DAV

DEUTSCHE
AKTUARVEREINIGUNG e.V.

Überarbeitete Version vom 04.09.2022.

Schriftliche Prüfung im Spezialwissen

Actuarial Data Science Advanced

gemäß Prüfungsordnung 4
der Deutschen Aktuarvereinigung e. V.

am 24.10.2020

Hinweise:

- Als Hilfsmittel ist ein Taschenrechner zugelassen.
- Die Gesamtpunktzahl beträgt 180 Punkte. Die Klausur ist bestanden, wenn mindestens 90 Punkte erreicht werden.
- Bitte prüfen Sie die Ihnen vorliegende Prüfungsklausur auf Vollständigkeit. Die Klausur besteht aus 10 Seiten.
- Alle Antworten sind zu begründen und bei Rechenaufgaben muss der Lösungsweg ersichtlich sein.

Mitglieder der Prüfungskommission:

Axel Kiermaier, Dr. René Külheim, Dr. Stefan Nörtemann,
Tobias Renner, Mareike Welter

Aufgabe 1. *[Data Mining (4.1) & Datenschutz (1.2)] [36 Punkte]*

Im Rahmen der wertorientierten Unternehmenssteuerung möchte die Abteilung „Risikomanagement" der *Moderne Zeiten Leben AG* (MZL AG) mit Sitz in Köln die Modellierung ihres Unternehmensmodells verbessern. Mit den bislang verwendeten Stornotafeln, die die Stornowahrscheinlichkeit in Abhängigkeit vom Alter der versicherten Person sowie der vergangenen Laufzeit des Versicherungsvertrages angaben, konnte das Stornoverhalten bislang nur unzureichend vorhergesagt werden. Der Vorstand der MZL AG beschließt daher ein Data Mining Projekt zur Prognose des Stornoverhaltens der Kunden und beauftragt Sie mit der Leitung des Projekts.

(a) *[10 Punkte]* Nennen Sie die Prozessschritte des *Cross Industry Standard Process for Data Mining* (CRISP-DM) und erläutern Sie jeden der Schritte jeweils in einem Satz.

(b) *[10 Punkte]* Welche Daten bzw. Arten von Daten würden Sie für die Analyse des Stornoverhaltens heranziehen und aus welchen Quellen erhalten Sie diese Daten?

Zeitsprung: Im Verlauf des Projekts ist es Ihrem Team gelungen, ein Verfahren des überwachten maschinellen Lernens zu trainieren, das die Stornowahrscheinlichkeiten in Ihrem Versichertenbestand vorhersagt.

(c) *[9 Punkte]* Nennen Sie drei Kriterien oder Kategorien für die Evaluierung der Vorhersagen. Erläutern Sie kurz in zwei Sätzen, wie Sie die Vorhersagen Ihres Modells evaluieren.

(d) *[7 Punkte]* Gemäß Artikel 6 der Datenschutzgrundverordnung ist die Verarbeitung personenbezogener Daten verboten. Begründen Sie, warum bzw. unter welchen Umständen Sie diese Daten für Ihre Stornoprognose dennoch verwenden durften.

Lösungsvorschlag:

(a) Geschäftsverständnis, Datenverständnis, Datenvorbereitung, Modellierung, Evaluierung, Bereitstellung / Anwendung.

Im ersten Schritt (Geschäftsverständnis) geht es um die Bewertung des Umfelds und die Festlegung von Geschäftszielen, aus denen dann Data-Mining Ziele sowie ein Projektplan abgeleitet wird.

Ziel des zweiten Schritts (Datenverständnis) ist das Sammeln, Beschreiben und Untersuchen der notwendigen Daten sowie die Überprüfung der Datenqualität.

Im dritten Schritt (Datenvorbereitung) werden die notwendigen Daten ausgewählt, bereinigt, formatiert, konsolidiert und integriert.

Im vierten Schritt (Modellierung) wird das Analyse- oder Lernverfahren ausgewählt, umgesetzt, kalibriert und auf die Problemstellung angewendet.

Der fünfte Schritt (Evaluierung) dient der Überprüfung, Evaluierung und Interpretation der Modelle und Modellergebnisse.

Im sechsten und letzten Schritt (Bereitstellung / Anwendung) werden die Ergebnisse bereitgestellt und in die produktive / operative Anwendung überführt.

(b) **Interne Daten**: Da wir Storno analysieren und prognostizieren möchten, benötigen wir Daten zu möglichst allen Versicherungsverträgen, die in der Vergangenheit storniert wurden. Weil wir a priori nicht wissen, welche Anlässe und Ursachen es für einen Rückkauf geben könnte, benötigen wir eigentlich alle Daten, die wir über die stornierten Verträge und über die Versicherungsnehmer (und ggf. die versicherte Personen) bekommen können, denn alles könnte relevant für den Rückkauf gewesen sein. In der Praxis werden wir uns mit jenen Informationen zufriedengeben müssen, die in einem Bestandsführungssystem und ggf. in einer Partnerdatenbank erfasst und verarbeitet wurden.

Ebenso wichtig sind jene Verträge, die gerade nicht storniert wurden. Auch hier können erfasste Informationen relevant für die Frage sein, warum ein Vertrag nicht storniert wurde.

Externe Daten: Schließlich gibt es eine Vielzahl externer Informationen, die ggf. ursächlich für einen Rückkauf waren, wie zum Beispiel eine Hochzinsphase (weil Versicherungsnehmer mit einem anderen Investment eine höhere Rendite zu erreichen glaubten.)

(c) **Genauigkeit** (ist ein Maß dafür, wie gut das Modell ein Ergebnis mit den Attributen in den Daten korreliert).

Zuverlässigkeit (bewertet die Art und Weise, wie ein Data-Mining-Modell auf verschiedenen Datensätzen funktioniert.) Ein Data-Mining-Modell ist zuverlässig, wenn es unabhängig von den gelieferten Testdaten die gleiche Art von Vorhersagen erzeugt oder die gleichen allgemeinen Arten von Mustern findet.

Nützlichkeit (hängt vom fachlichen Kontext ab.) Zentral ist hierbei die Frage, ob die Ergebnisse sich geeignet verallgemeinern lassen.

Eine Möglichkeit der Prüfung der Prognose ist der sogenannte **Backtest**. Beim Backtest wird Data Mining / ML-Verfahren mit Daten der Vergangenheit „gefüttert" und dann die Prognose mit den beobachteten Daten verglichen.

(d) In Artikel 6 DSGVO sind diverse Ausnahmen von dem Verbot der Verarbeitung personenbezogener Daten aufgeführt. Die Verarbeitung ist erlaubt, wenn sie der Erfüllung einer rechtlichen Verpflichtung dient. Und ebenso, wenn sie der Wahrung der berechtigten Interessen des Verantwortlichen oder eines Dritten dient.

Hier müssen wir nun den Zweck der Unternehmensmodellierung betrachten: Dient die Vorhersage der Stornowahrscheinlichkeit der Unternehmensmodellierung zum Beispiel im Kontext einer Solvency II Berechnung, so handelt es sich um eine rechtliche Verpflichtung. Geht es jedoch zum Beispiel um die Berechnung des Market Consistent Embedded Values (MCEV), so können die berechtigten Interessen des Versicherungsunternehmens angeführt werden.

Unabhängig davon können / sollten die Daten anonymisiert oder pseudonymisiert für obige Zwecke verarbeitet werden.

Aufgabe 2 [Modellselektion und Regularisierung (3.4), *Data Mining (4.1) & 4.2 Analytics (4.1)] [40 Punkte]*

Die Reserven der Komposit-Sparte in Ihrer Versicherung werden bisher unter Anwendung der klassischen Chain-Ladder Methoden ermittelt. Zur Optimierung der Best Estimate-Reservierung sollen für Personenschäden in der KFZ-Haftpflicht-Versicherung die Reserven zukünftig durch eine Einzelfall-Reservierung ermittelt werden. Hierzu haben Sie die Aufgabe die Einzelfall-Reservierung - d.h. die Prognose der Einzelschadenhöhe - unter Anwendung von Data Science Methoden umzusetzen.

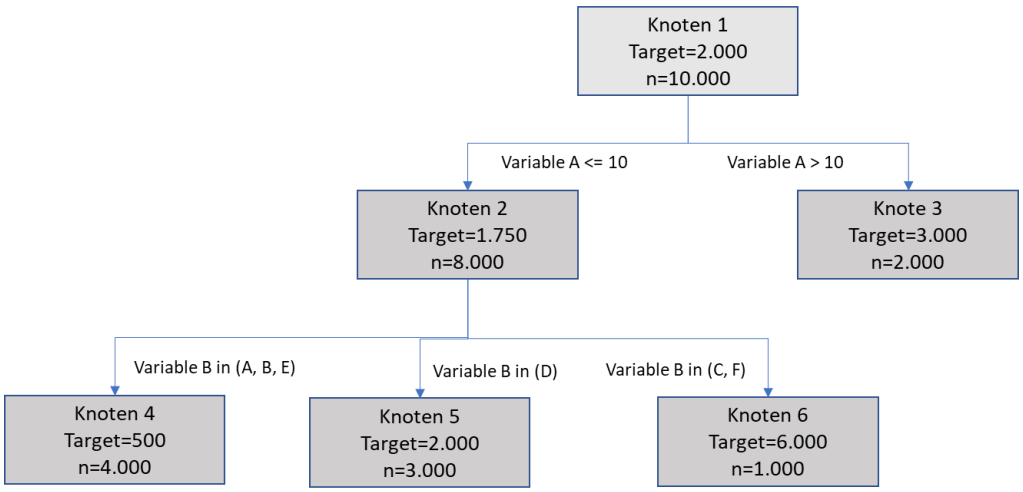
Als Datengrundlage stehen Ihnen **n = 10.000** Datensätze mit einer großen Anzahl **p** von Einflussfaktoren zur Verfügung.

a.) [5 Punkte] In einer ersten Analyse wurde ein multiples lineares Regressionsmodell erstellt. Bei der Überprüfung der Ergebnisse hat sich gezeigt, dass bei dem Modell ein Overfitting vorliegt. Erläutern Sie, was Overfitting bedeutet und wie ein Overfitting erkannt werden kann.

b.) [7 Punkte] Als Verlustfunktion in dem Regressionsmodell wurde der mittlere quadratische Fehler verwendet. Zur Verringerung des Overfitting soll eine Regularisierung unter Verwendung der Ridge Regression Methoden erfolgen. Benennen Sie die Verlustfunktion der Ridge Methode unter Verwendung des Parameters λ. Was ist die Bedeutung des Parameters λ?

c.) [6 Punkte] Bei der Erstellung des Modells sollen die Daten aufgeteilt werden. Beschreiben Sie das Vorgehen zur Trennung der Daten und erläutern Sie wie der Parameter λ bestimmt wird.

Neben dem Regressionsmodell soll ein Entscheidungsbaum erstellt werden. In der folgenden Grafik ist ein Teil des Entscheidungsbaums dargestellt:



d.) [5 Punkte] Erläutern Sie einen Vorteil von Entscheidungsbäumen gegenüber einem Regressionsmodell. Beschreiben Sie den Vorteil unter Berücksichtigung des dargestellten Entscheidungsbaums.

e.) [8 Punkte] Zusätzlich zu dem Entscheidungsbaum soll ein Random Forest mit einer Bagging Methode erstellt werden. Beschreiben Sie die Funktionsweise eines Random Forest und erläutern Sie einen Vorteil und einen Nachteil gegenüber einem Entscheidungsbaum.

f.) [9 Punkte] Als weiteres Modell soll ein Random Forest mit einer Boosting Methode erstellt werden. Hierbei soll die Methode AdaBoost (Adaptive Boosting) verwendet werden. Beschreiben Sie die verschiedenen Schritte zur Erzeugung des Random Forest mit der Boosting Methode.

Lösungsvorschlag:

(a) Bei einem Overfitting passt sich des Modells zu stark an den Daten an, auf denen es trainiert wurde. Das Overfitting beeinflusst negativ die Fähigkeit des Modells zur korrekten Vorhersage von neuen Daten.

Zur Erkennung von einem Overfitting werden die zugrundeliegenden Daten in Trainings- und Testdaten aufgeteilt und die Performance des Modells werden mit beiden Datenmengen überprüft. Sollte die Performance des Testdaten (viel) geringer ausfallen im Vergleich zu den Trainingsdaten, dann liegt ein Overfitting vor.

(b) Die Verlustfunktion des mittleren quadratischen Fehlers für eine lineare Regression ist wie folgt (mit M Datensätzen, P Einflussfaktoren und w_j Koeffizienten der Regression)

$$\sum_{i=1}^{M}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{M}(y_i - \sum_{j=0}^{p}(w_j - x_{ij}))^2$$

Durch die Ridge Regression wird die Verlustfunktion durch einen zusätzlichen (Bestrafungs-) Term erweitert, wie folgend dargestellt:

$$\sum_{i=1}^{M}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{p} w_j^2$$

Durch den zusätzlichen Term in der Verlustfunktion wird die Höhe der Koeffizienten beschränkt. Durch den Parameter λ wird die Höhe der Bestrafung der Koeffizienten (im Rahmen der Modellerstellung) beeinflusst. Desto größer der Parameter λ ist, desto größer ist die Bestrafung und hierdurch wird der Wert der Koeffizienten w_j beschränkt.

(c) Die Trennung der Daten und Bestimmung des Parameter λ kann wie folgt erfolgen:

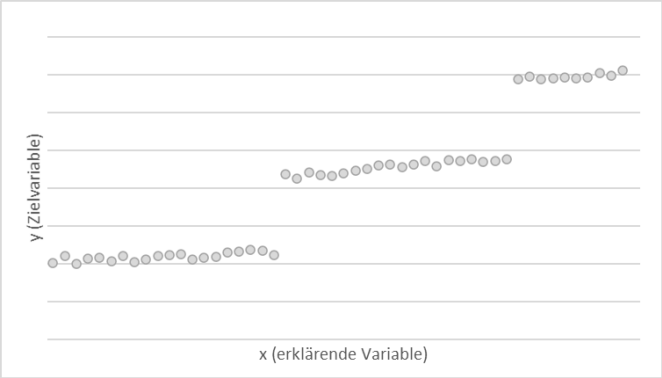
Die Daten (n=10.000) werden in Trainings-, Validation- und Testdaten aufgeteilt (z.B. im Verhältnis 60% / 20% / 20%). Die Daten werden anschließend zu folgenden Zwecken verwendet:

- Trainingsdaten: Die Trainingsdaten werden verwendet um das Modell zu trainieren.
- Validationdaten: Die Validationdaten werden verwendet um die Parameter des Modells zu bestimmten. Hierzu wird das Modell mehrfach mit den Trainingsdaten trainiert und unter Verwendung der Validationdaten werden die (optimalen) Parameter für das Modell bestimmt.
- Testdaten: Die Testdaten werden zu einer unverzerrten Bewertung des finalen Modells verwendet.

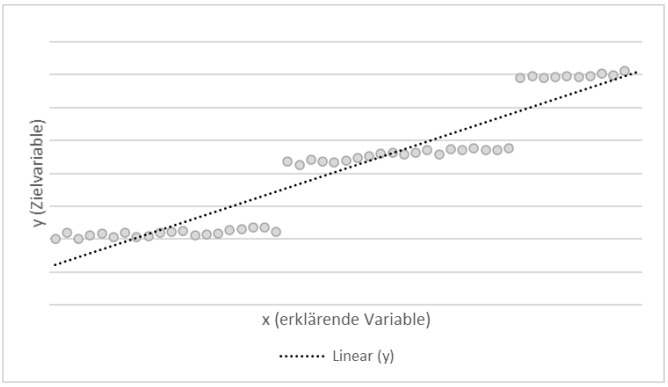
Der Paramater λ wird für das Regressionsmodell für den vorliegenden Anwendungsfall unter Verwendung der Trainings- und Validationdaten so bestimmt, damit die Modellperformance optimiert wird.

Alternative Antworten / Varianten zur Datentrennung und Bestimmung des Parameters λ sind möglich.

(d) Durch Entscheidungsbäume können nicht lineare Zusammenhänge erkannt werden, die in der Regressionsanalyse nicht ermittelt werden können. Dies wird an der folgenden Grafik veranschaulich, in dem ein Zusammenhang zwischen der erklärenden Variablen x und der Zielvariable y gesucht werden soll:



Durch eine Regression – z.B. einer linearen Regression - kann nur der lineare Zusammenhang zwischen x und y modelliert werden, wie folgend dargestellt:



Aus Grafik ist ersichtlich, dass der Zusammenhang zwischen x und y unzureichend modelliert wird.

Durch einen Entscheidungsbaum können durch die Wahl der Splits die nicht linearen Zusammenhänge modelliert werden. Dies ist folgend dargestellt:

In dem dargestellten Entscheidungsbaum verringert sich der Target-Wert von dem Knoten 1 zu dem Knoten 2 von 2.000 auf 1.750. In dem nachfolgenden Split erhöht sich der Target-Wert in Knoten 6 auf 6.000. Der Knoten 6 hat somit den höchsten Target-Wert im gesamten Baum, der durch einen nicht linearen Zusammenhang erkannt wurde.

(e) Ein Random Forest ist eine Ensemble Methode und wird aus einer Menge von Entscheidungsbäumen gebildet. Die einzelnen Entscheidungsbäumen werden mit einem Bootstrap-Datensatz erzeugt, der durch das zufällige Ziehen (mit Zurückliegen) aus der Datengrundlage gebildet wird. Die Vorhersage der einzelnen Bäume werden zu einer Gesamtvorhersage aggregiert.

Ein Vorteil von einem Random Forest im Vergleich zu einem einzelnen Entscheidungsbaum ist, dass die Gefahr des Overfitting reduziert werden. Ein einzelner Entscheidungsbaum kann in der Vorhersage eine hohe Varianz haben und hierdurch entsteht die Gefahr des Overfitting. Durch die Generierung von verschiedenen Bäumen wird in einem Random Forest die Varianz reduziert und somit die Gefahr des Overfitting reduziert.

Ein Nachteil von einem Random Forest im Vergleich zu einem einzelnen Entscheidungsbaum ist der Verlust der Interpretierbarkeit. Bei einem einzelnen Entscheidungsbaum können die Vorhersagen über den Blättern des Baumes und deren Pfaden interpretiert und leicht verständlich erläutern werden. Bei einem Random Forest ist die Interpretation, auf Grund der Vielzahl von Bäumen, nicht möglich.

Weitere Antworten zu den Vor- und Nachteilen eines Random Forest sind möglich.

(f) Die Erstellung des Random Forest mit der AdaBoost Methode erfolgt in den nachfolgenden Schritten. Hierbei sollen N Entscheidungsbäume erzeugt werden.

- Schritt 0: Festlegung der initialen Gewichte w_i pro Datenpunkt i mit $w_i = \frac{1}{N_{ges}}$ (mit N_{ges} Datenpunkten in den Trainingsdaten).
- Schritt 1: Training des j-ten Entscheidungsbaums (mit $j \in N$ beginnend mit j = 1).
- Schritt 2: Ermittlung der gewichteten Fehlerrate e_j für den j-ten Entscheidungsbaum. Hierbei wird die Fehlerrate jedes Datenpunkt mit w_i gewichtet.
- Schritt 3: Ermittlung der Gewichtung des Entscheidungsbaums w_tree_j in dem Ensemble (Random Forest):

  $$w\_tree_j = \eta * \log\left(\frac{1-e_j}{e_j}\right)$$

  (hierbei Verwendung der Learning Rate η als Parameter im Modell.)
- Schritt 4: Update der Gewichte w_i pro Datenpunkte i

  $$w_i = \begin{cases} w_i & falls\ y_i\ korrekt\ vorhergesagt\ wurde \\ w_i * e^{w\_tree_j} & falls\ y_i\ nicht\ korrekt\ vorhergesagt\ wurde \end{cases}$$
- Schritt 5: Wiederholung von Schritt 1 bis 4 bis die maximale Anzahl N von Entscheidungsbäume erzeugt wurde.
- Schritt 6: Ermittlung der finalen Vorhersage in dem Ensemble Modell (Random Forest) unter Berücksichtigung der Gewichtung w_tree_j zu jedem Entscheidungsbaum.

Aufgabe 3. *[Regressions- und Clustermethoden 2 (3.1)] [36 Punkte]*

(a) [10 Punkte] Im Rahmen von Klassifikationsverfahren beschäftigen Sie sich mit Support Vector Machines (SVMs). Beschreiben Sie kurz die generelle Funktionsweise von SVMs (Hinweis: Dabei können Sie sich auf den Fall eines linearen Kernels, auch mit Support Vector Classifier bezeichnet, beschränken. Gehen Sie zudem davon aus, dass die Trainingsdaten in die beiden Klassen „-1" und „+1" aufgeteilt sind).

(b) [16 Punkte] Der Maximal Margin Classifier ist die Lösung des folgenden Optimierungsproblems:

1. $Maximiere_{b,\omega_1,\ldots,\omega_p} M$ mit $\sum_{j=1}^{p} \omega_j^2 = 1$
2. $y_i(b + \omega_1 x_{i1} \ldots + \omega_p x_{ip}) \geq M$ für alle $1 \leq i \leq n$

Sie betrachten den folgenden (einfachen) Datensatz:

i	x_{i1}	x_{i2}	y_i
1	2	0	-1
2	3	-1	-1
3	3	2	1

Begründen Sie, warum Sie in diesem Fall den Maximal Margin Classifier berechnen können und berechnen Sie diesen (Hinweis: Nutzen Sie aus, dass alle angegebenen Punkte Stützvektoren darstellen). Zu welcher Klasse gehört die Beobachtung (3,3)?

(c) [10 Punkte] Gegeben sei der polynomiale Kernel

K(x,z) = $(< x, z > +\vartheta)^d$ mit $d = 2, \vartheta = 1$ *sowie* $x, z \in \mathbb{R}^2$.
Hierbei bezeichnet $< x, z > = \sum_{j=1}^{m} x_j z_j$ für $x, z \in \mathbb{R}^m$.

Zeigen Sie: $K(x,z) =< \phi(x), \phi(z) >$ mit $\phi(x) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$

Erklären Sie kurz die Idee hinter der Verwendung eines (nichtlinearen) Kernels bei den SVM und erweitern Sie damit die Beschreibung aus (a) zu generellen SVMs.

Lösungsvorschlag:

(a) Die Trainingsdaten bestehen aus den Daten $(x_1, y_1), \ldots, (x_n, y_n)$ mit $x_i \in \mathbb{R}^p$ und $y_i \in \{-1,1\}$. Bei Support Vector Machines (mit linearem Kernel auch als „Support Vector Classifier" oder „Soft margin classifier" bezeichnet) wird der Merkmalsraum mit Hilfe einer Hyperebene in zwei Klassen unterteilt (wobei die beiden Hälften mit „-1" und „+1" bezeichnet werden können). Dabei wird die Hyperebene so gewählt, dass der Abstand zu den Beobachtungen, die dieser am nächsten liegen, maximiert wird. Damit ist die Hyperebene von diesen Beobachtungen, die auch als „support vectors" bezeichnet werde, abhängig.

In realen Datensätzen wird keine exakte Trennung durch eine Hyperebene möglich sein, daher werden Verletzungen des „Randes" (der durch support vectors bestimmt wird) bzw. der korrekten Klassifikation in Kauf genommen.

Letztlich ist hierfür ein Optimierungsproblem zu lösen, welches in folgender Form formuliert werden kann:

1. $Maximiere_{b,\omega_1,\ldots,\omega_p} M$ mit $\sum_{j=1}^{p} \omega_j^2 = 1$
2. $y_i(b + \omega_1 x_{i1} \ldots + \omega_p x_{ip}) \geq M(1 - \varepsilon_i)$ für alle $1 \leq i \leq n$
3. $\varepsilon_i \geq 0, \sum_{i=1}^{n} \varepsilon_i \leq C$

C bestimmt dabei den Grad der zulässigen Verletzung des Randes bzw. der Hyperebene.

(b) Aus Abbildung 1 ist ersichtlich, dass sich die Punkte linear trennen lassen. Damit existiert der Maximal Margin Classifier. Im Folgenden wird x_{i1} mit x sowie x_{i2} mit y bezeichnet, wobei sich i aus dem Kontext ergibt.

Die untere Rand-Hyperebene ergibt sich damit aus den beiden Punkten der Klasse „-1" zu $H_-: y + x - 2 = 0$ und damit die obere zu $H_+: y + x - 5 = 0$.

Die trennende Hyperebene liegt in der Mitte dieser beiden, und lautet damit (in normierter Form): $H: \frac{1}{\sqrt{2}} \left[< \binom{1}{1}, \binom{x}{y} > - 3.5 \right] = 0$

Einsetzen von x=3 und y=3 in H ergibt H > 0. Damit gehört die Beobachtung (3,3) zur Klasse „+1".

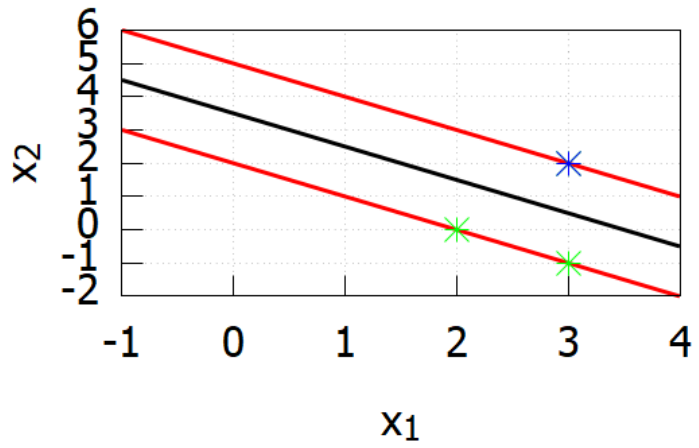


Abbildung 1: Der angegebene Datensatz mit den berechneten Hyperebenen

(c) Ausmultiplizieren von $K(x,z) = (< x,z > +1)^2$ ergibt

$x_1^2 z_1^2 + x_2^2 z_2^2 + 1 + 2x_1 x_2 z_1 z_2 + 2x_1 z_1 + 2x_2 z_2$.

Das ist auch das Ergebnis der Berechnung von $< \phi(x), \phi(z) >$.

Der „Support Vector Classifier" ist durch seine lineare Hyperebene eingeschränkt. Um einen flexibleren Grenzverlauf erreichen zu können, wird der Merkmalsraum in einen höherdimensionalen Raum mittels einer Mappingfunktion (hier mit $\phi(x)$ bezeichnet) transformiert. In diesem wird eine lineare Trennung vorgenommen.

In die SVMs gehen diese Transformationen über den Kernel K ein. Damit werden weit flexiblere Grenzverläufe als im linearen Fall möglich.

Aufgabe 4. *[No-Sql-Datenbanken (2.1)] [30 Punkte]*

Im Rahmen einer Konzepterstellung für eine Pandemie-App wird der Einsatz einer Graphendatenbank diskutiert. Die App soll auf Smartphones installiert werden und protokollieren, ob und wann andere Geräte mit ebenfalls installierter App in räumlicher Nähe geortet wurden. Die Speicherung der Daten soll stark verdichtet zentral in der oben genannten Datenbank erfolgen.

In den folgenden Codeausschnitten steht ein Objekt *graph* global im Zugriff zur Verfügung und repräsentiert die Verbindung zur Datenbank, die analog Neo4J funktioniert und über die Abfragesprache *Cypher* angesprochen werden kann.

Die Datenbank soll Personen als Knoten in einem Graphen vorhalten und Kontakte zwischen denselben als Kanten modellieren. Folgende Funktionen sind zunächst entworfen worden:

```
def create_person(id_, name, registered_at):
    """ Node für eine neue Person erstellen. """
    _QUERY_CREATE_SUSC = """MERGE (p:Susceptible { id_: {id_}, name: {name}, registered: {registered}})
                            RETURN p"""
    individual = graph.run(_QUERY_CREATE_SUSC, id_=id_, name=name,
                           registered=str(registered_at))
    return individual.next()["p"]


def add_contacts(id_, contacts):
    """ Kontakte der Person :id_ in die DB aufnehmen, der Parameter
        contacts ist eine Liste mit Tupeln des Formats (id_contact, day),
        wobei :id_contact die ID der Kontaktperson und :day (ein Integer)
        den laufenden Tag des Jahres bezeichnet.
        """

    # Transformation
    pars = dict()
    pars["contacts"] = [{"id_1": int(id_),
                         "id_2" : int(con[0]),
                         "day": int(con[1])} for con in contacts]

    # Das Statement führt eine Schleife über die Parameterliste durch, bestimmt
    # jeweils die betroffenen Personen-Knoten und fügt eine Kante
    # mit der Tagesinformation ein (wenn sie noch nicht existiert)
    _QUERY_ADD_CONTACTS = """UNWIND $contacts AS map
                             MATCH (p1:Susceptible{id_:map.id_1}), (p2:Susceptible{id_:map.id_2})
                             MERGE (p1)-[:CONTACT{at:map.day}]->(p2) """
    graph.run(_QUERY_ADD_CONTACTS, pars)
```

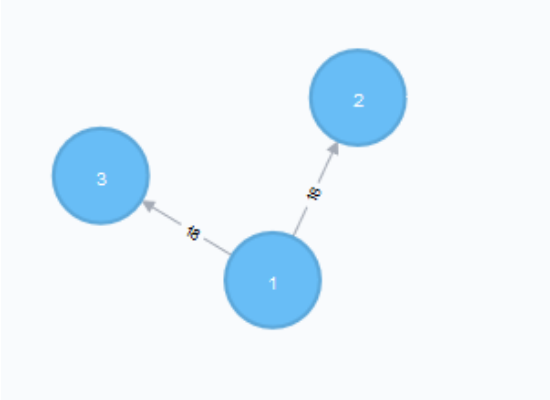
Hinweis: Die `MERGE`-Anweisung stellt sicher, dass ein Muster im Graphen vorhanden ist. Ist es noch nicht vorhanden, wird es neu erzeugt.

Die Ausführung der Anweisungen

```
new_person1 = create_person(id_=1, name="Hajo K", registered_at=datetime.datetime(year=2020, month=2, day=26))
new_person2 = create_person(id_=2, name="Jochen V", registered_at=datetime.datetime(year=2020, month=3, day=6))
new_person3 = create_person(id_=3, name="Karin M", registered_at=datetime.datetime(year=2020, month=3, day=16))

add_contacts(1, [(2, 18), (3, 18)])
```

führt dann zu folgendem Graphen:



(a) *[14 Punkte]* Erläutern Sie welche verschiedenen Knoten- und Kantenlabels verwendet werden und geben sie jeweils für die Attribute und deren Typen an. Zeichnen Sie den Graphen nach Ausführung der folgenden Anweisungen!

```
new_person4 = create_person(id_=4, name="Falk B", registered_at=datetime.datetime(year=2020, month=2, day=20))
new_person5 = create_person(id_=5, name="Rebekka L", registered_at=datetime.datetime(year=2020, month=2, day=20))

add_contacts(1, [(2, 19)])
add_contacts(3, [(4, 20), (4, 23)])
```

Es sollen Übertragungsrisiken identifiziert werden, die aus den Kontakten entstehen, wobei die zeitliche Dimension zunächst ignoriert wird. Hierzu liegen folgende Abfragen vor:

```
# direkte Kontakte, bidirektional
graph.run('MATCH (A)-[CONTACT]-(B) WHERE A.id_ = {id_param} RETURN distinct B', id_param=1).data()
```

```
[{'B': (f096f12:Susceptible {id_:2,name:"Jochen V",registered:"2020-03-06 00:00:00"})},
 {'B': (a746581:Susceptible {id_:3,name:"Karin M",registered:"2020-03-16 00:00:00"})}]
```

```
# transitive Kontakte, bidirektional
graph.run('MATCH (A)-[CONTACT*]-(B) WHERE A.id_ = {id_} and B.id_ <> {id_} RETURN distinct B', id_=1).data()
```

```
[{'B': (a746581:Susceptible {id_:3,name:"Karin M",registered:"2020-03-16 00:00:00"})},
 {'B': (eb00326:Susceptible {id_:4,name:"Falk B",registered:"2020-02-20 00:00:00"})},
 {'B': (f096f12:Susceptible {id_:2,name:"Jochen V",registered:"2020-03-06 00:00:00"})}]
```

(b) *[8 Punkte]* Erklären Sie umgangssprachlich, was die erste Abfrage erreichen will. Identifizieren Sie dann die Unterschiede zwischen der ersten und der zweiten Abfrage und geben sie begründete Mutmaßungen darüber ab, was sie bewirken sollen!

Zur Bestimmung von Infektionsrisiken unter näherungsweiser Berücksichtigung der zeitlichen Abfolge ist folgendes Code-Fragment entwickelt worden:

```
# Startwerte
start_id = 1                # ID der Ausgangsperson der Iteration

contacts = set()            # 'Menge' der schon identifizierten Kontakte
contacts.add(start_id)      # Ausgangselement zur Menge hinzufügen

def contacts_at_day(current_contact, day):
    """ Identifiziert alle Kontakte, die die Personen mit den ids
        im Argument :current_contacts am Tag :day haben.
    """
    # die Parameter $day und $contacts werden bei der Ausführung
    # der folgenden Query mit übergeben und stehen in der Query zur Verfügung
    qry_contacts_day = """
        UNWIND $contacts AS contact_id
          MATCH (A)-[c:CONTACT* {at:$day}]-(B) WHERE A.id_ = contact_id
        RETURN B.id_ as id_
    """

    # query mit Parametern ausführen und nur ID-Felder aus dem Queryergebnis zurückgeben
    new_contacts = graph.run(qry_contacts_day, {"day": day, "contacts": list(contacts)}).data()
    # Rückgabe als Menge entfernt Duplikate
    return {cnt["id_"] for cnt in new_contacts}

# Aufruf der Funktion
new_contacts = contacts_at_day(contacts, day=18)
print("Neue Kontakte", new_contacts)

contacts.update(new_contacts)
print("Alle Kontakte:", contacts)
```

```
Neue Kontakte {2, 3}
Alle Kontakte: {1, 2, 3}
```

(c) *[8 Punkte]* Entwickeln Sie auf dieser Grundlage den (Pseudo-)Code einer Funktion

```
def get_all_contacts(start_id):
    """ Gibt IDs aller Personen zurück, die direkt oder indirekt einen
        nachgewiesenen Kontakt zur Startperson innerhalb der ersten 100
        Tage hatten."""
    # ...
```

die ausgehend von einer Start-ID einer infizierten Person alle Personen unter Risiko findet, mit der der/die Infizierte innerhalb der Tage 1-100 Kontakt hatte unter (näherungsweiser) Berücksichtigung der zeitlichen Abfolge der Kontakte. Geben Sie auch das Ergebnis an, das sich in der Konstellation des Beispiels von oben ergibt.

Lösungsvorschlag:

(a) Es gibt in dem Graphen ausschließlich Nodes mit dem Label *Susceptible* und Kanten mit dem Label *CONTACT*.

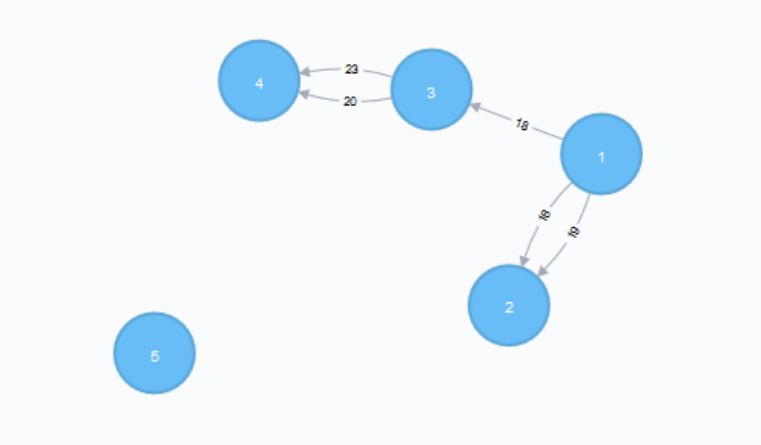
Attribute von *Susceptible:*

- *id_:* ganze Zahl
- *name:* Text
- *registered:* Text (gefüllt mit Datumsangabe)

Attribute von *CONTACT:*

- *at:* ganze Zahl

Nach den zusätzlichen Anweisungen hat der Graph folgendes Aussehen:



(b) Die erste Query erzeugt eine Liste von Knoten, die von einem Ausgangsknoten, der durch den Parameter *id_param* selektiert wird, durch eine Kante vom Typ CONTACT verbunden sind. Die Richtung der Kanten wird hierbei nicht berücksichtigt.

Die zweite Query unterscheidet sich in folgenden Punkten:

- Name des Parameters ist jetzt *id_* an Stelle von *id_param*, das ist belanglos
- Es wird eine Einschränkung gemacht: ein Knoten kann nur dann zur Ergebnismenge gehören, wenn er eine andere id hat als der Eingabeparameter *_id*. Dadurch schließt man genau diesen einen Knoten aus der Ergebnismenge aus
- Die Verbindung ist nun als [CONTACT*] an Stelle von [CONTACT] definiert. Es ist den Kommentaren zu entnehmen (und auch aus anderen Zusammenhängen in ähnlicher Form bekannt), dass das *-Symbol eine ganze Sequenz von CONTACT-Kanten parametrisiert.

Damit liefert die zweite Query eine Liste aller Knoten (außer dem Ausgangsknoten), die über beliebig viele Kanten vom Ausgangknoten erreicht werden können.

(c) Eine Möglichkeit, die gesuchte Funktion zu implementieren ist die folgende:

```
def get_all_contacts(start_id):
    """ Gibt IDs aller Personen zurück, die direkt oder indirekt einen
        nachgewiesenen Kontakt zur Startperson innerhalb der ersten 100
        Tage hatten."""

    contacts = set()              # 'Menge' der schon identifizierten Kontakte
    contacts.add(start_id)

    for day in range(101):
        new_contacts = contacts_at_day(contacts, day)
        contacts.update(new_contacts)

    contacts.remove(start_id)
    return contacts

start_id = 1                  # ID der Ausgangsperson der Iteration
print("Identifizierte Kontakte von der Person", start_id, "bis zum Tag 100 sind", get_all_contacts(start_id))
```

```
Identifizierte Kontakte von der Person 1 bis zum Tag 100 sind {2, 3, 4}
```

Aufgabe 5. *[No-Sql-Datenbanken (2.1)] [10 Punkte]*

Als Alternative zu den üblichen SQL-Datenbanken werden häufig *spaltenorientierten Datenbanken* eingesetzt. Erläutern Sie die Idee der sogenannten *spaltenorientierten Datenbanken*. Warum kann die zugrunde liegende Architektur Vorteile gegenüber gewöhnlichen (zeilenorientierten) Datenbanken sowohl in Bezug auf Flexibilität und Platzbedarf also auch in Bezug auf die Abarbeitungsgeschwindigkeit ergeben?

Lösungsvorschlag:

Spaltenorientierten Datenbanken speichern zweidimensional Tabellen spaltenweise:

Spalte A	**Spalte B**	**Spalte C**
1	AAA	T
2	BBB	T
3	CCC	F

Im Speicher/Disk ergibt das ein Layout wie folgt:

1	2	3		AAA	BBB	CCC		T	T	F

an Stelle des klassischen Layouts

1	AAA	T		2	BBB	T		3	CCC	F

Das Layout erlaubt ein leichteres Hinzufügen von Spalten, da diese in einem neuen Speicherbereich liegen und lediglich die Tabellendefinitions-Metadaten aktualisiert (und ggf. umkopiert werden müssen). Dadurch hat man eine höhere **Flexibilität**.

Das Layout kann **Speicherplatz einsparen**, da es möglich ist, die Daten in jeder Spalte separat zu komprimieren und durch die zu erwartende höhere Homogenität der Daten der einzelnen Spalten kann die Kompressionsrate deutlich höher ausfallen als bei zeilenweiser Speicherung.

Oft werden in Queries nur wenige Spalten abgefragt. Da von der Harddisk üblicherweise jedoch größere Blöcke gelesen werden (z.B. Kann die Mindestgröße mehrere Kilobyte betragen), kann das Auslesen einer einzigen Spalte einer zeilenweise gespeicherten Tabelle dazu führen, dass effektiv die ganze Tabelle vom Disk gelesen werden muss und entsprechend Energie und Zeit benötigt wird. Bei spaltenweiser Speicherung ergibt sich hier ein Vorteil, weil der Umfang nicht notwendiger Lesevorgänge deutlich reduziert werden kann.

Aufgabe 6. *[Data Mining (4.1)] [28 Punkte]*

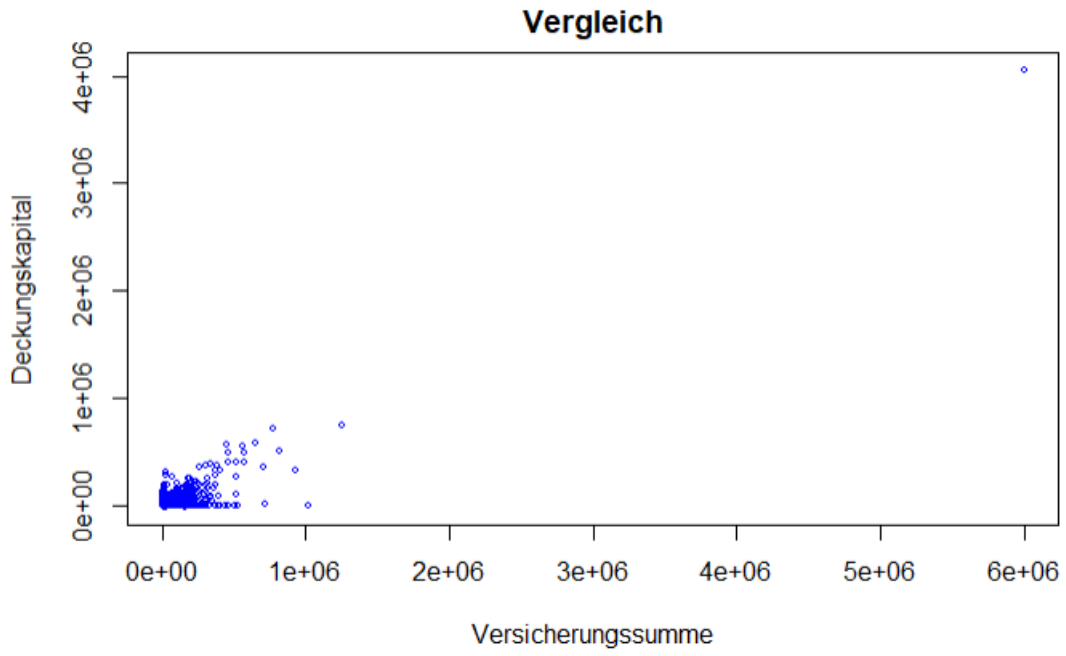
Sie sollen als Data Scientist in einem Data Mining Projekt Ihre Team-Kollegen speziell im Bereich Datenvisualisierung unterstützen. Der Kern der Daten ist ein Lebensversicherungsbestand. Im Projekt wird das CRISP-DM-Vorgehensmodell verwendet. Zur Vorbereitung stellen Sie zunächst – unabhängig von der eigentlichen Aufgabe – allgemeine Überlegungen an, wo und wie Sie Ihre Visualisierungskenntnisse optimal einbringen können, auch um Ihren Kollegen Anregungen zu geben, wann sie insbesondere auf Sie zukommen sollten.

(a) *[10 Punkte]* Überlegen Sie, in welchen Phasen und Aufgaben des Data Minings Datenvisualisierungen besonders gut eingesetzt werden können. Nennen Sie mindestens 3 verschiedene Phasen/Aufgaben. Welche Rolle spielt die Datenvisualisierung im Kontext der jeweiligen Aufgabe? Inwiefern hat die jeweilige Phase Auswirkungen darauf, worauf man bei der Datenvisualisierung besonders achten sollte? Welche Rolle spielt in dem Zusammenhang die Tatsache, dass der Data Mining Prozess – trotz aller Automatisierungsbemühungen – kein vollautomatischer Prozess ist? (Denken Sie dabei auch an die Definition von Data Mining!)

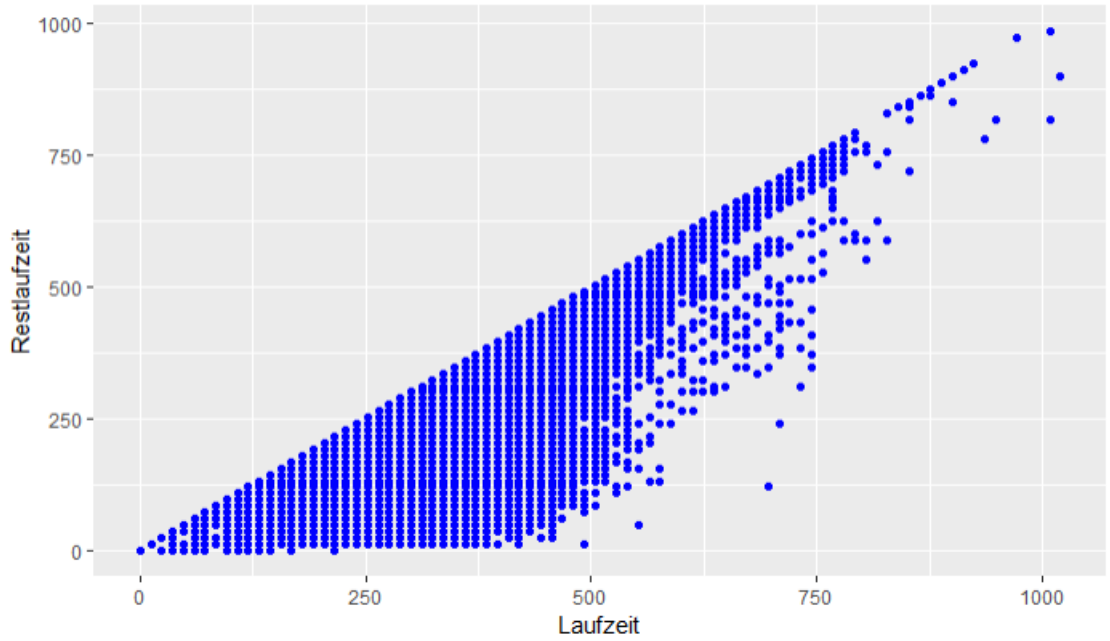
(b) *[6 Punkte]* Um Ihr Team optimal zu unterstützen, überlegen Sie zur weiteren Vorbereitung: Bei welchen Data Mining Anwendungsklassen (wie zum Beispiel *Klassifikation*) bewährt sich die Visualisierung von Daten besonders? Nennen Sie eine solche Anwendungsklasse und erläutern Sie dabei die Rolle der Datenvisualisierung. Generell: Was lässt sich gut durch Datenvisualisierung erkennen? Wodurch wird Visualisierung begrenzt und wie kann man mit dieser Begrenzung umgehen?

Der Versicherungsbestand besteht aus etwa 50.000 Verträgen und soll zur Beschleunigung verschiedener Berechnungen verdichtet werden. Zur Vorbereitung visualisieren Sie den Versicherungsbestand anhand verschiedener Kriterien.

(c) *[6 Punkte]* Dabei haben Sie folgende erste Grafik erhalten. Um was für eine Art von Grafik handelt es sich? Nennen Sie mindestens zwei Auffälligkeiten. Welche ersten Ideen haben Sie, mit diesen Auffälligkeiten umzugehen und so die Grafik zu verbessern („damit man mehr sieht")?



(d) *[6 Punkte]* Eine weitere Gegenüberstellung zeigt folgende Grafik:



Ist der Bestand eher jung oder eher alt? Wie könnten Sie diese Grafik so modifizieren, dass Sie diese Frage besser beantworten können? Woran würden Sie dann einen jungen Bestand erkennen? Wie könnten Sie diese Entscheidung jung / alt mit einem alternativen Visualisierungsansatz besser treffen?

Lösungsvorschlag:

(a) **Phasen/Aufgaben**:

Datenverständnis / Datenexploration bzw. Untersuchen von Daten → Beitrag zur Datenbeschreibung, Input für Datenvorbereitung/-transformation.

Datenverständnis / Überprüfen der Datenqualität → z.B. Visualisierung von Inkonsistenzen für den Datenqualitätsbericht.

Modellierung / Erstellen der Modelle → modellabhängig sind mehr oder weniger unterschiedliche Visualisierungen denkbar.

Modellierung / Bewerten des Modells → spezielle Grafiken zur Modellvalidierung, z.B. Lift, Profit, ROC, …

Bereitstellung / Abschlussbericht → Ergebnispräsentation.

Phasen-Abhängigkeit der Schwerpunkte bei der Visualisierung:

Zwei Hauptzwecke der Datenvisualisierung sind mithin das Entdecken von Mustern (was ja das Ziel des Data Minings ist) – von „first insights" bis zu komplexen Mustern – und die effiziente Kommunikation von Ergebnissen.

Steht die Kommunikation im Vordergrund (wie z.B. bei der Ergebnispräsentation), so spielt die leichte Verständlichkeit der Darstellung eine besondere Rolle. Diese lässt sich z.B. unterstützen durch: klare Beschriftung von Titel und Achsen, passende Skalierung der Achsen, eingängige und konsistente Farbgebung, ggf. eine Legende.

Kein vollautomatischer Prozess:

Der Mensch spielt eine entscheidende Rolle in vielen Phasen des Data Minings. Daher ist die Fähigkeit des Menschen wesentlich, Muster in Bildern zu erkennen (jedenfalls leichter als in „Datenfriedhöfen") und somit ist es wichtig, Daten so zu visualisieren, dass Muster erkennbar werden können.

(b) Segmentierung / Clusterbildung von Daten lässt sich häufig sehr gut grafisch darstellen. Datenvisualisierung kann hier der erste Schritt sein, um den Verdacht auf das Vorliegen einer Segmentierung zu formulieren und anschließend eine entsprechende Clusteranalyse aufzusetzen. Andererseits können auch die Ergebnisse einer Clusteranalyse in der Regel gut in grafischer Form an den Auftraggeber kommuniziert werden.

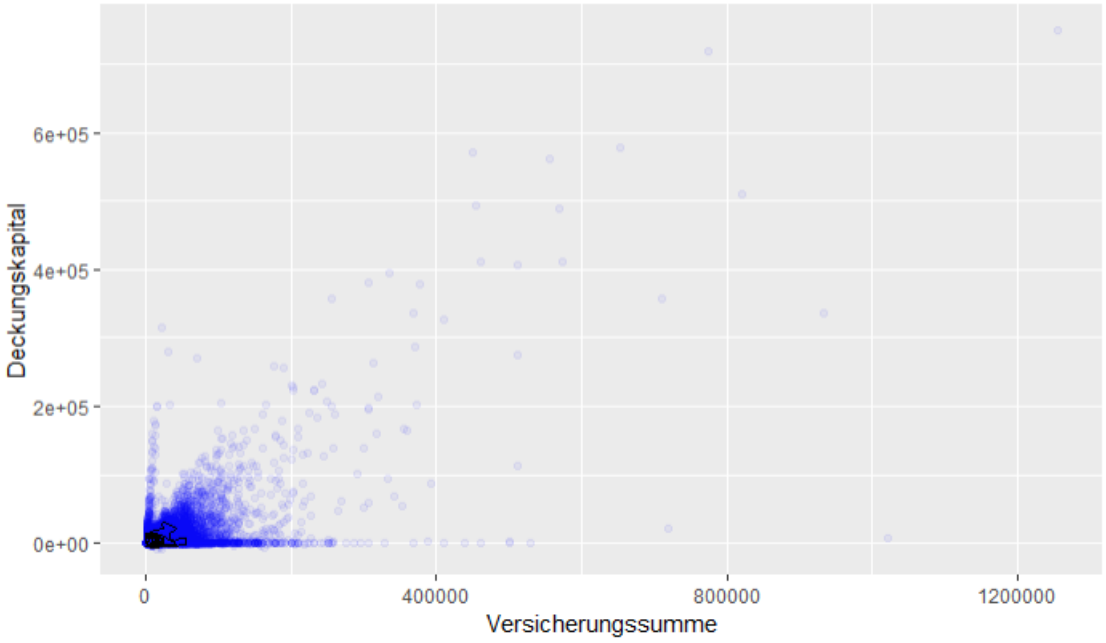
Gut erkennbar durch Visualisierung: Datenverteilung, Muster, Cluster, Ausreißer

Begrenzung der Visualisierung: Visualisierung ist nur sehr gut in 2D möglich; 3D ist schon schwierig, nur interaktiv (durch räumliches Drehen) gut möglich; höhere Dimensionen sind nur als Projektion erfassbar → Scattermatrix (in R: ggpairs()). Projektion in welcher Richtung? Ansatz z.B. für PCA.

(c) Die Grafik zeigt einen Scatterplot der Verteilung von Deckungskapital vs. Versicherungssumme. Ein Ausreißer dominiert die Darstellung. Durch Overplotting lässt sich nicht genau erkennen, wo die Datenpunkte „am Dichtesten" liegen.

Ansätze zur Verbesserung:

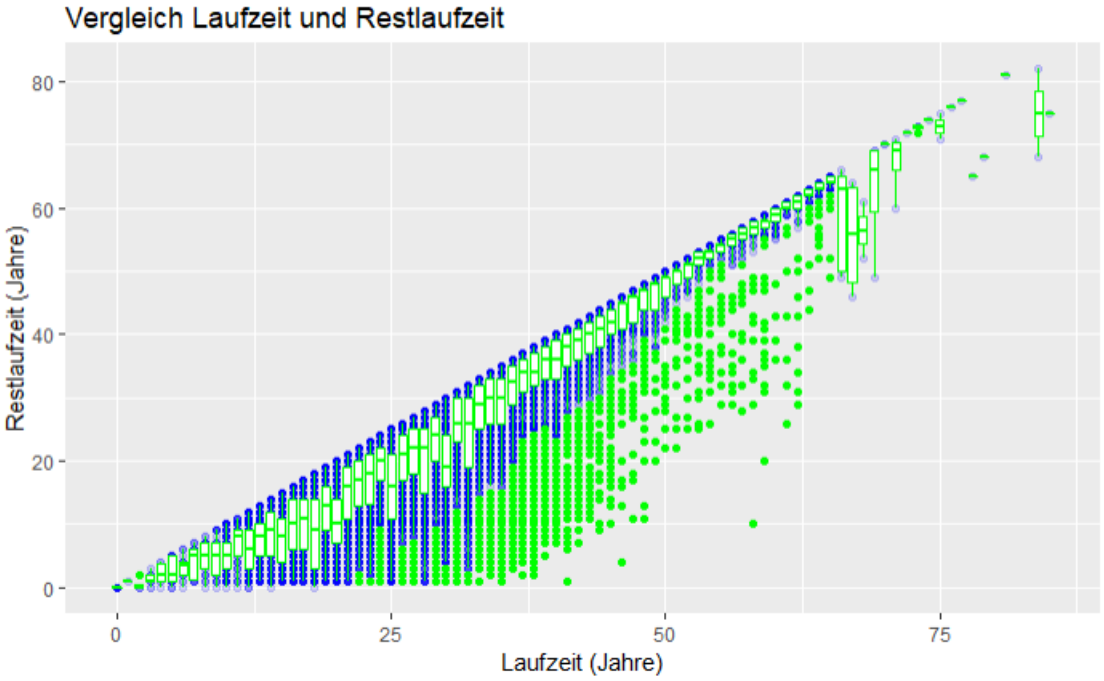
- Ausschluss des Ausreißers führt zu besserer Skalierung; die dichte Wolke wird etwas entzerrt. Für die weitere Verarbeitung muss der Vertrag natürlich wieder hinzugenommen werden.
- Generell weiteres „Reinzoomen".
- Änderung der Transparenz der Punkte (in R ggplot2: alpha). Einzelne Datenpunkte sind so von mehrfachen Punkten besser zu unterscheiden.
- Eventuell lässt sich durch zusätzliche Grafikelemente weitere Information abbilden, z.B. Dichte-Höhenlinien (in R ggplot2: geom_density_2d()).



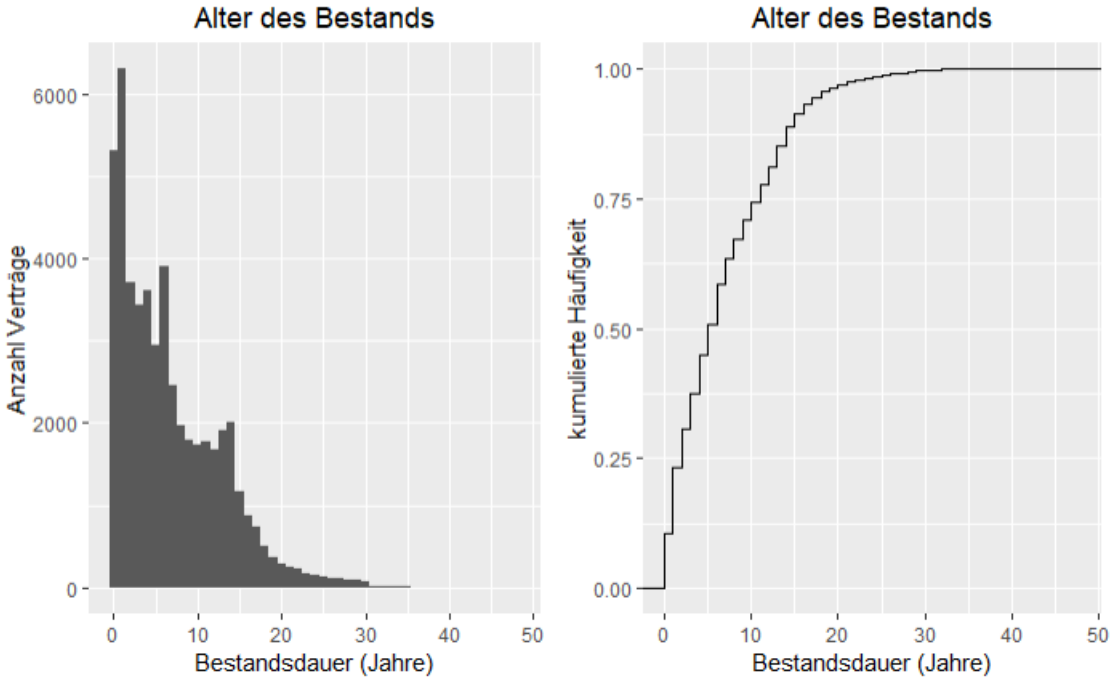
(Anm.: Im Beispiel muss man noch weiter zoomen, um etwas zu sehen, bzw. andere Techniken anwenden.)

(d) Auch hier kann man zunächst die Transparenz verändern (bringt aber nicht viel).

Die Grafik lässt sich ergänzen um Boxplots je Laufzeit. Wenn diese anzeigen, dass die Verteilung schief zur Diagonalen geneigt ist (also z.B. die Mediane nahe an der Diagonalen liegen), handelt es sich um einen eher jungen Bestand.



Klarer sieht man das Alter des Bestands z.B. mit einem Histogramm der Bestandsdauer (= verstrichene Vertragslaufzeit = Laufzeit ./. Restlaufzeit) oder noch besser mit einer empirischen kumulativen Verteilungsfunktion.



Anm.: Die Grafiken in den Musterlösungen zu Teilaufgabe (c) und (d) dienen nur zur Illustration der verbalen Antwort und waren nicht für die Lösung der Aufgabe gefordert.