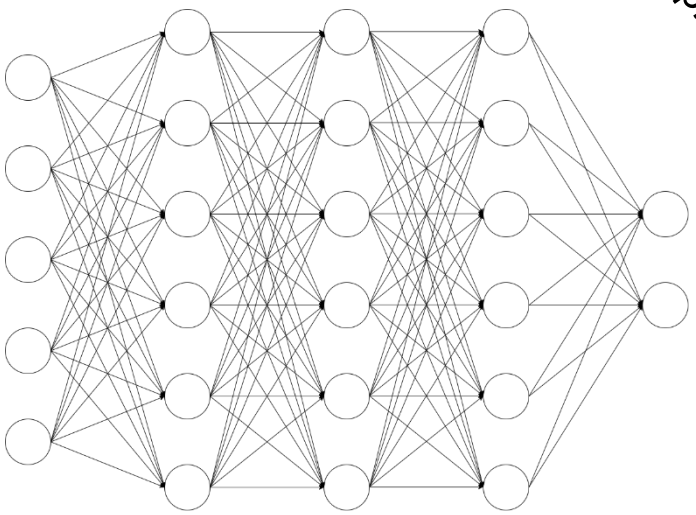
Bachelorarbeit BAA_DE – FS23

Dmitriy An

Aufbereitung und Analyse historischer Daten des Herstellprozesses von Metalloxid-Varistoren



Hochschule Luzern – Technik & Architektur

Kriens, 5. Juni 2023

Bachelorarbeit BAA_DE – FS23

Dmitriy An
Studiengang Digital Engineering
Vertiefung Digital Development

Aufbereitung und Analyse historischer Daten des Herstellprozesses von Metalloxid-Varistoren

Betreuender Dozent

Prof. Dr. Andrew Paice

Hochschule Luzern – Technik & Architektur

Kriens, 5. Juni 2023

I. Abstrakt

Künstliche Intelligenz und Machine Learning erfreuen sich steigender Beliebtheit für die Erkennung von Regeln und Mustern Daten, die bisher unbekannt waren. In Zusammenarbeit mit dem Industriepartner, Hitachi Energy AG, wurde das Potenzial von Machine Learning zur Vorhersage der Durchbruchsspannung von Metalloxid-Varistoren untersucht. Die bereitgestellten Daten beinhalten Informationen zu den Rohmaterialien, Prozessdaten aus der Fabrik und Testdaten. Aufgrund unvollständiger Rohmaterialdaten wurde der Datensatz jedoch ohne diese erstellt. Da die Prozessdaten nicht auf einzelne Varistoren rückverfolgbar waren, wurde der Datensatz auf Batches reduziert und mit Gaussverteilungen bzw. einem Durchschnitt und einer Standardabweichung versehen. Anschließend wurde er in ein Vector Space Format transformiert, um Machine Learning Modelle anwenden zu können. Zur Vorhersage der Durchbruchsspannung wurden verschiedene Modelle wie Deep Learning, Random Forest und Gradient Boosting verwendet. Bei den Ansätzen Random Forest und Gradient Boosting wurde eine durchschnittliche relative absolute Abweichung von knapp 1% vom Mittelwert der Gaussverteilungen erreicht, während das beste Deep Learning Modell eine Abweichung von 1,9% hatte. Die Standardabweichung konnte jedoch mit keinem Modell zufriedenstellend vorhergesagt werden. Es wird vermutet, dass ein grösserer Hyperparameterraum für Deep Learning Modelle bessere Ergebnisse liefern könnte. Die Nutzung von Random Forest und Gradient Boosting zeigt, dass die Vorhersage der Durchbruchsspannung möglich ist und eine Abweichung von bis zu 1% erreicht werden kann.

Schlüsselwörter: Künstliche Intelligenz, Machine Learning, Datenaufbereitung, Deep Learning, Random Forest, Gradient Boosting

II. Inhaltsverzeichnis

I. Abstrakt i
II. Inhaltsverzeichnis ii
III. Abbildungsverzeichnis iv
1 Einleitung 1
2 Grundlagen 3
2.1 Artificial Intelligence 3
2.2 Machine Learning 3
2.2.1 Herausforderungen von Machine Learning 4
2.3 Artificial Neural Network 4
2.4 Decision Trees 5
2.4.1 Random Forest 6
2.4.2 Gradient Boosting 6
2.5 Herstellung der Metalloxid-Varistoren 6
3 Datenaufbereitung 8
3.1 Rohmaterialdaten 8
3.1.1 Analysezertifikate 8
3.1.2 Neues SAP 9
3.1.3 JiVS 10
3.2 Prozessdaten 11
3.2.1 Factory Layer 11
3.2.2 Legacy DB 15
3.2.3 Langzeit-Stabilitätsmessungen 15
3.3 Testdaten (Manufacturing Execution System) 16
3.3.1 MOPA-Viewer 16
3.3.2 MES-Support 16
3.3.3 Legende 18
3.3.4 Datensatz 19
3.4 Zusammenfassung Datenaufbereitung 20
3.5 Verbesserungsmöglichkeiten 21
4 Datenanalyse 23
4.1 Deep Learning Modell mit nur den MES-Daten 23
4.2 Deep Learning Modell mit Factory Layer Daten 25
4.3 Deep Learning Modell mit komplexerer Architektur 26
4.4 Decision Tree Algorithmen 27

4.4.1 Random Forest 27
4.4.2 Gradient Boosting 27
5 Resultate 29
5.1 Datenaufbereitung 29
5.2 Datenanalyse 29
6 Schlussdiskussion 31
7 Literaturverzeichnis 33
8 Anhangsverzeichnis 34
8.1 Verfügbarkeit der Daten 35
8.2 Datenfluss 36
8.3 Legende MES-Daten (MOPA-Viewer) 37
8.4 Ablauf Deep Learning Funktion 40
8.5 Python-Skripte 41
8.6 Übersicht Python-Skripte 45
8.7 Projektauftrag 46
8.7.1 Ausgangslage 46
8.7.2 Qualitative Ziele 46
8.7.3 Quantitative Ziele 46
8.7.4 Anforderungskatalog 47
8.7.5 Projektplan 48
8.7.6 Risikomanagement 49

III. Abbildungsverzeichnis

Abbildung 1: Aufbau von einem Deep Learning Neural Network (IBM, What are Neural Networks? | IBM, n.d.) 4
Abbildung 2: Darstellung von einem Decision Tree mit dem Wurzelknoten (blau), internen Knoten (grün) und Blattknoten (grau) (IBM, What is a Decision Tree | IBM, kein Datum) 5
Abbildung 3: Ein Beispiel von einem Decision Tree in dem bestimmt wird, ob es gute Bedingungen zum Surfen gibt (IBM, What is a Decision Tree | IBM, kein Datum) 6
Abbildung 4: Übersicht der Python-Skripte und wie diese zusammenhängen 8
Abbildung 5: Angabe in Prozent anstelle Teile pro Million von Antimonoxid Sb2O3 Analysezertifikat 9
Abbildung 6: Analysezertifikat von Alu-Draht mit grossem Leerraum zwischen den Spalten 9
Abbildung 7: Ausschnitt JiVS gefiltert nach Zinkoxid ZnO 10
Abbildung 8: Verschiedene Werte für gleiche Bezeichnungen von Glaspulver. Gleiche Bezeichnungen sind mit gleichen Farben markiert 11
Abbildung 9: Zonen vom Sinterofen mit den Laufenden Nummern (blaue Quadrate) die 15 bis 130 Metalloxid-Varistoren beinhalten. Die Zahlen in blau sind die Ist-Temperaturen und die in grün sind die Soll-Temperaturen. 12
Abbildung 10: Einträge in der Factory Layer Datenbank aus dem Jahr 2018 die Fehlerhaft sind 12
Abbildung 11: Plan mit Abmessungen vom Sinterofen (alternative Bilder wurden nicht zur Verfügung gestellt) 13
Abbildung 12: Ausschnitt aus der Factory Layer Datenbank Tabelle «FLC_LogEntryReal». Die ersten acht Einträge haben die gleiche ID und dort wo auch eine gleiche DatapointID ist, gibt es eine neue Revisionsnummer. In der rechten Spalte befinden sich die Werte. 14
Abbildung 13: Ausschnitt aus der Legacy-Datenbank Tabelle «SO_AufbauDaten» mit den minimalen, maximalen und durchschnittlichen Temperaturen der einzelnen Zonen 15
Abbildung 14: Ausschnitt aus CSadmin mit Langzeit-Stabilitätsmessungen 16
Abbildung 15: Strings die Werte der letzten 10 Spalten beinhalten aufgrund von fehlenden Kommas 17
Abbildung 16: Ausschnitt Bill of Materials (BOM) welches nach Zinkoxid ZnO gefiltert wurde. Diese werden benötigt, um den Zusammenhang mit den Daten der Rohmaterialien zu machen. 17
Abbildung 17: Ausschnitt Production Orders die benötigt werden um den Zusammenhang mit dem MOPA-Viewer zu erstellen 18
Abbildung 18: Ausschnitt der Legende vom MOPA-Viewer aus dem MES 18
Abbildung 19: Statistische Informationen der Spalte «Up_Rest_Ist» aus dem MOPA-Viewer vom Profile Report die mit der Python-Bibliothek «ydata_profiling» erstellt wurde 19
Abbildung 20: Ausschnitt MOPA-Viewer gefiltert nach geschnittenen Varistoren (gekennzeichnet mit XX in Var_Typ) 19
Abbildung 21: Übersicht der Verfügbarkeit der Daten der einzelnen Quellen 20
Abbildung 22: Datenfluss in der Firma von den Daten die für die Qualität der MOVs relevant sein könnten. Die Schnittstelle zwischen Bill of Materials und Production Order ist nicht zur Verfügung gestellt worden 21
Abbildung 23: ReLU (rectified linear unit) als Aktivierungsfunktion 24
Abbildung 24: Resultate vom Deep Learning Modell mit individuellen Varistoren. Die Abweichung vom eigentlichen Wert beträgt im Durchschnitt 13.6% 24
Abbildung 25: Resultate vom Deep Learning Modell mit Batches als Datenpunkten. Die Abweichung vom eigentlichen Wert beträgt im Durchschnitt 6.1% für den Durchschnitt der Gaussverteilung 25

Abbildung 26: 10 beste Resultate aller 1440 Kombinationen. Die Hyperparameter und Resultate werden von jedem Modell einzeln gespeichert. 26
Abbildung 27: Resultate vom Random Forest mit einer durchschnittlichen relativen Abweichung von 1.11% vom Durchschnitt der Gausskurve und 17.7% von der Standardabweichung 27
Abbildung 28: Feature Importance bzw. Gewichte der einzelnen Variablen mit einem Gewicht von über 0.1%. Alle Gewichte zusammen ergeben 1 und je höher der Wert, desto grösseres Gewicht hat die Variabel. 27
Abbildung 29: Resultate vom Gradient Boosting. Die durchschnittliche relative Abweichung beträgt 1.25% 28
Abbildung 30: Resultat von der Gradient Boosting Bibliothek XGBoost mit einer durchschnittlichen relativen Abweichung von 1.25% 28

1 Einleitung

In den letzten Jahren hat die Anwendung von künstlicher Intelligenz und insbesondere Machine Learning in der Industrie stark zugenommen. Mithilfe von Machine Learning Methoden können ausreichend grosse Datenmengen analysiert werden, um Muster und Regeln zu erkennen, die zuvor nicht bekannt waren. Dies ermöglicht es, Prozesse zu optimieren und bessere Produkte zu entwickeln, was schlussendlich zu einem höheren Gewinn führen kann. Im Rahmen dieser Bachelorarbeit wurde im Auftrag des Industriepartners Hitachi Energy AG untersucht, wie mithilfe von Machine Learning die Qualitätskontrolle von Metalloxid-Varistoren verbessert werden kann. Hitachi Energy stellt Metalloxid-Varistoren her, die als Überspannungsableiter dienen. Diese Varistoren durchlaufen einen langen Produktionsprozess, bei dem Daten gesammelt werden. Am Ende des Prozesses werden alle Varistoren individuell getestet, und die Testergebnisse werden gespeichert. Die Durchbruchspannung der Varistoren kann variieren, weshalb Klassen eingeführt wurden, um sie zu gruppieren. Es gibt Haupt- und Nebenklassen, wobei die Varistoren in der Hauptklasse die erforderlichen Eigenschaften aufweisen, während die in den Nebenklassen dies nicht tun. Die Varistoren, die weder in die Haupt- oder Nebenklassen eingeteilt werden können, zählen zum Ausschuss. Die Varistoren dienen als Blöcke, die seriell zusammengesetzt werden können, um die Durchbruchspannung zu erhöhen. Wenn ein Varistor einer unteren Nebenklasse zugeteilt wird, kann dieser mit einem aus einer oberen Nebenklasse zusammengesetzt werden, um quasi zwei Varistoren aus der Hauptklasse zu simulieren. Hitachi Energy möchte schlussendlich den Ausschuss reduzieren und einen Überschuss an Nebenklassen vermeiden. In einem vorherigen Projekt wurden Literaturrecherchen gemacht über Machine Learning und Metalloxid-Varistoren und eine erste Übersicht der Daten wurde erschaffen und in diesem Projekt soll gezeigt werden, ob Machine Learning überhaupt angewendet werden kann und ob zusätzliche Daten benötigt werden.

Das Ziel dieser Arbeit ist es, die Qualitätskontrolle der Metalloxid-Varistoren mithilfe von Machine Learning zu verbessern. Hierfür soll zunächst ein Datensatz erstellt werden, der für das Machine Learning Modell verwendet werden kann. Anhand dieses Datensatzes soll ein Modell erstellt werden, das die Klasse vorhersagen kann. Zudem sollen Verbesserungsvorschläge gemacht werden, um den Prozess zukünftig zu optimieren. Die übergeordneten Ziele des Projekts sind, möglichst viele Varistoren in der Hauptklasse herzustellen, gezielt untere Nebenklassen herzustellen, falls ein Überschuss von oberen vorhanden ist und falls in einem frühen Prozessschritt ein Fehler auftritt, die nächsten Prozessschritte anzupassen, um den Fehler zu kompensieren.

Im Projektstart wurde ein Fabrikbesuch sowie eine Intensivwoche durchgeführt, um ein besseres Verständnis für die Situation zu erlangen. Anschliessend wurde ein Projektauftrag erstellt, in dem die Ausgangslage, Ziele, Anforderungen und das Risikomanagement beschrieben wurden. Ausserdem wurde ein grober Projektplan in Form eines Gantt-Charts erstellt.

Das erste Ziel des Projekts bestand darin, einen Datensatz vorzubereiten, an dem Machine Learning angewendet werden kann. Eine Hürde war, dass die Daten aus verschiedenen Quellen stammen. Einige Daten befanden sich in einem Archivsystem, für das zunächst Zugriffsberechtigungen eingeholt werden mussten, andere wurden von einer externen Firma verwaltet und Backups der Datenbanken mussten angefragt werden. Die Testdaten, die lokal exportiert wurden, enthielten zudem noch viele Fehler.

Schliesslich wurden mit Python alle vorhandenen und vollständigen Daten kombiniert und zu einem Datensatz zusammengeführt. Die Rohmaterialdaten wurden dabei nicht verwendet, da

sie nicht komplett waren. Zur Verbesserung der Handhabung der Daten und zur Steigerung der Modell-Performanz wurden Empfehlungen gegeben, die umgesetzt werden könnten.

Mit dem vorbereiteten Datensatz wurden mehrere Machine Learning Modelle erstellt. Die ersten Ergebnisse wurden mit Deep Learning erzielt, wobei auch andere Algorithmen, die auf Decision Trees basieren, ausprobiert wurden. Es stellte sich jedoch heraus, dass diese die besten Ergebnisse lieferten.

Im Rahmen dieser Dokumentation wurden zunächst die Grundlagen, die für diese Dokumentation notwendig sind, erläutert. Im dritten Kapitel wurde beschrieben, wie die Datenbeschaffung und Datentransformation durchgeführt wurde, um einen für das Machine Learning geeigneten Datensatz zu erstellen. Im darauffolgenden Kapitel wurde erläutert, wie das Machine Learning Modell erstellt wurde und welche Genauigkeiten dabei erzielt wurden. Im fünften Kapitel werden die Resultate der Datenaufbereitung und Datenanalyse dargestellt. Abschliessend werden in einer Schlussdiskussion die wichtigsten Erkenntnisse zusammengefasst und ein Ausblick auf mögliche zukünftige Entwicklungen gegeben.

2 Grundlagen

In diesem Kapitel werden die Definitionen und Unterschiede von Artificial Intelligence, Machine Learning, Neural Networks bzw. Deep Learning und Decision Trees erklärt und mit Beispielen ergänzt. Zusätzlich wird der Herstellprozess der Metalloxid-Varistoren kurz beschrieben. In der Literatur wurde im Zusammenhang von Metalloxid-Varistoren und Machine Learning nichts gefunden und aus diesem Grund wurde kein spezifischer Ansatz von einem ähnlichen Projekt verwendet.

2.1 Artificial Intelligence

Artificial Intelligence beschriebt die Simulation von menschlicher Intelligenz von Maschinen und Computersystemen. Es beinhaltet unter andrem Expertensysteme, Natürliche Sprachverarbeitung, Computer Vision und Machine Learning welches ein Kernthema in dieser Arbeit sein wird. Anwendungen der künstlichen Intelligenz wären Klassifizierungen und Chatbots.

Die Programmierung von Artificial Intelligence fokussiert sich auf das Lernen, die Denkfähigkeit und die Selbst-Korrektur. Beim Lernen werden anhand von Daten, Regeln bzw. Algorithmen erstellt, die beschrieben, wie eine Aufgabe gelöst werden kann. Die Denkfähigkeit beschreibt die Auswahl vom richtigen Algorithmus für das erwünschte Ziel und die Selbst-Korrektur beschriebt, dass die Algorithmen der Artificial Intelligence sich andauernd anpassen, um noch genauere Resultate zu liefern.

Artificial Intelligence wird in vielen erfolgreichen Unternehmen verwendet, weil es mithilfe von Machine Learning grosse Mengen an Daten schnell brauchbare Informationen herausgibt. Das Hauptproblem ist jedoch die Aneignung und Kosten für die Prozessierung dieser Mengen an Daten. (Burns, Lawskowski, & Tucci, 2023)

2.2 Machine Learning

Machine Learning ist eine Untergruppe von Artificial Intelligence mit dem Fokus auf den Gebrauch von Daten und Algorithmen, um den menschlichen Lernprozess zu imitieren und somit die Genauigkeit zu erhöhen.

Mit statistischen Methoden werden Algorithmen trainiert, um Klassen oder numerische Werte vorherzusagen und wichtige Erkenntnisse zu entdecken. Beim Klassifizieren wird versucht anhand den Datenpunkten die dazugehörige Klasse zu bestimmen und in diesem Beispiel wären es die Haupt- und Nebenklassen oder der Ausschuss. Beim Vorhersehen von numerischen Werten bzw. einer Regressionsanalyse, wird eine Zahl bestimmt und würde z. B. die Durchbruchspannung der Varistoren sein. Das Vorhersehen von numerischen Werten wird in diesem Projekt genauer untersucht. Anhand dieser Kenntnisse sollen Entscheidungen getroffen und wichtige Metriken beeinflusst werden.

Es gibt vier Methoden von Machine Learning:

- Supervised Learning
  Beschriftete Datensätze bzw. Datensätze dessen Resultat bekannt ist, werden für Algorithmen von Klassifizierungen und Vorhersagen benutzt. (Spam erkennen)
- Unsupervised Learning
  Unbeschriftete Datensätze werden durch Algorithmen analysiert und gruppiert ohne menschlichen Eingriff. (Kundengruppierung, Bilderkennung)
- Semi-Supervised Learning
  Beschriftete Datensätze werden verwendet, um unbeschriftete Datensätze zu beschriften und danach Supervised Learning anzuwenden.
- Reinforcement Machine Learning

Ähnlich wie Supervised Learning, aber anstatt des Modells mit einem ganzen Datensatz zu trainieren, lernt es ständig durch Ausprobieren. Eine Sequenz von erfolgreichen Resultaten wird für Empfehlungen verwendet.

(IBM, What is Machine Learning? | IBM, n.d.)

2.2.1 Herausforderungen von Machine Learning

Die grösste Herausforderung von Machine Learning und auch die, wo am meisten Zeit in Anspruch nimmt, ist die Datenbeschaffung und die Datenqualität. Wenn die Daten, die für ein Machine Learning Modell verwendet werden, schlecht sind bzw. ungenau oder fehlerhaft sind, wird auch das Modell selbst schlecht. Vor allem weil meistens viele Daten benötigt werden, um eine akzeptable Performanz zu erzielen, ist es umso schwieriger die Daten zu beschaffen.

Ein anderes Problem wäre Overfitting was passieren kann, wenn das Modell sich zu fest an den Regeln der Daten hält, welche verwendet wurden, um es zu trainieren. Dies führt dazu, dass das Modell ungenaue Resultate liefert, wenn neue Daten eingespeist werden.

Es gibt noch einige weitere Probleme wie Interpretierbarkeit, Skalierbarkeit und Datenschutz die aber im Rahmen von diesem Projekt nicht weiter angesprochen werden.

2.3 Artificial Neural Network

Ein Neural Network (Abbildung 1) ist eine Untergruppe von Machine Learning, welches durch das menschliche Gehirn inspiriert wurde. Es besteht aus einer Schicht, welche die Inputs aufnimmt (Input Layer), einer Schicht, welche den Output herausgibt (Output Layer) und mindestens einer Schicht dazwischen (Hidden Layers). Jede Schicht besteht aus mehreren künstlichen Neuronen und jedes davon ist mit jedem Neuron in den benachbarten Schichten verbunden. Jedes künstliche Neuron hat eine Gewichtung und einen Schwellwert. Wenn dieser Schwellwert überschritten wird, werden Daten an die nächste Schicht weitergeleitet.

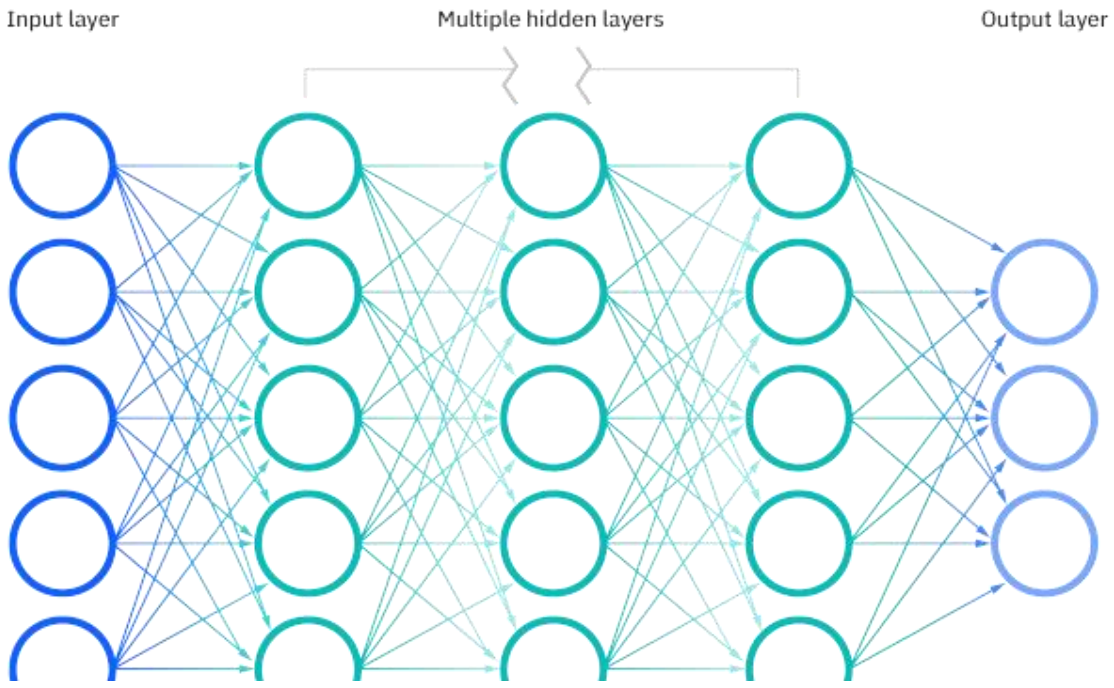


Abbildung 1: Aufbau von einem Deep Learning Neural Network (IBM, What are Neural Networks? | IBM, n.d.)

Neural Networks hängen von den Trainingsdaten ab und verbessern ihre Genauigkeit mit der Zeit. Wenn ein Neural Network eine genügende Genauigkeit aufweist, können Daten mit hoher Geschwindigkeit klassifiziert und gruppiert werden.

Neural Networks können grundsätzlich in drei verschiedene Typen gruppiert werden:

- Feedforward Neural Networks (Multi-Layer Perceptrons)
  Sie bestehen aus einem Input Layer, mindestens einem Hidden Layer und einem Output Layer. Es ist zu beachten, dass sie oft aus sigmoidalen Neuronen bestehen, weil die meisten Probleme nicht-linear sind. Eine Anwendung wäre zum Beispiel Klassifizierung oder Regression. Dieser Typ von Neural Network wird in diesem Projekt verwendet, aber mit ReLU als Aktivierungsfunktion, da sie effizienter ist.
- Convolutional Neural Networks
  Diese Neural Networks benutzen Prinzipien aus der Linearen Algebra wie Matrixmultiplikation, um Muster in Bilder zu erkennen.
- Recurrent Neural Networks
  Sie werden für Time-Series Daten (Zeitabhängige Daten) verwendet und haben einen Feedback Loop. Somit können Vorhersagen von beispielsweise den Kursen vom Aktienmarkt gemacht werden.

Ein Neural Network welches aus mehr als drei Schichten besteht, wird Deep Learning Algorithmus genannt.

2.4 Decision Trees

Ein Decision Tree (Abbildung 2) ist ein nichtparametrischer Lernalgorithmus, der für Klassifizierungs- und Regressionsaufgaben verwendet werden kann. Es hat eine hierarchische Baumstruktur und besteht aus einem Wurzelknoten, Zweigen, internen Knoten und Blattknoten.

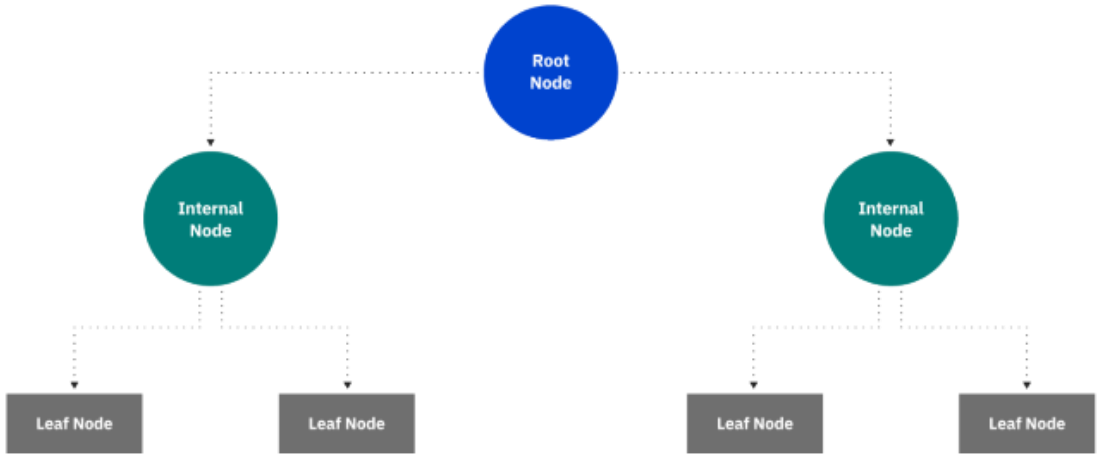


Abbildung 2: Darstellung von einem Decision Tree mit dem Wurzelknoten (blau), internen Knoten (grün) und Blattknoten (grau) (IBM, What is a Decision Tree | IBM, kein Datum)

Der Algorithmus von einem Decision Tree fängt bei dem Wurzelknoten an und je nach Eigenschaften vom Datenpunkt, wird zum nächsten Knoten fortgeschritten (Abbildung 3).

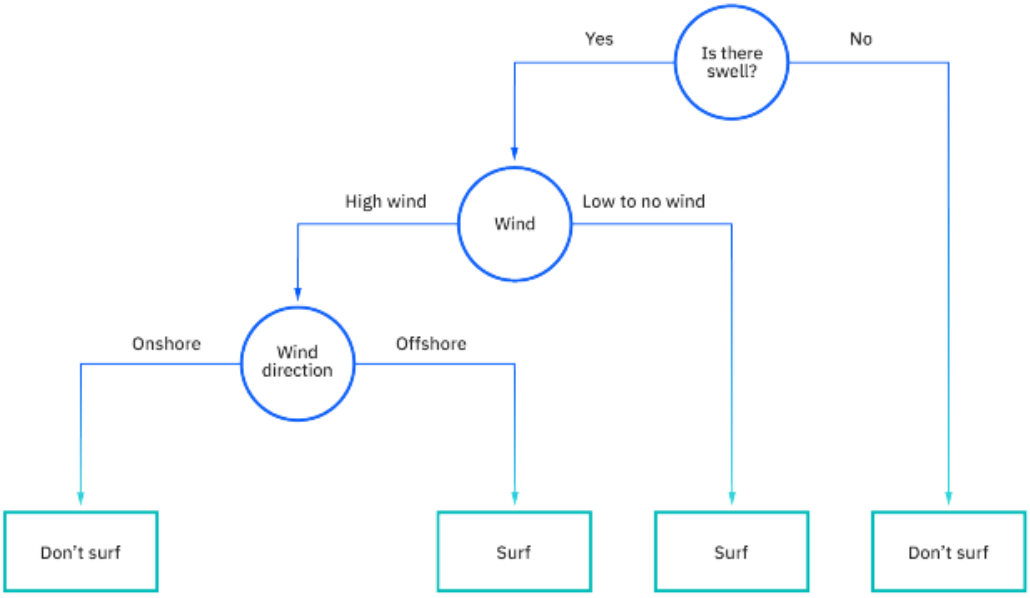


Abbildung 3: Ein Beispiel von einem Decision Tree in dem bestimmt wird, ob es gute Bedingungen zum Surfen gibt (IBM, What is a Decision Tree | IBM, kein Datum)

Der Vorteil von einem Decision Tree ist, dass dieser Algorithmus gut interpretierbar ist. Dies geht jedoch auf Kosten von der Präzision der Resultate. Es gibt aber Algorithmen, die auf Decision Trees basieren und trotzdem zu den besten Machine Learning Algorithmen gehören. (IBM, What is a Decision Tree | IBM, kein Datum)

2.4.1 Random Forest

Um genauere Resultate zu kriegen können mehrere Decision Trees verwendet werden, um danach mit dem Durchschnitt aller Bäume das endgültige Resultat vorherzusehen. Dieses Verfahren wird auch Bagging genannt, aber hat den Nachteil, dass die Decision Trees ähnlich aussehen werden, wenn eine Variabel die anderen dominiert. Aus diesem Grund wurde der Random Forest erstellt, der eine Erweiterung vom Bagging ist und dessen Nachteile umgehen soll. Beim Random Forest werden auch mehrere Decision Trees erstellt, aber es wird in jedem nur ein Subset der Variablen verwendet, damit auch nicht so dominante Variablen Einfluss auf das Endergebnis haben können.

2.4.2 Gradient Boosting

Eine weitere Methode die Decision Trees verwendet und trotzdem eine gute Performanz bietet ist Gradient Boosting. Ungleich wie beim Random Forest wo die Decision Trees unabhängig voneinander trainiert werden, wird beim Gradient Boosting ein Decision Tree anhand der Resultate vom vorherigen Decision Tree erstellt. Die Datenpunkte, die beim ersten Decision Tree schwer zu klassifizieren sind, erhalten im nächsten Decision Tree ein höheres Gewicht und soll somit besser sein als der vorherige. Dieser Kreislauf wird fortgeführt, bis die Anzahl Decision Trees die definiert wurde erreicht wurde.

2.5 Herstellung der Metalloxid-Varistoren

Zu Beginn des Herstellprozesses der Metalloxid-Varistoren, werden die Rohmaterialien für eines der beiden Rezepte vorbereitet. Die Charakteristiken der Rohmaterialien werden vom Lieferanten zur Verfügung gestellt und im SAP abgespeichert. Ein Rezept wird für die üblichen Varistoren verwendet, während das andere für hohe Spannungen benutzt wird. Nachdem die einzelnen Rohmaterialien gewogen wurden, werden diese mit Wasser umgerührt, bis es zu einer homogenen Paste wird. Um das Wasser beim Mixen zu entfernen, wird die Paste wieder in ein Pulver sprühgetrocknet und gesiebt, um grosse Partikel zu entfernen.

Das Pulver wird in zylinderförmige Scheiben gepresst, wobei die Höhe der Varistoren ausschlaggebend für die Durchbruchsspannung ist. Anschliessend werden die Scheiben in einem Sinterofen gesintert, was der wichtigste Teil vom Prozess ist. Die Varistoren befinden sich für etwa 48 Stunden im Sinterofen der verschiedenen Zonen mit verschiedenen Temperaturen hat. Die Temperaturen von den Zonen variieren zwischen 200-1200°C. Nach dem Sintern werden die Varistoren auf der Umfangsfläche meistens mit Glass beschichtet und auf der oberen und unteren Seite mit Aluminium. Alle Prozessdaten, die in der Fabrik aufgenommen werden, werden an eine Datenbank gesendet, die von einer externen Firma verwaltet wird. Diese Daten sind auf dem Factory Layer ersichtlich.

Zum Schluss werden die Varistoren noch getestet, um festzustellen welche elektrischen Eigenschaften sie besitzen. Je nach Resultaten werden sie in die Hauptklasse, eine von den Nebenklassen oder als Ausschuss klassifiziert. Ein Beispiel von den Tests wäre: Die benötigte Spannung damit 1mA durch den Varistor fliesst. Die Resultate der Tests werden im MES bzw. MOPA-Viewer hinterlegt und können lokal exportiert werden.

3 Datenaufbereitung

Dieses Kapitel beschreibt wie die Daten der Rohmaterialien, die Prozessdaten und die Testdaten angeeignet und aufbereitet wurden, um schlussendlich einen Datensatz zu erstellen. Dabei wird auf Probleme, die während der Aufbereitung ausgetreten sind und wie diese behandelt wurden eingegangen. Die Überkapitel spiegeln den eigentlichen Prozess dar während die Unterkapitel chronologisch, wie diese behandelt wurden aufgelistet sind. Die Daten von den Rohmaterialien haben keine Abhängigkeiten zu den anderen, aber die Prozessdaten und Testdaten werden am Schluss kombiniert (Abbildung 4). Für die Datenaufbereitung wurden Python-Skripte erstellt, die im Anhang (8.6 Übersicht Python-Skripte) genauer beschrieben wurden.

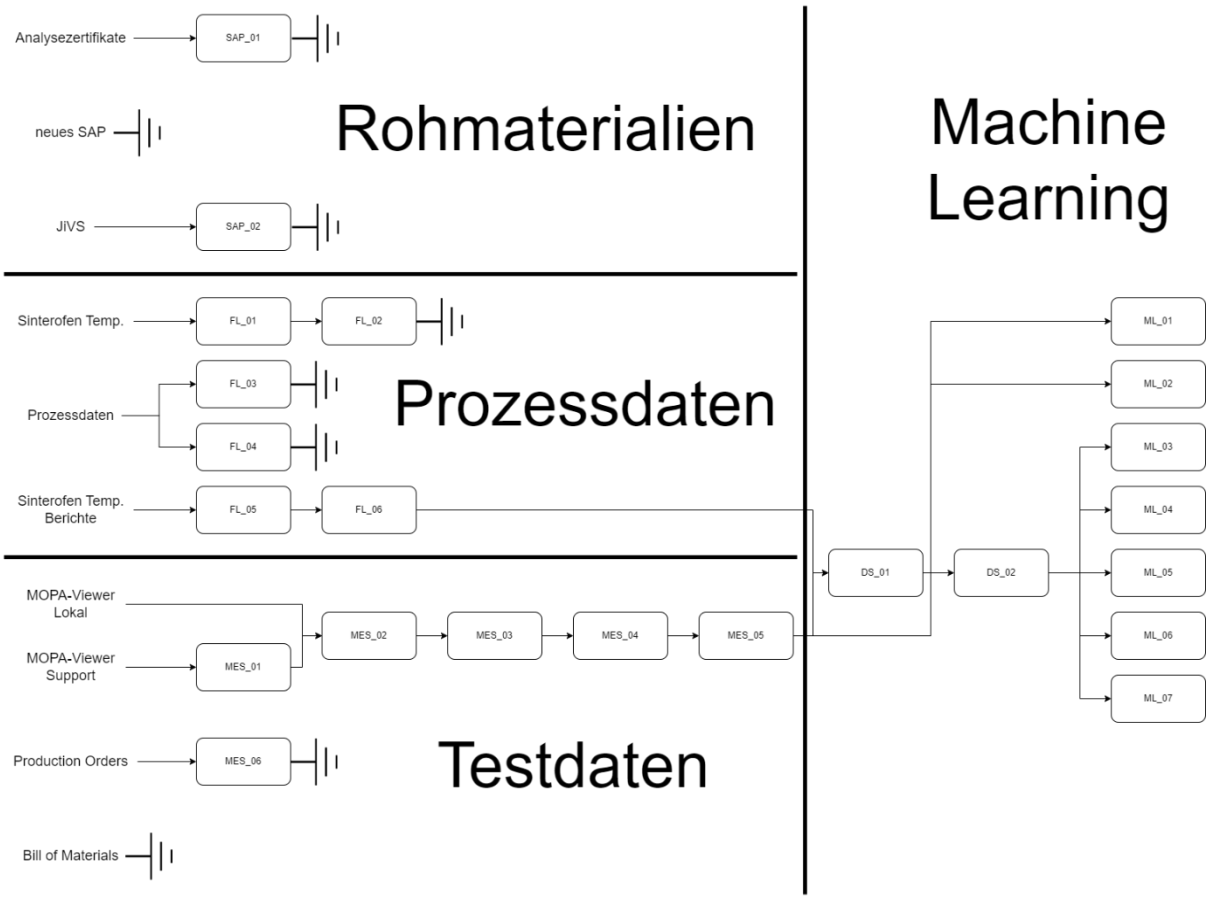


Abbildung 4: Übersicht der Python-Skripte und wie diese zusammenhängen

3.1 Rohmaterialdaten

Die Daten der Rohmaterialien beinhalten Informationen der Reinheiten bzw. Verunreinigungen und der Qualität der Materialien, die bei der Herstellung der Metalloxid-Varistoren verwendet werden. Insgesamt werden über 20 verschiedene Materialien für die Herstellung verwendet, wobei das Zinkoxid (ZnO) die Hauptkomponente ist.

3.1.1 Analysezertifikate

Bei jeder Lieferung von einem Material, stell der Lieferant ein Zertifikat zur Verfügung mit den Charakteristiken von der Charge. Die Charakteristiken beinhalten unter anderem die Reinheit vom Material und auch Angaben über die Menge von anderen Materialien in Prozent [%] oder Teile pro Million [ppm]. Über die Jahre wurden 209 Zertifikate von Hand gescannt und abgespeichert. Was zu beachten ist, ist dass die Analysezertifikate vom Lieferanten nicht konsistente Angaben hatten. Beispiele davon wären, dass ein Element verschiedene

Bezeichnungen in verschiedenen Zertifikaten hat oder dass eine Angabe in Prozent angegeben wurde, obwohl die Einheit als Teile pro Million eingetragen ist (Abbildung 5).

Item	Method	Unit	Content	Results
Sb_2O_3	Internal	%	99,8 min	99,94
As	ICP-OES	ppm	250 max	0,02
Pb	ICP-OES	ppm	300 max	0,02
Fe	ICP-OES	ppm	50 max	7
Cu	ICP-OES	ppm	20 max	8
Median Grain size (d_{50})	Internal	µm	< 3	2,30
Sieve Residue 45 µm	Internal	%	< 0,1	0,004
Moisture H_2O	Internal	%	< 0,1	0,03

Abbildung 5: Angabe in Prozent anstelle Teile pro Million von Antimonoxid Sb2O3 Analysezertifikat

Um die Arbeit zu erleichtern, wurde versucht mit Python-Bibliotheken wie Camelot und Tabula die gescannten PDFs in Tabellen umzuwandeln. Diese hatten jedoch Abhängigkeiten von anderen Programmen, die Einträge in den Umgebungsvariablen vom System benötigen. Das Erkennen der Tabellen stellte sich jedoch als schwieriger heraus als zuerst angenommen, weil Camelot nur Text-basierende PDF einlesen kann und die PDFs bild-basierend sind und Tabula zwar auch für Bild-basierende PDFs gemacht wurde, aber die grossen Lücken zwischen den Spalten wahrscheinlich für Schwierigkeiten sorgen (Abbildung 6). Zudem findet Tabula die Software «jib2-imageio» nicht, obwohl dies installiert und als Umgebungsvariable angegeben wurde. Beide Bibliotheken haben somit keine Tabellen in den PDFs erkannt und diese Methode wurde somit verworfen. Es wurde noch die Bibliothek pytesseract versucht, aber diese fand eine Software nicht, obwohl diese installiert wurde wie bei Tabula.

Lot No: P8454

Characteristic	Inspection Method	Value	Unit
Chemical Analysis			
Al		99,81	WT%
Cu		0,0026	WT%
Fe		0,1300	WT%
Mn		0,0012	WT%
Mg		0,0003	WT%
Si		0,0200	WT%
Ti		0,0002	WT%
Zn		0,0070	WT%
Others Each		0,0140	WT%
T.A.O.		0,0269	WT%
Chemical Analysis			
Tens.stren	DIN 1790-1	195,0	N/mm2

Abbildung 6: Analysezertifikat von Alu-Draht mit grossem Leerraum zwischen den Spalten

Um wenigstens erste Analysen zu machen, wurde angefangen die Analysezertifikate vom Zinkoxid von Hand abzuschreiben, bis erwähnt wurde, dass die Analysezertifikate auch in einer Software, namens SAP, abgespeichert werden.

3.1.2 Neues SAP

Aufgrund der Migration innerhalb der Firma, wurden die Daten aus den Zertifikaten in einem neuem SAP (System Applications and Products in Data Processing) abgespeichert. Nach

dem ersten Export, welche vier separate Tabellen enthält, wurde jedoch klar, dass die alten Daten vor der Migration nicht auf das neue SAP migriert wurden und es nur Eintrage ab Juli 2022 gibt. Weil Daten ab 2018 benötigt werden, wurden die Daten aus dem neuen SAP nicht weiterverfolgt.

3.1.3 JiVS

Nach weiterer Suche stellte sich heraus, dass die alten Daten der Analysezertifikate sich auf einer Archivierungs-Plattform namens JiVS befinden, welches für das Migrieren von Daten zuständig ist (Abbildung 7). Auf JiVS können alle Lose von jedem Material eingesehen werden, aber die Charakteristiken befinden sich in verschachtelten Tabellen. Das heisst, dass die Angaben der Charakteristiken nicht auf einmal exportiert werden können, sondern manuell einzeln exportiert werden müssen. Insgesamt gab es 927 Lose von 19 Materialien die Exportiert wurden und somit für weitere Analysen zur Verfügung gestanden sind. Die Übersicht von jedem Material musste auch exportiert werden, weil diese die Batch-Nummer vom Lieferanten beinhaltet, welche für die Zuordnung zum BOM (Bill of Materials) im MES benötigt wird.

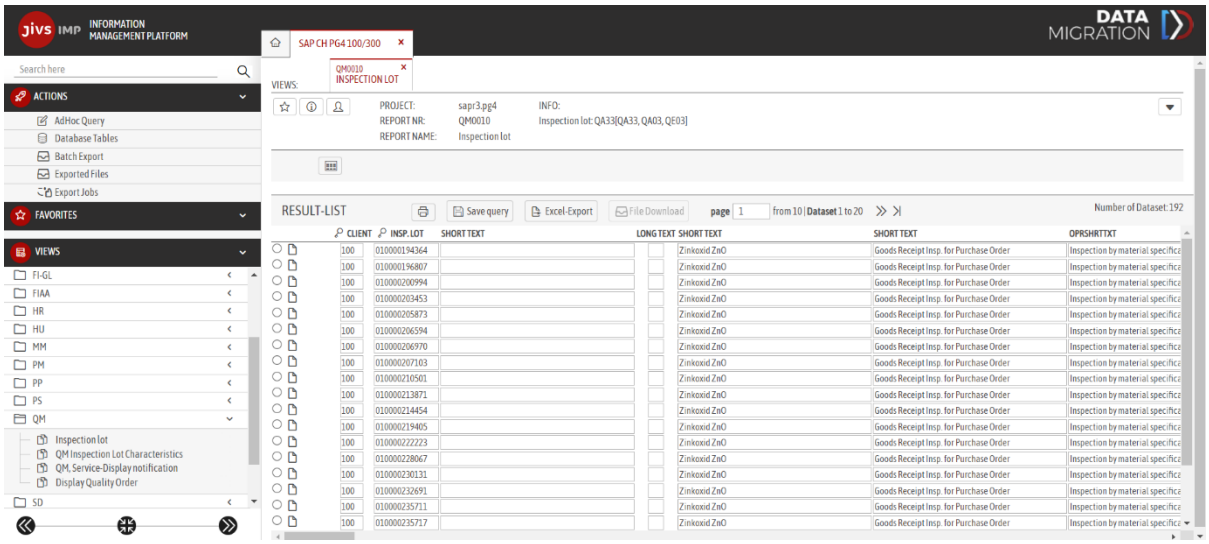


Abbildung 7: Ausschnitt JiVS gefiltert nach Zinkoxid ZnO

Ein Python-Skript wurde erstellt, um die Lose von jedem Material in ein CSV-File zu kombinieren und auch hier sind einige Probleme aufgetreten. Einerseits sind die Bezeichnungen nicht immer konsistent, einige Exports sind komplett leer bzw. haben keine Angaben (5 Exports) und es gibt viele (insgesamt über 45'000) Duplikate bei den Bezeichnungen in den einzelnen Losen.

Um die Duplikate zu entfernen, wurden alle doppelten Bezeichnungen wo einen Wert von null, einen identischen Wert oder eine leere Zelle hatten entfernt. Aber es stellte sich heraus, dass es trotzdem 24 Einträge gab, die die gleiche Bezeichnung im selben Los hatten, aber über einen anderen Wert verfügten (Abbildung 8). Weil dazu keine weiteren Informationen zur Verfügung gestanden sind, wurde provisorisch der Mittelwert verwendet. Weitere Analysen der kombinierten CSV-Files ergaben, dass einige Lose einen Hauptanteil von über 100% hatten und nur 11 der ursprünglich 24 Materialien nützliche Informationen beinhalteten. Als leere Spalten und Zeilen entfernt wurden, stellte sich heraus, dass beim Zinkoxid nur 31 von 192 Losen Angaben beinhalteten und die meisten davon hatten nur einen Eintrag für den Hauptanteil. Nach einem Gespräch mit einem Experten wurde entschieden, dass bei der Migration Daten verloren gegangen sind bzw. die Daten inkomplett sind und deren Ursache

weiterverfolgt werden muss. Weil dies gegen Ende vom Projekt entschieden wurde, wird das im Umfang von diesem Projekt nicht weiterverfolgt.

	A	B	C	D	E	F	G	H	I
1	Client	Insp. Lot	Node	Char.	MeasValue	IndResult	Status	Short text	Msmt unit
2	100	010000202917	00000001	0010			1	Identifikation und Menge	
3	100	010000202917	00000003	0010	17.86	00000001	0	Partikelgrössenverteilung D 84	MIM
4	100	010000202917	00000001	0020			1	Verpackung:	
5	100	010000202917	00000003	0020	6.505	00000001	0	Partikelgrössenverteilung D 50	MIM
6	100	010000202917	00000002	0030	22.69	00000001	0	Partikelgrössenverteilung D 84	MIM
7	100	010000202917	00000003	0030	1.281	00000001	0	Partikelgrössenverteilung D 16	MIM
8	100	010000202917	00000002	0040	7.72	00000001	0	Partikelgrössenverteilung D 50	MIM
9	100	010000202917	00000002	0050	2.07	00000001	0	Partikelgrössenverteilung D 16	MIM
10	100	010000202917	00000002	0060	5.29	00000001	0	Dichte (g / cm3)	
11	100	010000202917	00000002	0070	66.5	00000001	0	Ausdehnungskoeffizient RT -260°C, 10-7°C	
12	100	010000202917	00000002	0080	475	00000001	0	Erweichungspunkt (° C)	

Abbildung 8: Verschiedene Werte für gleiche Bezeichnungen von Glaspulver. Gleiche Bezeichnungen sind mit gleichen Farben markiert

3.2 Prozessdaten

Die Prozessdaten von den Pressen, dem Sinterofen, etc. werden in einem Factory Layer verwaltet, wo von einer externen Firma gepflegt wird. Von der externen Firma wurden Backups für Microsoft SQL Server erstellt und zur Verfügung gestellt. Insgesamt wurden acht Backups von Datenbanken gemacht, wobei sechs davon vom Factory Layer und die anderen zwei von einem alten System sind von der aber auch Daten im Factory Layer verwendet werden.

3.2.1 Factory Layer

Im Factory Layer befinden sich die Prozessdaten aus der Fabrik und beinhalten Daten von Anlagen wie den Sinterofen, Sprühturm und Pressen.

Das kleinere Backup kommt von der Online-Datenbank und beinhaltet Daten von den letzten 160 Tagen und ist 800MB gross. Die Archiv-Datenbanken wurden aufgrund der Grösse in Jahre von 2018 bis 2022 aufgeteilt, wobei die Datenbank vom Jahr 2018 auch Daten ab 2015 beinhaltet. Zusammen haben die Backups eine Grösse von 47.4GB, wobei die Tabellen von den Sinterofen Temperaturen einen Grossteil davon ausmachen.

3.2.1.1 Sinterofen Temperaturen

Die Tabellen von den Sinterofen Temperaturen («FLUD_SinterofenTemperaturen») wurden alle als CSV-Files exportiert und sind insgesamt 35GB gross, aus diesem Grund wurden sie zu PARQUET-Files umgewandelt, damit die Einlesezeit der Files schneller ist und um die Grösse der Files auf insgesamt 2GB zu reduzieren. Ein PARQUET-File ist ein Spaltenorientiertes Datenformat für effizientere Speicherung und schnelleres Einlesen von Daten (Apache Parquet, kein Datum). Es ist zu beachten, dass im Export vom Jahr 2018 nur ca. 12% der Einträge einen realistischen Wert hatten. Die Grösse der Tabellen liegt daran, dass jede Minute meistens 98 Temperaturwerte auf einmal aufgenommen werden und in sechs Spalten abgespeichert werden. Zwei davon sind die Sollwerte, zwei sind die Istwerte und die letzten zwei sind die Stellwerte. Zu jedem Eintrag wird der Index, der Zeitstempel, die Charge und eine Laufende Nummer angegeben, womit sich 10 Spalten ergeben. Weil der Zeitstempel für alle 98 Einträge immer identisch ist, können die Werte nur mit der Laufender Nummer identifiziert werden. Eine Laufende Nummer beinhaltet zwischen 15 und 130 Varistoren und wurde eingeführt, weil es zu viele Varistoren in einem Batch gibt (Abbildung 9).

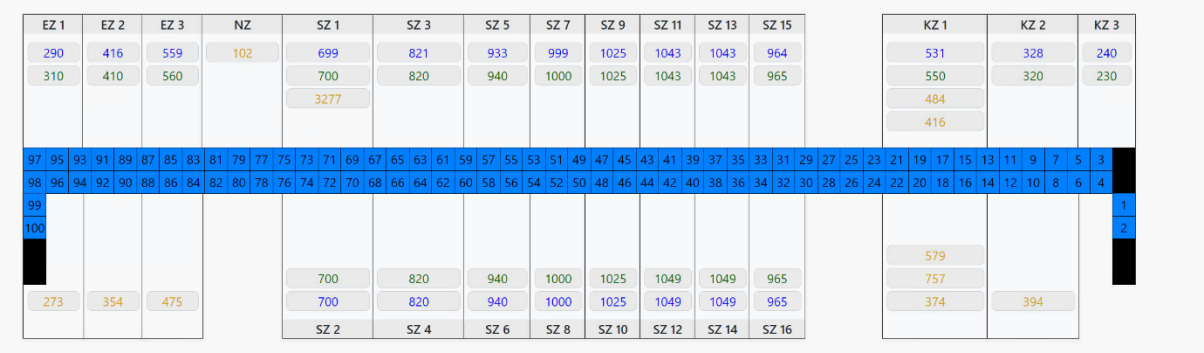


Abbildung 9: Zonen vom Sinterofen mit den Laufenden Nummern (blaue Quadrate) die 15 bis 130 Metalloxid-Varistoren beinhalten. Die Zahlen in blau sind die Ist-Temperaturen und die in grün sind die Soll-Temperaturen.

Aufgrund der Grösse der Files, konnten diese nicht kombiniert werden, weil 32GB RAM auf dem Rechner nicht ausreichen. Aus diesem Grund wurden die Soll- und Stellwerte entfernt und der zweite Istwert wurde auch entfernt, weil die Temperaturdifferenz vernachlässigbar ist. Das 5% Quantil ist -5.89°C und das 95% Quantil ist 3.89°C womit 90% der Werte der Differenz vom zweiten Istwert sich in diesem Intervall befinden. Die Q1 und Q3 Quartile sind in vier der sechs Jahren beide auf null. Somit konnte sichergestellt werden, dass die zweite Spalte fast identische Werte annimmt, wie die erste Spalte und wurde deshalb entfernt. Der Zeitstempel wurde in der Datenbank im UTC-Format abgespeichert und musste somit auch zur Schweizer Lokalzeit konvertiert werden. Nachdem die Datentypen der übrigen fünf Spalten angepasst wurden und somit weniger Speicherplatz benötigen, konnten alle Tabellen zu einer grossen Tabelle kombiniert werden.

Auch in dieser Tabelle befanden sich mehrere Fehler wie unmögliche Werte (z. B. 10^{36}°C) und Sensorfehler (z. B. 3200°C) oder Werte mit zu hoher Genauigkeit (z. B. 10^{-40}) die entfernt werden mussten (Abbildung 10).

SQLQuery1.sql - l...(CORP\DA2442 (68))*

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT *
  FROM [FactoryLayerGXP_Archiv2018].[dbo].[FLUD_SinterofenTemperaturen]
  WHERE [Los_Nummer] < 0
  OR [Los_LaufendeNummer] < 0
  OR [Temperatur_SP1] < -10000 OR [Temperatur_SP1] > 10000
  OR [Temperatur_SP2] < -10000 OR [Temperatur_SP2] > 10000
  OR [Temperatur_CV1] < -10000 OR [Temperatur_CV1] > 10000
  OR [Temperatur_CV2] < -10000 OR [Temperatur_CV2] > 10000
  OR [Temperatur_ST1] < -10000 OR [Temperatur_ST1] > 10000
  OR [Temperatur_ST2] < -10000 OR [Temperatur_ST2] > 10000
```

100 %

Results | Messages

	Los_Nummer	Los_LaufendeNummer	Temperatur_SP1	Temperatur_SP2	Temperatur_CV1	Temperatur_CV2	Temperatur_ST1	Temperatur_ST2
1	-1878982400	0	2.35329860203317E-40	2.35098870164458E-38	0	0	1.83670992315982E-40	0
2	0	-28671	2.35098870164458E-38	2.35329860203317E-40	2.35098870164458E-38	2.35329860203317E-40	2.35098870164458E-38	2.35329860203317E-40
3	-1878982400	0	2.35329860203317E-40	2.35098870164458E-38	0	0	1.83670992315982E-40	0
4	0	-28671	2.35098870164458E-38	2.35329860203317E-40	2.35098870164458E-38	2.35329860203317E-40	2.35098870164458E-38	2.35329860203317E-40
5	33947984	488	2.79185042677867E-37	1.23064833733934E-38	-3.17736876477326E+36	0	2.70007933429809E-38	3.85681057894277E-38
6	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07794433614791E-28	2.42144374635328E-41	-3.92578995690668E+23
7	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07794337318294E-28	2.42144374635328E-41	-3.92578995690668E+23
8	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07794241021797E-28	2.42144374635328E-41	-3.92578995690668E+23
9	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07794144725299E-28	2.42144374635328E-41	-3.92578995690668E+23
10	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07794048428802E-28	2.42144374635328E-41	-3.92578995690668E+23
11	1134231552	17307	2.42522725220696E-41	2.42522725220696E-41	-3.92237406666131E+23	8.07793952132305E-28	2.42144374635328E-41	-3.92578995690668E+23
12	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691356160	8.07793855835808E-28	2.43237387437502E-41	-31710932992
13	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691356160	8.07793759539311E-28	2.43237387437502E-41	-31710932992
14	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691356160	8.07793663242813E-28	2.43237387437502E-41	-31710932992
15	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691354112	-2.90143045257738E...	2.43237387437502E-41	-31710932992
16	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691354112	-2.90143026810994E...	2.43237387437502E-41	-31710932992
17	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691354112	-2.90143008364425E+26	2.43237387437502E-41	-31710932992
18	1137508352	17357	2.43223374452859E-41	2.43223374452859E-41	-31691354112	-2.90142899917506E...	2.43237387437502E-41	-31710932992

Query executed successfully. | localhost (16.0 RTM) | CORP\DA2442 (68) | FactoryLayerGXP_Archiv... | 00:00:50 | 66 rows

Abbildung 10: Einträge in der Factory Layer Datenbank aus dem Jahr 2018 die Fehlerhaft sind

Es wurde entschieden nur Temperaturen zwischen 20°C und 1250°C zu behalten. Zusätzlich befinden sich Einträge von Chargen, die nicht im MES zugeordnet werden können. Einerseits gibt es Testlose (1XXXXxxxx) bzw. Fake-Lose, die für Tests oder Versuche verwendet wurden, aber keine Dokumentation über den Grund der Durchführung haben. Und andererseits gibt es Dummys wo eine Laufende Nummer von null haben, welche dazu dienen den Ofen vorzuheizen. Nur die Einträge mit einer gültigen Chargennummer (5XXXXxxxx) wurden beibehalten, wobei hier auch wieder ein Batch in mehrere kleineren Batches aufgeteilt werden kann. Ein Beispiel dafür wäre 520790000 und 520791111, wobei beide Losnummern vom gleichen Batch kommen. Negative Werte und null-Werte für die Losnummern und Laufende Nummern wurden auch jeweils zweimal beobachtet.

Laut Angaben von einem Experten sollen die Temperaturen aus dem Sinterofen mit den verschiedenen Zonen modelliert werden. Dies ist jedoch schwer, weil diese Angaben in der Tabelle fehlt. Es wäre möglich diese Anhand den Längen der einzelnen Zonen (Abbildung 11) und der Schubgeschwindigkeit vom Ofen (ca. 500mm/h) zu berechnen, aber aufgrund des grossen Aufwands, wurde aus Zeitgründen dies nicht weiterverfolgt. Weil die Varistoren in der Fabrik manchmal von Hand von einer Anlage zur anderen verschoben werden, kann nur auf ganze Batches zurückverfolgt werden, anstatt auf einzelne Varistoren.

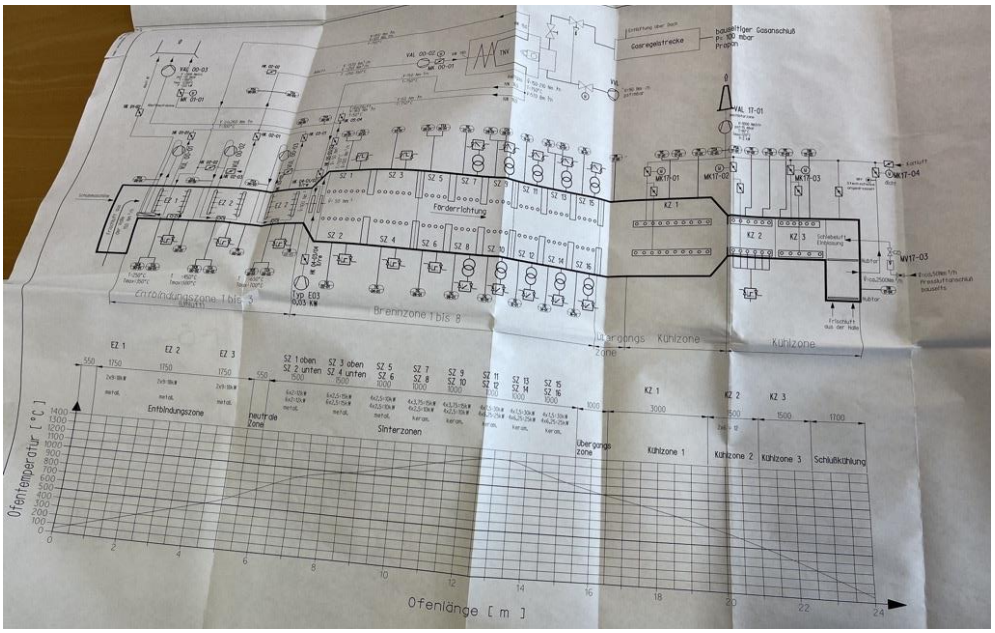


Abbildung 11: Plan mit Abmessungen vom Sinterofen (alternative Bilder wurden nicht zur Verfügung gestellt)

3.2.1.2 Restliche Prozessdaten

Später im Verlauf vom Projekt wurde entdeckt, dass es unter anderem noch Daten von der Pulveraufbereitung, dem Sprühturm, den Pressen, dem Orbitofen und der Schichtdicke in einer anderen Tabelle gibt. Alle Informationen über diese Werte wurden in sechs verschiedenen Tabellen abgelegt, wovon drei als Legende dienen. Die Sensoren wurden im Verlauf der letzten Jahre erweitert und somit hat es Sensoren die Daten seit 2015 generiert haben, aber auch Sensoren die erst Ende 2022 installiert wurden.

Von den drei Tabellen, die zu einer Legende gehören, teilt eine jedem Sensor eine eigene ID zu und enthält eine grobe Beschreibung («FLC_Datapoint»). Die zweite enthält eine Liste mit den verschiedenen Anlagen in der Fabrik und teilt jeder eine Gruppen-ID zu («FLC_DatapointGroup») und die letzte weist die Sensor-IDs den Gruppen-IDs zu («FLC_DatapointGroupPoint»). Anhand diesen drei Tabellen wurde eine Legende erstellt, wo

beschreibt was der Sensor misst, zu welcher Anlage dieser gehört und wann diese Sensoren integriert wurden.

Alle Werte der Sensoren wurden in zwei Tabellen abgespeichert, eine enthält nur numerische Werte («FLC_LogEntryReal», Abbildung 12) und die andere nur Strings («FLC_LogEntryString»). Beide haben eine Spalte für eine ID, die Sensor-ID, eine Revisionsnummer und den Wert. Die Tabelle mit den numerischen Werten hat das gleiche Problem wie die von den Sinterofen Temperaturen, wo es nicht möglich ist alle Tabellen von allen Jahren zu kombinieren, weil der Rechner zu wenig RAM hat. Eine dritte Tabelle enthält die IDs mit den dazugehörigen Zeitstempel, weil unter einer ID, mehrere Sensorwerte auf einmal eingetragen werden können.

Results | Messages

	LogEntryID	DatapointID	Revision	Value
1	50002876805997000	51	0	0
2	50002876805997000	52	0	7
3	50002876805997000	52	1	5
4	50002876805997000	53	0	5
5	50002876805997000	53	1	7
6	50002876805997000	53	2	1
7	50002876805997000	59	0	0
8	50002876805997000	60	0	102
9	50002876806795000	51	0	0
10	50002876806795000	52	0	1
11	50002876806795000	52	1	5
12	50002876806795000	53	0	5
13	50002876806795000	53	1	7
14	50002876806795000	53	2	1

Abbildung 12: Ausschnitt aus der Factory Layer Datenbank Tabelle «FLC_LogEntryReal». Die ersten acht Einträge haben die gleiche ID und dort wo auch eine gleiche DatapointID ist, gibt es eine neue Revisionsnummer. In der rechten Spalte befinden sich die Werte.

Das Ziel mit diesen Tabellen, war es eine Tabelle für jede Anlage zu erstellen, in den Zeilen nur einmalige IDs zu haben und als Spalten die Beschreibung aus der Tabelle «FLC_Datapoint» einzufügen. Um dies zu ermöglichen wurde ein Python-Skript erstellt mit folgendem Ablauf:

1. Files einlesen (bei Pandas latin1 als Kodierung verwenden → encoding='latin1')
2. Die drei Tabellen zuständig für die Legende kombinieren und diese exportieren
3. Aus den Tabellen mit den Werten nur die höchste Revision beibehalten (es wurden in jedem Jahr zwischen 13 und 19 Einträge in der numerischen Tabelle aufgrund einer neueren Revision entfernt ausser in der Online Datenbank, wo 131 Einträge entfernt wurden und zwischen 0 und 9 Einträge in der Tabelle mit Strings)
4. Eine Pivot-Tabelle für beide Tabellen mit den Werten erstellen, wo die Sensor-ID sich in den Spalten befindet und die Zeilen einmalige IDs der Einträge haben. Die Tabelle soll dann mit den Werten gefüllt werden
5. Beide Pivot-Tabellen mit der Tabelle wo die Zeitstempel beinhaltet kombinieren für eine einzige grosse Tabelle mit allen angaben
6. Aus der Legende den Spalten die dazugehörige Beschreibung zuteilen
7. Die Tabellen in die Anlagen aufteilen und exportieren

Weil dieser Prozess jährlich durchgeführt wurde, wurden Exports mit verschiedenen Dimensionen erstellt, weil im Jahr 2018 es nur 47 Sensoren hatte, aber dies im Jahr auf 133 Sensoren erweitert wurde.

Weil die restlichen Prozessdaten entweder keine Aussage über die Qualität der Varistoren machen oder weil die Daten erst seit kurzem in der Fabrik integriert wurden, werden diese nicht weiterverfolgt. Zudem wurde von einem Experten empfohlen die einzelnen Zonen vom

Sinterofen zu Modellieren, aber anhand der Daten von den Sinterofen Temperaturen, ist dies schwer einzuteilen, weil die Temperatur von einer Minute auf die andere mehrere 100°C zunehmen kann und es wird nicht angegeben in welcher Zone die Laufende Nummer sich gerade befindet. Später wurde auf eine weitere Datenquelle, das Legacy System, aufmerksam gemacht wo verwendet wird, um Berichte von den jeweiligen Zonen zu erstellen.

3.2.2 Legacy DB

Die Legacy Datenbank, die Daten seit 2009 hat, hat auch eine Online- und eine Archiv-Datenbank, wobei die Online-Datenbank 600MB gross ist und das Archiv 3.1GB. In dieser Datenbank befinden sich Berichte von den Sinterofen Temperaturen im Factory Layer und man kann nach dem tiefsten, dem Durchschnitt oder der höchsten Temperatur einer Zone von jedem Batch filtern. Auch hier mussten mehrere Tabellen kombiniert werden, damit alle Informationen auf einmal abgerufen werden können. Die Tabelle «SO_AufbauDaten» (Abbildung 13) enthält die einzelnen MIN-, AVG- und MAX-Werte und werden einer ID und einer Sensor-ID zugeteilt. In der Tabelle «SO_Sensoren_und_Zonen» befindet sich eine Legende von den Sensor-IDs und der dazugehörigen Beschreibung und in der Tabelle «Aufbau» werden die IDs zu den Losen zugeteilt.

Results | Messages

	SO_Aufbau_IDX	Sensor_IDX	T_MIN	T_MAX	T_AVG	SL_AVG	OD_AVG	START	ERR_1	ERR_2	Created	SO_Aufbau_AutoID
1	2050050187	1	261	267	310	0	0	0	0	0	2018-01-22 11:43:20.423	2558791
2	2050050187	2	401	424	410	0	0	0	0	0	2018-01-22 16:49:21.733	2559034
3	2050050187	3	555	564	560	0	0	0	0	0	2018-01-22 20:55:23.403	2559218
4	2050050187	4	699	706	700	0	0	0	0	0	2018-01-22 23:59:19.713	2559352
5	2050050187	5	699	701	700	0	0	0	0	0	2018-01-22 23:59:19.720	2559353
6	2050050187	6	815	827	820	0	0	0	0	0	2018-01-23 03:02:20.080	2559474
7	2050050187	7	819	821	820	0	0	0	0	0	2018-01-23 03:02:20.087	2559475
8	2050050187	8	936	944	940	0	0	0	0	0	2018-01-23 05:05:20.847	2559560
9	2050050187	9	939	941	940	0	0	0	0	0	2018-01-23 05:05:20.850	2559561
10	2050050187	10	1035	1044	1040	0	0	0	0	0	2018-01-23 08:08:21.377	2559684
11	2050050187	11	1039	1041	1040	0	0	0	0	0	2018-01-23 08:08:21.380	2559685
12	2050050187	12	1091	1106	1095	0	0	0	0	0	2018-01-23 10:10:21.253	2559766
13	2050050187	13	1092	1105	1090	0	0	0	0	0	2018-01-23 10:10:21.260	2559767
14	2050050187	14	1137	1141	1139	0	0	0	0	0	2018-01-23 12:13:21.927	2559851

Abbildung 13: Ausschnitt aus der Legacy-Datenbank Tabelle «SO_AufbauDaten» mit den minimalen, maximalen und durchschnittlichen Temperaturen der einzelnen Zonen

Auch in dieser Datenbank gab es einige Sensorfehler wie 3000°C, die herausgefiltert werden mussten und somit wurden nur Temperaturen zwischen 50°C und 1250°C beibehalten. Was festgestellt wurde, ist das die Legende aus den Tabellen «SO_Sensoren_und_Zonen» im der Online- und Archiv Datenbank nicht identisch sind. Weil die aus der Online Datenbank aktueller ist, wurde mit dieser weitergearbeitet.

Schlussendlich hatte die Tabelle den minimalen, den Durchschnitt und den maximalen Wert für folgende Angaben im Sinterofen:

- Alle Zonen
- Schubluft
- Ofendruck

Die Tabelle hatte somit 73 Spalten mit 1900 Zeilen, wobei einige Spalten konstante oder gar keine Werte hatten und der Datensatz somit auf 60 Spalten mit 1826 Zeilen reduziert wurde.

3.2.3 Langzeit-Stabilitätsmessungen

Von jedem Batch werden jeweils drei Varistoren als Proben für Langzeit-Stabilitätsmessungen verwendet, um die Charakterisierungs-Messungen zu ermitteln. Dafür werden diese Varistoren für mehrere Tage in einen Ofen verlegt und es wird andauernd Spannung angelegt. Diese Daten befinden sich in einer Software namens CSadmin welches aufgrund der vielen

Daten sehr langsam ist (Abbildung 14). Zudem kommen Störungen öfters vor und Exports sind nicht möglich, sondern sie müssen mit Copy-Paste selbst erstellt werden. Weil diesen Daten erst spät im Projektverlauf erwähnt wurden, wurden diese im Rahmen von diesem Projekt nicht weiterverfolgt.

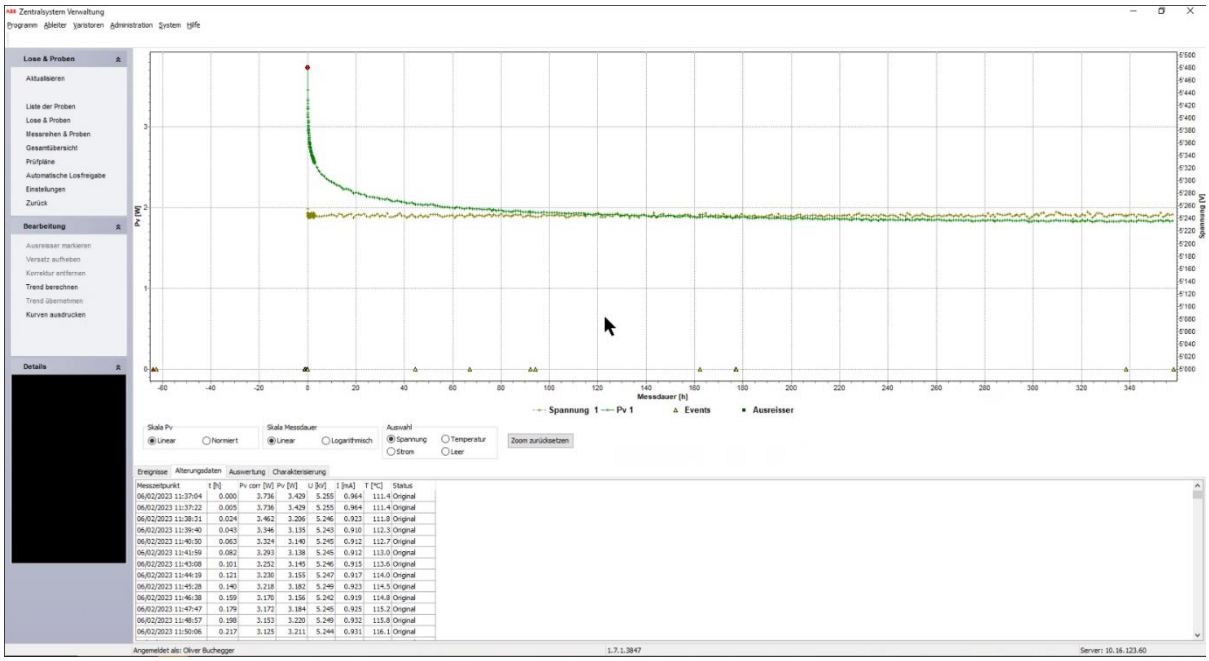


Abbildung 14: Ausschnitt aus CSadmin mit Langzeit-Stabilitätsmessungen

3.3 Testdaten (Manufacturing Execution System)

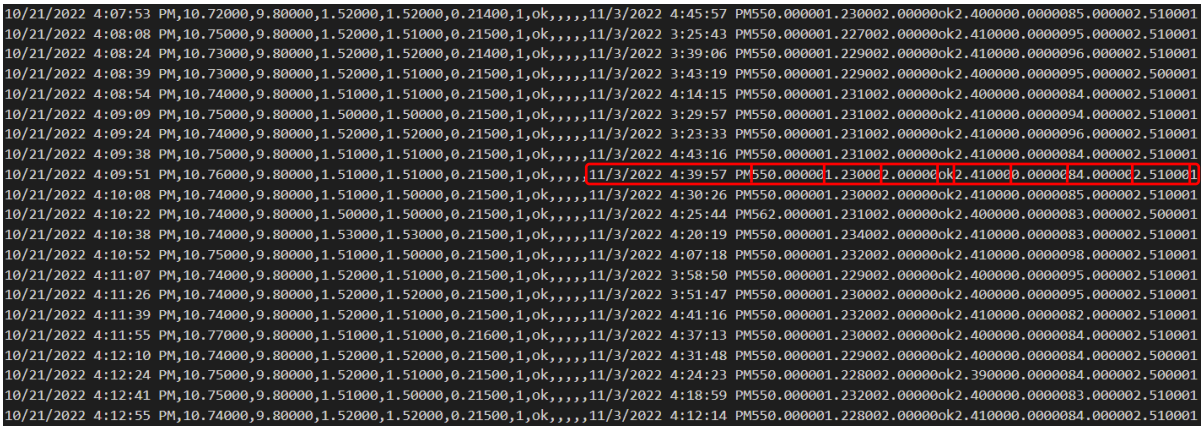
Die Testdaten werden vor Ort gemacht und jeder einzelne Metalloxid-Varistor wird in der Testanlage getestet. Die Testanlage wurde Ende 2022 mit neuen Tests erweitert und hat somit nur wenig Daten von diesen Tests. Im Anhang (8.3 Legende MES-Daten (MOPA-Viewer)) gibt es eine Legende mit allen Spalten aus dem MOPA-Viewer in der jede Spalte genauer beschrieben wurde.

3.3.1 MOPA-Viewer

Aus den 63 Spalten haben 32 Messdaten aus Tests, die in der Fabrik durchgeführt werden. Viele Spalten sind redundant, weil sie keine Werte haben oder nur Nullen haben und einige Tests sind erst Ende 2022 eingeführt worden und somit sind diese Spalten zum grössten Teil leer. Dies führte zum Problem, dass beim Einlesen der Exports das Datenformat nicht richtig erkannt wird und einer Spalte wo eigentlich einen Zeitstempel haben sollte, den Datentyp «float» zugewiesen wurde. Die Spalte «Pulver_Charge_Nr» hatte auch mehrere Errors und im Export vom Jahr 2020 waren fast alle Zeilen davon betroffen. Die Exports aus dem MOPA-Viewer mussten ausserdem auch jährlich exportiert werden, weil Microsoft Excel ein Zeilenlimit von ca. 1 Millionen Zeilen hat.

3.3.2 MES-Support

Wegen den fehlerhaften Exports aus dem MOPA-Viewer, wurde nach einem CSV-Export direkt beim MES-Support angefragt. Es wurde ein 1.4GB grosses CSV-File exportiert, wo aber die Jahre eine falsche Reihenfolge hatten. Zudem gab es auch beim CSV-Export einen Fehler, wo die Kommastellen in den Spalten von den Tests wo später eingeführt wurden falsch positioniert wurden und somit die letzten 10 Spalten in eine Spalte als Strings kombiniert wurden (Abbildung 15).



```
10/21/2022 4:07:53 PM,10.72000,9.80000,1.52000,1.52000,0.21400,1,ok,,,,,11/3/2022 4:45:57 PM550.000001.230002.00000ok2.400000.0000085.000002.510001
10/21/2022 4:08:08 PM,10.75000,9.80000,1.52000,1.51000,0.21500,1,ok,,,,,11/3/2022 3:25:43 PM550.000001.227002.00000ok2.410000.0000095.000002.510001
10/21/2022 4:08:24 PM,10.73000,9.80000,1.52000,1.52000,0.21400,1,ok,,,,,11/3/2022 3:39:06 PM550.000001.229002.00000ok2.410000.0000096.000002.510001
10/21/2022 4:08:39 PM,10.73000,9.80000,1.52000,1.51000,0.21500,1,ok,,,,,11/3/2022 3:43:19 PM550.000001.229002.00000ok2.400000.0000095.000002.500001
10/21/2022 4:08:54 PM,10.74000,9.80000,1.51000,1.51000,0.21500,1,ok,,,,,11/3/2022 4:14:15 PM550.000001.231002.00000ok2.400000.0000084.000002.510001
10/21/2022 4:09:09 PM,10.75000,9.80000,1.50000,1.50000,0.21500,1,ok,,,,,11/3/2022 3:29:57 PM550.000001.231002.00000ok2.410000.0000094.000002.510001
10/21/2022 4:09:24 PM,10.74000,9.80000,1.52000,1.52000,0.21500,1,ok,,,,,11/3/2022 3:23:33 PM550.000001.231002.00000ok2.410000.0000096.000002.510001
10/21/2022 4:09:38 PM,10.75000,9.80000,1.51000,1.51000,0.21500,1,ok,,,,,11/3/2022 4:43:16 PM550.000001.231002.00000ok2.410000.0000084.000002.510001
10/21/2022 4:09:51 PM,10.76000,9.80000,1.51000,1.51000,0.21500,1,ok,,,,,11/3/2022 4:39:57 PM550.000001.230002.00000ok2.410000.0000084.000002.510001
10/21/2022 4:10:08 PM,10.74000,9.80000,1.51000,1.50000,0.21500,1,ok,,,,,11/3/2022 4:30:26 PM550.000001.230002.00000ok2.410000.0000085.000002.510001
10/21/2022 4:10:22 PM,10.74000,9.80000,1.50000,1.50000,0.21500,1,ok,,,,,11/3/2022 4:25:44 PM562.000001.231002.00000ok2.400000.0000083.000002.500001
10/21/2022 4:10:38 PM,10.74000,9.80000,1.53000,1.53000,0.21500,1,ok,,,,,11/3/2022 4:20:19 PM550.000001.234002.00000ok2.410000.0000083.000002.510001
10/21/2022 4:10:52 PM,10.75000,9.80000,1.51000,1.50000,0.21500,1,ok,,,,,11/3/2022 4:07:18 PM550.000001.232002.00000ok2.410000.0000098.000002.510001
10/21/2022 4:11:07 PM,10.74000,9.80000,1.52000,1.51000,0.21500,1,ok,,,,,11/3/2022 3:58:50 PM550.000001.229002.00000ok2.400000.0000095.000002.510001
10/21/2022 4:11:26 PM,10.74000,9.80000,1.52000,1.52000,0.21500,1,ok,,,,,11/3/2022 3:51:47 PM550.000001.230002.00000ok2.400000.0000095.000002.510001
10/21/2022 4:11:39 PM,10.74000,9.80000,1.52000,1.51000,0.21500,1,ok,,,,,11/3/2022 4:41:16 PM550.000001.232002.00000ok2.410000.0000082.000002.510001
10/21/2022 4:11:55 PM,10.77000,9.80000,1.51000,1.51000,0.21600,1,ok,,,,,11/3/2022 4:37:13 PM550.000001.230002.00000ok2.400000.0000084.000002.510001
10/21/2022 4:12:10 PM,10.74000,9.80000,1.52000,1.52000,0.21500,1,ok,,,,,11/3/2022 4:31:48 PM550.000001.229002.00000ok2.400000.0000084.000002.500001
10/21/2022 4:12:24 PM,10.75000,9.80000,1.52000,1.51000,0.21500,1,ok,,,,,11/3/2022 4:24:23 PM550.000001.228002.00000ok2.390000.0000084.000002.500001
10/21/2022 4:12:41 PM,10.75000,9.80000,1.51000,1.50000,0.21500,1,ok,,,,,11/3/2022 4:18:59 PM550.000001.232002.00000ok2.400000.0000083.000002.510001
10/21/2022 4:12:55 PM,10.74000,9.80000,1.52000,1.52000,0.21500,1,ok,,,,,11/3/2022 4:12:14 PM550.000001.228002.00000ok2.410000.0000084.000002.510001
```

Abbildung 15: Strings die Werte der letzten 10 Spalten beinhalten aufgrund von fehlenden Kommas

Es wurde versucht den String in 10 verschiedene Strings zu trennen, aber dies ist nicht gelungen, weil die einzelnen Strings nicht immer die gleiche Anzahl Charaktere hatten. Das Datum und der Status variierten sehr stark in der Länge und somit wurde die Trennung vom String aufgrund von zu grossem Aufwand nicht weiterverfolgt.

Beim Support wurde auch die BOM (Bill of Materials) exportiert, die eine Batchnummer vom Lieferanten haben, damit diese zu den Rohmaterialien aus dem SAP zugeordnet werden können (Abbildung 16). Es wurden auch jährlich ab 2016 «Production Orders» exportiert wo insgesamt 27 Millionen Einträge hatten und 11 Millionen davon beinhalteten Materialien aus dem SAP wo für die Varistoren verwendet werden (Abbildung 17). Mit den «Production Orders» ist es möglich diese mit dem MOPA-Viewer über die Spalte «Pulver_Charge_Nr» zuzuteilen. Aufgrund der Grösse der Daten, musste ein Fabrikstopp gemacht werden, damit alle «Production Orders» exportiert werden konnten.

B	C	D	E	F	G	H	I	J	K
LastUpdateTs	Item	MaterialItemId	Description	Position	ActualQuantity	NetQuantity	QuantityFormula	Warehouse	WarehouseElement
2016-04-20 09:53:30.660	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-20 09:53:30.660	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-04-21 13:08:31.950	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-21 13:08:31.950	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-04-25 13:58:48.570	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-25 13:58:48.570	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-04-26 13:01:30.610	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-26 13:01:30.610	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-04-27 16:02:06.100	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-27 16:02:06.100	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-04-28 20:06:52.550	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-04-28 20:06:52.550	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-05-02 11:55:14.720	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-05-02 11:55:14.720	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL
2016-05-03 13:33:38.620	HATW434284P0001	83	Zinkoxid ZnO	140	1500.000	1500.000	KG	L112	NULL
2016-05-03 13:33:38.620	HATW434284P0001	83	Zinkoxid ZnO	140	0.001	0.001	KG	L112	NULL

Abbildung 16: Ausschnitt Bill of Materials (BOM) welches nach Zinkoxid ZnO gefiltert wurde. Diese werden benötigt, um den Zusammenhang mit den Daten der Rohmaterialien zu machen.

	A	B	C	D	E
1	1/10/23 7:25	1000101407	1000148853	HAAR401506P0017	METALOXIDE RESISTOR MAE5HXXGGA
2	1/10/23 7:25	1000101407	1000148853	HATW434284P0026	ALUMINIUMDRAHT KALIBER 14
3	1/10/23 7:25	1000101407	1000148853	HAAR401506P0080	MO-WIDERSTAND MAE5HXXGGA
4	1/10/23 7:25	1000101407	1000148854	HAAR401506P0017	METALOXIDE RESISTOR MAE5HXXGGA
5	1/10/23 7:25	1000101407	1000148854	HATW434284P0026	ALUMINIUMDRAHT KALIBER 14
6	1/10/23 7:25	1000101407	1000148854	HAAR401506P0080	MO-WIDERSTAND MAE5HXXGGA
7	1/10/23 7:25	1000101407	1000148855	HAAR401506P0017	METALOXIDE RESISTOR MAE5HXXGGA
8	1/10/23 7:25	1000101407	1000148855	HATW434284P0026	ALUMINIUMDRAHT KALIBER 14
9	1/10/23 7:25	1000101407	1000148855	HAAR401506P0080	MO-WIDERSTAND MAE5HXXGGA
10	1/10/23 7:25	1000101407	1000148856	HAAR401506P0017	METALOXIDE RESISTOR MAE5HXXGGA
11	1/10/23 7:25	1000101407	1000148856	HATW434284P0026	ALUMINIUMDRAHT KALIBER 14
12	1/10/23 7:25	1000101407	1000148856	HAAR401506P0080	MO-WIDERSTAND MAE5HXXGGA
13	1/10/23 7:25	1000101407	1000148857	HAAR401506P0017	METALOXIDE RESISTOR MAE5HXXGGA
14	1/10/23 7:25	1000101407	1000148857	HATW434284P0026	ALUMINIUMDRAHT KALIBER 14

Abbildung 17: Ausschnitt Production Orders die benötigt werden um den Zusammenhang mit dem MOPA-Viewer zu erstellen

Weil aber die Schnittstelle zwischen dem BOM und den «Production Orders» nicht zur Verfügung gestellt wurde, konnten die SAP-Daten mit den Rohmaterialien im Rahmen von diesem Projekt nicht weiterverfolgt werden.

3.3.3 Legende

Um eine Übersicht der Daten im MOPA-Viewer zu bekommen, wurde mit Hilfe von mehreren Experten in der Firma und einer vorhanden Legende von einer SPS-Steuerung eine neue Legende erstellt, die alle Spalten beschreibt, soweit dies möglich ist (Abbildung 18).

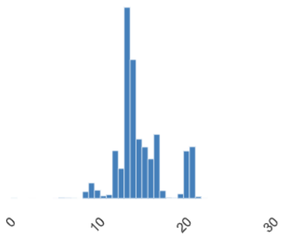
Spaltenname	Beschreibung	Einheit	Datentyp	Datenwerte
FAUF_2_Nr	Fertigungsauftrag		Integer	
MO_Charge_Nr	Metalloxid-Batchnummer/Nummer der Charge bzw. Los		Integer	
MO_Charge_FERT	Fertigungsvariante für das Los (N=Normal/V=Versuch)		String	N/V
MO_Charge_Segment	Segmentierung des Los beim Stillstand vom Sinterofen		Integer	1
MO_Charge_Lage	Lage im Ofen 1=Oben/2=Mitte/3=Unten A=aging N=keine Trennung → alle zusammen		String	1/2/3/A/N
Pulver_Charge_Nr	Batchnummer für Pulver		String	
Pruefplan_Nr	«Material_Nr»_Prod: Name der Prüfpläne für verschiedene Materialnummern (stimmt immer mit der Materialnummer überein)		String	
Pruefplan_Rev	Revisionsnummer von «Pruefplan_Nr» (jede Prüfplannummer kann mehrere Revisionsnummern bzw. verschiedene Tests haben)		Integer	1-10
Material_Nr	SAP-Materialnummer		String	
Material_Bezeichung	Produktbezeichnung: MO-WIDERSTAND «Var_Typ» (stimmt meistens mit dem Varistor-Typ überein (99.6%))		String	
Var_Typ	$Mb_1b_2z_1Hz_2z_3b_3b_4b_5(b_6)$ (z_i = Zahlen, b_i = Buchstaben) - M = Metalloxid - b_1 = Formgebung (A=Vollzylinder, E=Hohlzylinder, G=Rechteck, L=Monoblock) - b_2 = Prüfaufwand (A=Voll, B=Stichprobe, E=Reduziert, F=Energieäquivalent, H=erweiterte voll) - z_1 = Querschnitt (1=Rechteck 9.5x32.6, 3=38, 4=42, 5=47, 6=62, 7=75, 8=85, 9=108) - Hz_2z_3 = Höhe[mm] (H01 = 1.4, H46 = bis 46, HXX=geschnittene Varistoren) - b_3 = Zusammensetzung (Z = Z2, G = G9, E = E1) - b_4 = Beschichtung der Mantelfläche (S = Silikon auf Glas, G = Glas, L = Lack, N = keine) - b_5 = Beschichtung der Stirnfläche (A=Vollflächig mit Aluminium, B=Aluminium mit Rand, M=Vollflächig mit Messing, N=Messing mit Rand - b_6 = Up-Feldstärkenvariante (keine=standardmässig, P=erhöht)		String	

Abbildung 18: Ausschnitt der Legende vom MOPA-Viewer aus dem MES

Zusätzlich wurde mit der Python-Bibliothek «ydata_profiling» und der Funktion «ProfileReport» eine Übersicht über alle Spalten mit allen Eigenschaften wie Durchschnitt, Standardabweichung, Median, etc. erstellt (Abbildung 19).

Up_Rest_Ist
Real number (ℝ)

Distinct	1801	**Minimum**	0
Distinct (%)	< 0.1%	**Maximum**	34.23
Missing	8236	**Zeros**	1193
Missing (%)	0.2%	**Zeros (%)**	< 0.1%
Infinite	0	**Negative**	0
Infinite (%)	0.0%	**Negative (%)**	0.0%
Mean	14.8367697	**Memory size**	56.7 MiB



More details

Statistics | Histogram | Common values | Extreme values

Quantile statistics

Minimum	0
5-th percentile	11.78
Q1	13.25
median	13.94
Q3	16.34
95-th percentile	20.65
Maximum	34.23
Range	34.23
Interquartile range (IQR)	3.09

Descriptive statistics

Standard deviation	2.791499965
Coefficient of variation (CV)	0.1881474217
Kurtosis	0.6521694842
Mean	14.8367697
Median Absolute Deviation (MAD)	1.04
Skewness	0.6305675375
Sum	55036923.01
Variance	7.792472056
Monotonicity	Not monotonic

Abbildung 19: Statistische Informationen der Spalte «Up_Rest_Ist» aus dem MOPA-Viewer vom Profile Report die mit der Python-Bibliothek «ydata_profiling» erstellt wurde

3.3.4 Datensatz

Der Datensatz mit den Testdaten wurde erstellt, indem die Exports aus dem MOPA-Viewer und dem Export vom Support in Python mit einem Left-Join auf der Spalte «Var_ID» kombiniert wurde. Der Datensatz beinhaltete insgesamt 63 Spalten, wovon aber 14 keine Werte haben, ein Grossteil Messungen von Tests sind, die nicht für die Klassifizierung der Varistoren wichtig sind und einige die Prüfungen selber beschreiben. Schlussendlich wurde Das Datenset auf 14 Spalten reduziert, wovon nur fünf Messungen von Tests beinhalten.

In dem Datenset befinden sich auch Einträge über sogenannte «geschnittene Varistoren», wo von einem bisher getesteten Varistor abgeschnitten wurden (Abbildung 20). Diese Varistoren hatten in der Spalte «Var_Typ» eine Höhenangabe von «XX» und wurden aus dem Datenset entfernt. Folgende Datenpunkte wurden auch entfernt:

- Pulver_Charge_Nr = NaN
- MO_Charge_FERT = V (Versuch)
- MO_Charge_Lage = A (Aging)

G	H	I	J	K	L	M
Pruefplan_Nr	Pruefplan_Rev	Material_Nr	Material_Bezeichung	Var_Typ	Var_D	Var_H
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4
HAAR401506P0007_Prod	4	HAAR401506P0007	MO-WIDERSTAND MAE5HXXGGA	MAE5HXXGGA	47	12.4

Abbildung 20: Ausschnitt MOPA-Viewer gefiltert nach geschnittenen Varistoren (gekennzeichnet mit XX in Var_Typ)

Weil die Varistoren nicht einzeln im Factory Layer zurückverfolgt werden können, wurde die Präzision auf Batches in der Spalte «MO_Charge_Nr» reduziert wo sich daraufhin herausstellte, dass es nur 1166 verschiedene Batches gibt. Weil die Verteilung der Werte der Tests laut einem Experten Gaussverteilt sind, wurde zu jedem Test von jedem Batch ein Durchschnittswert und eine Standardabweichung zugeteilt. Der String in der Spalte «Var_Typ» beinhaltet Grundlegende Informationen über den Varistor wie Höhe, Durchmesser, Zusammensetzung und Formgebung. Aus dieser Spalte wurden die einzelnen Eigenschaften getrennt und in separaten Spalten abgelegt.

Um den Datensatz in ein «Vector Space»-Format zu transformieren, bzw. nur numerische Werte in der Tabelle haben, wurden für alle Spalten mit Klassen, Dummy-Variablen eingesetzt. Wenn z. B. eine Spalte drei Klassen hat, werden zwei Spalten erstellt mit 2 von den drei Klassen wo jeweils den Wert 1 oder 0 annehmen können. Um Korrelationen zu vermeiden, wird keine dritte Spalte erstellt, weil wenn die anderen zwei den Wert 0 annehmen, ist es klar, dass diese Spalte eine 1 hätte.

3.4 Zusammenfassung Datenaufbereitung

Um eine Übersicht über die Verfügbarkeit der Daten zu bekommen, wurde eine graphische Übersicht erstellt, wo die wichtigsten Daten auf einer Zeitachse geplottet werden (Abbildung 21). Weil viele Sensoren erst später Daten aufgenommen haben, muss ein Kompromiss zwischen der Anzahl Datenpunkte und der Anzahl Spalten mit nützlichen Informationen gemacht werden. Ein Vorschlag wie dies Umgesetzt wurde in diesem Projekt in schwarz zu sehen.

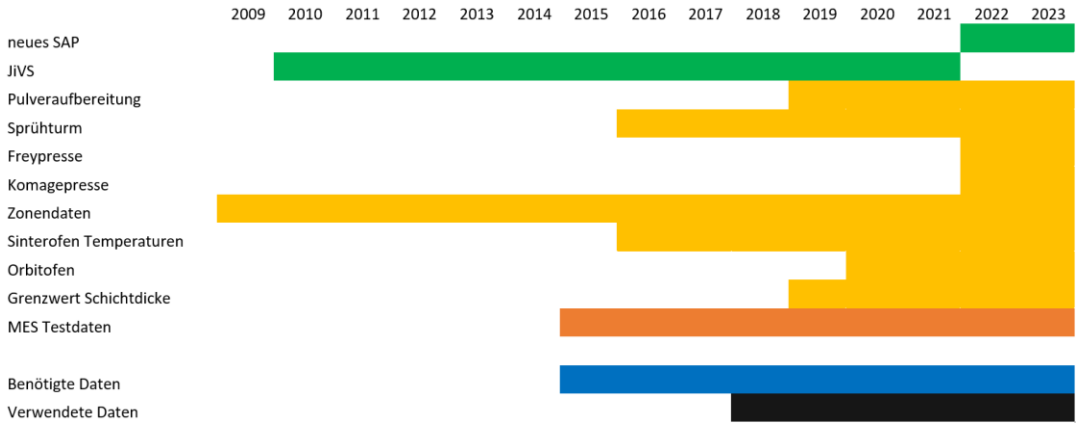


Abbildung 21: Übersicht der Verfügbarkeit der Daten der einzelnen Quellen

Weil alle Daten aus mehreren Quellen stammen und auch hier eine Übersicht nützlich wäre, wurde eine erstellt, wo die einzelnen Quellen dargestellt und die Schnittstellen beschrieben wurden. Die Verbindungen, die gestrichelt sind, stellen die Schnittstellen dar die noch nicht in der Firma integriert wurden und manuell erstellt werden mussten (Abbildung 22).

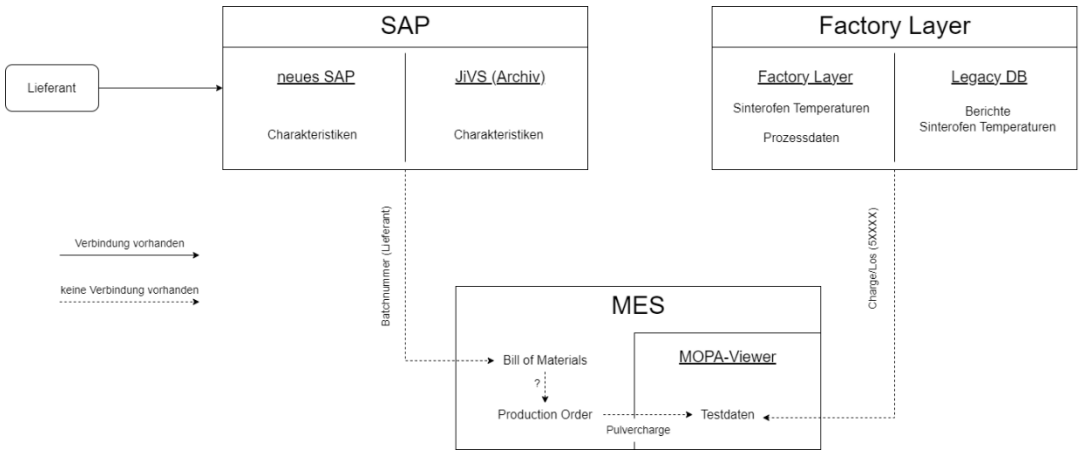


Abbildung 22: Datenfluss in der Firma von den Daten die für die Qualität der MOVs relevant sein könnten. Die Schnittstelle zwischen Bill of Materials und Production Order ist nicht zur Verfügung gestellt worden.

Mit Python-Bibliotheken konnten die Tabellen von den Analysezertifikaten nicht erkannt werden und im neuen SAP gibt es zu wenig Daten. Zudem konnten die Daten aus dem Archivierungssystem JiVS konnten nicht verwendet werden, weil die Schnittstelle zwischen den BOM und dem «Production Order» fehlt und die Daten dort auch inkomplett sind. Dies muss noch nachverfolgt werden, bevor sie für Machine Learning verwendet werden können.

Im Factory Layer gibt neben den Sinterofen Temperaturen auch andere Sensordaten, die die Qualität der Varistoren beeinflussen könnten, aber im Beispiel von den Pressen gibt es noch zu wenig Daten, weil diese erst Ende 2022 eingeführt wurden. Weil bei den Sinterofen Temperaturen es keine Angabe gibt in welcher Zone sich die Laufende Nummern befinden und die Rückverfolgbarkeit nur auf Präzision von ganzen Batches gemacht werden kann, wurden Berichte aus einer älteren Datenbank verwendet um den minimalen Wert, den Durchschnitt und den maximalen Wert jeder Zone von jedem Batch zu bestimmen. Die Daten aus der Legacy Datenbank befanden sich schon in einem «Vector Space»-Format und mussten somit nicht weiter behandelt werden.

Die Tabelle aus dem MES musste anhand von zwei verschiedenen Exports erstellt werden, aber enthält sonst keine weiteren Probleme. Die Varistoren mussten jedoch in Batches gruppiert werden, damit diese mit den Daten aus dem Factory Layer kombiniert werden konnten. Weil es fünf Test gibt, die zuständig sind für die Klassifizierung der Varistoren, wurden fünf Datensätze (5 Datensätze x 933 Zeilen x 67 Spalten) erstellt wovon jeweils zwei Spalten die Beschriftungen der Datenpunkte sind. Mit dem Durchschnitt und der Standardabweichung lässt sich eine Gaussglocke plotten und später auch vorhersehen.

3.5 Verbesserungsmöglichkeiten

Um zukünftig den Umgang mit den Daten zu erleichtern können einige Massnahmen eingeführt werden. Hier werden Empfehlungen aufgelistet, die umgesetzt werden könnten:

- Eine Datenbank, die für alle Qualitätsdaten von den Varistoren gewidmet ist mit aktiven Datapipelines, wo andauernd Abfragen an den anderen Datenbanken machen. Somit wird viel Aufwand vom Sammeln der Daten erspart, weil sonst immer eine Person vom Support oder externen Firma die Daten zur Verfügung stellen müsste.
- Einheiten wie [kV] oder [%] immer im Spaltennamen einbegreifen damit nicht jedes Mal ein Experte dies bestätigen muss. Dies wurde vor allem im MOPA-Viewer festgestellt, wo zahlreiche Tests durchgeführt werden und viele verschiedene Einheiten verwendet werden.

- Bei den Tests im MOPA-Viewer wurden Einträge von Tests erstellt, wo aber nur den Wert «0» annehmen können, was darauf hinweist, dass kein wirklicher Test stattfindet. Um Missverständnisse zu vermeiden ist es zu empfehlen die Einträge leer zu lassen bzw. als NaN anzugeben.
- Um ein schnelleres Verständnis von den Daten zu bekommen, soll immer eine Legende mitgeführt werden, welche bei Änderungen direkt angepasst werden soll. In diesem Projekt wurde eine Legende für den MOPA-Viewer und eine für die Sensordaten im Factory Layer erstellt, aber diese sollten von Experten noch zuerst validiert und ggf. erweitert werden.
- Die Rückverfolgbarkeit im Factory Layer verbessern. Ein Beispiel könnte sein die einzelnen Varistoren im MES MOPA-Viewer den Laufenden Nummern im Sinterofen zuordnen zu können oder sogar den Varistoren von Anfang an eine ID zuteilen und in jedem Prozessschritt diese mitführen. Dies gilt auch für die anderen Anlagen.
- Ein Experte hat noch erwähnt, dass neben dem Rezept, die Temperatur im Ofen und die Zeit im Ofen, die Atmosphäre auch eine wichtige Rolle spielen würde. Hierzu werden noch keine Daten aufgenommen, die Einfluss auf die Performanz der Machine Learning Modelle haben könnten. Die Sensoren können, nachdem sie installiert wurden, die Daten an die Factory Layer Datenbank senden.

4 Datenanalyse

In diesem Kapitel wird beschrieben, was für Machine Learning Modelle verwendet wurden. Zuerst wurden mehrere Deep Learning Modelle erstellt und später wurden auch noch Algorithmen, die auf Decision Trees basieren verwendet. Die Genauigkeit der Vorhersagen wird hier auch erwähnt.

4.1 Deep Learning Modell mit nur den MES-Daten

Weil zum Zeitpunkt vom Erstellen dieses Modells die Factory Layer Daten noch nicht bereit waren, wurden nur die MES-Daten aus dem MOPA-Viewer verwendet, um dieses Deep Learning Modell (2.3 Artificial Neural Network) zu trainieren. Aus diesem Grund wurde entschieden zwei verschiedene Modelle mit diesem Datensatz zu trainieren, um zu vergleichen, welche bessere Vorhersagen machen wird. Im ersten Datensatz werden die einzelnen Einträge nicht kombiniert und der Datensatz hatte somit 3.8 Millionen Datenpunkte. Mit dieser Methode konnten mehrere Spalten beachtet werden wie «MO_Charge_Lage», weil diese innerhalb von einem Batch gemischt sein kann. Im zweiten Datensatz wurden die kombinierten Batches mit dem Durchschnitt und Standardabweichung verwendet, um das Modell zu trainieren. Dies wurde so gemacht, weil die Verteilungen der Durchbruchsspannungen gaussförmig sind. Mit diesem Vorgehen wurde der Datensatz von 3.8 Millionen Datenpunkten auf etwa 1000 Datenpunkte reduziert.

Alle Machine Learning Modelle in diesem Projekt wurden in Python erstellt und für die Deep Learning Modelle wurde die PyTorch-Bibliothek verwendet. Der Datensatz wurde jeweils in ein Trainingsset und ein Testset aufgeteilt, damit am Schluss das Modell validiert werden kann. Die Daten wurden zudem noch mit dem z-Scaler skaliert für bessere Effizienz beim Trainieren vom Modell.

Die erste Architektur bestand aus einem Input Layer, der sich dynamisch an die Anzahl Spalten aus dem Datenset anpasst, drei Hidden Layers mit 128, 64 und 32 Neuronen und einem Output Layer mit nur einem Neuron, weil ein Fliesswert vorhergesagt werden soll. Alle Layer haben eine ReLU-Aktivierungsfunktion die Werte unterhalb von null auf null setzt und alle positiven Werte beibehaltet (Abbildung 23). Für das Kriterium vom Verlust wurde der MSE (Mean Squared Error) und als Optimierer ADAM (adaptive moment estimation) verwendet, der sich selbst dynamisch anpasst, um die bestmögliche Lernrate zu verwenden. Die Anzahl Epochen bzw. die Anzahl wie oft der Datensatz verwendet wird, um das Modell zu trainieren, wurde auf zehn begrenzt und nach jeder Epoche wurde der Verlust herausgegeben. Am Schluss vom Training wurde mit dem Testset der Verlust, der MAE (Mean Absolute Error) und RMAE (Relative Mean Absolute Error) berechnet. Als Resultat wurde für die Referenzspannung vom Wechselstrom (Uref_AC) ein RMAE zwischen 12% und 15% erreicht (Abbildung 24).

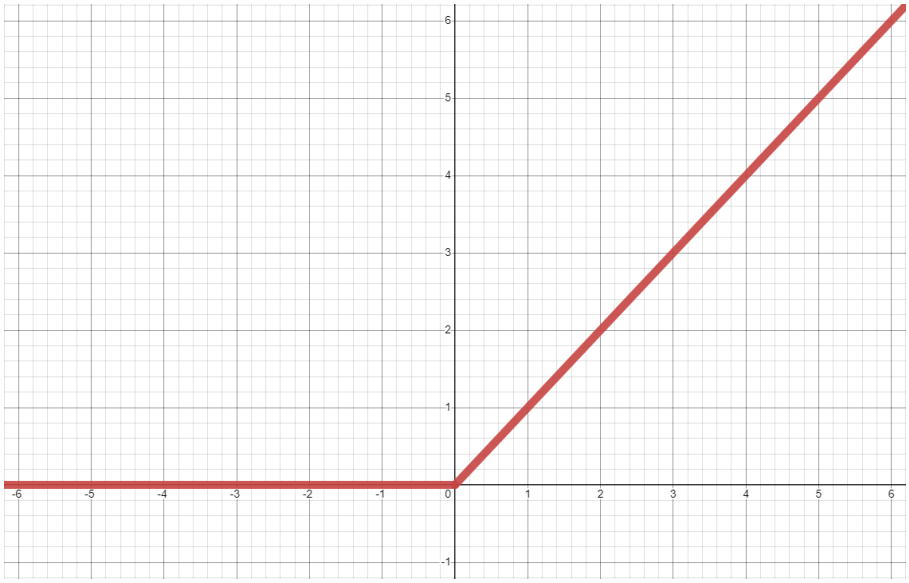


Abbildung 23: ReLU (rectified linear unit) als Aktivierungsfunktion

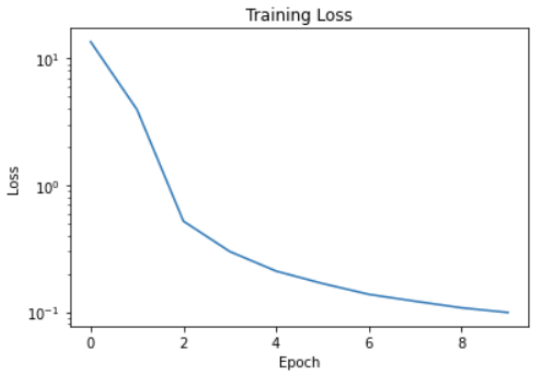
```
train_model(uref_ac)
```

```
Epoch [1/10], Loss: 1.0050
Epoch [2/10], Loss: 0.9844
Epoch [3/10], Loss: 0.9662
Epoch [4/10], Loss: 0.9491
Epoch [5/10], Loss: 0.9322
Epoch [6/10], Loss: 0.9158
Epoch [7/10], Loss: 0.8989
Epoch [8/10], Loss: 0.8810
Epoch [9/10], Loss: 0.8621
Epoch [10/10], Loss: 0.8420
Test Loss: 0.8168, MAE: 0.7374, RMAE: 0.1356
```

Abbildung 24: Resultate vom Deep Learning Modell mit individuellen Varistoren. Die Abweichung vom eigentlichen Wert beträgt im Durchschnitt 13.6%

In der zweiten Architektur wurden nur sieben Neuronen im Input Layer verwendet, 32 Neuronen im Hidden Layer und zwei im Output Layer. Im Output Layer wird in einem Neuron der Durchschnitt und im anderen die Standardabweichung einer Gaussverteilung vorhergesehen. Auch in diesem Modell wurde der MSE für das Verlustkriterium und ADAM für den Optimierer verwendet. Die Anzahl Epochen wurden hier auch auf zehn beschränkt, aber diese wurden anschliessend geplottet auf einer logarithmisch y-Achse, um zu erkennen ab wann das Modell sich nicht mehr verbessert. Obwohl dieses Modell weniger Inputs und eine simplere Architektur hatte, konnte das Modell genauere Vorhersagen machen als das erste, weil die RMAE vom Uref_AC zwischen 6% und 12% lag. Um die Resultate visuell zu veranschaulichen, wurde ein Skript erstell, wo die Gausskurven von der vorhergesagten und der eigentlichen Werten plottet (Abbildung 25).

```
Epoch [1/10], Average Loss: 13.5492
Epoch [2/10], Average Loss: 3.9469
Epoch [3/10], Average Loss: 0.5206
Epoch [4/10], Average Loss: 0.2999
Epoch [5/10], Average Loss: 0.2101
Epoch [6/10], Average Loss: 0.1677
Epoch [7/10], Average Loss: 0.1377
Epoch [8/10], Average Loss: 0.1214
Epoch [9/10], Average Loss: 0.1079
Epoch [10/10], Average Loss: 0.0990
```




```
Relative Absolute Difference Mean: 0.0615648229280164
Relative Absolute Difference Std: 0.6212748750110603
```

Abbildung 25: Resultate vom Deep Learning Modell mit Batches als Datenpunkten. Die Abweichung vom eigentlichen Wert beträgt im Durchschnitt 6.1% für den Durchschnitt der Gaussverteilung

4.2 Deep Learning Modell mit Factory Layer Daten

Als die Daten aus den Berichten von den Sinterofen Temperaturen zur Verfügung gestanden sind, wurde ein Ablauf erstellt, wie das Deep Learning Modell vorgehen soll (8.4 Ablauf Deep Learning Funktion). In diesem Ablauf wurde beschrieben welche Methoden verwendet und wie die Hyperparameter ausgewählt werden, um das Deep Learning Modell zu trainieren. Folgende Hyperparameter werden mit Grid Search gesucht:

- Anzahl Layers
- Anzahl Neuronen in jedem Layer
- Aktivierungsfunktion in jedem Layer
- Parameter für die L2 Regularisierung (die L2 Regularisierung oder auch Ridge Regression genannt bestraft eine hohe Anzahl von Inputs und senkt die Gewichte von weniger relevanten Inputs gegen null → vermeidet Overfitting)
- Wahrscheinlichkeit für den Dropout (Dropout blockiert einige Neuronen in einigen Zyklen, damit das Modell robustere Eigenschaften der Daten erkennt → vermeidet Overfitting)

Grid Search ist eine Methode, in der alle möglichen Parameterkombinationen ausprobiert werden, um die beste Parameterkombination zu bestimmen. Um den Trainingsprozess noch effizienter zu machen, soll noch zwischendurch ein Normalisierungslayer hinzugefügt werden und damit der Prozess nicht zu lange dauert wurde anstelle einer fixer Anzahl Epochen, Early Stopping verwendet welches, wenn nach einer bestimmter Anzahl Epochen keine Verbesserungen stattgefunden hat, den Prozess beendet. Der Datensatz hat 60 Inputs und zwei Werte, die vorhergesehen werden sollen. Nachdem der Datensatz randomisiert und skaliert wird, wird dieser in ein Trainingsset und Testset aufgeteilt. Am Trainingsset soll dann k-Fold Cross Validation angewandt werden, wo das Trainingsset in mehrere kleineren Sets aufgeteilt wird. In einem Beispiel von acht Folds, werden im ersten Durchlauf die ersten sieben für das Training vom Modell verwendet und mit dem achten wird das Modell validiert, um die Hyperparameter zu bestimmen. Im zweiten Durchlauf wird das zweiletzte Fold verwendet für

die Validierung und dieser Vorgang wird für alle Folds wiederholt. Dies hat gegenüber einem gewöhnlichen Validierungsset den Vorteil, dass mehr Daten für das Training verwendet werden können und dass der Verlust besser bestimmt werden kann, weil der Durchschnitt von jedem Fold berechnet wird. Zum Schluss soll noch mit dem Testset das beste Modell getestet und der Verlust vom Testset und vom Validierungsset soll über alle Epochen geplottet werden.

Dieser Ablauf bringt aber seine eigenen Probleme mit sich wie z. B., dass der Rechenaufwand vom Grid Search und k-Fold Cross Validation sehr gross ist. Als ein Skript erstellt wurde, um alle möglichen Kombinationen an Parametern zu erstellen wurde festgestellt, dass es über 183 Millionen mögliche Kombinationen an Parametern gibt und das, ohne die Aktivierungsfunktion und Normalisierungslayer zu beachten. Angenommen es wird eine Sekunde benötigt, um ein Modell zu trainieren, wird es sechs Jahre brauchen, um alle Kombinationen zu trainieren. Aus diesem Grund wurde der Parameterraum deutlich verkleinert, für alle Layers wurde die ReLU-Aktivierungsfunktion verwendet, der Normalisierungslayer wurde weggelassen und es wurde ein normales Validierungsset verwendet anstelle vom k-Fold Cross Validation. Somit wurde die Anzahl verschiedener Kombinationen auf 1440 reduziert. Danach wurde eine Funktion erstellt, welche eine Kombination von Parameter als Argument aufnimmt und die Architektur vom Deep Learning Modell dynamisch verändert. Die Resultate werden mit den verwendeten Parametern in ein Pandas DataFrame gespeichert, welches später wieder aufrufbar ist.

Das Trainieren von allen Kombinationen dauerte auf einem High Performance Laptop 40 Minuten und das Modell mit dem besten RMAE war 2.8% (Abbildung 26). Es ist aufgefallen, dass in den besseren Modellen ein vorheriger Layer weniger Neuronen hat als in den späterem. Der Grund dafür kann sein, weil aus den Berichten für jede Zone drei Werte vorhanden sind, der minimale, der Durchschnitt und der maximale Wert und aus diesem Grund 2 davon auf Grund von Korrelationen redundant sein könnten. Die Anzahl Epochen schwanken grundsätzlich zwischen 25 und 60 und die Dropout-Wahrscheinlichkeit war in den besten Modellen meistens auf 0.1. Es ist hier zu erwähnen, dass die Wahrscheinlichkeit von null nicht beachtet wurde.

	hidden_neurons	l2_reg	dropout_prob	early_stopping	last_epoch	last_epoch_loss	test_loss	rad_mean	rad_std
572	[16, 16, 64, 64]	0.0100	0.1	5	26	0.424188	0.002482	0.028	0.310
1312	[64, 32, 64, 32]	0.0010	0.1	5	35	0.419960	0.007882	0.030	1.000
1216	[64, 16, 64, 64]	0.0010	0.1	5	24	0.337920	0.007853	0.033	1.000
1068	[32, 64, 32, 64]	0.0001	0.1	5	26	0.394502	0.008597	0.034	1.000
1032	[32, 64, 16, 64]	0.0001	0.1	5	60	0.187147	0.007626	0.034	1.000
1208	[64, 16, 64, 32]	0.0100	0.1	5	36	0.548790	0.004128	0.035	0.374
68	[16, 64]	0.0100	0.1	5	50	0.441303	0.014354	0.035	0.276
856	[32, 16, 32, 64]	0.0010	0.1	5	26	0.373378	0.004238	0.035	0.322
346	[32, 64, 32]	0.0100	0.3	5	46	0.933941	0.006573	0.036	0.294
156	[16, 16, 32]	0.0001	0.1	5	66	0.273603	0.011351	0.036	1.000

Abbildung 26: 10 beste Resultate aller 1440 Kombinationen. Die Hyperparameter und Resultate werden von jedem Modell einzeln gespeichert.

4.3 Deep Learning Modell mit komplexerer Architektur

Anhand Empfehlungen von einem Experten wurde die Architektur angepasst, indem in jedem Layer 128 Neuronen verwendet werden, das Dropout und Early Stopping entfernt wird und der L2-Parameter tiefer gestellt wird. Weil es dadurch nur wenig verschiedene Kombinationen gibt, konnte die Anzahl Layers erhöht werden. Nachdem mehrere Modelle trainiert wurden, konnte ein RMAE von 1.9% erreicht werden. Die Modelle wurden alle für 100 Epochen trainiert

und der RMAE lag immer zwischen 1.9% und 5.3%. Das heisst, dass der Parameterraum vom Grid Search im vorherigen Unterkapitel zu klein war, um das optimale Modell zu erstellen.

4.4 Decision Tree Algorithmen

4.4.1 Random Forest

In Python wurde mit der sklearn-Bibliothek ein Random Forest (2.4.1 Random Forest) mit 100 Decision Trees erstellt und für mehrere Male trainiert. Als Resultat wurde ein RMAE von fast 1% erreicht (Abbildung 27) und die Gewichte der Variablen konnten herausgegeben werden (Abbildung 28).

```
Mean squared error of mean: 0.014935518972824035
Mean squared error of std: 0.0015464894298661497
Average relative absolute difference of mean: 0.011127041783607877
Average relative absolute difference of std: 0.17780453658511813
```

Abbildung 27: Resultate vom Random Forest mit einer durchschnittlichen relativen Abweichung von 1.11% vom Durchschnitt der Gausskurve und 17.7% von der Standardabweichung

```
T_MAX_SZ8: 0.0034201744093584094
T_MIN_SZ9: 0.0015017793472160423
T_AVG_SZ9: 0.014078315378751868
T_MAX_SZ9: 0.025130815033663947
T_MIN_SZ10: 0.0014391928876390742
T_AVG_SZ10: 0.018531319612749578
T_MAX_SZ10: 0.04079707910568173
T_MIN_SZ11: 0.0030131340383343108
T_AVG_SZ11: 0.0319850182487018
T_MAX_SZ11: 0.02203454814103539
T_AVG_SZ12: 0.011252841504529614
T_MIN_SZ13: 0.009784144581812561
T_AVG_SZ13: 0.028639098367905687
T_MAX_SZ13: 0.03536807339961846
T_MIN_SZ14: 0.0019063141440340938
T_AVG_SZ14: 0.015335375975551692
T_MAX_SZ14: 0.01425638294407027
T_MIN_SZ15: 0.004962350503004384
T_AVG_SZ15: 0.02071925152054025
T_MAX_SZ15: 0.02898976737676119
T_MIN_SZ16: 0.0014096184166086617
T_AVG_SZ16: 0.01840361275881505
T_MAX_SZ16: 0.02268980252130992
H_Zusammensetzung: 0.0025326413294860938
Var_D: 0.012576428986544465
Var_H: 0.600618838902247
```

Abbildung 28: Feature Importance bzw. Gewichte der einzelnen Variablen mit einem Gewicht von über 0.1%. Alle Gewichte zusammen ergeben 1 und je höher der Wert, desto grösseres Gewicht hat die Variabel.

4.4.2 Gradient Boosting

Auch hier wurde ein Modell mit der sklearn-Bibliothek erstellt und ein RMAE von fast 1% erreicht (Abbildung 29). Hier gibt es jedoch eine grössere Anzahl an relevanten Variablen, weil mehrere Verlust-Funktionen verwendet werden, um das Modell zu trainieren (2.4.2 Gradient Boosting).

```
Mean squared error of mean: 0.020371220662736986
Average relative absolute difference of mean: 0.01253405120778817
```

Abbildung 29: Resultate vom Gradient Boosting. Die durchschnittliche relative Abweichung beträgt 1.25%

Eine neue berühmte Bibliothek für Gradient Boosting ist XGBoost, weil die Vorteile gegenüber der von sklearn hat wie Regularisierung, Geschwindigkeit, Skalierbarkeit und noch einige mehr. Ein erster Durchlauf hat jedoch ein schlechteres RMAE von 1.2% ergeben als das von sklearn (Abbildung 30).

```
Mean squared error of mean: 0.020791519137341136
Average relative absolute difference of mean: 0.012491300845363791
```

Abbildung 30: Resultat von der Gradient Boosting Bibliothek XGBoost mit einer durchschnittlichen relativen Abweichung von 1.25%

5 Resultate

5.1 Datenaufbereitung

Weil die Daten, die für die Qualitätsvorhersagen aus mehreren Quellen kommen, die alle ihre eigenen Fehler und Probleme mit sich bringen, musste ein Grossteil der Zeit vom Projekt für die Datenaufbereitung investiert werden.

Die Daten aus dem SAP bzw. im JiVS enthalten mehrere Einträge von einem Material und die Charakteristiken konnten nicht alle auf einmal eingesehen werden. Aus diesem Grund wurde ein Python-Skript erstellt, welches diese kombiniert und pro Material ein einziges File mit allen Einträgen erstellt. Hier wurde jedoch festgestellt, dass die Daten inkomplett sind und diese zuerst noch alle aufgefunden werden müssen.

Im Factory Layer befindet sich eine grosse Menge an Daten wovon die Sinterofen Temperaturen allein einen Grossteil ausmachen. Damit ein Datensatz mit den Temperaturen erstellt werden kann, wurde erklärt und bewiesen, wieso fünf der sechs Spalten mit Temperaturen entfernt werden können und wie die Datentypen von den Spalten anzupassen sind. Neben den Sinterofen Temperaturen gibt es noch andere Prozessdaten von anderen Anlagen, die aber mehrere Tabellen benötigen, um sinnvolle Analysen zu machen. Auch hier wurde ein Python-Skript wurde erstellt, welches alle Tabellen aufnimmt und jeweils eine neue Tabelle pro Anlage herausgibt. Weil die Sinterofen Temperaturen schwer in einzelne Zonen vom Sinterofen einzuteilen sind und die restlichen Prozessdaten nicht genug Daten bieten, wurden diese im Umfang von diesem Projekt nicht für die Datenanalyse verwendet. Anstellt der Sinterofen Temperaturen, wurden Berichte über die einzelnen Zonen von einer älteren Datenbank verwendet wo aber auch in mehrere Tabellen aufgeteilt wurden und somit zuerst kombiniert werden mussten. Diese Berichte enthalten den minimalen und maximalen Temperaturwert sowie den Durchschnitt jeder Zone von jedem Batch. Die Schubluft und der Ofendruck sind auch vorhanden, aber enthalten viele Lücken in den Daten.

Aus dem MES wurden Testdaten aus mehreren Quellen exportiert und kombiniert, weil einige Fehler bei den Exports entstanden sind. Viele Spalten im Datensatz enthalten keine wertvollen Informationen oder sind sogar ganz leer und wurden entfernt. Der Datensatz wurde zudem noch auf Präzision von Batches reduziert, weil es keine Rückverfolgbarkeit auf einzelne Varistoren im Factory Layer oder SAP gibt.

Schlussendlich wurden nur die Daten aus dem Factory Layer mit den MES-Daten kombiniert und für alle Tests, die eine Rolle für die Klassifizierung spielen, wurde ein separater Datensatz erstellt. Insgesamt gab es fünf Datensätze mit 933 Zeilen und 67 Spalten, wovon die letzten zwei Spalten immer die sogenannten Beschriftungen der Datenpunkte sind. Ein Wert stellt den Durchschnitt dar und der andere die Standardabweichung. Zusammen kann eine Gaussverteilung gebildet werden, die aus den einzelnen Varistoren vor der Gruppierung abgeschätzt wurde.

Weil die Spaltenbezeichnungen die Information in den Spalten ausführlich bezeichnen, wurde keine Legende vom Datensatz erstellt. Alle Anomalien und NULL-Werte wurden entfernt und die fünf Datensätze haben alle eine Grösse von 117kB als PARQUET-Files.

5.2 Datenanalyse

Zuerst wurden zwei Deep Learning Modelle erstellt die jeweils mit nur den Daten aus dem MES, Vorhersagen machen sollen, weil zu dem Zeitpunkt die Daten aus dem Factory Layer nicht bereit waren. Das erste Modell verwendete die individuellen Datenpunkte, die noch nicht gruppiert wurden und das zweite verwendete die gruppierten Daten, wo eine Gaussverteilung vorhergesagt werden soll. Obwohl es beim ersten Modell mehrere Variablen im Datensatz

hatte, lieferte das Modell mit den gruppierten Daten ein besseres Resultat und hatte eine RMAE zwischen 6% und 12%. Hier wurde direkt die Abweichung von den gemessenen Tests berechnet, anstelle der Klassifizierung, weil die Varistoren sowieso anhand der Tests klassifiziert werden.

Als die Datensätze mit den Factory Layer Daten bereit waren, wurde eine Python-Funktion erstellt, welches ein Deep Learning Modell trainiert anhand von gegebenen Hyperparametern wie z. B. die Anzahl Neuronen oder Anzahl Layers in der Architektur. Zudem wurde eine Funktion erstellt, die anhand von gegebenen Listen mit möglichen Hyperparametern eine Liste mit allen verschiedenen Kombinationen erstellt. Weil die Anzahl Kombinationen sehr gross war, wurde der Parameterraum stark reduziert, bevor die eigentlichen Modelle trainiert wurden. Aus allen Modellen hatte das beste Modell ein RMAE von 2.8%, was schon drei Mal besser ist, als wenn nur die MES-Daten verwendet wurden.

Mit weiteren Vorschlägen von einem Experten wurden Änderungen am Modell durchgeführt, was schlussendlich zu einem komplexeren Modell geführt hat. Die Anzahl der Layers und Neuronen wurde erhöht und das Modell wurde über 100 Epochen trainiert. Es wurde mit der Anzahl Layers experimentiert und das beste Modell erreichte eine RMAE von 1.9%, was eine mehr als vierfache Verbesserung gegenüber dem Modell mit nur den MES-Daten ist.

Um noch andere Algorithmen neben Deep Learning zu probieren, wurden Modelle mit Random Forest und Gradient Boosting erstellt. Beide Modelle basieren auf Decision Trees und haben nur schon nach ersten Tests ein RMAE von fast 1% erreicht, was eine zehnfache Verbesserung gegenüber dem ersten Deep Learning Modell ist. Eine Bibliothek, die sich auf Gradient Boosting spezialisiert namens XGBoost wurde auch ausprobiert, aber erreichte nur eine RMAE von 1.2% welches leicht schlechter ist als der Random Forest und Gradient Boosting aus der sklearn-Bibliothek. Hier ist zu erwähnen, dass aus Zeitgründen nicht verschiedene Hyperparameter ausprobiert wurden und die Performanz von allen Modellen noch besser sein könnte.

6 Schlussdiskussion

Im Rahmen dieser Bachelorarbeit wurde ein Projekt durchgeführt, bei dem ein Datensatz aus mehreren Quellen zusammengesetzt und mit diesem ein Machine Learning Modell erstellt werden sollte, welches die Klassen von Metalloxid-Varistoren vorhersieht. Zu Beginn des Projekts wurde ein Fabrikbesuch und eine Intensivwoche durchgeführt, um sich mit der Situation vertraut zu machen. Daraufhin wurde ein Projektauftrag erstellt und eine Literaturrecherche durchgeführt, um die Grundbegriffe zu klären, die im Verlauf der Dokumentation verwendet werden.

Die Daten, welche Qualität der Metalloxid-Varistoren beeinflussen könnten, befanden sich in mehreren Quellen die einzeln exportiert werden mussten. Ein Teil der Rohmaterialdaten wurde als gescannte Analysezertifikate aufbewahrt und auch digital gespeichert. Die neuen Daten befinden sich in einem neuen SAP-System, welches Mitte 2022 migriert wurde und alle älteren Daten befinden sich in einem Archivierungssystem namens JiVS. Die Daten aus dem JiVS wurden analysiert und es wurde festgestellt, dass ein Grossteil der Daten nicht vorhanden ist, obwohl dieser laut Experten immer eingetragen wurde. Vom Hauptmaterial Zinkoxid gab es nur 31 von 192 Einträgen, die die Reinheit des Materials angeben, aber die Verunreinigungen von den anderen Materialien waren in noch weniger Einträgen vorhanden. Aus diesem Grund wurden die Daten von den Rohmaterialien nicht im Machine Learning Modell verwendet.

Im Factory Layer befanden sich Prozessdaten aus der Fabrik, die von einer externen Firma verwaltet und gepflegt wurden. Aus dem Factory-Layer konnten drei Arten von Daten exportiert werden: die minütlich aufgenommenen Sinterofen Temperaturen, Daten aus allen anderen Anlagen in der Fabrik und sogenannte Berichte über die Sinterofen Temperaturen. Die Daten mit den Temperaturen aus dem Sinterofen waren sehr umfangreich, und der Datensatz musste zuerst reduziert werden, damit dieser verwendet werden konnte. Da laut Experten die einzelnen Zonen im Sinterofen modelliert werden sollten und diese in der Tabelle nicht angegeben wurden, war es schwierig die Daten den einzelnen Zonen zuordnen zu können. Aus diesem Grund wurden die Berichte über die Temperaturen verwendet, welche die minimale, maximale und durchschnittliche Temperatur jeder Zone von jedem Batch beinhalteten. Die Prozessdaten aus den restlichen Anlagen wurden auch nicht beachtet, weil die wichtigsten davon erst seit kurzem eingeführt wurden und daher nicht genügend Daten vorhanden waren.

Die Testdaten des MES bzw. MOPA-Viewers enthalten Ergebnisse von mehreren Tests, von denen nur diejenigen verwendet wurden, die zur Klassifizierung der Metalloxid-Varistoren benötigt werden. Im MOPA-Viewer gibt es zwar einen Eintrag für jeden Metalloxid-Varistor, weil jeder separat geprüft wird, aber weil die Rückverfolgbarkeit nur auf ganze Batches im Factory Layer vorhanden ist, musste der Datensatz somit auf ganze Batches reduziert werden. Es wurde jedoch von jedem Batch ein Durchschnittswert und eine Standardabweichung berechnet, damit jedem Batch eine Gaussverteilung zugewiesen werden kann. Im MES gibt es zudem noch die Bill of Materials und die Production Orders, die benötigt werden, um den Zusammenhang zwischen dem SAP und MOPA-Viewer zu machen.

Um die Daten für Machine Learning zu verwenden, wurden sie aus dem Factory Layer und dem MES kombiniert und in ein Vector Space Format transformiert. Da Deep Learning für solche Datensätze gut funktioniert, wurde mit Python und PyTorch eine Funktion erstellt, die die Hyperparameter als Argumente akzeptiert und die Ergebnisse hinsichtlich Vorhersagegenauigkeit liefert. Anstatt die einzelnen Hyperparameter manuell anzupassen, wurde ein Skript geschrieben, das alle möglichen Kombinationen von Hyperparametern

generiert und das Modell damit trainiert. Der Hyperparameterraum musste jedoch stark reduziert werden, da es zu viele verschiedene Kombinationen gab.

Die ersten Resultate mit dem kompletten Datensatz ergaben, dass der Durchschnitt der Gaussverteilung eine RMAE (Relative Mean Absolute Error) von 2.8% hat. Später wurden noch einige Modelle mit komplexeren Architekturen probiert und der RMAE konnte auf 1.9% reduziert werden. Weitere bekannten Algorithmen wie Random Forest und Gradient Boosting die aus Decision Trees basieren wurden auch ausprobiert und ein RMAE von fast 1% wurde erreicht. In allen Modellen konnte die Standardabweichung jedoch nicht genau vorhergesagt werden und der RMAE von dieser liegt bei ca. 20%.

Im Nachhinein gibt es zwei Punkte, die dieses Projekt besser gemacht hätten. Der erste Punkt wäre, dass die Übersicht der Daten schon früher erstellt werden sollte. Somit konnte man schon früher feststellen, welche Daten fehlen und welche Schnittpunkte zu erstellen sind. Anhand dieser Übersicht konnte auch erkannt werden, wieso der zweite Punkt umgesetzt werden soll: Die Zeit, die für das SAP verbraucht wurde, was schlussendlich nicht verwendet werden konnte, in die Datenanalyse bzw. das Machine Learning zu investieren. Somit konnten die Hyperparameterräume vom Deep Learning, Random Forest und Gradient Boosting genauer untersucht und evtl. ein noch genaueres Resultat erzielt werden. Weil viel Zeit in das SAP investiert wurde, ist eines der Risiken im Risikomanagement eingetroffen: Die Datenaufbereitung dauert zu lange. Die Zeit, die für die Datenanalyse verwendet werden sollte, konnte nicht ganz ausgenutzt werden und schlussendlich wurden die Daten aus dem SAP sowieso nicht im Machine Learning verwendet. Es sind auch andere Risiken eingetroffen, aber diese stellten keine grossen Probleme dar und wie mit denen umgegangen wurde, wird im Verlauf der Dokumentation beschrieben. Aber grundsätzlich kann gesagt werden, dass mit einem Machine Learning Modell, welches ein RMAE von fast 1% vorhersehen konnte, das Ziel des Projektes erreicht wurde. Es wurde zudem noch ein Datensatz erstellt, der aber noch erweitert werden kann, wenn mehr Daten vorhanden wären. Die Anforderungen im Anforderungskatalog wurden somit Grossteils erfüllt. In diesem Projekt konnte das Wissen von Python im Umgang von Datenanalysen vertieft werden. Vor allem der Umgang mit grossen Datensätzen musste genauer behandelt werden. Die Priorisierung der Aufgaben in einem Machine Learning Projekt wurden auch klar und sollte in zukünftigen Projekten hilfreich sein.

Bei einer Weiterführung des Projekts könnten die SAP-Daten mit einbegriffen werden, wenn alle Daten aufgefunden wurden und anhand der Bill of Materials und den Production Orders können diese zu den Batches der Metalloxid-Varistoren zugeteilt werden. Aus dem Factory Layer können die Sinterofen Temperaturen theoretisch den einzelnen Zonen zugewiesen und als alternativen zu den Berichten verwendet werden und die restlichen Prozessdaten können auch mit einbegriffen werden, wenn genug Daten vorhanden sind. Wenn die Rückverfolgbarkeit im Factory Layer auf einzelne Varistoren oder Laufende Nummern im Sinterofen erstellt werden kann und mehrere Aufnahmen der Atmosphäre gemacht werden können, kann die Genauigkeit der Vorhersagen noch mehr zunehmen. Aus dem MOPA-Viewer können auch die geschnittenen Varistoren einbegriffen werden, wenn die Höhe auch aufgenommen wird. Wenn ein neuer Datensatz erstellt wurde, kann erneut ein Deep Learning Modell programmiert werden mit neuen Hyperparameter und vielleicht kann die Standardabweichung auch besser vorhergesagt werden. Zudem können auch Algorithmen wie Random Forest und Gradient Boosting genauer untersucht werden, da im Rahmen dieses Projektes nur standardmässige Hyperparameter verwendet wurden.

7 Literaturverzeichnis

Apache Parquet. (kein Datum). Abgerufen am 09. April 2023 von Apache Parquet: https://parquet.apache.org/

Burns, E., Lawskowski, N., & Tucci, L. (Februar 2023). *What is artificial intelligence (AI)? Definition, Benefits and Use Cases*. Abgerufen am 13. Februar 2023 von TechTarget: https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence

IBM. (kein Datum). *What are Neural Networks?* | *IBM*. Abgerufen am 13. Februar 2023 von IBM: https://www.ibm.com/topics/neural-networks

IBM. (kein Datum). *What is a Decision Tree* | *IBM*. Abgerufen am 02. Juli 2023 von IBM: https://www.ibm.com/topics/decision-trees#:~:text=data%20mining%20solutions-,Decision%20Trees,internal%20nodes%20and%20leaf%20nodes.

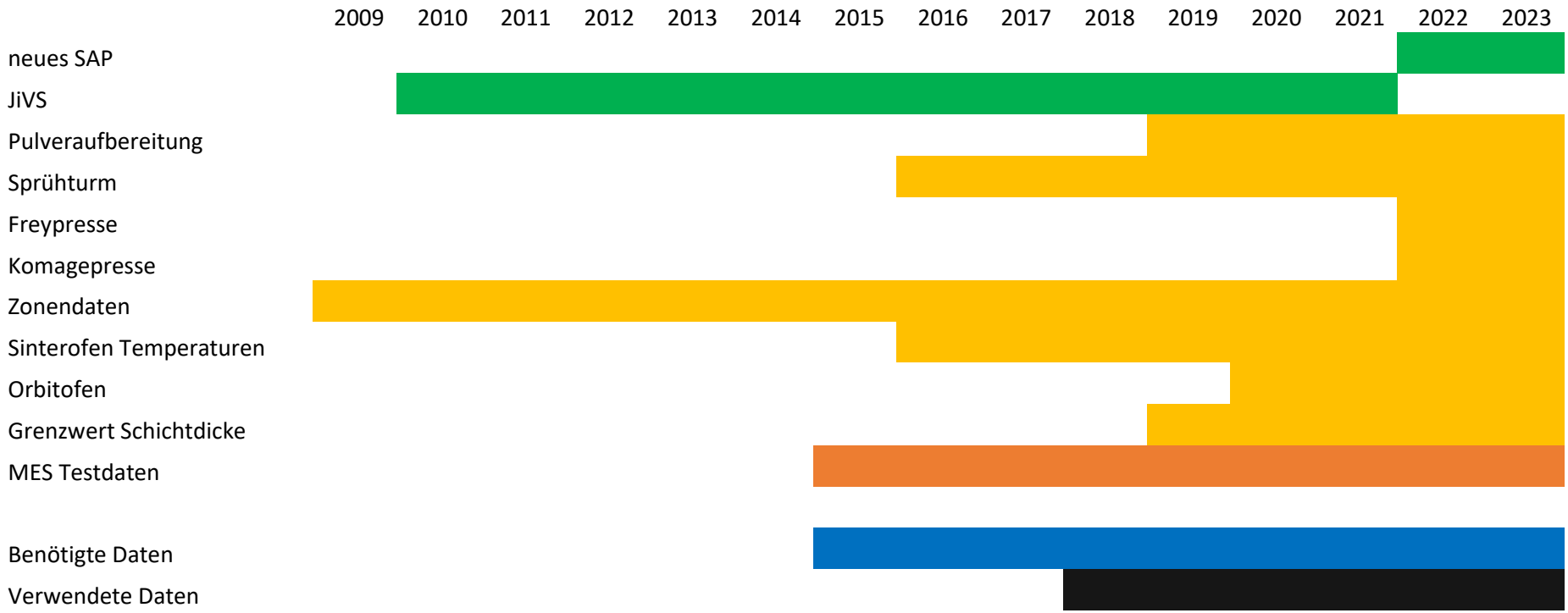
IBM. (kein Datum). *What is Machine Learning?* | *IBM*. Abgerufen am 13. Februar 2023 von IBM: https://www.ibm.com/topics/machine-learning

PyTorch. (kein Datum). *ReLu - PyTorch 2.0 documentation*. Abgerufen am 18. Mai 2023 von PyTorch: https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html

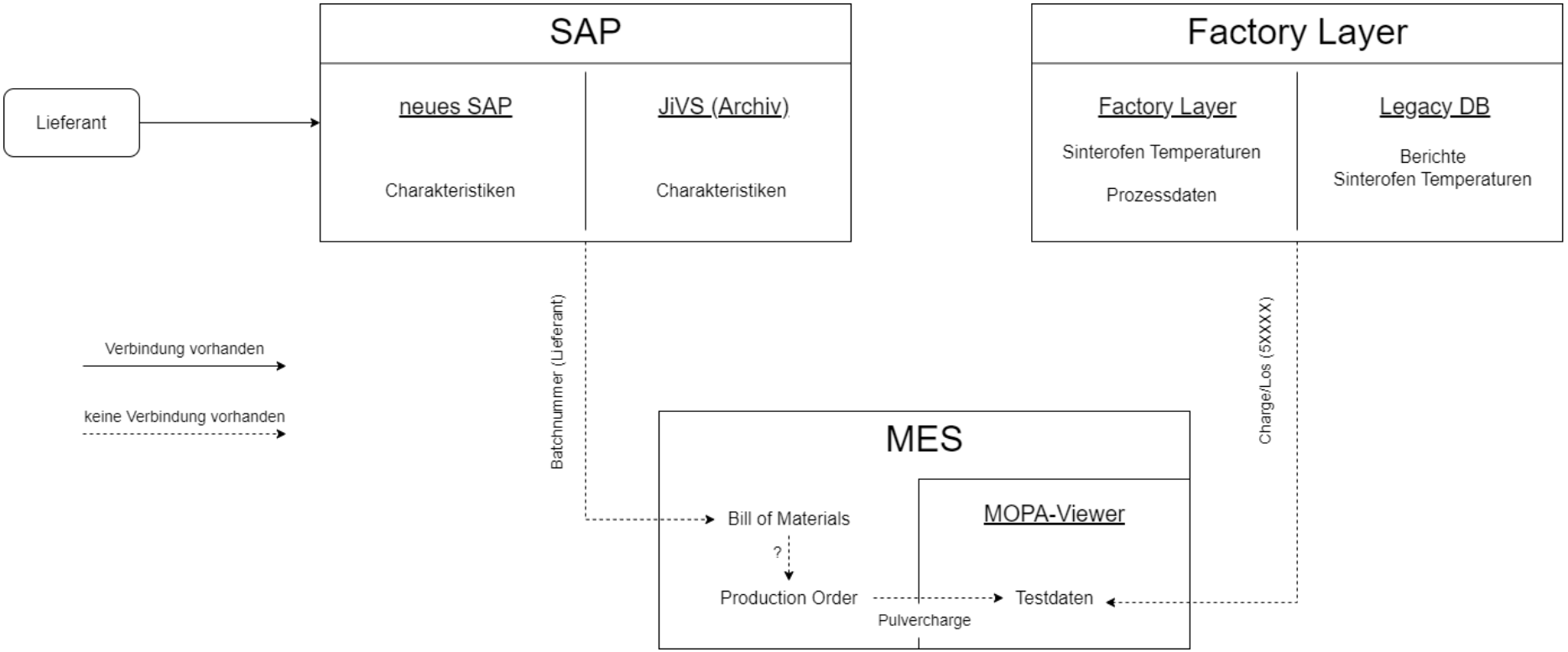
8 Anhangsverzeichnis

8.1 Verfügbarkeit der Daten 35
8.2 Datenfluss 36
8.3 Legende MES-Daten (MOPA-Viewer) 37
8.3 Ablauf Deep Learning Funktion 40
8.4 Python-Skripte 41
8.5 Projektauftrag 46

8.1 Verfügbarkeit der Daten



8.2 Datenfluss



8.3 Legende MES-Daten (MOPA-Viewer)

Spaltenname	Beschreibung	Einheit	Datentyp	Datenwerte
FAUF_2_Nr	Fertigungsauftrag		Integer	
MO_Charge_Nr	Metalloxid-Batchnummer/Nummer der Charge bzw. Los		Integer	
MO_Charge_FERT	Fertigungsvariante für das Los (N=Normal/V=Versuch)		String	N/V
MO_Charge_Segment	Segmentierung des Los beim Stillstand vom Sinterofen		Integer	1
MO_Charge_Lage	Lage im Ofen 1=Oben/2=Mitte/3=Unten A=aging N=keine Trennung → alle zusammen		String	1/2/3/A/N
Pulver_Charge_Nr	Batchnummer für Pulver		String	
Pruefplan_Nr	«Material_Nr»_Prod: Name der Prüfpläne für verschiedene Materialnummern (stimmt immer mit der Materialnummer überein)		String	
Pruefplan_Rev	Revisionsnummer von «Pruefplan_Nr» (jede Prüfplannummer kann mehrere Revisionsnummern bzw. verschiedene Tests haben)		Integer	1-10
Material_Nr	SAP-Materialnummer		String	
Material_Bezeichung	Produktbezeichnung: MO-WIDERSTAND «Var_Typ» (stimmt meistens mit dem Varistor-Typ überein (99.6%))		String	
Var_Typ	$Mb_1b_2z_1Hz_2z_3b_3b_4b_5(b_6)$ (z_i = Zahlen, b_i = Buchstaben) - M = Metalloxid - b_1 = Formgebung (A=Vollzylinder, E=Hohlzylinder, G=Rechteck, L=Monoblock) - b_2 = Prüfaufwand (A=Voll, B=Stichprobe, E=Reduziert, F=Energieäquivalent, H=erweiterte voll) - z_1 = Querschnitt (1=Rechteck 9.5x32.6, 3=38, 4=42, 5=47, 6=62, 7=75, 8=85, 9=108) - Hz_2z_3 = Höhe[mm] (H01 = 1.4, H46 = bis 46, HXX=geschnittene Varistoren) - b_3 = Zusammensetzung (Z = Z2, G = G9, E = E1) - b_4 = Beschichtung der Mantelfläche (S = Silikon auf Glas, G = Glas, L = Lack, N = keine) - b_5 = Beschichtung der Stirnfläche (A=Vollflächig mit Aluminium, B=Aluminium mit Rand, M=Vollflächig mit Messing, N=Messing mit Rand - b_6 = Up-Feldstärkenvariante (keine=standardmässig, P=erhöht)		String	
Var_D	Varistor Durchmesser (+/- 1mm)	mm	Integer	38/42/47/62/75/85/108
Var_H	Varistor Höhe (+/- 1mm)	mm	Float	2.2-46
t_Pruefung	Zeitpunkt der ersten Aufnahme des MOV		Timestamp	
Var_Id	Identifikationsnummer vom Varistor		Integer	
Status_Pruefung	Status, ob Prüfungen erfolgreich durchgeführt wurden (oder was für ein Fehler).		String	Ok FehlerMessung DefektBenutzer DefektPv_AC DefektUref_AC DefektKlasse FehlerStossstrom DefektKurzschluss DefektUref_DC
Var_Klasse_1	Klassen für einfacheres Zusammenbauen des Aktivteils		Integer	0-9999

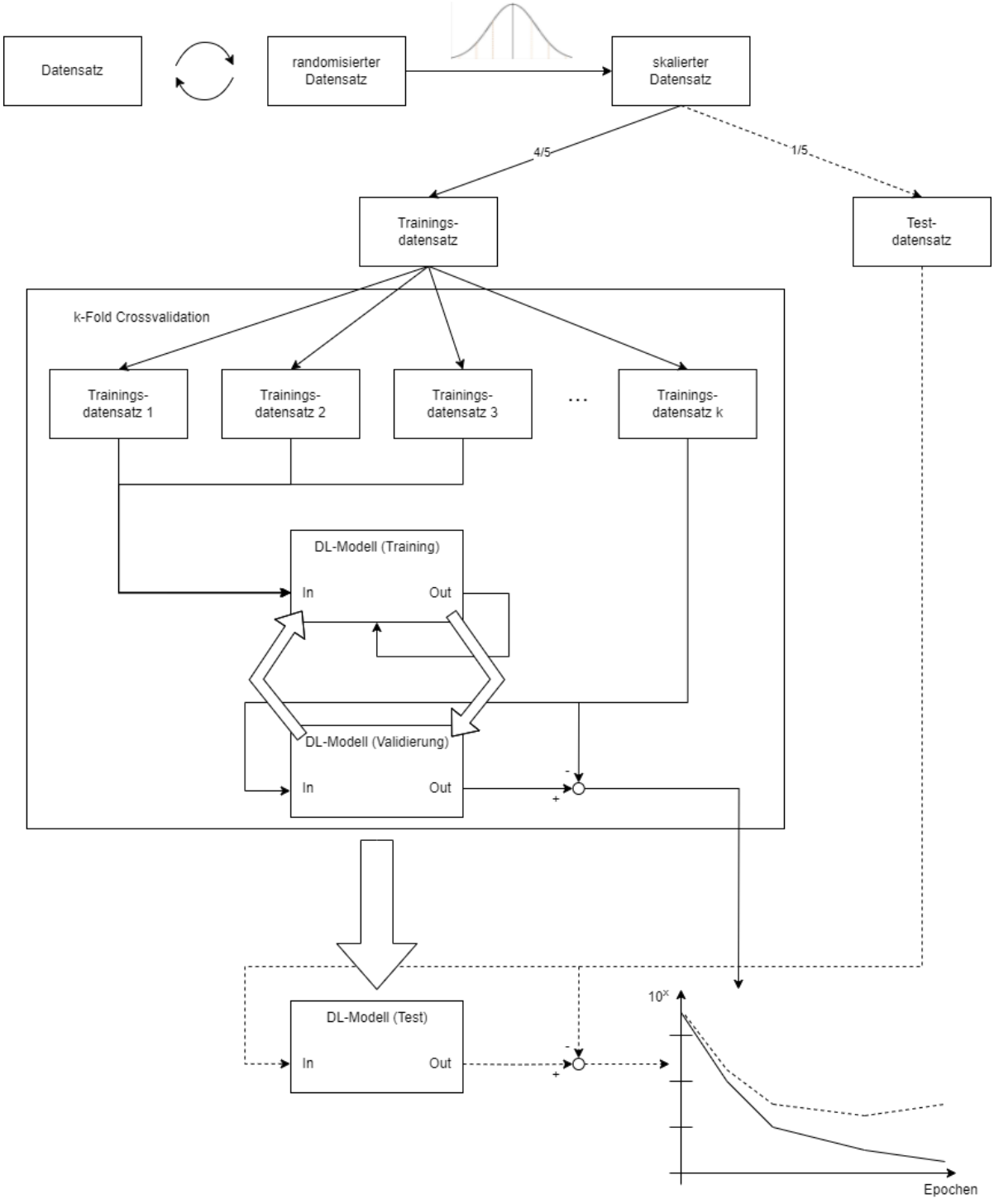
Var_Klasse_2	Doppelklassifizierungen für Spezialfälle, falls Anforderungen für beide erfüllt sind (kann nur kleiner sein als «Var_Klasse_1», 9999=keine Doppelklassifizierung)		Integer	0-9999
T_Mittelwert_Pruefung	Durchschnittliche Dauer einer Prüfung	s	Float	0/15.86-62.72
H_Pruefung	Prüfung der Höhe des Varistors (wird nicht aufgenommen)			
t_DC	Zeitstempel Gleichstromtest		Timestamp	
Iref_DC_Ist	Referenzgleichstrom (Spannung wird erhöht, bis 1mA erreicht wird)	mA	Float	0/0.98/1/1.01/1.21/1.51/nan
Iref2_DC_Ist	Wird nicht getestet		Integer	0/nan
Uref_DC_Ist	Referenzgleichspannung (Spannung bei der 1mA Strom durchgeflossen ist)	kV	Float	0-12.26
Uref_DC_korr	$\frac{Uref\_DC\_Ist}{1 - 0.0002663 * (Temp[°C] - 20)}$	kV	Float	0-12.29
Uref2_DC_Ist	Wird nicht getestet		Integer	0/nan
Uref2_DC_korr	Wird nicht getestet		Integer	0/nan
Pv_DC_Ist	Wird nicht getestet		Integer	0/nan
Pv_DC_korr	Wird nicht getestet		Integer	0/nan
Alpha_DC_Ist	Wird nicht getestet		Integer	0/nan
Status_DC	Status Gleichstromtest		String	Ok Nan FehlerMessung Defektkurzschluss
t_AC	Zeitstempel Wechselstromtest		Timestamp	
Iref_AC_Ist	Referenzwechselstrom	mA	Float	0-34.78 (0.09-12.48)
Uref_AC_Ist	Referenzwechselspannung	kV	Float	0-8.81
Uref_AC_korr	$\frac{Uref\_AC\_Ist}{1 - 0.00033 * (Temp[°C] - 20)}$	kV	Float	0-8.84
Pv_AC_Ist	Verlustleistung	mW	Integer	0/10-8671
Pv_AC_korr	$\frac{Pv\_AC\_Ist}{1 + 0.022 * (Temp[°C] - 20)}$	mW	Integer	0/9-6385
Status_AC	Status Wechselstromtest		String	Ok Nan FehlerMessung DefektKurzschluss
Faktor_a	Faktor a von abgebrochenem Machine Learning Projekt (wird nicht aufgenommen)			
Faktor_b	Faktor b von abgebrochenem Machine Learning Projekt (wird nicht aufgenommen)			
t_Rest	Zeitstempel Restspannungstest		Timestamp	
Ip_Rest_Ist	Stossstrom (Peak)	A	Float	0/0.7-14.49
Ul_Rest_Ist	Ladespannung	kV	Float	0-30.1
Up_Rest_Ist	Spannungspeak	kV	Float	0/0.26-34.23
Up_Rest_korr	$\frac{Up\_Rest\_Ist}{1 + 0.00049 * (Temp[°C] - 20)}$	kV	Float	0/0.02-34.04
Q_Rest_Ist	Restladung	C	Float	0-0.613

Anz_Rest_Ist	Anzahl Impulse für die Restspannungsmessung (zweiter Impuls, wenn Ladespannung angepasst werden muss)		Integer	1/2/nan
Status_Rest	Status Restspannungstest		String	Ok Nan FehlerMessung DefektKurzschluss FehlerStossstrom
t_Ladung	Wird nicht getestet			
Ip_Ladung_Ist	Wird nicht getestet			
Q_Ladung_Ist	Wird nicht getestet			
Anz_Ladung_Ist	Wird nicht getestet			
Status_Ladung	Wird nicht getestet			
t_Ladung_1	Start Zufahren Funkstrecke		Timestamp	
Ip_Ladung_Ist_1	Impulsstrom, um den Ladungseintrag zu erzielen vom letzten geprüften Impuls	A	Float	
Q_Ladung_Ist_1	Ladungs-Eintrag vom letzten geprüften Impuls	C	Float	
Impulse_Ist_1	Anzahl durchgeführte Impulse		Float	
Status_Ladung_1	Status Ladungstest		String	Ok FehlerMessung DefektKurzschluss
Ladung_Summe_1	Summe aller Impulse (fortlaufend addiert)	C	Float	
Temperatur_MO_1	Temperatur vor den Ladungstests (< 45°C)	°C	Float	
Schallpegel_MO_1	Aufgenommener Schallpegel vom Energieimpuls (über 150dB kann nicht mehr gemessen werden)	dB	Integer	77-145
Impulsdauer_1	Impulsdauer	ms	Float	
Impulsgruppe_1	Impulse werden in Gruppen aufgeteilt, um das Energieaufnahmevermögen mit Kühlphasen aufzuteilen		Integer	1-6

Diese Legende wurde anhand der Daten zwischen dem 22.01.2018 und 22.02.2023 erstellt.

Grau markierte Zeilen haben keine Messwerte.

8.4 Ablauf Deep Learning Funktion



8.5 Python-Skripte

Alle Python-Skripte wurden in einem Jupyter Notebook geschrieben und wurden in drei Gruppen eingeteilt. Im SAP-Ordner befinden sich die Python-Skripte, wo die Daten der Rohstoffe bearbeitet wurden. Im Factory Layer sind die Skripte von den Prozessdaten und den Berichten vom Sinterofen. Im MES befinden sich die Skripte, wo die Testdaten, Bill of Materials und Production Orders transformierten. Im Datensatz-Ordner befinden sich die Skripte, die Datensätze aus dem Factory Layer und MES kombinieren und im ML-Ordner befinden sich alle Skripte wo Machine Learning durchgeführt wurde. Die Namen der Skripte sind identisch zu den eigentlichen Files.

SAP:

- 01_sap_pdf2table
  Ein Versuch mithilfe von den Bibliotheken Comalot, Tabula und pytesseract die gescannten Analysezertifikate in Tabellen umzuwandeln, damit die Charakteristiken nicht von Hand abgeschrieben werden müssen. Dies ist jedoch gescheitert, weil die Tabellen entweder nicht gefunden wurden oder die benötigten Programme wurden nicht gefunden.

  Input: PDF-Analysezertifikate

- 02_jivs_combine
  Kombiniert die Charakteristiken von den Materialien, die aus dem JiVS exportiert wurden und exportiert diese als ein CSV-File, damit für jedes Material ein File mit allen Angaben vorhanden ist. Zusätzlich wird angegeben welche Beschreibungen doppelt in einem Material angegeben wurden, aber verschiedene Werte angenommen haben. Provisorisch wurden für diese Beschreibungen der Durchschnitt genommen.

  Input: Exports der Charakteristiken aus JiVS
  Output: Eine Tabelle für jedes Material mit allen Angaben ([Materialnummer].csv)

Factory Layer:

- 01_convert_time_sinterofen
  Transformiert den UTC-Zeitstempel aus dem Factory Layer zur Schweizer Lokalzeit, damit ein einheitliches Zeitformat mit dem MES vorhanden ist. Die Somer-/Winterzeit wird hier auch beachtet.

  Input: Sinterofen Temperatur Tabelle (sinterofen[Jahr].parquet)
  Output: Sinterofen Temperatur Tabelle mit konvertiertem Zeitstempel (sinterofen[Jahr]_t.parquet)

- 02_filter_sinterofen_temp
  Vom Sinterofen werden 5 der 6 Spalten mit den Temperaturen entfernt, nur die Werte mit dem Batchformat 5XXXXxxxx beibehalten und alle Laufende Nummern unter null entfernt. Zudem werden nur Temperaturen zwischen 0 und 1250°C beibehalten. Um den benötigten Speicherplatz zu verringern, werden noch Datentypen angepasst. Die reduzierte Tabelle wird anschliessend kombiniert und exportiert.

Input: Sinterofen Temperatur Tabelle mit konvertiertem Zeitstempel (sinterofen[Jahr]_t.parquet)
Output: Gefilterte und kombinierte Sinterofen Tabelle (sinterofen_filtered.parquet)

- 03_factory_layer_log_entrys_to_groups
  Aus den 6 Tabellen werden 3 verwendet, um eine Legende zu erstellen und die anderen 3 werden verwendet, um die Tabellen mit den Werten zu erstellen. 2 der Tabellen beinhalten alle Werte aller Anlagen. Im Skript wird für jede Anlage ein PARQUET-File exportiert, damit die Einträge zu den Anlagen in nur einer Tabelle zugeordnet werden können.

  Input: Von jedem Jahr 6 Tabellen aus dem Factory Layer Backup
  Output: Eine Tabelle pro Anlage ([Anlage].parquet)

- 04_factory_layer_log_entrys_to_groups_with_timeconverter
  Das gleiche wie in 03_factory_layer_log_entrys_to_groups, aber der Zeitstempel wird vom UTC-Zeitstempel zur Schweizer Lokalzeit konvertiert für ein einheitliches Zeitformat.

  Input: Von jedem Jahr 6 Tabellen aus dem Factory Layer Backup
  Output: Eine Tabelle pro Anlage ([Anlage].parquet)

- 05_combine_sinterofen_reports
  Mit den 3 Tabellen werden Berichte erstellt die Informationen von den Temperaturen jeder Zone von jedem Batch. Die kombinierte Tabelle wird als CSV-File exportiert.

  Input: 2 Tabellen aus der Online Datenbank und 3 Tabellen aus der Archiv Datenbank
  Output: Kombinierte Berichte vom Sinterofen (var_allvaristors_combined.csv)

- 06_reports_filter
  Um den Datensatz zu reduzieren werden redundante Spalten werden entfernt und nur die Batches, die im MES vorhanden sind, werden beibehalten. Die gefilterte Tabelle wird danach als CSV exportiert.

  Input: Kombinierte Berichte vom Sinterofen (var_allvaristors_combined.csv)
  Output: Bereinigte Berichte vom Sinterofen (var_allvaristors_cleaned.csv)

MES:

- 01_split_status_ladung_string
  In diesem Code wird versucht den String aus dem Export vom Support in die einzelnen Spalten zu trennen. Dies ist fehlgeschlagen, weil die Länge der einzelnen Strings nicht immer gleich lang ist.

  Input: MES Export vom Support (MOPAViewerOutput.csv)

- 02_combine_support_and_local
  Der Code kombiniert die exportierten Tabellen vom Support und die wo lokal exportiert wurde, weil beide Exports fehlerhaft sind.

Input: MES-Export vom Support und vom lokalen Export (MOPAViewerOutput.csv & alle Exports die lokal gemacht wurden)
Output: Kombinierter MES Datensatz (mes_mopa_viewer.parquet)

- 03_mes_filter
  Die Daten aus dem MES werden gefiltert, um den Datensatz zu in den Grössen und Komplexität zu reduzieren.

  Input: Kombinierter MES Datensatz (mes_mopa_viewer.parquet)
  Output: Gefilterter MES Datensatz (mes_mopa_viewer_filtered.parquet)

- 04_mes_transform
  Die Spalte «Var_Typ» wird aufgeteilt, damit die einzelnen Informationen in separaten Spalten vorhanden sind. Danach wird definiert welche Spalten im jeweiligen Batch immer gleich sind und nur die Spalten, die für das Machine Learning verwendet werden, werden beibehalten, bevor es exportiert wird.

  Input: Gefilterter MES Datensatz (mes_mopa_viewer_filtered.parquet)
  Output: Reduzierter MES Datensatz (mes_reduced.parquet)

- 05_mes_to_vector_space
  Das Datenset wird in das Vector Space Format transformiert damit es für Machine Learning verwendet werden kann, indem für jeden Wert einer Klasse eine neue Spalte zugewiesen wird.

  Input: Reduzierter MES Datensatz (mes_reduced.parquet)
  Output: MES Datensatz im Vector Space Format (mes_vs.parquet)

- 06_combine_and_filter_po
  Damit alle relevanten Angaben in einer Tabelle vorhanden sind, werden alle Exporte von den Production Orders kombiniert und nur die Datenpunkte, welche ein Material haben, welches im JiVS vorhanden sind, werden beibehalten.

  Input: Production Orders von jedem Jahr (DataExtraction_[Jahr].csv)

Datensatz:

- 01_create_fl_mes_dataset
  Die Berichte aus dem Factory Layer werden mit dem MES-Datenset welches sich im Vector Space Format befindet kombiniert, um einen Datensatz zu erstellen welches für das Machine Learning verwendet werden kann.

  Input: Bereinigte Berichte vom Sinterofen und MES Datensatz im Vector Space Format (var_allvaristors_cleaned.csv & mes_vs.parquet)
  Output: Datensatz mit Berichten vom Sinterofen im Factory Layer und MES (fl_mes_vs.parquet)

- 02_final_dataset
  Der Datensatz, der für die richtigen Analysen verwendet wird, wird vorbereitet indem für jeden Test ein separater Datensatz gemacht wird, damit später Programmieraufwand erspart werden kann.

Inputs: Datensatz mit Berichten vom Sinterofen im Factory Layer und MES (fl_mes_vs.parquet)
Outputs: 5 Datensätze von jedem Test ([Test].parquet)

Machine Learning:

- 01_dl_mes_individual
  Ein Deep Learning Modell wurde erstellt welches die einzelnen Varistoren als Inputs nimmt und die Genauigkeit der vorhergesagten Resultate herausgibt.

- 02_dl_mes_batch
  Ein Deep Learning Modell welche Batches der Varistoren als Inputs nimmt, welche eine Gaussverteilung als Label haben und die Genauigkeit der vorhergesagten Resultate herausgibt.

- 03_dl_model_with_all_possible_combinations
  In diesem Skript wurden Funktionen erstellt, die zusammen alle möglichen Kombinationen von einem gegebenen Parameterraum erstellen. Eine weitere Funktion wurde erstellt, bei der ein Datensatz und Parameter als Argumente gegeben werden können und ein Deep Learning Modell trainiert wird. Alle Resultate werden in einem DataFrame abgelegt.

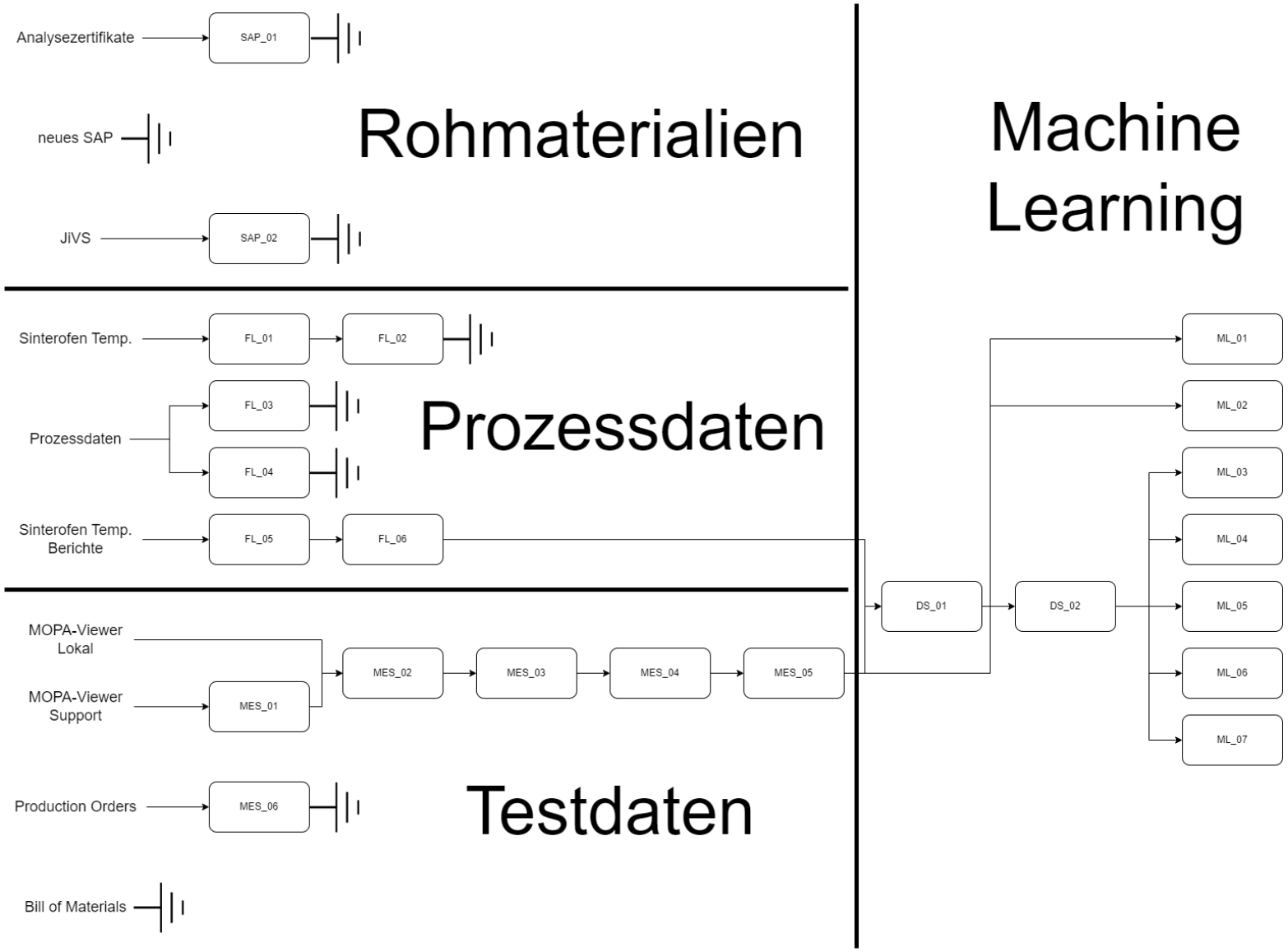
- 04_komplex_dl_model
  Die Funktion aus 04_dl_model_with_all_possible_combinations wurde leicht abgeändert und es wurden komplexere Architekturen vom Deep Learning Modell verwendet, weil der Parameterraum zu klein war.

- 05_random_forest
  Ein Random Forest der auch Vorhersagen macht und die Wichtigkeit der Variablen in Form von Gewichten wiedergibt.

- 06_gradient_boosting
  Gleich wie beim Random Forest, aber mit einem anderen Algorithmus namens Gradient Boosting.

- 07_xgboost
  Gleich wie beim Gradient Boosting, aber das Modell wurde mit einer Bibliothek erstellt, welches sich auf Gradient Boosting spezialisiert.

8.6 Übersicht Python-Skripte



8.7 Projektauftrag

8.7.1 Ausgangslage

Hitachi Energy AG hat einen langen Produktionsprozess für Metalloxid-Varistoren. Diese elektronischen Bauteile sind Blöcke, die zusammengesetzt werden können und dienen zum Überspannungsschutz. Aus dem Produktionsprozess wurden Daten, welche noch verschiedene Formate enthalten, gesammelt und in mehrere Datenbanken abgespeichert. Seit dem Jahr 2017 wurden insgesamt über 5'000'000 Datenpunkte gesammelt, die Werte über die Rohmaterialien, den Produktionsprozess und Testresultaten beinhalten. Mittels Machine Learning bzw. Deep Learning soll herausgefunden werden, ob die Daten ausreichen, um die Durchbruchspannung der Varistoren vorherzusehen und manipulieren.

8.7.2 Qualitative Ziele

Der erste Teil dieser Arbeit ist es ein Datensatz zu erstellen, welches für die Modellierung von Machine Learning Algorithmen geeignet ist. Im zweiten Teil Arbeit soll ein Machine Learning Modell erstellt werden, welches anhand von vorgegebenen Parametern die Durchbruchsspannung oder eine Klassifizierung vorherseht.

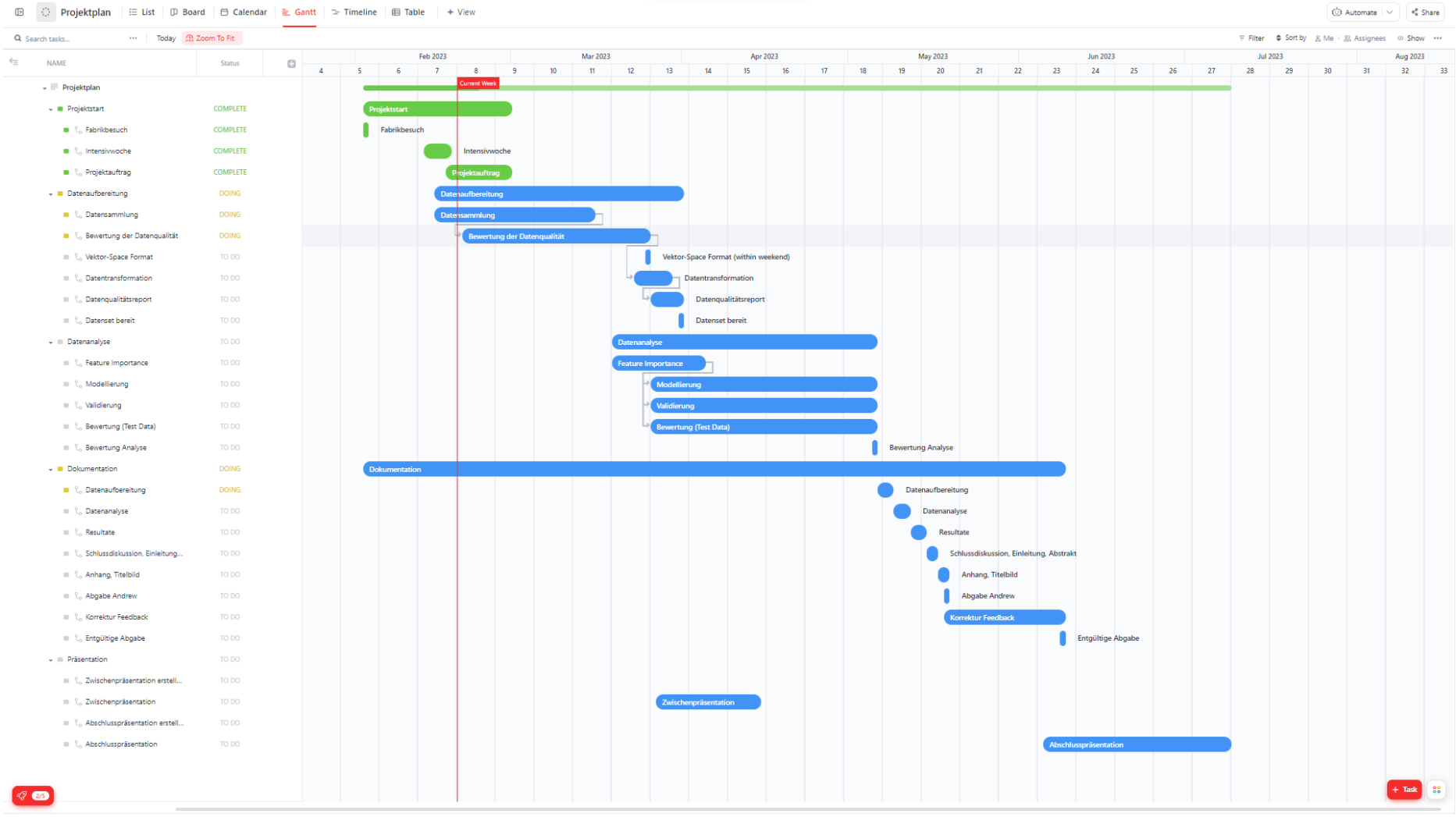
8.7.3 Quantitative Ziele

- Der Datensatz muss sich in einem Vektor-Space Format befinden.
- Alle Werte müssen sich in einem geeigneten Format für Machine Learning befinden.
- Bewertungen aus dem ersten Machine Learning Modell sind abgeschlossen.
- Die Dokumentation muss bis am 09.06.2023 fertiggestellt und verteilt werden.

8.7.4 Anforderungskatalog

Nr.	Beschreibung	Priorität
1	Datenaufbereitung	
1.1	Alle relevanten Spalten müssen sich in einem Dokument befinden.	Muss
1.2	Alle Daten im Datensatz befinden sich im gleichen Format.	Muss
1.3	Es muss eine Legende von Datensatz erstellt werden.	Muss
1.4	Alle Spalten im Datensatz müssen realistische Wertintervalle haben.	Muss
1.5	Anomalien im Datensatz müssen entfernt oder erklärt werden.	Muss
1.6	Der Datensatz beinhaltet keine NULL-Werte.	Muss
1.7	Duplikate im Datensatz müssen entfernt oder erklärt werden.	Muss
1.8	Alle Spalten im Datensatz müssen normalisiert werden.	Muss
1.9	Alle Spalten im Datensatz können mit dem Z-Score normalisiert werden.	Soll
1.10	Die Reihenfolge vom Datensatz muss randomisiert werden.	Muss
1.11	Der Datensatz soll unter 100GB gross sein.	Soll
1.12	Der Datensatz kann komprimiert unter 10GB gross sein.	Nice to have
1.13	Ein Datenqualitätsreport muss über die Daten geschrieben werden.	Muss
1.14	Das Dokument vom Datensatz kann das Format CSV/PARQUET/FTR haben.	Soll
2	Datenanalyse	
2.1	Der Datensatz muss ein Trainingsset beinhalten.	Muss
2.2	Der Datensatz muss ein Testset beinhalten.	Muss
2.3	Der Datensatz kann ein Validierungsset beinhalten	Soll
2.4	Die Validierung kann mit k-Fold Cross-Validation durchgeführt werden.	Kann
2.5	Es muss ein Deep Learning Modell trainiert werden.	Soll
2.6	Es kann ein Regressionsmodell trainiert werden.	Soll
2.8	Das Modell kann interpretierbar sein.	Nice to have
2.9	Das Modell kann anhand von Eingangsparametern eine Klassifizierung vorhersehen.	Soll
2.11	Das Modell kann Klassifizierungen anhand des Testsets mit einer Genauigkeit von 90% machen.	Nice to have
2.11	Statistische Analysen vom Modell/-en müssen visualisiert werden.	Soll
2.12	Das Vorgehen für die Reproduzierbarkeit muss dokumentiert werden.	Muss
3	Organisatorisches	
3.1	Die Daten dürfen nicht veröffentlicht werden, wenn nicht alle Parteien zugestimmt haben.	Muss
3.2	Der Datensatz kann auf einer Cloud abgespeichert werden.	Nice to have
3.3	Der Datensatz kann für alle Beteiligten am Projekt zugänglich sein.	Soll

8.7.5 Projektplan



8.7.6 Risikomanagement

Customer requirements, Competition, Specification, Expectations												
	The performance of the machine learning model is not sufficient.	Project termination	Lack of good data, computational limitations or not enough time	5	2	5	50	Use other machine learning algorithms	3	2	4	24
	The model is not interpretable	Quality control is limited	Parameters can't be adjusted in the factory based on the model	2	3	3	18	Additionally train an interpretable model	1	1	3	3
Development, Research, Design, IP, Resourcen												
	Information of the legends is not available	No understanding of the data and worse validation	Infromation was not written down	2	4	4	32	Asking experts in the field	2	1	4	8
	Anomalies or duplicates can't be explained	Bad data quality	Sensorfailures or experiments	2	4	3	24	Asking experts or remove them	2	2	2	8
	Columns contain many NULL values	Empty cells are not suitable for machine learning	Sensordata startet getting tracked later or errors in production	3	5	2	30	Replace with average/median or remove the column	2	5	2	20
	The dataset is too big for a pandas dataframe	the model can't be trained on the whole dataset	too many relevant variables and datapoints	3	2	3	18	Split the dataset into multiple files and reduce number of columns	1	2	3	6
	The performance of the hardware is not sufficient	Training the model takes too long	Too many variables	3	2	2	12	Use server in Dättwil	1	2	2	4
	The dataset is too complex	The model will be overfittet to the data	too many relevant variables	1	2	2	4	Reduce number of dimensions	1	1	2	2
	The data preparation takes too long	Not enough time to train a machine learning model	technical difficulties or too complex data	3	3	5	45	Start training the machine learning model without the whole dataset	2	2	3	12
SCM, Suppliers												
	No acces to all of the data	possibly worse performance of the machine learning model	Slow communications or technical difficulties	3	2	4	24	Proceed with an incomplete dataset	3	1	2	6
Others												
	The data gets leaked, without consent from all parties	The NDA will be violated	Hacker or virus	5	1	4	20	Only store data on company devices or the cloud	2	1	1	2
						GREEN	1					8
						YELLOW	10					3
						RED						
						TOTAL Risks	11					11