

# TRABAJO FINAL DE FUNDAMENTOS DE MACHINE LEARNING

Docente: Ruth Chirinos

Fecha de Entrega: 23/Sept/2023 23:59:59

\*\*

---

ENTREGAR EL NOTEBOOK EJECUTADO EN FORMATO ipynb y PDF. Trabajos sin ambos archivos ejecutados no tendran calificacion

---

\*\*

## ▼ Árboles de Decision en problemas de Clasificacion

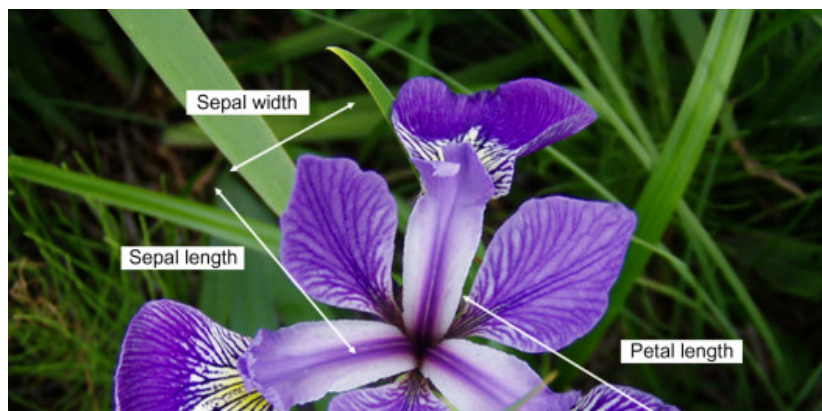
```
1
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 %matplotlib inline

1 from sklearn.preprocessing import LabelEncoder #para la division de conjunto de datos de testeo y training
2 from sklearn.model_selection import train_test_split #para el objeto de decision tree
3 from sklearn.tree import DecisionTreeClassifier #para verificar los resultados de testeo
4 from sklearn.metrics import classification_report, confusion_matrix #para visualizar el arbol
5 from sklearn.tree import plot_tree
```

## ▼ Dataset

El conjunto de datos de la flor de Iris es un conjunto de datos multivariado introducido por el estadístico y biólogo británico Ronald Fisher en su artículo de 1936 El uso de mediciones múltiples en problemas taxonómicos. A veces se le llama conjunto de datos de Iris de Anderson porque Edgar Anderson recopiló los datos para cuantificar la variación morfológica de las flores de Iris de tres especies relacionadas. El conjunto de datos consta de 50 muestras de cada una de las tres especies de Iris (Iris Setosa, Iris virginica e Iris versicolor) dando un total de 150 registros. De cada muestra se midieron cuatro características: el largo y el ancho de los sépalos y pétalos, en centímetros.

```
1 # Entendiendo el largo y el ancho de los sépalos y pétalos
2 # -----
3 from IPython.display import Image
4 Image(url='https://ars.els-cdn.com/content/image/3-s2.0-B9780128147610000034-f03-01-9780128147610.jpg')
```



## ▼ Analisis y Prediccion



```
1 # Damos Lectura del Dataset
2 # -----
3 df = sns.load_dataset('iris')
4
```

```
1 # EJERCICIO 1
2 # Liste las primeras 5 filas del dataset
3 df.head(5)
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Gracias a la libreria de 'seaborn' podras acceder al dataset de iris. Cuando observamos los datos que contiene el dataset podemos apreciar 4 columnas de características: sepal\_length, sepal\_width, petal\_length, and petal\_width respectivamente con una columna 'target' (objetivo) llamada 'species'. las primeras 4 características nos permiten identificar la especie de cada flor.

```
1 # EJERCICIO 2
2 #Obtenemos la informacion del dataset
3 # -----
4 df.info
```

```
<bound method DataFrame.info of      sepal_length  sepal_width  petal_length  petal_width  species
0          5.1         3.5         1.4         0.2     setosa
1          4.9         3.0         1.4         0.2     setosa
2          4.7         3.2         1.3         0.2     setosa
3          4.6         3.1         1.5         0.2     setosa
4          5.0         3.6         1.4         0.2     setosa
..          ...         ...         ...         ...         ...
145         6.7         3.0         5.2         2.3  virginica
146         6.3         2.5         5.0         1.9  virginica
147         6.5         3.0         5.2         2.0  virginica
148         6.2         3.4         5.4         2.3  virginica
149         5.9         3.0         5.1         1.8  virginica
```

```
[150 rows x 5 columns]>
```

```
1 # EJERCICIO 3
2 # Obtenemos la cantidad de filas y columnas usando las funciones del dataframe
3 # -----
4 print(f"EL DATAFLOW CONTIENE UNA DIMENSION DE {df.shape} QUE CORRESPONDE A FILAS Y COLUMNAS DEL DATASET ")

EL DATAFLOW CONTIENE UNA DIMENSION DE (150, 5) QUE CORRESPONDE A FILAS Y COLUMNAS DEL DATASET
```

## ▼ Analisis EDA


Ahora realizamos algunos EDA (Exploratory Data Analysis/Análisis Exploratorio de Datos) básicos en este conjunto de datos. Comprobemos la correlación de todas las características entre sí.

```
1 # Vamos a analizar las características del dataset revisando su distribución
2 # -----
3 sns.pairplot(data=df, hue = 'species')
4 # Donde:
5 # - hue = Variable en datos para asignar aspectos de la trama a diferentes colores.
```

```
<seaborn.axisgrid.PairGrid at 0x78a2f8631e70>
```

En la grafica confirmamos la existencia de las 3 especies:

- setosa
- versicolor
- virginica



```
1 # Vamos a apoyarnos usando la Matriz de Correlacion para continuar el analisis
2 # -----
3 '''
4 Una matriz de correlación es una herramienta estadística que muestra la intensidad
5 y la dirección de la relación entre dos o más variables. Se utiliza mucho en campos
6 como las finanzas, la economía, la psicología y la biología, porque ayuda a entender
7 cómo se relacionan entre sí distintas cosas.
8 '''
9 matriz_correlacion = df.corr()
10
11 # Dibujamos el resultado de la matriz de correlacion
12 fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(5, 5))
13
14 sns.heatmap(
15     matriz_correlacion,
16     annot      = True,
17     cbar       = False,
18     annot_kws  = {"size": 8},
19     vmin       = -1,
20     vmax       = 1,
21     center     = 0,
22     cmap       = sns.diverging_palette(20, 220, n=200),
23     square     = True,
24     ax         = ax
25 )
26
27 ax.set_xticklabels(
28     ax.get_xticklabels(),
29     rotation = 45,
30     horizontalalignment = 'right',
31 )
32
33 ax.tick_params(labelsize = 10)
```

```
<ipython-input-9-6dbfe88c81b1>:9: FutureWarning: The default value of numeric_only
matriz_correlacion = df.corr()
```



La matriz de correlación muestra los valores de correlación, que miden el grado de relación lineal entre cada par de variables. Los valores de correlación se pueden ubicar entre -1 y +1. Si las dos variables tienden a aumentar o disminuir al mismo tiempo, el valor de correlación es positivo.

Por ejemplo:

1. (Eje X) sepal\_length vs (Eje Y) petal\_width = 0.82, esto significa que ambas variables tienden a aumentar o disminuir al mismo tiempo.



#### EJERCICIO 4

En base a las dos graficas anteriores que observa de acuerdo a la distribucion de los datos y a la matriz de correlacion? (De 5 observaciones principales de los datos justificando cada una de ellas con la importancia de porque las eligio)

#### Respuesta:

##### OBSERVACIONES:

1. Las especies de versicolor y virginica tienen comportamientos de distribución entre los features de manera muy similar a diferencia de setosa la cual se comporta de una manera diferente. Esto es visiblemente evidente viendo los cuadros en el grafico 1.
  - Considero que esta observacion es muy importante porque nos evita generalizar conclusiones que apliquen indiferente de las especies cuando vemos que la especie de setosa no se comporta de la misma manera que las otras.
2. Es interesante ver como el largo del petalo influye positivamente en el largo del sepalo de la especie para versicolor y virginica mas no para setosa donde el tamaño de su sepalo puede variar pero no parece influir en el tamaño del petalo que es muy consistente.
  - Identificar cómo el largo del pétalo influye en el largo del sépalo de manera diferente en setosa en comparación con versicolor y virginica es relevante para comprender las relaciones entre las características y cómo varían según la especie.
3. Según el grafico 1, podemos afirmar que cuando analizamos sepalos podemos encontrar una mayor varianza con respecto a la media para las 3 especies. Sin embargo, cuando se trata de petalos vemos que la especie de setosa tiene una distribución con una varianza muy cercana a la media y esto nos da una idea de que el tamaño de petalos en setosa es comun que tengan un tamaño consistente.
  - Observar la varianza en el tamaño de sépalos y pétalos entre las especies proporciona información sobre la consistencia en el tamaño de los pétalos en setosa, lo cual es valioso para entender las características únicas de esta especie.
4. Podemos afirmar que el largo del petalo es positivamente correlacionado con el ancho del petalo en las especies de versicolor y virginica, esto se puede evidenciar en el mosaico de graficos con los cuadros (3,4) y (4,3) donde se evidencia una pendiente positiva entre los puntos y tambien con el grafico dos que nos informa de una correlacion cercana al 1(0.96)
  - Identificar la correlación positiva entre el largo y ancho del pétalo en versicolor y virginica proporciona una comprensión cuantitativa de la relación entre estas características, lo cual es útil para futuros análisis y aplicaciones.
5. Sin conocer a la planta setosa y basandonos el en cuadro (3,4) y (4,3) podemos imaginar que tiene petalos con forma cuadrangular donde el ancho y el largo son muy similares.

- Aunque no se dispone de información directa sobre la forma de los pétalos de setosa, realizar inferencias basadas en los gráficos y la correlación de características proporciona una perspectiva interesante sobre la posible forma de los pétalos en esta especie.

## ▼ Modelado

```

1 # Separando variables para el training y para el testing
2 # -----
3 # Ahora, separaremos la variable de destino (y) y las características (X) de la siguiente manera
4 target = df['species']
5 df1 = df.copy()
6 df1 = df1.drop('species', axis =1)
7

```

Es una buena práctica no eliminar ni agregar una nueva columna al conjunto de datos original. Haga una copia y luego modifíquela para que, en caso de que las cosas no salgan como esperábamos, tengamos los datos originales para comenzar de nuevo con un enfoque diferente. Solo por seguir la convención más utilizada, estamos almacenando df en X

```



1 # Definiendo los atributos
2 X = df1

```

```

1 # Revisamos los datos de X
2 X

```

	sepal_length	sepal_width	petal_length	petal_width	
0	5.1	3.5	1.4	0.2	
1	4.9	3.0	1.4	0.2	
2	4.7	3.2	1.3	0.2	
3	4.6	3.1	1.5	0.2	
4	5.0	3.6	1.4	0.2	
...	...	...	...	...	
145	6.7	3.0	5.2	2.3	
146	6.3	2.5	5.0	1.9	
147	6.5	3.0	5.2	2.0	
148	6.2	3.4	5.4	2.3	
149	5.9	3.0	5.1	1.8	

150 rows × 4 columns

```

1 # Revisamos los datos de target
2 target

```

```

0      setosa
1      setosa
2      setosa
3      setosa
4      setosa
...
145  virginica
146  virginica
147  virginica
148  virginica

```

```
149 virginica
Name: species, Length: 150, dtype: object
```

La columna 'target' tiene variables categoricas (setosa, virginica y versicolor), entonces lo codificaremos en valores numéricos para trabajar.

```
1 # Label encoding (Codificación de etiquetas)
2 # -----
3 le = LabelEncoder()
4 target = le.fit_transform(target)
5 target

array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

Obtenemos su codificación como arriba, setosa:0, versicolor:1, virginica:2

Nuevamente, para seguir la convención de nomenclatura estándar, nombrar destino como 'y'

```
1 y = target
```

Dividir el conjunto de datos en conjuntos de entrenamiento y prueba. seleccionar 20% de registros al azar para realizar pruebas

```
1 #EJERCICIO 5 - Splitting
2 # Complete el siguiente código para dividir los datos > 80 training :20 testing
3 # -----
4
5 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
6
7 print("Training split input- ", X_train.shape)
8 print()
9 print("Testing split input- ", X_test.shape)

Training split input- (120, 4)

Testing split input- (30, 4)
```

## EJERCICIO 6

Después de dividir el conjunto de datos, cuantos registros (filas) para entrenamiento tenemos y cuantos registros para fines de prueba?? **Respuesta:**

--- Reemplace aquí su respuesta---

```
1 # EJERCICIO 7
2 # Definimos el algoritmo de Decision Tree (Arbol de decision)
3 # -----
4 #Definición del decision tree
5 dtree = DecisionTreeClassifier(random_state=42)
6 #Entrenamiento
7 dtree.fit(X_train, y_train)
8
9 print('Decision Tree Classifier Created')

Decision Tree Classifier Created
```

En el código anterior, creamos un objeto de la clase `DecisionTreeClassifier`, almacenamos su dirección en la variable `dtree`, para que podamos acceder al objeto usando `dtree`. Luego ajustamos este árbol con nuestro `X_train` y `y_train`.

Finalmente, imprimimos la declaración Clasificador de árbol de decisión creado después de construir el árbol de decisión.

```
1 # Prediciendo los valores de los datos de testeo
2 # -----
3 y_pred = dtree.predict(X_test)
4 print("Reporte de Clasificacion - \n", classification_report(y_test,y_pred))
```

```
Reporte de Clasificacion -
              precision    recall  f1-score   support

     0       1.00      1.00      1.00        10
     1       1.00      1.00      1.00         9
     2       1.00      1.00      1.00        11

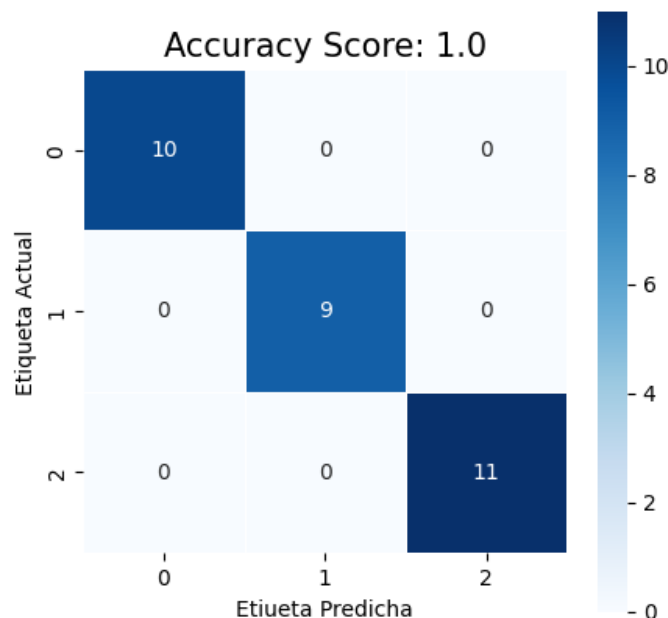
 accuracy          1.00
 macro avg          1.00
 weighted avg       1.00
```

Obtuvimos una precisión del 100% en el conjunto de datos de prueba de 30 registros.

tracemos la matriz de confusión de la siguiente manera

```
1 #EJERCICIO 8
2 # Obtenemos la matriz de confusion
3 # -----
4 cm = confusion_matrix(y_test, y_pred)
5 cm
6
7 plt.figure(figsize=(5,5))
8 sns.heatmap(data=cm,linewidths=.5, annot=True,square = True, cmap = 'Blues')
9 plt.ylabel('Etiqueta Actual')
10 plt.xlabel('Etiqueta Predicha')
11 all_sample_title = 'Accuracy Score: {}'.format(dtree.score(X_test, y_test))
12 plt.title(all_sample_title, size = 15)
```

```
Text(0.5, 1.0, 'Accuracy Score: 1.0')
```





**EJERCICIO 9**

Realice la interpretacion de TODOS los resultados de la matriz de confusion, se espera 9 interpretacion de resultados.

**Respuesta:**

- (1,1) El modelo predijo 10 especies con la etiqueta de Setosa y la etiqueta actual era Setosa
  - (1,2) El modelo predijo 0 especies con la etiqueta de Setosa y la etiqueta actual era Versicolor
  - (1,3) El modelo predijo 0 especies con la etiqueta de Setosa y la etiqueta actual era Virginica
- 
- (2,1) El modelo predijo 0 especies con la etiqueta de Versicolor y la etiqueta actual era Setosa
  - (2,2) El modelo predijo 9 especies con la etiqueta de Versicolor y la etiqueta actual era Versicolor
  - (2,3) El modelo predijo 0 especies con la etiqueta de Versicolor y la etiqueta actual era Virginica
- 
- (3,1) El modelo predijo 0 especies con la etiqueta de Virginica y la etiqueta actual era Setosa
  - (3,2) El modelo predijo 0 especies con la etiqueta de Virginica y la etiqueta actual era Versicolor
  - (3,3) El modelo predijo 11 especies con la etiqueta de Virginica y la etiqueta actual era Virginica

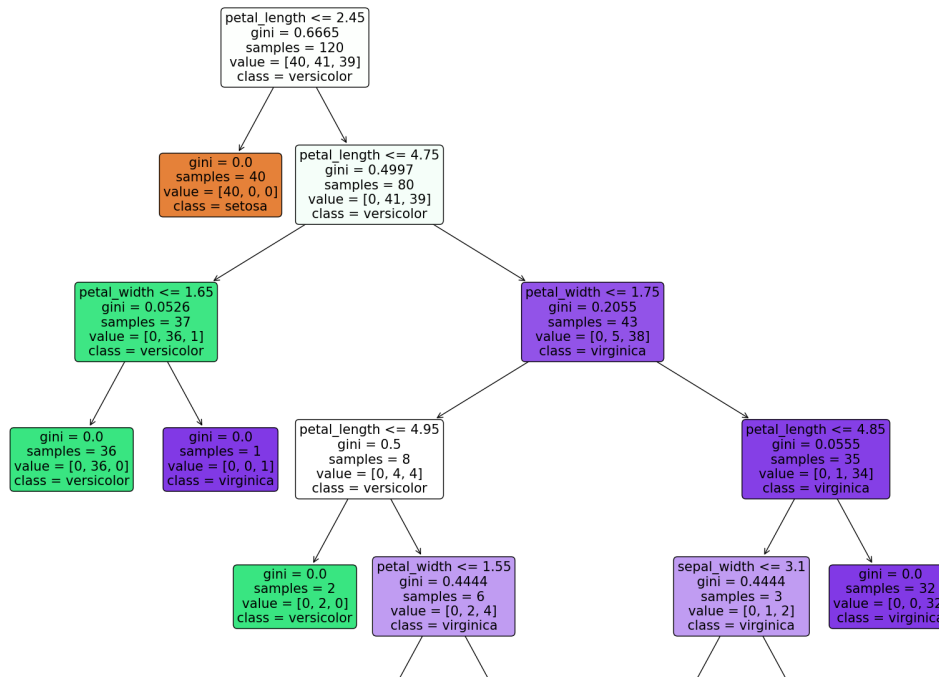
## ▼ Visualizamos el Decision Tree (Arbol de Decision)

Podemos trazar directamente el árbol que construimos usando los siguientes comandos

```

1 plt.figure(figsize = (20,20))
2 feature_names = df1.columns.tolist()
3 cn = ['setosa', 'versicolor', 'virginica']
4
5 dec_tree = plot_tree(
6     decision_tree=dtree,
7     feature_names = feature_names,
8     class_names = cn,
9     filled = True ,
10    precision = 4,
11    rounded = True,
12    fontsize = 15)

```



Podemos ver cómo se divide el árbol, cuáles son los gini de los nodos, los registros en esos nodos y sus etiquetas.

### EJERCICIO 10

Elija 3 caminos del árbol de decision y explique la evaluacio que el árbol ha realizado.

**Respuesta:**

1. Si el largo del petalo es  $\leq 2.45$

- False: es Setosa
- True: Especie Versicolor o Virginica
- ++++++

2. Si el ancho del petalo es  $\leq 1.65$

- False: es Versicolor
- True: Virginica
- ++++++

3. Si el largo del petalo es  $\leq 5.45$

- True: Es Versicolor
- False: Es Virginica

