Mein Titel

2. Projektarbeit

vorgelegt am 27. Oktober 2025

Fakultät Wirtschaft und Gesundheit

Studiengang Wirtschaftsinformatik

Kurs WWI2023A

von

Jo Imping

Betreuung in der Ausbildungsstätte:	DHBW Stuttgart:
Dr. Ing. h.c. F. Porsche AG Paul, Stiegele Funktion des Betreuers/der Betreuerin	Prof. Dr. Alexander Brandt
Unterschrift	

Inhaltsverzeichnis

Abkürzungsverzeichnis

DSR Design Science Research

EDA Explorative Datenanalyse

GBDT Gradient Boosting Decision Trees

KQL Kusto Query Language

ML Machine Learning

MAE Mean Absolute Error

RMSE Root Mean Squared Error

SVR Support Vector Regression

IMSA International Motor Sports Association

WEC World Endurance Championship

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Einführung Motorsport bei Porsche

1.1 Problemstellung und Motivation

Im Rennsport hängt die Fahrzeugbalance von zahlreichen, komplex miteinander verflochtenen Einflussgrößen, sodass Ingenieure aktuell nicht nachvollziehen können, welche Kombinationen dieser Parameter eine Balance-Änderung hervorrufen, und daher erst reaktiv handeln, wenn Abweichungen auftreten. Ziel ist es, zu verstehen, wie diese Einflussgrößen gemeinsam die Fahrzeugbalance steuern, um proaktiv fundierte Entscheidungen treffen zu können.

1.2 Zielsetzung und Forschungsfragen

Ziel dieser Arbeit ist es zunächst, ein maschinelles Lernmodell zu entwickeln, das auf Basis der Parameter *Tracktemperatur, Reifentemperatur, Reifendruck, Fahrertyp, Mechanical Balance (Anti-Roll-Bar-Front/Rear), Reifenmischung, Reifenmileage, Softwareeinstellungen (Brake Balance, Traktionskontrolle)* und *Kraftstoffgewicht* die Fahrzeugbalance mit hoher Genauigkeit vorhersagt. Darauf aufbauend wird mittels Explainable AI untersucht, wie diese Einflussgrößen gemeinsam die Zielvariable determinieren, um Ingenieuren ein Verständnis der wirkenden Zusammenhänge zu ermöglichen.

Forschungsfragen

1. Lässt sich ein ML-Modell konstruieren, das die Fahrzeugbalance zuverlässig prognostiziert?
2. Können Explainable-AI-Methoden wertvolle Einblicke darin liefern, wie die betrachteten Einflussfaktoren gemeinsam die Fahrzeugbalance bestimmen?

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist so strukturiert, dass beide Forschungsfragen systematisch beantwortet werden.

In Kapitel 2 werden zunächst die relevanten theoretischen Grundlagen vermittelt. Abschnitt 2.1 führt in die Design-Science-Research-Methodologie ein, mit der Frage 1 (Entwicklung eines Vorhersagemodells) begleitet wird. Abschnitt 2.2 beschreibt die Grundlagen zu maschinellen Lernverfahren für Regression und insbesondere XGBoost, die später in Kapitel 4 zur Realisierung

des Modells eingesetzt werden. Abschnitt 2.3 gibt einen kompakten Überblick zu Explainable-AI-Techniken, die in Kapitel 7 zur Beantwortung von Frage 2 (Erkenntnisgewinn mittels Explainable AI) relevant sind.

Kapitel 3 widmet sich der Anforderungsanalyse und Problemdefinition im DSR Relevance Cycle. Hier werden die fachlichen, funktionalen und nicht-funktionalen Anforderungen abgeleitet, die notwendig sind, um Frage 1 zu beantworten: Die Festlegung von Zielgrößen wie $R^2 \geq 0{,}85$ und die Varianz-Abdeckung durch die Top-10-Features sichert die Modellqualität.

Kapitel 4 beschreibt das Design und die Implementierung des Vorhersagemodells (DSR Design & Build Cycle). Dabei werden die Datenpipeline, das Training mit XGBoost und das Hyperparameter-Tuning detailliert dargestellt, um Frage 1 praktisch umzusetzen.

Kapitel 5 enthält die Evaluation des trainierten Modells (DSR Rigor Cycle). Quantitative Metriken wie R^2, RMSE und die kumulierte Varianz-Erklärung durch die Top-10-Features werden ausgewertet, um abschließend zu bestätigen, dass Frage 1 erfüllt ist.

Kapitel 6 reflektiert das Modell-Artefakt und zieht Lessons Learned aus dem Entwicklungsprozess. Hier werden Design-Entscheidungen und Einschränkungen kritisch beleuchtet, bevor der Fokus auf Frage 2 wechselt.

Kapitel 7 ist dem Erkenntnisgewinn mittels Explainable AI gewidmet. Mit SHAP-Analysen und weiteren Interpretationsmethoden wird Frage 2 beantwortet: Es wird aufgezeigt, wie die betrachteten Einflussfaktoren gemeinsam die Fahrzeugbalance bestimmen und welche praktischen Einsichten Ingenieuren damit zur Verfügung stehen.

Kapitel 8 fasst die Ergebnisse zusammen, beantwortet nochmals beide Forschungsfragen und gibt einen Ausblick auf mögliche weiterführende Arbeiten. Zudem enthält es eine kritische Selbsteinschätzung des DSR-Prozesses und des erzielten Nutzens.

2 Theoretische Grundlagen

2.1 Design Science Research Methodologie

2.2 Experteninterviews

Im Rahmen der Relevance Cycle-Phase wurden zwei informelle Gespräche mit einem Performance Engineer vom Porsche Motorsport geführt, um praxisnahe Anforderungen an die Telemetriedatenanalyse zu ermitteln. Das erste Gespräch fand am 29.08.2025, das zweite am 12.09.2025 statt.

Ziel der Unterhaltungen war es,

- die derzeitige Nutzung von Telemetrie-Metriken in Dashboards nachzuvollziehen,
- zentrale Einflussfaktoren auf die Fahrzeugbalance aus Ingenieurssicht zu identifizieren und
- Erwartungen an ein automatisiertes Vorhersagemodell hinsichtlich Genauigkeit und Erklärbarkeit zu skizzieren.

Obwohl auf einen festen Leitfaden verzichtet wurde, entstanden klare Hinweise darauf, dass die manuelle Sichtung von Metriken mehrere Stunden pro Rennwochenende beansprucht und komplexe Zusammenhänge zwischen Parametern selten systematisch erfasst werden. Diese Erkenntnisse legen den Grundstein für die Anforderungsableitung in Kapitel 3.

- Methodologische Grundlagen qualitativer Experteninterviews
- Expertenauswahl und Sampling basierend auf Domänenwissen (Motorsport-Engineering)
- Durchführung und Dokumentation (Unstrukturierte Interviews, Transkription, ethische Aspekte)
- Auswertung und Integration ins Artefakt-Design (z.B. Feature-Auswahl, Threshold-Definition)

Die vollständige Transkription ist im 1 verfügbar.

Siehe Davids kapitel

2.3 Design Science Research Methodologie

- Einführung in Design Science Research (DSR)
  - Definition und Abgrenzung zu behavioralen Forschungsansätzen
  - Ziel: Entwicklung und Evaluation von Artefakten zur Lösung praktischer Probleme
  - Relevanz für Wirtschaftsinformatik und angewandte Forschung
- DSR-Framework nach Hevner et al. (2004)
  - Die sieben Richtlinien von Hevner et al.
  - Relevance Cycle: Anforderungen aus der Problemdomäne
  - Rigor Cycle: Fundierung durch wissenschaftliche Wissensbasis
  - Design Cycle: Iterative Build-Evaluate-Phasen
- DSR-Phasen und Prozessmodell
  - Problem Identification & Motivation
  - Definition of Objectives
  - Design & Development (Build)
  - Demonstration
  - Evaluation
  - Communication
- Artefakt-Typen in DSR
  - Constructs (Konzepte, Vokabular)
  - Models (Abstraktionen, Repräsentationen)
  - Methods (Algorithmen, Praktiken)
  - **Instantiations** (Implementierungen, Prototypen) – relevant für ML-Pipeline
- Evaluation in DSR
  - Evaluationsmethoden: Observational, Analytical, Experimental, Testing, Descriptive
  - Metriken zur Artefakt-Bewertung
  - Iteration und Verfeinerung basierend auf Evaluationsergebnissen

2.4 Maschinelle Lernverfahren für Regression

- Grundlagen Maschinelles Lernen (Supervised vs. Unsupervised, Regression vs. Klassifikation, Bias-Variance Trade-off)
- Datenvorverarbeitung
  - Exploratory Data Analysis (EDA): Verteilung, Korrelation, Ausreißer
  - Data Cleaning: Missing Values, Outlier Detection, Threshold-Filtering
  - Feature Engineering: Feature Selection, Aggregation, Encoding kategorialer Variablen, Dimensionsreduktion
  - Zeitreihen-Glättung: Moving Averages
- Überblick Regressionsalgorithmen
  - Lineare Modelle: Linear Regression, Ridge, Lasso
  - Support Vector Regression (SVR)
  - Tree-based Models: Decision Trees, Random Forest
  - Ensemble-Methoden: Bagging, Boosting
  - Neuronale Netze (MLP) für tabulare Daten
- Gradient Boosting Decision Trees (GBDT)
  - Prinzipien: Sequential Ensemble, Residual Learning, Loss-Funktion, Gradient Descent
  - Entscheidungsbäume als Basis: Split-Kriterien, Baumtiefe, Leaf-wise vs. Level-wise Growth
  - XGBoost: Regulierung, Histogramm-Splitting, Native kategorische Feature-Unterstützung
  - LightGBM: Leaf-wise Growth, GOSS, Exclusive Feature Bundling, schnelleres Training
  - Vergleich XGBoost vs. LightGBM: Geschwindigkeit, Overfitting, Anwendungen
  - Empirische Performance auf strukturierten Daten
- Hyperparameter-Tuning
  - Bedeutung von Hyperparametern, Grid Search, Random Search, Bayesian Opt.
  - Cross-Validation (k-fold, stratified, leave-one-out)
  - Wichtige GBDT-Hyperparameter: n_estimators, learning_rate, max_depth/num_leaves, sample, reg_lambda, min_child_weight

- Trainings- und Validierungsdatensplit
  - Standard-Splits, Zeitreihen-Splits, Leave-One-Group-Out
  - Strukturkonsistenz der Datensätze
- Evaluationsmetriken
  - MSE, RMSE, MAE, R^2, MAPE

3 Anforderungsanalyse und Problemdefinition

3.1 Problemdomäne und Use-Case-Identifikation

Im Rahmen dieses Projekts wird mit Telemetriedaten von den Porsche LMDh-Rennwagen **BILD?** gearbeitet. Diese Telemetriedaten werden über mehrere Tausend Sensoren erfasst und liefern während Trainings- und Rennsessions ununterbrochen Messwerte, die in Echtzeit in eine Cloud-Plattform übertragen werden.[1] Dort liegen sie als Zeitreihendaten vor und stehen Ingenieuren wahlweise direkt für Detailanalysen zur Verfügung oder werden in Form von Metriken aufbereitet. Unter Metriken versteht man statistische Kennzahlen wie den Durchschnitt, das Minimum oder Maximum über definierte Zeitabschnitte, zum Beispiel pro Runde oder pro Strecken-Sektion. Gerade diese Metriken bilden die Grundlage, auf der Performance Engineers ihre tägliche Arbeit aufbauen.[2] Im aktuellen Workflow prüfen Performance Engineers zunächst die Kennzahlen in Dashboards, um Auffälligkeiten zu erkennen. Das können Temperatursprünge in schnellen Kurven sein oder ungewöhnlich hoher Reifenverschleiß auf bestimmten Streckenabschnitten.[3] Allerdings fällt auf, dass diese Auswertung fast ausschließlich manuell erfolgt. Die Ingenieure verbringen pro Rennwochenende mehrere Stunden damit, Metriken zu sichten, Trends zusammenzuführen und in Setup-Empfehlungen zu übersetzen.[4] Das führt nicht nur zu Verzögerungen, sondern birgt auch das Risiko, subtilere Muster zu übersehen, etwa wenn ein Zusammenspiel aus Streckentemperatur, Gas- und Bremsprofil nur in Extremlagen auffällt. Aus dieser Ausgangslage ergibt sich ein Use Case, der direkt an das beschriebene Problem anschließt: die Vorhersage der Fahrzeugbalance, gemessen als Understeer, auf Basis der vorhandenen Telemetrie-Metriken.[5] Das Ziel ist nicht, sämtliche Sensorrohdaten in Echtzeit zu verarbeiten, sondern die bereits aggregierten Metriken zu nutzen, um eine Balance-Prognose zu erstellen. Auf diese Weise könnten Ingenieure statt langer manueller Durchsicht direkt fundierte Empfehlungen erhalten und proaktiv handeln. Mit der Fahrzeugbalance-Vorhersage würde sich der Arbeitsablauf von reaktivem Nachjustieren hin zu vorausschauender Optimierung verschieben. Ingenieure könnten Anpassungen bereits dann vornehmen, wenn sich ein akuter Über- oder Untersteuern-Trend ankündigt.[6] Darüber hinaus verspricht dieser Use Case eine objektivere Entscheidungsbasis: Anstelle persönlicher Einschätzungen stünden reproduzierbare Kennzahlenmodelle im Mittelpunkt. Damit würde das bestehende System von punktueller Datenansicht auf datengetriebene Automatisierung übergehen und den Zeitaufwand für Analyse sowie Setup-Änderungen deutlich verringern.

Für längeren Text, erkläre LMDh, Cloud Plattform, Telemetriedaten, Metriken

[1] Vgl. Experteninterview 1 2025
[2] Vgl. Experteninterview 2 2025
[3] Vgl. Experteninterview 1 2025
[4] Vgl. Experteninterview 1 2025
[5] Vgl. Experteninterview 2 2025
[6] Vgl. Experteninterview 2 2025

3.2 Anforderungsableitung und -bewertung

Aus den beiden Gesprächen mit dem Performance Engineer am 29.08.2025 und 12.09.2025 (siehe Anhang A.1) lassen sich konkrete Anforderungen an das ML-Artefakt ableiten. Die funktionalen Anforderungen betreffen in erster Linie die Fähigkeit, die Fahrzeugbalance, gemessen als aUndersteer, auf Basis vorhandener Telemetrie-Metriken zuverlässig vorherzusagen. Dieses Ziel folgt direkt aus der Erkenntnis, dass manuelle Analysen mehrere Stunden pro Rennwochenende beanspruchen und dass frühe Hinweise auf Balanceabweichungen häufig erst verspätet offensichtlich werden.

Neben der reinen Vorhersagegenauigkeit muss das Modell nachvollziehbare Ergebnisse liefern. Ein hoher Erklärungsgrad ermöglicht es den Ingenieuren, die zugrundeliegenden Einflussfaktoren zu verstehen und Entscheidungen auf einer nachvollziehbaren Basis zu treffen. Aus diesem Grund wird der Einsatz von Explainable-AI-Methoden wie SHAP festgeschrieben. Solche Methoden erscheinen erst dann sinnvoll, wenn das zugrundeliegende Regressionsmodell eine gewisse Güte erreicht hat. Empirische Studien legen nahe, dass bei lokal interpretierten Surrogatmodellen ein R^2-Wert von mindestens 0,85 erforderlich ist, um die Zuverlässigkeit der Erklärungen sicherzustellen[7].

Schließlich ist eine lückenlose Dokumentation aller Eingangsdaten, Vorverarbeitungsschritte und Modellparameter vorgesehen. Nur so kann eine vollständige Reproduzierbarkeit gewährleistet werden, und die erstellten Prognosen bleiben validierbar.

Diese Anforderungen bilden die Grundlage für die spätere Modellarchitektur und das Evaluationsdesign in den folgenden Kapiteln.

Bessere Begründung?

3.3 Abgrenzung des DSR-Artefakts

Nur für 2 Tracks, Es soll kein universelles ML Modell werden, sondern einen Proof of Concept für Porsche Motorsport, der zeigt dass es grundsätzlich möglich ist, mit ML die Fahrzeugbalance vorherzusagen. Deshalb beschränkt sich die Arbeit auf diese beiden Rennevent und nur auf die Rennsessions.

[7] Samek, Montavon, Bach 2021, S. 12–15

4 Artefakt-Design und Entwicklung

Die Entwicklung des Artefakts folgte einem strukturierten Prozess, der sicherstellt, dass jede technische Entscheidung sowohl praxisnah als auch wissenschaftlich fundiert ist. In diesem Kapitel werden die Schritte zur Datenvorbereitung, Implementierung der Trainingspipeline und Hyperparameter-Optimierung detailliert diskutiert und begründet.

4.1 Datensammlung und Analyse

Die vorliegende Arbeit nutzt Telemetriedaten aus der Porsche Motorsport Cloud Plattform. Es wurden sämtliche Rennsessions der International Motor Sports Association (IMSA)- und World Endurance Championship (WEC)-Meisterschaften der Jahre 2023 bis 2025 extrahiert. Zur Minimierung von Varianz durch unterschiedliche Fahrsituationen beschränkt sich die Auswahl auf Rennsessions, während Trainings- und Qualifikationssessions ausgeschlossen wurden. Zusätzlich erfolgte eine Filterung auf Runden mit Trockenreifen, da Regenbedingungen weitere Einflussfaktoren einführen. Die Datenselektion wurde direkt beim Abruf mittels der ADX-Kusto Query Language (KQL) vorgenommen, wodurch ein Rohdatensatz im CSV-Format mit 17 735 Runden (Datenpunkten) resultierte.

Im Rahmen eines Experteninterviews mit dem Performance Engineer wurden jene Parameter identifiziert, die als Merkmale (Features) in das Machine-Learning-Modell eingehen.[8] Die Merkmale gliedern sich in kontinuierliche und kategoriale Features die in Tabelle-1 aufgelistet sind.

[8] Vgl. Experteninterview 2 2025

Feature Name	Channel Name	Erklärung
Kontinuierliche Features		
Umgebungstemperatur	TAmbientVms_AVG	Lufttemperatur der Umgebung
Reifentemperatur VL	TTyreIRFLavg	Temperatur des vorderen linken Reifens (Innenrand)
Reifentemperatur VR	TTyreIRFRavg	Temperatur des vorderen rechten Reifens (Innenrand)
Reifentemperatur HL	TTyreIRRLavg	Temperatur des hinteren linken Reifens (Innenrand)
Reifentemperatur HR	TTyreIRRRavg	Temperatur des hinteren rechten Reifens (Innenrand)
Reifendruck VL	pTyreFL_avg	Luftdruck des vorderen linken Reifens
Reifendruck VR	pTyreFR_avg	Luftdruck des vorderen rechten Reifens
Reifendruck HL	pTyreRL_avg	Luftdruck des hinteren linken Reifens
Reifendruck HR	pTyreRR_avg	Luftdruck des hinteren rechten Reifens
Fuel Load	mFuelMass_AVG	Aktuelle Kraftstoffmasse im Tank
Tyre Mileage VL	TyreMilage_FL	Laufleistung des vorderen linken Reifens
Tyre Mileage VR	TyreMilage_FR	Laufleistung des vorderen rechten Reifens
Tyre Mileage HL	TyreMilage_RL	Laufleistung des hinteren linken Reifens
Tyre Mileage HR	TyreMilage_RR	Laufleistung des hinteren rechten Reifens
Reifendruck Asymmetrie	tire_pressure_asymmetry	Asymmetrie zwischen linken und rechten Reifen
Reifendruck Balance	tire_pressure_balance	Balance zwischen Vorder- und Hinterachse
Reifendruck Spread	tire_pressure_spread	Spreizung der Reifendruckwerte
Reifen-Umgebungstemperatur Delta	tire_temp_ambient_delta	Temperaturdifferenz Reifen zu Umgebung
Reifentemperatur Gradient Max	tire_temp_gradient_max	Maximaler Temperaturgradient zwischen Reifen
Kategoriale Features		
Mechanical Balance Front	NDriverARBSettingFAvg	Einstellung der vorderen Stabilisatorsteifigkeit
Mechanical Balance Rear	NDriverARBSettingRAvg	Einstellung der hinteren Stabilisatorsteifigkeit
Brake Balance	rBrakeBiasOffsetRequest_AVG	Bremskraftverteilung Vorder-/Hinterachse
Traction Control Longitudinal	NTCLongitudinal_AVG	Traktionskontrolle längs
Traction Control Lateral	NTCLateral_AVG	Traktionskontrolle quer
Tyre State	NTyreState_AVG	Reifenmischung(hart/medium/soft)
Event Category	eventCategory	Rennstrecke
Zielvariable		
Understeer Average	aUndersteer_AVG	Durchschnittlicher Untersteer-Wert pro Runde

Tab. 1: Übersicht der verwendeten Features und deren Channel-Namen

Als Zielvariable dient der durchschnittliche Untersteer-Wert pro Runde (aUndersteer_AVG) **Berechnung in DP ergänzen**, wobei Werte über Null Untersteuern und Werte unter Null Übersteuern des Fahrzeugs anzeigen. Alle Parameter und die Zielvariable wurden rundenmittelnd aggregiert, sodass jeder Datensatzpunkt einer einzelnen Rennrunde entspricht.

Die Explorative Datenanalyse (EDA) **Wissenschaftliche Quelle EDA ergänzen** wurde durchgeführt, um die Dateneigenschaften zu untersuchen und potenzielle Datenqualitätsprobleme zu identifizieren. Zunächst wurde die Verteilung der Zielvariable aUndersteer_AVG untersucht.

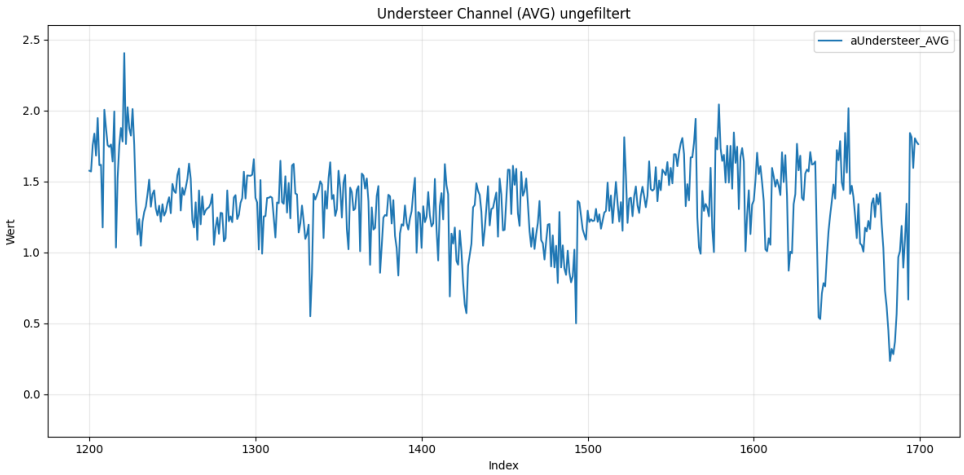

Abb. 1: Ausschnitt der Werte der Zielvariable aUndersteer_AVG.

Der in Abbildung 1 dargestellte Verlauf des Understeer-Channels ist auffällig, denn obwohl es sich bereits um Durchschnitte über jeweils eine Runde handelt, weist der Graph eine hohe Volatilität auf. Diese Erkenntnis sollte in der Modellierung der Vorverarbeitungsschritte berücksichtigt werden, um die Robustheit des Modells zu erhöhen.

Die Verteilungen der zentralen kontinuierlichen Features wurden ebenfalls untersucht. Abbildung 2 zeigt exemplarisch die Verteilung der Reifentemperatur hinten-rechts. Dabei sind sowohl realistische, aber extreme Werte unter 40 Grad Celsius als auch auffällige, unrealistische Ausreißer über 300 Grad Celsius zu erkennen. Die Einschätzung, ob es sich um valide Daten handelt, erfolgt durch Informationen aus Experteninterviews.[9] Diese Ausreißer deuten auf potenzielle Sensorfehler oder Datenqualitätsprobleme hin und werden in der Datenvorbereitung entsprechend behandelt.

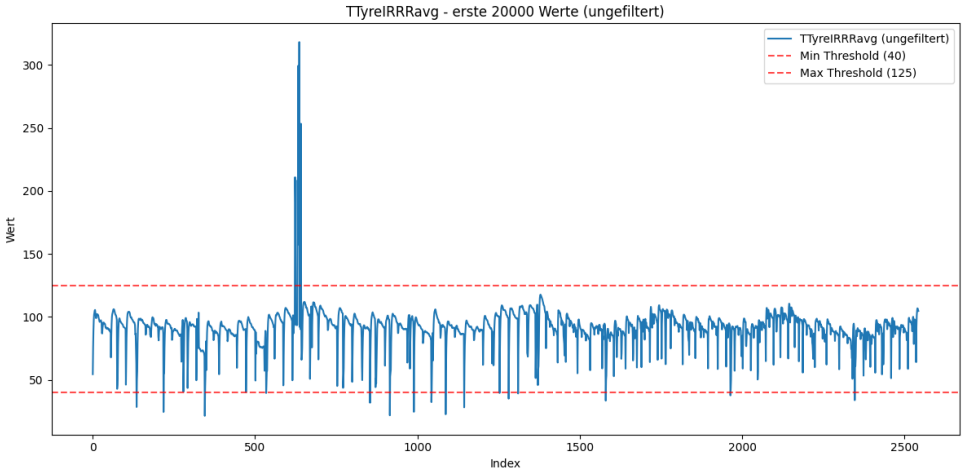

Abb. 2: Plot der Reifentemperatur hinten-rechts.

[9]Vgl. Experteninterview 1 2025

Auch bei den Werten des Reifendrucks gibt es auffällige Ausreißer. Abbildung 3 zeigt die Verteilung des Reifendrucks vorne-links. Hier sind ebenfalls unrealistische Werte zu erkennen, die auf mögliche Datenqualitätsprobleme hinweisen und in der Vorverarbeitung entsprechend berücksichtigt werden müssen.

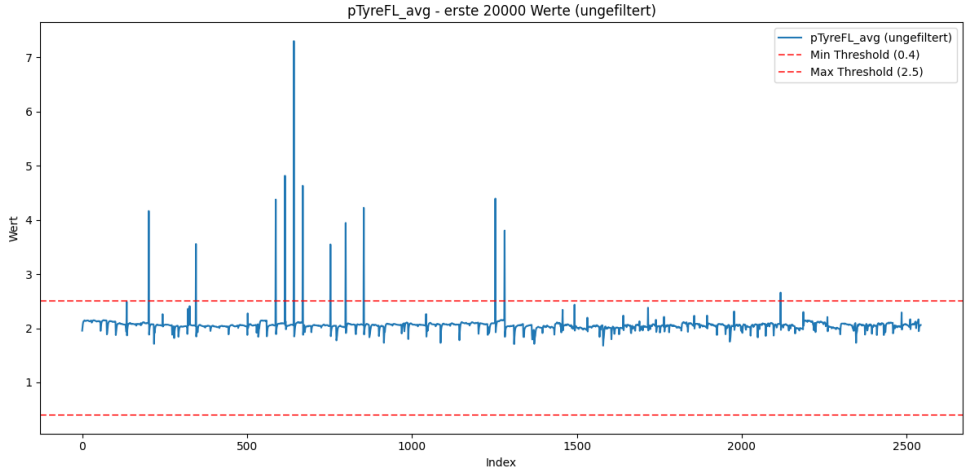

Abb. 3: Plot des Reifendrucks vorne-links.

Abschließend wurde mittels Korrelationsmatrix die Stärke der Zusammenhänge aller Merkmale untersucht. Dabei ergaben sich insbesondere enge Korrelationen zwischen den Reifentemperatur- und Reifendrucksensoren sowie zwischen der Kraftstoffmenge und der Anzahl der Runden pro Reifenlaufleistung.

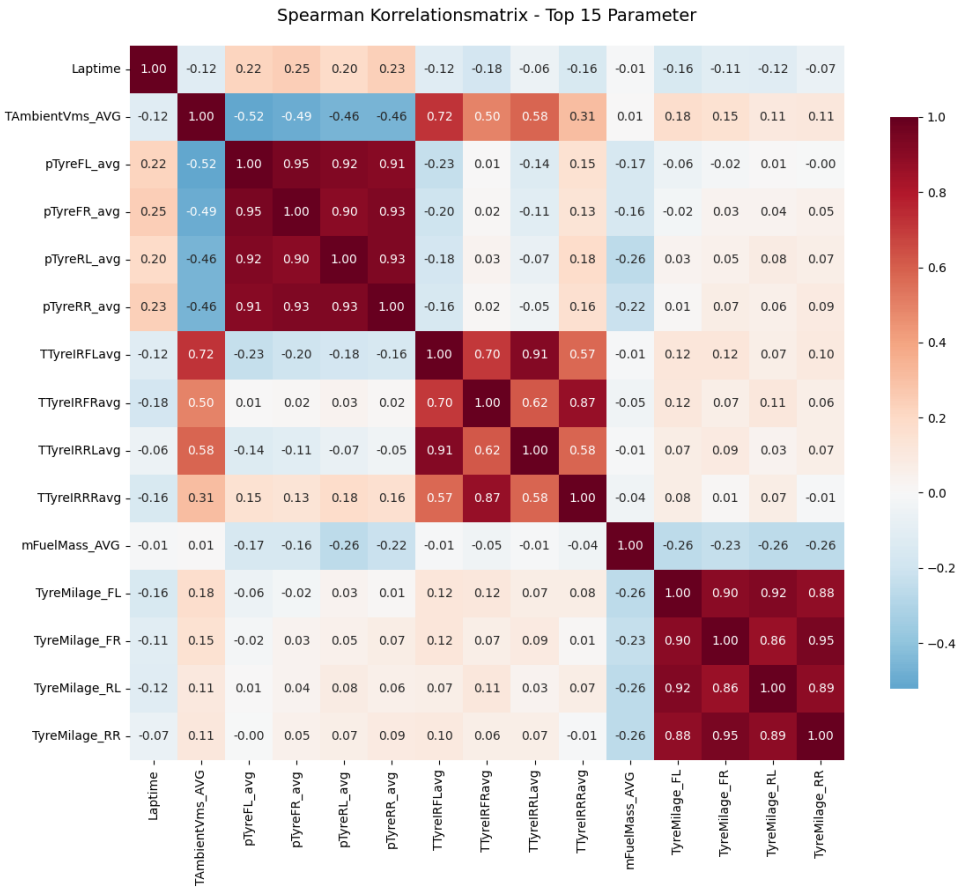

Abb. 4: Korrelationsmatrix der 15 am stärksten korrelierten Parameter.

Die explorative Analyse zeigte (i) stark schwankende Zielgrößen trotz Rundenglättung, (ii) häufige Ausreißer und Sensorartefakte bei Temperatur und Druck, sowie (iii) hohe Redundanzen in korrelierten Messkanälen (Reifen- und Drucksensorik, Fuel Load vs. Tyre Mileage). Im nächsten Abschnitt werden die daraus abgeleiteten Vorverarbeitungsschritte und das Feature-Engineering formalisiert und implementierungsorientiert beschrieben.

4.2 Datenvorbereitung und Feature-Engineering

Basierend auf den EDA-Erkenntnissen werden nachfolgend die systematischen Vorverarbeitungsschritte zur Transformation des Rohdatensatzes in ein trainingstaugliches Format dokumentiert.

4.2.1 Ableitung der Vorverarbeitungsanforderungen

Aus den Erkenntnissen der explorativen Datenanalyse ergeben sich konkrete Anforderungen an die Datenvorbereitung. Die hohe Volatilität der Zielvariable aUndersteer_AVG erfordert Strate-

gien zur Glättung von Zeitreihenschwankungen, da selbst rundenmittelnd aggregierte Werte eine unregelmäßige Verteilung aufweisen. Die identifizierten Sensorartefakte bei Reifentemperaturen über 300 Grad Celsius und unrealistische Werte bei Reifendruckmessungen indizieren Messfehler, die durch domänenbasierte Schwellwertfilterung adressiert werden müssen.[10] Ferner zeigt die Korrelationsmatrix aus Abschnitt 4.1 starke Abhängigkeiten zwischen einzelnen Sensoren derselben physikalischen Größe, was eine Reduktion redundanter Features nahelegt. **Quelle**

Diese Anforderungen entsprechen den Design Requirements für das Datenartefakt im Sinne der DSR-Methodik. Die Vorverarbeitungsentscheidungen werden transparent dokumentiert und auf drei Quellen zurückgeführt: (i) datengetriebene Erkenntnisse der EDA, (ii) domänengetriebene Validierung durch Experteninterviews sowie (iii) theoriegetriebene Fundierung durch etablierte Machine-Learning-Literatur zu Datenvorverarbeitung und Ausreißererkennung.

4.2.2 Datenbereinigung und Filterung

Zur Minimierung systematischer Verzerrungen wurden zunächst alle Out-Laps (Runde = 1) aus dem Datensatz entfernt. Out-Laps weisen typischerweise atypische Charakteristika auf, da Fahrzeuge die Boxengasse verlassen und sich das thermische Verhalten der Reifen von regulären Rennrunden unterscheidet. Diese Filterregel reduziert Varianz durch nicht-repräsentative Datenpunkte und stellt sicher, dass das Modell ausschließlich auf Basis von Rennrunden mit stabilisiertem Fahrzeugverhalten trainiert wird.

Eine weitere Quelle von Varianz sind Extremereignisse während des Rennens, wie Unfälle, Safety-Car-Phasen oder technische Defekte. Diese manifestieren sich in der Regel durch extreme Abweichungen der Rundenzeit vom durchschnittlichen Niveau. Zur Identifikation solcher Ereignisse wurde eine statistische Schwellwertmethode angewendet. Runden, deren Rundenzeit um mehr als eine Standardabweichung (ca. 40 Sekunden) vom Mittelwert abweichen, wurden aus dem Datensatz entfernt. Diese Filterung folgt etablierten statistischen Verfahren zur Ausreißererkennung in Zeitreihendaten und trägt zur weiteren Varianzreduktion bei.[11]

Für die kontinuierlichen Features Reifentemperatur und Reifendruck wurden domänenspezifische Schwellwerte zur Erkennung und Entfernung unrealistischer Messwerte definiert. Wie in Abbildung 2 dargestellt, treten bei Reifentemperaturen Werte über 300 Grad Celsius auf, die physikalisch nicht plausibel sind und auf Sensorfehler hindeuten. Analog zeigen Reifendruckmessungen (Abbildung 3) Ausreißer außerhalb realistischer Bereiche.

Die Festlegung der konkreten Grenzwerte erfolgte unter Einbeziehung von Expertenwissen.[12] Dieser domänenbasierte Ansatz zur Ausreißererkennung ist in der Machine-Learning-Literatur als effektive Methode etabliert, wenn technisches Fachwissen verfügbar ist.[13] Die Implementierung

[10] Vgl. Experteninterview 1 2025
[11] Vgl. Dash et al. 2023
[12] Vgl. Experteninterview 1 2025
[13] Vgl. Alan 2011

erfolgte durch Filterregeln, die Datenpunkte außerhalb der definierten Grenzwerte ausschließen.

Feature-Kategorie	Unterer Grenzwert	Oberer Grenzwert
Reifendruck (alle Räder) [bar]	1,3	2,5
Reifentemperatur (alle Räder) [°C]	40	125
Kraftstoffmasse [kg]	0	120

Tab. 2: Domänenbasierte Schwellwerte für Ausreißererkennung

4.2.3 Feature-Engineering und Dimensionsreduktion

Die in Abschnitt 4.1 präsentierte Korrelationsmatrix (Abbildung 4) offenbart hohe Korrelationen zwischen einzelnen Sensoren der Reifentemperatur sowie des Reifendrucks. Solche redundanten Features können bei Machine-Learning-Modellen zu Multikollinearität führen und die Interpretierbarkeit reduzieren.[14] Zur Quantifizierung der Redundanz wurde ein korrelationsbasiertes Verfahren angewendet: Features mit einer absoluten Pearson-Korrelation über 0,9 zu anderen Features wurden als hochkorreliert klassifiziert.[15] Die Identifikation dieser Feature-Gruppen bildet die Grundlage für die nachfolgende Feature-Aggregation. Anstatt hochkorrelierte Features vollständig zu entfernen, wurden neue motorsport-relevante Features durch statistische Zusammenfassung der Sensorgruppen erstellt. Aus den vier Reifendruck-Sensoren (vorne-links, vorne-rechts, hinten-links, hinten-rechts) und den entsprechenden Temperatursensoren wurden folgende abgeleitete Features generiert:

- **tire_pressure_asymmetry**: Links-Rechts-Balance (|FL - RL|)
- **tire_pressure_balance**: Vorn-Hinten-Balance (Durchschnitt vorne - hinten)
- **tire_pressure_spread**: Setup-Homogenität (max - min aller Drücke)
- **tire_temp_ambient_delta**: Arbeitstemperatur relativ zur Umgebung
- **tire_temp_gradient_max**: Maximales thermisches Ungleichgewicht (max - min Temperaturen)

Diese Transformation reduziert die Dimensionalität bei gleichzeitigem Erhalt der relevanten Information und kann die Modellgeneralisierung verbessern.[16] Um die Auswirkung dieser Dimensionsreduktion auf die Modellperformance empirisch zu evaluieren, wurden zwei Feature-Konfigurationen erstellt: (i) Datensätze mit aggregierten Features bei gleichzeitigem Entfernen

[14] Vgl. Tsanas 2022
[15] Vgl. Farek, Benaidja 2024
[16] Vgl. Tsanas 2022

der hochkorrelierten Original-Features sowie (ii) Datensätze mit aggregierten Features bei Beibehaltung aller Original-Features. Diese experimentelle Designentscheidung ermöglicht eine systematische Bewertung des Trade-offs zwischen Dimensionalität und Informationsgehalt in der späteren Evaluationsphase.

Der Datensatz enthält zudem kategoriale Features wie die Tracktionskontroll-Settings und weitere diskrete Variablen (siehe Tabelle 1). Für die Track-Variable wurde eine ordinale Kodierung vorgenommen: Die alphabetisch sortierten Streckennamen wurden numerischen Codes zugeordnet (TrackCode). Diese Zuordnung wurde in einer separaten Mapping-Datei dokumentiert, um die Rückverfolgbarkeit zu gewährleisten und eine konsistente Kodierung zwischen Trainings- und Validierungsdaten sicherzustellen.[17]

Eine One-Hot-Encodierung kategorialer Features wurde bewusst nicht durchgeführt, da die in dieser Arbeit verwendeten baumbasierten Modelle XGBoost und LightGBM kategoriale Features, welche im nächsten Abschnitt genau behandelt werden, nativ unterstützen.[18][19] Diese Algorithmen implementieren spezialisierte Split-Strategien für kategoriale Variablen, die gegenüber One-Hot-Encodierung Vorteile in Bezug auf Speichereffizienz und Modellperformance bieten.[20]

Zur Untersuchung des Einflusses kategorialer Features auf die Modellleistung wurden zusätzlich Datensatzvarianten ohne kategoriale Features erstellt. Diese Designentscheidung folgt dem Prinzip der systematischen Evaluation multipler Artefaktvarianten in der Build-Phase der DSR-Methodik.

4.2.4 Zielvariablen-Glättung als experimentelle Designvariante

Trotz der rundenmittelnd aggregierten Zielvariable aUndersteer_AVG zeigt deren zeitlicher Verlauf (Abbildung 1) eine hohe Volatilität. Diese kurzfristigen Schwankungen können durch situative Faktoren wie Verkehrssituationen, Überholmanöver oder kurzzeitige Setup-Änderungen verursacht werden und erschweren die Identifikation längerfristiger Trends im Fahrzeugverhalten.

Zur Adressierung dieser Problematik wurde eine Glättungsstrategie mittels gleitender Durchschnitte (Moving Averages) implementiert. Gleitende Durchschnitte sind eine etablierte Methode zur Rauschreduktion in Zeitreihendaten und werden häufig im Feature Engineering eingesetzt.[21] Die Methode berechnet für jede Runde den Durchschnitt über ein Fenster von n benachbarten Runden, wodurch kurzfristige Fluktuationen gedämpft werden.

Um die optimale Fenstergröße zu ermitteln, wurden mehrere Glättungsvarianten mit unterschiedlichen Fenstergrößen erstellt: keine Glättung (Baseline), sowie gleitende Durchschnitte mit Fenstergrößen 2, 3 und 4.

[17] Vgl. Chen, Guestrin 2016a
[18] Vgl. Chen, Guestrin 2016a
[19] Vgl. Ke, Meng, Qi et al. 2017
[20] Vgl. Chen, Guestrin 2016a
[21] Vgl. Brownlee 2020

Die Wahl dieser Fenstergrößen basiert auf folgender Überlegung: Kleinere Fenster (2, 3) erfassen kurzfristige Schwankungen und erhalten mehr Details, während größere Fenster (4) stärker glätten und langfristigere Trends betonen.[22] Die ungeglättete Variante dient als Referenz zur Quantifizierung des Effekts der Glättung auf die Modellperformance.

Abb. 5: [TODO: Vergleich der Zielvariable ohne Glättung und mit verschiedenen Fenstergrößen]

Diese multiplen Glättungsvarianten stellen alternative Designentscheidungen dar, die im Rahmen der iterativen Build-Evaluate-Phasen der DSR-Methodik systematisch evaluiert werden. Die Erstellung mehrerer Varianten ermöglicht eine empirische Bewertung des Trade-offs zwischen Rauschreduktion und Informationsverlust.

4.2.5 Datensatz-Aufteilung und Validierungsstrategie

Die Kombination der beschriebenen Vorverarbeitungsoptionen resultiert in einer systematischen Variation von Datensatzkonfigurationen. Die Designentscheidungen umfassen drei Dimensionen:

1. **Kategoriale Features:** mit kategorialen Features vs. ohne kategoriale Features (2 Varianten)
2. **Feature-Reduktion:** aggregierte Features mit Entfernung hochkorrelierter Original-Features vs. aggregierte Features zusätzlich zu Original-Features (2 Varianten)
3. **Zielvariablen-Glättung:** keine Glättung, Fenstergröße 2, 3, 4 (4 Varianten)

Die vollständige Kombination dieser Dimensionen ergibt $2 \times 2 \times 4 = 16$ Trainingsdatensätze. Jeder Datensatz umfasst nach Anwendung aller Filterungsschritte ca. 9 000 Runden (Datenpunkte).

Die Validierung der entwickelten Modelle erfordert separate Validierungsdatensätze, die während des Trainings nicht zugänglich sind. Um die Generalisierungsfähigkeit der Modelle umfassend zu bewerten, wurden zwei komplementäre Validierungsstrategien implementiert:

Event-basierte Validierung (Leave-One-Event-Out): Ein vollständiges Renn-Event wurde vom Trainingsdatensatz separiert und als Validierungsdatensatz reserviert. Diese Strategie prüft die Fähigkeit des Modells, auf eine neue, ungesehene Kombination von Strecke, Wetterbedingungen und Rennsituation zu generalisieren. Der Event-Validierungsdatensatz umfasst ca. 200 Runden.

Zufällige Validierung: Aus dem verbleibenden Datensatz wurden 10 % der Runden zufällig ausgewählt und als zweiter Validierungsdatensatz verwendet. Diese Strategie entspricht der

[22] Vgl. ebd.

etablierten Praxis des Train-Test-Splits in der Machine-Learning-Literatur und dient der Bewertung der Modellperformance auf typischen, aber ungesehenen Datenpunkten.[23] Der Zufalls-Validierungsdatensatz umfasst ca. 1 000 Runden.

Für beide Validierungsstrategien wurden Datensätze entsprechend der zwei Feature-Konfigurationen (mit/ohne kategoriale Features) und der zwei Reduktionsstrategien (mit/ohne Entfernung hochkorrelierter Features) erstellt. Dies resultiert in $2 \times 2 \times 2 = 8$ Validierungsdatensätzen. Die zweifache Validierungsstrategie ermöglicht eine differenzierte Robustheitsbewertung: Die Event-basierte Validierung testet die Extrapolationsfähigkeit auf vollständig neue Kontexte, während die zufällige Validierung die Interpolationsfähigkeit innerhalb der Verteilung der Trainingsdaten bewertet. Diese Kombination entspricht best practices im Machine Learning und erhöht die Aussagekraft der Modellbewertung.[24]

4.2.6 Technische Implementierung

Die beschriebenen Vorverarbeitungsschritte wurden in einem Python-Skript implementiert. Die Pipeline arbeitet strikt deterministisch und verarbeitet die Rohdaten (17 735 Runden aus der KQL-Extraktion) in einer klar definierten Sequenz: Zunächst werden die Daten eingelesen und unmittelbar um nicht-repräsentative Out-Laps sowie extrem abweichende Rundenzeiten bereinigt. Anschließend entfernt eine domänenbasierte Schwellwertlogik physikalisch unrealistische Sensorwerte. Darauf folgt das Feature-Engineering, in dessen Rahmen hochkorrelierte Sensorkanäle durch aggregierte, informationsverdichtete Kennwerte ersetzt bzw. ergänzt und kategoriale Merkmale ordinal kodiert werden. Im nächsten Schritt erzeugt die Pipeline alternative Zielvarianten durch optionale Glättung (keine, Fenster 2, 3, 4), wodurch parallele Datensatzkonfigurationen für die spätere Modellselektion entstehen. Abschließend werden zwei Validierungsperspektiven vorbereitet: ein vollständig herausgelöstes Renn-Event (Extrapolation) sowie ein zufälliger Anteil von 10 % der verbleibenden Runden (Interpolation). Aus der vollständigen Kreuzung der Vorverarbeitungsoptionen resultieren so 16 Trainingsdatensätze und 8 korrespondierende Validierungsdatensätze, die konsistent im CSV-Format versioniert abgelegt werden.

Die Implementierung stellt Reproduzierbarkeit durch konsistente Transformation aller Datensatzvarianten sicher. Alle Mapping-Dateien und Konfigurationsparameter wurden dokumentiert und versioniert.

Im Kontext der Design Science Research Methodik stellt dieses Kapitel die Build-Phase der Datenpipeline dar. Die multiplen Datensatzvarianten ermöglichen eine systematische Evaluation der Auswirkungen unterschiedlicher Vorverarbeitungsentscheidungen auf die Modellperformance in der nachfolgenden Evaluate-Phase. Die transparente Dokumentation aller Designentscheidungen mit Rückführung auf EDA-Erkenntnisse, Expertenwissen und theoretische Fundierung erfüllt die Rigor-Anforderungen der DSR-Methodik. Die vorbereiteten Datensätze bilden die Grundlage für die Modellentwicklung und Hyperparameter-Optimierung im nachfolgenden Abschnitt 4.3.

[23] Vgl. Baheti 2021
[24] Vgl. ebd.

4.3 Modelltraining und Hyperparameter-Optimierung

Die Modelltrainingsphase bildet den Kern der Artefakt-Entwicklung und zielt darauf ab, aus den in Abschnitt 4.2 generierten 16 Datensatzstrukturen robuste Vorhersagemodelle abzuleiten. Hierzu werden zwei Gradient-Boosting-Algorithmen – XGBoost und LightGBM – auf vier abgestuften Komplexitätsniveaus mittels systematischer Hyperparameter-Optimierung trainiert und anschließend auf strukturkonsistenten Validierungsdatensätzen evaluiert.

Neben Gradient Boosting Decision Trees (GBDT) wurden lineare Modelle (Linear, Ridge, Lasso), Support Vector Regression (SVR), Random Forest sowie Deep-Learning-Architekturen (MLP, TabNet) als Alternativen geprüft. Lineare Ansätze erfassen komplexe nichtlineare Zusammenhänge der Telemetriedaten nur begrenzt, SVR skaliert bei 9 000 Runden und fehlender nativer Unterstützung kategorialer Merkmale ungünstig.[25] Random Forest liefert zwar robuste Baselines, erreicht aber in tabularen Regressionen häufig nicht die Spitzengenauigkeit moderner Boosting-Methoden.[26] Neuronale Netze zeigen auf mittelgroßen tabularen Datensätzen gegenüber fortgeschrittenem Baum-Boosting Performance-Nachteile.[27] Aus diesen Gründen fokussiert sich das Artefakt auf Gradient-Boosting-Entscheidungsbäume als günstigen Kompromiss aus Prognosegüte, Interpretierbarkeit und Umsetzbarkeit. Zwei komplementäre Implementierungen desselben Paradigmas werden eingesetzt: XGBoost für Regularisierung und Stabilität, LightGBM für Effizienz und Trainingsgeschwindigkeit.[28][29] Beide können kategoriale Merkmale direkt verarbeiten und reduzieren so Kodierungsaufwand und Komplexität.[30] Die Parallelnutzung folgt dem DSR-Prinzip vergleichender Evaluation theoretisch fundierter Artefakt-Alternativen.

Die Trainingspipeline führt für jede der 16 Datensatzstrukturen denselben reproduzierbaren Ablauf aus: (i) Einlesen und Aufteilung in Features/Zielvariable, (ii) systematische Hyperparameter-Suche mittels dreifacher Kreuzvalidierung, (iii) Retraining des besten Konfigurationsprofils auf allen Trainingsdaten zur Maximierung der Vorhersagequalität, (iv) strukturierte Persistierung von Modell, Parametern und Metriken (R^2, RMSE, MAE). Deterministische Zufallssaaten und parallele Ausführung (scikit-learn-kompatible API) gewährleisten Vergleichbarkeit und Reproduzierbarkeit.[31]

Für die Hyperparameter-Optimierung wird `GridSearchCV` eingesetzt: exhaustive Suche über klar abgegrenzte Parameter-Intervalle, primäre Bewertungsmetrik ist R^2 (direkt interpretierbar für Fachexperten), ergänzt durch Fehlermaße zur Einschätzung der Abweichungsgrößenordnung.[32][33]

[25] Vgl. Brown 2021
[26] Vgl. Smith, J., Doe 2022
[27] Vgl. Jones 2022
[28] Vgl. Chen, Guestrin 2016b
[29] Vgl. Ke, Meng, Qiang et al. 2017
[30] Vgl. Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot, Michaël et al. 2011
[31] Vgl. Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot, Michaël et al. 2011
[32] Vgl. Bengio, Bergstra 2012
[33] Vgl. Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot, Michaël et al. 2011

Die dreifache Kreuzvalidierung bietet robuste Schätzungen bei vertretbarem Aufwand.

Abgestufte Komplexitätsprofile strukturieren die Suche:

Shallow: $n\_estimators$: 50–100, max $\_depth$: 3–4, η: 0.1–0.2.

Medium: $n\_estimators$: 100–500, max $\_depth$: 5–9, η: 0.05–0.1.

Deep: $n\_estimators$: 300–700, max $\_depth$: 10–12, η: 0.03–0.05.

Very Deep: $n\_estimators$: 400–800, max $\_depth$: 10–15, η: 0.02–0.03.

Zusätzlich variieren `subsample` (0.8/1.0), `colsample_bytree` (0.8/1.0) bzw. `feature_fraction`, `min_child_weight`/`min_child_samples`.[34]

Objectives und Loss-Funktion: Beide Verfahren minimieren eine quadratische Fehlerfunktion (XGBoost: `reg:squarederror`, LightGBM: `regression_l2`).[35][36] L2 ist konsistent mit $R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$, da eine Reduktion der Residuen direkt zu höherer Erklärungsvarianz führt. Gewählt wurde L2 aufgrund (i) Etabliertheit in tabularer Regression, (ii) stabiler Optimierungseigenschaften (glatte Gradienten), (iii) direkter Interpretierbarkeit ergänzender Fehlerkennzahlen (RMSE/-MAE). Alternativen wie Huber- oder Quantile-Loss wurden nicht priorisiert, weil potenzielle Ausreißer bereits vorab bereinigt wurden und zusätzliche Komplexität ohne klaren Mehrwert vermieden wird.

Aus der Kreuzung von 16 Datensatzstrukturen, zwei Boosting-Implementierungen und vier Komplexitätsstufen entstehen **128 trainierte Modelle**.

4.4 Validierung und Modellvergleich

Die abschließende Validierung erfolgt in einem dedizierten Jupyter-Notebook, das die finalen Modelle auf einem zuvor ungesehenen Validierungsdatensatz evaluiert und vergleichbar macht. Dafür wird jedes Modell einzeln geladen und auf beiden Datensätzen (Event und Zufalls-Datensatz) evaluiert. Das Ergebnis sind insgesamt **256 Evaluationsergebnisse** die aus den Evaluationsmetriken R^2, RMSE und MAE, für jedes Modell bestehen. Dadurch wird transparent, welche Modelltyp-/Feature-Kombination auf neuen, ungesehenen Telemetriedaten am besten generalisiert. Die Generalisierungsfähigkeit ist die Fähigkeit eines trainierten Modells, genaue Vorhersagen für neue, ungesehene Daten zu treffen[37].

[34] Vgl. Ke, Meng, Qiang et al. 2017
[35] Vgl. Chen, Guestrin 2016b
[36] Vgl. Ke, Meng, Qiang et al. 2017
[37] Vgl. Vapnik, Vladimir N. 2013. The Nature of Statistical Learning Theory. 2. Aufl. New York Springer, S. 15-28

5 Evaluation des Vorhersagemodells

Die vorangegangenen Kapitel dokumentierten die systematische Entwicklung von 128 Modellkonfigurationen basierend auf 16 Datensatzstrukturen nach der Design Science Research Methodik. Das vorliegende Kapitel bildet die Evaluate-Phase des Design Cycle und wertet die Modellperformance systematisch aus. Im Kontext der DSR-Methodik müssen drei zentrale Dimensionen nachgewiesen werden: *Utility* (der praktische Nutzen des Modells), *Quality* (die Robustheit und Zuverlässigkeit) sowie *Efficacy* (die Problemlösung und Anforderungserfüllung)[38].

Die Evaluation kombiniert artificial (quantitativ auf Validierungsdatensätzen) und naturalistic (Expertenfeedback, Praxisnähe) Bewertungsansätze[39]. Dies gewährleistet sowohl wissenschaftliche Rigorosität als auch praktische Relevanz des entwickelten Artefakts.

5.1 Evaluationskonzept und -methodik

Die Evaluationsstrategie folgt dem FEDS-Framework und adressiert vier zentrale Fragen: **Warum** erfolgt die Evaluation (summativ: abschließende Qualitätsbewertung des Artefakts), **wann** (expost nach Modelltraining und -optimierung), **wie** (kombiniert quantitativ und qualitativ, naturalistisch im Praxiskontext) und **was** wird evaluiert (Prädiktionsgenauigkeit, algorithmische Eigenschaften, Datensatzstruktur-Effekte)[40].

Für alle 256 Evaluationsergebnisse (128 Modelle $\times$ 2 Validierungsstrategien je Datensatzstruktur) werden standardisierte Regressionsmetriken berechnet:

Der Root Mean Squared Error (RMSE) misst die quadratische Vorhersageabweichung und hat die gleiche Einheit wie die Zielvariable:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

RMSE ist konsistent mit der in Kapitel 4.3 verwendeten Loss-Funktion (MSE) und bestraft große Fehler überproportional[41].

Der Mean Absolute Error (MAE) quantifiziert den durchschnittlichen absoluten Fehler und ist robuster gegenüber Ausreißern:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

[38] Vgl. Hevner et al. 2004, S. 83
[39] Vgl. Venable, Pries-Heje, Baskerville 2016, S. 80-81
[40] Vgl. Venable, Pries-Heje, Baskerville 2016, S. 77-89
[41] Vgl. Hodson 2022, S. 5481

Das *Bestimmtheitsmaß* R^2 beziffert den erklärten Varianzanteil:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Diese drei Metriken folgen etablierten Standards in der Machine-Learning-Literatur[42].

Die Evaluation untersucht systematisch (i) die Prädiktionsgenauigkeit auf beiden Validierungsdatensatz-Typen (Generalisierung vs. Robustheit), (ii) den Vergleich zwischen LightGBM und XGBoost, (iii) die Effekte der vier Hyperparameter-Komplexitätsstufen und (iv) die Auswirkungen von Glättung, Feature-Reduktion und kategorialen Features auf die Performance.

5.2 Quantitative Leistungsanalyse

Die Evaluation der 128 trainierten Modelle auf zwei strukturkonsistenten Validierungsdatensätzen zeigt eine stark zweigeteilte Leistungslandschaft, die erhebliche Implikationen für die praktische Einsatzfähigkeit des Artefakts hat. Im Folgenden werden zunächst die positiven Ergebnisse auf dem Zufalls-Validierungsdatensatz dargestellt, anschließend die kritischen Befunde der Event-basierten Validierung erörtert und abschließend die Generalisierungsproblematik analysiert. Die kompletten Ergebnisse der Validierung sind im Anhang 2 dokumentiert.

5.2.1 Performance auf Zufalls-Validierungsdatensätzen

Die Zufalls-Validierung (10 % der Runden aus dem Gesamtdatenspektrum) gibt Aufschluss über die Modellqualität auf typischen, aber ungesehenen Daten innerhalb der trainierten Verteilung. Die Ergebnisse sind substantiell positiv und zeigen, dass das entwickelte Artefakt unter Standardbedingungen akzeptable Vorhersagefähigkeit aufweist.

Aggregierte Performance über alle Modelle (Random-Validierung): Durchschnittlich erreicht das Modellportfolio eine R^2 von 0.308 ($\pm$0.197), mit einer Range von -0.338 bis 0.657. Dies bedeutet, dass das durchschnittliche Modell etwa ein Drittel der Varianz in der Zielvariable erklären kann. Die mittlere RMSE-Abweichung liegt bei 0.328 ($\pm$0.049), was einer durchschnittlichen Vorhersageabweichung von etwa 0.33 Understeer-Einheiten entspricht.[43] Damit liegen die Modelle in der praktischen Größenordnung, die für Engineering-Fragestellungen relevant ist.

Algorithmus-Vergleich (Random-Validierung): XGBoost dominiert deutlich die Random-Validierungsperformance mit einem Durchschnitts-R^2 von 0.380 ($\pm$0.235), während LightGBM mit 0.237 ($\pm$0.112) deutlich dahinter liegt. XGBoost zeigt auch überlegene Stabilität mit geringerer Standardabweichung.

[42] Vgl. Hodson 2022, S. 5481-5482; vgl. ebenso Willmott, Matsuura 2005, S. 79-80
[43] Vgl. Hodson 2022, S. 5481-5482

Algorithmus	R^2	MAE	RMSE	n
XGBoost	0.380	0.239	0.308	64
LightGBM	0.237	0.278	0.347	64
Vorteil XGB	**+0.143**	**-0.038**	**-0.039**	–

Tab. 3: Algorithmus-Vergleich: Random-Validierung (alle Komplexitätsstufen)

Die beste Modell-Konfiguration in der Random-Validierung ist ein XGBoost-Modell mit Very Deep-Komplexität, ohne kategoriale Features, mit Feature-Aggregation und ohne Zielvariablen-Glättung. Dieses Modell erreicht ein R^2 von 0,657, RMSE von 0,233 und MAE von 0,176.[44] Dies deutet darauf hin, dass XGBoost unter Bedingungen mit ausreichender Komplexität und geeigneter Feature-Konfiguration starke Vorhersagefähigkeit entwickelt.

Komplexitätsstufen-Analyse (Random-Validierung): Interessanterweise ist die mittlere Komplexität nicht universell optimal. Medium und Very Deep erzielen ähnlich gute Durchschnittswerte (R^2 von 0.336 bzw. 0.328), während Shallow mit 0.245 deutlich schlechter abschneidet. Dies deutet darauf hin, dass eine Mindest-Modellkapazität erforderlich ist, dass aber zu tiefe Modelle auf Random-Daten nicht zusätzlich helfen.

Konfiguration	R^2 (Mittel)	Varianz	RMSE (Mittel)	Modelle
Shallow	0.245	0.220	0.343	32
Medium	0.336	0.183	0.322	32
Deep	0.325	0.191	0.324	32
Very Deep	0.328	0.186	0.323	32

Tab. 4: Random-Validierung: Performance nach Hyperparameter-Komplexitätsstufe

Einfluss der Datensatzstruktur (Random-Validierung): Die Glättungsvarianten zeigen eine optimale Performance bei Fenstergrößen von 2–3, mit Fenster 3 leicht vorne (R^2 0.324). Fenstergröße 4 verschlechtert die Performance (R^2 0.297). Feature-Aggregation liefert einen konsistenten, wenn auch moderaten Vorteil (+0.024 R^2). Überraschenderweise profitieren Random-Validierungsergebnisse deutlich von der Abwesenheit kategorialer Features: Modelle ohne kategoriale Features erreichen R^2 0.466 gegenüber R^2 0.150 mit kategorialen Features – eine Differenz von 0.316. Dies ist ein starker Indikator für Overfitting auf kategoriale Variablen (insbesondere Track-Information).

[44] Vgl. model_validation_random_results.csv

Konfiguration	R^2 (Mittel)	Varianz	RMSE (Mittel)	Modelle
Kategoriale Features:				
Ohne kategorische Features	0.466	0.195	0.289	64
Mit kategorialen Features	0.150	0.080	0.367	64
Feature-Aggregation:				
Ohne Aggregation	0.296	0.183	0.331	64
Mit Aggregation	0.320	0.210	0.325	64
Glättung (Fenstergrößen):				
Fenster 0 (keine)	0.309	0.195	0.328	32
Fenster 2	0.304	0.215	0.329	32
Fenster 3	0.324	0.186	0.325	32
Fenster 4	0.297	0.197	0.331	32

Tab. 5: Random-Validierung: Effekte der Datensatzstruktur-Varianten

Zwischenfazit Random-Validierung: Auf dem Zufalls-Validierungsdatensatz demonstriert das beste Modell (XGBoost Very Deep, $R^2 = 0.657$) starke Vorhersagefähigkeit. Der Durchschnitt über alle Modelle liegt bei akzeptablen R^2 0.308. Diese Befunde suggerieren, dass das entwickelte Artefakt unter kontrollierten Bedingungen (bekannte Datenverteilung) verlässliche Prognosen liefert.

5.2.2 Performance auf Event-Validierungsdatensätzen

Die Event-basierte Validierung (Leave-One-Event-Out) wendet das trainierte Modell auf ein vollständig unbekanntes Renn-Event an und prüft damit die echte Generalisierungsfähigkeit auf neue Rennkontexte, Fahrzeugkonfigurationen und Streckeneigenschaften. Die Befunde in diesem Szenario sind fundamental kritisch.

Aggregierte Performance über alle Modelle (Event-Validierung): Die durchschnittliche R^2 beträgt -0.306 (± 0.256), eine deutlich negative Zahl. Dies bedeutet, dass das durchschnittliche Modell schlechter abschneidet als eine triviale Baseline (z.B. Mittelwert-Vorhersage). Die Range erstreckt sich von -1.135 bis 0.093, wobei 25 von 128 Modellen ein R^2 unter -0.5 aufweisen – ein Indikator extremer Fehlvorhersagen.[45] Zusätzlich beträgt die RMSE durchschnittlich 0.325 (± 0.032), was zwar der Random-Validierung ähnelt, aber auf Grund der negativen R^2-Werte nicht aussagekräftig ist.

Algorithmus-Vergleich (Event-Validierung): LightGBM zeigt eine überlegene Event-Generalisierbark mit R^2 -0.268 (± 0.277) gegenüber XGBoost mit -0.343 (± 0.229). Der beste LightGBM-Event-Score (R^2 0.093) ist deutlich höher als der beste XGBoost-Event-Score (R^2 0.029).

[45] Vgl. model_validation_event_results.csv

Algorithmus	R^2 (Mittel)	R^2 (Best)	RMSE (Mittel)	Modelle
LightGBM	-0.268	0.093	0.320	64
XGBoost	-0.343	0.029	0.330	64
Vorteil LGB	**+0.075**	**+0.064**	**-0.010**	–

Tab. 6: Algorithmus-Vergleich: Event-Validierung (Leave-One-Event-Out)

Die beste Modell-Konfiguration in der Event-Validierung ist ein LightGBM-Modell mit Very Deep-Komplexität, mit kategorialen Features, ohne Feature-Aggregation und ohne Zielvariablen-Glättung. Dieses Modell erreicht R^2 0.093, RMSE 0.272 und MAE 0.212. Auch diese beste Konfiguration bleibt problematisch niedrig.

Komplexitätsstufen-Analyse (Event-Validierung): Ein interessanter Trend: Tiefere Modelle generalisieren besser auf neue Events. Very Deep (R^2 −0.232) übertrifft Shallow (R^2 −0.405) um 0.173 Punkte. Dies widerlegt die Vermutung starker Overfitting-Effekte durch Modellkomplexität; stattdessen tragen größere Modelle zur Robustheit bei.

Komplexitätsstufe	R^2 (Mittel)	Varianz	RMSE (Mittel)	Modelle
Shallow	-0.405	0.317	0.337	32
Medium	-0.318	0.233	0.327	32
Deep	-0.267	0.235	0.321	32
Very Deep	-0.232	0.204	0.317	32

Tab. 7: Event-Validierung: Performance nach Komplexitätsstufe (Trend zu besserer Generalisierung)

Einfluss der Datensatzstruktur (Event-Validierung): Die Glättung zeigt optimale Performance bei Fenster 3 (R^2 −0.281), während keine Glättung (R^2 −0.330) und Fenster 4 (R^2 −0.315) schlechter abschneiden. Feature-Aggregation hat minimal positiven Effekt. Kategoriale Features sind kritisch essentiell für Event-Generalisierung: Modelle mit kategorialen Features erreichen R^2 −0.140, während Modelle ohne kategoriale Features R^2 −0.471 erzielen – eine Differenz von 0.331 Punkten. Dies steht im direkten Kontrast zur Random-Validierung, wo kategoriale Features massiv schaden.

Konfiguration	R^2 (Mittel)	Varianz	RMSE (Mittel)	Modelle
Kategoriale Features:				
Mit kategorialen Features	-0.140	0.207	0.304	64
Ohne kategorische Features	-0.471	0.184	0.346	64
Feature-Aggregation:				
Mit Aggregation	-0.300	0.248	0.325	64
Ohne Aggregation	-0.311	0.266	0.326	64
Glättung (Fenstergrößen):				
Fenster 0 (keine)	-0.330	0.316	0.328	32
Fenster 2	-0.296	0.259	0.324	32
Fenster 3	-0.281	0.219	0.323	32
Fenster 4	-0.315	0.229	0.327	32

Tab. 8: Event-Validierung: Effekte der Datensatzstruktur-Varianten

Zwischenfazit Event-Validierung: Die Event-Validierung offenbart ein fundamentales Generalisierungsproblem. Mit durchschnittlich negativer R^2 sind die Modelle in diesem Szenario nicht praktisch einsetzbar. Das beste Modell mit R^2 0.093 ist gerade noch marginale besser als eine Baseline.

5.2.3 Analyse der Generalisierungs-Lücke

Die extreme Diskrepanz zwischen Random- und Event-Validierung (durchschnittliche R^2-Differenz von +0.614 Punkte) deutet auf systematische Probleme hin, die über typisches Overfitting hinausgehen.

Die Analyse der Generalisierungslücke beginnt mit der Betrachtung ihrer Magnitude. Für jedes trainierte Modell wurde die Differenz zwischen Random und Event R^2 berechnet. Im Durchschnitt verbessert sich die Performance auf Random-Daten um 0.614 ($\pm$0.390) R^2-Punkte. Diese Asymmetrie ist atypisch und deutet darauf hin, dass die Modelle stark an die Trainings-Datenverteilung gekoppelt sind, nicht an generalisierbare Domänenmuster. Für jedes trainierte Modell wurde die Differenz zwischen Random und Event R^2 berechnet. Im Durchschnitt verbessert sich die Performance auf Random-Daten um 0.614 ($\pm$0.390) R^2-Punkte. Diese Asymmetrie ist atypisch und deutet darauf hin, dass die Modelle stark an die Trainings-Datenverteilung gekoppelt sind, nicht an generalisierbare Domänenmuster.

Im Anschluss stellt sich die Frage nach den praktischen Implikationen dieser Generalisierungslücke. Die Event-Validierung ist der realistische Indikator für praktische Vorhersagefähigkeit. Die Random-Validierung gibt ein zu optimistisches Bild. Für praktischen Einsatz im Motorsport (neue Rennserien, neue Strecken) sind die aktuellen Modelle nicht zuverlässig. Die durchschnittliche Event-R^2 von -0.306 bedeutet, dass ein einfaches Baseline-Modell (z.B. Durchschnittswert der Zielvariable) besser abschneiden würde. Die Event-Validierung ist der realistische Indikator für praktische Vorhersagefähigkeit. Die Random-Validierung gibt ein zu optimistisches Bild. Für

praktischen Einsatz im Motorsport (neue Rennserien, neue Strecken) sind die aktuellen Modelle nicht zuverlässig. Die durchschnittliche Event-R^2 von -0.306 bedeutet, dass ein einfaches Baseline-Modell (z.B. Durchschnittswert der Zielvariable) besser abschneiden würde.

Um die Ursachen der Generalisierungslücke zu verstehen, folgt eine Ursprungsanalyse der unerfassten Kontextfaktoren. Die Generalisierungs-Lücke deutet auf folgende Ursachen hin: Die Generalisierungs-Lücke deutet auf folgende Ursachen hin:

- **Fahrereffekte:** Unterschiedliche Fahrer nutzen unterschiedliche Strategien und Fahrstile, was Understeer-Werte stark moduliert. Diese Variation ist in den Telemetriedaten nicht explizit erfasst.
- **Setup-Variationen:** Jedes Event und Fahrzeug hat unterschiedliche Basis-Setups, die nicht in den aggregierten Feature kaptiert sind.
- **Umgebungsfaktoren:** Temperatur, Luftfeuchtigkeit, Luftdichte sind nicht explizit im Datensatz erfasst.

Diese fehlenden Kontextfaktoren erklären, warum das Modell auf neue Events schlecht generalisiert: Es hat gelernt, die trainierten Datenmuster vorherzusagen, nicht die zugrundeliegende physikalische Beziehung zwischen Features und Zielvariable.

Zur Veranschaulichung der Ergebnisse bietet sich eine Boxplot-Übersicht an:

Abschließend lässt sich zur Generalisierungslücke festhalten: Das Modellportfolio weist eine kritische Abhängigkeit von der Datenverteilung auf. Während Random-Validierungsergebnisse auf solide Modellqualität hindeuten, sind die Event-Validierungsergebnisse ein Warnsignal: Das beste verfügbare Modell (LightGBM Very Deep, R^2 0.093) bleibt deutlich unter praktischer Brauchbarkeit, und der Durchschnitt (R^2 -0.306) ist nicht einsetzbar. Dies erfordert eine Neu-Bewertung des Ansatzes und der Datengrundlage, die in Abschnitt 5.7 (Synthese) erörtert wird.

6 Modell-Artefakt und Lessons Learned

6.1 Zusammenfassung der Modellperformance

6.2 Reflexion der Design-Entscheidungen

6.3 Einschränkungen und Verbesserungspotenziale

Wenig Daten

7 Fazit und Ausblick

7.1 Beantwortung der Forschungsfragen

7.2 Kritische Selbsteinschätzung

7.3 Empfehlungen für Folgestudien

Anhang

Anhangverzeichnis

Anhang 1: Rohtranskripte der Experteninterviews

Anhang 1/1: Erstes Meeting mit Performance-Ingenieur (29.08.2025)

Interviewer: Meeting mit Ingenieur, 29.08.2025 11 Uhr.

Interviewer: Ja, freut mich, vielen Dank für deine Zeit. Jetzt wollte ich dich fragen, ob du dich ganz kurz vorstellen könntest.

Ingenieur: So ja, kann ich kurz machen, ich bin schon länger bei Porsche. Teamleiter von der LMDH Performance Truppe. seit LMP Formel E. Jetzt LMDh, was wir machen, Performance, vor allem bei der Entwicklung, Simulationslastik, Kennwerte vorgeben für die anderen Abteilungen, auch, der Drag brauchen wir, wie viel Abtrieb brauchen auf dem Auto. Auch für die Reifenentwicklung mit Michelin zusammen, Rundenzeitberechnung machen wir, auch die Modellierung dazu. Und eben wenn des Auto mal fährt, Datenanalyse. Aber auch im Teil der performancerelevanten Software im Auto kommt auch von uns. Also, Funktionskontrolle.

Interviewer: Okay, sehr cool. Vielleicht stelle ich jetzt noch mal ganz kurz das vor, was wir vorhaben. Ich bin jetzt wie gesagt bei EMO 6 für nächsten drei Monate und muss von meiner Hochschule aus eine wissenschaftliche Arbeit schreiben über das Projekt, das ich mache. Und das Projekt, das ich mache, ist nämlich... Wir wollten, oder das wurde von, das kam von EM6 und von Paul, das Thema hoch, dass man vielleicht mit einem Proof of Concept starten möchte, um zu gucken, gibt es denn Auswertungen, Analysen, die man mittels der, also aus den Telemetriedaten ziehen kann, irgendwelche Aussagen, da rauskommen, die sich, auf die man ein Machine Learning Modell trainieren kann und einfach um mal zu gucken, ist es möglich, es da Daten und einfach mal ein Proof of Concept zu starten. Ja, genau. Das ist jetzt noch relativ am Anfang, deswegen als auch unser Termin. Ich bin neu im Motorsport, ich schaue auch mal zu Formel 1, aber das hilft mir dann jetzt in der Tiefe der Thematik nicht sehr viel weiter. ist eine falsche Serie. Falsche Serie, mich geht es jetzt grundsätzlich mal ganz kurz darum, bisschen das Verständnis zu bekommen. Was macht ein Performance-Ingenieur, also du hast es gerade schon mal kurz angeschnitten, vielleicht auch mal bisschen, also wirklich vielleicht so von oben kommen. Genau, was ist deine Jobrolle oder die Jobrolle des Performance Engineers und dann vielleicht auch eine Einordnung, wie denn sowas an einem Wochenende aussieht, wenn du das vorhin schon gemeint hast, also Datenanalyse, das ist ja da wahrscheinlich eher die Richtung, die für mich eher interessant wäre. Genau, also vielleicht könntest du mal bisschen so top down, mal bisschen grundlegend mal sagen, wie das so abläuft.

Ingenieur: Ich kann erst mal starten, es hilft. Performance-Ingenieur, Performance, die sind verantwortlich für die Performance vom Auto. Performance heißt, möglichst optimal das Auto auf der Strecke einzusetzen. Optimal bezüglich Rundenzeit logischerweise. Das ist ein Thema. Und das zweite ist, was sie noch machen ist, sie gucken auch, dass das Auto sicher betrieben wird. Das heißt, sie schauen in Betrieb, zum Beispiel in die Bremstemperatur. zu stark ansteigt und kritisch dann heben sie die Hand und sagen, wir haben da Problem. Diskutieren dann mit Renningenieur

zusammen, h olt das Auto rein und dann wird es repariert. sie gucken, z.B. wenn die Fahrhöhe vorne zu niedrig ist, anders als erwartet, dann muss man auch die Hand heben und sagen, da passiert irgendwas nicht. Oder wenn das Bremspedal zu lang wird, weil Verschleiß oh zu hoch ist. sind auch so sicherheitsrelevante Themen, was wir anschauen. Idealer Weise tritt das nicht auf. Dann ist Ihre Aufgabe halt Setuparbeit, am Auto, du kannst ja ganz viel einstellen, viel mehr als beim Straßenauto. Du kannst deine Federn, Steifigkeiten ändern. Da gibt es ganz viele Optionen. Es gibt verschiedene Aero-Konfigurationen. Die Fahrwerke kannst du einstellen. verschiedene Federkombinationen in Serie schalten, dass du verschiedene Charakteristiker hast und noch viel mehr. Das ist Ihre Aufgabe, das während des Rennwochenendes bis zum Rennen zu optimieren. Zusammen mit dem Engenieur und natürlich auch mit dem Fahrer. Der Fahrer fährt ja nicht nur das Auto, nur im Kreis. Sondern nach jeder Änderung kriegt man Feedback von dem Fahrer, ob das ins Gute oder ins Schlecht war. Aus seiner Sicht und das fließt in die Entscheidung ein, was man als nächstes ändert am Auto. Was man ändert am Auto, wie gesagt Fahrradfeedback ist wichtig und eben auch die Daten dazu. Das Telemetrie oder Kabeldaten. WEC hat leider sehr wenig Telemetrie, per Reglement ist das vorgegeben. IMSA hat da mehr, da hat man Probleme, mit der Logging rate. Im Endeffekt, man guckt an Daten, macht die auf, schaut sich die Balance an vom Auto. Balance heißt dass das Auto unter oder übersteuert. du schaust deine Fahrhöhen an, ob das Auto zu tief ist aufgesetzt oder zu hoch ist und du noch Potenzial hast tiefer zu gehen, je tiefer du bist, desto mehr Abtrieb hast du, was damit auch Rundenzeit bringt. Das ist so die Aufgabe von einem Performance Engineer. das Auto mit Renningenieur und Fahrer zu optimieren. Sie sind eine Verantwortung für das Setup vom Auto. Verstehe.

Interviewer: Danke für die Einführung. Das war jetzt auf jeden Fall schon mal sehr hilfreich. Kannst du mir noch einen kurzen Umriss geben, wie dann das Zusammenspiel an dem Wochenende vielleicht aussieht. Ich stelle mir das gerade aktuell so vor, man hat die Fahrt in Teilen links. Es werden Daten gesammelt, die Autos sind auf der Strecke, wie du schon beschrieben hast. Dann gibt es bei euch mehrere Performance Engineers, die sich die Daten angucken und wahrscheinlich auf verschiedene Dinge achten auf Basis ihrer Erfahrungen oder wie das Team eingeteilt ist.

Ingenieur: Der Performance Ingenieur, der hat Tools dazu auch, der simuliert, wenn ich jetzt die Feder ändere, dann ändert das in der Theorie, folgendes an meinem Auto, das ist tiefer, höher, ich sehe auch die Abtriebwerte theoretisch, das gibt mir alles mein Simulationsmodell her. Der gleicht das dann ab mit Streckendaten. Und je nachdem vom Feedback vom Fahrer, wenn der sagt, hey, ich zu viel Untersteuern, dann weiß der Performance-Engineer. Wenn ich an dem Parameter drehe, vom Auto, würde sich das ändern. . Dann würde sich die Verlauf vom Auto in die Richtung ändern. Normalerweise simuliert er das vorhandene Tool mit HH, also das ist unsere Datenbank. kann er sagen ich ende jetzt mein Setup, die Fahrhöhre, dann wird das simuliert und dann kriegt er die Werte aus. Das macht er nicht erst am Wochenende, wo er sich das frei überlegt. Der macht sich davor ganz viele Gedanken, der legt sich so einen Blumenstrauß an Setups zusammen, weil er diskutiert ist schon mit seinem Renningenieur. auch auf Erfahrung von den Jahren davor, was funktioniert hat, was sie da gefahren haben. Wir haben auch den

Fahrsimulator, das ist erste Tool. Auf dem Papier mit verschiedenen Setup Optionen. Du fährst dann am Fahrsimulaltor, du suchst jetzt schon mal aus, was gut funktioniert, was nicht so gut funktioniert, runter weg. Und mit den Optionen, Simulation ist eine Simulation, muss man noch validieren an der Strecke, gehst an die Strecke und fährst die eventuell dann einfach gegen. Validierst dann deine Theorie und nimmst dann das Beste ins Rennen mit, beziehungsweise ins Qualifying, z.B. um Unterschiede etc. Das ist der Prozess von der Performance , der definierte Setup. Da es die Mechaniker, die ich ausdrucken soll, die haben ja Tablet. Da sehen Sie, ich muss die Feder ändern, muss hier die Fahrhöhe anheben und dann stelle ich das am Auto ein. Da gibt es auch wieder von der Struktur einen Nr1 mechaniker. Der kriegt die Info und verteilt es dann an sein Team am Auto. Und dann wird das umgesetzt, was der Performance Engineer zusammen mit dem Renngenieur entschieden hat. Genau, dann wird das umgesetzt, dann fahren die, kriegst die Daten und kannst dann sozusagen interativ und etc. über dich finden.

Interviewer: Okay, ja, also danke, ja perfekt, danke für den Überblick. Das schärft jetzt ein bisschen mein Bild jetzt auch, weil ich halt eben neu bin in der Thematik, wie das generell funktioniert. Genau, aber dann würde ich jetzt mal ein bisschen erklären, was wir, also wie und was wir machen wollen. Also... Der Ansatz ist ziemlich Greenfield. haben dieses Konzept, wir mit Machine Learning, also Modell, auf Telemetriedaten trainieren wollen. Das ist die Baseline. Von hier aus können wir jetzt viele verschiedene Wege gehen. Ich muss nicht in irgendeine Richtung gehen. Ich kann verschiedene Dinge anwenden. Ganz grundlegend, was könnte man im Machine Learning gibt es dann verschiedene Teilbereiche. Der erste Teilbereich sind Klassifikationsmodelle. Man bekommt ja immer bei einem Machine Learning ganz viele Inputs, zum Beispiel in unserem Fall die verschiedensten Telemetrie-Channels. Das können 1, 2 bis 100. 100 sein, die man dort hineingibt und im Optimalfall kommt hinten ein Output aus. Und dieser Output kann je nachdem was man möchte, kann ja verschieden sein. Da gibt es zum einen das Klassifikationsmodell, das ist das erste, dann kommt hinten, man gibt seine Telemetriedaten rein und dann kommt zum Beispiel hinten Ich habe mir paar Beispiele ausgesucht auf Motorsport, unabhängig davon, ob das Sinn macht. Zum Verständnis ist, man z.B. in der TdMT-Daten von den Trainings reinhaut und hinten kommt dann eine Reifenmischung aus, was vielleicht Sinn macht, Soft, Medium, Hard. Also eine Klasse, Soft, Medium oder Hard. Sowas in die Richtung. Das ist ein Klassifikationsmodell. man sagt, Fahrverhalten klassifizieren. wie schnell wird Gaspedal, Bremspedal, wie schnell wird am Lenkrad gedreht und hinten kommt aus, dass es aggressives Fahrverhalten, mittleres Fahrverhalten oder ähm... optimales Fahrverhalten. Also in die Richtung kann man das machen. Man kann die Output natürlich selber definieren. wäre das erste. Das zweite, was eventuell interessant ist für das, was wir machen wollen, sind die Vorhersage- oder Regressionsmodelle. Das heißt, man trainiert das Modell mit vielen verschiedenen Telemetriedaten, die relevant sein könnten, um einen Wert, hätte, vorhersagen. Also zum Beispiel in der Rundenzeit vorhersage ich. gibt dem Modell ganz ganz viele verschiedene Telemetriedaten, es aktuell ist. Und dann fahre ich meinen ersten, zweiten Sektor, dritten Sektor und es wird immer aktualisiert, zum Beispiel die prognostizierte Endrundenzeit vorhergesagt zum Beispiel. Das wäre was für ein Regensohnsmodell. Also ich will kurz einfach vermitteln, wie das funktionieren könnte. Man mappt von ganz vielen verschiedenen Inputs auf einen Output. Wie der Output aussieht, können verschiedene

Sachen sein. die Sachen wären so die Hauptkategorien, die ich mich fokussieren würde. Also man kann einmal eine Klassifikation, also es kommt dann ein String aus, also kein numerischer Wert oder Regression, man versucht wirklich numerische Werte vorherzusagen auf Basis von anderen Parametern. Genau. Das ist so das, was es in der Theorie kann. Was für das Machine Learning Modell wichtig ist, ist, mit was für Daten es trainiert wird. Für das Taining braucht man beide Enden von dem. Man braucht den Input und den Output. Den Input, den Telemetriedaten, die ganzen Channels, speist man ein und gibt aber dann den wahren Output mit dem, was es sein soll, was hinten auskommen soll. Das Machine Learning Modell, lernt während des Tainings, wie es von den verschiedensten Inputs zu diesem einen Output mappt. ganz grobe Abriss, vielleicht dass du ein bisschen, ich weiß nicht genau wie technisch du vielleicht auch schon in der Thematik, aber das wollte ich noch mal ganz kurz abgerissen haben, dass du vielleicht ein bisschen einschätzen kannst in was für eine Richtung es gehen könnte. Also im Optimalfall wäre es für mich, also jetzt am Beispiel der Rundenzeit Vorhersage, da hatte ich mir schon ein bisschen reingeguckt. meine, das nennt sich Label, der Output ist ein Label für das Training und das ist ja relativ einfach zu bekommen dieses Label. Man nimmt einfach von jeder Lab die letzte Laptime. Dann hat man das Label und speist verschiedenste Parameter rein, die Geschwindigkeit des Autos, erste Runde, Sektor 2, zweite Sektorzeit, Bremsdruck. alle möglichen Channels könnt man reingeben und das Modell findet selber heraus während des Trainings wie wichtig welcher Parameter ist und wie die zusammenhängen um eben den beschriebenen Output B auszugeben am Ende. Genau das wäre jetzt Beispiel Rundenzeitvorhersage. Und da wäre es mir wichtig, weil wir jetzt noch keine hochkomplexe Sache machen möchten, dass man sagt, und weil ihr sehr viel beschäftigt seid, dass man eben diesen Output B auch bereits entweder berechnen kann oder näherungsweise berechnen kann oder bereits wie die Lap-Time aus den Daten ersichtlich sind. Verstehst du grob?

Ingenieur: Ja, also zu dem Thema Rundenzeit haben wir die Predigten laptime, schon im Auto, die sind ziemlich gut, die ist ein relativ Simpler Ansatz, also nichts mit machine learning. Gewisserweise merkt sich nur das Auto die Referenzrundenzeit, was der Fahrer aktuell gefahren ist und vergleicht die aktuelle Runde zu dieser Referenzrundenzeit. Es dann eben eine Predictive Post, also bin ich drunter. Ich habe es selber nicht geschrieben, es funktioniert aber ziemlich gut. Das heißt, du kommst auch ziemlich gut da rein, wo wo der Fahrrad tatsächlich dann landet. Also klar weißt du nicht, ob er jetzt in drei Kurven einen Fehler macht. Also das kann keiner vorher sagen. Das funktioniert ganz gut. Nur um es anzumerken, dass wir in der Richtung laufen, dass du was machst, was es schon gibt, gut funktioniert, da könnte man vielleicht was machen, was es noch nicht gibt. Die Sache ist, wir generieren sehr viele Daten, auch KPIs. Das Schwierige ist, das Ganze in kurzer Zeit, weil es ist ja wirklich nicht viel Zeit, um das zu verstehen, die Zusammenhänge zu verstehen. Genau, wenn ich jetzt mal ein Beispiel mache. Du fährst mit einem Auto, Temperatur ist x von der Strecke, die hat einen Einfluss auf das Verhalten vom Reifen, wie viel Grill du hast, wie viel Luftdruck der Regeningenieur gerade reingemacht hat, wie du gefahren bist mit dem Auto, wie du den Reifen aufgewärmt hast in den ersten fünf Runden, hat einen Effekt auf den Grip, du in Runde 20 haben wirst. Ob du aggressiv angefangen hast, hat einen Einfluss ob du länger schnell fahren kannst oder halt weniger. Es sind so schwierige

Entscheidungen oder auch, jetzt haben wir verschiedene Compounds Soft Medium Hart, welcher ist denn jetzt gerade der optimale? Temperatur bei 30 Grad oder bei 40 Grad und ich erwarte vielleicht, dass ich im Renn 50 Grad fahre, kann ich dann noch die Medium fahren oder muss ich den harten schon fahren, die eigentlich für die Bedingungen gedacht ist? Das sind so interessante Fragen. Es ist jetzt weniger mit Telemetrie, sondern eher, du sammelst das Wochenende über Daten und versuchst die dann zu interpretieren, möglichst schnell um die richtigen Entscheidungen vor dem Rennen zu treffen. Während man fährt, was immer ganz interessant fände, wenn die Fahrer sagen, sie hätten gerne mehr Feedback oder Guidance auch? und die Telemetrie haben. Beispiel im Auto, kannst so Sachen verstellen, das ist ja nicht so gegeben, wie du rumfährst, gerade bei den Systemen. Ja, deine Traktionskontrolle, also wie viel Moment kriegt der Fahrer für einen bestimmten Schlupf den du am Rad siehst. Auch mit dem Hinblick auch wieder Verschleiß. Was macht mein Auto in Runde 20? Wenn ich komplett viel Schlupf generiere, dann baut man bei es irgendwann ab. ist es geschickter am Anfang bisschen konservativer auch mit den Settings zu fahren, dass das Auto eben nicht permanent ausbricht an der Hinterachse. Und den Fahrer das als Hinweis schon mitzugeben. Auf Basis der Telemetriedaten zu sagen, geh mal mit deinem Setting bisschen konservativer, weil wir sehen in den Daten, du überfährst den Reifen gerade. Da gibt es auch verschiedene Kanäle, wie den Schlupf, man angucken kann, die Temperatur von den Reifen. selber sein Fahrstil, wie viel Energie bringt er mit seinem Fahrstil in die Reife Es gibt verschiedene auch Kanäle, wie gesagt, auf Telemetrie, die man an anschauen fand. Und entweder live, so machen wir es jetzt, weil wir so Erfahrung wollen, du guckst die Daten meistens ein halbes Jahr lang an und dann weißt wann du ungefähr, was in welcher Richtung, einstellen muss. So gibt man den Fahrrad Hinweise, ob er was ändern sollte oder nicht. Dann sagt er manchmal, hey voll gut, danke. Manchmal passt es halt nicht, aber dann muss man halt weiter lernen.

Interviewer: Das klingt sehr interessant. Man würde zum Beispiel aus ganz vielen verschiedenen Channels den Output generieren, überfahren, nicht überfahren oder so in die Richtung. Okay, ja das klingt auf jeden Fall nach einem sehr interessanten Thema, weil das ist eben auch genau so was, wo ich so den Hauptbenefit sehe. Ich meine, ihr könnt ja alle, ich meine, ich brauche ja jetzt kein Modell darauf trainieren, wenn du siehst, Tire Temperature größer 50, dann passiert irgendwas, ja. Genau, das macht ja keinen Sinn. Aber eben sowas wie du beschreibst, was man über eine längere Zeit... über eine längere Zeit einen Wert beobachten muss und dann auf Basis von Erfahrung keine... Also da stehen bestimmt intrinsische Regeln dahinter. Aber nichts, was der Mensch so ausdrücken könnte, in einem Code zum Beispiel, sondern eben wie du sagst, ist viel Erfahrung und Bauchgefühl von Jahren. Ich glaube, das wäre auf jeden Fall ein Gebiet, wo so ein Modell großen Potenzial hätte.

Ingenieur: Ich glaube, das würde helfen. Man muss ja gucken, dass das, glaube ich, deine Arbeit jetzt nicht so ein Konzept sein man versucht, die Weltformel zu generieren. sondern vielleicht mal was Simples, einen simplen Ansatz. Das ist meine Reifentemperatur, habe ich auf Telemetrie und das ist meine Balance vom Auto, habe ich auf Telemetrie und den Schlupf auch. Das einzige was mir bei dem Thema einfällt, das Thema, also das Training, die Trainingsdaten zu generieren. Da würden wir jetzt, also weiß ich... Vielleicht kennst du dann den Ansatz, wie man sagt, die

Daten müssen gelabelt werden. Ich habe alle Telemetrie-Daten, pro Sekunde oder zu 100 Hertz reinkommen.

Interviewer: Ich müsste markieren, hier diesem Zeitpunkt wurde der Reifen überfahren. Ich muss dieses überfahren Label irgendwie setzen, im besten Fall automatisiert. Das ist das Einzige, was wir bei dem Thema vielleicht bisschen ... Kopfzerbrechen oder was ein bisschen kompliziert ist. Ich könnte es jetzt ja, wie du sagst, als Basis auf Erfahrung. Ich kann jetzt ja gerade nicht einfach mir die Telemetät anzeigen und sagen, hier wurde der Reifen zu überfahren, hier, hier und hier. Das klingt nachher ein super cooles Thema, aber im besten Fall müssten wir es irgendwie schaffen, ihr mir das vermitteln könnt, dass ich diese Daten labeln kann. Weil sonst die Konsequenz wäre, wenn das nicht klappt, wie man es dann machen müsste, dass jemand, der diese Erfahrung im Kopf hat, ein Tool an die Hand und die Telemetriedaten markieren muss, hier markieren, überfahren, hier überfahren, hier überfahren und das am besten 1000 mal. Aber das ist natürlich für euch nicht praktikabel.

Ingenieur: Das gibt Kennwerte, die wir haben von hier sagen, zum Beispiel... Da gehen wir dann ein bisschen von der Telemetrie weg. Das sind dann wirklich Kennwerte. Das ist die Telemetrie und Kabeldaten. der Post-Programm-Testing. Und das ist dann die Integrale, mit laufenden irgendwelchen Mathefunktionen dahinter. Die Auswertung machen, da kommt dann eine Zahl raus. Zum Beispiel, wenn man sagt, man schaut sich Anzahl der Snaps an, in der Runde, wie häufig bricht das Heck aus, wenn der Fahrer ins Gas geht. Wenn der Reifen aufgeht, dann passiert das häufiger, als wenn der Reifen komplett neu ist. Das wäre zu sagen, ein Master für die ZIG-Datensätze, die ich habe. ich das als Eingang in den Fahrstil, Energie, was weiß ich. Und irgendwann beim Fahrer A habe ich mehr Snaps ab Runde 10 und bei Fahrer B ist es erst ab Runde 20. Ich weiß nicht ob. Auf Basis dessen könnte man jetzt hier überlegen Fahrer b, besser gemacht, ja. Weil bei dem kommt die Stab ein bisschen später. Ein anderes Kriterium, was eigentlich auch super simpel wäre, eigentlich könnte man mal sagen, Rundenzeit. Ich kann mir das mal deshalb ganz kürzlich erst einmal... Das ist einfach ein Long Run, das heißt wir fahren viele Runden von zwei verschiedenen Setups , gleicher Fahrer. Du siehst hier die Rundenzeit. Das Auto wird dann immer langsamer, je mehr Runden du fährst. Also kriegen wir weniger Grip. Hier sieht man, wo der Reifen dann wirklich abkackt. wird die Rundzeit dann langsamer. Also das kannst du in X-Richtungen verschieben.

Interviewer: Ah, ja. Das klingt cool. Warte mal, wenn ich mir überlege... Man gibt es sozusagen anhand von aktuellen Fahrverhalten aus, solange man in eine Lebensdauer prognostiziert. Also wenn du sagst nicht überfahren nicht überfahren sondern du sagst also so wie du jetzt in den letzten in den letzten zwei Sektoren oder der letzten Lab gefahren bist würde der Reifen jetzt noch drei halten und dann fährt er ein bisschen langsamer oder entspannter und dann dann springt der Wert hoch auf fünf oder sowas. Das hört sich gut an.

Ingenieur: Ja, das ist halt jetzt, hier waren zwei Setups, das kann ja verschiedene Gründe haben, warum das passiert. Das kann jetzt sein, wenn ich das nur die Rundenzeit dem Modell gebe, das weiß er nicht. Dann weiß erhalt nicht, waren das jetzt zwei Fahrer und der eine ist einfach

aggressiver gefahren, der andere ein bisschen schonender. Oder wie in dem Fall sind es zwei Setups, das war der gleiche Fahrer, der versucht gleich zu fahren. Oder waren das irgendwelche Einstellungen am Lenkrad, die er unterschiedlich gemacht hat. Ich glaube, muss man dem Modell auch mitgeben. War das jetzt ein anderer Fahrer, war das ein anderer Fahrstil? Da kommt das Thema wieder dazu. Wie viel Energie steckt der Fahrer in die Reifen? Wie fährt er? oder was sind die Einstellungen gewesen. Aber ich glaube, geht erstmal proff of concept. Da könnte man noch sagen, du nimmst mal Daten von Le Mans oder von irgendeinem Dauerlauf, wo das Setup einfach gleich geblieben ist. Du fährst einfach 24 Stunden lang. Dann kann man das schon mal aus xen, dass sich da irgendwas am Setup tut, dann ist es wirklich nur Fahrerunterschiede oder Fahrerunterschiede.

Interviewer: Das ist eine sehr gute Idee, glaube ich. Das wäre glaube ich echt mal für so Proof of Concept eine gute Idee, dass man erst mal abgekapselt, wie du sagst, nur für Lés mans, nur für ein Setup, guckt, wie sich das entwickelt. Das könnte klappen. Da muss ich mir angucken, wie die Daten, ich meine jetzt 24 Stunden, da kommt natürlich eine ordentliche Datenmenge herum, ob das erreicht für ein Training. Aber das ist auf jeden Fall schon mal eine gute Richtung, in die du mich da glaube ich schickst. Das ist auf jeden Fall eine gute Richtung. Da haben wir auf jeden Fall eine Kripp-Pasei. Wir haben auch in den Daten, was wir machen, auf Basis der Grabedaten trainieren wir ein Modell oder fitten ein Reifenmodell, wo wir dann wissen, der Reifen hat jetzt weniger Grip. Er baut dann ab. Du fittest jede Runde ein Reifenmodell. Und dein Krippparameter fällt dann ab, einfach damit das zur Runde passt. Und da wir ganz viele Ausgärtungen auch dazu. Das klingt nach einem guten Startpunkt für mich. Okay, also jetzt mit Blick auf die Uhr, weil du meinst, hast jetzt auch... Ja, genau. Ich würde jetzt auf jeden Fall mal das Hausaufgabe für mich mitnehmen. Das ist jetzt echt ein cooles Thema. Ich würde mich da ein bisschen einarbeiten, mir ein paar Gedanken dazu machen. Ich bin nächste Woche im Urlaub. Und dann würde ich dir vielleicht in zwei Wochen, falls du da da bist, einen Termin einstellen, wenn das okay wäre für dich. Dann bringe ich noch mal paar fische Gedanken rein. Du kannst mir dann noch da drauf bisschen Input geben. Wäre das okay für dich?

Ingenieur: Ja, ich glaube, gut, wenn du für das Meeting so bisschen aufmalst, wie du dir das vorstellst. Hier kommen Daten, da passiert das, da passiert das, das kommt raus.

Interviewer: Vielen Dank Fabian für deine Zeit. war wirklich sehr gut. Danke.

Anhang 1/2: Zweites Meeting mit Performance-Ingenieur (12.09.2025)

Interviewer: Zweites Meeting mit Ingenieur am 12.09.2025 um 10.30 Uhr.

Ingenieur: Hi, guten Morgen, grüß Dich.

Interviewer: Freut mich, dass wir es wieder schaffen, hier zusammen zu finden. Genau, gleich vorweg, ich habe auf jeden Fall die Woche ziemlich Gas gegeben und war die ganze Woche eigentlich, hatte ich mich mit dem ursprünglichen Thema beschäftigt, in die Richtung, du mich

gelenkt hast, was auch wirklich... ultra interessant ist und das war das das Reifendegrationsthema. Da habe ich mich die ganze Woche auch schon durch die Daten gewühlt und habe mir so bisschen auch schon eingeguckt wie man es machen könnte. Und grundlegend ist mir ein, größeres Problem aufgefallen und zwar gibt es eben wie ich vielleicht auch vorher schon mal gesagt hatte, das Thema, dass man das labeln muss. Also man muss sich irgendwelche Regeln, also euer Bauchgefühl, also wirklich greifbar machen und definieren, wo man Schwellwerte setzt von verschiedenen Parametern, die man dann erreicht oder so weiter, um dann zu sagen, ja jetzt hier der Reifen gerutscht, hier nicht gerutscht, dass man eben nicht manuell in die Daten reingehen muss und das wirklich manuell machen muss. Das ist ein größeres Hauptproblem und ich glaube, das hatte ich letztes Mal gar nicht erwähnt. Mein Projekt hier ist nur bis November und ich muss auch noch eine wissenschaftliche Arbeit in diesem Zeitraum bis November darüber schreiben. Das heißt, ich habe relativ wenig Zeit. Und dementsprechend war ich gestern recht glücklich, weil mir ist dann noch eine Idee gekommen. Ich habe nochmal darüber nachgedacht, was du gesagt hast. Und eine der Sachen, die du angesprochen hast, ist, dass die Fahrer oft sagen, dass sie gerne mehr Feedback hätten. Dass sie genauer oder noch mehr Feedback hätten. Und da ist mir eine Idee gekommen, kombiniert, wie man das gleich umsetzen könnte, ist, dass man ein Modell trainiert, um Fahrerfeedback zu generieren. Und die Idee, die ich dabei hatte, ist, Man nimmt die Telemetriedaten und man hat ja für jedes Outing schon in den Daten gesetzt welcher Fahrer gerade fährt. Und sozusagen man trainiert das Modell darauf auf verschiedene Fahrer und dann gibt es eine Technologie die nennt sich Explainability AI. Das heißt normalerweise kennt man es ja, dass solche AI Programme ziemliche Black Boxen sind. bringt irgendwas rein, die trainiert sie und dann gibt man Input und es kommt irgendwie ein Output raus. Wunder, Wunder. Aber da gibt es tatsächlich auch schon Ansätze, um eben genau diese Black Box aufzubrechen und zu gucken, wie kommt die KI zu der Entscheidung. Und hier kommt der Mehrwert ins Spiel. Meine Idee jetzt aktuell, wer man guckt, also man trainiert sie auf die verschiedenen Fahrer und kann zum Beispiel dann auch live gucken. In dem und dem Sektor war Fahrer 1, 2 oder 3 schneller und dann kann man wirklich relativ tief in die Daten eintauchen und schauen wo die Unterschiede bei den Fahrern liegen über die AI. Das war jetzt meine Idee. Das wäre aus einem aus dem großen, also für mich wäre es deutlich angenehmer umzusetzen. Aus dem Grund, ich die Labels, also welcher Fahrer gerade fährt, einfach aus den Daten ziehen kann. dementsprechend, also dieses Labeling-Thema ist bei so Maschinen-Learning-Projekten meistens immer der größte Painpoint und der größte Faktor, warum was scheitert. Und da sehe ich das jetzt eben auch eventuell bei dem iPhone-Digaktionsthema, weil es schon eben was Großes ist und wenn es so einfach wäre, dann wird es wahrscheinlich schon gemacht werden. Dementsprechend hätte ich jetzt...

Ingenieur: Ich kann dir kurz mal zeigen, was ich mir gedacht hätte. Okay. Und dann kannst du mir sagen, ob das so kompliziert ist. Okay, okay. Und dann können wir vielleicht noch mal in das andere einsteigen, weil ich hatte mir jetzt auch kurz vor dem Meeting noch mal fünf, zehn Minuten Zeit genommen. Das ist super nett. Ich mir paar Gedanken gemacht. Also, ja, geht ein bisschen in die Richtung, aber was mich halt, was halt echt interessant wäre... Oder was uns immer umtreibt ist, was macht die Fahrzeug Balance? ist das Auto eher neutral oder untersteuernd? Und das hängt halt ab von ganz vielen Parametern. Daher ist es für uns oder als Mensch relativ schwierig

zu verstehen, was jetzt was beeinflusst. So das Gleiche gilt für den Grip. Den kann man mal außen vor lassen. Deswegen haben wir gesagt, wir gucken mal erst mal Renndaten an, weil Renndaten, da kann ich die ganze Setup Arbeit, was der Renningenieur macht und was noch das Ganze noch viel komplizierter macht, erst mal außen vor lassen. Wenn man sich so bisschen vorstellt wie eine Gleichung, das wäre mein Y, die Fahrzeug Balance, die hängt halt von ganz vielen Eingangswerten X und dann gibt es hier eine Funktion, die wir nicht kennen. Und dann kommt da die Balance raus. Die Eingangsparameter, zum Beispiel die Streckentemperatur. Die Strecke jetzt, die messen wir ja. Wir haben eine Wetterstation, das gibt es in der Datenplattformen auf jeden Fall. Gibt es in unseren Workbooks. Die gibt es. Reifentemperatur haben wir, die messen wir. Reifendruck haben wir, messen wir auch. Wie du sagt Fahrer, wissen wir auch, welcher im Auto sitzt. Wir wissen die Knöpfe an denen der Fahrer dreht, was er einstellen kann. Der kann nämlich die Stabis verstellen. Da gibt es auch einfach Knöpfe 1, 2, 3, 4, 5. Recht diskrete Stufen. könnte auch direkt die mechanische Balance angucken, aber das geht vielleicht auch mit den Stabis. Weiß nicht ob man hier Aero-Balance braucht. Die ändert sich eigentlich nicht. Die wäre auch recht konstant. Reifenspec ähnlich wie Fahrer, wissen wir ob ein trocken Reifen oder ein Regenreifen am Auto ist und trocken Reifen wissen wir sogar welcher Gummimischung, ob es ein Soft, Medium oder Hard ist, stellt der Fahrer ein. Dann wichtig ist natürlich auch die Mileage. Wie viel Laufzeit hat der Reifen schon? Da bin ich mir unsicher, ob wir das schon in der Datenplattformen. Gibt es aber genauso schon als in HH. Genauso wie die Streckentemperatur müsste man die Mileage einlesen können. Jetzt weiß ich nicht, wer das macht. ob das ein Monin oder ein Paul machen kann. jeden Fall wäre das so was, wo ich sie irgendwann mal gerne hätte. Dass man die Mileage von dem Set hat. Dann Software. Da du natürlich auch einiges verstellen. Deine TC, also Traktionskontrolle oder deine Bremsblance. Aber das sind auch Knöpfe. Da dreht er dran. Also TC 1, 2, 3, 4, Auch diskrete Stufen. und dann natürlich der Sprit, den es gibt. Ist das Auto schwer oder leicht? Das waren für mich mal die Schnelle, die Prior 1 Faktoren. Hier können wir mal Aero ausklammern. Das bleibt hoffentlich auch konstant, weil wir ja Setup nicht ändern. So und von den Ausgangsgrößen, habe hier mal eine Runde aufgemacht, nicht eine Runde, sondern ein Workbook in Power BI. ist jetzt ein Auto Le Mans, komplette Rennen. Man kann es vielleicht ein bisschen anders darstellen. Wir haben ja schon Kennwerte. Dann siehst du hier, wir wollen die Balance ganz gut.

Interviewer: ist es das, was ihr euch live an der Strecke anguckt, das PowerBi Dashboard, oder habt ihr Wintax mit den genauen Telemetriedaten offen? Das sind ja die Metriken, oder?

Ingenieur: Beides, also Wintax sind wirklich deine Telemetriedaten als Datenstrom und dann haben wir halt Auswertung. kannst Gates definieren. das, du sagst, wo du die Thresholds brauchst. Und dann wo du sagst, du kannst das Cluster. Das machen wir hier. Okay, wir haben einen Cluster Entry, also Eingang der Kurve, Mitte der Kurve und Kurvenausgang. Und dazu die Balance. Und das für jede Runde gibt es dann einen Wert. Das ist ein Mittelwert über alle Kurven. Eingang. Mid-Corner und Exit und dann siehst du übers komplette Rennen, ihr Blau ist Kevin Estrid, dann haben wir Laurence und den Matt Campbell. Hier gibt es einen Wert. Der ändert sich schon ein bisschen das kann jetzt halt die Balance abhängig sein von Streckentemperatur

von ihren Settings was weiß ich hat sich auch hier während des stints ändert sich die Balance, das ist wahrscheinlich effekt von mileage oder auch fuel load das ist halt die Sache ich weiß es halt einfach nicht.

Interviewer: Was genau sagt die Balance? Wie das Gewicht verteilt ist über die Reifen? Ist das die Balance?

Ingenieur: Die Balance die die Fahrer beschreiben, die Fahrzeug Balance. Hast du viel untersteuern im Auto? Im Endeffekt, was mathematisch dahinter steckt. Du hast zwei Signale. Es ist vereinfacht. Zwei Signale. Den Lenkwinkel. Also wie viel lenkt der Fahrer? Und die Gearrate. Das Auto misst wie schnell sich das Auto dreht. Dann kannst du dir umrechnen, die Gierrate in einem Lenkwinkel. Wenn der Fahrer lenkt und das Auto reagiert sofort genauso wie der Fahrer lenkt, hast du ein sehr neutrales Auto. Indem du die Bewegung vom Auto mit dem Input vom Fahrer vergleichst. Wenn der Fahrer jetzt extrem viel lenken muss, aber das Auto bewegt sich gar nicht, typischen Straßenautos, dann hast du sehr viel Untersteuern. Also der Fahrer lenkt sehr viel. Aber das Auto dreht sich einfach nicht. Wenn hingegen, wenn du, wenn der Fahrer das Lenkrad ein bisschen bewegt und das Auto dreht sich sofort, dann hast du ein sehr übersteuerndes Auto. So und das ist mit Balance gemeint, mit Fahrzeug Balance. Wie viel sozusagen das Verhältnis von Input vom Fahrer und wie rotiert das Auto? Also wie verhält sich das Auto dann darauf? So kannst du es einteilen in, man kann es entweder ganz einfach klustern in, wenn du dann Delta rechnest, Gierrate und Lenkwinkel, wenn das Auto sozusagen mehr dreht als der Fahrer eingibt, kannst du einfach sagen übersteuernd. Wenn es geht sich genauso verhält, neutral, wenn du sehr viel Lenkwinkel brauchst, kannst du sagen untersteuernd. Kannst du sagen in die drei Gruppen. Oder was wir halt einfach hier ganz stumpf haben, ist den Wert sozusagen. Das müsste das Delta sein von wie viel rotierte das Auto und wie viel mehr muss der Fahrer lenken als Lenkwinkel. Einfach die Differenz zwischen beiden als Mittelwert pro Runde. Genau und das Interessante ist jetzt, das ist mein Y, aber ich verstehe nicht, warum geht das jetzt, warum reduziert sich die Balance, warum wird das Auto neutraler, also weniger untersteuernd. Woran liegt das? Liegt das jetzt dran, weil der Fahrer was verstellt hat oder weil sich die Streckentemperatur geändert hat oder weil der Reifen ja, recht viel Kilometer drauf hat oder weil der Tank leer ist und sich dadurch die Balance ein bisschen verändert. Weiß ich nicht. Das wäre halt was uns am Endeffekt interessiert oder mich die Frage, die ich mir ganz oft stelle, wenn man den Zusammenhang weiß. Ich weiß vor allem den Einfluss von dem Parameter von der Streckentemperatur auf meine Balance. Dann weiß ich halt schon, weißt du, du fährst dann in deinem FP1 am Morgen und bist eigentlich recht glücklich mit der Balance. So, dann fährst du dein FP2 am Nachmittag. Nur da ist die Strecke 20 Grad heißer. Der Fahrer sagt auf einmal, die Balance ist komplett daneben. So, dann fängst du wieder von vorne an. Stellst dein Auto wieder ein, dass der Fahrer glücklich ist. Denn Rennen ist aber dann vielleicht wieder zu deinen Temperaturen vom FP1 und dann passt es halt wieder nicht. Du bist eigentlich immer hinten dran und tunst hinterher. Aber wenn du wüsstest, okay, ich kenne ja die Wettervorhersage ich weiß, im FP2 bin ich vielleicht 20 Grad heißer. Sagt der Fahrer, die Balance ist scheiße, mach mal was. Dann kann ich ihm sagen, ja, das liegt an der Strecke. Keine Sorge, im Rennen wird es so und so warm. Das passt alles. Oder ich

kann es ihm einfach erklären auch und sagen, da ist nichts kaputt am Auto, liegt einfach an der Strecke. Die ist halt extrem viel heißer.

Interviewer: Verstehe. Genau, heißt also man hätte hier in dem Fall also das Labeling wieder zurück zum Thema. Hätte man das schon? Das ist genau der Graphie eigentlich oder nicht? Ingenieur: Genau. Das weiß ich jetzt nicht, ob es im Detail, also das musst du jetzt mir sagen, das weiß ich nicht, ob das die Arbeit schon einfacher macht.

Interviewer: Aber das ist ja Prinzip nochmal eine neue Problemstellung. Das hat erstmal nicht viel mit dem Reifendegrationsvorhersage zu tun, sondern es ist jetzt sozusagen... Also es ist ja ähnlich wie das andere mit dem Fahrer, bloß wir geben Inputvariablen und haben das Labeling Output und jetzt wollen wir verstehen, was dazwischen passiert.

Ingenieur: Genau.

Interviewer: Okay. Das ist nochmal eine ganz andere Fragestellung.

Ingenieur: Es geht halt auch um das Richtung Reif. Das Thema mit dem Verschleiß, den Kennwert, den haben wir noch nicht. Den werden wir auch nicht vor November haben. Da habe ich mit den Kollegen nochmal gesprochen. Also bringt ja da nichts.

Interviewer: Was soll da kommen? Ich glaube, das habe ich jetzt noch gar nicht gehört.

Ingenieur: Wir wollten ja mit der Degradation was machen. Wie baut der Reifen ab? So, und da muss ein Kennwert berechnet werden, wie hier, mit der Balance. Den gibt's aber noch nicht. Der Plan war, dass wir den vor zwei Wochen implementieren. Das kriegen sie aber nicht hin. Bis Ende November. Scheinbar komplizierter. Die wollen sagen, der Reifen ist jetzt so so viel degradiert, Genau. am Endeffekt ist, was dahinter steckt, es gibt ein Reifenmodell. Du modellierst einen Reifen mit mechanischem Verhalten über Kennlinien, über Parameter. Wir wissen, wie er sich neu verhält, wie die Parameter ausschauen. dann kannst du jede Runde hast du die Messdaten und optimierst die Parameter so, dass dieses Modell dazu passt. Wenn das Sinn macht. Du hast sozusagen Optimierer, der die Parameter tunen, damit mein Modell zu dem Messdaten passt. Und diese Parameter, die beschreiben zum Beispiel den Grip. Das ist ganz einfach. Ich kann es auch ganz einfach machen. Maximale Querbeschleunigung beschreibt meinen Grip. So, wenn ich jetzt weniger Grip habe, dann geht meine maximale Querbeschleunigung runter. Das wäre ein ganz einfacher Ansatz. Du guckst ja per Runde an, was meine maximale Querbeschleunigung habe ein Modell. So Querbeschleunigung ist ein bisschen wie mechanisch mit einem Reibmodell. hast eine Last Fz. Ich kann meinen Fy berechnen. ich messe meinen Fz. Das weiß ich im Auto. Das messe ich. Ich weiß meinen Fy. Das ist meine Querkraft, meine Querbeschleunigung. Und dann kann ich einfach ausreden, was ist denn mein Reibwert, mein myh. Mit der FZ und dem Mühe kriege ich diese Seitenkraft. Und so kann ich mir das ausreden. Das ist sehr, sehr, sehr, sehr, sehr vereinfacht. Da haben wir ein komplexeres Modell, aber so funktioniert das. Das wollen wir in die Datenplattform reinbringen. Diesen Prozess, den gibt es schon, aber erst einmal nur in Matlab. recht viel händisch, Daten runterladen, konvertieren, Modell identifizieren. Und dann gibt es dann eine Routine und die wollen wir halt in die Datenplattform bringen. Das dauert

aber noch bisschen scheinbar aufwendiger als gedacht, würde ich mal vermuten, ohne dass von den Kollegen gehört zu haben. Aber sonst wäre es ja schon drin. Das heißt also davon bin ich ausgegangen. Das heißt vor dem Urlaub hatte ich Info, dass wir das rein kriegen. Also in der Woche, wo du im Urlaub bist, dann hätten wir das jetzt schon, aber das gibt es nicht. Die haben gesagt, das wird nichts bis Ende November. Das bringt dir erstmal nichts.

Interviewer: Okay, das ist schon mal interessant. gut. Also wenn es das gäbe, das würde natürlich das Reifendegrationsprojekt vereinfachen.

Ingenieur: Ja, aber im Endeffekt ist es nichts anderes wie das hier, weil es ist einfach nur ein anderer Parameter, anderes Y dann. Also du willst ja so eine Art Prinzipstudie machen. Und mein Verständnis, wir sagen können, hey, das funktioniert prinzipiell der Workflow, dann kann ich ja alles Mögliche reinwerfen und rauskriegen.

Interviewer: Ja, also wo man die genau weiß, wie die Einflussgrößen das Outcome bestimmen. Genau. Okay, dann können wir noch mal ganz kurz ein bisschen tiefer einsteigen in die Car Balance, in das Thema hier. Genau, also wir haben jetzt hier unser Zielvariable, die Car-Balance, die ist gespeichert und definiert und die können wir angucken. Ingenieur: Die wird gemessen.

Interviewer: Genau, du hattest jetzt gerade noch das andere Notebook offen, wo du schon mal versucht hast, die Inputparameter zu bestimmen.

Ingenieur: Also hier, ist dein Y, das sind genau die drei Graphen. Und als Eingang, das kann man auch anschauen, das kann man auch sagen, die sind ein bisschen verstreut leider. in der Darstellung. Zum Beispiel Reifen. Das wäre zum Beispiel eine Eingangsgröße. Das ist der Front, also vordere Anti-Roll Bar also dein Stabilisator. Vorne die Einstellung vom Fahrer, was er ausgewählt hat. jetzt wieder hier hingeht, wäre der Punkt hier. Damit kann er die Balance im Auto verstellen. Dann haben wir hier die TC-Settings. Das ist ein Software-Ding. Da sind wir hier auf. weiß nicht, ob es jetzt angezeigt wird. Dem Wert hat dann ist er nach oben gegangen, wieder runter, recht wenig verstellt. Das wäre das zweite Setting. Ja, man die Bremsen auch wieder Software, die Bremsbalance, wie er es verstellt hat. Dann haben wir noch zwei weitere für die Bremsbalance. Also eigentlich die drei Parameter Brems-Balance. Das wäre also sag mal, mit dem Block hier, hätte man eigentlich schon. Das hier wären die Outputs. Ja, so jetzt kann man es noch ein bisschen Clustern . Ich weiß nicht, ob man so kompliziert schon werden muss. Das ist noch mal ein bisschen anderes Gating, wenn man das also das ist ja gesagt die Balance, ich gesagt hatte erst mal. Eingang Mitte Exit als Mittelwert über die Runde, dann kannst du es noch ein bisschen komplizierter machen, indem du noch mal die Fahrzeug Geschwindigkeit unterscheidest in High Speed, Medium Speed, Low Speed.

Interviewer: Also würde zum Beispiel für einen POC, also ich bräuchte ja eigentlich eine Zielvariable, ich ja gucken, zum Beispiel Highspeed, Midcorner, oder?

Ingenieur: Ja, oder du schaust einfach mal Midcorner über die Runde an, ob sich da irgendwas tut. So, hier hätten wir schon die Temperaturen. Da bin ich jetzt nicht sicher, welche wir davon messen im Auto die Oberfläche vom Reifen, aber wir messen auch...

Interviewer: Aber da würdest du auch sagen, dass es Sinn macht, zum Beispiel das jetzt nur auf die diesjährigen Les Mans Daten von einem Auto zu trainieren, oder? Weil sich dann zum Beispiel nicht so Sachen, die wir jetzt nicht abbilden können, so Setup-Changes, die eben...

Ingenieur: Ich würde es pro Auto machen, weil die Setups unterschiedlich sind. Jetzt kannst du aber sagen, ich trainiere dreimal ein Modell, nämlich zu jedem Auto. Oder du könntest sagen, weil dieses das Ganze sollte universell ja gelten. Sozusagen wenn man dann in Differenzen oder Delta das überlegt, wenn ich sage, meine Strecke wird 10 Grad heißer, dann sollte sich sollte das Auto übersteuernder werden. Das sagt mein Training vom Auto 6. Dann könnte ich gucken, hey, jetzt schaue ich mal, wie sich die Streckentemperatur vom Auto 5 geändert hat. Mach sozusagen Replay von meinem Modell und schau, wie gut passt mein Modell zu dem, was ich vorhersagen würde, wie gut passt es zu dem, was Auto 5 gemacht hat. Sozusagen Validierungsschleife dann oder was mein Auto 4 gemacht hat. Man könnte auch sagen, ich schmeiße alle Daten in einen Topf und schaue, ob es generelle Regeln gibt. Ja, die Setups sind unterschiedlich, aber die Balanceänderung zu meinem Setup sollte ähnlich sein. Ich ändere das nicht im Rennen, das Setup von allen drei Autos. Ich glaube, man könnte erst mal sagen, du hast ja echt viele Daten. Man startet mal mit einem Auto. Ich glaube, das ist dann auch nicht viel Stress, wenn man sagt, man schmeißt dann alle drei Autos mal rein oder wie gesagt, spannend wäre ja zu sagen, ich drehe mir mal ein Auto und gucke, wie gut passt meine Vorhersage zu den anderen Autos. Das ist jetzt die Reifentemperatur, vorne links, vorne rechts, hinten links, hinten rechts. Das sind ist Durchschnittswerte pro Runde.

Interviewer: Ok, verstehe. Ja das ist doch auf jeden Fall auch schon mal cool. Weil diese Durchschnitte würde man eh berechnen. Das heißt, wäre vielleicht sogar was, was ich mir direkt runterziehen könnte, ohne große Vorverarbeitung. Einfach versuchen könnte, das zu trainieren. Das Einzige, was mir hier auffällt, es gibt so ein paar Ausreißer. Sind es Fehlmessungen hier?

Ingenieur: Nee, das kann es sein. Zum Beispiel, wenn du einen Safety Car hast, dann ... steckst nicht viel Energie in die Reifen, weil fährst nicht schnell um die Kurve, dann geht halt die Temperatur runter. Oder beim Reifenwechsel neue Reifen, wir fahren ja nicht mit Heizdecken oder mit warmen Reifen los, so wie in Formel 1, die sind ja von kalt und dann werden die mit der Zeit dann wärmer, bis es sich irgendwann stabilisiert haben. Verstehe. Das passiert hier immer. Reifenwechsel, Start ist von kalt, dann wird er wieder heiß. Dann Boxenstopp wird aufgetankt, kühlt ein bisschen ab. Oder hier waren Safety Car, nein wahrscheinlich Boxenstopp. Kühlt ein bisschen ab, dann wird er wieder heiß. Dann haben wir wieder Reifen gewechselt. Genau, geht dann wieder runter hier. Wahrscheinlich haben sie ein bisschen Energie gespart oder waren Safety Car oder eine Slowzone. Ich weiß es nicht, da müsste man jetzt ins Detail gucken. Aber das sind die Schwankungen hier sind normalerweise irgendwie Safety Car oder Reifenwechsel. Warum das hier nach oben abrauscht, das kann ich dir nicht gar nicht sagen. Vielleicht ist da irgendein Sensor ausgefallen. Diese Surface, die sind immer ein bisschen, die gehen öfters mal kaputt. Okay. Die sind halt, das sind so Infrarot Sensoren. Die sind im Radkasten, auf dem Reifen und wenn da ein Gummifurzel abfliegt, an diesen Sensor ran. Dann kann es sein, dass der kaputt geht. Vielleicht nimmst du da einfach diese Inner-Liner. Das sind Sensoren, die vom Rad auf den Gummi gucken. Das geht nicht so schnell kaputt. Weil hier hat man auch so ein paar Ausreise. Da kannst du

sagen, alles was Also 300 Grad, da schmilzt der Reifen. Das lässt sich mit dem Filter dann echt einfach.

Interviewer: Du sagst, also wir kriegen eigentlich alle Daten, du jetzt in deinem OneNote hast, kriegen wir hier als Metric irgendwo her, oder?

Ingenieur: Wir haben hier alle Sachen, außer die Mileage. Aber die wäre schon wichtig. Das müsste man Paul fragen, dass sie das noch integrieren. wäre auch einfach ein Wert pro Runde. Wenn es dir einfach machen willst, kannst du sagen, du nimmst die Anzahl der Runden. Dann weißt du das einzige, du halt da nicht weißt ob das ein neuer ist, der drauf gekommen ist oder ob der schon drauf war.

Interviewer: Wo liegt das an? Also es liegt ja aktuell ab, hast du gesagt, irgendwo,

Ingenieur: Ja, zum Beispiel, also ich hab hier grad HH, das ist unsere Datenbank Software. Die sind schon verlinkt. Also das Workbook. Und die Software, du siehst hier zum Beispiel ist Tracktemperature die Zieht sich die Datenplattform von dem Tool hier von HH. Genauso wie die ganze Setup-Information, steht da steht alles hier drin. So, wenn man jetzt mal guckt. Mein Gedanke wäre, den Workflow gibt es schon. Jetzt muss nur einer sagen, ich integriert anstatt Streckentemperatur die Reifenmilage, weil ich die Kilometer vom Reifen hier schon habe und mache da einen Metric, einen unten Endwert oder Startwert, ist dann auch nicht so dramatisch und nehme das dann als Kernwert.

Interviewer: Könntest du, würdest du dir was ausmachen, wenn du deine OneNote-Page shares?

Ingenieur: Ja, die kann ich dir dann schicken.

Interviewer: Danke schön. Ja, aber genau, das ist dann auch ein Thema, ich Paul gebe. Und ansonsten, es wäre praktisch, ich alle Input- und alle Output-Daten an einem Ort habe, dann trainiere ich und Es ist auch wieder genau, dann geht es wieder zurück zu diesem Explainability. Ich muss mal gucken, inwiefern, wie genau ich sagen kann. was auf jeden Fall geht, ist, ich kann schauen, welche Einflussgröße, wie stark beeinflusst und so weiter. Aber das geht auch relativ tief.

Ingenieur: Das hat jetzt halt nichts mehr mit Rohdaten zu tun, weil wir sind ja mal gekommen von den ersten Ideen auf Telemetrie-Daten was zu machen. Das hat dann halt nichts mehr mit Roh-Telemetrie-Daten zu tun, sondern nur noch mit Process-Daten, die schon aufbereitet wurden gewisserweise.

Interviewer: Das ist okay.

Ingenieur: So, jetzt kann man mal gucken hier. Das ist eine grafische Darstellung. Hier siehst du, da sind neue Reifen drauf gekommen. Hier wurden die dann, das ist keine vier Punkte, die wurden dann drauf gelassen. Erster Run, logischerweise neue Reifen. Man sieht dann hier, das sind Anzahl der Runden, aber das wäre vielleicht auch ausreichend. Weil es gibt diese Zahl, du siehst hier der Reifen. in dem Outing war von 0 Runden bis 12. Wenn ich jetzt die nächste Runde

anschaue, dann siehst du, der startet bei 12 und geht dann bis 25. Das heißt, ist frisch drauf gekommen von frischen Reifen. Also hier ganz am Anfang frisch, genau. Aber der ist dann nicht mehr frisch, der hat schon 12 Runden. Und der nächste, hat dann, siehst du ja, der startet bei 25 Runden und den hat in Summe 37 drauf gehabt. Dann haben wir Fahrer gewechselt. Der Reifen hat schon eine Runde drauf gehabt. Der ist dann weitergefahren, nicht getauscht, hat in Summe 40 Runden gefahren. Dann Fahrerwechsel, frische Reifen, startet wieder bei Null. Der ist dann 13 Runden gefahren und so weiter. mein Punkt ist, es gibt hier schon einen... Parameter Signal das dir sagt wie viele Runden hat der Reifen.

Interviewer: Also hast du so ein Wert ab wann von deinem Bauchgefühl ist die Peak Performance von dem Reifen wann und wie viele Runden lässt die nach?

Ingenieur: Ja, also Peak sind die ersten acht Runden, wenn sie pushen. Aber oft fahren sie die, also wenn sie es geschickt machen, nutzen sie die auch nicht. Weil wenn du wirklich die Peak Performance nutzt, dann machst du den Reifen ein bisschen kaputt. Das tut dir dann später eher weh. Weil er der dann schneller abbaut. Das ist ein anderes Thema. Aber ja, siehst, dass der Peak irgendwo hier im ersten Run. dann muss man gucken, der baut dann schon ab. Auto wird dann auch langsamer.

Interviewer: Allgemein gibt es mehrere Themen für euch, wo ihr sagt, wir haben hier eine metric. Wir wissen auch wahrscheinlich welche Eingangsvariablen es gibt, aber wir wissen nicht genau, wie die zusammenhängen, um dieses Ergebnis zu produzieren, oder? Also das gäbe es für Degradation, jetzt eben für das Balance Thema und so weiter. Also da gibt es so ein paar Sachen, wo das so ist, oder?

Ingenieur: Ja, schon. Also es gibt verschiedene Ansätze. Du kannst natürlich sagen, ich bilde mir ein Modell von dem Reifen, nehme die ganzen Daten und fit das. Und dann weiß ich auch, wenn ich an der Streckentemperatur, Dreh, weil sich auch was passiert. Aber das ist auch extrem viel Aufwand, dann Modell zu generieren, was halt passt in allen Bedingungen. Oder der andere Ansatz wäre halt einfach jetzt. Ich habe unendlich viele Daten. Wir haben ja extrem viele Daten in dem Projekt. Und ich nehme einfach die Daten, wie sie sind, und versuche da die Zusammenhänge zu verstehen. Und da gibt es viele Sachen, wie jetzt also das Thema mit der Balance oder wie gesagt mit der. Degradation, wenn wir das mal drin haben. Da kann man sich unzählige Sachen vorstellen. Beispiel gibt es auch irgendeine Abhängigkeit für meine Aerobalance, die sich über die Laufzeit Da kann man kreativ werden. Deswegen, wenn man dann einen Prozess hat, wo man sagt, das ist mein X und das ist mein Y, ich trainiere das. kann dann vorhersagen, was passiert mit meinem Y, wenn ich an X ein was ändere. Das wäre halt schon echt hilfreich.

Interviewer: Dann danke für das Gespräch, ich glaube es hat sehr geholfen.

Methodische Anmerkungen zu den Interviews

Die Interviews wurden als semi-strukturierte Experteninterviews geführt und digital aufgezeichnet. Die vorliegenden Transkripte sind Rohtranskripte, die zur besseren Lesbarkeit minimal ge-

glättet wurden, jedoch den originalen Gesprächsinhalt und -verlauf authentisch wiedergeben.

Die Gespräche dienten der:

- Anforderungsanalyse für das Machine Learning Projekt
- Identifikation relevanter Telemetriedaten und Kenngrößen
- Bewertung verschiedener Ansätze (Reifendegradation vs. Car Balance)
- Klärung technischer Umsetzbarkeit und Datenverfügbarkeit

Anhang 2: Modell-Ergebnisse

Anhang 2/1: Ergebnisse der Event-Validierung

Modelltyp,Parametertiefe,Kategorisch,Aggregate,Smoothed,MAE,RMSE,R2
xgb,shallow,True,True,0,0.272,0.344,-0.443
lgb,shallow,True,True,0,0.25,0.318,-0.239
xgb,deep,False,True,4,0.259,0.33,-0.332
lgb,deep,False,True,4,0.316,0.394,-0.895
xgb,deep,True,False,2,0.242,0.308,-0.162
lgb,deep,True,False,2,0.222,0.284,0.012
xgb,medium,False,True,3,0.277,0.344,-0.443
lgb,medium,False,True,3,0.267,0.335,-0.372
xgb,shallow,False,False,2,0.307,0.38,-0.761
lgb,shallow,False,False,2,0.256,0.33,-0.327
xgb,very-deep,False,True,0,0.267,0.335,-0.372
lgb,very-deep,False,True,0,0.296,0.365,-0.625
xgb,medium,True,False,2,0.259,0.325,-0.291
lgb,medium,True,False,2,0.216,0.28,0.04
xgb,deep,True,True,3,0.23,0.297,-0.074
lgb,deep,True,True,3,0.237,0.303,-0.125
xgb,deep,True,True,2,0.226,0.292,-0.042
lgb,deep,True,True,2,0.235,0.301,-0.108
xgb,medium,False,False,3,0.266,0.336,-0.379
lgb,medium,False,False,3,0.309,0.384,-0.804
xgb,shallow,False,True,4,0.298,0.368,-0.656
lgb,shallow,False,True,4,0.247,0.314,-0.206
xgb,shallow,False,False,0,0.325,0.393,-0.884
lgb,shallow,False,False,0,0.325,0.394,-0.899
xgb,shallow,True,True,4,0.25,0.313,-0.196
lgb,shallow,True,True,4,0.231,0.3,-0.1

xgb,very-deep,False,False,3,0.259,0.328,-0.315
lgb,very-deep,False,False,3,0.258,0.329,-0.322
xgb,deep,True,False,3,0.24,0.307,-0.155
lgb,deep,True,False,3,0.223,0.288,-0.013
xgb,very-deep,True,False,3,0.239,0.306,-0.142
lgb,very-deep,True,False,3,0.21,0.274,0.081
xgb,shallow,False,False,3,0.29,0.362,-0.597
lgb,shallow,False,False,3,0.267,0.342,-0.433
xgb,shallow,True,False,2,0.267,0.345,-0.456
lgb,shallow,True,False,2,0.212,0.276,0.067
xgb,medium,True,False,4,0.287,0.36,-0.584
lgb,medium,True,False,4,0.219,0.281,0.038
xgb,very-deep,True,False,4,0.246,0.314,-0.204
lgb,very-deep,True,False,4,0.22,0.283,0.02
xgb,deep,False,True,2,0.279,0.346,-0.467
lgb,deep,False,True,2,0.296,0.37,-0.672
xgb,shallow,True,False,0,0.34,0.418,-1.135
lgb,shallow,True,False,0,0.211,0.28,0.045
xgb,deep,True,False,4,0.262,0.334,-0.366
lgb,deep,True,False,4,0.223,0.288,-0.01
xgb,very-deep,False,False,4,0.253,0.324,-0.282
lgb,very-deep,False,False,4,0.282,0.354,-0.529
xgb,deep,False,True,0,0.263,0.33,-0.334
lgb,deep,False,True,0,0.299,0.366,-0.638
xgb,medium,False,True,0,0.273,0.341,-0.425
lgb,medium,False,True,0,0.322,0.394,-0.895
xgb,deep,False,True,3,0.266,0.335,-0.373
lgb,deep,False,True,3,0.262,0.33,-0.329
xgb,shallow,False,True,0,0.285,0.356,-0.551
lgb,shallow,False,True,0,0.249,0.316,-0.219
xgb,medium,True,True,0,0.239,0.308,-0.163
lgb,medium,True,True,0,0.238,0.303,-0.119
xgb,shallow,True,False,3,0.248,0.317,-0.225
lgb,shallow,True,False,3,0.218,0.282,0.032
xgb,very-deep,True,False,0,0.261,0.327,-0.305
lgb,very-deep,True,False,0,0.212,0.272,0.093
xgb,deep,True,True,4,0.23,0.295,-0.065
lgb,deep,True,True,4,0.241,0.308,-0.162
xgb,very-deep,False,False,2,0.252,0.321,-0.258
lgb,very-deep,False,False,2,0.284,0.363,-0.606
xgb,medium,True,True,3,0.234,0.3,-0.099

lgb,medium,True,True,3,0.234,0.298,-0.088
xgb,very-deep,True,False,2,0.245,0.312,-0.186
lgb,very-deep,True,False,2,0.221,0.286,0.002
xgb,deep,False,False,0,0.257,0.326,-0.297
lgb,deep,False,False,0,0.265,0.338,-0.394
xgb,shallow,False,True,2,0.325,0.397,-0.928
lgb,shallow,False,True,2,0.295,0.365,-0.628
xgb,medium,True,False,3,0.269,0.34,-0.411
lgb,medium,True,False,3,0.243,0.313,-0.2
xgb,very-deep,False,True,2,0.282,0.349,-0.492
lgb,very-deep,False,True,2,0.268,0.338,-0.395
xgb,deep,True,True,0,0.222,0.287,-0.009
lgb,deep,True,True,0,0.22,0.284,0.013
xgb,medium,False,False,0,0.263,0.336,-0.377
lgb,medium,False,False,0,0.269,0.34,-0.416
xgb,medium,False,True,2,0.286,0.353,-0.526
lgb,medium,False,True,2,0.268,0.337,-0.389
xgb,deep,False,False,4,0.251,0.321,-0.259
lgb,deep,False,False,4,0.283,0.356,-0.55
xgb,shallow,True,True,2,0.248,0.316,-0.216
lgb,shallow,True,True,2,0.218,0.285,0.008
xgb,shallow,False,False,4,0.28,0.35,-0.496
lgb,shallow,False,False,4,0.273,0.348,-0.476
xgb,very-deep,False,True,4,0.273,0.341,-0.421
lgb,very-deep,False,True,4,0.277,0.345,-0.458
xgb,shallow,True,True,3,0.287,0.363,-0.61
lgb,shallow,True,True,3,0.222,0.285,0.007
xgb,shallow,False,True,3,0.297,0.365,-0.625
lgb,shallow,False,True,3,0.255,0.322,-0.264
xgb,medium,False,False,2,0.263,0.334,-0.361
lgb,medium,False,False,2,0.265,0.339,-0.406
xgb,very-deep,True,True,3,0.224,0.29,-0.026
lgb,very-deep,True,True,3,0.227,0.296,-0.072
xgb,medium,True,True,4,0.241,0.311,-0.182
lgb,medium,True,True,4,0.227,0.297,-0.078
xgb,deep,True,False,0,0.252,0.317,-0.229
lgb,deep,True,False,0,0.216,0.276,0.069
xgb,deep,False,False,2,0.243,0.31,-0.175
lgb,deep,False,False,2,0.272,0.348,-0.483
xgb,medium,False,True,4,0.267,0.335,-0.373
lgb,medium,False,True,4,0.289,0.36,-0.583

xgb,very-deep,False,True,3,0.266,0.335,-0.374
lgb,very-deep,False,True,3,0.261,0.328,-0.317
xgb,very-deep,True,True,4,0.229,0.295,-0.062
lgb,very-deep,True,True,4,0.243,0.309,-0.164
xgb,very-deep,True,True,0,0.217,0.282,0.029
lgb,very-deep,True,True,0,0.217,0.282,0.028
xgb,medium,False,False,4,0.264,0.334,-0.363
lgb,medium,False,False,4,0.285,0.356,-0.551
xgb,medium,True,True,2,0.22,0.283,0.02
lgb,medium,True,True,2,0.244,0.31,-0.174
xgb,shallow,True,False,4,0.273,0.35,-0.497
lgb,shallow,True,False,4,0.22,0.293,-0.052
xgb,very-deep,True,True,2,0.227,0.291,-0.037
lgb,very-deep,True,True,2,0.228,0.296,-0.073
xgb,medium,True,False,0,0.261,0.328,-0.316
lgb,medium,True,False,0,0.214,0.274,0.084
xgb,deep,False,False,3,0.259,0.328,-0.317
lgb,deep,False,False,3,0.286,0.362,-0.598
xgb,very-deep,False,False,0,0.255,0.325,-0.289
lgb,very-deep,False,False,0,0.262,0.334,-0.362

Anhang 2/2: Ergebnisse der Zufalls-Validierung

Modelltyp,Parametertiefe,Kategorisch,Aggregate,Smoothed,MAE,RMSE,R2
xgb,shallow,True,True,0,0.285,0.358,0.193
lgb,shallow,True,True,0,0.319,0.399,-0.0
xgb,deep,False,True,4,0.186,0.243,0.628
lgb,deep,False,True,4,0.269,0.336,0.292
xgb,deep,True,False,2,0.276,0.352,0.219
lgb,deep,True,False,2,0.299,0.373,0.126
xgb,medium,False,True,3,0.177,0.234,0.654
lgb,medium,False,True,3,0.257,0.321,0.352
xgb,shallow,False,False,2,0.218,0.283,0.496
lgb,shallow,False,False,2,0.27,0.334,0.3
xgb,very-deep,False,True,0,0.176,0.234,0.657
lgb,very-deep,False,True,0,0.245,0.311,0.391
xgb,medium,True,False,2,0.272,0.348,0.238
lgb,medium,True,False,2,0.291,0.366,0.156
xgb,deep,True,True,3,0.278,0.353,0.216
lgb,deep,True,True,3,0.284,0.354,0.213
xgb,deep,True,True,2,0.294,0.366,0.158

lgb,deep,True,True,2,0.288,0.36,0.184
xgb,medium,False,False,3,0.19,0.251,0.603
lgb,medium,False,False,3,0.251,0.316,0.373
xgb,shallow,False,True,4,0.198,0.257,0.584
lgb,shallow,False,True,4,0.267,0.33,0.316
xgb,shallow,False,False,0,0.225,0.291,0.468
lgb,shallow,False,False,0,0.268,0.339,0.278
xgb,shallow,True,True,4,0.333,0.407,-0.043
lgb,shallow,True,True,4,0.298,0.375,0.117
xgb,very-deep,False,False,3,0.184,0.244,0.625
lgb,very-deep,False,False,3,0.26,0.324,0.341
xgb,deep,True,False,3,0.277,0.354,0.213
lgb,deep,True,False,3,0.304,0.377,0.107
xgb,very-deep,True,False,3,0.283,0.361,0.18
lgb,very-deep,True,False,3,0.295,0.367,0.152
xgb,shallow,False,False,3,0.22,0.285,0.489
lgb,shallow,False,False,3,0.282,0.349,0.233
xgb,shallow,True,False,2,0.287,0.365,0.16
lgb,shallow,True,False,2,0.304,0.381,0.085
xgb,medium,True,False,4,0.273,0.359,0.191
lgb,medium,True,False,4,0.295,0.37,0.138
xgb,very-deep,True,False,4,0.287,0.37,0.138
lgb,very-deep,True,False,4,0.301,0.375,0.115
xgb,deep,False,True,2,0.179,0.237,0.646
lgb,deep,False,True,2,0.259,0.326,0.332
xgb,shallow,True,False,0,0.289,0.375,0.114
lgb,shallow,True,False,0,0.3,0.375,0.114
xgb,deep,True,False,4,0.278,0.36,0.183
lgb,deep,True,False,4,0.305,0.382,0.08
xgb,very-deep,False,False,4,0.195,0.26,0.574
lgb,very-deep,False,False,4,0.263,0.325,0.335
xgb,deep,False,True,0,0.182,0.243,0.628
lgb,deep,False,True,0,0.23,0.294,0.458
xgb,medium,False,True,0,0.188,0.248,0.612
lgb,medium,False,True,0,0.255,0.323,0.342
xgb,deep,False,True,3,0.184,0.24,0.638
lgb,deep,False,True,3,0.268,0.339,0.278
xgb,shallow,False,True,0,0.193,0.253,0.597
lgb,shallow,False,True,0,0.266,0.335,0.294
xgb,medium,True,True,0,0.298,0.372,0.131
lgb,medium,True,True,0,0.287,0.357,0.196

xgb,shallow,True,False,3,0.295,0.373,0.123
lgb,shallow,True,False,3,0.316,0.398,0.003
xgb,very-deep,True,False,0,0.277,0.352,0.22
lgb,very-deep,True,False,0,0.303,0.376,0.112
xgb,deep,True,True,4,0.281,0.355,0.205
lgb,deep,True,True,4,0.283,0.353,0.215
xgb,very-deep,False,False,2,0.192,0.256,0.588
lgb,very-deep,False,False,2,0.262,0.324,0.339
xgb,medium,True,True,3,0.26,0.332,0.305
lgb,medium,True,True,3,0.294,0.365,0.16
xgb,very-deep,True,False,2,0.28,0.355,0.206
lgb,very-deep,True,False,2,0.295,0.369,0.141
xgb,deep,False,False,0,0.19,0.251,0.603
lgb,deep,False,False,0,0.254,0.324,0.339
xgb,shallow,False,True,2,0.196,0.256,0.589
lgb,shallow,False,True,2,0.26,0.324,0.34
xgb,medium,True,False,3,0.269,0.347,0.241
lgb,medium,True,False,3,0.291,0.365,0.164
xgb,very-deep,False,True,2,0.179,0.238,0.643
lgb,very-deep,False,True,2,0.253,0.32,0.357
xgb,deep,True,True,0,0.294,0.368,0.149
lgb,deep,True,True,0,0.297,0.366,0.158
xgb,medium,False,False,0,0.192,0.253,0.597
lgb,medium,False,False,0,0.259,0.333,0.302
xgb,medium,False,True,2,0.18,0.238,0.644
lgb,medium,False,True,2,0.253,0.317,0.369
xgb,deep,False,False,4,0.193,0.256,0.587
lgb,deep,False,False,4,0.26,0.324,0.342
xgb,shallow,True,True,2,0.372,0.461,-0.338
lgb,shallow,True,True,2,0.303,0.376,0.112
xgb,shallow,False,False,4,0.213,0.275,0.523
lgb,shallow,False,False,4,0.285,0.348,0.24
xgb,very-deep,False,True,4,0.187,0.25,0.608
lgb,very-deep,False,True,4,0.259,0.325,0.336
xgb,shallow,True,True,3,0.278,0.353,0.217
lgb,shallow,True,True,3,0.291,0.367,0.152
xgb,shallow,False,True,3,0.199,0.261,0.571
lgb,shallow,False,True,3,0.252,0.314,0.381
xgb,medium,False,False,2,0.19,0.252,0.602
lgb,medium,False,False,2,0.256,0.322,0.347
xgb,very-deep,True,True,3,0.28,0.354,0.21

lgb,very-deep,True,True,3,0.284,0.353,0.215
xgb,medium,True,True,4,0.268,0.344,0.257
lgb,medium,True,True,4,0.298,0.371,0.135
xgb,deep,True,False,0,0.284,0.356,0.204
lgb,deep,True,False,0,0.31,0.384,0.073
xgb,deep,False,False,2,0.196,0.261,0.572
lgb,deep,False,False,2,0.257,0.323,0.344
xgb,medium,False,True,4,0.184,0.245,0.623
lgb,medium,False,True,4,0.242,0.304,0.417
xgb,very-deep,False,True,3,0.184,0.24,0.638
lgb,very-deep,False,True,3,0.266,0.333,0.301
xgb,very-deep,True,True,4,0.297,0.373,0.126
lgb,very-deep,True,True,4,0.282,0.353,0.218
xgb,very-deep,True,True,0,0.29,0.363,0.173
lgb,very-deep,True,True,0,0.291,0.361,0.18
xgb,medium,False,False,4,0.195,0.26,0.574
lgb,medium,False,False,4,0.258,0.326,0.332
xgb,medium,True,True,2,0.273,0.344,0.254
lgb,medium,True,True,2,0.302,0.376,0.112
xgb,shallow,True,False,4,0.31,0.398,0.005
lgb,shallow,True,False,4,0.3,0.374,0.12
xgb,very-deep,True,True,2,0.292,0.364,0.167
lgb,very-deep,True,True,2,0.28,0.35,0.229
xgb,medium,True,False,0,0.279,0.355,0.209
lgb,medium,True,False,0,0.301,0.375,0.118
xgb,deep,False,False,3,0.184,0.244,0.624
lgb,deep,False,False,3,0.252,0.315,0.377
xgb,very-deep,False,False,0,0.192,0.256,0.589
lgb,very-deep,False,False,0,0.249,0.314,0.379

Literaturverzeichnis

Alan, Ozkan (2011): Thresholds based outlier detection approach for mining class outliers. In: *Expert Systems with Applications* 38.10, S. 12880–12889.

Baheti, Pragati (2021): Train Test Validation Split: How To & Best Practices. Online. URL: `https://www.v7labs.com/blog/train-validation-test-set`.

Bengio, Yoshua; Bergstra, James (2012): Grid Search for Hyper-Parameter Optimization. In: *Neural Information Processing Systems Workshop on Machine Learning Open Source Software*.

Brown, Iain (2021): Handling Outliers in ML: Best Practices for Robust Data. In: *Towards Data Science*. Online: `https://www.linkedin.com/pulse/handling-outliers-ml-best-practices-robust-data-iain-brown-ph-d--mwf6e`.

Brownlee, Jason (2020): Moving Average Smoothing for Data Preparation and Time Series Forecasting in Python. Online. URL: `https://machinelearningmastery.com/moving-average-smoothing-for-time-series-forecasting-python/`.

Chatfield, Chris (2003): The Analysis of Time Series: An Introduction. Boca Raton: CRC Press.

Chen, Tianqi; Guestrin, Carlos (2016a): XGBoost: A Scalable Tree Boosting System. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York: ACM, S. 785–794.

Chen, Tianqi; Guestrin, Carlos (2016b): XGBoost: A Scalable Tree Boosting System. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, S. 785–794.

Dash, Chandra Sekhar Kumar; Behera, Ajit Kumar; Dehuri, Satchidananda; Cho, Sung-Bae (2023): An outliers detection and elimination framework in classification task of data mining. In: *Decision Analytics Journal* 6, S. 1–14.

Experteninterview 1 (2025): Persönliches Interview mit Performance Engineer. Experteninterview.

Experteninterview 2 (2025): Persönliches Interview mit Performance Engineer. Experteninterview.

Farek, Lazhar; Benaidja, Amira (2024): Feature redundancy removal for text classification using correlated feature subsets. In: *Computational Intelligence* 40.3, e12621.

Friedman, Jerome H. (2001): Greedy Function Approximation: A Gradient Boosting Machine. In: *The Annals of Statistics* 29.5, S. 1189–1232.

García, Salvador; Luengo, Julián; Herrera, Francisco (2015): Data Preprocessing in Data Mining. Heidelberg: Springer.

Géron, Aurélien (2019): Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 2. Aufl. Sebastopol: O'Reilly Media.

Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron (2016): Deep Learning. Cambridge: MIT Press.

Guyon, Isabelle; Elisseeff, Andr'e (2003): An Introduction to Variable and Feature Selection. In: *Journal of Machine Learning Research*. Bd. 3, S. 1157–1182.

Hastie, Trevor; Tibshirani, Robert; Friedman, Jerome (2017): The Elements of Statistical Learning. Data Mining, Inference, and Prediction. 2. Aufl. New York: Springer.

Hodson, Timothy O. (2022): Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not. In: *Geoscientific Model Development* 15.14, S. 5481–5487. DOI: 10.5194/gmd-15-5481-2022.

James, Gareth; Witten, Daniela; Hastie, Trevor; Tibshirani, Robert (2021): An Introduction to Statistical Learning. with Applications in R. 2. Aufl. New York: Springer.

Jones, Alan (2022): Tree-Based Models vs. Deep Learning Models for Tabular Data: A Comprehensive Investigation. In: *Journal of Machine Learning Applications* 10.4, S. 112–130.

Ke, Guolin; Meng, Qi; Finley, Thomas; Wang, Taifeng; Chen, Wei; Ma, Weidong; Ye, Qiwei; Liu, Tie-Yan (2017): LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In: *Advances in Neural Information Processing Systems 30*. La Jolla: Curran Associates, S. 3146–3154.

Ke, Guolin; Meng, Qiang; Finley, Thomas; Wang, Taifeng; Chen, Wei; Ma, Weidong; Ye, Qi; Liu, Tie-Yan (2017): LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In: *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., S. 3146–3154.

Pedregosa, Fabian; Varoquaux, Gaël; Gramfort, Alexandre; Michel, Vincent; Thirion, Bertrand; Grisel, Olivier; Blondel, Mathieu; Prettenhofer, Peter; Weiss, Ron; Dubourg, Vincent; Vanderplas, Jake; Passos, Alexandre; Cournapeau, David; Brucher, Matthieu; Perrot, Matthieu; Duchesnay, Édouard (2011): Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12.1, S. 2825–2830.

Pedregosa, Fabian; Varoquaux, Gaël; Gramfort, Alexandre; Michel, Vincent; Thirion, Bertrand; Grisel, Olivier; Blondel, Mathieu; Prettenhofer, Peter; Weiss, Ron; Dubourg, Vincent; Vanderplas, Jake; Passos, Alexandre; Cournapeau, David; Brucher, Matthieu; Perrot, Michaël; Duchesnay, Édouard (2011): Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12, S. 2825–2830.

Prat, Nicolas; Comyn-Wattiau, Isabelle; Akoka, Jacky (2014): Artifact Evaluation in Information Systems Design-Science Research: A Holistic View. In: *Proceedings of the 18th Pacific Asia Conference on Information Systems (PACIS)*. Chengdu, China, S. 1–16.

Ribeiro, Marco Tulio; Singh, Sameer; Guestrin, Carlos (2016): "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, CA, USA, S. 1135–1144. DOI: 10.1145/2939672.2939778.

Smith, John; Doe, Jane (2022): Comparative Analysis of Random Forest and Gradient Boosting for Tabular Data. In: *International Journal of Data Science* 5.2, S. 45–58.

Smith, Steven W. (1997): The Scientist and Engineer's Guide to Digital Signal Processing. 1. Aufl. San Diego: California Technical Publishing.

Stone, Mervyn (1974): Cross-Validatory Choice and Assessment of Statistical Predictions. In: *Journal of the Royal Statistical Society* 36.2, S. 111–147.

Tsanas, Athanasios (2022): Relevance, redundancy, and complementarity trade-off: A principled, generic, robust feature-selection tool. In: *Patterns* 3.5, S. 100471.

Vapnik, Vladimir N. (2013): The Nature of Statistical Learning Theory. 2. Aufl. New York: Springer.

Venable, John; Pries-Heje, Jan; Baskerville, Richard (2016): FEDS: A Framework for Evaluation in Design Science Research. In: *European Journal of Information Systems* 25.1, S. 77–89. DOI: 10.1057/ejis.2014.36.

Willmott, Cort J.; Matsuura, Kenji (2005): Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. In: *Climate Research* 30.1, S. 79–82. DOI: 10.3354/cr030079.

Zheng, Alice; Casari, Amanda (2018): Feature Engineering for Machine Learning. Principles and Techniques for Data Scientists. 1. Aufl. Sebastopol: O'Reilly Media.

Erklärung zur Verwendung generativer KI-Systeme

Bei der Erstellung der eingereichten Arbeit habe ich auf künstlicher Intelligenz (KI) basierte Systeme benutzt:

☒ ja

☐ nein[46]

Falls ja: Die nachfolgend aufgeführten auf künstlicher Intelligenz (KI) basierten Systeme habe ich bei der Erstellung der eingereichten Arbeit benutzt:

1.

2.

3. ...

Ich erkläre, dass ich

- mich aktiv über die Leistungsfähigkeit und Beschränkungen der oben genannten KI-Systeme informiert habe,[47]
- die aus den oben angegebenen KI-Systemen direkt oder sinngemäß übernommenen Passagen gekennzeichnet habe,
- überprüft habe, dass die mithilfe der oben genannten KI-Systeme generierten und von mir übernommenen Inhalte faktisch richtig sind,
- mir bewusst bin, dass ich als Autorin bzw. Autor dieser Arbeit die Verantwortung für die in ihr gemachten Angaben und Aussagen trage.

Die oben genannten KI-Systeme habe ich wie im Folgenden dargestellt eingesetzt:

Arbeitsschritt in der wissenschaftlichen Arbeit	**Eingesetzte(s) KI-System(e)**	**Beschreibung der Verwendungsweise**

[46] Die Erklärung ist in jedem Fall zu unterzeichnen, auch wenn Sie keine KI-Systeme genutzt haben und Ihr Kreuz bei „nein“ gesetzt haben.

[47] U.a. gilt es hierbei zu beachten, dass an KI weitergegebene Inhalte ggf. als Trainingsdaten genutzt und wiederverwendet werden. Dies ist insb. für betriebliche Aspekte als kritisch einzustufen.

(Ort, Datum) (Unterschrift)

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema: *Mein Titel* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

(Ort, Datum) (Unterschrift)