# Beschreibende Statistik: Häufigkeitsverteilungen mit R

## 1.1 Häufigkeitstabelle mit nominalskalierten Variablen

Einlesen der Daten (Variable als Faktor, d.h. als nominalskalierte Variable):

```
BezugUeberstund = c(1,2,2,2,1,1,2,1,1,2,2,1,1,2,2,1,1)
BezugUS=as.data.frame(BezugUeberstund )
BezugUS$BezugUeberstund=as.factor(BezugUS$BezugUeberstund)
```

Durch das Gleichheitszeichen erfolgt die Zuweisung eines Objektes zu einem Symbol.

- Dem Symbol "BezugUeberstund" wird im ersten Befehl der Vektor (= n-Tupel) aus 17 Einsen und Zweien zugewiesen.
- In der zweiten Zeile wird der Vektor in ein sogenanntes Dataframe verwandelt (Tabelle von Urdaten in R, pro Variable eine Spalte).
- Im dritten Befehl wird die metrische Variable "BezugUS$BezugUeberstund" in einen Faktor umgewandelt.

Um das jeweilig Eingelesene zu begutachten, kann man die Struktur eines Objektes mit "str" anzeigen lassen. z.B. für das Objekt "BezugUS" erhält man mittels "str" (Das Ergebnis eines Befehles steht jeweils nach zwei Rautezeichen):

```
str(BezugUS)

## 'data.frame': 17 obs. of  1 variable:
##  $ BezugUeberstund: Factor w/ 2 levels "1","2": 1 2 2 2 1 1 2 1 1 2 ...
```

Das Objekt "BezugUS" ist also ein Dataframe mit einer Variable und 17 Beobachtungen (= Zeilen, Anzahl der untersuchten Objekte der statistischen Untersuchung). Die Variable ist nominalskaliert (factor) mit zwei Ausprägungen (levels), und zwar liegen die Ausprägungen "1" und "2" vor.
Je nach Objekttyp wird durch Zuweisung Unterschiedliches abgespeichert. Oft werden von R im Hintergrund Objekte abgespeichert, mit denen man weiterrechnen kann. Diese Objekte werden durch "str" angezeigt. Beispiele folgen gleich.
Mit dem folgenden Befehle kann man die ersten 6 Zeilen eines Dataframes anzeigen lassen:

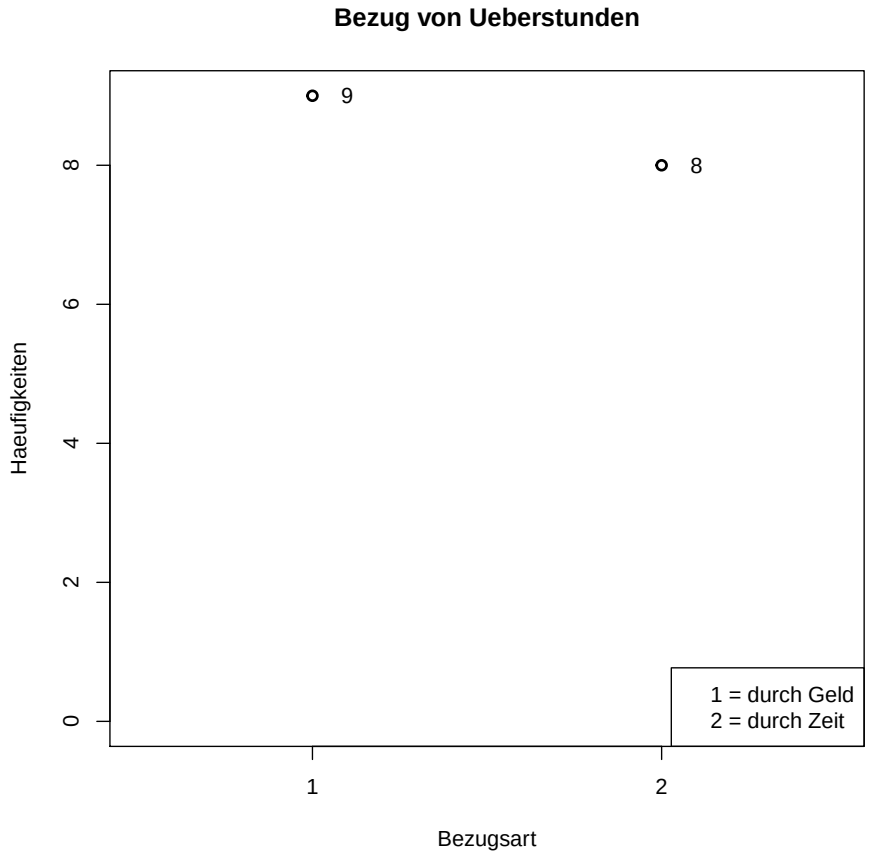
```
head(BezugUS)

##   BezugUeberstund
## 1               1
```

```
## 2                2
## 3                2
## 4                2
## 5                1
## 6                1
```

## 1.2 Darstellung des Graphen der Häufigkeitsverteilung

Will man die Häufigkeitsverteilung (ist ja eine Funktion) durch die Darstellung des Graphen der Funktion darstellen (also die entsprechenden Punkte), so kann man folgendes ausführen lassen:

```
tab=table(BezugUS$BezugUeberstund)
b=plot(tab,type = "p", xlim=c(0.5,2.5),
       main="Bezug von Ueberstunden",
       xlab="Bezugsart",
       ylab="Haeufigkeiten")
legend("bottomright",legend=c("1 = durch Geld","2 = durch Zeit"))
text(x=b+0.1,y=tab,labels=tab)
```




- "table" erstellt eine absolute Häufigkeitsverteilung.
- Weist man das Resultat von plot-Befehlen (ebenso bei barplot) einem Symbol zu (hier "b"), so wird unter "b" nicht die Graphik abgespeichert,

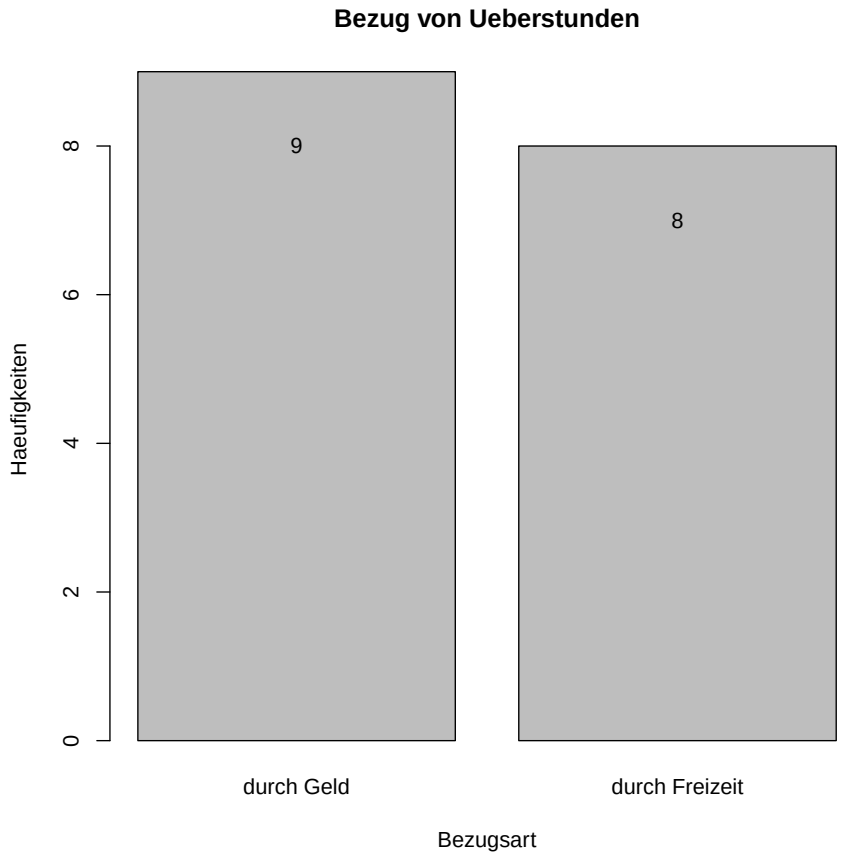
sondern ein Vektor oder eine Matrix, wodurch die Lage der Säulen auf der x-Achse angegeben wird.

- "type = p" gibt an, dass ein Punkt zu stehen hat
- "x-lim" begrenzt die Zeichnung auf der x-Achse,
- "main" gibt den Haupttitel an,
- "xlab" gibt die Beschriftung der x-Achse an, "ylab" die der y-Achse.
- "legend" erzeugt eine Legende. "bottomright" gibt an, dass diese links unten stehen soll. Man kann aber auch die Koordinaten der Lage der Legende angeben, s. "help(legend)" für die verschiedenen Möglichkeiten.
- "text" erzeugt einen Text in der Graphik, "x" gibt an, wo dieser in der x-Richtung, "y", wo dieser in der y-Richtung stehen soll. Die dritte Stelle gibt an, was stehen soll. Es könnte hier auch ein Text mit zwei Einträgen stehen, z.B. labels=c("9","8")
- Statt "b" können auch ganze Wörter gewählt werden - c, i und Namen von Funktionen des Programms sollte man vermeiden, da sie im Programm anderweitig verwendet werden.

Um die Befehle besser zu verstehen, kann man einfach welche auslassen, und schauen, was passiert.

## 1.3 Darstellung mit Säulendiagrammen

```
tab=table(BezugUS$BezugUeberstund)
a=barplot(tab,names.arg=c("durch Geld","durch Freizeit"),
          main="Bezug von Ueberstunden",
          xlab="Bezugsart",
          ylab="Haeufigkeiten")
text(x=a,y=tab-1,labels=tab)
```




"names.arg" gibt die Beschriftungen unter den Säulen an.

## 1.4 Darstellung mit Kreisdiagramm

### 1.4.1 mit absoluten Häufigkeiten

```
tab=table(BezugUS$BezugUeberstund)
beschr=paste(c("Durch Geld: ","Durch Freizeit: "),tab)
pie(tab,clockwise=T,labels=beschr,main="Bezug von Ueberstunden")
```

### 1.4.2 mit relativen Häufigkeiten in Prozent

```
tab=table(BezugUS$BezugUeberstund)
beschr=paste(c("Geld:","Freizeit:"),
             round(tab/sum(tab)*100),"%")
pie(tab,clockwise=T,labels=beschr,xlim=c(0,15),
    main="Bezug von Ueberstunden")
```

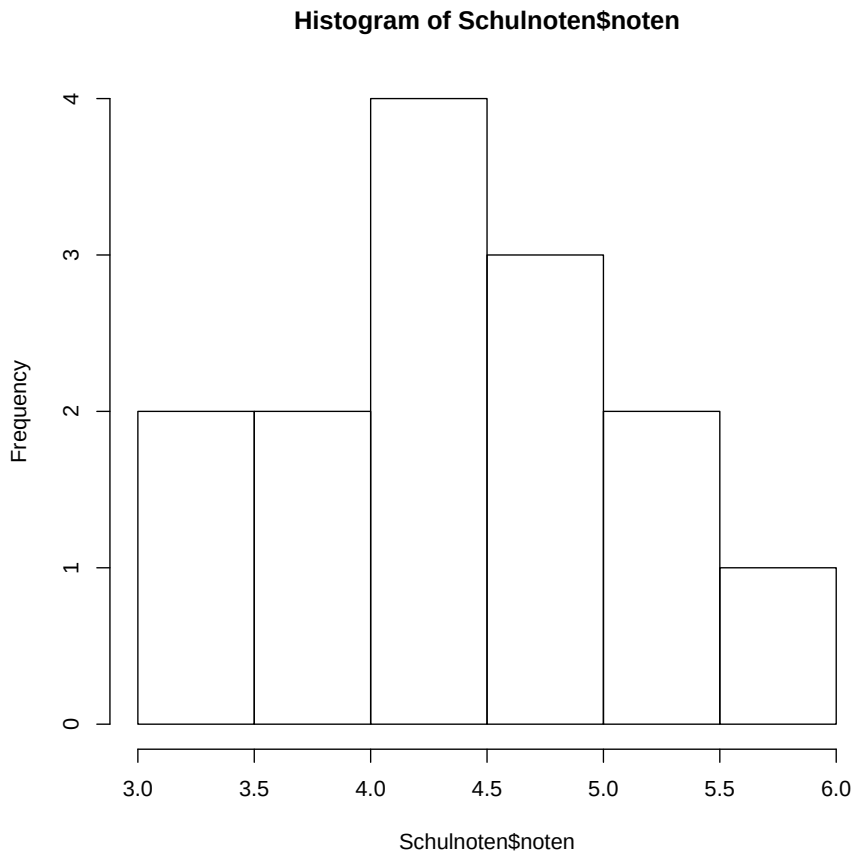

Durch "paste" werden Schriftzeichen zusammengefügt. So ergibt z.B.

```
paste(c("Haus", "Hans"), c("50","20"))

## [1] "Haus 50" "Hans 20"
```

## 1.5 Diagramme für zu klassifizierende Daten

### 1.5.1 Uebung 4 - ordinalskalierte Daten

```
noten=c(5.2,4.1,3.8,3.1,4.8,5.6,5.3,3.5,4.3,4.6,4.2,3.7,4.6,4.5)
Schulnoten=as.data.frame(noten)
a=hist(Schulnoten$noten)
```

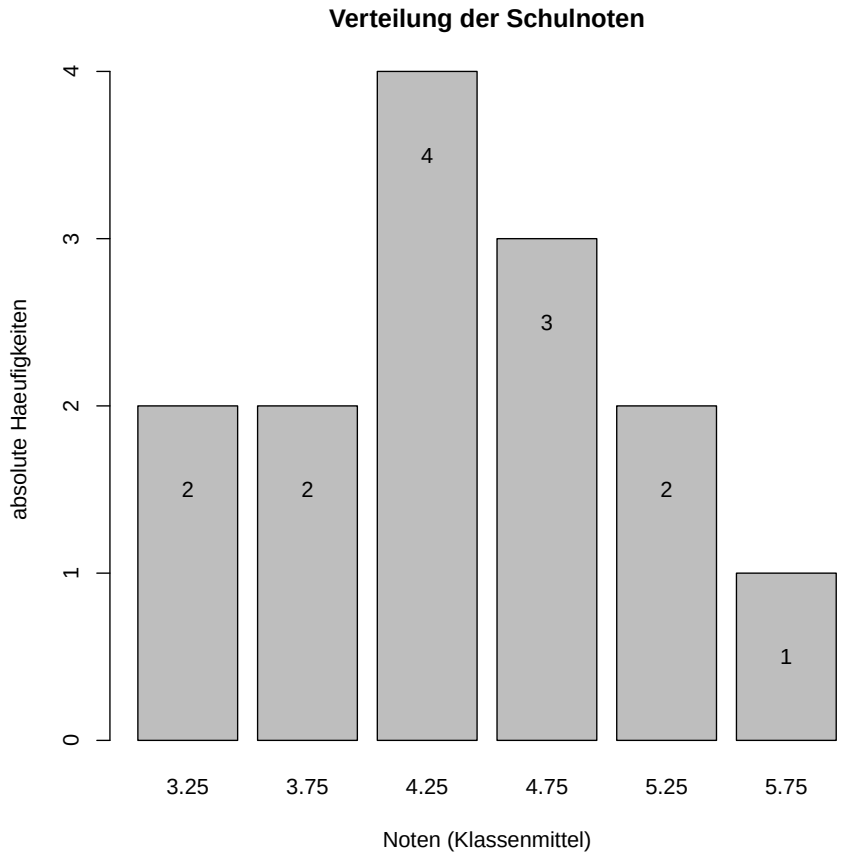



```
str(a)

## List of 6
##  $ breaks  : num [1:7] 3 3.5 4 4.5 5 5.5 6
##  $ counts  : int [1:6] 2 2 4 3 2 1
##  $ density : num [1:6] 0.286 0.286 0.571 0.429 0.286 ...
##  $ mids    : num [1:6] 3.25 3.75 4.25 4.75 5.25 5.75
##  $ xname   : chr "Schulnoten$noten"
```

```
##  $ equidist: logi TRUE
##  - attr(*, "class")= chr "histogram"
```

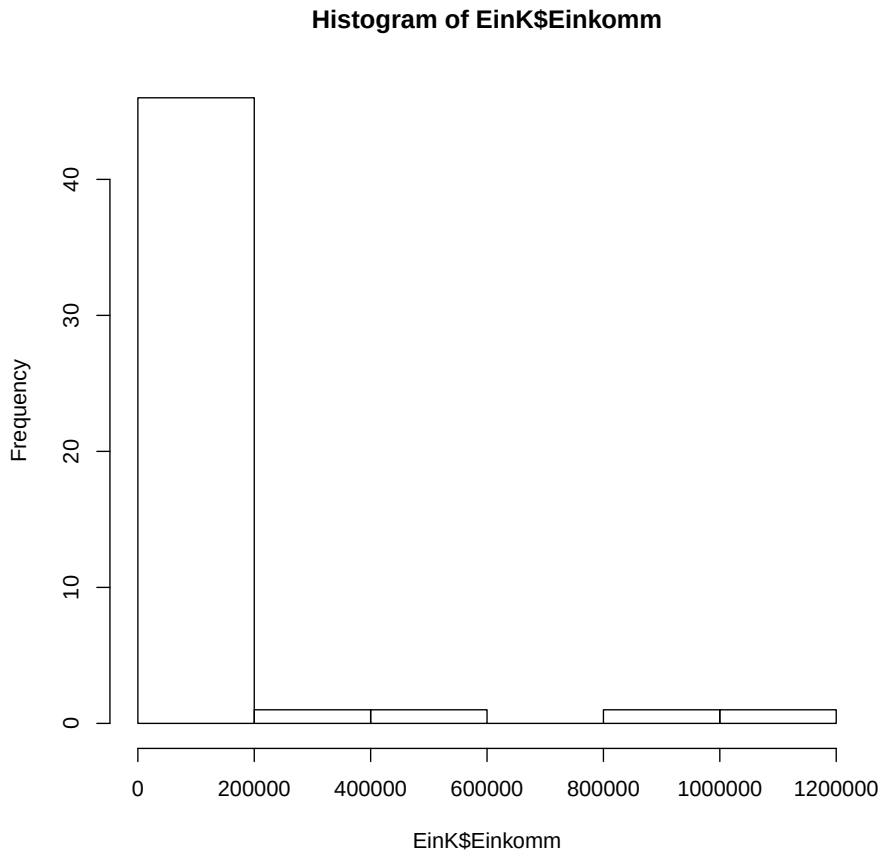
Das Histogramm wäre für ordinalskalierte Daten wie Noten nicht eine geeignete Darstellungsart. Wir verwenden die Graphik nicht. Den Befehl verwenden wir wegen den Hintergrundinformationen, die durch den Befehl erzeugt werden und die unter "str" aufgerufen werden können Möchte man z.B. die Häufigkeiten weiterverwenden, so verwendet man diese mit "a$counts". "breaks" gibt die Intervallgrenzen an, "mids" die Intervallmitten.

```
b=barplot(a$counts,main="Verteilung der Schulnoten",
          ylab="absolute Haeufigkeiten",
          xlab="Noten (Klassenmittel)",
          names.arg=a$mids)
text(x=b,y=a$counts-0.5,a$counts)
```



### 1.5.2 Uebung 6 - metrisch skalierte Daten

```
Einkomm=c(5100,8155.2,9818.9,10491.3,15414.2,17927.5,18271.2,19116.2,20504.3,
22652.7,24256.4,24933.8,25165.6,25497,28259.1,28374.4,29005.2,30391.8,
32043,32790.8,33479.3,33574.7,34620.8,35007.3,35104.2,35134,35444,
35791,37068.6,38009.3,38023.1,38558.9,41378.4,41639.5,41923.3,44239.4,
44958.3,45663.2,45956.8,49524.6,50165.9,50651.5,53899.3,54790.3,64980.4,
142576.8,213860.7,523444.5,800433,1059815.1)
EinK=as.data.frame(Einkomm)
a=hist(EinK$Einkomm)
```

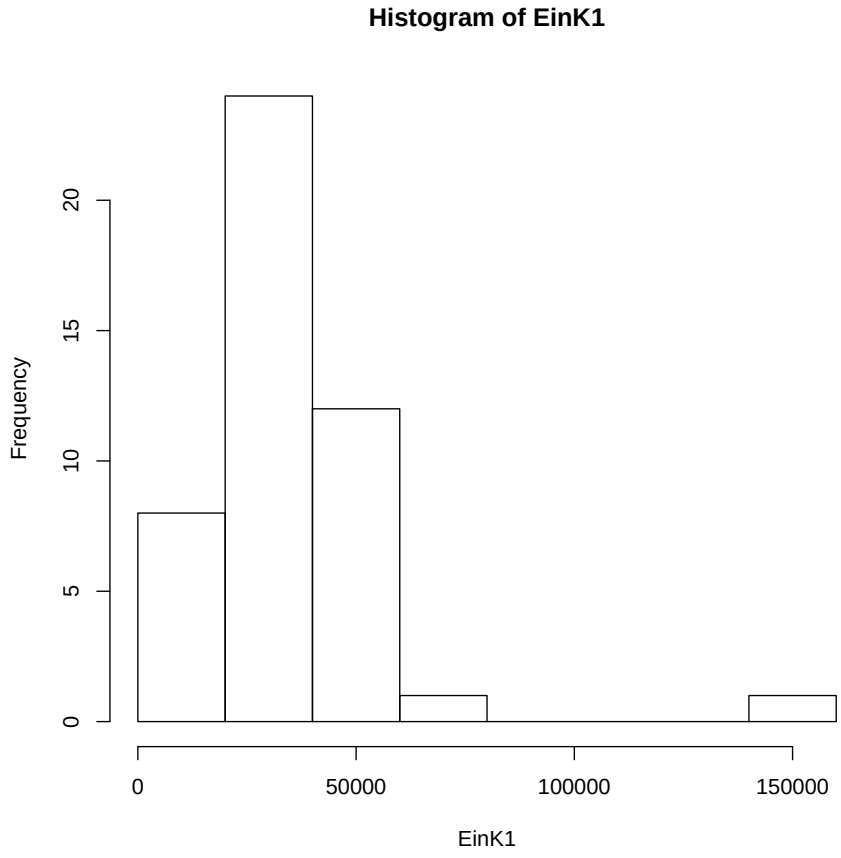



"hist" ist hier ein geeigneter Befehl, da die Daten metrisch skaliert sind. Allerdings gibtg es einige Ausreisser, die das Bild völlig verfälschen. Unter "a$counts" können wir die Verteilung der Daten genauer anschauen:

```
a$counts

## [1] 46  1  1  0  1  1
```

Man sieht, dass alle Daten ausser vier in einer Klasse massiert sein. Damit ergibt sich nicht ein informatives Bild. Am besten lässst man die grössten vier Daten weg - was natürlich kommuniziert werden muss. Man ordnet die Daten mittels:

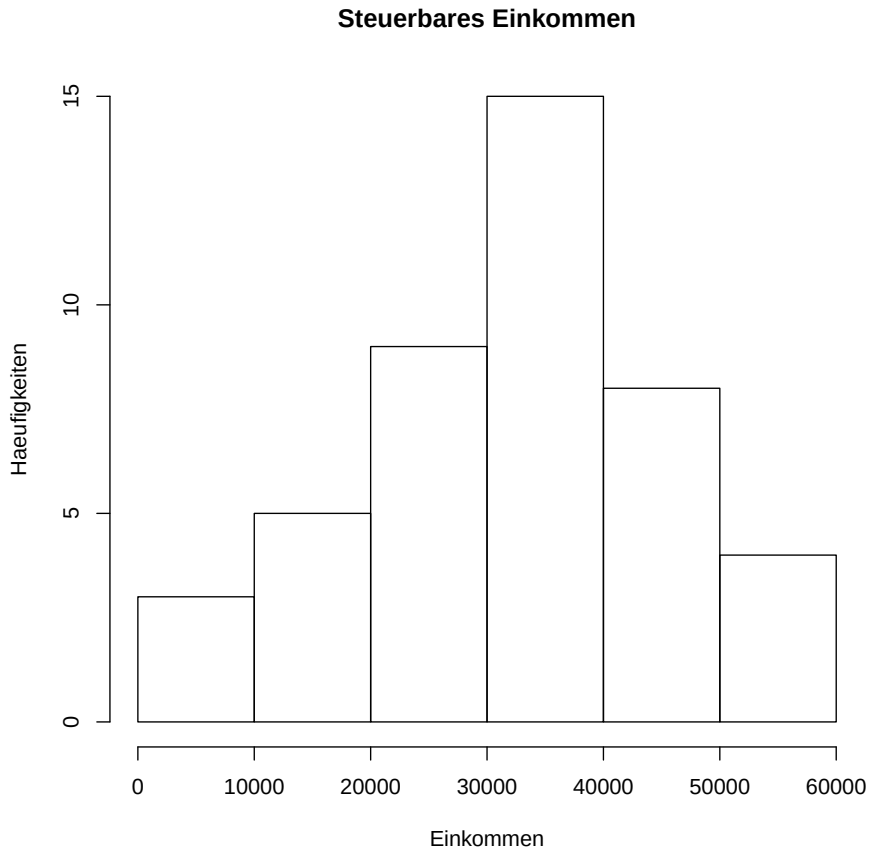
```
EinK=EinK[order(EinK$Einkomm),]
n=length(EinK)
EinK1=EinK[1:(n-4)]
hist(EinK1)
```




Wird Dataframname[order(Dataframname$Variablenname),] auf Dataframes mit mehr als einer Variable angewendet, so wird das nach der Variable beordnete Dataframe abgespeichert. Weist das Dataframe nur eine Variable auf, so wird

ein Vektor abgespeichert. Mit "length" berechnet man die Länge eines Vektors. Durch EinK[1:(n-4)] werden die Zeilen 1 bis n-4 des Dataframes in einem Vektor abgespeichert. Es zeigt sich, dass immer noch Ausreisser vorkommen, die vorher in der Hauptklasse lagen.

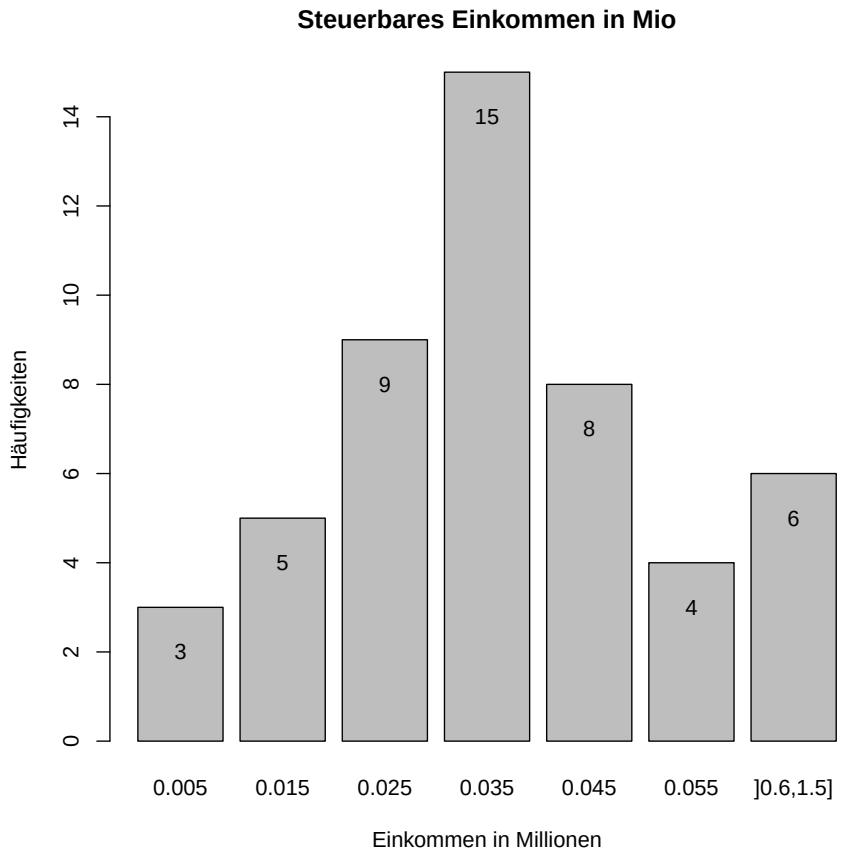
```
EinK1=EinK[1:(n-6)]
b=hist(EinK1,
        main="Steuerbares Einkommen",
        xlab="Einkommen",
        ylab="Haeufigkeiten")
```




löst das Problem.

Eine Variante würde darin bestehen, dass man ein Säulendiagramm erstellt und die Säulen entsprechend beschriftet:

```
ha=c(b$counts,6)
aa=barplot(ha,names.arg=c(b$mids/1000000,"]0.6,1.5]"),
        main="Steuerbares Einkommen in Mio",
        xlab="Einkommen in Millionen",
        ylab="Hufigkeiten")
dis=aa[2]-aa[1]
ma=max(aa)
text(x=c(aa,ma+dis),y=c(ha-1,2),labels=c(ha,6))
```




Ausser fürs letzte Intervall werden also die Klassenmitten angegeben.

## 1.6 Häufigkeitsverteilungen

1 für "sehr unzufrieden", 2 "mässig zufrieden", 3 "zufrieden" und 4 "sehr zufrieden":

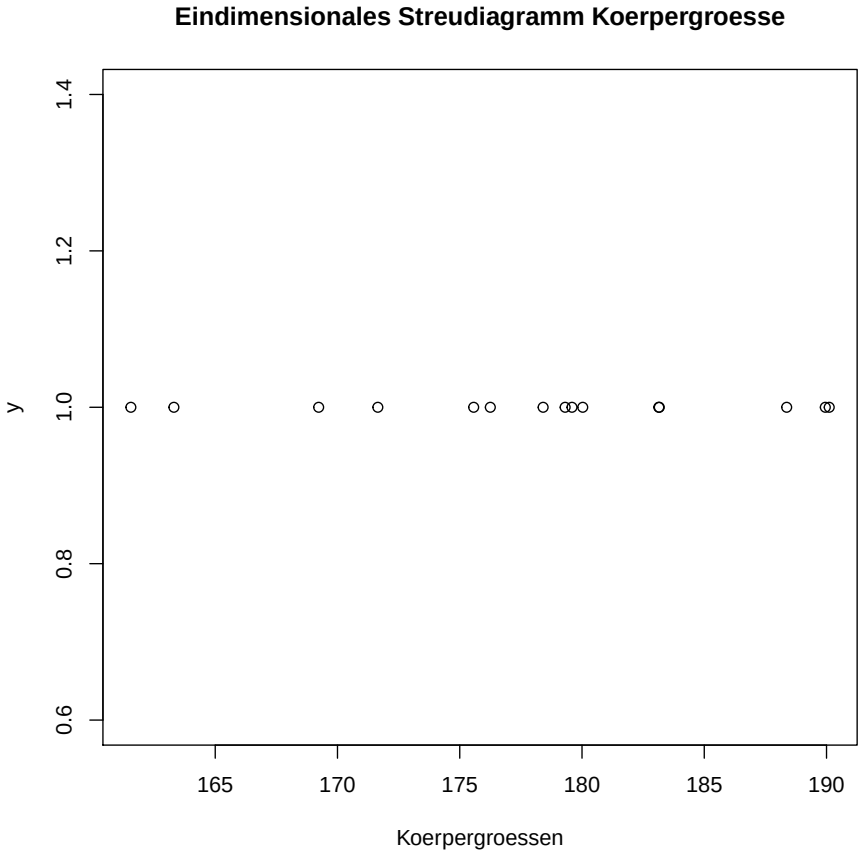
```
Zufried=c(rep(1,5),rep(2,8),rep(3,4),rep(4,2))
zufr=as.data.frame(Zufried)
tab=table(zufr$Zufried)
names(tab)=c("sehr unzufr.", "maessig zufr.",
             "zufr.", "sehr zufr.")
tab1=rbind(tab,cumsum(tab))
tab2=rbind(tab1,tab/sum(tab))
tab3=rbind(tab2,cumsum(tab/sum(tab)))
rownames(tab3)=c("absolut","kum. absolut","rel","kum. rel")
tab3
```

```
##              sehr unzufr. maessig zufr.      zufr. sehr zufr.
## absolut         5.0000000     8.0000000  4.0000000  2.0000000
## kum. absolut    5.0000000    13.0000000 17.0000000 19.0000000
## rel             0.2631579     0.4210526  0.2105263  0.1052632
## kum. rel        0.2631579     0.6842105  0.8947368  1.0000000
```

- Mit "sum" wird die Summe eines Vektors berechnet.
- Mit "cumsum" wird die kumulierte Summe berechnet.

## 1.7 Eindimensionales Streudiagramm

```
Groesse=c(176.25,179.59,163.31,189.95,161.55,169.23,188.37,183.16,
178.41,190.11,183.14,171.65,179.31,175.57,180.03)
KG=as.data.frame(Groesse)
n=length(KG$Groesse)
y=rep(1,n)
plot(KG$Groesse,y,
     main="Eindimensionales Streudiagramm Koerpergroesse",
     xlab="Koerpergroessen")
```




Die Einsen von y dienen nur dazu, die Daten von der x-Achse wegzuschieben. Der Befehl "rep" erzeugt einen Vektor mit einer Konstante. Im Beispiel wird n mal die 1 erzeugt.

## 1.8 Empirische Verteilungsfunktion

Punkteform:

```
n=length(KG$Groesse)
Grgeord=KG$Groesse[order(KG$Groesse)]
plot(Grgeord,seq(1/n,1,1/n),
     main="empirische Verteilungsfunktion Koerpergroesse",
     xlab="Koerpergroessen",
     ylab="kum. rel. Haeufigkeiten")
```


"seq" erzeugt eine Folge von der ersten Angabe zur zweiten in den Schritten der dritten Angabe. Im Beipiel wird ein Vektor von Zahlen von 1/n bis 1 in gleichen Schritten von 1/n erzeugt.

Treppenfunktionsform:

```
plot(ecdf(KG$Groesse),
     main="empirische Verteilungsfunktion Koerpergroesse",
     xlab="Koerpergroessen",
     ylab="kum. rel. Haeufigkeiten")
```