

Experiment No.13

Aim- IMPLEMENTATION OF NLP PROBLEM.

Algorithm-

Natural language processing (NLP) is a branch of artificial intelligence within computer science that focuses on helping computers to understand the way that humans write and speak. This is a difficult task because it involves a lot of unstructured data.

Natural Language ToolKit (NLTK) is a commonly used NLP library in python to analyze textual data.

- 1- Tokenization is the process of breaking up strings (i.e sentences) into individual words called tokens.
- 2- Corpus (Latin for “body”) refers to a collection of texts.
- 3- Part-of-speech (POS) tagger is the process of labeling a word in a text as corresponding to a particular pos tag: nouns, verbs, adjectives, adverbs, etc.
- 4- Averaged Perceptron tagger is the average perceptron tagger uses the perceptron algorithm to predict which pos tag is most likely given the word.
- 5- JavaScript Object Notation (JSON) is a syntax for storing and exchanging data.

Project

So in the nltk, there is a package called “twitter samples.” We will use this Twitter Corpus to count the number of nouns and adjectives present in the dataset.

You can find a detailed description from the original source of this project: <https://www.digitalocean.com/community/tutorials/how-to-work-with-language-data-in-python-3-using-the-natural-language-toolkit-nltk>

Source code-

```
#Count total number of adjective and noun--
```

```
import nltk
```

```
nltk.download('twitter_samples')
```

```
[nltk_data] Downloading package twitter_samples to /root/nltk_data...
```

```
[nltk_data] Package twitter_samples is already up-to-date!
```

```
True
```

```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
```

```
[nltk_data] /root/nltk_data...
```

```
[nltk_data] Unzipping taggers/averaged_perceptron_tagger.zip.
```

```
True
```

```
nltk.download('brown')
```

```
[nltk_data] Downloading package brown to /root/nltk_data...
```

```
[nltk_data] Unzipping corpora/brown.zip.
```

```
True
```

```
→ #check if nltk is working
```

```
from nltk.corpus import brown
```

```
brown.words()
```

```
#import data and tagger
```

```
from nltk.corpus import twitter_samples
```

```
from nltk.tag import pos_tag_sents
```

```
#tokenize the tweets
```

```
tweet_tokens= twitter_samples.tokenized('positive_tweets.json')
```

```
#tag the tokenized tweets
```

```
tweets_tagged= pos_tag_sents(tweet_tokens)
```

```
#set counters or accumulators
```

```
#JJ- Adjectives; NN- Nouns
```

```
JJ_count=0
```

```
NN_count=0
```

```
#Loop through the tagged tweets
```

```
for tweet in tweets_tagged:
```

```
    for pair in tweet:
```

```
        tag= pair[1]
```

```
        if tag == 'JJ':
```

```
            JJ_count += 1
```



```
        if tag == 'NN':
```

```
            NN_count +=1
```





```
print('Total number of Adjectives= ', JJ_count)
```

```
print('Total number of Nouns= ', NN_count)
```



Output-

 NLP FOR Count total number of adjective and noun .ipynb 

File Edit View Insert Runtime Tools Help [All changes saved](#)



+ Code + Text

 4s 

```
if tag == 'JJ':  
    JJ_count += 1  
if tag == 'NN':  
    NN_count +=1  
print('Total number of Adjectives= ', JJ_count)  
print('Total number of Nouns= ', NN_count)
```

Total number of Adjectives= 6094
Total number of Nouns= 13180

Experiment No.14

Aim- DEEP LEARNING PROJECT IN PYTHON.

Algorithm & Code-

The steps to cover in this are as follows:

1. Load Data.
2. Define Keras Model.
3. Compile Keras Model.
4. Fit Keras Model.
5. Evaluate Keras Model.
6. Tie It All Together.
7. Make Predictions

1- Load Data.

a. Dataset used –

first neural network with keras tutorial

```
from numpy import loadtxt
```

```
from keras.models import Sequential
```

```
from keras.layers import Dense
```

b. Code –

```
# load the dataset
```

```
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=',')
```

```
# split into input (X) and output (y) variables
```

```
X = dataset[:,0:8]
```

```
y = dataset[:,8]
```

2- Define Keras Model.

a. Code –

```
# define the keras model
```

```
model = Sequential()
```

```
model.add(Dense(12, input_dim=8, activation='relu'))
```

```
model.add(Dense(8, activation='relu'))
```

```
model.add(Dense(1, activation='sigmoid'))
```

3- Compile Keras Model.

a. Code –

```
# compile the keras model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

4- Fit Keras Model.

a. Code –

```
# fit the keras model on the dataset
model.fit(X, y, epochs=150, batch_size=10, verbose=0)
```

5- Evaluate Keras Model.

a. Code –

```
# evaluate the keras model
_, accuracy = model.evaluate(X, y, verbose=0)
print('Accuracy: %.2f' % (accuracy*100))
```

6- Tie It All Together Code-

```
# first neural network with keras tutorial
from numpy import loadtxt
from keras.models import Sequential
from keras.layers import Dense

# load the dataset
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=',')
# split into input (X) and output (y) variables
X = dataset[:,0:8]
y = dataset[:,8]

# define the keras model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# compile the keras model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# fit the keras model on the dataset
model.fit(X, y, epochs=150, batch_size=10, verbose=0)

# evaluate the keras model
_, accuracy = model.evaluate(X, y, verbose=0)
print('Accuracy: %.2f' % (accuracy*100))
```

Output –

```
(base) PS C:\Users\Anupriya Johri\desktop\AI project> python keras_first_network.py
2021-05-16 10:32:50.176838: W tensorflow/stream_executor/platform/default/dso_loader.c
or: cudart64_110.dll not found
2021-05-16 10:32:50.176975: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignor
machine.
2021-05-16 10:32:54.055641: W tensorflow/stream_executor/platform/default/dso_loader.c
cuda.dll not found
2021-05-16 10:32:54.055821: W tensorflow/stream_executor/cuda/cuda_driver.cc:326] fail
2021-05-16 10:32:54.553751: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169]
i
2021-05-16 10:32:54.553994: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176]
2021-05-16 10:32:54.666342: I tensorflow/core/platform/cpu_feature_guard.cc:142] This
Library (oneDNN) to use the following CPU instructions in performance-critical operat
To enable them in other operations, rebuild TensorFlow with the appropriate compiler f
2021-05-16 10:33:53.263879: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc
ered 2)
Accuracy: 77.60
(base) PS C:\Users\Anupriya Johri\desktop\AI project>
```

7- Make Predictions

a. Code –

```
# first neural network with keras make predictions
from numpy import loadtxt
from keras.models import Sequential
from keras.layers import Dense
# load the dataset
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=',')
# split into input (X) and output (y) variables
X = dataset[:,0:8] y = dataset[:,8]
# define the keras model model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
# compile the keras model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
# fit the keras model on the dataset
model.fit(X, y, epochs=150, batch_size=10, verbose=0)
# make class predictions with the model
predictions = model.predict_classes(X)
# summarize the first 5 cases
for i in range(5):
print('%s => %d (expected %d)' % (X[i].tolist(), predictions[i], y[i]))
```


b. Output –

```
(base) PS C:\Users\Anupriya Johri\desktop\AI project> python keras_prediction.py
2021-05-16 10:35:34.935972: W tensorflow/stream_executor/platform/default/dso_loader
or: cudart64_110.dll not found
2021-05-16 10:35:34.936108: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ign
machine.
2021-05-16 10:35:37.491066: W tensorflow/stream_executor/platform/default/dso_loader
cuda.dll not found
2021-05-16 10:35:37.491273: W tensorflow/stream_executor/cuda/cuda_driver.cc:326] fa
2021-05-16 10:35:37.495274: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:16
i
2021-05-16 10:35:37.495432: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:17
2021-05-16 10:35:37.496001: I tensorflow/core/platform/cpu_feature_guard.cc:142] Thi
Library (oneDNN) to use the following CPU instructions in performance-critical oper
To enable them in other operations, rebuild TensorFlow with the appropriate compiler
2021-05-16 10:35:38.089867: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.
ered 2)
E:\software\anaconda\lib\site-packages\keras\engine\sequential.py:450: UserWarning:
r 2021-01-01. Please use instead: * np.argmax(model.predict(x), axis=-1)`, if your
tmax` last-layer activation). * `(model.predict(x) > 0.5).astype("int32")`, if your
last-layer activation).
warnings.warn('`model.predict_classes()` is deprecated and '
[6.0, 148.0, 72.0, 35.0, 0.0, 33.6, 0.627, 50.0] => 1 (expected 1)
[1.0, 85.0, 66.0, 29.0, 0.0, 26.6, 0.351, 31.0] => 0 (expected 0)
[8.0, 183.0, 64.0, 0.0, 0.0, 23.3, 0.672, 32.0] => 1 (expected 1)
[1.0, 89.0, 66.0, 23.0, 94.0, 28.1, 0.167, 21.0] => 0 (expected 0)
[0.0, 137.0, 40.0, 35.0, 168.0, 43.1, 2.288, 33.0] => 1 (expected 1)
(base) PS C:\Users\Anupriya Johri\desktop\AI project>
```

Keras Project Summary –

In this project, we discovered how to create our first neural network model using the powerful Keras Python library for deep learning.

Specifically, we learnt the six key steps in using Keras to create a neural network or deep learning model, step-by-step including:

- a. How to load data.
- b. How to define a neural network in Keras.
- c. How to compile a Keras model using the efficient numerical backend.
- d. How to train a model on data.
- e. How to evaluate a model on data.
- f. How to make predictions with the model.

Result - The program executed successfully.