Python Basics

Two bioinformatics examples

Example 1

- This example will use the following data file, provided in the "examples" folder.
  - gene_info.bed: A tab delimited table with gene information. Each row in the file contains information about a gene.

- Read the gene info from the "gene_info.bed" file into a list.
- Calculate the gene length from the start and end positions.
- Output the largest 5 genes.
- Output the smallest 5 genes.

Parse a bed file

Download the bed file from https://genome.ucsc.edu/goldenPath/help/examples/bedExample2.bed

Save the file in a local directory

The first a few rows

```
chr7 54028 73584 uc003sii.2 0 - 54028 54028 255,0,0 . AL137655
chr7 60328 61569 uc010krx.1 0 - 60328 60328 255,0,0 . PDGFA
chr7 62967 63529 uc003sij.2 0 - 62967 63366 255,0,0 . DQ576410
chr7 64068 64107 uc003sil.1 0 - 64068 64068 255,0,0 . DQ584609
chr7 65159 65220 uc003sim.1 0 - 65159 65159 255,0,0 . DQ600587
```

From this file, we want to generate a new file with gene name, gene length, chromosome, start, end

https://genome.ucsc.edu/goldenPath/help/examples/bedExample2.bed

```
# keep in mind, don't use 'w' here, otherwise you will end up with a
blanck file.

data = []
with open(‘C:/users/xiaoman/downloads/bedExample2.bed', 'r') as f_in:
  for i, line in enumerate(f_in):

    # these code only for demo purpose
    if i == 0:
      print('orginal', line)
      print('Before strip', line.split('\t'))
      print('After strip', line.strip('\n').split('\t'))

    content = line.strip('\n').split('\t')
    chrom, chrom_start, chrom_end, gene_name = content[:4]
    # change data type
    chrom_start, chrom_end = int(chrom_start), int(chrom_end)
    length = chrom_end - chrom_start + 1
    data.append([length, chrom, chrom_start, chrom_end, gene_name])

# now we sort
data.sort()
```

See the results

```
# Output the first three rows
data[:3]

# Output the last three rows.
data[-3:]


# Output the largest 5 genes.
for idx in range(len(data)-1, len(data)-6, -1):
  print(f"chrom : {data[idx][1]}, chrom_start : {data[idx][2]}, chrom_end :
{data[idx][3]}, gene_name: {data[idx][4]}, length : {data[idx][0]}")


# Output the smallest 5 genes.
for idx in range(0, 5):
  print(f"chrom : {data[idx][1]}, chrom_start : {data[idx][2]}, chrom_end :
{data[idx][3]}, gene_name: {data[idx][4]}, length : {data[idx][0]}")
```

Another solution with dictionary

```
# keep in mind, don't use 'w' here, otherwise you will end up
with a blanck file.
data = [] # list to store data
dic = {} # dictionary to store length
with open('bedExample2.bed', 'r') as f_in:
  for i, line in enumerate(f_in):
    # these code only for demo purpose
    if i == 0:
      print(line.split('\t'))

    content = line.strip('\n').split('\t')
    chrom, chrom_start, chrom_end, gene_name = content[:4]
    # change data type
    chrom_start, chrom_end = int(chrom_start), int(chrom_end)
    length = chrom_end - chrom_start + 1
    data.append((chrom, chrom_start, chrom_end, gene_name))
    dic[(chrom, chrom_start, chrom_end, gene_name)] = length

# now we sort
data.sort(key = lambda x : (dic[x], x))
```

See the results

```
# Output the first three rows
data[:3]

# Output the last three rows.
data[-3:]


# Output the largest 5 genes.
for idx in range(len(data)-1, len(data)-6, -1):
  print(f"chrom : {data[idx][1]}, chrom_start : {data[idx][2]}, chrom_end :
{data[idx][3]}, gene_name: {data[idx][4]}, length : {data[idx][0]}")


# Output the smallest 5 genes.
for idx in range(0, 5):
  print(f"chrom : {data[idx][1]}, chrom_start : {data[idx][2]}, chrom_end :
{data[idx][3]}, gene_name: {data[idx][4]}, length : {data[idx][0]}")
```

Example 2

- Find the most frequent k-mers in a sequence file. You can use k=6, and the sequence file from

https://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_41/gencode.v41.lncRNA_transcripts.fa.gz

An example sample sequence file with an example sequence

>sequence_name|sequence_description
AATGGCTACTCCATAGGTAGAGCAGCCCTGAGGGCTGCTGGTTGCCTGTTTTTAT
GATTATTTCTTGATTAT

Warm up

```
# To read the file, we can do
with open('/content/gencode.v41.lncRNA_transcripts.fa','r') as f_in:
  for i,line in enumerate(f_in):
    # do something here
    pass
```

```
curr_name, curr_seq, dic_name_to_seq = '', '', {}

with open('gencode.v41.lncRNA_transcripts.fa','r') as f_in:
    for i,line in enumerate(f_in):
        # if line.startswith('>'):
        if line [0]==('>'):
          if curr_seq:
            dic_name_to_seq[curr_name] = curr_seq
          curr_name = line.strip('\n')[1:]
          curr_seq = ''
        else:
          curr_seq += line.strip('\n')

if curr_seq:
    dic_name_to_seq[curr_name] = curr_seq

print(f'The number of sequences we obtained from file, {len(dic_name_to_seq)}')
```

```
k = 6

counter = defaultdict(int)

for i in range(k, len(target_seq)+1):

  counter[target_seq[i-k:i]] += 1

sorted_k_mers = sorted(list(counter.keys()), \
                       key = lambda x : counter[x], \
                       reverse = True)

print(f"Most frequent 6-mers:{sorted_k_mers[0]}")

print(f"Most frequent 6-mers occured {counter[sorted_k_mers[0]]} times.")

print('\nTop 5 K-mers and frequencies:')
for k_mers in sorted_k_mers[:5]:
  print(f'\t{k_mers} =  {counter[k_mers]} times')
```

```
def find_top5_k_mers(k, seq):

  counter = Counter()

  for i in range(k, len(seq)+1):

    counter[seq[i-k:i]] += 1

  sorted_k_mers = sorted(list(counter.keys()), key = lambda x : counter[x], reverse = True)

  print(f"Most frequent {k}-mers:{sorted_k_mers[0]}")

  print(f"Most frequent {k}-mers occured {counter[sorted_k_mers[0]]} times.")

  print(f'\nTop 5 {k}-mers and frequencies:')
  for k_mers in sorted_k_mers[:5]:
    print(f'\t{k_mers} =  {counter[k_mers]} times')



   find_top5_k_mers(5, target_seq)
```

Congrats!, you are done with Python Basics