Privacy Attack

SU Hong

What is Privacy Breach in ML?

- The model should reveal no more about the input to which it is applied than would have been known about this input without applying the model.
- If applying the data input to the model will provide more potential knowledge than not using this data input to the model, then it is considered as data breach.

Examples of Privacy Breach

- A simplest case: consider that training a model has uncovered **a high correlation between two attributes X and Y**. Then for members in this population or the model's training set, if we know some member's X value, then we can infer Y value (which may be sensitive data). Vice versa.
  - X: a person's externally observable phenotype feature
  - Y: a person's genetic predisposition to a certain disease

- If data are not applied into this model, given a member's X value, we will know no more information (such as the Y value). Hence, it is a privacy breach caused by the ML model.
- Nowadays there are many cloud "**machine learning as a service**" by Google and Amazon. Privacy breach may be a severe problem.

Two Types of Privacy Attack

- Membership Inference Attack
  - Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017, May). **Membership inference attacks against machine learning models.** In *2017 IEEE symposium on security and privacy (SP)* (pp. 3-18). IEEE.
  - Choquette-Choo, C. A., Tramer, F., Carlini, N., & Papernot, N. (2021, July**). Label-only membership inference attacks.** In *International conference on machine learning* (pp. 1964-1974). PMLR.
- Model Inversion Attack
  - Fredrikson, M., Jha, S., & Ristenpart, T. (2015, October**). Model inversion attacks that exploit confidence information and basic countermeasures.** In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security* (pp. 1322-1333).

Membership Inference Attack

- ***Membership inference attack***
  - Adversarial goal: determine whether or not an individual data instance x^* is part of the training dataset $\mathcal{D}$ for a model
- The attack typically assumes black-box query access to the model

Why studying membership inference attack?

  - Consider a model to learn the link: cancer patient's morphological data <-> reaction to a drug
  - Only knowing a person's morphological data cannot directly tell whether this person has cancer or not.
  - But, if knowing this person's data is used in the training set, then it infers that this person has cancer.

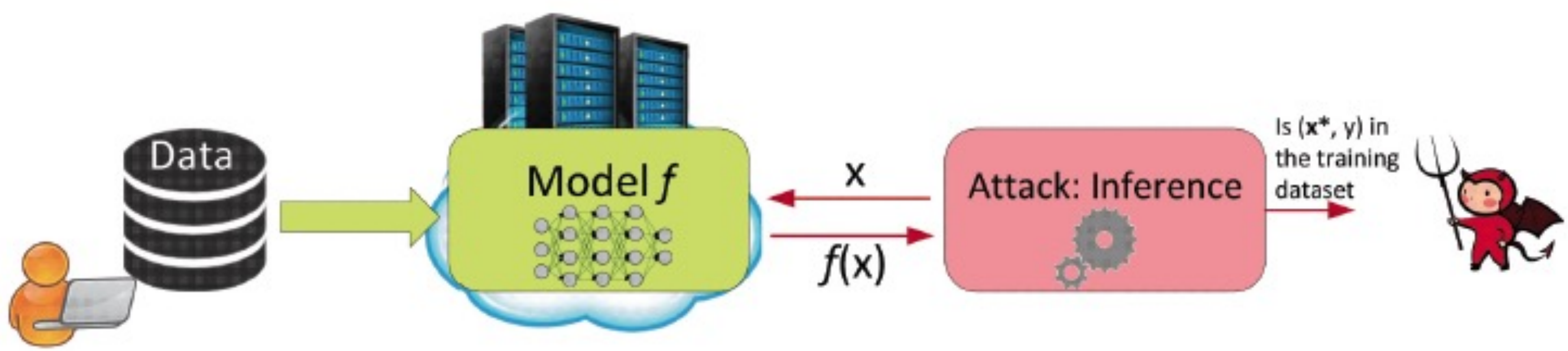


Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Membership Inference Attack

- ***Attack Motivation***
  - ML models often behave differently on the data they were trained on or “see” for the first time. (e.g., overfitting)
- **Objective:**
  - Construct an attack model to recognize such differences of the target model
  - Use these differences to distinguish members from non-members of the training set based solely on the target model’s output.

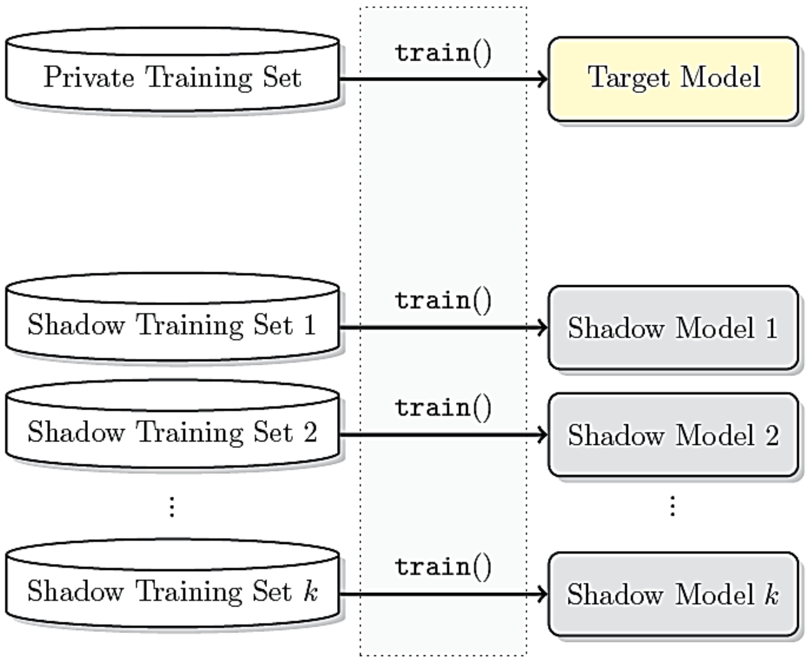
Shadow Training Attack

- [Shokri (2016) Membership Inference Attacks Against Machine Learning Models]()

Observation: Similar models trained on relatively similar data records using the same service behave in a similar way.

Shadow training approach:

- Create several shadow models to substitute the target model
- Each shadow model is trained on a dataset that has a similar distribution (discuss later) as the private training dataset of the target model

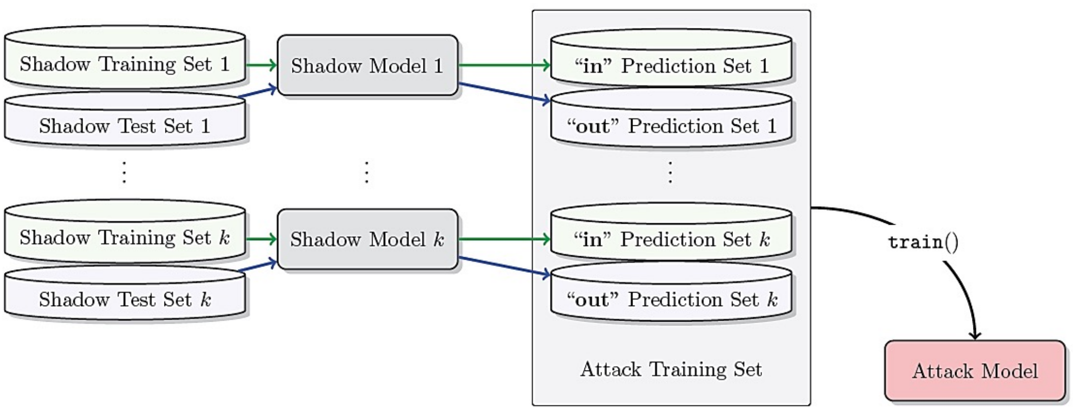


Shadow Training Attack

Observation: Similar models trained on relatively similar data records using the same service behave in a similar way.

Input data of attack models:

- From shadow training set ($\mathbf{x}_{train}$,y_{train})
  - Compute probability vectors $\mathbf{Y}_{train}$ from the shadow models, add (y_{train}, $\mathbf{Y}_{train}$, in) to attack training set.
- From shadow testing set ($\mathbf{x}_{test}$,y_{test})
  - Data in shadow testing set are disjoint from shadow training set (not used to train the shadow model)
  - Compute probability vectors $\mathbf{Y}_{test}$ from the shadow models, add (y_{test}, $\mathbf{Y}_{test}$, out) to attack training set.



- Split the attack training set into several partitions, each associated with one class label.
- Train a separate model for each class label. The attack model input is, for each label y, given **Y**, predicts the *in* or *out* membership for its original **x**.

Shadow Training Attack

- The attack models for each class are afterward used to predict whether individual inputs instances were members of the private training set of the target model
- The assumption in this attack is that the output probability vectors of the shadow models are different for samples that are members of the shadow training sets, in comparison to samples from the shadow test sets
- Experiments showed that increasing the number of shadow models improves the accuracy, but it also increases the computational recourses

Generating training data for shadow model

- Model-based synthesis
- Statistics-based synthesis
- Noisy real data

Generating training data for shadow model

Model-based synthesis

Intuition:

Records that are classified by the target model with high confidence should be statistically similar to the target's training dataset.

Steps:

- Fix a class c
- In each iteration, proposed a new candidate record by changing *k* randomly selected features.

Algorithm 1 Data synthesis using the target model

```
procedure SYNTHESIZE(class : c)
    x ← RANDRECORD( )                          ▷ initialize a record randomly
    y_c^* ← 0
    j ← 0
    k ← k_max
    for iteration = 1 ··· iter_max do
        y ← f_target(x)                        ▷ query the target model
        if y_c ≥ y_c^* then                    ▷ accept the record
            if y_c > conf_min and c = arg max(y) then
                if rand() < y_c then           ▷ sample
                    return x                   ▷ synthetic data
                end if
            end if
            x^* ← x
            y_c^* ← y_c
            j ← 0
        else
            j ← j + 1
            if j > rej_max then                ▷ many consecutive rejects
                k ← max(k_min, ⌈k/2⌉)
                j ← 0
            end if
        end if
        x ← RANDRECORD(x^*, k)                 ▷ randomize k features
    end for
    return ⊥                                   ▷ failed to synthesize
end procedure
```

Generating training data for shadow model

Statistics-based synthesis

- The attackers may have some statistical information about the population of the training data.

Noisy real data

- The attackers may have access to some data that is similar to the training data, but in a ”noisy” version. E.g., not sampled from exactly the same population, or sampled in a non-uniform way.

Mitigation

- **Restrict the prediction vector to top *k* classes**
- **Increase entropy of the prediction vector** $\frac{e^{z_i/t}}{\sum_j e^{z_j/t}}, t > 0$

  (extreme case: t -> inf, output become uniform, no difference in probability, no leaking information)

- **Use regularization.** E.g., L_2 regularization or dropout

Problem comes…

Is it safe by hiding the whole confidence vector?

Label-Only Attack

A naïve baseline attack model (Gap Attack)

- Predict any mis-classified data point as a non-member of the training set.

$$1/2 + (\mathrm{acc}_{\mathrm{train}} - \mathrm{acc}_{\mathrm{test}})/2\ , \qquad (1)$$

Label-Only Attack

Label-Only Membership Inference Attack

Attack intuition

- Compute label only "proxies" by evaluating its robustness to strategic input perturbations of data point *x*.
- Data points that exhibit high robustness are training data points.
- Non-training points are closer to the decision boundary and thus more susceptible to perturbations. (may not be universally true)

Data Augmentation Attack

Label-Only Membership Inference Attack

Attack intuition

- Models trained with data augmentation have the capacity to overfit them.
- Leak more information by the augmented data.

Algorithm (Assume knowing model architecture and training data distribution)

Create a binary MI classifier $f(x; h)$

Given a target point (x_0, y_{true}), $f(x_0; h) = 1$ if x_0 is a training member

Use x_0 to create augmented data points $\{\hat{x}_1, \cdots, \hat{x}_N\}$

Compute $\left(h(x_0), h(\hat{x}_1), \cdots, h(\hat{x}_N)\right) \rightarrow \left(y_0, y_1, \cdots, y_N\right)$

Let $b_i \leftarrow \mathbb{1}(y_{true} = (y_i))$

Apply $f(b_0, \cdots, b_N) \rightarrow \{0, 1\}$ to classify x_0

Decision Boundary Distance Attack

Attack intuition

- Training members are often far away from the decision boundary. If one can estimate the distance of x_0 to the decision boundary, x_0 is highly likely to be a member if the distance is large.
- **Motivation of computing distance (in a binary linear classification case)**

$$\frac{|w^T x + b|}{\|w\|_2}$$

Decision Boundary Distance Attack

Need to estimate the distance to decision boundary to attack!

A white box baseline

- Use the Carlini & Wagner (2017) attack that given (x,y), by adversarial perturbation, find the closest point x' to x such that $\arg\max h(x') \neq y$

Label-only attacks

- HopSkipJump (Chen et al., 2019)

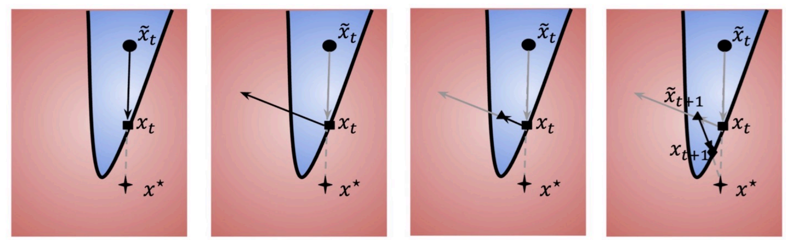


Figure 2: Intuitive explanation of HopSkipJumpAttack. (a) Perform a binary search to find the boundary, and then update $\tilde{x}_t \to x_t$. (b) Estimate the gradient at the boundary point x_t. (c) Geometric progression and then update $x_t \to \tilde{x}_{t+1}$. (d) Perform a binary search, and then update $\tilde{x}_{t+1} \to x_{t+1}$.

Robustness to random noise

- A point's distance to the boundary is directly related to the model's accuracy when it is perturbed by isotropic Gaussian noise.

$$\hat{x}_i = x + \mathcal{N}(0, \sigma^2 \cdot I)$$

Mitigation

Data augmentation suffers!

- Though data augmentation is the common regularization method, models trained with data augmentation are more vulnerable.

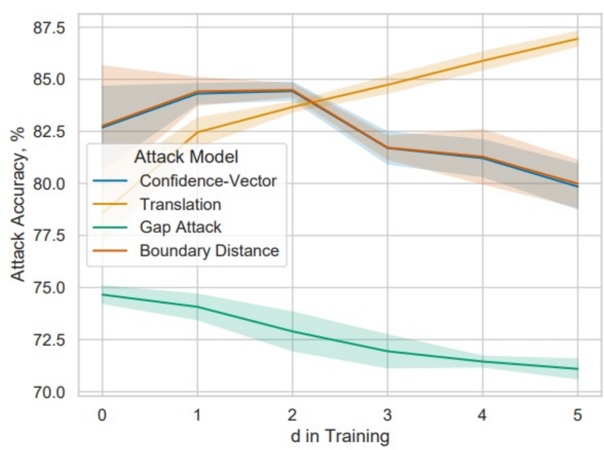
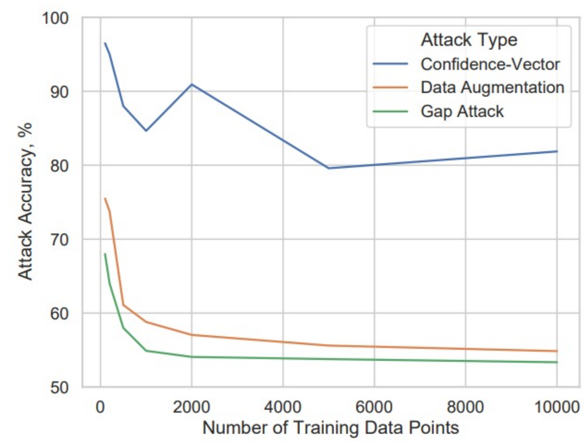
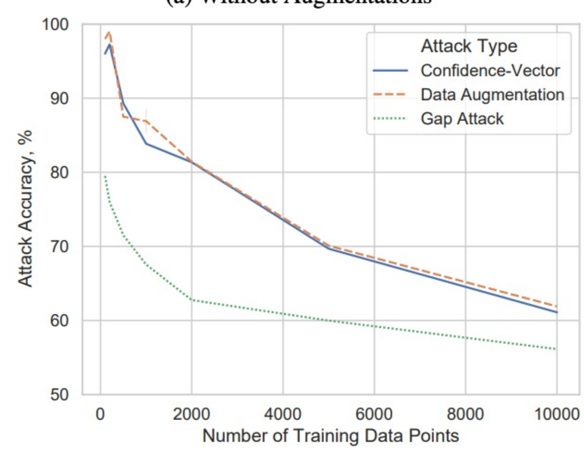


Figure 3. **Accuracy of MI attacks on CIFAR-10 models trained with data augmentation** on a subset of 2500 images. As in our attack, d controls the number of pixels by which images are translated during training, where no augmentation is $d = 0$. For models trained with significant amounts of data augmentation, MI attacks become *stronger* despite it generalizing better.

tained with the simpler pipeline above: *though test accuracy improves, our data augmentation attacks match or outperform the confidence-vector attack.*

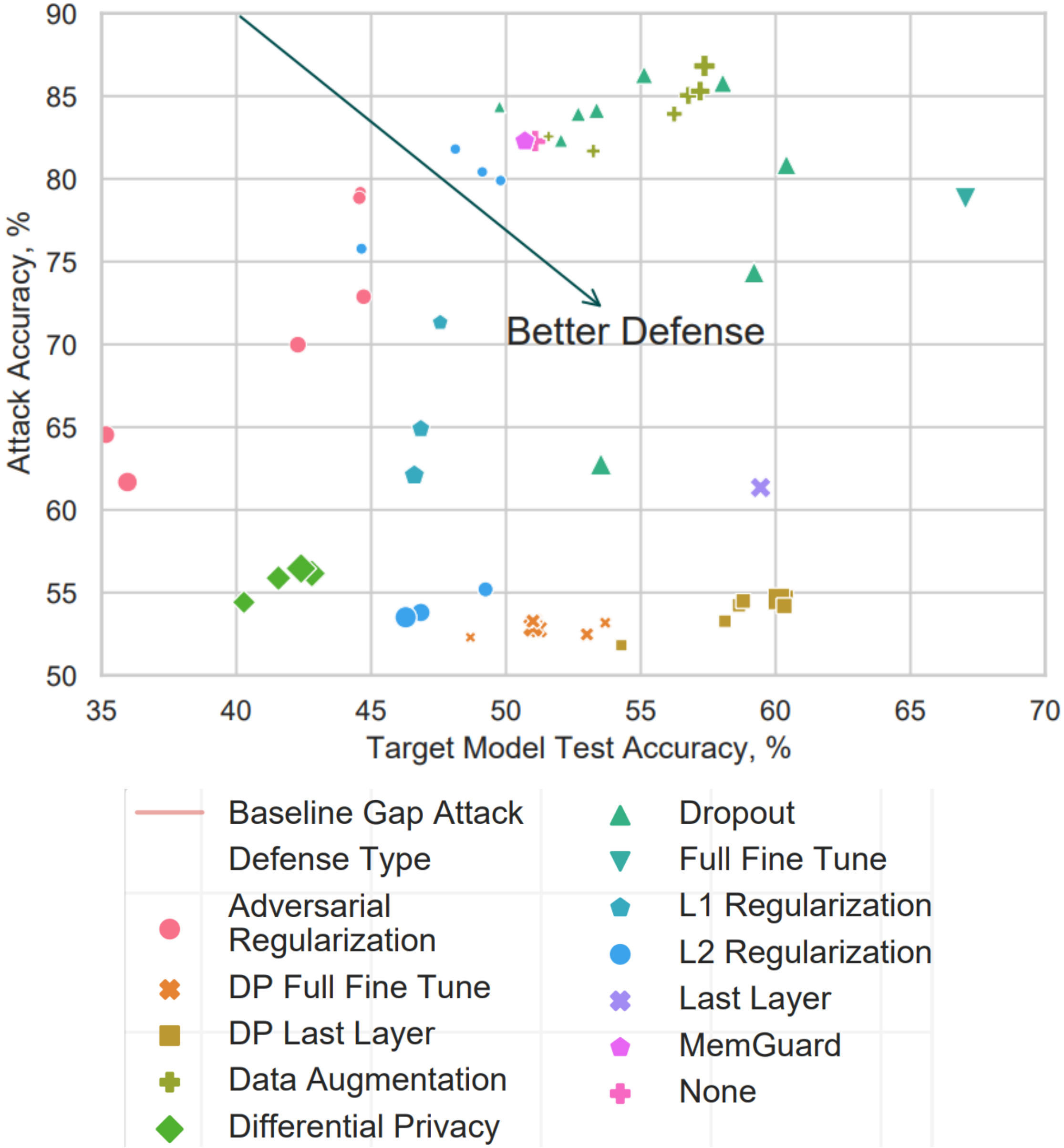


(a) Without Augmentations



(b) With Augmentations

Mitigation



Differential privacy works by adding a controlled amount of "noise" to the data or the results of computations on the data, before releasing or sharing them. By adding this noise, the algorithm produces results that are slightly altered from the true results, in a way that is mathematically guaranteed to not compromise the privacy of any individual in the dataset.

Model Inversion Attack

- ***Model Inversion (MI) Attack***
  - Adversarial goal: recreate certain features of data instances x^* or statistical properties (such as class average of x^*) of the training dataset $\mathcal{D}$ for the model
- A.k.a. attribute inference attack, reconstruction attack, or data extraction attack
- Various attacks have been developed to either recover partial information about the training data (such as sensitive features of the dataset, or typical representatives for specific classes in the dataset) or full data samples

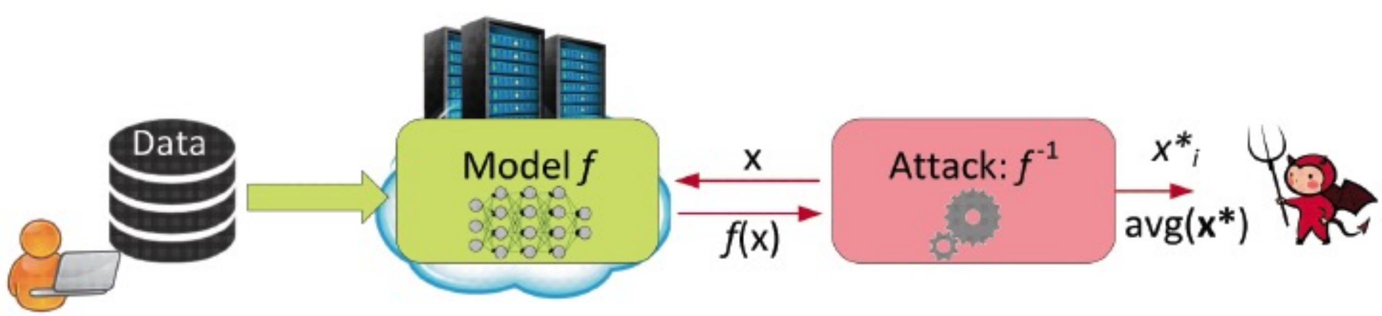


Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Examples of MI Attacks

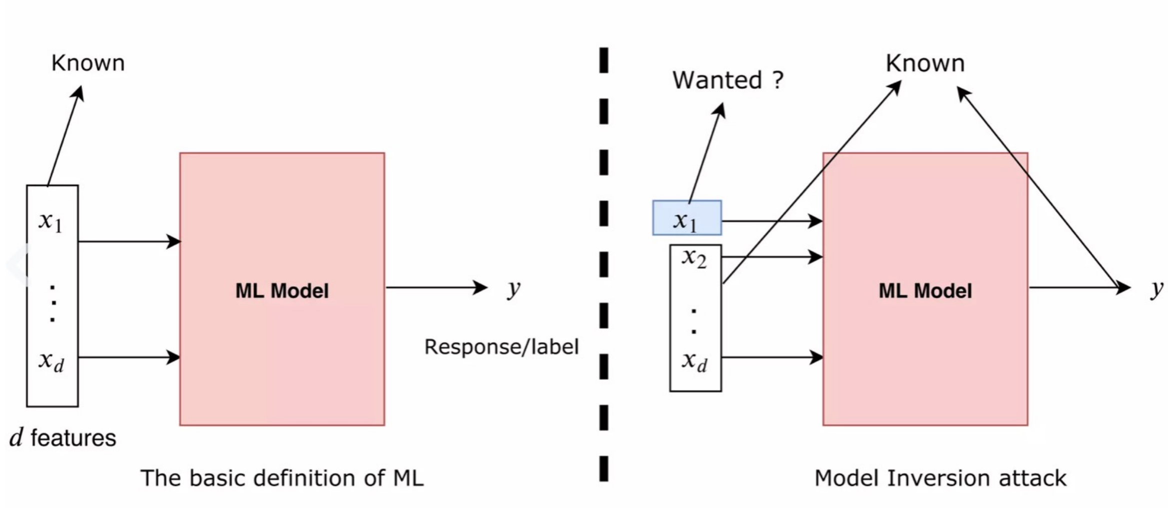
- **Fredrickson (2015) Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures**

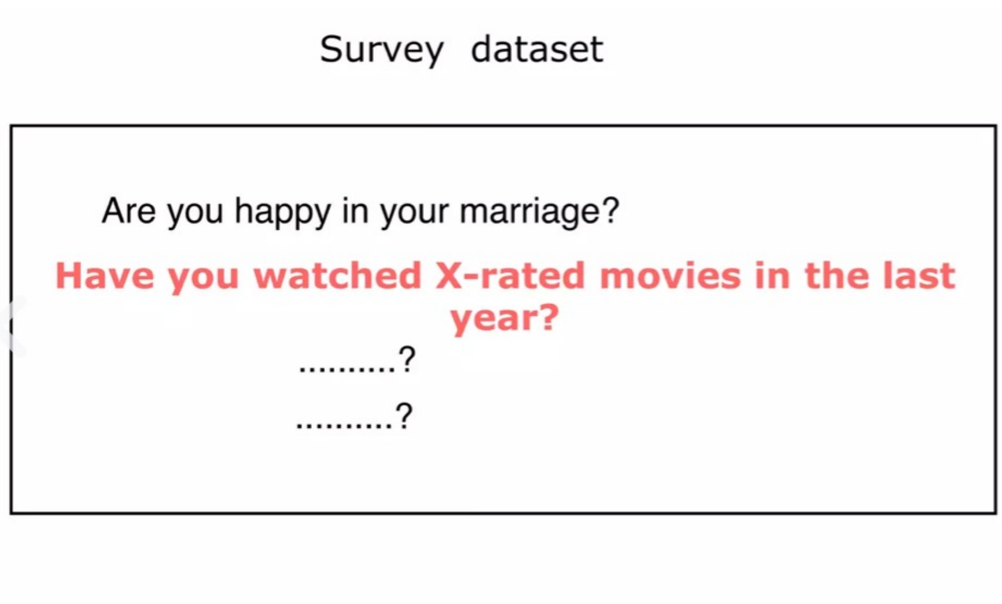
(a) Face recognition by model in- version attack

(b) Training set image of the victim

Figure: The attacker is given only the person's name and access to a facial recognition system that returns a class confidence score.

Examples of MI Attacks





Decision tree model

Fredrikson et al. attack

Purpose of the Attack:

The attack assume the genetic marker as the sensitive attribute x_1. The goal is given auxiliary information side($\mathbf{x}$,y) = (x_2, …, x_p, y) for a patient instance, infer the patient's genetic marker x_1.

1. Input: $\mathbf{z}_K = (x_1, \ldots, x_k, y), f, p_{1,\ldots,d,y}$ 2. Find the *feasible set* $\hat{\mathbf{X}} \subseteq \mathbf{X}$, i.e., such that $\forall \mathbf{x} \in \hat{\mathbf{X}}$ (a) $\mathbf{x}$ matches $\mathbf{z}_K$ on known attributes: for $1 \le i \le k, \mathbf{x}_i = x_i$. (b) f evaluates to y as given in $\mathbf{z}_K$: $f(\mathbf{x}) = y$. 3. If $\lvert\hat{\mathbf{X}}\rvert = 0$, return $\perp$. 4. Return x_t that maximizes $\sum_{\mathbf{x}\in\hat{\mathbf{X}}:\mathbf{x}_t=x_t} \prod_{1\le i\le d} p_i(\mathbf{x}_i)$	1. Input: $\mathbf{z}_K = (x_1, \ldots, x_k, y)$, f, π, $p_{1,\ldots,d,y}$ 2. Find the *feasible set* $\hat{\mathbf{X}} \subseteq \mathbf{X}$, i.e., such that $\forall \mathbf{x} \in \hat{\mathbf{X}}$ (a) $\mathbf{x}$ matches $\mathbf{z}_K$ on known attributes: for $1 \le i \le k$, $\mathbf{x}_i = x_i$. 3. If $\lvert\hat{\mathbf{X}}\rvert = 0$, return $\perp$. 4. Return x_t that maximizes $\sum_{\mathbf{x}\in\hat{\mathbf{X}}:\mathbf{x}_t=x_t} \pi_{y,f(\mathbf{x})} \prod_{1\le i\le d} p_i(\mathbf{x}_i)$
(a) $\mathcal{A}_0$: Model inversion without performance statistics.	**(b)** $\mathcal{A}_\pi$: Model inversion with performance statistics π.

Figure 2: Model inversion algorithm.

$$\pi(y, y') = \mathbf{Pr}\left[\mathbf{z}_y = y \mid f(\mathbf{z}_\mathbf{x}) = y'\right] \qquad (5)$$

p_i is the marginal distribution of x_i, which can be estimated by sampling

Attack on Decision Tree

What is decision tree?

$$f(\mathbf{x}) = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}), \text{ where } \phi_i(\mathbf{x}) \in \{0, 1\}$$

Attack on Decision Tree

Extension on decision tree

$$f(\mathbf{x}) = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}), \text{ where } \phi_i(\mathbf{x}) \in \{0, 1\}$$

Extended version:

$$f(x) = \arg\max_j \left(\sum_{i=1}^{m} w_i[j]\phi_i(\mathbf{x})\right)$$

$$\tilde{f}(\mathbf{x}) = \left[\frac{w_{i^*}[1]}{\sum_i w_{i^*[i]}}, \cdots, \frac{w_{i^*}[|Y|]}{\sum_i w_{i^*[i]}}\right]$$

Attack on Decision Tree

Inverson problem:

Fix a decision tree $f(\mathbf{x}) = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}), \text{ where } \phi_i(\mathbf{x}) \in \{0, 1\}$

Assume the sensitive attribute as x_1. Given auxiliary information side($\mathbf{x}$,y) = (x_2, …, x_d, y), the goal is to inverse the value of x_1. (Can be generalized to more than 1 hidden attributes.)

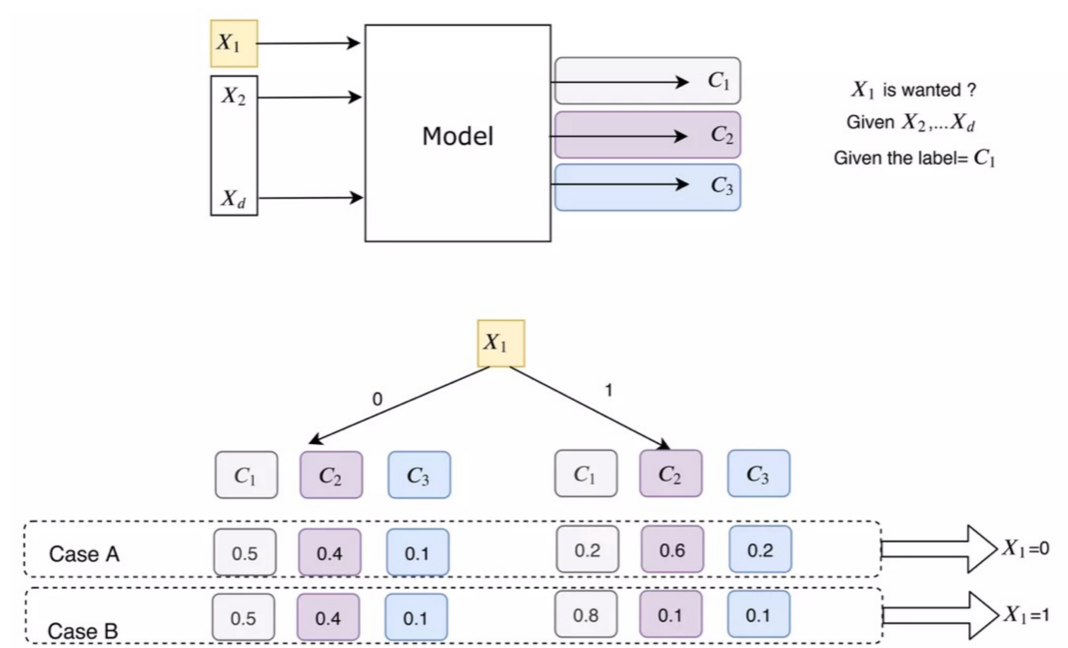
Attack on Decision Tree

Black-box attack:

adversary $\mathcal{A}^f(\mathsf{err}, \mathbf{p}_i, \mathbf{x}_2, \ldots, \mathbf{x}_t, y)$:

for each possible value v of $\mathbf{x}_1$ **do**
    $\mathbf{x}' = (v, \mathbf{x}_2, \ldots, \mathbf{x}_t)$
    $\mathbf{r}_v \leftarrow \mathsf{err}(y, f(\mathbf{x}')) \cdot \prod_i \mathbf{p}_i(\mathbf{x}_i)$
Return $\arg\max_v \mathbf{r}_v$

$$\mathsf{err}(y, y') \propto \Pr[\, f(\mathbf{x}) = y' \mid y \text{ is the true label} \,]$$



Attack on Decision Tree

White-box attack:

Provide more information:

Knows $N = \sum_{i=1}^{m} n_i$,

N is number of samples in the training set,

n_i is number of samples falling in region i.

Define $p_i = \frac{n_i}{N}$

$\phi_i(v) = \mathbb{1}(\exists \mathbf{x}' \in \mathbb{R}^d, \mathbf{x}'_1 = v \wedge \mathbf{x}' \text{falls in region } i)$

$$\begin{aligned}
&\Pr\left[\,\mathbf{x}_1 = v \mid (s_1 \vee \cdots \vee s_m) \wedge \mathbf{x}_K = \mathbf{v}_K\,\right] \\
&\quad\propto \sum_{i=1}^{m} \frac{p_i \phi_i(v) \cdot \Pr\left[\,\mathbf{x}_K = \mathbf{v}_K\,\right] \cdot \Pr\left[\,\mathbf{x}_1 = v\,\right]}{\sum_{j=1}^{m} p_j \phi_j(v)} \\
&\quad\propto \frac{1}{\sum_{j=1}^{m} p_j \phi_j(v)} \sum_{1 \le i \le m} p_i \phi_i(v) \cdot \Pr\left[\,\mathbf{x}_1 = v\,\right] \qquad (1)
\end{aligned}$$

Attack on Decision Tree

White-box attack:

$$
\begin{aligned}
&\Pr[\,\mathbf{x}_1 = v \mid (s_1 \vee \cdots \vee s_m) \wedge \mathbf{x}_K = \mathbf{v}_K\,] \\
&\quad \propto \sum_{i=1}^{m} \frac{p_i \phi_i(v) \cdot \Pr[\,\mathbf{x}_K = \mathbf{v}_K\,] \cdot \Pr[\,\mathbf{x}_1 = v\,]}{\sum_{j=1}^{m} p_j \phi_j(v)} \\
&\quad \propto \frac{1}{\sum_{j=1}^{m} p_j \phi_j(v)} \sum_{1 \le i \le m} p_i \phi_i(v) \cdot \Pr[\,\mathbf{x}_1 = v\,] \qquad (1)
\end{aligned}
$$

$$
P(x_1 = v) = \frac{\sum_{i=1}^{n} [T(x_i) = T(x)] \cdot [x_{i,1} = v]}{\sum_{i=1}^{n} [T(x_i) = T(x)]}
$$

Attack on Facial Recognition Models

Reconstruction attack (white-box setting):

Attackers knows the label (e.g., a person's name) and wish to produce an image of the person.

Algorithm 1 Inversion attack for facial recognition models.

function $\textsc{MI-Face}(\mathit{label}, \alpha, \beta, \gamma, \lambda)$

  $c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \tilde{f}_{label}(\mathbf{x}) + \textsc{AuxTerm}(\mathbf{x})$

  $\mathbf{x}_0 \leftarrow \mathbf{0}$

  for $i \leftarrow 1 \ldots \alpha$ **do**

    $\mathbf{x}_i \leftarrow \textsc{Process}(\mathbf{x}_{i-1} - \lambda \cdot \nabla c(\mathbf{x}_{i-1}))$

    if $c(\mathbf{x}_i) \geq \max(c(\mathbf{x}_{i-1}), \ldots, c(\mathbf{x}_{i-\beta}))$ **then**

      break

    if $c(\mathbf{x}_i) \leq \gamma$ **then**

      break

  return $[\arg\min_{\mathbf{x}_i}(c(\mathbf{x}_i)), \min_{\mathbf{x}_i}(c(\mathbf{x}_i))]$

Mitigation

Decision Tree:

Level at which the sensitive feature occurs may affect the accuracy of the attack.

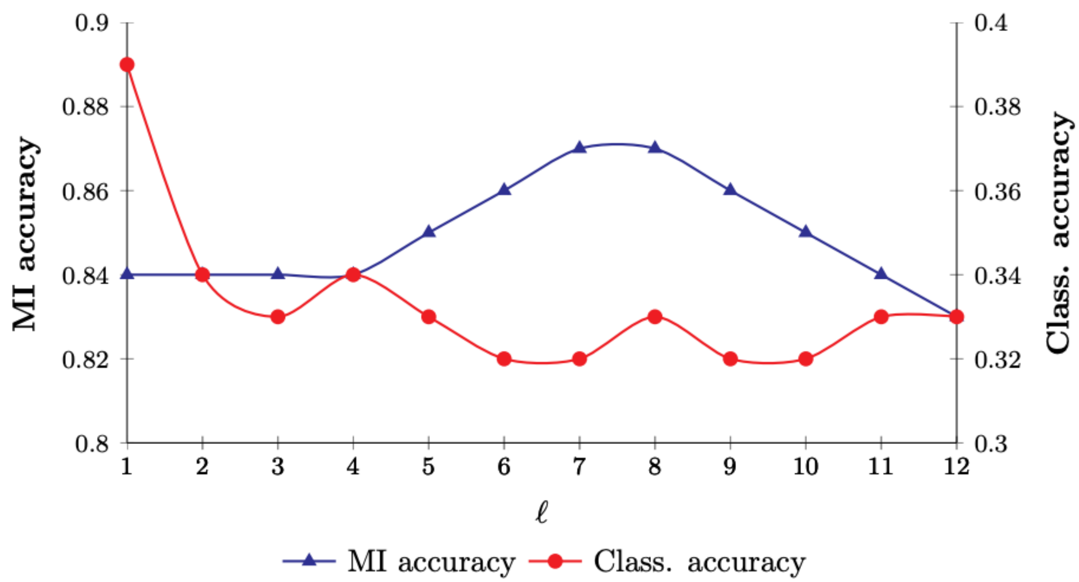


Figure 11: White-box MI vs. classification accuracy on decision trees trained on FiveThirtyEight data with the sensitive feature at each priority level ℓ.

Mitigation

Facial Recognition:

Degrade the quality or precision of the gradient information & confidence scores.

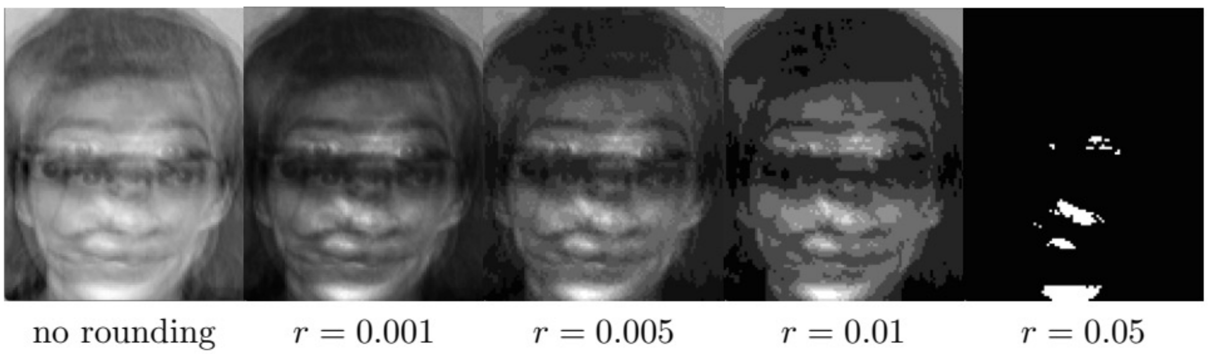


Figure 12: Black-box face reconstruction attack with rounding level r. The attack fails to produce a non-empy image at $r = 0.1$, thus showing that rounding yields a simple-but-effective countermeasure.