# Czym jest sieć neuronowa?

## Oto nasza pierwsza sieć neuronowa.

Aby uruchomić sieć neuronową, otwieramy notatnik Jupyter i uruchamiamy poniższy kod:

**Sieć**

```
weight = 0.1

def neural_network(input, weight):
    prediction = input * weight
    return prediction
```

Następnie uruchamiamy poniższy fragment:

**Jak użyć sieci, aby coś przewidzieć**

```
number_of_toes = [8.5, 9.5, 10, 9]

input = number_of_toes[0]

pred = neural_network(input,weight)
print(pred)
```

Właśnie zbudowaliśmy naszą pierwszą sieć neuronową i użyliśmy jej do przewidywań! Gratulacje! Ostatni wiersz powoduje wypisanie prognozy (`pred`). Powinna ona wynosić 0.85. Zatem czym jest sieć neuronowa? Na razie jest to jedna lub więcej wag, przez które możemy mnożyć *dane wejściowe*, aby uzyskać *prognozę*.

### Czym są dane wejściowe?

Jest to liczba, którą zarejestrowaliśmy gdzieś w rzeczywistym świecie. Zazwyczaj jest to coś łatwo dostępnego, jak dzisiejsza temperatura, średnia uderzeń baseballisty (BA) czy wczorajsza cena jakiejś akcji.

### Czym jest prognoza?

*Prognoza* jest tym, co sieć neuronowa może nam powiedzieć *na podstawie otrzymanych danych wejściowych*, na przykład „dla podanej temperatury istnieje **0%** prawdopodobieństwa, że ludzie będą dziś nosić dresy" albo „dla podanego BA danego gracza istnieje **30%** prawdopodobieństwa, że uderzenie doprowadzi do *home run*", albo „na podstawie wczorajszej ceny akcji dzisiejsza cena będzie wynosić **101,52**".

### Czy te prognozy są zawsze poprawne?

Nie. Sieci neuronowe będą popełniać błędy, ale mogą się na nich uczyć. Dla przykładu jeśli uzyskana prognoza będzie zbyt wysoka, sieć skoryguje swoją wagę, aby następnym razem podać niższą prognozę i *vice versa*.

**Jak sieć się uczy?**

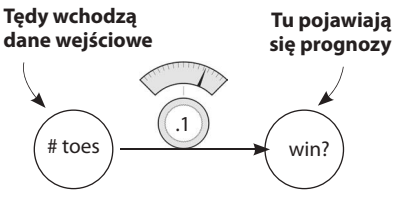
Metodą prób i błędów! Najpierw próbuje dokonać jakiejś prognozy. Następnie może stwierdzić, czy prognoza ta była zbyt wysoka, czy zbyt niska. Następnie zmienia wagę (w górę lub w dół), aby przewidywać bardziej dokładnie następnym razem, gdy otrzyma te same dane wejściowe.

# Co robi ta sieć neuronowa?

## Mnoży dane wejściowe przez wagę. „Skaluje" dane wejściowe o określony czynnik.

W poprzednim podpunkcie wykonaliśmy pierwszą prognozę przy użyciu sieci neuronowej. W swej najprostszej postaci sieć neuronowa wykorzystuje siłę *mnożenia*. Przyjmuje wejściową obserwację (w tym przypadku liczbę 8,5) i *mnoży* ją przez wagę. Gdyby waga była równa 2, wówczas sieć neuronowa *podwoiłaby dane wejściowe*. Jeśli waga to 0,01, sieć *podzieli* dane wejściowe przez 100. Jak można zauważyć, niektóre wartości wagi *powiększają*, a inne *zmniejszają* dane wejściowe.

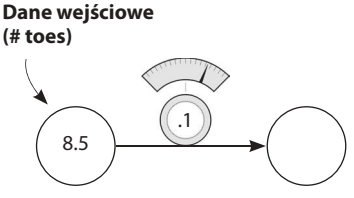
**❶ Pusta sieć**




```
weight = 0.1
def neural_network(input, weight):
    prediction = input * weight
    return prediction
```

Interfejs sieci neuronowej jest prosty. Akceptuje ona zmienną `input` jako *informację* oraz zmienną `weight` (waga) jako *wiedzę*, po czym zwraca `prediction` (prognozę). Każda sieć neuronowa, jaką kiedykolwiek spotkamy, będzie działać w ten sposób. Wykorzystuje *wiedzę* zawartą w wagach do interpretowania *informacji* w danych wejściowych. Później zobaczymy sieci neuronowe akceptujące większe, bardziej złożone wartości zmiennych `input` i `weight`, ale w tle zawsze będą te same przesłanki.

**❷ Wstawianie jednej obserwacji wejściowej**




```
number_of_toes = [8.5, 9.5, 10, 9]
input = number_of_toes[0]
pred = neural_network(input, weight)
print(pred)
```