ENTREGA FINAL.

Predicción de datos usando modelos de data science

ABSTRACT.

La ciencia de datos juega un papel preponderante y en crecimiento en cuanto al análisis y predicciones de los resultados de los eventos deportivos. Estos se usa principalmente en el fútbol europeo donde casi todos los equipos y selecciones del continente, tienen departamentos de datos dedicados a identificar jugadores, estudiar lesiones y el reclutamiento de nuevos talentos. También, se ve mucha aplicación en el básquet y béisbol en EE.UU. por lo mismos motivos.

En el mundo de tenis, los datos están presentes como algo descriptivo del juego y que, hace años, está presente en las transmisiones de TV como en las páginas web del circuito profesional. Actualmente, la data de los partidos se usa mucho para las casas de apuestas para determinar las probabilidades del ganador del partido. No se usa mucho para el desarrollo del club o equipo como se usa en el fútbol, todavía es algo "verde".

Cómo amante del tenis, busco crear un modelo que me permita saber de antemano, cuál es el jugador que tiene mas chances de ganar un partido.

En nuestro caso, busco hacer una análisis completo de cada perfil de jugador y entender cuáles son los mejores match-ups y que probabilidades tienen de ganar.

TIPOS DE DATOS

w (Ganador) – l (Perdedor)

0	w_ace	998 non-null	int64: Aces
1	w_df	998 non-null	int64: Dobles faltas
2	w_svpt	998 non-null	int64: Puntos ganados con el primer saque
3	w_1stIn	998 non-null	int64: Cantidad de puntos jugados con el primer saque
4	w_1stWon	998 non-null	int64: Cantidad de puntos ganados con el primer saque
5	w_2ndWon	998 non-null	int64: Cantidad de puntos ganados con el segundo saque
6	w_SvGms	998 non-null	int64: Cantidad de games jugados con el saque
7	w_bpSaved	998 non-null	int64: Break points salvados
8	w_bpFaced	998 non-null	int64: Break points enfrentados
9	l_ace	998 non-null	int64: Aces
10	l_df	998 non-null	int64: Dobles faltas
11	l_svpt	998 non-null	int64:Puntos ganados con el primer saque
12	l_1stIn	998 non-null	int64:Cantidad de puntos jugados con el primer saque
13	l_1stWon	998 non-null	int64: Cantidad de puntos ganados con el primer saque
14	l_2ndWon	998 non-null	int64: Cantidad de puntos ganados con el segundo saque
15	l_SvGms	998 non-null	int64:Cantidad de games jugados con el saque
16	l_bpSaved	998 non-null	int64: Break points salvados
17	l_bpFaced	998 non-null	int64: Break points enfrentados
18	w_1serve_per	998 non-null	int32: porcentaje de aciertos con el primer saque
19	w_1servewon_per	998 non-null	int32: porcentaje de puntos ganados con el primer saque
20	l_1servewon_per	998 non-null	int32: porcentaje de puntos ganados con el primer saque
21	break_points_made	998 non-null	int32: Break points hechos
22	break_points_recieved	998 non-null	int32: Break points recibidos.

INTRODUCCIÓN

Para el caso de este análisis preliminar, haremos un breve EDA con un jugador conocido: Novak Djokovic.

Usaremos un dataset ya modificado en entragas anteriores, con el data wrangling ya hecho.

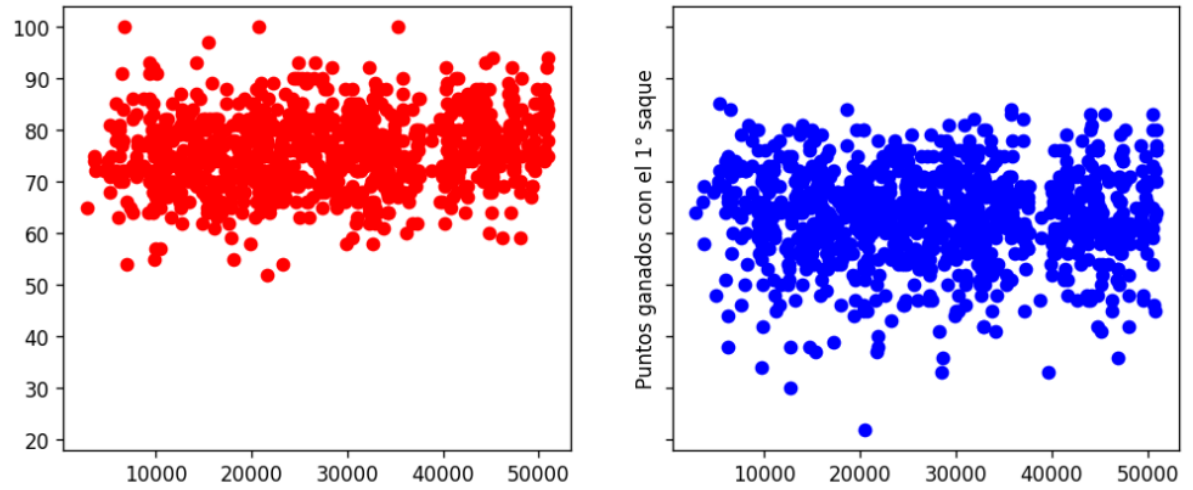
Tomaremos los partidos donde Novak haya jugado. Buscamos tanto del lado del ganador como perdedor.

Hacemos una descripción de los datos brindados de sus triunfos. El énfasis va a ser los campos w_ y l_ ya que indican los números del ganador (w_ > Novak Djokovic) contra los del perdedor (l_ > Rivales). Esto nos dará un detalle a priori, de los datos que son importantes a analizar.

HIPÓTESIS: PORCENTAJES DE PRIMEROS SAQUES.

Una de las primeras cuestiones a revisar, es la efectividad del saque. Es necesario tener un porcentaje alto de primeros saques para ganar los partidos. Lo comparamos con el de sus rivales. Por lo general, un buen porcentaje de primeros saques, implica que toma la iniciativa en los puntos que dependen de el. Además, suele ser mucho mas fuerte que el segundo saque y lleva mas riesgos a sus rivales, ya que se les dificulta devolverlos.

```
fig, ax = plt.subplots(1,2,sharex=True, sharey=True, dpi=120, figsize=(10, 4))

y1 = df_novak_1['w_1servewon_per']
x1 = df_novak_1.index

y2 = df_novak_1['l_1servewon_per']
x2 = df_novak_1.index

ax[0].scatter(x1, y1, color='red')
ax[1].scatter(x2, y2, color='blue')

#plt.scatter(x, y)
plt.xlabel('Partido')
plt.ylabel('Puntos ganados con el 1° saque')
plt.show()
"Cantidad de puntos convertidos con el primer saque."
```



HIPÓTESIS: SUPERFICIES.

Determinamos en las superficies donde mayor cantidad de victorias ha conseguido.

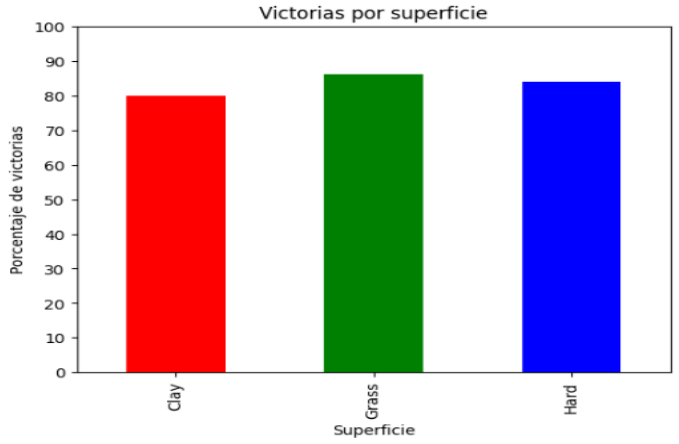
Si bien la mayoría de los torneos se juegan en superficie dura, hay jugadores donde se destacan mejor en arcilla o césped.

Una superficie favorable para el jugador, da una ventaja importante.

Se descarta la superficie Carpet porque, desde 2009, no se usa.

```
counts_vic_1 = df_novak_1['surface'].value_counts()
counts_der_1 = df_novak_2['surface'].value_counts()

dict_colors = {'Clay': 'r', 'Grass':'g', 'Hard':'b'}
suma_count_1 = counts_vic_1 + counts_der_1
por_victories_sup_1 = (((counts_vic_1/suma_count_1)*100).astype(int))
por_victories_sup_1 = por_victories_sup_1.drop('Carpet')
print(por_victories_sup_1.sort_index())

lista_ticks = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

por_victories_sup_1.sort_index().plot.bar(x=list(dict_colors.keys()), color=dict_colors.values())

plt.xlabel('Superficie')
plt.ylabel('Porcentaje de victorias')
plt.title('Victorias por superficie')
plt.yticks(lista_ticks)
plt.show()
```

```
surface
Clay     80
Grass    86
Hard     84
Name: count, dtype: int32
```



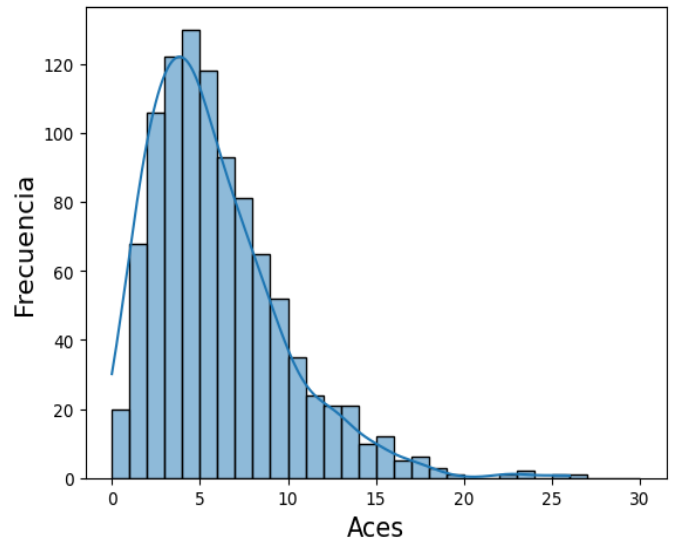
HIPÓTESIS: ACES.

Otra variable a tener en cuenta es la cantidad de aces. En algunos jugadores, es algo determinante ya que pueden definir su estilo de juego.

No suele ser algo tan determinante ya que, hay mucho riesgo involucrado por fallar el primero y asegurar el segundo saque.

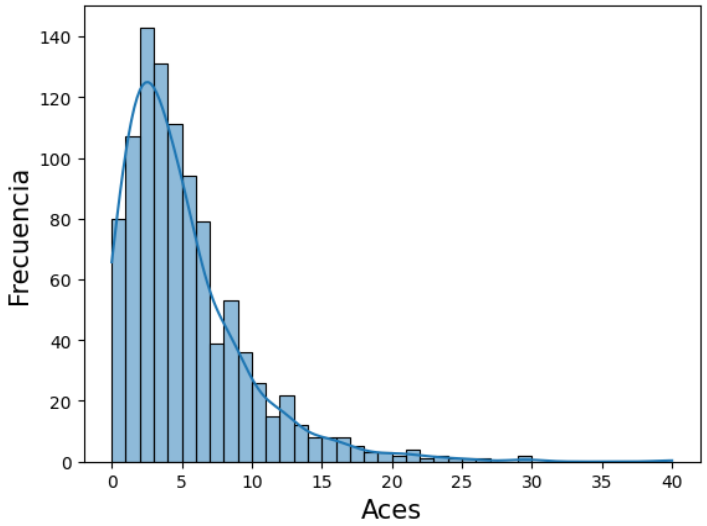
```
chart = sns.histplot(data=df_novak_1, x="w_ace", bins=30, kde=True, binrange=[0,30])
chart.set_xlabel('Aces', fontdict={'size': 15})
chart.set_ylabel('Frecuencia', fontdict={'size': 15})
```

```
Text(0, 0.5, 'Frecuencia')
```




```
chart = sns.histplot(data=df_novak_1, x="l_ace", bins=30, kde=True, binrange=[0,30])
chart.set_xlabel('Aces', fontdict={'size': 15})
chart.set_ylabel('Frecuencia', fontdict={'size': 15})
```

```
Text(0, 0.5, 'Frecuencia')
```

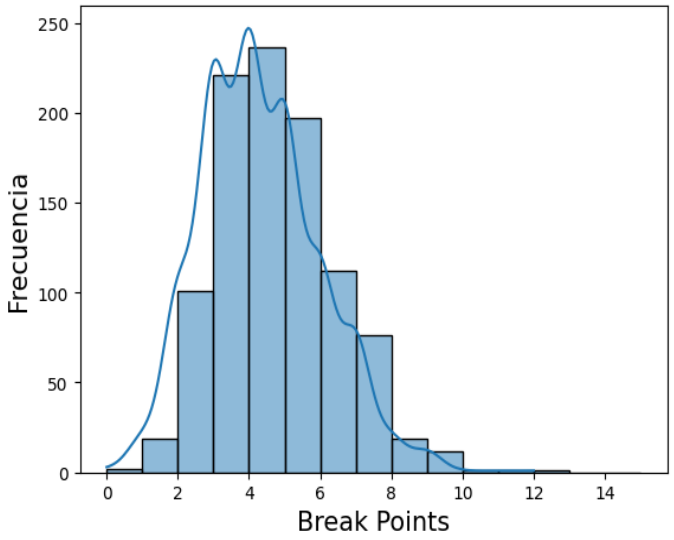


HIPÓTESIS: BREAK POINTS.

Break points son aquellos puntos decisivos que se ganan contra el saque del rival. Esto es un indicador de un jugador que juega muy bien cuando recibe el saque de rival y es capaz de ganarle games sin su saque. Conseguir break points es fundamental para romper el ciclo normal del set.Es un factor muy importante. También, no recibir break points en contra.¶

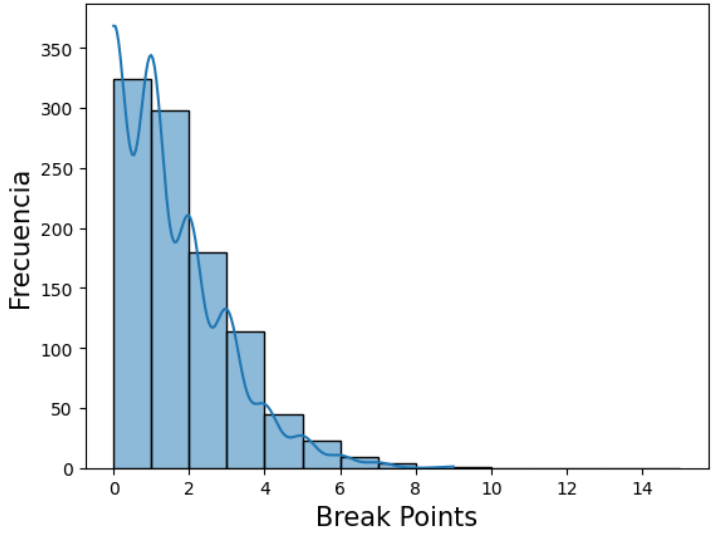
```
chart = sns.histplot(data=df_novak_1, x="break_points_made", bins=15, kde=True, binrange=[0,15])
chart.set_xlabel('Break Points', fontdict={'size': 15})
chart.set_ylabel('Frecuencia', fontdict={'size': 15})
```

```
Text(0, 0.5, 'Frecuencia')
```




```
chart = sns.histplot(data=df_novak_1, x="break_points_recieved", bins=15, kde=True, binrange=[0,15])
chart.set_xlabel('Break Points', fontdict={'size': 15})
chart.set_ylabel('Frecuencia', fontdict={'size': 15})
```

```
Text(0, 0.5, 'Frecuencia')
```



INSIGHTS

La mayor ventaja la saca con los puntos ganados con el primer servicio. Es el factor más importante de su juego.

El porcentaje de primeros saques es más alto que el de su rivales.

En general, está entre 10% a 15% mas alto que el de sus rivales. En algunos casos es mucho mayor. Djokovic genera un gran resultado con sus primeros saques. Es un claro indicador de que influye en las victorias.

El porcentaje victorias en las superficies son muy parejos. Rinde muy bien en cualquier superficie.

La mayor cantidad de victorias se da en césped.

INSIGHTS

En los aces, no saca mucha ventaja. Sin embargo, es ligeramente superior que sus rivales.

La cantidad de aces es ligeramente superior a la de sus rivales. No parece influir mucho en la victoria.

Sin embargo, hace una pequeña diferencia durante el juego.

En los break points, saca ventaja sobre sus rivales. La diferencia de puntos le garantiza, al menos, un set.

Djokovic saca una gran ventaja en brak points en sus victorias. En contrapartida, recibe pocos durante sus victorias.

MODELO: RIVAL

Para el modelo, se agrega a un rival para hacer la comparativa entre dos tenistas. En este caso, se agrega a Rafael Nadal al modelo y se hace el mismo EDA que a Novak Djokovic.

MODELO: PREPARACION

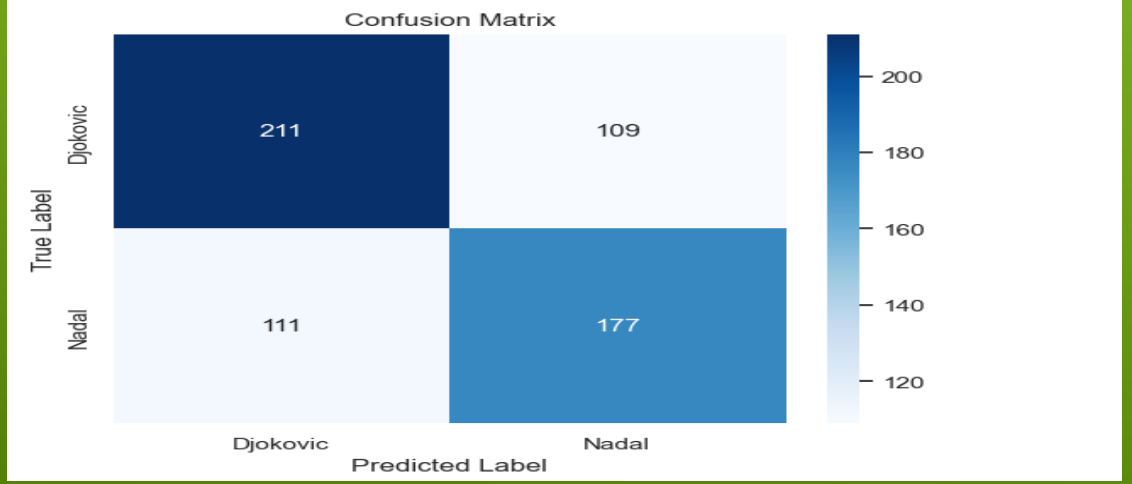
Para el modelo, se eliminan las columnas irrelevantes y que no tengan un dato numérico.

Se toma como valor el la columna target para clasificar los datos, dependiendo de cada jugador.

MODELO: SELECCION

Se toma el modelo de RandomForestClassifier para hacer el modelo de predicción. Luego de hacer una matriz de confusión, arroja estos resultados.

```
cm = confusion_matrix(y_test, y_pred)

cm_df = pd.DataFrame(cm, index=["Djokovic", "Nadal"], columns=["Djokovic", "Nadal"])

fig, ax = plt.subplots()

sns.heatmap(cm_df, annot=True, fmt="d", cmap="Blues")

ax.set_xlabel("Predicted Label")
ax.set_ylabel("True Label")
ax.set_title("Confusion Matrix")
plt.show()
```



MODELO: MEJORAS

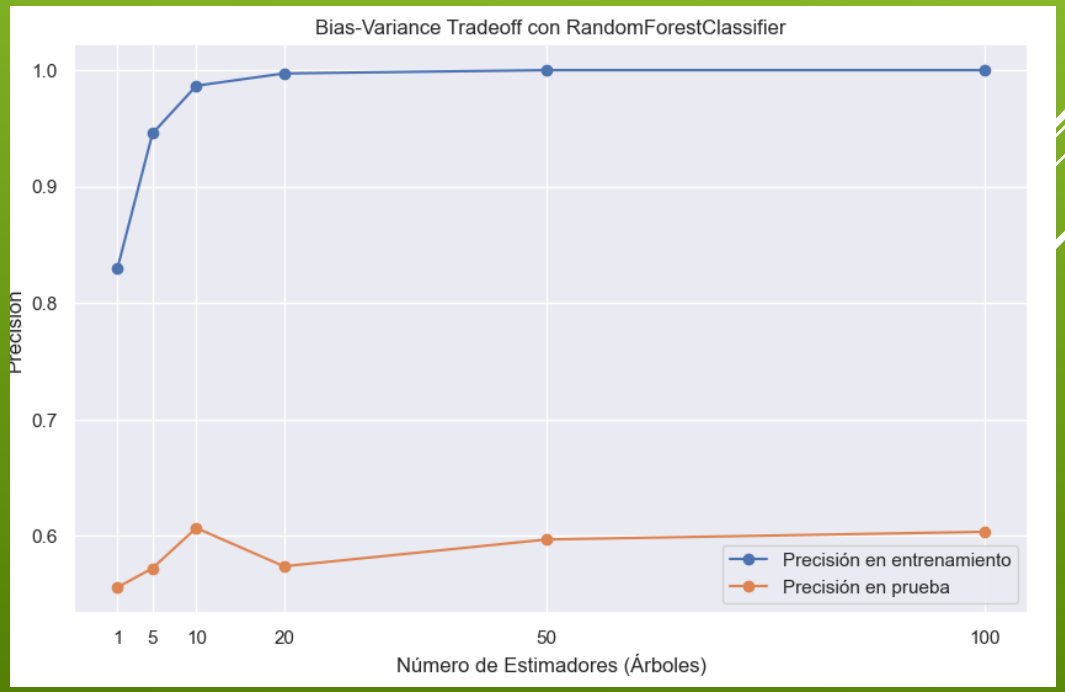
Se aplica Bias-Variance Tradeoff para determinar la presencia de underfitting o overfitting.

El modelo presenta overfitting por la diferencia entre las precisiones. Ajustamos el modelo para trabajar con un número menor de profundidad y estimadores. Además, sacamos algunas variables que no son importantes en el modelo y que generan "ruido".

A su vez, se le aplica StandardScaler y PCA para la estadarización de la muestras.

MODELOS: MEJORAS

Resultados del bias-variance tradeoff.



MODELO: MEJORAS

Luego, se aplica el cross-validation para ver la mejora

Gracias al cross-validation, la precisión mejora. Al 66%.

```
Fold 1: Score = 0.69
Fold 2: Score = 0.66
Fold 3: Score = 0.65
Fold 4: Score = 0.63
Fold 5: Score = 0.66
Mean Score: 0.66
Standard Deviation: 0.02
```

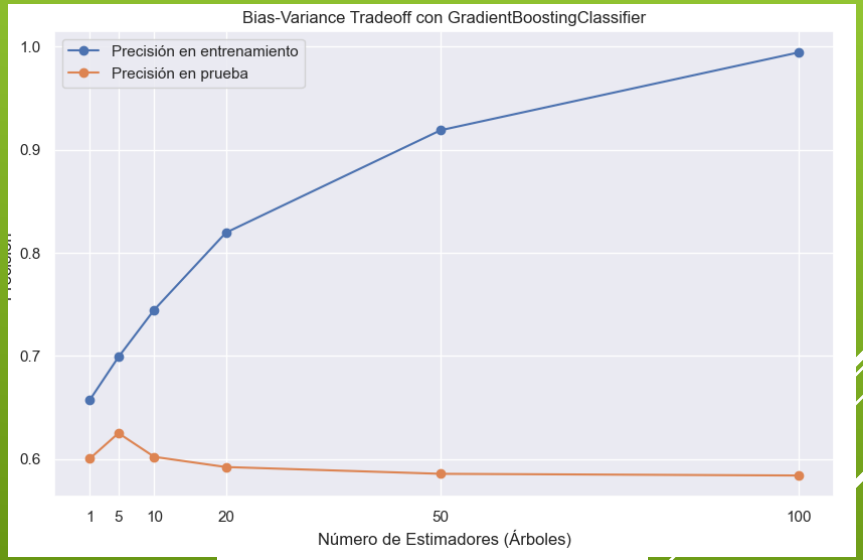
MODELOS: OTRA PRUEBA

Ahora, aplicamos el GradientBoosting para determinar si obtenemos un mejor resultado que con RandomForestClassifier.

Lo hacemos usando las mismas métricas que las realizadas por el RandomForestClassifier, teniendo en cuenta.

MODELOS: OTRA PRUEBA

Resultados del GradientBoostingClassifier.




```
Fold 1: Score = 0.60
Fold 2: Score = 0.59
Fold 3: Score = 0.63
Fold 4: Score = 0.56
Fold 5: Score = 0.56
Mean Score: 0.58
Standard Deviation: 0.03
```

CONCLUSIONES

Por las métricas de precision, acurracy, mean, std y cross validation, el RandomForestClassifier es mejor modelo que el GradientForestClassifier para este dataset.

Según la matriz de confusión, Djokovic tiene más chances de ganar que Nadal en un enfrentamiento.