Aufgabe: Standardfehler Jackknife

- Schreiben Sie eine R-Funktion, die den Jackknife Standardfehler der Interquartilsabstände zufälliger Abstände zweier Punkte im n-dimensionalen Einheitswürfel berechnet. Der Stichprobenumfang betrage wie im Bootstrap-Fall 100 bzw. 1000.

- Simulieren Sie für die Dimensionen $1 \leq n \leq 10$ die Schätzer für die Interquartilsabstände und ihre Standardfehler mittels Jackknife.

- (30 min) Na ja, wohl etwas wenig angesetzt ...

Ausführliche Musterlösung

- Der erste Schritt, um eine Programmieraufgabe zu lösen, deren Programmcode nicht direkt vollständig vor dem geistigen Auge steht, muss sein, die Aufgabenstellung in kleinere Teile zu zerlegen.

- Bei der gestellten Aufgabe sind dies beispielsweise:

  1. Erzeugen einer Stichprobe der gewünschten Art.
  2. Eine Implementierung der zu untersuchenden Statistik. In diesem Fall die Berechnung des Interquartilsabstandes.
  3. Berechnung der Jackknife-Replikationen $\hat{\theta_{(i)}}$.
  4. Dann noch der Vollständigkeit halber die Schleifen über die Dimension und die beiden Stichprobenumfänge.

- Im Einzelnen:
  - Erzeugen der Stichproben: Der Abstand zweier Punkte $x, y \in \mathcal{R}^n$ ist definiert als

$$|x - y| = \sqrt{\sum_{1}^{n} (x_i - y_i)^2}.$$

    Ganz naiv in `R` umgesetzt:
    `sqrt ( sum( (x-y)^2 ) ).`
    Auch eine Schleife wäre ok, aber nicht gemäß der Philosophie von `R`!
  - Damit ist der Abstand zweier zufälliger Punkte im n-dimensionalen Einheitsquadrat gegeben durch:
    `sqrt ( sum( (runif(n)-runif(n))^2 ) ).`
  - Also kann man sich eine Stichprobe von Abständen vom Umfang m verschaffen mit
    `replicate(m, sqrt ( sum( (runif(n)-runif(n))^2 ) ) ).`
  - Später sollen Stichproben für verschiedene Dimensionen und Stichpro-

benumfänge geszogen werden, man benötigt also eine Funktion zur Stichprobenerzeugung, die die entsprechenden Größen als Parameter akzeptiert:

```
unitsquare.distance.sample <-
    function(dimension=1, anz=250){
       replicate(anz, sqrt(sum((runif(dimension) -
                                runif(dimension))^2))) }
```

– Die Stichprobenerzeugung hat man hier im Griff. Bis das Programm soweit läuft, können zunächst die Konstanten auf einen kleinen Wert festgelegt werden. Eine "Probierstichprobe" wird erzeugt! Dieser Schritt fehlt normalerwiese in Musterlösungen, da dort ja nur das Ergebnis des Programmierprozesse gezeigt wird. In der Praxis muss etwas herumprobiert werden, macht man Tippfehler etc.

```
n<-1 ; m<-100
stichprobe<-replicate(m, sqrt( sum( (runif(n)-runif(n))^2 )))
```

– Die gesuchte Statistik ist der Interquartilsabstand der Stichprobe. Dies

läßt sich direkt in R übersetzen:

```
quantile(stichprobe, 0.75) - quantile(stichprobe, 0.25)
```

– Diese Größe muss für jede Jackknife-Teilstichprobe ausgerechnet werden. Es bietet sich also an, eine Funktion zu schreiben, die als Parameter eine Stichprobe, also einen Vektor, akzeptiert und den Interquartilsabstand zurückliefert.

```
iq <- function(stichprobe)
    quantile(stichprobe, 0.75) - quantile(stichprobe, 0.25)
```

– Nun die Schleife über die Jackknifestichproben, mit Speichern der Einzelergebnisse:

```
jackknife.estimators <- rep(NA, m)
for (i in 1:length(stichprobe))
    jackknife.estimators[i] <- iq(stichprobe[-i])
```

– Zuletzt muss noch der Standardfehler für die Jackknife-Stichproben berechnet werden. Die Formel lässt sich direkt in R formulieren:

```
sdj <- sqrt((anzahl-1)/anzahl *
        sum( (jackknife.estimators -
              mean(jackknife.estimators))^2 ) )
```

- Als Gesamtlösung also:

```
unitsquare.distance.sample <- function(dimension=1, anz=250){
  replicate(anz,
    sqrt(sum((runif(dimension) - runif(dimension))^2)))}

for (dimension in 1:10){
  for (anzahl in c(100, 1000)){
  jackknife.estimators <- rep(NA, anzahl)
  stichprobe <- unitsquare.distance.sample(dimension, anzahl)
  for (i in 1:length(stichprobe))
         jackknife.estimators[i] <-iq(stichprobe[-i])
  sdj <- sqrt((anzahl-1)/anzahl *
    sum( (jackknife.estimators -
                  mean(jackknife.estimators))^2 ) )
  cat ("anzahl ", anzahl,"dimension ", dimension,
       "mittel ", mean(jackknife.estimators),
       "standardfehler ", sdj, "\n") }}
```

Aufgabe *delete-d* Jackknife

- Programmieren Sie eine Funktion, die für ein wählbares d alle *delete-d* Jackknife Stichproben erzeugt.
- Vermutlich schwierig! Leiten Sie eine rekursive Prozedur für die Erzeugung der Stichproben her!
- Lösung: Man muss erkennen, dass man die ausgelassenen Werte rekursiv behandeln kann.
- In einem ersten Schritt wird lediglich die Position eines fehlenden Wertes festgelegt und dann aus dem Restvektor rekursiv noch d-1 Elemente entfernt.

- Lösung:

```
dJackknifesamples <- function(x, d=1, restart=1, leftout=NULL)
  {
    for (nextleftout in restart:(length(x)-(d-1)))
      {
        if (d==1) {
          cleftout <- c(leftout, nextleftout)
          cat(x[-cleftout], "\n")
        }
        if (d >1 )
           dJackknifesamples(x, d-1,nextleftout+1,
                                    c(leftout,nextleftout))
      }
  }
```

Kreuzvalidierung - Crossvalidation

- Genau wie beim Jackknife handelt es sich bei der Kreuzvalidierung um eine Methode, die in vielen Bereichen der Statistik Anwendung findet.

- Bereits 1948 hat Kurtz die einfache Kreuzvalidierung vorgeschlagen, Mosier (1951) hat das Prinzip auf mehr als eine Teilung der Grundgesamtheit erweitert (double-crossvalidation), Krus und Fuller (1982) machten einen weiteren Schritt hin zu allgemeinen Partitionierungen der Grundgesamtheit.

- Das grundsätzliche Ziel der Kreuzvalidierung ist die Bestätigung (oder auch Nicht-Bestätigung) von Ergebnissen statistischer Analysen (Klassifikation, Modellbildung).

Kreuzvalidierung zur Verbesserung von Prognosen

- Kreuzvalidierung ähnelt insofern dem Testen von Hypothesen, dass überprüft werden soll, ob Analyseergebnisse auf einer (Teil-)Stichprobe rein zufällig sind oder auf neuen Daten reproduziert werden können.

- "Neu" bedeutet in diesem Fall, dass man Datenpunkte anschaut, die nicht bei der Modellbildung betrachtet wurden.

- Die Vorstellung dabei ist, dass man statistische Analysen hautpsächlich durchführt, um zukünftige Beobachtungen zu prognostizieren.

- Benutzt man die KQ Methodik, um beispielsweise eine Regression durchzuführen, so wird die Modellbildung nur anhand der Intrapolation be-

trieben. Das beste Modell ist dasjenige, welches die vorliegenden Daten optimal approximiert.

- Bei diesem Vorgehen wird nicht gemessen, ob sich ein bestimmtes Modell auch zur Extrapolation, also zur Prognose eignet.

- Um die Prognosefähigkeit eines Modells zu überprüfen, wäre deshalb ein Gütemaß vernünftig, dass auf der Extrapolation bekannter Daten auf neue Beobachtungen beruht.

Die Prognosegüte als Modellgütemaß

- Ein solches Maß ist die Prognosegüte, gemessen als der mittlere quadratische Prognosefehler.

- **Definition:** Gegeben sei eine Stichprobe $\mathbf{x} := (x_i, y_i),\ 1 \leq i \leq n$. Weiterhin werde mittels eines statistischen Verfahrens ein Modell $\hat{f}$ aus dieser Stichprobe geschätzt, mit $y = \hat{f}(x) + \varepsilon$. Zu einem Punkt x heißt dann $y^* = \hat{f}(x)$ die Prognose an der Stelle x. Weiterhin sei eine zweite Stichprobe $\mathbf{x}^* = (x_j, y_j), n+1 \leq j \leq n+m$ von Umfang m gegeben. Dann heißt

$$F_P := \frac{1}{m} \sum_{i=n+1}^{n+m} (y_i - y_i^*)^2$$

der mittlere quadratische Prognosefehler des Modells $y^* = \hat{f}(x)$ bezüglich der Stichprobe $\mathbf{x}^*$.

- Die Stichprobe $\mathbf{x}$ heißt auch Trainingsdaten, die Stichprobe $\mathbf{x}^*$ heißt auch Testdaten.

- Ist das Ziel also die Prognose, so wäre also das Modell optimal, welches F_P minimiert!

- Eine häufige Anwendung dieses Ansatzes ist die Modellselektion. Beispielsweise ist es bei Regressionsmodellen immer eine Frage, welche Einflussfaktoren in das Modell aufgenommen werden sollen und ob auch höhere Potenzen der Einflussfaktoren eine wichtige Rolle in der Prognose zukünftiger Werte spielen.

- Mit dem mittleren quadratischen Prognosefehler gibt es nun ein Kriterium zwischen Modellen zu diskrimieren, die direkt auf der eigentlichen Zielgröße, der Prognose bzw. Extrapolation beruht und nicht nur auf der Interpolation!

Die Ansätze der Kreuzvalidierung

- Leider gibt es aber nur die eine Stichprobe und zukünftige (unbeobachtete) Punkte sind eben noch nicht zur Hand.

- Nach der Einführung von Jackknife und Bootstrap ist die Idee naheliegend, die eine Möglichkeit der Schätzung eines Prognosefehlers bietet, selbst, wenn nur eine Stichprobe vorliegt: Die vorhandenen Daten werden in 2 (oder K) Teilstichproben zerlegt!

- Danach wird auf einer (bzw. $K-1$) der Teilstichproben, dem Trainingsdatensatz, jeweils das zu untersuchende Modell geschätzt, sowie jeweils auf dem Komplement, dem Testdatensatz. der Prognosefehler F_P berechnet.

- Das Modell mit dem kleinsten F_P ist dann das Modell mit der besten Prognosefähigkeit unter den zur Verfügung stehenden Alternativen.

- Die verschiedenen Verfahren der Kreuzvalidierung unterscheiden sich durch die Zahl der Trainingsdatensätze, die aus der vorliegenden Stichprobe generiert werden.

- **Die einfache Kreuzvalidierung** zerlegt die Stichprobe in zwei Teilstichproben, dem Trainings- und dem Testdatensatz. Auf dem Trainingsdatensatz werden die Modelle geschätzt, auf dem Testdatensatz wird der Prognosefehler berechnet. Der Testdatensatz sollte nicht zu klein gewählt werden.

- **Die doppelte Kreuzvalidierung** betrachtet die beiden Stichproben wechselseitig als Trainigs- und Testdatensatz.

- Bei **der Multi-Kreuzvalidierung** wird der Datensatz wiederholt durch Ziehen ohne Zurücklegen aus der Stichprobe $\mathbf{x}$ in Trainings- und Testdatensatz aufgeteilt und jeweils die doppelte Kreuzvalidierung durchgeführt.

- **Die K-fache Kreuzvalidierung** zerlegt die Stichprobe $\mathbf{x}$ in K gleich große Teilstichproben $x_k^-, 1 \leq k \leq K$. Jede dieser Teilstichproben nimmt einmal die Rolle des Testdatensatzes ein, während das Modell auf den übrigen Daten geschätzt wird. Der Prognosefehler wird dann über alle Teilstichproben gemittelt.

- **Die leave-one-out Kreuzvalidierung** ist ein Spezialfall der K-fach Kreuzvalidierung, bei der $K = n$ gilt.

- Für große Datensätze genügt eine dreifach Kreuzvalidierung, um die Prognosegüte zu schätzen, für kleine Datensätze sollte man die leave-one-out Methode bevorzugen.

- Kreuzklassifikation ist eine sehr mächtige Methode, in dem Sinne, dass man sie an vielen Stellen in der Statistik anwenden kann. Momentan ist die intensivste Verwendung bei Klassifikationsverfahren im maschinellen Lernen zu finden. Dort wird der Prognosefehler dann als Fehlklassifikationsrate definert.

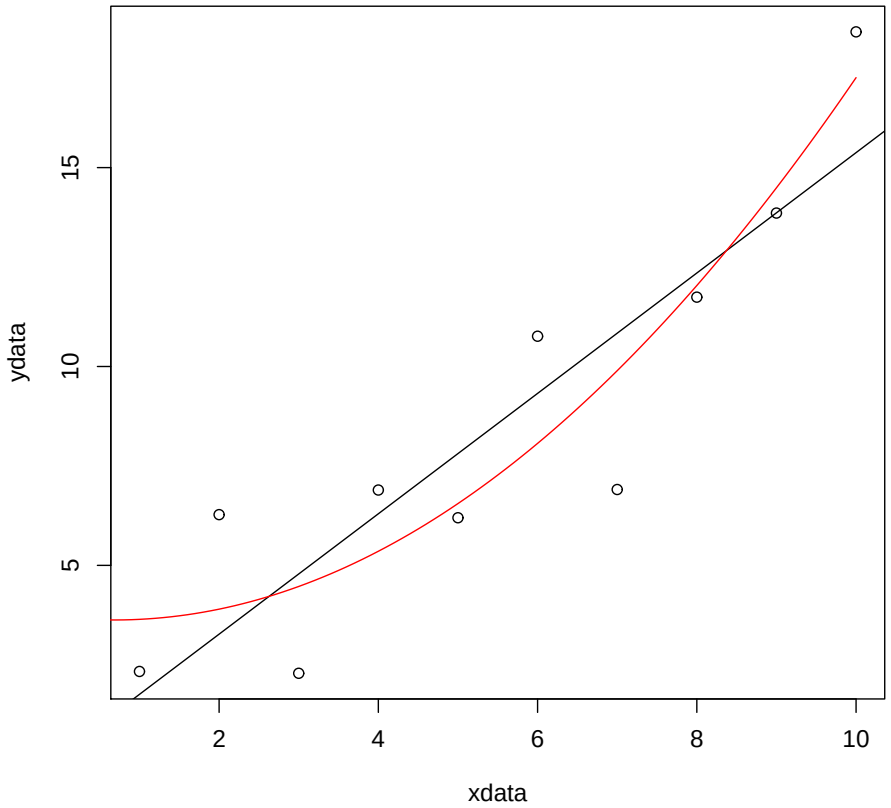
Beispiel zur Modellselektion mittels Kreuzvalidierung

- Hier soll Anhand des Beispiels der Modellselektion im Regressionsmodell das Prinzip der Kreuzvalidierung vorgeführt werden.

- Gegeben seien $n = 10$ Beobachtungen $(x_i, y_i), 1 \leq i \leq n$, die über einen unbekannten polynomialen Zusammenhang verbunden sind.

```
> xdata
 [1]  1  2  3  4  5  6  7  8  9 10
> ydata
 [1]  2.333203  6.275799  2.286967  6.893979  6.197243 10.761091
 [8] 11.743258 13.857420 18.412786
```

Plot der Daten und zweier Modelle



Modellselektion mittels Kreuzvalidierung

- In Sinne der linearen Regression gilt also

$$y_i = \sum_{q=0}^{p} \beta_q x_i^q + \varepsilon_i \; ; \varepsilon_i \text{ i.i.d.} \sim N(0, \sigma^2).$$

- Wie kann man entscheiden, welches p die besten Prognosen bietet?

Beispielhafte Berechnung des Prognosefehlers mittels einfacher Kreuzvalidierung in R

- Zunächst werden Trainings- und Testdatensatz bestimmt:

```
> training <- sample(1:10, 7, replace=FALSE)
> test <- seq(1,10)[-training]
```

- Bestimmung des mittleren quadratischen Prognosefehlers nacheinander für $p = 1, 2, 3, 4$.

```
> x <- xdata[training] ;  y <- ydata[training]
```

```
## p=1
> mean( (predict(lm(y ~ x ), data.frame(x=xdata[test])) -
                                                ydata[test])^2)
[1] 13.20345
## p=2
>  mean( (predict(lm(y ~ x +I(x^2)), data.frame(x=xdata[test]))
           - ydata[test])^2)
[1] 10.05201
## p=3
>  mean( (predict(lm(y ~ x +I(x^2) + I(x^3)),
           data.frame(x=xdata[test])) - ydata[test])^2)
[1] 8.792859
## p=4
>  mean( (predict(lm(y ~ x +I(x^2) + I(x^3) + I(x^4)),
           data.frame(x=xdata[test])) - ydata[test])^2)
[1] 9.520582
```

- Es ist zu sehen, dass der Prognosefehler im Gegensatz zur Fehlervarianz in der Regression nicht fortlaufend abnimmt, wenn weitere Regressoren in das Modell aufgenommen werden!

- Modellselektion als Kreuzvalidierung wird deshalb auch als Mittel gegen Overfitting genutzt.

- Auf Basis der durchgeführten einfachen Kreuzvalidierung würde man demnach zur Prognose zukünftiger Beobachtungen das Modell

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

  wählen und die β_i aus allen Daten schätzen!

- Es ist zu beachten, dass die Methode der Kreuzvalidierung mit jedem Gütekriterium der Modellselektion anwendbar ist!

Aufgaben zur Kreuzvalidierung

- Wiederholen Sie die Modellselektion aus dem Beispiel mit leave-one-out Kreuzvalidierung!
- Überlegen Sie hierzu, wie systematisch die nötigen Trainings- und Testdatensätze erzeugt werden können.
- Für jede Aufteilung muss das Modell neu geschätzt werden und für den jeweils ausgelassenen Punkt eine Prognose erstellt werden.
- Schließlich müssen die so gewonnenen Prognosefehler gemittelt werden.
- Lösung hier.