Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Gard Westrum Gravdal

Machine learning approaches to genomic prediction and marker association

Master's thesis in Applied Physics and Mathematics (MTFYMA)
Supervisor: Stefanie Muff
June 2025

NTNU
Norwegian University of
Science and Technology

Gard Westrum Gravdal

Machine learning approaches to genomic prediction and marker association

Master's thesis in Applied Physics and Mathematics (MTFYMA)
Supervisor: Stefanie Muff
June 2025

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Kunnskap for en bedre verden

Department of Mathematical Sciences

TMA4900 - Master Thesis

Machine learning approaches to genomic prediction and marker association

Gard Westrum Gravdal

Supervised by Stefanie Muff

Date: 18/06/2025

Abstract

Advances in genomic sequencing have increased the size and dimensionality of datasets from wild populations, while developments in machine learning offer new opportunities for phenotype prediction and identification of trait-associated genetic markers. In quantitative genetics, the genomic animal model remains a benchmark for genomic prediction, particularly for capturing additive genetic effects. However, flexible machine learning models such as XGBoost, which can account for both additive and non-linear interaction effects, present a promising alternative.

In this thesis, we compare the accuracy of XGBoost with a state-of-the-art genomic animal model implemented in a Bayesian framework, as well as with computationally efficient ridge regression approaches. We also explore dimensionality reduction through principal component analysis (PCA) and examine subset selection strategies to improve prediction accuracy in high-dimensional settings. Beyond prediction, we investigate methods for identifying associations between single nucleotide polymorphisms (SNPs) and phenotypes. Traditional genome-wide association studies (GWAS) rely on statistical significance thresholds and exhibit limitations highlighted by the missing heritability dilemma. To address these limitations, we evaluate alternatives using Shapley additive explanation (SHAP) values, which provide feature importance from XGBoost, and coefficient-based importance measures from ridge and lasso regression. These methods were applied to a genomic dataset from a wild meta-population of house sparrows (*Passer domesticus*) in Northern Norway, focusing on the traits body mass, wing length, and tarsus length. Additionally, methods were applied to phenotypes simulated under controlled settings to assess them across a range of genetic architectures.

Our findings reaffirm the Bayesian animal model as the most accurate genomic prediction model and ridge regression as a fast and reliable baseline. While XGBoost did not outperform the animal model, subset selection strategies suggest potential for further refinement. Our findings highlight the limitations of GWAS and suggest that SHAP-based approaches can provide a valuable alternative for ranking SNP importance. In contrast, coefficient-based methods from ridge and lasso regression exhibit limited utility for inference. Our findings suggest that flexible machine learning models like XGBoost and interpretable approaches such as SHAP may refine or even surpass traditional methods for genomic prediction and inferring trait-marker associations in complex traits.

Sammendrag

Fremskritt innen genomsekvensering har økt både størrelsen og dimensjonaliteten på datasett fra ville populasjoner, samtidig som utviklingen innen maskinlæring har åpnet for nye muligheter for prediksjon av fenotyper og identifikasjon av genetiske markører assosiert med ulike egenskaper. Innen kvantitativ genetikk er den genomiske dyremodellen fortsatt en referansemetode for genomisk prediksjon, spesielt når det gjelder å fange opp additive genetiske effekter. Samtidig representerer fleksible maskinlæringsmodeller som XGBoost, som kan modellere både additive og ikke-lineære interaksjonseffekter, et lovende alternativ.

I denne oppgaven sammenlignes prediksjonsnøyaktigheten til XGBoost med en avansert genomisk dyremodell implementert i et Bayesiansk rammeverk, samt med beregningseffektive varianter av ridge-regresjon. Vi undersøker også metoder for dimensjonsreduksjon gjennom prinsipal komponent analyse (PCA) og vurderer strategier for utvalg av SNP-delmengder for å forbedre prediksjon i høy-dimensjonale settinger. Utover prediksjon ser vi på metoder for å identifisere sammenhenger mellom enkeltnukleotidpolymorfismer (SNP-er) og fenotyper. Tradisjonelle genomomfattende assosiasjonsstudier (GWAS) er basert på signifikansgrenser og har kjente begrensninger, blant annet knyttet til det såkalte "manglende arvelighet" (missing heritability) dilemmaet. For å håndtere disse begrensningene vurderer vi alternativer ved å bruke Shapley additive forklaringer (SHAP) verdier, som gir mål på variabelbetydning fra XGBoost, samt koeffisientbaserte mål for viktighet fra ridge- og lasso-regresjon. Metodene anvendes på et genomisk datasett fra en vill metapopulasjon av gråspurv (*Passer domesticus*) i Nord-Norge, med fokus på egenskapene kroppsmasse, vingelengde og tarsuslengde. I tillegg anvendes metodene på simulerte fenotyper under kontrollerte forhold for å vurdere ytelsen på tvers av ulike genetiske arkitekturer.

Våre resultater bekrefter at den Bayesiske dyremodellen er den mest presise for genomisk prediksjon, mens ridge-regresjon fremstår som et raskt og pålitelig alternativ. Selv om XGBoost ikke overgikk dyremodellen, indikerer strategier for delmengdeutvalg at det finnes potensial for videre forbedringer. Resultatene våre fremhever også begrensningene ved GWAS, og antyder at SHAP-baserte metoder kan være et verdifullt alternativ for rangering av SNP-betydning. Til sammenligning viser koeffisientbaserte metoder fra ridge- og lasso-regresjon begrenset nytteverdi for inferens. Våre funn antyder at fleksible maskinlæringsmodeller som XGBoost og tolkbare metoder som SHAP kan videreutvikle eller til og med overgå tradisjonelle metoder for genomisk prediksjon og for å avdekke sammenhenger mellom genetiske markører og komplekse egenskaper.

Table of contents

List of Figures

1 Introduction

Quantitative traits are complex, continuous traits, such as body mass or wing length, that typically exhibit considerable variation within and among populations. A central question in evolutionary biology is the nature of the underlying forces that generate this variation (Walsh and Lynch, 2018). Increasing our understanding of quantitative traits can drive progress in several scientific fields. Advances in the mapping of quantitative traits have already contributed to a deep understanding of evolution, human health and diseases (Huang, 2015; Marioni et al., 2018), conservation biology (Arenas et al., 2021), and selective breeding in animals and plants (Hickey et al., 2017). The variation in quantitative traits is complex because these traits are influenced by both genetic effects, arising from the combined influence of many loci, each with a small effect, and non-genetic factors, such as environmental influences (Wood et al., 2014; McGaugh et al., 2021). A main objective in quantitative genetics is to partition the variation in the expression of a trait, otherwise known as the phenotypic variation, into its various components. Of particular interest is the additive genetic variance (V_A), which reflects the portion of phenotypic variance attributable to the linear additive effects of alleles. Determining the components of phenotypic variation is further complicated by the fact that the genetic component arises from multiple factors, including pleiotropy and gene-gene interaction effects such as epistasis and dominance effects.

A traditional method for estimating the genetic and environmental components of phenotypic variance is the *animal model*, a linear mixed model (LMM) that accounts for relatedness among individuals along with other hierarchical structures in the population (Henderson, 1988; Kruuk, 2004; Wilson et al., 2010; Bérénos et al., 2014). Many of the classical linear models, such as the linear regression model, assume independence between observations. However, the assumption of independence is violated in animal populations because genetic relatedness introduces correlation between individuals. Originally, relatedness was inferred from pedigrees, yielding the pedigree-based animal model (Henderson, 1988). Recent times have seen a decline in the cost of sequencing genetic markers, in particular the single nucleotide polymorphism (SNP), with growing availability of genetic material as a result (Misztal et al., 2020). Due to their abundance in the genome, SNPs serve as effective genetic markers, making the SNPs a useful alternative for accounting for relatedness between individuals (Bush and Moore, 2012). Modeling relatedness from SNP data leads to the genomic animal model, which has become the preferred approach in modern applications (VanRaden, 2008; Ashraf et al., 2021).

Prediction of the genetic component of the phenotype, known as *genomic prediction*, is another major goal in quantitative genetics (Meuwissen et al., 2001). The field of genomic prediction has seen a similar evolution from the use of traditional pedigree-based animal models, which led to the *best linear unbiased prediction* (BLUP, Henderson, 1975), to the genomic animal model, which introduced the *genomic best linear unbiased prediction* (GBLUP, VanRaden, 2008), and even to methods that directly use genetic markers in marker-based regression (Meuwissen et al., 2001). In terms of computational complexity, marker-based regression increases linearly in computation time with the number of individuals, whereas the genomic relatedness matrix used in the genomic animal model grows quadratically with the number of individuals and linearly with the number of markers (Aspheim et al., 2024).

Using SNPs directly in marker-based regression leads to statistical problems related to the dimensionality of the genomic dataset, because the number of SNP-covariates m typically far exceed the number of individuals n, known as the $m \gg n$ problem (Van De Geer and Van Houwelingen, 2004). In addition to rendering simple linear regression with ordinary least squares (OLS) unsuitable, the high-dimensionality in genomic datasets introduced a concern regarding overfitting in genomic prediction models. To address the high dimensionality, Meuwissen et al. (2001) first proposed Bayesian methods such as BayesA and BayesB, which shrink or select marker effects to improve predictive accuracy and limit overfitting. Another regularization approach is *ridge regression* (Whittaker et al., 2000), proposed as a robust alternative well-suited to scenarios where many predictors have small additive effects (Tibshirani, 1996), which typically aligns with the architecture of complex traits (Wood et al., 2014). Dimensionality reduction with *principal component analysis* (PCA, Abdi and Williams, 2010) represents an alternative strategy to address the $m \gg n$ challenge. PCA offers an efficient approach by transforming the high-dimensional SNP-matrix in-

tro a smaller set of orthogonal components, the principal components (PCs), that capture the main structure of genetic variation. The PCs can thus replace the original SNP markers in regression, reducing the model complexity while preserving informative signal. *Principal component ridge regression* (PCRR) is a model that has shown particular promise in several genomic prediction studies (Ødegård et al., 2018; Zelioli, 2023; Aspheim et al., 2024).

While genomic prediction is a widely used tool in animal and plant breeding (Meuwissen et al., 2001; Boubacar et al., 2013), it is not in widespread use for quantitative genetic studies of wild populations, though some studies have been made (Bérénos et al., 2014; Ashraf et al., 2021; Zelioli, 2023; Aspheim et al., 2024). The growing availability of genomic data also for wild populations, along with the development of flexible modern predictions models from machine learning and neural networks (Lourenço et al., 2024), motivate the search for alternative approaches that could potentially challenge the traditional methods in genomic prediction. *XGBoost* has emerged as a popular model in prediction tasks because of its flexibility and accuracy (Chen and Guestrin, 2016). Using regression trees as weak learners, XGBoost can model complex interactions among features (Molnar, 2022), which from a genomic perspective, makes XGBoost ideal at capturing interactions between SNPs, such as epistatic effects and dominance effects (Johnsen et al., 2021). The ability to capture interaction effects in addition to the additive genetic effect gives the XGBoost potential to challenge traditional methods in genomic prediction, such as the genomic animal model or ridge regression.

Although genomic prediction can be useful in many applications, it does not explicitly infer which parts of the genome are important to the expression of a trait. Predictions from machine learning models such as XGBoost are famously hard to interpret, meaning it is difficult to understand how the model arrived at its prediction (Boehmke and Greenwell, 2019; Molnar, 2022). Assessing which parts of the genome are important for the expression of a trait is also of interest in many applications, and another main task in the studies of quantitative genetics is to assess which SNPs are associated with the phenotype. The conventional approach of assessing SNP-trait associations is through methods in *genome-wide association studies* (GWAS), widely employed in previous research (*e.g.*, Murcray et al., 2009; Bush and Moore, 2012; Visscher et al., 2012; McKinney and Pajewski, 2012; Brodie et al., 2016; Uffelmann et al., 2021; Yengo et al., 2022). However, GWAS has known limitations. Its reliance on p-values poses challenges, and much more of the genome than previously thought is required to be able to explain the heritability of complex polygenic traits (McKinney and Pajewski, 2012). Complex polygenic traits are influenced by a large number of alleles, most of which have a very small effect size, making it hard for the SNPs to pass the strict significance threshold for the p-values, contributing to the "missing heritability" dilemma (Yang et al., 2010). Several studies have shown that huge sample sizes are needed to overcome the missing heritability issue (Eichler et al., 2010; Yang et al., 2010; Yengo et al., 2022). In GWAS, SNP-trait associations are typically evaluated using null-hypothesis significance testing and a stringent genome-wide threshold to account for multiple testing. While this convention helps control false positives, the broader reliance on significance thresholds has been widely debated and criticized within many scientific fields (Goodman, 2008; Head et al., 2015; Ioannidis, 2018; Muff, Nilsen et al., 2022). Issues with GWAS motivate the search for new alternative methods of assessing which SNPs are associated with the phenotype of interest.

In *interpretable machine learning* (IML), methods for explaining the predictions made by complex machine learning models are being developed (Boehmke and Greenwell, 2019; Molnar, 2022). In the context of genomic prediction, IML methods offer a promising approach for exploring how individual genetic markers contribute to traits, and may therefore serve as an alternative way to assess SNP-trait associations beyond conventional GWAS. One such method is the *Shapley additive explanation* (SHAP, *e.g.,* Molnar, 2022) framework, which extends the concept of Shapley values from cooperative game theory (Shapley, 1952), to machine learning models. In the SHAP approach, an importance metric is computed for each covariate of a machine learning prediction, quantifying the covariate's contribution to the outcome. For linear models, the SHAP method is analogous to the Lindeman, Merenda and Gold (LMG, Lindeman et al., 1980) decomposition of explained variance. In the context of genomic studies, aggregating SHAP values across individuals will yield an interpretable measure of average SNP importance. The SHAP approach offers an appealing alternative to p-value based GWAS by directly reflecting estimated marker effects and their influence on trait variation.

Regression methods that yield effect size estimates in the form of model coefficients may also provide an interpretable basis for identifying SNP-trait associations. The *least absolute shrinkage and selection operator* (lasso, Tibshirani, 1996) and ridge regression are two widely used linear regression methods that differ in how they apply regularization. The differing regularization strategies lead to different modeling behavior, where ridge regression tends to distribute effect sizes more evenly across correlated variables and lasso instead selects a few among them. While the shrinkage introduced by both ridge and lasso compromises the straightforward interpretation of coefficients in OLS, the estimated coefficients can still be informative about the effects of predictors on the response. In the context of marker-based regression, lasso and ridge yield coefficient estimates that reflect the contribution of individual SNPs to phenotypic variation, and may thus serve as measures of SNP-trait association. Compared to p-values from GWAS, which reflect a combination of effect size and statistical uncertainty, the coefficient estimates offer a more direct indication of SNP effect sizes. As such, they may provide an alternative approach to SNP-level inference and offer insights into the genetic architecture of complex traits, potentially addressing aspects of the missing heritability problem.

A central challenge in evaluating genomic prediction and SNP-trait association methods lies in the limited control over the underlying genetic architecture and environmental factors influencing real-world traits. The lack of control makes it difficult to assess the accuracy and robustness of different methods or to understand why one method may outperform another in a given study. As a consequence, it often remains unclear why a prediction model is accurately capturing the genetic signal or if an association method is truly identifying the SNPs that contribute to phenotypic variation. To overcome these limitations, studies in quantitative genetics often turn to simulation-based approaches, where phenotypes are generated under controlled and fully known conditions (Meuwissen et al., 2001; Zelioli, 2023; Aspheim et al., 2024). Simulations of phenotypes offer a powerful framework for systematically testing assumptions and evaluating model performance across a wide range of genetic architectures. By generating fictive traits with known properties, such as the number and distribution of causal variants, effect sizes, and trait heritability, the ability of inference methods to recover the true genetic signal can be assessed.

In this thesis, we investigated genomic prediction and SNP-trait association in a wild insular meta-population of house sparrows (*Passer domesticus*) in Northern Norway. We focused on the three phenotypic traits tarsus length, wing length, and body mass, and explored a range of methodological approaches using SNP data from two genomic datasets. One dataset offered high genomic resolution with a high number of SNPs, while the second dataset provided greater sample size at the cost of fewer SNPs. Genomic prediction was carried out using six models. These included three XGBoost models that differed in their SNP input, comparing the use of all SNPs with various subset selection approaches. In addition, a genomic animal model was implemented in a Bayesian framework via integrated nested Laplace approximations (INLA, Rue et al., 2009), serving as a state-of-the-art benchmark model. Finally, two ridge regression variants were applied. One used SNPs directly as predictors, and the other was a PCRR model using PCs derived from the SNP matrix. For identifying SNP-trait associations, we compared a GWAS-method called *Genome-wide Efficient Mixed Model Association algorithm* (GEMMA, Zhou and Stephens, 2012) with machine learning-based SHAP values derived from XGBoost. To assess method performance under controlled conditions, we simulated phenotypes according to marker-based regression fitted to the SNP data. Ten traits were simulated across a spectrum of genetic architectures, ranging from highly polygenic to highly oligogenic. These simulated phenotypes were used to evaluate genomic prediction using the XGBoost model fitted on all SNPs, the Bayesian genomic animal model, and both ridge regression variants. SNP-trait associations for the simulated traits were analyzed using SHAP, GWAS, and coefficient-based inference from both ridge and lasso regression. Together, these studies aimed to evaluate XGBoost against established methods for genomic prediction, and to explore alternative approaches to GWAS for identifying SNP-trait associations in complex traits in wild populations. Software and technical considerations, along with all code used in this thesis, is publicly available in the GitHub repository https://github.com/GardGravdal/MasterThesis.

This thesis builds on the work from the project thesis (Gravdal, 2024), and as such, includes overlapping material where the content remains applicable and unchanged.

This thesis contributes to sustainability by improving our understanding of genetic variation and

evolutionary processes in wild populations, supporting the United Nations Sustainable Development Goal 15: Life on Land, which focuses on the protection and sustainable use of terrestrial ecosystems and biodiversity.

2 Background

2.1 The foundations of quantitative genetics

2.1.1 Basic concepts and terminology in quantitative genetics

In most multicellular organisms, each individual inherits two sets of chromosomes, one from each parent, giving them two copies of every gene. A gene is located at a specific position on a specific chromosome, called a locus, and different versions of a gene are known as alleles (Hartl and Conner, 2004). When an individual has two different alleles, one may have a stronger influence on the outward appearance of a trait, dominating the other recessive allele. For example, an allele that codes for the brown eye (iris) color is typically dominant over the alleles that code for green or blue eye color, and green takes precedence over blue (White and Rabago-Smith, 2011). The dominant alleles are typically represented by an uppercase letter (*e.g.*, "A"), while recessive alleles are represented by lowercase letters (*e.g.*, "a") as introduced by Sutton (1903). Since an individual has two sets of chromosomes, for a given gene with two alleles, an individual can have one of three possible combinations of alleles: "AA", "Aa", or "aa", corresponding to dominant homozygous, heterozygous, and recessive homozygous combinations respectively. The specific combination of alleles that an individual carries is referred to as their genotype (Hartl and Conner, 2004).

2.1.2 Partition of phenotypic variance

An individual's outward appearance, known as the phenotype (P), results from the combined effects of the genetic components (G) and environmental factors (E), which mathematically can be expressed as $P = G + E$. Most phenotypes are not determined by alleles at a single locus. Instead, many are continuously distributed and are referred to as quantitative traits, which are influenced by multiple loci and environmental factors. Measuring variation in quantitative traits can be complex because it involves several factors, such as the polygenic basis, pleiotropy, epistatic effects, dominance effects and environmental factors:

- *The polygenic basis:* Refers to a trait having a number of loci and alleles affecting it.
- *Pleiotropy:* The phenomenon of a single gene influencing multiple traits.
- *Dominance effects:* Result from interactions between alleles at the same locus.
- *Epistatic effects:* Result from interactions between genes at different loci.

Additionally, there are often interactions between genes and the environment, known as gene-environment (G×E) interactions, which further complicate the expression of quantitative traits (Hartl and Conner, 2004).

The total phenotypic variation of a quantitative trait, denoted V_P, is often partitioned into two parts, namely the variation due to the genetic component (V_G) and the variation due to the environmental factors (V_E). The variation due to the genetic component can be further decomposed into the variation due to additive genetic effects (V_A), the variation due to epistatic effects (V_I) and the variation due to dominance effects (V_D), which can be expressed as $V_G = V_A + V_I + V_D$ (Hartl and Conner, 2004). Additive genetic effects are the effects on a phenotype that are accumulated by summing over the individual contributions from alleles for a polygenic trait. The *breeding value* is an individual-specific quantity that is defined as the additive effect of an individual's genotype on a trait expressed relative to the population mean phenotype (Wilson et al., 2010). The name stems from the fact that the breeding value is the effect of an individual's genes on the value of a given trait in its offspring (Hartl and Conner, 2004). From these definitions, we can further define the *heritability* as the proportion of the total phenotypic variance that is due to genetic causes, or in other words, the relative importance of genetic variance in determining phenotypic variance.

We define the broad sense heritability as

$$h_B^2 = \frac{V_G}{V_P} \ ,$$

while the narrow sense heritability (or simply heritability) is defined as (Hartl and Conner, 2004)

$$h^2 = \frac{V_A}{V_P} \ .$$

The narrow sense heritability is often preferred to the broad sense heritability since the additive genetic effect provides a measure of the genetic component that can be directly passed on additively from parents to offspring.

2.1.3 Single nucleotide polymorphism

The basic building-blocks in DNA are nucleotides. The bases in DNA are the nucleotides adenine (A), cytosine (C), guanine (G) and thymine (T). The strands of DNA are held together by pairs of these bases, where adenine pairs with thymine and guanine pairs with cytosine (Pu et al., 2018). A SNP (often pronounced "snip") is, as the name suggests, a difference in a single nucleotide (Brookes, 1999). Such mutations are normal and occur approximately every 1000 nucleotide (Sachidanandam et al., 2001). We classify the mutation as a SNP if a variant is found in at least 1% of the population (Karki et al., 2015). The terminology "allele" is used also for SNPs, meaning a variant of a SNP. Most SNPs are biallelic, which means they come in two variants (Sachidanandam et al., 2001). From the biallelic nature of the SNPs, we also inherit the terms heterozygous and homozygous. For example, if a SNP comes in the form of the two nucleotides A and C, we can define the "genotypes" AA, CC and AC. The latter genotype would then be heterozygous, while the other two are homozygous. By using this terminology, we simplify the encoding of the SNPs. The SNP data is typically encoded $\{0, 1, 2\}$, where 0, 1, 2 denote the number of minor alleles in the genotype. The minor allele of a SNP is the second most frequent allele of that SNP. Note that this encoding is consistent with the assumption of additive genetic effects, which is important for the statistical models which will be discussed later in the thesis. We typically describe the position of a SNP at two levels: which chromosome it is located in, and the SNPs base pair (bp) number describing where in the chromosome it is located.

Most SNPs do not have an effect on the phenotype as they fall in the non-coding regions of the DNA, but some SNPs can change gene function or the level of its expression (Brodie et al., 2016). Although most SNPs do not directly have an effect on the phenotype, they are widely used as genome-wide markers because they are highly abundant and they serve effectively as markers for genomic regions due to linkage disequilibrium (LD, Bush and Moore, 2012). LD is the genetic correlation between two or more alleles from the fact that they are inherited together more often than would be expected by random chance (Hartl and Conner, 2004). For example, alleles in loci that are close to each other on the chromosome will tend to be inherited together, since they are less likely to be broken apart by recombination during meiosis. SNPs can thus be genetic markers for regions of DNA associated with the specific trait of interest. Such a DNA region is called a quantitative trait locus (QTL) (Hartl and Conner, 2004). SNPs are thus excellent markers for studying complex traits (Syvänen, 2001; Oraguzie et al., 2008).

2.1.4 Animal model

As outlined in Section 2.1.2, the variation in phenotype for a quantitative trait is complex, and is influenced by several factors such as environmental effects and genetic effects. One of the main goals in quantitative genetics is to understand how the variation in phenotype is distributed, and in particular to estimate V_A, the component of phenotypic variance that is due to the additive genetic effect. The animal model is a LMM which attempts to estimate the distribution of the variability in phenotype for the different components. A LMM attempts to handle different sources of variability in the data stemming from *fixed effects* and *random effects* in situations when the data is grouped in hierarchical structures (Chen and Chen, 2017). Mixed effects models enable us

to partition the variance. Some of the typical hierarchical structures that can arise in an animal model is through permanent environmental effects, relatedness or individual-specific effects such as age or sex (Wilson et al., 2010; Aspheim et al., 2024). The animal model is a LMM where the breeding value is treated as a random effect (Wilson et al., 2010). In the simplest case, when we model a single phenotype y_i on a single individual i, the animal model can be expressed as

$$y_i = \mu + a_i + \epsilon_i \ ,$$

where μ is the population mean for the trait (fixed effect), a_i is the breeding value (random effect), and ϵ_i is an (independent) residual term (random effect). In a LMM, we assume that each random effect originates from some distribution, often Gaussian, with zero mean and unknown variance that is of interest to measure. The variance of the additive genetic effect is defined as V_A. The variance of the residual term, σ^2_ϵ, covers the remaining variance in the phenotype. Assuming both the random effects follow a normal distribution and a sample of n individuals, the vectors $\boldsymbol{a}$ and $\boldsymbol{\epsilon}$ are each of dimension $n \times 1$, with $\boldsymbol{a} \sim N(0, \mathbf{G}V_A)$ and $\boldsymbol{\epsilon} \sim N(0, \mathbf{I}_n\sigma^2_\epsilon)$. Note that $\mathbf{I}_n$ denotes the $n \times n$ identity matrix, such that $\boldsymbol{\epsilon}$ is consistent with the assumption of independence between residual errors. The matrix $\mathbf{G}V_A$ denotes the variance-covariance matrix of breeding values.

Traditionally, pedigrees were used to model the covariance between breeding values of different individuals. In this method, pedigree data is used to calculate the *relatedness matrix* $\mathbf{G}$ for individuals i and j, $G_{ij} = 2\kappa_{ij}$. Here κ_{ij} is the coefficient of coancestry, which is defined as the probability that an allele that is drawn at random from individual i is *identical by descent* (IBD) to one drawn at random from individual j (Wilson et al., 2010). IBD refers to alleles that are descended by DNA replication from a single allele present in an ancestor (Hartl and Conner, 2004). Then the relatedness G_{ij} can intuitively be thought of as the "expected proportion of relatedness", for example 0.5 for a parent and its offspring.

An alternative to the use of pedigree data is to model the covariance of individuals through a *genomic relationship matrix* $\mathbf{G}$ derived from genetic markers such as SNPs. These methods became more popular as genome sequencing became relatively inexpensive and SNP-markers could cover the genome with high density (VanRaden, 2008). Several methods to derive a genomic relationship matrix were proposed by vanRaden, and here we will present what is denoted as method 1 (VanRaden, 2008). Let $\mathbf{M}^*$ be the $n \times m$ matrix of markers (*e.g.*, the SNP matrix) for individuals $i = 1, ...n$ and SNPs $j = 1, ..., m$ encoded as $\{-1, 0, 1\}$ instead of $\{0, 1, 2\}$. The new encoding ensures that the diagonal of the matrix $\mathbf{M}^*(\mathbf{M}^*)^T$ count the number of heterozygotes for each individual, while the off-diagonal elements measures (#equal alleles - #different alleles) for each individual (*i.e.* a measure of similarity). Now we let the frequency of the minor allele be p_i, and we define $\mathbf{P} = \{P_{ij}\}_{ij} \in \mathbb{R}^{n\times m}$ for $P_{ij} = 2(p_i - 0.5)$. We can then define the matrix $\mathbf{C}^* := \mathbf{M}^* - \mathbf{P}$. The transformation by subtracting $\mathbf{P}$ has the effect of giving credit to more rare alleles compared to more common ones in calculating the genomic relationship matrix. Finally, we define the genomic relationship matrix as

$$\mathbf{G} = \frac{\mathbf{C}^*\mathbf{C}^{*T}}{2\sum_{i=1}^{n} p_i(1-p_i)} \ . \tag{1}$$

Scaling by the divisor $2\sum_{i=1}^{n} p_i(1-p_i)$ makes it so that the behavior of $\mathbf{G}$ is analogous to $\mathbf{G}$ calculated by pedigrees. Using the genomic relationship matrix instead of the relatedness matrix leads to the so-called *genomic animal model*.

2.1.5 Genomic prediction

Another main goal of quantitative genetics is to estimate the genetic contribution to the phenotype, and in particular the breeding value, a field of study often called *genomic prediction* (Meuwissen et al., 2001). Estimating breeding values entails extracting the additive genetic effects from the phenotype. For example, an animal breeder is only interested in knowing the breeding value of a trait, and the environment or other components of the phenotype is "noise" that they want to remove to understand the genetic merit of a trait. Animal models can be used to estimate breeding values by employing a method called BLUP (Henderson, 1975).

In short, BLUP is a method that predicts an unknown random variable by a linear function on the observed data (response). BLUP is an unbiased estimator of the random variable, meaning

that the expected value of the predicted value is equal to the expected value of the variable it is estimating. BLUP is the best estimator in the sense that there exists no other unbiased estimator that has smaller variance than the BLUP. In a typical pedigree based animal model, we can express the response $\boldsymbol{y}$ as

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{D}\boldsymbol{r} + \boldsymbol{\epsilon} \ ,$$

where the response $\boldsymbol{y} \in \mathbb{R}^n$ is the phenotype. Further, $\boldsymbol{\beta}$ denotes the vector of fixed effects, such as the population mean $\boldsymbol{\mu}$, with corresponding design matrix $\mathbf{X}$ of appropriate dimension. $\boldsymbol{r} \sim N(\mathbf{0}, \mathbf{V}_r)$ is the vector containing the random effects, such as the breeding value $\boldsymbol{a} \sim N(\mathbf{0}, \mathbf{G}V_A)$, with the corresponding design matrix $\mathbf{D}$ of appropriate dimension. Finally, $\boldsymbol{\epsilon} \sim N(0, \mathbf{I}_n\sigma^2_\epsilon) \in \mathbb{R}^n$ is the vector of residual errors. Assuming we have observed the response $\boldsymbol{y}$, the BLUP of $\boldsymbol{r}$, denoted $\hat{\boldsymbol{r}}$, is given as

$$\hat{\boldsymbol{r}} = E\left[\boldsymbol{r}|\boldsymbol{y}\right] = \left(\mathbf{D}^T\mathbf{V}_r^{-1}\mathbf{D}\right)^{-1}\mathbf{D}^T\mathbf{V}_r^{-1}\boldsymbol{y} \ . \tag{2}$$

The derivation of equation (2) is out of the scope of this thesis, but we refer curious readers to Henderson (1975). If instead a genomic animal model is used, we simply replace the relatedness matrix $\mathbf{G}$ with a genomic relationship matrix such as that of method 1 given by vanRaden in equation (1). If a genomic relationship matrix is used, we call the estimator $\hat{\boldsymbol{r}}$ the GBLUP.

An alternative to the (G)BLUP methods of estimating the breeding value is by regressing the phenotype $\boldsymbol{y}$ directly on all the markers, which leads to the method of marker-based regression (Meuwissen et al., 2001). Now the breeding value a_i is replaced by a sum over all the additive effects of the genome-wide markers, which as we recall is our definition of the breeding value. This model is consistent with the observation that many quantitative traits are polygenic, with small contributions from a large number of loci (Wood et al., 2014). Assuming we have a dataset of m SNPs and n individuals, we can model the phenotype $\boldsymbol{y} \in \mathbb{R}^n$ in marker-based regression as

$$\boldsymbol{y} = \mu\mathbf{1}_n + \boldsymbol{\Gamma}\boldsymbol{b} + \mathbf{M}_c\boldsymbol{u} + \mathbf{Z}\boldsymbol{d} + \boldsymbol{\epsilon} \ , \tag{3}$$

where μ is the population mean, $\mathbf{1}_n \in \mathbb{R}^n$ denotes the vector of ones, and $\boldsymbol{\Gamma}$ is matrix containing fixed effects with the corresponding regression parameter vector $\boldsymbol{b}$ of appropriate dimensions. Moreover, $\mathbf{M}_c \in \mathbb{R}^{n\times m}$ is the matrix of marker codes (SNPs), and where the matrix has been mean-centered such that each column (SNP) sum to zero. The vector $\boldsymbol{u} \sim N(\mathbf{0}, \mathbf{I}_m\sigma^2_u) \in \mathbb{R}^m$ is a random vector containing the allele substitution effects of each SNP, which will also simply be referred to as the vector of SNP effects. Further, $\boldsymbol{d}$ is the vector of random environmental effects with the corresponding design matrix $\mathbf{Z}$ of appropriate dimensions. Finally, $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{I}_n\sigma^2_\epsilon)$ is the vector of residuals. The vector of breeding values is in this case given by $\boldsymbol{a} = \mathbf{M}_c\boldsymbol{u} \in \mathbb{R}^n$ (Aspheim et al., 2024).

It can be shown that the estimated breeding values by equation (3) is identical to the breeding values estimated by GBLUP as long as the same SNPs are used and the effect sizes $\boldsymbol{u}$ all stem from the same distribution (VanRaden, 2008). One issue with the marker-based regression method is that since $m > n$, the linear system is underdetermined, and estimating by standard least squares regression is not possible. Moreover, the genomic datasets used in marker-based regression is typically characterized by having a high dimensionality, where the number of marker-features far exceed the number of individuals. The high-dimensionality of $m \gg n$ introduces several statistical challenges and is a recognized scientific problem in its own right (Van De Geer and Van Houwelingen, 2004).

2.2 Statistical learning

2.2.1 The basics of machine learning

Machine learning can at a low level be described as a set of methods that computers use to make and improve predictions or behaviors based on data (Molnar, 2022). At its core, machine learning is about developing algorithms that can infer mappings from inputs to outputs by optimizing a function based on a dataset. The naming of *learning* was coined by Arthur Samuel (Zhou, 2021), and describes the process of using machine learning algorithms to build models from data. We may also use the term *training* interchangeably. It is customary to set aside an independent set

for testing the model, called the testing set. The set of data not in the testing set, the set that is used in the training phase, is often called the training data, and the set of all the training samples is called the training set. It is worth noting that the performance of a machine learning model is generally speaking very dependent on the sample size: the bigger the sample size, the more data we have for training the model, leading to a better model fit.

In this thesis we will explore methods based on *supervised learning* and *unsupervised learning*, which are subcategories of machine learning (Hastie et al., 2009). In supervised learning, the goal is to learn a predictive model that maps variables, often called covariates, predictors or features, to an output, often called the response. The response works as a label to guide the learning of a predictive model, hence we say that the learning is supervised (Hastie et al., 2009; Nasteski, 2017). Mathematically, the task of supervised learning can be phrased as follows. Given dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathcal{X}$ represents the input features and $y_i \in \mathcal{Y}$ represents the corresponding response, the goal is to find a function $f : \mathcal{X} \mapsto \mathcal{Y}$ that minimizes some predefined loss function, sometimes also referred to as the *objective function*, defined as $L\left(y, f(\mathbf{x}; \boldsymbol{\theta})\right)$, $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$. The function f aims to model the connection between $\mathbf{x}$ and y and contains a set of flexible parameters $\boldsymbol{\theta}$ that are found by minimizing the loss function. Supervised learning can be formulated as the following optimization problem

$$\hat{y} = f(\mathbf{x}; \hat{\boldsymbol{\theta}}), \quad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} L\left(y, f(\mathbf{x}, \boldsymbol{\theta})\right) \ . \tag{4}$$

In unsupervised learning, the goal is to learn patterns in a dataset $\mathcal{D}$, often consisting of features or covariates $\mathbf{X} \in \mathcal{X}$, without a label in form of a response $\boldsymbol{y}$. A consequence of the lack of a reference in a response is that it is difficult to construct a test set that provides an objective criterion for evaluating unsupervised learning methods. As a result, unsupervised learning methods typically capture more exploratory patterns in the data, such as similarities, differences and other relationships and hidden patterns between the features in the dataset.

The field of machine learning encompasses many different models which differ in many ways, such as the choice of loss function $L(\cdot)$ and choice of function $f(\cdot)$. The aim of the loss function is to provide a measure of how well or poorly a model $y^* = f(\mathbf{x}, \boldsymbol{\theta}^*)$ is performing by assigning a penalty (or "loss"). This will allow the model to adjust its parameters in order to minimize the loss, which is designed to minimize the test error and thus improve the performance on unseen data, *i.e.* the data that is not in the training set. Some loss functions contain *regularization* parameters, which are tools used by machine learning algorithms to reduce the error in the test data. The idea behind using regularization in order to minimize the test error is closely linked to the concept of the bias-variance trade-off, which underlines that the bias and the variance make up the reducible part of the expected prediction error, also known as the generalization error (Hastie et al., 2009). Generally speaking, as the model complexity increases, the (squared) error due to bias decreases while the error due to variance increases since the model better fits the training data, leading to a trade-off between bias and variance. Regularization aims to find the correct balance between bias an variance, and in the process tries to avoid overfitting the testing data by "punishing" more complex or flexible models in order to generalize the model.

There exists many different choices of $f(\cdot)$, all with their own strengths and weaknesses. Some functions may be better at capturing linear relationships, such as the regular linear regression model, while others may be better at capturing non-linear effects or interactions between features, such as regression trees. The best choice of function $f(\cdot)$ and loss function $L(\cdot)$ is dependent on the dataset $\mathcal{D}$ and the task at hand. A model that performs well on one task might perform poorly on another. The fact that there exists no optimal machine learning model that can perform well in any given task or situation is reflected in the "No free lunch" theorem (Zhou, 2021). The conclusion to take from the theorem is that we need to understand the data we are working with and the task at hand before we can decide which machine learning model is the best choice.

2.2.2 Classical linear regression methods: ridge and lasso

The linear regression model is one of the oldest and most fundamental supervised learning techniques, and its origins can be dated back to the introduction of OLS by Carl Fredrich Gauss

and Adrien-Marie Legendre (Echagüe and Belenkiy, 2008). Suppose we have the dataset $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$, where $y_i \in \mathbb{R}$ is the response and $\boldsymbol{x}_i \in \mathbb{R}^m$ are the covariates, for individuals $i = 1, ..., n$. The linear regression model assumes a linear relationship between the response and covariates, which can be expressed as

$$y_i = x_{i1}\beta_1 + x_{i2}\beta_2 + ... + x_{im}\beta_m + \epsilon_i \ ,$$

or in matrix form, as

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \ . \tag{5}$$

Here, $\mathbf{X} = \{x_{ij}\}_{ij} \in \mathbb{R}^{n\times m}$ is the matrix of covariates and $\boldsymbol{\beta} \in \mathbb{R}^m$ is the vector of regression coefficients. Further, $\boldsymbol{y} \in \mathbb{R}^n$ represents the response vector and $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{I}_n\sigma_\epsilon^2) \in \mathbb{R}^n$ is the vector of independent residual errors. In linear regression, the estimator of $\boldsymbol{\beta}$, $\hat{\boldsymbol{\beta}}_{OLS}$, is derived from OLS and is the solution to the following optimization problem

$$\hat{\boldsymbol{\beta}}_{OLS} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} |y_i - \sum_{j=1}^{m} x_{ij}\beta_j|^2 = \arg\min_{\boldsymbol{\beta}} \|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \ .$$

It can be shown that the unique solution to the optimization problem is given as

$$\hat{\boldsymbol{\beta}}_{OLS} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\boldsymbol{y} \ . \tag{6}$$

Furthermore, it can be shown that the estimator $\hat{\boldsymbol{\beta}}_{OLS}$ is the best linear unbiased predictor, which explains the appeal of the linear regression model.

One of the assumptions of the linear regression model is that there is no multicollinearity, meaning that the covariates are not highly correlated. When the covariates are highly correlated, the matrix $\mathbf{X}^T\mathbf{X}$ will be nearly singular and ill-conditioned, leading to an unstable OLS estimator $\hat{\boldsymbol{\beta}}_{OLS}$, where the estimates can vary widely with small changes in input data. Furthermore, when the number of covariates m exceed the number of samples n, $m > n$, the OLS solution lacks a unique solution. To address these limitations, various regularized linear regression methods have been developed.

One such method is ridge regression, a simple and classical regularization technique first proposed by Hoerl and Kennard (1970) to improve linear regression performance in the presence of multicollinearity. Ridge regression addresses multicollinearity by adding an ℓ_2 regularization term to the objective function, resulting in the following optimization problem

$$\arg\min_{\boldsymbol{\beta}} \{\|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda_R\|\boldsymbol{\beta}\|_2^2\}, \ \lambda_R \geq 0 \ , \tag{7}$$

where λ_R is a regularization parameter that controls the strength of the penalty. It can be shown that the optimization problem has the unique solution (Hoerl and Kennard, 1970)

$$\hat{\boldsymbol{\beta}}_R = \left(\mathbf{X}^T\mathbf{X} + \lambda_R\mathbf{I}_m\right)^{-1}\mathbf{X}^T\boldsymbol{y} \ . \tag{8}$$

The parameter λ_R has a shrinkage effect on the regression coefficients, pulling them toward zero. The regularization introduces a bias in the estimate of $\boldsymbol{\beta}$, so $\hat{\boldsymbol{\beta}}_R$ is a biased estimator of $\boldsymbol{\beta}$. However, adding λ_R to the diagonal elements of $\mathbf{X}^T\mathbf{X}$ resolves the near-singularity and ill-conditioning of this matrix. As a result, the ridge estimator $\hat{\boldsymbol{\beta}}_R$ has lower variance and is more stable than the OLS estimator $\hat{\boldsymbol{\beta}}_{OLS}$. In summary, ridge regression employs a different strategy than linear regression by introducing a bias to reduce variance, aiming to achieve a better balance in the bias-variance trade-off. Often, this leads to improved prediction accuracy, particularly when some covariates are correlated. It is worth noting that before tuning and fitting regularized models such as ridge regression, it is customary to center the covariates. That is, to subtract the mean of each covariate from its values so that they are centered around zero. Centering helps prevent the model from attributing predictive power to differences in covariate means, thereby avoiding potential bias in the coefficient estimates.

An alternative interpretation of ridge regression can be derived from a Bayesian perspective. To describe the Bayesian perspective, we first have to cover some of the basics in Bayesian statistics. The Bayesian approach to statistics is an alternative to the "classical" frequentist approach

(Bolastad, 2009). In the Bayesian approach, the laws of probability are applied directly to the problem. For any continuous random variables X and Y, the conditional probability density of X given $Y = y$ is given as

$$f(x|y) = \frac{f(y|x)f(x)}{f(y)} \ .$$

In the Bayesian approach, the probabilities are extended to include the prior belief in different values of unknown parameters θ, and then updated via Bayes theorem such that

$$f(\theta|x) \propto f(x|\theta)f(\theta) \ , \tag{9}$$

where the updated probability distribution $f(\theta|x)$ is called the *posterior probability distribution*, or simply the posterior distribution, $f(x|\theta)$ is the *likelihood* and $f(\theta)$ is the *prior probability distribution*, or simply the prior.

Having derived some of the basics of Bayesian statistics, we can now turn our attention to Bayesian perspective of ridge regression. For ridge regression the optimization problem in equation (7) can be viewed as the result of a maximum posteriori (MAP) estimation under a Gaussian prior on $\boldsymbol{\beta}$. First, we assume $\boldsymbol{y}$ follows the linear model given equation (5), or in other words, we have the likelihood

$$\boldsymbol{y}|\boldsymbol{\beta}, \sigma^2, \mathbf{X} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}_n) \ , \tag{10}$$

with independent Gaussian residual errors $\epsilon \sim N(\mathbf{0}, \sigma^2\boldsymbol{I}_n)$. Further, we place a Gaussian prior on $\boldsymbol{\beta}$ of the form

$$\boldsymbol{\beta}|\tau^2 \sim N(\mathbf{0}, \tau^2\mathbf{I}_m) \ .$$

To obtain the posterior distribution, we apply Bayes theorem, and find that

$$p(\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2, \tau^2) \propto p(\boldsymbol{y}|\boldsymbol{\beta}, \mathbf{X}, \sigma^2)p(\boldsymbol{\beta}|\tau^2) \ .$$

Maximizing the posterior distribution is equivalent to minimizing the negative log-posterior, which up to an additive constant gives the following expression

$$-\log p(\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2, \tau^2) \propto \frac{1}{2\sigma^2}\|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \frac{1}{2\tau^2}\|\boldsymbol{\beta}\|_2^2 \ .$$

Minimizing the expression in terms of $\boldsymbol{\beta}$ leads to an objective function equivalent to the ridge regression optimization problem defined in equation (7), where the regularization parameter is given by $\lambda_R = \frac{\sigma^2}{\tau^2}$.

Another approach to addressing multicollinearity in linear regression is lasso regression, which was suggested by Tibshirani (1996) as an alternative to ridge regression that offers a more interpretable model that also performs variable selection. Similar to ridge regression, lasso adds a regularization term to the OLS objective function. However, while ridge uses the ℓ_2-norm penalty, lasso employs the ℓ_1-norm, resulting in the following optimization problem (Tibshirani, 1996)

$$\arg\min_{\boldsymbol{\beta}} \ \{\|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda_L\|\boldsymbol{\beta}\|_1\}, \ \lambda_L \geq 0 \ , \tag{11}$$

where λ_L is a tuning parameter controlling the strength of the regularization, and the regularization term is defined as $\lambda_L\|\boldsymbol{\beta}\|_1 = \lambda_L \sum_{j=1}^{m} |\beta_j|$.

The different regularization techniques lead to varying characteristics in the ridge and lasso regression models. Unlike ridge regression, lasso does not generally have a closed-form solution because the ℓ_1-norm introduces a non-linear and non-differentiable term in the objective function. However, for most datasets, lasso has a unique solution, and can be efficiently computed using optimization techniques (Tibshirani, 2013). For detailed methods on computing the lasso estimate, the reader is referred to Tibshirani (1996). A key advantage of the ℓ_1-norm compared to the ℓ_2-norm is its ability to shrink some coefficients exactly to zero when the regularization parameter λ_L is sufficiently large, effectively performing variable selection. Consequently, lasso regression often results in a more interpretable model, while retaining a degree of stability similar to ridge regression.

From a Bayesian perspective, the lasso estimate can be interpreted as a MAP estimate under a Laplace prior for $\boldsymbol{\beta}$. Specifically, assume the same linear model as in ridge regression given in equation (5), leading to the same likelihood as in equation (10). Placing a Laplace prior on $\boldsymbol{\beta}$,

$$p(\boldsymbol{\beta}|\tau) \propto \exp(-\frac{1}{2\tau}\|\boldsymbol{\beta}\|_1) \ ,$$

and assuming independence between the coefficients, let $\lambda_L = \frac{\sigma^2}{\tau}$, and we get a MAP for $\boldsymbol{\beta}$ that minimizes equation (11).

In this thesis, we performed genomic prediction using datasets characterized by high dimensionality ($m \gg n$) and often a large number of SNP-covariates with small effects. In this environment, ridge regression is the preferred model choice, as previous findings show (*e.g.*, Tibshirani, 1996). Lasso regression, on the other hand, tends to perform better in a setting where a relatively small number of predictors have substantial effects, and the remaining predictors have effects that are negligible or zero (James et al., 2013). Therefore, ridge regression is generally more suited than lasso for genomic prediction. Beyond prediction, inference in genomic data is often of interest, where the goal is inferring associations between SNP-covariates and the phenotypic response. For inference purposes, the lasso model is often more suitable, as it offers better interpretation from the variable selection feature. However, due to the high-dimensional nature of genomic data and the assumed presence of many small-effect SNPs, ridge regression may still provide more reliable inference, particularly for polygenic traits.

2.2.3 Principal Component Analysis

PCA is a widely used unsupervised learning technique. PCA, like other unsupervised learning methods, identifies structure in the data without reference to a response variable. Unsupervised learning methods like PCA rely on the covariates to find patterns such as correlations, clusters or directions of maximum variance. PCA can be useful in many applications, and some of its uses include

- Visualizing the data while highlighting important patterns.
- Pre-processing data before other statistical tasks.
- Reducing the dimensionality in the data while retaining the useful information.

Assume we have a dataset of observed covariates $\mathbf{X} \in \mathbb{R}^{n\times m}$, where we have observed m covariates for n individuals. The task in PCA is to transform the covariate space into a new coordinate system where the axes of the new coordinate system point in the directions of maximum variance in the data. The data can then be projected onto the new coordinate system, generating the PCs, which can display the pattern of similarity of the observations as points in maps (Abdi and Williams, 2010). The process can be described as follows.

The first step in PCA is to center the data such that the mean of each column of $\mathbf{X}$ is zero. Without centering, the resulting PCs will pick up a bias from the mean shift of the covariates instead of the intrinsic variability in the data, and centering ensures that the PCs align with the true directions of maximum variance. From each column mean $\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$, $j = 1, ..., m$, we define the column mean vector $\bar{\boldsymbol{x}} = \{\bar{x}_j\}_{j=1}^{m}$, and the centered matrix $\mathbf{X}_c$ is defined as

$$\mathbf{X}_c = \mathbf{X} - \mathbf{1}_n\bar{\boldsymbol{x}}^T \ .$$

We now state without proof that the eigenvectors of the (sample) covariance matrix of $\mathbf{X}_c$ form the orthonormal basis that points in the directions of maximum variance in $\mathbf{X}_c$. For a proof of this statement, see Appendix B. We first define the covariance matrix of $\mathbf{X}_c$ as

$$\mathbf{C}_x := \text{Cov}(\mathbf{X}_c) = \frac{1}{n-1}\mathbf{X}_c^T\mathbf{X}_c \ .$$

The next step is to diagonalize the matrix $\mathbf{C}_x$ in terms of its eigenvectors, sometimes also referred to as performing an *eigen-decomposition*, which for the symmetric matrix $\mathbf{C}_x$ is defined as

$$\mathbf{C}_x = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \ , \ \mathbf{V} = \{\boldsymbol{v}_j\}_{j=1}^m \ ,$$

where $\boldsymbol{v}_j \in \mathbb{R}^m$ are the eigenvectors of $\mathbf{C}_x$ and $\mathbf{\Lambda} = \text{diag}(\lambda_j)$, $j = 1, ..., m$, is the diagonal matrix of eigenvalues, where the eigenvalues are ordered such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$. The eigenvectors that form the columns of $\mathbf{V}$ define a new orthonormal basis, and we can find the *principal component score matrix* $\mathbf{W}$ by projecting $\mathbf{X}_c$ onto $\mathbf{V}$

$$\mathbf{W} = \mathbf{X}_c\mathbf{V} \in \mathbb{R}^{n \times m} \ .$$

The columns of $\mathbf{W}$ correspond to the PCs $\boldsymbol{w}_j$, $j = 1, ..., m$. PCA can be interpreted geometrically in the following way

- $\boldsymbol{v}_j$: Axis j after the transformation, where $\boldsymbol{v}_1$ points in the direction of most variance in the data, $\boldsymbol{v}_2$ points in the direction of most variance in the data subject to being orthogonal to $\boldsymbol{v}_1$, and so on, such that $\mathbf{V}$ defines an orthonormal basis.
- $\boldsymbol{w}_j$: PC number j, the projection of $\mathbf{X}_c$ onto $\boldsymbol{v}_j$.
- λ_j: Eigenvalue associated with eigenvector $\boldsymbol{v}_j$, the proportion of variance explained by $\boldsymbol{w}_j$ (for proof, see Appendix B).

In summary, PCA rotates the original coordinate system to the directions of highest variance in $\mathbf{X}_c$. The PCs are useful if we want to perform dimensionality reduction on the original data $\mathbf{X}$, as we may choose only the first $k \leq \min\{n, m\}$ PCs, which in the case of $m \geq n$ are defined as

$$\mathbf{W}_k := \mathbf{X}_c\mathbf{V}_k \in \mathbb{R}^{n \times k} \ , \ \mathbf{V}_k = \{\boldsymbol{v}_j\}_{j=1}^k \ .$$

The proportion of total variance explained by the first k PCs can then be calculated by

$$\frac{\sum_{j=1}^k \lambda_j}{\sum_{j=1}^m \lambda_j} \ .$$

If we set $k = \min\{n, m\}$, we can transform the original data $\mathbf{X}$ from dimension $(n \times m)$ to $(k \times k)$ while retaining all the variance in the data, which can serve as a very useful pre-processing step in a lot of statistical methodologies. In this thesis, PCA will be performed on SNP-covariates from a genomic dataset, effectively performing dimensionality reduction on the covariate space. The resulting $k = n$ PCs will then be utilized as covariates in a ridge regression model in a genomic prediction setup.

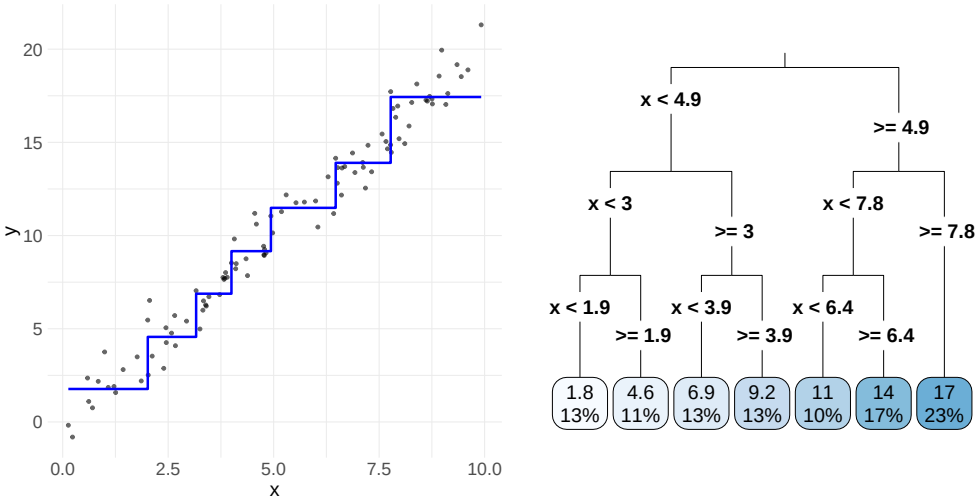
Note that in implementations, PCA is usually done by singular-value decomposition (SVD) of $\mathbf{X}$ instead of eigenvalue decomposition of the covariance matrix. While the resulting PC scores will be identical, the SVD approach is much more memory efficient, since it negates the need to calculate the $(m \times m)$ covariance matrix, which quickly becomes computationally expensive for large dimensions. For the SVD approach of PCA, we refer the reader to Abdi and Williams (2010).

2.2.4 Regression trees

Tree-based methods were developed from a simple and intuitive concept. The tree-based methods partition the feature space into rectangles called regions R, and then fit a simple constant model $c \in \mathbb{R}$, which in regression tasks is typically the mean of each sample in the region, in that region. An illustration of a regression tree is seen in Figure 1a. To model a tree-based method, we need a rule on how to split the feature space and create the different regions. Most tree-based methods use a process called recursive binary splitting (Hastie et al., 2009). This is a greedy approach that chooses which feature and split-point that "best fits" the response $\boldsymbol{y}$ in each step, where the best fit is decided by the chosen loss-function. In regression, the loss-function aims to minimize the variance in the response $\boldsymbol{y}$ (Molnar, 2022). The splitting into regions continues until some stopping

criterion is met. The greedy approach does not guarantee globally optimal solution, but it can provide a good approximate model.

The terminology of "tree" stems from the fact that each tree can be visualized by a figure called the *binary decision tree*. These figures provide an intuition for the tree model. A binary decision tree should be read from the top down, such that we envision the "root" of the tree at the top. Each node in the tree represent a split in the feature space, and the leaves at the end of the tree are called leaf nodes or terminal nodes. Each leaf node represent a region in the feature space, and each sample falls into exactly one leaf node (Molnar, 2022). Figure 1b illustrates the decision tree corresponding to the regression tree in Figure 1a. In this thesis we will be working



(a) Regression tree

(b) Decision tree

Figure 1: A simple illustration of a regression tree with corresponding decision tree. Data is 100 samples drawn from $\boldsymbol{y} \sim 2\boldsymbol{x} + \boldsymbol{\epsilon}$, where $x_i \sim \text{unif}(0, 10),\ \epsilon_i \sim N(0, 1),\ \forall\ i = 1, ..., 100$.

with a continuous response and the task is regression. We can model the response of a regression tree model through the m variables $x_{i1}, x_{i2}, \ldots, x_{im}$ and the responses y_i, $i = 1, ...n$, where n is the sample size. Suppose we have partitioned into regions $R_1, R_2, ..., R_J$ each with constants c_j, $j = 1, 2, ..., J$. We can then model the response as

$$f(\boldsymbol{x}_i) = \sum_{j=1}^{J} c_j I(\boldsymbol{x}_i \in R_j)\ ,$$

where $I(\cdot)$ denotes the indicator function (Hastie et al., 2009).

As with any supervised learning method, the bias-variance trade-off needs to be considered when fitting a regression tree model. Regression trees are very prone to overfitting when the tree grows too large. So far we have not discussed any stopping criteria for the regression trees. In the next section we will discuss boosted regression trees which use regression trees as weak learners, and there we discuss some of the methods used to avoid overfitting. One advantage of the regression tree model is that they are ideal in capturing the interactions between features in the model. On the other hand, trees fail to deal with linear relationships (Molnar, 2022). In the regression tree model, a linear relationship between an input feature and the response is approximated by splits, leading to an inefficient "step function" in the feature space, something that is clearly visible in Figure 1a, where a linear regression model likely would be a better choice. Regression trees also have inherent lack of smoothness, such that slight changes in input feature can lead to big impact on the predicted outcome.

2.2.5 Boosted regression trees

Boosting is a supervised learning technique that involves sequentially building simple models, often referred to as weak learners, with each new weak learner "learning" from the previous one. The final model is a weighted sum of all the weak learners, which together fits a powerful ensemble model (Hastie et al., 2009). We begin by introducing a simple illustrative boosting scheme. Assume we have the dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where y_i is the response and $\boldsymbol{x}_i$ the corresponding covariates. We start with a simple model, often simply the mean of the response $F_0(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^n y_i = \bar{y}$. We fit the next model $f_1(\boldsymbol{x})$ on the residuals of the previous model $y - F_0(\boldsymbol{x})$. However, in doing so, we run the risk of overfitting the data, especially as we keep sequentially adding more weak learners in the same way. The solution is to constrain the contribution of each weak learner by scaling it with a constant $\eta \in [0, 1]$ called the *learning rate*. For each step k, we get the composite model $F_k(\boldsymbol{x}) = F_{k-1}(\boldsymbol{x}) + \eta f_k(\boldsymbol{x})$. Hence, the final model is given as

$$F_K(\boldsymbol{x}) = F_0(\boldsymbol{x}) + \sum_{k=1}^{K} \eta f_k(\boldsymbol{x}) \ . \tag{12}$$

Most boosting algorithms follow the fundamental principles of this illustrative scheme, but they differ in several key aspects, particularly in the choice of weak learners $f(\cdot)$. Modern boosting algorithms have evolved to include several sophisticated strategies for managing the bias-variance trade-off, and are even better at balancing flexibility with generalization to avoid overfitting.

Boosted regression trees is a boosting technique which uses simple regression trees as weak learners to fit the ensemble model. A tree-ensemble model extends the general framework in equation (12) by utilizing regression trees as weak learners. Given the dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}$ ($|\mathcal{D}| = n, \boldsymbol{x}_i \in \mathbb{R}^m, y_i \in \mathbb{R}$), with sample size n and m covariates, and assuming the model incorporates K regression trees, the resulting predictive model can be expressed as

$$\hat{y}_i = f(\boldsymbol{x}_i) = \sum_{k=1}^{K} f_k(\boldsymbol{x}_i), \ \ f_k \in \mathcal{F} \ , \tag{13}$$

where we define the space of regression trees as $\mathcal{F} = \{f(\boldsymbol{x}) = \omega_{q(\boldsymbol{x})} \mid q : \mathbb{R}^m \mapsto \{1, 2, ...J\}, \ \omega : \{1, 2, ..., J\} \mapsto \mathbb{R}\}$ (Chen and Guestrin, 2016). Here, $q(\boldsymbol{x}_i)$ represents the tree structure itself, mapping the covariates $\boldsymbol{x}_i$ to the corresponding leaf node. Number of leaves in the tree is denoted J. We can then interpret $f_k(\boldsymbol{x}_i)$, $\boldsymbol{x}_i \in R_j$, as the independent tree structure $q(\boldsymbol{x}_i)$ with corresponding leaf score (weight) ω_j (Chen and Guestrin, 2016).

2.2.6 XGBoost model

XGBoost extends the principles of boosted regression trees described in Section 2.2.5 by incorporating several improvements and layers of complexity. XGBoost uses a regularized loss function that makes it less prone to overfitting. The regularized loss function is also twice differentiable, such that XGBoost can utilize second-order gradients to refine the optimization process. XGBoost also includes techniques like tree pruning and parallelized computation for faster model training. These innovations make XGBoost a powerful and scalable boosting framework. In this section we will go through some of the specific implementation details of XGBoost derived by Chen and Guestrin (2016). Assuming XGBoost follows the predictive model given in equation (13), the regularized loss function used by XGBoost, hereby the *objective function*, is given in general form as

$$\mathcal{L}(y, f(\boldsymbol{x})) = \sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i)) + \sum_{k=1}^{K} \Omega(f_k) \ , \tag{14}$$

where Ω denotes the regularization parameter defined as

$$\Omega(f) = \gamma J + \frac{1}{2}\lambda \|\omega\|_2^2 \ .$$

Here $\|\cdot\|_2$ denotes the ℓ_2-norm. The hyperparameter γ controls the complexity of the model by penalizing each additional leaf node in a tree, thus effectively acting as a threshold for the minimum

improvement in the loss reduction required to add a new leaf. The hyperparameter λ is the ℓ_2 regularization parameter, while ω is the leaf weight assigned to a specific leaf in the tree. The loss function $L(\cdot)$ given in equation (14) is some twice differentiable loss function.

To minimize the objective function, XGBoost utilizes its twice differentiable loss function by implementing a version of Newton's method. Assuming we are at the k'th boosting step, the composite objective function is given as

$$\mathcal{L}^{(k)} = \sum_{i=1}^{n} L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i) + f_k(\boldsymbol{x_i})\right) + \Omega(f_k) \ .$$

In order to decrease the loss, we utilize Newton's method to find the next suitable f_k. The loss function is approximated by a Taylor series expansion

$$L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i) + f_k(\boldsymbol{x}_i)\right) \approx L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i)\right) + g_i f_k(\boldsymbol{x}_i) + \frac{1}{2} h_i f_k(\boldsymbol{x}_i)^2 \ ,$$

where $g_i = \partial_{f^{(k-1)}(\boldsymbol{x}_i)} L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i)\right)$ and $h_i = \partial^2_{f^{(k-1)}(\boldsymbol{x}_i)} L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i)\right)$ denote the gradient and Hessian respectively, with respect to the current prediction. Substituting the loss function by its quadratic Taylor expansion in the neighborhood of $f^{(k-1)}(\boldsymbol{x}_i)$ and removing the constant term $L\left(y_i, f^{(k-1)}(\boldsymbol{x}_i)\right)$ gives the following simplified objective function at boosting step k

$$\tilde{\mathcal{L}}^{(k)} = \sum_{i=1}^{n} \left[g_i f_k(\boldsymbol{x}_i) + \frac{1}{2} h_i f_k(\boldsymbol{x}_i)^2 \right] + \Omega(f_k) \ . \tag{15}$$

Having derived the simplified objective, we are closer to our two main goals, namely to find the optimal weight for each leaf ω_j^* and to find a good splitting criterion. To find the optimal weights, we start by defining the index set I_j as the set of indices i that are in the j'th leaf node of f_k, or more formally, $I_j = \{i \mid q(\boldsymbol{x}_i) = j\}$. By expanding $\Omega(f_k)$ and writing equation (15) at leaf level, we get the following expression

$$\begin{aligned}
\tilde{\mathcal{L}}^{(k)} &= \sum_{i=1}^{n} \left[g_i f_k(\boldsymbol{x}_i) + \frac{1}{2} h_i f_k(\boldsymbol{x}_i)^2 \right] + \gamma J + \frac{1}{2} \sum_{j=1}^{J} \omega_j^2 \\
&= \sum_{j=1}^{J} \left[\omega_j \left(\sum_{i \in I_j} g_i \right) + \frac{1}{2} \omega_j^2 \left(\sum_{i \in I_j} h_i + \lambda \right) \right] + \gamma J \ .
\end{aligned}$$

Assuming a fixed tree structure $q(\boldsymbol{x})$, we can find the optimal weight ω_j^* for a general region R_j by differentiating and equating to zero

$$\begin{aligned}
0 &= \partial_{\omega_j} \sum_{j=1}^{J} \left[\omega_j \left(\sum_{i \in I_j} g_i \right) + \frac{1}{2} \omega_j^2 \left(\sum_{i \in I_j} h_i + \lambda \right) \right] + \partial_{\omega_j} \gamma J \\
&= \sum_{i \in I_j} g_i + \omega_j \left(\sum_{i \in I_j} h_i + \lambda \right) \\
&\Rightarrow \omega_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \ .
\end{aligned}$$

We now move on to the goal of finding a good splitting criterion. Having found the optimal weight ω_j^*, we can find the optimal loss by inserting ω_j^* into equation (15), giving us the expression

$$\tilde{\mathcal{L}}_*^{(k)} = -\frac{1}{2} \sum_{j=1}^{J} \frac{\left(\sum_{i \in I_j} g_i \right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma J \ . \tag{16}$$

Finding the optimal split is a very demanding optimization task. Instead, a greedy approach of recursive binary splitting is used. Here, the objective $\tilde{\mathcal{L}}_*^{(k)}$ from equation (16) becomes useful as

it provides a way to measure the "quality" of a tree structure $q(\boldsymbol{x})$ (Chen and Guestrin, 2016). Splits can then be evaluated by comparing the objective $\tilde{\mathcal{L}}_*^{(k)}$ before and after a split. We take the greedy approach of choosing the split with the biggest reduction in the objective $\tilde{\mathcal{L}}_{(k)}$. Formally, for a given node j, let I_j denote the indices of the node. We propose a split for the node into left and right child nodes with I_L and I_R as their respective index sets, such that $I_j = I_L \cup I_R$. The total loss before the split is given by

$$\mathcal{L}_{before} = -\frac{1}{2}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma \ .$$

The total loss after the split is

$$\begin{aligned}\mathcal{L}_{after} &= \mathcal{L}_{(L)} + \mathcal{L}_{(R)} \\ &= -\frac{1}{2}\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} - \frac{1}{2}\frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} + 2\gamma \ .\end{aligned}$$

Hence, the total change in loss (*i.e.* the gain) from making this split is

$$\mathcal{L}_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda}\right] - \gamma \ ,$$

which is greedily maximized in the top-down greedy binary splitting approach. For more specific implementation details of XGBoost, the reader is referred to Chen and Guestrin (2016).

2.2.7 Hyperparameters of XGBoost

In this section we will look at the some of the most important hyperparameters ($\boldsymbol{\theta}$) of XGBoost. As XGBoost is a tree ensemble, the number of trees to include in the ensemble is an important factor in the model. The number of trees is set by the hyperparameter *n_estimators* in XGBoost. As a general rule, the more trees we include, the more reliable the predictions of the model are. However, the number of trees necessary for a good result is highly dependent on the hyperparameter learning_rate (η). The concept of learning rate and its function to constrain the contribution of each tree to the final model in order to reduce overfitting was already introduced in Section 2.2.4. We can metaphorically think about the relationship of these variables as if walking a fixed distance, where learning_rate is the step size and n_estimators is the number of steps. When increasing the number of estimators (trees), the learning rate should decrease. Increasing the number of trees will also increase the time complexity, which is the constraining factor when choosing a good value for n_estimator.

The hyperparameter *subsample* is a concept that is borrowed from the random forest model, among other models. It decides the proportion of data used for building each tree in the model. The subset of data points is chosen at random. By fitting each tree on only a subset of the data drawn at random, we ensure a diverse tree sample with less correlation between trees. Without subsample, we risk that each tree will be similar since they, after all, use the same greedy approach of choosing a split. Subsample thus helps combat overfitting. The hyperparameter *colsample_bytree* is similar, except it decides the proportion of randomly sampled covariates to train each tree on. The use of colsample_bythree, sometimes also called feature bagging, has similar effects as subsample, and allows us to explore more of the covariate space. Feature bagging may also help detect outliers (Lazarevic and Kumar, 2005).

The hyperparameter *max_depth* decides the maximum depth that a tree can grow. The deeper a tree grows, the higher the complexity and probability of overfitting. However, a deeper tree with more decision paths will allow the model to capture more patterns in the data. Somewhat correlated to the max_depth is the hyperparameter *min_child_weight*, which in a regression setting decides the minimum number of data points allowed in a leaf node. As a tree keeps making splits

and growing in depth, there will be fewer and fewer data points in each leaf node. Sometimes it may be better to allow a tree to grow deep and keep exploring specific patterns, and instead fix the minimum number of data points allowed in a leaf node to prevent overfitting.

Hyperparameters are the parameters that are set before training of a model begins and they define a lot of the internal structure of the model. They are crucial in machine learning algorithms like XGBoost as they control the learning process of the model. Tuning of hyperparameters is a process that is done prior to fitting a model and is a topic that is covered in Section 2.2.8.

2.2.8 Bayesian optimization

The analysis of hyperparameters for XGBoost in Section 2.2.7 underlines that choosing the right set of hyperparameters is important for the performance of a model. In addition, when it comes to optimizing the objective function, the hyperparameters are often highly correlated. Finding the optimal set of hyperparameters, commonly referred to as *hyperparameter tuning*, therefore leads to a complicated optimization problem. Figure 2 shows a simple simulated example of what such an optimization problem might look like in a two-dimensional hyperparameter space. One naive way of searching for a good set of hyperparameters is to simply evaluate the objective function on a uniform grid in some finite subspace of the hyperparameter space. This approach suffers from the curse of dimensionality (Kuo and Sloan, 2005), as the number of function evaluations needed grows exponentially with increasing number of dimensions d. Another approach is to choose a fixed number of function evaluations, then uniformly randomly choose some points in a subspace of the hyperparameter space to evaluate, and finally choose the set of hyperparameters with the smallest objective value. However, this approach is also inefficient, as it will treat all the grid points the same way regardless of their relevance to the objective function. One can thus wonder whether it would not be better to have a method that learns from previous evaluations to guide the future sampling. In fact, this is the idea behind *Bayesian optimization*.

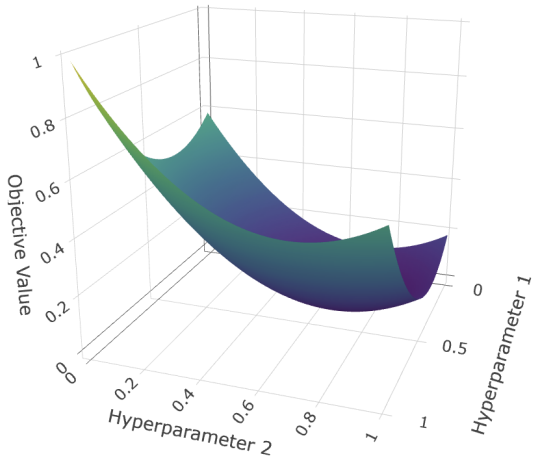


Figure 2: Simulated example of hyperparameter space $(\theta_1, \theta_2) \in [0, 1]^2$ with the objective function $\mathcal{L}(\theta_1, \theta_2) = (\theta_1 - 0.7)^2 + (\theta_2 - 0.3)^2$.

Bayesian optimization is a method that can be used in optimizing the objective function in machine learning algorithms in order to tune the hyperparameters. The methods in Bayesian optimization leverage the previously evaluated objective points to strategically select the new point for evaluation, and thereby improving efficiency (Frazier, 2018). The problem that Bayesian optimization tries to solve can be expressed through the following optimization problem

$$\arg\max_{\boldsymbol{\theta} \in H} \ \mathcal{L}(\boldsymbol{\theta}) \ , \tag{17}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ are the hyperparameters we want to tune. Moreover, $H \subset \mathbb{R}^d$ is the subset in which we will search for the optimal hyperparameters, and is often called the *search space*. Bayesian

optimization is typically a good choice when the objective $\mathcal{L}$ is "expensive" to evaluate in terms of time complexity and when the dimension d is not very small. If d is small and the objective "cheap" to evaluate, the naive grid method might be a good alternative since we can evaluate $\mathcal{L}(\cdot)$ on a refined grid which when minimized should give a reasonably good approximation for the best hyperparameters. An advantage of Bayesian optimization is that it only needs evaluations of the objective, and it does not require the objective to be differentiable. This makes the Bayesian optimization technique flexible, as many objective functions in machine learning are not differentiable. If Bayesian optimization is used to tune the hyperparameters of a machine learning model, $\mathcal{L}(\cdot)$ in equation (17) can be interpreted as the negative of the objective function, as the aim is to minimize the objective function in this case.

We can describe the Bayesian optimization process in two steps. The first step is to create a computationally cheap approximation of the objective $\mathcal{L}(\cdot)$, often called a surrogate model (Frazier, 2018). We build the surrogate model from function evaluations of the objective function. The surrogate model provides predictions with mean and variation in the whole domain of interest. The second step is to evaluate the *acquisition function*. The aim of the acquisition function is to provide a "score" for each point in the domain by using the predictions of the surrogate model, and then the next point to evaluate is decided by maximizing the score. Since the goal in Bayesian optimization is to find the global optimum, the score needs to strike a balance between exploring the search space and exploiting regions where the objective function gives promising results. This dilemma is a famous problem in reinforcement learning in machine learning, and is often referred to as the exploration vs. exploitation trade-off (Pack Kaelbling et al., 1996). There are a number of choices for the surrogate model and the acquisition function. Here we will cover two of the most common ones, namely the surrogate model by *Gaussian process regression* and *expected improvement* (EI) acquisition function.

In Bayesian optimization, the surrogate is developed using methods from Bayesian statistics, and the most common choice is to use Gaussian process regression (Frazier, 2018). In short, the goal of Gaussian process regression is to provide a posterior distribution of $\mathcal{L}(\cdot)$, which provides a measure for the mean and uncertainty of $\mathcal{L}(\cdot)$ in the whole hyperparameter space, given the objective evaluations calculated at that point in time. When a new objective evaluation is made, the posterior distribution can be updated with this evaluation, increasing the "knowledge" of the posterior distribution, similar to the process described in equation (9). Suppose we have evaluated $\mathcal{L}(\cdot)$ at the points $\theta_1, ..., \theta_k \in \mathbb{R}^d$ giving us the function evaluations $\boldsymbol{\mathcal{L}}^{(k)} := (\mathcal{L}_1, ..., \mathcal{L}_k)^T$, where $\mathcal{L}_i := \mathcal{L}(\boldsymbol{\theta}_i)$ and $\boldsymbol{\mathcal{L}}^{(k)}$ is the vector of k objective evaluations. We then have the prior distribution

$$\boldsymbol{\mathcal{L}}^{(k)} \sim N_k(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \ , \tag{18}$$

where $\boldsymbol{\mu}_k \in \mathbb{R}^k$ is the mean and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{k \times k}$ is the covariance matrix, often called the *kernel* in Bayesian optimization. The mean vector and kernel are calculated by some chosen mean and kernel functions, denoted $\mu(\boldsymbol{\theta})$ and $\Sigma(\boldsymbol{\theta})$ respectively. Popular choices of mean and kernel functions are outside the scope of this thesis, for a more detailed explanation we refer to Frazier (2018). However, it is worth mentioning that the mean function approximates the expected value of $\mathcal{L}(\boldsymbol{\theta})$ and that the kernel function has the property that two points, $\boldsymbol{\theta}_i$, $\boldsymbol{\theta}_j$, that are close to each other have a high positive correlation in order to model the belief that points close by tend to have similar function evaluations (Frazier, 2018). The idea behind the posterior distribution is that we can infer the value of some new point $\boldsymbol{\theta}_{k+1}$ using the information from previous points. Assume we have the function evaluations $\boldsymbol{\mathcal{L}}^{(k)} = (\mathcal{L}_1, ..., \mathcal{L}_k)^T$, which we give a prior distribution by equation (18). We can then calculate the posterior distribution $\mathcal{L}(\boldsymbol{\theta}_{k+1})|\mathcal{L}(\boldsymbol{\theta}^{(k)})$ by computing the conditional distribution, which is given as (Bousquet et al., 2003)

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}_{k+1})|\mathcal{L}(\boldsymbol{\theta}^{(k)}) &\sim N(\mu_{cond}, \sigma^2_{cond}), \\ \mu_{cond} &= \Sigma(\boldsymbol{\theta}_{k+1}, \boldsymbol{\theta}^{(k)})\Sigma(\boldsymbol{\theta}^{(k)}, \boldsymbol{\theta}^{(k)})^{-1}\left(\mathcal{L}^{(k)} - \mu(\boldsymbol{\theta}^{(k)})\right) + \mu(\boldsymbol{\theta}_{k+1}), \\ \sigma^2_{cond} &= \Sigma(\boldsymbol{\theta}_{k+1}, \boldsymbol{\theta}_{k+1}) - \Sigma(\boldsymbol{\theta}_{k+1}, \boldsymbol{\theta}^{(k)})\Sigma(\boldsymbol{\theta}^{(k)}, \boldsymbol{\theta}^{(k)})^{-1}\Sigma(\boldsymbol{\theta}^{(k)}, \boldsymbol{\theta}_{k+1}) \ . \end{aligned} \tag{19}$$

Now that we have a way to approximate the mean and uncertainty of $\mathcal{L}(\cdot)$ in the search space for a given number of function evaluations using the posterior distribution, we need a method

of choosing *which* new point $\boldsymbol{\theta}_{k+1}$ to evaluate. In other words, we need an acquisition function. One of the most popular acquisition functions is the expected improvement acquisition function (Frazier, 2018). The expected improvement can be formulated in the following way. Assume the best objective evaluation so far is given by

$$\mathcal{L}^* := \max_i \mathcal{L}_i, \ i = 1, ..., n \ .$$

If the new objective evaluation $\mathcal{L}_{k+1}$ is higher than $\mathcal{L}^*$, we have an *improvement* of $\mathcal{L}_{k+1} - \mathcal{L}^*$, otherwise we have no improvement, and we set the improvement to be 0. We can express the improvement as

$$[\mathcal{L}_{k+1} - \mathcal{L}^*]^+ := \begin{cases} \mathcal{L}_{k+1} - \mathcal{L}^*, & \mathcal{L}_{k+1} > \mathcal{L}^* \\ 0, & \text{otherwise} \end{cases}$$

The tricky part is that we do not know the improvement until after we have made the evaluation $\mathcal{L}_{k+1}$. Expected improvement is a method that tries to maximize the expectation of this improvement, and it can be expressed as

$$E[I_k(\boldsymbol{\theta})] := E_k \left[[\mathcal{L}_{k+1} - \mathcal{L}^*]^+ \right] \ ,$$

where the expectation $E_k[\cdot] = E[\cdot|\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_k, \mathcal{L}_1, ..., \mathcal{L}_k]$ is calculated under the posterior distribution of $\mathcal{L}(\cdot)$ at $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_k$ (Frazier, 2018). For a more detailed explanation we again refer the reader to Frazier (2018). Note that maximizing the expected improvement under the posterior distribution will tend to choose a new point $\boldsymbol{\theta}_{new}$ at points where the posterior mean and posterior uncertainty (*i.e.* variation) is large, which is consistent with respecting the exploration vs. exploitation trade-off. A simple example of the Gaussian process posterior with expected improvement acquisition function is illustrated in Figure 3 for a one-dimensional hyperparameter space.

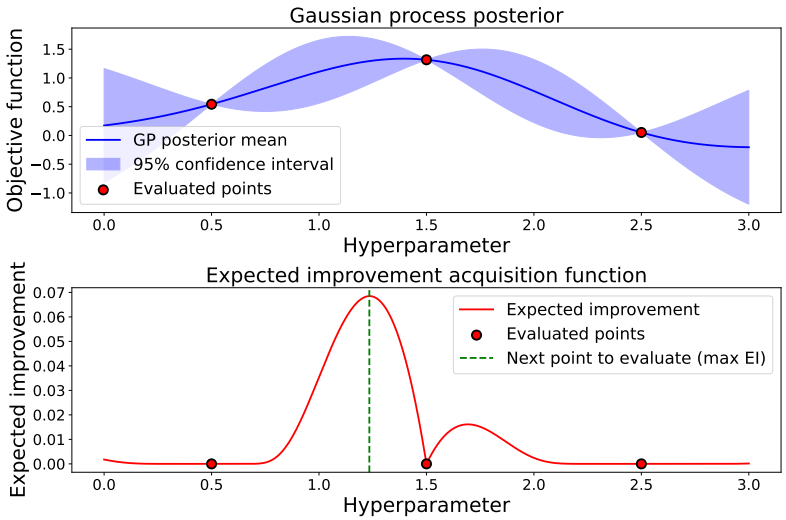


Figure 3: Objective function $\mathcal{L}(\theta) = \exp(-(\theta - 1.5)^2) - 0.5\exp(-(\theta - 2.5)^2)$ with Gaussian process posterior mean and 95% confidence interval (above) and expected improvement acquisition function (below). Evaluated points shown in red. Illustration in 1-dimensional hyperparameter space.

The algorithm for the whole Bayesian optimization process is given in Algorithm 1. The algorithm assumes only one initial randomly uniformly distributed point $\boldsymbol{\theta}_1 \in H$ is chosen, but it is possible to initialize $\mathcal{L}$ with more than one such point. Note that in the algorithm we define $(\mathcal{L}_1, \mathcal{L}^{(0)})^T$ to be equal to $\mathcal{L}_1$ such that the first iteration of the for-loop is well-defined.

2.2.9 Interpretable machine learning

Statistical learning methods can at a very low level be divided into the task of *prediction* and the task of *inference*. Assuming we have the dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{x}_i \in \mathbb{R}^m$

Algorithm 1 Bayesian optimization procedure

Choose the number of trials (evaluations) n to run
Place a Gaussian process prior on $\mathcal{L}$
Observe $\mathcal{L}$ at some uniformly randomly chosen point $\boldsymbol{\theta}_1 \in H$
Initialize $\mathcal{L}^* = \mathcal{L}(\boldsymbol{\theta}_1)$ and $\boldsymbol{\theta}_1 = \boldsymbol{\theta}^*$
for $i = 2, ..., n-1$:
  Update posterior $\mathcal{L}(\boldsymbol{\theta})|(\mathcal{L}_i, \mathcal{L}^{(i-1)})^T$ using all evaluations
  Decide $\boldsymbol{\theta}_{i+1}$ by maximizing the acquisition function over the posterior $\mathcal{L}(\boldsymbol{\theta})|(\mathcal{L}_i, \mathcal{L}^{(i-1)})^T$
  if $\mathcal{L}(\boldsymbol{\theta}_{i+1}) > \mathcal{L}^*$:
    $\mathcal{L}^* = \mathcal{L}(\boldsymbol{\theta}_{i+1})$
    $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{i+1}$
    end if
  end for
Return the best set of hyperparameters $\boldsymbol{\theta}^*$

denotes the covariates and $\boldsymbol{y} = \{y_i\}_{i=1}^n$, $y_i \in \mathbb{R}$ is the response, the goal of prediction can in short be described as finding a model that predicts the response $\boldsymbol{y}$ while using the covariates $\mathbf{X}$ as input. Moreover, the model should aim to minimize the test error by minimizing the loss function, or in other words, it should balance bias and variance to better predict the response on unseen data by capturing the underlying relationship between the covariates $\mathbf{X}$ and the response $\boldsymbol{y}$. The XGBoost model described in Section 2.2.6 is an example of a powerful prediction model. A prediction model is often described as a "black box" model, in the sense that we are not interested in what is going on "under the hood" of the model, as long as it produces accurate predictions for $\boldsymbol{y}$ (James et al., 2013). However, in many situations, we want more information than just the prediction itself. When the goal is inference, we are more interested in how an algorithm uses the data to arrive at the prediction. For example, we might want to know which covariates are (the most) associated with the response. In the context of genomic studies, genomic prediction aims at estimating breeding values based on genetic markers (SNPs) and other covariates. In the inference setting, one of the tasks is to find which SNPs are most associated with the breeding value (trait).

In machine learning, we often discuss the *interpretability* of a model. We can tentatively describe the interpretability of a model as the degree to which a human can comprehend the decisions of the model, however, there is no real consensus on what interpretability is in machine learning, nor is it clear how to measure it (Molnar, 2022). The terms interpretability and explainability are often used interchangeably. Complex and flexible models such as XGBoost are typically more accurate in predicting non-linear, faint, or rare phenomena, but unfortunately more accuracy often comes at the expense of interpretability (Boehmke and Greenwell, 2019). Interpreting machine learning models is an emerging field that has become known as IML (Boehmke and Greenwell, 2019).

IML encompasses methods that are interpretable as standalone models, such as regression trees, where the decision tree can mimic the way a human makes binary decisions. On the other hand, it also includes methods that can explain complex models such as XGBoost after they have been fitted, which is a process often called post-hoc explanation (Molnar, 2022). These model explanations can be categorized as either global or local. Global methods are methods that provide insights into the average behavior of the model across the entire dataset. Local methods, on the other hand, focus on individual predictions (Molnar, 2022). We can categorize the different methods in IML as either model-agnostic or model-specific. Model-agnostic methods are methods that can be applied to any machine learning model (Boehmke and Greenwell, 2019). Model-specific methods work only for a specific type of model, usually exploiting the specific structure of the model. Model-agnostic methods allow us to utilize the predictive power of the machine learning model and the interpretability component of the model-agnostic method, while also making it easier to compare feature importance across models. Section 2.3.2 covers the SHAP method, which is a model-agnostic method of explaining XGBoost post-hoc that will be implemented on a genomic dataset later in this thesis.

2.3 Methods of inference in genomic studies

2.3.1 Genome-wide association studies

GWAS encompass approaches of finding which SNPs are associated with the trait of interest, and thus encompass methods of inference in genomic studies (Uffelmann et al., 2021). It is the genes that encode for polypeptides that make up proteins and which eventually can affect a phenotype. As discussed in Section 2.1.3, SNPs are nevertheless effective as genetic markers, as they can be in LD with genes that code for the trait of interest, and as such can be used to map QTL. One of the methods GWAS use to find out if a SNP is significantly associated with a trait is to regress each SNP M_{ij} on the phenotype y_i. Assuming we have $j = 1, ..., m$ different SNPs for $i = 1, ..., n$ individuals, we can express this as

$$y_i = M_{ij}u_j + \epsilon_i, \quad i = 1, ..., n \ , \tag{20}$$

where ϵ_i is the residual term for individual i and u_j the effect of SNP j, M_{ij}, in the matrix of SNPs $\mathbf{M} \in \mathbb{R}^{n \times m}$. Hence, we fit a univariate regression model for each SNP. A univariate model is used instead of the full model since the standard genomic dataset typically has a higher number of covariates m than sample size n, $m > n$, leading to an underdetermined system. We can now test the significance of each SNP by a standard hypothesis test $H_0 : \ u_j = 0$ against alternative hypothesis $H_1 : u_j \neq 0$. This test can be done by known methods such as the Wald test or the likelihood test. Doing this for each SNP, we get m different p-values. These p-values are typically visualized by a so-called Manhattan plot, where the SNPs are ordered in their chromosome and bp position along the x-axis while the y-axis shows the $-\log_{10}(p)$ values. Regions of interest can then be spotted from outliers in p-values.

When doing GWAS-analysis on wild populations, chances are that some of the individuals in the sample are related. Relatedness naturally implies that those individuals will share more genetic material than on average otherwise, and they may also share environmental effects. This creates a correlation between related individuals in genetic structure or otherwise, such that the assumption of independence of observations in linear regression is broken. An alternative to the standard GWAS method given in equation (20) is to instead fit a univariate LMM where we include a random intercept on the identity of individuals. The point is then that the random intercept should take care of the correlation between individuals due to relatedness. The random intercept is typically assumed to follow a normal distribution with zero mean, and where the covariance is modeled by a relatedness matrix $\mathbf{G}$ similarly to the pedigree-based or genomic animal model.

One potential issue with GWAS is that the results can be context-dependent. As already discussed, relatedness can be one source of false positive/negative results in GWAS. The methods described also do not factor in epistatic effects, that is, gene-by-gene interaction effects, which also can influence the results of GWAS (McKinney and Pajewski, 2012). Similarly, effects from gene-by-sex interactions (Kyrgiafini et al., 2023) and gene-by-environment interactions (Murcray et al., 2009) are not taken into account. Another problem in GWAS, that has become known as the problem of "missing heritability", is the observation that GWAS-hits often explain only a small proportion of the heritability of most quantitative traits (Eichler et al., 2010). The missing heritability problem is seen in for example the highly heritable trait human height, and believed to arise because such traits are complex and polygenic and thus affected by many genes with alleles of small effect sizes (Yang et al., 2010; Yengo et al., 2022).

Most of the associations observed in genetic studies are due to a confounding effects, that is, a factor which is associated with both the SNP (or other genetic marker) and the trait of interest (Aulchenko, 2011). If we would have controlled for this factor, for example in a LMM, the association between the SNP and the trait would be removed. If the factor in question has a causative relation to the trait of interest, this is an association we would want to pick up on in GWAS, and as such it is a "good" association. However, another type of confounding is confounding by population structure (Aulchenko, 2011), which can be a concern in GWAS when dealing with several slightly genetically different populations, as it can lead to false positives (Berg et al., 2019). This effect is relevant to this thesis as we will be working with data from the meta-population of house sparrow in the Helgeland island system, where differentiation due to dispersal and genetic drift has been demonstrated (Jensen, Moe et al., 2013; Saatoglu et al., 2021).

Another potential issue with GWAS is its use of p-values. Let us first formally define what a (one-sided) p-value is.

Definition 2.1 (p-value)**.** The p-value is the probability of obtaining the observed statistic, or more extreme results, assuming the null-hypothesis is true.

$$p = P(X \geq x_{obs}|H_0) \ .$$

The general use of p-values has become a highly debated topic in the many scientific communities, particularly the standard method of using null-hypothesis significance testing with an arbitrary p-value cutoff (Goodman, 2008; Head et al., 2015; Ioannidis, 2018; Muff, Nilsen et al., 2022). One of the issues discussed is that the interpretation of p-values is confusing for many, and they can often be misinterpreted, as pointed out by Goodman (2008). Some are also worried about the potential to (mis)use p-values to get misleading results in form of "p-hacking", which is a type of bias that occurs when researchers collect or select data or statistical analyses until nonsignificant results become significant (Head et al., 2015). An issue that is directly related to genomic studies is the fact that the value of a p-value stem from both the magnitude of the effect size and the uncertainty of the effect size. Figure 4 demonstrates two different scenarios where the p-value is approximately equal to 0.01, but have completely different effect sizes. The pattern observed in Figure 4 brings our attention back to the issue of missing heritability. Complex polygenic traits require huge sample sizes to reduce the uncertainty to a level were significance is achieved, as was shown for example by the analysis of human height (Yang et al., 2010; Yengo et al., 2022). Since p-values do not provide an estimate for the effect size, it also makes it more difficult to compare results across different experiments using p-values.

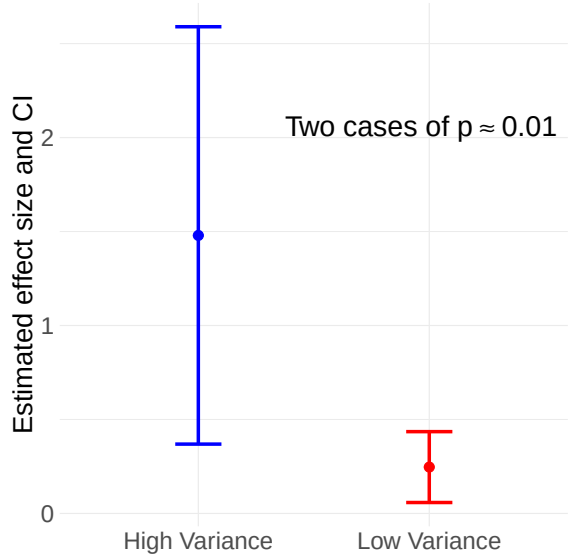


Figure 4: Simulated example of two hypothesis tests with $H_0 : u_i = 0$ for two linear models $\boldsymbol{y}_k \sim \boldsymbol{x}\boldsymbol{u}_k + \boldsymbol{\epsilon}_k$, $k = 1, 2$. Both significance tests result in p-value ≈ 0.01. Linear models simulated with sample sizes $n_{1,2} = 24, 130$, effects $u_{1,2} = 1.5, 0.2$ and variables $\epsilon_1,\ \epsilon_2,\ x \sim N(0, 3^2),\ N(0, 1^2),\ N(0, 1^2)$.

2.3.2 Variable importance based on SHAP values

The interpretable machine learning method SHAP (Molnar, 2022), which was developed from the concept of *Shapley values* from coalition game theory (Shapley, 1952), has shown promise in studies aiming for inference on importance of covariates in machine learning (*e.g.*, Ekanayake et al., 2022). While it is not yet a standard approach in genomic studies, SHAP may offer an alternative strategy to GWAS for identifying SNPs associated with a trait of interest. To better understand the SHAP framework, we begin by introducing the theoretical foundation of Shapley values. The

Shapley value is a method for assigning payouts to *players* depending on their contribution to the total value of a *game* (Molnar, 2022). To explain Shapley values, we first have to introduce some terminology and define what a game is. For a set of players S in the set of all combinations of players U, $S \subset U$, we define the value function $\nu : S \mapsto \mathbb{R}$ as the value function of a game if it has the following properties

$$\begin{aligned} &\nu(\emptyset) = 0 \ , \\ &\nu(S) \geq \nu(S \cap T) + \nu(S - T) \quad \forall S, T \subset U \ . \end{aligned}$$

Further, we define the carrier of ν as any set $N \subset U$ such that $\nu(S) = \nu(N \cup S)\ \forall S \subset U$. Carriers are useful as the players outside any carrier do not make a contribution to the payout through the value function. Now we define $\Pi(U)$ as the set of permutations of U. Then, for $\pi \in \Pi(U)$, πS is the image of S under π. Further, we define the function $\pi\nu$ by $\pi\nu(\pi S) = \nu(S)\ \forall S \subseteq U$. With the groundwork done, we can define the Shapley value $\phi_i(\nu)$ of a game ν as

$$\phi_i(\nu) = \sum_{S \subseteq N, i \in S} \frac{(|S|-1)!(|N|-|S|)!}{|N|!}\nu(S) - \sum_{S \subseteq N, i \notin S} \frac{|S|!(|N|-|S|-1)!}{|N|!}\nu(S), \quad \forall i \in N \ .$$

Another formulation for the Shapley values is

$$\phi_i(\nu) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} \left(\nu(S \cup \{i\}) - \nu(S)\right), \quad \forall i \in N \ . \tag{21}$$

Here, $|S|!$ represents the number of permutations of the players in S. The number of permutations for the remaining players not in S is given by $(|N|-|S|-1)!$. The denominator $|N|!$ represents the total number of permutations for all players, and scaling by this constant normalizes the weights so that they sum to one over all subsets in the sum. The weights scale the marginal contribution of player i, $\nu(S \cup \{i\}) - \nu(S)$, by the fraction of permutations where S appears in a random ordering of players. Hence, the Shapley value averages the marginal contribution of player i across all possible subsets of players S.

It can be shown that the Shapley value is the unique solution to three properties (Shapley, 1952). The first property is the symmetry property, which states that for each π in $\Pi(U)$, $\phi_{\pi i}(\pi\nu) = \phi_i(\nu)\ \forall i \in U$. The second property is that of efficiency, which states that for each carrier N of ν, $\sum_N \phi_i(\nu) = \nu(N)$. The final property is the property of linearity, such that for any two games ν and w, we have that $\phi(\nu + w) = \phi(v) + \phi(w)$. The intuitive understanding of the three properties and the Shapley value will become more clear as we relate it to predictions in machine learning.

We connect these concepts from game theory to machine learning by treating the prediction $f(\boldsymbol{x}_i)$ as the "game" and the covariates $\boldsymbol{x}_i \in \mathbb{R}^m$ as the "players" in the game. Shapley values then allow us to fairly attribute the contributions of each feature to the prediction. Assume we have the training set $\mathcal{D}_{train} = \{y_i, \boldsymbol{x}_i\}_{i=1}^{n_{train}}$ and let $f(\boldsymbol{x}_i^*)$, $\boldsymbol{x}_i^* \notin \mathcal{D}_{train}$ be the model trying to predict $y_i^* \notin \mathcal{D}_{train}$. We can then decompose the prediction as

$$f(\boldsymbol{x}_i^*) = \phi_0 + \sum_{j=1}^{m} \phi_{ij} \ ,$$

where $\phi_0 = E[f(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{D}_{train}]$ is the global mean prediction and ϕ_{ij} is the SHAP value for covariate j and observation $\boldsymbol{x}_i^*$ (Aas et al., 2021). Hence, the SHAP values sum to the deviation of prediction $f(\boldsymbol{x}_i^*)$ from the global mean prediction. The SHAP values still satisfy the three properties of Shapley values, and through the lens of SHAP values, the three properties can be stated as

- Efficiency: The sum of SHAP values equals the total game value:
  $\sum_{j=1}^{m} \phi_{ij} = f(\boldsymbol{x}_i^*) - \phi_0 = \nu(U)$, where U is the set of all covariates.
- Symmetry: If two covariates contribute equally to all subsets S, they receive the same SHAP value:
  $\nu(S \cup \{m\}) = \nu(S \cup \{t\}) \quad \forall S \subseteq U \setminus \{m, t\} \ \Rightarrow \ \phi_m(\nu) = \phi_t(\nu) \ .$

- Linearity: For two combined models ν and w, SHAP values add linearly: $\phi_j(\nu + w) = \phi_j(\nu) + \phi_j(w)$.

We can also state a fourth property from the property of ν having carriers. We define the dummy property as the property that covariates that do not influence the prediction, receive a SHAP value of zero:

$$\nu(S \cup \{m\}) = \nu(S) \quad \forall S \subseteq U \backslash \{m\} \;\Rightarrow\; \phi_m(\nu) = 0 \; .$$

It is worth stressing that SHAP values are unique in satisfying these properties, and it is because of these properties that we can call the SHAP values "fair".

To compute the SHAP values in the context of machine learning models, we need to define the contribution of a subset of covariates $\nu(S)$, $S \subseteq \{1, ..., m\}$. In SHAP, $\nu(S)$ often represents the expected model output conditional on the values of S (Aas et al., 2021). This allows us to connect the game theory function to feature importance in machine learning. The value function is then given as

$$\nu(S) = E[f(\boldsymbol{x}|\boldsymbol{x}_S = \boldsymbol{x}_S^{(i)})] \; .$$

Here, $\boldsymbol{x}_S^{(i)}$ denotes the covariates in S for observation i.

In summary, SHAP values explain individual predictions of machine learning models such as XGBoost. Thus, SHAP values are local explanation methods. The SHAP value of a specific covariate for a specific prediction gives a measure for the importance of this feature to the prediction relative to the other covariates. The SHAP values also need to be interpreted relative to the global mean prediction, since all the SHAP values for a specific prediction, sum to the deviation of this prediction from the global mean prediction. To better understand the intuition behind SHAP values, it is worth noting that in the case of a linear regression model, SHAP values are equivalent to the LMG measures of relative variable importance (Lindeman et al., 1980; Coleman, 2017). The LMG method averages the additional contribution of each covariate to the model's R-squared over all possible orderings of covariates (Grömping, 2006). While the LMG method is applicable only to linear models, the SHAP values can in theory be computed for any predictive model, including non-linear models.

Calculation of a single Shapley value given by equation (21) requires us to evaluate all possible combinations of subsets of covariates from the set $\{1, ..., m\}$, meaning a total of 2^m possible subsets. The large number of evaluations makes the calculation infeasible in practice for large m, and in particular for the large genomic datasets used in this thesis. In this thesis we will use SHAP values to explain predictions from the XGBoost model. For tree-based methods, the *Tree Shapley additive explanations* (TreeSHAP, Lundberg et al., 2018) is a good alternative way to compute the Shapley values. TreeSHAP is a high-speed model-specific method of estimating Shapley values of tree ensembles that reduces the time complexity from $O(TJ_{max}2^m)$ to $(TJ_{max}D^2)$. Here, T denotes the number of trees, J_{max} denotes the maximum number of leaves in a tree, D is the maxium depth of a tree and m is the number of covariates in the tree model. The TreeSHAP method provides the same exact Shapley values as SHAP does. It has a faster computation time since it exploits the structure of the trees to calculate the SHAP values without evaluating every subset explicitly. In short, TreeSHAP calculates the marginal contributions of each covariate based on how the covariate values affect the model's output when the covariate is included in different paths or branches of the tree. TreeSHAP ensures that the Shapley values remain consistent with the original SHAP method and thus still maintain the same fairness properties. The exact algorithm is beyond the scope of this thesis, we refer to Lundberg et al. (2018).

3 Methods

3.1 Data description

In this thesis we will be working with morphological and genotypic data from a long-term study of an insular house sparrow off the island system off Helgeland coast in northern Norway (Jensen, Steinsland et al., 2008; Pärn et al., 2011; Muff, Niskanen et al., 2019). The study on the house sparrows has been running continuously since 1993. The island system consists of 18 different islands, and the different islands differ in environmental conditions, habitat type and population sizes, such that each island is a population in itself. However, since there is migration between the islands, we can view the whole island system as a meta-population. For the data collected on the house sparrows, we use the naming convention of *inner* for the main islands closest to the mainland, *outer* for the main islands further away from the mainland and *other* for other smaller islands (Muff, Niskanen et al., 2019). The genomic data we will be working with consists of two datasets, which will be referred to as the 180k dataset and the 70k dataset. The 180k dataset contains 182 848 sequenced SNPs and 4625 recordings from 1984 unique individuals. The 70k dataset contains 65 247 sequenced SNPs and 24 951 recordings from 12 560 unique individuals. The 180k dataset is thus characterized by having a lot of genomic material, but at the cost of fewer individuals. The 70k dataset has a higher sample size, but contains less genomic material. The morphological data available contains recorded phenotypes for many different traits. In this thesis we will be focusing on the traits *body mass*, the length of the *tarsus* bone and *wing length*. Several other house sparrow variables are recorded and used in this thesis, and can be seen listed in Table 1. The non-genetic house sparrow data listed will allow us to control for some of the variation in various conditions on the different individuals, which will be useful for the statistical models used in this thesis. For a detailed explanation of the data recording and SNP-sequencing process, we refer to Jensen, Steinsland et al. (2008). In this thesis, we performed studies on genomic prediction and inference on SNP-trait associations. For the genomic prediction, we exclusively used the 180k dataset, while for the inference studies, the 70k dataset was used.

Variable	**Description**
Sex	Sex of individual (1 male, 2 female)
FGRM	Inbreeding coefficient
Month	Month of capture
Age	Age of individual at capture
Outer	Proportion of genetic material from *outer* islands
Other	Proportion of genetic material from *outer* islands
Hatch island	Island the individual was born
Hatch year	Year the individual was born
Island current	Island individual was captured at
Ringnr	Unique individual ID

Table 1: Description of house sparrow variables used in the thesis.

3.2 Prediction of genetic contribution

3.2.1 Genomic prediction with XGBoost model

Machine learning, together with deep learning, are the front-runners in prediction based problems in data analysis, and various prediction methods have been tested in animal and plant breeding (Gill et al., 2022). However, genomic prediction in general has hardly been utilized in studying quantitative genetics in wild populations (Meuwissen et al., 2001; Ashraf et al., 2021; Hunter et al., 2022). To close this gap, we will in this thesis utilize the XGBoost model on the genomic data from house sparrow datasets introduced in Section 3.1 to perform genomic prediction, where the goal is to predict the genetic contribution on the phenotype for the traits body mass, tarsus length and wing length. In Section 2.2.4 we discussed some of the strengths and weaknesses of the regression

trees, and we recall that regression trees are ideal at capturing non-linear interaction effects. All the three house sparrow traits that we analyze in this thesis are assumed to be complex polygenic traits with some epistatic effects between SNPs. One advantage of the XGBoost model in this scenario is that it will be ideal in capturing these epistatic effects that linear-based approaches, such as the traditional animal model or the classical approach with ridge regression, might not pick up on.

As discussed in Section 2.1.1, the variance of quantitative traits is complex to predict since it involves several effects, some of them non-genetic such as the environmental effects. If we train the XGBoost model with the phenotypic data $\boldsymbol{y} \in \mathbb{R}^n$ as the response label, the noise from these non-genetic effects will reduce the precision of the predictions. Thus, we first perform a pre-processing of the phenotypic data in order to remove as much of the noise as possible. To do this, we make use of some of the variables listed in Table 1 by including them in a LMM with the phenotype as response. The following variables are included in the LMM for both the 180k and the 70k datasets: Sex, Month, Age, Island current, Hatch year and Ringnr. For the 70k dataset we also include the variables Outer and Other. We can describe the LMM with the equation

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{d} + \boldsymbol{\rho} + \boldsymbol{\epsilon} \ ,$$

where $\boldsymbol{\beta}$ is the vector of fixed effects, such as the fixed environmental effects, with corresponding design matrix $\mathbf{X}$ of appropriate dimensions. The vector $\boldsymbol{d}$ contains the random environmental effects, with corresponding design matrix $\mathbf{Z}$ of appropriate dimensions. The vector $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is the residual error containing errors from factors not accounted for in the model. The vector $\boldsymbol{\rho} \in \mathbb{R}^n$ contains the identity effects of the individuals, and is assumed to explain the variance between individuals that is not explained by factors accounted for in the model, such as permanent environmental effects and the genetic effect. In other words, $\boldsymbol{\rho}$ captures the variance due to the genetic component of the phenotype, V_G, which is the target of prediction for the XGBoost model. The processed response we will use in training of the XGBoost model is therefore the estimate of the identity effect, given as

$$\boldsymbol{y}^{(p)} = \hat{\boldsymbol{\rho}} \ .$$

Hence, we use the pre-processed phenotype $\boldsymbol{y}^{(p)} \in \mathbb{R}^n$ as response and the SNP data $\mathbf{M} \in \mathbb{R}^{n \times m}$ as covariates in the XGBoost model.

For the implementation of the XGBoost model, we need to choose which loss function $L(\cdot)$ to use in the objective function given in equation (14). We have several options in the choice of loss functions, all with different strengths and weaknesses. In this thesis we will be using the *mean absolute error* (MAE) loss function, defined as

$$\text{MAE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) := \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \ .$$

The mean absolute error measures the magnitude of errors between predicted values $\hat{\boldsymbol{y}}$ and response $\boldsymbol{y}$ with a sample size n. Using the MAE as a loss function, the sum of absolute deviations are minimized, thus MAE measures the conditional median. Another popular loss function is the mean squared error, defined as

$$\text{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \ .$$

The main strength of the MAE, as opposed to MSE, is that MAE is less sensitive to outliers since it estimates the conditional median, as opposed to the conditional mean in MSE. The observant reader will notice that the MAE is not a differentiable function, and that from the analysis of XGBoost in Section 2.2.6 we stated that the objective function in equation (14) requires a differentiable loss function. In this thesis we will use the Python package *xgboost* v2.1.1, which contains the option of using the MAE as a loss function (*XGBoost Parameter* 2022). Using MAE will be slower than using differentiable loss functions since non-derivative line search methods are used instead (*Configure XGBoost* 2024).

To improve the accuracy of the XGBoost model in the high-dimensional genomic setting, applying feature selection prior to model training is worth considering. The full SNP datasets, particularly the 180k dataset, contain far more SNP-covariates than samples, which introduces challenges

typical of high-dimensional data. In the high-dimensional setting, we increase the risk of overfitting and reduce the ability of the prediction model to generalize. Importantly, reducing the covariate space improves computational efficiency. This is particularly relevant in the Bayesian hyperparameter optimization procedure used for XGBoost, which will be derived later in this section. Training XGBoost on fewer SNPs reduces the time per model evaluation, allowing us to perform more objective evaluations, which ultimately leads to a more thorough exploration of the (reduced) hyperparameter space during tuning. A more thorough tuning of a subset of SNPs is expected to lead to a better fit on the chosen SNPs and may ultimately lead to a more accurate prediction model. In addition, we expect many SNPs to be correlated due to the LD-structure. As a result, a well-chosen subset, particularly one that includes representative SNPs from each region of LD, can explain a proportion of phenotypic variation comparable to that of the full SNP dataset.

In this thesis, we will apply two subset selection methods on the 180k dataset and perform genomic prediction using XGBoost with the resulting subsets of SNPs as covariates. In the first subset method, we simply randomly choose 50 000 SNPs from the 180k dataset. The resulting XGBoost model using the 50 000 random SNPs as covariates will be referred to as the XGB_50k-model. In the second subset method, we choose the 10 000 SNPs most correlated with the phenotypic response in the training set as covariates. The details on how we partition into training and testing sets and exactly how the 10 000 SNPs are selected will be explained in detail in Section 3.2.4. The resulting XGBoost model using the subset of 10 000 SNPs will be referred to as the XGB_10k-model. In addition to the two subset approaches, we will perform genomic prediction with an XGBoost model using all the SNPs available in the 180k dataset, which will be referred to as the XGB-model.

For the hyperparameter tuning of the XGBoost models, we will apply the Bayesian optimization method using Python package `Optuna` (Akiba et al., 2019). The `Optuna` package utilizes the *tree-structured Parzen estimator* (TPE, Bergstra et al., 2011). The TPE algorithm is a parameter-sampling algorithm. In short, the TPE algorithm does not model the posterior distribution of the objective function. Instead, it treats the hyperparameters $\boldsymbol{\theta}$ as random variables, and splits the hyperparameter space into two regions based on the probability density of the hyperparameters. One region contains hyperparameters for which objective function is below a certain threshold $l(\boldsymbol{\theta})$, while the other contains the hyperparameters for which the objective function is above the threshold $t(\boldsymbol{\theta})$. Thus $l(\boldsymbol{\theta})$ contains the "good" hyperparameters and $t(\boldsymbol{\theta})$ contains the "bad" hyperparameters, since the goal is to minimize the objective function. $l(\boldsymbol{\theta})$ and $t(\boldsymbol{\theta})$ are thus probability densities of hyperparameters $\boldsymbol{\theta}$ conditioned on good and bad evaluations respectively. The acquisition function in the TPE algorithm is proportional to the ratio (Watanabe, 2023)

$$\frac{l(\boldsymbol{\theta})}{t(\boldsymbol{\theta})},$$

which effectively guides the search toward regions of the hyperparameter space where good results are more likely while balancing exploration and exploitation. The specific implementation details are out of the scope of this thesis, we refer to Bergstra et al. (2011). TPE requires a predefined search space in which it will search for the optimal hyperparameters. The hyperparameter space applied to all XGBoost models used in this thesis is given in Table 2. Note the choice of fixing n_estimators at 600. The choice of fixing n_estimators is made since this hyperparameter is very correlated with the hyperparameter learning_rate, as per the analysis in Section 2.2.7. Fixing n_estimators will allow the model to exploit and explore the learning_rate more.

Hyperparameter	**Search space**
n_estimators	600
learning_rate	$[\ln 10^{-3},\ \ln 0.1]$
max_depth	$[4,\ 14]$
Subsample	$[0.05,\ 1.0]$
Colsample_bytree	$[0.05,\ 1.0]$
min_child_weight	$[2,\ 25]$

Table 2: Search space used in `Optuna` hyperparameter tuning for all XGBoost models.

Each XGBoost model in this thesis uses the `Optuna` package for hyperparameter optimization. The

number of trials (*i.e.* total evaluations of the objective function) in the hyperparameter tuning is constrained by time-complexity considerations, and time per model evaluation increases with the number of SNP-covariates included in the XGBoost model. Here, we apply 20, 40 and 80 trials to the XGB, XGB_50k and XGB_10k models respectively. In summary, while the XGB model benefits from a richer set of SNP covariates and thus more genomic information available, its limited number of tuning trials may hinder its ability to find optimal hyperparameters. The XGB_10k model, with fewer SNP-covariates but more optimization trials, is more likely to identify a well-tuned set of hyperparameters, whereas the XGB_50k model offers a balance between genomic information available and tuning depth.

3.2.2 Genomic prediction with the Bayesian animal model

The animal model is one of the traditional ways to perform genomic prediction, and here we implemented a state-of-the-art genomic-based Bayesian animal model, which will act as a benchmark model for the other prediction models to be compared to. For the XGBoost model, we performed a pre-processing of the phenotype in order to remove noise from factors such as environmental effects. The animal model, which is a LMM model, has the breeding values $\boldsymbol{a} \in \mathbb{R}^n$ explicitly included as a random effect, such that any pre-processing on the phenotype is not necessary. We will in this thesis fit an animal model on the form

$$\boldsymbol{y} = \mu \mathbf{1}_n + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{a} + \mathbf{D}\boldsymbol{r} + \boldsymbol{\epsilon} \ , \tag{22}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ denotes the phenotypic response, and where we assume the breeding values $\boldsymbol{a}$ follows the Gaussian distribution, $\boldsymbol{a} \sim N(\mathbf{0}, \mathbf{G}V_A)$, with $\mathbf{G}$ being the genomic relationship matrix derived by method 1 as described in Section 2.1.4. μ denotes the population mean while $\boldsymbol{\beta}$ and $\boldsymbol{r} \sim N(\mathbf{0}, \mathbf{V}_r)$ denote the fixed and random effects respectively, with their respective design matrices $\mathbf{X}$ and $\mathbf{D}$ of appropriate dimensions. Finally, $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \boldsymbol{I}_n\sigma_\epsilon^2) \in \mathbb{R}^n$ represents the independent residual errors. For the prediction of the house sparrow phenotypes, the animal model will be fitted using variables Sex, FGRM, Month, Age, Outer, Other, Hatch island, Hatch year and Ringnr from the house sparrow variables in Table 1.

A Bayesian approach can be used to estimate the posterior probability distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$ from a LMM, where $\boldsymbol{\theta}$ denotes the hyperparameters of the LMM. In this approach, the posterior probability distribution obtained can provide estimates for statistics such as the median of the posterior distribution. The LMM describing the animal model in equation (22) can be expressed as a *latent Gaussian model* (LGM, Martino and Riebler, 2019). For LGMs, there exists a specific Bayesian method called INLA which provides approximations for the posterior distribution (Rue et al., 2009; Martino and Riebler, 2019). In this thesis, we will use INLA to fit a *Bayesian animal model*, giving us approximations for the posterior distribution $p(\boldsymbol{a}|\boldsymbol{y})$. Taking the median of this distribution gives us an estimate for the breeding values called genomic estimated breeding values (GEBV). We want to show that the genomic animal model can be expressed as a LGM, which in general from is written as (Martino and Riebler, 2019)

$$\begin{aligned} \boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_i p(y_i|\delta_i, \boldsymbol{\theta}) \ , \\ \boldsymbol{x}|\boldsymbol{\theta} &\sim N(\mathbf{0}, \mathbf{Q}^{-1}(\boldsymbol{\theta})) \ , \\ \boldsymbol{\theta} &\sim p(\boldsymbol{\theta}) \ . \end{aligned}$$

Here, $\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}$ represents the likelihood, where $\boldsymbol{x}$ is the joint distribution of all parameters in the linear predictor $\boldsymbol{\delta}$. $\boldsymbol{y}$ is the (observed) response and $\boldsymbol{\theta}$ are the hyperparameters. Further, $\boldsymbol{x}|\boldsymbol{\theta}$ represents the latent field with precision matrix (*i.e.* inverse covariance matrix) $\mathbf{Q}(\boldsymbol{\theta})$, and $p(\boldsymbol{\theta})$ represents the hyperprior probability distribution. By putting a Gaussian prior on the fixed effects $\boldsymbol{\beta}$, the genomic animal model in equation (22) can be expressed as a LGM with variables $\boldsymbol{x} = (\boldsymbol{\beta}, \boldsymbol{a}, \boldsymbol{r})^T$, $\boldsymbol{\theta} = (V_A, \sigma_\epsilon^2, \mathbf{V}_r)$ and linear predictor $\boldsymbol{\delta} = \mu\mathbf{1}_n + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{a} + \mathbf{D}\boldsymbol{r}$. Thus, we can compute the GEBV using INLA by deriving the approximate sample distribution $\tilde{p}(\boldsymbol{a}|\boldsymbol{y})$ and taking the median. The exact INLA procedure is out of the scope of this project, we refer to Rue et al. (2009). In this thesis, we trained the Bayesian animal model using package `R-INLA` (Rue et al., 2009).

3.2.3 Genomic prediction with ridge regression

Ridge regression was introduced in genomic prediction by Whittaker et al. (2000) as an alternative to the procedure of using subset selection coupled with the OLS estimator. Ridge regression emerged as a viable alternative in genomic prediction as it is well-suited for handling the high dimensionality and LD-induced correlation structure in SNP-covariates that is characteristic of genomic datasets. Bayesian regression approaches such as BayesA and BayesB (Meuwissen et al., 2001), along with subsequent extensions like BayesR and other models from the "Bayesian alphabet" (Gianola, 2013), have been widely applied in genomic prediction due to their capacity to flexibly model marker-specific variances and account for a range of effect sizes, including the presence of markers with large or negligible effects. However, ridge regression provides a computationally cheaper alternative, since it avoids the added complexity and hyperparameter tuning required for the Bayesian counterparts. The shrinkage effect due to the regularization in ridge regression makes the ridge model ideal when a substantial proportion of SNPs is expected to contribute to the phenotypic variation, and where each effect is small and additive (De Vlaming and Groenen, 2015), and thus, we expect ridge regression to perform particularly well for the prediction of highly polygenic traits. Regularization is especially important for genomic datasets where the number of markers m often far exceeds the number of samples n, the so-called $m \gg n$ problem (Van De Geer and Van Houwelingen, 2004). In such settings, overfitting becomes a significant risk because the model has too much flexibility relative to the amount of data. By penalizing large coefficient estimates, ridge regression constrains this flexibility and improves the robustness of predictions.

In this thesis, ridge regression models will serve as baseline methods in genomic prediction. Ridge regression is a simple and computationally cheap method relative to the other prediction methods used in this thesis, and thus it provides a useful reference point against which the accuracy of the more complex approaches, namely the Bayesian animal model and XGBoost, can be compared. Ridge regression is simpler in the sense that it involves only a single regularization parameter (λ_R), in contrast to the many hyperparameters used in XGBoost, and ridge regression does not require the computationally intensive calculation of the inverse of the genomic relationship matrix $\mathbf{G}$ that characterizes the Bayesian animal model. From both a modeling and computational perspective, ridge regression is less complex than both the Bayesian animal model and XGBoost, and the more sophisticated methods must demonstrate improved predictive accuracy in order to justify their additional complexity.

As discussed already for genomic prediction with the XGBoost model, when the aim is predicting the genetic contribution to the phenotype of an individual, we also have to account for the non-genetic effects on the phenotype to obtain valid results. In order to account for non-genetic effects on the phenotypic response in the ridge regression model, we use the same approach as with the XGBoost model, namely to perform pre-processing on the phenotype using a LMM. In this thesis, we used the same pre-processed phenotype $\boldsymbol{y}^{(p)}$ for genomic prediction with ridge regression as we used for the XGBoost models on the 180k dataset, as described in Section 3.2.1.

We recall from Section 2.2.2 that the ridge regression model generates predictions according to the linear model given in equation (5), where coefficients of estimate $\hat{\boldsymbol{\beta}}_R$ are computed according to equation (8). From the marker-based regression perspective in equation (3), the centered SNP matrix $\mathbf{M}_c$ takes the place of the matrix of covariates $\boldsymbol{X}$, and the ridge-coefficients $\hat{\boldsymbol{u}}_R$ are estimates of the allele substitution effects of SNPs, $\boldsymbol{u}$. In other words, the ridge regression model can be expressed as

$$f(\mathbf{M}_c; \lambda_R) = \hat{\boldsymbol{y}}^{(p)} = \mathbf{M}_c \hat{\boldsymbol{u}}_R + \boldsymbol{\epsilon} \ , \tag{23}$$

where the ridge-estimated coefficients can be expressed as

$$\hat{\boldsymbol{u}}_R = (\mathbf{M}_c^T \mathbf{M}_c + \lambda_R \mathbf{I}_m)^{-1} \mathbf{M}_c^T \boldsymbol{y}^{(p)} \ . \tag{24}$$

In this thesis, we perform genomic prediction with ridge regression using the package `R-glmnet` (Friedman et al., 2010). The ridge regression model contains a single hyperparameter, namely the ℓ_2-regularization parameter λ_R, which we need to optimize. Hyperparameter optimization is performed with the `R-glmnet` λ-optimization function `R-cv.glmnet()`, which utilizes a *cross-*

validation (CV) strategy for finding the best hyperparameters. The CV approach for hyperparameter tuning will be described in detail in Section 3.2.4. Typically, hyperparameter optimization algorithms need a predefined search space for the hyperparameters, in which the algorithm searches for the optimal hyperparameter(s), seen for example for the `Optuna` package used in the XGBoost hyperparameter tuning. The `R-cv.glmnet()` function does not require a user-defined search space, as it computes the search space for λ_R according to Algorithm 2. By default, `R-cv.glmnet()` tests for $K = 100$ values in Algorithm 2, but it uses a stopping criterion that might stop the sequence early. Note that the algorithm is implemented for a general *elastic net* model, which ridge and lasso regression are special cases of (Friedman et al., 2010). In the second step of Algorithm 2, λ_{max} is computed as the smallest λ such that all regression coefficients are zero. For ridge regression, all coefficients will not go to zero unless $\lambda_R \to \infty$. In the implementation of `R-cv.glmnet()`, for the specific case of ridge regression, an elastic net model very close to ridge regression is used as an approximation for the ridge regression model in the tuning of the λ_R hyperparameter. After the tuning of λ_R, we generated ridge regression models using the `R-glmnet()` function. We refer to Friedman et al. (2010) for a detailed review of the `R-glmnet` package.

Algorithm 2 λ optimization procedure for ridge and lasso regression

1. Center the SNP matrix $\mathbf{M}$ to acquire centered matrix $\mathbf{M}_c$
2. Compute λ_{max}, i.e., the smallest λ such that all regression coefficients are zero
3. Define $\lambda_{min} = \kappa\lambda_{max}$, where $\kappa = 10^{-2}$ for $n < m$ and $\kappa = 10^{-4}$ for $m \leq n$
4. Create exponentially decaying λ-sequence:

$$\lambda_k = \exp\left(\log(\lambda_{max}) - \frac{k-1}{K-1}\log(\frac{\lambda_{max}}{\lambda_{min}})\right) \ , \ k = 1, ..., K$$

5. Return discrete search space $\lambda_{seq} = \{\lambda_k\}_{k=1}^{K}$

As genotypic technologies continue to advance and costs decline, the size of genomic datasets used in genomic prediction grows large, even up to full sequence of genomic information available (Ødegård et al., 2018). The increase in available data allows for better statistical modeling, however it also introduces statistical challenges related to computational cost and high-dimensionality, where the number of covariates m far exceed the number of individuals n. One effective strategy to address this issue is dimensionality reduction through PCA. Previous findings show that, for populations with high levels of relatedness and small effective population size, which for example is typical in plant breeding, a relatively small number of PCs are needed to explain the majority of genetic variation (Palaniyappan et al., 2024). Relatively few PCs are needed due to substantial LD in small effective population sizes, where closely related individuals share long genomic segments (Ødegård et al., 2018). Wild populations do not typically have the levels of relatedness observed in breeding populations, however previous studies have demonstrated that PCA-based approaches can be effective also in wild populations (Aspheim et al., 2024).

PCRR is a prediction model that performs PCA on the SNP matrix $\mathbf{M}$, effectively performing dimensionality reduction, and uses the PCs as covariates in a ridge regression model. Using dimensionality reduction techniques such as PCA as a pre-processing step in genomic prediction models may offer a solution to the increasing sizes of genomic datasets. Another advantage of using PCs is that the PCs are orthogonal, which ensures that they are a set of uncorrelated covariates, consistent with the assumption in a lot of linear models. Previous findings have demonstrated that PCRR can yield accurate predictions of breeding values (Ødegård et al., 2018; Zelioli, 2023; Aspheim et al., 2024).

In this thesis, we performed genomic prediction on the 180k dataset using two ridge regression models, namely a PCRR model and a model using SNPs directly as covariates, which will be referred to as the RR model. For the PCRR model, PCA was performed on the SNP matrix $\mathbf{M}$ using the function `R-PCA()` (Lê et al., 2008), while both RR and PCRR utilized the `R-glmnet` package for fitting the model and tuning the λ_R hyperparameter. The PCA generates a total of n PCs (since $m > n$), and we chose to include all n PCs in the PCRR model, such that both the RR- and PCRR-model used all the available genetic information in the 180k dataset. Note that we do not perform the typical standardization of the respective covariates (SNPs in RR, PCs in PCRR) in either of the ridge regression models. For the RR model, standardization will not have

a big effect as all covariates are already on the same scale. For PCRR, standardization will be detrimental to the model accuracy, as the variance of a PC is, by design, equal to the variance it explains in the centered SNP-matrix. As demonstrated by Aspheim et al. (2024), leaving the PCs unstandardized allows ridge regression to automatically shrink the effects of low-variance (and thus, typically less informative) PCs more strongly. Standardizing the PCs removed this implicit regularization, placing all PCs on "equal footing", which reduced prediction accuracy (Aspheim et al., 2024). For the PCRR model, genomic prediction is performed according to

$$f(\tilde{\mathbf{M}}_c; \lambda_R) = \hat{\boldsymbol{y}}^{(p)} = \tilde{\mathbf{M}}_c \tilde{\boldsymbol{u}}_R + \boldsymbol{\epsilon} \ ,$$

where $\tilde{\mathbf{M}}_c \in \mathbb{R}^{n \times n}$ is the (centered) principal component score matrix with column vectors corresponding to the PCs from a PCA of the matrix $\mathbf{M}$ and $\tilde{\boldsymbol{u}}_R \in \mathbf{R}^n$ is the vector of PC coefficients (estimates of genetic effects), generated according to

$$\tilde{\boldsymbol{u}}_R = (\tilde{\mathbf{M}}_c^T \tilde{\mathbf{M}}_c + \lambda_R \mathbf{I}_n)^{-1} \tilde{\mathbf{M}}_c^T \boldsymbol{y}^{(p)} \ .$$

3.2.4 Measuring accuracy of prediction models

So far, we have described the method of predicting the genetic contribution to the phenotype using six different prediction methods, namely the three XGBoost methods XGB, XGB_10k and XGB_50k, the two ridge regression models RR and PCRR, and finally the Bayesian animal model. In order to compare the prediction accuracy of the methods, we need a fair accuracy measure. This section will describe the accuracy measure of computing the *Pearson correlation* between the mean phenotype $\bar{y}_i$ and prediction of genetic component $\hat{y}_i^*$. The mean phenotype is used to account for repeated measurements of individuals. We describe the mean phenotype for individual i as $\bar{y}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} y_{ij}$, where n_i is the number of repeats for individual i and y_{ij} is the observation j for individual i. The genetic component $\hat{\boldsymbol{y}}^*$ that is predicted using the Bayesian animal model is the GEBV, while for the three XBGoost models and the two ridge regression models, $\hat{\boldsymbol{y}}^*$ corresponds to the prediction of the pre-processed phenotype $\hat{\boldsymbol{y}}^{(p)}$ derived in Section 3.2.1. For all six models, we use the accuracy measure

$$\text{Corr}(\hat{\boldsymbol{y}}^*, \bar{\boldsymbol{y}}) \ ,$$

where the Pearson correlation is defined as

$$\text{Corr}(\boldsymbol{x}, \boldsymbol{y}) := \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \ .$$

In order to assess the performance of the models based on the accuracy measured by the Pearson correlations, we utilize a 10-fold CV procedure for all prediction models. In 10-fold CV, we split the whole dataset into training and testing sets ten times, where each instance of splitting is referred to as a CV iteration, and the ten parts are referred to as folds. In each CV iteration, 10% of the data is assigned to the testing set, and the remaining 90% is assigned to the training set. Each individual in the dataset is included in the testing set of exactly one CV iteration, ensuring that the entire dataset is used for testing. In addition, it is customary to shuffle the dataset at the start of the CV-procedure, ensuring that we do not pick up a bias from the ordering of the data. For each iteration, a model is trained on the training set and tested using the testing set, giving us ten different test results. The CV procedure thus allows us to evaluate the accuracy and consistency of the methods by applying them on diverse test sets while utilizing the whole dataset. An illustration of the 10-fold CV procedure is shown in Figure 5. The Bayesian animal model does not require any hyperparameter optimization, and the 10-fold CV procedure can be applied directly the model, producing ten Pearson correlation measures, one from each test set.

For the three XGBoost models and the two ridge regression variants, hyperparameter tuning is required to optimize model performance. In a regular 10-fold CV procedure, ten different models are used for prediction, one for each iteration. Thus, all three XGBoost models and both ridge regression models need to find ten different sets of hyperparameters. When using CV on a model that requires hyperparameter tuning, a procedure called *nested CV* is typically used to find these ten sets. As we recall, the goal in hyperparameter tuning is to find the hyperparameters that

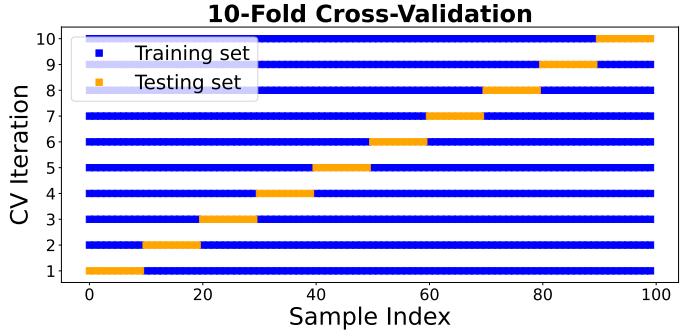


Figure 5: Partitioning of dataset in 10-fold cross-validation technique used on Bayesian animal model for calculating GEBV.

minimize the objective function. The objective function, such as that of XGBoost given in equation (14), contains a loss function $L(\cdot)$ that is the chosen measure of error between the predicted value $\hat{y}$ and true value y. In other words, to tune the hyperparameters, we need to set aside yet another part of the data as a second "testing set", referred to as the *validation set*, which acts as an independent testing set in the validation of hyperparameters.

Nested CV, as the name suggests, nests two CV-loops. To explain the process, we start by examining the process of the first iteration of a regular 10-fold CV. Here, the dataset is split into ten folds, where one fold is held out as the independent test set, and the rest is used as the training set. In nested CV, the next step is to find the best hyperparameter setting in the search space. All hyperparameter settings are validated by a applying a 10-fold CV procedure on the training set, using a loss function to validate in each iteration. The hyperparameter setting with the lowest average loss is chosen as the best set of hyperparameters. Using this best hyperparameter setting, a final model is trained on the training set and tested on the test set. This procedure is repeated for every iteration of the original CV-procedure.

For both ridge regression variants, hyperparameter tuning of λ_R is performed using the `R-cv.glmnet()` function from the `R-glmnet` package. We recall from Section 3.2.3 that `R-cv.glmnet()` automatically generates the search space of λ_R in form of sequence of λ_R-values, denoted λ_{seq}. The `R-cv.glmnet()` package is implemented to perform a version of nested 10-fold CV where each $\lambda_k \in \lambda_{seq}$ is evaluated as a hyperparameter setting, and where MSE is used as loss function. In the nested CV procedure, the predictions from the ten test sets are used to generate ten Pearson correlation accuracy measures.

The nested CV procedure used for the ridge regression models requires many evaluations of the objective function and the training of multiple models. The Bayesian hyperparameter tuning used in XGBoost models is computationally expensive. To address this, we use a simplified "custom CV" procedure for the three XGBoost models, where we omit the inner loop of nested CV for the hyperparameter tuning. We perform a single 10-fold CV procedure where, for each iteration, we divide the dataset into independent training, validation and testing sets in proportions of 80% 10% and 10% respectively. Tuning is only done once in each iteration. For each iteration, we do:

1. Tune hyperparameters with Bayesian hyperparameter optimization using training and validation sets.
2. A model is trained on the union of the training and validation sets, using the hyperparameters obtained in the fist step.
3. Predictions $\hat{\boldsymbol{y}}^*$ of the training samples are generated by the trained model.
4. Pearson correlation between the test set and corresponding predictions are computed.

For the XGB_10k model, in each iteration, the subset of 10 000 SNPs is selected before hyperparameter tuning by choosing the 10 000 SNPs in the training set that are most correlated with the

response $\boldsymbol{y}^{(p)}$. The custom CV approach generates ten accuracy measures for each of the three XGBoost models (XGB, XGB_10k and XGB_50k), with one measure obtained from each test set. An illustration of the custom CV procedure is shown in Figure 6.

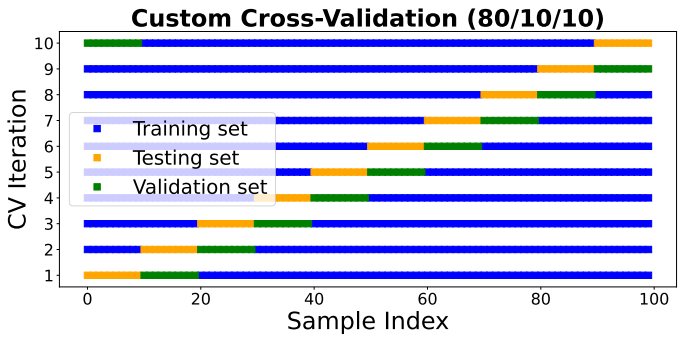


Figure 6: Partitioning of dataset in custom 10-fold CV technique used in the fitting of XGBoost models used in prediction of genetic contribution and calculation of SHAP values.

So far, we have described three different 10-fold CV procedures, one for the Bayesian animal model, one for the two ridge regression variants and a third procedure for the three XGBoost models. In implementation, we ensured that we use the same testing set for all six models, such that we minimize the variance between the models due to differences in testing data. The ten resulting Pearson correlation scores for each model can be visualized using a box-plot, where the median accuracy and variation in accuracy score can be compared and analyzed.

3.3 Methods of inference with GWAS and SHAP

In addition to performing genomic prediction, this thesis explores methods of inference in genomic studies. A method within GWAS called GEMMA will be implemented in order to assess which SNPs are associated with the trait. We apply GEMMA, hereby simply referred to as GWAS, by fitting a univariate LMM, where we include the random intercept on the identity of individuals in order to factor in the correlation between individuals due to relatedness. Assuming we have $i = 1, ..., n$ individuals and $j = 1, ..., m$ SNPs, the univariate LMM for SNP j is given in form

$$\boldsymbol{y} = \mathbf{1}_n \mu + \boldsymbol{M}_j u_j + \boldsymbol{\rho} + \boldsymbol{\epsilon} \ ,$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the vector of phenotypes, μ is the population mean phenotype, $\boldsymbol{\rho} \in \mathbb{R}^n$ is the random intercept on the identity and $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is the independent random vector of residual errors. Moreover, $\boldsymbol{M}_j \in \mathbb{R}^n$ is the vector of SNP genotypes for SNP j and u_j is the effect size of SNP j. All the random effects are assumed to be multivariate normally distributed with zero mean, and $\boldsymbol{\rho}$ is assumed to have covariance matrix given by the genomic relatedness matrix. Using GWAS, we then test the hypothesis $H_0 : u_j = 0$ against $H_1 : u_j \neq 0$ for each SNP in turn, using one of the three test statistics Wald, likelihood ratio or score (Zhou and Stephens, 2012). The corresponding m p-values for each SNP can then be visualized in a Manhattan plot. When doing multiple hypothesis testing, it is common to use a stricter threshold for significance in order to decrease the number of false positives. We will in this thesis use the Bonferroni threshold (α_b), which scales the significance threshold used in a single hypothesis test, typically $\alpha = 0.05$, by the number of tests, in this case m, such that

$$\alpha_b = \frac{\alpha}{m} \ .$$

In order to draw inference from the XGBoost model and explain its predictions, we utilize the Python package `shap` to generate SHAP values for each prediction of the XGBoost model in the 70k dataset. To generate the predictions for every individual in the 70k dataset, we utilize the custom CV procedure. The process of generating SHAP values from the XGBoost predictions can be described the following way. Let $\mathcal{D}^{(k)} = \{\mathcal{D}^{(k)}_{train}, \mathcal{D}^{(k)}_{test}\}$, $k = 1, ..., 10$, be the dataset with

partition number k. The validation set is in this case included in the training set. We further define the training set of iteration k as

$$\mathcal{D}_{train}^{(k)} = \{y_i^{(k)}, \boldsymbol{x}_i^{(k)}\}_{i=1}^{n_{train}^{(k)}} ,$$

and the testing set of iteration k as

$$\mathcal{D}_{test}^{(k)} = \{y_i^{(k),*}, \boldsymbol{x}_i^{(k),*}\}_{i=1}^{n_{test}^{(k)}} .$$

Here, $n_{train}^{(k)}$ and $n_{test}^{(k)}$ denote the training and testing sample sizes of iteration k, respectively. Further, let $m^{(k)}$ denote the number of covariates used in iteration k, such that $\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_i^{(k),*} \in \mathbb{R}^{m^{(k)}}$ $\forall$ i. From these definitions, the process of computing the SHAP values for each XGBoost prediction in the 70k dataset is given in Algorithm 3.

Algorithm 3 SHAP value computation process

For $k = 1, ..., 10$:
  Train XGBoost model f_k on $\mathcal{D}_{train}^{(k)}$
  For $i = 1, ..., n_{test}^{(k)}$:
    $\hat{y}_i^{(k)} = f_k(\boldsymbol{x}_i^{(k),*}) = E[f_k(\mathbf{x}^{(k)}) \mid \mathbf{x}^{(k)} \in \mathcal{D}_{train}^{(k)}] + \sum_{j=1}^{m^{(k)}} \phi_{ij}^{(k)}$
    end for
  end for
Return SHAP values $\phi_{ij}^{(k)},\ k = 1, ..., 10,\ i = 1, ..., n_{test}^{(k)},\ j = 1, ..., m^{(k)}$.

We recall from Section 2.3.2 that the SHAP values generate local explanations, meaning they explain a single prediction. In this thesis we are interested in global explanations, meaning we want insight into the average behavior of the model, or more precisely, the average contribution of the SNPs on the predictions. One way to generate an approximate global model from SHAP values is to take the absolute value of every SHAP value, then for each SNP j, $j = 1, ..., m$, sum the j'th SHAP value from every prediction, and divide them by the total number of predictions $|\mathcal{D}| = n$. Formally, the mean |SHAP| value is defined as

$$\text{mean } |\text{SHAP}|_j := \frac{1}{n} \sum_{k=1}^{10} \sum_{i=1}^{n_{test}^{(k)}} |\phi_{ij}^{(k)}|,\ j = 1, ..., m .$$

Each mean |SHAP| value now gives an approximation for the average contribution of a SNP to the prediction. In other words, it is a measure of feature importance. However, we have to be careful in interpreting the mean |SHAP| values. From Algorithm 3, we see that each SHAP value $\phi_{ij}^{(k)}$ depends on the iteration-specific global prediction mean $E[f_k(\mathbf{x}^{(k)}) \mid \mathbf{x}^{(k)} \in \mathcal{D}_{train}^{(k)}]$ and prediction from iteration-specific XGBoost model f_k. Thus, there is a bias introduced on the mean |SHAP| values from the iteration effects. Because of this, the mean |SHAP| values lose some of the inherent "fairness" from the properties of Shapley values. Nevertheless, the mean |SHAP| value gives a reasonable measure of feature importance.

The mean |SHAP| values can be visualized similarly to the p-values in a Manhattan-style plot, where the SNPs are ordered in their chromosome and bp position along the x-axis like in GWAS, and where the y-axis shows the mean |SHAP| values. A method of visually comparing the two methods is to plot the mean |SHAP| values against the p-values from the GWAS and then fit a regression line between them. This can give us an indication on whether the features that are important for predictions in the XGBoost model are the same ones that generate small p-values for the GWAS. We do not necessarily expect the outliers in p-values and mean |SHAP| to overlap given the different characteristics of the methods. We assume that the mean |SHAP| values, generated from the predictions of the XGBoost model, will favor the SNPs that have interaction effects, such as epistatic effects or dominance effects. On the other hand, the p-values of GWAS are generated by a LMM, thus we expect the linear additive genetic effects to be the driving factor behind (true) outliers in p-values.

3.4 Methods of inference with ridge and lasso

For the classical linear regression model, the OLS estimator has a big advantage in being informative and easy to interpret, if the assumptions of the linear model are met. The OLS estimate for covariate j, $\hat{\beta}_{OLS}^{(j)}$, can be interpreted as the expected change in the response $\boldsymbol{y}$ from one unit change in predictor $\boldsymbol{X}_j$, holding all other variables constant. Here, $\boldsymbol{X}_j$ is the column vector corresponding to predictor j. In other words, the OLS estimator can be interpreted in terms of marginal effects. As discussed in Section 2.2.2, ridge and lasso regression offer a solution to datasets that have high collinearity between predictors by introducing regularization. However, when regularization methods are used, a bias is introduced, and the coefficients no longer have the straightforward marginal interpretation that they do under the OLS estimator.

In practice, ridge tends to distribute influence across correlated predictors, which can make it difficult to identify the importance of individual variables. Lasso regression utilizes the ℓ_1-norm, which tends to produce sparse models. This sparsity is generally advantageous for model interpretability. In practice, for a set of correlated covariates, lasso tends to select a subset of the correlated covariates while shrinking the other covariates in the set towards zero, effectively performing variable selection. In addition, the selection is not based on the estimated effect size of covariates, but are instead chosen rather arbitrarily, such that important covariates may be shrunk toward zero. In summary, the coefficient estimates from ridge and lasso regression may still offer insight into importance of covariates, however the interpretation is not as clear as for the OLS estimator.

Turning our attention back to the genomic setting, using coefficients from ridge or lasso regression may offer an alternative approach for inferring SNP-trait associations. GWAS can be effective in identifying individual SNPs with strong marginal effects, however GWAS may lack the power to detect SNPs with small but meaningful effects, particularly for traits with a polygenic architecture. One advantage coefficient-based ridge and lasso approaches have over GWAS is that all SNPs may be modeled simultaneously, where the joint effect on the phenotype is modeled. In addition, the coefficients may be more suitable in finding SNPs with small but meaningful effects on the phenotype. Coefficients are also more informative than p-values as SNP-importance metrics, as they offer more insight into the effect size of a SNP.

Here, we propose a method of inferring SNP-trait association by estimating the phenotypic variance explained based on coefficients from ridge and lasso regression. First, we assume a linear model of the form

$$\boldsymbol{y} = \mathbf{M}\boldsymbol{u} + \boldsymbol{\epsilon} \ ,$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the response vector of phenotypic values, $\mathbf{M} \in \mathbb{R}^{n \times m}$ is the SNP-matrix, $\boldsymbol{u} \in \mathbb{R}^m$ is the vector of SNP-effects and $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma_\epsilon^2 \mathbf{I}_n) \in \mathbb{R}^n$ is the vector of independent residual errors, for observations $i = 1, ..., n$ and SNPs $j = 1, ..., m$. The core idea is that we can model the *phenotypic variance explained* (PVE), also referred to as the contribution to phenotypic variance, by SNP j, as

$$\text{Var}(\boldsymbol{M}_j u_j) = u_j^2 \text{Var}(\boldsymbol{M}_j) \ , \tag{25}$$

where $\boldsymbol{M}_j$ is the column vector corresponding to SNP number j. The additive component of phenotypic variance can then be approximated by

$$\text{Var}(\boldsymbol{y}) \approx \sum_{j=1}^{m} u_j^2 \text{Var}(\boldsymbol{M}_j) \ ,$$

where we have only accounted for the variance due to the orthogonal additive effects of SNPs, ignoring the variance σ_ϵ^2 and covariance between SNPs. The idea is that, by extracting the coefficient estimate $\hat{\boldsymbol{u}}$ from a ridge or lasso regression model, we can estimate the phenotypic variance contribution of SNP j by $\hat{u}_j^2 \text{Var}(\boldsymbol{M}_j)$, which is a metric that can be used as an importance measure for SNP-covariates as an alternative to p-values in GWAS or the mean |SHAP| values in the SHAP-approach to SNP-trait association. Note that the variance of $\boldsymbol{M}_j$ is equal to the variance of its mean-centered version. The estimated PVE from ridge and lasso regression models can be visualized in a Manhattan-style plot similarly to the approaches with p-values and mean |SHAP| values, where we plot the chromosome and bp position along the x-axis.

Here, we apply two regression methods of inference on SNP-trait association, one using ridge regression and one using lasso, both estimating the PVE by SNPs based on the fitted coefficients. The process of generating coefficient estimates $\hat{\boldsymbol{u}}_t$, for $t \in \{R, L\}$, where $\hat{\boldsymbol{u}}_L$ and $\hat{\boldsymbol{u}}_R$ represents the lasso and ridge estimates respectively, is similar to the CV-procedure used in the process for genomic prediction with ridge regression. We again use the `R-glmnet` package and `R-cv.glmnet()` function to perform hyperparameter optimization on the respective λ-values for both ridge and lasso regression in a 10-fold CV procedure, where the dataset $\mathcal{D}$ is partitioned ten times into ten folds. For iteration number i, we define the partitioning of the dataset as $\mathcal{D}^{(i)} = \{\mathbf{M}^{(i)}, \boldsymbol{y}^{(i)}\}$, where $\mathbf{M}^{(i)}$ and $\boldsymbol{y}^{(i)}$ denote the corresponding partitions of the SNP matrix and response vector respectively. The process of generating coefficient estimates $\hat{\boldsymbol{u}}_t$ is the same for the ridge and lasso models and is described in Algorithm 4. The algorithm is written for a general phenotype vector $\boldsymbol{y}$.

Algorithm 4 Computation of coefficient estimates $\hat{\boldsymbol{u}}$ used in inference analysis

Input: Dataset $\mathcal{D} = \{\mathbf{M}, \boldsymbol{y}\}$
for $i = 1, ..., 10$:
  for each λ_k in λ_{seq}:
    $f_k^{(i)} := f(\mathbf{M}^{(i)}; \lambda_k) = \hat{\boldsymbol{y}}_k^{(i)}$
    $\text{MSE}_k^i := \text{MSE}(\boldsymbol{y}^{(i)}, \hat{\boldsymbol{y}}_k^{(i)})$
    end for
  end for
$\lambda^* := \min_{\lambda_k} \frac{1}{10} \sum_{i=1}^{10} \text{MSE}_k^i$
Fit model $f(\mathbf{M}; \lambda^*)$
Return estimator $\hat{\boldsymbol{u}}_t$ from model $f(\mathbf{M}; \lambda^*)$

Note that more modern regression approaches for SNP-trait association have been explored using BayesR (*e.g.*, Pasam et al., 2017). By modeling SNP effects as a mixture of normal distributions with varying variances, BayesR can distinguish between markers with large effects, small effects, and no effect at all. This flexibility makes BayesR particularly well-suited for SNP-trait association analysis. However, applying BayesR can be computationally intensive and involves more complex model fitting and parameter tuning. In this thesis, we instead investigate the feasibility of using ridge and lasso for SNP-trait association, as these methods are computationally simpler and faster to run, while still providing a way to estimate the relative importance of SNPs based on their contributions to phenotypic variance.

3.5 Simulation of phenotype

A challenge in genomic prediction and inference is the lack of control over key factors in different traits, such as the genetic architecture and environmental effects, and validating the results is therefore difficult. Therefore, it often remains unclear why one prediction method outperforms another or whether inference methods such as GWAS and SHAP accurately capture the SNPs associated with the trait of interest. To address these challenges, we turn to simulation of phenotypes on fictive traits, where we can fully control for the genetic architecture and remove environmental noise. Simulated phenotypes allow us to systematically test assumptions, evaluate the prediction and inference methods under known conditions, and compare their behavior across various genetic architectures. Previous studies have employed similar simulation-based approaches to investigate genomic prediction in controlled settings (Meuwissen et al., 2001; Zelioli, 2023; Aspheim et al., 2024).

Ideally, we want the characteristics of the simulated phenotypes to be realistic, such that they mimic the distribution of phenotypes of real traits. Here, we have sampled phenotypes according to the marker-based regression model given in equation (3), while excluding fixed and random effects ($\boldsymbol{\Gamma b} = \mathbf{Z}\boldsymbol{d} = \mathbf{0}$), or in other words, according to equation

$$\boldsymbol{y} = \mathbf{M}_c\boldsymbol{u} + \boldsymbol{\epsilon} \ . \tag{26}$$

In addition, we have to specify the distribution for the SNP effects $\boldsymbol{u}$. We want to be able to carry

out simulations for a diverse set of architectures, so we sampled allele substitution effects from the mixture distribution

$$p(u_j|V_G) = \pi_0\delta_0(\boldsymbol{u}_j) + \pi_1 g(\sigma_u^2, V_G) \ , \tag{27}$$

where $g(\sigma_u^2, V_G) \sim N(0, \sigma_u^2 V_G)$, indicating that the conditional distribution $p(u_j|V_G)$ follows $\delta_0(u_j)$ with probability π_0 and $N(0, \sigma_u^2 V_G)$ with probability π_1, where $\pi_0 + \pi_1 = 1$. Here, $\delta_0(u_j)$ represents the Dirac delta function centered at $u_j = 0$. The mixture distribution in equation (27) allows us to model a range of architectures from polygenic traits ($\pi_1 >> \pi_0$) to oligogenic traits ($\pi_0 >> \pi_1$). We can then represent the simulated breeding values as $\boldsymbol{a} = \mathbf{M}_c\boldsymbol{u}$, exactly like in the marker-based regression model. For simulated phenotypes, we also have the option to control the heritability of the fictive trait, and in this thesis we decided to simulate all phenotypes with a heritability of $h^2 = 0.33$, following previous simulation studies (Zelioli, 2023; Aspheim et al., 2024). Choosing heritability $h^2 = 0.33$ entails that one third of the variation in phenotype is attributed to genetic variation (the additive effect of SNPs). To this end, we simulated effect sizes $\boldsymbol{u}$ according to equation (27) using $\sigma_u^2 = 10^{-2}$ and $V_G = 0.33$ and then generated the breeding values $\boldsymbol{a} = \mathbf{M}_c\boldsymbol{u}$, which were scaled such that $\text{Var}(\boldsymbol{a}) = 0.33$. Finally, we generated the simulated phenotypes according to equation (26), where we sampled $\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{\epsilon}$, using $\sigma_\epsilon^2 = 0.67$. The whole process of generating samples from simulated phenotypes is summarized in Algorithm 5. In order to assess

Algorithm 5 Simulation of phenotype

Input:
$\mathbf{M}_c \in \mathbb{R}^{n\times p}$
Architecture type π_0
For $j = 1, ..., p$:
  $\tilde{u} \sim \text{unif}(0, 1)$
  if $\tilde{u} > \pi_0$:
    $u_j \sim N(0, \sigma_u^2 V_G)$
  else:
    $u_j = 0$
    end if
  end for
$\boldsymbol{a} = \mathbf{M}_c\boldsymbol{u}$
$\boldsymbol{\epsilon} \sim N_n(\mathbf{0}, \sigma_\epsilon^2\mathbf{I}_n)$
return $\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{\epsilon}$

the prediction and inference methods across a diverse set of architectures, we simulate phenotypes of ten fictive traits on the both the 180k and 70k dataset. The architectures of these traits follow the architectures used in Zelioli (2023), and are shown in Table 3.

π_i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
π_0	0.01	0.05	0.1	0.2	0.4	0.6	0.8	0.9	0.95	0.99
π_1	0.99	0.95	0.9	0.8	0.6	0.4	0.2	0.1	0.05	0.01

Table 3: Architectures used for simulating the ten phenotypes.

Having generated the simulated phenotypes, we applied the genomic prediction and SNP-trait association methods developed in this thesis to the simulated phenotypes, enabling us to gain information on the characteristics of the methods across different genetic architectures. Genomic prediction was performed on simulated phenotypes on the 180k dataset using XGB, RR, PCRR and the Bayesian animal model. Note that for the prediction of simulated phenotypes, we can omit the pre-processing of the phenotype used in XGBoost models and the ridge models, since we do not have any fixed non-genetic effects. We performed SNP-trait association analysis on the simulated phenotypes on the 70k dataset using GWAS, the coefficient-based ridge and lasso methods and mean |SHAP| values generated from XGB models.

Knowing the true effect sizes of SNPs $\boldsymbol{u}$ from simulated phenotypes, we can use them to verify the ability of the SNP-trait association methods in identifying important SNPs. A good metric for the true association of a SNP with the trait of interest is the additive contribution the SNP makes to

the phenotypic variation, or, in other words, the PVE by SNPs, defined in equation (25). Here, we generated Manhattan plots of PVE by SNPs in similar style to the SNP-trait association methods analyzed on the simulated phenotypes. The PVEs serve as a reference for the "true" association between SNP and trait that the SNP-trait association methods we analyze can be compared to. In addition to Manhattan-style plots, we generate scatter-plots where we analyze the correlation between the PVEs and metrics used in the SNP-trait association methods, namely p-values for GWAS, mean |SHAP| values for the SHAP-method and estimated PVEs for the ridge and lasso regression models.

We note that the phenotypes are simulated using only linear additive effects, which aligns with the assumptions of the linear models RR, PCRR and the Bayesian animal model. On the other hand, XGBoost, using regression trees as weak learners, are not expected to outperform linear methods in a purely linear setting. The method used for simulating the phenotypes thus favor the linear models and not the XGB model. For the SNP-trait association methods, the linear relationship between SNPs and simulated phenotypes aligns with the assumptions of GWAS and the ridge and lasso regression models. On the other hand, the linearity is not ideal for the SHAP-method, since the mean |SHAP| values, which here are generated from XGBoost predictions, are consequently not ideal in a purely linear environment. It is worth keeping in mind these implications from the linear relationship between SNPs and phenotype in the analysis of genomic prediction accuracy and SNP-trait association methods.

Discrepancies between the analysis of observed house sparrow trait phenotypes and simulated phenotypes may provide insights into the characteristics of the observed house sparrow traits. For example, the XGBoost models predicting much more accurately for the observed phenotypes relative to the simulated phenotypes, might indicate that the observed traits differ in characteristic from the simulated traits. For example, it might indicate that the observed traits are not dominated by linear additive effects, but instead have a considerable contribution from interaction effects, such as dominance or epistatic effects, which the XGBoost models should be ideal in capturing. If the prediction methods vary a lot in prediction accuracy across different genetic architectures on the simulated phenotypes, we may also gain insight into the genetic architecture of observed traits by analyzing the pattern in prediction accuracy across the genetic architectures, and comparing with the accuracy in prediction of observed traits.

3.6 Methods of inference on genetic architecture

In addition to identifying individual SNP-trait associations, understanding the overall genetic architecture of a trait, that is, how genetic influence on a phenotype is distributed across SNPs, is of interest. We therefore turn our attention to methods of inferring the genetic architecture of a trait, or specifically, we develop methods of determining to what degree a trait exhibits a polygenic or oligogenic pattern of genetic contribution. The methods we apply here are extensions of some of the SNP-trait association analysis methods derived so far, namely the SHAP-method and the coefficient-based ridge and lasso methods. These three methods may allow us inference about the genetic architecture of a trait, as they all involve metrics that estimate feature importance or effect sizes of SNPs. This is in contrast to the p-values used in GWAS, as p-values stem from both the magnitude of the effect size and the uncertainty of the effect size. The methods derived in this section thus underline one of the downsides of using the p-value as a metric for assessing SNP-trait associations.

In the SHAP-based approach to identify SNP-trait associations, the mean |SHAP| value over the SHAP values generated from XGBoost predictions is used as a global measure of feature importance for each SNP. A higher mean |SHAP| value indicates that a SNP has a large average contribution to the deviation of the XGBoost model's prediction from the overall mean prediction, and the SNP is therefore considered more influential in the XGBoost genomic prediction model. This property of mean |SHAP| values makes them a useful tool for characterizing the genetic architecture of a trait. Specifically, by examining the distribution of feature importance across SNPs, we can gain insight into whether a trait exhibits a polygenic architecture, where many SNPs contribute with a small estimated importance, or an oligogenic architecture, where only a few SNPs exhibit large estimated feature importance. To visualize this, we computed the mean |SHAP|

value for each SNP, normalized them so that they sum to one, and sorted them in descending order. We then constructed a cumulative sum of these values and plotted the sum against the number of SNPs included. The resulting curve, which we will refer to as the *cumulative feature importance plot*, reveals how quickly feature importance accumulates across SNPs. A steep initial rise followed by a plateau would suggest that a small number of SNPs dominate the predictive power, indicating an oligogenic architecture, whereas a gradual slope would indicate more evenly distributed contributions to the predictive power, which is characteristic of a polygenic architecture.

Cumulative feature importance was computed for coefficient-based ridge and lasso methods of determining SNP-trait associations, using the same approach to that employed on mean |SHAP| values. Here, in place of the mean |SHAP| value, we used the estimated PVE by SNPs, $\hat{u}_j^2 \text{Var}(\boldsymbol{M}_j)$ for SNP j, generated from either the ridge model or the lasso model, depending on the method used. For the ridge and lasso model, the process of generating cumulative feature importance plots was the same as for the SHAP-approach: we normalized the estimated measures of PVE so that they sum to one, ordered them in descending order, then cumulatively added them. For the ridge and lasso methods, instead of prediction power proportion like in the SHAP-approach, the proportion of PVE by SNPs is then plotted against the number of SNPs in the cumulative feature importance plot. We note that similar coefficient-based approaches for estimating variance explained by SNPs have been explored using the BayesR model (*e.g.*, Erbe et al., 2012; Moser et al., 2015).

In interpreting the cumulative feature importance plots from ridge and lasso, it is important to consider the implications of working with a high-dimensional $m \gg n$ genomic dataset. As discussed in Section 2.2.2, ridge and lasso regression can be viewed from a Bayesian perspective, where the estimated regression coefficients correspond to the mode of the posterior distribution obtained by combining a likelihood with a prior on the effect sizes. For ridge regression, a Gaussian prior is used, while for lasso regression, a Laplace prior is used. In high-dimensional settings, the limited sample size means that the likelihood carries relatively little information, so the posterior distribution, and hence the resulting coefficient estimates, tends to reflect the shape of the prior distribution. Thus, we expect the ridge regression coefficients to be approximately normally distributed with effects relatively evenly distributed among SNPs, which is characteristic of polygenic traits. In contrast, lasso regression, due to its Laplace prior, encourages sparsity by shrinking many coefficients exactly to zero, producing a pattern more consistent with oligogenic trait architectures. The problem of "lack of Bayesian learning" is well-known in genomic datasets, and approaches for addressing it have been investigated (Habier et al., 2011; Erbe et al., 2012).

We have now outlined three methods for generating cumulative feature importance plots, namely based on mean |SHAP| values, ridge regression and lasso regression. In order to evaluate the effectiveness of the methods in inferring genetic architecture, we need a benchmark. One key advantage of using simulated phenotypes is that the true genetic architecture is known, which allows us to generate a reference cumulative feature importance plot based on the true PVE by each SNP. The reference is constructed in the same way as the ridge and lasso plots, namely by computing and normalizing the per-SNP contributions to phenotypic variance, except we use the true simulated SNP effect sizes $\boldsymbol{u}$ rather than the estimated coefficients from ridge and lasso. Because we have simulated phenotypes under ten different genetic architectures ranging from highly polygenic to strongly oligogenic, we provide a baseline for what the trajectory of the PVE-curve should look like for each genetic architecture type. By comparing each method's cumulative feature importance curve to the corresponding true PVE curve, we can assess how well each method captures the underlying genetic architecture.

In addition to the feature importance plots, the distributions of mean |SHAP| values, ridge and lasso coefficient estimates and true SNP effects may be used to gain information on the genetic architecture. While the mean |SHAP| values, being feature importance estimates, are not directly analogous to effect sizes of SNP-covariates, we still expect the distribution of mean |SHAP| values to exhibit the same characteristics of the true SNP effects for the underlying genetic architecture, provided that the XGBoost model is able to capture the true relationship between the SNP-covariates and phenotypic response. We expect, for example, the distribution of mean |SHAP| values to be sparse if they are generated from predictions of highly oligogenic traits.

4 Results

4.1 Distribution of phenotype from observed traits and simulated effects

We simulated phenotypes both on the 70k dataset and the 180k dataset for all ten architecture types (Table 3). The distributions of true SNP effect sizes ($\boldsymbol{u}$) for simulated traits on the 70k dataset show that, in the most polygenic architecture ($\pi_0 = 0.01$), effect sizes closely follow a normal distribution centered around zero (Figure 7). As π_0 increases, indicating a higher proportion of SNPs with zero effect, the distribution becomes more sparse, reflecting increasingly oligogenic architectures. The exact same pattern as for the 70k dataset can be seen in the distribution of SNP effect sizes for simulated traits on the 180k dataset (Figure 24).

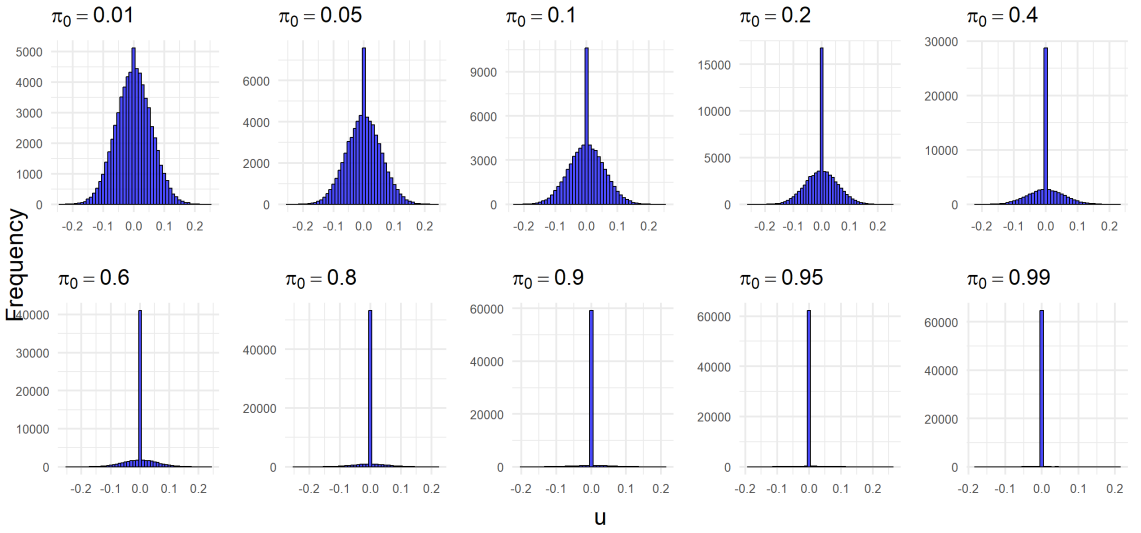


Figure 7: Distribution of SNP effect sizes ($\boldsymbol{u}$) from simulated traits on the 70k dataset across all ten architectures.

We performed analysis on three house sparrow traits from which the phenotypes were observed and measured: tarsus length, body mass and wing length. To understand the underlying structure in the data prior to statistical analysis, we analyzed the distribution of phenotypes from each trait across the 70k dataset (Figure 8). The trait body mass appears to closely follow the normal distribution, as we observe a relatively symmetric, bell-shaped distribution centered around 32 grams, with a variance of 5.521. Tarsus length seems to be approximately normally distributed, with a unimodal distribution centered around 19.5 mm, however the trait does not appear to be completely symmetric, as there seems to be a slight right skewness. Tarsus length has low variation compared to the other two traits, with a variance of 0.641. Wing length, as opposed to the other two traits, appears to not be unimodal, and instead displays a broad multimodel-like distribution with a variance of 5.430.

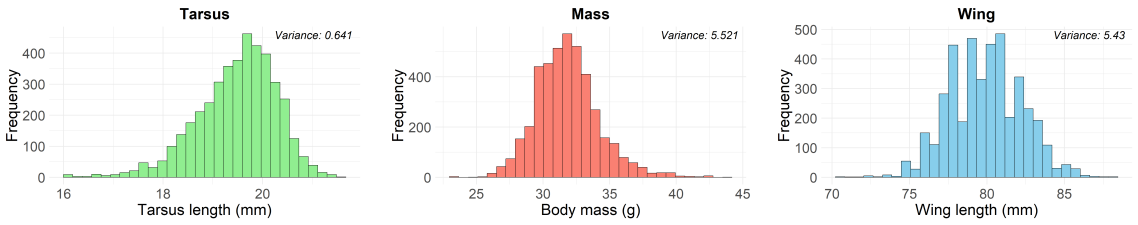


Figure 8: Distribution of observed phenotypes from traits tarsus length, body mass and wing length in the 70k dataset.

4.2 Genomic prediction accuracy

4.2.1 Accuracy on simulated phenotypes

From the genomic prediction on simulated phenotypes, we observe that the accuracies of the three linear models, that is, the Bayesian animal model, ridge regression with SNP covariates (RR) and ridge regression with principal component covariates (PCRR), are nearly indistinguishable across all ten genetic architectures (Figure 9). Between the three linear models, RR seems to have predicted slightly less accurately for some of the architectures, particularly for the oligogenic architecture type $\pi_0 = 0.95$, where the median accuracy of RR is clearly lower than the median accuracies of the Bayesian animal model and PCRR. For the three linear models, the highest median accuracies were recorded for architecture type $\pi_0 = 0.4$, and prediction accuracy seems to be independent of architecture type. For the genomic prediction of simulated traits, XGBoost predicted considerably less accurately than all other models for all types of architecture, with lowest median accuracy in every architecture. XGBoost appears to be slightly more accurate on average for more polygenic architectures, with the two lowest median accuracies recorded on architecture type $\pi_0 = 0.9$ and $\pi_0 = 0.99$, however there are no clear patterns in variability across genetic architectures for XGBoost either.

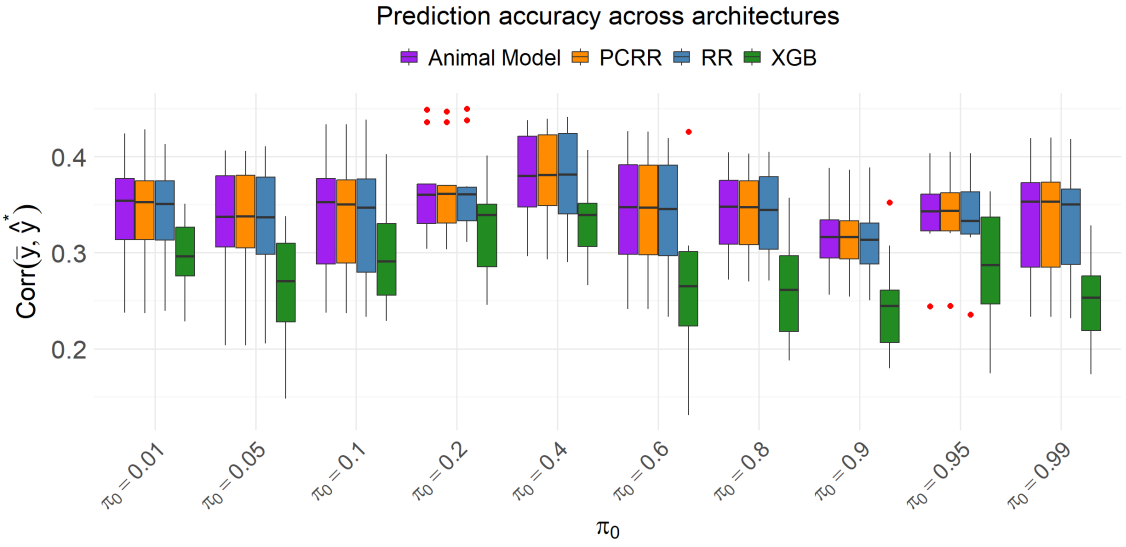


Figure 9: Genomic prediction accuracies on the 180k dataset for the models: Bayesian animal model (Animal model), Ridge regression using SNPs (RR), Ridge regression using PCs from a PCA of all SNPs (PCRR), and XGBoost (XGB) using all SNPs on simulated phenotypes for all ten architecture types.

4.2.2 Accuracy for the house sparrow phenotypes

In the genomic prediction of observed phenotypes of house sparrow traits, we observe that the trait body mass has a much lower average accuracy than wing length and tarsus length considering all prediction models analyzed (Figure 10). The trait tarsus length appears to have the highest accuracy on average considering all prediction models, with all prediction models recording their highest median accuracy for this trait. The pattern of trait-specific differences in prediction accuracy is consistent with previous findings by Aspheim et al. (2024), where body mass also showed lower accuracy than tarsus length and wing length across various prediction models. We observe that the Bayesian animal model recorded consistently high accuracy for all traits. For the trait wing length, the Bayesian animal model was considerably more accurate than all other prediction models, with a median accuracy of approximately 0.37 for this trait. The Bayesian animal model recorded the highest median accuracy score for the trait body mass also, but in this case only slightly higher than the median accuracy of RR and PCRR. For the trait tarsus length, we observe a pattern

similar to that seen in the genomic prediction of simulated traits, where the prediction accuracies of the Bayesian animal model and the two ridge models closely aligned. The prediction accuracies of RR and PCRR are nearly identical across all three observed traits, which is consistent with our findings from the simulated trait analysis, where the two models also closely aligned in accuracy across all ten genetic architectures. Overall, RR and PCRR show slightly lower prediction accuracy than the Bayesian animal model, but achieve higher accuracy than all three XGboost models on average for the observed traits.

In the prediction of observed phenotypes, we recall that three XGBoost models were analyzed, namely the XGBoost model using all SNPs as covariates (XGB), the XGBoost model using 50k randomly chosen SNPs as covariates (XGB_50k), and the XGBoost model using the 10k SNPs most correlated with the response as covariates (XGB_10k). None of the three XGBoost models were consistently as accurate as the three linear models. XGB_50k was the least accurate XGBoost model overall, and also the least accurate prediction model overall, with the lowest median accuracy out of all the models on all three observed traits. XGB_50k was particularly inaccurate for the trait wing length, were it recorded a median accuracy roughly equal to the lowest minimum accuracy out of any of the other prediction models for wing length (approximately 0.24 for XGB and XGB_10k). For the traits wing length and body mass, XGB_10k has a higher median accuracy than XGB, and XGB has higher median accuracy than XGB_10k for the trait tarsus length. Overall, XGB_10k is the most accurate XGBoost model on average, slightly more accurate than XGB.

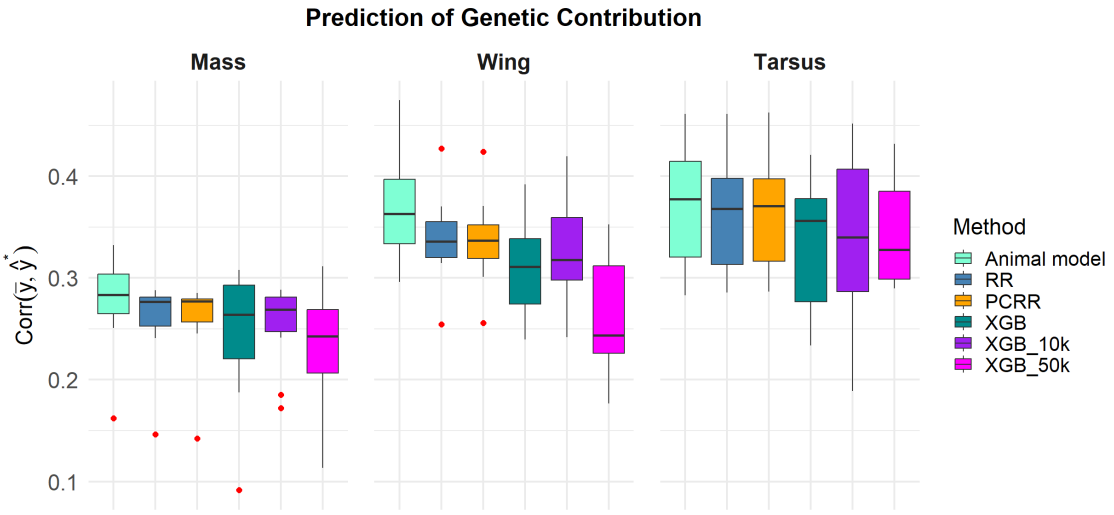


Figure 10: Genomic prediction accuracies on the 180k dataset for the models: Bayesian animal model (Animal model), ridge regression (RR, PCRR), where all SNPs and all PCs from a PCA of all SNPs were used respectively as covariates, and XGBoost (XGB, XGB_50k, XGB_10k), where all SNPs, 50k randomly chosen SNPs and the 10k SNPs most correlated with response were used respectively as covariates. Prediction performed on the three observed traits: body mass (Mass), wing length (Wing) and tarsus length (Tarsus).

4.3 Inference on SNP-trait association

4.3.1 Association analysis on simulated traits

The true proportion of PVE by each SNP, computed from simulated traits on the 70k dataset, was used as reference for evaluating SNP importance across different SNP-trait association methods. For the ridge-based approach of SNP-trait association, we estimated the PVE of SNPs using ridge coefficients. The correlation between the ridge-estimated and true proportions of PVE by SNPs was relatively weak in the most oligogenic architecture ($\pi_0 = 0.99$), with a value of 0.472 (Figure 31). For the most oligogenic architecture, the ridge-estimated PVE is still able to identify some of

the SNPs with the highest PVE. For example, the SNP with the second highest ridge-estimated PVE is also the SNP with the second highest true PVE in this architecture. In fact, the Manhattan plots show that for the most oligogenic architecture, all three SNPs with the highest true PVE are placed in the top seven ridge-estimated PVE, however for architecture type $\pi_0 = 0.01$, none of the three SNPs with the highest phenotypic variance contribution are ranked anywhere near the top in the ridge-estimated PVE (Figure 32). In summary, the ridge-based approach of SNP-trait association is able to identify some important SNPs in the most oligogenic architectures, however the ability to assess SNP-trait associations appears limited, particularly for polygenic architectures.

Similar to the ridge-based approach of SNP-trait association, we estimated the PVE by SNPs by utilizing the coefficients from lasso regression. From the lasso-estimated PVE by SNPs, we observe a relatively strong correlation between PVE and the corresponding lasso-estimated value in the most oligogenic architecture type (0.804), and lasso is able to identify most of the SNPs with the highest PVE (Figure 33). For example, the two SNPs with the highest PVE are also estimated by lasso to be the top two SNPs in PVE. For the second most oligogenic architecture, $\pi_0 = 0.95$, the correlation between PVE and lasso-estimated PVE is much lower (0.260) than in the most oligogenic case, and the correlation keeps decreasing with increasing polygenic architecture, leading to a correlation of 0.046 for the most polygenic architecture type, $\pi_0 = 0.01$. In summary, the lasso-based approach appears slightly better than the ridge-based approach in estimating SNP-trait associations overall, particularly for oligogenic traits.

In the GWAS analysis of SNP-trait association for simulated traits, we compared the resulting p-values from GWAS to the PVE by SNPs. For every genetic architecture, we observe a relatively weak correlation between p-values from GWAS and phenotypic variance contributions, with the highest correlation recorded as being 0.148 in architecture type $\pi_0 = 0.99$ (Figure 11). Even though the correlation between PVE by SNPs and p-values is weak, GWAS is able to identify some of the most important SNPs for highly oligogenic architectures, seen for example from the architectures $\pi_0 = 0.99$ and $\pi_0 = 0.95$. However, even for highly oligogenic architectures, GWAS falsely identifies many SNPs that have little or no PVE, for example the SNP with the highest $-\log_{10}(p)$-value for architecture type $\pi_0 = 0.95$.

In the SHAP-based approach of SNP-trait association, we estimated SNP importances by computing mean |SHAP| values and compared them to the PVE by SNPs. Between the mean |SHAP| values and PVE, we observe a moderate correlation of 0.659 in the most oligogenic architecture, $\pi_0 = 0.99$ (Figure 12). For the most oligogenic architecture, mean |SHAP| values are able to identify most of the important SNPs, and all SNPs that have a mean |SHAP| value higher than 0.010 have at least some non-zero phenotypic variance contribution for this architecture. However, a lot of SNPs that have true PVE of up to 0.010 are given a mean |SHAP| value close to or equal to zero even for the most oligogenic architecture, indicating that the mean |SHAP| values are not able to identify all the important SNPs in the most oligogenic architecture. For only slightly polygenic architectures, the correlation between mean |SHAP| values and PVE breaks down, and already for the second most oligogenic architecture, $\pi_0 = 0.95$, we observe a correlation as low as 0.169. For the second most oligogenic architecture, many SNPs that have zero or close to zero phenotypic variance contribution are given relatively high mean |SHAP| values, for example the SNP with the second highest mean |SHAP| value (0.01). For the most polygenic architecture structures, we observe very weak correlation between mean |SHAP| values and PVE by SNPs. For example, we observe a correlation as low as 0.055 for the most polygenic architecture, $\pi_0 = 0.01$.

We generated Manhattan-style plots from the SHAP-based and GWAS-based approaches to SNP-trait association and compared them to the PVE by SNPs. From the Manhattan plots of p-values from GWAS, mean |SHAP| values and phenotypic variance contributions on simulated traits, we observe that both GWAS and the SHAP analysis are able to identify some of the important SNPs for the most oligogenic architectures ($\pi_0 = 0.99$ and $\pi_0 = 0.95$), but the ability to identify important SNPs quickly breaks down for more polygenic architectures for both methods (Figure 13 and Figure 14). For example, we observe that for architecture $\pi_0 = 0.99$, the SNP with the highest mean |SHAP| value has the second highest phenotypic variance contribution, while for architecture $\pi_0 = 0.95$, none of the three SNPs with the highest phenotypic variance contribution is among the top ten highest mean |SHAP| values. We observe only two SNPs passing the stringent bonferroni threshold for significance, namely in the architectures $\pi_0 = 0.99$ and $\pi_0 = 0.4$ (Figure 27). We

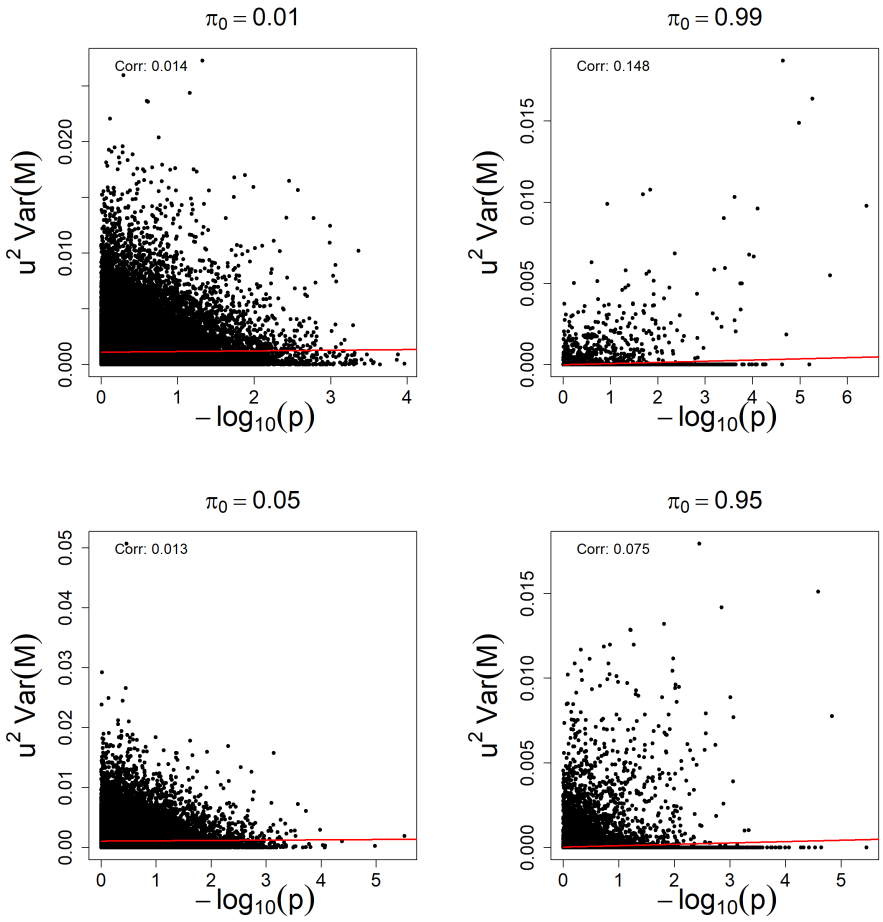


Figure 11: Correlation between p-values from GWAS and PVE on simulated phenotypes, where phenotypes are simulated on the architectures $\pi_0 = 0.01,\ 0.99,\ 0.05,\ 0.95$. Analysis performed on the 70k dataset.

further observe that the significant SNP in architecture $\pi_0 = 0.4$ has a near zero phenotypic variance contribution, indicating that GWAS is susceptible to false positives in highly polygenic architectures (Figure 28).

4.3.2 Association analysis on house sparrow traits

We performed SNP-trait association analysis on the 70k dataset using mean |SHAP| values and GWAS on the observed house sparrow traits, and we found that p-values from GWAS and mean |SHAP| values show weak correlation for all three observed traits (Figure 15). However, it appears that the correlation between mean |SHAP| values and p-values tends to be stronger than the correlation between p-values and PVE in simulated traits. For example, in none of the ten simulated architectures was the correlation between p-values and PVE higher than the correlation between mean |SHAP| values and p-values for the observed traits tarsus length (0.197) and wing length (0.160). A noteworthy observation is that the SNP with the highest mean |SHAP| value for wing length also has one of the lowest p-values in the GWAS analysis, ranking it as the third lowest among the SNPs for wing length.

From the Manhattan plots of mean |SHAP| values and p-values of observed traits, we see that there are no SNPs reaching the stringent bonferroni threshold for significance (Figure 30). Wing length and tarsus length are the traits with SNPs closest to significance in the GWAS analysis with at least one SNP each exceeding the $-\log_{10}(5)$ level. In general, the p-value distributions of

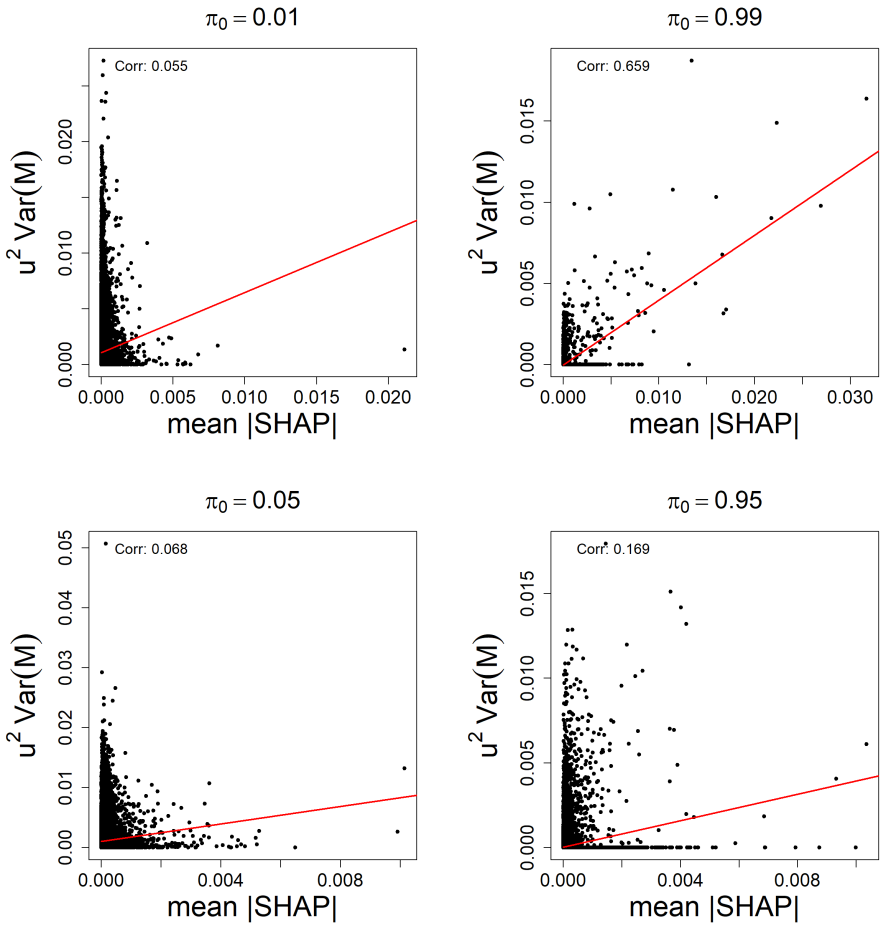
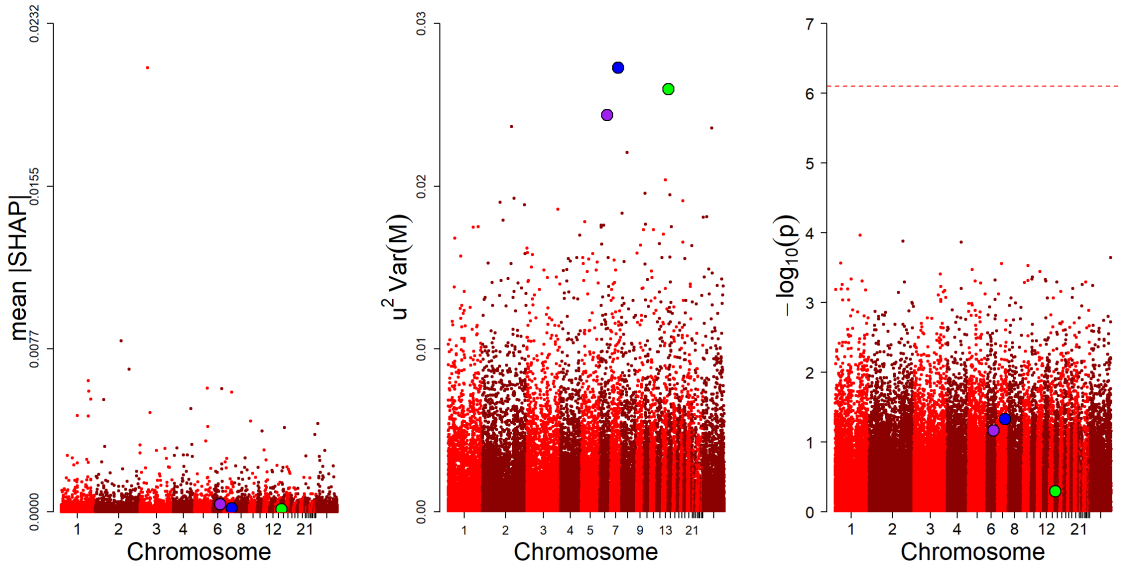


Figure 12: Correlation between mean |SHAP| values from XGBoost predictions and PVE on simulated phenotypes, where phenotypes are simulated on the architectures $\pi_0 = 0.01,\ 0.99,\ 0.05,\ 0.95$. Analysis performed on the 70k dataset.

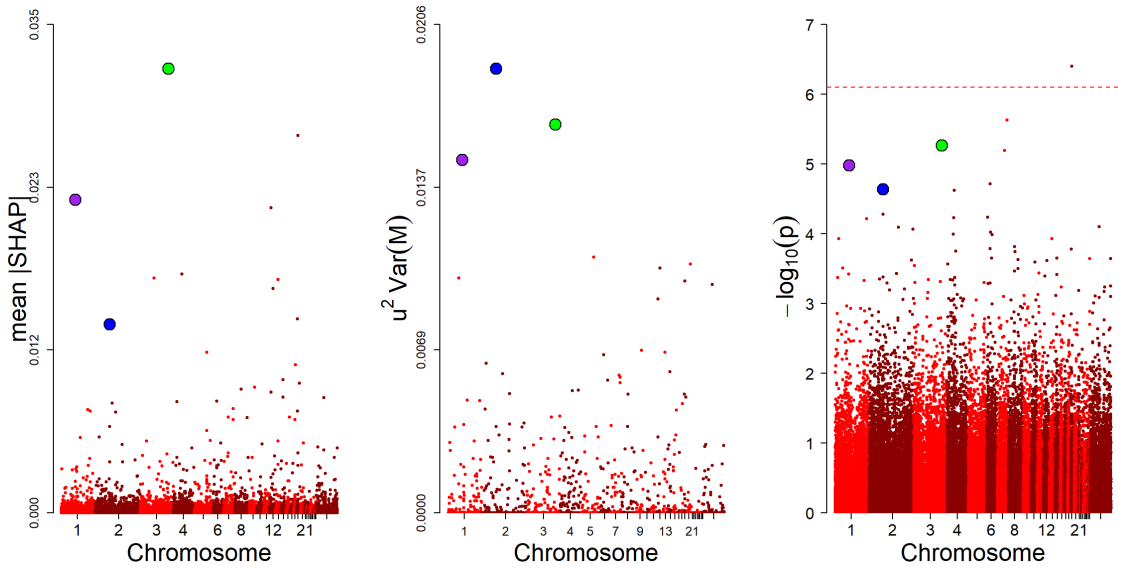
tarsus length and wing length indicate stronger associations than those observed for body mass. We observe a difference in scale of mean |SHAP| values across the three traits. For tarsus length, the values reach the order of 10^{-2}, whereas for wing length and body mass, they reach to the order of 10^{-1}. In summary, neither GWAS nor the SHAP Manhattan plots identify any genomic regions that clearly stand out for any of the three observed traits analyzed. However, we note the SNP on chromosome two with the highest mean |SHAP| value for wing length, as it appears to be a notable outlier, and, as discussed previously in this section, this SNP also has one of the lowest p-values in the GWAS analysis for wing length.

4.3.3 Inference on genetic architecture for simulated and observed traits

Using the simulated traits on the 70k dataset with known genetic architectures as reference, we evaluated how well ridge regression, lasso regression and SHAP-based SNP importance estimates capture patterns of genetic architecture, and applied the SHAP-based approach to observed house sparrow traits. The cumulative PVE by SNPs shows a clear dependence on genetic architecture, where more oligogenic trait architectures converge toward 1.0 more rapidly as SNPs are included (Figure 16). For example, in the most polygenic architecture, only around 60% of the variance is explained by the 10 000 SNPs with the largest contributions to phenotypic variance. In contrast, for the two most oligogenic architectures ($\pi_0 = 0.99$ and $\pi_0 = 0.95$), all, or nearly all of the variance is explained by the top 1000 SNPs in terms of PVE.



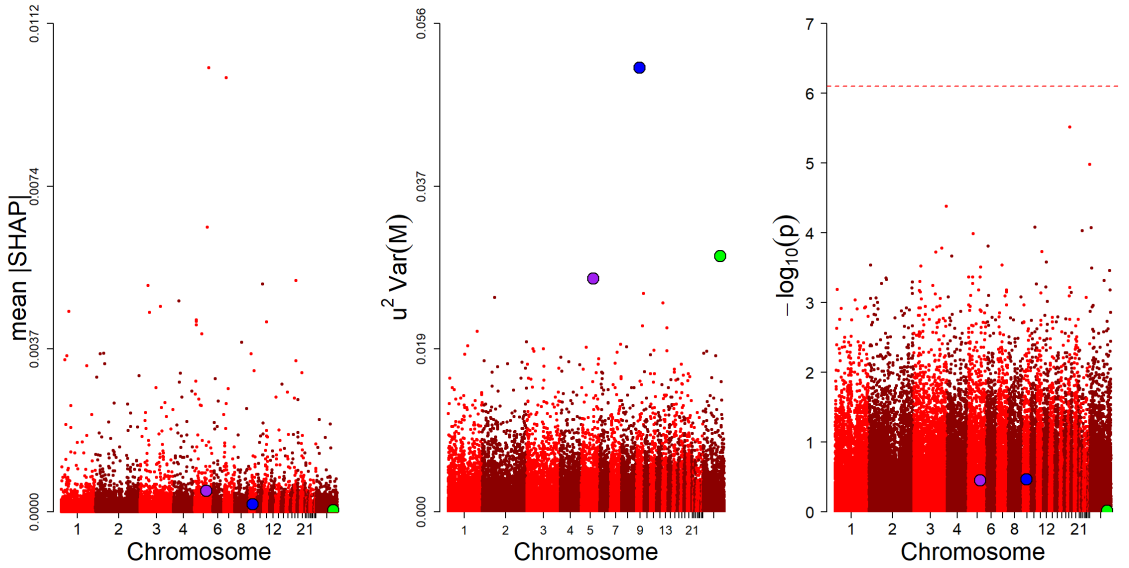
(a) $\pi_0 = 0.01$



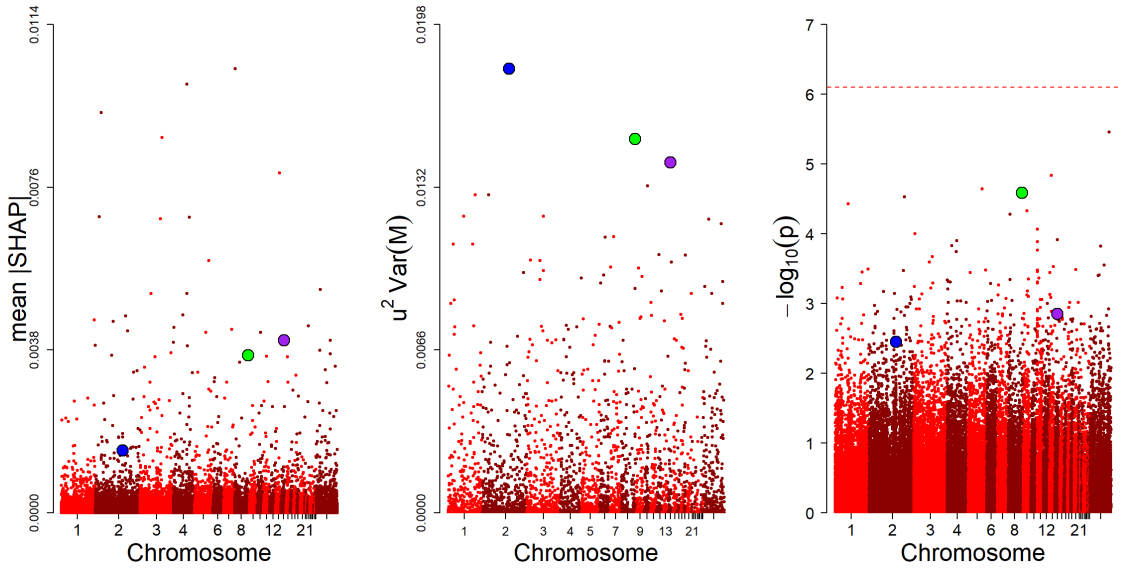
(b) $\pi_0 = 0.99$

Figure 13: Manhattan plots of mean |SHAP| values (left), PVE (middle), and p-values from GWAS (right) for simulated phenotypes on architectures $\pi_0 = 0.01,\ 0.99$. Analysis performed on the 70k dataset. The three SNPs with highest true PVE are highlighted.

From the distributions of mean |SHAP| values on simulated phenotypes, we observe that the distributions display a sharp peak near zero with relatively large outliers, irrespective of the underlying genetic architecture of the simulated trait (Figure 17). Notably, the mean |SHAP| value distributions appear highly similar across the different genetic architectures, with a pattern that is characteristic of cumulative PVE by SNPs of oligogenic architectures (see Figure 7). The cumulative mean |SHAP| value curves (SHAP curves) of simulated phenotypes (Figure 18), on the other hand, seem to follow a trajectory similar to the trajectory we see from the cumulative PVE by SNPs of polygenic architectures (see Figure 16). For example, at 10 000 SNPs, the highest predictive power proportion we observe for SHAP curves is approximately 0.6, which is comparable to the proportion of PVE by the most polygenic architecture type ($\pi_0 = 0.01$) at 10 000 SNPs. Even though all ten SHAP curves seem to follow polygenic trajectories, we can clearly see that the SHAP curves from oligogenic traits tend to converge faster than the SHAP curves from polygenic



(a) $\pi_0 = 0.05$



(b) $\pi_0 = 0.95$

Figure 14: Manhattan plots of mean |SHAP| values (left), PVE (middle), and p-values from GWAS (right) on simulated phenotypes on architectures $\pi_0 = 0.05,\ 0.95$. Analysis performed on the 70k dataset. The three SNPs with highest true PVE are highlighted.

traits. However, the convergence pattern seems highly variable, and some oligogenic SHAP curves converge more quickly than some polygenic ones.

Turning to ridge regression, we examined how the distribution of estimated SNP coefficients and the cumulative ridge-estimated PVE by SNPs relate to underlying genetic architecture. For ridge regression, we observe that the distributions of estimated regression coefficients across all ten architecture types closely resemble a normal distribution centered near zero (Figure 19). Normally distributed coefficients are characteristic of polygenic architectures, as illustrated by the distribution of SNP effect sizes on architecture $\pi_0 = 0.01$ in the simulation study (Figure 7). As for the mean |SHAP| value distributions, the similarity in distribution of ridge coefficient distributions makes it difficult to distinguish between different genetic architectures, which is further supported by the ridge-estimated cumulative PVE curves (ridge curves), which show substantial overlap

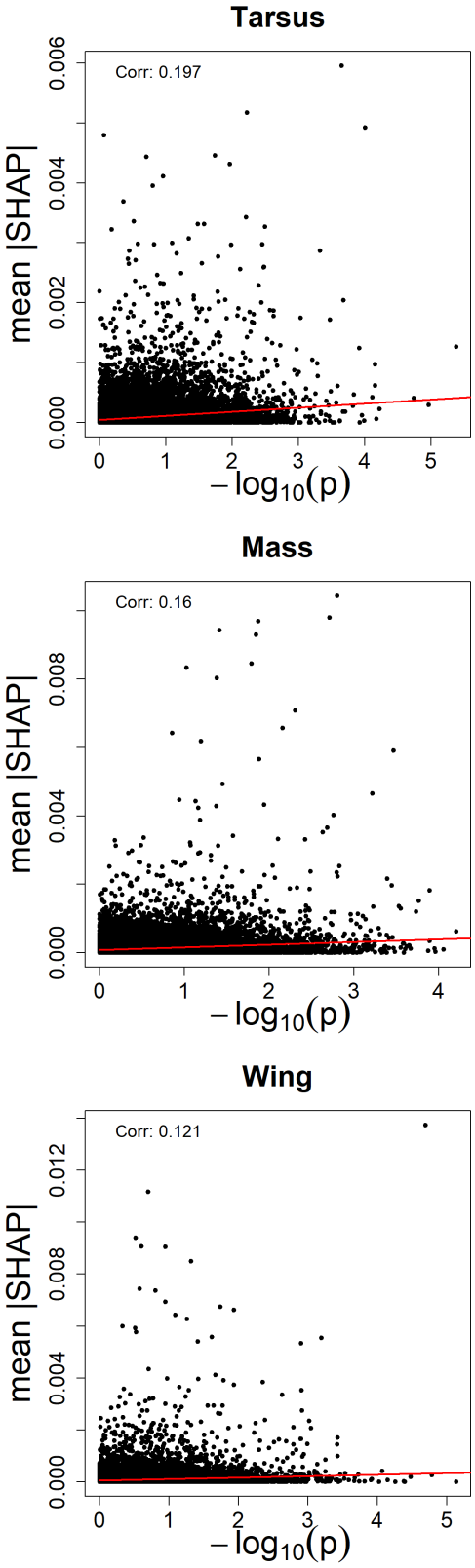


Figure 15: Correlation between mean |SHAP| values from XGBoost predictions and p-values from GWAS on the three observed traits body mass (Mass), tarsus length (Tarsus) and wing length (Wing). Analysis performed on the 70k dataset.

across all ten architectures, indicating that ridge coefficients alone offer little inference about ge-

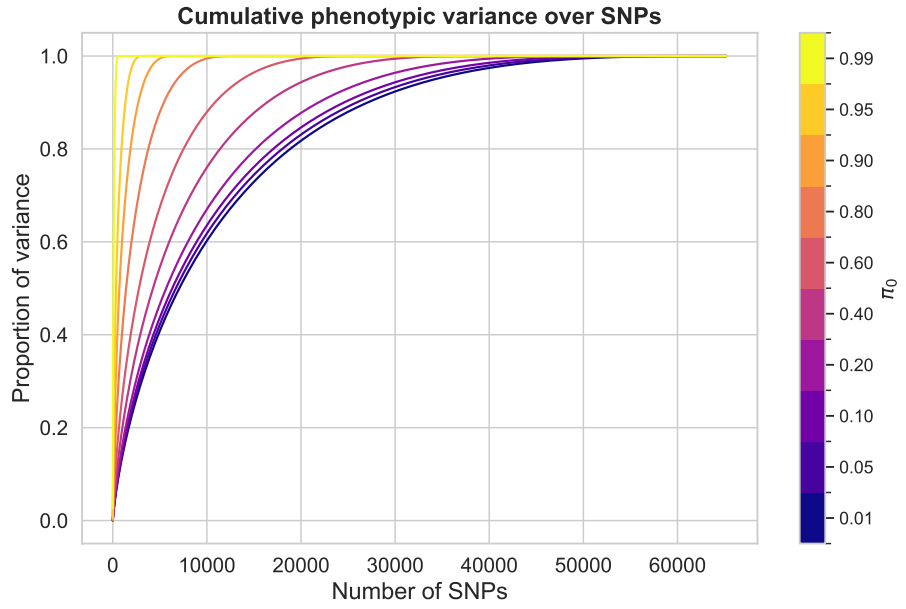


Figure 16: Cumulative phenotypic variance curves generated from simulated SNP effects $\boldsymbol{u}$ on the 70k dataset and for all ten architecture types.

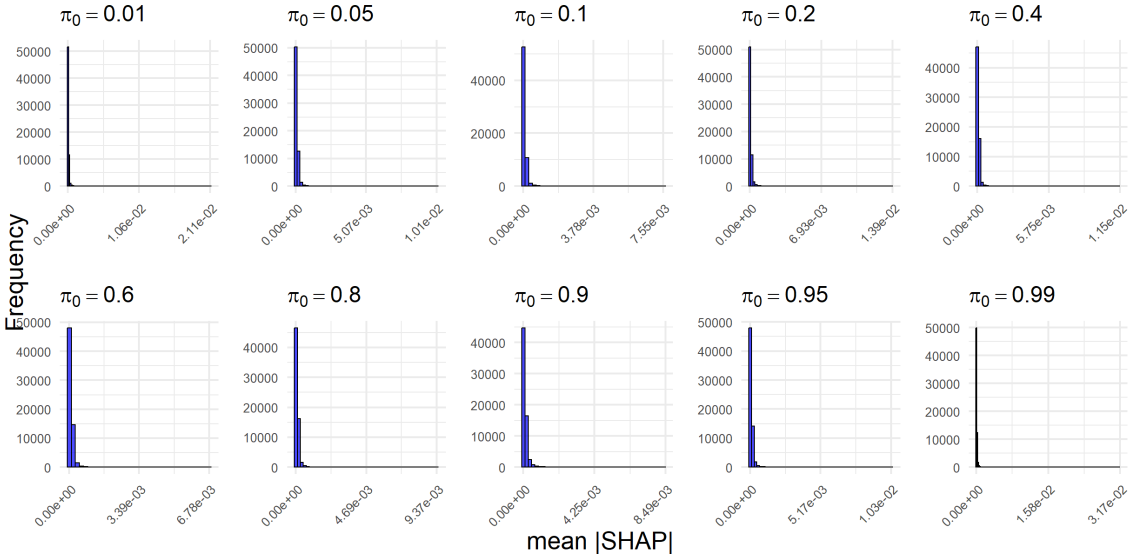


Figure 17: Distributions of mean |SHAP| values generated from XGBoost predictions on simulated phenotpes across all architecture types on the 70k dataset.

netic architecture (Figure 20). The ridge curves all seem to follow a trajectory similar to the trajectories we see for cumulative PVE of polygenic traits (Figure 16). In fact, the ridge curves closely resemble the cumulative PVE curve of architecture $\pi_0 = 0.2$. For example, at 20 000 SNPs included, approximately 83% of the variance in phenotype is explained in architecture type $\pi_0 = 0.2$, which is approximately the same proportion of ridge-estimated PVE by the ridge curves using 20 000 SNPs.

Similar to the ridge-based analysis, we evaluated how lasso regression reflects genetic architecture through the distribution of estimated SNP coefficients and the cumulative lasso-estimated phenotypic variance. We observe that the distributions of lasso coefficients, across all ten architecture types, have a sharp peak near zero with a few relatively large (positive and negative) outliers (Figure 21). As mentioned in the analysis of mean |SHAP| value distributions previously in this section, this distribution pattern is characteristic of oligogenic architectures. The oligogenic distri-

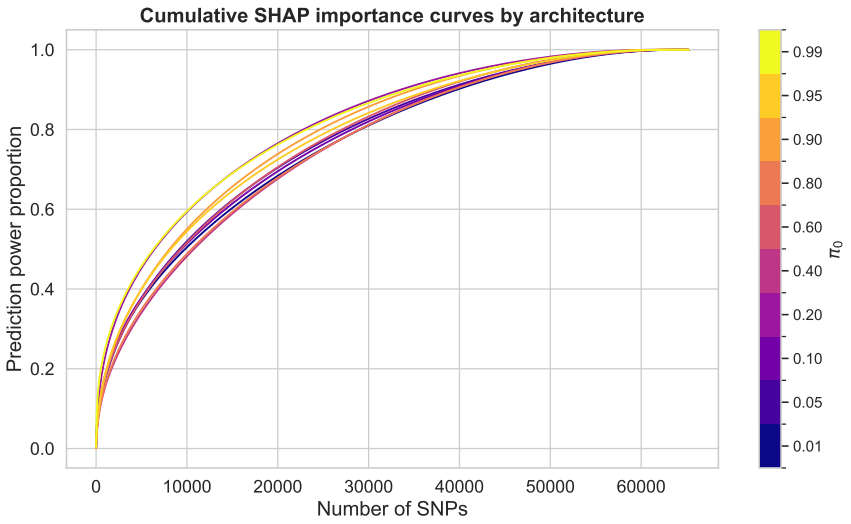


Figure 18: Cumulative mean |SHAP| importance curves generated from mean |SHAP| values of XGBoost predictions on all architecture types of simulated phenotypes on the 70k dataset.

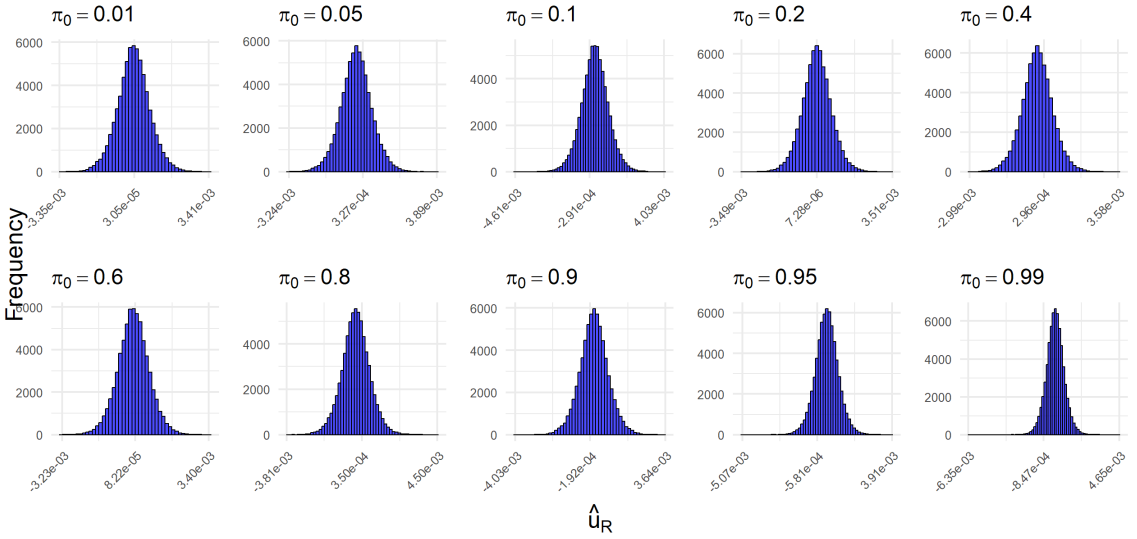


Figure 19: Distributions of ridge coefficients generated from ridge predictions on simulated phenotypes across all architecture types on the 70k dataset.

bution pattern of lasso coefficients appear very similar in all architecture types, making it difficult to infer the genetic architecture from the distributions alone. The cumulative lasso-estimated PVE curves (lasso curves; Figure 22) is consistent with the observation of oligogenic distributions, as the lasso curves of all ten architectures appear to follow the trajectories of oligogenic trait curves in cumulative PVE (see Figure 16). In fact, all the lasso curves appear to follow the trajectory of the cumulative PVE curve of the most polygenic trait ($\pi_0 = 0.99$), all quickly rising towards 1.0 in proportion of estimated PVE, indicating that lasso places the majority of the contribution in phenotypic variance on a few SNPs. The oligogenic assumption is also clearly observed when comparing the Manhattan plot of PVE and lasso-estimated PVE in the most polygenic architecture (Figure 34).

We applied the SHAP-based approach to inferring genetic architecture to observed house sparrow traits, and compared the resulting SHAP curves to those from simulated traits with known architectures. From the SHAP curves of observed traits (Figure 23), we observe a pattern consistent with the SHAP curves from the simulated traits, namely that the SHAP curves of the three observed traits exhibit curve trajectories similar to the cumulative phenotypic variance curves of oligogenic

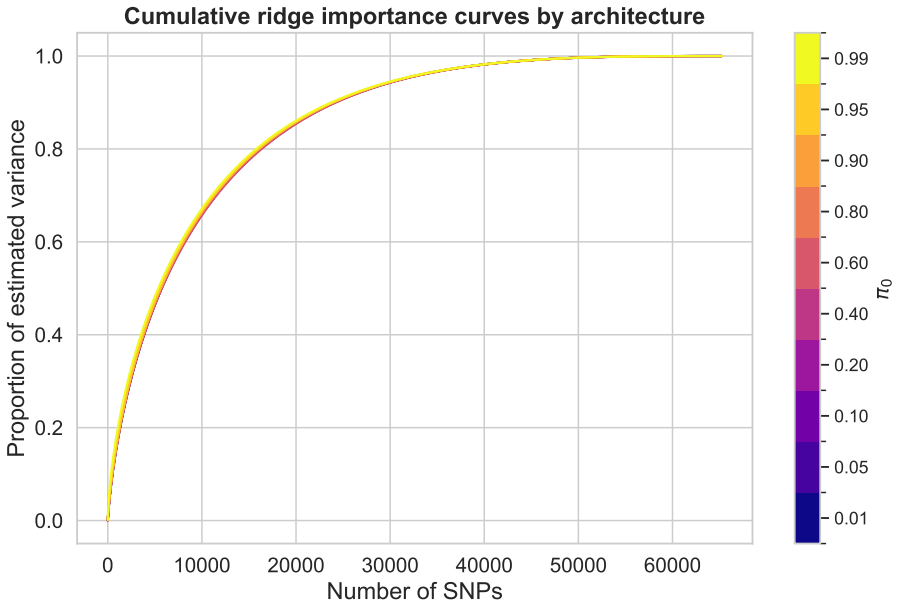


Figure 20: Cumulative feature importance curves generated from ridge-estimated PVE, $\hat{u}^2_{R,j}\text{Var}(\boldsymbol{M}_j)$, where ridge coefficients $\hat{\boldsymbol{u}}^2_R$ were generated from ridge predictions. Analysis performed on all architecture types π_0 of simulated traits on the 70k dataset.

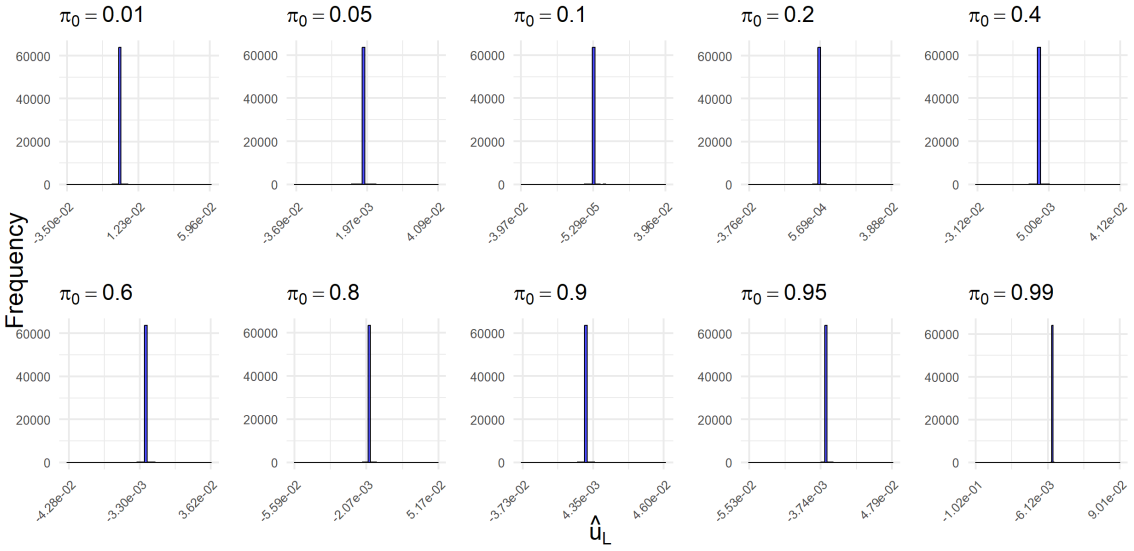
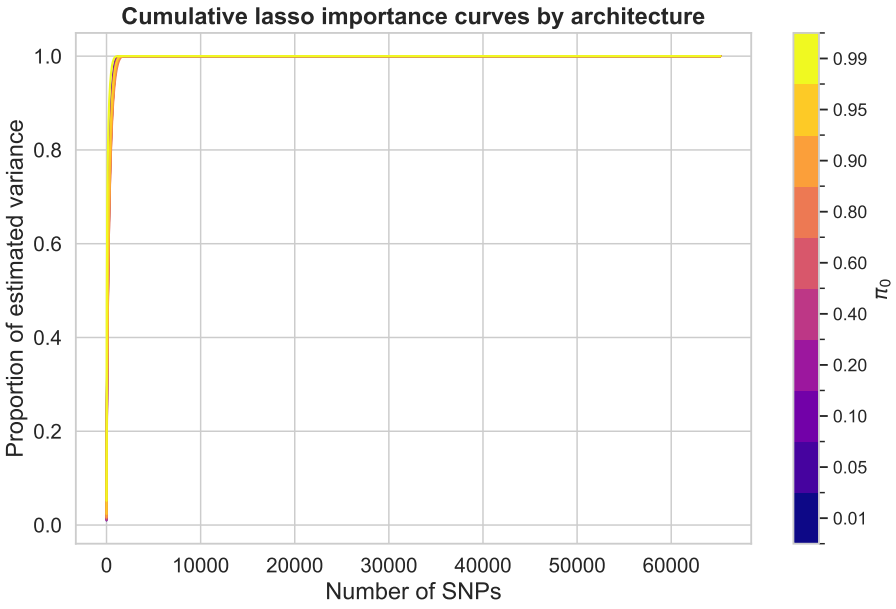


Figure 21: Distributions of lasso coefficients generated from lasso predictions on simulated phenotypes across all architecture types on the 70k dataset.

traits (Figure 16). The SHAP curves of tarsus length and wing length reach a proportional predictive power of approximately 0.6 at 10 000 SNPs, which closely aligns with the simulated SHAP curve of architecture $\pi_0 = 0.99$ (Figure 20). The body mass curve follows a trajectory more similar to the simulated SHAP curves of architectures $\pi_0 = 0.95$ and $\pi_0 = 0.9$, which suggests that body mass exhibits a more polygenic architecture compared to tarsus length and wing length.



!ht

Figure 22: Cumulative feature importance curves generated from lasso-estimated PVE where lasso coefficients $\boldsymbol{u}_L^2$ were generated from lasso predictions. Analysis performed on all architecture types π_0 of simulated traits on the 70k dataset.

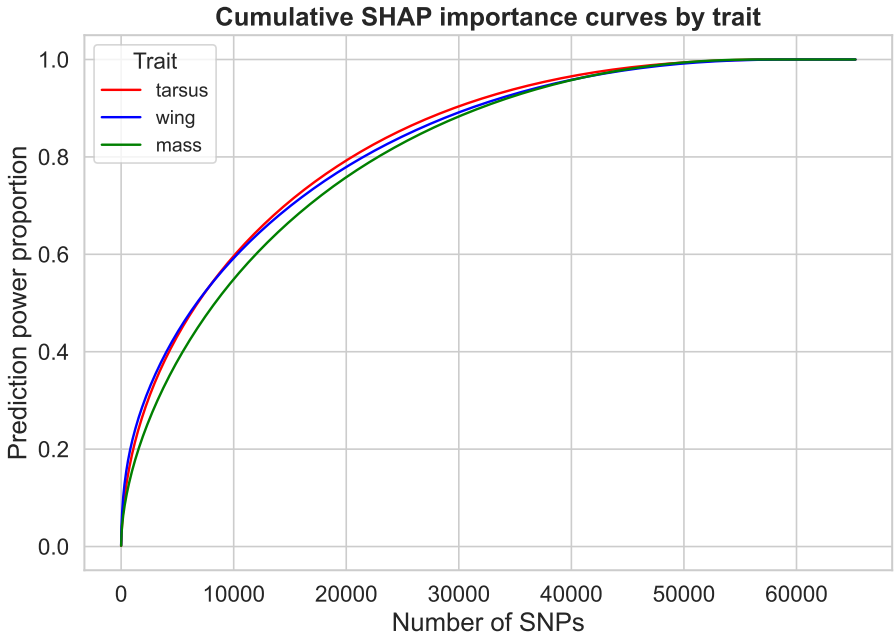


Figure 23: Cumulative SHAP importance curves generated from mean |SHAP| values of XGBoost predictions on the observed traits body mass (mass), wing length (wing) and tarsus length (tarsus) on the 70k dataset.

5 Discussion

We have in this thesis investigated boosted regression trees in genomic prediction using the XGBoost model and SNP-trait association methods in a wild meta-population of house sparrows (*Passer domesticus*) inhabiting the Helgeland island system in Northern Norway. Our analysis focused on the three morphological traits tarsus length, wing length, and body mass. We applied a range of statistical and machine learning models to two SNP datasets that differ in genomic resolution and sample size. The small 180k dataset provided dense genomic information for roughly 2 000 unique individuals and was used for genomic prediction studies, while the larger 70k dataset with roughly 13 000 unique individuals offered increased sample size and was used for SNP-trait association studies. For genomic prediction, we evaluated the accuracy of three XGBoost models that differed in their SNP input, one which used all available SNPs, one which employed a random subset of 50 000 SNPs, while the final model utilized the top 10 000 SNPs most strongly correlated with the phenotype. The XGBoost models were compared to a state-of-the-art Bayesian genomic animal model and two ridge regression models, where one model utilized SNPs directly as covariates, while the other utilized PCs generated from PCA of the SNP matrix. All models were applied on the three house sparrow traits to assess their predictive accuracy. To investigate SNP-trait associations, we compared a traditional GWAS approach using univariate mixed models to SHAP values derived from XGBoost predictions.

Our aim was to assess the utility of XGBoost as an alternative to established genomic prediction methods such as the Bayesian animal model, ridge regression and principal component based ridge regression, and to explore alternative approaches to GWAS for identifying SNP-trait associations in complex traits in wild populations. We also simulated ten traits across a range of genetic architectures according to marker-based regression, allowing us to evaluate model performance in a controlled setting. We performed genomic prediction across the genetic architectures using the Bayesian animal model, both ridge regression variants and the XGBoost model using all SNPs. We also carried out SNP-trait association analyses using GWAS, SHAP values and coefficient-based approaches with ridge and lasso on the simulated phenotypes across the genetic architectures.

We found that the distribution of phenotypes of both body mass and tarsus length exhibit characteristics that align with the normal distribution, while wing length phenotypes did not appear to follow a unimodal distribution. A possible explanation for the difference in distribution of phenotypes is that the degree of variation between sexes may differ across traits. The results might indicate that wing length show greater sex-based variation, resulting in a multimodal phenotype distribution, while body mass and tarsus length vary less in phenotype between sexes. This type of sex-based phenotypic variation is precisely the kind of effect we targeted for removal when generating the pre-processed phenotype used in the XGBoost- and ridge-models. Hence, the results indicate that the effects of phenotype pre-processing is most pronounced for the trait wing length.

From the genomic prediction results on the simulated phenotypes, we found that the three linear models analyzed, the Bayesian animal model, and the two ridge variants, had nearly indistinguishable prediction accuracies across all genetic architectures. On the other hand, the Bayesian animal model gave more accurate predictions than the two ridge models on average for the observed house sparrow traits wing length, tarsus length and body mass. The different strategies in accounting for non-genetic effects on the observed phenotypes between the Bayesian animal model and the two ridge models might explain the discrepancy in accuracy score between simulated and observed phenotypes. We recall that for the prediction of observed traits, the two ridge models account for non-genetic effects in a two-step approach by performing pre-processing on the phenotype prior to model training, whereas the Bayesian animal model includes the non-genetic effects directly in the linear mixed model. However, for the simulated phenotypes, there are no non-genetic effects except for the random residual errors, hence the prediction accuracy on simulated phenotypes reflects only the ability to predict additive effects. In other words, the deviation of the ridge variants from the Bayesian animal model in accuracy score for the observed phenotypes may reflect the impact of phenotype pre-processing. In support of this, we observed that the difference in prediction accuracy between the Bayesian animal model and the ridge regression variants is largest for the trait wing length, which is also the trait that indicated the most pronounced effect from phenotype pre-processing. However, the deviation between the Bayesian animal model and the

ridge regression variants on observed phenotypes could also be influenced by other factors, such as differences in the models' capacity to capture non-additive effects.

The genomic prediction results on both observed and simulated phenotypes indicate that XGBoost generally achieved lower accuracy than the three linear models in predicting the genetic contribution to phenotype. None of the three XGBoost models analyzed here achieved a higher median accuracy than the linear models for any of the traits. One possible reason is that XGBoost, which builds predictions using ensembles of regression trees, may not capture additive genetic effects as effectively as models that are explicitly linear. A linear relationship between feature and response is approximated by splits in a regression tree, which generally speaking is inefficient. While linear models are well-suited to capturing additive effects outside the range of the training data, tree-based models like XGBoost assign constant values to the terminal nodes of trees and lack the ability to extrapolate beyond observed data, which may be particularly limiting in high-dimensional genomic settings. Furthermore, in the implementation of the Bayesian hyperparameter optimization, we focused on a predefined set of parameters assumed to be the most important. Many hyperparameters were left at their default values so that we could explore the hyperparameter space of the chosen parameters more thoroughly. Here, we might not have chosen the best set of hyperparameters to optimize, and a different choice of hyperparameters could perhaps have led to more accurate predictions. A more comprehensive hyperparameter optimization, including increased function evaluations, might also improve model performance but was outside the scope of our current computational constraints. Additionally, we used a custom CV strategy for the XGBoost models due to computational constraints, but the more rigorous nested CV approach would likely provide more reliable hyperparameters, leading to increased prediction accuracy. Finally, the choice of MAE as the loss function for the XGBoost models may not have been optimal, and using MSE instead could have placed greater emphasis on outliers, potentially improving the prediction accuracy. It is worth noting that XGBoost also relies on the pre-processed phenotype and may therefore be similarly affected by the potential limitations introduced in the two-step correction procedure, as observed with the ridge regression approaches.

Among the three XGBoost models evaluated, the XGBoost model with a random subset of 50 000 SNPs was the least accurate across all traits. In fact, the random subset model was considerably less accurate than the XGBoost model using all SNPs even though the random subset model used twice as many evaluations in the Bayesian hyperparameter optimization (40 evaluations, compared to 20). This result suggests that 50 000 randomly selected SNPs does not reliably yield a representative set of SNPs capable of capturing the full phenotypic variation attributable to the genetic markers in the dataset. In contrast, the XGBoost model using the 10 000 SNPs most correlated with the phenotype demonstrated substantially better predictive accuracy, despite using far fewer SNPs than the random subset model. The accuracy of the XGBoost model using the SNPs most correlated with the phenotype was comparable to, or even slightly better than, the XGBoost model using the full set of approximately 180 000 SNPs. Hence, the results indicate that the strategy of selecting the SNPs most correlated with the phenotype is a promising approach for capturing the relevant genetic signal. Using the same approach but adjusting the subset size could potentially lead to even greater accuracy, possibly approaching the accuracy of the linear models. Given the encouraging results of the subset approach using SNPs correlated with the phenotype, alternative subset selection strategies are worth exploring. For instance, selecting the most correlated SNPs within non-overlapping genomic windows could help ensure even genomic coverage while still prioritizing markers associated with the phenotype. A similar subset selection approach has been explored for genomic prediction using neural networks (Singsaas, 2024).

Due to time constraints, a systematic analysis of computational complexity was not conducted. However, based on practical experience during model fitting, we found that the ridge regression models were by far the most efficient. In contrast, the XGBoost models, particularly the model using all available SNPs, were substantially more computationally demanding due to the Bayesian hyperparameter procedure. However, reducing the number of SNPs in the two subset approaches led to a noticeable decrease in computational cost. For the observed traits, the Bayesian animal model consistently achieved the highest accuracy in prediction, however the two ridge regression variants emerge as practical and fast alternatives that still yield relatively accurate predictions. From a computational complexity perspective, it is worth noting that we included all n PCs from the PCA in the PCRR model. However, Aspheim et al. (2024) showed that for a Bayesian

implementation of PCRR, prediction accuracy remains robust even when using only a subset of the leading PCs, indicating that PCRR has the potential to be even more computationally efficient without sacrificing accuracy. Hence, PCRR appears to be a promising alternative for future applications in the face of expanding genomic datasets.

The prediction accuracy for body mass was considerably lower on average across all models compared to wing length and tarsus length, while the prediction accuracy for tarsus length was slightly higher than for wing length. The lower accuracy for body mass suggests that it likely has lower heritability than wing length and tarsus length, since genomic prediction accuracy has been shown to increase with trait heritability (Meher et al., 2022). Previous studies have also found that the average prediction accuracy for body mass is lower than that for wing length and tarsus length in the house sparrow, across various models (Aspheim et al., 2024). Moreover, tarsus length demonstrated the least variation in accuracy between models, with all six models predicting with relatively similar accuracy on this trait. Notably, the three XGBoost models were only slightly less accurate than the three linear models for the prediction of tarsus length. This result might indicate that tarsus length is more influenced by interaction effects, such as dominance or epistasis, compared to the other two traits, as we expect XGBoost to be more capable of modeling such interactions. More broadly, the smaller accuracy gap between XGBoost and the linear models on the observed traits, compared to the simulated traits, may point to the presence of interaction effects in all three observed traits. When evaluating the impact of genetic architecture on the prediction accuracy on simulated phenotypes, we observe no clear pattern for the linear models, as they predicted with similar accuracy across polygenic and oligogenic architectures. XGBoost, however, exhibited slightly more accurate predictions on oligogenic traits. Overall, the impact of genetic architecture on prediction accuracy appears limited. This contrasts with previous studies, for example Meher et al. (2022), where GBLUP achieved higher genomic prediction accuracy for traits controlled by many small-effect QTLs.

The analysis of SNP-trait association methods using GWAS, SHAP values from XGBoost and coefficient-based estimates of PVE from ridge and lasso regression revealed several challenges in reliably identifying SNPs associated with the expression of a trait. Across all methods, the strength of correlation between the method's SNP importance metric and true PVE by SNPs was highest under the most oligogenic architecture and deteriorated rapidly as the genetic architecture became more polygenic. This pattern reflects the broader difficulty in SNP-level inference when many small effects are involved, which is a fundamental part of the missing heritability dilemma. It is important to note, however, that the true PVE values used as a benchmark in this analysis are themselves an approximation. Specifically, they account only for the orthogonal additive effects of individual SNPs and ignore both residual variance and the covariance between SNPs. The omission of the component arising from covariance between SNPs is particularly important to note. Covariance among SNPs due to LD can contribute substantially to the total genetic variance, but have been omitted due to computational considerations. Additionally, in our simulations, SNP effects were drawn independently from the same mixture distribution independent of genomic position. While this setup provides a controlled setting, it differs from real-world scenarios where effect sizes are often clustered in genomic regions (*e.g.*, QTL hotspots). Such spatial dependencies could, in reality, amplify signals in SNP neighborhoods and improve association performance for all four methods.

Among the four approaches, GWAS showed the weakest overall correlation with true PVE across all simulated architectures. This is expected, as GWAS p-values reflect a combination of effect size and statistical uncertainty. In fact, the result reflects a motivation behind looking for an alternative to GWAS: the p-value is not the most informative metric for assessing the association between SNPs and the phenotype. Interestingly, despite the low correlation with PVE, GWAS were still able to identify some of the top-ranking SNPs under highly oligogenic architectures. However, even for highly oligogenic architectures, GWAS falsely identified several SNPs that had little or no PVE. Overall, the GWAS results underline the motivation for searching for an alternative.

For the SHAP-based approach of assessing SNP-trait associations, we used mean |SHAP| values as an importance measure for SNPs. This method did relatively well for highly oligogenic architectures, where it identified key SNPs (in terms of true PVE) with few "false positives" and showed moderate correlation with true PVE. However, even in the highly oligogenic setting, some influential SNPs were missed, and the method's ability in finding associated SNPs declined markedly

as the trait architecture became more polygenic. For traits where more than 1% of SNPs were expected to have causal effects, the accuracy of the SHAP approach deteriorated, consistent with the pattern seen for all association methods analyzed in this thesis.

The SHAP-based approach of assessing SNP-trait associations might be particularly disadvantaged by our choice of PVE-benchmark. It is important to note that our benchmark for true PVE includes only orthogonal additive effects and ignores other sources of variance, particularly covariance between SNPs. However, the SHAP values were derived from predictions of the XGBoost model, which is well-suited in capturing non-linearity and interaction effects. Therefore, the mean |SHAP| values likely also reflect interaction effects among SNPs. While this capacity to capture interactions is a strength of XGBoost in prediction, it poses a limitation when comparing SHAP-derived importance measures to an additive-only PVE benchmark. Consequently, the correlation between mean |SHAP| values and PVE may be underestimated, whereas linear association methods are less affected by this mismatch. This implies a limitation of the benchmark itself, rather than the SHAP method. If a benchmark that accounts for covariance and interactions was used, the SHAP approach might align more closely with true SNP importance, while linear methods might appear less accurate by comparison.

Moreover, it is essential to consider that SHAP values explain the predictions of the underlying machine learning model, in this case, XGBoost. As shown in the genomic prediction results, XGBoost was not among the most accurate predictors of the genetic component of the phenotype. If the predictive model fails in capturing the true relationships between SNP-covariates and the trait, then the SHAP values explaining those predictions will likewise fall short as indicators of true SNP-trait associations. This is a limitation of XGBoost, not the SHAP framework itself. Since SHAP can, in principle, be applied to any machine learning model, future studies could explore its use in combination with more accurate predictive models to potentially improve inference of SNP-trait relationships.

For the lasso and ridge approaches to inferring SNP-trait associations, we found that both methods offered more promising results than GWAS in identifying important SNPs. Lasso-estimates of PVE, in particular, showed the highest correlation with true PVE when the number of causal SNPs was small. This result reflects the lasso model's inherent sparsity-inducing property, which makes it well-suited to detecting a small number of large-effect SNPs. However, this same property becomes a limitation under polygenic architectures, where lasso failed to retain the many small but collectively important SNPs. Ridge regression, in contrast, is better equipped to handle polygenic architectures, as ridge retains all SNPs and shrinks their effects rather than eliminating them. This allows the ridge model to capture diffuse signals from many small-effect SNPs, but at the cost of lower specificity. In oligogenic architectures, the ridge regression distributes effect sizes across correlated SNPs, diluting the signal of truly causal loci and assigning non-zero effects to irrelevant SNPs. Consequently, ridge-estimated PVE showed lower correlation with true PVE than both lasso and SHAP measures in the highly oligogenic setting, but higher correlation than both in the polygenic setting. In summary, ridge regression tends to spread signal broadly, while lasso selects a sparse subset of SNPs. Given these trade-offs, an approach with more realistic assumptions on the distribution of effect sizes, such as a Bayesian alphabet method, might better accommodate varying genetic architecture, potentially improving the identification of important SNPs. Pasam et al. (2017) performed a similar study using BayesR in inferring SNP-associations in wheat with promising results.

In our association analysis of observed house sparrow traits, GWAS identified no significant SNPs for either of the traits tarsus length, wing length or body mass. A possible explanation for the lack of significant hits could be that identifying specific SNPs associated with the trait of interest requires very large sample sizes to reduce the uncertainty in the p-values, as was found for example in the analysis of human height (Yengo et al., 2022). We note that no significant SNPs were found even though we used the large 70k dataset, which has a more favorable sample-to-SNP ratio than the 180k dataset. Our results underline the difficulty in detecting individual SNP effects in complex traits, particularly in wild populations. Despite the lack of statistically significant hits, the GWAS results for wing length and tarsus length suggest stronger SNP-trait associations on average than for body mass. This pattern mirrors what we observed in the genomic prediction analyses, where body mass consistently exhibited the lowest prediction accuracy across models.

Both these findings suggest that body mass is less heritable than wing length and tarsus length.

We observe that GWAS and SHAP have limited overlap between SNPs identified as important. This discrepancy is expected, given the fundamental differences between p-values and mean |SHAP| values. GWAS estimates the effect of each SNP independently via univariate regression models, yielding p-values that are sensitive to both effect size and uncertainty. In contrast, SHAP values are derived from the XGBoost model, and evaluate the contribution of each SNP in the context of all others. In addition, SHAP values capture interactions and non-linear effects, which are assumed to be missed by GWAS. Interestingly, for wing length, the SNP with the highest mean |SHAP| value also ranked among the top hits in GWAS. While overall correlation between SHAP and GWAS was weak, this particular agreement is notable. Simulations showed that when GWAS and SHAP identified the same SNP as important, it often corresponded to a SNP with high true PVE. Hence, this SNP is a promising candidate for further study, as it may be in LD with a QTL.

We further examined the ability of ridge and lasso regression to infer underlying genetic architecture from model coefficients across simulated trait architectures. The distributions of ridge coefficients across all simulated architectures were highly similar and closely resembled normal distributions centered near zero. This pattern is mirrored in the ridge curves, which overlapped across all genetic architectures and followed trajectories consistent with polygenic traits. These observations are a direct consequence of the Gaussian prior on effect sizes used in ridge regression. Lasso regression, while differing in its prior assumptions, exhibited similar limitations. The lasso coefficient distributions showed sharp peaks at zero with a few large-magnitude coefficients across all architectures, which is typical for oligogenic traits. The alignment with oligogenic architectures is further supported by the lasso curves, which were overlapping across all trait architectures and closely resembled those expected for oligogenic traits. Similarly to the ridge regression case, the lasso results are a direct consequence of the Laplace prior on effect sizes. Overall, these patterns illustrate how ridge and lasso tend to reflect their prior assumptions rather than learning the true underlying architecture, consistent with "lack of Bayesian learning" dilemma (*e.g.*, Habier et al., 2011)

Our results highlight that lasso and ridge regression are poorly suited for inferring genetic architecture from coefficient estimates, even though ridge regression showed promising results in genomic prediction, and both methods identified some relevant SNPs in highly oligogenic traits. Because ridge and lasso regression cannot reliably infer genetic architecture from their coefficient estimates, alternative methods such as BayesR were developed to address this limitation. Bayesian approaches such as BayesR are more appropriate for inferring genetic architecture, as BayesR explicitly models multiple effect size classes and can better capture the true distribution of SNP effects. Previous studies using BayesR to infer genetic architecture has shown promising results (Erbe et al., 2012; Moser et al., 2015).

In contrast to the lasso and ridge methods of inferring genetic architecture, the SHAP approach showed more promising results. While the distribution of mean |SHAP| values across SNPs showed a generally similar pattern across all architectures, peaking near zero and with a few large outliers, the SHAP curves revealed a differences between architectures. The SHAP curves loosely followed the expected trends based on genetic architecture: oligogenic traits tended to produce curves that converged more quickly, while polygenic traits showed more gradual accumulation of importance across SNPs. Nevertheless, the variability among the SHAP curves remained high, and in some cases, convergence rates did not align cleanly with the known architectures. The variability among SHAP curves limits the precision of SHAP curves as a tool for architecture inference.

A possible factor in the sparse behavior of the distributions of mean |SHAP| values is the way XGBoost builds its ensemble of decision trees in a high-dimensional $m \gg n$ setting. The distributions of mean |SHAP| values suggest that XGBoost's trees, constrained by depth limits, split only on a fraction of the available SNPs. Many small-effect SNPs may therefore never appear in any tree and thus receive near-zero SHAP values. Because XGBoost grows trees by repeatedly choosing the single feature that most reduces the loss at each split, it tends to focus on the strongest signals and may ignore many weakly predictive SNPs. In highly oligogenic architectures, where only a few SNPs have large effects, this greedy approach aligns well with the true architecture. The alignment with oligogenic architecture for mean |SHAP| values is also in line with our genomic

prediction results on simulated phenotypes, where XGBoost achieved slightly higher accuracy for oligogenic traits. For future work, applying the SHAP framework to a prediction model that more accurately captures the distribution of SNP effects may improve its usefulness for inferring genetic architecture.

The development of strong predictive models in machine learning, along with the growing availability of genomic data, motivates the search for alternative approaches that could challenge traditional methods in genomic prediction. GWAS has known limitations, particularly due to reliance on p-values (Yang et al., 2010; Yengo et al., 2022), which motivates the search for alternative measures of SNP importance. Boosted regression trees and regularized linear models represent promising directions for improving predictive performance in genomic applications. Ensemble tree methods can flexibly capture non-additive effects and complex interactions, while ridge regression and dimension-reduction approaches such as principal component ridge regression remain robust and computationally efficient for large genomic datasets. However, regression methods such as ridge and lasso are less suitable for inferring SNP-trait associations, as they do not accurately capture the true distribution of marker effects and suffer from a "lack of Bayesian learning" (Habier et al., 2011). At the same time, new methods for interpreting machine learning models are emerging. In particular, using SHAP values to explain model predictions offers a novel way to assess SNP-trait associations beyond traditional significance testing, and could contribute to more transparent and informative genomic analyses. Looking ahead, integrating machine learning with biologically informed models and advanced interpretability tools, such as SHAP values, will be key to fully realizing the potential of genomic prediction and to deepening our understanding of the complexity within natural populations.

References

Aas, Kjersti, Martin Jullum and Anders Løland (2021). 'Explaining individual predictions when features are dependent: More accurate approximations to Shapley values'. In: *Artificial Intelligence* 298. ISSN: 00043702. DOI: 10.1016/j.artint.2021.103502.

Abdi, Hervé and Lynne J Williams (2010). *Principal component analysis*. DOI: 10.1002/wics.101.

Akiba, Takuya et al. (2019). 'Optuna: A Next-generation Hyperparameter Optimization Framework'. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, pp. 2623–2631. ISBN: 9781450362016. DOI: 10.1145/3292500.3330701.

Arenas, Sebastián et al. (2021). 'Evaluating the accuracy of genomic prediction for the management and conservation of relictual natural tree populations'. In: DOI: 10.1007/s11295-020-01489-1. URL: https://doi.org/10.1007/s11295-020-01489-1.

Ashraf, B et al. (2021). *Genomic prediction in the wild: A case study in Soay sheep*. English. Tech. rep. DOI: 10.1111/mec.16262.

Aspheim, Janne C. H. et al. (2024). *Bayesian marker-based principal component ridge regression – a flexible multipurpose framework for quantitative genetics in wild study systems*. DOI: 10.1101/2024.06.01.596874. URL: http://biorxiv.org/lookup/doi/10.1101/2024.06.01.596874.

Aulchenko, Yurii S (2011). *Analysis of Complex Disease Association Studies*. Ed. by Eleftheria Zeggini and Andrew Morris. Academic Press. ISBN: 978-0-12-375142-3. DOI: 10.1016/B978-0-12-375142-3.10009-4. URL: https://doi.org/10.1016/B978-0-12-375142-3.10009-4.

Bérénos, Camillo et al. (2014). 'Estimating quantitative genetic parameters in wild populations: A comparison of pedigree and genomic approaches'. In: *Molecular Ecology* 23.14, pp. 3434–3451. ISSN: 1365294X. DOI: 10.1111/mec.12827.

Berg, Jeremy J et al. (2019). 'Reduced signal for polygenic adaptation of height in UK Biobank'. In: 8. ISSN: 2050-084X. DOI: 10.7554/eLife.39725.001. URL: https://doi.org/10.7554/eLife.39725.

Bergstra, James et al. (2011). *Algorithms for Hyper-Parameter Optimization*. Tech. rep. URL: https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

Boehmke, Brad and Brandon Greenwell (2019). *Hands-On Machine Learning with R*. 1st ed. ISBN: 978-1138495685.

Bolastad, William (2009). *Understanding Computational Bayesian Statistics*. ISBN: 9780470046098. DOI: 10.1002/9780470567371.

Boubacar, Sidi et al. (2013). *Evaluation of approaches for estimating the accuracy of genomic prediction in plant breeding*. Tech. rep. URL: http://www.biomedcentral.com/1471-2164/14/860.

Bousquet, Olivier, Ulrike von Luxburg and Gunnar Rätsch (2003). *Advanced Lectures on Machine Learning*. English. Springer. ISBN: 978-3-540-28650-9. DOI: https://doi.org/10.1007/b100712. URL: https://link.springer.com/book/10.1007/b100712.

Brodie, Aharon, Johnathan Roy Azaria and Yanay Ofran (2016). 'How far from the SNP may the causative genes be?' In: *Nucleic Acids Research* 44.13, pp. 6046–6054. ISSN: 13624962. DOI: 10.1093/nar/gkw500.

Brookes, Anthony J. (1999). 'The essence of SNPs'. In: 234. DOI: https://doi.org/10.1016/S0378-1119(99)00219-X.

Bush, William S. and Jason H. Moore (2012). 'Chapter 11: Genome-Wide Association Studies'. In: *PLoS Computational Biology* 8.12. ISSN: 15537358. DOI: 10.1371/journal.pcbi.1002822.

Chen, Ding and John Chen (2017). *Monte-Carlo Simulation- Based Statistical Modeling*. Springer. ISBN: 978-981-10-3307-0. DOI: https://doi.org/10.1007/978-981-10-3307-0. URL: http://www.springer.com/series/13402.

Chen, Tianqi and Carlos Guestrin (2016). 'XGBoost: A scalable tree boosting system'. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Vol. 13-17-August-2016. Association for Computing Machinery, pp. 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf.

Coleman, Charles D (2017). *Decomposing the R-squared of a Regression Using the Shapley Value in SAS*. Tech. rep.

Configure XGBoost (2024). Date accessed: 29/05/2025. URL: https://xgboosting.com/configure-xgboost-regabsoluteerror-objective-mean-absolute-error/.

De Vlaming, Ronald and Patrick J.F. Groenen (2015). *The current and future use of ridge regression for prediction in quantitative genetics*. DOI: 10.1155/2015/143712.

Echagüe, Eduardo Vila and Ari Belenkiy (2008). 'Groping Toward Linear Regression Analysis: Newton's Analysis of Hipparchus' Equinox Observations'. In: DOI: https://doi.org/10.48550/arXiv.0810.4948.

Eichler, Evan E. et al. (2010). *Missing heritability and strategies for finding the underlying causes of complex disease*. DOI: 10.1038/nrg2809.

Ekanayake, I. U., D. P.P. Meddage and Upaka Rathnayake (2022). 'A novel approach to explain the black-box nature of machine learning in compressive strength predictions of concrete using Shapley additive explanations (SHAP)'. In: *Case Studies in Construction Materials*. ISSN: 22145095. DOI: 10.1016/j.cscm.2022.e01059.

Erbe, M. et al. (2012). 'Improving accuracy of genomic predictions within and between dairy cattle breeds with imputed high-density single nucleotide polymorphism panels'. In: *Journal of Dairy Science* 7. ISSN: 00220302. DOI: 10.3168/jds.2011-5019.

Frazier, Peter I. (2018). 'A Tutorial on Bayesian Optimization'. In: URL: http://arxiv.org/abs/1807.02811.

Friedman, Jerome, Trevor Hastie and Rob Tibshirani (2010). *Regularization Paths for Generalized Linear Models via Coordinate Descent*. Tech. rep. URL: http://www.jstatsoft.org/.

Gianola, Daniel (2013). 'Priors in whole-genome regression: The Bayesian alphabet returns'. In: *Genetics*. ISSN: 00166731. DOI: 10.1534/genetics.113.151753.

Gill, Mitchell et al. (2022). 'Machine learning models outperform deep learning models, provide interpretation and facilitate feature selection for soybean trait prediction'. In: *BMC Plant Biology* 22.1. ISSN: 14712229. DOI: 10.1186/s12870-022-03559-z.

Goodman, Steven (2008). 'A Dirty Dozen: Twelve P-Value Misconceptions'. In: *Seminars in Hematology* 45.3, pp. 135–140. ISSN: 00371963. DOI: 10.1053/j.seminhematol.2008.04.003.

Gravdal, Gard W (2024). *Genomic prediction in wild populations with interpretable machine learning*. Tech. rep.

Grömping, Ulrike (2006). *Relative Importance for Linear Regression in R: The Package relaimpo*. Tech. rep. URL: http://www.jstatsoft.org/.

Habier, David et al. (2011). 'Extension of the bayesian alphabet for genomic selection'. In: *BMC Bioinformatics*. ISSN: 14712105. DOI: 10.1186/1471-2105-12-186.

Hartl, Daniel L. and Jeffrey K. Conner (2004). *A Primer of Ecological Genetics*. English. Oxford University Press Inc. ISBN: 978-0878932023.

Hastie, Trevor, Robert Tibshirani and Jerome Friedman (2009). *The Elements of Statistical Learning*. Springer New York, NY. ISBN: 978-0-387-84857-0. DOI: https://doi.org/10.1007/978-0-387-84858-7.

Head, Megan L. et al. (2015). 'The Extent and Consequences of P-Hacking in Science'. In: *PLoS Biology* 13.3. ISSN: 15457885. DOI: 10.1371/journal.pbio.1002106.

Henderson, C R (1975). *Best Linear Unbiased Estimation and Prediction under a Selection Model*. Tech. rep. 2, pp. 423–447. URL: https://www.jstor.org/stable/2529430.

— (1988). 'Applications of Linear Models in Animal Breeding'. In: *Journal of Dairy Science*. URL: https://learnanimalbreeding.com/files/Henderson_1984.pdf.

Hickey, John M. et al. (2017). *Genomic prediction unifies animal and plant breeding programs to form platforms for biological discovery*. DOI: 10.1038/ng.3920.

Hoerl, Arthur E and Robert W Kennard (1970). *Ridge Regression: Biased Estimation for Nonorthogonal Problems*. Tech. rep. 1. DOI: doi.org/10.1080/00401706.1970.10488634.

Horn, Roger A and Charles R Johnson (2014). *Matrix Analysis*. Cambridge University Press. ISBN: 9781139785884.

Huang, Qingyang (2015). *Genetic Study of Complex Diseases in the Post-GWAS Era*. DOI: 10.1016/j.jgg.2015.02.001.

Hunter, D. C. et al. (2022). 'Using genomic prediction to detect microevolutionary change of a quantitative trait'. In: *Proceedings of the Royal Society B: Biological Sciences* 289.1974. ISSN: 14712954. DOI: 10.1098/rspb.2022.0330.

Ioannidis, John P.A. (2018). 'Why most published research findings are false'. In: *Getting to Good: Research Integrity in the Biomedical Sciences*, pp. 2–8. ISSN: 15491277. DOI: 10.1371/journal.pmed.0020124.

James, Gareth et al. (2013). *An Introduction to Statistical Learning*. 1st ed. Springer. ISBN: 978-1461471370.

Jensen, Henrik, Rune Moe et al. (2013). 'Genetic variation and structure of house sparrow populations: Is there an island effect?' In: *Molecular Ecology* 22.7, pp. 1792–1805. ISSN: 09621083. DOI: 10.1111/mec.12226.

Jensen, Henrik, Ingelin Steinsland et al. (2008). 'Evolutionary dynamics of a sexual ornament in the house sparrow (Passer domesticus): The role of indirect selection within and between sexes'. In: *Evolution* 62.6, pp. 1275–1293. ISSN: 00143820. DOI: 10.1111/j.1558-5646.2008.00395.x.

Johnsen, Pål V. et al. (2021). 'A new method for exploring gene–gene and gene–environment interactions in GWAS with tree ensemble methods and SHAP values'. In: *BMC Bioinformatics* 22.1. ISSN: 14712105. DOI: 10.1186/s12859-021-04041-7.

Karki, Roshan et al. (2015). *Defining "mutation" and "polymorphism" in the era of personal genomics*. DOI: 10.1186/s12920-015-0115-z.

Kruuk, Loeske E.B. (2004). *Estimating genetic parameters in natural populations using the 'animal model'*. DOI: 10.1098/rstb.2003.1437.

Kuo, Frances Y and Ian H Sloan (2005). *Lifting the Curse of Dimensionality*. Tech. rep. URL: http://www.yaroslavvb.com/papers/kuo-lifting.pdf.

Kyrgiafini, Maria Anna et al. (2023). 'Gene-by-Sex Interactions: Genome-Wide Association Study Reveals Five SNPs Associated with Obesity and Overweight in a Male Population'. In: *Genes* 14.4. ISSN: 20734425. DOI: 10.3390/genes14040799.

Lazarevic, Aleksandar and Vipin Kumar (2005). *Feature Bagging for Outlier Detection*. Tech. rep. DOI: https://doi.org/10.1145/1081870.1081891. URL: https://dl.acm.org/doi/pdf/10.1145/1081870.1081891.

Lê, Sébastien et al. (2008). *FactoMineR: An R Package for Multivariate Analysis*. Tech. rep. DOI: 10.18637/jss.v025.i01. URL: http://www.jstatsoft.org/.

Lindeman, Richard H et al. (1980). *Introduction to Bivariate and Multivariate Analysis*. Tech. rep. 375.

Lourenço, Vanda M et al. (2024). 'Genomic prediction using machine learning: a comparison of the performance of regularized regression, ensemble, instance-based and deep learning methods on synthetic and empirical data'. In: *BMC Genomics* 25.1. ISSN: 14712164. DOI: 10.1186/s12864-023-09933-x.

Lundberg, Scott M., Gabriel G. Erion and Su-In Lee (2018). 'Consistent Individualized Feature Attribution for Tree Ensembles'. In: DOI: https://doi.org/10.48550/arXiv.1802.03888. URL: http://arxiv.org/abs/1802.03888.

Marioni, Riccardo E. et al. (2018). 'GWAS on family history of Alzheimer's disease'. In: *Translational Psychiatry* 8.1. ISSN: 21583188. DOI: 10.1038/s41398-018-0150-6.

Martino, Sara and Andrea Riebler (2019). 'Integrated Nested Laplace Approximations (INLA)'. In: URL: http://arxiv.org/abs/1907.01248.

McGaugh, Suzanne E., Aaron J. Lorenz and Lex E. Flagel (2021). *The utility of genomic prediction models in evolutionary genetics*. DOI: 10.1098/rspb.2021.0693.

McKinney, B. A. and Nicholas M. Pajewski (2012). *Six degrees of epistasis: Statistical network models for GWAS*. DOI: 10.3389/fgene.2011.00109.

Meher, Prabina Kumar, Sachin Rustgi and Anuj Kumar (2022). 'Performance of Bayesian and BLUP alphabets for genomic prediction: analysis, comparison and results'. In: *Heredity*. ISSN: 13652540. DOI: 10.1038/s41437-022-00539-9.

Meuwissen, T H E, B J Hayes and M E Goddard (2001). *Prediction of Total Genetic Value Using Genome-Wide Dense Marker Maps*. Tech. rep. DOI: 10.1093/genetics/157.4.1819. URL: https://doi.org/10.1093/genetics/157.4.1819.

Misztal, Ignacy, Daniela Lourenco and Andres Legarra (2020). 'Current status of genomic evaluation'. In: *Journal of Animal Science* 98.4. ISSN: 15253163. DOI: 10.1093/jas/skaa101.

Molnar, Christoph (2022). *A Guide for Making Black Box Models Explainable*. 2nd ed. URL: https://christophm.github.io/interpretable-ml-book.

Moser, Gerhard et al. (2015). 'Simultaneous Discovery, Estimation and Prediction Analysis of Complex Traits Using a Bayesian Mixture Model'. In: *PLoS Genetics*. ISSN: 15537404. DOI: 10.1371/journal.pgen.1004969.

Muff, Stefanie, Erlend B. Nilsen et al. (2022). *Rewriting results sections in the language of evidence*. DOI: 10.1016/j.tree.2021.10.009.

Muff, Stefanie, Alina K. Niskanen et al. (2019). 'Animal models with group-specific additive genetic variances: Extending genetic group models'. In: *Genetics Selection Evolution* 51.1. ISSN: 12979686. DOI: 10.1186/s12711-019-0449-7.

Murcray, Cassandra E., Juan Pablo Lewinger and W. James Gauderman (2009). 'Gene-environment interaction in genome-wide association studies'. In: *American Journal of Epidemiology* 169.2, pp. 219–226. ISSN: 00029262. DOI: 10.1093/aje/kwn353.

Nasteski, Vladimir (2017). 'An overview of the supervised machine learning methods'. In: *HORIZONS.B* 4, pp. 51–62. ISSN: 18578578. DOI: 10.20544/horizons.b.04.1.17.p05.

Ødegård, Jørgen et al. (2018). 'Large-scale genomic prediction using singular value decomposition of the genotype matrix'. In: *Genetics Selection Evolution*. ISSN: 12979686. DOI: 10.1186/s12711-018-0373-2.

Oraguzie, Nnadozie C et al. (2008). *Association Mapping in Plants*. Springer. ISBN: 978-0387514925.

Pack Kaelbling, Leslie et al. (1996). *Reinforcement Learning: A Survey*. Tech. rep., pp. 237–285. DOI: https://doi.org/10.1613/jair.301. URL: https://www.jair.org/index.php/jair/article/view/10166/24110.

Palaniyappan, S. et al. (2024). 'Principal component analysis (PCA) as a genetic diversity tool to understand the variation of rice mutant culture'. In: *Electronic Journal of Plant Breeding*. ISSN: 0975928X. DOI: 10.37992/2024.1504.113.

Pärn, Henrik et al. (2011). 'Spatial heterogeneity in the effects of climate and density-dependence on dispersal in a house sparrow metapopulation'. In: *Proceedings of the Royal Society B: Biological Sciences* 279.1726, pp. 144–152. ISSN: 14712970. DOI: 10.1098/rspb.2011.0673.

Pasam, Raj K. et al. (2017). 'Detection and validation of genomic regions associated with resistance to rust diseases in a worldwide hexaploid wheat landrace collection using BayesR and mixed linear model approaches'. In: *Theoretical and Applied Genetics*. ISSN: 00405752. DOI: 10.1007/s00122-016-2851-7.

Pedregal, Pablo (2004). *Texts in Applied Mathematics*. Springer New York, NY. DOI: https://doi.org/10.1007/b97412.

Pu, Fang, Jinsong Ren and Xiaogang Qu (2018). *Nucleobases, nucleosides, and nucleotides: Versatile biomolecules for generating functional nanomaterials*. DOI: 10.1039/c7cs00673j.

Rue, Håvard, Sara Martino and Nicolas Chopin (2009). 'Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations'. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 71.2, pp. 319–392. DOI: 10.1111/j.1467-9868.2008.00700.x.

Saatoglu, Dilan et al. (2021). 'Dispersal in a house sparrow metapopulation: An integrative case study of genetic assignment calibrated with ecological data and pedigree information'. In: *Molecular Ecology* 30.19, pp. 4740–4756. ISSN: 1365294X. DOI: 10.1111/mec.16083.

Sachidanandam, Ravi et al. (2001). *A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms*. Tech. rep. DOI: https://doi.org/10.1038/35057149.

Shapley, L. S. (1952). *A VALUE FOR n-PERSON GAMES*. Tech. rep. URL: https://www.rand.org/content/dam/rand/pubs/papers/2021/P295.pdf.

Singsaas, Hansen (2024). *Neural networks for genomic prediction in the wild*. Tech. rep.

Sutton, Walter S (1903). *THE CHROMOSOMES IN HEREDITY*. Tech. rep. URL: https://www.journals.uchicago.edu/doi/pdf/10.2307/1535741.

Syvänen, Ann-Christine (2001). 'Accessing genetic variation: genotyping single nucleotide polymorphisms'. In: DOI: https://doi.org/10.1038/35103535.

Tibshirani, Robert (1996). *Regression Shrinkage and Selection via the Lasso*. Tech. rep. 1. DOI: doi.org/10.1111/j.2517-6161.1996.tb02080.x.

Tibshirani, Ryan J. (2013). 'The lasso problem and uniqueness'. In: *Electronic Journal of Statistics*. ISSN: 19357524. DOI: 10.1214/13-EJS815.

Uffelmann, Emil et al. (2021). *Genome-wide association studies*. DOI: 10.1038/s43586-021-00056-9.

Van De Geer, Sara A. and Hans C. Van Houwelingen (2004). *High-dimensional data: p¿¿n in mathematical statistics and bio-medical applications*. Tech. rep. DOI: 10.3150/bj/1106314843.

VanRaden, P. M. (2008). 'Efficient methods to compute genomic predictions'. In: *Journal of Dairy Science* 91.11, pp. 4414–4423. ISSN: 15253198. DOI: 10.3168/jds.2007-0980.

Visscher, Peter M. et al. (2012). *Five years of GWAS discovery*. DOI: 10.1016/j.ajhg.2011.11.029.

Walsh, Bruce and Michael Lynch (2018). *Evolution and Selection of Quantitative Traits*. Oxford University Press. ISBN: 978-0198830870.

Watanabe, Shuhei (2023). 'Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance'. In: URL: http://arxiv.org/abs/2304.11127.

White, Désirée and Montserrat Rabago-Smith (2011). *Genotype-phenotype associations and human eye color*. DOI: 10.1038/jhg.2010.126.

Whittaker, John C, Robin Thompson and Mike C Denham (2000). 'Marker-assisted selection using ridge regression'. In: *Genetical Research*. DOI: 10.1017/S0016672399004462.

Wilson, Alastair J. et al. (2010). 'An ecologist's guide to the animal model'. In: *Journal of Animal Ecology* 79.1, pp. 13–26. ISSN: 00218790. DOI: 10.1111/j.1365-2656.2009.01639.x.

Wood, Andrew R. et al. (2014). 'Defining the role of common variation in the genomic and biological architecture of adult human height'. In: *Nature Genetics* 46.11, pp. 1173–1186. ISSN: 15461718. DOI: 10.1038/ng.3097.

XGBoost Parameter (2022). Date accessed: 29/05/2025. URL: https://xgboost.readthedocs.io/en/stable/parameter.html.

Yang, Jian et al. (2010). 'Common SNPs explain a large proportion of the heritability for human height'. In: *Nature Genetics* 42.7, pp. 565–569. ISSN: 10614036. DOI: 10.1038/ng.608.

Yengo, Loïc et al. (2022). 'A saturated map of common genetic variants associated with human height'. In: *Nature* 610.7933, pp. 704–712. ISSN: 0028-0836. DOI: 10.1038/s41586-022-05275-y.

Zelioli, Chiara (2023). *Bayesian Modelling for Genomic Prediction in Quantitative Genetics: a Comparative Study across Genetic Architectures*. Tech. rep.

Zhou, Xiang and Matthew Stephens (2012). *Software tool and univariate linear mixed models*. Tech. rep. URL: https://github.com/genetics-statistics/GEMMA/blob/master/doc/manual.pdf.

Zhou, Zhi Hua (2021). *Machine Learning*. Springer Nature, pp. 1–458. ISBN: 9789811519673. DOI: 10.1007/978-981-15-1967-3.

A Additional figures

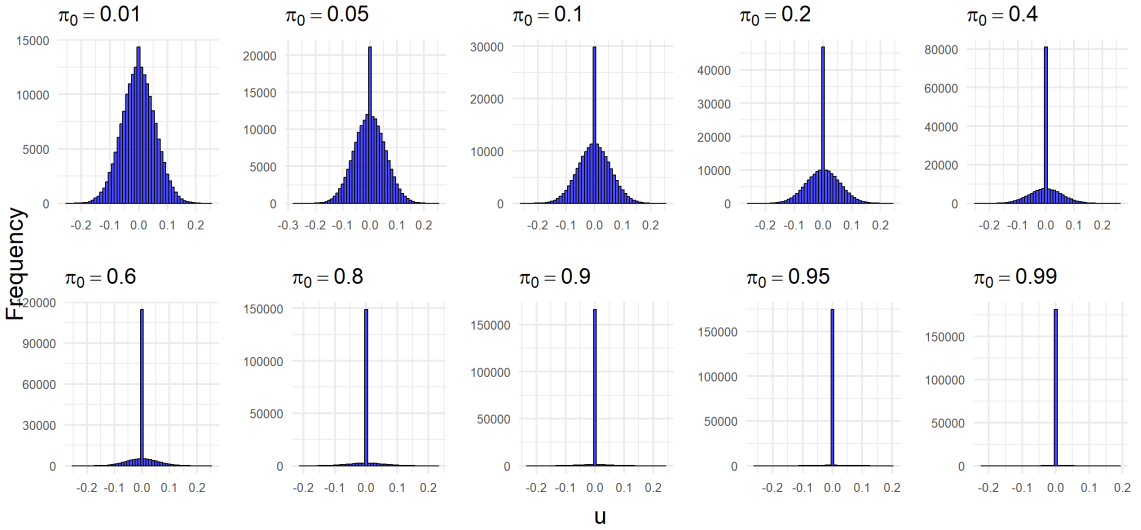
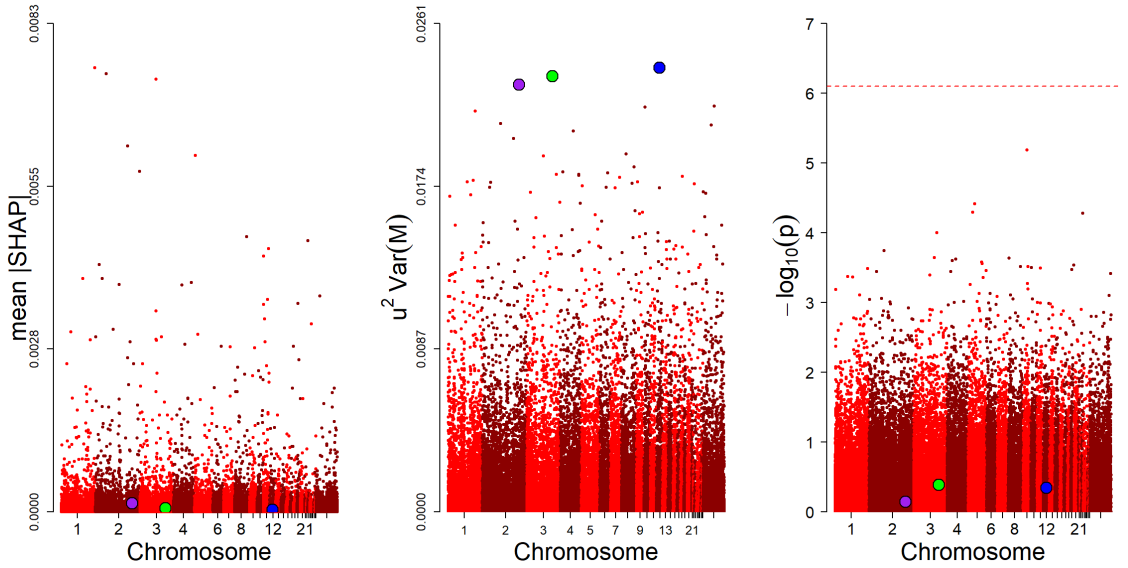
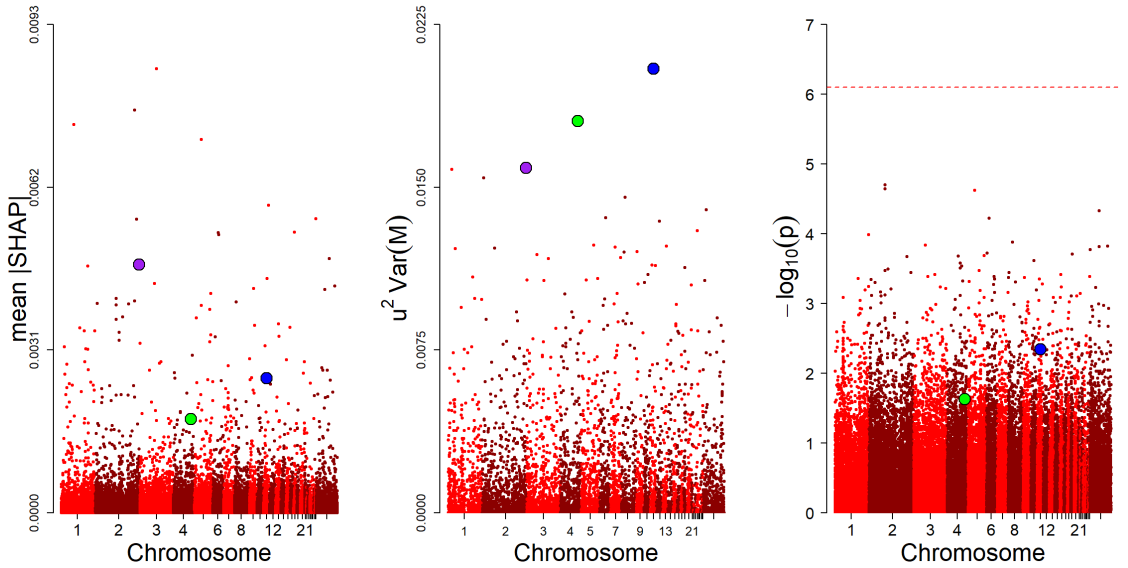


Figure 24: Distribution of SNP effect sizes ($\boldsymbol{u}$) from simulated traits on the 180k dataset across all ten architectures.

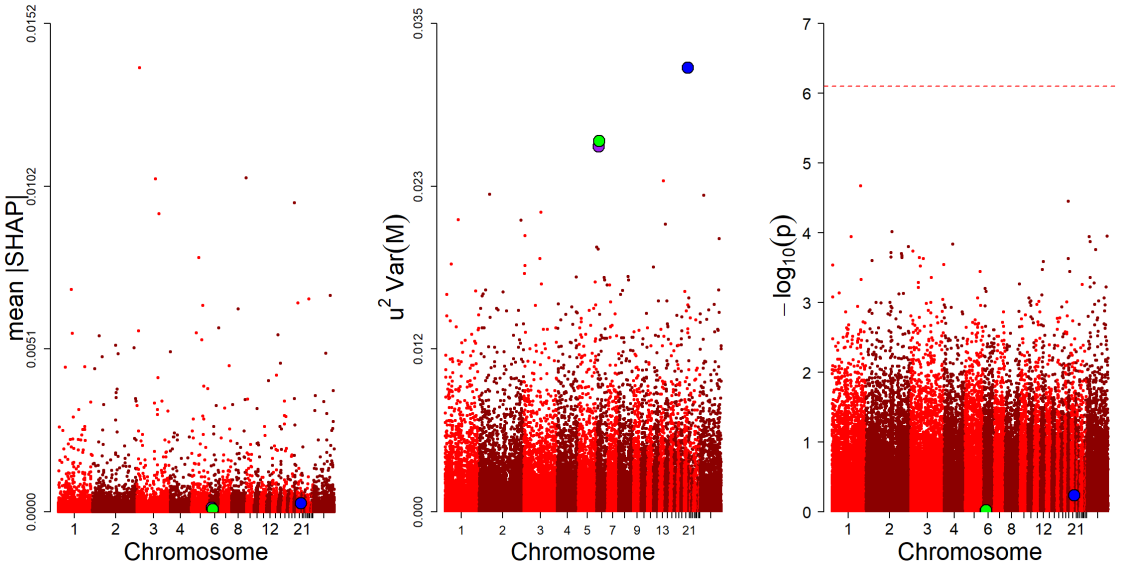


(a) $\pi_0 = 0.1$

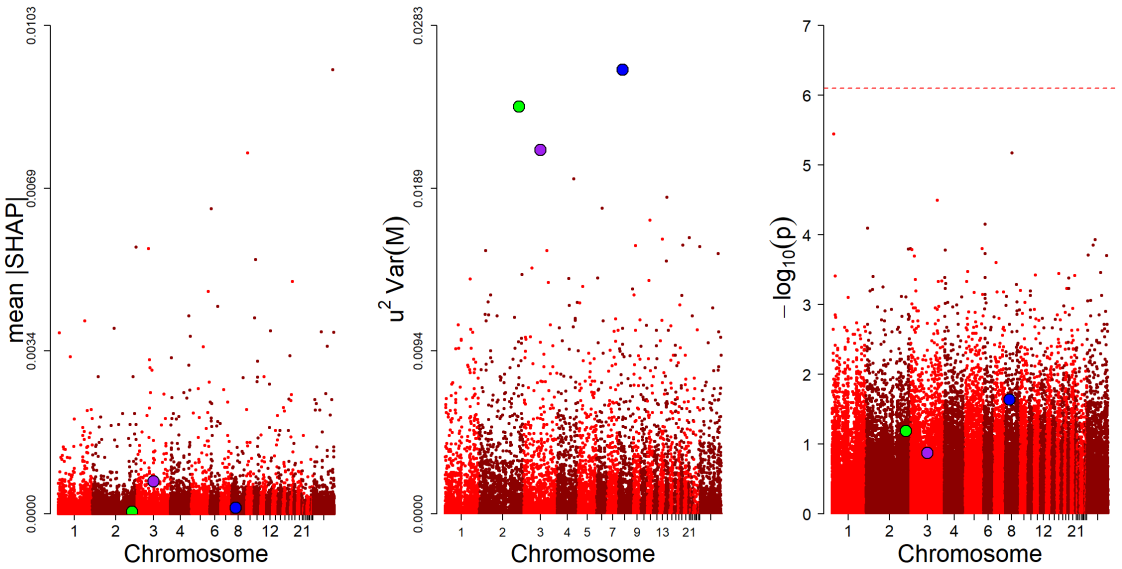


(b) $\pi_0 = 0.9$

Figure 25: Manhattan plots of mean |SHAP| values (left), PVE (middle), and p-values from GWAS (right) from simulated phenotypes on architectures $\pi_0 = 0.1,\ 0.9$. Analysis performed on the 70k dataset. The three SNPs with highest phenotypic variance contributions are highlighted.

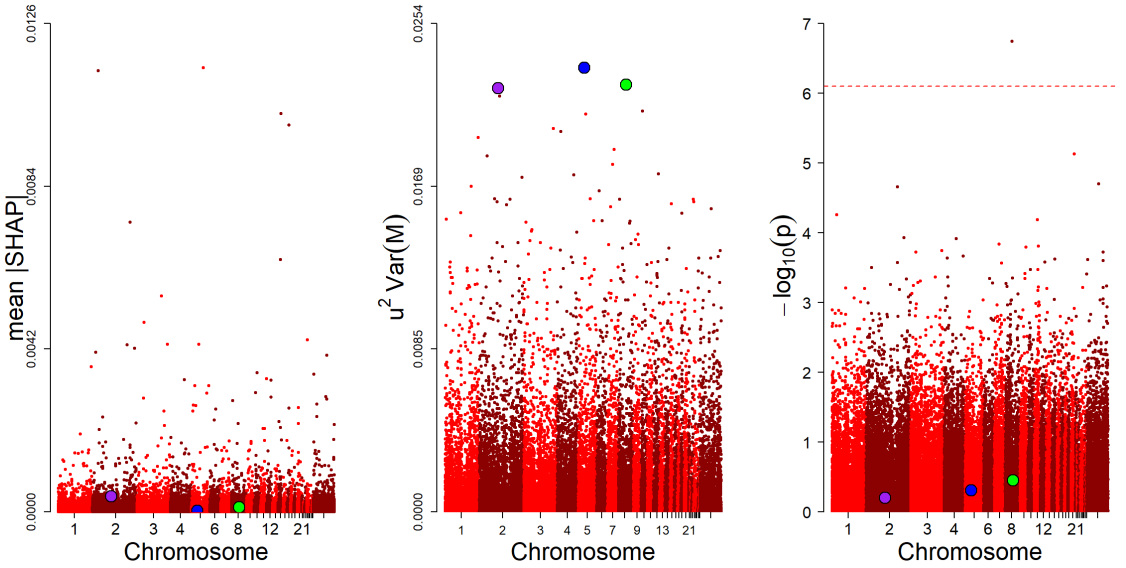


(a) $\pi_0 = 0.2$

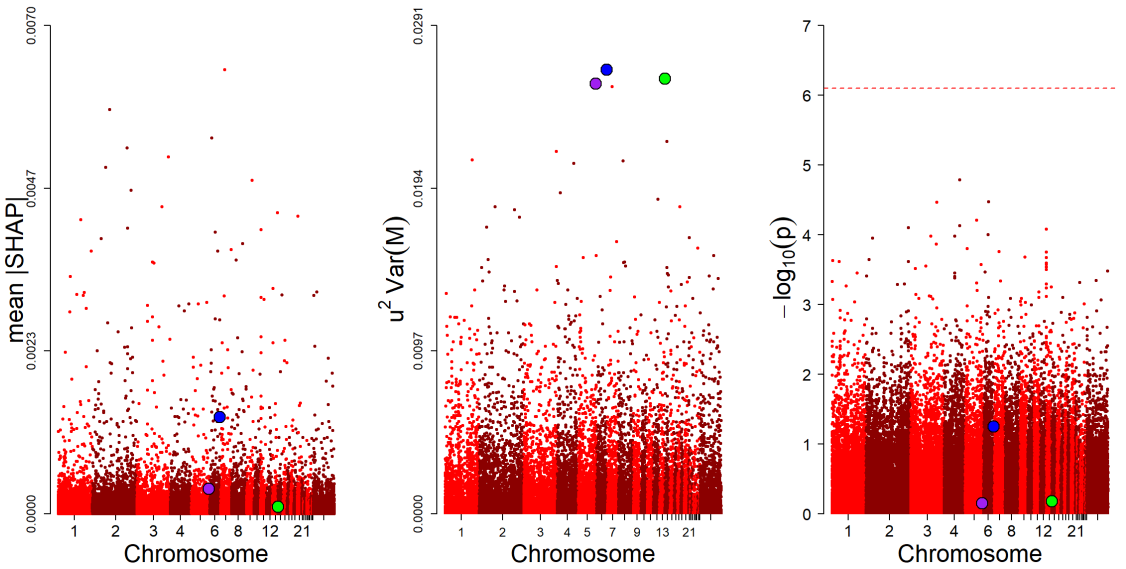


(b) $\pi_0 = 0.8$

Figure 26: Manhattan plots of mean |SHAP| values (left), PVE (middle), and p-values from GWAS (right) for simulated traits on architectures $\pi_0 = 0.2,\ 0.8$. Analysis performed on the 70k dataset. The three SNPs with highest phenotypic variance contributions are highlighted.



(a) $\pi_0 = 0.4$



(b) $\pi_0 = 0.6$

Figure 27: Manhattan plots of mean |SHAP| values (left), PVE (middle), and p-values from GWAS (right) for simulated traits on architectures $\pi_0 = 0.4,\ 0.6$. Analysis performed on the 70k dataset. The three SNPs with highest phenotypic variance contributions are highlighted.

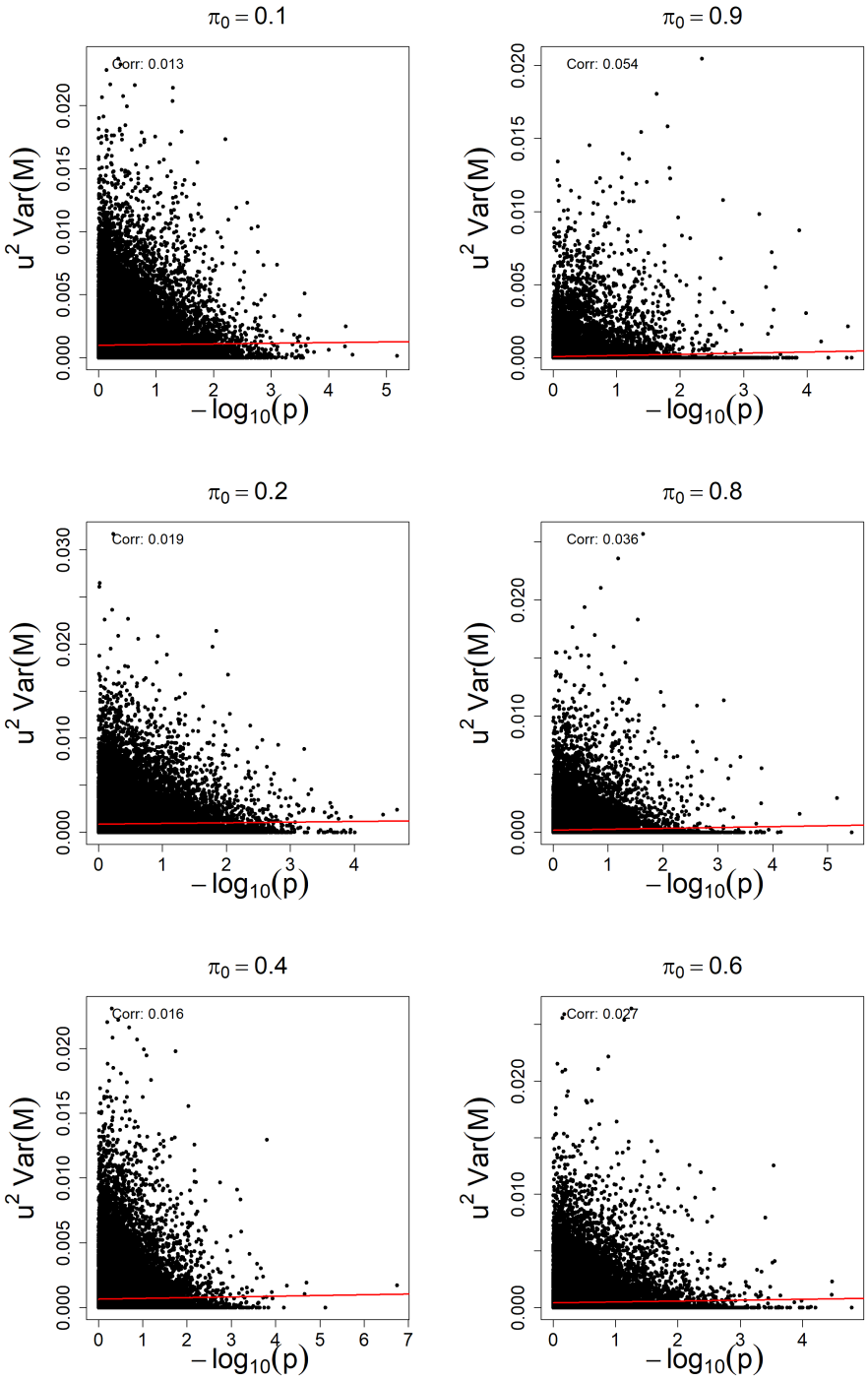


Figure 28: Correlation between p-values from GWAS and PVE of simulated phenotypes, where phenotypes were simulated on the architectures $\pi_0 = 0.1,\ 0.9,\ 0.2,\ 0.8,\ 0,4,\ 0.6$. Analysis performed on the 70k dataset.

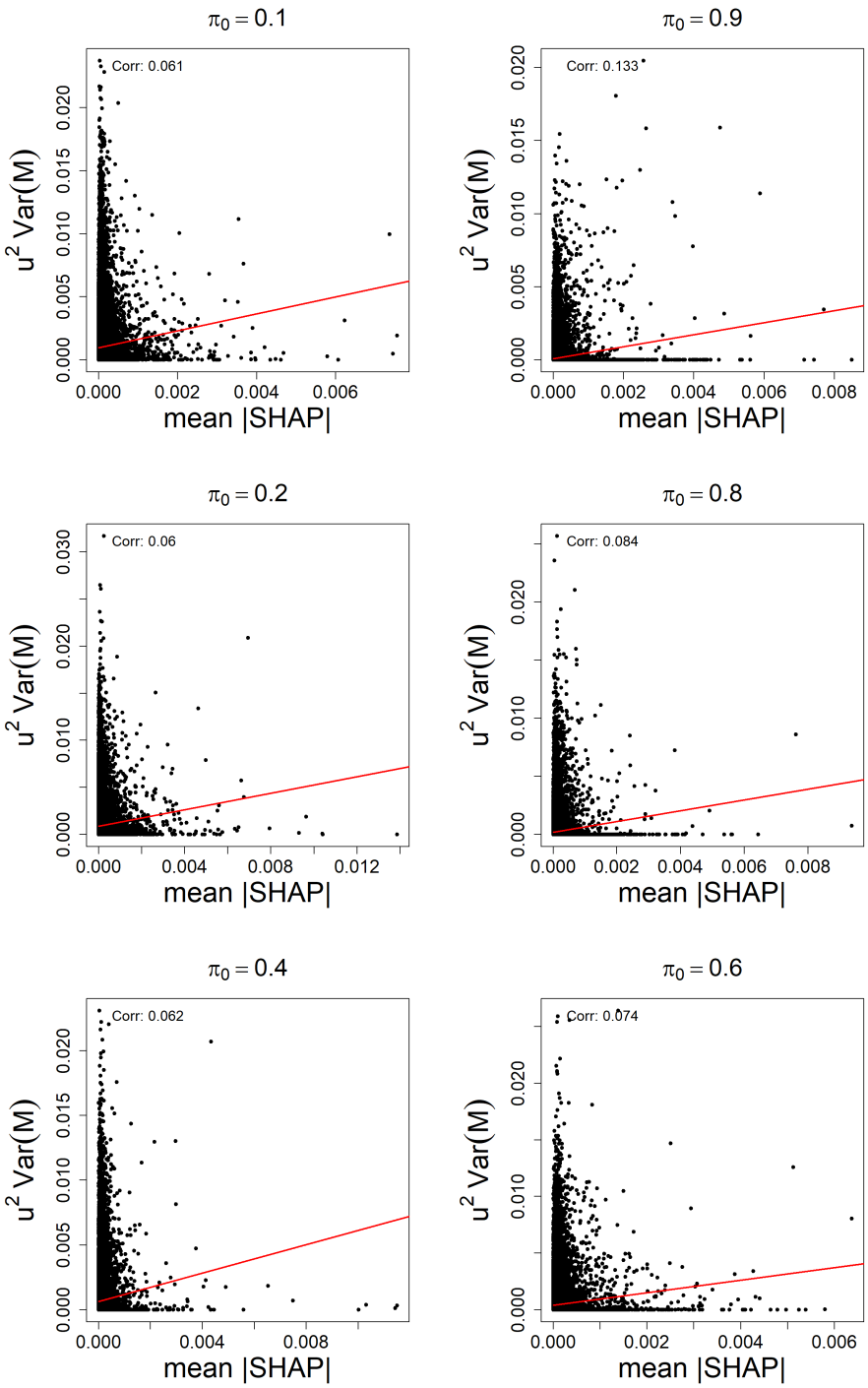
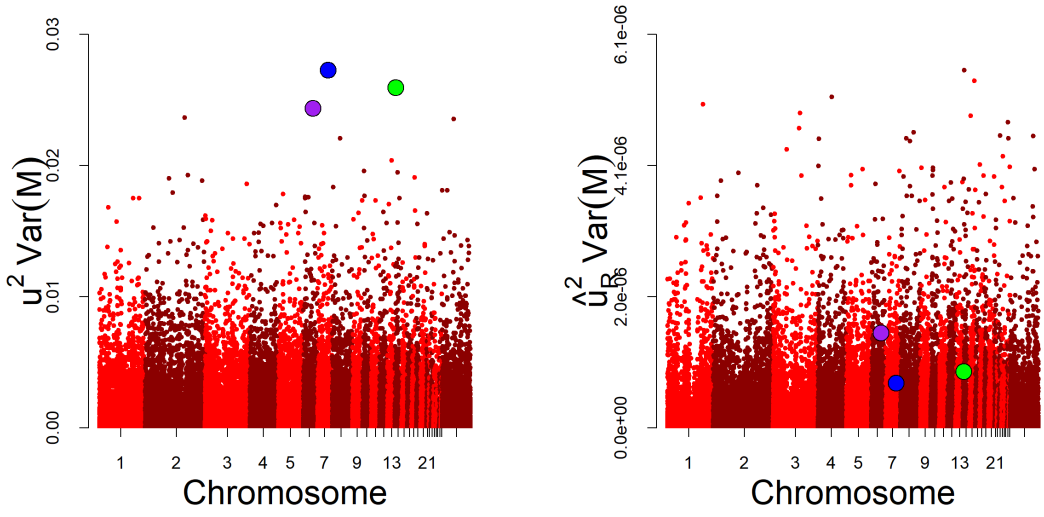
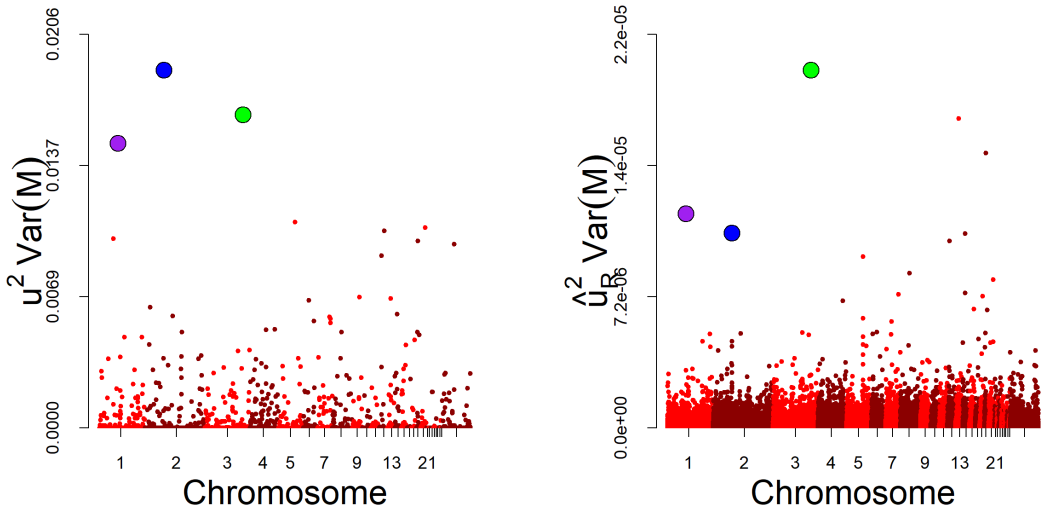


Figure 29: Correlation between mean |SHAP| values from XGBoost predictions and PVE by SNPs on simulated phenotypes, where phenotypes are simulated on the architectures $\pi_0 = 0.1,\ 0.9,\ 0.2,\ 0.8,\ 0,4,\ 0.6$. Analysis performed on the 70k dataset.

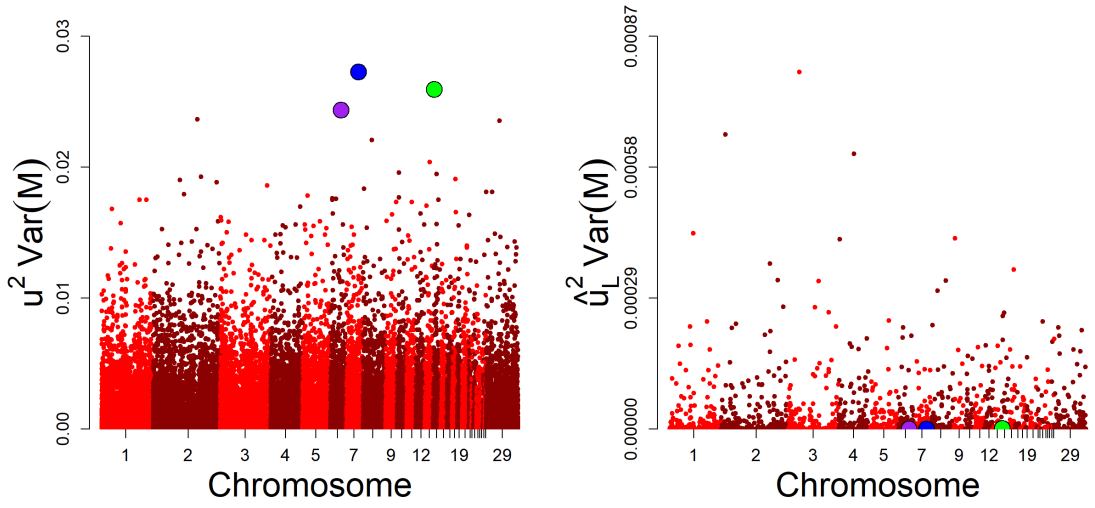


(a) $\pi_0 = 0.01$

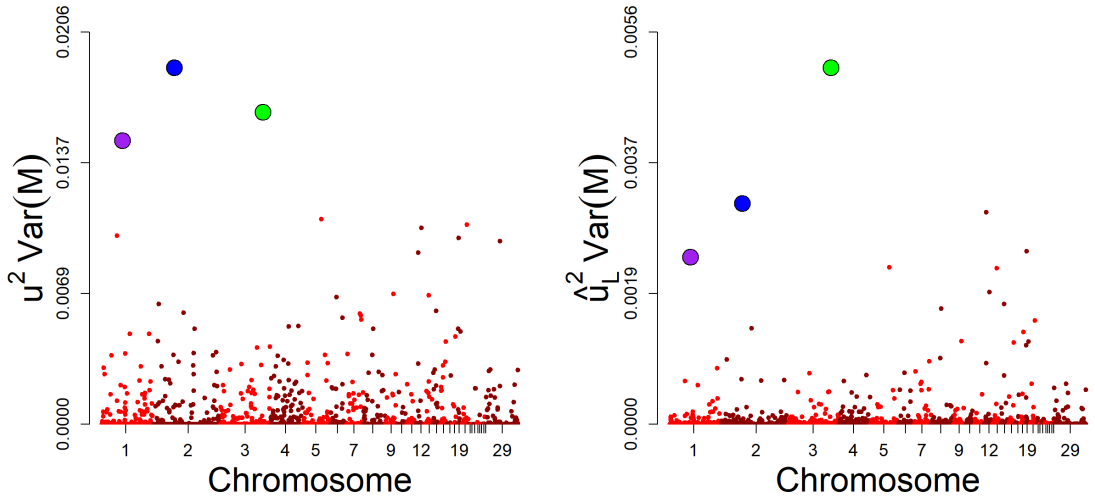


(b) $\pi_0 = 0.99$

Figure 32: Manhattan plots of ridge estimated PVE and true PVE on simulated phenotypes on architectures $\pi_0 = 0.01,\ 0.99$. Analysis performed on the 70k dataset. The three SNPs with the highest true PVE are highlighted.

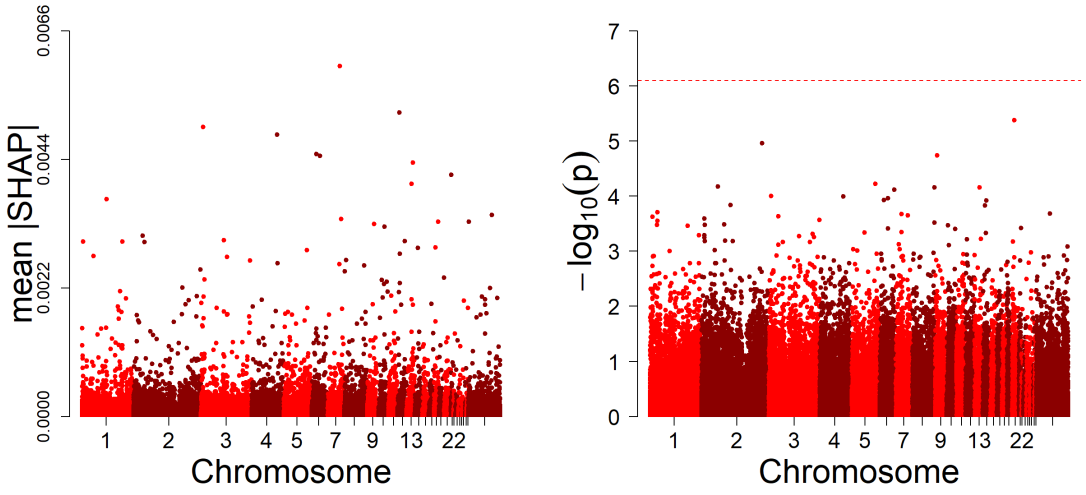


(a) $\pi_0 = 0.01$

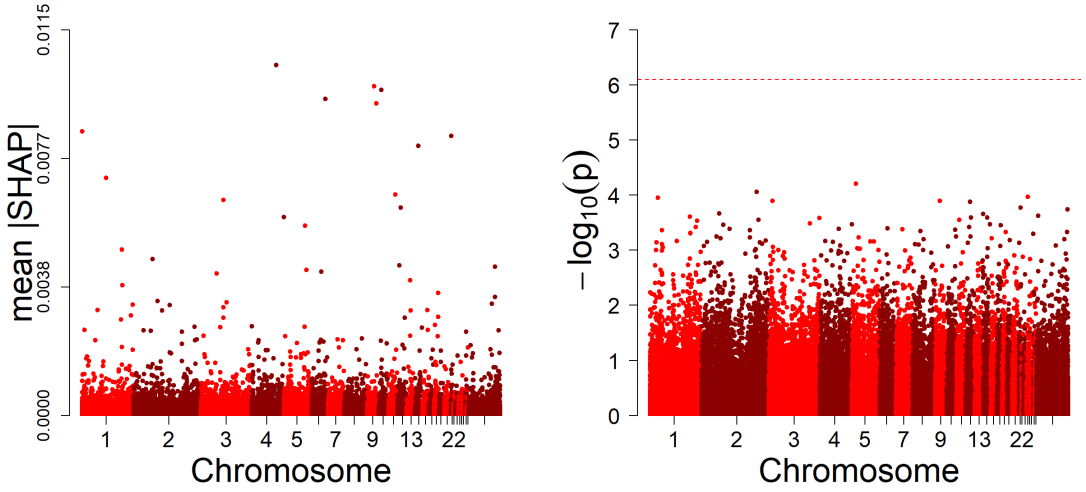


(b) $\pi_0 = 0.99$

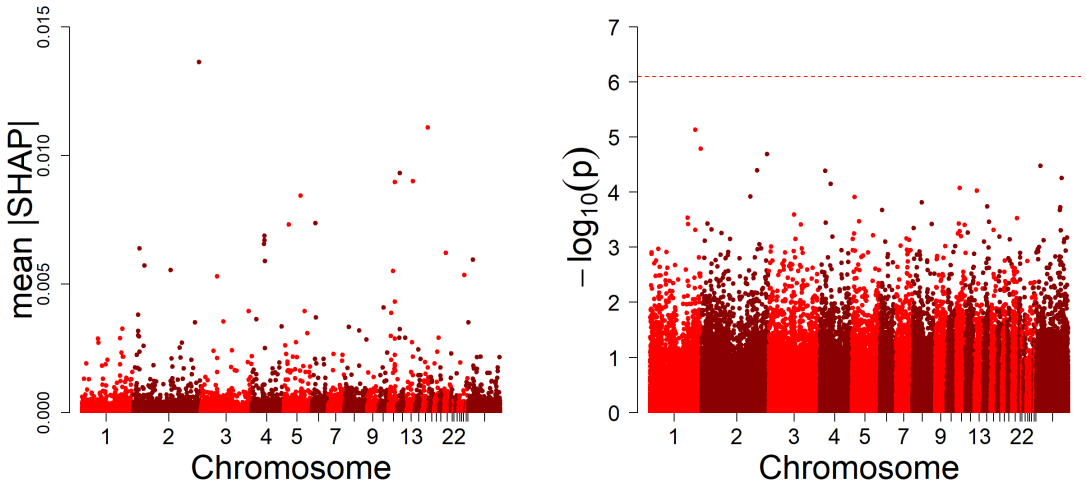
Figure 34: Manhattan plots of lasso estimated PVE and true PVE on simulated phenotypes on architectures $\pi_0 = 0.01,\ 0.99$. Analysis performed on the 70k dataset. The three SNPs with the highest true PVE are highlighted.



(a) Tarsus



(b) Mass



(c) Wing

Figure 30: Manhattan plots of mean |SHAP| values and p-values from GWAS for the three observed traits body mass (Mass), tarsus length (Tarsus) and wing length (Wing). Analysis performed on the 70k dataset.

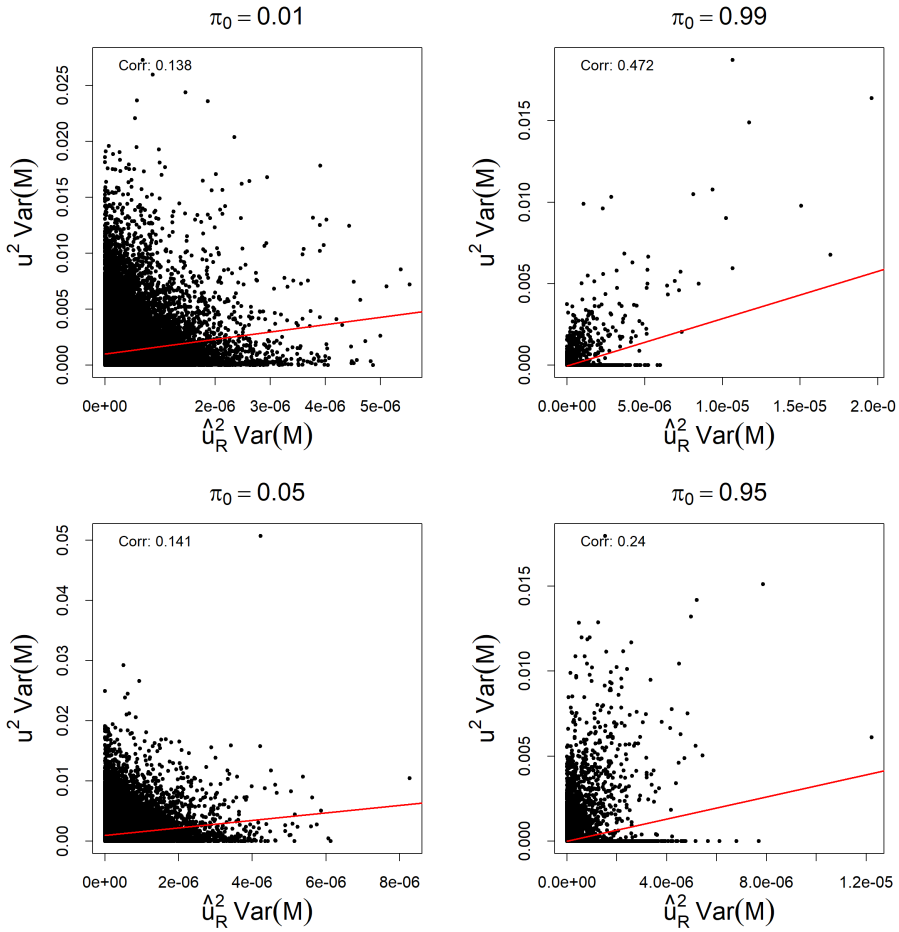


Figure 31: Correlation between ridge estimated PVE and true PVE on simulated phenotypes, where phenotypes are simulated on the architectures $\pi_0 = 0.01,\ 0.99,\ 0.05,\ 0.95$. Analysis performed on the 70k dataset.

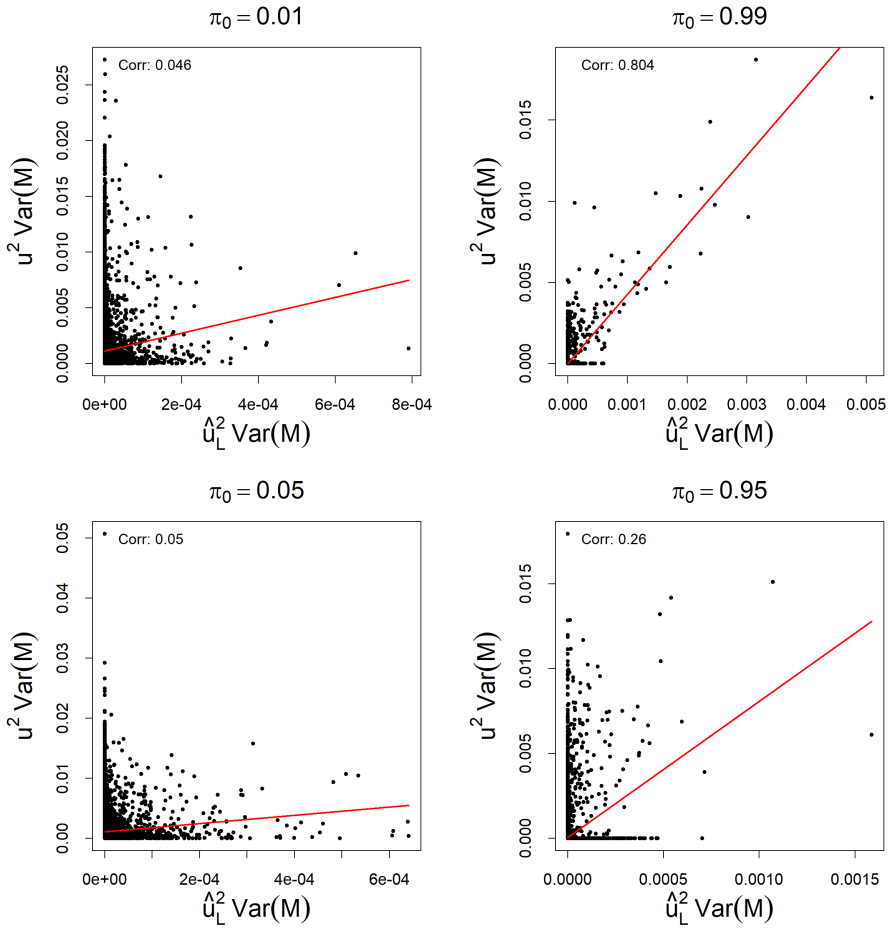


Figure 33: Correlation between lasso estimated PVE and true PVE on simulated phenotypes, where phenotypes are simulated on the architectures $\pi_0 = 0.01,\ 0.99,\ 0.05,\ 0.95$. Analysis performed on the 70k dataset.

B Proofs

For a centered matrix $\mathbf{X}_c \in \mathbb{R}^{n \times m}$ and orthonormal matrix $\mathbf{V} = \{\boldsymbol{v}_j\}_{j=1}^m \in \mathbb{R}^{m \times m}$, $\|\mathbf{v}_j\|_2 = 1,\ j = 1, ..., m$, the vectors $\boldsymbol{v}_j$ define the directions of maximum variance in $\mathbf{X}_c$ under the constraint that each vector $\boldsymbol{v}_j$ is orthogonal to all the previous vectors $\boldsymbol{v}_i\ ,\ i < j$, if and only if $\mathbf{V}$ is the matrix of normal eigenvectors of the covariance matrix $\mathbf{C}_x := \text{Cov}(\mathbf{X}_c)$. Additionally, the variance of the projection vector $\mathbf{X}_c\boldsymbol{v}_j$ is equal to the eigenvalue λ_j corresponding to eigenvector $\boldsymbol{v}_j$.

Proof. First, we want to show that $\boldsymbol{v}_1$ points in the direction of maximum variance in $\mathbf{X}_c$ when $\boldsymbol{v}_1$ is an eigenvector of $\mathbf{C}_x$. We want to maximize the variance of the projection $\mathbf{X}_c\boldsymbol{v}_1$. For a general $\boldsymbol{v}_j$, the variance of the projection $\mathbf{X}_c\boldsymbol{v}_j$ is defined as

$$\begin{aligned}\sigma_j^2 := \text{Var}(\mathbf{X}_c\boldsymbol{v}_j) &= \frac{1}{n-1}(\mathbf{X}_c\boldsymbol{v}_j)^T(\mathbf{X}_c\boldsymbol{v}_j) \\ &= \boldsymbol{v}_j^T\frac{\mathbf{X}_c^T\mathbf{X}_c}{n-1}\boldsymbol{v}_j = \boldsymbol{v}_j^T\mathbf{C}_x\boldsymbol{v}_j\ .\end{aligned}$$

Since the vector $\boldsymbol{v}_1$ is a unit vector, we want to maximize σ_1^2 subject to the constraint $\|\boldsymbol{v}_1\|_2^2 = \boldsymbol{v}_1^T\boldsymbol{v}_1 = 1$, which leads to the following optimization problem

$$\max \sigma_1^2 \quad \text{subject to } \boldsymbol{v}_1^T\boldsymbol{v}_1 - 1 = 0\ .$$

This optimization problem can be solved using a Lagrangian multiplier (Pedregal, 2004). The method can be summarized as follows. The function $f(x)$ subject to equality constraint $g(x) = 0$, can be maximized by finding the stationary points of the Lagrangian multiplier, defined as

$$\mathcal{L}(x, \lambda_\mathcal{L}) = f(x) + \lambda_\mathcal{L} g(x),\ \lambda_\mathcal{L} \in \mathbb{R}\ .$$

Thus, we find the maximum variance of $\mathbf{X}_c\boldsymbol{v}_1$ subject to $\|\boldsymbol{v}_1\|_2^2 = 1$ by equating the partial derivatives of the following Lagrangian multiplier with zero

$$\mathcal{L}(\boldsymbol{v}_1, \lambda_\mathcal{L}) = \sigma_1^2(\boldsymbol{v}_1) + \lambda_L(\boldsymbol{v}_1^T\boldsymbol{v}_1 - 1)\ ,$$

and we find that

$$\begin{aligned}&\partial_{\boldsymbol{v}_1}\mathcal{L} = 0 \\ \Rightarrow\ &\mathbf{C}_x\boldsymbol{v}_1 = \lambda_\mathcal{L}\boldsymbol{v}_1\ ,\end{aligned}$$

which by the definition of the eigenvector implies that $\boldsymbol{v}_1$ is an eigenvector of $\mathbf{C}_x$. Thus, the direction $\boldsymbol{v}_1$ that maximizes the variance σ_1^2, must be an eigenvector of $\mathbf{C}_x$. Hence, it follows that the total variance of $\mathbf{X}_c$ in the direction of most variability is given as

$$\begin{aligned}\sigma_1^2 &= \boldsymbol{v}_1^T\mathbf{C}_x\boldsymbol{v}_1 = \boldsymbol{v}^T\boldsymbol{v}\lambda_1 \\ &= \lambda_1\ ,\end{aligned}$$

where λ_1 is the largest eigenvalue of $\mathbf{C}_x$, and where the eigenvalues of $\mathbf{C}_x$ are ordered $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$.

Similarly, it follows that maximizing the variance of $\mathbf{X}_c\boldsymbol{v}_j$ subject to $\boldsymbol{v}_j^T\boldsymbol{v}_j = 1$ and $\boldsymbol{v}_j^T\boldsymbol{v}_i = 0,\ j > i$, leads to the following optimization problem

$$\max \sigma_j^2 \quad \text{subject to } \boldsymbol{v}_j^T\boldsymbol{v}_j = 1,\ \boldsymbol{v}_j^T\boldsymbol{v}_i = 0,\ j > i\ ,$$

for all $j = 2, ..., p$. Since each $\boldsymbol{v}_j$ maximizes $\boldsymbol{v}_j^T\mathbf{C}_x\boldsymbol{v}_j$ under the constraints $\boldsymbol{v}_j^T\boldsymbol{v}_j = 1$ and $\boldsymbol{v}_j^T\boldsymbol{v}_i = 0$ for $i < j$, it follows that each $\boldsymbol{v}_j$ is an eigenvector of $\mathbf{C}_x$ corresponding to eigenvalue λ_j.

By the *spectral theorem* (Horn and Johnson, 2014), the eigenvectors of the symmetric matrix $\mathbf{C}_x$ form an orthonormal basis. Therefore, the matrix $\mathbf{V} = \{\boldsymbol{v}_j\}_{j=1}^m$, whose columns are the eigenvectors of $\mathbf{C}_x$, diagonalizes $\mathbf{C}_x$ and defines the directions of maximum variance in the data $\mathbf{X}_c$ under the constraint that each vector $\boldsymbol{v}_j$ is orthogonal to all previous vectors $\boldsymbol{v}_i$, for $i < j$.

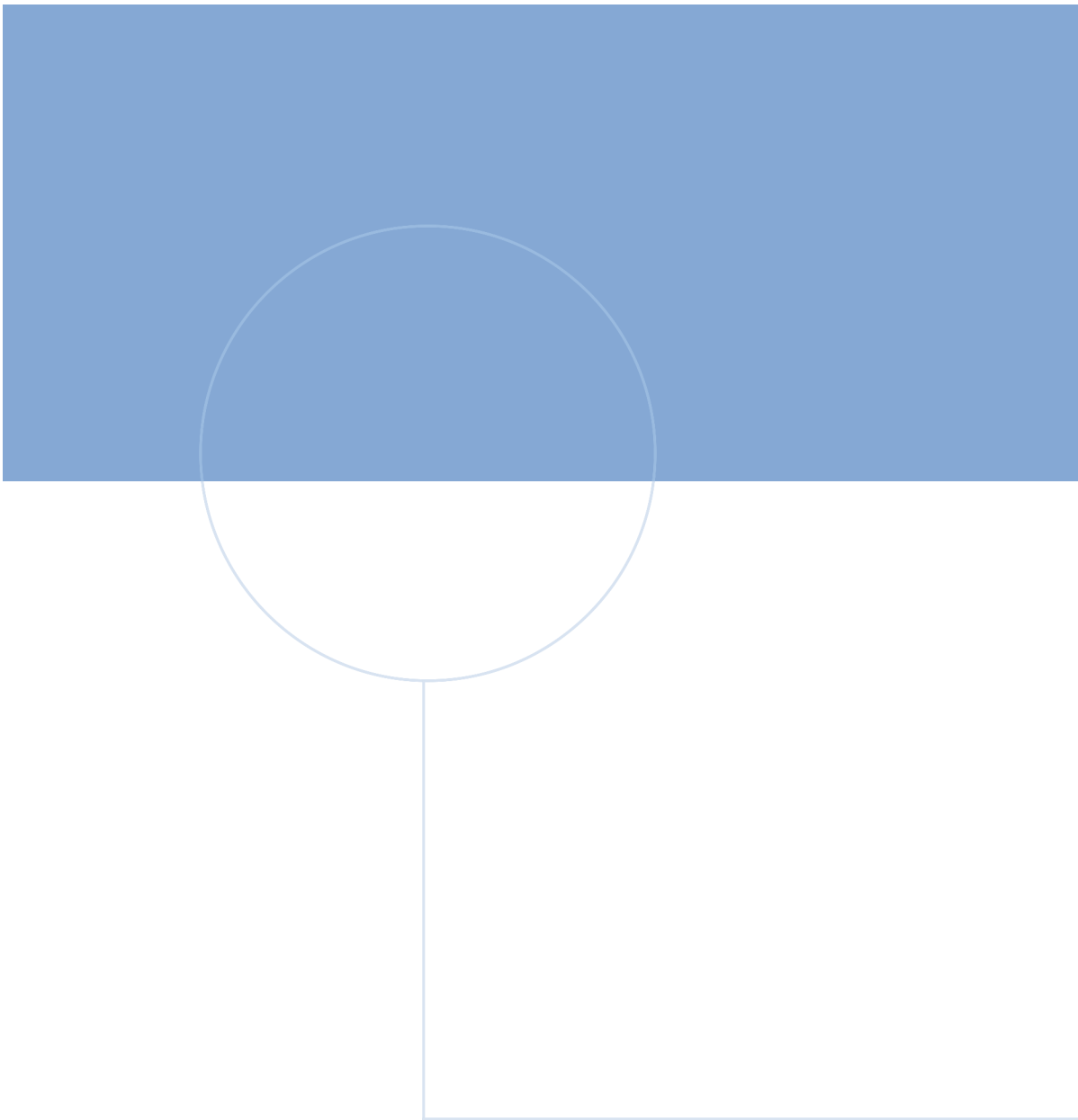
Furthermore, since

$$\mathrm{Var}(\mathbf{X}_c \boldsymbol{v}_j) = \boldsymbol{v}_j^T \mathbf{C}_x \boldsymbol{v}_j = \lambda_j \ ,$$

the variance explained by the component $\mathbf{X}_c \boldsymbol{v}_j$ is equal to the eigenvalue λ_j, and the total variance in $\mathbf{X}_c$ is given by

$$\mathrm{Tr}(\mathbf{C}_x) = \sum_{j=1}^{m} \lambda_j \ .$$

□



NTNU

Norwegian University of
Science and Technology