Advanced Machine Learning

Zusammenfassung

Stephan Stofer

4. März 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Data Classification

Daten werden in zwei Klassen unterteilt. *Numerische* und *Kategorische* Daten. Bei numerische Daten gibt es *stetige* oder *diskrete* Zahlen. Bei Kategorischen sind entweder *ordinal* oder *nominal*. Ordinale haben eine Hierarchie.

1.1 Data Quality Assessment

Daten sind sehr wichtig, der beste ml-Algo nützt nicht, wenn Daten rubbish sind. Mögliche Fehlerquellen:

- Technische Fehler
- Qualität
- schlecht Design
- menschliche Fehler
- Input in Web-Apps (ungeprüfte Eingabefelder)
- Exporte der Daten, falsche Formate - oder Pre-Processing
- Falschangaben durch Benutzer
- Daten haben immer ein Ablaufdaten! (z.B. emailadressen, Adressen)

Ein DQA kommt immer zuerst! Schützt auch die Reputation gegegnüber Kunden.

1.1.1 Data Cleaning

Prozess um Fehler in Daten zu beheben (automatisch)/bereinigen. Duplikate entfernen, null-values entfernen, Datenformate ml-friendly aufbereiten (data wrangling). Die Änderungen müssen dokumentiert und versioniert werden, den data provider darüber informieren und die Ursache für die data quality issues untersuchen.

1.1.2 Approaches to DQA

Dies ist detektiv-Arbeit. Wenn etwas verdächtig erscheint, weitergraben! Die Daten werden überprüft, ob sie vertrauenwürdig sind (plausibilieren).

- Datenquellen und vertrauenwürdigkeit prüfen
- statistische Kennzahlen interpretieren
- daten visualisieren
- Datenranges prüfen (Alter sollte unter 200 sein, Salär > 0, usw.)
- Korrelation zwischen Attributen prüfen (Tachostand und Preis eines Autos)
- Redundanz -> je weniger umso bessere Daten
- Anomalieprüfung in Syntax und Semantik
- NULL Werte und Duplikate erforschen

1.1.3 Statistische Kennzahlen

Geben uns einen Fingerabruck und erste Plausibilisierung der Daten. Die wichtigsten Kennzahlen sind:

- Mittelwert - *mean* - O(n)
- Modus - *mode* die Zahl die am meisten vorkommt
- Median - *median* - O(n * log n), ist aussagekräftiger

1.1.3.1 Schiefheit

Der Mean, Modus und Median geben Auskunft über die Schiefheit der Daten. Wir haben eine negative, Links-Schiefe *skewness* wenn $mean - mode < 0$, wenn positiv, Rechts-Schiefe $mean - mode > 0$

1.1.3.2 Median

Sortiere Datenreihe. Der Median enthält 50% der Daten. Die Quantile entsprechen je 25%. Die Interquartils Differenz (IQR) entspricht Q3 - Q1.

1.1.3.3 Boxplots

Sehr nützlich zur grafischen Darstellung. *Outliers* sind die Werte die grösser sind als $Q3+1.5*IQR$ respektive $Q1-1.5*IQR$. Minimum bzw. Maximum sind die Werte, die gerade noch ind diese Grenze $1.5*IQR$ reinpassen.

Wenn viele Outliers müssen Daten genau angeschaut werden, ob sie trotzdem plausibel sind.

1.1.3.4 Five Number Summary of a Data Distribution

In mit Python kann sehr einfach die $Q1, Q2, Q3$, min und max einer Datenreihe ausgegeben werden:

```
import numpy as np
import pandas as pd

s = pd.Series(np.random.rand(100))
s.describe()
```

Auch Boxplots sind sehr einfach:

```
import matplotlib.pyplot as plt
plt.boxplot(x = [data.Mileage, data.Price], labels=['Mileage', 'Price'])
```

1.1.3.5 Datenverteilung

Die Verteilung wird mit der Varianz betrachtet, wobei diese *sample variance* die Besselkorrektur $(n-1)$ nutzt. Die Standardabweichung entspricht aus der $\sqrt{Var(x)}$

1.1.3.6 Covarianz

Die Covarianz zeigt die Variablität von zwei Datensätzen auf. Ist der Wert positiv, verhalten sich die beiden Daten ähnlich. Ist sie negativ, entsprechend nicht. Ist aber schwierig zu interpretieren, weil sie nicht normiert ist.

1.1.3.7 Covarianzmatrix

Die covarianzmatrix ist sehr wichtig in ML. Sie enthält alle Covarianzen aller Varianzpaare. Die Diagonale kann durch die Varianz von X ersetzt werden.

1.1.3.8 Pearson Korrelation

Covarianz wird durch die Standardabweichung dividiert. Deshalb ergeben sich Werte zwischen 1 (perfekte Korrelation) und -1 (perfect anti-correlation). Damit kann die Datenreihe verglichen werden. Die Korrelationsmatrix kann als Heatmap gut dargestellt werden.

1.2 Replacement Strategies für NULL Values

Kommen immer wieder vor. ML-Algos können selten damit umgehen und müssen bereinigt werden. Je nach Datenumfang sind versch. Verfahren denkbar:

- Zeilen mit NULL Werten löschen
- Fehlende Daten manuell einsetzen
- Globale Konstanten einsetzen (UNKNOWN, $infty$)
- Tendenzen verwenden (Mittelwert für symmetrische Daten, Medien für Schiefedaten)
- Tendenzen auch pro "Klasse" (Eigenschaften) berechnen (z.B Krebskranke und gesunde Patienten)
- Regressionsmodell (sehr aufwändig und ungewohnt in Praxis)

1.2.1 Feature Engineering

Features entsprechen Spalten. Null-Values können also mit ML erzeugen. Information verfügbar für ML-Algo machen.

1.2.2 Vector Space Model

Entspricht einem Datenset welches ausser dem Key nur numerische Werte enthält. Kategorische Daten können sehr einfach in nummersiche Daten transformiert werden. Zum Beispiel werden die Farben alle zu Spalten und entsprechende Zugehörigkeit mit 1 bzw. 0 gekennzeichnet. Diese werden als Dummy-Variable bezeichnet.

```
import pandas as pd
data = pd.read_csv('cars.csv')
data = pd.get_dummies(data)
```

Python code um Daten entsprechend aufzubereiten.

1.2.2.1 Dummy Variable Trap

Mit dem einfügen von Dummy-Variablen muss die *Multikolloniarität* im Auge behalten werden. Wenn n-Dummy Variablen erzeugt werden und $n-1$ Spalten alle 0 sind, wissen wir zu 100, dass die $n$te Spalte 1 sein muss. Dies führt zu unterterminierten Matrizen. Die Matrix kann nicht invertiert werden. Um das zu verhindern, muss eine Spalte gelöscht werden! Es gibt aber Verfahren, die immun dagegen sind (z.B. Entscheidungsbäume).

1.3 Pandas Profiling

Effizient in drei Zeilen Code!

```
import pandas_profiling
data = pd.read_csv('cars.csv‘)
data.profile_report()
```

1.4 Fazit

Bei jedem ML-Projekt ist in Data Quality Assessment Pflicht

2 this is a test

I wrote this text.

```
Alice -> Bob: Authentication Request
Bob --> Alice: Authentication Response

Alice -> Bob: Another authentication Request
Alice <-- Bob: another authentication Response
```

3 Machine Learning Fundamentals

Grundlagen des ML

3.1 Vector Space Model

Ein Vektor Space Model enthält nur numerische Daten (ausser dem Key). Zum Überführen von kategorischen Daten gibt es mehrere Techniken für die Tranformation. Von nun an gehen wir jeweils davon aus, dass die Daten numerisch vorhanden sind. Der Key wird normalerweise nicht zu den Daten gezählt.

3.2 Data Records

Beinahe alle ML-Verfahren setzen ein *VSM* voraus. Wenn alle Daten als nummersiche Werte voriegen, kann eine jede Zelle als Spalte interpretiert werden.

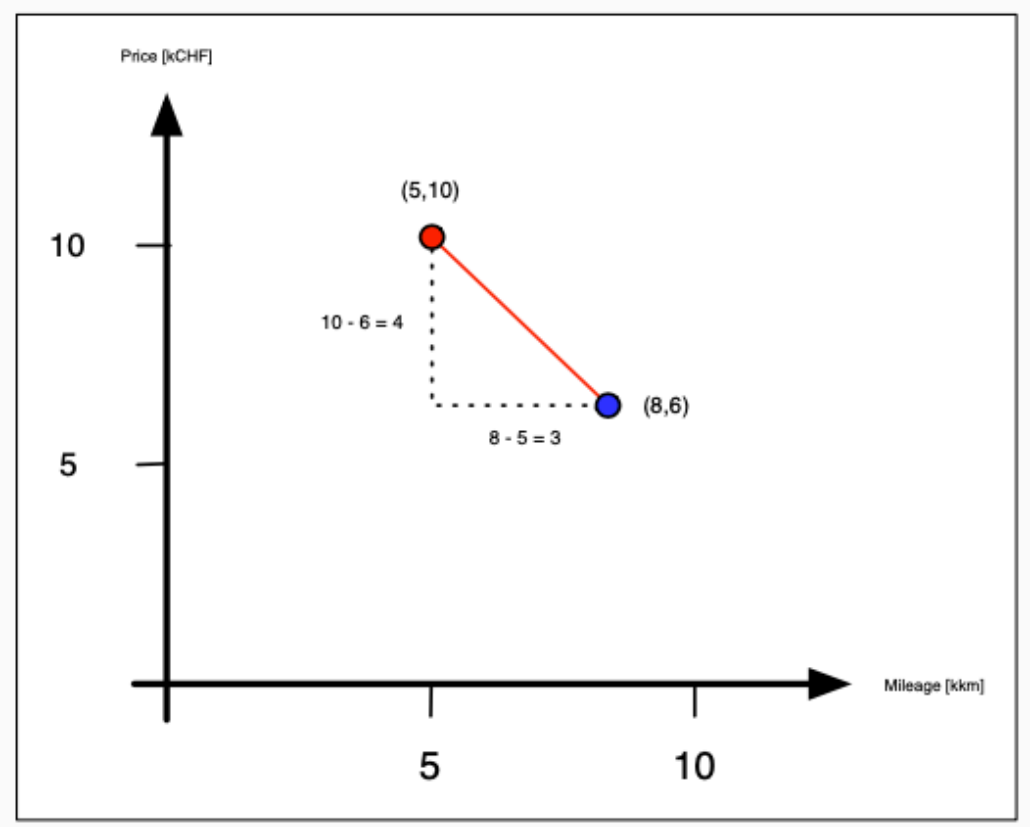

Abbildung 3.1: Geometrische Interpretation von Daten

Die Länge der roten Linie kann mit Hilfe von $\sqrt{a^2 + b^2}$ berechnet werden und als Distanz interpretieret werden.

3.3 Numerical Encoding of Text

Kann man auch ganze Texte oder Bilder in ein Vector Space Model überführen? Bei Bilder können mit Hilfe von Filter transformiert werden. Bei Text funktioniert es mit Hilfe des **TF-IDF Scores** - *term frequency-inverse document frequency*. Diese verbinden Wörter zu numerischen *Vektoren*. Als Basis dient dazu einen Referenz-*Korpus*. Dazu nehmen man alle Wikipediaartikel und stellt diese in Kolonnen dar. Jedes Wort wird nun gemessen anhand der Häufigkeit im ersten Artikel. Danach dividiert durch das Reziproke vom Gesamtwert dieses Wortes über alle Artikel.

3.4 Distance and Similarity

Das Inverse der Distanz ist die Similarität. Je weiter voneinander entfernt, desto unterschiedlicher, respk. je näher umso ähnlicher (Similarität). Auf diesem Konzept basieren alle ML-Algorithmen ausserhalb von Neuronalen Netzen und evtl. Entscheidungsbäume. Beispiele für Distanz und Similarität sind:

- Dating-Site empfiehlt anderes Profil mit den ähnlichsten Vorstellungen
- Auto-Verkaufseite schlägt Preis für Auto anhand der 20 letzten Verkäufe dieses Modell vor
- Webshop empfiehlt Produkte anhand ähnlicher Warenkörbe anderer Benutzer

Wir brauchen also einen Weg die Distanz/Similarität aussagekräftig zu berechnen.

3.5 Euclidiean Distance or L^2-Norm

Ist eine generalisierte Form der Pythagoras Formel welche für eine beliebig grosse Anzahl an Dimensionen verwendet werden kann. Ist eine Zahl $[0, \infty[$ und findet den ähnlichsten Punkt durch die *Minimierung der Distanz* zwischen zwei Punkten. Die Euklidische Distanz wird auch L^2-Norm genannt.

$$euclid(X, Y) = \|X - Y\|_2 = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} = \sqrt{\sum_{i=1}^{2} x_i^2 - 2x_i y_i + y_i^2}$$

3.6 Semantik of Similarity

Je nach Daten ergibt die Distanz nicht exakt wider, wie wir das erwarten würden. Bei unterschiedlichen Anwendungszwecke genügt die Similarität nicht den Ansprüchen. So würde die Distanz bei einem Text der Copy-Pasted wurde weit auseinander liegen (obwohl gleicher Inhalt). Wir müssen also semantisch sinvoll die richtige Methode auswählen. Die Distanzfunktion muss auf die Domäne angepasst werden.

3.7 Cosine Similarity Intuition

Die Kosinus Similarität misst den Winkel θ zwischen zwei Vektoren X und Y. Zwei Punkte auf einer Geraden haben den Winkel 0, aber hätten einen grosse *Euklidische Similarität.*

Die Similarität kann bei vielen Algorithmen per Parameter übergeben werden.

3.8 Cosine Similarity

Durch das Normieren $\frac{X}{\|X\|_2}$ der Vektoren werden die Punkte auf dem Einheitskreis abgebildet. Das Skalarprodukt entspricht der Projektion eines Vektors Y auf den Vektor X. Der Kosinus liegt zwischen $[1, 1]$, für die Distanz verwenden wir aber ein positiven Wert, weshalb wir den Betrag verwenden, wird Kosinus Similarität. 0 bedeutet kleinste Distanz - maximale Similarität und 1 grösste Distanz - Dissimilarität.

$$\text{Kosinusdistanz} = 1 - \text{Kosinus Similarität}$$

3.9 Euklid vs. Kosinus

Die beiden Varianten können ganz unterschiedliche Resultate liefern. Werden die Punkte auf den Einheitskreis normiert, ergeben beide Verfahren dasselbe Resultat.

3.10 Manhattan Distance

Auf einem Schachbrett oder auf der Strasse von Manhattan kann nicht die Euklidische Distanz zur Bestimmung der Entfernung verwendet werden, da man keiner Geraden folgen kann. Die Manhattan Distanz ist definiert durch

$$manhattan(X, Y) = \sum i = 1^n \|x_i - y_i\|$$

Der Wertebereich der Distanz liegt zwischen $[0, \infty[$

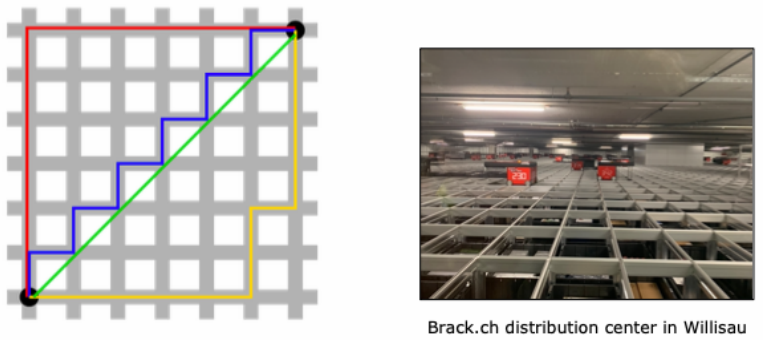

Abbildung 3.2: Manhattan Distanz bei Schachbrett-Muster

3.11 Levenshtein or Edit Distance for Strings

Die Levenshtein Methode zählt die minimale Anzahl an nötigen Operationen an einem Wort um dieses in ein anderes zu überführen. Jede Operation gibt Strafpunkte und werden addiert. Die Summe entspricht der *Levenshtein Distance*. Je grösser umso weniger Similarität zwischen den beiden Worten. Die Operationen werden wie folgt gewertet:

- +1, wenn Buchstabe gelöscht wird
- +1, wenn Buchstabe hinzugefügt wird
- +2, wenn Buchstabe ausgetauscht wird (löschen und hinzufügen) - manchmal wird nur +1 gewertet

Die effiziente Implementierung ist sehr Herausfordernd! Nichts desto trotz, sehr wichtiger und oft verwendeter Algorithmus, zum Beispiel als Wörterbuch. Betrachtet bei Fehler alle Wörter mit kleinster Distanz (Abweichung) und schlägt diese als Korrektur vor.

3.12 Jaccard Similarity for Sets

Betrachtet die Menge einzelnen Wörtere und dividiert sie durch die Gesamte Anzahl an Wörter. Der Wertebereich liegt zwischen $[0, 1]$ und ist definiert durch

$$jaccard(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

Sie kann bei Recommender Systemen angewendet werden. Zum Beispiel wenn zwei Warenkörbe gleiche Produkte enthalten oder wenn zwei Texte gleiches Jargon verwenden.

3.13 Haversine Distance for GEO Data

Flugzeuge reisen nach dieser Distanz, da sie die Krümmung der Erde berücksichtigt. Sie misst in der Atmosphäre die Distanz, ergo ist die Similarität klein, wenn die Distanz gross und vice-versa.

3.14 From Points to Distributions

Manchmal müssen auch Verteilungen beurteilt werden und nicht nur die Punkt zu Punkt Distanz. Es gibt auch die distribution-to-distribution, welche bei der Bildanalyse verwendet wird.

3.15 Mahalanobis Distance between Point and Distribution

Misst die Entfernung des Punkts als wie viele Standardabweichungen der Punkt vom Mittelwert entfernt liegt (Art Euklidische Distanz).

- Erst wird die Verteilung normalisiert indem korrelierende Variablen in unkorrelierende tranformiert werden
- dann wird skaliert, dass die Varianz gleich 1 wird
- zum Schluss wird die euklidsche Distanz zwischen Punkt X und dem Mittelwert berechnet

3.16 Distance may be sensitive to Scale of Axes

Die Einheit von Features können Daten verfälschen (zB. km -> m). Sie dominieren dann das Ergebnis einer anderen Einheit die viel kleiner ausfällt. Deshalb müssen vor dem betreiben von Machine Learning Daten normalisiert werden. Idealerweise haben alle Feature den gleichen Wertebereich.

3.17 Min-Max Normalization

Variante um Daten zu normalisieren und transformiert die Werte ins Intervall $[0, 1]$. Der Grösste Werte in der Spalte erhält den Wert 1, der kleinste der Wert 0. Die Werte dazwischen werden skaliert.

$$x \mapsto \frac{x - \min_X}{\max_X - \min_X}$$

Dies erlaubt die Interpretation in Prozent und man hat keine negativen Werte. Kann aber nicht für *supervised learning* verwendet werden, weil min/max nicht bekannt sind (nach dem Training).

3.18 Z-Score Normalisierung

Die Daten werden so transformiert, dass der Mittelwert 0 ist und die Standardabweichung 1.

$$x \mapsto \frac{x - \mu_X}{\sigma_X}$$

Kann für supervised und unsupervised learning verwendet werden. Der Nachteil ist die fehlende Prozentinterpretation. Kann zu negativen Werten führen (negative Preise oder Anzahl von etwas). Zur Interpretation müsste zurücktransformiert werden.

3.19 Normalization Parameters

Bei Min/Max Normalisation werden (min/max) benötigt, beim Z-Score (mean, std). Wichtig, Parameter aus Trainings-Daten bestimmen (nicht Testdaten), Min/Max nur für supervised learning verwenden (ausser Daten enthalten globale Min/Max). Nomalisierten Parameter speichern um später zu Skalieren.

3.20 K-Nearest Neighbors Classification (k-NN)

Ist wahrscheinlich der einfachste ML-A. Ist ein Verfahren für Regression und Klassifizierung. Zeigt auf, wie wichtig Distanz bzw. Similarität für ML ist.K-Nearest Neighbors muss ein von Hand definierten Parameter k mitgegeben werden. $k = 3$ bedeutet, dass K-Nearest die drei nächsten Punkte suchen und die Klassifikation eines neuen Punkts anhand deren Eigenschaften klassifiziert wird. k-NN mit $k = 1$ wird einfach das Labels des nahesten Punktes übernommen. Bei $k > 1$ wird ein mehrheitsvoting gegenüber den k-nahesten Punkte gemacht.

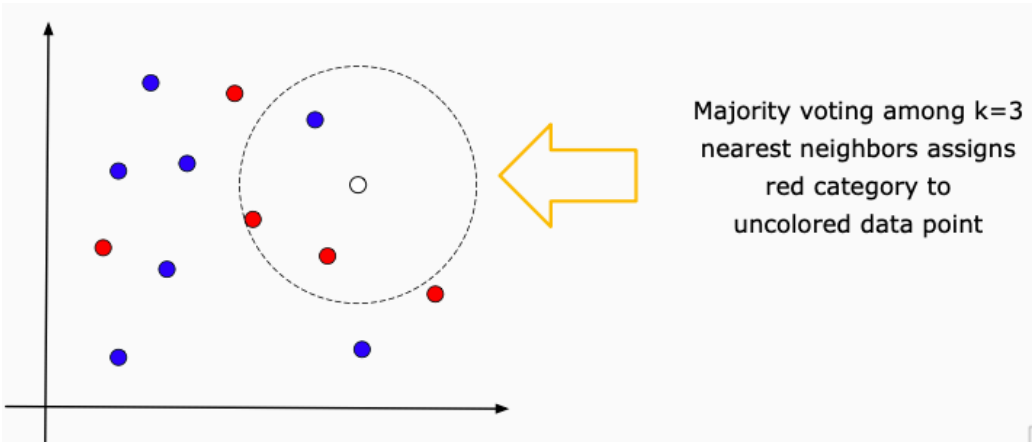

Abbildung 3.3: K-Nearest Neighbors Classification

3.21 K-Nearest Neighbors Regression

Löst Regressionsproblem zum Beispiel Preisvorhersage aus einem Regressionsmodell der k-nahesten Punkte. Funktionsweise mit Parameter k analog k-NN-Klassifizierung.

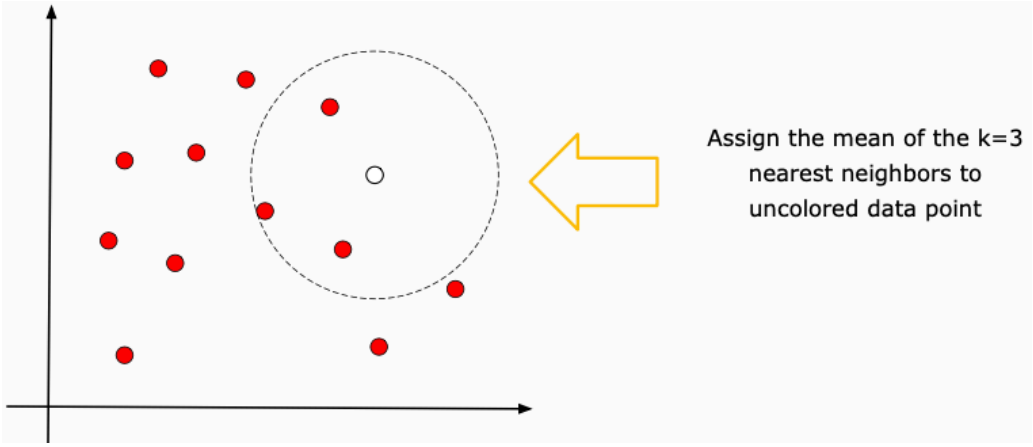

Abbildung 3.4: K-Nearest Neighbors Regression

3.22 Hyperparameter

Der Wert von k wird Hyperparameter genannt und entspricht der Anzahl Nachbarn. Das Resultat kann stark vom gewählten k differieren. Es ist also wichtig dieses möglichst optimal zu wählen (es gibt keine Optimierungsmöglichkeit durch einen Compi). Als weiteren Paramter kann die Distanz/Similaritätswert übergeben werden.

3.23 Facts on K-Nearest Neighbors

- Sehr langsam, weil jedesmal Similarität zu allen Punkten berechnet und dann die Distanz berechnet und nahesten ausgewählt. Dies wird bei jedem Datenpunkt gemacht
- Für kleine Dataset gute Baseline, legt Massstäbe fest für andere Algos
- Mehrheitswahl bedeutet alle Nachbarn sind gleich stimmberechtigt, egal wie weit sie entfernt liegen
- Alternativ kann per Parameter die Distanz berücksichtigt werden $\frac{1}{d}$ mit $d = \text{distance} > 0$
- benötigt am wenigsten Daten, reichen Daten nicht aus, gibt es keine Möglichkeit für ML