Homework_H5.4_Multiple_Linear_Regression_Jupyter_Notebook

March 1, 2021

1 multiple Linear Regression (mLR) with scikit-learn

1.0.1 Provided by Nora Baitinger, Antonia Wermerskirch, Paul Jaworski

Location: DHBW Stuttgart, Date: 2.11.2020

Extented by H. Völlinger; DHBW; 2.11.2020

The implementation of mLR is very similar to that of sLR: 1. Import all needed packages 2. Provide data to work with 3. Create and fit regression model with data from previous step 4. Check the fitted model for statisfaction 5. Apply model for predicitions

2 Step 1: Import all needed dependencies

numpy - uses numerical mathematics

IPython - uses display, Math and Latex to for printing the formula

sklearn - Use/call the LinearRegression module

sys - version information to pythonImport of libraries

[1]:
```
# For mLR we use the numpy and scikit-learn python library. Ladder provides the␣
↪LinearRegression module we need.
import numpy as np
from sklearn.linear_model import LinearRegression
from IPython.display import display, Math, Latex
import sys

# check version of numpy an sklearn

# version of numpy:
print("numpy {}".format(np.__version__))
# version of sklearn
import sklearn as sk
print("sklearn {}".format(sk.__version__))
# version of python
# print python version, for some imports this version number is viewed as␣
↪theirs.
```

```
print("python {}".format(sys.version))
```

```
numpy 1.19.2
sklearn 0.23.2
python 3.8.5 (default, Sep  3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
```

3 Step 2: Provide data and Check data

Provide the data given in the task

[2]:
```
x = [[1, 2], [3, 3], [2, 2], [4, 3]]
x1 = [1, 2]; x2 = [3, 3]; x3 = [2, 2]; x4 =[4, 3]
y = [3, 4, 4, 6]
x, y = np.array(x), np.array(y)
```

3.0.1 Print the data and check it

[3]:
```
# You can print x and y to check the correctness:
print('(x11,x21) looks like:     ', x1)
print('(x12,x22) looks like:     ', x2)
print('(x13,x23) looks like:     ', x3)
print('(x14,x24) looks like:     ', x4)
print('(y1,y2,y3,y4) looks like:', y)
```

```
(x11,x21) looks like:      [1, 2]
(x12,x22) looks like:      [3, 3]
(x13,x23) looks like:      [2, 2]
(x14,x24) looks like:      [4, 3]
(y1,y2,y3,y4) looks like: [3 4 4 6]
```

4 Step 3: Create and fit the model

Now we need to create and fit the regression model. To do that we need to create an instance of LinearRegression and call the .fit()-method on it.

[4]:
```
#The variable fitted_model represents in this case the fitted model according␣
 ↪to the given data.
fitted_model = LinearRegression().fit(x, y)
```

5 Step 4: Get needed results

[5]:
```
#  ² can be obtained by calling the score() method on the model
r_square = fitted_model.score(x, y)
print('coefficient of determination:', r_square)
# The values of the estimators of regression coefficients can be obtained by:
print('intercept:', fitted_model.intercept_)
print('coefficients:', fitted_model.coef_)
# intercept holds the bias b0, coef holds b1 and b2 in an array
```

```
coefficient of determination: 0.9473684210526316
intercept: 4.25
coefficients: [ 1.5 -1.5]
```

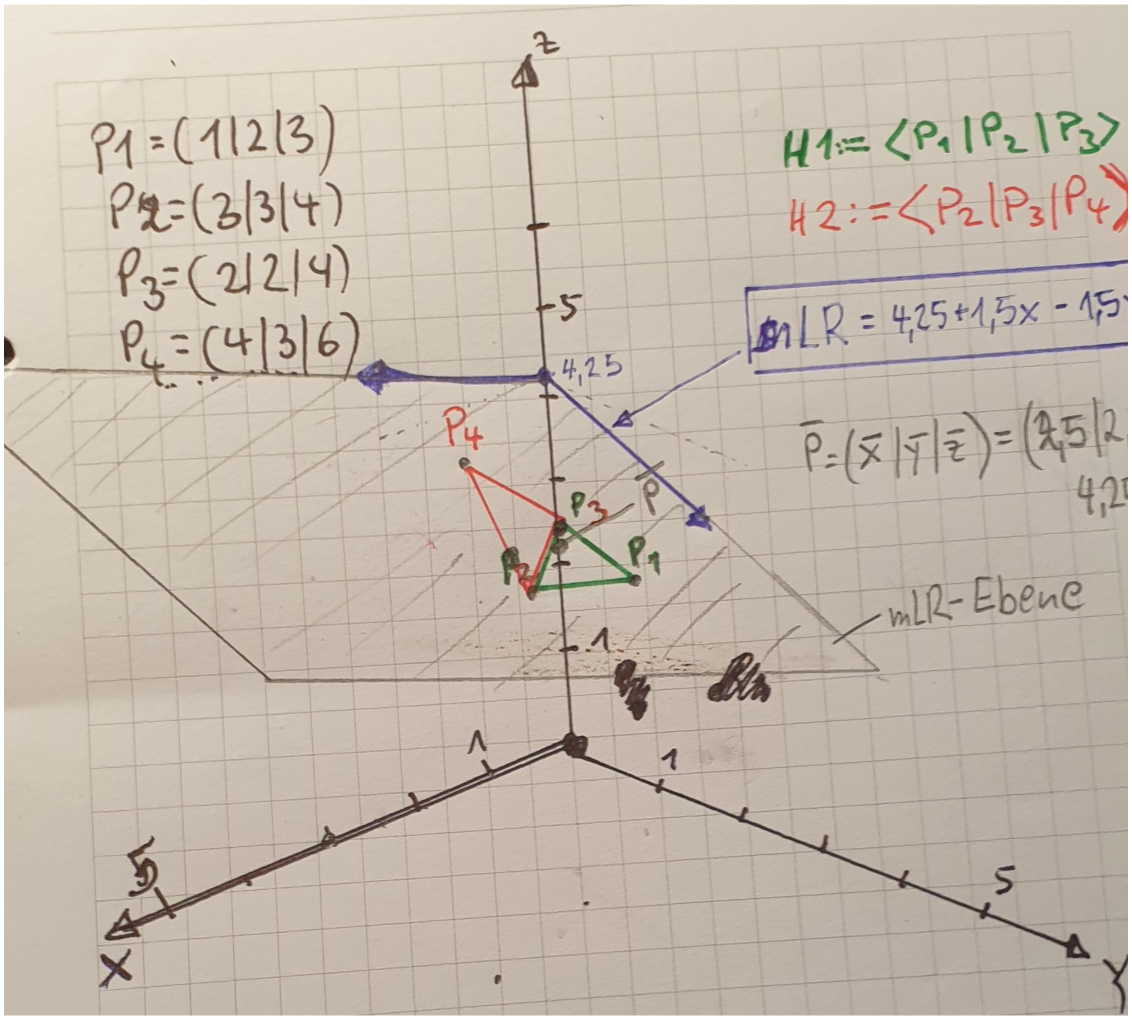
5.0.1 Visualization of the Result

Here I imported an hand-done sketch.

This will be replaced later by a python graphics

[6]:
```
from IPython.display import Image
Image('Images/Homework_H5_4-mLR_plane-Sketch.jpg')
```

[6]:



6 Step 5: Predict response using the model

The process of predicting values in mLR is analogous to that in sLR. The prediction is performed by predict() :

```
[7]: prediction = fitted_model.predict(x)
     print('predicted response:', prediction, sep='\n')
```

```
predicted response:
[2.75 4.25 4.25 5.75]
```

Another way to predict y-values is to multiply each column of the input with the appropriate weight, summing the results up and adding the intercept to the sum.

```
[8]: prediction = fitted_model.intercept_ + np.sum(fitted_model.coef_ * x, axis=1)
     print('predicted response:', prediction, sep='\n')
```

```
predicted response:
[2.75 4.25 4.25 5.75]
```

```
[9]: # print current date and time
     import time
     print("date+time",time.strftime("%d.%m.%Y %H:%M:%S"))
     print ("***** End of Homework H5.4c *******")
```

```
date+time 01.03.2021 10:42:11
***** End of Homework H5.4c *******
```