Trabajo práctico 3
Redes Neuronales y Aprendizaje Profundo

Ignacio Ezequiel Cavicchioli
Padrón 109428
icavicchioli@fi.uba.ar

2/12/2025

Índice

1. Introducción

Este documento presenta el desarrollo de las consignas del trabajo práctico N°3 de la materia de **Redes Neuronales y Aprendizaje Profundo**. El código correspondiente fue realizado en *Jupyter notebooks*, *Python*, adjuntados a la entrega en formato PDF. Toda imagen o implementación requeridas para el análisis se explicitarán en el presente archivo, por lo que la lectura del código en sí queda a discreción del lector. La teoría relevante será presentada y discutida en la sección pertinente.

2. Teoría sobre redes de Kohonen y SOM

Las *Self-Organizing Maps* (SOM), o *redes de Kohonen*, son un tipo de red neuronal no supervisada diseñada para proyectar datos de alta dimensionalidad sobre una rejilla de menor dimensión, preservando en lo posible la topología del espacio original. Este proceso permite visualizar, agrupar y extraer estructuras latentes en los datos.

La palabra "rejilla" se refiere a la estructura informática/topológica donde viven las neuronas. Esta puede ser:

- 1D: una línea (vector de neuronas) que se cierra o no en los extremos.
- 2D: una matriz, que se puede o no cerrar en sus lados.
- 3D o más: un tensor de orden mayor.

El mecanismo de entrenamiento está inspirado en el aprendizaje hebbiano: las neuronas que responden de manera similar a un mismo estímulo se refuerzan conjuntamente. Sin embargo, la SOM incorpora además un esquema de *competitive learning*, donde sólo la neurona ganadora (y sus vecinas en la rejilla) actualizan sus pesos (o lo que corresponda según la función de vecindad). Esto genera la organización progresiva del mapa y produce un *feature mapping* explícito, ordenando los patrones según similitud.

Las SOM actúan como un método de reducción de dimensionalidad no lineal, construyendo representaciones espaciales que capturan regularidades estadísticas de los datos sin requerir etiquetas. Esto mismo se va a ver en la tercer consigna de este trabajo, donde se pasa de un espacio dimensional de N dimensiones al espacio 2D de la rejilla de neuronas, y se usa la matriz U para ratificar la existencia de *clusters*.

3. Ejercicio 1

3.1. Consignas

Construya una red de Kohonen de 2 entradas que aprenda una distribución uniforme dentro del círculo unitario. Mostrar el mapa de preservación de topología. Probar con distribuciones uniformes dentro de otras figuras geométricas.

3.2. Desarrollo

Dado lo expuesto en la teoría sobre redes de Kohonen y lo visto en clase, se espera que las redes implementadas y entrenadas puedan aprender las distribuciones uniformes que se les van a dar en este inciso. Visualmente, esto debería notarse como que las neuronas están aproximadamente equidistantes las unas de las otras, siempre dentro de los límites de la distribución.

Ahora bien, para generar muestras aleatorias uniformes en un círculo hay que pensar que se quiere la misma probabilidad de caer en cualquier parte de su área. Como el área de esta figura depende del radio, sortear un r y θ distribuidos uniformemente entre $(0, 1)$ y $(0, 360°)$ respectivamente generaría una distribución conjunta con mayor densidad de muestras en el centro, indicativo de mayor probabilidad.

Para balancear esto, se ajusta la distribución del radio sorteado a $r = R\sqrt{u}$ con $u \sim \mathcal{U}\{0, 1\}$.

Con un r y *theta* correctamente generados, se procede a hacer una conversión a coordenadas cartesianas, que van a usar las neuronas.

Las otras figuras geométricas fueron generadas de forma similar, haciendo uso de cambios de coordenadas para mantener la uniformidad.

Para entrenar la red de Kohonen utilizada en este ejercicio, se implementó una versión vectorizada del algoritmo, que es más rápida que *loopear* por todas las neuronas. La inicialización de los pesos se realizó tomando muestras aleatorias del propio conjunto de datos. Durante el entrenamiento, los pesos se modificaron mediante una función de vecindad gaussiana, de varianza linealmente decreciente con las iteraciones, que debería ayudar con la organización del mapa. La tasa de aprendizaje se mantuvo constante.

Para cada muestra, se determina la neurona ganadora (*Best Matching Unit*, BMU) y se actualizan tanto ella como sus vecinas de acuerdo con la función de vecindad. Esto se hace una cantidad de iteraciones determinada por el usuario. Para este caso, el algoritmo se ejecutó hasta ver un resultado invariante.

las imágenes de a continuación muestran las figuras generadas con las neuronas en sus ubicaciones aprendidas. Las figuras tienen 1000 muestras (puntos de datos), y hay 25 neuronas distribuidas en una matriz de $5x5$, razón por la cual se forma una *lattice* (rejilla) de polígonos de 4 lados; cada neurona se conecta a otras 4.

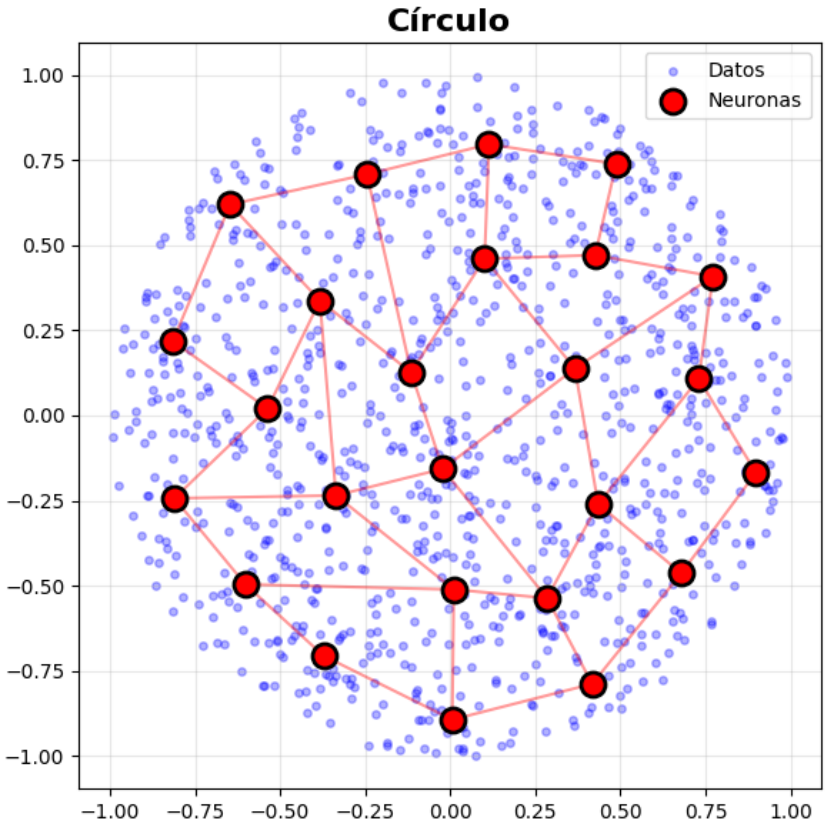

Figura 1: Distribución de datos con forma de círculo y posiciones finales de las neuronas de la red luego del entrenamiento.

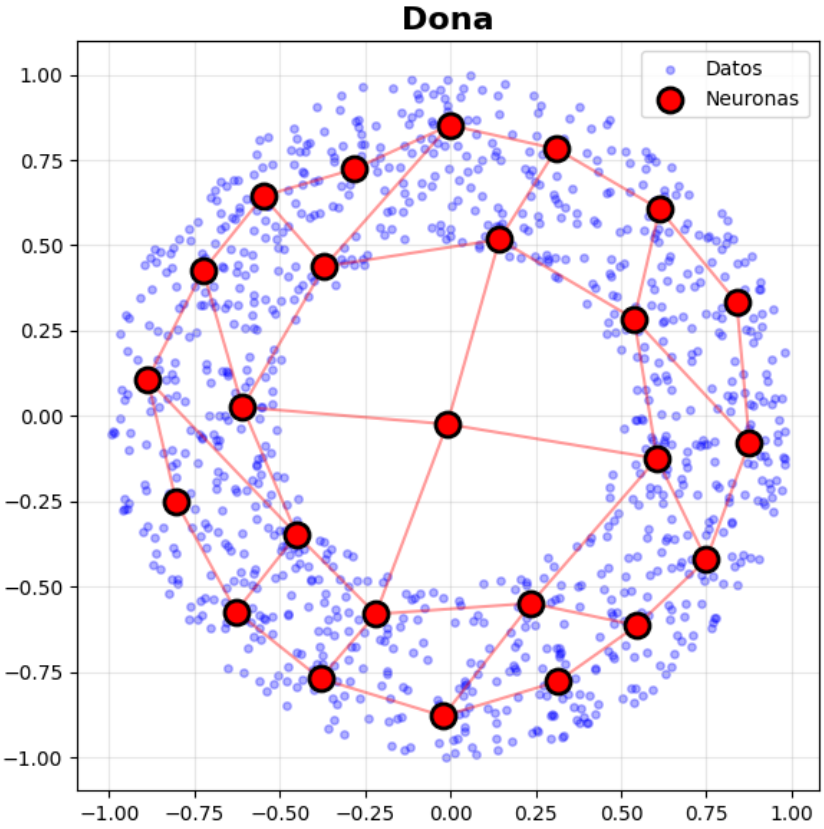

Figura 2: Distribución de datos con forma de dona y posiciones finales de las neuronas de la red luego del entrenamiento.

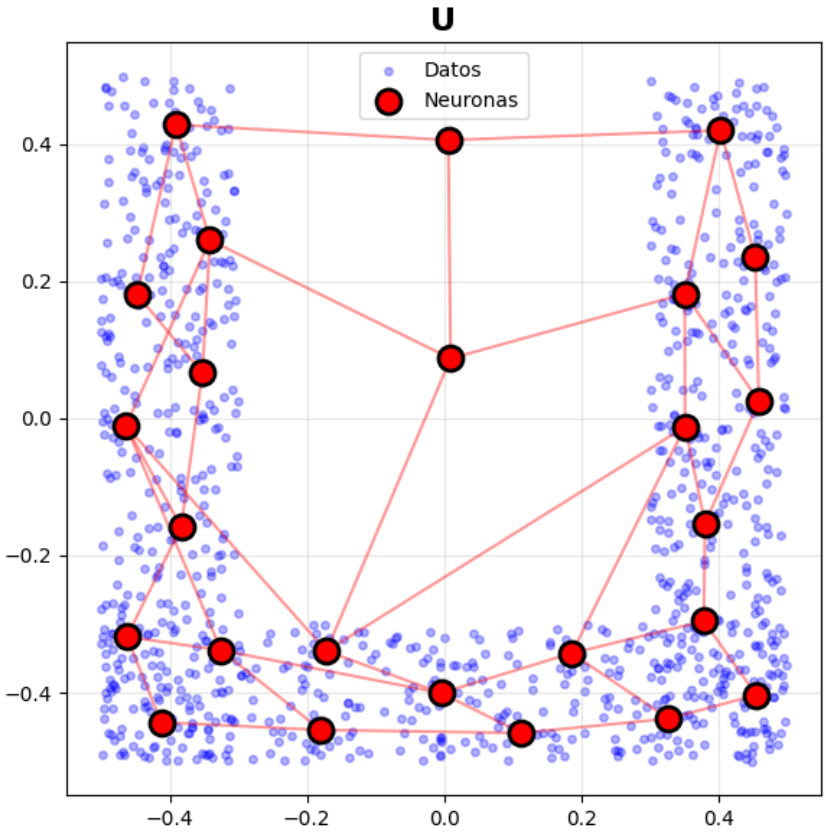

Figura 3: Distribución de datos con forma de U y posiciones finales de las neuronas de la red luego del entrenamiento.

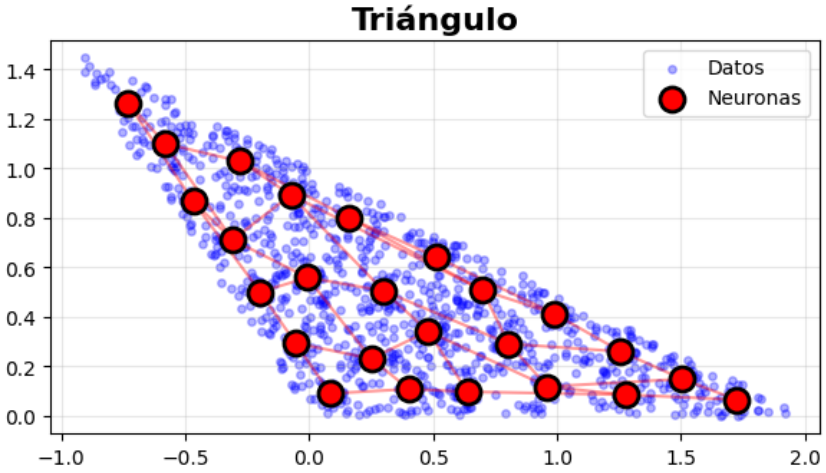

Figura 4: Distribución de datos con forma de triángulo y posiciones finales de las neuronas de la red luego del entrenamiento.

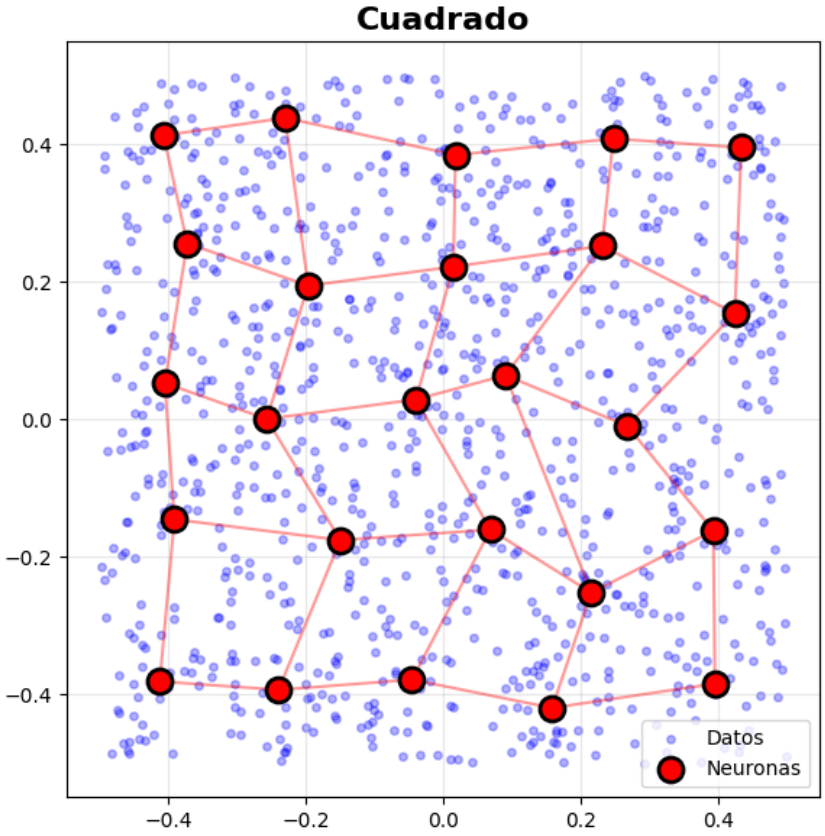

Figura 5: Distribución de datos con forma de cuadrado y posiciones finales de las neuronas de la red luego del entrenamiento.

3.3. Análisis

Se observa que la red implementada es capaz de aprender considerablemente bien las distribuciones de las figuras 1, 4 y 5, que son el círculo, triángulo y cuadrado respectivamente. La posición de las neuronas es aproximadamente equidistante las unas de las otras, respondiendo a la distribución de datos subyacente. También se pudieron generar los mapas de preservación de topología.

En las formas U y dona (fig. 2 y 3), las neuronas aprendieron la distribución pero hay algunas unidades que quedaron por fuera de las figuras: en la U quedaron 2 neuronas mientras que en la dona, 1. Se considera que este efecto tiene su origen en que esas neuronas nunca ganan, y son movidas por todas las otras neuronas con una "fuerza" inversamente proporcional a la distancia. Por esto mismo hay neuronas en el centro de la dona (todas las neuronas dentro de la dona tiran con la misma fuerza por estar a la misma distancia), y en la U tenemos neuronas más arriba o abajo (las neuronas de los laterales tiran hacia si mismas y contrarrestan los efectos de las de la parte de abajo de la U).

En las otras geometrías esto no sucede.

Para entender esto se puede ver la imagen 6, que tiene algunos de los vectores de fuerza imaginaria en verde. En ella se va a notar con claridad lo enunciado más arriba.

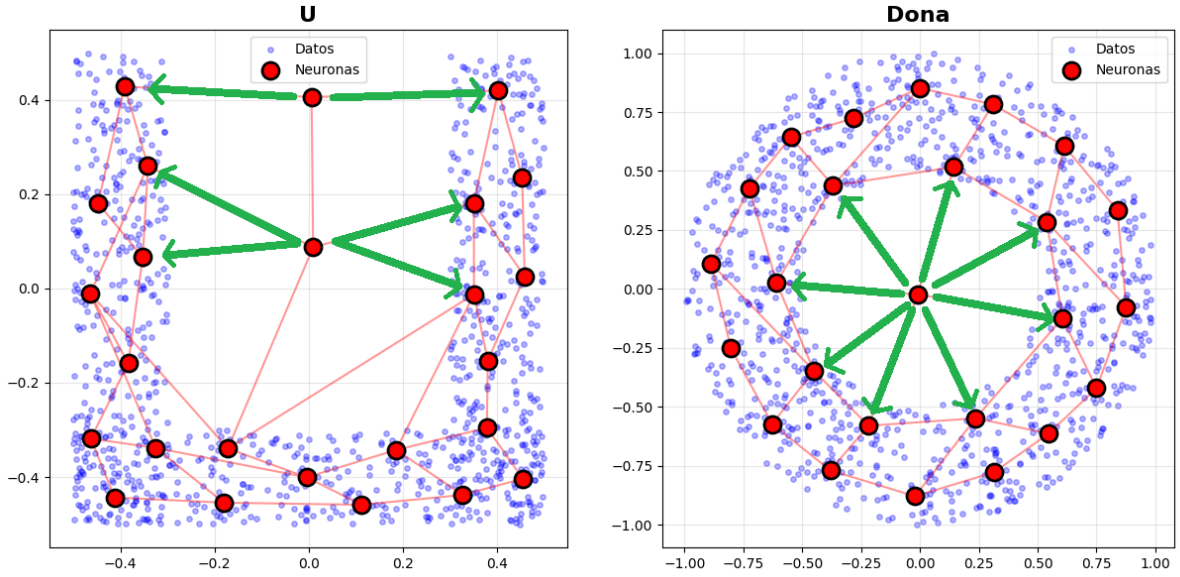

Figura 6

Otro detalle interesante es que, si se observa con detenimiento, las neuronas están sobre los puntos de mayor densidad de muestras, indicando que:

1. El aprendizaje está funcionando porque las neuronas se acercan a donde ganan más, que coincide con las zonas donde hay mayor concentración de muestras.

2. Las distribuciones generadas no fueron perfectamente uniformes: la presencia de cúmulos locales de mayor densidad hace que algunas neuronas se agrupen en esas regiones.

4. Ejercicio 2

4.1. Consignas

Resuelva (aproximadamente) el "Traveling salesman problem" para 200 ciudades con una red de Kohonen.

4.2. Desarrollo

Para este ejercicio se usó un *dataset* de coordenadas (X,Y) de 312 ciudades estadounidenses, a partir del cual se crearon sets de menos muestras para ir mostrando la resolución del problema del "Traveling salesman problem" en orden de menor a mayor complejidad.

Además, se modificó el código de entrenamiento de la red de Kohonen para que la rejilla de neuronas no fuera una grilla 2D sino una línea cuyos extremos están conectados entre sí. Es decir, la topología utilizada fue un anillo. Esta configuración asegura que, la red final sea un camino cerrado, lo cual es consistente con la solución del problema del vendedor viajante, donde la ruta debe iniciar y finalizar en el mismo punto. Aparte, la naturaleza de como se organiza el mapa según cercanía debería resultar en una solución buena del problema (seguramente no óptima).

La Figura 7 muestra la distribución completa de las ciudades del dataset.

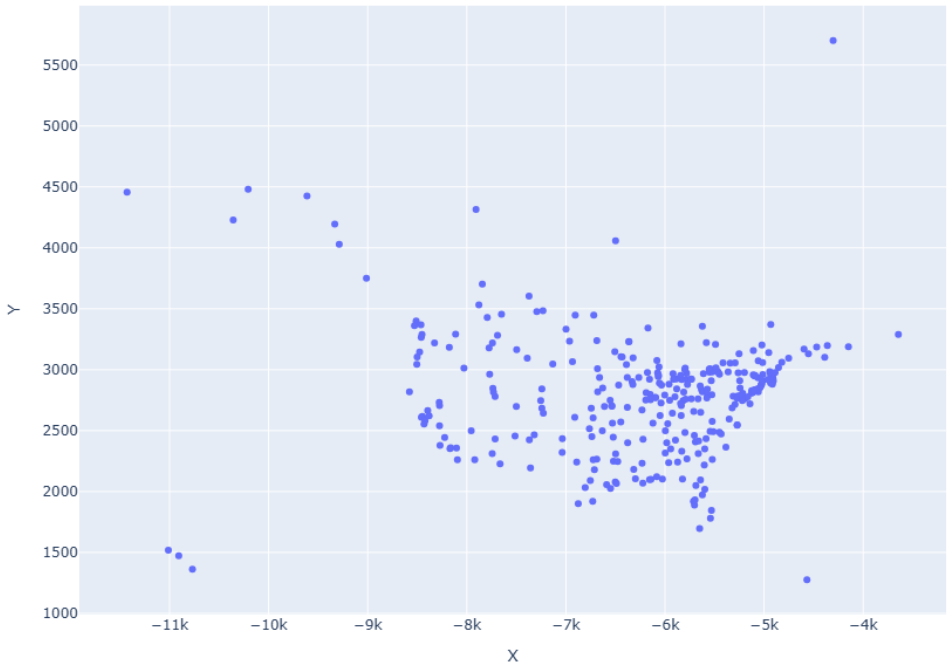

Figura 7: Todas las muestras de ciudades de USA del dataset

Para los mapas de menos ciudades, se usaron 3000 iteraciones con una varianza de 2. Para los mapas más grandes, se usaron 1000 iteraciones y varianza de 8, pero todos con *learning rate* de 0,5. Este ajuste de parámetros fue empírico, y siempre se buscó la solución que pareciera mejor.

Iniciamos con un conjunto de 20 ciudades y una red de 20 neuronas. La Figura 8 muestra únicamente el camino resultante (sin visualizar neuronas ni muestras). Luego, en la Figura 9 puede verse una situación típicamente notada cuando la cantidad de neuronas coincide con la cantidad de ciudades: algunas neuronas quedan exactamente sobre las ciudades, pero otras quedan entre ellas, sin representar ninguna muestra en particular. Es análogo a lo observado en la consigna anterior, donde algunas neuronas quedan fuera de la distribución.

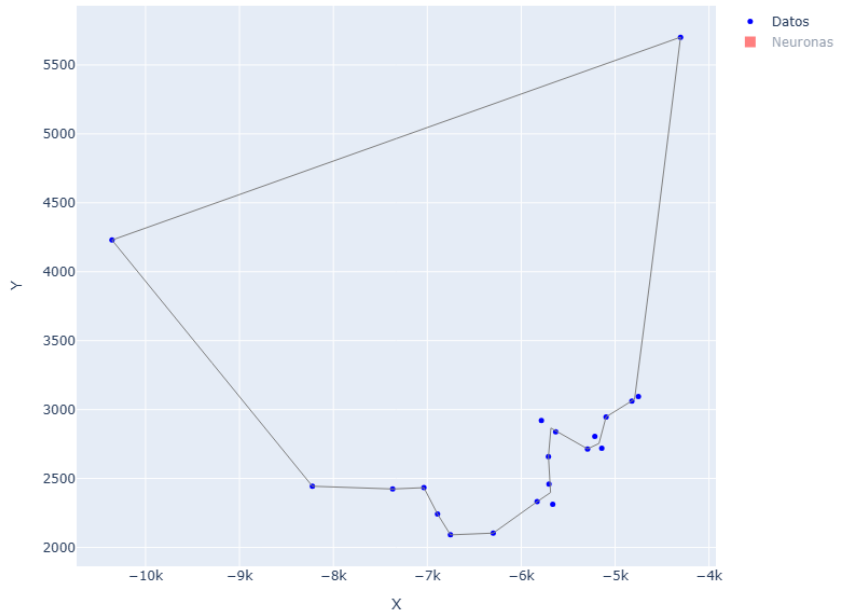

Figura 8: 20 ciudades con 20 neuronas - El camino sin las neuronas o ciudades

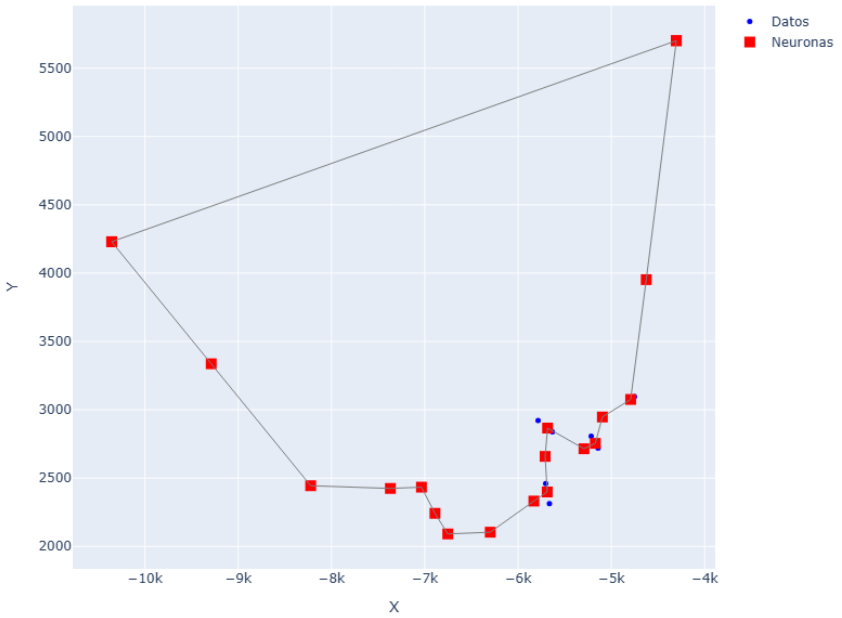

Figura 9: 20 ciudades con 20 neuronas - Pueden aparecer neuronas que quedan entre ciudades

En mi opinión, no se busca que el modelo aprenda a la perfección la distribución, pero estos casos de generalización (una unidad entre 2 muestras) no son deseables cuando el resultado final deseado es un trayecto que pase por todos los puntos. Para corregir este comportamiento, se aumentó la cantidad de neuronas al doble de ciudades, con la intención de que hayan más unidades, logrando más ajuste a los datos y caminos que si pasan por todos los puntos a visitar.

Ahora, con 40 neuronas para 20 ciudades (Figuras 10 y 11) se observa que, al haber neuronas sobrantes, siempre hay unidades representando correctamente las muestras y el trayecto final pasa por todos los puntos. Se notó que las unidades sobrantes se acomodan sobre los trayectos rectos del viaje, como se observó en la consigna anterior con la U o dona. No "entorpecen" (hasta se podrían remover) y, a priori, no parecerían empeorar la solución.

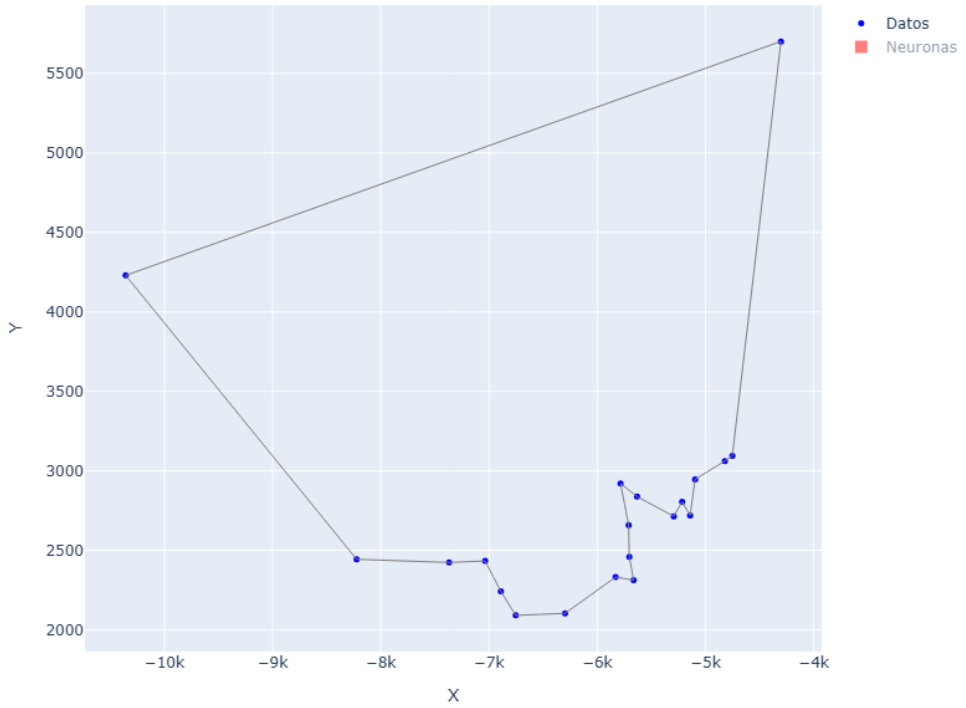

Figura 10: 20 ciudades con 40 neuronas - El camino sin las neuronas

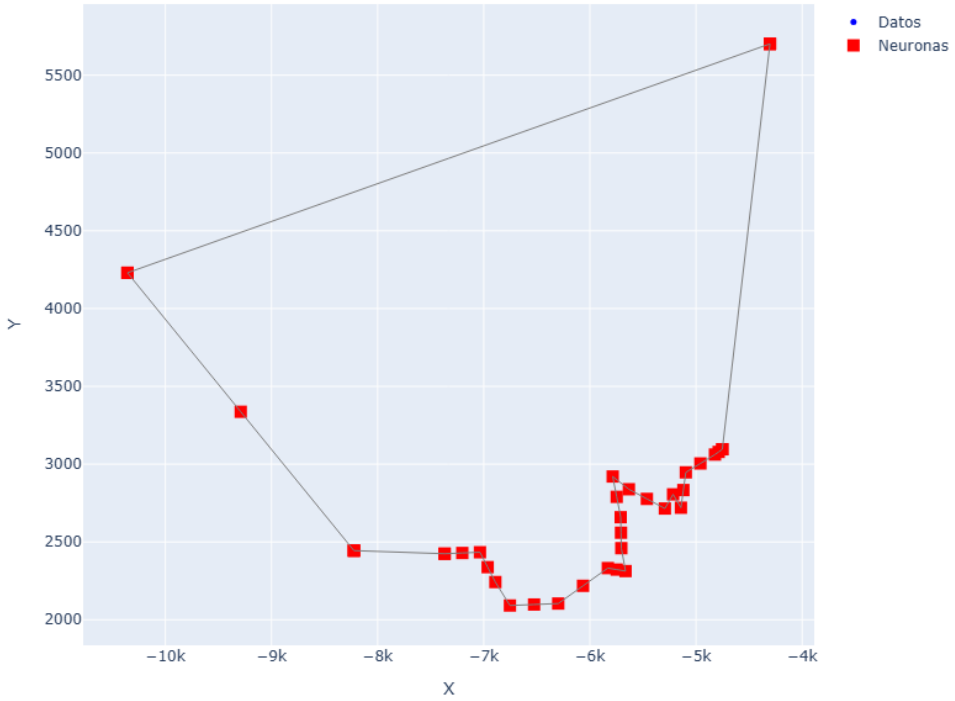

Figura 11: 20 ciudades con 40 neuronas

Pasando a un conjunto mayor, con 50 ciudades y 100 neuronas (Figuras 12 y 13), se observa que las neuronas logran ajustarse a la distribución, aún si quedan algunos puntos intermedios que no corresponden a ninguna ciudad (se notó en el visualizador integrado en el *notebook*). El trayecto resultante pasa por todas las ciudades.

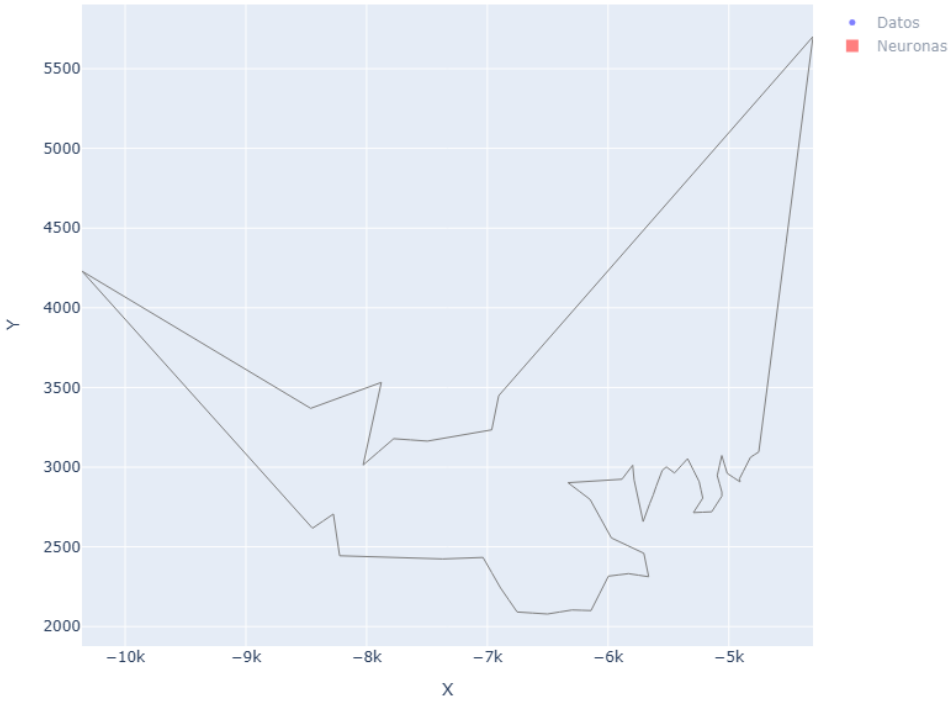

Figura 12: 50 ciudades con 100 neuronas - El camino sin las neuronas o ciudades

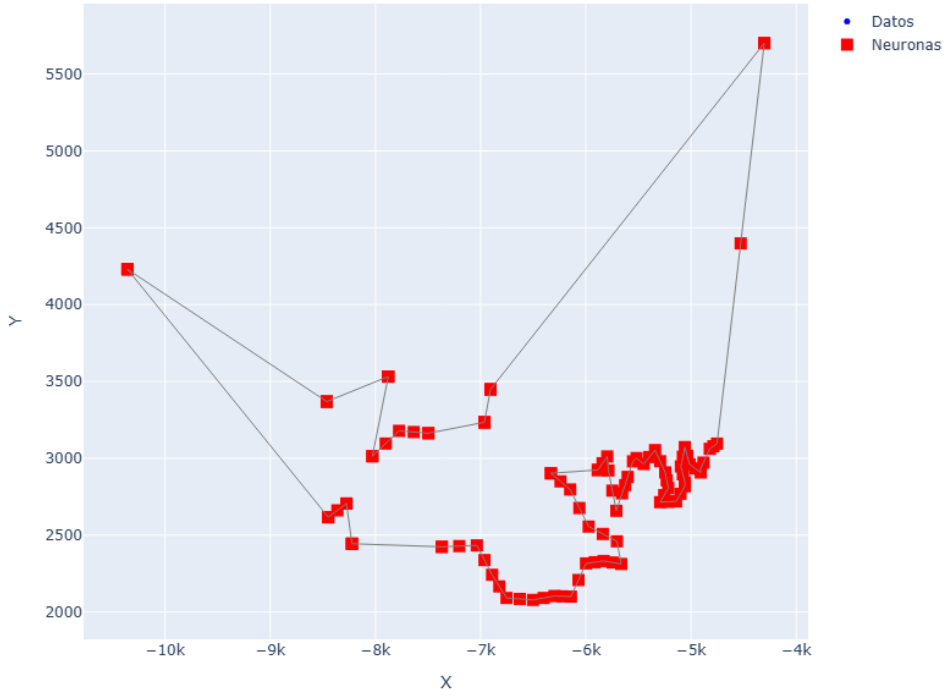

Figura 13: 50 ciudades con 100 neuronas

Para 100 ciudades con 200 neuronas (Figuras 14 y 15), la red continúa manteniendo un comportamiento adecuado, ajustándose al contorno general del mapa y generando un camino continuo y ordenado.

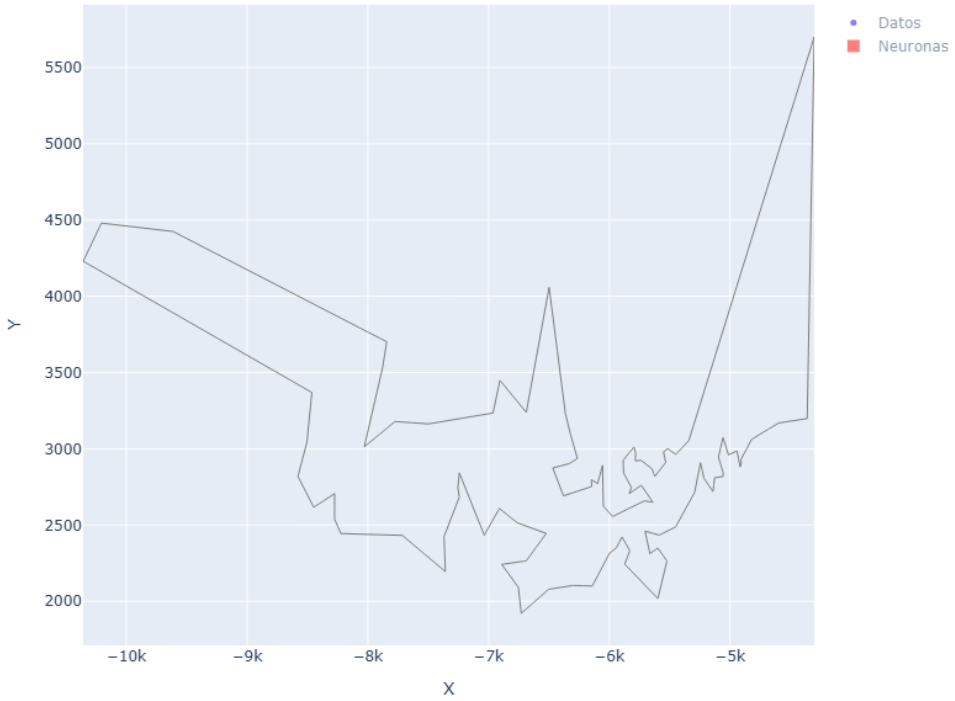

Figura 14: 100 ciudades 200 neuronas - El camino sin las neuronas o ciudades

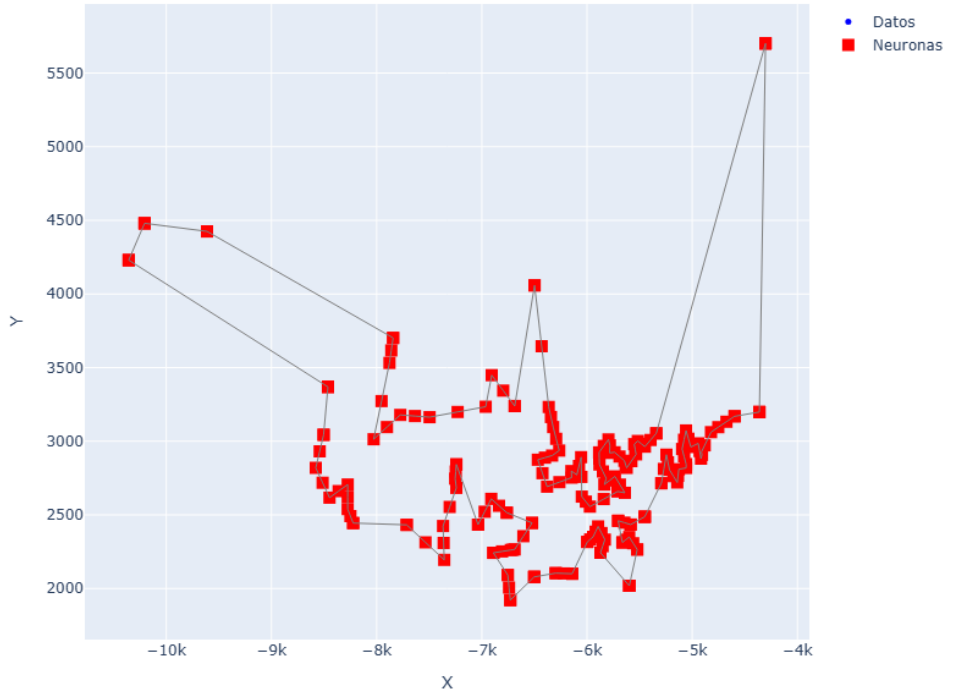

Figura 15: 100 ciudades 200 neuronas

Al aumentar a 200 ciudades y 400 neuronas, los fenómenos de 1 unidad entre 2 ciudades se vuelve más prolifero, como se ve en la figura 19, Lo mismo sucede con el fenómeno de la neurona atrapada entre 2 ciudades, como muestran las figuras 16 y 17.

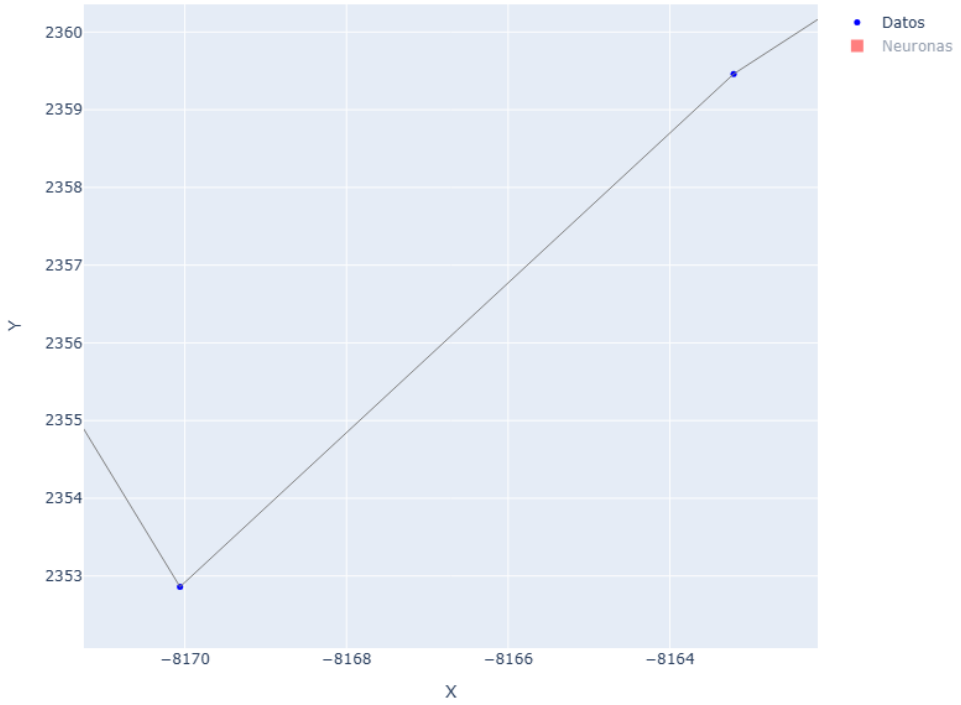

Figura 16: 200 ciudades 400 neuronas - Dos ciudades representadas por tres neuronas, una queda en el medio

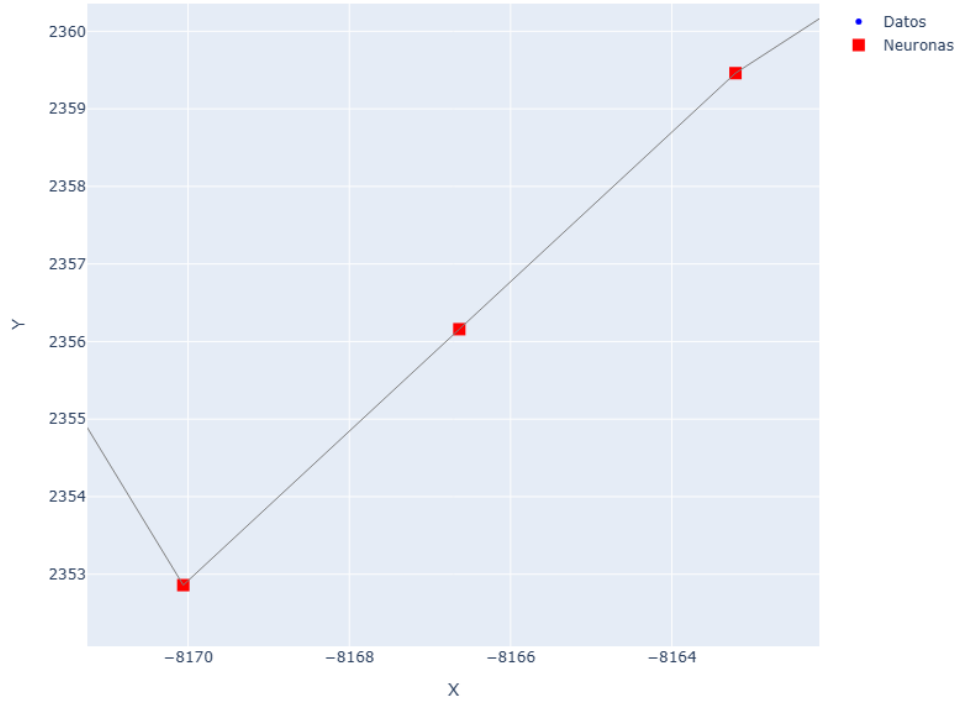

Figura 17: 200 ciudades 400 neuronas - Se observa la neurona intermedia entre dos ciudades cercanas

El camino completo para este caso se muestra en las Figuras 18 y 20. En la imagen 19 se hace *zoom* al camino, y se ve que claramente la solución no pasa por encima de todas las ciudades, lo que técnicamente invalidaría este camino como solución. Sin embargo, como estas ciudades están muy cerca (razón por la cual sucede este fenómeno de que la neurona queda en el medio), la corrección del trayecto es trivial, solo se debe elegir una de las 2 combinaciones que minimice la distancia.

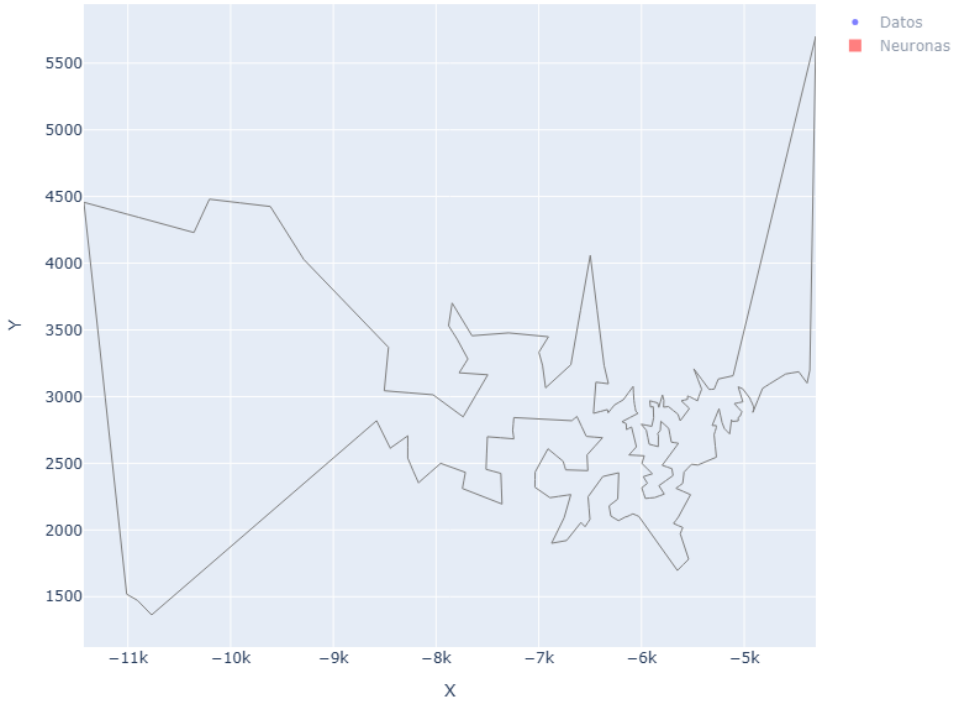

Figura 18: 200 ciudades 400 neuronas - el camino sin las neuronas o ciudades

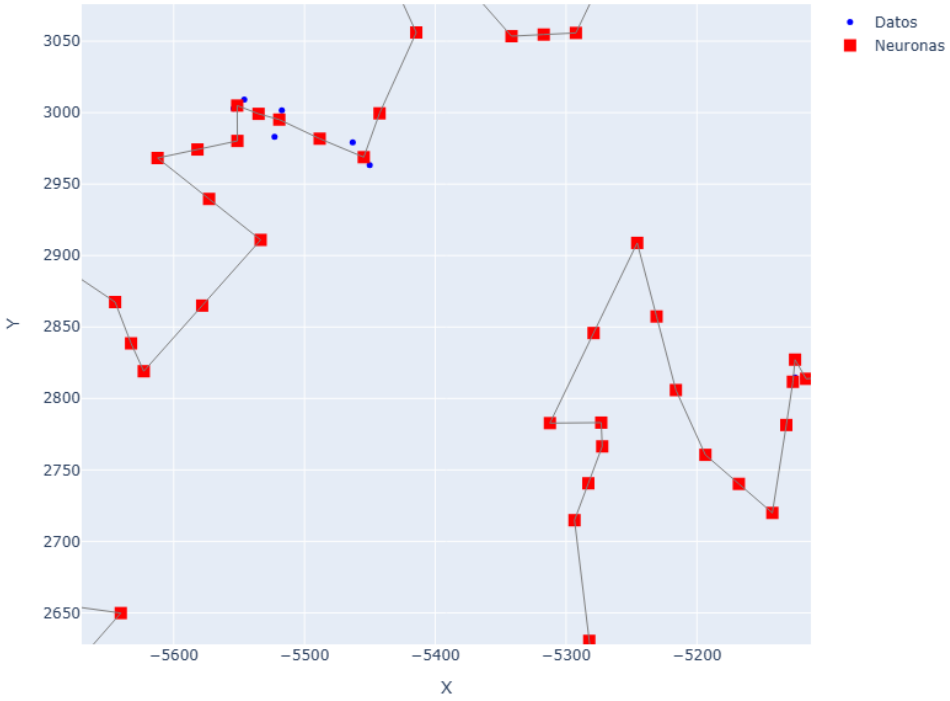

Figura 19: 200 ciudades 400 neuronas - zoom para más detalle

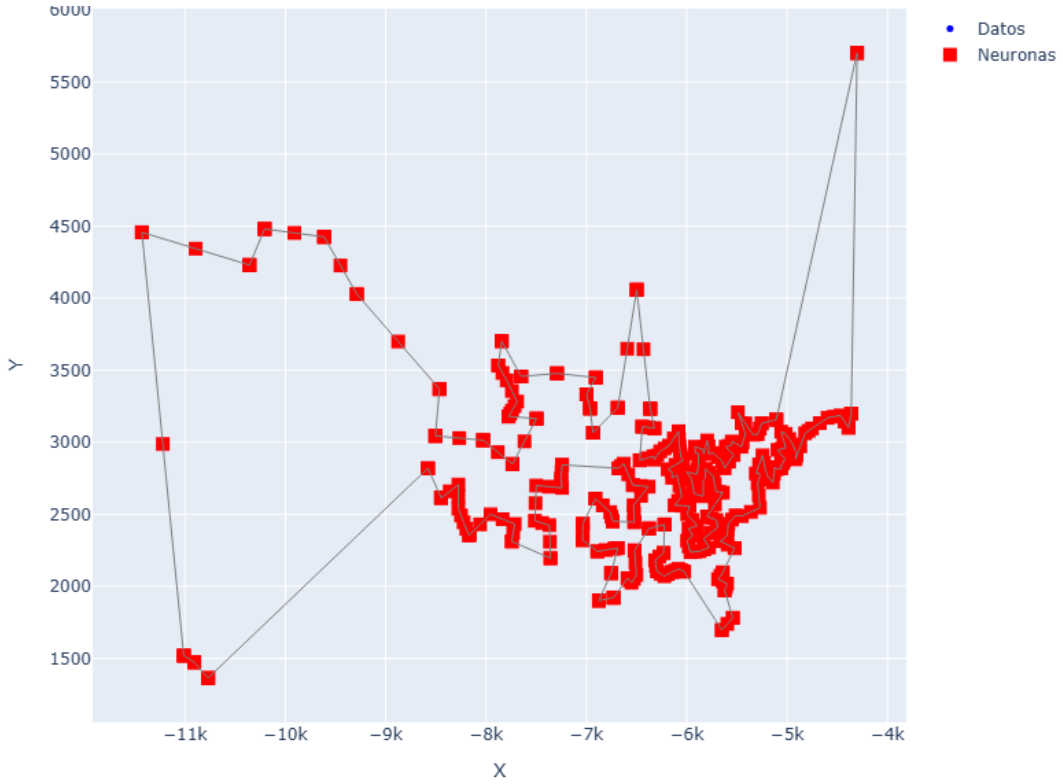

Figura 20: 200 ciudades 400 neuronas

Finalmente, se entrenó el modelo con las 312 ciudades del dataset completo y 624 neuronas.

Las Figuras 21 y 22 muestran el resultado final, donde el mapa neuronal se ajusta al contorno general del país, produciendo una ruta continua que enlaza todas las ciudades.

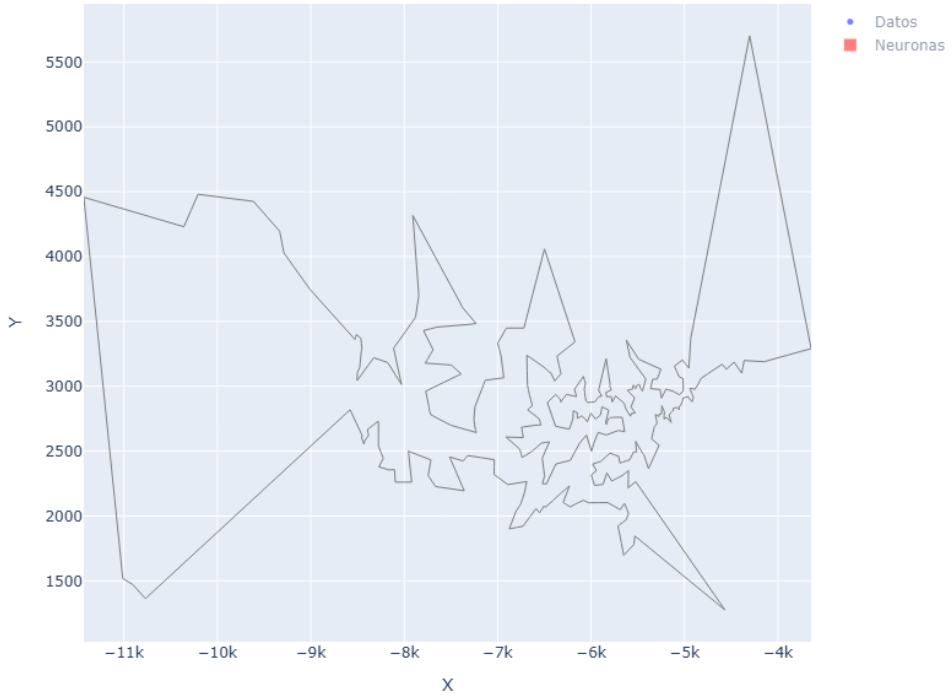

Figura 21: 312 ciudades 624 neuronas - el camino sin las neuronas o ciudades

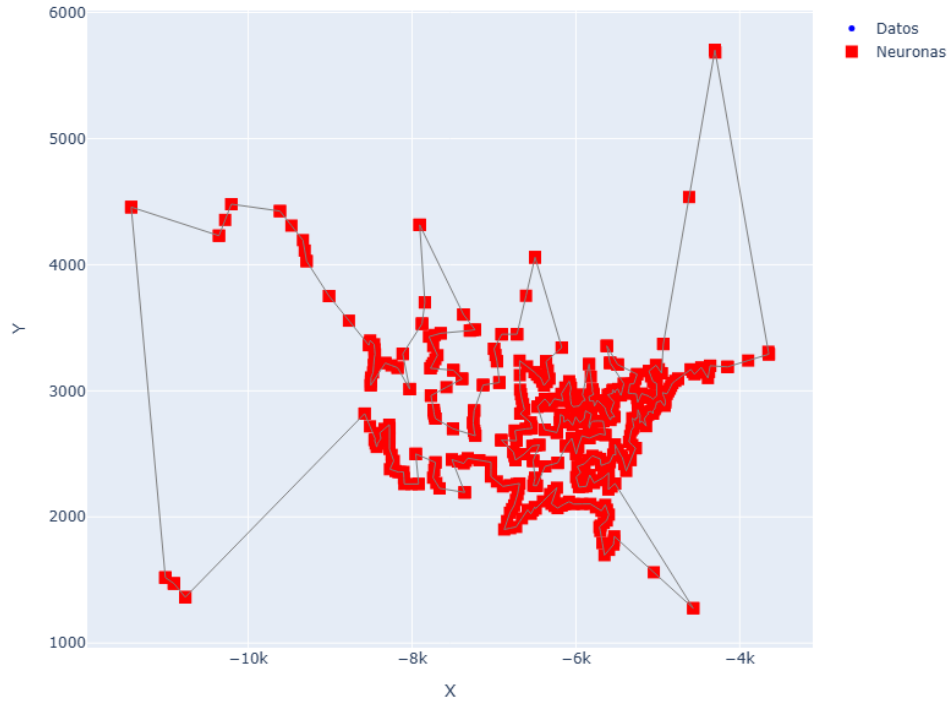

Figura 22: 312 ciudades 624 neuronas

4.3. Análisis

Ante todo, se destaca que se pudo resolver el problema del *travelling salesman* de forma aproximada por medio de redes de Kohonen. El *approach* de que las neuronas dupliquen la cantidad de ciudades ayudó a minimizar la aparición de neuronas que quedan ubicadas entre dos ciudades muy cercanas, a costas de tener más unidades innecesariamente ubicadas en el trayecto.

Estos problemas no invalidan la solución porque la corrección del trayecto es trivial; alcanza con elegir el orden que minimiza la distancia. Pero es importante remarcar que, estrictamente, la solución es aproximada, y, si se fuera a usar de verdad, requeriría de un ajuste manual y verificación.

Aunque el método no garantiza optimalidad, los trayectos generados parecen coherentes, y resultan en un viaje que parecería ser corto

5. Ejercicio 3

En el campus encontrará el archivo "datos_para_clustering.mat" que contiene una matriz de datos de 500 mediciones de una variable de 100 dimensiones.

a) Utilice una red de Kohonen para reducir la dimensionalidad de los datos.

b) Verifique la presencia de clusters, e indique cuantos puede visualizar, haciendo uso de la matriz U.

5.1. Consignas

5.2. Desarrollo

5.3. Análisis

6. Conclusiones

Cada consigna tiene su análisis, así que no es necesario en detalle sobre algún ejercicio particular. Considero que el trabajo sirvió como una introducción extremadamente completa los temas de redes neuronales basadas en perceptrones, métodos de optimización y redes convolucionales. A mi entender, las consignas fueron resueltas de forma satisfactoria, ahondando en algunos temas no requeridos pero que hacen al entendimiento de la materia.