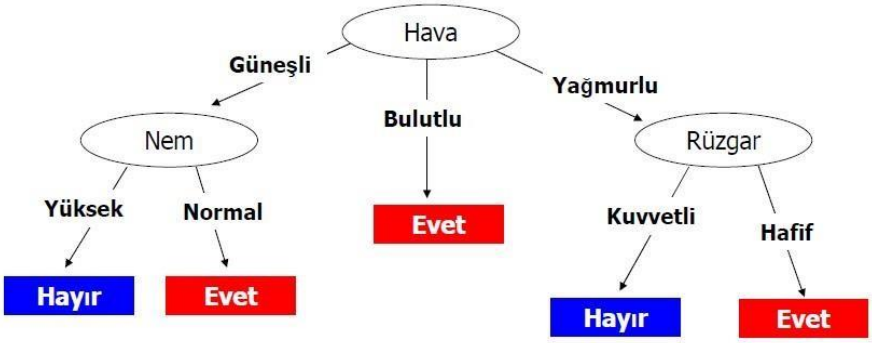
6.2.1. Karar Ağaçları

Karar ağacı algoritması, denetimli öğrenme kategorisine girer. Hem regresyon hem de sınıflandırma problemlerini çözmek için kullanılılar. Karar ağacı, her yaprak düğümün bir sınıf etiketine karşılık geldiği ve özniteliklerin ağacın iç düğümünde temsil edildiği sorunu çözmek için ağaç temsilini kullanır. Karar ağacını kullanarak herhangi bir boole fonksiyonunu ayrık öznitelikler üzerinde temsil edebiliriz.

Karar ağacı öğrenmesi, bir karar ağacını, bir öğeyle ilgili (dallarda temsil edilen) gözlemlerden öğenin hedef değeri (yapraklarda temsil edilen) ile ilgili sonuçlara gitmek için bir tahmin modeli olarak kullanır. İstatistik, veri madenciliği ve makine öğrenmesinde kullanılan öngörülü modelleme yaklaşımlarından biridir. Hedef değişkenin ayrı bir değer kümesi alabileceği ağaç modellerine sınıflandırma ağaçları denir; bu ağaç yapılarında yapraklar sınıf etiketlerini ve dallar bu sınıf etiketlerine yol açan özelliklerin birleşimlerini temsil eder. Hedef değişkenin sürekli değerler alabileceği karar ağaçlarına (tipik olarak gerçek sayılar) regresyon ağaçları denir. Karar analizinde, bir karar ağacı, kararları ve karar almayı görsel ve açık bir şekilde temsil etmek için kullanılabilir. Veri madenciliğinde, bir karar ağacı verileri tanımlar, ancak sonuçta ortaya çıkan sınıflandırma ağacı karar verme için bir girdi olabilir.

Karar ağaçları metotu, giriş verisinin bir algoritma yardımıyla gruplara bölünerek tüm elemanlarının aynı sınıf etiketine sahip olması için yapılan sınıflama işlemidir. Giriş verisinin bir kümeleme algoritması yardımıyla tekrar tekrar gruplara bölünmesine dayanır. Grubun tüm elemanları aynı sınıf etiketine sahip olana kadar kümeleme işlemi derinlemesine devam eder.



Karar ağacı kullanılırken yapılan bazı varsayımlar aşağıdadır:

- Başlangıçta, tüm eğitim seti kök olarak kabul edilir.
- Özellik değerlerinin kategorik olması tercih edilir. Değerler sürekli ise, model oluşturmadan önce ayrıklaştırılırlar.
- Öznitelik değerleri temelinde, kayıtlar özyinelemeli olarak dağıtılır.
- Öznitelikleri kök veya dahili düğüm olarak sıralamak için istatistiksel yöntemler kullanılır.

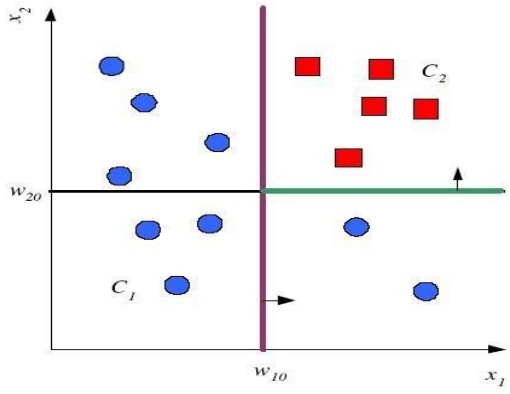
Karar ağacı tipleri ikiye ayrılır:

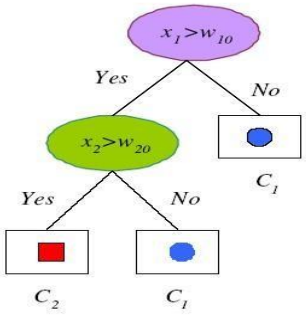
- Entropiye dayalı sınıflandırma ağaçları (ID3, C4.5)
- Regresyon ağaçları (CART).

Karar ağaçları çok boyutlu (özellikli) veriyi belirlenmiş şartlara bağlı olarak parçalara böler. Her adımda verinin hangi özelliği üzerinde işlem yapılacağına karar vermek çok büyük bir kombinasyonun çözümüyle mümkündür. Örneğin, 5 özellik ve 20 örneğe sahip bir veride 10^6 dan fazla sayıda farklı karar ağacı oluşturulabilir. Bu sebeple her parçalanmanın metodolojik olması gerekir. Quinlan'e göre veri bir özelliğine göre bölündüğünde elde edilen her bir veri kümesinin belirsizliği minimum ve dolayısıyla bilgi kazancı maksimum ise en iyi seçim yapılmıs demektir. Buna göre önerdiği ilk algoritma ID3'te tek tek özellik vektörleri incelenir ve en yüksek bilgi kazancına sahip özellik, ağaçta dallanma yapmak için tercih edilir.

Karar Ağacı Algoritması:
Karar ağaçları eğiticili öğrenme için çok yaygın bir yöntemdir. Algoritmanın adımları:

1) T öğrenme kümesini oluşturulur.
2) T kümesindeki örnekleri en iyi ayıran nitelikler belirlenir.
3) Seçilen nitelik ile ağacın düğümleri oluşturulur ve herbir düğümde alt düğümler veya ağacın yapraklarını oluşturulur. Alt düğümlere ait alt veri kümesinin örneklerini belirlenir
4) 3. adımda oluşturulan her alt veri kümesi için
    - Örneklerin hepsi aynı sınıfa aitse
    - Örnekleri bölecek nitelik kalmamışsa
    - Kalan niteliklerin değerini taşıyan örnek yoksa işlemi sonlandır. Diğer durumda alt veri kümesini ayırmak için 2. adımdan devam edilir.





Ezber (Overfitting: Aşırı Uyum):

- Tüm makine öğrenmesi yöntemlerinde verinin ana hatlarının modellenmesi esas alındığı için öğrenme modelinde ezberden (overfitting) kaçınılmalıdır.
- Tüm karar ağaçları önlem alınmazsa ezber yapar. Bu yüzden ağaç olusturulurken veya olusturulduktan sonra budama yapılmalıdır.

Ağaç Budama:

Budama, sınıflandırmaya katkısı olmayan bölümlerin karar ağacından çıkarılması islemidir. Bu sayede karar ağacı hem sade hem de anlaşılabilir hale gelir. İki çesit budama yöntemi vardır;

- Ön budama
- Sonradan budama

Ön budama islemi ağaç oluşturulurken yapılır. Bölünen nitelikler, değerleri belli bir esik değerinin (hata toleransının) üstünde değilse o noktada ağaç bölümleme islemi durdurulur ve o an elde bulunan kümedeki baskın sınıf etiketi, yaprak olarak olusturulur.

Sonradan Budama: Sonradan budama islemi ağaç olusturulduktan sonra devreye girer. Alt ağaçları silerek yaprak olusturma, alt ağaçları yükseltme, dal kesme seklinde yapılabilir.

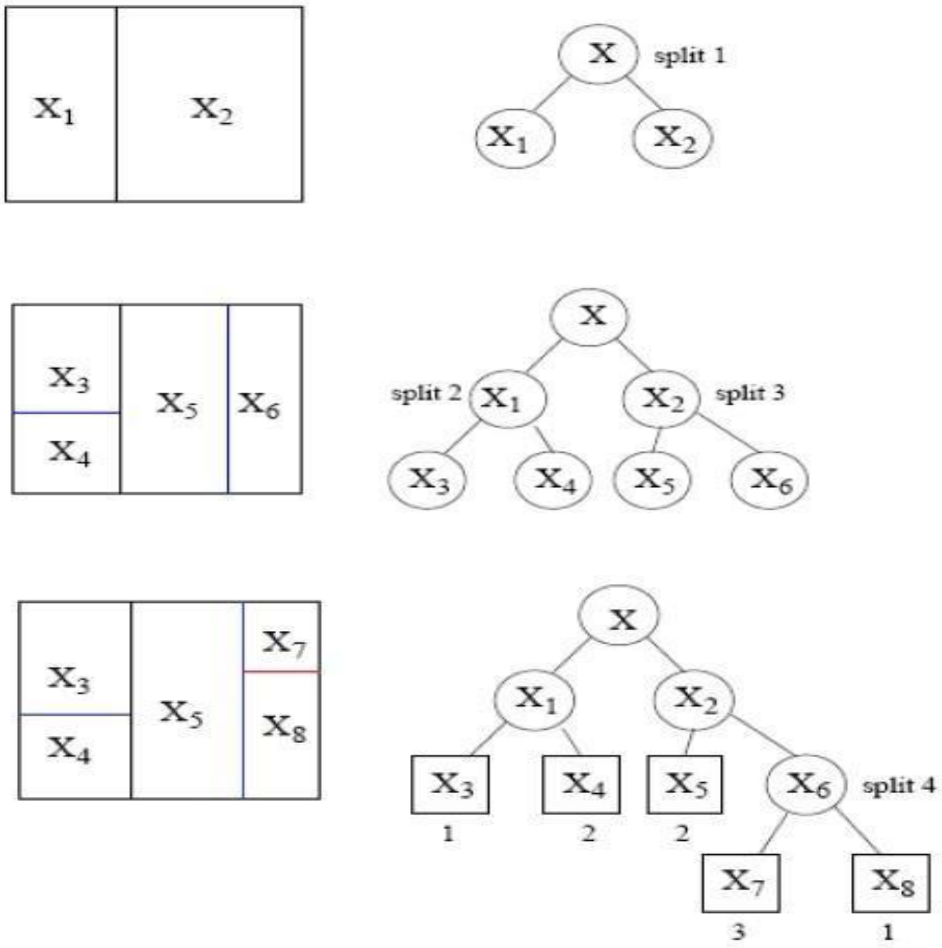
Aşırı uyumu önlemek için ağacı büyütmeyi durdurabiliriz, ancak durdurma kriteri miyop olma eğilimindedir. Bu nedenle standart yaklaşım, "dolu" bir ağaç yetiştirmek ve ardından budama yapmaktır. Düğümdeki noktalar için yanlış bir sınıflandırma yapma olasılığı olduğu da unutulmamalıdır. Tüm ağaç için yanlış sınıflandırma olasılığını elde etmek için, toplam olasılık formülüne göre yaprak düğüm içi hata oranının ağırlıklı toplamı hesaplanır.

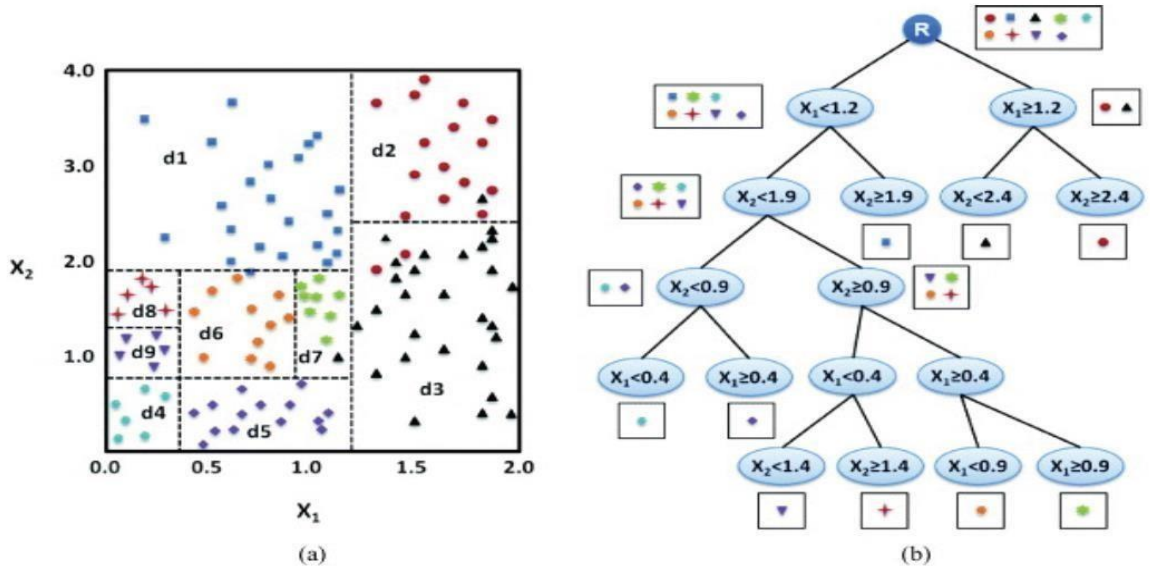
Spesifik olarak, ana düğüm için ağırlıklı yanlış sınıflandırma oranının, sol ve sağ alt düğümlerin ağırlıklı yanlış sınıflandırma oranlarının toplamından daha büyük veya buna eşit olacaktır. Yeniden ikame hata oranını en aza indirirsek, her zaman daha büyük bir ağacı tercih edeceğimiz anlamına gelir. Aşırı uyuma karşı hiçbir savunma yoktur. Aşırı büyümüş ağaç er ya da geç budanacaktır. Ne zaman duracağınıza karar vermenin birkaç yolu vardır: **O** Tüm terminal düğümleri saf olana kadar devam edilir.

- Her bir terminal düğümündeki veri sayısı belirli bir eşikten, örneğin 5'ten, hatta 1'den büyük olmayana kadar devam edilir.
- Ağaç yeterince büyük olduğu sürece, ilk ağacın boyutu kritik değildir.

Buradaki anahtar, ilk ağacı yeniden budamadan önce yeterince büyük yapmaktır!

Sınıflandırma Ağaçları:





Karar Ağacında en büyük zorluk, her seviyede kök düğüm için özniteliğin tanımlanmasıdır. Bu işlem öznitelik seçimi olarak bilinir. İki popüler öznitelik seçim ölçüsü bulunmaktadır:

1) Bilgi Kazancı
2) Gini İndeksi

1) Bilgi Kazancı

Eğitim örneklerini daha küçük alt kümelere bölmek için karar ağacında bir düğüm kullandığımızda entropi değişir. Bilgi kazancı, entropideki bu değişimin bir ölçüsüdür.

Tanım: Diyelim ki S bir örnekler kümesi, A bir nitelik, Sv, S'nin A = v ile alt kümesi ve Değerler (A), A'nın tüm olası değerlerinin kümesidir, o zaman

$$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|}.Entropy(S_v)$$

Örnek:

X = {a,a,a,b,b,b,b,b} kümesi
için Toplam örnek: 8 b: 5
örnekleri a: 3'ün örnekleri

$$Entropy H(X) = -\left[\left(\frac{3}{8}\right) log_2 \frac{3}{8} + \left(\frac{5}{8}\right) log_2 \frac{5}{8}\right]$$

= -[0.375 * (-1.415) + 0.625 * (-0.678)]

=-(-0.53-0.424)

= 0.954

Bilgi Kazanımını Kullanarak Karar Ağacı Oluşturma Gereklilikler:

- Kök düğümle ilişkili tüm eğitim örnekleriyle başlanır
- Her bir düğümün hangi öznitelikle etiketleneceğini seçmek için bilgi kazancı kullanılır.
- Not: Hiçbir kökten yaprağa yol, aynı ayrık özniteliği iki kez içermemelidir
- Her alt ağacı, ağaçta o yolda sınıflandırılacak eğitim örneklerinin alt kümesinde yinelemeli olarak oluşturulur.

Sınır vakaları:

- Tüm pozitif veya tüm negatif eğitim örnekleri kalırsa, o düğümü buna göre “evet” veya “hayır” olarak etiketlenr.
- Hiçbir öznitelik kalmazsa, o düğümde kalan eğitim örneklerinin çoğunluk oyu ile etiketlenir.
- Örnek kalmadıysa, ebeveynin eğitim örnekleri çoğunluk oyu ile etiketlenir.

Örnek:
Şimdi aşağıdaki veriler için Bilgi kazanımını kullanarak bir Karar Ağacı çizelim. Eğitim seti:
3 özellik ve 2 sınıf

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Burada 3 özellik ve 2 çıktı sınıf var. Bilgi kazanımını kullanarak bir karar ağacı oluşturmak. Her bir özellik alınr ve her bir özellik için bilgi kazancı hesaplanır.

ID3 Algoritması:

Sadece kategorik veri ile çalısan bir yöntemdir. Her iterasyonun ilk adımında veri örneklerine ait sınıf bilgilerini tasıyan vektörün entropisi belirlenir. Daha sonra özellik vektörlerinin sınıfa bağımlı entropileri hesaplanarak ilk adımda hesaplanan entropiden çıkartılır. Bu sekilde elde edilen değer ilgili özellik vektörüne ait kazanç değeridir. En büyük kazanca sahip özellik vektörü ağacın o iterasyonda belirlenen dallanmasını gerçeklestirir.

Örnek:

2 özellik vektörü (V1 ve V2) ile S sınıf vektörüne sahip 4 örnekli veri kümesi verilmistir. ID3 algoritması ile ilk dallanma hangi özellik üzerinde gerçeklesir ?

- H(S) - H(V1,S)
- H(S) - H(V2,S)

V1	V2	S
A	C	E
B	C	F
B	D	E
B	D	F

Sınıf Entropisi

$$H(S) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

V1 Entropisi

$$H(V1) = \frac{1}{4}H(A) + \frac{3}{4}H(B)$$

$$= \frac{1}{4}0 - \frac{3}{4}\left(\frac{1}{3}\log_2\frac{1}{3} + \frac{2}{3}\log_2\frac{2}{3}\right) = 0 + \frac{3}{4}0.9183 = 0.6887$$

V2 Entropisi

$$H(V2) = \frac{1}{2}H(C) + \frac{1}{2}H(D) = \frac{1}{2} + \frac{1}{2} = 1$$

V1 seçilir...

C4.5 Algoritması:

ID3 algoritmasının nümerik özellik içeren veriye uygulanabilen seklidir. ID3'ten tek farkı nümerik özelliklerin kategorik hale getirilebilmesini sağlayan bir esikleme yöntemini içermesidir. Temel mantık nümerik özellik vektöründeki tüm değerler ikili olarak ele alınarak ortalamaları esik olarak denenir. Hangi esik değeriyle bilgi kazanımı en iyi ise o değer seçilir. Seçilen esiğe göre özellik vektörü kategorize edilir ve ID3 uygulanır.

Örnek:
Kredilendirmede "Mükemmel", "İyi" ve "Kötü" değerleri alabilen bir özelliğimiz vardır.

Toplam 14 gözlem var. Bunlardan 7'si Normal Sorumluluk sınıfına, 7'si ise Yüksek Sorumluluk Sınıfına aittir. Yani kendi başına eşit bir bölünmedir. En üst satırda özetlersek, kredi notu özelliği için Mükemmel değerine sahip 4 gözlem olduğunu görebiliriz. Ayrıca, “Mükemmel” Kredi Notu için hedef değişkenimin nasıl bölündüğünü de görülebiliyor. Kredi notu için “Mükemmel” değeri alan gözlemler için Normal Sorumluluk sınıfına ait 3 adet ve Yüksek Sorumluluk sınıfına ait sadece 1 adet gözlem bulunmaktadır. Benzer şekilde diğer Kredi Notu değerleri için de beklenmedik durum tablosundan bu değerler bulunabilir.

Credit Rating	Liability		
	Normal	High	Total
Excellent	3	1	4
Good	4	2	6
Poor	0	4	4
Total	7	7	14

Bu örnek için, beklenmedik durum tablosunu, hedef değişkeninin entropisini kendi başına hesaplamak için kullanılacak ve ardından özellik, kredi notu hakkında ek bilgiler verilen hedef değişkenin entropisi hesaplanacak. Bu, "Kredi Notu"nun hedef değişkenin "Yükümlülük" için ne kadar ek bilgi sağladığını hesaplanmasına olanak sağlayacak.

$$E(Liability) = -\frac{7}{14}\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\log_2\left(\frac{7}{14}\right)$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= 1$$

Hedef değişkenin entropisi, “Normal” ve “Yüksek” sınıf etiketi arasındaki eşit bölünme nedeniyle maksimum düzensizlikte 1'dir. Bir sonraki adım, kredi puanı hakkında ek bilgi verilen hedef değişken Yükümlülük'ün entropisini hesaplamaktır. Bunun için her Kredi Puanı değeri için Sorumluluk entropisi hesaplanacak ve her bir değerde sonuçlanan gözlemlerin oranının ağırlıklı ortalaması kullanılarak bunlar eklenecektir. Neden ağırlıklı ortalama kullanıldığı, bunu karar ağaçları bağlamında tartışıldığında daha da netleşecektir.

$$E(Liability \mid CR = Excellent) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0.811$$

$$E(Liability \mid CR = Good) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right) \approx 0.918$$

$$E(Liability \mid CR = Poor) = -0\log_2(0) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$$

Weighted Average:

$$E(Liability \mid CR) = \frac{4}{14} \times 0.811 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0 = 0.625$$

Credit Rating	Liability		
	Normal	High	Total
Excellent	3	1	4
Good	4	2	6
Poor	0	4	4
Total	7	7	14

Kredi Derecelendirme özelliği verilen hedef değişken için entropi elde edildi. Artık bu özelliğin ne kadar bilgilendirici olduğunu görmek için Kredi Notundan Yükümlülük Bilgi Kazancı hesaplanmalıdır.

Inf ormation Gain:

$$IG(Liability, CR) = E(Liability) - E(Liability \mid CR)$$

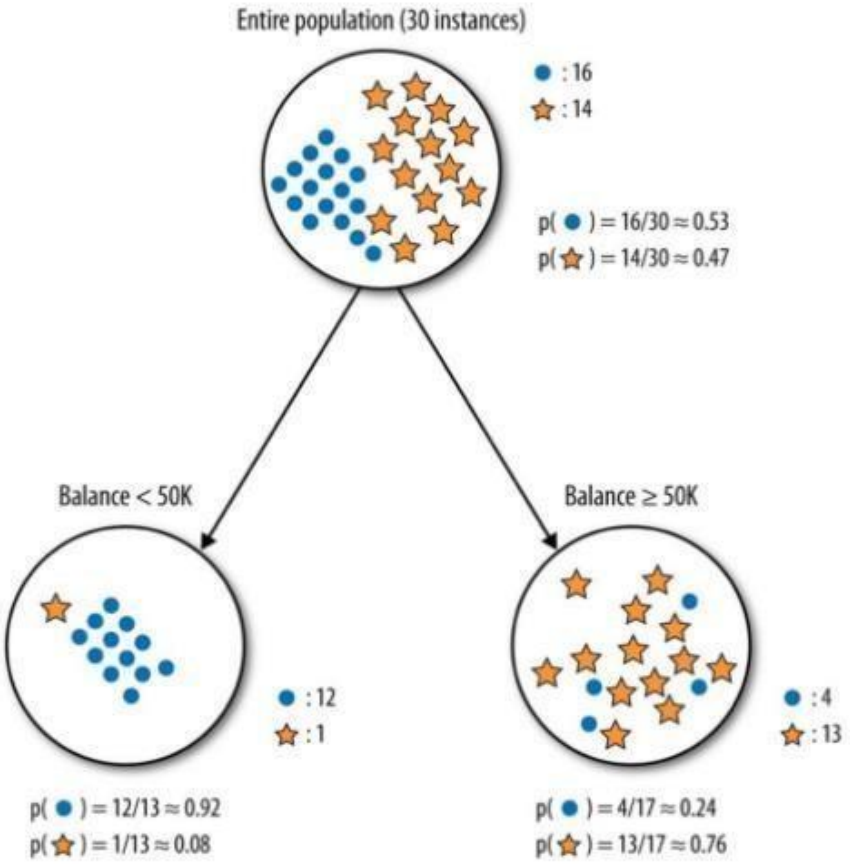
$$= 1 - 0.625 = 0.375$$

Kredi Notunu bilmek, hedef değişkenimiz olan Sorumluluk etrafındaki belirsizliği azaltmamıza yardımcı oldu! İyi bir özelliğin yapması gereken de bu değil mi? Hedef değişkenimiz hakkında bize bilgi verir misiniz? İşte tam olarak bu, karar ağaçlarının, her bir bölünmede hedef değişkeni tahmin etmeye yaklaşmak ve aynı zamanda ağacı bölmeyi ne zaman durduracağını

belirlemek için düğümlerini hangi özelliğin üzerine böleceğini belirlemek için entropi ve bilgi kazancını nasıl ve neden kullandığıdır! (elbette maksimum derinlik gibi hiper parametrelere ek olarak).

Örnek: Karar Ağacı

Bir kişiye verilen bir kredinin bir zararla sonuçlanıp sonuçlanmayacağını tahmin etmek için bir karar ağacı oluşturduğumuz bir örneği ele alalım. Tüm popülasyonumuz 30 örnekten oluşmaktadır. 16'sı silinen sınıfa, diğer 14'ü silinmeyen sınıfa aittir. İki değer alabilen "Bakiye" -> "< 50K" veya ">50K" ve üç değer alabilen "Konut" -> "KENDİ", "KİRALIK" veya "DİĞER" olmak üzere iki özelliğimiz var. Entropi ve Bilgi Kazanımı kavramlarını kullanarak bir karar ağacı algoritmasının hangi özniteliğin ilk olarak bölüneceğine ve hangi özelliğin daha fazla bilgi sağladığına veya hedef değişkenimiz hakkındaki belirsizliği ikisinden daha fazla azalttığına nasıl karar vereceğini göstereceğim.



Özellik 1: Denge

Noktalar, sınıf hakkı olan veri noktalarıdır ve yıldızlar, silinmeyenlerdir. Ana düğümü öznitelik dengesine bölmek bize 2 alt düğüm verir. Sol düğüm, silme sınıfından 12/13 (0.92 olasılık) gözlem ile toplam gözlemlerin 13'ünü ve sınıfın yazılmayan sınıfından sadece 1/13 (0.08

olasılık) gözlemi alır. Sağ düğüm, silinmeyen sınıftan 13/17(0.76 olasılık) ve silinen sınıftan 4/17 (0.24 olasılık) ile toplam gözlemin 17'sini alır.

Ana düğümün entropisini hesaplayalım ve Denge üzerinde bölerek ağacın ne kadar belirsizliği azaltabileceğini görelim.

$$E(Parent) = -\frac{16}{30}\log_2\left(\frac{16}{30}\right) - \frac{14}{30}\log_2\left(\frac{14}{30}\right) \approx 0.99$$

$$E(Balance < 50K) = -\frac{12}{13}\log_2\left(\frac{12}{13}\right) - \frac{1}{13}\log_2\left(\frac{1}{13}\right) \approx 0.39$$

$$E(Balance > 50K) = -\frac{4}{17}\log_2\left(\frac{4}{17}\right) - \frac{13}{17}\log_2\left(\frac{13}{17}\right) \approx 0.79$$

Weighted Average of entropy for each node:

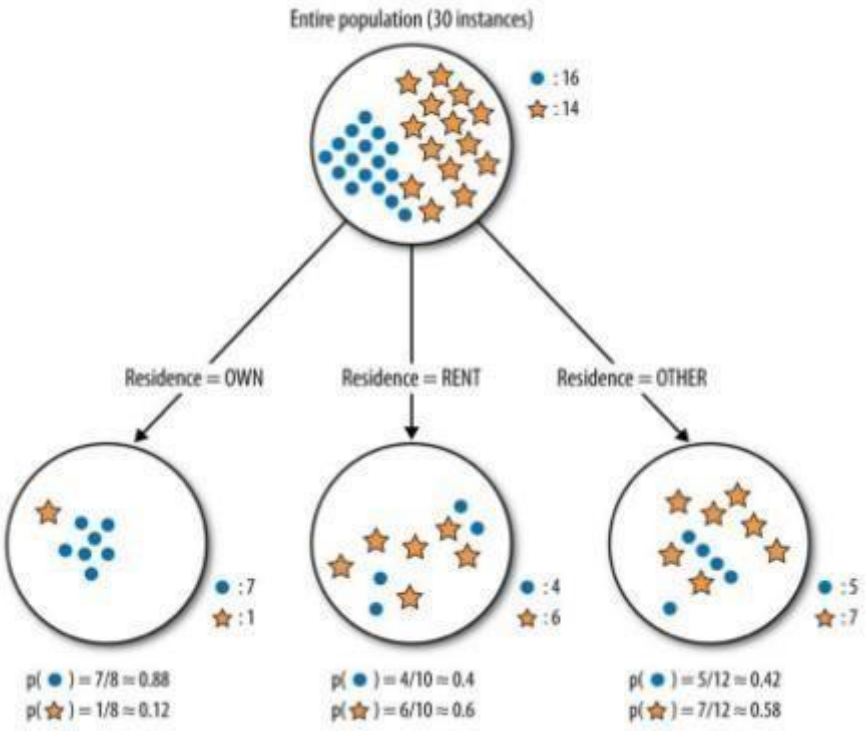
$$E(Balance) = \frac{13}{30} \times 0.39 + \frac{17}{30} \times 0.79$$
$$= 0.62$$

Information Gain:

$$IG(Parent, Balance) = E(Parent) - E(Balance)$$
$$= 0.99 - 0.62$$
$$= 0.37$$

Özelliğe göre bölme, “Denge” hedef değişkenimiz üzerinde 0.37'lik bir bilgi kazanımına yol açar. Aynı şeyi nasıl karşılaştırdığını görmek için “Konut” özelliği için yapalım.

Özellik 2: Konut



Ağacı Residence'ta bölmek bize 3 alt düğüm verir. Sol alt düğüm, silinen sınıftan 7/8 (0.88 olasılık) gözlem ile toplam gözlemlerin 8'ini ve silinmeyen sınıftan sadece 1/8 (0.12 olasılık) gözlem alır. Orta alt düğümler, silinen sınıftan 4/10 (0,4 olasılık) ve silinmeyen sınıftan 6/10(0,6 olasılık) gözlem ile toplam gözlemlerin 10'unu alır. Sağ alt düğüm, silme sınıfından 5/12 (0.42 olasılık) gözlem ve silinmeyen sınıftan 7/12 (0.58) gözlem ile toplam gözlemlerin 12'sini alır. Ana düğümün entropisini zaten biliyoruz. “Konut”tan elde edilen bilgi kazancını hesaplamak için bölmeden sonraki entropiyi hesaplamamız yeterlidir.

$$E(Residence = OWN) = -\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) \approx 0.54$$

$$E(Residence = RENT) = -\frac{4}{10}\log_2\left(\frac{4}{10}\right) - \frac{6}{10}\log_2\left(\frac{6}{10}\right) \approx 0.97$$

$$E(Residence = OTHER) = -\frac{5}{12}\log_2\left(\frac{5}{12}\right) - \frac{7}{12}\log_2\left(\frac{7}{12}\right) \approx 0.98$$

Weighted Average of entropies for each node:

$$E(Residence) = \frac{8}{30} \times 0.54 + \frac{10}{30} \times 0.97 + \frac{12}{30} \times 0.98 = 0.86$$

Weighted Average of entropies for each node:

$$E(\textit{Residence}) = \frac{8}{30} \times 0.54 + \frac{10}{30} \times 0.97 + \frac{12}{30} \times 0.98 = 0.86$$

Information Gain:

$$\begin{aligned} IG(\textit{Parent}, \textit{Residence}) &= E(\textit{Parent}) - E(\textit{Residence}) \\ &= 0.99 - 0.86 \\ &= 0.13 \end{aligned}$$

Denge özelliğinden gelen bilgi kazanımı, Konut'tan elde edilen bilgiden neredeyse 3 kat daha fazla! Geri dönüp grafiklere bir göz atarsanız, Denge'de bölünen alt düğümlerin, Yerleşim düğümlerinden daha saf göründüğünü görebilirsiniz. Bununla birlikte, ikamet için en soldaki düğüm de çok saftır, ancak ağırlıklı ortalamaların devreye girdiği yer burasıdır. Bu düğüm çok saf olmasına rağmen, toplam gözlemlerin en az miktarına sahiptir ve bir sonuç, Yığındaki bölmeden toplam entropiyi hesapladığımızda saflığının küçük bir kısmına katkıda bulunur. Bu önemlidir çünkü bir özelliğin genel bilgi gücünü arıyoruz ve sonuçlarımızın bir özellikteki nadir bir değer tarafından çarpıtılmasını istemiyoruz.

Kendi başına Balance özelliği, hedef değişkenimiz hakkında Konuttan daha fazla bilgi sağlar. Hedef değişkenimizde daha fazla düzensizliği azaltır. Bir karar ağacı algoritması, Balance kullanarak verilerimizde ilk bölmeyi yapmak için bu sonucu kullanır. Bundan sonra, karar ağacı algoritması, bir sonraki hangi özelliği böleceğine karar vermek için her bölmede bu süreci kullanacaktır. Gerçek dünya senaryosunda, ikiden fazla özellik ile ilk bölme en bilgilendirici özellik üzerinde yapılır ve daha sonra her bölmede, her bir ek özellik için bilgi kazancının yeniden hesaplanması gerekir, çünkü her birinden elde edilen bilgi kazancı ile aynı olmaz. özellik kendi başına. Entropi ve bilgi kazancı, sonuçları değiştirecek bir veya daha fazla bölme yapıldıktan sonra hesaplanmalıdır. Bir karar ağacı, önceden tanımlanmış bir derinliğe ulaşana ya da hiçbir ek bölünme, genellikle hiper parametre olarak da belirlenebilen belirli bir eşiğin ötesinde daha yüksek bir bilgi kazanımına neden olana kadar derinleştikçe ve derinleştikçe bu işlemi tekrarlar!

Example: Decision Tree - Classification

Karar ağacı, bir ağaç yapısı şeklinde sınıflandırma veya regresyon modelleri oluşturur. Bir veri kümesini giderek daha küçük alt kümelere ayırırken aynı zamanda ilgili bir karar ağacı aşamalı olarak geliştirilir. Nihai sonuç, karar düğümleri ve yaprak düğümleri olan bir ağaçtır. Bir karar düğümünün iki veya daha fazla şubesi vardır (örn. Güneşli, Bulutlu ve Yağmurlu). Yaprak

düğümü bir sınıflandırmayı veya kararı temsil eder. Kök düğüm adı verilen en iyi tahmin ediciye karşılık gelen bir ağaçtaki en üstteki karar düğümü. Karar ağaçları hem kategorik hem de sayısal verileri işleyebilir.

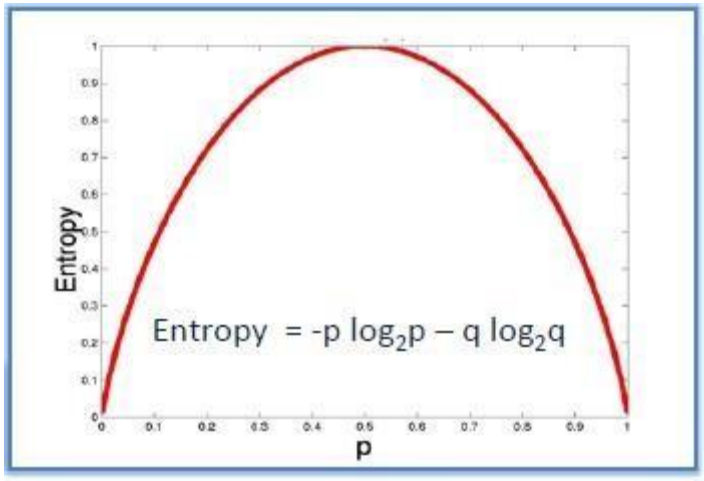
Algorithm

The core algorithm for building decision trees called ID3 by J. R. Quinlan which employs a topdown, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct a decision tree. In ZeroR model there is no predictor, in OneR model we try to find the single best predictor, naive Bayesian includes all predictors using Bayes' rule and the independence assumptions between predictors but decision tree includes all predictors with the dependence assumptions between predictors.

J. R. Quinlan tarafından ID3 olarak adlandırılan karar ağaçlarının, temel algoritması, olası dallar alanında yukarıdan aşağıya, geri izleme olmadan bir arama yapar. ID3, bir karar ağacı oluşturmak için Entropi ve Bilgi Kazancı kullanır. ZeroR modelinde tahmin edici yoktur, OneR modelinde tek en iyi tahmin ediciyi bulmaya çalışılır, saf Bayesian, Bayes kuralını ve tahmin ediciler arasındaki bağımsızlık varsayımlarını kullanan tüm tahmin edicileri içerir, ancak karar ağacı, tahmin ediciler arasındaki bağımlılık varsayımlarına sahip tüm tahmin edicileri içerir.

Entropy

Bir karar ağacı, bir kök düğümden yukarıdan aşağıya oluşturulur ve verileri benzer değerlere sahip (homojen) örnekler içeren alt kümelere ayırmayı içerir. ID3 algoritması, bir örneğin homojenliğini hesaplamak için entropiyi kullanır. Örnek tamamen homojen (Olma olasılığı 1 ise olmma olasılığı sıfırdır. Tüm olasıklar toplamı bire eşit olmak zorundadır) ise entropi sıfırdır ve örnek eşit olarak bölünmüşse entropisi birdir.



Entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

y=Log2(x)=Log10(x)/0.3

Log10(2)=0.3, Log10(3)=0.477, Log10(5)=0.7, Log10(7)=0.845

Bir karar ağacı oluşturmak için aşağıdaki gibi sıklık tablolarını kullanarak iki tür entropi hesaplamamız gerekir:

a) Bir özelliğin sıklık tablosunu kullanan entropi:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

5/14=0.36

9/14=0.64

b) İki özelliğin sıklık tablosunu kullanan entropi:

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)*__E__(3,2) + **P**(Overcast)*__E__(4,0) + **P**(Rainy)*__E__(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

Information Gain

Bilgi kazancı, bir veri kümesi bir özniteliğe bölündükten sonra entropideki azalmaya dayanır. Bir karar ağacı oluşturmak, en yüksek bilgi kazancını (yani en homojen dalları) döndüren özniteliği bulmakla ilgilidir.

Adım 1: Hedefin entropisini hesaplanır.

Entropy(PlayGolf) = Entropy (5,9)

= Entropy (0.36, 0.64)

= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)

= 0.94

Adım 2: Veri kümesi daha sonra farklı niteliklere bölünür. Her dal için entropi hesaplanır. Daha sonra, bölme için toplam entropi elde etmek için orantılı olarak eklenir. Ortaya çıkan entropi, bölünmeden önceki entropiden çıkarılır. Sonuç, Bilgi Kazanımı veya entropideki azalmadır.

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

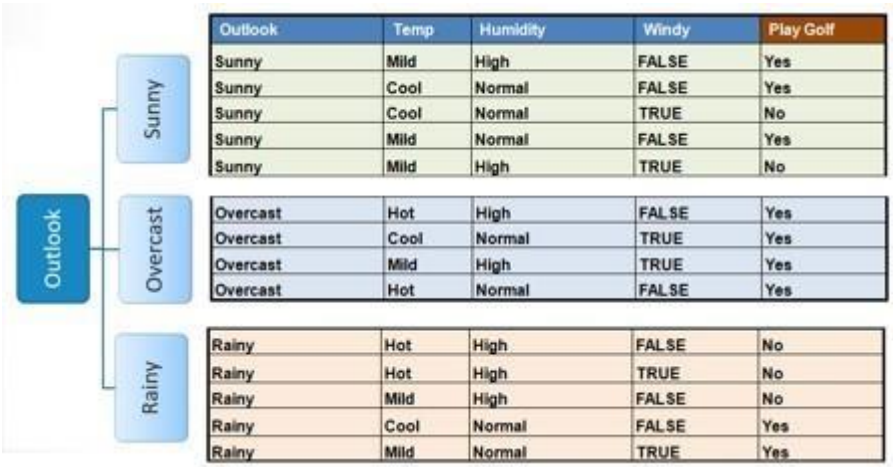
$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

Adım 3: Karar düğümü olarak en büyük bilgi kazancına sahip öznitelik seçilir, veri seti dallarına bölünür ve aynı işlemi her dalda tekrarlanır.

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



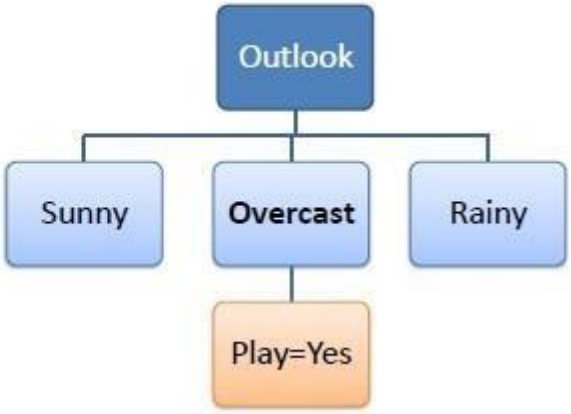
Outlook	Temp	Humidity	Windy	Play Golf
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No

Overcast	Hot	High	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes

Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes

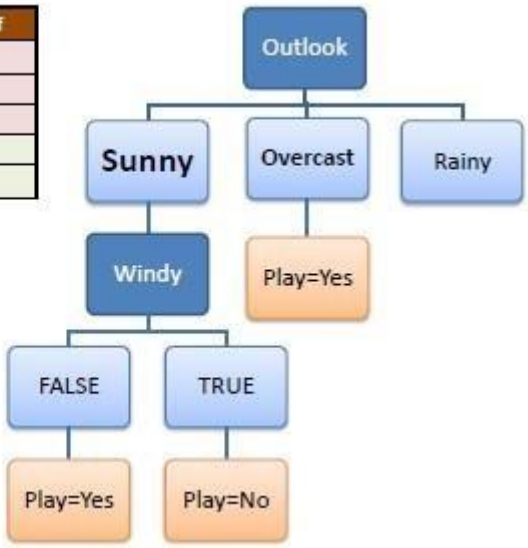
Step 4a: Entropisi 0 olan bir dal, bir yaprak düğümdür.

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes



Adım 4b: Entropisi 0'dan büyük olan bir dalın daha fazla bölünmesi gerekir.

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Adım 5: ID3 algoritması, tüm veriler sınıflandırılana kadar yaprak olmayan dallarda özyinelemeli olarak çalıştırılır.

Karar Ağacından Karar Kurallarına

Bir karar ağacı, kök düğümden yaprak düğümlere tek tek eşlenerek kolayca bir dizi kurala dönüştürülebilir.

R_1: IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2: IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3: IF (Outlook=Overcast) THEN Play=Yes

R_4: IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5: IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes