Instituto Politécnico Nacional

Escuela Superior de Cómputo

Evolutionary Computing

Lab Session 8: Self Organizing Map (SOM)

Alumno: David Arturo Oaxaca Pérez

Grupo: 3CV11

Profesor: Jorge Luis Rosas Trigueros

Fecha de realización: 21/10/2021

Fecha de entrega: 22/10/2021

Marco teórico

Los mapas auto-organizados (SOM por sus siglas en inglés) son un tipo de red neuronal artificial introducido por Teuvo Kohonen en 1980, estos son entrenados usando aprendizaje sin supervisión. Esta forma de aprendizaje es un algoritmo de inteligencia artificial identifica patrones en los *data sets* que contienen puntos de datos los cuales no están clasificados ni etiquetados, es decir, permite a un algoritmo identificar patrones de un *data set* por sí mismo.

El uso de un SOM puede ayudar a reducir las dimensiones de una entrada de datos, por ejemplo, en algunos casos como puede ser la aplicación de un SOM en la astronomía se pueden tener cantidades gigantescas de columnas y dimensiones que pueden ser reducidas en un mapa mucho más simplificado, que representa el mismo *data set* pero de una manera más sencilla de interpretar (SuperDataScience Team, 2018).

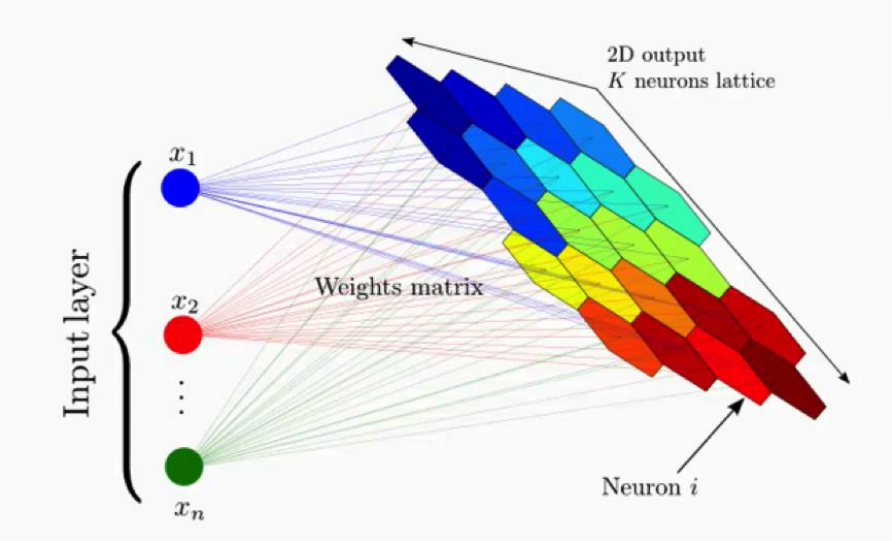


Figura 1. Representación de un SOM con un *set* de datos de múltiples columnas como input. Cortesía: Super data science

En muchos casos como en la comparación de países usando indicadores de bienestar para alinearlos dentro de un SOM, estos son representados dentro del mapa en grupos que representan un espectro de dichos indicadores.

El entrenamiento de un SOM no emplea un tipo de propagación hacia atrás para actualizar los vectores de pesos, sino que usa un aprendizaje competitivo, este esta basado en tres procesos:

- **Competencia:** A cada neurona en un SOM se le asigna un vector de peso con la misma dimensionalidad que la entrada, se calcula la distancia entre cada neurona y los datos de entrada, la neurona que tenga la distancia (usualmente usando la fórmula de distancia euclidiana) más cercana será el "ganador" de la competencia.

- **Corporación:** Se actualizará el vector de la neurona ganadora en el proceso final, pero también se actualizarán los vectores de sus vecinos, los cuales se escogerán por su distancia a la neurona ganadora y el tiempo transcurrido.
- **Adaptación:** Después de escoger a la neurona ganadora y a sus vecinos, estos actualizados según un ajuste basado en la distancia entre la neurona y los datos de entrada (Khazri, 2021).

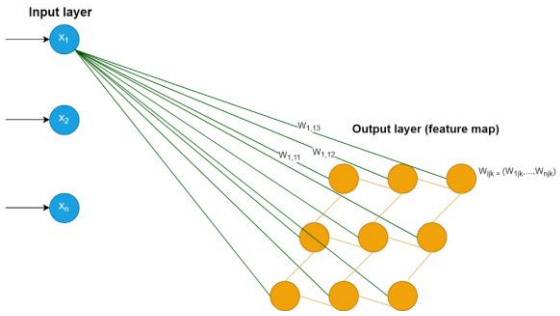


Figura 2. Representación de la competencia en el entrenamiento de neuronas de un SOM. Cortesía: towards data science

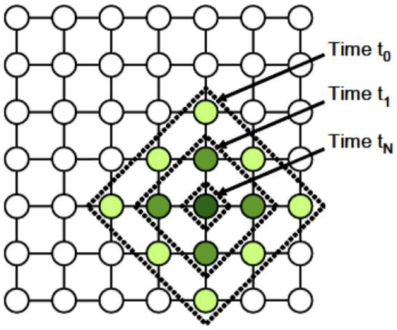


Figura 3. Representación de la corporación en el entrenamiento de neuronas de un SOM en la neurona ganadora y sus vecinos. Cortesía: towards data science

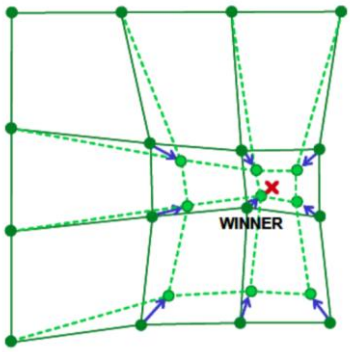


Figura 4. Representación de la adaptación en el entrenamiento de neuronas de un SOM. Cortesía: towards data science

El algoritmo para el entrenamiento de un SOM visto en clase es el siguiente:

1. Aleatorizar los vectores de pesos de los nodos del mapa, s = 0
2. Atravesar cada vector dado como entrada en el set de datos de entrada D(t)
    a. Atravesar cada nodo en el mapa.
    b. Usar la fórmula de la distancia euclidiana para encontrar la similaridad entre el vector de entrada y los vectores de pesos de los nodos del mapa.
    c. Registrar el nodo que produce la distancia más corta (este nodo, en la posición u será la unidad que mejor coincide o BMU por sus siglas en inglés)
    d. Actualizar los nodos cercanos a la BMU incluyendo la BMU acercándolos al vector de entrada por cada vecino en la posición v: W(s+1) = W(s) + T(u,v,s)*a(s)*(D(t)-W(s))
3. Incrementar s y repetir desde el paso dos hasta que el criterio para detenerse sea alcanzado.

En este algoritmo:

- u es la posición de la unidad que mejor coincide (BMU).
- v es la posición de un vecino de la BMU.
- W(s) es el vector de peso en la posición v.
- T(u, v, s) es la función que controla el efecto D(t) sobre W en v.
- a(s) es la función que usualmente se desvanece conforme s se incrementa.

Material y equipo

- El programa fue realizado en Google Colaboratory usando Python 3.9 como lenguaje.
- El sistema que se utilizó para acceder a Google Colaboratory fue una laptop de marca Lenovo, modelo Ideapad S540 con Ryzen 7 y 8 GB de RAM, con sistema operativo Windows 10.

Desarrollo de la practica

Para el desarrollo de esta práctica lo primero que se tuvo que hacer fue implementar las modificaciones vistas en clase para que se pudiese trabajar con valores mayores a 1, previamente se necesitaba normalizar el vector que era enviado para entrenar a la red neuronal, dividiendo todos los elementos de este entre el más grande. La modificación necesaria en este caso fue cambiar la distancia mínima obtenida en la función para la best matching unit (BMU), se estaba utilizando la raíz de esta y se reemplazo por la forma en que normalmente se calcularía la distancia.

```
#Returns best matching unit's index
def best_match(self, target_FV=[0.0]):

    minimum=self.FV_distance(self.nodes[0].FV, target_FV) #Minimum distance
    minimum_index=1 #Minimum distance unit
    temp=0.0
    for i in range(self.total):
        temp=0.0
        temp=self.FV_distance(self.nodes[i].FV,target_FV)
        if temp<minimum:
            minimum=temp
            minimum_index=i
```

Figura 5. Cambio realizado en el cálculo de la distancia mínima para la función best_match()

Después de ese cambio, en la clase SOM, que es la que nos apoyara en la creación de un mapa auto-organizado, lo que se necesitó hacer fue conseguir 10 indicadores de bienestar para 40 países, los indicadores seleccionados fueron los siguientes:

- Acceso a la electricidad
- Ingreso nacional neto ajustado
- Ingreso nacional neto ajustado per cápita
- Proporción de dependencia de edad
- Tasa de nacimientos
- Importación de materiales agriculturales
- Consumo de energías renovables
- Expectativas de vida desde el nacimiento
- Producción de electricidad a partir de petróleo, gas y carbón
- Proporción de fertilidad total (Nacimientos por mujer)

Y los países utilizados para el *data set* fueron:

- Argentina (ARG)
- Albania (ALB)
- Austria (AUT)
- Bangladesh (BGD)
- Belgium (BEL)
- Brazil (BRA)
- Bolivia (BOL)
- Canada (CAN)
- Chile (CHL)
- Colombia (COL)
- China (CHN)
- Costa Rica (CRI)
- Dinamarca (DNK)
- Egipto (EGY)
- Ecuador (ECU)
- El Salvador (SLV)
- Finlandia (FIN)
- Francia (FRA)
- Alemania (DEU)
- Hungría (HUN)
- India (IND)
- Italia (ITA)
- Jamaica (JAM)
- Japón (JPN)
- México (MEX)
- Nueva Zelanda (NZL)
- Noruega (NOR)
- Pakistán (PAK)
- Polonia (POL)
- Perú (PER)
- Paraguay (PRY)
- Filipinas (PHL)
- Portugal (PRT)
- Romania (ROU)
- Sudáfrica (ZAF)
- Arabia Saudita (SAU)
- España (ESP)
- Suecia (SWE)
- Estados Unidos (USA)
- Turquía (TUR)

Estos se escribieron y se acomodaron en el archivo de texto “countries.txt” donde el primer dato en cada línea es la abreviación del país a donde corresponden los datos y los siguientes 10 son dichos indicadores, todos separados por comas. En el programa lo que se hará será separar a estos en dos listas, una que contendrá todos los países y otra que contendrá los indicadores de estos.

```
f = open('countries.txt','r')
cont = 0
for line in f:
    muestra=[]
    line_split = line.split(',')
    countries.append(line_split[0])
    line_split = line_split[1:]

    for x in line_split:
        muestra.append(float(x))

    training_set.append([muestra,[cont]])
    cont+=1

f.close()
```

Figura 6. Lectura del archivo para la obtención de los países e indicadores con los que se trabajara.

Posteriormente se procederá a entrenar el SOM usando las funciones de la clase con el mismo nombre y se incorporaran los países al mapa usando las coordenadas obtenidas de la mejor predicción.

```
print( "Training for the countries function..." )
a.train(2850,training_set)

print( "\nPredictions for some countries..." )

for i in range(40):
  prediccion = a.predict(test_set[i][0], True)
  coordinates.append([prediccion[1],prediccion[2],countries[i]])
  #print(coordinates[i])
  print( f"Prediction para el pais de indice {i} ({countries[i]}):" \
  f"{round(prediccion[0][0],7)} | {round(prediccion[0][0])} | {prediccion[1], prediccion[2]}")

print( "\n")
map2D = []
for y in range(16):
    row=[]
    for x in range(16):
        row.append(round(a.nodes[(x)*a.width+y].PV[0]))
    map2D.append(row)
```

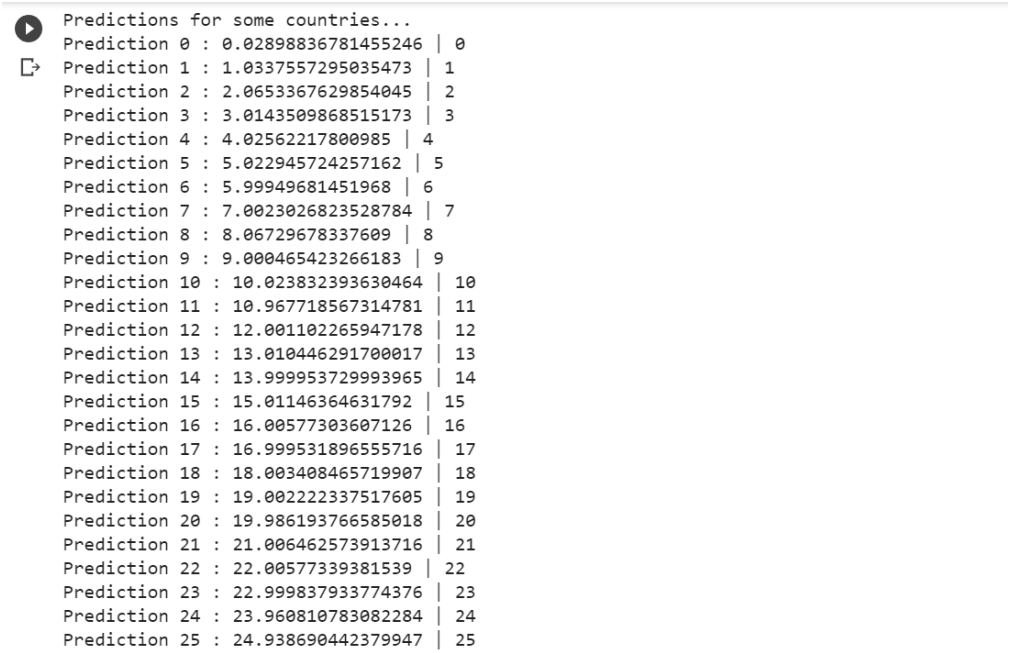
Figura 7. Partes del código donde se inicia el entrenamiento del SOM, se realizan las predicciones y se incorporan los países al mapa según la mejor predicción.

Eso de manera general, describe el proceso realizado para obtener un mapa auto organizado usando los indicadores de bienestar de los países seleccionados, pero para lograr obtener un mapa con el que pudiéramos obtener un nivel contante de predicciones acertadas, se necesitó hacer uso de múltiples pruebas que requerían

tanto de diferentes dimensiones, como números de iteraciones en el entrenamiento y diferentes proporciones de aprendizaje. Primero se probó con una altura y anchura más limitada, de 10x10, esto requería en muchas ocasiones una mayor cantidad de iteraciones de entrenamiento y en muchas ocasiones se llegaba a predicciones erróneas en los países que estaban en los extremos, como los primeros y los últimos de la lista que parecían estar más lejos de los grupos, por ejemplo, el país que se encontraba en el índice 0, muy rara vez se encontraba una predicción cercana a este y de la misma manera no se encontraban muchos nodos que coincidieran en el mapa.

Después de eso se experimentó con mapas más grandes para que hubiera mayor posibilidad de exploración a la hora de realizar una predicción, por ejemplo, se probó con un mapa de 20x20 en el que efectivamente se obtenían predicciones muy efectivas con muchas menos iteraciones, aun eran necesarias cerca de 2000 iteraciones pero los resultados obtenidos eran bastantes favorables, aunque el entrenamiento era mucho más tardado, de esta manera se continuo probando con mapas de varias dimensiones hasta encontrar uno que no fuera tan grande (Y por lo tanto más tardado de entrenar) pero que tampoco necesitara tantas iteraciones.

El siguiente mapa de 15x15 requirió 3000 iteraciones para su entrenamiento, lo cual no fue tan tardado y obtuvo buenos resultados.



```
Predictions for some countries...
Prediction 0 : 0.02898836781455246 | 0
Prediction 1 : 1.0337557295035473 | 1
Prediction 2 : 2.0653367629854045 | 2
Prediction 3 : 3.0143509868515173 | 3
Prediction 4 : 4.02562217800985 | 4
Prediction 5 : 5.022945724257162 | 5
Prediction 6 : 5.99949681451968 | 6
Prediction 7 : 7.0023026823528784 | 7
Prediction 8 : 8.06729678337609 | 8
Prediction 9 : 9.000465423266183 | 9
Prediction 10 : 10.023832393630464 | 10
Prediction 11 : 10.967718567314781 | 11
Prediction 12 : 12.001102265947178 | 12
Prediction 13 : 13.010446291700017 | 13
Prediction 14 : 13.999953729993965 | 14
Prediction 15 : 15.01146364631792 | 15
Prediction 16 : 16.00577303607126 | 16
Prediction 17 : 16.999531896555716 | 17
Prediction 18 : 18.003408465719907 | 18
Prediction 19 : 19.002222337517605 | 19
Prediction 20 : 19.986193766585018 | 20
Prediction 21 : 21.006462573913716 | 21
Prediction 22 : 22.00577339381539 | 22
Prediction 23 : 22.999837933774376 | 23
Prediction 24 : 23.960810783082284 | 24
Prediction 25 : 24.938690442379947 | 25
```

Figura 8. Primeras predicciones del SOM con dimensiones de 15x15.

```
Prediction 26 : 26.071348739536617 | 26
Prediction 27 : 27.000003746797294 | 27
Prediction 28 : 27.992760017895865 | 28
Prediction 29 : 28.98825939638798 | 29
Prediction 30 : 29.997625549391802 | 30
Prediction 31 : 30.98066325523527 | 31
Prediction 32 : 31.93449390317608 | 32
Prediction 33 : 32.99929304166288 | 33
Prediction 34 : 33.98026227587431 | 34
Prediction 35 : 34.99997724451897 | 35
Prediction 36 : 35.97430295003089 | 36
Prediction 37 : 36.90679814734224 | 37
Prediction 38 : 37.99640470905859 | 38
Prediction 39 : 38.93451552819177 | 39
```

Figura 9. Complementación de predicciones del SOM con dimensiones de 15x15.

```
[3, 19, 'IND', 20, 27, 'PAK', 27, 6, 'BOL', 6, 16, 16, 16, 30, 'PRY']
['BGD', 4, 20, 20, 27, 27, 18, 9, 6, 6, 16, 'FIN', 16, 28, 28]
[3, 17, 31, 31, 29, 29, 10, 'COL', 9, 2, 3, 16, 36, 27, 'NOR']
[34, 31, 'PHL', 31, 29, 'PER', 28, 9, 12, 3, 'AUT', 20, 'SWE', 36, 19]
['ZAF', 34, 29, 7, 17, 28, 16, 12, 'DNK', 12, 14, 21, 37, 13, 'CRI']
[34, 33, 5, 'BRA', 5, 15, 'SLV', 15, 12, 24, 'NZL', 25, 2, 2, 10]
[13, 13, 15, 15, 15, 15, 15, 15, 22, 25, 24, 8, 1, 'ALB', 1]
['EGY', 13, 24, 'MEX', 24, 23, 23, 33, 33, 32, 7, 'CAN', 7, 2, 16]
[13, 18, 22, 24, 23, 'JPN', 23, 33, 'ROU', 33, 8, 7, 7, 17, 'FRA']
[28, 22, 'JAM', 22, 23, 22, 1, 17, 33, 26, 19, 19, 14, 17, 17]
['POL', 28, 22, 31, 39, 2, 'ARG', 1, 18, 19, 'HUN', 19, 27, 4, 4]
[28, 19, 10, 37, 'TUR', 37, 0, 17, 'DEU', 18, 19, 28, 36, 6, 'BEL']
[24, 10, 'CHN', 12, 39, 39, 21, 21, 18, 18, 21, 36, 'ESP', 34, 4]
[35, 23, 10, 10, 38, 38, 22, 'ITA', 20, 8, 20, 32, 36, 25, 14]
['SAU', 35, 23, 36, 38, 'USA', 37, 21, 9, 'CHL', 20, 'PRT', 32, 14, 'ECU']
```

Figura 10. SOM obtenido colocando los países en los lugares de la predicción que mejor coincidio.

Después de seguir probando con distintos parámetros obtuvimos los mejores resultados con un mapa de 16x16, 2800 iteraciones y una proporción de aprendizaje de 0.07, se acomodaron los formatos de impresión y se incorporaron otras adiciones en el output para hacer de este visualmente más entendible.

```
Predictions for some countries...
Prediction para el pais de indice 0 (ARG):0.1868039 | 0 | (4, 9)
Prediction para el pais de indice 1 (ALB):1.0271444 | 1 | (12, 0)
Prediction para el pais de indice 2 (AUT):2.2967313 | 2 | (9, 4)
Prediction para el pais de indice 3 (BGD):3.0001143 | 3 | (15, 15)
Prediction para el pais de indice 4 (BEL):4.0104967 | 4 | (0, 3)
Prediction para el pais de indice 5 (BRA):4.9910162 | 5 | (2, 9)
Prediction para el pais de indice 6 (BOL):6.0011324 | 6 | (15, 6)
Prediction para el pais de indice 7 (CAN):7.0217763 | 7 | (8, 2)
Prediction para el pais de indice 8 (CHL):8.0029422 | 8 | (9, 10)
Prediction para el pais de indice 9 (COL):8.9998463 | 9 | (12, 7)
Prediction para el pais de indice 10 (CHN):10.0200202 | 10 | (8, 12)
Prediction para el pais de indice 11 (CRI):10.9870887 | 11 | (9, 0)
Prediction para el pais de indice 12 (DNK):11.992834 | 12 | (9, 6)
Prediction para el pais de indice 13 (EGY):13.0033304 | 13 | (0, 13)
Prediction para el pais de indice 14 (ECU):14.0018883 | 14 | (1, 0)
Prediction para el pais de indice 15 (SLV):14.9991063 | 15 | (3, 1)
Prediction para el pais de indice 16 (FIN):15.9940253 | 16 | (12, 4)
Prediction para el pais de indice 17 (FRA):16.9987109 | 17 | (6, 0)
Prediction para el pais de indice 18 (DEU):18.0265325 | 18 | (2, 6)
Prediction para el pais de indice 19 (HUN):19.0385107 | 19 | (5, 3)
Prediction para el pais de indice 20 (IND):19.999512 | 20 | (13, 13)
```

Figura 11. Primeras predicciones del SOM con dimensiones de 16x16.

```
Prediction para el pais de indice 21 (ITA):20.9980421 | 21 | (0, 7)
Prediction para el pais de indice 22 (JAM):22.0189771 | 22 | (4, 14)
Prediction para el pais de indice 23 (JPN):22.9610271 | 23 | (0, 10)
Prediction para el pais de indice 24 (MEX):23.9956286 | 24 | (3, 12)
Prediction para el pais de indice 25 (NZL):24.6979001 | 25 | (10, 2)
Prediction para el pais de indice 26 (NOR):26.0431431 | 26 | (15, 3)
Prediction para el pais de indice 27 (PAK):27.0000177 | 27 | (15, 10)
Prediction para el pais de indice 28 (POL):27.9697939 | 28 | (6, 13)
Prediction para el pais de indice 29 (PER):28.9995303 | 29 | (12, 10)
Prediction para el pais de indice 30 (PRY):29.9974682 | 30 | (15, 0)
Prediction para el pais de indice 31 (PHL):30.9985476 | 31 | (11, 15)
Prediction para el pais de indice 32 (PRT):31.9962512 | 32 | (7, 8)
Prediction para el pais de indice 33 (ROU):32.9813699 | 33 | (6, 6)
Prediction para el pais de indice 34 (ZAF):33.9774895 | 34 | (2, 15)
Prediction para el pais de indice 35 (SAU):34.9998464 | 35 | (8, 15)
Prediction para el pais de indice 36 (ESP):35.959867 | 36 | (3, 4)
Prediction para el pais de indice 37 (SWE):36.9246668 | 37 | (13, 2)
Prediction para el pais de indice 38 (USA):37.8089568 | 38 | (4, 7)
Prediction para el pais de indice 39 (TUR):38.9616653 | 39 | (6, 10)
```

Figura 12. Complementación de predicciones del SOM con dimensiones de 16x16.

14	14	4	BEL	4	13	21	ITA	21	23	JPN	23	13	EGY	13	24
ECU	14	4	4	5	18	18	21	13	6	22	23	13	13	24	34
14	15	15	35	36	19	DEU	18	5	BRA	5	23	24	24	34	ZAF
15	SLV	15	36	ESP	35	20	36	14	3	3	24	MEX	24	23	33
15	15	17	20	35	36	38	USA	19	ARG	0	24	24	22	JAM	22
17	17	19	HUN	19	33	33	38	19	3	36	39	28	28	22	22
FRA	17	19	19	19	33	ROU	33	32	38	TUR	39	28	POL	28	31
17	12	7	7	20	33	33	32	PRT	33	39	25	11	27	32	35
11	7	CAN	7	2	7	12	22	32	20	8	10	CHN	10	35	SAU
CRI	12	16	3	AUT	7	DNK	12	17	8	CHL	8	10	10	35	35
11	24	NZL	23	2	7	12	12	10	8	8	8	10	31	31	31
1	13	25	20	16	16	9	9	9	29	29	29	25	26	31	PHL
ALB	4	37	18	FIN	16	9	COL	9	29	PER	29	20	20	26	31
1	34	SWE	35	16	16	9	9	9	29	29	29	20	IND	20	18
30	34	36	27	26	7	6	6	14	27	27	27	20	20	12	3
PRY	30	26	NOR	26	6	BOL	6	17	27	PAK	27	26	12	3	BGD

Figura 8. SOM obtenido colocando los países en los lugares de la predicción que mejor coincidio con un formato de impresión más legible.

En este último mapa podemos ver como los países se encuentran ordenados en el SOM que muestra un espectro representativo de los indicadores de bienestar para cada país, por ejemplo, hacia la derecha en la parte baja podemos encontrar a los países donde aún hay cierto porcentaje de su población que no cuenta con electricidad en su hogar.

Conclusiones y recomendaciones

En esta práctica se aprendieron bastantes cosas pues en lo que es la creación de un SOM se requiere implementar una red neuronal y un algoritmo de aprendizaje sin supervisión, si bien gran parte de esto fue proporcionado en clase, fue muy interesante ver cómo es que era el algoritmo visto ya en código y como podíamos utilizar este para la creación de un SOM distinto al del ejemplo.

Lo que se fue realizando ayudo a explorar un poco más los conceptos trabajados en clase que son necesarios para la construcción de un mapa auto-organizado, como lo es el tamaño de este, su altura y anchura que pueden ayudar a que exista una mayor exploración a la hora de hacer una predicción y de igual manera en los *clusters* que se forman en el mapa, también la importancia de la cantidad de iteraciones que se requieren para su entrenamiento y como estas pueden cambiar según las características del data set que sea dado en la entrada.

Algunas de las recomendaciones que se pueden sugerir para esta práctica, es probar con distintas alturas y anchuras para los mapas como se fue viendo a lo largo del desarrollo, pues esto puede llevar a mejores resultados si es que se requiere una mayor exploración, además de que es una buena forma de explorar nuevas alternativas si uno se encuentra atorado con algunas predicciones que parecen fallar constantemente. Está en general fue una práctica que requirió hacer bastantes pruebas y ver los objetivos desde diferentes enfoques para poder llevarla a cabo con resultados que fueran satisfactorios, pues en muchos casos se necesitaban demasiadas iteraciones o mapas muy grandes lo que terminaba haciendo a los tiempos de entrenamiento muy largos, lo que llevo a probar con distintos enfoques en el desarrollo del SOM con países, algo que se volviera clave en esta sesión de laboratorio.

Bibliografía

SuperDataScience Team. (2018, 18 septiembre). *SuperDataScience*. Super Data Science. Recuperado 12 de noviembre de 2021, de https://www.superdatascience.com/blogs/self-organizing-maps-soms-how-do-self-organizing-maps-work

Khazri, A. (2021, 13 octubre). *Self Organizing Maps - Towards Data Science*. Medium. Recuperado 12 de noviembre de 2021, de https://towardsdatascience.com/self-organizing-maps-1b7d2a84e065