Recommendation System

focuses on user privacy and scalability

Group 35

April 23, 2023

Significance

Our topic on the recommended system is significant and relevant, mainly focusing on two practical concerns.

1. **User Privacy**
   With the increasing worry of misusing or exploiting users' information, companies focus more on ***user privacy***.
2. **Scalability**
   Data are often very large in scale. To accommodate diverse work environments in recommendation systems, it is essential to adopt versatile models. The inherent design and flexibility of our models allow for effortless scalability.

Challenge

From demand analysis, we are faced with three major challenges:

1. How to remain highly accurate with data adding noise?
2. How to achieve rapid computational speed for both a small and a large scale of data?
3. How to deal with new input more effectively and efficiently?

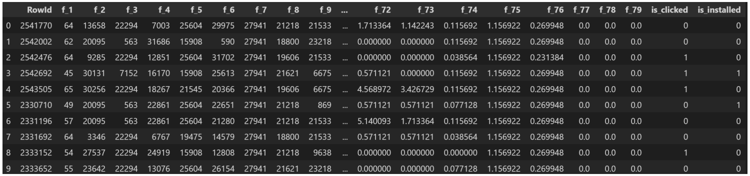
Novelty

To address the concerns:

1. **Hash and add Laplace Noise** $\Rightarrow$ protect users' privacy.
2. **Ensemble model** $\Rightarrow$ rapid computational speeds & incorporate new data sets based on the existing recommendation model.

About out data sets

- The data[1] consists of impressions/clicks/installs $\Rightarrow$ predict users' clicks and installation.
- We did cross-validation on the data and formed training and test data.
- The details are shown in the appendix.



	RowId	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	...	f_72	f_73	f_74	f_75	f_76	f_77	f_78	f_79	is_clicked	is_installed
0	2541770	64	13658	22294	7003	25604	29975	27941	21218	21533	...	1.713364	1.142243	0.115692	1.156922	0.269948	0.0	0.0	0.0	0	0
1	2542002	62	20095	563	31686	15908	590	27941	18800	23218	...	0.000000	0.000000	0.115692	1.156922	0.269948	0.0	0.0	0.0	0	0
2	2542476	64	9285	22294	12851	25604	31702	27941	19606	21533	...	0.000000	0.000000	0.038564	1.156922	0.231384	0.0	0.0	0.0	1	0
3	2542692	45	30131	7152	16170	15908	25613	27941	21621	6675	...	0.571121	0.000000	0.115692	1.156922	0.269948	0.0	0.0	0.0	1	1
4	2543505	65	30256	22294	18267	21545	20366	27941	19606	6675	...	4.568972	3.426729	0.115692	1.156922	0.269948	0.0	0.0	0.0	1	0
5	2330710	49	20095	563	22861	25604	22651	27941	21218	869	...	0.571121	0.571121	0.077128	1.156922	0.269948	0.0	0.0	0.0	0	1
6	2331196	57	20095	563	22861	25604	21280	27941	21218	21533	...	5.140093	1.713364	0.115692	1.156922	0.269948	0.0	0.0	0.0	0	0
7	2331692	64	3346	22294	6767	19475	14579	27941	18800	21533	...	0.571121	0.571121	0.038564	1.156922	0.269948	0.0	0.0	0.0	0	0
8	2333152	54	27537	22294	24919	15908	12808	27941	21218	9638	...	0.000000	0.000000	0.000000	1.156922	0.269948	0.0	0.0	0.0	1	0
9	2333652	55	23642	22294	13076	25604	26154	27941	21621	23218	...	0.000000	0.000000	0.077128	1.156922	0.269948	0.0	0.0	0.0	0	0

Figure 1: Data description

Data pre-processing–focus on privacy

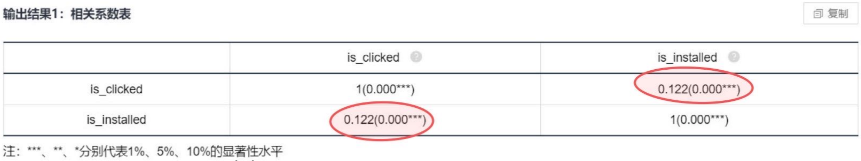
Data analyzing:

1. To perform better:
    - General data analysis — correlation
    - Outlier — accuracy
2. To protect privacy:
    - Hash
    - Laplacian

General data analysis

Conclusions from previous tests

- Correlation parameter is 0.122
- They do not appear to have an obvious cause-and-effect relation

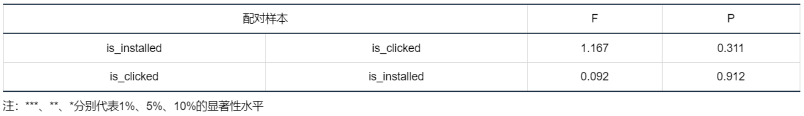

输出结果1：相关系数表

复制

	is_clicked	is_installed
is_clicked	1(0.000***)	0.122(0.000***)
is_installed	0.122(0.000***)	1(0.000***)

注：***、**、*分别代表1%、5%、10%的显著性水平

(a) Correlation test—Kendall's -τ-b correlation analysis



配对样本		F	P
is_installed	is_clicked	1.167	0.311
is_clicked	is_installed	0.092	0.912

注：***、**、*分别代表1%、5%、10%的显著性水平

(b) Cause and effect test—Granger Causality

Figure 2: Test for two prediction values:is_clicked and is_installed

Outlier

Figure out outliers

- Normalization test—whether data follow Normal distribution
- 3-σ outlier test

Two methods to deal with outliers

- replace outliers with the mean of data
- replace outliers with NaN and delete these rows in the data

Possible reasons for the accuracy decline

- outliers may contain valuable information or indicate real-world phenomena. For example, in some scientific studies, outliers may represent rare or extreme events of particular interest.

Hash

The benefits of using Hash

- Uniqueness of the hash value — enlarge the data difference
- Irreversibility of the hash function — privacy
- Stability — a specific input leads to a specific output

Compared to the recommendation system

- Ad categorical features are hashed to 32-bit to anonymize the data
- One hash value vs. One feature vector

Laplace Mechanism

To protect data in advance, we added Laplace Noise.

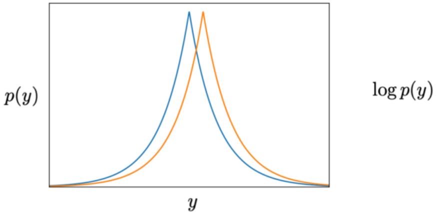


Figure 3: Laplace Distribution

- X_{ij} := the i-th observations the j-th features; D := Datasets.
- $X_{ij} \sim \text{Laplace}(f(D)_{ij}, \frac{\Delta f}{\varepsilon})$
- By reviewing essay:
  "*In many situations, it may not even be possible to implement differential privacy ...... **lower and upper bounds**.*" [2]

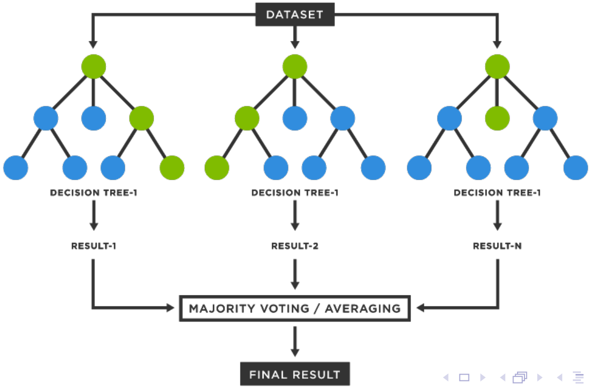
Designing logic

To achieve fast computations and easily integrate new data, we have chosen the **ensemble model** over the transductive model because,

- it offers better prediction accuracy
- it reduced variance and improved generalization.

Random Forest

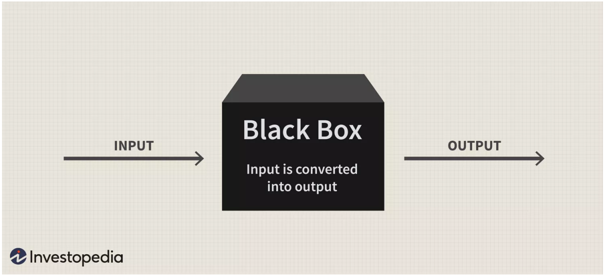
As one of the most popular ensemble learning methods, Random Forest can use for classification, regression, and feature selection tasks. Even though the algorithm is simple, the results can usually outshine more complicated algorithms.



Why are we trying NN as our model

In addition to ensemble models, we have decided to explore the use of **Neural Networks** since

- Neural Networks work as a "black box" – we want to see how can it behave for encrypted data.



- Borrowing idea from federated learning.

Avoid overfitting

- To **minimize overfitting**, we incorporate the dropout layers between layers during training.
- How does this work? We use TensorFlow's Dropout function to nullify input units between layers randomly. To maintain the sum of elements of non-zero inputs, we scale non-zero output by:

$$\textit{input} \rightarrow \begin{cases} 1/(1-\textit{rate}) & \text{if output is not chosen to be 0} \\ 0 & \text{otherwise} \end{cases}$$

Here "rate" is a hyper-parameter scaling between $(0, 1)$.

Implementation of algorithm

Our algorithm and pseudo-code to implement NN here:

Algorithm 1 Layout 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 metric
4 Train the model on $X\_train, y\_train$ with 10 epochs
5 return $y\_pred \leftarrow$ Predict the output for $X\_test$
```

Results

Y_1 and Y_2 are the accuracies of the two labels we want to predict.
x is the number of decision trees.
Different number of decision trees from 70 to 130 does not make significant differences.

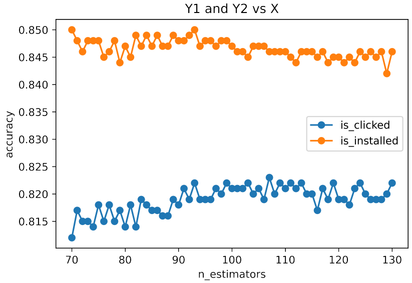


Figure 5: Relationship between accuracy and number of trees

Output results of our algorithms

We trained on 170000 training data and tested on 20000 test data

		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

Analysis of results

- RF has an overall lower computational cost and generates much faster.
- RF works better after the first encryption.
- NN works better after the second encryption (usually not) Possible Explanations:
  - Information Compression: Laplace noise-based encryption may remove redundant features, allowing the neural network to learn more efficiently and improve prediction accuracy.
  - Reduced Overfitting: Shallow neural networks can overfit original data due to limited complexity. Encrypted data may reduce the overfitting likelihood, leading to better test set performance.

What we have done and what we will do in the future

1. What we did
   - We explored several algorithms for recommendation systems:
     - Random forest algorithm can rapidly absorb new data without retraining the whole model. This significantly saves computational resources.
     - Neural Network can generate a surprisingly good result after getting encrypted.
   - Combined with our concern for users' privacy.
     - We innovatively combined hash operations and Laplace noise for users' data encryption, effectively protecting user privacy without significantly sacrificing accuracy.
2. Future work: We will try to achieve higher levels of encryption and provide a reasonable interpretation of the results.

Thanks for your listening!

[1] Sharechat recsys 2023 dashboard.
`https://sharechat.com/recsys2023/dashboard`.
[Accessed: April 22, 2023].

[2] Sarathy, R., and Muralidhar, K.
Evaluating laplace noise addition to satisfy differential privacy for numeric data.
Trans. Data Priv. 4, 1 (2011), 1–17.

About our data sets:

1. Users:
   Demographic features: age, gender, and geographic location (The user's location is hashed to a 32-bit)
   Features on non-ad: These embeddings are trained based on the user's consumption of the various non-ad content on the Sharechat/Moj app.
   Features on ads: These embeddings are trained based on the past apps installed by the user on our platform.
2. Ads:
   CB: These features represent different characteristics of an ad, including the size of the ad, the category of the ad, etc. The features are hashed to 32-bit.
   Ad embedding: These represent the ad's actual video/image content.

3. Ads and Users:
   CF: These features represent the user interaction with ads, advertisers, and categories of advertisers over different lengths of a time window
   is_clicked: whether the user clicks the ad
   is_installed: whether the user installs the things recommended by the ad