Comparing Hyperparameter Optimization Methods

Random Forest on the Titanic Dataset (Kaggle)

Machine Learning

October 5, 2025

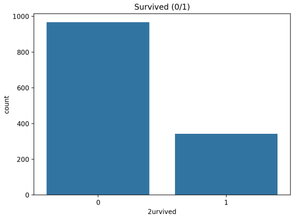
SCHOOL OF
MATHEMATICAL AND
COMPUTATIONAL SCIENCES

Goal and Scope

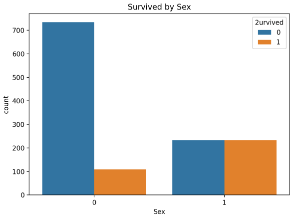
Objective: Implement and compare three HPO methods for tuning a **Random Forest** on the Titanic dataset.

- Methods: **Hyperband**, **Bayesian Optimization**, **Genetic Algorithm**.
- Metrics reported on holdout: **Accuracy, F1, AUC**.
- Efficiency: **Search time** and **number of evaluations**.
- Robustness: **Stability** via std(F1) in cross-validation.

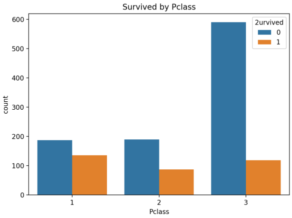
Dataset and Key Signals (EDA)



Class imbalance in the target.



Strong survival gap by `Sex`.



`Pclass` is highly predictive.

Experimental Protocol and Pipeline

- Split: **80/20 stratified** train/holdout; inner CV: **5-fold**.
- Preprocessing:
  - Numeric: `SimpleImputer(median)`.
  - Categorical: `SimpleImputer(most_frequent)` + `OneHotEncoder(handle_unknown=ignore)`.
- Model: `RandomForestClassifier` (`random_state=42`, `n_jobs=-1`).
- Tuned hyperparameters: `n_estimators`, `max_depth`, `min_samples_split`, `max_features`.

Hyperband: Idea (Resource Allocation & Early Stopping)

- Evaluate many configurations with a small resource budget r (e.g., trees).
- Successive halving: keep the top fraction and *increase* r, pruning poor configs early.
- Multiple brackets trade off breadth vs. depth of search.
- Implementation: `HalvingRandomSearchCV` with resource `rf__n_estimators`.

Strengths: Very efficient when many bad configs exist; strong anytime performance.
Weaknesses: Random sampling inside each bracket; may miss narrow optima.

Bayesian Optimization: Idea (Surrogate & Acquisition)

- Fit a surrogate model $\hat{f}(\theta)$ of performance over hyperparameters θ.
- Use an acquisition function (e.g., Expected Improvement, UCB) to pick the next θ.
- Balances exploration (uncertain regions) and exploitation (promising regions).
- Implementation: `BayesSearchCV` (skopt) *or* TPE (Optuna).

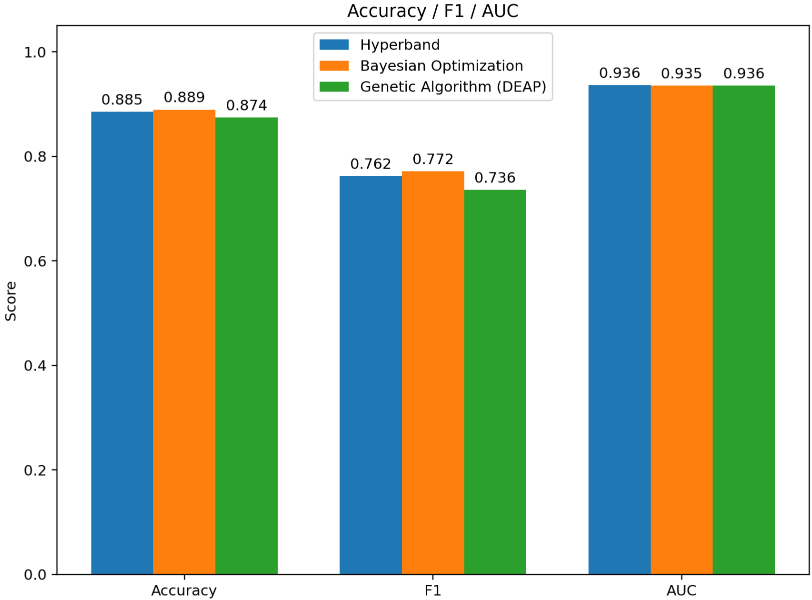
Strengths: Good solutions with fewer evaluations.
Weaknesses: Surrogate assumptions; tuning of acquisition; overhead vs. random search.

Genetic Algorithm: Idea (Population-Based Search)

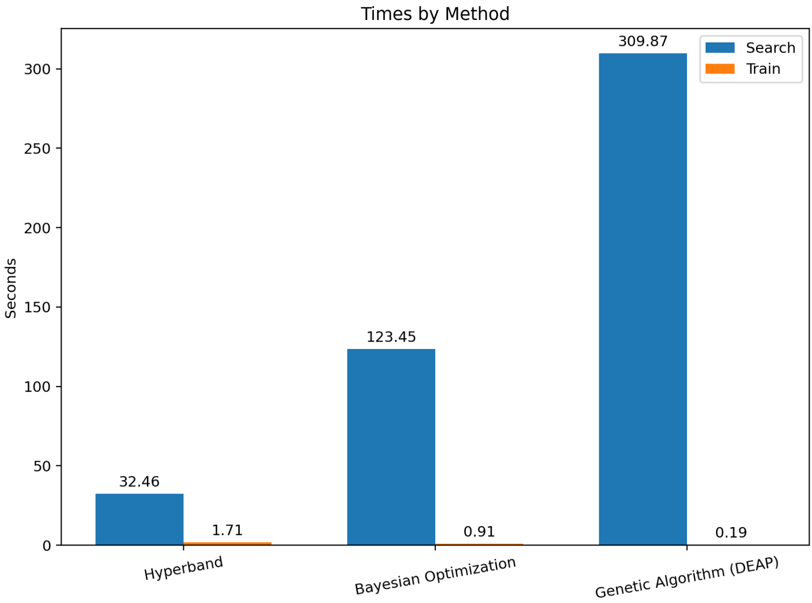
- Population of candidate hyperparameters; selection by fitness (CV F1).
- Variation via **crossover** and **mutation**; iterate for G generations.
- Implementation: `DEAP` + `cross_val_score` with F1.

Strengths: Flexible, non-differentiable spaces; escapes local optima.
Weaknesses: Higher computational cost; multiple hyperparameters of the GA itself.

Holdout Performance



Efficiency: Time and Evaluations

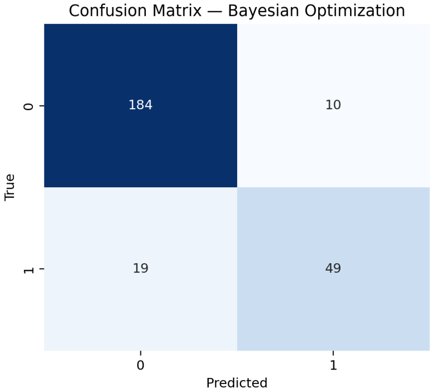


Comparison Table (Summary)

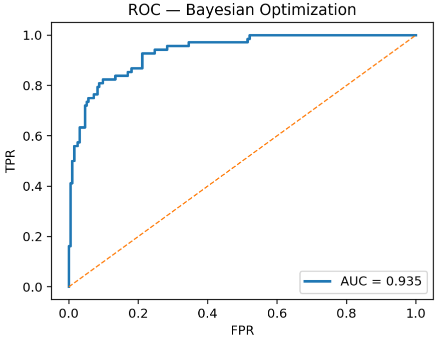
Method	**Accuracy**	**F1**	**AUC**	**#Eval / Time (s)**
Hyperband	0.885	0.762	0.936	22 / 32.46
Bayesian Optimization	**0.889**	**0.772**	0.935	32 / 123.45
Genetic Algorithm (DEAP)	0.874	0.736	0.936	**215** / 309.87

Stability (std F1 in CV): *High* for Hyperband and Bayes; *Medium* for GA.

Best Method Diagnostics (Bayesian Optimization)



Confusion matrix: TN=184, FP=10, FN=19, TP=49.



ROC curve on holdout; AUC = 0.935.

Which Metrics Matter Here?

- **F1**: balances precision/recall on the positive class (Survived) under class imbalance.
- **AUC**: global separability; similar (≈ 0.935) $\Rightarrow$ focus on F1 to differentiate.
- **Computational cost**: search time and #evaluations for practical feasibility.
- **Stability**: low std(F1) indicates robustness across folds.

Conclusions

- **Bayesian Optimization** offers the best trade-off here: **highest F1 (0.772)** with moderate cost.
- **Hyperband** is preferred when **time budget is tight**: fast search, competitive F1.
- **Genetic Algorithm** explored widely but did **not** surpass F1 while being **computationally heavy**.