The RL classifier: A Reinforcement Learning Approach for Classification

Yongchao Huang [*]

August 2, 2025

Abstract

Standard classification algorithms are typically trained by optimizing metrics such as cross-entropy loss, which treats all misclassifications equally. However, in many real-world scenarios, such as medical diagnosis or fraud detection, the costs of different errors are highly asymmetric. This paper proposes a novel framework for classification by framing it as a Reinforcement Learning (RL) problem. We treat the classifier as an agent, the selection of a class label as an action, and the correctness of that action as a reward signal. This paradigm allows for the design of highly customized, cost-sensitive reward functions that directly reflect the true objective of the task. We demonstrate the viability of this *RL classifier* using foundational policy-gradient algorithms on the classic *Iris* dataset. Our results show that this approach not only matches the performance of traditional methods on standard metrics but also provides a flexible and powerful framework for building classifiers that are optimized for complex, real-world cost structures.

1 Introduction

Classification, a fundamental task in machine learning, aims to assign a categorical label to a given input. Traditional approaches are broadly divided into two categories: discriminative and generative classification models. Discriminative classifiers, such as Logistic Regression or Support Vector Machines, learn a direct mapping from inputs to class labels by optimizing a smooth loss function such as cross-entropy. Generative classifiers [5], e.g. Naive Bayes, learn the joint probability distribution of the data. While effective, both paradigms are fundamentally tied to these predefined optimization objectives, which often assume that all misclassification errors are equally costly.

This RL-based approach offers distinct advantages over these traditional methods. The RL classifier, by contrast, is optimized to maximize an arbitrary reward signal. This allows a practitioner to directly encode complex, real-world business logic into the model's objective, such as asymmetric penalties for misclassification (e.g. a false negative in a medical diagnosis being far more costly than a false positive), a feat that is difficult or impossible to achieve with standard loss functions alone.

2 Related Work

The concept of framing supervised learning tasks within a reinforcement learning paradigm has a rich history. The foundational idea can be traced back to early policy gradient methods, which sought to optimize a parameterized policy directly without an explicit value function. The REINFORCE

[*] Author email: yongchao.huang@abdn.ac.uk

algorithm, introduced by Williams [11], provided a seminal framework for this, demonstrating how to adjust a network's weights based on a reward signal, thereby bridging the gap between traditional supervised learning and RL. Our work directly builds on this by treating a classifier as a policy to be optimized. While effective, the REINFORCE algorithm is known to suffer from high variance in its gradient estimates, which can lead to unstable training [3]. A major advancement to address this was the development of Actor-Critic methods. The theoretical groundwork for Actor-Critic algorithms was laid by Konda and Tsitsiklis [6], who proved their convergence properties. These methods reduce variance by introducing a Critic to learn a baseline value function, against which the Actor's actions are assessed. This approach was popularized and made highly effective in the deep learning era with the Asynchronous Advantage Actor-Critic (A3C) algorithm and its synchronous counterpart A2C [8], which we employ in our later experiments.

Applying RL to classification is an active area of research, particularly in domains where the decision-making process is complex. For example, RL has been used to create attention mechanisms for image classification [9] and to develop cost-sensitive classifiers where the penalty for different types of errors varies significantly [13]. This paper contributes to this line of research not by introducing a new algorithm, but by providing a clear, progressive, and tutorial-like demonstration of how these foundational RL techniques can be used to build a flexible and powerful classifier, with a specific focus on showcasing the practical benefits of custom reward engineering for asymmetric cost objectives. The specific problem formulation in this paper, i.e. a single-step decision process where an agent observes a context (state), selects an action, and receives an immediate reward (without observing rewards for other action), is formally known as the *contextual bandit* problem [7, 1, 2]. Our work treats each classification instance as a contextual bandit problem, where the features are the context (state) and the class label is the action.

3 Approach

To reframe a classification task as an RL problem, we define the standard components as follows:

- **State (S):** the input features X of a data sample.
- **Action (A):** the agent's choice of a class label from a discrete set of K classes, $A \in \{0, 1, ..., K-1\}$.
- **Reward (R):** a scalar value determined by the correctness of the chosen action. For a simple accuracy-based objective, the reward can be defined as: $R = +1$ if the predicted class is correct, and $R = 0$ (or a negative value) otherwise. This function can be designed to be arbitrarily complex to reflect asymmetric costs in real-world applications.

For this framework, we employ and compare two policy-gradient algorithms: *REINFORCE* and *Advantage Actor-Critic* (A2C).

3.1 REINFORCE

REINFORCE [11] is a foundational policy-gradient method. We use it to establish a baseline due to its clarity and directness; it learns a policy (the classifier) without the added complexity of a separate value function network. The agent's policy, $\pi_\theta(A|S)$, is represented by a neural network with parameters θ. This network takes a state S and outputs a probability distribution over all possible actions (classes) using a *softmax* activation function.

The core of REINFORCE is to update the policy's parameters by taking steps in the direction of the gradient of the expected reward. The policy gradient theorem provides a way to estimate this gradient [11, 12]:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} G_t \nabla_\theta \log \pi_\theta(A_t|S_t)\right] \tag{1}$$

where $J(\theta)$ is the expected total reward (the objective function) for a policy parameterized by θ, and G_t is the return, which represents the cumulative discounted reward from timestep t onwards.

In our single-step classification (*contextual bandit* [7]) setting, this simplifies significantly[1]. For each data point, the agent takes an action A_t (i.e. predicts a class) based on the policy $\pi_\theta(A_t|S_t)$ and receives an immediate reward R_t. The update rule for the policy parameters becomes:

$$\theta \leftarrow \theta + \alpha \cdot R_t \cdot \nabla_\theta \log \pi_\theta(A_t|S_t) \tag{2}$$

Intuitively, this update 'reinforces' the action taken. If the reward R_t is positive, the update increases the log-probability of taking that action again in that state. If the reward is zero or negative, the probability is unchanged or decreased. While simple, this approach is known to suffer from high variance in its gradient estimates, as the update relies on the outcome of a single sampled action.

3.2 Advantage Actor-Critic (A2C)

To address the high variance of REINFORCE, we also implement an A2C algorithm [6, 8]. A2C introduces a second neural network, the **Critic**, which learns the state-value function $V_\phi(S)$ with parameters ϕ. The Critic's role is to estimate the expected return from a given state [2].

Instead of using the raw reward R_t to scale the gradient, A2C uses the **Advantage** function, $A(S_t, A_t)$. The advantage captures how much better an action was than the expected baseline for that state:

$$A(S_t, A_t) = R_t - V_\phi(S_t) \tag{3}$$

The original policy network is now referred to as the **Actor**. The Actor's update rule is modified to use the advantage:

$$\theta \leftarrow \theta + \alpha \cdot A(S_t, A_t) \cdot \nabla_\theta \log \pi_\theta(A_t|S_t) \tag{4}$$

By subtracting the baseline value $V_\phi(S_t)$, the advantage signal reduces the variance of the gradient updates, leading to more stable and efficient learning. The Critic itself is trained concurrently by minimizing the difference between its predictions $V_\phi(S_t)$ and the actual observed returns, typically using a mean squared error loss.

4 A Case Study: Classifying the Iris Dataset

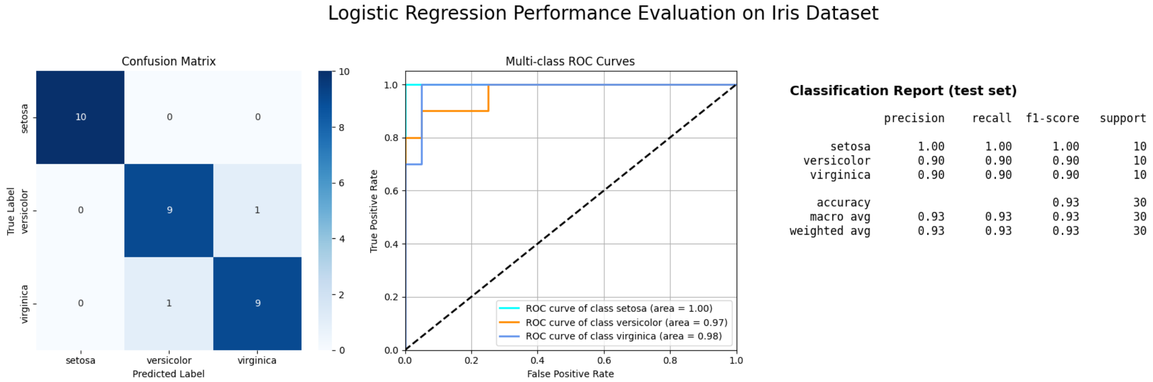
To demonstrate the RL classifier in practice, we conduct a series of experiments on the classic *Iris* dataset. This dataset contains 150 samples from three species of Iris flowers (*Setosa*, *Versicolour*, and *Virginica*), with four measured features for each sample. The data is split into a training set (80%) and a test set (20%), with features scaled to have zero mean and unit variance. Our experimental narrative is structured in four parts: a traditional benchmark followed by three progressive RL stages.

[1] As we are not using looking-ahead cumulative rewards, the discount factor can be safely ignored.

[2] A different type of Critic learns the action-value function $Q(s, a)$.

4.1 Benchmark with Logistic Regression

To ground our results, we first establish a performance benchmark using a standard discriminative classifier, Logistic Regression. The model is trained and evaluated on the exact same data splits as the RL agents for a fair comparison. The results, shown in Fig.1, are excellent. The model achieves a test accuracy of 93% with near-perfect precision and recall for all classes, and it trains in a fraction of a second. This sets a high standard for our RL agents to meet on a simple accuracy-based objective.



Classification Report (test set)

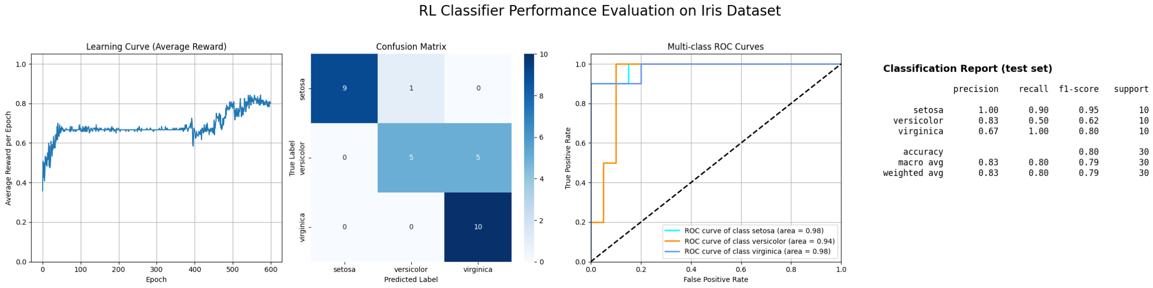
```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       0.90      0.90      0.90        10
   virginica       0.90      0.90      0.90        10

    accuracy                           0.93        30
   macro avg       0.93      0.93      0.93        30
weighted avg       0.93      0.93      0.93        30
```

Figure 1: Benchmark results: performance of the Logistic Regression classifier. The model achieves 93% accuracy, providing a strong baseline for comparison.

4.2 Stage 1: RL Classification with REINFORCE

Next, we establish an RL baseline using the vanilla REINFORCE algorithm. The objective is to maximize classification accuracy, so we use a simple reward signal: a reward of +1 for a correct classification and 0 otherwise. The agent is trained for 600 epochs.

The results are shown in Fig.2. The learning curve shows that the agent successfully learns, with the average reward increasing from a random-guess baseline to a stable value around 0.8. The final test accuracy is 80%. While promising, this is significantly lower than the logistic regression benchmark. The learning curve is noisy, and the confusion matrix reveals that the agent struggles with the boundary between the *versicolor* and *virginica* classes. This confirms that while the basic concept is viable, the high variance of gradient estimation inherent to the REINFORCE algorithm limits its performance.



Classification Report (test set)

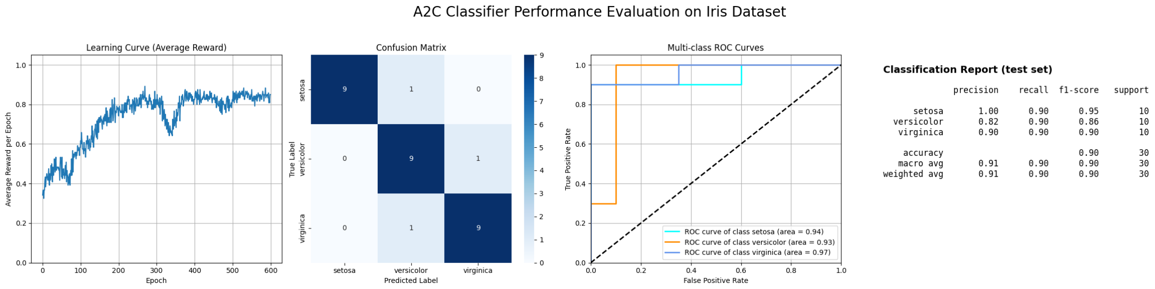
```
              precision    recall  f1-score   support

      setosa       1.00      0.90      0.95        10
  versicolor       0.83      0.50      0.62        10
   virginica       0.67      1.00      0.80        10

    accuracy                           0.80        30
   macro avg       0.83      0.80      0.79        30
weighted avg       0.83      0.80      0.79        30
```

Figure 2: Stage 1 results: performance of the REINFORCE agent. The agent learns but struggles with the harder class boundaries due to high variance, achieving 80% accuracy.

4.3 Stage 2: Improving Stability with Actor-Critic (A2C)

To address the high variance observed in Stage 1, we upgrade the agent to an A2C architecture. We keep all other settings, including the simple +1/0 reward function and 600 training epochs, identical for a fair comparison.

The results, presented in Fig.3, show a definitive improvement. The learning curve is visibly more stable than in the REINFORCE experiment. The final test accuracy increases to 90%, approaching the performance of the logistic regression benchmark. This clearly demonstrates that adding a Critic to provide a baseline for the reward signal successfully addresses the primary weakness of the vanilla policy-gradient method, leading to superior and more reliable performance on a standard accuracy objective.



	setosa	versicolor	virginica
setosa	9	1	0
versicolor	0	9	1
virginica	0	1	9

Classification Report (test set)

```
              precision    recall  f1-score   support

      setosa       1.00      0.90      0.95        10
  versicolor       0.82      0.90      0.86        10
   virginica       0.90      0.90      0.90        10

    accuracy                           0.90        30
   macro avg       0.91      0.90      0.90        30
weighted avg       0.91      0.90      0.90        30
```

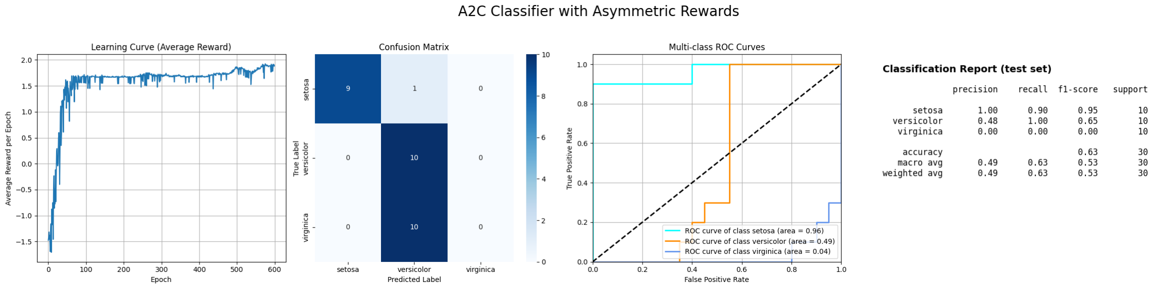
Figure 3: Stage 2 results: performance of the A2C agent. The learning is more stable, and the agent correctly distinguishes the classes, achieving 90% accuracy.

4.4 Stage 3: Asymmetric Cost Optimization with A2C

Having established a robust agent that performs comparably to a traditional classifier, we now demonstrate the core advantage of the RL approach. We design a new reward function for a hypothetical scenario where correctly identifying *versicolor* is critical, and failing to do so incurs a large penalty. The new reward structure is:

- Correctly identifying *versicolor*: Reward = +5
- Correctly identifying any other class: Reward = +1
- Misclassifying a true *versicolor* (False Negative): Reward = -10
- Any other misclassification: Reward = 0

The A2C agent is now trained using this asymmetric reward function. The results in Fig.4 are exactly as hypothesized. The agent's policy has fundamentally changed. The confusion matrix shows that the agent achieves a perfect recall of 1.00 for the *versicolor* class, meaning it makes zero of the most costly errors. To achieve this, it has learned a cautious strategy, misclassifying all *virginica* samples as *versicolor* because that error has no penalty. The overall accuracy drops to 63%, but the agent has perfectly optimized for the complex, asymmetric objective we defined. This result powerfully illustrates the flexibility of the RL framework to solve problems beyond simple accuracy maximization.


A2C Classifier with Asymmetric Rewards

Learning Curve (Average Reward)

Confusion Matrix

Multi-class ROC Curves

Classification Report (test set)

```
              precision    recall  f1-score   support

      setosa       1.00      0.90      0.95        10
  versicolor       0.48      1.00      0.65        10
   virginica       0.00      0.00      0.00        10

    accuracy                           0.63        30
   macro avg       0.49      0.63      0.53        30
weighted avg       0.49      0.63      0.53        30
```

Figure 4: Stage 3 results: performance of the A2C agent with an asymmetric reward function. The agent achieves perfect recall on the critical *versicolor* class, demonstrating its ability to learn a cost-sensitive policy at the expense of overall accuracy.

5 Discussion

Our progressive case study on the Iris dataset highlights several important lessons. The journey from a simple REINFORCE agent to a cost-sensitive A2C agent illustrates a practical workflow for developing and refining an RL-based model for a classification task. The initial experiment confirmed that even a basic policy-gradient method can solve a standard classification task, but its performance is degraded by high variance, a well-known limitation of the REINFORCE algorithm. The introduction of a critic in Stage 2 stabilized the learning process and improved overall accuracy, demonstrating the value of the Actor-Critic algorithm and achieving performance comparable to a strong Logistic Regression baseline.

The most significant finding comes from Stage 3. By designing an asymmetric reward function, we successfully guided the agent to learn a policy that a traditional classifier, optimizing for accuracy or cross-entropy, would not find. The agent learned to prioritize the perfect recall of the *versicolor* class above all else, willingly sacrificing overall accuracy to avoid the specific, heavily penalized error of a false negative on that class. This is not a model failure but a success in optimizing for a complex, real-world objective. It demonstrates that the RL classifier's behavior is directly and precisely controllable through the design of its reward signal.

This flexibility has profound implications for practical applications. In domains such as medical diagnosis, where missing a disease (a false negative) is far more catastrophic than a false alarm (a false positive), or in spam detection, where misclassifying an important email is worse than letting a spam email through, this method allows for the direct encoding of such domain-specific costs into the model's objective function.

However, this approach is not without its drawbacks. RL algorithms are generally more time-consuming, complex to implement and tune than standard supervised learning methods, often involving more hyperparameters. Furthermore, on-policy methods like the ones used here can be sample-inefficient on larger datasets. Future work could explore the application of more advanced, off-policy algorithms such as *DQN* [10] or *Rainbow* [4] to improve sample efficiency, or test this framework on more complex, high-dimensional datasets where the benefits of custom reward shaping could be even more pronounced.

6 Conclusion

In this work, we empirically demonstrated that a classification task can be effectively reframed and solved as a reinforcement learning problem. By treating the classifier as an agent that chooses a class label as its action, we can leverage policy-gradient methods to optimize its behavior by maximizing a reward signal. Our progressive case study illustrated a clear path from a simple REINFORCE baseline to a more stable and powerful A2C agent that matched the performance of a traditional Logistic Regression model. Crucially, we showed that by designing a custom, asymmetric reward function, the A2C agent could learn a specialized, cost-sensitive policy that prioritized avoiding specific critical errors over maximizing overall accuracy. This work serves as a practical guide and proof-of-concept for researchers and practitioners, showcasing the RL classifier as a flexible and powerful tool for classification tasks, particularly in scenarios that demand objectives more complex than what standard loss functions can provide.

Code Availability

The code used in this work is available at: `https://github.com/YongchaoHuang/rl_classification`

References

[1] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful Contextual Bandits. In *Proceedings of The 27th Conference on Learning Theory*, pages 1109–1134. PMLR, May 2014. ISSN: 1938-7228.

[2] Alberto Bietti, Alekh Agarwal, and John Langford. A contextual bandit bake-off. *J. Mach. Learn. Res.*, 22(1):133:5928–133:5976, January 2021.

[3] Evan Greensmith, Peter Bartlett, and Jonathan Baxter. Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

[4] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: combining improvements in deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, pages 3215–3222, New Orleans, Louisiana, USA, February 2018. AAAI Press.

[5] Yongchao Huang. Classification via score-based generative modelling, July 2022. arXiv:2207.11091 [cs].

[6] Vijay Konda and John Tsitsiklis. Actor-Critic Algorithms. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[7] John Langford and Tong Zhang. The Epoch-Greedy algorithm for contextual multi-armed bandits. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'07, pages 817–824, Red Hook, NY, USA, December 2007. Curran Associates Inc.

[8] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning, June 2016. arXiv:1602.01783 [cs].

[9] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, volume 2 of *NIPS'14*, pages 2204–2212, Cambridge, MA, USA, December 2014. MIT Press.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. Publisher: Nature Publishing Group.

[11] Ronald J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.*, 8(3-4):229–256, May 1992.

[12] Junzi Zhang, Jongho Kim, Brendan O'Donoghue, and Stephen Boyd. Sample Efficient Reinforcement Learning with REINFORCE, December 2020. arXiv:2010.11364 [cs].

[13] Xinmin Zhang, Saite Fan, and Zhihuan Song. Reinforcement learning-based cost-sensitive classifier for imbalanced fault classification. *Science China Information Sciences*, 66(11):212201, October 2023.