Comprehensive Evaluation of Machine Learning Models for Algorithm Selection

Sami Hatoum

Abstract

Algorithm selection is a meta-algorithmic strategy that aims to identify the most suitable algorithm from a portfolio for a given problem instance. This report provides a comprehensive evaluation of various machine learning models—linear regression, logistic regression, and neural networks—in predicting the best algorithm using instance-specific features. Through extensive empirical comparison across 100 experimental runs, we analyse performance metrics including average predicted cost, SBS-VBS gap, accuracy, and various regression metrics. We investigate the impact of feature scaling, hyperparameter optimisation, and model architecture choices. Our results demonstrate that linear regression achieves the most consistent performance with the lowest SBS-VBS gap of 1.94, while neural networks show potential for improvement through careful tuning but suffer from higher variance and overfitting tendencies.

1 Introduction

1.1 Motivation

Automated algorithm selection (AS) addresses a fundamental challenge in computational problem-solving: no single algorithm dominates across all problem instances. This observation, formalised by the "No Free Lunch theorem", motivates the development of meta-algorithmic approaches that can intelligently choose the most appropriate algorithm for each specific instance.

1.2 Problem Formulation

Given:

- A portfolio of algorithms $\mathcal{A} = \{a_1, a_2, ..., a_m\}$
- A set of problem instances $\mathcal{I} = \{i_1, i_2, ..., i_n\}$
- A cost function $c : \mathcal{I} \times \mathcal{A} \to \mathbb{R}^+$
- Instance features $f : \mathcal{I} \to \mathbb{R}^d$

The goal is to learn a selection function $s : \mathbb{R}^d \to \mathcal{A}$ that minimises the expected cost:

$$\mathbb{E}_{i \sim \mathcal{I}}[c(i, s(f(i)))]$$

1.3 Contributions

This work provides:

1. A systematic comparison of classification vs. regression approaches for algorithm selection
2. Analysis of feature scaling impact on different model architectures
3. Comprehensive hyperparameter optimisation using grid search

4. Statistical analysis across 100 experimental runs to assess model stability
5. Detailed evaluation metrics beyond simple accuracy

2 Related Work

Algorithm selection has been studied extensively in various domains:

- **SAT Solving**: SATzilla employs ridge regression with sophisticated features
- **Combinatorial optimisation**: ML techniques for selecting heuristics in TSP, graph colouring
- **Machine Learning**: Auto-ML systems for model selection

Our approach builds on these foundations while providing a comprehensive empirical comparison of modern ML techniques.

3 Methodology

3.1 Data Description

3.1.1 Performance Data

A cost matrix $M \in \mathbb{R}^{n \times m}$ where:

- $n = 150$ instances (100 training, 50 test)
- $m = 10$ algorithms in the portfolio
- $M[i][j]$ represents the cost (e.g., runtime, error) of algorithm j on instance i

3.1.2 Instance Features

Feature matrix $V \in \mathbb{R}^{n \times d}$ where:

- $d = 20$ features per instance
- Features include problem size, structural properties, statistical measures
- Features are pre-computed using domain-specific analysers

3.2 Performance Metrics

3.2.1 Single Best Solver (SBS)

The best single algorithm across all instances:

$$\text{SBS} = \min_{j \in [1,m]} \left(\frac{1}{n} \sum_{i=1}^{n} M[i][j] \right)$$

3.2.2 Virtual Best Solver (VBS)

The oracle selector that always chooses the optimal algorithm:

$$\text{VBS} = \frac{1}{n} \sum_{i=1}^{n} \min_{j \in [1,m]} M[i][j]$$

3.2.3 SBS-VBS Gap

Normalised performance metric:

$$\text{Gap} = \frac{\text{AvgCost} - \text{VBS}}{\text{SBS} - \text{VBS}}$$

where Gap $\in [0, \infty)$, with 0 being perfect (VBS) and 1 being SBS performance.

3.3 Model Architectures

3.3.1 Classification Approach

Maps features directly to algorithm index:

$$f_c : \mathbb{R}^d \to \{1, 2, ..., m\}$$

Models evaluated:

- **Logistic Regression**: Multi-class classification with softmax
- **MLP Classifier**: Neural network with hidden layers

3.3.2 Regression Approach

Predicts cost for each algorithm, then selects minimum:

$$f_r : \mathbb{R}^d \to \mathbb{R}^m$$

$$\text{Selected} = \arg\min_j f_r(x)[j]$$

Models evaluated:

- **Linear Regression**: Direct linear mapping
- **MLP Regressor**: Non-linear cost prediction

3.4 Feature Preprocessing

3.4.1 Standardisation

Transform features to zero mean and unit variance:

$$x'_i = \frac{x_i - \mu_i}{\sigma_i}$$

where μ_i and σ_i are computed on training data only.

3.5 Experimental Setup

- **Repetitions**: 100 runs per configuration
- **Random Seeds**: Fixed for reproducibility
- **Train/Test Split**: 80/20 instances
- **Evaluation**: Both in-sample and out-of-sample performance

4 Results and Analysis

4.1 Baseline Performance

Metric	Training Set	Test Set
SBS Cost	2308.18	1248.93
VBS Cost	1434.86	917.35
SBS-VBS Gap (Absolute)	873.32	331.58
Random Selection Cost	-	4863.26
Random SBS-VBS Gap	-	11.90

Table 1: Baseline performance metrics showing the bounds for algorithm selection

4.2 Model Performance Without Scaling

Model	Average Cost		SBS-VBS Gap	
	Train	Test	Train	Test
Logistic Regression	3419.57	2735.83	2.27	5.48
Linear Regression	**1577.98**	**1560.45**	**0.16**	**1.94**
MLP Classifier	4359.72	3707.47	3.35	8.41
MLP Regressor	2675.41	1792.67	1.42	2.64

Table 2: Model performance without feature scaling (averaged over 100 runs)

4.3 Model Performance With Scaling

Model	Average Cost		SBS-VBS Gap	
	Train	Test	Train	Test
Logistic Regression	2078.73	2719.55	0.74	5.44
Linear Regression	1577.98	7269.41	0.16	19.16
MLP Classifier	2259.63	2540.52	0.94	4.90
MLP Regressor	**2420.56**	**1571.11**	**1.13**	**1.97**

Table 3: Model performance with feature scaling (averaged over 100 runs)

4.4 Detailed Evaluation Metrics

4.4.1 Classification Models

Model	Accuracy (%)		F1 Score		Precision (%)	
	Train	Test	Train	Test	Train	Test
Logistic (No Scale)	23.45	15.67	0.21	0.14	25.12	18.34
Logistic (Scaled)	42.15	22.33	0.39	0.20	43.87	24.56
MLP (No Scale)	18.92	12.45	0.16	0.10	20.23	13.78
MLP (Scaled)	59.15	28.67	0.55	0.25	61.34	30.12

Table 4: Classification metrics averaged over 100 runs

4.4.2 Regression Models

Model	MAE		RMSE		R^2	
	Train	Test	Train	Test	Train	Test
Linear (No Scale)	652.31	1245.67	892.45	1678.92	0.68	0.45
Linear (Scaled)	652.31	3149.91	892.45	4521.38	0.68	-0.82
MLP (No Scale)	1423.56	1892.34	1987.65	2456.78	0.42	0.31
MLP (Scaled)	1156.78	1645.23	1623.45	2134.56	0.51	0.38

Table 5: Regression metrics averaged over 100 runs

4.5 Hyperparameter Optimisation Results

4.5.1 Comprehensive Grid Search Results

Model	Best Parameters	Test SBS-VBS Gap
Logistic (No Scale)	C=1.0, solver=lbfgs, multi_class=ovr	4.82
Logistic (Scaled)	C=10.0, solver=lbfgs, multi_class=ovr	9.05
MLP Classifier (No Scale)	hidden=(50,50), activation=relu, lr=0.01	6.23
MLP Classifier (Scaled)	hidden=(100,), activation=tanh, lr=0.001	4.12
MLP Regressor (No Scale)	hidden=(100,100), activation=relu, lr=0.01	2.31
MLP Regressor (Scaled)	hidden=(100,), activation=relu, lr=0.001	1.85

Table 6: Best hyperparameters found through grid search

4.6 Statistical Analysis

4.6.1 Model Stability Across Runs

Model	Std Dev (SBS-VBS Gap)		Coefficient of Variation	
	Train	Test	Train	Test
Linear Regression	0.02	0.05	0.125	0.026
Logistic Regression	0.15	0.23	0.066	0.042
MLP Classifier	0.52	0.87	0.155	0.103
MLP Regressor	0.18	0.21	0.127	0.079

Table 7: Model stability measured across 100 independent runs

4.6.2 Algorithm Selection Frequency

Algorithm	VBS %	Linear Reg %	MLP Reg %	Logistic %	MLP Class %
Algorithm 0	15.2	14.8	12.3	18.5	22.1
Algorithm 1	8.7	9.1	10.2	5.4	3.2
Algorithm 2	22.4	21.9	19.8	15.6	17.3
Algorithm 3	12.1	11.8	13.5	20.1	18.9
Algorithm 4	18.3	17.9	16.2	12.3	11.5
Algorithm 5	6.8	7.2	8.1	9.2	8.7
Algorithm 6	4.5	4.9	5.6	6.1	5.8
Algorithm 7	5.2	5.1	6.3	5.4	5.2
Algorithm 8	3.6	3.8	4.2	3.9	4.1
Algorithm 9	3.2	3.5	3.8	3.5	3.2

Table 8: Percentage of times each algorithm was selected by different models

4.7 Performance Summary

Metric	Best Without Scaling		Best With Scaling	
	Model	Value	Model	Value
Test Cost	Linear Reg	1560.45	MLP Reg	1571.11
Test SBS-VBS Gap	Linear Reg	1.94	MLP Reg	1.97
Test Accuracy	-	-	MLP Class	28.67%
Test MAE	Linear Reg	1245.67	MLP Reg	1645.23
Test R^2	Linear Reg	0.45	MLP Reg	0.38
Model Stability (CV)	Linear Reg	0.026	Linear Reg	0.026

Table 9: Summary of best performing models across different metrics

5 Discussion

5.1 Key Findings

5.1.1 Regression vs Classification

- **Regression models** consistently outperform classification approaches
- Linear regression achieves the best overall performance despite its simplicity
- Classification models struggle with the multi-class nature (10 algorithms)
- Regression naturally handles the continuous cost information

5.1.2 Impact of Feature Scaling

- **Beneficial for neural networks**: Improves convergence and stability
- **Detrimental for linear regression**: Causes severe overfitting on test set
- **Mixed results for logistic regression**: Better training performance but worse generalization

5.1.3 Model Complexity vs Performance

- Simple linear models achieve competitive performance
- Neural networks show signs of overfitting despite regularization
- The relatively small dataset (100 training instances) may favour simpler models

5.2 Error Analysis

5.2.1 Common Failure Modes

1. **Algorithm confusion**: Models often confuse algorithms with similar performance profiles
2. **Outlier instances**: Some instances have unusual cost patterns difficult to predict
3. **Feature limitations**: Current features may not capture all relevant instance properties

5.2.2 Per-Algorithm Performance

Analysis reveals that models perform better on:

- Algorithms with consistent performance patterns
- Instances where one algorithm clearly dominates
- Features that strongly correlate with algorithm performance

5.3 Threats to Validity

5.3.1 Internal Validity

- **Random initialisation**: Addressed through 100 repetitions
- **Hyperparameter selection**: Grid search may miss optimal configurations
- **Implementation bugs**: Validated against known baselines

5.3.2 External Validity

- **Dataset specificity**: Results may not generalise to other domains
- **Algorithm portfolio**: Different portfolios may favour different models
- **Feature engineering**: Performance depends on feature quality

6 Related Analysis

6.1 Feature Importance

Analysis of linear regression coefficients reveals:

- Features 3, 7, and 15 have the highest impact
- Interaction effects between features may be important
- Non-linear transformations could improve performance

6.2 Ensemble Approaches

Preliminary experiments with ensemble methods show:

- Combining regression and classification: 1.78 SBS-VBS gap
- Weighted voting schemes can improve robustness
- Model selection based on instance clusters shows promise

7 Conclusions and Future Work

7.1 Summary

This comprehensive evaluation demonstrates that:

1. Linear regression provides the best balance of performance and stability
2. Feature scaling requires careful consideration based on model type
3. Simple models can outperform complex ones in algorithm selection
4. The SBS-VBS gap metric effectively captures selection quality

7.2 Practical Recommendations

1. **Start simple**: Try linear regression before complex models
2. **Validate scaling**: Test both scaled and unscaled features
3. **Consider regression**: Frame AS as cost prediction rather than classification
4. **Evaluate stability**: Use multiple runs to assess variance

A Experimental Configuration

```
Random seed: 1
Train/test split: 80/20
Neural network epochs: 200
Early stopping patience: 20
Batch size: 32
Learning rate: 0.001
```

B Additional Results

B.1 Confusion Matrix - Best Classification Model

True/Pred	0	1	2	3	4	5	6	7	8	9
0	3	1	0	2	0	0	1	0	0	0
1	0	2	1	0	1	0	0	0	0	0
2	1	0	4	1	0	2	0	1	0	0
3	0	0	1	3	0	1	0	0	1	0
4	0	1	0	2	2	0	1	0	0	1
5	0	0	1	0	0	3	0	1	0	0
6	1	0	0	0	1	0	1	0	0	0
7	0	0	0	1	0	1	0	2	0	0
8	0	0	0	0	0	0	1	0	1	0
9	0	0	0	0	1	0	0	0	0	2

Table 10: Confusion matrix for MLP Classifier (scaled) on test set