessing decisions.
- The dataset consists of separate training and testing files, eliminating the need for data splitting.

2.2 Feature Analysis

- Assessed the distribution of the dependent variable.
- Dropped unnecessary 'CostumerID' column.
- Investigated correlations between independent variables and the dependent variable.

- Checked for multicollinearity using variance inflation factor.
- Identified missing values and dropped empty row which was present in training dataset.
- Changed object type columns to integer type by introducing dummy variables.
- Scaled integer value columns using min-max scalar.

3 Model Evaluation

3.1 Cost Function Formulation

- Derived the cost function based on log-likelihood maximization.

$$\log L(\theta) = \sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \tag{1}$$

- Considered the dependent variable's dependency on independent variables.
- Final cost function formulation:

$$loss = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \tag{2}$$

  where $\hat{y}_i$ is the predicted probability for sample i and n is the number of samples.

3.2 Gradient Descent Algorithm Implementation

- Implemented the cost function using three variants of Gradient Descent:
  - **Stochastic Gradient Descent (SGD)** - Updates parameters using one training example per iteration.
  - **Full Batch Gradient Descent (BGD)** - Updates parameters using the entire dataset in each iteration.
  - **Mini-Batch Gradient Descent (MBGD)** - Uses small batches of data to balance computational efficiency and convergence speed.
- Checked for learning rate 0.1, 0.01, 0.001, 0.0001 and epochs = 1000 and plotted log-likelihood vs epochs. learning rate 0.001 and 0.0001 are too slow and 0.1 is unstable so we used 0.01 learning rate and 1000 epochs for training.
- Computed parameter updates iteratively to optimize the likelihood function.

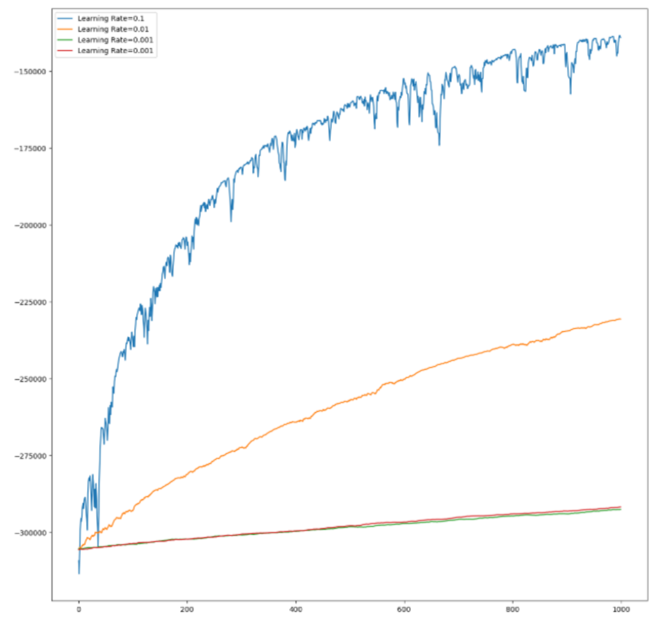

Figure 1: Log likelihood vs epochs for SGD

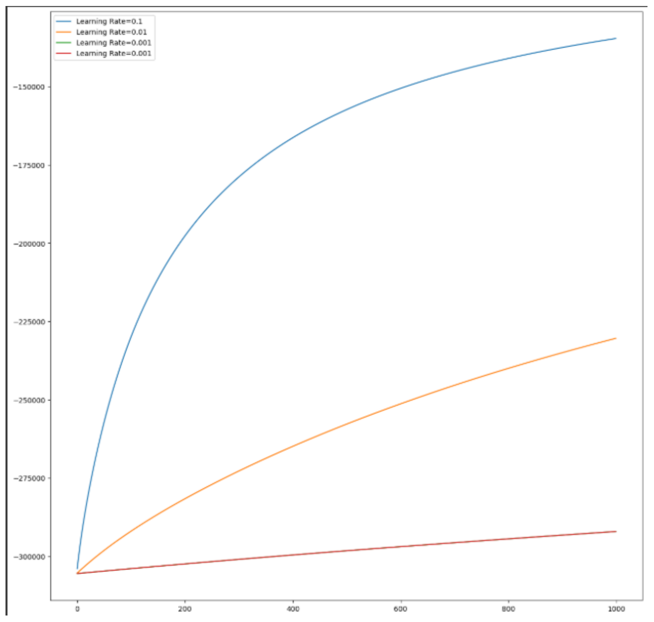

Figure 2: Log likelihood vs epochs for GD

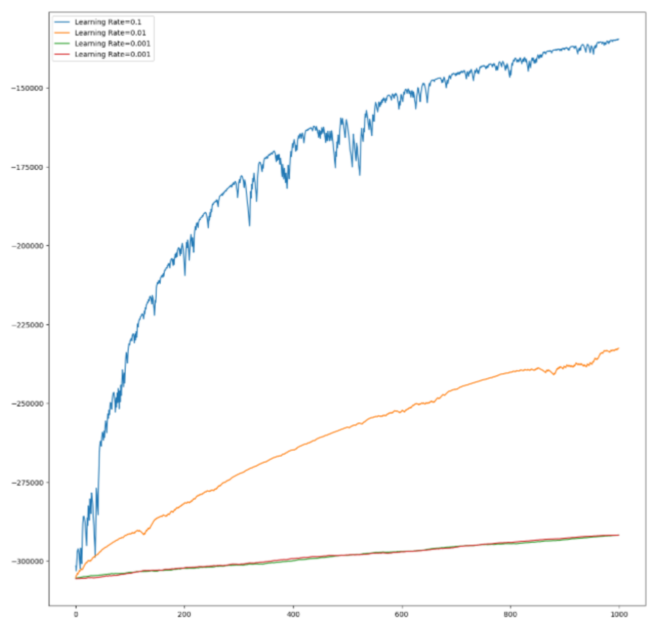

Figure 3: Log likelihood vs epochs for BGD

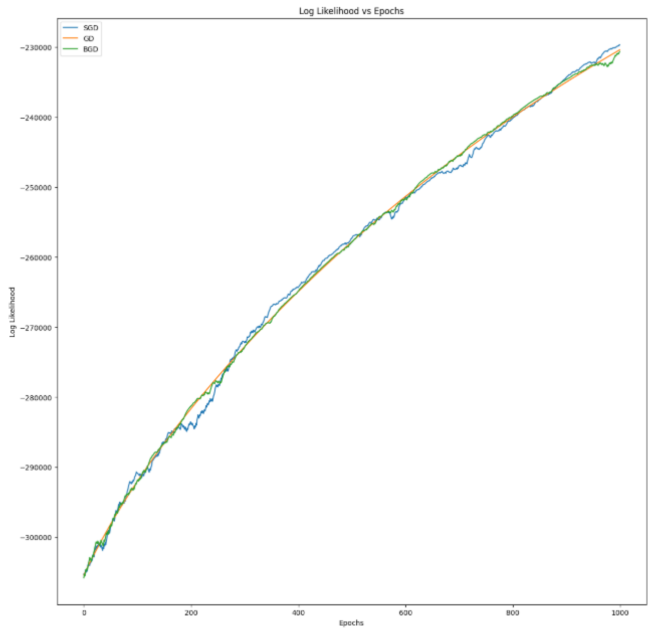

Figure 4: comparison for all three gradient descent for learning rate 0.01

3.3 Model Testing and Evaluation

- Tested the three trained models using the test dataset.
- Reported standard classification metrics:
  - Accuracy
  - Precision, Recall, and F1-score
  - Confusion Matrix
  - ROC Curve and AUC

Method	Accuracy	Precision	Recall	F1-score
SGD	0.541150	0.508017	0.992293	0.671997
GD	0.547426	0.511479	0.992884	0.675156
BGD	0.580747	0.531262	0.976388	0.688115

Table 1: Performance metrics for different Gradient Descent methods

- Visualized performance differences using:
  - Precision recall curve for different gradient descent variants.
  - ROC curves for probability-based evaluation.
  - Confusion matrices for classification assessment.

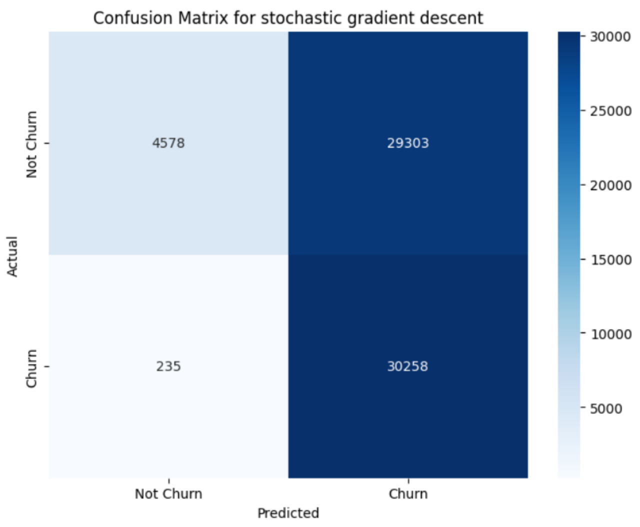

Figure 5: confusion matrix for SGD

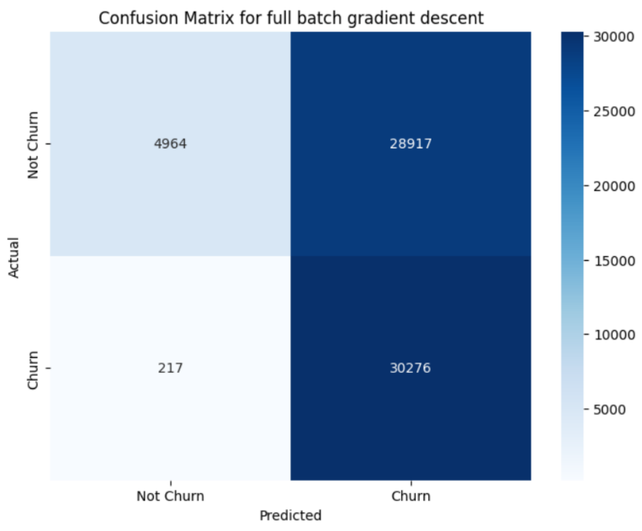

Figure 6: confusion matrix for GD

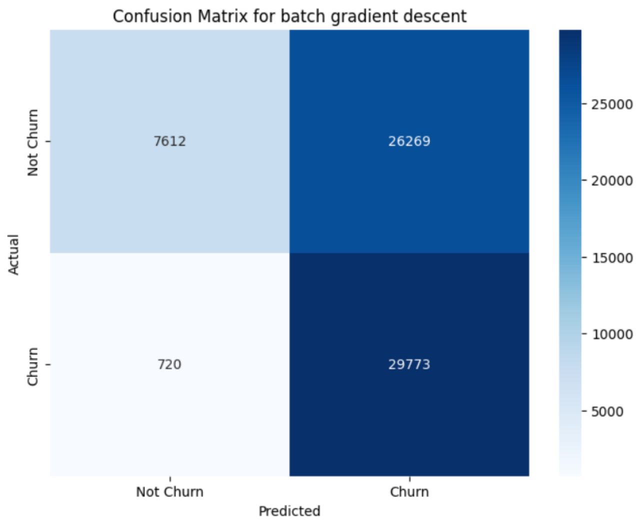

Figure 7: confusion matrix for BGD

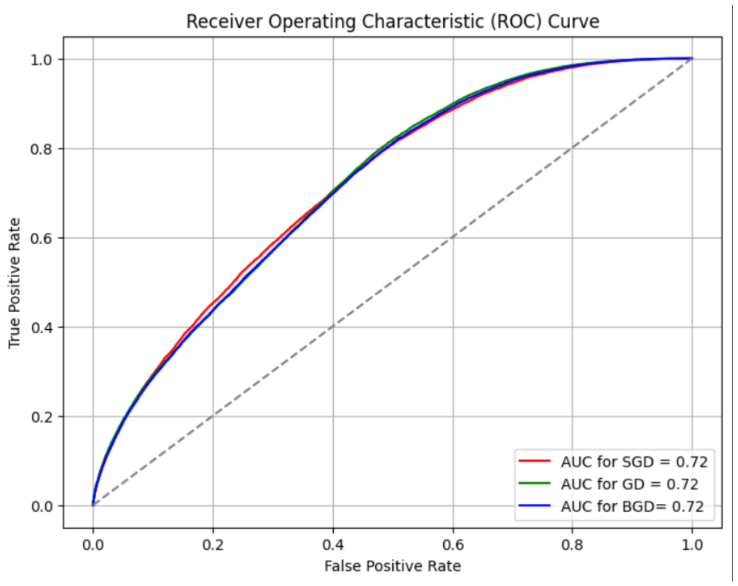

Figure 8: ROC curve

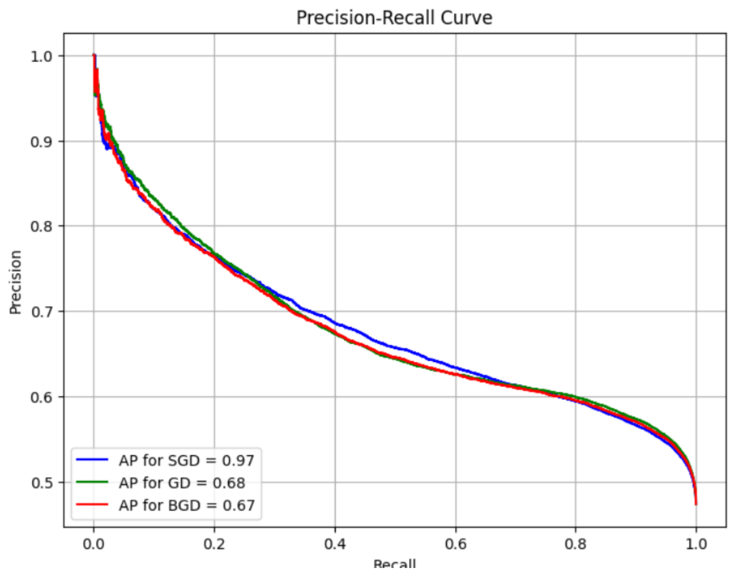

Figure 9: Precision Recall curve

4 Performance Measures

4.1 Comparison of Gradient Descent Variants

- SGD and GD has almost same accuracy and BGD a higher accuracy.
- GD takes maximum time to train, then BGD and SGD takes least time to train.
- BGD has highest precision, lowest recall and higher F1-score then SGD and GD.
- Overall performance of these three gradient descent is not that much great it looks like that it is more biased towards predicting churn.

5 Conclusion

- we implemented logistic regression using maximum likelihood estimation and three types of gradient descent (SGD, GD, BGD). BGD works best in these three gradient descent because SGD is a faster technique and GD is a stable technique and BGD is a mixture of SGD and GD. we can improve overall efficiency of all three gradient descent by using appropriate threshold value.