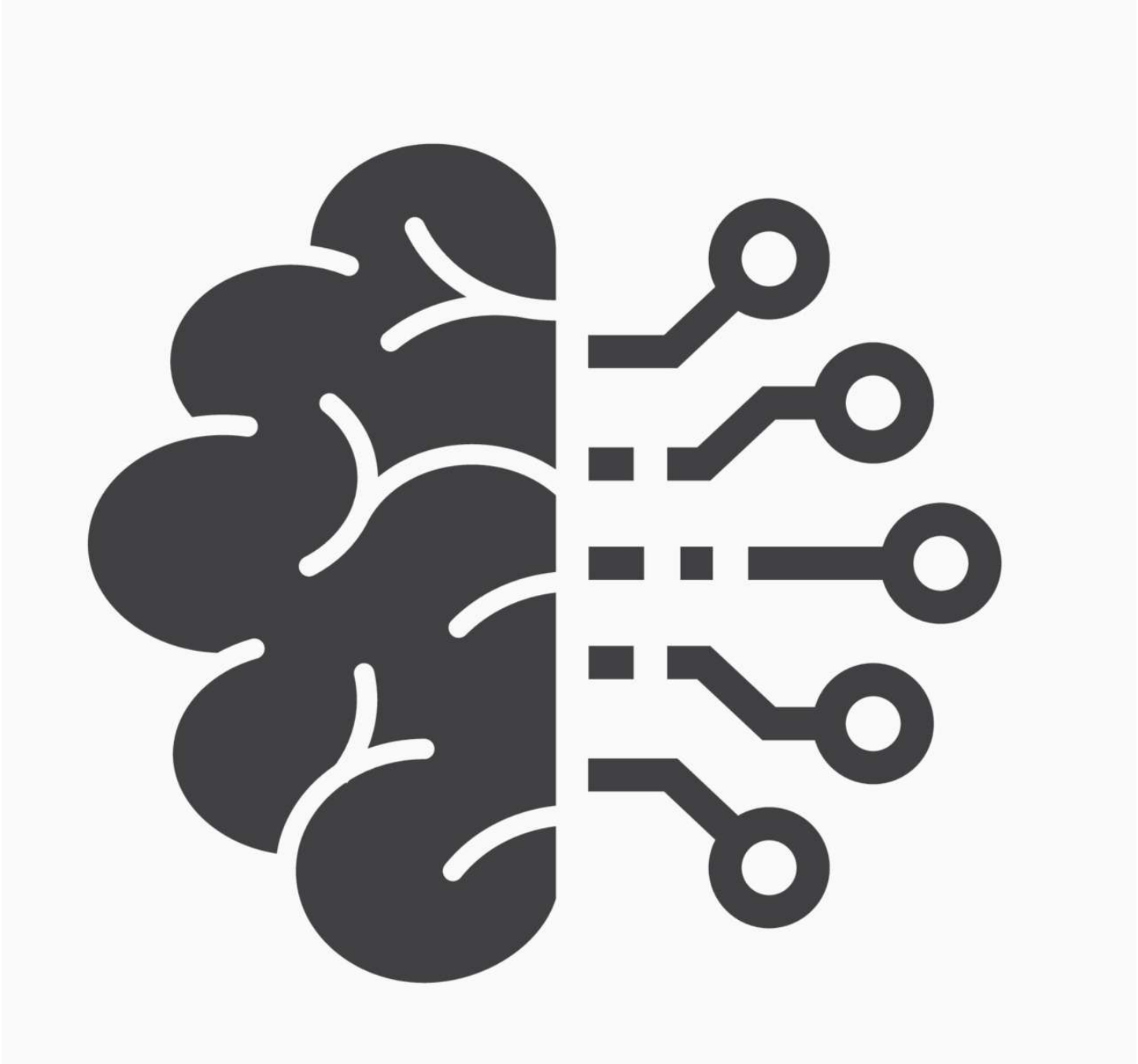


Abbildung 1 Titelbild

Maschinelles Lernen & Wissensbasierte Systeme

Energy Forecasting & Anomalie Detection

Yaren Akinc, Kerem Akkaya und Haris Salii

Summary

Requirements.txt

Inhaltsverzeichnung

1 Einleitung

Dieses Projekt wurde im Rahmen des Bachelor Studiengangs «Business Artificial Intelligence» an der «Fachhochschule Nordwestschweiz» (FHNW) im Modul «Maschinelles Lernen & Wissensbasierte Systeme», durchgeführt.

Ziel des Moduls ist die Entwicklung einer Künstlichen Intelligenz, die eine selbstgewählte Problemstellung löst. Dabei wird der gesamte Entwicklungsprozess von der Problemdefinition über die Modellentwicklung bis hin zur Evaluation und einem möglichen Deployment durchlaufen. Das erforderliche technische und theoretische Know-How wird sich hierbei im Selbststudium, mit Unterstützung des Dozenten, angeeignet.

In diesem Projekt arbeitete das Projektteam gemeinsam an der Entwicklung einer datengetriebenen KI-Lösung, auf Basis eines Allgemein anwendbaren Machine-Learning Frameworks.

1.1 Ziele & Aufgabenstellung

Die Aufgabenstellung wurde von und nach einer eigenständigen Suche nach einem geeigneten Datensatz definiert. Mit dem Datensatz der IWB, welches den Strombernrauch an dessen Kunden beinhaltete, fanden wir einen guten Business Case, an welches wir unser Projekt anwenden konnten.

Konkrete Aufgabenstellung:

- Nachfrageprognosen für Stromverbrauch:
  - Nächste 15 Minuten Verbrauchsintervall
  - Tagesverbrauch an einem beliebigen Tag
  - Verbrauch einer Woche
- Anomalieerkennung:
  - Untypische Muster im Datensatz erkennen und begründen können
  - Echtzeit Erkennung

Als Datengrundlage dienen viertelstündliche Messwerte des Netzbezugs im Kanton Basel-Stadt, inklusive Netzverluste. Nicht berücksichtigt wird lokal erzeugter und direkt vor Ort verbrauchter Photovoltaikstrom. Der Datensatz enthält sowohl grundversorgte Kunden als auch freie Kunden, deren kombinierter Verbrauch die Netzlast bestimmt.

Ergänzend werden Wetterdaten mit einer zeitlichen Auflösung von zehn Minuten verwendet. Diese beinhalten unter anderem Messungen zu Niederschlag, Sonneneinstrahlung und Windstärke. Die Zielvariable des Modells ist der Gesamtstromverbrauch, also die Summe beider Kundengruppen, da dieser für Netzführung und Lastplanung massgeblich ist.

1.1.1 Projektziel

Das übergeordnete Ziel des Projekts ist die Entwicklung eines zuverlässigen und reproduzierbaren Machine Learning Systems zur Prognose des Stromverbrauchs.

Konkrete Ziele:

- Aufbauz eines konsistenten, reproduzierbaren Feature Sets aus den vorhandenen Daten

- Ableitung zusätzlicher relevanter Merkmale und Ergänzung durch weitere relevante Datensätze
- Training eines Regressionsmodells zur Vorhersage des Stromverbrauchs
- Bewertung der Modelgüte anhand transparenter Metriken, erklärbaren Modellen und Grafiken
- Entwicklung einer Anomalie Erkennung, im Vorhandenen Datensatz und als Klassifikationsmodell

Die Prognosen dienen insbesondere der Planungssicherheit, der Kosteneffizienz sowie der operativen Steürung. Sie unterstützt Entscheidungen zur Eigenproduktion, zm Marktzukauf von Strom sowie zur Optimierung von Speicher und Anlagebetrieb.

1.1.2 Persönliche Ziele

Mit der Freiheit im Projekt setzten wir uns unseren eigenen Bedingungen. Das Ziel war es Maschine-Learning Experten zu werden. Dabei wollten wir Anwendungsunabhängig bleiben und entschieden uns alle Aspekte des Projektes selber in Python zu programmieren. Auf den Datensatz wandten wir den CRISP-DM Zyklus einen Industrie Standard an, zur Datenaufbereitung, an. Wir waren also insgesamt sehr praxisorientiert und wollten, dass wir die Kenntnisse aus dem Klassenzimmer in die Privatwirtschaft bringen können.

2 Theorie (Alle)

2.1 CRISP-DM

Im Rahmen unseres ML-Projektes haben wir uns für die CRISP-DM-Zyklus-Vorgehensweise (Cross Industry Standard Process for Data Mining) entschieden. Dieses Vorgehensmodell gliedert den gesamten Datenanalyseprozess, von der Problemdefinition bis zur praktischen Nutzung des Modells, in sechs klar definierte Phasen.

CRISP-DM bietet uns eine strukturierte und iterative Vorgehensweise, um systematisch Daten zu verstehen, Modelle zu entwickeln und deren Nutzen im realen Umfeld zu evaluieren.

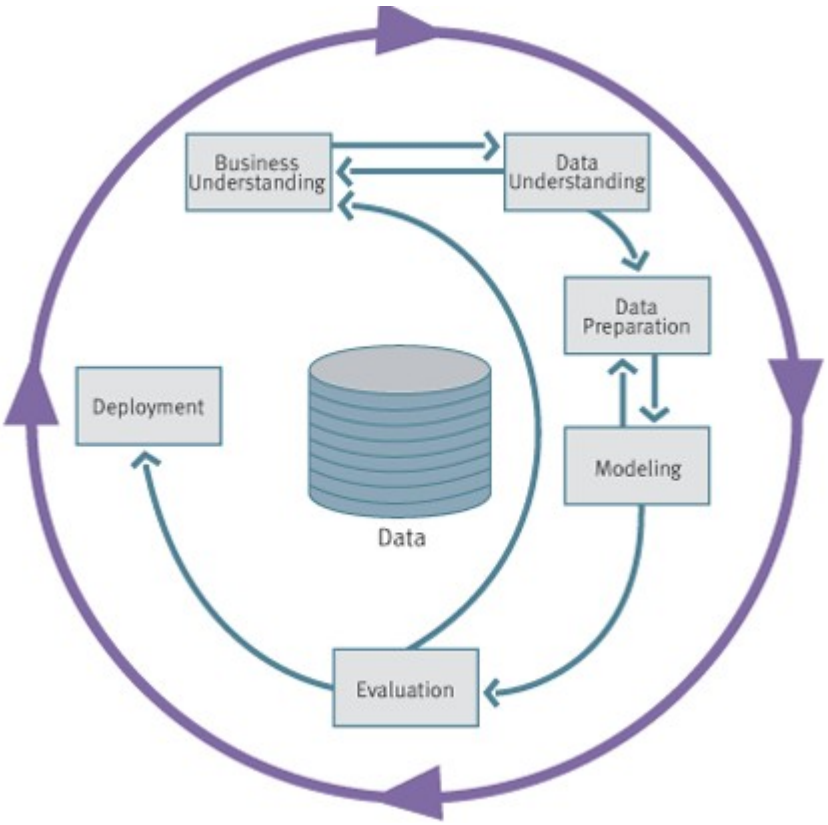


Abbildung 2 CRISP-DM-Zyklus

2.1.1 Business Understanding Data Understanding

In dieser Phase wird das Problem fachlich verstanden und in eine formalisierte Aufgabenstellung für das Machine Learning übersetzt.

Zudem erfolgt eine erste Analyse der vorhandenen Datenqüllen und -qualität.

2.1.2 Data Preperation

Die Daten werden aufbereitet, bereinigt und für das Modelltraining vorbereitet (Handling fehlender Werte,, Diskretisierung, Skalierung)

2.1.3 Modeling

Auswahl und Training des Modells (versch. Kriterien, Stärken & Schwächen und Parameter kennen)

2.1.4 Evaluation

Überprüfung der Modellleistung anhand geeigneter Metriken.

Ziel ist es, die Ergebnisse kritisch zu bewerten, Probleme zu erkennen und den wirtschaftlichen Nutzen zu beurteilen.

2.1.5 Deployment

Das entwickelte Modell oder die gewonnenen Erkenntnisse in die Praxis überführen.

(Das trainierte Prognosemodell wird als wiederverwendbare Python-Datei gespeichert. Ein separates Skript ruft das Modell regelmässig auf und erstellt täglich eine Vorhersage des Stromverbrauchs für den Folgetag. Die Prognosen werden automatisch in einer CSV-Datei oder als Diagramm gespeichert.

Dadurch ist eine einfache, reproduzierbare Nutzung der Ergebnisse möglich, ohne manülle Eingriffe.

2.2 Train/Test split

In der Entwicklung der Modellversionen für die Stromverbrauchsprognose in Basel-Stadt wurden wesentliche Erkenntnisse zur Validierungsstrategie und Modellgüte gewonnen. Bei der methodischen Herangehensweise standen primär das klassische Holdout-Verfahren sowie der TimeSeriesSplit zur Auswahl, wobei die Menge der Daten eine entscheidende Rolle spielt. Während in frühen Versionen ein Verhältnis von 80/20 genutzt wurde, hat sich im Projektverlauf ein Standardwert von 70/30 etabliert, der bereits sehr gute Ergebnisse liefert.

Ein zentraler Aspekt ist die Wahl des Zeitraums, die bei Stromverbrauchsdaten aufgrund der stabilen Lastprofile zwischen 2015 und 2024 weniger kritisch ist als beispielsweise bei Bitcoin-Chartdaten. Da die Verbrauchswerte im Testzeitraum nicht massiv von den Trainingsdaten abweichen, kann das Modell das Erlernte effektiv anwenden. Im Gegensatz dazu würde ein Training mit Bitcoin-Preisen von 2017 bei 100 Dollar für einen Test im Jahr 2025 bei 100.000 Dollar aufgrund der extremen Differenz scheitern.

Das Holdout-Verfahren stellt die Standardvariante dar, bei der das Datenset chronologisch in Training und Test gesplittet wird, was eine solide Basis ohne methodische Fehler bietet. Spezieller ist der TimeSeriesSplit, eine Form der Kreuzvalidierung für Zeitreihen. Eine normale Cross-Validation wäre hier falsch und sogar verheerend, da sie zu Data Leakage führen würde. Beim TimeSeriesSplit wird die zeitliche Kausalität gewahrt, indem Daten immer nur für die Zukunft getestet werden, die dem jeweiligen Trainingsblock folgt. Beispielsweise dienen die Jahre 2015 bis 2018 als Training für einen Test im Jahr 2019, woraufhin das Training bis 2019 erweitert wird, um das Jahr 2020 zu validieren. Die Anzahl dieser Durchläufe wird dabei über die sogenannten Folds bestimmt.

2.3 Algorithmen

2.4 Angewandte Tools

2.4.1 Framework

Programmiersprache:
Das Projekt wurde in der Programmiersprache Python umgesetzt. Diese wurde gewählt, da das Projektteam über die grössten Vorkenntnisse in Python verfügte und die Sprache besonders gut für die Datenaufbereitung und der Industrie Standard für die Entwicklung von Machine Learning Modellen geeignet ist. Dies ist durch die grosse Auswahl an Bibliotheken, die Python hat gewährleistet.

Entwicklungsumgebung:
Die Entwicklung fand in Visual Studio Code (VSC) statt, einer integrierten Entwicklungsumgebung zur Erstellung, Bearbeitung und Verwaltung von Softwareprojekten. Die Anwendung ist sehr flexibel und unterstützt verschiedene Sprachen und Extension.

Code Verwaltung:
GitHub wurde als Versionsverwaltungsplattform eingesetzt, um den Qüllcode zentral zu speichern und die gemeinsame Bearbeitung im Projektteam zu ermöglichen. Die Plattform basiert auf dem Versionskontrollsystem Git und erlaubt es, Änderungen am Code nachvollziehbar zu versionisieren. Durch Funktionen wie Versionshistorie, Branching und kollaborative Workflows unterstützt GitHub eine strukturierte Entwicklung, erhöht die Transparenz im Entwicklungsprozess und trägt zur Qualitätssicherung des Projekts bei.

2.4.2 Python Libraries

Ergänzend zur Programmiersprache Python wurde im Projekt eine Vielzahl von Bibliotheken eingesetzt, die Sammlungen vorgefertigter Programmfunktionen bereitstellen. Diese Bibliotheken kamen sowohl in der Datenaufbereitung als auch insbesondere im Modelling zum Einsatz. Die Nutzung etablierter Libraries ist im Bereich des Machine Learning üblich, da die vollständige Eigenimplementierung von Modellfunktionen den Rahmen des Projekts überschritten hätte.

Für die Verwendung der Bibliotheken wurde eine virtülle Umgebung eingerichtet. Dabei handelt es sich um eine isolierte Laufzeitumgebung, in der projektspezifische Abhängigkeiten unabhängig vom globalen System installiert und verwaltet werden. Dies stellt die Stabilität, Reproduzierbarkeit und Nachvollziehbarkeit der Softwareumgebung sicher.

Im Folgenden werden die im Projekt verwendeten Python Bibliotheken aufgeführt und kurz erläutert.

Operating System Interface (os)
Ermöglicht die Interaktion mit dem Betriebssystem, z. B. Zugriff auf Dateien, Verzeichnisse und Umgebungsvariablen. Unterstützt die plattformunabhängige Verwaltung von Dateipfaden.

System Specific Parameters and Functions (sys)
Bietet Zugriff auf systemnahe Parameter wie Laufzeitinformationen und den Python Suchpfad. Wird genutzt, um den Programmablauf zu steürn oder Modulpfade anzupassen.

Object Oriented File System Paths (pathlib)
Bietet eine objektorientierte Handhabung von Dateipfaden. Vereinfacht das Lesen, Schreiben und Verwalten von Dateien plattforMAEbergreifend.

Python Data Analysis Library (pandas)
Ermöglicht effiziente Datenmanipulation und Analyse mit Tabellenstrukturen wie DataFrames. Unterstützt Zeitreihen, Aggregationen und Feature Engineering.

Numerical Python (NumPy)
Stellt leistungsfähige Arrays und mathematische Funktionen bereit. Grundlage für numerische Berechnungen und viele Machine Learning Algorithmen.

Matplotlib Plotting Library (matplotlib.pyplot)
Ermöglicht die Erstellung von Diagrammen wie Linien, Histogrammen oder Streudiagrammen. Wird zur explorativen Datenanalyse und Visualisierung von Modellvorhersagen verwendet.

Statistical Data Visualization Library (Seaborn)
Erweitert Matplotlib um statistische Visualisierungen. Unterstützt die Darstellung von Verteilungen, Korrelationen und Trends.

Imbalanced Learn Random Over Sampling (imblearn RandomOverSampler)
Gleicht Klassenungleichgewichte durch Überabtastung der Minderheitsklasse aus. Verhindert Verzerrungen im Modelltraining.

Object Copying Utilities (copy)
Ermöglicht flache oder tiefe Kopien von Objekten. Verhindert unbeabsichtigte Änderungen an Datenstrukturen.

TensorFlow Machine Learning Framework (TensorFlow)
Framework für neuronale Netze und Deep Learning. Unterstützt effizientes Training grosser Modelle durch automatische Differenzierung.

Scikit-learn
Scikit-learn ist eine Bibliothek für Machine Learning, die Tools für Datenvorverarbeitung, Modelltraining und Evaluation bereitstellt. Im Projekt wurden insbesondere StandardScaler zur Feature-Standardisierung, train_test_split zur Datenaufteilung, LinearRegression als Baseline-Modell, RandomForestRegressor und GradientBoostingRegressor für präzisere Vorhersagen sowie Metriken wie MSE, MÄ und R^2 zur Bewertung der Modellleistung verwendet. Scikit-learn ermöglicht damit eine effiziente, reproduzierbare und modulare Umsetzung datengetriebener Modelle.

Regression Evaluation Metrics (Mean Squared Error, Mean Absolute Error, R Squared)
Messen die Genauigkeit von Regressionsmodellen. Quantifizieren Abweichungen zwischen vorhergesagten und tatsächlichen Werten.

Time Series Correlation Analysis Tools (Autocorrelation Function und Partial Autocorrelation Function)
Analysieren zeitliche Abhängigkeiten in Zeitreihen. Unterstützen die Auswahl geeigneter Lag-Strukturen für Modelle.

3 Vorgehensweise und Ergebnisse

3.1 Vorgehensweise

Am Anfang des Moduls haben wir ein Brainstorming gemacht und uns gefragt, in welche Richtung wir unser ML-Projekt gestalten möchten. Im Vorgänger-Modul «Maschine Learning» im FS25 konnten wir eine solide Grundkenntnis sammeln und mit der Anwendung «Tableau Prep» und «Orange» unsere erste Datenaufbereitung beginnen und unser erstes Model entwickeln & evaluieren. Schlussendlich haben wir uns für ein klassisches ML-Projekt, einer Regression entschieden, dass wir Anstelle einer Anwendung wie «Orange» mit Python durchführen wollten. Wie im vorherigen Kapitel beschrieben, haben wir uns für eine Vorgehensweise mit der CRISP-DM-Zyklus entschieden, damit wir unsere Grundkenntnisse vertiefen können und zusätzlich mit der Programmiersprache Python unsere ersten Modelle entwickeln können.

Die nachfolgenden Punkte unter behandeln unsere Vorgehensweisen, Erkenntnisse, sowie Ergebnisse im Projekt.

3.2 Business Understanding (Kerem)

In der Phase *Business Understanding* wurde die fachliche Problemstellung präzise definiert und in eine für Machine Learning geeignete Aufgabenstellung überführt. Ausgangspunkt war die Frage, wie sich der Stromverbrauch im Kanton Basel-Stadt auf Basis historischer Verbrauchs- und Wetterdaten zuverlässig prognostizieren lässt und welchen Mehrwert solche Prognosen für die Praxis bieten.

Der Fokus des Projekts liegt auf der Vorhersage des aggregierten Stromverbrauchs im IWB-Verteilnetz, da dieser für Netzführung und Lastplanung entscheidend ist. Prognosen werden auf unterschiedlichen zeitlichen Granularitäten erstellt (15 Minuten, Tag, Woche), um sowohl operative als auch planerische Fragestellungen abzudecken. Ergänzend wird untersucht, inwiefern Abweichungen zwischen prognostiziertem und tatsächlichem Verbrauch zur Erkennung ungewöhnlicher Verbrauchsmuster genutzt werden können.

Da IWB als einziger Netzbetreiber im Stadtgebiet fungiert, bilden die gemessenen Bezugsdaten den realen Energiefluss im Netz vollständig ab. Unabhängig davon, ob Kunden grundversorgt sind oder ihren Strom am freien Markt beziehen, ist deren Last für die Netzplanung gleichermassen relevant. Der geschäftliche Nutzen des Modells ergibt sich insbesondere aus einer verbesserten Planungssicherheit, einer effizienteren Nutzung von Eigenproduktion und Marktzukauf sowie aus der Unterstützung operativer Entscheidungen.

Auf Basis dieses Verständnisses wurde das Problem als Regressionsaufgabe formuliert, bei der der zukünftige Stromverbrauch als Zielvariable dient. Die daraus abgeleiteten Anforderungen bilden die Grundlage für die nachfolgenden Phasen der Datenaufbereitung, Modellierung und Evaluation im CRISP-DM-Zyklus.

3.3 Data Understanding

Bevor die eigentliche Datenaufbereitung (Data Preparation) beginnen konnte, war eine fundierte Auseinandersetzung mit der Herkunft und dem Nutzen der Datenqüllen notwendig. Das Projekt stützt sich auf zwei Hauptpfeiler:

1. **Stromverbrauchsdaten:** Bezogen aus dem Open-Data-Portal Basel-Stadt, umfassend den Zeitraum von 2012 bis 2025 in einem hochauflösenden 15-Minuten-Intervall.
2. **Meteorologische Daten:** Bezogen vom Bundesamt für Meteorologie (MeteoSchweiz) für den Zeitraum 2010 bis 2024 in 30-Minuten-Intervallen.

Aufgrund der Vielzahl an Wetterstationen in der Region Basel gestaltete sich die Auswahl repräsentativer Daten als schwierig. Wir haben uns daher für die Homogenreihe der Station Binningen entschieden. Obwohl sich diese im Kanton Basel-Landschaft befindet, sind die Daten aufgrund ihrer Konsistenz und räumlichen Nähe als Referenz für das gesamte Stadtgebiet geeignet. Insgesamt umfasst der finale Datensatz 39 Features (27 meteorologische und 12 verbrauchsbasierte) mit einer Gesamtanzahl von über 20 Millionen Einzeldatenpunkten.

IWB ist Versorger und zugleich Betreiber des einzigen Verteilnetzes im Stadtgebiet. Jeglicher aus dem öffentlichen Netz bezogene Strom läuft über die IWB-Infrastruktur unabhängig davon, ob es sich um grundversorgte Kunden (Bezug direkt von IWB) oder freie Kunden (Einkauf am Markt) handelt. Die Bezugsdaten spiegeln somit den tatsächlichen Energiefluss durch das Netz wider.

Besonderheiten der Strommessung und Solarstrom:

- **Solar-Exklusion:** Lokale Solarenergie ist nicht als separater positiver Erzeugungsfaktor in den Messungen enthalten.
- **Indirekte Sichtbarkeit:** Der Einfluss von Photovoltaik ist in den Daten indirekt erkennbar: Erzeugt eine lokale Anlage Strom für den Eigenverbrauch, sinkt der messbare Bezug aus dem öffentlichen Netz entsprechend.
- **Datenqualität:** IWB garantiert eine vollständige Dokumentation aller Energieflüsse, indem fehlende Messwerte durch Netzbilanzdaten ergänzt werden.

3.3.1 Grafische Datenanalyse

Mithilfe der Python-Bibliothek *Matplotlib* wurden die Daten visualisiert, um Trends, Muster und Anomalien zu identifizieren. Die wichtigsten Erkenntnisse der grafischen Analyse werden nachfolgend schrittweise erläutert:

3.3.1.1 *Plot 01: Gesamte Zeitreihe (2012–2025) – Der langfristige Trend*

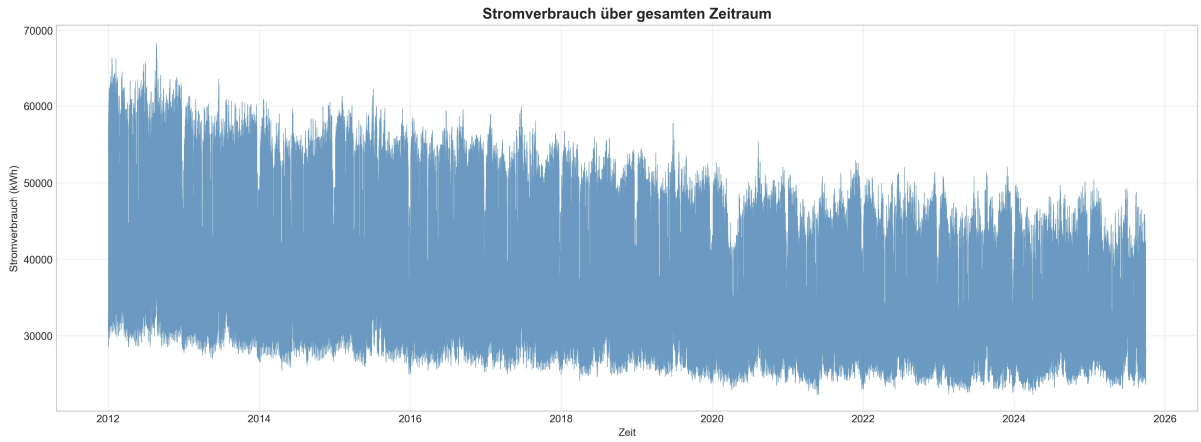


Abbildung 3 Stromverbrauch über gesamten Zeitraum

Diese Darstellung visualisiert den gesamten Stromverbrauch über den Zeitraum von 13 Jahren.

- **Abwärtstrend:** Es ist ein deutlicher, kontinuierlicher Rückgang des Verbrauchs über die Jahre erkennbar. Dies lässt auf eine gesteigerte Energieeffizienz oder einen bewussteren Umgang mit Ressourcen schliessen.
- **Corona-Anomalie:** Im Jahr 2020 zeigt sich eine signifikante Abweichung (Anomalie). Während der Hochphase der Pandemie und der damit verbundenen Lockdowns sank der Stromverbrauch aufgrund der eingeschränkten öffentlichen und wirtschaftlichen Aktivität spürbar.

3.3.1.2 *Plot 03: Der Tag-Nacht-Zyklus*

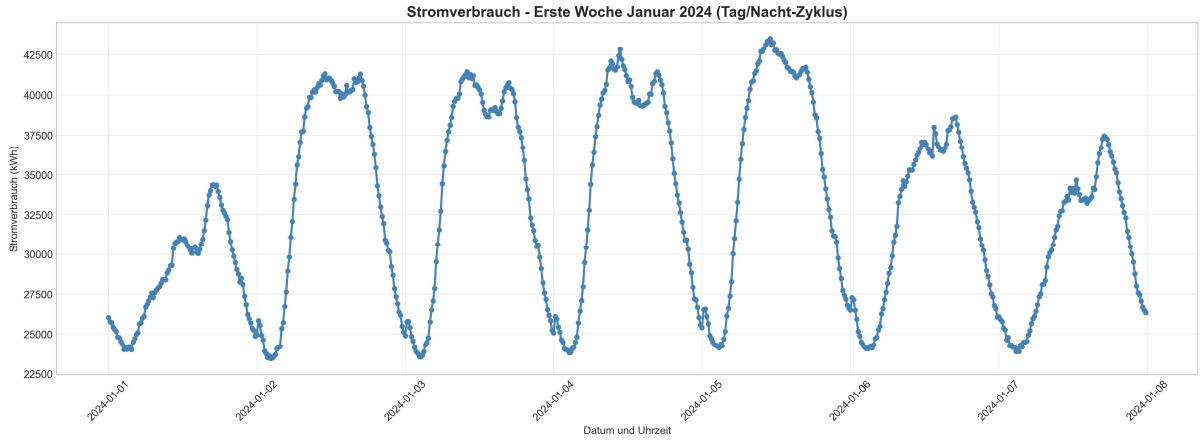


Abbildung 4 Stromverbrauch Jan 24 Woche

Durch das "Hineinzoomen" auf das Intervall einer einzelnen Woche wird die tägliche Dynamik sichtbar.

- **Zyklizität:** Der Graph zeigt ein rhythmisches Muster. Tagsüber steigt der Verbrauch massiv an, während er in der Nacht auf ein Minimum sinkt, sobald Haushalte und viele Gewerbebetriebe ihre Aktivität reduzieren.

3.3.1.3 Plot 06: Boxplot nach Wochentagen, Werktage vs. Wochenende

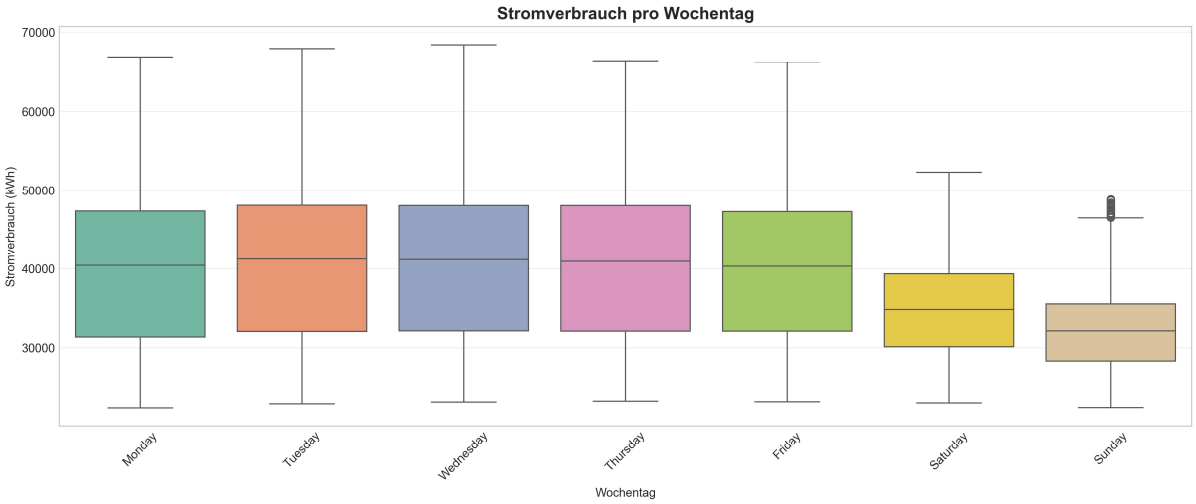


Abbildung 5 Stromverbrauch pro Wochentag (Boxplots)

In dieser Analyse wurde der Stromverbrauch gemittelt und nach den Wochentagen (Montag bis Sonntag) gruppiert dargestellt.

- **Wochenend-Effekt:** Es zeigt sich ein klares Muster: An den Wochenendtagen (Samstag und insbesondere Sonntag) ist der Stromverbrauch deutlich geringer als an den Werktagen. Dies ist direkt auf die Schliessung vieler Industriebetriebe, Fabriken und Büros zurückzuführen.

3.3.1.4 Plot 12: Korrelations-Heatmap mit Merkmalsbeziehungen

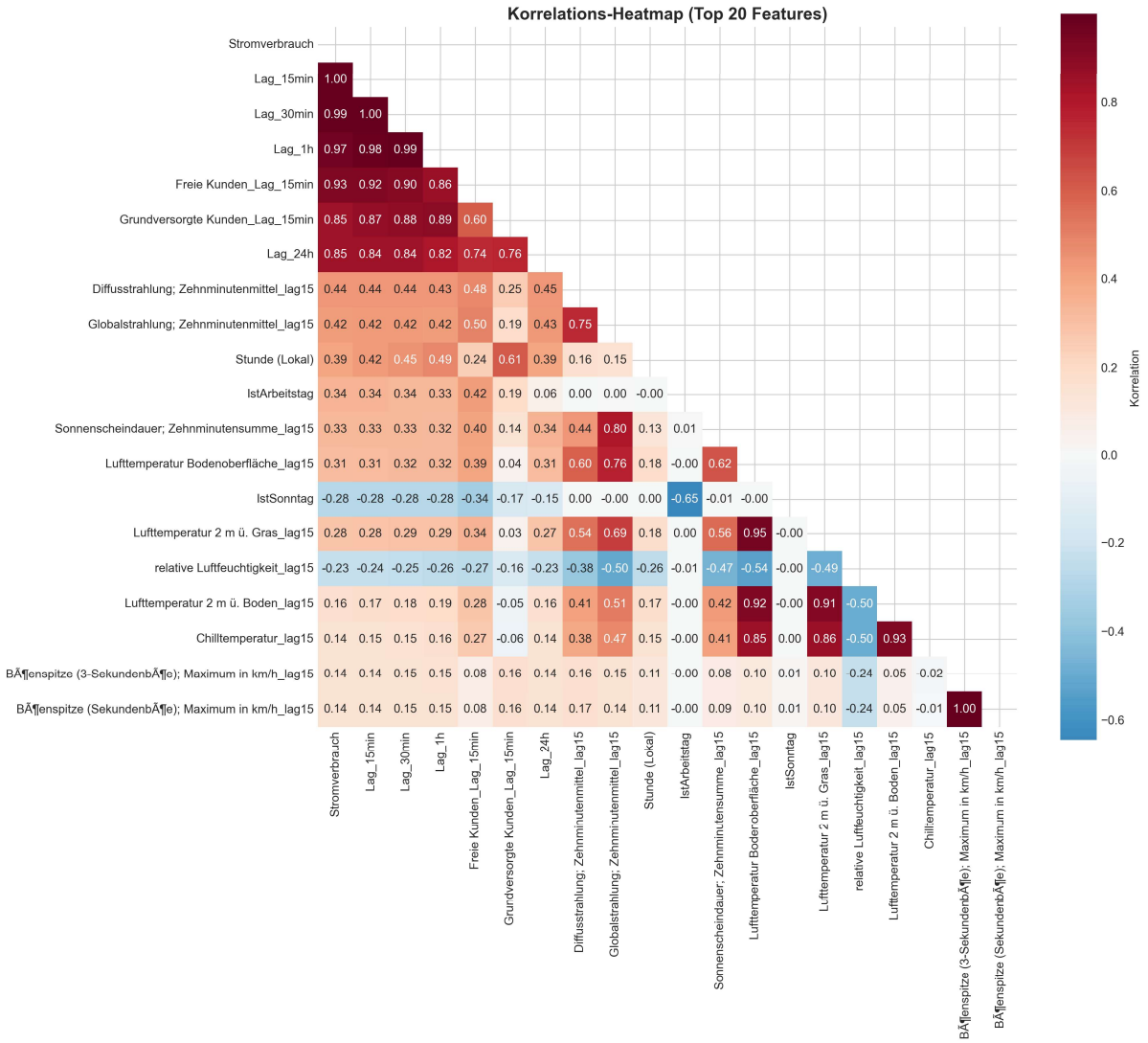


Abbildung 6 Korrelations-Heatmap Featurebeziehungen

Die Heatmap visualisiert die statistischen Zusammenhänge zwischen den 39 Features.

- **Starke Autokorrelation:** Besonders auffällig ist die extrem starke Korrelation bei Lag-Features (z. B. `Lag_15min`). Das bedeutet, der Verbrauch vor 15 Minuten ist ein sehr präziser Indikator für den aktüllen Verbrauch.
- **Relevanz:** Wie wir mit dieser dominanten Korrelation umgehen, ob sie die Vorhersage stützt oder zu "Lazy Learning" führt, wird im weiteren Verlauf der Dokumentation bei der Modellentwicklung kritisch bewertet.

3.4 Data Preparation

Die in dieser Arbeit verwendeten Datenqüllen wurden im vorhergehenden Abschnitt Data Understanding inhaltlich analysiert und fachlich eingeordnet.

Der folgende Abschnitt beschränkt sich bewusst auf die Erfassung und Bereitstellung der verwendeten Rohdaten. Bewertungen zur Datenqualität, zeitlichen Auflösung, Repräsentativität sowie fachliche Annahmen werden ausschliesslich im Abschnitt Data Understanding behandelt.

3.4.1.1 Data Collection

Für die Analyse wurden Stromverbrauchsdaten aus dem Messsystem der IWB sowie meteorologische Zeitreihendaten aus einer externen Wetterqülle bezogen.

Events und externe Feature als zusätzliche Datenqüllen wie Events in Basel, Ferien, Nacht-Zyklus, Schulferien, Feiertagse und Touristenzeiten (Fantasy Basel, ArtBasel, Herbstmesse) konnten wir leider aus zeitlichen Gründen nicht mit im Projekt implementieren, jedoch sind diese für die Optimierung geplant.

Beide Daten wurden in ihrer Rohform geladen und für die weiteren Verarbeitungsschritte bereitgestellt. In diesem Schritt erfolgten keine strukturellen Anpassungen oder inhaltlichen Transformationen.

3.4.1.2 Data Preprocessing

3.4.1.2.1 Strom

Zeiten

Zu Beginn hatten wir zwei Optionen als Index für den die Daten vom Stromverbrauch: «Start der Messung» und «Start der Messung (Text)» Später konnten wir herausfinden, dass die Variante mit «(Text)» die lokale Zeit beinhaltete und die die ohne waren die Intervalle in UTC-Zeit. Schlussendlich haben wir die beiden Features umbenannt und haben, die UTC-Zeiten als Index verwendet und die mit den Lokalen im CSV drin gelassen und später beim Modelltraining ausselektiert.

Des Weiteren konnten wir feststellen, dass die Messungen mit lokaler Zeit getätigt wurden. Uns war bewusst, wenn wir die UTC-Zeiten als Index für unser Model verwenden möchten, die Kalenderfeatures wie Tag, Monat, etc. alle lokale Werte enthalten.

Da die Stromverbrauchsdaten in lokaler Zeit (Europe/Zurich) gemessen wurden, führt dies jedes Jahr zu folgenden Effekten:

März – Zeit wird vorgestellt:

- Die Stunde 02:00–03:00 existiert nicht.
- → 4 fehlende 15-Minuten-Messwerte. (bei Lokaler Zeit)

Oktober-Zeit wird zurückgestellt. Die Stunde 02:00–03:00 würde doppelt vorkommen.Da das Messsystem nun keine doppelten lokalen Zeitstempel speichern darf, wird eine der beiden Stunden verworfen. Dies war besonders komplex für uns, da wir bei der Zeitumstellung im Winter doppelte Messungen bei der lokalen Zeitangabe und keinen Zeitunterbruch bei den UTC-Zeiten erwarteten. Schlussendlich gab es ebenfalls 4 fehlende Messwerte aufgrund der Zeitumstellung.

Intervalle

Des Weiteren wurde im Rahmen der Datenaufbereitung geprüft, ob eine Aggregation der Zeitreihen von 15 Minuten auf 30 Minuten sinnvoll ist. Diese Überlegung ergab sich aus der unterschiedlichen zeitlichen Auflösung der verwendeten Datenqullen. Die meteorologischen

Daten liegen in einer 10-Minuten-Auflösung vor und könnten durch einfaches Sampling ohne Interpolation auf ein 30-Minuten-Intervall reduziert werden. Auch die Stromverbrauchsdaten in 15-Minuten-Auflösung lassen sich durch Sampling problemlos auf 30 Minuten überführen. Zudem hätte eine geringere zeitliche Auflösung die Anzahl der Datenpunkte reduziert, was potenziell zu stabileren Modellen führen kann.

Nach fachlicher Bewertung wurde diese Option jedoch verworfen. Die IWB misst und verrechnet Stromverbrauch standardmässig in 15-Minuten-Intervallen, ebenso basiert der europäische Stromhandel auf dieser zeitlichen Granularität. Eine Aggregation auf 30 Minuten würde relevante kurzfristige Verbrauchsschwankungen glätten und insbesondere kurzzeitige Anomalien verfälschen, da diese häufig eine Dauer von weniger als 30 Minuten aufweisen. Die meteorologischen Daten lassen sich hingegen ohne inhaltlichen Informationsverlust von 10 auf 15 Minuten interpolieren.

Fehlende Werte

Über alle Jahre fehlen genau 52 Zeitpunkte.Diese Werte sind real nie gemessen worden und können ohne Informationsverlust entfernt werden.

Zusätzlich konnten wir im Datensatz beobachten, dass der Stromverbrauch von Grundversorgten und Freien Kunden erst ca. ab dem Jahr 2020 eingeführt wurden. Bevor wir uns entschieden haben, wie wir nun mit diesen zwei Features weiterfahren, wollten wir bei der Modellierung testen, ob wir überhaupt einen erheblichen unterschied mit diesen Features erzielen können. Des Weiteren haben wir im späteren Verlauf festgestellt, dass diese Features Data Leakage beinhalten. Da der zu vorhersagender Stromverbrauch nochmals in zwei Kundengruppen unterteilt ist, beinhalten diese Werte, die zum Zeitpunkt der Vorhersage nicht bekannt sind. Im folgenden Kapitel Feature Creation stellen wir unsere mögliche Lösung vor

3.4.1.2.2 Wetter

Join

Wie bereits erwähnt waren die Wetterdaten in 10-Minuten-Intervallen verfügbar. Die Zeitstempel waren in UTC-Zeiten und es gab keine Fehlenden Werte im Datensatz.

Im Ersten Schritt haben wir den Index vom Datenset des Stromverbrauchs als Schablone verwendet und im Zweiten Schritt die numerischen indexiert und interpoliert und die kategorischen Spalten mittels merge_asof mit Toleranz 7:30 min reindexiert.

Data Leakage

Bei zeitabhängigen Prognosemodellen darf ausschliesslich Information verwendet werden, die zum Prognosezeitpunkt tatsächlich verfügbar ist. Meteorologische Messwerte zum gleichen Zeitpunkt wie der Zielwert können in der realen Anwendung nicht bekannt sein und dürfen daher nicht direkt verwendet werden. Beispielsweise kann man den Stromverbrauch um 12:00 Uhr nicht mit der Temperatur um 12:00 Uhr vorhersagen, weil man die Temperatur um 12:00 Uhr zu diesem Zeitpunkt nicht kennt.

Eine mögliche Erweiterung des Modells wäre die Anbindung einer externen Wetterprognose über eine API, um zukünftige Wetterwerte einzubeziehen. Aus zeitlichen Gründen wurde diese Option in dieser Arbeit nicht umgesetzt. Stattdessen wird ein alternativer Ansatz verwendet, der ausschliesslich auf historisch verfügbaren Wetterinformationen basiert. Die konkrete Umsetzung wird in Kapitel Feature Creation beschrieben.

Datentypen

Daten wie Monat, Wochentag, Quartel und etc.haben wir zu kategorischen Daten konvertiert. Textülle Features wie „stationabbr“ haben wir entfernt, da diese für das Modelltraining irrelevant sind.

3.4.1.3 Feature Engineering

3.4.1.3.1 Feature Creation

Im Rahmen der Feature-Erstellung wurden zusätzliche erklärende Variablen abgeleitet. Dazu gehört eine boolesche Variable zur Unterscheidung zwischen Arbeitstagen und Sonntagen. Zudem wurden Lag-Features auf Basis historischer Stromverbrauchsdaten erstellt, um zeitliche Abhängigkeiten und wiederkehrende Muster im Verbrauchsverhalten abzubilden.

Zusätzlich wurde ein Feature Diff_15 definiert, welches die Differenz zwischen dem Stromverbrauch zum Zeitpunkt t und dem Verbrauch zum Zeitpunkt $t-15$ Minuten beschreibt. Dieses Feature erfordert jedoch den aktuellen Verbrauchswert und ist daher nur unter bestimmten Annahmen zulässig. Der Umgang mit zeitlich nicht verfügbaren Informationen sowie die daraus resultierenden Anpassungen der Feature-Definitionen werden untenstehenden Kapitel behandelt.

Data Leakage

Zur Vermeidung von Data Leakage werden alle zeitkritischen Variablen ausschliesslich zeitlich verzögert in das Modell eingebunden. Dies betrifft insbesondere Wetterdaten sowie die Variablen Grundversorgte Kunden und Freie Kunden, die jeweils mit einem Versatz von 15 Minuten verwendet werden.

Das Feature Diff_15 wurde so definiert, dass es ausschliesslich auf vergangenen Werten basiert, beispielsweise als Differenz zwischen dem Stromverbrauch vor 15 und vor 30 Minuten. Dadurch wird sichergestellt, dass keine Informationen aus dem Prognosezeitpunkt in das Modell einfliessen.

3.4.1.3.2 Feature Selection

Selektierte Features:
Kalenderbasierte Zeitfeatures aus dem Stromverbrauch wurden in jedem Fall verwendet. Lag-Features des Stromverbrauchs kamen bedingt zum Einsatz, während bei den Wetterdaten sämtliche nicht redundanten Messgrössen beibehalten wurden; die Reduktion redundanter Wettervariablen wird im folgenden Abschnitt beschrieben.

Ausselektierte Features:
Zur Vermeidung von Redundanz und unnötiger Komplexität wurde das Feature-Set reduziert.

Entfernte Features (Wetter & Kundenvariablen):

- Grundversorgte Kunden_Lag_15min
- Freie Kunden_Lag_15min
- Böenspitze (3-Sekundenböe) in km/h und m/s (Lag 15)
- Böenspitze (Sekundenböe) in km/h (Lag 15)
- Luftdruck reduziert auf Meeresniveau (QFF, QNH) (Lag 15)

- Luftdruck auf Barometerhöhe (Lag 15)
- Lufttemperatur 2 m über Gras (Lag 15)
- Lufttemperatur Bodenoberfläche (Lag 15)
- Windgeschwindigkeit vektoriell (Lag 15)
- Windgeschwindigkeit (Zehnminutenmittel) in km/h (Lag 15)
- Windrichtung (Zehnminutenmittel) (Lag 15)
- Relative Luftfeuchtigkeit (Lag 15)

Ebenfalls haben wir die textuelle Variable Station, Time, Date, Date and Time (aus Wetterqulle) entfernt. Bei den Wetterdaten hatten wir ebenfalls Datums-Features nach dem Joinen. Die mehreren DateTime-Features mussten auf eine Variante reduziert werden, um Redundanzen zu vermeiden.

Stromverbrauchsdaten:
Bei den Stromverbrauchsdaten wurden keine Features entfernt. Die vorhandenen Variablen sind konsistent, nicht redundant und fachlich sinnvoll, sodass eine weitere Reduktion keinen Mehrwert gebracht hätte.

Ziel der Feature-Reduktion:
Ziel war ein kompaktes, informationsreiches Feature-Set ohne redundante Messungen. Die Reduktion betraf ausschliesslich die Wetterdaten, da dort mehrere Variablen dieselbe physikalische Information in unterschiedlichen Varianten abbildeten.

3.5 Modeling

Im Schritt Modeling wurden geeignete Prognosemodelle für den Stromverbrauch in 15-Minuten-Intervallen entwickelt und implementiert. Dazu wurden mehrere Forecast-Setups (One-Step, Multi-Output und rekursiver Multi-Step) definiert, passende Features zusammengestellt und verschiedene Modellkandidaten (LightGBM, XGBoost, Random Forest, Linear Regression) trainiert. Ziel dieses Abschnitts ist die Beschreibung des Vorgehens, der umgesetzten Modellvarianten sowie der wichtigsten Erkenntnisse aus der Modellierungsphase; eine vertiefte Bewertung der Modellgüte folgt im Schritt Evaluation.

3.5.1 Modellierungsstrategie und Forecast-Setups

Zur Abbildung der unterschiedlichen Anforderungen an die Stromverbrauchsprognose wurden mehrere Forecast-Setups definiert. Diese unterscheiden sich hinsichtlich Prognosehorizont, Modellstruktur und Einsatzgebiet. Ziel war es, sowohl kurzfristige Vorhersagen als auch stabile 24-Stunden-Prognosen abzudecken und die jeweiligen Stärken und Schwächen der Ansätze zu analysieren.

3.5.1.1 One-Step-Forecast (t+1)

Beim One-Step-Forecast sagt das Modell jeweils den nächsten 15-Minuten-Wert voraus. Dieses Setup ist direkt mit dem Business Case verknüpft, da kurzfristige Prognosen insbesondere für operative Entscheidungen relevant sind, beispielsweise zur kurzfristigen Laststeuerung oder zur schnellen Reaktion auf unerwartete Verbrauchsänderungen.

Der One-Step-Ansatz zeichnet sich durch eine hohe Reaktionsfähigkeit aus, da das Modell ausschliesslich auf die unmittelbare Zukunft optimiert wird. Gleichzeitig dient er als Referenz, um die kurzfristige Prognosequalität der eingesetzten Modelle isoliert zu bewerten.

3.5.1.2 Multi-Output Forecast (t+1 – t+96)

Der Multi-Output-Forecast stellt eines der zentralen Modellierungs-Setups dieser Arbeit dar. In diesem Ansatz sagt ein einzelnes Modell gleichzeitig alle 96 Werte für die nächsten 24 Stunden voraus und prognostiziert damit den vollständigen Tageslastgang in einem Schritt.
Dieses Setup ist eng mit dem Business Case verknüpft, da für operative und taktische Entscheidungen häufig konsistente 24-Stunden-Prognosen benötigt werden. Im Gegensatz zum One-Step-Forecast werden hier tageszeitliche Muster und typische Lastverläufe explizit modelliert.
Ein wesentlicher Vorteil dieses Ansatzes liegt darin, dass sich Prognosefehler nicht schrittweise fortpflanzen, da alle Prognosehorizonte direkt vorhergesagt werden.

3.5.1.3 Rekursiver Multi-Step Forecast (t+1 – t+96)

Beim rekursiven Multi-Step-Forecast wird ein One-Step-Modell iterativ angewendet. Die jeweils vorhergesagten Werte werden als Eingabe für den nächsten Prognoseschritt verwendet, bis der gesamte 24-Stunden-Horizont erreicht ist.

Dieses Setup bildet ein praxisnahes Anwendungsszenario ab, da es dem Einsatz eines einzelnen kurzfristigen Modells entspricht und längere Prognosen schrittweise aufbaut. Gleichzeitig ermöglicht dieser Ansatz einen direkten Vergleich mit dem Multi-Output-Forecast hinsichtlich Stabilität und Fehlerverhalten über längere Prognosehorizonte.

3.5.1.4 Baseline-Modell

Als Baseline wurde ein einfaches Zeitreihenmodell eingesetzt, das ausschliesslich zeitliche Muster aus den historischen Daten lernt und den Stromverbrauch für die gesamte Testperiode vorhersagt. Dieses Modell dient nicht der optimalen Prognose, sondern als Referenz, um den Mehrwert der komplexeren Machine-Learning-Modelle einzuordnen.

3.5.2 Feature Engineering und Feature-Set

Das Feature-Set wurde in Kalender-, Lag- und Wetter-Features unterteilt und im Code so umgesetzt, dass diese je nach Experiment flexibel aktiviert oder deaktiviert werden konnten. Zeitliche Variablen wie Monat, Wochentag, Stunde und Tag des Jahres wurden initial mittels One-Hot-Encoding abgebildet, jedoch aufgrund der hohen Dimensionalität durch zyklische Sinus- und Kosinus-Transformationen ersetzt. Dadurch konnten periodische Muster kompakt und effizient modelliert werden.

Die Lag-Features bilden vergangene Verbrauchswerte in unterschiedlichen zeitlichen Abständen ab und stellten eine zentrale Informationsqülle für die Prognose dar. Wetter-Features wurden ausschliesslich in verzögerter Form integriert, um zeitliche Kausalität sicherzustellen. Es zeigte sich, dass bei 24-Stunden-Prognosen insbesondere Kalender- und Lag-Features relevant sind, während bei kurzfristigen One-Step-Prognosen vor allem Lag-Features den grössten Beitrag zur Vorhersage leisten.

3.5.3 Datenaufteilung

Die Daten wurden zeitlich konsistent in Trainings- und Testdaten im Verhältnis 70 % zu 30 % aufgeteilt. Als Betrachtungszeitraum wurde der 31.08.2020 bis 31.12.2024 gewählt. Es wurde bewusst darauf verzichtet, den maximal verfügbaren Zeitraum zu verwenden, da sehr frühe Daten unter anderem durch COVID-19 und vergangene Krisen stark verzerrte Verbrauchsmuster enthalten und für ein erstes stabiles Modell wenig repräsentativ sind.

Zusätzlich war der gewählte Zeitraum durch die Verfügbarkeit der Wetter-Features begrenzt, welche nur bis Ende 2024 vorlagen. Auch die Lag-Features für freie Kunden und grundversorgte Kunden waren erst ab 2020 vollständig verfügbar, weshalb der Analysezeitraum entsprechend angepasst wurde.

3.5.4 Modelkandidaten und Auswahl

Zu Beginn wurden lineare Regression, Random Forest und Gradient Boosting als Modellkandidaten evaluiert. Dabei zeigte sich, dass Gradient Boosting auf den vorliegenden Daten die besten Prognosemetriken erzielte und Random Forest leicht übertraf. In einem weiteren Schritt wurde LightGBM getestet, welches sowohl eine schnellere Trainingszeit als auch eine bessere Performance aufwies. Dies ist unter anderem auf die histogrammbasierte Verarbeitung der Entscheidungsbäume zurückzuführen. LightGBM wurde daher als Hauptmodell für die weiteren Forecast-Setups verwendet.

Ergänzend wurde Prophet für langfristige Tagesprognosen eingesetzt, da das Modell explizit Trends und Saisonalitäten abbildet. Aufgrund von Concept Drift sowie strukturellen Veränderungen im Stromverbrauch (z. B. Krisen oder Nachfrageverschiebungen) waren die Prognosen jedoch weniger stabil. Dennoch erreichte Prophet bei Tagesprognosen einen MAE von rund 100 000 sowie einen R2 von 0.795, was für eine grobe langfristige Einordnung ausreichend ist, jedoch nicht für operative Kurzfristprognosen geeignet ist.

Als Auswahlkriterien für das Hauptmodell wurden primär RMSE und MAE auf dem Testdatensatz herangezogen. Beim Multi-Output-Ansatz wurden ergänzend Block-Metriken über 24 Stunden sowie horizontweise Fehler betrachtet, um die Qualität der gesamten Prognoseperiode zu bewerten.

3.5.5 Training und Implementations-Setup

Das Training der Modelle erfolgte über einen einheitlichen Pipeline-Ansatz, bestehend aus Feature-Auswahl, einfachem Preprocessing (Passthrough) und dem jeweiligen Regressionsmodell. Zentrale Parameter wie Feature-Set, betrachteter Zeitraum und Testanteil wurden konfigurierbar umgesetzt, um verschiedene Experimente reproduzierbar durchführen zu können. Die trainierten Modelle wurden mittels joblib gespeichert und durch die Verwendung fester random_state-Werte wurde die Reproduzierbarkeit der Resultate sichergestellt

3.5.6 Ergebnisse aus dem Modeling

Für alle drei Forecast-Setups (One-Step, Multi-Output und rekursiv) wurden Modelle trainiert, verglichen und jeweils ein bestes Modell identifiziert. Der Multi-Output-Ansatz lieferte konsistente 24-Stunden-Prognosen, während der rekursive Ansatz eine alternative Modellierungslogik für denselben Prognosehorizont darstellt. Insgesamt zeigten Gradient-Boosting-Modelle, insbesondere LightGBM, eine deutlich bessere Performance als einfache Baseline-Modelle.

Die Analyse der Feature-Relevanz mittels Permutation Importance zeigte, dass Lag-Features sowie zyklische Kalenderfeatures den grössten Beitrag zur Vorhersage leisten. Wetter-Features hatten einen ergänzenden, jedoch insgesamt geringeren Einfluss. Diese Erkenntnisse bestätigen die gewählte Modellierungsstrategie und die Fokussierung auf zeitliche Muster im Stromverbrauch.

3.6 Evaluation & Interpretation

3.6.1 Evaluationsstrategie

Die Evaluation erfolgte getrennt für kurzfristige (One-Step) sowie für 24h-Forecasts. Der Fokus lag dabei auf den Testdaten, um die Generalisierungsfähigkeit der Modelle zu beurteilen.

Zur Abbildung einer realistischen Anwendungssituation wurde neben einer globalen Bewertung zusätzlich ein business-nahes 24h-Block-Setup verwendet, bei dem die Vorhersagen jeweils um 00:00 Uhr lokaler Zeit gestartet werden. Dieser Ansatz entspricht dem operativen Einsatz eines Day-Ahead-Lastgang-Forecasts.

3.6.2 Modellbewertung

Zur Bewertung wurden die Metriken MAE, RMSE und R^2 herangezogen. Beim Multi-Output-Modell erfolgte die Evaluation sowohl global über alle Prognosehorizonte als auch blockweise in einem 24h-Setup (Start 00:00 lokal). Beim rekursiven Ansatz wurde zwischen der Qualität des zugrunde liegenden One-Step-Modells (t+1) und der daraus abgeleiteten 24h-Mehrschritt-Prognose unterschieden.

Für den eigentlichen Modellvergleich wurde primär das 24h-Block-Setup auf den Testdaten verwendet, da dieses den operativen Zielkontext eines Day-Ahead-Lastgang-Forecasts am besten widerspiegelt.

Zur Einordnung der Metriken ist zu berücksichtigen, dass sich der prognostizierte Stromverbrauch typischerweise im Bereich von 25’000 bis 45’000 Einheiten pro 15-Minuten-Intervall bewegt. Ein MAE unter 500 ist daher als sehr gut zu bewerten, Werte zwischen 500 und 1’000 gelten als gut und für die operative Planung geeignet, während Werte über 1’500 auf eine eingeschränkte Prognosequalität hindeuten. Analog dazu sind RMSE-Werte unter 1’000 als stabil zu betrachten, während deutlich höhere Werte auf grössere Abweichungen bei Lastspitzen hinweisen. Ein R^2 oberhalb von 0.95 signalisiert eine hohe Erklärungskraft des Modells, während niedrigere Werte auf eine geringere Stabilität im Tagesverlauf schliessen lassen.

3.6.3 Ergebnisse und Modellwahl

```
=== MODELLVERGLEICH (1-Step) ===
      MAE_train  RMSE_train  R2_train  MAPE_%_train    MAE_test   RMSE_test   R2_test  MAPE_%_test
lgbm  298.14805  394.222264  0.997115      0.859917  412.544204  564.303486  0.993507     1.188092
```

Abbildung 7 Metriken 1-Step-Forecast

Das Modell des One-Step_forecasts erreicht sowohl auf Trainings- als auch auf Testdaten sehr gute Fehlerwerte sowie ein hohes Bestimmtheitsmass. Diese Ergebnisse bestätigen, dass das zugrunde liegende Regressionsmodell kurzfristige Abhängigkeiten im Stromverbrauch konsistent und stabil abbilden kann.

```
=== REKURSIV 24h BLOCK Start 00:00 lokal (Train) ===
Tage ausgewertet: 1108
MAE_mean  : 516.30 | RMSE_mean: 660.73 | R2_mean: 0.9575 | MAPE_mean: 1.51 %
MAE_median: 451.20

=== REKURSIV 24h BLOCK Start 00:00 lokal (Test) ===
Tage ausgewertet: 474
MAE_mean  : 858.15 | RMSE_mean: 1073.77 | R2_mean: 0.8632 | MAPE_mean: 2.48 %
MAE_median: 677.15
```

Abbildung 8 Metriken 24h- Multi-Step-Forecast (rekursive)

Im Test erreicht der rekursive 24h-Multi-Step-Forecast im 24h-Block-Setup (Start 00:00 lokal) einen deutlich tieferen mittleren absoluten Fehler (MAE) sowie ein hoehes R^2 im Vergleich zum folgenden Multi-Output-Modell. Die Ergebnisse zeigen, dass der rekursive Ansatz robuste Tageslastgänge erzeugt und sich stabil auf unbekannte Daten überträgt.

```
=== MULTI-OUTPUT 24h GLOBAL (Test)  ===
MAE       : 1069.02
RMSE      : 1551.08
R2        : 0.9510
MAPE      : 3.10 %
```

```
=== MULTI-OUTPUT 24h GLOBAL (Train) ===
MAE       : 568.11
RMSE      : 744.87
R2        : 0.9897
MAPE      : 1.63 %
```

```
=== MULTI-OUTPUT 24h BLOCK Start 00:00 lokal (Train) ===
Tage ausgewertet: 1108
MAE_mean  : 701.25 | RMSE_mean: 866.82 | R2_mean: 0.9649 | MAPE_mean: 2.03 %
MAE_median: 649.74

=== MULTI-OUTPUT 24h BLOCK Start 00:00 lokal (Test) ===
Tage ausgewertet: 475
MAE_mean  : 1405.26 | RMSE_mean: 1723.25 | R2_mean: 0.7809 | MAPE_mean: 4.07 %
MAE_median: 1234.68
```

Abbildung 9 Metriken 24h-Multi-Outout-Forecast

Zwar erreicht derMulti-Output-24h-Forecast gute globale Metriken über alle Prognosezeitpunkte hinweg, jedoch zeigt sich im blockweisen Tagesvergleich ein deutlich

grösserer Performance-Abfall zwischen Trainings- und Testdaten (leichtes Overfitting). Insbesondere im 24h-Block-Setup ist der mittlere Fehler signifikant höher als beim rekursiven Ansatz, was auf eine geringere Stabilität bei konkreten Tagesprognosen hindeutet. Auf Basis dieser Ergebnisse wurde der rekursive Ansatz als finales Modell für den Day-Ahead-Forecast gewählt.

Zusammenfassend zeigt der Vergleich, dass der rekursive Multi-Step-Forecast im 24h-Block-Setup die beste Prognosequalität für vollständige Tageslastgänge liefert. Auf Basis dieser Ergebnisse wurde der rekursive Ansatz als finales Modell für den Day-Ahead-Forecast ausgewählt.

3.6.3.1 Tagesverbrauch an einem beliebigen Tag

Im Folgenden sind exemplarische Tagesprognosen für den 15.11.2024 dargestellt, um die Prognosequalität der untersuchten Modellansätze visuell zu vergleichen:

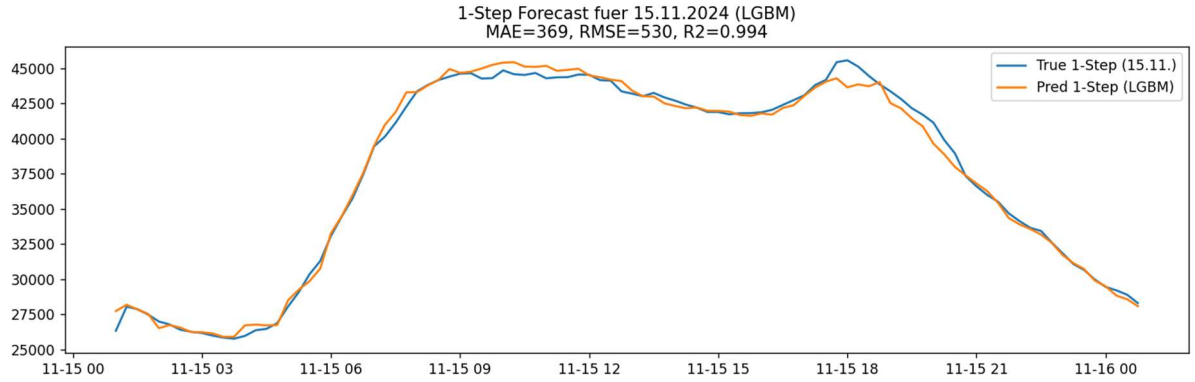


Abbildung 10 1-Step-Forecast 15.11.2024

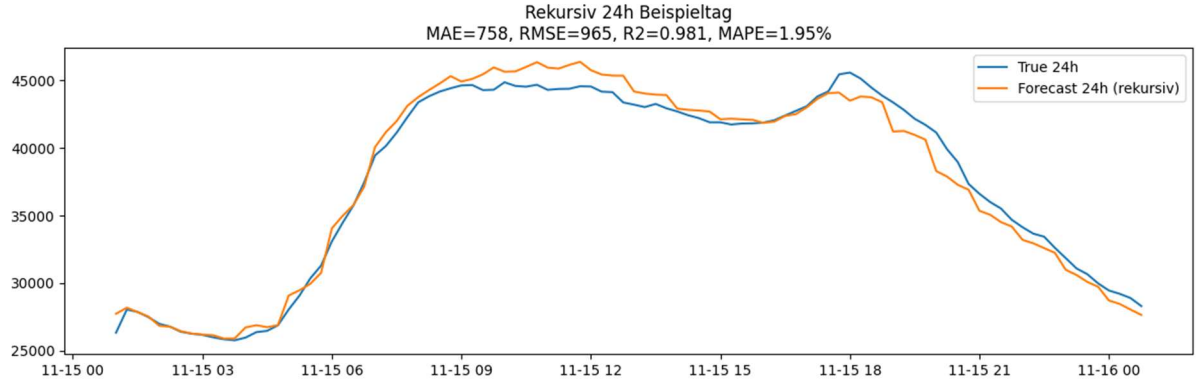


Abbildung 11 recursive 24-Step-Forecast 15.11.2024

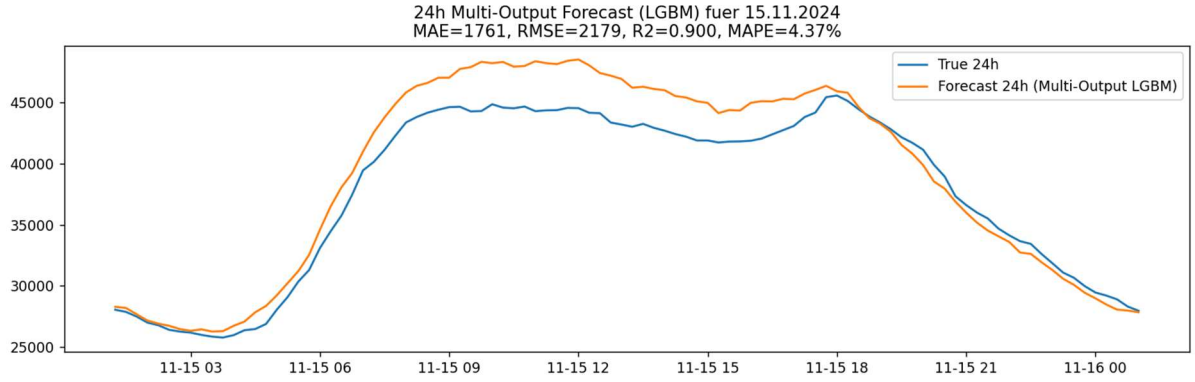


Abbildung 12 Multi-Output-Forecast 15.11.2024

Die exemplarischen Tagesprognosen für den 15.11.2024 zeigen, dass alle Modelle die grundlegende Tagesstruktur korrekt erfassen. Der rekursive Ansatz bildet jedoch den Verlauf über den gesamten Tag hinweg konsistenter ab als der Multi-Output-Forecast. Während Nacht- und Frühstunden sehr genau prognostiziert werden, treten während der Tagesstunden grössere Abweichungen auf. Diese sind plausibel und lassen sich durch nicht modellierte Effekte wie kurzfristige Verbrauchsspitzen, industrielle Lastwechsel oder besondere Ereignisse erklären.

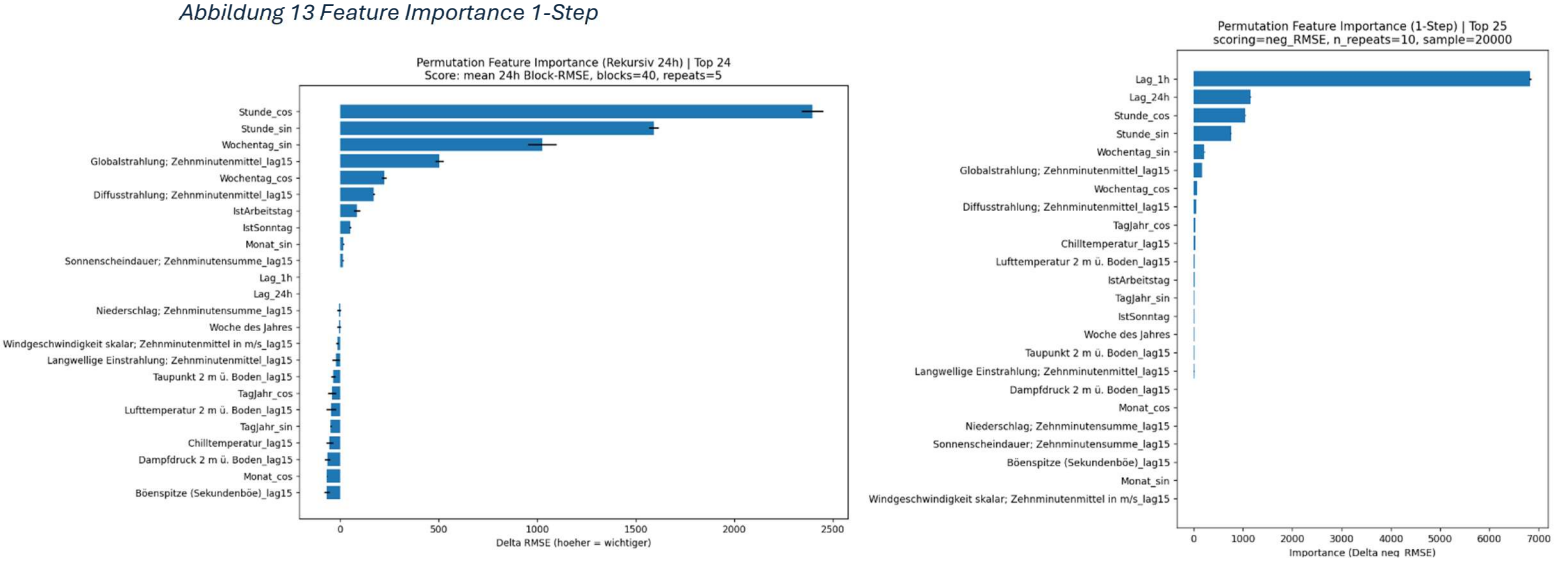
Zusätzliche Tests an Sondertagen (z. B. 31.12.) zeigten eine erhöhte Prognoseabweichung. Dies unterstreicht die Bedeutung expliziter Feiertags- und Ferieninformationen, die aus zeitlichen Gründen nicht integriert wurden, jedoch ein klares Verbesserungspotenzial für zukünftige Arbeiten darstellen.

3.6.4 Analyse der Feature Wirkung

Die Analyse der Feature-Relevanz mittels Permutation Importance zeigt deutliche Unterschiede zwischen den Forecast-Setups und bestätigt die jeweilige Modelllogik.

Im One-Step-Forecast dominieren erwartungsgemäss kurzfristige Lag-Features, insbesondere der 1h- und 24h-Lag. Diese liefern dem Modell eine sehr direkte Information über den aktuellen Systemzustand und erklären den Grossteil der kurzfristigen Prognoseleistung. Kalender- und Wetterfeatures spielen hier lediglich eine untergeordnete Rolle, da ihr Einfluss im sehr kurzen Prognosehorizont gering ist.

Abbildung 13 Feature Importance 1-Step



Im rekursiven 24h-Forecast zeigen die Ergebnisse der Feature-Importance klar, dass zeitliche Strukturmerkmale den grössten Beitrag zur Prognose leisten. Insbesondere die zyklischen Stundenfeatures (Sinus/Kosinus der Stunde) sowie der Wochentag sind entscheidend für die Modellleistung. Diese Features geben dem Modell den typischen Tages- und Wochenrhythmus vor und stabilisieren den rekursiven Vorhersageprozess über den gesamten Prognosehorizont hinweg.

Wetterfeatures wie Global- und Diffusstrahlung sind ebenfalls relevant, ihr Einfluss ist jedoch geringer als jener der zeitlichen Merkmale. Im rekursiven Setup wirken diese Informationen unterstützend, da der Prognoseverlauf bereits stark durch wiederkehrende zeitliche Muster und die rekursive Nutzung vergangener Verbrauchswerte geprägt ist. Einzelne Wetter- und Kalenderfeatures zeigen sogar eine geringe oder negative Permutationsimportance, was darauf

hindeutet, dass sie im rekursiven Kontext redundant sind oder den Vorhersageprozess leicht destabilisieren können.

Insgesamt bestätigen diese Ergebnisse, dass der rekursive Ansatz vor allem von klaren zeitlichen Strukturen lebt. Dies erklärt die hohe Stabilität des Modells im 24h-Block-Setup und begründet die Wahl des rekursiven Forecasts als finales Modell für den Day-Ahead-Einsatz.

4 Fazit und Schlussfolgerung

4.1 Reflexion zum Arbeitsaufwand und Lernprozess

Ein wesentlicher Teil unseres Projekts bestand nicht nur darin, funktionierenden Code zu schreiben, sondern die Daten und jeden einzelnen Verarbeitungsschritt wirklich zu verstehen. Der grösste Aufwand, deutlich mehr als das eigentliche Modellieren, lag im gemeinsamen, schrittweisen Durchgehen des gesamten Pipelineskripts. Jede Zeile wurde hinterfragt: Welche Annahmen stecken dahinter? Warum funktioniert eine Methode genauso? Wo entstehen versteckte Fehlerquellen wie Data Leakage oder fehlerhafte Zeitkonvertierungen?

Gerade die Kombination aus lokalen Zeitstempeln, UTC-Konvertierung, Interpolation und Lag-Features führte zu vielen Stellen, an denen scheinbar kleine Details grosse Auswirkungen haben. Ein prägnantes Beispiel war der Moment, in dem wir erkannten, dass eine Berechnung wie Stromverbrauch(t) – Stromverbrauch(t−1) das Modell mit Zukunftsinformationen füttert – ein klassischer Data-Leakage-Fehler, der erst auffällt, wenn man die Logik wirklich durchdringt.

Diese intensive Auseinandersetzung hat uns gezeigt, dass Datenaufbereitung mehr ist als "Code ausführen": Es geht darum, implizite Fehler zu erkennen, Muster zu verstehen und die Daten so aufzubereiten, dass ein Modell überhaupt eine realistische Chance hat, sinnvoll zu lernen. Besonders herausfordernd waren dabei:

- **Zeitreihenspezifische Komplexität:** Die Handhabung von Zeitumstellungen (Sommer-/Winterzeit), fehlenden Messwerten und der korrekten zeitlichen Ausrichtung verschiedener Datenquellen erforderte höchste Sorgfalt.
- **Feature Engineering ohne Data Leakage:** Die Entwicklung von Lag-Features und zeitverzögerten Wetterdaten unter strikter Beachtung der zeitlichen Kausalität stellte eine kontinuierliche Herausforderung dar.
- **Modellvergleich und -bewertung:** Die Entwicklung mehrerer Forecast-Setups (One-Step, Multi-Output, rekursiv) und deren faire Evaluation erforderte ein tiefes Verständnis der jeweiligen Stärken und Schwächen.

In diesem Prozess haben wir nicht nur technische Lösungen entwickelt, sondern vor allem gelernt, skeptisch, strukturiert und logisch zu denken. Rückblickend war dies der mit Abstand wertvollste Teil der gesamten Arbeit.

4.2 Erreichte Projektziele

Das Projekt verfolgte das übergeordnete Ziel, ein zuverlässiges und reproduzierbares Machine Learning System zur Prognose des Stromverbrauchs zu entwickeln. Dieses Ziel wurde erfolgreich erreicht:

Technische Zielerreichung

Im Rahmen des Projekts wurde ein robustes Feature-Set aus 39 Features erstellt, bestehend aus Kalender-, Lag- und Wetterfeatures, die zeitlich korrekt aufbereitet und frei von Data Leakage sind. Mehrere Regressionsmodelle wurden trainiert und evaluiert, wobei LichtGBM in einem rekursiven 24h-Setup die beste Performance zeigte und einen MAE von circa 858 sowie ein R^2 von 0.86 auf den Testdaten erreichte. Die Modellgüte wurde dabei anhand klar definierter Metriken wie MAE, RMSE und R^2 bewertet, sowohl global über den gesamten Testzeitraum als

auch in einem business-relevanten 24h-Block-Setup, das den operativen Einsatz als Day-Ahead-Prognose widerspiegelt

Methodische Erkenntnisse

Der durchgeführte Vergleich zwischen One-Step-, Multi-Output- und rekursivem Forecast zeigte deutlich, dass der rekursive Ansatz für Day-Ahead-Prognosen am stabilsten ist. Die Permutation-Importance-Analyse bestätigte dabei, dass zyklische Zeitfeatures wie Stunde und Wochentag sowie kurze lags die wichtigsten Prädiktoren darstellen, während Wetterfeatures primär unterstützend wirken. Die konsequente Anwendung des CRISP-DM-Zyklus erwies sich als strukturgebend und hilfreich, insbesondere durch die iterative Natur des Prozesses, die es ermöglichte, kontinuierlich Verbesserungen vorzunehmen und die Modellqualität schrittweise zu optimieren.

4.2 Modellnutzung nach Zeithorizont

Basierend auf den Evaluationsergebnissen lassen sich klare Empfehlungen für den operativen Einsatz ableiten. Der ***One-Step-Forecast*** eignet sich für sehr kurzfristige Entscheidungen und Nachregelungen im 15-Minuten-Bereich. Mit der höchsten Genauigkeit von etwa 298 MAE auf den Testdaten reagiert dieses Modell unmittelbar auf aktuelle Verbrauchsänderungen, ist jedoch stark von Lag-Features abhängig. Typische Anwendungsfälle sind die Echtzeit-Laststeuerung, kurzfristige Ausgleichsmassnahmen im Netz sowie die Überwachung von Abweichungen vom geplanten Tagesverlauf.

Als Hauptmodell empfiehlt sich der ***rekursive 24h-Forecast*** für die Day-Ahead-Planung und operative Tagesvorbereitung. Er bietet die beste Stabilität für vollständige Tageslastgänge mit einem MAE von circa 858 und einem R^2 von 0.86 und liefert konsistente Prognosen über 24 Stunden hinweg durch eine ausgewogene Nutzung von Zeit-, Lag- und Wetterfeatures. Dieser Ansatz wird insbesondere für die tägliche Lastplanung und Ressourcendisposition, die marktbasierte Strombeschaffung am Day-Ahead-Markt, die Kapazitätsplanung für Eigenproduktion sowie die Vorbereitung auf erwartete Lastspitzen eingesetzt.

Der ***Multi-Output 24h-Forecast*** hingegen eignet sich für globale und aggregierte Einschätzungen. Er zeigt zwar gute globale Metriken, weist jedoch höhere Abweichungen im blockweisen Vergleich auf, da alle 96 Werte simultan prognostiziert werden, was zu einer geringeren Stabilität im Vergleich zum rekursiven Ansatz führt. Dieser Forecast-Typ findet Anwendung in der aggregierten Wochenplanung, bei strategischen Übersichtsanalysen und in Szenarien mit weniger stringenten Genauigkeitsanforderungen.

4.3 Gesamtfazit

Das Projekt hat erfolgreich ein funktionsfähiges System zur Stromverbrauchsprognose entwickelt, das sowohl kurzfristige als auch mittelfristige Vorhersagen ermöglicht. Der rekursive 24h-Forecast bietet eine solide Grundlage für operative Planungsentscheidungen und erreicht eine Prognosequalität, die für den praktischen Einsatz geeignet ist.

Über die technischen Ergebnisse hinaus hat das Projekt wertvolle Einblicke in die Komplexität realer Machine-Learning-Anwendungen gegeben. Die Erkenntnis, dass 80% des Aufwands in der Datenaufbereitung und -validierung liegt und nur 20% in der eigentlichen Modellierung, entspricht den Erfahrungen der Industrie und unterstreicht die Bedeutung einer sorgfältigen Datenvorbereitung.

Die entwickelte Lösung ist nicht nur ein Prognosemodell, sondern ein vollständiger, reproduzierbarer Workflow, der die Grundlage für weitere Verbesserungen und Anwendungen bildet. Damit wurden sowohl die fachlichen Projektziele als auch die persönlichen Lernziele erreicht.

5 Empfehlungen & Zukunftsaussichten (Kerem)

5.1 Kurzfristige Optimierungen

Feiertage/Ferienzeiten Features

Die wichtigste kurzfristige Verbesserung besteht in der expliziten Modellierung von Feiertagen, Ferienzeiten und lokalen Grossveranstaltungen. Die Tests an Sondertagen (z.B. Silvester) zeigten deutlich erhöhte Prognosefehler, was auf die Relevanz dieser Informationen hinweist.

Konkrete Umsetzung:

- Integration einer Feature-Spalte für Schweizer Feiertage und Schulferien
- Kennzeichnung von Basel-spezifischen Events (Art Basel, Herbstmesse, Fasnacht)
- Kategorisierung in verschiedene Event-Typen (Feiertag, Ferientag, Grossevent)
- Optionale Gewichtung nach erwarteter Auswirkung auf den Stromverbrauch

Erwarteter Mehrwert: Reduktion des MAE um 5-10% an Sondertagen, stabilere Jahresprognosen.

Hyperparameter-Tuning

Das aktuelle Modell verwendet Standard- oder manuell justierte Parameter. Ein systematisches Hyperparameter-Tuning könnte die Performance weiter verbessern.

Konkrete Umsetzung:

- Grid Search oder Bayesian Optimization für LightGBM-Parameter
- Fokus auf num_leaves, learning_rate, max_depth, min_child_samples
- Cross-Validation mittels TimeSeriesSplit zur robusten Parameterauswahl
- Separate Optimierung für One-Step- und rekursives Setup

Erwarteter Mehrwert: Verbesserung des MAE um 2-5%, höhere Stabilität bei Extremwerten.

5.2 Mittelfristige Erweiterungen

Integration von Wetterprognosen

Das aktuelle Modell nutzt ausschliesslich historische Wetterdaten. Für eine echte Vorhersage sollten Wetterprognosen integriert werden.

Konkrete Umsetzung:

- Anbindung an Wetter-APIs (z.B. MeteoSwiss API, OpenWeatherMap)
- Abruf von 24h-Wettervorhersagen (Temperatur, Niederschlag, Sonneneinstrahlung)
- Integration als zusätzliche Feature-Spalten mit Prognosehorizont-Kennzeichnung
- Hybridansatz: Historische Werte für Training, Prognosen für Prediction

Erwarteter Mehrwert: Bessere Vorhersage wetterabhängiger Verbrauchsmuster (Heizung, Kühlung), insbesondere bei extremen Wetterereignissen.

5.3 Deployment und Nutzung

Ein weiterer wichtiger Zukunftsaspekt ist das Deployment des Modells in einer nutzerorientierten Anwendung. Denkbar wäre eine webbasierte oder mobile Applikation, über die Endkunden ihren zukünftigen Stromverbrauch prognostizieren können.

Solch ein System könnte Haushalten oder Unternehmen dabei helfen, geplante Stromkäufe besser einzuschätzen, Verbrauchsspitzen zu vermeiden und den eigenen Energiebedarf bewusster zu steuern. In Kombination mit aktuellen Strompreisen oder Tarifmodellen könnten Prognosen zudem als Entscheidungsgrundlage für zeitlich optimierte Strombezüge dienen.

Damit würde das Modell nicht nur auf Netzbetreiber-Ebene, sondern auch auf Kundenseite einen direkten Mehrwert schaffen.

5.4 Regionale Flexibilität und Skalierung

Ein naheliegender nächster Schritt besteht darin, das entwickelte Modell auf weitere Regionen oder Städte anzuwenden. Durch den Einsatz vergleichbarer Verbrauchs- und Wetterdaten könnte untersucht werden, inwiefern die gelernten Zusammenhänge über Basel-Stadt hinaus generalisierbar sind.

Ein besonderer Mehrwert ergibt sich aus der Anwendung solcher Prognosemodelle in Schwellen- und Entwicklungsländern, in denen Stromnetze häufig begrenzte Kapazitäten aufweisen und Angebot sowie Nachfrage starken Schwankungen unterliegen.

In diesem Kontext wäre es sinnvoll, das Modell bewusst zu vereinfachen, beispielsweise durch eine Reduktion der Feature-Anzahl oder den Einsatz weniger komplexer Algorithmen. Solche vereinfachten Modelle sind robuster, leichter wartbar und benötigen weniger Rechenressourcen, wodurch sie auch in Umgebungen mit eingeschränkter Datenverfügbarkeit oder technischer Infrastruktur praktikabel einsetzbar sind.

Dadurch können Lastspitzen frühzeitig erkannt, Energie effizienter verteilt und die Versorgungssicherheit nachhaltig verbessert werden.

5.5 Schlussbemerkung zu Zukunftsperspektiven

Die vorgestellten Empfehlungen reichen von kurzfristig umsetzbaren Verbesserungen (Feiertags-Features, Hyperparameter-Tuning) über mittelfristige Erweiterungen (Wetterprognose-Integration, Anomalie-Erkennung) bis hin zu langfristigen Visionen (vollautomatisierte Pipeline, Nutzer-Applikation, internationale Skalierung).

Hohe Priorität (Quick Wins):

- Feiertags- und Ferienfeatures
- Hyperparameter-Tuning
- Rolling Features

Mittlere Priorität (Qualitätsverbesserung):

- Wetterprognose-Integration
- Anomalie-Erkennung
- Prediction Intervals
-

Niedrigere Priorität (Erweiterte Features):

- Unsupervised Learning / Clustering
- Deep Learning Modelle
- Ensemble-Methoden

Die entwickelte Lösung bietet eine solide Grundlage für all diese Erweiterungen. Der modulare Aufbau des Codes und die klare Trennung von Datenverarbeitung, Feature-Engineering und Modellierung erleichtern zukünftige Anpassungen. Damit ist das Projekt nicht nur ein funktionsfähiges Prognose-System, sondern auch eine Plattform für kontinuierliche Verbesserung und Innovation im Bereich der Stromverbrauchsprognose.