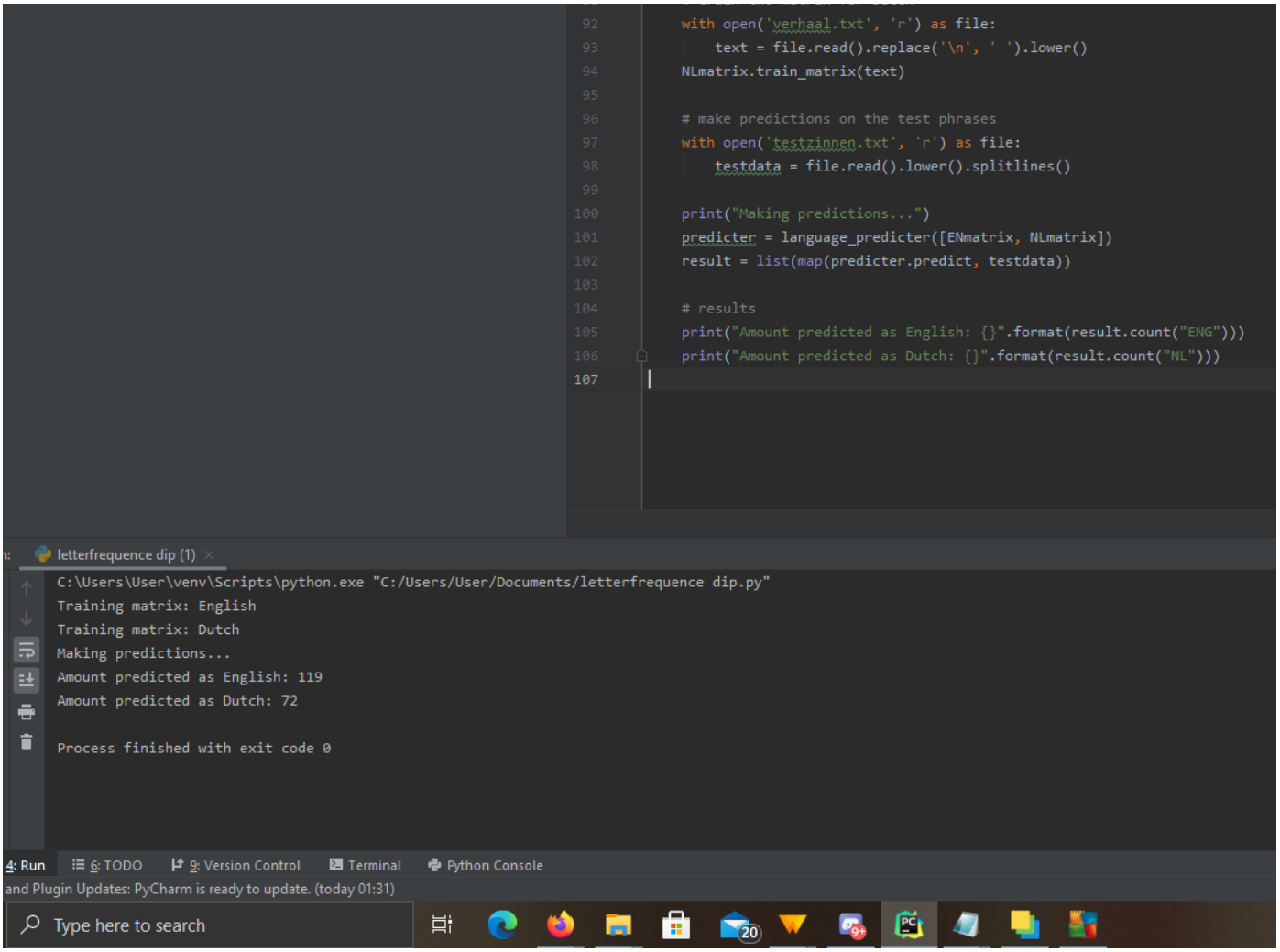
Documentatie

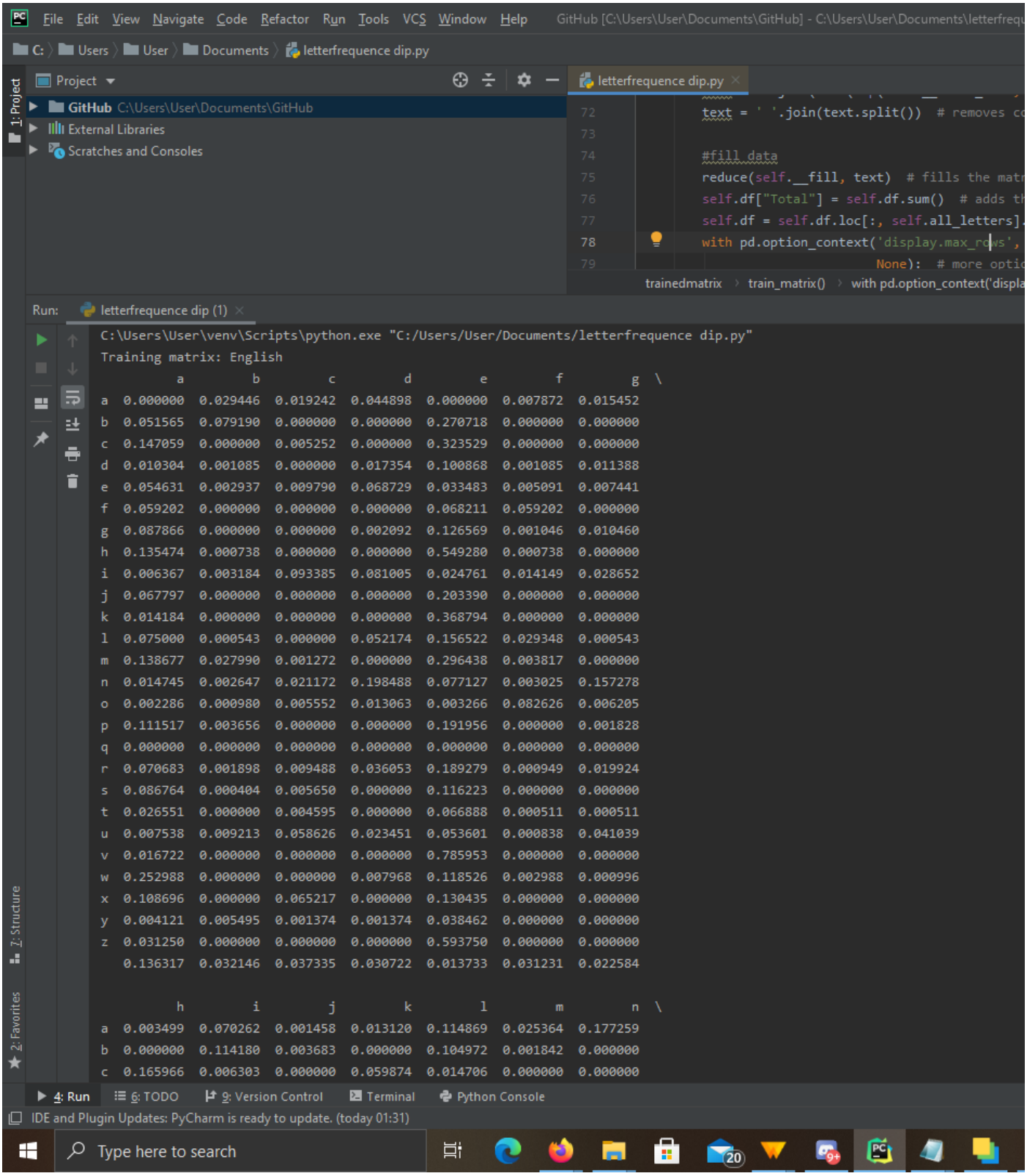
Hieronder zijn screenshots te zien waaronder het resultaat van het programma, de uitkomst van getrainde bigram matrices, en elke prediction van elke test regel in volgorde. Het resultaat van het programma laat alleen 72 van de 72 nederlandse regels zien, dat komt omdat het bestand met test regels een lege regel had die verwijdert moest worden om het programma te laten werken. Het programma heeft een accuracy van 100%

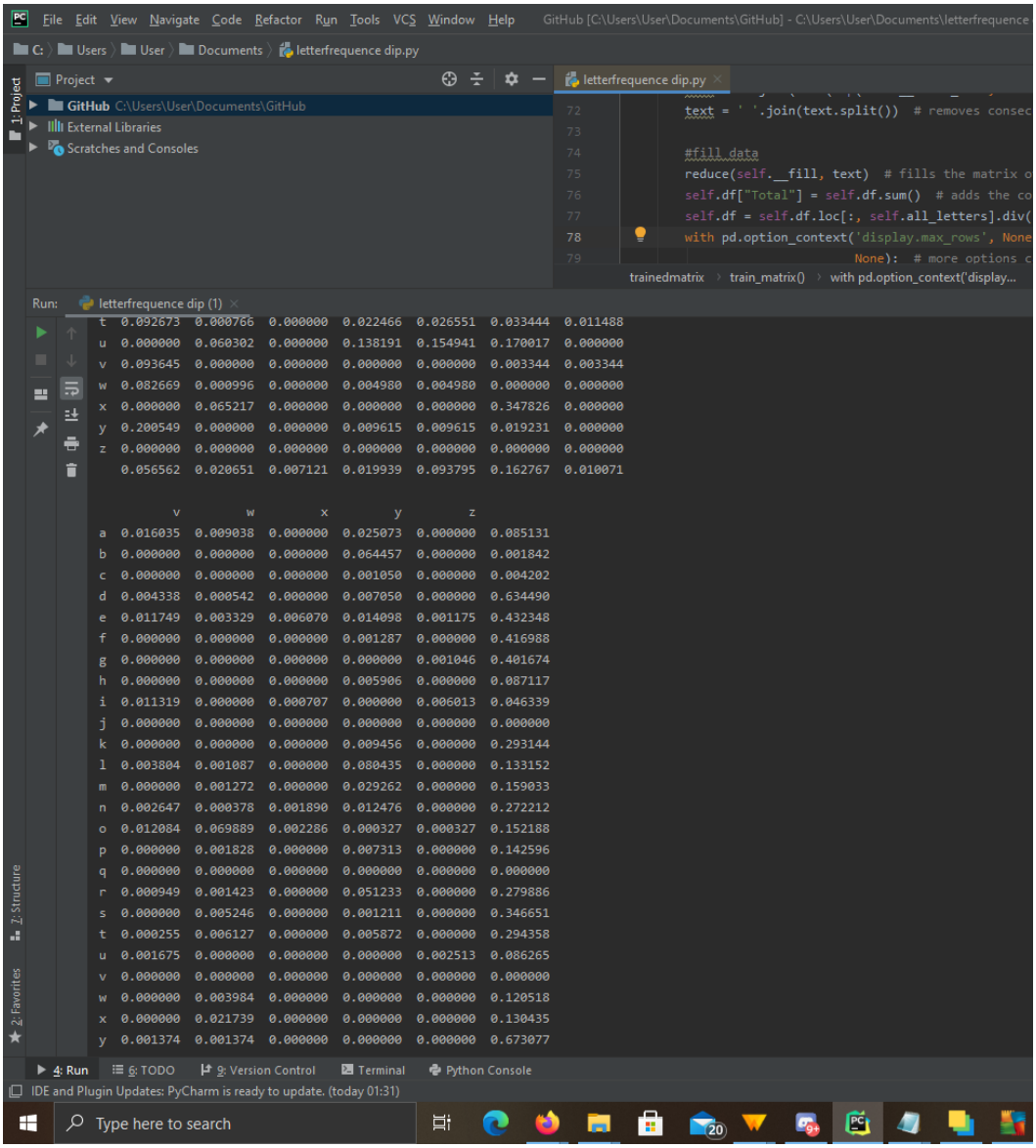
Het programma werkt als volgt. Eerst traint het voor elke taal een matrix door deze op te vullen met hoe vaak een combinatie aan letter voorkomt. Deze matrix is opgeslagen in de vorm van een Pandas dataframe. De tekst wordt uit een txt file gehaald en gegeven aan de class om te beginnen met de matrix trainen. Eerst wordt de tekst schoon gemaakt door speciale karakters en dubbele spaties weg te halen. Hier wordt een Map functie voor gebruikt. Daarna wordt de uitkomst van de schoongemaakte tekst in een Reduce functie gezet om het in de dataframe te zetten. Elk paar letters in de tekst wordt gelezen en dan wordt de juiste locatie in de dataframe verhoogt met 1. Hierna wordt er een “Total” kolom aangemaakt om het totaal aantal van elke rij (dus hoe vaak de eerste letter met een paar voorkomt), hier wordt gebruik van gemaakt om de aantallen te veranderen naar percentages. Zo wordt een matrix getrained.

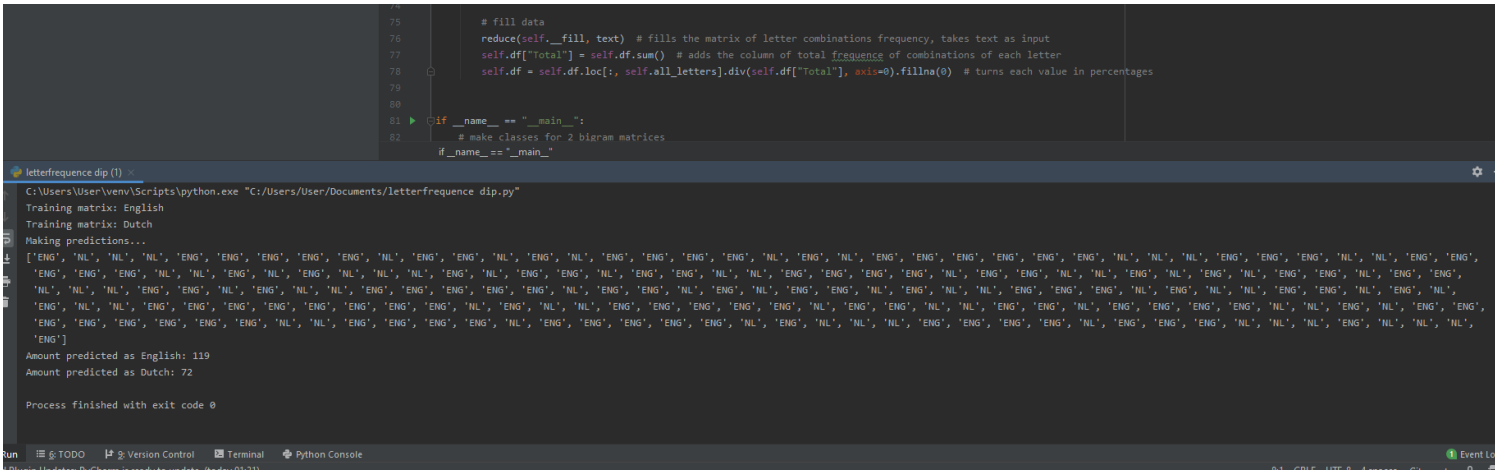
Hierna wordt gebruikt gemaakt van de aangemaakte predict class. Hier wordt voor een stuk tekst een prediction gemaakt om te kijken in welke taal de tekst is. Hier wordt dan ook gebruik gemaakt van de getrainde matrices die je moet meegeven. Voor de tekst waar een voorspelling van wordt gemaakt gebeurt precies hetzelfde met het schoonmaken en maken van een matrix en die vullen. Hieruit komt dus een matrix, die dan wordt vergeleken met de getrainde matrices. We kijken hoe vaak een lettercombinatie dichterbij de ene of de andere matrix in de buurt zit en de matrix die het vaakst in de buurt zit wordt van zijn taal dan als de juiste voorspelling gezien. Hierna maken we de matrix leeg voor de volgende stuk tekst.

Na dat elke stuk tekst een voorspelling heeft gekregen (dat overigens ook met een Map functie werd gedaan) krijgen we dus een lijst met hoe vaak de ene en de andere voorspelling is gedaan, dit wordt opgetelt.


with open('verhaal.txt', 'r') as file:
text = file.read().replace('\n', ' ').lower()
NLmatrix.train_matrix(text)
# make predictions on the test phrases
with open('testzinnen.txt', 'r') as file:
testdata = file.read().lower().splitlines()
print("Making predictions...")
predicter = language_predicter([ENmatrix, NLmatrix])
result = list(map(predicter.predict, testdata))
# results
print("Amount predicted as English: {}".format(result.count("ENG")))
print("Amount predicted as Dutch: {}".format(result.count("NL")))
letterfrequence dip (1)
C:\Users\User\venv\Scripts\python.exe "C:/Users/User/Documents/letterfrequence dip.py"
Training matrix: English
Training matrix: Dutch
Making predictions...
Amount predicted as English: 119
Amount predicted as Dutch: 72
Process finished with exit code 0
4: Run
6: TODO
9: Version Control
Terminal
Python Console
and Plugin Updates: PyCharm is ready to update. (today 01:31)
Type here to search


File Edit View Navigate Code Refactor Run Tools VCS Window Help
GitHub [C:\Users\User\Documents\GitHub] - C:\Users\User\Documents\letterfrequ
C: Users User Documents letterfrequence dip.py
Project
GitHub C:\Users\User\Documents\GitHub
External Libraries
Scratches and Consoles
letterfrequence dip.py
text = ' '.join(text.split()) # removes co
#fill data
reduce(self.__fill, text) # fills the matr
self.df["Total"] = self.df.sum() # adds th
self.df = self.df.loc[:, self.all_letters].
with pd.option_context('display.max_rows',
None): # more optio
trainedmatrix train_matrix() with pd.option_context('displa
Run: letterfrequence dip (1)
C:\Users\User\venv\Scripts\python.exe "C:/Users/User/Documents/letterfrequence dip.py"
Training matrix: English
a b c d e f g \
a 0.000000 0.029446 0.019242 0.044898 0.000000 0.007872 0.015452
b 0.051565 0.079190 0.000000 0.000000 0.270718 0.000000 0.000000
c 0.147059 0.000000 0.005252 0.000000 0.323529 0.000000 0.000000
d 0.010304 0.001085 0.000000 0.017354 0.100868 0.001085 0.011388
e 0.054631 0.002937 0.009790 0.068729 0.033483 0.005091 0.007441
f 0.059202 0.000000 0.000000 0.000000 0.068211 0.059202 0.000000
g 0.087866 0.000000 0.000000 0.002092 0.126569 0.001046 0.010460
h 0.135474 0.000738 0.000000 0.000000 0.549280 0.000738 0.000000
i 0.006367 0.003184 0.093385 0.081005 0.024761 0.014149 0.028652
j 0.067797 0.000000 0.000000 0.000000 0.203390 0.000000 0.000000
k 0.014184 0.000000 0.000000 0.000000 0.368794 0.000000 0.000000
l 0.075000 0.000543 0.000000 0.052174 0.156522 0.029348 0.000543
m 0.138677 0.027990 0.001272 0.000000 0.296438 0.003817 0.000000
n 0.014745 0.002647 0.021172 0.198488 0.077127 0.003025 0.157278
o 0.002286 0.000980 0.005552 0.013063 0.003266 0.082626 0.006205
p 0.111517 0.003656 0.000000 0.000000 0.191956 0.000000 0.001828
q 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
r 0.070683 0.001898 0.009488 0.036053 0.189279 0.000949 0.019924
s 0.086764 0.000404 0.005650 0.000000 0.116223 0.000000 0.000000
t 0.026551 0.000000 0.004595 0.000000 0.066888 0.000511 0.000511
u 0.007538 0.009213 0.058626 0.023451 0.053601 0.000838 0.041039
v 0.016722 0.000000 0.000000 0.000000 0.785953 0.000000 0.000000
w 0.252988 0.000000 0.000000 0.007968 0.118526 0.002988 0.000996
x 0.108696 0.000000 0.065217 0.000000 0.130435 0.000000 0.000000
y 0.004121 0.005495 0.001374 0.001374 0.038462 0.000000 0.000000
z 0.031250 0.000000 0.000000 0.000000 0.593750 0.000000 0.000000
0.136317 0.032146 0.037335 0.030722 0.013733 0.031231 0.022584
h i j k l m n \
a 0.003499 0.070262 0.001458 0.013120 0.114869 0.025364 0.177259
b 0.000000 0.114180 0.003683 0.000000 0.104972 0.001842 0.000000
c 0.165966 0.006303 0.000000 0.059874 0.014706 0.000000 0.000000
4: Run 6: TODO 9: Version Control Terminal Python Console
IDE and Plugin Updates: PyCharm is ready to update. (today 01:31)
Type here to search


File Edit View Navigate Code Refactor Run Tools VCS Window Help
GitHub [C:\Users\User\Documents\GitHub] - C:\Users\User\Documents\letterfrequence
C: Users User Documents letterfrequence dip.py
Project
GitHub C:\Users\User\Documents\GitHub
External Libraries
Scratches and Consoles
letterfrequence dip.py
text = ' '.join(text.split()) # removes consec
#fill data
reduce(self.__fill, text) # fills the matrix o
self.df["Total"] = self.df.sum() # adds the co
self.df = self.df.loc[:, self.all_letters].div(
with pd.option_context('display.max_rows', None
None): # more options c
trainedmatrix train_matrix() with pd.option_context('display...
Run: letterfrequence dip (1)
t 0.092673 0.000766 0.000000 0.022466 0.026551 0.033444 0.011488
u 0.000000 0.060302 0.000000 0.138191 0.154941 0.170017 0.000000
v 0.093645 0.000000 0.000000 0.000000 0.000000 0.003344 0.003344
w 0.082669 0.000996 0.000000 0.004980 0.004980 0.000000 0.000000
x 0.000000 0.065217 0.000000 0.000000 0.000000 0.347826 0.000000
y 0.200549 0.000000 0.000000 0.009615 0.009615 0.019231 0.000000
z 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.056562 0.020651 0.007121 0.019939 0.093795 0.162767 0.010071
v w x y z
a 0.016035 0.009038 0.000000 0.025073 0.000000 0.085131
b 0.000000 0.000000 0.000000 0.064457 0.000000 0.001842
c 0.000000 0.000000 0.000000 0.001050 0.000000 0.004202
d 0.004338 0.000542 0.000000 0.007050 0.000000 0.634490
e 0.011749 0.003329 0.006070 0.014098 0.001175 0.432348
f 0.000000 0.000000 0.000000 0.001287 0.000000 0.416988
g 0.000000 0.000000 0.000000 0.000000 0.001046 0.401674
h 0.000000 0.000000 0.000000 0.005906 0.000000 0.087117
i 0.011319 0.000000 0.000707 0.000000 0.006013 0.046339
j 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
k 0.000000 0.000000 0.000000 0.009456 0.000000 0.293144
l 0.003804 0.001087 0.000000 0.080435 0.000000 0.133152
m 0.000000 0.001272 0.000000 0.029262 0.000000 0.159033
n 0.002647 0.000378 0.001890 0.012476 0.000000 0.272212
o 0.012084 0.069889 0.002286 0.000327 0.000327 0.152188
p 0.000000 0.001828 0.000000 0.007313 0.000000 0.142596
q 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
r 0.000949 0.001423 0.000000 0.051233 0.000000 0.279886
s 0.000000 0.005246 0.000000 0.001211 0.000000 0.346651
t 0.000255 0.006127 0.000000 0.005872 0.000000 0.294358
u 0.001675 0.000000 0.000000 0.000000 0.002513 0.086265
v 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
w 0.000000 0.003984 0.000000 0.000000 0.000000 0.120518
x 0.000000 0.021739 0.000000 0.000000 0.000000 0.130435
y 0.001374 0.001374 0.000000 0.000000 0.000000 0.673077
4: Run 6: TODO 9: Version Control Terminal Python Console
IDE and Plugin Updates: PyCharm is ready to update. (today 01:31)
Type here to search


fill data
reduce(self.__fill, text) # fills the matrix of letter combinations frequency, takes text as input
self.df["Total"] = self.df.sum() # adds the column of total frequence of combinations of each letter
self.df = self.df.loc[:, self.all_letters].div(self.df["Total"], axis=0).fillna(0) # turns each value in percentages
if __name__ == "__main__":
# make classes for 2 bigram matrices
if __name__ == "__main__"
letterfrequence dip (1)
C:\Users\User\venv\Scripts\python.exe "C:/Users/User/Documents/letterfrequence dip.py"
Training matrix: English
Training matrix: Dutch
Making predictions...
['ENG', 'NL', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'ENG',
'ENG', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'NL', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'NL', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'ENG',
'NL', 'NL', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'NL', 'ENG',
'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'NL', 'ENG', 'ENG',
'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'NL', 'NL', 'NL', 'ENG', 'ENG', 'ENG', 'ENG', 'NL', 'ENG', 'ENG', 'ENG', 'NL', 'NL', 'NL', 'ENG', 'NL', 'NL', 'NL',
'ENG']
Amount predicted as English: 119
Amount predicted as Dutch: 72
Process finished with exit code 0
Run 6: TODO 9: Version Control Terminal Python Console