Recommendation System

focuses on user privacy and scalability

Group 35

Runze Zhao 120090715
Yiwen Lu 120090811
Zening Xiong 120090591
Shunuo Shi 120090216

Submitted to DDA4210

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

DATE SUBMITTED
[Revised May 5, 2023]

1 Introduction

This report focuses on a recommendation system that prioritizes user privacy and scalability. We encountered three main challenges: maintaining accuracy with noisy data, achieving fast computation for different dataset sizes, and handling new input effectively. To overcome these challenges, we proposed two novel approaches: using hash and Laplace noise techniques for privacy protection and implementing an ensemble model for quick computations and smooth integration of new datasets. These innovative strategies provide promising solutions for practical concerns in recommendation systems.

2 Data Pre-processing

The datasets utilized in our study are derived from real-world data obtained from SpareChat.[1]

2.1 Test data based on intuition

We performed data analysis to ensure user privacy while achieving better performance:

1. General data analysis to identify correlations and cause-and-effect relationships.

   (a) Correlation of two predicted values—Kendall's tau-b method

   $$\tau_b = \frac{n_c - n_d}{\sqrt{(n_0 - n_1) \times (n_0 - n_2)}}$$

   - n_c, n_d: the number of concordant and discordant pairs, respectively (where the rankings of both variables agree).
   - n_0: the total number of pairs.
   - n_1: the number of pairs where variable 1 ranks higher than variable 2.
   - n_2: the number of pairs where variable 2 has a higher rank than variable 1.
     The correlation parameter is 0.122. They only have a weak positive correlation.

   (b) Cause and effect evaluation—Granger causality test

   $$F = \frac{(RSS_r - RSS_u)/q}{RSS_u/(T - 2k - q)}$$

   - RSS_r: the residual sum of squares from the restricted model (which only includes lagged values of the second time series).
   - RSS_u: the residual sum of squares from the unrestricted model (which includes lagged values of both time series).
   - q: the number of lagged values included in the models.
   - k: the number of additional independent variables included in the models (if any).
   - T: the number of observations in the time series.

   The results show that these two values do not appear cause-effect relation.

2. Handling outliers to improve accuracy.

   (a) Normalization test—Shapiro-Wilk test

   $$W = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

   - W: the test statistic

- a_i: the i-th element of the vector of constants that depend on the sample size and are used to calculate the expected values of the order statistics under normality
- $x_{(i)}$: the i-th order statistic (i.e., the i-th smallest value in the sample)
- x_i: the i-th observation in the sample
- $\bar{x}$: the sample mean

(b) 3σ test for normal distribution to detect the outliers

2.2 Hashing

To protect user data, we utilized hashing and added Laplace noise to the data:

1. Hashing ad categorical features to anonymize the data—SHA-256 algorithm

$$h = \text{SHA-256}(m)$$

- h: the resulting hash value
- m: the message to be hashed

2. SHA-256 algorithm

   (a) Padding:

      i. Append a single '1' bit to the end of the message.

      ii. Append '0' bits until the length of the message in bits is congruent to 448 modulo 512.

      iii. Append a 64-bit representation of the length of the original message in bits.

   (b) Processing each block:

      i. Initialize message schedule array W with the 64 constant words K_i.

      ii. Initialize eight working variables a, b, c, d, e, f, g, h with the hash values from the previous block (or the initial hash values for the first block).

      iii. Perform 64 rounds of operations on $a, \ldots, h$ using current message block m_i and W.

   (c) Compute final hash value: Concatenate the values of $a, \ldots, h$ in that order.

2.3 Laplace Mechanism

To enhance data privacy, we added Laplace noise to the numerical data learned from the lecture.

$$X_{ij} \sim \text{Laplace}(f(D)_{ij}, \frac{\Delta f}{\varepsilon})$$

Here, X_{ij} := the i-th observations the j-th features; D := Numercial Datasets. The noise is generated based on the scale of the data, ensuring privacy while preserving statistical properties. Additionally, we set lower and upper bounds for the modified values based on the minimum and maximum values of each data type[2], ensuring the modified data remains within bounds. These measures aim to enhance privacy by introducing controlled noise while maintaining data utility and statistical properties.

3 Algorithm Design

3.1 Ensemble Learning: Random Forests

We chose Random Forest as our ensemble model for its simplicity, effectiveness, and versatility in classification. It offers high prediction accuracy and reduced variance.

Algorithm 1 Random Forest Algorithm

```
Input: Training data $D = (x_1, y_1), \ldots, (x_n, y_n)$, trees' number $T$, features' number considered at each split $m$
Output: Random Forest model $F$
for $t \leftarrow 1$ to $T$ do
    Sample a bootstrap dataset $D_t$ from $D$  Grow a decision tree $T_t$ from $D_t$ by recursively splitting the nodes using the
      following steps:
        1. Randomly select $m$ features from the total $p$ features.
        2. Calculate the best feature/split-point among the selected features using Gini Index/Entropy.
        3. Split the node into two child nodes using the best-split point.
        4. Repeat steps 1-3 on each child node until the maximum depth is reached.
end
return Random Forest model $F = T_1, T_2, \ldots, T_T$
```

3.2 Neural Network

In conjunction with developing the ensemble model, we further delved into exploring and applying Neural Networks (NN). These sophisticated models present several notable advantages:

1. Neural Networks inherently operate as a "black box", enabling the seamless handling of encrypted data. This characteristic facilitates the processing and analysis of complex datasets while preserving the privacy and security of the underlying information.

2. To mitigate the risk of overfitting, we have strategically implemented dropout layers within the NN architecture, enhancing the model's generalizability and performance on unseen data.

The Dropout Neural Network Algorithm is a robust framework for building and training neural network models. It takes the training data (X_train and y_train) and the testing data (X_test) as inputs and generates a prediction vector, y_pred, containing the model's anticipated results for the test dataset. In conclusion, this algorithm offers a structured and efficient approach to developing and training neural network models applicable to diverse domains and applications.

Algorithm 2 Dropout Neural Network Algorithm

```
Input  : $X\_train$, $y\_train$, $X\_test$
Output: $y\_pred$
Function LayoutNN($X\_train, y\_train, X\_test$)
        1. Initialize, the neural network model
        2. Add input layer, hidden layers, dropout layers, and output layer using designed functions
        3. Compile the model with loss, optimizer, and metrics
        4. Train the model on $X\_train, y\_train$ with 10 epochs
        5. return $y\_pred \leftarrow$ Predict the output for $X\_test$
end
```

4 Performance

4.1 Results

We evaluated the performance of our algorithms and obtained the following results:

		Acc(Is Clicked)	Acc(Is Installed)
RF	train(hash)	0.8527	0.8536
	test(hash)	0.8512	**0.8590**
	train(hash+noise)	0.8250	0.8375
	test(hash+noise)	0.8157	0.8262
NN	train(hash)	0.8388	0.8427
	test(hash)	**0.8520**	0.8475
	train(hash+noise)	0.8848	0.8995
	test(hash+noise)	**0.8573**	**0.8483**

Table 1: Table for our Classification result

4.2 Analysis of Results

- Random Forest (RF) showed lower computational cost and faster results overall.
- RF performed better after the first encryption (hash).
- Neural Network (NN) performed better after the second encryption (hash + noise).
- Possible explanations:
  - Information Compression: Laplace noise-based encryption may remove redundant features, improving neural network efficiency and prediction accuracy.
  - Reduced Overfitting: Encrypted data reduces overfitting in shallow neural networks, resulting in improved performance on the test set.

5 Summary

Throughout this report, we have made significant strides in developing and applying recommendation systems. The key accomplishments that have been achieved are enumerated below:

- We have conducted an in-depth exploration and analysis of various algorithms applicable to recommendation systems. This investigation has allowed us to gain a comprehensive understanding of the strengths and weaknesses of each algorithm, as well as their suitability for specific use cases.
- We have successfully designed and implemented an ensemble model that leverages the capabilities of the Random Forest algorithm. This approach has facilitated rapid computational speeds and seamless integration of new data, thus contributing to the recommendation system's overall performance.
- In addition, we have employed Neural Networks as a means to process encrypted data and improve prediction accuracy. By harnessing the power of these advanced computational models, we have achieved a higher level of precision and effectiveness in our recommendation systems.
- Lastly, we have taken significant steps to address concerns related to user privacy. This has been achieved through robust techniques such as hashing and the Laplace noise mechanism. By incorporating these methods, we have ensured that user data remains secure and protected while enabling our recommendation systems' efficient functioning.

In summary, this report details the substantial progress we have made in the realm of recommendation systems, with particular emphasis on developing effective models, incorporating advanced algorithms, and safeguarding user privacy. Our accomplishments serve as a testament to our commitment to delivering cutting-edge solutions that address the diverse needs of our users.

6 References

1. ShareChat Dataset. Retrieved from: <dataset_link>

2. Sarathy, V., & Muralidharan, P. (2011). Evaluating differential privacy: data, utility, and lower and upper bounds. In Proceedings of the 8th annual ACM workshop on privacy in the electronic society (pp. 91-102).