Analytical Report:

Regularization Techniques and Heuristic Optimization Methods for Logistic Regression

This analytical report provides a comprehensive examination of regularization techniques and metaheuristic optimization methods in logistic regression, drawing from both theoretical foundations and empirical analysis. The investigation combines conceptual understanding with mathematical derivations and implementation results to provide insights into how different optimization approaches perform on classification tasks.

1. Introduction

Logistic regression remains one of the most widely used classification algorithms in machine learning due to its simplicity, interpretability, and effectiveness. However, traditional optimization methods for finding optimal parameters can suffer from issues such as slow convergence, getting trapped in local optima, and overfitting. This report explores how regularization techniques can mitigate overfitting, while metaheuristic optimization algorithms can efficiently navigate complex parameter spaces to find optimal solutions.

2. Conceptual Understanding

2.1 Logistic regression overview:

Logistic regression is a statistical model used for binary classification that estimates the probability of an instance belonging to a particular class. Unlike linear regression, logistic regression applies a sigmoid function to the linear combination of input features to constrain the output between 0 and 1, making it suitable for probability estimation.

The core equation of logistic regression is:

$$P(y_i = 1|x_i) = \sigma(\beta_0 + \beta^T x_i) = \frac{1}{1+e^{-(\beta_0+\beta^T x_i)}}$$

Where,

- σ = Sigmoid function
- β_0 = Intercept term
- β = Coefficient vector
- x_i = Feature vector from i^{th} iteration

2.2 Regularization Techniques:

Regularization addresses overfitting by adding penalty terms to the loss function that discourage complex models with large coefficient values. The two most common regularization techniques are L1 (Lasso) and L2 (Ridge) regularization.

2.1.1 L1 Regularization (Lasso)

L1 regularization adds a penalty term proportional to the absolute magnitude of coefficients:

$$min_w ||y - Xw||_2^2 + \alpha||w||_1^1$$

L1 regularization is characterized by:

- Non-differentiability at zero, leading to sparse solutions
- Geometric constraints forcing coefficients to lie within an -ball (diamond/octahedron)
- Feature selection capabilities by setting some coefficients exactly to zero
- Better performance when many features are irrelevant

2.1.2 L2 Regularization (Ridge)

L2 regularization adds a penalty term proportional to the squared magnitude of coefficients:

$$min_w ||y - Xw||_2^2 + \alpha||w||_2^2$$

L2 regularization is characterized by:

- Smooth, differentiable penalty that shrinks coefficients uniformly
- Geometric constraints forcing coefficients to lie within an -ball (sphere)
- Dense solutions with all coefficients non-zero but reduced in magnitude
- Better performance when most features contribute to the prediction

2.3 Heuristic Optimization Methods:

2.3.1 Particle Swarm Optimization (PSO)

PSO is a population-based algorithm inspired by the social behavior of bird flocking or fish schooling. In PSO, several particles navigate the parameter space to find optimal solutions.

Each particle maintains:

- Its current position (representing a potential solution)
- Its velocity (direction and speed of movement)

- Its personal best solution (the best solution it has found so far)
- Awareness of the global best solution (the best solution found by any particle in the swarm)

The particles update their velocities based on three components:

1. Momentum: The particle maintains a portion of its initial velocity
2. Cognitive intelligence: The particle is attracted toward its personal best position
3. Social intelligence: The particle is attracted toward the global best position

The velocity update rule is given by:

$$V_{(t+1)} = (\omega \cdot V_t) + (c_1 \cdot r_1 \cdot (P_t - X_t) + (c_2 \cdot r_2 \cdot (G_t - X_t)$$

Where,

- V_t is the current velocity
- ω is the inertia weight controlling momentum
- c_1 and c_2 are the acceleration coefficients for cognitive and social components
- r_1 and r_2 are the random numbers between 0 and 1
- P_t is the personal best position
- G_t is the global best position
- X_t is the current position

PSO balances exploitation (refining known good solutions) through momentum and exploration (searching new areas of the parameter space) through cognitive and social components.

2.3.2 Genetic Algorithm (GA)

Genetic Algorithms are inspired by biological evolution and natural selection. GA maintains a population of candidate solutions and evolves them through generations using genetic operators.

GA generates new solutions through:

1. **Crossover:** New solutions are created by combining parts of two parent solutions, mimicking genetic recombination. For example, segments of one parent's solution may be combined with segments from another parent.
2. **Mutation:** Random changes are introduced to existing solutions, such as adding random noise or altering small portions randomly. This ensures exploration of previously unexplored regions of the search space.

These mechanisms help maintain diversity in the population, which is crucial for avoiding local optima. Crossover promotes exploration by recombining existing knowledge, while mutation introduces randomness to prevent premature convergence.

In GA, the exploration-exploitation balance is controlled primarily by:

- Crossover rate: Higher rates promote exploration by generating diverse offspring
- Mutation rate: Higher rates encourage exploration of new areas, while lower rates focus on exploitation of existing solutions
- Population size: Larger populations maintain more diversity but require more computational resources

2.3.3 Simulated Annealing

Simulated Annealing is inspired by the annealing process in metallurgy, where materials are heated and then slowly cooled to reduce defects. SA uses a temperature parameter that gradually decreases over time, controlling the algorithm's willingness to accept worse solutions.

The probability of accepting a worse solution is given by:

$$P = exp(- \Delta E/T)$$

Where,

- $\Delta E$is the change in the objective function value
- T is the current temperature

SA decides whether to accept worse solutions using this probabilistic approach. If the new solution is better, it is always accepted. If the new solution is worse, it may still be accepted with a probability that decreases as the temperature cools.

By occasionally accepting worse solutions, especially early in the process when the temperature is high, SA can escape local minima and explore other regions of the search space. This stochastic behavior allows the algorithm to "jump out" of suboptimal solutions.

The exploration-exploitation balance in SA is controlled by:

- Temperature (T): High temperatures promote exploration by allowing acceptance of worse solutions
- Cooling schedule: Determines how quickly the temperature decreases, with slower cooling allowing more exploration

3. Mathematical Derivations

3.1 Negative Log-Likelihood for Logistic Regression

The Negative Log-Likelihood for Logistic Regression is derived as:

$$NLL = (-1) \cdot log(L(\beta_0, \beta)) = (-1) \cdot \sum_{i=1}^{N} [y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)]$$

Where, $p_i = \sigma(\beta_0 + \beta^T x_i)$

This derivation represents the cross-entropy loss function commonly used in logistic regression.

3.2 Gradient Derivation for Logistic Regression

The derived equation for gradient calculations w.r.t. intercept β_0 and feature vector β are given as:

$$\frac{\partial NLL}{\partial \beta_0} = \sum_{i=1}^{N} [p_i - y_i]$$

$$\frac{\partial NLL}{\partial \beta} = \sum_{i=1}^{N} [p_i - y_i] x_i$$

These gradient equations are used in optimization algorithms to update the parameters.

3.3 Regularization Effects on Search Space

$$NLL_{L1} = (-1) \sum_{i=1}^{N} [y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)] + \lambda ||\beta||_1^1$$

$$NLL_{L2} = (-1) \sum_{i=1}^{N} [y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)] + \lambda ||\beta||_2^2$$

Above equation represents penalised objective functions using L1 regularization and L2 regularization, respectively.

The effect of regularization on the search space differs between L1 and L2:

1. L1 regularization alters the search space into a “diamond” or “tetrahedron” shape, promoting sparsity by driving some coefficients exactly to zero.
2. L2 regularization transforms the search space into a “spherical” or “circular” shape, shrinking all coefficients proportionally without necessarily zeroing them out.

These geometric effects have important implications for optimization:

- L1 regularization can simplify models by eliminating irrelevant features
- L2 regularization stabilizes models, especially when features are correlated

4. Experimental Analysis

4.1 Dataset Description and Exploratory Data Analysis

The analysis uses the Breast Cancer Wisconsin dataset from scikit-learn, which contains features computed from digitized images of fine needle aspirates (FNA) of breast masses.

- The dataset includes:

- **559** instances
- **30** features representing various properties of cell nuclei
- Binary classification task: benign (**357** samples) or malignant (**212** samples)

- The data preprocessing steps included

- Check for **missing values** (none found)
- Splitting into training (**70%**) and test (**30%**) sets
- **Standardizing** features using StandardScaler

4.2 Implementation Details

Three metaheuristic optimization algorithms were implemented to train the logistic regression model:

1. **Particle Swarm Optimization (PSO)**

   Parameters: 100 particles, 75 iterations, inertia weight (w) = 0.7, cognitive (c1) and social (c2) coefficients = 1.5

2. **Genetic Algorithm (GA)**

   Parameters: 50 individuals, 100 generations, crossover rate = 0.8, mutation rate = 0.1

3. **Simulated Annealing (SA)**

   Parameters: Initial temperature = 273, cooling rate = 0.95, maximum iterations = 1000

Each algorithm was implemented with both L1 and L2 regularization options, with regularization parameter $\lambda = 0.01$.

The objective function for optimization was the negative log-likelihood with regularization:

$$NLL_{L2} = (-1) \sum_{i=1}^{N} [y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)] + \lambda ||\beta||_r^r$$

r = 1, for L1 regularization

r = 2, for L2 regularization

4.3 Results and Discussion

4.3.1 Particle Swarm Optimization

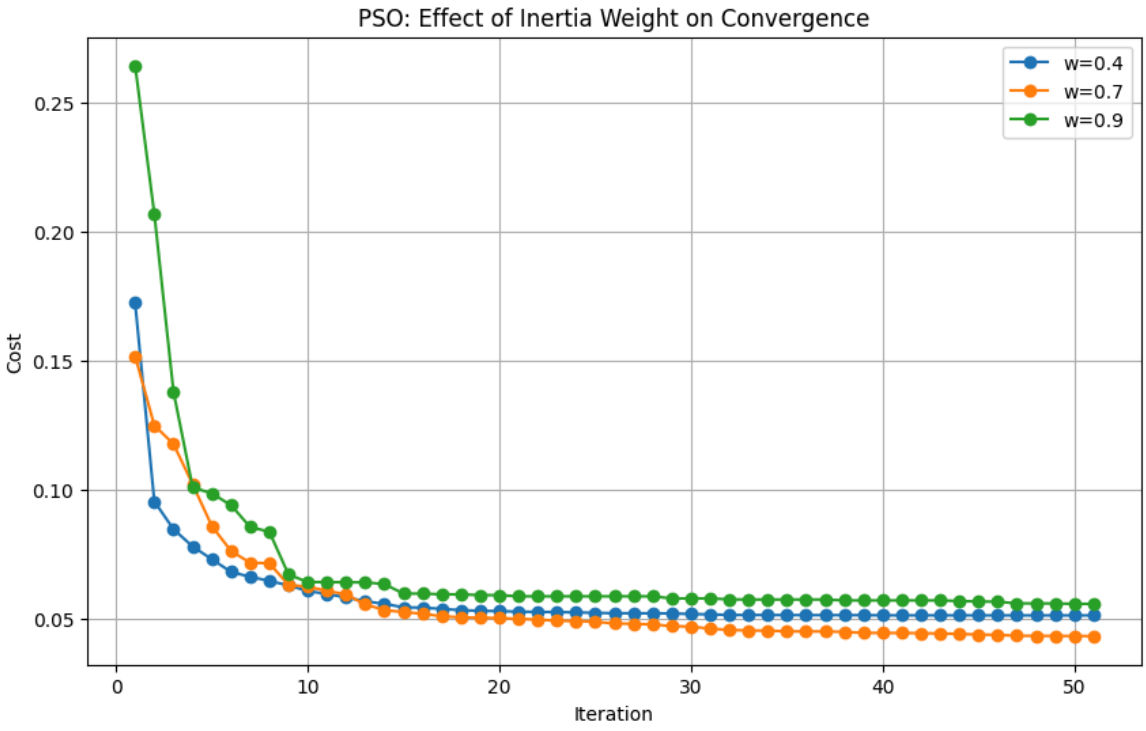
PSO showed rapid initial convergence, with significant cost reduction in the first 20 iterations. The algorithm achieved high accuracy on both training and test sets:

- Test accuracy: **97.66%**
- Training accuracy: **98.74%**

Hyperparameter analysis revealed that the inertia weight (w) significantly influenced convergence:

- w = 0.4: Fast convergence but risk of premature convergence
- w = 0.7: Balanced exploration and exploitation
- w = 0.9: Slower convergence but more exploration of the parameter space

L1 regularization with PSO resulted in a sparser model (9.7% of coefficients near zero) compared to L2 regularization (6.5% of coefficients near zero), demonstrating the feature selection capability of L1 regularization.



4.3.2 Genetic Algorithm

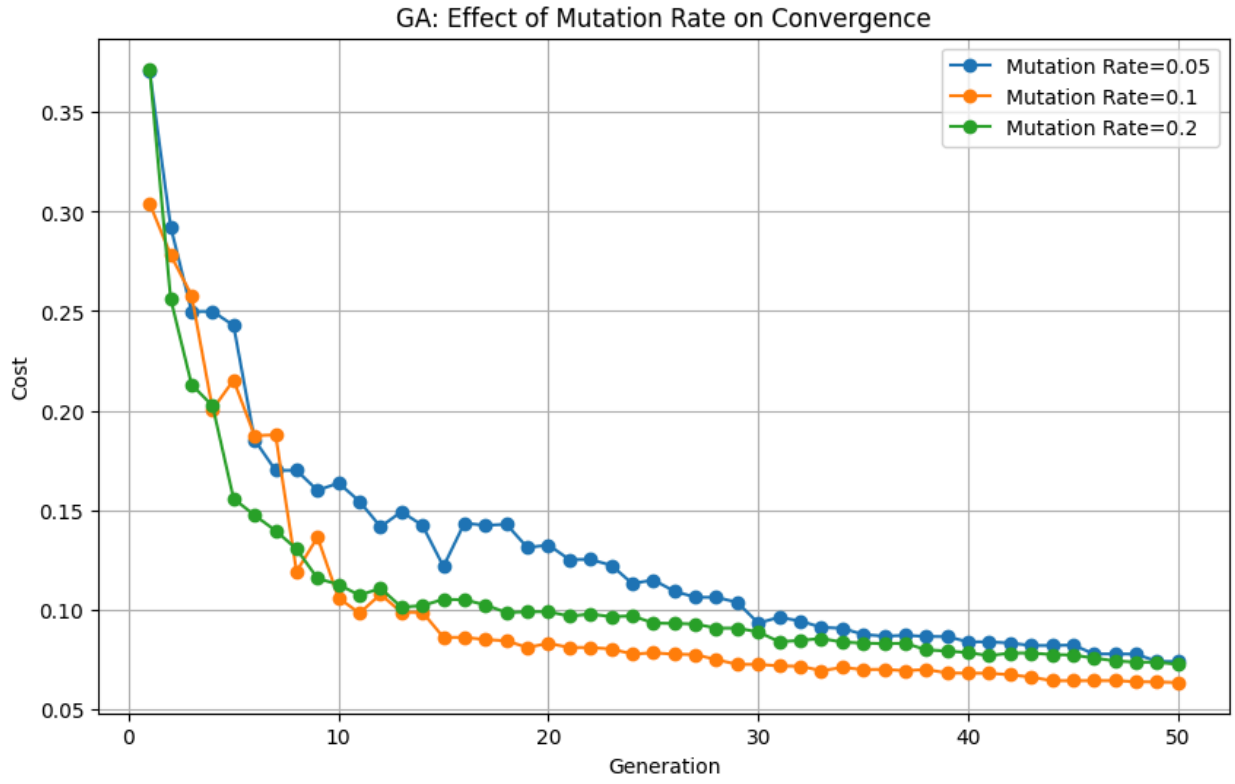
GA showed more gradual convergence compared to PSO but achieved excellent final results:

- Test accuracy: 99.42%
- Training accuracy: 98.24%

The mutation rate significantly affected GA's performance:

- Low mutation rate (0.05): Faster convergence but risk of local optima
- Medium mutation rate (0.1): Balanced exploration and exploitation
- High mutation rate (0.2): More exploration but potentially slower convergence

The GA implementation produced a confusion matrix with only 1 false positive and 0 false negatives on the test set, indicating superior classification performance.



4.3.3 Simulated Annealing

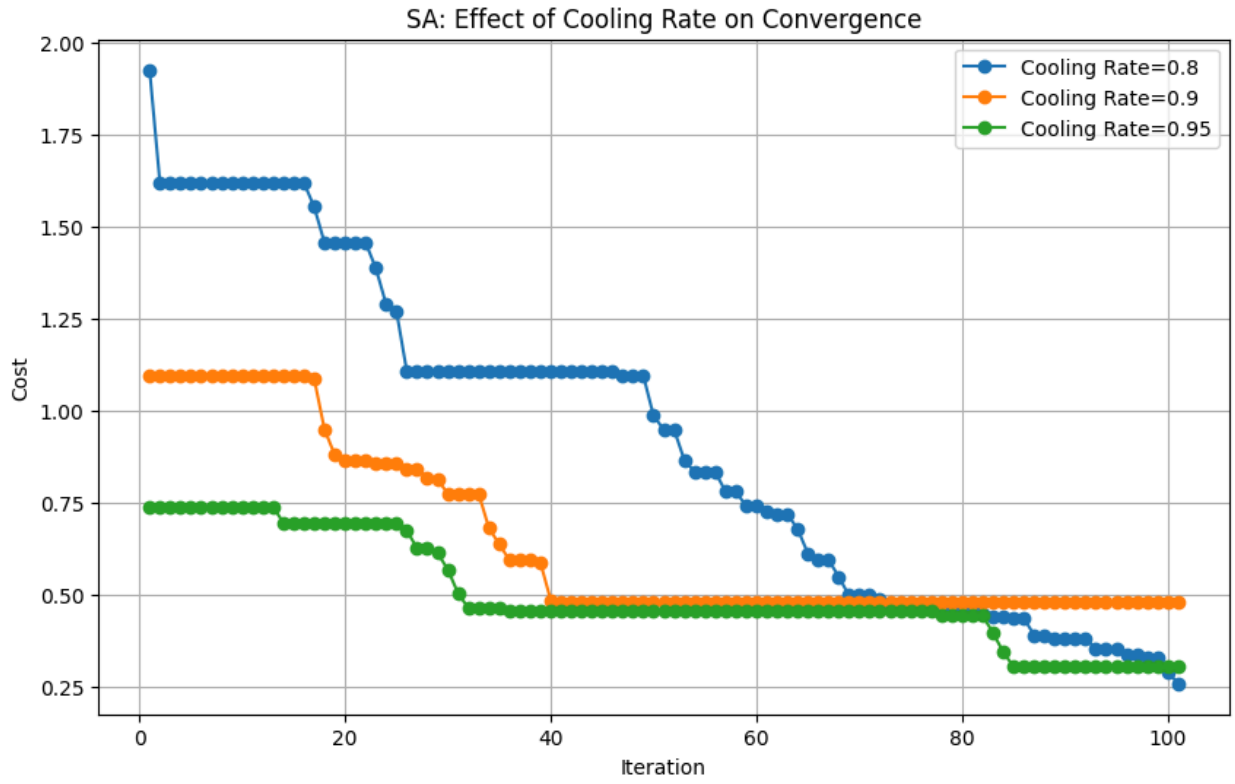
SA demonstrated steady improvement over iterations, with performance dependent on the cooling schedule:

- Test accuracy: 98.25%
- Training accuracy: 98.74%

Cooling rate analysis showed:

- Fast cooling (0.8): Quick convergence but potential for local optima
- Medium cooling (0.9): Balanced approach
- Slow cooling (0.95): More thorough exploration of the parameter space

SA with L2 regularization produced a confusion matrix with 1 false positive and 2 false negatives on the test set.



4.3.4 Comparative Analysis

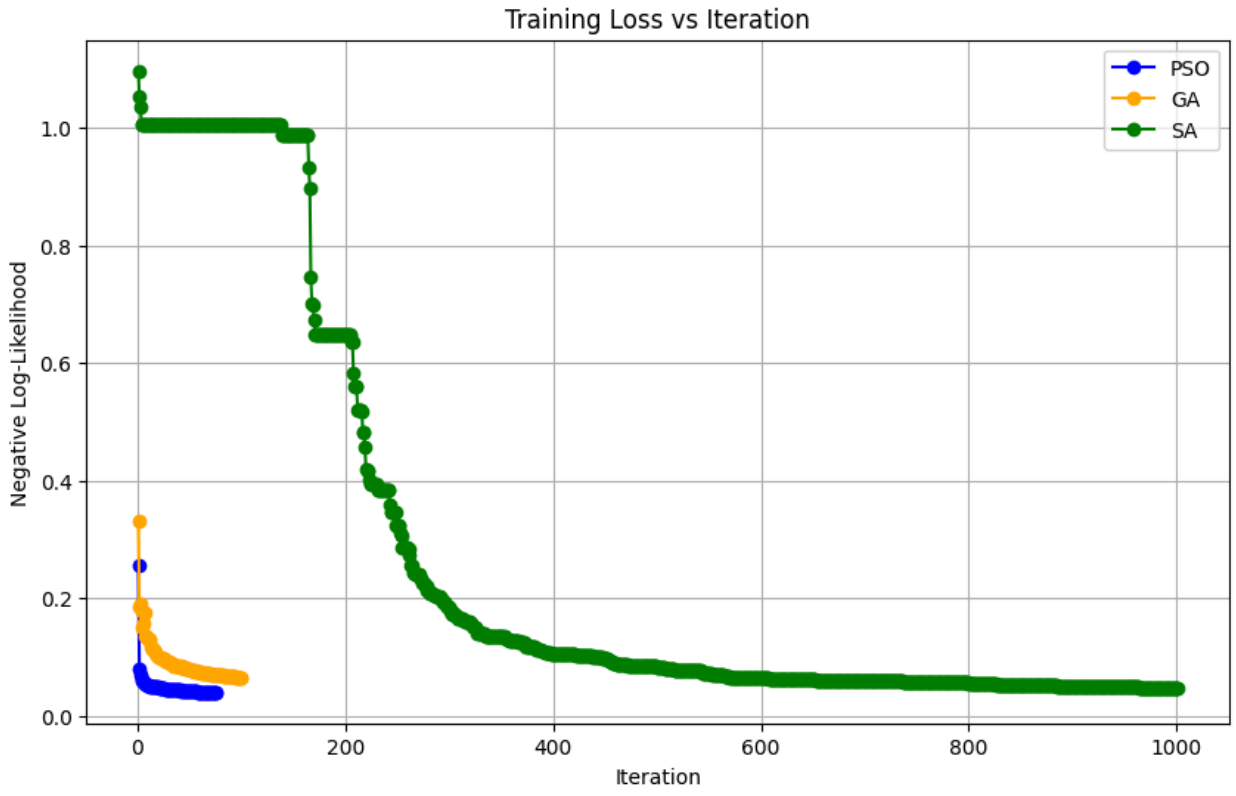
Algorithm	Test Accuracy	Training Accuracy	AUC
PSO	97.66%	98.74%	0.99
GA	99.42%	98.24%	0.99
SA	98.25%	98.74%	0.99

- Training, Testing accuracy, and AUC score for all algorithms

All three algorithms achieved high performance, with GA slightly outperforming the others on test accuracy. The ROC curves for all algorithms showed excellent classification ability, with AUC values approaching 0.99.

L1 vs. L2 regularization comparison revealed:

- L1 regularization produced sparser models with some coefficients driven to zero
- L2 regularization maintained more non-zero coefficients but with smaller magnitudes
- L1 regularization showed more feature differentiation, with larger differences between important and less important features



5. Conclusion

This report has explored the theoretical foundations and practical implementation of regularization techniques and metaheuristic optimization methods for logistic regression. The key findings include:

1. All three metaheuristic algorithms (PSO, GA, and SA) successfully optimized the logistic regression model for the breast cancer classification task, achieving high accuracy and AUC values.

2. GA demonstrated the highest test accuracy (99.42%), making it particularly suitable for this classification problem.
3. Hyperparameter tuning significantly influences algorithm performance:
    - PSO: Inertia weight balances exploration and exploitation
    - GA: Mutation rate affects diversity and convergence
    - SA: Cooling schedule determines the exploration-exploitation trade-off
4. L1 regularization promotes sparsity and implicit feature selection, while L2 regularization provides smooth coefficient shrinkage.
5. The geometric effects of regularization on the search space directly impact the optimization process and resulting model complexity.

The study demonstrates the effectiveness of metaheuristic optimization methods for training regularized logistic regression models, especially for complex, high-dimensional problems where traditional optimization methods might struggle.