Casus

Eén van onze Data Scientists heeft een nieuw ML model opgeleverd, in een Jupyter Notebook. Als Data Engineering-team nemen we deze in beheer. Het model voorspelt de kosten die we als verzekeraar gaan maken bij een patiënt die in het ziekenhuis opgenomen wordt.

In opdracht 1 hebben we een eerste container werkend gemaakt. In de tussentijd heeft één van de Senior Data Engineers een *pickle*[1] gemaakt van het getrainde ML model. Dit model gaan we nu gebruiken binnen het raamwerk van de applicatie.

Globaal stappenplan

Belangrijk is hier dat we de juiste versies van de software gebruiken – gelukkig is dit voor ons afgevangen door de containers in combinatie met de *virtual environments*. Omdat we van een werkende container uitgaan, volstaan de volgende aanpassingen:

- Aanpassen van *requirements.txt* zodat de juiste Python-modules meekomen
- Aanpassen van het Python-script zodat
    - De juiste modules worden ingeladen
    - Het *pickled* model wordt ingeladen
    - Er een voorspelling kan worden gedaan op basis van zes variabelen
- Interactief maken van deze *Proof of Concept*

Uitwerking

Inloggen op VM

Log in op de VM op het meegeleverde RDP-bestand. Het is een Windows 11-VM met daarop Docker reeds geïnstalleerd.

Loginnaam: student

Wachtwoord: SigmaDataLearning.nl

We werken nu in de map “Opdracht 2”!

Aanpassen van *requirements.txt*

Naast de reeds aanwezige Python-modules zijn ook de volgende modules nodig:

Module	Versie
pycaret	Gelijk aan 1.0.0
numpy	Groter dan of gelijk aan 1.17c

Pas *requirements.txt* aan, en kijk of *docker build* nog steeds werkt (kijk in het vorige opdrachtdocument als je het commando nog even wilt terugvinden)

Aanpassen van het Python-script

Voordat we het ML-model kunnen inladen, moeten we de juiste modules importeren.

[1] Een *pickle* is een Python-object dat in een bestand opgeslagen is. Dit is een manier om bijvoorbeeld ML model niet bij elke uitvoer opnieuw te hoeven trainen, maar een getraind model te kunnen hergebruiken.

1. Voeg daarom bovenin het script de volgende drie regels toe:

```
from pycaret.regression import *
import pandas as pd
import numpy as np
```

Met de benodigde modules in Python ingeladen, kunnen we nu het model inladen. Zoals benoemd staat het *pickled* bestand 'deployment_28042020.pkl' hiervoor klaar in de map. Deze kunnen we inladen met het commando `load_model.`

2. Voeg direct na de regel `app = Flask(__name__)` de volgende regel toe:

```
model=load_model('deployment_28042020')
```

Vervolgens passen we de homepage aan zodat we voor een willekeurig persoon in regio 'northwest' een voorspelling kunnen doen.

3. Vervang de functie `home()` door de volgende twee functies:

```
@app.route('/predict',methods=['POST'])
def predict():
    int_features=[x for x in request.form.values()]
    final=np.array(int_features)
    col = ['age', 'sex', 'bmi', 'children', 'smoker', 'region']
    data_unseen = pd.DataFrame([final], columns = col)
    print(int_features)
    print(final)
    prediction=predict_model(model, data=data_unseen, round = 0)
    prediction=int(prediction.Label[0])
    return render_template('home.html',pred='Verwachte kosten:
{}'.format(prediction))

@app.route('/')
def home():
    return render_template("home.html")
```

4. Doe nu een *docker build* en *docker run* voor bovenstaande wijzigingen.
   Mocht er iets niet goed gaan, denk er dan aan dat je de container eerst stopt voordat je een nieuwe *docker run* probeert. Wanneer je dat niet doet, houdt de container poort 5000 bezet (en kan een nieuwe container zich niet als webserver op die poort registreren).

Als het goed is, krijg je een voorspelling dat deze persoon (35 jaar oud, man, BMI van 19, 3 kinderen, niet-roker uit regio Noord-west) bij zijn opname in het ziekenhuis een verwachte rekening van 15.910 zal gaan krijgen.