Stromerzeugung bei unterschiedlichen Wetterlagen – Datenaufbereitung und Analyse zur Korrelation Mithilfe von Regressionsalgorithmen in Weka

Korbinian Eller, Kay Gietenbruch

Abstract

HALLO WELT

In dieser Ausarbeitung wird das Vorgehen beim Sammeln von Daten, deren Aufbereitung und die Analyse mithilfe der Regressionsalgorithmen in Weka erläutert und die Ergebnisse dokumentiert.

Ziel ist es, durch Wetterdaten des Deutschen Wetterdienstes (DWD) auf Stromerzeugungszahlen der erneuerbaren Energien Solar und Wind (größte Beeinflussung durch Wetter) zu schließen.

Idee

Zur gegebenen Themenstellung ”Einen Themenbereich der KI vertiefen” war eine erste Idee die Klassifikation von Datensätzen. Dies war sehr ähnlich einer der letzten Aufgabenstellungen der Übungsstunden des Fachs. Es war vor allem aufgrund der Implementierung in der Programmiersprache C und damit dem greifbar machen des Algorithmus interessant.

Der Gedanke der Klassifikation war schnell gefestigt, nun fehlten noch die Daten, mit denen gearbeitet werden soll. Über Daten wie Covid-19 Erkrankungszahlen, Bußgeldbescheide, Denkmalstandorte oder Amazon Personendaten ist eine Idee herausgestochen.

”Es wäre doch interessant, Wetterdaten und Stromdaten in Korrelation miteinander zu bringen und so die Stromerzeugung anhand des vorherrschenden Wetters deuten zu können.” Und das war dann das Noema, mit dem fortgefahren werden sollte. Das Ziel ist, eine repräsentative Menge der Daten zu sammeln, um eine Klassifikation sinnvoll ausführen zu können. Trotz all dem war die Beschränkung der Daten auch ausschlaggebend. Festgelegt wurde sich dann auf den Zeitraum eines Jahres und wegen der Zeit, in der die Daten gesammelt wurden (Ende 2023) war das Jahr 2022 passend für die Aufgabe.

Das Projekt konnte indessen in mehrere Schritte unterteilt werden: die Daten sammeln, die Daten aufbereiten, den Klassifikator programmieren, testen und verbessern und die Arbeit dokumentieren.

Nach Rücksprache mit dem Betreuer des Projekts Herrn Prof. Dr. Baumann wurde jedoch klar, dass eine Klassifizierung der Daten nicht das geeignetste Modell für die Analyse der Korrelation ist. So wurde der Plan neu geschrieben und eine Analyse mithilfe von Regressionsalgorithmen in dem Machine Learning Programm Weka stand ab dem Zeitpunkt im Vordergrund. Dafür sollten die Daten noch angepasst und dann mit Weka und den Regressionsalgorithmen experimentiert werden. In dem Sinne, dass das am bestgeeignete Modell zur Regression gefunden wird!

Datenrecherche

Die Daten, mit denen die Regression in Weka betrieben werden soll, müssen zuerst zusammengetragen werden. Sowohl die Daten des DWDs, respektive die Wetterdaten, als auch die der Bundesnetzagentur(BNetzA), welche die Daten der Stromerzeugung bereitstellt, stehen leider nicht auf der jeweiligen Websites als Download in Form von z.B. ”.csv” oder ”.xls” Dateiformaten zur Verfügung. Hier war also eine andere Herangehensweise gefragt. Nun muss zwischen den Daten des DWD und denen der BNetzA unterschieden werden, da diese in unterschiedlicher Form vorliegen und somit auch einem unterschiedlichen Sammel-Prozess unterlaufen sind.

Es ist anzumerken, dass dieser Arbeit einige Dateien beigelegt sind, unter denen sich auch ”.md” Dateien befinden, in welchen sehr speziell auf die einzelnen Schritte und die Struktur eingegangen wurde.

BNetzA Daten

Die Bundesnetzagentur stellt ihre recht simplen Daten (Im Vergleich zum DWD) auf der Website ”Strommarktdaten” sehr anschaulich in einem Flächendiagramm dar. Aus solch einem Diagramm Werte in eine Datenbank zu übertragen wäre aber bei den stündlichen Werten von dem ganzen Jahr 2022, sprich 8760 Zeilen einer CSV Datei und jeweils zwölf Datentypen undenkbar. Glücklicherweise liefert die BNetzA nicht eine Grafik an die Website, sondern die stündlichen Daten in mehreren Dateien vom Typ ”.json” an den Browser, wo dann ein Diagramm erstellt wird.

Das heißt, durch einen einfachen ”cURL” Befehl auf der Kommandozeile, kann eine solche Datei, wenn der vollständige Name bekannt ist, heruntergeladen werden. Die Namen bestehen aus einer Zahlenkodierung der unterschiedlichen Daten-Arten (z.B. Kernenergie - 1224, Wind Onshore - 4067), dem Kürzel DE für Deutschland, der

”Auflösung” der Daten (z.B. hour, day) und dem Epoch Zeitstempel für den frühesten aufgezeichneten Wert in dieser Datei. Diese Dateien erfassen immer eine ganze Woche, das heißt die Epoch Zeit Codes unterscheiden sich immer um genau 604.800.000 Millisekunden, da dies genau 7 Tage abbildet. So konnte das Jahr 2022 schnell in Epoch Codes abgebildet werden und es stand fest welche Dateien benötigt werden, da alle Type erzeugten Stroms geladen wurden, weil es relativ problemlos machbar war.

Der Aufbau der angesprochenen ”.json” Dateien ist ein Array mit 180 Datenpunkten, die jeweils zwei Werte haben. Einmal den Epoch Zeit Code und die Höhe des gemessenen Stroms des jeweiligen Typs in MWh. Am Anfang der Datei beschreibt eine Versionsnummer und ein Zeitpunkt (ebenfalls Epoch) die Erstellung der Datei. Der nächste Schritt war nun, die Dateien in einem iterierendem Python Programm auf der eigenen Festplatte zu speichern. Einziges Problem hierbei war nur, dass die Dateien immer montags um 0 Uhr beginnen, aber das Jahr 2022 nicht auf einen Montag begonnen bzw. auf einen Sonntag geendet ist. Folglich sind mehr Dateien, als es eigentlich braucht, um ein Jahr abzubilden, geladen worden.

DWD - Daten

Der Deutsche Wetterdienst stellt seine Daten auf einem opendata Server zu Verfügung. Nachdem sich in der Dateistruktur zurechtgefunden wurde, sind auch die ersten Probleme aufgekommen:

- Welche Wetterdaten sollen benutzt werden?
- Wie werden die gewünschten Zeiträume gefunden?
- Welche Dateien enthalten welche Daten?

Einiges, das im ersten Moment sehr unklar erschien! Die Lösungen waren dann wie folgt:

Datenart Es wurde sich auf Eigenschaften beschränkt, die aus subjektiver und ungeschulter Sicht stark in die Stromproduktion als Faktoren miteinfließen. In erster Linie handelt es sich um erneuerbare Energien, also galt es bestimmte Naturphänomene, die Photovoltaik-Anlagen oder Windkraftanlagen beeinflussen, auszumachen. Dies sind:

- Lufttemperatur, Feuchtigkeit
- Bedeckungsgrad des Himmels
- Niederschlag
- Sonnenscheindauer
- Sichtweite
- Wind

Das waren die Unterteilungen der Messwerte auf Seite des DWD

Zeiträume Da unter jeder der eben aufgezählten Sektionen in dem Dateisystem des Servers eine Unterteilung in ”recent” und ”historical” stattfand und der gewünschte Zeitraum das Jahr 2022 war, galt es herauszufinden, wie die Aufteilung zustande kam. Leider gibt es keine klare Unterteilung und somit blieb nichts anders übrig, als in beiden Ordnern nachzuschauen.

Dateien In den zeitunterteilten Ordnern liegen genau eine Datei, die die Liste der Stationen mit ihren Eigenschaften aufzählt, welche den betrachteten Wert aufzeichnen sollen und hunderte komprimierte Ordner, benannt nach Stationscode, Daten der Erfassung (nur in ”historical”, nicht in ”recent”) und erfasste Eigenschaft. Teilweise reichten die erfassten Daten von 1970 bis 2001 oder von Mitte 2022 bis Anfang 2023, es war also kein System hinter den Aufzeichnungen zu Erkennen. Die ”.zip” Ordner enthalten mehrere Dateien, teilweise ”.html” und teilweise ”.txt” Dateien und das sogar manchmal in doppelter Ausführung. Trotz alledem gibt es in jedem Ordner eine ausschlaggebende ”.txt” Datei, die die vom Ordnernamen versprochenen Daten beinhalten.

In den angesprochenen Dateien ist allerdings eine gewohnte Struktur wiederzufinden. Aufgebaut wie eine ”csv” Datei nur mit Semikola getrennte Spalten. Die erste Spalte besteht aus dem Datum gepaart mit der Stundenzahl des Tages im 24h-Format, zu welchem Zeitpunkt die Daten der Reihe von den Messinstrumenten ausgelesen wurden. Die nächsten Spalten (1-3, je nach Datei) bestehen aus den gewünschten Attributen wie Sonnenscheindauer. Das Ende der Zeile macht immer ein ”eor” als Zeilenende-Indikator. Manche Werte, unterschiedlich von Station zu Station und je nach Tageszeit sind gar nicht aufgelistet bzw. als fehlend (-999 in der Datei) gekennzeichnet. Für einige Stationen, wie zuvor schon thematisiert, stehen überhaupt keine Daten für bestimmte Messattribute in Form von Dateien in dem gewünschten Zeitraum zur Verfügung. Oft wurden auch nur in einem Teil des Jahres 2022 Daten erhoben, bzw. zur Veröffentlichung auf dem Server freigegeben, was zu ”halben” Dateien führt.

Trotz all diesen Hürden war es klar, dass das Sammeln dieser Daten nicht ”per Hand” realisierbar ist, sondern durch Automatisierung geschehen soll. Ohne Vorkenntnisse in Python war das Vorhaben in der Tat anspruchsvoll, aber die überaus breit gefächerte Dokumentation der Sprache und den vielen Bibliotheken erleichterten das Programmieren sehr. Zu den Vorlesungen zu erscheinen, schien sich in diesem Zuge zu lohnen, als die Benutzung von RegEx (und somit den Grundlagen der Informatik) das Filtern der Dateien um einiges erleichterte. Einmal geschrieben, iterierte das Skript durch alle oben genannten Datenarten und sammelte und entpackte die Dateien auf der eigenen Festplatte. Die Wetterdaten vollständig heruntergeladen blieb noch die Daten der Bundesnetzagentur.

Datenaufbereitung

Der nächste Schritt ist die Datenaufbereitung, bei der die eben gesammelten Daten in vorzugsweise ”.csv” Dateien geschrieben/konvertiert werden.

Zuerst wurden zwei eigene Dateien erstellt, die dann später noch raffiniert und letztendlich sogar partiell vereinigt wurden.

Stromdaten

Die Daten der Bundesnetzagentur tabellarisch zu sortieren war keine Kunst, da es glücklicherweise passende Python

Bibliotheken gibt, die ".json" Dateien in Listen einlesen können und erleichtern somit die Erstellung einer Datei im ".csv" Format.

Ein Beschneiden der Daten war nun noch zu erledigen, da wie oben genannt die Epoch Codes nicht genau passten. Also wurden die überflüssigen Codes (die vor dem 01.01.2022 und die nach dem 12.31.2022) abgeschnitten. Dann wurden die Epoch Codes mit passender Bibliotheken in Daten übersetzt und eine "header"-Zeile wurde der Datei vorangesetzt. Durch die Epoch Codes konnte das Zeitumstellungssystem in Deutschland ignoriert werden, da die Unix-Zeit das nicht beachtet.

Wetterdaten

Eine andere Handhabung gebührt hier den Wetterdaten. Das Komplexe an diesen Daten ist die Aufteilung. Gemeint damit, ist die hohe Anzahl an Stationen, die nicht immer die Gleichen/gleich viele Messwerte besitzen. Wie soll also eine Datenbank daraus kreiert werden. Das Vorgehen war hier essentiell, denn die gleichen Messwerte unterschiedlicher Stationen sollten nebeneinander liegen, aber es sind wie eben genannt nicht immer gleich viele für die einzelnen Werte. Folglich ist entschieden worden, dass alle Stationen in eine ".csv" Datei geschrieben werden und wenn es für eine Station gar keine Werte gibt, wird dies mit dem Wert -777 an der respektiven Stelle in der Datei vermerkt.

Bevor das aber geschehen kann, wurden die ".txt" Dateien die für jede Station vorliegen zuerst in ".csv" Dateien geschrieben, um später eine einheitliche Verschmelzung garantieren zu können. So wurde für jede Station ein Ordner angelegt, in dem im besten Falle 6 Dateien gespeichert wurden. Je nach Attribut wurden aus den Textdateien unterschiedliche Spalten ausgelesen. Um den Zeitfaktor in Takt zu halten, bildete die erste Zeile jeder ".csv" immer Datum und Stundenwert ab.

Um dann die große allumfassende tabellarisch strukturierte Datei aufzubauen wurde mit RegExs die Dateistruktur durchsucht und die Namen der Stationen wurden somit für die "header"-Zeile und für die Anzahl der Stationen erhalten. So konnten dann die Werte der kleinen spezifischen ".csv" Dateien in die große übernommen werden. All diese Operationen wurden natürlich auch per Python Skript verübt.

Qualität Nachdem das Übernehmen der Werte funktioniert hatte, reichte ein Blick, um festzustellen, dass die Datenlage nicht besonders qualitativ war.

In, dem Gefühl nach, der Hälfte der Einträge war entweder eine -999 oder eine -777. Tatsächlich auch nicht zu verwunderlich, da von den über eintausend Stationen, mehr als 80% nicht alle 6 Datentypen aufwiesen.

Mit diesen Daten war nicht zu arbeiten, das stand fest. Die Frage war nur: "Wie kann die Datenlage verbessert werden?". Und genau dieses Problem wurde durch ein erweitertes Kultivieren der Daten adressiert.

Verfeinerung

Speziell die Wetterdaten waren so nicht zu gebrauchen. Aber wie sollen die Daten beschränkt/verbessert werden, um eine repräsentative Menge zu erreichen, die auch noch das ganze Jahr 2022 abdeckt? Diese Frage war die zentrale Problemstellung der Datenaufbereitung für dieses Projekt. Und die Antwort darauf war nicht einfach zu finden. Erst einmal war es nicht möglich, die Datenlage ohne Begrenzung der Daten zu verbessern, da bereits alle vom DWD bereitgestellten und gewünschten Daten für diesen Zeitraum berücksichtigt wurden. Also war die andere und wohl einzig verbleibende Option die Begrenzung auf eine bestimmte Menge der Daten. Bei dieser Aufgabenstellung war der erste Gedanke, die Stationen für ein großes Dokument zu nutzen, die mindestens 5 von den 6 Wetterdatentypen aufweisen. Einfacher gesagt als getan, da durch einen Fehler in dem Python Programm, das die Textdateien in ".csv" Dateien übersetzt hat, auch leere Dateien in den Stationsordnern umherirrten. Auch bei erneutem Überarbeiten des Quellcodes war nicht klar, warum das geschieht. Also traten Dateien nur mit "header"-Zeile und ohne Daten auf, aber das ganz unregelmäßig und ohne zu erkennendes Muster. Trotzdem noch möglich, aber komplizierter als gedacht kam die Idee auf, einfach nur die Stationen zu benutzen, die alle Datentypen aufwiesen. Das Problem bestand dann immer noch, keine Frage, aber der Anteil der Fehlwerte sollte geringer sein. Fraglich dabei ist nur, ob die dann übergebliebene Anzahl an Stationen noch repräsentativ für das Wetter ist.

Das konnte erst festgestellt werden, als ein, im Gegensatz zum Vorherigen, kleineres ".csv" Dokument erstellt wurde. Der Anblick war um einiges vielversprechender! 157 Stationen blieben nun noch übrig, die genau wie in der großen Datei geordnet, also den Wetterdatentypen nach und begonnen mit dem Datum und der Aufzeichnungsstunde in die Datei geschrieben wurden. Weitaus weniger, sowohl anteilsmäßig als natürlich auch absolut, Fehlwerte konnten mit bloßem Auge festgestellt werden. Die Menge an Daten war trotzdem noch sehr repräsentativ, da 157 Stationen á 10 Wetterattribute eine beachtliche Menge an Spalten ist.

Trotzdem wurde festgestellt, dass es einen Bereich in dem Dokument gibt, der nur aus Fehlwerten besteht (-999). Das waren die Werte der Niederschlagsform. Auch in der Datenrecherche wurde bereits festgestellt, dass diese Spalte meist einfach nur Fehlwerte beinhaltet. Kurzerhand wurde entscheiden, dass dieser Indikator/Zahlencode keine besonders ausschlaggebende Bedeutung besitzt und er wurde in der nächste Verfeinerung der Dateien aus allen Stationen weggelassen.

Strom als Additum für Weka Anders als für den selbstgebauten Klassifikator, müssen für die Regression in Weka die Wetterdaten und dazugehörigen Stromdaten innerhalb einer Datei vorliegen, um sie sinnvoll zu verwerten. Da die erneuerbaren Energien Sonnenstrom und Windstrom essentiell waren, wurden 3 unterschiedliche Dateien kreiert. Allen diente die eben beschriebene Datei als Basis und als letzte Spalte wurde die Stromerzeugung in MWh angehängt. Einmal Solarstrom, einmal offshore Windkraft und einmal onshore Windkraft.

Um das Testen unterschiedlicher Algorithmen in Weka schneller zu machen, wurden noch kompaktere Dateien erstellt, bei denen der Durchschnitt aller Wetterdaten als

Wert für diesen Wetterdatentyp dient. Fehlwerte wurden nicht in den Durchschnitt mit einberechnet. Bei der Sonnenscheindauer waren oft zwischen 22Uhr und 3Uhr morgens, keine Aufzeichnungen bei allen 157 Stationen und dieser Wert wurde dann in der kompakteren Datei als realistische Abbildung auf 0 gesetzt. Auch an diese Dateien wurden dann analog zum vorherigen Prozess die Stromdaten als letzte Spalte angehängt.

Um die Dateien dann noch schneller in Weka zu laden wurden alle ".csv" Dateien in ein ".arff" Format übersetzt, mit welchem sich Weka besser zurechtfindet.

Auswahl geeigneter Modelle

Bei der Wahl eines KI Algorithmus muss man die grundlegende Unterteilung in Klassifikationsalgorithmen und Regressionsalgorithmen beachten. Bei der Klassifikation werden die Daten in Kategorien eingeteilt. Bei der Regression ist der Output numerisch. Da es das Ziel war numerische Stromdaten vorherzusagen, eignen sich Regressionsalgorithmen für diese Aufgabe.

Mögliche Algorithmen wären beispielsweise die Polynomregression, ein Modell, mit dem es möglich ist, nicht lineare Beziehungen zu modellieren. Die Lasso-Regression ist eine Regressionsmethode, die durch die Einführung eines Strafterms in der Kostenfunktion der linearen Regression die Vorhersagegenauigkeit verbessert. Ein weiteres lineares Modell ist die Ridge-Regression, bei der der Kostenfunktion ein Strafterm hinzugefügt wird. Die Bayessische lineare Regression ist ein probabilistischer Ansatz zur linearen Regression auf der Grundlage des Bayes-Theorems.

Für die Tests wurde die Weka Version 3.8.6 verwendet. Weka bietet eine Vielzahl von unterschiedlichen Regressionsalgorithmen an. Es erschien sinnvoll, die Tests auf die folgenden 5 Algorithmen zu konzentrieren:

Lineare Regression Bei der Linearen Regression werden alle Input Variablen gewichtet und zu einer Linearkombination zusammengefügt. Mit dieser Gleichung ist es möglich, einen numerischen Wert vorherzusagen. Die lineare Regression ist jedoch nur sinnvoll, wenn eine lineare Abhängigkeit zwischen den Daten besteht.

Multilayer Perceptron Das MLP ist ein künstliches neuronales Netzwerk. Es besteht aus mindestens 3 Schichten. Dem Input Layer, mindestens einem Hidden Layer und dem Output Layer. Die Schichten bestehen aus miteinander verknüpften und gewichteten Neuronen.

KNN (lazy IBK) K Nearest Neighbours kann auch für Regression verwendet werden.Beim KNN wird der Abstand zu allen anderen Punkten berechnet, um den am nächsten liegenden Punkt zu finden. Bei der KNN-Regression wird dem Datenpunkt der Durchschnitt der k nächsten Nachbarn zugeordnet.

Support Vector Regression (SMOreg) Bei der Support Vector Regression wird eine Hyperebene erstellt, mit deren Hilfe der Abstand zwischen den vorhergesagten Ausgaben und den tatsächlichen Ausgaben minimiert wird. Dazu kommen Kernel Funktionen zum Einsatz.

Decision Tree (REPtree) Entscheidungsbäume bilden Daten als Entscheidungsregeln ab. Diese Entscheidungsregeln können graphisch dargestellt werden. Der von Weka implementierte REPTree nutzt "reduced error pruning". Dabei wird ein Bottom-Up Algorithmus verwendet, um den Entscheidungsbaum zu kürzen.

Analyse mit Weka

Ein Hindernis für die tatsächliche Analyse der Daten ist die Größe der Dateien und die damit verbundene Laufzeit. Von den ausgewählten Algorithmen waren nur KNN und Reptree dazu in der Lage, die gesamten Dateien in annehmbarer Zeit zu bearbeiten. Beim Vergleich der Daten wird der Relative Absolute Error als Hauptkriterium verwendet. Dieser unterscheidet sich kaum vom Root Relative Squared Error. Der Root Relative Squared Error ist meist konstant größer als der Relative Absolute Error. Deswegen hätte es keinen Unterschied beim Vergleichen der Algorithmen gegeben.

RAE in %	solar	wind onshore	wind offshore
knn k=1	23.0258	10.1998	17.7827
REPtree	26.2540	31.5993	47.0417

Interessanterweise liefert KNN bei den Wind Dateien ein besseres Ergebnis. REPtree verhält sich entgegengesetzt und liefert für Solar ein besseres Ergebnis. Die Ergebnisse für die Wind Dateien verschlechterten sich bei REPtree deutlich.

Da beispielsweise das Multilayer Perceptron nach über 10 Stunden nicht fertig wurde, mussten die Dateien gekürzt werden, um mit den restlichen Algorithmen arbeiten zu können. Eine Kürzung zu 10% der ursprünglichen Menge führte zu einem Crash,weswegen die Algorithmen mit 150 Instanzen getestet wurden.

RAE in %	solar
knn k=1	28.3896
REPtree	35.0764
Linear Regression	31.2483
Multilayer Perceptron	59.8705
SMOreg	35.1404

Die lineare Regression dauert auch bei den gekürzten Dateien ca. 45 Minuten. Das Multilayer Perceptron braucht mehrere Stunden.

Jedoch liefern die gekürzten Dateien schlechtere Werte als die Gesamtdateien. Für KNN verschlechterte sich von einem absolute error 23,0% auf 28,3%. REPtree verschlechterte sich von 16,0% auf 35,1%. Vier der Algorithmen landen bei ca. 30%. Deutlich schlechter ist nur das Multilayer Perceptron mit 59,9%.

Nur bei den gekürzten Datein gibt es bei zwei Algorithmen einen merklichen Unterschied zwischen dem Relative Absolute Squared Error und dem Root Mean Squared Error. Beim KNN, Multilayer Perceptron und Reptree liegen die Root Mean Squared Errors deutlich über den Absolute Squared Errors. Bei der Linearen Regression und dem SMOreg sind beide Werte fast gleich.

Da die Daten eine große Menge an Fehlwerten enthalten, stellte sich die Frage, ob die Ergebnisse durch verschiedene Filtereinstellungen verbessert werden können. Da die Fehlwerte in den Daten auf entweder -777 oder -999 festgelegt sind, ist es möglich, mit dem NumericCleaner Filter alle Fehlwerte zu markieren. Da die Daten keine hohen Negative Werte enthalten, ist es möglich, nach allen hohen negativen Werte zu filtern. Danach können alle als fehlend markierten Werte durch den Replace Missing Values Filter durch den Durchschnitt dieser Spalte ersetzt werden.

RAE in %	solar	wind onshore	wind offshore
knn k=1	21.9235	10.0296	13.8952
REPtree	16.8667	27.7824	38.2625

In den ganzen Dateien haben sich die Werte kaum verändert. Die Fehlwerte hatten keinen merkliche Einwirkung auf diese Ergebnisse.

RAE in %	solar
knn k=1	28.1755
REPtree	26.5653
Linear Regression	26.8595
Multilayer Perceptron	99.3440
SMOreg	29.7308

Jedoch in den gekürzten Dateien verbesserte sich der Absolute Error bei REPtree (35,0 % zu 26,6 %), linear Regression (31,2 % zu 26,9 %) und SMOreg (35,1% zu 29,7%). KNN blieb gleich (28,4 % zu 28,2 %). Multilayer Perceptron jedoch verschlechterte sich katastrophal (59,9 zu 99,3 %)

Eine weitere Option ist der Normalize Filter. Durch ihn werden alle Werte in den Bereich -1 und 1 übersetzt (scale: 2 translation: -1). Die Daten unterscheiden sich oft drastisch in Bezug auf ihre Größe. Besonders bei Distanz basierten Algorithmen kann dies zu Problemen führen.

RAE in %	solar	wind onshore	wind offshore
knn k=1	21.9235	10.0296	13.8952
REPtree	16.8361	27.8090	38.2492

Jedoch führte diese Einstellung zu keinen merklichen Veränderungen.

Vergleich der Algorithmen

Interessanterweise haben KNN, REPTree, Lineare Regression und SMOreg bei den gekürzten Dateien sehr ähnlich performt. Leider war es nicht sinnvoll, Lineare Regression und SMOreg auf den ganzen Dateien zu testen, da die Laufzeit zu lang ist. Die Ergebnisse dieser beiden Algorithmen wären sicherlich interessant gewesen. Auffallend ist besonders die schlechte Performance des Multilayer Perceptrons. Vielleicht wäre es möglich gewesen, durch die Änderung von Einstellungen dieses Ergebnis zu verbessern. Aufgrund der langen Laufzeit des MLP (3-5 Stunden für die auf 150 Instanzen gekürzten Dateien) ist es jedoch schwer möglich, an den Einstellungen “herumzuspielen”.

Fazit

Mit den geringen Kenntnissen in Python und in der Handhabung mit großen Datenmengen bzw. diese aufzubereiten, war der Prozess des Sammelns und Kultivierens überraschend aufwändig. Die Unverfügbarkeit von bestehenden zusammenhängenden Datensätzen machte deren Benutzung für diese Aufgabenstellung allerdings unmöglich.

Die Arbeit mit Weka war grundsätzlich weniger interessant als das eigenständige Programmieren der KI. Weka ist ein praktisches Tool. Aber für Personen, die an der Funktionsweise von Künstlichen Intelligenzen interessiert sind, ist es relativ unspektakulär, da man durch die Nutzung von Weka nicht lernt, wie die Künstlichen Intelligenzen aufgebaut sind und funktionieren.

Grundsätzlich liefern die Algorithmen für die ganzen Dateien zufriedenstellende Werte. Dass die gekürzten Dateien schlechtere Ergebnisse liefern, war zu erwarten. Durch weniger Trainingsdaten können die Algorithmen weniger gut trainiert werden, wodurch sie schlechter abschneiden. Das liegt auch an der Varianz der Daten, die in den kompakten Dateien verloren geht. Überraschend war der geringe Einfluss den, die Fehlwerte auf das Ergebnis haben. Anscheinend spielen die Fehlwerte auf Grund der großen Datenmenge keine Rolle. Außerdem ist es erstaunlich, dass die Normalisierung der Werte keinen Einfluss auf das Ergebnis hat.

References

Bundesnetzagentur. 2023. BNetzA (01.01.2022-12.31.2022). *https://www.smard.de/home/marktdaten*.

Deutscher-Wetterdienst. 2023. DWD (01.01.2022-12.31.2022). *https://www.smard.de/home/marktdaten*.

Jenifa, A. aufgerufen am. 30.01.2024. *https://geekflare.com/de/regression-vs-classification/*.

Witten, I. H.; Frank, E.; and Hall, M. A., eds. 2011. *Data Mining: Practical Machine Learning Tools and Techniques*. Burlington, MA: Morgan Kaufmann, third edition.