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

1 Вступ **2**

1.1 Постановка задачі 2

2 Тренування моделі **3**

2.1 Модифікована логістична регресія 3

3 Класифікація атак **5**

3.1 Метод Швидкого Градієнту 5

3.2 Нецілеспрямовані атаки 6

4 Альтернативні рішення **8**

5 Висновок **9**

Література **10**

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

Розділ 2

Тренування моделі

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

2.1 Модифікована логістична регресія

Задача прикладу $x \in R^{n_x}$, знайти $\hat{y} = P(y = 1 \mid x), 0 \leq \hat{y} \leq 1$
Шукатимемо у вигляді $\hat{y} = \sigma(\omega^T x + b)$, де

- параметри $\omega \in R^{n_x}, b \in R$ - невідомі, потрібно знайти оптимальні для для даної задачі
- $\sigma(z) = \frac{1}{1+e^{-z}}$ - сигмоїд

Для кожного прикладу з тренувального датасету потрібно обчислити:

- $z^{(i)} = \omega^T x^{(i)} + b$
- $y^{(i)} = \sigma^{(i)}(z^{(i)})$

де $\sigma(z^{(i)}) = \frac{1}{1+e^{-z^{(i)}}}$, так щоб $\hat{y}^{(i)} \approx y^{(i)}$
Для кожного прикладу визначена функція втрати:

$$L(\hat{y}^{(i)}, y^{(i)}) = -y^{(i)} \log\left(\hat{y}^{(i)}\right) - (1 - y^{(i)}) \log\left(1 - \hat{y}^{(i)}\right)$$

На всіх прикладах обчислюємо штрафну функцію як:

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

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 градієнтний спуск

Розділ 3

Класифікація атак

3.1 Метод Швидкого Градієнту

Першим методом атаки який буде розглядатись є методу швидкого градієнту (Fast Gradient Sign Method). Ідея полягає в тому ..=.

$$z^{(i)} = \omega^T x^{(i)} + b \quad \hat{y}^{(i)} = a^{(i)} = \sigma(z^{(i)}) \quad \sigma(z^{(i)}) = \frac{1}{1 + e^{-z^{(i)}}}$$

$$L(\hat{y}^{(i)}, y^{(i)}) = -y^{(i)} \log\Big(\hat{y}^{(i)}\Big) - (1 - y^{(i)}) \log\Big(1 - \hat{y}^{(i)}\Big)$$

$$da^{(i)} = \frac{\partial L(a^{(i)}, y^{(i)})}{\partial a^{(i)}} = -y^{(i)} \cdot \frac{1}{a^{(i)}} - (1 - y^{(i)}) \cdot \frac{1}{1 - a^{(i)}} = -\frac{y^{(i)}}{a^{(i)}} + \frac{1 - y^{(i)}}{1 - a^{(i)}}$$

$$\begin{aligned} dz^{(i)} &= \frac{\partial L(a^{(i)}, y^{(i)})}{\partial a^{(i)}} \cdot \frac{\partial a^{(i)}}{\partial z^{(i)}} = -\frac{y^{(i)} a^{(i)}(1 - a^{(i)})}{a^{(i)}} + \frac{(1 - y^{(i)}) a^{(i)}(1 - a^{(i)})}{1 - a^{(i)}} = \\ &= -y^{(i)} + a^{(i)} y^{(i)} + a^{(i)} - y^{(i)} a^{(i)} = a^{(i)} - y^{(i)} \end{aligned}$$

$$dx_j^{(i)} = \frac{\partial L(a^{(i)}, y^{(i)})}{\partial x_j^{(i)}} \cdot \frac{\partial z^{(i)}}{\partial x_j^{(i)}} = (w^T)_j^{(i)} (a^{(i)} - y^{(i)}) = (w^T)_j^{(i)} dz$$

3.2 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()
{
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

```

Algorithm 1: How to write algorithms

```
Result: Write here the result
initialization;
while While condition do
    instructions;
    if condition then
        instructions1;
        instructions2;
    else
        instructions3;
    end
end
```

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).

Бібліоґрафія

[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)