

# Programowanie zespołowe

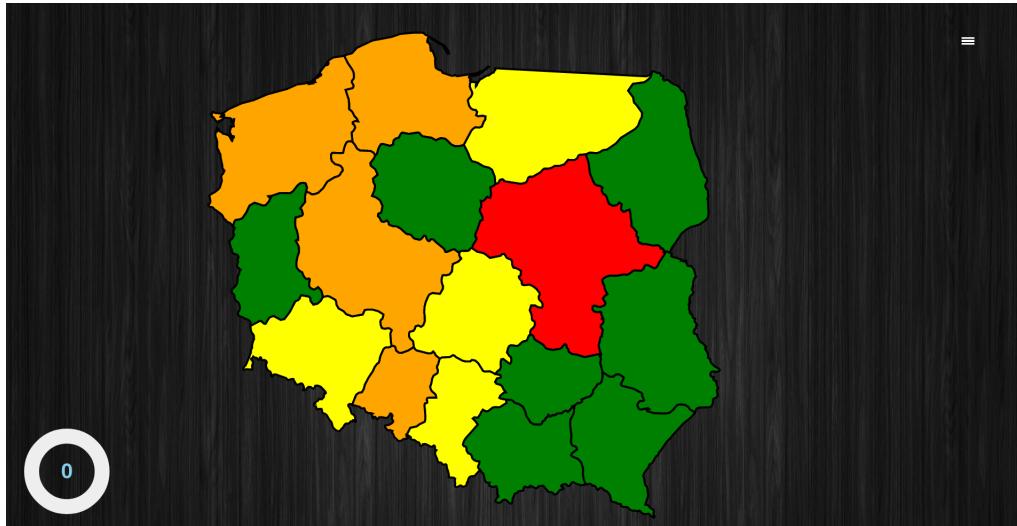
## Zadanie 3 - faza projektowania

Piotr Popis  
Mateusz Wałejko  
Adrian Majcher  
Jakub Kazimierski

November 30, 2020

## 1 Projekt interfejsu użytkownika

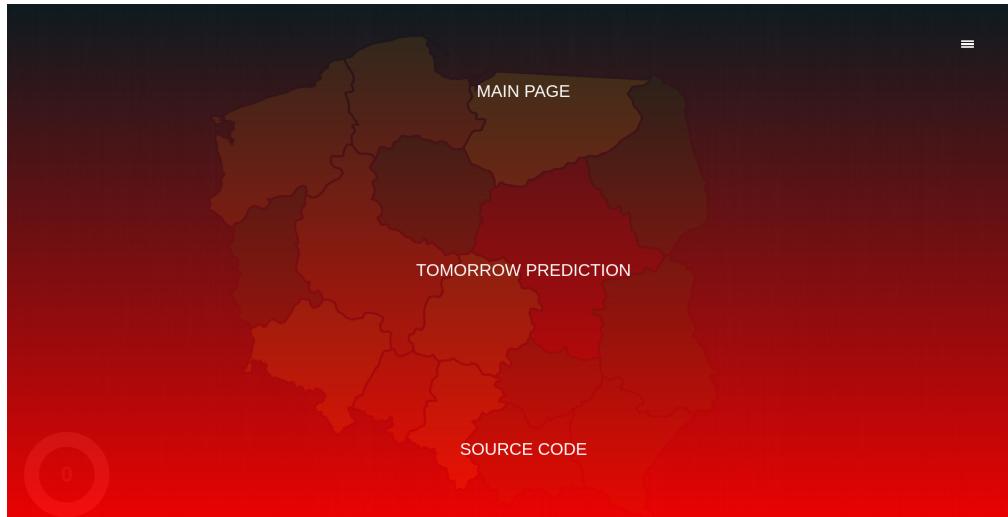
Użytkownik wchodząc na stronę główną wyświetla mapę pokolorowaną zgodnie z dzisiejszymi przypadkami tak jak na poniższym zrzucie.



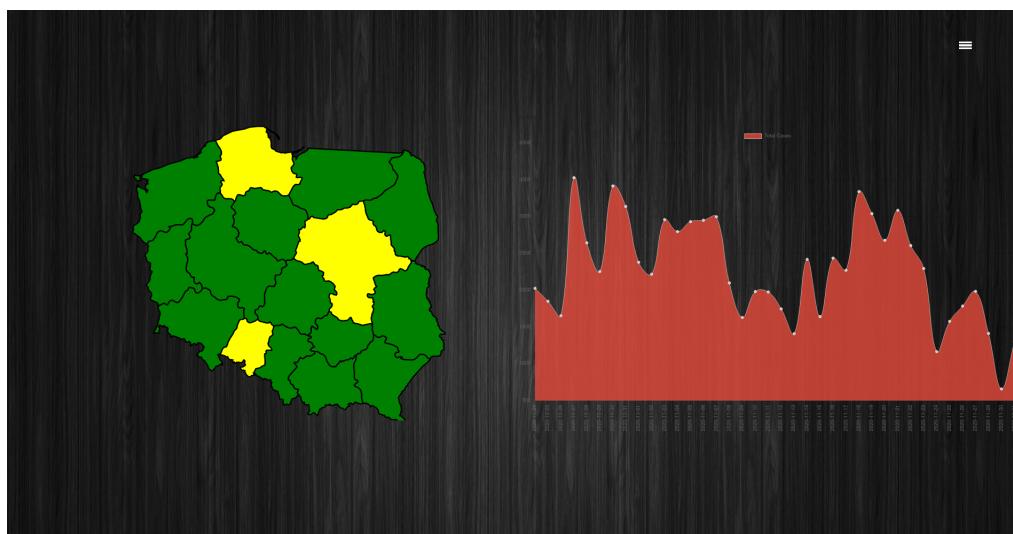
W lewym dolnym rogu ekranu znajduje się knob, służy do przestuwania się w czasie (wstecz). Mapa dynamicznie wyświetla dane na dzień, który wskazuje właśnie knob.



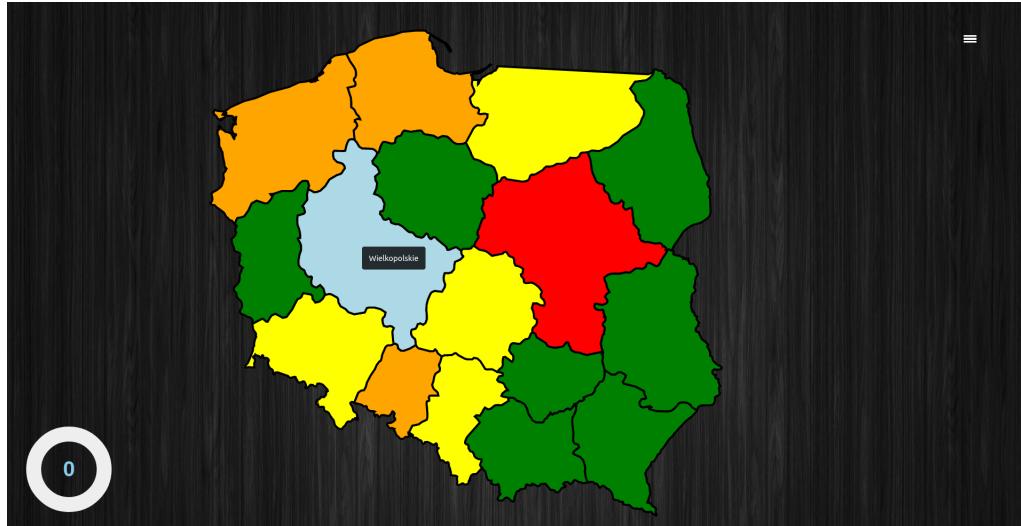
W prawym górnym rogu znajduje się natomiast przycisk wyświetlający menu.



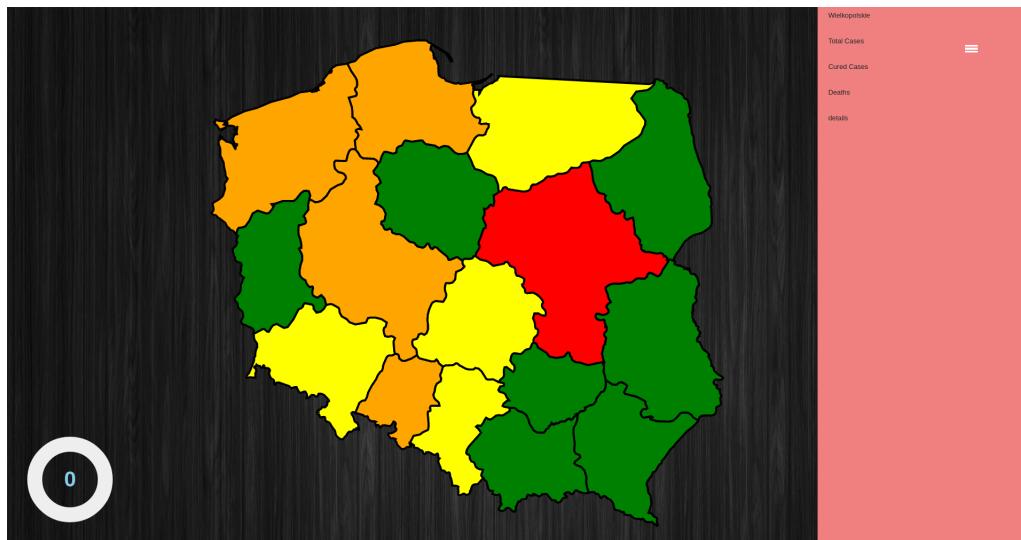
Korzystając z menu możemy przejść do sekcji predykcji. Która wyświetla wykres dla wskazanego województwa z poprzednich dni( + dzień predykcji ( z założenia jutrzeszy) do ilości przypadków zakażeń ( daily) oraz mapę pokolorowaną zgodnie z predykcjami na kolejny dzień. Wykres dynamycznie zmienia się po kliknięciu w dane województwo.



Każde z województw jest responsywne reaguje na akcje typu hover(Województwo zostaje podświetlone na odpowiedni kolor.( tutaj blue light)): oraz click( z prawej strony wysuwa się różowy panel z danymi ogólnymi dla województwa



i referencją do strony statystyk.) Na stronie statystyk znajdująć się będą



znajdować się wykresy, diagramy oraz wskaźniki.

## 2 Baza danych

Baza została zaprojektowana w sposób sprzyjający działaniu naszej aplikacji. Mianowicie zapis danych jest wykonywany okresowo raz dziennie, więc może

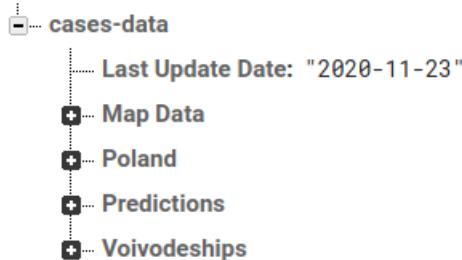
być bardziej kosztowny w stosunku do funkcji pobierania danych (load).

Z tego powodu naszą bazę podzieliśmy na cztery główne dokumenty i jedno wyszczególnione pole.

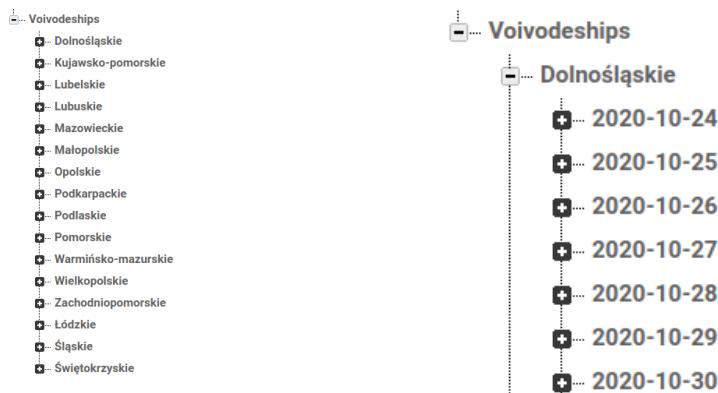
1. **Last Update Date** dla przyspieszenia pracy aplikacji zawiera pole szybkiego dostępu do daty.
2. **Map Data** zawiera okrojone dane potrzebne dla mapy.
3. **Predictions** zawiera dane predykowane na dzień kolejny do dnia zgodnego z polem **Last Update Date**
4. **Poland** zawiera dane dla całej Polski.
5. **Voivodeships** zawiera dane dla wszystkich województw z osobna.

Strukturę całej bazy przedstawiamy poniżej.

### covid-spread-analyzer



Struktura dokumentu wewnętrznego Voivodeships.



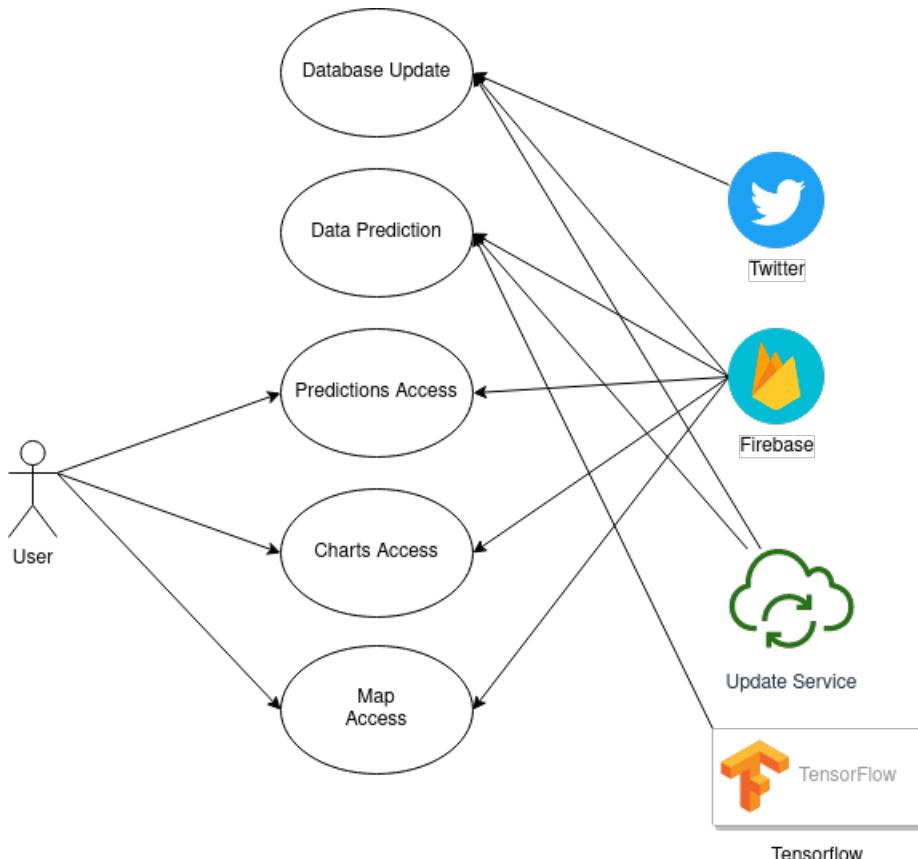
Przykładowy schemat (uproszczony) bazy danych w formacie json.

```
1 {  
2     "Last Update Date": "2020-11-26",  
3     "Map Data": {  
4         "2020-11-26": {  
5             "date": "2020-11-26",  
6             "daily_infected": 0,  
7             "daily_cured": 0,  
8             "daily_deceased": 0,  
9             "Voivodeships": {  
10                 "dolnoslaskie": {  
11                     "daily_infected": 0,  
12                     "daily_cured": 0,  
13                     "daily_deceased": 0  
14                 },  
15             },  
16         },  
17     },  
18 },  
19     "Poland": {  
20 }
```

```

21         "2020-11-26": {
22             "date": "2020-11-26",
23             "total infected": 0,
24             "total cured": 0,
25             "total deceased": 0,
26             "infected now": 0,
27             "daily infected": 0,
28             "daily cured": 0,
29             "daily deceased": 0,
30             "occupied respirators": 0,
31             "free respirators": 0,
32             "total tests": 0,
33             "daily tests": 0,
34             "test positivity ratio": 0,
35             "infected now per 100k": 0,
36             "total infected delta": 0,
37             "total infected per 100k": 0,
38             "daily infected per 100k": 0,
39             "daily infected per 100k week avg": 0,
40             "daily deceased per 100k infected": 0,
41         },
42     },
43 },
44 "Voivodeships": {
45     "Dolnoslaskie": {
46         "2020-11-26": {
47             "date": "2020-11-26",
48             "total infected": 0,
49             "total cured": 0,
50             "total deceased": 0,
51             "infected now": 0,
52             "daily infected": 0,
53             "daily cured": 0,
54             "daily deceased": 0,
55             "occupied respirators": 0,
56             "free respirators": 0,
57             "infected now per 100k": 0,
58             "total infected delta": 0,
59             "total infected per 100k": 0,
60                 "daily infected per 100k": 0,
61                 "daily infected per 100k week avg": 0,
62             "daily deceased per 100k infected": 0
63         },
64     },
65 },
66 },
67 }
68 }
```

### 3 Przypadki użycia



Standardowe przypadki użycia zobrazowane są na powyższym diagramie. Szczegółowo opiszemy najciekawsze z nich.

**Przypadek użycia:** Wyświetlanie mapy dla wybranego dnia.

**Warunki początkowe:** Użytkownik ma dostęp do internetu oraz serwer działa.

**Warunki końcowe:** Użytkownik widzi wizualizację danych na mapie dla wybranego dnia.

**Przebieg:**

1. Użytkownik wchodzi na stronę domową serwisu.
2. Serwis wyświetla stronę domową, użytkownik widzi responsywną mapę reprezentującą dane z obecnego dnia.

3. Użytkownik za pomocą knopa wybiera dzień, który ma być wizualizowany na mapie.
4. Serwis prezentuje mapę zgodnie z danymi z wybranego dnia.

**Przypadek użycia:** Wyświetlanie szczegółowych statystyk danego obszaru.

**Warunki początkowe:** Użytkownik ma dostęp do internetu oraz serwer działa.

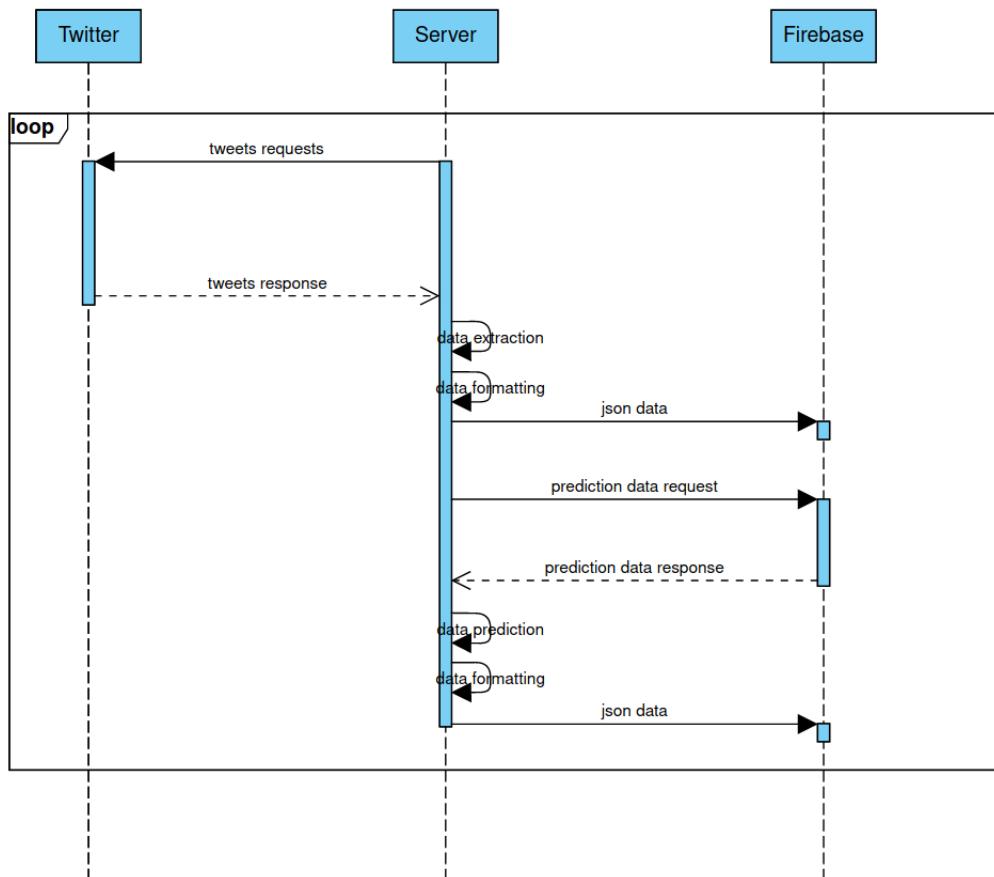
**Warunki końcowe:** Użytkownik widzi cały zespół statystyk (diagramy, wykresy, wskaźniki) dla wybranego obszaru.

**Przebieg:**

1. Użytkownik wchodzi na stronę domową serwisu.
2. Serwis wyświetla stronę domową, użytkownik widzi responsywną mapę
3. Użytkownik kliką w wybrany obszar.
4. Serwis wyświetla specjalny panel z ogólnymi informacjami dla wybranego obszaru i referencją do strony z danymi szczegółowymi.
5. Użytkownik wybiera opcję przejścia do strony ze statystykami.
6. Serwis przekierowuje użytkownika do strony statystyk.

## 4 Diagramy sekwencji

### 4.1 Aktualizacja bazy danych



Aplikacja automatycznie dokonuje okresowej aktualizacji bazy danych (raz dziennie, zgodnie z harmonogramem aktualizacji danych przez ministerstwo zdrowia).

Proces aktualizacji rozpoczyna się od wysłania żądania przysłania danych za pośrednictwem api Twittera (dokładniej wrapla Tweepy). Twitter w odpowiedzi zwraca dane (tweety) od określonego dnia. Po pobraniu danych, po stronie serwera następuje obróbka otrzymanych danych.

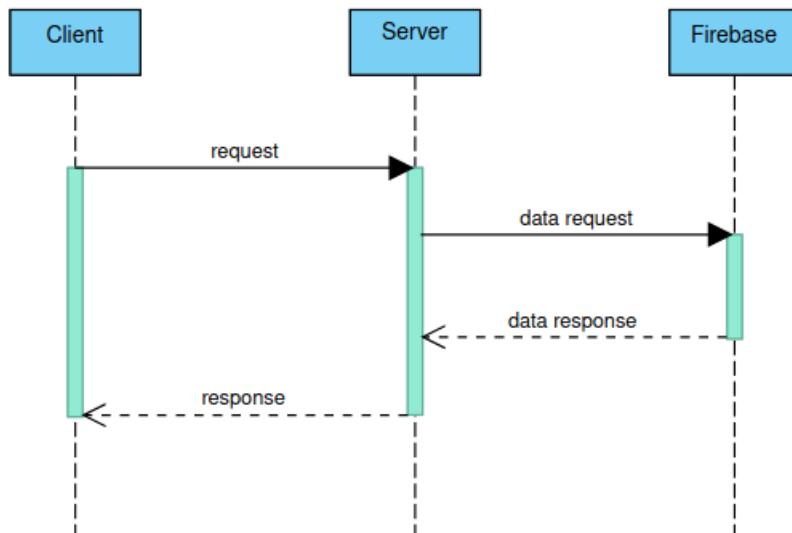
1. W pierwszej kolejności wyłuskiwane są dane z konkretnych tweetów.

2. Dane są parsowane do określonego formatu JSON.
3. Następuje wysyłanie obiektu json do bazy danych korzystając z specjalnego algorytmu uzupełniającego.

Następnym etapem aktualizacji jest predykcja danych.

1. Serwer wysyła żądanie o dotychczasowo zgromadzone dane do firebase.
2. Firebase zwraca odpowiedź z danymi.
3. Serwer korzystając z istniejącego modelu sieci neuronowej wykonuje predykcję na przyszły dzień.
4. Następnie dane zostają sformatowane do odpowiedniej struktury.
5. Dane w formacie JSON (JavaScript Object Notation) są bezpośrednio wysyłane do bazy danych.

## 4.2 Użycie klienta



1. Klient wysyła żądanie do serwera.

2. Serwer wysyła żądanie do Firebase.
3. Firebase zwraca dane do serwera.
4. Serwer uzupełnia template danymi pobranymi z Firebase.
5. Wypełniony template jest wyświetlany jako response na żądanie Klienta.

## 5 Projekt sieci neuronowej

Naszym problemem z punktu widzenia sieci neuronowej jest rowiązanie problemu eksploracyjnego. Sieci neuronowe są znacznie lepsze w rozwiązywaniu problemów interpolacyjnych. W skrócie znaczy to tyle, że jeśli uczyliśmy sieć zdjęciami psów gatunku Owczarek Niemiecki to niekoniecznie sieć rozpozna Yorka jako psa, ale odpowiednie dobranie funkcji aktywacji, ilości warstw oraz ilości neuronów na każdej z nich pozwala jednak na bliskie, wystarczająco precyzyjne predykcje.

**Projekt struktury** polega na skonfigurowaniu powyższych czynników i dopasowaniu ich do naszych danych. Ze względu na to, że głównym językiem naszej aplikacji poza JavaScript'em jest Python wygodnie będzie nam skorzystać z biblioteki numpy. Nasze dane z przeszłości wygodnie jest ustawić w liście [1, 23, 3, 22, ..., 33, 3]. Taki *ksztat* listy pozwala nam na łatwe przekształcenie jej do postaci:

[1],

[23],

[3],

[22],

.

[33],

[3]]

Jak widać taka postać wymaga od nas ustawienia wymiaru 1 na warstwie inputu, a że chcemy wynik uzyskać w bardzo podobnej formie, wymiar output będzie taki sam. W tym miejscu możemy powiedzieć, że analiza kształtu,

formatu naszych danych jednoznacznie doprowadziła do modelu o kształcie  $1 - x * -1$ .

**Warstwy wewnętrzne** jeśli nasz zbiór danych tutaj byłby liniowo rozłączny (to znaczy, że pewne podgrupy wartości y można oddzielić pewną prostą) to można owe zupełnie pominąć. Niestety my takiego założenia postawić nie możemy. Co dalej? W dodatku nie ma żadnej reguły dotyczącej architektury sieci neuronowej. Nie ma też obecnie żadnego teoretycznego powodu, aby używać sieci neuronowych z więcej niż dwoma ukrytymi warstwami. W rzeczywistości w przypadku wielu praktycznych problemów nie ma powodu, aby używać więcej niż jednej warstwy ukrytej.

**$N_h$  rule** Istnieje natomiast jedna praktyczna zasada, która pozwala określić całkowitą liczbę neuronów. Jest to jednak bardzo ogólna formuła i trzeba brać na nią odpowiednią poprawkę.

$$N_h = \frac{N_s}{\alpha \cdot (N_i + N_o)}$$

$N_i$  = number of input neurons.

$N_o$  = number of output neurons.

$N_s$  = number of samples in training data set.

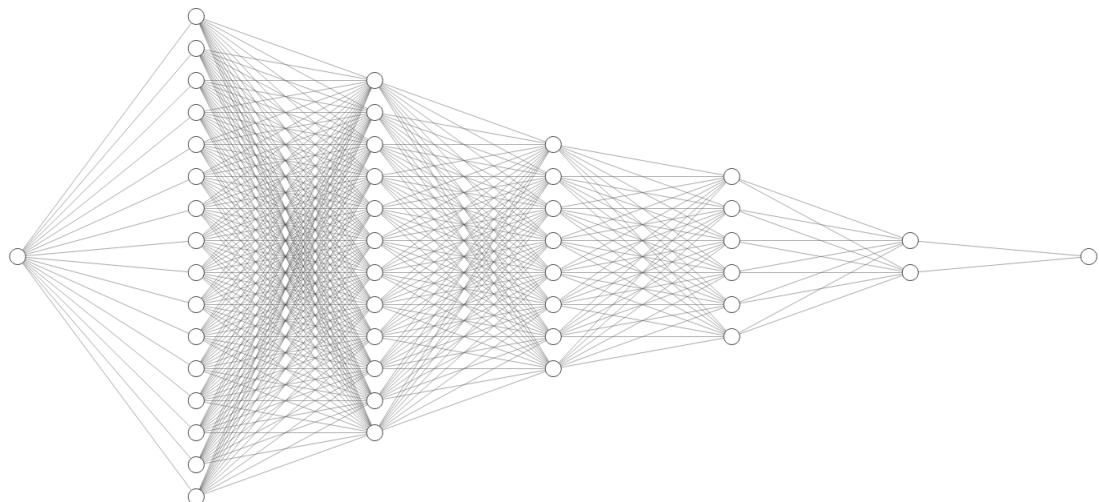
$\alpha$  = an arbitrary scaling factor usually 2-10. My dysponujemy około 120 dniami danych. Możemy zatem obliczyć, że nasza sieć powinna mieć w sumie około 50 neuronów.

**Aktywacje** musielibyśmy dobrać odpowiednią funkcję aktywacji. Tutaj musielibyśmy przeprowadzić analizę naszych danych. O ile dla tylko jednego województwa jesteśmy zauważycie pewne anomalie o tyle wyznaczyć wspólną dla wszystkich województw dla tak niewielkiej ilości próbek danych jest bardzo cięzkie. Wiemy, że wartości naszego outputu są na pewno większe niż 1 i możemy założyć, że są to wartości z przedziału  $(0, 10^6)$  zatem odpowiednimi funkcjami na wyjściu i wejściu są funkcje softmax. Pozostałe funkcje trzeba dobrać tak, aby pasowały do tak niestandardowych danych. Możemy jednak założyć, że latem jest mniejsza ilość przypadków. Aby dopasować odpowiednie funkcje aktywacji, ilości neuronów i samą ilość warstw wewnętrznych skorzystaliśmy z algorytmu metaheurystycznego, mianowicie simulated annealing, który dobrał te współczynniki. Poniżej znajduje się wizualizacja predykcji dla tak zaprojektowanej sieci.

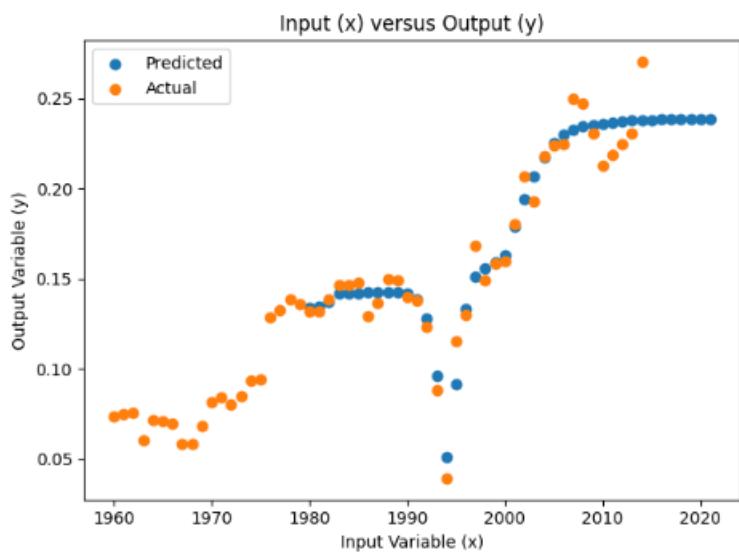
**Podsumowanie** Ostatecznie nasza sieć wygląda w następujący sposób:

1. Składa się z 7 warstw
2. Suma całkowita neuronów to 52.
3. Funkcje aktywacji na wyjściu i wejściu to *softmax*.
4. Funkcje warstw wewnętrznych zostały dobrane przy użyciu SA.  
[*tanh, tanh, relu, tanh, tanh, tanh*]

Przedstawiamy strukturę sieci do przewidywania ilości dziennych przypadków zakażeń.



Przykładowe predykcja dla danych o dość podobnym rozłożeniu do tych na których będziemy pracować.



Jak widać nasz model świetnie sobie radzi i wyniki są bardzo dobre.