CROP BREEDING AND APPLIED BIOTECHNOLOGY

ARTICLE

Technical nuances of machine learning: implementation and validation of supervised methods for genomic prediction in plant breeding

Crop Breeding and Applied Biotechnology
21(S): e381421S2, 2021
Brazilian Society of Plant Breeding.
Printed in Brazil
http://dx.doi.org/10.1590/1984-70332021v21Sa15

Alencar Xavier [1*]

Abstract: *The decision-making process in plant breeding is driven by data. The machine learning framework has powerful tools that can extract useful information from data. However, there is still a lack of understanding about the underlying algorithms of these methods, their strengths, and pitfalls. Machine learning has two main branches: supervised and unsupervised learning. In plant breeding, supervised learning is used for genomic prediction, where phenotypic traits are modeled as a function of molecular markers. The key supervised learning algorithms for genomic prediction are linear methods, kernel methods, neural networks, and tree ensembles. This manuscript provides an insight into the implementation of these algorithms and how cross-validations can be used to compare methods. Examples for genomic prediction come from plant breeding.*

Keywords: *Statistical learning, plant breeding, genomics prediction, algorithms.*

INTRODUCTION

Modern plant breeding relies on the ability to identify superior germplasm, and machine learning provides the tools to enable data-driven genotype selections. A key motivation for using machine learning in plant breeding comes from the ever increasing volume of data, which is generated by high-throughput phenotyping and genotyping (Lin and Lane 2017, Perakakis et al. 2018), and complemented by rich environmental information from weather stations and satellites (Costa-Neto et al. 2021, Shahhosseini et al. 2021). More data require a change in the analytical mindset as traditional statistical methods may not be suited for large datasets, and novel algorithms might help to extract new meaningful patterns from data. Knowing and understanding the different types of machine learning methods help breeders to choose the right method for a given task, and to parameterize them with their knowledge (Zampieri et al. 2019).

Machine learning can be viewed as the combination of numeric optimization and statistics (Hastie et al. 2009, Goodfellow et al. 2016). Two main branches exist: unsupervised and supervised learning. Unsupervised learning deals with associations among features, whereas supervised learning fits a response variable (*y*) as a function of features (*X*). In genomic prediction, supervised machine learning methods train models that use phenotypes of agronomic traits as a function of molecular markers, usually single nucleotide polymorphisms (SNP).

***Corresponding author:**
E-mail: alenxav@gmail.com
ORCID: 0000-0001-5034-9954

Received: 03 May 2021
Accepted: 10 May 2021
Published: 31 May 2021

[1] Corteva Biostatics Adjunct Faculty, Purdue University Johnston, Iowa, USA 50131

This manuscript reviews supervised learning methods used for genomic prediction in plant breeding, with focus on both implementation and evaluation by cross-validations.

METHODS

The machine learning literature often diverges from the notation utilized in traditional statistics. This section uses algebraic notation familiar to plant breeders in order to describe supervised machine learning methods, which can be categorized as linear methods, kernel methods, neural networks, and tree ensembles (Izenman 2008, Hastie et al. 2009, Shalev-Shwartz and Ben-David 2014, Goodfellow et al. 2016, Deisenroth et al. 2020).

Linear methods

The response variable is modeled as a linear combination of explanatory variables that are called features in the machine learning literature. Although these methods are called "linear", the features may or may not be linear. Linear models can be written as

$$y = Xb + e$$

where y is an n×1 vector that contains response variables with n observations, X is an n×p design matrix of p feature variables, b is a p×1 vector of regression coefficients, and e is an n×1 vector of residuals. In genomic prediction, the features are scores at different marker loci and the coefficients are marker effects. Equation systems of linear models can be solved efficiently using an iterative procedure called *coordinate descent*, which marginalizes the model to solve one univariate equation at a time. The marginal model for j^{th} element of b can be derived as follows:

$$y = Xb + e$$
$$y = X_{-j}b_{-j} + x_jb_j + e$$
$$y - X_{-j}b_{-j} = x_jb_j + e$$
$$y_j = x_jb_j + e$$

where y_j equals the response variable y adjusted for the effects of features times their coefficients, except for the ones at position j of b. This so-called conditioning is based on isolating the j^{th} coefficient from the other coefficients, which is achieved by partitioning Xb into the two terms $X_{-j}b_{-j}$ and x_jb_j.

Linear methods differ by their univariate solution for b_j. These methods include least square error (Stigler 1981), least absolute error (Geary 1935), ridge regression "RR" (Hoerl and Kennard 1970), least absolute shrinkage and selection operator "LASSO" (Tibshirani 1996) and the elastic-net (Zou and Hastie 2005), which were introduced in the years 1722, 1935, 1970, 1996, and 2005, respectively. A general property of linear methods is that coefficients b_j are solved with the objective of increasing the model goodness of fit while subjected to a constraint, modeled with a parameter λ. The constraint on the coefficients is commonly referred to as "complexity" or "bias". The various solutions for the regression coefficient b_j are summarized in Table 1.

The parameter λ controls the balance between the model variance and complexity. Higher values of λ reduce the goodness of fit on the training data by reducing the complexity of the model but increase its predictive potential. The least squares and least absolute error solutions are not subjected to any constraint, which causes models to overfit if the number of parameters surpasses the number of observations. In addition, least absolute error has no analytical derivation and multiple solutions have been proposed over time, generally based on estimators that utilize the median

Table 1. Solutions for coefficient b_j for univariate linear systems ($y_j = x_jb_j + e$) with different methods (Legarra and Misztal 2008, Hastie et al. 2009, Xavier et al. 2016)

	Least Squares	Least Absolute	Ridge	LASSO	Elastic Net
Solution	$\frac{Cov(x_j,y_j)}{Var(x_j)} = \frac{x_j' y_{-j}}{x_j' x_{-j}}$	$\frac{Median(x_j\#y_j)}{Var(x_j)}$	$\frac{x_j' y_{-j}}{x_j' x_{-j} + \lambda}$	$\frac{x_j' y_{-j} - \lambda}{x_j' x_j}$	$\frac{x_j' y_{-j} - \lambda_1}{x_j' x_j + \lambda_2}$
Minimizes	$\hat{e}'\hat{e}$	$\Sigma\lvert\hat{e}\rvert$	$\hat{e}'\hat{e} + \lambda\hat{b}'\hat{b}$	$\hat{e}'\hat{e} + \lambda\Sigma\lvert\hat{b}\rvert$	$\hat{e}'\hat{e} + \lambda_1\Sigma\lvert\hat{b}\rvert + \lambda_2\hat{b}'\hat{b}$
Unique solution	Yes	No	Yes	No	Yes

(Li and Arce 2004). The λ parameter in ridge regression inflates the denominator and causes a shrinkage of the regression coefficients. For LASSO coefficients, λ causes shrinkage by deflating the nominator and removing it from the model altogether if $\lambda > |x_j'y_j|$. Elastic-net was proposed as the combination of ridge regression and LASSO solutions.

In ridge regression, coordinate descent can be efficiently computed by Gauss-Seidel (Legarra and Misztal 2008) to avoid the explicit conditioning ($y_j = y - X_{-j}b_{-j}$). A Gauss-Seidel algorithm with residual update operates by updating coefficient *j* with subsequent update of residuals:

$$\hat{b}_j^{t+1} = \frac{x_j'\hat{e}^t + x_j'x_j\hat{b}_j^t}{x_j'x_j + \lambda}$$

$$\hat{e}^{t+1} = \hat{e}^t - x_j(\hat{b}_j^{t+1} - \hat{b}_j^t)$$

Because the coefficients are solved one at a time, a single linear model can accommodate a combination of methods when a subset of coefficients is subjected to one constraint, whereas another set of coefficients is subjected to another constraint. A popular example in plant breeding is the mixed model, where fixed effects are solved via (generalized) least squares and random effects as ridge regression. Moreover, in models with multiple random terms, each set of coefficients is subjected to its own shrinkage parameter λ.

Under traditional machine learning settings, the shrinkage parameter λ is commonly tuned via cross-validation. An exception is found for the ridge regression models containing one or multiple λ parameters, where statisticians and geneticist infer λ using Bayesian or likelihood methods. Lambda has an analytical solution through variance components as $\lambda = \sigma_e^2\sigma_b^{-2}$ (Xavier et al. 2016). More details are provided in subsection 'analytical tuning'.

Kernel methods

A symmetric matrix *K*, called kernel, is calculated from the features (*X*) to describe the similarity among observations (Signoretto and Suykens 2015). Kernel methods are used in plant breeding for pedigree-based selection, genomic prediction, and spatial adjustments. This framework does not allow to store the model and both training and prediction data must be available when solving the equations. Some functions to build kernels are presented in Table 2. Main kernel-based methodologies include: kernel ridge regression "KRR" (e.g., GBLUP and Kalman filters), kriging, splines, stationary auto-regressive (i.e., $AR1 \times AR1$), reproducing kernel Hilbert spaces "RKHS", support-vector regression "SVR", and reduced ranking kernels, i.e., principal components.

When the kernel is based on linear combinations of the features, i.e., $K = XX'$, it yields the same solution as linear methods. Kernel methods provide a convenient way to compute interactions, which is often necessary to parameterize epistasis in genomic prediction (Xu 2013), especially when it is not feasible to apply linear methods for pairwise combinations of markers. The interaction kernels are computed as $K_{A\times B} = K_A \# K_B$, using the Hadamard product (#, element-wise multiplication) and the main-effect kernels K_A and K_B.

The computation of predictions from kernel methods are found through a linear system of equations (Misztal and Legarra 2017). A kernel can also be decomposed linearly, either by Cholesky or Eigen decomposition, and solutions of the model can be obtained iteratively as with linear methods (de los Campos et al. 2010). Different factorization methods for obtaining kernel solutions are summarized in Table 3. The equations are subjected to the parameter of the constraint λ presented for linear methods, which is found through cross-validation or estimated from variance components.

Table 2. Computation of kernels (*K*) as a function of a set of features (*X*)

	Linear Kernel	Gaussian Kernel "RBF"	Arccosine Kernel
Solution	$\alpha^{-1}x_i'x_j$	$exp\left[-\kappa \sum_{j=1}^{J} (x_i - x_j)^2\right]$	$\pi^{-1}(x_i'x_i)(x_j'x_j)\, sin(\theta_{ij}) + (\pi - \theta_{ij})\, cos(\theta_{ij})$
Parameter	$\alpha = \sum_{j=1}^{J} Var(x_j)$	κ is tuned	$\theta_{ij} = cos^{-1}Corr(x_i, x_j)$
Known for	GBLUP, Kalman Filter	RKHS, SVR	Deep Kernel method
Reference	VanRader (2008)	González-Camacho et al. (2012)	Cuevas et al. (2019)

Table 3. Solving kernel linear systems $y = g + e$, where $g = K(X)$ (Stranden and Garrick 2009, de los Campos et al. 2010, Signoretto and Suykens 2015)

Factorization	None	Inverse	Eigen	Cholesky
Transformation	$f(K) = K$	$f(K) = K^{-1}$	$f(K) = UDU'$	$f(K) = LL'$
Solution	$(K + \lambda I)g = Ky$	$(I + \lambda K^{-1})g = y$	$(I + \lambda D^{-1})g = U'y$	$(L'L + I\lambda)g = L'y$

Reduction of dimensionality of a kernel is achieved by Eigen decomposition. If the kernel K is decomposed into eigenvectors (matrix U) and their respective eigenvalues (diagonal matrix D), one may reconstruct K with a selected number of eigenpairs, which are usually sorted by the amount of variance they explain. Pocrnic et al. (2018) suggested to use enough eigenpairs to capture 98% of the total variation in the kernel (sum of the eigenvalues), which may increase the accuracy of prediction and speed up convergence. Reduction of dimensionality is also used in principal component regressions, where a set of eigenvectors, which are extracted from of a given kernel, become features of a linear model.

Inverse kernels (K^{-1}) are referred to as *precision matrices*. It is often possible to compute the precision matrix, whether dense or sparse, without having to build the kernel itself in order to save computing time. A classic example is the inverse of the pedigree relationship matrix proposed by Henderson (1975). The $AR1{\times}AR1$ covariance structure among observations, commonly utilized in the spatial adjustment of field experiments in plant breeding, is another example where the sparse precision matrix is computed instead of the kernel itself. The construction of sparse precision matrices directly from the feature matrix (X) is an active area of research in machine learning (Cai et al. 2011, Liu and Luo 2015).

Neural networks

Neural networks are generalizations of linear methods that utilize *latent spaces* (H). Latent spaces are linear combination of features (XB) subjected to a non-linear transformation called an *activation function* (α), thus $H = \alpha(XB)$. Each latent variable, corresponding to a column of H, is called a *node*. Latent spaces can be used to predict response variables ($y = Hb + e$) or can be transformed into more complex latent spaces, $H_{i+1} = \alpha(H_iB)$, which creates "hidden layers" in the network (Shalev-Shwartz and Ben-David 2014, Goodfellow et al. 2016). More layers increase the complexity and non-linearity of the latent spaces. Networks with two or more hidden layers are referred to as *deep neural network* (DNN). A neural network with a single layer and linear activation function, i.e., $\alpha(x) = x$, is analogous to a principal component regression (PCR), where the number of nodes corresponds to the number of eigenvectors obtained via single-value decomposition. With the implementation of hidden layers, a progression is drawn from a linear method to a DNN as follows:

$y = Xb + e$	Linear method
$y = (XB_1)b_2 + e$	Principal components regression
$y = \alpha(XB_1)b_2 + e$	Neural network
$y = \alpha(\alpha(XB_1)B_2)\, b_3 + e$	Deep neural network

There is no known analytical solution for the joint estimation of regression coefficients across layers. The regression coefficients in the neural network (B_1, B_2, b_3) are estimated using an iterative procedure called *gradient descent*. Gradient descent, although not regarded as a good algorithm for solving systems of equations, is the preferred methodology for deep networks due to computational simplicity, that is, it only requires one matrix multiplication. In a linear system of equations ($y = Xb + e$), gradient descent solves for regression coefficients as follows:

$$\hat{b}^{t+1} = \hat{b}^{t} - r\nabla$$

$$\hat{b}^{t+1} = \hat{b}^{t} - r\,[-2n^{-1}X'(y - X\hat{b}^{t}) + 2n^{-1}\lambda\hat{b}^{t}]$$

$$\hat{b}^{t+1} = \hat{b}^{t} - r\,(-2n^{-1}X'\hat{e} + 2n^{-1}\lambda\hat{b}^{t})$$

$$\hat{b}^{t+1} = \hat{b}^{t} + 2n^{-1}X'\hat{e} - 2n^{-1}\lambda\hat{b}^{t}$$

$$\hat{b}^{t+1} = \hat{b}^{t} + 2n^{-1}r(X'\hat{e} - \lambda\hat{b}^{t})$$

where r is the *learning rate* or step-size, n is the number of observations, ∇ is the gradient obtained from the first derivative of the ridge function: $\hat{e}'\hat{e} + \lambda\hat{b}'\hat{b}$. The vector $\hat{e}$ corresponds to the residuals ($\hat{e} = y - X\hat{b}$) and λ is the constraint utilized for the case of ridge regression. The learning rate controls the convergence rate of the network. The residuals must be *back-propagated* to the inner layers to solve coefficients within the latent spaces. Neural networks use random values as starting point for the regression coefficients to enable the propagation of residuals in the first iteration. Fitting a DNN involves three steps:

1. Fit neural network

a. $H_1 = \alpha(XB_1)$

b. $H_2 = \alpha(H_1B_2)$

c. $\hat{y} = H_2b_3$

2. Compute gradient

a. $e_3 = y - \hat{y}$

b. $E_2 = \alpha(e_3B'_3)$

c. $E_1 = \alpha(E_2B'_2)$

3. Update coefficients

a. $b_1^{t+1} = b_1^{t} + 2r_1n^{-1}r(X'E_1 - \lambda_1b_1^{t})$

b. $b_2^{t+1} = b_2^{t} + 2r_2n^{-1}r(H'_1E_2 - \lambda_2b_2^{t})$

c. $b_3^{t+1} = b_3^{t} + 2r_3n^{-1}r(H'_2e_3 - \lambda_3b_3^{t})$

The intercepts have been omitted for simplicity. Otherwise an additional column filled with 1's is added to the feature matrices (X, H_1 and H_2), which is not subjected to a constraint ($\lambda = 0$). Learning rates (r_1,r_2,r_3) may vary across parameters and change with increasing iterations. A few algorithms have become the gold standard to track and update rates, which are the adaptive momentum "Adam" (Kingma and Adam 2014) and the root mean square propagation "RMSProp" (Xu et al. 2021). Optimizing learning rates may increase the computational cost of fitting networks because it involves storing gradients from previous iterations, but this computational cost is offset by faster convergence. For large datasets, coefficients are solved using stochastic gradient descent (SGC). In each iteration of SGD, a small subset of the data, known as "batch", is randomly sampled and used to update the coefficients. The batch size limits the amount of data utilized in each iteration, which can considerably reduce the computation time involved in updating the regression coefficients (Takase 2021). To determine convergence, it is useful to prepare a "test set" to assess the prediction power of the network after each iteration.

The activation functions (α) used for element-wise transformation of latent spaces is a key parameter in DNNs. Hyperbolic tangent function (Tanh) and Rectifier Linear Unit (ReLU) are the two most commonly used transformations in genomic prediction (Montesinos-López et al. 2021). Activation functions must be easy to compute, because they are used to fit the network and propagate the residuals. Tanh is a non-linear function that transforms the input values, estimated from the linear combinations of features, into values ranging from -1 to 1. Any value below -2 or above 2 is set to -1 or 1 by Tanh, respectively, hence Tanh is sensitive to the scale of the inputs and may benefit from the normalization of the response variables. ReLU sets negative values to zero without altering positive values. The "leaky" version of ReLU shrinks negative values by a pre-defined constant, δ, instead of eliminating them altogether, thus: $\alpha(x,\delta) = x$ (if $x > 0$), otherwise $x = \delta x$. The ReLU activation function is believed to provide universal approximation for any polynomial function as the number of nodes increase (Hanin 2019). Prediction from DNN based on the ReLU activation can benefit from a procedure called *dropout* (Baldi 2014), which refers to ignoring a proportion of nodes, chosen at random in each iteration, to mitigate codependency of latent spaces (Hahn and Choi 2020).

Tree ensembles

Models based on regression trees (Breiman et al. 1984) are algorithmic-driven and do not involve explicit algebra. Regression trees are *weak learners* and benefit from ensembles, which refers to the process of averaging multiple trees through sampling techniques, such as bagging and boosting. Hierarchically, the implementation of tree ensembles is based on three components: 1) Recursive partitioning; 2) Tree building; 3) Ensemble method.

1) Recursive partitioning: Defines the breakdown of a single feature (x) that best separates the response variable (y) into two groups, called *nodes*, based on some metric to be minimized. The squared error is the most common metric for modeling continuous variables. The main algorithm utilized to find partitioning threshold is named Classification and Regression Tree Algorithm (CART). In genomic prediction, there are only two possible splitting points for when recursive partitioning is applied to a diploid SNP marker (x) with genotypes numerically coded as 012: "$x \leq 1$" or "$x < 1$", representing (AA,Aa)/(aa) and (AA)/(Aa,aa), respectively. The same marker may provide further partitioning of the data in the node containing two classes. The output of the recursive partitioning function must include (1) the split point that minimizes the square error and (2) the amount of variance explained. Both information will be used to build trees.

2) Tree building: Trees are built from sequentially through recursive splits. Given the set of the features (X), a tree is built by checking which of the features offers the highest reduction in squared error by recursive partitioning. This will define which feature, and where, the first split of the tree occurs. The data is split into the two node and all features in X are reassessed within node to generate the second split rule. The process is repeated within subsequent node until no further reduction in squared error is achieved. The complexity of the regression trees controlled by (1) limiting tree depth, (2) limiting the number of observations per node, and (3) by the defining a minimum reduction of squared error that justify partitioning.

3) Ensemble method: The ensemble of multiple trees is done in one of two ways: bagging and boosting. Bagging trees yield a model known as random forest (Breiman 2001), whereas boosting trees yield as model referred to as gradient boosting machine "GBM" (Friedman 2001), with variations known as AdaBoost (Freund and Schapire 1997) and XGBoost (Chen and Guestrin 2016).

Bagging consists of creating a predefined number of trees (e.g., $M = 500$), where each tree T_m fits a subset of individuals $y_{n \subset N}$ as a function of a subset of features $X_{n \subset N, p \subset P}$. The final random forest prediction is the average across trees calculated as $\hat{y} = M^{-1} \sum_{m=1}^{M} T_m(X)$. Features utilized in each tree are sampled at random from a fraction of all features available (e.g., $\sqrt{P}$). Random forest has been used in genomic prediction and genome-wide association analysis (Schwarz et al. 2010, Chen and Ishwaran 2012, Botta et al. 2014, Wright and Ziegler 2015, Xavier and Rainey 2020).

Boosting consists of fitting regression trees sequentially to improve fitness. Boosting starts with fitting a regression tree T_1 using a random set of the observations (y_1), sampled from the data ($y_1 \subset y$) and model it as a function of the corresponding features X_1. Next, a second set of observations (y_2) is sampled and the vector of gradients, computed as $e_2 = y_2 - T_1(X_2)$, serves as response variable to fit T_2 as a function of X_2. Likewise, subsequent trees T_{m+1} fit from random samples of observations, y_{m+1}, with gradients estimated from the current set of trees, $e_{m+1} = \sum_{m=1}^{M} T(X_{m+1})$. Thus, the third tree, T_3, fits the gradients e_3, estimated as $e_3 = y_3 - T_1(X_3) - T_2(X_3)$, as a function of X_3. The final boosting prediction is the sum from all trees as $\hat{y} = \sum_{m=1}^{M} T_m(X)$.

Both ensemble methods are controlled by multiple parameters at both the tree and ensemble level to control the complexity of the model. Due to the large number of correlated features, genomic prediction models from tree ensembles can provide more stable solutions as the number trees increases and the depth of the trees decreases.

Analytical tuning

The parameter λ that controls the shrinkage of both ridge regression and kernel ridge regression has an analytical solution based on variance components ($\lambda = \sigma^2_e \sigma^{-2}_u$) and, therefore, does not require cross-validation for tuning. Most genetic analyses infer variance components instead of tuning based on cross-validation, because they are used to estimate heritability. A common likelihood function used for variance components, for both ridge regression and kernel methods, is Restricted Maximum Likelihood "REML" (Corbeil and Searle 1976).

REML is used to for linear models of the form $y = Xb + Zu + e$, where the response variable (y) is modeled as a function of features and their effects that are solved via least squares (fixed effects, Xb) and ridge regression (random effects, Zu). The regression coefficients u and residuals e are assumed to follow a normal distribution, described as $u \sim N(0, K\sigma^2_u)$ and $e \sim N(0, R\sigma^2_e)$, respectively. The model assumes that the response variable is normally distribute as $y \sim N(Xb, V)$, where the variance-covariance matrix is defined as $V = ZKZ'\sigma^2_u + R\sigma^2_e$. The matrix R is the residual correlation kernel, usually set as $R = I$; matrices X and Z are design matrices of b and u, respectively, K is the kernel matrix of the random term. REML is a convex function with unique solutions for variance components, but it is considerably more complex to optimize than simpler functions such as the mean squared error. The expectation of variance components based on the first derivative is (Searle et al. 2009):

$$\sigma_i^2 = \frac{yPV_iPy}{tr(PV_i)}$$

where, P is the projection matrix, $P = V^{-1} - V^{-1}X(X'V^{-1}X)^{-1}X'V^{-1}$. The matrix V_i corresponds to the first derivative of V with respect to the i^{th} variance component, such that $\partial V/\partial\sigma_e^2 = R$ and $\partial V/\partial\sigma_u^2 = ZKZ'$. Alternatives to REML variance components that may provide lower computational cost include Bayesian methods (Sorensen and Gianola 2007, Rue et al. 2017) and likelihood approximations (Schaeffer 1986, Vanraden and Jung 1988).

CROSS-VALIDATION

Collecting new data is costly, therefore prediction models are usually evaluated with data at hand. Data can be split into two subsets, one is used for training, and the other one is used for testing. This procedure is called *cross-validation* (CV).

Cross-validation schemes:

The main types of cross-validation schemes used in plant breeding are: Random K-fold, leave-one-out (LOO), and holdout, where each cross-validation scheme serves a different purpose.

1) Random K-fold. In this scheme, a fraction of $1/k$ of the dataset is sampled for testing, and the prediction metric is averaged across samples. Random k-fold validation is broadly used for comparing methods as it gives a sense for the prediction on new individuals for an observed set of environments. The exact interpretation may vary according to the population structure of the dataset. In genomic prediction, random k-fold provides a general assessment of the predictive potential of the model.

2) Leave-one-out. This scheme is useful to assess data with known structure, including the prediction of unobserved populations (i.e. leave-population-out) and genotype-by-environment interactions (i.e. leave-location-out, leave-geography-out). The interpretation of the result relies on which set of observations are being left out. LOO can mitigate contamination of data structure that random data splits do not consider. In the context of cross-validation, *contamination* refers to observations shared in both training and validation set (Runcie and Chang 2019).

3) Holdout. This scheme aims to replicate the true application of the prediction model. A common example is to train the model with data from previous growing seasons for predicting environments and selection candidates of the most recent growing season. These selection candidates come from new breeding crosses with linkage disequilibrium patterns that are usually present in the training dataset (Pszczola and Calus 2016).

Validation metrics

Key metrics for assessing the machine learning models in plant breeding are: Correlation, prediction error, and success.

1) Correlation. It is measured by Pearson or Spearman coefficients between observed and predicted values. The Pearson correlation coefficient is a metric that captures differences in both ranking and magnitude. Pearson correlations also referred here to as *predictability*, can be re-scaled into accuracy by dividing it by the squared root of the heritability (Lehermeier et al. 2013). Spearman's coefficient, or rank-order correlation, is the main alternative to the Pearson's coefficient. It does not capture magnitude, but it is a direct measure of ranking. In genomic prediction, the Spearman correlation is used to measure the overall alignment of two selection methods (Heslot et al. 2012).

2) Prediction error. Generally measured as the mean squared prediction error (MSPE), is not commonly applied in plant breeding for selection purposes, but it suits models of disease risk, probabilities of events, and other predictions where any shrinkage on prediction values is detrimental. Prediction error is the preferred metric to tune parameters of neural networks and tree-based machines using cross-validation, which is nested within the training set. Under the framework of linear models, prediction error can be computed analytically from the hat matrix, without explicit need for cross-validation (Xu 2017).

3) Success. Success is for models where decisions follow clear rules that characterize success. At the observation level, success is a Bernoulli process that corresponds to whether predefined criteria were achieved. Selection itself is an example of a measure of success, as genotypes are either selected or not. At the population level, success is a Binomial process as it measures the number of successes within a constrained number of attempts. When there are various classes of success, accuracy is measured from a *confusion matrix* (i.e., predicted vs. observed classes), as the percentage of correctly classified observations. Conversely, a metric such as the Jaccard score suits success when it is defined as the product of multiple independent criteria: it defines 1 if all criteria are met, and 0 otherwise. When only two decisions are possible, both accuracy and Jaccard score converge to the same metric. For example, selection accuracy that is defined as the intersection between the predicted top *X*% and observed top *X*% (Qiao et al 2000).

By adequate choice of both scheme and metric in a cross-validation, it can be tailored to the specific application. Guidelines to benchmark genomic prediction methods targeting additive genetics were formalized by Daetwyler et al. (2013). The prediction scenarios were summarized by Crossa et al. (2017) into four types based on their application to plant breeding: Untested genotypes and unobserved environments (CV00), tested genotypes in unobserved environments (CV0), untested genotypes in observed environments (CV1), and tested genotypes in observed environments (CV2).

Information

Cross-validation studies also provide an insight into population stratification, because validations are performed within and across sub-populations. Werner et al. (2020) presents pitfalls and discrepancies in predictability when validations are performed across-families or are nested within-family. When factors that define population structure are known, such as family or sub-population structure, cross-validations enable studying predictability on multiple levels (Xavier and Rainey 2020). The theory surrounding the information captured by molecular markers was formalized by Habier et al. (2013), and the expected genetic information captured under different validation scenarios can be summarized as follows:

Single-family: Linkage.

Within-family: Linkage and LD.

Across-family: Pedigree relationships, Linkage and LD.

Leave-family-out: Pedigree relationships and LD.

The three sources of information are summarized as follows: Pedigree relationship refers to population structure; linkage disequilibrium (LD) means associations between markers and QTL as measured across unrelated founder individuals; and Linkage, or co-segregation, captures inheritance of parental haplotypes.

There are practical implications that have been derived from understanding the sources of genomic information. For example, validation structure information postulates that within-family predictions must yield lower predictability than across-families, because within-family predictions do not benefit from pedigree relationships (Legarra et al. 2008). In the machine learning literature, the discrepancy in accuracy reported across-populations and within-populations is referred to as the Simpson paradox (Fabris and Freitas 2000). The various prediction models are likely to differ with regards to how well they capture the different sources of information. In addition, the nature of the information can change according to the parameterization. Molecular markers can be informative of additive, dominance, and epistasic effects. It was noted by Pérez-Rodríguez et al. (2012) and Howard et al. (2014) that non-parametric methods may be able to capture non-additive effects, i.e., dominance and epistasis, even when the markers are not explicitly coded to capture these types of variation. Conversely, in scenarios where the genetic architecture is predominantly additive, semi-parametric methods (e.g., DNN, RKHS, SVR) may fail to provide an improvement in accuracy over simpler linear methods (Montesinos-López et al. 2021).

Case of study

Different types of cross-validations are illustrated in Figure 1. The validation set was split into tested and untested environments to provide an insight into the non-predictable component of genotype-by-environment interactions. Cross-validation was performed on soybean yield from the SoyNAM dataset, which contains 40 families, 5349 individuals, and 4408 SNPs. Model were trained with data collected in 2012 and validated on data collected in 2012 (tested environment) and 2013 (untested environment), using individual not included in the training set.

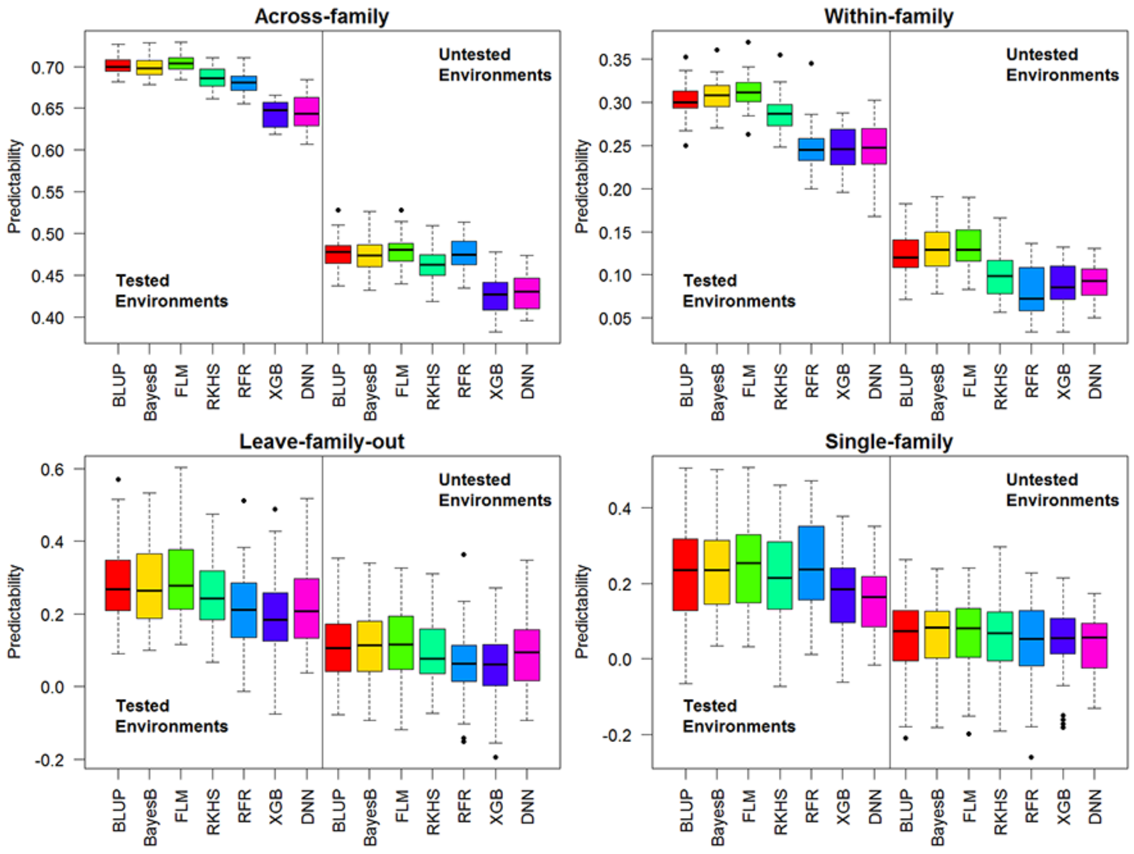

Figure 1. Four cross-validation schemes illustrating predictability of various methods utilized for genomic prediction. Grain yield models from the SoyNAM population, validated upon unobserved individuals from tested and untested environments.

The genomic prediction models were REML-based ridge regression (BLUP), fast Laplace model 'FLM' (Xavier 2019), reproducing kernel Hilbert spaces 'RKHS' (de los Campos et al. 2010), BayesB (Meuwissen et al. 2001), random forest regression (RFR), extreme gradient boosting machine (XGB), and a two hidden-layers deep neural network (DNN). Data is available in the R package SoyNAM (Xavier et al. 2021) and the models are implemented in the R packages bWGR (Xavier et al. 2019), BGLR (Pérez and de los Campos 2014), ranger (Wright and Ziegler 2015), xgboost (Chen and Guestrin 2016) and keras (Arnold 2017). Validations for single-family, within-family, and across-family were performed as 5-fold random cross validation that was repeated 20 times. Results illustrate that the across-family predictability, as the Pearson correlations between predicted and observed values, is considerably higher than predictability within-family across all methods, because the pedigree relationships among parents is captured (Wientjes et al. 2015). Within-family predictions are more accurate than predictions within a single-family, because the former captures linkage disequilibrium in addition to linkage. In all scenarios, predictability is lower when validations are performed over untested environments.

CONCLUSIONS

Plant breeding can benefit greatly from data-driven solutions that identify and select superior germplasm through machine learning, especially with the ever-increasing amount of phenotypic and genotypic data collected with high-throughput technologies. This manuscript provides an overview of the implementation of major supervised machine learning algorithms. The prediction models are explored using various cross-validation approaches, which can be designed to be informative of the strengths and weakness of different models.

ACKNOWLEDGEMENTS

David Habier is acknowledged for the extensive and thorough revision of the manuscript.

REFERENCES

Arnold TB (2017) kerasr: R interface to the keras deep learning library. **Journal of Open Source Software 2**: 296.

Baldi P and Sadowski P (2014) The dropout learning algorithm. **Artificial Intelligence 210**: 78-122.

Botta V, Louppe G, Geurts P and Wehenkel L (2014) Exploiting snp correlations within random forest for genome-wide association studies. **PloS One 9**: e93379.

Breiman L (2001) Random forests. **Machine Learning 45**: 5-32.

Breiman L, Friedman JH, Stone CJ and Olshen RA (1984) **Classification and regression trees**. Chapman and Hall/CRC, Boca Raton, 368p.

Cai T, Liu W and Luo X (2011) A constrained 1 minimization approach to sparse precision matrix estimation. **Journal of the American Statistical Association 106**: 594-607.

Chen X and Ishwaran H (2012) Random forests for genomic data analysis. **Genomics 99**: 323-329.

Corbeil RR and Searle SR (1976) Restricted maximum likelihood (REML) estimation of variance components in the mixed model. **Technometrics 18**:31-38.

Costa-Neto G, Fritsche-Neto R and Crossa J (2021) Nonlinear kernels, dominance, and envirotypingdata increase the accuracy of genome-based prediction in multi-environment trials. **Heredity 126**: 92-106.

Crossa J, Pérez-Rodríguez P, Cuevas J, Montesinos-López O, Jarquín D, de los Campos G, Burgueño J, González-Camacho JM, Pérez-Elizalde S, Beyene Y, Dreisigacker S, Singh R, Zhang X, Gowda M, Roorkiwal M, Rutkoski J and Varshney RK (2017) Genomic selection in plant breeding: methods, models, and perspectives. **Trends in Plant Science 22**: 961-975.

Cuevas J, Montesinos-López O, Juliana P, Guzmán C, Pérez-Rodríguez P, González-Bucio J, Burgueño J, Montesinos-López A and Crossa J (2019) Deep kernel for genomic and near infrared predictions in multi-environment breeding trials. **G3: Genes, Genomes, Genetics 9**: 2913-2924.

Daetwyler HD, Calus MPL, Pong-Wong R, de Los Campos G and Hickey JM (2013) Genomic prediction in animals and plants: simulation of data, validation, reporting, and benchmarking. **Genetics 193**: 347-365.

de Los Campos G, Gianola D, Rosa GJM, Weigel KA and Crossa J (2010) Semi-parametric genomic-enabled prediction of genetic values using reproducing kernel hilbert spaces methods. **Genetics Research 92**: 295-308.

Deisenroth MP, Faisal AA and Ong CS (2020) **Mathematics for machine learning**. Cambridge University Press, New York, 417p.

Fabris CC and Freitas AA (2000) Discovering surprising patterns by detecting occurrences of simpson's paradox. In Bramer M, Macintosh A and Coenen F (eds) **Research and Development in Intelligent Systems XVI**. Springer, London, p. 148-160.

Freund Y and Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application toboosting. **Journal of Computer and System Sciences 55**: 119-139.

Friedman JH (2001) Greedy function approximation: a gradient boosting machine. **The Annals of Statistics 29**: 1189-1232.

Geary, RC (1935) The ratio of the mean deviation to the standard deviation as a test of normality. **Biometrika 27**: 310-332.

González-Camacho JM, de Los Campos G, Pérez P, Gianola D, Cairns JE, Mahuku G, Babu R and Crossa J (2012) Genome-enabled prediction of genetic values using radial basis function neural networks. **Theoretical and Applied Genetics 125**: 759-771.

Goodfellow I, Bengio Y, Courville A and Bengio Y (2016) **Deep learning**. Volume 1, MIT Press, Cambridge, 800p.

Habier D, Fernando RL and Garrick DJ (2013) Genomic BLUP decoded: a look into the black box of genomic prediction. **Genetics 194**: 597-607.

Hahn S and Choi H (2020) Understanding dropout as an optimization trick. **Neurocomputing 398**: 64-70.

Hanin B (2019) Universal function approximation by deep neural nets with bounded width and relu activations. **Mathematics 7**: 992.

Hastie T, Tibshirani R and Friedman J (2009) **The elements of statistical learning: data mining, inference, and prediction**. Springer Science & Business Media, New York, 744p.

Henderson CR (1975) Inverse of a matrix of relationships due to sires and maternal grandsires. **Journal of Dairy Science 58**: 1917-1921.

Heslot N, Yang H-P, Sorrells ME and Jannink J-L (2012) Genomic selection in plant breeding: acomparison of models. **Crop Science 52**: 146-160.

Hoerl AE and Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. **Technometrics 12**: 55-67.

Howard R, Carriquiry AL and Beavis WD (2014) Parametric and nonparametric statistical methods for genomic selection of traits with additive and epistatic genetic architectures. **G3: Genes, Genomes, Genetics 4**: 1027-1046.

Izenman AJ (2008) **Modern multivariate statistical techniques**. Regression, Classification and Manifold Learning. Springer, New York, 733p.

Kingma DP and Adam B (2014) A method for stochastic optimization. **ArXiv preprint ArXiv**: 1412.6980.

Legarra A and Misztal I (2008) Computing strategies in genome-wide selection. **Journal of Dairy Science 91**: 360-366.

Legarra A, Robert-Granié C, Manfredi E and Elsen J-M (2008) Performance of genomic selection in mice. **Genetics 180**: 611-618.

Lehermeier C, Wimmer V, Albrecht T, Auinger H-J, Gianola D, Schmid VJ and Schön C-C (2013) Sensitivity to prior specification in Bayesian genome-based prediction models.**Statistical Applications in Genetics and Molecular Biology 12**: 375-391.

Li Y and Arce GR (2004) A maximum likelihood approach to least absolute deviation regression. **EURASIP Journal on Advances in Signal Processing 2004**: 1-8.

Lin E and Lane H-Y (2017) Machine learning and systems genomics approaches for multi-omics data. **Biomarker Research 5**: 1-6.

Liu W and Luo X (2015) Fast and adaptive sparse precision matrix estimation in high dimensions. **Journal of Multivariate Analysis 135**: 153-162.

Meuwissen THE, Hayes BJ, and Goddard ME (2001). Prediction of total genetic value using genome-wide dense marker maps. **Genetics** 157:1819-1829.

Misztal I and Legarra A (2017) Invited review: efficient computation strategies in genomic selection. **Animal 11**: 731-736.

Montesinos-López OA, Montesinos-López A, Pérez-Rodríguez P, Barrón-López JA, Martini JWR, Fajardo-Flores SB, Gaytan-Lugo LS, Santana-Mancilla PCand Crossa J (2021) A review of deep learning applications for genomic selection. **BMC Genomics 22**: 1-23.

Perakakis N, Yazdani A, Karniadakis GE and Mantzoros C (2018) Omics, big data and machine learning as tools to propel understanding of biological mechanisms and to discover novel diagnostics and therapeutics. **Metabolism-Clinical and Experimental 87**: A1-A9.

Pérez P and de Los Campos G (2014) Genome-wide regression and prediction with the BGLR statisticalpackage. **Genetics 198**: 483-495.

Pérez-Rodríguez P, Gianola D, González-Camacho JM, Crossa J, Manès Y and Dreisigacker S (2012) Comparison between linear and non-parametric regression models for genome-enabled prediction in wheat. **G3: Genes, Genomes, Genetics 2**: 1595-1605.

Pocrnic I, Lourenco DAL, Masuda Y, Legarra A and Misztal I (2018) Limited dimensionality of genomic informationand effective population size. In **11th Proceedings of the world congress on genetics applied to livestock production (WCGALP)**. hal-02735484.

Pszczola M and Calus MPL (2016) Updating the reference population to achieve constant genomic prediction reliability across generations. **Animal 10**: 1018-1024.

Qiao CG, Basford KE, DeLacy IH and Cooper M (2000) Evaluation of experimental designs and spatial analyses in wheat breeding trials. **Theoretical and Applied Genetics 100**: 9-16.

Rue H, Riebler A, Sørbye SH, Illian JB, Simpson DP, Lindgren FK (2017). Bayesian computing with INLA: a review. **Annual Review of Statistics and Its Application** 7:395-421.

Runcie D and Cheng H (2019) Pitfalls and remedies for cross validation with multi-trait genomic prediction methods. **G3: Genes, Genomes, Genetics 9**: 3727-3741.

Schaeffer LR (1986) Pseudo expectation approach to variance component estimation. **Journal of Dairy Science**. 69:2884-2889.

Schwarz DF, König IR and Ziegler (2010) On safari to random jungle: a fast implementation of random forests for high-dimensional data. **Bioinformatics 26**: 1752-1758.

Searle SR, Casella G, McCulloch CE (2009) **Variance components**. John Wiley & Sons.

Shahhosseini M, Hu G, Huber I and Archontoulis SV (2021) Coupling machine learning andcrop modeling improves crop yield prediction in the US corn belt. **Scientific Reports 11**: 1-15.

Shalev-Shwartz S and Ben-David S (2014) **Understanding machine learning: From theory to algorithms**. Cambridge University Press, New York, 449p.

Signoretto M and Suykens JAK (2015) **Kernel methods**. Springer, Heidelberg, p. 577-605.

Sorensen D and Gianola D (2007). **Likelihood, Bayesian, and MCMC methods in quantitative genetics**. Springer Science & Business Media.

Stigler SM (1981) Gauss and the invention of least squares. **The Annals of Statistics 9**: 465-474.

Stranden I and Garrick DJ (2009) Derivation of equivalent computing algorithms for genomic predictions and reliabilities of animal merit. **Journal of dairy science 92**: 2971-2975.

Takase T (2021) Dynamic batch size tuning based on stopping criterion for neural network training. **Neurocomputing 429**: 1-11.

Tibshirani R (1996) Regression shrinkage and selection via the lasso. **Journal of the Royal Statistical Society: Series B (Methodological) 58**: 267-288.

Van Raden PM and Jung YC (1988) A general purpose approximation to restricted maximum likelihood: the tilde-hat approach. **Journal of Dairy Science**. 71:187-194.

VanRaden PM (2008) Efficient methods to compute genomic predictions. **Journal of Dairy Science 91**: 4414-4423.

Werner CR, Gaynor RC, Gorjanc G, Hickey JM, Kox T, Abbadi A, Leckband G, Snowdon RJ and Stahl A (2020) How population structure impacts genomic selection accuracy in cross-validation: Implications for practical breeding. **Frontiers in Plant Science 11**: 2028.

Wientjes YCJ, Veerkamp RF, Bijma P, Bovenhuis H, Schrooten C and Calus MPL (2015) Empirical and deterministic accuracies of across-population genomic prediction. **Genetics Selection Evolution 47**: 1-14.

Wright MN and Ziegler A (2015) ranger: A fast implementation of random forests for high dimensional data in C++ and R. **arXiv***:1508.04409*.

Xavier A (2019) Efficient estimation of marker effects in plant breeding. **G3: Genes, Genomes, Genetics 9**: 3855-3866.

Xavier A and Rainey KM (2020) Quantitative genomic dissection of soybean yield components. **G3: Genes, Genomes, Genetics 10**: 665-675.

Xavier A, Beavis W, Specht J, Diers B, Mian R, Howard R, Graef G, Nelson R, Schapaugh W, Wang D, Shannon G, McHale L, Cregan P, Song Q, Lopez M, Muir W, Rainey K (2021) Soynam: Soybean nested association mapping dataset. R package version 1.6.1.

Xavier A, Muir WM and Rainey KM (2019) bWGR: Bayesian whole-genome regression. **Bioinformatics 36**: 1957-1959.

Xavier A, Muir WM, Craig B and Rainey KM (2016) Walking through the statistical black boxes of plant breeding. **Theoretical and Applied Genetics 129**: 1933-1949.

Xu D, Zhang S, Zhang H and Mandic DP (2021) Convergence of the rmsprop deep learning method with penalty for nonconvex optimization. **Neural Networks 139**: 17-23.

Xu S (2013) Mapping quantitative trait loci by controlling polygenic background effects. **Genetics 195**: 1209-1222.

Xu S (2017) Predicted residual error sum of squares of mixed models: an application for genomic prediction. **G3: Genes, Genomes, Genetics 7**: 895-909.

Zampieri G, Vijayakumar S, Yaneske E and Angione C (2019) Machine and deep learning meet genome-scale metabolic modeling. **PLoS Computational Biology 15**: e1007084.

Zou H and Hastie T (2005) Regularization and variable selection via the elastic net. **Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67**: 301-320.