DAV

DEUTSCHE
AKTUARVEREINIGUNG e.V.

Schriftliche Prüfung im Spezialwissen

# Actuarial Data Science Advanced

gemäß Prüfungsordnung 4
der Deutschen Aktuarvereinigung e. V.

am 26.10.2019

*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 9 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, Dr. Martin Seehafer, Mareike Welter

**Aufgabe 1.** *[Unüberwachtes maschinelles Lernen*, Lernziele 3.1.4, 4.1.6, 4.2.1, 4.2.2*] [50 Punkte]*

Sie möchten eine stochastische Projektionsrechnung (10.000 Kapitalmarktpfade) für einen großen Lebensversicherungsbestand (ca. 1 Mio. Versicherungsverträge) durchführen und haben ein Problem mit der Laufzeit: Die Berechnungen dauern einige Stunden. Daher möchten Sie den Bestand geeignet verdichten.

(a) *[8 Punkte]* Erläutern Sie kurz die Idee der Bestandsverdichtung. Nennen Sie ein Beispiel für ein Verdichtungsverfahren.

(b) *[6 Punkte]* Erläutern Sie kurz (!) den Prozess der Verdichtung, welche Schritte sind notwendig?

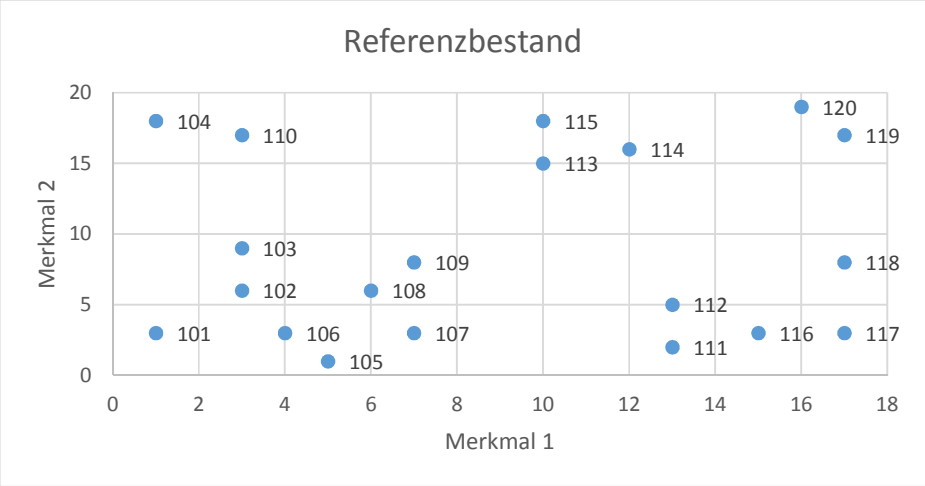
(c) *[10 Punkte]* Geben Sie drei Merkmale an, die für eine Digitalisierung des Bestands geeignet sind. Geben Sie zwei Validierungsregeln an.

(d) *[8 Punkte]* Geben Sie einen Algorithmus an, der für eine Clusterbasierte Bestandsverdichtung geeignet ist. Erläutern Sie kurz die Funktionsweise des Algorithmus.

(e) *[6 Punkte]* Welche „Stellschrauben" (Hyperparameter) haben Sie bei der Clusterbasierten Bestandsverdichtung, um das Ergebnis durch erneute Verdichtung zu bessern?

(f) *[12 Punkte]* Zeichnen Sie in die folgende Grafik 4 geeignete Cluster ein, bestimmen die Centroid-Verträge und ermitteln die Gewichte für jeden Centroid-Vertrag (ohne zu rechnen!).

(Hinweis: Blaue Punkte repräsentieren einen Vertrag mit zwei Merkmalen. Die Zahl bezeichnet die Vertragsnummer).



**Lösungsvorschlag:**

(a) Die Idee bei der Verdichtung ist die Erzeugung eines Teilbestandes (des Versichertenbestandes), der dieselben (oder ähnliche) Eigenschaften in Bezug auf ein oder mehrere relevante Merkmale aufweist, wie der Ausgangsbestand.

Anstatt mit dem Gesamtbestand, rechnet man nun mit dem kleineren Teilbestand und verkürzt damit die Laufzeit der Projektionsrechnungen.

Ein verbreitetes Verfahren der Bestandsverdichtung ist die Clusterbildung. (Daneben gibt es Stichprobenverfahren, aktuarielle Verfahren, …).

(b) Zunächst legt man sogenannte **Validierungsregeln** fest, an Hand derer man die Güte eine Verdichtung bestimmt; also prüft, ob der verdichtete Bestand geeignet ist.

Dann ist der Bestand geeignet zu digitalisieren, damit die Aufgabe einem algorithmischen Lösungsverfahren zugänglich ist. Dazu bestimmt man sogenannte **Merkmale** und übersetzt deren Ausprägungen in Zahlen. So erhält man einen Vektorraum, in dem jeder Vektor einen Vertrag repräsentiert.

In diesem Vektorraum wendet man ein **mathematisches Verfahren** zur Bestimmung des Teilbestands an.

Schließlich **validiert** man mit Hilfe der Validierungsregeln die Verdichtung. Und wiederholt iterativ das Vorgehen, bis man eine (für den Berechnungszweck) geeignete Verdichtung erhält.

(c) Geeignete Merkmale können sein: Versicherungssumme, Bruttobeitrag, Geschlecht der versicherten Person, Stand der Deckungsrückstellung, Stand der verzinslichen Ansammlung, Verlaufswerte, z.B. künftige Cashflows, künftige Deckungsrückstellungen,…

Eine Validierungsregel besteht aus einem Validierungskriterium und einer tolerierbaren Abweichung. Zum Beispiel:

1. Deckungsrückstellung in den Jahren 0; 1; 2; 5 und 10 nach Projektionsbeginn mit einer tolerierbaren Abweichung von max. 10.000 € oder max. 1%
2. Summe der Ablaufleistungen in allen Projektionsjahren mit tolerierbarer Abweichung von 0,5 %

(d) Für eine Clusterbasierte Bestandsverdichtung ist zum Beispiel der k-means Algorithmus geeignet. Vereinfacht beschrieben wird dabei wie folgt vorgegangen:

(i) Wähle K anfängliche Cluster-Centroide, c1, c2, c3, ..., cK

(ii) Für jeden Datenpunkt x, finde den nächsten (nearest neighbour) Cluster-Centroiden

(iii) Für jedes Cluster (gegeben durch vorherigen Schritt), aktualisiere den Cluster-Centroiden (bilde den Mittelwert über alle Cluster-Elemente, d.h. dem Cluster zugeordnete Datenpunkte)

(iv) Gehe zu (ii) bis sich die Cluster nicht mehr ändern (z.B. SSE konstant) oder die Änderung nur noch minimal ist (vorgegebenes Epsilon)

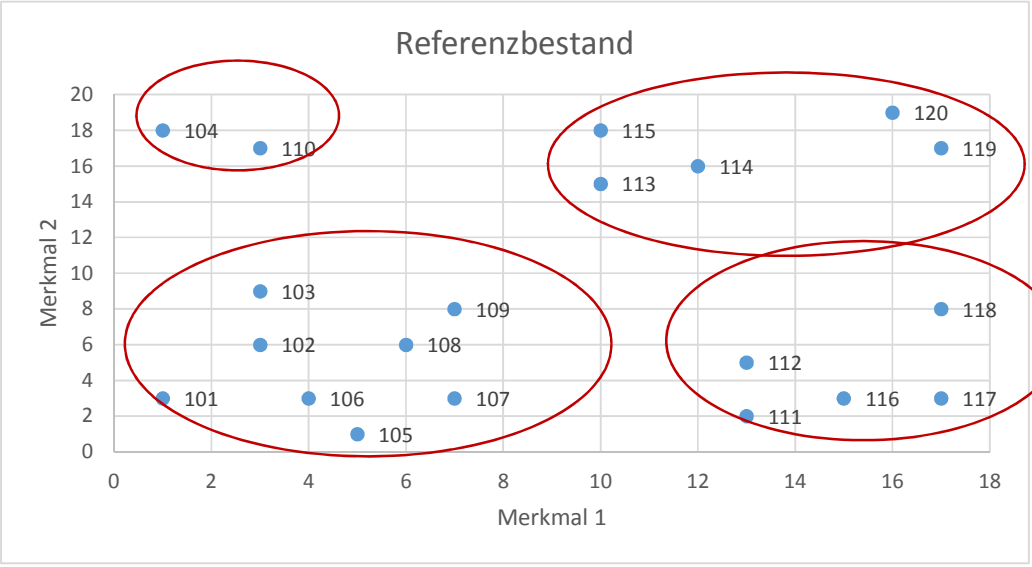
(e) Mögliche Stellschrauben für weitere Verdichtungen sind:

1. Wahl der Merkmale
2. Verdichtungsgrad (invers zu K)
3. Wahl der Startcentroide

(f) Eine mögliche Lösung ist folgende:

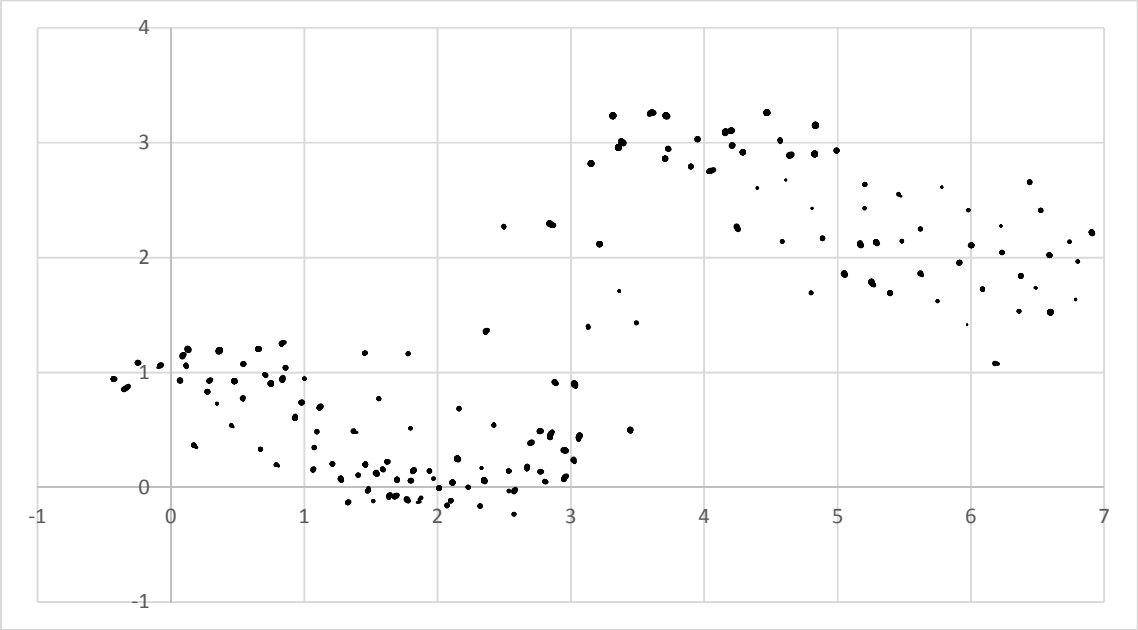
Augenscheinlich ergeben sich (ohne zu rechnen) folgende Cluster und Centroid-Verträge:

- 104 mit Gewicht 2 bzw. alternativ 110 mit Gewicht 2
- 106 mit Gewicht 8
- 114 mit Gewicht 5
- 116 mit Gewicht 5



**Aufgabe 2.** *[*Module *Mathematik / Statistik* (*Regressions- und Clustermethoden 2*) und *Insurance Analytics* (*Data Mining 2*)*] [27 Punkte]*

Gegeben ist der folgende Datensatz, bestehend aus Beobachtungen $(x_i, y_i)$:




(a) *[6 Punkte]* Bei der Modellierung von Daten $(x_i, y_i), i = 1, \dots, n$ mit Polynom-Splines ist eine richtige Wahl von Knoten ein wichtiger Bestandteil der Modellierung. Nennen Sie zwei mögliche Strategien zur Wahl von Knoten und beschreiben Sie diese.

(b) *[9 Punkte]* Aufgrund einer Ihrer Strategien aus a) haben Sie die folgenden Knotenpunkte auf der x-Achse ermittelt:

$$\kappa_1 = 1 < \kappa_2 = 3 < \kappa_3 = 5$$

Für eine Modellierung verwenden Sie die Funktionen $C_0(x) = I(x \leq \kappa_1)$, $C_j(x) = I(\kappa_j < x \leq \kappa_{j+1})$ sowie $C_M(x) = I(\kappa_M < x)$ (mit $j = 1, \dots, M-1$). Dabei bezeichnet $I(\cdot)$ die Indikatorfunktion.

Erstellen Sie auf Basis der ermittelten Knotenpunkte und Basisfunktionen ein additives Modell mit den Parametern $\beta_0, \dots, \beta_M$.

Berechnen Sie, für gegebenes M und vorgegebene Knoten, $\sum_{j=0}^{M} C_j(x)$ für $x \in \mathrm{R}$ und begründen Sie Ihre Antwort.

(c) *[6 Punkte]* Auf Basis eines vorgegebenen Datensatzes haben Sie die folgenden Parameter berechnet:

$$\beta_0 = 1, \beta_1 = -1, \beta_2 = 2 \; sowie \; \beta_3 = 1$$

Skizzieren Sie das ermittelte Modell. Gleichen Sie das ermittelte Modell mit den vorgegebenen Datenpunkten visuell ab.

(d) *[6 Punkte]* Nennen Sie die 6 Phasen des Prozessmodells CRISP-DM und geben Sie für jede Phase 2 Aufgaben an. Welchem Prozessschritt würden Sie die Aufgabenteile (a) bis (c) zuordnen?

**Lösungsvorschlag:**

Aufgabenteil (a):

1. Äquidistante Knoten: Unterteilung des Wertebereichs [a,b] in m-1 Intervalle der Breite h = (b-a)/(m-1). Damit ergeben sich die Knoten zu: $\kappa_j = a + (j-1) * h, j = 1, \dots, m$

2. Quantilbasierte Knoten: Verwendung der (j-1)/(m-1)-Quantile (j=1,…,m) der Kovariablenausprägungen $x_1, \dots, x_n$. Damit werden viele Knoten in Bereiche gelegt, in denen auch eine große Zahl von Beobachtungen liegt.

3. Visuelle Knotenwahl anhand eines Streudiagramms: Gezielte Setzung von Knoten. Hiermit kann die Knotendichte an die Variabilität der Daten angepasst werden. Zudem können inhaltliche Überlegungen berücksichtigt werden.

Aufgabenteil (b):

In diesem Fall ist ein passendes Modell (M = 3):

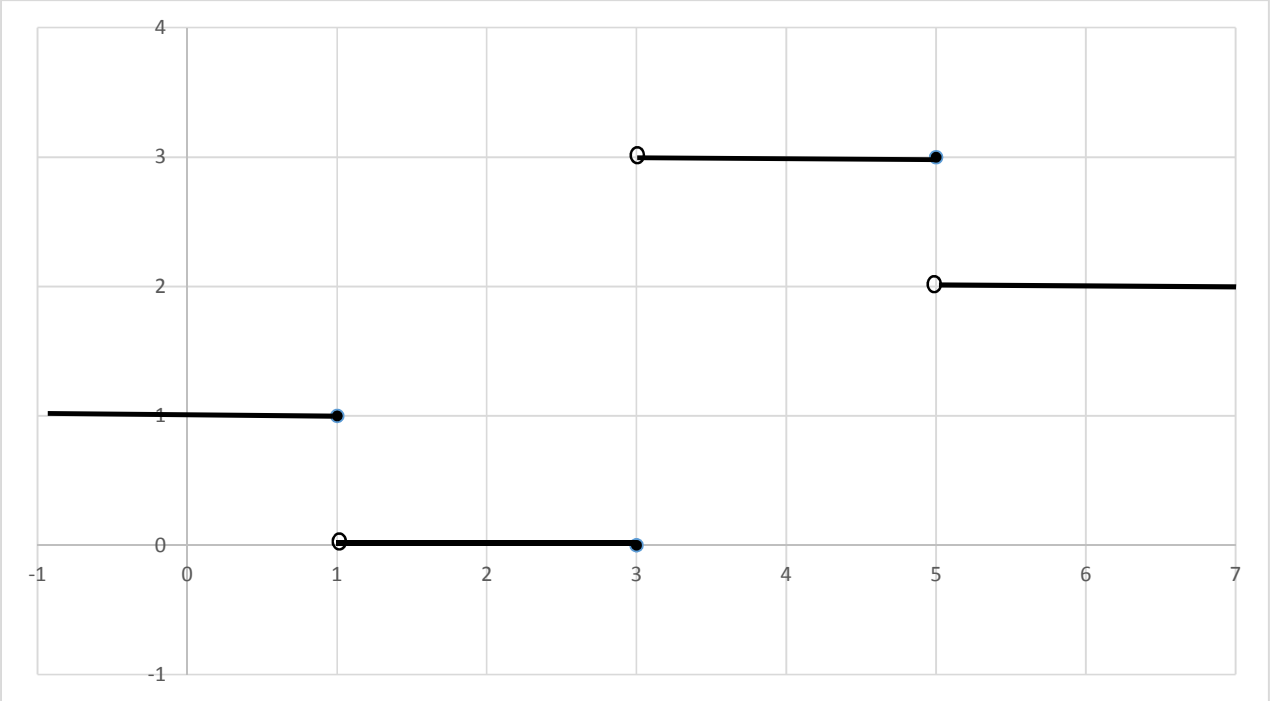
$$f(x) = \beta_0 + \beta_1 * C_1(x) + \beta_2 * C_2(x) + \beta_3 * C_3(x)$$

Es gilt $\sum_{j=0}^{M} C_j(x) = 1$ für $x \in \mathrm{R}$, da die zugrundeliegenden Intervalle disjunkt sind und ihre Vereinigung R ergibt.

Aufgabenteil (c):

Mit den angegebenen Informationen ergibt sich

$$f(x) = 1 - I(1 < x \leq 3) + 2 * I(3 < x \leq 5) + I(5 < x)$$




Aufgabenteil (d):

Die 6 Phasen des CRISP-DM Prozesses und Aufgaben:

1. Geschäftsverständnis (Bestimmung von Geschäftszielen, Bewertung der Situation, Bestimmung von Data-Mining-Zielen, Erstellen eines Projektplans)

2. Datenverständnis (Sammlung ursprünglicher Daten, Beschreiben von Daten, Untersuchen von Daten, Überprüfung der Datenqualität)

3. Datenvorbereitung (Auswählen / Bereinigen / Erstellen / Integrieren / Formatieren von Daten)

4. Modellierung (Auswählen der Modellbildungsverfahren, Generierung eines Testdesigns, Erstellen der Modelle, Bewerten des Modells)

5. Evaluierung (Evaluieren der Ergebnisse, Überprüfungsprozess, Bestimmung der nächsten Schritte)

6. Bereitstellung (Planung der Bereitstellung, Planen von Überwachung und Anpassung, Erstellen eines Abschlussberichts, Durchführung einer abschließenden Projektbewertung)

Die Aufgabenteile (a) – (c) wären dem Punkt 4 (Modellierung) zuzuordnen.

**Aufgabe 3.** *[*Modul *Insurance Analytics] [47 Punkte]*

In der Sparte Wohngebäude in Ihrer Versicherung sinkt der Bestand in den letzten Jahren deutlich. Zur Reduzierung der Kündigungen haben Sie die Aufgabe, ein Prognosemodell für Kündigungen (Stornomodell) zu erstellen.

Grundmenge der Analyse sind 200.000 Verträge (=Einzeldatensätze), von denen 20 % von den Versicherungsnehmern gekündigt wurden. Zur Prognose der Kündigungen wurde ein Entscheidungsbaum mit 250 Knoten erstellt. Durch den Entscheidungsbaum wurden folgende Vorhersagen zur Kündigung und Nicht-Kündigung ermittelt:

| | Trainings-daten | Test-daten |
|---|---|---|
| Anzahl Fälle mit korrekter Vorhersage der Kündigung | 19.000 | 11.000 |
| Anzahl Fälle mit korrekter Vorhersage der Nicht-Kündigung | 77.000 | 51.000 |

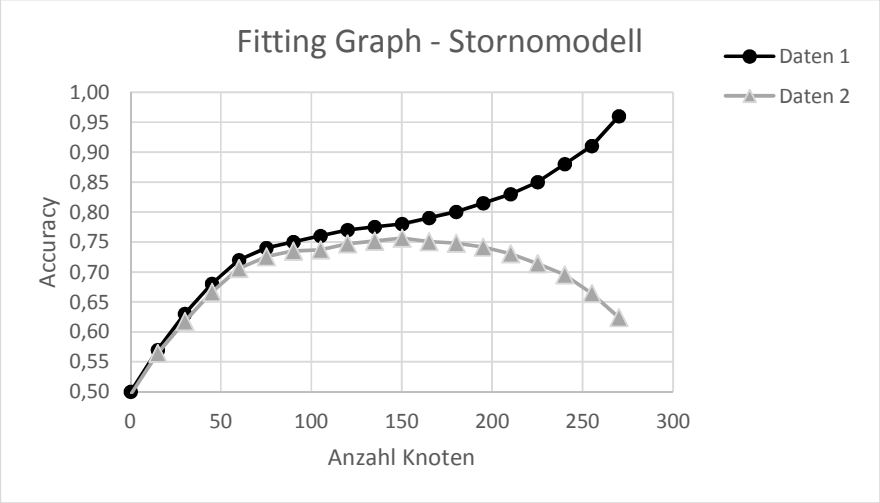
(a) *[3 Punkte]* Nennen Sie die Berechnungsformeln zur „Accuracy" und „Error Rate".

(b) *[12 Punkte]* Bestimmen Sie die Confusion Matrizen für die Trainings- und Testdaten.

Bewerten und begründen Sie, ob ein Overfitting oder Underfitting vorliegt.

Die Trainings- und Testdaten wurden zufällig aus der Grundmenge bestimmt. Nennen und beschreiben Sie eine alternative Methode zur Bestimmung der Trainings- und Testdaten.

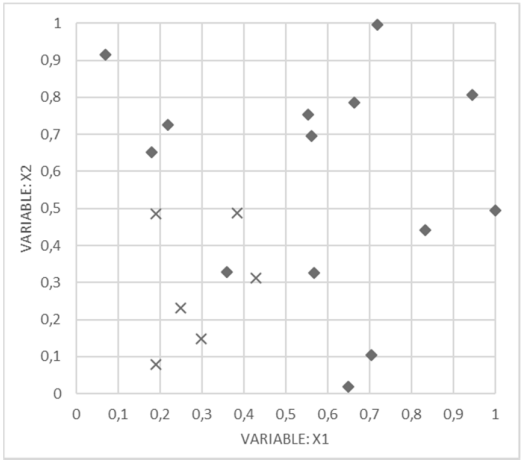
In der folgenden Grafik ist ein Fitting Graph des Stornomodells dargestellt:




(c) *[8 Punkte]* Kennzeichnen Sie die Datenreihe „Daten 1" und „Daten 2" mit Test- und Trainingsdaten. Begründen Sie Ihre Antwort.

Bestimmen Sie anhand des Fitting Graph die aus Ihrer Sicht „optimale" Größe des Entscheidungsbaums. Begründen Sie Ihre Antwort.

In der nachfolgenden Grafik sind exemplarische 20 Testdatensätze der Vertragsdaten dargestellt. Ein „X" entspricht hierbei der Kündigung eines Vertrags.




(d) [5 Punkte] Bestimmen Sie unter Verwendung eines Entscheidungsbaums zwei Splitting Criteria, um eine möglichst hohe Accuracy zu erhalten. Zeichnen Sie die Grenzen in die Grafik ein.

Zusätzlich zu dem Entscheidungsbaum wird ein Random Forest trainiert.

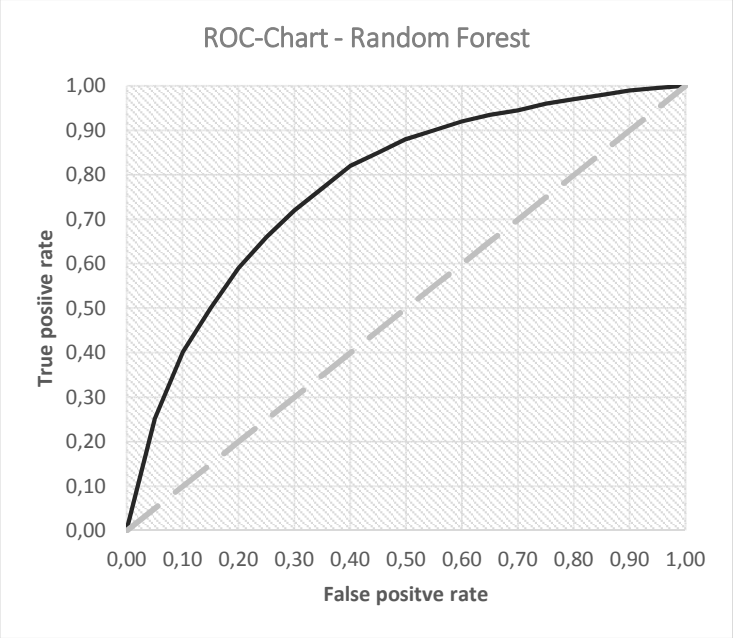
(e) *[4 Punkte]* Benennen Sie die wesentliche Eigenschaft eines Random Forest im Vergleich zu einem Entscheidungsbaum.

Der Vergleich der beiden Methoden soll über eine Cross-Validation erfolgen. Hierzu wurden die Grunddaten in 5 Teilmengen aufgeteilt.

(f) *[9 Punkte]* Beschreiben Sie die Funktionsweise der Cross-Validation anhand der 5 Teilmengen.

Benennen und erläutern Sie einen Vorteil der Cross-Validation.

Zur Bewertung der Modellqualität des Random Forest wurde ein ROC-Chart auf den Trainingsdaten erstellt. Das ROC-Chart ist folgend dargestellt:




(g) *[6 Punkte]* Zur Reduzierung des Stornos sollen Kunden mit einer hohen Stornowahrscheinlichkeit kontaktiert werden Die Vorgaben vom Vorstand ist, dass maximal 8.000 Versicherungsnehmer kontaktiert werden dürfen, die nicht vorhaben zu kündigen. Wie viele Kunden können Sie anhand des ROC-Chart maximal kontaktieren? Gehen Sie von einem Bestand von 100.000 Verträgen aus, von denen 20.000 Versicherungsnehmer den Plan haben zu kündigen. Begründen Sie Ihre Antwort.

**Lösungsvorschlag:**

(a) Die Berechnung der „Accuracy" und „Error Rate" lautet:

a. $Accuracy = \frac{Anzahl\ korrekt\ vorhergesagter\ Werte}{Gesamtanzahl\ der\ Vorhersagen}$

b. $Error\ Rate = \frac{Anzahl\ falsch\ vorhergesagter\ Werte}{Gesamtanzahl\ der\ Vorhersagen}$

(b) Zur Bestimmung der Confusion Matrizen sind folgende Werte zu ermitteln:

a. $True\ Positive\ (TP) = Anzahl\ korrekt\ vorhergesagter\ Kündigungen$

b. $False\ Positive\ (FP) = Anzahl\ falsch\ vorhergesagter\ Kündigungen$

c. $True\ Negative\ (TN) = \ Anzahl\ korrekt\ vorhergesagter\ Nicht\ Kündigungen$

d. $False\ Negative\ (FN) = Anzahl\ falsch\ vorhergesagter\ Nicht\ Kündigungen$

Aus diesen Werten ergibt sich eine Confusion Matrix:

| | |
|---|---|
| TP | FN |
| FP | TN |

Mit den angegebenen Daten ergeben sich die folgenden Werte und die Confusion Matrizen:

| **Confusion Matrix - Trainingsdaten** | | | |
|---|---|---|---|
| | | Vorhersage | |
| | | Kündigung | Keine Kündigung |
| Tatsächlicher Wert | Kündigung | 19.000 | 1.000 (= 20.000-19.000 ) |
| | Keine Kündigung | 3.000 (= 80.000-77.000) | 77.000 |

| **Confusion Matrix - Testdaten** | | | |
|---|---|---|---|
| | | Vorhersage | |
| | | Kündigung | Keine Kündigung |
| Tatsächlicher Wert | Kündigung | 11.000 | 9.000 (= 20.000-11.000 ) |
| | Keine Kündigung | 29.000 (= 80.000-51.000) | 51.000 |

Da die Trainingsdaten eine hohe Accuracy aufweisen und die Testdaten jedoch eine geringe Accuracy haben, ist davon auszugehen, dass ein Overfitting vorliegt und das Modell nicht verallgemeinert.

Eine (weitere) Möglichkeit zur Bestimmung der Test- und Trainingsdaten ist ein stratifiziertes (stratified) Sampling. Bei dieser Datenermittlung werden die Grunddaten in verschiedene Teilmengen aufgeteilt und aus diesen Teilmengen zufällig Daten ermittelt.

(c) Es gibt folgende Zuordnung der Daten 1 und Daten 2:

a. Daten 1 = Trainingsdaten

b. Daten 2 = Testdaten

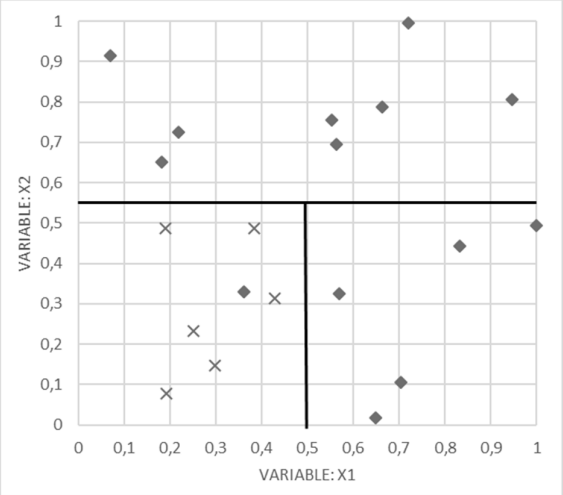
Mit steigender Komplexität des Modells steigt die „Accuracy" kontinuierlich in den Daten 1, dieses Verhalten tritt bei Trainingsdaten auf. In den Daten 2 ist eine sinkende „Accuracy" ab einer gewissen Komplexität des Modells zu sehen (=Overfitting), dieses Verhalten tritt bei den Testdaten auf.

Bei einer Modellkomplexität von über 150 Knoten steigt die Differenz in der „Accuracy" zwischen den Trainings- und Testdaten und ein Overfitting tritt auf. Bei einer Modellgröße von 150 Knoten ist noch kein Overfitting zu erkennen und die Testdaten haben die größte „Accuracy". Aus diesen Gründen ist eine Modellkomplexität von 150 Knoten zu wählen.

(d) Die Daten können durch die folgenden Splitting Criteria getrennt werden.

- Splitting Criterion 1: X2 < 0,55
- Splitting Criterion 2: X1 < 0,5

Hierdurch werden die Daten wie in der folgenden Grafik dargestellt getrennt:




(e) Bei einem Random Forest werden mehrere / viele Entscheidungsbäume aufgebaut. Unter Verwendung einer Ensemble Methode werden aus dem Satz von Bäumen („weak learner") ein Modell („strong learner") zur Vorhersage erzeugt.

(f) Bei der Cross Validation werden die Daten in fünf Iterationen in Trainings- und Testdaten aufgeteilt. Die folgende Grafik zeigt eine mögliche Aufteilung der Trainings- und Testdaten in den fünf Iterationen:

| | Iteration | | | | |
|---|---|---|---|---|---|
| | *1* | *2* | *3* | *4* | *5* |
| Trainingsdaten | 2<br>3<br>4<br>5 | 1<br>3<br>4<br>5 | 1<br>2<br>4<br>5 | 1<br>2<br>3<br>5 | 1<br>2<br>3<br>4 |
| Testdaten | 1 | 2 | 3 | 4 | 5 |

In jeder Iteration wird ein Modell mit den entsprechenden Trainingsdaten erstellt und mit den Testdaten die Modellperformance bewertet.

Durch die Cross-Validation werden mehrere Bewertungen zur Performance der Modelle erstellt (bei einer Aufteilung in Trainings- und Testdaten erfolgt eine Bewertung der Modellqualität). Durch die mehrfache Bewertung der Modellqualität kann der Erwartungswert und die Varianz der Modellqualität ermittelt werden und hierdurch weitere Informationen zur Modellqualität bestimmt werden.

(g) Die False Positive Rate aus dem ROC-Chart entspricht dem Anteil der falsch vorhergesagten Kündigungen in der Menge der Personen, die nicht vorhaben zu kündigen. Vorgabe des Vorstands ist es, dass maximal 8.000 falsch vorhergesagte Kündigungen akzeptiert werden. Dies entspricht einer False Positive Rate von 10 % (8.000 von 80.000). Aus dem ROC-Chart ist weiterhin ersichtlich, dass bei einer False Positive Rate von 10 % eine True Positive Rate von 40 % existiert. Dies entspricht der Anzahl von 8.000 Personen, bei denen die Kündigung korrekt vorhergesagt werden kann. Insgesamt können somit 16.000 Personen kontaktiert werden.

**Aufgabe 4.** [Modul *Informationstechnologie* (Datenmanagement 2, Datenverarbeitungstechnologien 2)*] [56 Punkte]*

Ein KFZ-Versicherer möchte Daten über Vertragswerkstätten in einer Datenbank speichern, die sich die Sachbearbeiter bei Eingang von Schadenmeldungen ansehen können. Für jede Werkstatt sollen u.a. gespeichert werden:

- Werkstatt-ID
- Name der Werkstatt
- Marken und ggf. Typen, die repariert werden können
- Adresse der Werkstatt im Klartext und GPS-Koordinaten
- E-Mail, Telefonnummer
- Geschäftszeiten

Die Daten werden in einer Dokumenten-DB festgehalten.

(a) *[12 Punkte]* Nennen Sie drei Klassen von No-SQL-Datenbanken.
Nennen Sie drei allgemeine Gründe, die dafür sprechen könnten, Dokumentendatenbanken einzusetzen und erläutern Sie jeweils, warum diese im relationalen Modell nicht oder nur schwierig abzubilden sind.

Typische Anfragen an die Datenbank sind etwa:

i. Suche nach Werkstätten für eine bestimmte Fahrzeugmarke in der Nähe eines vorgegebenen Orts
ii. Geschäftszeiten einer bestimmten Werkstatt am heutigen Tag

(b) *[14 Punkte]* Definieren Sie eine mögliche JSON-Struktur für die zu Grunde liegenden Dokumente. Spezifizieren Sie dabei auch die möglichen Unterobjekte, wo Sie dies für sinnvoll halten, so dass die oben genannten Anfragen verarbeitet werden können.
Mit Blick auf Ihr Datenmodell: Geben Sie an, welche Vorverarbeitungsschritte der Eingaben Sie für die Anfragen i. und ii. als sinnvoll erachten, um möglichst „die richtigen" Antworten zu erhalten.

(c) *[4 Punkte]* Bei der Datenerfassung wurden nicht alle Einträge mit GPS-Koordinaten versehen. Beschreiben Sie ein Vorgehen, wie Sie diese Daten nun nachträglich (durch ein Skript/einen Batch-Lauf) ergänzen können.

(d) *[4 Punkte]* Wir betrachten die Beziehung Werkstatt zu Fahrzeugmarken. Wie würden Sie diese Beziehung üblicherweise in einem relationalen Schema abbilden? Erklären Sie, warum sich hieraus ein Vorteil hinsichtlich Datenkonsistenz im relationalen Modell ergibt!

Teilweise gehören die einzelnen Werkstätten zu größeren *Ketten*, mit denen zusammen abgerechnet wird. Eine Kette ist hierbei u.a. charakterisiert durch eine (gemeinsame) Anschrift und eine zugehörige Bankverbindung. Ihre DB bietet als einzige Konsistenzgarantie die Atomarität von Änderungen auf einzelnen Dokumenten.

(e) *[3 Punkte]* Erläutern Sie, was die genannte Konsistenzgarantie genau bedeutet.

(f) *[8 Punkte]* Geben Sie eine Möglichkeit an, wie die Daten hinsichtlich der Kettenzugehörigkeit in der Datenbank vorgehalten werden könnten, ohne zusätzliche Dokumente anzulegen. Bewerten Sie den Ansatz hinsichtlich Datenkonsistenz.
Geben Sie eine zweite Möglichkeit der Speicherung der Kettenzugehörigkeit an, ohne die bestehenden Dokumente zu ändern. Kann es auch in diesem Fall (etwa beim Anlegen oder Löschen von Werkstätten) kurzzeitig Risiken von Dateninkonsistenzen geben? Erläutern Sie dies!

Das Unternehmen beschließt, die Rückmeldungen der Kunden zur Zufriedenheit mit den Werkstätten auch mit in die Datenbank aufzunehmen (Geschwindigkeit der Abwicklung, Zufriedenheit auf einer Skala von 1-5, Datum der Rückmeldung, Auftragsnummer). Die Rückmeldungen werden als neues Listen-Feld namens „Feedback" jeweils innerhalb des Werkstatteintrages vorgenommen:

„Feedback": [{feedback_A}, {feedback_B}, ...] wobei die einzelnen Feedbackeinträge wie folgt aufgebaut sind:

```
"feedback_A": {
  "score": 3,
  "datum": "2019-03-12",
  "kundennr": "XXX",
  "auftragsnummer": "YYY",
  "kommentar": "Hier Freitext des Kunden"
}
```

(g) *[11 Punkte]* Zur Überprüfung der Zusammenarbeit mit den Werkstätten ist eine Auswertung zu erstellen, die eine Liste mit Einträgen analog dem folgenden Beispiel liefert:

{

„Werkstatt-ID": 123,

„score": 3,

„kommentar": „Beispielfreitext"

}

Erklären Sie, welche Einzelschritte (Projektion, ...) in der abzusetzenden Query erforderlich sind, um sämtliche dieser Datensätze zu erhalten, die mindestens eine Bewertung haben (umgangssprachliche Beschreibung)!

Die mit dieser Query extrahierten Daten werden nun in eine interaktive Notebookumgebung geladen. Spezifizieren Sie eine Sequenz von map/reduce/reduceByKey Operationen, die zusammen als Ergebnis den mittleren Score pro Werkstatt-ID ermitteln.

**Lösungsvorschlag:**

(a) Dokumenten-Datenbanken, Key-Value-Stores, Graphen-Datenbanken, Column-Store-Datenbanken,...
Mögliche Gründe für den Einsatz von Dokumentendatenbanken:

1. Performanz: Dok.-DBs können sehr schnelle Antwortzeiten anbieten, da
    i. Auf Grund des Datenmodell-Designs je Anfrage meist nur ein (oder wenige) Records in der Datenbank lokalisiert werden müssen, in dem dann gleich sämtliche benötigte Informationen vorgefunden werden
    ii. Im Vergleich zum Relational Modell viele zusätzliche Prüfungen (z.B. Foreign Key Constraints) nicht vorgesehen sind.
2. Unterstützung verteilter Systeme, hohe Last:
    i. Da *Joins* nicht vorgesehen/notwendig sind, ermöglicht das Dok.-DB Paradigma, die Daten auf mehrere Server zu verteilen (Sharding-Szenario). Dadurch können insgesamt größere Datenmengen in der DB vorgehalten werden, einzelne Anfragen ggf. durch einen einzelnen Shard beantwortet bzw. parallelisiert werden, wodurch ein Laufzeitgewinn entstehen kann.
3. Schemafreiheit:
    i. Durch die Möglichkeit, verschiedenartig strukturierte Dokumente zusammen in einem Container vorzuhalten, gewinnt man Flexibilität, die es ermöglicht, heterogene Daten zusammen zu verwalten. Gerade im frühen Stadium einer Neuentwicklung, vereinfacht diese Freiheit auch die Erweiterung des Datenmodells.

(b) Eine Möglichkeit, die Datensätze anzugeben, wäre die folgende:

```
{
  "Werkstatt-ID": "akjhjk62b299",
  "ws_name": "Widholzens Pannenservice",
  "marken": [
    {
      "hersteller": "TOYOTA",
      "typen": ["ALL"]
    },
    {
      "hersteller": "FIAT",
      "typen": ["PANDA"]
    }
  ],
  "adresse": {
    "gps": [48.135125, 11.581981],
    "stadt": "MÜNCHEN",
    "strasse": "Ingolstädter Straße",
    "nummer": 99
  },
  "e-mail": "wiholzens@aol.com",
  "tel": "0190...",
  "geschaeftszeiten": {
    "MO": [[9,12], [14,18]],
    "DI": [[9,12], [14,18]],
    "MI": [[9,12], [14,18]],
    "DO": [[9,12], [14,18]],
    "FR": [[9,12], [14,18]],
    "SA": [[10, 13]]
  }
}
```

Hierbei sind mehrere Konventionen eingearbeitet:

- Einträge bei gps immer in der Reihenfolge „lat", „lng"
- „typen"=["ALL"] bedeutet, dass alle Fabrikate eines Herstellers angenommen werden

Sinnvolle Vorverarbeitungsschritte der Eingaben:

1. Ort und Marke: die Eingaben sollten getrimmt (=um führende und abschließende Leerzeichen bereinigt) und in Großbuchstaben konvertiert werden. Sinnvoll ist auch ein Vorab-Abgleich mit den gültigen Modellen, wozu u.U. eine separate Datenbankabfrage vorher notwendig ist.
2. Geschäftszeiten: Um in der Struktur oben sinnvoll abfragen zu können, muss aus dem Eingabe-Datum zunächst der Wochentag extrahiert werden. Dies kann auf Client-Seite oder, falls das Feature vorhanden ist, auf Serverseite als Teil der Anfrage durchgeführt werden.

(c) Einen entsprechenden Batch-Lauf könnte man wie folgt aufbauen:

- Lesen aller Datensätze ohne GPS-Information aus der Datenbank, wobei jeweils mindestens die „Werkstatt-ID" und die „Adresse" extrahiert wird
- Abfrage der GPS-Koordinaten bei einem Datenprovider anhand der Adressdaten (wir gehen davon aus, dass ein Webservice zur Verfügung steht)
- Updates der Records, für die eine gültige Antwort zurückgeliefert wurde

Anschließend sollten die Records manuell geprüft werden, für die keine gültige Antwort geliefert werden konnte, hier könnten die Daten fehlerhaft sein.

(d) Die Beziehung Werkstatt:Fahrzeugmarke ist N:M, d.h. eine Werkstatt bearbeitet kein, ein oder mehrere Marken und eine Marke wird in einer, keiner oder von mehreren Werkstätten angenommen. Im Relationalen Modell hätte man eine Tabelle für die Werkstätten, eine für die Marken und eine weitere, die alle verknüpften Records (Werkstatt-ID, Marken-ID) enthält. Hierdurch wird eine Datenkonsistenz erreicht: Im oben beschriebenen Modell in der Dok-DB mit Freitextfeldern für die Marken kann es zu Fehlern bei der Erfassung kommen (z.B. Falschschreibung).

(e) Konsistenz wird nur auf der Ebene einzelner Dokumente garantiert, d.h. alle Änderungen, die durch eine einzelne Query an einem Dokument vorgenommen werden, werden zugleich wirksam oder scheitern. Kein anderer Leser sieht ein Dokument in einem Zustand, in dem es nur teilweise aktualisiert wurde. Änderungen, die mehrere Dokumente zugleich betreffen, sieht unter Umständen ein anderer Leser in einem Zustand, in dem sie sich erst auf einen Teil der betroffenen Daten ausgewirkt haben.

(f) 1. Möglichkeit: Jedes Dokument könnte ein Feld namens „Kette" erhalten. Dafür muss aber die Zusatzinformation (wie Kontoverbindung) ebenfalls in jedem Werkstatt-Eintrag dupliziert werden, wodurch redundante Daten entstehen, die bei Änderungen immer simultan nachgezogen werden müssen.
2. Möglichkeit: Man kann beispielsweise einen separaten Container mit den Datensätzen für die Ketten anlegen, wobei jede Kette eine Liste mit allen Werkstatt-Ids enthält, die zur Kette gehören. Beim Anlegen oder Löschen von Werkstätten, müssen ggf. zwei Records geändert werden (die Werkstatt und der Ketteneintrag), was i.V.m. der eingeschränkten Konsistenzgarantie zu Inkonsistenzen führen kann, da ein anderer Leser u.U. nur eine der Aktualisierungen zur Kenntnis nimmt.

(g) In der Query wird man (nicht notwendig in der gleichen Reihenfolge):

1. Filtern: Selektieren aller Datensätze mit nichtleerer „Feedback"-Liste im Datensatz
2. Auf die Felder *Werkstatt-ID, Feedback* projizieren
3. Da mehrere Kommentare in einem Array innerhalb eines Dokuments vorliegen, wird ein unwind-Schritt benötigt, der aus jedem Array-Element ein separates Dokument erstellt.
4. Erneut Projizieren auf die Felder „*Werkstatt-ID*" „Feedback.score" und „Feedback.kommentar"
5. Die Feldnamen richtig umbenennen

Um den mittleren Score je Werkstatt zu ermitteln kann man in folgenden Schritten vorgehen:

1. In einem Map-Schritt integriert man zunächst einen Counter in jeden Record:
   ```
   {
    "Werkstatt-ID": 123,
    "score": 3,
    "count": 1
   }
   ```
2. In einem „Reduce-By-Key"-Schritt mit dem *key* „Werkstatt-ID" summiert man die Scores und die counts jeweils pro Werkstatt. Typischer Ergebnis-Datensatz:
   ```
   {
     "Werkstatt-ID": 123,
     "score": 385,
     "count": 110
    }
   ```
   Nach diesem Schritt gibt es je Werkstatt nur noch einen Datensatz.
3. In einem Map-Schritt bildet man den Durchschnitt, indem man *score* durch *count* teilt, ein typischer Datensatz sieht dann so aus:
   ```
   {
     "Werkstatt-ID": 123,
     "avg_score": 3.5
   }
   ```