Model validation and comparison

Source: `vignettes/model-comparison.Rmd`

The `mbg` package allows for easy customization of geostatistical models: for example, it is simple to test models with varying covariate sets, approaches for relating covariates to the outcome, and combinations of model effects. This leads to the obvious question of which model is best for the situation at hand. In this article, we explore how to compare models using standard predictive validity metrics and k-fold cross-validation.

Predictive validity metrics

In this tutorial, we will generate the following metrics to assess and compare models:

Log predictive density

In a Bayesian context, we would ideally like to evaluate a posterior predictive distribution $p_{posterior}(\theta)$ against some new data $\tilde{y}$ drawn from the true underlying distribution. If we had this newly-observed data, we could calculate the probability of observing each new point given the predictive distribution, then multiply these across *N* observations to get the overall *predictive density (PD)*:

$$PD = p(\tilde{y}|\theta) = \prod_{i=1}^{N} p(\tilde{y}_i|\theta)$$

When comparing two models against the same new data, the model with the higher predictive density can be considered more consistent with that data.

To make the computation more tractable, we can take the log of both sides to calculate *log predictive density (LPD)*:

$$LPD = \sum_{i=1}^{N} \log p(\tilde{y}_i|\theta)$$

Because we are working with posterior samples rather than the full distribution, we will average density across the set of predictive samples S to approximate LPD:

$$\widehat{LPD} = \sum_{i=1}^{N} \log \left(\frac{1}{S} \sum_{s=1}^{S} p(\tilde{y}_i | \theta_s) \right)$$

In the notation above, we are repeatedly evaluating each data point i (for i = 1 to *N*) against each predictive posterior draw *s* (for *s* = 1 to *S*).

The LPD is expressed as a negative number—smaller negative numbers (those closer to zero) indicate a higher predictive density and therefore a better predictive fit.

Watanabe-Aikake information criterion

The Watanabe-Aikake information criterion (WAIC, also called the "widely applicable information criterion") estimates how well a Bayesian model might fit to new data without actually performing cross-validation. It is calculated as the log pointwise predictive density (LPD, for data used to train the model) and penalized by a term that is larger for more flexible models.

Like other information criteria, WAIC is expressed as a positive number, where smaller numbers indicate a better fit.

Root mean squared error

Root mean squared error will be familiar to anyone who has taken an introductory statistics course:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}$$

In this context, y_i is the observed *rate* of the outcome (calculated as the number of positives divided by the sample size) at data point i, while $\hat{y}_i$ is the model's *mean prediction* at the pixel that overlaps with data point i. RMSE can be calculated "in-sample" against data that was used to fit the model, or "out-of-sample" against data that was held back from the model.

RMSE is simple to interpret; it can also be used as a point of comparison against frequentist estimates like those generated by the machine learning submodels. However, using RMSE to evalulate geostatistical models has downsides:

- Differences in data sample size are ignored: an observation of 1 positive/4 sampled is identical to an observation of 250 positive/1,000 sampled when calculating y_i.
- The uncertainty of model estimates is not taken into account: two models with the same mean estimates but different uncertainty intervals would end up with the same $\hat{y}_i$ values and therefore identical RMSE.

For these reasons, we prefer to use the LPD or WAIC for model evaluation, and to use RMSE either as a backup or a tool for specifically evaluating the model's mean estimates.

Model cross-validation

When calculating predictive validity metrics like LPD and RMSE, we have a tension between (1) wanting to use all available data to fit the model and (2) wanting to hold back some data for model validation. We can deal with this tension using cross-validation.

In **leave one out cross validation**, we evaluate each data point y_i against a model that was trained on the entire dataset *except* for y_i; we calculate each model's predictive validity against the one unobserved data point, then summarize those predictive validity metrics (summing LPD and RMSE) across all subset models. In the aggregate, the performance of each subset model against held-out data points should be similar to the full model's predictive performance against a theoretical unobserved dataset drawn from the same distribution.

Running one model per data point is often too time-intensive to be practical. We can approximate the performance of leave-one-out cross-validation using **k-folds cross-validation**, where the data is randomly split into *k* (often *k* = 5 or 10) "folds" or holdouts. *k* subset models are run: in each, the fold is reserved as a validation set, and the rest of the data is used to train the subset. Predictive validity metrics are repeatedly calculated for the held-out observations and then combined. When *k* is large, the combined metrics will approach the values observed in leave-one-out cross-validation.

In this tutorial, we will calculate all three of our standard metrics for the full model; we will then calculate "out-of-sample" LPD and RMSE using 10-fold cross validation.

Setup

In this tutorial, we will continue to use example data on child stunting from Benin. We will compare the default `mbg` model, as described in the introductory vignette, to a stacked ensemble model with department-level fixed effects, as described in the spatial ML models article.

Start this tutorial by loading the example data and preparing the ID raster:

```
# Load packages
library(data.table)
library(sf)
library(terra)
library(mbg)

# Load input data, covariates, and department boundaries
outcomes <- data.table::fread(
```

```
  system.file('extdata/child_stunting.csv', package = 'mbg')
)
covariates <- list(
  access = terra::rast(system.file('extdata/access.tif', package = 'mbg')),
  evi = terra::rast(system.file('extdata/evi.tif', package = 'mbg')),
  temperature = terra::rast(system.file('extdata/temperature.tif', package = 'mbg'))
)
covariates$intercept <- covariates[[1]] * 0 + 1
departments <- sf::st_read(
  system.file('extdata/Benin_departments.gpkg', package = 'mbg'),
  quiet = TRUE
)

# Create ID raster
id_raster <- mbg::build_id_raster(
  polygons = departments,
  template_raster = covariates[[1]]
)
```

In-sample model comparison

First, run both the standard and stacking model types using the full dataset. These are the “in-sample” models, as no data was reserved for validation.

```
# Standard model (in-sample)
standard_model_is <- mbg::MbgModelRunner$new(
  input_data = outcomes,
  id_raster = id_raster,
  covariate_rasters = covariates,
  verbose = FALSE
)
standard_model_is$run_mbg_pipeline()

# Stacked generalization model (in-sample)
# Same cross-validation settings
cross_validation_settings <- list(method = 'repeatedcv', number = 5, repeats = 5)
submodel_settings <- list(enet = NULL, gbm = list(verbose = FALSE), treebag = NULL)
stacking_model_is <- mbg::MbgModelRunner$new(
  input_data = outcomes,
  id_raster = id_raster,
  covariate_rasters = covariates,
  use_stacking = TRUE,
  stacking_cv_settings = cross_validation_settings,
  stacking_model_settings = submodel_settings,
  stacking_prediction_range = c(0, 1),
  stacking_use_admin_bounds = TRUE,
  admin_bounds = departments,
  admin_bounds_id = 'department_code',
  verbose = FALSE
```

```
)
stacking_model_is$run_mbg_pipeline()
#> Loading required package: ggplot2
#> Loading required package: lattice
```

Use the `MbgModelRunner$get_predictive_validity()` method to calculate in-sample LPD, WAIC, and RMSE for each model:

```
standard_model_metrics <- standard_model_is$get_predictive_validity()
standard_model_metrics$model_type <- "Standard"
stacking_model_metrics <- stacking_model_is$get_predictive_validity()
stacking_model_metrics$model_type <- "Stacked ensemble"

metrics_in_sample <- rbind(
  standard_model_metrics,
  stacking_model_metrics
)
metrics_in_sample
#>      rmse_is    lpd_is  waic_is       model_type
#>        <num>     <num>    <num>           <char>
#> 1: 0.1285057 -1206.084 2397.115         Standard
#> 2: 0.1163582 -1156.990 2304.211 Stacked ensemble
```

The stacked ensemble model has a slightly lower RMSE and a higher (less negative) LPD, indicating a closer fit to the observed data. However, this may just indicate a more flexible model fit—overfitting could lead to worse model performance when comparing against new, unobserved data.

We can also calculate the in-sample RMSE of each component ML model from the stacked ensemble model:

```
outcomes[, data_rate := indicator / samplesize]
ml_submodels <- stacking_model_is$model_covariates

submodel_rmse <- data.table::data.table(
  rmse_is = c(
    mbg::rmse_raster_to_point(
      estimates = ml_submodels$enet,
      validation_data = outcomes,
      outcome_field = 'data_rate'
    ),
    mbg::rmse_raster_to_point(
      estimates = ml_submodels$gbm,
      validation_data = outcomes,
      outcome_field = 'data_rate'
    ),
    mbg::rmse_raster_to_point(
      estimates = ml_submodels$treebag,
```

```
      validation_data = outcomes,
      outcome_field = 'data_rate'
    )
  ),
  model_type = c('Elastic net', 'Gradient boosted machines', 'Bagged regression trees')
)
submodel_rmse
#>      rmse_is                model_type
#>        <num>                    <char>
#> 1: 0.1357682               Elastic net
#> 2: 0.1268371 Gradient boosted machines
#> 3: 0.1157924   Bagged regression trees
```

Out-of-sample model comparison

To better approximate model performance with unobserved data, we will perform 10-fold validation using the input dataset. This requires running both the standard and stacking models 10 times on different subsets of the observed point data. For each of the 10 sub-models, we generate predictive validity metrics based on the held-out portion of the data.

Note that WAIC is an in-sample predictive validity metric: because `in_sample = FALSE`, WAIC is not generated during these runs.

```
# Assign a new `holdout_id` field in the data, which are shuffled integers from 1 to 10
n_holdouts <- 10
outcomes$holdout_id <- seq_len(n_holdouts) |>
  rep(length.out = nrow(outcomes)) |>
  sample()

# For each of the 10 holdouts, get both models' *out-of-sample* predictive validity
metrics_by_holdout <- lapply(seq_len(n_holdouts), function(holdout){
  # Split into (observed) training data and (unobserved) testing data
  train <- outcomes[holdout_id != holdout,]
  test <- outcomes[holdout_id == holdout,]
  # Run both models
  standard_model_oos <- mbg::MbgModelRunner$new(
    input_data = train,
    id_raster = id_raster,
    covariate_rasters = covariates,
    verbose = FALSE
  )
  standard_model_oos$run_mbg_pipeline()
  stacking_model_oos <- mbg::MbgModelRunner$new(
    input_data = train,
    id_raster = id_raster,
    covariate_rasters = covariates,
    use_stacking = TRUE,
```

```
    stacking_cv_settings = cross_validation_settings,
    stacking_model_settings = submodel_settings,
    stacking_prediction_range = c(0, 1),
    stacking_use_admin_bounds = TRUE,
    admin_bounds = departments,
    admin_bounds_id = 'department_code',
    verbose = FALSE
  )
  stacking_model_oos$run_mbg_pipeline()

  # Compare to the test data
  standard_metrics <- standard_model_oos$get_predictive_validity(
    in_sample = FALSE,
    validation_data = test
  )[, model_type := 'Standard']
  stacking_metrics <- stacking_model_oos$get_predictive_validity(
    in_sample = FALSE,
    validation_data = test
  )[, model_type := 'Stacked ensemble']
  this_holdout_metrics <- rbind(standard_metrics, stacking_metrics)
  this_holdout_metrics$holdout_id <- holdout

  # Return the combined metrics for this holdout
  return(this_holdout_metrics)
}) |>
  data.table::rbindlist()
```

rmse_oos	lpd_oos	model_type	holdout_id
0.1179354	-118.7490	Standard	1
0.1252479	-120.7753	Stacked ensemble	1
0.1291436	-120.6255	Standard	2
0.1192507	-117.7843	Stacked ensemble	2
0.1496129	-126.8926	Standard	3
0.1504705	-128.2862	Stacked ensemble	3
0.1414287	-125.6910	Standard	4
0.1419527	-128.7975	Stacked ensemble	4
0.1363021	-131.5173	Standard	5
0.1386044	-136.6189	Stacked ensemble	5

rmse_oos	lpd_oos	model_type	holdout_id
0.1391739	-123.5967	Standard	6
0.1345811	-121.4608	Stacked ensemble	6
0.1388889	-118.9464	Standard	7
0.1414557	-121.3938	Stacked ensemble	7
0.1252319	-122.7438	Standard	8
0.1268031	-124.5918	Stacked ensemble	8
0.1466681	-127.1125	Standard	9
0.1383814	-124.0828	Stacked ensemble	9
0.1374990	-123.0405	Standard	10
0.1483755	-129.3895	Stacked ensemble	10

The models trade off for better performance across the ten holdout folds. To get an overall comparison of out-of-sample predictive validity, we combine these metrics across holdouts:

```
out_of_sample_overall <- metrics_by_holdout[
  , .(rmse_oos = mean(rmse_oos), lpd_oos = sum(lpd_oos)),
  by = model_type
]
```

model_type	rmse_oos	lpd_oos
Standard	0.1361885	-1238.915
Stacked ensemble	0.1365123	-1253.181

In this case, the standard model has better out-of-sample LPD and RMSE, suggesting that it is the superior model for predicting new child stunting observations in Benin.

Summary

We can generalize this tutorial into a general strategy for comparing predictive models in a particular country/outcome/dataset context:

1. Select the universe of model types to test

2. Before running any models, select the criteria for model selection. We recommend out-of-sample LPD as the primary criterion for model selection, with out-of-sample RMSE as a tie-breaker for models with near-identical LPD.
3. Split the data into *k* holdouts. Run a model for each holdout, evaluating the subset model against the held-out data. *Model comparisons should always be run against the same outcomes dataset and holdouts.*
4. Based on the criteria selected in (2), choose the model with the top performance
5. Run a full model with this same specification using all observations (no holdouts). These are the best results from this comparison round.

Further reading

Hastie, T., Tibshirani, R., & Friedman, J. (2017). Model assessment and selection. *The elements of statistical learning: data mining, inference, and prediction (2E)*, 219-259. https://hastie.su.domains/ElemStatLearn/

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and computing*, 27, 1413-1432. https://link.springer.com/article/10.1007/s11222-016-9696-4

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of machine learning research*, 11(12). https://www.jmlr.org/papers/volume11/watanabe10a/watanabe10a.pdf

---