Klassifikationsverfahren	Logistic Regression, Decision Tree, Random Forest, Support Vector Machine, K Nearest Neighbour, Naïve Bayes
Die absolute Baseline (Constant)	Einfache Methode Vorhersage ML Häufigste Wert (Modus) des Klassenattributs **Beispiel:** Angenommen, du hast Trainingsdaten mit folgenden Klassenlabels: • **Ja, Ja, Nein, Ja, Ja** Hier ist die häufigste Klasse („Modus“) **„Ja“**. Das „Constant“-Modell würde also immer **„Ja“** vorhersagen, egal welche Eingabedaten gegeben werden. • Diese Methode dient oft als **Baseline,** um zu sehen, ob komplexere Modelle überhaupt besser abschneiden. In der Software Orange wird diese Methode als „Constant“ bezeichnet
Baseline: One Rule	Die **One Rule**-Baseline sucht das **Attribut**, das am besten mit dem Klassenattribut korreliert. **Vorgehen:** 1. Für jedes Attribut: ○ Zähle, welcher Klassenwert am häufigsten mit einem Attributwert auftritt. ○ Erstelle daraus eine Regel: **"Wenn A = Wert, dann C = häufigster Wert"**. 2. Berechne die Anzahl der Fehler dieser Regeln. 3. Wähle das Attribut mit den **wenigsten Fehlern**. **Beispiel:** • Kreditkartenbetrug: ○ Wenn **Kartentyp = Standard**, dann **kein Betrug**. ○ Wenn **Kartentyp = Gold**, dann **Betrug**. ○ Wenn **Kartentyp = Premium**, dann **kein Betrug**.
DT	• Ein Entscheidungsbaum teilt die Daten schrittweise auf. • Bei jedem Knoten kommen bestimmte Datenpunkte an. Diese Menge nennt man **Dt**. • Je nach Entscheidung (z.B. "mehr als 2 Reparaturen?") wird die Menge in **Teilgruppen** zerlegt.
Entscheidungsbäume Split Attribut wählen	Man wählt das **Attribut**, das die Daten am besten in **homogene Gruppen** aufteilt. • **Homogen:** Die Gruppe enthält fast nur eine Klasse (z.B. 9x „Ja“, 1x „Nein“) ☺ • ~~**Heterogen: Die Gruppe enthält gemischte Klassen (z.B. 5x „Ja“, 5x „Nein“)** ☹~~ Ziel: Möglichst **homogene Gruppen** bilden, um die Vorhersage zu verbessern.
Entscheidungsbäume Wichtige Parameter	Entscheidungsbäume werden oft **"abgeschnitten" (Pruning)**, um Überanpassung (Overfitting) zu vermeiden. **Wichtige Parameter:** 1. **Mindestgrösse der Blätter Dt:** ○ Ein Blattknoten (Endpunkt) muss mindestens eine bestimmte Anzahl an Datenpunkten haben (z.B. 50). 2. **Tiefe einschränken:** ○ Die maximale Tiefe des Baums wird begrenzt (z.B. maximal 2 Ebenen). **Ziel:** Den Baum **einfacher** und **allgemeiner** machen, damit er nicht zu stark auf die Trainingsdaten angepasst ist.
K Nearest Neighbour (kNN)	**KNN** ist ein Algorithmus, der die **nächsten Nachbarn** verwendet, um eine Vorhersage zu treffen. **Wie funktioniert KNN?**

	1. **Eingaben:** a. **Trainingsmenge:** Die bekannten Daten. b. **Ähnlichkeitsmass:** Meist die **Distanz** (z.B. euklidisch). c. **k-Wert:** Anzahl der nächsten Nachbarn, die berücksichtigt werden. **Vorhersage:** a. **Ähnlichkeit berechnen:** Finde die **k ähnlichsten Datenpunkte** zur neuen Instanz. b. **Mehrheitsentscheidung:** Die **häufigste Klasse** unter den k Nachbarn wird vorhergesagt.
kNN-Nachteile	**Zu klein:** a. Wenn **k** zu niedrig ist (z.B. **k=1**), kann ein **Ausreisser (Noise Point)** die Vorhersage stark beeinflussen. Beispiel: Ein einzelner falscher Wert kann die gesamte Entscheidung ändern. **Zu gross:** Wenn **k** zu hoch ist (z.B. **k=20**), werden **zu viele Nachbarn** berücksichtigt. Das kann dazu führen, dass die **Entscheidungsgrenze verschwimmt** und die Vorhersage ungenau wird. Vorteil: man kann Vorhersage trotzdem erklären indem man zeigt welche Nächste Nachbarn entscheidung beeinflusst haben,
Logistische Regression	Kombiniert alle Attribute zu einer gewichteten summe $z = w_1 x_1 + \cdots w_n x_n + b$ • **wi:** Gewicht des Attributs • **xi:** Wert des Attributs • **b:** Bias (Verschiebung) Um aus dieser Summe Wahrscheinlichkeit →Sigmoid Funktion $\phi(z) = \frac{1}{1 + e^{-z}}$ • Wenn ϕ(z)>0.5ϕ(z)>0.5 → **Ja (Beispiel)** • Wenn ϕ(z)≤0.5ϕ(z)≤0.5 → **Nein (Beispiel)**
Logistische Regression Nachteile	**Nachteil:** Die logistische Regression behandelt **alle Attribute unabhängig voneinander**. • Das bedeutet: Sie erkennt keine **Kombinationen** von Attributen. • Beispiel: o Angenommen, die Vorhersage lautet **"Ja"**, wenn: ▪ **Region = Stadt** und **Monate < 8** o Die logistische Regression kann diese **Verknüpfung** nicht erkennen, weil sie die Attribute **einzeln** betrachtet. • Problem: o Manchmal spielt ein Attribut nur in **Kombination mit einem anderen** eine Rolle. o Beispiel: **"Monate < 8"** allein bedeutet nichts, aber **zusammen mit "Stadt"** schon.

Gradient Boosting	Grundidee: Gradient Boosting kombiniert mehrere schwache Modelle (z.B. einfache Entscheidungsbäume), um ein stärkeres, genaueres Modell zu bauen. Wie funktioniert Gradient Boosting? 1. **Modell lernen:** o Starte mit einem einfachen Modell (z.B. ein kleiner Entscheidungsbaum). 2. **Fehler identifizieren:** o Finde heraus, wo das Modell falsch liegt. 3. **Weiteres Modell trainieren:** o Trainiere ein neues Modell, das die Fehler des ersten Modells korrigiert. 4. **Wiederholen:** o Mache das mehrmals, sodass jedes neue Modell die Fehler des vorherigen verbessert. 5. **Vorhersagen kombinieren:** o Kombiniere die Vorhersagen aller Modelle (jüngere Modelle haben weniger Gewicht). **Vorteil:** • Sehr genaue Vorhersagen, da die Fehler schrittweise reduziert werden. **Nachteil:** • Schwer zu interpretieren, da viele Modelle miteinander kombiniert werden.

Kriterien	Entscheidungsbäume	Neuronale Netze	Naïve Bayes	kNN	SVM	Regelbasierte Lernverfahren
Allgemeine Genauigkeit	**	****	**	**	****	**
Lerngeschwindigkeit (Anzahl der Attribute & Instanzen)	***	*	****	****	*	**
Klassifikationsgeschwindigkeit	****	****	****	*	****	****
Toleranz gegenüber fehlenden Werten	***	*	**	*	**	**
Toleranz gegenüber irrelevanten Attributen (Feature Selection)	***	**	*	*	****	**
Toleranz gegenüber redundanten Attributen	**	**	**	*	**	**
Toleranz gegenüber stark abhängigen Attributen (z. B. Paritätsprobleme)	**	***	*	*	**	*
Umgang mit diskreten/binären/kontinuierlichen Attributen	****	*** (nicht diskret)	*** (nicht kontinuierlich)	*** (nicht direkt diskret)	** (nicht diskret)	*** (nicht direkt kontinuierlich)
Toleranz gegenüber Rauschen	**	**	**	*	**	**
Umgang mit Overfitting	**	**	***	**	****	**
Möglichkeiten für inkrementelles Lernen	**	***	***	***	**	***
Erklärbarkeit/Transparenz der Klassifikation	****	*	**	*	*	****
Modell-Parameter-Handhabung	**	*	***	**	**	**