DSc Assignment-2

Manan Aggarwal 2022273
Shobhit Raj 2022482

November 25, 2024

1 Question 1

(b) After applying the provided hash function, by taking the last 4 characters of md5 hash of the input and converting it from base 16 to base 10, we were able to obtain the hashes of the words, which we stored in word_hashes.txt file, the first five hashes were: 24484, 349, 14811, 30807 and 5815.

(c) The experiment was ran 5 times, with h1 being the universal hash function and h2 being the random hash function. Random hash function h2 exhibited a much higher variability, having bigger differences between max and min chains. It can be clearly seen that the universal hash function performs much better than the random hash function, with the bucket size never exceeding 2 in the case of universal hash function. The minimum size of buckets for both the hash functions was 0.

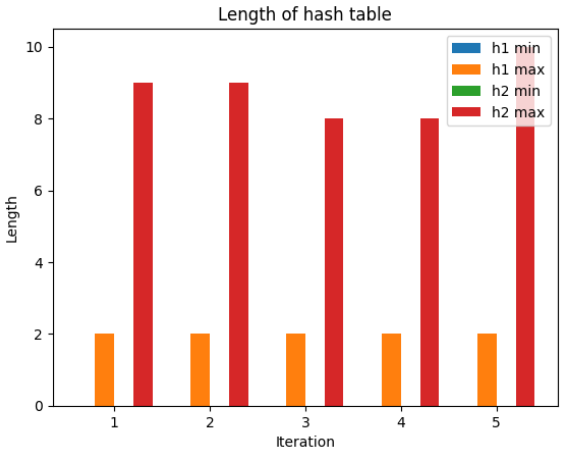

Figure 1: min and max bucket size for universal and random hash functions

(d) The output from the flajolet-martin algorithm varied quite a bit, where we obtained an average of 164510 over 10 experiments.

```
Estimated number of unique elements (1): 46340.95001184158
Estimated number of unique elements (2): 370727.60009473265
Estimated number of unique elements (3): 23170.47500592079
Estimated number of unique elements (4): 46340.95001184158
Estimated number of unique elements (5): 185363.80004736633
Estimated number of unique elements (6): 370727.60009473265
Estimated number of unique elements (7): 46340.95001184158
Estimated number of unique elements (8): 370727.60009473265
Estimated number of unique elements (9): 92681.90002368316
Estimated number of unique elements (10): 92681.90002368316
Average: 164510.3725420376
```

Figure 2: Output of 10 experiments of flajolet-martin

2 Question 2

The values of n and d obtained from the kddcup99 dataset were 494021 and 41 respectively.

(a) After projecting the data onto $\mathbb{R}^{20}$, we ran the KMeans clustering algorithm to group the data, where we obtained the sum of squared distances from the respective centroids as loss.

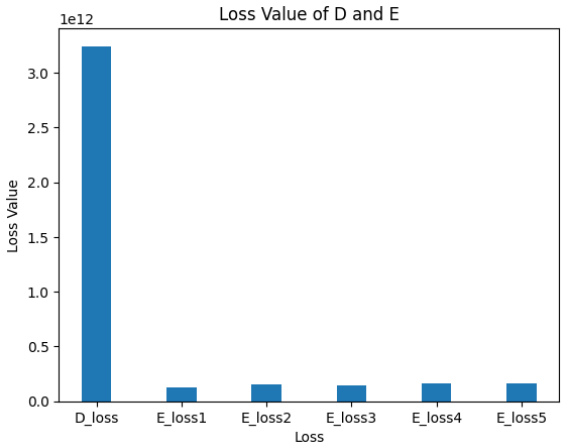

Figure 3: KMeans Clustering Loss

We ran the algorithm once on the original dataset and then 5 times on all the projected datasets.

Projection	Loss
D_loss	3244311308432.4795
E_loss1	124091136058.02016
E_loss2	154484520599.05118
E_loss3	139732065530.1511
E_loss4	166468970845.66122
E_loss5	159947934773.04117

Table 1: Projection and Loss Values From KMeans Clustering

(b) We made a JL matrix family with sparse values and $sparsity = \frac{1}{3}$. We then projected the data onto $\mathbb{R}^{410\times41}$. The experiments were ran in a way similar to (a).

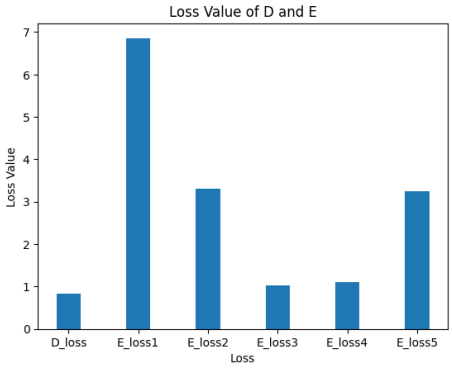

Figure 4: Linear Regression Loss

The results obtained show that the JL matrices increased the loss due to the loss in information due to projection but still performs very optimally given the low computational complexity.

Projection	Loss
D_loss	0.8240358245982908
E_loss1	6.859092328881431
E_loss2	3.2982698145385596
E_loss3	1.0246018287059169
E_loss4	1.1060038306219648
E_loss5	3.241389593287925

Table 2: Projection and Loss Values From KMeans Clustering