GP-ML-DC: An Ensemble Machine Learning-Based Genomic Prediction Approach with Automated Two-Phase Dimensionality Reduction via Divide-and-Conquer Techniques

Quanzhong Liu[1,4], Haofeng Ma[1], Zhuangbiao Zhang[2], Zhunhao Hu[1], Xihong Wang[2], Ran Li[2], Yudong Cai[2,*], Yu Jiang[2,3,*]

[1]College of Information Engineering, Northwest A&F University, Yangling, 712100, China;

[2]Key Laboratory of Animal Genetics, Breeding and Reproduction of Shaanxi Province, College of Animal Science and Technology, Northwest A&F University, Yangling 712100, China

[3]Key Laboratory of Livestock Biology, Northwest A&F University, Yangling, Shaanxi 712100, China.

[4]Shaanxi Engineering Research Center of Agricultural Information Intelligent Perception and Analysis, Northwest A&F University, Yangling 712100, China

[*]To whom correspondence should be addressed: (1) Yudong Cai and Yu Jiang, Key Laboratory of Animal Genetics, Breeding and Reproduction of Shaanxi Province, College of Animal Science and Technology, Northwest A&F University, Yangling 712100, China. Email: yu.jiang@nwafu.edu.cn, yudong.cai@nwafu.edu.cn.

Abstract

Traditional machine learning (ML) and deep learning (DL) methods for genome prediction often face challenges due to the imbalance between the limited number of samples (n) and the large number of single nucleotide polymorphisms (SNPs) (p), where n is much smaller than p. To address this, we propose GP-ML-DC, an innovative genome predictor that combines traditional ML and DL models with a unique two-phase, parameter-free dimensionality reduction technique. Initially, GP-ML-DC reduces feature dimensionality by characterizing genes as features. Building on big data methodologies, it employs a divide-and-conquer approach to segment gene regions into multiple haplotypes, further decreasing dimensionality. Each haplotype segment is processed by a sub-task based on traditional ML, followed by integration via a neural network that synthesizes the results of all sub-tasks. Our experiments, conducted on four cattle milk-related traits using ten-fold cross-validation and independent testing, show that GP-ML-DC significantly surpasses current state-of-the-art genome predictors in prediction performance.

Introduction

Predicting phenotypes directly from genotypes offers significant advantages in modern breeding [1]. To address the limitations of traditional marker-assisted selection (MAS) for predicting complex traits in crop and livestock breeding, genomic selection (GS) [2] has been introduced. GS utilizes dense molecular markers across the entire genome to evaluate the breeding values of individuals. The initial step in GS involves building a genomic prediction (GP) model using the training population, thereby establishing genotype-phenotype relationships. Once constructed, this model can predict phenotypes or breeding values for test populations based solely on their genotypic data. Commonly, GP models employ statistical methods, such as linear mixed models [2], including best linear unbiased prediction (BLUP), genomic BLUP (GBLUP), HIBLUP [3], and Bayesian linear regression models [4], which have been extensively applied in both animal and crop breeding [5, 6].

Linear mixed models have indeed propelled advancements in breeding by capturing linear relationships; however, they often fail to account for non-linear effects on complex traits [7]. Recently, machine learning (ML)-based GP methods have emerged, enhancing predictions of complex traits due to their ability to capture non-linear relationships. Performance comparisons among traditional linear models, conventional ML, and deep learning (DL)-based models indicate that no single approach consistently outperforms others universally [8]. Reviews [1, 5, 7, 9, 10] of diverse ML and DL predictors for GP similarly conclude that no single method excels across all contexts, underscoring the ongoing challenge of predicting complex phenotypes.

Despite advancements, ML and DL methods face challenges due to the imbalance where the number of individuals (n) is significantly less than the number of SNPs (p) [5, 7], hindering the performance of both statistical and ML models [11]. To address this, feature selection algorithms are often employed to reduce SNP dimensionality and improve GP performance [12]. Yet, standard feature selection algorithms such as filter, wrapper, and embedded methods [13] face difficulties in fully enhancing performance and often overlook non-linear SNP relationships or depend on a threshold of dimensionality reduction. While haplotype-based methods reduce dimensionality effectively [14, 15], they depend on hyperparameters for defining haplotype block boundaries such as window size or an LD threshold [16]. Using genes as feature descriptors can somewhat reduce

dimensionality [17, 18]; however, the dataset's dimensionality, indicated by the number of annotated genes (q), often remains high ($n \ll q$). Consequently, the issue of having more features than samples ($n \ll q$) persists. Addressing these high-dimensional features continues to pose a significant challenge for ML-based GP.

To resolve these issues, we propose a machine learning-based genomic predictor, GP-ML-DC, aimed at enhancing GS performance. GP-ML-DC introduces a parameter-free, gene-based feature selection algorithm, reducing dimensionality from SNPs to genes. Inspired by big data principles, it employs a divide-and-conquer strategy [19] to split gene regions into 16 core haplotypes, treating predictions from each as meta-features. This results in two phases of automatic dimension reduction: from p (the number of SNPs) to q (the number of genes), and from q to 16 meta-features, which are then inputted into a neural network for final phenotype predictions. GP-ML-DC is designed for end-to-end application. We conducted a five-time ten-fold cross validation on four milk-related traits (DMY, MFY, MPY, and SCS) using datasets from dairy farms in Hebei province, China, and compared GP-ML-DC with three state-of-the-art predictors including GBLUP (based on statistics models), LightGBM [20] (based on traditional ML), and DNNGP [21] (based on DL). Results demonstrate that GP-ML-DC improved predictions by 15.2% for DMY, 21.4% for MFY, 10.0% for MPY, and 19.5% for SCS compared to LightGBM (the second-best predictor). Independent test using datasets from dairy farms in Ningxia province, China indicate that GP-ML-DC improved predictions by 24.2% for DMY, 19.1% for MFY, 17.9% for MPY, and 8.4% for SCS compared to LightGBM (the second-best predictor).

Results

Design of genome prediction model

Figure 1 outlines the structure of the proposed GP-ML-DC model, which consists of two main modules: data mapping and ensemble ML-based prediction. The data mapping module involves two stages: feature engineering and data division. In the data mapping phase, we first perform feature engineering by extracting 16 features for each gene from variant information files. This yields a matrix M of size $q \times 16$ for each sample, where q denotes the number of genes. The raw data from n samples is thereby transformed into a three-dimensional data tensor $D(1..n, 1..q, 1..16)$, thereby achieving dimensionality reduction from p to q (called the first phase of dimensionality reduction).

We then perform the data division by extracting a matrix $D(:,:,f)$ for the fth feature from D, with $f \in [1,16]$.

In the ensemble ML-based prediction module, there are two stages: conquering and combining. During the conquering stage, a LASSO model is trained on the extracted $D(:,:,f)$ for each feature f to learn a meta-feature using a cross-validation strategy from stacking ensemble learning [22]. The 16 meta-features are then combined to construct a training set for the combining stage, further reducing feature dimensionality from q to 16 (called the second phase of dimensionality reduction). In the combining stage, a densely connected network is subsequently trained on this dataset to predict testing samples. Further details can be found in the **Methods** section.

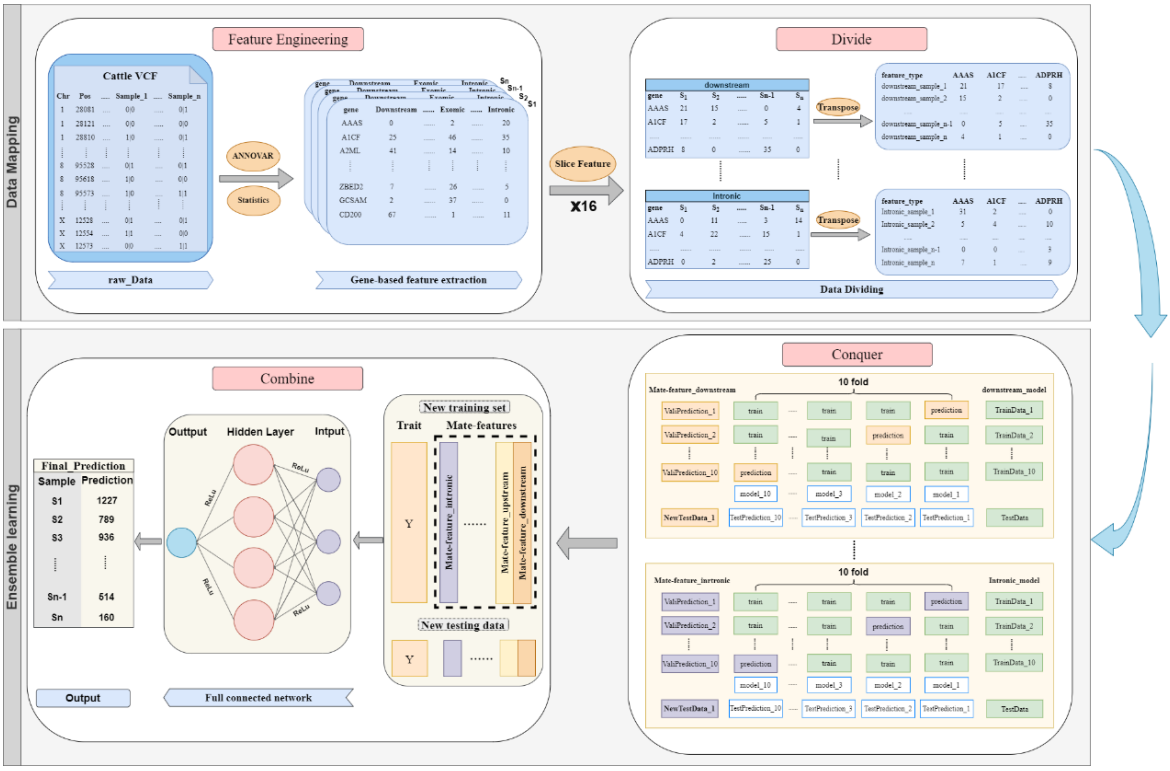

Figure 1. The workflow of GP-ML-DC

Performance comparison with other methods across ten-fold cross validation

We conducted five rounds of ten-fold cross-validation on four traits (DMY, MFY, MPY, and SCC) using cow datasets (refer to **Methods**) from Hebei province, China. The performance of GP-ML-DC was evaluated against three state-of-the-art methods: the GBLUP statistics model, LightGBM [20] based on traditional ML, and DNNGP [21] based on DL. As shown in **Figure 2A**, GP-ML-DC outperformed GBLUP (0.664 versus 0.4994 on DMY, 0.6114 versus 0.4192 on MFY, 0.6244 versus

0.4768 on MPY, 0.5102 versus 0.3418 on SCS), LightGBM (0.664 versus 0.5762 on DMY, 0.6114 versus 0.5452 on MFY, 0.6244 versus 0.5674 on MPY, 0.5102 versus 0.4268 on SCS), and DNNGP (0.664 versus 0.4396 on DMY, 0.6114 versus 0.4470 on MFY, 0.6244 versus 0.4378 on MPY, 0.5102 versus 0.3732 on SCS) in terms of PCC (Pearson's correlation coefficient). Additionally, as shown in **Figure 2B**, GP-ML-DC also surpassed LightGBM (the second-best performer) concerning R^2 values (0.445 versus 0.3318 on DMY, 0.3746 versus 0.2986 on MFY, 0.3918 versus 0.3230 on MPY, 0.2366 versus 0.1712 on SCS).

To further validate the model's robustness, we plotted 50 prediction results from five rounds of ten-fold cross-validation, as shown in **Figure 2C** and **Figure 2D**, for PCC and R^2 performance, respectively. As can be seen, compared with other three methods, GP-ML-DC demonstrated superior median PCC and R^2 values across 50 iterations for each trait. Additionally, we further conducted a Levene's test and Student's t-test to examine the statistical differences in performance between GP-ML-DC and other three predictors, with a significance cut-off of 0.05. Statistical results for PCC performance are provided in **Tables S1-S3**, and for R^2 performance are provided in **Table S4-S6**. In terms of PCC, variance differences across all four traits were not significant between GP-ML-DC and GBLUP (**Table S1**), as well as between GP-ML-DC and LightGBM (**Table S2**). However, mean differences were statistically significant (**Table S1-S2**). For the statistical results between GP-ML-DC and DNNGP (**Table S3**), variance differences were not significant for three of the four traits, except for the MPY trait; mean differences were statistically significant across all four traits. Similar R^2 results are observed in **Table S4-S6.** These results indicated significant differences in the means of PCC and R^2 across the four traits between GP-ML-DC and the other methods, underscoring the superior and more robust performance of GP-ML-DC.

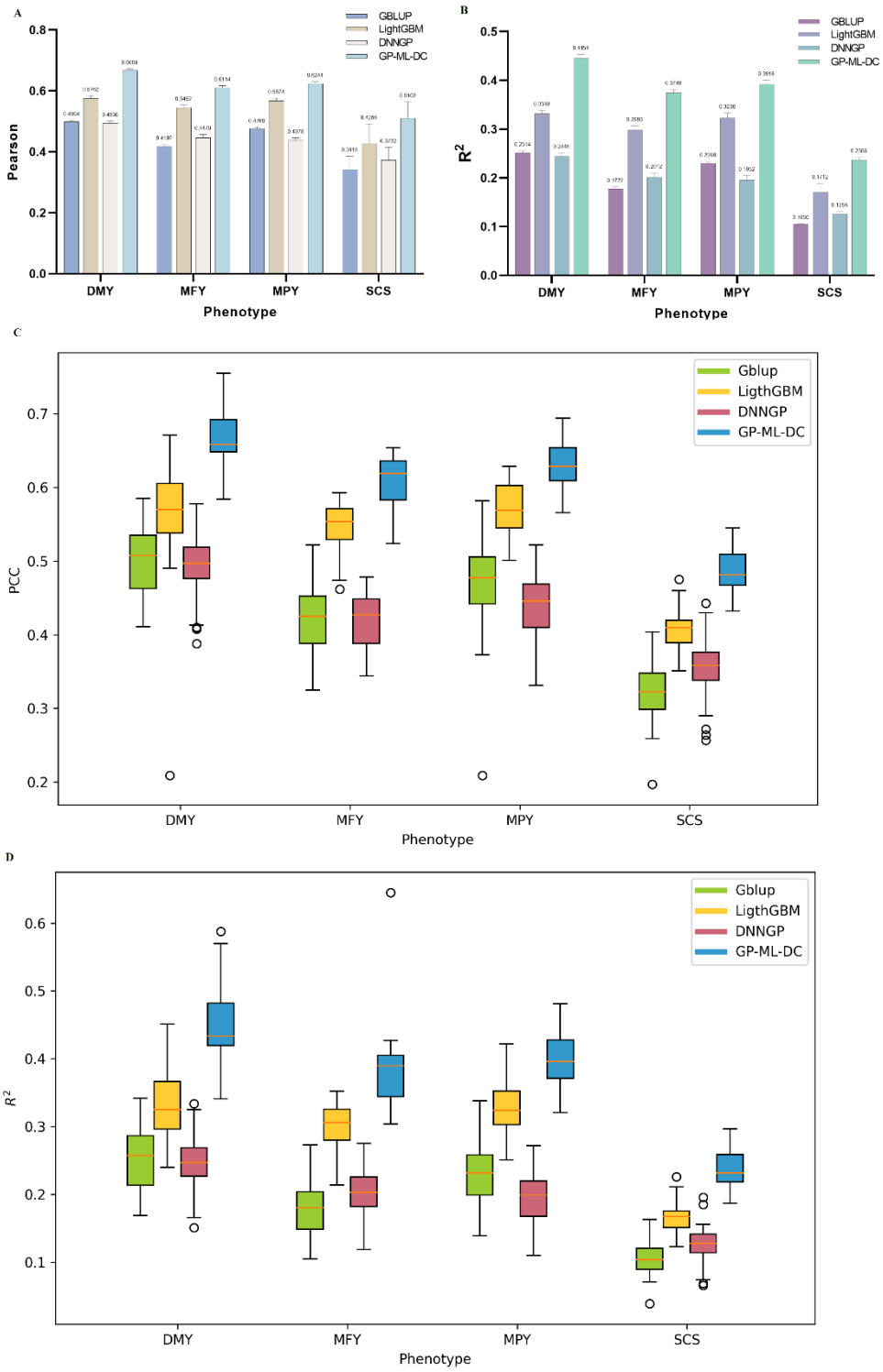

Figure 2. Performance comparison of four methods across four traits from the Hebei dataset using five-time ten-fold cross-validation. (A) Comparation results for the average PCC. (B) Comparation results for the average R^2. (C) Distribution of 50 PCC scores from five-time ten-fold cross-validation. (D) Distribution of 50 R^2 scores from five-time ten-fold cross-validation.

Independent testing and generalization performance comparison

Given the importance of generalization in ML models, we assessed how well GP-ML-DC transferred GP capabilities to different populations. Unlike traditional ML models, which rely on SNP-based matrices that may vary between groups, GP-ML-DC uses gene-based predictions that are applicable across diverse environments, such as populations from various farms. To assess the model's generalization capability, we utilized a model trained on the Hebei dataset to independently predict four milk-related traits in a Ningxia dataset (refer to the **Methods** section). We compared the performances of GP-ML-DC with GBLUP, LightGBM, and DNNGP based on PCC and R^2 values by comparing predicted and observed values. As illustrated in **Figure 3**, GP-ML-DC consistently outperformed other three methods in PCC and R^2 values for DMY, MFY, and MPY traits. Specifically, GP-ML-DC showed superior performance compared to LightGBM (the second-best performer), with PCC values of 0.442 versus 0.356 for DMY, 0.392 versus 0.329 for MFY, and 0.402 versus 0.199 for MPY, as well as R^2 values of 0.195 versus 0.127 for DMY, 0.154 versus 0.108 for MFY, and 0.162 versus 0.116 for MPY. Additionally, GP-ML-DC achieved the second-highest PCC of 0.298 and R^2 of 0.089 for the SCS trait. These findings indicate GP-ML-DC can effectively transfer GP across different populations on various farms.

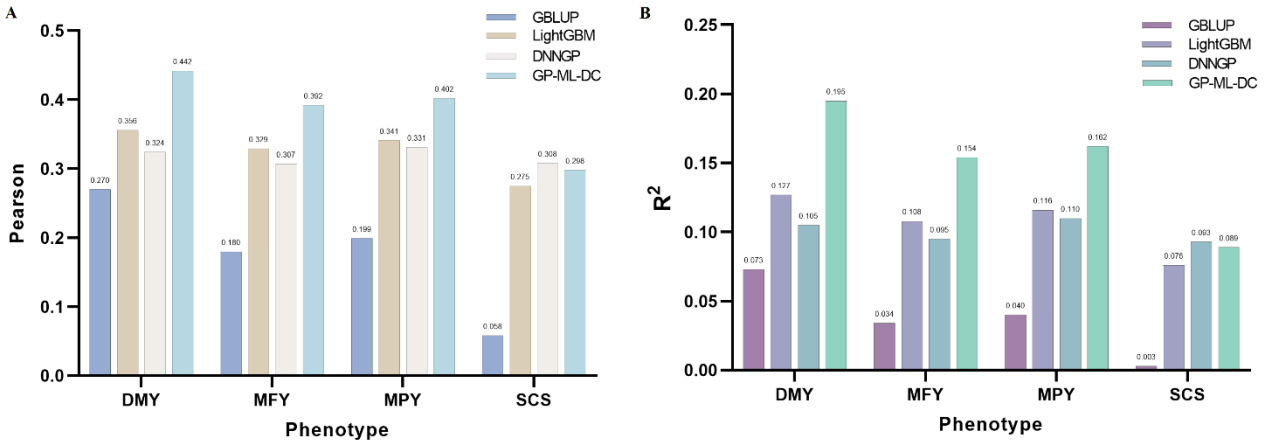

Figure 3. Performance comparison of four methods across four traits for the independent test on the Hebei dataset. (A) Comparation results for the PCC of four methods. (B) Comparation results for the R^2 of four methods.

Performance evaluation on the 50K SNP chip extracted by our model

We applied **Algorithm 4**, detailed in the **Methods** section, to the Hebei dataset to extract a 50K SNP chip, named SNP50N. To evaluate its effectiveness, we compared SNP50N with SNP50V3 [23], a widely recognized standard 50K SNP chip in the research community, using the same dataset. We

performed ten-fold cross-validation tests using GBLUP on both SNP50N and SNP50V3 of the Hebei dataset. Results of these comparisons are presented are illustrated in **Figure 4**. As depicted, the GBLUP performance for SNP50N showed marked improvements in both PCC and R^2 over SNP50V3 across all traits analyzed. For the DMY trait, enhancements were 10.3% and 21.4%, respectively (**Figure 4A-B**); for MFY, 11.8% and 24.3% (**Figure 4C-D**); for MPY, 10.6% and 22.2% (**Figure 4E-F**); and for SCS, 1.9% and 2.9% (**Figure 4G-H**).

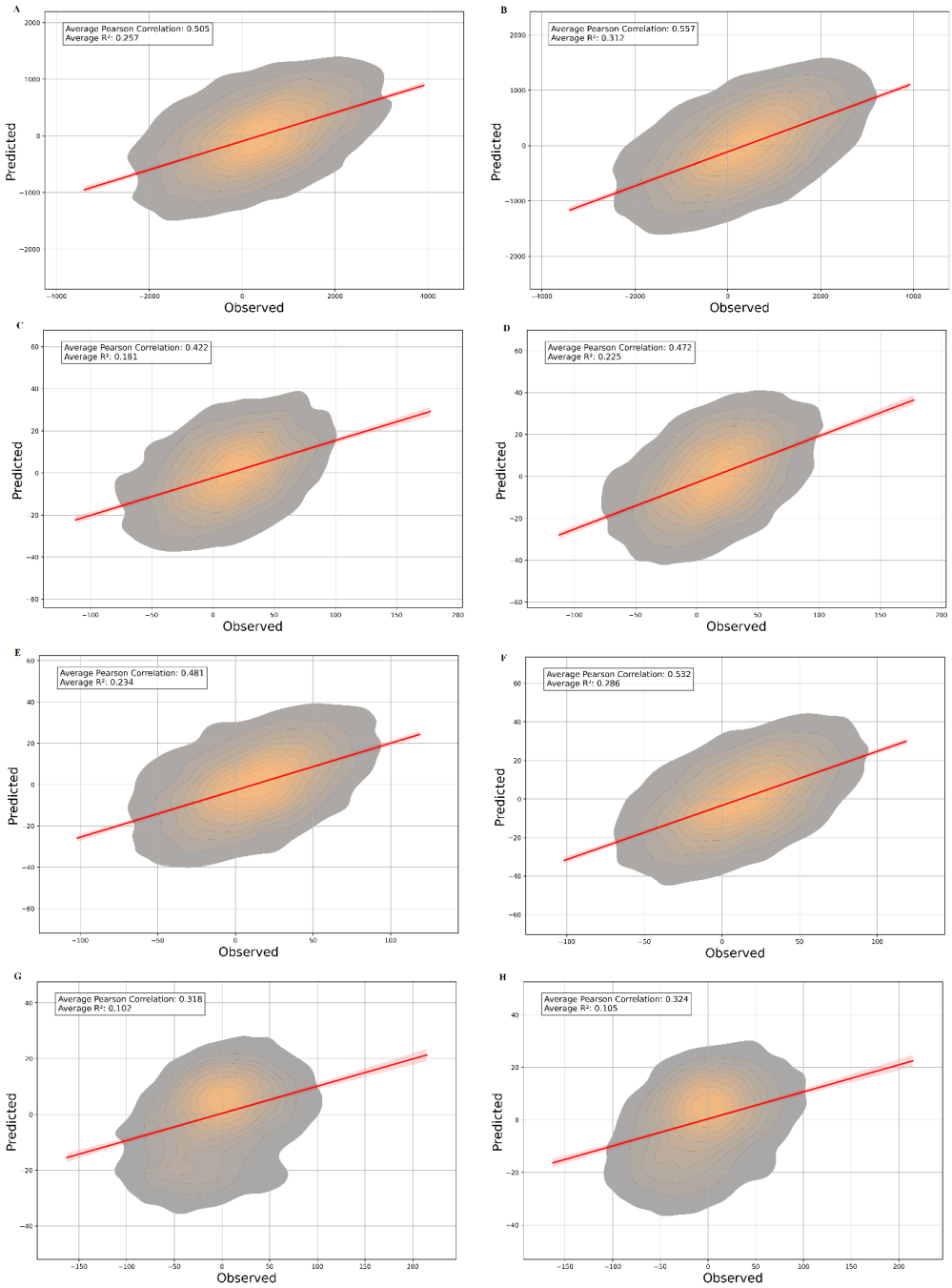

Figure 4. Performance comparisons of GBLUP between SNP50V3 and SNP50N extracted in this study. (A)

Performances on SNP50V3 for the DMY trait. (B) Performances on SNP50N for the DMY trait. (C) Performances on SNP50V3 for the MFY trait. (D) Performances on SNP50N for the MFY trait. (E) Performances on SNP50V3 for the MPY trait. (F) Performances on SNP50N for the MPY trait. (G) Performances on SNP50V3 for the CSC trait. (H) Performances on SNP50N for the CSC trait.

Discussion and conclusions

Genomic predictors often encounter challenges due to a limited number of samples compared to the vast number of genetic variants. Machine learning-based predictors also struggle with inconsistent SNP profiles across individuals or groups within the same species, limiting their broader applicability. Traditional feature selection methods attempt to address these issues by reducing the dimensionality of variables, but often at the expense of losing critical information necessary for effective genomic selection. In this study, we introduce GP-ML-DC, an innovative genome prediction model designed to address these challenges. This model embeds a novel two-phase, parameter-free dimensionality reduction process that utilizes genes to describe the genome, thereby enhancing its applicability across different individuals or groups within the species. Based on the concept of big data techniques, the approach goes beyond the traditional counting of SNPs in a gene region. It segments genes into haplotypes based on variant types using a divide-and-conquer strategy. Traditional machine learning models analyze each haplotype, while deep learning integrates these analyses to assess the collective contribution of all haplotypes to trait prediction. This ensemble strategy leverages the strengths of both traditional machine learning and deep learning in genomic selection.

Our experimental analysis, conducted through five rounds of ten-fold cross-validation, indicates that GP-ML-DC significantly outperforms three other representative predictors, including those based on statistical, machine learning, and deep learning methods. The 50K SNP chip produced by our method exhibits superior performance compared to the standard 50K SNP chip widely recognized in the research community. Additionally, the GP-ML-DC model can be seamlessly applied across diverse populations within the same species. Independent testing on milk-related trait prediction across various farms revealed that GP-ML-DC consistently achieves superior results compared to other predictors. To our knowledge, GP-ML-DC is the first genome prediction tool that combines the concept of big data with artificial intelligence techniques. We anticipate that GP-ML-DC will establish itself as a leading genome prediction tool, providing accurate phenotype predictions and enhancing the breeding process.

Methods

Datasets

In this study, we collected data through the Dairy Herd Improvement (DHI) program from over 55,871 Holstein cows across Hebei and Ningxia provinces in China. Specifically, 26068 cows were sourced from nine large-scale farms in Ningxia, while 29083 cows came from four large-scale farms in Hebei. Our genetic evaluation focused on key milk production traits, including daily milk yield (DMY), milk fat yield (MFY), milk protein yield (MPY), and somatic cell score (SCS), utilizing a random regression test-day model [24]. From the overall dataset, we identified a subset of 6,411 cows possessing comprehensive pedigree and phenotypic records. Among these, 6,010 cows had documented first-lactation data for DMY, MFY, MPY, and SCS. This subset included 2,725 were from Ningxia and the remainder from Hebei.

The blood samples were used from the cows for extractions of the DNA using the DNeasy Blood & Tissue kit (QiaGen). We used 0.5 μg genomic DNA to build a paired end library according to the standard library preparation protocols, and sequencing was performed in DNBseq platform. The low-quality reads in 6010 cows were first filtered using fastp [25] with default parameters. Then, clean reads were mapped to the bovine reference genome ARS‐UCD1.2 using BWA‐MEM [26] v0.7.17. We used SAMtools [27] to convert and sort bam files.

Data mapping

Feature engineering

In GP, the training population typically consists of several hundred individuals or samples, denoted as n, and millions of SNPs, denoted as p, with the condition $n \ll p$. This imbalance poses a significant challenge for the performance of statistical models and machine learning-based methods [11]. One effective strategy to address this issue is reducing feature dimensionality by using annotated genes to represent the genome, thereby shifting from p to q (the number of annotated genes) [17]. However, the issue ($n \ll q$) still persists. By considering each annotated gene as a haplotype, GP performance can be improved. Nevertheless, genes are typically longer than standard haplotypes, which might hinder performance. To mitigate this, gene regions can be divided into multiple haplotypes, allowing these haplotypes to represent the gene more effectively, thus enhancing GP

performance.

To implement this approach, we utilized the ANNOVAR software [28] to annotate each gene with 16 variant types, including: 'upstream', 'downstream', 'UTR', 'intronic', 'nonsynonymous', 'synonymous SNV', 'exonic', 'ncRNA exonic', 'ncRNA intronic', 'ncRNA splicing', 'stoploss', 'stopgain', 'splicing', 'gwas_postive', and 'gwas_negative'. A detailed description of these 16 variant types is provide in **Table S7**. The feature description for the *j*th gene in the *i*th sample ($1 \leq i \leq n$, $1 \leq j \leq q$) is denoted as g_{ij}, defined as:

$$g_{ij} = V\left(N_{ij}(SNP_{T_1}), N_{ij}(SNP_{T_2}), \dots, N_{ij}(SNP_{T_{15}}), N_{ij}(SNP_{T_{16}})\right) \quad (1)$$

$$S_i = V(g_{i1}, g_{i2}, \dots, g_{i(q-1)}, g_{iq}) \quad (2)$$

where SNP_{T_v} denotes the *v*th($v \in [1,16]$) type of variant, and $N_{ij}(SNP_{T_v})$ is the count of the *v*th ($v \in [1,16]$) type of variants in the *j*th gene for the *i*th sample. The vector function $V(.)$ constructs a vector of 16 values representing $N_{ij}(SNP_{T_v})$. Consequently, S_i becomes a vector consisting of q vectors, forming a $q \times 16$ matrix to describe the *i*th sample by genes instead of SNPs. This method achieves the first phase of dimensionality reduction from p to q in the dataset. The process is further detailed in **Algorithm 1**.

Algorithm 1. GENE-ENCODING(x)
Input: x: population VCF data;
1. $n \leftarrow$ #samples in x, $q \leftarrow$ #genes
2. **for** i = 1 to n **do**
2 **for** j = 1 to q **do**
3. compute $N_{ij}(SNP_{T_v})$ ($v \in [1,16]$) in x_i
4. generate g_{ij} using Formula 1
5. **end for**
6. generate S_i using Formula 2
7. **end for**
8. let $D = \{S_1, \dots, S_i, \dots, S_n\}$
9. **Output:** $\boldsymbol{D}$

Data division

Based on **Algorithm 1**, the raw data of *n* samples is transformed into a three-dimensional data tensor, $D(1..n, 1..q, 1..16)$, defined as:

$$D = V(S_1, S_2, \dots, S_i, S_n) \quad (3)$$

To effectively work with D, we developed a novel divide-and-conquer algorithm for GS, detailed in **Algorithm 2**. This algorithm consists of two phases: data division and data conquering. During the division phase (lines 4-5 in **Algorithm 2**), matrices $D_v(1..n, 1..q, v)$ are extracted for the vth feature (the vth type of variants) from D, where $v \in [1,16]$. Each $D_v(1..n, 1..q, v)$ denotes a $n \times q$ matrix, with each row represents a sample and each column corresponds to a gene, and the value in each cell indicating the occurrence count of the vth type of variants in the corresponding gene and sample. In this way, the three-dimensional data tensor D is divided into 16 two-dimensional matrices.

Ensemble machine learning-based model

Dataset conquering

The prediction model in the proposed GP-ML-DC framework consists of two stages: conquering and combining, as detailed in **Algorithm 2**. In the conquering stage, we launch a subtask $M_v(1 \leq v \leq 16)$ by training the LASSO model on the extracted matrix $D_v(1..n, 1..q, v)$ for the vth type of feature (variant). Each M_v learns the contribution of the vth haplotype (e.g., some variant region, the v type of variants) of each gene to the phenotypes. The prediction outcome of M_v provides a partial contribution to the overall genome prediction task. By drawing on the principles of stacking ensemble learning [29], we regarded the prediction value of M_v as a meta-feature of the final genome prediction task. To mitigate overfitting, a ten-fold cross-validation strategy is employed to learn meta-features, the process (lines 6-15 in **Algorithm 2**) is described as follows. The matrix $D_v(1..n, 1..q, v)$ is randomly divided into ten folds (line 6 in **Algorithm 2**). A LASSO model is then trained on nine folds while predicting the remaining fold (lines 8-9 in **Algorithm 2**), with the prediction results serving as meta-features for the samples in the validation fold (lines 11-14 in **Algorithm 2**). This process repeats for all ten folds, resulting in ten trained models stored in M_v (line 16 in **Algorithm 2**), and converting $D_v(1..n, 1..q, v)$ into a one-dimensional vector of length n, where each element represents the meta-feature of a sample. Upon completing all 16 subtasks($M_1, M_2, \ldots, M_{16}$), the three-dimensional dataset $D(1..n, 1..q, 1..16)$ is transformed into a new dataset where each sample is represented by a vector of 16 meta-feature values (line 13 in **Algorithm 2**). This process accomplishes the second phase of dimensionality reduction from q to 16 features.

Algorithm 2. ML-DIVIDE-CONQUER (x, y)
Input: x: training population VCF data;
y: traits

1. $D(1..n, 1..q, 1..16) \leftarrow$ GENE-ENCODING (x)
2. let D' be a $n \times 16$ matrix
3. let $M[0..16]$ be a new array
4. **for** v = 1 to 16 **do**
5. extract $D_v(1..n, 1..q, v)$ from D
6. randomly split D_v into 10 folds with equal-size: $D_v = \{D_{v1}, D_{v2} \ldots, D_{v10}\}$
7. $M_v = null$ //make M_v be an empty list
8. **for** k = 1 to 10 **do**
9. learn a model M_{vk} by training a LASSO on $D_v \setminus D_{vk}$
10. $M_v.append(M_{vk})$ # add M_{vk} into M_v
11. **for** $x_i \in D_{vk}$ **do**
12. $y_pre = M_{vk}.prediction(x_i)$
13. $D'[i][v] = y_pre$
14. **end for**
15. **end for**
16. $M[v] = M_v$
17. **end for**
18. $M[0]$ = den (den (D', y)) # trained NN model is stored in $M[0]$
19. **return** M

Result combining

To integrate the contributions of all haplotypes (e.g., various types of variant regions) within genes, we trained a neural network with two fully connected layers on $D'(1..n, 1..16)$. This network consolidates prediction results from each type of variant region (a haplotype) surrounding each gene for the final prediction. To enhance the representation capability of network. The first layer is followed by a ReLU activation function to enhance representational capacity. The output layer with a a single neuron generates the final prediction result. The process is described in line 17 of **Algorithm 2**.

Prediction

For phenotype prediction of a testing population, given their variant calling genotype data x_test, and corresponding traits y_train from a training population x_train (variant calling genotype data)

The phenotype prediction process for x_test is detailed in **Algorithm 3**. Initially, we apply the GENE-ENCODING (x_test) process on x_test as detailed in **Algorithm 1**, mapping x_test into D, which is characterized by genes. Following this, we invoke the ML-DIVIDE-CONQUER (x_train,y_train) method on x_train and y_train as outlined in **Algorithm 2**, producing a list of models, denoted as M. Here, $M[0]$ (line 18 in the **Algorithm 2**) denotes the final model for genome prediction, while $M[v](1 \leq v \leq 16)$ contains 10 models that are used to derive meta-features via ten-fold cross-validation for the vth type of variants (lines 7-10 in the **Algorithm 2**). Using $M[v]$, predictions are made on D_v (the matrix denoting the *v*th type of feature in D) to generate meta-features (lines 10-18 in the **Algorithm 3**). The average of 10 prediction values provides the meta-feature's value for each type of variants in testing dataset (line 19 in the **Algorithm 3**). Finally, $M[0]$ is launched to predict the phenotype using the datasets consisting of meta-features as input (line 21 in the **Algorithm 3**).

Extraction of variants from genes

Unlike the traditional GP methods that rely on SNP genotypes, the GP-ML-DC approach represents genetic information using genes as descriptors. However, SNPs embedded within these genes can still be extracted to form a subset. To evaluate the effectiveness of these extracted SNPs, we employ **Algorithm 4** to create a 50K SNP chip, allowing for comparison with the standard 50K chip typically used for analyzing milk traits in cows within the research community. In our analysis, a training population dataset $D(1..n, 1..q, 1..16)$, as defined in **Formula 3**, is organized using genes and can be derived using **Algorithm 1**. According to lines 3-8 in **Algorithm 4**, for the vth feature (the vth type of variants), where $v \in [1,16]$, a matrix $D_v(1..n, 1..q, v)$ is extracted from D. A LASSO model is then trained on D_v to determine gene contributions to traits, resulting in a ranked list l_v. Subsequently, in lines 10-17 in **Algorithm 4**, SNP variants from the top k genes in each l_v($v \in [1,16]$) are added to the $SNP50$ set, where k is less than the size of l_v. SNPs in the $SNP50$ set are governed by a linkage disequilibrium (LD) threshold of 0.7, until the set $SNP50$ reaches or exceeds 50K.

Algorithm 3. GENOME-PREDICTION (x_train, y_train, x_test)

Input: x_train: training population VCF data;
y_train: traits in samples in training population;
x_test: testing population VCF data

```
D(1..n, 1..q, 1..16) ← GENE-ENCODING (x_test)
M ← ML-DIVIDE-CONQUER (x_train, y_train)
let D' be a n × 16 matrix
for i = 1 to n do
    for j = 1 to 16 do
        D'[i][j] = 0
    end for
end for
for v = 1 to 16 do
    D_v = D(1..n, 1..q, v)
    M_v = M[v]
    for k = 1 to 10 do
        M_vk = M_v[k]
        for each x_i ∈ D_v do
            p_x_i = M_vk.prediction(x_i)
            D'[i][v] = D'[i][v] + p_x_i
        end for
    end for
    D'[i][v] = D'[i][v]/10
end for
y_pre = M[0].preiction(D'[i][v])
return y_pre
```

Algorithm 4. SNP50_Extraciton (x)
Input: x: population VCF data;
y: traits
$D(1..n, 1..q, 1..16) \leftarrow$ GENE-ENCODING (x)
$l = null$
for v = 1 to 16 **do**
extract $D_v(1..n, 1..q, v)$ from D
training a LASSO on D_v
$l_v \leftarrow$ sort genes according to their contributions to y
$l.append(l_v)$
end for
$SNP50 = null$
for $i = 0$ to $sizeof(l[1])$ **do**
for v = 1 to 16 **do**
$SNP50.append(l[v][i])$
end for
filter SNPs in $SNP50$ using an LD threshold of 0.7.
If $sizeof(SNP50) \geq 50000$ **then**
break
end for
Output: $SNP50$

Evaluation metrics and strategy

In the GS community, predictors typically fall into three categories: statistical models, ML-based models, and DL-based models. To evaluate the proposed model, we selected a state-of-the-art method from each category: the GBLUP statistics model, LightGBM [20] based on traditional ML, and DNNGP [21] based on DL. We employed two commonly used evaluation metrics in GS: Pearson's correlation coefficient (PCC) and R-squared (R^2), calculating these metrics to compare predicted and observed values in assessing model performance.

k-fold cross validation is a widely used strategy for evaluating GP models in the GS the research community. In this study, we conducted ten-fold cross validation on the Hebei dataset to benchmark the performance of GP-ML-DC against the three selected models. Each method underwent identical ten-fold partitioning. To mitigate potential bias from a single ten-fold cross validation, we performed five rounds of ten-fold cross-validation across four traits. The average results from these rounds were taken as the final performance measure.

It is essential to evaluate the generalization capability of ML models. While traditional ML and DL methods generally utilize SNPs-based genotype matrix for GP, inconsistent SNP spaces across individuals or groups can hinder their application in GP [17]. To further evaluate the effective of GP-ML-DC in predicting various individuals or populations, we conducted an independent test: predicting four traits in the Ningxia dataset using a model trained on the Hebei dataset.

Acknowledgments

We thank the High-Performance Computing (HPC) of Northwest A&F University (NWAFU) for providing computing resource, as well as artificial Intelligence platform of the College of Information Engineering, Northwest A&F Universit y.

Funding

This work has been supported by the National Key R&D Program of China (2023YFD1300400, 2022YFF1000100), and Shaanxi Livestock and Poultry Breeding Double-chain Fusion Key Project (2022GD-TSLD-46-0401).

Competing interests

The authors declare no competing interests.

References

1. John M, Haselbeck F, Dass R, Malisi C, Ricca P, Dreischer C, Schultheiss SJ, Grimm DG: **A comparison of classical and machine learning-based phenotype prediction methods on simulated data and three plant species.** *Front Plant Sci* 2022, **13:**932512.
2. Meuwissen THE, Hayes BJ, Goddard ME: **Prediction of total genetic value using genome-wide dense marker maps.** *GENETICS* 2001, **157:**1819-1829.
3. Yin L, Zhang H, Tang Z, Yin D, Fu Y, Yuan X, Li X, Liu X, Zhao S: **HIBLUP: an integration of statistical models on the BLUP framework for efficient genetic evaluation using big genomic data.** *Nucleic Acids Res* 2023, **51:**3501-3512.
4. Gianola D, de los Campos G, Hill WG, Manfredi E, Fernando R: **Additive genetic variability and the Bayesian alphabet.** *Genetics* 2009, **183:**347-363.
5. Nayeri S, Sargolzaei M, Tulpan D: **A review of traditional and machine learning methods applied to animal breeding.** *Anim Health Res Rev* 2019, **20:**31-46.
6. Chen J, Tan C, Zhu M, Zhang C, Wang Z, Ni X, Liu Y, Wei T, Wei X, Fang X, et al: **CropGS-Hub: a comprehensive database of genotype and phenotype resources for genomic prediction in major crops.** *Nucleic Acids Res* 2024, **52:**D1519-D1529.
7. Azodi CB, Bolger E, McCarren A, Roantree M, de Los Campos G, Shiu SH: **Benchmarking Parametric and Machine Learning Models for Genomic Prediction of Complex Traits.** *G3 (Bethesda)* 2019, **9:**3691-3702.
8. Abdollahi-Arpanahi R, Gianola D, Penagaricano F: **Deep learning versus parametric and ensemble methods for genomic prediction of complex phenotypes.** *Genet Sel Evol* 2020, **52:**12.
9. Chafai N, Hayah I, Houaga I, Badaoui B: **A review of machine learning models applied to genomic prediction in animal breeding.** *Front Genet* 2023, **14:**1150596.
10. Montesinos-Lopez OA, Chavira-Flores M, Kiasmiantini, Crespo-Herrera L, Saint Piere C, Li H, Fritsche-Neto R, Al-Nowibet K, Montesinos-Lopez A, Crossa J: **A review of multimodal deep learning methods for genomic-enabled prediction in plant breeding.** *Genetics* 2024.
11. Lakiotaki K, Papadovasilakis Z, Lagani V, Fafalios S, Charonyktakis P, Tsagris M, Tsamardinos I: **Automated machine learning for genome wide association studies.** *Bioinformatics* 2023, **39**.
12. Heinrich F, Lange TM, Kircher M, Ramzan F, Schmitt AO, Gultas M: **Exploring the potential of incremental feature selection to improve genomic prediction accuracy.** *Genet Sel Evol* 2023, **55:**78.
13. Piles M, Bergsma R, Gianola D, Gilbert H, Tusell L: **Feature Selection Stability and Accuracy of Prediction Models for Genomic Prediction of Residual Feed Intake in Pigs Using Machine Learning.** *Front Genet* 2021, **12:**611506.
14. Li H, Wang Z, Xu L, Li Q, Gao H, Ma H, Cai W, Chen Y, Gao X, Zhang L, et al: **Genomic prediction of carcass traits using different haplotype block partitioning methods in beef cattle.** *Evol Appl* 2022, **15:**2028-2042.
15. Pook T, Schlather M, de Los Campos G, Mayer M, Schoen CC, Simianer H: **HaploBlocker: Creation of Subgroup-Specific Haplotype Blocks and Libraries.** *Genetics* 2019, **212:**1045-1061.
16. Weber SE, Frisch M, Snowdon RJ, Voss-Fels KP: **Haplotype blocks for genomic prediction: a comparative evaluation in multiple crop datasets.** *Front Plant Sci* 2023, **14:**1217589.
17. Shen Z, Shen E, Zhu Q-H, Fan L, Zou Q, Ye C-Y: **GSCtool: A Novel Descriptor that Characterizes the Genome for Applying Machine Learning in Genomics.** *Advanced Intelligent Systems* 2023, **5**.

18. Shen Z, Shen E, Yang K, Fan Z, Zhu QH, Fan L, Ye CY: **BreedingAIDB: A database integrating crop genome-to-phenotype paired data with machine learning tools applicable to breeding.** *Plant Commun* 2024, **5:**100894.
19. Xu C, Zhang YQ, Li RZ, Wu XD: **On the Feasibility of Distributed Kernel Regression for Big Data.** *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING* 2016, **28:**3041-3052.
20. Yan J, Xu Y, Cheng Q, Jiang S, Wang Q, Xiao Y, Ma C, Yan J, Wang X: **LightGBM: accelerated genomically designed crop breeding through ensemble learning.** *Genome Biol* 2021, **22:**271.
21. Wang K, Abid MA, Rasheed A, Crossa J, Hearne S, Li H: **DNNGP, a deep neural network-based method for genomic prediction using multi-omics data in plants.** *Mol Plant* 2023, **16:**279-293.
22. Raschka S: **MLxtend: Providing machine learning and data science utilities and extensions to Python' s scientific computing stack.** *Journal of Open Source Software* 2018, **3(24):**638.
23. de Oliveira HR, Silva FF, Brito LF, Guarini AR, Jamrozik J, Schenkel FS: **Comparing deregression methods for genomic prediction of test-day traits in dairy cattle.** *J Anim Breed Genet* 2018, **135:**97-106.
24. Pereira RJ, Bignardi AB, El Faro L, Verneque RS, Vercesi AE, Albuquerque LG: **Random regression models using Legendre polynomials or linear splines for test-day milk yield of dairy Gyr (Bos indicus) cattle.** *JOURNAL OF DAIRY SCIENCE* 2013, **96:**565-574.
25. Chen S, Zhou Y, Chen Y, Gu J: **fastp: an ultra-fast all-in-one FASTQ preprocessor.** *Bioinformatics* 2018, **34:**i884-i890.
26. Li H, Durbin R: **Fast and accurate long-read alignment with Burrows-Wheeler transform.** *Bioinformatics* 2010, **26:**589-595.
27. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R, Genome Project Data Processing S: **The Sequence Alignment/Map format and SAMtools.** *Bioinformatics* 2009, **25:**2078-2079.
28. Wang K, Li M, Hakonarson H: **ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data.** *Nucleic Acids Res* 2010, **38:**e164.
29. Zhou z: **Ensemble Methods Foundations and Algorithms.** *CRC Press* 2012.