Karar ağaçları, makine öğrenimi alanında sınıflandırma ve regresyon problemlerini çözmek için kullanılan bir algoritma türüdür. Bu algoritma, veri kümesindeki özelliklere ve hedef değişkenine dayalı olarak bir ağaç yapısı oluşturur. Her iç düğüm, bir özellik testi yapar ve bu test sonucuna göre farklı dallara ayrılır. Yaprak düğümleri ise sonuçları temsil eder.

Karar ağaçları, veriyi basit ve anlaşılır bir şekilde modellemek için kullanılır. Ayrıca, algoritmanın çıktısı insan tarafından yorumlanabilir, böylece veri setinin yapısını ve özelliklerini daha iyi anlamak mümkün olur.

Algoritma, genellikle aşağıdaki adımları içerir:

1- Kök Düğümünün Oluşturulması: Başlangıçta, veri kümesindeki tüm örnekler göz önünde bulundurularak bir kök düğüm oluşturulur.
2- En İyi Bölme Noktasının Seçilmesi: Veri kümesini en iyi bölme noktasına göre bölmek için bir özellik seçilir. Bu seçim, genellikle veri setinin homojenliğini artıracak şekilde yapılır.
3- Dalların Oluşturulması: Seçilen özellik ve bölme noktasına göre kök düğüm altında yeni dallar oluşturulur.
4- Dalların Büyütülmesi veya Düğümün Yaprak Düğümü Olarak İşaretlenmesi: Belirli bir duruma ulaşıldığında, dallar ya yeni dallar oluşturularak büyütülür ya da yaprak düğümü olarak işaretlenir.
5- Karar Ağacının Oluşturulması: Bu adımlar, veri kümesi bölünemez hale gelene kadar tekrarlanır. Sonuç olarak, bir karar ağacı oluşturulur.

Karar ağaçları, sınıflandırma problemlerinin yanı sıra regresyon problemleri için de kullanılabilir. Sınıflandırma için kullanıldığında, her yaprak düğümü bir sınıfı veya etiketi temsil ederken, regresyon için kullanıldığında, yaprak düğümleri bir sayısal değeri temsil eder.

Karar ağaçlarının avantajları arasında basitlik, yorumlanabilirlik ve veri seti içindeki önemsiz özelliklerin otomatik olarak eleme yeteneği bulunur. Ancak, tek başına karar ağaçları genellikle aşırı uyum (overfitting) eğilimindedir. Bu nedenle, ensemble yöntemler, özellikle rastgele ormanlar gibi, genellikle daha iyi performans sağlar.

Karar ağacı kodu :

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Veri setini yükleme
iris = load_iris()
X = iris.data
y = iris.target

# Veriyi eğitim ve test setlerine ayırma
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Karar ağacı modelini oluşturma ve eğitme
classifier = DecisionTreeClassifier(random_state=42)
classifier.fit(X_train, y_train)

# Test seti üzerinde tahmin yapma
y_pred = classifier.predict(X_test)

# Doğruluk skorunu hesaplama
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Doğruluk:", accuracy)
```