Simulation von Phänotypen mittels optimierender linearer Modelle

Thomas Rahimi[1], Regina Roessler[2], Stefanie Klingel[3] und Dirk Hinrichs[1]

Abstract: Für die Tierzucht stellen phänotypische Informationen eine wesentliche Grundlage für Zuchtfortschritt dar [Le16]. Derartige Informationen müssen die Varianz der in der Population vorhandenen Tiere widerspiegeln [VCF93]. Im vorliegenden Versuch wurde die Simulation von Phänotypen mittels linearer Modelle auf Grundlage des LASSO-Algorithmus [Ti96] getestet. Die Umsetzung erfolgt dabei mittels der PCA-Kompression der Zielvariablen, unter Anwendung der gemessenen Koeffizienten. Die Ergebnisse zeigen ähnliche Lagemaße wie die erhobenen Werte. Allerdings weichen die Ergebnisse einzelner simulierter Eigenschaften deutlich ab und die meisten simulierten Eigenschaften weisen eine geringere Varianz als gemessene Werte auf. Zusammenfassend kann gezeigt werden, dass der LASSO-Algorithmus für die Simulation von Phänotypen geeignet ist, allerdings sind noch Optimierungen notwendig.

Keywords: Machine Learning, Phänotypen, Tierzucht, Inferenz, lineare Modelle

1 Einleitung

In der Tierzucht spielen genaue und zuverlässig phänotypische Informationen über die Zuchttiere und ihre Nachkommen eine wichtige Rolle [Le16]. Die ausreichende Erfassung derartiger Informationen stellt besonders bei kleinen Populationen eine Herausforderung dar. Das Fehlen phänotypischer Informationen von vielen Tieren einer Population führt zur Überschätzung der Effekte einzelner Tiere, für die entsprechende Daten vorliegen [VCF93].

Eine Möglichkeit, diese Auswirkungen zu verringern, besteht in der generischen Erzeugung phänotypischer Daten, die auf gemessenen Werten aus der Population basieren. In der Literatur wurden dazu mehrere Ansätze aufgezeigt, die auf unterschiedlichen, statistischen Lernverfahren basieren. Unter der Maßgabe einer parametrischen Verteilung lassen sich die Daten im nicht-überwachten Lernen mittels k-mean und bei nicht-parametrischer Verteilung mittels k-median erzeugen [Gé17]. Eine weitere Möglichkeit besteht in tiefen neuronalen Netzen, die höhere Flexibilität bezüglich der Verteilungsfunktion und der Korrelation mit gemessenen Werten ermöglichen [Be09]. In der vorlie-

1 Universität Kassel, FB11, FG Tierzucht, Nordbahnhofstraße 1a, 37213 Witzenhausen, thomas.rahimi@agrar.uni-kassel.de, dhinrichs@agrar.uni-kassel.de
2 Universität Kassel, FB11, FG Tierzucht, Nordbahnhofstraße 1a, 37213 Witzenhausen
Mittlerweile: Universität Kassel, FB11, FG Tierhaltung in den Tropen und Subtropen, Steinstraße 19, 37213 Witzenhausen, regina.roessler@uni-kassel.de
3Arche Warder e.V., Langwedeler Weg 11, 24646 Warder, klingel@arche-warder.de

genden Arbeit wurde der Least Absolute Shrinkage and Selection Operator (LASSO) als linearer Prediktionsalgorithmus verwendet. Die umgesetzte Implementierung ähnelt der Referenzimplementierung des LASSO-Algorithmus in der Identifikation von Risikofaktoren für Krebserkrankungen [HTF09, Ti96]. Durch die Nutzung eines linearen Verfahrens unterscheidet sich der gewählte Ansatz deutlich von der bei [SKK19] gewählten Methode eines Random-Forrests zur Simulation von Schlachtdaten bei Schafen. Ziel der vorliegenden Datenanalyse ist die Simulation von Aufzucht- und Schlachtdaten von Schafen mit der Fragestellung, ob sich diese mittels eines linearen Verfahrens simulieren lassen.

2 Material und Methoden

2.1 Daten

Die der Modellierung zugrunde liegenden Daten wurden im Rahmen eines Vergleichsversuchs zwischen männlichen Lämmern der Rassen Weißköpfiges Fleischschaf, Texel-, Suffolk,- und Charollaisschaf erhoben. Dabei wurden von 89 Lämmern Aufzuchtdaten wie Tageszunahme, Fleischigkeit, Bemuskelung und Gewicht erfasst. Zusätzlich wurden von 53 Lämmern Schlachtdaten, wie warmes und kaltes Schlachtgewicht, erhoben.

2.2 Training und Test der Simulations-Funktion

Die Simulation wurde in Python3 mit den Funktionen der scikit-learn-Bibliothek (sklearn) implementiert [Pe11]. Es wurde im Wesentlichen der LASSO-Algorithmus, wie von [Ti96] vorgeschlagen, verwendet, da er eine automatische Selektion von Koeffizienten für das lineare Modell, die durch den Kontrollparameter λ beeinflusst werden kann, ermöglicht [HTF09, 68ff]. Für die vorliegenden Daten bietet der LASSO-Algorithmus weitere Vorteile. Da keine Umweltdaten zu den erfassten Daten vorliegen, was den gegenwärtigen Ansatz von der bei [SKK19] angewendeten Methode unterscheidet, wird ein linearer Zusammenhang zwischen den Werten für Aufzucht und Schlachtung angenommen. Außerdem erlaubt die resultierende Koeffizientenmatrix eine bessere Erklärbarkeit des Modells und bietet somit die Möglichkeit der Optimierung des Ansatzes. In sklearn sind mehrere Implementierungen des LASSO-Algorithmus enthalten, die sich vor allem in der Lösung des quadratischen Teils der Gleichung unterscheiden. Von diesen Implementierungen wurde die Least Angle Regression (LARS) gewählt, die eine besonders effiziente Implementierung darstellt [HTF09, 73ff].

Durch Veränderung des λ lässt sich die Zahl der einbezogenen Koeffizienten für die Gleichung modifizieren. Ein Wert von $\lambda = 0$ führt zu einer linearen Gleichung unter Einbeziehung aller Koeffizienten, bei Erhöhung des Werts für λ entstehen lineare Gleichungen mit zunehmender Anzahl an Koeffizienten [Ti96]. Die Werte für λ werden

anhand der vorliegenden Trainingsdaten ermittelt. Dafür wird eine Erweiterung des LASSO-Algorithmus in sklearn verwendet, mittels dessen sich Optimierungen in linearen Modellen umsetzen lassen. Die Zahl der Iterationen für die Optimierung ist dabei auf die halbe Länge des Datensets gesetzt. Dadurch lassen sich angepasste Werte für λ bestimmen und die Übereinstimmung zwischen gemessenen und simulierten Daten erhöhen.

Alle Zielvariablen der Simulation werden für die Durchläufe mittels Hauptkomponentenanalyse (PCA) komprimiert, wobei die resultierenden Daten 95 % der vorhandenen Varianz widerspiegeln müssen. Da das Datenset mit 89 Tieren eher klein ist, erfolgt die PCA über Trainings- und Testdaten in einem Schritt. Dadurch lassen sich gemeinsame Eigenvektoren und Mittelwerte (ϕ) bestimmen, die benötigt werden, um die PCA zur PCA_{rev} mittels folgender Formel zurückzurechnen:

$$PCA_{rev} = PC_{scores} \cdot Eigenvector^{T} + \phi$$

Die Hauptkomponenten (PC_{scores}) werden in der Simulation durch den LASSO-Algorithmus bestimmt. Das Training des Modells erfolgt mit einer Teilung der Daten in einen Trainingsdatensatz (80 % der Tiere) und einen Testdatensatz (20 % der Tiere) durch eine Ziehung einer zufälligen, stratifizierten Stichprobe, mit der Rasse als Konstante. Die Daten der Koeffizienten werden bezüglich Mittelwert und Standardabweichung normalisiert. Nach dem Abschluss des Modelltrainings und den zugehörigen Tests wird das Modell als ausführbare Datei gespeichert, um reproduzierbare Ergebnisse zu erhalten.

2.3 Erzeugen eines generischen Datensets

Die Erstellung des generischen Datensets nutzt verschiedene Mechanismen, um Daten für die Simulation zu erzeugen. Da die trainierten Modelle Koeffizienten benötigen, um die Zielvariablen zu simulieren, müssen diese in mehreren Schritten erzeugt werden. Ein Teil der Koeffizienten besteht aus vollständig zufälligen Werten, denen keine Verteilungsfunktion zugrunde liegt, während weitere Koeffizienten als normalverteilt innerhalb eines definierten Wertebereichs und bei den restlichen Koeffizienten ein linearer Zusammenhang mit den übrigen Koeffizienten angenommen wird. Bei den vollständig zufälligen Variablen werden zufällige Werte mittels der Python3-Standardbibliothek erzeugt. Für die als normalverteilt angenommenen Variablen werden die Zufallsfunktionen aus der numpy-Bibliothek [WCV11] und bei Variablen im linearen Zusammenhang wird eine erneute Simulation mittels des LASSO-Algorithmus verwendet. Die Vorgehensweise bezüglich Training und Test ist dabei dieselbe wie im vorhergehenden Abschnitt beschrieben. Auf Grundlage der erhaltenen Koeffizienten wird in der Folge das trainierte Modell ausgeführt, um die Aufzucht- und Schlachtdaten zu simulieren.

2.4 Qualitätskontrollen der Algorithmen

Während Training und Test erfolgt eine kontinuierliche Kontrolle der Genauigkeit des Modells durch Ermittlung der Mittleren Abweichungsquadrate (RSME) mittels folgender Formel aus den gemessenen (r_{pred}) und den vorhergesagten Werten (r_{meas}):

$$RSME = \sqrt{\sum \frac{\left(r_{pred} - r_{meas}\right)^2}{n_{samples}}}$$

Weiterhin wird für den Vergleich zwischen gemessenen und simulierten Testdaten ein paarweiser t-Test mit einem Signifikanzniveau von $p > 0{,}05$ durchgeführt, für den die Implementierung in scipy verwendet wird [Vi20]. Für die simulierten Daten werden die Lagemaße der Quantile und des Medians mit den gemessenen Daten verglichen.

3 Ergebnisse

3.1 Training und Test der Simulations-Funktion

Der direkte Vergleich zwischen simulierten und gemessenen Trainingsdaten bringt für die meisten Eigenschaften hohe Übereinstimmungen zwischen den Daten mit Abweichungsquadraten im überwiegend niedrigen einstelligen Bereich. Nur für das Gewicht des Nierenfetts, die tägliche Zunahme sowie die Nettozunahme lassen sich geringere Übereinstimmungen zwischen gemessenen und simulierten Daten finden, mit Abweichungsquadraten bis zu 93,731 für Nierenfett. In der Auswertung der Lagemaße zeigt sich, dass schlechtere Übereinstimmungen zwischen gemessenen und simulierten Daten zu höheren Standardabweichungen der Eigenschaften führen. Der t-Test ergibt dennoch für alle Eigenschaftspaare aus simulierten und gemessenen Daten p-Werte über dem gewählten Signifikanzniveau von 0,05 und bestätigt damit die Hypothese, dass die gemessenen Daten aus Stichproben derselben Grundgesamtheit stammen. Der Vergleich zwischen gemessenen und simulierten Daten des Modells zeigt außerdem, dass bei den meisten Eigenschaften eine geringere Varianz vorliegt. Ausnahmen sind hier nur die tägliche Zunahme und das Lebendgewicht am Beginn und Ende des Prüfzeitraums.

Die verwendeten Werte für λ finden sich in Tabelle 1. Diese sind eher niedrig und nehmen für keine Zielvariable 0 an, wodurch eine Selektion der Koeffizienten erfolgt.

Simulationswert	λ
Aufzucht – PCA 1	0,568722
Schlachtung – PCA 1	0,629635
Schlachtung – PCA 2	0,135855

Tab.1: λ-Werte für die einzelnen Simulationen

3.2 Erzeugung eines generischen Datensets

Der LASSO-Algorithmus liefert Ergebnisse, die gemäß der Lagemaße weitestgehend der Verteilung und Varianz in den gemessenen Variablen entsprechen. Die im Training der Simulations-Funktion aufgetretenen Ungenauigkeiten bei einzelnen Eigenschaften, zum Beispiel Nierenfett, führen in der Erzeugung des generischen Datensatzes zu messbaren Abweichungen der Lagemaße bei diesen Eigenschaften. Während für die übrigen Werte Übereinstimmungen zwischen generischen und gemessenen Daten bezüglich der Lagemaße vorliegen, trifft dies für das Gewicht des Nierenfetts, die täglichen Zunahmen sowie das Lebendgewicht nicht zu.

Im Vergleich der Varianzen zwischen generierten und gemessenen Daten fällt eine deutlich geringere Varianz für die meisten Eigenschaften in den generierten Daten auf. Die Ausnahmen sind in diesem Fall die Schlachtgewichte, Nettozunahme, Schulter- und Keulendurchmesser sowie die Keulenbreite.

4 Diskussion

Die vorliegenden Ergebnisse zeigen, dass die Simulation von Phänotypen mittels linearer Modelle für viele Eigenschaften mit zufriedenstellender Genauigkeit funktioniert. Allerdings liefert die gewählte Methode, bei ähnlichen Werten für Quantile und Median, deutlich andere Varianzen. Eine Erklärung für den Verlust an Varianz stellt die PCA-Kompression dar, deren Ergebnis 95 % der vorhandenen Variablen widerspiegelt. Eventuelle Optimierungen bietet hier sowohl die PCA-Kompression, als auch die Rückrechnung zu unkomprimierten Werten. Besonders der addierte Mittelwert sollte für die entsprechenden Eigenschaften darauf hin überprüft werden, ob er die Population gut darstellt. Weitere Optimierungen bestehen besonders für den Wert, den λ annimmt. In der momentanen Implementierung ist der Wert indirekt abhängig vom Stichprobenumfang. Dieses Vorgehen neigt zu Overfitting des Modells, wie bei [SKK19] diskutiert. Hier ist die Vergrößerung der Stichprobe notwendig, um das Modell weiter zu optimieren.

5 Fazit

Es zeigt sich, dass eine Simulation von Phänotypen mittels linearer Modelle bei zufriedenstellenden Genauigkeiten möglich ist. Allerdings zeigen sich die Limitierungen, gerade bezüglich der Größe des verwendeten Datensets für Training und Test des Modells.

Literaturverzeichnis

[Be09] Bengio, Yoshua: *Learning Deep Architectures for AI* : Now Publishers Inc, 2009 — ISBN 978-1-60198-294-0

[Gé17] Géron, Aurélien ; Rother, K. (Übers.): *Praxiseinstieg Machine Learning mit Scikit-Learn und TensorFlow: Konzepte, Tools und Techniken für intelligente Systeme*. Heidelberg : O'Reilly, 2017 — ISBN 978-3-96009-061-8

[HTF09] Hastie, T. ; Tibshirani, R. ; Friedman, J. H.: *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. 2. ed. New York, NY, 2009 — ISBN 978-0-387-84857-0

[Le16] Leroy, G. ; Besbes, B. ; Boettcher, P. ; Hoffmann, I. ; Capitan, A. ; Baumung, R.: Rare phenotypes in domestic animals: unique resources for multiple applications. In: *Animal Genetics* Bd. 47 (2016), Nr. 2, S. 141–153

[Ma19] Maltecca, Christian ; Lu, Duc ; Schillebeeckx, Constantino ; McNulty, Nathan P. ; Schwab, Clint ; Shull, Caleb ; Tiezzi, Francesco: Predicting Growth and Carcass Traits in Swine Using Microbiome Data and Machine Learning Algorithms. In: *Scientific Reports* Bd. 9 (2019), Nr. 1, S. 6574

[Pe11] Pedregosa, F. ; Varoquaux, G. ; Gramfort, A. ; Michel, V. ; Thirion, B. ; Grisel, O. ; Blondel, M. ; Prettenhofer, P. ; u. a.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* (2011), Nr. 12, S. 2825–2830

[SKK19] Shahinfar, Saleh ; Kelman, Khama ; Kahn, Lewis: Prediction of sheep carcass traits from early-life records using machine learning. In: *Computers and Electronics in Agriculture* Bd. 156 (2019), S. 159–177

[Ti96] Tibshirani, R.: Regression Shrinkage and Selection via the Lasso. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), Nr. Vol. 58, No. 1, S. 267–288

[VCF93] Verrier, E. ; Colleau, J. J. ; Foulley, J. L.: Long-term effects of selection based on the animal model BLUP in a finite population. In: *TAG. Theoretical and applied genetics. Theoretische und angewandte Genetik* Bd. 87 (1993), Nr. 4, S. 446–454

[Vi20] Virtanen, Pauli ; Gommers, Ralf ; Oliphant, Travis E. ; Haberland, Matt ; Reddy, Tyler ; Cournapeau, David ; Burovski, Evgeni ; Peterson, Pearu ; u. a.: SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python. In: *Nature Methods* Bd. 17 (2020), Nr. 3, S. 261–272. — arXiv: 1907.10121

[WCV11] van der Walt, Stéfan ; Colbert, S Chris ; Varoquaux, Gaël: The NumPy Array: A Structure for Efficient Numerical Computation. In: *Computing in Science & Engineering* Bd. 13 (2011), Nr. 2, S. 22–30