

פעילות 16 - רשת נוירונים מבוססת Keras לסיווג הודעות טקסט

מקורות:

<https://builtin.com/data-science/how-build-neural-network-keras>

<http://tech.couponall.in/tag/imdb/>

<https://stackoverflow.com/questions/51363709/how-to-predict-sentiment-analysis-using-keras-imdb-dataset>

<https://kite.com/python/docs/nltk.punkt>

בפעילות זה נתרגל את היסודות בסיווג טקסט. נבין כיצד מבצעים עיבוד להודעות טקסט, כיצד מייצגים מילים במכונה לומדת, כדי להכניס את רצף המילים לרשת נוירונים וכיצד מערכת לומדת מבצעת סיווג למשפטים.

כפי שכבר למדנו בפעילויות הקודמות חלק מרכזי בפיתוח מכונה לומדת בכלל ורשת נוירונים מלאכותית בפרט הוא ארגון מערך הנתונים. בפעילות זו נעשה שימוש במערך בשם IMDB הכולל 50000 ביקורות על סרטים באנגלית מתווייגות באופן בינארי לביקורות חיוביות (1 לוגי) וביקורות שליליות (0 לוגי).

כל הביקורות במערך הנתונים IMDB מקודדות כרצף של אינדקסים לפי מילים. כאשר כל מילה מקבלת מספר שלם המייצג את התדירות הכוללת של כל אותה מילה במערך. כלומר המילה שמופיעה הכי הרבה פעמים בכלל מערך הנתונים תקודד למספר השלם 1 המילה שמקודדת כמספר 2 היא במקום השני במספר ההופעות שלה במערך הנתונים וכך הלאה.

מערך הנתונים IMDB מחולק ל-2 קבוצות זהות, 25000 ביקורות מתווייגות לצורך שלב האימון של המכונה ו-25000 ביקורות לשלב הבדיקה של המכונה.

מערך הנתונים נוצר בשנת 2011 על ידי צוות חוקרים מאוניברסיטת סטנפורד.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).

קישור למאמר:

https://ai.stanford.edu/~amaas/papers/wvSent_acl2011.pdf

כדי לעבוד עם מערך הנתונים IMDB נשתמש בפעולה `imdb.load_data` כדי להוריד את קובץ הנתונים ישירות מהאינטרנט. נדגים זאת על ידי הקוד הבא:

```
from keras.datasets import imdb
(train_data, train_lbl), (test_data, test_lbl) = imdb.load_data(num_words=10000)
print(len(train_data), 'train data')
print(len(test_data), 'test data')
```

נקבל את הפלט הבא:

```
25000 train data
25000 test data
```

מפלט התוכנית ניתן ללמוד השפעה `imdb.load_data` הורידה 25000 הודעות אימון ו-25000 הודעות בדיקה.

נבחן כיצד נראית הודעה:

```
from keras.datasets import imdb
(train_data, train_lbl), (test_data, test_lbl) = imdb.load_data(num_words=10000)
print(train_data[123])
```

נקבל את הפלט הבא:

```
[1, 307, 5, 1301, 20, 1026, 2511, 87, 2775, 52, 116, 5, 31, 7, 4, 91, 1220, 102, 13, 28, 110, 11, 6, 137, 13, 115, 219, 141, 35, 221, 956, 54, 13, 16, 11, 2714, 61, 322, 423, 12, 38, 76, 59, 1803, 72, 8, 2, 23, 5, 9, 67, 12, 38, 85, 62, 358, 99]
```

נו טוב, מה כבר אפשר להבין מאוסף המספרים הזה. למעשה ניתן להבין שכל מספר הוא אינדקס של מילה בתוך ההודעה. ככל שהמספר קטן יותר המילה שכיחה יותר. ככל שהאינדקס גדול המילה נדירה יותר.

נבחן את 100 המילים השכיחות במערך:

```
from keras.datasets import imdb
from heapq import nsmallest

index = imdb.get_word_index()
index = {k:(v+3) for k,v in index.items()}
index["<PAD>"] = 0
index["<START>"] = 1
index["<UNK>"] = 2

MostPopular = nsmallest(100, index, key = index.get)
for val in MostPopular:
    print(val, ":", index.get(val))
```

נקבל את הפלט הבא:

```

<PAD> : 0    you : 25    there : 50   we : 75
<START> : 1  are : 26    what : 51   much : 76
<UNK> : 2    his : 27    good : 52   been : 77
the : 4      have : 28   more : 53   bad : 78
and : 5      he : 29     when : 54   get : 79
a : 6        be : 30     very : 55   will : 80
of : 7       one : 31    up : 56     do : 81
to : 8       all : 32    no : 57     also : 82
is : 9       at : 33     time : 58   into : 83
br : 10      by : 34     she : 59    people : 84
in : 11      an : 35     even : 60   other : 85
it : 12      they : 36   my : 61     first : 86
i : 13       who : 37    would : 62  great : 87
this : 14    so : 38     which : 63  because : 88
that : 15    from : 39   only : 64   how : 89
was : 16     like : 40   story : 65  him : 90
as : 17      her : 41    really : 66 most : 91
for : 18     or : 42     see : 67    don't : 92
with : 19    just : 43   their : 68  made : 93
movie : 20   about : 44  had : 69    its : 94
but : 21     it's : 45   can : 70    then : 95
film : 22    out : 46    were : 71   way : 96
on : 23      has : 47    me : 72     make : 97
not : 24     if : 48     well : 73   them : 98
you : 25     some : 49   than : 74   too : 99

```

נבחן את 10 המילים הנדירות במערך:

```

from keras.datasets import imdb
from heapq import nlargest

index = imdb.get_word_index()
index = {k:(v+3) for k,v in index.items()}
index["<PAD>"] = 0
index["<START>"] = 1
index["<UNK>"] = 2

RareWords = nlargest(10, index, key = index.get)
for val in RareWords:
    print(val, ":", index.get(val))

```

נקבל את הפלט הבא:

```

'l' : 88587
voorhees' : 88586
artbox : 88585
copywrite : 88584
pipe's : 88583
wheelers : 88582
sics : 88581
transacting : 88580
chicatillo : 88579
ev : 88578

```

להלן קוד המתרגם הודעת טקסט כך שניתן יהיה לקרוא אותה:

```
from keras.datasets import imdb

(train_data, train_lbl), (test_data, test_lbl) = imdb.load_data(num_words=10000)

index = imdb.get_word_index()
index = {k:(v+3) for k,v in index.items()}
index["<PAD>"] = 0
index["<START>"] = 1
index["<UNK>"] = 2

id_to_word = {value:key for key,value in index.items()}
print(" ".join(id_to_word[id] for id in train_data[123] ))
```

נקבל את הפלט הבא:

```
<START> beautiful and touching movie rich colors great settings good acting and one of the most charming
movies i have seen in a while i never saw such an interesting setting when i was in china my wife liked i
t so much she asked me to <UNK> on and rate it so other would enjoy too
```

התג <UNK> מציין שהמילה הנ"ל לא נמצאת באחד מ-10000 המילים הפופולריות ועל פי קוד התוכנית שכתבנו ((imdb.load_data(num_words=10000)) היא לא חלק ממאגר המילים שירד.

נבחן את פלט התוכנית על ידי מבט על המילה השלישית and. האינדקס של המילה הוא 5 ואכן במקום זה קיימת המילה and.

מכונה לומדת מקבלת מערך אינדקסים בגודל קבוע שבה כל מיקום (אינדקס של האיבר) מיוצג כ-1 אם המילה קיימת ו-0 אם המילה לא קיימת במשפט.

להלן קטע קוד הממיר מערך מספרים המייצגים מילים במפשט למערך אינדקסים בגודל קבוע.

לצורך הדגמה פלט התוכנית יהיה מערך אינדקסים בגודל של 40 מקומות. כלומר רק 40 המילים הפופולריות ביותר יקבלו ערך

```
import numpy as np
from keras.datasets import imdb
from nltk import word_tokenize
import string

def Preparing_string(text_string, dimension = 40):
    text_string = text_string.lower()
```

```

table = str.maketrans(dict.fromkeys(string.punctuation))
text_string = text_string.translate(table)
word2index = imdb.get_word_index()
test=[]
for word in word_tokenize(text_string):
    test.append(word2index[word])
print(text_string)
print(test)
out = np.zeros(dimension)
for _ , sequence in enumerate(test):
    if sequence < dimension:
        out[sequence] = 1
print ("\nOutput:",out)

```

Preparing_string("First off, this is NOT a war film. It is a movie about the bond of men in war. It is by far the best movie I've seen in a very, very long time. I had high expectations and was not disappointed. At first I was eager to see the one shot idea Sam Mendes went into this with but, after awhile, I stopped paying attention to that. While everything about the movie was well done I was so caught up in the two central characters that nothing else mattered. I will watch this again and again.")

נקבל את הפלט הבא:

```

first off this is not a war film it is a movie about the bond of men in war it i
ted at first i was eager to see the one shot idea sam mendes went into this with
as so caught up in the two central characters that nothing else mattered i will

[83, 122, 11, 6, 21, 3, 322, 19, 9, 6, 3, 17, 41, 1, 1645, 4, 346, 8, 322, 9, 6,
13, 4508, 5, 64, 1, 28, 321, 323, 1281, 11748, 432, 80, 11, 16, 18, 100, 5233,
2, 12, 161, 331, 12943, 10, 77, 103, 11, 171, 2, 171]

Output: [0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 0. 1. 0. 0.
0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

```

את המכונה שלנו נבנה כך שתקבל מערך אינדקסים (אפסים ואחדים) בגודל של 10000 ולא 40 כמו בדוגמה. משמעות הדבר היא שמצד אחד המערכת מסוגלת לקבל מגוון של עד 10000 מילים במשפט. מצד שני נקבל מכונה לומדת הכוללת מעל חצי מיליון קשרים בין הנירונים.

להלן קוד המכונה הלומדת:

```

import numpy as np
from keras import models
from keras import layers

```

```

from keras.datasets import imdb
TOP_WORDS = 10000

def Convert_to_vectors(data_list, dimension = TOP_WORDS):
    out = np.zeros((len(data_list), dimension))
    for i, sequence in enumerate(data_list):
        out[i, sequence] = 1
    return out

(train_data, train_lbl), (test_data, test_lbl) = imdb.load_data(num_words=TOP_WORDS)
train_data = Convert_to_vectors(train_data)
test_data = Convert_to_vectors(test_data)

model = models.Sequential()
model.add(layers.Dense(50, activation = "relu", input_shape=(TOP_WORDS, )))
model.add(layers.Dropout(0.3, noise_shape=None, seed=None))
model.add(layers.Dense(50, activation = "relu"))
model.add(layers.Dropout(0.2, noise_shape=None, seed=None))
model.add(layers.Dense(50, activation = "relu"))
model.add(layers.Dense(1, activation = "sigmoid"))
model.summary()

model.compile(
    optimizer = "adam",
    loss = "binary_crossentropy",
    metrics = ["accuracy"]
)

results = model.fit(
    train_data, train_lbl,
    epochs= 5,
    batch_size = 10,
    validation_data = (test_data, test_lbl)
)

```

)

```
print("Test-loss:", results.history["loss"])  
print("Test-Accuracy:", results.history["accuracy"])
```

נקבל את הפלט הבא:

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 50)	500050
dropout_1 (Dropout)	(None, 50)	0
dense_2 (Dense)	(None, 50)	2550
dropout_2 (Dropout)	(None, 50)	0
dense_3 (Dense)	(None, 50)	2550
dense_4 (Dense)	(None, 1)	51
Total params: 505,201		

ניתן ללמוד שהמכונה כוללת 10000 מבואות, מבוא אחד לכל מילה, 2 שכבות פנימיות הכוללות 50 נירונים כל אחת ומוצא אחד. סה"כ מעל חמישים מיליון קשרים בין הנירונים. הפעולה model.fit מאמנת את המכונה על ידי 25000 נתונים ב-5 סבבי אימון.

```
Train on 25000 samples, validate on 25000 samples  
Epoch 1/5  
25000/25000 [=====] - 47s 2ms/step - loss: 0.3377 - accuracy: 0.8561 - val_loss: 0.2822 - val_accuracy: 0.8828  
Epoch 2/5  
25000/25000 [=====] - 48s 2ms/step - loss: 0.2002 - accuracy: 0.9206 - val_loss: 0.2998 - val_accuracy: 0.8761  
Epoch 3/5  
25000/25000 [=====] - 50s 2ms/step - loss: 0.1337 - accuracy: 0.9485 - val_loss: 0.3519 - val_accuracy: 0.8706  
Epoch 4/5  
25000/25000 [=====] - 46s 2ms/step - loss: 0.0929 - accuracy: 0.9666 - val_loss: 0.4625 - val_accuracy: 0.8728  
Epoch 5/5  
25000/25000 [=====] - 45s 2ms/step - loss: 0.0680 - accuracy: 0.9763 - val_loss: 0.5331 - val_accuracy: 0.8687  
Test-loss: [0.3376910100914839, 0.20015975978411735, 0.1337492174546933, 0.0929183293148737, 0.06798984598161041]  
Test-Accuracy: [0.85608, 0.92064, 0.94852, 0.9666, 0.97632]
```

לאחר שלב האימון נקבל מערכת הכוללת דיוק של 97.6 אחוז.

כדי לבדוק את המערכת עם נתונים עתידיים, כלומר האפשרות לכתוב ביקורת חדשה ולבדוק אותה על המכונה, נכתוב את הפעולה הבא:

```
def Preparing_string(text_string, dimension = 1000):  
    text_string = text_string.lower()  
    table = str.maketrans(dict.fromkeys(string.punctuation))  
    text_string = text_string.translate(table)  
    word2index = imdb.get_word_index()  
    test=[]  
    for word in word_tokenize(text_string):  
        test.append(word2index[word])  
    out = np.zeros(dimension)
```

```
for _ , sequence in enumerate(test):
```

```
    if sequence < dimension:
```

```
        out[sequence] = 1
```

```
return out
```

הפעולה `Preparing_string` מקבלת כקלט מחרוזת טקסט. הפעולה מבצעת את הפעולות הבאות:

1. פעולה שעוברת על המילים של מחרוזת וממירה את כל האותיות לקטנות.
2. הסרת כל סימני הפיסוק.
3. המרת המילים למספר האינדקס של כל מילה (במידה והאינדקס של המילה גדול מ-`dimension` ערך המילה יהיה 1) כלומר מילים נדירות לא יכנסו למכונה.
4. המרת המספרים לאינדקס המיקום של כל מילה. (במדומה למערך אינדקסים)

נדגים את השלבים תוך שימוש בנתונים הבאים:

ניקח ביקורת מאתר:

https://www.imdb.com/title/tt8579674/reviews?ref_=tt_urv

★ 10/10

Not a war film
goatrope67 15 January 2020

First off, this is NOT a war film. It is a movie about the bond of men in war. It is by far the best movie I've seen in a very, very long time. I had high expectations and was not disappointed. At first I was eager to see the one shot idea Sam Mendes went into this with but, after awhile, I stopped paying attention to that. While everything about the movie was well done I was so caught up in the two central characters that nothing else mattered. I will watch this again and again.



1917 (2019)
User Reviews

+ Review this title

המחרוזת המקורית:

First off, this is NOT a war film. It is a movie about the bond of men in war. It is by far the best movie I've seen in a very, very long time. I had high expectations and was not disappointed. At first I was eager to see the one shot idea Sam Mendes went into this with but, after awhile, I stopped paying attention to that. While everything about the movie was well done I was so caught up in the two central characters that nothing else mattered. I will watch this again and again.

המחרוזת לאחר המרה לאותיות קטנות והסרת סימני הפיסוק:

first off this is not a war film it is a movie about the bond of men in war it is by far the best movie ive seen in a very very long time i had high expectations and was not disappointed at first i was eager to see the one shot idea sam mendes went into this with but after awhile i stopped paying attention to that while everything about the movie was well done i was so caught up in the two central characters that nothing else mattered i will watch this again and again

המרת מילים למספרים:

[83, 122, 11, 6, 21, 3, 322, 19, 9, 6, 3, 17, 41, 1, 1645, 4, 346, 8, 322, 9, 6, 31, 227, 1, 115, 17, 18778, 107, 8, 3, 52, 52, 193, 55, 10, 66, 309, 1395, 2, 13, 21, 682, 30, 83, 10, 13, 4508, 5, 64, 1, 28, 321, 323, 1281, 11748, 432, 80, 11, 16, 18, 100, 5233, 10, 2227, 2645, 689, 5, 212

12, 134, 282, 41, 1, 17, 13, 70, 221, 10, 13, 35, 1056, 53, 8, 1, 104, 1372, 102, 12, 161, 331, 12943, 10, 77, 103, 11, 171, 2, 171]

יצירת מערך אינדקסים:

```
[0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 1.
0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 1. 0. 0.
0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

להלן קוד התוכנית הסופית:

```
import numpy as np
from keras import models
from keras import layers
from keras.datasets import imdb
from nltk import word_tokenize
import string
TOP_WORDS = 10000

def Preparing_string(text_string, dimension = TOP_WORDS):
    text_string = text_string.lower()
    table = str.maketrans(dict.fromkeys(string.punctuation))
    text_string = text_string.translate(table)

    word2index = imdb.get_word_index()
    test=[]
    for word in word_tokenize(text_string):
        test.append(word2index[word])

    results = np.zeros(dimension)
    for _ , sequence in enumerate(test):
        if sequence < dimension:
            results[sequence] = 1

    print("\nOriginal string:", text_string,"\n")
    print("\nIndex conversion:", test,"\n")
```

```

results = np.reshape(results,(1, TOP_WORDS))

print("\nConvert to vectors:", results,"\n")

return results

def vectorize(sequences, dimension = TOP_WORDS):

    results = np.zeros((len(sequences), dimension))

    for i, sequence in enumerate(sequences):

        results[i, sequence] = 1

    return results

(training_data, training_targets), (testing_data, testing_targets) =
imdb.load_data(num_words=TOP_WORDS)

data = np.concatenate((training_data, testing_data), axis=0)

targets = np.concatenate((training_targets, testing_targets), axis=0)

data = vectorize(data)

targets = np.array(targets).astype("float32")

test_x = data[:10000]

test_y = targets[:10000]

train_x = data[10000:]

train_y = targets[10000:]

model = models.Sequential()

model.add(layers.Dense(50, activation = "relu", input_shape=(TOP_WORDS, )))

model.add(layers.Dropout(0.3, noise_shape=None, seed=None))

model.add(layers.Dense(50, activation = "relu"))

model.add(layers.Dropout(0.2, noise_shape=None, seed=None))

model.add(layers.Dense(50, activation = "relu"))

model.add(layers.Dense(1, activation = "sigmoid"))

model.summary()

model.compile(

    optimizer = "adam",

    loss = "binary_crossentropy",

```

```

        metrics = ["accuracy"]
    )

results = model.fit(
    train_x, train_y,
    epochs= 10,
    batch_size = 500,
    validation_data = (test_x, test_y)
)

```

data_string = Preparing_string("First off, this is NOT a war film. It is a movie about the bond of men in war. It is by far the best movie I've seen in a very, very long time. I had high expectations and was not disappointed. At first I was eager to see the one shot idea Sam Mendes went into this with but, after awhile, I stopped paying attention to that. While everything about the movie was well done I was so caught up in the two central characters that nothing else mattered. I will watch this again and again.")

```
print("predict:",model.predict(data_string))
```

```
print("predict_classes:",model.predict_classes(data_string))
```

```
print("-----1 is Good-----\n")
```

data_string = Preparing_string("I don't feel like I know the characters at all. I have no idea why the two soldiers were friends or what they had been through together. The cinematography tried so hard to make this an emotional shocking movie that it had the opposite effect. War scenes with gratuitous up close views of corpses and body parts that don't add anything to the story got old quick.")

```
print("predict:",model.predict(data_string))
```

```
print("predict_classes:",model.predict_classes(data_string))
```

```
print("-----0 is Bad-----\n")
```

פלט התוכנית יהיה (פלט זה מדגים סיווג של 5 ביקורות 3 טובות ו-2 גרועות):

Original string: first off this is not a war film it is a.....

```
predict: [[0.5612222]]
```

```
predict_classes: [[1]]
```

```
-----1 is Good-----
```

Original string: one of the best films

```
predict: [[0.99998033]]
```

```
predict_classes: [[1]]
```

```
-----1 is Good-----
```

Original string: i felt dirty i felt tired i felt hungry

```
predict: [[0.5503042]]
```

```
predict_classes: [[1]]
```

```
-----1 is Good-----
```

Original string: i dont feel like i know the characters at

```
predict: [[0.06769704]]
```

```
predict_classes: [[0]]
```

```
-----0 is Bad-----
```

Original string: predictable and horrendous the acting was terrible

```
predict: [[0.21110523]]
```

```
predict_classes: [[0]]
```

```
-----0 is Bad-----
```

קיבלנו התאמה מלאה בין תוכן הביקורת לבין יכולת המכונה לזהות אם מדובר בביקורת חיובית או שלילית.

תנאי השימוש

תנאי השימוש במסמך זה הם לפי הסטנדרט הבא:

You are free:

to Share – to copy, distribute and transmit the material
to Remix – to adapt the material

Under the following conditions:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for commercial purposes.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.