Data Science
CSE558
Assignment - 2

Submit by 2359 of 25/11/24 Maximum marks: 50

- **Form a group of two person and submit your assignment. Only one of the group member should submit the assignment before deadline.**
- **you are encouraged to discuss with other groups. Mention their names in your submission. If you have used the internet to understand the solution you are writing, then mention the URL.**
- **Write answers and code of your own; do not copy from others. We will follow the standard plagiarism policy, which can be found link**
- **Clearly state any extra information (such as assumption or encoding) used to reach your answers from the given questions. Write all the steps that you followed.**
- **Prepare one zip file with all your code/python notebook and plots (if any). Name it "roll_no-DSA2", e.g., "20001-DSA2". Submit the zip file through Google Classroom. A delay in submission would cost you a deduction in marks as per class policy. You will be graded based on a viva. If you submit images of your handwritten notes then you will be evaluated on at max 50% of the full marks for the question.**

Best wishes!

---

1. Consider the data file word.txt. Each line contains a word. Answer the following questions.
   (a) Create a hash table data structure. Make this a **class** in python. It should have the array **A**, which should be a python **list** of sized m for some user-speci ed value of **m**, specified when the class is instantiated. Each position of this list should also store another python list as the chain of elements which are mapped to this bucket. You can choose the prime number to be p = 524287. Use **random** library in python to choose **a** as specified in class and then create the function such that it hashes element **x** to $\mathbf{h}_a(\mathbf{x}) = ((\mathbf{ax}) \mod \mathbf{p}) \mod \mathbf{m}$. **(5)**
   (b) We will experiment with a hash function that maps strings to buckets. Consider the data file word.txt. We will use the md5 hash function to create a hashtable from this. You can read how to use the md5 library in here. Use the **hexdigest()** method and then take the last 4 digits. This will create a hash function that maps any string into an index between 0 and $16^4 = 65536$. For each string in the input file, output the hash value. **(5)**
   (c) We will experiment with the maximum chain length for two hash functions. The first one is the universal hash function you created in problem (a). The

second one is a random hash function, which you can create as the following line of Python code for a given value **x**. We will store the targets of this random hash function in a python dictionary as **h[x] = int(m * random.random())**, where **h[x]** is the bucket index of the table where x should be stored. Choose m = 500000. Now, for each of the two hash functions, calculate the hash bucket index for all numbers in $1, \ldots, m$, and find out the maximum and minimum number of elements in a bucket. Do this experiment 5 times for each of the hash functions. Present your result in a table or bar graph. Which of these hash functions has more of a difference between the maximum and minimum chain? **(10)**

(d) Use Flajolet-Martin to estimate the number of unique words in the list. Consider the id of every word as the last 4 digits as used in problem (b) and use the universal hashing of problem (a) with m = 500000 to compute the hash values of every id. Compute the trailing zeroes of every hash function and maintain the largest trailing zeros as **z** until the end of the ids. Finally, return $\mathbf{2}^{z+1/2}$. **(10)**

2. We will use random projection on the KDD Cup dataset. Fetch this dataset from sklearn using the following command in your python code
from sklearn.datasets import fetch_kddcup99. Compute the number of samples as **n** and number of features as **d** from the data (i.e., $\mathbf{D} \in \Re^{n \times d}$).

(a) Define a family of JL matrices of size $\mathbf{d} \times \mathbf{20}$. Let matrix $\mathbf{M} \in \Re^{d \times 20}$ be a uniformly random sample from the family. Notice that **M** is a linear transformation that maps a vector from $\Re^d$ to $\Re^{20}$. Compute a mapping of every point in **D** as $\mathbf{E = DM}$ and compute k-means clustering, using $\mathbf{k = 15}$. Let **A** be the centroids of DM and B be the centroids of D. Compaute the loss of the k-means clusteiring on **D** using **A** and **B** respectively. Repeat this experiment 5 times for each random matrix **M**. Present your result in a table or bar graph. **(10)**

(b) We will run a linear regression on the dataset. Consider the label feature as the target/response as **y**. It has 23 classes. Define a family of JL matrices of size $\mathbf{10} \times \mathbf{n}$. Notice that the matrix-matrix between a sample from the JL family and **D** could be very expensive. So, you may use a sparse JL matrix as shown in the class. Let matrix $\mathbf{M} \in \Re^{10d \times n}$ be a uniformly random sample from the sparse JL family. Compute $\mathbf{E = MD}$ and $\mathbf{z = My}$ and solve linear regression on **(E, z)**. Also, solve the linear regression on **(D,y)**. Let **a** be the solution of **(E, z)** and **b** be the solution of **(D,y)**. Compute the loss of the linear regression on **(D,y)** using **a** and **b**, respectively. Repeat this experiment 5 times for each random matrix **M**. Present your result in a table or bar graph. **(10)**