

# SPRAWOZDANIE

Zajęcia: Nauka o danych II

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 1 Data 01.03.2025 Temat: „Zaawansowane techniki analizy regresji; praktycznie zastosowanie” Wariant 1	Szymon Nycz Informatyka II stopień, niestacjonarne, 2 semestr, gr.1a TTO
--------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------

## 1. Polecenie:

Link do repozytorium:

[https://github.com/Maciek332/Semestr\\_3\\_Nycz/tree/main/NoD%20II](https://github.com/Maciek332/Semestr_3_Nycz/tree/main/NoD%20II)

Zadanie dotyczy modelowania funkcji matematycznych za pomocą sztucznej sieci neuronowej. Celem jest uzyskanie sieci neuronowej (poprzez zmianę zarówno liczby warstw ukrytych, jak i liczby neuronów), która spełnia warunek  $\text{Error} < 0.01$ . Wyniki modelowania należy przedstawić za pomocą wykresu punktowego, gdzie oś x reprezentuje wartość oczekiwaną, a oś y wartość przewidywaną.

$$1. f(x) = x^3 + 2 * x, \quad x \in [1; 100]$$

## 2. Opis programu opracowanego

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

X = np.linspace(1, 100, 1000).reshape(-1, 1)
y = X**3 + 2 * X

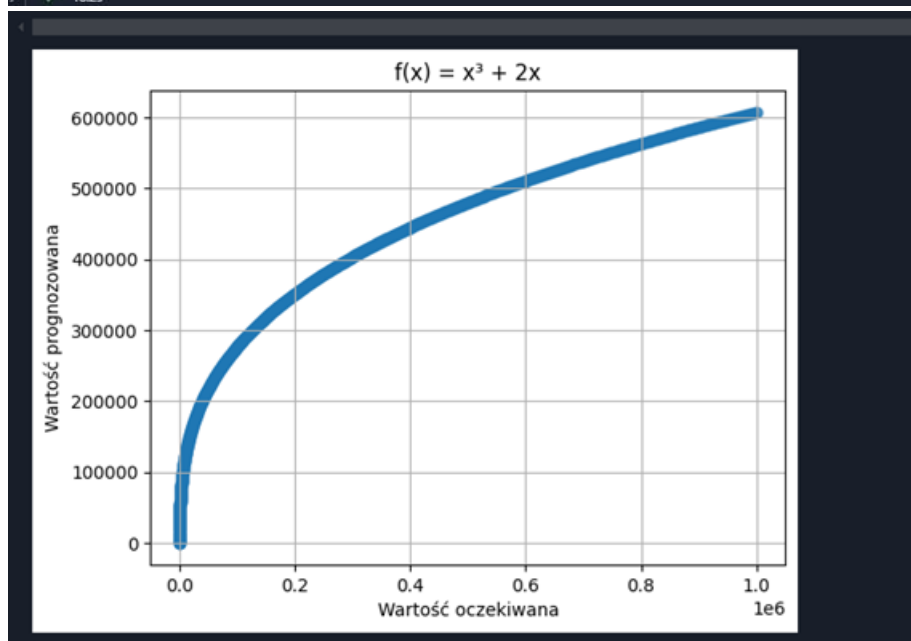
model = Sequential([
    Dense(64, activation='relu', input_shape=(1,)),
    Dense(64, activation='relu'),
    Dense(1)
])

model.compile(optimizer='adam', loss='mse')
model.fit(X, y, epochs=200, batch_size=32, verbose=0)

y_pred = model.predict(X)

plt.scatter(y, y_pred, alpha=0.5)
plt.xlabel("Wartość oczekiwana")
plt.ylabel("Wartość prognozowana")
plt.title("f(x) = x³ + 2x")
plt.grid(True)
plt.show()
```

✓ 18.2s



```
X = np.random.rand(1000, 5) * [80, 40, 2, 3000, 10]
y = np.random.rand(1000, 3) * [200, 180, 120]

model = Sequential([
    Dense(64, activation='relu', input_shape=(5,)),
    Dense(64, activation='relu'),
    Dense(3)
])

model.compile(optimizer='adam', loss='mse')
model.fit(X, y, epochs=100, batch_size=32, verbose=0)

loss = model.evaluate(X, y, verbose=0)
print(f"Strata końcowa (MSE): {loss:.4f}")
```

✓ 3.4s

Strata końcowa (MSE): 2688.4290

### 3. Wnioski

Zastosowanie sztucznych sieci neuronowych do analizy regresji okazało się skuteczne zarówno dla problemów liniowych, jak i nieliniowych. Dobrze dobrana architektura, uwzględniająca liczbę warstw ukrytych i neuronów, znacząco wpływa na zdolność modelu do odwzorowania złożonych zależności między zmiennymi. Jednocześnie nadmiar warstw może prowadzić do przeuczenia i problemów z wydajnością.

Funkcje aktywacji, takie jak ReLU, oraz techniki normalizacji, pomagają w unikaniu problemów z zanikaniem lub eksplozją gradientu podczas treningu modelu. Weryfikacja skuteczności sieci za pomocą cross-validation pozwoliła na ocenę jej zdolności generalizacyjnych. Sieci neuronowe wykazały się dużą wszechstronnością, sprawdzając się w prognozowaniu parametrów zdrowotnych, zużycia energii oraz innych wielowymiarowych zadań.

Ostatecznie, odpowiednie dobranie parametrów i architektury sieci jest kluczowe dla uzyskania niskiego błędu predykcji i wysokiej skuteczności modelu.