ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ФРАНКА
ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ ТА ІНФОРМАТИКИ
КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ МАТЕМАТИКИ

КУРСОВА РОБОТА

на тему:

Аналіз атак на лінійні моделі машинного навчання

студента III курсу
групи ПМп-31
Середовича Віктора

Науковий керівник:
доцент Ю.А.Музичук

Завідуючий кафедри
обчислювальної математики
проф.

Львів — 2020

Зміст

Розділ 1

Вступ

>**TODO** Машинне начання та штучний інтелект активно використовується у різних областях нашого життя, та допомагає у вирішенні таких задач як розпізнавання дорожніх знаків, облич, визначення ризику захворювання та багато іншого. А з поширенням його використання, також збільшуєтья і ризик нападів зловмисників на ці алгоритми, що може привести, до трагічних наслідків. Тому варто досліджувати тему нападів на алгоритми машинного навчання, та знати як захистити свою модель.

В межах теми цієї роботи будуть розглядатись різні атаки на лінійні моделі машинного начання, та методи їх захисту.

1.1 Постановка задачі

Мета даної роботи полягає у тому, щоб дослідити ефективність атак на лінійні моделі машинного навчання, та визначити методи захисту від них.

Виходячи з мети, визначені завдання роботи:

- Практична реалізація та дослідження методів атак
- Визначення методів захисту

1.2 Класифікація атак

Нехай існує класифікатор $f(x) : x \to y$, де $x \in X, y \in Y$, який передбачає значення y для вхідного x. Метою змагального прикладу є знайти такий x^*, який знаходиться в околі x, але хибно класифікується класифікатором. Зазвичай максимальний рівень шуму який може бути в змагальному прикладі може бути не більше за певну L_p норму $\|x^* - x\|_p < \epsilon$, де $p = 1, 2, \infty$.

Нецілеспрямовані атаки (Untargeted Attacks)
Нецілеспрямованими атаками є атаки метою яких є внести в приклад x, який правильно передбачається як $f(x) = y$ незначний шум так, щоб класифікатор зробив хибне передбачення $f(x^*) \neq y$.

Цілеспрямовані атаки (Targeted Attacks)
Цілеспрямованими атаками називають такі атаки, метою яких є внести у вхідний приклад x такі пертурбації щоб класифікатор передбачив якийсь конкретний клас $f(x^*) = y^*$, де y^* є заданою ціллю $y^* \neq y$.

Атаки на закриту модель (White Box attacks) - описує такий сценарій коли в нападника є повний доступ до параметрів і градієнтів моделі.

Атаки на відкриту модель (Black Box attacks) - це атаки при яких в нападника нема доступу до параметрів моделі.

Розділ 2

Модель і датасет

2.1 Мультикласова логістична регресія

В якості лінійного методу машинного навчання, на котрий будуть сдійснюватись атаки буде використовуватись модифікований алгоритм логістичної регресії для мультикласової класифікації.

Нехай маємо набір тренувальних даних:

$$(x^{(1)}, y^{(1)}), \quad (x^{(2)}, y^{(2)}), \quad ... \quad , (x^{(m)}, y^{(m)})$$

$$x \in \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix} \qquad y \in \begin{bmatrix} y_1 \\ \dots \\ y_C \end{bmatrix} \quad y_1, ..., y_C$$

$$X = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x^{(1)} & x^{(2)} & \dots & x^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix} \qquad Y = \begin{bmatrix} \vdots & \vdots & & \vdots \\ y^{(1)} & y^{(2)} & \dots & y^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix}$$

X - матриця в якої кожен стовпець є набором характеристик $i-$ого прикладу, $i = 1, \dots, m$

Y - матриця класів в якої кожен стовпець це масив розмірності C, з одиницею на місті справжнього класу

m - кількість характеристик

n - кількість характеристик в кожному прикладі

C - кількість класів

Задача прикладу $x \in R^n$, знайти $\hat{y} = P(y = 1 \mid x), \quad 0 \leq \hat{y} \leq 1$

$\hat{y} = softmax(\omega^T x + b), \quad$ де $\omega \in R^n, b \in R$ - невідомі параметри

Функція активації матиме вигляд:

$$softmax(z) = \frac{e^z}{\sum_{k=1}^{c} e_k^z} \tag{2.1}$$

Для кожного прикладу з тренувального датасету потрібно обчислити:

$z^{(i)} = \omega^T x^{(i)} + b, \quad$ де $y^{(i)} = softmax(z_i) = \frac{e^{z_i}}{\sum_{k=1}^{c} e_k^z}$, так щоб $\hat{y}^{(i)} \approx y^{(i)}$

В якості функції втрати буде використовуватись функція кросс-ентропії:

$$\xi(y, x) = -\sum_{i=1}^{c} y_i \log(\hat{y}_i) \tag{2.2}$$

Задача полягає в тому щоб знайти параметри $w \in R_x^n, b \in R$ що мінімізують функцію $J(\omega, b)$ Для цього будемо використовувати алгоритм градієнтного спуску

2.2 Датасет

Для аналізу атак на лінійні методи машинного навчання буде використовуватись MNIST база даних яка складаэться з рукописних цифр. На її основі буде здійснюватись тренування моделі та аналіз атак.

Розділ 3

Методи атак

3.1 Атака Шумом

3.2 Fast Gradient Sign методи

Першим методом атаки який буде розглядатись є методу швидкого градієнту (Fast Gradient Sign Method).

3.2.1 FGSM

Першим методом атаки який буде розглядатись є методу швидкого градієнту (Fast Gradient Sign Method).

Ідея цього методу полягає в тому, щоб знайти такий adversarial приклад x^* який максимізую функцію втрати $J(x^*, y)$, де J є функцією кросс-ентропії.

$$X^* = X + \epsilon \cdot sign(\Delta_x J(x, y))$$

$$\Delta_x J(x, y)$$

Обчислення градієнту

Для методу логістичної регресії необхідно обчислити градієнт від функції кросс-ентропії(посилання).

$$z_i = \omega^T x^{(i)} + b \quad \hat{y}_i = a_i = softmax(z_i) \quad softmax(z_i) = \frac{e^{z_i}}{\sum_{k=1}^{c} e_k^z}$$

Похідна softmax функції

Якщо $i = j$,

$$\frac{\partial \hat{y}_i}{\partial z_j} = \frac{\partial \frac{e^{z_i}}{\sum_{k=1}^{c} e_k^z}}{\partial z_j} = \frac{e^{z_i} \sum_{k=1}^{c} e^{z_k} - e^{z_j} e^{z_i}}{(\sum_{k=1}^{c} e^{z_k})^2} =$$
$$= \frac{e^{z_j}}{\sum_{k=1}^{c} e^{z_k}} \times \frac{(\sum_{k=1}^{c} e^{z_k} - e^{z_j})}{\sum_{k=1}^{c} e^{z_k}} = y_i(1 - y_j)$$

Якщо $i \neq j$,

$$\frac{\partial \hat{y}_i}{\partial z_j} = \frac{\partial \frac{e^{z_i}}{\sum_{k=1}^{c} e_k^z}}{\partial z_j} = \frac{0 - e^{z_j} e^{z_i}}{\sum_{k=1}^{c} e^{z_k}} \times \frac{e^{z_i}}{\sum_{k=1}^{c} e^{z_k}} = -y_i y_j$$

Функція кросс-ентропії

$$\xi(y, x) = -\sum_{i=1}^{c} y_i \log(\hat{y}_i)$$

Функція кросс-ентропії

$$\frac{\partial \xi}{\partial z_i} = -\sum_{j=1}^{c} \frac{\partial y_j \log(\hat{y}_j)}{\partial z_i} = -\sum_{j=1}^{c} y_j \frac{\partial \log(\hat{y}_j)}{\partial z_i} = -\sum_{j=1}^{c} y_j \frac{1}{\hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial z_i} =$$
$$= -\frac{y_i}{\hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_i} - \sum_{j \neq i} \frac{y_j}{\hat{y}_j} \frac{\partial \hat{y}_j}{\partial z_i} = -\frac{y_i}{\hat{y}_i} \hat{y}_i(1 - \hat{y}_i) - \sum_{j \neq i} \frac{y_j}{\hat{y}_j}(-\hat{y}_i \hat{y}_j) =$$
$$= -y_i + y_i \hat{y}_i + \sum_{j \neq i} y_j \hat{y}_i = -y_i + \sum_{j=1}^{c} y_j \hat{y}_i = \hat{y}_i - y_i$$

, i = 1, ..., C

Обчислення урадієнту "має бути вище"

$$\frac{\partial \xi}{\partial x_i} = \frac{\partial \xi}{\partial z_i} \frac{\partial z_i}{\partial x_i} = (\hat{y}_i - y_i) \frac{\partial(\omega^T x_i + b)}{\partial x_i} = (\hat{y}_i - y_i) x_i = dz \cdot x$$

Обчислення урадієнту "має бути вище"

Algorithm 1 $I - FGSM$

Input: Приклад x, значення цілі y, класифікатор f
Input: значення пертурбації ϵ, значення цілі y, кількість ітерації T.
Output: Adversarial x^* з нормою $\|x^* - x\|_{inf} \leq \epsilon$;
$a = \epsilon/T$;
$x^* = x$;
for $t = 0 \quad to \quad T - 1$ **do**
    $x^* = x^* + \alpha \cdot sign(\Delta_x J(x, y))$;
end for
return $x^* = x^*_T$.

3.2.2 T-FGSM

3.2.3 I-FGSM

3.2.4 MI-FGSM

Algorithm 2 $MI - FGSM$

Input: Приклад x, значення цілі y, класифікатор f.;
Input: значення пертурбації ϵ, значення цілі y, кількість ітерації T.;
Output: Adversarial x^* з нормою $\|x^* - x\|_{inf} \leq \epsilon$;
$a = \epsilon/T$;
$g0 = 0$; $x^*_0 = x$;
for $t = 0 \quad to \quad T - 1$ **do**
    $g_{t+1} = \mu \cdot g_t + \frac{\Delta_x J(x^*, y)}{\|\Delta_x J(x^*, y)\|_1}$;
    $x^*_{t+1} = x^*_t + \alpha \cdot sign(g_{t+1})$;
end for
return $x^* = x^*_T$.

3.3 DeepFool

3.4 Нецілеспрямовані атаки

Основна частина даної роботи полягала у написанні програми. Нижче наводимо основний алгоритм її роботи, на мові C:

```
#include &lt;stdio.h&gt;
int main()
{
```

Algorithm 3 *DeepFool*

Input: Приклад x, значення цілі y, класифікатор f.;
Input: значення цілі y, кількість ітерації T.;
Output: Adversarial x^* з нормою $\|x^* - x\|_{inf} \leq \epsilon$;
$x_0 = x$; $i = 0$;
while $\hat{f}(x_i) = \hat{f}(x_0)$ **do**
    for $k \neq \hat{f}(x_0)$ **do**
        $w'_k = \Delta f_k(x_i) - \Delta f_{\hat{k}(x_0)}(x_i)$;
        $f'_k = \Delta f_k(x_i) - \Delta f_{\hat{k}(x_0)}(x_i)$;
    end for
    $\hat{l} = \arg\min_{k \neq \hat{f}(x_0)} \frac{|f'_k|}{\|w'_k\|_2}$;
    $r_i = \frac{|f'_{\hat{l}}|}{\|w'_{\hat{l}}\|_2^2} w'_{\hat{l}}$;
    $x_{i+1} = x_i + r_i$;
    $i = i + 1$;
end while
return $x^* = x^*_T$.

```
printf("Hello, world!\n");
return 0;
}

```

[language=Python]

```
def fit(self, X, w, b, y, alpha, max_iters, predict_func):
  # Check that X and y have correct shape
  self.w = w
  self.b = b

  self.y_ = np.expand_dims(y.T, axis=1)
  self.X_ = X.T

  self.num_iters = 0
  self.X_ = self._gradient_descent(self.X_, self.y_, self.w, self.b,
 alpha, max_iters, predict_func)

  def _cost_function(self, X, Y, A):
  m = X.shape[0]
  if m == 0:
  return None

  J = (1 / m) * np.sum(-Y * np.log(A) - (1 - Y) * np.log(1 - A))
  return J

```

Розділ 4

Альтернативні рішення

Деякі дослідники пишуть свої роботи в програмах типу Microsoft Word. Але то не є труйово[1].

Розділ 5

Висновок

Дана робота містить значний мій вклад, і перевершує попередні досягнення в багатьох напрямках. Окрім того, даний напрямок досліджень має значні перспективи подальшого розвитку. (Особливо добре було б, якби хтось вирішив проблему кирилиці в listings).

Бібліографія

[1] Вікіпідручник *Як написати курсову?* (http://uk.wikibooks.org/wiki/%D0%AF%D0%BA_%D0%B2%D1%87%D0%B8%D1%82%D0%B8%D1%81%D1%8C_%D0%BA%D1%80%D0%B0%D1%89%D0%B5%3F/%D0%9A%D1%83%D1%80%D1%81%D0%BE%D0%B2%D1%96)