TP4: Aprendizaje No Supervisado

Grupo 6
Alberto Abancens (62581)
Kevin Catino (61643)
Agustín Galarza (61481)
Abril Occhipinti (61159)
Maiwenn Boizumault (65988)
Agustin Benvenuto (61448)

Contenidos

01

Ejercicio 1.1

02

Ejercicio 1.2

03

Ejercicio 2

01

Ejercicio 1

Ejercicio Europa

Variables:

- **Country:** Nombre del país
- **Area:** área
- **GDP:** producto bruto interno
- **Inflation:** inflación anual
- **Life.expect:** expectativa de vida media en años
- **Military:** presupuesto militar
- **Pop.growth:** tasa de crecimiento poblacional
- **Unemployment:** tasa de desempleo

Red de Kohonen

Problema a resolver:

- Asociar países que posean las mismas características geopolíticas, económicas y sociales.
- Realizar al menos un gráfico que muestre los resultados.
- Realizar un gráfico que muestre las distancias promedio entre neuronas vecinas.
- Analizar la cantidad de elementos que fueron asociados a cada neurona.

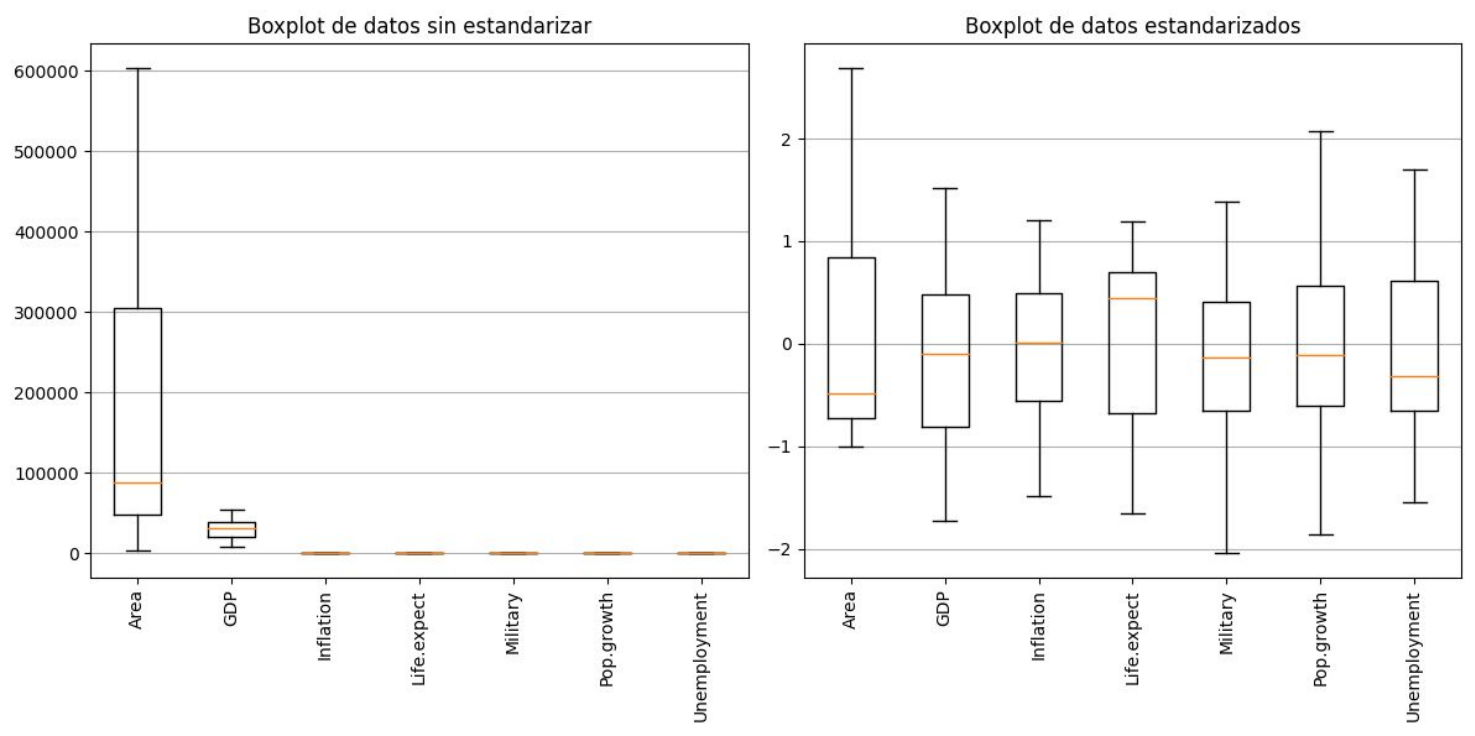
Red de Kohonen

- **Método de similitud:** Distancia Euclídea
- **Radio:**
  - Variable: R(i) = R(0)/i
  - Fijo: R(i) = R(0)
- **Estandarización:** Z-Score

$$\tilde{X}_i = \frac{X_i - \overline{X}_i}{s_i} \qquad s_i = \sqrt{\frac{1}{n-1}\sum_{j=1}^{n}(X_i^j - \overline{X}_i)}$$

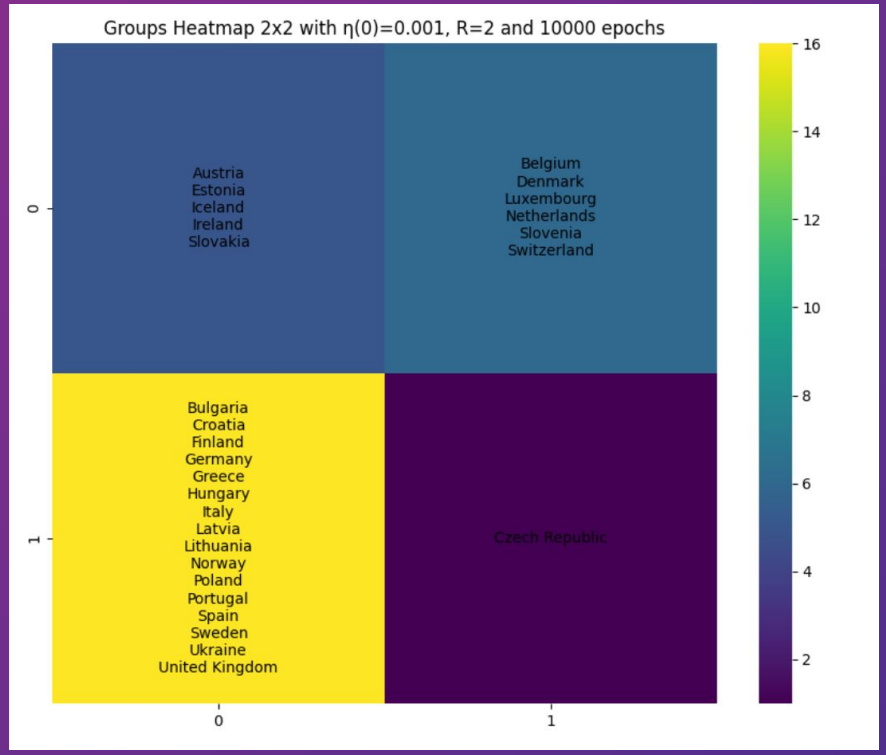
- **Inicialización de los pesos:** Valores al azar del conjunto de entrenamiento

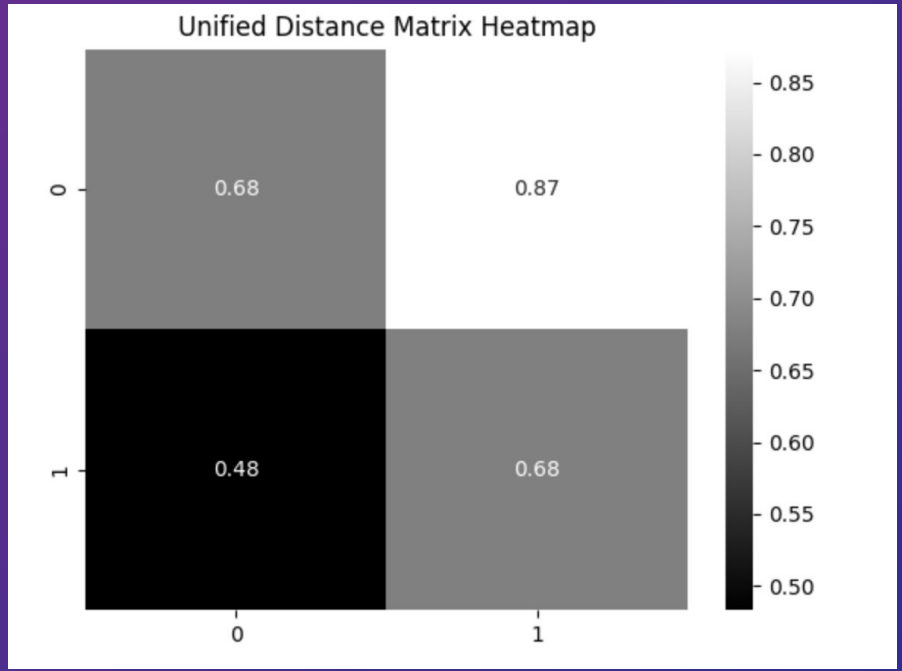
Estandarización de los datos



Resultados Red de Kohonen

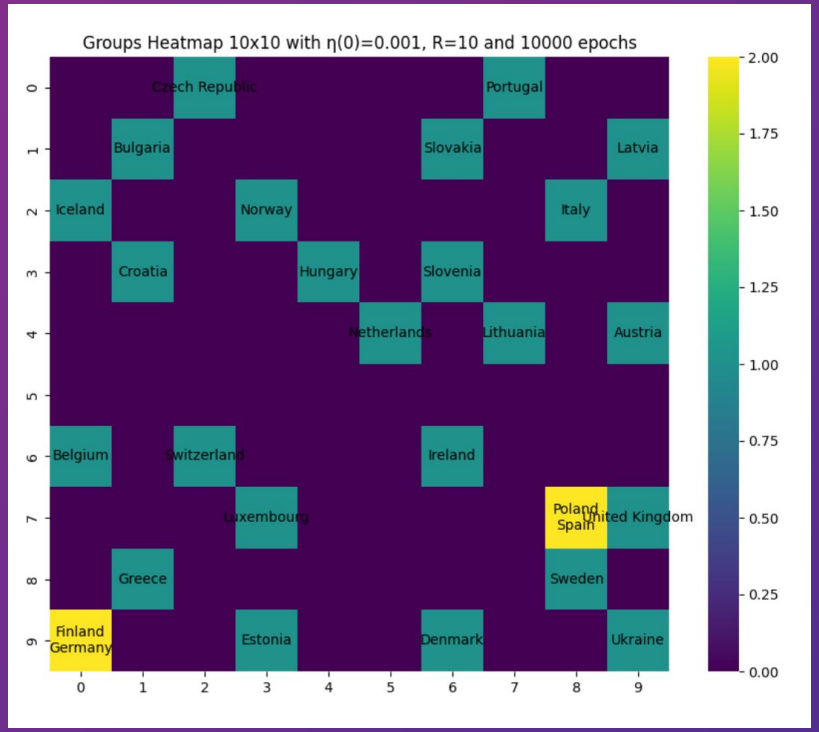
K= 2
Learning rate = 0.001
Épocas = 10000
Radio(0) = 2

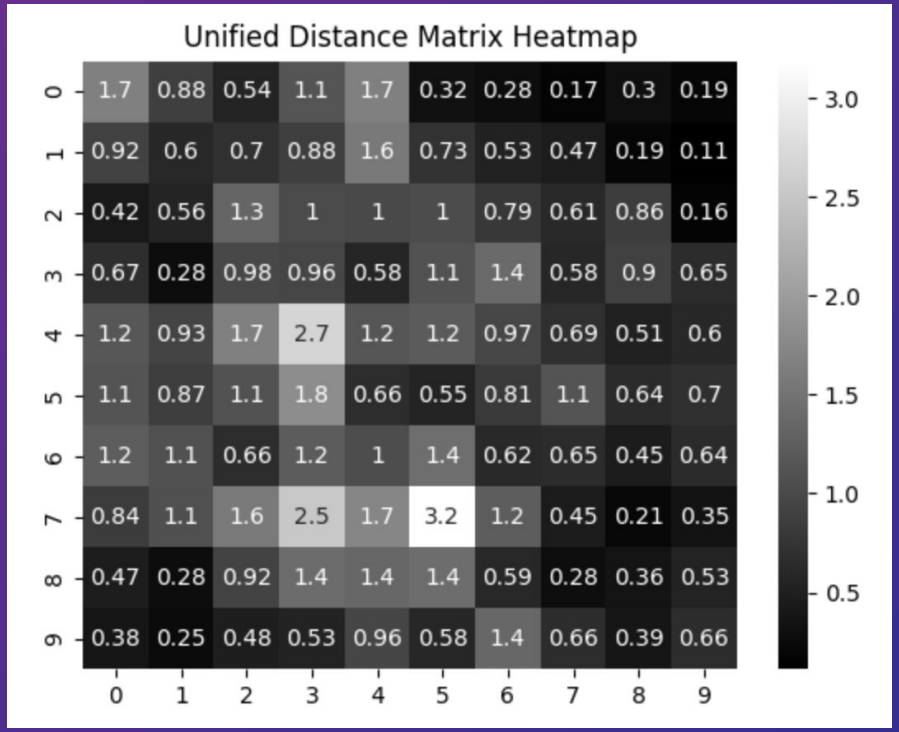




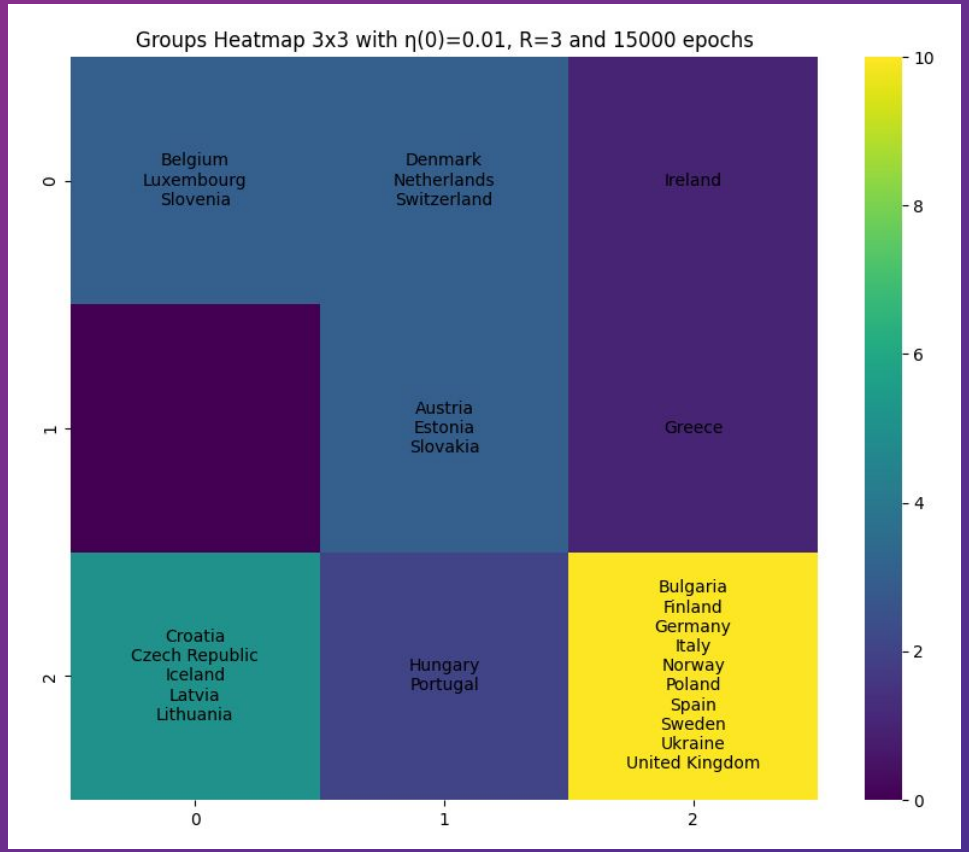
Resultados Red de Kohonen

K= 10
Learning rate = 0.001
Épocas = 10000
Radio(0) = 10

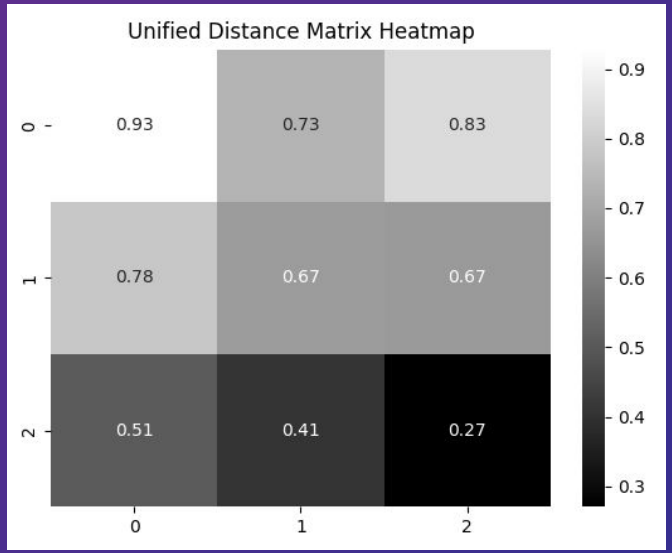




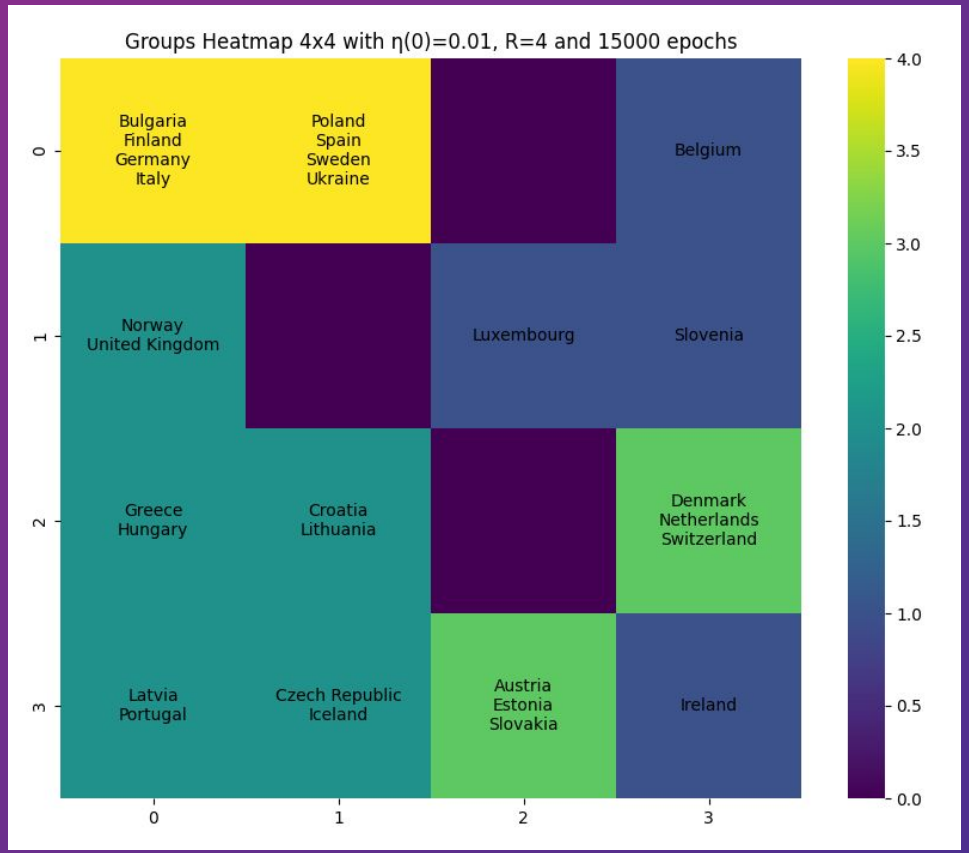
Resultados Red de Kohonen



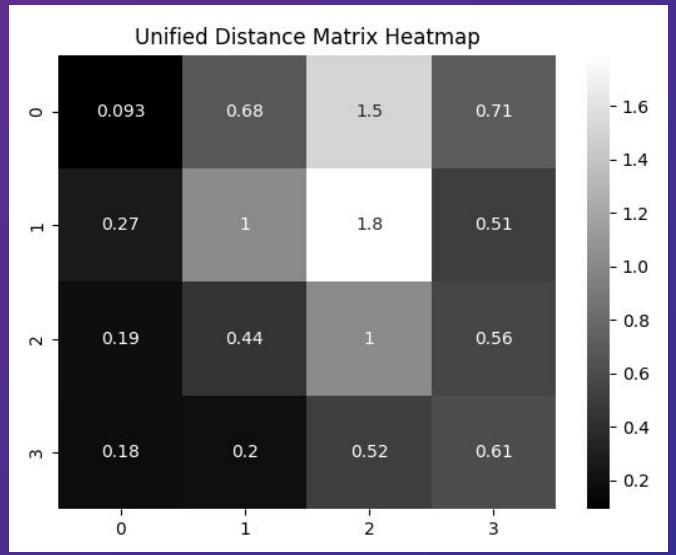
K= 3
Learning rate = 0.01
Épocas = 10000
Radio(0) = 3



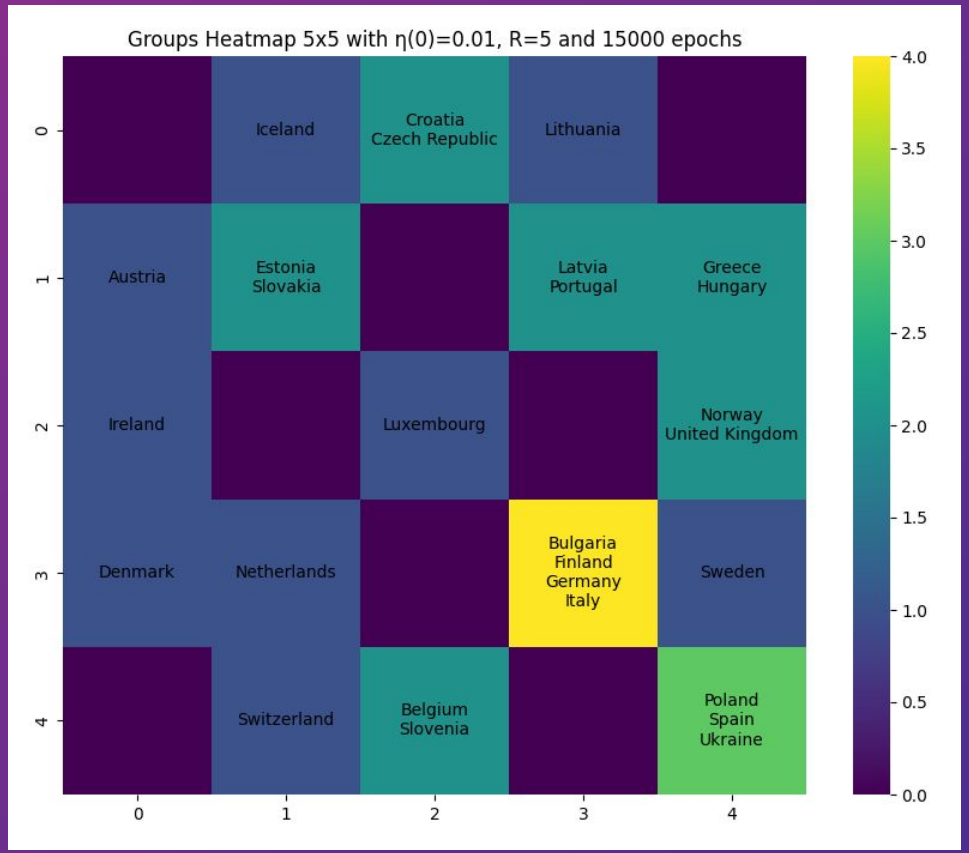
Resultados Red de Kohonen



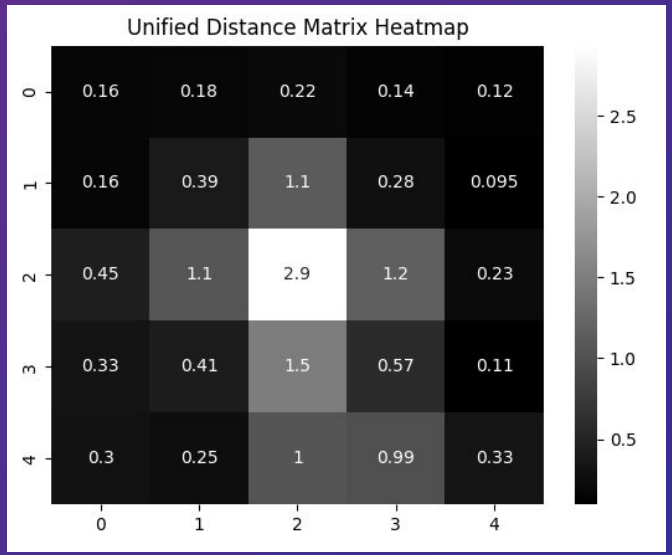
K= 4
Learning rate = 0.01
Épocas = 10000
Radio (0) = 4



Resultados Red de Kohonen

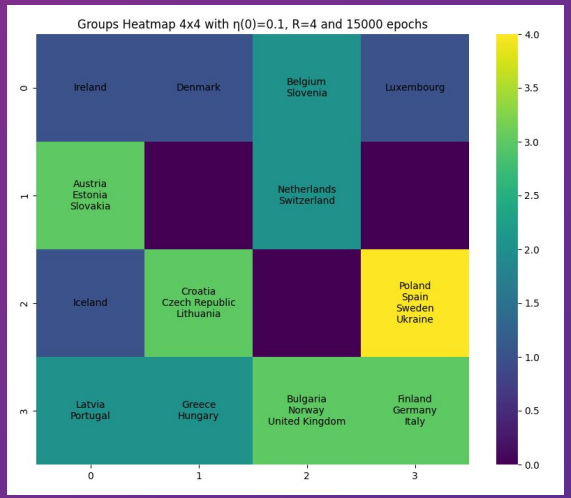


K= 5
Learning rate = 0.01
Épocas = 10000
Radio(0) = 5

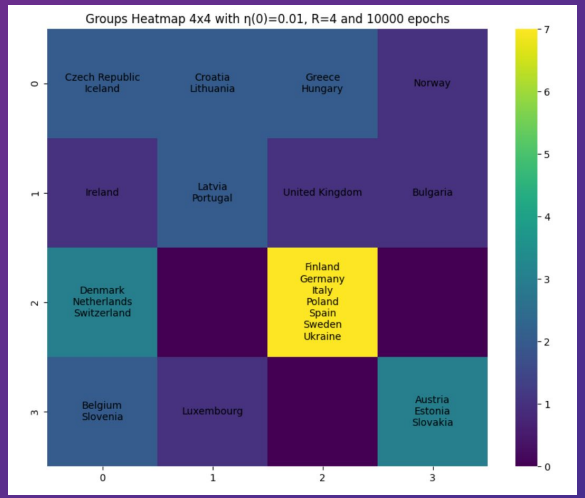


Resultados Red de Kohonen Variando Learning Rate

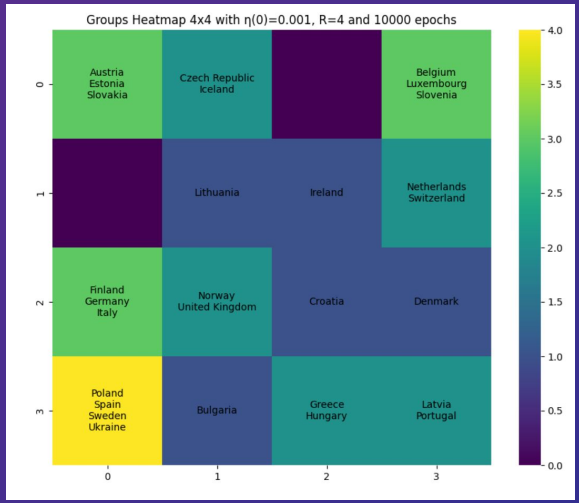
K= 4
Learning rate = 0.1
Épocas = 10000
Radio(0) = 4



K= 4
Learning rate = 0.01
Épocas = 10000
Radio(0) = 4

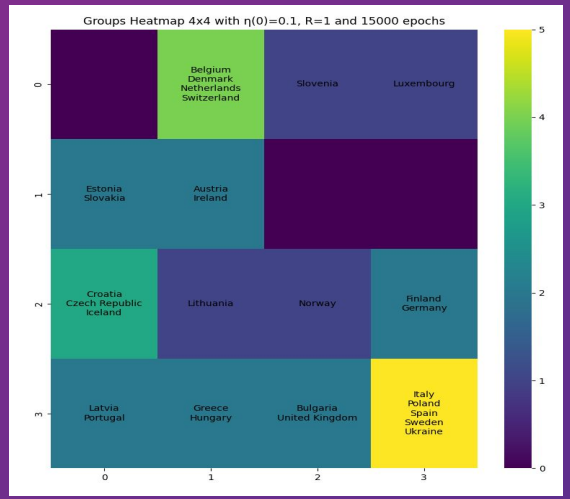


K= 4
Learning rate = 0.001
Épocas = 10000
Radio(0) = 4

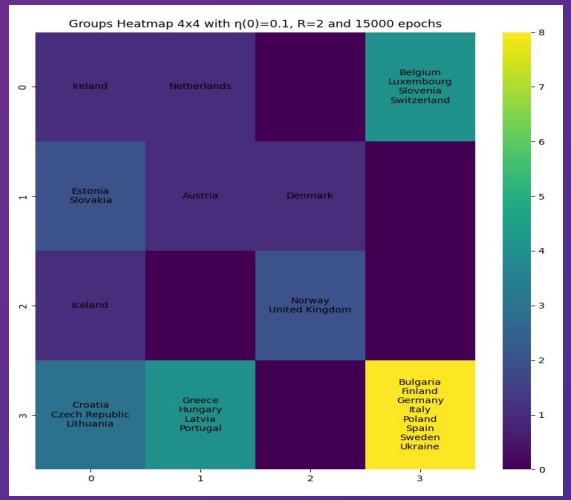


Resultados Red de Kohonen - Radio fijo

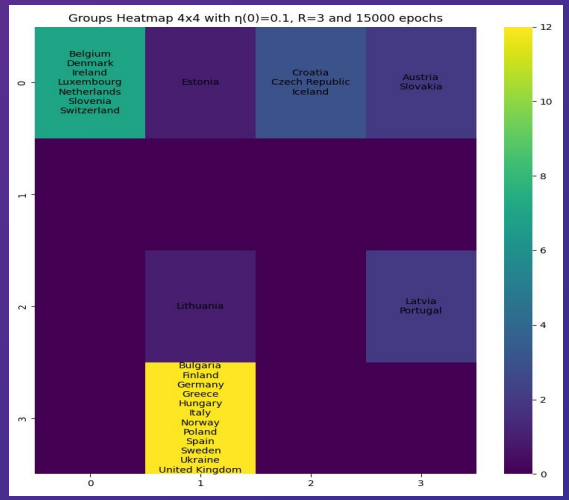
K= 4
Learning rate = 0.1
Épocas = 10000
Radio = 1



K= 4
Learning rate = 0.1
Épocas = 10000
Radio = 2



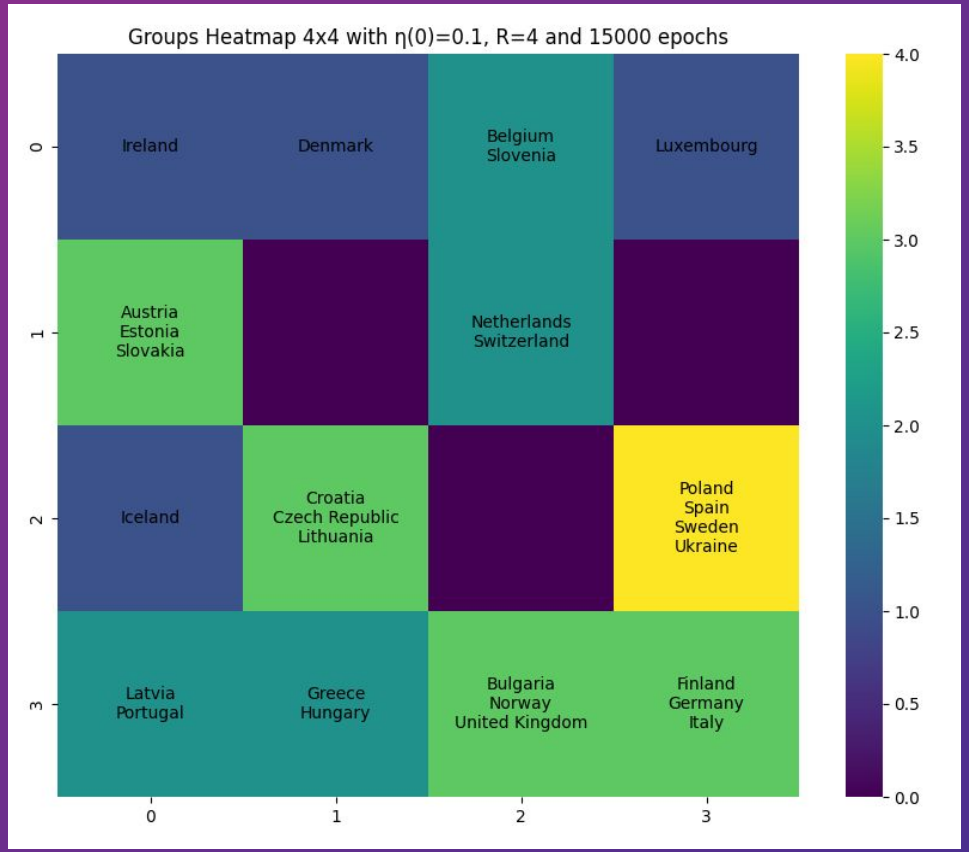
K= 4
Learning rate = 0.1
Épocas = 10000
Radio = 3

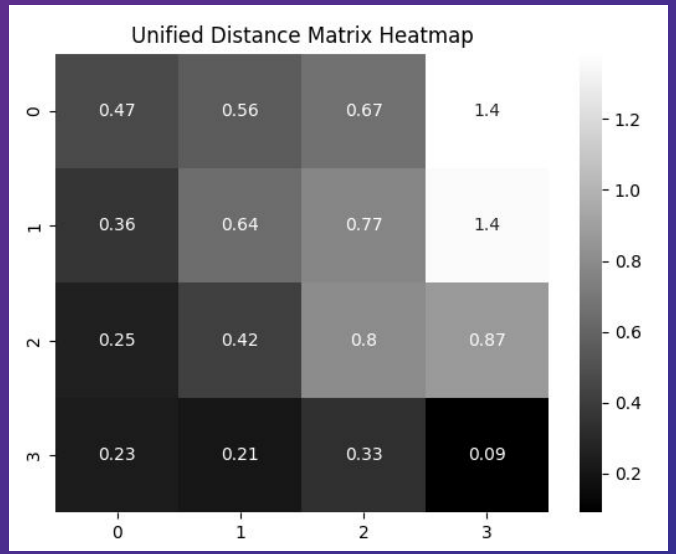


Resultados Red de Kohonen

Configuración elegida:

K= 4
Learning rate = 0.1
Épocas = 15000
Radio(0) = 4



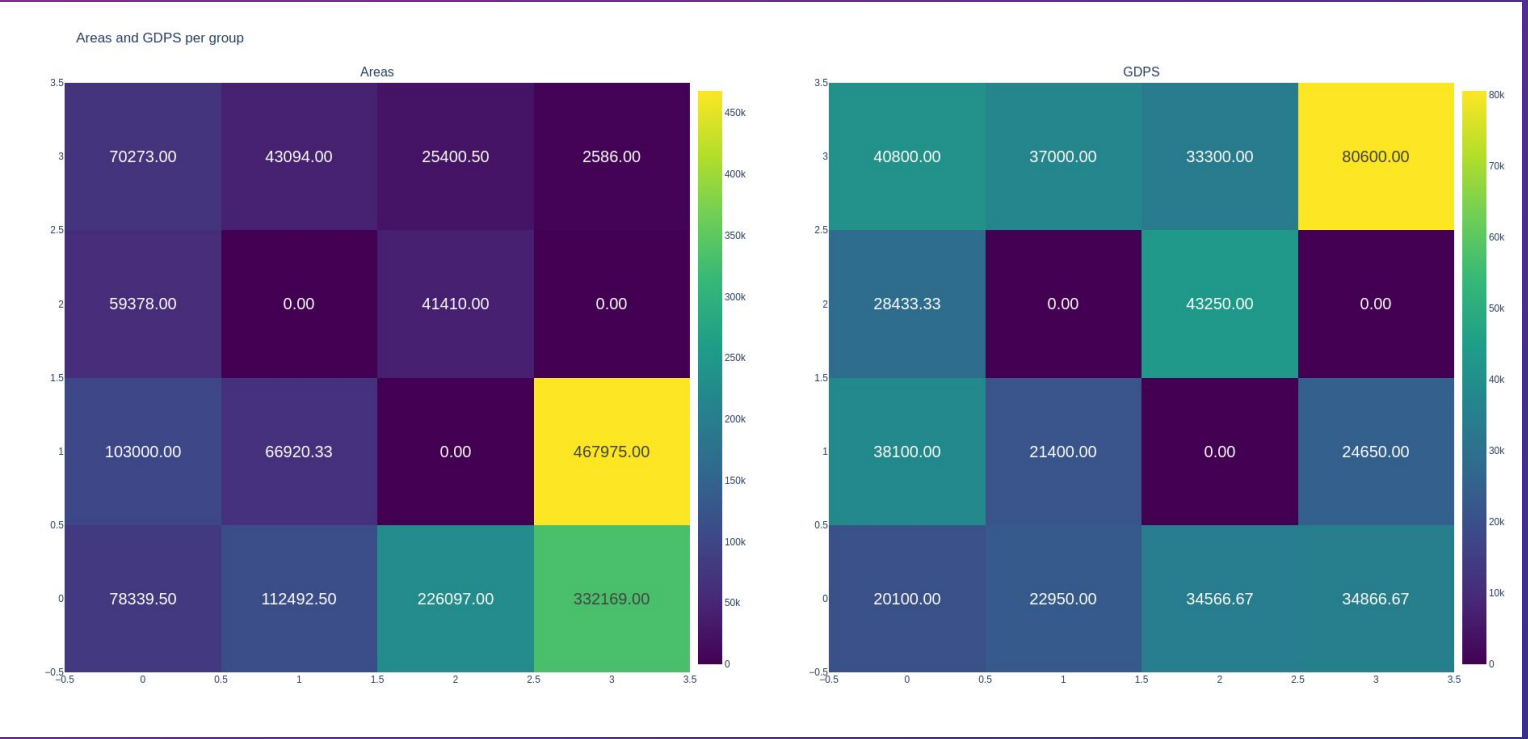


Resultados Red de Kohonen

Mirando area y GDPS

Configuración elegida:

K= 4
Learning rate = 0.1
Épocas = 15000
Radio(0) = 4

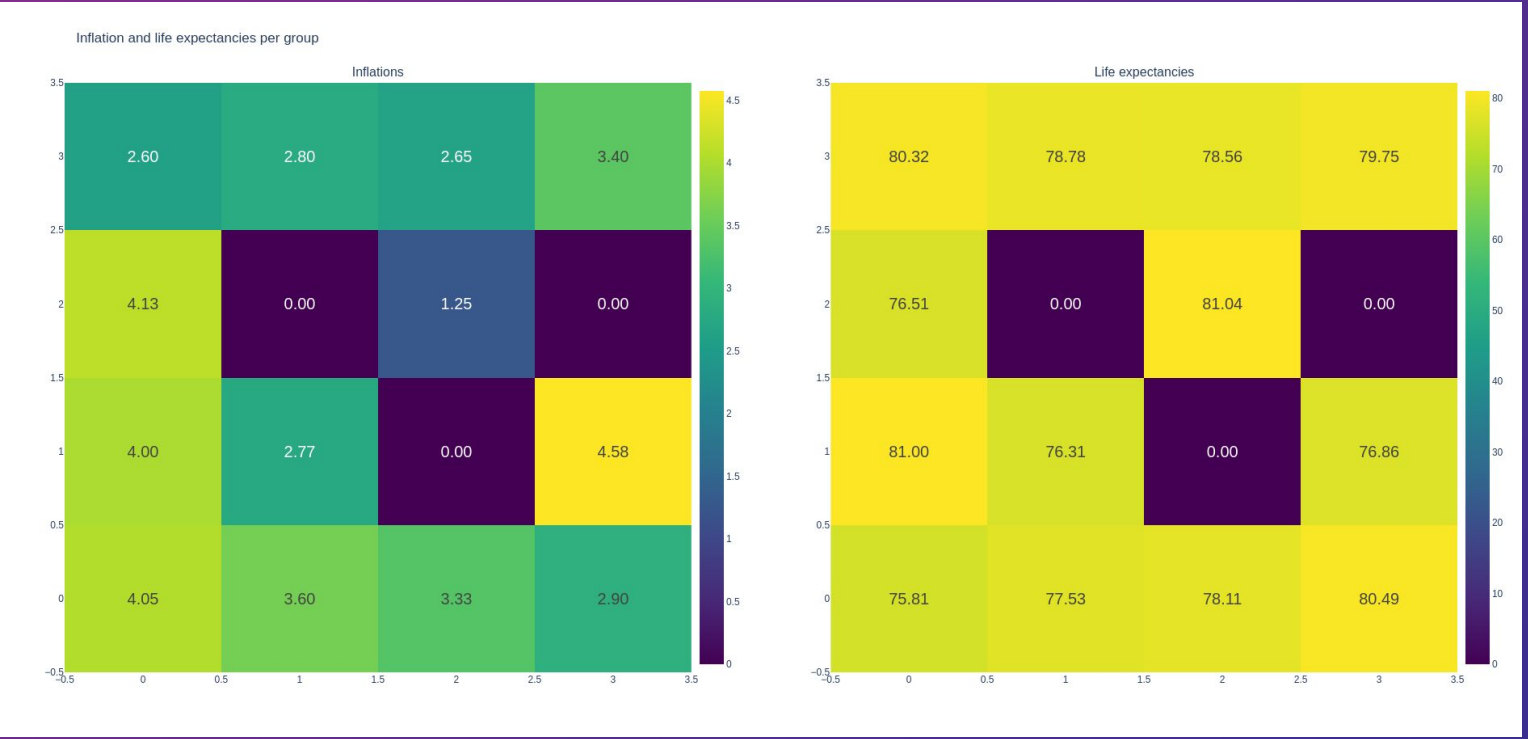


Resultados Red de Kohonen

Mirando Inflación y Expectativa de vida

Configuración elegida:

K= 4
Learning rate = 0.1
Épocas = 15000
Radio(0) = 4

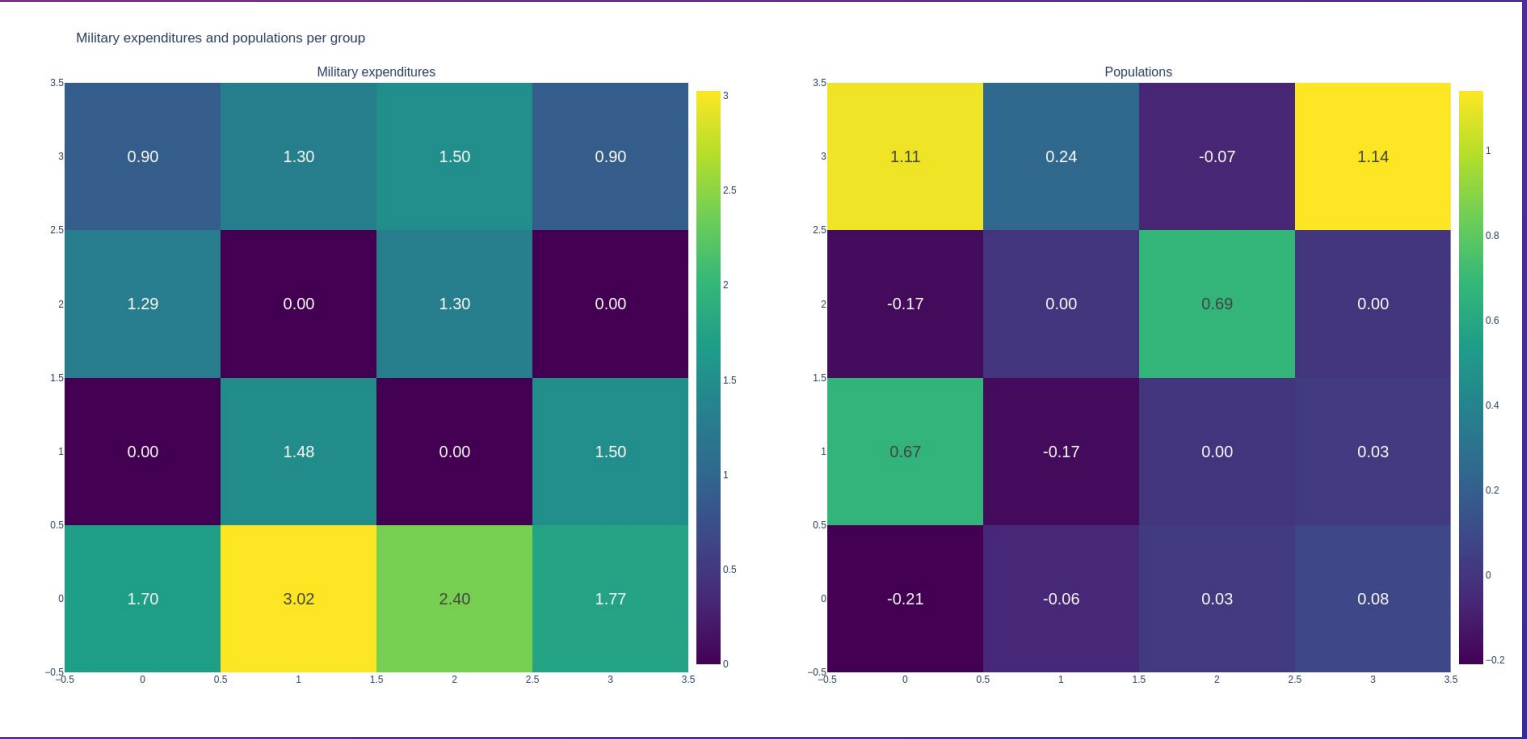


Resultados Red de Kohonen

Mirando Militar y Población

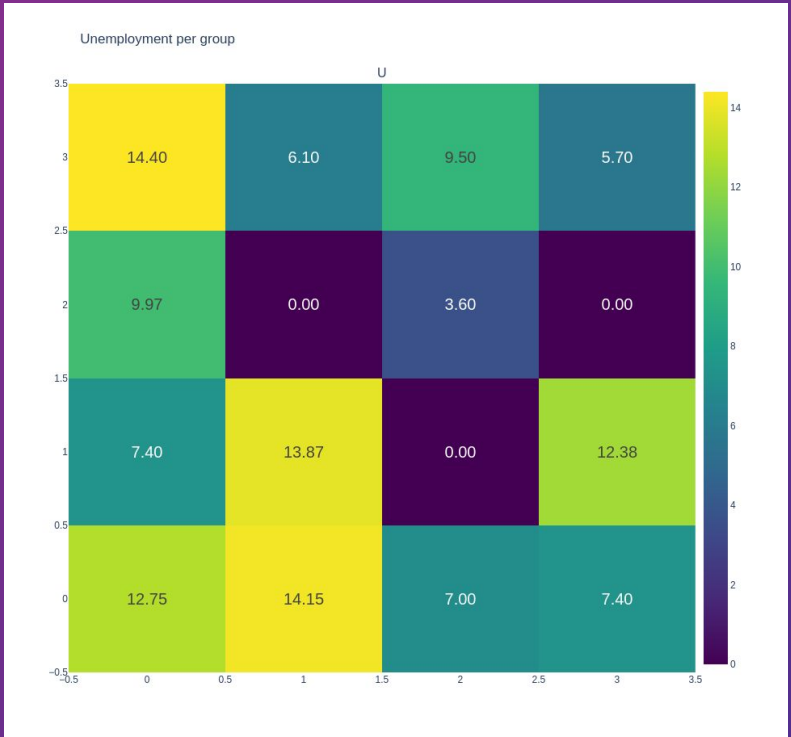
Configuración elegida:

K= 4
Learning rate = 0.1
Épocas = 15000
Radio(0) = 4



Resultados Red de Kohonen

Mirando Desempleo



Configuración elegida:

K= 4
Learning rate = 0.1
Épocas = 15000
Radio(0) = 4

Conclusiones

- Variar el learning rate no parece afectar mucho la clusterización
- Cuando K es muy grande (por ej K=10), los países quedan demasiado distribuidos, y muchas neuronas quedan “muertas”
- Con radio fijo, al usar valores grandes para el radio obtenemos grupos muy grandes, y al usar radio
- La variable que más varía entre países es el **área** y la más similar es la **expectativa de vida**

02

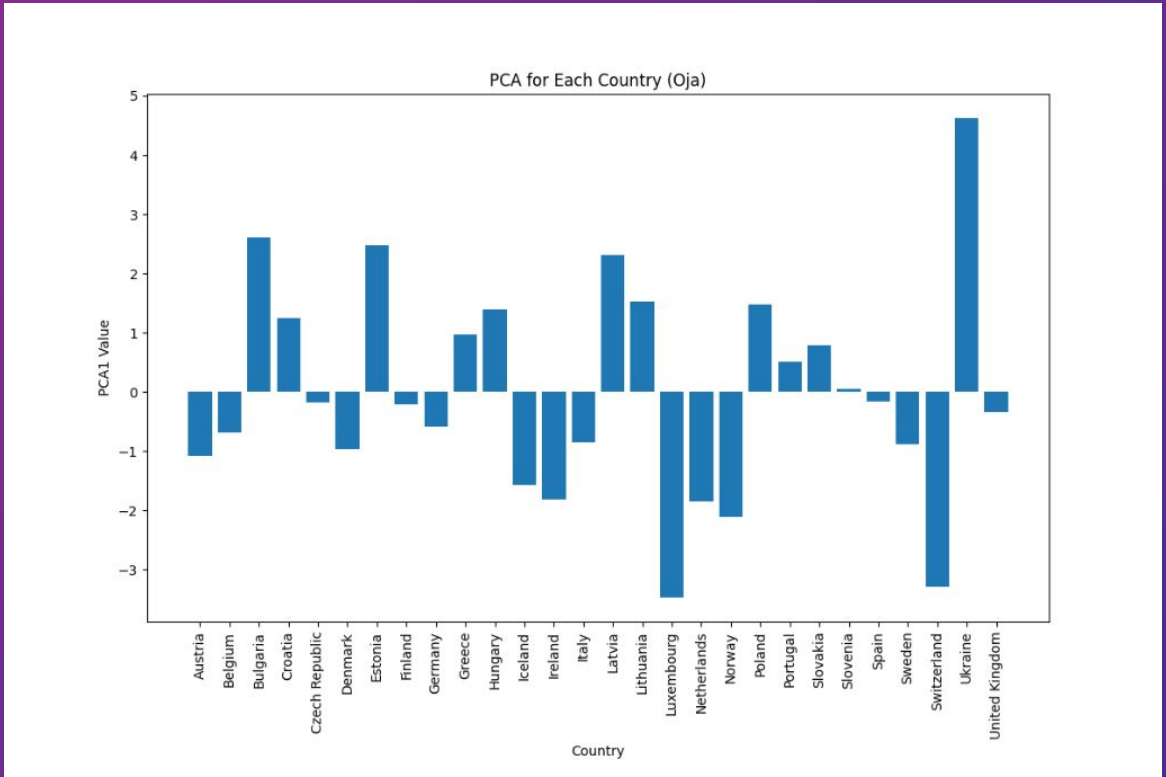
Ejercicio 1.2

Modelo de Oja

Implementar una red neuronal utilizando la regla de Oja para resolver los siguientes problemas:

- Calcular la primer componente principal para este conjunto de datos.
- Interpretar el resultado de la primer componente.
- Comparar el resultado del ejercicio de Oja con el resultado de calcular la primer componente principal con una librería.

Primer Componente Principal



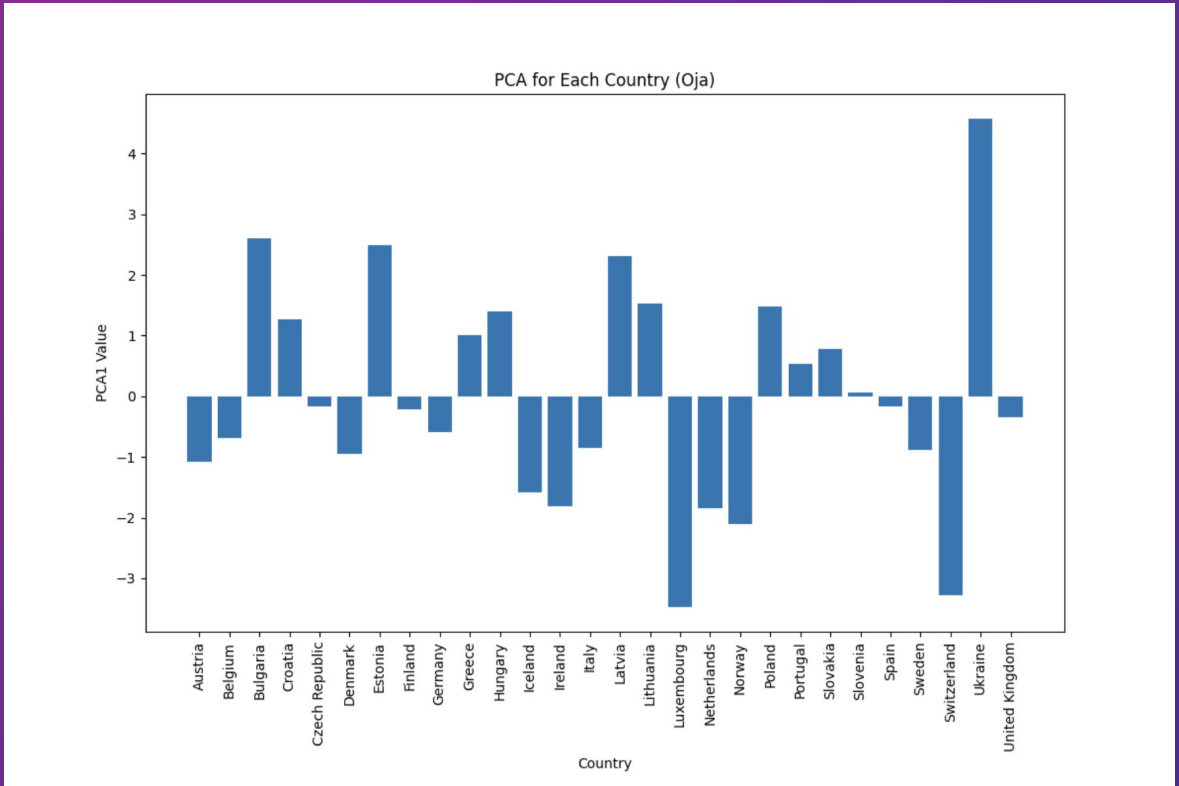
lr = 0.001

Epochs = 10000

Principal Component 1:

- ['Area: 0.125', 'GDP: -0.501', 'Inflation: 0.407', 'Life.expect:-0.483', 'Military: 0.188', 'Pop.growth:-0.476', 'Unemployment: 0.272']

Primer Componente Principal



lr = 0.1

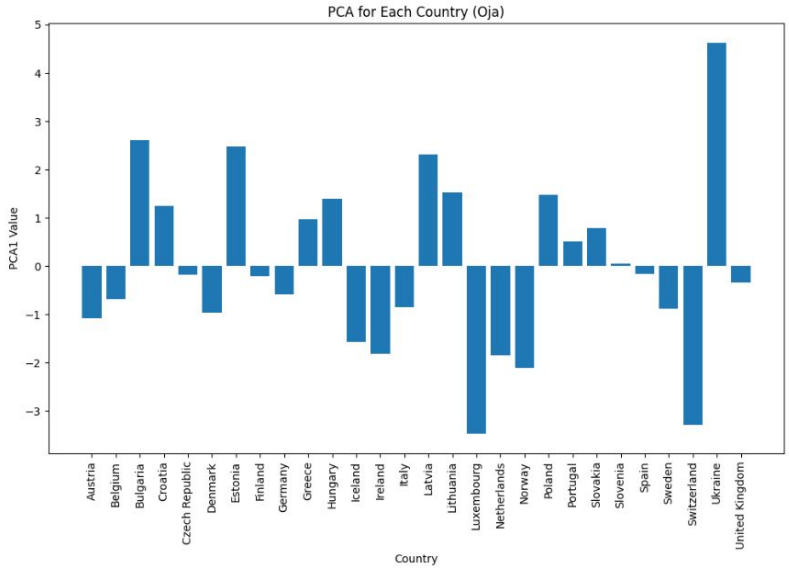
Epochs = 10000

Principal Component 1:

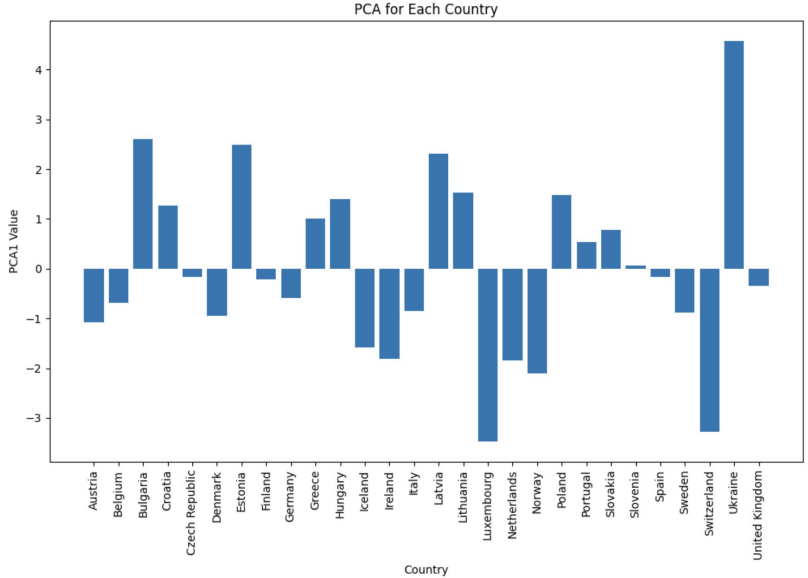
- ['Area: 0.132', 'GDP: -0.500', 'Inflation: 0.414', 'Life.expect:-0.484', 'Military: 0.182', 'Pop.growth:-0.474', 'Unemployment: 0.268']

Comparación con Sklearn

Oja



Sklearn



Comparación con Sklearn

Oja

Principal Component 1:

- ['Area: 0.125', 'GDP: -0.501', 'Inflation: 0.407', 'Life.expect:-0.483', 'Military: 0.188', 'Pop.growth:-0.476', 'Unemployment: 0.272']

Sklearn

Principal Component 1:

- ['Area:0.125', 'GDP:-0.501', 'Inflation:0.407', 'Life.expect:-0.483', 'Military:0.188', 'Pop.growth:-0.476', 'Unemployment:0.272']
- Autovalor: 3.3467 (explica el 46%)

Conclusiones

- Para este análisis un PCA1 bajo indica una buena situación general del país (Luxemburgo, Suiza, Noruega).

- GDP, Inflación, Expectativa de vida y Crec. de Población influencian más que el Desempleo, Presupuesto militar y el Área a la hora de diferenciar países en cuanto a su situación actual.

- Los pesos de la red neuronal de Oja efectivamente convergen al valor de PCA1.

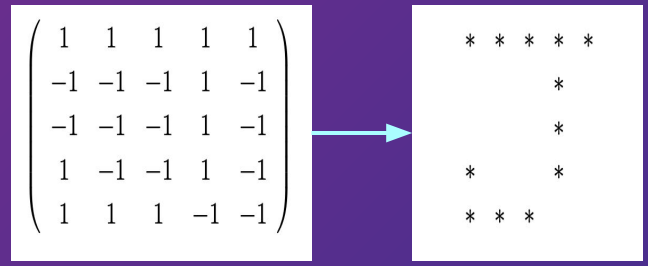
- Una tasa de aprendizaje más chica resultó más efectiva para que la red de Oja converja a PCA1.

03

Ejercicio 2

Ejercicio Patrones - Modelo de Hopfield

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 & -1 \\ 1 & 1 & 1 & -1 & -1 \end{pmatrix}$$



Construir patrones de letras del abecedario utilizando 1 y −1 y matrices de 5×5

- Almacenar 4 patrones de letras. Implementar el modelo de Hopfield para asociar matrices ruidosas de 5 × 5 con los patrones de las letras almacenadas. Los patrones de consulta deben ser alteraciones aleatorias de los patrones originales. Mostrar los resultados que se obtienen en cada paso hasta llegar al resultado final.

- Ingresar un patrón muy ruidoso e identificar un estado espúreo.

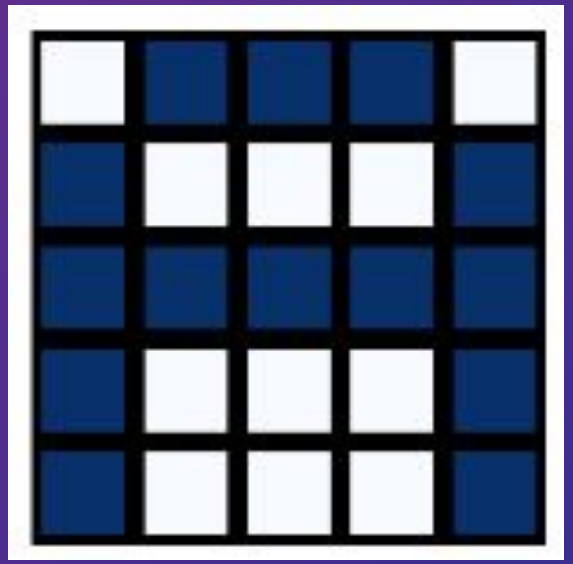
Descripción del problema

- Implementar el modelo de Hopfield para reconocer patrones similares a los ya almacenados
- Generar un conjunto de letras utilizando matrices de 1 y -1
- Analizar la ortogonalidad del conjunto para encontrar los mejores grupos de 4 letras
- Analizar la cantidad de recuperos correctos en función del ruido
- Mostrar como la función de energía es decreciente

Almacenamiento de letras

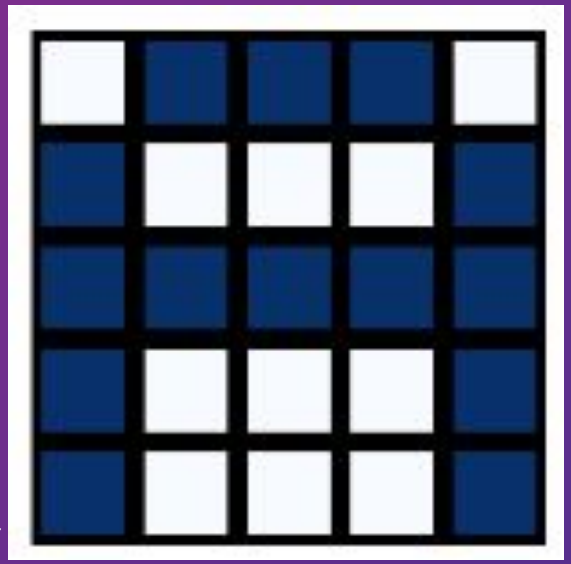
Las letras están almacenadas en un archivo .json de la siguiente forma

```
[
  {
    "name": "A",
    "data": [
      [-1, 1, 1, 1, -1],
      [1, -1, -1, -1, 1],
      [1, 1, 1, 1, 1],
      [1, -1, -1, -1, 1],
      [1, -1, -1, -1, 1]
    ]
  }, {
```

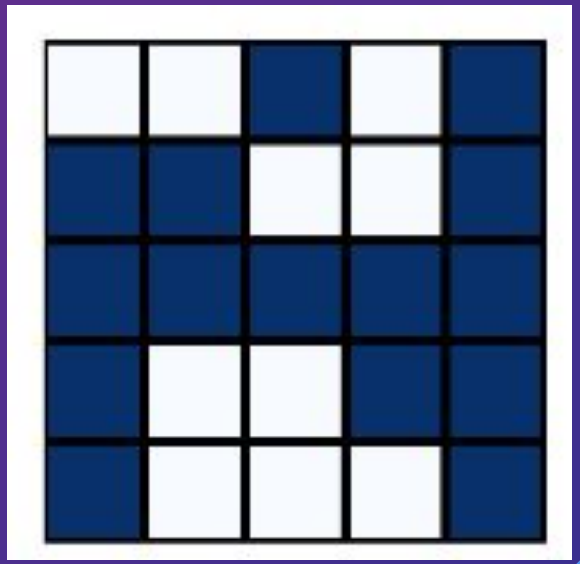


Almacenamiento de letras - Ruido

Letra A sin ruido

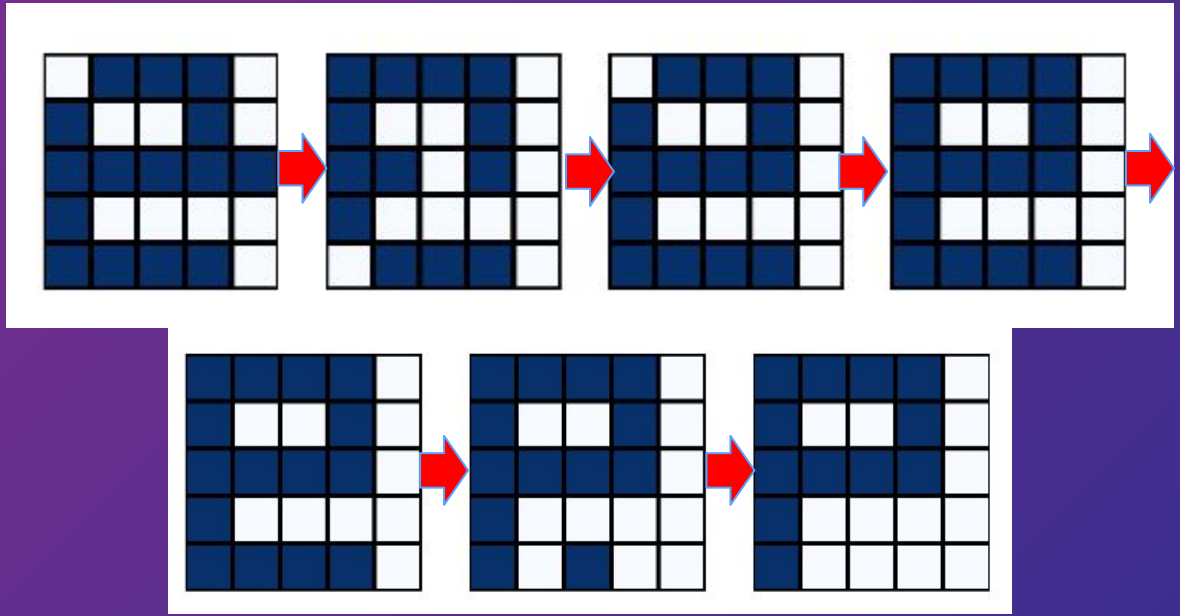


Letra A con 0.2 de ruido



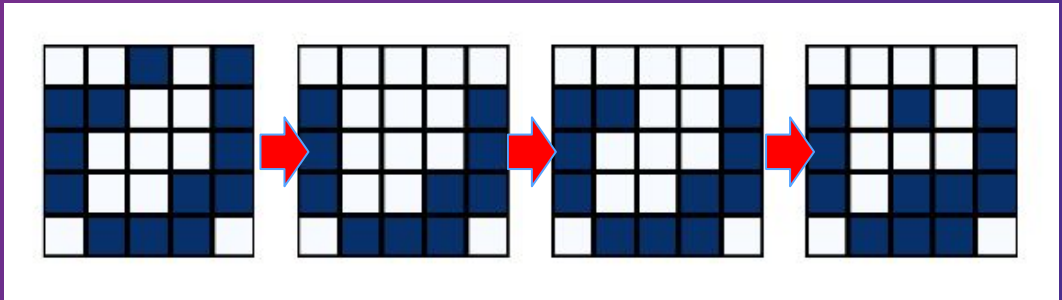
Funcionamiento del modelo

- Evolución paso a paso de los estados
- Modelo entrenado con: O, P, W, Z
- Patrón ruidoso ingresado: P con 0.2 de ruido



Ejemplo de estado espurio

- Evolución paso a paso de los estados
- Modelo entrenado con: O, P, W, Z
- Patrón ruidoso ingresado: O con 0.2 de ruido



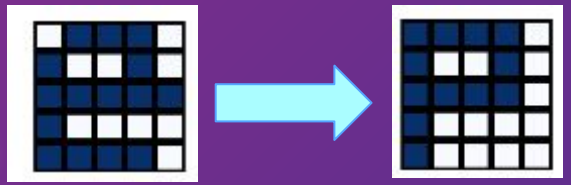
Función de energía

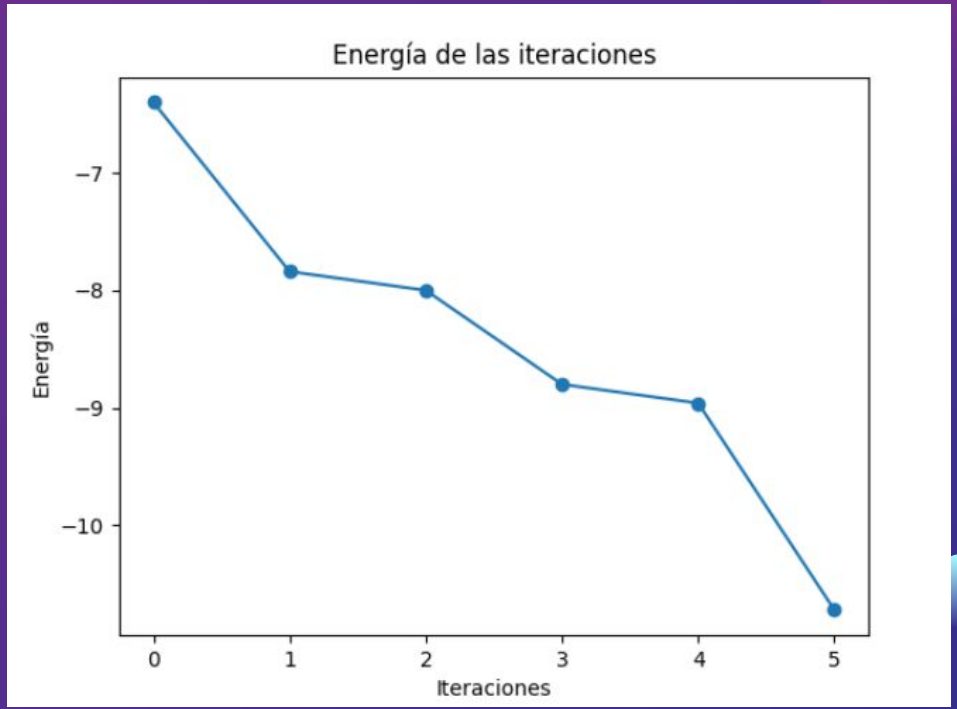
Siempre decrece o permanece constante cuando el sistema evoluciona

$$H(w) = -\frac{1}{2}\sum_{i,j} w_{ij} S_i S_j$$

Función de energía - [O, P, W, Z]

Ruido: 0.2





Análisis de ortogonalidad

Tomamos todas las combinaciones de 4 letras y calculamos el producto interno entre si.

La idea es que cuanto más cercano a 0 mejor debería comportarse el modelo usando tales conjuntos

Nos fijamos en :

- |<, >| Medio
- |<, >| Max

Análisis de ortogonalidad

Mejores candidatos:

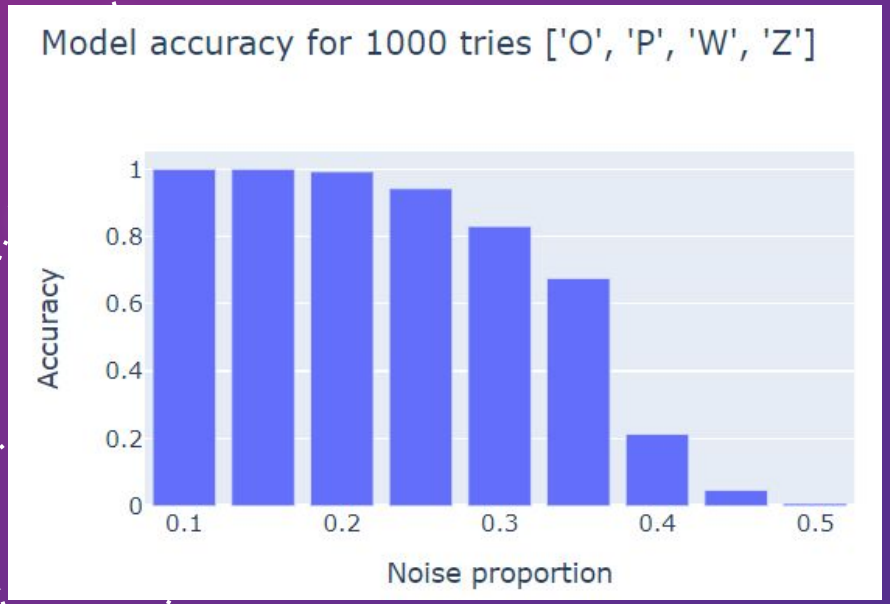
	|<,>| medio	grupo
0	1.666667	(O, P, W, Z)
1	2.000000	(A, J, K, U)
2	2.000000	(D, R, T, V)
3	2.000000	(F, I, L, X)
4	2.000000	(F, O, V, Z)

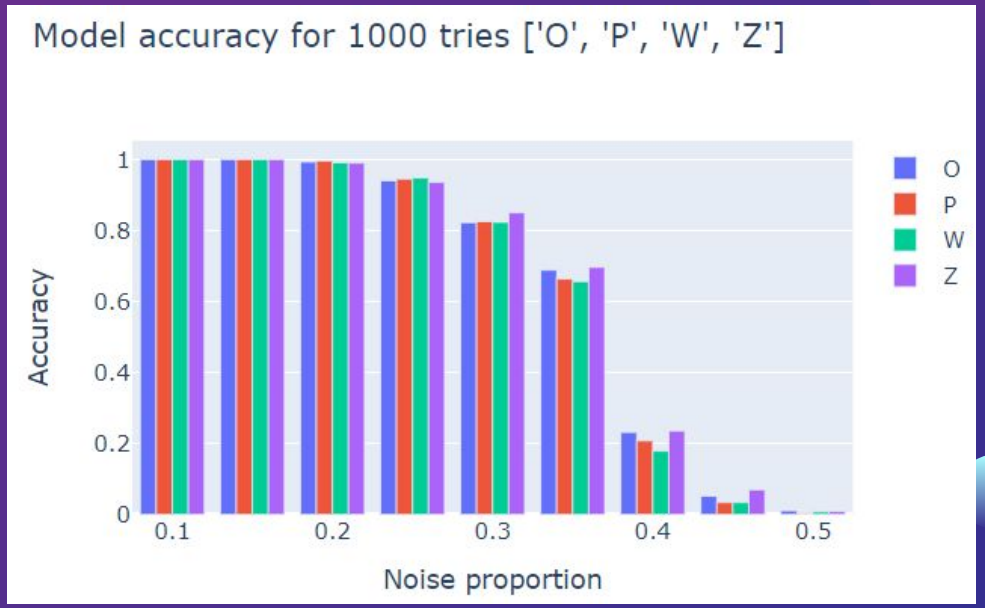
Peores candidatos:

	|<,>| medio	grupo
14945	16.000000	(B, C, D, O)
14946	16.000000	(C, D, O, X)
14947	16.333333	(K, M, N, W)
14948	16.666667	(C, D, G, O)
14949	18.333333	(H, M, N, W)

Análisis de ruido - [O, P, W, Z]

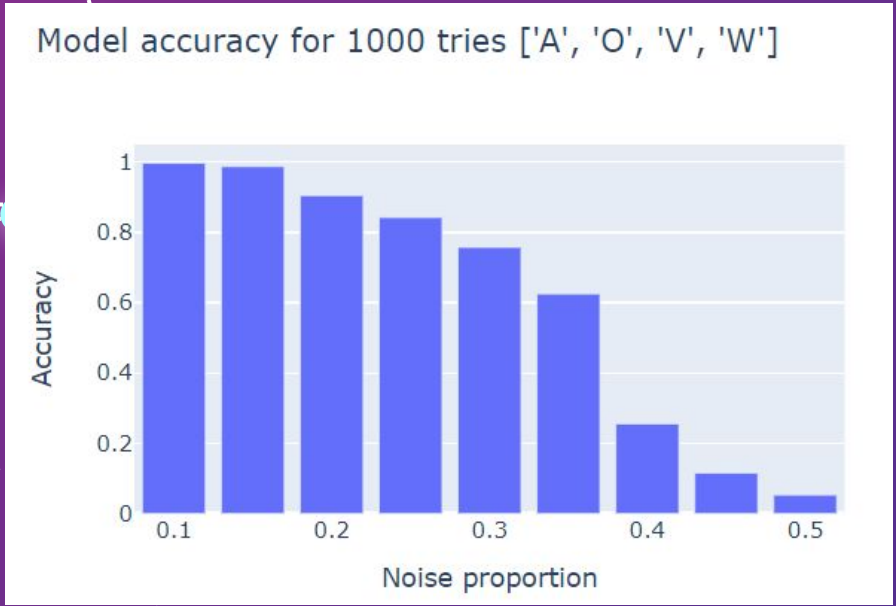
|<, >| Medio: 1.667

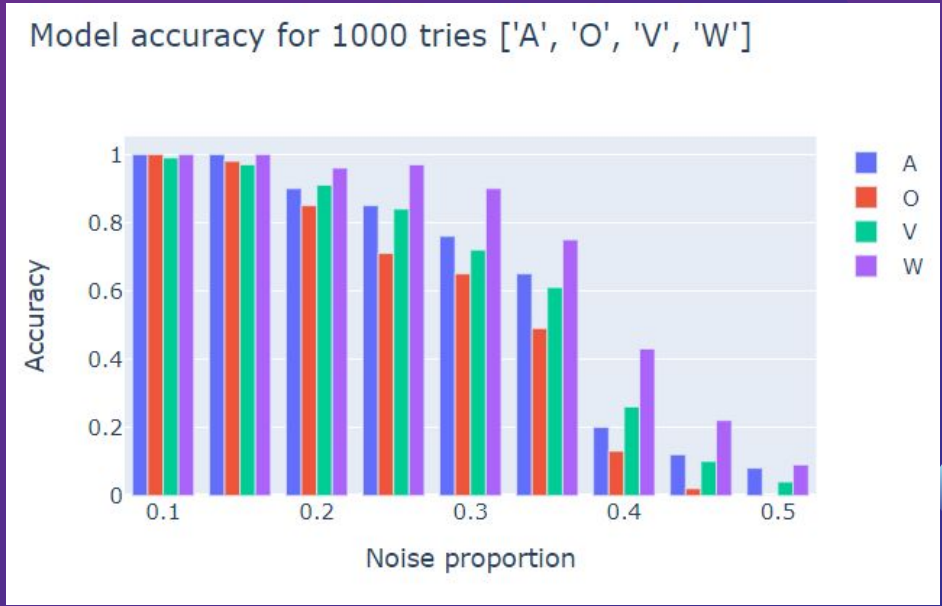




Análisis de ruido - [A, O, V, W]

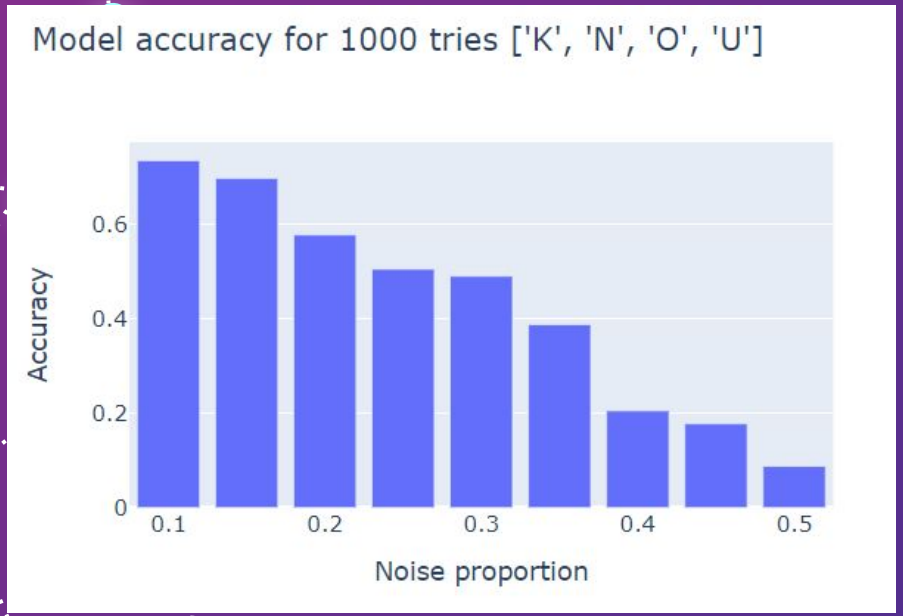
|<, >| Medio: 6.333

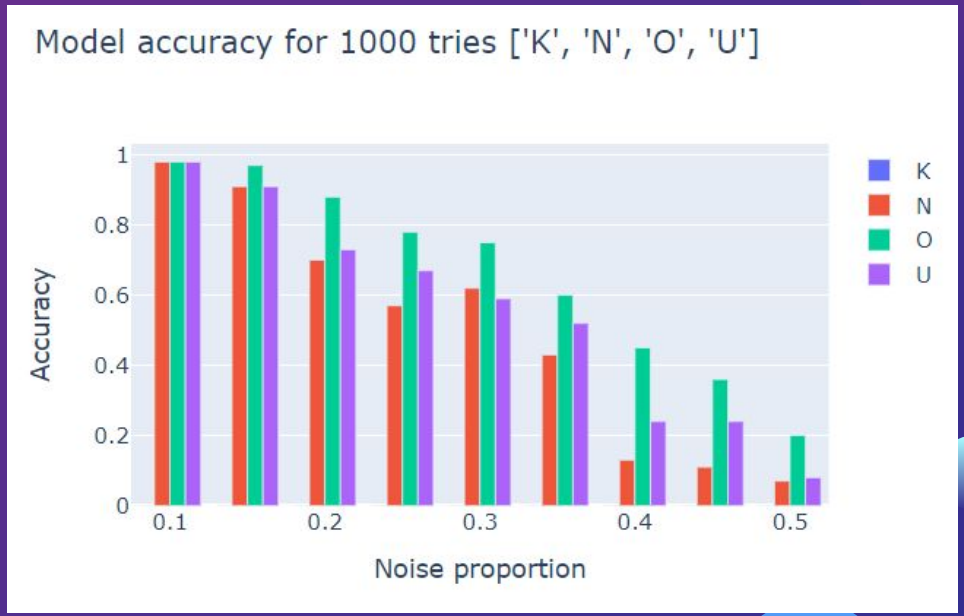




Análisis de ruido - [K, N, O, U]

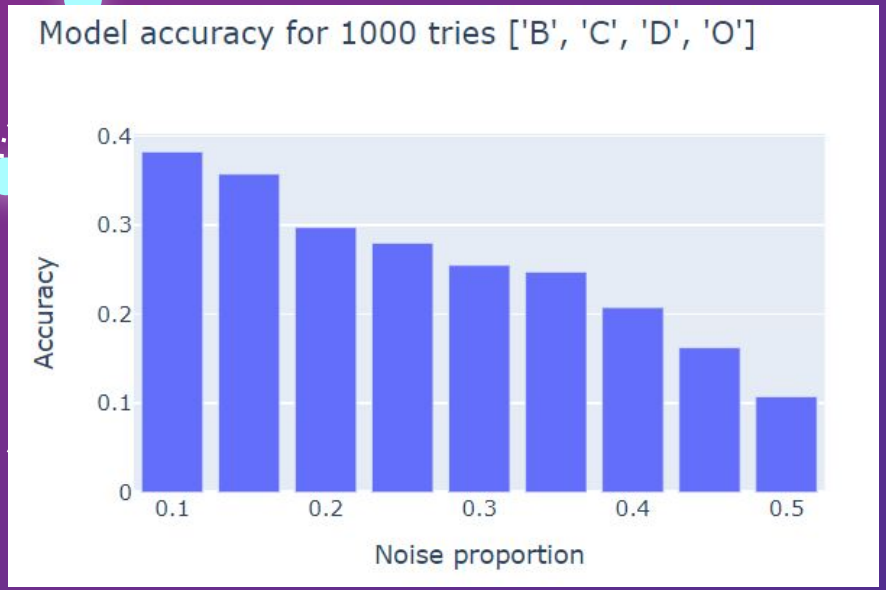
|<, >| Medio: 8.0

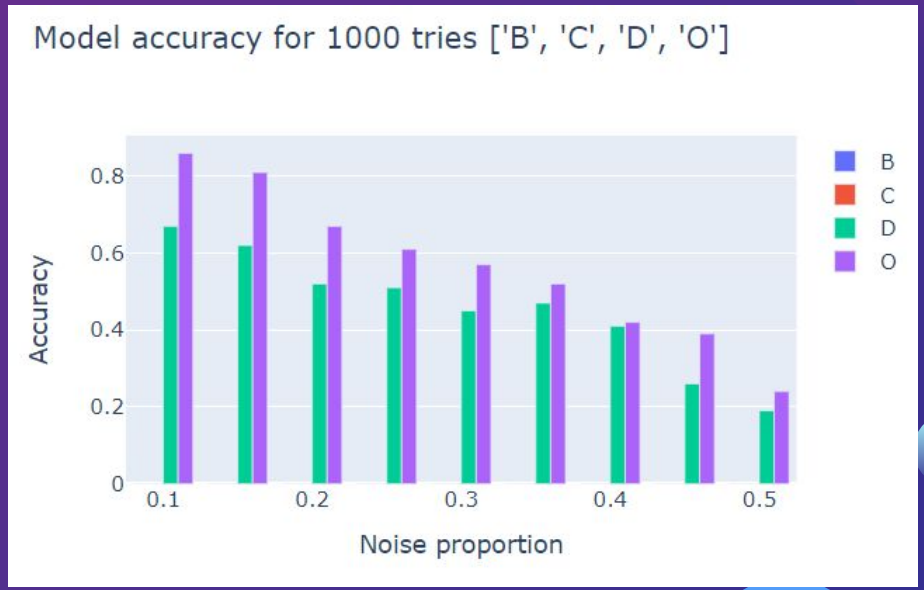




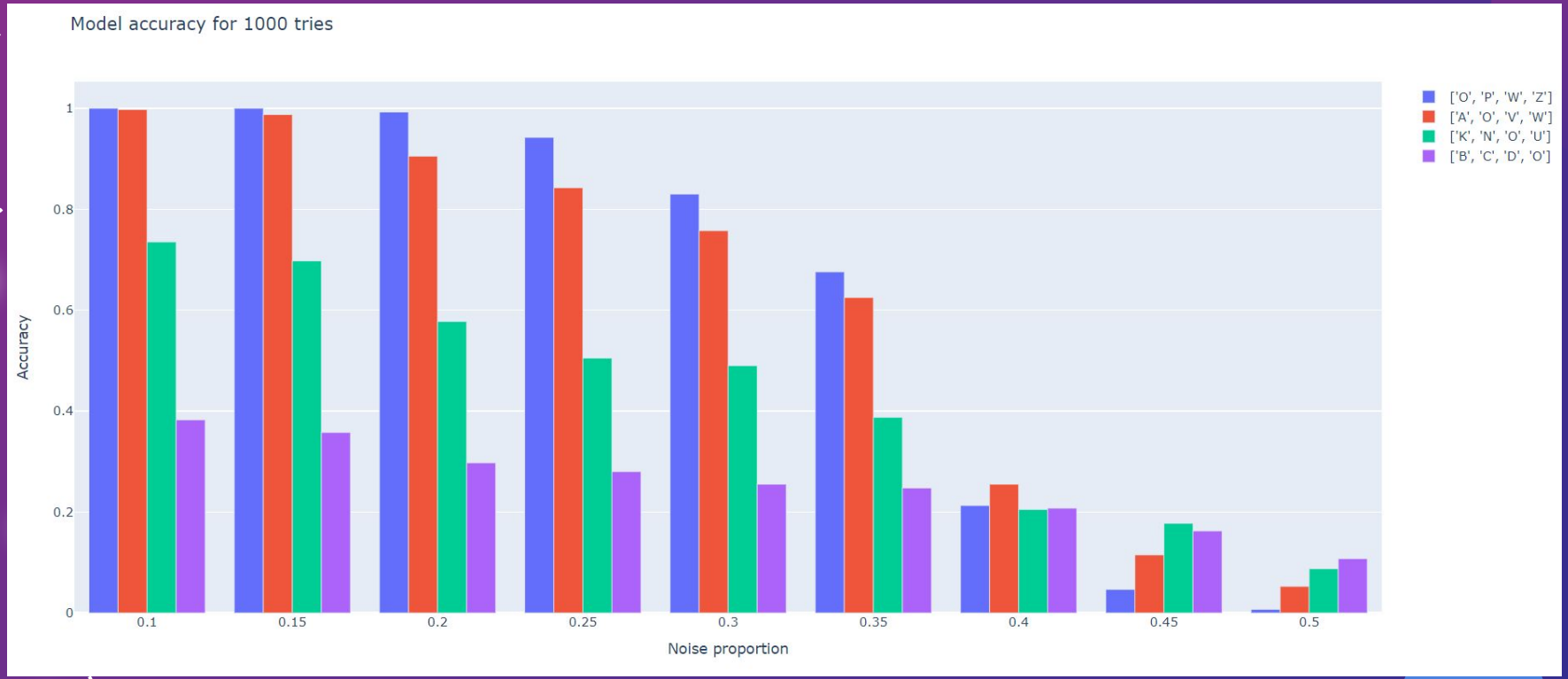
Análisis de ruido - [B, C, D, O]

|<, >| Medio: 16.00





Análisis de ruido - Todos los conjuntos



Análisis de ortogonalidad

|<, >| Medio

```
3      2.000000  (F, I, L, X)
4      2.000000  (F, O, V, Z)
```

|<, >| Max, Apariciones

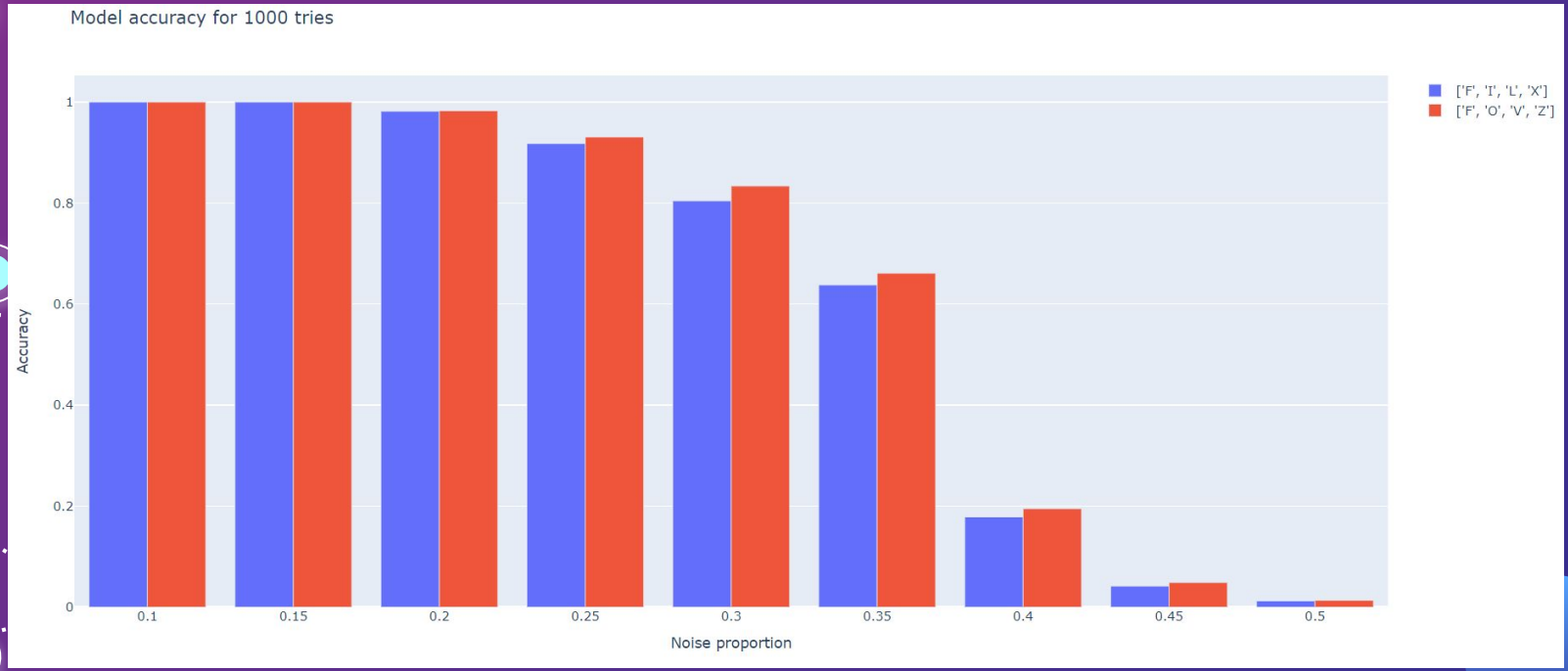
```
2    (3, 3.0)   (F, O, V, Z)
19   (5, 1.0)   (F, I, L, X)
```

Ambos tienen el mismo promedio pero distinta distribución de los datos
Hay impacto?

Análisis de ruido - [F, I, L, X] vs [F, O, V, Z]

FILX: |<, >| Medio: 2
|<, >| Max, Count: 5, 1

FOVZ: |<, >| Medio: 2
|<, >| Max, Count: 3, 3



Conclusiones

- El análisis de ortogonalidad fue sumamente importante para encontrar el conjunto que mejor funciona del modelo
- Pudimos ver que para un mismo promedio de productos internos, la distribución de los mismos hace una diferencia significativa
- Confirmamos que la función de energía es decreciente o constante a medida que evoluciona el modelo

FIN