**QA Framework for Predictive Modeling**

Authors: Ellie White,

05/19/2023

# Scope

* Objectives for producing a quality assurance plan for predictive modeling is to ensure accuracy, completeness, reliability, and efficiency and to avoid ineffective experimental designs, inappropriate analyses, and flawed reasoning.
* This document lists thoughts that are unique to predictive modeling and attempts to not repeat best practices and processes that overlap with other sections in the [QA framework](https://doimspp.sharepoint.com/:w:/r/sites/IIDDStaff/Shared%20Documents/IPDS%20and%20FSP/ITRP%20-%20Internal%20Technical%20Review%20Procedures/Draft%20QA%20Framework%20for%20Review.docx?d=w40b8bc63e4e542e4829e7237d6ec15cb&csf=1&web=1&e=k9Imwj) like version control, peer review, continuous integration and deployment etc.

# The Predictive Modeling Commandments

* Justify discarding data points.
  + E.g., we remove basins from an analysis because the watershed is considered “altered” or the gage is right below a reservoir.
  + E.g., Sometimes to avoid spurious correlations (caused by extreme/outlier data points or subgroups), we must remove data points. Reported correlations should be accompanied by a scatterplot.
* Test **parametric** modeling assumptions – assumptions like multivariate normality, no or little multicollinearity, no auto-correlation, homoscedasticity
  + E.g., linear model residuals should show no patterns (assumption is that errors come from a random distribution) – check normality with Q-Q plots
  + E.g., linear model standardized residuals should have a constant variance (homoscedasticity)
  + Check for outliers and high-leverage points (Cook’s distance scores)
* Use a big enough sample size - otherwise analysis may be overconfident and avoid drawing conclusions based on insufficient data.
  + In the case of imbalanced classes, construct replications (with tiered re-sampling)
* Avoid information leaks (cross-validation).
  + Test set should be set aside for final analysis and no alterations to the modeling framework should be allowed after the test set is applied and the model is evaluated. Best practice is to have a training – validation – test splits. Training and validation splits can be used iteratively, and test set is preserved for final model evaluation.
  + Good references:
    - Roberts, D. R., Bahn, V., Ciuti, S., Boyce, M. S., Elith, J., Guillera‐Arroita, G., ... & Dormann, C. F. (2017). Cross‐validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, *40*(8), 913-929. [[LINK](https://onlinelibrary.wiley.com/doi/full/10.1111/ecog.02881)]
* Set your seed once or do a replication study.
  + The more tests you run the more likely you are to encounter a false positive result (p-hacking). Report results with caution as there is always a chance the stars have aligned to give you the result you want.
* Establish a process for systematically tuning model hyperparameters.
  + Consider techniques such as grid search, random search, or Bayesian optimization.
  + Step through the tires of complexity if the gains justify the effort (in most NN architectures they don’t).
* Construct appropriate evaluation metrics that align with the domain and problem.
  + E.g., In the case of streamflow droughts, should the problem be a regression type of problem predicting streamflow or a classification problem predicting the drop below a threshold.
* Model selection for release – right now I think we generally say the latest = best. Not good
* Set up a monitoring system to track the performance of ML models as they train.
* Feature selection plan – selection bias
* Model inference, issue of confounding variables