Assignment Number 2: Example Dependent Cost Sensitive Classification

Team Members: CS24MTECH1400 Anurag Sarva

CS24MTECH14006 Gulshan Hatzade

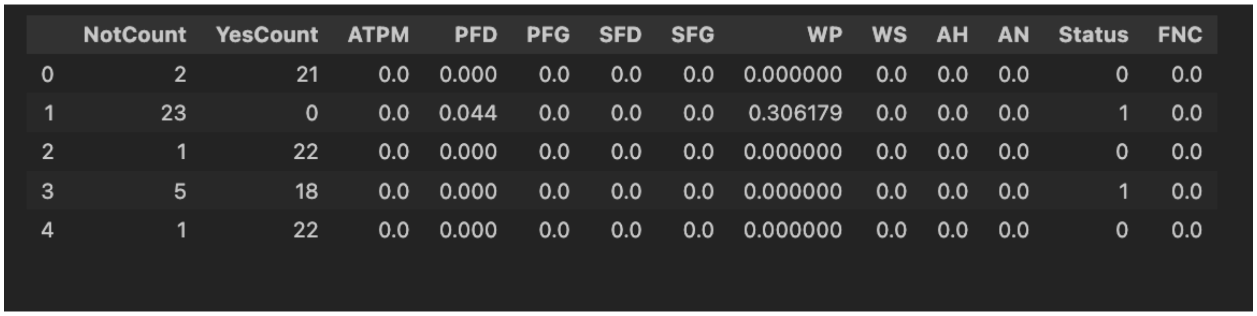
CS24MTECH14018 Mainak Adhikari

1. Problem Statement

Cost-sensitive classification is an essential technique for handling imbalanced datasets where the misclassification cost varies for different instances. In this report, we compare Standard Logistic Regression and Bahnsen's Cost-Sensitive Logistic Regression approaches to minimize classification cost effectively. The objective is to analyze the impact of varying probability thresholds (offset values) on the overall cost function.

2. Dataset Description

The dataset used for this study is provided in a .csv file named costsensitiveregression - costsensitiveregression.csv



	NotCount	YesCount	ATPM	PFD	PFG	SFD	SFG	WP	WS	AH	AN	Status	FNC
0	2	21	0.0	0.000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0	0.0
1	23	0	0.0	0.044	0.0	0.0	0.0	0.306179	0.0	0.0	0.0	1	0.0
2	1	22	0.0	0.000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0	0.0
3	5	18	0.0	0.000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	1	0.0
4	1	22	0.0	0.000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0	0.0

Table 1. The first 5 entries of dataset

It contains 13 columns:

- **Columns A to K (1-11)**: Independent variables representing the features of the data point.
- **Column L (12th column)**: Dependent variable (class label: 0 or 1).

- **Column M (13th column)**: False Negative Cost, which varies for each row based on business considerations.

Cost Considerations

- **False Negative Cost** (FN) varies row-wise and is specified in Column M.
- **True Positive (TP) Cost**: Constant at 3.
- **False Positive (FP) Cost**: Constant at 3.
- **True Negative (TN) Cost**: Constant at 0.

3. Methodology

3.1 Standard Logistic Regression

1. **Data Preprocessing**: The dataset is normalized using MinMaxScaler to bring all features within a uniform range.
2. **Splitting the Dataset**: The data is divided into an 80% training set and a 20% testing set.
3. **Model Training**: A standard logistic regression model is trained using the training dataset.
4. **Threshold-Based Prediction**: The model predicts probabilities, and a default threshold of 0.5 is used to classify instances as class 0 or 1.
5. **Cost Calculation**: Based on the misclassification type (False Positive or False Negative), the associated cost is computed using the predefined cost function.
6. **Performance Analysis**: The overall classification cost is calculated, providing insight into the model's efficiency in minimizing misclassification penalties.

   Uses the traditional log-loss function :

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} -y_i \log h_\theta(x_i) - (1 - y_i) \log(1 - h_\theta(x_i))$$

3.2 Cost-Sensitive Logistic Regression (Bahnsen's Approach)

1. **Dynamic Offset Exploration**: Instead of using a fixed threshold (0.5), various threshold values (offsets) are tested to evaluate their impact on classification cost.

2. **Optimization via Differential Evolution**: The differential_evolution algorithm is used to determine the optimal threshold that minimizes the total misclassification cost.

3. **Prediction with Optimized Offset**: Once the optimal threshold is identified, predictions are made accordingly.

4. **Cost Calculation**: Misclassification costs are computed as in standard logistic regression but with an optimized decision boundary.

5. **Performance Evaluation**: The model's average training and testing costs are compared to ensure it generalizes well without overfitting.

   Modifies the cost function by **incorporating instance-dependent classification costs**:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} y_i(CTP_i h_\theta(x_i) + CFN_i(1 - h_\theta(x_i))) + (1 - y_i)(CFP_i h_\theta(x_i) + CTN_i(1 - h_\theta(x_i)))$$

   This **removes the log function** and explicitly accounts for **True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN) costs**.

3.3 Cost Function Calculation

For each prediction:

- **False Negative (y_true=1, y_pred=0)**: Adds cost from Column M.

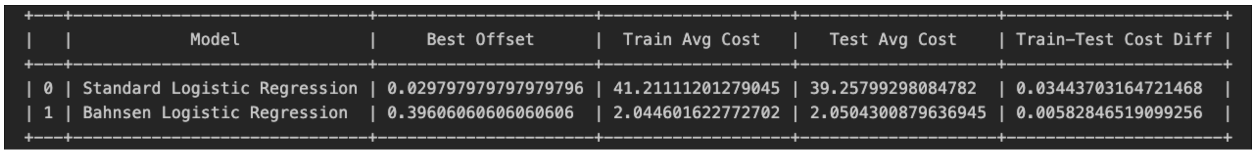
- **True Positive (y_true=1, y_pred=1)**: Adds 3.

- **False Positive (y_true=0, y_pred=1)**: Adds 3.

- **True Negative (y_true=0, y_pred=0)**: Adds 0.

The average classification cost is then calculated for both models.

3.4 Comparison Between Standard and Cost-Sensitive Logistic Regression



```
+---+------------------------------+----------------------+-------------------+--------------------+----------------------+
|   |            Model             |     Best Offset      |  Train Avg Cost   |   Test Avg Cost    | Train-Test Cost Diff |
+---+------------------------------+----------------------+-------------------+--------------------+----------------------+
| 0 | Standard Logistic Regression | 0.029797979797979796 | 41.21111201279045 | 39.25799298084782  | 0.03443703164721468  |
| 1 | Bahnsen Logistic Regression  | 0.39606060606060606  | 2.044601622772702 | 2.0504300879636945 | 0.00582846519099256  |
+---+------------------------------+----------------------+-------------------+--------------------+----------------------+
```

Table 2. Comparison table

1. **Performance Improvement:**

   The **Bahnsen Logistic Regression** significantly reduces the average cost compared to the **Standard Logistic Regression**.

   Training cost drops from **41.21** to **2.04**, and test cost decreases from **39.26** to **2.05**.

2. **Better Generalization:**

   The **Train-Test Cost Difference** for Bahnsen Logistic Regression is **0.0053**, which is much smaller than **0.0344** for Standard Logistic Regression.

   This indicates that the Bahnsen model generalizes better and is more robust to unseen data.

3. **Effect of Best Offset:**

   The optimal offset for Bahnsen Logistic Regression (**0.3960**) is much higher than for Standard Logistic Regression (**0.0298**), showing that threshold optimization plays a crucial role in cost reduction.

The **Bahnsen Logistic Regression** outperforms the Standard Logistic Regression model by a large margin, achieving **lower average costs and better generalization**. This highlights the importance of **optimized decision thresholds** in reducing misclassification costs.

4. Results and Analysis

4.1 Standard Logistic Regression Results

Predictions are made at different threshold, we considered 100 offsets between 0.01 to 0.99 .

Standard Logistic Regression - Training Cost Function vs. Offset

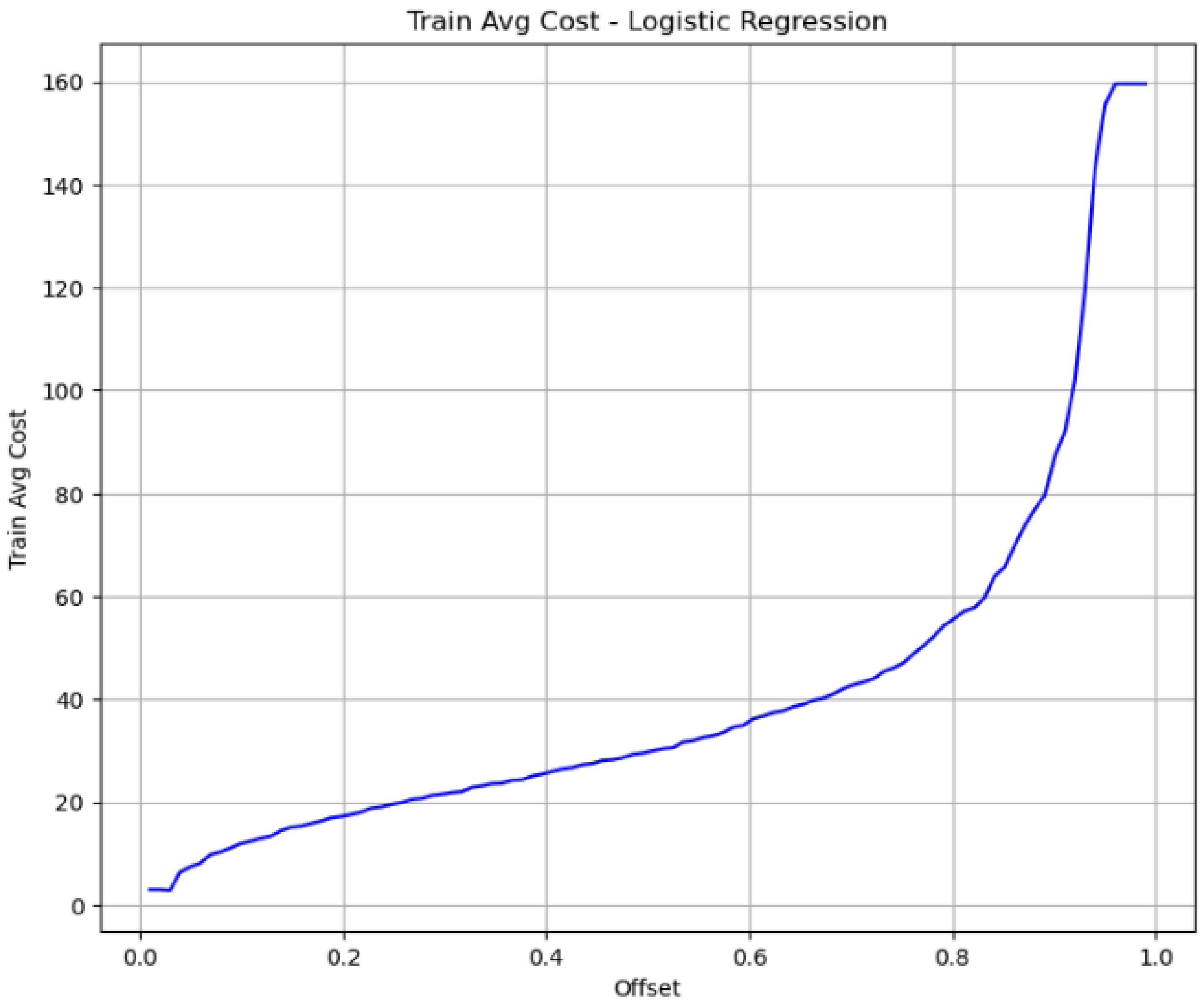


Figure 1: Standard Logistic Regression - Training Cost Function vs. Offset

Standard Logistic Regression - Testing Cost Function vs. Offset

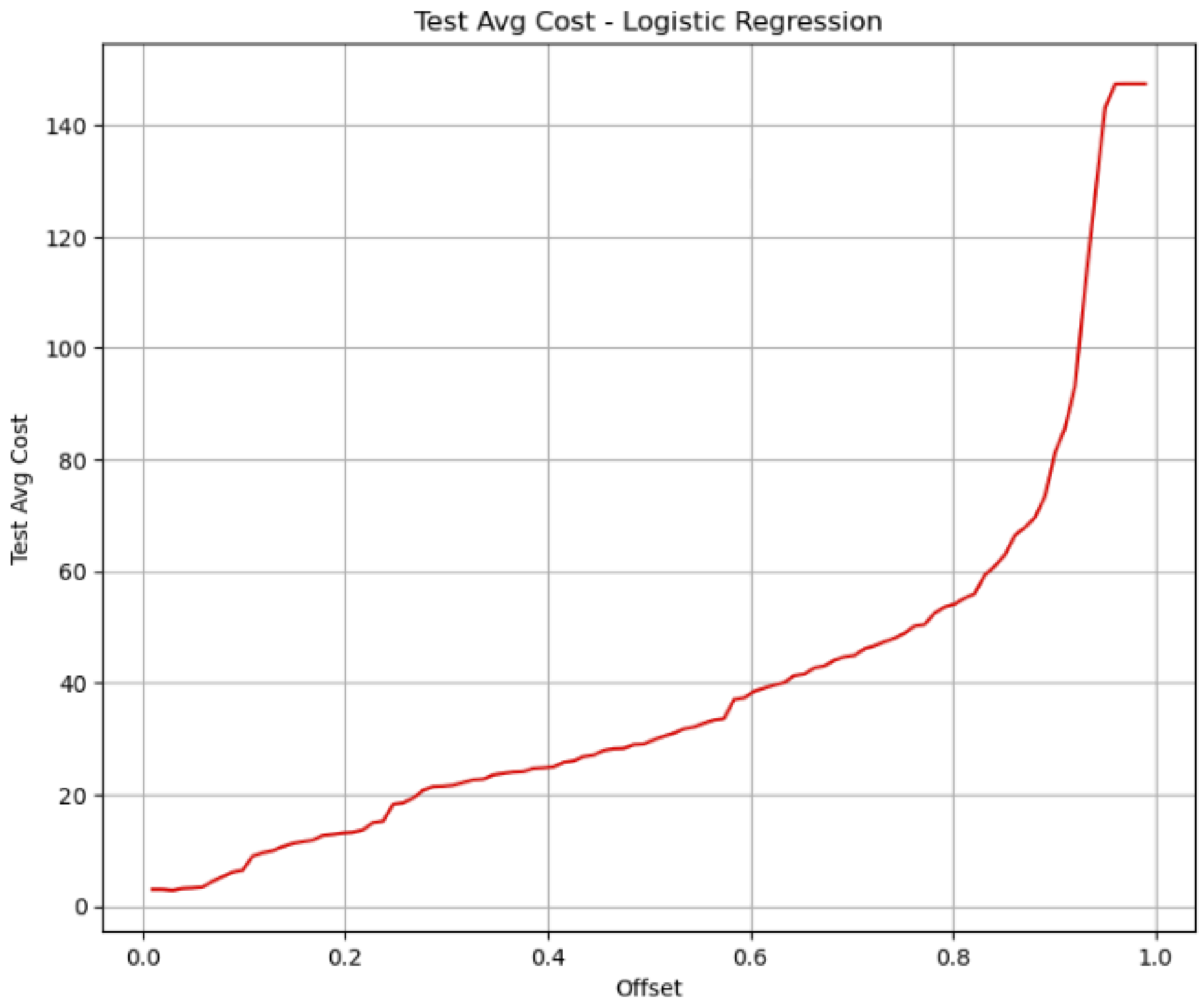


Figure 2: Standard Logistic Regression - Testing Cost Function vs. Offset

Both graphs show an increasing cost trend as the offset increases.

The sharp rise at the end suggests a point where the model struggles to optimize effectively.

Offset represents probability threshold, it might indicate that a high value leads to instability in the model.

The test cost follows a similar pattern to the train cost.

4.2 Cost-Sensitive Logistic Regression Results

- Predictions are made at different threshold, we considered 100 offsets between 0.01 to 0.99 .
- The optimized offset reduces the overall cost compared to the standard approach.
- The differential evolution method finds the optimal threshold that minimizes the cost function.

Cost-Sensitive Logistic Regression - Training Cost Function vs. Offset

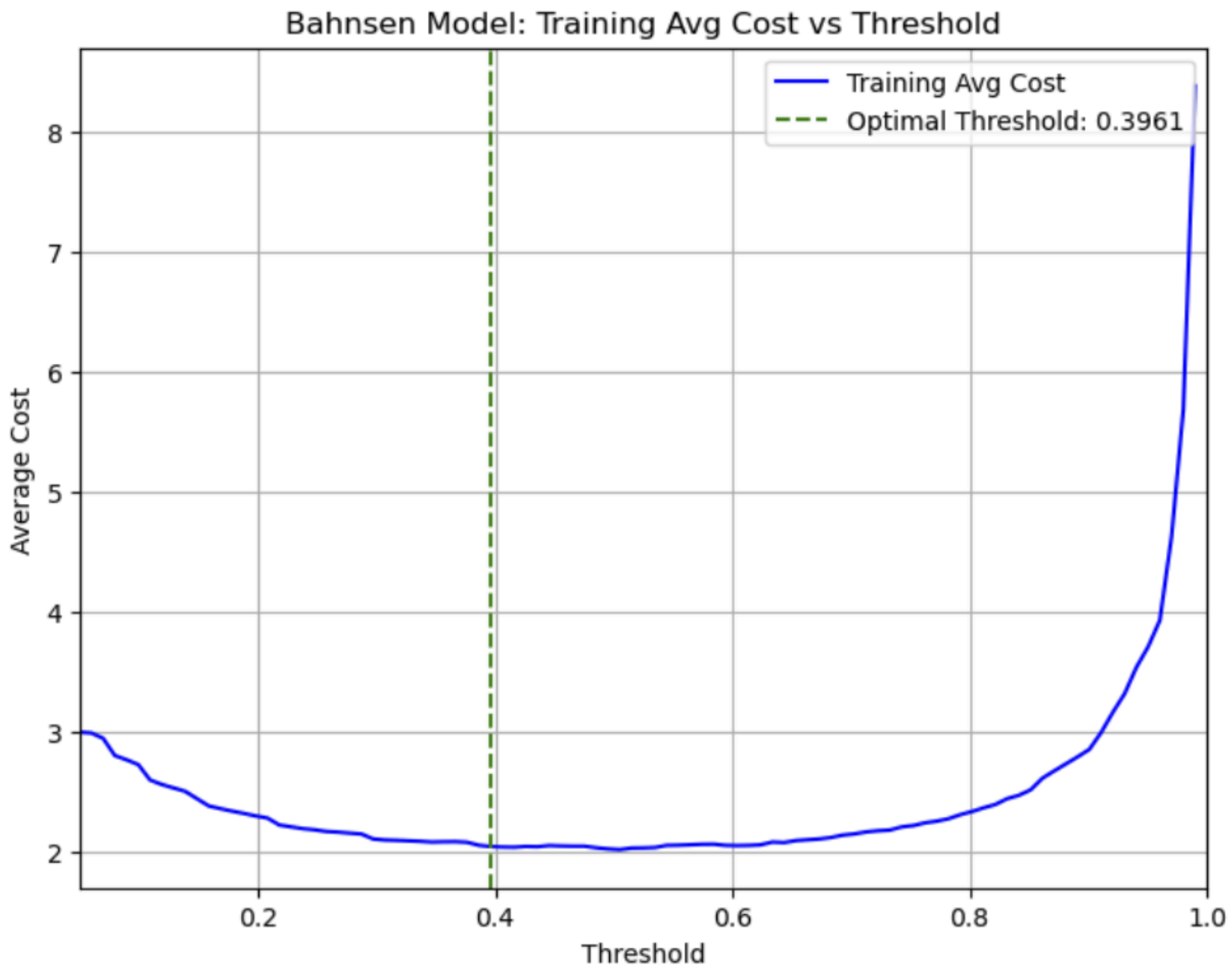


Figure 3: Cost-Sensitive Logistic Regression - Training Cost Function vs. Offset

Cost-Sensitive Logistic Regression - Testing Cost Function vs. Offset

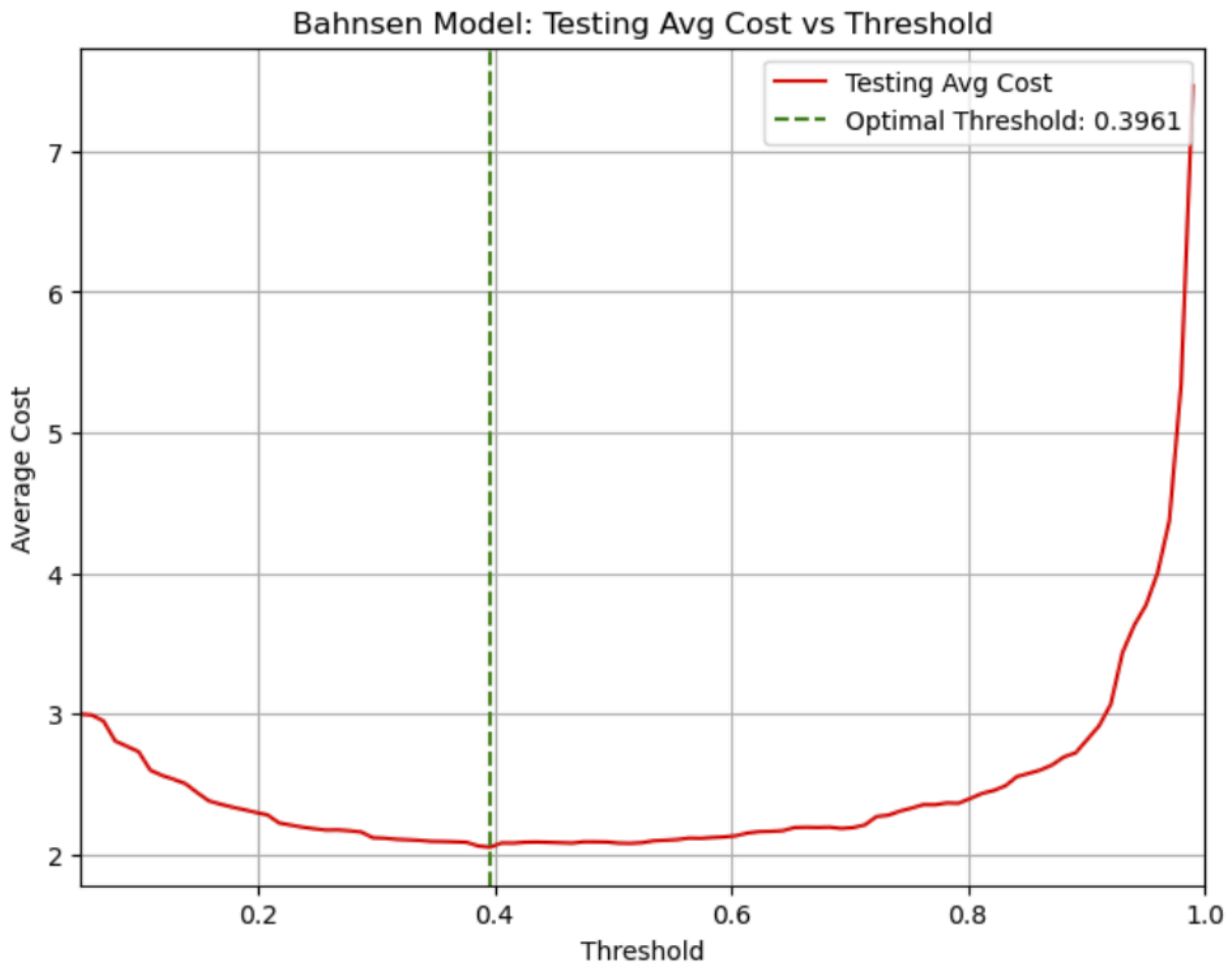


Figure 4: Cost-Sensitive Logistic Regression - Testing Cost vs. Offset

The average cost initially decreases and then increases sharply as the threshold increases.

The optimal threshold (~0.3961) minimizes the cost in both training and testing.

This suggests that tuning the threshold is crucial for achieving the best performance.

5. Conclusion

1. Logistic regression exhibits increasing costs with larger offsets, potentially indicating poor model fit.

2. The Bahnsen model benefits from threshold tuning, with an optimal threshold (~0.3961) that minimizes cost effectively.

3. The Bahnsen model appears to provide better cost control compared to logistic regression.
4. The cost vs. offset graph clearly illustrates that an optimized threshold can improve decision-making in cost-sensitive scenarios.

Thus, for applications where false negative costs vary across samples, Bahnsen’s approach is more effective than the standard logistic regression model.