

PRACTICAL NO 7

Aim: To Implement Bloom Filters for Filter Stream Data

Software: Python Idle

Steps:

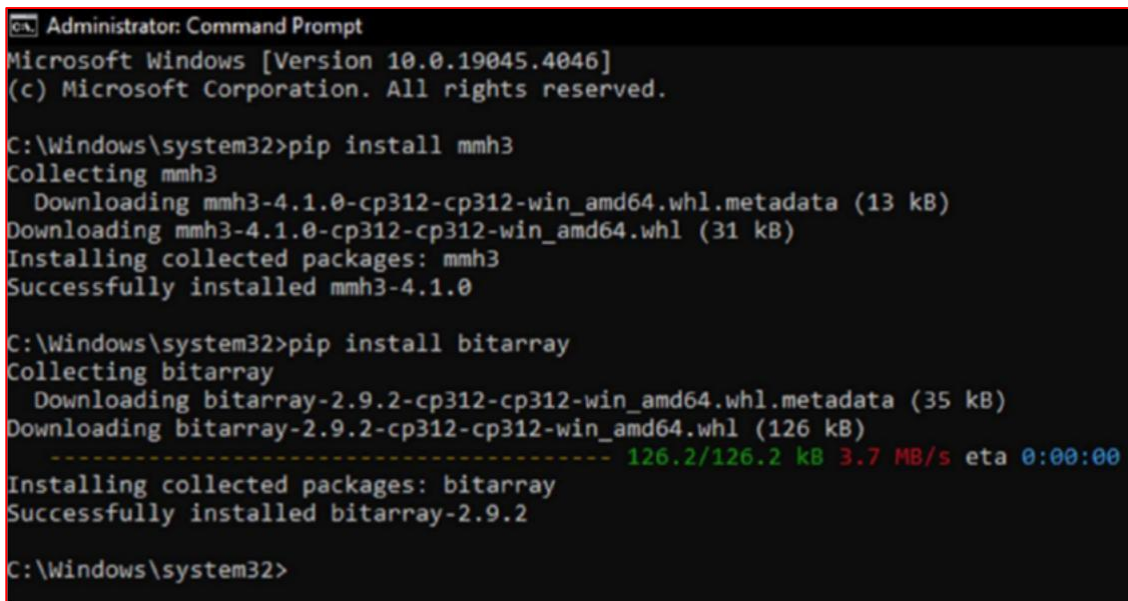
1) Open command prompt as administrator and perform below commands:

Code:

```
pip install mmh3
```

```
pip install bitarray
```

Output:



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19045.4046]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>pip install mmh3
Collecting mmh3
  Downloading mmh3-4.1.0-cp312-cp312-win_amd64.whl.metadata (13 kB)
  Downloading mmh3-4.1.0-cp312-cp312-win_amd64.whl (31 kB)
Installing collected packages: mmh3
Successfully installed mmh3-4.1.0

C:\Windows\system32>pip install bitarray
Collecting bitarray
  Downloading bitarray-2.9.2-cp312-cp312-win_amd64.whl.metadata (35 kB)
  Downloading bitarray-2.9.2-cp312-cp312-win_amd64.whl (126 kB)
----- 126.2/126.2 kB 3.7 MB/s eta 0:00:00
Installing collected packages: bitarray
Successfully installed bitarray-2.9.2

C:\Windows\system32>
```

2) Open IDLE Python

Note:-

set environment variable

C:\Users\sies\AppData\Local\Programs\Python\Python312\Scripts

Paste the above defined directory as a new path in environment variable

Here we need to create two python files “**bloomfilter.py**” and “**bloom_test.py**”

Now, open idle python and create file 1 as bloomfilter.py paste the below code

Code:

“bloomfilter.py”

```
import math
import mmh3
from bitarray import bitarray

class BloomFilter(object):

    """
    Class for Bloom filter, using murmur3 hash function
    """

    def __init__(self, items_count, fp_prob):
        """
        items_count : int
        Number of items expected to be stored in bloom filter
        fp_prob : float
        False Positive probability in decimal
        """
        # False possible probability in decimal
        self.fp_prob = fp_prob

        # Size of bit array to use
        self.size = self.get_size(items_count, fp_prob)

        # number of hash functions to use
        self.hash_count = self.get_hash_count(self.size, items_count)

        # Bit array of given size
        self.bit_array = bitarray(self.size)

        # initialize all bits as 0
        self.bit_array.setall(0)

    def add(self, item):
        """
        Add an item in the filter
        """
        digests = []
        for i in range(self.hash_count):

            # create digest for given item.
```

```

# i work as seed to mmh3.hash() function
# With different seed, digest created is different
digest = mmh3.hash(item, i) % self.size
digests.append(digest)

# set the bit True in bit_array
self.bit_array[digest] = True

def check(self, item):
    """
    Check for existence of an item in filter
    """
    for i in range(self.hash_count):
        digest = mmh3.hash(item, i) % self.size
        if self.bit_array[digest] == False:

            # if any of bit is False then, its not present
            # in filter
            # else there is probability that it exist
            return False
        return True

    @classmethod
    def get_size(self, n, p):
        """
        Return the size of bit array(m) to used using
        following formula
        
$$m = -(n * \lg(p)) / (\lg(2)^2)$$

        n : int
        number of items expected to be stored in filter
        p : float
        False Positive probability in decimal
        """
        m = -(n * math.log(p)) / (math.log(2)**2)
        return int(m)

    @classmethod
    def get_hash_count(self, m, n):
        """
        Return the hash function(k) to be used using
        following formula
        
$$k = (m/n) * \lg(2)$$


        m : int
        size of bit array
        n : int

```

```

number of items expected to be stored in filter
'''
k = (m/n) * math.log(2)
return int(k)

```

Now, open idle python and create file 2 as bloom_test.py paste the below code

“bloom_test.py”

Code:

```

from bloomfilter import BloomFilter
from random import shuffle

n = 20 #no of items to add
p = 0.05 #false positive probability

bloomf = BloomFilter(n,p)
print("Size of bit array: {}".format(bloomf.size))
print("False positive Probability: {}".format(bloomf.fp_prob))
print("Number of hash functions: {}".format(bloomf.hash_count))

# words to be added
word_present = ['abound','abounds','abundance','abundant','accessible',
'bloom','blossom','bolster','bonny','bonus','bonuses',
'coherent','cohesive','colorful','comely','comfort',
'gems','generosity','generous','generously','genial']

# word not added
word_absent = ['bluff','cheater','hate','war','humanity',
'racism','hurt','nuke','gloomy','facebook',
'geeksforgeeks','twitter']

for item in word_present:
bloomf.add(item)

shuffle(word_present)
shuffle(word_absent)

test_words = word_present[:10] + word_absent
shuffle(test_words)
for word in test_words:
if bloomf.check(word):
if word in word_absent:
print("{} is a false positive!".format(word))
else:

```

```
print("{}' is probably present!".format(word))
else:
print("{}' is definitely not present!".format(word))
```

2) Now run the “bloom_test.py” file

Output:

```
File "C:/Users/sies/AppData/Local/Programs/Python/Python312/bloom_test.py", line 1, in <module>
from bloomfilter import BloomFilter
ModuleNotFoundError: No module named 'bloomfilter'

>>>
===== RESTART: C:/Users/sies/AppData/Local/Pr
Size of bit array:124
False positive Probability:0.05
Number of hash functions:4
'bonny' is probably present!
'facebook' is definitely not present!
'geeksforgeeks' is definitely not present!
'blossom' is probably present!
'racism' is definitely not present!
'bonus' is probably present!
'gems' is probably present!
'bluff' is definitely not present!
'twitter' is a false positive!
'hurt' is definitely not present!
'cheater' is definitely not present!
'hate' is definitely not present!
'war' is definitely not present!
'accessible' is probably present!
'gloomy' is definitely not present!
'nuke' is definitely not present!
'comfort' is probably present!
'bolster' is probably present!
'generously' is probably present!
'humanity' is a false positive!
'bonuses' is probably present!
'abundance' is probably present!

>>>
===== RESTART: C:/Users/sies/AppData/Local/Pr
Size of bit array:124
False positive Probability:0.05
Number of hash functions:4
'bluff' is definitely not present!
'humanity' is a false positive!
'blossom' is probably present!
'facebook' is definitely not present!
'bonuses' is probably present!
'geeksforgeeks' is definitely not present!
'bonny' is probably present!
'hate' is definitely not present!
'abounds' is probably present!
'genial' is probably present!
'abundance' is probably present!
'war' is definitely not present!
'cheater' is definitely not present!
'racism' is definitely not present!
'nuke' is definitely not present!
```