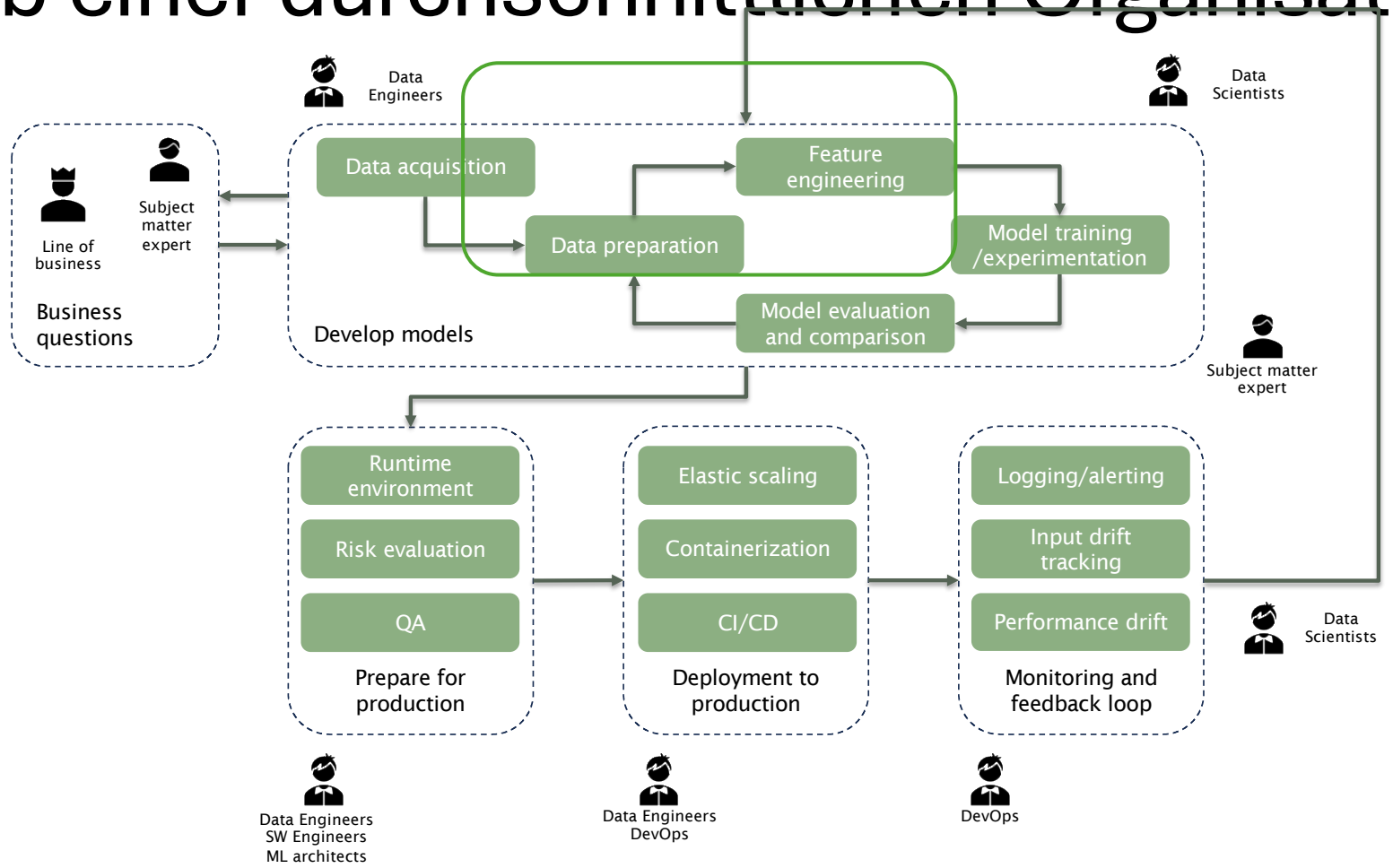
Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

Unüberwachtes Lernen: Feature Engineering Theorie

Violeta Vogel

Technik und Informatik, BFH

Das realistische Bild eines ML-Lebenszyklus innerhalb einer durchschnittlichen Organisation



Feature Engineering

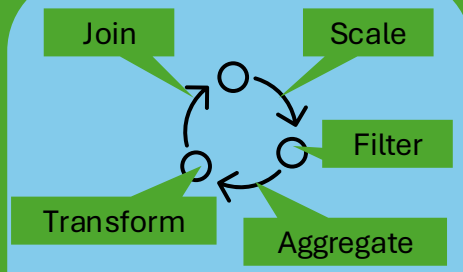
- Feature: Eingaben, mit denen ML-Modelle während Trainings Vorhersagen treffen
- Feature Engineering ist der Prozess, Rohdaten durch Transformation, Auswahl und Neuerstellung in aussagekräftige Eingabevariablen (Features) für maschinelles Lernen umzuwandeln
- Ziele:
  - Verbesserung der Modelleistung
  - Reduzierung der Komplexität
  - Erhöhung der Vorhersagekraft

Datenvorbereitung: Feature Engineering Flow

Rohe Daten

Daten in den Quellsystemen: Strukturiert, unstrukturiert

Feature Engineering



Beispiele: Kundenalter

Features

Customer	Age group	Max purchase amount
1	7	12
2	8	33
Product ID	Categorie	Total purchased
3001	5	1254
3001	6	908

Trainieren / Entwickeln

Produktion

Governance

Feature Engineering

- Techniken:
  - Imputation: Auffülen der fehlender Werte
  - Encoding: Umwandlung kategorialer Daten in Zahlen
  - Skalisierung / Normalisierung: Daten auf einen ähnlichen Wertebereich bringen
  - Binning (Klassierung): Kontinuierliche Variablen in Kategorien bringen
  - Feature Creation: Erstellen neuer Variablen
  - Umgang mit Ausreissern
- Prozess: Findet nach Datenbereinigung vor dem Modelltraining statt, erfordert Fachwissen

Imputation

- Imputation ist ein statistisches Verfahren,
  - bei dem fehlende Datenpunkte in einem Datensatz durch plausible Schätzwerte ersetzt werden.
  - Anstatt unvollständige Datensätze (Beobachtungen) einfach zu löschen, wird versucht, die Lücken so zu füllen, dass der gesamte Datensatz für Analysen oder Machine-Learning-Modelle nutzbar bleibt

Imputation

- Imputation wird überall dort eingesetzt, wo unvollständige Daten die Analysequalität beeinträchtigen könnten:
  - Klinische Forschung & Medizin: Ergänzung fehlender Patientenwerte in Studien, um alle Probanden in die Auswertung einbeziehen zu können.
  - Machine Learning: Vorbereitung von Trainingsdaten, da viele Algorithmen keine fehlenden Werte verarbeiten können.
  - Marktforschung & Umfragen: Ausfüllen von Lücken, wenn Teilnehmer bestimmte Fragen nicht beantwortet haben.
  - Zeitreihenanalyse: Schätzung fehlender Sensordaten oder Wirtschaftsindikatoren über einen Zeitverlauf

Imputation

- Gängige Methoden:
  - Mittelwert-/Median-Imputation: Ersetzt Lücken durch den Durchschnitt aller vorhandenen Werte.
  - K-Nearest-Neighbors (KNN): Sucht nach den "ähnlichsten" Datenpunkten und nutzt deren Werte als Ersatz.
  - Multiple Imputation (MI): Erstellt mehrere verschiedene Schätzungen für denselben fehlenden Wert, um die Unsicherheit statistisch abzubilden.

Imputation

Vorteile	Nachteile
Datenerhalt: Verhindert den Verlust wertvoller Informationen, die durch das Löschen ganzer Zeilen entstehen würden	Verzerrungsgefahr: Falsche Imputationsmethoden können statistische Zusammenhänge verfälschen oder künstliches Rauschen erzeugen
Statistische Power: Grössere Stichprobenumfänge führen zu robusteren Ergebnissen und präziseren Modellen	Unterschätzung der Varianz: Einfache Methoden (wie Mittelwertimputation) verringern die natürliche Streuung der Daten.
Vollständigkeit: Ermöglicht den Einsatz von Standard-Algorithmen, die zwingend vollständige Matrizen erfordern.	Rechenaufwand: Komplexe Verfahren wie die Multiple Imputation oder KNN-Imputation benötigen deutlich mehr Zeit und Fachwissen

Encoding

- Encoding ist der Prozess, bei dem kategorische Daten (Texte, Labels, Klassen) in ein numerisches Format umgewandelt werden.
- Da die meisten Machine-Learning-Algorithmen auf mathematischen Gleichungen basieren, können sie Text nicht direkt verarbeiten – Encoding schlägt hier die Brücke

Encoding

- Encoding Einsatzbereiche
  - Klassifizierung: Umwandlung von Zielvariablen (z. B. "Spam" vs. "Kein Spam").
  - Feature Engineering: Aufbereitung von Merkmalen wie "Farbe", "Stadt" oder "Berufsgruppe" für Regressions- oder Clustering-Modelle.
  - NLP (Natural Language Processing): Vorstufe zur Vektorisierung von Wörtern in der Textverarbeitung.

Encoding

Methode	Funktionsweise	Beste Anwendung
Label Encoding	Weist jeder Kategorie eine Zahl zu (0, 1, 2...).	Ordinale Daten (Daten mit Rangfolge, z. B. "Klein", "Mittel", "Gross").
One-Hot Encoding	Erstellt für jede Kategorie eine eigene Spalte (0 oder 1).	Nominale Daten ohne Rangfolge (z. B. "Länder", "Farben")

Encoding

- Encoding Vorteile:
  - Algorithmus-Kompatibilität: Ermöglicht den Einsatz von mathematischen Modellen auf nicht-numerischen Daten
  - Informationserhalt: Strukturierte Kategorien bleiben für das Modell interpretierbar.

- Encoding Nachteile:
  - Fluch der Dimensionalität: One-Hot Encoding kann bei vielen Kategorien (z. B. Postleitzahlen) den Datensatz massiv aufblähen und das Modell verlangsamen.
  - Falsche Hierarchien: Label Encoding suggeriert bei nominalen Daten eine mathematische Rangfolge (z. B. ist "Bern=2" doppelt so viel wert wie "Zürich=1"), was das Modell verwirren kann

Skalierung / Normalisierung

- Skalierung und Normalisierung sind Techniken, um numerische Merkmale in einen vergleichbaren Wertebereich zu bringen.
- Da Machine-Learning-Algorithmen oft Abstände berechnen, würden Merkmale mit grossen Zahlenwerten (z. B. „Einkommen“ in Tausendern) Merkmale mit kleinen Werten (z. B. „Alter“ in zweistelligen Zahlen) fälschlicherweise dominieren

Skalierung / Normalisierung

- Zentrale Methoden
  - Min-Max-Skalierung (Normalisierung): Transformiert Werte in einen festen Bereich, meist 0 bis 1. (Ideal, wenn die Verteilung nicht normal ist).
  - Standardisierung (Z-Transformation): Zentriert die Daten um den Mittelwert 0 mit einer Standardabweichung von 1. (Ideal für Algorithmen wie SVM oder PCA).

Skalierung / Normalisierung

- Einsatzbereiche:
  - Distanzbasierte Algorithmen: K-Nearest-Neighbors (KNN) oder K-Means Clustering, da diese extrem empfindlich auf unterschiedliche Skalen reagieren.
  - Gradientenabstiegs-Verfahren: Neuronale Netze und Logistische Regression konvergieren (lernen) deutlich schneller, wenn die Daten skaliert sind.
  - Hauptkomponentenanalyse (PCA): Um sicherzustellen, dass Merkmale mit grosser Varianz nicht allein die Richtung der Hauptkomponenten bestimmen.

Skalierung / Normalisierung

Vorteile	Nachteile
Schnellere Konvergenz (stabiler Zustand des Modells): Optimierungsalgorithmen finden das Minimum der Kostenfunktion wesentlich effizienter	Verlust der Originalskala: Die direkte Interpretierbarkeit der Werte geht verloren (z. B. "0.75" statt "50.000 Euro")
Gleichberechtigung der Features: Verhindert, dass Merkmale mit grossen Absolutwerten das Modell „kidnappen“	Empfindlichkeit gegenüber Ausreissern: Besonders bei der Min-Max-Skalierung können extreme Ausreisser die restlichen Daten in einen winzigen Bereich quetschen.
Stabilität: Verbessert die numerische Stabilität vieler mathematischer Modelle	Zusatzschritt: Erfordert eine strikte Trennung von Training und Test (Skalierungsparameter dürfen nur vom Trainingsset stammen).

Feature creation

- Feature Creation (oder Feature Construction) ist die hohe Schule des Feature Engineering:
  - Hier werden aus bestehenden Rohdaten durch mathematische Verknüpfungen, Logik oder Domänenwissen völlig neue Variablen erschaffen.
  - Anstatt nur vorhandene Daten zu putzen, fügt man dem Datensatz neues „Wissen“ hinzu, das dem Algorithmus hilft, komplexe Muster leichter zu erkennen.

Fläche = Breite * Länge

Feature creation

- Einsatzbereiche:
  - E-Commerce: Kombination von „Letztes Kaufdatum“ und „Heute“ zu einem neuen Feature „Tage seit dem letzten Kauf“ (Recency).
  - Finanzwesen: Berechnung des „Schulden-Einkommens-Verhältnis“ statt nur Einzelwerte für Schulden und Einkommen zu nutzen.
  - Geodaten: Umwandlung von Längen- und Breitengraden in die „Distanz zum Stadtzentrum“.
  - Zeitreihen: Extraktion von „Wochentag“ oder „Ist Feiertag?“ aus einem Zeitstempel, um saisonale Effekte zu erfassen.

Feature creation

Vorteile	Nachteile
Leistungssteigerung: Oft der grösste Hebel, um die Genauigkeit (Accuracy) eines Modells massiv zu verbessern	**Overfitting-Gefahr**: Erstellt man zu viele spezifische Features, lernt das Modell eventuell nur das „Rauschen“ der Trainingsdaten auswendig
Vereinfachung: Komplexe Zusammenhänge werden für das Modell „sichtbar“, die es allein durch Rohdaten nicht finden würde.	**Rechenaufwand & Komplexität**: Die Pipeline wird komplizierter; die Datenaufbereitung dauert länger und verbraucht mehr Speicher
Interpretierbarkeit: Ein Feature wie „BMI“ (Body Mass Index) ist für Menschen oft aussagekräftiger als Gewicht und Grösse isoliert.	**Daten-Leaking**: Wenn Informationen aus der Zukunft (Target Leakage) in ein Feature einfliessen, wird das Modell im Test wertlos.

Ausreiser

- Ein Ausreisser (Outlier) ist ein Datenpunkt, der sich signifikant von den restlichen Werten eines Datensatzes unterscheidet – er liegt also weit ausserhalb der erwarteten Verteilung

- Wichtig:
  - Ein Ausreisser kann entweder ein Messfehler (kaputter Sensor) oder eine echte Anomalie (Lottogewinn) sein.
  - Die Entscheidung, wie man damit umgeht, erfordert tiefes Verständnis der Daten

Ausreiser

- Einsatzbereiche:
  - In der Datenanalyse ist der Umgang mit Ausreissern ein zweischneidiges Schwert:
  - Betrugserkennung (Fraud Detection):
    - Hier sind Ausreisser das Ziel.
    - Eine ungewöhnlich hohe Kreditkartenzahlung im Ausland ist ein Ausreisser, der auf Diebstahl hindeutet.
  - Qualitätskontrolle: In der Industrie signalisieren abweichende Sensorwerte (z. B. Hitze-Spitzen) defekte Maschinen.
  - Datenbereinigung: Vor dem Training von Modellen werden Ausreisser oft entfernt oder angepasst, um die allgemeine Vorhersagegenauigkeit nicht zu verzerren.

Ausreiser

Vorgehen	Vorteile	Nachteile
Beibehalten	Verhindert Informationsverlust; wichtig, wenn Ausreisser reale, wertvolle Ereignisse sind (z. B. Börsencrashs)	Kann statistische Kennzahlen wie den Mittelwert massiv verfälschen und Modelle in die Irre führen
Entfernen	Erzeugt ein "sauberes" Modell, das für die Mehrheit der Fälle sehr präzise funktioniert	Kann das Modell "blind" für seltene, aber reale Extremereignisse machen; führt zu künstlich reduzierter Varianz
Anpassen (Capping)	Extremwerte werden auf ein Maximum begrenzt (z. B. 99. Perzentil), was den Einfluss dämpft, ohne den Punkt zu löschen.	Verändert die ursprünglichen Daten und kann die wahre Verteilung verzerren

Was ist wichtig bei Feature Engineering?

1. Domänenwissen (Fachverstand)
2. Vermeidung von Data Leackage: Sind die features aus dem Training auch in der Produktion erfügbar?
3. Iteration und Experimentierfreude: FE ist kein linearer Prozess
4. Einfachkeit versus Komplexität: Keep it simple
5. Konsistenz: Die Logik im Training und in der Produktion muss dieselbe sein
6. Skalierbarkeit und Rechenaufwand: Effizienz ist entscheidend

Wichtigsten Prinzipien des Feature Engineering

1. Relevanz: Verlasse dich nicht auf den Algorithmus; füttere ihn mit Wissen, was er nicht selbst aus Rohdaten ablesen kann
2. Generalisierung statt Auswendiglernen: Features sollen allgemeine Muster abbilden, keine Einzelfälle
3. Vermeidung von Redundanz: Weniger ist mehr. Nutze Korrelationmatrizen, um überflüssiges zu eliminieren
4. Robustheit gegenüber Fehlern: Nutze Binning oder Long-Transfomationen, um extreme Schwankungen zu glätten und das Modell stabiler zu machen
5. Trennung von Training und Test: Schau niemals in die Zukunft (= Testdaten), während du Features baust
6. Skalierbarkeit und Reproduzierbarkeit: Automatisieren dein FE in Pipelines, damit der Prozess vom Rohdatum zum fertigen Feature fehlerfrei wiederholbar ist

ML-Data Pipeline: Target Variable

- Die Target-Variable (auch Zielvariable, Label, y) ist die Grösse, die ein Machine-Learning-Modell vorhersagen soll.
- *Ohne Target-Variable gibt es kein überwachtes Lernen.*
- Kernidee
  - Die Target-Variable ist das Ergebnis, das man aus den Eingabedaten ableiten möchte.
- Beispiele:
  - In der Preisvorhersage ist der Wohnungspreis die Target-Variable.
  - In der Spam-Erkennung ist das Label Spam / Nicht-Spam.
  - In der Bildklassifikation ist die Target-Variable z. B. Katze / Hund.

ML-Data Pipeline: Target Variable

- Einordnung im ML-Kontext
- Features → Eingaben (x)
- Target-Variable → vorherzusagende Ausgabe (y)
- Das Modell lernt eine Funktion: F(x) -> y
- Die Target-Variable bestimmt:
  - welche Modellart geeignet ist
  - welche Metriken sinnvoll sind
  - wie man die Daten aufbereitet
  - wie man den Train/Test-Split gestaltet

ML-Data Pipeline: Target Variable Beispiele

- Regression
  - Target ist eine numerische Grösse: Wohnungspreis, Temperatur, Umsatz
- Klassifikation
  - Target ist eine Kategorie: Spam / Nicht-Spam, Krank / Gesund, Produktklasse
- Zeitreihen
  - Target ist ein zukünftiger Wert: Nachfrage am nächsten Tag, Energieverbrauch in der nächsten Stunde

ML-Data Pipeline

- dieses Vorgehen ist unabhängig vom Tool, gilt also auch für andere Machine Learning Umgebungen (R, pySpark, Azure Machine Learning, etc.)
- eine Spezialität von scikit-learn ist aber, dass auch Features und Target als eigenständige Objekte übergeben werden müssen
  - Features als Matrix (X)
  - Target als Vektor (y)
- *dieser "Features - Target - Split" erfolgt gewöhnlich vor dem Train - Test - Split*

Während das Trainieren des Modells oft nur einen Knopfdruck bedeutet, ist Feature Engineering die eigentliche Handarbeit, die über Erfolg oder Misserfolg entscheidet.