“Feinstaub in Graz”

Hausübung 3

7. November 2018

Originaldaten: https://luftdaten.info/

1 Tasks (25pt)

Thema dieses Assignments ist die Visualisierung der Sensordaten auf einer geographischen Karte. Einen Referenzoutput (.html Download) finden Sie im Wiki[1]!

1.1 Functions (5 pt)

Ziel dieser Übung ist der Umgang Visualisierungstools und eine saubere Strukturierung in Funktionen. Dies bedeutet, dass der Großteil Ihres Codes über Funktionen implementiert sein soll. Einzige Ausnahme hier ist das Parsen der Kommandozeile (Extrahieren der Parameter). Checks der Paramater (z.B., ob die Inputdatei vorhanden ist) müssen in eine eigene Funktion. Ihr Programm besteht dann also aus einer Reihe von Funktionen die dann, je nach Parameter aufgerufen werden sollen. Als Guideline gilt, dass jede unabhängige Funktionalität in eine eigene Funktion muss. Wennimmer Sie zweimal in Ihrem Programm identen Code finden (oder auch öfter), dann ist dies ein gutes Indiz, dass dies ebenfalls in eine eigene Funktion sollte. Ihr Programm sollte aus mindestens 10 sinnvoll gestalteten Funktionen bestehen. Pro Funktion werden hier 0.5p vergeben (bis max 5).

1.2 Command Line Parameters (3p)

Ihr Programm soll mit einer Reihe von Kommandozeilenparametern umgehen können. Verwenden Sie das `argparse` package[2] um diese Funktionialität zu implementieren. Implementieren Sie die folgenden Parameter:

`-i --input:` Pfad zur Inputdatei (dem Datensatz)

- Datentyp: String
- Optional: Nein

`-o --output:` Pfad zur den Outputdateien

- Datentyp: String
- Optional: Nein

`-s --sensors:` Namen der gewünschten Sensoren

- Datentyp: Liste aus Integern

[1] https://palme.iicm.tugraz.at/wiki/Info1BM#.C3.9Cbungs_Assignments
[2] https://docs.python.org/3.7/library/argparse.html

- Optional: Ja
- Defaultwert: `[ ]` (leere Liste)

`-w --width:` Breite der Karte

- Datentyp: Integer
- Optional: Ja
- Defaultwert: 800

`-h --height:` Höhe der Karte

- Datentyp: Integer
- Optional: Ja
- Defaultwert: 800

`-z --zoom:` Zoom der Karte

- Datentyp: Integer
- Optional: Ja
- Defaultwert: 13

Für alle Parameter, die als Strings übergeben werden, arbeiten sie case-insensitiv. Erlauben Sie also jede Variante der Groß-Kleinschreibung!

1.3 Basic I/O Checks (1p)

Überprüfen Sie die dateibezogenen Parameter auf Ihre Gültigkeit. Implementieren Sie die folgenden Checks:

(i) Verifizieren, dass die Inputdatei existiert und auch tatsächlich eine Datei ist.
(ii) Verfizieren Sie, dass der gewünschte Output Pfad existiert.

Sollte einer der Tests fehlschlagen geben Sie eine Fehlermeldung aus und beenden Sie das Programm mit `exit()`. Alternativ können Sie auch eine Exception[3] raisen.

1.4 Datensatz Lesen

1.4.1 Lesen(1p)

Der Datensatz besteht aus dem Output von der letzten Hausübung. Sie können sich diesen entweder selbst erstellen oder eine Referenzversion aus dem Wiki herunterladen[4].

Lesen Sie den Datensatz ein. Sie können hier eventuell Ihren Code der letzten Hausübung wiederverwenden. Allerdings müssen Sie darauf achten, dass sich die Spalten (Header) geändert haben!

Lesen Sie Zeile für Zeile ein und speichern Sie die Werte. Der Kommandozeilenparameter `--sensors` dient hier als Filter. Überspringen Sie Zeilen die Sensoren (IDs) beschreiben die nicht explizit gewünscht sind. Sollte der Parameter nicht angegenen sein (also den Defaultwert `[ ]` haben, dann speichern Sie alle Sensoren.

1.4.2 Prüfen (1p)

Prüfen Sie auch hier wieder Ihren Datensatz nachdem Sie ihn eingelesen haben. Überprüfen Sie dabei, ob alle gewünschten Sensoren gefunden wurden. Sollte ein Sensor keine Einträge haben, dann geben Sie eine Notiz/Warnung für den Benutzer aus. Ihr Programm soll dennoch weiterarbeiten. Ausnahme ist hier, wenn kein einziger Gewünschter Sensor gefunden wurde. Hier beenden Sie das Programm via `exit()` oder Exception.

Als zweiten Vorbereitungsschritt müssen Sie verifizieren, dass die Reihenfolge der Messwerte von jedem Sensor chronologisch korrekt ist. Die ist ident mit der Anforderung aus der letzte Hausübung. Verwenden Sie diesmal allerdings die `numpy.argsort()`[5] Funktion!

[3] https://docs.python.org/3/tutorial/errors.html
[4] https://palme.iicm.tugraz.at/wiki/Info1BM#.C3.9Cbungs_Assignments
[5] https://docs.scipy.org/doc/numpy-1.15.4/reference/generated/numpy.argsort.html?highlight=argsort#numpy.argsort

1.5 Plotten auf der Karte (5p)

Zum Erstellen der Karte verwenden wir das `folium` Package[6]. Machen Sie sich mit den Package vertraut und installieren Sie es, falls notwendig (`pip install folium`). Beim Erstellen der Karte sind einige Parameter notwendig. Die Breite und Höhe der Karte (in Pixel) entnehmen Sie aus den Kommandozeilenparametern `--width` und `--height` und `zoom_start` entspricht dem Parameter `--zoom`. Als tileset (`tiles`) verwenden Sie `"OpenStreetMap"`. Dies können Sie direkt in den Code schreiben. Schlussendlich benötigt Ihre Karte noch einen Mittelpunkt (`location`). Berechnen Sie hierfür jeweils den Median der Längen und Breidengrade aus den (gewünschten) Sensoren.

Für jeden Sensor erstellen Sie einen Marker (`folium.Marker`) und platzieren Sie ihn auf der Karte. Speichern Sie am Ende Ihres Programms die Karte im `--output` mit dem Namen `map.html`.

1.6 Bokeh Popups auf Markern (9p)

Für jeden Marker auf der Karte (also jedem gewünschten Sensor) erstellen Sie ein Popup, welches beim Klick auf den Marker erscheint. Der Plot soll so aussehen, wie Ihre Outputs der letzten Hausübung. Farben sind jedoch frei wählbar. Bei dem bereitgestellten Codestück wurde `fill_between()` von Matplotlib verwendet. Bokeh besitzt mit Band[7] ähnliche Funktionalität. Plotten Sie zusätzlich zu diesem Band die Werte auch via `line()`. Ansonsten werden die Achsen nicht richtig angepasst.

Verwenden Sie auch das Bokeh HoverTool[8] um einen Mouseover Effekt bei den Popups zu erreichen. Dies soll (wie in der Referenz) jeweils das Datum, den P1 und den P2 Wert an der aktuellen Mausposition anzeigen. Achten Sie darauf, dass das Datum wiklich als Datum angezeigt wird (und nicht als große Zahl). Setzen Sie beim Erstellen des HoverTools den Parameter `mode` auf `"vline"` und den Parameter `point_policy` auf `"follow_mouse"`. Dies führt zu einem Verhalten wie bei der Referenz.

Damit das HoverTool richtig funktioniert binden Sie es nur an die Linie von P1 (die Sie zusätzlich zum Band geplottet haben). Dies können Sie mit dem parameter `renderers()` beim HoverTool erreichen.

Die Höhe und Breite des Bokeh Plots berechnen Sie folgendermaßen:

- Höhe = `--height` Parameter / 4
- Breite = `--width` Parameter / 2.5

Wenn Sie das Folium Popup erstellen, müssen Sie dort erneut Werte für Höhe und Breite angeben. Setzen Sie diese auf den Wert des Bokeh Plots +25. Dadurch ist genug Abstand geboten und es entstehen keine Scrollbalken.

Sie können folgenden Code verwenden um mit Ihrem Bokeh Plot ein Folium Popup zu erstellen. Dies muss dann nur noch beim Marker eingebunden werden.

```
from bokeh.resources import CDN # You will need this import

# bokeh_plot is the variable where you have stored the plot you want to display in the popup
# replace <width> and <height> with your values. Make sure they are integers!

popup = folium.Popup(
    folium.IFrame(
        folium.Html(
            file_html(
                bokeh_plot,
                CDN
            ),
            script=True),
        height=<height>,
        width=<width>)
)
```

Hinweis: Sie können sich zum Testen auch jeden einzelnen dieser Popoup Plots speichern. Entfernen Sie dies aber zur Abgabe.

[6] https://github.com/python-visualization/folium
[7] https://bokeh.pydata.org/en/latest/docs/user_guide/examples/plotting_band.html
[8] https://bokeh.pydata.org/en/latest/docs/user_guide/examples/tools_hover_tooltip_formatting.html

1.7 Bonus: Schneeballmarker (3p)

Erstellen Sie weitere Marker auf der Karte. Diese sollen jene Orte Markieren, an denen Sie über die Weihnachtsfeiertage eine Schneeballschlacht hatten. Diese müssen nicht im Raum Graz sein! Sofern möglich mit einem Beweisfoto im Popup. Sollte dies jedoch an Schneemangel scheitern sind auch Keks-Marker die jeweils anzeigen, an welchen Orten Sie Weihnachtskeke gegessen hatten. Sie können die Websize https://www.latlong.net/ verwenden, um Länge und Breitengrad von Orten zu Berechnen

2 Beschränkungen

- Fügen Sie keine nutzlosen Komponenten Ihrer Abgabe hinzu
- Sofern möglich, verwenden Sie eingebaute Funktionen
- Importieren Sie nur erlaube Packages
- Verlangen Sie keine extra Kommandozeilenparameter
  - Sie dürfen extra Parameter verwenden. Ihr Programm muss aber auch ohne deren Verwendung funktionieren!

2.1 Erlaubte Packages

Allgemein: `argparse`, `copy`, `collections`, `os`, `sys`
Math: `numpy`, `pandas`
Visualisierung: `matplotlib`, `seaborn`, `bokeh`, `folium`

3 Datei Header

All Ihre Quelldateien in Ihrer Abgabe müssen gleich zu Beginn einen Kommentar mit folgenden Informationen enthalten:

- Author: – Ihr Name
- MatNr: – Ihre Matrikelnummer
- Description: – Generelle Beschreibung der Datei
- Comments: – Kommentare, Erklärungen, usw

Bitte einfach den folgenden Code in Ihre Dateien kopieren und den Inhalt anpassen. Je nach PDF-Reader müssen Sie eventuell die Leerzeichen/Einrückungen per Hand anpassen. Bei jupyter Notebooks fügen Sie den Kommentarheader bitte in die erste Zelle ein.
Beispielheader:

```
########################################################################
# Author:      [FIRST NAME] [LAST NAME]
# MatNr:       [MATR NR]
# Description: [SHORTDESCRIPTION]
# Comments:    [ANY RELEVANT COMMENTS.
#              CAN BE MULTILINE]
########################################################################
```

4 Coding Standard

Für diese Lehrveranstaltung orientieren Sie sich am offiziellen PEP 8 Standard[9]. Dieser Beschreibt grundsätzliche Formalitäten im Bezug auf Ihren Code. Folgendes ist besonders zu Beachten:

[9] `https://www.python.org/dev/peps/pep-0008/`

Sprache. Code schreibt man in Englisch. Im internationalen Zeitalter ist es notwendig, dass auch jemand am anderen Ende der Welt verstehen kann, was Sie programmiert haben. Ihr gesamter Quellcode muss daher auf Englisch geschrieben sein. Dies betrifft sowohl die Kommentare also auch Variablennamen und Ähnliches.

Leerzeichen statt Tabulatoren. Python basiert auf Einrückungen, anstatt auf geschwungenen Klammern . Theoretisch gibt es die Möglichkeit, Leerzeichen (spaces) oder Tabulatoren (tabs) zu verwenden. PEP8 schreibt aber klar 4 Leerzeichen als Einrückungen vor. Die meisten Python Programmierumgebungen werden automatisch 4 Leerzeichen einfügen, wenn Sie auf die Tabulator-Taste drücken.

Sprechende Namen. Verwenden Sie kurze, aber sprechende Namen für Ihre Variablen, Funktionen, (und Ähnliches). Es muss eindeutig aus dem Namen hervorgehen, was die Aufgabe des Elements ist. Für simple Iterationen kann ein einfaches i ausreichend sein, aber dies kann schnell zu Chaos führen. Sollten Sie Variablen haben, die keine Aufgabe haben und nicht verwendet werden, schreibt PEP 8 vor, einen einfachen Unterstrich (_) zu verwenden. Sollten Sie sich unsicher sein, dann beschreiben Sie ihre Variablen (und Namen) in einem Codekommentar.

120 **Zeichen Zeilenlänge.** PEP8 sieht Zeilenlängen von maximal 79 Zeichen vor. Ein anderes, etwas großzügigeres Limit, welches sich in der Szene etabliert hat, sieht eine Länge von 120 Zeichen vor. In dieser LV verwenden wir daher das erweiterte Limit. Bitte achten Sie darauf, dass keine Zeile in Ihrem Code diese Länge von 120 Zeichen überschreitet (gilt auch für Kommentare). *Hinweis:* Zeilenlänge inkludiert die Einrückungen mittels Leerzeichen!

5 Automatisierte Tests

Ihr Programm wird automatisiert getestet. Achten Sie daher, dass Sie die angabe genau einhalten. Dies gilt im Besonderen für vorgeschriebene Variablen- und Funktionsnamen!
Vergewissern Sie sich, dass Ihre Abgabe allen Beschränkungen, die in dieser Angabe erwähnt sind, konform ist. Zusätzlich wird jede Abgabe auch von einem Mitglied des Tutorenteams begutachtet. Ihre Tutoren führen auch die finale Bewertung (Puntke) durch.

6 Abgabe

6.1 Deadline

15. Jänner 2019 um 23:59:59.

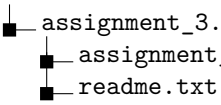
Eine Spätabgabe ist nicht vorgesehen. Ausnahme bilden hier Notfälle. Sollte das Abgabesystem nicht online sein, verlängert sich die Deadline automatisch um 24 Stunden. Sollten Sie, aus diversen Gründen, nicht in der Lage sein, Ihre Abgabe hochzuladen, kontaktieren Sie Ihren Tutor *VOR* der Deadline. (*Hinweis:* Urlaub oder Ähnliches wird nicht als Grund akzeptiert!)

6.2 Hochladen der Abgaben

Assignments werden stets als Archive abgegeben. Erlaubt sind hier die Formate *.zip*, und *.tar.gz*.
Zusätzlich zu ihren Quelldateien, soll Ihre Abgabe auf eine Datei namens *readme.txt* beinhalten. Das Vorhandensein der Datei ist Pflicht, ihr Inhalt aber optional. Sie soll folgenden Inhalt haben: (i) Die Zeit, die Sie benötigt haben, um die Aufgabenstellung zu absolvieren. (ii) Feedback, wo Sie Probleme hatten. Ihr Feedback ermöglicht es, verbreitete Probleme zu erkennen und die Vorlesung und Tutoriumseinheiten entsprechend anzupassen.
Abgaben erfolgen auf der Palme Website. Bitte prüfen Sie vor der Abgabe diese Kriteren:

- Datei- und Ordnerstruktue (siehe unten)
- Kommentarheader in jeder Quelldatei
- Coding Standard

6.3 Struktur der Abgabe



Datensatz und Outputdateien dürfen NICHT mit abgeben werden.