Universidad Nacional Autónoma de México

Facultad de Ciencias

Redes Neuronales

Semestre 2020-2

Profesor: Sergio Hernández López

Ayudante: Rafael López Martínez

Ayudante de Laboratorio: Omar Alejandro Suarez Guerrero

Análisis comparativo sobre Restricted Boltzmann Machine contra método de agrupamiento K-means.

Gonzalez Duran Erick 314271103
Guevara Castro Miguel Andres 314091057
Salcedo Ramos Carlos Uriel 314189820

Introducción

Al tener una gran cantidad de animaciones japonesas, siempre se trata de buscar animaciones que puedan conseguir el interés de los usuarios para continuar consumiendo estos productos.Por lo que mediante una red Restricted Boltzmann Machine y un método de agrupamiento K-mean se busca realizar un análisis de resultados sobre las recomendaciones realizadas por la red neuronal contra K-means, dado un dataset de animaciones el cual contiene nombre, género, tipo de animación, episodios y rating.

Objetivo

Comparación de los resultados al utilizar Red neuronal artificial estocástica y un método de agrupamiento como sistemas recomendadores de animación japonesa.

Métodos

Obtención de datos (Kaggle)

A partir de la base de datos Kaggle disponible en línea, se obtuvieron dos datasets, en formato < .csv > cada uno contiene información correspodiente a cada data set, por un lado el dataset animes contiene nombre, genero y rating, entre otros; mientras que la información que se encuentra en usuarios es la id usuario, id anime y rating dado por el usuario.

K-means

1) Procesamiento de los datos de entrada

   Los datos que usamos están divididos en 2 datasets, los cuales queremos manipular de manera conjunta, esto se logró agregando unas columnas y eliminando algunas filas que para nosotros no son necesarias.
   Un ejemplo de esto es la obtención de una media de calificación dada por el usuario en el dataset de rating. La media de calificación nos ayudó a eliminar cualquier anime (del dataset de rating, no de animes) que el individuo haya visto y le haya puesto una calificación inferior a su media de calificación.
   Al final, solo fue necesario hacer una tabla mezcla entre el dataset de animes y de rating, lo que generó una tabla como se muestra a continuación:

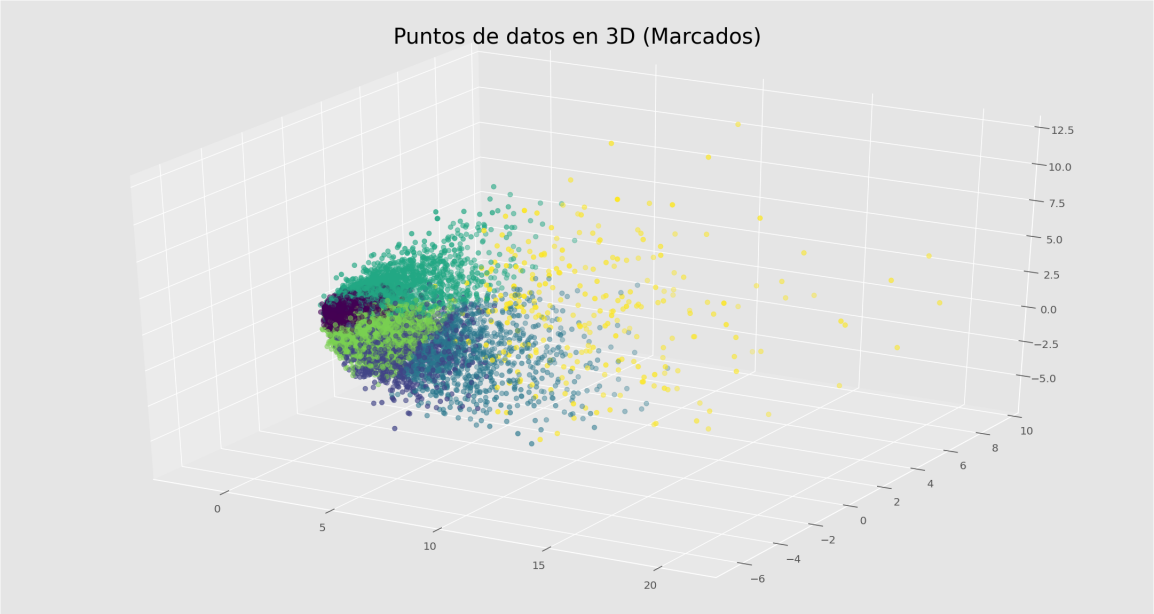
	anime_id	name	genre	type	episodes	rating	members	user_id	rating_usuario	prom_rating
0	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	152	10	7.699301
1	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	244	10	8.729242
2	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	271	10	7.372287
3	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	322	10	8.356322
4	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	398	10	-0.832298
5	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	462	8	7.374593
6	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	490	10	8.062500
7	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	548	10	8.112360
8	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	570	10	8.388889
9	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	598	10	8.680328

   Esta tabla es la que se manipulara a lo largo de todo el experimento.

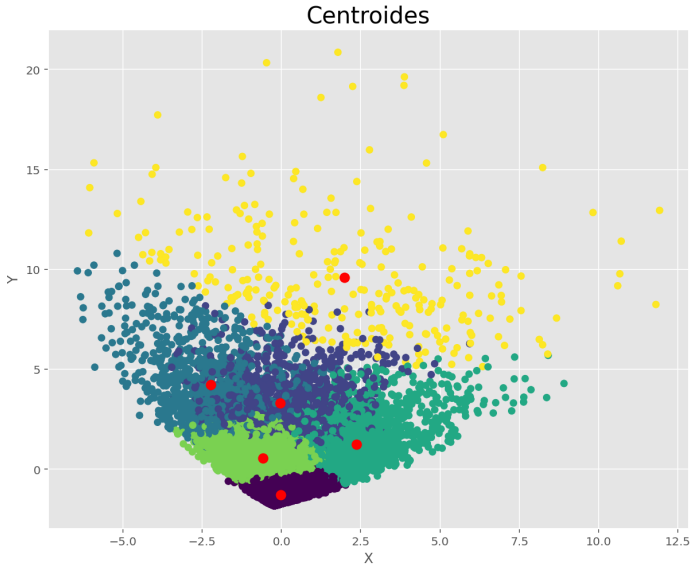
2) Entrenamiento
K-means es un método de clustering, es decir, no tenemos una fase de entrenamiento como tal, esta fase es reemplazada con una fase de ajuste.
Para realizar el ajuste se usó el método fit_transform de sklearn.decomposition.PCA y el método fit de sklearn.cluster.KMeans.

- Fit_transform: Ajusta el modelo con el dataset y aplica una reducción dimensional.
- Fit: Crea el clustering basado en Kmeans, al final nos regresa un array con los pesos de cada observación en el dataset.

Con los datos obtenidos de la fase de ajuste, generamos una gráfica que nos ayuda a visualizar como nuestros datos fueron agrupados en K-grupos, en este caso, usamos 6 clusters.



Los centroides (puntos rojos) son los elementos que a grandes rasgos, nos ayudan a clasificar y diferenciar nuestros datos.



RBM (Restricted Boltzmann Machine)

1) Depuramiento de los datos de entrada

   Al recibir todos los datos de cada dataset (animes y usuarios), nos dimos cuenta de que para nuestro fin, habian datos innecesarios en el dataset de usuarios, ya que habian animes no puntuados por ciertos usuarios, por lo que depuramos los datos del dataset de forma que solo quedaran usuarios con calificaciónes validas (entre el 1.0 al 10.0).

2) Entrenamiento Para el entrenamiento se tomará un conjunto de 1000 usuarios donde se creó un nodo por cada usuario para la capa visible con la información de las calificaciones de cada anime que haya calificado el usuario, y 50 nodos para la capa oculta. Ahora se procesarán los datos de entrada por medio de la creación de la matriz de pesos, la capa de activacion, siendo en esta ocación usada la funcion de activación sigmoide y por último el muestreo de Gibbs que nos permite dar una predicción probabilística dado un valor ó suceso, y pasado k iteraciónes nos permite obtener el vector de salida a partir de los datos originales ya que despues se reconstruiran los datos de salida obtenidos en las capas ocultas hacia las capas visibles por medio de otra capa de activación con la funcion sigmoide y se le aplicara el muestreo de Gibbs a los resultados para tener la prediccion de los datos de entrada. Y por otra parte para la actualización de pesos en cada iteración del entrenamiento se usuará Constrastive Divergence (ayuda a mejorar los resultados de salida), la taza de aprendizaje y los pesos. Todo esto se hara por 15 epocas con un lote de tamaño 100 y se mostrará el porcentaje de error conforme los resultados de las epocas.

Resultados

RBM

	anime_id	name	genre	type	episodes	rating_x	members	anime_indice_x	Que tan recomendable es	rating_y
841	20	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841	0.179105	NaN
40	1535	Death Note	Mystery, Police, Psychological, Supernatural, ...	TV	37	8.71	1013917	40	0.117979	9.0
19	1575	Code Geass: Hangyaku no Lelouch	Action, Mecha, Military, School, Sci-Fi, Super...	TV	25	8.83	715151	19	0.054991	NaN
643	9919	Ao no Exorcist	Action, Demons, Fantasy, Shounen, Supernatural	TV	25	7.92	583823	643	0.054588	NaN
10	4181	Clannad: After Story	Drama, Fantasy, Romance, Slice of Life, Supern...	TV	24	9.06	456749	10	0.052776	NaN
164	853	Ouran Koukou Host Club	Comedy, Harem, Romance, School, Shoujo	TV	26	8.39	422271	164	0.051441	6.0
582	269	Bleach	Action, Comedy, Shounen, Super Power, Supernat...	TV	366	7.95	624055	582	0.050866	9.0
830	10793	Guilty Crown	Action, Drama, Sci-Fi, Super Power	TV	22	7.81	460959	830	0.039709	NaN
440	3588	Soul Eater	Action, Adventure, Comedy, Fantasy, Shounen, S...	TV	51	8.08	580184	440	0.034689	8.0
449	22319	Tokyo Ghoul	Action, Drama, Horror, Mystery, Psychological,...	TV	12	8.07	618056	449	0.029477	NaN

Figure 1: Resultados RBM

Recomendaciones generadas por RBM que podemos observar en la imagen 1, donde el valor de recomendación si llega a 0.1 es algo muy recomendable dado los datos del usuario a inspeccionar; además se puede observar animes que el ya calificó y vio dandonos una idea de que tan cernaos son los resultados a sus gustos.

K-Means

```
keyword_occurences = count_word(datac0, 'genre', set_keywords)
keyword_occurences[0:10]
```

```
[[' Romance', 8],
 [' School', 7],
 ['Comedy', 7],
 ['Action', 6],
 [' Drama', 5],
 [' Supernatural', 5],
 [' Shounen', 5],
 [' Fantasy', 4],
 [' Slice of Life', 3],
 [' Ecchi', 3]]
```

Figure 2: Resultados K-Means

Como ya se mencionó anteriormente, K-means es un método de clustering, es decir, solo nos ayuda a agrupar datos con características similares. Al final solo obtuvimos que características eran las predominantes en cada cluster, en este caso, que géneros de anime son las más vistos y mejor puntuados en el cluster.

Conclusiones

RBM

En RBM pudimos obtener resultados útiles a partir de los datos introduccidos como entrenamiento ya que se puede observar animes de todo tipo de generos, duración y calificaciones, que se asemejan a lo que el usuario ha visto y calificado, además se eligió esa cantidad de elementos de entrenamiento ya que al introducir aún más, se tardaba mucho más, las predicciones salian mucho más especializadas pero con un bajo nivel de recomendación por lo mismo de no tener casi los mismo datos para su recomendación, permitiendonos que con la cantidad propuesta exista una mayor variedad de recomendaciones que tengan sentido al usuario sin limitarlo mucho a solo los generos que el ha visto.

K-Means

Con el resultado obtenido en K-means, es más que obvio que no es una recomendación, pero, es una herramienta muy útil y muy intuitiva para poder obtener características que nos pueden ayudar a generar una recomendación basada en esas características.
Ahora, dados los resultados obtenidos, pensamos que sería una buena idea crear una función que en base al resultado del clustering y un individuo perteneciente al cluster que estamos observando, genere una lista de animes que el individuo no haya visto y que tengan el mayor número de características similares al del cluster, es decir, que los géneros del anime coincidan en su mayoría con los de la lista que obtenemos, lamentablemente no pudimos implementarla a tiempo para la presentación de este proyecto.

Conclusiones Generales

Las recomendaciones generadas a partir de RBM son más específicas mientras que el resultado que obtuvimos en K-means es diferente ya que muestra un conjunto de características particulares de un cluster.
Por lo que podemos concluir que una complementa a la otra ya que con las características de K-means en conjunto a RBM, se pude generar un resultado completo.
RBM al ser tan especifico es mejor para una recomendación especifica por ejemplo cuando el usuario termina de ver un anime y quiere ver otro nuevo, mientras que K-means es mejor cuando el usuario quiere saber varios generos que se le recomienda.

Bibliography

[1] Kaggle,https://www.kaggle.com

[2] Sayantini, *What are Restricted Boltzmann Machines?*, Mayo 2020 https://www.edureka.co/blog/restricted-boltzmann-machine-tutorial/

[3] *K-Means en Python paso a paso*, https://www.aprendemachinelearning.com/k-means-en-python-paso-a-paso/

[4] Deep Learning A-Z™: Self Organizing Maps (SOM) - K-Means Clustering, https://www.slideshare.net/KirillEremenko/deep-learning-az-self-organizing-maps-som-kmeans-clustering

[5] API Reference: KMeans, https://scikit-learn.org/stable/modules/classes.html#module-sklearn.cluster