Serie 10

Aufgabe 10.1

In dieser Aufgabe verwenden wir den Datensatz `Auto.csv`.

```
df = pd.read_csv(".../Auto.csv", index_col=0)
```

a) Stellen Sie das Modell auf für eine einfache lineare Regression mit **`mpg`** als Zielvariable und **`horsepower`** als Prädiktor.

b) Verwenden Sie den **`.params`**- oder **`.summary()`**- Befehl um diese Regression durchzuführen.

   Verwenden Sie den **`summary()`**-Befehl um die Resultate auszudrucken. Kommentieren Sie diesen:

   i) Gibt es einen Zusammenhang zwischen der Zielgrösse und dem Prädiktor?

   ii) Wie interpretieren Sie die Koeffizienten für **`const`** und **`horsepower`**? Ist der Zusammenhang positiv oder negativ?

   iii) Bestimmen Sie die Vertrauensintervalle (mit **`[0.025 0.975]`**) und interpretieren Sie diese?

   iv) Interpretieren Sie den R^2-Wert.

c) Plotten Sie die Zielvariable und den Prädiktor mit der zugehörigen Regressionsgeraden (**`abline_plot`**). Wie interpretieren Sie diesen Plot im Vergleich zum **`summary()`**-Output.

Aufgabe 10.2

The `Boston.csv` data set records **`medv`** (medianhouse value) for 506 neighborhoods around Boston. We will seek to predict **`medv`** using 13 predictors such as **`rm`** (average number of rooms per house), **`age`** (average age of houses), and **`lstat`** (percent of households with low socioeconomic status).

```
df = pd.read_csv(".../Boston.csv", index_col=0)
```

a) Which column names are available?

b) We will start by using the `OLS()` function to fit a simple linear regression model, with `medv` as the response and `lstat` as the predictor.

   i) Define the simple regression model using the two variables above.

   ii) The basic syntax is `OLS(y, x).fit()` , where `y` is the response, `x` is the predictor, and data is the data set in which these two variables are kept.

```
fit = sm.OLS(...).fit()
fit.summary()
```

c) Find the coefficient of the regression model.

   Interpret these values and the corresponding p–values in the summary above.

d) In order to obtain a confidence interval for the coefficient estimates, we can use the `confint(...)` command.

   Give an interpretation of these values.

e) We will now plot `medv` and `lstat` along with the least squares regression line using the `plot(kind="scatter")` and `abline_plot()` function.

f) Interpret the R^2 value in the `summary`–output above.

Kurzlösungen vereinzelter Aufgaben

Musterlösungen zu Serie 10

Lösung 10.1

Einlesen der Datei

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

df = pd.read_csv("/home/euler/Dropbox/Statistics/Themen/Einfache_Lineare_Regress
index_col=0)

df.head()

##      mpg  cylinders  displacement  ...  year  origin                      nam
## 1  18.0          8         307.0  ...    70       1  chevrolet chevelle malib
## 2  15.0          8         350.0  ...    70       1          buick skylark 32
## 3  18.0          8         318.0  ...    70       1         plymouth satellit
## 4  16.0          8         304.0  ...    70       1              amc rebel ss
## 5  17.0          8         302.0  ...    70       1               ford torir
##
## [5 rows x 9 columns]
```

a) Lineare Regression:

$$\texttt{mpg} = \beta_0 + \beta_1 \cdot \texttt{horsepower}$$

b) Output:

```
Y = df["mpg"]
X = df["horsepower"]
X = sm.add_constant(X)

## //usr/lib/python3/dist-packages/numpy/core/fromnumeric.py:2495: FutureWarning: Method .ptp is deprec
##   return ptp(axis=axis, out=out, **kwargs)

fit = sm.OLS(Y,X).fit()

fit.summary()

## <class 'statsmodels.iolib.summary.Summary'>
## """
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                    mpg   R-squared:                       0.606
## Model:                            OLS   Adj. R-squared:                  0.605
## Method:                 Least Squares   F-statistic:                     599.7
## Date:                Tue, 05 May 2020   Prob (F-statistic):           7.03e-81
```

```
## Time:                        09:04:16   Log-Likelihood:                -1178.7
## No. Observations:                 392   AIC:                             2361.
## Df Residuals:                     390   BIC:                             2369.
## Df Model:                           1
## Covariance Type:            nonrobust
## ==============================================================================
##                  coef    std err          t      P>|t|      [0.025      0.975]
## ------------------------------------------------------------------------------
## const         39.9359      0.717     55.660      0.000      38.525      41.347
## horsepower    -0.1578      0.006    -24.489      0.000      -0.171      -0.145
## ==============================================================================
## Omnibus:                       16.432   Durbin-Watson:                   0.920
## Prob(Omnibus):                  0.000   Jarque-Bera (JB):               17.305
## Skew:                           0.492   Prob(JB):                     0.000175
## Kurtosis:                       3.299   Cond. No.                         322.
## ==============================================================================
##
## Warnings:
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
## """
```

i) Der p-Wert für **horsepower** ist fast 0 und somit wird die Nullhypothese ($\beta_1 = 0$) verworfen. Der Treibstoffverbrauch hängt von den PS ab.

ii) Der Wert 39.93 für den **const** gibt den Benzinverbrauch (miles pro gallon) bei 0 PS an. Dieser Wert hat hier natürlich keine praktische Bedeutung.

Interessanter ist der Wert -0.15 für **horsepower**. Dieser bedeutet, dass pro PS dass Auto 0.15 Meilen weniger weit kommt für eine Gallone (≈ 3.8 l) Benzin.

Der Zusammenhang ist also negativ: je mehr PS umso weniger weit kommt pro Gallone.

iii) Vertrauensintervall:

```
fit.conf_int()
```

```
##                     0          1
## const       38.525212  41.346510
## horsepower  -0.170517  -0.145172
```

Die wahren Werte für **const** und **horsepower** liegen zu 95 % in den entsprechenden Intervallen. Die Intervalle sind recht schmall, so dass die Aussagekraft dieser Intervalle recht gross ist.

iv) Der R^2-Wert ist 0.606. Dieser gibt an, dass die Variabilität zu 60 % durch das Modell ist.

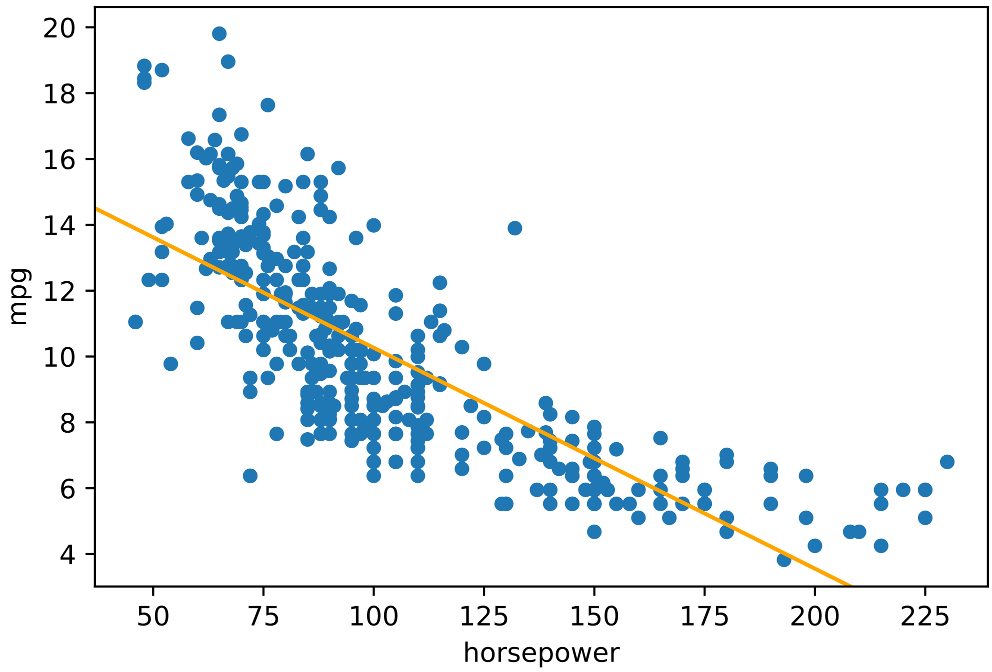
Das ist ok, aber nicht besonders gut, da noch andere Prädiktoren Einfluss auf den Benzinverbrauch haben.

```
fit.rsquared
```

```
## 0.6059482578894346
```

c) Plot:

```
ax = df.plot(kind="scatter",x="horsepower", y="mpg")
abline_plot(model_results=fit, ax = ax, color="orange")
plt.show()
```



Die sinkende Tendenz ist deutlich sichtbar, deshalb der tiefe p-Wert. Allerdings fällt die Punktwolke nicht linear (schwacher R^2-Wert).

Lösung 10.2

Load the data set

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

df = pd.read_csv("../../../Themen/Einfache_Lineare_Regression/Daten/Boston.csv", index_col=0)

df.head()
```

```
##        crim    zn  indus  chas    nox  ...  tax  ptratio   black  lstat  medv
## 1  0.00632  18.0   2.31     0  0.538  ...  296     15.3  396.90   4.98  24.0
## 2  0.02731   0.0   7.07     0  0.469  ...  242     17.8  396.90   9.14  21.6
## 3  0.02729   0.0   7.07     0  0.469  ...  242     17.8  392.83   4.03  34.7
## 4  0.03237   0.0   2.18     0  0.458  ...  222     18.7  394.63   2.94  33.4
## 5  0.06905   0.0   2.18     0  0.458  ...  222     18.7  396.90   5.33  36.2
##
## [5 rows x 14 columns]
```

a) Column names

```
df.columns

## Index(['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad',
##        'ptratio', 'black', 'lstat', 'medv'],
##       dtype='object')
```

b) i) The model is defined as follows:

$$\texttt{medv} = \beta_0 + \beta_1 \cdot \texttt{lstat}$$

ii) Output

```
Y = df["medv"]
X = df["lstat"]
X = sm.add_constant(X)

fit = sm.OLS(Y,X).fit()

fit.summary()

## <class 'statsmodels.iolib.summary.Summary'>
## """
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                   medv   R-squared:                       0.544
## Model:                            OLS   Adj. R-squared:                  0.543
## Method:                 Least Squares   F-statistic:                     601.6
## Date:                Tue, 05 May 2020   Prob (F-statistic):           5.08e-88
## Time:                        09:04:16   Log-Likelihood:                -1641.5
## No. Observations:                 506   AIC:                             3287.
## Df Residuals:                     504   BIC:                             3295.
## Df Model:                           1
## Covariance Type:            nonrobust
## ==============================================================================
##                  coef    std err          t      P>|t|      [0.025      0.975]
## ------------------------------------------------------------------------------
## const         34.5538      0.563     61.415      0.000      33.448      35.659
## lstat         -0.9500      0.039    -24.528      0.000      -1.026      -0.874
## ==============================================================================
## Omnibus:                      137.043   Durbin-Watson:                   0.892
## Prob(Omnibus):                  0.000   Jarque-Bera (JB):              291.373
## Skew:                           1.453   Prob(JB):                     5.36e-64
## Kurtosis:                       5.319   Cond. No.                         29.7
## ==============================================================================
##
## Warnings:
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
## """
```

c) Coefficient:

```
fit.params
```

```
## const     34.553841
## lstat     -0.950049
## dtype: float64
```

Substitute these values in the simple linear regression model above

$$\texttt{medv} = 34.554 - 0.95 \cdot \texttt{lstat}$$

The values 34.55 is the intercept, which is the value for `lstat = 0` (zero percent of lower status of the population). The median house value is \$34 554 in neighborhoods with 0 percent population of lower status.

The value -0.95 is the slope of the regression line. We can interpret this value as follows: for each additional percent in population of lower status, the median house value drops by \$950.

d) Confidence intervals

```
fit.conf_int()
```

```
##                0           1
## const  33.448457  35.659225
## lstat  -1.026148  -0.873951
```

The true value of the intercept is with 95 % probability in the interval

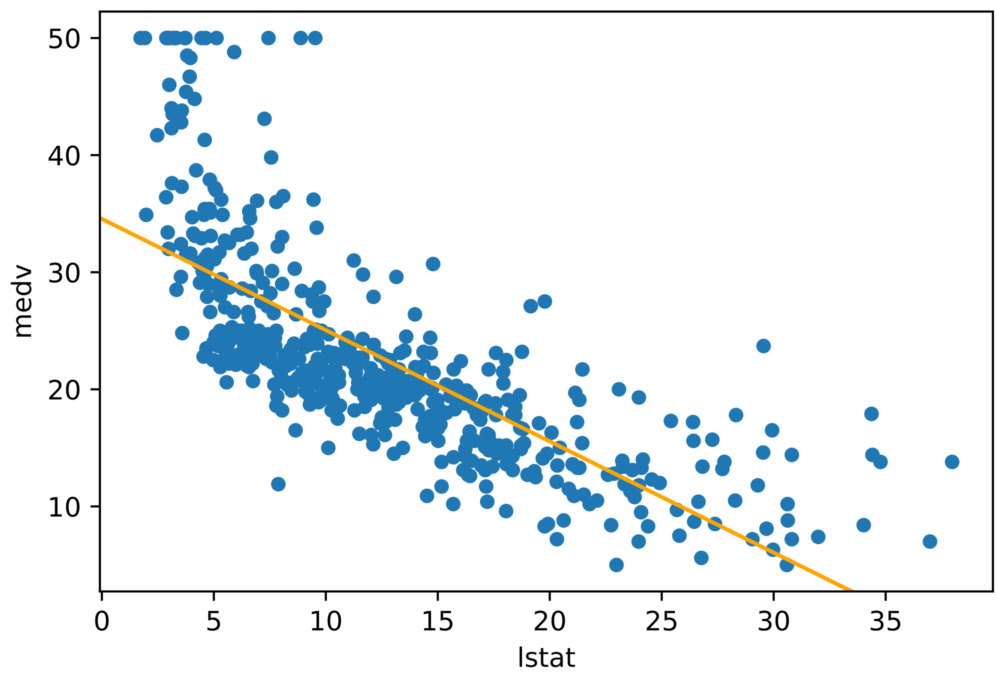
$$[33.45,\ 35.66]$$

The true value of the slope is with 95 % probability in the interval

$$[-1.02,\ -0.87]$$

e) Plot:

```
ax = df.plot(kind="scatter",x="lstat", y="medv")
abline_plot(model_results=fit, ax = ax, color="orange")
plt.show()
```



f) The R^2 value is

```
fit.rsquared

## 0.5441462975864795
```

The amount of variability which is explained by the model is 0.544. About half of the variation is explained by the model.