INFLUENCIA DEL PODER SOCIAL EN EL RENDIMIENTO DE LA NBA

INTRODUCCIÓN

Hoy en día, sabemos que los jugadores de las mayores ligas mundiales de cualquier deporte sufren influencias de los más diferentes lados que pueden afectar su rendimiento en pista y, por consecuencia, de sus equipos.

En este trabajo, será analizado cuales son las principales variables que pueden afectar el rendimiento de los equipos y jugadores en pista y cómo estas variables se correlacionan entre si a través de un análisis descriptivo utilizando algunas de las herramientas de data science y machine learning enseñadas en el Máster.

Las variables que serán analizadas serán las siguientes:

Valor Inmobiliario
De acuerdo con la ciudad del Equipo

Ranking ELO
Ranking/Indicador ELO

Wikipedia / Twitter
Accesos

INFLUENCIA DEL PODER SOCIAL EN EL RENDIMIENTO DE LA NBA

Indicadores
Desde Basketball-Reference

Público
Asistencia a los partidos

Salarios
Salarios y Patrocinadores

BASES DE DATOS

Para conseguir los datos de las variables arriba mencionadas, ha sido necesario descargar desde diversas fuentes de datos algunas informaciones de la temporada 2016-2017 de la NBA:

Fuente de Datos	Nombre Archivo	Filas	Resumen
Basketball-Reference	nba_2017_attendance.csv	30	Asistencia a las Arenas
Forbes	nba_2017_endorsements.csv	8	Mejores Jugadores
Forbes	nba_2017_team_valuation.csv	30	Todos Equipos
ESPN	nba_2017_salary.csv	450	Algunos Jugadores
NBA	nba_2017_pie.csv	468	Todos Jugadores
ESPN	nba_2017_real_plus_minus.csv	468	Todos Jugadores
Basketball-Reference	nba_2017_br.csv	468	Todos Jugadores
Five ThirtyEight	nba_2017_att_val_elo_win_housing.csv	420	Precios Residencia
Five ThirtyEight	nba_2017_elo.csv	30	Elo Ranking

Una de las dificultades que me encontrado en este proceso ha sido estructurar y limpiar los datos de fuentes distintas.

LIBRERÍAS

Pandas: Leer ficheros csv y mostrarlos en forma de dataframe

Statsmodels: Combinado con pandas, proporciona códigos más sencillos para análisis de modelos estadísticos

Matplotlib.pyplot: Librería de gráficos

Seaborn: Combinado con matplotlib, proporciona gráficos más visuales para resultados de modelos estadísticos

```
import pandas as pd
import statsmodels.api as sm
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt
import seaborn as sns
color = sns.color_palette()
%matplotlib inline
```

CARGA Y LIMPIEZA DE DATOS

Público

```
attendance_df = pd.read_csv("../data/nba_2017_attendance.csv");attendance_df.head()
```

	TEAM	GMS	PCT	TOTAL_MILLIONS	AVG_MILLIONS
0	Chicago Bulls	41	104	0.888882	0.021680
1	Dallas Mavericks	41	103	0.811366	0.019789
2	Sacramento Kings	41	101	0.721928	0.017608
3	Miami Heat	41	100	0.805400	0.019643
4	Toronto Raptors	41	100	0.813050	0.019830

Salarios y Patrocinadores de Jugadores

```
endorsement_df = pd.read_csv("../data/nba_2017_endorsements.csv");endorsement_df.head()
```

	NAME	TEAM	SALARY_MILLIONS	ENDORSEMENT_MILLIONS
0	LeBron James	Cleveland Cavaliers	31	55
1	Kevin Durant	Golden State Warriors	27	36
2	Stephen Curry	Golden State Warriors	12	35
3	James Harden	Houston Rockets	27	20
4	Russell Westbrook	Oklahoma City Thunder	27	15

Valoración de los Equipos

```
valuations_df = pd.read_csv("../data/nba_2017_team_valuations.csv");valuations_df.head()
```

	TEAM	VALUE_MILLIONS
0	New York Knicks	3300.0
1	Los Angeles Lakers	3000.0
2	Golden State Warriors	2600.0
3	Chicago Bulls	2500.0
4	Boston Celtics	2200.0

Salarios por Equipo y Posición

```
salary_df = pd.read_csv("../data/nba_2017_salary.csv");salary_df.head()
```

	NAME	POSITION	TEAM	SALARY
0	LeBron James	SF	Cleveland Cavaliers	30963450.0
1	Mike Conley	PG	Memphis Grizzlies	26540100.0
2	Al Horford	C	Boston Celtics	26540100.0
3	Dirk Nowitzki	PF	Dallas Mavericks	25000000.0
4	Carmelo Anthony	SF	New York Knicks	24559380.0

CARGA Y LIMPIEZA DE DATOS

PIE (Player Impact Estimate)

Mide la contribución estadística general de un jugador contra las estadísticas totales en los partidos que juega, siendo posible saber si su impacto en los partidos han sido positivo o negativo en relación al rendimiento del equipo.

```
pie_df = pd.read_csv("../data/nba_2017_pie.csv");pie_df.head()
```

	PLAYER	TEAM	AGE	GP	W	L	MIN	OFFRTG	DEFRTG	NETRTG	...	AST RATIO	OREB%	DREB%	REB%	TO
0	Russell Westbrook	OKC	28	81	46	35	34.6	107.9	104.6	3.3	...	23.4	5.3	27.9	16.7	
1	Boban Marjanovic	DET	28	35	16	19	8.4	104.3	102.4	1.9	...	5.1	16.6	31.3	23.9	
2	Demetrius Jackson	BOS	22	5	1	4	3.4	124.2	117.8	6.3	...	31.1	9.1	11.8	10.3	
3	Anthony Davis	NOP	24	75	31	44	36.1	104.2	102.5	1.7	...	7.3	6.7	26.9	17.0	
4	James Harden	HOU	27	81	54	27	36.4	113.6	107.3	6.3	...	27.6	3.5	21.2	12.3	

RPM (Real Plus-Minus)

Tratase de un indicador de ESPN que básicamente informa el mismo que el PIE pero también analiza los impactos positivos y negativos en términos ofensivos y defensivos de cada jugados

```
plus_minus_df = pd.read_csv("../data/nba_2017_real_plus_minus.csv");plus_minus_df.head()
```

	NAME	TEAM	GP	MPG	ORPM	DRPM	RPM	WINS
0	LeBron James, SF	CLE	74	37.8	6.49	1.93	8.42	20.43
1	Stephen Curry, PG	GS	79	33.4	7.27	0.14	7.41	18.80
2	Jimmy Butler, SG	CHI	76	37.0	4.82	1.80	6.62	17.35
3	Russell Westbrook, PG	OKC	81	34.6	6.74	-0.47	6.27	17.34
4	Draymond Green, PF	GS	76	32.5	1.55	5.59	7.14	16.84

Triples

Es el ranking del % de aciertos de tres puntos de cada jugador

```
br_stats_df = pd.read_csv("../data/nba_2017_br.csv");br_stats_df.head()
```

	Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	...	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PS
0	1	Russell Westbrook	PG	28	OKC	81	81	34.6	10.2	24.0	...	0.845	1.7	9.0	10.7	10.4	1.6	0.4	5.4	2.3	3
1	2	James Harden	PG	27	HOU	81	81	36.4	8.3	18.9	...	0.847	1.2	7.0	8.1	11.2	1.5	0.5	5.7	2.7	2
2	3	Isaiah Thomas	PG	27	BOS	76	76	33.8	9.0	19.4	...	0.909	0.6	2.1	2.7	5.9	0.9	0.2	2.8	2.2	2
3	4	Anthony Davis	C	23	NOP	75	75	36.1	10.3	20.3	...	0.802	2.3	9.5	11.8	2.1	1.3	2.2	2.4	2.2	2
4	5	DeMar DeRozan	SG	27	TOR	74	74	35.4	9.7	20.9	...	0.842	0.9	4.3	5.2	3.9	1.1	0.2	2.4	1.8	2

CARGA Y LIMPIEZA DE DATOS

Ranking ELO

El ELO es un indicador que, en forma de ranking, identifica la puntuación de cada equipo de la NBA basado en el valor acumulado de todos los jugadores de cada equipo.
Este indicador contiene la información de registros de victorias y derrotas, clasificando los equipos de acuerdo con la fuerza del oponente contra el que se juega.

```
elo_df = pd.read_csv("../data/nba_2017_elo.csv");elo_df.head()
```

	ELO	TEAM	CONF
0	1770	Golden State Warriors	West
1	1661	San Antonio Spurs	West
2	1636	Los Angeles Clippers	West
3	1617	Utah Jazz	West
4	1602	Houston Rockets	West

CORRELACIÓN ENTRE VARIABLES

Público ~ Valoración por Equipos

Empezando el análisis para mirar la correlación entre las variables, he unido el público total de los partidos y el valor de mercado de cada equipo.

```
attendance_valuation_df = attendance_df.merge(valuations_df, how="inner", on="TEAM")
```

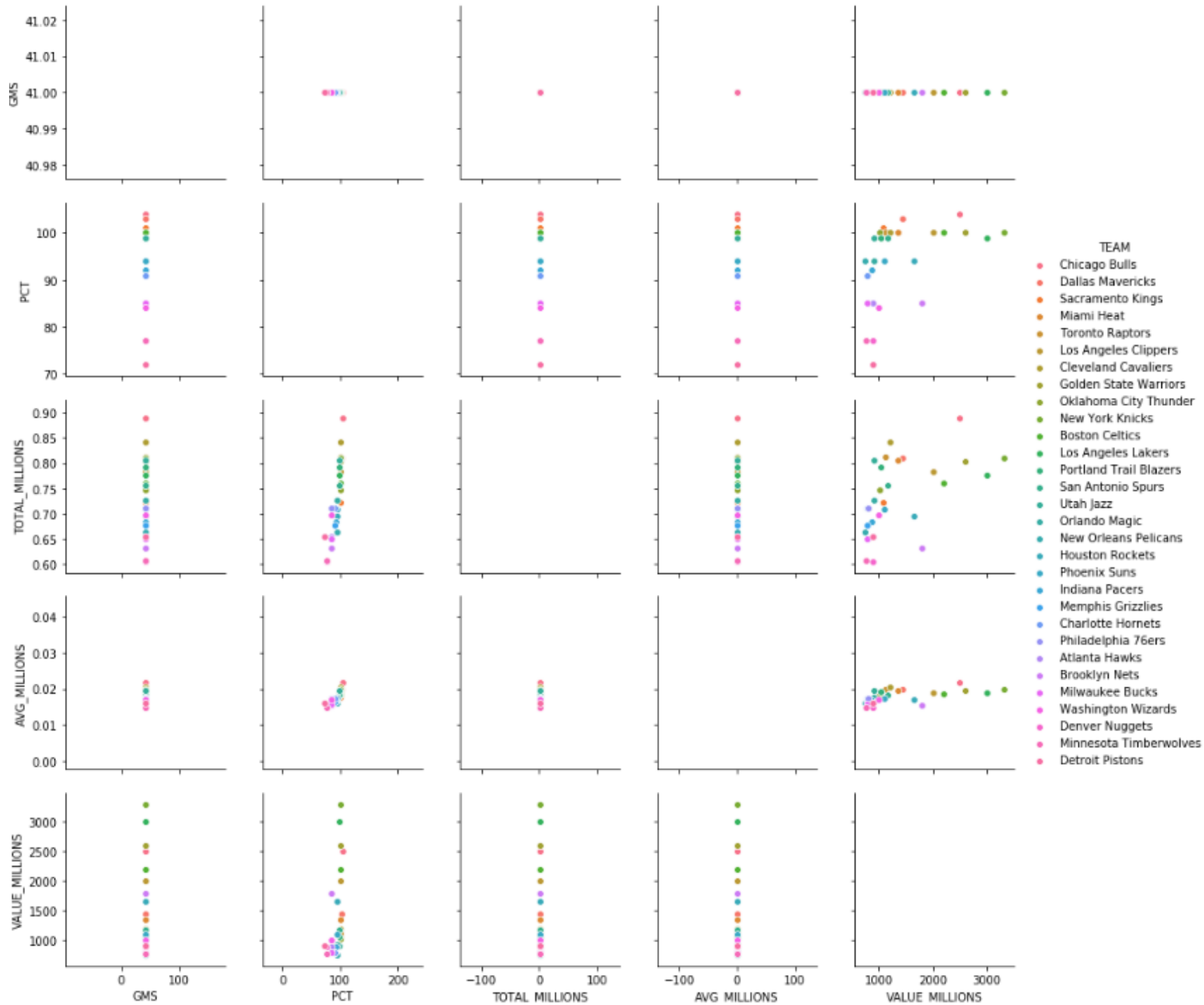
```
attendance_valuation_df.head()
```

	TEAM	GMS	PCT	TOTAL_MILLIONS	AVG_MILLIONS	VALUE_MILLIONS
0	Chicago Bulls	41	104	0.888882	0.021680	2500.0
1	Dallas Mavericks	41	103	0.811366	0.019789	1450.0
2	Sacramento Kings	41	101	0.721928	0.017608	1075.0
3	Miami Heat	41	100	0.805400	0.019643	1350.0
4	Toronto Raptors	41	100	0.813050	0.019830	1125.0

CORRELACIÓN ENTRE VARIABLES

Público ~ Valoración por Equipos

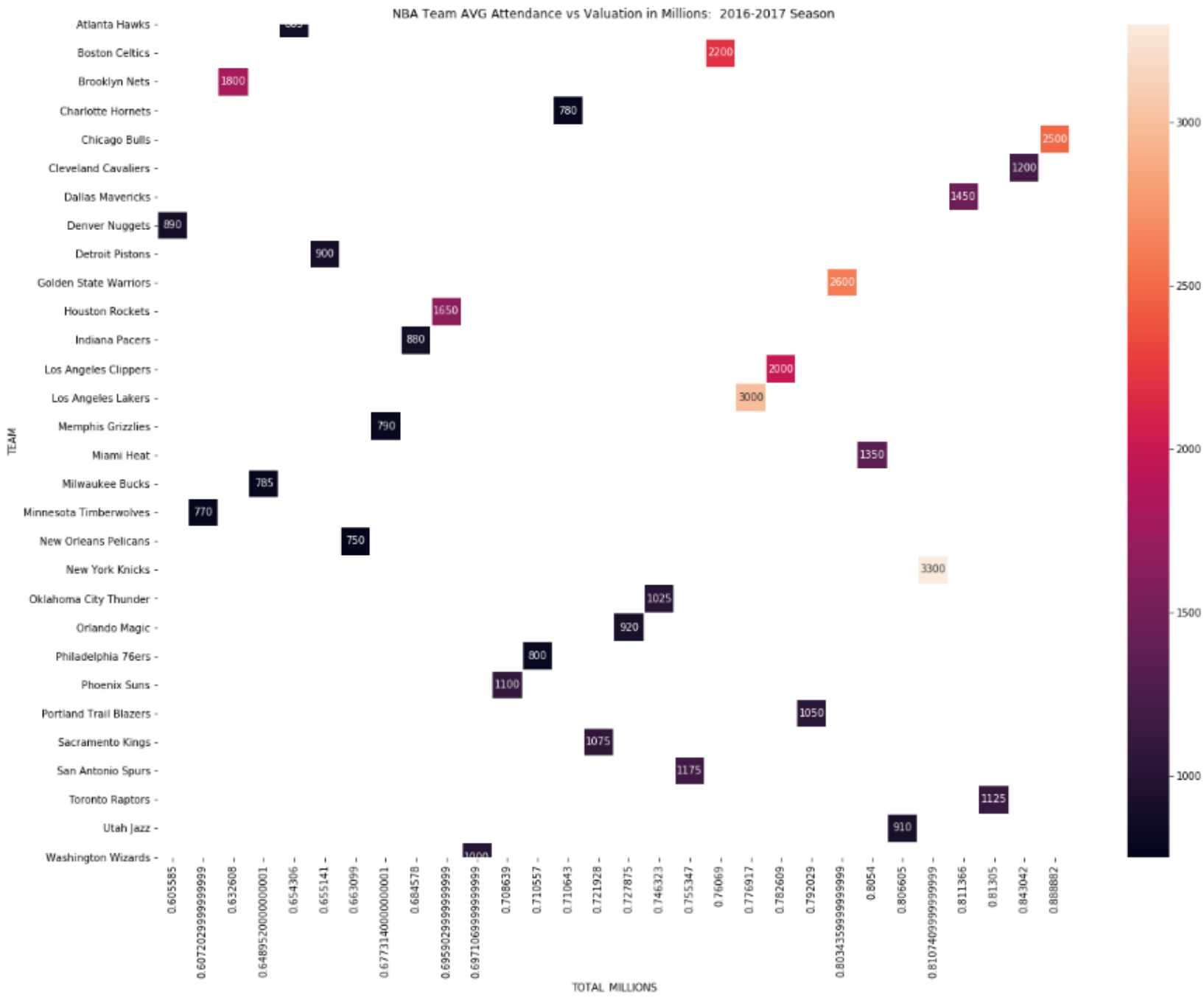
A través de un conjunto de gráficos de correlación de variables por equipos, se puede observar que hay una fuerte correlación lineal (positiva) entre las dos variables (AVG_MILLIONS ~ VALUE_MILLIONS), pues los gráficos presentan casi una línea recta entre los dos puntos cuando se combinan.



CORRELACIÓN ENTRE VARIABLES

Público ~ Valoración por Equipos

Remodelando la data para el gráfico de correlación de calor de Seaborn, se puede ver mejor como las variables se comportan por equipo.
El valor en cada cuadrado es el valor de cada equipo, en eje X el publico total y en el eje Y los equipos y el valor de cada equipo.
Se puede observar algunos outliers, como por ejemplo Brooklyn Nets, que esta evaluado en 1.8 billiones de dólares (que es un valor considerable) y tiene una de las asistencias más bajas entre todos los equipos.



REGRESIÓN LINEAL

Público ~ Valoración por Equipos

Con la intención de validar el modelo y saber el valor del R cuadrado referente a la correlación entre las variables analizadas, se aplica una regresión lineal (el paquete StatsModel permite un código sencillo):

```
results = smf.ols('VALUE_MILLIONS ~TOTAL_MILLIONS', data=attendance_valuation_df).fit()
```

```
print(results.summary())
```

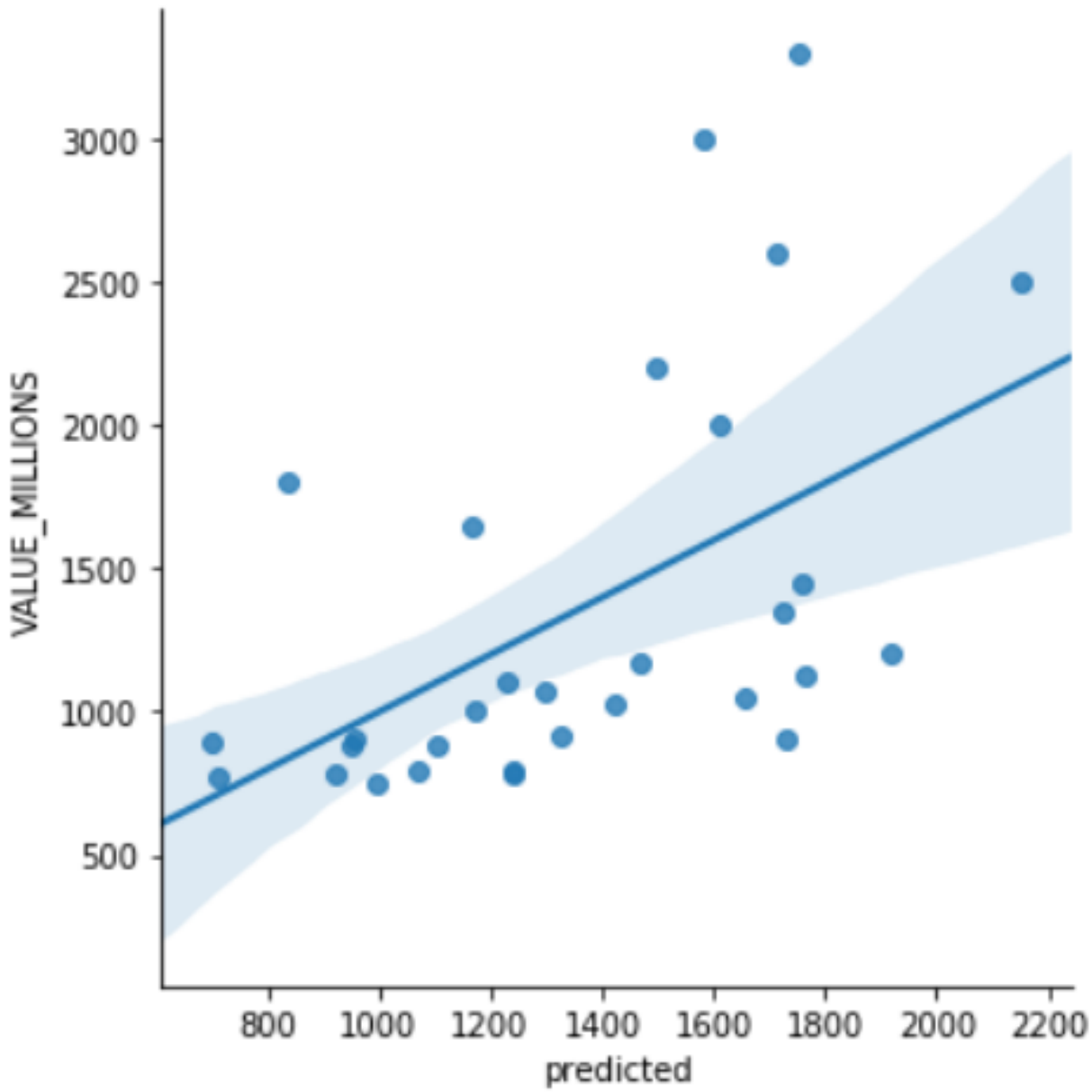
```
                            OLS Regression Results
==============================================================================
Dep. Variable:         VALUE_MILLIONS   R-squared:                       0.282
Model:                            OLS   Adj. R-squared:                  0.256
Method:                 Least Squares   F-statistic:                     10.98
Date:                Fri, 24 Jan 2020   Prob (F-statistic):           0.00255
Time:                        19:20:44   Log-Likelihood:                -234.04
No. Observations:                  30   AIC:                             472.1
Df Residuals:                      28   BIC:                             474.9
Df Model:                           1
Covariance Type:            nonrobust
==================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept      -2408.0191   1141.332     -2.110      0.044   -4745.931     -70.107
TOTAL_MILLIONS  5132.3400   1549.033      3.313      0.003    1959.290    8305.390
==============================================================================
Omnibus:                        7.692   Durbin-Watson:                   1.930
Prob(Omnibus):                  0.021   Jarque-Bera (JB):                6.355
Skew:                           1.100   Prob(JB):                       0.0417
Kurtosis:                       3.495   Cond. No.                         21.3
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

La imagen muestra el resultado de la regresión. El R cuadrado muestra que aproximadamente el 28 por ciento de la valoración puede explicarse por la asistencia de público, y el valor P de 0.044 cae dentro del rango de ser estadísticamente significativo.

REGRESIÓN LINEAL

Público ~ Valoración por Equipos

De forma visual, se puede notar como se ven los datos con la regresión lineal aplicada
Los puntos dentro del azul que están en el intervalo de confianza son los 28% de los equipos tienen una valoración explicada por su publico.



AÑADIENDO VARIABLES - ELO

Público ~ Valoración por Equipos ~ ELO

Con la intención de obtener un R cuadrado más significativo y testear las relaciones entre las variables, he decidido añadir el ranking ELO de cada equipo.

```
attendance_valuation_elo_df = attendance_valuation_df.merge(elo_df, how="inner", on="TEAM")
```

```
attendance_valuation_elo_df.head()
```

	TEAM	GMS	PCT	TOTAL_MILLIONS	AVG_MILLIONS	VALUE_MILLIONS	ELO	CONF
0	Chicago Bulls	41	104	0.888882	0.021680	2500.0	1519	East
1	Dallas Mavericks	41	103	0.811366	0.019789	1450.0	1420	West
2	Sacramento Kings	41	101	0.721928	0.017608	1075.0	1393	West
3	Miami Heat	41	100	0.805400	0.019643	1350.0	1569	East
4	Toronto Raptors	41	100	0.813050	0.019830	1125.0	1600	East

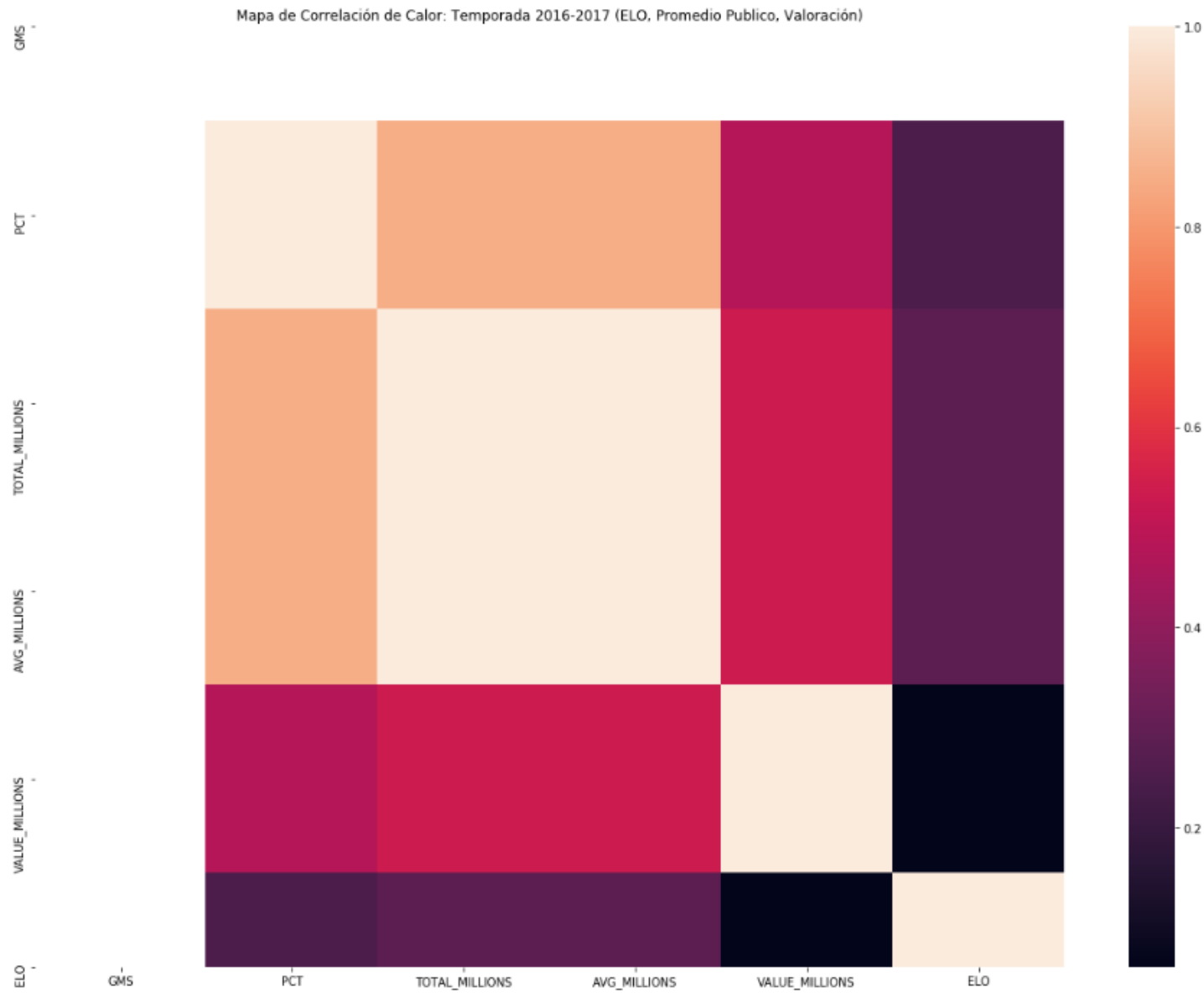
CORRELACIÓN ENTRE VARIABLES - ELO AÑADIDO

Público ~ Valoración por Equipos ~ ELO

Después de añadir la variable ELO al modelo, he añadido un nuevo mapa de calor de correlación. Hay algunas correlaciones positivas para examinar más de cerca. En el mapa de calor a continuación, cuanto más claro es el color, más altamente correlacionadas están las dos columnas. Si la matriz muestra el mismo valor en comparación con sí misma, entonces la correlación es 1, y el cuadrado es beige. En el caso de TOTAL y ELO, parece haber una correlación de 0.5.

```
corr_elo = attendance_valuation_elo_df.corr()
plt.subplots(figsize=(20,15))
ax = plt.axes()
ax.set_title("Mapa de Correlación de Calor: Temporada 2016-2017 (ELO, Promedio Publico, Valoración)")
sns.heatmap(corr_elo,
            xticklabels=corr_elo.columns.values,
            yticklabels=corr_elo.columns.values)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x254a9d39b08>
```



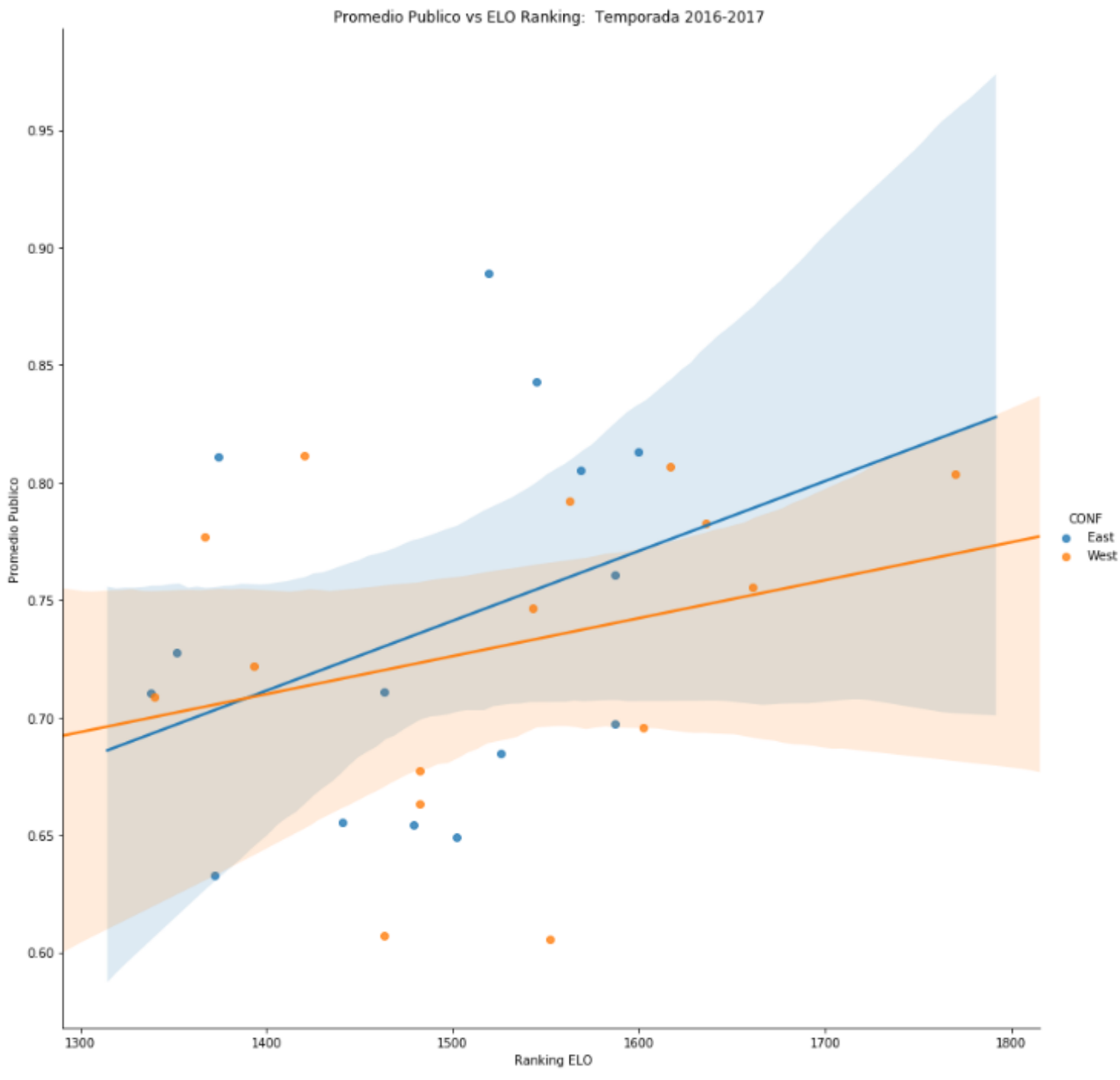
CORRELACIÓN ENTRE VARIABLES - ELO AÑADIDO

El Gráfico abajo muestra la correlacón entre ELO versus asistencia por conferencia. Parece haber una relación lineal débil entre lo bueno que es un equipo (ELO RANK) versus la asistencia. La gráfica a continuación colorea las gráficas de dispersión este y oeste por separado, junto con un intervalo de confianza. La relación lineal débil está representada por la línea recta que pasa por los puntos en el espacio X, Y.
Además se puede identificar que un pequeño cambio en el ranking ELO de los equipos de la conferencia este (Azul) impacta más que cambios en el ranking de los equipos del oeste.

```
ax = sns.lmplot(x="ELO", y="TOTAL_MILLIONS", data=attendance_valuation_elo_df, hue="CONF", size=12)
ax.set(xlabel=' Ranking ELO', ylabel='Promedio Publico', title="Promedio Publico vs ELO Ranking:  Temporada 2016-2017 ")
```

```
C:\Users\Usuario\Anaconda3\lib\site-packages\seaborn\regression.py:546: UserWarning: The `size` paramter has been renamed
  warnings.warn(msg, UserWarning)
```

```
<seaborn.axisgrid.FacetGrid at 0x254a9a4cb08>
```



REGRESIÓN LINEAL - ELO AÑADIDO

Público ~ Valoración por Equipos ~ ELO

La salida de la regresión con el ELO añadido muestra un R cuadrado de 8% y un valor P de 0.027, por lo que también hay una señal estadísticamente significativa aquí, pero es muy débil.

OLS Regression Results

Dep. Variable:	TOTAL_MILLIONS	R-squared:	0.082
Model:	OLS	Adj. R-squared:	0.049
Method:	Least Squares	F-statistic:	2.493
Date:	Fri, 24 Jan 2020	Prob (F-statistic):	0.126
Time:	19:21:56	Log-Likelihood:	37.584
No. Observations:	30	AIC:	-71.17
Df Residuals:	28	BIC:	-68.37
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.4378	0.188	2.334	0.027	0.053	0.822
ELO	0.0002	0.000	1.579	0.126	-5.84e-05	0.000

Omnibus:	0.340	Durbin-Watson:	0.652
Prob(Omnibus):	0.844	Jarque-Bera (JB):	0.507
Skew:	0.087	Prob(JB):	0.776
Kurtosis:	2.387	Cond. No.	2.17e+04

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.17e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

AÑADIENDO VARIABLES - VALOR INMOBILIÁRIO

Como se ha notado que el ELO no aporta mucho al modelo, pero con la misma intención de analizar variables conjuntas y obtener un R cuadrado un valor *p* más significativo, se ha añadido al modelo la variable del valor inmobiliario de las residencias donde los equipos están ubicados.
El valor inmobiliario es representado por el valor medio de las viviendas en las ciudades donde están ubicadas los equipos.

```
val_housing_win_df = pd.read_csv("../data/nba_2017_att_val_elo_win_housing.csv");val_housing_win_df.head()
```

	TEAM	GMS	PCT_ATTENDANCE	WINNING_SEASON	TOTAL_ATTENDANCE_MILLIONS	VALUE_MILLIONS	ELO	CONF	COUNTY	MEDIAN_HOME
0	Chicago Bulls	41	104	1	0.888882	2500	1519	East	Cook	
1	Dallas Mavericks	41	103	0	0.811366	1450	1420	West	Dallas	
2	Sacramento Kings	41	101	0	0.721928	1075	1393	West	Sacremento	
3	Miami Heat	41	100	1	0.805400	1350	1569	East	Miami-Dade	
4	Toronto Raptors	41	100	1	0.813050	1125	1600	East	York-County	

```
val_housing_win_df.columns
```

```
Index(['TEAM', 'GMS', 'PCT_ATTENDANCE', 'WINNING_SEASON',
       'TOTAL_ATTENDANCE_MILLIONS', 'VALUE_MILLIONS', 'ELO', 'CONF', 'COUNTY',
       'MEDIAN_HOME_PRICE_COUNTY_MILLIONS', 'COUNTY_POPULATION_MILLIONS'],
      dtype='object')
```

REGRESIÓN LINEAL - VALOR INMOBILIÁRIO AÑADIDO

Público ~ Valoración por Equipos ~ Valor Inmobiliario

Finalmente, después de aplicar la regresión lineal con el valor inmobiliario añadido, se nota una mejora significativa del R cuadrado (más de 80%) lo que se concluye que, en la mayoría de los equipos, la asistencia del publico a los partidos puede explicarse por cuando los mismos valen y consecuentemente de cuanto valen las viviendas en las ciudades donde están ubicados los equipos.

In [42]:
```
mf.ols('VALUE_MILLIONS ~COUNTY_POPULATION_MILLIONS+TOTAL_ATTENDANCE_MILLIONS+MEDIAN_HOME_PRICE_COUNTY_MILLIONS', data=val_hous
ts.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:         VALUE_MILLIONS   R-squared:                       0.811
Model:                            OLS   Adj. R-squared:                  0.789
Method:                 Least Squares   F-statistic:                     37.12
Date:                Fri, 24 Jan 2020   Prob (F-statistic):           1.52e-09
Time:                        19:22:00   Log-Likelihood:                -214.03
No. Observations:                  30   AIC:                             436.1
Df Residuals:                      26   BIC:                             441.7
Df Model:                           3
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
Intercept                    -1608.0453    622.423     -2.584      0.016   -2887.454    -328.636
COUNTY_POPULATION_MILLIONS     111.0593     26.277      4.227      0.000      57.047     165.072
TOTAL_ATTENDANCE_MILLIONS     2921.7506    874.687      3.340      0.003    1123.807    4719.695
MEDIAN_HOME_PRICE_COUNTY_MILLIONS     0.0014      0.000      6.826      0.000       0.001       0.002
==============================================================================
Omnibus:                        1.898   Durbin-Watson:                   1.641
Prob(Omnibus):                  0.387   Jarque-Bera (JB):                0.939
Skew:                           0.402   Prob(JB):                        0.625
Kurtosis:                       3.323   Cond. No.                     9.06e+06
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 9.06e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

NORMALIZACIÓN - MINMAX SCALER

Preparación para Clusterización - Normalización

Con la intención de preparar los datos para una futura clusterización, se aplica el método MinMax Scaler para obtener valores entre 0 y 1 de cada equipo y normalizarlos.
Las variables consideradas son: el publico total, Ranking ELO, valor de los equipos y valor de las viviendas.
Los NaN se tratan como valores perdidos: no se tienen en cuenta y se mantienen en la transformación.

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler

#Only cluster on these values
numerical_df = val_housing_win_df.loc[:,["TOTAL_ATTENDANCE_MILLIONS", "ELO", "VALUE_MILLIONS", "MEDIAN_HOME_PRICE_COUNTY_MILLONS"]]

#Scale to between 0 and 1
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
print(scaler.fit(numerical_df))
print(scaler.transform(numerical_df))

#Add back to DataFrame
from sklearn.cluster import KMeans
k_means = KMeans(n_clusters=3)
kmeans = k_means.fit(scaler.transform(numerical_df))
val_housing_win_df['cluster'] = kmeans.labels_
val_housing_win_df.head()
```

```
MinMaxScaler(copy=True, feature_range=(0, 1))
[[1.         0.41898148 0.68627451        nan]
 [0.72637903 0.18981481 0.2745098         nan]
 [0.41067502 0.12731481 0.12745098        nan]
 [0.70531986 0.53472222 0.23529412        nan]
 [0.73232332 0.60648148 0.14705882        nan]
 [0.62487072 0.68981481 0.49019608        nan]
 [0.83819102 0.47916667 0.17647059        nan]
 [0.6983872  1.         0.7254902         nan]
 [0.49678606 0.47453704 0.10784314        nan]
 [0.72417286 0.08333333 1.                nan]
 [0.54749962 0.57638889 0.56862745        nan]
 [0.60477873 0.06712963 0.88235294        nan]
 [0.65812204 0.52083333 0.11764706        nan]
 [0.52863955 0.74768519 0.16666667        nan]
 [0.70957335 0.64583333 0.0627451         nan]
 [0.43166712 0.03240741 0.06666667        nan]
 [0.20301662 0.33333333 0.                nan]
 [0.31881029 0.61111111 0.35294118        nan]
 [0.36376665 0.00462963 0.1372549         nan]
 [0.27883458 0.43518519 0.05098039        nan]
 [0.25319364 0.33333333 0.01568627        nan]
 [0.3708405  0.28935185 0.01176471        nan]
 [0.37053693 0.         0.01960784        nan]
 [0.17197852 0.32638889 0.05294118        nan]
 [0.09538753 0.0787037  0.41176471        nan]
 [0.15307963 0.37962963 0.01372549        nan]
 [0.32306025 0.57638889 0.09803922        nan]
 [0.         0.49537037 0.05490196        nan]
 [0.00571132 0.28935185 0.00784314        nan]
```

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
print(scaler.fit(numerical_df))
print(scaler.transform(numerical_df))
```

```
MinMaxScaler(copy=True, feature_range=(0, 1))
[[1.         0.41898148 0.68627451 0.08776879]
 [0.72637903 0.18981481 0.2745098  0.11603661]
 [0.41067502 0.12731481 0.12745098 0.13419221]
 [0.70531986 0.53472222 0.23529412 0.16243496]
 [0.73232332 0.60648148 0.14705882 0.16306188]
 [0.62487072 0.68981481 0.49019608 0.31038806]
 [0.83819102 0.47916667 0.17647059 0.00476459]
 [0.6983872  1.         0.7254902  0.39188139]
 [0.49678606 0.47453704 0.10784314 0.04993825]
 [0.72417286 0.08333333 1.         1.        ]
 [0.54749962 0.57638889 0.56862745 0.23139615]
 [0.60477873 0.06712963 0.88235294 0.31038806]
 [0.65812204 0.52083333 0.11764706 0.184816  ]
 [0.52863955 0.74768519 0.16666667 0.08156228]
 [0.70957335 0.64583333 0.0627451  0.13983449]
 [0.43166712 0.03240741 0.06666667 0.10657639]
 [0.20301662 0.33333333 0.         0.10350448]
 [0.31881029 0.61111111 0.35294118 0.09062441]
 [0.36376665 0.00462963 0.1372549  0.10350448]
 [0.27883458 0.43518519 0.05098039 0.00946649]
 [0.25319364 0.33333333 0.01568627 0.01573569]
 [0.3708405  0.28935185 0.01176471 0.10977023]
 [0.37053693 0.         0.01960784 0.03140869]
 [0.17197852 0.32638889 0.05294118 0.14738888]
 [0.09538753 0.0787037  0.41176471 0.40756065]
 [0.15307963 0.37962963 0.01372549 0.        ]
 [0.32306025 0.57638889 0.09803922 0.27336844]
 [0.         0.49537037 0.05490196 0.21885775]
 [0.00571132 0.28935185 0.00784314 0.12758322]
 [0.17492596 0.23842593 0.05882353 0.10663281]]
```

CLUSTERIZACIÓN

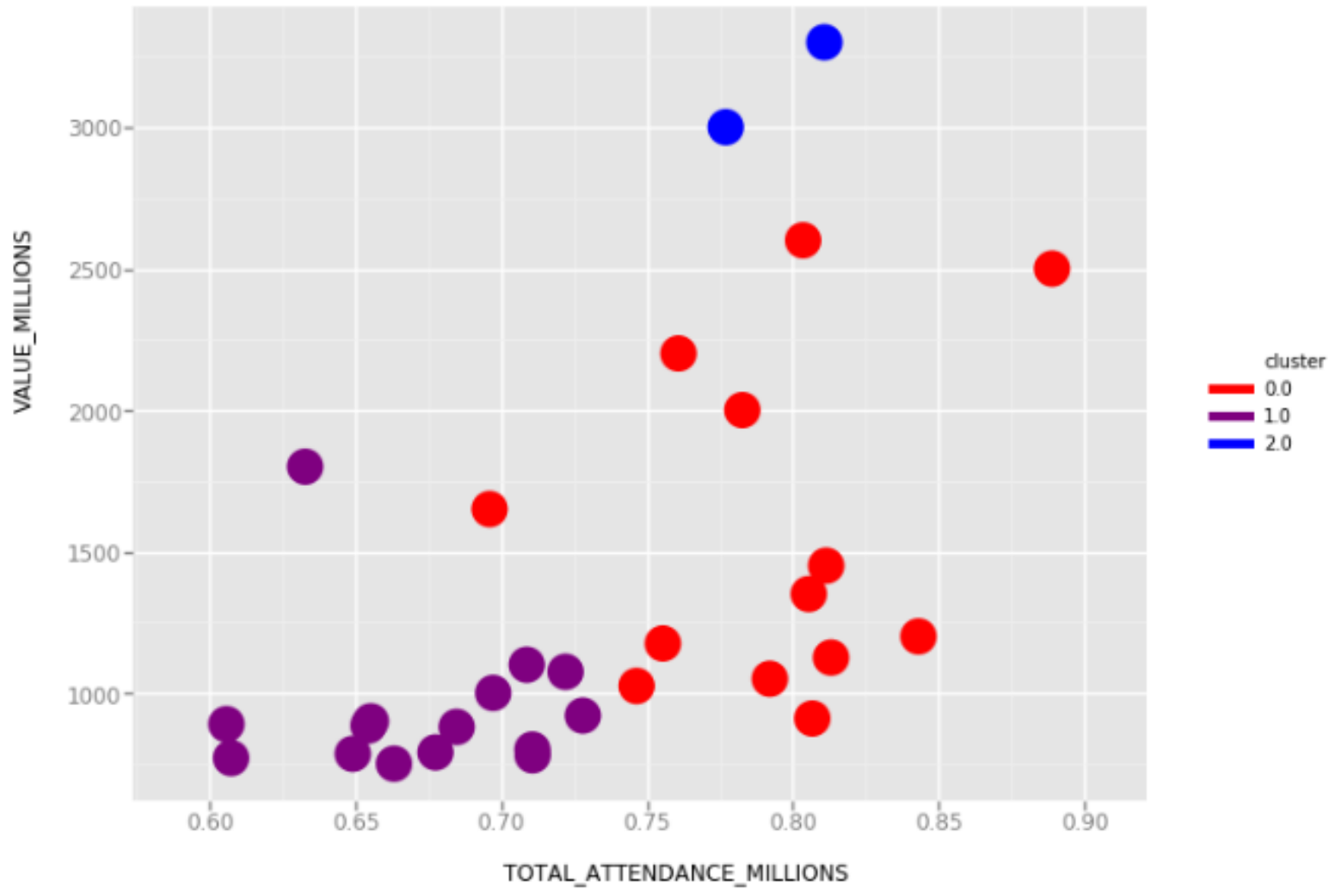
La idea en este apartado es categorizar los equipos de NBA a partir de la correlación entre el valor del equipo versus el público total que asiste a los partidos.

```
from sklearn.cluster import KMeans
k_means = KMeans(n_clusters=3)
kmeans = k_means.fit(scaler.transform(numerical_df))
val_housing_win_df['cluster'] = kmeans.labels_
val_housing_win_df.head()
```

]:

	TEAM	GMS	PCT_ATTENDANCE	WINNING_SEASON	TOTAL_ATTENDANCE_MILLIONS	VALUE_MILLIONS	ELO	CONF	COUNTY	MEDIAN_HOME_PRICE_COUNTY_MILLIONS	COUNTY_POPULATION_MILLIONS	cluster
0	Chicago Bulls	41	104	1	0.888882	2500	1519	East	Cook	269900.0	5.20	0
1	Dallas Mavericks	41	103	0	0.811366	1450	1420	West	Dallas	314990.0	2.57	0
2	Sacramento Kings	41	101	0	0.721928	1075	1393	West	Sacremento	343950.0	1.51	1
3	Miami Heat	41	100	1	0.805400	1350	1569	East	Miami-Dade	389000.0	2.71	0
4	Toronto Raptors	41	100	1	0.813050	1125	1600	East	York-County	390000.0	1.10	0

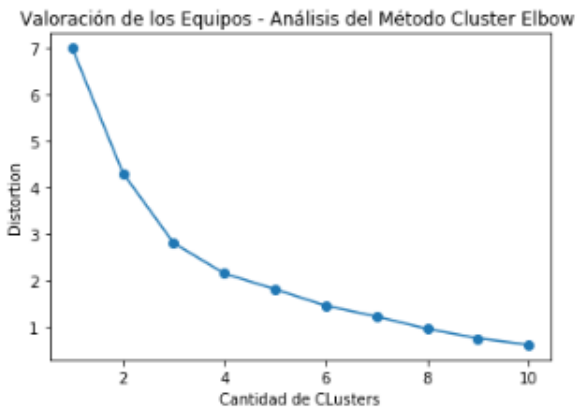
```
from ggplot import *

ggplot(val_housing_win_df, aes(x="TOTAL_ATTENDANCE_MILLIONS", y="VALUE_MILLIONS", color="cluster")) +\
geom_point(size=400) + scale_color_gradient(low = 'red', high = 'blue')
```



CLUSTERIZACIÓN

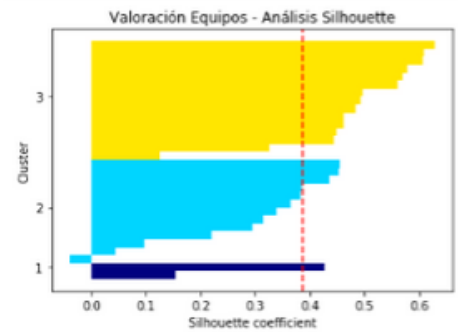
Para verificar si la clusterización en tres grupos ha sido una buena elección, se ejecuta el modelo de Elbow y de Silueta donde se puede notar que la escoja de clusterizar los equipos en tres ha sido una buena idea.

```
distortions = []
for i in range(1, 11):
    km = KMeans(n_clusters=i,
            init='k-means++',
            n_init=10,
            max_iter=300,
            random_state=0)
    km.fit(scaler.transform(numerical_df))
    distortions.append(km.inertia_)

plt.plot(range(1,11), distortions, marker='o')
plt.xlabel('Cantidad de CLusters')
plt.ylabel('Distortion')
plt.title("Valoración de los Equipos - Análisis del Método Cluster Elbow ")
plt.show()
```




```
km = KMeans(n_clusters=3,
            init='k-means++',
            n_init=10,
            max_iter=300,
            random_state=0)
y_km = km.fit_predict(scaler.transform(numerical_df))
```

```
import numpy as np
from matplotlib import cm
from sklearn.metrics import silhouette_samples
cluster_labels = np.unique(y_km)
n_clusters = cluster_labels.shape[0]
silhouette_vals = silhouette_samples(scaler.transform(numerical_df),
                                     y_km,
                                     metric='euclidean')
y_ax_lower, y_ax_upper = 0, 0
yticks = []
for i, c in enumerate(cluster_labels):
    c_silhouette_vals = silhouette_vals[y_km == c]
    c_silhouette_vals.sort()
    y_ax_upper += len(c_silhouette_vals)
    color = cm.jet(float(i)/n_clusters)
    plt.barh(range(y_ax_lower, y_ax_upper), c_silhouette_vals, height=1.0, edgecolor='none',color=color)
    yticks.append((y_ax_lower + y_ax_upper)/2)
    y_ax_lower += len(c_silhouette_vals)
silhouette_avg = np.mean(silhouette_vals)
plt.axvline(silhouette_avg,
            color="red",
            linestyle="--")
plt.yticks(yticks, cluster_labels + 1)
plt.ylabel('Cluster')
plt.xlabel('Silhouette coefficient')
plt.title('Valoración Equipos - Análisis Silhouette')
plt.show()
```

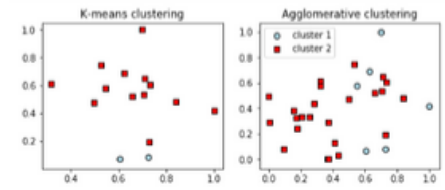


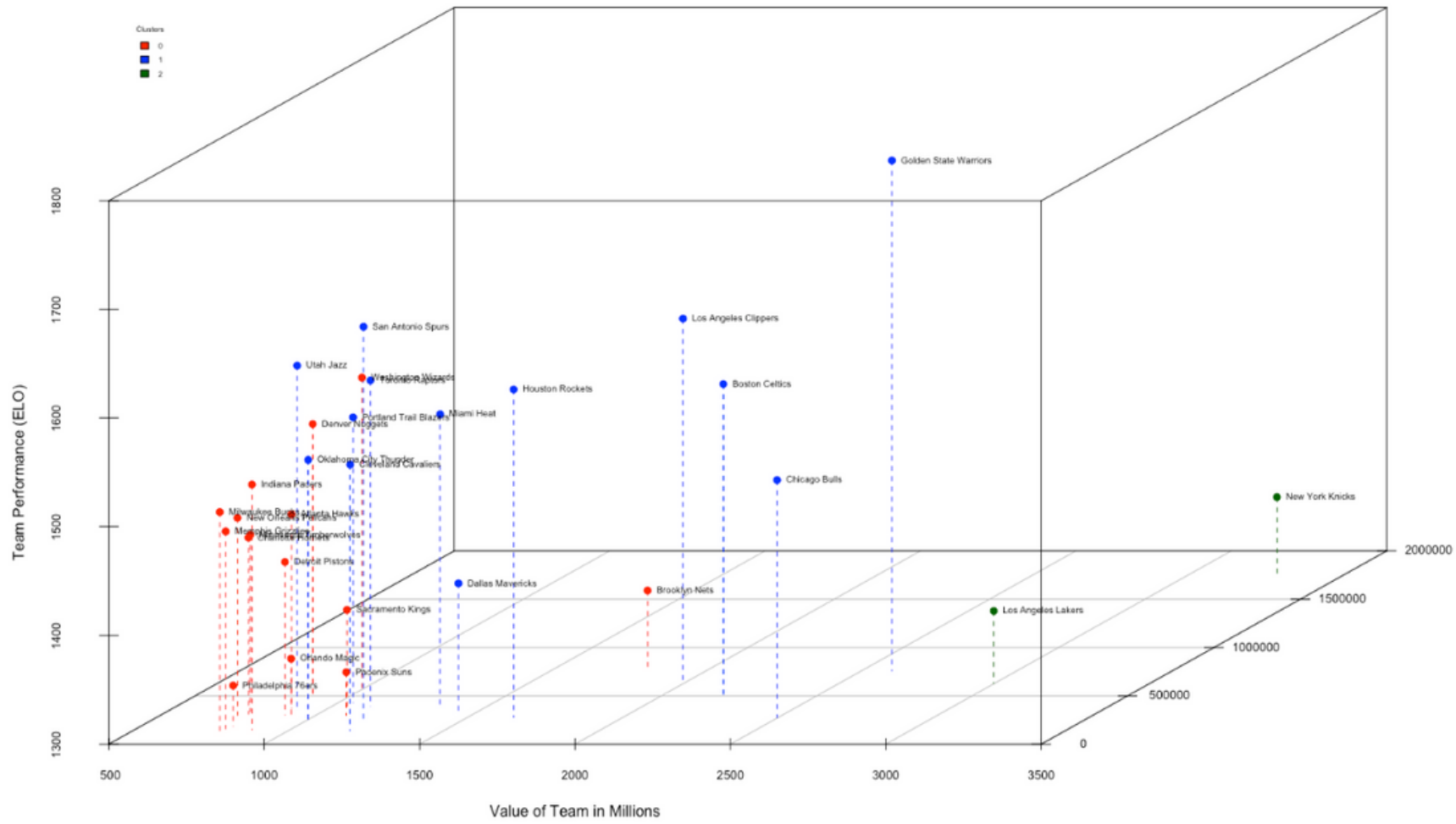
CLUSTERIZACIÓN - RESULTADO

Como resultado final de la clusterización, se puede notar a través del Gráfico 3D la correlación entre variables y como los equipos se categorizan por su desempeño (Ranking ELO), Valor del equipo y valor de las viviendas.

```
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 3))
km = KMeans(n_clusters=3,
            random_state=0)
X = scaler.transform(numerical_df)
y_km = km.fit_predict(X)
ax1.scatter(X[y_km==0,0],
            X[y_km==0,1],
            c='lightblue',
            edgecolor='black',
            marker='o',
            s=40,
            label='cluster 1')
ax1.scatter(X[y_km==1,0],
            X[y_km==1,1],
            c='red',
            edgecolor='black',
            marker='s',
            s=40,
            label='cluster 2')
ax1.set_title('K-means clustering')
from sklearn.cluster import AgglomerativeClustering

X = scaler.transform(numerical_df)
ac = AgglomerativeClustering(n_clusters=2,
                             affinity='euclidean',
                             linkage='complete')
```

```
y_ac = ac.fit_predict(X)
ax2.scatter(X[y_ac==0,0],
            X[y_ac==0,1],
            c='lightblue',
            edgecolor='black',
            marker='o',
            s=40,
            label='cluster 1')
ax2.scatter(X[y_ac==1,0],
            X[y_ac==1,1],
            c='red',
            edgecolor='black',
            marker='s',
            s=40,
            label='cluster 2')
ax2.set_title('Agglomerative clustering')
plt.legend()
plt.show()
```





ANÁLISIS DE JUGADORES

Después de analizar los equipos, ahora serán analizados los jugadores y su relación entre las poder social (Twitter y Wikipedia), salarios y rendimiento en la pista.
He empezado cargando las librerías y las fuentes de datos necesarias:
- Público
- Salarios
- PIE (Player Impact Estimate)
- RPM (Real Plus Minus)

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
color = sns.color_palette()
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:100% !important; }</style>"))
%matplotlib inline
attendance_valuation_elo_df = pd.read_csv("../data/nba_2017_att_val_elo.csv")
salary_df = pd.read_csv("../data/nba_2017_salary.csv")
pie_df = pd.read_csv("../data/nba_2017_pie.csv")
plus_minus_df = pd.read_csv("../data/nba_2017_real_plus_minus.csv")
br_stats_df = pd.read_csv("../data/nba_2017_br.csv")
```

Público

```
attendance_valuation_elo_df = pd.read_csv("../data/nba_2017_att_val_elo.csv");attendance_valuation_elo_df.head()
```

	Unnamed: 0	TEAM	GMS	PCT	TOTAL_MILLIONS	AVG_MILLIONS	VALUE_MILLIONS	ELO	CONF
0	0	Chicago Bulls	41	104	0.888882	0.021680	2500.0	1519	East
1	1	Dallas Mavericks	41	103	0.811366	0.019789	1450.0	1420	West
2	2	Sacramento Kings	41	101	0.721928	0.017608	1075.0	1393	West
3	3	Miami Heat	41	100	0.805400	0.019643	1350.0	1569	East
4	4	Toronto Raptors	41	100	0.813050	0.019830	1125.0	1600	East

Salarios

```
salary_df = pd.read_csv("../data/nba_2017_salary.csv");salary_df.head()
```

	NAME	POSITION	TEAM	SALARY
0	LeBron James	SF	Cleveland Cavaliers	30963450.0
1	Mike Conley	PG	Memphis Grizzlies	26540100.0
2	Al Horford	C	Boston Celtics	26540100.0
3	Dirk Nowitzki	PF	Dallas Mavericks	25000000.0
4	Carmelo Anthony	SF	New York Knicks	24559380.0

ANÁLISIS DE JUGADORES

Carga de Datos

PIE (Player Impact Estimate)

```
pie_df = pd.read_csv("../data/nba_2017_pie.csv");pie_df.head()
```

	PLAYER	TEAM	AGE	GP	W	L	MIN	OFFRTG	DEFRTG	NETRTG	...	AST RATIO	OREB%	DREB%	REB%	TO RATIO	EFG%	TS%	USG%	PACE	PIE
0	Russell Westbrook	OKC	28	81	46	35	34.6	107.9	104.6	3.3	...	23.4	5.3	27.9	16.7	12.2	47.6	55.4	40.8	102.31	23.0
1	Boban Marjanovic	DET	28	35	16	19	8.4	104.3	102.4	1.9	...	5.1	16.6	31.3	23.9	5.7	54.5	60.6	24.8	97.20	19.6
2	Demetrius Jackson	BOS	22	5	1	4	3.4	124.2	117.8	6.3	...	31.1	9.1	11.8	10.3	0.0	87.5	75.3	17.2	87.46	19.4
3	Anthony Davis	NOP	24	75	31	44	36.1	104.2	102.5	1.7	...	7.3	6.7	26.9	17.0	8.4	51.8	58.0	32.6	100.19	19.2
4	James Harden	HOU	27	81	54	27	36.4	113.6	107.3	6.3	...	27.6	3.5	21.2	12.3	14.1	52.5	61.3	34.1	102.98	19.0

5 rows × 22 columns

RPM (Real Plus Minus)

```
plus_minus_df = pd.read_csv("../data/nba_2017_real_plus_minus.csv");plus_minus_df.head()
```

	NAME	TEAM	GP	MPG	ORPM	DRPM	RPM	WINS
0	LeBron James, SF	CLE	74	37.8	6.49	1.93	8.42	20.43
1	Stephen Curry, PG	GS	79	33.4	7.27	0.14	7.41	18.80
2	Jimmy Butler, SG	CHI	76	37.0	4.82	1.80	6.62	17.35
3	Russell Westbrook, PG	OKC	81	34.6	6.74	-0.47	6.27	17.34
4	Draymond Green, PF	GS	76	32.5	1.55	5.59	7.14	16.84

Basketball Reference (Estadísticas Generales)

```
br_stats_df = pd.read_csv("../data/nba_2017_br.csv");br_stats_df.head()
```

	Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	...	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PS/G
0	1	Russell Westbrook	PG	28	OKC	81	81	34.6	10.2	24.0	...	0.845	1.7	9.0	10.7	10.4	1.6	0.4	5.4	2.3	31.6
1	2	James Harden	PG	27	HOU	81	81	36.4	8.3	18.9	...	0.847	1.2	7.0	8.1	11.2	1.5	0.5	5.7	2.7	29.1
2	3	Isaiah Thomas	PG	27	BOS	76	76	33.8	9.0	19.4	...	0.909	0.6	2.1	2.7	5.9	0.9	0.2	2.8	2.2	28.9
3	4	Anthony Davis	C	23	NOP	75	75	36.1	10.3	20.3	...	0.802	2.3	9.5	11.8	2.1	1.3	2.2	2.4	2.2	28.0
4	5	DeMar DeRozan	SG	27	TOR	74	74	35.4	9.7	20.9	...	0.842	0.9	4.3	5.2	3.9	1.1	0.2	2.4	1.8	27.3

5 rows × 30 columns

ANÁLISIS DE JUGADORES

Limpieza y Unión de Datos

En esta parte, se realiza la limpieza de los datos, quedando solamente las columnas de interés para los análisis.

RPM

```
plus_minus_df.rename(columns={"NAME":"PLAYER", "WINS": "WINS_RPM"}, inplace=True)
players = []
for player in plus_minus_df["PLAYER"]:
    plyr, _ = player.split(",")
    players.append(plyr)
plus_minus_df.drop(["PLAYER"], inplace=True, axis=1)
plus_minus_df["PLAYER"] = players
plus_minus_df.head()
```

	TEAM	GP	MPG	ORPM	DRPM	RPM	WINS_RPM	PLAYER
0	CLE	74	37.8	6.49	1.93	8.42	20.43	LeBron James
1	GS	79	33.4	7.27	0.14	7.41	18.80	Stephen Curry
2	CHI	76	37.0	4.82	1.80	6.62	17.35	Jimmy Butler
3	OKC	81	34.6	6.74	-0.47	6.27	17.34	Russell Westbrook
4	GS	76	32.5	1.55	5.59	7.14	16.84	Draymond Green

Basketball Reference (Estadísticas Generales)

```
nba_players_df = br_stats_df.copy()
nba_players_df.rename(columns={'Player': 'PLAYER','Pos':'POSITION', 'Tm': "TEAM", 'Age': 'AGE', "PS/G": "POINTS"}, inplace=True)
nba_players_df.drop(["G", "GS", "TEAM"], inplace=True, axis=1)
nba_players_df = nba_players_df.merge(plus_minus_df, how="inner", on="PLAYER")
nba_players_df.head()
```

	Rk	PLAYER	POSITION	AGE	MP	FG	FGA	FG%	3P	3PA	...	TOV	PF	POINTS	TEAM	GP	MPG	ORPM	DRPM	RPM	WINS_RPM
0	1	Russell Westbrook	PG	28	34.6	10.2	24.0	0.425	2.5	7.2	...	5.4	2.3	31.6	OKC	81	34.6	6.74	-0.47	6.27	17.34
1	2	James Harden	PG	27	36.4	8.3	18.9	0.440	3.2	9.3	...	5.7	2.7	29.1	HOU	81	36.4	6.38	-1.57	4.81	15.54
2	3	Isaiah Thomas	PG	27	33.8	9.0	19.4	0.463	3.2	8.5	...	2.8	2.2	28.9	BOS	76	33.8	5.72	-3.89	1.83	8.19
3	4	Anthony Davis	C	23	36.1	10.3	20.3	0.505	0.5	1.8	...	2.4	2.2	28.0	NO	75	36.1	0.45	3.90	4.35	12.81
4	5	DeMar DeRozan	SG	27	35.4	9.7	20.9	0.467	0.4	1.7	...	2.4	1.8	27.3	TOR	74	35.4	2.21	-2.04	0.17	5.46

5 rows × 34 columns

ANÁLISIS DE JUGADORES

Limpieza y Unión de Datos

PIE

```
pie_df_subset = pie_df[["PLAYER", "PIE", "PACE", "W"]].copy()
nba_players_df = nba_players_df.merge(pie_df_subset, how="inner", on="PLAYER")
nba_players_df.head()
```

	Rk	PLAYER	POSITION	AGE	MP	FG	FGA	FG%	3P	3PA	...	TEAM	GP	MPG	ORPM	DRPM	RPM	WINS_RPM	PIE	PACE	W
0	1	Russell Westbrook	PG	28	34.6	10.2	24.0	0.425	2.5	7.2	...	OKC	81	34.6	6.74	-0.47	6.27	17.34	23.0	102.31	46
1	2	James Harden	PG	27	36.4	8.3	18.9	0.440	3.2	9.3	...	HOU	81	36.4	6.38	-1.57	4.81	15.54	19.0	102.98	54
2	3	Isaiah Thomas	PG	27	33.8	9.0	19.4	0.463	3.2	8.5	...	BOS	76	33.8	5.72	-3.89	1.83	8.19	16.1	99.84	51
3	4	Anthony Davis	C	23	36.1	10.3	20.3	0.505	0.5	1.8	...	NO	75	36.1	0.45	3.90	4.35	12.81	19.2	100.19	31
4	5	DeMar DeRozan	SG	27	35.4	9.7	20.9	0.467	0.4	1.7	...	TOR	74	35.4	2.21	-2.04	0.17	5.46	15.5	97.69	47

5 rows × 37 columns

Salarios

```
nba_players_df.to_csv("../data/nba_2017_players_stats_combined.csv")
```

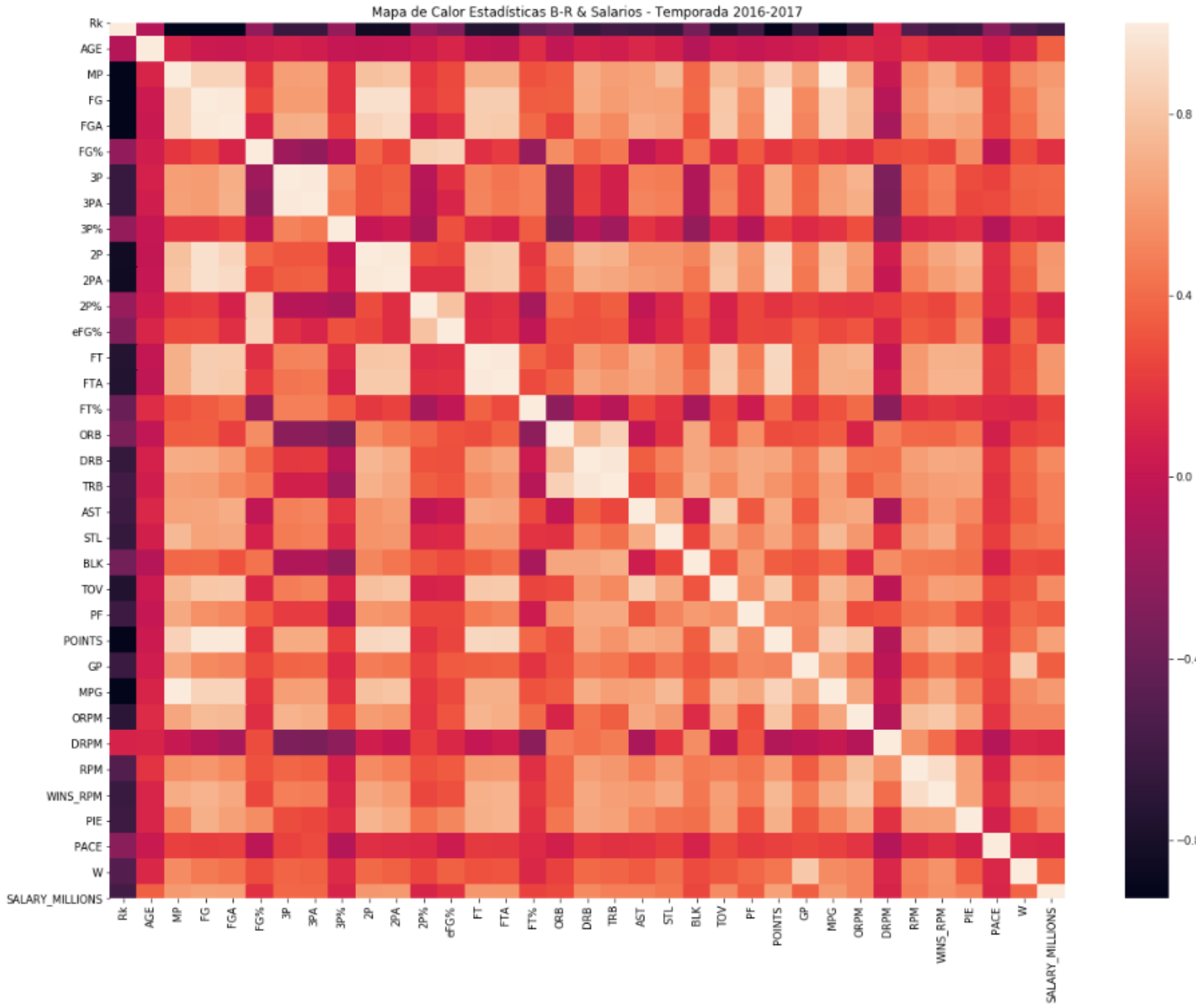
```
salary_df.rename(columns={'NAME': 'PLAYER'}, inplace=True)
salary_df["SALARY_MILLIONS"] = round(salary_df["SALARY"]/1000000, 2)
salary_df.drop(["POSITION","TEAM", "SALARY"], inplace=True, axis=1)
salary_df.head()
```

	PLAYER	SALARY_MILLIONS
0	LeBron James	30.96
1	Mike Conley	26.54
2	Al Horford	26.54
3	Dirk Nowitzki	25.00
4	Carmelo Anthony	24.56

ANÁLISIS DE JUGADORES

Correlación entre Variables

Después de cargar, limpiar y unir los datos en un data frame, se crea un mapa de calor para que se pueda observar cómo se correlacionan las 35 columnas y 342 filas.



ANÁLISIS DE JUGADORES

Regresión Lineal

Salario ~ Victorias

OLS Regression Results

Dep. Variable:	W	R-squared:	0.324
Model:	OLS	Adj. R-squared:	0.322
Method:	Least Squares	F-statistic:	162.9
Date:	Thu, 06 Feb 2020	Prob (F-statistic):	9.34e-31
Time:	19:40:21	Log-Likelihood:	-1334.8
No. Observations:	342	AIC:	2674.
Df Residuals:	340	BIC:	2681.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	22.8208	0.808	28.233	0.000	21.231	24.411
WINS_RPM	2.1419	0.168	12.765	0.000	1.812	2.472

Omnibus:	3.692	Durbin-Watson:	1.400
Prob(Omnibus):	0.158	Jarque-Bera (JB):	3.607
Skew:	0.207	Prob(JB):	0.165
Kurtosis:	2.716	Cond. No.	6.08

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Salario ~ Puntos

OLS Regression Results

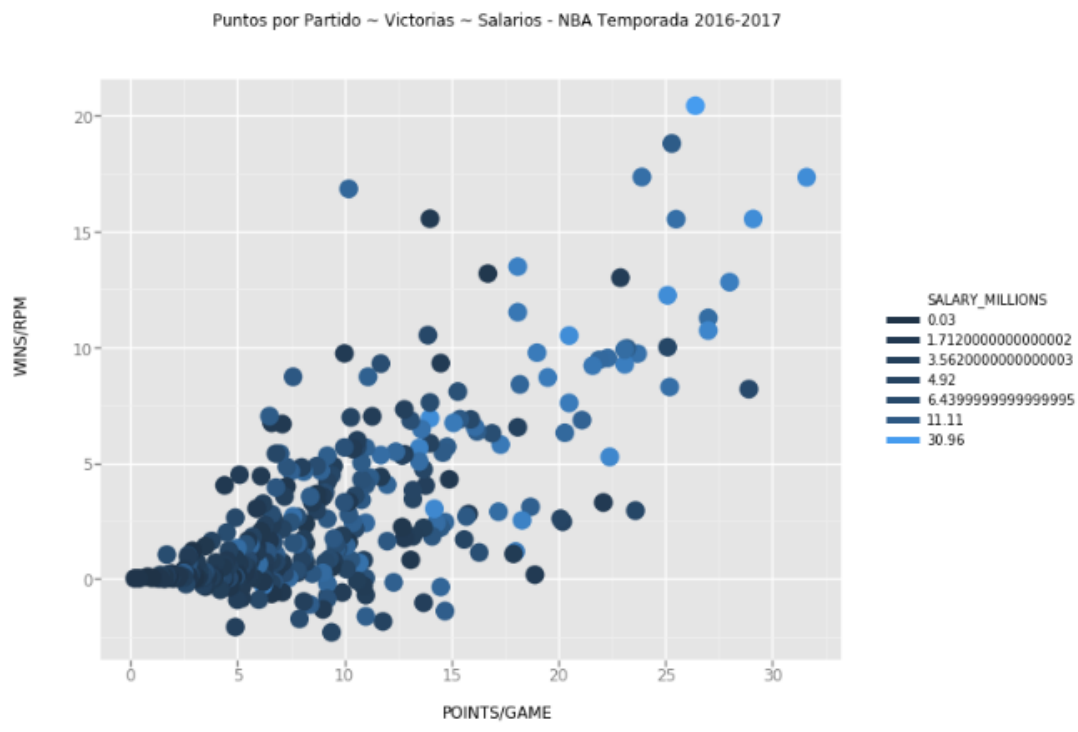
Dep. Variable:	SALARY_MILLIONS	R-squared:	0.404
Model:	OLS	Adj. R-squared:	0.402
Method:	Least Squares	F-statistic:	230.2
Date:	Thu, 06 Feb 2020	Prob (F-statistic):	4.49e-40
Time:	19:40:40	Log-Likelihood:	-1037.4
No. Observations:	342	AIC:	2079.
Df Residuals:	340	BIC:	2086.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0479	0.494	2.123	0.034	0.077	2.019
POINTS	0.6612	0.044	15.174	0.000	0.575	0.747

Omnibus:	15.215	Durbin-Watson:	1.926
Prob(Omnibus):	0.000	Jarque-Bera (JB):	17.006
Skew:	0.444	Prob(JB):	0.000203
Kurtosis:	3.636	Cond. No.	20.6

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Salario ~ Puntos por Partido ~ Victorias



ANÁLISIS DE JUGADORES

Twitter y Wikipedia Añadido

Finalmente,he añadido al modelo los datos de Twitter, donde se mide la asiduidad de posts de los jugadores y Wikipedia , donde se analiza los accesos a las páginas correspondientes a búsquedas de cada jugador.

```
wiki_df = pd.read_csv("../data/nba_2017_player_wikipedia.csv");wiki_df.head()
```

	Unnamed: 0	names	pageviews	timestamps	wikipedia_handles
0	0	Russell Westbrook	3400	2016010100	Russell_Westbrook
1	1	Russell Westbrook	2893	2016010200	Russell_Westbrook
2	2	Russell Westbrook	3209	2016010300	Russell_Westbrook
3	3	Russell Westbrook	2531	2016010400	Russell_Westbrook
4	4	Russell Westbrook	2599	2016010500	Russell_Westbrook

```
wiki_df.rename(columns={'names': 'PLAYER', "pageviews": "PAGEVIEWS"}, inplace=True)
```

```
median_wiki_df = wiki_df.groupby("PLAYER").median()
```

```
median_wiki_df_small = median_wiki_df[["PAGEVIEWS"]]
```

```
median_wiki_df_small.reset_index(level=0, inplace=True);median_wiki_df_small.head()
```

	PLAYER	PAGEVIEWS
0	A.J. Hammons	1.0
1	Aaron Brooks	10.0
2	Aaron Gordon	666.0
3	Aaron Harrison	487.0
4	Adreian Payne	166.0

```
nba_players_with_salary_wiki_df = nba_players_with_salary_df.merge(median_wiki_df_small)
```

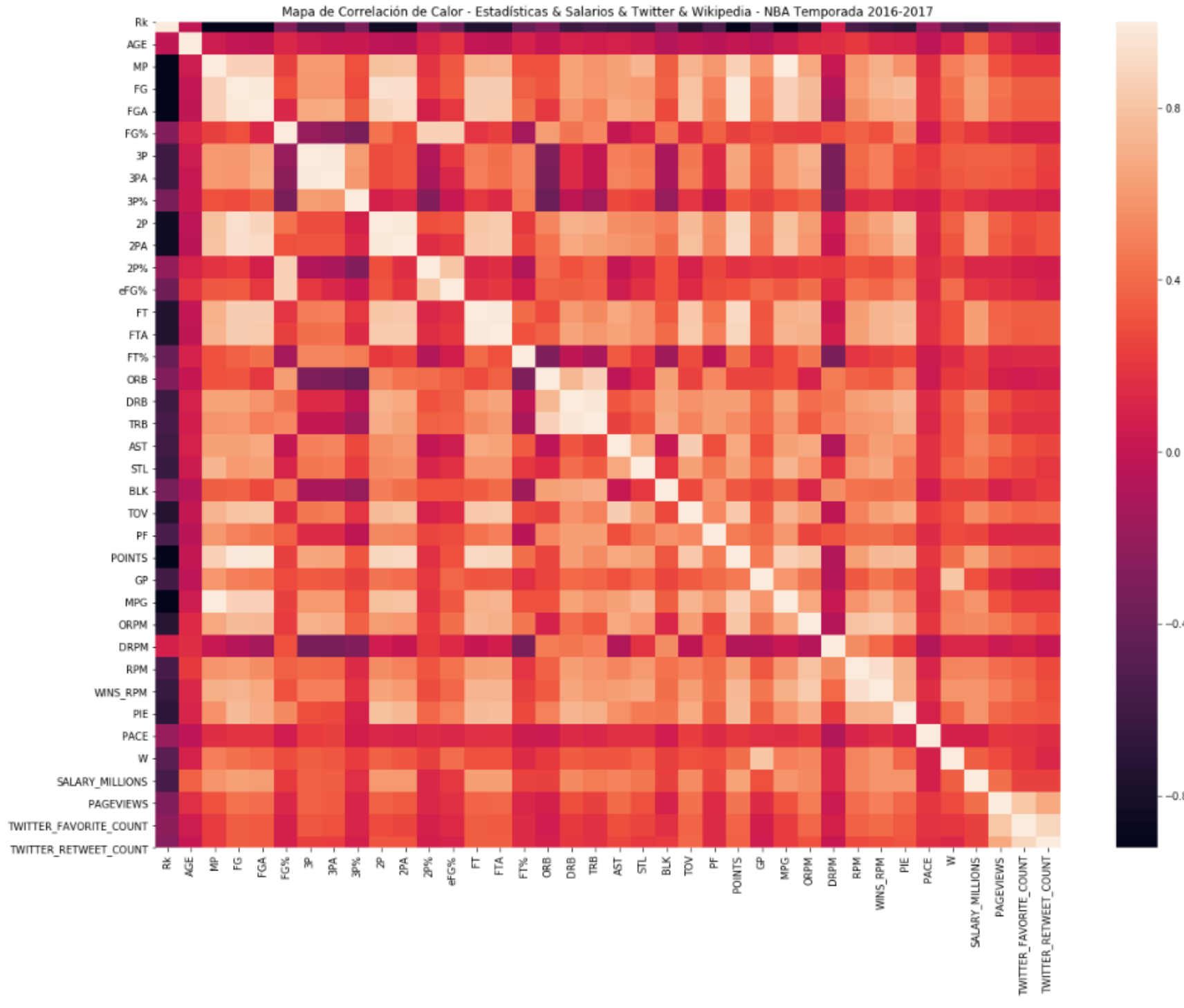
```
twitter_df = pd.read_csv("../data/nba_2017_twitter_players.csv");twitter_df.head()
```

	PLAYER	TWITTER_FAVORITE_COUNT	TWITTER_RETWEET_COUNT
0	Russell Westbrook	2130.5	559.0
1	James Harden	969.0	321.5
2	Isaiah Thomas	467.5	155.5
3	Anthony Davis	368.0	104.0
4	DeMar DeRozan	0.0	186.0

ANÁLISIS DE JUGADORES

Twitter y Wikipedia Añadido

El mapa de calor a continuación muestra la salida combinada de la correlación de 35 columnas y 342 filas. Un par de cosas inmediatas que resaltan es que el salario está altamente correlacionado con ambos puntos y las victorias.
Otra correlación interesante es que las visitas a la página de Wikipedia están fuertemente correlacionadas con los recuentos de favoritos de Twitter. Esta correlación tiene sentido intuitivamente porque ambas son medidas de compromiso y popularidad de los jugadores de la NBA entre los aficionados.
Una última fuerte correlación a resaltar es la de AGE (edad) versus PACE (ritmo), donde la fuerte correlación se puede explicar por cuanto más avanzada la edad, aparentemente, menor el PACE.



ANÁLISIS DE JUGADORES

Patrocinios ~ Visitas Pagina WIkipedia

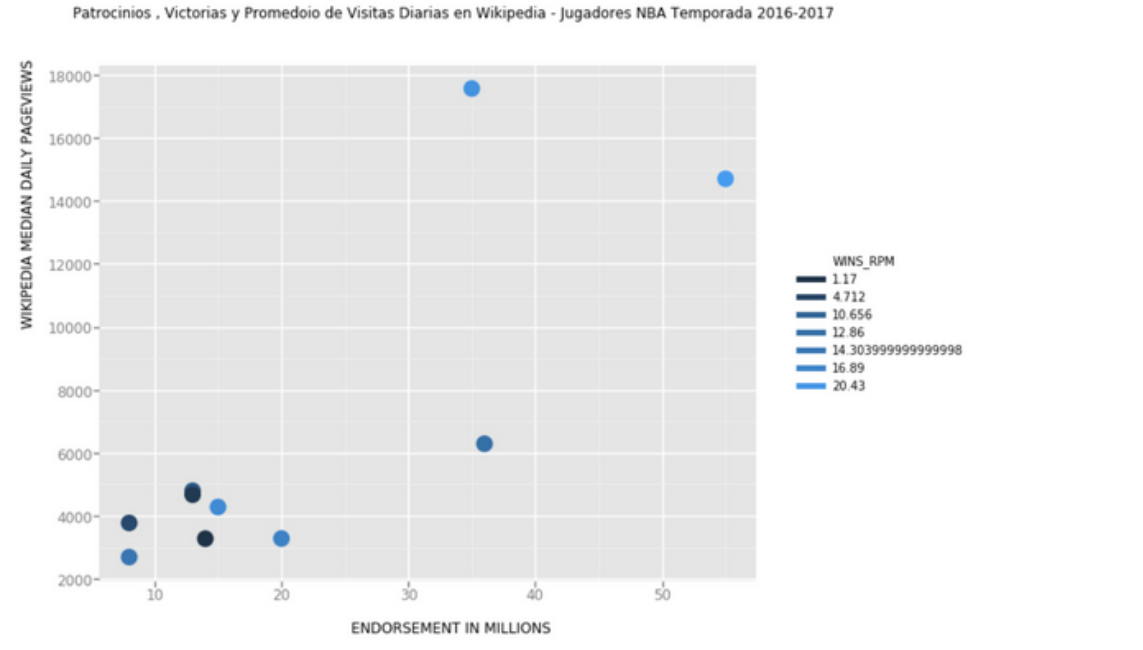
La última correlación de variables a ser analizada será para verificar cómo se explica la popularidad de los jugadores a través de unión entre patrocinios y visitas en wikipedia de cada jugador, siendo posible verificar aún en el gráfico como se comportan la correlación a partir del numero de victorias obtenidas, donde se puede notar una significativa correlación de más de 65% y un valor p de 0,259.

```
results = smf.ols('ENDORSEMENT_MILLIONS ~PAGEVIEWS', data=endorsement_value_stats_df).fit()
```

```
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     ENDORSEMENT_MILLIONS   R-squared:                       0.656
Model:                              OLS   Adj. R-squared:                  0.613
Method:                   Least Squares   F-statistic:                     15.26
Date:                  Sun, 09 Feb 2020   Prob (F-statistic):            0.00450
Time:                          00:46:52   Log-Likelihood:                -35.645
No. Observations:                    10   AIC:                             75.29
Df Residuals:                         8   BIC:                             75.90
Df Model:                             1
Covariance Type:              nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      6.0826      5.011      1.214      0.259      -5.474      17.639
PAGEVIEWS      0.0024      0.001      3.906      0.005       0.001       0.004
==============================================================================
Omnibus:                        0.873   Durbin-Watson:                   2.742
Prob(Omnibus):                  0.646   Jarque-Bera (JB):                0.698
Skew:                           0.523   Prob(JB):                        0.706
Kurtosis:                       2.237   Cond. No.                     1.36e+04
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.36e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

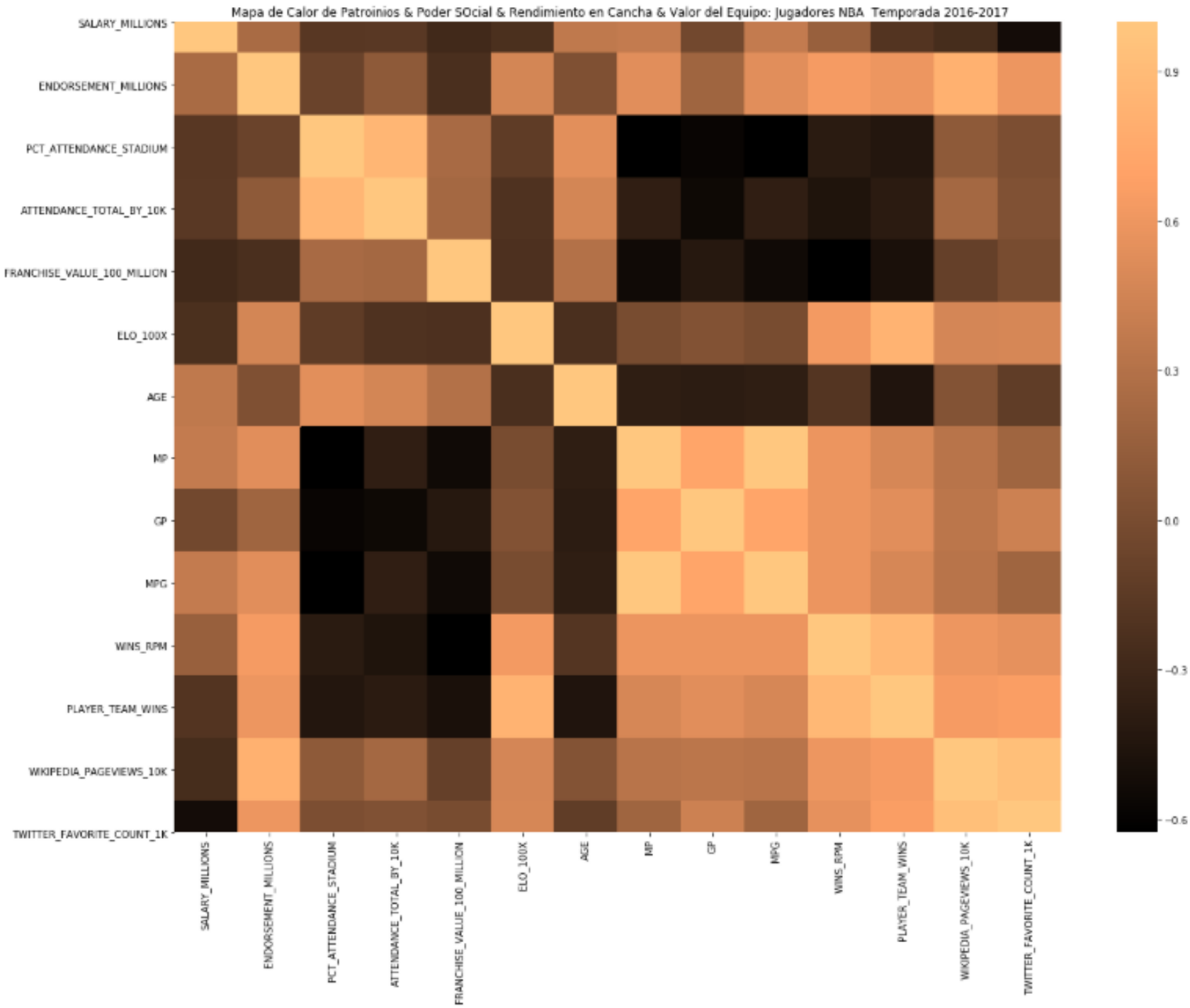
```
p + xlab("ENDORSEMENT IN MILLIONS") + ylab("WIKIPEDIA MEDIAN DAILY PAGEVIEWS") + ggtitle(" Patrocinios , Victorias y Promedoi
```



ANÁLISIS DE JUGADORES

Análisis Final

Por último, he generado otros dos mapas de calor para analizar el poder social de los jugadores de la NBA de acuerdo con las variables clave verificadas.
El primer gráfico muestra la correlación entre las variables de patrocinios, poder social, rendimiento de los jugadores en pista y el valor del equipo.



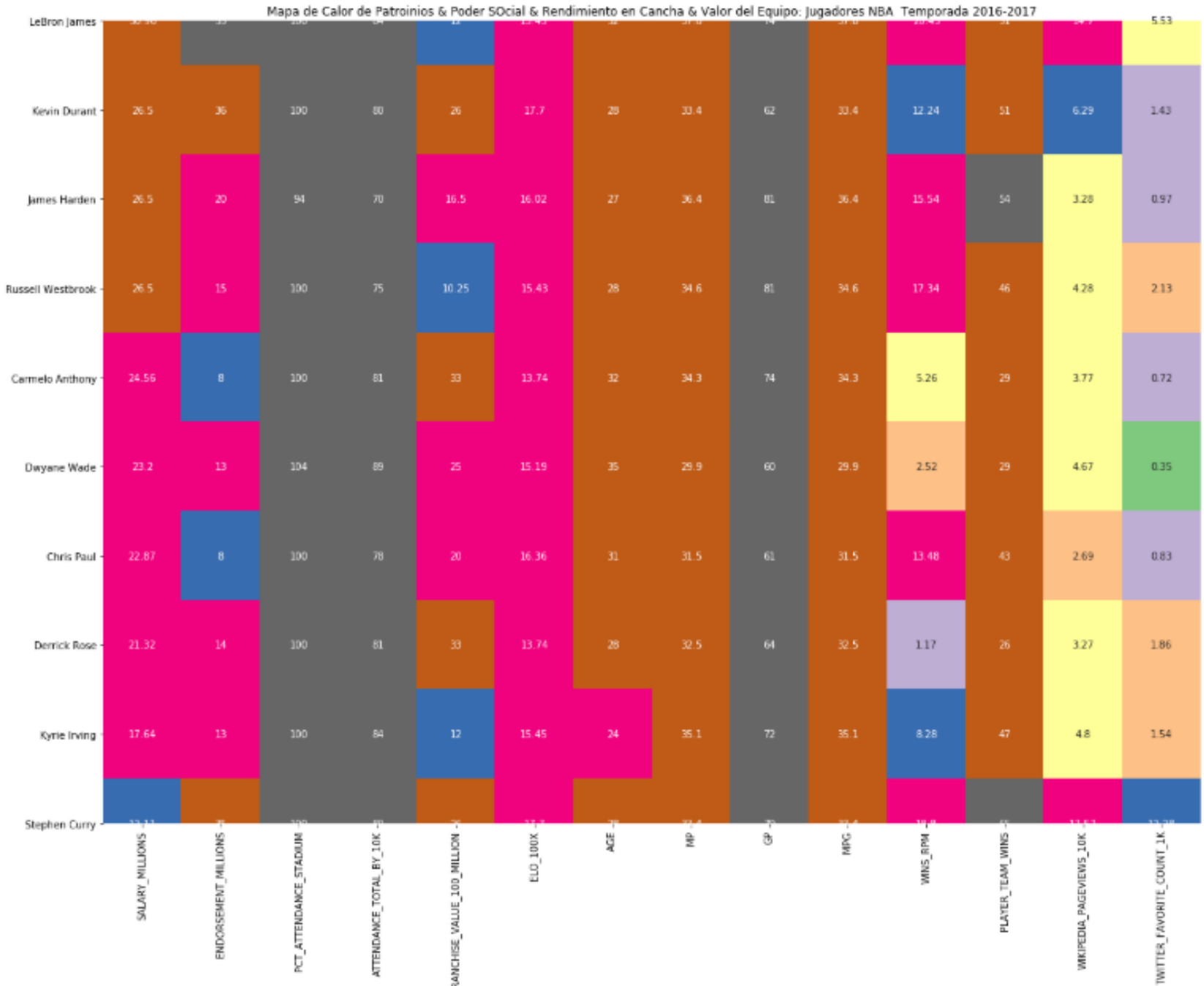
ANÁLISIS DE JUGADORES

Análisis Final

El segundo gráfico muestra la misma correlación entre las variables de patrocinios, poder social, rendimiento de los jugadores en pista y el valor del equipo pero separado por jugador. Debido a la variación del valor de las variables y con la intención de presentar un contraste claro entre los valores, he optado por un mapa de calor que traza una escala de colores a partir del valor de cada variable clave.

```
from matplotlib.colors import LogNorm
plt.subplots(figsize=(20,15))
pd.set_option('display.float_format', lambda x: '%.3f' % x)
norm = LogNorm()
ax = plt.axes()
grid = endorsements.select_dtypes([np.number])
ax.set_title("Mapa de Calor de Patroinios & Poder SOcial & Rendimiento en Cancha & Valor del Equipo: Jugadores NBA  Temporada
sns.heatmap(grid,annot=True, yticklabels=endorsements["PLAYER"],fmt='g', cmap="Accent", cbar=False, norm=norm)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x254c0deb0c8>
```



CONCLUSIONES

Análisis Equipos

Respecto este trabajo he sacado tres principales conclusiones del análisis realizado:

1. La valoración de un equipo de la NBA se ve afectada por el promedio de asistencia.
2. El ranking ELO también está relacionado con la asistencia. En términos generales, cuanto mejor es un equipo, más fanáticos asisten a los juegos.
3. La Conferencia del Este tiene una asistencia mediana más baja y calificaciones de ELO.

Análisis Jugadores

Las principales conclusiones técnicas que se pueden sacar de estos análisis es que el salario pagado a los jugadores no es el mejor predictor de victorias.
Los aficionados se involucran más con jugadores altamente calificados a nivel de juego (en comparación con los altamente remunerados, por ejemplo).
Los ingresos de patrocinio se correlacionan con la cantidad de victorias que un equipo tiene para un jugador, por lo que es posible que quieran tener cuidado con el equipo al que cambian.
Parece que hay una audiencia diferente que asiste a los partidos en persona y la audiencia que participa en las redes sociales. La audiencia en persona parece molesta si su equipo no gana partidos.

De forma general, es imposible no concluir que la cantidad de estadísticas utilizadas en NBA para análisis de equipos y jugadores son extremamente completas, explicando porque es una de las ligas más famosas, de alto nivel y rentables del mundo.
A continuación, es posible calcular la correlación entre otras variables o otras opciones de clusterización para posibles futuros proyectos.