Opdracht
Proof-of-concept geautomatiseerde data workflow.
Linux for Data Scientists, 2023-2024

Bert Van Vreckem en Thomas Parmentier

E-mail: bert.vanvreckem@hogent.be en thomas.parmentier@hogent.be

Samenvatting

Eén van de taken van een *data engineer* is om de *data scientist* bij te staan in het automatiseren van de workflow van het verzamelen van data, deze in de juiste vorm gieten zodat ze kan geanalyseerd worden tot het genereren van rapporten. De bedoeling van de opdracht die in dit document beschreven wordt, is om een proof-of-concept op te zetten van zo'n workflow waarbij je gebruik maakt van de tools die op een Linux-systeem beschikbaar zijn. Het resultaat van deze opdracht telt mee voor 30% van het examencijfer.

Inhoudsopgave

1. Opdracht

De opdracht bestaat uit verschillende stappen:

1. Verzamel ruwe tijdseriedata over een bepaalde periode
2. Transformeer de ruwe data in een geschikt formaat
3. Simuleer het analyseren van de data
4. Genereer een rapport op basis van de resultaten

Binnen elke fase gebruik je waar mogelijk Linux-tools om het proces te automatiseren. Voor de analyse kan je Python gebruiken (zie de lessen Data Science & AI), maar in de andere fasen vermijd je het gebruik ervan.

1.1. Verzamelen ruwe tijdseriedata

Je kiest zelf:

- Ofwel een combinatie van **twee of meer** eenvoudigere dataset(s) of service(s). Bedoeling is dan om de input uit deze verschillende services in de volgende fase te combineren tot één dataset.
- Ofwel een enkele dataset op voorwaarde dat de complexiteit van het uitlezen, bewaren en filteren van de data op zich voldoende hoog is.

Lees op regelmatige tijdstippen data uit of stuur een request naar een service(s) en sla het resultaat op.

Enkele ideeën als inspiratie:

- Een publiek toegankelijke REST-service, bv.
  - Open Dataportaal stad Gent
  - Lijst van publieke REST APIs (Postman)
  - Public API's (Github-repo)
  - Free APIs for developers (Rapid API)
  - Big list of free and open APIs (Ana Kravitz)

- Een IoT-device waar je informatie kan van uitlezen, bv. een temperatuursensor, home automation-systeem, ...
  - vb. CO2-meter in lokaal B.4.037 (kan je bv. combineren met observaties over het weer uit een publieke API)
- Een dynamische webpagina die je regelmatig uitleest, bv. met sportuitslagen, evolutie van financiële markten, enz.

Schrijf een script dat de gewenste data op het moment van uitvoeren opvraagt en opslaat in een bestand.

- Het script neemt geen argumenten en produceert geen uitvoer op stdout of stderr.
- De gewenste data wordt opgeslagen in een bestand
  - De directory is instelbaar ahv een variabele
  - De bestandsnaam bevat een timestamp zodat ze correct gesorteerd worden wanneer je `ls -l` doet (bv. in de vorm `data-JJJJMMDD-uummss.json`).
- Eventuele informatieve boodschappen over het verloop van de download, foutboodschappen, enz. worden opgeslagen in een logbestand.

De bedoeling is dat het script op regelmatige tijdstippen uitgevoerd wordt over een bepaalde periode. We gebruiken hiervoor `cron`. Eens de les over `cron` gegeven is, kan je een `crontab` schrijven die bepaalt om de hoeveel tijd het downloadscript uitgevoerd wordt.

Als je beschikt over een Raspberry Pi, Linux-gebaseerde NAS (bv. Synology) of gelijkaardig systeem dat lange tijd kan ingeschakeld blijven, dan kan je het script en de `crontab` daarop installeren en over langere tijd data laten verzamelen. Als je hier niet over beschikt, kan je je laptop een of enkele nachten laten aanstaan en vanuit een Linux-VM werken.

Het resultaat van dit proces zal een directory zijn met vele bestanden in JSON, HTML, XML of een ander tekstgebaseerd bestandsformaat. Een belangrijk principe binnen data-analyse is dat **ruwe data nooit gewijzigd mag worden.** Dat is een van de voorwaarden om er voor te zorgen dat het analyseproces reproduceerbaar is. In het verdere verloop van het proces blijft de inhoud van deze bestanden dus ongewijzigd. Het is dus een goed idee om deze bestanden read-only te maken!

Voorbeeld: een sensor houdt temperatuur, CO_2-gehalte en luchtvochtigheid in een ruimte bij en laat toe de actuele waarde uit te lezen in JSON formaat. Je script slaat het resultaat op in een bestand, bv. `sensor-data-2022-10-04-12-15.json`:

```
{
    "timestamp": "2022-10-04T12:15:01+02:00",
    "co2": "512",
    "temp": "21",
    "humidity: "56"
}
```

1.2. Data transformeren

De JSON- of HTML-bestanden bevatten meestal een hele hoop data waar we niet in geïnteresseerd zijn. In de tweede fase gaan we enkel het nuttige cijfermateriaal eruit halen en in CSV-formaat omzetten.

Schrijf hiervoor een tweede script. Het loopt over alle bestanden die je in de eerste fase hebt verzameld en gebruikt tekstfilters (zie hst 2 van de cursus) om de data uit de verschillende bronnen te combineren in één CSV-bestand.

Om het resultaat makkelijk te kunnen analyseren, zorg je dat je CSV-bestand een hoofding heeft met de naam van alle velden/variabelen en dat elke observatie een aparte regel vormt, bv.:

```
timestamp,co2,temp,humidity
2022-10-04T12:15:01+02:00,512,21,56
2022-10-04T12:30:00+02:00,823,23,61
2022-10-04T13:00:00+02:00,1501,24,65
...
```

Hou er rekening mee dat de ruwe data meestal complexer is van structuur en dat je soms te maken hebt met ontbrekende of corrupte data, bv. omwille van een onderbroken internetverbinding of een hardwareprobleem in een sensor.

Let er ook op dat er twee gangbare formaten zijn voor CSV-bestanden:

- "Angelsaksisch": komma als scheidingsteken en punt als decimaalteken
- "Europees": kommapunt als scheidingsteken en komma als decimaalteken

1.3. Data analyseren

De analysestap zal in reële situaties eerder het werk zijn van de data scientist. Bovendien is dit geen leerdoel van dit opleidingsonderdeel, maar eerder van *Data Science & AI* (2e semester van het 2e modeltraject). De requirements in deze fase willen we dus minimaal houden.

Schrijf een script (Python is hier wellicht meer geschikt voor dan Bash!) dat demonstreert hoe de CSV-data kan geanalyseerd worden. Genereer bv. een grafiek van de evolutie over de tijd van de gemeten waarden afbeeldt en opslaat in een JPG of PNG. Bereken eventueel ook enkele basisstatistieken. Ongetwijfeld vind je op het Internet voorbeeldcode die je kan aanpassen aan jouw specifieke situatie.

Je mag hier ook verder in gaan als je wilt (forecasting, ...), maar dat is niet essentieel.

1.4. Rapport genereren

De laatste stap is om de resultaten van de analyse in een rapport te verwerken. In de praktijk is ook dit vooral de verantwoordelijkheid van de *data scientist* die de resultaten interpreteert en er eventuele aanbevelingen rond formuleert. Maar als *data engineer* kan je ook een rol spelen in dit proces door op zijn minst een deel van het werk te automatiseren.

In deze fase verwerk je de grafieken en berekende statistieken uit de vorige fase in een document. Het is niet de bedoeling dat hier een realistische paper of onderzoeksrapport uit komt! Je mag het opvullen met vultekst, bv. *lorem ipsum*, of het resultaat van een tekstgenerator (bv. SCIgen).

Markdown is een dankbaar formaat om mee aan de slag te gaan. Je stelt eerst een documentsjabloon op, dat je aan de hand van een script aanvult met de afbeeldingen of cijfermateriaal in tabelvorm. In dat document is het ook nuttig om een timestamp toe te voegen dat aangeeft wanneer het gegenereerd is.

Vervolgens zijn er vele mogelijkheden om het resulterende Markdown-bestand om te zetten in een "afgewerkt" formaat, bijvoorbeeld:

- Publiceren op Github Pages (Github zet Markdown automatisch om in HTML)
- Een *static site generator* gebruiken om de Markdown zelf om te zetten in HTML
  - vb. Hugo, Jekyll, mkdocs, ...
- Omzetten in een reveal.js-presentatie
- Omzetten in PDF met bv. Pandoc of een gelijkaardige tool.
- Een combinatie

Je voorziet minstens één vorm, meerdere zijn optioneel. Als je andere ideeën hebt mag je die ook uitwerken!

1.5. Gehele workflow automatiseren

Op dit moment heb je verschillende scripts geschreven die elk een deel van het werk op zich nemen. Voorzie de mogelijkheid dat de gehele workflow automatisch kan lopen, d.w.z. dat als je nieuwe data ophaalt, die meteen verwerkt wordt in het CSV-bestand en dat een nieuw rapport gegenereerd wordt waarin de laatste nieuwe gegevens verwerkt zijn.

Ofwel hou je enkel het meest recente rapport bij, ofwel voorzie je een "backup" van eerdere versies.

Er zijn meerdere mogelijkheden om dit aan te pakken:

- Schrijf een "overkoepelend" script dat de hele workflow achter elkaar uitvoert
- Schrijf een Makefile die, wanneer er een nieuw bestand met ruwe data toegevoegd wordt, de opeenvolgende stappen van de workflow uitvoert
- Automatiseer het proces met Github Actions: wanneer je een nieuw bestand met ruwe data naar de repository pusht, worden de opeenvolgende stappen van de workflow uitgevoerd en is het resultaat te zien op Github Pages
- ...

2. Praktisch

Alle code die je schrijft voor deze opdracht, en ook de ruwe data die je genereert, hou je bij in je Github-repository die je ook voor de labo-opdrachten gebruikt. Maak een nieuwe directory `data-workflow`. Voorzie een `README.md` waarin je toelicht welke data je precies ophaalt, over welke periode

je ze verzamelt hebt, en welke keuzes je gemaakt hebt. Schrijf een handleiding waarin je het gebruik van de scripts uitlegt en hoe je het resultaat kan bekijken. Zorg voor een overzichtelijke directorystructuur!

De deadline voor deze opdracht is **zondag 10 december 2023, 23:59**. Er moet geen afzonderlijk verslag of rapport geschreven worden. De inhoud van de `data-workflow` directory op het moment van de deadline vormt je inzending. Na deze deadline worden er geen wijzigingen meer aangebracht binnen deze directory.

> **Belangrijk! Het heeft geen zin om de lectoren uitstel te vragen op de deadline.** Enkel het opleidingshoofd heeft de bevoegdheid om uitzonderingen toe te staan. Als je vindt dat er omstandigheden zijn waardoor uitstel in jouw geval gerechtvaardigd is, richt je je gemotiveerde aanvraag dus meteen en vóór het verstrijken van de deadline aan het opleidingshoofd.

Je demonstreert het resultaat aan de lector bij wie je de cursus volgt. Dat kan op twee manieren:

- Live tijdens een contactmoment in week 11, 12 of (als de lector dan nog een sessie organiseert) 13, of tijdens het laatste contactmoment voor afstandsstudenten (TIAO).
- Aan de hand van een Panopto schermopname die je deelt met de lector van jouw klasgroep, ten laatste op **zondag 17 december 2023, 23:59**.

Eerst voer je een syntaxcontrole uit op de geschreven scripts, en ook een controle met ShellCheck. Je scripts moeten zo geschreven zijn dat deze geen fouten of opmerkingen geven.

In de demo voer je de scripts uit, toon je aan dat er nieuwe data binnengehaald wordt en dat die samen met reeds eerder verzamelde data verwerkt wordt in een rapport.

Leg ook kort uit hoe je scripts werken, en welke technieken je gebruikt hebt.

Om aan te geven dat je de opdracht hebt uitgevoerd, registreer je je Github-repository en (indien van toepassing) een link naar je Panopto-opname via het daartoe bestemde formulier. Je zal een link op Chamilo vinden. Je doet dit liefst voordat je een live demo geeft bij de lector ofwel ten laatste voor het verlopen van de deadline voor het indienen van een Panopto-opname.

2.1. Beoordeling

Deze opdracht telt mee voor 30% van het examencijfer voor dit opleidingsonderdeel. De score wordt bepaald op basis van volgende criteria:

- **Afwezig**
  - Je hebt de URL naar je Github-repo niet ingediend via het formulier, je repo bevat niet de directory `data-workflow` en/of je hebt geen demo gegeven of opgenomen en doorgestuurd voor het verlopen van de deadline.
- **De standaard is in het geheel niet behaald (0/10)** indien aan één of meerdere van de hieronder opgesomder criteria voldaan is:
  - De directory `data-workflow` is aangemaakt maar bevat overduidelijk niet de nodige inhoud die nodig is om de opdracht realiseren.
  - Niet alle fasen van de workflow zijn geïmplementeerd.
  - De deadlines zijn niet gerespecteerd.
    - Binnen de directory `data-workflow` zijn nog inhoudelijk significante commits gebeurd na het verlopen van de deadline.
    - De link naar de Github repository en/of Panopto opname is na de deadline van 17 december ingediend.
  - De ingediende link naar een Panopto-opname is niet toegankelijk voor de lector.
  - Je hebt de code van een medestudent overgenomen. In dit geval kan een verdere tuchtprocedure opgestart worden (zie OER, deel 6, art. 41-43).
- **De standaard is gedeeltelijk behaald (1-4/10)**
  - Een of meerdere van de scripts bevatten nog syntaxfouten (`bash -n` of `shellcheck`).
  - De scripts werken niet correct, veroorzaken fouten bij het uitvoeren.
  - De opdracht is met een minimum aan inspanningen uitgevoerd, de workflow is erg eenvoudig, er is slechts één eenvoudige dataset/service gebruikt.
  - Er is geen README.md voorzien met toelichting over de gekozen casus, het opzet van de workflow.

- **De standaard is behaald (5-6/10)**
  - Er is een README.md voorzien met toelichting over de gekozen casus, het opzet van de workflow.
  - De gekozen casus is voldoende complex.
  - Voor elke fase is er een shellscript. Enkel het data-analysescript is eventueel in Python geschreven.
  - Elk shellscript doorstaat een controle van de syntax (`bash -n`) en ShellCheck.
  - Elk script werkt zoals beschreven in de opdracht.
  - Er wordt minstens één vorm van een rapport gegenereerd waarin alle tot nu toe verzamelde data verwerkt wordt.
- **De standaard is overtroffen (7-10/10)**
  - Alle criteria van het vorige niveau zijn voldaan.
  - Er is meer dan één soort rapport gegenereerd (bv. webpagina en PDF).
  - De ruwe data is complex qua structuur, maar je toont aan dat je de beschikbare technieken om deze te verwerken beheerst.
  - Je hebt (open source) tools en commando's gebruikt die niet in de cursus aan bod zijn gekomen.
  - Er is een duidelijke inspanning geleverd om een excellent resultaat te bekomen.
  - De codestijl is van een hoog niveau: gebruik van variabelen, functies, structuur script, naamgeving, ...

Belangrijk! Als je afwezig bent voor deze opdracht, betekent dat meteen een afwezigheid voor het gehele opleidingsonderdeel. Het heeft in dat geval geen zin om deel te nemen aan het examen. Ook als je 0/10 behaalt kan je niet slagen voor het vak, ongeacht het resultaat van de andere onderdelen (zie Studiewijzer, *Evaluatie*).