Guillermo García Hernández
2ºDAW 25/26
IA

ML Clasificación

```
Resultados de precisión en el conjunto de test:
Con test size = 0.2
SVM: 1.0000
Naive Bayes: 0.9667
LDA: 1.0000
QDA: 1.0000
Decision Tree: 1.0000
Random Forest: 0.9667
K-Nearest: 0.9667
Neural Network: 1.0000
```

```
Resultados de precisión en el conjunto de test:
Con test size = 0.4
SVM: 0.9333
Naive Bayes: 0.9333
LDA: 0.9500
QDA: 0.9667
Decision Tree: 0.9500
Random Forest: 0.9333
K-Nearest: 0.9500
Neural Network: 0.9833
```

La precisión varía al cambiar el test_size. Con 0.2 la mayoría de modelos obtienen una precisión cercana a . Y con 0.4 algunas bajan.

```
Resultados de precisión en el conjunto de test:
Con n_neighbors=3
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9600
K-Nearest: 0.9333
Neural Network: 0.9733
```

```
Resultados de precisión en el conjunto de test:
Con n_neighbors=5
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9600
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con n_neighbors=10
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9467
K-Nearest: 0.9467
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con Decision Tree (max_depth=2)
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.8933
Random Forest: 0.9333
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con Decision Tree (max_depth=4)
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9467
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con Decision Tree (max_depth=None):
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9467
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con Random Forest (n_estimators=50)
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9200
K-Nearest: 0.9600
Neural Network: 0.9733
```

```
Resultados de precisión en el conjunto de test:
Con Random Forest (n_estimators=100)
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9467
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con Random Forest (n_estimators=200)
SVM: 0.9467
Naive Bayes: 0.9467
LDA: 0.9600
QDA: 0.9600
Decision Tree: 0.9600
Random Forest: 0.9467
K-Nearest: 0.9600
Neural Network: 0.9867
```

```
Resultados de precisión en el conjunto de test:
Con load_breast_cancer
SVM: 0.9158
Naive Bayes: 0.9368
LDA: 0.9579
QDA: 0.9544
Decision Tree: 0.9263
Random Forest: 0.9439
K-Nearest: 0.9368
c:\Users\U_38010700\AppData\Local\Programs\Python\Python313\Lib
  warnings.warn(
c:\Users\U_38010700\AppData\Local\Programs\Python\Python313\Lib
  warnings.warn(
Neural Network: 0.9404
```

4. Completa esta tabla en tu cuaderno:

Modelo	Idea básica	Parámetro clave	Ventaja	Limitación
Naive Bayes	Usa probabilidades para decidir a qué clase pertenece algo.	No tiene parámetros importantes.	Es muy rápido y fácil de usar.	Supone que las variables son independientes
Árbol	Separa los datos con reglas tipo "si... entonces..."	max_depth (qué tan profundo puede ser el árbol).	Se entiende fácilmente y se puede dibujar.	Si el árbol es muy grande, se puede pasar de memoria
Bosque Aleatorio	Combina muchos árboles y hace una votación entre ellos.	n_estimators (cuántos árboles usa).	Da muy buena precisión y evita errores de un solo árbol.	Es más lento y no se puede interpretar tan fácil.
KNN	Clasifica mirando los vecinos más cercanos.	n_neighbors (cuántos vecinos tiene en cuenta).	Es simple y no necesita entrenamiento complicado.	Es lento si hay muchos datos y depende mucho de la escala.
Red Neuronal	Imita el cerebro conectando "neuronas" que aprenden patrones	max_iter, alpha (controlan el entrenamiento).	Aprende relaciones muy complejas.	Necesita bastantes datos y tarda más en entrenar.

5. Reflexiona

¿Por qué algunos modelos funcionan mejor que otros?

Porque cada modelo aprende de forma diferente.
Algunos se adaptan mejor cuando los datos son simples o lineales (como LDA o SVM), y otros funcionan mejor cuando hay más variables o relaciones más complejas (como Random Forest o Redes Neuronales).

¿Cuál sería más adecuado si necesitas rapidez?

Modelos como Naive Bayes, LDA o KNN (con pocos vecinos) son los más rápidos, porque casi no necesitan entrenamiento y calculan resultados enseguida.

¿Y si necesitas explicabilidad?

El Árbol de Decisión es el más fácil de entender,
porque muestra reglas claras del tipo “si… entonces…”.
También Naive Bayes y LDA son bastante interpretables.