FinalProyect

July 15, 2019

1 0. Introducción

A través del sitio de kaggle se obtuvo una base de datos con información de 6,820 películas lanzadas de 1986 a 2016. Cada filme cuenta con 9 variables cualitativas y 7 variables cuantitativas. Cada variable está definida de la siguiente forma.

company: compañía productora de la película
country: país de origen
director: director
genre: género principal de la cinta
name: nombre del filme
rating: clasificación de la pel?cula (G, R, PG, etc.). Notar que esta clasificación no es uniforme ya que se realiza dependiendo el país de origen y hay algunas películas sin clasificar (NOT RATED)
star: protagonista
writer: escritor
released: fecha de estreno (formato año-mes-día)
budget: se refiere al presupuesto de la película. Notar que algunas películas fueron realizadas sin presupuesto por lo que el valor de la variable es cero
gross: monto recaudado por la cinta (ingreso total antes de costos)
runtime: duración del filme en minutos
score: calificación promedio de la película (de 1 a 10) basado en las calificaciones de los usuarios de IMDb
votes: número de usuarios que votaron
year: año de lanzamiento
profit: ganancia total del filme (se calculó manualmente restando de gross el budget)

```
In [1]: import pandas as pd
        import numpy as np
        import sklearn as sk
        import seaborn as sns
        import matplotlib.pyplot as plt
        from factor_analyzer import FactorAnalyzer
        from sklearn.cluster import KMeans
        %matplotlib inline
```

```
In [2]: #Aquí cargamos la base
        movies_df =  pd.read_csv("Movies.csv",encoding='latin1')
```

```
#Creamos la variable profit
profit=movies_df.gross-movies_df.budget
movies_df['profit']=profit
#movies_df.head()
```

2 1. Análisis Descriptivo

2.1 1.1 Análisis Cualitativo

Primero analizamos algunas estadísticas principales de la base de 6,820 películas con value_counts() para las variables categóricas.

La base histórica tiene información de los lanzamientos de películas desde 1986 hasta 2016. Universal Pictures, Warner Bros y Paramount Pictures son las 3 principales compañías productoras.El género más popular es la comedia (30% del total de la base), seguido de Drama (21%) y Acción (20%) y las clasificaciones de películas más comunes fueron R (Reservado para mayores de edad) con el 50%, luego PG-13 con el 30% (Personas en General mayores a 13 años) y PG con el 14% (Apto para todo público).

En cuanto a datos demográficos, Estados Unidos es indiscutiblemente el país con más cintas concentrando el 71% del total seguido de Reino Unido con el 10% y Francia con el 4%. Sabiendo esto, resulta natural que los 6 directores y protagonistas más repetidos son de origen estadounidense. Sin embargo, esta situación no se cumple del todo en los escritores de películas, pues Luc Besson es de origen francés y William Shakespeare era inglés. Aunque todos son de género masculino.

```
In [3]: #Con el method info obtenemos el nombre de las columnas y su tipo de dato asignado por
        movies_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6820 entries, 0 to 6819
Data columns (total 16 columns):
company     6820 non-null object
country     6820 non-null object
director    6820 non-null object
genre       6820 non-null object
name        6820 non-null object
rating      6820 non-null object
star        6820 non-null object
writer      6820 non-null object
released    6820 non-null object
budget      6820 non-null int64
gross       6820 non-null int64
runtime     6820 non-null int64
score       6820 non-null float64
votes       6820 non-null int64
year        6820 non-null int64
profit      6820 non-null int64
dtypes: float64(1), int64(6), object(9)
memory usage: 852.6+ KB
```

```
In [4]: movies_df.company.value_counts().reset_index().head(5)
```

Out[4]:

	index	company
0	Universal Pictures	302
1	Warner Bros.	294
2	Paramount Pictures	259
3	Twentieth Century Fox Film Corporation	205
4	New Line Cinema	172

```
In [5]: movies_df.country.value_counts().reset_index().head(5)
```

Out[5]:

	index	country
0	USA	4872
1	UK	698
2	France	283
3	Canada	150
4	Germany	134

```
In [6]: movies_df.genre.value_counts().reset_index().head(5)
```

Out[6]:

	index	genre
0	Comedy	2080
1	Drama	1444
2	Action	1331
3	Crime	522
4	Adventure	392

```
In [7]: movies_df.rating.value_counts().reset_index().head(5)
```

Out[7]:

	index	rating
0	R	3392
1	PG-13	1995
2	PG	951
3	NOT RATED	308
4	G	147

```
In [8]: movies_df.director.value_counts().reset_index().head(10)
```

Out[8]:

	index	director
0	Woody Allen	33
1	Clint Eastwood	24
2	Steven Spielberg	21
3	Steven Soderbergh	21
4	Ron Howard	20
5	Ridley Scott	19
6	Joel Schumacher	19
7	Barry Levinson	18
8	Spike Lee	18
9	Oliver Stone	17

```
In [9]: movies_df.star.value_counts().reset_index().head(10)
```

```
Out[9]:                index  star
        0       Nicolas Cage    42
        1     Robert De Niro    38
        2  Denzel Washington    36
        3          Tom Hanks    35
        4       Bruce Willis    33
        5        Johnny Depp    32
        6         Tom Cruise    27
        7       Adam Sandler    27
        8        Ben Stiller    27
        9        John Cusack    26
```

```
In [10]: movies_df.writer.value_counts().reset_index().head(10)
```

```
Out[10]:                  index  writer
         0          Woody Allen      32
         1           Luc Besson      25
         2         Stephen King      22
         3          John Hughes      18
         4          David Mamet      14
         5  William Shakespeare      14
         6      Pedro Almodóvar      13
         7          Tyler Perry      12
         8           Wes Craven      12
         9            Joel Coen      11
```

2.2 1.2 Análisis Cuantitativo

Ahora, realizamos un análisis cuantitativo de las variables númericas con describe(). La duración promedio de las películas es aproximadamente de 1 hora y 47 minutos; se observa una correlación positiva de mediana intensidad con la calificación o score alcanzado por las cintas lo cual sugiere que las películas con mayor puntaje suelen ser de larga duración. La media del score de una película es de 6.3 con un intervalo que va de 5.3 a 7.3 por su desviación estándar. De igual forma, el score está medianamente correlacionado con el número de votos sobre una película. Es decir, cuando más usuarios califican una pel?cula, mayor suele ser el puntaje que la cinta alcanza.

Por otro lado, también notamos que el budget, gross y profit tienen mucha dispersión entre las películas dado que hay muchos montos que corresponden a valores extremos respecto a la media. Por ejemplo, el capital promedio invertido en una cinta es de 24.5M de dólares pero su desviación estándar es de 37M. Finalmente, se encontró que entre mayor presupuesto asignado tuvo un filme, mayor fue su ganancia total recaudada. Sin embargo, no se observa una correlación lineal positiva fuerte entre el presupuesto y el profit (ganancia después de costos).

```
In [11]: #Con describe calculamos las estadísticas para las variables numéricas
         movies_df.describe()
```

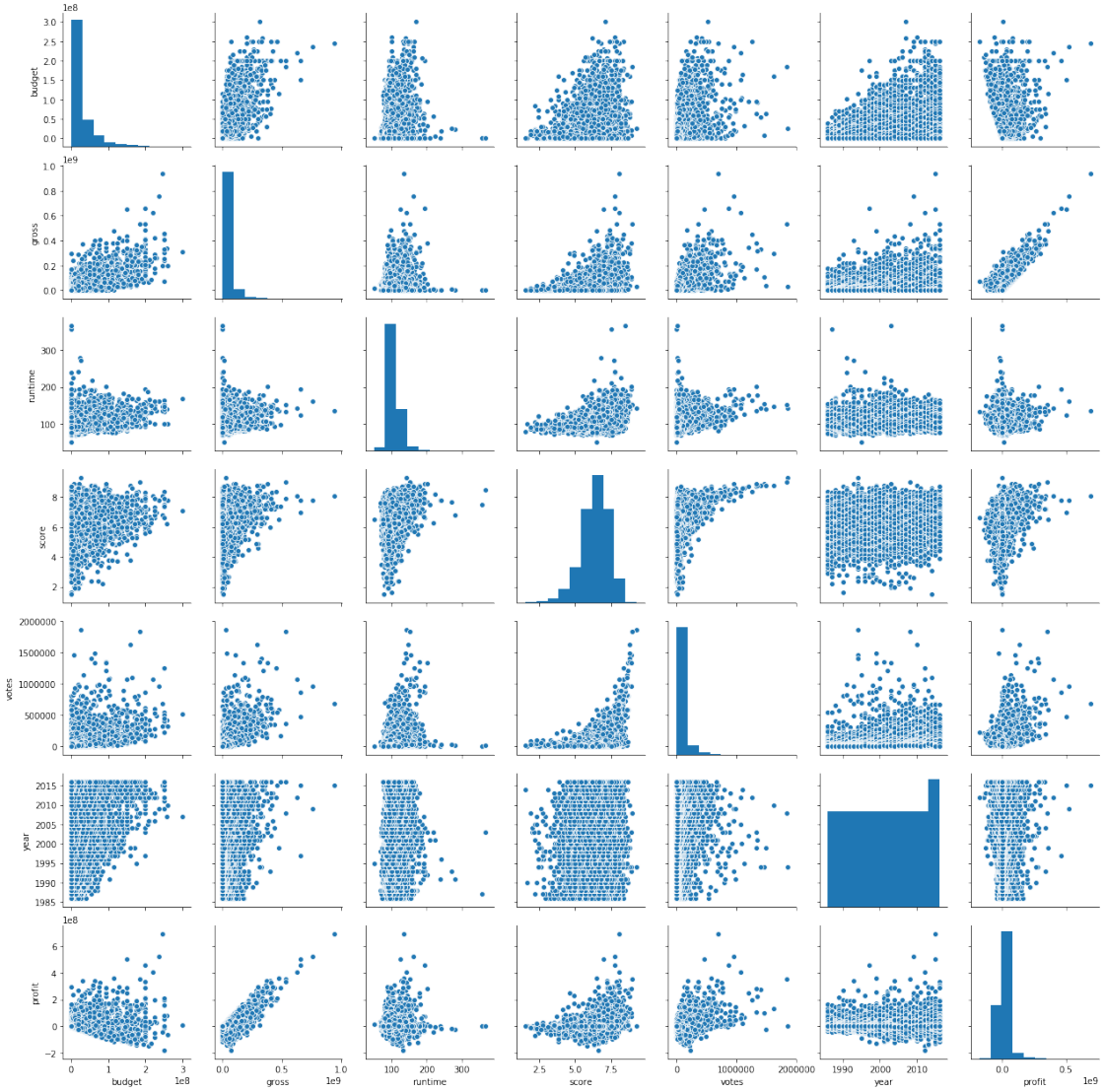
```
Out[11]:              budget         gross     runtime        score         votes  \
         count  6.820000e+03  6.820000e+03  6820.00000  6820.000000  6.820000e+03
```

```
mean   2.458113e+07  3.349783e+07   106.55132     6.374897  7.121952e+04
std    3.702254e+07  5.819760e+07    18.02818     1.003142  1.305176e+05
min    0.000000e+00  7.000000e+01    50.00000     1.500000  2.700000e+01
25%    0.000000e+00  1.515839e+06    95.00000     5.800000  7.665250e+03
50%    1.100000e+07  1.213568e+07   102.00000     6.400000  2.589250e+04
75%    3.200000e+07  4.006534e+07   115.00000     7.100000  7.581225e+04
max    3.000000e+08  9.366622e+08   366.00000     9.300000  1.861666e+06
```

```
              year        profit
count  6820.000000  6.820000e+03
mean   2001.000293  8.916700e+06
std       8.944501  4.109262e+07
min    1986.000000 -1.769219e+08
25%    1993.000000 -5.290186e+06
50%    2001.000000  9.187625e+05
75%    2009.000000  1.404740e+07
max    2016.000000  6.916622e+08
```

In [12]:
```
#Aquí agregamos la matriz con los histogramas de cada variables en la diagonal y
#los diagramas de dispersion de cada variable vs el resto
sns.pairplot(movies_df)
#Referencia de seaborn: https://seaborn.pydata.org/generated/seaborn.pairplot.html
```

Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x7f71b14b7978>
```



In [13]:
```
#Aquí calculamos la matriz de correlación de las variables poniendo en tonos rojos la
#y con tonos azules las correlaciones más débiles

cmap = cmap=sns.diverging_palette(5, 250, as_cmap=True)

def magnify():
    return [dict(selector="th",
                 props=[("font-size", "7pt")]),
            dict(selector="td",
                 props=[('padding', "0em 0em")]),
            dict(selector="th:hover",
                 props=[("font-size", "12pt")]),
            dict(selector="tr:hover td:hover",
```

```
                    props=[('max-width', '200px'),
                           ('font-size', '12pt')])
]

corr = movies_df.corr()
corr.style.background_gradient(cmap='coolwarm')

#Referencia para la matriz de correlación https://datascience.stackexchange.com/quest
```

Out[13]: <pandas.io.formats.style.Styler at 0x7f71aa7d3390>

3 2.0 Muestra de Datos

Para los objetivos de análisis de la segunda parte del proyecto que incluyen el análisis factorización y clusterización de k-medias, se decidió reducir la base a una muestra de 36 películas (con 4 de los filmes más populares por cada uno de los 8 géneros de clasificación) dado que la original contaba con demasiadas observaciones: 6,820 cintas. Se extrajo un arreglo con las variables numéricas y no se calcula la variable profit pues está en función de Budget y Gross.

In [14]:
```
#Aquí cargamos los datos de la muestra
movies_sample_df = pd.read_csv("MoviesSample.csv",encoding='latin1')
```

3.1 2.1. Análisis de factores

Se realizó un análisis factorial con el método de Máxima Verosimilitud. Para ello, previamente se asume que nuestra muestra seleccionada de películas "X" tiene una distribución normal y se estandarizaron las variables cuantitativas de la muestra.

In [15]:
```
#Crearemos un dataframe con las variables cuantitativas para poder realizar el anális
MoviesC_sample = movies_sample_df[['budget','gross','runtime','score','votes','year']]
#Ahora estandarizaremos las variables cuantitativas de la base original
MoviesCstd=(MoviesC_sample-MoviesC_sample.mean())/MoviesC_sample.std()
MoviesCstd.index = movies_sample_df.name
MoviesCstd.columns = ['budget','gross','runtime','score','votes','year']
#Mostramos cómo quedó la muestra de películas
MoviesCstd
```

Out[15]:

name	budget	gross	runtime	\
Leap!	-0.482414	-0.734391	-1.311420	
Resident Evil: The Final Chapter	-0.299758	-0.617070	-0.336879	
Día del atentado	-0.208430	-0.574906	1.070792	
Hidden Figures	-0.573741	0.571649	0.745945	
Assassin's Creed	1.252814	-0.385118	0.096251	
Sing	0.339536	1.413446	-0.282737	
Rogue One	2.622731	3.596976	1.070792	
Manchester by the Sea	-0.875123	-0.443078	1.287357	
Jackie	-0.865990	-0.724403	-0.715867	

Moana	1.709453	1.233559	-0.336879
Fantastic Beasts and Where to Find Them	2.257420	1.110815	1.070792
Doctor Strange	1.983437	1.099177	0.096251
Ouija: Origin of Evil	-0.865990	-0.549735	-0.770008
Miss Peregrine's Home for Peculiar Children	0.978831	-0.113293	0.745945
Snowden	-0.299758	-0.660787	1.124934
La bruja de Blair	-0.939053	-0.667796	-1.311420
Sully	0.065553	0.202144	-0.932432
Mi papá es un gato	-0.482414	-0.677049	-1.419702
Bad Moms	-0.665069	0.103639	-0.715867
La vida secreta de tus mascotas	0.339536	2.231121	-1.419702
The Conjuring 2	-0.299758	0.013685	1.124934
Me Before You	-0.665069	-0.371781	-0.391020
Neighbors 2: Sorority Rising	-0.391086	-0.379730	-1.148996
Revenant: El Renacido	1.435470	0.690537	2.316040
Daddy's Home	-0.117103	0.413011	-0.932432
Spotlight	-0.665069	-0.465087	0.800087
Paranormal Activity: The Ghost Dimension	-0.847725	-0.688200	-1.365561
The Martian	0.942300	1.064084	1.666346
Escobar	-0.719866	-0.839912	0.366957
Crimenes ocultos	-0.117103	-0.830746	1.287357
Still Alice	-0.939053	-0.684412	-0.661726
Whiplash	-0.970104	-0.731630	-0.336879
The Invasion	0.430864	-0.715123	-0.770008
The Butterfly Effect	-0.792928	-0.357658	-0.012031
The Arrival	-0.573741	-0.723655	0.096251
The Manhattan Project	-0.701600	-0.808282	0.204533

name	score	votes	year
Leap!	-0.057805	-0.921875	0.490904
Resident Evil: The Final Chapter	-1.306395	-0.606893	0.490904
Día del atentado	0.566490	-0.649499	0.490904
Hidden Figures	0.982687	-0.194384	0.490904
Assassin's Creed	-0.994248	-0.119757	0.331692
Sing	0.254342	-0.455786	0.331692
Rogue One	1.086736	1.384023	0.331692
Manchester by the Sea	1.086736	0.050842	0.331692
Jackie	-0.057805	-0.655401	0.331692
Moana	0.774588	0.035405	0.331692
Fantastic Beasts and Where to Find Them	0.566490	0.753062	0.331692
Doctor Strange	0.670539	1.272821	0.331692
Ouija: Origin of Evil	-0.786149	-0.752325	0.331692
Miss Peregrine's Home for Peculiar Children	-0.161854	-0.228867	0.331692
Snowden	0.462441	-0.386937	0.331692
La bruja de Blair	-1.930690	-0.791031	0.331692
Sully	0.670539	0.034900	0.331692
Mi papá es un gato	-1.618543	-0.890502	0.331692

```
Bad Moms                                      -0.682100 -0.489569  0.331692
La vida secreta de tus mascotas               -0.265903 -0.117056  0.331692
The Conjuring 2                                0.566490 -0.002576  0.331692
Me Before You                                  0.566490 -0.155056  0.331692
Neighbors 2: Sorority Rising                  -1.202346 -0.443646  0.331692
Revenant: El Renacido                          1.190785  2.400500  0.331692
Daddy's Home                                  -0.786149 -0.511653  0.172480
Spotlight                                      1.294834  0.878399  0.172480
Paranormal Activity: The Ghost Dimension      -2.346887 -0.858630  0.172480
The Martian                                    1.190785  2.773389  0.172480
Escobar                                       -0.265903 -0.874443  0.172480
Crimenes ocultos                              -0.369953 -0.657169  0.172480
Still Alice                                    0.670539 -0.324236  0.172480
Whiplash                                       1.711031  2.279814  0.172480
The Invasion                                  -0.994248 -0.543415 -1.101218
The Butterfly Effect                           0.878637  1.525316 -1.578855
The Arrival                                   -0.578051 -0.804467 -2.852553
The Manhattan Project                         -0.786149 -0.953300 -4.444675
```

A partir de la prueba de hipótesis de Kaiser-Meyer-Olkin (KMO) se revisó que el análisis de factores fuese adecuado para nuestro conjunto de datos. KMO estima la proporción de varianza sobre todas las variables observadas. Los valores de la prueba KMO están entre 0 y 1. Un valor de KMO menor a 0.6 sugiere que es inadecuado utilizar análisis factorial sobre el dataset. Para el caso de nuestra muestra, el valor de la prueba es de 0.59 por lo que se decide que el análisis de factores es adecuado para el conjunto de datos.

```
In [16]: fa1 = FactorAnalyzer(rotation=None,n_factors=1,method='ml')

         fa1.fit(MoviesCstd)

         from factor_analyzer.factor_analyzer import calculate_kmo
         kmo_all,kmo_model=calculate_kmo(MoviesCstd)
         round(kmo_model,2)
```

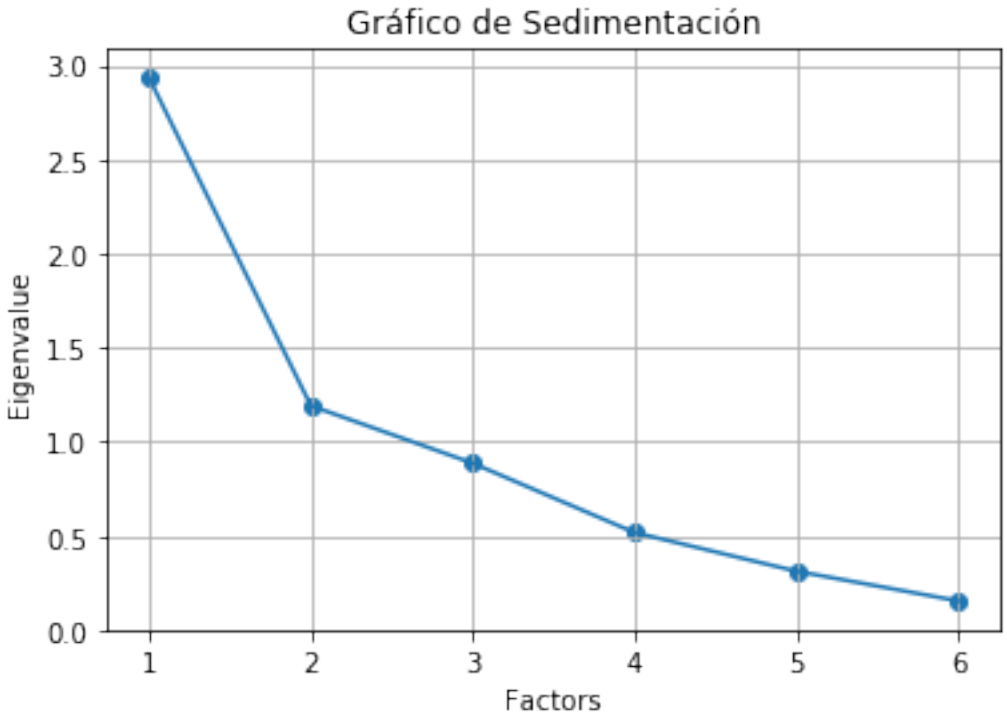
```
Out[16]: 0.59
```

A continuación se ejecutó un análisis con un factor mostrando los eigenvalores de las 6 variables numéricas. Este resultado se utiliza en conjunto con el gráfico de sedimentación para encontrar cuál es el número de factores que necesitamos tomar para el análisis. En este caso, se deben tomar 2 factores ya que es el número de eigenvalores mayores a 1 y además es donde se identifica el codo en el gráfico de sedimentación.

```
In [17]: fa1 = FactorAnalyzer(rotation=None,n_factors=1,method='ml')
         fa1.fit(MoviesCstd)
         ev, v = fa1.get_eigenvalues()
         ev
```

```
Out[17]: array([2.9334078 , 1.19220097, 0.8868652 , 0.51851482, 0.31328847,
                0.15572273])
```

```
In [18]: # Crea el gráfico de sedimentación usando matplotlib
         plt.scatter(range(1,MoviesCstd.shape[1]+1),ev)
         plt.plot(range(1,MoviesCstd.shape[1]+1),ev)
         plt.title('Gráfico de Sedimentación')
         plt.xlabel('Factors')
         plt.ylabel('Eigenvalue')
         plt.grid()
         plt.show()
```



Después ejecutamos un análisis con 2 factores sin rotar como lo sugería el análisis previo. Encontrando que la proporción de varianza acumulada es de 0.60 lo cual es suficiente para continuar con el análisis.

Respecto a las cargas factoriales, podemos ver que el primer factor está más influenciado por el presupuesto de la cinta y su calificación obtenida de forma positiva. Aunque estas mismas variables también tienen el peso más grande sobre el factor 2, el presupuesto ahora tiene una influencia negativa y la calificación sigue siendo positiva. La variable menos relevante es el año del lanzamiento de la película.

```
In [19]: fa2 = FactorAnalyzer(rotation=None,n_factors=2,method='ml')
         fa2.fit(MoviesCstd)
         fa2.get_factor_variance()

Out[19]: (array([2.70964445, 0.89604176]),
          array([0.45160741, 0.14934029]),
          array([0.45160741, 0.6009477 ]))
```

```
In [20]: fa2.loadings_,
         #Creamos un dataframe usando las cargas del análisis de 2 factores
         cargas = [('budget',0.79595406, 0.601217),
                 ('gross',0.69853914,-0.31200213),
                 ('runtime',0.61986979, 0.18802457),
                 ('score',0.79598224, 0.60117779),
                 ('votes',0.72614652, 0.1990978),
                 ('year',0.2074358, -0.02875128)]
         dffa2 = pd.DataFrame(cargas, columns =['variable','factor1','factor2'])
         dffa2
```

```
Out[20]:   variable   factor1   factor2
         0   budget  0.795954  0.601217
         1    gross  0.698539 -0.312002
         2  runtime  0.619870  0.188025
         3    score  0.795982  0.601178
         4    votes  0.726147  0.199098
         5     year  0.207436 -0.028751
```

Ahora haremos una rotación con el método varimax para facilitar la interpretación de los factores y graficaremos las variables utilizando las cargas factoriales como coordenadas.

El factor 1 tiene una carga extremadamente alta del score de la cinta, también el número de votos y la duración en minutos de la película tienen pesos relevantes por ello, bien podríamos nombrarlo "Factor de Estadísticas del Filme". En contraste, el factor 2 cuenta con alta influencia del presupuesto del filme y su monto total recaudado por lo que éste podría llamarse "Factor económico" además, claramente tiene una interpretación más sencilla. Nuevamente, el año de lanzamiento no figura como una variable con cargas importantes para ningún factor.

```
In [21]: vfa2 = FactorAnalyzer(rotation="varimax",n_factors=2,method='ml')
         vfa2.fit(MoviesCstd)
```

```
Out[21]: FactorAnalyzer(bounds=(0.005, 1), impute='median', is_corr_matrix=False,
                 method='ml', n_factors=2, rotation='varimax', rotation_kwargs={},
                 use_smc=True)
```

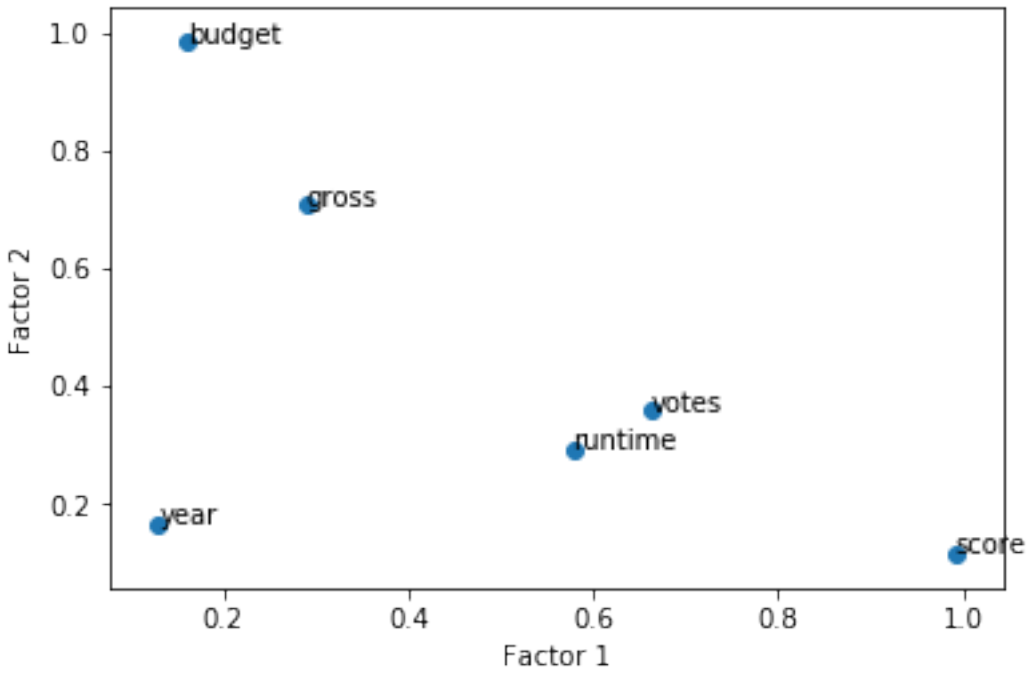
```
In [22]: vfa2.loadings_,
         #Creamos otro dataframe usando las cargas del análisis de 2 factores
         cargas = [('budget',0.15993481, 0.98459413),
                 ('gross',0.2893608,0.7082179),
                 ('runtime',0.57800573, 0.29240584),
                 ('score',0.99079535, 0.11544281),
                 ('votes',0.66248107, 0.35783729),
                 ('year',0.13008159, 0.16411894)]
         dfvfa2 = pd.DataFrame(cargas, columns =['variable','factor1','factor2'])
         dfvfa2
```

```
Out[22]:   variable   factor1   factor2
         0   budget  0.159935  0.984594
         1    gross  0.289361  0.708218
```

```
2  runtime  0.578006  0.292406
3    score  0.990795  0.115443
4    votes  0.662481  0.357837
5     year  0.130082  0.164119
```

In [23]:
```
#dfvfa2.plot(kind='scatter',x='factor1',y='factor2',color='red')
#plt.show()

factor1 = [0.159935, 0.289361, 0.578006, 0.990795, 0.662481, 0.130082]
factor2 = [0.984594, 0.708218, 0.292406, 0.115443, 0.357837, 0.164119]
variables = ['budget', 'gross', 'runtime', 'score', 'votes', 'year']

fig, ax = plt.subplots()
ax.scatter(factor1, factor2)

for i, txt in enumerate(variables):
    ax.annotate(txt, (factor1[i], factor2[i]))

ax.set_ylabel("Factor 2")
ax.set_xlabel("Factor 1")
```

Out[23]:
```
Text(0.5, 0, 'Factor 1')
```



4 3.1 K-Means Clustering

K-medias es un método de agrupamiento, que tiene como objetivo la partición de un conjunto de n observaciones en k grupos en el que cada observación pertenece al grupo cuyo valor medio es más cercano. Para este caso práctico, el algoritmo nos ayudará a agrupar las 36 películas de la muestra en k clusters con características similares.

Para determinar el número de grupos k, ajustaremos clusters de tamaño 1 a 15 para después decidir cuál es el k óptimo a partir del gráfico de codo con los scores generados en cada agrupamiento.

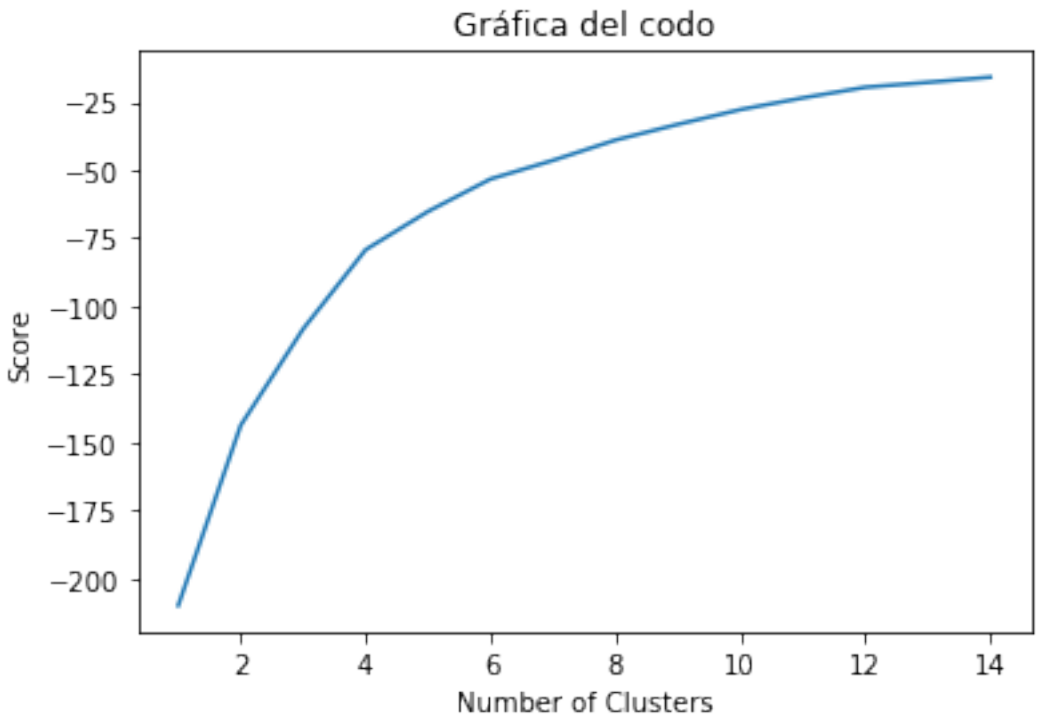
```
In [24]: #Cargamos nuevamente los datos
         movies_sample_df = pd.read_csv("MoviesSample.csv",encoding='latin1')
         MoviesC_sample = movies_sample_df[['budget','gross','runtime','score','votes','year']]
         MoviesC_sample.index = movies_sample_df.name
         MoviesCstd=(MoviesC_sample-MoviesC_sample.mean())/MoviesC_sample.std()
         MoviesCstd.index = movies_sample_df.name
         MoviesCstd.columns = ['budget','gross','runtime','score','votes','year']
         #MoviesCstd.head()
         #Referencia para clusterización: https://www.kaggle.com/sirpunch/k-means-clustering
```

```
In [25]: clusters = range(1, 15)
         kmeans = [KMeans(n_clusters=n) for n in clusters]
         score = [kmeans[i].fit(MoviesCstd).score(MoviesCstd) for i in range(len(kmeans))]
```

Los valores de los score indican qué tan lejanas están las observaciones del cluster center. Se busca mantener este valor cercano a 0, es decir, un valor positivo o negativo grande indican que el cluster center está lejos de las observaciones.

Our Elbow point is around cluster size of 5. We will use k=5 to further interpret our clustering result. I'm prefering this number for ease of interpretation in this demo. We can also pick a higher number like 9.

```
In [26]: plt.plot(clusters,score)
         plt.xlabel('Number of Clusters')
         plt.ylabel('Score')
         plt.title('Gráfica del codo')
         plt.show()
```



K-Means para k=4

Ahora procederemos a hacer la clusterización de los 4 grupos. Posteriormente, añadiremos las etiquetas de los números de grupo a nuestra base muestra original (sin los datos estandarizados) para identificarlos. Calculamos el tamaño de cada cluster e identificamos que existen 2 grupos grandes de 15 y 13 películas y un grupo muy pequeño con sólo 2 filmes, esto sucede porque los grupos buscan homogeneidad dentro de sus elementos sin importar con cuántos cuenta el grupo.

Luego, se muestran las cintas dentro de cada cluster para identificarlas.

```
In [27]: kmeans = KMeans(n_clusters=4)
         kmeans.fit(MoviesCstd)
         MoviesCstd['cluster'] = kmeans.labels_
         MoviesC_sample['cluster'] = kmeans.labels_
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  after removing the cwd from sys.path.
```

```
In [28]: tamanio_cluster = list(MoviesC_sample.groupby(['cluster']).count()['budget'].values)
         tamanio_cluster

Out[28]: [15, 6, 13, 2]
```

In [29]:
```
#Primer Grupo
MoviesC_sample[MoviesC_sample['cluster']==tamanio_cluster.index(sorted(tamanio_cluste
```

Out[29]:

name	budget	gross	runtime	score	votes	year	cluster
Leap!	30000000	12760985	89	6.8	9517	2017	0
Resident Evil: The Final Chapter	40000000	26830068	107	5.6	58140	2017	0
Assassin's Creed	125000000	54645723	115	5.9	133338	2016	0
Sing	75000000	270329045	108	7.1	81466	2016	0
Jackie	9000000	13958679	100	6.8	50652	2016	0
Ouija: Origin of Evil	9000000	34904885	99	6.1	35690	2016	0
La bruja de Blair	5000000	20747013	89	5.0	29715	2016	0
Sully	60000000	125070033	96	7.5	157212	2016	0
Mi papá es un gato	30000000	19637449	87	5.3	14360	2016	0
Bad Moms	20000000	113257297	100	6.2	76251	2016	0
La vida secreta de tus mascotas	75000000	368384330	87	6.6	133755	2016	0
Neighbors 2: Sorority Rising	35000000	55291815	92	5.7	83340	2016	0
Daddy's Home	50000000	150357137	96	6.1	72842	2015	0
Paranormal Activity: The Ghost Dimension	10000000	18300124	88	4.6	19280	2015	0
The Invasion	80000000	15071514	99	5.9	67939	2007	0

In [30]:
```
#Segundo Grupo
MoviesC_sample[MoviesC_sample['cluster']==tamanio_cluster.index(sorted(tamanio_cluste
```

Out[30]:

name	budget	gross	runtime	score	votes	year	cluster
Rogue One	200000000	532177324	133	7.9	157609	2016	1
Moana	150000000	248757044	107	7.6	56362	2016	1
Fantastic Beasts and Where to Find Them	180000000	234037575	133	7.4	117205	2016	1
Doctor Strange	165000000	232641920	115	7.5	168259	2016	1

```
Revenant: El Renacido                     135000000  183637894      156    8.0
The Martian                               108000000  228433663      144    8.0

                                           votes  year  cluster
name
Rogue One                                 365473  2016        1
Moana                                     157290  2016        1
Fantastic Beasts and Where to Find Them   268073  2016        1
Doctor Strange                            348307  2016        1
Revenant: El Renacido                     522384  2016        1
The Martian                               579946  2015        1
```

In [31]:
```
#Tercer Grupo
MoviesC_sample[MoviesC_sample['cluster']==tamanio_cluster.index(sorted(tamanio_cluste
```

Out[31]:
```
                                             budget      gross  runtime  \
name
Día del atentado                           45000000   31886361      133
Hidden Figures                             25000000  169380936      127
Manchester by the Sea                       8500000   47695120      137
Miss Peregrine's Home for Peculiar Children 110000000  87242834      127
Snowden                                    40000000   21587519      134
The Conjuring 2                            40000000  102470008      134
Me Before You                              20000000   56245075      106
Spotlight                                  20000000   45055776      128
Escobar                                    17000000     106869      120
Crimenes ocultos                           50000000    1206135      137
Still Alice                                 5000000   18754371      101
Whiplash                                    3300000   13092000      107
The Butterfly Effect                       13000000   57938693      113

                                           score   votes  year  cluster
name
Día del atentado                             7.4   51563  2017        2
Hidden Figures                               7.8  121818  2017        2
Manchester by the Sea                        7.9  159673  2016        2
Miss Peregrine's Home for Peculiar Children  6.7  116495  2016        2
Snowden                                      7.3   92094  2016        2
The Conjuring 2                              7.4  151427  2016        2
Me Before You                                7.4  127889  2016        2
Spotlight                                    8.1  287421  2015        2
Escobar                                      6.6   16839  2015        2
Crimenes ocultos                             6.5   50379  2015        2
Still Alice                                  7.5  101773  2015        2
Whiplash                                     8.5  503754  2015        2
The Butterfly Effect                         7.7  387284  2004        2
```

In [32]:
```
#Cuarto Grupo
MoviesC_sample[MoviesC_sample['cluster']==tamanio_cluster.index(sorted(tamanio_cluste
```

```
Out[32]:                         budget     gross  runtime  score  votes  year  \
         name
         The Arrival            25000000  14048372      115    6.3  27641  1996
         The Manhattan Project  18000000   3900000      117    6.1   4666  1986

                                cluster
         name
         The Arrival                  3
         The Manhattan Project        3
```

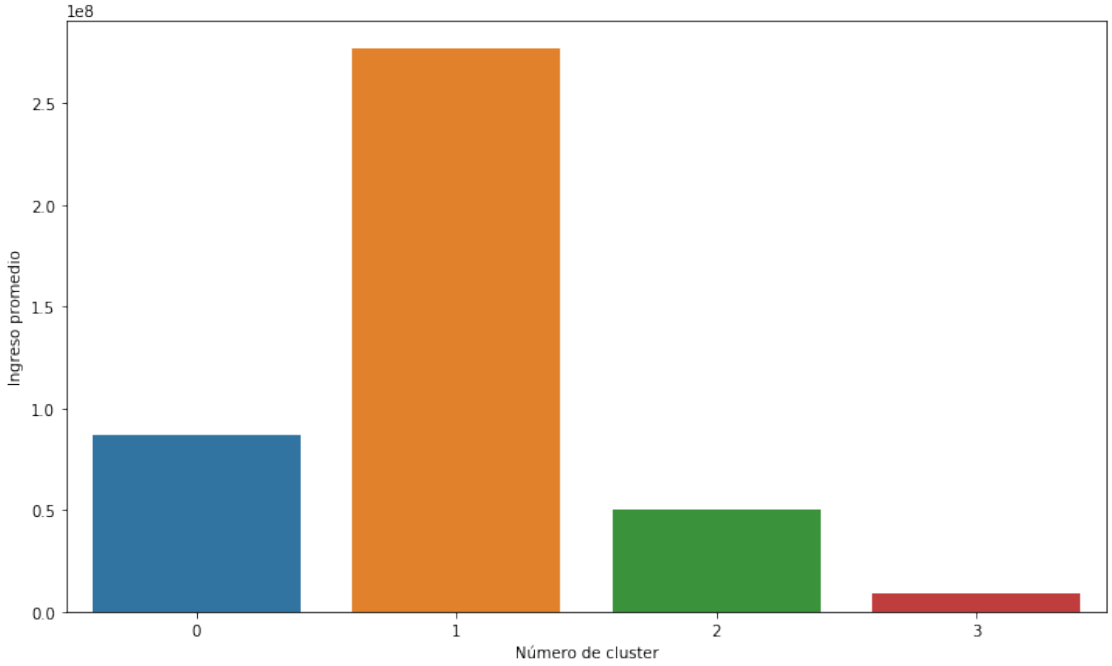
A continuación damos un vistazo a los centros de gravedad de los 4 grupos que obtuvimos para describirlos. Con ello, identificamos que el cluster 1 es el líder ya que promedia los presupuestos (156 M), ganancias (276 M), duraciones (2h:11min), score (7.7) y votos (373 k) más altos. El cluster 0 también destaca por tener largas duraciones (2h), un alto score (7.4) y el segundo lugar en ingresos totales (69 M) a pesar de ser tercer lugar en presupuesto (35 M). Finalmente, el cluster 3 se caracteriza por tener el budget, gross y votos promedio más bajos.

```
In [33]: MoviesC_sample.groupby(['cluster']).mean()
```

```
Out[33]:                budget         gross     runtime     score          votes  \
         cluster
         0        4.353333e+07  8.663641e+07   96.800000  6.080000   68233.133333
         1        1.563333e+08  2.766142e+08  131.333333  7.733333  373578.833333
         2        3.052308e+07  5.020475e+07  123.384615  7.446154  166800.692308
         3        2.150000e+07  8.974186e+06  116.000000  6.200000   16153.500000

                         year
         cluster
         0         2015.400000
         1         2015.833333
         2         2014.846154
         3         1991.000000
```

Finalmente, se toma la variable gross para hacer un barplot con el monto recaudado en promedio por cada cluster. Aquí notamos que el grupo con mayores ingresos promedio (276 M de dólares) cuenta con sólo 6 películas: "Rogue One", "Moana", "Fantastic Beasts and Where to Find Them", "Doctor Strange", "Revenant: El Renacido" y "The Martian". Los cluster 0 y 2 tienen un ingreso promedio muy parecido por alrededor de 70 M de dólares mientras que el cluster 3 con las 2 películas sólo promedia 9 M de dólares.

```
In [34]: plt.figure(figsize=(12,7))
         axis = sns.barplot(x=np.arange(0,4,1),y=MoviesC_sample.groupby(['cluster']).mean()['g
         x=axis.set_xlabel("Número de cluster")
         x=axis.set_ylabel("Ingreso promedio")
```


1e8
2.5
2.0
1.5
1.0
0.5
0.0
Ingreso promedio
0
1
2
3
Número de cluster