

Software-Projekt

Entwicklung einer Web-Applikation zur Erkennung von Zusammenhängen in den Daten aus Einzelblattmontagen

Von

Nepomuk Bense

Sascha Bollmann

Mehmet Erdede

Zelgai Nemati

Alexander Siewert

Studiengang: Systems Engineering

Betreuender Professor: Prof. Dr.-Ing Klaus-Dieter Thoben

Betreuer: M. Sc. Maria Teresa Alvela Nieto

M. Sc. Aljoscha Sander

Abgabedatum: 01. September 2021

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht.

Bremen, den 01. September 2021

Bollmann, Sascha



Name, Vorname

Bremen, den 01. September 2021

Siewert, Alexander



Name, Vorname

Bremen, den 01. September 2021

Bense, Nepomuk

Name, Vorname

Bremen, den 01. September 2021

Nemati, Zelgai

Name, Vorname

Bremen, den 01. September 2021

Erdede, Mehmet

Name, Vorname

Inhaltsverzeichnis

1	EINLEITUNG	5
2	WEB-APPLIKATION	6
2.1	Technologieauswahl	6
2.2	Architektur	11
2.3	Versions-Management	12
2.4	Integrationsautomatisierung (CI/CD)	14
2.5	Webseite Funktionalität	16
3	MASCHINELLES LERNEN	17
3.1	Arbeitsprozess / Planung und Anforderungen	17
3.2	Recherchephase	18
3.3	Technologieauswahl	19
3.4	Datensatz und Preprocessing	20
3.4.1	Turbinen bzw. Geometrie Daten	22
3.4.2	Wetterdaten	22
3.4.3	Feature Übersicht und Datenqualität	25
3.4.4	Preprocessing	27
3.5	Resampling der Daten und Korrelationsmatrix	28
3.6	Auswahl der Machine Learning Algorithmen	33
3.6.1	ROLFs	35
3.6.2	Multilayer Perceptron bzw. MLP und Neuronales Netz	39
4	FAZIT	43
5	LITERATURVERZEICHNIS	44

Begriffe und Abkürzungen

Algorithmus	Eine Abfolge von Anweisungen in Codeform
API	Systemschnittstelle
Backend	Tiefgehende Logik einer Internetseite
Backpropagation	Verfahren zur Fehlerrückführung zum Lernen von Neuronalen Netzen
Cluster	Ballungsraum von korrelierenden Eingabeparametern
CSS	Cascading Style Sheets ist eine Programmiersprache zum Implementieren von Stylesheets
Compiler	Computerprogramm das programmierten Code in eine andere Programmiersprache übersetzt
Framework	Bestehende Softwarelösung für ein Problem
Frontend	Sichtbarer Teil einer Internetseite
IDE	Software-Entwicklungsumgebung
ML	Machine Learning
OSBI	Offshore single blade installation
Overfitting	Neuronalen Netz, dass zu stark an die Testdaten angepasst ist.
Preprocessing	Vorverarbeitung von Daten für das Maschinelle Lernen
REST	Webservice Architektur
SSL	Skript zur sicheren Datenübertragung im WEB

Stylesheet	Code zum Designen einer Homepage
Supervised	Zielgerichtetes maschinelles Lernen
Token	Zufällig generierte Zeichenfolge für die Authentifizierung
UML	Unified Modelling Language
Unsupervised	Maschinelles Lernen ohne konkrete Zielvorgabe
WEB-Anwendung	Webseite bzw. Internetseite

1 Einleitung

„Offshore Wind liefert bereits heute einen substantziellen Teil des Energiemixes.“ [San20]

Heutzutage werden die meisten Offshore-Windenergieanlagen komponentenweise installiert und um die Kosten weiter zu senken, ist es insbesondere notwendig, den Installationsprozess zu verbessern. *„Die Installation der Blätter stellt dabei die größte Herausforderung dar“ [San20]*, denn hier ist hohe Präzision und Sorgfalt erforderlich, um die Blattschraube in den Nabenflansch einzusetzen. Wind und Wellen üben Last auf die Strukturen aus und die daraus resultierenden Relativbewegungen zwischen Turm und den Rotorblättern erschwert die Blattmontage. [San20]

„Überschreitet die Relativbewegung einen bestimmten Schwellenwert, ist keine Installation mehr möglich und es kommt zu einer kostspieligen Verzögerung.“ [San20]

„Basierend auf den Messdaten, welche während der Installation des Windparks in der Nordsee aufgezeichnet wurden“ [San20], werden Methoden des maschinellen Lernens verwendet, um den Zusammenhang zwischen Umgebungsvariablen wie Windgeschwindigkeit und der Beschleunigung von Turm, Gondel und Rotorblättern zu identifizieren und prognostizieren. [San20]

Ziel des Softwaretechnikprojektes ist die Erkennung von Zusammenhängen in Messdaten aus der Einzelblattmontage. Hierzu soll eine WEB-Applikation entwickelt werden, die basierend auf aufgezeichneten Daten, Zusammenhänge berechnet, damit Ingenieure diese Ergebnisse nutzen können, um ein besseres Verständnis der Rotorblattmontage und die gegebenen Umweltbedingungen zu erhalten. [San20]

2 Web-Applikation

2.1 Technologieauswahl

Die Technologieauswahl zu Beginn eines Software-Projektes bestimmt den weiteren Projektverlauf [Sch09]. Um dieser Entscheidung gerecht zu werden, bedarf es dem Einsatz einer systematischen Technologieanalyse. Im Rahmen dieser Projektarbeit wurde der folgende Ansatz gewählt, der sich in der Softwaretechnik Literatur etabliert hat [Sam08] [Das05] [Ahu05]:

1. Erstellen eines Kriterienkatalogs
2. Anwenden der Kriterien auf die zu untersuchenden Technologien
3. Bewertung und Gewichtung der einzelnen Ergebnisse
4. Entscheidung auf dieser Grundlage treffen

Diese Systematik zur Entscheidungsfindung wird folgend mehrmals praktisch angewandt, da moderne WEB-Anwendungen aus einem Kanon verschiedener Technologien bestehen. Um diesen Sachverhalt zu verdeutlichen, kann das Beispiel herangezogen werden, dass eine abstrakte Datenbanksprache wie MySQL nicht dafür verwendet werden kann benutzerfreundliche Oberflächen zu designen, da diese Sprache für einen anderen Zweck programmiert wurde und somit nicht mit einer Stylesheet-Sprache wie CSS verglichen werden kann.

Tabelle 2-1 Entscheidungsmatrix Frontend

Frontend							
Technologie Optionen		Beliebtheit	Konnektivität	Teaminterne Erfahrung	Funktions- umfang	Usability	Geschwindigkeit
Markup Language	HTML	Green	Green	Green	Green	Green	Green
	HAML	Yellow	Yellow	Red	Yellow	Yellow	Green
	restructuredText	Yellow	Yellow	Red	Yellow	Yellow	Yellow
Stylesheet Language	CSS	Yellow	Green	Green	Yellow	Red	Green
	SASS	Green	Green	Green	Green	Green	Green
	Tailwind CSS	Green	Green	Red	Green	Yellow	Yellow
Template Library	Bootstrap	Green	Green	Red	Yellow	Green	Green
	Material UI	Green	Green	Yellow	Green	Green	Green
Frontend Framework	React.JS	Green	Green	Green	Green	Green	Green
	Angular	Yellow	Green	Red	Green	Green	Green
	Vue.JS	Yellow	Green	Red	Green	Yellow	Green

Der Tabelle 2-1 ist die Entscheidungsgrundlage für die benötigten Frontend Technologien zu entnehmen. Dabei steht ein grün markiertes Feld für „gut“ ein gelb markiertes Feld für „befriedigend“ und ein rot markiertes Feld für „schlecht“. Diese Art der Bewertung ist auch den anderen Entscheidungsmatrizen zu entnehmen. Es wird eine Markup-Sprache benötigt, welche das Grundgerüst der Homepage bilden soll, eine Stylesheet Sprache, welche das Grundgerüst in ein benutzerfreundliches UI wandelt, eine Template Bibliothek welche die benötigten Icons, Schriftarten und vordefinierte Komponenten zur Verfügung stellt und ein Frontend Framework, welches die statische Homepage in eine dynamische wandelt. Diesen Untergruppen sind verschiedene lila markierte Optionen zugeordnet. Es soll eine Option pro Untergruppe für den weiteren Projektverlauf gewählt werden. Die gewählten Technologien haben wir in der Tabelle 2-3 zusammengefasst.

Tabelle 2-2 Entscheidungsmatrix Backend

Backend							
Technologie Optionen		Beliebtheit	Konnektivität	Teaminterne Erfahrung	Funktionsumfang	Usability	Geschwindigkeit
Authentifizierung	Firebase Authentication						
	JWT						
	Node.JS Passport						
Datenbank	MySQL						
	MongoDB						
API	GraphQL						
	REST						

Die Entscheidungsgrundlage für die Backend Technologien ist der Tabelle 2-2 zu entnehmen. Es wird eine Authentifizierungsschnittstelle benötigt, welche die Benutzerdaten verwalten soll, eine Datenbank, welche die hochgeladenen Datensätze speichert, eine API-Technologie, um die vom Benutzer getätigten Anfragen z.B. an die Datenbank zu überprüfen und ein Framework zum Programmieren des neuronalen Netzwerks. Auch hier wurden die ausgewählten Technologien in der Tabelle 2-4 zusammengefasst. Die Technologieauswahl zur Implementierung der Neuronalen Netze wird im Kapitel 3.3 thematisiert.

Tabelle 2-3 Ausgewählte Technologien Frontend




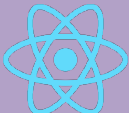
Technologie	Funktion	Kurzbeschreibung
HTML5 	Homepage Grundgerüst	HTML5 ist die aktuellste Version die von jedem Internetbrowser unterstützt wird und dient zur Strukturierung von Internetseiten. Nicht nur die Konnektivität bzw. Kompatibilität, sondern auch der Funktionsumfang dieser Markup Sprache sind dafür verantwortlich, dass sie sich hoher Beliebtheit erfreut.
SASS 	Styling Benutzeroberfläche	SASS steht für „Syntactically Awesome Stylesheets“, ist ein Präprozessor für die Stylesheet-Sprache CSS und erweitert diese um z.B. Funktionen, Variablen und Schleifen. Da alle etablierten Internetbrowser lediglich CSS verarbeiten können, wird der geschriebene SASS Code vom mitinstallierten Compiler nach jeder Codeänderung in CSS-Code übersetzt.
Material UI 	Icons und Schriftarten	Material UI ist eine vom Unternehmen Google entwickelte Bibliothek. Sie enthält diverse Komponenten, Schriftarten und Icons, die einer bestimmten Designkonvention entsprechen.
React.JS 	Dynamische Funktionen	React ist die weltweit meistbenutzte Bibliothek zum Erstellen von Benutzeroberflächen und ist in JavaScript programmiert. Dem Programmierer ist es mithilfe dieser Bibliothek möglich einzelne wiederverwendbare Komponenten zu programmieren, um somit redundantem Code vorzubeugen.

Tabelle 2-4 Ausgewählte Technologien Backend

Technologie	Funktion	Kurzbeschreibung
Firebase Authentication 	Authentifizierungs-Datenbank	Um hohe Sicherheitsstandards einzuhalten, verwendet die Webseite ein tokenbasiertes Authentifizierungssystem. Firebase Auth. ist eine vom Unternehmen Google zur Verfügung gestellte Authentifizierungsschnittstelle, die jedem Benutzer einen individuellen Token zuweist. Diese Tokens werden dann im lokalen Browserspeicher des jeweiligen Benutzers (mithilfe vom einem SSL Skript) abgelegt.
MongoDB 	Datenverwaltung	MongoDB ist eine sogenannte NoSQL Datenbank. Die interne Struktur ist somit nicht durch statische Schlüsselwerte bestimmt und die hinterlegten Objekte sind in Ihrer Wertzuweisung flexibel. Dadurch kann die Datenverarbeitung der Webseite stetig angepasst werden, ohne den internen Datenbankaufbau ändern zu müssen.
Flask 	Schnittstellen-Programmierung (API)	Flask ist eine Technologie zum Programmieren von Systemschnittstellen (APIs) und basiert auf REST. Im Rahmen dieses Softwareprojekts wurden alle APIs, die der Abbildung 2-1 zu entnehmen sind, mit Flask programmiert.

2.2 Architektur

Die UML (Unified Modelling Language) ist eine Modellierungssprache, bzw. eine Familie von Notationen und Diagrammtypen, die dazu verwendet wird, die interne Struktur von Systemen abzubilden [Fow04]. Zu dieser Familie von Diagrammtypen gehört auch das Anwendungsfalldiagramm (auf Englisch Use-Case-Diagramm). Die im Rahmen einer Architekturanalyse erstellten Anwendungsfalldiagramme, zeigen das nach außen sichtbare Verhalten des Systems bzw. der einzelnen Systemkomponenten. Dadurch können die Kommunikationswege zwischen den einzelnen Subsystemen besser nachvollzogen werden. [Rup12]

Der Abbildung 2-1 ist die Architektur der programmierten Webseite als Anwendungsfalldiagramm zu entnehmen. MVC (Modal-View-Controller) ist eine Dreischichtenarchitektur im Bereich der Webprogrammierung, bei der die Daten (Modal) durch gesicherte Schnittstellen (Controller) verwaltet und dem Benutzer gesondert präsentiert werden (View). [Kal14]

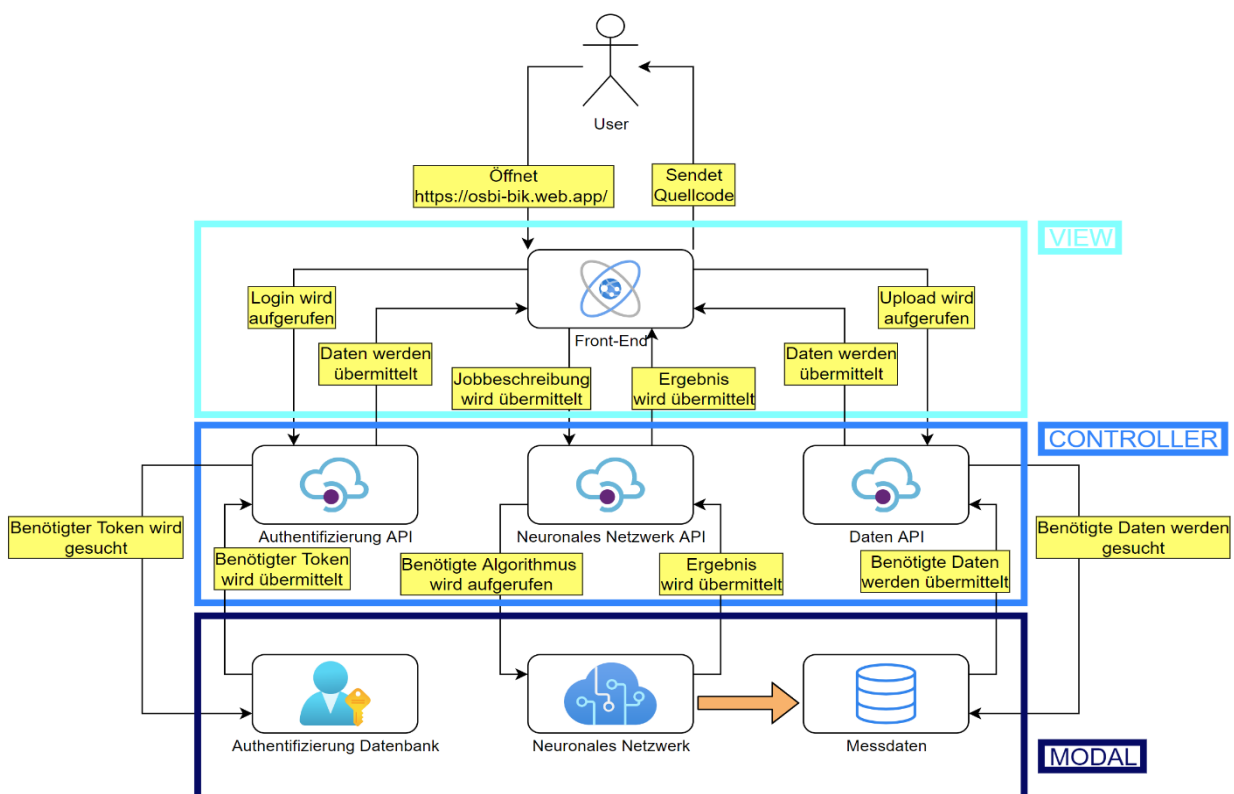


Abbildung 2-1 Webseite Architektur

2.3 Versions-Management

Unter dem Begriff Versions-Management, versteht man die Verwaltung von Quellcodeänderungen, bzw. den Vergleich zwischen unterschiedlichen Quellcodezuständen. Dadurch ist es möglich frühere Stände des Quellcodes wieder herzustellen, wenn z.B. ein Softwarefehler in der neusten Version entdeckt wurde. Außerdem wird die Transparenz des Entwicklungsprozesses erhöht, indem zusätzlich festgehalten wird, wann und von wem die Änderungen gemacht wurden. Softwaretechnikern ist es dann auf Grundlage dieser Daten möglich verschiedenste Metriken zu erheben, um somit den Entwicklungsprozess optimieren zu können. Eine weitere Grundfunktion des Versions-Managements ist das Branch-and-Merge-System. Durch die Möglichkeit Quellcodezustände zu vergleichen können beliebig viele Programmierer an unterschiedlichen Zuständen (Branches) bzw. Dateien arbeiten und dann Ihre Änderungen zusammenfügen (mergen). Dieses Vorgehen ist der Abbildung 2-2 zu entnehmen, bei der 2 Programmierer an unterschiedlichen Zuständen arbeiten und Ihre Änderungen dann zusammenfügen. [Aug18]

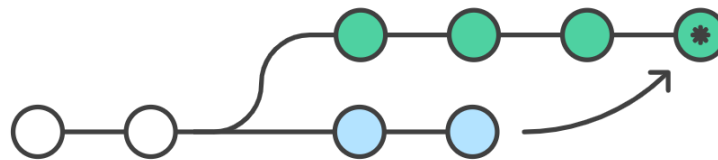


Abbildung 2-2 Branch-and-Merge-System

Quelle: <https://cdn-media-1.freecodecamp.org/images/VonhijTBQgjwtRXz31wLzF7iWDnDFk2o8EWi>

Um Versions-Management zu betreiben, benötigt man ein Versionskontrollsystem (auf Englisch Version-Control-System). Im Rahmen dieser Projektarbeit wird Git in Kombination mit GitHub als solches System verwendet. Dabei ist Git das eigentliche Versionskontrollsystem und GitHub ein Cloud Anbieter zur Verwaltung eines solchen Datenbestandes. Dieser Datenbestand ist öffentlich zugänglich (Open Source) und kann unter folgender URL aufgerufen werden:

<https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation>

Um diese grundlegend qualitätssichernden Aspekte des Versions-Managements zu erweitern, kann die Branchstruktur optimiert werden. Die Branchstruktur, in diesem Softwareprojekt, besteht aus einem Hauptbranch (Master) und zwei hierarchisch angeordneten Testumgebungen (Staging 1 und 2), welches der Abbildung 2-3 zu entnehmen ist. Im Kapitel 2.4 wird dann die Interaktion bzw. Kommunikation zwischen diesen Branches, im Kontext der Integrationsautomatisierung, thematisiert.

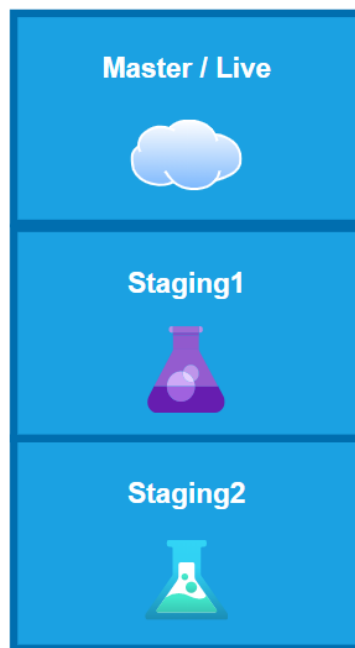


Abbildung 2-3 Branchstruktur

Die hier erwähnte Branchstruktur (des Softwareprojekts), kann unter folgendem Link aufgerufen und konfiguriert werden:

<https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/branches>

2.4 Integrationsautomatisierung (CI/CD)

Unter Integrationsautomatisierung versteht man die Überwachung des kompletten Software-Lebenszyklus. Diese besteht aus zwei Kernkonzepten, nämlich der kontinuierlichen Integration (Continuous Integration) und der kontinuierlichen Serveraktualisierung (Continuous Deployment). Bei der kontinuierlichen Integration (CI) wird nach jeder bestätigten Codeänderung und jedem Merge ein branchabhängiges Skript ausgeführt, welche unter dem folgenden Link aufgerufen werden können:

[https://github.com/Web-App-zur-](https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/tree/main/.github/workflows)

[Einzelblattmontagen/offshoreSingleBladeInstallation/tree/main/.github/workflows](https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/tree/main/.github/workflows)

Die Skripte bestehen aus mehreren Arbeitspaketen (Tasks) die in chronologischer Reihenfolge abgearbeitet werden. Der Abbildung 2-4 sind die Tasks der drei Skripte zu entnehmen:

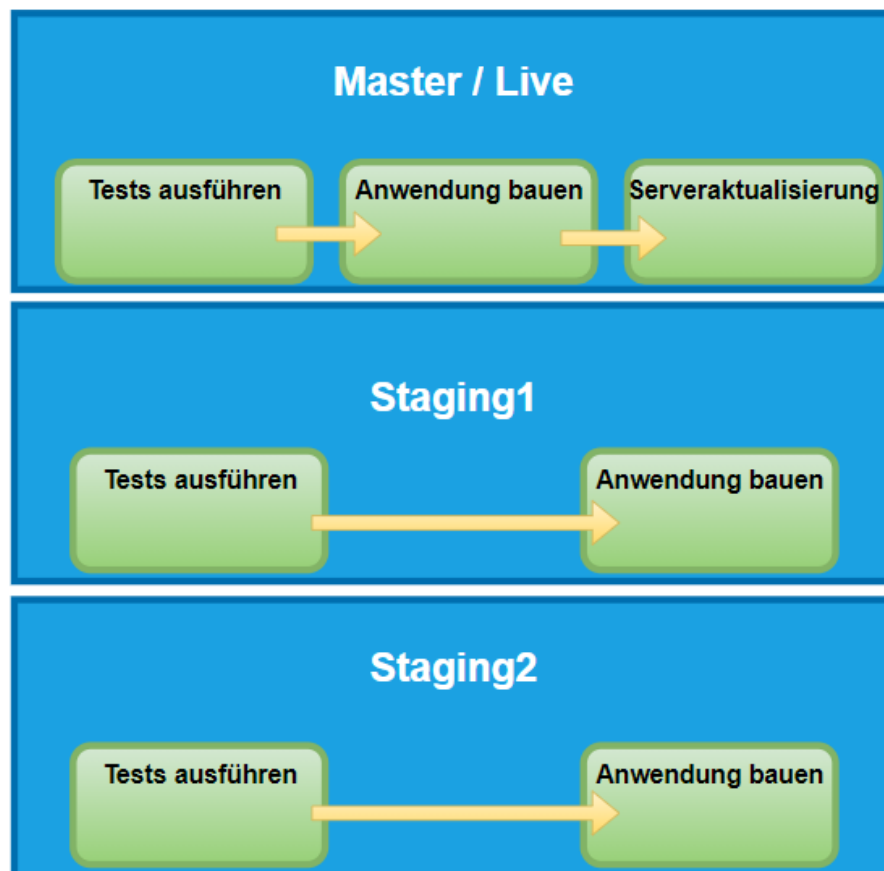


Abbildung 2-4 CI/CD Taskübersicht

Dieses Konzept stellt sicher, dass die Korrektheit jeder isolierten Softwarekomponente gegeben ist, da alle programmierten Tests (bei der Skriptausführung) ausgeführt werden und die Anwendung gebaut wird, um syntaktische Fehler auszuschließen. Die korrekte Interaktion zwischen einzelnen Komponenten wird dadurch aber nicht gewährleistet, weshalb Integrationstests notwendig sind. Diese Integrationstests werden auf den beiden Testumgebungen in explorativer Form durchgeführt und ggf. freigegeben oder abgelehnt. Wenn durch die Integrationstests keine Fehler gefunden wurden und die Codeänderung freigegeben wurde, wird sie auf den darüberliegenden Branch gemerged. Falls bei den Integrationstests Fehler gefunden wurden, wird die Codeänderung auf den darunterliegenden Branch gemerged und der Fehler kann vom Programmierer behoben werden. Dieser Prozess kann der Abbildung 2-5 entnommen werden.

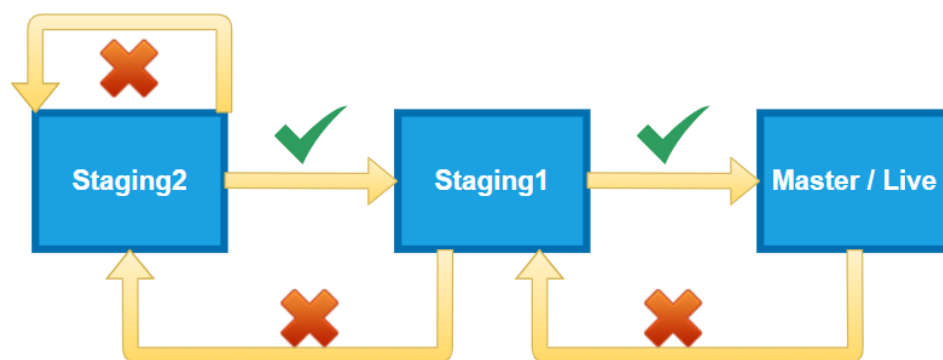


Abbildung 2-5 Software-Lebenszyklus

Die kontinuierliche Serveraktualisierung (CD) soll sicherstellen, dass die öffentlich erreichbare Webseite aktuell ist und dem letzten Stand des Master-Banches entspricht, weshalb diese nur als Task im Master-Skript zu finden ist.

2.5 Webseite Funktionalität

Die im Rahmen dieser Projektarbeit programmierte Webseite ist unter folgender URL erreichbar:

<https://osbi-bik.web.app/>

Initial wird einem die Landing-Page angezeigt, welche die Funktionalität der Webseite zusammenfasst und einige Features (wie z.B. das SSL Zertifikat) hervorhebt. Nachdem der Login Prozess erfolgreich durchgeführt wurde, wird das Dashboard angezeigt. Der Aufbau des Dashboards ist der Abbildung 2-6 zu entnehmen.

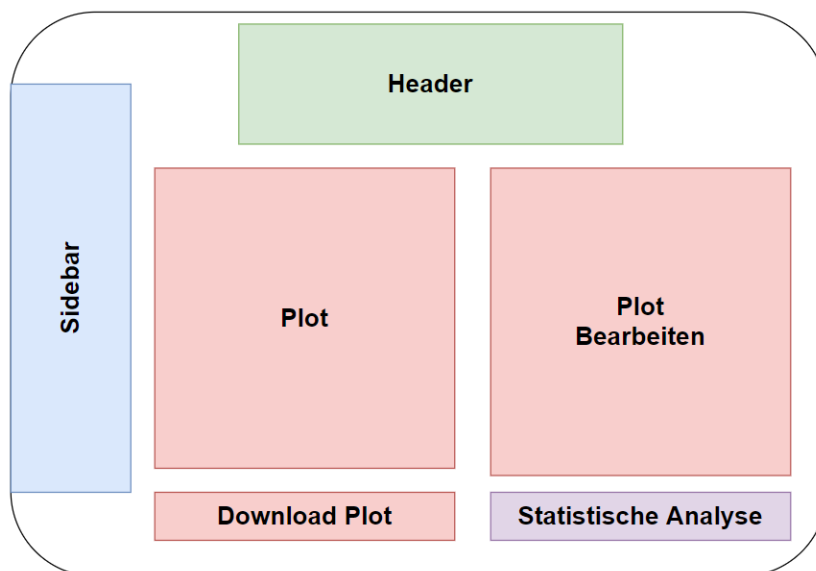


Abbildung 2-6 Webseite Aufbau

Im Header kann man die zu analysierenden Daten hochladen oder Testdaten herunterladen, um die Funktionalität des Dashboards zu testen. Auf der linken Seite befindet sich die Sidebar, welche die Hilfesektion enthält und unter anderem zur öffentlich zugänglichen Dokumentation weiterleitet. Im Zentrum des Dashboards befindet sich der aus den hochgeladenen Daten erzeugte Plot, der individuell bearbeitet und heruntergeladen werden kann. Außerdem befindet sich in der unteren rechten Ecke des Dashboards eine Sektion, in der die wichtigsten statistischen Daten zusammengefasst sind. Falls man eine detaillierte Auswertung des Datensatzes wünscht, kann man sich einen Report in PDF-Format herunterladen.

3 Maschinelles Lernen

3.1 Arbeitsprozess / Planung und Anforderungen

Unsere Aufgabe war der Aufbau mehrerer Analysemodelle zur Erkennung von Zusammenhängen in der Offshore Montage von Rotorblättern an Windkraftanlagen (unter Einbezug von Wetterdaten).

Im Rahmen dieses Projekts wurde kein explizites Ziel als Ergebnis der neuronalen Netze gesetzt, da noch nicht absehbar war, ob dabei sinnvolle Schlüsse gezogen werden können. Aufgrund dessen fallen supervised Methoden aus der Auswahl, bei denen man im Backpropagation des Netzes die Gewichte immer mehr an die gewünschte Ausgabe anpasst.

Im Allgemeinen werden die zur Verfügung gestellten Dateien so bearbeitet, dass kohärente Datensätze mit gleichen Zeitstempeln entstehen. Erst durch das Preprocessing lassen sich die geforderten ML-Algorithmen implementieren. Hierbei sind wir stark eingeschränkt durch teilweise differente Messdaten und die Art der Messungen. Wenn man nun die Wetterdaten mit einbringt, könnte man feststellen, ob es Zusammenhänge bei den Veränderungen gibt (z. B. wenn der Wert des Lidars oder der Boje bei zwei unterschiedlichen Installationen ähnlich steigt, ob sich das auch annähernd gleich in den Schwingungen des Windrades zeigt.).

Unser Ziel ist es die Installationsdaten und die dazugehörigen Wetterdaten zu analysieren und mit Hilfe von Neuronalen Netzalgorithmen Korrelationen zwischen den verschiedenen Montagen der Rotorblätter zu finden. Somit könne man beispielsweise in Folgeprojekten bei bestimmten Wettersituationen einen geeigneten Installationszeitraum finden und eine Prognose über die Dauer der Montage abgeben.

3.2 Recherchephase

Das Thema maschinelles Lernen, insbesondere das Benutzen von Neuronalen Netzen, war für uns alle ein neues Thema, somit galt es sich im ersten Schritt erstmal mit dem Thema vertraut zu machen. Bei der Recherche haben wir uns mehrere Quellen herangezogen von Literatur bis hin zu Lehrvideos über das gesamte Themengebiet. Zudem galt es sich auch die Skriptsprache Python anzueignen. Das haben wir vor allem durch Tutorials auf YouTube und diesbezügliche Dokumentationen getan.

Wichtig war auch sich schonmal mit den Daten vertraut zu machen und diese in verschiedenen Plots sinnvoll darzustellen. So konnte man sehen in welchem Bereich sich die Schwankungen der Windräder befanden. Zudem traten auch ein paar Konvertierungsprobleme mit den Messwerten auf, die man somit schnell beseitigen konnte. Einige Beispiele der Daten sind der Abbildung 3-1 zu entnehmen.

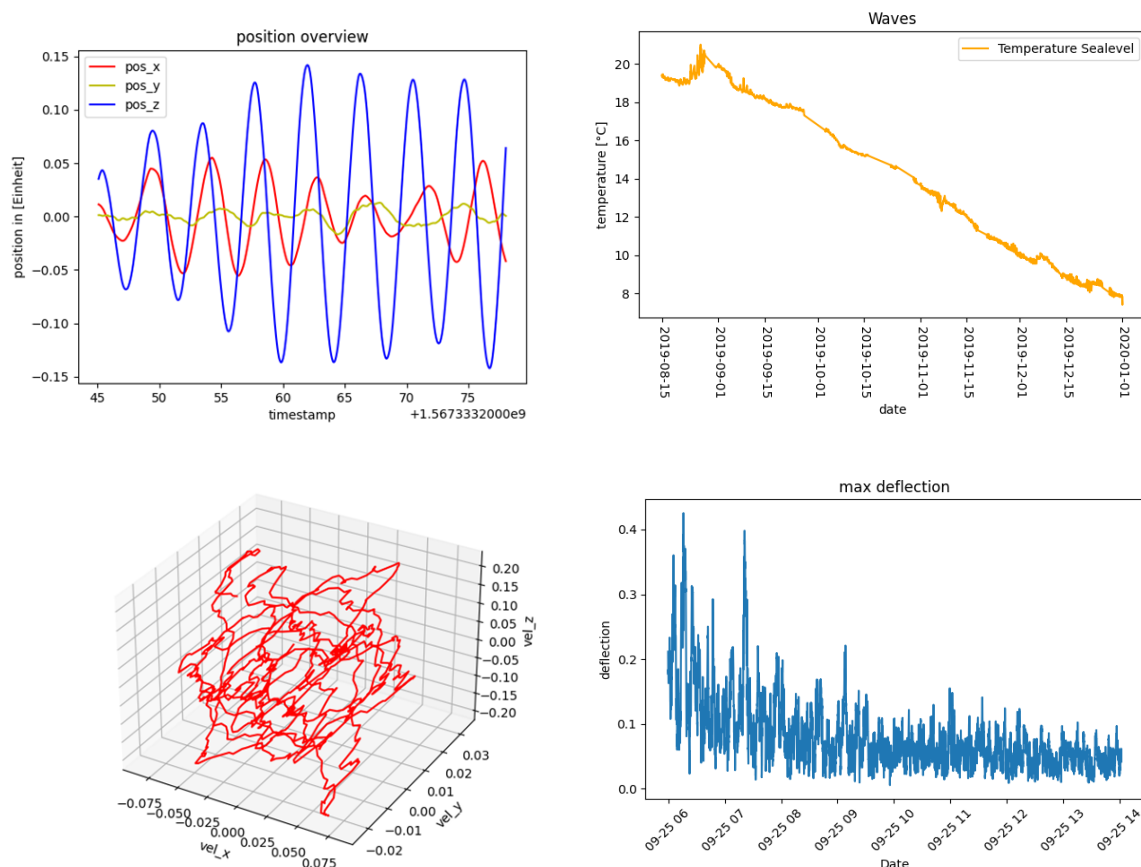


Abbildung 3-1 Plots aus der Einarbeitungsphase

3.3 Technologieauswahl

Wie in den Projektanforderungen eingangs gefordert, wurde die gesamte Datenanalyse und Interpretation in Python umgesetzt. Python ist aktuell die meistgenutzte Programmiersprache zur Datenanalyse und maschinellen Lernen. Bibliotheken wie Numpy (Array-Daten), Pandas (vergleichbar zu Excel) und Sklearn (Standardbibliothek für ML) sind für viele Programmierer gängiges Werkzeug bei der Arbeit mit Datensätzen.

Tabelle 3-1 Entscheidungsmatrix Backend Fortsetzung

Backend						
Technologien		Usability	Konnektivität	Teaminterne Erfahrung	Geschwindigkeit	Debugging
Machine Learning	Keras	Green	Green	Red	Yellow	Green
	Tensorflow	Red	Green	Red	Green	Red
	Pytorch	Red	Green	Red	Green	Yellow

Bei der Entwicklung der ML-Algorithmen haben wir uns für das Framework Keras entschieden. Dieses von einem Google-Ingenieur implementierte und frei verfügbare Framework dient der Datenanalyse und Bereitstellung von Methoden zur Implementierung von Neuronalen Netzen aller Art. Der Vorteil von Keras ist, dass es über Tensorflow laufen kann und somit noch umfangreicher ist, ohne jedoch die komplexeren Funktionen und Architektur von Tensorflow zu beinhalten. Keras ist einfach in der Implementierung und ist besonders leicht verständlich, was dem Nutzer ein schnelles Prototyping von Modellen ermöglicht.

Als IDE haben wir uns für PyCharm (Jet Brains) entschieden, das als Editor übersichtlich und kostenlos für Studentische Zwecke zur Verfügung steht. Diese Entscheidung wurde begünstigt durch unsere Erfahrungen mit dem gleich aufgebauten IntelliJ von Jet Brains, durch frühere Kurse zum Programmieren in Java.

3.4 Datensatz und Preprocessing

Wir arbeiten auf einem großen Datensatz, welcher aus verschiedenen CSV Dateien mit unterschiedlichen Headern (Spaltennamen) besteht. Weiter ist aufgrund der Größe der Daten (mehrere Millionen Zeilen) die benötigte Rechenkapazität hoch. Es ist daher unmöglich die Daten in einer einzigen Datei zusammenzufassen und sequenziell zu verarbeiten. Dies wäre die einfachste, aber auch rechenintensivste Methode.

Die Vorbereitung der Daten bildet bei solchen Prozessen eine bedeutende Rolle, da wir es in unserem Fall mit vielen verschiedenen Werten zu tun hatten (Beschleunigung, Geschwindigkeit und Position in x, y und z-Richtung uvm.) ist eine Festlegung auf eine Reihenfolge der einzelnen Messwerte innerhalb der CSV-Datei somit essenziell.

Die Messreihen der Installationen haben zudem alle unterschiedliche Zeitstempel- und Dauer, deshalb wäre es sinnvoll diese auf eine bestimmte Länge zu minimieren, um somit die Verläufe der einzelnen Montagen besser vergleichen zu können.

Messdaten zur Installation der Einzelblätter und Turbinen unterteilen wir in die Sektionen Telemetrie- und Geometriedaten. Die einzelnen Features, aufgeteilt nach Dateityp sind der Tabelle 3-2 zu entnehmen.

Tabelle 3-2 Features der unterschiedlichen Datensätze

Dateityp	Header	Frequenz
Telemetrie	epoch,acc_x,acc_y,acc_z,vel_x,vel_y,vel_z,pos_x,pos_y,pos_z,deflection	3 Hz /dreimal pro Sekunde
Geometrie	epoch,max_deflection,max_deflection_i, ddt_max_deflection,axis_ratio,eccentricity,ddt_axis_ratio,ddt_eccentricity,axis_angle_signed,axis_angle_unsigned,axis_azimuth,ddt_axis_angle_signed,ddt_axis_angle_unsigned,p2p_angle_unsigned,p2p_angle_signed,p2p_azimuth,p2p_azimuth_unwrapped,ddt_p2p_azimuth_unwrapped,ddt_p2p_azimuth,ddt_p2p_angle_unsigned,ddt_p2p_angle_signed	0,33 Hz/alle drei Sekunden
WMB/Wellen	epoch, Tp, Dirp, Sprp, Tz, Hm0, Tl, T1, Tc, Tdw2, Tdw1, Tpc, nu, eps, QP, Ss, TRef, TSea, Bat, Percentage, Hmax, Tmax, H(1/10), T(1/10), H(1/3), T(1/3), Hav, Tav, Eps, #Waves	alle 30 Minuten

Lidar/Wind	epoch, wind_speed_0, wind_dir_0, wind_dir_0_corr, height_0, wind_speed_1, wind_dir_1, wind_dir_1_corr, height_1, wind_speed_2, wind_dir_2, wind_dir_2_corr, height_2, wind_speed_3, wind_dir_3, wind_dir_3_corr, height_3, wind_speed_4, wind_dir_4, wind_dir_4_corr, height_4, wind_speed_5, wind_dir_5, wind_dir_5_corr, height_5, wind_speed_6, wind_dir_6, wind_dir_6_corr, height_6, wind_speed_7, wind_dir_7, wind_dir_7_corr, height_7, wind_speed_8, wind_dir_8, wind_dir_8_corr, height_8, wind_speed_9, wind_dir_9, wind_dir_9_corr, height_9, wind_speed_10, wind_dir_10, wind_dir_10_corr, height_10, heading	1 Hz/ jede Sekunde
-------------------	--	--------------------

Im Falle, dass alle Dateitypen vorhanden sind, arbeiten wir in einem Feature Raum von 106 Features, was eine große Datenmenge mit teilweise redundanten Informationen ist. Zum Reduzieren der Datenmenge und der verbundenen Rechenleistung haben wir uns auf eine manuelle Reduktion der Features geeinigt. In Absprache mit den Betreuern einigten wir uns auf bestimmte Features, die in den folgenden Unterkapiteln näher erläutert werden.

3.4.1 Turbinen bzw. Geometrie Daten

Die ausgewählten Werte von den Turbinen-Daten beziehen sich auf das Schwingungsverhalten während der Einzelblattmontage. Dabei wird nach jeden 3 Sekunden eine Ellipse in den Positionsverlauf der Turbine modelliert. Die wichtigen Parameter sind hierbei die Semi Major Axis und die Semi Minor Axis (siehe Abbildung 3-2).

Tabelle 3-3 Ausgewählte Features der Geometriedaten

Feature Bezeichnung	Beschreibung
max deflection	Dieses Feature steht für den Betrag der Semi Major Axis und zeigt die maximale Auslenkung der letzten 3 Sekunden an.
axis ratio	Die axis ratio gibt das Verhältnis der Beträge von den beiden Parametern Semi Minor- und Semi Major Axis an. Siehe Abbildung 3-3.
p2p azimuth unwrapped	Dieses Feature vom Turbinen-Datensatz beschreibt den Winkel der maximalen Auslenkung (Semi Major Axis) an Richtung Norden orientiert. Um wiederholende Winkel-Werte zu vermeiden, steigt der gemessene Winkel-Wert kontinuierlich an und fängt nicht bei 360° wieder mit 0° an. So lässt sich ein zeitlicher Ellipsenverlauf verfolgen. Siehe Abbildung 3-2.

3.4.2 Wetterdaten

Der Wetter-Datensatz besteht aus zwei verschiedenen Datensätzen. Die Wind Daten (in der Tabelle 3-4 grau hinterlegt) wurden mit einem LIDAR-Sensor und einer Frequenz von 1Hz aufgezeichnet. Diese Daten bestehen aus den gleichen Parametern (Windgeschwindigkeit, Windrichtung an Norden orientiert und Windrichtung vom Sensor aus), bloß wurden mehrere Sensoren auf unterschiedlichen Höhen der Turbine angebracht. Hier betrachten wir nur die Werte der Sensoren aus der Installationshöhe (Nr. 3). Anders als der Wind-Datensatz sind die Wellen-Daten (in der Tabelle 3-4 grün

hinterlegt) mit einer geringen Frequenz aufgenommen worden. So gibt es erst alle 30min einen neuen Wellendaten-Eintrag.

Tabelle 3-4 Ausgewählte Features der Wetterdaten

Feature Bezeichnung	Beschreibung
Tmax	Tmax ist die Periodendauer von der höchsten gemessenen Welle in Sekunden (Hmax).
Tav	Analog zu Hav ist Tav die durchschnittliche Periodendauer aller gemessenen Wellen in Sekunden.
Hmax	Besonders ausschlaggebend für das Schwingungsverhalten sind die Wellenhöhen. Hmax liefert hierbei die Höhe der höchsten gemessenen Welle der letzten 30min in Metern.
Hav	Hav beschreibt die durchschnittliche Höhe aller zuletzt gemessenen Wellen in Zentimetern.
Tsea	Tsea gibt die Wassertemperatur der Wasseroberfläche in °C an.
Dirp	Dirp steht für directional peak und gibt die Wellenrichtung an, an der die kleinste Periodendauer bzw. die höchste Frequenz gemessen wurde.
wind_speed_3	Der Parameter wind speed 3 gibt die Windgeschwindigkeit in m/s an der Installationsturbine wieder.
wind_dir_3_corr	Damit die Windrichtung zum Beispiel mit der Wellenrichtung vergleichbar ist, haben wir uns auf die Windrichtung in Grad von Norden aus orientiert entschieden. Außerdem ist der Wert so besser menschlich zu interpretieren.

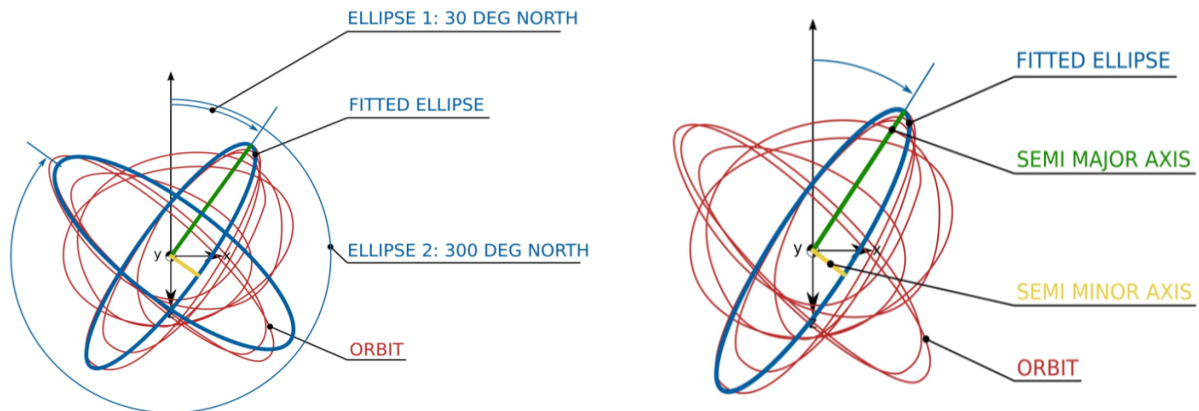


Abbildung 3-2 Turbinen Geometriedatenerklärung [San21]

$$AR = \frac{\text{SEMI MINOR AXIS}}{\text{SEMI MAJOR AXIS}}$$

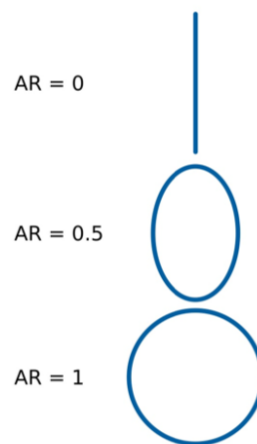


Abbildung 3-3 Beschreibung der Axis Ratio [San21]

Aufgrund der Tatsache, dass Geometriedaten nur für die Installationszustände hammerhead, tnhb1 und tnhb2 vorliegen, wurde sich hauptsächlich auf diese konzentriert. Durch die geringe Größe von tnhb1 und tnhb2 wurden diese nicht weiter analysiert. Der hieraus resultierende Datensatz, innerhalb des Codes als 'hammerhead_reduced.csv' genannt, weist folgende Merkmale auf, die der Abbildung 3-4 entnommen werden können.

	max_deflection	axis_ratio	p2p_azimuth_unwrapped	wind_speed_3	wind_dir_3_corr	Dirp	TSea	Hmax	Tmax	Hav	Tav
count	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000	429356.000000
mean	0.137587	0.296192	6887.775005	7.062412	190.188618	0.149201	0.010190	0.001762	0.007179	0.060135	0.002928
std	0.065815	0.202449	7391.040637	4.252998	104.665088	0.036730	0.001179	0.000892	0.008162	0.024558	0.000500
min	0.000000	0.000000	-6337.188697	0.000000	0.000000	0.000000	0.005944	0.000433	0.001683	0.017289	0.001800
25%	0.090705	0.125596	-38.207635	4.900000	144.620000	0.142944	0.010361	0.001156	0.003128	0.043544	0.002600
50%	0.135403	0.271322	7534.529872	8.200000	250.606667	0.155444	0.010500	0.001489	0.003694	0.054906	0.002911
75%	0.169012	0.419584	14520.945143	9.566667	267.615000	0.170333	0.010889	0.002178	0.004850	0.078817	0.003294
max	0.852206	6.713549	23045.158420	21.400000	314.833333	0.199222	0.011056	0.005544	0.040089	0.130783	0.004339

Abbildung 3-4 hammerhead_reduced.csv Merkmale

3.4.3 Feature Übersicht und Datenqualität

Tabelle 3-5 Qualität unterschiedlicher Datensätze

Verschiedene Möglichkeiten					
Kombination Nr/Qualität.	WMB	LIDAR	GEOMETRY	Feature Anzahl	mit Primärschlüssel 'epoch'
1			X	4	4
2		X		3	3
3		X	X	7	6
4	X			7	8
5	X		X	11	10
6	X	X		10	11
7	X	X	X	14	12

Der gezeigten Tabelle 3-5 ist zu entnehmen, dass insgesamt 24 Datensätze unterschiedlicher Qualitäten vorliegen. Glücklicherweise ist die Mehrzahl der Datensätze von Qualität 07 mit einer Anzahl von 21. Im Folgenden werden wir uns in der Analyse überwiegend mit den vollständigen Datensätzen beschäftigen, da diese die höchste Informationsdichte besitzen und für die Erkenntnisgewinnung am vielversprechendsten sind.

Tabelle 3-6 Form und Qualität der Datensätze

Dateiname	Form	Qualität
hammerhead_turbine04.csv	(169174, 12)	7
hammerhead_turbine05.csv	(272178, 10)	5
hammerhead_turbine06.csv	(10620, 12)	7
hammerhead_turbine07.csv	(29251, 12)	7
hammerhead_turbine08.csv	(26692, 12)	7
hammerhead_turbine09.csv	(101626, 10)	5
hammerhead_turbine10.csv	(9786, 12)	7
hammerhead_turbine11.csv	(7662, 6)	3
hammerhead_turbine12.csv	(6331, 12)	7
hammerhead_turbine13.csv	(29527, 12)	7
hammerhead_turbine14.csv	(5493, 12)	7
tnhb1_turbine04.csv	(10993, 12)	7
tnhb1_turbine05.csv	(5165, 10)	5
tnhb1_turbine06.csv	(10567, 12)	7
tnhb1_turbine07.csv	(5611, 12)	7
tnhb1_turbine08.csv	(8222, 12)	7
tnhb1_turbine11.csv	(3128, 6)	3
tnhb1_turbine12.csv	(51168, 12)	7
tnhb1_turbine13.csv	(163136, 12)	7
tnhb1_turbine14.csv	(4152, 12)	7
tnhb1_turbine16.csv	(146132, 12)	7
tnhb2_turbine05.csv	(4061, 10)	5
tnhb2_turbine06.csv	(4138, 12)	7
tnhb2_turbine07.csv	(4318, 12)	7

3.4.4 Preprocessing

Das Vorverarbeiten der Daten (Preprocessing) ist der wichtigste Schritt für das aufbauen eines Maschinellen Modells. Deshalb wurde als erstes, für mehr Übersicht im Datensatz, die Daten in verschiedene Listen sog. Globs zusammengefügt. Anschließend wurden auch alle vorhandenen Wetterdaten nach den Monaten in Listen zusammengeführt. Nun folgte das Aussortieren der überflüssigen Daten. Dafür haben wir uns als erstes die Zeiträume des gesamten Datensatzes der Installationen notiert. Das wurde dann mit den Zeiträumen der vorhandenen Wetterdaten abgeglichen. Für diese gemeinsamen Zeiträume haben wir ein Skript geschrieben, welches die Wetterdaten an den Installationstagen plottet. Unseren Betreuern war es wichtig nur die Daten zu betrachten, an denen relativ konstante Wetterbedingungen herrschten. Somit wurden alle Plots der Wetterdaten manuell von uns aussortiert. Die Aussortierung orientierte sich besonders stark an den gemessenen Wellenhöhen, Wellenrichtungen und den Windgeschwindigkeiten. Als konstant wurde von uns eine maximale Wellenhöhendifferenz von 25cm, eine maximale Wellenrichtungsdifferenz von 50° und keine stark schwankende Windgeschwindigkeit erachtet. In Abbildung 3-5 ist ein Beispiel für die Einteilung konstanter Wetterzeiträume aufgeführt. Dabei bilden die sich überschneidenden Zeiträume den zum Schluss ausgewählten Datensatz.

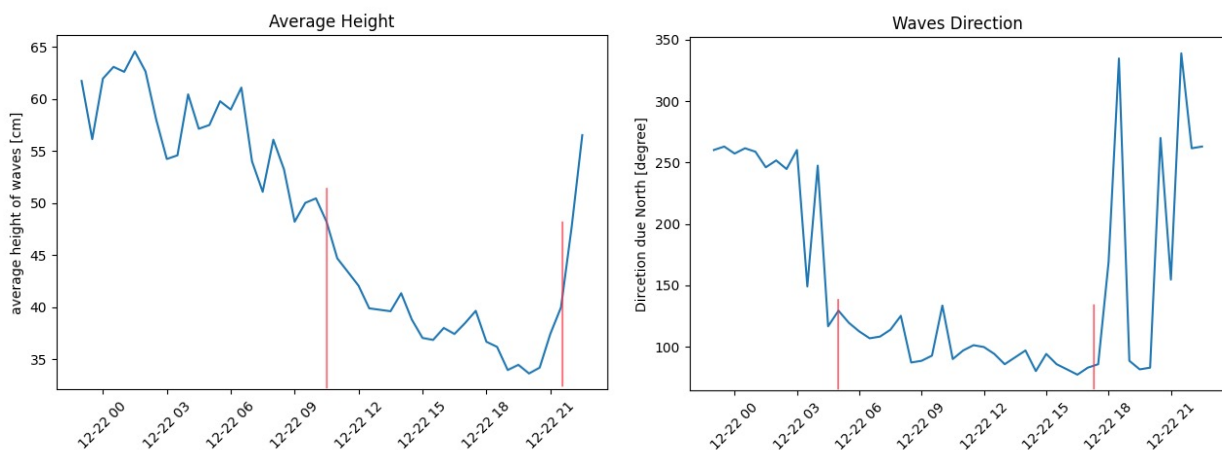
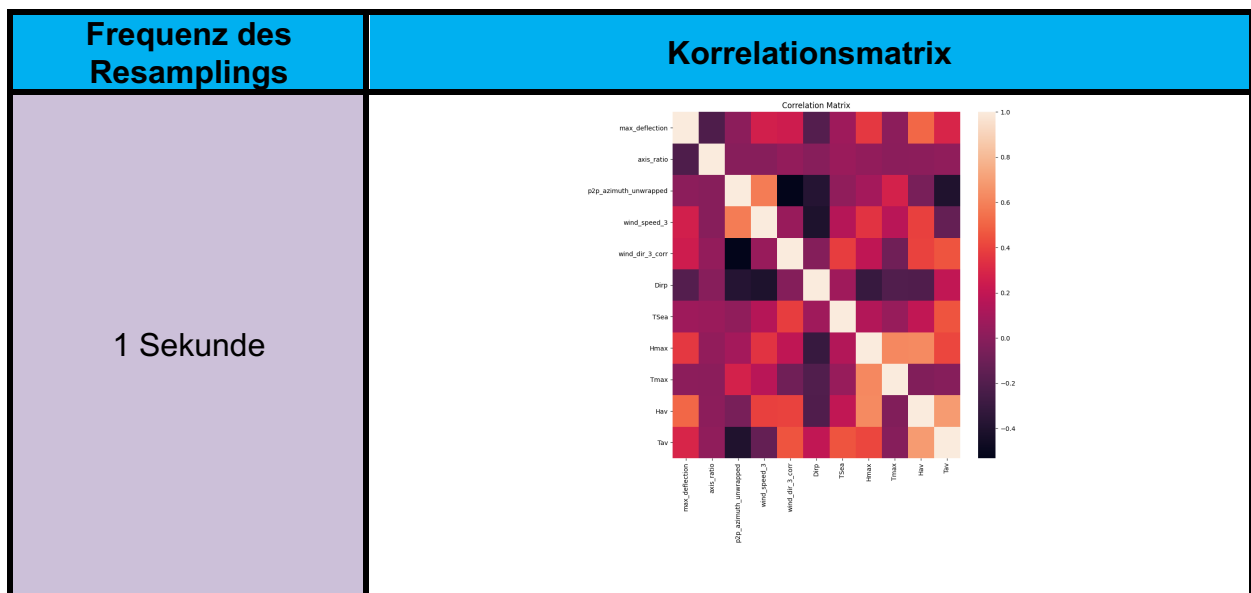


Abbildung 3-5 Beispiel konstanter Wetterdaten

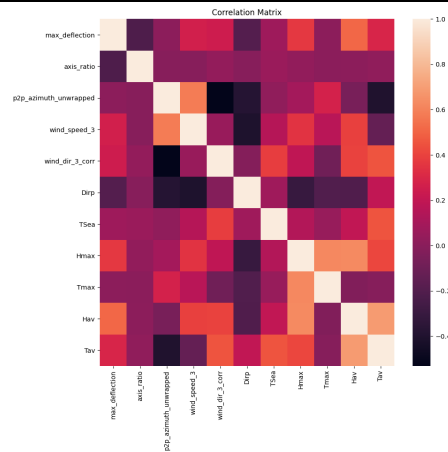
3.5 Resampling der Daten und Korrelationsmatrix

Nachdem die bereitgestellten Daten auf die tatsächlich interessanten und relevanten Zeiträume gekürzt und zusammengefügt wurden, folgte das Ermitteln von linearen Zusammenhängen zwischen den elf Features. Die in der Tabelle 3-7 dargestellten Korrelationsmatrizen nach Pearson R berechnen diesen Zusammenhang zwischen den einzelnen Features. Begründet durch die unterschiedlichen Frequenzen der Wellen-, Wind-, Telemetrie- und Geometriedaten (siehe Abschnitt Datensatz), wurden diese auf die gemeinsame Frequenz von 3 Sekunden resampled und zusammengefügt. Dabei entstehen vor allem bei den Wmb-Datensätzen, welche eine Frequenz von 30 Minuten haben, viele redundante Spalten mit gleichen Informationen. Deshalb wird untersucht, inwiefern die Samplingfrequenz Auswirkungen auf die Korrelationsmatrix hat.

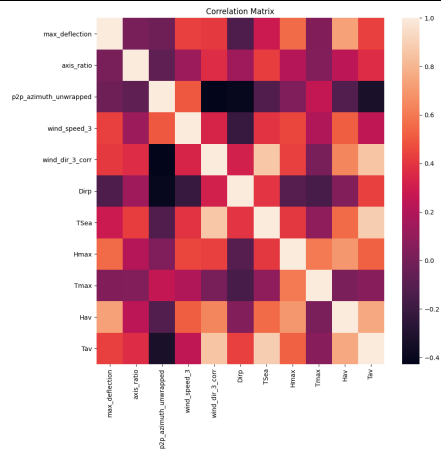
Tabelle 3-7 Korrelationsmatrizen für verschiedene Samplingfrequenzen



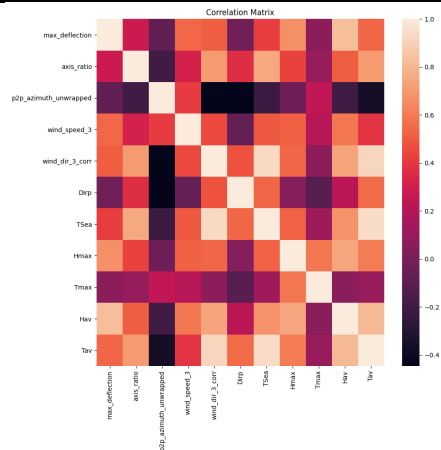
1 Minute

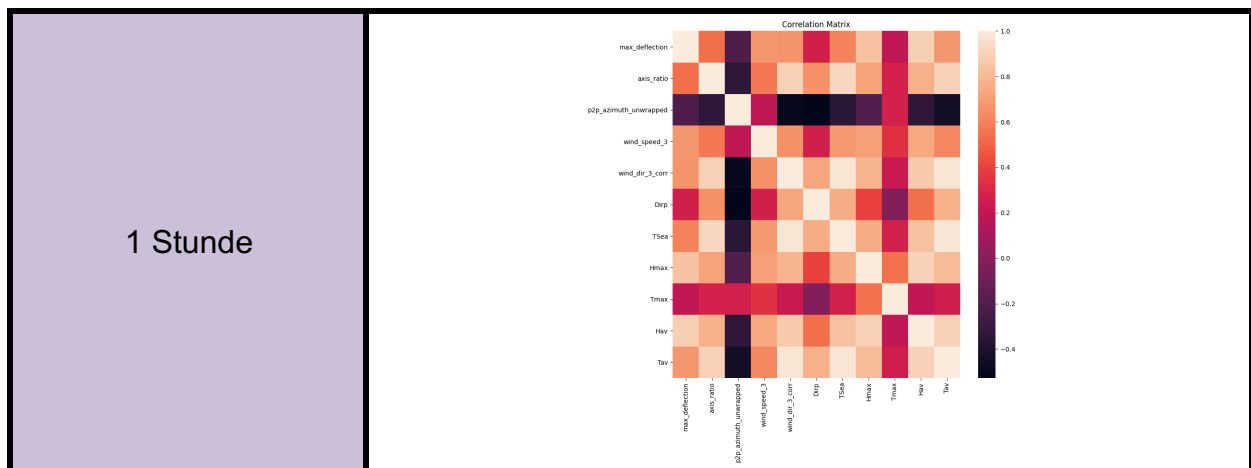


10 Minuten



30 Minuten





Wie zu erwarten, gehen die Pearson-Korrelationsfaktoren gegen 1, was durch die heller werdende Färbung der Korrelationsmatrix begründet ist. Diese Tendenz zur Proportionalität ist jedoch größtenteils nur in den Korrelationsmatrizen mit kleineren Samplingfrequenzen eindeutig zu erkennen. Für einen Überblick zu den ermittelten Abhängigkeiten der Daten untereinander, haben wir 55 Korrelationsmatrizen, die alle oben aufgeführten Samplingraten enthalten, manuell zu dem Geometrie-Datensatz aller vorhandenen Turbinen ausgewertet. Das Augenmerk lag dabei darauf, inwiefern die Wetterdaten tendenziell zu den Werten max deflection, axis ratio und p2p azimuth unwrapped stehen.

Bei der Auswertung haben wir uns intern auf drei Kategorien geeinigt: proportional, antiproportional und kein linearer Zusammenhang. Da die Pearson-Korrelationsfaktoren einen Wertebereich von -1 bis 1 besitzen haben wir die Grenzen für keinen linearen Zusammenhang von -0.33 bis 0.33 gesetzt, um drei gleich große Wertebereiche zu erhalten. Alle höheren, bzw. niedrigeren Pearson-Korrelationsfaktoren sind von uns als proportional, bzw. antiproportional interpretiert worden. Unten sind die Ergebnisse von der Auswertung der Korrelationsmatrizen in den Tabellen 3-8 bis 3-10 aufgeführt.

Tabelle 3-8 Verhältnis der Wetterdaten zum Parameter max deflection

Proportionales Verhältnis	Antiproportionales Verhältnis
Hav	-
wind speed	-
Tav	-
Hmax	-
TSea	-

Tabelle 3-9 Verhältnis der Wetterdaten zum Parameter axis ratio

Proportionales Verhältnis	Antiproportionales Verhältnis
wind dir corr	Hav
Dirp	wind speed
Tav	-
Hmax	-

Tabelle 3-10 Verhältnis der Wetterdaten zum Parameter p2p azimuth unwrapped

Proportionales Verhältnis	Antiproportionales Verhältnis
Tav	wind dir corr
Hmax	Hav
Tmax	wind speed
-	Dirp

Auffällig ist, dass keine Wetterparameter antiproportional zu der max deflection sind. Somit scheinen alle Wetterparameter proportionalen oder keinen linearen Einfluss auf die maximale Auslenkung während der Installation zu haben. Den größten proportionalen Einfluss auf die maximale Auslenkung besitzen die durchschnittliche Wellenhöhe (Hav), die Windgeschwindigkeit (wind speed) und die durchschnittliche Periodendauer (Tav). Dass die Windgeschwindigkeit und die Wellenhöhe bei fast allen Datensätzen proportional zu der maximalen Auslenkung sind, ist nicht sehr überraschend.

Vielmehr ist überraschend, dass sich laut den Korrelationsmatrizen die Wasseroberflächentemperatur und die durchschnittliche Wellenperiode proportional zu der maximalen Auslenkung verhalten. Von der durchschnittlichen Wellenperiode lässt sich schlussfolgern, dass die Anzahl der Wellen die Auslenkung ebenfalls vergrößert.

Der Parameter axis ratio steht in direkter mathematischer Beziehung zu der maximalen Auslenkung. Es gilt: je größer die maximale Auslenkung, desto kleiner ist der Wert der axis ratio. Aufgrund dessen lässt sich erklären, dass anders als bei der max deflection, die Werte der durchschnittlichen Wellenhöhe und die Windgeschwindigkeit antiproportional zu der axis ratio stehen. Jedoch ist es widersprüchlich, dass die Parameter Hmax und Tav ein proportionales Verhältnis besitzen sollen, da diese proportional zu der max deflection stehen. Die Parameter mit der höchsten Tendenz zu einem proportionalen Verhältnis zu der axis ratio sind die Windrichtung (wind dir corr) und die Wellenrichtung mit kleinster Periodendauer (Dirp).

Die Verhältnisse zu dem fortlaufenden Ellipsenwinkel (p2p azimuth unwrapped) sind am schwierigsten zu interpretieren. Inwiefern die Windrichtung, die Wellenrichtung mit kleinster Periodendauer und die Wellenhöhe ein antiproportionales Verhältnis besitzen, ist uns nicht ersichtlich und bedarf weiterer Untersuchung.

Die Korrelationsmatrix lässt sich auf der Website für den eigenen Datensatz generieren und wird in Form eines Bildes auf dem herunterladbaren Bericht als PDF-Datei dargestellt.

3.6 Auswahl der Machine Learning Algorithmen

Innerhalb des Projekts wurden aufgrund der hohen Komplexität der Daten und der geringen Spezifikation der Ergebnisse viele Machine Learning Algorithmen ausprobiert und auf Teildatensätzen eruiert. Einige von den genannten Methoden sind der Clustering Algorithmus K-Means, künstliche Neuronale Netze zur mehrdimensionalen Regression oder Kohonen-Maps/Self-Organizing-Maps, um nur Beispiele zu nennen.

Die endgültige Wahl zur Einbettung in die Webapplikation fiel auf eine Kombination zwischen einer supervised und unsupervised Methode. Unsupervised Learning wird genutzt zum Finden von Zusammenhängen, außerhalb von Rauschen, in unbekannten Datenmengen. Die Auswertung und die Güte der Ergebnisse sind hierbei kritisch zu betrachten. Die weitaus höhere Komplexität gegenüber Supervised Learning ist ebenfalls zu unterstreichen. Gegenüber steht der Bereich des Supervised Learnings zur Abbildung von Strukturen oder Funktionen. Hierbei wird aus Eingabewerten eine Vorhersage generiert. Anders als beim Unsupervised Learning kann verifiziert werden, ob der Algorithmus richtig liegt. Beispielsweise lassen sich hierdurch künstliche Neuronale Netze bilden, welche über den Prozess der Backpropagation komplexe Zusammenhänge darstellen und wiedergeben.

Zur Beachtung der Vorgaben wurde in erster Instanz eine Clusteranalyse des bereinigten Datensatzes durchgeführt. Die ermittelten Cluster werden als Zuordnung der Datenpunkte zu einer bestimmten Klasse genutzt und das einstig unsupervised Problem zu einem supervised Problem umgeformt. Die ermittelten Klassen dienen nun als Zielgröße eines Neuronalen Netzes. Auf den gleichen Daten wird das neuronale Netz zur Klassifikation trainiert, um die Ergebnisse des Clusterings zu verifizieren. Der entscheidende Vorteil dieses Vorgehens ist ein "vier Augen" Prinzip, welches über grafische und mathematische Analyse ein Ergebnis voraussagt. Das Vorgehen ist in der Abbildung 3-6 grafisch dargestellt.

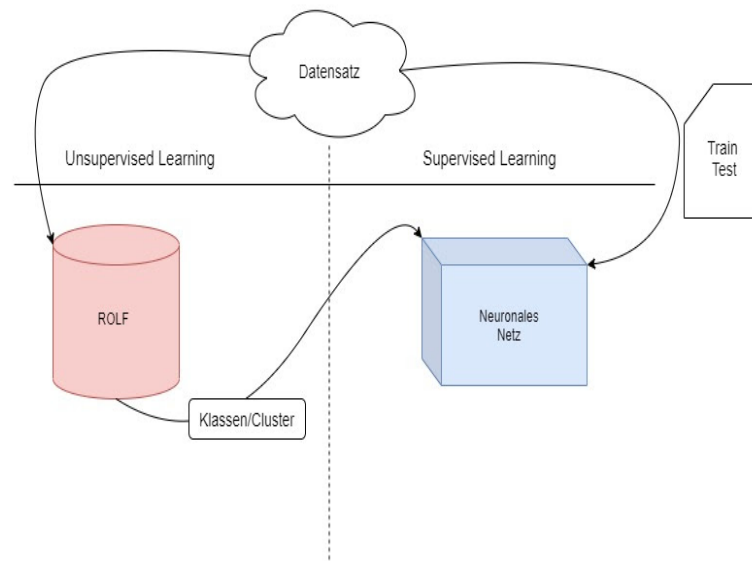


Abbildung 3-6 Von unsupervised zu supervised

Wir haben uns für zwei Algorithmen entschieden, welche in den folgenden Unterkapiteln thematisch vertieft werden.

3.6.1 ROLFs

Nach der Recherche und Einarbeitungsphase in die umfangreiche Thematik der Machine Learning Algorithmen zum Clustern, stießen wir auf eine Erweiterung des k-Means Algorithmus, der sogenannten “regional and online learnable fields” bzw. ROLFs. Dieser Algorithmus basiert auf der Grundidee der Clusteranalyse zur Ähnlichkeitsanalyse und entsprechend auch zum Finden von Korrelationen in beliebigen Räumen. Anders als der K-Means Algorithmus benötigen wir keine Norm, welche den Abstand von Punkten berechnet (beispielsweise euklidische/Manhattan usw. Norm). Viel eher *arbeitet „das Regional and Online Learnable Fields mit einer Menge K von Neuronen, die versuchen, eine Menge von Punkten durch ihre Verteilung im Eingaberaum möglichst gut abzudecken. Hierfür werden während des Trainings bei Bedarf Neuronen hinzugefügt, verschoben oder in ihrer Größe verändert“.* [Kri05]

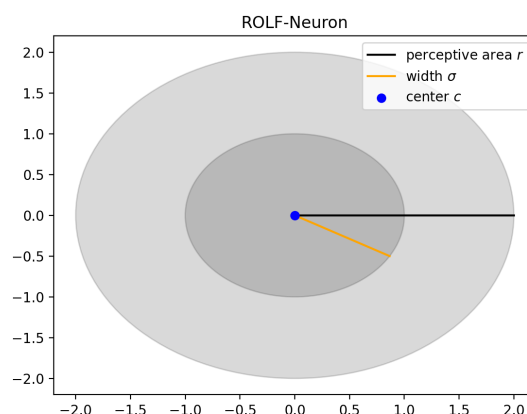


Abbildung 3-7 ROLF Neuron

Quelle: <https://towardsdatascience.com/regional-and-online-learnable-fields-cbb18b74e01e>

Ein ROLF Neuron besitzt zwei Parameter, mit welchen er eine perzeptive Fläche bildet, das Zentrum c und den Radius σ . Berechnet wird die Fläche durch die Formel $r = \sigma * \rho$, währenddessen ρ global festgelegt wird.

Der entscheidende Punkt der ROLF Anwendung ist das Trainieren des Felds. Dafür werden die Daten mit unbekannten Zusammenhängen eingefügt und von den eben erklärten Neuronen dargestellt. Hierzu dient der perzeptive Bereich, welcher für jeden Datenpunkt eingefügt wird, sofern dieser nicht in einem bereits bestehenden Bereich fällt.

Hierdurch erreicht man eine Darstellung von großen Datenmengen durch bestenfalls sehr wenige Neuronen und verringert den Speicherbedarf. Im Vergleich zum K-Means Algorithmus benötigt man keinen Hyperparameter, welcher im Vorhinein angibt, wie viele Cluster zu finden seien, sondern adaptiert diesen Parameter iterativ über den Lernprozess. In ROLFs ist ein Cluster gefunden, sobald sich zwei perzeptive Räume überschneiden. Dies ist der Abbildung 3-8 zu entnehmen.

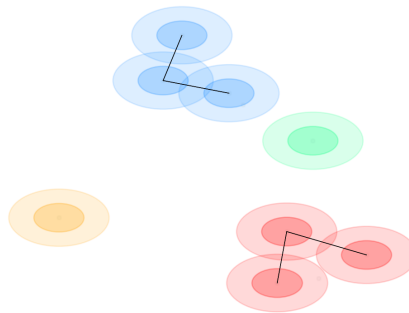


Abbildung 3-8 Clusterbildung von ROLF-Neuronen

Quelle: <https://towardsdatascience.com/regional-and-online-learnable-fields-cbb18b74e01e>

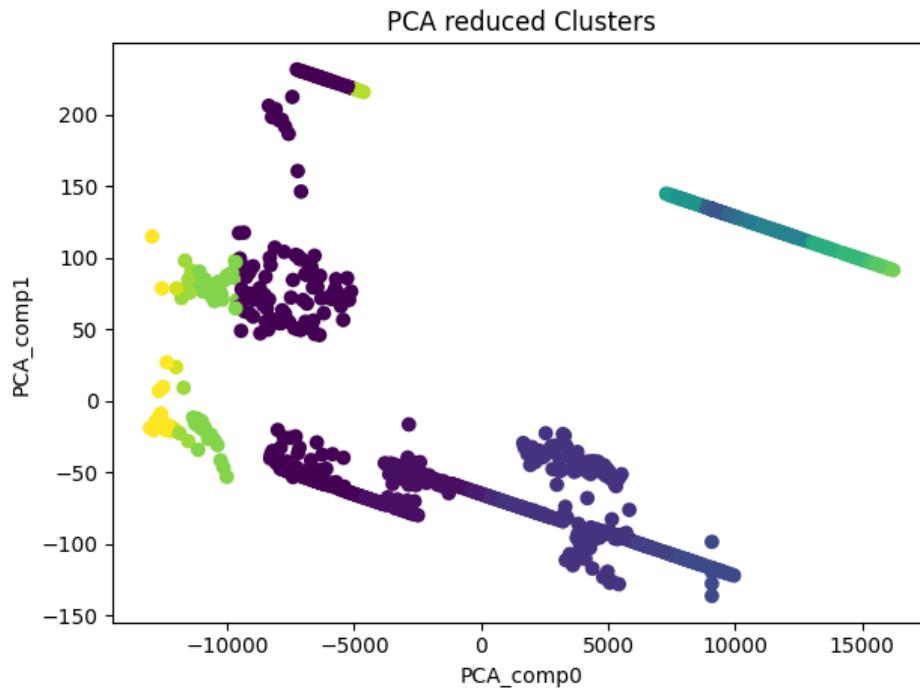
Weiter ist die Geschwindigkeit bei der Predict / Winner Methode für große Datenmengen deutlich schneller, da nicht sehr viele euklidische Distanzen berechnet, sondern durch Suchverfahren gescannt werden und der Datenpunkt in einen abgedeckten Bereich fällt. Hierdurch entsteht die Eigenschaft der Onlinenutzung, welche auch im Namen wiederzufinden ist.

Leider ist die automatische Auswertung für Cluster nur begrenzt möglich, da keine vordefinierten Ziele oder Merkmale, Ziel des Projektes waren. Vielmehr ging es darum eventuell bestehende Zusammenhänge aufzufinden und sinnvoll in eine Webapplikation einzubinden.

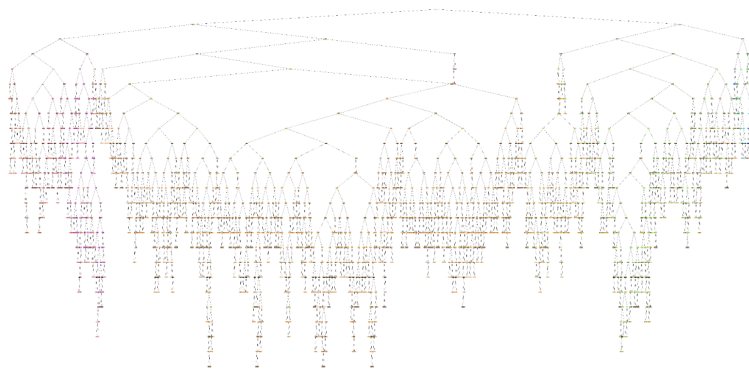
Der ROLF-Cluster Algorithmus fand in folgender Konfiguration:

```
rolf = ROLF(p=2,lr_center = 0.01, lr_sigma = 0.01,initSigma = 20, strategy = 'max')
```

29 Cluster. Da der 11-dimensionale Hyperraum nicht darstellbar ist, wurde mittels einer Principal Component Analysis bzw. PCA die Anzahl der Feature auf zwei reduziert und das Ergebnis in Form eines Scatterplots dargestellt. Neuronen des gleichen Typs sind einheitlich farbig markiert.

**Abbildung 3-9** ROLF Neuronencluster

Besonders auffällig sind die fast horizontalen Streifen entlang der X-Achse. Einige Punktwolken entstehen um die Koordinaten $[-7500, 75]$, $[5000, -50]$. Zur genaueren Untersuchung der Cluster wurde das unsupervised Problem in ein Supervised Problem umgeformt, ähnlich wie beim Neuronalen Netz. Durch den Algorithmus eines Decision Trees, welcher besonders gut visualisier- und interpretierbar ist, plotten wir die Entscheidungsgrenzen zum Klassifizieren der Daten. Daraus entsteht das folgende Baumdiagramm in Abbildung 3-10.

**Abbildung 3-10** Baumdiagramm zur Klassifizierung

Der Decision-Tree ist ein Algorithmus der Schwellwerte für einen durchgegebenen Datensatz berechnet und die einzelnen Datenpunkte in Klassen einordnet. Wie in der oberen Abbildung 3-10 dargestellt, ist das ein großer und komplexer Klassifizierungsprozess bei unserer gefundenen Menge von 29 Klassen.

Der Vorteil des Entscheidungsbaudiagramms ist, komplexe mehrdimensionale Datenzusammenhänge interpretierbar und mit geringerer Komplexität darzustellen. Wir haben uns für die Darstellung der Daten in den jeweiligen Klassen für ein Histogramm entschieden, da der Entscheidungsbaum für unseren Datensatz dennoch schwer zu interpretieren ist. Dieses Histogramm zeigt die Häufigkeitsmenge der hochgeladenen Daten in den vom Cluster-Algorithmus berechneten 29 Klassen an. Das Histogramm lässt sich, wie auch die Korrelationsmatrix, in dem PDF-Report unserer Website generieren.

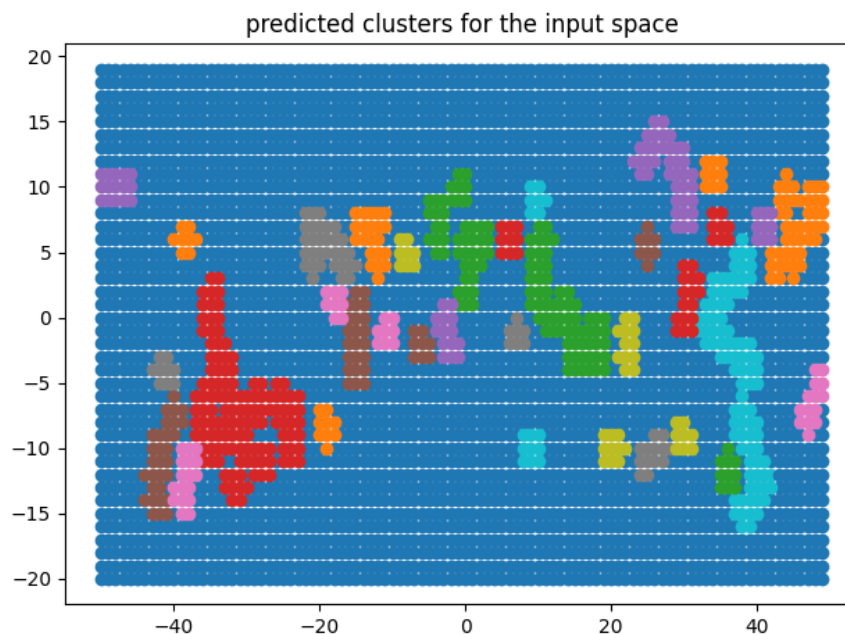


Abbildung 3-11 ROLF Clusteranordnung

3.6.2 Multilayer Perceptron bzw. MLP und Neuronales Netz

Das Multilayer Perceptron, kurz MLP genannt, ist die Weiterentwicklung des Singlelayer Perceptrons und baut auf diesem auf. Erstmals erfunden von Frank Rosenblatt im Jahr 1958, arbeitet das Perceptron als linearer Klassifizierer und unterteilt den Entscheidungsraum in zwei Sektionen. Der folgenden Abbildung 3-12 ist der Aufbau eines Perceptrons zu entnehmen.

Perceptron - Visualized

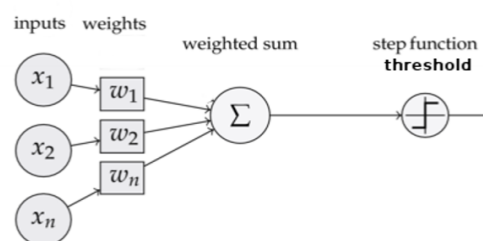


Abbildung 3-12 Visualisierung eines Perceptron

Quelle: <https://www.commonlounge.com/discussion/560f976082d14089ab42b81cd24012c3>

Erweitert man dieses Konzept auf viele parallel und aneinandergereihte Schichten, entsteht ein Multilayerperceptron.

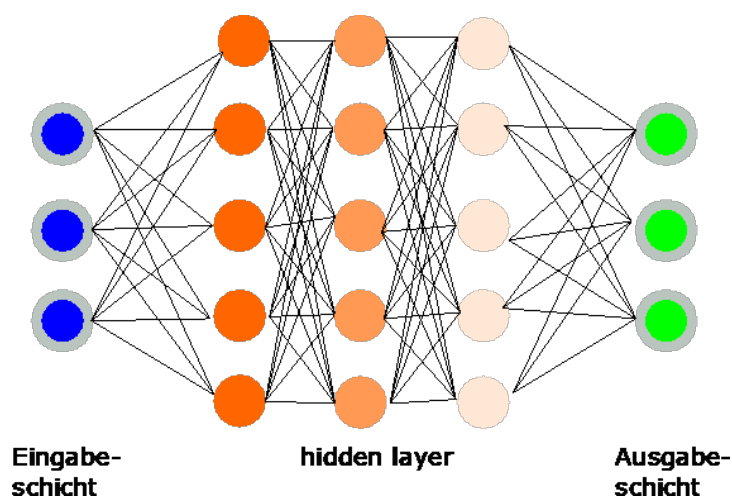


Abbildung 3-13 Multilayerperceptron

Quelle: <https://novustat.com/wp-content/uploads/2020/08/hidden-layer-kuenstliche-neuronale-netze.png>

Der entscheidende Vorteil der MLPs ist die Separierfähigkeit in viele verschiedene Räume.

In dem Projekt wird ein MLP, auch künstliches Neuronales Netz (kNN) genannt, zur Klassifikation der Daten genutzt. Die ermittelten Klassen erhalten wir aus der vorangestellten Clusteranalyse. Neuronale Netze variieren stark in ihrem Aufbau, allgemein gilt, dass ein NN aus einem Input-, Hidden- und Output Layer besteht (Zu sehen in Abbildung 3-13). Die Eingabeschicht wird bestimmt durch die Größe des Eingabevektors und der Dimension des Featureraums, in diesem Fall mit der Form: (11,). In der Ausgabeschicht spiegelt jedes Neuron eine Klasse wider, weshalb 137 Ausgabeneuronen enthalten sind. Die Zusammensetzung der Hidden Layer wurde über eine Trial-and-Error Verfahren ausgewählt, hierbei war das entscheidende Maß die Genauigkeit des Netzes. (Vgl. Backpropagation).

Die entsprechende Architektur ist der folgenden Abbildung 3-14 zu entnehmen, welche über den keras-Befehl `model.summary()` generiert wurde.

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 11)	132
dense_1 (Dense)	(None, 110)	1320
dense_2 (Dense)	(None, 11)	1221
dense_3 (Dense)	(None, 29)	348
Total params: 3,021		
Trainable params: 3,021		
Non-trainable params: 0		

Abbildung 3-14 Modell Zusammenfassung

Für den Lernprozess der Algorithmen wurden die vorhandenen Datensätze in Test und Trainingsdaten unterteilt. Eine Aufteilung in 90% Training und 10% Testdaten hat sich aufgrund der Größe der Daten für sinnvoll erwiesen.

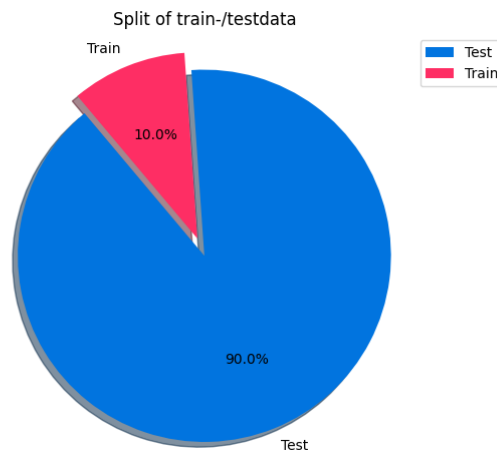


Abbildung 3-15 Verhältnis Training und Testdaten

Bewusst wurde sich gegen das, als Usual Practice angesehene, Skalieren der Werte entschieden, da diese bereits größtenteils in einem Bereich zwischen $[-1,1]$ liegen und die Rechenzeit im Backend unnötig verlängern würde. Dabei geht es hauptsächlich um Werte der Wellen.

Weitere Bestandteile des Netzes sind die loss-Funktion, der Optimizer und die Metrik, nach welcher ausgewertet werden soll. Die Entwurfsentscheidungen sind in Tabelle 3-11 nachzulesen.

Tabelle 3-11 Entwurfsentscheidungen

Bestandteil	Wahl	Erklärung
loss	categoricalCrossentropy	muss Minimiert werden, Ungenauigkeit des NN
Optimizer	ADAM	Schema zur Anpassung der Lernrate
Metrik	Akkuratheit	Wert welcher mini/maximiert werden soll
activation function	ReLu	$f(x) = \max(0, x)$

Die folgenden Abbildungen zeigen den Lernprozess des Neuronalen Netzes und spiegeln den Verlauf von Richtigkeit und Verlust des Modells aufgetragen über die Zeitachse dar. Die erreichte Genauigkeit von 87.84% und der Verlust von <0.5 stellt ein zufriedenstellendes Ergebnis dar und lässt auf kein Overfitting schließen. Das Overfitting

ist eines der größten Probleme eines Neuronalen Netzes. Die Problematik ist das Verinnerlichen der Trainingsdaten und die geringe Abstraktionsfähigkeit für fremde Daten. Präventive Möglichkeiten zur Vorbeugung von Overfitting sind eine geringe Epochenanzahl, Einfügen von sogenannten Dropout-Schichten, bei denen während des Lernens Neuronen abgekapselt werden und umliegende Neuronen dann die Gewichtung abfedern müssen.

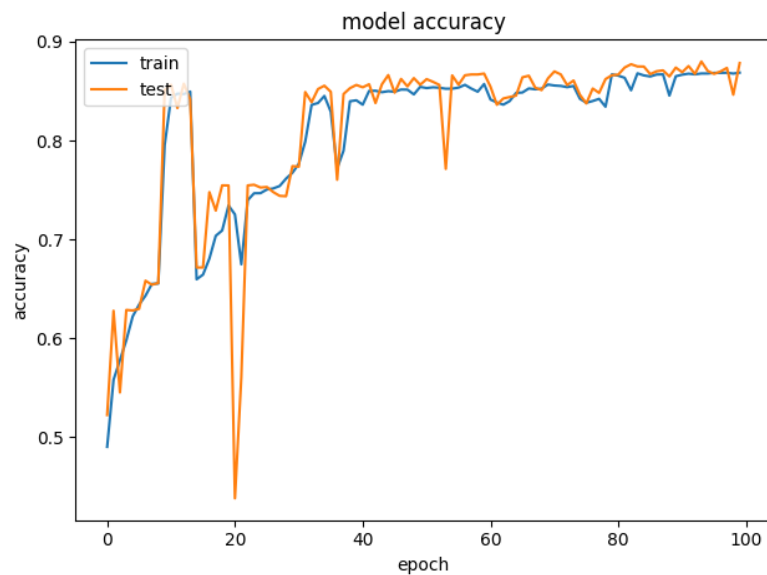


Abbildung 3-16 Modell Akkuratheit

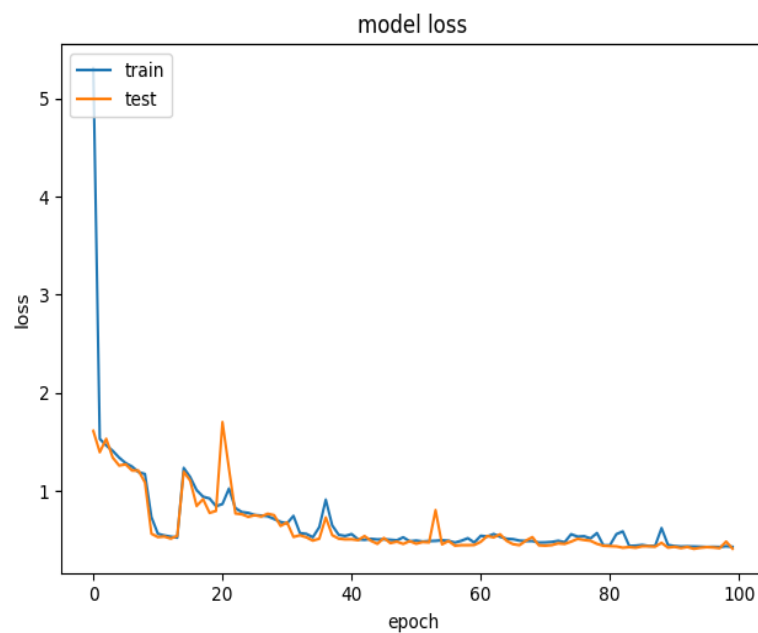


Abbildung 3-17 Modell Verlust

4 Fazit

Die Realisierung, der im Rahmen dieser Projektarbeit geforderten Funktionalitäten (der WEB-Applikation), ist weitestgehend gut gelungen.

Nun ist es Anwendern möglich Datensätze mit einem normierten Header auf der Webseite hochzuladen, sich diese zu plotten und mithilfe eines PDF-Reports zu analysieren.

Besonders gut ist es uns gelungen die Homepage auf einem Server zu hinterlegen bzw. sie somit öffentlich erreichbar zu machen und durch Integrationsmanagement diverse Arbeitsschritte zu automatisieren (was neben der Zeitersparnis einen qualitätssichernden Effekt hat).

Neben dieser Möglichkeit die Homepage im Internet aufzurufen, kann man auch die Webseite lokal auf seinem individuellen Endgerät ausführen. Das hat den Vorteil, dass man dadurch größere Datenmengen analysieren kann (durch die erhöhte Rechenkapazität). Zur lokalen Ausführung der Homepage sind einige Installationsschritte notwendig, die unter folgendem Link dokumentiert sind:

<https://github.com/Web-App-zur-Einzelblattmontagen/offshoreSingleBladeInstallation/wiki/Setup-for-Local-Development>

Da unsere Gruppe insgesamt wenig Vorwissen im Bereich des maschinellen Lernens bzw. der Datenanalyse hatte und trotz der, während des kompletten Projektverlaufs, geltenden Covid19 Kontakteinschränkungen, konnten wir uns gemeinsam als Gruppe in diese thematische Herausforderung einarbeiten.

Ein enger teaminterner Austausch, ein strukturiertes Zeitmanagement, eine klare Rolleneinteilung und Agilität waren dabei ausschlaggebend für den erfolgreichen Projektverlauf.

5 Literaturverzeichnis

[Ahu05] Ahuja, S. 2005, Comparison of web services technologies from a developer's perspective. Proceedings of the International Conference on Information Technologie: Coding and Computing. (Seite 791-792)

[Aug18] Augsten, S. 2018, Was ist Versionskontrolle, dev-insider.de/was-ist-versionskontrolle-a-701050/, aufgerufen am 23.07.2021

[Das05] Daschiel, S. 2005, Evaluierung von Web-frameworks, Erstellung eines Prototyps und Konzeption der multimedialen Wissensdokumentation für die Kunstuniversität linz. Master's thesis

[Fow04] Fowler, M. 2004, UML konzentriert, 3. Auflage (Seite 21)

[Kal14] Kalelkar, M. 2014, Implementation of Modal-View-Controller Architecture Pattern for Business Intelligence Architecture (Seite 1)

[Kri05] Kriesel D. 2005, Ein kleiner Überblick über Neuronale Netze (ZETA2-DE), (Seite 184) aufgerufen am 14.08.2021

[Rup12] Rupp, C. 2012, UML2 glasklar, 4. Auflage (Seite 242)

[Sam08] Samkari, K. 2008, Comparison matrix for web hci. Information and Communications Technologies: From Theory to Applications. (Seite 1-5)

[San20] Sander, A. 2020, Projektbeschreibung, <https://elearning.uni-bremen.de/dispatch.php/course/details/?cid=458fb55f3c2537964c5344a533b125c9>, aufgerufen am 27.08.2021

[San21] Sander, A. 2021, Kinematics of tower and blades during installation – Analysis of in–situ measurement data

[Sch09] Schmidt, M. 2009, Software-Metriken als Mittel zur Technologieauswahl am Beispiel zweier Implementierungen einer Web-Applikation in Rails und JEE. (Seite 53)