LSI_Project_2

October 29, 2019

[7]:
```
# Input file reading using readlines method
with open (r'C:\Users\Ram Kumar R S\Downloads\CYCS_dog.txt','r') as f:
    x=f.readlines()    # x is list that contains the data separated by a ','␣
 ↪separator

# Method to split the header and sequence
list_of_headers=[]
sequence=[]
for i in x:
    if i[0]==">": # appending the header of the sequence
        list_of_headers.append(i.strip('\n').strip('>'))
    else: # appending the sequence itself
        sequence.append(i.strip('\n'))

# Method to modify the sequence in such a way that it distinguishes one sequence␣
 ↪to another

seq=''
for i in sequence:
    if i!='':    # checks for string characters
        seq=seq+i
    else:
        seq=seq+'\n'

# Given below is a list with each sequences separated by ','
list_of_sequences=seq.split('\n')

# The total length of dog Cytochrome C gene
complete_sequence=seq.replace('\n','')
len(complete_sequence)

# It extract the required words from the fasta file header
modified_headers=[]
for i in list_of_headers:    # extracts the index value of -2 from the reverse␣
 ↪reading
    modified_headers.append(i.split(':')[-2])
```

```
# Calculate the number of exons, introns, cds
count_exon=0
count_intron=0
count_CDS=0
for i in list_of_headers:
    if "exon" in i:  # counting exons
        count_exon+=1
    elif "intron" in i:  # counting intron
        count_intron+=1
    elif "cds" in i:  # counting cds
        count_CDS+=1

# It calculates individual nucleotide counts, at content, gc content for each␣
 ↪sequence
actg_counts=[] # empty list to hold the counts of nucleotides
at_content=[]  # empty list to hold the at content of the sequences
gc_content=[]  # empty list to hold the gc content of
for i in list_of_sequences:
    a,c,t,g=i.count('A'),i.count('C'),i.count('T'),i.count('G')  # count method␣
 ↪applied
    at=((i.count('A')+i.count("T"))/(len(i)))*100                # at content␣
 ↪calucation
    gc=((i.count('G')+i.count('C'))/(len(i)))*100                # gc content␣
 ↪calculation
    actg_counts.append((a,c,t,g))     # appending of the counts
    at_content.append(at)             # appending the at content
    gc_content.append(gc)             # appending the gc content


# Output writing to a file
with open("Result2.txt",'w') as fout:
    fout.write('The given dataset which is complete dog Cytochrome C Gene has {}␣
 ↪introns, {} exons and {} CDS  in it.\n'.
 ↪format(count_intron,count_exon,count_CDS))
    fout.write('The length of the complete dog Cytochrome C Gene is {}.\n\n'.
 ↪format(len(complete_sequence)))
    fout.write('The complete statistics is as follows:\n\n\n')
    for i in range(len(list_of_sequences)):
        fout.write('The statistics of {} is below:\n'.
 ↪format(modified_headers[i]))
        fout.write('The count of nucleotides are:\n\tA is {},\n\tC is {},\n\tT␣
 ↪is {} and\n\tG is {}.\n'.
 ↪format(actg_counts[i][0],actg_counts[i][1],actg_counts[i][2],actg_counts[i][3]))
        fout.write('The length of the sequence is {}.\n'.
 ↪format(len(list_of_sequences[i])))
```

```
        fout.write('The AT content rounded to two decimal value is {:.2f}.\n'.
↪format(at_content[i]))
        fout.write('The GC content rounded to two decimal value is {:.2f}.\n\n'.
↪format(gc_content[i]))
        print('Alright, Everything done!','printing for', modified_headers[i])
```

```
Alright, Everything done! printing for ENSCAFT00000004569.3 cds
Alright, Everything done! printing for ENSCAFT00000004569.3 ENSCAFE00000296668
exon
Alright, Everything done! printing for ENSCAFT00000004569.3 ENSCAFE00000031066
exon
Alright, Everything done! printing for ENSCAFT00000004569.3 ENSCAFE00000031069
exon
Alright, Everything done! printing for ENSCAFT00000004569.3 intron 1
Alright, Everything done! printing for ENSCAFT00000004569.3 intron 2
```

[ ]: