Classificatie van persoonlijkheidskenmerken

Rijksuniversiteit Groningen

Faculteit der Letteren - Informatiekunde

Information Retrieval - LIX018B05

Begeleider: Gosse Bouma

Mart Busger op Vollenbroek S2174634

Rolf Kuipers S2214415

11 januari 2015

Inhoudsopgave

Introductie **3**

Literatuurbespreking **4**

The Big five *4*

Workshop vloggers *4*

Rainbow documentatie *4*

Methode en experiment **5**

Data *5*

Methode *5*

Experiment *6*

Resultaten en discussie **8**

Resultaten *8*

Discussie en verder onderzoek *9*

Literatuurlijst **10**

Bijlagen **11**

Bijlage 1: resultaten 5 x 10-fold Cross Validation tests *11*

Bijlage 2: Python scripts voor de class "Brief", data opslag en file writing *12*

Script 1: class "Brief" *12*

Script 2: Data opslag *14*

Script 3: File writer *15*

Introductie

Om tegenwoordig aangenomen te worden bij een bedrijf moet er aan tal van aspecten gedacht worden. Er zijn screeningprocedures, waarbij gekeken wordt naar de privéomstandigheden van de selectiekandidaten, sollicitatiegesprekken tussen de kandidaten en de Human Resources afdeling, maar een van de belangrijkste aspecten is misschien wel de sollicitatiebrief met daarin motivaties. De Rijksoverheid gebruikt deze brieven in combinatie met persoonlijkheidstests om een inschatting te maken van de sollicitanten. De persoonlijkheidstests zijn zo gestructureerd dat ze een aantal vooraf bepaalde persoonlijkheidskenmerken meten op een schaal van 1 t/m 5. Niet alle persoonlijkheidskenmerken zijn voor dit onderzoek van belang. Er wordt gekeken naar de zogenoemde "Big 5": extravert, innovatief, stabiel, vriendelijk en zorgvuldig. Een 5 zou op de schaal van extraversie inhouden dat de persoon in kwestie extravert is en een 1 houdt in dat de persoon introvert is. Dit geldt net zo voor de andere 4 persoonlijkheidskenmerken en hun tegenpolen. Deze brieven, de combinatie van motivatieteksten en de relevante persoonlijkheidskenmerken, vormen de relevante data voor het onderzoek.
Zeker op de hedendaagse arbeidsmarkt zijn er veel mensen die op zoek zijn naar een baan en al deze mensen versturen sollicitatiebrieven. Het verwerken van de sollicitatiebrieven en de persoonlijkheidstests zijn tijdrovende bezigheden, zeker voor een organisatie als de Rijksoverheid. Als de persoonlijkheidstests gebaseerd zouden kunnen worden op de motivatieteksten uit de sollicitatiebrieven, zou dit veel tijd en moeite kunnen schelen in het sollicitatieproces. In dit onderzoek wordt daarom gezocht naar mogelijkheden om op basis van de motivatieteksten van de sollicitanten hun persoonlijkheidskenmerken te meten op een schaal van 1 t/m 5. Als dit mogelijk blijkt, kan er veel tijd bespaard worden door een programma de tekst te laten analyseren en op basis daarvan de persoonlijkheidskenmerken van de sollicitanten te bepalen. Door deze resultaten te analyseren kan er vaak direct een scheiding gemaakt worden tussen geschikte en ongeschikte kandidaten voor de betreffende functie.
Om dit onderzoek verder vorm te geven, zal de onderzoeksvraag als volgt geformuleerd worden:

Hoe kunnen op een zo accuraat mogelijke manier de persoonlijkheidskenmerken van sollicitanten bepaald worden op basis van hun motivatieteksten om daarmee tijd in het selectieproces te besparen?

De hypothese die hieruit volgt is dat motivatieteksten van sollicitanten een goede manier zijn om de Big five persoonlijkheidskenmerken van sollicitanten te bepalen.
De rest van dit verslag is als volgt ingedeeld: hoofdstuk 2 zal wetenschappelijke literatuur behandelen om op basis waarvan onderzoek gedaan gaat worden, hoofdstuk 3 zal de gebruikte methoden en technische aspecten van het experiment behandelen, hoofdstuk 4 zal de rapportage van de resultaten en een discussie bevatten, dan zal de literatuurlijst volgen en ten slotte zijn de bijlagen toegevoegd.

Literatuurbespreking

The Big five

The Big five persoonlijkheidskenmerken zijn begin twintigste eeuw ontwikkeld door verschillende onderzoekers. Een van de eerste vermeldingen komt van McDougall (1932): “personality may with advantage be broadly analyzed into five distinguishable but inseparable factors, namely, intellect, character, temperament, disposition, and temper”. Hier is in feite de basis gelegd voor het verdere onderzoek dat in de twintigste eeuw gedaan is. McDougall beargumenteert dat deze vijf persoonlijkheidskenmerken gezamelijk grotendeels de persoonlijkheid van een persoon kunnen bepalen. Omdat deze persoonlijkheidskenmerken zo belangrijk geacht worden, zijn ze in latere jaren door veel wetenschappers als punt van onderzoek gekozen en worden ze nog steeds gebruikt als belangrijke pijlers voor de persoonlijkheid van personen. Vandaar dat het voor instanties als de Rijksoverheid ook van belang is om deze kenmerken vast te leggen.

Workshop vloggers

Er is al eerder onderzoek gedaan naar het bepalen van persoonlijkheidskenmerken op basis van teksten. Verhoeven, Company en Daelemans (2014) hebben op basis van transcripts van Youtube-video’s geprobeerd te persoonlijkheden van vloggers (video bloggers) te bepalen. Dit onderzoek is geïnspireerd door voorgaand onderzoek door Soler en Wanner (2014) waarin gekeken is naar content gebaseerde eigenschappen die het geslacht van een persoon moeten inschatten. Het uitgangspunt dat Verhoeven et al. genomen hebben luidt als volgt: "de content gebaseerde eigenschappen zeggen meer over de persoon dan over de tekst en dit kan gebruikt worden om persoonlijkheidskenmerken te bepalen”. In dit onderzoek is gebruik gemaakt van Scikit-learn’s algoritme en er is getest met unigrammen, trigrammen, “Linguistic inquery and Word count” (LIWC) en combinaties van de drie hiervoor genoemde benaderingen. In dit onderzoek worden binaire scores gebruikt om te bepalen of een persoon bijvoorbeeld extravert is of niet. Dat wil zeggen dat er of een score van 0 voor niet extravert (introvert) of een score van 1 voor extravert toegekend wordt.

Rainbow documentatie

Om verschillende manieren van tekst classificatie te testen, is er gebruik gemaakt van de commandline tool Rainbow. Rainbow is een document classifier front-end van libbow. De documentatie van deze tool kan aangeroepen worden met met het commando ‘rainbow —help’. In deze documentatie staan alle opties en methoden die mogelijk zijn om met behulp van Rainbow tekst te classificeren. Voorbeelden van methoden zijn naivebayes en nbshrinkage.

Methode en experiment

Data

Voor het onderzoek is een dataset van de Rijksoverheid toegankelijk gemaakt. In deze dataset staan ± 9000 sollicitatiebrieven met daarin persoonlijke gegevens zoals geslacht, leeftijd, etniciteit, motivatieteksten en de resultaten van een persoonlijkheidstest. Niet alle data uit de corpus zijn relevant voor ons onderzoek, zo spelen geslacht en etniciteit en bepaalde persoonlijkheidskenmerken geen rol voor het bepalen van de Big five in het onderzoek. De corpus is aangeleverd in het csv-formaat, waardoor de data niet direct ingelezen kunnen worden in het programma. Er moeten een aantal stappen ondernomen om de data geschikt te maken voor het gebruik met Rainbow.

Methode

Voor het onderzoek is gebruik gemaakt van een aantal scripts die in Python 3.4 geschreven zijn. Allereerst wordt er een class genaamd “Brief" aangemaakt, met de Big five als de methoden van de class (figuur 1).

```
def setInnovatie(self, innovatie):
    self.innovatie = float(innovatie)

    if self.innovatie < 3.0:
        self.innoclass = 'non-innovatief'
    if self.innovatie == 3.0:
        self.innoclass = 'neutraal'
    if self.innovatie > 3:
        self.innoclass = 'innovatief'
```

Figuur 1: voorbeeld methode Innovativiteit

In figuur 1 is tevens te zien hoe gemeten wordt of een persoon innovatief of non-innovatief is. In de resultaten van de persoonlijkheidstest uit de sollicitatiebrieven worden de scores van de persoonlijkheidskenmerken gegeven op een schaal van 1 t/m 5, deze worden in ons programma als volgt geïnterpreteerd: als de score 1 of 2 is, is de persoon (in het geval van figuur 1) non-innovatief, bij een score van 3 is de persoon niet non-innovatief of innovatief, en bij een score van 4 of 5 is de persoon innovatief.
Een andere set-methode in de class “Brief" is de setTekst methode. Hierin worden de motivatieteksten gestemd. Hiervoor is de Porter-stemmer gebruikt uit de Natural Language Toolkit (nltk), dit is in figuur 2 te zien.

```
def setTekst(self, tekst):
    self.tekst = tekst

    tokens = nltk.word_tokenize(self.tekst)
    porter = stem.porter.PorterStemmer()
    stemming = [porter.stem(i) for i in tokens]
    self.tekst = " ".join(stemming)
```

Figuur 2: Porter-stemmer in "Brief"

De Porter-stemmer zorgt ervoor dat Rainbow de tekst beter kan lezen en daardoor beter in kan schatten wat voor persoonlijkheidskenmerk de betreffende persoon heeft. Bepaalde woorden worden ingekort en van sommige meervouden wordt enkelvoud gemaakt zodat deze beter met elkaar te vergelijken zijn.
Om vervolgens de data uit de corpus in de class te krijgen, wordt er gebruik gemaakt van de Pandas module. Deze module is gemaakt om csv-bestanden te importeren in

```
from Brief import Brief
import pandas as pd
import pickle

def main():
    # Maak gebruik van pandas-module om kolommen uit csv. bestand te halen
    df = pd.read_csv('corpus.csv')
    ID = df['User Number']
    motivatie = df['Motivatie']
    maatschap = df['Maatschappelijkebetrokkenheid']
    extraversie = df['csel_extr']
    innovatie = df['csel_inno']
    stabiliteit = df['csel_stab']
    vriendelijkheid = df['csel_vrie']
    zorgvuldigheid = df['csel_zrgv']
```

Figuur 3: gebruik van Pandas module

Python. Dit kan gedaan worden met behulp van de kolomnamen in het csv-bestand. Nadat de de data uit de kolommen gehaald zijn, wordt er voor elke brief een object van de “Brief" class aangemaakt met behulp van de data. Voor de opslag van deze data wordt gebruik gemaakt van de Pickle module.

Het laatste onderdeel van de methode is het aanmaken van de bestanden die gebruikt kunnen worden door Rainbow. Het script dat hiervoor gebruikt is, laadt de data met behulp van Pickle in en plaatst deze in vooraf aangegeven dictionaries. Deze twee dictionaries worden aangemaakt in de paden die aangegeven zijn in het script (figuur 4). Vervolgens moet er gekozen worden hoe de data naar de dictionaries geschreven worden. Hier zijn drie benaderingen voor gebruikt: normale tekst, bigrammen of een combinatie van beide. Voordat het script gebruikt wordt, moet eerst in het script aangegeven worden welke benadering gebruikt gaat worden. In figuur 5 en 6 is te zien hoe gekozen kan worden tussen de drie benaderingen:

```
import pickle
from Brief import Brief
import uuid
import os.path
import nltk
from nltk import bigrams
from nltk import trigrams

def main():
    # Retrieve pickle data created by data-retriever program.
    data = pickle.load(open('data.pickle', 'rb'))

    # Select path for the two dictionaries.
    # --- In this case the doubles of instabiel/stabiel
    path1 = '/home/rolfkuipers/Dropbox/InformationRetrieval/instabieldouble'
    path2 = '/home/rolfkuipers/Dropbox/InformationRetrieval/stabieldouble'
```

Figuur 4: Laden van de data en creëren van paden voor dictionaries

```
for obj in data:
    f = uuid.uuid4()
    if obj.getStabClass() == 'instabiel':
        name = os.path.join(path1, f.hex)

    ### OPTION 1    -- Normal text ###
        tekst = obj.Tekst()

    ### OPTION 2    -- Bigrams ###
        # tekst = ""
        bigram = bigrams(tekst.split())
        for gram in bigram:
            tekst = tekst + " " + str(gram)

    ### OPTION 3    -- Trigrams ###
        #trigram = trigrams(obj.Tekst().split())
        #for tri in trigram:
            #tekst = tekst + " " + str(tri)

        bestand = open(name, 'w')
        bestand.write(tekst)
        bestand.close()
```

Figuur 5: instabiel met bigram-methode

```
if obj.getStabClass() == 'stabiel':
    name = os.path.join(path2,f.hex)
### OPTION 1    -- Normal text ###
    tekst = obj.Tekst()

### OPTION 2    -- Bigrams ###
    # tekst = ""
    bigram = bigrams(tekst.split())
    for gram in bigram:
        tekst = tekst + " " + str(gram)

### OPTION 3    -- Trigrams ###
    #trigram = trigrams(obj.Tekst().split())
    #for tri in trigram:
        #tekst = tekst + " " + str(tri)

    bestand = open(name, 'w')
    bestand.write(tekst)
    bestand.close()
```

Figuur 6: stabiel met bigram-methode

Experiment

Als de data naar de betreffende dictionaries geschreven zijn kan de tekst met behulp van Rainbow geclassificeerd worden. Eerst dienen de gemaakte dictionaries geïndexeerd te worden met de 4000 meest informatieve woorden. Dit kan gedaan worden met het commando 'rainbow –T 4000 --index instabieldouble/ stabieldouble/‘. In dit onderzoek worden twee grote tests uitgevoerd door gebruik te maken van twee methoden in Rainbow: naivebayes en nbshrinkage.

Er is gekozen voor om vijf keer een 10-fold Cross-Validation test uit te voeren voor beide methoden. Op deze manier zijn er voldoende resultaten om statistische tests uit te voeren op de resultaten. Er is gekozen om te trainen op 90% van de data en te testen op 10% van de data.
De gebruikte commando's voor de naivebayes methode en de nbshrinkage methode zijn respectievelijk:
rainbow --test=10 --method=naivebayes --test-set=0.1 | rainbow-stats
rainbow --test=10 --method=nbshrinkage --test-set=0.1 | rainbow-stats

Resultaten en discussie

Resultaten

Omdat er twee grote tests uitgevoerd zijn, de naivebayes- en de nbshrinkage test, zijn er twee sets met resultaten. Omwille de duidelijkheid is er gekozen om de gemiddelde resultaten van cross validation tests in dit verslag te vermelden, voor de volledige resultaten zie bijlage 1. De gemiddelden van de verschillende methoden en benaderingen zijn terug te vinden in tabel 1.

	Naivebayes			Nbshrinkage		
	Normaal	Bigrammen	Combinatie	Normaal	Bigrammen	Combinatie
Extraversie	81.812	80.288	81.918	82.746	82.028	82.682
Vriendelijkheid	80.834	79.244	81.462	83.574	84.052	83.542
Zorgvuldigheid	81.538	78.974	82.460	84.426	83.764	83.860
Innovativiteit	80.922	79.690	81.014	82.748	82.782	82.306
Stabiliteit	80.998	79.208	81.536	82.758	81.392	82.470

Tabel 1: gemiddelde resultaten naivebayes en nbshrinkage

In de tabel staan waarden die de accuracy van de tekstclassificatie weergeven. Een score van 81.812 voor extraversie betekent dat rainbow in dit geval 81.8% van de motivatieteksten in de juiste klasse voorspelt. Voor alle Big five zijn er twee klassen: extravert/introvert, vriendelijk/ onvriendelijk, zorgvuldig/onzorgvuldig, innovatief/non-innovatief en stabiel/instabiel.
Uit de resultaten blijkt dat bij het gebruik van de naivebayes-methode het verschil tussen de drie benaderingen het grootst is. Door een ANOVA test uit te voeren met de gegevens van de naivebayes methode en de drie benaderingen als groepen te gebruiken, blijkt dat dit verschil significant is ($F = 26.90$, $p < 0.0001$). Dit betekent dat dat de combinatie van de normale tekst en bigrammen het meeste succes heeft bij de naivebayes-methode.
Bij het gebruik van de nbshrinkage-methode zijn de verschillen tussen de drie benaderingen niet aanwezig. Een ANOVA test met wederom de drie benaderingen als groepen wijst ook uit dat dit klopt ($F = 0.3319$, $p = 0.72$). Omdat de resultaten niet significant zijn, kan er niet aangetoond worden dat er een verschil tussen de benaderingen bestaat.
Omdat de combinatie-benadering bij de naivebayes-methode de hoogste resultaten oplevert en er geen significant verschil is tussen de benaderingen bij de nbshrinkage-methode, hebben we besloten om de twee combinatie benaderingen met elkaar te vergelijken. Omdat dit twee onafhankelijke groepen met dezelfde type resultaten zijn, wordt er een unpaired t-test gebruikt. Uit tabel 2 kan worden afgeleid dat door het gebruiken van de nbshrinkage-methode de resultaten significant hoger zijn dan de resultaten bij het gebruik van de naivebayes-methode ($t = 6{,}6537$, $p < 0{,}0001$).

	Naivebayes (combinatie)	Nbshrinkage (combinatie)
Mean	81,6776	82,9720
Standard deviation	0,5923	0,7717
t-value	6,6537	
p-value	< 0,0001	

Tabel 2: resultaten unpaired t-test

Discussie en verder onderzoek

Op basis van het onderzoek en de resultaten die het onderzoek opgeleverd heeft, blijkt dat de nbshrinkage-methode de hoogste resultaten oplevert om de Big five persoonlijkheidskenmerken te voorspellen op basis van motivatieteksten uit sollicitatiebrieven. Normaal gesproken zorgen shrinkage-methoden ervoor dat de resultaten dichter bij het gemiddelde komen te liggen, zoals te zien is in tabel 1 en de ANOVA test die uitwijst dat de het verschil tussen de benaderingen niet meer significant is. Echter, shrinkage heeft er ook voor gezorgd dat de resultaten overall beter zijn geworden. Dit heeft de unpaired t-test uit tabel 2 uitgewezen. Hiermee is ook antwoord gegeven op de onderzoeksvraag die in de introductie geformuleerd is. Met behulp van rainbow en de nbshrinkage-methode kan er in 83.0% de juiste klasse berekend worden.
In dit onderzoek is niet gekeken waarom shrinkage gezorgd heeft voor hogere resultaten bij het classificeren van de Big five. Dit is een van de punten voor verder onderzoek. Daarnaast kan er gekeken worden naar ander rainbow-methoden. In dit onderzoek is er gebruik gemaakt van van de naivebayes-methode en de nbshrinkage-methode, maar er zijn nog tal van andere methoden die wellicht hogere resultaten op kunnen leveren. Er zijn pogingen gedaan met fidf_words, tfidf_log_words, tfidf_log_occur, tfidf, nbsimple, knn, emsimple en kl, maar al deze methoden leverden veel lagere resultaten op de naivebayes en nbshrinkage. Ten slotte kan er nog verder onderzoek gedaan worden naar wat nu precies de indicators zijn voor de Big five en daarmee onderzoeken wat precies het verschil is tussen de motivatieteksten.

Literatuurlijst

Company, J. S., & Wanner, L. How to Use Less Features and Reach Better Performance in Author Gender Identification.
http://www.lrec-conf.org/proceedings/lrec2014/pdf/104_Paper.pdf

McCallum, A., & Nigam, K. (1998, July). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization* (Vol. 752, pp. 41-48).
http://staff.icar.cnr.it/manco/Teaching/2005/datamining/articoli/multinomial-aaaiws98.pdf

McDougall, W. (1932). Of the words character and personality. *Journal of Personality*, *1*(1), 3-16.
http://publikationen.ub.uni-frankfurt.de/oai/container/index/docId/12354

Judge, T. A., Higgins, C. A., Thoresen, C. J., & Barrick, M. R. (1999). The big five personality traits, general mental ability, and career success across the life span. *Personnel psychology*, *52*(3), 621-652.
http://people.tamu.edu/~mbarrick/Pubs/1991_Barrick_Mount.pdf

Bijlagen

Bijlage 1: resultaten 5 x 10-fold Cross Validation tests

Naivebayes

A = Normale tekst B = Bigrammen C = Normal + Bigram

	1	2	3	4	5
A	80.94	81.50	82.50	82.34	81.78
B	80.31	80.49	80.31	80.15	80.18
C	81.55	81.93	82.22	82.13	81.76

Introvert / extravert

4429 1647

	1	2	3	4	5
A	80.42	81.14	81.61	80.84	80.16
B	79.34	78.86	79.10	79.58	79.34
C	82.39	80.84	81.47	81.14	81.47

Onvriendelijk / vriendelijk

2671 2780

	1	2	3	4	5
A	80.86	81.17	81.83	82.51	81.32
B	79.36	78.57	78.35	78.94	79.65
C	82.38	82.23	82.45	82.31	82.93

Onzorgvuldig / zorgvuldig

3146 2316

	1	2	3	4	5
A	81.16	80.61	80.78	80.65	81.41
B	79.91	79.34	79.63	79.57	80.00
C	81.50	80.97	80.61	81.12	80.87

Non-innovatief / innovatief

3864 2007

	1	2	3	4	5
A	80.49	80.67	81.16	81.30	81.37
B	79.51	79.04	78.86	79.11	79.52
C	81.64	81.60	81.74	81.45	81.25

Instabiel / stabiel

4043 1825

Nbshrinkage

A = Normale tekst B = Bigrammen C = Normal + Bigram

	1	2	3	4	5
A	82.24	83.23	83.10	83.08	82.08
B	82.32	82.03	81.85	81.50	82.44
C	82.65	83.11	82.92	81.76	82.97

Introvert / extravert
4429 1647

	1	2	3	4	5
A	83.08	83.80	84.07	83.43	83.49
B	83.82	83.08	84.59	84.09	83.96
C	83.19	84.04	83.78	82.77	83.93

Onvriendelijk / vriendelijk
2671 2780

	1	2	3	4	5
A	83.85	84.16	84.98	84.38	84.76
B	84.03	83.57	84.10	83.38	83.74
C	84.10	83.61	84.18	84.43	82.98

Onzorgvuldig / zorgvuldig
3146 2316

	1	2	3	4	5
A	82.71	83.37	82.67	82.74	82.25
B	83.17	82.18	83.03	82.57	82.96
C	82.21	82.35	81.89	82.78	82.30

Non-innovatief / innovatief
3864 2007

	1	2	3	4	5
A	82.27	82.80	82.10	82.78	83.84
B	81.26	81.26	81.52	81.66	81.26
C	82.28	82.65	81.91	82.44	83.07

Instabiel / stabiel
4043 1825

Bijlage 2: Python scripts voor de class "Brief", data opslag en file writing

Script 1: class "Brief"

```
#!/usr/bin/Python3.4

# Brief.py
# Rolf Kuipers & Mart Busger op Vollenbroek
# Information Retrieval, Rijksuniversiteit Groningen

import nltk
from nltk import stem
#from nltk.corpus import stopwords

class Brief:
	def __init__(self, ID):
		self.ID = ID

	def setTekst(self, tekst):
		self.tekst = tekst
```

```
		tokens = nltk.word_tokenize(self.tekst)
		#newtokens = [w for w in tokens if w.lower() not in stopwords.words('dutch')]
		#snowball = stem.snowball.DutchStemmer()
		porter = stem.porter.PorterStemmer()
		stemming = [porter.stem(i) for i in tokens]
		self.tekst = " ".join(stemming)

def setInnovatie(self, innovatie):
		self.innovatie = float(innovatie)

		if self.innovatie < 3.0:
			self.innoclass = 'non-innovatief'
		if self.innovatie == 3.0:
			self.innoclass = 'neutraal'
		if self.innovatie > 3:
			self.innoclass = 'innovatief'

def getInnoClass(self):
		return self.innoclass

def setExtraversie(self, extraversie):
		self.extraversie = float(extraversie)

		if self.extraversie < 3.0:
			self.extraclass = 'introvert'
		if self.extraversie == 3.0:
			self.extraclass = 'neutraal'
		if self.extraversie > 3.0:
			self.extraclass = 'extravert'

def getExtraClass(self):
		return self.extraclass

def setVriendelijkheid(self,vriendelijkheid):
		self.vriendelijkheid = float(vriendelijkheid)

		if self.vriendelijkheid < 3.0:
			self.vriendclass = 'onvriendelijk'
		if self.vriendelijkheid == 3.0:
			self.vriendclass = 'neutraal'
		if self.vriendelijkheid > 3.0:
			self.vriendclass = 'vriendelijk'

def getVriendClass(self):
		return self.vriendclass

def setStabiliteit(self, stabiliteit):
		self.stabiliteit = float(stabiliteit)

		if self.stabiliteit < 3.0:
			self.stabclass = 'instabiel'
		if self.stabiliteit == 3.0:
			self.stabclass = 'neutraal'
		if self.stabiliteit > 3.0:
			self.stabclass = 'stabiel'

def getStabClass(self):
		return self.stabclass

def setZorgvuldigheid(self,zorgvuldigheid):
		self.zorgvuldigheid = float(zorgvuldigheid)

		if self.zorgvuldigheid < 3.0:
```

```
                self.zorgclass = 'onzorgvuldig'
        if self.zorgvuldigheid == 3.0:
                self.zorgclass = 'neutraal'
        if self.zorgvuldigheid > 3.0:
                self.zorgclass = 'zorgvuldig'

def getZorgClass(self):
        return self.zorgclass

def Tekst(self):
        return self.tekst

def Extraversie(self):
        return self.extraversie

def Innovatie(self):
        return self.innovatie

def Vriendelijkheid(self):
        return self.vriendelijkheid

def Stabiliteit(self):
        return self.stabiliteit

def Zorgvuldigheid(self):
        return self.zorgvuldigheid
```

Script 2: Data opslag

```
#!/usr/bin/Python3.4

# Store_data.py
# Rolf Kuipers & Mart Busger op Vollenbroek
# Information Retrieval, Rijksuniversiteit Groningen

from Brief import Brief
import pandas as pd
import pickle

def main():
        # Maak gebruik van pandas-module om kolommen uit csv. bestand te halen
        df = pd.read_csv('corpus.csv')
        ID = df['User Number']
        motivatie = df['Motivatie']
        maatschap = df['Maatschappelijkebetrokkenheid']
        extraversie = df['csel_extr']
        innovatie = df['csel_inno']
        stabiliteit = df['csel_stab']
        vriendelijkheid = df['csel_vrie']
        zorgvuldigheid = df['csel_zrgv']

        # Helaas werkte (door index-range fouten) alle object in 1 loop creëren met alle attributen niet.
        # Daarom eerst alleen het ID toevoegen en daarna alle andere aspecten d.m.v. Setters.
        brief_dic = []
        for num in ID:
                brief_dic.append(Brief(num))
        for i in range(len(motivatie)):
                tekst = motivatie[i] + maatschap[i]
                brief_dic[i].setTekst(tekst)
                brief_dic[i].setInnovatie(innovatie[i])
                brief_dic[i].setExtraversie(extraversie[i])
                brief_dic[i].setStabiliteit(stabiliteit[i])
                brief_dic[i].setVriendelijkheid(vriendelijkheid[i])
```

```
                brief_dic[i].setZorgvuldigheid(zorgvuldigheid[i])

        pickle.dump(brief_dic, open("data.pickle", 'wb'))

main()
```

Script 3: File writer

```
# file_writer_2.py
# Rolf Kuipers & Mart Busger op Vollenbroek
# Information Retrieval, Rijksuniversiteit Groningen

""" Creates files in given dictionary for later use."""

import pickle
from Brief import Brief
import uuid
import os.path
import nltk
from nltk import bigrams
from nltk import trigrams

def main():
        # Retrieve pickle data created by data-retriever program.
        data = pickle.load(open('data.pickle', 'rb'))

        # Select path for the two dictionaries.
        # --- In this case the doubles of instabiel/stabiel
        path1 = '/home/rolfkuipers/Dropbox/InformationRetrieval/instabieldouble'
        path2 = '/home/rolfkuipers/Dropbox/InformationRetrieval/stabieldouble'

        # Check every object for the given property
        # --- In this case instabiel or stabiel.
        for obj in data:
                f = uuid.uuid4()
                if obj.getStabClass() == 'instabiel':
                        name = os.path.join(path1, f.hex)

                ### OPTION 1      -- Normal text ###
                        tekst = obj.Tekst()

                ### OPTION 2      -- Bigrams ###
                        # tekst = ""
                        bigram = bigrams(tekst.split())
                        for gram in bigram:
                                tekst = tekst + " " + str(gram)

                ### OPTION 3      -- Trigrams ###
                        #trigram = trigrams(obj.Tekst().split())
                        #for tri in trigram:
                                #tekst = tekst + " " + str(tri)

                        bestand = open(name, 'w')
                        bestand.write(tekst)
                        bestand.close()
                if obj.getStabClass() == 'stabiel':
                        name = os.path.join(path2,f.hex)
                ### OPTION 1      -- Normal text ###
                        tekst = obj.Tekst()

                ### OPTION 2      -- Bigrams ###
                        # tekst = ""
                        bigram = bigrams(tekst.split())
                        for gram in bigram:
```

```
                        tekst = tekst + " " + str(gram)

        ### OPTION 3      -- Trigrams ###
                #trigram = trigrams(obj.Tekst().split())
                #for tri in trigram:
                        #tekst = tekst + " " + str(tri)

                bestand = open(name, 'w')
                bestand.write(tekst)
                bestand.close()


main()
```