Machine Learning Performance Improvement Cheat Sheet

32 Tips, Tricks and Hacks To Make Better Predictions

Jason Brownlee

MACHINE LEARNING MASTERY

Machine Learning Performance Improvement Cheat Sheet

Edition: v1.1

Contents

Before We Get Started...

The most valuable part of machine learning is predictive modeling. This is the development of models that are trained on historical data and make predictions on new data. And the number one question when it comes to predictive modeling is: *How can I get better results?*

This cheat sheet contains my best advice distilled from years of my own application and studying top machine learning practitioners and competition winners. With this guide, you will not only get unstuck and lift performance, you might even achieve world-class results on your prediction problems. Let's dive in.

Overview

This cheat sheet is designed to give you ideas to lift performance on your machine learning problem. All it takes is one good idea to get a breakthrough. One good idea when your stuck on a problem is worth a ton of gold, and this guide contains 32 ideas of things to try! Find that one idea, make progress, then come back and find another.

I have divided the guide into 4 sub-topics:

1. Improve Performance With Data.
2. Improve Performance With Algorithms.
3. Improve Performance With Tuning.
4. Improve Performance With Ensembles.

The gains often get smaller the further you go down the list. For example, a new framing of your problem or more data is often going to give you more payoff than tuning the parameters of your best performing algorithm. Not always, but in general.

Improve Performance With Data

You can get big wins with changes to your training data and problem definition. Perhaps even the biggest wins.

Strategy: Create new and different perspectives on your data in order to best expose the structure of the underlying problem to the learning algorithms.

Tactics:

- **Get More Data**. *Can you get more or better quality data?* Modern nonlinear machine learning techniques like deep learning continue to improve in performance with more data.
- **Invent More Data**. *If you can't get more data, can you generate new data?* Perhaps you can augment or permute existing data or use a probabilistic model to generate new data.
- **Clean Your Data**. *Can you improve the signal in your data?* Perhaps there are missing or corrupt observations that can be fixed or removed, or outlier values outside of reasonable ranges that can be fixed or removed in order to lift the quality of your data.
- **Resample Data**. *Can you resample data to change the size or distribution?* Perhaps you can use a much smaller sample of data for your experiments to speed things up or over-sample or under-sample observations of a specific type to better represent them in your dataset.
- **Reframe Your Problem**: *Can you change the type of prediction problem you are solving?* Reframe your data as a regression, binary or multiclass classification, time series, anomaly detection, rating, recommender, etc. type problem.
- **Rescale Your Data**. *Can you rescale numeric input variables?* Normalization and standardization of input data can result in a lift in performance on algorithms that use weighted inputs or distance measures.
- **Transform Your Data**. *Can you reshape your data distribution?* Making input data more Gaussian or passing it through an exponential function may better expose features in the data to a learning algorithm.
- **Project Your Data**: *Can you project your data into a lower dimensional space?* You can use an unsupervised clustering or projection method to create an entirely new compressed representation of your dataset.

- **Feature Selection**. *Are all input variables equally important?* Use feature selection and feature importance methods to create new views of your data to explore with modeling algorithms.
- **Feature Engineering**. *Can you create and add new data features?* Perhaps there are attributes that can be decomposed into multiple new values (like categories, dates or strings) or attributes that can be aggregated to signify an event (like a count, binary flag or statistical summary).

Outcome: You should now have a suite of new views and versions of your dataset.

Next: You can evaluate the value of each with predictive modeling algorithms.

Improve Performance With Algorithms

Machine learning is all about algorithms.

Strategy: Identify the algorithms and data representations that perform above a baseline of performance and better than average. Remain skeptical of results and design experiments that make it hard to fool yourself.

Tactics:

- **Resampling Method**. *What resampling method is used to estimate skill on new data?* Use a method and configuration that makes the best use of available data. The k-fold cross-validation method with a hold out validation dataset might be a best practice.
- **Evaluation Metric**. *What metric is used to evaluate the skill of predictions?* Use a metric that best captures the requirements of the problem and the domain. It probably isn't classification accuracy.
- **Baseline Performance**. *What is the baseline performance for comparing algorithms?* Use a random algorithm or a zero rule algorithm (predict mean or mode) to establish a baseline by which to rank all evaluated algorithms.
- **Spot-Check Linear Algorithms**. *What linear algorithms work well?* Linear methods are often more biased, are easy to understand and are fast to train. They are preferred if you can achieve good results. Evaluate a diverse suite of linear methods.
- **Spot-Check Nonlinear Algorithms**. *What nonlinear algorithms work well?* Nonlinear algorithms often require more data, have greater complexity but can achieve better performance. Evaluate a diverse suite of nonlinear methods.
- **Steal from Literature**. *What algorithms are reported in the literature to work well on your problem*? Perhaps you can get ideas of algorithm types or extensions of classical methods to explore on your problem.
- **Standard Configurations**. *What are the standard configurations for the algorithms being evaluated*? Each algorithm needs an opportunity to do well on your problem. This does not mean tune the parameters (yet) but it does mean to investigate how to configure each algorithm well and give it a fighting chance in the algorithm bake-off.

Outcome: You should now have a shortlist of well-performing algorithms and data representations.

Next: The next step is to improve performance with algorithm tuning.

Improve Performance With Tuning

Algorithm tuning might be where you spend the most of your time. It can be very time-consuming. You can often unearth one or two well-performing algorithms quickly from spot-checking. Getting the most from those algorithms can take, days, weeks or months.

Strategy: Get the most out of well-performing machine learning algorithms.

Tactics:

- **Diagnostics**. *What do diagnostics tell you about your algorithm?* Perhaps you can review learning curves to understand whether the method is over or underfitting the problem, and then correct. Different algorithms may offer different visualizations and diagnostics. Review what the algorithm is predicting right and wrong.
- **Try Intuition**. *What does your gut tell you?* If you fiddle with parameters for long enough and the feedback cycle is short, you can develop an intuition for how to configure an algorithm on a problem. Try this out and see if you can come up with new parameter configurations to try on your larger test harness.
- **Steal from Literature**. *What parameters or parameter ranges are used in the literature?* Evaluating the performance of standard parameters is a great place to start any tuning activity.
- **Random Search**. *What parameters can use random search?* Perhaps you can use random search of algorithm hyperparameters to expose configurations that you would never think to try.
- **Grid Search**. *What parameters can use grid search?* Perhaps there are grids of standard hyperparameter values that you can enumerate to find good configurations, then repeat the process with finer and finer grids.
- **Optimize**. *What parameters can you optimize?* Perhaps there are parameters like structure or learning rate than can be tuned using a direct search procedure (like pattern search) or stochastic optimization (like a genetic algorithm).
- **Alternate Implementations**. *What other implementations of the algorithm are available?* Perhaps an alternate implementation of the method can achieve better results on the same data. Each algorithm has a myriad of micro-decisions that must be made by the algorithm implementor. Some of these decisions may affect skill on your problem.

- **Algorithm Extensions**. *What are common extensions to the algorithm?* Perhaps you can lift performance by evaluating common or standard extensions to the method. This may require implementation work.
- **Algorithm Customizations**. *What customizations can be made to the algorithm for your specific case?* Perhaps there are modifications that you can make to the algorithm for your data, from loss function, internal optimization methods to algorithm specific decisions.
- **Contact Experts**. *What do algorithm experts recommend in your case?* Write a short email summarizing your prediction problem and what you have tried to one or more expert academics on the algorithm. This may reveal leading edge work or academic work previously unknown to you with new or fresh ideas.

Outcome: You should now have a shortlist of highly tuned algorithms on your machine learning problem, maybe even just one.

Next: One or more models could be finalized at this point and used to make predictions or put into production. Further lifts in performance can be gained by combining the predictions from multiple models.

Improve Performance With Ensembles

You can combine the predictions from multiple models. After algorithm tuning, this is the next big area for improvement. In fact, you can often get good performance from combining the predictions from multiple *good enough* models rather than from multiple highly tuned (and fragile) models.

Strategy: Combine the predictions of multiple well-performing models.

Tactics:

- **Blend Model Predictions**. Can you combine the predictions from multiple models directly? Perhaps you could use the same or different algorithms to make multiple models. Take the mean or mode from the predictions of multiple well-performing models.
- **Blend Data Representations**. Can you combine predictions from models trained on different data representations? You may have many different projections of your problem which can be used to train well-performing algorithms, whose predictions can then be combined.
- **Blend Data Samples**. Can you combine models trained on different views of your data? Perhaps you can create multiple subsamples of your training data and train a well-performing algorithm, then combine predictions. This is called bootstrap aggregation or bagging and works best when the predictions from each model are skillful but in different ways (uncorrelated).
- **Correct Predictions**. Can you correct the predictions of well-performing models? Perhaps you can explicitly correct predictions or use a method like boosting to learn how to correct prediction errors.
- **Learn to Combine**. Can you use a new model to learn how to best combine the predictions from multiple well-performing models? This is called stacked generalization or stacking and often works well when the submodels are skillful but in different ways and the aggregator model is a simple linear weighting of the predictions. This process can be repeated multiple layers deep.

Outcome: You should have one or more ensembles of well-performing models that outperform any single model.

Next: One or more ensembles could be finalized at this point and used to make predictions or put into production.

Final Word Before You Go

This cheat sheet is jam packed full of ideas to try to improve performance on your problem.

How To Get Started

You do not need to do everything. You just need one good idea to get a lift in performance. Here's how to handle the overwhelm:

1. Pick one group
   (a) Data.
   (b) Algorithms.
   (c) Tuning.
   (d) Ensembles.
2. Pick one method from the group.
3. Pick one thing to try of the chosen method.
4. Compare the results, keep if there was an improvement.
5. Repeat.

Share Your Results

Did you find this guide useful? Did you get that one idea or method that made a difference?

I would love to hear about it. Send me an email at `Jason@MachineLearningMastery.com`.