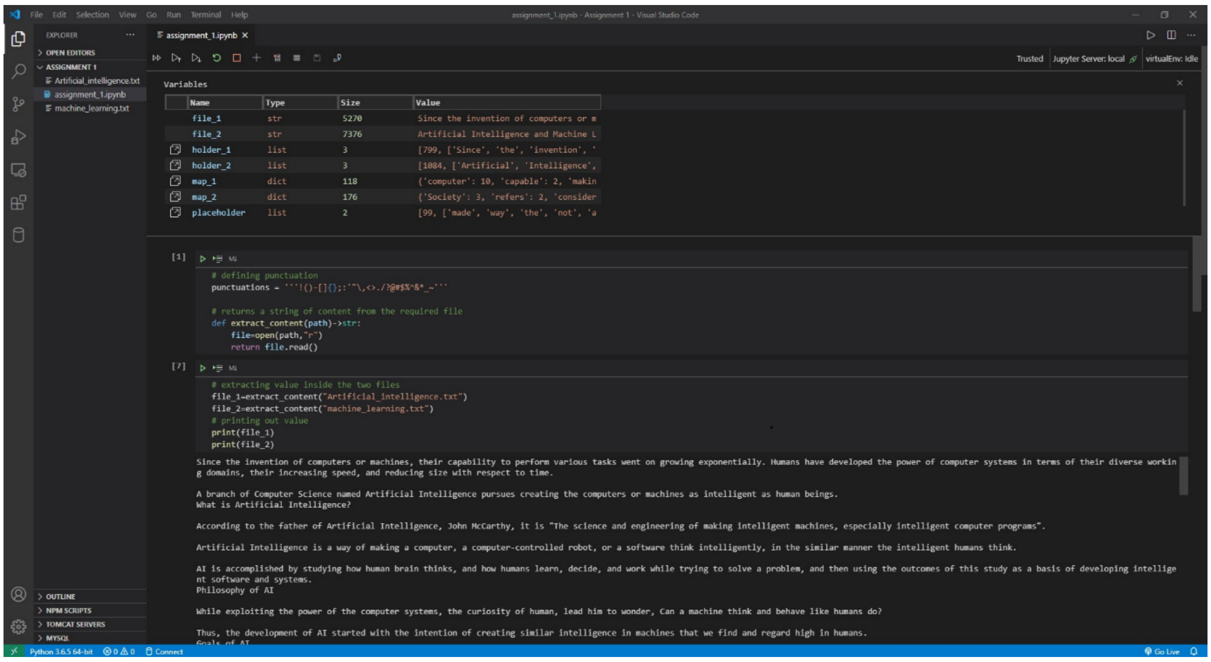
# Web mining Assignment 1

19BCE2311 Gaurav Singh

## Screenshots



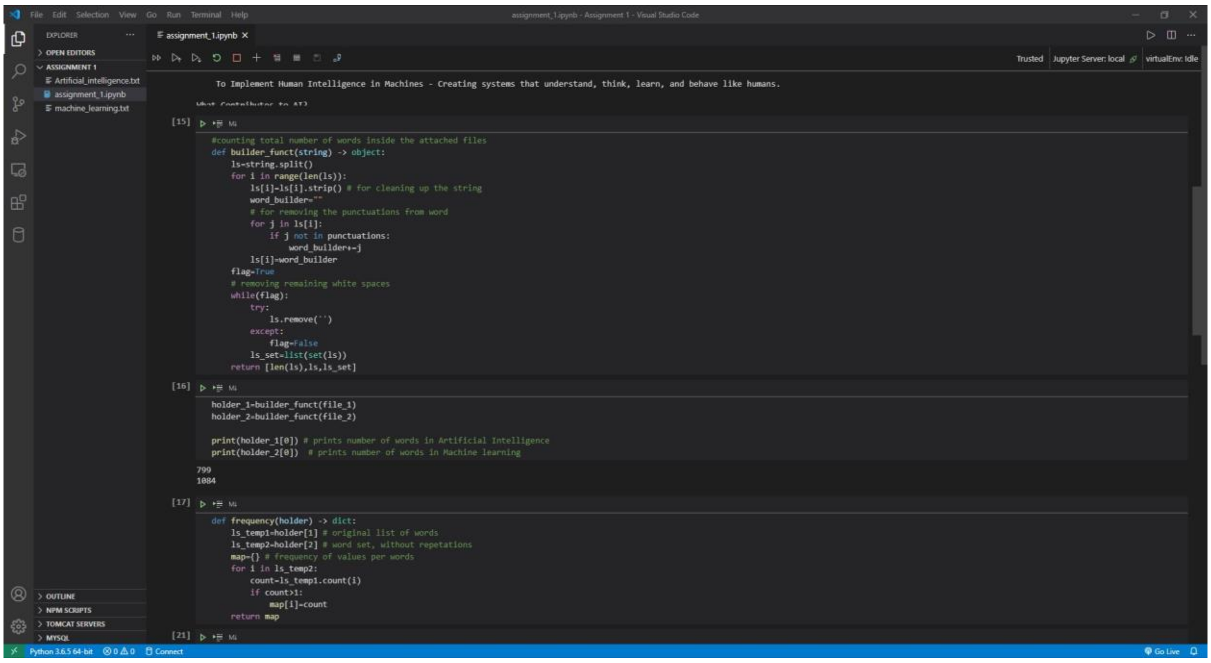
Variables

| Name | Type | Size | Value |
|---|---|---|---|
| file_1 | str | 5270 | Since the invention of computers or m |
| file_2 | str | 7376 | Artificial Intelligence and Machine L |
| holder_1 | list | 3 | [799, ['Since', 'the', 'invention', ' |
| holder_2 | list | 3 | [1084, ['Artificial', 'Intelligence', |
| map_1 | dict | 118 | {'computer': 10, 'capable': 2, 'makin |
| map_2 | dict | 176 | {'Society': 3, 'refers': 2, 'consider |
| placeholder | list | 2 | [99, ['made', 'way', 'the', 'not', 'a |

```
[1]
# defining punctuation
punctuations = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''

# returns a string of content from the required file
def extract_content(path)->str:
    file=open(path,"r")
    return file.read()
```

```
[7]
# extracting value inside the two files
file_1=extract_content("Artificial_intelligence.txt")
file_2=extract_content("machine_learning.txt")
# printing out value
print(file_1)
print(file_2)
```

```
Since the invention of computers or machines, their capability to perform various tasks went on growing exponentially. Humans have developed the power of computer systems in terms of their diverse workin
g domains, their increasing speed, and reducing size with respect to time.

A branch of Computer Science named Artificial Intelligence pursues creating the computers or machines as intelligent as human beings.
What is Artificial Intelligence?

According to the father of Artificial Intelligence, John McCarthy, it is "The science and engineering of making intelligent machines, especially intelligent computer programs".

Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intellige
nt software and systems.
Philosophy of AI

While exploiting the power of the computer systems, the curiosity of human, lead him to wonder, Can a machine think and behave like humans do?

Thus, the development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.
```



```
    To Implement Human Intelligence in Machines - Creating systems that understand, think, learn, and behave like humans.
```

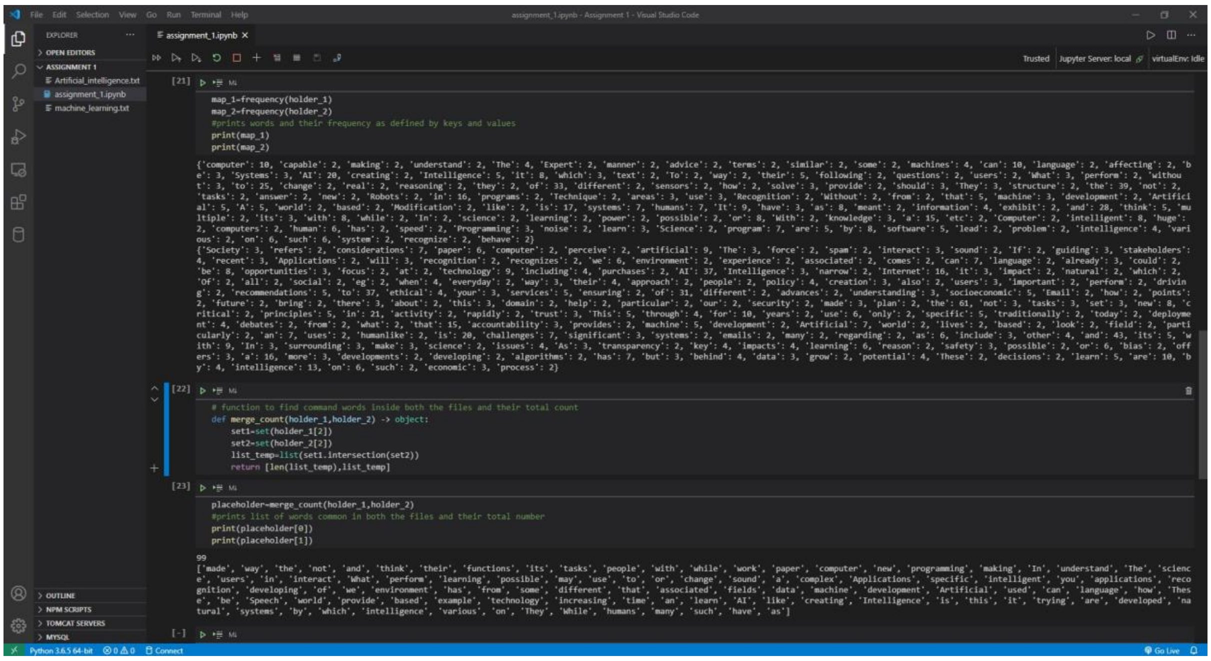
```
[15]
#counting total number of words inside the attached files
def builder_funct(string) -> object:
    ls=string.split()
    for i in range(len(ls)):
        ls[i]=ls[i].strip() # for cleaning up the string
        word_builder=""
        # for removing the punctuations from word
        for j in ls[i]:
            if j not in punctuations:
                word_builder+=j
        ls[i]=word_builder
    flag=True
    # removing remaining white spaces
    while(flag):
        try:
            ls.remove('')
        except:
            flag=False
        ls_set=list(set(ls))
    return [len(ls),ls,ls_set]
```

```
[16]
holder_1=builder_funct(file_1)
holder_2=builder_funct(file_2)

print(holder_1[0]) # prints number of words in Artificial Intelligence
print(holder_2[0])  # prints number of words in Machine learning
```

```
799
1084
```

```
[17]
def frequency(holder) -> dict:
    ls_temp1=holder[1] # original list of words
    ls_temp2=holder[2] # word set, without repetations
    map={} # frequency of values per words
    for i in ls_temp2:
        count=ls_temp1.count(i)
        if count>1:
            map[i]=count
    return map
```

```
[21]
```



```
[21]
map_1=frequency(holder_1)
map_2=frequency(holder_2)
#prints words and their frequency as defined by keys and values
print(map_1)
print(map_2)

{'computer': 10, 'capable': 2, 'making': 2, 'understand': 2, 'The': 4, 'Expert': 2, 'manner': 2, 'advice': 2, 'terms': 2, 'similar': 2, 'some': 2, 'machines': 4, 'can': 10, 'language': 2, 'affecting': 2, 'b
e': 3, 'Systems': 3, 'AI': 20, 'creating': 2, 'Intelligence': 5, 'it': 8, 'which': 3, 'text': 2, 'To': 2, 'way': 2, 'their': 5, 'following': 2, 'questions': 2, 'users': 2, 'What': 3, 'perform': 2, 'withou
t': 3, 'to': 25, 'change': 2, 'real': 2, 'reasoning': 2, 'they': 2, 'of': 33, 'different': 2, 'sensors': 2, 'how': 2, 'solve': 3, 'provide': 2, 'should': 3, 'They': 3, 'structure': 2, 'the': 39, 'not': 2,
'tasks': 2, 'answer': 2, 'new': 2, 'Robots': 2, 'in': 16, 'programs': 2, 'Technique': 2, 'areas': 3, 'use': 3, 'Recognition': 2, 'Without': 2, 'from': 2, 'that': 5, 'machine': 3, 'development': 2, 'Artifici
al': 5, 'A': 5, 'world': 2, 'based': 2, 'Modification': 2, 'like': 2, 'is': 17, 'systems': 7, 'humans': 7, 'It': 9, 'have': 3, 'as': 8, 'meant': 2, 'information': 4, 'exhibit': 2, 'and': 28, 'think': 5, 'mu
ltiple': 2, 'its': 3, 'with': 8, 'while': 2, 'In': 2, 'science': 2, 'learning': 2, 'power': 2, 'possible': 2, 'or': 8, 'With': 2, 'knowledge': 3, 'a': 15, 'etc': 2, 'Computer': 2, 'intelligent': 8, 'huge':
2, 'computers': 2, 'human': 6, 'has': 2, 'speed': 2, 'Programming': 3, 'noise': 2, 'learn': 3, 'Science': 2, 'program': 7, 'are': 5, 'by': 8, 'software': 5, 'lead': 2, 'problem': 2, 'intelligence': 4, 'vari
ous': 2, 'on': 6, 'such': 6, 'system': 2, 'recognize': 2, 'behave': 2}
{'Society': 3, 'refers': 2, 'considerations': 7, 'paper': 6, 'computer': 2, 'perceive': 2, 'artificial': 9, 'The': 3, 'force': 2, 'spam': 2, 'interact': 3, 'sound': 2, 'If': 2, 'guiding': 3, 'stakeholders':
4, 'recent': 3, 'Applications': 2, 'will': 3, 'recognition': 2, 'recognizes': 2, 'we': 6, 'environment': 2, 'experience': 2, 'associated': 2, 'comes': 2, 'can': 7, 'language': 2, 'already': 3, 'could': 2,
'be': 8, 'opportunities': 3, 'focus': 2, 'at': 2, 'technology': 9, 'including': 4, 'purchases': 2, 'AI': 37, 'Intelligence': 3, 'narrow': 2, 'Internet': 16, 'it': 3, 'impact': 2, 'natural': 2, 'which': 2,
'Of': 2, 'all': 2, 'social': 2, 'eg': 2, 'when': 4, 'everyday': 2, 'way': 3, 'their': 4, 'approach': 2, 'people': 2, 'policy': 4, 'creation': 3, 'also': 2, 'users': 3, 'important': 2, 'perform': 2, 'drivin
g': 2, 'recommendations': 5, 'to': 37, 'ethical': 4, 'your': 3, 'services': 5, 'ensuring': 2, 'of': 31, 'different': 2, 'advances': 2, 'understanding': 3, 'socioeconomic': 5, 'Email': 2, 'how': 2, 'points':
2, 'future': 2, 'bring': 2, 'there': 3, 'about': 2, 'this': 3, 'domain': 2, 'help': 2, 'particular': 2, 'our': 2, 'security': 2, 'made': 3, 'plan': 2, 'the': 61, 'not': 3, 'tasks': 3, 'set': 3, 'new': 8, 'c
ritical': 2, 'principles': 5, 'in': 21, 'activity': 2, 'rapidly': 2, 'trust': 3, 'This': 5, 'through': 4, 'for': 10, 'years': 2, 'use': 6, 'only': 2, 'specific': 5, 'traditionally': 2, 'today': 2, 'deployme
nt': 4, 'debates': 2, 'from': 2, 'what': 2, 'that': 15, 'accountability': 3, 'provides': 2, 'machine': 5, 'development': 2, 'Artificial': 7, 'world': 2, 'lives': 2, 'based': 2, 'look': 2, 'field': 2, 'parti
cularly': 2, 'an': 7, 'uses': 2, 'humanlike': 2, 'is': 20, 'challenges': 7, 'significant': 3, 'systems': 2, 'emails': 2, 'many': 2, 'regarding': 2, 'as': 6, 'include': 3, 'other': 4, 'and': 43, 'its': 5, 'w
ith': 9, 'In': 3, 'surrounding': 3, 'make': 3, 'science': 2, 'issues': 4, 'As': 3, 'transparency': 2, 'key': 4, 'impacts': 4, 'learning': 6, 'reason': 2, 'safety': 3, 'possible': 2, 'or': 6, 'bias': 2, 'off
ers': 3, 'a': 16, 'more': 3, 'developments': 2, 'developing': 2, 'algorithms': 2, 'has': 7, 'but': 3, 'behind': 4, 'data': 3, 'grow': 2, 'potential': 4, 'These': 2, 'decisions': 2, 'learn': 5, 'are': 10, 'b
y': 4, 'intelligence': 13, 'on': 6, 'such': 2, 'economic': 3, 'process': 2}

[22]
# function to find command words inside both the files and their total count
def merge_count(holder_1,holder_2) -> object:
    set1=set(holder_1[2])
    set2=set(holder_2[2])
    list_temp=list(set1.intersection(set2))
    return [len(list_temp),list_temp]

[23]
placeholder=merge_count(holder_1,holder_2)
#prints list of words common in both the files and their total number
print(placeholder[0])
print(placeholder[1])

99
['made', 'way', 'the', 'not', 'and', 'think', 'their', 'functions', 'its', 'tasks', 'people', 'with', 'while', 'work', 'paper', 'computer', 'new', 'programming', 'making', 'In', 'understand', 'The', 'scienc
e', 'users', 'in', 'interact', 'What', 'perform', 'learning', 'possible', 'may', 'use', 'to', 'or', 'change', 'sound', 'a', 'complex', 'Applications', 'specific', 'intelligent', 'you', 'applications', 'reco
gnition', 'developing', 'of', 'we', 'environment', 'has', 'from', 'some', 'different', 'that', 'associated', 'fields', 'data', 'machine', 'development', 'Artificial', 'used', 'can', 'language', 'how', 'Thes
e', 'be', 'Speech', 'world', 'provide', 'based', 'example', 'technology', 'increasing', 'time', 'an', 'learn', 'AI', 'like', 'creating', 'Intelligence', 'is', 'this', 'it', 'trying', 'are', 'developed', 'na
tural', 'systems', 'by', 'which', 'intelligence', 'various', 'on', 'They', 'While', 'humans', 'many', 'such', 'have', 'as']

[-]
```

Code :

```
# defining punctuation
punctuations = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''

# returns a string of content from the required file
def extract_content(path)->str:
    file=open(path,"r")
    return file.read()


# extracting value inside the two files
file_1=extract_content("Artificial_intelligence.txt")
file_2=extract_content("machine_learning.txt")
# printing out value
print(file_1)
print(file_2)


#counting total number of words inside the attached files
def builder_funct(string) -> object:
    ls=string.split()
    for i in range(len(ls)):
        ls[i]=ls[i].strip() # for cleaning up the string
        word_builder=""
        # for removing the punctuations from word
        for j in ls[i]:
            if j not in punctuations:
                word_builder+=j
        ls[i]=word_builder
```

```python
    flag=True
    # removing remaining white spaces
    while(flag):
        try:
            ls.remove('')
        except:
            flag=False
        ls_set=list(set(ls))
    return [len(ls),ls,ls_set]


holder_1=builder_funct(file_1)
holder_2=builder_funct(file_2)

print(holder_1[0]) # prints number of words in Artificial Intelligence
print(holder_2[0])  # prints number of words in Machine learning


def frequency(holder) -> dict:
    ls_temp1=holder[1] # original list of words
    ls_temp2=holder[2] # word set, without repetations
    map={} # frequency of values per words
    for i in ls_temp2:
        count=ls_temp1.count(i)
        if count>1:
            map[i]=count
    return map


map_1=frequency(holder_1)
map_2=frequency(holder_2)
#prints words and their frequency as defined by keys and values
print(map_1)
print(map_2)


# function to find command words inside both the files and their total count
def merge_count(holder_1,holder_2) -> object:
    set1=set(holder_1[2])
    set2=set(holder_2[2])
    list_temp=list(set1.intersection(set2))
    return [len(list_temp),list_temp]


placeholder=merge_count(holder_1,holder_2)
#prints list of words common in both the files and their total number
print(placeholder[0])
print(placeholder[1])
```