



PRÁCTICA

K-Means en Python paso a paso

⌚ March 12, 2018 by Na8

K-Means es un algoritmo **no supervisado** de **Clustering**. Se utiliza cuando tenemos un montón de datos **sin etiquetar**. El objetivo de este algoritmo es el de encontrar “K” grupos (clusters) entre los datos crudos. En este artículo repasaremos sus conceptos básicos y veremos un ejemplo paso a paso en python que podemos descargar.



K-MEANS

CON PYTHON
PASO A PASO

WWW.APRENDEMACHINELEARNING.COM

Cómo funciona K-Means

El algoritmo trabaja iterativamente para asignar a cada “punto” (las filas de nuestro conjunto de entrada forman una coordenada) uno de los “K” grupos basado en sus características. Son agrupados en base a la similitud de sus features (las columnas). Como resultado de ejecutar el algoritmo tendremos:

- Los “centroids” de cada grupo que serán unas “coordenadas” de cada uno de los K conjuntos que se utilizarán para poder etiquetar nuevas muestras.

- Etiquetas para el conjunto de datos de entrenamiento. Cada etiqueta perteneciente a uno de los K grupos formados.

Los grupos se van definiendo de manera “orgánica”, es decir que se va ajustando su posición en cada iteración del proceso, hasta que converge el algoritmo. Una vez hallados los centroids deberemos analizarlos para ver cuales son sus características únicas, frente a la de los otros grupos. Estos grupos son las etiquetas que genera el algoritmo.

Casos de Uso de K-Means

El algoritmo de Clustering K-means es **uno de los más usados** para encontrar grupos ocultos, o sospechados en teoría sobre un conjunto de datos no etiquetado. Esto puede servir para confirmar -o desterrar- alguna teoría que teníamos asumida de nuestros datos. Y también puede ayudarnos a descubrir relaciones asombrosas entre conjuntos de datos, que de manera manual, no hubiéramos reconocido. Una vez que el algoritmo ha ejecutado y obtenido las etiquetas, será fácil clasificar nuevos valores o muestras entre los grupos obtenidos.

Algunos casos de uso son:

- Segmentación por Comportamiento: relacionar el carrito de compras de un usuario, sus tiempos de acción e información del perfil.
- Categorización de Inventory: agrupar productos por actividad en sus ventas
- Detectar anomalías o actividades sospechosas: según el comportamiento en una web reconocer un troll -o un bot- de un usuario normal

Aquí se listan las **aplicaciones más frecuentes** en Machine Learning

Datos de Entrada para K-Means

Las “features” o características que utilizaremos como entradas para aplicar el algoritmo k-means deberán ser de valores numéricos, continuos en lo posible. En caso de valores categóricos (por ej. Hombre/Mujer o Ciencia Ficción, Terror, Novela,etc) se puede intentar pasarlo a valor numérico, pero no es recomendable pues no hay una “distancia real” - como en el caso de géneros de película o libros-. Además es recomendable que los valores utilizados estén normalizados, manteniendo una misma escala. En algunos casos también funcionan mejor datos porcentuales en vez de absolutos. No conviene utilizar features que estén correlacionados o que sean escalares de otros. Recordar los **7 pasos para el Aprendizaje Automático**.

El Algoritmo K-means



El algoritmo utiliza un proceso iterativo en el que se van ajustando los grupos para producir el resultado final. Para ejecutar el algoritmo deberemos pasar como entrada el conjunto de datos y un valor de K. El conjunto de datos serán las características o features para cada punto. Las posiciones iniciales de los K centroids serán asignadas de manera aleatoria de cualquier punto del conjunto de datos de entrada. Luego se itera en dos pasos:

1- Paso de Asignación de datos

En este paso, cada “fila” de nuestro conjunto de datos se asigna al centroide más cercano basado en la distancia cuadrada Euclídea. Se utiliza la siguiente fórmula (donde $\text{dist}()$ es la distancia Euclídea standard):

$$\underset{c_i \in C}{\operatorname{argmin}} \text{dist}(c_i, x)^2$$

2-Paso de actualización de Centroid

En este paso los centroides de cada grupo son recalculados. Esto se hace tomando una media de todos los puntos asignados en el paso anterior.

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$

El algoritmo itera entre estos pasos hasta cumplir un criterio de detención:

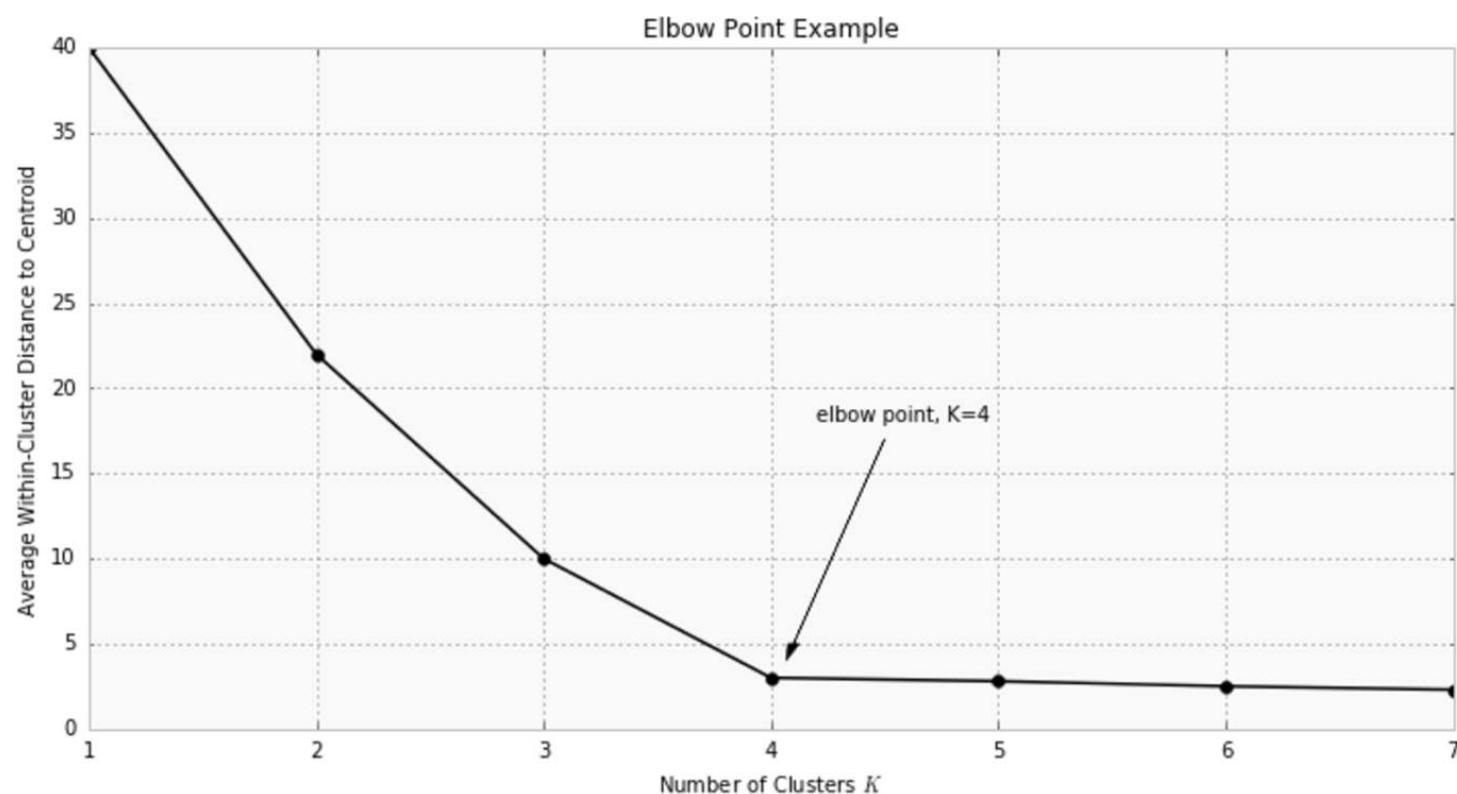
- * si no hay cambios en los puntos asignados a los grupos,
- * o si la suma de las distancias se minimiza,
- * o se alcanza un número máximo de iteraciones.

El algoritmo converge a un resultado que puede ser el óptimo local, por lo que será conveniente volver a ejecutar más de una vez con puntos iniciales aleatorios para confirmar si hay una salida mejor. Recuerden siempre seguir [los 7 pasos para construir IA](#)

Elegir el valor de K

Este algoritmo funciona pre-seleccionando un valor de K. Para encontrar el número de clusters en los datos, deberemos ejecutar el algoritmo para un rango de valores K, ver los resultados y comparar características de los grupos obtenidos. En general no hay un modo exacto de determinar el valor K, pero se puede estimar con aceptable precisión siguiendo la siguiente técnica:

Una de las métricas usada para comparar resultados es la **distancia media entre los puntos de datos y su centroid**. Como el valor de la media diminuirá a medida de aumentemos el valor de K, deberemos utilizar la distancia media al centroide en función de K y entonrar el “punto codo”, donde la tasa de descenso se “afila”. Aquí vemos una gráfica a modo de ejemplo:



Un ejemplo K-Means en Python con Sklearn

Como ejemplo utilizaremos de entradas un conjunto de datos que obtuve de un proyecto propio, en el que se analizaban rasgos de la personalidad de usuarios de Twitter. He filtrado a 140 “famosos” del mundo en diferentes áreas: deporte, cantantes, actores, etc. Basado en una metodología de psicología conocida como “Ocean: The Big Five” tendemos como características de entrada:

- usuario (el nombre en Twitter)
- “op” = Openness to experience – grado de apertura mental a nuevas experiencias, curiosidad, arte
- “co” =Conscientiousness – grado de orden, prolijidad, organización
- “ex” = Extraversion – grado de timidez, solitario o participación ante el grupo social
- “ag” = Agreeableness – grado de empatía con los demás, temperamento
- “ne” = Neuroticism, – grado de neuroticismo, nervioso, irritabilidad, seguridad en sí mismo.
- Wordcount – Cantidad promedio de palabras usadas en sus tweets
- Categoría – Actividad laboral del usuario (actor, cantante, etc.)

Utilizaremos el algoritmo K-means para que agrupe estos usuarios -no por su actividad laboral- si no, por sus similitudes en la personalidad. Si bien tenemos 8 columnas de entrada, **sólo utilizaremos 3** en este ejemplo, de modo que podamos ver en un gráfico tridimensional -y sus proyecciones a 2D- los grupos resultantes. Pero para casos reales, podemos utilizar todas las dimensiones que necesitemos. Una de las hipótesis que podríamos tener es: “Todos los cantantes tendrán personalidad parecida” (y así con cada

rubro laboral). Pues veremos si lo probamos, o por el contrario, los grupos no están relacionados necesariamente con la actividad de estas Celebridades.



Requerimientos para el Ejercicio

Necesitaremos tener Python 2.7 o 3.5+. Mejor si tenemos instalada una suite como [Anaconda](#) o [Canopy](#) (que funcionan en Windows, Mac y Linux). Puedes seguir este tutorial donde explico [cómo instalar tu ambiente de desarrollo](#). Crearemos un [Jupiter notebook](#) para seguir paso a paso el ejercicio e importaremos un [archivo de entrada csv](#). Utilizaremos los paquetes scikit-learn, pandas, matplotlib y numpy.

Agrupar usuarios Twitter de acuerdo a su personalidad con K-means

Implementando K-means en Python con Sklearn

Comenzaremos importando las librerías que nos asistirán para ejecutar el algoritmo y graficar.

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sb
5 from sklearn.cluster import KMeans
6 from sklearn.metrics import pairwise_distances_argmin_min
7
8 %matplotlib inline
9 from mpl_toolkits.mplot3d import Axes3D
10 plt.rcParams['figure.figsize'] = (16, 9)
11 plt.style.use('ggplot')

```

Importamos [el archivo csv](#) -para simplificar, suponemos que el archivo se encuentra en el mismo directorio que el notebook- y vemos los primeros 5 registros del archivo tabulados.

```

1 dataframe = pd.read_csv(r"analisis.csv")
2 dataframe.head()

```

	usuario	op	co	ex	ag	ne	wordcount	categoria
0	3gerardpique	34.297953	28.148819	41.948819	29.370315	9.841575	37.0945	7
1	aguerosergiokun	44.986842	20.525865	37.938947	24.279098	10.362406	78.7970	7
2	albertochicote	41.733854	13.745417	38.999896	34.645521	8.836979	49.2604	4
3	AlejandroSanz	40.377154	15.377462	52.337538	31.082154	5.032231	80.4538	2
4	alfredocasero1	36.664677	19.642258	48.530806	31.138871	7.305968	47.0645	4

También podemos ver una tabla de información estadística que nos provee Pandas dataframe:

```
1 dataframe.describe()
```

	op	co	ex	ag	ne	wordcount	categoria
count	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000
mean	44.414591	22.977135	40.764428	22.918528	8.000098	98.715484	4.050000
std	8.425723	5.816851	7.185246	7.657122	3.039248	44.714071	2.658839
min	30.020465	7.852756	18.693542	9.305985	1.030213	5.020800	1.000000
25%	38.206484	19.740299	36.095722	17.050993	6.086144	66.218475	2.000000
50%	44.507091	22.466718	41.457492	21.384554	7.839722	94.711400	3.500000
75%	49.365923	26.091606	45.197769	28.678867	9.758189	119.707925	7.000000
max	71.696129	49.637863	59.824844	40.583162	23.978462	217.183200	9.000000

El archivo contiene diferenciadas 9 categorías -actividades laborales- que son:

1. Actor/actriz
2. Cantante
3. Modelo
4. Tv, series
5. Radio
6. Tecnología
7. Deportes
8. Política
9. Escritor

Para saber cuantos registros tenemos de cada uno hacemos:

```
1 print(dataframe.groupby('categoria').size())
```

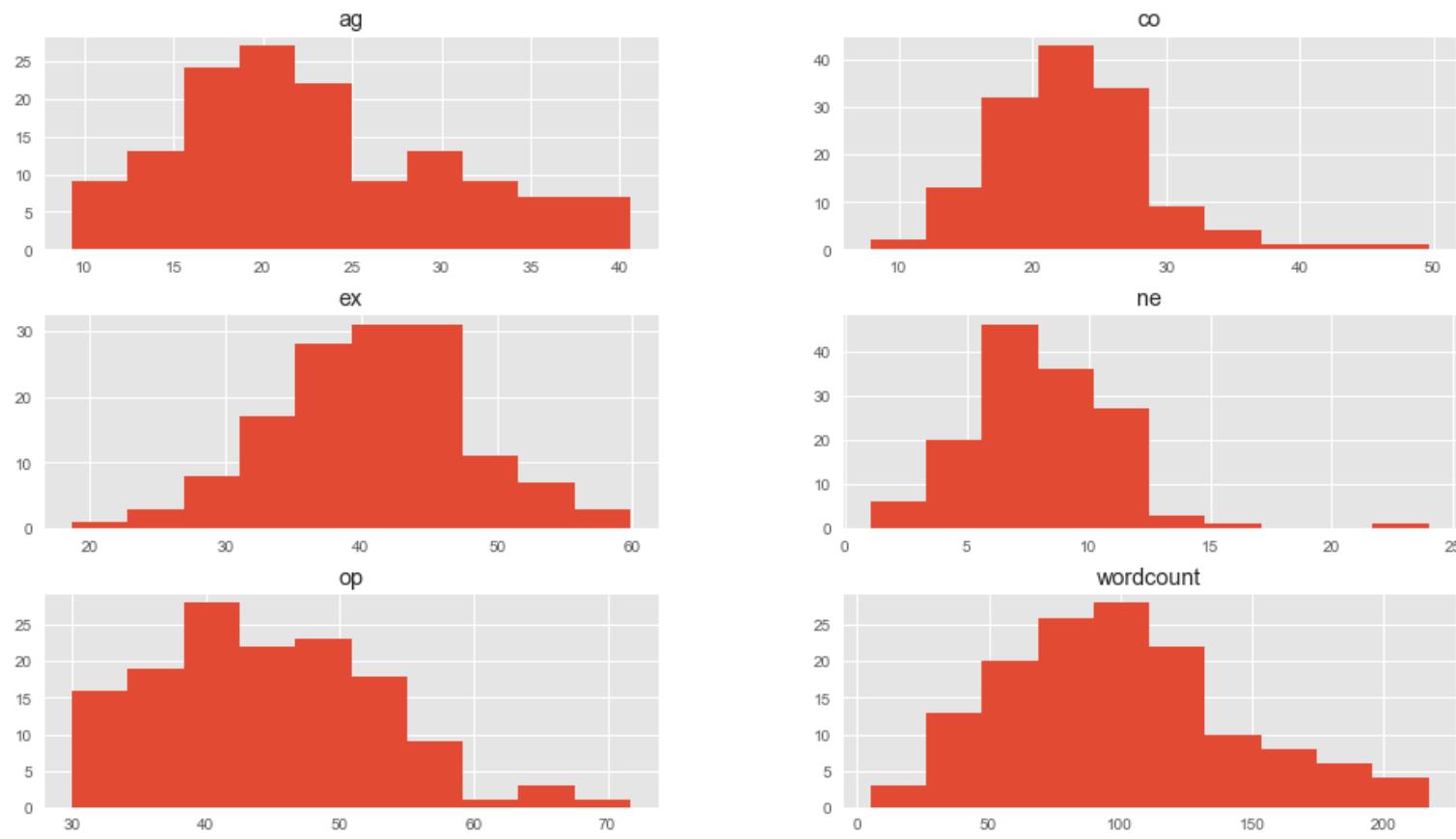
```
categoria
1    27
2    34
3     9
4    19
5     4
6     8
7    17
8    16
9     6
dtype: int64
```

Como vemos tenemos 34 cantantes, 27 actores, 17 deportistas, 16 políticos,etc.

Visualización de Datos

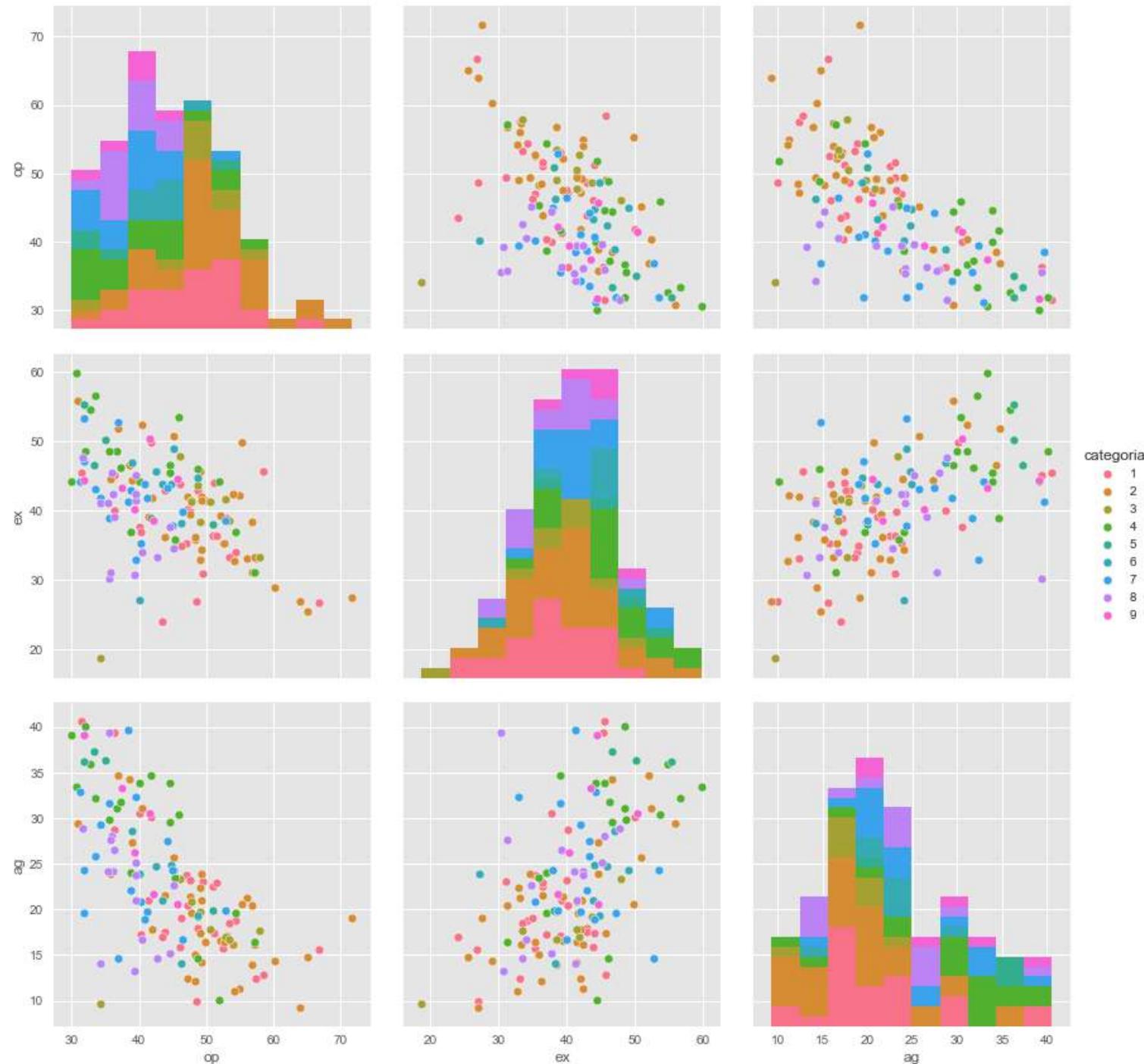
Veremos graficamente nuestros datos para tener una idea de la dispersión de los mismos:

```
1 dataframe.drop(['categoria'],1).hist()
2 plt.show()
```



En este caso seleccionamos 3 dimensiones: op, ex y ag y las cruzamos para ver si nos dan alguna pista de su agrupación y la relación con sus categorías.

```
1 sb.pairplot(dataframe.dropna(), hue='categoria', size=4, vars=["op", "ex", "ag"], kind='scatter')
```



Revisando la gráfica no pareciera que ha a algún tipo de agrupación o correlación entre los usuarios y sus categorías.

Definimos la entrada

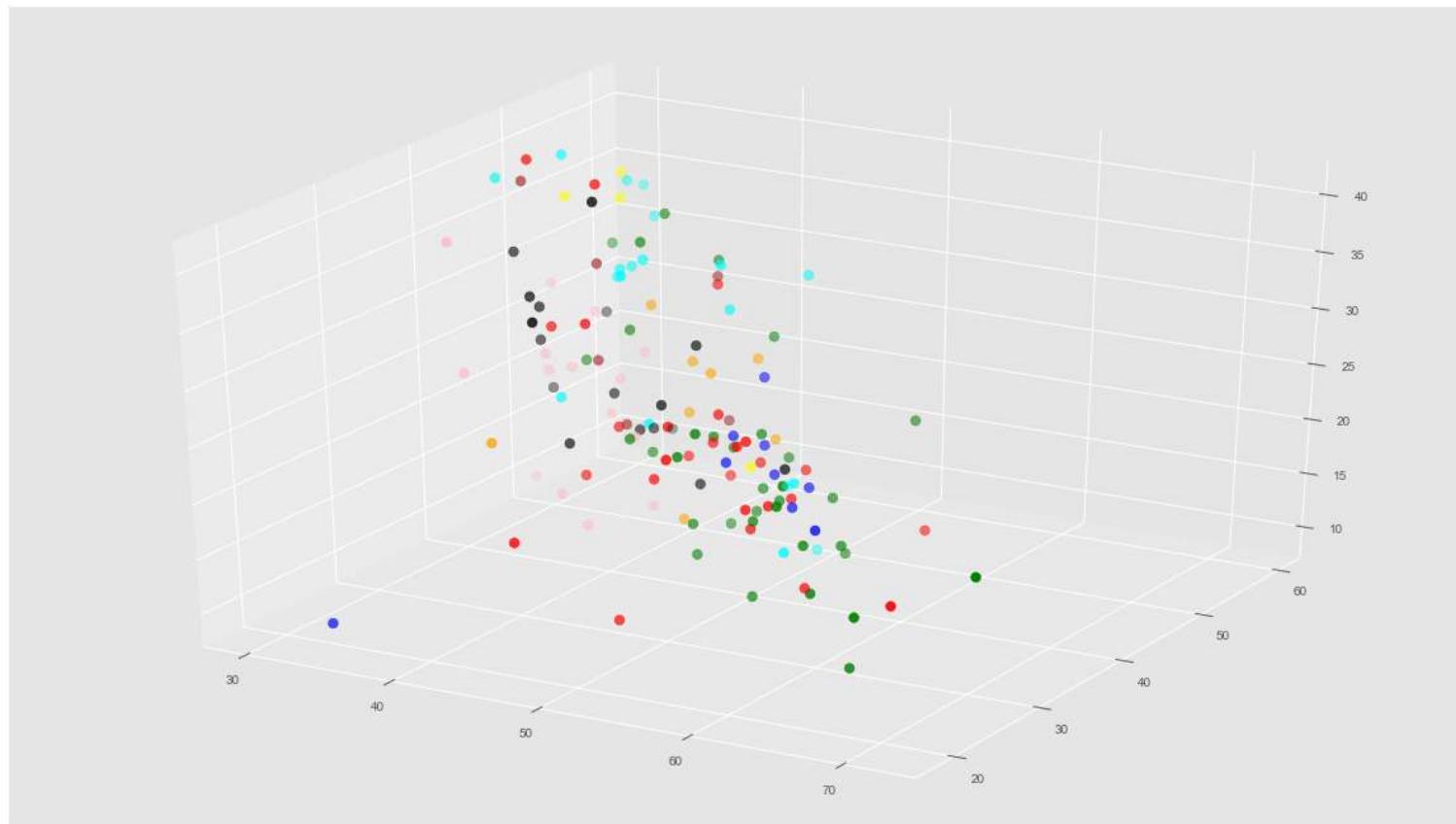
Concretamos la estructura de datos que utilizaremos para alimentar el algoritmo. Como se ve, sólo cargamos las columnas op, ex y ag en nuestra variable X.

```
1 X = np.array(dataframe[["op", "ex", "ag"]])
2 y = np.array(dataframe['categoria'])
3 X.shape
```

(140, 3)

Ahora veremos una gráfica en 3D con 9 colores representando las categorías.

```
1 fig = plt.figure()
2 ax = Axes3D(fig)
3 colores=['blue', 'red', 'green', 'blue', 'cyan', 'yellow', 'orange', 'black', 'pink', 'brown', 'purple']
4 asignar=[]
5 for row in y:
6     asignar.append(colores[row])
7 ax.scatter(X[:, 0], X[:, 1], X[:, 2], c=asignar, s=60)
```

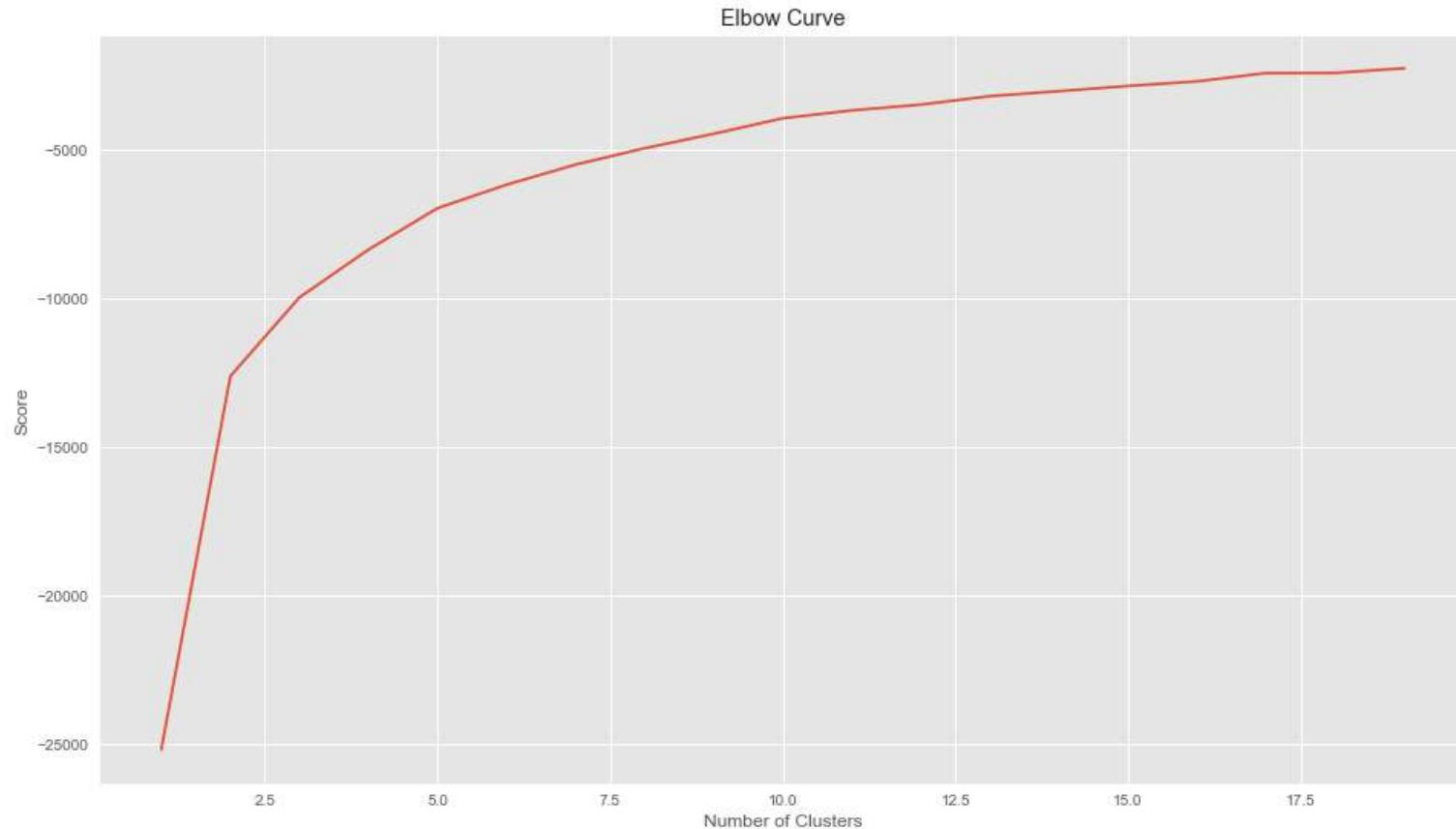


Veremos si con K-means, podemos “pintar” esta misma gráfica de otra manera, con clusters diferenciados.

Obtener el valor K

Vamos a hallar el valor de K haciendo una gráfica e intentando hallar el “punto de codo” que comentábamos antes. Este es nuestro resultado:

```
1 Nc = range(1, 20)
2 kmeans = [KMeans(n_clusters=i) for i in Nc]
3 kmeans
4 score = [kmeans[i].fit(X).score(X) for i in range(len(kmeans))]
5 score
6 plt.plot(Nc,score)
7 plt.xlabel('Number of Clusters')
8 plt.ylabel('Score')
9 plt.title('Elbow Curve')
10 plt.show()
```



Realmente la curva es bastante “suave”. Considero a 5 como un buen número para K. Según vuestro criterio podría ser otro.

Ejecutamos K-Means

Ejecutamos el algoritmo para 5 clusters y obtenemos las etiquetas y los centroids.

```

1 kmeans = KMeans(n_clusters=5).fit(X)
2 centroids = kmeans.cluster_centers_
3 print(centroids)

[[ 49.80086386  40.8972579   17.48224326]
 [ 39.63830586  44.75784737  25.86962057]
 [ 58.58657531  31.02839375  15.6120435 ]
 [ 34.5303535   48.01261321  35.01749504]
 [ 42.302263    33.65449587  20.812626  ]]

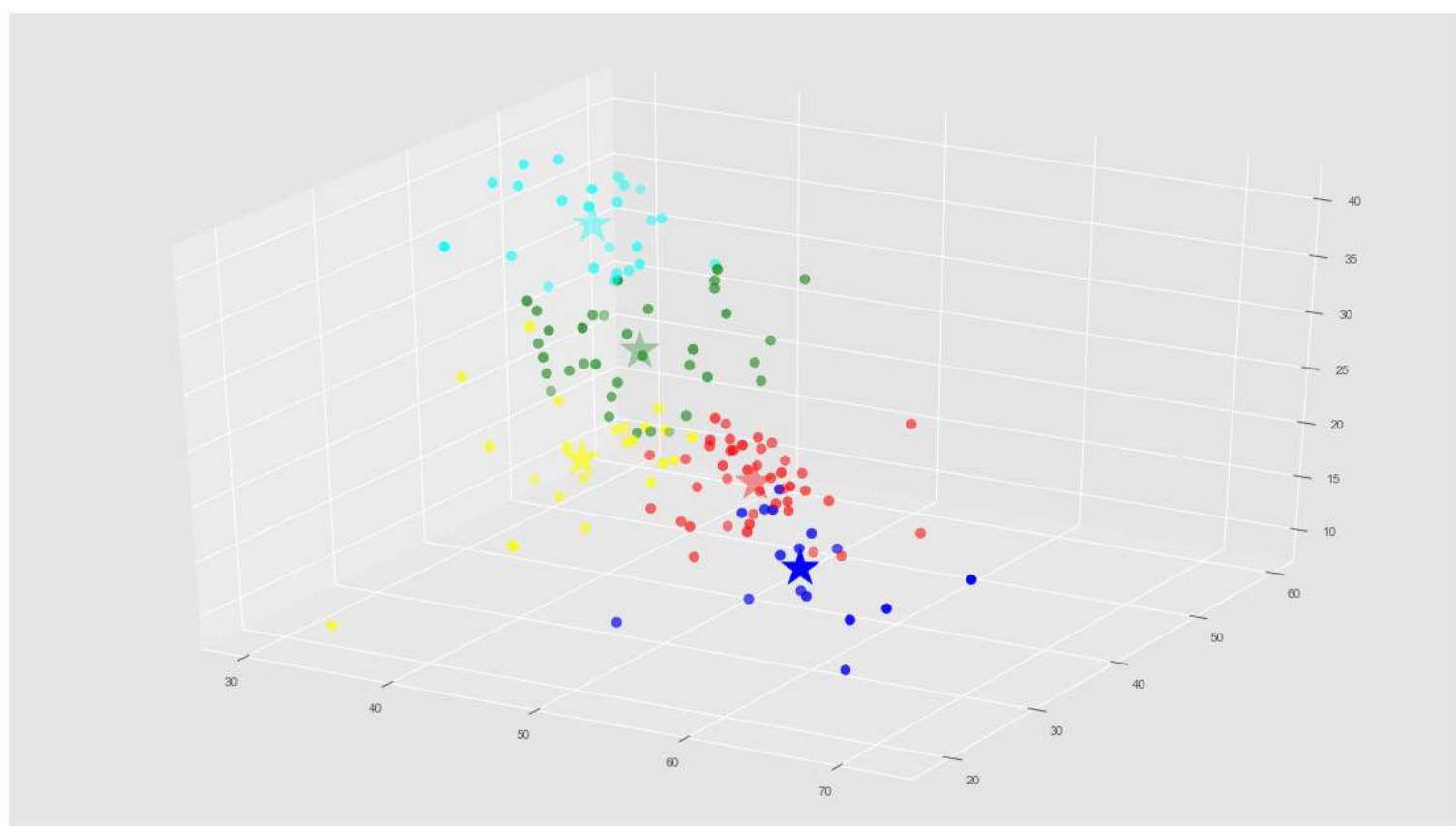
```

Ahora veremos esto en una gráfica 3D con colores para los grupos y veremos si se diferencian: (las estrellas marcan el centro de cada cluster)

```

1 # Predicting the clusters
2 labels = kmeans.predict(X)
3 # Getting the cluster centers
4 C = kmeans.cluster_centers_
5 colores=['red','green','blue','cyan','yellow']
6 asignar=[]
7 for row in labels:
8     asignar.append(colores[row])
9
10 fig = plt.figure()
11 ax = Axes3D(fig)
12 ax.scatter(X[:, 0], X[:, 1], X[:, 2], c=asignar,s=60)
13 ax.scatter(C[:, 0], C[:, 1], C[:, 2], marker='*', c=colores, s=1000)

```



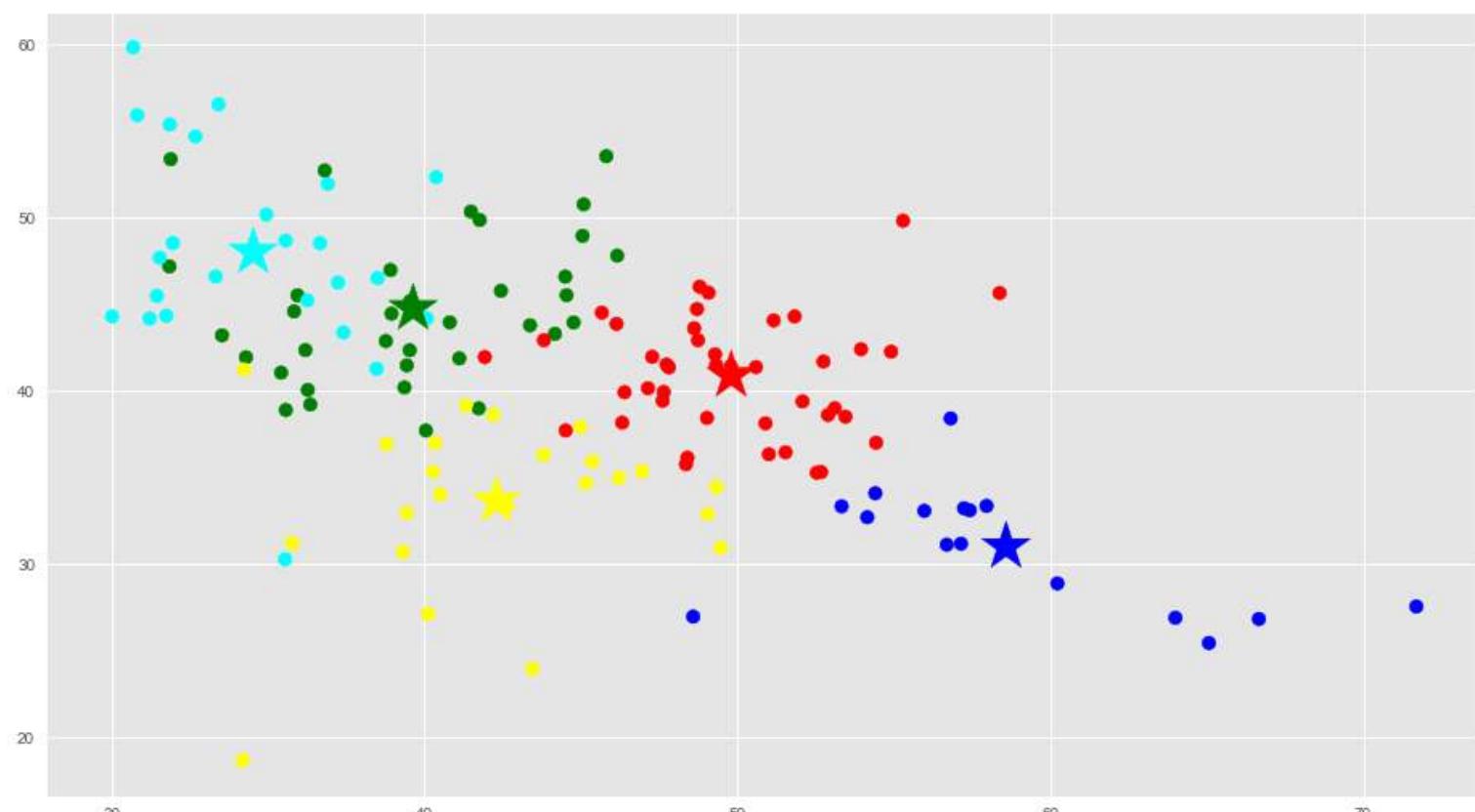
Aquí podemos ver que el Algoritmo de K-Means con K=5 ha agrupado a los 140 usuarios Twitter por su personalidad, teniendo en cuenta las 3 dimensiones que utilizamos: Openess, Extraversion y Agreeableness. Pareciera que no hay necesariamente una relación en los grupos con sus actividades de Celebrity.

Haremos 3 gráficas en 2 dimensiones con las proyecciones a partir de nuestra gráfica 3D para que nos ayude a visualizar los grupos y su clasificación:

```

1 # Getting the values and plotting it
2 f1 = dataframe['op'].values
3 f2 = dataframe['ex'].values
4
5 plt.scatter(f1, f2, c=asignar, s=70)
6 plt.scatter(C[:, 0], C[:, 1], marker='*', c=colores, s=1000)
7 plt.show()

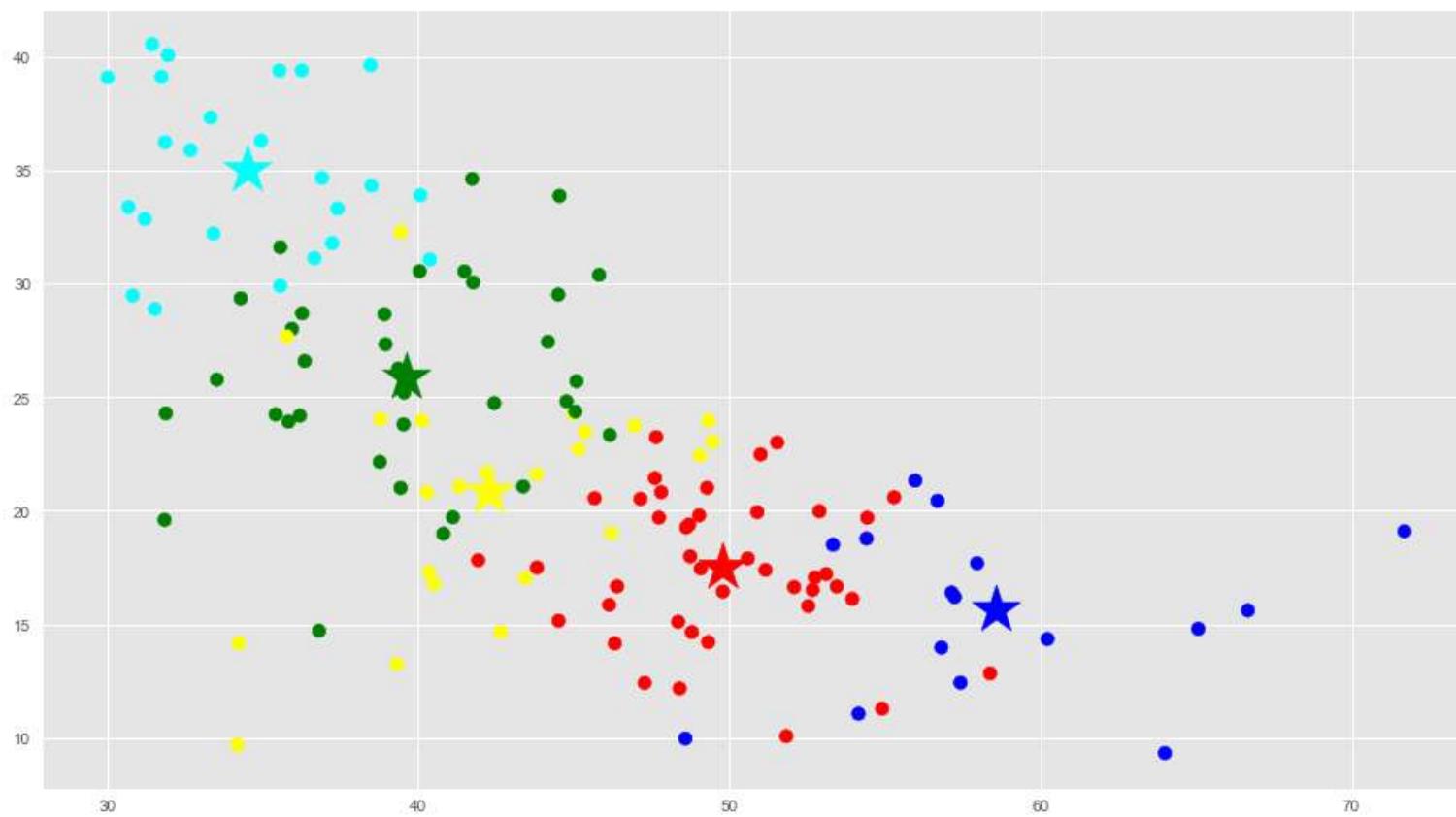
```



```

1 # Getting the values and plotting it
2 f1 = dataframe['op'].values
3 f2 = dataframe['ag'].values
4
5 plt.scatter(f1, f2, c=asignar, s=70)
6 plt.scatter(C[:, 0], C[:, 2], marker='*', c=colores, s=1000)
7 plt.show()

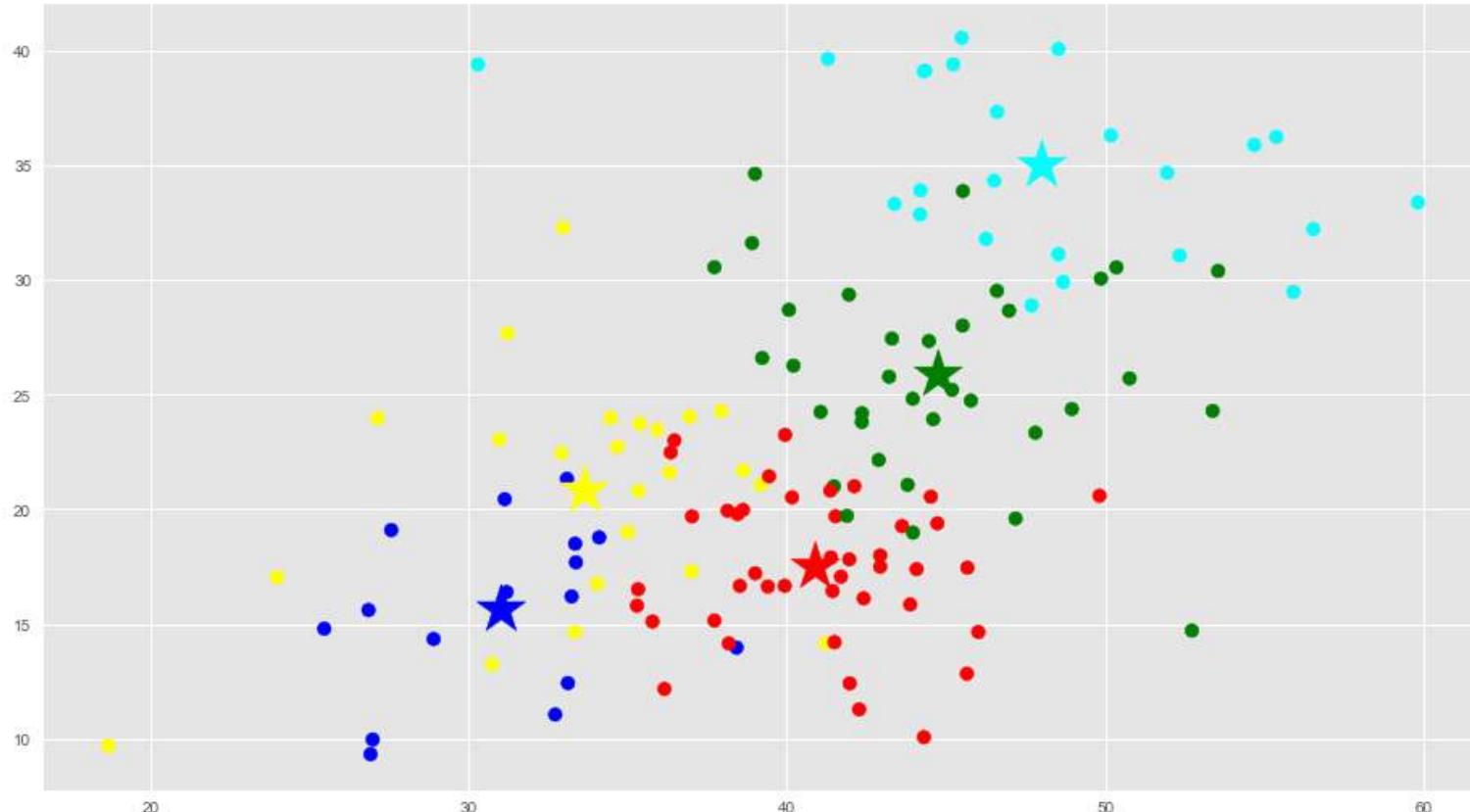
```



```

1 f1 = dataframe['ex'].values
2 f2 = dataframe['ag'].values
3
4 plt.scatter(f1, f2, c=asignar, s=70)
5 plt.scatter(C[:, 1], C[:, 2], marker='*', c=colores, s=1000)
6 plt.show()

```



En estas gráficas vemos que están bastante bien diferenciados los grupos.

Podemos ver cada uno de los clusters cuantos usuarios tiene:

```

1 copy = pd.DataFrame()
2 copy['usuario']=dataframe['usuario'].values
3 copy['categoria']=dataframe['categoria'].values
4 copy['label'] = labels;
5 cantidadGrupo = pd.DataFrame()
6 cantidadGrupo['color']=colores
7 cantidadGrupo['cantidad']=copy.groupby('label').size()
8 cantidadGrupo

```



	color	cantidad
0	red	42
1	green	33
2	blue	16
3	cyan	27
4	yellow	22

Y podemos ver la diversidad en rubros laborales de cada uno. Por ejemplo en el grupo 0 (rojo), vemos que hay de todas las actividades laborales aunque predominan de actividad 1 y 2 correspondiente a Actores y Cantantes con 11 y 15 famosos.

```

1 group_referrer_index = copy['label'] ==0
2 group_referrals = copy[group_referrer_index]
3
4 diversidadGrupo = pd.DataFrame()
5 diversidadGrupo['categoria']=[0,1,2,3,4,5,6,7,8,9]
6 diversidadGrupo['cantidad']=group_referrals.groupby('categoria').size()
7 diversidadGrupo

```

	categoria	cantidad
0	0	NaN
1	1	11.0
2	2	15.0
3	3	6.0
4	4	3.0
5	5	1.0
6	6	2.0
7	7	2.0
8	8	1.0
9	9	1.0

De categoría 3 “modelos” hay 6 sobre un total de 9.

Buscaremos los usuarios que están más cerca a los centroids de cada grupo que podríamos decir que tienen los rasgos de personalidad característicos que representan a cada cluster:

```

1 #vemos el representante del grupo, el usuario cercano a su centroid
2 closest, _ = pairwise_distances_argmin_min(kmeans.cluster_centers_, X)
3 closest

```

```
array([21, 107, 82, 80, 91]) #posicion en el array de usuarios
```

```

1 users=dataframe['usuario'].values
2 for row in closest:
3     print(users[row])

```

<

carmenelectra
Pablo_Iglesias_

 JudgeJudy

JPVarsky

kobebryant



En los centros vemos que tenemos una modelo, un político, presentadora de Tv, locutor de Radio y un deportista.

Clasificar nuevas muestras

Y finalmente podemos agrupar y etiquetar nuevos usuarios twitter con sus características y clasificarlos. Vemos el ejemplo con el usuario de David Guetta y nos devuelve que pertenece al grupo 1 (verde).

```
1 X_new = np.array([[45.92, 57.74, 15.66]]) #davidguetta
2
3 new_labels = kmeans.predict(X_new)
4 print(new_labels)
```

[1]

Conclusiones

El algoritmo de K-means nos ayudará a crear clusters cuando tengamos grandes grupos de datos sin etiquetar, cuando queramos intentar descubrir nuevas relaciones entre features o para probar o declinar hipótesis que tengamos de nuestro negocio.

Atención: Puede haber casos en los que **no existan grupos naturales**, o clusters que contengan una verdadera razón de ser. Si bien K-means siempre nos brindará "k clusters", quedará en nuestro criterio reconocer la utilidad de los mismos o bien revisar nuestras features y descartar las que no sirven o conseguir nuevas. También tener en cuenta que en este ejemplo estamos utilizando como medida de similitud entre features la **distancia Euclídea** pero podemos utilizar otras diversas funciones que podrían arrojar mejores resultados (como **Manhattan**, **Lavenshtein**, **Mahalanobis**, etc).

Hemos visto una descripción del algoritmo, aplicaciones y un ejemplo python paso a paso, que podrán descargar también desde los siguientes enlaces:

- Notebook Jupiter Online
- Descargar **archivo csv** y **notebook ejercicio K-means**
- **Visualizar** y descargar desde jbagnato Github

Si aún no viste el ejercicio de **Regresión Lineal**, **Regresión Logística** **paso a paso** ó **Arbol de decisión** puedes leerlos en español y seguir practicando



Los invito a escribirme sus comentarios al respecto y si tienen dudas o inconvenientes con el ejercicio. También pueden suscribirse al blog en donde intentaré escribir artículos de interés y aprendizaje de Machine Learning.



Suscríbete al Blog Aprende Machine Learning

El próximo artículo quincenal sobre Machine Learning te llegará al email para continuar estudiando!.

Email:

ENVIAR

El libro del Blog

En junio 2020 he alcanzado 1 millón de visitas al blog y he lanzado el borrador del [libro del Blog](#). Apreciaré mucho tu ayuda! Contiene Extras descargares como el “Lego Dataset” utilizado en el artículo de [Detección de Objetos](#). De pago pero también gratuito!

Libros sobre Machine Learning

Libros Relacionados

Algoritmos clasificación Definición Ejercicio Jupyter Notebook

Machine Learning Modelos Python recursos

QUÉ ES OVERFITTING Y UNDERFITTING Y CÓMO SOLUCIONARLO

INSTALAR AMBIENTE DE DESARROLLO PYTHON ANACONDA PARA APRENDIZAJE AUTOMÁTICO



73 comments



David Martínez · March 12, 2018

¡Muy interesante Juan!

El ejemplo práctico de los famosos que muestran rasgos comunes en sus personalidades es genial.

Me iría bien que explicaras de forma sencilla cómo instalar y configurar el entorno de desarrollo y cómo trabajar con él.

¡Un abrazo!

Reply



Na8 · March 12, 2018

Hola David, muchas gracias, siempre da ánimos que guste el artículo que tanto cuesta escribir! Aún te debo una pregunta anterior, mil disculpas!!!. Voy a intentar escribir un artículo tutorial sobre instalar el ambiente de desarrollo con Python y Anaconda - que es el más popular- y sirve para Windows, Mac, Linux y permite crear Jupyter Notebooks muy fácilmente. Ya tengo dos deudas contigo! Abrazo y felicitaciones también por tus proyectos, tu nueva web, muy buena!

Reply



David Martínez · March 13, 2018

Hola Juan!

No hay deudas pendientes, no te preocupes.

Me alegra que hayas mostrado interés por mis proyectos y que le hayas echado un vistazo a mi Portfolio.

Ahora mismo estoy metido en un proyecto de visualización 3D. He dejado un poco de lado el deep learning, pues me dió la sensación de que faltaba aún mucho por investigar. Aunque sigo teniendo interés por el tema y cómo evoluciona. Por otro lado, también tengo interés por la robótica y me he creado mi propio robot. Ahora mismo, estoy en la fase de añadir sensores al robot y que éste reaccione a los eventos.

Como ves estoy con muchas cosas.

¡Un abrazo Juan!

[Reply](#)*Na8* · March 13, 2018

Que bueno David!, me da curiosidad sobre tu robot. ¿En que lo estás haciendo? con Arduino? Raspberry pi? Saludos y seguimos en contacto!

[Reply](#)*David Martínez* · March 13, 2018

¡Hola Juan!

El robot es de la marca Allbot y yo he optado por la placa Arduino.

Me he descargado softwares por internet y, de momento, sólo mueve los 8 servos, pero quería añadir sensores de luz y de audio.

¡Un abrazo Juan!

[Reply](#)*Jared (@chinchillajared)* · March 20, 2018

Hola juan muchas gracias por el articulo sin duda me ah servido mucho en mi estudio de machine learning sin duda alguna, uno de los mejores artículos que eh visto en mi vida gracias por el aporte y espero que siga adelante con este maravilloso trabajo.

[Reply](#)*Na8* · March 20, 2018

Hola Jared, muchas gracias a ti por el comentario! Me da mucho ánimo para seguir escribiendo nuevos artículos. Si todo va bien, publicaré uno la semana que viene. Me alegra mucho saber que te ha ayudado mi post!. Espero que sigamos en contacto ya que me interesa mucho vuestra opinión o si tiene alguna inquietud en particular que pueda ayudar a resolver. Saludos!

[Reply](#)

 Canvas · April 1, 2018

Gran articulo, sobre un ejemplo muy bien escogido, y muy bien explicado.



Reply



Na8 · April 4, 2018

Muchas gracias Canvas, espero te gusten los otros ejemplos de los demás algoritmos. Intento que sean originales. Saludos

Reply



Osvaldo · April 16, 2018

Estimado Juan

Me pareció excelente tu articulo y me dio una idea practica de como aplicar esto en mi trabajo, ahora eso si tengo la duda en los últimos comandos que aplicaste tipo consulta. Me podría indicar como seria el código mas menos para ver en un listado todos los usuarios en forma ascendente, no solamente el que esta mas cerca del correspondiente centroide.

Saludos

Reply



Na8 · April 17, 2018

Hola Osvaldo, gracias por escribir y comentar, y me alegra que te sirva el artículo para aplicar en tu trabajo. Si bien no aparece como listar a los usuarios de cada cluster en el artículo, en el Jupyter Notebook sí que aparece. El código es el siguiente:

```
#miramos los usuarios del grupo 0
for index, row in copy.iterrows():
    if row["label"] == 0:
        print (row["usuario"], row["categoria"],row["label"])
```

Obviamente podrías iterar los 5 grupos de 0 a 4.

Espero que te sirva! Saludos y espero sigas visitando el Blog.

Reply



Osvaldo · April 18, 2018



Estimado Juan

Gracias por la pronta respuesta, me costo un poco la lógica de la programación, pero lo pude entender y aplicar que es lo que me interesa.

Eso si leyendo el código me encontré con la variable "group_referrer_index" revisando por separado en la consola jupyter salía lo siguiente :

0 True

1 True

2 False

3 True

4 True

5 False

6 False

7 True

107 True

108 False

109 False

110 True

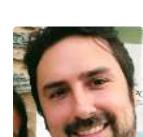
111 False

Name: label, Length: 112, dtype: bool

Por lo que entendí comparó copy['label'] == 1 con los valores de categoría y indicando que si es verdadero o falso.

Si tuvieras tiempo para explicar ese tema ya que no me interesa ser un mero copiador de códigos jaaj...

Reply



Na8 · April 20, 2018

Osvaldo, te comento esa parte del código:

Lo que hace es comparar con una de los clusters, (por ejemplo con la 1 si pones copy['label'] == 1) y Pandas devuelve un array con true y false en cada una de las 112 filas indicando si esa fila pertenece a cluster 1 true o False (es el listado que copiaste en tu mensaje).

Luego, en la siguiente linea hace:

```
group_referrals = copy[group_referrer_index]
```

Con lo cual filtra de todas las filas solamente las que estén en True, es decir, la del cluster 1 y se lo asigna a la variable group_referrals.

En el ejemplo de arriba, de las 112 filas deberían quedar 33.

Espero que te ayude esa explicación!
Saludos y lo que necesites, me escribes!

Reply



Osvaldo · April 25, 2018

Muchas gracias por la Respuesta :

La duda que me surgió al tratar de aplicar tu ejemplo a mi proyecto es que si se pueden agregar mas variables a la técnica, tu en el ejemplo incluiste "op","ex","ag", descartando la variable co.

Se puede en definitiva agregar una cuarta variable o mas.....

Saludos

Reply



Na8 · April 25, 2018

Hola Osvaldo, sí, claro que se puede agregar una cuarta, quinta o todas las features que necesites! Eso es lo bueno del machine learning: es extensible, flexible y allí radica todo su poder.

En el ejemplo que hice, tomé sólo 3 características de entradas (op, ex, ag) para poder hacer una gráfica en 3D y visualizarlo en el ejercicio. Pero se podía haber hecho con 5 o más -en cuyo caso habría que graficar de a partes-.

Sólo te diré que tengas cuidado al momento de seleccionar las características que vas a utilizar, pues a veces podemos empeorar los resultados por utilizar demasiadas. Hay que hacer un trabajo de selección de features y utilizar las que realmente aportan valor a nuestra máquina. También recordar que si tenemos 2 (o más) columnas que están correlacionadas, sólo utilizar una.

Por último te invito a que me escribas cuando puedas para contarme más sobre tu proyecto y sobre tus avances! Espero que puedas lograr buenos resultados!

Saludos

Reply



Kevin Puscán · December 6, 2018

Estimado Juan, excelente articulo, felicitaciones.

En el caso de que desee aplicar k-means para clasificar imágenes, cada pixel de la imagen tendría que ser un



feature, verdad? De ser así cual sería la mejor estructura para la información del pixel, bastaría con convertir el hexadecimal a decimal o existe alguna manera más recomendable?

Saludos

Reply



Na8 · December 19, 2018

Hola Kevin, gracias por comentar. Realmente no se me había ocurrido utilizar k-means para clasificar imágenes puesto que ya conocía la manera de [clasificar imágenes con Redes Neuronales \(CNN\)](#). Pero no parece una mala idea. De hecho a partir de tu comentario he revisado en google y hay algunos papers al respecto. Aquí te dejo unos enlaces interesantes que encontré:

- [Scalable Deep Learning for Image Classification with K-Means and SVM](#)
- [K-Means Clustering Tutorial](#)

Como recomendable te sugeriría que hicieras clasificación con CNN, puedes leer mi último artículo sobre el tema! [Cómo funcionan las CNN y un ejercicio práctico en Python para clasificar imágenes](#)

Saludos.

Reply



Kevin Puscán · December 27, 2018

Muchas gracias por tu respuesta Juan, revisaré la documentación que me has proporcionado, nuevamente muchas gracias.

Saludos.



Sebastian marcos · May 2, 2018

Hola Juan, te hago una consulta a ver si me podes orientar, tengo una base de datos en la cual tengo que buscar entre dos tablas una relación entre los

campos, pero desconocemos cual podria ser esa relacion, existe algun algoritmo para esto?

Reply



Na8 · May 8, 2018

Hola Sebastian, mira, pues si justamente tienes que hacer minería de datos, el algoritmo de k-means creo que es adecuado, pues es un algoritmo sin supervisión (con lo cual no necesitas etiquetas). Deberías experimentar cruzando varios datos de tus columnas de las dos tablas e ir probando el algoritmo, encontrar el valor K adecuado y ver si las relaciones que genera K-means te sirven. Si tienes problemas o no entiendes algo, escríbeme e intentaré ayudarte!

Saludos

Reply



Osvaldo · June 4, 2018

Hola Juan

Resultaría meter un set de datos nuevos a evaluar como en una matriz, y tratarlo como dataframe, quiero hacer una pequeña modificacion para evaluar varios datos a la vez.

Se me ocurría meterlo como matriz en un archivo SCV con su correspondiente formato :

```
dataframe2 = pd.read_csv(r"datos nuevos.csv")
```

```
new_labels = kmeans.predict(dataframe2)
print(new_labels)
```

Algo asi.....

Reply



Na8 · June 5, 2018

Hola Osvaldo, sinceramente no termino de comprender tu consulta. Si puedes escríbeme de nuevo para ver si te puedo ayudar. Saludos!

Reply



waflessnet · July 21, 2018

Hola ! Increíble tutorial !! Soy un novato, este no es mi rubro, pero me quedo muy claro !! muchas gracias por compartir un caso tan practico así pude entender como funciona y también pude aplicarlo!!

Saludos

Reply



Na8 · July 24, 2018

Hola muchas gracias por escribir y participar! Siempre me carga las pilas estos comentarios!! gracias y espero que leas los próximos artículos que se vienen

Reply



Jose Mario Uzcategui · August 17, 2018

Muy buena tu página y excelente las explicaciones. Gracias por compartir tus conocimientos. Me gustaría saber como usar datos en tiempo real; por ejemplo de un sensor de temperatura, procesarlos, graficarlos y generar una salida de alarma.

Reply



Na8 · August 17, 2018

Hola Jose Mario, gracias por escribir!. Me parece interesante tu propuesta de hacer sistemas que funcionen en tiempo real con inteligencia artificial. Intentaré escribir un artículo usando un sensor de temperatura y una red neuronal podría hacer predicciones con visualizaciones incluidas 😊

Si quieres contacta conmigo por Twitter y me comentas con más detalle.

Saludos!

Reply



Guillermo · September 20, 2018

Excelente blog de ML...tengo este año siguiendo el tema y este ha sido el Blog que mas dudas me ha eliminado...
Estoy haciendo los ejercicios en Python y utilizo Orange para hacerlos de forma gráfica. Particularmente, en este ejercicio, Orange me da la opción óptima para K=2 pero tu lo tienes en 5...aparte del "codo" tienes algún otro criterio? cual seria la diferencia para no tomar 2 y si tomar 5?

Reply



Na8 · September 25, 2018

Hola Guillermo, muchas gracias por escribir y me ha gustado mucho tu comentario! Te cuento que no conocía Orange, pero lo intentaré probar en estos días. Para determinar el mejor valor de k, puedes utilizar muchos, diversos criterios, te dejo este enlace: [Determining number of clusters](#) y también este: [Efficient estimation of the number of clusters](#) y: [Learning the k in k-means](#).

Ten en cuenta que muchas veces puede que tenga que ver con una decisión de "negocio" el tomar un valor u otro, por ejemplo para mi ejercicio yo preferí tomar 5 grupos en vez de 2 para que quede mejor explicada la solución, más visual. Pero puede que sea mejor solución tomar 2 clusters, "más óptimo" a nivel de minimizar la varianza entre los dos grupos, en comparación con 5. Sin embargo, para mi "negocio" particular, sigo eligiendo 5 grupos 😊

Espero que se entienda mi respuesta y te ayude! Saludos y seguimos en contacto!

Reply



lesly · December 3, 2018

gracias por los detalles! muy practico en verdad, lo usare como ejemplo con mis alumnos también.

Reply



Na8 · December 3, 2018

Hola Lesly, puedes usarlo libremente! Saludos y espero que sigas visitando el blog!

Reply



Daniel · January 14, 2019

Excelente artículo: claro y al grano.
Ahí va mi pregunta: ¿cuántos datos y features consideras necesario por cluster?
Tengo menos de 40 datos y bastantes features (8). ¿Alguna idea? ¿Es operativo?
Gracias y enhorabuena por el post.

Reply



Jose Martinez Heras · January 15, 2019

Lo bueno del K-Means es que fue inventado antes del Big Data, cuando la cantidad de datos era bastante pequeña. Así que puede funcionar bien en tu problema, especialmente con un número pequeño de grupos.

Para que funcione mejor, puedes probar primero a reducir la dimensionalidad para pasar de 8 features a menos features. Por ejemplo, puedes usar PCA configurándolo para que conserve el 90% de la varianza. Así te va a funcionar bastante mejor.

Una última recomendación, sería que escalases los datos para que cada feature tenga el mismo rango. Esto es importante porque los K-Means dependen mucho de la distancia que se use. Y la distancia es muy sensible a la escala de los features. Si no los escalas, K-Means va a asignar mas importancia a los features con mayor magnitud.

Suerte!

Reply



Daniel · January 15, 2019

Gracias por la pronta respuesta.

En efecto, pensaba hacer el análisis de componentes principales (que capture entre 80% y 90% de la varianza).
<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

El escalamiento:

<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>

Una pregunta más. Al evaluar cuántos clusters utilizar usas una medida ('score') que no entiendo muy bien. ¿Es el error cuadrático medio? ¿Medido sobre qué? En sklearn pone esto:

"Opposite of the value of X on the K-means objective".

Gracias y ánimo con el blog, está muy bien.

Saludos.

Reply



idelviso · January 17, 2019

Hola,

Me interesaría comparar un listado de frases consigo misma. He conseguido un vector de la siguiente forma:

frase 1 (3,1,0,3,...,0)

frase 2 (0,3,1,3,...,0)

Donde 0 es que no se parece nada a la frase que ocupa la posición del vector y 3 que es muy parecida.

Lo malo es que es un vector muy grande y me gustaría agruparlo en parecidos.

¿Crees que sería adecuada esta forma de agrupación para agrupar las frases más parecidas? o por el contrario debería utilizar otra.

También me da problemas que no puedo saber de primera mano cuantos grupos necesito hacer y debería ser auto agrupable.

Pd. es para hacer una autoclasificación de textos.

Gracias

Reply



Na8 · January 18, 2019

Hola idelviso, mira creo que deberías probar en el campo de NLP ya que por lo que me dices es un problema de resolver clasificación de textos. Si puedes lee mi artículo con la [teoría de Procesamiento del Lenguaje natural](#) y luego deberías buscar alguna solución usando word2vec ó de [clasiificación automática de textos](#).

Puede que mi próximo artículo sea sobre NLP con deep learning y aborde este asunto usando vectores de palabras en vez del bag of word. Te mantendré al tanto. Saludos

Reply



idelviso · January 18, 2019

Hola, Precisamente con word2vec es como he conseguido el vector que te comentaba, que ya tiene los pesos que me interesan.

Voy a probar lo que me comentas y te diré.

Gracias 😊

[Reply](#)

Na8 · January 18, 2019

Jajaja, vale, no te había entendido! Pues me dices y si no, le damos una vuelta a ver si se nos ocurren alternativas

[Reply](#)

Angel · February 13, 2019

Hola,

Tengo una duda acerca de la gráfica de Elbow Curve, ya que no entiendo bien que nos quieren decir los valores del eje y (Score).

Un saludo!

[Reply](#)

Na8 · February 14, 2019

Hola Angel, el score que calcula k-means con sklearn es la suma de las distancias de los puntos de cada cluster a su centroid. Y lo devuelve en valor negativo.

En el caso que hubiera K=1 un solo grupo, ese valor será muy grande, pues será la suma de TODOS los puntos al centro.

En el caso extremo de que K=140, tendremos un cluster por cada punto, y su distancia será cero.

Entonces buscaremos un punto que nos de equilibrio entre la cantidad de clusters y la distancia media de los puntos de ese grupo a su centro.

Saludos!

[Reply](#)

nelson cardona · February 19, 2019

hola

Estoy trabajando en un proyecto social en lugar bastante complejo en Colombia, quiero aplicar todo este articulo en la comunidad. En este momento la duda que me surge es de como hago para sacar los datos.

ej:

3gerardpique,"34.297953","28.148819","41.948819","29.370315","9.841575","37.

0945","7"

como se generan los números.

muchas gracias, espero su colaboracion.

Reply



Na8 · February 26, 2019

Hola Nelson Cardona, te cuento que los valores provenían de un proyecto mío que actualmente no está online. Era una web que calculaba los valores Big Five "OCEAN" de las cuentas de Twitter. Deberías investigar si puedes desarrollar tu mismo un algoritmo o buscar si en la web encuentras algún servicio que te provea esas valoraciones. Si estos días encuentro algún recurso, te vuelvo a escribir. Saludos.

PD: Si no encuentras nada de nada, vuelve a contactar conmigo y veo si puedo rescatar algo de ese viejo proyecto mío.

Reply



Ivan Montes · May 9, 2019

Hola juan,

me gusta mucho tu blog, es algo que me dedico a leer todos los días, estudiando, aprendiendo y aplicando.

quería consultarte algo y espero que puedas darme una guía sobre qué camino tomar, actualmente estoy tratando de categorizar productos, es una tarea bastante dura porque tengo 7 millones de productos que quiero categorizar, actualmente tenía planeado hacerlo por la imagen, hice un modelo de ML que categoriza imágenes y las imágenes tienen un ID que es el del producto, luego hago otro Modelo los lista y actualiza la categoría en base de datos, pero siento que podría ser más simple, que podría ser mejor.

¿Qué otra recomendación me das para poder categorizar esa cantidad de productos, evitando una cantidad enorme de errores como los tengo actualmente?

Reply



Na8 · May 16, 2019

Hola Ivan, muchas gracias por escribirme. El caso que cuentas parece muy interesante. ¿Me podrías escribir por email (ó al formulario) para que desde esa vía de comunicación nos podamos

pasar más detalles? A ver si juntos se nos ocurre una mejor solución! Saludos.

Reply



Andrea_SG · June 25, 2019

Hola Ivan, excelente articulo!! gracias por la explicación, como puedo ver el archivo csv para trabajar con tu codigo ??

Ya que yo ahora mismo estoy trabajando sobre lo mismo con un problema de ubicación de bahias para entrega de productos de cadenas de suministro.

Saludos amigo.

Reply



Na8 · June 25, 2019

Hola Andrea, el archivo está enlazado al final del artículo, luego de las conclusiones y también está disponible en mi github. Saludos

Reply



Andrea_SG · June 27, 2019

Saludos! Gracias amigo.

Reply



JUAN · September 16, 2019

Hola Juan, me encanta tu blog y voy a recomendárselo a todos mis conocidos interesados en este mundillo. Quería preguntarte...

Como ves la viabilidad de un proyecto de clusterización con k-means pero dentro de una iteración que permita ir añadiendo features y PCA de forma que cada vez la identificación de los clusters sea cada vez mas precisa pero pudiendo usar esa identificación para recomendación desde la primera iteracion.

Imagino que debe ser algo parecido a lo que hace amazon o google perfilando al cliente a grosso modo al principio con recomendaciones generalistas y cuanto mas interactua contigo mas afina.

Esto que te comento lo recomiendas con estos algoritmos comentados o

recomiendas otro modelo???

Te vuelvo a dar la enhorabuena por tu trabajo en este blog!

Reply



Na8 · September 21, 2019

Hola Juan, gracias por escribir. Te cuento que está bien lo que propones y describes. Te comento algunos temas que se me ocurren que deberás tener en cuenta para implementar tu solución:

- usar pipeline de sklearn
- mantener tu objeto modelo guardado en archivo (y probablemente en memoria, en donde esté ejecutando)
- Podrás hacer "fit" con las nuevas muestras e ir mejorando el modelo poco a poco
- Investiga el tema "cold start" en sistemas de recomendación, para usuarios nuevos

Por ahora eso!

Saludos!

Reply



JUAN · September 22, 2019

Muy amable por responder, me pongo a ello!!!!
Gracias y un saludo!

Reply



Miguel Angel Esparza Calero (@EsparzaCalero) · September 26, 2019

Donde obtener el DataSet completo ?

Reply



Juan · October 2, 2019

Hola Juan

Me atrevo a escribirte porque somos varios los que estamos viendo tu blog y nos pasa lo mismo con k-means, y es que siempre algo nos falla a la hora de



introducir más de tres features, quitando por supuesto las gráficas que dejarían de poderse ejecutar y dejando sólo las gráficas que si se pueden seguir ejecutando como la del elbow, por ejemplo....

Nos podrías enseñar cómo quedaría tu código, para poder trabajar con muchas features por favor??.

Muchas gracias por tu trabajo , nos tienes a todos envenenados con el ML. !!

Reply



damian · October 25, 2019

estimado: muy, muy, muy útil tu tutorial, de lo mejor que encontré en la web.
Excelente.

estoy dando mis primeros pasos en el machine learning y gracias a tu código
pude ir armando/modificando/creando/destruyendo/copiando/etc/etc mi
propio modelo de clustering.

necesito hacerte una consulta técnica: cómo es el código para dar salida
(imprimir) a los puntos que rodean a cada cluster?

en mi caso tengo 6 centroides, con equis cantidad de puntos a su rededor
¿cómo haría para obtener la lista de cada unos de esos puntos, de todos los
centroides?

ej:

label0 = 1,2,3

label1=4,5,6,7,8,9,10,11

labelN=12

Desde ya muchas gracias por tu ayuda

Reply



Esteban Manuel Sánchez García · May 14, 2020

Buenas Juan,

Lo primero, ¡muy buen blog, enhorabuena!

Tras esto preguntarte que porque has cogido las variables de "op", "ex", "ag"
cuando, (pese a que es poco), son las que más correlacionadas están que es

justo lo contrario que propones en la sección “Datos de Entrada para K-Means”



co

ex

ag



```
op 1.000000 0.204787 -0.508838 -0.662599 -0.238146 0.270123
co 0.204787 1.000000 -0.425614 -0.431095 0.180469 0.346416
ex -0.508838 -0.425614 1.000000 0.497006 -0.118141 -0.291869
ag -0.662599 -0.431095 0.497006 1.000000 0.003082 -0.381874
ne -0.238146 0.180469 -0.118141 0.003082 1.000000 0.134665
wordcount 0.270123 0.346416 -0.291869 -0.381874 0.134665 1.000000
```

Es el dataframe que me da a mi de correlación al hacer “dataframe.corr”.

Un saludo,
Esteban Sánchez

Reply



Na8 · May 23, 2020

Hola Esteban gracias por escribir!. Es muy buena tu observación. En la tabla de correlación que enviaste veo que la correlación más alta es la de “ag” con “op” que tienen -0.66 (inversa) si bien es alta, aún está alejada de ser un 100% por lo tanto no hay una relación muy fuerte. Si tuviéramos valores por encima del 0.8 o más sí que podría ser mejor descartar una de esas variables.

Saludos!

Reply



Estuardo Luna · May 23, 2020

Hola Juan, me encantó tu explicación. Solo tengo una duda como calcularías en este caso el Dunn Index? para medir el perfomance del modelo. Gracias

Reply



Na8 · May 23, 2020

Hola, gracias por el comentario. Debo decir que no conocía el Dunn Index. Al ser un algoritmo no supervisado es difícil de evaluar. En mi caso las veces que lo usamos se fue contrastando con casos reales para ver si tenía sentido y viendo si aplicaban o no los conjuntos generados. Siempre que se pueda, ayuda mucho hacer gráficas (si es necesario haciendo reducción de dimensión).

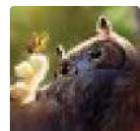
[Reply](#)

Estuardo Luna · May 24, 2020

Gracias Juan por tu pronta respuesta, intenté implementar el dunn index pero el resultado no se como interpretarlo ya que es: 0.060218600643665146

Asimismo como implementarías una matriz de confusión a tu ejemplo? es posible hacerlo o implementarlo?

Gracias.

[Reply](#)

kunkrox · May 23, 2020

Hola Juan! veo que el post tiene su tiempito, igualmente por si lo lees quisiera hacerte alguna consulta.

Recién estoy empezando en esto, disculpa si mi consulta no tiene mucho sentido.

Como es un algoritmo sin supervisar, entiendo que por mas que no estén etiquetados los datos trabaja con ellos.

mi consulta es como llega a la conclusion de que personalidad o que rubro utiliza ?

O sea, para que los resultados no sean un desastre tengo igualmente que ver y direccionarlo.

En este caso veo muchas columnas en el archivo CSV que no comprendo, son numeros flotantes, en la columna AG entiendo que seria años pero que otros parametros está utilizando para la predicción (op,co,ex,etc)?

Muchas Gracias.

Saludos.

[Reply](#)

Na8 · May 23, 2020

Hola gracias por escribir! Te contesto:

Las columnas op, co, ex, ag, ex se refiere a una puntuación sobre 5 rasgos de la personalidad de cada cuenta twitter analizada, según una metodología de psicología.

La idea de kmeans y de los algoritmos no supervisados no es guiarlos! Todo lo contrario, la idea es que el algoritmo solito logre agruparlos por características de "cercanía". Pero sin direccionar, porque la gracia es justamente ver que nuestro sentido común de humanos a veces no es capaz de ver relaciones complejas u ocultas

que los modelos matemáticos sí que logran desentrañar.

Saludos!

Reply



César Brochero · May 25, 2020

Hola Juan!

Excelente el ejercicio, soy bastante nuevo en el tema pero la verdad que me ha resultado muy fácil de seguirlo. Te felicito. Te hago consulta referente a la posibilidad de exportar el resultado (clusters) a un csv para poderlo consumir con una herramienta de BI.

Cómo podría adaptar el siguiente comando para exportar a un csv el resultado.

```
for index, row in copy.iterrows():
    print (row["usuario"], row["categoria"],row["label"])
```

Agradezco mucho tu ayuda..

Saludos

Reply



Na8 · May 31, 2020

Hola César, sólo deberías hacer el to_csv al dataframe de pandas.
En el ejemplo que pusiste, debería ser
copy.to_csv("nombre_archivo.csv")
Saludos!

Reply



mauricio · November 7, 2020

amigo siempre me causa curiosidad el titulo.. “aprende machine learning antes que sea tarde” me lo explicas porfavor? sin duda los mejores post 😊

Reply



Na8 · November 7, 2020

Hola Mauricio, es una broma refiriéndose a que es mejor que nosotros aprendamos a dominar a las máquinas, antes de que ellas

sean las que dominen el mundo 😊

Reply



Daniel · December 4, 2020

Hola Juan! me encantó la claridad de la explicación y el ejemplo paso a paso, era lo que estaba buscando y finalmente lo encontré. Gracias por tomarte el tiempo de escribirlo y brindarlo a la comunidad!

Reply



Na8 · December 5, 2020

Hola Daniel, muchas gracias! Saludos

Reply



Anderson Reyes · January 5, 2021

Hola! Saludos desde Nicaragua!

Felicitaciones por tan buen trabajo!

Tengo una duda respecto a la prueba de las cinco grandes personalidades.

Cómo realizaste el análisis? Tienes algún post relacionado con ese tema? Tienes algún enlace/libro/documento guía para entererlo mejor? Estoy empezando en todo este mundo y me es fascinante tu procesamiento de datos.

Espero tu respuesta.

Un abrazo!

Reply



José David Ríos Cristancho · August 19, 2021

hola que pena quiero saber como haces para que las graficas se vean mas lejos, me refiero a que digamos en el eje x vaya de 0-100 en vez de 0 a 60

muchas gracias

Reply



Na8 · November 29

Hola gracias por escribir!.

Pues se hace zoom en matplotlib usando los métodos de `set_xlim()` o `set_ylm()`.

O por ejemplo con `ax.set(xlim=[0,100], ylim=[0,60])`

Saludos!

Reply



Alan · November 24

mejor explicado imposible! mil gracias

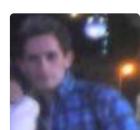
Reply



Na8 · November 29

Muchas gracias! un saludo

Reply



Andrés Paz · December 15

Hola !!! genial que compartas este tipo de conocimiento. Mi duda es por ejemplo si cada clúster es un conjunto (A,B,C) definida por los centroides y en cada ocasión que quiera clasificar a un nuevo cliente en un conjunto o paquete el algoritmo me va a generar nuevos centroides aleatorios por lo cual en cada ejecución de cada cliente tendría grupos diferentes (D,E,F) , de tal forma que la clasificación de #davidguetta no sería en los mismos conjuntos que el cliente #andrespaz obteniendo como resultado una clasificación ambigua ¿Cómo podría solucionar esto? ¿guardando la primera clasificación generada y esa clasificación usarla para los nuevos clientes evitando la nueva generación de centroides? Espero puedas comprender mi duda, saludos.

Reply

 wilfer echavarria · May 18

es muy buen material me ha servido mucho y mas que esta en español y muy bien explicado

[Reply](#)

Na8 · May 18

Muchas gracias! Un saludo

[Reply](#)

Leave a Reply

Enter your comment here...

Visita nuestra Guía de Aprendizaje

Buscar

 Search ...

Contacto

Suscripción

Recibe los artículos de Aprende Machine Learning en tu casilla de correo. Cada 15 días y sin Spam!

Email: Your email address here

[ENVIAR](#)



Proudly powered by WordPress | Theme: Eighties by Justin Kopepasah.

