Assignment10_1

November 7, 2021

1 10.1.a

[28]:
```
import string
```

[29]:
```
def tokenize(sentence):
    tokens = []
    # Split the sentence by spaces
    words = sentence.split()
    # convert to lower case
    words = [word.lower() for word in words]
    # Remove punctuation
    table = str.maketrans('', '', string.punctuation)
    tokens = [w.translate(table) for w in words]
    return tokens
```

[30]:
```
sentence = "Punctuation is important, my teacher said. Without punctuation␣
 ↪marks, your writing would be very confusing! "
w_token = tokenize(sentence)
print(type(w_token))
print(w_token)
```

```
<class 'list'>
['punctuation', 'is', 'important', 'my', 'teacher', 'said', 'without',
'punctuation', 'marks', 'your', 'writing', 'would', 'be', 'very', 'confusing']
```

2 10.1.b

[26]:
```
from nltk.util import ngrams
def ngram(tokens, n):
    n_grams = []
    n_grams = ngrams(tokenize(tokens), n) # using the tokenize function from 10.
 ↪1.a
    return [ ' '.join(grams) for grams in n_grams]
```

[31]:
```
sentence = "Punctuation is important, my teacher said. Without punctuation␣
 ↪marks, your writing would be very confusing! "
ng_token = ngram(sentence,3)
```

```
print(type(ng_token))
print(ng_token)
```

```
<class 'list'>
['punctuation is important', 'is important my', 'important my teacher', 'my
teacher said', 'teacher said without', 'said without punctuation', 'without
punctuation marks', 'punctuation marks your', 'marks your writing', 'your
writing would', 'writing would be', 'would be very', 'be very confusing']
```

3 10.1.c

[32]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
```

[36]:
```
def one_hot_encode(tokens, num_words):
    label_encoder = LabelEncoder()
    integer_encoded = label_encoder.fit_transform(tokens)
    print(integer_encoded)
    integer_encoded = integer_encoded.reshape(len(integer_encoded), num_words)
    ### One hot encoding
    onehot_encoder = OneHotEncoder(sparse=False)
    results = onehot_encoder.fit_transform(integer_encoded)
    return results
```

[40]:
```
print ("List of words", w_token)
one_hot_encode(w_token, 1) # w_token is from 10.1.a; it contains list of words
```

```
List of words ['punctuation', 'is', 'important', 'my', 'teacher', 'said',
'without', 'punctuation', 'marks', 'your', 'writing', 'would', 'be', 'very',
'confusing']
[ 6  3  2  5  8  7 10  6  4 13 12 11  0  9  1]
```

[40]:
```
array([[0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 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., 1.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 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., 1., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

[ ]: