SEMINARARBEIT

Machine Learning für Finanzanwendungen

basiert auf
„An Introduction to Machine Learning in Quantitative Finance“
by Hao Ni, Xin Dong, Jinsong Zheng and Guangxi Yu

unter der Anleitung von

Dipl.-Ing. Dr.techn. Stefan Gerhold

durch

Erik Rakhmatulin
Matrikelnummer: 11817729

Wien,
3. April 2022

Inhaltsverzeichnis

1 Einleitung

Diese Seminararbeit basiert auf der Arbeit „An Introduction to Machine Learning in Quantitative Finance“ von Hao Ni and Xin Dong, Jinsong Zheng, and Guangxi Yu [1].

Die Authoren dieses Buches beantworten die folgenden Fragen:

- Was ist Machine Learning?
- Welche Rolle und Bedeutung hat es im heutigen Finanzwesen?

Es werden die konkreten praktischen Beispiele angegeben. Einer der wichtigsten Teile dieser Arbeit sind die Algorithmen von Machine Learning und ihre diversen Arten. In meiner Arbeit werde ich mich auf den Algorithmus „Supervised Learning“ sowie seine Art „Linear Regression“ konzentrieren.

2 Anwendung von Machine Learning im Finanzbereich

2.1 Was ist Machine Learning?

1959 definierte Arthur Samuel Machine Learning als ein Studiengebiet, das Computern die Fähigkeit verleiht, zu lernen, ohne explizit programmiert zu werden [2]. Manchmal ist es schwierig, bestimmte Aufgaben durch explizite Programmieranweisungen zu erfüllen. In diesem Fall besteht der beste Ansatz darin, Computern die Möglichkeit zu geben, aus den Daten zu lernen.

Wie wir auf der Abbildung 2.1 sehen können, ist die Bilderkennung ein Beispiel für eine maschinelle Lernaufgabe. Bei dieser Aufgabe ist es schwierig, dem Computer durch explizite Programmieranweisungen zu sagen, wie er Blumen in Gemälden erkennen soll, da die Position, Form und Farbe der Blumen in jedem Gemälde unterschiedlich sind. Aber der maschinelle Lernalgorithmus kann automatisch lernen, welche Blume sich auf den Gemälden befindet, wenn wir den Algorithmus mit genügend Daten von Gemälden und den entsprechenden Blumenetiketten füttern. Derselbe Algorithmus kann angewendet werden, um eine menschliche Pose in Fotografien zu identifizieren. Hier sieht man, dass Algorithmen des maschinellen Lernens auf verschiedene Aufgaben angewendet werden können und nicht explizit für jede spezifische Aufgabe programmiert werden müssen.

Abbildung 2.1: Bildklassifizierung im Machine Learning

Algorithmen für Machine Learning sind datengesteuert und werden betrieben, indem ein Modell aus Beobachtungen von Daten erstellt wird, um eine unsichtbare Situation ohne viel menschliches Eingreifen vorherzusagen. Sie eignen sich gut für die Komplexität des Umgangs mit unterschiedlichen Datenquellen. Im Gegensatz zur traditionellen Analyse lebt maschinelles Lernen von wachsenden Datensätzen. Je mehr Daten in ein maschinelles Lernsystem eingespeist werden, desto mehr kann es lernen und desto bessere Ergebnisse kann es erzielen. Daher ist maschinelles Lernen ideal, um die Herausforderungen anzugehen und die Chancen von Big Data zu nutzen.

Das Kernziel eines Algorithmus von Machine Learning ist die Verallgemeinerung seiner Erfahrung. In diesem Zusammenhang ist Generalisierung die Fähigkeit einer lernenden Maschine, auf der Grundlage der Beobachtung eines Lerndatensatzes mit unsichtbaren Daten genau zu arbeiten. Es ist wichtig zu beachten, dass ein ausgeklügelter Algorithmus für maschinelles Lernen eine hohe Genauigkeit im Trainingssatz erzielen kann. Dies garantiert jedoch aufgrund des potenziellen Risikos des Overfitting-Problems keine hohe Vorhersagegenauigkeit für unsichtbare Daten. Das Overfitting-Phänomen wird normalerweise durch die Fehlinterpretation von Rauschen als Signal durch überentwickelte Algorithmen verursacht, was zu einer schlechten Vorhersagekraft führt [3]. Um geeignete Algorithmen für Machine Learning zu erstellen, müssen wir daher sowohl die Modellkomplexität als auch die Datengröße berücksichtigen.

Algorithmen von Machine Learning können hauptsächlich in Supervised Learning, Unsupervised Learning und Reinforcement Learning unterteilt werden:

- **Supervised Learning** zielt darauf ab, aus einem gekennzeichneten Trainingssatz zu lernen, sodass das resultierende Modell effektiv auf unsichtbare Daten angewendet werden kann. Ausgehend von der Analyse eines gegebenen Trainingsdatensatzes, bei dem es sich um eine Sammlung von Eingabe-Ausgabe-Paaren handelt, erzeugt der überwachte Lernalgorithmus eine abgeleitete Funktion, um Vorhersagen über die Ausgabewerte zu treffen [4].
- **Unsupervised Learning** untersucht, wie Systeme eine Funktion zur Beschreibung einer verborgenen Struktur aus unbeschrifteten Daten ableiten können.
- **Reinforcement Learning (RL)** ist eine Lernmethode für die Identifizierung der optimalen Aktionen, die mit ihrer Umgebung interagieren, indem sie die entsprechenden Belohnungen maximieren. Diese Methode ermöglicht es Maschinen und Software-Agenten, das ideale Verhalten innerhalb eines bestimmten Kontexts basierend auf einer Feedback-Schleife aus der Umgebung automatisch zu bestimmen.

Im Gegensatz zum supervised Learning werden unsupervised Learning Algorithmen verwendet, wenn die Daten weder klassifiziert noch gekennzeichnet sind. Im Vergleich zu supervised Learning und unsupervised Learning unterscheidet sich RL in Bezug auf die Ziele. Während das Ziel beim supervised Learning darin besteht, Ähnlichkeiten und Unterschiede zwischen Datenpunkten zu finden, besteht das Ziel beim Reinforcement Learning darin, ein geeignetes Aktionsmodell zu finden, das die Belohnung des Agenten maximiert. Supervised Learning zielt jedoch darauf ab, die Zuordnung zwischen der Eingabe und der gekennzeichneten Ausgabe aus den Daten zu lernen.

Lassen Sie uns auf das Handelsbeispiel zurückkommen, in dem Sie vielleicht den Unterschied zwischen RL und supervised Learning sehen. Das Aufdecken des Musters zwischen Marktinformationen und zukünftigen Preisbewegungen ist ein supervised Lernproblem. Im Gegensatz dazu geht es beim optimalen Handelsproblem darum, die beste Handelsstrategie (Aktion) zu finden, um den endgültigen Wohlstand (Belohnung) zu maximieren, was in die Kategorie des Reinforcement Learning fällt.

Die jüngsten erfolgreichen Anwendungen von Machine Learning wurden hauptsächlich auf die Weiterentwicklung der Hardwaretechnologie zurückgeführt, die die Speicherung, Verarbeitung und Analyse von Big Data durch maschinelle Lernalgorithmen in der Praxis möglich gemacht hat. Die Erfolge bei maschinellen Lernanwendungen konzentrieren sich hauptsächlich auf supervised Learning und Reinforcement Learning. Hier zählen wir einige Beispiele für Anwendungen von Machine Learning auf. Einer der beliebtesten maschinellen Lernalgorithmen, das sogenannte Deep Learning, hat bei verschiedenen Aufgaben, wie der Bilderkennung und der Spracherkennung, state-of-the-art Ergebnisse erzielt.

2.2 Quantitative Finanzen

2.2.1 Herausforderungen von Finanzdaten

Mit dem Aufkommen der Big Data Ära haben wir eine dramatische Zunahme der verfügbaren Echtzeitfinanzmarktdatensätze erlebt, von Online-Transaktionsaufzeichnungen bis hin zu Hochfrequenz-Limit-Orderbüchern (eine Aufzeichnung ausstehender Kauf- oder Verkaufsaufträge). Das Extrahieren von Informationen aus diesen Finanzdatenströmen mithilfe von maschinellen Lerntechniken ist aufgrund ihres niedrigen Signal-Rausch-Verhältnisses und ihrer komplexen Multimodalität eine Herausforderung. Diese Probleme können die Fehlinterpretation von Rauschen in den Daten als Signale verursachen und somit zu potenziellen finanziellen Verlusten und sogar Finanzkrisen führen [5].

Außerdem ist eine zunehmende Menge an unstrukturierten Daten – z. B. Finanznachrichten, Satellitenbilder für Erdbeobachtungsdaten oder Chats in Investitionsforen – potenziell wertvoll, um nützliche Informationen für Finanzdienstleister bereitzustellen. Die traditionelle statistische Analyse kann diese unstrukturierten Daten, die auch als alternative Daten bezeichnet werden, jedoch nicht verarbeiten und analysieren.

Darüber hinaus können einige Finanzdaten in Bezug auf die Datenverfügbarkeit eingeschränkt sein – z. B. existieren einige Finanzinstrumente nur für einen kurzen Zeitraum, was zu unzureichenden Daten für ausgefeilte maschinelle Lerntechniken führen kann. Nicht zuletzt sind Finanzdaten möglicherweise nicht stationär, was bedeutet, dass sie Regimeänderungen unterliegen und ältere Daten für Vorhersagen weniger relevant werden können [Sirignano and Cont (2018)].

2.2.2 Aktuelle Entwicklung von Machine Learning für Finanzanwendungen

In den letzten Jahren haben immer mehr Finanzunternehmen damit begonnen, fortschrittliche Techniken von Machine Learning einzusetzen, um sich einen Wettbewerbsvorteil zu verschaffen. Immer mehr Hedgefonds haben sich von traditionellen Analysemethoden entfernt und Algorithmen von Machine Learning zur Vorhersage von Fondstrends und zur Portfolioauswahl [6] eingeführt. Ein Artikel mit dem Titel „Seven Ways Fintechs Use Machine Learning to Outsmart the Competition“ erläutert verschiedene erfolgreiche Anwendungen des maschinellen Lernens im Fintech-Sektor zur Verbesserung der Finanzanalyse [7].

Ein perfektes Beispiel dafür ist COIN von JPMorgan Chase, was für Contract Intelligence steht. Rund 360.000 Stunden benötigt ein Mensch, um 12.000 Handelskreditverträge zu prüfen, während der COIN juristische Dokumente und Verträge mit Bilderkennungssoftware analysiert und seine Aufgabe in Sekundenschnelle [8] erledigt.

Da die Menge der verfügbaren Daten und der Zugriff darauf gewachsen sind, wurde die Nutzung von maschinellem Lernen, um bessere Investitions- oder andere Geschäftsentscheidungen zu treffen, in verschiedenen Anwendungen im Finanzwesen validiert. Dies stärkt das Vertrauen der Menschen in das große Potenzial des fortschrittlichen maschinellen Lernens im Finanzbereich. Machine Learning bietet ein effizientes Werkzeug zur Analyse von Daten, die von individuellen Daten (z. B. Stimmung aus Nachrichten, Twitter) bis hin zu Geschäftsprozessen (z. B. Kredithistorie) reichen.

2.2.3 Die Zukunft von den quantitativen Finanzen

In Zukunft ist der Trend zur Einführung von Machine Learning in der quantitativen Finanzierung unvermeidlich. Interpretierbarkeit, Datenfusion und Hardwaretechnologie sind die dominierenden Trends von Machine Learning für die nachhaltige Weiterentwicklung der quantitativen Finanzierung.

Die Interpretierbarkeit von Black-Box-Lernalgorithmen ist ein aufstrebendes Forschungsgebiet. Es soll Menschen ohne technischen Hintergrund ermöglichen, mehr Vertrauen in Black-Box-Algorithmen zu haben und die Ergebnisse der Algorithmen mit ihrem Domänenwissen zu überprüfen. Zhang und Zhu (2018) bieten einen Überblick über aktuelle Studien zum Verständnis der Darstellungen von neuronalen Netzen, die zu den beliebtesten Algorithmen gehören. Obwohl (tiefe) neuronale Netze eine überlegene Leistung in Bezug auf die Vorhersagegenauigkeit gezeigt haben Bei verschiedenen Aufgaben kann die Interpretierbarkeit ihr schwächster Punkt sein. Gunning (2017) bietet mehrere Folien, um die jüngste Entwicklung des erklärbaren maschinellen Lernens zusammenzufassen, einschließlich der Visualisierung und Interpretierbarkeit von maschinellen Lernalgorithmen. Die Erklärbarkeit von maschinellen Lernalgorithmen ist in Finanzanwendungen sehr wichtig. Der Aufbau von Vertrauen in maschinelle Lernalgorithmen für Investoren und Aufsichtsbehörden ist ein Muss für den weit verbreiteten Einsatz von maschinellem Lernen im Finanzwesen. Erklärbares maschinelles Lernen spielt eine entscheidende Rolle beim Aufbau dieses Vertrauens.

Der zweite wichtige Aspekt der zukünftigen maschinellen Lernforschung im Bereich der quantitativen Finanzen ist die Entwicklung effizienter Algorithmen für die Datenfusion. Finanzbezogene Daten können unterschiedlicher Art sein, z. B. Nachrichtendaten, Transaktionshistorie usw. Es ist entscheidend zu untersuchen, wie nützliche Informationen aus verschiedenen Arten von Finanzdaten extrahiert werden können.

Schließlich kann das Machine Learning ohne die Entwicklung von Hardware und Hochleistungsrechnen nicht weiter voranschreiten. Die meisten Deep-Learning-Algorithmen wurden vor Jahrzehnten vorgeschlagen, aber ihre erfolgreichen Anwendungen sind hauptsächlich in den letzten Jahren entstanden. Dies liegt daran, dass die Computertechnologie zum Zeitpunkt der Einführung dieser Algorithmen kaum bereit war, die Leistungsfähigkeit von Deep-Learning-Algorithmen zu realisieren. Mit Blick auf die Zukunft wachsen die Datenmengen

exponentiell und die Algorithmen werden immer ausgefeilter, was eine umfangreiche Computertechnologie erfordert, um den Anforderungen gerecht zu werden. Verteilte Systeme, Quantencomputer usw. sind vielversprechende Forschungsgebiete mit großem Potenzial und großer Wirkung.

3 Supervised Learning

Supervised Learning ist die maschinelle Lernaufgabe, eine Funktion abzuleiten, die eine Eingabe einer Ausgabe zuordnet, basierend auf beispielhaften Eingabe-Ausgabe-Paaren. Abhängig davon, ob die Ausgabe eine kontinuierliche Variable oder eine kategoriale Variable ist, kann das überwachte Lernen weiter in zwei Arten unterteilt werden:

- **Regression**;
- **Classification**.

Im Abschnitt 3.1 konzentrieren wir uns zunächst auf Regressionsprobleme. Ein allgemeiner Regressionsrahmen umfasst das Modell, die Verlustfunktion, die Optimierung, die Vorhersage und die Validierung. Jede Komponente des Regressions-Frameworks wird in den folgenden Abschnitten ausführlich besprochen. Im Abschnitt 3.2 erklären wir, wie man von der Regression zur Klassifikation kommt. Schließlich besprechen wir, wie ein Ensemble aus mehreren Modellen verwendet werden kann, um die Leistung des überwachten Lernens zu verbessern.

3.1 Die Struktur von Regression

Lassen Sie uns den Standardaufbau des Regressionsproblems einführen. Zur Konkretheit betrachten wir den Fall einer skalaren Ausgabe. Angenommen, es gibt ein Dataset $D = \{(x_i, y_i)\}_{i=1}^N$, wobei (x_i, y_i) das i_{th} input–output Paar (auch genannt die i_{th} Probe) bezeichnet. Jede input Probe x_i ist ein d-dimensionaler Vector, d.h. $x_i := (x_i^{(1)}, ..., x_i^{(d)}) \in \mathbb{R}^d$, . Angenommen, es gibt $f : \mathbb{R}^d \longrightarrow \mathbb{R}$, sodass:

$$y_i = f(x_i) + \epsilon_i, \tag{3.1}$$

wobei $y_i \in \mathbb{R}$ und ϵ_i unabhängig und identisch verteilt Zufallsvariablen sind mit $\mathbb{E}[\epsilon_i | x_i] = 0$. Nehmen Sie für die Regularitätsannahme von f an, dass f eine stetige Funktion ist. Zur Vereinfachung der Notation übernehmen wir auch die Matrixform für $D = (X, Y)$, wobei

$$X = \begin{pmatrix} x_1^{(1)}, & x_1^{(2)}, & \dots, & x_1^{(d)} \\ \vdots & \vdots & & \vdots \\ x_N^{(1)}, & x_N^{(2)}, & \dots, & x_N^{(d)} \end{pmatrix} \text{ und } Y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}, \tag{3.2}$$

wobei X eine N Matrix ist und Y ein $N \times 1$ Vektor ist.

Die erste Frage, die wir uns stellen, ist, wie man die entsprechende Ausgabe für jede gegebene neue Eingabe x^* schätzt. Im Kontext des Regressionsproblems besteht eine strenge mathematische Formulierung dieser Frage darin, $\mathbb{E}[y | x = x^*]$ zu schätzen, d.h. $f(x^*)$ für jedes neu eingegebene Data x^*, was der Schätzung von f entspricht. Daher wird f auch Mittelwertfunktion des Regressionsproblems genannt.

Die nächste wichtige Frage ist, wie man den besten Schätzer für f aus verschiedenen möglichen Schätzern auswählt, was darauf hinausläuft, was „besser“ bedeutet, und wie man die Leistung jedes Schätzers quantifiziert.

Im Folgenden erläutern wir die Herangehensweise an die beiden oben genannten Fragen und fassen dies als allgemeinen Rahmen für die Regression zusammen. Erinnern Sie sich, dass das Ziel des Regressionsproblems darin besteht, die feste, aber unbekannte Mittelwertfunktion f aus dem gekennzeichneten Datensatz D zu lernen, sodass die Gleichung (3.1) gilt. Ein natürlicher Schritt besteht darin, das Modell f_θ zu postulieren, um die unbekannte Mittelwertfunktion f zu beschreiben, wobei θ die Modellparameter sind, die das Modell f_θ vollständig charakterisieren. Auf diese Weise wird das Problem, f zu finden, in die Suche nach den besten Parametern θ übersetzt, die zu den Daten passen.

Um die besten Parameter θ zu finden, müssen wir quantifizieren, was wir mit „den besten Parametern“ meinen. Dadurch motiviert, schlagen wir die Verlustfunktion vor, um die Diskrepanz zwischen dem vom Modell geschätzten Output $f_\theta(x)$ und dem tatsächlichen Output y zu quantifizieren. Nach Auswahl der Verlustfunktion $L(\theta|D)$ wird der optimale Parametersatz θ^* als derjenige definiert, der die Verlustfunktion minimiert. In den meisten Fällen gibt es keine geschlossene Formel für den optimalen Parametersatz θ^*, und wir müssen die numerische Optimierungsmethode verwenden. Egal wie wir den Schätzer der optimalen Parameter θ^* erhalten, entweder durch geschlossene Formel oder numerische Methoden, sobald wir θ^* haben, sind wir bereit, Vorhersagen zu treffen. Genauer gesagt, für jede neue Eingabe x_* ist der Schätzer der bedingten Erwartung der Ausgabe $[y_*|x_*]$ gegeben durch $f^*_\theta(x_*)$. Schließlich müssen wir die Güte der Anpassung quantifizieren, indem wir die Metriken angeben, z. B. mittlerer quadratischer Fehler (MSE), R-Quadrat (R^2). Diese Metriken sind möglicherweise nicht mit denen identisch, die in der Verlustfunktion verwendet werden.

Dataset:	$D = (x_i, y_i)_{i=1}^N$
Model:	$f_\theta(x) \approx \mathbb{E}[y\|x] = f(x), \forall x \in \mathbb{R}^d$
Empirischer Verlust:	$L(\theta\|D) = \frac{1}{N}\sum_{i=1}^{N} d(f_\theta(x_i), y_i) \rightarrow$ Minimieren
Optimierung:	$\theta^* = argmin_\theta(L(\theta\|D))$
Prognose:	$\hat{y}_* = f_{\theta^*}(x_*)$
Validierung:	Berechnen Sie die Indikatoren für die Anpassungsgüte

Tabelle 3.1: Die Struktur von Regression.

Tabelle 3.1 fasst den gesamten Prozess zusammen, den wir oben beschrieben haben. Datensatz, Modell, empirischer Verlust, Optimierung, Prognose und Validierung sind die Schlüsselelemente von Supervised Learning. Wir folgen diesem allgemeinen Rahmen, um in den folgenden Kapiteln mehrere überwachte Lernalgorithmen vorzustellen, und fassen jeden Algorithmus im Rahmenkasten zusammen.

3.2 Modell

In diesem Unterabschnitt stellen wir verschiedene Arten von Modellen vor, die von linearen Modellen bis hin zu nichtlinearen Modellen reichen, und erläutern die Grundidee hinter den meisten nichtlinearen Modellen – die sogenannte Basiserweiterung. Bei der Regression ist das vorgeschlagene Modell eine Familie parametrischer Funktionen, sagen wir f_θ, wobei θ den Parametersatz bezeichnet, der das Modell vollständig charakterisiert. Der Einfachheit halber konzentrieren wir uns auf den Fall der eindimensionalen Ausgabe.

Beginnen wir mit dem einfachsten Modell - dem linearen Modell - wo wir annehmen, dass $f_\theta : \mathbb{R}^d \to \mathbb{R}$ eine lineare Funktion ist, also $\forall x = (x^{(1)}, x^{(2)}, ..., x^{(d)}) \in \mathbb{R}^d$,

$$f_\theta(x) = \theta^T x = \sum_{j=1}^{d} \theta^{(j)} x^{(j)},$$

wobei $\theta = (\theta^{(1)}, ..., \theta^{(d)}) \in \mathbb{R}^d$ der Parametersatz ist. Dies ist das Modell, das von linearen Regressionsmethoden übernommen wird.

Lineare Modelle sind jedoch möglicherweise nicht reichhaltig genug, um die komplexe funktionale Beziehung zwischen dem Input und dem Output zu beschreiben. Dadurch motiviert, gibt es verschiedene Arten von nichtlinearen Modellen. Wir listen einige beliebte nichtlineare Modelle wie folgt auf, aber die Liste ist nicht vollständig.

- **Polynomiales Modell**, d.h.,

  $$f_\theta(x) = x\mu + x^T,$$

  wobei $\theta = (\mu, \sum)$, und $\mu \in \mathbb{R}^d$ und $\sum \in \mathbb{R}^d \times \mathbb{R}^d$;

- **Spline-Model**, d.h.,

  $$f_\theta(x) = {}^{M}_{i=1} C_i (x - l_i)^+,$$

  wobei $\theta = (l_i, C_i)_{i=1}^M$ die Parameter des Modells sind;

- **Regression Tree Modell**:

  $$f(x) = \sum_{m=1}^{M} c_m \Pi(x \in R_m),$$

  wobei $R_1, R_2, ..., R_M$ ist eine Partition des Eingaberaums mit M disjunkten Regionen. Das Baummodell ermöglicht die Aufteilung des Eingaberaums durch Aufteilen von Variablen und Punkten, was mit der Topologie übereinstimmt, die ein Baum haben sollte (z. B. Abb. 3.1);

- **Neural Network Modell**: neuronale Netzwerkmodelle basieren auf einer Sammlung verbundener Neuronen (Knoten). Es gibt verschiedene Arten, die in der Abb. 3.2 dargestellt sind.

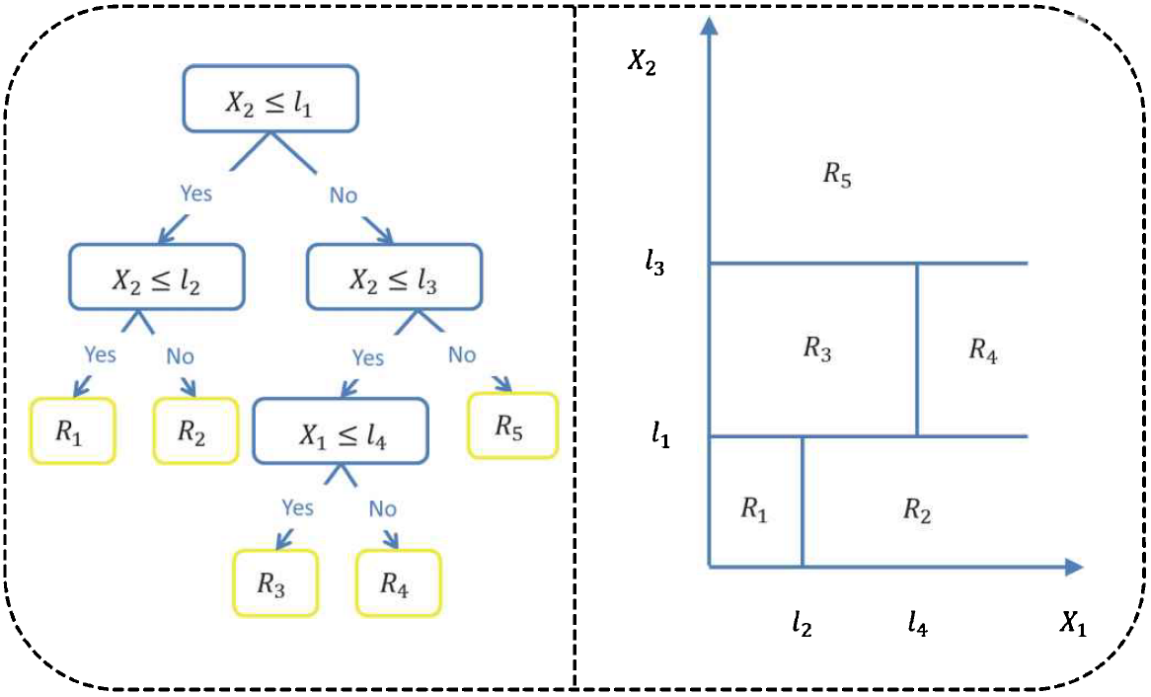


Abbildung 3.1: Ein Beispiel von Tree Modell.

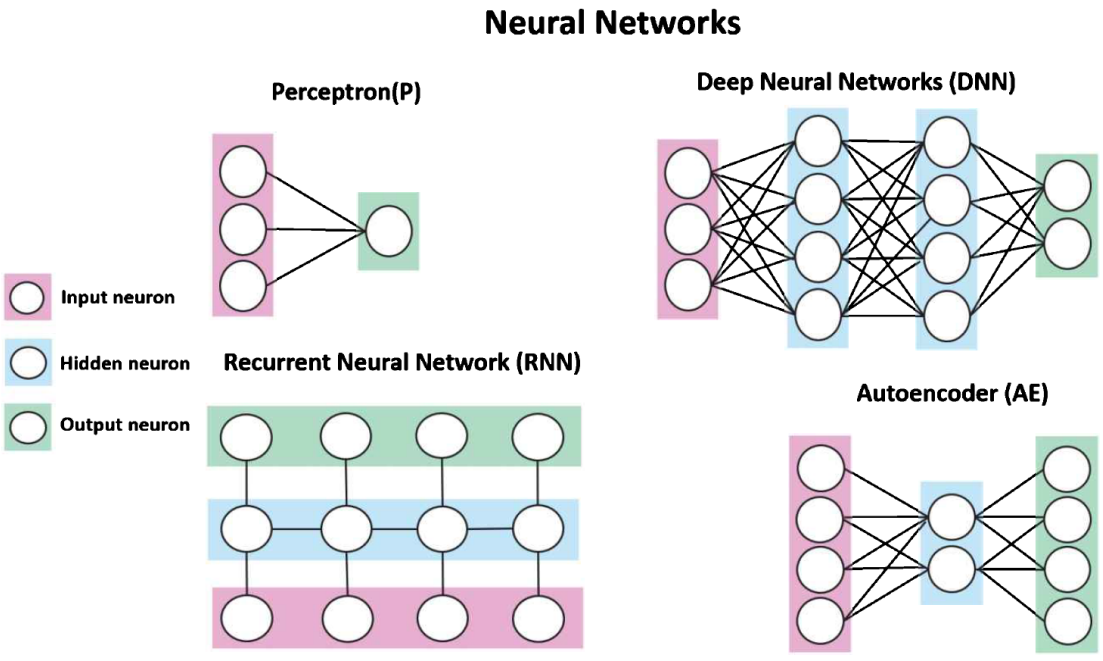


Abbildung 3.2: Beispiele für die Haupttypen neuronaler Netzwerkarchitekturen.

3.3 Verlustfunktion

In der Statistik wird die Verlustfunktion (auch Kostenfunktion genannt) vorgeschlagen, um die Differenz zwischen geschätzten und tatsächlichen Werten für Ausgangsdaten zu quantifizieren. Sie dient als Nutzfunktion zur Parameterschätzung. Die Verlustfunktion ist ein Maß für Parameter. Je kleiner der Wert der Verlustfunktion ist, der anzeigt, dass die geschätzte Ausgabe näher an der tatsächlichen Ausgabe liegt, desto besser ist der Parameter.

Das Konzept der Verlustfunktion stellt den Preis dar, der für die Ungenauigkeit von Vorhersagen bei Lernproblemen gezahlt wird. Eine der am häufigsten verwendeten Verlustfunktionen in der Regression ist die quadratische Verlustfunktion, die als quadratischer Fehler zwischen dem geschätzten Ergebnis des Modells und dem tatsächlichen Ergebnis definiert ist (siehe Definition 3.3.1).

Definition 3.3.1 (Quadratische Verlustfunktion)**.** Sei f_θ die vollständig charakterisierten Modellparameter θ. Die quadratische Verlustfunktion ist definiert als $\forall(x, y) \in E \times \mathbb{R}$,

$$Q_\theta(x, y) = (y - f_\theta(x))^2.$$

Wir können die Verlustfunktion für jede Probe auswerten. Die Mittelung der Verlustfunktion aller Stichproben führt zum empirischen Risiko, das den durchschnittlichen Verlust über den gesamten Datensatz bezeichnet.

Definition 3.3.2 (Empirisches Risiko)**.** Q_θ bezeichne eine Verlustfunktion, wobei θ ein Modellparametersatz ist. Dann ist das mit L bezeichnete empirische Risiko wie folgt definiert:

$$L(\theta|D) = \frac{1}{N} \sum_{i=1}^{N} Q_\theta(x_i, y_i),$$

wobei $D = (x_i, y_i)_{i=1}^N$.

Im Folgenden bezeichnen wir das empirische Risiko oft als Verlustfunktion.

4 Lineare Regression

In diesem Kapitel konzentrieren wir uns auf die lineare Regression und erklären, wie sie in den allgemeinen Rahmen des überwachten Lernens passt. Wir beginnen mit dem einfachsten linearen Modell – Ordinary Least Squares (OLS) – und diskutieren dann die natürliche Erweiterung von OLS, d. h. das lineare Modell mit Regularisierung.

4.1 Ordinary Least Squares Methode

4.1.1 Derivation

Ordinary Least Squares (OLS) ist das einfachste lineare Modell in der Regression und wurde in vielen verschiedenen Bereichen, z. B. Wirtschaft, Politikwissenschaft und Ingenieurwesen, weit verbreitet. Wir folgen dem allgemeinen Aufbau des Regressionsproblems in Abschnitt 3.1. OLS geht davon aus, dass das Modell zwischen der Eingabe $x \in \mathbb{R}^d$ und der Ausgabe $y \in \mathbb{R}$ linear und als Formel vorliegt,

$$y = x\theta + \epsilon, \tag{4.1}$$

wobei x ein d-dimensionaler Zeilenvektor ist, θ ein d-dimensionaler Spaltenvektor ist und ϵ ein skalarer Rauschterm ist, wobei der bedingte Mittelwert für die Eingabe x Null ist. θ repräsentiert die festen, aber unbekannten linearen Koeffizienten des OLS-Modells. Wie der Name OLS schon sagt, ist die Verlustfunktion von OLS die Summe der quadrierten Residuen

$$L(\theta|X,Y) = \sum_{i=1}^{N}(y_i - x_i\theta)^2 = (Y - X\theta)^T(Y - X\theta),$$

wobei $(x_i, y_i)_{i=1}^N$ eine Sammlung der Input-Output-Paare ist und (X, Y) wie in der Gleichung 3.2 definiert ist.

Anmerkung 4.1. Für den Fall, dass das lineare Modell einen Schnittpunkt ungleich Null enthält, d. h.,

$$y = \theta_0 + x\theta + \epsilon, \tag{4.2}$$

wir formulieren es als die Eingabe $\tilde{x} = (1, x)$ um, indem wir die Konstante 1 zu einer zusätzlichen Koordinate der Eingabevariablen x hinzufügen, und dann kann Gleichung (4.2) umgeschrieben werden als

$$y = \tilde{x}\tilde{\theta} + \epsilon, \tag{4.3}$$

wobei $\tilde{\theta} = \binom{\theta_0}{\theta}$.

Die Verlustfunktion kann durch die Annahme der Normalverteilung der Residuen mit dem Maximum-Log-Likelihood-Schätzer des linearen Modells verbunden werden. Beachten Sie, dass, wenn man identische und unabhängige Verteilungen der Stichproben-Rauschreste $(\epsilon_i)_{i=1}^N$ mit einem bedingten Mittelwert von Null und einer konstanten bedingten Varianz

annimmt, die Minimierung der Verlustfunktion von OLS gleichbedeutend mit der Maximierung des Log- Wahrscheinlichkeitsverhältnis der Beobachtung der Ausgabe $(y_i)_{i=1}^N$ bedingt durch $(x_i)_{i=1}^N$.

Ein großer Vorteil von OLS ist, dass es eine analytische Formel für den optimalen Parameterschätzer für θ liefert, bezeichnet mit $\hat{\theta}$. Durch Matrizenrechnung erhält man leicht die folgenden Formeln für $\hat{\theta}$ (Gleichung (4.4) in Lemma 4.1).

Lemma 4.1 (OLS-Schätzer). Bei gegebenem Input-Output-Datensatz $D = (X, Y)$ und unter der Annahme, dass die Standardeinstellung des OLS-Modells gilt und die Existenz des Inversen von $X^T X$, ist der Schätzer des optimalen Parameters $\hat{\theta}$ ist wie folgt gegeben:

$$\hat{\theta} = (X^T X)^{-1} X^T Y. \tag{4.4}$$

Jetzt verstehen wir, wie man den optimalen Parameter $\hat{\theta}$ erhält, und in der Praxis verwenden wir einfach direkt Gleichung 4.4, um die optimalen Parameter aus den Trainingsdaten D zu berechnen. Der letzte Schritt besteht darin, den Indikator für die Anpassungsgüte auszuwählen und diese Metrik sowohl für Trainings- als auch für Testsätze zu bewerten. Die übliche Wahl für die Anpassungsgüte in OLS sind der mittlere quadratische Fehler (RMSE), R^2 und der angepasste R^2.

Zusammenfassend sind die Schlüsselelemente von OLS in der Tabelle 4.1 angegeben.

Dataset:	$D = (x_i, y_i)_{i=1}^N$
Modell:	$y = f_\theta(x) + \epsilon = x\theta + \epsilon$
Verlustfunktion:	$L(\theta \mid X, Y) = (Y - X\theta)^T (Y - \theta) \to \min$
Optimierung:	$\hat{\theta} = (X^T X)^{-1} X^T y$
Prognose:	$\hat{y}_* = x_* \hat{\theta}$
Validierung:	Berechne RMSE, R^2, angepassten R^2 oder p-Wert

Tabelle 4.1: Ordinary Least Squares (OLS).

4.1.2 Vor-und Nachteile

Lassen Sie uns in diesem Unterabschnitt die Vor- und Nachteile von OLS diskutieren. OLS hat den Vorteil der Einfachheit des Modells und der analytischen Formel der optimalen Parameter, was die Berechnung für optimale Parameter im Vergleich zu anderen Regressionsmethoden relativ einfach macht. Außerdem hat es aufgrund des linearen Modells von OLS eine große Interpretierbarkeit, und man kann erkennen, welche Eingabevariablen dazu beitragen, die Ausgabe am stärksten zu beeinflussen, indem man einfach die entsprechenden linearen Koeffizienten einordnet.

OLS hat jedoch auch verschiedene Einschränkungen. Zunächst einmal kann das lineare Modell zu einfach sein, um die Beziehung zwischen der Eingabe und der Ausgabe in Problemen der realen Welt zu modellieren. Daher besteht das potenzielle Risiko des sogenannten Underfitting-Problems.

Zweitens, selbst wenn ein lineares Modell ein vernünftiges Modell ist, kann OLS nach Lemma 4.1 nicht direkt angewendet werden, wenn $X^T X$ möglicherweise nicht invertierbar ist. Untersuchen wir die möglichen Gründe dafür, dass $X^T X$ nicht invertierbar ist, da dies uns helfen wird zu verstehen, in welchen Situationen OLS nicht anwendbar ist. Es gibt zwei hinreichende Bedingungen dafür, dass $X^T X$ nicht invertierbar ist: erstens die lineare Co-Abhängigkeit der Eingabedaten; zweitens, dass die Dimension der Eingabevariablen d strikt größer ist als die Stichprobengröße N. In dieser Situation kann die zufällige Auswahl eines möglichen zu einer geringen Vorhersagekraft des geschätzten Modells im Testdatensatz führen (Overfitting-Problem). Eine Möglichkeit, das Problem der Kolinearität der Eingabedaten zu lösen, besteht darin, die Principal Component Analysis (PCA) zur Dimensionsreduktion zu verwenden.

Schließlich ist die Verlustfunktion von OLS als gleichgewichtetes quadratisches Residuum definiert, was dazu führt, dass sie empfindlich gegenüber Ausreißern ist. Mögliche Lösungen umfassen die Vorverarbeitung der Daten, um Ausreißer zu entfernen, oder die Modifikation der Verlustfunktion, um die Möglichkeit des Auftretens von Ausreißern zu berücksichtigen.

Eine Zusammenfassung der Vor- und Nachteile von OLS finden Sie in der Tabelle 4.2.

Vorteile
-Das lineare Modell ist einfach und hat weniger wahrscheinlich das Problem der Überanpassung;
-Es gibt eine analytische Formel für die optimalen Parameter, und die Berechnungskosten sind gering;
-Es ist leicht verständlich und hat eine große Interpretierbarkeit.
Nachteile
-Das lineare Modell ist möglicherweise zu restriktiv, um Daten zu beschreiben (Underfitting-Problem);
-Auch wenn das lineare Modell ein vernünftiges Modell ist, hat OLS ein Problem, wenn $X^T X$ nicht invertierbar ist;
-Es ist empfindlich gegenüber Ausreißern.

Tabelle 4.2: Vor- und Nachteile von OLS.

4.2 Lineares Modell mit Regularisierung

Im vorherigen Unterabschnitt haben wir die potenziellen Probleme von OLS besprochen. Das Overfitting-Problem tritt auf, wenn die Dimension der Eingabe strikt größer als die Stichprobengröße ist. Hier erklären wir, wie Sie die Regularisierungsmethode verwenden können, um bei diesem Problem zu helfen.

4.2.1 Regularisierung

Regularisierung ist ein Prozess, bei dem zusätzliche Einschränkungen der Norm der parametrisierten Funktionsfamilie eingeführt werden, um solche Arten von schlecht gestellten Problemen zu lösen und eine Überanpassung zu verhindern.

Die Methode der regularisierten linearen Regression basiert auf OLS, fügt dem Optimierungsprozess jedoch zusätzliche Einschränkungen hinzu. Insbesondere betrachten wir das Constraint-Optimierungsproblem für die Least-Squared-Verlustfunktion:

$$\min_{\beta}(Y - X\beta)^T(Y - X\beta),$$

unterliegen dem $||\beta|| \leq t$.

Durch den Lagrange-Multiplikator entspricht dies dem Optimierungsproblem ohne Nebenbedingungen, indem der Verlustfunktion ein Strafterm hinzugefügt wird, dh für gegebenes $t > 0$,

$$\min_{\beta,\tilde{\lambda}}(Y - X\beta)^T(Y - X\beta) + \lambda(||\beta|| - t).$$

Dies motiviert uns, das folgende unbeschränkte Optimierungsproblem zu betrachten, d. h. für gegebenes $\tilde{\lambda}$,

$$\min_{\beta,\tilde{\lambda}}(Y - X\beta)^T(Y - X\beta) + \tilde{\lambda}(||\beta||), \tag{4.5}$$

wobei die genaue Beziehung zwischen t und $\tilde{\lambda}$ datenabhängig ist.

Im Prinzip kann $\tilde{\lambda}$ direkt berechnet werden, wenn man t, X und Y kennt. In der Praxis ist die Wahl des richtigen t genauso schwierig wie die Wahl von $\tilde{\lambda}$. Das Verfahren des regularisierten linearen Modells wird daher wie folgt skizziert:

- Geben Sie die Menge möglicher Werte für $\tilde{\lambda}$ an, die mit L bezeichnet werden;
- Lösen Sie für jedes $\tilde{\lambda} \in L$ das in der Gleichung 4.5 angegebene Optimierungsproblem ohne Nebenbedingungen und erhalten Sie den optimalen Parameter $\beta(\tilde{\lambda})$ und die entsprechende MSE($\tilde{lambda}$);
- Wenden Sie eine Kreuzvalidierung an, um das Beste gemäß der entsprechenden MSE auszuwählen, und der entsprechende optimale Parameter wird auf $\hat{\beta}$ gesetzt, wobei

$$\hat{\beta} = \beta(arg\min_{\tilde{\lambda} \in L} MSE(\tilde{\lambda})).$$

.

Das Universum der regularisierten linearen Regressionsmethoden wird nach der Norm der Parameter im Strafterm kategorisiert. Die l_p-Norm ist die in diesem Zusammenhang zu verwendende Standardnorm. Geben wir die Definition der l_p-Norm an. Die l_p-Norm eines beliebigen Elements $x \in \mathbb{R}^d$, bezeichnet mit $||x||_p$, ist wie folgt definiert:

$$||x||_p = (\sum_{i=1}^{d} |x^{(i)}|^p)^{1/p}.$$

Die drei Haupttypen der regularisierten linearen Regression sind in der Tabelle 4.3 aufgeführt. Wir konzentrieren uns auf die ersten beiden Methoden, d. h. Ridge Regression und Lasso Regression, vergleichen ihre Stärken und Schwächen und stellen die Verbindung zwischen ihnen basierend auf neueren Arbeiten her [Hoff (2017)].

Ridge Regression:	$\mathrm{L}(\beta\|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda\|\|\beta\|\|_2^2;$
Lasso Regression:	$\mathrm{L}(\beta\|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda\|\|\beta\|\|_1;$
Elastic Net:	$\mathrm{L}(\beta\|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda(\frac{1-\alpha}{2}\|\|\beta\|\|_2^2 + \alpha\|\|\beta\|\|_1).$

Tabelle 4.3: Haupttypen von regularisierten linearen Regressionsmethoden.

4.2.2 Ridge Regression

Betrachten wir die Ridge-Regression, in der die Verlustfunktion definiert ist als

$$L(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda||\beta||_2^2,$$

wobei $\lambda > 0$ ein Hyperparameter ist. Dank der Differenzierbarkeit der quadrierten l_2-Norm liefert der Parameterschätzer der Ridge-Regression ebenso eine analytische geschlossene Formel wie die von OLS (Lemma 3.2).

Lemma 4.2 (Ridge Regression Schätzer). Fixiere $\lambda > 0$. Bei gegebenem Input-Output-Datensatz $D = (X, Y)$ und unter der Annahme, dass die Standardeinstellung des linearen Regressionsmodells gilt, ist der Schätzer des optimalen Parameters $\hat{\theta}$ in der Ridge-Regression wie folgt gegeben:

$$\hat{\theta} = (X^TX + \lambda I)^{-1}X^TY, \tag{4.6}$$

wobei I eine Identitätsmatrix ist.

Anmerkung 4.2. Wenn $\lambda = 0$, wird die Ridge-Regression auf OLS reduziert. Selbst wenn X^TX nicht invertierbar ist, ist $X^TX + \lambda I$ für $\lambda > 0$ invertierbar, was die Gültigkeit der Ridge-Regression sicherstellt und das potenzielle Overfitting-Problem von OLS löst.

Eine Zusammenfassung der Ridge-Regression finden Sie in der Tabelle 4.4.

Dataset:	$D = (x_i, y_i)_{i=1}^N$
Modell:	$y = f_\theta(x) + \epsilon = x\theta + \epsilon$
Verlustfunktion:	$L(\theta\|X,Y) = (Y - X\theta)^T(Y - \theta) + \lambda\|\|\theta\|\|_2^2 \to \min$
Optimisierung:	$\hat{\theta} = (X^TX + \lambda I)^{-1}X^Ty$
Prognose:	$\hat{y}_* = x_*\hat{\theta}$
Validierung:	Berechne RMSE, R^2, angepassten R^2 oder p-Wert

Tabelle 4.4: Ridge Regression.

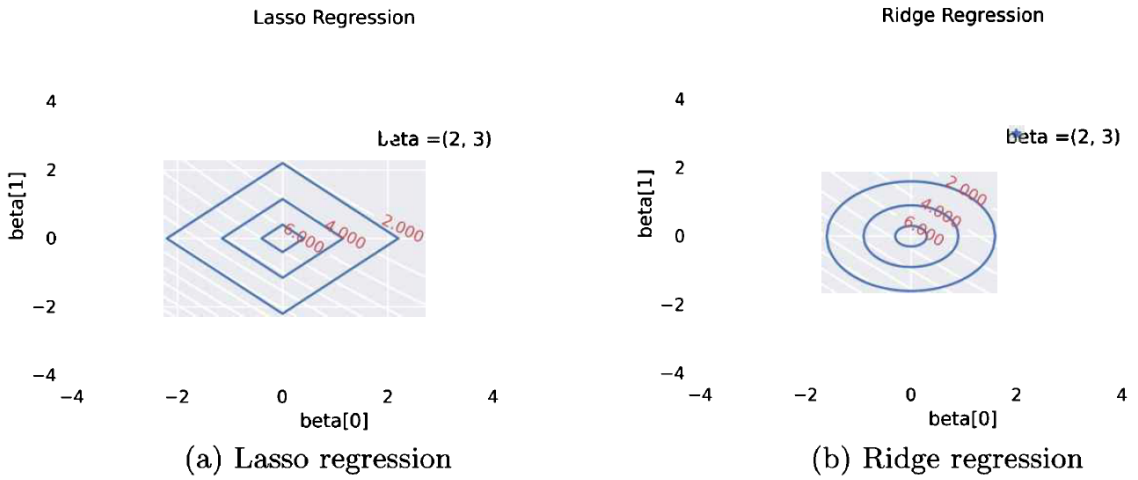
4.2.3 Lasso Regression

Die Verlustfunktion der Lasso-Regression ist definiert als

$$L(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda||\beta||_1,$$

wobei $\lambda > 0$ ein Hyperparameter ist. Wir können sehen, dass der einzige Unterschied zwischen diesen beiden Methoden die Form der Parameter im Strafterm ist.

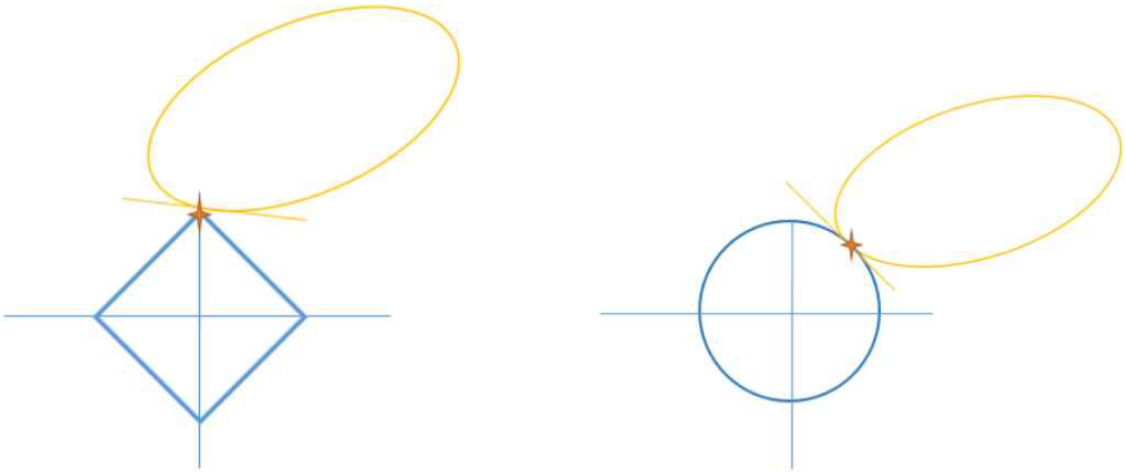
Im Allgemeinen hat die Lasso-Regression keine Lösung in geschlossener Form, so dass numerische Techniken erforderlich sind. Im Gegensatz zur Lasso-Regression lässt die Ridge-Regression eine analytische Lösung für die optimalen Parameter zu (Lemma 4.2). Der Vorteil von Lasso besteht darin, dass einige Koeffizienten auf genau Null gesetzt werden. Auf diese Weise kann Lasso für die Merkmalsauswahl verwendet werden. Der Grund dafür, dass Ridge-Regressionslösungen nicht genau Null erreichen, Lasso-Lösungen jedoch, liegt an der unterschiedlichen Natur der Geometrien der l_1-Norm und der l_2-Norm (siehe Abb. 4.1).



(a) Lasso regression (b) Ridge regression

Abbildung 4.1: Geometrie der Lasso- und Ridge-Regressionslösungen. [9]

Im zweidimensionalen Fall ist der Unterschied in der Abbildung unten zu sehen. Lasso-Regressionsbeschränkungen richten eine Raute an der Achse aus (Abbildung 4.1a). Die von den Lösungen eingeschriebenen Konturen (die blauen Kreise in der Abbildung 4.1b) können den Diamanten leicht an seiner Ecke schneiden. Dies impliziert, dass die Lasso-Lösung einige Koordinaten optimaler linearer Koeffizienten auf Null zwingt. Dies ist im rechten Fall für die Ridge-Regression viel unwahrscheinlicher, wo die l_2-Normeinschränkung einen Kreis einschreibt, was in der Abbildung 4.2 gezeigt wird.



(a) Lasso regression (b) Ridge regression

Abbildung 4.2: Typische Fälle der Lasso- und Ridge-Regressionslösungen.

Der Vergleich verschiedener Regularisierungsmethoden ist in der Tabelle 4.5 zusammengefasst.

-Die Lasso-Regression erfordert numerische Techniken zur Lösung, kann aber zur Merkmalsauswahl verwendet werden. -Ridge Regression hat eine geschlossene Lösung für die optimalen Parameter. -Elastic Nets ist eine Kombination aus Lasso- und Ridge-Regression.

Tabelle 4.5: Vergleich der drei regularisierten linearen Regressionsmethoden.

4.2.4 Numerisches Beispiel

In diesem Abschnitt verwenden wir synthetische Daten, um ein spärliches lineares Modell zu simulieren, bei dem die meisten linearen Koeffizienten dieses Modells Null sind. In diesem Beispiel können wir sehen, wie sich die verschiedenen regularisierten linearen Regressionsmethoden im Vergleich zu OLS verhalten.

Beispiel 4.1 (Sparse Linear Model). Wir simulieren 1200 Samples der Eingabe-Ausgabe-Paare $(x_i, y_i)_{i=1}^{1}200$ basierend auf dem folgenden dünnbesetzten linearen Modell:

$$y = x\beta + \epsilon,$$

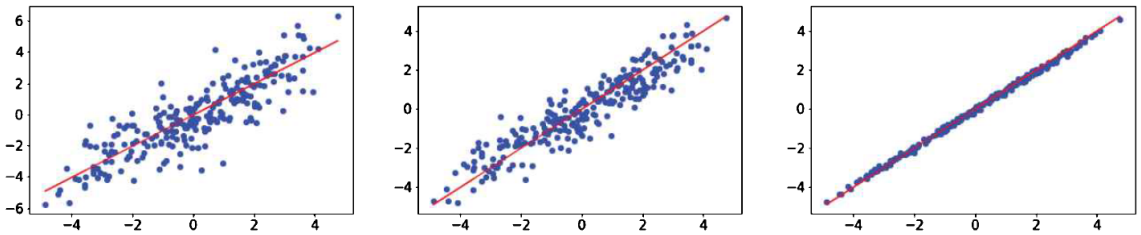
wobei $x \in \mathbb{R}^8 00, y \in \mathbb{R} und \beta$. Hier $[\beta^{(1)}, \beta^{(5)}, \beta^{(10)}] = [1.0, 3.0, 2.7]$. Wir verwenden 80% der Daten für das Trainingsset und den Rest für das Testset.

Wir verwenden das Scikit-Learn-Python-Paket, um die obige lineare Regression bequem zu implementieren. Der Schlüsselteil des Python-Codes ist in der Tabelle 4.6 angegeben.

```
#Skip the code for simulation of data # Split the dataset into the training set
and test set from sklearn.model_selection import train_test_split X_train, X_test,
Y_train, Y_test= train_test_split(X, Y, test_size = 0.2)

## OLS Regression from sklearn.metrics import r2_score, mean_squared_error from
sklearn import linear_model model =
sklearn.linear_model.LinearRegression()
model.fit(X_train, Y_train) Y_test_pred = model.predict(X_test) print ("r^2 on test
data : {}".format(r2_score(Y_test,Y_test_pred)) print("rmse on test data : {}".format( np.sqrt(mean_squred_error(Y_test, Y_test_pred))))

##Ridge Regression from sklearn.linear_model import RidgeCV ridgecv =
RidgeCV().fit(X_train, Y_train) Y_test_pred_Ridge =
ridgecv.predict(X_test)
print("r^2 on test data : {}".format( r2_score(Y_test,
Y_test_pred_Ridge)))
print("rmse on test data : {}".format( np.sqrt(mean_squared_error(Y_test, Y_test_pred_Ridge))))

## Lasso Regression from sklearn.linear_model import Lasso, LassoCV from
sklearn.feature_selection import SelectFromModel lassocv =
LassoCV(cv=20).fir(X_train, Y_train) Y_test_pred_Lasso =
lassocv.predict(X_test)
print("r^2 on test data :{}".format( r2_score(Y_test,
Y_test_pred_Lasso)))
print("rmse on test data : {}".format( np.sqrt(mean_squared_error(Y_test, Y_test_pred_Lasso))))
```

Tabelle 4.6. Python Code für OLS, Ridge Regression und Lasso Regression

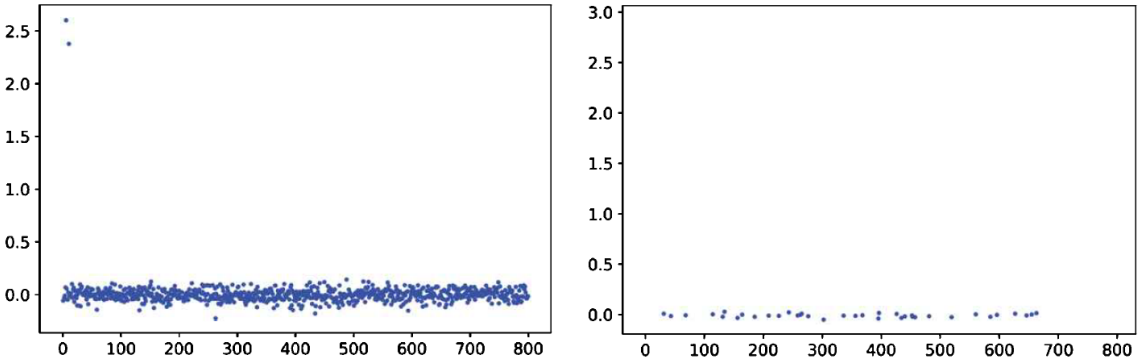
In der Abbildung 4.3 zeichnen wir die geschätzte Ausgabe gegen die tatsächliche Mittelwertfunktion der Ausgabe unter Verwendung von OLS, Lasso-Regression und Ridge-Regression auf dem Testdatensatz. Die rote Linie in jedem Subplot stellt die Identitätsfunktion dar, die einer perfekten Anpassung entspricht. Je näher die blaue Punktwolke an der roten Linie liegt, desto besser ist das entsprechende Regressionsverfahren. Die Abbildung 4.3 legt nahe, dass Lasso in diesem Beispiel am besten funktioniert.



(a) OLS (b) Ridge Regression (c) Lasso Regression

Abbildung 4.3: Vergleich von OLS, Lasso Regression und Ridge Regression.

Die Abbildung 4.4 zeichnet den Schätzer für lineare Koeffizienten ungleich Null über den entsprechenden Index der Koordinate der Eingabevariablen. Es unterstützt einen der Hauptunterschiede zwischen Lasso- und Ridge-Regression, d.h. die linearen Koeffizienten-Schätzer von Ridge-Regressionen sind dicht, und viele von ihnen sind nahe bei Null, aber nicht genau Null, während das Lasso viele der linearen Koeffizienten-Schätzer genau zu Null setzt. In einer ähnlichen Situation wie im Beispiel 4.1, wo die Daten auf der Grundlage des spärlichen linearen Modells simuliert werden, ist die Lasso-Regression der Ridge-Regression aufgrund ihrer Fähigkeit zur Merkmalsauswahl vorzuziehen.



(a) Ridge Regression (b) Lasso Regression

Abbildung 4.4: Visualisierung linearer Koeffizienten in regularisierten Regressionsmethoden.

5 Zusammenfassung

Obwohl Machine Learning eine neuere Technologie ist, gibt es viele Akademiker und Branchenexperten, unter denen Machine Learning sehr beliebt ist. Man kann mit Sicherheit sagen, dass es in diesem Bereich noch viel mehr Innovationen geben wird. Und die Einführung von Machine Learning hat auch seine eigenen Rückschläge aufgrund von Datensensibilität, Infrastrukturanforderungen, der Flexibilität von Geschäftsmodellen usw. Aber die Vorteile überwiegen die Nachteile und helfen viele Probleme mit Machine Learning zu lösen.

Da maschinelle Lerntechniken viel mehr sicherer sind als menschliche Praktiken, ist es die beste Wahl für den Finanzbereich. Es würde dazu beitragen, Banken und anderen Finanzinstituten Chancen zu eröffnen, indem es ihnen hilft, enorme Verluste aufgrund von Zahlungsausfällen zu vermeiden. Finanzen sind in allen Ländern der Welt eine sehr kritische Angelegenheit, und der Schutz vor Bedrohungen und die Verbesserung ihrer Abläufe würden allen helfen, schneller zu wachsen und zu gedeihen.

Literaturverzeichnis

[1] H. Ni, X. Dong, J. Zheng, and G. Yu, *An Introduction to Machine Learning in Quantitative Finance*. World Scientific, 2021.

[2] A. L. Samuel, "Some studies in machine learning using the game of checkers," 1959.

[3] D. M. Hawkins, "The problem of overfitting," 2004.

[4] P. N. Stuart J. Russell, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

[5] https://www.turing.ac.uk/research/research-projects/analysing-noisy-data-streams.

[6] https://www.jpmorgan.com/insights/research/machine-learning.

[7] https://igniteoutsourcing.com/fintech/machine-learning-in-finance/.

[8] https://www.independent.co.uk/news/business/news/jp-morgan-software-lawyers-coin-contract-intelligence-parsing-financial-deals-second
html.

[9] https://jamesmccammon.com/2014/04/20/lasso-and-ridge-regression-in-r/.