Serie 13

Aufgabe 13.1

Wir betrachten den Datensatz `Carseats.csv`. Wir möchten **`Sales`** (Anzahl Kinderautositze) aufgrund von verschiedenen Prädiktoren in 400 verschiedenen Standorten vorhersagen. Die Beschreibung der verschiedenen Variablen finden Sie unter

https://rdrr.io/cran/ISLR/man/Carseats.html

Der Datensatz enthält qualitative Prädiktoren, wie **`ShelveLoc`** als Indikator der Lage im Gestell, das heisst der Platz in einem Geschäft, wo der Autositz ausgestellt ist. Der Prädiktor nimmt die drei Werte **`Bad`**, **`Medium`** und **`Good`** an. Für qualitative Variablen generiert **`Python`** Dummy-Variablen automatisch.

a) Lesen Sie Datei ein.

b) Finden Sie mit **`ols()`** ein multiples Regressionsmodell um **`Sales`** aus **`Price`**, **`Urban`** und **`US`** vorherzusagen.

c) Interpretieren Sie die Koeffizienten in diesem Modell. Achten Sie darauf, dass einige Variablen qualitativ sind.

d) Schreiben Sie das Modell in Gleichungsform. Achten Sie darauf, dass Sie die qualitativen Variablen richtig behandeln.

e) Für welche Prädiktoren kann die Nullhypothese $H_0 : \beta_j = 0$ verworfen werden?

f) Auf der Basis der vorhergehenden Frage, finden Sie ein kleineres Modell, das nur Prädiktoren verwendet für die es Hinweise auf einen Zusammenhang mit der Zielvariablen gibt.

g) Wie genau passen die Modelle in a) und e) die Daten an?

Aufgabe 13.2

Wir führen noch eine multiple lineare Regression für `Auto` aus der letzten Übung durch.

```python
import pandas as pd
import statsmodels.api as sm
from statsmodels.graphics.regressionplots import abline_plot
from statsmodels.formula.api import ols
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv("../../../Themen/Einfache_Lineare_Regression/Jupyter_Notebooks_de/Auto.csv")

df.columns


## Index(['Unnamed: 0', 'X1', 'mpg', 'cylinders', 'displacement', 'horsepower',
##        'weight', 'acceleration', 'year', 'origin', 'name'],
##       dtype='object')


df = df.drop(["Unnamed: 0", "X1", "name"], axis=1)
df.head()


##      mpg  cylinders  displacement  ...  acceleration  year  origin
## 0  7.650          8         307.0  ...          12.0    70       1
## 1  6.375          8         350.0  ...          11.5    70       1
## 2  7.650          8         318.0  ...          11.0    70       1
## 3  6.800          8         304.0  ...          12.0    70       1
## 4  7.225          8         302.0  ...          10.5    70       1
##
## [5 rows x 8 columns]
```

a) Eliminieren Sie mit der backward-Methode im Jupyter Notebook `backward_py.ipynb` schrittweise alle Variablen, bis alle Variablen p-Werte unter 0.05 haben.

b) Bestimmen Sie mit dem Programm im Jupyter Notebook `r_squared_adj_py.ipynb` das optimale Modell.

Kurzlösungen vereinzelter Aufgaben

Musterlösungen zu Serie 13

Lösung 13.1

a) Datensatz:

```
import pandas as pd
import statsmodels.formula.api as smf
import numpy as np

df = pd.read_csv("../../../Themen/Einfache_Lineare_Regression/Jupyter_Notebooks_de
0", axis=1)

df.head()
```

```
##    Sales  CompPrice  Income  Advertising  ...  Age  Education Urban   US
## 0   9.50        138      73           11  ...   42         17   Yes  Yes
## 1  11.22        111      48           16  ...   65         10   Yes  Yes
## 2  10.06        113      35           10  ...   59         12   Yes  Yes
## 3   7.40        117     100            4  ...   55         14   Yes  Yes
## 4   4.15        141      64            3  ...   38         13   Yes   No
##
## [5 rows x 11 columns]
```

b) Output:

```
fit = smf.ols("Sales~Price+Urban+US", data=df).fit()

fit.summary()
```

```
## <class 'statsmodels.iolib.summary.Summary'>
## """
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                  Sales   R-squared:                       0.239
## Model:                            OLS   Adj. R-squared:                  0.234
## Method:                 Least Squares   F-statistic:                     41.52
## Date:                Tue, 19 May 2020   Prob (F-statistic):           2.39e-23
## Time:                        06:55:36   Log-Likelihood:                -927.66
## No. Observations:                 400   AIC:                             1863.
## Df Residuals:                     396   BIC:                             1879.
## Df Model:                           3
## Covariance Type:            nonrobust
## ================================================================================
##                    coef    std err          t      P>|t|      [0.025      0.975]
## --------------------------------------------------------------------------------
## Intercept       13.0435      0.651     20.036      0.000      11.764      14.323
## Urban[T.Yes]    -0.0219      0.272     -0.081      0.936      -0.556       0.512
## US[T.Yes]        1.2006      0.259      4.635      0.000       0.691       1.710
## Price           -0.0545      0.005    -10.389      0.000      -0.065      -0.044
## ==============================================================================
## Omnibus:                        0.676   Durbin-Watson:                   1.912
## Prob(Omnibus):                  0.713   Jarque-Bera (JB):                0.758
## Skew:                           0.093   Prob(JB):                        0.684
## Kurtosis:                       2.897   Cond. No.                         628.
## ==============================================================================
##
## Warnings:
```

```
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
## """
```

c) Interpretation der Koeffizienten:

- Der Koeffizient 13.04 ist ein bisschen schwierig zu interpretieren. Gemäss dem Modell unter d) sind dies die mittleren Verkaufszahlen in Geschäften, die in ländlichen Gegenden ausserhalb der USA erreicht werden, wobei der Preis der Kindersitze noch \$0 ist (nicht sehr realsistisch).
- Der Koeffizient -0.05 besagt, dass für eine Zunahme von einem Dollar durchschnittlich 0.05 Einheiten Kindersitze weniger verkauft werden.
- Der Koeffizient -0.021 besagt, dass verglichen zu ländlichen Gegenden durchschnittlich 0.021 Einheiten weniger verkauft werden. Der p-Wert ist allerdings sehr hoch, so dass dies eher eine zufällige Abweichung ist.
- Der Koeffizient 1.2 besagt, dass verglichen zu Geschäften ausserhalb der USA, 1.2 Einheiten mehr verkauft werden. Vielleicht sind in den USA Kindersitze pflicht.

d) Modell: Für `Urban` wählen wir die Dummy-Variable:

$$x_{2i} = \begin{cases} 1 & \text{falls } i\text{-te Person lebt in der Stadt} \\ 0 & \text{falls } i\text{-te Person lebt auf dem Land} \end{cases}$$

Für `US` wählen wir die Dummy-Variable

$$x_{3i} = \begin{cases} 1 & \text{falls } i\text{-te Person lebt in den USA} \\ 0 & \text{falls } i\text{-te Person lebt nicht in den USA} \end{cases}$$

Das Modell lautet dann

$$\begin{aligned} y_i &= \beta_0 + \beta_1 \cdot \texttt{Price} + \beta_2 x_{2i} + \beta_3 x_{3i} + \epsilon_i \\ &= \beta_0 + \beta_1 \cdot \texttt{Price} + \begin{cases} \beta_2 + \beta_3 + \epsilon_i & \text{falls } i\text{-te Person urban in den USA lebt} \\ \beta_2 + \epsilon_i & \text{falls } i\text{-te Person urban nicht in den USA lebt} \\ \beta_3 + \epsilon_i & \text{falls } i\text{-te Person ländlich in den USA lebt} \\ \epsilon_i & \text{falls } i\text{-te Person ländlich nicht in den USA lebt} \end{cases} \end{aligned}$$

e) Für alle ausser `Urban`

f) Output:

```
fit = smf.ols("Sales~Price+US", data=df).fit()

fit.summary()
```

```
## <class 'statsmodels.iolib.summary.Summary'>
## """
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                  Sales   R-squared:                       0.239
## Model:                            OLS   Adj. R-squared:                  0.235
## Method:                 Least Squares   F-statistic:                     62.43
## Date:                Tue, 19 May 2020   Prob (F-statistic):           2.66e-24
## Time:                        06:55:36   Log-Likelihood:                -927.66
## No. Observations:                 400   AIC:                             1861.
## Df Residuals:                     397   BIC:                             1873.
## Df Model:                           2
## Covariance Type:            nonrobust
## ==============================================================================
##                  coef    std err          t      P>|t|      [0.025      0.975]
## ------------------------------------------------------------------------------
## Intercept     13.0308      0.631     20.652      0.000      11.790      14.271
## US[T.Yes]      1.1996      0.258      4.641      0.000       0.692       1.708
## Price         -0.0545      0.005    -10.416      0.000      -0.065      -0.044
## ==============================================================================
## Omnibus:                        0.666   Durbin-Watson:                   1.912
## Prob(Omnibus):                  0.717   Jarque-Bera (JB):                0.749
## Skew:                           0.092   Prob(JB):                        0.688
## Kurtosis:                       2.895   Cond. No.                         607.
## ==============================================================================
##
## Warnings:
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
## """
```

Modell: Für **US** wählen wir die Dummy-Variable

$$x_{2i} = \begin{cases} 1 & \text{falls } i\text{-te Person lebt in den USA} \\ 0 & \text{falls } i\text{-te Person lebt nicht in den USA} \end{cases}$$

Das Modell lautet dann

$$\begin{aligned} y_i &= \beta_0 + \beta_1 \cdot \texttt{Price} + \beta_2 x_{2i} + \epsilon_i \\ &= \beta_0 + \beta_1 \cdot \texttt{Price} + \begin{cases} \beta_2 + \epsilon_i & \text{falls } i\text{-te Person in den USA lebt} \\ \epsilon_i & \text{falls } i\text{-te Person nicht in den USA lebt} \end{cases} \\ &= 13.03 - 0.055 \cdot \texttt{Price} + \begin{cases} 1.2 + \epsilon_i & \text{falls } i\text{-te Person in den USA lebt} \\ \epsilon_i & \text{falls } i\text{-te Person nicht in den USA lebt} \end{cases} \end{aligned}$$

g) Bei beiden Modellen ist zwar der Zusammenhang belegt (p-Wert für F-Wert praktisch 0), aber wenn wir die R^2-Werte betrachten, so ist der mit 0.2393 relativ schlecht. Das heisst, obwohl der Zusammenhang gesichtert ist die Passung schlecht, da nur 23 % der Variabilität der **Sales** durch das Modell erklärt werden kann.

Lösung 13.2

a) Output:

```
pd.options.display.float_format = '{:.10f}'.format
predictors = set(df.columns)
predictors.remove("mpg")
selected = list(predictors)
formula = "{} ~ {}".format("mpg", ' + '.join(selected))
ols(formula, data=df).fit().pvalues
```

```
## Intercept      0.0002401841
## weight         0.0000000000
## acceleration   0.4154780178
## horsepower     0.2196328232
## origin         0.0000004666
## cylinders      0.1277964676
## year           0.0000000000
## displacement   0.0084446495
## dtype: float64
```

Wir eliminieren `acceleration`:

```
predictors.remove("acceleration")
selected = list(predictors)
formula = "{} ~ {}".format("mpg", ' + '.join(selected))
ols(formula, data=df).fit().pvalues
```

```
## Intercept      0.0002223379
## weight         0.0000000000
## horsepower     0.0280312088
## origin         0.0000004434
## cylinders      0.1172362373
## year           0.0000000000
## displacement   0.0102873320
## dtype: float64
```

Wir eliminieren `cylinders`:

```
predictors.remove("cylinders")
selected = list(predictors)
formula = "{} ~ {}".format("mpg", ' + '.join(selected))
ols(formula, data=df).fit().pvalues
```

```
## Intercept      0.0000615641
## weight         0.0000000000
## horsepower     0.0427696404
## origin         0.0000008804
## year           0.0000000000
## displacement   0.0406407586
```

```
## dtype: float64
```

b) Output:

```
import statsmodels.formula.api as smf

def forward_selected(data, response):
    """Linear model designed by forward selection.

    Parameters:
    -----------
    data : pandas DataFrame with all possible predictors and response

    response: string, name of response column in data

    Returns:
    --------
    model: an "optimal" fitted statsmodels linear model
           with an intercept
           selected by forward selection
           evaluated by adjusted R-squared
    """
    remaining = set(data.columns)
    remaining.remove(response)
    selected = []
    current_score, best_new_score = 0.0, 0.0
    while remaining and current_score == best_new_score:
        scores_with_candidates = []
        for candidate in remaining:
            formula = "{} ~ {}".format(response,
                                       ' + '.join(selected +
[candidate]))
            score = smf.ols(formula, data).fit().rsquared_adj
            scores_with_candidates.append((score, candidate))
        scores_with_candidates.sort()
        best_new_score, best_candidate = scores_with_candidates.pop()
        if current_score < best_new_score:
            remaining.remove(best_candidate)
            selected.append(best_candidate)
            current_score = best_new_score
    formula = "{} ~ {}".format(response,
                               ' + '.join(selected))
    model = smf.ols(formula, data).fit()
    return model
```

```
model = forward_selected(df, "mpg")
print(model.model.formula)
```

```
## mpg ~ weight + year + origin + displacement + horsepower + cylinders
```