Universität Leipzig
Fakultät für Mathematik und Informatik

Quantifizierung von Unsicherheit in Vorhersagen nicht-funktionaler Eigenschaften konfigurierbarer Softwaresysteme mit Conformal-Prediction

Masterarbeit

Stefan Jahns
geb. am: 29.08.1995 in Celle

Matrikelnummer 3711038

1. Gutachter: Prof. Dr. Ing. Norbert Siegmund

Datum der Abgabe: 2. August 2024

Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Leipzig, 2. August 2024

.....
Stefan Jahns

Zusammenfassung

In der vorliegenden Masterarbeit wird die Anwendung von Conformal-Prediction zur Vorhersage nicht-funktionaler Eigenschaften in konfigurierbaren Softwaresystemen untersucht. Angesichts der zunehmenden Komplexität und Flexibilität moderner Softwarelösungen ist die präzise Vorhersage von Performance-Metriken entscheidend für die Effizienz und Benutzerzufriedenheit. Diese Arbeit zielt darauf ab, die Potenziale und Herausforderungen der Conformal-Prediction-Methode zu identifizieren und zu diskutieren, insbesondere im Kontext der spezifischen Anforderungen konfigurierbarer Softwaresysteme.

Zunächst werden die Grundlagen konfigurierbarer Softwaresysteme und relevante Aspekte des maschinellen Lernens erläutert. Anschließend wird die Methodik der Conformal Prediction detailliert beschrieben und deren Anwendung auf Performance-Vorhersagen demonstriert. Die Ergebnisse bezüglich der Anwendbarkeit von Conformal Prediction im Bereich konfigurierbarer Softwaresysteme zeigen, dass Conformal Prediction eine vielversprechende Technik ist, um Unsicherheiten in Vorhersagen zu quantifizieren und die Zuverlässigkeit von Softwarekonfigurationen zu verbessern. Allerdings stellt die Interpretation der Ergebnisse eine nicht triviale Aufgabe dar.

Die Arbeit schließt mit einer Diskussion der Ergebnisse und einem Ausblick auf zukünftige Forschungsrichtungen, die darauf abzielen, die Integration von Conformal Prediction in den Softwareentwicklungsprozess weiter zu optimieren. Dabei werden mögliche Sampling-Strategien und Algorithmen sowie bestehende Herausforderungen bei der Bestimmung der Unsicherheiten in diesem Bereich erörtert.

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen **4**
2.1 Konfigurierbare Softwaresysteme 4
2.1.1 Allgemein 4
2.1.2 Formale Beschreibung 5
2.1.3 Sampling-Strategien 5
2.2 Maschinelles Lernen 7
2.2.1 Allgemein 7
2.2.2 Verwendete Algorithmen 8
2.2.3 MAPE 9
2.2.4 Hyperparameter Tuning 9
2.2.5 Unsicherheiten in der Vorhersage 10
2.2.6 Bayessche-Methoden 11
2.3 Conformal-Prediction 12
2.3.1 Allgemein 12
2.3.2 Vorbedingungen 16
2.3.3 Verschiedene Verfahren 16
2.3.4 Conformal-Prediction vs Bayessche-Methoden . . . 19

3 Methodik **21**
3.1 Forschungsfragen 21
3.2 Metriken 22
3.3 Experiment-Aufbau 23
3.3.1 Implementation 25
3.4 Operationalisierung 26

4 Auswertung **29**
4.1 Ergebnisse 29
4.1.1 FF1 29
4.1.2 FF2 35

4.1.3 FF3 38
4.1.4 FF4 40
4.2 Diskussion 43
4.2.1 FF1 - Verschiedene Samplegrößen 43
4.2.2 FF2 - Verschiedene Sampling-Strategien 44
4.2.3 FF3 - Korrelation zum MAPE 45
4.2.4 FF4 - Erkennen von Unsicherheiten 46
4.3 Gefährdungen der Validität 48

5 Fazit **50**
5.1 Zusammenfassung 50
5.2 Ausblick 51

Literaturverzeichnis **53**

Danksagung

An dieser Stelle möchte ich mich bei allen bedanken, die mich während der Erstellung dieser Masterarbeit unterstützt haben.

Ich möchte mich ebenfalls bei M.Sc. Johannes Dorn für seine wertvolle Betreuung und hilfreichen Ratschläge bedanken. Sein fachliches Wissen und seine Unterstützung waren für den Fortschritt und die Fertigstellung dieser Arbeit von großer Bedeutung.

Mein Dank geht auch an Prof. Dr. Norbert Siegmund für seine konstruktiven Anmerkungen und seine Unterstützung. Seine Expertise hat maßgeblich zur Weiterentwicklung meiner Ideen beigetragen.

Mein besonderer Dank gilt meinen Eltern für ihre fortwährende Unterstützung und Geduld, die mir während des gesamten Prozesses sehr geholfen haben.

Ich bin allen genannten Personen sehr dankbar für ihre Unterstützung und ihre wertvollen Beiträge.

Kapitel 1

Einleitung

In der modernen Softwareentwicklung gewinnen konfigurierbare Softwaresysteme zunehmend an Bedeutung [10]. Diese Systeme ermöglichen es, Softwarelösungen flexibel an die spezifischen Anforderungen und Präferenzen der Benutzer anzupassen [2]. Die Fähigkeit, verschiedene Konfigurationen zu erstellen und zu verwalten, ist entscheidend für die Effizienz und Effektivität von Softwareanwendungen in einer Vielzahl von Anwendungsbereichen, von Unternehmenssoftware bis hin zu eingebetteten Systemen.

Ein zentrales Anliegen bei der Entwicklung und dem Einsatz konfigurierbarer Softwaresysteme ist die Vorhersage ihrer nicht-funktionalen Eigenschaften, wie beispielsweise Performance, Zuverlässigkeit und Sicherheit [24]. Diese Eigenschaften sind oft entscheidend für die allgemeine Systemleistung und damit für die Benutzerzufriedenheit. Die Herausforderung besteht darin, dass die Auswirkungen einzelner Konfigurationsoptionen auf diese nicht-funktionalen Eigenschaften häufig komplex und nicht-linear sind [24]. Zusätzlich treten bei der Vorhersage verschiedene Unsicherheiten auf. Derzeit konzentrieren sich die meisten Ansätze jedoch hauptsächlich auf die Bestimmung der Einflüsse einzelner Konfigurationsoptionen, ohne die damit verbundenen Unsicherheiten zu berücksichtigen [8]. Ein tiefes Verständnis dieser Unsicherheiten ermöglicht es Entwicklern, fundierte Entscheidungen zu treffen, Risiken zu minimieren und die Qualität der Softwareprodukte zu verbessern.

Unsicherheiten können aus verschiedenen Quellen stammen, darunter ungenaue Modelle, unvollständige Daten oder die inhärente Variabilität der Systeme selbst [14]. Das das Erkennen und Quantifizieren von Unsicherheiten ist nicht nur für die Validierung der Vorhersagen von Bedeutung, sondern auch für die Entscheidungsfindung im Entwicklungsprozess. Beispielsweise könnten Konfigurationsoptionen, die erhebliche Unsicherheiten in die Vorhersagen einbringen, auf mögliche Performance-Bugs untersucht werden, die allgemein schwer zu identifizieren sind [11].

Ein Ansatz zur Lösung des Problems der Bestimmung dieser Unsicherheiten sind die Bayesschen-Methoden [8]. Diese Methoden erfordern jedoch zusätzliches Wissen über das Softwaresystem, das häufig nicht vorliegt und daher nur angenähert werden kann. Falls diese Annäherung nicht korrekt ist, kann es zu einer verminderten Verlässlichkeit ihrer Aussagen führen [7].

In dieser Arbeit wird die Methodik der CONFORMAL-PREDICTION untersucht, die sich als vielversprechender Ansatz zur Bewertung und Quantifizierung von Unsicherheiten in den Vorhersagen von nicht-funktionalen Eigenschaften konfigurierbarer Softwaresysteme erweist. CONFORMAL-PREDICTION bietet die Möglichkeit, Konfidenzintervalle für Vorhersagen zu generieren, die die Unsicherheiten in den Modellen berücksichtigen und somit eine robustere Entscheidungsgrundlage schaffen [16].

Die Methode CONFORMAL-PREDICTION basiert auf dem Gesetz der großen Zahlen und nutzt die Abweichungen von bereits beobachteten Werten zu ihren vorhergesagten Werten, um Konfidenzintervalle für neue Werte zu bestimmen [28]. Ein wesentlicher Vorteil von CONFORMAL-PREDICTION besteht darin, dass es keine Änderungen am Vorhersagemodell selbst erfordert und zugleich Garantien bezüglich der berechneten Konfidenzintervalle bietet [16]. Dies bedeutet, dass die in Ansätzen ohne Berücksichtigung von Unsicherheiten verwendeten Modelle übernommen werden können.

Ziel dieser Arbeit ist es, die Anwendbarkeit von CONFORMAL-PREDICTION in diesem Kontext zu evaluieren und zu analysieren, inwieweit diese Methodik dazu beitragen kann, ein besseres Verständnis der Unsicherheiten, die durch einzelne Konfigurationsoptionen entstehen, zu erlangen. Zu diesem Zweck haben wir Experimente an acht realen Softwaresystemen durchgeführt, um zu überprüfen, ob CONFORMAL-PREDICTION konsistent gute Ergebnisse unter den besonderen Bedingungen konfigurierbarer Softwaresysteme liefert. Zusätzlich wurden die Vorteile und neuen Erkenntnisse bezüglich der Vorhersage nicht-funktionaler Eigenschaften und der Bestimmung von Unsicherheiten betrachtet.

Im Detail haben wir vier Hauptaspekte zu CONFORMAL-PREDICTION im Bereich von konfigurierbaren Softwaresystemen untersucht:

- **Datenverfügbarkeit**: Wir haben geprüft, ob die üblicherweise verfügbaren Daten für die Vorhersage der nicht-funktionalen Eigenschaften auch für CONFORMAL-PREDICTION ausreichen.

- **Relevanz der Sampling-Strategie** Wir haben evaluiert, ob CONFORMAL-PREDICTION im Zusammenhang mit bestimmten Sampling-Strategien funktioniert, selbst wenn diese nicht unbedingt die Bedingungen für die Vorhersage der Konfidenzintervalle garantieren.

- **Vergleich zur bisherigen Bewertung:** Wir haben untersucht, inwieweit die Bewertung von CONFORMAL-PREDICTION von bisherigen Bewertungen zur Güte von Vorhersagen abweicht.
- **Nutzung von Unsicherheitsinformationen:** Wir haben analysiert, inwieweit die von CONFORMAL-PREDICTION gegebenen Informationen zur Unsicherheit einzelner Konfigurationsoptionen genutzt werden können, um Fehler in der Vorhersage bestimmter Einflüsse zu erkennen.

Durch die Beantwortung dieser Fragen wird ein Beitrag zur Verbesserung der Vorhersagegenauigkeit und zur Optimierung der Softwarekonfigurationen geleistet, was letztlich zu einer höheren Qualität und Zuverlässigkeit der entwickelten Systeme führt.

Struktur der Arbeit Die vorliegende Arbeit ist in mehrere Kapitel unterteilt, die aufeinander aufbauen:

- Kapitel 1 bietet eine Einführung in das Thema und erläutert die Relevanz konfigurierbarer Softwaresysteme in der modernen Softwareentwicklung.
- Kapitel 2 legt die theoretischen Grundlagen dar, einschließlich einer detaillierten Beschreibung konfigurierbarer Softwaresysteme, der verwendeten maschinellen Lernalgorithmen und CONFORMAL-PREDICTION.
- Kapitel 3 stellt die Methodik vor, die zur Anwendung von CONFORMAL-PREDICTION in dieser Forschung verwendet wurde.
- Kapitel 4 enthält die Auswertung der Ergebnisse, gefolgt von einer Diskussion der neuen Erkenntnisse.
- Kapitel 5 fasst die wichtigsten Ergebnisse zusammen und gibt einen Ausblick auf zukünftige Forschungsrichtungen.

Kapitel 2

Grundlagen

Dieses Kapitel führt die Grundlagen zum Verstehen der Zusammenhänge zwischen konfigurierbaren Softwaresystemen und CONFORMAL-PREDICTION ein. Das Kapitel teilt sich in drei Abschnitte. Zuerst wird die Domäne von konfigurierbaren Softwaresystemen erklärt, anschließend werden die für diese Arbeit relevanten Bereiche des maschinellen Lernens kurz abgehandelt und zum Schluss wird eine Einführung in die Methodik von CONFORMAL-PREDICTION gegeben.

2.1 Konfigurierbare Softwaresysteme

2.1.1 Allgemein

Die Konfiguration von Software in modernen Entwicklungsprozessen besitzt eine Vielzahl an Aspekten, welche nicht nur genutzt werden, um Software an Benutzeranforderungen anzupassen, sondern auch, um die Umgebung und Infrastruktur von Software-Systemen zu konfigurieren [25]. Siegmund et al. haben ein Konfigurationsmodell entworfen, welches dabei hilft, die verschiedenen Aspekte der Softwarekonfiguration zu erfassen und zu verstehen. Es umfasst die Aspekte, wie Softwareanpassung an Benutzeranforderungen, Konfiguration der Umgebung und Infrastruktur von Software-Systemen sowie die Interaktion verschiedener Stakeholder aus unterschiedlichen Hintergründen im Konfigurationsprozess.

In dieser Arbeit beschränken wir uns auf die Vorhersage von Performance, also nicht-funktionaler Anforderungen, zur Ausführungszeit. Dabei wird die Performance für eine ausgewählte valide Softwarekonfiguration vorhergesagt. Für eine repräsentative Auswahl an Softwarekonfigurationen für ein Softwaresystem bietet die Domäne der konfigurierbaren Softwaresysteme eigene Samplingstrategien, welche eine entscheidene Rolle bei der Vorhersage spielen [10].

Die davon in dieser Arbeit verwendeten werden in Unterabschnitt 2.1.3 erklärt.

2.1.2 Formale Beschreibung

Ein konfigurierbares Softwaresystem s besitzt eine Menge O^s an konfigurierbaren Optionen. Wir unterscheiden zwischen binären Optionen O_{Bin} und numerischen Optionen O_{Num}, wobei $O_{Bin} \cap O_{Num} = O^s$. Binäre Optionen können nur die zwei Werte 0 und 1 annehmen ($dom : O_{Bin} \to \{0, 1\}$), numerische Optionen werden durch ihren numerischen Wertebereich definiert ($dom : O_{Bin} \to \mathbb{N}$). Zum Beispiel sind binäre Optionen, ob eine Verschlüsselung der Daten durchgeführt werden soll, während numerische Optionen zum Beispiel die maximale Größe des Caches angibt. Zusätzlich zu den Optionen gibt es noch eine Menge an Constraints, welche zusätzliche Regeln darstellt, zum Beispiel, welche binären Optionen nicht gleichzeitig den Wert 1 annehmen dürfen. Eine valide Zuweisung von Werten für jede Option liefert uns eine valide Konfiguration $c \in C$. c wird dabei als Vektor kodiert, welcher die Wertzuweisung für jede Option beinhaltet. C ist die Menge, welche alle validen Konfigurationen für ein Softwaresystem beinhaltet.

2.1.3 Sampling-Strategien

Sampling-Strategien variieren darin, wie sie Konfigurationen für eine Lernmengen $C^{train} \subset C$ auswählen, von zufälliger Auswahlen bis hin zu strukturierten Ansätzen basierend auf spezifischen Kriterien. Diese Strategien bestimmen, wie Konfigurationen für das Training und Testen von Modellen ausgewählt werden und beeinflussen die Genauigkeit der Vorhersagen. Ein Vergleich von 18 verschiedenen Sampling-Strategien mit Fokus auf deren Einfluss auf die Genauigkeit der Vorhersage zeigte, dass es keine perfekte Strategie gibt. Es kann von System zu System unterschiedlich sein, welche Sampling-Strategie das bessere Ergebnis erzielt [10]. Im Folgenden werden die für diese Arbeit genutzte Auswahl an Sampling-Strategien im Detail erläutert.

Random

Das *Random-Sampling* beinhaltet die zufällige Auswahl von Konfigurationen ohne spezifische Kriterien. Diese Strategie dient als Baseline, da sie domänenunabhängig genutzt werden kann. Zu beachten ist, dass es aber bei konfigurierbaren Softwaresystemen normalerweise nicht trivial ist, valide Randomsamples zu erstellen [5]. In unserem Fall können die Sampels zufällig aus den gemessenen Konfigurationen gezogen werden.

Option-wise

Das *option-wise-Sampling* zielt darauf ab, eine Lernmenge auszuwählen, bei der jede binäre Konfigurationsoption $o_i \in O_{Bin}$ mindestens einmal ausgewählt wird $o_i = 1$, während die Auswahl anderer Optionen minimiert wird. Diese Strategie ermöglicht es, die individuellen Einflüsse der binären Optionen zu isolieren.

T-wise

T-wise-Sampling wählt eine Lernmenge aus, bei der jedes T-Tupel binärer Optionen $o_i, ..., o_j \in O_{Bin}$ mindestens einmal ausgewählt wird $o_i = ... = o_j = 1$. Zum Beispiel wählt T2 alle Paare von Optionen aus, während T3 alle Dreiergruppen auswählt. Durch die Minimierung anderer Kombinationen von Optionen sollen T-wise-Strategien helfen, die Interaktionen zwischen binären Optionen herauszufiltern.

Distance-Based

Das *Distance-based-Sampling* basiert auf zwei Hauptparametern: dem Distanzmaß und der Wahrscheinlichkeitsverteilung. Das Distanzmaß, wie z.B. die Manhattan-Distanz, misst die Entfernung einer Konfiguration von einem beliebig festen Referenzpunkt im Konfigurationsraum. Danach wird eine diskrete Wahrscheinlichkeitsverteilung verwendet, um Konfigurationen basierend auf ihren Distanzwerten auszuwählen. Dies soll zur Diversifizierung der Stichprobe beitragen, da es die Samples gemäß der gegebenen Wahrscheinlichkeitsverteilung über den Konfigurationsraum verteilt. Distance-Based Sampling ist dabei bezogen auf die Anzahl an Optionen des Softwaresystems $|O^s|$ skalierbarer als Random-Sampling. Distance-based-Sampling bietet jedoch keine Garantie dafür, dass die Samples gleichmäßig über den Konfigurationsraum C verteilt sind [13].

Placket-Burman

Das *Plackett-Burman-Sampling* ist eine Sampling-Strategie für numerische Optionen, die durch spezifische Ausgangswerte definiert ist. Die Anzahl der Experimente, die durchgeführt werden und die Level der Features, wobei letzteres die Anzahl der unterschiedlichen Werte ist, die die einzelnen Features annehmen können. Ziel dieser Strategie ist es, die Einflüsse numerischer Optionen effizient zu untersuchen, indem Konfigurationen mithilfe eines aus den beiden Parametern generierten Patterns, welches die Werte der numerischen Optionen festlegt, bestimmt werden [24].

Die Strategien mit festen Kriterien für die Optionen, also Option-wise und T-wise nennt man coverage-based, die anderen Strategien, die den Configurationsraum gleichmäßig abbilden sollen, nennt man random-based.

2.2 Maschinelles Lernen

2.2.1 Allgemein

Maschinelles Lernen bezeichnet statistische Algorithmen, welche anhand von Erfahrungen in Bezug auf eine Klasse von Aufgaben lernen, wann sich die Leistung, bestimmt durch ein Leistungsmaß, bei diesen Aufgaben mit diesen Erfahrungen verbessern. Dafür werden anhand von Lernalgorithmen mathematische Modelle erstellt, welche zur Lösung der Aufgabe verwendet werden können [18].

Das Ziel unserer Aufgabe, der Vorhersage von Performance-Werten bei konfigurierbaren Softwaresystemen (wie zum Beispiel dem Energieverbrauch), ist es, die unbekannte Funktion $f : C \to \mathbb{R}$ zu approximieren, welche den Zusammenhang zwischen ausgewählter Konfiguration und Performance-Wert darstellt. Die Funktion f bekommt in unserem Fall als Eingabevariable $c \in C^s$, eine valide Konfiguration und gibt als Ausgabe den dazugehörige Performance-Wert $f(c) \in \mathbb{R}$. Wir stellen einen Datenpunkt z dar, als Tupel aus Eingabe und dazugehöriger Ausgabe $z = (c, f(c))$ und alle möglichen Datenpunkte als

$$Z := \{(c, f(c)) : \forall c \in C\}.$$

Wir nutzen überwachtes Lernen, dass heißt zum Lernen brauchen wir Trainingsdaten

$$Z^{train} := \{(c, f(c)|c \in C^{train} \subset C\}$$

bestehend aus einer Menge an Tupeln aus Konfiguration und den dazugehörigem Performance-Wert. Außerdem brauchen wir zum validieren des Modells, mit zum Beispiel der Metrik MAPE (siehe Unterabschnitt 2.2.3), eine zu den Trainingsdaten disjunkte Menge an Testdaten

$$Z^{test} := \{(c, f(c)|c \in C^{test}, C^{test} \cap C^{train} = \emptyset\}.$$

Diese Daten wurden in unserem Fall durch Experimente in anderen Arbeiten bereits herausgefunden oder künstlich erzeugt.

Der nächste Schritt ist die Auswahl der Art des Modells, also die Art der Repräsentation der Daten. Wenn wir zum Beispiel ein lineares Modell wählen, welches die Funktion f als gewichtete lineare Kombination des Eingabevektors versucht darzustellen, haben wir ein leicht zu interpretierendes Modell,

können aber komplexere Zusammenhänge innerhalb der Daten nicht lernen. Andererseits, wenn wir sehr komplexe Modelle wie Neuronale Netze wählen, verlieren wir die Interpretierbarkeit und benötigen meist mehr Daten, können aber vielleicht eine Vorhersage mit geringerem Fehler erzielen [18]. Eine große Rolle dabei spielt noch das Tuning der Modelle, wie in Unterabschnitt 2.2.4 erklärt.

Das trainierte Modell $\hat{\mu} : C \to \mathbb{R}$ kann dann genutzt werden, um Performance-Werte von Konfigurationen vorherzusagen, dabei ist das Modell gut, wenn $\hat{\mu}(c)$ möglichst nah an $f(c)$ für alle möglichen Werte für c kommt.

2.2.2 Verwendete Algorithmen

Für diese Arbeit werden folgende vier Klassen von Machine-Learning-Algorithmen verwendet:

- **Lineare Modelle:** Lineare Modelle sind einfache Modelle, welche die Einflüsse der Eingabe auf die Ausgabe als Linearkombination von Eingabevektor und zu Gewichtsvektor darstellt. Um Interaktionen von Einflüssen darzustellen, muss der Eingabevektor um die Interaktionen, bestehend als Produkt der entsprechenden Optionswerte, erweitert werden [24]. Zur Regularisierung, damit nur relevante Einflüsse gelernt werden, kann das Modell zum Linear-Lasso-Modell erweitert werden [19].

- **Entscheidungsbäume:** Entscheidungsbäume führen rekursiv eine Aufteilung des Konfigurationsraums basierend auf Entscheidungen über den Wert der Konfigurationsoptionen durch [10]. Diese Partitionierung kann als binärer Baum dargestellt werden, bei dem jede Verzweigung eine Entscheidung ist, welcher Wert eine bestimmte Option annimmt. Zur Vorhersage einer neuen Konfiguration geht man den Baum dann entsprechend der Werte der einzelnen Optionen entlang. Der Vorteil von Bäumen ist, dass sie direkt Interaktionen lernen können.

- **Random-Forest:** Für Random-Forest werden mehrere Entscheidungsbäume mit zufälligen Stichproben aus Z^{train} trainiert. Dabei werden auch die Entscheidungen der Partitionierung oft mit nur zufälligen Teilmengen der Optionen getroffen. Am Ende werden die Bäume zu einem Forest (Wald) zusammengeführt. Das Ergebnis einer Vorhersage wird dann mithilfe einer gewählten Aggregat-Funktion aus den einzelnen Ergebnissen der Bäume gewonnen [12].

- **KernelRidge:** KernelRidge-Methoden nutzen lineare Lösungsmethoden zusammen mit dem Kernel-Trick um nichtlineare Probleme zu lösen, dabei werden die Daten mithilfe eines Kernels, welcher zum Beispiel qua-

dratisch oder exponentiell sein kann, in einen Raum projeziert, in dem eine lineare Lösung des Problems existiert. Der bekannteste Vertreter dieser Methoden ist die Support-Vector-Maschine [22].

2.2.3 MAPE

MAPE steht für Mean Absolute Percentage Error, zu Deutsch der mittlere absolute prozentuale Fehler. MAPE ist eine Metrik, welche in der Statisik und Regressionsanalyse verwendet wird, um die Genauigkeit von Vorhersagemodellen zu bewerten.
MAPE berechnet den durchschnittlichen prozentualen Fehler L_{MAPE} zwischen den Vorhersagen $\hat{\mu}(C^{test})$ eines Modells und den tatsächlichen Werten $f(C^{test})$:

$$L_{MAPE}(\hat{\mu}) = \mathbb{E}\left(\frac{|\hat{\mu}(C^{test}) - f(C^{test})|}{|C^{test}|}\right) \tag{2.1}$$

Diese Metrik kann zur Findung des Regression Modells mit dem geringsten Fehler in der Vorhersage genutzt werden, wie anderen Arbeiten schon gezeigt haben [6]. MAPE ist besonders nützlich, da die Metrik eine intuitive Interpretation bietet und sich gut für Anwendungen eignet, bei denen relative Fehler wichtiger sind als absolute Fehler. Deswegen wird in vielen Bereichen von Vorhersage MAPE verwendet, zum Beispiel im Finanzbereich oder in der Energieverbrauchsvorhersage, wo der relative Fehler meist wichtiger ist als der absolute [6]. Probleme bei MAPE ergeben sich, wenn die Vorhersagewerte sich nahe an Null befinden, denn dann kann L_{MAPE} schnell gegen Unendlich gehen und bietet keine vernünftige Interpretierbarkeit mehr.

2.2.4 Hyperparameter Tuning

Hyperparameter sind Parameter, die die Struktur oder das Verhalten eines Lernalgorithmus steuern, im Gegensatz zu den Modellparametern, die durch das Training des Modells gelernt werden. Beispiele für Hyperparameter sind die Lernrate, der Alpha-Wert der Linear-Lasso-Regression oder die maximale Tiefe eines Baumes. Hyperparameter-Tuning ist ein wichtiger Prozess im Maschinenlernbereich, bei dem die optimalen Einstellungen für die Hyperparameter eines Modells gefunden werden.

Das Ziel des Hyperparameter-Tunings ist es, die Hyperparameter so anzupassen, dass das Modell die bestmögliche Leistung auf neuen, nicht gesehenen Daten erzielt. Dies kann bei einem Modell den Unterschied zwischen Vorhersagen mit geringem oder großem Fehler ausmachen. In der Arbeit von Fu and Menzies wird gezeigt, dass selbst einfache Modelle, welche gut optimierte

Hyperparameter haben, besser werden können, als komplexere Modelle, wie Deep-Learning [9].

Traditionelle Ansätze zum Hyperparameter-Tuning umfassen Rastersuche, bei der eine vordefinierte Menge von Hyperparameterkombinationen systematisch ausprobiert wird, und manuelle Suche, bei der Expertenwissen verwendet wird, um Hyperparameterwerte zu wählen. Bei einer Evaluierung des alternativen Ansatzes, der Methode der Zufallssuche für Hyperparameter, wurde gezeigt, dass diese bessere Ergebnisse in kürzerer Zeit erziehlt [4].

Der Tuning Prozess wird oft mit Kreuzvalidierung kombiniert, um die Leistung des Modells auf validierten Daten zu bewerten. Das heißt, die Trainingsdaten werden in k viele disjunkte Untermengen geteilt und dann wird das Modell auf $k-1$ dieser Mengen trainiert und mit der übrigen Menge validiert. Dies passiert k mal, sodass jede Menge einmal zum Validieren genutzt wird.

2.2.5 Unsicherheiten in der Vorhersage

Es gibt verschiedene Quellen von Unsicherheit, die in mathematischen Modellen, Simulationen oder Vorhersagen vorkommen können. Kennedy and O'Hagan erläutern in ihrer Arbeit, dass es wichtig sei, die verschiedenen Quellen in den eigenen Vorhersagen zu identifizieren und quantifizieren, um fundierte Entscheidungen treffen zu können [14]. Für die Identifizierung schlagen sie folgende Arten der Unsicherheit vor:

- **Parameterunsicherheit:** Diese Art von Unsicherheit entsteht aus der Unkenntnis oder Variabilität von Modellparametern. Oft sind die wahren Werte dieser Parameter nicht genau bekannt und müssen geschätzt werden.

- **Modellinkompatibilität:** Modellunsicherheit tritt auf, wenn das mathematische Modell, das zur Beschreibung eines Systems verwendet wird, nicht alle relevanten Einflussfaktoren oder Prozesse genau abbildet. Dies kann zu Abweichungen zwischen Modellvorhersagen und tatsächlichen Beobachtungen führen.

- **Beobachtungsfehler:** Diese Art von Unsicherheit entsteht aus begrenzten oder ungenauen Daten, die zur Kalibrierung oder Validierung eines Modells verwendet werden. Messfehler, Datenlücken oder unvollständige Informationen können zu Datenunsicherheit führen.

- **Variabilität:** Variabilität bezieht sich auf natürliche Schwankungen oder Zufälligkeiten in einem System, die zu Unsicherheiten in den Modellergebnissen führen können.

- **Modellvereinfachungen:** Wenn ein Modell Annahmen oder Vereinfachungen über das zugrunde liegende System macht, können Unsicherheiten entstehen. Diese Unsicherheiten können durch die Komplexität des Systems oder durch unberücksichtigte Effekte verursacht werden.
- **Externe Einflüsse:** Unsicherheiten können auch durch externe Faktoren entstehen, die das System beeinflussen, wie z.B. bei der Performance Vorhersage ein Hardwareaustausch, ein Systemupdate oder die Raumtemperatur.

Nachdem die für das eigene Verfahren wichtigen Quellen identifiziert wurden, müssen diese quantifiziert werden, dass heißt ihre Größe, Verteilung und Auswirkung auf das Modell mithilfe mathematischer Methodiken zu bestimmen. Dafür können zum Beispiel probabilistische Ansätze wie Bayessche-Methoden (siehe Unterabschnitt 2.2.6) oder Monte-Carlo-Simulation genutzt werden. Auch Conformal-Prediction (siehe Abschnitt 2.3) ist eine Methode zur Bestimmung der Modellunsicherheit.

Der nächste Schritt, nach der Quantifizierung der Unsicherheiten, ist die Analyse der Unsicherheiten um die Robustheit der Modellergebnisse zu bewerten und Sensitivitätsanalysen durchzuführen. Letzteres sind statistische Verfahren, die verwendet werden, um die Auswirkungen von Veränderungen in den Eingangsparametern eines Modells auf die Ergebnisse des Modells zu untersuchen. Mit diesen Analysen kann man potenzielle Risiken oder Schwachstellen im Modell zu identifizieren.

2.2.6 Bayessche-Methoden

Bayessche-Regression ist eine statistische Methode, die die Bayessche-Statistik verwendet, um Regressionsmodelle zu schätzen. Im Gegensatz zur klassischen Regression berücksichtigt die Bayessche-Regression Unsicherheiten in den Modellparametern, indem sie Prior-Verteilungen für die Parameter festlegt und diese mit den Beobachtungsdaten aktualisiert, um Posterior-Verteilungen zu erhalten. Hierbei drückt die Prior-Verteilung das anfängliche Wissen oder die Annahmen über die Parameter vor Berücksichtigung der Beobachtungsdaten aus. Sie dient als Ausgangspunkt für die Schätzung der Parameter und kann aufgrund früherer Studien, Expertenwissen oder theoretischer Überlegungen festgelegt werden. Die Prior-Verteilung spiegelt entsprechend des Vorwissen darüber wieder, welche Werte die Parameter wahrscheinlich annehmen. Die Posterior-Verteilung wird durch die Aktualisierung der Prior-Verteilung mit den Beobachtungsdaten berechnet. Sie kombiniert das anfängliche Wissen aus der Prior-Verteilung mit den Informationen aus den Daten, um eine aktualisierte Schätzung der Parameter zu liefern. Dabei wird

die Wahrscheinlichkeit, bestimmte Beobachtungen zu sehen, wenn die Parameter festgelegt sind, als Likelihood bezeichnet. Die Posterior-Verteilung gibt an, wie wahrscheinlich verschiedene Werte der Parameter sind, nachdem die Beobachtungsdaten berücksichtigt wurden [14, 18].

2.3 Conformal-Prediction

In diesem Abschnitt wird Conformal-Prediction vorgestellt. Zuerst gibt es eine kurze historische Übersicht und danach eine allgemeine Einleitung, was Conformal-Prediction ist. Im Anschluss werden dann die Vorbedingungen, sowie eine Auswahl an Algorithmen genauer erklärt.

Conformal-Prediction ist ein von Prof. Vladimir Vovk entwickeltes Framework zur Bestimmung von Unsicherheit im Bereich des Maschinellen Lernens [16]. Zusammen mit Alexander Gammerman, Craig Saunders und Vladimir Vapnik wurde das Rahmenkonzept von Conformal-Prediction zwischen 1996 und 1999 entwickelt. Frühe Meilensteine umfassen den Beweis von 2002, dass bei der transduktiven Variante von Conformal-Prediction die Fehlerwahrscheinlichkeit unabhängig über Zeitschritte hinweg ist, die Entwicklung von Split-Conformal Predictors im selben Jahr durch Papadopoulos et al. und die Prägung des Begriffs "Conformal-Prediction "durch Glenn Shafer im Jahr 2003 [26, 20]. In den folgenden Jahren wurden weitere Varianten wie Venn Predictors, Cross-Conformal Predictors und Venn-Abers Predictors entwickelt. Währenddessen wurde Conformal-Prediction durch die Arbeit von Jing Lei, Larry Wasserman und Kollegen in den USA populär gemacht, sodass allein im Jahr 2023 über 1000 Paper zu Conformal-Prediction veröffentlicht wurden. Eine genauere Übersicht der Geschichte von Conformal-Prediction wird in den Büchern von Vovk et al. und Manokhin gegeben [31, 16].

2.3.1 Allgemein

Conformal-Prediction gibt uns für jede Punktvorhersage eines gegebenen Maschine-Learning-Modells zusätzlich ein valides Konfidenzintervall. Genauer gesagt, heißt das, dass unser Conformal-Prediction Modell, ein Intervall ausgibt, welches ein vorher festgelegtes Konfidenzlevel, zum Beispiel 90%, hat. In diesem Intervall liegt dann mit 90% Wahrscheinlichkeit der reale Wert zu unserer Vorhersage. Bezogen auf alle Vorhersagen liegen dann mindestens 90% der realen Werte innerhalb der Intervallgrenzen, dann gilt das Intervall als valide. Wie viele der realen Werte innerhalb der Intervallgrenzen liegen wird bei Conformal-Prediction als Coverage bezeichnet. Wenn der Be-

weis, dass das Verfahren für die Bestimmung der Intervalle unter den in Unterabschnitt 2.3.2 genannten Vorbedingungen valide ist, wurde bereits in 2005 gezeigt [28].

In bisherigen Arbeiten wurden die folgenden Vorteile von CONFORMAL-PREDICTION aufgeführt [16]:

- **Validität:** Die Validität der Konfidenzintervalle wird bei CONFORMAL-PREDICTION, im Gegensatz zu anderen für Konfidenzintervalle genutzten Frameworks, automatisch garantiert.
- **Verteilungsunabhängig:** CONFORMAL-PREDICTION funktioniert unabhängig von der Verteilung der Daten, oder der Größe der Datensets. Im Gegensatz zu klassischen statistischen Methoden, welche eine bestimmte Verteilung oder Menge an Daten benötigen, um valide Intervalle vorherzusagen, benötigt CONFORMAL-PREDICTION nur *Exchangebility*, welche eine Abschwächung der Annahme von unabhängigen, identisch verteilten Zufallsvariablen (i.i.d.) ist.
- **Nicht-Beeinflussend:** Die zugrunde liegende Punktvorhersage wird durch CONFORMAL-PREDICTION nicht verändert, das heißt, man bekommt bei CONFORMAL-PREDICTION zusätzlich das Konfidenzintervall, ohne dass sich das Ergebnis der Punktvorhersage ändert. Man kann CONFORMAL-PREDICTION also einfach zusammen mit einem bestehenden Modell verwenden.
- **Modellagnostisch:** Es ist für die meisten CONFORMAL-PREDICTION Algorithmen es egal, wie das darunterliegende Modell aussieht, es muss nicht einmal ein Maschine-Learning-Modell sein, sondern kann auch eine Heuristik basierend auf Expertenwissen sein. Das heißt, man kann CONFORMAL-PREDICTION immer nutzen, um eine zusätzliche Bestimmung der Unsicherheit des Modells oder der Heuristik zu bekommen.

Effizienz bei CONFORMAL-PREDICTION Solange die Daten exchangeble sind, ist das vorhergesagte Konfidenzintervall valide. Um jedoch das Intervall möglichst *effizient*, also möglichst klein zu bekommen, gibt es abgesehen von der Güte der genutzten Punktvorhersage drei zu beachtende Punkte:

- **Auswahl des CONFORMAL-PREDICTION Algorithmus:** Wie Manokhin in seiner Doktorarbeit beschreibt, nutzt nicht jeder Algorithmus alle gegebenen Daten komplett aus, was zu einer geringeren Effizienz führt [17]. Eine genaue Erklärung zu den verschiedenen Algorithmen und ihren Unterschieden wird später im Unterabschnitt 2.3.3 gegeben.

- **Auswahl des Non-Conformity-Measure:** Das *Non-Conformity-Measure* gibt an, wie stark sich neu betrachtete Punkte von den bisher betrachteten Punkten unterscheiden. Dabei gibt es jedem bisherigen Datenpunkt einen Conformity-Score, welcher als ein numerischer Wert ausgegeben wird. Bei jedem neuen Punkt können wir basierend auf den bisherigen Scores im Vergleich bestimmen, wie groß das Konfidenzintervall ist. Ein ausführliches Beispiel dafür gibt Shafer and Vovk in seinem Paper anhand des Iris Datasets [23].
- **Menge der Daten:** CONFORMAL-PREDICTION sollte zwar auch mit kleinen Datenmengen valide sein, aber nicht unbedingt effizient. Auch wird in einer anderen Arbeit gezeigt, dass die Vergrößerung des Kalibrierungssets, einer zusätzlichen Trainingsmenge für induktive Verfahren, bei induktiven Algorithmen eine asymptotische Annäherung an die gewünschte Coverage zur Folge hat [1].

Non-Conformity-Measure Die Wahl des Non Conformity Measure kann erheblichen Einfluss auf die Effizienz, also die Größe, der Intervalle haben. Dabei ist das Non-Conformity-Measure mathematisch beschrieben eine reale Funktion $A(B, z)$, die angibt, wie sehr sich der neuer Punkt z von allen schon gesehenen Punkten, der Multimenge B, unterscheidet. Es gilt, je größer A, desto stärker unterscheidet sich z von B. In der Praxis wird dafür im Allgemeinen ein Distanzmaß $d(z, z')$ genutzt, um die Vorhersage des Punktes $\hat{\mu}_B(c)$ mit $z = (c, f(c))$ zu vergleichen:

$$A(B, z) := d(\hat{\mu}_B(c), f(c)). \tag{2.2}$$

Dabei ist das Distanzmaß relativ unwichtig, denn die Konfidenzintervalle ändern sich nicht, wenn das Non-Conformity-Measure A monoton transformiert wird [23]. Die wichtige Funktion ist also die Punktvorhersage, also die Wahl des Maschine-Learning-Modells (bzw. der Heuristik). Für den in dieser Arbeit betrachteten Fall von Regression ist der Ansatz eindeutig, denn umso näher die Vorhersage an den einzelnen Punkten ist, desto kleiner die Intervalle.

Die einzelnen Werte des Non-Conformity-Measure werden als Scores bezeichnet. Mit ihnen werden die Intervalle für neue Punkte berechnet, indem zuerst das $1 - \alpha$ Quantil bestimmt wird, für ein 90% Konfidenzintervall ist $\alpha = 0.1$. Im Anschluss wird daraus zurückgerechnet, wie groß das Intervall sein muss.

Kleines Beispiel Für ein kleines Beispiel betrachten wir eine Menge von 10 Zahlen $B = \{b_0, ..., b_9\}$ und als Vorhersage nehmen wir den Mittelwert $\overline{z}_B$. Dann ist das Non-Conformity-Measure $A(B, z) := |\overline{z}_B - z|$, also sind die

Tabelle 2.1: Beispiel für die Berechnung des 80% Konfidenzintervall Anhand von Non-Conformity-Scores, die Vorhersage Werte sind die Werte, nachdem $b_0 ... b_9$ gesehen wurden.

	b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
Echter Wert	7	3	4	3	6	12	2	8	7	5	?
Vorhersage	5,7	5,7	5,7	5,7	5,7	5,7	5,7	5,7	5,7	5,7	5,7 ± 3,7
NC-Score	1,3	2,7	1,7	2,7	0,3	6,3	3,7	2,3	1,3	0,7	-

Scores der Abstand zum Mittelwert, wie in Tabelle 2.1 aufgeführt. Wenn wir das 80% Konfidenzintervall bekommen sollen, also $\alpha = 0.2$ ist, dann schauen wir uns den zweitgrößten Score an, denn $\alpha \cdot |B| = 2$. Das Quantil wird also über den zweitgrößten Score s_q definiert, so dass unsere nächste Vorhersage b_{10} das Intervall $\mathcal{CI} = \overline{z}_B \pm s_q$ ist.

Metriken Um zu überprüfen, ob das genutzte Non-Conformity-Measure sinnvoll gewählt wurde, sollte man vor allem zwei Metriken betrachten.

Zum einen zeigt die Verteilung der Intervallbreiten, wie gut zwischen Eingaben unterschieden werden kann, welche wenig oder viel Unsicherheit in ihrer Vorhersage haben und sowie die Größe der Unsicherheit im Mittel ist. Wenn das Intervall überall nahezu gleich groß ist, ist die Vorhersage entweder überall gleich unsicher, ein Fall, der in der Praxis unwahrscheinlich ist, oder das Non-Conformity-Measure wurde schlecht gewählt, so dass die Scores nicht die Unsicherheit bezogen auf die Eingabe widerspiegeln.

Zum anderen muss man darauf achten, ob die Coverage *marginal* oder *konditional* ist. Marginal heißt in diesem Fall, dass die Fehler, also dass ein Punkt außerhalb des Intervalls ist, sich auf bestimmte Bereiche des Eingaberaums beschränken, während bei konditional die Fehler gleichmäßig über den Raum verteilt sind. Auch dies hängt von der Güte des Non-Conformity-Measure ab. Zum Bestimmen, ob die Coverage konditional ist, wurde die *Feature-stratified Coverage Metrik* (FSC Metrik) eingeführt [1]. Gegeben sei eine Menge an Features $O = \{o_1, ..., o_{|O|}\}$, welche die diskreten Werte $\{0, ..., G\}$ für ein G annehmen können. Seien $c \in C^{test}$ die Punkte in einem Validierungsset, nun sei $C^{test}_{o,g}$ die Menge an Beobachtungen, bei denen $o = g$ für $g \in 0, ..., G$ und $o \in O$. Bei guter konditionaler Coverage sollte die Coverage überall gleich groß sein, deswegen reicht es den minimalen Wert über alle Belegungen für ein Feature o zu betrachten:

$$FCM(C^{test}, o) = \min_{g \in \{1,...,G\}} \frac{1}{|C^{test}_{o,g}|} \sum_{c \in C^{test}_{o,g}} \mathbb{1}\{\hat{\mu}(c) \in \mathcal{CI}(c)\} \tag{2.3}$$

Diese Metrik berechnet für jeden Wert eines betrachteten Features die Coverage und gibt uns das Minimum davon zurück. Wenn eine konditionale Coverage vorliegt, sollte der Wert für alle Features $1 - \alpha$ sein, umso stärker er unterhalb liegt, umso schlechter ist die konditionale Coverage. Für kontinuierliche (numerische) Features kann man die Metrik nutzen, indem man die Werte in eine endliche Anzahl an Klassen einteilt, ähnlich wie beim Binning für Histogramme.

2.3.2 Vorbedingungen

Für CONFORMAL-PREDICTION werden, wie bei normalen Maschine Learning Algorithmen, die Daten in diskunkte Trainings- und Validierungssets geteilt, oder im Spezialfall von INDUKTIVER-CONFORMAL-PREDICTION in disjunkte (Trainings-), Kalibrierungs- und Validierungssets. Die einzige Bedingung, welche diese Mengen erfüllen müssen ist, dass sie exchangeable sind. Das heißt, dass die Reihenfolge der Punkte über alle Mengen, keine Rolle spielen darf.

Mathematisch wird eine geordnete Reihe als exchangeable genannt, wenn die Wahrscheinlichkeit $P(z_1, \ldots, z_k) = P(z_{\pi(1)}, \ldots, z_{\pi(k)})$, wobei $\pi(1), \ldots, \pi(k)$ eine beliebige Permutation der Indizes $1, \ldots, k$ ist. In der Praxis ist diese Annahme schwächer, als die am häufigsten verwendete Annahme der unabhängig, identisch verteilten Zufallsvariable (iid), denn es wird die identische Verteilung vorausgesetzt, nicht aber die Unabhängigkeit der Samples. Als Beispiel sind zufällig gezogene Samples ohne Zurücklegen nach dieser Definition exchangeble, aber genau genommen nicht unabhängig, also nicht iid [15].

Übertragen auf CONFORMAL-PREDICTION heißt das, dass das Trainings- und Validationset eine identische Verteilung der vorherzusagenden Werte haben muss. Bei INDUKTIVER-CONFORMAL-PREDICTION wird in [15] erklärt, dass nur das Kalibrierungs- und Validierungsset exchangeable sein muss, da die Breite der Intervalle nur aus den Scores des Kalibrierungssets bestimmt werden.

2.3.3 Verschiedene Verfahren

Bei den Verfahren von CONFORMAL-PREDICTION wird zwischen zwei Ansätzen unterschieden. Der *induktive* Ansatz, welcher das Aufteilen der Daten in Trainings- und Kalibrierungsset erfordert und der *transduktive* Ansatz, welcher viele Modellanpassungen erfordert, jedoch kein Aufteilen der Daten benötigt. Historisch gesehen wurde zuerst die TRANSDUKTIVE-CONFORMAL-PREDICTION entwickelt, und später wurde die INDUKTIVE-CONFORMAL-PREDICTION als ein wichtiger Spezialfall erkannt. Bezogen auf die Effizienz schneiden induktive Algorithmen im Allgemeinen schlechter ab als die transduktiven

Algorithmen, sie werden aber in der Praxis oft wegen ihrer geringen Rechenzeit genutzt [17]. Für diese Arbeit werden drei Verfahren genutzt:

- JACKKNIFE+, ein transductive Ansatz vorgestellt im Jahr 2021 [3]
- CROSSVALIDATION+ (CV+), ein auf Cross-Conformal-Prediction von Vovk basierendes Verfahren, welches einen Mittelweg zwischen den Ansätzen geht [1].
- CONFORMALIZED-QUANTIL-REGRESSION (CQR), ein in der Praxis gerne genutzter induktiver Ansatz, vorgestellt im Jahr 2019 [21].

JACKKNIFE+

Die JACKKNIFE+ Methode ist eine Abwandlung der JACKKNIFE Methode, welche auf dem Konzept des Resamplings oder Subsamplings von verfügbaren Daten basiert [3]. Beim JACKKNIFE wird für jeden der Punkte im Trainingsset $z_i := (c_i, f(c_i)) \in Z^{train}$ ein Leave-One-Out Modell $\hat{\mu}_{-i}$ erstellt, mithilfe aller Punkte außer diesem (dem i-tem). Danach wird der nonconformity Score für diesen Punkt berechnet, als $R_i^{LOO} = |f(c_i) - \hat{\mu}_{-i}(c_i)|$ und wie beim induktiven CONFORMAL-PREDICTION als Kalibrierung verwendet. Die Funktion zur Bestimmung des Intervalls $\mathcal{CI}$ ist dann:

$$\mathcal{CI}_{n,\alpha}^{Jackknife}(c_{n+1}) = \left[\hat{q}_{n,\alpha}^{-}\left\{\hat{\mu}(c_{n+1}) - R_i^{LOO}\right\}, \hat{q}_{n,\alpha}^{+}\left\{\hat{\mu}(c_{n+1}) + R_i^{LOO}\right\},\right] \quad (2.4)$$

Die JACKKNIFE Methode gibt allerdings keine allgemein gültige Garantie für die Coverage, deswegen wurde die JACKKNIFE+ Methode entwickelt. Diese nutzt anders als die JACKKNIFE Methode, jedes Leave-One-Out Modell für die Vorhersage des nächsten Punktes und der Berechnung der Scores. Die abgewandelte Formel ist:

$$\mathcal{CI}_{n,\alpha}^{\text{JACKKNIFE}+}(c_{n+1}) = \left[\hat{q}_{n,\alpha}^{-}\left\{\hat{\mu}_{-i}(c_{n+1}) - R_i^{LOO}\right\}, \hat{q}_{n,\alpha}^{+}\left\{\hat{\mu}_{-i}(c_{n+1}) + R_i^{LOO}\right\},\right] \quad (2.5)$$

Die Abbildung 2.1 zeigt, wie sich diese Änderung auf die Berechnung des vorhergesagten Intervalls auswirkt. JACKKNIFE+ gibt dabei eine Coverage Garantie von $1 - 2\alpha$, wobei in der Praxis meist die gewünschte Coverage von $1 - \alpha$ erreicht wird [16].

CROSSVALIDATION+

Der Cross Validation Ansatz funktioniert ähnlich zur JACKKNIFE+ Methode. Für die CROSSVALIDATION+ Methode wird dabei das Trainingsset Z^{train} in K gleichgroße disjunkte Untermengen $S_1, S_2, ..., S_K$ geteilt. Danach werden K viele Regressionen $\hat{\mu}_{-S_k}$ auf den Trainingsdaten ohne S_k durchgeführt. Die

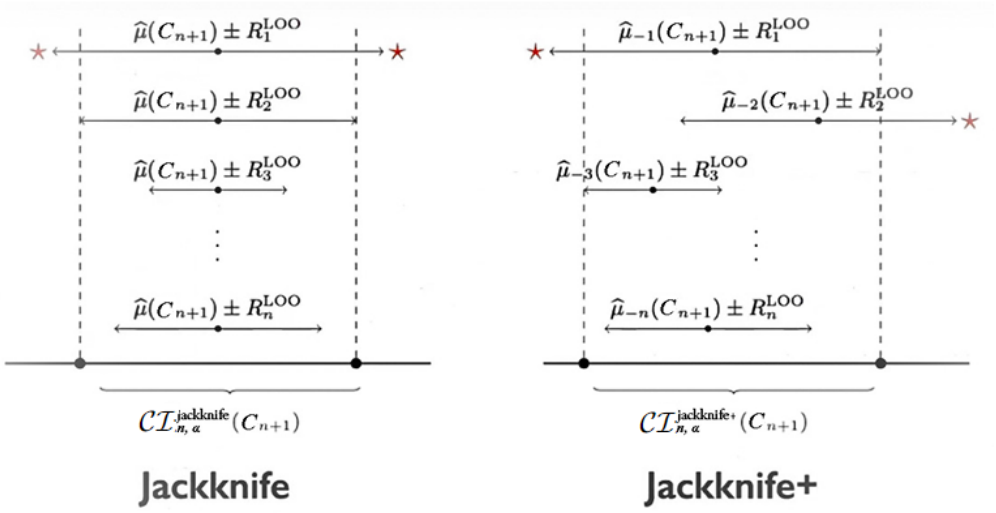


Abbildung 2.1: Intervallvorhersage von JACKKNIFE und JACKKNIFE+. Es wird gezeigt, wie das $n+1$te Intervall anhand der bisherigen n Datenpunkte berechnet wird.

jeweiligen non Conformity Scores werden dann als $|f(c_i) - \hat{\mu}_{-S_k}(c_i)|$ berechnet, wobei der Punkt $(c_i, f(c_i)) \in S_k$ ist. Die JACKKNIFE+ Methode ist also der Spezialfall, bei dem $K = |Z^{train}|$ ist.

Es wurde gezeigt, dass JACKKNIFE+ leicht schmalere Intervalle berechnet als CROSSVALIDATION+, aber bei größeren Datenmengen sollte man CROSSVALIDATION+, wegen des deutlich geringeren Rechenaufwands bei gleicher Coverage Garantie von $1 - 2\alpha$, wählen [3].

CONFORMALIZED-QUANTIL-REGRESSION

CONFORMALIZED-QUANTIL-REGRESSION ist eine Kombination von CONFORMAL-PREDICTION und direkter Quantil Regression, welche die Intervalle mit der Coverage Garantie von $1 - 2\alpha$ und geringem Rechenaufwand erstellt. Dabei ist CONFORMALIZED-QUANTIL-REGRESSION sehr anpassungsfähig gegenüber Heteroskedastizität [21]. CONFORMALIZED-QUANTIL-REGRESSION funktioniert dabei in zwei Schritten:

Erstens die Trainingsphase mit der Menge Z^{train}, in der zusätzlich zu unserer Punktvorhersage noch die $1 - \frac{\alpha}{2}$ und $\frac{\alpha}{2}$ Quantile berechnet werden. Dabei kann man auf bestehende Verfahren, wie zum Beispiel Pinball-Loss, zurückgreifen [1]. Diese Verfahren können Quantile berechnen, geben aber keine Garantie für die Coverage [1].

Im zweiten Schritt, der Kalibrierungsphase, werden die aus dem ersten

Schritt erzeugten Intervalle konformisiert. Dies erfolgt, indem man Fehler, welche von den Intervallen zu den gemessenen Werten aus dem Kalibrierungsset $Z^{calib} := \{(c, f(c))|c \in C^{calib}\}$ gemacht werden, betrachtet und die Intervallbreite entsprechend angepasst.

Sei dafür $\mathcal{A}$ ein beliebiger Quantil Regression Algorithmus, welcher uns die konditionierten Quantile $\hat{q}_{\alpha_{lo}}$ und $\hat{q}_{\alpha_{hi}}$ ausgibt:

$$\{\hat{q}_{\alpha_{lo}}, \hat{q}_{\alpha_{hi}}\} \leftarrow \mathcal{A}(\{(c, f(c)) : c \in C^{train}\}), \tag{2.6}$$

dann werden die Conformity Scores, also die Fehler der Intervalle $[\hat{q}_{\alpha_{lo}}(c), \hat{q}_{\alpha_{hi}}(c)]$ berechnet, als:

$$E_c := max\{\hat{q}_{\alpha_{lo}}(c) - f(c), f(c) - \hat{q}_{\alpha_{hi}}(c)\} \tag{2.7}$$

für alle $c \in C^{calib}$. Der Score E gibt also die stärke des Fehlers zu dem näherem Quantil an, wobei ein positiver Wert für Unterschätzen und ein negativer Wert für Überschätzen der Coverage steht. Zuletzt werden die konformen Intervalle für neue Daten c_{n+1}berechnet mit:

$$\mathcal{CI}(c_{n+1}) = \left[\hat{q}_{\alpha_{lo}}(c_{n+1}) - Q_{1-\alpha}(E, C^{calib}), \hat{q}_{\alpha_{hi}}(c_{n+1}) + Q_{1-\alpha}(E, C^{calib})\right] \tag{2.8}$$

wobei

$$Q_{1-\alpha}(E, C^{calib}) := (1-\alpha)(1 + \tfrac{1}{|C^{calib}|}) \text{ empirischstes Quantil von } \{E_c : c \in C^{calib}\} \tag{2.9}$$

die Anpassung der Quantil Intervalle ist.

2.3.4 Conformal-Prediction vs Bayessche-Methoden

In einer anderen Arbeit wurde bereits ein Vergleich zwischen Bayessche-Methoden, Conformal-Prediction, Ensembleprognosen und direkten Intervallbestimmungsverfahren (wie einfache Quantil Regression) durchgeführt [7]. Der Vergleich zwischen Bayessche-Methoden und Conformal-Prediction in Bezug auf die Schätzung von Vorhersageintervallen für Regressionsprobleme bietet interessante Einblicke bezüglich folgender Kriterien:

Marginal Validität Die marginale Validität bezieht sich darauf, ob die Wahrscheinlichkeitsaussagen, die von einem Modell gemacht werden, tatsächlich mit den beobachteten Daten übereinstimmen.

- **Bayessche-Methoden**: Bei genauer Inferenz mit korrekten Priors können Bayessche-Methoden eine gute marginale Validität aufweisen, aber in der Praxis kann dies schwierig sein, insbesondere bei komplexen Modellen.

- **Conformal-Prediction**: Conformal-Prediction, insbesondere mit den in Unterabschnitt 2.3.3 genannten Verfahren, bietet eine marginale Validität, da es Vorhersageintervalle erzeugt, welche die tatsächlichen Datenpunkte mit einer bestimmten Wahrscheinlichkeit umschließen, wenn die Voraussetzung der Exchangebility gegeben ist.

Skalierbarkeit Die Skalierbarkeit bezieht sich darauf, wie gut eine Methode auf große Datensätze oder komplexe Modelle angewendet werden kann.

- **Bayessche-Methoden**: Bayessche-Methoden können bei genauer Inferenz an Skalierbarkeitsgrenzen stoßen, insbesondere wenn die Berechnung von Posterior-Verteilungen aufwändig ist.

- **Conformal-Prediction**: Conformal-Prediction ist im Allgemeinen skalierbar, insbesondere wenn die Induktiven Ansätze verwendet werden.

Domänenwissen: Für diesen Vergleich ist es wichtig, wie viel zusätzliches Wissen die Verfahren benötigen:

- **Bayessche-Methoden**: Bayessche-Methoden erfordern oft ein gewisses Maß an Domänenwissen, insbesondere bei der Festlegung von Priors und der Modellierung komplexer Zusammenhänge. Dies kann die Anwendung der Methode in bestimmten Bereichen einschränken .

- **Conformal-Prediction**: Im Vergleich dazu erfordert Conformal-Prediction weniger spezifisches Domänenwissen. Die Methode kann auf eine Vielzahl von Anwendungsgebieten angewendet werden, ohne dass umfangreiches Vorwissen erforderlich ist.

Zusammenfassend können Bayessche-Methoden, wenn das nötige Domänenwissen und Prior-Wissen vorhanden ist, bessere Ergebnisse liefern als Conformal-Prediction. Dieses liefert jedoch auch ohne umfangreiches Domänenwissen sehr gute Ergebnisse und ist zusätzlich noch besser skalierbar, benötigt aber dafür auch mehr Trainingsdaten [7, 16].

Kapitel 3

Methodik

3.1 Forschungsfragen

Diese Arbeit untersucht, ob Conformal-Prediction für die Vorhersagen nicht-funktionaler Eigenschaften bei konfigurierbaren Softwaresystemen verwendet werden kann und ob es die Vorhersagen sinnvoll verbessern kann. Es ist erforderlich zu prüfen, inwieweit die Einschränkungen der beschriebenen Sampling-Strategien in Bezug auf die Samplegröße und die Vorbedingungen von Conformal-Prediction die Intervallvorhersage beeinflussen. Darüber hinaus muss gezeigt werden, dass Conformal-Prediction einen Mehrwert in diesem Bereich bietet, sei es durch eine abweichende Bewertung von Machine-Learning-Modellen im Vergleich zu bisherigen Metriken oder durch zusätzliche Informationen zu den Einflüssen einzelner Features.

Daraus ergeben sich die folgenden vier konkreten Forschungsfragen:

FF 1 Welche Ergebnisse hinsichtlich der Korrektheit der Coverage liefert Conformal-Prediction bei den für konfigurierbare Softwaresysteme üblichen Samplegrößen?

FF 2 Wie groß ist der Effekt der Verwendung unterschiedlicher Sampling-Methoden aus dem Bereich der Performance-Vorhersage von konfigurierbaren Softwaresystemen auf die Relevanz der Ergebnisse?

FF 3 Korreliert die Gütebewertung der Modelle von Conformal-Prediction mit den bisher verwendeten Scoring-Funktionen?

FF 4 Welche Informationen liefert Conformal-Prediction in Bezug auf die Bestimmung von Unsicherheiten bei der Vorhersage nicht-funktionaler Eigenschaften von konfigurierbaren Softwaresystemen?

Während die ersten beiden Forschungsfragen die Anwendbarkeit von Conformal-Prediction untersuchen, sollen die beiden letzten den Nutzen von

Conformal-Prediction überprüfen.

Die Forschungsfragen 1 und 2 überschneiden sich dahingehend, dass unabhängig von der Exchangebility die Größe der Samples bei den coverage-based Sampling-Strategien (Option-wise und T-wise) durch das betrachtete System festgelegt wird. Es muss daher untersucht werden, ob bei Forschungsfrage 2 die festgelegte Größe für das Ergebnis verantwortlich ist oder ob die möglicherweise nicht erfüllten Vorbedingungen von Conformal-Prediction eine Rolle spielen.

3.2 Metriken

Conformal Prediction liefert, abgesehen von den einfachen Verfahren, unterschiedliche Intervalle für die einzelnen Konfigurationen. Mithilfe von Validationsets Z^{test} können wir fünf Metriken analysieren:

1. Coverage (Cov)
2. Abweichung der Coverage (Cov_{err})
3. FSC Metrik (Gleichung 2.3)
4. Verteilung der Intervallbreiten (absolut und prozentual zur Vorhersage)
5. konditionale Verteilung der Intervallbreiten

Die gewünschte Coverage wird im Voraus festgelegt, in dieser Arbeit auf 90%, bzw. $\alpha = 0.1$. Zur Überprüfung der Anwendbarkeit wird die Abweichung zwischen gewünschter Coverage Cov_{ziel} und empirischer Coverage Cov_{real} betrachtet, wobei ein kleinerer Wert bessere Anwendbarkeit bedeutet. Die empirische Coverage wird berechnet, indem die Anzahl der im Vorhersageintervall liegenden Punkte des Validationsets durch die Gesamtzahl der Punkte in der Menge geteilt wird.Für die Punkte im Validationset ($(c, f(c)) \in Z^{test}$), bestehend aus Konfiguration c und gemessener Performance $f(c)$, ergibt sich die empirische Coverage als:

$$\frac{1}{|C^{test}|} \sum_{c \in C^{test}} \mathbb{1}\{f(c) \in \mathcal{CI}(c)\} \tag{3.1}$$

und die Abweichung bzw. der Fehler der Coverage Cov_{err} wird berechnet als:

$$|Cov_{ziel} - Cov_{real}| \tag{3.2}$$

Die Intervallbreite $\mathcal{CI}(c)$ gibt an, wie unsicher das Modell an der jeweiligen Stelle ist. Der Mittelwert über alle Punkte von C^{test} hinweg gibt bei gleicher Coverage einen Wert zur Güte des Modells:

$$\frac{1}{|C^{test}|} \sum_{c \in C^{test}} \mathcal{CI}(c) \tag{3.3}$$

Hierbei bedeutet eine kleinere mittlere Intervallbreite, dass das Modell im Durchschnitt weniger Unsicherheit aufweist und somit besser vorhersagt.

Zur Bestimmung der Unsicherheit einzelner Features betrachten wir die Coverage und Intervallbreite der Testpunkte, bei denen das Feature ausgewählt ist, im Vergleich zu allen Punkten des Validationsets. Die zu betrachtenden Punkte für die einzelnen Features werden, wie in der FSC-Metrik in Abschnitt 2.3 beschrieben, ausgewählt. Zusätzlich zur Evaluation, ob der tatsächliche Wert innerhalb des Intervalls liegt, wird die Verteilung der Intervallbreiten analysiert. Hierbei wird nicht nur das Minimum betrachtet, sondern auch die Unterschiede zwischen den einzelnen Features und Feature-Interaktionen.

Als Vergleichsmetrik für die Güte der Modelle wird der mittlere absolute Prozentfehler (MAPE) verwendet (siehe Unterabschnitt 2.2.3).

3.3 Experiment-Aufbau

Für die ersten beiden Forschungsfragen werden Experimente mit einer Pipeline durchgeführt,die die notwendigen Daten - Vorhersagewert und Konfidenzintervall pro Datenpunkt in Z^{test} - liefert, aus denen anschließend die weiteren Metriken berechnet werden können. Die dabei betrachteten unabhängigen Variablen sind:

- Conformal-Prediction Algorithmus
- Sampling-Strategie
- Maschine-Learning-Algorithmus
- Konfigurierbares Softwaresystem

Der Aufbau der Pipeline ist in Abbildung 3.1 dargestellt. Im ersten Block werden die jeweiligen Sampling-Strategien für die einzelnen konfigurierbaren Softwaresysteme festgelegt. Dazu wird die jeweilige FeatureModels.xml verwendet, die den Konfigurationsraum des Softwaresystems beschreibt. Hierbei wird berücksichtigt, ob der später verwendete Conformal-Prediction-Algorithmus ein zusätzliches Konfigurationsset benötigt, wie in dem Unterabschnitt 2.3.2 erklärt. Dieses Set kann auch eine andere Sampling-Strategie

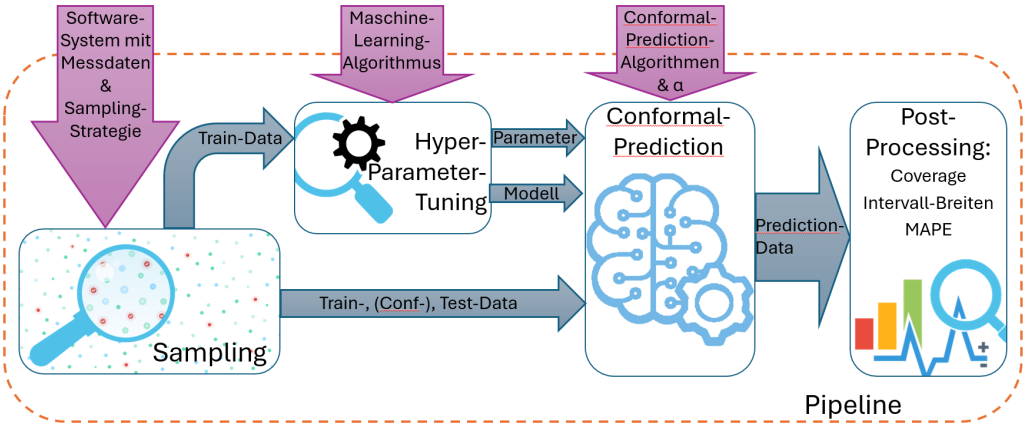


Abbildung 3.1: Schematischer Aufbau der Pipeline zum durchführen der Experimente

als das Trainingsset nutzen. In unserem Fall verwenden wir für das Konfigurationsset immer eine random-based Strategie, da die Vorteile von coverage-based Strategien nicht direkt auf die Konfiguration von CONFORMALIZED-QUANTIL-REGRESSION übertragbar sind. Es wird zudem darauf geachtet, dass die Mengen disjunkt bleiben. Um die Vergleichbarkeit der coverage-based und random-based Strategien zu gewährleisten, wird die Anzahl der Samples bei den random-based Strategien an die der coverage-based angepasst. Beispielsweise wird die Anzahl der Samples beim distance-based Sampling auf 45 gesetzt, wenn ein System 45 Messpunkte beim option-wise Sampling benötigt. Falls ein induktiver CONFORMAL-PREDICTION Ansatz mit einem transduktiven verglichen wird, wird die Größe des Trainingssets des transduktiven Ansatzes auf die Gesamtgröße des Trainings- und Kalibrierungssets des induktiven Algorithmus gesetzt. Abschließend wird das Validierungsset mit einer festgelegten Größe von 400 zufällig und disjunkt zu den anderen Mengen gesampelt. Alle Konfigurationen der genutzten Systeme wurden zuvor vollständig gemessen, sodass die Daten direkt eingelesen werden können. Eine Übersicht über die Systemgrößen zeigt Tabelle 3.1.

Im zweiten Block werden mithilfe des Trainingssets die optimalen Modellparameter durch eine Zufallssuche, wie in Unterabschnitt 2.2.4 erläutert, bestimmt. Der zu optimierende Wert ist der MAPE, und die verwendeten Modelle sind die in Unterabschnitt 2.2.2 beschriebenen: Linear-Lasso-Regression, Decision-Tree, Random-Forest und KernelRidge-Regression.

Der dritte Block führt die drei in Unterabschnitt 2.3.3 aufgeführten CONFORMAL-PREDICTION-Algorithmen aus: CROSSVALIDATION+, JACKKNIFE+

Tabelle 3.1: Systemübersicht mit Sampling-Größen für Option-wise (OW) und T-wise (T2,T3), wenn numerische Optionen vorhanden, dann zusammen mit Placket-Burman mit Werten 3(level) und $|O_{num}| + 1$(measurements). Außerdem die Gesamtanzahl der Konfigurationen C^s die für jedes System s vorliegen.

System	Anzahl Optionen		Sampling Größen			Anzahl Konfigurationen
	$\|O_{Bin}\|$	$\|O_{Num}\|$	OW	T2	T3	$\|C^s\|$
x264	22	-	15	101	407	4608
hsqldb	18	-	13	73	235	865
Apache	19	-	10	48	144	581
7z	11	3	81	279	531	68642
nginx	14	2	39	213	693	4417
vp8	10	4	55	140	208	2736
postgreSQL	6	3	54	144	234	864
lrzip	10	3	81	243	387	5184

und Conformalized-Quantil-Regression. Der induktive Ansatz Conformalized-Quantil-Regression kann das im zweiten Block berechnete Machine-Learning-Modell direkt nutzen und benötigt nur das Konfigurationsset, während die anderen Ansätze lediglich die besten Parameter übernehmen und das Modell auf dem Trainingsset neu trainieren müssen. Alle Ansätze treffen dann eine Vorhersage auf dem Validierungsset, die die Punktvorhersage der Modelle sowie die entsprechenden Intervalle ausgibt.

Im letzten Block werden die erforderlichen Kenngrößen, wie die empirische Coverage Cov_{real}, der Coverage-Fehler Cov_{err}, konditionale Coverage pro Feature und zweifach Feature-Interaktion, sowie die verschiedenen Verteilungen der Intervallbreiten und der MAPE auf dem Validierungsset berechnet.

Um eine allgemeingültige Aussage treffen zu können, wird die Pipeline für jede Kombination aus System und Sampling-Strategie 30 Mal ausgeführt. Dabei werden alle gültigen Kombinationen von Maschine-Learning-Modellen und Conformal-Prediction Verfahren auf den generierten Trainings-, Konfigurations- und Validierungssets getestet. Zusätzlich werden auf den Sets zur Vergleichbarkeit für Intervall-Prediction die Bayesschen Methoden aus Unterabschnitt 2.2.6 ausgeführt.

3.3.1 Implementation

Die Implementierung der Pipeline erfolgte in Python 3.11 und ist auf GitHub[1] verfügbar.

[1] `https://github.com/Stefan577/Masterthesis-Pipeline`

Im ersten Schritt der Pipeline, der Generierung der Samplingsets, wird SPLConqueror[2] verwendet. Diese Sets werden anschließend mit den vorhandenen Messdaten verknüpft und in einer strukturierten Ergebnisordnerhierarchie für die Reproduzierbarkeit gespeichert.

Der zweite Schritt wird mit der RandomSearch aus der Scikit-Learn Bibliothek[3] durchgeführt. Aus dieser Bibliothek stammen auch die MAPE-Metrik und die verwendeten Machine-Learning-Modelle, mit Ausnahme des Decision-Tree Modells, das für die Conformalized-Quantil-Regression benötigt wird, wird und aus der LightGBM Bibliothek[4] entnommen wurde.
Die für den nächsten Schritt benötigten Funktionalitäten von Conformal--Prediction stammen aus dem in Unterabschnitt 2.3.3 vorgestellten MAPIE Framework[5]. Dieses umfasst die einzelnen Conformal-Prediction Algorithmen inklusive der Conformity Scores sowie die Funktionen zur Berechnung der Coverage.

Im letzten Schritt der Pipeline werden die Validierungssets mit den Conformal-Prediction Modellen vorhergesagt und zusammen mit den in Abschnitt 3.2 genannten Metriken in den Ergebnisordnern gespeichert.

Zusätzlich stehen mehrere Jupiter-Notebooks zur Exploration der Daten im entsprechendem Repository[6] zur Verfügung.

3.4 Operationalisierung

FF 1 Zur Beantwortung der ersten Forschungsfrage analysieren wir die empirische Coverage Cov_{real} und den Coverage-Fehler Cov_{err}. Hierfür nutzen wir die Random-Sampling-Strategie, da sie, wie das zufällig Ziehen ohne Zurücklegen aus Unterabschnitt 2.3.2, die Voraussetzung der *Exchangebility* erfüllt. Dies ermöglicht uns, den Einfluss der Samplegröße zu untersuchen. Dabei betrachten wir sowohl die absolute Größe der Samplesets als auch deren relative Größe im Verhältnis zur Anzahl an möglichen Konfigurationen C^s eines Systems.

Die Anwendbarkeit von Conformal-Prediction wird anhand des Grenzwerts des Coverage-Fehlers bestimmt. Dieser Grenzwert leitet sich aus den Coverage-Garantien für CrossValidation+ und Jackknife+ ab. Der Fehler muss kleiner als α sein, um die Coverage-Garantien nicht zu verletzen. Wir analysieren CrossValidation+ und Jackknife+ gemeinsam und prüfen, ob sie sich unterschiedlich verhalten, was durch den Kruskal-Wallis-Signifikanz-

[2] `https://github.com/se-sic/SPLConqueror`
[3] `https://scikit-learn.org`
[4] `https://lightgbm.readthedocs.io`
[5] `https://mapie.readthedocs.io`
[6] `https://github.com/Stefan577/Masterthesis-Explore`

test getestet wird. Dieser Test eignet sich für metrische, nicht normalverteilte Daten in unserem Versuchsaufbau.

CONFORMALIZED-QUANTIL-REGRESSION wird separat betrachtet, da hierfür ein zusätzliches Konfigurationsset erforderlich ist. Für diese Methode überprüfen wir insbesondere den Einfluss der Änderung des Trainingssets als auch die Änderung der Größe des Kalibrierungssets, welche in anderen Arbeiten bereits untersucht wurde [1].

FF 2 Zur Beantwortung der zweiten Forschungsfrage verwenden wir alternative Sampling-Strategien, wie in Unterabschnitt 2.1.3 beschrieben, anstelle der Random-Sampling-Strategie. Wir untersuchen, inwieweit sich die Verteilung der Coverage bzw. ihres Fehlers Cov_{err} im Vergleich zur Random-Sampling-Strategie unterscheidet. Sollten die Verteilungen ähnlich sein, wird erneut mit dem Kruskal-Wallis-Test überprüft, ob überhaupt ein signifikanter Unterschied besteht. Dabei vergleichen wir stets gleiche Samplegrößen mit unterschiedlichen Sampling-Strategien. Auch hier wird CONFORMALIZED-QUANTIL-REGRESSION separat betrachtet, wobei der Fokus auf dem Trainingsset liegt, da die Vorteile von Option-wise und T-wise sich primär auf das Training von Modellen beziehen.

FF 3 Die dritte Forschungsfrage untersucht zunächst, ob die mittlere prozentuale Intervallbreite der Konfidenzintervalle des Validierungssets mit dem MAPE korreliert. Dafür werden sowohl der Pearson-Korrelationskoeffizient (r-Test) zur Bestimmung einer linearen Korrelation als auch der Spearman-Korrelationskoeffizient (ρ-Test) zur Bestimmung einer monotonen Korrelation verwendet. Die Daten, auf denen diese durchgeführt werden, sollen pro Softwaresystem betrachtet werden. Diese Tests werden pro Softwaresystem durchgeführt, sowohl unter Einbeziehung aller Läufe als auch nur der Läufe mit korrekter Vorbedingung (Random-Sampling).

Im zweiten Schritt wird die empirische Coverage Cov_{real} mit einbezogen. Hierfür trainieren wir pro System ein lineares Modell, wobei die Eingabewerte die mittlere prozentuale Intervallbreite und die empirische Coverage und die Zielvariable der MAPE sind. Wir nutzen die Läufe mit korrekter Vorbedingung, die in einem 80/20-Split in Trainings- und Testdaten aufgeteilt werden. Um zu prüfen, ob die empirische Coverage die Vorhersage verbessert, wird auch ein Modell ohne diese Variable trainiert und die Modelle anhand des quadratischen Fehlers beim Validierungsset verglichen.

FF 4 Zur Beantwortung der vierten Forschungsfrage verwenden wir ein Toy-System. Hierfür nutzen wir ein Featuremodell eines der verwendeten Softwaresysteme und hinterlegen eine bekannte Funktion zur Berechnung einer nicht

funktionalen Eigenschaft. Diese Funktion beinhaltet Einflüsse einzelner Optionen, 2-fach Interaktionen und eine zufällige Komponente, um die Messungenauigkeit zu simulieren.

Wir verwenden ein lineares Lasso-Modell ohne Erweiterung des Eingabevektors, sodass die Interaktionen nicht gelernt werden können (siehe Unterabschnitt 2.2.2). Die restliche Pipeline bleibt unverändert. Am Ende untersuchen wir, ob anhand der konditionalen Coverage oder der Verteilung der konditionalen Intervallbreiten erkennbar ist, welche Features oder Feature-Interaktionen nicht gelernt wurden, um zu überprüfen, ob Conformal-Prediction uns die Unsicherheiten eines nicht gelernten Features oder einer nicht gelernten Interaktion aufzeigen kann.

Kapitel 4

Auswertung

4.1 Ergebnisse

4.1.1 FF1

Um die erste Forschungsfrage zu beantworten, analysieren wir die Ergebnisse der Experimente mit der Random-Sampling-Strategie. Random-Sampling erfüllt die Voraussetzungen aller Conformal-Prediction-Algorithmen, sodass wir uns auf die Größe der Samples konzentrieren können.Die Samplegrößen wurden je nach Softwaresystem basierend auf den Größen der Coverage-based Sampling-Strategien aus Tabelle 3.1 festgelegt. Die Größe Option-wise wird als Random-1, die T-wise-Größen als Random-2 und Random-3 bezeichnet.

Einzelsystem In Abbildung 4.1 ist die Verteilung der Coverage-Fehler Cov_{err} für das Softwaresystem Apache dargestellt. Die Zeilen des Plots zeigen die Conformal-Prediction-Methoden: CrossValidation+ in der ersten Spalte und Jackknife+ in der zweiten. Conformalized-Quantil-Regression wird aufgrund des zusätzlichen Konfigurationssets separat betrachtet. Die Spalten der Abbildung 4.1 repräsentieren die Machine Learning-Algorithmen: Entscheidungsbaum, KernelRidge, Linear-Lasso-Modell und RandomForest. Jeder Subplot zeigt die Verteilungen der Coverage-Fehler über 30 Läufe für die Samplegrößen Random-1, 2 und 3.Es ist erkennbar, dass mit zunehmender Größe des Trainingssamples die Verteilung der Coverage-Fehler sich immer mehr auf den Bereich um Null konzentriert, also der Coverage-Fehler im Schnitt kleiner wird. Random-3 überschreitet in keinem Fall den Grenzwert von $\alpha = 0.1$, Random-2 zeigt einige Ausreißer über dem Grenzwert, und Random-1 überschreitet ihn häufig.

Nach dem Shapiro-Wilk-Test mit Bonferroni-korrigiertem p-Wert von 0.002

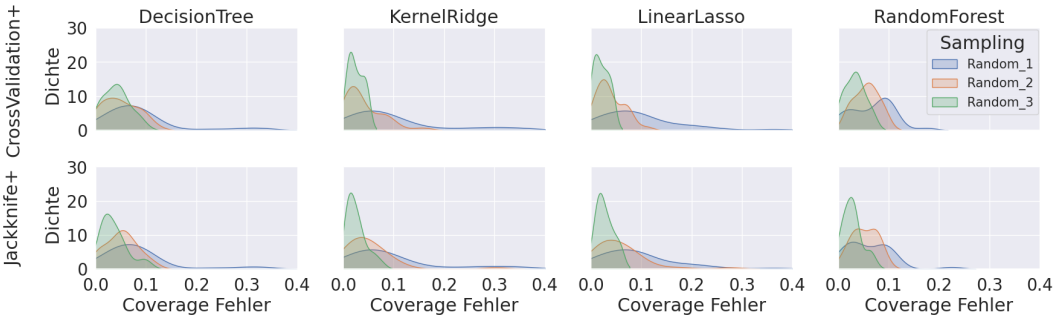


Abbildung 4.1: Cov_{err} Verteilung für APACHE mit Random-Sampling

Tabelle 4.1: P-Werte Kruskal-Wallis Signifikantstest zum Vergleich von Cov_{err} zwischen CROSSVALIDATION+ und JACKKNIFE+

	DecisionTree	KernelRidge	LinearLasso	RandomForest
Random_1	1	1	1	0.57
Random_2	0.62	0.24	0.11	0.62
Random_3	0.40	0.76	0.14	0.25

sind 10 der 24 Verteilungen aus Abbildung 4.1 nicht normalverteilt. Um die Verfahren CROSSVALIDATION+ und JACKKNIFE+ zu vergleichen, verwenden wir den Kruskal-Wallis Signifikanztest. Die Ergebnisse dieses Tests sind in Tabelle 4.1 dargestellt. Da alle p-Werte über 0.05 liegen, gibt es keinen signifikanten Unterschied zwischen CROSSVALIDATION+ und JACKKNIFE+ in Bezug auf den Coverage-Fehler.

Übersicht aller Systeme Abbildung 4.2 stellt die empirische Coverage (Cov_{real}) des RandomForest-Modells für alle Systeme und die drei Random-Sampling-Größen dar. Die Systeme sind dabei nach der Größe des Random-1-Samples sortiert, beginnend mit 7Z, das das größte Sampleset aufweist, und endend mit APACHE, das das kleinste Sampleset besitzt. Für Random-2 und 3 ändert sich diese Reihenfolge (vergleiche Tabelle 3.1). APACHE, das System mit den kleinsten absoluten Samplegrößen, zeigt ein ähnliches Verhalten wie die anderen Systeme: Mit zunehmender Samplegröße nähert sich die Verteilung der Coverage zunehmend der Ziel-Coverage Cov_{Ziel} von $1 - \alpha$, die in der Abbildung als grün gestrichelte Linie dargestellt ist.

Betrachtet man in Abbildung 4.2 das System 7Z mit einer Samplingset-Größe von 531, was weniger als 1% der Gesamtzahl der möglichen Konfigurationen ($|C^{7Z}|$) entspricht, und HSQLDB mit einer Samplingset-Größe von 235, was 27% der Gesamtzahl der möglichen Konfigurationen ($|C^{HSQLDB}|$) ausmacht, so erkennt man, dass eine größere absolute Anzahl zu einer schmaleren

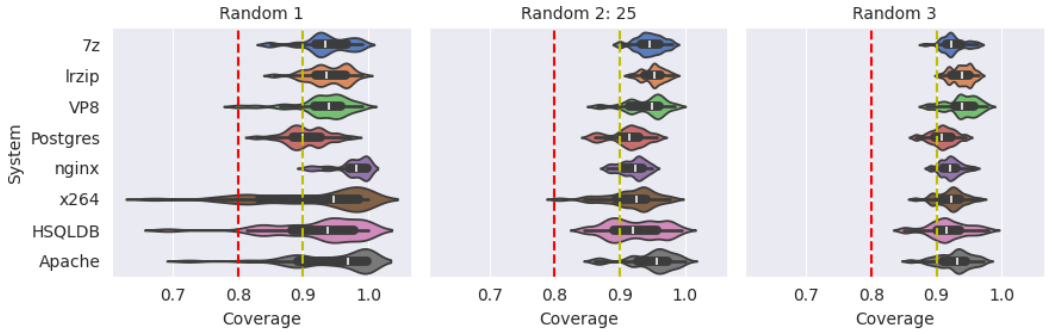


Abbildung 4.2: Coverage-Verteilung für CrossValidation+ mit RandomForest und Random-Sampling. Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

Verteilung führt als eine größere relative Anzahl im Verhältnis zu den möglichen Konfigurationen.

Die rot gestrichelte Linie repräsentiert die theoretisch garantierte minimale Coverage von $1 - 2\alpha$, die für die Verfahren CrossValidation+ und Jackknife+ als theoretisch minimale Coverage angegeben wird. Bei der Random-1-Samplegröße wird diese von der Hälfte der Systeme überschritten. Bei Random-2 liegt das System x264 als Ausreißer genau auf dieser Linie, und bei Random-3 wird die theoretisch minimale Coverage in keinem einzigen Fall unterschritten.

Conformalized-Quantil-Regression Abbildung 4.3 illustriert die Verteilung der Coverage bei der Conformalized-Quantil-Regression. Abbildung 4.3a zeigt die Verteilung bei variierenden Trainingsset-Größen und konstantem Konfigurationsset, während Abbildung 4.3b die Verteilung bei konstantem Trainingsset und unterschiedlichen Konfigurationsset-Größen $|C^{calib}| \in 25, 75, 150$ darstellt. Die grün gestrichelte Linie repräsentiert sowohl die angestrebte Coverage als auch die theoretisch garantierte minimale Coverage. Zur besseren Vergleichbarkeit mit den anderen Abbildungen wird die rot gestrichelte Linie bei $1 - 2\alpha$ angezeigt.

Das Konfigurationsset in Abbildung 4.3a hat eine Größe von 25. Der Algorithmus erfordert eine Mindestgröße von $\frac{2}{\alpha} = 20$; jedoch führte eine Setgröße von genau 20 in einigen Läufen zu Fehlern, weshalb der Wert auf 25 festgelegt wurde. In diesem Fall zeigen nicht alle Systeme das Verhalten wie bei CrossValidation+ oder Jackknife+. Eine deutliche Verbesserung durch die Vergrößerung des Trainingssets ist nicht überall erkennbar.

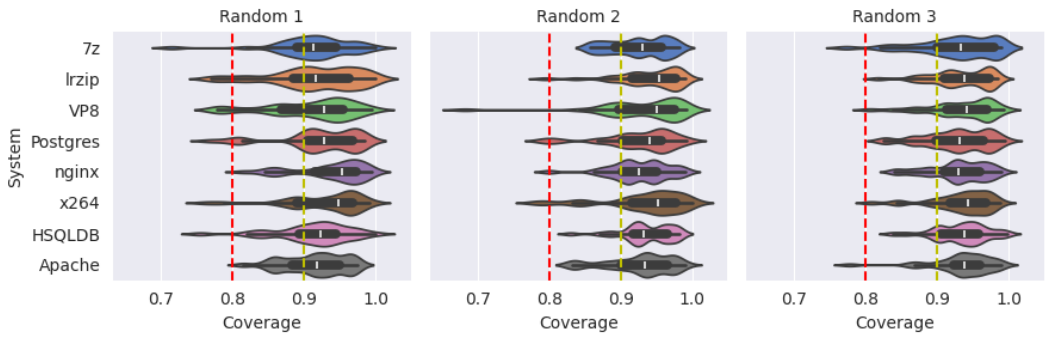
Der Kruskal-Wallis-Signifikanztest mit Bonferroni-Korrektur zeigt bei sechs Systemen einen signifikanten Unterschied zwischen den Sampling-Größen (siehe Tabelle 4.2). Der Unterschied in den Verteilungen ist im Vergleich zu den Methoden ohne Kalibrierungsset, wie zum Beispiel bei CrossValidation+

Tabelle 4.2: P-Werte Kruskal-Wallis Signifikanztest zum Vergleich von Cov_{err} bei CONFORMALIZED-QUANTIL-REGRESSION mit verschiedenen Random-Sampling-Größen. bonferroni-korregierter p-Wert: 0.00625

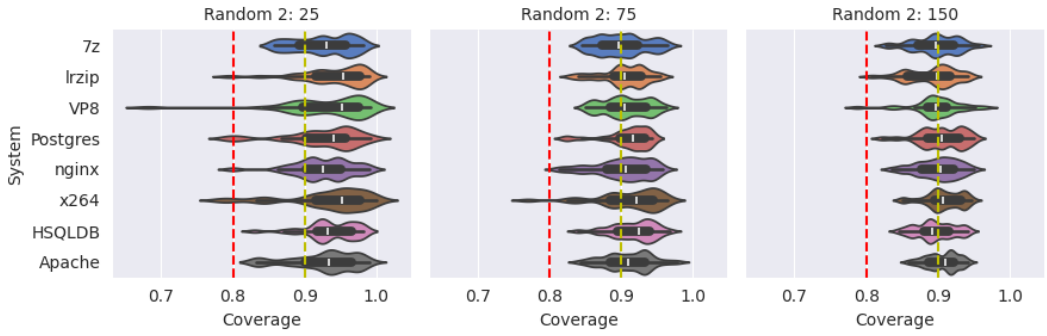
System	7z	APACHE	HSQLDB	POSTGRESQL	VP8	LRZIP	NGINX	X264
p-Wert	9.64e-13	2.29e-25	3.76e-29	0.986	6.21e-25	2.20e-22	1.66e-17	0.015

in Abbildung 4.2, sehr gering.

Abbildung 4.3b zeigt, dass sich die empirische Coverage bei zunehmender Größe des Kalibrierungssets der gewünschten Coverage immer mehr annähert. Bei einer Kalibrierungsgröße von 150 liegen die Mittelwerte aller Verteilungen nahe an der gewünschten Coverage, jedoch gibt es bei den Systemen VP8 und LRZIP einzelne Ausreißer, die auch über $1 - 2\alpha$ hinausgehen. Betrachtet man beispielsweise das System APACHE, das keine Ausreißer aufweist, so liegen die Coverage-Werte bei $|C^{calib}| = 25$ zwischen 0.82 und 0.99, bei $|C^{calib}| = 75$ zwischen 0.84 und 0.98, und bei $|C^{calib}| = 150$ zwischen 0.86 und 0.94.



(a) Konstantes Kalibrierungsset, $|C^{calib}| = 25$.



(b) Konstantes Trainingsset, Random 2. $|C^{calib}| \in \{25, 75, 150\}$

Abbildung 4.3: Coverage-Verteilung für CONFORMALIZED-QUANTIL-REGRESSION mit LGBM und Random-Sampling. Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

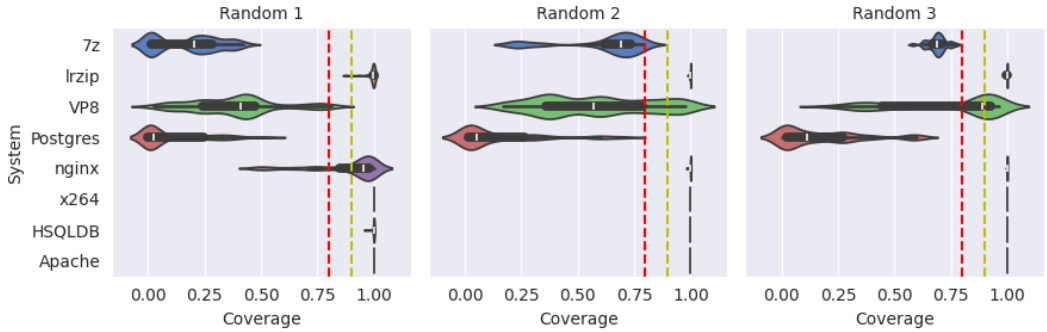


Abbildung 4.4: Coverage-Verteilung für Bayessche-Regression.
Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

Bayessche-Regression Abbildung 4.4 zeigt die Verteilung der Coverage für die als Baseline genutzte Bayessche Regression. Die grün gestrichelte Linie repräsentiert die gewünschte Coverage, während die rot gestrichelte Linie erneut zur besseren Vergleichbarkeit der Grafiken eingezeichnet ist, jedoch für die Bayessche Regression keine Relevanz besitzt.

Aus der Grafik wird ersichtlich, dass kein einheitliches Verhalten zwischen allen Systemen feststellbar ist. Beispielsweise nimmt das System VP8 nahezu jeden möglichen Wert der Coverage von 0% bis 100% an, wohingegen die Systeme Apache und x264 stets eine Coverage von 100% aufweisen.

Einzellauf Vergleich Die Abbildungen 4.5 und 4.6 zeigen die Vorhersagen der Konfigurationen des Validationsets für einen zufällig ausgewählten Lauf der Systeme VP8 und Apache. Auf der X-Achse ist der gemessene Wert dargestellt, während auf der Y-Achse der vorhergesagte Wert zusammen mit dem Konfidenzintervall angezeigt wird. Die rot gestrichelte Linie repräsentiert die optimale Vorhersage; sobald das eingezeichnete Intervall diese Linie kreuzt, liegt der echte Wert innerhalb des Intervalls.

Für jeden Lauf werden die Vorhersagen der Methoden CrossValidation+ mit dem Linear-Lasso-Modell, Conformalized-Quantil-Regression mit dem Linear-Modell und Bayessche-Regression gezeigt. Ein Lauf umfasst dabei ein Trainings- und ein Validationsset mit denen alle Modelle trainiert und Validiert wurden. Zur besseren Lesbarkeit wurden nur 100 zufällig ausgewählte Punkte aus dem Validationsset eingezeichnet.

Abbildung 4.5 verdeutlicht, dass alle drei Methoden ähnliche Größenordnungen von Intervallgrößen und Abweichungen zur optimalen Vorhersage aufweisen. Die Coverage beträgt von links nach rechts 92%, 99,25% und 96,5%. Die Conformalized-Quantil-Regression überschätzt das Intervall deutlich nach oben, dafür weniger stark nach unten. Die anderen beiden Methoden schätzen

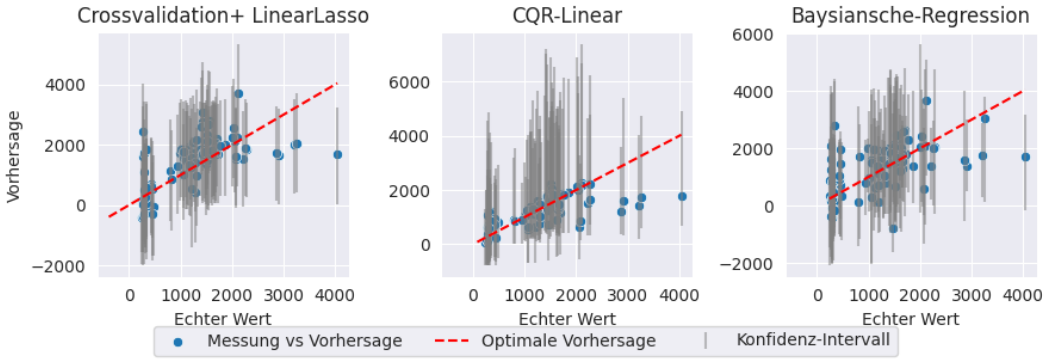


Abbildung 4.5: Scatterplots: Messungen vs Vorhersage mit Konfidenz-Intervallen von einem Lauf mit Random-3-Sampling für VP8

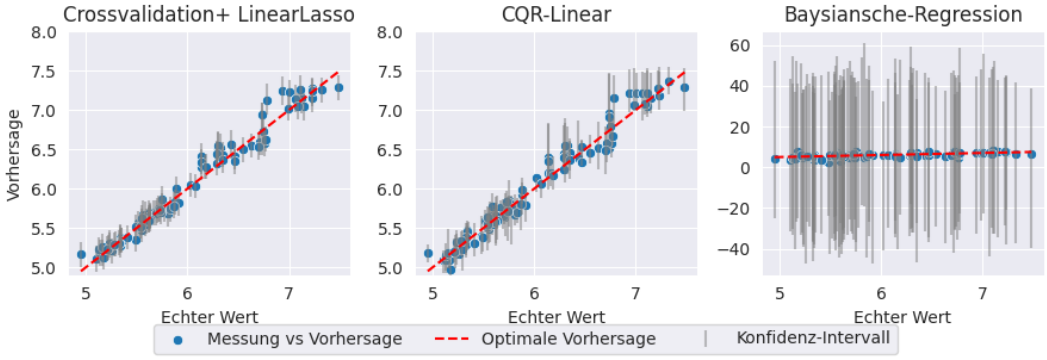


Abbildung 4.6: Scatterplots: Messungen vs Vorhersage mit Konfidenz-Intervallen von einem Lauf mit Random-3-Sampling für Apache. Zu beachten ist die unterschiedliche Skalierung der Y-Achsen.

das Intervall im Mittel in beide Richtungen gleich stark. Abbildung 4.6 verwendet aufgrund der extremen Überschätzung der Intervalle durch die Bayessche Regression keine geteilte Skalierung der Y-Achse. Während alle drei Methoden ähnlich gute Punktvorhersagen liefern, passen nur die Intervalle von Cross-Validation+ und Conformalized-Quantil-Regression dazu, mit einer Coverage von 88,5% (CV+) und 91,75% (CQR), was nahe an der gewünschten Coverage liegt. Die Bayessche Regression liefert hingegen Intervallbreiten, die bis zu zwanzigfach größer sind als die Spannweite der tatsächlichen Werte, wodurch die Coverage bei 100% liegt.

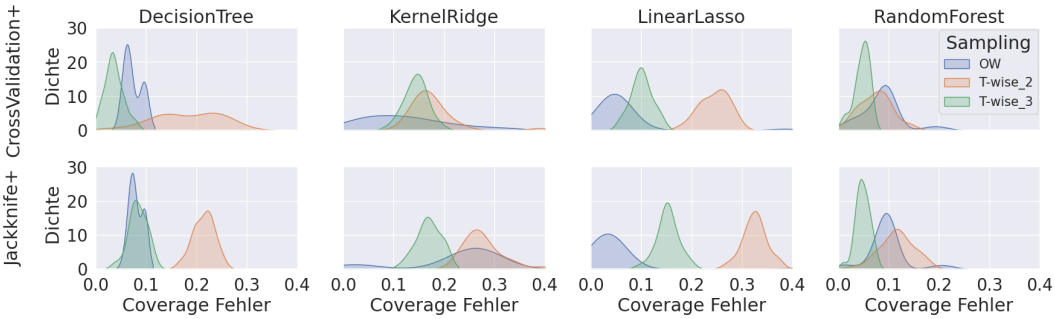


Abbildung 4.7: Cov_{err} Verteilung für x264 mit Option-wise-/T-wise-Sampling

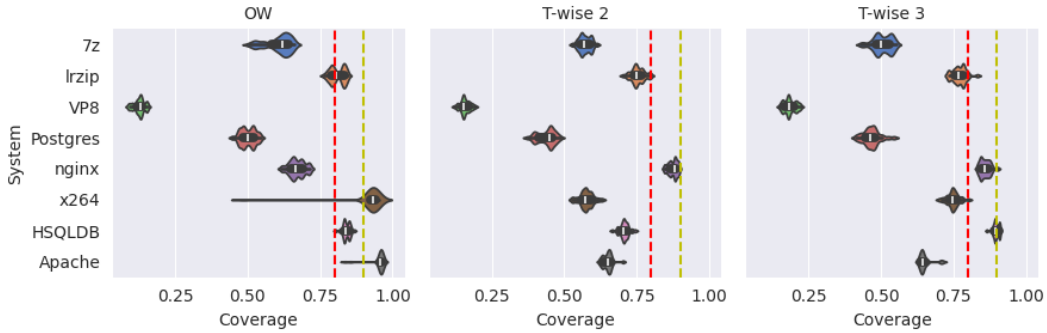


Abbildung 4.8: Coverage-Verteilung für JACKKNIFE+ mit Linear-Lasso-Modell und Option-wise/T-wise-Sampling. Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

4.1.2 FF2

Für die zweite Forschungsfrage betrachten wir die Ergebnisse der Sampling-Strategien, Option-wise, T-wise und Distance-based. Diese werden hinsichtlich der Verteilungen der Coverage und dem Coverage-Fehler Cov_{err} mit den Ergebnissen aus Unterabschnitt 4.1.1 verglichen.

Coverage-Based Für die Coverage-Based Sampling-Strategien, d.h. Option-wise und T-wise, betrachten wir Abbildung 4.7, die die Verteilung der Coverage-Fehler analog zu Abbildung 4.1 zeigt. Das hierfür genutzte System ist x264. Es ist ersichtlich, dass die Coverage-Fehler nicht mehr umgekehrt proportional zur Größe des Trainingssets verlaufen. Beispielsweise produziert beim Linear-Lasso-Modell das kleinste Trainingsset den geringsten und das mittlere Trainingsset den größten Fehler. Zudem sind die Fehler insgesamt deutlich größer als beim Random-Sampling. Dieses uneindeutige Verhalten zeigt sich bei allen getesteten Systemen, wie in Abbildung 4.8 zu sehen ist. In dieser Abbildung verhält sich nur das System NGINX ähnlich wie beim Random-Sampling, während keines der anderen Systeme dieses Verhalten aufweist. Ein weiterer

Unterschied zum Random-Sampling ist die geringere Varianz der einzelnen Coverage-Verteilungen.

Distance-based Betrachten wir die Verteilung der Coverage-Fehler für Distance-based-Sampling mit den drei Größen des Trainingssets, so zeigt sich ein ähnliches Verhalten wie beim Random-Sampling, wie in Abbildung 4.9 ersichtlich ist. Hier verhält sich der Coverage-Fehler wieder umgekehrt proportional zur Größe des Trainingssets. Der Kruskal-Wallis-Signifikanztest zeigt, dass die Verteilungen von Distance-based- und Random-Sampling für x264 keinen signifikanten Unterschied aufweisen, wie die p-Werte in Tabelle 4.3 belegen.

In Abbildung 4.10 betrachten wir die Coverage des Distance-based-Sampling über alle Systeme hinweg. Alle Parameter außer der Sampling-Strategie wurden wie in Abbildung 4.2 für FF 1 gesetzt. Auffällig ist, dass die Systeme VP8, PostgreSQL und nginx ein abweichendes Verhalten zeigen und stets außerhalb der theoretisch garantierten minimalen Coverage liegen.

Tabelle 4.3: P-Werte Kruskal-Wallis Signifikanztest zum Vergleich von Cov_{err} beim Softwaresystem x264 zwischen den Paaren von Distancebased- und Random-Sampling mit gleicher Sampling-Größe. Bonferroni-korregierter p-Wert: 0.0021

	CrossValidation+				Jackknife+			
	DecisionTree	KernelRidge	LinearLasso	RandomForest	DecisionTree	KernelRidge	LinearLasso	RandomForest
R1 vs D1	0.2971	0.4245	0.9469	0.0066	0.9000	0.3710	0.8591	0.9941
R2 vs D2	0.1100	0.5942	0.8416	0.5008	0.7673	0.7673	0.8765	0.7281
R3 vs D3	0.5535	0.4109	0.7502	0.0696	0.0912	0.2243	0.7784	0.0471

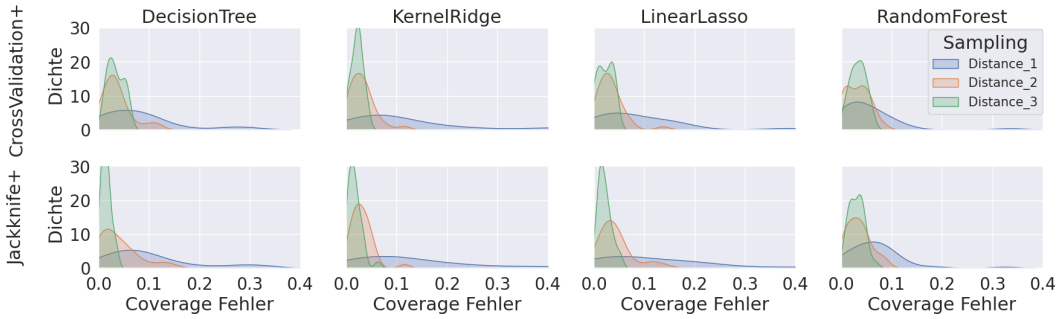


Abbildung 4.9: Cov_{err} Verteilung für x264 mit Distance-based-Sampling

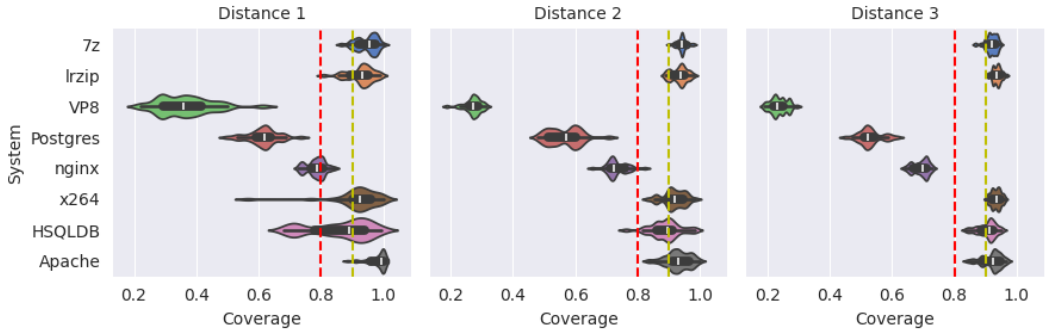


Abbildung 4.10: Coverage-Verteilung für CrossValidation+ mit RandomForest und Distance-based-Sampling. Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

Conformalized-Quantil-Regression Abbildung 4.11 illustriert das Verhalten der Coverage-Verteilung der Conformalized-Quantil-Regression bei den Sampling-Strategien Option-wise und T-wise. Die Verteilungen zeigen ein ähnliches Verhalten wie bei der Conformalized-Quantil-Regression mit Random-Sampling. Dies impliziert, dass die Wahl der Sampling-Strategie des Trainingssets keinen so großen Einfluss hat wie bei CrossValidation+ oder Jackknife+.

Beim Vergleich zwischen Random-Sampling und Option-wise/Twise-Sampling weisen ein Drittel der Verteilungen keinen signifikanten Unterschied auf, wie Tabelle 4.4 zeigt. Betrachtet man die Verteilungen, so liegen die Mittelwerte aller Verteilungen zwischen $1 - \alpha$ und 1, und die Varianz befindet sich, abgesehen von einigen Ausreißern, im gleichen Bereich.

Tabelle 4.4: P-Werte des Kruskal-Wallis Signifikanzstests zum Vergleich der Coverage Verteilung bei Conformalized-Quantil-Regression zwischen den Paaren mit gleicher Sampling-Größe. bonferroni-korregierter p-Wert: 0.0021

Sampling \ System	7z	Apache	HSQLDB	PostgreSQL	VP8	lrzip	nginx	x264
OW vs Random 1	1.43e-4	3.28e-49	0.9670	5.83e-6	1.04e-26	0.2060	4.44e-4	0.0791
T-wise 2 vs Random 2	4.43e-4	0.8700	1.05e-13	3.30e-7	0.0062	2.73e-70	0.0723	5.22e-9
T-wise 3 vs Random 3	0.0373	1.16e-14	4.36e-16	3.24e-15	2.14e-7	0.0089	5.34e-10	2.20e-8

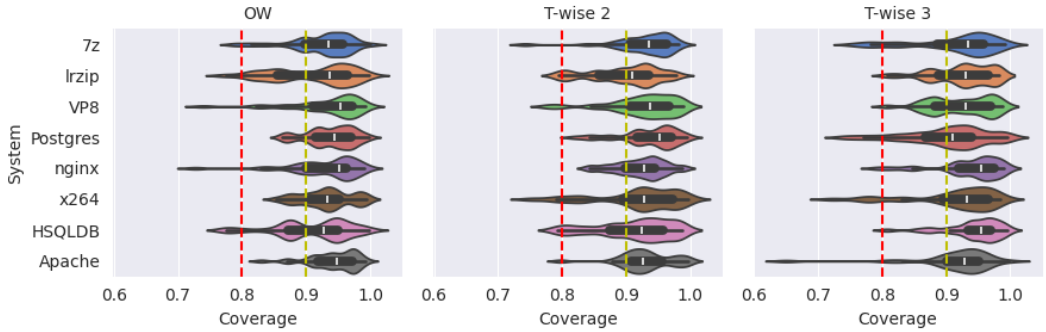


Abbildung 4.11: Coverage-Verteilung für Conformalized-Quantil-Regression mit LGBM und Option-wise-/T-wise-Sampling. $|C^{calib}| = 25$. Grüne Linie: $1 - \alpha$, Rote Linie: $1 - 2\alpha$

4.1.3 FF3

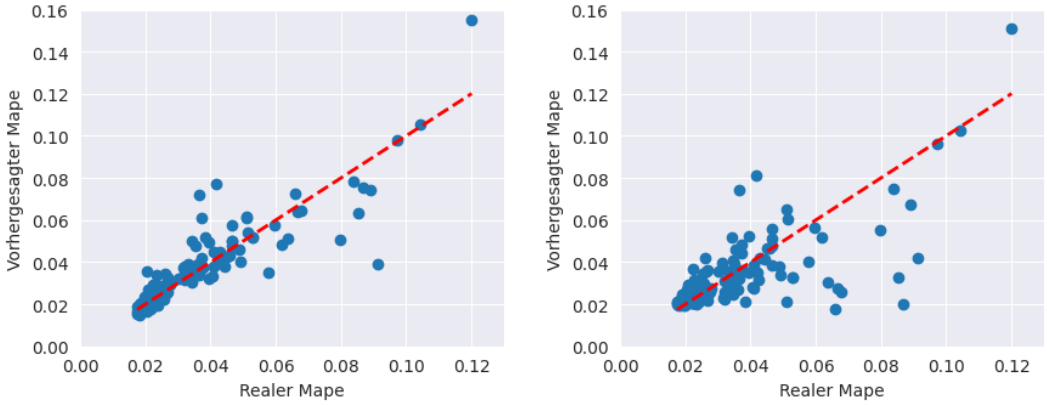
Zur Untersuchung der dritten Forschungsfrage wird zuerst überprüft, ob die mittlere prozentuale Intervallbreite mit dem MAPE korreliert. Dafür wurden der Pearson-r- und Spearman-ρ-Test verwendet. Diese Tests wurden einmal für die Daten aller Sampling-Strategien und einmal spezifisch für die Sampling-Strategien Random-1, -2 und -3 für jedes System durchgeführt. Die Ergebnisse der beiden Tests sind in Tabelle 4.5 dargestellt.

Bei der Betrachtung aller Sampling-Strategien zeigt sich für zwei Systeme eine negative Korrelation, während alle anderen eine positive Korrelation aufweisen. Eine lineare Beziehung ist nicht bei allen Systemen gegeben, wie die Ergebnisse des Pearson-r-Tests zeigen.

Bei Betrachtung der Random-Sampling-Strategien sind die Korrelationskoeffizienten im Allgemeinen höher. Sechs der Systeme weisen eine hohe Korrelation mit $|\rho| \geq 0.5$ auf, ein System zeigt eine mittlere Korrelation mit $0.5 > |\rho| \geq 0.3$, kein System weist eine geringe Korrelation mit $0.3 > |\rho| \geq 0.1$ auf und nur das System VP8 zeigt überhaupt keine Korrelation mit $0.1 > |\rho|$ auf.Bei diesen Sampling-Strategien gibt es nur positive Korrelationen zwischen der mittleren prozentualen Intervallbreite und dem MAPE.

Im zweiten Schritt wird die Korrelation von Coverage und mittlerer prozentualer Intervallbreite zum MAPE mithilfe eines linearen Regressionsmodells untersucht. Dieses Modell wird mit den beiden Werten trainiert, um danach den MAPE vorherzusagen. Abbildung 4.12a zeigt diese Vorhersage für das System x264, wobei auf der X-Achse der echte MAPE und auf der Y-Achse der vorhergesagte MAPE eingetragen sind. Für Abbildung 4.12b wurde nur die Intervallbreite genutzt.

Die Abbildungen zeigen, dass die Vorhersage mithilfe beider Metriken eine geringe Abweichung zur optimalen Vorhersage (rot gestrichelte Linie) aufweist,



(a) Eingabe: Coverage, Intervallbreite
$R^2 - Score$: 0.80

(b) Eingabe: Intervallbreite
$R^2 - Score$: 0.55

Abbildung 4.12: Vorhersage von MAPE mithilfe einer linearen Regression für die Random-Sampling-Strategie von dem System x264

wie auch der $R^2 - Score$ von 0.8 zeigt. Die Vorhersage unter Verwendung von nur der Intervallbreite ergibt mit 0.55 einen deutlich schlechteren $R^2 - Score$.

Tabelle 4.6 zeigt die verschiedenen Koeffizienten, die das lineare Modell gelernt hat, wenn jede Kombination aus Modell und Conformal-Prediction-Methode für x264 getrennt betrachtet wird. Die Werte der einzelnen Koeffizienten unterscheiden sich teilweise sehr stark zwischen verschiedenen Modellen. Innerhalb desselben Modells liegen die Werte in einem ähnlichen Bereich. Wenn man das System bei gleichem Modell ändert, dann ändern sich auch die Koeffizienten. Bei Änderung des Systems innerhalb desselben Modells ändern sich auch die Koeffizienten. Zum Beispiel betragen die Koeffizienten bei Apache für das Linear-Lasso-Modell mit CrossValidation+: Coverage: -0.0089 und Intervall: 0.0203.

Tabelle 4.5: Ergebnisse Pearson-r und Spearman-ρ Korrelationstest zwischen dem Mape und der Intervallbreite für einzelne Softwaresysteme. Links über alle genutzten Sampling-Strategien und rechts über Random-Sampling mit allen Samplegrößen. Hohe Korrelation := Grün, mittlere Korrelation:= Türkis,
geringe Korrelation := Orange, keine Korrelation := Rot

	Alle Sampling-Strategien				Random-Sampling-Strategien			
	Pearson		Spearman		Pearson		Spearman	
System	r	p-Wert	ρ	p-Wert	r	p-Wert	ρ	p-Wert
HSQLDB	0.814	0	0.565	4.37e-273	0.819	1.12e-262	0.606	2.40e-109
Apache	0.784	0	0.825	0	0.814	4.16e-257	0.859	1.88e-315
x264	0.272	3.75e-56	0.740	0	0.645	3.10e-128	0.765	1.57e-208
7z	0.110	8.67e-6	0.567	3.30e-138	0.766	2.31e-105	0.875	1.61e-171
lrzip	0.012	6.27e-1	0.698	2.28e-237	0.600	4.80e-54	0.758	7.19e-102
nginx	0.392	1.44e-60	0.377	5.59e-56	0.383	2.48e-20	0.694	8.27e-79
PostgreSQL	-0.129	1.73e-7	-0.219	5.28e-19	0.251	3.45e-9	0.343	2.35e-16
VP8	-0.018	3.37e-1	-0.389	4.72e-98	-0.002	9.62e-1	-0.049	1.45e-1

Tabelle 4.6: Koeffizienten der Vorhersage von Mape bei x264 für unterschiedliche Modelle und Conformal-Prediction Methoden.

Modell / Koeff.	Linear Lasso		DesisionTree		RandomForest		KernelRidge		Lin	LGBM
	CV+	Jackk+	CV+	Jackk+	CV+	Jackk+	CV+	Jackk+	CQR	
Coverage	-0.0058	-0.0067	-0.0051	-0.0052	-0.0026	-0.0028	-0.0037	-0.0050	-0.0080	-0.0028
Intervall	0.0079	0.0073	0.0061	0.0072	0.0086	0.0096	0.0094	0.0090	0.036	0.0078

4.1.4 FF4

Zur Beantwortung der vierten Forschungsfrage analysieren wir ein Toy-System. Zur Bestimmung des vorherzusagenden Wertes verwenden wir eine lineare Funktion mit den in Tabelle 4.7 angegebenen Koeffizienten. Alle nicht aufgeführten Features haben in dieser Funktion sowie in der vom Lineare-Lasso-Modell gelernten Funktion den Wert Null. Das Modell kann lediglich Einzel-Einflüsse erlernen und nicht die beiden Interaktionen, die in der gegebenen Funktion vorkommen. Die Spalten in der Tabelle sind grün markiert, wenn der Koeffizient mit einem Fehler von weniger als 5% korrekt erlernt wurde, orange bei einem Fehler von weniger als 30% und rot bei größeren Fehlern. Das Feature *ref_5* wird trotz des erlernten Koeffizienten als korrekt betrachtet, während die Features *no_8x8dct* und *no_fast_pskip* als falsch erlernt gelten. Als Sampling-Strategie wurde Random-3 genutzt und als Conformal-Prediction-Methode die Jackknife+ Methode verwendet.

Wir betrachten die konditionale Coverage in Tabelle 4.8 und die konditionale Verteilung der Intervallbreiten in Abbildung 4.13a. Der genutzte Referenzpunkt ist das *root*-Feature, welches immer aktiv ist.

Die konditionale Coverages zeigen nur minimale Schwankungen ($< 5\%$ von

Tabelle 4.7: Gegebene und gelernte Einflüsse der Funktion für das Toy-System basierend auf dem Featuremodell von x264

	root	*no_8x8dct*	*no_deblock*	*no_fast_pskip*	*no_weightb*	*rc_lookahead_20*	*rc_lookahead_40*	*rc_lookahead_60*	*ref_5*	*core1*	*core2*	*core4*	*no_8x8dct + no_fast_pskip*	*no_fast_pskip + no_weightb*
gegeben	4	0	-0,5	0	0,75	0,25	0,35	0,45	0	0,2	0,3	-0,5	0.75	0.5
gelernt	4,07	0,35	-0,48	0,61	0,96	-0,042	0	0,11	0,003	0,13	0,24	-0,48	-	-

Cov_{Ziel}) in beide Richtungen auf, unabhängig davon, wie die einzelnen Features gelernt wurden. Bei den konditionalen Intervallbreiten weisen nur die acht in Tabelle 4.9 gelb markierten Features und Interaktionen einen signifikanten Unterschied zum *root*-Feature auf. Davon sind die Features *no_deblock* und *core4* korrekt erlernt, während die restlichen Features falsche Koeffizienten aufweisen. Betrachtet man die Verteilungen dieser acht Features und Interaktionen in Abbildung 4.13a, so haben *no_deblock* und *core4* im Mittel größere prozentuale Intervalle, die anderen kleinere. Die Varianz ist bei *no_weightb*, *core1*, *core2*, *core3* und den Interaktionen geringer als bei den anderen Features.

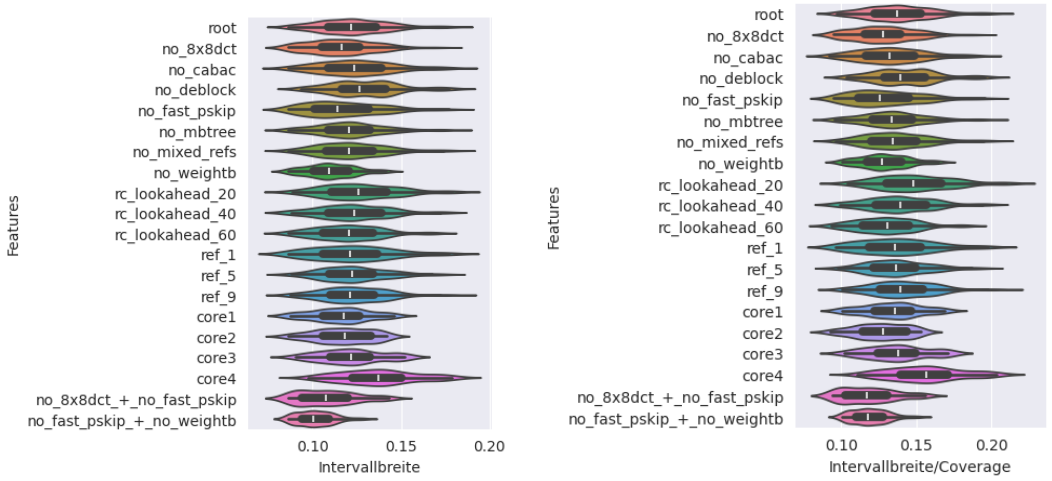
Um zusätzlich die Coverage zu berücksichtigen, teilen wir die konditionalen Intervallbreiten durch diese. Da eine kleinere Coverage ein zu kleines Intervall bedeutet, wird dieses durch das Teilen korrigiert. In Abbildung 4.13b wird diese angepasste Intervallbreite dargestellt. Hier ist zu sehen, dass nun neun teilweise andere Features einen deutlichen Unterschied aufweisen, wie in der zweiten Zeile von Tabelle 4.9 anhand der p-Werte ersichtlich ist. Bei der angepassten Intervallbreite haben die Features *no_deblock* und *core1* keinen signifikanten Unterschied mehr zum *root*-Feature, wohingegen die Features *rc_lookahead_40*, *rc_lookahead_60* und *core2* einen signifikanten Unterschied aufweisen. Von den nun zum *root*-Feature unterschiedlichen Featuren haben rc_lookahead_40 und core 4 eine größere mittlere Intervallbreite als *root*, alle anderen eine kleinere. Von den signifikant unterschiedlichen Features wurde *core4* korrekt gelernt, während alle anderen nicht korrekt gelernt wurden.

Tabelle 4.8: Konditionale Coverage, mittlere prozentuale Intervallbreite und Division von mittlerer prozentualer Intervallbreite und konditionaler Coverage der einzelnen Features des Toy-Systems.

	root	no_8x8dct	no_cabac	no_deblock	no_fast_pskip	no_mbtree	no_mixed_refs	no_weightb	rc_lookahead_20	rc_lookahead_40	rc_lookahead_60	ref_1	ref_5	ref_9	core1	core2	core3	core4	no_8x8dct_+_no_fast_pskip	no_fast_pskip_+_no_weightb
Cov	0.888	0.907	0.935	0.906	0.905	0.900	0.894	0.858	0.849	0.886	0.923	0.895	0.896	0.871	0.864	0.926	0.886	0.879	0.917	0.853
$\overline{\mathcal{CI}}$	0.123	0.116	0.125	0.129	0.117	0.121	0.122	0.110	0.127	0.123	0.120	0.123	0.123	0.123	0.116	0.117	0.121	0.137	0.108	0.101
$\frac{\overline{\mathcal{CI}}}{Cov}$	0.138	0.128	0.133	0.143	0.130	0.134	0.136	0.129	0.149	0.139	0.130	0.137	0.137	0.141	0.134	0.126	0.137	0.155	0.118	0.118

Tabelle 4.9: P-Werte Kruskal Wallis Signifikantstest zum Vergleich von den Verteilung der konditionalen Intervalle zwischen den einzelnen Features und dem *root* Feature. Bonferroni-korregierter p-Wert: 0.0026

	no_8x8dct	no_cabac	no_deblock	no_fast_pskip	no_mbtree	no_mixed_refs	no_weightb	rc_lookahead_20	rc_lookahead_40	rc_lookahead_60	ref_1	ref_5	ref_9	core1	core2	core3	core4	no_8x8dct_+_no_fast_pskip	no_fast_pskip_+_no_weightb
$\mathcal{CI}$	1.2e-4	0.3708	2.4e-4	5.7e-4	0.2916	0.5087	1.6e-14	0.0849	0.9896	0.1071	0.7244	0.9341	0.7869	0.0014	0.0103	0.7855	3.9e-9	9.5e-12	8.0e-25
$\frac{\mathcal{CI}}{Cov}$	4.1e-8	0.0071	0.0401	1.3e-6	0.0284	0.1824	4.7e-7	5.5e-5	0.8708	8.4e-5	0.3877	0.5176	0.1249	0.1545	3.7e-6	0.8634	8.6e-11	2.6e-17	1.1e-17



(a) Prozentuale konditionale Intervallbreite

(b) Prozentuale konditionale Intervallbreite geteilt durch die konditionale Coverage

Abbildung 4.13: Verteilung der prozentualen, konditionalen Intervallbreiten für ein Toy-System

4.2 Diskussion

4.2.1 FF1 - Verschiedene Samplegrößen

Erkenntnisse Die Ergebnisse aus Unterabschnitt 4.1.1 zeigen, dass bei den transduktiven Ansätzen von CONFORMAL-PREDICTION vor allem die absolute Größe der Trainingsdaten und weniger die relative Größe einen Einfluss auf die empirische Coverage hat. Das gewählte Softwaresystem scheint ebenfalls eine Rolle zu spielen. Aufgrund der vorliegenden Daten können wir jedoch nur vermuten, dass dies mit der Verteilung der Fehler zusammenhängt. Es lässt sich nicht eindeutig feststellen, ob die Beobachtungsfehler, die Variabilität oder die Einflüsse des Softwaresystems ausschlaggebend sind. Diese zusätzlichen Einflüsse des Softwaresystems machen es unmöglich, eine allgemeingültige Samplegröße festzulegen, die stets den gewählten Fehlerthreshold von α einhält.

Es zeigt sich, dass CONFORMAL-PREDICTION nicht für den Einsatz mit Samplegrößen nahe der Anzahl der Optionen geeignet ist. Stattdessen funktionieren Größen im Bereich von T-wise 2 und darüber hinaus am besten. Der gewählte Threshold ist dabei relativ groß gewählt. Je nach Anwendungsfall kann ein Konfidenzintervall von 80% statt 90% nicht erwünscht sein. Da sich der Fehler umgekehrt proportional zu der Samplegröße verhält, müssen in solchen Fällen mehr Daten verwendet werden.

Die beiden genutzten Methoden, CROSSVALIDATION+ und JACKKNIFE+, zeigten dabei kein unterschiedliches Verhalten in Bezug auf den Coverage-Fehler, obwohl JACKKNIFE+ als Spezialfall von CROSSVALIDATION+ deutlich mehr Modelle trainiert und aggregiert. Für die empirische Coverage macht dieser Mehraufwand unerwarteterweise keinen Unterschied.

Für die induktive Methode zeigt sich, dass die Größe des Trainingssets nur einen geringen Einfluss hat, während das Kalibrierungsset einen deutlich größeren Einfluss aufweist. Bei unseren Experimenten mit CONFORMALIZED-QUANTIL-REGRESSION traten einzelne Ausreißer in den Verteilungen auf, zum Beispiel beim Softwaresystem VP8. Dies zeigt, dass die Coverage im Durchschnitt zwar gut angenähert wird, aber bei unseren Konfigurationsgrößen keine absolute Garantie dafür besteht. Die Coverage konvergiert ähnlich zu der in anderen Arbeiten genannten Formel $\mathcal{O}(|C^{Calib}|^{-1/2})$ gegen Cov^{Ziel} [1]. Bei unseren Kalibrierungsset-Größen gibt die Formel die Werte 0.2, 0.12 und 0.08 an. Betrachten wir zum Beispiel das Apache-System, welches keine Ausreißer hat, so passen diese Werte zur Breite der Verteilung.

Vergleichen wir CONFORMAL-PREDICTION mit BAYESSCHE-REGRESSION, so zeigt CONFORMAL-PREDICTION ein durchgehend besseres und vor allem konsistentes Verhalten. Die Ursache für die schlechten Ergebnisse bei der Bayesian Regression muss noch genauer untersucht werden. Es zeigt sich jedoch,

dass, wie in Unterabschnitt 2.3.4 genannt, CONFORMAL-PREDICTION im Gegensatz zu den Bayesschen-Methoden kein Vorwissen und weniger Anpassung an die Daten, beziehungsweise das Softwaresystem benötigt.

Antwort der Forschungsfrage CONFORMAL-PREDICTION eignet sich für mittlere bis größere der üblichen Samplegrößen bei konfigurierbaren Softwaresystemen. Zufriedenstellende Ergebnisse wurden für die gewählten Systeme bei Samplegrößen in der Größenordnung von T-wise 2 und T-wise 3 erzielt. Allerdings ist CONFORMAL-PREDICTION für Samplegrößen in der Größenordnung von Option-wise nicht gut anwendbar. Eine größere absolute Anzahl der Samples ist dabei wichtiger als deren relative Größe zur Anzahl der möglichen Konfigurationen eines Systems.

4.2.2 FF2 - Verschiedene Sampling-Strategien

Erkenntnisse Die Ergebnisse zur zweiten Forschungsfrage, ob die Auswahl der Sampling-Strategie einen Einfluss auf CONFORMAL-PREDICTION hat, zeigen, dass die Sampling-Strategien Option-wise und T-wise für das Trainingsset der transduktiven Methoden nicht verwendet werden können. Diese Sampling-Strategien verletzen die Vorbedingungen von CONFORMAL-PREDICTION, da sie nicht die gleiche Verteilung der Performance-Werte wie das zufällig gezogene Validierungsset aufweisen, was bereits in anderen Arbeiten gezeigt wurde [13]. Eine Verletzung dieser Vorbedingung führt zu falschen Konfidenzintervallen und unvorhersehbarem Verhalten. Dies erklärt auch das in Unterabschnitt 4.1.2 beobachtete Phänomen, dass die Coverage in beide Richtungen schwankt, obwohl das Trainingsset von Option-wise zu T-wise 2 & 3 größer wird und immer eine Obermenge der kleineren ist. Die entsprechenden Verteilungen der Performance-Werte nähern sich dabei möglicherweise mal der Gesamtverteilung an und entfernen sich wieder. Daher sind diese Sampling-Strategien für transduktives CONFORMAL-PREDICTION nicht geeignet.

Die Sampling-Strategie Distance-based erfüllt die Vorbedingungen bei binären Optionen gut genug, sodass kein Unterschied zum Random-Sampling erkennbar ist. Allerdings gibt es keine Garantie dafür, dass Distance-based Sampling tatsächlich eine gleiche Verteilung besitzt [13]. Es wurde auch festgestellt, dass das verwendete Distance-based Sampling nicht für numerische Optionen geeignet ist[1]. Für weitere Experimente wäre es interessant zu untersuchen, welche numerischen Sampling-Strategien mit Distance-based kombiniert werden können, um CONFORMAL-PREDICTION anzuwenden.

[1] In der Dokumentation von SPLConqueror wird bereits darauf hingewiesen, dass diese Sampling-Strategie nur für binäre Optionen verwendet werden sollte, was dieses Ergebnis bestätigt.

Die für das numerische Sampling genutzte Plackett-Burman-Strategie wurde in dieser Arbeit nur zusammen mit Option-wise und T-wise verwendet und kann daher nicht auf ihre Verwendbarkeit bewertet werden. Diese Entscheidung basierte auf der Annahme, dass sowohl Distance-based als auch Random-Sampling für numerische Optionen geeignet wären.

Conformalized-Quantil-Regression ermöglicht die Verwendung jeder Sampling-Strategie für das Training der Modelle, sofern das zusätzliche Kalibrierungsset die Vorbedingungen von Conformal-Prediction erfüllt. Das Verhalten der Coverage hängt, wie bereits in der ersten Forschungsfrage erwähnt, fast ausschließlich vom Kalibrierungsset ab und nicht vom Trainingsset.

Der Nachteil dieser Methode liegt zum einen im zusätzlich benötigten Kalibrierungsset, was insbesondere bei begrenzten Daten im Bereich der konfigurierbaren Softwaresysteme die Frage aufwirft, ob die Daten nicht besser für das Training verwendet werden sollten. ZZum anderen werden die Intervallgrenzen der Quantile Regression über den gesamten Verlauf konstant verschoben, wodurch keine zusätzlichen Informationen über die Unsicherheiten einzelner Einflüsse im Vergleich zur reinen Quantile Regression gewonnen werden. Es könnten jedoch andere induktive Verfahren auf ihre Nützlichkeit überprüft werden. Eine alternative Art der Verschiebung, die lokale Unterschiede berücksichtigt, wäre ebenfalls denkbar.

Antwort der Forschungsfrage Die Wahl der Sampling-Strategie ist entscheidend für die Anwendung von Conformal-Prediction. Die Random-Sampling-Strategie funktioniert zuverlässig in allen Fällen, während die Distance-based-Strategie nur für das Sampling von binären Optionen geeignet ist. Andere Sampling-Strategien können lediglich bei induktiven Verfahren zum Trainieren des Modells verwendet werden und erfordern zusätzlich ein Kalibrierungsset, das beispielsweise durch Random-Sampling erzeugt wurde.

4.2.3 FF3 - Korrelation zum MAPE

Erkenntnisse Die Tests zur Untersuchung der Korrelation zwischen mittlerer prozentualer Intervallbreite und MAPE zeigen eine hohe Korrelation, wenn Sampling-Strategien mit korrekten Vorbedingungen, wie Random-Sampling, verwendet werden. Dies deutet auf einen Zusammenhang zwischen Coverage, mittlerer prozentualer Intervallbreite und MAPE hin. Diese Korrelation ist jedoch systemabhängig und konnte nicht für jedes System nachgewiesen werden, was darauf hinweist, dass zusätzliche Faktoren eine Rolle spielen könnten.

Bei Betrachtung von Coverage und mittlerer prozentualer Intervallbreite zeigt das lineare Modell eine signifikante Verbesserung gegenüber einer Vor-

hersage, die lediglich auf der mittleren prozentualen Intervallbreite basiert. Dies wirft die Frage auf, ob bei ausreichend großen Trainingssets die Coverage so stabil wird, dass sie nicht mehr benötigt wird. In einem solchen Fall wäre es möglich, den MAPE ohne Validierungsset abzuschätzen, da zur Bestimmung der Intervallbreiten lediglich die Vorhersage ohne den echten Wert aus dem Validierungsset erforderlich wäre. Die in konfigurierbaren Softwaresystemen verwendeten Größen reichen jedoch hierfür nicht aus.

Unsere Experimente zeigen auch, dass sich die Koeffizienten der Vorhersage je nach Modell und System ändern. Dies könnte bedeuten, dass die Intervallbreiten möglicherweise für das Hyperparameter-Tuning von Modellen verwendet werden könnten, was jedoch noch genauer untersucht werden müsste, Sie können allerdings nicht als Ersatzmetrik für MAPE genutzt werden, um Modelle zu vergleichen. Diese unterschiedlichen Koeffizienten könnten durch die eventuell anderen Verteilungen der Fehler entstehen. Diese unterschiedlichen Koeffizienten könnten durch verschiedene Verteilungen der Fehler verursacht werden. Ein hypothetisches Beispiel zeigt, dass ein einzelner extremer Ausreißer bei 30 Punkten im Validierungsset das 90%-Konfidenzintervall nicht beeinflusst, jedoch den MAPE stark verändern kann. Um dieses Problem zu lösen, könnten weitere Experimente durchgeführt werden, um zu untersuchen, wie sich die Intervallbreiten bei verschiedenen Ziel-Coverages verhalten und um eine vollständige Verteilung der Fehler zu approximieren. Es gibt bereits Verfahren zur Erstellung solcher Fehlerverteilungen mit CONFORMAL-PREDICTION, die genutzt werden könnten [29, 30].

Antwort der Forschungsfrage Es besteht eine Korrelation zwischen der Gütebewertung von CONFORMAL-PREDICTION, also der Intervallbreite, und dem MAPE. Diese Korrelation ist jedoch nicht perfekt monoton, selbst wenn die für unsere Sampling-Größen relevante Coverage berücksichtigt wird sind noch weitere Faktoren unbekannt. Für eine erste Abschätzung kann die Intervallbreite dennoch in den meisten Fällen herangezogen werden.

4.2.4 FF4 - Erkennen von Unsicherheiten

Erkenntnisse Für die vierte Forschungsfrage betrachten wir zunächst die konditionale Coverage. Es fällt auf, dass diese für alle Features nahe der angestrebten Coverage von 0,9 liegt. Das bedeutet, dass unabhängig von der gewählten Teilmenge des Validierungssets, in der ein bestimmtes Feature immer aktiv ist, die Intervalle stets 90% der Werte abdecken. Mit dem zusätzlichen Wissen darüber, welche Features korrekt gelernt wurden und welche nicht, lässt sich erkennen, dass sowohl gelernte als auch nicht gelernte Features teils kleinere, teils größere Coverage-Werte als der Durchschnitt aufweisen. Für einen

CONFORMAL-PREDICTION-Algorithmus, der eine gute konditionale Coverage liefern soll, ist dies zu erwarten. Dieses Ergebnis allein gibt jedoch keine Auskunft darüber, welche Features korrekt gelernt wurden und welche nicht.

Da die Schwankungen in der konditionalen Coverage gering waren, müssten bei nicht gelernten Features die Intervallbreiten größer sein als im Durchschnitt. Auffällig ist, dass die mittleren prozentualen Intervallbreiten ebenfalls nur geringe Schwankungen aufweisen, wodurch keine klare Unterscheidung zwischen gelernten und nicht gelernten Features möglich ist. Es wurde auch festgestellt, dass einige der nicht gelernten Features keinen signifikanten Unterschied in der Verteilung der prozentualen Intervallbreiten im Vergleich zum *root* Feature aufweisen, was bedeutet, dass sie nicht über die Intervallbreiten identifiziert werden können. Unter den Features mit unterschiedlichen Verteilungen befindet sich zum Beispiel *core4*, welches vermutlich aufgrund seines eigenen negativen Einflusses auf den Performance-Wert größere prozentuale Intervallbreiten als das *root*-Feature aufweist, obwohl es korrekt gelernt wurde. Die beiden Interaktionen wurden nicht korrekt gelernt, sondern ihr Einfluss wurde auf die entsprechenden Einzeleffekte verteilt. Sie haben, anders als erwartet, sehr kleine prozentuale Intervallbreiten, was ebenfalls auf ihren Einfluss auf den Performance-Wert zurückzuführen ist.

Um die unterschiedlichen konditionalen Coverages im Zusammenhang mit den Verteilungen der Intervallbreiten zu untersuchen, wurden diese durch die entsprechende Coverage geteilt. Dadurch zeigten sich signifikante Unterschiede bei anderen Features. Allerdings weisen auch hier nicht alle Verteilungen von falsch gelernten Features einen signifikanten Unterschied auf. Diese Metrik hat zudem die gleichen Schwächen wie bei dem *core4*-Feature und den Interaktionen.

Mit dem bisherigen Experiment und den betrachteten Werten ist es nicht möglich, eine klare Aussage darüber zu treffen, welche Features und Interaktionen gelernt wurden und welche nicht. Weitere Möglichkeiten für zukünftige Experimente könnten darin bestehen, den gelernten Einfluss auf die konditionalen Intervalle zusätzlich zu betrachten, also zu untersuchen, wie groß dieser Einfluss ist und ob er positiv oder negativ ist. Dazu müssten idealerweise viele Toy-Systeme systematisch untersucht werden, um eine allgemeinere Aussage treffen zu können. Zudem könnte man, um den möglicherweise komplexeren Zusammenhang zwischen gelerntem Einfluss, Intervallgröße und Coverage zu untersuchen, entsprechende Modelle trainieren.

Andere CONFORMAL-PREDICTION-Algorithmen könnten vielfältigere Intervalle erzeugen oder die Lokalität stärker in die Vorhersage einbeziehen.Ein alternativer Non-Conformity-Score könnte ebenfalls hilfreich sein, beispielsweise einer, der nicht nur den Abstand zwischen Vorhersage und realem Wert, sondern auch die absolute Größe berücksichtigt. Dies könnte ähnlich wie die

Nutzung des MAPE besser geeignet sein, um die Unsicherheit unabhängig vom Eigeneinfluss der Features zu bestimmen. Diese Änderungen am Experiment könnten möglicherweise zu einer eindeutigen Bestimmung der falsch gelernten Einflüsse führen und somit ein genaueres Bild der Unsicherheiten des Modells liefern.

Antwort der Forschungsfrage Die Analyse der Unsicherheit einzelner Features mit CONFORMAL-PREDICTION erweist sich als komplex, wie die initialen Untersuchungen zeigen. Es müssen weitaus mehr Faktoren berücksichtigt werden als ursprünglich angenommen, sodass weitere Untersuchungen erforderlich sind, um diese Forschungsfrage abschließend beantworten zu können.

4.3 Gefährdungen der Validität

Interne Validität Um die interne Validität unserer Studie zu gewährleisten, haben wir für jedes Experiment mindestens 30 Läufe mit zufälliger Initialisierung und umfassendem Hyperparameter-Tuning durchgeführt. Der Mean Absolute Percentage Error (MAPE) war in den meisten Fällen optimal und somit nahe beieinander. Diese Konsistenz in den Ergebnissen deutet darauf hin, dass unsere Modelle robust sind und nicht durch Zufall oder spezifische Parameterkonfigurationen beeinflusst werden, so dass die Ergebnisse des CONFORMAL-PREDICTION nicht durch Modelle mit schlecht gewählten Parametern beeinflusst wurden. Allerdings könnte dies für die Frage der Korrelation zwischen MAPE und Intervallbreite hinderlich sein, da wir nur kleine MAPE Werte betrachtet haben. Darüber hinaus haben wir bei der Anwendung der BAYESSCHE-REGRESSION durch die Verwendung von MCMC-Sampling den wahren Posterior und die prädiktiven Posterioren approximiert. Obwohl die Verwendung von 1000 Samples als recht gut angesehen wird, könnte eine Erhöhung der Anzahl der Samples zu noch genaueren Ergebnissen führen und die Unsicherheit weiter verringern. Die BAYESSCHE-REGRESSION hatte allerdings in manchen Fällen ein Problem mit der Auswahl der Parameter, was zu den ungewöhnlich großen Intervallen geführt haben könnte, hier sollte die Implementierung für ein finales Ergebnis noch einmal überprüft, beziehungsweise angepasst werden. Die Wahl der Performance-Funktion des Toy-Systems wurde mit Noise belegt um den realen Fall besser abzubilden, aber sie wurde möglichst einfach im Bezug auf die Einflüsse der Features gehalten, was nicht unbedingt den realem Fall bei konfigurierbaren Softwaresystemen entspricht.

Externe Validität Die externe Validität unserer Ergebnisse könnte durch die Auswahl der Systeme und Modelle beeinträchtigt sein. Unsere Studie um-

fasste acht Systeme und vier verschiedene Modelle aus mehreren Bereichen, wobei darauf geachtet wurde, dass immer zwei Systeme aus dem gleichen Anwendungsgebiet stammen, um die Generalisierbarkeit der Ergebnisse zu verbessern. Allerdings wurden nur drei CONFORMAL-PREDICTION Methoden getestet, wobei zwei aus dem für uns interessanten Fall (transduktive Methode) stammen und eine als Vergleichsmethode (induktive Methode) dient. Dies könnte die Übertragbarkeit der Ergebnisse auf andere Kontexte einschränken. Auch wurden nicht alle möglichen Sampling-Strategien oder ihre Kombinationen für binäre und numerische Features berücksichtigt. Wir haben aber aus beiden Bereichen des Samplings für konfigurierbare Softwaresysteme, dem Random-based und dem Coverage-based Sampling, Strategien getestet, um eine erste Aussage auf die Anwendbarkeit von CONFORMAL-PREDICTION treffen zu können. Um die externe Validität weiter zu stärken und die Generalisierbarkeit der Ergebnisse zu verbessern, empfehlen wir zukünftige Replikationen der Experimente mit einer breiteren Palette von CONFORMAL-PREDICTION Methoden und Sampling-Strategien.

Kapitel 5

Fazit

5.1 Zusammenfassung

Das Verständnis der Zusammenhänge zwischen den einzelnen Optionen und deren Auswirkungen auf die nicht-funktionalen Eigenschaften stellt bei konfigurierbaren Softwaresystemen eine erhebliche Herausforderung in der Softwareentwicklung dar. Insbesondere die Bestimmung von Unsicherheiten bei der Vorhersage von bislang nicht getesteten Konfigurationen erweist sich als problematisch. In dieser Arbeit wurde die Anwendung von Conformal-Prediction zur Vorhersage nicht-funktionaler Eigenschaften und zur Bestimmung der damit verbundenen Unsicherheiten in konfigurierbaren Softwaresystemen umfassend untersucht. Unsere Forschung zielte darauf ab, die Potenziale und Herausforderungen dieser Methodik zu identifizieren und zu bewerten, insbesondere im Hinblick auf die spezifischen Anforderungen und Gegebenheiten im Bereich konfigurierbarer Softwaresysteme.

Es wurde gezeigt, dass Conformal-Prediction mit gewissen Einschränkungen im Bereich von der konfigurierbaren Softwaresysteme anwendbar ist. Dabei spielt die Anzahl der Daten, die zum Trainieren der Modelle verwendet werden, eine wesentliche Rolle. Für die konsistente Vorhersage korrekter Konfidenzintervalle reicht ein Trainingsset in der Größenordnung der Anzahl der Optionen bei den getesteten Systemen nicht aus. Es wurde festgestellt, dass die absolute Anzahl der Daten entscheidend ist und nicht die relative Größe im Verhältnis zum System. Zudem wurden die Limitationen der verwendeten Sampling-Strategien und deren Einfluss auf die Intervall-Vorhersage aufgezeigt. Die Wahl der Sampling-Methoden hat einen signifikanten Effekt auf die Korrektheit und damit Relevanz der Ergebnisse, was für die praktische Anwendung von Conformal-Prediction in der Softwareentwicklung von besonderer Bedeutung ist. Dies unterstreicht die Notwendigkeit, geeignete Datenmengen und -strategien zu wählen, um valide Vorhersagen zu gewährleisten.

Es wurde festgestellt, dass Random-Sampling für alle Conformal-Prediction-Algorithmen geeignet ist, während Distance-based-Sampling als Ersatz für das Sampling binärer Optionen verwendet werden kann. Option-wise und T-wise Sampling-Methoden sind hingegen nur für Trainingssets von induktiven Conformal-Prediction-Algorithmen anwendbar, bei denen die Sampling-Strategie keine spezifischen Voraussetzungen erfüllen muss.

Es wurde auch festgestellt, dass Conformal-Prediction konsistentere Ergebnisse liefert als bisher verwendete Intervall-vergebende Verfahren.

Der Ersatz der Metrik MAPE durch die Konfidenzintervalle von Conformal-Prediction zur Einsparung des Validierungssets hat sich bei konfigurierbaren Softwaresystemen als nicht möglich erwiesen. Es wurde jedoch ein Zusammenhang zwischen den Metriken von Conformal-Prediction und MAPE festgestellt, der allerdings noch von weiteren unbekannten Faktoren beeinflusst wird. Zudem wurde ein erster Einblick in die Bestimmung der Unsicherheiten in der Vorhersage der nicht-funktionalen Eigenschaften mithilfe von Conformal-Prediction gegeben. Ein besonderer Fokus lag dabei auf den Unsicherheiten, die durch einzelne Konfigurationsoptionen und deren Interaktionen in die Vorhersage eingebracht werden. Dies hat verdeutlicht, dass dieses Gebiet zu komplex ist, um allein durch eine direkte Interpretation der Intervallbreiten von Conformal-Prediction gelöst zu werden.

Diese Erkenntnisse können als Grundlage für weitere Forschungen in diesem Bereich dienen, um die bisherigen Vorhersagen im Bereich der konfigurierbaren Softwaresysteme besser zu analysieren und zu bewerten. Zusammenfassend lässt sich sagen, dass die vorliegende Arbeit einen wertvollen Beitrag zur Forschung im Bereich der konfigurierbaren Softwaresysteme und der Anwendung von Conformal-Prediction leistet. Die Ergebnisse bieten nicht nur theoretische Einsichten, sondern auch praktische Implikationen für die Softwareentwicklung, insbesondere in Bezug auf die Vorhersage von nicht-funktionalen Anforderungen.

5.2 Ausblick

Zukünftige Forschungen könnten darauf aufbauen, die Anwendbarkeit von Conformal-Prediction insbesondere mit anderen Sampling-Strategien und Conformal-Prediction-Algorithmen weiter zu untersuchen, um eine klare Aussage geben zu können, für welche Sampling-Strategien welche Conformal-Prediction-Algorithmen zu nutzen sind. Zudem könnte man versuchen, die unbekannten Faktoren für die Korrelation zur MAPE zu bestimmen, sodass im Idealfall der MAPE ohne Validierungset eingeschätzt werden kann, was ein großer Vorteil hinsichtlich der Nutzung der verfügbaren Daten darstellen kann.

Auch könnte die Methodik zur Bestimmung der Unsicherheit weiter verfeinern werden, um ihre Anwendbarkeit im praktischen Kontext der konfigurierbaren Softwaresysteme gewährleisten zu können und neue Einblicke hinsichtlich der Vorhersage nicht-funktionaler Eigenschaften zu liefern.

Literaturverzeichnis

[1] Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification, 2021. URL `https://arxiv.org/abs/2107.07511`.

[2] Sven Apel, Don Batory, Christian Kästner, and Gunter Saake. *Feature-Oriented Software Product Lines: Concepts and Implementation*. Springer Berlin Heidelberg, 2013. ISBN 9783642375217. doi: 10.1007/978-3-642-37521-7.

[3] Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1), February 2021. ISSN 0090-5364. doi: 10.1214/20-aos1965.

[4] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, 2012. URL `https://api.semanticscholar.org/CorpusID:15700257`.

[5] Supratik Chakraborty, Daniel J. Fremont, Kuldeep S. Meel, Sanjit A. Seshia, and Moshe Y. Vardi. Distribution-aware sampling and weighted model counting for sat. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, page 1722–1730. AAAI Press, 2014.

[6] Arnaud De Myttenaere, Boris Golden, Bénédicte Le Grand, and Fabrice Rossi. Mean absolute percentage error for regression models. *Neurocomputing*, 2016. doi: 10.48550/ARXIV.1605.02541.

[7] Nicolas Dewolf, Bernard De Baets, and Willem Waegeman. Valid prediction intervals for regression problems. *Artificial Intelligence Review*, 2021. doi: 10.48550/ARXIV.2107.00363.

[8] Johannes Dorn, Sven Apel, and Norbert Siegmund. Mastering uncertainty in performance estimations of configurable software systems. *Empirical Software Engineering*, 28(2), January 2023. ISSN 1573-7616. doi: 10.1007/s10664-022-10250-2.

[9] Wei Fu and Tim Menzies. Easy over hard: a case study on deep learning. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE'17. ACM, August 2017. doi: 10.1145/3106237.3106256.

[10] Alexander Grebhahn, Norbert Siegmund, and Sven Apel. Predicting performance of software configurations: There is no silver bullet. *CoRR*, abs/1911.12643, 2019. URL `http://arxiv.org/abs/1911.12643`.

[11] Xue Han and Tingting Yu. An empirical study on performance bugs for highly configurable software systems. In *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '16, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450344272. doi: 10.1145/2961111.2962602. URL `https://doi.org/10.1145/2961111.2962602`.

[12] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. ISSN 0162-8828. doi: 10.1109/34.709601.

[13] Christian Kaltenecker, Alexander Grebhahn, Norbert Siegmund, Jianmei Guo, and Sven Apel. Distance-based sampling of software configuration spaces. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, May 2019. doi: 10.1109/icse.2019.00112.

[14] Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63(3):425–464, September 2001. ISSN 1467-9868. doi: 10.1111/1467-9868.00294.

[15] Henrik Linusson. *Nonconformity Measures and Ensemble Strategies : An Analysis of Conformal Predictor Efficiency and Validity*. PhD thesis, Stockholm University, Department of Computer and Systems Sciences, 2021.

[16] Valeriy Manokhin. *Practical Guide to Applied Conformal Prediction*. Packt Publishing, Limited, 2024. ISBN 9781805122760.

[17] Valery Manokhin. *Machine Learning for Probabilistic Prediction*. PhD thesis, Royal Holloway, University of London, 2022.

[18] Tom M. Mitchell. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997. ISBN 0070428077.

[19] Andrew Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Twenty-first international conference on Machine learning - ICML '04*, ICML '04. ACM Press, 2004. doi: 10.1145/1015330.1015435.

[20] Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. *Inductive Confidence Machines for Regression*, pages 345–356. Springer Berlin Heidelberg, 2002. ISBN 9783540367550. doi: 10.1007/3-540-36755-1_29.

[21] Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/5103c3584b063c431bd1268e9b5e76fb-Paper.pdf`.

[22] Bernhard Schölkopf, Alexander J. Smola, and Francis Bach. *Learning with Kernels - Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2018. ISBN 9780262536578.

[23] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(12):371–421, 2008. URL `http://jmlr.org/papers/v9/shafer08a.html`.

[24] Norbert Siegmund, Alexander Grebhahn, Sven Apel, and Christian Kästner. Performance-influence models for highly configurable systems. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE'15. ACM, August 2015. doi: 10.1145/2786805.2786845.

[25] Norbert Siegmund, Nicolai Ruckel, and Janet Siegmund. Dimensions of software configuration: on the configuration context in modern software development. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, nov 2020. doi: 10.1145/3368089.3409675.

[26] V. Vovk. On-line confidence machines are well-calibrated. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, SFCS-02. IEEE Comput. Soc, 2002. doi: 10.1109/sfcs.2002.1181895.

[27] Vladimir Vovk. Cross-conformal predictors. *Annals of Mathematics and Artificial Intelligence*, 74(1–2):9–28, July 2013. ISSN 1573-7470. doi: 10.1007/s10472-013-9368-4.

[28] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, 2005. ISBN 0387001522. doi: 10.1007/b106715.

[29] Vladimir Vovk, Ilia Nouretdinov, Valery Manokhin, and Alexander Gammerman. Cross-conformal predictive distributions. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, Evgueni Smirnov, and Ralf Peeters, editors, *Proceedings of the Seventh Workshop on Conformal and Probabilistic Prediction and Applications*, volume 91 of *Proceedings of Machine Learning Research*, pages 37–51. PMLR, 11–13 Jun 2018.

[30] Vladimir Vovk, Jieli Shen, Valery Manokhin, and Min-ge Xie. Nonparametric predictive distributions based on conformal prediction. *Machine Learning*, 108(3):445–474, August 2018. ISSN 1573-0565. doi: 10.1007/s10994-018-5755-8.

[31] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer International Publishing, 2022. ISBN 9783031066498. doi: 10.1007/978-3-031-06649-8.