

EA

ARGENTINA



FIFA 19



# ¿PODREMOS PREDDECIR QUE POSICIÓN OCUPA UN JUGADOR EN BASE A SUS ESTADÍSTICAS DE EL VIDEOJUEGO FIFA 19

VAMOS A VERLO !!



# PRESENTACIÓN DE LOS DATOS

El archivo FIFA 19.csv contiene los datos de los jugadores de el videojuego con todas sus características.

Fue extraído de la página Kaggle.com

# VEAMOS LOS PASOS REALIZADOS

# 01 DATA WRANGLING

- Verificamos las características de nuestras variables con un df.info()
- Revisamos si existían registros Nulos
- Analizamos que la consistencia de nuestro dataset esté lo mas prolifa posible

# DATA WRANGLING

```
In [34]: #Verificamos si existen registros nulos  
df.isnull().sum()
```

```
Out[34]: Name      0  
Age       0  
Nationality 0  
Overall    0  
Potential  0  
..  
Height    0  
Weight    0  
Right Foot 0  
Offensive Work Rate 0  
Defensive Work Rate 0  
Length: 82, dtype: int64
```



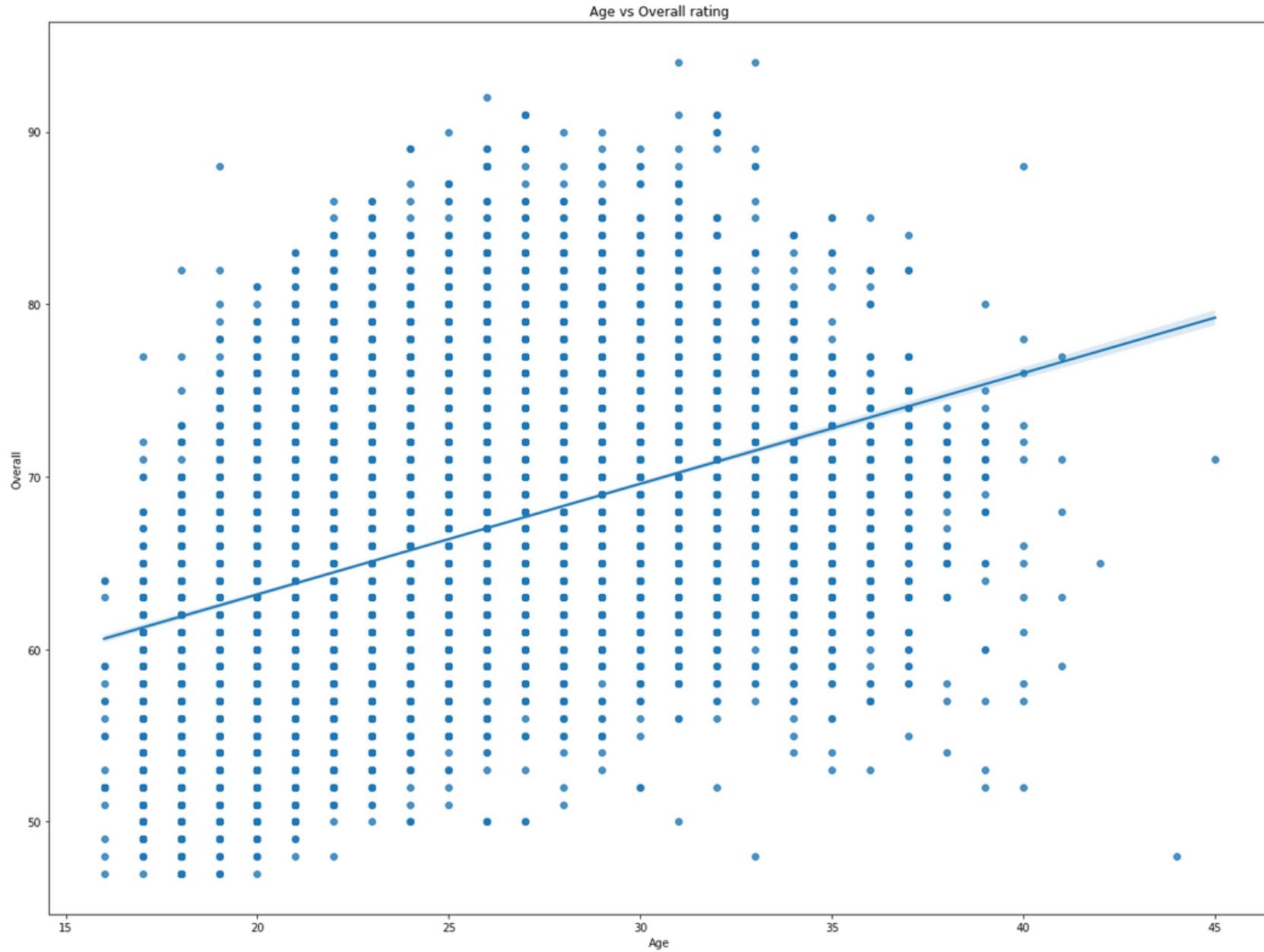
Verificamos que los datos a trabajar se encuentran en correctas condiciones, no hay nulos, los tipos de datos son los correctos. Podemos asegurar la fiabilidad, credibilidad y calidad de los datos limpios para utilizarlo en la toma de decisiones.

02

# ANÁLISIS EXPLORATORIO DE DATOS

- Realizamos un HeadMap de las variables para ver la relación entre cada una de las variables
- Con ello nos percatamos algunos casos evidentes de análisis univariado y bivariado
- ejecutamos una correlación Lineal para dejar en evidencia nuestro análisis

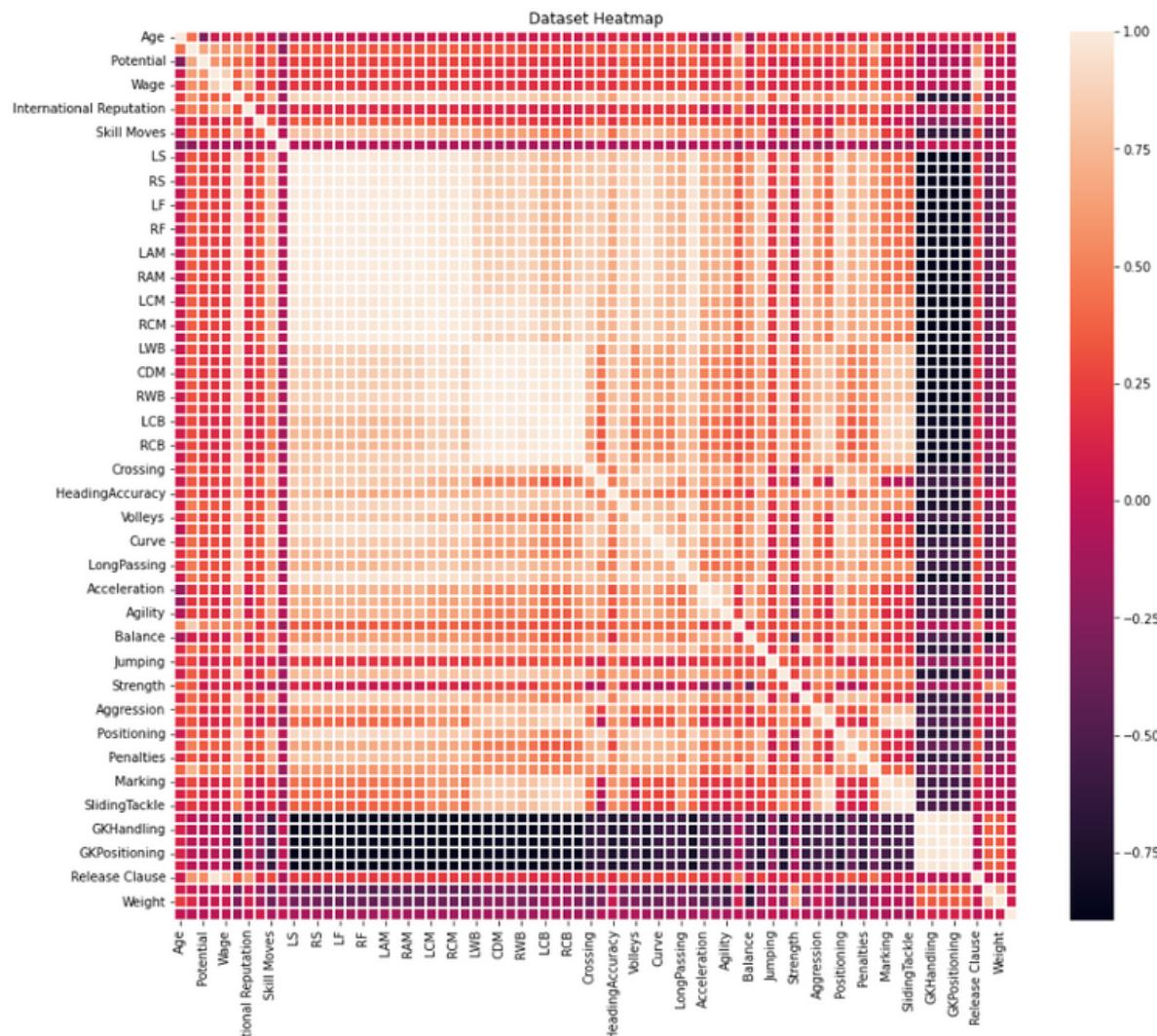
# Análisis de Edad vs Overall Rating



# HEADMAP

In [35]:

```
plt.figure(figsize=(15,13))
sns.heatmap(df.corr(),linewdiths=1)
plt.title('Dataset Heatmap')
plt.show()
```



03

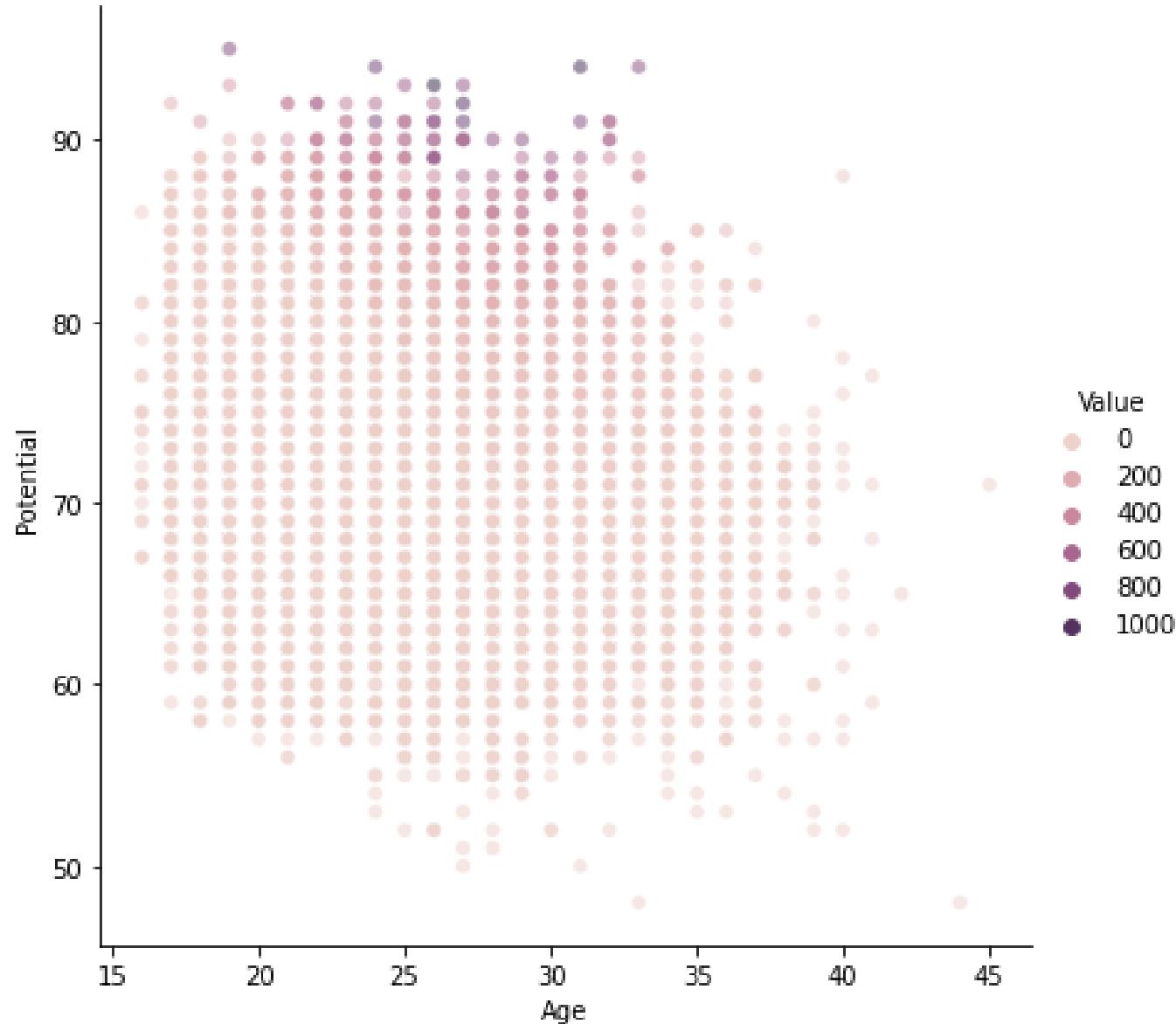
# SEGMENTACIÓN DE DATOS

En esta oportunidad segmentamos o separamos las variables que estaban demás para así dejar solamente las variables que vamos a trabajar para ello creamos un dataframe nuevo seleccionando así las variables mas relevantes

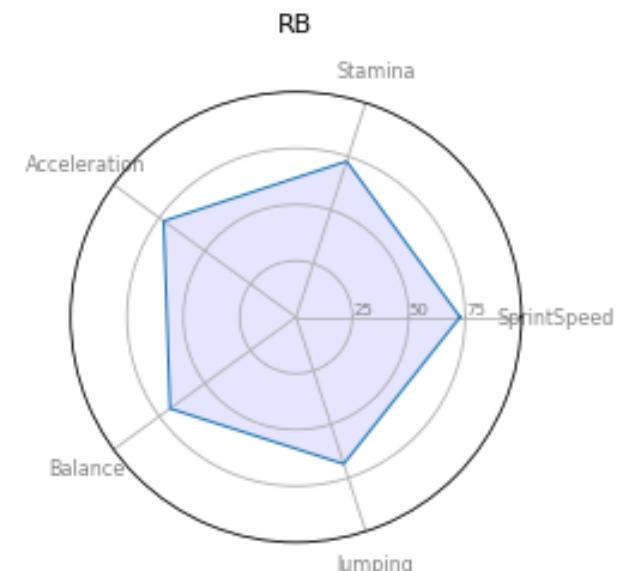
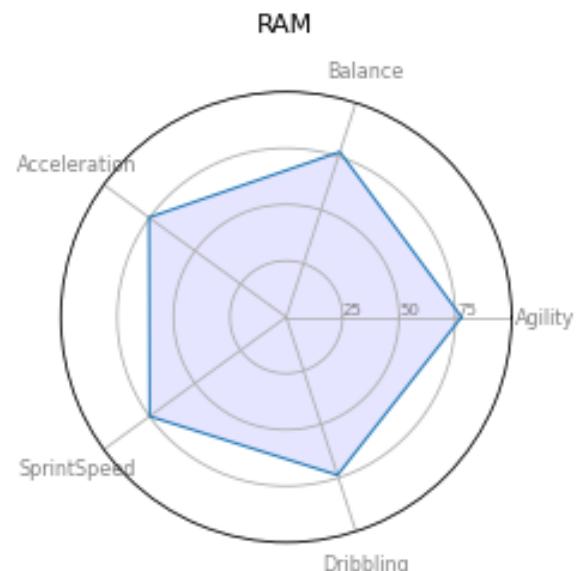
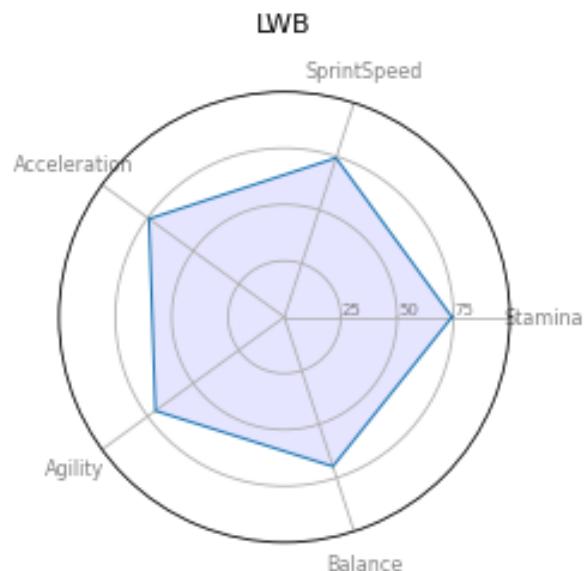
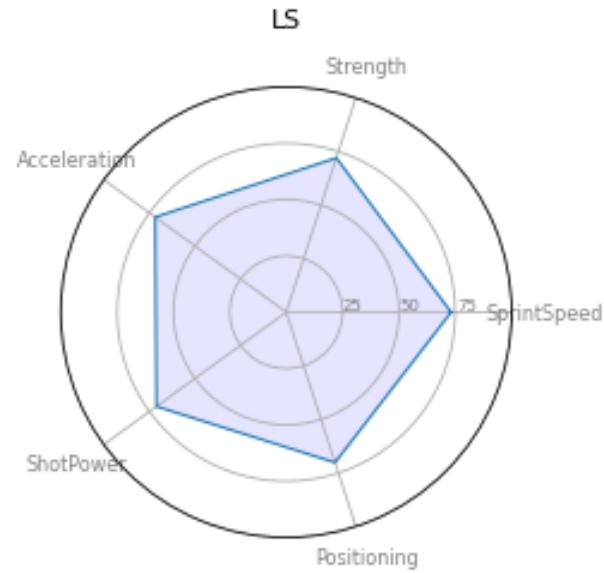
# 04 ANALISIS MULTIVARIADO

Luego de identificar las variables que target que tienen correlación decidimos graficar la distribución por area de cada característica dependiendo de la posición del jugador

# Análisis de Edad vs Potencial



# Distribución por Área de cada característica por posición de jugador



05

# PREDICCIÓN DE POSICIONES DE LOS JUGADORES

Ya seleccionadas las variables target y ordenadas vamos por nuestro objetivo que es Predecir la posición de un jugador dado una serie de atributos distribuidos entre :

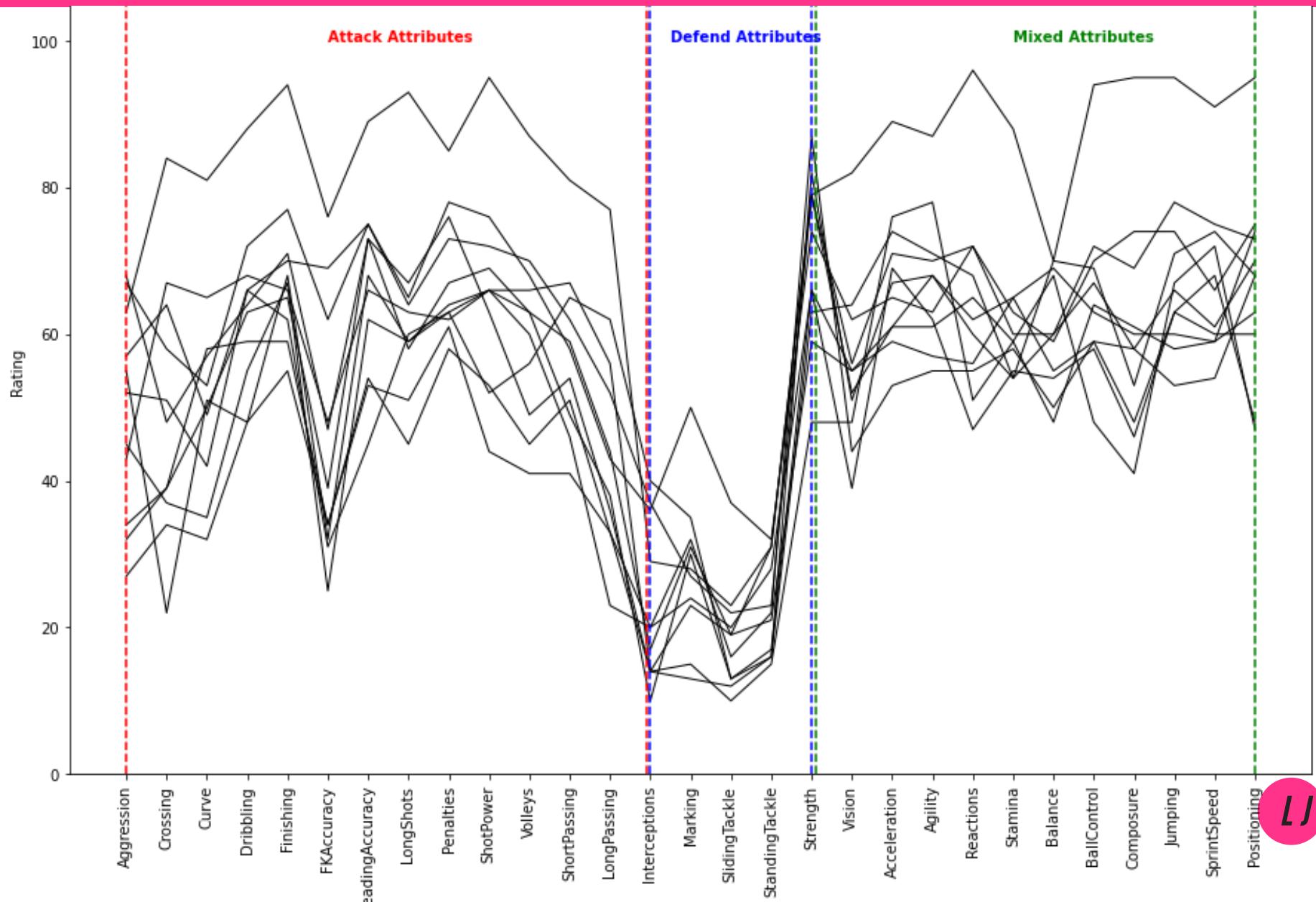
- Ofensivos
- Defensivos
- Mixtos

# 06

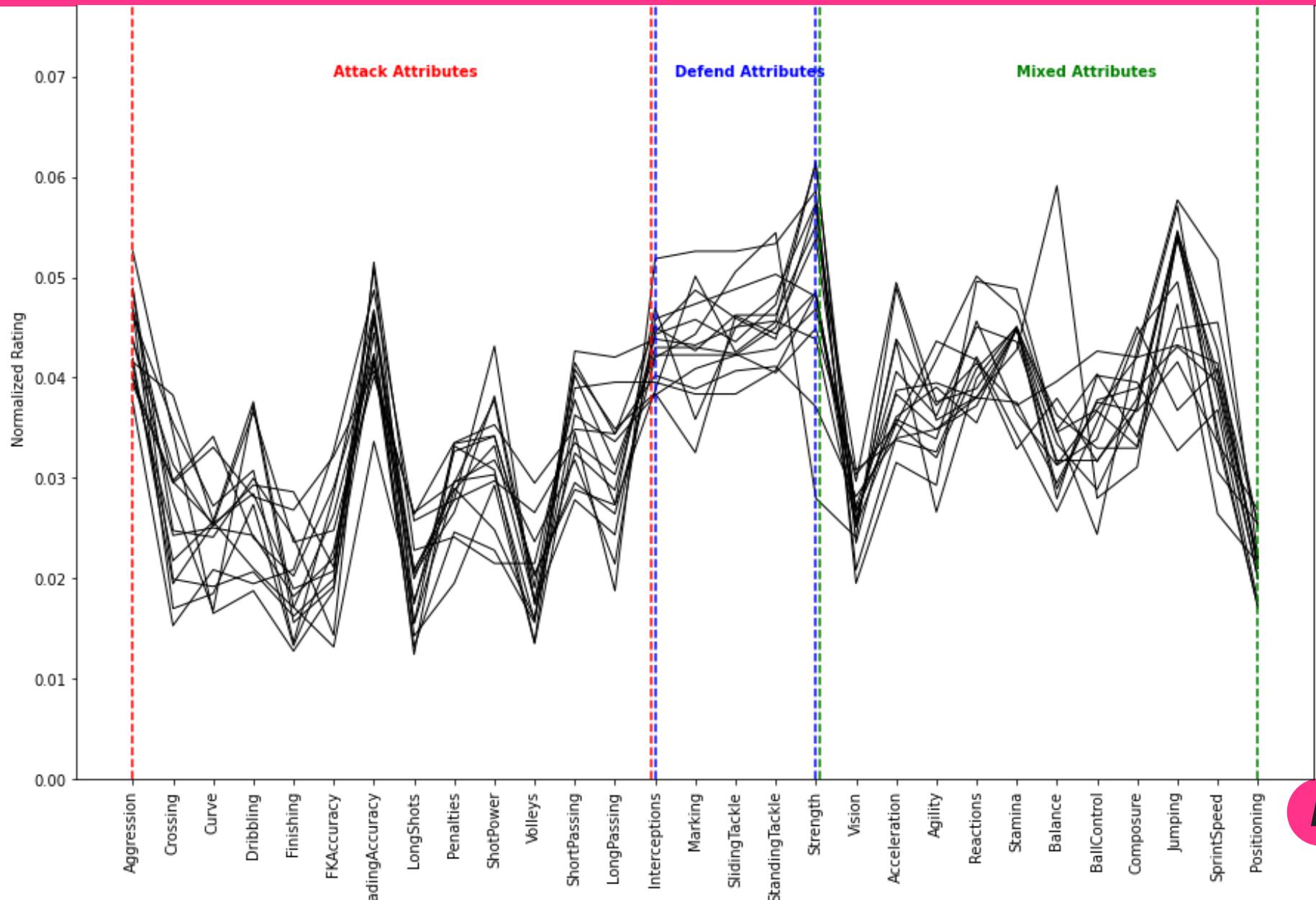
## NORMALIZACIÓN DE DATOS

Verificaremos el patrón de atributos tomando como muestra 200 columnas de nuestro Dataframe

# Distribución de Atributos (sin normalizar)



# Distribución de Atributos (Ya normalizados)



07

# PREDICCIÓN POR REGRESIÓN LOGÍSTICA

Los datos, después de la normalización, se verifican mas acorde a las posiciones de los jugadores como se demostro.

Vamos a normalizar todo el conjunto de datos

Reclasificaremos el valor target (posiciones) en grupos binarios como se muestra a continuación:

1 = posiciones de ataque = RF, ST, RW, LW, RM, CM, LM, CAM, CF, RCM, LF, RS

0 = posiciones defendidas = CDM, CB, LB, RB, RWB, LWB, RCB, LCM, LS, LCB, LAM, RDM, RAM

# 08 IMPORTAMOS LIBRERÍAS

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.dummy import DummyClassifier
```

# 09

## ENTRENAMOS EL MODELO

```
#Entrenamos el modelo
```

```
X_train, X_test, y_train, y_test = train_test_split(df_new_normalized.iloc[:, :-1], df_new_normalized.iloc[:, -1], random_state=0)

print('X train shape: {}'.format(X_train.shape))
print('X test shape: {}'.format(X_test.shape))
print('y train shape: {}'.format(y_train.shape))
print('y test shape: {}'.format(y_test.shape))
```

# 10

# APLICANDO REGRESIÓN LOGISTICA AL MODELO

```
clf_d = DummyClassifier(strategy = 'most_frequent').fit(X_train, y_train)
acc_d = clf_d.score(X_test, y_test)
print ('Dummy Classifier (most frequent class): {}'.format(acc_d))

clf = LogisticRegression().fit(X_train, y_train)
acc = clf.score(X_test, y_test)
print ('Logistic Regression Accuracy: {}'.format(acc))

Dummy Classifier (most frequent class): 0.4963386727688787
Logistic Regression Accuracy: 0.8725400457665904
```

# ¿PODREMOS MEJORAR AÚN MÁS LA PRESICIÓN DEL ALGORITMO?

¡¡¡VEAMOSLO!!!

11

# LIMITANDO LOS ATRIBUTOS

Limitaremos los atributos de 28 a 15 ya que  
consideramos que son los principales así mejoraremos  
la exactitud de nuestro algoritmo

# LIMITANDO LOS ATRIBUTOS

```
64]:  
target_cols = Coef_table[:15]['Attributes'].tolist()  
  
clf_2 = LogisticRegression().fit(X_train[target_cols], y_train)  
acc_2 = clf_2.score(X_test[target_cols], y_test)  
print ('Logistic Regression Accuracy (15 features): {}'.format(acc_2))
```

```
Logistic Regression Accuracy (15 features): 0.8734553775743708
```

La precisión ha mejorado poco. esto se debe a que la distribucion de ciertos atributos es bastante uniforme entre atacante, defensa, mixtos.

Observaciones: -el atributo de Positioning es bastante significativo en el modelo y probablemente no debería clasificarse en atributos mixtos, mientras que algunos atributos de ataque como el Crossing no son tan significativos (probablemente porque LB / RB también tienen índices de cruce altos)

-Los atributos más significativos son en su mayoría defensivos, lo que significa que son más útiles para ayudarnos a identificar el rol de un jugador. En otras palabras, es extremadamente raro que un jugador en un rol ofensivo como ST sea bueno en esas habilidades defensivas, mientras que es relativamente más fácil para un jugador en un rol defensivo ser bueno en algunas habilidades de ataque como el centro y la precisión de los tiros libres. , etc.

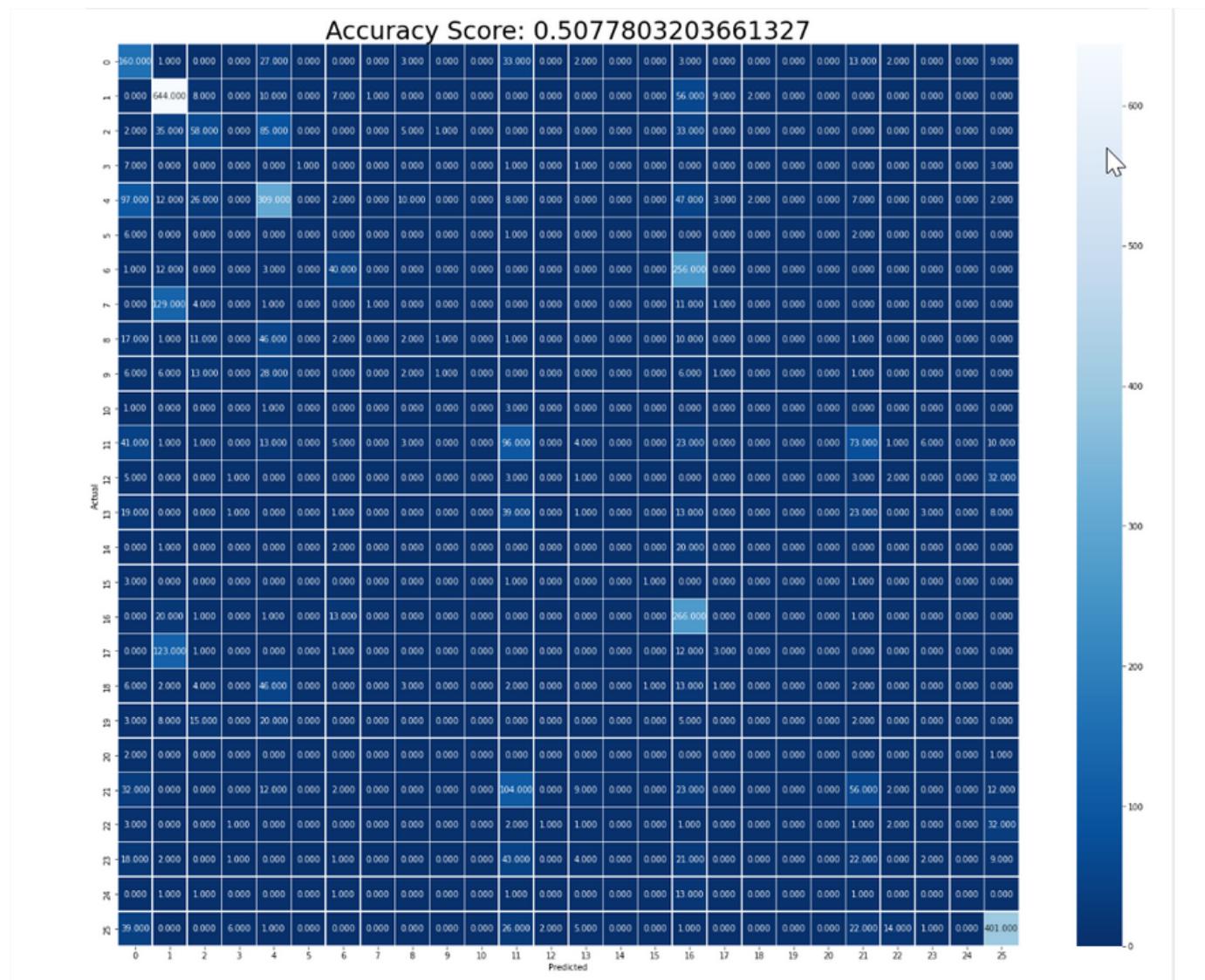


12

# PROBANDO CON DISTINTOS ALGORITMOS

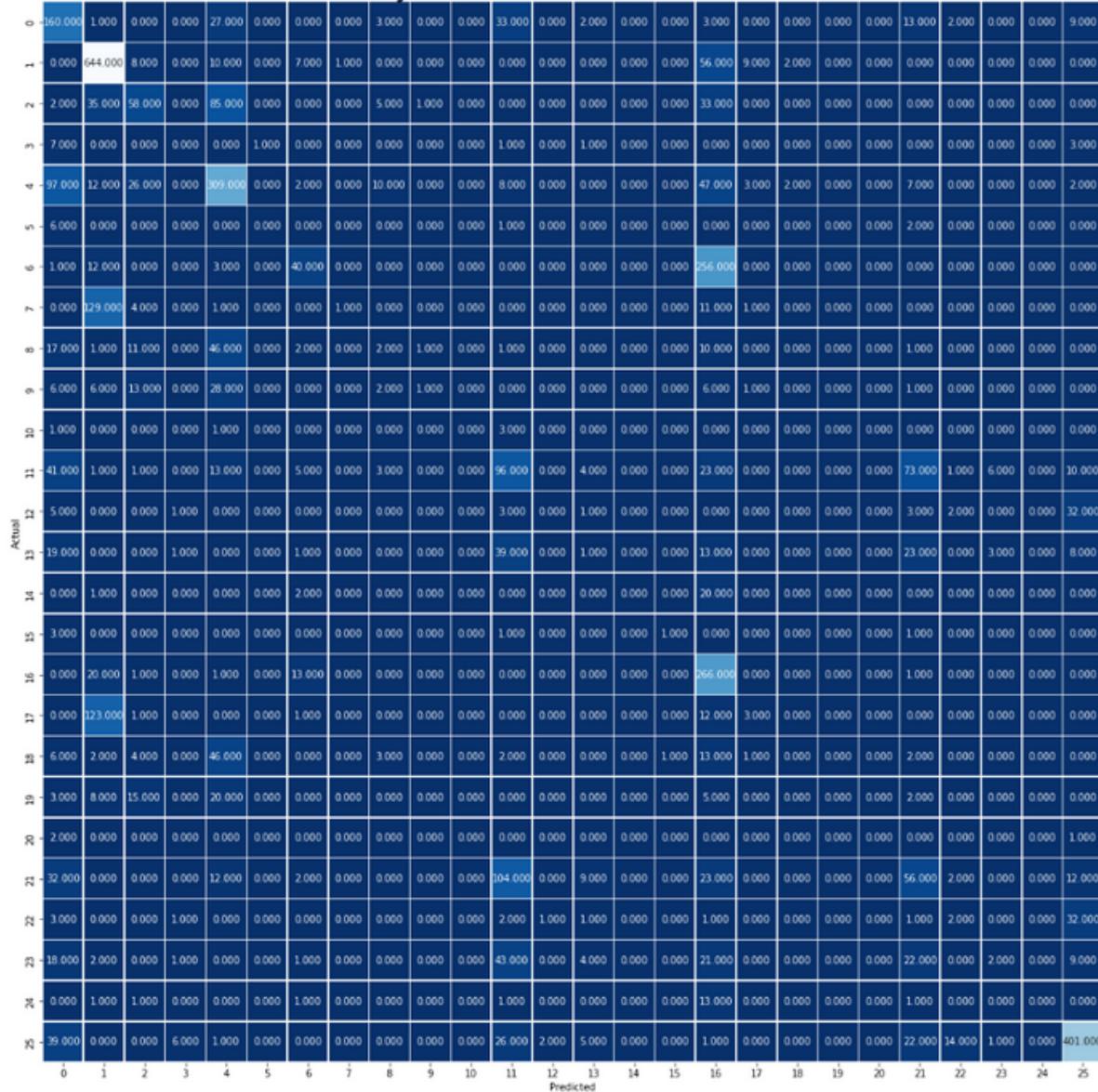
Probaremos con distintos algoritmos de predicción  
para así poder tener la mejor conclusión

# USANDO REGRESIÓN LOGÍSTICA

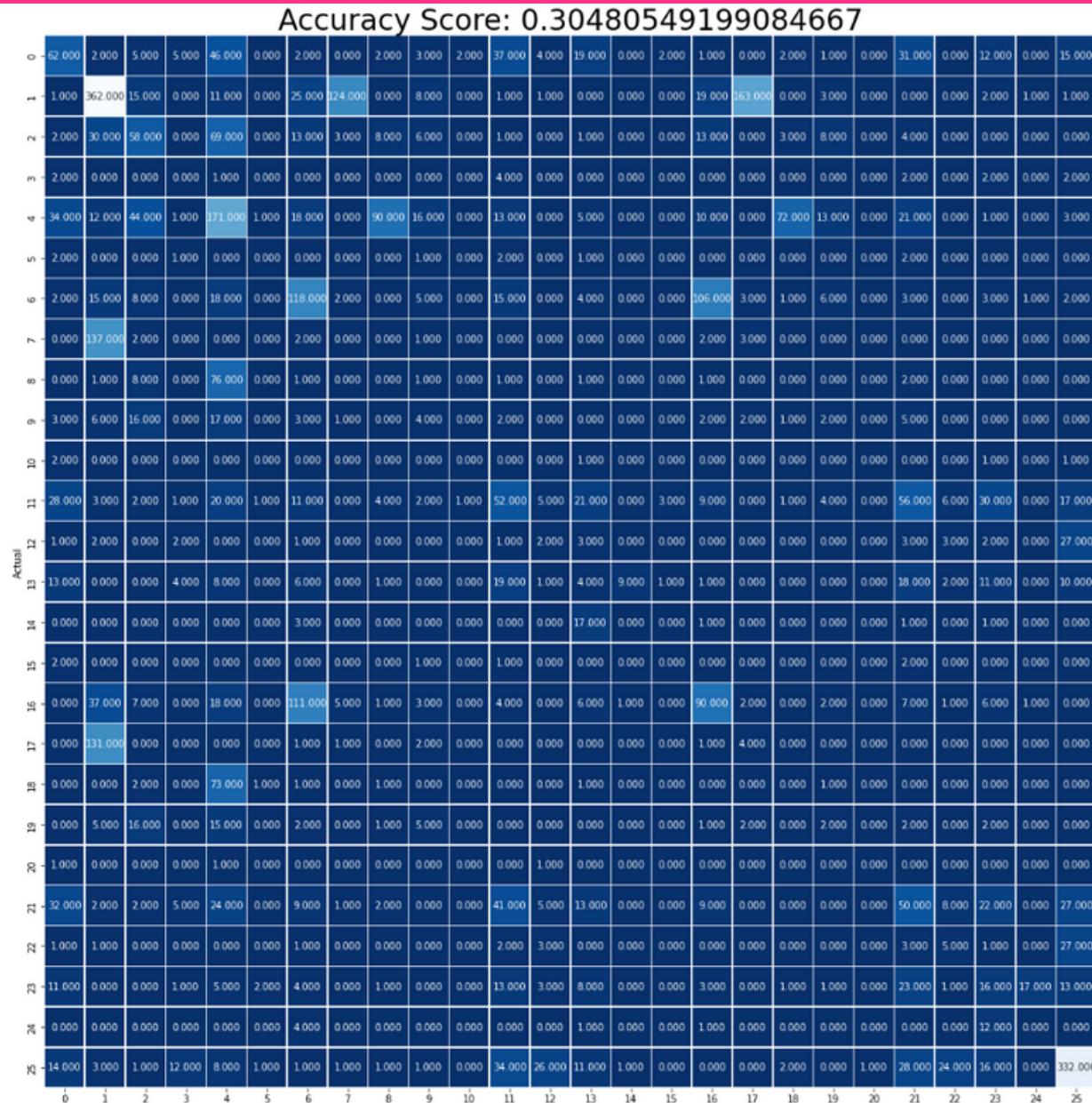


# REDES NEURONALES

Accuracy Score: 0.46750572082379865



# DECISION TREES



-350  
-300  
-250  
-200  
-150  
-100  
-50  
0

LJ

# 13

## MEJORANDO Y OPTIMIZANDO EL MODELO ELEGIDO

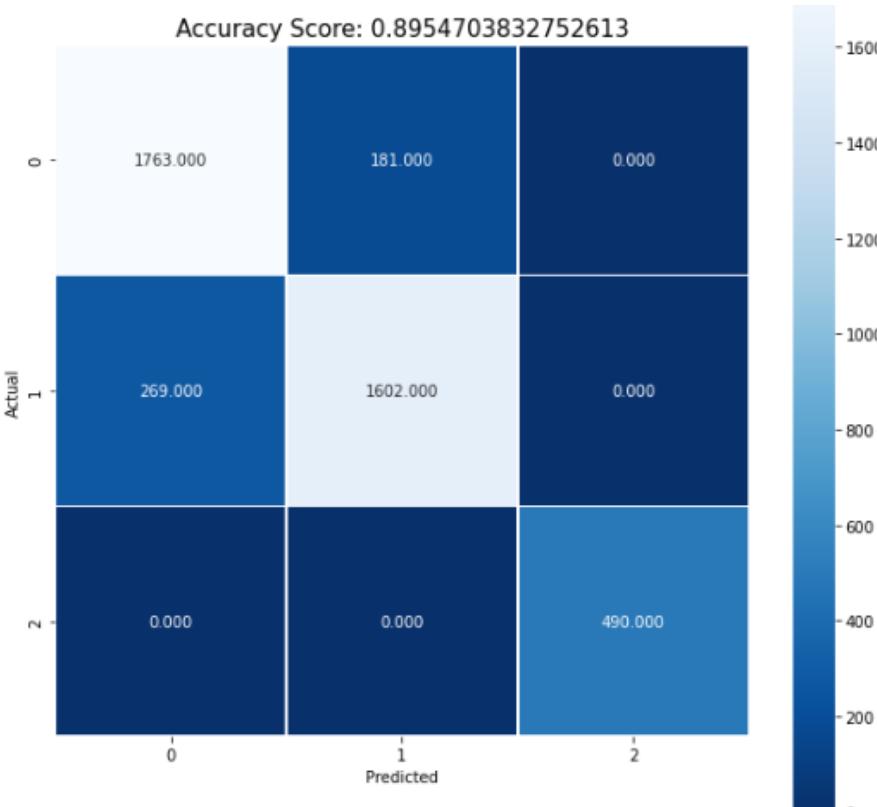
Como recordaremos en el grafico de distribucion de la variable posicion de jugador en el dataset veiamos que la posicion GK (arquero) era representativa de la masa de datos, por lo cual nos lleva a la siguiente interrogante,



¿Será esa posición una variable sensible que nos pueda ayudar a mejorar nuestro modelo de predicción?

¡VEAMOS!

# MEJORANDO Y OPTIMIZANDO



Vemos que el modelo alcanzó un **90%** de predicción, tal como lo habíamos anticipado, el alto número de variable GK en el dataset es altamente sensible a la predicción de 3 variables (atacantes, defensores y Arqueros).



# 14

## HORA DE PONER A PRUEBA NUESTRO MODELO



Es hora de poner a prueba nuestro modelo, para ello vimos que alcanzamos un 90% aplicando regresion logistica sobre 3 target (atacante, defensa, arquero), aplicaremos este modelo para crear un jugador personalizado que en base a sus atributos responda a la mejor posicion dentro del campo de juego.

# APLICANDO REGRESIÓN LOGÍSTICA

In [99]:

```
test
```

Out[99]:

	GKDiving	GKHandling	GKKicking	GKPositioning	GKReflexes	Aggression	Crossing	Curve	Dribbling	Finishing	...	Acceleration	Agility	Reactions	Stamina	Balance	Ball
139	11	14	12	7	15	40	85	78	107	80	...	73	72	81	66	72	

1 rows × 34 columns

In [100...]

```
predict_position = logistic_regression_allp.predict(test)
```

In [101...]

```
position_mapper = {0:"Defensor",1:"Atacante",2:"GK"}
```

In [102...]

```
position_mapper[predict_position[0]]
```

Out[102...]

```
'Atacante'
```

Como vemos en base a los atributos representativos de las posiciones Atacante, defensor y Arquero, el modelo es capaz de predecir su posición.

NO ES MAGIA... ES DATA SCIENCE



LJ



# CONCLUSIONES

# Y FUTURAS LINEAS

Conforme a lo desarrollado podemos concluir que el modelo es fuerte en su predicción de posición de jugadores cuando todo el conjunto de jugadores son tomados en cuenta, dando como referencia 3 posiciones posibles (atacantes, defensores y arquero). Hemos de verificar que la variable Posición está muy dispersa afectando enormemente al modelo, debemos aconsejar al equipo de desarrollo desestimar ciertas posiciones de jugadores, o bien aglutinarlas en alguna representativa, ya que a nuestro entender, no aportan a la mecánica del juego. Los atributos están bien distribuidos siendo esto el principal aspecto a tener en cuenta para poder designar a cada jugador su posición más óptima dentro del campo de juego. Estamos seguros de que una mejora del modelo podrá lograrse con una mejor asignación de atributos a una menor cantidad de posiciones posibles (actualmente son 27 posiciones) Dando por finalizado nuestra exposición, esperemos aportar información relevante a la Gerencia para futuros lanzamientos del videojuego FIFA.