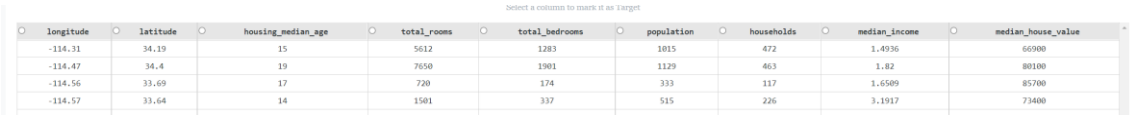
CASO DE ESTUDIO: Predicción de los precios de la vivienda en California.

En este caso de estudio se utilizará mapas autoorganizados(SOM's) para entrenar dos SOM's, uno con topología **rectangular** y el otro con topología **hexagonal**. Se compararán ambos modelos y se analizarán **las relacione**s entre los datos **encontradas** gracias a la naturaleza de este tipo de redes neuronales.

1.Datos utilizados.

Para este estudio se hará uso del dataset "California Housing", que contiene el precio medio de venta de las viviendas de California en el año 1990 según el censo.

Dicho dataset contiene 17.000 filas con datos únicos y 9 características:



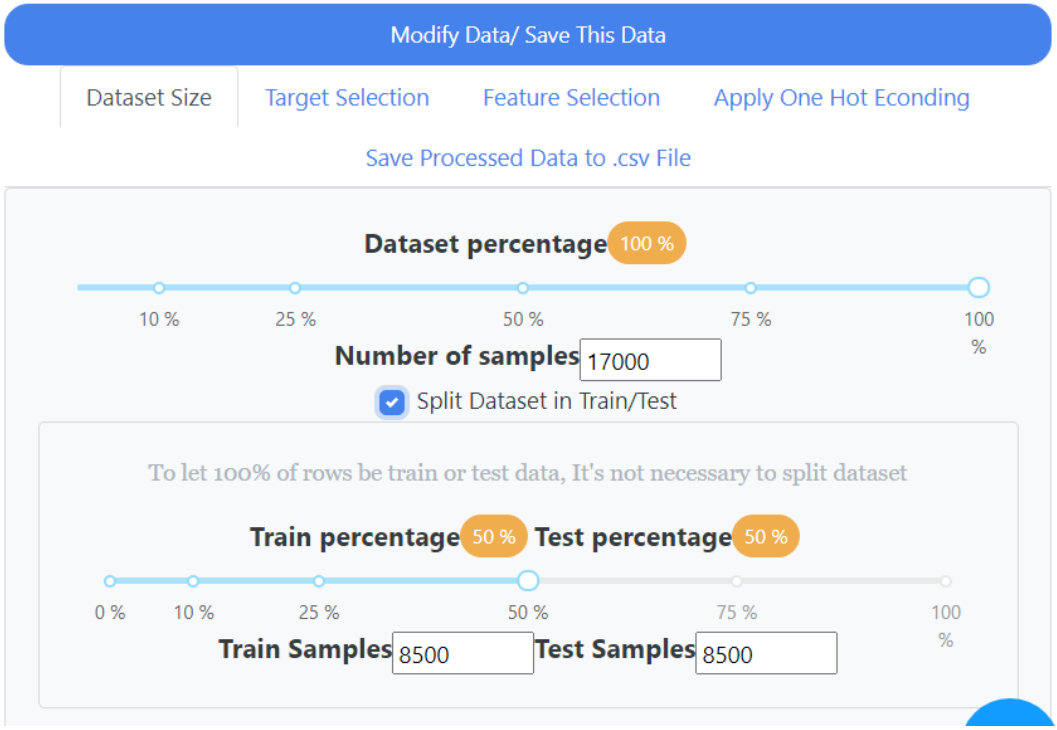
Select a column to mark it as Target

longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
-114.31	34.19	15	5612	1283	1015	472	1.4936	66900
-114.47	34.4	19	7650	1901	1129	463	1.82	80100
-114.56	33.69	17	720	174	333	117	1.6509	85700
-114.57	33.64	14	1501	337	515	226	3.1917	73400

- **Longitude y Latitude:** Coordenadas geográficas de la vivienda.
- **housing_median_age:** Edad promedio de una casa dentro de un bloque
- **total_rooms**: Número total de habitaciones dentro de un bloque de edificios.
- **total_bedrooms:** Número total de dormitorios dentro de un bloque de edificios.
- **population:** Número total de personas viviendo en dicho bloque.
- **Households:** Número total de hogares en el bloque.
- **median_income:** Ingresos medios de las personas que viven en un hogar, por cada bloque(en dólares)
- **median_house_value:** Valor medio de la vivienda dentro de un bloque(en dólares)

2.Procesado de los Datos.

- Se seleccionará como **target** o característica a predecir el **median_house_value.**
- Se **dividirán** los datos (17.000 filas) en dos conjuntos, de entrenamiento y testeo (train y test) **al 50%,** utilizando para ello la aplicación.



3.Búsqueda de hiper-parámetros óptimos: Grid Search

Para poder comprar ambas topologías, rectangular y hexagonal, en igualdad de condiciones, ambas se entrenarán con los mismo hiperparámetros.

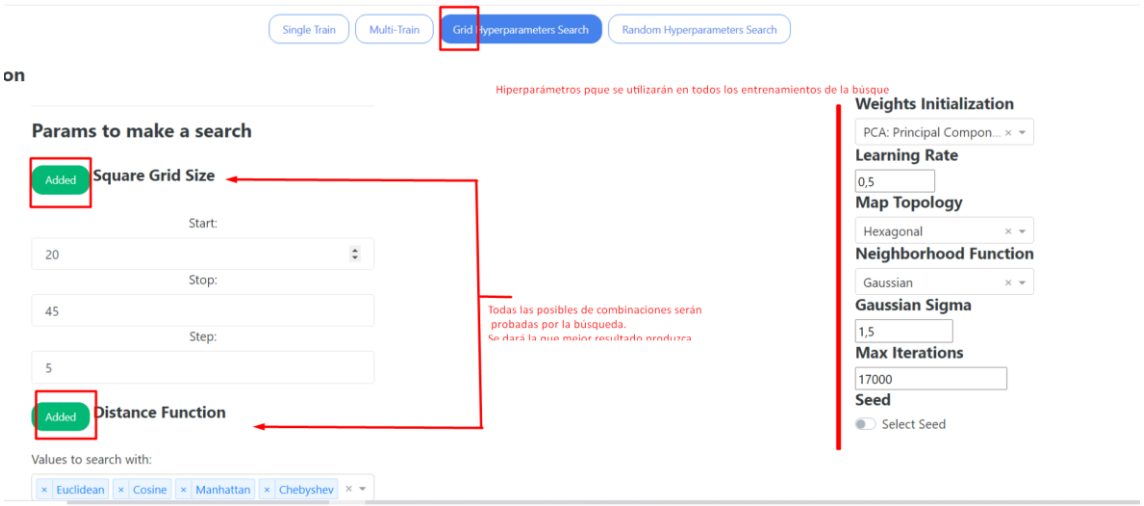
La aplicación implementa 2 algoritmos de afinamiento de hiperparámetros, para entrenarlos con unos valores que produzcan unos resultados óptimos, estos son Grid Search(Búsqueda en Matriz) y Random Search(Búsqueda Aleatoria). Dado que la dimensión de los datos es pequeña(17.000 datos), utilizaremos búsqueda pro fuerza bruta, es decir, probando todas las combinaciones con **Grid Search.**

Dado que la inicialización de los pesos de las neuronas del modelo, de manera empírica, han demostrado producir mejores resultados cuando se emplea **PCA: Principal Component Analisys,** para esta búsqueda no incluiremos este hiperparámetro, que tendrá siempre el mismo valor.

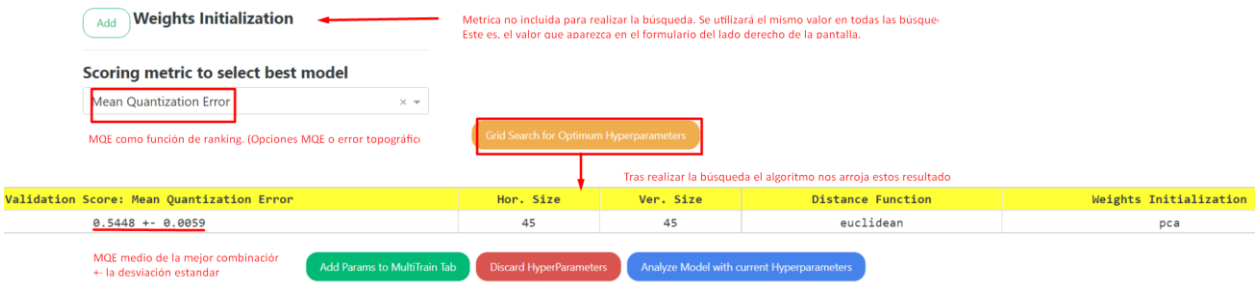
De los 2 posibles hiperparámetros que nos quedan para hacer la búsqueda, escogemos el siguiente rango:

- **Square Grid Size:** Tamaño del SOM a entrenar(Número de NeuronasVverticales x Número de Neuronas Horizontales), con un tamaño de 20 (**20x 20 neuronas)** a un tamaño final a probar de 45(**45 x 45 neuronas),** con saltos de **5** neuronas.Esto nos deja las siguientes combinaciones de tamaños: [20,25,30,35,40,45] a probar.
- **Distance Function:** Función de activación para elegir a la BMU en cada ronda. Elegimos las **4** posible opciones para que la búsqueda las testee: **distancia euclídea, coseno, manhattan y chebysev.**

El resto de hiperparámetros son los elegidos por defecto por la aplicación y que se pueden ver en la captura de debajo:



Tras realizar la búsqueda, utilizando **el Error de Cuantización Medio (MQE) como función de ranki**ng, para elegir como mejor combinación el modelo que de un **resultado más bajo**

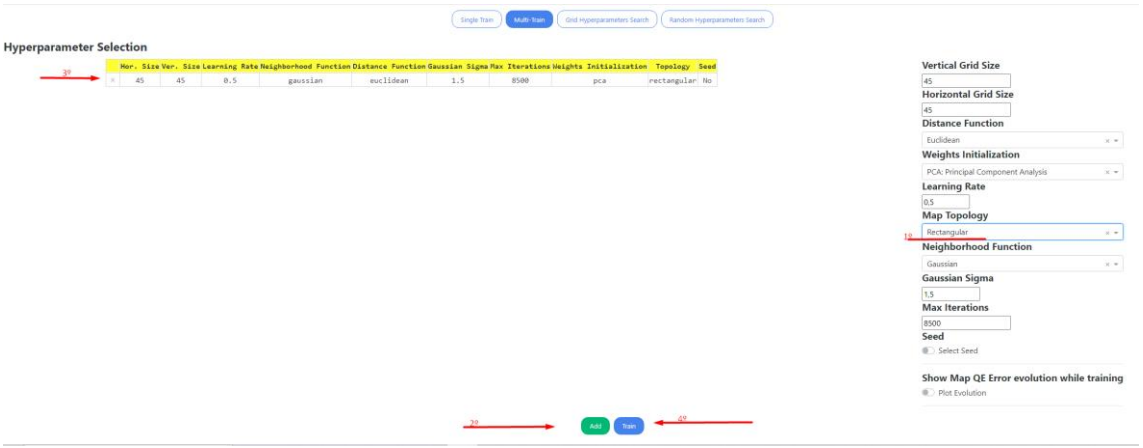


La búsqueda nos arroja **unos hiperparámetros óptimos de 45 neuronas** para el tamaño(vertical y horizontal) **y distancia euclídea** como función de distancia.

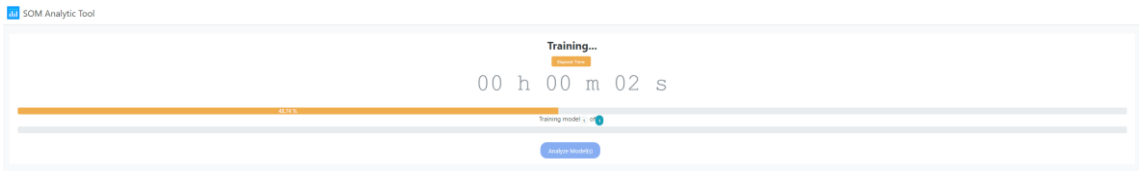
La aplicación nos da ahora <u>3 opciones (3 botones de colores):</u>

1. *Añadir dichos parámetros al TAB: Multi-Train* Dado que queremos comparar estos hiperparámetros con una topología rectangular para hacer la comparativa elegimos esta opción. Nuestro modelo aparecerá en la siguiente ventana de análisis de datos, tras realizar un entrenamiento desde la pestaña de Multi-Train.
2. *Descartar dichos resultados.* Si hemos elegido unos rangos de búsqueda para los hiperparámetros y estos nos han dado un MQE medio muy alto, tal vez queramos descartarlos y realizar otra búsqueda cambiándolos.
3. *Analizar los datos del modelo directamente.* Dado que se hace un re-entrenamiento del modelo con los mejores parámetros antes de terminar la búsqueda, podemos pasar a la ventana de análisis de datos directamente, para analizar el modelo con dichos hiperparámetros.

En nuestro caso, como queremos una comparativa con un modelo con otros hiperparámetros(topología rectangular), elegimos **el primer botón:Add Params to MultiTrain Tab(Color verde),** y en lo alto de la interfaz, seleccionamos la Tab de Multi-Train.



Por defecto nos aparecen los hiperparámetros ya encontrado en la busqueda. Cambiamos la topología, lo añadimos a la tabla de entrenamiento con el botón **Add** y luego pulsamos el botón de **Train**.



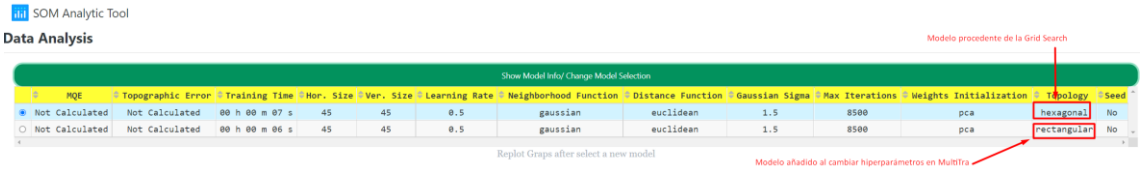
Tras finalizar el entrenamiento analizamos ambos modelos.

NOTA: En la terminal que ejecuta dicha aplicación se puede ver el proceso en el que se encuentra la aplicación, útil para datasets grandes que requieren un tiempo de dibujo de los modelos de unos minutos. De esto modo podemos comprobar que la aplicación no se ha quedado congelada.

```
 -->Shuffling Data...
 -->Shuffling Complete.
 -->Standardizing Train Data...
 -->Standardizing Complete.
-->Training SOM  0 ...
-->Training Complete!
        Elapsed Time: 4.7293455600738525 seconds
```

4.Modelos entrenados. Análisis.

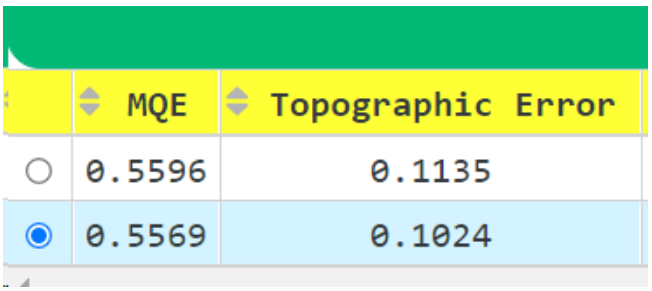
En la página de análisis de modelos, si pulsamos el botón “Show Model Info/Change Model Selection”, podemos ver una lista de todos los entrenamientos disponibles. Podemos cambiar la selección de ellos en el radio que aparece al principio de la tabla.



En nuestro caso, nos encontramos con el modelo hexagonal con el que hicimos la búsqueda(Grid Search) de parámetros óptimos, así como el modelo añadido en MultiTrain. Podemos diferenciarlos porque escogimos en primero con topología hexagonal, y el segundo con rectangular.

4.1.Estadísticas

Calculamos el MQE y el erro Topográfico para ambos modelos:



Observamos como ambos modelos consiguen resultados similares.

En ambos casos un **MQE** más bien **pequeño**, que quiere decir que los **datos están bien representados** de media por cada neurona, al haber sido entrenado el modelo con **una cantidad suficientemente representativa de variedad de ejemplos.**

Por otro lado vemos como el **error topográfico** también es **pequeño**. Al no ser 0, podemos afirmar que **la topología no ha sido respetada para todos los ejemplos**, es decir, la relación

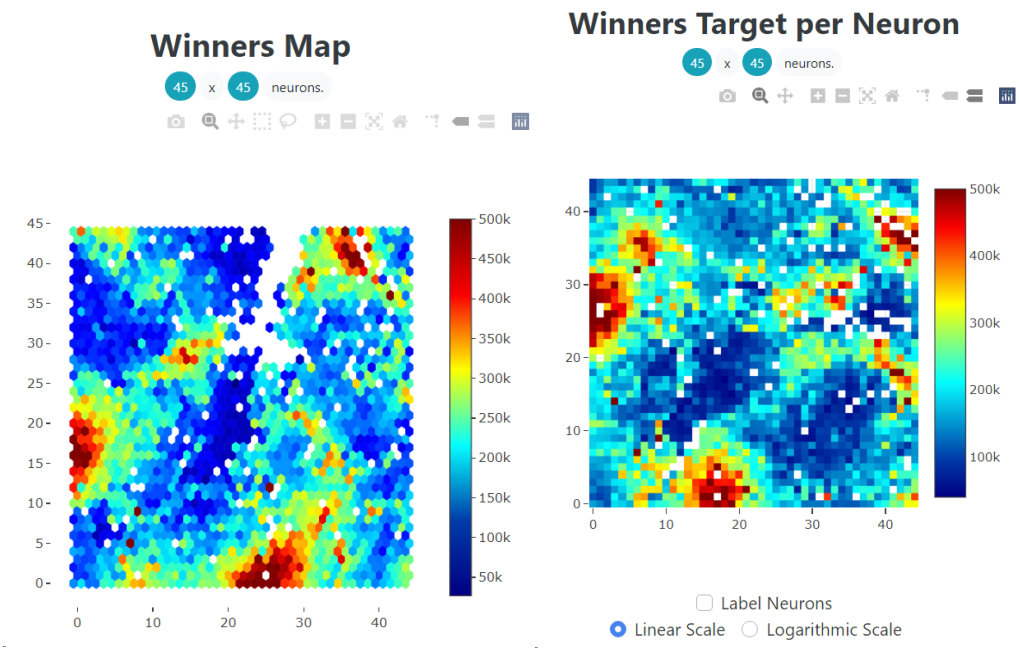
conjunto de características similares – zona en el mapa, no está respetada para todos los datos. Sin embargo, a pesar de poder tener un valor más pequeño, se considera **aceptable**.

4.2.Mapa de Neuronas Ganadoras.

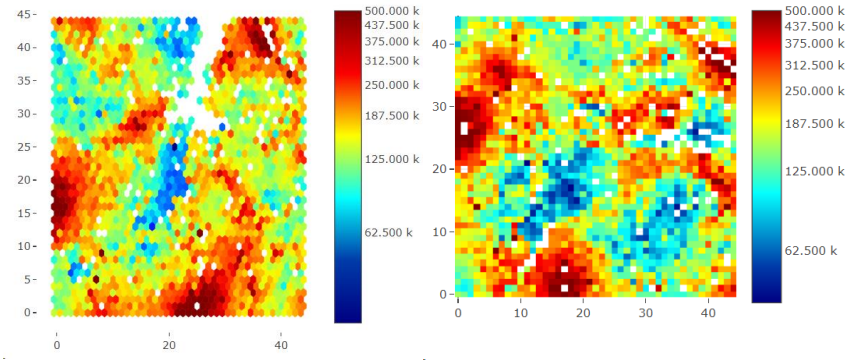
El mapa de neuronas ganadoras se calcula con el target(clase a predecir). Para ello se cogen los datos de entrada, y para cada uno de ellos se calcula su BMU. Al final del proceso, cada neurona tendrá varios datos (o ninguno) de la cual ella es su BMU. Si una neurona no tiene ningún dato, será pintada en color blanco en la aplicación.

Un modelo bien entrenado, organizará targets similares o que guarden relación entre sí en posiciones cercanas, y targets distintos en posiciones lejanas. Una BMU representará solo datos similares, y, por tanto, muy probablemente un único target.

Dibujamos el mapa de neuronas ganadoras. A la **izquierda** el modelo con neuronas **hexagonales**, a la **derecha** con neuronas **cuadradas**.



Mapas de neuronas ganadoras: Escala Lineal

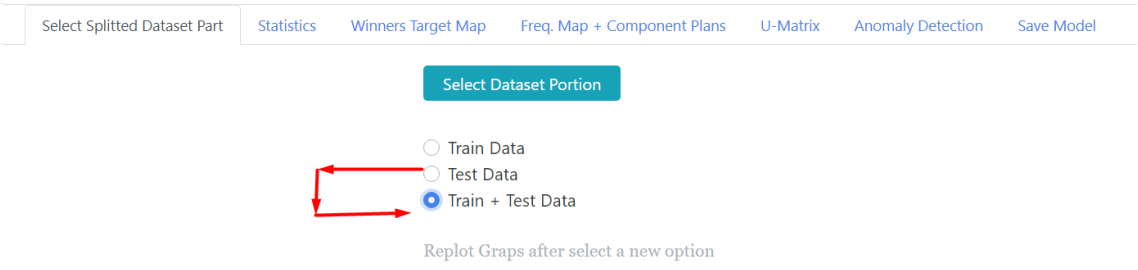


Mapas de neuronas ganadoras: Escala Logarítmica

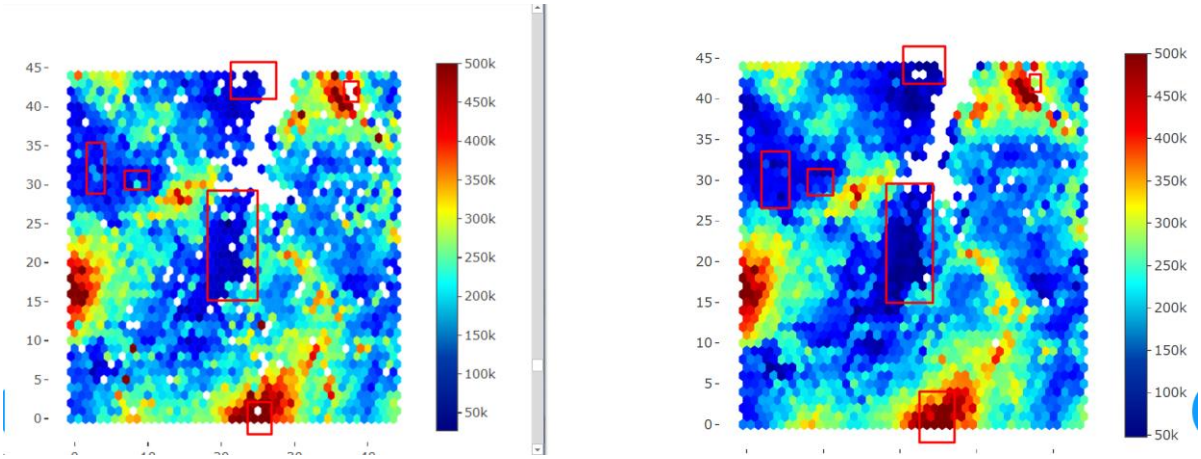
Como podemos observar, ambas toplogías siguen los mismo patrones espaciales, dado que han sido entrenadas con los mismo hiperparámetros. **Sin embargo, la visualización en el grid hexagonal resulta en una precepción más definida y natural, como si de una imagen con más resolución en pixeles se tratara.**

Dado que el dataset está dividido, podemos **escoger los conjuntos de Train y de Test para pintar el mapa** de neuronas ganadoras: con más ejemplos las zonas quedarán visualmente más representadas, y probablemente espacios en blancos sean pintados, como resultado de que al menos un dato del conjunto haya sido elegido BMU de esa neurona.

NOTA: Cambiar los conjuntos de datos para las gráficas no cambia el modelo, solo la representación con más o menos precisión de los datos en el análisis.



Redibujamos la gráfica(pro motivos de espacio en este ejemplo solo la hexagonal) con el botón de “Plot”:



Izquierda: Usando solo datos del conjunto de entrenamiento/

Derecha/Usando: todo el dataset(conjunto entrenamiento y testeo)

Podemos observar ligeras diferencias, no muchas, dado que los 8500 datos añadidos para la graficación son los empleados en el entrenamiento. Aún así podemos observar **neuronas que**

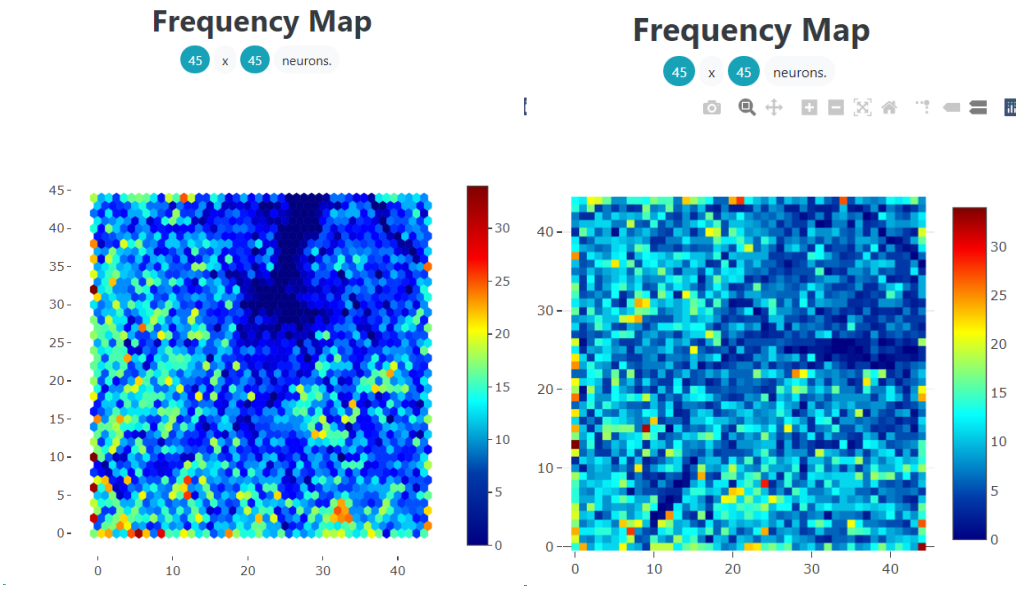
no tenían una clase a representar por falta de ejemplos, y no de pesos, ya que todas las neuronas tienen pesos que guardan relación con su vecindario y ante un dataset aún más completos, es muy probable que acaben representando a datos intermedios de los de sus vecinas.

También podemos observar **zonas en los que las clases intensifican su valor/color**. Véase en rectángulo mayor central mas grande y el cuadrado pequeño que se le sitúa encima.

Se puede observar como esas neuronas representan viviendas de poco valor (en torno a los 50-100k) Con un dataset más amplio, como el usado en la derecha, se puede observar que eso es debido a que las neuronas centrales concentran los valores más bajos 50k, y sus vecinas van subiendo ligera y gradualmente el valor al que representan(100k las más próximas y 150k las más lejanas.) **Es decir, esa zona del mapa se ha especializado en representar las viviendas de bajo valor,** que si bien no es la única, es la que concentra a un vecindario de representantes mayor.

4.3.Mapa de frecuencias.

Dicho mapa representa el número de veces que ha sido elegido una neurona como BMU de un dato de entrada. El mapa de neuronas ganadoras es un mapa de frecuencias que después tiene en cuenta el target de los datos.



El mapa de frecuencias nos da **una medida de fiabilidad en la representación de los datos** de varias maneras.

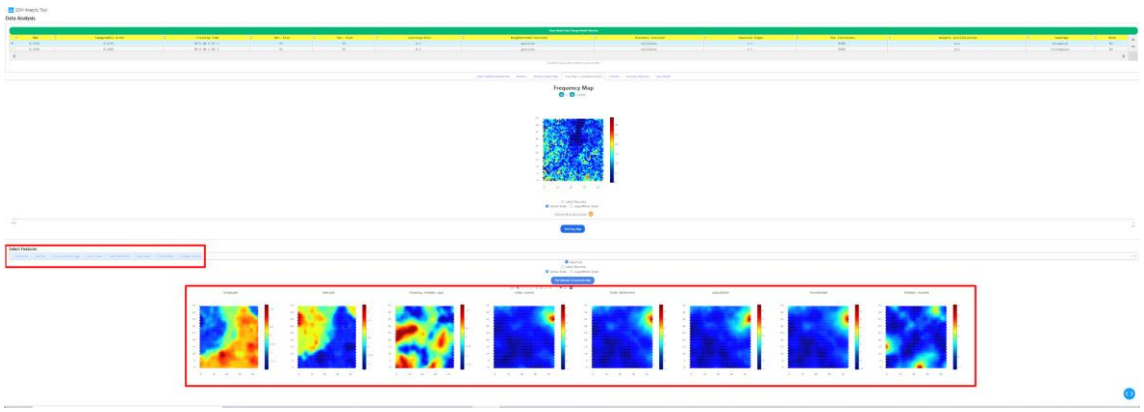
Una de ellas es, en el caso de las neuronas que no han sido elegidas ninguna vez como BMU(Frecuencia 0). Si estas neuronas son casos puntales, es probable que , a pesar de que el mapa represente bien los datos y tenga un bajo mqe y error tipográfico, sea debido a que harían falta más ejemplos para apreciar mejor los gradientes de relaciones en el mapa. Sin embargo si esas neuronas forman grupos de neuronas en el mapa, y no es por falta de ejemplos, el modelo contará con un alto mqe y error topográfico. Esto se traduce en que la topología no se preserva y que los datos no están bien representados, ya sea por falta de ejemplos suficientes o por una mala elección de los hiperparámetros durante el entrenamiento.

En el caso de neuronas aisladas con un número de 'hit' o de frecuencia muy alto, con neuronas contiguas, cuya frecuencia sea considerablemente mucho más baja, es síntoma de mala representación de los datos. En este caso dicha neurona representa muchos ejemplos, que deberían ser repartidos por más neuronas en un mapa más grande. Este fenómeno se puede apreciar en la U-Matrix de modo paralelo, donde se puede apreciar en ese mapa una zona muy oscura en las mismas coordenadas de dicha neurona. Síntoma de que los pesos de la neurona errática y sus vecinas son muy distintos y hacen un salto abrupto y no gradual en el mapa.

Neuronas contiguas con un número de frecuencia muy alto pero que forman regiones en el mapa , se puede traducir o en que el número de ejemplos a representar son muchos y muy parecidos, o de nuevo a una mala elección del el tamaño del mapa durante el entrenamiento, entre otros hiperparámetros.

4.4.Mapa de componentes.

Los mapas de componentes representan para cada neurona, los pesos del modelo entrenado para nada atributo. En nuestro caso el modelo ha sido entrenado con 8 de los 9 atributos originales del dataset, y es por ello por lo que vemos 8 mapas de componentes.

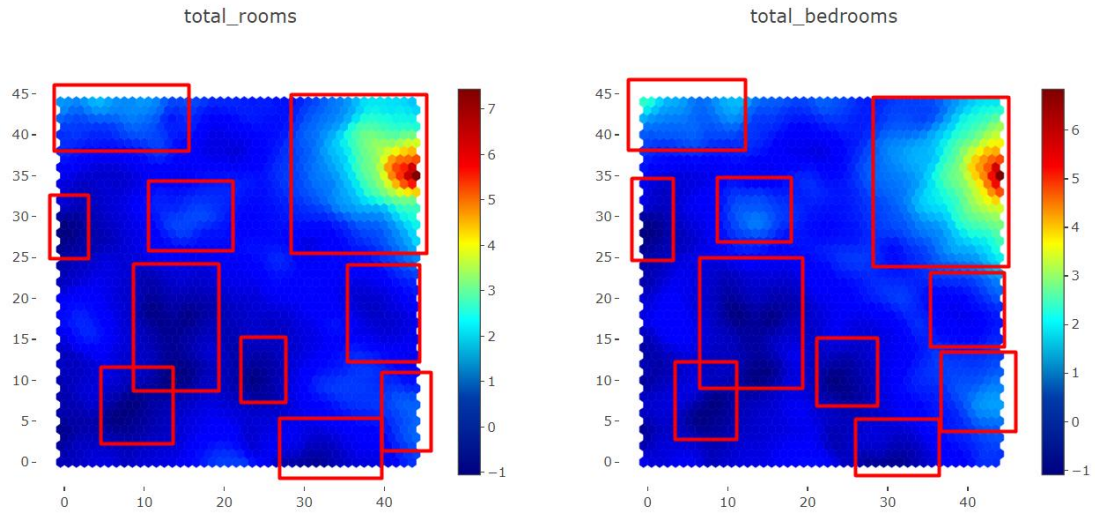


Es decir, cada neurona del modelo entrando es un vector de dimensión número de características.

Los mapas de componentes sirven para descubrir visualmente relaciones y patrones entre atributos y atributos-y clase a predecir.

Veámoslo con 3 ejemplos.

Si observamos los atributos total bedrooms y total rooms:

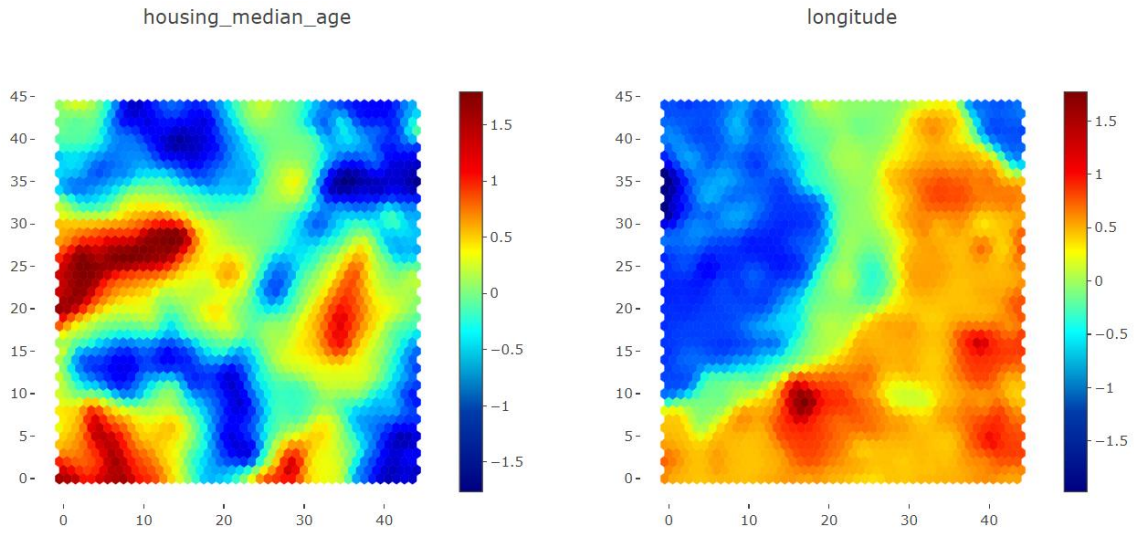


Podemos observar claramente como cualquier cambio de valor en una de las características se refleja de manera prácticamente igual en todos los aspectos. Es decir, **tienen un muchos patrones de comportamiento comunes.** En este caso de estudio concreto, dado que conocemos el significado de los atributos, sabemos que implícitamente uno implica al otro, y que la relación es de proporcionalidad directa.

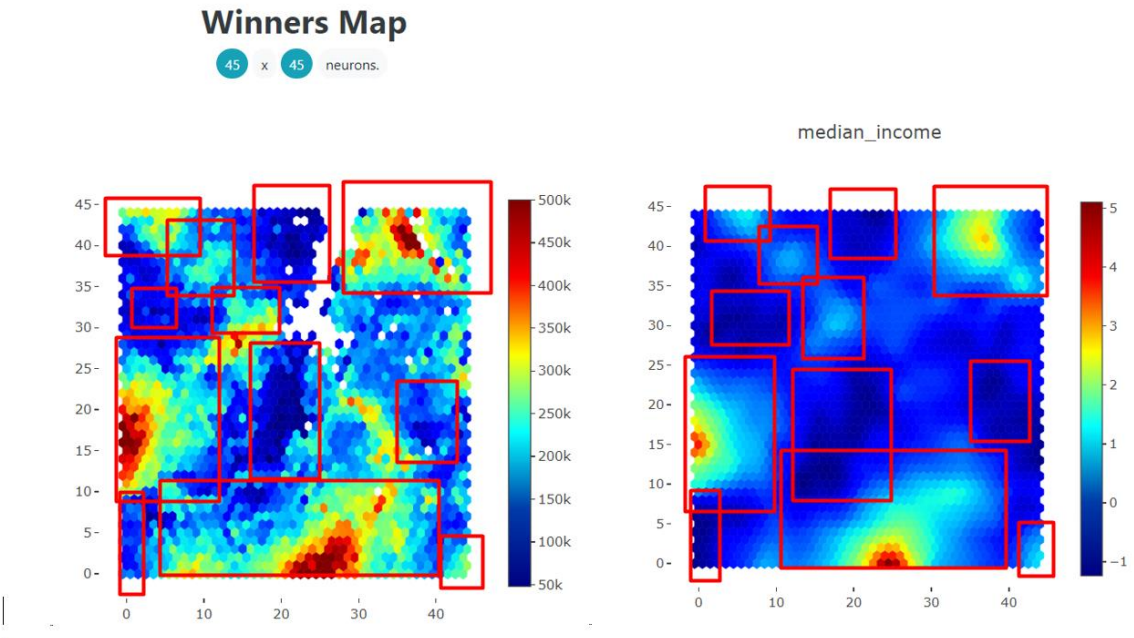
Esto **nos da mucha información**, para empezar, dado la relación directa de este ejemplo, podríamos eliminar uno de los atributos del modelo al ser redundante, y así obtener un modelo más pequeño, ahorrando computacionalmente memoria y tiempo de computación.

Podría darse el caso de que la suma conjunta de varios patrones en varias características se tradujeran en una tercera característica. Es decir, **encontraríamos patrones de comportamiento**. Esto podría compararse de manera conjunta con el mapa de neuronas ganadoras y determinar patrones entre mapas de componentes y este último.

Con atributos que esperamos que no guarden en la vida real ningún tipo de relación, vemos como se traduce de manera muy gráfica en los mapas de componentes, como es el ejemplo de la antigüedad media de una casa y la longitud de sus coordenadas geográficas.



Por último, podemos comparar el mapa de neuronas ganadoras con un mapa de componentes:



Como se ven en la imagen, el conjunto de patrones se traduce de la misma manera del mapa de componentes al mapa de neuronas ganadoras. En la imagen se han destacado solo los principales. Sin embargo, vemos como están completamente relacionados, como esperaríamos en la vida real, puesto que la renta de las familias incide directamente en el precio que pagan por sus casas. Regiones que concentran rentas más altas, tienen viviendas más caras. Lo mismo sucede con las rentas y precio de viviendas bajos y medios**. De este dataset, podemos concluir que el componente determinante de elección de precio de la vivienda es la renta de las familias.** De este modo, si no hubiésemos sabido por la naturaleza

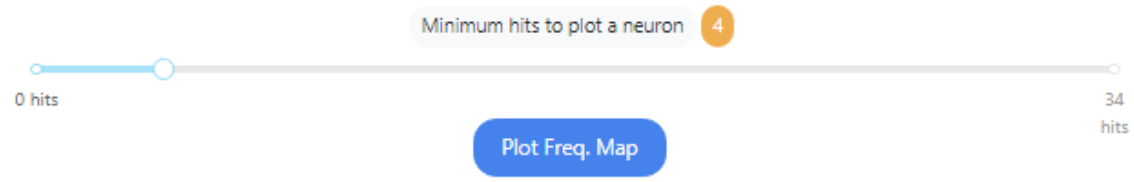
de los datos la relación entra ambas características, habríamos **descubierto patrones de comportamiento comunes, además con una relación directa.**

En datos más complejos, como por ejemplo datos médicos, podríamos derivar conclusiones de patrones de comportamiento entre varias características y la clase a predecir, que tal vez de otro modo no hubiéramos apreciado.

4.5. Mapa de frecuencias como filtro de calidad en los Mapas de componentes.

Como vimos en el punto 4.3 , a veces ciertas neuronas no representan a ningún dato, ya que no han sido elegidas nunca como BMU de ningún vector de entrada. Como también vimos en el punto 4.4, los mapas de componentes nos ayudan a ver comportamientos, patrones y anomalías comunes a las características de los datos.

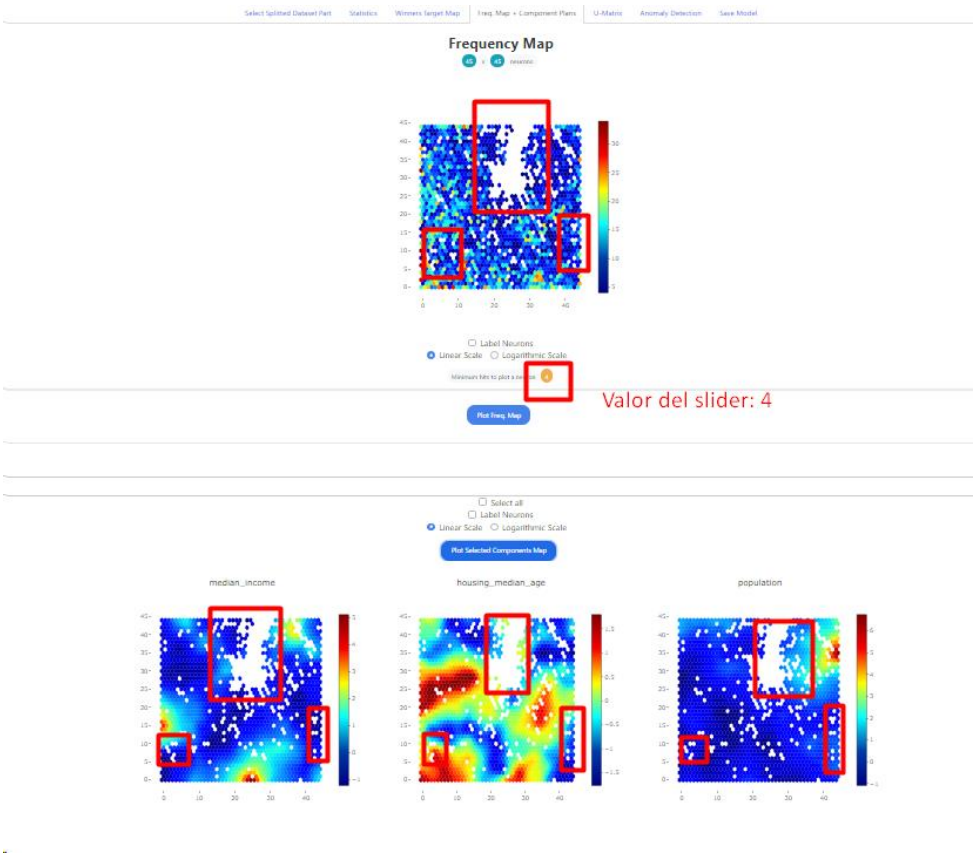
Es por ello, que debajo del mapa de frecuencias se encuentra un slider, que nos permite seleccionar el número mínimo de veces que una neurona ha tenido que ser elegida BMU. Esto nos garantiza que los datos pintados representen a datos reales, y que lo hagan con un número mínimo de ejemplos para asegurarnos de que representan a datos reales de un dataset y no a anomalías puntuales.



Al utilizar dicho slider, eliminamos de las gráficas las neuronas que no tengan un mínimo numero de frecuencia, y esto se hace tanto en el mapa de frecuencias como en todos los mapas de componentes dibujados.

De este modo, **validamos** de algún modo, que **los patrones que visualicemos no se deban al azar ni a datos puntuales, si no que están respaldados por datos reales.**

En el ejemplo anterior, si aplicamos un mínimo de 4, nos queda el siguiente mapa de frecuencias y los siguientes mapas de componentes elegidos:

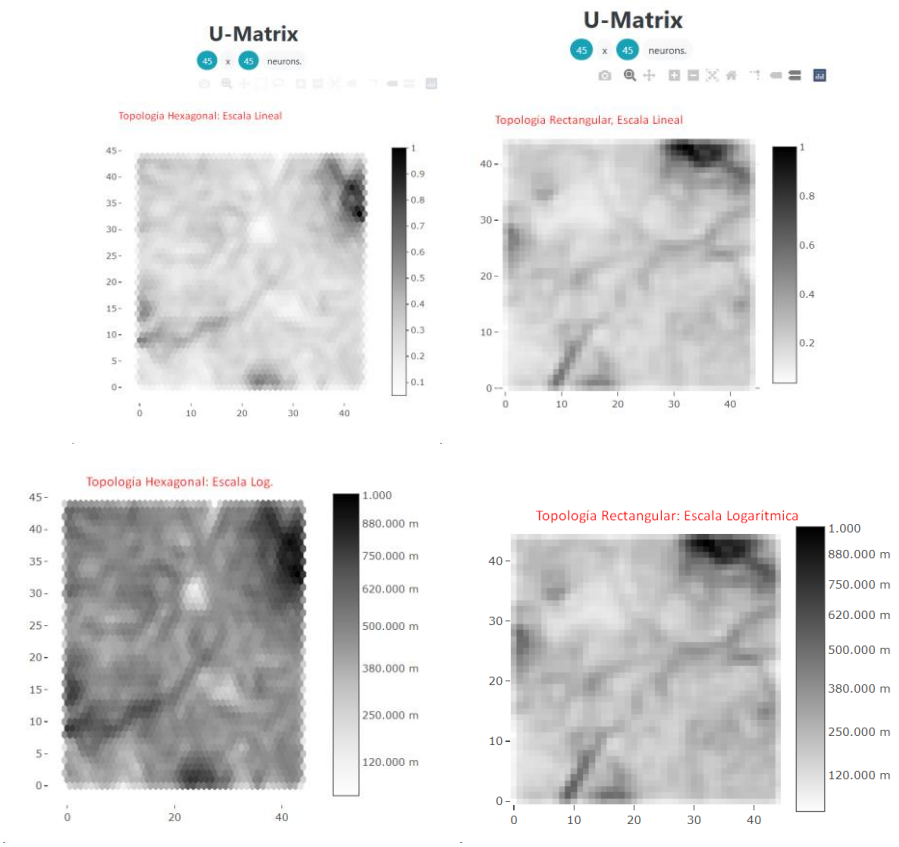


Donde todos ellos dejan de dibujar las mismas neuronas.

4.6. U-Matriz: Matriz de distancias unificadas

La matriz de distancias unificadas nos proporciona diversa información.

En primera instancia, nos muestra la “diferencia abrupta entre los pesos de neuronas contiguas”. Esto es, neuronas cuyos vectores de pesos estén mucho más lejos que del resto de vecindario, tendrán valores muy oscuros en la Matriz-U. Esto quiere decir que **si no se preserva la topología del mapa,** como comentábamos en el punto 4.3, **esto se verá reflejado en la Matriz-U.**



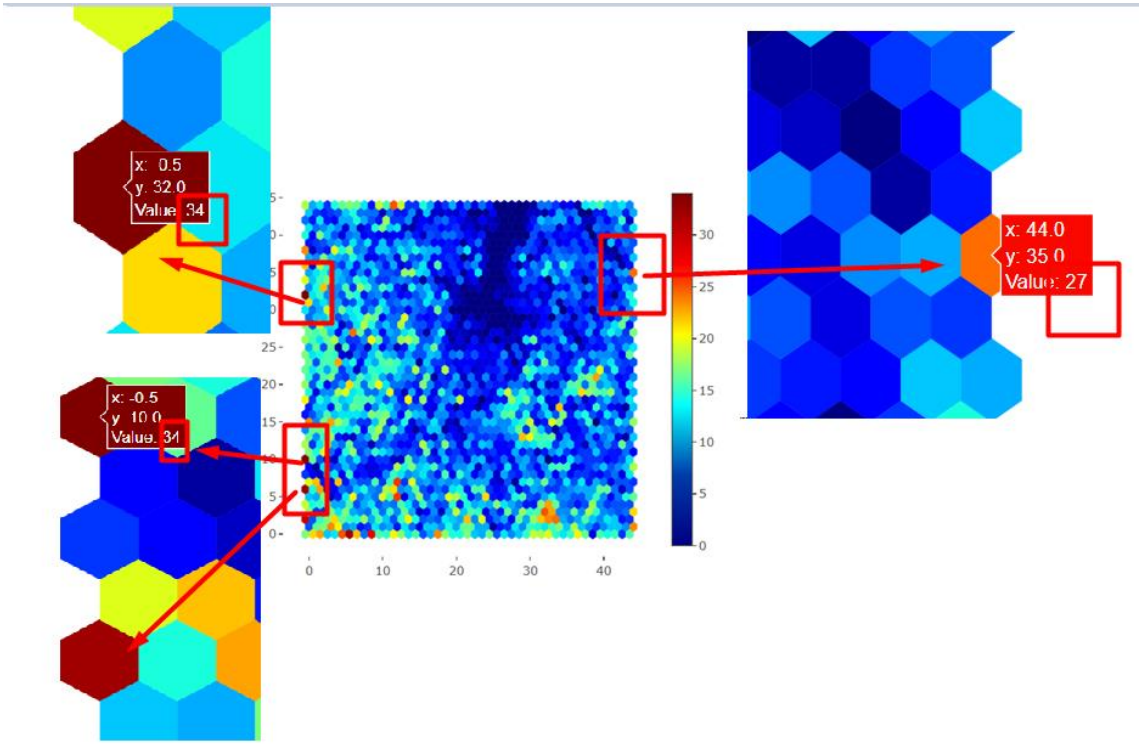
En cambio, las zonas claras, indicarán que dichas neuronas representan los cambios en los datos de forma gradual y proporcional al tamaño del mapa, como se espera de un modelo bien entrenado y fiel a los datos.

Sin embargo zonas oscuras no quieren decir que necesariamente el modelo esté mal entrenado o mal representado. En el caso de **datos anómalos**, estos suelen verse reflejados normalmente muy bien en dichas matrices.

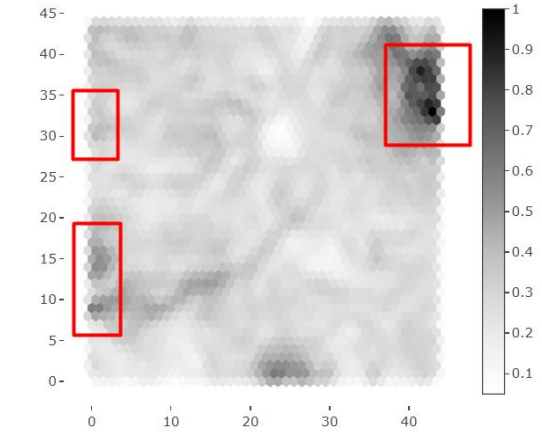
Normalmente cuando el mapa tiene zonas muy blancas y grises, se suele utilizar la escala logarítmica, para apreciarse mejor los comportamientos descritos arriba.

En el ejemplo anterior podemos comparar la mal representación de ciertos datos a los bordes del mapa. Si observamos el mapa de frecuencias de nuevo(de la topología hexagonal por simplificar la explicación)

Vemos como hay neuronas que tiene un índice **de frecuencia mucho mayor que sus vecinas,** que se traduce en un **color rojizo alrededor de colores azulados**, comportamiento que explicábamos en la sección del mapa de frecuencias.



Si observamos de nuevo su Matriz U, vemos como lo saltos grandes se traducen en colores más negros:



En la neurona del rectángulo de la derecha, vemos un color muy **negro** en la matriz u. Esto es debido a que el salto en el mapa de frecuencias es muy alto. El valor de la neurona anómala es de 27 en el mapa de frecuencias(que guarda relación directa con el valor del vector de pesos a la hora de mapear ejemplos),mientras que el de la neurona contigua a la izquierda es de 10. Esto es debido a que los **vectores de pesos de las neuronas están muy alejados, más que lo normal en neuronas contiguas**. Geográficamente se podría ver como el pico de una montaña en un valle. El valle está inclinado, pero la montaña sobresale.

En cambio, en los rectángulos de la derecha, vemos como el color es más grisáceo, porque las distancias entre los vectores de pesos son altas, pero no tanto. En el mapa de frecuencia se puede ver como esto es así: ahora el salto de la neurona izquierda superior es de 34 a 17. Esto quiere decir que un número de datos menor en diferencia de una neurona a otra se han mapeado en dichas neuronas, y las actualizaciones de pesos de los vecinos han ido igualando esa diferencia, por lo que ahora el color en la Matriz U es más grisáceo.

4.Conclusiones

Hemos visto un caso de estudio práctico sobre el uso de la herramienta y visto la posible información que nos pueden llegar a dar las distintas gráficas.

Durante este caso de estudio ,usando la aplicación, hemos detectado patrones de comportamiento y relaciones entre las características de los datos a analizar.

También hemos visto que a pesar de haber obtenido unos errores de cuantización y topográfico relativamente bajos, hay puntos en el mapa, concretamente 3 neuronas, en los que la representación en el mapa de los datos podría haber sido ligeramente mejor. Esto se podría haber conseguido utilizando otros valores la búsqueda óptima de hiperparámetros “Random Search” o “Grid Search” incluidos en la herramienta, para conseguido un modelo mejor. Dado que no era el objetivo de este caso de estudio, no se ha intentado afinar dicha representación.