Practicando con el modelo: Árboles de decisión

Sistemas de aprendizaje automático

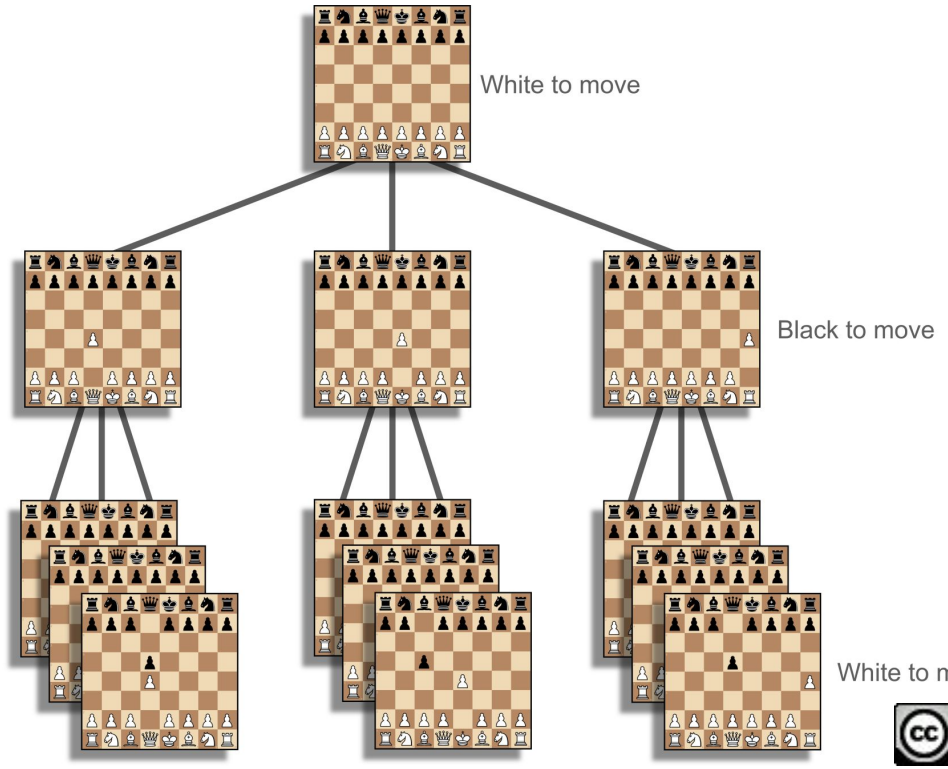
Profesor:
David Campoy Miñarro

¿Qué es un árbol de decisión?

Son algoritmos para hacer **clasificaciones** realizando particiones sucesivas, mediante la técnica **segmentación jerárquica**, agrupando observaciones similares. Analizan situaciones que presentan varias posibilidades de decisión y toman la que consideran mejor. También se emplean para realizar tareas de **regresión**.

- **Nodos:** son las variables de entrada y plantean las opciones de decisión.
- **Ramas:** indican los valores que pueden tomar las variables de entrada y unen los nodos entre sí.
- **Hojas:** muestran los valores de las variables de salida, es decir, el resultado de la decisión.

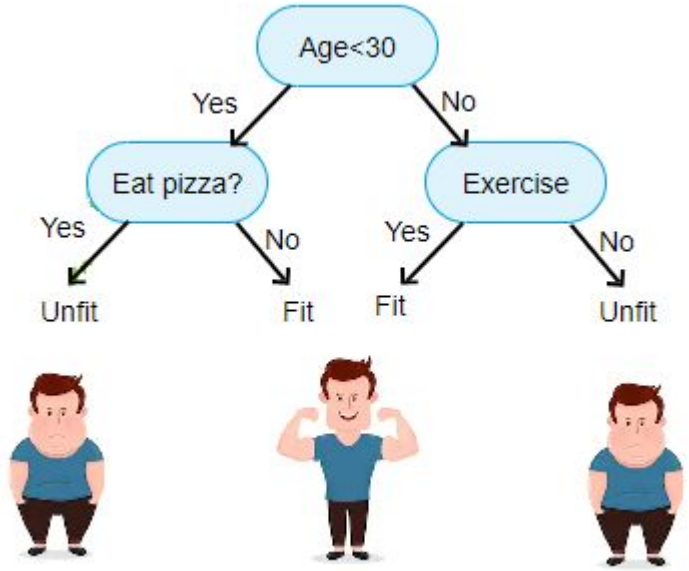
La profundidad de un árbol es el número de máximo de nodos en una rama.



Árboles de clasificación y regresión CART

El algoritmo de árboles de clasificación y regresión (CART) genera árboles de decisión binarios, por lo que cada nodo se divide en dos ramas.

- **Clasificación y predicción:**
  - predecir si un correo electrónico es spam.
  - predecir el precio de una casa
- **Interpretación:** son fáciles de entender.
- **Tratamiento de diferentes tipos de datos**
- **Escalabilidad y velocidad**
- **Identificación de características importantes:** pueden ayudar a identificar qué características son más importantes para la predicción.

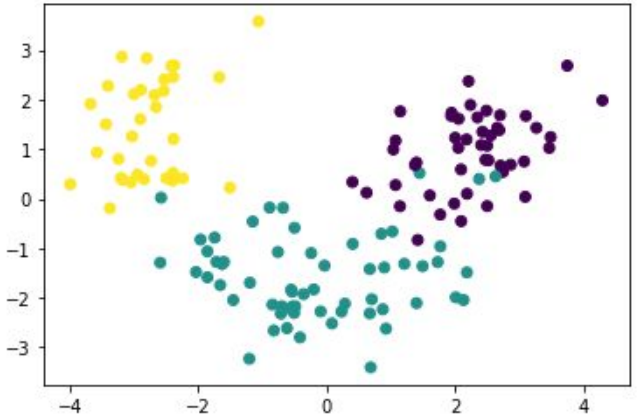


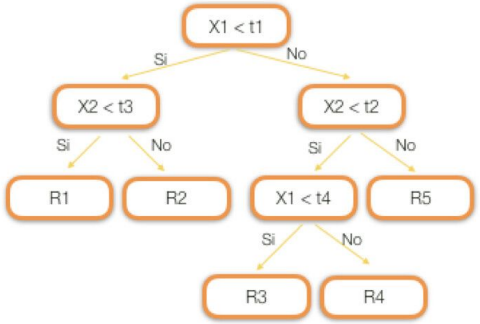
LDA versus árboles de decisión CART

Datos no lineales: Los árboles de decisión, como CART, son más adecuados cuando se enfrentan a relaciones no lineales.

Supuestos lineales cumplidos: Si los supuestos de LDA se cumplen (clases bien separadas, distribuciones normales), puede proporcionar clasificaciones óptimas y una buena reducción de dimensionalidad.

Interpretación vs. precisión: Si se prioriza la interpretación y visualización del modelo, los árboles de decisión son más adecuados. Si se busca un modelo más ajustado a supuestos específicos y se valora la precisión en casos de clases bien separadas, LDA podría ser más apropiado.

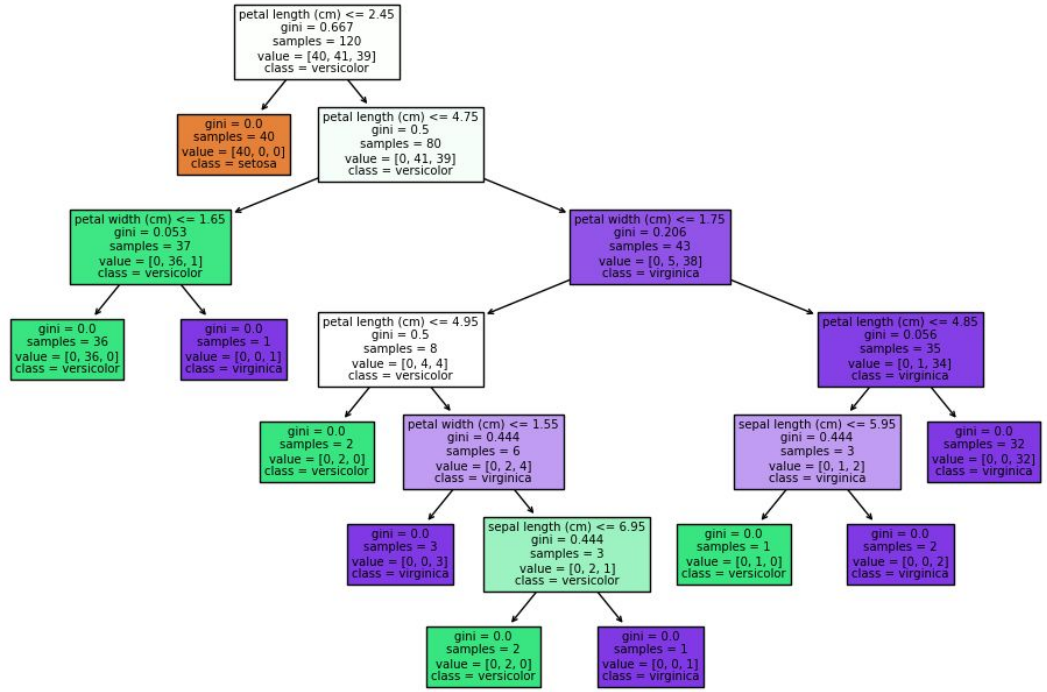




Actividad 1.

Este programa utiliza el modelo árboles CART para clasificar las flores del dataset Iris.

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Precisión del modelo de Árbol de Clasificación:", accuracy)
plt.figure(figsize=(12, 8))
plot_tree(clf, filled=True, feature_names=iris.feature_names,
class_names=iris.target_names)
plt.show()
```



Actividad 2. CART en finanzas

El objetivo de este programa es aplicar el algoritmo de clasificación CART a un conjunto de datos de crédito alemán para predecir si un cliente será aprobado o no para un crédito.

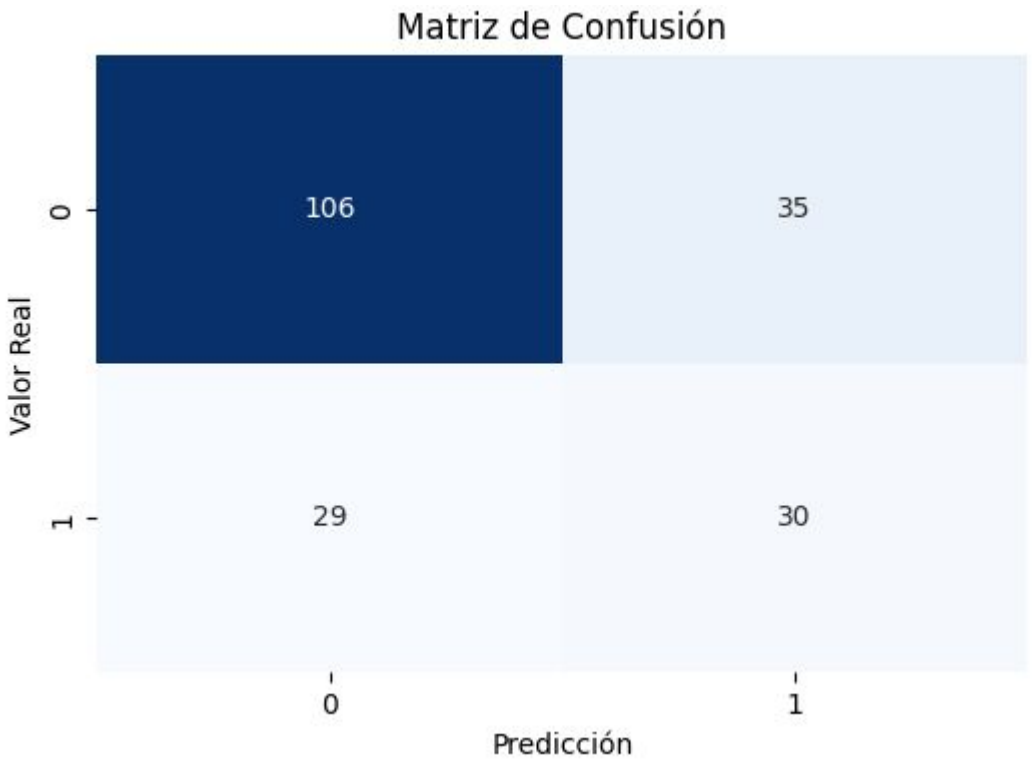
Datos de crédito alemán

`https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data`

```
A11 6 A34 A43 1169 A65 A75 4 A93 A101 4 A121 67 A143 A152 2 A173 1 A192 A201 1
A12 48 A32 A43 5951 A61 A73 2 A92 A101 2 A121 22 A143 A152 1 A173 1 A191 A201 2
A14 12 A34 A46 2096 A61 A74 2 A93 A101 3 A121 49 A143 A152 1 A172 2 A191 A201 1
A11 42 A32 A42 7882 A61 A74 2 A93 A103 4 A122 45 A143 A153 1 A173 2 A191 A201 1
A11 24 A33 A40 4870 A61 A73 3 A93 A101 4 A124 53 A143 A153 2 A173 2 A191 A201 2
A14 36 A32 A46 9055 A65 A73 2 A93 A101 4 A124 35 A143 A153 1 A172 2 A192 A201 1
A14 24 A32 A42 2835 A63 A75 3 A93 A101 4 A122 53 A143 A152 1 A173 1 A191 A201 1
A12 36 A32 A41 6948 A61 A73 2 A93 A101 2 A123 35 A143 A151 1 A174 1 A192 A201 1
A14 12 A32 A43 3059 A64 A74 2 A91 A101 4 A121 61 A143 A152 1 A172 1 A191 A201 1
A12 30 A34 A40 5234 A61 A71 4 A94 A101 2 A123 28 A143 A152 2 A174 1 A191 A201 2
A12 12 A32 A40 1295 A61 A72 3 A92 A101 1 A123 25 A143 A151 1 A173 1 A191 A201 2
A11 48 A32 A49 4308 A61 A72 3 A92 A101 4 A122 24 A143 A151 1 A173 1 A191 A201 2
A12 12 A32 A43 1567 A61 A73 1 A92 A101 1 A123 22 A143 A152 1 A173 1 A192 A201 1
A11 24 A34 A40 1199 A61 A75 4 A93 A101 4 A123 60 A143 A152 2 A172 1 A191 A201 2
```

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Descargar el conjunto de datos de crédito alemán desde una URL
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data'
column_names = [
    "Status_of_existing_checking_account", "Duration_in_month", "Credit_history", "Purpose",
    "Credit_amount", "Savings_account", "Present_employment_since", "Installment_rate",
    "Personal_status_and_sex", "Other_debtors", "Present_residence_since", "Property",
    "Age", "Other_installment_plans", "Housing", "Number_of_existing_credits", "Job",
    "Number_of_people_being_liable", "Telephone", "Foreign_worker", "Credit_approval"
]

# Cargar el conjunto de datos en un DataFrame de Pandas
data = pd.read_csv(url, delimiter=' ', header=None, names=column_names)

# Convertir variables categóricas usando One-Hot Encoding
categorical_cols = [
    "Status_of_existing_checking_account", "Credit_history", "Purpose", "Savings_account",
    "Present_employment_since", "Personal_status_and_sex", "Other_debtors", "Property",
    "Other_installment_plans", "Housing", "Job", "Telephone", "Foreign_worker"
]
numeric_cols = [col for col in data.columns if col not in categorical_cols and col != "Credit_approval"]

# Aplicar One-Hot Encoding a las variables categóricas
data = pd.get_dummies(data, columns=categorical_cols, drop_first=True)

# Separar características (X) y etiquetas (y)
X = data.drop('Credit_approval', axis=1)
y = data['Credit_approval']

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear un clasificador de Árbol de Decisión CART
clf = DecisionTreeClassifier()
```

```
# Entrenar el clasificador con los datos de entrenamiento
clf.fit(X_train, y_train)

# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(X_test)

# Calcular la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo de Árbol de Clasificación:
{accuracy:.2f}")

# Matriz de confusión para evaluar el rendimiento del modelo
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
cbar=False)
plt.xlabel('Predicción')
plt.ylabel('Valor Real')
plt.title('Matriz de Confusión')
plt.show()
```



¿Qué te parecen los resultados? Vamos a intentar mejorarlos con el algoritmo de **bosque aleatorio**.

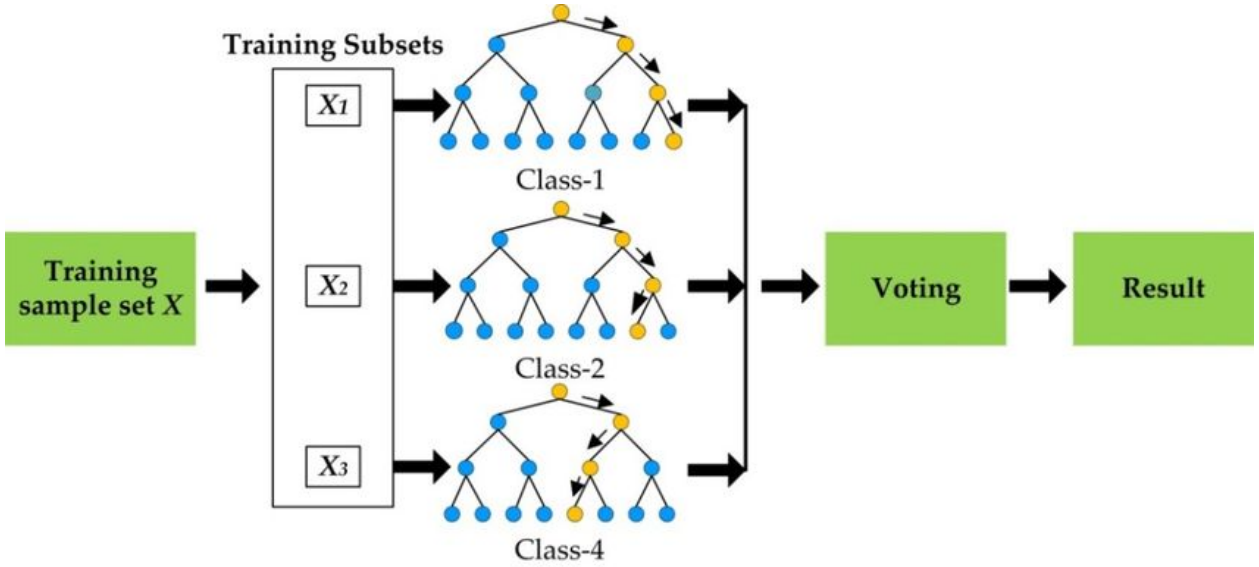
Árbol de decisión: bosque aleatorio, Random Forest (RF)

Mayor precisión: predicción de múltiples árboles, el *Random Forest* tiende a tener una mayor precisión en comparación con un solo árbol de decisión.

Reducción del sobreajuste: Random Forest reduce el sobreajuste que puede ocurrir en un solo árbol, mejorando su capacidad para generalizar a nuevos datos.

Datos atípicos: Al tomar decisiones basadas en múltiples árboles, el modelo tiende a ser más resistente a valores atípicos o ruido en los datos.

Grandes conjuntos de datos: puede manejar conjuntos de datos grandes y con muchas características, lo que lo hace adecuado para una variedad de problemas.



El *Random Forest* funciona combinando muchos árboles de decisión individuales.

Cada árbol se entrena con una porción aleatoria de los datos y realiza su propia predicción.

Luego, el *Random Forest* combina estas predicciones para llegar a una respuesta final.

Actividad 3. Random Forest en finanzas

El objetivo de este programa es aplicar el algoritmo de clasificación *Random Forest* a un conjunto de datos de crédito alemán para predecir si un cliente será aprobado o no para un crédito. **Compara los resultados obtenidos con el modelo CART.**

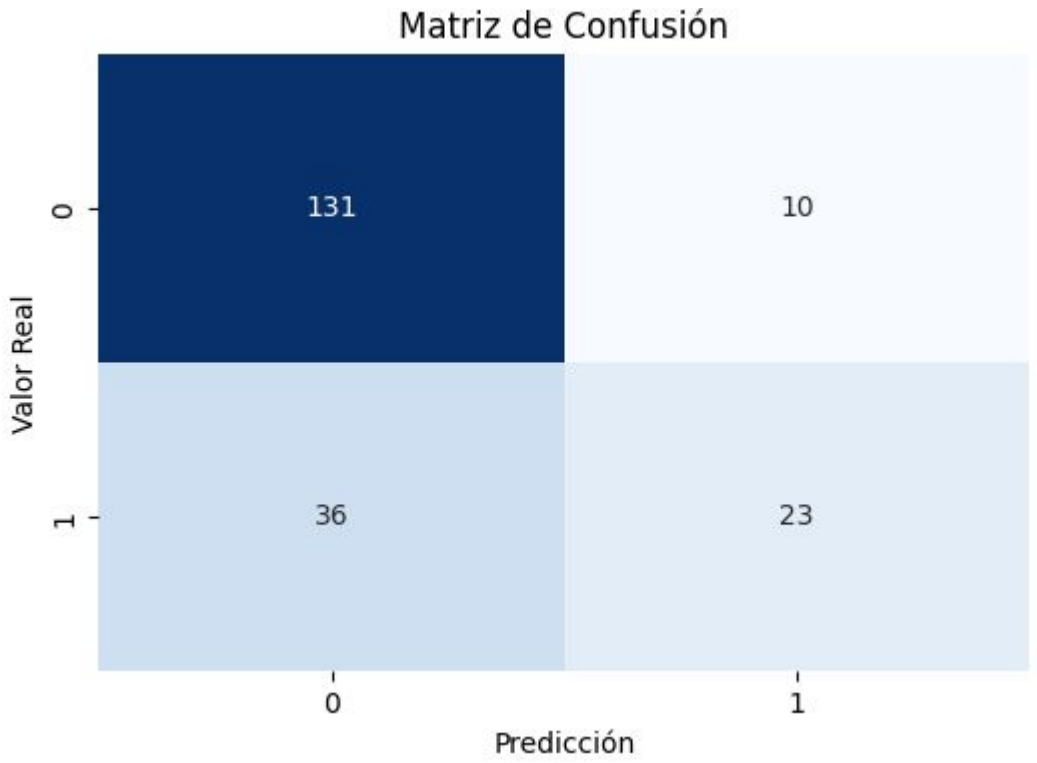
Datos de crédito alemán

`https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data`

```
A11 6 A34 A43 1169 A65 A75 4 A93 A101 4 A121 67 A143 A152 2 A173 1 A192 A201 1
A12 48 A32 A43 5951 A61 A73 2 A92 A101 2 A121 22 A143 A152 1 A173 1 A191 A201 2
A14 12 A34 A46 2096 A61 A74 2 A93 A101 3 A121 49 A143 A152 1 A172 2 A191 A201 1
A11 42 A32 A42 7882 A61 A74 2 A93 A103 4 A122 45 A143 A153 1 A173 2 A191 A201 1
A11 24 A33 A40 4870 A61 A73 3 A93 A101 4 A124 53 A143 A153 2 A173 2 A191 A201 2
A14 36 A32 A46 9055 A65 A73 2 A93 A101 4 A124 35 A143 A153 1 A172 2 A192 A201 1
A14 24 A32 A42 2835 A63 A75 3 A93 A101 4 A122 53 A143 A152 1 A173 1 A191 A201 1
A12 36 A32 A41 6948 A61 A73 2 A93 A101 2 A123 35 A143 A151 1 A174 1 A192 A201 1
A14 12 A32 A43 3059 A64 A74 2 A91 A101 4 A121 61 A143 A152 1 A172 1 A191 A201 1
A12 30 A34 A40 5234 A61 A71 4 A94 A101 2 A123 28 A143 A152 2 A174 1 A191 A201 2
A12 12 A32 A40 1295 A61 A72 3 A92 A101 1 A123 25 A143 A151 1 A173 1 A191 A201 2
A11 48 A32 A49 4308 A61 A72 3 A92 A101 4 A122 24 A143 A151 1 A173 1 A191 A201 2
A12 12 A32 A43 1567 A61 A73 1 A92 A101 1 A123 22 A143 A152 1 A173 1 A192 A201 1
A11 24 A34 A40 1199 A61 A75 4 A93 A101 4 A123 60 A143 A152 2 A172 1 A191 A201 2
```

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Descargar el conjunto de datos de crédito alemán desde una URL
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data'
column_names = [
    "Status_of_existing_checking_account", "Duration_in_month", "Credit_history", "Purpose",
    "Credit_amount", "Savings_account", "Present_employment_since", "Installment_rate",
    "Personal_status_and_sex", "Other_debtors", "Present_residence_since", "Property",
    "Age", "Other_installment_plans", "Housing", "Number_of_existing_credits", "Job",
    "Number_of_people_being_liable", "Telephone", "Foreign_worker", "Credit_approval"
]
# Cargar el conjunto de datos en un DataFrame de Pandas
data = pd.read_csv(url, delimiter=' ', header=None, names=column_names)

# Convertir variables categóricas usando One-Hot Encoding
categorical_cols = [
    "Status_of_existing_checking_account", "Credit_history", "Purpose", "Savings_account",
    "Present_employment_since", "Personal_status_and_sex", "Other_debtors", "Property",
    "Other_installment_plans", "Housing", "Job", "Telephone", "Foreign_worker"
]
numeric_cols = [col for col in data.columns if col not in categorical_cols and col != "Credit_approval"]

# Aplicar One-Hot Encoding a las variables categóricas
data = pd.get_dummies(data, columns=categorical_cols, drop_first=True)

# Separar características (X) y etiquetas (y)
X = data.drop('Credit_approval', axis=1)
y = data['Credit_approval']

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear un clasificador de Bosques Aleatorios (Random Forest)
clf = RandomForestClassifier()

# Entrenar el clasificador con los datos de entrenamiento
clf.fit(X_train, y_train)
```

```
# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(X_test)

# Calcular la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo de Bosques Aleatorios:
{accuracy:.2f}")

# Matriz de confusión para evaluar el rendimiento del modelo
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
cbar=False)
plt.xlabel('Predicción')
plt.ylabel('Valor Real')
plt.title('Matriz de Confusión')
plt.show()
```



¿Qué te parecen los resultados?

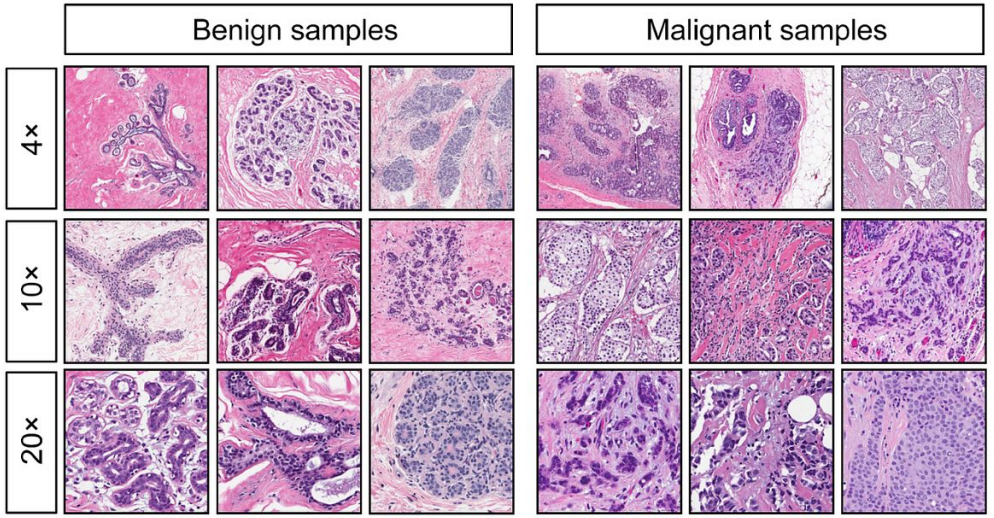
Compara los resultados entre los algoritmos: CART y Random Forest.

Actividad 4. Random Forest en salud

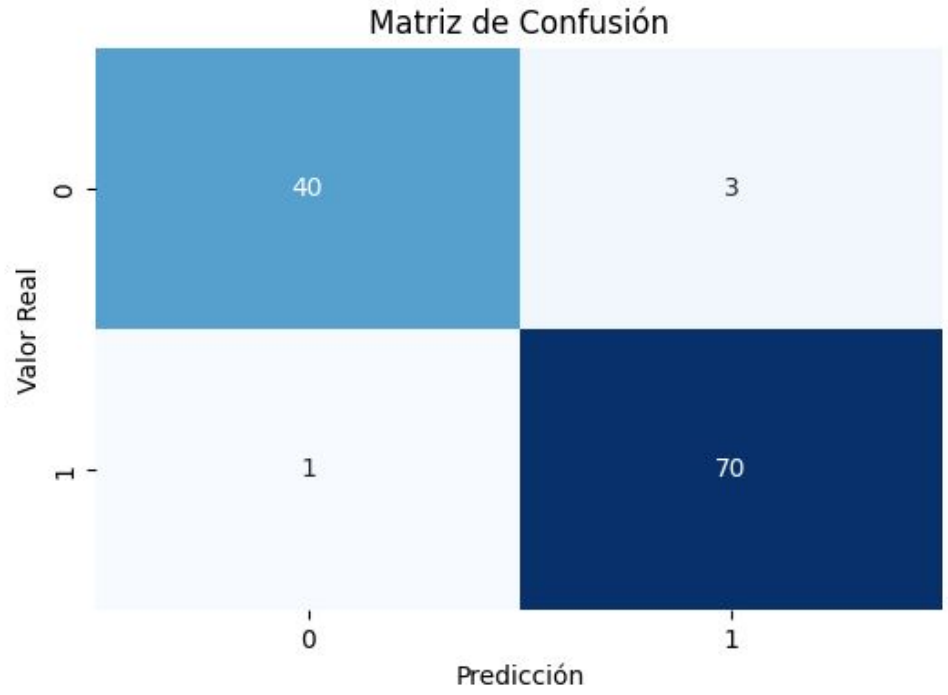
Averiguar si un paciente tiene cáncer basándonos en características médicas. Utilicemos el conjunto de datos "Breast Cancer Wisconsin (Diagnostic)" que contiene características extraídas de imágenes de tumores de mama.

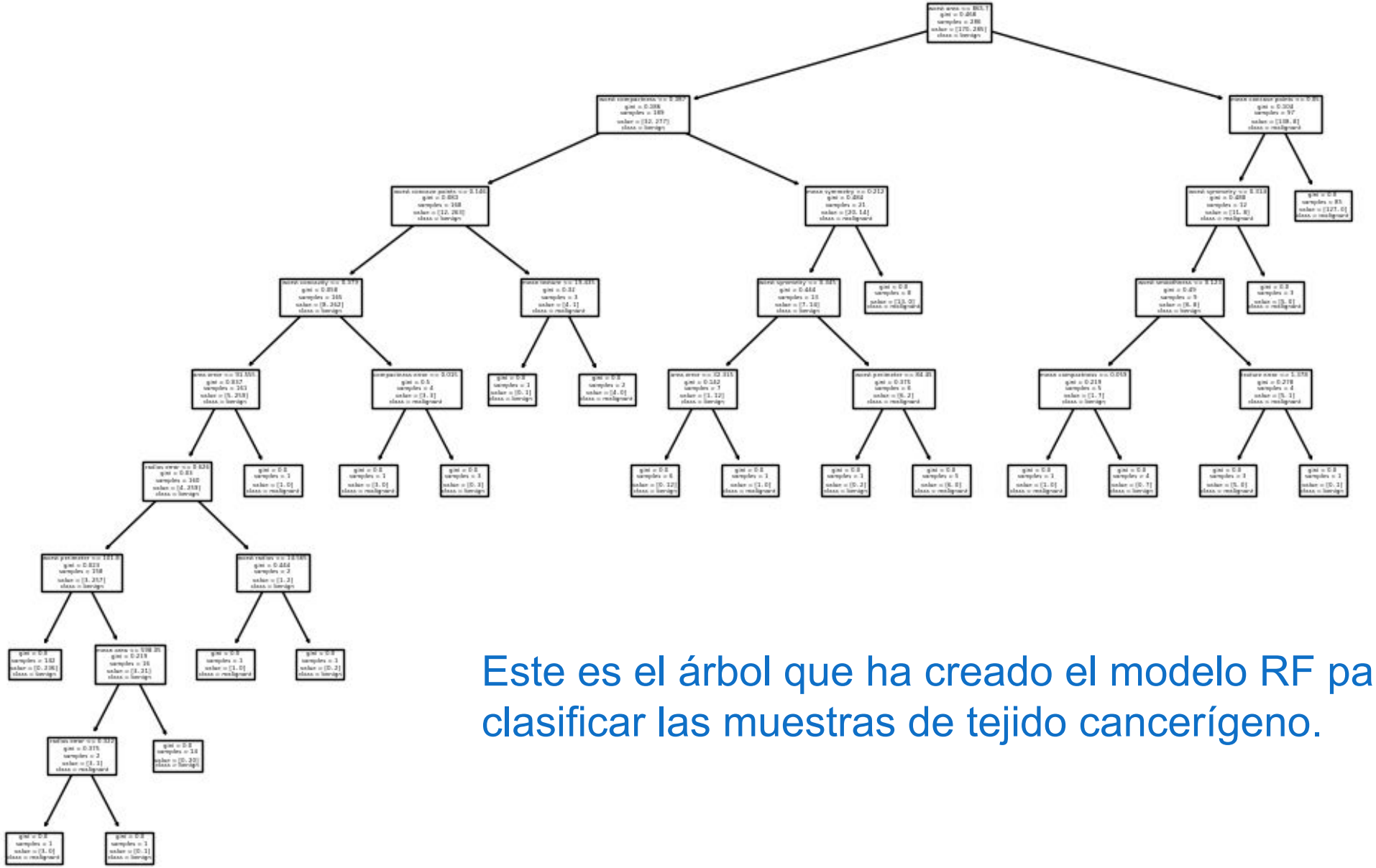
```
https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic
```

El objetivo es predecir si un tumor es benigno o maligno basándonos en características extraídas de imágenes utilizando *Random Forest*.



```python
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Cargar el conjunto de datos Breast Cancer Wisconsin (Diagnostic)
data = load_breast_cancer()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target)

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear un clasificador de Bosques Aleatorios (Random Forest)
clf = RandomForestClassifier()

# Entrenar el clasificador con los datos de entrenamiento
clf.fit(X_train, y_train)

# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(X_test)

# Calcular la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo de Bosques Aleatorios: {accuracy:.2f}")

# Matriz de confusión para evaluar el rendimiento del modelo
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicción')
plt.ylabel('Valor Real')
plt.title('Matriz de Confusión')
plt.show()
```





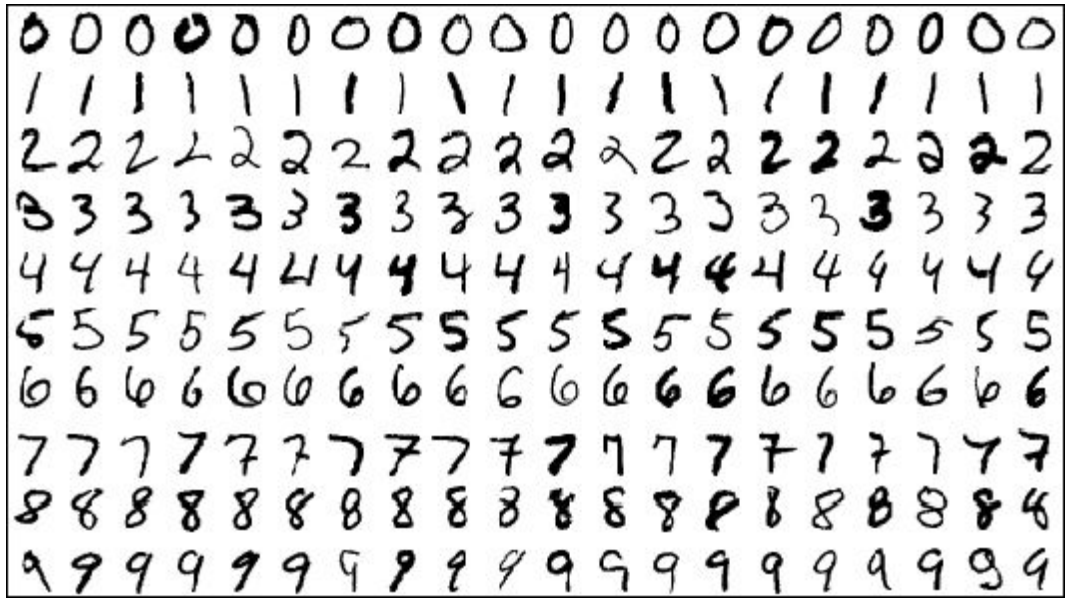
Este es el árbol que ha creado el modelo RF para clasificar las muestras de tejido cancerígeno.

Actividad 5. RF en reconocimiento de patrones (opcional)

Consideremos un conjunto de datos de dígitos escritos a mano. Usaremos el conjunto de datos MNIST, que contiene imágenes de dígitos escritos a mano del 0 al 9.

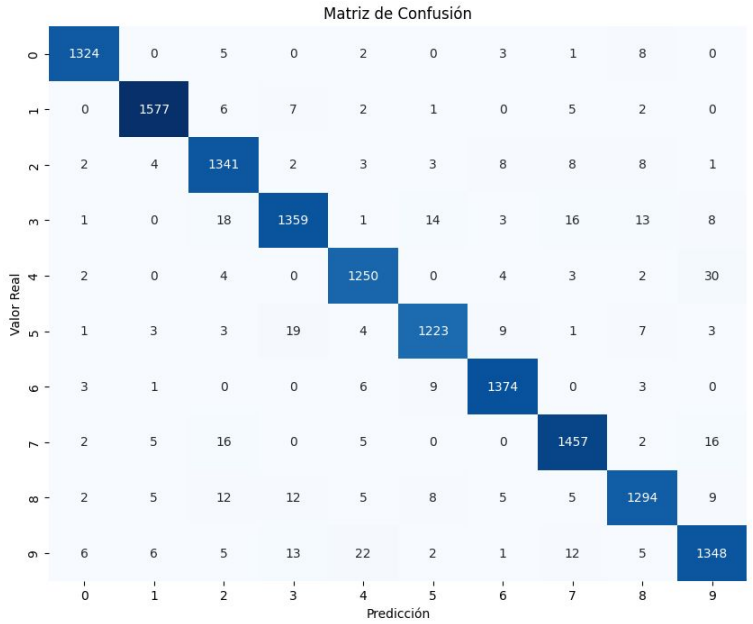
```
https://www.openml.org/search?type=data&sort=runs&id=554&status=active
```

El objetivo es reconocer dígitos escritos a mano correctamente basándonos en las imágenes proporcionadas en el conjunto de datos utilizando el modelo *Random Forest*.

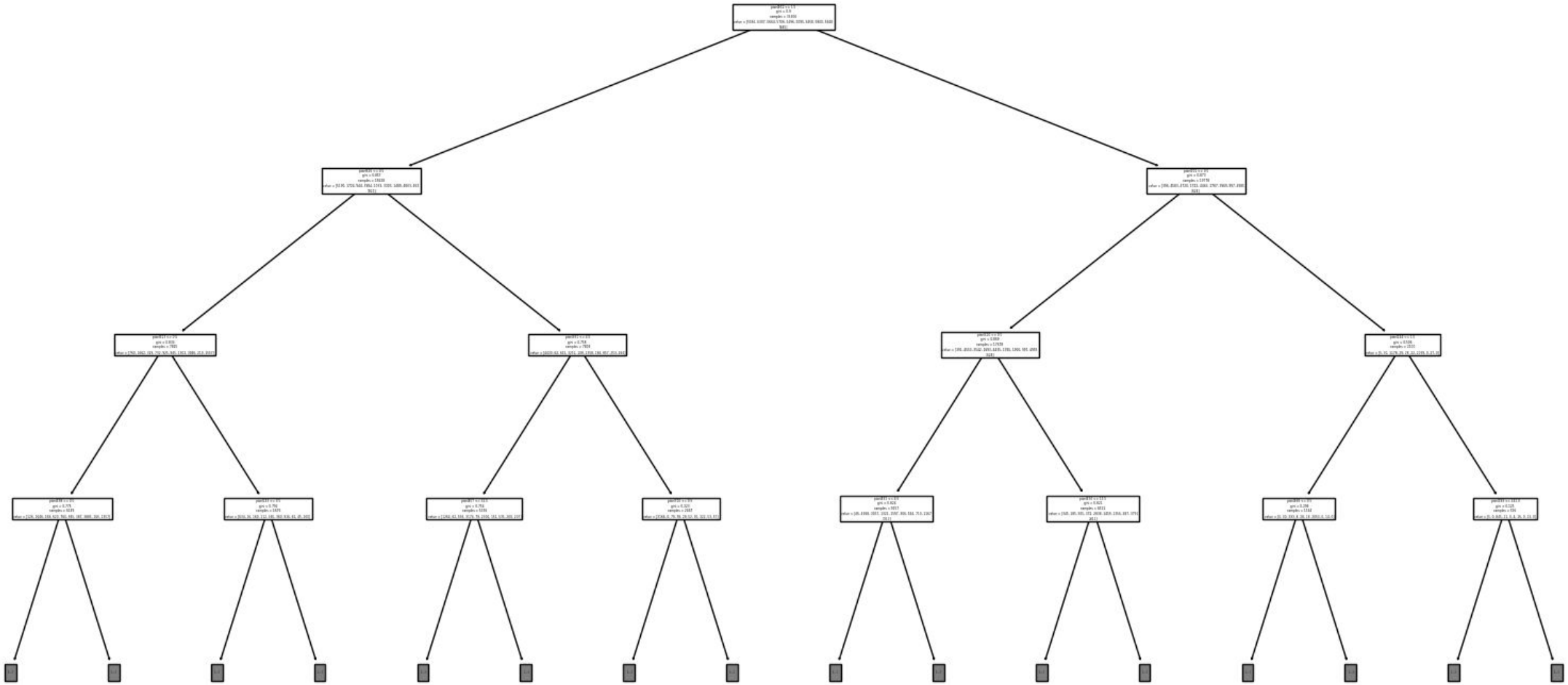


```python
from sklearn.datasets import fetch_openml
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import numpy as np

# Cargar el conjunto de datos MNIST
mnist = fetch_openml('mnist_784', version=1)
X, y = mnist["data"], mnist["target"]

# Convertir etiquetas a números enteros
y = y.astype(np.uint8)

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Crear un clasificador de Bosques Aleatorios (Random Forest)

clf = RandomForestClassifier(n_estimators=100)

# Entrenar el clasificador con los datos de entrenamiento
clf.fit(X_train, y_train)

# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(X_test)
```

```python
# Calcular la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo de Bosques Aleatorios: {accuracy:.2f}")

# Matriz de confusión para evaluar el rendimiento del modelo
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicción')
plt.ylabel('Valor Real')
plt.title('Matriz de Confusión')
plt.show()
```



Prueba con distintos valores de *n_estimadors*



Actividad 6. Detector de mensajes de SPAM (opcional)

Conjunto de datos "*Sentiment140*" de Twitter, el cual contiene mensajes de Twitter clasificados como positivos o negativos.

```
https://raw.githubusercontent.com/crwong/cs224u-project/master/data/sentiment/training.1600000.processed.noemoticon.csv
```

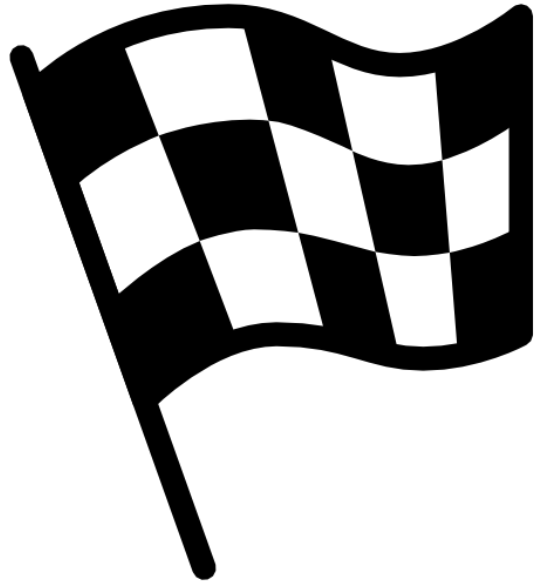
El objetivo es reconocer si un texto escrito en Twitter es SPAM o no lo es. Vamos a utilizar el **algoritmo de bosque aleatorio**.

```
"0","1467810369","Mon Apr 06 22:19:45 PDT 2009","NO_QUERY","_TheSpecialOne_","@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D"
"0","1467810672","Mon Apr 06 22:19:49 PDT 2009","NO_QUERY","scotthamilton","is upset that he can't update his Facebook by texting it... and might cry as a result  School today also. Blah!"
"0","1467810917","Mon Apr 06 22:19:53 PDT 2009","NO_QUERY","mattycus","@Kenichan I dived many times for the ball. Managed to save 50%  The rest go out of bounds"
"0","1467811184","Mon Apr 06 22:19:57 PDT 2009","NO_QUERY","ElleCTF","my whole body feels itchy and like its on fire "
"0","1467811193","Mon Apr 06 22:19:57 PDT 2009","NO_QUERY","Karoli","@nationwideclass no, it's not behaving at all. i'm mad. why am i here? because I can't see you all over there. "
"0","1467811372","Mon Apr 06 22:20:00 PDT 2009","NO_QUERY","joy_wolf","@Kwesidei not the whole crew "
"0","1467811592","Mon Apr 06 22:20:03 PDT 2009","NO_QUERY","mybirch","Need a hug "
"0","1467811594","Mon Apr 06 22:20:03 PDT 2009","NO_QUERY","coZZ","@LOLTrish hey  long time no see! Yes.. Rains a bit ,only a bit  LOL , I'm fine thanks , how's you ?"
"0","1467811795","Mon Apr 06 22:20:05 PDT 2009","NO_QUERY","2Hood4Hollywood","@Tatiana_K nope they didn't have it "
"0","1467812025","Mon Apr 06 22:20:09 PDT 2009","NO_QUERY","mimismo","@twittera que me muera ? "
"0","1467812416","Mon Apr 06 22:20:16 PDT 2009","NO_QUERY","erinx3leannexo","spring break in plain city... it's snowing "
"0","1467812579","Mon Apr 06 22:20:17 PDT 2009","NO_QUERY","pardonlauren","I just re-pierced my ears "
"0","1467812723","Mon Apr 06 22:20:19 PDT 2009","NO_QUERY","TLeC","@caregiving I couldn't bear to watch it.  And I thought the UA loss was embarrassing . . . . ."
"0","1467812771","Mon Apr 06 22:20:19 PDT 2009","NO_QUERY","robrobbierobert","@octolinz16 It it counts, idk why I did either. you never talk to me anymore "
"0","1467812784","Mon Apr 06 22:20:20 PDT 2009","NO_QUERY","bayofwolves","@smarrison i would've been the first, but i didn't have a gun.    not really though, zac snyder's just a doucheclown."
"0","1467812799","Mon Apr 06 22:20:20 PDT 2009","NO_QUERY","HairByJess","@iamjazzyfizzle I wish I got to watch it with you!! I miss you and @iamlilnicki  how was the premiere?!"
"0","1467812964","Mon Apr 06 22:20:22 PDT 2009","NO_QUERY","lovesongwriter","Hollis' death scene will hurt me severely to watch on film  wry is directors cut not out now?"
"0","1467813137","Mon Apr 06 22:20:25 PDT 2009","NO_QUERY","armotley","about to file taxes "
"0","1467813579","Mon Apr 06 22:20:31 PDT 2009","NO_QUERY","starkissed","@LettyA ahh ive always wanted to see rent  love the soundtrack!!"
"0","1467813782","Mon Apr 06 22:20:34 PDT 2009","NO_QUERY","gi_gi_bee","@FakerPattyPattz Oh dear. Were you drinking out of the forgotten table drinks? "
"0","1467813985","Mon Apr 06 22:20:37 PDT 2009","NO_QUERY","quanvu","@alydesigns i was out most of the day so didn't get much done "
"0","1467813992","Mon Apr 06 22:20:38 PDT 2009","NO_QUERY","swinspeedx","one of my friend called me, and asked to meet with her at Mid Valley today...but i've no time *sigh* "
"0","1467814119","Mon Apr 06 22:20:40 PDT 2009","NO_QUERY","cooliodoc","@angry_barista I baked you a cake but I ated it "
"0","1467814180","Mon Apr 06 22:20:40 PDT 2009","NO_QUERY","viJILLante","this week is not going as i had hoped "
"0","1467814192","Mon Apr 06 22:20:41 PDT 2009","NO_QUERY","Ljelli3166","blagh class at 8 tomorrow "
"0","1467814438","Mon Apr 06 22:20:44 PDT 2009","NO_QUERY","ChicagoCubbie","I hate when I have to call and wake people up "
"0","1467814783","Mon Apr 06 22:20:50 PDT 2009","NO_QUERY","KatieAngell","Just going to cry myself to sleep after watching Marley and Me.  "
"0","1467814883","Mon Apr 06 22:20:52 PDT 2009","NO_QUERY","gagoo","im sad now  Miss.Lilly"
"0","1467815199","Mon Apr 06 22:20:56 PDT 2009","NO_QUERY","abel209","ooooh.... LOL  that leslie.... and ok I won't do it again so leslie won't  get mad again "
```

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Cargar el conjunto de datos Sentiment140 de Twitter
url =
'https://raw.githubusercontent.com/crwong/cs224u-project/master
/data/sentiment/training.1600000.processed.noemoticon.csv'
column_names = ['target', 'ids', 'date', 'flag', 'user',
'text']
data = pd.read_csv(url, encoding='ISO-8859-1',
names=column_names)

# Seleccionar una muestra aleatoria para reducir el tamaño del
conjunto de datos
data = data.sample(frac=0.9, random_state=42)

# Dividir los datos en características (X) y etiquetas (y)
X = data['text']
y = data['target']

# Convertir el texto a características numéricas usando TF-IDF
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(X)

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Crear un clasificador de Bosques Aleatorios (Random Forest)
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# Entrenar el clasificador con los datos de entrenamiento
clf.fit(X_train, y_train)

# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(X_test)

# Calcular la precisión del modelo
accuracy = accuracy_score(y_test, y_pred)
print(f"Precisión del modelo de Bosques Aleatorios: {accuracy:.2f}")

# Matriz de confusión para evaluar el rendimiento del modelo
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicción')
plt.ylabel('Valor Real')
plt.title('Matriz de Confusión')
plt.show()
```

¿Qué has aprendido?

- Utilizar el modelo de aprendizaje: árboles de decisión.
- Algoritmo de árboles de clasificación y regresión (CART).
- Algoritmo de bosque aleatorio o *Random Forest* (RF).
- Comparar la precisión entre RF y CART.
- Aplicar en distintos sectores el modelo de árbol de decisión.
- ¿te parece poco?



"Ya no requiere un presupuesto multimillonario para que la IA funcione en su empresa. Representa una oportunidad para nivelar el campo de juego para las empresas más pequeñas."

Nichole Jordan, managing partner en Grant Thornton LLP