FRAUD ANALYTICS

Assignment Number - 2

Example Dependent Cost Sensitive Classification

Sakshi Badole cs24mtech11008
Laveena Herman cs24mtech11010
S Anjana Shanker cs24mtech14015

1 Introduction

In many real-world applications, such as fraud detection and medical diagnosis, misclassifications lead to significant losses. A false negative might result risks, while a false positive may incur a lower cost. This imbalance proposes the use of cost-sensitive classification, where the learning algorithm explicitly accounts for varying costs of misclassification during training.

This assignment analyzes different approaches to cost-sensitive classification, focusing on logistic regression models. In particular, we compare the standard logistic regression approach with a cost-sensitive loss function(Bahnsen's approach) that integrates misclassification costs to minimize the overall expected cost. Through detailed experimentation and evaluation, this study provides insights into how these methods adjust decision thresholds and improve performance in cost-critical settings.

2 Cost Functions: Standard vs. Cost-Sensitive Approaches

In standard logistic regression, we minimize the *log-loss* (or cross-entropy loss):

$$J_{\text{standard}}(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\Big[y^{(i)}\log\big(h_\theta(x^{(i)})\big) + \big(1-y^{(i)}\big)\log\big(1-h_\theta(x^{(i)})\big)\Big], \tag{1}$$

where

$$h_\theta(x) = \sigma\big(\theta^T x\big) = \frac{1}{1+\exp\big(-\theta^T x\big)}$$

is the predicted probability for the positive class, and m is the number of training samples. Here, every misclassification is penalized equally.

In a cost-sensitive approach, however, different misclassification outcomes incur different costs. Define the costs for true positive (C_{TP}), false positive (C_{FP}), false negative (C_{FN}), and true negative (C_{TN}). The cost-sensitive objective is then:

$$J^c(\theta) = \frac{1}{m}\sum_{i=1}^{m}\Big(y^{(i)}\Big(h_\theta(x^{(i)})\,C_{TP}+\big(1-h_\theta(x^{(i)})\big)\,C_{FN}\Big)+\big(1-y^{(i)}\big)\Big(h_\theta(x^{(i)})\,C_{FP}+\big(1-h_\theta(x^{(i)})\big)\,C_{TN}\Big)\Big). \tag{2}$$

More simply, for each instance i, the cost contribution is:

$$J_i^c(\theta) = \begin{cases} C_{TP}, & \text{if } y^{(i)} = 1 \text{ and } h_\theta(x^{(i)}) \approx 1, \\ C_{TN}, & \text{if } y^{(i)} = 0 \text{ and } h_\theta(x^{(i)}) \approx 0, \\ C_{FP}, & \text{if } y^{(i)} = 0 \text{ and } h_\theta(x^{(i)}) \approx 1, \\ C_{FN}, & \text{if } y^{(i)} = 1 \text{ and } h_\theta(x^{(i)}) \approx 0. \end{cases} \tag{3}$$

For example, if we set

$$C_{TP} = 3, \quad C_{FP} = 3, \quad C_{FN} = C_i \text{ (row-specific)}, \quad C_{TN} = 0,$$

we define a weight for each sample as:

$$w_i = \begin{cases} C_i, & \text{if } y^{(i)} = 1, \\ 3, & \text{if } y^{(i)} = 0. \end{cases}$$

Then the cost-sensitive objective becomes a *weighted log-loss*:

$$J_{\text{cost-sensitive}}(\theta) = -\frac{1}{m}\sum_{i=1}^{m} w_i\Big[y^{(i)}\log\big(h_\theta(x^{(i)})\big) + \big(1-y^{(i)}\big)\log\big(1-h_\theta(x^{(i)})\big)\Big]. \tag{4}$$

Thus, by weighting instances according to their potential misclassification cost, the model directly minimizes the overall expected cost—a critical consideration in applications like fraud detection.

3 Modified Logistic Regression: Cost and Gradient

Given: $X \in \mathbb{R}^{m\times n}$, $y \in \{0,1\}^m$, $\theta \in \mathbb{R}^n$, fn_cost, λ

Forward Pass:

$$p = \sigma(X\theta), \quad \sigma(z) = \tfrac{1}{1+e^{-z}}$$

$$\text{cost}_i = y_i\Big(3\,p_i + (1-p_i)\,fn\_cost\Big) \; + \; (1-y_i)\,\big(3\,p_i\big)$$

$$J_{\text{base}} = \frac{1}{m}\sum_{i=1}^{m}\text{cost}_i$$

$$J_{\text{reg}} = \frac{\lambda}{2m}\sum_{j=2}^{n}\theta_j^2$$

$$J = J_{\text{base}} + J_{\text{reg}}$$

Backward Pass:

$$\text{factor}_i = \Big(y_i\,(3 - fn\_cost) \; + \; (1-y_i)\,3\Big)\,p_i\,(1-p_i)$$

$$\nabla_\theta J \; = \; \frac{1}{m}\,X^T\,\text{factor} \; + \; \frac{\lambda}{m}\begin{bmatrix} 0 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

Return J and $\nabla_\theta J$

4 Bahnsen-based Genetic Algorithm for Cost Minimization

```
1:  Input: Data $(X, y)$, misclassification costs $(cFN, cTP, cFP, cTN)$, population size,
    number of generations, mutation rate, elite fraction, patience.
2:  Output: Best parameter vector $\theta^*$.
3:
4:  Initialize: Generate a random population of candidate parameter vectors (optionally
    including a constant vector).
5:  for each generation $g = 1$ to $n\_generations$ do
6:      Evaluate: Compute the cost-sensitive objective for each candidate (using a
    weighted cost based on the Bahnsen formulation).
7:      Sort: Order candidates by increasing cost and update the best solution.
8:      if no improvement for patience generations then
9:          break                                                ▷ Early stopping
10:     end if
11:     Elitism: Retain the top fraction of candidates.
```

```
    Reproduction:
    while new population size < desired size do
        Select two parents using a hybrid (tournament/roulette) method.
        Generate offspring using uniform crossover.
        Apply adaptive mutation (with decaying mutation rate) to offspring.
        Add offspring to the new population.
    end while
    Update the population with the new generation.
end for
return the best candidate $\theta^*$.
```

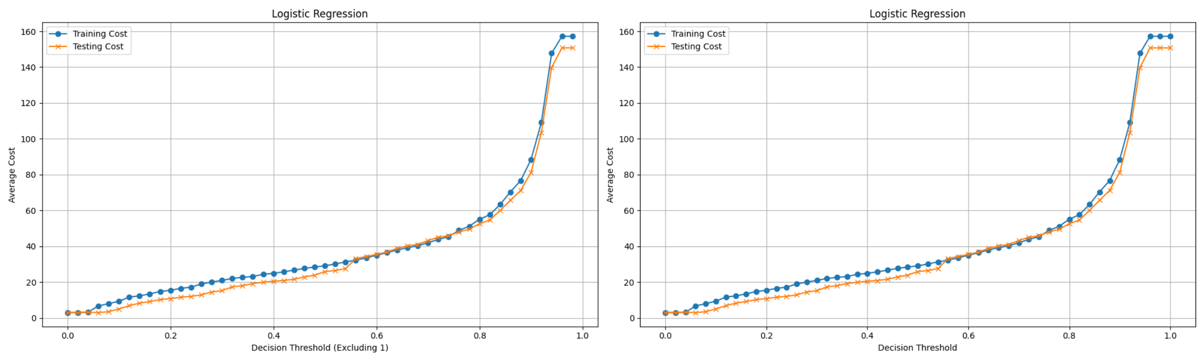
5 Dataset Description

- The dataset contains **147,636 rows** and **13 columns**.
- Columns **A to K** represent the **independent variables**.
- Column **L (Status)** is the **dependent variable**.
- Column **M (FNC)** represents the **false negative cost**, which varies for each row based on business details.
- The cost structure is:
  - **True Positive and False Positive Cost:** Fixed at 3.
  - **True Negative Cost:** Fixed at 0.
  - **False Negative Cost:** Varies significantly (min: 0, max: 1703186)
- The **dependent variable (Status)** is binary, indicating classification outcomes.

6 Logistic Model Results

This section presents the results for the standard logistic regression model. The training and testing costs for each threshold is analyzed below.

6.1 Plots



(a) Training Cost

Figure 1: Logistic Model: Training and Testing Cost

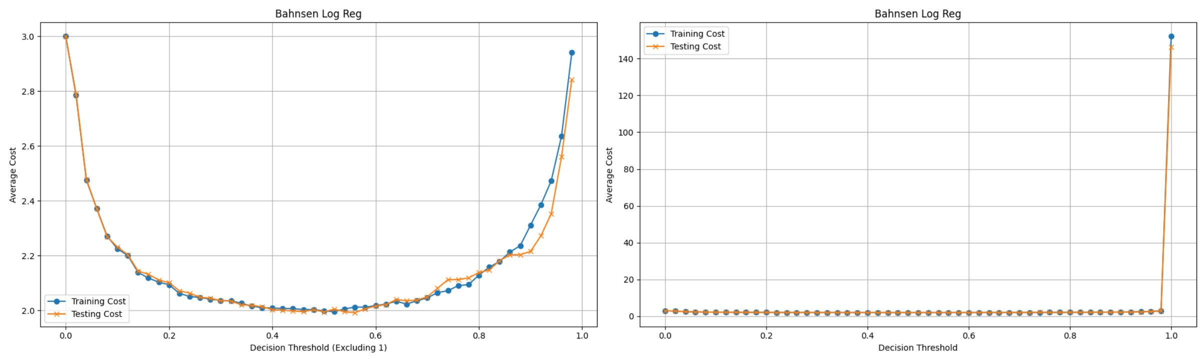
6.2 Observations

- The training and testing costs are increasing significantly with the increase in threshold.
- The model records the highest accuracy ranges.
- The lowest cost is observed when threshold is very low.

7 Cost Sensitive Model - Bahnsen Results

This section presents the results for the cost-sensitive model. The training and testing costs for each threshold is analyzed below.

7.1 Plots



(a) Training Cost

Figure 2: Cost Sensitive Model: Bahnsen - Training and Testing Costs

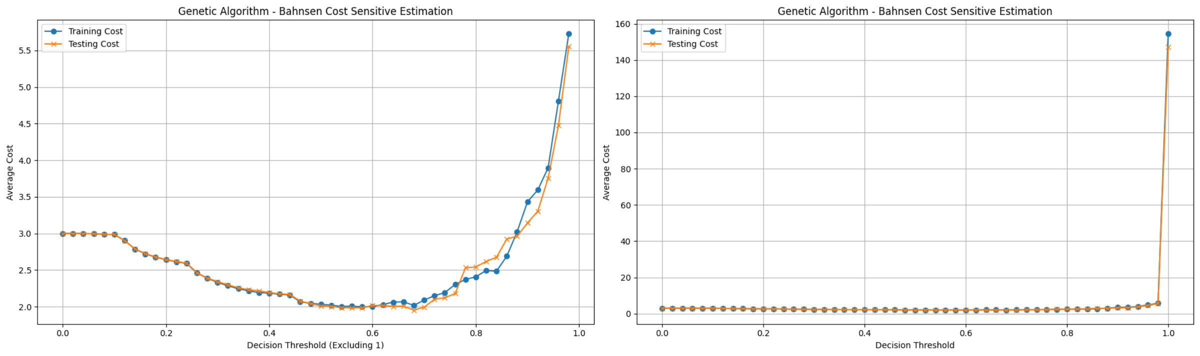
7.2 Observations

- The curve proves that as the threshold increases, the cost decreases initially, reaches a minimum, and then rises up.
- The cost is very low when compared with standard logistic regression.
- The cost doesn't change significantly when the threshold is varied.

8 Genetic Algorithm Model Results

This section presents the results for the application of a genetic algorithm on the cost-sensitive Bahnsen model. The training and testing costs for each threshold is analyzed below.

8.1 Plots



(a) Training Cost

Figure 3: Genetic Algorithm Model: Training and Testing Costs

8.2 Observations

- The cost doesn't vary significantly when the threshold is varied.
- The Bahnsen cost can be optimized with the use of a genetic algorithm.
- The cost is significantly lower than the logistic regression cost.

9 Comparison of All Models

This section presents a comparison of the three models (Logistic Regression, Cost-Sensitive Bahnsen, and Genetic Algorithm) across different thresholds.

9.1 Comparison Graph

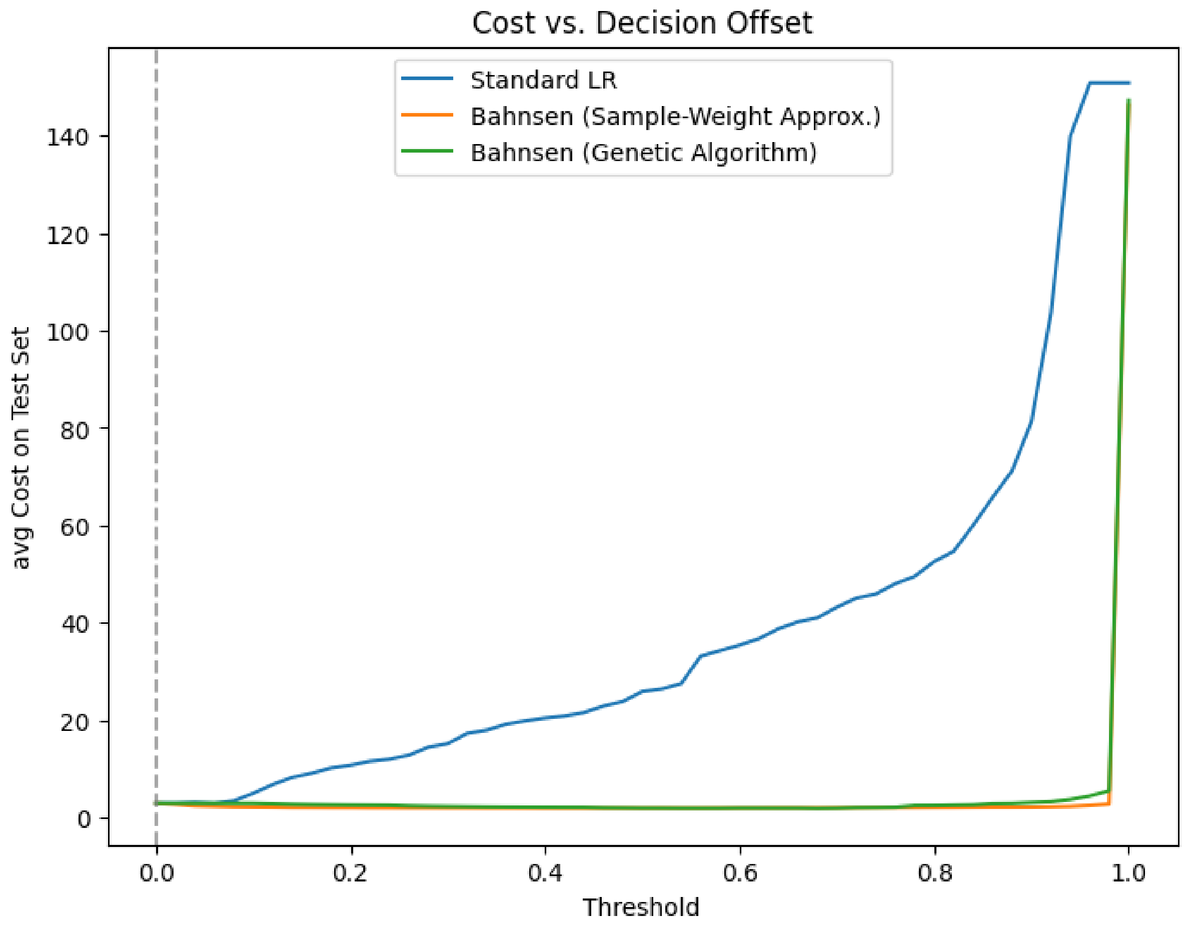


Figure 4: Comparison of Training and Testing Costs Across All Models

9.2 Model Performance Across Thresholds

The following tables compare the accuracy, training cost, and testing cost of the three models across five different thresholds.

Table 1: Model Accuracy Across Thresholds

Threshold	Logistic Regression	Bahnsen Model	Genetic Algorithm
0.1	0.756	0.543	0.303
0.3	0.856	0.617	0.521
0.5	0.866	0.649	0.639
0.7	0.862	0.677	0.716
0.9	0.830	0.707	0.783

Table 2: Training Cost Across Thresholds

Threshold	Logistic Regression	Bahnsen Model	Genetic Algorithm
0.1	9.322	2.225	2.987
0.3	20.906	2.037	2.331
0.5	29.063	1.999	2.032
0.7	41.957	2.046	2.090
0.9	88.404	2.311	3.431

Table 3: Testing Cost Across Thresholds

Threshold	Logistic Regression	Bahnsen Model	Genetic Algorithm
0.1	5.026	2.232	2.987
0.3	15.205	2.036	2.342
0.5	25.907	1.995	2.011
0.7	43.210	2.050	1.994
0.9	81.294	2.216	3.144

10 Observations and Conclusion

10.1 Key Observations

Table 4: Minimum Cost Threshold

Approach	Threshold	Training Cost	Testing Cost	Accuracy
Std Logistic Regression	0.06	6.670	2.968	0.300
LR + Bahnsen	0.56	2.013	1.933	0.658
LR + Bahnsen + Genetic Algorithm	0.68	2.017	1.953	0.706

- **Training Cost:** Logistic Regression shows a sharp cost increase (from 9.322 to 88.404), while Bahnsen and Genetic Algorithm models remain stable around 2.0-2.3. (From Threshold 0.4 - 0.8)
- **Testing Cost:** Logistic Regression incurs the highest testing cost increase (5.026 to 81.294), whereas Bahnsen and Genetic Algorithm models maintain low costs. (From Threshold 0.4 - 0.8)
- **Threshold Impact:** Higher thresholds increase costs significantly for Logistic Regression, while Bahnsen and Genetic Algorithm models remain stable.
- **Accuracy:** Logistic Regression achieves the highest accuracy (0.866 at threshold = 0.5), while Bahnsen and Genetic Algorithm models improve at higher thresholds.

10.2 Conclusion

- Logistic Regression with Bahnsen Model with cost function gave consistently low costs.
- Incorporation of Genetic Algorithm in this model maintains stable cost and reasonable accuracy.