```
#5. Implement the Continuous Bag of Words (CBOW) Model. Stages can be:
a. Data preparation
b. Generate training data
c. Train model
d. Output
```

```
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib as mpl
import matplotlib.pylab as pylab
import numpy as np
%matplotlib inline
from nltk.tokenize import sent_tokenize, word_tokenize
import warnings
warnings.filterwarnings(action = 'ignore')
import gensim
from gensim.models import Word2Vec
import re
import bs4 as bs
import urllib.request
import nltk
```

```
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
#a. Data preparation
scrapped_data=urllib.request.urlopen("https://en.wikipedia.org/wiki/Machine_learning")
article=scrapped_data.read()
paresed_article=bs.BeautifulSoup(article,'lxml')
paragraphs=paresed_article.find_all('p')
article_text=""
for p in paragraphs:
  article_text+=p.text
sentences=article_text
```

```
print(article_text)
```

```
earning classifier systems (LCS) are a family of rule-based machine learning algori
nductive logic programming (ILP) is an approach to rule learning using logic progra
nductive logic programming is particularly useful in bioinformatics and natural lar
 machine learning model is a type of mathematical model that, after being "trained"
arious types of models have been used and researched for machine learning systems, 
rtificial neural networks (ANNs), or connectionist systems, are computing systems v
n ANN is a model based on a collection of connected units or nodes called "artifici
he original goal of the ANN approach was to solve problems in the same way that a h
eep learning consists of multiple hidden layers in an artificial neural network. Th
ecision tree learning uses a decision tree as a predictive model to go from observa
upport-vector machines (SVMs), also known as support-vector networks, are a set of 
egression analysis encompasses a large variety of statistical methods to estimate t
 Bayesian network, belief network, or directed acyclic graphical model is a probabi
 Gaussian process is a stochastic process in which every finite collection of the r
iven a set of observed points, or input–output examples, the distribution of the (u
aussian processes are popular surrogate models in Bayesian optimization used to do 
 genetic algorithm (GA) is a search algorithm and heuristic technique that mimics t
he theory of belief functions, also referred to as evidence theory or Dempster–Shaf
ypically, machine learning models require a high quantity of reliable data to perfc
ederated learning is an adapted form of distributed artificial intelligence to trai
here are many applications for machine learning, including:
n 2006, the media-services provider Netflix held the first "Netflix Prize" competit
ecent advancements in machine learning have extended into the field of quantum chem
achine Learning is becoming a useful tool to investigate and predict evacuation dec
lthough machine learning has been transformative in some fields, machine-learning p
he "black box theory" poses another yet significant challenge. Black box refers to 
n 2018, a self-driving car from Uber failed to detect a pedestrian, who was killed 
achine learning has been used as a strategy to update the evidence related to a sys
ifferent machine learning approaches can suffer from different data biases. A machi
anguage models learned from data have been shown to contain human-like biases.[127]
ecause of such challenges, the effective use of machine learning may take longer tc
xplainable AI (XAI), or Interpretable AI, or Explainable Machine Learning (XML), is
ettling on a bad, overly complex theory gerrymandered to fit all the past training 
earners can also disappoint by "learning the wrong lesson". A toy example is that a
dversarial vulnerabilities can also result in nonlinear systems, or from non-patter
esearchers have demonstrated how backdoors can be placed undetectably into classify
lassification of machine learning models can be validated by accuracy estimation te
n addition to overall accuracy, investigators frequently report sensitivity and spe
achine learning poses a host of ethical questions. Systems that are trained on data
hile responsible collection of data and documentation of algorithmic rules used by 
I can be well-equipped to make decisions in technical fields, which rely heavily or
ther forms of ethical challenges, not related to personal biases, are seen in healt
ince the 2010s, advances in both machine learning algorithms and computer hardware 
 physical neural network or Neuromorphic computer  is a type of artificial neural r
mbedded Machine Learning is a sub-field of machine learning, where the machine lear
oftware suites containing a variety of machine learning algorithms include the foll
```

```
sentences="""Alice 23 opened the door and found that it led into a small 90
passage, not much larger than a rat-hole:  she knelt down and
looked along the passage into the loveliest garden you ever saw.
How she longed to get out of that dark hall, and wander about
among those beds of bright flowers and those cool fountains, but
she could not even get her head through the doorway; `and even if
my head would go through,' (thought) $poor Alice, `it would be of
very little use without my shoulders.  Oh, how I wish
I could shut up like a telescope!  I think I could, if I only
```

```
know how to begin.'  For, you see, so many out-of-the-way things
had happened lately, that Alice had begun to think that very few
things indeed were really impossible.


"""


sentences = re.sub('[^A-Za-z0-9]+', ' ', sentences)
sentences = re.sub(r'(?:^| )\w(?:$| )', ' ', sentences).strip()
print(sentences)
```

```
Alice 23 opened the door and found that it led into small 90 passage not much larger
```

```
# remove special characters
sentences = re.sub('[^A-Za-z]+', ' ', sentences)

# remove 1 letter words
sentences = re.sub(r'(?:^| )\w(?:$| )', ' ', sentences).strip()

# lower all characters
sentences = sentences.lower()




print (sentences)
```

```
alice opened the door and found that it led into small passage not much larger than r
```

```
all_sent=nltk.sent_tokenize(sentences)
all_words=[nltk.word_tokenize(sent) for sent in all_sent]


print (sentences)
```

```
alice opened the door and found that it led into small passage not much larger than r
```

```
print (all_words)
```

```
[['alice', 'opened', 'the', 'door', 'and', 'found', 'that', 'it', 'led', 'into', 'sma
```

```
#b. Generate training data
from nltk.corpus import stopwords
for i in range(len(all_words)):
  all_words[i]=[w for w in all_words[i] if w not in stopwords.words('english')]
```

```
print (all_words)
```

```
[['alice', 'opened', 'door', 'found', 'led', 'small', 'passage', 'much', 'larger', 'r
```

```
#c. Train model
data =all_words
```

```
print  (all_words)
```

```
[['alice', 'opened', 'door', 'found', 'led', 'small', 'passage', 'much', 'larger', 'r
```

```
data1=data[0]
```

```
model1 = gensim.models.Word2Vec(data, min_count = 1,vector_size = 52, window = 5)
```

```
vocabulary=list(model1.wv.index_to_key)
print(vocabulary)
```

```
['could', 'alice', 'passage', 'think', 'things', 'even', 'head', 'get', 'would', 'eve
```

```
wrd='door'
#wrd=['subset','machine', 'learning','closely','related']
v1=model1.wv[wrd]
similar_words=model1.wv.most_similar(wrd)
for x in similar_words:
  print(x)
```

```
('beds', 0.36491504311561584)
('much', 0.3305249512195587)
('shut', 0.32979297637939453)
('cool', 0.25908932089805603)
('wish', 0.243193581700325)
('oh', 0.24176433682441711)
('begun', 0.2212926298379898)
('begin', 0.17681987583637238)
('loveliest', 0.14279094338417053)
('things', 0.13509944081306458)
```

```
print(data1)
```

```
['alice', 'opened', 'door', 'found', 'led', 'small', 'passage', 'much', 'larger', 'ra
```

```
#print(data)
```

```
dat = []
```

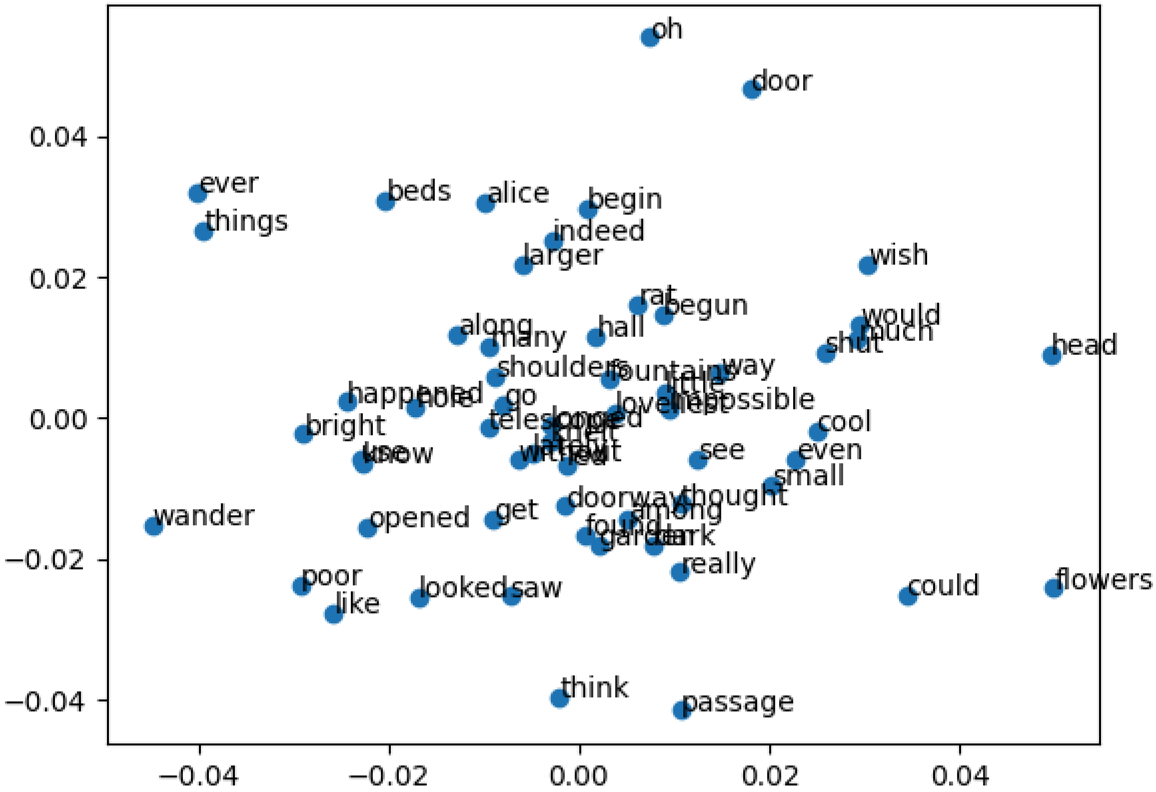
```
for i in range(0, len(data) ):
    context = [data1[i - 2], data1[i - 1], data1[i+1], data1[i + 2]]
    target = data1[i]
    dat.append((context, target))
print(dat[:5])
#print(dat[1][0])
```

```
[(['really', 'impossible', 'opened', 'door'], 'alice')]
```

```
#for val in dat:
 # print(val[0],val[1])
i=0
print(dat[i][0],dat[i][1])
print(model1.predict_output_word(dat[i][0]))
```

```
['really', 'impossible', 'opened', 'door'] alice
[('even', 0.016949415), ('lately', 0.016949397), ('among', 0.016949339), ('saw', 0.01
```

```
#d. Output
from sklearn.decomposition import PCA
from matplotlib import pyplot

# Get the word vectors from the Word2Vec model
X = model1.wv.vectors

# Create a PCA object with 2 components
pca = PCA(n_components=2)

# Fit and transform the word vectors using PCA
result = pca.fit_transform(X)

# Get the list of words corresponding to the word vectors
words = list(model1.wv.index_to_key)

# Create a scatter plot of the reduced word vectors
pyplot.scatter(result[:, 0], result[:, 1])

# Annotate each point with the corresponding word
for i, word in enumerate(words):
    pyplot.annotate(word, xy=(result[i, 0], result[i, 1]))

pyplot.show()
```


oh
door
ever
beds
alice
begin
things
indeed
larger
wish
rat
begun
along
hall
would
many
much
shut
head
shoulders
fountains
way
happened
hole
go
little
loveliest
impossible
bright
telescope
cool
see
even
small
wander
opened
get
doorway
thought
among
found
garden
dark
really
poor
like
looked
saw
could
flowers
think
passage
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04