You Are What You Search: Attribute Inference Attacks Through Web Search Queries

Tianyu Du[1(✉)], Tao Tao[2], Bijing Liu[3], Xueqi Jin[4], Jinfeng Li[1], and Shouling Ji[1,5]

[1] Zhejiang University, Hangzhou, China
{zjradty,lijinfeng_0713,sji}@zju.edu.cn
[2] State Grid Hangzhou Power Supply Company, Hangzhou, China
taotao980925@aliyun.com
[3] NARI Group Corporation, Beijing, China
beginjing@163.com
[4] State Grid Zhejiang Electric Power Co., LTD., Hangzhou, China
jxqoffice@163.com
[5] Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, Hangzhou, China

Abstract. Most, if not all, existing attribute inference attacks leverage users' social friendship information and/or social behavioral information to infer the attributes of a target user. In this paper, we study this problem in a novel angle. Specifically, we study whether a user's private attributes (e.g., age, gender, and education) can be inferred based on his or her query history, which is the first such attempt to the best of our knowledge. We present a thorough description of our query-based attribute inference attack and experimentally evaluate our method on a real-world dataset provided by Sogou. Experimental results show that our method can achieve 70.21% for the precision, 68.82% for the recall, and 69.50% for the F1-score on average. When predicting users' gender, the proposed method has precision of 84.56%. This suggests that query records indeed disclose a significant amount of information about users.

Keywords: Attribute inference · Privacy · Query classification · Ensemble learning

1 Introduction

Datasets that contain personal information gradually become public to support research and for other purposes, which is prone to posing privacy risks. For example, users' friends and behavior could disclose users' personal information [9,12], which many people may view as sensitive. An attribute inference attack is a process wherein attackers infer users' private attributes such as age, gender, education or even political orientation and religious belief from public information.

Once users' attributes have been inferred, they can be used for security-sensitive activities like spear phishing [1] and backup authentication [11]. Attackers can be anyone who are interested in users' attributes, like advertisers and hackers. In addition to the privacy leakage, attackers who know the attributes about a person could link users across different sites [2] or with offline records [5] to complete individual profiles, which may cause more severe security risks. Previous attribute inference techniques can be roughly divided into two categories. One is based on social friends, since people connected by social ties tend to share similar attributes. The other is based on behavior, because people who have similar behaviors are inclined to share the same interests and attributes.

Intuitively, characteristics and interests of a user can be disclosed subconsciously through query records. For example, men are more interested in military and cars than women. People who have high education are inclined to get information about society and economy. People in the 19–23 age range often search more about college life and social topics. Based on this conjecture, we may leverage query records to conduct an attribute inference attack. A crucial issue for this kind of attack is to effectively classify massive queries that are short, ambiguous and noisy, which is a challenging problem. First, queries can be as simple as a single character, or as complex as a piece of code segment. Second, some queries are easy to understand, while others may contain multiple meanings. Furthermore, some words may just have the meaning defined in the dictionary, whereas others may have some special meanings on the Internet. Finally, the meanings of queries may also change over time.

Our Work. In this paper, our goal is to leverage queries to infer user attributes, which is the first such attempt to the best of our knowledge. We depart from prior work in the following way: rather than analyzing the risks of attribute inference using social friends information or behavior information, we adopt a combined and in-depth feature representation method that effectively interprets the typically short, ambiguous and noisy query records. We comprehensively evaluate our method for inferring age, gender and education using a dataset with 100,000 users provided by Sogou, one of the main search engines in China. From the experimental results, our method can achieve 70.21% for the precision, 68.82% for the recall, and 69.5% for the F1-score on average. When predicting users' gender, the proposed method has precision of 84.56%. These results imply that an attacker can use users' queries to infer their attributes at high accuracy.

Our Contributions. In conclusion, our key contributions are as follows:

1. We propose a query-based attribute inference framework that exploits users' query records to conduct attribute inference. Our results have serious implications for user privacy – private attributes can be inferred from users' query records.
2. Our method has two distinguishing features: (i) it combines three different features; and (ii) it combines three different kinds of classifiers. Moreover, we analyze the impacts of different feature representation methods and feature selection methods on the given dataset, respectively.

3. Leveraging a real world dataset (100,000 users, 12,608,914 queries), we conduct evaluations to examine the performance. Experimental results demonstrate that our method achieves a satisfying inference performance. For instance, it can successfully infer the age, gender, education for 61%, 84%, 63% of the users, respectively. Our best result for averaged precision, recall and F-score is 70.21%, 68.82%, 69.50%, respectively. This suggests that the query records carries much information about users. Therefore, users' private information suffer from serious risks.

2 Related Work

2.1 Attribute Inference

Previous works that attempt to infer the attributes of the users were mostly based on users' social relationships and behavioral records. He et al. [12] used Bayes network to analyze the relations among people in social network. Lindamood et al. [14] revised Naive Bayes (NB) classifier to integrate social links and other attributes of users to conduct attribute inference. Thomas et al. [16] used information about users' friends and wall posts to infer attributes such as religious views, political views and gender based on multi-label classification. Fang et al. [8] leveraged the correlations between user attributes to improve the performance of attribute inference. Weinsberg et al. [18] studied the inference of gender using rating scores and found that Logistic Regression (LG) classifier outperforms Support Vector Machine (SVM) classifier and NB classifier. Chaabane et al. [6] correlated the music with Wikipedia pages and used topic model to find the latent correlations between music. They identified that users that like similar music are inclined to have same attributes. Gong et al. [9] integrated social friends and user behavior to infer the missing attributes of targeted users. These studies are orthogonal to ours, since they leveraged social network, user behaviors and other information rather than queries.

2.2 Query Classification

Query classification is a difficult problem due to the ambiguous, short and noisy features of web search queries. There are massive algorithms for web search query and short text classification. Cao et al. [5] used neighboring queries and their corresponding clicked URLs in search sessions as context information to solve the problem of query classification based on Conditional Random Field (CRF) models. Beitzel et al. [4] examined three approaches to query classifications and found that combining three techniques has the best performance. The three approaches they used are matching against a list of manually labeled queries, supervised learning of classifiers and mining of selectional preference rules from large unlabeled query logs. Beitzel et al. [3] found that training classifiers from manually classified queries outperforms the bridged classifier using document taxonomy. Hu et al. [13] leveraged Wikipedia to discover massive intent concepts for query classification. Specifically, the intent of any input query is identified through mapping the query into the Wikipedia representation space. Since we focus on Chinese query classification, some extensively used knowledge bases such as Probase, WordNet, Freebase, and YAGO are inapplicable to our problem.

2.3 Ensemble Learning

Ensemble learning is a popular and powerful technique using multiple learning algorithms to get better predictive performance [21]. Intuitively, some classifiers may perform better than others on certain classes. By combining the classification results, different classifiers supplement each other, making the results more robust than single classifier. Many researchers focus on what kinds of base model should be used and how to combine these models [7,15,17,20]. Dietterich [7] reviewed some ensemble methods and explained why ensembles can often perform better than any single classifier. Miskin et al. [15] applied ensemble methods for blind source separation and deconvolution of images. Xia et al. [20] combined features and classification algorithms to produce a more accurate sentiment classification. Verbaeten et al. [17] used ensemble methods to find out noisy training samples in classification tasks. More precisely, they used dozens of filter techniques based on ensemble methods to identify mislabeled training samples and remove them.

3 Methodology

In this section, we first discuss the overall approach to conduct query data-based attribute inference, and then detail the three feature representation methods and one ensemble method.

3.1 Overall Approach

First, we apply text segmentation to divide queries into meaningful units. Second, we use three feature representation methods to express the queries from different aspects. Specifically, We use the Term Frequency - Inverse Document Frequency (TF-IDF) method to obtain word weight vectors from pre-processed data and use term frequency to select the most important feature. Furthermore, we use the Latent Dirichlet Allocation (LDA) model to get the latent user-topic distribution, and use the word2vec method to get word embedding as other features. Finally, we train several classifiers using benchmark machine learning techniques (e.g., Support Vector Machines (SVMs)) to classify these features. To overcome the drawbacks of using a single classifier, we adopt an ensemble method to take advantage of multiple classifiers and achieve better performance.

3.2 Query Representation

We use three different methods to represent queries and get three different kinds of features. The result of TF-IDF is the weight vector of each document. The result of LDA is the topic distribution of each document. The result of word2vec is word embedding, and we take the average of word embeddings in one document as the document embedding. These results are used as the input features of classifiers.

TF-IDF. TF-IDF is evolved from IDF [6]. The intuition is that a frequently used term in different documents is less important and should be assigned with less weight. The basic idea of TF-IDF is that the importance of a term for a given document can be evaluated by term frequency and inverse document frequency. Therefore, the formula of TF-IDF used for term weighting is:

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i}) \tag{1}$$

where $w_{i,j}$ is the weight for term i in document j, $tf_{i,j}$ is the term frequency of term i in document j, df_i is the document frequency of term i and N is the number of documents. However, the tremendous amount of the vocabulary in the dataset makes it computationally expensive to weight all terms. Therefore, we use a term frequency threshold to filter out the trivial terms and use the remaining terms as the input features of classifiers.

Latent Dirichlet Allocation (LDA). For many problems that need a semantic understanding of short text, inferring latent topics is an important task. Conventional topic modeling techniques such as LDA have been studied extensively for various tasks in information retrieval and text mining to extract latent topics from document corpus [10]. The fundamental idea of LDA is that each document is a multinomial distribution over topics and each topic is a multinomial distribution over words. Before we describe the employed LDA model, we briefly introduce its conventional terminology.

- **Word:** an item from the vocabulary of size W.
- **Document:** a sequence of N words represented by $d = \{w_1, w_2, \cdots, w_N\}$ where w_n is the n-th word in the sequence.
- **Corpus:** a collection of M documents represented by $D = \{d_1, d_2, \cdots, d_M\}$ where d_m is the n-th document in the corpus.
- **Topic:** denoted by $Z = \{z_1, z_2, \cdots, z_K\}$ with the assumption that there are K topics in the corpus.

Assume the topic distribution of a document is θ, and word distribution of a topic is ϕ. Then the generative process of each document in a corpus is as follows:

1. Choose the number of word N, $N \sim Poisson(\xi)$
2. Choose θ, $\theta \sim Dir(\alpha)$
3. For $w_n \in d$:
   (a) Choose a topic z_n, $z_n \sim Multinomial(\theta)$

(b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability ϕ^{z_n}.

There are several popular solutions for the LDA model, such as Gibbs Sampling and the Expectation Maximization (EM) algorithm. In this paper, the LDA model is trained by the scikit-learn Python package.

Word2vec. Given two words semantically related while rarely co-occur in short texts, the LDA model cannot learn the semantic relatedness of them. Furthermore, people will also use their background knowledge to understand short texts rather than just use the content words. The recent advances in word embedding provide powerful methods for learning semantic relations, which can be employed to improve the topic modeling for short texts.

Word2vec is a kind of word embedding techniques released by Google in 2013 [4]. It takes a text document as input and outputs word vectors, which can be used as features in many natural language processing (NLP) problems. It has two main model architectures: the Continuous Bag-of-Words (CBOW) model and the skip-gram model. CBOW uses the context to predict the current word, and skip-gram uses the current word to predict the context. Since word2vec has been shown to be well suited for conducting Chinese document classification [8], we apply it for query classification in this paper. In particular, we train the word2vec model using the gensim Python package.

3.3 Feature Selection

Term Frequency. Term frequency is the number of words in the collection of documents. We compute the term frequency for each term in corpus and remove terms whose frequency is less than predetermined threshold. The basic idea is that rare terms are non-informative and uninfluential. Removing rare terms can reduce the dimension of the feature space. Term frequency is the simplest technique for feature selection. However, it is usually considered as an "ad hoc" for selecting features.

χ^2 statistic (CHI2). The χ^2 statistic measures the independence between t and c. It is defined to be:

$$\chi^2(t,c) = \frac{N \times (AD - BC)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \tag{2}$$

where A is the number of times t and c co-occur, B is the number of times t occurs without c, C is the number of times c occurs without t, D is the number of times neither c nor t occurs. The χ^2 statistic will be zero if t and c are independent. However, it only considers whether term t occurs in a document regardless of its times. It makes this method exaggerate the importance of low-frequency terms, so it is usually combined with other factors such as term frequency to overcome its drawbacks.

3.4 Stacked Generalization

In this paper, we use stacked generalization or stacking proposed by Wolpert [19] to combine multiple classifiers, which uses the weights learned from a validation dataset to combine the results of different base classifiers where each one may be somewhat biased. The basic idea is to learn a level-1 classifier based on the results of level-0 classifiers.

Level-0 Generalizers. For a data set $D = (\boldsymbol{x_i}, y_i), i = 1, 2, \cdots, n$ where $\boldsymbol{x_i}$ is a feature vector of the i-th example and y_i is corresponding label, we splits D into K sub-dataset $D_1, D_2, \cdots, D_K$ and perform K-fold cross-validation. At each k-th fold, $D_{-k} = D - D_k$ is used as training part and D_k is used as test part. Then, N algorithms $L_1, L_2, \cdots, L_N$ are applied to the D_{-k} to obtain N level-0 classifiers $C_1, C_2, \cdots, C_N$. The results of the N level-0 classifiers on each example in D_k with its corresponding label form a level-1 sub-dataset MD_k. After cross-validation process, the union $MD = \cup MD_k, k = 1, 2, \cdots, K$ makes up the full level-1 data set. It is worth noting that all the N learning algorithms should be trained on the whole data set D to get the final level-0 classifiers $C_1, C_2, \cdots, C_N$.

Level-1 Generalizers. After we get the level-0 classifiers, we use another learning algorithm L_M to get the level-1 classifier C_M. To classify a new example, the results of all the level-0 classifiers form a level-1 vector of N dimension. Then, the vector is assigned with a label by the level-1 classifier as the final classification result.

4 Experiments

In this section, we first introduce the dataset, the evaluation settings and metrics. Then we perform several experiments to demonstrate the effectiveness of our method. Furthermore, we investigate the impacts of some parameters, and then compare the performance of different feature representation methods. Afterwards, we investigate the performance of different classifiers and verify the effectiveness of ensemble learning.

4.1 Dataset

We use the dataset provided by Sogou[1], which contains 100,000 users' records. Each record is corresponding to one user's data and contains the age, gender and education labels as well as the queries made by that user. Each query is a combination of numbers, punctuation, URLs, acronyms, codes and words.

Table 1 describes the meaning of numbers in each attribute. Figure 1(a) shows the frequency of query length. The query's length vary from one character to more than thirty characters, but most queries' length are limited to ten characters.

Intuitively, one user's age and education can be correlated. For example, a 16 years old user can hardly get education higher than high school. Furthermore, the elder users tend to have higher education. To verify our conjecture, we plot

[1] http://www.datafountain.cn/data/science/player/competition/detail/description/239.

Table 1. Dataset description

Label	1	2	3	4	5	6
Age	0–18	19–23	24–30	31–40	41–50	51–999
Gender	Male	Female	-	-	-	-
Education	Ph.D	M.S.	B.S.	High school	Middel school	Primary school

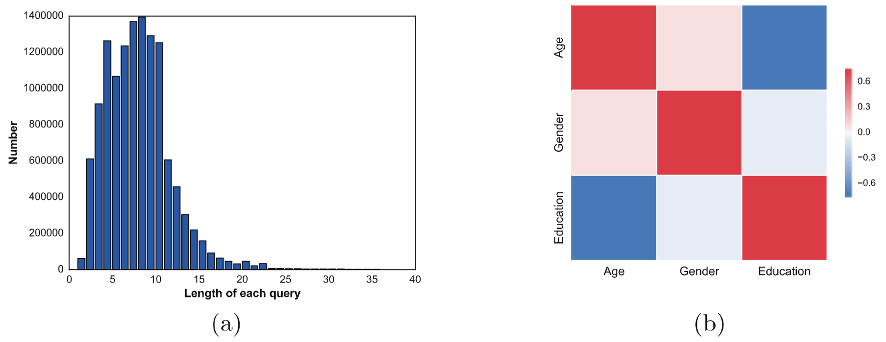


Fig. 1. (a) Frequency of query length. (b) Correlation of three attributes.

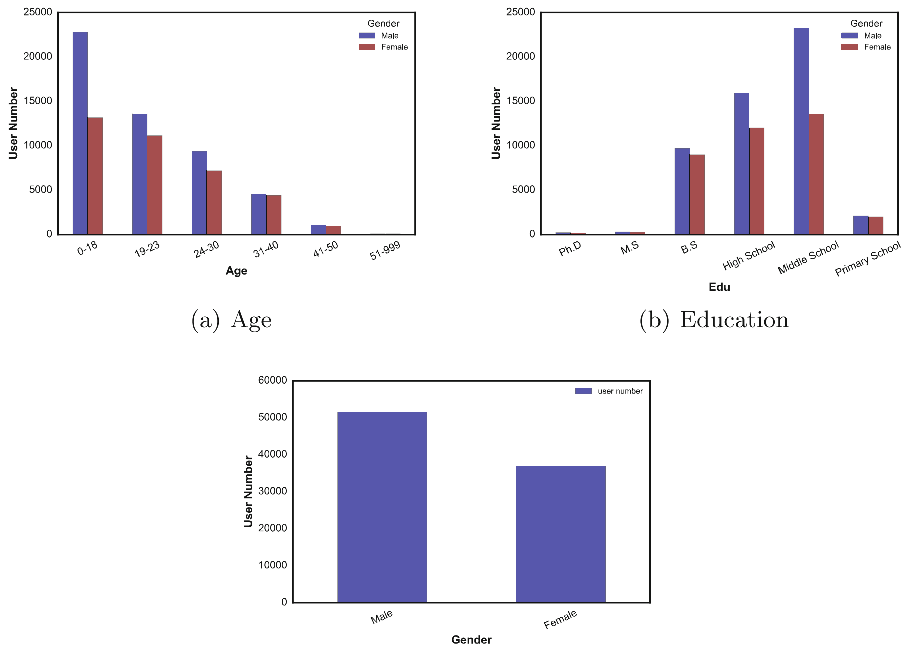
the correlation between different attributes in Fig. 1(b). In Fig. 1(b), the depth of color represent the degree of correlation, and the red means there is positive correlation between two attributes, while the blue means there is negative correlation between them. Apparently, Fig. 1(b) verifies our conjecture.

Figure 2(a) shows the histogram of the number of users in each class in terms of age and gender, and Fig. 2(b) shows the histogram of the number of users in each class in terms of education and gender. Apparently, the numbers of users in different classes are significantly imbalanced. In Fig. 2(a), the largest group "0–18" contributes about 40.64% of the age label in the whole dataset, while the smallest one "51–99" contributes about 0.20%. In Fig. 2(b), the largest group "middle school" contributes about 41.64% of the education label in the whole dataset, while the smallest one "Ph.D" contributes about 0.40%.

4.2 Evaluation Settings and Metrics

Preprocessing. Notice that there are some "0" labels in the dataset, which means the user's attribute is unknown. For preciseness, we remove the record with at least one "0" label in preprocessing. Then, we adopt jieba[2] Python package to segment the queries into words and give them part-of-speech (POS) tags. The preprocessed data is obtained after removing stop words and punctuation characters. Therefore, short queries which contain stop words are not considered

[2] https://pypi.python.org/pypi/jieba/.



(a) Age (b) Education

(c) Gender

Fig. 2. The number of users in each age/education/gender category.

in this evaluation. We omit the details about how to select the stop words as it is not the keystone.

Evaluation Settings. In our following experiments, we randomly and uniformly sample 10% of the users in the whole dataset. We then remove their attribute labels and take them as the testing dataset to evaluate our method. We take the remaining 90% of the users as the training dataset.

Evaluation Metrics. The performance of a classifier is usually measured by precision, recall and F1-score, which are standard measures in machine learning evaluation. The definitions of precision, recall and F1-score are described as follows:

$$Precision = \frac{A}{B}$$
$$Recall = \frac{A}{C}$$
$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision \cdot Recall}$$

where the A denotes the sum of users correctly classified as c_i, B denotes the sum of users classified as c_i, C denotes the sum of users whose true class is c_i.

4.3 Results and Analysis

Effect of Parameter Tuning. To select the most important features, there is an crucial parameter that affect the performance of classifiers - term frequency threshold. We filter out words that don't reach the term frequency threshold. The results of different word frequency threshold are showed in Fig. 3(a). All experiments use the same classifier: SVM with linear kernel, C = 0.1 and other default setting.

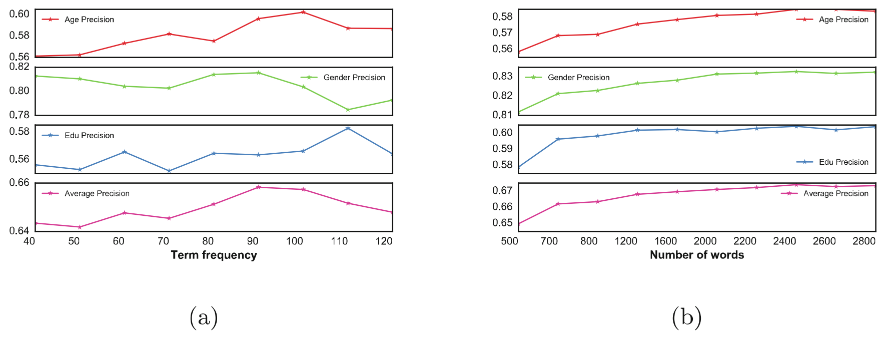
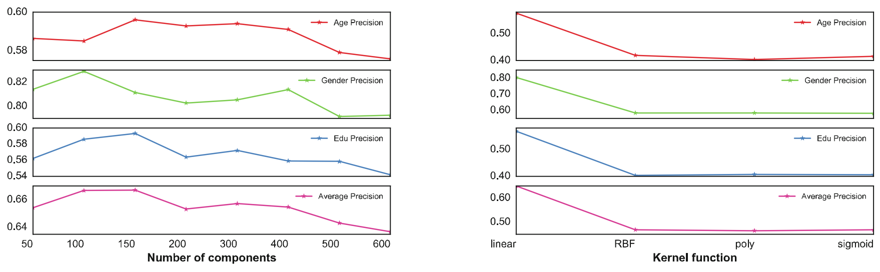


Fig. 3. Effect of parameter tuning. (a) The values of precision with different term frequency thresholds. (b) The values of precision got by the χ^2 algorithm with different number of words.

From Fig. 3(a), we can see that average precision have a peak value when word frequency threshold is 90. As the threshold of term frequency increases, the precision increases at the beginning, and then tends to decrease. The reason is that we need a certain amount of words to get the characteristics of the user. However, too many words may introduce some noise, while too few words can not capture the whole information of users' characteristics. Both of them can reduce the precision of classifiers. Therefore, in the following experiments we choose threshold = 90 in the following experiments.

We have tried another feature selection method - χ^2 test, and the results are shown in Fig. 3(b). To our surprise, this method performs better than term frequency in off-line evaluation but performs worse in online evaluation. We think it may caused by the different distribution of attributes. Therefore, we still use term frequency in the following experiments in terms of generalization ability.

Another parameter is the number of components kept by PCA. Intuitively, more components will contain more information of original data and therefore achieve better performance. However, it is not worth to keep all components because some components only contain trivial information. Figure 4(a) shows the performance of SVM classifier by varying the number of components kept by PCA. The value of precision increases initially, and then decreases. The performance of component = 100 and component = 150 have similar performance, but component = 150 will cost more computation time. Therefore, in the following experiments we only use the component = 100 for classification.



(a) (b)

Fig. 4. (a) The values of precision got by the PCA algorithm with different number of components. (b) The values of precision got by the SVMs with different kernel functions.

Comparison of Feature Representation Methods. Table 2 shows the results of different feature representation methods by using the SVM classifier with linear kernel. It is clear that no single kind of features produces better classification results than the combination of all the features. However, traditional "bag-of-words" based methods like TF-IDF can still achieve high precision despite its simplicity. Furthermore, combining three different features performs the best and single LDA performs the worst. The poor performance of LDA is due to the fact that it leverages word co-occurrence to get topics from a corpus of documents. Despite its effectiveness on many problems, it shows bad performance when it comes to short texts such as queries due to the scarce word co-occurrence. Therefore, in the following experiments, we use the combination of three kinds of features as the input data for classifiers.

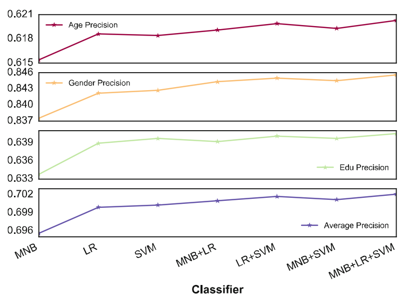
Table 2. Comparison of different feature representation methods

Feature	AgePre	GenPre	EduPre	AvgPre
TF-IDF	59.00%	83.55%	61.06%	67.87%
word2vec	57.34%	83.75%	55.74%	65.61%
LDA	56.45%	78.68%	53.89%	63.01%
TF-IDF+word2vec	61.54%	83.96%	63.28%	69.59%
TF-IDF+LDA	60.08%	83.94%	62.19%	68.74%
TF-IDF+word2vec+LDA	61.84%	84.27%	63.97%	70.03%

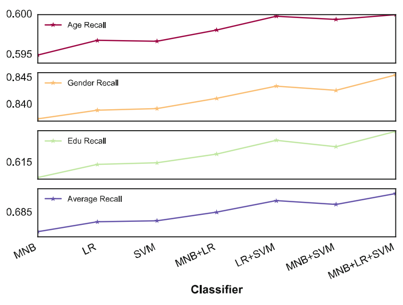
Comparison of SVMs with Different Kernel Functions. To our knowledge, the kernel function can effect the performance of SVM classifier. Therefore, we do several experiments for the sake of finding the SVM classifier with the most appropriate kernel function. Before classification, we use TF-IDF to get the weight of each word, and use word frequency threshold of 90 times to filter out trivial words. Then, we use PCA with 200 components to reduce dimension.

Figure 4(b) shows the performance of SVM classifiers with different kernel functions. Apparently, the SVM classifier with linear kernel greatly outperforms other SVM classifiers. Therefore, we only use SVM with linear kernel for classification in the following experiments.

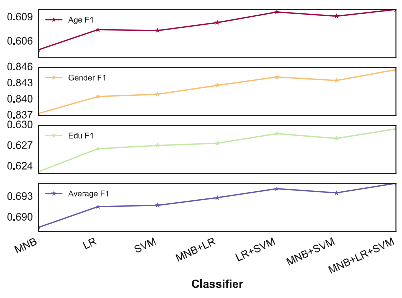
Comparison of Classifiers. We now study the performance of different classifiers including SVM classifier, MNB classifier and LR classifier. We choose these classifiers as level-0 classifiers for computational reasons. Furthermore, to illustrate the benefits of combining different classifiers, we run the stacked generalization algorithm described in Sect. 3. The used level-1 classifier is a SVM classifier with an RBF kernel and other default settings. Given three level-0 classifiers, we can totally obtain seven ensemble classifiers. Figure 5 show the precision, recall and F1-score performance of the three single classifiers and four ensemble classifiers on the testing dataset, respectively.



(a) Precision



(b) Recall



(c) F1-score

Fig. 5. The values of precision, recall and F1-score got by the different classifiers and their combination.

We find several interesting observations from Fig. 5. First, the SVM classifier obtains the highest precision, recall and F1-score among the three classifiers. Second, under all the metrics, ensemble classifiers perform better than any single classifier, which verifies the effectiveness of ensemble learning. Specifically, the ensemble classifier of the three level-0 classifiers performs better than the ensemble classifiers of any two level-0 classifiers, which achieves 70.21% for the precision, 68.82% for the recall, and 69.5% for the F-score on average. When predicting users' gender, the ensemble classifier of the three level-0 classifiers has precision of 84.56%. This finding verifies our conjecture that different kinds of classifiers complement each other, and their combination will enhance the performance. We list the specific value for further comparison in Tables 3, 4 and 5.

Table 3. Comparison of classifiers by precision

Model	AgePre	GenPre	EduPre	AvgPre
MNB	0.6154	0.8375	0.6338	0.6956
LR	0.6186	0.8422	0.6389	0.6999
SVM	0.6184	0.8427	0.6397	0.7003
MNB+LR	0.6191	0.8443	0.6392	0.7010
LR+SVM	0.6199	0.8450	0.6401	0.7017
MNB+SVM	0.6193	0.8445	0.6397	0.7012
MNB+SVM+LR	0.6203	0.8456	0.6405	0.7021

Table 4. Comparison of classifiers by recall

Model	AgeRecall	GenRecall	EduRecall	AvgRecall
MNB	0.5931	0.8162	0.6134	0.6742
LR	0.5968	0.8321	0.6191	0.6827
SVM	0.5967	0.8328	0.6197	0.6831
MNB+LR	0.5981	0.8359	0.6195	0.6845
LR+SVM	0.5998	0.8370	0.6210	0.6859
MNB+SVM	0.5994	0.8365	0.6201	0.6853
MNB+SVM+LR	0.6000	0.8379	0.6217	0.6865

Table 5. Comparison of classifiers by F1-score.

Model	AgeF1	GenF1	EduF1	AvgF1
MNB	0.6040	0.8267	0.6234	0.6847
LR	0.6075	0.8371	0.6288	0.6912
SVM	0.6074	0.8377	0.6295	0.6915
MNB+LR	0.6084	0.8401	0.6292	0.6926
LR+SVM	0.6097	0.8410	0.6304	0.6937
MNB+SVM	0.6092	0.8405	0.6297	0.6931
MNB+SVM+LR	0.6100	0.8417	0.6310	0.6942

5 Discussion

Limitations. The possibility that there may be more than one people using the same account will interfere classifiers and harm the attack performance. For example, different family members will search different information, causing the query record reflect different people's characteristics.

Other Methods to Improve Performance. The key problem is how to automatically find the exact means behind the queries that can represent the users' characteristics. Correctly categorizing the queries has the potential to bring major gains in accuracy. One possible feature representation method is to enrich the queries with external knowledge such as knowledge base (e.g., WordNet), and then map queries to intermediate objects and finally maps them to target categories. However, there is no available knowledge base that can be exploited to categorize Chinese short texts. Furthermore, investigating more classifiers and fully understanding the limitations and advantages of each classifier may also improve the performance.

6 Conclusion

In this paper, we present a query data based inference attack wherein adversaries exploit users' query records to disclose their attributes. In the attack, we first use a combined feature representation method based on TF-IDF, LDA and word2vec to obtain features from users' query records and then employ stacked generalization to combine multiple classification results and build an ensemble classifier. We evaluate the performance of our method on a real-world dataset provided by Sogou and the experimental results demonstrate that our query-based inference attack can effectively predict users' attributes. Based on our research, we can imagine that the development of more sophisticated query-based techniques will pose a great threat to user privacy.

A few fascinating directions for future work include representing features, eliminating the noise in queries, leveraging unsupervised methods, as well as defending against our attribute inference attacks.

Acknowledgments. This work was partly supported by NSFC under No. 61772466, the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. R19F020013, the Technology Project of State Grid Zhejiang Electric Power co. LTD under NO. 5211HZ17000J, the Provincial Key Research and Development Program of Zhejiang, China under No. 2017C01055, the Fundamental Research Funds for the Central Universities, and the Alibaba-ZJU Joint Research Institute of Frontier Technologies.

References

1. Spear Phishing Attacks. http://www.microsoft.com/protect/yourself/phishing/spear.mspx
2. Bartunov, S., Korshunov, A., Park, S.T., Ryu, W., Lee, H.: Joint link-attribute user identity resolution in online social networks. In: Proceedings of the Workshop on Social Network Mining and Analysis in the 6th International Conference on Knowledge Discovery and Data Mining. ACM (2012)
3. Beitzel, S.M., Jensen, E.C., Chowdhury, A., Frieder, O.: Varying approaches to topical web query classification. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 783–784. ACM (2007)
4. Beitzel, S.M., Jensen, E.C., Frieder, O., Grossman, D., Lewis, D.D., Chowdhury, A., Kolcz, A.: Automatic web query classification using labeled and unlabeled training data. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 581–582. ACM (2005)
5. Cao, H., Hu, D.H., Shen, D., Jiang, D., Sun, J.T., Chen, E., Yang, Q.: Context-aware query classification. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 3–10. ACM (2009)
6. Chaabane, A., Acs, G., Kaafar, M.A.: You are what you like! Information leakage through users' interests. In: Network and Distributed System Security Symposium, pp. 1–14 (2012)
7. Dietterich, T.G.: Ensemble methods in machine learning. In: International Workshop on Multiple Classifier Systems, pp. 1–15. Springer (2000)
8. Fang, Q., Sang, J., Xu, C., Hossain, M.S.: Relational user attribute inference in social media. IEEE Trans. Multimed. **17**(7), 1031–1044 (2015)
9. Gong, N.Z., Liu, B.: You are who you know and how you behave: attribute inference attacks via users' social friends and behaviors. In: USENIX Security Symposium, pp. 979–995 (2016)
10. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. Natl. Acad. Sci. **101**(suppl 1), 5228–5235 (2004)
11. Gupta, P., Gottipati, S., Jiang, J., Gao, D.: Your love is public now: questioning the use of personal information in authentication. In: Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, pp. 49–60. ACM (2013)
12. He, J., Chu, W.W., Liu, Z.V.: Inferring privacy information from social networks. In: International Conference on Intelligence and Security Informatics, pp. 154–165. Springer (2006)
13. Hu, J., Wang, G., Lochovsky, F., Sun, J.T., Chen, Z.: Understanding user's query intent with wikipedia. In: Proceedings of the 18th International Conference on World Wide Web, pp. 471–480. ACM (2009)
14. Lindamood, J., Heatherly, R., Kantarcioglu, M., Thuraisingham, B.: Inferring private information using social network data. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009, vol. 10, p. 1145 (2009). https://doi.org/10.1145/1526709.1526899
15. Miskin, J., MacKay, D.J.: Ensemble learning for blind image separation and deconvolution. In: Advances in Independent Component Analysis, pp. 123–141. Springer (2000)

16. Thomas, K., Grier, C., Nicol, D.M.: unFriendly: multi-party privacy risks in social networks. In: International Symposium on Privacy Enhancing Technologies Symposium. LNCS, vol. 6205, pp. 236–252. Springer (2010)
17. Verbaeten, S., Van Assche, A.: Ensemble methods for noise elimination in classification problems. In: International Workshop on Multiple Classifier Systems, pp. 317–325. Springer (2003)
18. Weinsberg, U., Bhagat, S., Ioannidis, S., Taft, N.: BlurMe: inferring and obfuscating user gender based on ratings. In: Proceedings of the 6th ACM Conference on Recommender Systems, pp. 195–202. ACM (2012)
19. Wolpert, D.H.: Stacked generalization. Neural Netw. **5**(2), 241–259 (1992)
20. Xia, R., Zong, C., Li, S.: Ensemble of feature sets and classification algorithms for sentiment classification. Inf. Sci. **181**(6), 1138–1152 (2011)
21. Zhou, Z.H.: Ensemble Methods: Foundations and Algorithms. Chapman and Hall/CRC, New York (2012)