Supplemental Materials

1. Computational Methods Overview

Below, we provide general overview of the genomic prediction models that we applied in this work.

I. **Tree-based Methods** are a way to combine several models to improve the predictions with a response variable. It has 3 phases such as generation, pruning, and the integration phase [52].

1. **RF_py:** It is the random forest ensemble of random regression trees implemented by the RandomForestRegressor method [53]. The outputs of the base regression models are averaged to get the model output. The hyper-parameters are 'n_estimators', the number of the trees in the forest, 'max_depth' is the maximum depth of the tree set, and 'max_features' is the parameter to select the number of features to have a best fit.
2. **extraTrees_py:** It is an ensemble of extremely randomized regression trees [54] implemented by the ExtraTreesRegressor method from sklearn library. The only hyper-parameters are 'n_estimators' and 'min_samples_split', minimum number of samples required to split an internal node.
3. **GB_py:** It combines weak base learners into a strong ensemble by iteratively adding base learners in a forward stage-wise fashion, and in each iteration the new model is trained to fit the residual of the

previous ensemble. It optimizes arbitrary differentiable loss functions defining the fitting criteria [55]. The sklearn.ensemble python library provides regression method of Gradient Boosting. The hyper-parameters 'learning_rate', 'max_features', and 'max_depth' are tuned in the study.

4. **adaboost_py:** It is adaptive boosting meta-estimator [56] which first fits on the original dataset and then weight of instances are tuned according to error of the current outcome on additional copies of the regressor on the same dataset. The hyper-parameter 'n_estimators' is the number of estimators at which boosting is terminated and the 'learning_rate' is also tuned and in case of good fit, the learning stops early.
5. **Xgboost_py**: It is an extreme gradient boosting [55], implemented using the scikit-learn API for XGBoost regression. The 'learning_rate', 'max_depth', 'min_child_weight', γ, and 'colsample_bytree' are tuning parameters.

II. **Deep Learning**

1. **mlp_py:** Multilayer perceptrons (mlp) is class of feedforward artificial neural network and usually mean fully connected networks and that is each neuron in one layer is connected to all neurons in the next layer. The MLP consists at-least 3 layers; input, hidden and output layer. Formally, one hidden layer mlp regression function is

represented as follows:

$$g(x) = b + Wtanh(c + Vx) \tag{1}$$

Where, x is the input p-vector, V is a $n \times p$ matrix, weights from input to hidden, c is a n-vector, known as hidden unit biases, b is an q-vector, known as output units biases, and W is an $q \times h$ matrix, weights from hidden to output. In the current study, we developed a mlp_py model by two hidden layers with (32, 16) number of nodes and (relu, relu) activation function. The function is a mathematical equation that placed on each neuron and it decides whether that neuron has to activate or not, it basically determines where inputs of neuron are significant for the prediction. We tuned batch_size and epochs parameters for the model training and the performance is calculated using R and optimizer is 'adam'. The Adam stands for Adaptive Moment Estimation and it keeps an exponentially decaying average of past gradients, similar to momentum.

III. **Penalized Linear Regression**

1. **LASSO_r**: Least Absolute Shrinkage and Selection Operator (LASSO) is one the most famous parametric linear regression model, implemented using sklearn Python module and trained with l_1 regularization [57, 58]. The LASSO adds 'absolute value of magnitude' of coefficient as penalty term to the loss function. Tuning parameter

is λ.

$$\hat{\beta}_{LASSO} := \arg\min_{\beta}\{\frac{1}{n}||Y - X\beta||_2^2 + \lambda||\beta||_1\} \tag{2}$$

In the equation 4, $||\beta||_1$ is a 1-norm of the coefficient vector β and $\lambda||\beta||_1$ is known as l_1 penalty.

2. **ridge_r**: It is linear model, implemented using sklearn library and combined linear least squares with l_2 regularization [57]. It imposes l_2 penalty on the size of the coefficients in order to address some of the problems of ordinary least squares method. The ridge coefficients minimize a penalized residual sum of squares. The hyper-parameter λ is tuned.

$$\hat{\beta}_{Ridge} := \arg\min_{\beta}\{\frac{1}{n}||Y - X\beta||_2^2 + \lambda||\beta||_2^2\} \tag{3}$$

In the equation 3, $||\beta||_2^2$ is the squared 2-norm of the coefficient vector β and l_2 penalty since it is based on l_2 norm of the parameter.

3. **elasticnet_r**: It is a linear model implemented using sklearn module, combined l_1 and l_2-norm regularization of the coefficients [57]. The parameter α is ElasticNet mixing parameters. It is known as l_2 penalty, if it is 1 then it is l_1 penalty and if it is between 0 and 1 then penalty is a combination of l_1 and l_2. Along with it, λ value is also tuned.

$$\hat{\beta}_{ElasticNet} := \arg\min_{\beta}\{\frac{1}{n}||Y - X\beta||_2^2 + \lambda_2||\beta||^2 + \lambda_1||\beta||_1\} \tag{4}$$

IV. **Bayesian Regression**

The equation 5 represents simple linear function of GP for wheat GY trait y:

$$y = X\beta + Zm + e; \tag{5}$$

Where, y is response variable and it is the vector of phenotype, X and Z are incidence matrices i.e. SNPs, β and m are coefficients of SNPs and e is residual term. ML parametric techniques make assumption about the function and these models are faster than non-parametric models. It requires less amount of data for model fitting, gives less complexity however, it might not be as powerful as non-parametric model. Here, we implemented three classical methods; BayesA_r [1], BRR_r, and BL_r [59, 60] using BGLR R package [36]. Each model has tuned the two metaparamters; $nIter$ with (10000, 9000, and 8000) for the (dataset1, dataset2, and dataset3) and $burnIn$ with 3000 for three datasets, meaning the number of iterations, made in the Gibbs sampler and used as burn-in respectively.

2. Supplemental Tables

Table S1: Test (outside the brackets) and train (in brackets) correlation (R) scores for dataset1. Scores are shown for the 12 prediction models per feature selection method.

Test and Train Results of the dataset 1 (PUL17_F5)				
Model	VT	MI	Corr_Matrix	BayesA
RF_py	0.46(0.46)	0.47(0.52)	0.43(0.48)	0.45(0.53)
extraTrees_py	0.43(0.37)	0.39(0.43)	0.32(0.41)	0.43(0.44)
GB_py	0.48(0.43)	0.51(0.48)	0.40(0.48)	0.43(0.50)
adaboost_py	0.46(0.41)	0.45(0.45)	0.40(0.43)	0.45(0.45)
Xgboost_py	0.50(0.45)	0.44(0.47)	0.42(0.44)	0.49(0.48)
mlp_py	0.35(0.33)	0.42(0.33)	0.42(0.34)	0.42(0.61)
LASSO_r	0.42(0.37)	0.42(0.41)	0.34(0.32)	0.36(0.38)
ridge_r	0.49(0.41)	0.48(0.50)	0.44(0.48)	0.50(0.75)
elasticnet_r	0.42(0.39)	0.44(0.39)	0.35(0.29)	0.50(0.57)
BayesA_r	0.47(0.33)	0.46(0.50)	0.44(0.32)	0.50(0.75)
BRR_r	0.46(0.32)	0.46(0.50)	0.44(0.30)	0.50(0.75)
BL_r	0.46(0.31)	0.45(0.49)	0.43(0.32)	0.50(0.74)

VT = variance threshold, Corr_Matrix = correlation matrix, MI = mutual information. Score in bold indicates the best model on the corresponding dataset. Each model name is denoted by the method name and programming platform in which they were implemented.

Table S2: Test (outside the brackets) and train (in brackets) correlation (R) scores for dataset2. Scores are shown for the 12 prediction models per feature selection method.

Test and Train Results of the dataset2 (PUL18_DH)				
Model	VT	MI	Corr_Matrix	BayesA
RF_py	0.58(0.64)	0.62(0.66)	0.67(0.63)	0.68(0.67)
extraTrees_py	0.55(0.55)	0.49(0.61)	0.67(0.60)	0.67(0.63)
GB_py	0.58(0.63)	0.63(0.66)	0.61(0.57)	0.72(0.68)
adaboost_py	0.51(0.60)	0.53(0.62)	0.59(0.59)	0.63(0.59)
Xgboost_py	0.55(0.63)	0.59(0.65)	0.66(0.62)	0.67(0.67)
mlp_py	0.53(0.54)	0.52(0.58)	0.53(0.50)	0.80(0.59)
LASSO_r	0.55(0.56)	0.57(0.53)	0.58(0.51)	0.55(0.54)
ridge_r	0.56(0.58)	0.57(0.61)	0.59(0.51)	0.65(0.78)
elasticnet_r	0.54(0.57)	0.58(0.56)	0.58(0.53)	0.65(0.66)
BayesA_r	0.56(0.65)	0.56(0.62)	0.60(0.62)	0.63(0.83)
BRR_r	0.56(0.65)	0.55(0.62)	0.57(0.60)	0.63(0.84)
BL_r	0.56(0.65)	0.56(0.62)	0.60(0.62)	0.63(0.84)

VT = variance threshold, Corr_Matrix = correlation matrix, MI = mutual information. Score in bold indicates the best model on the corresponding dataset. Each model name is denoted by the method name and programming platform in which they were implemented.

Table S3: Test (outside the brackets) and train (in brackets) correlation (R) scores for dataset3. Scores are shown for the 12 prediction models per feature selection method.

Test and Train Results of the dataset 3 (LND18_DH)				
Model	VT	MI	Corr_Matrix	BayesA
RF_py	0.45(0.42)	0.42(0.44)	0.46(0.34)	0.40(0.47)
extraTrees_py	0.32(0.34)	0.37(0.35)	0.38(0.31)	0.39(0.35)
GB_py	0.42(0.42)	0.44(0.40)	0.46(0.33)	0.32(0.42)
adaboost_py	0.40(0.40)	0.32(0.41)	0.32(0.31)	0.42(0.33)
Xgboost_py	0.37(0.41)	0.40(0.44)	0.35(0.31)	0.46(0.30)
mlp_py	0.38(0.32)	0.26(0.35)	0.45(0.30)	0.59(0.37)
LASSO_r	0.35(0.23)	0.34(0.29)	0.20(0.23)	0.22(0.30)
ridge_r	0.38(0.32)	0.41(0.35)	0.42(0.32)	0.41(0.62)
elasticnet_r	0.37(0.29)	0.34(0.30)	0.22(0.18)	0.41(0.46)
BayesA_r	0.41(0.53)	0.42(0.55)	0.44(0.44)	0.42(0.55)
BRR_r	0.42(0.51)	0.43(0.53)	0.45(0.44)	0.42(0.57)
BL_r	0.41(0.58)	0.41(0.54)	0.44(0.46)	0.42(0.56)

VT = variance threshold, Corr_Matrix = correlation matrix, MI = mutual information. Score in bold indicates the best model on the corresponding dataset. Each model name is denoted by the method name and programming platform in which they were implemented.

3. Supplemental Figures

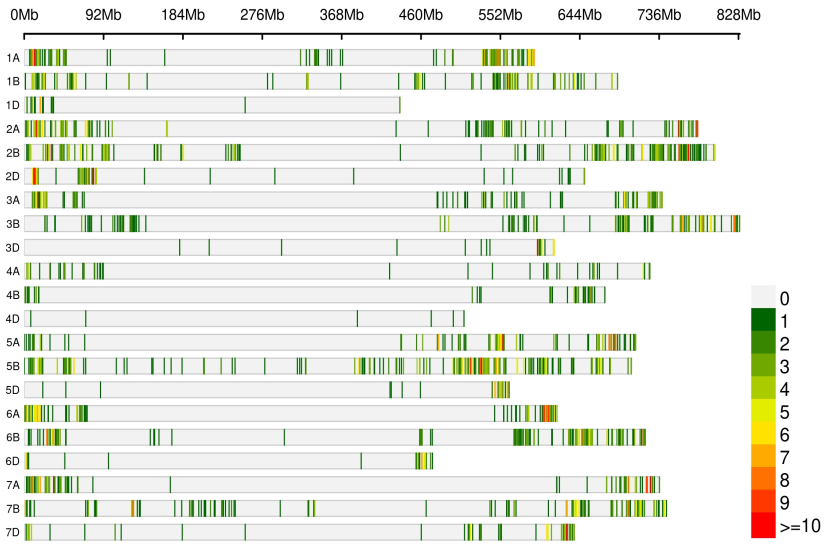

Figure S1: Chromosome-wide SNP density plot of the SNPs selected in dataset1 by variance threshold. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

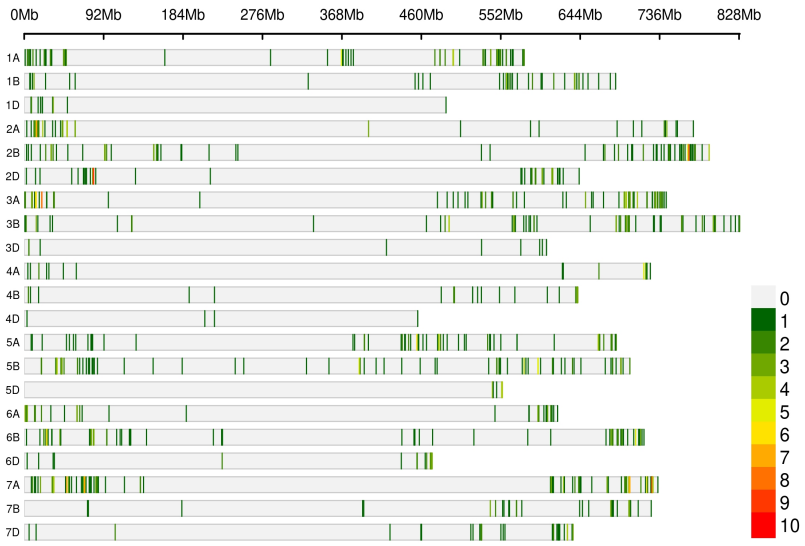

Figure S2: Chromosome-wide SNP density plot of the SNPs selected in dataset1 by mutual information. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

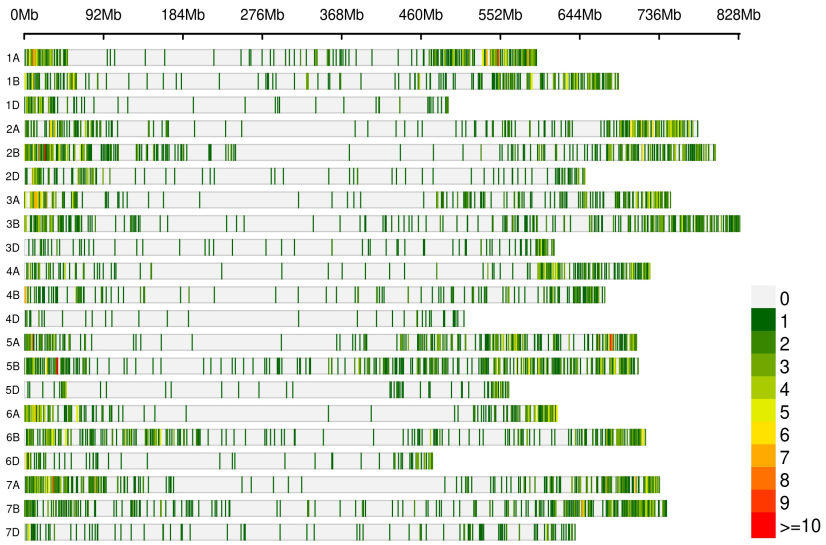

Figure S3: Chromosome-wide SNP density plot of the SNPs selected in dataset1 by Corr_Matrix. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

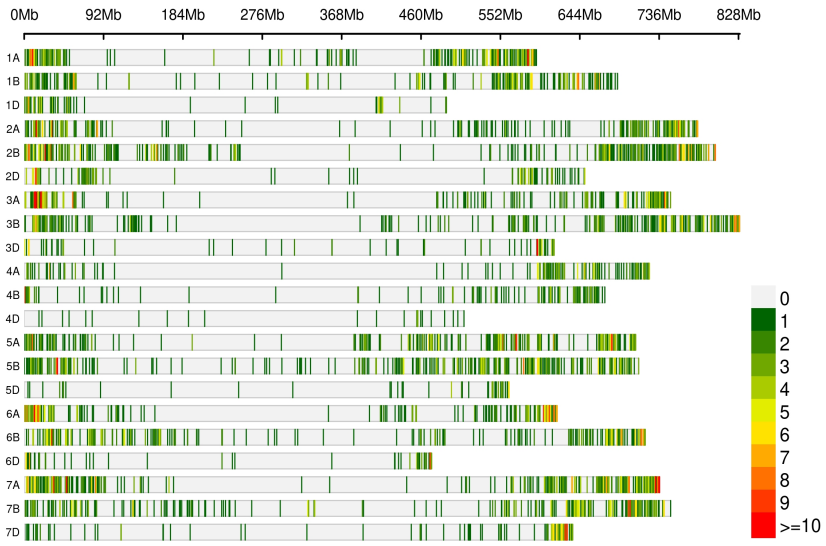

Figure S4: Chromosome-wide SNP density plot of the SNPs selected in dataset1 by BayesA. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

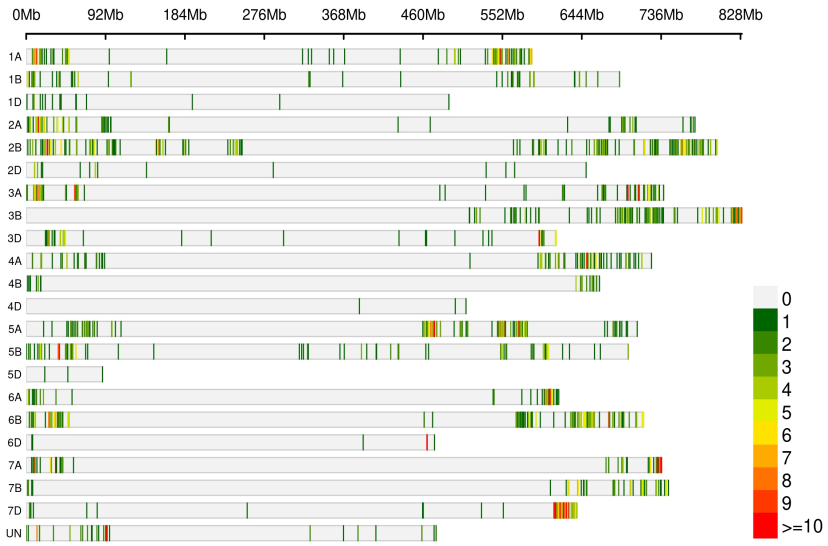

Figure S5: Chromosome-wide SNP density plot of the SNPs selected in dataset2 by variance threshold. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

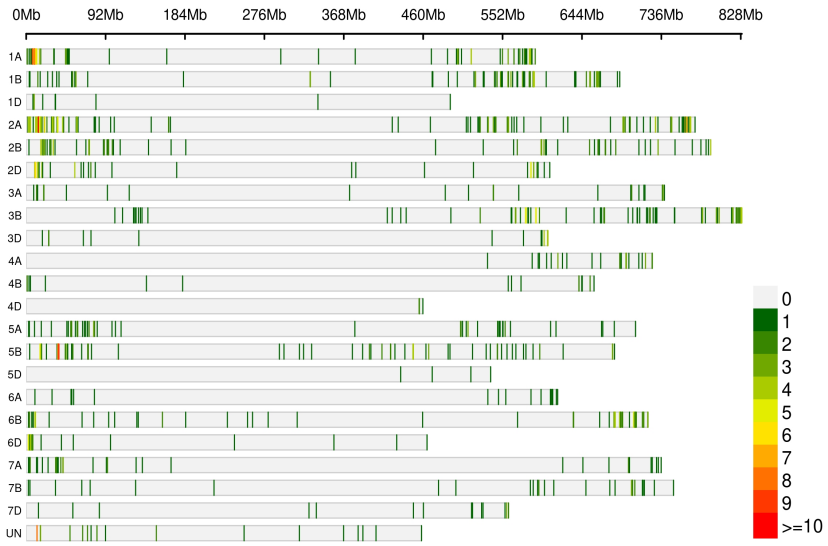

Figure S6: Chromosome-wide SNP density plot of the SNPs selected in dataset2 by mutual information. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

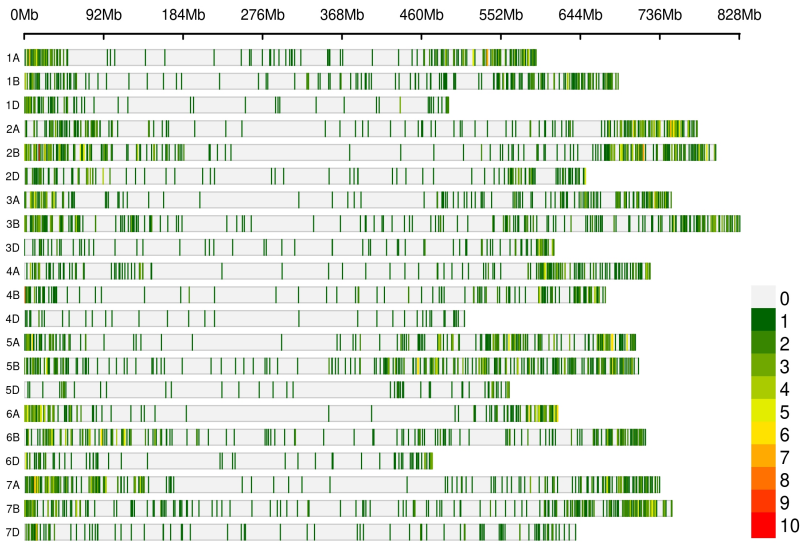

Figure S7: Chromosome-wide SNP density plot of the SNPs selected in dataset2 by Corr_Matrix. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

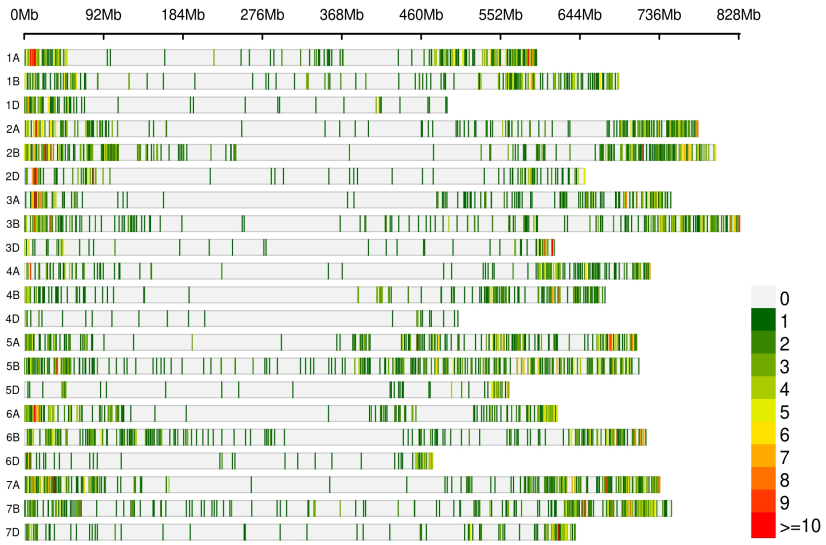

Figure S8: Chromosome-wide SNP density plot of the SNPs selected in dataset2 by BayesA. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

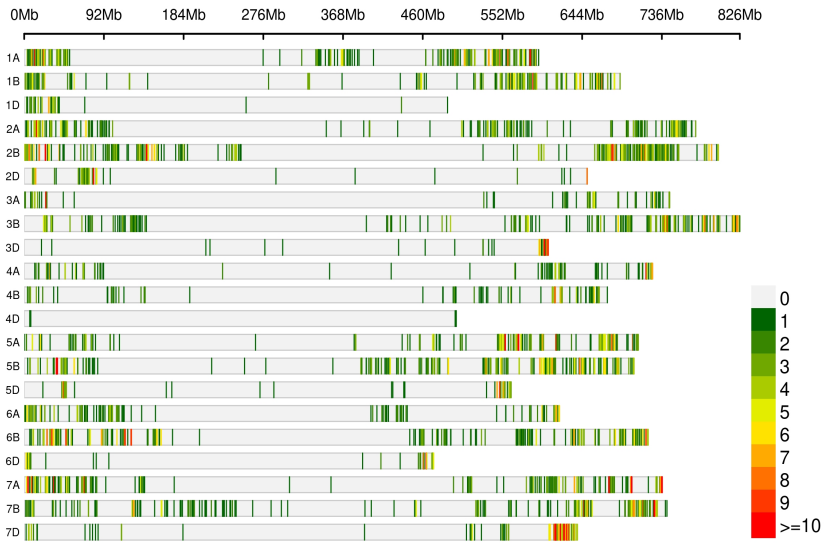

Figure S9: Chromosome-wide SNP density plot of the SNPs selected in dataset3 by variance threshold. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

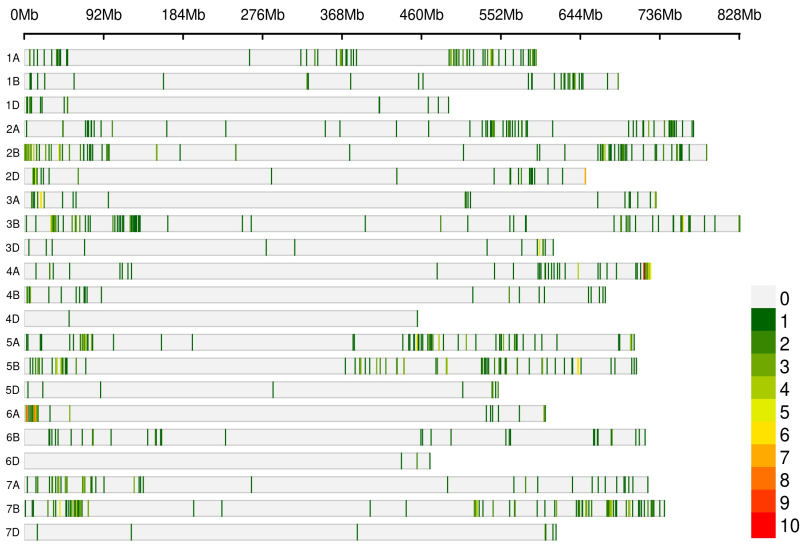

Figure S10: Chromosome-wide SNP density plot of the SNPs selected in dataset3 by mutual information. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

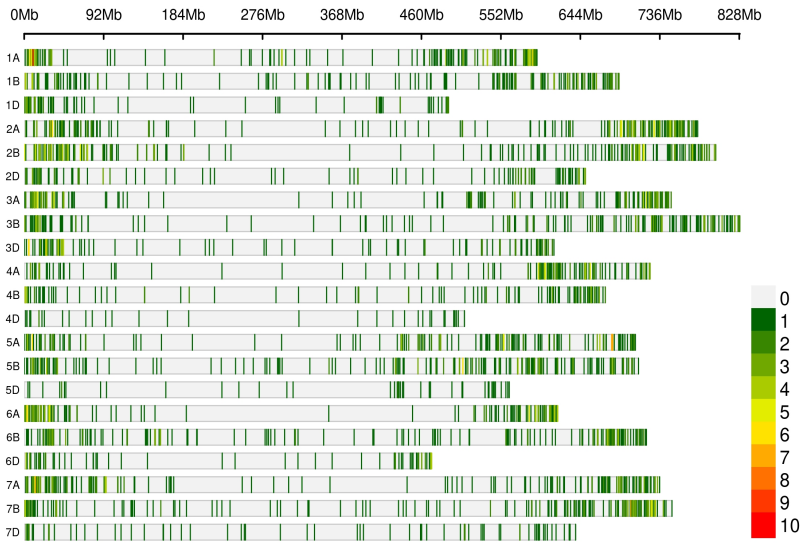

Figure S11: Chromosome-wide SNP density plot of the SNPs selected in dataset3 by Corr_Matrix. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

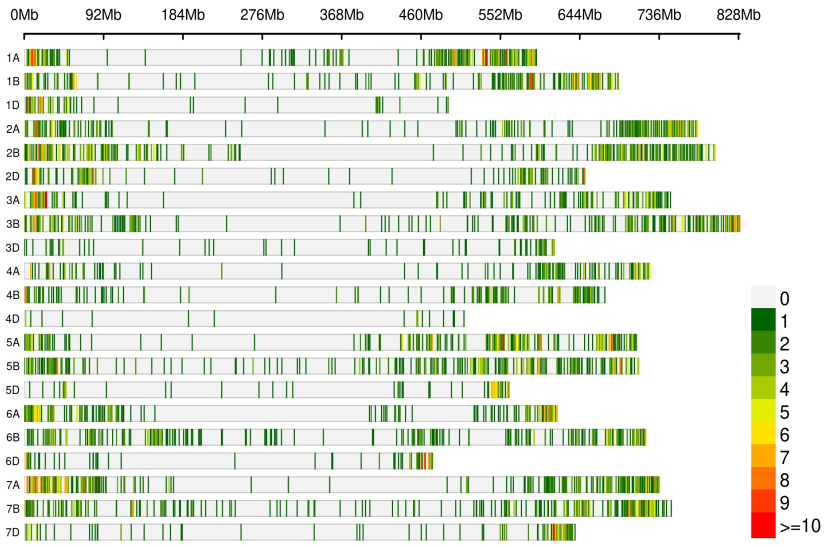

Figure S12: Chromosome-wide SNP density plot of the SNPs selected in dataset3 by BayesA. Window size is set to 1 Mb. In the legend, bin range is between 0 and 10, meaning the bin whose SNP number is smaller or bigger than the range will be used the same color [61].

References

[1] Hayes, B.; Goddard, M.; Goddard, M. Prediction of total genetic value using genome-wide dense marker maps. *Genetics* **2001**, *157*, 1819–1829. https://doi.org/10.1093/genetics/157.4.1819.

[2] Bernardo, R. Molecular markers and selection for complex traits in plants: learning from the last 20 years. *Crop Sci.* **2008**, *48*, 1649–1664. https://doi.org/10.2135/cropsci2008.03.0131.

[3] Scheben, A.; Yuan, Y.; Edwards, D. Advances in genomics for adapting crops to climate change. *Curr. Plant Biol.* **2016**, *6*, 2–10. https://doi.org/10.1016/j.cpb.2016.09.001.

[4] Xu, Y.; Liu, X.; Fu, J.; Wang, H.; Wang, J.; Huang, C.; Prasanna, B.M.; Olsen, M.S.; Wang, G.; Zhang, A. Enhancing genetic gain through genomic selection: From livestock to plants. *Plant Commun.* **2020**, *1*, 100005. https://doi.org/10.1016/j.xplc.2019.100005.

[5] González-Camacho, J.M.; Ornella, L.; Pérez-Rodríguez, P.; Gianola, D.; Dreisigacker, S.; Crossa, J. Applications of machine learning methods to genomic selection in breeding wheat for rust resistance. *Plant Genome* **2018**, *11*, 170104. https://doi.org/10.3835/plantgenome2017.11.0104.

[6] Sandhu, K.S.; Patil, S.S.; Aoun, M.; Carter, A.H. Multi-Trait Multi-Environment Genomic Prediction for End-Use Quality Traits in Winter Wheat. *Front. Genet.* **2022**, *13*. https://doi.org/10.3389/fgene.2022.831020.

[7] Farooq, M.; van Dijk, A.D.; Nijveen, H.; Mansoor, S.; de Ridder, D. Genomic prediction in plants: Opportunities for machine learning-based approaches. *F1000Research* **2022**. https://doi.org/10.12688/f1000research.122437.1.

[8] Crossa, J.; Campos, G.D.L.; Pérez, P.; Gianola, D.; Burgueno, J.;

Araus, J.L.; Makumbi, D.; Singh, R.P.; Dreisigacker, S.; Yan, J.; et al. Prediction of genetic values of quantitative traits in plant breeding using pedigree and molecular markers. *Genetics* **2010**, *186*, 713–724. https://doi.org/10.1534/genetics.110.118521.

[9] Habier, D.; Fernando, R.L.; Kizilkaya, K.; Garrick, D.J. Extension of the bayesian alphabet for genomic selection. *BMC Bioinform.* **2011**, *12*, 1–12. https://doi.org/10.1186/1471-2105-12-186.

[10] Saini, D.K.; Chopra, Y.; Singh, J.; Sandhu, K.S.; Kumar, A.; Bazzer, S.; Srivastava, P. Comprehensive evaluation of mapping complex traits in wheat using genome-wide association studies. *Mol. Breed.* **2022**, *42*, 1–52. https://doi.org/10.1007/s11032-021-01272-7.

[11] Meher, P.K.; Rustgi, S.; Kumar, A. Performance of Bayesian and BLUP alphabets for genomic prediction: analysis, comparison and results. *Heredity* **2022**, *128*, 519–530. https://doi.org/10.1038/s41437-022-00539-9.

[12] Sandhu, K.; Patil, S.S.; Pumphrey, M.; Carter, A. Multitrait machine- and deep-learning models for genomic selection using spectral information in a wheat breeding program. *Plant Genome* **2021**, *14*, e20119. https://doi.org/0.1002/tpg2.20119.

[13] Montesinos-López, O.A.; Gonzalez, H.N.; Montesinos-López, A.; Daza-Torres, M.; Lillemo, M.; Montesinos-López, J.C.; Crossa, J. Comparing

gradient boosting machine and Bayesian threshold BLUP for genome-based prediction of categorical traits in wheat breeding. *Plant Genome* **2022**, e20214. https://doi.org/10.1002/tpg2.20214.

[14] Sandhu, K.S.; Aoun, M.; Morris, C.F.; Carter, A.H. Genomic selection for end-use quality and processing traits in soft white winter wheat breeding program with machine and deep learning models. *Biology* **2021**, *10*, 689. https://doi.org/10.3390/biology10070689.

[15] Sandhu, K.S.; Lozada, D.N.; Zhang, Z.; Pumphrey, M.O.; Carter, A.H. Deep learning for predicting complex traits in spring wheat breeding program. *Front. Plant Sci.* **2021**, *11*, 613325. https://doi.org/10.3389/fpls.2020.613325.

[16] Bellman, R.E. *Adaptive Control Processes: A Guided Tour*; Princeton University Press: Princeton, NJ, USA, 2015; Volume 2045.

[17] Chandrashekar, G.; Sahin, F. A survey on feature selection methods. *Comput. Electr. Eng.* **2014**, *40*, 16–28.

[18] Van Der Maaten, L.; Postma, E.; Van den Herik, J. Dimensionality reduction: A comparative. *J. Mach. Learn Res.* **2009**, *10*, 13.

[19] Jain, R.; Xu, W. HDSI: High dimensional selection with interactions algorithm on feature selection and testing. *PLoS ONE* **2021**, *16*, e0246159. https://doi.org/10.1371/journal.pone.0246159.

[20] Zhou, W.; Bellis, E.S.; Stubblefield, J.; Causey, J.; Qualls, J.; Walker, K.; Huang, X. Minor QTLs mining through the combination of GWAS and machine learning feature selection. *BioRxiv* **2019**, https://doi.org/10.1101/712190.

[21] Azodi, C.B.; Bolger, E.; McCarren, A.; Roantree, M.; de Los Campos, G.; Shiu, S.H. Benchmarking parametric and machine learning models for genomic prediction of complex traits. *G3 Genes Genomes Genet.* **2019**, *9*, 3691–3702. https://doi.org/10.1534/g3.119.400498.

[22] Grinberg, N.F.; Orhobor, O.I.; King, R.D. An evaluation of machine-learning for predicting phenotype: Studies in yeast, rice, and wheat. *Mach. Learn.* **2020**, *109*, 251–277. https://doi.org/10.1007/s10994-019-05848-5.

[23] Le Mouël, C.; Lattre-Gasquet, D.; Mora, O.= *Land Use and Food Security in 2050: A Narrow Road*; éditions Quae: Paris, France, 2018.

[24] Lozada, D.N.; Ward, B.P.; Carter, A.H. Gains through selection for grain yield in a winter wheat breeding program. *PLoS ONE* **2020**, *15*, e0221603.

[25] Pandas—Python Data Analysis Library. Available online: `https://pandas.pydata.org/` (accessed on 2 April 2021).

[26] McKinney, W.; Team, P. Pandas-Powerful python data analysis toolkit.

Pandas—Powerful Python Data Anal Toolkit **2015**, *1625*. Available online: `https://pandas.pydata.org/` (accessed on 2 April 2021).

[27] Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*; O'Reilly Media: Sebastopol, CA, USA, 2019.

[28] Duch, W. Filter methods. In *Feature Extraction*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 89–117. https://doi.org/10.1007/978-3-540-35488-8_4.

[29] Bermingham, M.L.; Pong-Wong, R.; Spiliopoulou, A.; Hayward, C.; Rudan, I.; Campbell, H.; Wright, A.F.; Wilson, J.F.; Agakov, F.; Navarro, P.; et al. Application of high-dimensional feature selection: Evaluation for genomic prediction in man. *Sci. Rep.* **2015**, *5*, 10312. https://doi.org/10.1038/srep10312.

[30] Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.

[31] Variance Threshold Feature Selection Using Sklearn. Available online: `https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.VarianceThreshold.html` (accessed on 22 June 2021).

[32] Plotting a Diagonal Correlation Matrix. Available online: `https://seaborn.pydata.org/examples/many_pairwise_correlations.html` (accessed on 29 June 2021).

[33] Kraskov, A.; Stögbauer, H.; Grassberger, P. Estimating mutual information. *Phys. Rev. E* **2004**, *69*, 066138.

[34] Vergara, J.R.; Estévez, P.A. A review of feature selection methods based on mutual information. *Neural Comput. Appl.* **2014**, *24*, 175–186. https://doi.org/10.1007/s00521-013-1368-0.

[35] O'Hara, R.B.; Sillanpää, M.J. A review of Bayesian variable selection methods: What, how and which. *Bayesian Anal.* **2009**, *4*, 85–117.

[36] Pérez, P.; de los Campos, G. BGLR: A statistical package for whole genome regression and prediction. *Genetics* **2014**, *198*, 483–495. https://doi.org/10.1534/genetics.114.164442.

[37] de los Campos, G.; Pataki, A.; Pérez, P. The BGLR (Bayesian Generalized Linear Regression) R-Package. 2015. Available online: `http://bglr.r-forge.r-project.org/` (accessed on 08 April 2022).

[38] Bengio, Y.; Grandvalet, Y. No unbiased estimator of the variance of k-fold cross-validation. *J. Mach. Learn. Res.* **2004**, *5*, 1089–1105.

[39] Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.

[40] Probst, P.; Boulesteix, A.L.; Bischl, B. Tunability: Importance of hyperparameters of machine learning algorithms. *J. Mach. Learn. Res.* **2019**, *20*, 1934–1965.

[41] Sanner, M.F. Python: A programming language for software integration and development. *J. Mol. Graph Model.* **1999**, *17*, 57–61.

[42] Ihaka, R.; Gentleman, R. R: A language for data analysis and graphics. *J. Comput. Graph. Stat.* **1996**, *5*, 299–314. https://doi.org/10.2307/1390807.

[43] Scikit-Learn Machine Learning in Python. Available online: `https://scikit-learn.org/stable/` (accessed on 15 July 2020).

[44] XGBoost Documentation. Available online: `https://xgboost.readthedocs.io/en/latest/` (accessed on 15 July 2020).

[45] Gulli, A.; Pal, S. *Deep Learning with Keras*; Packt Publishing Ltd.: Birmingham, UK, 2017.

[46] SPSS Tutorials: Pearson Correlation. Available online: `https://libguides.library.kent.edu/SPSS/PearsonCorr` (accessed on 1 July 2020).

[47] Pinheiro, J.; Bates, D.; DebRoy, S.; Sarkar, D.; Heisterkamp, S.; Van Willigen, B.; Maintainer, R. Package 'nlme'. *Linear Nonlinear Mixed Eff. Model. Version* **2017**, *3*. `https://CRAN.R-project.org/package=nlme` (accessed on 12 June 2022).

[48] González-Recio, O.; Forni, S. Genome-wide prediction of discrete traits using Bayesian regressions and machine learning. *Genet. Sel. Evol.* **2011**, *43*, 1–12. https://doi.org/10.1186/1297-9686-43-7.

[49] Montesinos-López, O.A.; Montesinos-López, A.; Pérez-Rodríguez, P.; Barrón-López, J.A.; Martini, J.W.; Fajardo-Flores, S.B.; Gaytan-Lugo, L.S.; Santana-Mancilla, P.C.; Crossa, J. A review of deep learning applications for genomic selection. *BMC Genom.* **2021**, *22*, 1–23. https://doi.org/10.1186/s12864-020-07319-x.

[50] Belkin, M.; Hsu, D.; Ma, S.; Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 15849–15854. https://doi.org/10.1073/pnas.1903070116.

[51] Tong, H.; Nikoloski, Z. Machine learning approaches for crop improvement: Leveraging phenotypic and genotypic big data. *J. Plant Physiol.* **2021**, *257*, 153354. https://doi.org/10.1016/j.jplph.2020.153354.

[52] Mendes-Moreira, J.; Soares, C.; Jorge, A.M.; Sousa; Jorge F. Ensemble approaches for regression: A survey. *Acm computing surveys (csur).* **2012**, *45, 1–40.* https://doi.org/10.1145/2379776.2379786.

[53] Breiman, L. *Random forests*; Springer: Berlin/Heidelberg, Germany, 2001; Volume 45, pp. 5–32.

[54] Geurts, P.; Ernst, D.; Wehenkel, L. Extremely randomized trees. *Machine learning.* **2006**, *63, 3–42.* https://doi.org/10.1007/s10994-006-6226-1.

[55] Friedman, J.H. Greedy function approximation: a gradient boosting machine. *Annals of statistics.* **2001**, *29, 1189–1232.* https://doi.org/10.1214/aos/1013203451.

[56] Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of computer and system sciences* **1997**, *55*, 119–139. https://doi.org/10.1006/jcss.1997.1504.

[57] Zou, H.; Hastie, T. Regularization and variable selection via the elastic net. *J. of the royal statistical society: series B (statistical methodology)* **2005**, *67*, 301–320. https://doi.org/10.1111/j.1467-9868.2005.00503.x.

[58] Tibshirani, R. The lasso method for variable selection in the Cox model. *Statistics in medicine* **1997**, *4*, 385–395.

[59] Park, T.; Casella, G. The bayesian lasso. *J. of the American Statistical Assoc.* **2008**, *103*, 681–686.

[60] de Los Campos, G.; Naya, H.; Gianola, D.; Crossa, J.; Legarra, A.; Manfredi, E.; Weigel, K.; Cotes, J.M. Predicting quantitative traits with regression models for dense molecular markers and pedigree. *Genetics* **2009**, *182*, 375–385. https://doi.org/10.1534/genetics.109.101501.

[61] Yin, L.; Zhang, H.; Tang, Z.; Xu, J.; Yin, D.; Zhang, Z.; Yuan, X.; Zhu, M.; Zhao, S.; Li, X.; Xiaolei L. rMVP: a memory-efficient, visualization-enhanced, and parallel-accelerated tool for genome-wide association study. *Genomics, proteomics & bioinformatics* **2021**, *19*, 619–628. https://doi.org/10.1016/j.gpb.2020.10.007.