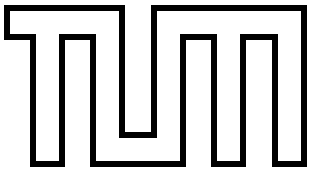


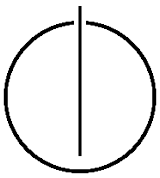
FAKULTÄT FÜR INFORMATIK

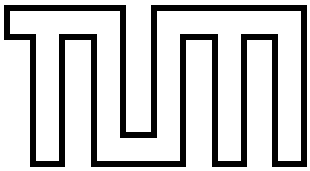
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Abschlussarbeit in Informatik

Effiziente statistische Methoden für Datenbanksysteme

Thomas Heyenbrock





FAKULTÄT FÜR INFORMATIK

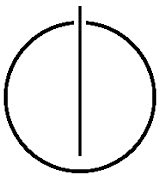
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Abschlussarbeit in Informatik

Effiziente statistische Methoden für Datenbanksysteme

Efficient statistical methods for database systems

Autor:	Thomas Heyenbrock
Aufgabensteller:	Prof. Alfons Kemper, Ph.D.
Betreuer:	Maximilian E. Schüle, M.Sc.
Datum:	15.01.2017



Ich versichere, dass ich diese Abschlussarbeit selbständig verfasst und nur die angegebenen
Quellen und Hilfsmittel verwendet habe.

München, den 3. Januar 2018 Thomas Heyenbrock

Abstract

An abstracts abstracts the thesis!

Contents

Outline of the Thesis

Teil I: Introduction and Theory

CHAPTER 1: INTRODUCTION

This chapter presents an overview of the thesis and it purpose. Furthermore, it will discuss the sense of life in a very general approach.

CHAPTER 2: THEORY

No thesis without theory.

Teil II: The Real Work

CHAPTER 3: OVERVIEW

This chapter presents the requirements for the process.

1. Einführung und typische statistische Problemstellungen

Here starts the thesis with an introduction. Please use nice latex and bibtex entries [1]. Do not spend time on formating your thesis, but on its content.

1.1. Latex Introduction

There is no need for a latex introduction since there is plenty of literature out there.

2. Grundlagen statistischer Methoden

Bei der Regressionsanalyse geht es im Allgemeinen darum, das Verhalten einer Größe Y in Abhängigkeit einer oder mehrerer anderer Größen $X_1, X_2, \ldots, X_n$ zu modellieren. Die Größe Y wird abhängig genannt, die Größen X_i nennt man unabhängig. Für diese Arbeit wollen wir zunächst einige Annahnmen über diese voraussetzen. Diese Punkte gelten immer, falls nicht explizit etwas anderes festgelegt wird.

- Die genannten Größen sind Zufallsvariablen. Das sind Funktionen deren Werte die Ergebnisse eines Zufallsvorgangs darstellen.
- Die Zufallsvariablen sind auf der Menge $M = \{1, \ldots, m\}$ definiert und bilden in die reellen Zahlen ab:

  $$Y : M \to \mathbb{R}, \;\; X_1 : M \to \mathbb{R}, \;\; \ldots, \;\; X_n : M \to \mathbb{R}$$

  Das bedeutet die Zufallsvariablen sind metrisch skaliert. Die m Zahlen in der Menge M entsprechen den m Datenpunkten, die wir als Datenbasis für die Regressionsanalyse besitzen.
- Wir verwenden die folgenden Abkürzungen für die Werte der Zufallsvariablen:

  $$\begin{aligned} y_i &:= Y(i) \quad \text{für alle } i \in M, \\ x_{i,j} &:= X_j(i) \quad \text{für alle } i \in M \;\; \text{und } 1 \leq j \leq n \end{aligned}$$

- Einen Datenpunkt aus unserer Datenbasis fassen wir als Vektor der Länge $(n+1)$ auf. Damit lässt sich die Datenbasis schreiben als:

  $$(y_1, x_{1,1}, \ldots, x_{1,n}), \ldots, (y_m, x_{m,1}, \ldots, x_{m,n})$$

Das Modell definieren wir anhand einer Funktion f, welche für Werte der unabhängigen Variablen einen geschätzten Wert für die abhängige Variable liefert. Idealerweise existiert eine Funktion, die zum Einen eine einfache Darstellung (z.B. durch eine arithmetische Formel) besitzt und zum Anderen alle unabhängigen Werte der Datenmenge exakt prognostiziert. Das bedeutet:

$$y_i = f(x_{i,1}, \ldots, x_{i,n}) \quad \text{für alle } \; 1 \leq i \leq m$$

Falls eine Formel wie hier für alle Datenpunkte gelten soll, verwenden wir als Abkürzung auch die Zufallsvariablen selbst, also:

$$Y = f(X_1, \ldots, X_N)$$

Im Allgemeinen ist es nicht möglich eine Funktion f zu finden, die beide Eigenschaften erfüllt. Man versucht also eine Funktion mit einer möglichst einfachen Form zu finden, die die Datenmenge möglichst gut approximiert. Wir definieren für jeden Datenpunkt den Fehler e_i, der sich durch die nicht exakte Modellfunktion f ergibt:

$$e_i = y_i - f(x_{i,1}, \ldots, x_{i,n})$$

Ziel der Regressionsanalyse ist es nun eine Funktion f zu finden, die diese Fehlerterme minimiert. Diese Optimierung geschieht global, also für die gesamte Datenmenge und nicht nur für einzelne Datenpunkte.

2.1. Lineare Regression

Bei der linearen Regression geht man von einem linearen Zusammenhang zwischen der abhängigen und den unabhängigen Variablen aus. Die Funktion f ist also von folgender Form:

$$f(x_1, \ldots, x_n) = \alpha + \sum_{i=1}^{n} \beta_i \cdot x_i \quad \text{mit} \quad \beta_i \in \mathbb{R}$$

Das Maß für die Qualität einer Funktion f definiert durch die Parameter $\alpha, \beta_1, \ldots, \beta_n$ ist die Summe der quadrierten Fehlerterme:

$$E(\alpha, \beta_1, \ldots, \beta_n) = \sum_{j=1}^{m} e_j^2 = \sum_{j=1}^{m} (y_j - f(x_{j,1}, \ldots, x_{j,n}))^2 = \sum_{j=1}^{m} \left(y_j - \alpha - \sum_{i=1}^{n} \beta_i \cdot x_{i,j} \right)^2$$

Wir suchen also die Parameter $\hat{\alpha}, \hat{\beta}_1, \ldots, \hat{\beta}_n$ für die gilt:

$$E(\hat{\alpha}, \hat{\beta}_1, \ldots, \hat{\beta}_n) = min\left\{ E(\alpha, \beta_1, \ldots, \beta_n) \mid \alpha \in \mathbb{R}, \beta_1 \in \mathbb{R}, \ldots, \beta_n \in \mathbb{R} \right\}$$

Um dieses Minimierungsproblem zu lösen berechnen wir die partiellen Ableitungen von E.

$$\begin{aligned}
\frac{\partial E}{\partial \alpha} &= -2 \cdot \sum_{j=1}^{m} (y_j - f(x_{j,1}, \ldots, x_{j,n})) = -2 \cdot \sum_{j=1}^{m} \left(y_j - \alpha - \sum_{i=1}^{n} \beta_i \cdot x_{i,j} \right) \\
\frac{\partial E}{\partial \beta_k} &= -2 \cdot \sum_{j=1}^{m} x_{k,j} \cdot (y_j - f(x_{j,1}, \ldots, x_{j,n})) \\
&= -2 \cdot \sum_{j=1}^{m} x_{k,j} \cdot \left(y_j - \alpha - \sum_{i=1}^{n} \beta_i \cdot x_{i,j} \right) \quad \text{für } 1 \leq k \leq n
\end{aligned}$$

Durch Nullsetzen der partiellen Ableitungen erhält man ein lineares Gleichungssystem mit $(n+1)$ Gleichungen und ebensovielen Unbekannten.

$$\frac{\partial E}{\partial \alpha} = 0, \quad \frac{\partial E}{\partial \beta_1} = 0, \quad \ldots, \quad \frac{\partial E}{\partial \beta_n} = 0$$

Die Lösung dieses Gleichungssystems (falls eine existent) ist das gesuchte Minimum.

2.1.1. Einfache lineare Regression

Man spricht von einfacher linearer Regression, wenn man mit nur eine unabhängige Variable arbeitet. Anschaulich möchte mann hier die bestmögliche Schätzgerade durch eine gegebene Punktwolke legen.

Wir nennen die unabhängige Variable in diesem Kapitel statt X_1 einfach nur X. Ebenso schreiben wir $\beta_1 = \beta$ und $x_{1,j} = x_j$. Dann können wir das lineare Gleichungssystem zum Auffinden des Minimums explizit aufschreiben:

$$0 = -2 \cdot \sum_{j=1}^{m} (y_i - \alpha - \beta \cdot x_j)$$
$$0 = -2 \cdot \sum_{j=1}^{m} x_j \cdot (y_i - \alpha - \beta \cdot x_j)$$

Für dieses Gleichungssystem kann die Lösung explizit angegeben werden, wobei wir hier nicht näher auf die Herleitung dieses Ergebnisses eingehen wollen:

$$\hat{\beta} = \frac{\sum_{j=1}^{m} (x_j - \bar{x})(y_j - \bar{y})}{\sum_{j=1}^{m} (x_j - \bar{x})^2}$$
$$\hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

Dabei bezeichnen $\bar{x}$ und $\bar{y}$ die Mittelwerte von X respektive Y.

2.1.2. Multible lineare Regression

Bei multibler linearer Regression existieren mindestens zwei unabhängige Variablen. Hier ist es nicht mehr zweckmäßig eine explizite Lösung anzugeben. Hier sind alternative Methoden zur Berechnung der Parameter nötig.

Neben einer Vielzahl von Algorithmen, die ein Optimierungsproblem iterativ lösen, gibt es auch die Möglichkeit die Parameter durch Matrizenmultiplikation zu berechnen. Definieren wir dazu die folgenden Matrizen und Vektoren:

$$X = \begin{pmatrix} 1 & x_{1,1} & \dots & x_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m,1} & \dots & x_{m,n} \end{pmatrix} \in \mathbb{R}^{m \times (n+1)}$$

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{R}^{m \times 1}, \quad b = \begin{pmatrix} \hat{\alpha} \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_n \end{pmatrix} \in \mathbb{R}^{(n+1) \times 1}$$

Dabei ist b der Vektor mit gesuchten Parametern für die Minimierung der kleinsten Quadrate. Falls die Matrix $X^T X$ invertiertbar ist, gilt die folgende Formel für die Berechnung der gesuchten Parameter:

$$b = (X^T X)^{-1} X^T y$$

2.2. Logistische Regression

Die logistische Regression findet Anwendung im Falle, dass die abhängige Variable eine binäre Variable ist, also eine Variable, die nur zwei Werte annehmen kann. Oft handelt es sich um eine Eigenschaft, die ein bestimmter Datensatz besitzt oder nicht, wie zum Beispiel ein Premium-Abonnement für eine Web-Service oder der Besitz eines Auto. Auch das Geschlecht einer Person ist ein Beispiel für eine binäre Variable. Wir bezeichnen die beiden möglichen Werte einer solchen Variablen hier immer mit 0 und 1. Die Zuordnung vom Merkmal zur Zahl ist frei wählbar.

Lineare Regression eignet sich oft nicht zur Modellierung einer binären Variablen, da eine lineare Funktion in der Regel unbeschränkt ist, also insbesondere Werte größer als 1 und kleiner als 0 annimmt. Um diesem Problem abzuhelfen wählen wir eine Funktion, die beliebige Zahlen auf das Interval $[0, 1]$ abbildet. Im Falle der logistischen Regression verwendet man die gleichnamige logistische Funktion:

$$l : \mathbb{R} \to (0, 1), \quad x \mapsto \frac{1}{1 + e^{-x}}$$

Diese Funktion wendet man nun auf die Linearkombination aller unabhängigen Variablen mit Parametern $\beta_1, \ldots, \beta_n$ und konstantem Term α an. Zur Vereinfachung definieren wir für das restliche Kapitel die Variable c wiefolgt:

$$c := \alpha + \sum_{i=1}^{n} \beta_i \cdot x_{i,j}$$

Das Ergebnis der Funktion l für den i-ten Datensatz bezeichnen wir mit π_i

$$\pi_i = \pi_i(\alpha, \beta_1, \ldots, \beta_n) := l\left(\alpha + \sum_{j=1}^{n} \beta_j \cdot x_{i,j}\right) = \frac{1}{1 + e^{-c}}$$

Wir stellen hierbei fest, dass folgende Identität gilt:

$$\begin{aligned}
\pi(-\alpha, -\beta_1, \ldots, -\beta_n) &= \frac{1}{1 + e^{c}} = \frac{1 + e^{c} - e^{c}}{1 + e^{c}} \\
&= 1 - \frac{e^{c}}{1 + e^{c}} = 1 - \frac{1}{e^{-c} + 1} \\
&= 1 - \pi(\alpha, \beta_1, \ldots, \beta_n)
\end{aligned}$$

Anschaulich repräsentiert π_i die Wahrscheinlichkeit dafür, dass die abhängige Variable eines Datensatzes mit unabhängigen Variablen $x_{i,1}, \ldots, x_{i,n}$ gleich 1 ist, also:

$$\pi_i = P(Y_i = 1 | X_1 = x_{i,1}, \ldots, X_n = x_{i,n})$$

Man möchte die Parameter $\alpha, \beta_1, \ldots, \beta_n$ nun so schätzen, dass die Wahrscheinlichkeit für das Auftreten der vorhandenen Datenbasis maximiert wird. Diese Wahrscheinlichkeit ist

gegeben durch:

$$\begin{aligned} L(\alpha, \beta_1, \ldots, \beta_n) &= \prod_{i=1}^{m} P(Y_i = y_i | X_1 = x_{i,1}, \ldots, X_n = x_{i,n}) \\ &= \prod_{i=1}^{m} y_i \cdot \pi_i(\alpha, \beta_1, \ldots, \beta_n) + (1 - y_i) \cdot (1 - \pi_i(\alpha, \beta_1, \ldots, \beta_n)) \\ &= \prod_{i=1}^{m} y_i \cdot \pi_i(\alpha, \beta_1, \ldots, \beta_n) + (1 - y_i) \cdot \pi_i(-\alpha, -\beta_1, \ldots, -\beta_n)) \end{aligned}$$

Da alle y_i im Fall der logistischen Regression entweder gleich 0 oder gleich 1 sind, ist immer nur einer der beidem Summanden in jedem Faktor nicht null. Diese Fallunterscheidung kann man auch in das Vorzeichen der Parameter verschieben, da sich die beiden möglichen Faktoren nur darin unterscheiden. Dann erhält man:

$$\begin{aligned} L(\alpha, \beta_1, \ldots, \beta_n) = \prod_{i=1}^{m} \pi((&2 \cdot y_i - 1) \cdot \alpha, \\ &(2 \cdot y_i - 1) \cdot \beta_1, \\ &\ldots, \\ &(2 \cdot y_i - 1) \cdot \beta_n) \end{aligned}$$

Das Verfahren der Maximierung dieser Wahrscheinlichkeit bezeichnet man auch als Maximum-Likelihood-Methode. Die Funktion L nennt man dementsprechend auch Likelihoodfunktion. Oft maximiert man nicht L direkt, sondern eher $\ln(L)$. Der Sinn ist, dass man das Produkt damit in eine Summe einzelner Logarithmen umwandeln kann, welche wiederum einfacher abzuleiten ist. Das darf man machen, da der Logarithmus eine stetig wachsende Funktion ist und die Werte von L stets zwischen 0 und 1 liegen.

Wir suchen also die Parameter $\hat{\alpha}, \hat{\beta}_1, \ldots, \hat{\beta}_n$ mit:

$$L(\hat{\alpha}, \hat{\beta}_1, \ldots, \hat{\beta}_n) = \max \{L(\alpha, \beta_1, \ldots, \beta_n) \mid \alpha \in \mathbb{R}, \beta_1 \in \mathbb{R}, \ldots, \beta_n \in \mathbb{R}\}$$

In diesem Fall kommt man leider nicht mehr an einer iterativen Lösung vorbei, da die partiellen Ableitungen und das entstehende lineare Gleichungssystem nicht mehr exakt lösbar sind. Eine der einfachsten Methoden zur Lösung von Optimierungsproblemen ist das Gradientenverfahren, welches im kommenden Teilkapitel kurz eingeführt wird.

2.2.1. Gradientenverfahren

Das Gradientenverfahren ist ein iterativer Algorithmus zur Lösung von Optimierungsproblemen. Nachdem wir hier bei der logistischen Regression eine Funktion maximieren wollen führen wir das Gradientenverfahren dementsprechend ein. Man kann dasselbe Verfahren aber auch zur Lösung von Minimierungsproblem einsetzen. Gegeben sei also eine Funktion der folgenden Form, die maximiert werden soll:

$$L : \mathbb{R}^{n+1} \to \mathbb{R}, \;\; (\alpha, \beta_1, \ldots, \beta_n) \mapsto L(\alpha, \beta_1, \ldots, \beta_n)$$

Beim Gradientenverfahren beginnt man mit beliebigen Startwerten $\alpha_0, \beta_{0,1}, \ldots, \beta_{0,n}$ und einer Schrittweite $s \in \mathbb{R}^+$. Vom Startpunkt aus geht man nun in die Richtung des steilsten

Anstieges der Funktion und erhält dadurch neue Werte. Diese Richtung ist gerade der sogenannte Gradient der Funktion L.

Der Gradient ist ein Vektor, der sich aus den partiellen Ableitungen von L nach jeweils einer Variablen zusammensetzt und wird wiefolgt notiert:

$$\text{grad}(L) = \begin{pmatrix} \partial L/\partial\alpha \\ \partial L/\partial\beta_1 \\ \vdots \\ \partial L/\partial\beta_n \end{pmatrix}$$

Der Gradient von L ist also wiederum eine Funktion, die Werte $\alpha, \beta_1, \ldots, \beta_n$ auf einen Vektor der Länge $n+1$ abbildet. Der iterative Schritt des Verfahrens definiert sich wiefolgt:

$$\begin{pmatrix} \alpha_{i+1} \\ \beta_{i+1,1} \\ \vdots \\ \beta_{i+1,n} \end{pmatrix} = \begin{pmatrix} \alpha_i \\ \beta_{i,1} \\ \vdots \\ \beta_{i,n} \end{pmatrix} + s \cdot \text{grad}(L)(\alpha_i, \beta_{i,0}, \ldots, \beta_{i,n})$$

Danach muss noch getestet werden, dass L für die neuen Parameter auch wirklich einen größeren Wert annimmt also zuvor. Falls nicht, muss die Schrittweite s verkleintert werden, zum Beispiel um einen festen zuvor definierten Faktor.

Das Verfahren konvergiert nicht zwingend, falls die Funktion nach oben unbeschränkt ist. In unserem Fall ist die Likelihoodfunktion L aber durch 1 nach oben beschränkt. Trotzdem konvergiert das Gradientenverfahren nur mit Sicherheit gegen ein lokales Maximum von L, welches nicht zwingend auch ein globales Maximum sein muss.

2.2.2. Gradient bei logistischer Regression

Um das Gradientenverfahren bei logistischer Regression einsetzen zu können, muss der Gradient für den Logarithmus der Likelihoodfunktion bekannt sein. In diesem Kapitel bilden aso wir die partiellen Ableitungen nach allen Parametern.

Um $L_{log} = \ln(L)$ partiell ableiten zu können, berechnen wir zuerst die partiellen Ableitungen aller π_i. Für die partielle Ableitung nach α ergibt sich mit der Kettenregel folgende Funktion:

$$\begin{aligned} \frac{\partial \pi_i}{\partial \alpha} &= -\left(1 + \exp\left(-\alpha - \sum_{j=1}^{n} \beta_j \cdot x_{i,j}\right)\right)^{-2} \cdot \exp\left(-\alpha - \sum_{j=1}^{n} \beta_j \cdot x_{i,j}\right) \cdot (-1) \\ &= \left(1 + \exp\left(-\alpha - \sum_{j=1}^{n} \beta_j \cdot x_{i,j}\right)\right)^{-1} \cdot \left(1 + \exp\left(\alpha + \sum_{j=1}^{n} \beta_j \cdot x_{i,j}\right)\right)^{-1} \\ &= \pi_i(\alpha, \beta_1, \ldots, \beta_n) \cdot \pi_i(-\alpha, -\beta_1, \ldots, -\beta_n) \end{aligned}$$

Die partiellen Ableitungen einem der β_k für $k = 1, \ldots, n$ kann fast analog gebildet werden. Bei der Anwendung der Kettenregel auf die innerste lineare Funktion bleibt jedoch noch der konstanter Faktor $x_{i,k}$ übrig.

$$\frac{\partial \pi_i}{\partial \beta_k} = x_{i,k} \cdot \pi_i(\alpha, \beta_1, \ldots, \beta_n) \cdot \pi_i(-\alpha, -\beta_1, \ldots, -\beta_n)$$

Nun zur den eigentlichen partiellen Ableitunge. Zuerst noch einmal zu der Funktion, die wir nun ableiten wollen. Definieren wir $\tilde{\alpha} := (2y_i - 1)\alpha$ und $\tilde{\beta}_i := (2y_i - 1)\beta_i$ für $i = 1, \ldots, n$. Dann erhält man:

$$\begin{aligned} L_{log}(\alpha, \beta_1, \ldots, \beta_n) &= \ln(L(\alpha, \beta_1, \ldots, \beta_n)) \\ &= \sum_{i=1}^{m} \ln\left(y_i \cdot \pi(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)\right) \end{aligned}$$

Leitet man nach α ab, so erhält man:

$$\begin{aligned} \frac{\partial L_{log}}{\partial \alpha} &= \sum_{i=1}^{m} \frac{\partial}{\partial \alpha} \left(\ln\left(\pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)\right)\right) \\ &= \sum_{i=1}^{m} \left(\pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)\right)^{-1} \cdot \frac{\partial \pi_i}{\partial \tilde{\alpha}} \cdot \frac{\partial \tilde{\alpha}}{\partial \alpha} \\ &= \sum_{i=1}^{m} \left(\pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)\right)^{-1} \cdot \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n) \cdot (1 - \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)) \cdot (2y_i - 1) \\ &= \sum_{i=1}^{m} (1 - \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)) \cdot (2y_i - 1) \end{aligned}$$

Für die partielle Ableitung nach β_k erhält man analog:

$$\frac{\partial L_{log}}{\partial \beta_k} = \sum_{i=1}^{m} x_{i,k} \cdot (1 - \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)) \cdot (2y_i - 1)$$

Betrachten wir die Summanden der partiellen Ableitungen nun getrennt für die beiden möglichen Werten von y_i. Ist $y_i = 0$ dann gilt:

$$\begin{aligned} (1 - \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)) \cdot (2y_i - 1) &= (1 - \pi_i(-\alpha, -\beta_1, \ldots, -\beta_n)) \cdot (-1) \\ &= -1 + (1 - \pi_i(\alpha, \beta_1, \ldots, \beta_n)) \\ &= -\pi_i(\alpha, \beta_1, \ldots, \beta_n) \end{aligned}$$

Für $y_i = 1$ ergibt sich folgendes:

$$\begin{aligned} (1 - \pi_i(\tilde{\alpha}, \tilde{\beta}_1, \ldots, \tilde{\beta}_n)) \cdot (2y_i - 1) &= (1 - \pi_i(\alpha, \beta_1, \ldots, \beta_n)) \cdot (2 - 1) \\ &= 1 - \pi_i(\alpha, \beta_1, \ldots, \beta_n) \end{aligned}$$

Damit können wir die partiellen Ableitungen weiter vereinfachen:

$$\begin{aligned} \frac{\partial L_{log}}{\partial \alpha} &= \sum_{i=1}^{m} y_i - \pi_i(\alpha, \beta_1, \ldots, \beta_n) \\ \frac{\partial L_{log}}{\partial \beta_k} &= \sum_{i=1}^{m} x_{i,k} \cdot (y_i - \pi_i(\alpha, \beta_1, \ldots, \beta_n)) \end{aligned}$$

Diese Darstellung der partiellen Ableitungen erlaubt es uns später in SQL den Gradienten zu berechnen. Der Term innerhalb der Summe wird einfach für jeden Datenpunkt berechnet, danach wird die resultierende Spalte zusammen mit einer Gruppierung summiert.

3. Anwendung statistischer Methoden

In diesem Kapitel werden nun mehrere Programmiersprachen vorgestellt, die sich für Regressionsanalyse eignen. Im letzten Teilkapitel wird demonstriert, wie man solche Methoden mit vorhandener SQL-Syntax umsetzen und durchführen kann.

3.1. Beispieldaten

Um Regressionsanalyse auch praktisch betreiben zu können, arbeiten wir in dieser Arbeit mit einem Satz an Beispieldaten. Diese Daten wurden mit einem Python-Skript erstellt, welches im als Ganzes im Anhang zu finden ist. Dabei werden die einzelnen Merkmale eines Datensatzes mit Absicht so erstellt, dass eine Korrelation zwischen diesen bewusst erzeugt oder nicht erzeugt wird. Diese Beispieldaten liegen in Form einer csv-Datei vor, welche in jeder Sprache einfach eingelesen werden kann.

Wir betrachten hier fiktive Kunden von Amazon. Für jeden Kunden wissen wir das Alter, die Anzahl seiner Käufe, die Summe des ausgegebenen Geldes und ob der Kunde Amazon-Prime Mitglied ist oder nicht. Der ausgegebene Betrag wird in Cent angegeben, um mit ganzen Zahlen rechnen zu können. Die Prime-Mitgliedschaft wird mit einer 1 symbolisiert, während eine 0 das Gegenteil bedeutet.

Insgesamt wurden für diese Arbeit 100.000 solcher Datensätze erzeugt. In der folgenden Tabelle sind die ersten 10 Datensätze beispielhaft dargestellt.

age	**purchases**	**money**	**prime**
30	1	4421	0
30	11	23346	1
33	1	4010	0
31	19	52517	1
29	3	8046	0
28	12	25295	0
41	16	38236	1
23	3	7098	1
25	1	2707	0
38	20	50976	1

Wir definieren uns außerdem drei Fragestellungen, welche wir jeweils mit einer Art der in Kapitel 2 vorgestellten Regressionen beantworten werden:

1. Zuerst wollen wir wissen, ob das ausgegebene Geld mit der Anzahl der Käufe in linearem Zusammenhang steht. Diese Fragen können wir mit einfacher linearer Regression beantworten. $money$ ist hierbei die abhängige Variable und $purchases$ ist die unabhängige Variable.

2. Die zweite Frage ist ähnlich der ersten, nur wollen wir hier wissen, ob neben der Anzahl der Käufe auch das Alter des Kunden einen linearen Einfluss auf das ausgegebene Geld hat. Hier haben wir nun zwei unabhängige Variablen, nämhlich age und $purchases$. Die abhängige Variable bleibt $money$. Diese Frage beantworten wir also mit multipler linearer Regression.

3. Als letztes interessiert uns, ob eine Prime-Mitgliedschaft von der Summe des ausgegebenen Geldes zusammenhängt. $money$ ist also nun die unabhängige Variable, während $prime$ die abhängige Variable ist. Außerdem ist $prime$ eine binäre Variable. Deshalb nutzen wir hier also logistische Regression.

3.2. Regression in R

Das R-Projekt oder einfach nur R ist eine Sprache für statistische Berechnungen und graphische Darstellung. Damit ist R wie geschaffen für Regressionsanalyse. Von allen hier behandelten Sprachen ist R damit auch die einfachste und direkteste für Regression.

In R sind einfache Datenstrukturen wie Vektoren, Matrizen und Listen als Datentypen vorhanden. Darauf aufbauend existieren sogenannten Dataframes. Diese sind eine Liste von Vektoren der gleichen Länge und werden gerne zur Repräsentation von Datentabellen verwendet. Die Vektoren der Liste entsprechen dann den Spalten der Tabelle.

In R lassen sich außerdem sehr einfach sogenannte Modelle definieren, welche als Eingabe nur die Daten und eine Formel benötigen. Eine Formel ist von der Form $y \sim modell$ und enthält den funktionalen Zusammenhang zwischen der abhängigen und den unabhängigen Variablen. Mit diesen linearen Modellen berechnet man schnell und einfach die Parameter zu einem gegebenen Datensatz unter Verwendung der lm-Funktion in R.

3.2.1. Einfache lineare Regression

Betrachten wir also Frage Nummer 1 aus dem vorherigen Teilkapitel. Die Formel lautet dann einfach $money \sim purchases$. Um die Parameter zu berechnen sind nur wenige Zeilen R-Code nötig. Man liest die Daten aus der csv-Datei, erstellt das Modell mit der Formel und berechnet die Parameter. Die print-Funktion druckt am Ende das Ergebnis:

```
data <- read.csv2("sample.csv", sep = ",", header = TRUE)
modell <- as.formula("money ~ purchases")
slr <- lm(modell, data = data)
print(slr)
```

Das Ergebnis des obigen Codes ist folgendes:

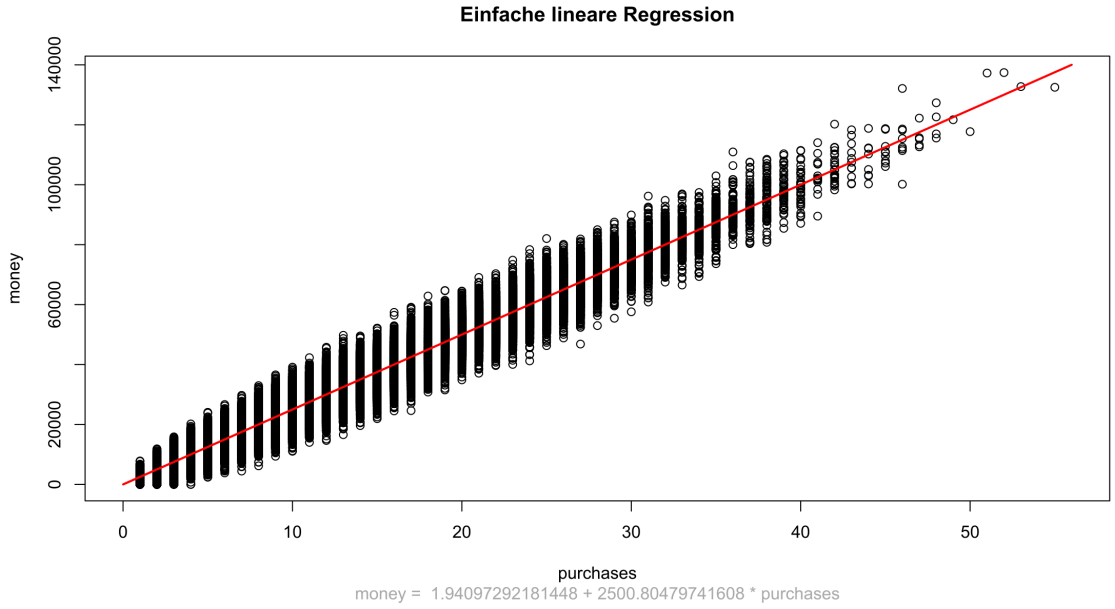
```
Coefficients:
(Intercept)    purchases
      1.941     2500.805
```

Der Wert unter $(Intercept)$ entspricht dabei dem Paramter α in unserer Notation, der Wert unter $purchases$ entspricht β.

Wir wollen dieses Ergebnis kurz interpretieren. Der kleine Wert für α entspricht der Intuition, dass ein Kunde ohne Käufe auch kein Geld ausgegeben hat. Der relativ große

Wert von ca. 2500 für β zeigt, dass die Anzahl der gekauften Artikel sehr einen großen Einfluss auf das ausgegebene Geld hat. Die Kunden geben pro gekauftem Artikel etwa 2500 Cent, also 25 Euro aus.

R verfügt auch über Möglichkeiten zur graphischen Darstellung. Lässt man die die Datenpunkte und die lineare Ausgleichsfunktion mit den berechneten Parameter plotten, erhält man dieses Ergebnis:



3.2.2. Multiple lineare Regression

Bei multipler lineare Regression unterscheidet sich der R-Code nur in der Wahl der Formel. Hier wollen wir $money$ durch eine Summe von $purchases$ und age modellieren, deshalb lautet die Formel hier $money \sim purchases + age$. Man erhält das folgende Ergebnis.

```
Coefficients:
(Intercept)     purchases           age
    -16.4842     2500.8042        0.5318
```

Auch hier eine kurze Interpretation dieses Ergebnisses: Der Wert für α ist wieder relativ klein, der Wert für das β zu $purchases$ ist fast exakt derselbe wie bei einfacher linerarer Regression, was bei denselben Daten auch zu erwarten war. Der Wert für das β zu age ist dagegen nahe bei null. Das bedeutet, dass das Alter neben der Anzahl der Käufe keinen signifikanten Einfluss auf das ausgegebene Geld hat.

3.2.3. Logistische Regression

Bei logistischer Regression nutzen wir nun nicht mehr ein lineares Modell wie bisher, sondern ein generalisiertes lineares Modell. Logistische Regression ist im Wesentlichen ein Spezialfall dieses Modelles. Die zugehörige Funktion nennt sich glm in R. Um logistische Regression damit betreiben zu können, wählt man den Parameter $family$ dieser Funktion als $binomial$.

Man braucht wie auch bei linearer Regression ein Formel für das Modell. Diese bildet man analog wie bisher, indem mal die abhängige Variable mit den unabhängigen Variablen über eine Tilde verbindet. Im Fall der dritten Fragestellung aus Kapitel 3.1 wählen die also die Formel $prime \sim money$.

Der gesamte R-Code für die logistische Regression lautet also wiefolgt:

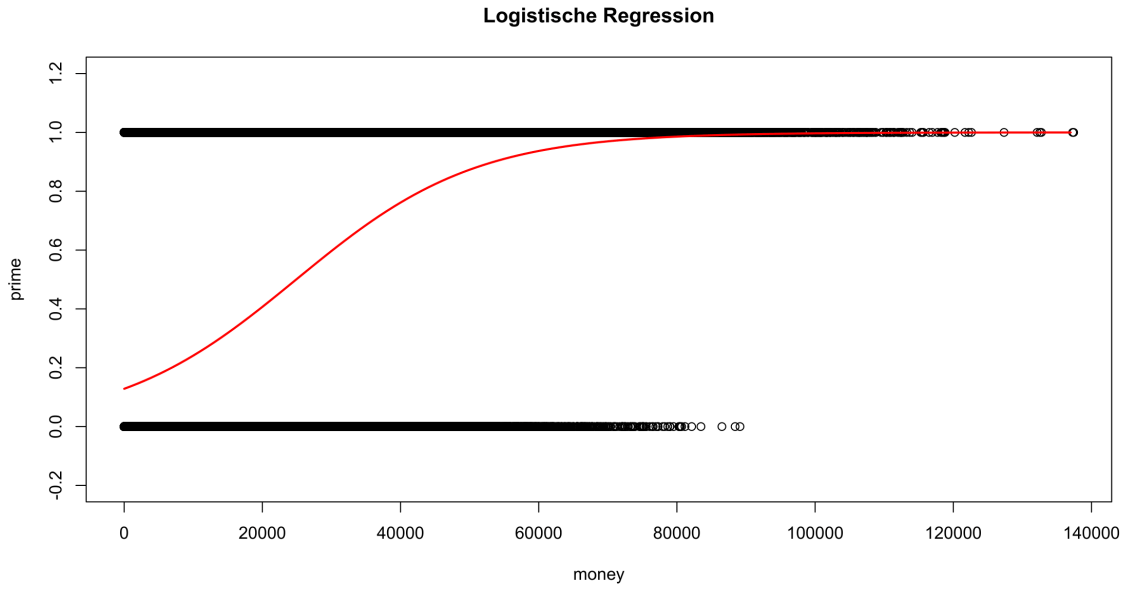
```
data <- read.csv2("sample.csv", sep = ",", header = TRUE)
modell <- as.formula("prime ~ money")
logit <- glm(modell, family = binomial, data = data)
print(logit)
```

Nach der Ausführung erhält man das folgende Ergebnis:

```
Coefficients:
(Intercept)        money
-1.9145608    0.0000769

Degrees of Freedom: 99999 Total (i.e. Null);  99998 Residual
Null Deviance:              138600
Residual Deviance: 100500      AIC: 100500
```

Eine anschauliche Interpretation der zurückgegebenen Parameter ist nicht mehr so einfach. Wir lassen uns das Ergebnis daher wieder als Plot visualisieren:



Für Kunden, die weniger als 100 Euro ausgegeben haben ist die Wahrscheinlichkeit Prime-Mitglied zu sein mit etwa 25% relativ gering. Je höher die Summe aber wird, desto größer wird auch diese Wahrscheinlichkeit. So ist ein Kunde mit mehr als 800 Euro Ausgaben so gut wie immer ein Prime-Mitglied.

3.3. Regression in Tensorflow

3.3.1. Einfache lineare Regression

3.3.2. Multiple lineare Regression

3.3.3. Logistische Regression

3.4. Regression in SQL

3.4.1. Einfache lineare Regression

3.4.2. Multiple lineare Regression

3.4.3. Logistische Regression

4. Statistische Methoden in Datenbanken

Here starts the thesis with an introduction. Please use nice latex and bibtex entries [1]. Do not spend time on formating your thesis, but on its content.

4.1. Latex Introduction

There is no need for a latex introduction since there is plenty of literature out there.

5. Erweiterungspotenzial in Datenbanksystemen

Here starts the thesis with an introduction. Please use nice latex and bibtex entries [1]. Do not spend time on formating your thesis, but on its content.

5.1. Latex Introduction

There is no need for a latex introduction since there is plenty of literature out there.

6. Fazit

Here starts the thesis with an introduction. Please use nice latex and bibtex entries [1]. Do not spend time on formating your thesis, but on its content.

6.1. Latex Introduction

There is no need for a latex introduction since there is plenty of literature out there.

Appendix

A. Detailed Descriptions

Here come the details that are not supposed to be in the regular text.

Bibliography

[1] Leslie Lamport. *LaTeX : A Documentation Preparation System User's Guide and Reference Manual*. Addison-Wesley Professional, 1994.