CS 550 Massive Data Mining & Learning (MDML)
Final project
Team Number 30 (ec1005, am2971, rkd55)

4.(d) Optional Task: Trustworthiness of Recommendation Systems

Primary Topic: Privacy Protection For Recommender Systems
Project Sub Topic: **Differential Privacy In Recommender Systems**

Primary Research Paper: Pursuing Privacy in Recommender Systems: the View of Users and Researchers from Regulations to Applications (Belli et. al., 2021)

Primary GitHub Repository Referenced:
https://github.com/sisinflab/recsys2021-pursuing-privacy/blob/master/notebooks/1_dp_matrix_factorization.ipynb

Introduction

The Netflix Prize data privacy fiasco was perhaps one of the first incidents that brought online user data privacy to the limelight when a group of scientists in 2010 accurately de-anonymized the Netflix publicly released dataset by leveraging the IMDB data. Moreover it was found from another study that 87% of all Americans can be identified from the US Census data alone by knowing only their gender, zip-code and birth date. Moreover, public awareness and new regulations forced technology researchers and practitioners to study solutions to user privacy endangerment.

Recommendation systems have become increasingly popular in recent years, as they provide personalized suggestions to users for products, services, or content based on their past behavior and preferences. However, with the rise of privacy concerns, the use of recommendation systems has also come under scrutiny. Recommendation systems are often accused of being intrusive and violating users' privacy by collecting and analyzing their personal data, such as browsing history, search queries, and social media activity, to provide targeted recommendations. This has led to a growing need to balance the benefits of recommendation systems with the potential risks to user privacy, and to develop new approaches that address these issues. In this context, it is important to understand the different types of privacy issues that can arise in recommendation systems, as well as the challenges and opportunities associated with developing privacy-preserving recommendation algorithms.

Anonymizing data is a common technique used to protect privacy by removing personal information from a dataset. However, anonymization alone does not guarantee differential privacy, which is a stronger notion of privacy protection that ensures that an individual's data cannot be linked to their identity, even when combined with other external data sources.

The reason anonymization is not sufficient for preserving differential privacy is that it relies on the assumption that the dataset is sufficiently large and diverse, making it difficult for an adversary to identify a specific individual within the dataset. However, recent studies have shown that anonymized data can still be vulnerable to privacy attacks, such as linkage attacks, where an adversary combines multiple anonymized datasets to re-identify individuals. This means that even if an individual's data is anonymized, it may still be possible to infer sensitive information about them, violating their privacy.

In contrast, differential privacy offers a more rigorous approach to privacy protection by adding noise to the data before releasing it, ensuring that the probability of identifying a specific individual remains low, even if an adversary has access to external information. This makes it a more robust and reliable approach to privacy protection, even when dealing with small or highly sensitive datasets.

Differential Privacy

Differential privacy is a concept in the field of data privacy that provides a mathematical framework for measuring and quantifying the amount of privacy that is preserved when collecting and analyzing data. It is based on the idea of adding noise to data in such a way that the privacy of individuals whose data is being collected is protected, while still allowing useful insights to be derived from the data.

Differential privacy was first introduced in 2006 by Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. The concept has since gained widespread attention in both academia and industry, particularly in the context of large-scale data collection and analysis.

The basic idea of differential privacy is to add random noise to data in a way that makes it difficult to determine whether any particular individual is present in the data set. This is accomplished by adding noise to the data in a manner that preserves the statistical properties of the data, while masking any information about specific individuals.

Differential privacy is an important concept in the field of data privacy, as it provides a rigorous and quantifiable way to protect the privacy of individuals whose data is being collected, while still allowing useful insights to be derived from the data. It has many practical applications in fields such as healthcare, finance, and government, where the collection and analysis of large amounts of sensitive data is common.

Differential Privacy In Recommender Systems

Recommender systems are widely used in online platforms to suggest products, services, or content to users based on their past preferences or behaviors. However, collecting and analyzing user data in recommender systems raises privacy concerns, as users may not want their personal information to be shared or analyzed.

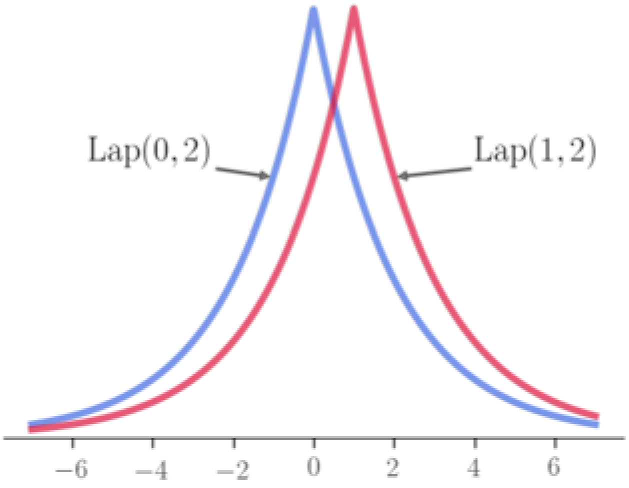
Differential privacy offers a way to address these privacy concerns in recommender systems by adding noise to the data used to make recommendations. By doing so, the privacy of individual users is protected while still allowing accurate and useful recommendations to be made.

One common approach to implementing differential privacy in recommender systems is to use randomized response techniques. For example, in a binary preference setting (where a user either likes or dislikes a product), instead of recording the exact preference of the user, the system can randomly flip the user's preference with a certain probability. The probability of flipping is determined by the desired privacy level, and it ensures that the recommendation algorithm cannot determine the true preference of any individual user with a high degree of certainty.

Another approach to differential privacy in recommender systems is to add noise directly to the model parameters used for making recommendations. This approach is called "local differential privacy" because the noise is added directly on the user's device or browser before being sent to the recommender system. By adding noise to the model parameters, the system can still make accurate recommendations while preserving the privacy of individual users.

In summary, differential privacy offers a promising approach to addressing privacy concerns in recommender systems. By adding noise to the data used for making recommendations, it is possible to protect the privacy of individual users while still providing accurate and useful recommendations.

Laplacian Noise



Laplacian noise is a type of random noise that can be added to data to provide differential privacy guarantees. Laplacian noise is named after the Laplace distribution, which is a probability distribution that is used to generate the noise.

In differential privacy, Laplacian noise is often used to add randomness to the output of a computation. The amount of noise added is determined by a privacy parameter called epsilon (ε), which controls the strength of the privacy guarantee.

The Laplacian distribution is characterized by a mean value and a scale parameter. The mean value is typically set to zero, while the scale parameter is determined by the privacy parameter epsilon and the sensitivity of the data being protected.

The sensitivity of a function is a measure of how much the output of the function can change in response to small changes in the input data.

For example, the sensitivity of a counting query is 1, since changing a single record in the data set can change the output of the query by at most 1.

The amount of Laplacian noise added to the output of a computation is proportional to the sensitivity of the function and inversely proportional to the privacy parameter epsilon. This means that as epsilon gets smaller, more noise is added to the output to ensure strong privacy guarantees.

Laplacian noise is commonly used in differentially private algorithms for data analysis, including histogram queries, mean and median calculations, and machine learning algorithms such as logistic regression and support vector machines. The addition of Laplacian noise helps to protect sensitive information in the data while still providing accurate results.

The Matrix Factorization method

Matrix factorization is a machine learning technique used for collaborative filtering in recommendation systems. The goal of matrix factorization is to decompose a large matrix into two smaller matrices that capture the latent features or characteristics of the original matrix.

In the context of recommendation systems, the original matrix typically represents the user-item ratings matrix, where each row corresponds to a user and each column corresponds to an item. The entries in the matrix represent the ratings that users have given to different items. Matrix factorization aims to find two smaller matrices, one representing the users and the other representing the items, that when multiplied together, closely approximate the original ratings matrix.

The factorization of the original matrix into two smaller matrices can be represented as A ≈ U × V^T, where A is the original ratings matrix, U is the user matrix, and V is the item matrix. The goal is to find U and V such that the difference between A and U × V^T is minimized.

The matrix factorization problem is typically solved using gradient descent or stochastic gradient descent. During training, the algorithm updates the values of U and V iteratively to minimize the difference between A and U × V^T.

Once the user and item matrices have been learned, they can be used to generate recommendations for new users or items by computing the dot product of the corresponding rows or columns in U and V, respectively.

Matrix factorization has been shown to be an effective technique for collaborative filtering in recommendation systems, and is widely used in applications such as movie recommendations, product recommendations, and music recommendations.

Method To Train A Model Using Differential Privacy

Training a model using differential privacy involves modifying the standard machine learning algorithm to incorporate privacy-preserving mechanisms. Here is a general overview of the steps involved in training a model using differential privacy:

1. Define the privacy budget (Epsilon): The first step is to define the level of privacy required for the model. This is typically done by setting a privacy budget, which is a measure of the maximum amount of privacy that can be lost during the training process.

2. Modify the model: The next step is to modify the machine learning model to incorporate differential privacy. This can involve adding noise to the data, randomizing the model parameters, or using other techniques to preserve privacy.

3. Choose an appropriate algorithm: There are several different algorithms available for training models using differential privacy, each with their own strengths and weaknesses. It's important to choose an algorithm that is appropriate for the specific application and data being used.We used Stochastic Gradient Descent(SGD) in our experiments.

4. Train the model: Once the modifications have been made and the algorithm chosen, the model can be trained using the modified algorithm. During training, the privacy-preserving mechanisms will be applied to ensure that the privacy budget is not exceeded.

5. Evaluate the model: Once the model has been trained, it can be evaluated using standard machine learning metrics to determine its accuracy and performance.

6. Adjust the privacy budget: If the model does not meet the required level of privacy, the privacy budget can be adjusted and the model retrained.

Overall, training a model using differential privacy requires a careful balance between privacy and accuracy. The privacy-preserving mechanisms must be strong enough to protect individual privacy, but not so strong as to significantly affect the accuracy of the model.

We trained the Standard Matrix Factorization model with two variations:

(i) Input Training Data Perturbed With Laplacian Noise
(ii) Perturb the Learning Algorithm i.e. SGD by introducing Laplacian Noise

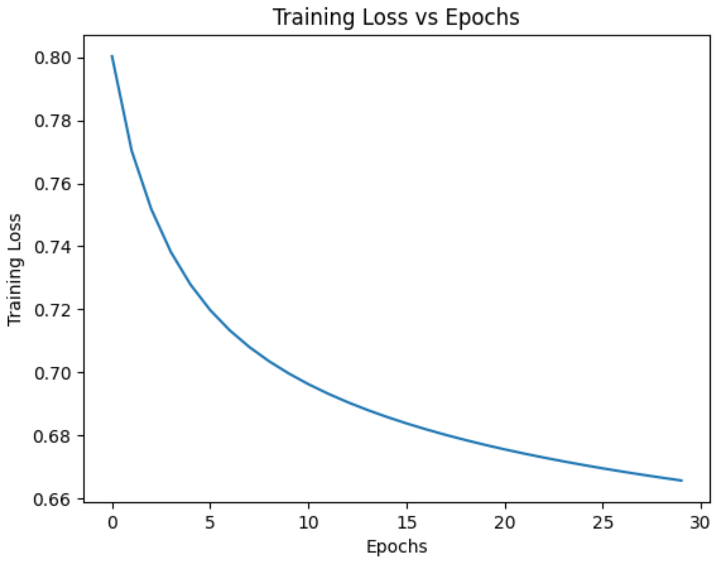
A higher epsilon(ε) value would mean that the model has more leeway for privacy i.e. it has a higher budget for the amount of information leakage it can allow and hence proportionally a lower amount of Laplacian noise shall be added during the training phase. A lower degree of Laplacian noise should ideally make way for higher model test accuracy. Conversely, a lower privacy budget would translate to higher compensatory noise being added to the model and thereby it is expected that with lower values of the epsilon(ε), the model accuracy should deteriorate.

Our experiments are designed to inject noise in training data and also in the SGD algorithm to verify the above mentioned property of differential privacy on a Recommender System Model.

Results

Case 1:

Training Loss Curve Of A Standard Matrix Factorization ML Model Without Any Differential Privacy Related Noise Applied (i.e. No Input Perturbations)



(Training Model with No Noise Added)

Test Dataset Model Accuracy Results

Precision@10: 0.05868852459016405

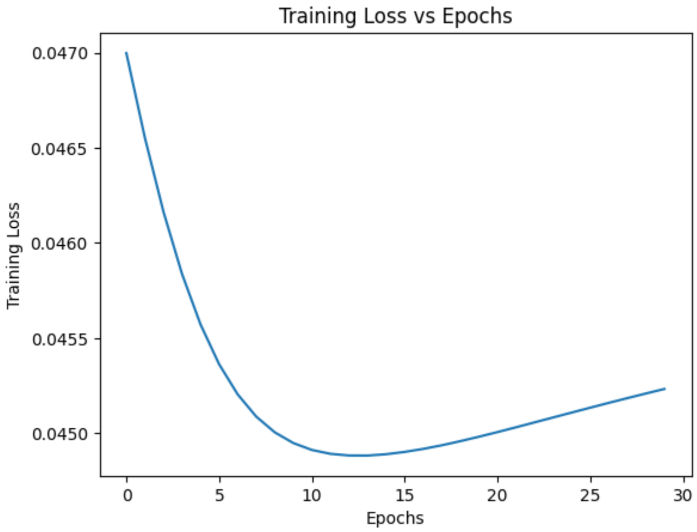
Recall@10: 0.03136039986110803

Analysis of Results

We observe from the above curve that the average loss training loss for the Matrix Factorization model (which is under no influence of any type of noise steadily) reduces with gradient descent for increasing training epochs. In other words the reconstruction of the user-rating matrix utilizing the user and item vectors with 100 latent factors is steadily heading towards a small loss value or that the actual user-ratings matrix i.e. the ground truth and the one learned by the Matrix-Factorization technique using gradient stochastic descent is approaching convergence. This observation is in-line with our expectations. Also, we observe that the precision and recall values for the test dataset for the top 10 items is approximately 0.06 and and 0.03 which although is modest but shall become the benchmark for comparison with the cases where we add Laplacian noise. We should however note that the dataset used in this study is the MovieLens small dataset which is a very condensed version of the actual dataset and has a high sparsity which means that the accuracy of the model is not completely representative of that in a real world scenario where user ratings matrices are quite dense and contain millions of user interactions.

Case 2:

Training Loss Curve Of A Standard Matrix Factorization ML Model With Differential Privacy Related Laplacian Noise Applied to Training Data (i.e. Input with Perturbations)



(Training Model with Noise Added to Input Training Dataset)

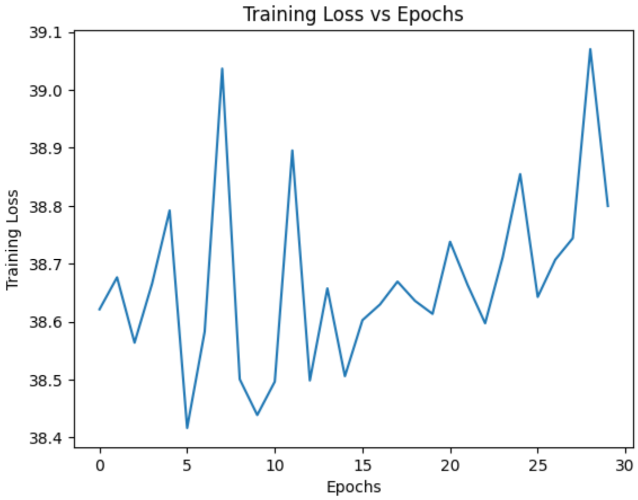
Test Dataset Model Accuracy Results

Precision@10: 0.017377049180327852
Recall@10: 0.006416994061097673

Analysis of Results

When Laplacian noise is added to the input training ratings matrix (i.e. the training dataset only) in a manner in which the differential privacy is maintained to a reasonable extent i.e. with epsilon (or privacy budget) equal to 5, the above loss curve is observed when training the Matrix Factorization model using stochastic gradient descent. It can be observed that although the training loss reduced for the first 10 epochs, it steadily increased and displayed an upward trend which did not change for the rest of the epochs in the experiment. Moreover, it is also worth noting here that even the smallest loss value for this ‘input-perturbed’ case is more than that of the case where no noise was added to the training dataset containing user ratings. The precision and recall on the test dataset(which has no perturbations) is approximately 0.02 and 0.006. Both of these values are lower than the no-perturbations case which means that adding Laplacian noise to enforce a certain degree of privacy has compromised the accuracy of the Matrix Factorization model.

Case 3:
Training Loss Curve Of A Standard Matrix Factorization ML Model With Differential Privacy Related Laplacian Noise Applied To The Stochastic Gradient Descent Algorithm



(Training Model with Noise Added to SGD algorithm)

Test Dataset Model Accuracy Results

Precision@10: 0.019672131147540954
Recall@10: 0.008757988504225734

Analysis of Results

This is an example of DP-SGD (Differential Privacy for Stochastic Gradient Descent) i.e. when the Stochastic Gradient algorithm perturbations are scaled based on the sensitivity of the error. This is another form of Differential privacy where the model learning is perturbed in a bounded manner to preserve privacy. We observe from the above that the training loss curve shows no signs of decay with the increasing number of epochs and rather shows moderately large fluctuations. Moreover, we observe that there is an upswing in the training loss from approximately epoch number 28. The precision and recall of this model is approximately 0.02 and 0.009 respectively which when compared to case 1(model trained with no input noise) has significantly lesser accuracy but almost comparable accuracy with case 2 (model trained with perturbed training data).

A Comparative Study Of How Model Accuracy Changed With Change In Epsilon Values(i.e. With increase in Differential Privacy Budget) for Input Perturbations and DP-SGD

Input Dataset Perturbations	**Case A**	
Epsilon value	**Precision@10**	**Recall@10**
0.2	0.014918	0.005736
1	0.016393	0.005225
5	0.021147	0.008939
Learning Algorithm(SGD) Perturbations	**Case B**	
Epsilon value	**Precision@10**	**Recall@10**
0.2	0.003442	0.000859
1	0.003442	0.000859
5	0.025081	0.010593

For each epsilon value, the Matrix Factorization method was trained for 30 epochs

Analysis of Results

We can observe from the above table that for case A i.e. when Laplacian Noise is added to the input training data, the value of precision increases for higher values of epsilon and recall also shows a similar upward trend. For case B. i.e. when perturbations were added to the learning algorithm (SGD), the precision remained the same for epsilon values 0.2 and 1 and increased for the value 5. Recall on the other had remained the same for 0.2 and 1 and increased for 5.

One more important observation from the above table is that the Matrix Factorization model accuracy (both precision and recall) significantly deteriorates(by almost 10 fold) when perturbations are added to the SGD learning algorithm than input training data perturbations.

Conclusion and Future Work

It can be concluded from our observations that adding Laplacian Noise to the Matrix Factorization model does in fact impact the accuracy of the model. The degree of added noise is controlled by the parameter epsilon(ϵ) which is the privacy budget. A lower value of epsilon(ϵ) indicates a low privacy leak tolerance and hence addition of higher noise addition as the inverse of the epsilon value is the input to the Laplace probability density function. We conclude that the noise added to the SGD algorithm leads to a lower model accuracy as compared to the input training data noise injection technique and the Training Loss curve fluctuates and does not converge for higher epoch values.

We experimented with a very small academic dataset of MovieLens. We could have used a much larger MovieLens dataset in our experiments to validate our findings. Instead of Laplacian Noise, we could have added the Gaussian Noise and validated the trend observed in our experiments. Moreover, it would have been also beneficial if we could have varied the hyper-parameters like learning rate, number of latent factors etc. in our experiments.