

Assignment codeerskills

T. Busker

3 december 2024

Deze reeks opdrachten begeleidt studenten stap voor stap bij het ontwikkelen van een volledig Python-project voor de analyse van toetsgegevens. Naast het technisch verwerken, valideren en visualiseren van data, ligt de nadruk op het leren opzetten van een professioneel en reproduceerbaar project. Studenten maken kennis met essentiële projectstructuren zoals een duidelijke `README.md`, een instelbaar installatieproces, en het gebruik van een `environment.yml` voor conda en/of `requirements.txt` om een consistente werkomgeving te garanderen. Door deze combinatie van programmeervaardigheden, data-analyse en projectorganisatie leren studenten hoe zij een volledig data-analyseproject van begin tot eind kunnen opzetten en onderhouden. Studenten leren om:

1. Een Python-project correct op te zetten met een duidelijke mapstructuur en documentatie.
2. Een reproduceerbare werkomgeving te maken via een `environment.yml`.
3. Command-line tools te bouwen met `argparse`.
4. Data voor te bereiden en te valideren met pandas.
5. Een grote taak te decomponeren in overzichtelijke functies en modules.
6. Statistische analyses uit te voeren op student- en vraaggegevens.
7. Visualisaties te genereren ter evaluatie van vraagkwaliteit.
8. Cronbach's alpha te berekenen en te interpreteren.
9. Alles te combineren tot één volledig analyseproces.
10. De code te documenteren en testen zodat het project onderhoudbaar blijft.

Opdracht 0: Projectstructuur & installatie

- Maak een basisprojectstructuur (mappen voor `src`, `data`, `output`, `tests`, enz.) in een nieuwe project map.
- Schrijf een eerste versie van `README.md` met:
 - Installatie-instructies
 - Projectbeschrijving
 - Gebruik van het script
- Maak een `environment.yml` om een conda-omgeving te definiëren.

- Test of de omgeving succesvol kan worden geïnstalleerd.
-

Opdracht 1: Command-Line argumenten

Command-line argumenten kunnen parseren in Python met de `argparse`-module.

- Maak een functie `parse_arguments` die verschillende argumenten accepteert zoals `input_file`, `output_file`, en andere parameters voor de analyse.
 - Zorg ervoor dat de standaardwaarden zijn ingesteld zoals, `DEFAULT_OUTPUT_FILE`, en andere gedefinieerde constanten.
-

Opdracht 2: Decorators in python

Begrijpen hoe decorators werken en hoe ze kunnen worden gebruikt om functionaliteit aan functies toe te voegen.

- Implementeer de decorator `inject_args` die command-line argumenten injecteert in een functie.
 - Zorg ervoor dat je de decorator kunt toepassen op functies die data-analyse uitvoeren.
-

Opdracht 3: Gegevensvoorbereiding

Data in DataFrame kunnen voorbereiden voor analyse.

- Maak een functie `prepare_data` die een Excel-bestand laadt en ervoor zorgt dat specifieke kolommen aanwezig zijn.
 - Voeg een kolom `participated` toe die aangeeft of een student volledig heeft deelgenomen.
 - Vul NaN-waarden voor bepaalde kolommen conditioneel in.
-

Opdracht 4: Decompositie

Leer het principe van decompositie door een complexe taak in kleinere, beter beheersbare taken te splitsen.

- Analyseer het gehele proces van examengegevensanalyse en identificeer de belangrijkste componenten.
 - Breek de functies op in kleinere, mogelijke herbruikbare functies. Bijvoorbeeld: splits de validatie van gegevens, statistische analyses, en visualisaties in aparte modules of functies.
-

Opdracht 5: Gegevensvalidatie

Implementeren validatieregels voor scores en controleren van de datakwaliteit.

- Maak de functie `validate_grades` die controleert of de scores in de DataFrame de maximale punten niet overschrijden.
 - Print fouten in de punten toekenning en berekende cijfers.
-

Opdracht 6: Statistische analyse

Uitvoeren van analyses op studentgegevens en vraagstatistieken.

- Implementeer de functie `analyze_student` die de deelname en cijfers van studenten analyseert en rapporten genereert.

- Maak de functie `analyze_questions` die statistieken berekent voor elke vraag, inclusief p-waarden en correlaties.
-

Opdracht 7: Visualisatie

Visualiseren van data om inzicht in data te krijgen.

- Implementeer de functie `create_visual` die een scatterplot genereert van p-waarden versus RIR-waarden.
 - Gebruik kleurcodering op basis van drempels voor RIR-waarden.
-

Opdracht 8: Cronbach's alpha

Verbeter concept van Cronbach's Alpha en hoe je het kunt berekenen.

- Maak de functie `calculate_cronbach_alpha` die de betrouwbaarheid van de toets meet.
 - Interpreteer de waarde met de functie `classify_cronbach_alpha`.
-

Opdracht 9: Hoofdfunctie

Integreer alle functies in een samenhangende analyse.

- Implementeer de functie `analyze_exam` die alle stappen van gegevensvoorbereiding tot visualisatie omvat.
 - Zorg ervoor dat je de uitvoer naar een bestand kunt omleiden indien nodig.
-

Opdracht 10: Testen en documentatie

Zorg ervoor dat de code goed gedocumenteerd is en test de functionaliteit.

- Voeg docstrings toe aan alle functies om de functionaliteit en parameters te beschrijven, en voeg typehinting toe aan elke functie.
- Test de volledige workflow van het script met voorbeeldgegevens.