Name:Viviyan Richards

Roll no:205229133

EXERCISE-1

1. Import dependencies

In [1]:
```
import gensim
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
from nltk.tokenize import word_tokenize
from sklearn import utils
```

In [2]:
```
data = ["I love machine learning. Its awesome.",
"I love coding in python",
"I love building chatbots",
"they chat amagingly well"]
```

In [3]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Angelan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[3]:
```
True
```

In [4]:
```
tagged_data = [TaggedDocument(words=word_tokenize(d.lower()),
tags=[str(i)]) for i, d in enumerate(data)]
```

In [5]:
```
vec_size = 20
alpha = 0.025
```

In [6]:
```
model = Doc2Vec(vector_size=vec_size,
alpha=alpha,
min_alpha=0.00025,
min_count=1,
dm =1)
# build vocabulary
model.build_vocab(tagged_data)
# shuffle data
tagged_data = utils.shuffle(tagged_data)
# train Doc2Vec model
model.train(tagged_data,
total_examples=model.corpus_count,
epochs=30)
model.save("d2v.model")
print("Model Saved")
```

```
Model Saved
```

In [7]:
```
from gensim.models.doc2vec import Doc2Vec
model= Doc2Vec.load("d2v.model")
#to find the vector of a document which is not in training data
test_data = word_tokenize("I love chatbots".lower())
v1 = model.infer_vector(test_data)
print("V1_infer", v1)
```

```
V1_infer [ 0.0032945   0.0009504   0.01332451  0.01152915  0.01895528  0.023096
65
 -0.00325777 -0.00802977  0.0097452  -0.023578    0.01137165  0.01260952
 -0.00895888  0.00068234 -0.00607778 -0.00854787  0.00213298  0.01996933
  0.00269892  0.00989255]
```

In [8]:
```
similar_doc = model.docvecs.most_similar('1')
print(similar_doc)
```

```
[('0', 0.1854361891746521), ('2', -0.03567519038915634), ('3', -0.0862233042716
98)]
```

In [9]:
```
print(model.docvecs['1'])
```

```
[ 0.00233202 -0.0020763  -0.01821837 -0.02302309  0.00686011  0.01970871
  0.02488494 -0.01114094  0.02446651  0.00846515 -0.00418958 -0.00347237
  0.01749527 -0.02282372 -0.00218709 -0.01023882 -0.01316169  0.02423306
  0.01739944 -0.01872601]
```

In [10]:
```
docs=["the house had a tiny little mouse",
"the cat saw the mouse",
"the mouse ran away from the house",
"the cat finally ate the mouse",
"the end of the mouse story"
]
```

In [11]:
```
tagged_data = [TaggedDocument(words=word_tokenize(d.lower()),
tags=[str(i)]) for i, d in enumerate(docs)]
```

In [16]:
```
vec_size = 20
alpha = 0.025
# create model
model = Doc2Vec(vector_size=vec_size, alpha=alpha, min_alpha=0.00025,min_count=1,
```

In [17]:
```
model.build_vocab(tagged_data)
```

In [18]:
```
tagged_data = utils.shuffle(tagged_data)
```

In [19]:
```
model.train(tagged_data,total_examples=model.corpus_count,epochs=30)
model.save("d2v.model")
print("Model Saved")
```

```
Model Saved
```

In [20]:
```
from gensim.models.doc2vec import Doc2Vec
model= Doc2Vec.load("d2v.model")
```

In [21]:
```
test_data = word_tokenize("cat stayed in the house".lower())
v1 = model.infer_vector(test_data)
print("V1_infer", v1)
```

```
V1_infer [ 0.00522686 -0.02354816 -0.00739392  0.01496425 -0.02397058  0.019779
99
  0.02080073  0.01991114 -0.00643659  0.00969159  0.01156812  0.00580886
 -0.00234774 -0.00706865  0.02135056  0.0247726  -0.00312962  0.02308429
 -0.01610882 -0.01583623]
```

In [22]:
```
similar_doc = model.docvecs.most_similar('2')
print(similar_doc)
```

```
[('0', 0.07634814828634262), ('4', -0.017351791262626648), ('1', -0.02717830240
726471), ('3', -0.48652878403663635)]
```

In [ ]:
```
```