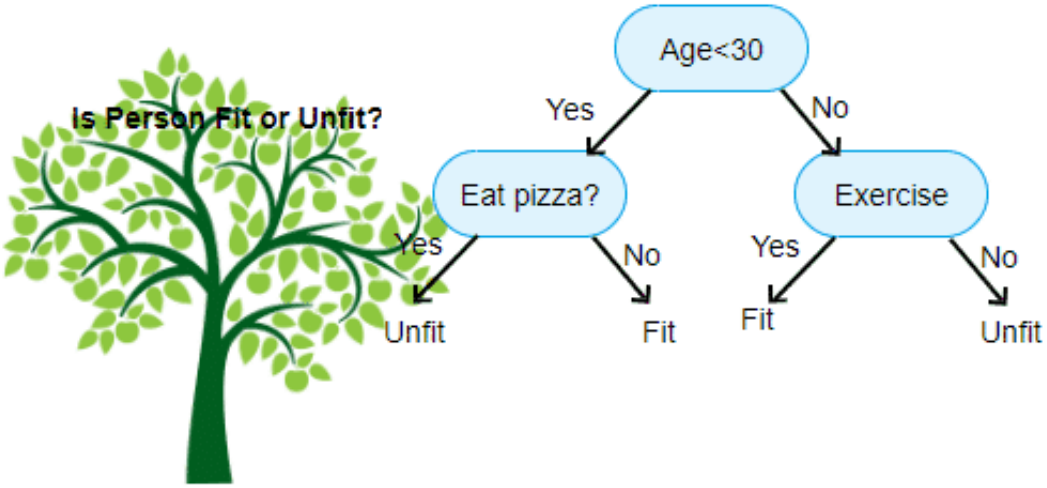
UNIVERSITAT Carlemar

Tema 4

Árbol de decisiones



Gerard Alba Soler

Matias Davila
Aprendizaje automático

UNIVERSITAT Carlemar

1) Describiendo la implementación del modelo de aprendizaje automático. En el código que nos ha entregado el profesor podemos apreciar la utilización de un modelo de aprendizaje automatico, en este modelo se usa un conjunto de datos de cáncer de mama de Wisconsin de Scikit-learn. Este modelo busca descubrir las pequeñas diferencias en células con esto categorizarlas e identificar si acaso hay existencia de tumores o cáncer, benigno o maligno respectivamente.
   Para esto importamos las bibliotecas necesarias como Scikit-learn, numpy, decisiontreeclassifier y train_test_split. Cargamos el conjunto de datos a través de la función load_breast_cancer.
   Luego separamos los datos identificando datos de prueba y datos de entrenamiento (X_train, X_test, Y_train & Y_test), gracias a la función train_test_split() y el uso de stratify garantizamos que la proporción de etiquetas para los conjuntos de prueba y entrenamiento sean similar.
   Luego se crea un modelo de árbol de decisión DecisionTreeClassifier con un valor random_state igual a 0 o establecido en 0. Este se ajusta con el método fit() para luego calcular y mostrar la precisión del modelo en los conjuntos de datos que utilizamos gracias al método score().

2) Comentario del código.

```python
# Importa las bibliotecas necesarias
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

# Carga el conjunto de datos de cáncer de mama
cancer = load_breast_cancer()

# Divide los datos en conjuntos de entrenamiento y prueba
X_train, X_test, Y_train, Y_test = train_test_split(cancer.data, cancer.target, stratify=cancer.target, random_state=42)

# Crea un modelo de árbol de decisión con profundidad máxima limitada a 4
tree = DecisionTreeClassifier(max_depth=4, random_state=0)

# Ajusta el modelo a los datos de entrenamiento
tree.fit(X_train, Y_train)

# Calcula y muestra la precisión en los datos de entrenamiento y prueba
print("Precision en datos entrenamiento: {:.3f}".format(tree.score(X_train, Y_train)))
print("Precision en datos de test: {:.3f}".format(tree.score(X_test, Y_test)))
```

   En este código ajustamos el árbol de decisión para evitar un posible sobreajuste esto gracias al uso de max_depth que se establece en 4, al utilizar esto se observa que hay una precisión ligeramente menor en los datos de entrenamiento, pero una precisión mayor en los datos de prueba.

3) En el punto 3 trabajamos directamente con el grafico o la visualización de los datos que agrupamos en los modelos previos. Para esto el código hace uso de la función export_graphviz() la cual exporta el árbol de decisión, a través de tree.dot se entra a un archivo DOT por ultimo con graphviz visualizamos este árbol.
4)

Matias Davila
Aprendizaje automático