DSL253 statistical programming

Assignment_1 report

Shashank yadav
12342010

INTRODUCTION:

This report analyzes the probability and entropy of English alphabets and words in large text files. The analysis will help as a foundation for a publishing firm's development of specialized printing machines tailored to the patterns of the English language. and observe how randomness in data can lead to improved accuracy of overall mean and standard deviation and variance

The report is divided into three parts:

1. Alphabet Probability and Entropy Analysis: This involves determining the probability of each alphabet and word in English using file A, file B, file C and file D listing the top ten most frequent alphabets and total entropy.
2. Calculate the mean and variance of randomly generated numbers from a uniform distribution.
3. Calculating the mean and standard deviation of the numbers generated by normal distribution.

DATA:
File A contains a random characters.
File B ,File C, File D contains words and sentences

Methodology

For question 1 A:

Calculating freq of each alphabet and probabilities :

- Opened the file fileA for reading line by line.

```
seen={}   # creating a dictionary that will store all character and thier freq
with open("fileA.txt") as file:   # opeing the file
    for line in file:    # extracting each line as a for loop
        for i in line:   # extracting each charachter in the  line
            if 65<=ord(i)<=90 or 97<=ord(i)<=122:   # if thier assci val lies between range then it's a valid character
                if i in seen:
                    seen[i][0]+=1   # if it's present than icrement freq
                else:
                    seen[i]=[1]     # if it is not in seem than make it's entry
print(seen)  # print the final result
```

- Iterated through each line and extracted individual characters.
- Filtered valid alphabetic characters by checking if their ASCII values were within the range of uppercase (A-Z) and lowercase (a-z).
- Used a dictionary seen to store each character as a key and its frequency as a value.
- Incremented the frequency of characters already present in the dictionary; for new characters, initialized their frequency to 1.
- Calculated the total frequency of all characters.

```
total_freq=sum([k[0] for k in seen.values()])     # calculating the total freq
total_freq
```

```
10050
```

```
for i,j in seen.items():
    print("the probability of occurence of",i,"is",j[0]/total_freq)
    j.append(j[0]/total_freq)    # calcuating the mean and storing it in the same dictionary for the future use
```

- Determined the probability of occurrence for each character by dividing its frequency by the total frequency. Updated the dictionary to include the calculated probability for future reference.

Sorting and Top 10 Analysis:

- Sorted the dictionary by frequency values in ascending order.
- Extracted the top 10 most frequent characters using array slicing and printed them for visualization.
- Generated a bar chart using matplottlib to represent the probabilities of characters in file A.

```
sorted_freq=dict(sorted(seen.items(), key=lambda item: item[1]))    # sorting the dict by it's val to fin the most frequent at the bottom
print("the top 10 most ouccuring characters are :")
check=0
most=list(sorted_freq.keys())  #soring the most freq character in  a list

print(most[-1:-11:-1])   # printing the top 10 most freq characters using array slicing

import matplotlib.pyplot as plt
probabilities=[k[1] for k in seen.values() ]
plt.bar(seen.keys() ,height=probabilities)   # creating a pie chart
plt.show()
```

For question 1 B:

Building entropy function :

- Using the same techniques from part A to calculate the probability
- Passing them to the entropy function to get the result.

```
import math
def entropy(p):   #making a founcntion to calculate the entropy

    return sum(k*math.log2(k) for k in p)*-1
```

```
file_b={}  # creating a freq names dictionary
with open("fileB.txt") as file:
    for line in file:
        for i in line:
            if 65<=ord(i)<=90:    # if ascii vale is in this range than it is in capital so make it small
                i=chr(ord(i)+32)
            if 97<=ord(i)<=122:  # if ascii  value in this e=range than it is in small alphabet whoch we can store
                if i in file_b:
                    file_b[i][0]+=1
                else:
                    file_b[i]=[1]
print(file_b)
total_freq_B=sum([k[0] for k in file_b.values()])
for i,j in file_b.items():
    print("the probability of occurence of",i,"is",j[0]/total_freq_B)
    j.append(j[0]/total_freq_B)  # calculating and storing mean for fututre use
```

```
probablity=[k[1] for k in file_b.values()]   # storing the probability to pass into founction
total_entropy=entropy(probablity)  # calculating the entropy
print("the entropy is :",total_entropy)
```

For question 1 C:

- Using extra filters to escape the special filters by lowercasing the alphabet and replacing the special characters used library like re ,counter.
- Than repeating the steps of part A and b

```
import re
from collections import Counter
import math

def process(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        text = file.read().lower()
    clean_text = re.sub(r'[^a-z\s]', '', text)      #remove the pecial characetrs from the txt
    return clean_text

def calculate_probability_and_entropy(text):
    words = text.split()
    total_words = len(words)
    word_counts = Counter(words)
    probabilities = {word: count / total_words for word, count in word_counts.items()}      #calculating the probability of eac entri

    entropy = -sum(p * math.log2(p) for p in probabilities.values())
    return probabilities, entropy
cleaned_text_c = process("fileC.txt")
word_probs_c, entropy_c = calculate_probability_and_entropy(cleaned_text_c)
print(f"Top 10 words in File C: {Counter(cleaned_text_c.split()).most_common(10)}")
print(f"Entropy of words in File C: {entropy_c:.4f}")

cleaned_text_d = process("fileD.txt")
word_probs_d, entropy_d = calculate_probability_and_entropy(cleaned_text_d)
print(f"Top 10 words in File D: {Counter(cleaned_text_d.split()).most_common(10)}")
print(f"Entropy of words in File D: {entropy_d:.4f}")
```

For question 2:

Generating numbers :

- Using the numpy inbuilt fn np.random.uniform to get numbers from a uniform distribution between 0,1 of size i
- Vary I from 5 to 1000 by a loop and store the numbers in an array.

```
import random
import numpy as np
for i in range(5,1000):   # number of numbers generated is will be in range(5 to 1000)
  random_array = np.random.uniform(0.0, 1.0, i)      # using inbuilt unifirm fn to make array of random values form unifirm fn
  print("mean:",sum(random_array)/i,"variance:",np.var(random_array))    # calculating the mean and variance using numoy inbuilt fn
```

To calculate the mean:

- To calculate the mean we have used np.mean() in built fn.

For question 3:

Generating numbers:

- Using the numpy inbuilt fn np.random.normal to get numbers from a normaldistribution between 0,1 of size i
- Vary I from 5 to 1000 by a loop and store the numbers in an array.

To calculate the mean:

- To calculate the mean we have used np.mean() in built fn.

```
import random
import numpy as np
for i in range(5,1000): # number of numbers generated is will be in range(5 to 1000)
    arr=np.random.normal(4, 3, i) # using the inbuilt fn of normal distribution to make a array to from nomal/guassian distribution
    print("mean:",sum(arr)/i,"standard deviation:",np.std(arr))   # calculating the mean and standarddevistaion using numpy in built fn
```

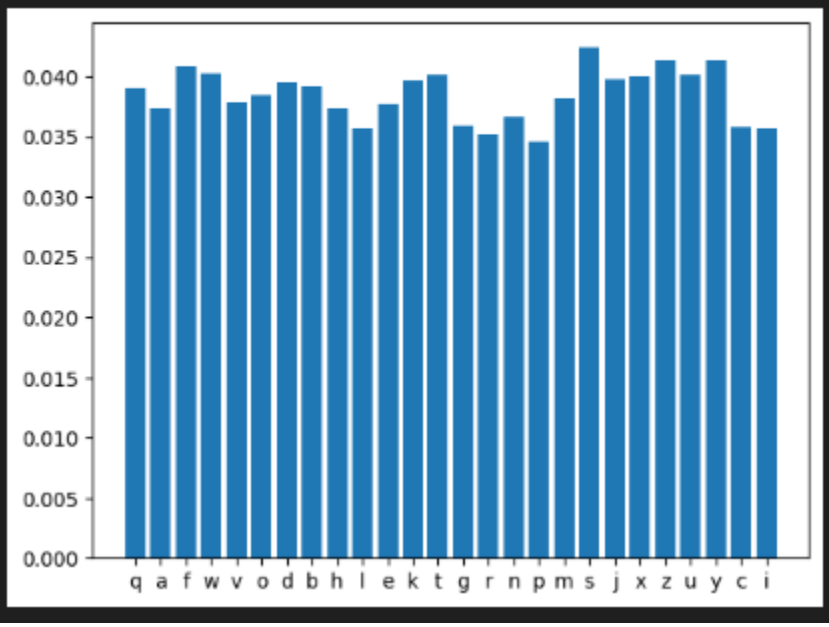
RESULTS:

Question 1A:

```
the probability of occurence of q is 0.039104477611940296
the probability of occurence of a is 0.037313432835820895
the probability of occurence of f is 0.040796019900497512
the probability of occurence of w is 0.040298507462686567
the probability of occurence of v is 0.037810945273631841
the probability of occurence of o is 0.038507462686567163
the probability of occurence of d is 0.039502487562189055
the probability of occurence of b is 0.039203980099502488
the probability of occurence of h is 0.037412935323383086
the probability of occurence of l is 0.035721393034825871
the probability of occurence of e is 0.037711442786069652
the probability of occurence of k is 0.039601990049751244
the probability of occurence of t is 0.040099502487562189
the probability of occurence of g is 0.035920398009950249
the probability of occurence of r is 0.035223880597014925
the probability of occurence of n is 0.036616915422885572
the probability of occurence of p is 0.034626865671641791
the probability of occurence of m is 0.038208955223880597
the probability of occurence of s is 0.042388059701492536
the probability of occurence of j is 0.039800995024875622
the probability of occurence of x is 0.04
the probability of occurence of z is 0.041293532338308458
the probability of occurence of u is 0.040099502487562189
the probability of occurence of y is 0.041293532338308458
the probability of occurence of c is 0.035820895522388060
the probability of occurence of i is 0.035621890547263682
```

{'q': [393], 'a': [375], 'f': [410], 'w': [405], 'v': [380], 'o': [387], 'd': [397], 'b': [394], 'h': [376], 'l': [359], 'e': [379], 'k': [398], 't': [403], 'g': [361], 'r': [354], 'n': [368], 'p': [348], 'm': [384], 's': [426], 'j': [400], 'x': [402], 'z': [415], 'u': [403], 'y': [415], 'c': [360], 'i': [358]}

```
the top 10 most occcuring characters are :
['s', 'y', 'z', 'f', 'w', 'u', 't', 'x', 'j', 'k']
```

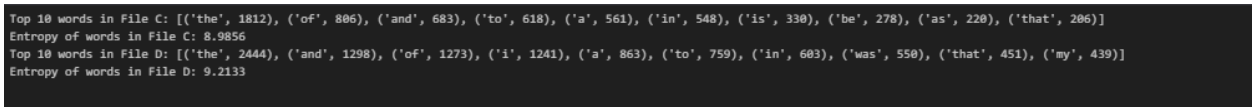


Question 1B :

Entropy =4.175975257302761

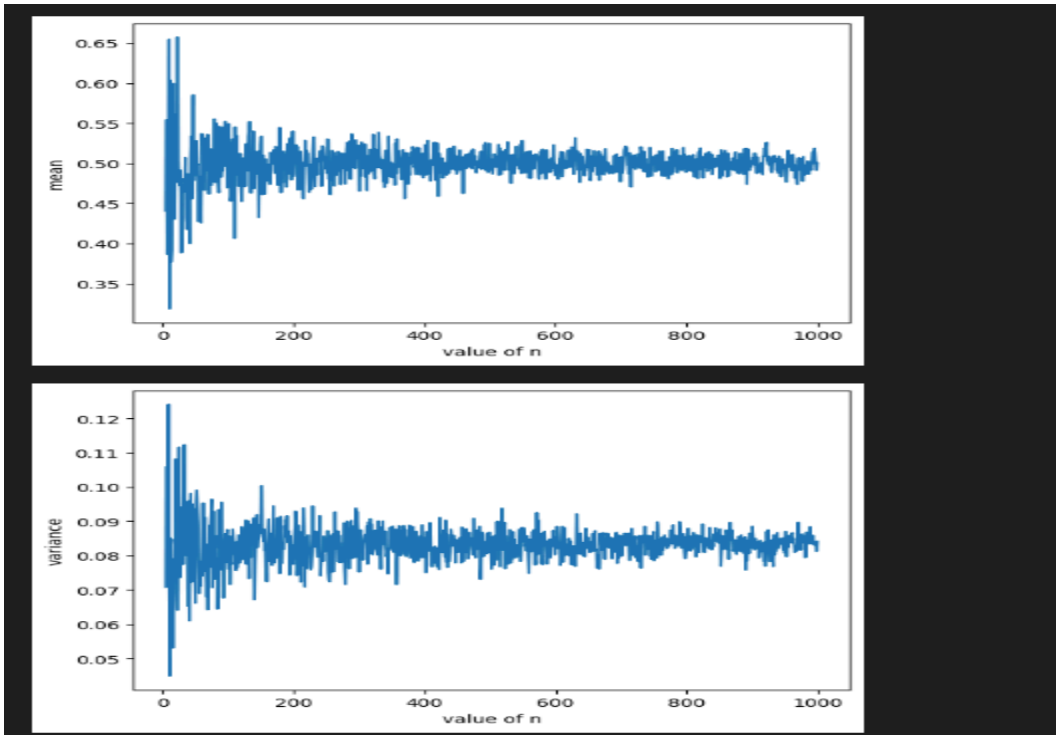

{'j': [1322], 'a': [63844], 'n': [55278], 'e': [103429], 'y': [17616], 'r': [48538], 'u': [23988], 't': [68576], 'o': [61867], 'b': [11421], 'i': [57146], 'g': [15506], 'p': [12601], 'h': [46393], 'c': [19349], 'l': [32947], 'f': [17260], 's': [50845], 'd': [37972], 'v': [7736], 'm': [22636], 'w': [18974], 'k': [6193], 'q': [957
the probability of occurence of j is 0.0016463344773822036
the probability of occurence of a is 0.07950724536610394
the probability of occurence of n is 0.06883969534894815
the probability of occurence of e is 0.12755854325555081
the probability of occurence of y is 0.021937842778793418
the probability of occurence of r is 0.060446129245973826
the probability of occurence of u is 0.029873125146327006
the probability of occurence of t is 0.08540017631968817
the probability of occurence of o is 0.07784521566732586
the probability of occurence of b is 0.014222984921469098
the probability of occurence of i is 0.07116598339219622
the probability of occurence of g is 0.019318183363304426
the probability of occurence of p is 0.015692481656197542
the probability of occurence of h is 0.05777488306292933
the probability of occurence of c is 0.024895081444889832
the probability of occurence of l is 0.041030092304328 12
the probability of occurence of f is 0.021494503085943143
the probability of occurence of s is 0.06311911989598953
the probability of occurence of d is 0.047287986789874914
the probability of occurence of v is 0.009633920945582395
the probability of occurence of m is 0.028189430858246865 5
the probability of occurence of w is 0.023629089359947
the probability of occurence of k is 0.007637002573535294 45
the probability of occurence of q is 0.001191786758589084
the probability of occurence of x is 0.0016251637617123872
the probability of occurence of z is 0.0004122062874534867

Question 1C:


Top 10 words in File C: [('the', 1812), ('of', 806), ('and', 683), ('to', 618), ('a', 561), ('in', 548), ('is', 330), ('be', 278), ('as', 220), ('that', 206)]
Entropy of words in File C: 8.9856
Top 10 words in File D: [('the', 2444), ('and', 1298), ('of', 1273), ('i', 1241), ('a', 863), ('to', 759), ('in', 603), ('was', 550), ('that', 451), ('my', 439)]
Entropy of words in File D: 9.2133

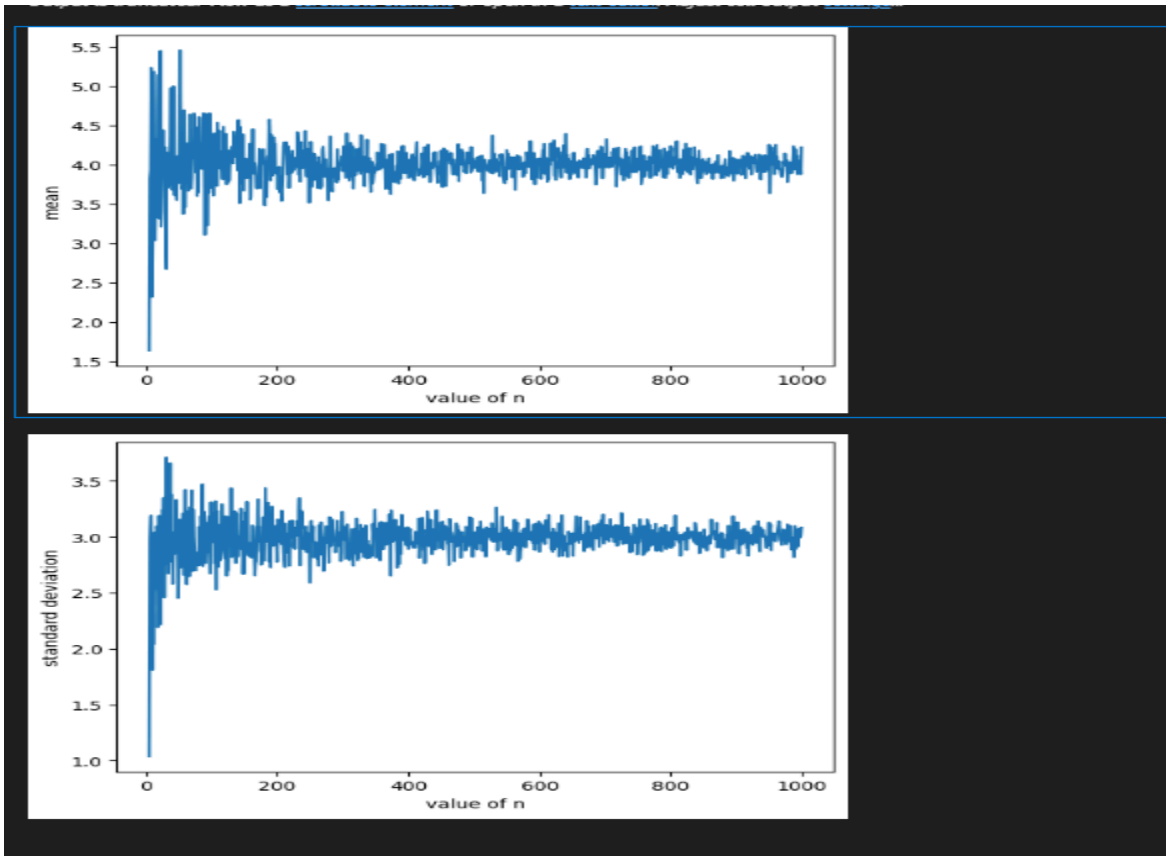
Question 2:

```
mean: 0.7118946125365341 variance: 0.07934604812277028
mean: 0.5318078839059855 variance: 0.15817229895653315
mean: 0.43170208964061435 variance: 0.11664043944013665
mean: 0.7048705978170586 variance: 0.022453061883499152
mean: 0.3349522962906185 variance: 0.06884347729124247
mean: 0.4266009383588932 variance: 0.08268101765018904
mean: 0.3093206137223783 variance: 0.09420151483676208
mean: 0.4189286651289074 variance: 0.07533314419988321
mean: 0.6042335177589593 variance: 0.11761428337286856
mean: 0.5213986622103471 variance: 0.07959441300550182
mean: 0.44401520723924454 variance: 0.07852073898764239
mean: 0.6520277938672187 variance: 0.06579826814808337
mean: 0.5082203579719786 variance: 0.09549726251176846
mean: 0.5078050978364201 variance: 0.080447675695358
mean: 0.5180205370133641 variance: 0.09995398065946257
mean: 0.5571473374334655 variance: 0.06965255596974965
mean: 0.49432475412030774 variance: 0.06482387518553392
mean: 0.4589776915933302 variance: 0.05474141451443286
mean: 0.40030589631683533 variance: 0.09583711100563373
mean: 0.4475948930114923 variance: 0.07148074929922058
mean: 0.48002860648809254 variance: 0.08072606856310084
mean: 0.5763973310676751 variance: 0.0812404457457705
mean: 0.44858831000043115 variance: 0.09397478120886002
mean: 0.49344017708442595 variance: 0.09771987259858343
mean: 0.3991185590053006 variance: 0.07757959829496744
...
mean: 0.5066381484527582 variance: 0.08378716562571699
mean: 0.5065938191062042 variance: 0.08058844754245559
mean: 0.5025945032658935 variance: 0.08279884575145675
mean: 0.5103991914221571 variance: 0.08632616944056311
```



Question 3:

```
mean: 5.017811603183711 standard deviation: 3.138473697892923
mean: 3.851505852955635 standard deviation: 1.2290099012732516
mean: 4.2943670712318545 standard deviation: 3.323903555954965
mean: 4.004971170387709 standard deviation: 2.0470081433565808
mean: 3.8527423703935515 standard deviation: 2.206598671459161
mean: 3.134859011164067 standard deviation: 2.421529546200185
mean: 4.561571887399658 standard deviation: 4.36092190286566
mean: 3.727964242247843 standard deviation: 2.8079619687912643
mean: 6.027443471692122 standard deviation: 2.3347796988614613
mean: 3.2279527015446594 standard deviation: 2.7642881131274
mean: 4.886316522551817 standard deviation: 3.2320020954023936
mean: 4.130307775182029 standard deviation: 2.3573351429845717
mean: 3.5795932866354585 standard deviation: 2.6676709830127048
mean: 4.227427274928599 standard deviation: 2.8276111448952754
mean: 2.8103454389471625 standard deviation: 1.8472314171837585
mean: 4.282784662302992 standard deviation: 2.6504067477756514
mean: 3.8186086651887843 standard deviation: 2.0894217721686195
mean: 4.000477024253133 standard deviation: 2.413184015860296
mean: 4.286100863661222 standard deviation: 3.160922388314517
mean: 3.6194270908891126 standard deviation: 3.3066801390692038
mean: 4.290802729385087 standard deviation: 2.3711412119705555
mean: 3.743018425042666 standard deviation: 2.758100681165911
mean: 3.466638719823516 standard deviation: 2.7162948390326442
mean: 3.6990064600088277 standard deviation: 3.3519380859552514
mean: 4.429342362534743 standard deviation: 2.456663301358106
...
mean: 3.9590054167238833 standard deviation: 2.9489732748222512
mean: 4.246183147902913 standard deviation: 2.9914913775873604
mean: 3.913470564958174 standard deviation: 3.1651915867098874
mean: 3.987111228196663 standard deviation: 2.9837897377044165
```



Discussion

For question 1

- The most frequently occurring characters were s, and w in file A.
- Words like “the”,“of” and “two” were most frequent in file B which shows that these words are very common arbitrary sentences. The same goes for File C and file D.
- The probability distribution highlighted the dominance of a small subset of alphabets, consistent with natural language trends.

Question 2

Uniform Distribution (0,1):

- Generated random numbers show mean and variance values that approached the theoretical mean (0.5) and variance (1/12 ≈ 0.0833) as the

Question 3

Normal Distribution (4,3):

- Similarly, random numbers from a Gaussian distribution with a mean of 4 and standard deviation of 3 closely approximated these values as the sample size grew.
- The results verify the stability and predictability of random number generators under specified distributions.

Conclusion

- The probabilities of English alphabets and words were calculated, and the top 10 most frequent characters and words were identified. The results map with linguistic patterns like “the”,” to”, and “of ” are the most frequent with this data we can customize the printing machine for the publishing firm.

- Random numbers generated from uniform and normal distributions demonstrated convergence toward theoretical mean and variance values as the sample size increased. This validates the reliability of random number generators and the law of large numbers.