De video waar ik mij op het gebaseerd om het project aan te vatten is: *Introduction to data visualization with Python.* Een cursus voorzien op PluralSight.

Om aan data visualisatie te doen wordt aangeraden om het *Anaconda* pakket te downloaden via https://www.anaconda.com/distribution/

In dit pakket zitten alle benodigdheden om met Python aan te slag te gaan om data visueel voor te stellen. Libraries die in dit pakket voorzien zijn:

* Matplotlib: wordt gebruikt om de data effectief te visualiseren
* Pandas: om data te importeren en te verwerken
* Jupyter: een interactieve Python notebook

Ik heb er voor gekozen om de 64-bit grafische interface voor Windows te installeren.

Eens geïnstalleerd kan je via *Anaconda Navigator,* ***Jupyter*** *Notebook* openen. Hiermee kan je code schrijven, organiseren en testen.

Om **Matplotlib** te gebruiken, dien je eerst de pyplot module te importeren. In de Jupyter Notebook geef je volgend commando in: *“from matplotlib import pyplot as plt”* Dit commando geeft aan dat je de pyplot module moet geïmporteerd worden vanuit matplotlib en plt moet genoemd worden.

Om vervolgens een eerste grafiek te tonen, kan je in Jupyter het volgende commando ingeven:

plt.plot([1,2,3],[1,4,9])

Wil je meerdere lijnen op een grafiek tonen:

plt.plot([1,2,3],[1,4,9])

plt.plot([1,2,3],[10,20,30])

Dit zegt dat je op de X-as punten 1, 2 ,3 moet gebruiken en op de Y-as punten 1, 4, 9. Om dit dan te tonen:

plt.show()

Labels toevoegen doe je op de volgende manier:

plt.xlabel('x-as')

plt.ylabel('y-as')

plt.title('test')

Een legend toevoegen:

plt.legend([‘set1’,’set2’])

Exporteren van de grafiek (afbeelding):

* Download afbeelding vanuit de browser
* Met het commando *plt.savefig(‘naamVanDeFile’)* het bestand wordt dan opgeslagen in dezelfde map als deze waar het project in zit.

**Pandas**

Eerst opnieuw imports doen:

from matplotlib import pyplot as plt

import pandas as pd

Eerst gaan we dummy data ingeven:

data = {'Year':[2008, 2012, 2016],

'attendees':[112,321,729],

'average age':[24,43,31]}

Vervolgens steken we deze data in een variabele om er een dataframe van te maken:

df = pd.DataFrame(data)

Wanneer je deze data dan toont met *df* krijg je de data set te zien in tabelvorm.

Om slecht één enkele kolom te selecteren, kan je dan df[‘Year’] (of data.Year) ingeven. Het resultaat is dan een *Panda series* datatype en is vergelijkbaar met een list in Python.

Hier kan je dan vergelijkingsoperatoren op toepassen: df['Year'] < 2013 die dan een reeks van Booleans zullen terug geven.

Wanneer je deze vergelijking in een nieuwe variabele steekt, dan kan je met df[‘nieuweVariabele’] de data van deze vergelijking laten zien. Wat Boolean Indexing wordt genoemd.

Om de waardes dan tov elkaar af te zetten en te tonen in een grafiek, kunnen we ons beroepen op het voorgaande:

plt.plot(df['Year'], df['attendees'])

plt.plot(df['Year'], df['average age'])

plt.legend(['attendees', 'average age'])

plt.show()

Data importeren:

Importeren vanuit een csv file.

1. In het csv-bestand de ‘,’ vervangen door het ‘.’. Anders zal de data van gewicht geïnterpreteerd worden als een object en niet als een float.
2. Pandas en pyplot importeren

import pandas as pd

from matplotlib import pyplot as plt

1. Zorg ervoor dat het csv bestand in dezelfde map zit dan het project.
2. De dataset inladen en toekennen aan een variabele: data = pd.read\_csv('voetbal.csv',sep=’;’)
   1. data geeft dan de ganse tabel weer
   2. data.head() geeft de eerste 5 rijen weer.
3. Waarde van de geboortedatum aanpassen + de kolom inzet vullen:

start = pd.to\_datetime('2011-01-01')

end = pd.to\_datetime('2011-12-31')

start\_u = start.value//10\*\*9

end\_u = end.value//10\*\*9

data['inzet'] = ''

for x, row in data.iterrows():

y = (pd.to\_datetime(np.random.randint(start\_u, end\_u), unit='s'))

data.at[x,'geboortedatum'] = (pd.to\_datetime(np.random.randint(start\_u, end\_u), unit='s'))

if y < pd.to\_datetime('2011-04-01'):

data.at[x, 'inzet'] = 'zeer goed'

elif y < pd.to\_datetime('2011-10-01'):

data.at[x, 'inzet'] = 'goed'

else:

data.at[x, 'inzet'] = 'matig'

1. Gewicht sorteren

gewicht\_gesorteerd = sorted(set(data.gewicht))

1. Grafiek weergeven met de correlatie tussen lengte en gewicht

for x in gewicht\_gesorteerd:

y = data[data.gewicht == x]

plt.scatter(y.gewicht, y.lengte)

plt.xlim(19,31)

plt.ylim(110,140)

plt.xlabel('gewicht')

plt.ylabel('lengte')

plt.show()

TEKST Opdracht 6:

Vraag:

Bereken mbv Python het gemiddelde(a) en de modus van kolom D (aantal gemaakte goals) per positie(b)

1. We willen per positie op het veld het aantal gemaakte goals berekenen. Het gemiddelde kan berekend worden door het aantal goals te delen door het aantal spelers die op deze positie spelen. Hiervoor moeten we dus 2 dingen bijhouden: 1 het aantal gemaakte goals op die positie & 2 het aantal spelers die op die positie spelen.

De for-loop van opgave 5 kan aangepast worden zodat we ook het aantal spelers kunnen bijhouden per positie. Ook houden we een extra dictionary bij (dictionary🡺 mapping tussen een key en een value als data structuur in python)

Vb

sum\_array = {} (🡪 deze kunnen we hergebruiken omdat deze het aantal goals bevat/positie)

player\_per\_position={} (🡪 tellen we het aantal spelers mee op/ positie)

for position in data['positie']:

sum\_array[position]=0

player\_per\_position[position]=0

(De waardes voor de dictionary instantiëren op 0)

for x, row in data.iterrows():

position=data.at[x, 'positie']

sum\_array[position]=sum\_array[position]+ data.at[x, 'aantal gemaakte goalen']

player\_per\_position[position]=player\_per\_position[position] + 1

(🡪 per keer iemand op dezelfde positie, value achter die positie met 1 verhoogd)

Nu kunnen we het gemiddelde uitrekenen per positie want, we hebben het aantal goals- en het aantal spelers op die positie. Dit kan bekomen worden met een for-lus en een nieuwe dictionary waar we elk gemiddelde insteken en na de for lus de dictionary maar gewoon moeten uitlezen en afdrukken.

modus\_array={} 🡪 voor deel b van opgave 6

average\_goals\_position={} (🡪 data structuur voor het gemiddelde/positie in op te slaan)

for position in sum\_array.keys(): (we nemen hier keys 🡪 we willen enkel de posities uitezen

zodat we van deze keys gebruik kunnen maken in for-lus

zelf)

average\_goals\_position[position]=sum\_array[position] / player\_per\_position[position]

(de berekening van het gemiddelde / positie)

modus\_array[position] = []

print(average\_goals\_position[position])

1. We moeten de modus berekenen per positie. De modus is het meest voorkomende waarde uit een verzameling gegevens vb: A={1,2,5,8,5,9,5} dan is de modus van A, 5 omdat deze waarde het vaakst voorkomt.

In de for lus van opgave 6 hebben we een extra dictionary toegevoegd en deze al een lege array per positie laten bijhouden, zo bekomen we

modus\_array= { ‘keeper’: [], ‘staart’: [], ‘linkervleugel’: [], ‘rechtervleugel’: [], ‘piloot’: []}

dit doen we zodat we per positie het aantal goals gemaakt speler kunnen gaan bijhouden zodat we een verzameling krijgen. Deze verzameling per positie hebben we nodig om de methode: (mode(verzameling)) te kunnen gebruiken die bij de package statistics hoort.

Stap 1) we importen het package voor de mode-methode te kunnen gebruiken:

from statistics import mode

Stap 2)

We lopen de gehelen Excel sheet af per row en haalden de gemaakte goalen uit elke row en slaan deze op in onze eerder aangemaakte dictionary (modus\_array) zodat we alle doelpunten / positie hebben als een gegevens verzameling.

for x, row in data.iterrows():

position = data.at[x, 'positie']

modus\_array[position].append(data.at[x,'aantal gemaakte goalen'])

Stap 3)

We doorlopen onze gegevens verzameling per positie en passen de methode die we in stap 1 geïmporteerd hebben toe op de gegevens verzamelingen en printen het resultaat meteen af.

for key in modus\_array.keys():

print ('Mode for ' + key + ': ' + str((mode(modus\_array[key]))))