

**PROJECT REPORT**

**PRIVACY PRESERVING K-MEANS CLUSTERING**

**CSE - 664**

**APPLIED CRYPTOGRAPHY AND COMPUTER SECURITY**

Submitted by:

SUDHARCHITH SONTY

UBID:50169912

UBMAIL: [sudharch@buffalo.edu](mailto:sudharch@buffalo.edu)

**Problem Statement:**

* The primary task in data mining is the development of models about aggregated data, extracting implicit un-obvious patterns and relationships from a warehouse of data sets. This information can be useful to increase the efficiency of the organization and aids in future planning.
* However, of late, internet phishing caused significant security and economic concerns on the users and enterprises worldwide. Diversified communication channels via internet services such as electronic commerce, online-banking, research, and online trade exploiting both human and software vulnerabilities suffered from tremendous financial loss.
* Some of the concerns that provide the motivation for privacy preserving data mining solutions are listed below:
  + - Privacy Concerns
    - Proprietary information disclosure
    - Concerns about Association breaches
    - Misuse of mining
* Preservation of privacy in data mining has emerged as an absolute prerequisite for exchanging confidential information in terms of data analysis, validation, and publishing. Ever-escalating internet phishing posed severe threat on widespread propagation of sensitive information over the web.
* Therefore, enhanced privacy preserving data mining methods are ever-demanding for secured and reliable information exchange over the internet. The dramatic increase of storing customers’ personal data led to an enhanced complexity of data mining algorithm with significant impact on the information sharing.
* Amongst several existing algorithms, the Privacy Preserving Data Mining (PPDM) renders excellent results related to inner perception of privacy preservation and data mining.
* So, the question is; Can we develop accurate models without access to precise information in individual data records? And if we do so, some of the aspects that are to be kept in mind are as follows:
  + - Restrict Access to data (Protect Individual records).
    - Protect both the data and its source.
* Truly, the privacy must protect all the three mining aspects including association rules,

classification, and clustering.

**Literature Survey:**

* Present literature presents classifiers which are trained on unencrypted datasets and subsequently classify encrypted data points.
* The paper presented by **Stephen Tu, Shafi Goldwasser, Raphael Bost, Raluca Ada Popa;** titled: **Machine learning classification1** over encrypted data adopts such an approach for hyperplane, naive Bayes and decision tree classifiers.
* Work has also been done on securing data for a classifier in distributed settings, in which individual worker nodes are treated as untrusted.
* The approach presented by **L. Guo, Y. Guo, Y. Fang, K. Xu, H. Yue: Privacy-preserving machine learning algorithms for big data systems2** uses the MapReduce framework for this approach.
* Applying neural networks on homomorphic encryption schemes has been studied in some detail. Since fully homomorphic encryption is slow, the approach suggested **by Pengtao Xie et. al. Crypto-nets: Neural networks over encrypted data**3 proves that homomorphic encryption that applies only to degree bounded polynomials is a viable alternative. However, the speed and practicality of this method is still unclear.
* The epsilon-differential privacy model promised in the method proposed by **Kamalika Chaudhuri and Claire Monteleoni: Privacy-preserving logistic regression; Advances in Neural Information Processing Systems4** also provided interesting directions of research. This paper provides strong guarantees on securing a logistic regression model.
* There are also several papers examining the security a machine learning system against adversarial attacks to further strengthen the system.

**Literature Involved:**

* A public key encryption scheme is a set of three functions G, E, and D. The function G is a key generation function and when G is called with a random argument it generates a key-pair: (Pk, Sk) = G(r), where Pk, is called the public key, and Sk is called the secret key.
* The two keys satisfy the following decryption condition: D (Sk, c) = m, where c = E(Pk, m, r) is called the cipher text, m is called the message or plaintext, and r is a random number. Furthermore, it is computationally infeasible to compute the message m when given only Pk and E (Pk, m, r).
* An encryption scheme is said to be additively homomorphic if E (Pk, m0, r) E (Pk, m1, r0 ) = E(Pk, m0 + m1, r00), for some value r00.
* A (t, n) secret sharing scheme is a set of two functions S and R. The function S is a sharing function and takes a secret s as input and creates n secret shares: S(s) = (s1, . . ., sn). The two functions satisfy that for any set I ⊆ {1, . . ., n} of t indices R (I, sI1, . . ., sIt) = s.
* Furthermore, we require that it is impossible to recover s from a set of t − 1 secret shares. A secret sharing scheme is additively homomorphic if R (I, sI1 + s 0 I1, . . ., sIt + s 0 It’s + s0.
* A very simple (n, n) secret sharing scheme which is additively homomorphic is S(s) = (r1, . . ., rn−1, r), where ri is random for i ∈ {1, . . ., n − 1}, and r = s − Pn−1 i=1 ri. To recover s all secret shares are added: s = r + Pn−1 i=1 ri.
* An encryption scheme is additively homomorphic if there is some operation ⊗ on encryptions such that for all clear text values a and b, E(a)⊗E(b) = E(a+b).

**Security Model:**

* The algorithm is a simple deterministic algorithm, Re-cluster, for I/O-efficient k-clustering. It examines each data item only once and uses only sequential access to the data.
* Also, we scale it to a privacy-preserving version of the Re-cluster algorithm, for two-party horizontally partitioned databases. This protocol reveals the cluster centers (or the cluster assignments to data, if both parties desire) to both parties only at the end of the protocol.
* Two parties, Alice and Bob, own databases D1 = {d1, . . ., dm} and D2 = {dm+1, . . ., dn}, respectively. They wish to jointly compute a k-clustering of D1 ∪ D2 492 (that is, the data is horizontally partitioned).
* Both parties learn the final k cluster centers, and nothing else. Alternatively, with additional computation and communication, each party could learn the cluster to which each of their data objects belongs.
* If there were a trusted third party to whom Alice and Bob were both willing to send their data, this party could then compute the clustering and send the cluster centers to Alice and Bob.

**Objectives and Goals:**

* Secure multiparty computation protocol that can carry out the required computation without requiring a trusted third party, which is also an I/O efficient protocol and should be scalable, across horizontally partitioned databases.
* The protocol should provide the same amount of security as provided while using a trusted third party.

**Assumptions:**

* We assume that Alice and Bob are semi-honest, meaning that they follow their protocol as specified, but may try to use the information they have learned (such as messages they receive) to infer information about the other party’s data.
* We say that Alice and Bob have random shares of a value x drawn from a field F of size N (or simply Alice and Bob have random shares of x) to mean that Alice knows a value a ∈ F and Bob knows a value b ∈ F such that (a + b) mod N = x, where a and b are uniformly random in field F.
* We assume that a finite field F of a sufficiently large size N is chosen such that all computations can be done in that field, and all computations throughout the remainder of the paper take place in F. We assume all protocols used are Semantically Secure.

**The K-Means Clustering Algorithm (Divide And Conquer):**

* The algorithm runs in the typical “divide, conquer and combine” fashion. This strategy would require us to divide the database into two equal halves, recursively produce k cluster centers from each of the halves, and then merge these 2k centers into the k final centers.
* However, we take a slightly different tack. We produce 2k cluster centers from each recursive call, and then merge the total of 4k centers thus received (by the two recursive calls at each level) into 2k centers.
* Finally, we use the same merge technique to produce the k final centers from the 2k clusters returned from the top-most level of the recursion tree.
* The key step is the merging of 4k centers into 2k centers after the two recursive calls (Merge Centers). We do this by repeatedly choosing a best pair of clusters Ci and Cj for merging, and replacing them in the clustering with Ci ∪ Cj
* A best pair of clusters is one with least error. We use a variation of the notion of error defined in Ward’s algorithm. Let C1 and C2 be two clusters being considered for a merge.
* Let C. weight denote the number of objects in cluster C. In Ward’s Algorithm, the error of C1 ∪ C2 is errorw(C1∪C2) = , where dist. (C1, C2) is the distance between the centers of C1 and C2.
* We define the error as errorr (C1 ∪ C2) = C1. weight ∗ C2. weight ∗ dist2 (C1, C2).

|  |
| --- |
| **Algorithm Re-cluster**  **Input:** Database D, Integer k  **Output:** Cluster centers S   * S ′ = Recursive Cluster (D, 2k) // Produce 2k clusters * S = Merge Centers (S ′, k) // Compress to k clusters   Output S |
| **Subroutine Recursive Cluster**  **Input:** Database D, Integer k  **Output:** Cluster centers S  If (|D| ≤ k) then S = D Else   * D1 = First half of D * D2 = Second half of D * S1 = Recursive Cluster (D1, k) * S2 = Recursive Cluster (D2, k) * S = Merge Centers (S1 ∪ S2, k)   Output S |
| **Subroutine Merge Centers**  **Input:** Cluster centers S, Integer k  **Output:** Cluster centers S, such that |S| = k While (|S| > k)   * Compute the merge error for all pairs of centers in S. * Remove from S the pair with the lowest merge error, and insert the center of the merged cluster, with its weight as the sum of the weights of the pair. |

**Cryptographic Design:**

* Alice computes k cluster centers for her data D1 = {d1, . . ., dm} and Bob computes k cluster centers for his data D2 = {dm+1, . . ., dn}. Alice and Bob randomly share these cluster centers with each other.
* Bob chooses a random permutation φ1 and random values ri, si, 1 ≤ i ≤ k and using secret share they obtain random shares of Alice’s k-cluster centers.
* Alice chooses a random permutation φ2 and random values pi, qi, 1 ≤ i ≤ k and using secret share they obtain random shares of Bob’s k-cluster centers.
* The clusters are classified as vectors that contain tuples (C, W), where C is the cluster center and W is the weight of the cluster or the number of vectors that are a part of it.
* Bob chooses a random φ3 and random values αi, βi 1 ≤ i ≤ 2k and using secret share they obtain random shares of Alice’s k-cluster centers. Bob adds the shares just obtained to his data.
* Alice chooses a random permutation φ4 and random values γi, δi 1 ≤ i ≤ 2k and using secret share they obtain random shares of Bob’s k-cluster centers. Alice adds the shares just obtained to her data.
* Alice and Bob repeat the protocol to securely merge clusters k times to merge 2k clusters into k clusters

**Algorithm Flow:**

**The Paillier Cryptosystem:**

* We choose two random prime numbers p and q such that gcd (pq, (p-1) (q-1) ) is 1
* Compute two values n =pq and ƛ = (p-1) (q-1)
* We assume that p and q are of equal length.
* Now we select an Integer g = n +1, μ = [(p-1) (q-1)]-1
* The public encryption key is (n, g)
* The private decryption key is (ƛ, M)
* L is a predefined function L(x) =

### Encryption

* Let {\displaystyle m}mm be a message to be encrypted where {\displaystyle m\in \mathbb {Z} \_{n}}c ∈ Ζ\*n2
* Select random {\displaystyle r}r where {\displaystyle r\in \mathbb {Z} \_{n}^{\*}}r ∈ Ζ\*n
* Compute cipher text as: {\displaystyle c=g^{m}\cdot r^{n}{\bmod {n}}^{2}}gm. rn.mod n2

### Decryption

* Let {\displaystyle c}c be the cipher text to decrypt, where {\displaystyle c\in \mathbb {Z} \_{n^{2}}^{\*}} c ∈ Ζ\*n2
* Compute the plaintext message as: {\displaystyle m=L(c^{\lambda }{\bmod {n}}^{2})\cdot \mu {\bmod {n}}}m = L (cƛ mod n2). μ mod n

**Shamir Secret Sharing:**

* A (t, n) secret sharing scheme is a set of two functions S and R. The function S is a sharing function and takes a secret s as input and creates n secret shares: S(s) = (s1, . . ., sn). The two functions satisfy that for any set I ⊆ {1, . . ., n} of t indices R (I, sI1, . . ., sIt) = s.
* Furthermore, we require that it is impossible to recover s from a set of t − 1 secret shares.
* Knowledge of any t or more Si pieces makes S easily computable.
* Knowledge of any t-1 or fewer Si pieces’ leaves S completely undetermined (in the sense that all its possible values are equally likely

**Uml Diagrams:**

****

**Class Diagram**



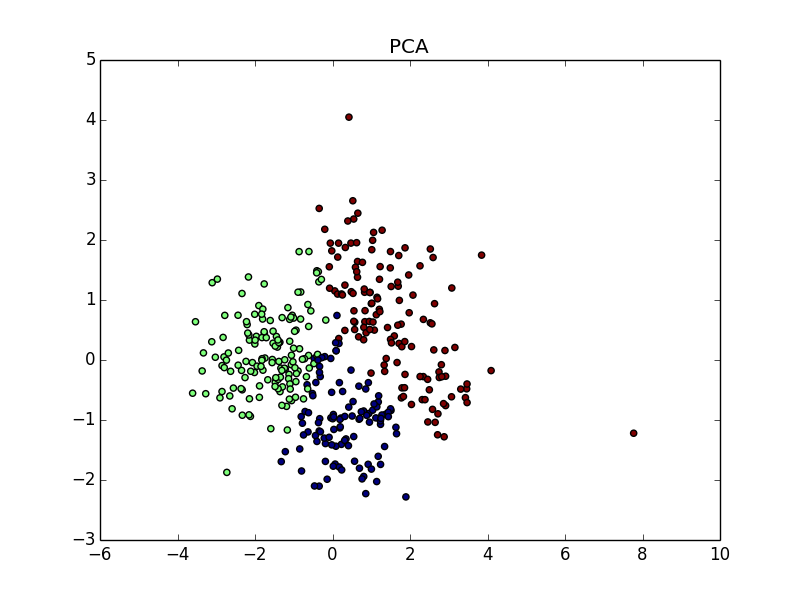
**Usecase Diagram**

**Project Setup:**

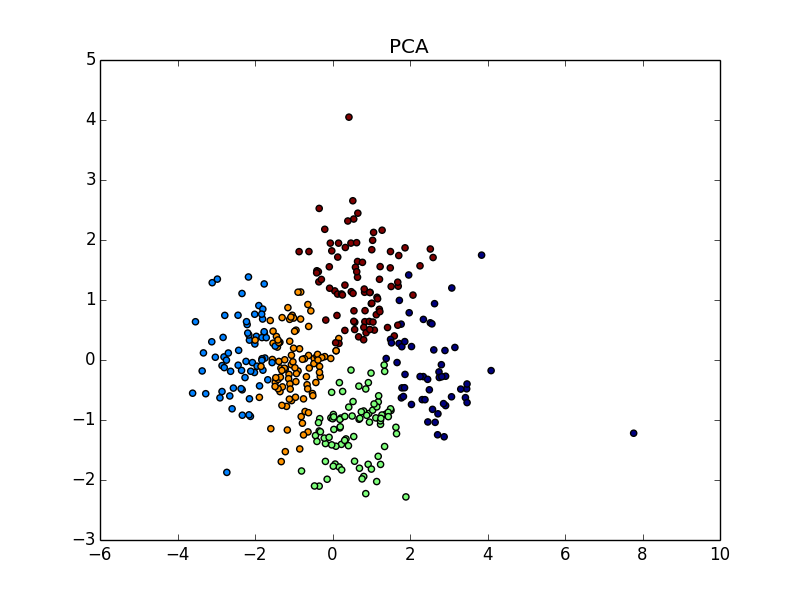
* The algorithm was run on a Mac Book Pro 2012 Model carrying an Intel Core i5 – First Generation, 16 GB of DDR3 RAM running MacOS Sierra.
* The entire code was written in JAVA, JSP, JavaScript and some amount of Python; the IDE used was Eclipse.

**Results of The Clustering Algorithm:**

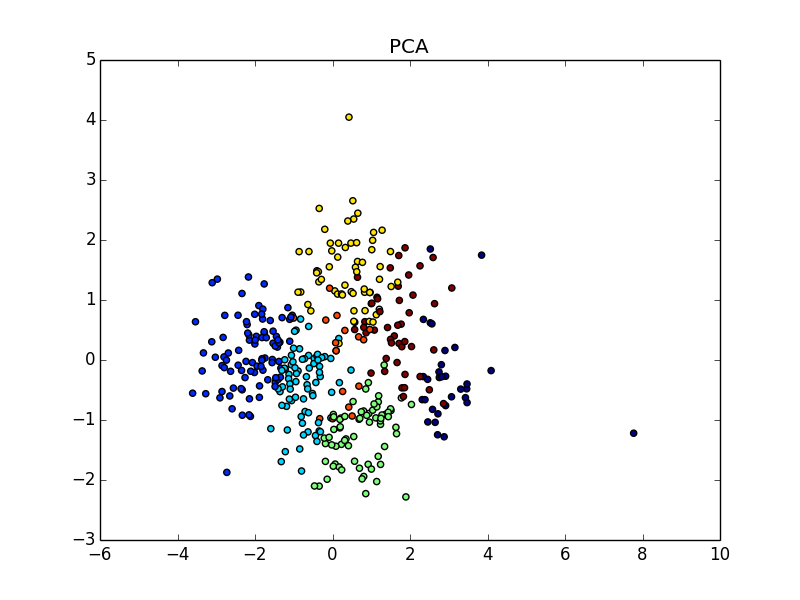
**k =3**

****

**k = 5**

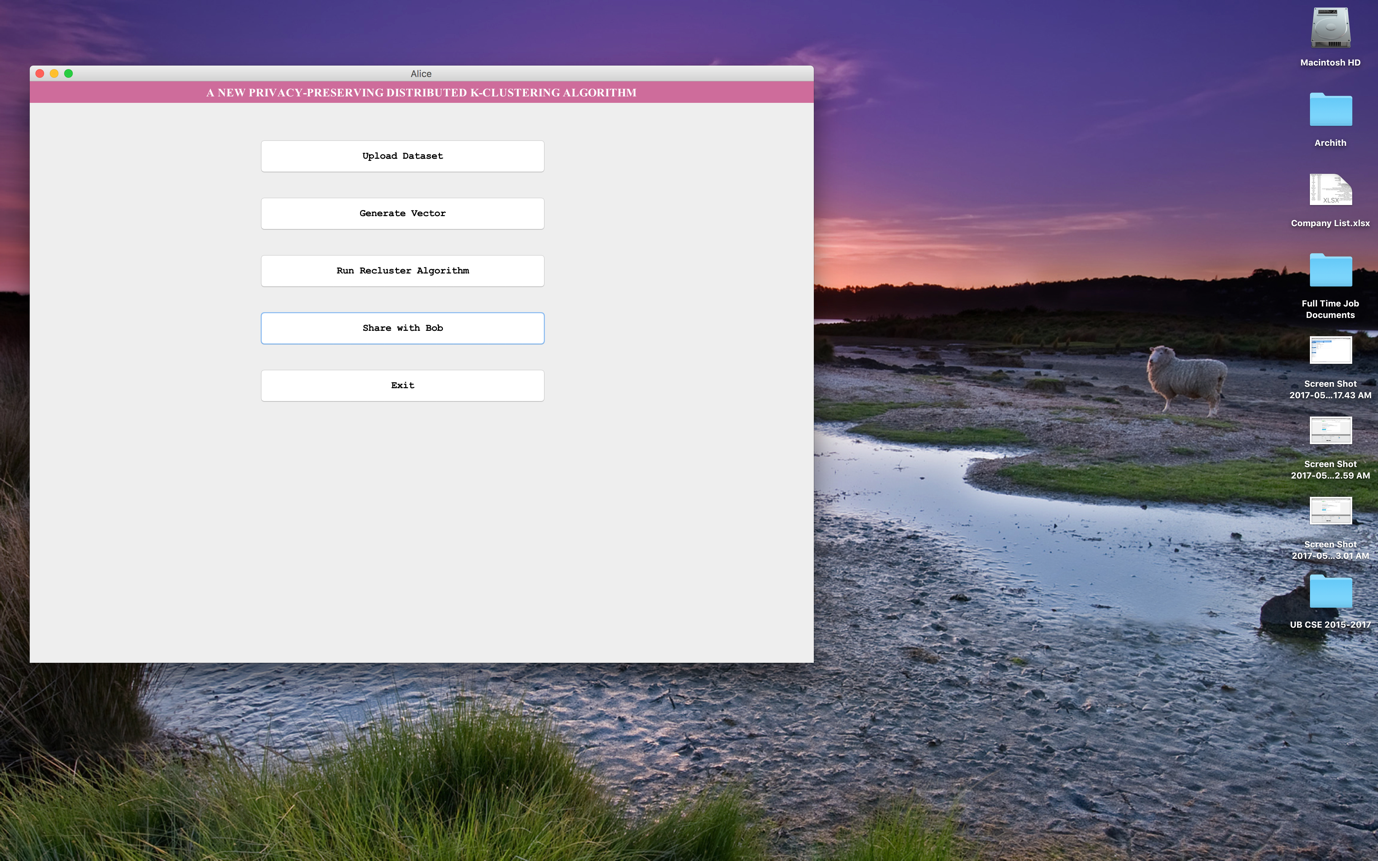
****

**k = 7**

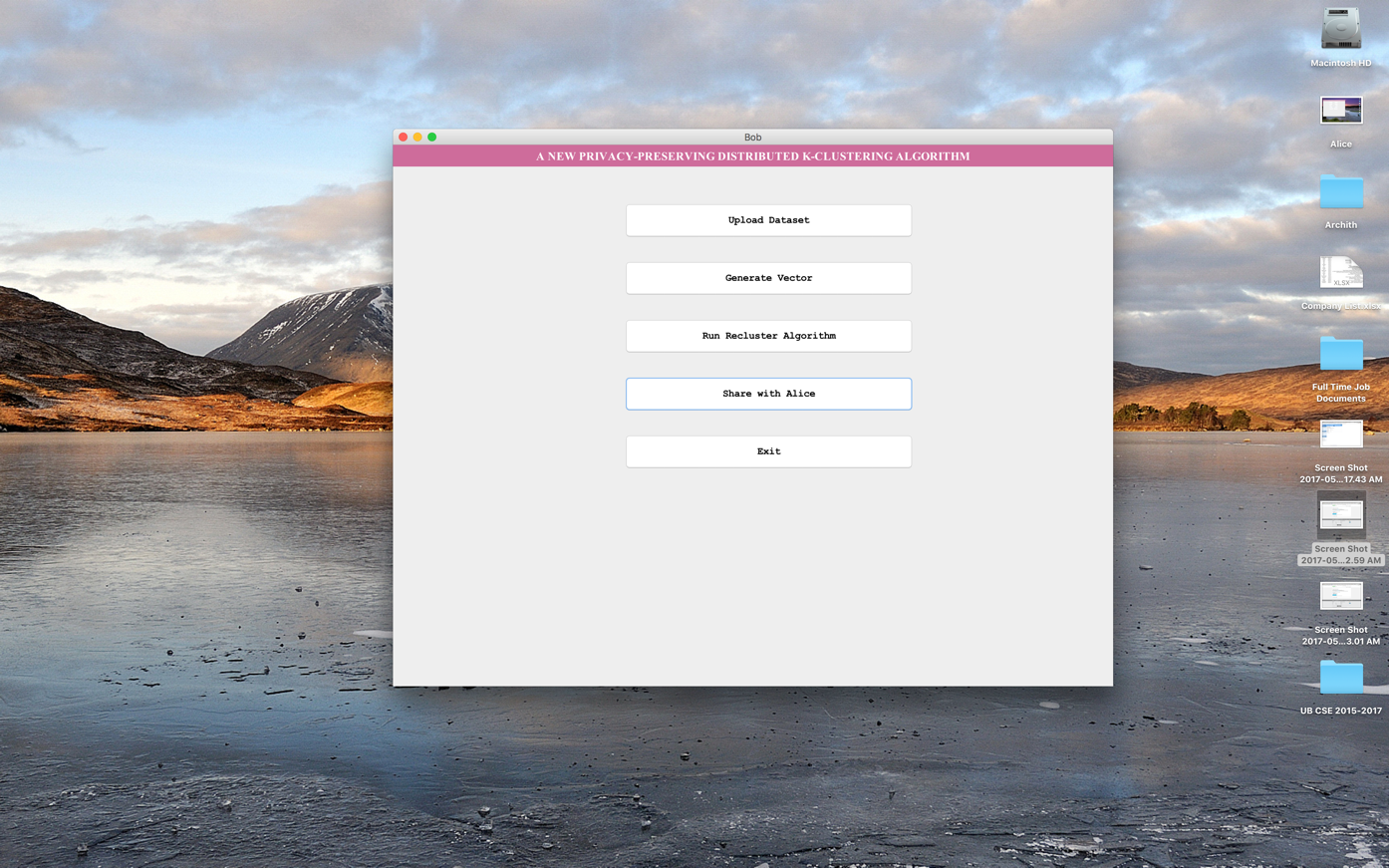
****

**Application Results:**

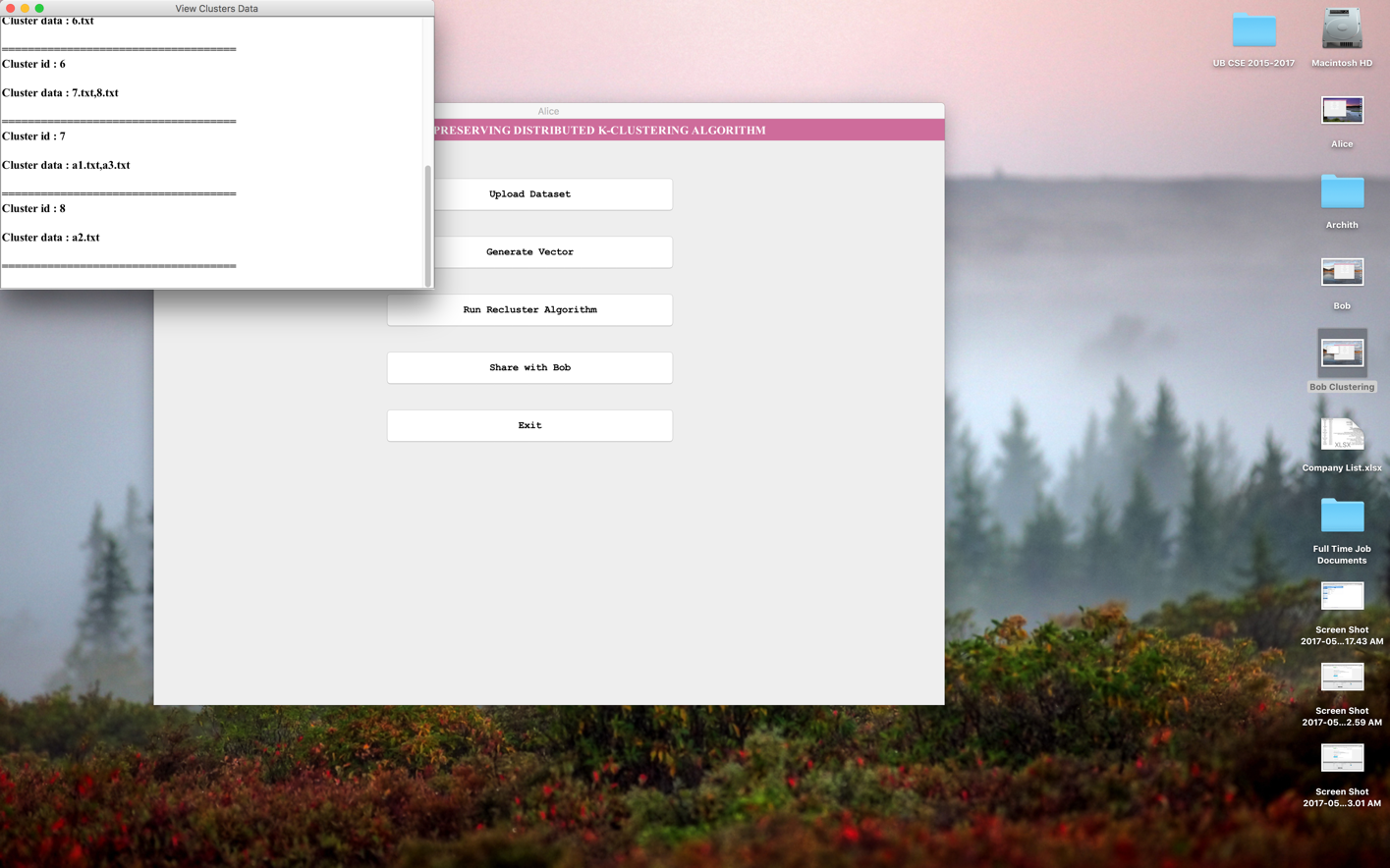
**Alice Home Screen**

****

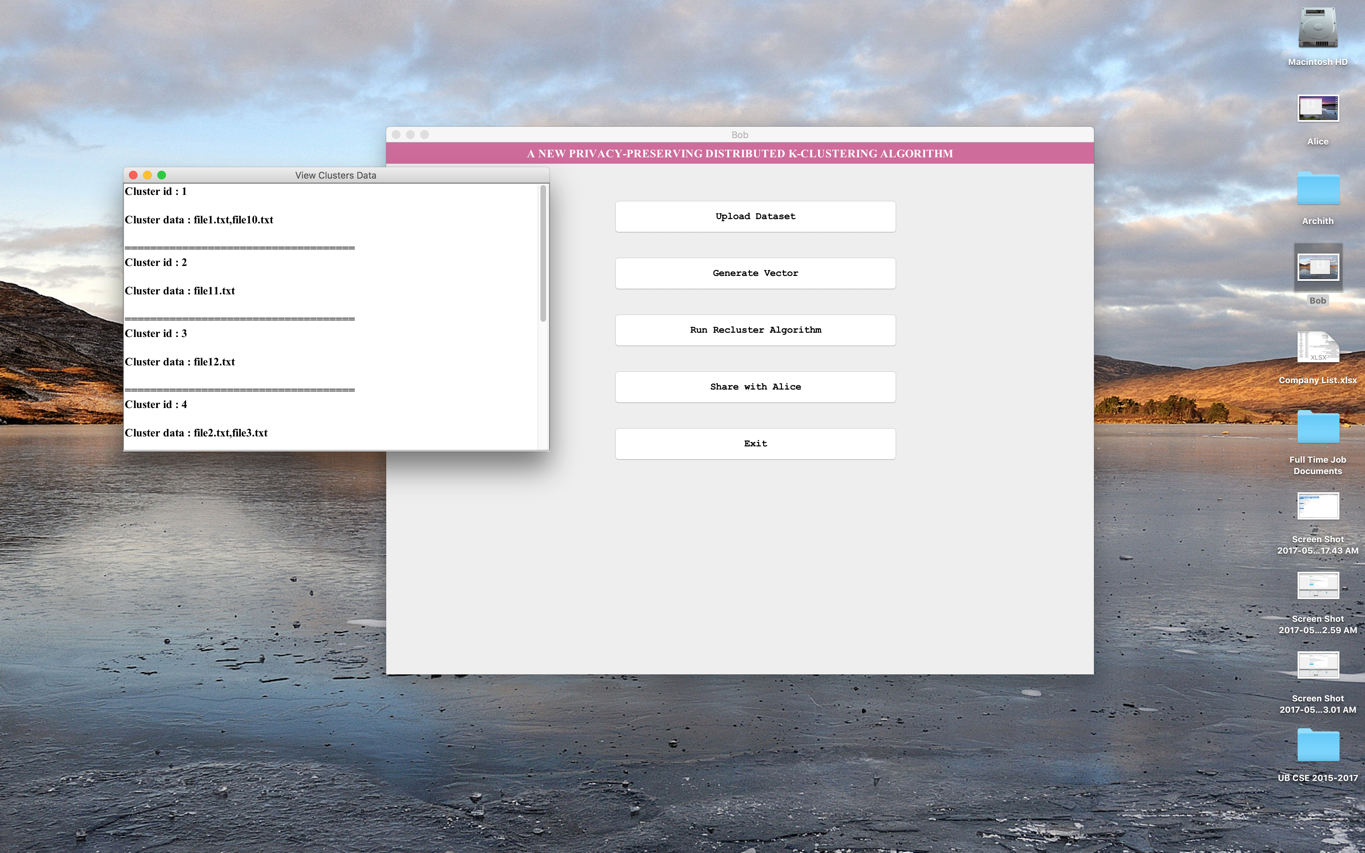
bob Home Screen

****

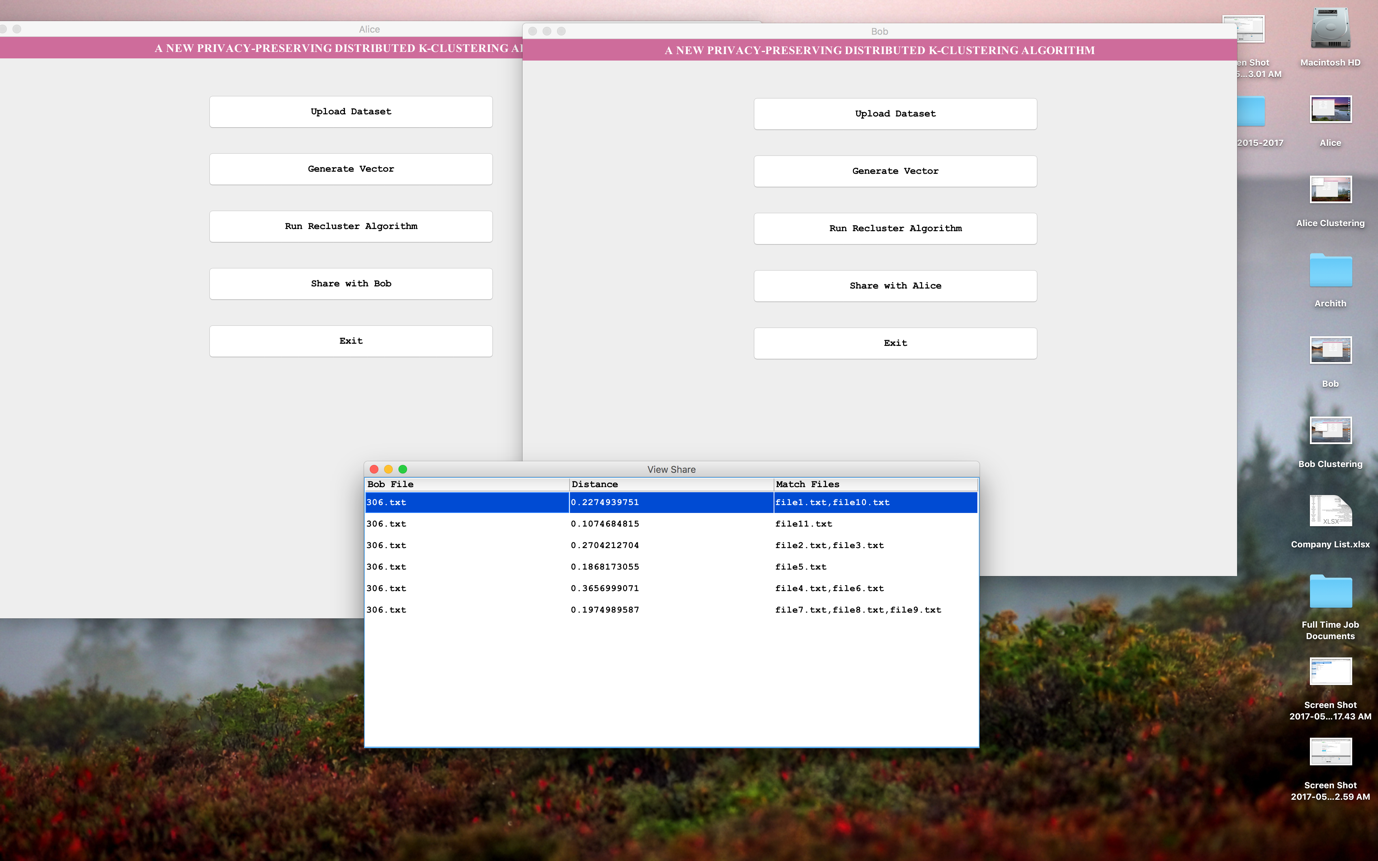
**Alice Custering**



**bob clustering**



Merging



Performance:

* The overall computational complexity of the privacy-preserving version of the protocol is O(k3ℓ) encryptions, and O(nk3ℓ) multiplications for Alice and O(k3ℓ) exponentiations, O(k2) encryptions and O(nk3ℓ) multiplications for Bob.
  + k denotes the number of clusters,
  + n denotes the size of the data
  + ℓ denotes the number of attributes in the data
  + c denotes the maximum number of bits for an encryption.
  + The communication complexity is O(k3cℓ) bits, which does not depend on n.
* For fixed k, Re-cluster runs in O(n) time and uses O(logn) space.

**Results with Respect to The Objectives:**

* The K – Clustering algorithm ran perfectly for all input data sets without any problems.
* The communication between the two parties was established using a secure Homomorphic encryption scheme.
* The secret sharing scheme could not be implemented as given in the objectives and a work around had to be implemented.

**Difficulties and Work Around:**

* The given permute share protocol, could not be implemented due to computational complexity and compatibility issues and hence a simpler Adi Shamir Secret Sharing Scheme was employed. Everything else is as per the desired implementation.
* Due to the structure of the code , an incorporation of Shamir’s secret sharing algorithm into the code could not be done and hence, a simple end to end public key encryption with a private key decryption was implemented using a Homomorphic Encryption scheme.
* The Paillier Cryptosystem was used to perform the encryption.

**Testing:**

* Testing for the K-means algorithm was done on multiple datasets. Performance was evaluated based on the Jaccard Similarity score which was always above 0.5.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | | **Test Case Status** | | **Test Priority** | |
| **Step** | **Expected** | **Actual** |  | |  | |
| 01 | Upload dataset | To check whether dataset is uploaded or not | Without upload dataset | We cannot run the re clustering algorithm | Loaded dataset | High | | High | |
| 02 | Generate vector | Verify whether vector generated or not | Without generates vector | We cannot run the re clustering algorithm | Vector generated | High | | High | |
| 03 | Run Re cluster algorithm | To check whether running Re cluster algorithm or not | Without generate vectors | We cannot run the re clustering algorithm | View clusters data | High | | High | |
| 04 | Share | To check whether sharing files or not | If file does not share with Bob or Alice | It cannot display share data | View share data (i.e. view distance and view match files) | High | | High | |

**Security Guarantee:**

* When the parties like to compute the cluster centers they exchange their shares to each other. All the intermediate results are not available between the two parties.
* The protocol achieves the same privacy as when a trusted third party is used.
* Both parties cannot learn any information from the encrypted messages that are communicated since the encryption scheme chosen is semantically secure.
* The protocols are semantically secure. They do not leak any information. The cluster centers at the end of the merge clusters protocol is obtained as a random share between the two parties.
* Additional complexity and computation can be used to achieve specific results as to which object belongs to which data party and so.

Description of data sets:

* The data sets used to run the K-means algorithm are sets of multi-dimensional vectors that constitute genomic data.Other datasets have been downloaded from the Internet and the algorithm has been run on them as well.
* The data used for the privacy preserving version of the algorithm is a set of arbitrary files that contain random strings as messages.

**References:**

**main paper for algorithm**

* [**http://www.siam.org/meetings/sdm06/proceedings/048jagannag.pdf**](http://www.siam.org/meetings/sdm06/proceedings/048jagannag.pdf) **- A New Privacy Preserving Algorithm by Geeta Jagannathan and Rebecca N.Wright.**

**Other References:**

* Privacy Preserving Data Mining - Cynthia Dwork and Frank McSherry

<http://web.stanford.edu/group/mmds/slides/mcsherry-mmds.pdf>

* Privacy Preserving Data Mining – Yehuda Lindell, Benny Pinkas

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.124.9334&rep=rep1&type=pdf>

* A comprehensive review on privacy preserving data mining by Yousra Abdul, Alsahib S. Aldeen, Mazleena Salleh and Mohammad Abdur Razzaque

<https://springerplus.springeropen.com/articles/10.1186/s40064-015-1481-x>

* Privacy Preserving Data Mining – Moheeb Rajab

<http://www.cs.jhu.edu/~fabian/courses/CS600.624/slides/privacy-preserving.pdf>

* Privacy Preserving Data Mining – Rakesh Agarwal, Ramakrishnan Srikant

<https://www.utdallas.edu/~muratk/courses/privacy08f_files/agrawal00privacypreserving.pdf>

* <http://www.cs.tau.ac.il/~fiat/crypt07/papers/Pai99pai.pdf> - The Paillier Cryptosystem
* <http://www.cs.ucsb.edu/~veronika/MAE/summary_clusteringdatastreams_theorypractice_guha.pdf>
* <https://pdfs.semanticscholar.org/0ab4/26e3244128fb1cf717db2d65cbb9d41dad0e.pdf>
* <http://www.carch.ac.cn/~huimin/papers/CF.pdf>
* R. Agrawal and R. Srikant. Privacy-preserving data mining. In ACM SIGMOD, pages 439–450, May 2000.
* W. Du and M. Atallah. Privacy-preserving cooperative statistical analysis. In 17th ACSAC, pages 102–112, 2001.
* G. Jagannathan and R. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In 11th KDD, pages 593–599, 2005.