Stochastische Ausarbeitung

Bela

10.04.2023

Contents

1 Einleitung

Viel Spaß mit meiner Ausarbeitung :).

2 Aufagabe 1

2.1 χ^2-Anpassungstest

In der Multinomialverteilung haben wir 4 Kategorien, welche jeweils Binomial verteilt sind. Für große n ist die Binomialverteilung normalverteilt mit $\mu = n \cdot p$ und $\sigma = \sqrt{n \cdot p \cdot (1-p)}$. Sei a_1, a_2, a_3, a_4 die Anzahl der Beobachtungen in den Kategorien. Damit ist $\dfrac{a_j - n \cdot p_j}{\sqrt{n \cdot p_j \cdot (1-p_j)}} \sim N(0,1)$. Also ist $\dfrac{(a_j - n \cdot p_j)^2}{n \cdot p_j \cdot (1-p_j)} \sim (N(0,1))^2$

Damit ist die Summe $\sum_{j=1}^{4} \dfrac{(a_j - n \cdot p_j)^2}{n \cdot p_j \cdot (1-p_j)} \sim \chi_3^2$.

Da die p-Werte der χ^2-Verteilung bekannt sind, kann so ein einfacher Hypothesentest durchgeführt werden:

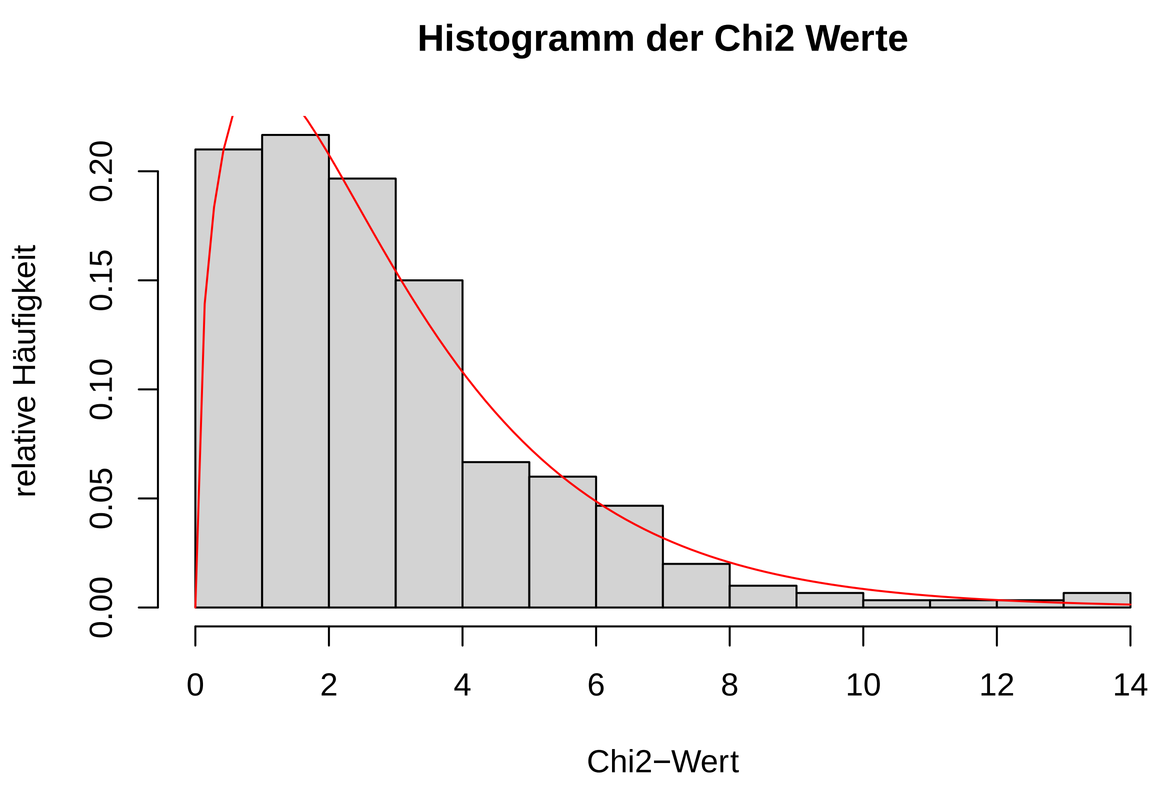
$$H_0 : p_1 = \frac{1}{8}, p_2 = \frac{1}{4}, p_3 = \frac{1}{2}, p_4 = \frac{1}{8}$$
$$H_1 : Nichtallep_j habenwertewie H_0$$

Simulieren wir nun den Versuch:

```
set.seed(123)
simulateAnpassungstest <- function(){
  n <- 1000
  p <- c(1/8, 1/4, 1/2, 1/8)
  a <- rmultinom(1, n, p)
  sum((a - n*p)^2/(n*p))
}
```

```
results <- c()
for (i in 1:300){
  results =c(results ,simulateAnpassungstest())
}
```

```
hist(results, freq=FALSE, main = "Histogramm der Chi2 Werte", xlab = "Chi2-Wert", ylab = "relative Häuf
curve(dchisq(x, 3), add = TRUE, col = "red")
```



2.2 $t-$Test mit unbekannter Varianz

Beim zweiseitigen $t-$Test mit unbekannter Varianz wird die Nullhypothese $H_0 : \mu = \mu_0$ gegen die Alternative $H_1 : \mu \neq \mu_0$ getestet. Als Teststatistik wird $\dfrac{\sqrt{n}(\bar{X}_n - \mu_0)}{S_n}$ verwendet.
Herleitung:

Seien $X_1, X_2, \ldots, X_n \sim N(\mu, \sigma^2)$ unabhängig und identisch verteilt.
Das arithmetische Mittel $\bar{X}_n$ ist normalverteilt mit $\mu = \mu$ und $\sigma = \frac{\sigma}{\sqrt{n}}$.

Bei bekannter Varianz wäre $\sqrt{n}\frac{\bar{X}_n - \mu_0}{\sigma} \sim N(0,1)$. Wir müssen allerdings die Varianz mit der empirischen Varianz ersetzen, also ist $\sqrt{n}\frac{\bar{X}_n - \mu_0}{S_n} \sim t_{n-1}$.(T-Verteilung folgt aus Störung durch die Varianzschätzung)

Das heißt für den Test müssen wir nur die Teststatistik $T = \frac{\sqrt{n}(\bar{X}_n - \mu_0)}{S_n}$ berechnen und anschließend deren p-Wert bestimmen:

Test:

```
set.seed(123)
mu0 <- 2
sigma <- 4
data <- rnorm(1000, mu0, sigma)
alpha <- 0.05

dataVar <- var(data)
dataMean <- mean(data)

T <- sqrt(length(data))*(dataMean-mu0)/sqrt(dataVar)

#Teststatistik
print(paste("T = ", T))
```

```
## [1] "T =  0.514279000759497"
```

```
#p-Wert von T
print(paste("p-Wert von T = ", pt(T, length(data)-1, lower.tail = FALSE)))
```

```
## [1] "p-Wert von T =  0.303585342303021"
```

```
#Testresultat
if (pt(T, length(data)-1, lower.tail = FALSE) > 1-alpha){
  print("Nullhypothese wird verworfen")
} else {
  print("Nullhypothese wird nicht verworfen")
}
```

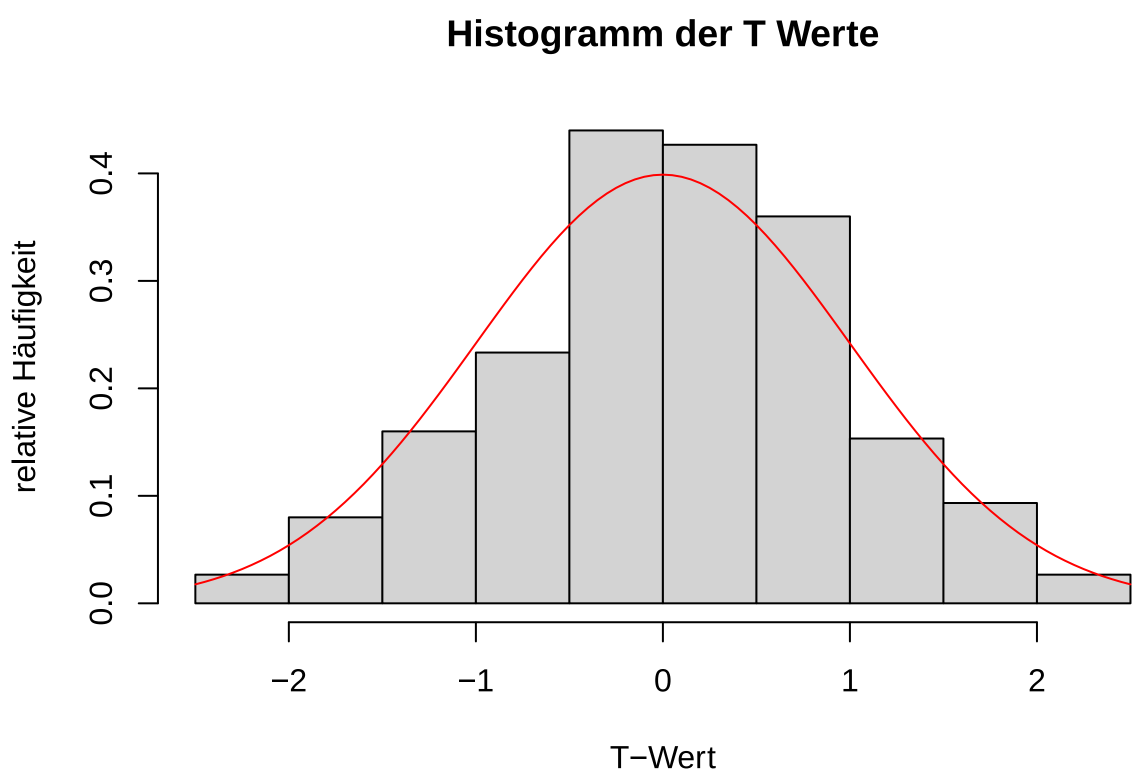
```
## [1] "Nullhypothese wird nicht verworfen"
```

Verteilung der Teststatistik:

```
set.seed(123)
Tdata = c()

for (i in 1:300){
  data <- rnorm(1000, mu0, sigma)
  mu0 <- 2
  alpha <- 0.05
```

```r
  T <- sqrt(length(data))*(mean(data)-mu0)/sqrt(var(data))
  Tdata = c(Tdata, T)
}
hist(Tdata, freq=FALSE, main = "Histogramm der T Werte", xlab = "T-Wert", ylab = "relative Häufigkeit")
curve(dt(x, length(data)-1), add = TRUE, col = "red")
```



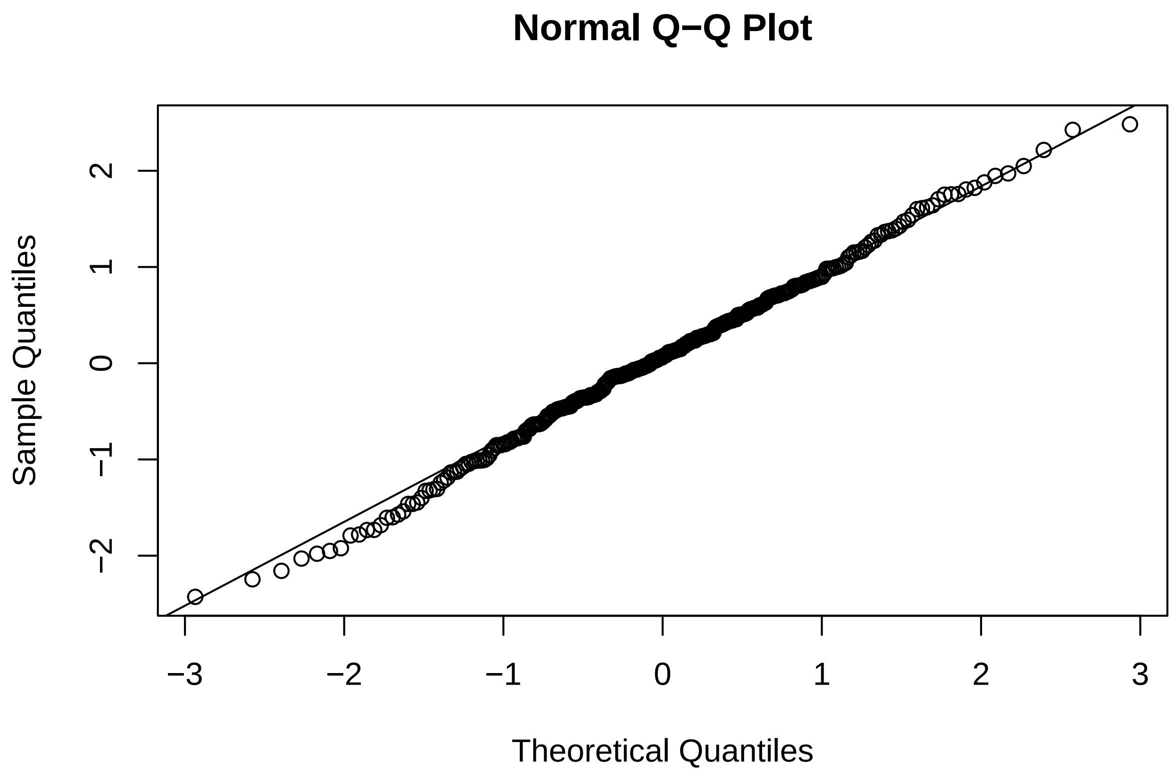

```r
#Kolmogorov-Smirnoff-Test
print(ks.test(Tdata, "pt", length(data)-1))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  Tdata
## D = 0.068075, p-value = 0.124
## alternative hypothesis: two-sided
```

Da der p-Wert des Kolmogorov-Smirnoff-Tests kleiner als $1 - \alpha = 0.95$ ist, kann die Nullhypothese nicht verworfen werden, also ist die Verteilung der Teststatistik t-verteilt.

QQ-Plot:

```r
qqnorm(Tdata)
qqline(Tdata)
```



Ergebnis: Alles deutet darauf hin, dass die Teststatistik t-verteilt ist.

2.3 $t-$Test mit bekannter Varianz

Genau wie oben, nur dass wir die Varianz nicht schätzen müssen. Deshalb ist die Teststatistik $\sqrt{n}\frac{\bar{X}_n - \mu_0}{\sigma} \sim N(0,1)$. Der Rest bleibt gleich:

```
set.seed(123)
mu0 <- 2
sigma <- 4
data <- rnorm(1000, mu0, sigma)
alpha <- 0.05

dataMean <- mean(data)

T <- sqrt(length(data))*(dataMean-mu0)/sigma

#Teststatistik
print(paste("T = ", T))
```

```
## [1] "T =  0.51000790152087"
```

```
#p-Wert von T unter Normalverteilung
print(paste("p-Wert von T = ", pnorm(T, length(data)-1, lower.tail = FALSE)))
```

```
## [1] "p-Wert von T =  1"
```

```r
#Testresultat
if (pnorm(T, lower.tail = FALSE) > 1-alpha){
  print("Nullhypothese wird verworfen")
} else {
  print("Nullhypothese wird nicht verworfen")
}
```

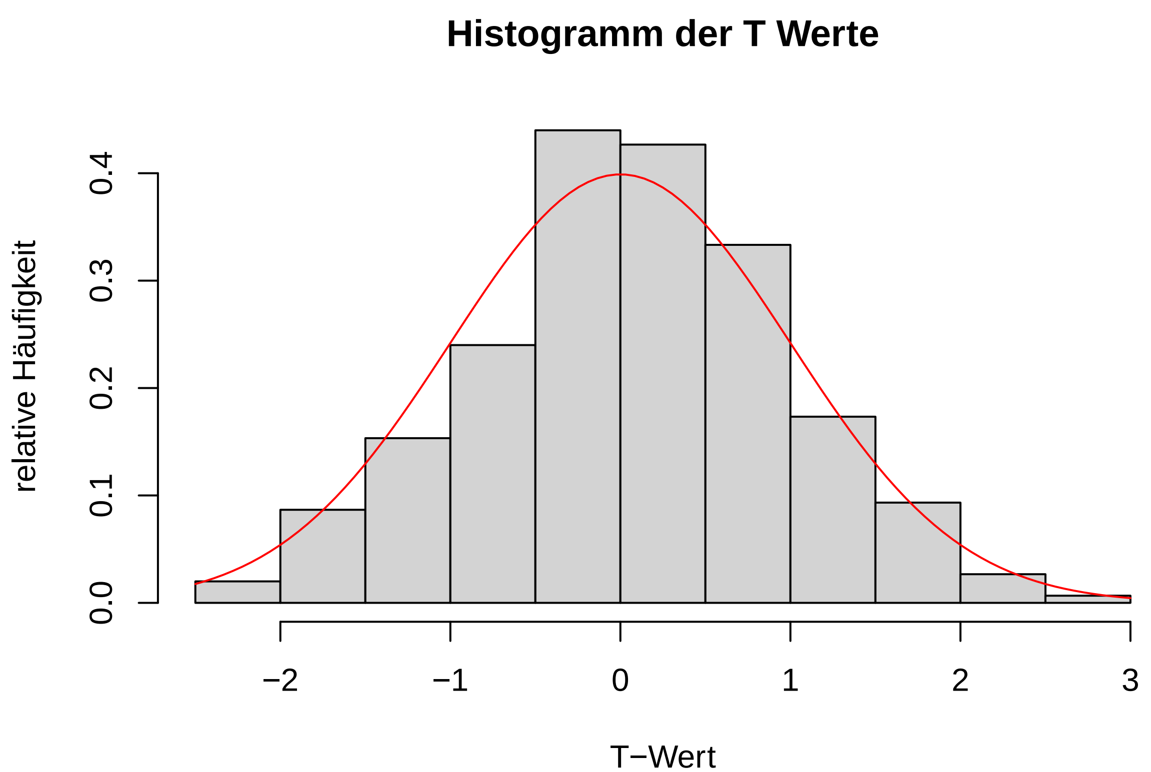
```
## [1] "Nullhypothese wird nicht verworfen"
```

Verteilung der Teststatistik:

```r
set.seed(123)
Tdata = c()

for (i in 1:300){
  data <- rnorm(1000, mu0, sigma)
  alpha <- 0.05

  T <- sqrt(length(data))*(mean(data)-mu0)/sigma
  Tdata = c(Tdata, T)
}
hist(Tdata, freq=FALSE, main = "Histogramm der T Werte", xlab = "T-Wert", ylab = "relative Häufigkeit")
curve(dnorm(x), add = TRUE, col = "red")
```



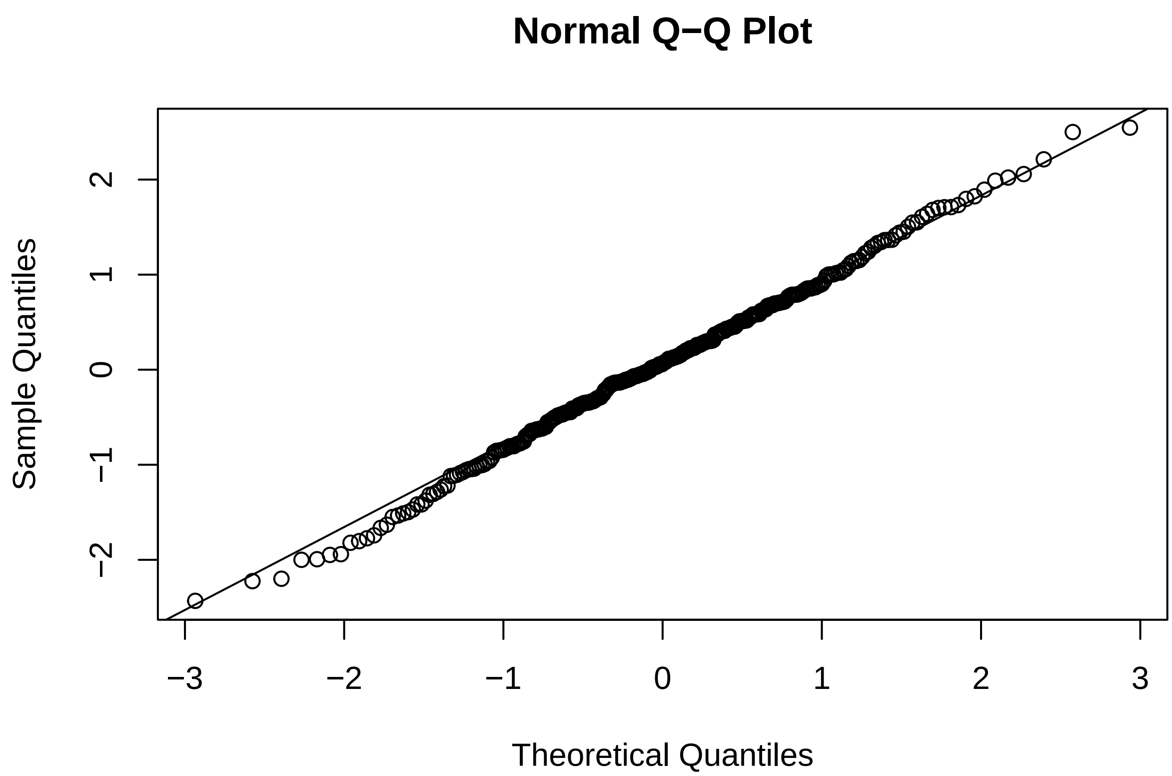
```r
#Kolmogorov-Smirnoff-Test
print(ks.test(Tdata, "pnorm", lower.tail = FALSE))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  Tdata
## D = 0.99456, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

Da der p-Wert des Kolmogorov-Smirnoff-Tests kleiner als $1 - \alpha = 0.95$ ist, kann die Nullhypothese nicht verworfen werden, also ist die Verteilung der Teststatistik t-verteilt.

QQ-Plot:

```r
qqnorm(Tdata)
qqline(Tdata)
```



Ergebnis: Alles deutet darauf hin, dass die Teststatistik normalverteilt ist.

3 Aufgabe 2

Lineare Regression mit R

3.1 Daten einlesen

```
data <- read.table("Datensaetze/wine.txt", header = TRUE)
```

3.2 Überblick über die Daten

```
head(data)
```

```
##   year price temp h.rain w.rain
## 1 1952    37 17.1    160    600
## 2 1953    63 16.7     80    690
## 3 1955    45 17.1    130    502
## 4 1957    22 16.1    110    420
## 5 1958    18 16.4    187    582
## 6 1959    66 17.5    187    485
```

```
summary(data)
```

```
##       year          price             temp           h.rain     
##  Min.   :1952   Min.   : 10.00   Min.   :15.00   Min.   : 38.0  
##  1st Qu.:1960   1st Qu.: 14.00   1st Qu.:16.15   1st Qu.: 88.0  
##  Median :1967   Median : 22.00   Median :16.40   Median :123.0  
##  Mean   :1967   Mean   : 28.81   Mean   :16.47   Mean   :144.8  
##  3rd Qu.:1974   3rd Qu.: 35.00   3rd Qu.:17.00   3rd Qu.:185.5  
##  Max.   :1980   Max.   :100.00   Max.   :17.60   Max.   :292.0  
##      w.rain     
##  Min.   :376.0  
##  1st Qu.:543.5  
##  Median :600.0  
##  Mean   :608.4  
##  3rd Qu.:705.5  
##  Max.   :830.0  
```

3.3 Lineares Modell

```
model <- lm(price ~ temp + h.rain + w.rain, data = data)
summary(model)
```

```
## 
## Call:
## lm(formula = price ~ temp + h.rain + w.rain, data = data)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -16.580  -8.601  -4.057   6.813  29.064 
## 
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -365.45179   77.63849  -4.707 9.66e-05 ***
## temp          22.50086    4.28502   5.251 2.51e-05 ***
## h.rain        -0.09296    0.03746  -2.481   0.0208 *
## w.rain         0.06103    0.02247   2.717   0.0123 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.33 on 23 degrees of freedom
## Multiple R-squared:  0.6421, Adjusted R-squared:  0.5954
## F-statistic: 13.75 on 3 and 23 DF,  p-value: 2.389e-05
```

In dem linearen Modell hat der Temperaturkoeffzient ein positives Vorzeichen, was bedeutet, dass der Preis mit steigender Temperatur steigt.

Der Koeffizient für Niederschlag bei der Ernte hat ein negatives Vorzeichen, was bedeutet, dass der Preis mit steigendem Niederschlag bei der Ernte sinkt. Der Koeffizient für Niederschlag im Winter hat ein positives Vorzeichen, was bedeutet, dass der Preis mit steigendem Niederschlag im Winter steigt.