ICP7

Answer 1)

Program Description: In this program, we have to change the classifier in the given code provided
Source Code:

I) Using SVM to check the accuracy.
2) Changing the tfidf vectorizer to use bigram and check the accuracy changes
TfidfVectorizer(ngram_range=(1,2))
3) Setting argument **stop_words='english'** to see how accuracy changes.

Explanation:

Step 1) Imported the necessary libraries and created our environment.

```
# Imported the necessary libraries and created our environment
# fetch_20newsgroups. Specify a download and cache folder for the datasets
from sklearn.datasets import fetch_20newsgroups
# The sklearn.feature_extraction module can be used to extract features
from sklearn.feature_extraction.text import TfidfVectorizer
# The sklearn. metrics module implements several loss, score, and utility functions to measure
# classification performance.
from sklearn import metrics
# Get Naive Bayes algorithm
from sklearn.naive_bayes import MultinomialNB
# Support Vector Machines (SVM) and Support Vector Classifier (SVC)
from sklearn.svm import SVC
```

Step 2) Use of MultinomialNB to train vector trasformed data and get score.

First Create train dataset from 20newsgroup. The 20newsgroup is standard text classification dataset which is collection of app. 20,000 newsgroups documents.

```
# 20newsgroup is standard text classification dataset which is collection of app. 20,000 newsgroups documents.
# Create train dataset from 20newsgroup
twenty_train = fetch_20newsgroups(subset='train', shuffle=True)
```

Then create TFIDF vector to get weightage of words in input dataset. The TFIDF score increases with every occurrence of word in document.
Use TFIDF vector transform on train dataset.

```
# Create TFIDF vector to get weightage of words in input dataset.
tfidf_Vect = TfidfVectorizer()
# Use TFIDF vector transform on train dataset
X_train_tfidf = tfidf_Vect.fit_transform(twenty_train.data)
# print(tfidf_Vect.vocabulary_)
```

Use MultinomialNB to implement Naive Bayes algorithm and then train the model with TFIDF vector transformed dataset and train datasets target.

```
# Use MultinomialNB to implement Naive Bayes algorithm
clf = MultinomialNB()
# Train model with TFIDF vector from input dataset and train datasets target
clf.fit(X_train_tfidf, twenty_train.target)
```

Next steps are to create test dataset from 20newsgroups. Use TFIDF transform on test dataset and get the predicted values from trained model and test TFIDF vector.

```
# Create test dataset from 20newsgroups
twenty_test = fetch_20newsgroups(subset='test', shuffle=True)
# Use TFIDF transform on test dataset
X_test_tfidf = tfidf_Vect.transform(twenty_test.data)
# get the predicted values from trained model and test TFIDF vector
predicted = clf.predict(X_test_tfidf)
```

Calculate accuracy score by comparing predicted vs actual values.

```
# calculate accuracy score by comparing predicted vs actual values.
nb_score = round(metrics.accuracy_score(twenty_test.target, predicted) * 100, 2, )
```

OUTPUT:

```
Accuracy Score (Naive Bayes):  77.39
```

Step 2) **Use of SVM classifier.**

Let's create **method "get_score"** which will be useful in further part of code as well to avoid redundancy. This method will perform following things.

1) Get TFIDF vector transformed dataset (X_train) and the actual TFIDF vector to be used (tfidf_vect).
2) Create SVC classifier based model and train it using input TFIDF vector transformed train data
3) Create test TFIDF vector transformed data using input TFIDF vector.
4) Get the predicted values based of test TFIDF vector dataset
5) Calculate accuracy score by comparing predicted vs actual values.
6) Return Accuracy Score (score).

```
def get_score(X_train, tfidf_vect):
    clf = SVC()
    clf.fit(X_train, twenty_train.target)
    twenty_test = fetch_20newsgroups(subset='test', categories=categories, shuffle=True)
    X_test_tfidf = tfidf_vect.transform(twenty_test.data)
    predicted = clf.predict(X_test_tfidf)
    score = round(metrics.accuracy_score(twenty_test.target, predicted) * 100, 2, )
    return score
```

Next step is to create TFIDF vector to get weightage of words in input dataset. Use TFIDF transform on train dataset.

```
# Get train data
twenty_train = fetch_20newsgroups(subset='train', categories=categories, shuffle=True)
# Create TFIDF vector to get weightage of words in input dataset.
tfidf_vect1 = TfidfVectorizer()
# Use TFIDF transform on train dataset
X_train_tfidf = tfidf_vect1.fit_transform(twenty_train.data)
```

Invoke **get_score** method to get svc classifier based accuracy score.

```
# Invoke get_score method to get svc classifier based score
score1 = get_score(X_train_tfidf, tfidf_vect1)
print('Accuracy score (SVM): ', score1, '\n')
```

OUTPUT:

```
Accuracy score (SVM):  89.01
```

Step 3) Calculate score using **Bigram**.
ngram is the string of n words in a row.
Here 1 is the minimum and 2 is the maximum sixe of ngram.
Here we will use bigram TFIDF vector i.e. ngram with n = 2. We will transform our train data using this bigram vector.

```
# Use bigram TFIDF vector i.e. ngram with n = 2
tfidf_vect2 = TfidfVectorizer(ngram_range=(1, 2))
# Use bigram TFIDF transform on train dataset
X_train_tfidf2 = tfidf_vect2.fit_transform(twenty_train.data)
```

Invoke get_score method to get svc classifier and Bigram trained model based score

```
# Invoke get_score method to get svc classifier and Bigram trained model based score
score2 = get_score(X_train_tfidf2, tfidf_vect2)
print('Accuracy score (Bigram): ', score2, '\n')
```

OUTPUT:

```
Accuracy score (Bigram):  87.75
```

Step 4) Use of **stop_words** TFIDF vector parameter.
Use stop_words = 'english' to remove less-meaningful english words.
Use stop_words='english' parameter of TFIDF vector. We will transform our train data using this vector.

```
# Use stop_words='english' parameter of TFIDF vector.
tfidf_vect3 = TfidfVectorizer(stop_words='english')
# Use stop_words based TFIDF transform on train dataset
X_train_tfidf3 = tfidf_vect3.fit_transform(twenty_train.data)
```

Invoke get_score method to get svc classifier and stop_words TFIDF trained model based score

```
# Invoke get_score method to get svc classifier and stop_words TFIDF trained model based score
score3 = get_score(X_train_tfidf3, tfidf_vect3)
print('Accuracy score (Stop_words): ', score3, '\n')
```

OUTPUT:

```
Accuracy score (Stop_words):  90.68
```

Step 5) Score Comparison:
We can see from output that SVM algorithm gives much better accuracy score i.e. good prediction than Naive Bayes algorithm.
Also use of Stop_words parameter for TFIDF vector gave much better result than using Bigram and vector with default parameters.
In last score does not improved much when we used bigram for our input data.

```
Accuracy Score (Naive Bayes):  77.39

Accuracy score (SVM):  89.01

Accuracy score (Bigram):  87.75

Accuracy score (Stop_words):  90.68
```

Answer 2)

Program Description: A program to extract the following web URL text using BeautifulSoup
https://en.wikipedia.org/wiki/Google
Output will be saved to input.txt

Explanation:

Step 1) Import the essential libraries:
Import the BeautifulSoup class creator from the package bs4.
bs4(beautiful soup) for parsing the HTML page content.

```
# Imported the essential libraries and created our environment
# library to fetch the page content
import requests
# Import the BeautifulSoup class creator from the package bs4.
# bs4(beautiful soup) for parsing the HTML page content.
from bs4 import BeautifulSoup
```

Step 2) Get data from URL:
Request the server the content of the web page by using get(), and store the server's response in the variable response.

```
# Given url
url = 'https://en.wikipedia.org/wiki/Google'
#  res for inspecting the results of the request
res = requests.get(url)
html_page = res.content
```

Step 3) Parsing the html page:
The **'html.parser'** argument indicates that we want to do the parsing using Python's built-in HTML parser.

```
soup = BeautifulSoup(html_page, 'html.parser')
```

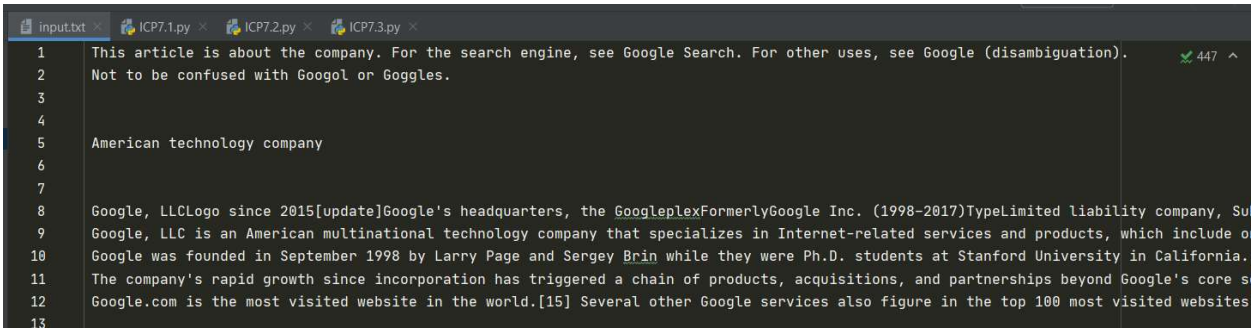
Step 4) Use of find method to extract all the div container that have a class attribute of mw-parser-output.

```
data = soup.find('div', {'class': 'mw-parser-output'})
print(data.text)
```

Step 5) Writing the parsed and filtered data to output file 'Input.txt'

```
with open('input.txt', 'w', encoding='utf-8') as f:
    f.write(str(data.text))
```

OUTPUT: File input.txt



Answer 3)

Program Description: A program to apply the following NLP functions on the "input.txt" and showing the output:

1) Tokenization
2) POS
3) Stemming
4) Lemmatization
5) Trigram
6) Named Entity Recognition

Explanation:

Step 1) Import nltk library: NLTK (Natural Language Processing Toolkit) library is required for importing different NLP functions.

```
# Imported the essential libraries and created our environment
# The Natural Language Toolkit (NLTK) is a Python package for natural language processing
import nltk
```

Step 2) Importing Text data.

```
# opening the input.txt in read mode
text = open('input.txt', encoding="utf8").read()
```

Step 3) Tokenization:

Use of Tokenization functions: Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens.

A) Word tokens: This can be done on words or sentences.

i) word_tokenize() for splitting sentences into word tokens.

```
# Tokenization
# word_tokenize() for splitting sentences into word tokens.
wtokens = nltk.word_tokenize(text)
# printing wtokens
print("========================== Word  Tokenization ==============================")
print(wtokens)
```

Output: Each word of all the sentences in the input text file is tokenized.

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
============================ Word  Tokenization ==============================

['This', 'article', 'is', 'about', 'the', 'company', '.', 'For', 'the', 'search', 'engine', ',', 'see', 'Google
 'disambiguation', ')', '.', 'Not', 'to', 'be', 'confused', 'with', 'Googol', 'or', 'Goggles', '.', 'American',
 '[', 'update', ']', 'Google', "'s", 'headquarters', ',', 'the', 'GoogleplexFormerlyGoogle', 'Inc.', '(', '1998-
 'SubsidiaryIndustryInternetCloud', 'computingComputer', 'softwareComputer', 'hardwareArtificial', 'intelligence
 'ago', '(', '1998-09-04', ')', '[', 'a', ']', 'in', 'Menlo', 'Park', ',', 'California', ',', 'U.S.FoundersLarry
 ',', 'Mountain', 'View', ',', 'California', ',', 'U.S.Area', 'servedWorldwideKey', 'peopleSundar', 'Pichai', '(
```

ii) sent_tokenize for function to tokenize sentences out of paragraph.

```
# sent_tokenize function to tokenize sentences out of paragraph.
stokens = nltk.sent_tokenize(text)
# printing Sentence Tokenization
print("\n============================ Sentence  Tokenization ======================\n")
print(stokens)
```

Output:Each Sentence of input text file is tokenized.

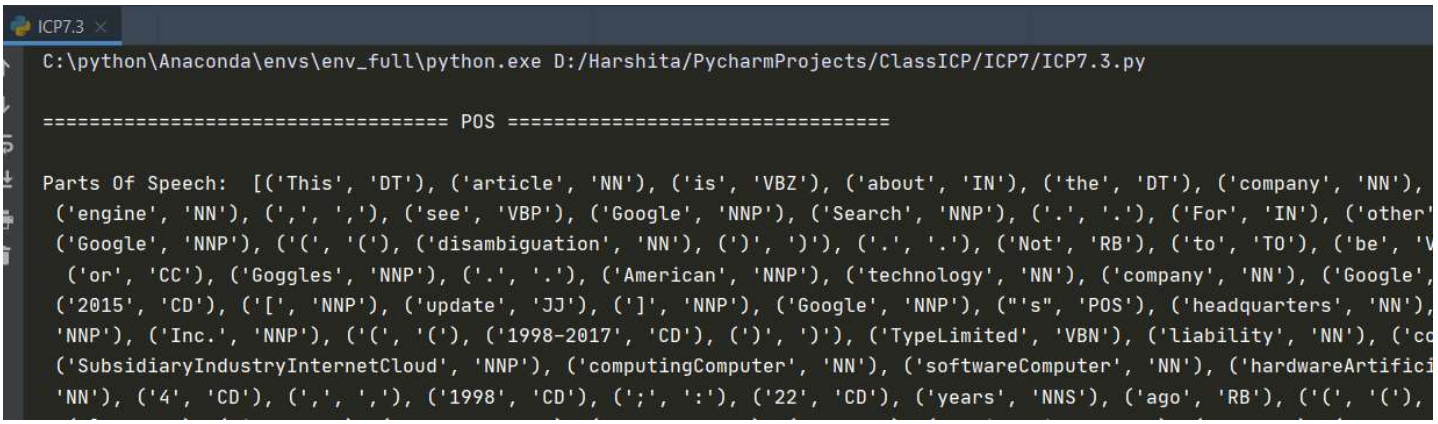
```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py

========================= Sentence  Tokenization ======================

['This article is about the company.', 'For the search engine, see Google Search.', 'For other uses, see Google (disambiguation).', 'N
 "American technology company\n\n\nGoogle, LLCLogo since 2015[update]Google's headquarters, the GoogleplexFormerlyGoogle Inc. (1998-20
 SubsidiaryIndustryInternetCloud computingComputer softwareComputer hardwareArtificial intelligenceAdvertisingFoundedSeptember\xa04, 1
 Menlo Park, California, U.S.FoundersLarry PageSergey BrinHeadquarters1600 Amphitheatre Parkway, Mountain View, California, U.S.Area s
 (CEO)Ruth Porat (CFO)ProductsList of productsRevenue66,001,000,000 US dollar[5] (2014)\xa0Operating income16,496,000,000 US dollar[5]
 States dollar[5] (2014)\xa0Total assets131,133,000,000 US dollar[5] (2014)\xa0Number of employees114,096\xa0(Q3 2019[update])ParentAl
 references[6][7][8][9]\nGoogle, LLC is an American multinational technology company that specializes in Internet-related services and
 technologies, a search engine, cloud computing, software, and hardware.", 'It is considered one of the Big Four technology companies
 alongside Amazon, Apple, and Microsoft.', '[10][11][12]\nGoogle was founded in September 1998 by Larry Page and Sergey Brin while the
 in California.', 'Together they own about 14 percent of its shares and control 56 percent of the stockholder voting power through sup
```

Step 4) Part Of Speech Tagging (POS) : The process of classifying the words in a text(corpus) into their parts of speech and labeling them accordingly is known as part-of-speech tagging, POS-tagging, or simply tagging.

Use nltk._tag for POS Tagging each tokenized word.

```
# POS
# The POS tagger in the NLTK library outputs specific tags for certain words.
n_pos = nltk.word_tokenize(text)
pos_t = nltk.pos_tag(n_pos)
print("\n================================== POS ================================\n", '\n')
print("Parts Of Speech: ", pos_t)
```

Output: Each Tokenized word is POS Tagged. Ex. “This” tagged as DT (Determiner).



Step 5) Stemming: It is process of reducing inflected word to its ‘Root Word’.

Different types of Stemming functions available in nltk.step library.

```
# Stemming
from nltk.stem import PorterStemmer
from nltk.stem import LancasterStemmer
from nltk.stem import SnowballStemmer
```

i) PorterStemmer: Create instance of PorterStemmer class. After that do Stemming on each tokenized word (wtokens) and join them to form sentences.

```
# Create instance of PorterStemmer class
ps = PorterStemmer()
# Do stemming on each tokenized word and join them to form sentences.
stemmed_output = ' '.join([ps.stem(w) for w in wtokens])
print("========== Stemming ==========", '\n')
print(stemmed_output)
```

Output: The inflected words in all the sentences are reduced to their roots. Ex. “disambiguation” reduced to “disambigu”.

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
========== Stemming ==========

thi articl is about the compani . for the search engin , see googl search . for other use , see googl ( disambigu ) .
compani googl , llclogo sinc 2015 [ updat ] googl 's headquart , the googleplexformerlygoogl inc. ( 1998-2017 ) typel
computingcomput softwarecomput hardwareartifici intelligenceadvertisingfoundedseptemb 4 , 1998 ; 22 year ago ( 1998-0
pagesergey brinheadquarters1600 amphitheatr parkway , mountain view , california , u.s.area servedworldwidekey people
productsrevenue66,001,000,000 US dollar [ 5 ] ( 2014 ) oper income16,496,000,000 US dollar [ 5 ] ( 2014 ) net income1
assets131,133,000,000 US dollar [ 5 ] ( 2014 ) number of employees114,096 ( Q3 2019 [ updat ] ) parentalphabet inc.we
llc is an american multin technolog compani that special in internet-rel servic and product , which includ onlin adv
```

ii) LancasterStemmer: Create instance of LancasterStemmer class. After that do Stemming on each tokenized word (wtokens) and join them to form sentences.

```
# Create instance of LancasterStemmer class
ls=LancasterStemmer()
# Do stemming on each tokenized word and join them to form sentences.
lsstemmed_output = ' '.join([ls.stem(w) for w in wtokens])
print("================================== LancasterStemmer ==================================", '\n')
print(lsstemmed_output)
```

OUTPUT: The inflected words in all the sentences are reduced to their roots. Ex. “used” reduced to “us”.

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
================================== LancasterStemmer ==================================

thi artic is about the company . for the search engin , see googl search . for oth us , see googl ( disambigu ) . not to be
  , llclogo sint 2015 [ upd ] googl 's headquart , the googleplexformerlygoogl inc. ( 1998-2017 ) typelimit liabl company ,
 softwarecomput hardwareart intelligenceadvertisingfoundedseptemb 4 , 1998 ; 22 year ago ( 1998-09-04 ) [ a ] in menlo park
 brinheadquarters1600 amphith parkway , mountain view , californ , u.s.area servedworldwidekey peoplesund picha ( ceo ) ruth
 us doll [ 5 ] ( 2014 ) op income16,496,000,000 us doll [ 5 ] ( 2014 ) net income14,444,000,000 unit stat doll [ 5 ] ( 2014 )
 of employees114,096 ( q3 2019 [ upd ] ) parentalphabet inc.websitegoogle.comfootnotes / ref [ 6 ] [ 7 ] [ 8 ] [ 9 ] googl ,
 internet-related serv and produc , which includ onlin advert technolog , a search engin , cloud comput , softw , and hardw
 the u.s. inform technolog industry , alongsid amazon , appl , and microsoft . [ 10 ] [ 11 ] [ 12 ] googl was found in septer
  stud at stanford univers in californ . togeth they own about 14 perc of it shar and control 56 perc of the stockhold vot p
 californ priv held company on septemb 4 , 1998 , in californ . googl was then reincorp in delaw on octob 22 , 2002 . [ 13 ]
  and googl mov to it headquart in mountain view , californ . picknam the googleplex . in august 2015 , googl annount plan
```

iii) SnowballStemmer: Create instance of SnowballStemmer class. After that do Stemming on each tokenized word (wtokens) and join them to form sentences.
Choose the language out of all supported - here ‘english’.

```
# Create instance of SnowballStemmer class
# Choose the language out of all supported - here english
ws=SnowballStemmer("english")
# Do stemming on each tokenized word and join them to form sentences.
wsstemmed_output = ' '.join([ws.stem(w) for w in wtokens])
print("================================== SnowballStemmer ==================================", '\n')
print(wsstemmed_output)
```

OUTPUT: The inflected words in all the sentences are reduced to their roots. Ex. “confused” reduced to “confus”.

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
================================== SnowballStemmer ==================================

this articl is about the compani . for the search engin , see googl search . for other use , see googl ( disambigu ) . not to be confus with
 compani googl , llclogo sinc 2015 [ updat ] googl 's headquart , the googleplexformerlygoogl inc. ( 1998-2017 ) typelimit liabil compani ,
 computingcomput softwarecomput hardwareartifici intelligenceadvertisingfoundedseptemb 4 , 1998 ; 22 year ago ( 1998-09-04 ) [ a ] in menlo
 pagesergey brinheadquarters1600 amphitheatr parkway , mountain view , california , u.s.area servedworldwidekey peoplesundar pichai ( ceo )
 productsrevenue66,001,000,000 us dollar [ 5 ] ( 2014 ) oper income16,496,000,000 us dollar [ 5 ] ( 2014 ) net income14,444,000,000 unit sta
 assets131,133,000,000 us dollar [ 5 ] ( 2014 ) number of employees114,096 ( q3 2019 [ updat ] ) parentalphabet inc.websitegoogle.comfootnot
  llc is an american multin technolog compani that special in internet-rel servic and product , which includ onlin advertis technolog , a se
```

Step 6) Lemmatization: This is also Text normalization technique for reducing inflected words to their root.

Create instance of WordNetLemmatizerclass. After that do Stemming on each tokenized word (wtokens) and join them to form sentences.

```
# Create instance of WordNetLemmatizer class
lemmatizer = WordNetLemmatizer()
# Do stemming on each tokenized word and join them to form sentences.
lemmatized_output = ' '.join([lemmatizer.lemmatize(w) for w in wtokens])
print("================================== Lemmatization ==================================", '\n')
print(lemmatized_output)
```

OUTPUT: The inflected words in all the sentences are reduced to their roots. Ex. “companies” reduced to “company”.

Main Difference we can see between Stemming and Lemmatization is, Stem is not necessarily actual language word but Lemma is.

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
================================== Lemmatization ==================================

This article is about the company . For the search engine , see Google Search . For other us , see Google ( disambiguation ) .
 technology company Google , LLCLogo since 2015 [ update ] Google 's headquarters , the GoogleplexFormerlyGoogle Inc. ( 1998-20
 SubsidiaryIndustryInternetCloud computingComputer softwareComputer hardwareArtificial intelligenceAdvertisingFoundedSeptember
 Park , California , U.S.FoundersLarry PageSergey BrinHeadquarters1600 Amphitheatre Parkway , Mountain View , California , U.S.
 Ruth Porat ( CFO ) ProductsList of productsRevenue66,001,000,000 US dollar [ 5 ] ( 2014 ) Operating income16,496,000,000 US do
 States dollar [ 5 ] ( 2014 ) Total assets131,133,000,000 US dollar [ 5 ] ( 2014 ) Number of employees114,096 ( Q3 2019 [ updat
  reference [ 6 ] [ 7 ] [ 8 ] [ 9 ] Google , LLC is an American multinational technology company that specializes in Internet-r
 advertising technology , a search engine , cloud computing , software , and hardware . It is considered one of the Big Four te
 industry , alongside Amazon , Apple , and Microsoft . [ 10 ] [ 11 ] [ 12 ] Google wa founded in September 1998 by Larry Page a
 Stanford University in California . Together they own about 14 percent of it share and control 56 percent of the stockholder v
```

Step7) Trigram: This is n-gram of 3 items. i.e different combinations of 3 consecutive words in input text. Import ngrams from nltk. Use ngram on word tokens with n = 3.

```
# Trigram
# Import ngrams from nltk.
from nltk.util import ngrams
# Use n-gram for n = 3
trigrams = ngrams(wtokens,3)
print("================================== Trigrams ==================================",'\n')
print(list(trigrams))
```

OUTPUT: As we can see in output, ngram created different combinations of 3 consecutive words using given input work tokens (wtoken).

```
C:\python\Anaconda\envs\env_full\python.exe D:/Harshita/PycharmProjects/ClassICP/ICP7/ICP7.3.py
=================================== Trigrams ==========================================

[('This', 'article', 'is'), ('article', 'is', 'about'), ('is', 'about', 'the'), ('about', 'the', 'company'),
 ('For', 'the', 'search'), ('the', 'search', 'engine'), ('search', 'engine', ','), ('engine', ',', 'see'), (',
  '.'), ('Search', '.', 'For'), ('.', 'For', 'other'), ('For', 'other', 'uses'), ('other', 'uses', ','), ('use
 ('Google', '(', 'disambiguation'), ('(', 'disambiguation', ')'), ('disambiguation', ')', '.'), (')', '.', 'No
 , ('be', 'confused', 'with'), ('confused', 'with', 'Googol'), ('with', 'Googol', 'or'), ('Googol', 'or', 'Goo
 'American', 'technology'), ('American', 'technology', 'company'), ('technology', 'company', 'Google'), ('comp
 'since'), ('LLCLogo', 'since', '2015'), ('since', '2015', '['), ('2015', '[', 'update'), ('[', 'update', ']'
 'headquarters'), ("'s", 'headquarters', ','), ('headquarters', ',', 'the'), (',', 'the', 'GoogleplexFormerlyG
 ('GoogleplexFormerlyGoogle', 'Inc.', '('), ('Inc.', '(', '1998–2017'), ('(', '1998–2017', ')'), ('1998–2017',
 ('TypeLimited', 'liability', 'company'), ('liability', 'company', ','), ('company', ',', 'SubsidiaryIndustryI
```

Step 8) Named Entity Recognition: NER is used for categorizing input text into Person, Organization, Location etc.
Import ne_chunk for getting Named Entity Recognition.
Use ne_chunk on POS tags created in above part of program.

```
# Named Entity Recognition
# Import ne_chunk for getting Named Entity Recognition
from nltk import ne_chunk
#Use ne_chunk on POS tags created above
noe = ne_chunk(pos_t)
print("\nNamed Entity Recognition :", noe)
```

OUTPUT: As we can see in output, the input text is categorized based on Person, Organization, Location etc.

```
(ORGANIZATION Stanford/NNP University/NNP)
in/IN
(GPE California/NNP)
by/IN
(PERSON Larry/NNP Page/NNP)
and/CC
(PERSON Sergey/NNP Brin/NNP)
```