Exercise - 1

(1) Import dependencies

```
import gensim
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
from nltk.tokenize import word_tokenize
from sklearn import utils
data = ["I love machine learning. Its awesome.",
        "I love coding in python",
        "I love building chatbots",
        "they chat amagingly well"]
import nltk
nltk.download('punkt')
tagged_data = [TaggedDocument(words=word_tokenize(d.lower()),
    tags=[str(i)]) for i, d in enumerate(data)]
vec_size = 20
alpha = 0.025
model = Doc2Vec(vector_size=vec_size,
    alpha=alpha
    min_alpha=0.00025,
    min_count=1
    dm=1)
model.build_vocab(tagged_data)
```

Natural Language Processing Lab
Lab4. Computing Document Similarity using Doc2Vec Model

EXERCISE-1

1. Import dependencies

```
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
from nltk.tokenize import word_tokenize
from sklearn import utils
```

2. Create dataset

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

3. Create TaggedDocument

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

4. Train Model

```
# model parameters
vec_size = 20
alpha = 0.025

# create model
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")
```

5. Find Similar documents for the given document

```
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)

# to find most similar doc using tags
```

```
total_examples = model.corpus_count,
    epochs = 20)
model.save("doc.Model")
print("Model Saved")

from gensim.models.doc2vec import Doc2Vec
model = Doc2Vec.load("doc.model")
test_data = word_tokenize("I love chatbots".lower())
v1 = model.infer_vector(test_data)
print("V1_infer", v1)

similar_doc = model.docvecs.most_similar('1')
print(similar_doc)
print(model.docvecs[1])

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"
]
```

```
similar_doc = model.docvecs.most_similar('1')
print(similar_doc)

# to find vector of doc in training data using tags or
# in other words, printing the vector of document at index 1 in training data

print(model.docvecs['1'])
```

EXERCISE-2

Question1. Train the following documents using Doc2Vec model

```
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"
]
```

Question2. Find the most similar TWO documents for the query document *"cat stayed in the house"*.

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

vec_size = 20
,alpha = 0.025
model = Doc2Vec(vector_size=vec_size, alpha=alpha,
                min_alpha=0.00025,
                min_count=1).

model.build_vocab(tagged_data)
tagged_data = utils.shuffle(tagged_data)
```

```
model.train(tagged_data,total_examples=model.corpus_count,epochs=20)
model.save("d2v.model")
print("Model Saved")

from gensim.models.doc2vec import Doc2Vec
model = Doc2Vec.load("d2v.model")
test_data = word_tokenize("I love chocolate".lower())
V1 = model.infer_vector(test_data)
print("V1_infer", V1)
#Similar_doc = model.docvecs.most_similar('1')
print(Similar_doc)
```