ЛАБОРАТОРНА РОБОТА № 8

«Ресурси Keras. TensorFlow. Навчання лінійної регресії».

Мета: Дослідження ресурсу Keras і TensorFlow. Застосування TensorFlow.

Хід роботи

Завдання: Використовуючи засоби TensorFlow, реалізувати код наведений нижче та дослідити структуру розрахункового алгоритму. Для виконання розрахунків, можна використовувати онлайн – середовище google – colab (перехід за посиланням: http://neuralnetworksanddeeplearning.com/chap4.html)

Лістинг програми:

```
import numpy as np
import tensorflow.compat.v1 as tf

tf.disable_v2_behavior()  # Використовуємо TensorFlow 1.x у середовищі 2.x

# Генеруємо дані для навчання
x_data = np.random.rand(1000).astype(np.float32)  # 1000 випадкових точок
y_data = 2.0 * x_data + 1.0 + np.random.normal(0, 2, x_data.shape)  # y = 2x + 1 +
шум

# Оголошення placeholder для X та y
X = tf.placeholder(tf.float32, shape=[None, 1], name="X")  # Матриця розмірності
(міні-батч × 1)
y = tf.placeholder(tf.float32, shape=[None], name="y")  # Вектор довжини розмір
міні-батча

# Ініціалізація параметрів моделі
k = tf.Variable(tf.random.normal([1]), name="k")  # k ініціалізується нормальним
розподілом
b = tf.Variable(tf.zeros([1]), name="b")  # b початково нульове

# Модель лінійної регресії
y_pred = tf.squeeze(tf.matmul(X, tf.reshape(k, [-1, 1])) + b)  # Передбачення мо-
делі

# Функція втрат
loss = tf.reduce_mean(tf.square(y_pred - y))

# Оптимізатор
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.1)
train_op = optimizer.minimize(loss)

# Ініціалізація змінних
init = tf.global_variables_initializer()

# Розмір міні-батчу
batch_size = 100
```

					ДУ «Житомирська політехніка».24.122.09.000 – Лр8		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Марчук Н.А.			Звіт з лабораторної роботи	Літ.	Арк. 1 / Аркушів
Перевір.		Маєвський О.В.					
Керівник						ФІКТ Гр. КН-21-1[1]	
Н. контр.							
Зав. каф.							

```python
# Функція для вибору міні-батчу
def get_batch(x, y, batch_size):
    indices = np.random.choice(len(x), batch_size)
    return x[indices].reshape(-1, 1), y[indices]


# Тренування моделі
with tf.Session() as sess:
    sess.run(init)

    for epoch in range(2001):
        x_batch, y_batch = get_batch(x_data, y_data, batch_size)
        _, loss_value, k_value, b_value = sess.run(
            [train_op, loss, k, b],
            feed_dict={X: x_batch, y: y_batch}
        )

        if epoch % 100 == 0:
            print(f"Епоха {epoch}: втрата={loss_value:.4f}, k={k_value[0]:.4f},
b={b_value[0]:.4f}")
```

Виконання програми:

```
Епоха 0: втрата=10.5871, k=0.0458, b=0.4735
Епоха 100: втрата=3.8438, k=1.7868, b=1.1637
Епоха 200: втрата=4.3645, k=1.9981, b=1.0643
Епоха 300: втрата=3.9306, k=2.1164, b=1.0176
Епоха 400: втрата=3.6749, k=2.2057, b=0.9481
Епоха 500: втрата=3.3587, k=2.2359, b=0.8880
Епоха 600: втрата=4.5355, k=2.2240, b=1.0006
Епоха 700: втрата=4.9841, k=2.1202, b=0.9442
Епоха 800: втрата=3.9010, k=2.1773, b=0.8356
Епоха 900: втрата=4.0620, k=2.3153, b=0.8825
Епоха 1000: втрата=3.6823, k=2.1806, b=0.9528
Епоха 1100: втрата=3.8126, k=2.2818, b=0.9794
```

```
Епоха 1200: втрата=4.0564, k=2.2866, b=0.8594
Епоха 1300: втрата=3.3904, k=2.2803, b=0.9808
Епоха 1400: втрата=4.9532, k=2.0750, b=0.8521
Епоха 1500: втрата=4.3895, k=2.2215, b=0.9446
Епоха 1600: втрата=3.7443, k=2.1104, b=0.7287
Епоха 1700: втрата=4.4301, k=2.1737, b=0.8572
Епоха 1800: втрата=3.6537, k=2.1722, b=0.7842
Епоха 1900: втрата=3.8527, k=2.1306, b=0.9159
Епоха 2000: втрата=3.9089, k=2.0296, b=0.9212
```

Посилання на ГітХаб: https://github.com/Kn211mna/AI-YT

Висновок: в ході виконання лабораторної роботи опрацював спеціалізовані бібліотеки та мову програмування Python дослідити методи неконтрольованої класифікації даних у машинному навчанні.