PROJECT PROGRESS REPORT
PRIVACY PRESERVING K-MEANS CLUSTERING
CSE 664
APPLIED CRYPTOGRAPHY AND COMPUTER SECURITY

Submitted by:
SUDHARCHITH SONTY
UBID:50169912
UBMAIL: sudharch@buffalo.edu

ORIGINAL PROJECT PROPOSAL:

- An implementation of Randomization on a given data set using the Value Distortion Method proposed.
- An implementation of the Decision Tree Algorithm over the said randomized data.
- An implementation of Distributed K-Means with encryption using a common Public Key.

CURRENT PROJECT PROPOSAL:

- The project proposal now comprises of an implantation of the Distributed K-Means algorithm with a single public key for encryption.
- An equally contributed multiparty k-means clustering is applied on vertically partitioned data, wherein each data site contributed k-means clustering evenly.
- As per to the basic concept, data sites collaborated to encrypt k values (each associated to a distance between the centre and point) with a common public key in each step of clustering.
- Then, it securely compared k values and outputted the index of the minimum without displaying the intermediate values. In some setting, this is practical and more efficient than Vaidya–Clifton protocol.
- We concentrate on a distributed scenario where the data is partitioned vertically over multiple sites (different attributes for the same entity can be stored at different sites) and the involved sites would like to perform clustering without revealing their local databases. For this setting, we use a protocol for privacy preserving k-means clustering based on additive secret sharing.
- In this case, each site has a different projection of the database. We choose the popular k-means clustering algorithm and propose a new protocol for distributed privacy preserving k-means clustering. Instead of using computationally costly public key encryption schemes, we utilize additive secret sharing as a cryptographic primitive to implement a secure multiparty computation protocol to do privacy preserving clustering.

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: $(P_k, S_k) = G(r)$, where P_k, is called the public key, and S_k is called the secret key.
- The two keys satisfy the following decryption condition: $D(S_k, c) = m$, where $c = E(P_k, m, r)$ 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 $E(P_k, m, r)$.
- An encryption scheme is said to be additively homomorphic if $E(P_k, m0, r)\ E(P_k, m1, r0) = E(P_k, m0 + m1, r00)$, for some value r00.

- The well-known Paillier homomorphic encryption scheme takes messages of size n bits, and creates 2n-bit cipher texts. In practical applications, messages are of 32 bits, so a cipher text is approximately 64 times larger than the message.

- 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.

- If even one secret share is missing nothing is known about s. We use this simple additive secret sharing scheme in this paper. Notice that the communication cost when secret sharing a number between n parties is exactly n−1 (one random number send to each of the other n − 1 parties).

RELATED WORK:

- 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 classification**[1] 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 systems**[2] 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 Systems**[4] 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.

IMPLEMENTATION DETAILS:

- Homomorphic encryption is used to do a secure computation of the closest cluster, whereas we use secret sharing.

- The power of secret sharing in this setting is that communication overheads are considerably lower. Our algorithm performs distributed k-means clustering with r parties. The data is vertically partitioned such that each of the r parties has some of the attributes of the dataset. The number of entities in the dataset is n.

- The goal of the r parties is to perform k-means clustering on their aggregated data without revealing the values of the attributes they own to the other parties. The algorithm will divide the entries into k clusters and each party learns the cluster means corresponding to their own attributes, and the index of the cluster into which each entity is assigned.

- Ideally, no party should learn anything else than this. Let μc, c ∈ {1, ..., k}, represent the cluster means of the result. Let μci be the projection of cluster mean c onto the attributes of party i (μc = (μc1, . . ., μcr)). As output of privacy preserving k-means clustering party i gets: • The final mean μci for each cluster c ∈ {1, . . ., k}. • The cluster index for each entity j ∈ {1, ..., n}.

- The public key encryptions are the bottleneck of the method described. In contrast, we use additive secret sharing to achieve privacy, which gives us lower computation and communication overhead].

- Our algorithm follows the standard k-means clustering method. Firstly, initial cluster means are selected, and all entities in the dataset are assigned to the closest initial clusters. After the initial assignment of clusters, the cluster means are recalculated and each entity is reassigned to the cluster with the closest cluster mean. The process continues until a termination criterion is met.

- The algorithm that is being used terminates when there is no change in the assignment of clusters. The algorithm presented by Clifton and Vaidya terminates when the change in cluster means between two iterations is less than a given threshold. Our termination criterion corresponds to setting the threshold to 0.

- The security of our algorithm relies on three ideas:

    - Each party sends secret shares its sub-distance to all the other parties, and the sum of the sub-distances is computed on the secret shares (where [12] adds random blinding values in a protocol which applies encryption).

    - The comparison of distances is performed on secret shares so that only the comparison result is learned. The actual values of the distances are not learned.

    - The ordering of clusters is permuted so that for each entity in the database, only the index of the closest cluster is learned. Relative orderings of the entity's distances to each cluster mean μc cannot be learned. This is very simple when we work with secret shares.

FUNCTIONALITY IMPLEMENTED:

- Implemented the basic K-means algorithm on a large dataset and verified some of the results.
- Also, implemented the K – means algorithm on a single node Hadoop cluster using Map-Reduce.
- Read through the Paillier Encryption scheme and implemented a rough version of it.

CHALLENGES ENCOUNTERED:

- Problems were faced in trying to run the Map Reduce on a multi-node Hadoop cluster and trying to get both the nodes to communicate with each other. (Unresolved).
- Trying to get the dataset distributed vertically across the multi-node cluster. (Unresolved).
- Had issues figuring out the right encryption scheme to reduce overhead (Resolved).

FUNCTIONALITIES TO BE IMPLEMENTED

- The project is still in the research stage, with an estimated 10-15% of the project being completed.
- Creating multiple parties and establishing communication protocols between the parties facilitating the sharing of the intermediate outputs produced by each distributed data set.
- Perfecting the encryption scheme being used in the algorithm to ensure safety.

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

ARTICLES READ AFTER SUBMISSION OF PROJECT PROPOSAL:

- Distributed Privacy Preserving k-Means Clustering with Additive Secret Sharing*
  http://cscdb.nku.edu/pais08/papers/p-pais02.pdf

- Privacy Preserving K-Means Clustering
  https://courses.csail.mit.edu/6.857/2016/files/8.pdf

- A New Privacy-Preserving Distributed k-Clustering Algorithm
  http://www.siam.org/meetings/sdm06/proceedings/048jagannag.pdf

- Privacy-Preserving and Outsourced Multi-user K-Means Clustering
  http://ieeexplore.ieee.org/document/7423068/?reload=true

- Privacy-preserving distributed k-means clustering over arbitrarily partitioned data
  http://dl.acm.org/citation.cfm?id=1081942

- Privacy-Preserving Decision Trees over Vertically Partitioned Data
  http://www.utdallas.edu/~muratk/publications/vaidya-clifton-kantarcioglu-patterson.pdf

APPENDIX:

1.) Stephen Tu, Shafi Goldwasser, Raphael Bost, Raluca Ada Popa, Machine learning classification - http://www.icmlc.org/icmlc2009/076_icmlc2009.pdf
2.) L. Guo, Y. Guo, Y. Fang, K. Xu, H. Yue: Privacy-preserving machine learning algorithms for big data systems - http://ieeexplore.ieee.org/document/7164918/
3.) Pengtao Xie et. al. Crypto-nets: Neural networks over encrypted data- https://arxiv.org/abs/1412.6181
4.) Kamalika Chaudhuri and Claire Monteleoni: Privacy-preserving logistic regression; Advances in Neural Information Processing Systems - https://papers.nips.cc/paper/3486-privacy-preserving-logistic-regression