Wirtschaftsinformatik 2

Beispielaufgabe Schleifen – erste Statistiken

Aufgabe 1

Programmieren Sie eine Funktion `selektiere(df)`, der Sie einen Dataframe übergeben und die folgende Aufgabe erfüllt:

> Selektieren Sie die Spalten des Dataframe von der 2. bis zur 4. Spalte.

Wenn Sie die Funktion mit einem Dataframe mit folgenden Daten aufrufen:

Produkt	Anzahl	Einkaufspreis	Softwarekategorie	Versandart	Gewinn
PDF Reader	1	26.0	Utilities	Download	2.08
Kalkulation	5	82.95	Office	CD-Versand	23.04
Windows 4711	2	112.0	Betriebssysteme	Download	6.74
Windows 0815	10	87.0	Betriebssysteme	Download	26.1
Writer	1	58.45	Office	CD-Versand	5.22

erhalten Sie folgendes Ergebnis:

Einkaufspreis	Softwarekategorie	Versandart
26.0	Utilities	Download
82.95	Office	CD-Versand
112.0	Betriebssysteme	Download
87.0	Betriebssysteme	Download
58.45	Office	CD-Versand

Lösung (im Jupyter-Notebook):

```
import pandas as pd
from pandas import Series
from pandas import DataFrame

url = "https://data.hsbo.de/Gewinnbeispiel.csv"
gewinndatenDF = pd.read_csv(url)

def selektiere(df):
    reduzierterDF = df.loc[:, 'Einkaufspreis':'Versandart']
    return reduzierterDF
```

oder kürzer

```
def selektiere(df):
    return df.loc[:, 'Einkaufspreis':'Versandart']
```

Der Aufruf erfolgt durch:

```
selektiere(gewinndatenDF)
```

In Ibix brauchen Sie nur die Funktion `selektiere(df)` hochladen. Die Zeilen zum Import von pandas, DataFrame und Series sollten Ihnen klar sein.

Im Dialogfenster wählen Sie die csv-Datei aus und die Datei wird hochgeladen.

Durch die Zeilen

```
url = "https://data.hsbo.de/Gewinnbeispiel.csv"
gewinndatenDF = pd.read_csv(url)
```

wird ein neuer Dataframe erzeugt und direkt mit den Daten der csv-Datei gefüllt. Wenn Sie im Jupyter-Notebook

```
gewinndatenDF
```

eingeben, sehen Sie die importierten Daten in der gewohnten DataFrame Darstellung.

Die Zeile

```
reduzierterDF = df.iloc[:, 1:4]
```

erzeugt nun den in der Aufgabenstellung gewünschten DataFrame. Der erste :"vor dem Komma in diesem Kommando sagt aus, dass alle Zeilen zum Ergebnis gehören. Die Auswahl erfolgt also nur anhand der Spalten. Der zweite :"bedeutet, dass alle Spalten zwischen 1 und 4 für das Ergebnis ausgewählt werden. Hier müssen Sie beachten, dass die Spaltenzählung bei 0 beginnt. Also müssen wir bei der Spalte 1 statt 2 beginnen. Die Selektion geht nun aber trotzdem bis zur Spalte 4, da die rechte Spalte des Auswahlbereichs nicht in das Ergebnis aufgenommen wird. Weitere Selektionsmöglichkeiten (und auch diese) können Sie dem Foliensatz entnehmen.

Natürlich können wir uns den Weg über die Variable reduzierterDF sparen und die Selektion direkt im return-Befehl durchführen. Dies erklärt die kürzere Lösung.

Von nun an werden wir uns den Import von pandas, Series und Dataframe sowie die Erstellung von gewinndatenDF aus der csv-Datei in den Lösungen schenken. Übernehmen Sie dies aus Aufgabentyp 1.

Aufgabe 2 Aufgabentyp 2

Programmieren Sie eine Funktion `filter(df)`, der Sie einen Dataframe übergeben. Die Funktion wählt aus dem Dataframe alle Zeilen mit *Anzahl* kleiner als 5 aus und gibt einen Dataframe mit den gefilterten Daten zurück.

Wenn Sie die Funktion mit einem Dataframe mit folgenden Daten aufrufen:

Produkt	Anzahl	Einkaufspreis	Softwarekategorie	Versandart	Gewinn
PDF Reader	1	26.0	Utilities	Download	2.08
Kalkulation	5	82.95	Office	CD-Versand	23.04
Windows 4711	2	112.0	Betriebssysteme	Download	6.74
Windows 0815	10	87.0	Betriebssysteme	Download	26.1
Writer	1	58.45	Office	CD-Versand	5.22

erhalten Sie folgendes Ergebnis:

Produkt	Anzahl	Einkaufspreis	Softwarekategorie	Versandart	Gewinn
PDF Reader	1	26.0	Utilities	Download	2.08
Windows 4711	2	112.0	Betriebssysteme	Download	6.74
Writer	1	58.45	Office	CD-Versand	5.22

Lösung (im Jupyter-Notebook):

```
def filter(df):
    filterSeries = df['Anzahl']<5
    gefilterterDf = df[filterSeries]
    return gefilterterDf
```

oder kürzer

```
def filter(df):
    return df[df['Anzahl']<5]
```

Aufruf erfolgt durch

```
filter(gewinndatenDF)
```

Die Zeile

```
filterSeries = df['Anzahl']<5
```

erzeugt eine Series mit bool'schen Werten. `filterSeries` sieht folgendermaßen aus:

0	True
1	False
2	True
3	False
4	True

Die Indices von `filterSeries` sind die Zeilenindices des des Dataframe. Die Werte sind `True`, wenn der Wert der Spalte *Anzahl* in den jeweiligen Zeilen < 5 sind, anderenfalls `False`.

In

```
gefilterterDf = df[filterSeries]
```

wird nun aus dem übergegebenen Dataframe und `filterSeries` der neue Dataframe erzeugt. Zeilen des übergebenen Dataframes, in denen `filterSeries` den Wert false hat, werden nicht in den resultierenden Dataframe übernommen. Dies sind aber genau die Zeilen, in denen die Spalte *Anzahl* des Dataframes kleiner als 5 ist. So entsteht das gewünschte Resultat.

Natürlich können wir uns den Weg über die Variablen `filterSeries` und `gefilterterDf` sparen und die Erzeugung der Series und des Ergebnis-DataFrames direkt im `return`-Befehl durchführen. Dies erklärt die kürzere Lösung.

Aufgabe 3 Aufgabentyp 3

Programmieren Sie eine Funktion `berechne_statistik(df)`, mit der Sie Statistiken berechnen. Die Funktion erhält als Parameter einen Dataframe.

Für das unten dargestellte Dataframe

Produkt	Anzahl	Einkaufspreis	Softwarekategorie	Versandart	Gewinn
PDF Reader	1	26.0	Utilities	Download	2.08
Kalkulation	5	82.95	Office	CD-Versand	23.04
Windows 4711	2	112.0	Betriebssysteme	Download	6.74
Windows 0815	10	87.0	Betriebssysteme	Download	26.1
Writer	1	58.45	Office	CD-Versand	5.22

berechnen Sie folgende Statistik:

Anzahl Datensätze mit *Versandart* "Download"

Für das oben dargestellten Beispiel ergibt das folgendes Ergebnis:

3

Lösung (im Jupyter-Notebook):

```
def berechne_statistik(df):
    df = df[ df['Versandart'] == "Download" ]
    anzahlenSeries = df.count()
    return anzahlenSeries['Versandart']
```

Der Aufruf erfolgt mit `berechne_statistik(gewinndatenDF)`. Die Zeile

```
df = df[ df['Versandart'] == "Download" ]
```

erzeugt in der in Aufgabentyp 2 beschriebenen Art einen neuen Dataframe aus den übergebenen Daten. Dabei werden die Zeilen des übergebenen Dataframes, in denen *Versandart* nicht "Downloadïst herausgefiltert. Der alte Dataframe wird dabei durch den neuen Überschrieben. Wir verwenden die verkürzte Vorgehensweise aus Aufgabentyp 2.

```
anzahlenSeries = df.count()
```

erzeugt nun eine neue Series. `anzahlenSeries` sieht folgendermaßen aus:

Produkt	3
Anzahl	3
Einkaufspreis	3
Softwarekategorie	3
Versandart	3
Gewinn	3

Der Index ist der Spaltenindex des Dataframes, die Werte die Anzahl Datensätze der einzelnen Spalten des Dataframes. Die sind alle gleich, nämlich 3. Beachten Sie, dass das nicht so sein muss. Enthält eine Spalte NaN-Werte, so werden diese nicht in die Zählung aufgenommen und die Anzahl Datensätze einer solchen Spalte weicht ab.

Wir geben nun `anzahlenSeries['Versandart']` zurück und erhalten das gewünschte Ergebnis.