Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

HTW Berlin
Thema: Default Prediction with UCI Credit Card Default Data Set

Zaviera, Manin Tiola
Matrikelnummer: 565405

Fachbereich: Informatik, Kommunikation und Wirtschaft.
Studiengang: Finanzmathematik, Aktuarwissenschaften und Risikomanagement (M), WiSe 24/25
Lehrveranstaltung: Seminar WiSe 2024/2025

Berlin, 17. Februar 2025

Dozent : Dr. Alla Petukhina

Zusammenfassung

Diese Arbeit befasst sich mit der Vorhersage von Kreditkarten Zahlungsausfällen unter Verwendung des UCI Credit Card Default [6] Datensatzes der Universität UCI in Los Angeles.
Ziel der Untersuchung ist es, verschiedene Machine-Learning-Modelle zu analysieren und deren Leistungsfähigkeit systematisch zu vergleichen, um fundierte Erkenntnisse darüber zu gewinnen, welches Modell eine höhere Vorhersagegenauigkeit für Zahlungsausfälle liefert.
Dabei wird untersucht, welche Merkmale und Algorithmen die höchste Vorhersagegenauigkeit ermöglichen. Durch diesen Vergleich soll herausgearbeitet werden, welches Modell sich besser für die präzise Identifikation von Kreditrisiken eignet und somit einen wertvollen Beitrag zur Optimierung von Risikobewertungsverfahren im Finanzsektor leisten kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Vorhersage von Kreditkarten-Zahlungsausfällen ist ein zentrales Thema im Finanzsektor. Banken und Finanzinstitute setzen zunehmend auf maschinelle Lernverfahren, um Risiken besser bewerten und fundierte Entscheidungen bei der Kreditvergabe treffen zu können. Eine präzise Vorhersage von Zahlungsausfällen kann nicht nur finanzielle Verluste minimieren, sondern auch zu einer faireren Kreditvergabe beitragen.
Zahlreiche Studien haben bereits die Leistungsfähigkeit verschiedener Machine-Learning-Methoden zur Klassifikation von Zahlungsausfällen untersucht. Besonders hervorzuheben sind die Arbeiten von Clement Fung et al. [4] sowie Islam, Sheikh Rabiul et al. [5] , die das UCI Credit Card Dataset von I-Cheng, Yey [6] im Jahr 2018 verwendet haben, um unterschiedliche Ansätze zur Vorhersage von Zahlungsausfällen zu analysieren.
Eine zentrale Fragestellung dieser Arbeit ist die Bewertung der Performance verschiedener Modellierungsansätze. Abbildung 1 veranschaulicht einen Vergleich unterschiedlicher Methoden zur Klassifikation von Zahlungsausfällen. Dabei wird der Machine-Learning-Ansatz mit einem heuristischen Verfahren sowie dem aktuellen Stand der Technik (state-of-the-art.) verglichen. Die Ergebnisse zeigen, dass der Machine-Learning-Ansatz insbesondere bei Genauigkeit (Accuracy) und Präzision (Precision) bessere Werte erzielt als die anderen Verfahren, was ihn als bevorzugte Methode für diese Art von Vorhersagen qualifiziert.

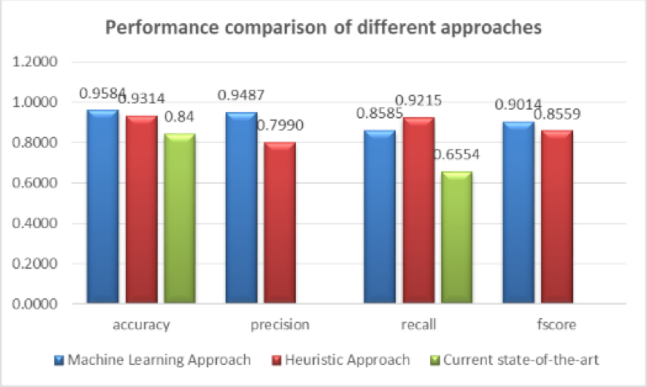


Abbildung 1: Vergleich der Leistung verschiedener Ansätze, 2018, [S.6] [5]

In dieser Arbeit werden daher Logistische Regression und XGBoost als Klassifikationsmodelle untersucht. Diese beiden Modelle wurden aufgrund ihrer weit verbreiteten Nutzung in der Finanzbranche sowie ihrer bewährten Leistungsfähigkeit in Klassifikationsproblemen ausgewählt. Während die logistische Regression ein einfaches und interpretierbares Modell darstellt, bietet XGBoost eine leistungsstarke, nicht-lineare Alternative mit hoher Vorhersagegenauigkeit. Ziel dieser Arbeit ist es, die Leistungsfähigkeit dieser Modelle systematisch zu vergleichen, um fundierte Erkenntnisse darüber zu gewinnen, welches Modell sich besser für die Vorhersage von Kreditkarten-Zahlungsausfällen eignet.

2 Motivation und Forschungsfragen

2.1 Motivation

Die steigende Nutzung von Kreditkarten weltweit erhöht die Notwendigkeit, Zahlungsausfälle präzise vorherzusagen. Eine zuverlässige Risikobewertung kann nicht nur Finanzinstitute vor erheblichen Verlusten bewahren, sondern auch eine fairere Kreditvergabe für Verbraucher ermöglichen.

Die Hauptmotivation dieser Arbeit ist die Entwicklung und Evaluierung von maschinellen Lernmodellen, die Finanzinstitute dabei unterstützen, das Risiko eines Zahlungsausfalls frühzeitig zu erkennen. Durch die Analyse finanzieller und persönlicher Merkmale von Kreditnehmern sollen Modelle geschaffen werden, die eine fundierte Entscheidungsgrundlage für Banken bieten. Dies ermöglicht eine bessere Einschätzung von Kreditrisiken und trägt dazu bei, verantwortungsbewusste Kreditvergaben zu gewährleisten.
Darüber hinaus liegt ein weiteres zentrales Anliegen dieser Untersuchung in der Leistungsanalyse verschiedener Data-Mining-Algorithmen zur Kreditrisikovorhersage. Es soll untersucht werden, wie effizient und zuverlässig unterschiedliche Algorithmen in der Praxis arbeiten, um so die leistungsfähigsten Modelle für diese spezifische Anwendung zu identifizieren.

2.2 Forschungsfragen

Die Analyse der Kreditkartenschulden in den USA [3] zeigt zwei bedeutende Trends: Erstens die stetig steigende Verschuldung der Verbraucher seit 2004 mit markanten Ausschlägen während der Finanzkrise 2010 und der COVID-19-Pandemie 2020. Zweitens die sogenannte *Delinquency Rate*, also die Rate der nicht bedienten Kredite, die während wirtschaftlicher Krisen stark ansteigt. Diese Beobachtungen führen zu folgenden Forschungsfragen:

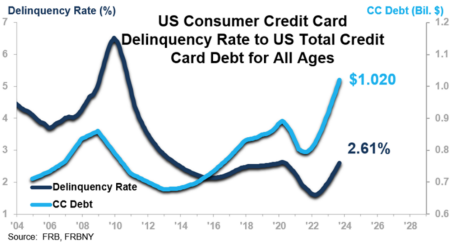


Abbildung 2: US Consumers Posting Record-High Credit Card Debt Balances.

- Können moderne Machine-Learning-Modelle präzisere und robustere Vorhersagen für Kreditausfälle liefern?
- Inwieweit können maschinelle Lernmethoden dazu beitragen, die Vorhersagegenauigkeit von Kreditausfällen zu verbessern?
- Welche Merkmale sind am besten geeignet, um Zahlungsausfälle frühzeitig zu identifizieren und somit das Risiko für Finanzinstitute zu minimieren?

3 Datenbeschreibung und Datentransformation

Der Datensatz [6] stammt aus dem UCI Machine Learning Repository und enthält Informationen zu 30.000 Kreditkartenkunden in Taiwan. Eine Übersicht über die verschiedenen Merkmale ist in Tabelle 1 zu sehen. Die wichtigsten Merkmale sind folgende:

- **Demografische Merkmale**: Geschlecht, Alter, Bildungsniveau, Familienstand.
- **Kreditinformationen**: Höhe des gewährten Kredits in NTD (Neuer Taiwan Dollar), einschließlich des individuellen Verbraucherkredits sowie eventueller Familienkredite.
- **Historische Zahlungsausfälle**: Monatliche Zahlungsinformationen von April bis September 2005. Der Zahlungsstatus wird skaliert von -1 bis 9.
- **Monatliche Abrechnungen**: Höhe der Rechnungsstellung für die Monate April bis September 2005.
- **Frühere Zahlungen**: Höhe der geleisteten Zahlungen für die Monate April bis September 2005.

Diese Untersuchung [6] konzentriert sich auf die Vorhersage von Zahlungsausfällen von Kunden in Taiwan und vergleicht die Vorhersagegenauigkeit verschiedener Data-Mining-Methoden. Im Bereich des Risikomanagements ist die Wahrscheinlichkeit eines Zahlungsausfalls eine wertvollere Metrik als eine binäre Klassifikation (kreditwürdig oder nicht kreditwürdig).
Da die tatsächliche Wahrscheinlichkeit eines Zahlungsausfalls nicht bekannt ist, wurde in dieser Studie die Sorting Smoothing Methode verwendet, um eine genauere Schätzung der Ausfallwahrscheinlichkeit zu ermöglichen. Die geschätzte Wahrscheinlichkeit eines Zahlungsausfalls wird dabei als unabhängige Variable X betrachtet, während die tatsächliche Zahlungsausfallrate die abhängige Variable Y darstellt.

Kategorische Variablen
Geschlecht, EDUCATION, MARRIAGE.
Beispiel: Erste Zeile = 2, 2, 1.

Kontinuierliche Variablen
LIMIT_BAL, AGE, BILL_AMT bleiben unverändert.

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13
ID	LIMIT_BAL	SEX	EDUCATI	MARRIAG	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	20000	2	2	1	24	2	2	-1	-1	-2	-2	3913	3102
2	120000	2	2	2	26	-1	2	0	0	0	2	2682	1725
3	90000	2	2	2	34	0	0	0	0	0	0	29239	14027
4	50000	2	2	1	37	0	0	0	0	0	0	46990	48233
5	50000	1	2	1	57	-1	0	-1	0	0	0	8617	5670
6	50000	1	1	2	37	0	0	0	0	0	0	64400	57069
7	500000	1	1	2	29	0	0	0	0	0	0	367965	412023
8	100000	2	2	2	23	0	-1	-1	0	0	-1	11876	380
9	140000	2	3	1	28	0	0	2	0	0	0	11285	14096
10	20000	1	3	2	35	-2	-2	-2	-2	-1	-1	0	0
11	200000	2	3	2	34	0	0	2	0	0	-1	11073	9787
12	260000	2	1	2	51	-1	-1	-1	-1	-1	2	12261	21670
13	630000	2	2	2	41	-1	0	-1	-1	-1	-1	12137	6500
14	70000	1	2	2	30	1	2	2	0	0	2	65802	67369
15	250000	1	1	2	29	0	0	0	0	0	0	70887	67060
16	50000	2	3	3	23	1	2	0	0	0	0	50614	29173
17	20000	1	1	2	24	0	0	2	2	2	2	15376	18010
18	320000	1	1	1	49	0	0	0	-1	-1	-1	253286	246536
19	360000	2	1	1	49	1	-2	-2	-2	-2	-2	0	0
20	180000	2	1	2	29	1	-2	-2	-2	-2	-2	0	0

Zahlungshistorie (PAY_0 bis PAY_6)
Diese Variablen werden als *ordinales* Merkmal behandelt: Größere Werte bedeuten schlechtere Zahlungshistorie.

4 Methodologie: XGBoost und Logistische Regression

Für diese Arbeit werden die Modelle Logistische Regression und Gradient Boosting eingesetzt, da es sich um ein multivariates Klassifikationsproblem handelt. Der zugrunde liegende Datensatz umfasst mehrere erklärende Variablen und eine multiklassige Zielvariable, die klassifiziert werden muss.
Die logistische Regression ([S.79] [7]) wird aufgrund ihrer schnellen Trainingsgeschwindigkeit und einfachen Handhabung für Klassifikationsprobleme gewählt. Butch Quinto beschreibt sie als einen linearen Klassifikator, der Wahrscheinlichkeiten für verschiedene Klassen berechnet und sowohl für binäre als auch für multiklassige Klassifikationen geeignet ist [2][S.104]. Besonders in Finanzanwendungen erweist sie sich als nützlich, da sie klare Entscheidungsgrenzen definiert und somit eine zuverlässige Trennung der Klassen ermöglicht.
XGBoost hingegen ist ein leistungsstarkes Gradient Boosting Modell, das insbesondere bei komplexen Klassifikationsaufgaben mit hoher Dimensionalität und nichtlinearen Zusammenhängen überlegen ist. Es nutzt eine sequentielle Baumbildung, um Fehler systematisch zu minimieren und bietet Mechanismen zur Regularisierung, die Overfitting verhindern.

4.1 Definition

Quinto [2] [S.133] definiert XGBoost als eine der leistungsfähigsten derzeit verfügbaren Implementierungen von gradientengeboosteten Entscheidungsbäumen. Es wurde am 27. März 2014 von Tianqi Chen als Forschungsprojekt veröffentlicht und hat sich seither als dominierender Machine-Learning-Algorithmus für Klassifikation und Regression etabliert. XGBoost basiert auf den allgemeinen Prinzipien des Gradient Boosting, das schwache Lernalgorithmen kombiniert, um einen stärkeren Gesamtalgorithmus zu erzeugen. Während herkömmliche gradientengeboostete Bäume sequentiell erstellt werden also aus den Daten lernen, um ihre Vorhersagen in nachfolgenden Iterationen zu verbessern – werden bei XGBoost die Bäume parallel aufgebaut (Quinto, 2020, S. 10).

4.2 Ziel

XGBoost erzielt eine überlegene Vorhersageleistung durch die Kontrolle der Modellkomplexität und die Reduzierung von Überanpassung mittels integrierter Regularisierung. Dazu nutzt XGBoost einen approximativen Algorithmus, um optimale Aufteilungspunkte für kontinuierliche Merkmale zu finden. Diese Methode basiert auf der Gruppierung von kontinuierlichen Merkmalen in diskrete Bins, was das Training erheblich beschleunigt. Darüber hinaus enthält XGBoost eine zusätzliche Baumwachstumsmethode, die einen histogrammbasierten Algorithmus verwendet. Dieser ermöglicht eine effizientere Gruppierung kontinuierlicher Merkmale in diskrete Bins. Während die approximative Methode in jeder Iteration eine neue Menge an Bins erstellt, nutzt der histogrammbasierte Ansatz vorhandene Bins über mehrere Iterationen hinweg wieder ([S. 10] [2]).
Durch den Vergleich dieser beiden Methoden kann die Leistungsfähigkeit linearer und nichtlinearer Modelle untersucht werden, um das beste Modell für die Vorhersage von Zahlungsausfällen zu identifizieren.

5 Mathematische Analyse

5.1 XGBoost

XGBoost [12] wird für überwachtes Lernen verwendet, bei dem wir Trainingsdaten (mit mehreren Merkmalen) x_i nutzen, um eine Zielvariable y_i vorherzusagen. Bevor wir speziell auf Entscheidungsbäume eingehen, beginnen wir mit einer Übersicht über die grundlegenden Elemente des überwachten Lernens.
Das Modell im überwachten Lernen bezieht sich normalerweise auf die mathematische Struktur, mit der die Vorhersage $\hat{y}_i$ aus den Eingaben x_i erstellt wird. Ein häufiges Beispiel ist ein *lineares Modell*, bei dem die Vorhersage durch

$$\hat{y}_i = \sum_j \theta_j x_{ij}$$

gegeben ist eine lineare Kombination von gewichteten Eingangsmerkmalen. Der Vorhersagewert kann je nach Aufgabe unterschiedliche Interpretationen haben, z. B. für Regression oder Klassifikation. Beispielsweise kann eine logistische Transformation verwendet werden, um die Wahrscheinlichkeit einer positiven Klasse in der logistischen Regression zu erhalten, oder der Wert kann als Ranking-Score genutzt werden, wenn es darum geht, die Ausgaben zu ordnen.
Die Parameter sind der nicht festgelegte Teil des Modells, den wir aus den Daten lernen müssen. In linearen Regressionsproblemen sind die Parameter die Koeffizienten θ. Normalerweise verwenden wir θ, um die Parameter zu bezeichnen (da es in einem Modell viele Parameter geben kann, ist unsere Definition hier vereinfacht).

5.1.1 Zielfunktion: Trainingsverlust und Regularisierung

Die Zielfunktion [12] eines Modells setzt sich aus zwei Hauptkomponenten zusammen: dem *Trainingsverlust* und einem *Regularisierungsterm*. Sie dient dazu, die optimale Anpassung des Modells an die Daten sicherzustellen, während gleichzeitig eine Überanpassung vermieden wird. Mathematisch ist sie definiert als:

$$L(\Theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Hierbei bedeuten:

- Θ: Der Vektor der Merkmale (Features).
- y_i: Der wahre Zielwert.
- $\hat{y}_i$: Die vorhergesagte Ausgabe des Modells.
- $l(y_i, \hat{y}_i)$: Die Verlustfunktion, die den Fehler zwischen Vorhersage und tatsächlichem Wert misst (z. B. Mittlerer Quadratischer Fehler, MSE).
- $\Omega(f_k)$: Der Regularisierungsterm, der die Modellkomplexität bestraft, um eine bessere Generalisierbarkeit zu gewährleisten.

Der erste Summand in der Formel stellt sicher, dass das Modell möglichst genaue Vorhersagen trifft, während der zweite Summand verhindert, dass das Modell zu komplex wird und sich zu stark an die Trainingsdaten anpasst. Dieses Zusammenspiel wird auch als *Bias-Varianz-Tradeoff* bezeichnet.
Ein wesentliches Merkmal von Zielfunktionen ist, dass sie aus zwei Teilen bestehen:

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta)$$

Hierbei ist $L(\theta)$ die Trainingsverlustfunktion, und $\Omega(\theta)$ der Regularisierungsterm.
Die häufig verwendete Verlustfunktion zur Messung des Modellfehlers ist der **Mittlere Quadratische Fehler (Mean Squared Error, MSE)**:

$$L(\theta) = l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$$

Eine weitere gängige Verlustfunktion ist der logistische Verlust (**Log-Loss**), der in der logistischen Regression verwendet wird:

$$L(\theta) = l(y_i, \hat{y}_i) = -\left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)\right]$$

5.1.2 Regularisierungsterm

Die Regularisierung spielt eine zentrale Rolle bei der Kontrolle der Modellkomplexität [12] und dient dazu, Überanpassung (Overfitting) zu verhindern. Durch das Hinzufügen eines Regularisierungsterms zur Zielfunktion wird die Komplexität des Modells begrenzt, sodass es nicht nur die Trainingsdaten gut abbildet, sondern auch auf neue, unbekannte Daten verallgemeinerbar bleibt.
In der Praxis führt ein zu komplexes Modell dazu, dass es sich zu stark an die Trainingsdaten anpasst und somit Rauschen in den Daten fälschlicherweise als relevante Muster interpretiert. Die Regularisierung sorgt für eine Balance zwischen Modellanpassung und Generalisierungsfähigkeit. Eine visuelle Darstellung dieses Effekts wird in Abbildung 3 veranschaulicht, in der verschiedene Regularisierungsstärken miteinander verglichen werden. Dabei zeigt sich, dass eine zu geringe Regularisierung zu einer hohen Varianz führt, während eine zu starke Regularisierung die Modellleistung durch zu hohe Verzerrung (Bias) verschlechtern kann.
Um die Wirkung der Regularisierung mathematisch zu beschreiben, wird der Regularisierungsterm $\Omega(F)$ definiert. Dieser setzt sich aus zwei Bestandteilen zusammen: einem Parameter γ, der die Anzahl der Blätter im Entscheidungsbaum bestraft, und einem zweiten Term, der die Blattgewichte reguliert:

$$\Omega(F) = \gamma T + \frac{\lambda}{2} \sum_{j=1}^{T} w_j^2$$

- **T**: Anzahl der Blätter in einem Entscheidungsbaum.
- w_j: Das Gewicht des j-ten Blatts.
- γ: Regularisierungsparameter, der die Anzahl der Blätter bestraft.
- λ: Regularisierungsparameter, der die Größe der Blattgewichte bestraft.

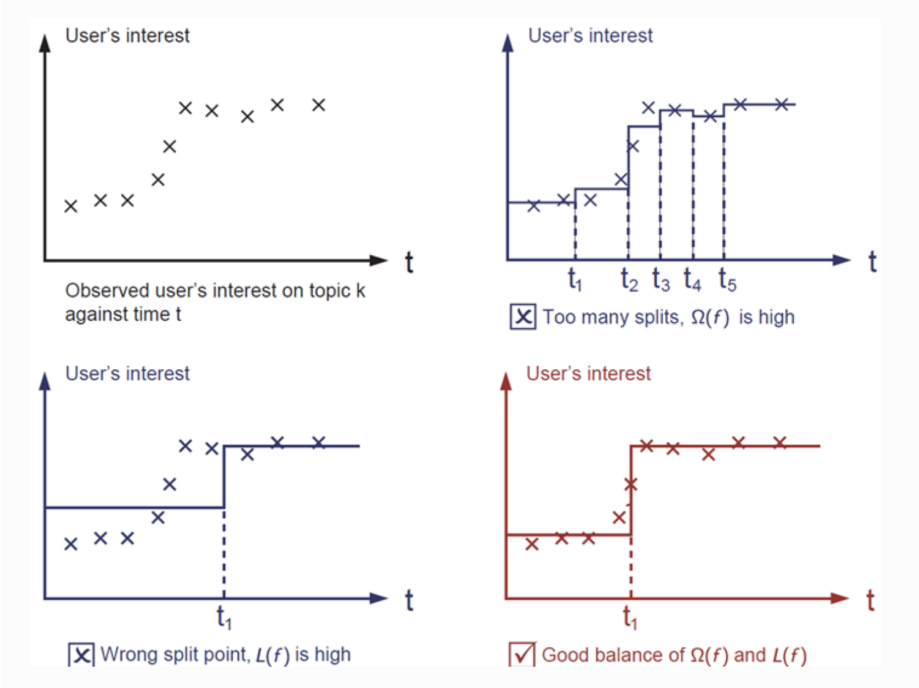


Abbildung 3: Regularisierung
`https://xgboost.readthedocs.io/en/latest/tutorials/model.html`

5.1.3 Additives Modell

XGBoost verwendet ein additives Modell, bei dem Entscheidungsbäume sequentiell aufgebaut werden. In jedem Schritt t wird ein neuer Baum $f_t(x)$ hinzugefügt, um die Zielfunktion zu minimieren. Die Vorhersage für eine gegebene Instanz x_i wird dabei sukzessive verbessert: [12]

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$$

Hierbei bedeuten:

- $\hat{y}_i^{(t-1)}$: Die Vorhersage nach $t-1$ Bäumen.
- $f_t(x_i)$: Die Vorhersage des neuen Baums im Schritt t.

Ein Beispiel für ein Baum-Ensemble aus zwei Entscheidungsbäumen ist in Abbildung 4 dargestellt. Die Vorhersagewerte der einzelnen Bäume werden aufsummiert, um das endgültige Ergebnis zu erhalten. Ein wichtiger Aspekt dieses Verfahrens ist, dass sich die Bäume gegenseitig ergänzen und somit eine robustere Vorhersage ermöglichen.
Mathematisch lässt sich dieses Modell wie folgt darstellen. Der Vorhersageprozess beginnt mit einem initialen Wert von Null:

$$\hat{y}_i^{(0)} = 0$$

Anschließend werden sukzessive neue Bäume hinzugefügt, um die Modellleistung zu verbessern:

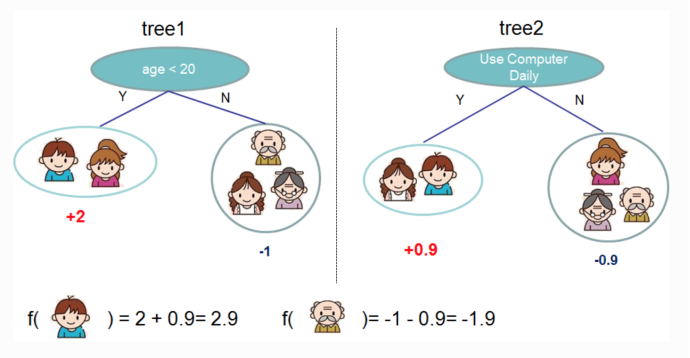


Abbildung 4: Baum-Ensemble
`https://xgboost.readthedocs.io/en/latest/tutorials/model.html`

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i)$$

$$\vdots$$

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

Dieses schrittweise Hinzufügen von Bäumen erlaubt es dem Modell, sich iterativ zu verbessern und sich optimal an die zugrunde liegenden Datenstrukturen [12] anzupassen. Durch dieses Verfahren wird sichergestellt, dass das Modell flexibel und leistungsfähig bleibt, ohne die Gefahr einer Überanpassung an die Trainingsdaten zu stark zu erhöhen.

5.1.4 Approximation mit Taylor-Entwicklung

Um die Zielfunktion effizient zu optimieren, verwendet XGBoost eine Näherung auf Basis der zweiten Ordnung der Taylor-Entwicklung. Diese erlaubt es, die Verlustfunktion in Bezug auf die vorhergesagten Werte $\hat{y}_i$ zu approximieren:

$$L^{(t)} = \sum_{i=1}^{n} \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

Hierbei stehen:

- g_i: Erste Ableitung (Gradient) der Verlustfunktion.
- h_i: Zweite Ableitung (Hesse-Matrix) der Verlustfunktion.

Durch diese Näherung wird die Optimierungsaufgabe darauf reduziert, die Funktion $f_t(x)$ zu finden, die die Zielfunktion minimiert. Der Gradient gibt die

Richtung der Fehlerreduktion an, während die Hesse-Matrix zur Anpassung der Schrittgröße für eine bessere Konvergenz verwendet wird.
Der Gradient g_i und die Hesse-Matrix h_i sind definiert als:

$$g_i = \frac{\partial}{\partial \hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

$$h_i = \frac{\partial^2}{\partial \hat{y}_i^{(t-1)^2}} l(y_i, \hat{y}_i^{(t-1)})$$

Diese Ableitungen ermöglichen eine effiziente Anpassung der Modellparameter und verbessern die Konvergenzgeschwindigkeit des Algorithmus.

5.1.5 Strukturbewertung eines Entscheidungsbaums

Ein zentraler Bestandteil der Modelloptimierung in XGBoost ist die Bewertung der Baumstruktur. [12] Durch eine mathematische Umformulierung des Baum-Modells lässt sich die Zielfunktion für den t-ten Baum wie folgt darstellen:

$$\text{obj}^{(t)} \approx \sum_{i=1}^{n} \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2 \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

Durch die Gruppierung der Summationsindizes kann dies weiter vereinfacht werden zu:

$$\sum_{j=1}^{T} \left(\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right) + \gamma T$$

Hierbei ist I_j die Menge der Datenpunkte, die dem Blatt j zugewiesen sind. Da alle Datenpunkte im selben Blatt denselben Wert erhalten, kann die Formel durch folgende Definitionen weiter verdichtet werden:

$$G_j = \sum_{i \in I_j} g_i, \quad H_j = \sum_{i \in I_j} h_i$$

Damit ergibt sich die Zielfunktion als:

$$\text{obj}^{(t)} = \sum_{j=1}^{T} \left(G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2 \right) + \gamma T$$

Da w_j voneinander unabhängig sind, lässt sich das optimale w_j^* für eine gegebene Baumstruktur $q(x)$ durch Ableiten berechnen:

$$w_j^* = -\frac{G_j}{H_j + \lambda}$$

Setzt man dies in die Zielfunktion ein, erhält man die beste erreichbare Reduktion der Zielfunktion:

$$\text{obj}^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T$$

Diese Formel misst, wie gut eine gegebene Baumstruktur $q(x)$ ist. Je größer die Reduktion, desto besser eignet sich die Struktur zur Fehlerreduktion im Modell.

Um das Konzept besser zu verstehen, betrachten wir eine beispielhafte Baumstruktur (siehe Abbildung 5). Hierbei werden die Gradientenstatistiken g_i und h_i für jede Instanz gesammelt und den entsprechenden Blättern zugewiesen. Die Summen dieser Werte ermöglichen es, mithilfe der obigen Formeln die Qualität der Baumstruktur zu bewerten. Der erhaltene Wert dient als Maß für die Modellkomplexität und erlaubt eine gezielte Optimierung der Entscheidungsbäume.

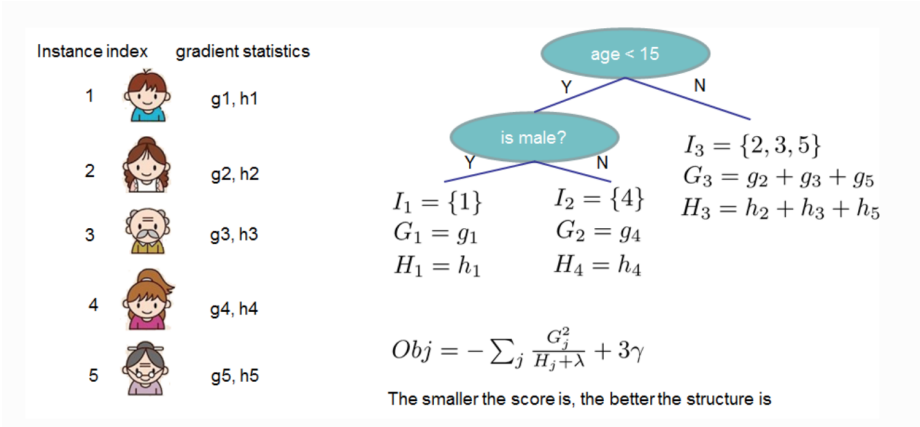


Abbildung 5: Baum-Struktur
`https://xgboost.readthedocs.io/en/latest/tutorials/model.html`

5.1.6 Split-Kriterium

Um den besten Split in einem Entscheidungsbaum zu bestimmen, berechnet XGBoost den **Gain**, der durch das Aufteilen eines Knotens erzielt wird. Das Split-Kriterium ist gegeben durch [12]:

$$Gain = \frac{1}{2}\left[\frac{(G_L)^2}{H_L+\lambda} + \frac{(G_R)^2}{H_R+\lambda} - \frac{(G)^2}{H+\lambda}\right] - \gamma$$

Hierbei bedeuten:

- G, H: Gradient und Hesse-Matrix (Fehler und Unsicherheit).
- λ, γ: Regularisierungsparameter.

Falls der Gewinn eines Splits kleiner als γ ist, wird der Split nicht durchgeführt. Dies dient dazu, überflüssige Knoten zu vermeiden und das Modell effizient zu halten.
Die Formel kann in vier Hauptbestandteile zerlegt werden:

1. Der Score des neuen linken Blattes G_L, H_L
2. Der Score des neuen rechten Blattes G_R, H_R

3. Der Score des ursprünglichen Blattes
4. Die Regularisierung auf das zusätzliche Blatt

Um das optimale Split-Kriterium effizient zu berechnen, werden die Datenpunkte in aufsteigender Reihenfolge sortiert und dann von links nach rechts gescannt (siehe Abbildung 7). Dadurch kann der beste Split schnell und effizient gefunden werden.

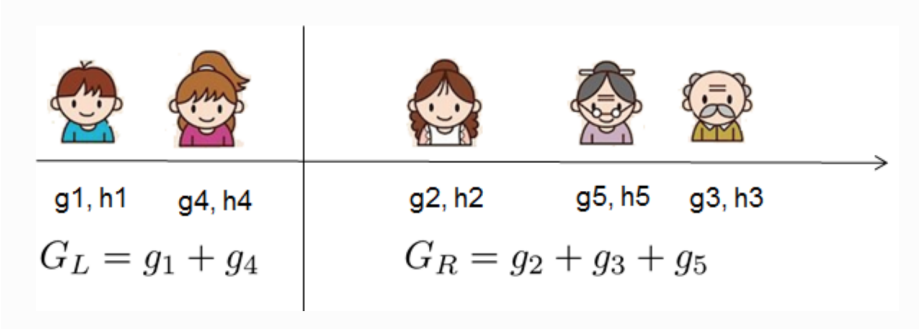


Abbildung 6: Split-Kriterium
`https://xgboost.readthedocs.io/en/latest/tutorials/model.html`

5.1.7 Optimierung der Blattgewichte

Nach der Aufteilung eines Knotens wird für jedes Blatt das optimale Gewicht berechnet, um die Zielfunktion weiter zu minimieren:

$$\omega_j = -\frac{\sum_{i \in l_j} g_i}{\sum_{i \in l_j} h_i + \lambda}$$

Hierbei bedeuten:

- l_j: Die Menge der Datenpunkte in Blatt j.
- g_i, h_i: Gradient und Hesse-Matrix für jeden Datenpunkt im Blatt.

Diese Formel stellt sicher, dass die Blattwerte optimal angepasst werden, um die Fehlerreduktion zu maximieren und die Modellleistung weiter zu verbessern.

5.2 Logistische Regression

5.2.1 Modellformulierung

Da unser Datensatz eine *multinomiale Klassifikation* (Rückzahlungsstatus) erfordert, verwenden wir die Softmax-Regression.
Die *logistische Funktion* (Sigmoidfunktion) ist laut Stefan Richter [10] [S.225]definiert als:

$$\mathbb{P}(Y = 1 \mid \mathbf{X}) = \frac{e^{\beta_0 + \beta^T \mathbf{X}}}{1 + e^{\beta_0 + \beta^T \mathbf{X}}} \quad (1)$$

mit

$$z = \beta_0 + \sum_{i=1}^{n} \beta_i x_i \tag{2}$$

wobei:

- x_i die Eingangsmerkmale (z. B. Kreditlimit, Zahlungshistorie) sind,
- β_i die Modellparameter (Gewichte) sind.

Für die *multinomiale logistische Regression* (Softmax-Regression) ergibt sich:

$$\mathbb{P}((Y = k \mid \mathbf{X}) = \frac{e^{\beta_{k0} + \beta_{\mathbf{k}}{}^T \mathbf{X}}}{\sum_{l=1}^{K} e^{\beta_{l0} + \beta_{\mathbf{l}}{}^T \mathbf{X}}} \tag{3}$$

wobei K die Anzahl der Klassen ist.

5.2.2 Log-Loss-Funktion

Die Optimierung des Modells erfolgt über die *Log-Loss-Funktion* (negative Log-Likelihood):

$$J(\beta) = -\sum_{i=1}^{m} \sum_{k=1}^{K} y_{i,k} \log \mathbb{P}(Y = k \mid \mathbf{X_i}) \tag{4}$$

Hierbei gilt:

- $\log\left(\frac{\mathbb{P}(Y=k|\mathbf{X})}{\mathbb{P}(Y=k'|\mathbf{X})}\right) = (\beta_{k0} - \beta_{k'0}) + (\beta_k - \beta_{k'})^T \mathbf{X}$
- $k' = 1, \ldots, K$
- $y_{i,k}$ ist die binäre Indikatorvariable, die angibt, ob die Beobachtung i zur Klasse k gehört.
- $\mathbb{P}(Y = k \mid \mathbf{X_i})$ ist die durch das Modell vorhergesagte Wahrscheinlichkeit.

5.2.3 Optimierung durch Gradientenabstieg

Die logistische Regression wird mithilfe des Gradientenabstiegs trainiert. Die Gradienten der Log-Loss-Funktion bestimmen die Richtung der Parameteranpassung:

$$\beta_j := \beta_j - \alpha \frac{\partial J(\beta)}{\partial \beta_j} \tag{5}$$

wobei α die *Lernrate* ist.

5.2.4 Interpretation der Modellkoeffizienten

Jeder Koeffizient β_j gibt an, wie stark das entsprechende Merkmal die Wahrscheinlichkeit beeinflusst, dass ein Kunde einen Zahlungsverzug hat.

- Ein **positiver Wert** bedeutet ein höheres Risiko für Zahlungsverzug.
- Ein **negativer Wert** bedeutet ein geringeres Risiko.

6 Bewertung der Ergebnisse

6.1 Konfusionsmatrix

Laut Butch Quinto [2], S.20, besteht bei Klassifikationsaufgaben jeder Datenpunkt aus einem bekannten Label und einer durch ein Modell vorhergesagten Klasse. Durch den Vergleich dieser beiden Werte lassen sich die Ergebnisse in vier Kategorien einteilen: True Positive (TP), wenn sowohl die vorhergesagte Klasse als auch das tatsächliche Label positiv sind; True Negative (TN), wenn beide negativ sind; False Positive (FP), wenn die vorhergesagte Klasse positiv, das tatsächliche Label jedoch negativ ist; und False Negative (FN), wenn die vorhergesagte Klasse negativ, das tatsächliche Label jedoch positiv ist. Diese vier Werte bilden die Grundlage der meisten Metriken zur Evaluierung von Klassifikationsmodellen und werden häufig in einer sogenannten Konfusionsmatrix dargestellt (siehe Tabelle 1).
Die Modelle werden anhand verschiedener Leistungsmetriken bewertet, darunter Genauigkeit (Accuracy), Präzision, Recall und F1-Score. Zudem werden die Feature-Importance-Analyse sowie die Vorhersagegenauigkeit und Robustheit der Modelle untersucht. Ein weiterer wichtiger Aspekt ist die Interpretierbarkeit der Modelle, insbesondere im Hinblick auf ihre praktische Anwendbarkeit in der Kreditrisikobewertung.

Zielvariable	Vorhergesagt: 0	Vorhergesagt: 1
Kein Zahlungsausfall (0)	TN = **16.350**	FP = **1.164**
Zahlungsausfall (1)	FN = **3.205**	TP = **1.781**

Tabelle 1: Konfusionsmatrix für den XGBoost-Klassifikator

1. Interpretation

Die Konfusionsmatrix zeigt die Verteilung der tatsächlichen und vorhergesagten Klassen:

- **TN = 16.350:** Fälle, in denen das Modell korrekt keinen Zahlungsausfall vorhergesagt hat.
- **FP = 1.164:** Kunden, die fälschlicherweise als Zahlungsausfall klassifiziert wurden.
- **FN = 3.205:** Tatsächliche Zahlungsausfälle, die das Modell nicht erkannt hat.
- **TP = 1.781:** Fälle, in denen das Modell korrekt einen Zahlungsausfall vorhergesagt hat.

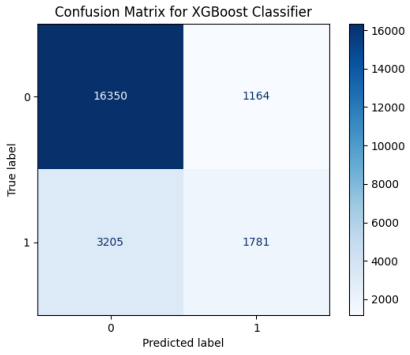


Abbildung 7: Diagramm der Konfusionsmatrix [Quelle: Eigene Darstellung]

2. Leistungsbewertung

Basierend auf der Konfusionsmatrix berechnen wir die wichtigsten Metriken. Die Evaluationsmetriken für die Modellbewertung sind definiert als:
Genauigkeit (Accuracy):

$$\text{Accuracy} = \frac{16350 + 1781}{16350 + 1781 + 1164 + 3205} \approx 81.8\% \tag{6}$$

Präzision für Zahlungsausfälle (Precision, Klasse 1):

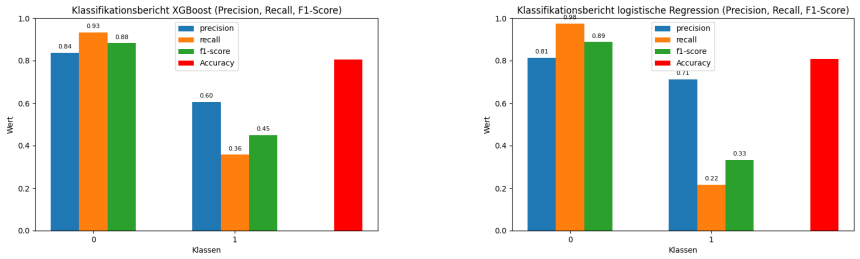
$$\text{Precision} = \frac{1781}{1781 + 1164} \approx 60.5\% \tag{7}$$

Recall (Empfindlichkeit) für Zahlungsausfälle (Klasse 1):

$$\text{Recall} = \frac{1781}{1781 + 3205} \approx 35.7\% \tag{8}$$

6.2 Modellbewertung

Die folgende Abbildungen 8 zeigen den Vergleich der wichtigsten Klassifikationsmetriken (Präzision, Recall, F1-Score und Accuracy) zwischen dem XGBoost- und dem logistischen Regressionsmodell.

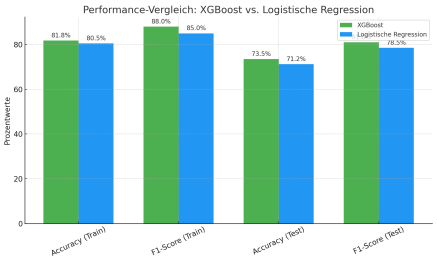


(a) XGBoost-Modells (b) Logistische Regression-Modell

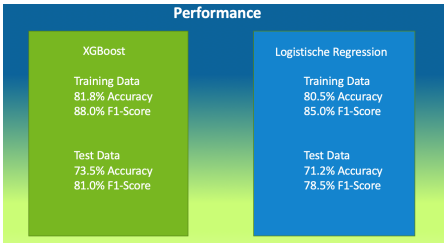
Abbildung 8: Vergleich der Klassifikationsmetriken zwischen XGBoost und logistischer Regression [Quelle: Eigene Darstellung]

Analyse der Modellleistung

Das **XGBoost-Modell**: (Abbildung 8a) zeigt insgesamt eine hohe Modellleistung. Die Präzision für Klasse 0 beträgt 0.84 und für Klasse 1 liegt sie bei 0.60, was bedeutet, dass in 60 % der Fälle ein vorhergesagter Zahlungsausfall korrekt war. Der Recall ist für Klasse 0 mit 0.93 sehr hoch, jedoch für Klasse 1 mit 0.36 vergleichsweise niedrig, was darauf hinweist, dass viele tatsächliche Zahlungsausfälle nicht erkannt werden. Der F1-Score ergibt sich für Klasse 0 zu 0.88 und für Klasse 1 zu 0.45, wodurch deutlich wird, dass das Modell Schwierigkeiten bei der Vorhersage von Zahlungsausfällen hat. Die Gesamtgenauigkeit des Modells beträgt 80 %, was darauf hindeutet, dass die Mehrheit der Fälle korrekt klassifiziert wurde.



(a) Vergleich Performance



(b) Vergleich Performance

Abbildung 9: Ergebniss der Klassifikationsmetriken [Quelle: Eigene Darstellung]

Im Vergleich dazu zeigt das **Logistische Regression Modell**: (Abbildung 8b) eine unterschiedliche Leistung. Die Präzision beträgt 0.81 für Klasse 0 und 0.71 für Klasse 1, was darauf hindeutet, dass vorhergesagte Zahlungsausfälle häufiger korrekt sind als beim XGBoost-Modell. Während der Recall für Klasse 0 mit 0.98 nahezu perfekt ist, liegt er für Klasse 1 mit nur 0.22 auf einem sehr niedrigen Niveau. Dies bedeutet, dass lediglich 22 % der tatsächlichen Zahlungsausfälle erkannt werden. Der F1-Score für Klasse 0 beträgt 0.89, während er für Klasse 1 lediglich 0.33 erreicht, was die unausgewogene Leistung des Modells verdeutlicht. Mit einer Gesamtgenauigkeit von 80 % liegt das Modell auf einem ähnlichen Niveau wie XGBoost, jedoch werden deutlich weniger Zahlungsausfälle erkannt.

7 Fazit und Verbesserungspotenzial

Die Analyse zeigt, dass sowohl XGBoost als auch die logistische Regression spezifische Stärken und Schwächen aufweisen. Während XGBoost eine höhere Erkennungsrate für Zahlungsausfälle bietet, zeichnet sich die logistische Regression durch eine bessere Präzision aus. Die Herausforderung besteht darin, ein Modell zu wählen, das nicht nur eine hohe Gesamtgenauigkeit besitzt, sondern auch zuverlässig tatsächliche Zahlungsausfälle identifiziert, ohne zu viele Fehldiagnosen zu produzieren.
Für die praktische Anwendung in der Kreditrisikobewertung ist es entscheidend, eine ausgewogene Strategie zu verfolgen, die sowohl Vorhersagegenauigkeit als auch Interpretierbarkeit berücksichtigt. Die Kombination beider Modelle oder der Einsatz von Ensemble-Methoden könnte helfen, deren jeweilige Schwächen zu kompensieren. Weitere Optimierungen, beispielsweise durch eine feinere Anpassung der Hyperparameter oder eine bessere Berücksichtigung der Klassenverteilung, könnten zusätzlich die Modellleistung verbessern. Ein hybrider Modellansatz könnte die Stärken beider Methoden vereinen, während eine tiefere Analyse der SHAP-Werte fundiertere Kreditrisikobewertungen ermöglicht. Die Wahl des Modells erfordert stets eine Abwägung zwischen Genauigkeit, Interpretierbarkeit und Anwendbarkeit.
Ein vielversprechender Ansatz wäre der Einsatz von Reinforcement Learning, das durch kontinuierliches Lernen aus Echtzeitdaten adaptive Entscheidungen ermöglichen kann. Dies könnte insbesondere in dynamischen Finanzumfeldern von Vorteil sein, in denen sich das Zahlungsverhalten der Kunden über die Zeit verändert und traditionelle Modelle an ihre Grenzen stoßen.

Eigenschaft	Wert
Datensatzart	Multivariat
Fachbereich	Business
Zugehörige Aufgabe	Klassifikation
Anzahl der Instanzen	30.000
Anzahl der Merkmale (Features)	23
Zielvariable (Y)	Zahlungsverzug: 1 = Ja, 0 = Nein
Merkmalsarten	Ganzzahlen, Reelle Zahlen
Fehlende Werte	Nein
Datenaufteilung	Split Training: 75%, Test: 25%
Target Variable	Rückzahlungsstatus
Kreditbetrag (X1)	Höhe des gewährten Kredits (in NT-Dollar), einschließlich individueller Verbraucherkredite sowie Familienkredite
Geschlecht (X2)	1 = männlich, 2 = weiblich
Bildungsniveau (X3)	1 = Hochschulabschluss, 2 = Universität, 3 = Gymnasium, 4 = Andere
Familienstand (X4)	1 = verheiratet, 2 = ledig, 3 = andere
Alter (X5)	Alter der Person in Jahren
Vergangene Zahlungshistorie (X6 - X11)	Monatliche Zahlungsinformationen von April bis September 2005. Zahlungsstatus-Skala: -1 = rechtzeitig bezahlt, 1 = Zahlungsverzug für einen Monat, bis 9 = Zahlungsverzug von neun Monaten oder mehr.
Monatliche Abrechnungen (X12 - X17)	Höhe der Rechnungsstellung für die Monate April bis September (in NT-Dollar).
Frühere Zahlungen (X18 - X23)	Höhe der geleisteten Zahlungen für die Monate April bis September 2005 (in NT-Dollar).

Tabelle 2: Zusammenfassung Variablen des UCI Credit Card Datensatzes [6]

Literatur

[1] **Analytics Vidhya Content Team**: *An End-to-End Guide to Understand the Math behind XGBoost*, *analyticsvidhya.com*, 2018. https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

[2] **Butch Quinto**: *Next-Generation Machine Learning with Spark, Covers XGBoost, LightGBM, Spark NLP, Distributed Deep Learning with Keras, and More, 2020*, ISBN-13 (eBook): 978-1-4842-5669-5. https://doi.org/10.1007/978-1-4842-5669-5

[3] **Delinquency Rate**: https://blog.itreconomics.com/blog/us-consumers-posting-record-high-credit-card-debt-l

[4] **Fung, Clement et al.**: *"Dancing in the Dark: Private Multi-Party Machine Learning in an Untrusted Setting." ArXiv abs/1811.09712 (2018)*: n. pag. https://github.com/DistributedML/TorML?tab=readme-ov-file

[5] **Islam, Sheikh Rabiul et al.**: *"Credit Default Mining Using Combined Machine Learning and Heuristic Approach."* https://arxiv.org/abs/1807.01176

[6] **I-Cheng, Yeh**: *UCI Credit Card Default of Clients [Dataset]. UCI Machine Learning Repository; 2009* https://doi.org/10.24432/C55S3H
erhoben am 01/25/2016

[7] **Nayebi Hooshang**: *Advanced Statistics for Testing, Assumed Causal Relationships Multiple Regression Analysis; (eBook): 978-3-030-54754-7; Published: 15 August 2020*
https://doi.org/10.1007/978-3-030-54754-7

[8] **Philip Hyunsoo Cho**: *Fast Histogram Optimized Grower*, 8x to 10x Speedup, *DMLC*, 2017. https://github.com/dmlc/xgboost/issues/1950

[9] **Reena Shaw**: *XGBoost: A Concise Technical Overview*, *KDNuggets*, 2017. https://www.kdnuggets.com/2017/10/xgboost-concise-technical-overview.html

[10] **Stefan Richter**: *Statistisches und maschinelles Lernen;*
(eBook) ISBN: 978-3-662-59354-7; Published: 29 August 2019 https://doi.org/10.1007/978-3-662-59354-7

[11] **Tshepo Chris Nokeri**: *Data Science Solutions with Python: Fast and Scalable Models Using Keras, PySpark MLlib, H2O, XGBoost, and Scikit-Learn*, ISBN-13 (eBook): 978-1-4842-7762-1. https://doi.org/10.1007/978-1-4842-7762-1

[12] : **XGBoost**; *"Introduction to Boosted Trees," xgboost.readthedocs. io, 2019*, https://xgboost.readthedocs.io/en/latest/tutorials/model.html

[13] **Wolfgang Karl Härdle Xiaorui ZUO**: *DEDA Digital Economy & Decision Analytics Introduction to ML, Ladislaus von Bortkiewicz Professor of Statistics Humboldt-Universität zu Berlin* theIDA.net

Eidesstattliche Erklärung

Ich versichere hiermit, dass ich die vorliegende Hausarbeit selbstständig und ohne fremde Hilfe angefertigt und keine anderen als die angegebenen Quellen benutzt habe. Alle von anderen Autoren wörtlich übernommenen Stellen, wie auch die sich an die Gedankengänge anderer Autoren eng anlehnenden Ausführungen meiner Arbeit, sind besonders gekennzeichnet. Diese Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Berlin, 17. Februar 2025

Zaviera Manin Tiola