ML Notes
Ayşe Bilge İnce
30 January 2017

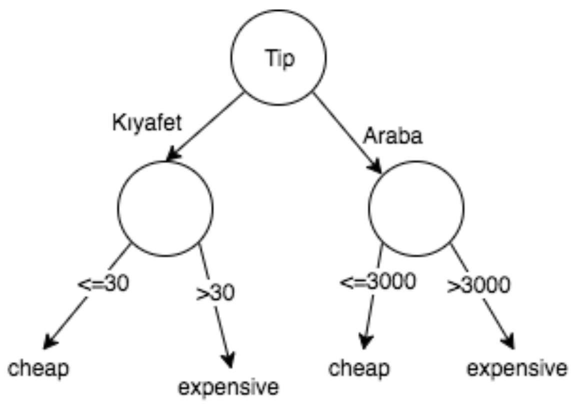
Machine Learning 101

Öğrenme Yöntemleri

1. Eğitmenli Öğrenme
2. Denetimsiz Öğrenme
3. Destekli Öğrenme

Karar Ağacı (ID3)

Parametrik olmayan, eğitimli bir yöntemdir. Sınıflandırma ve regresyon için kullanılırlar. Bir karar ağacı ara karar düğümü ve terminal yapraklardan oluşur. Yaprak düğümlerin bir output değeri vardır. Bu output sınıflandırmada sınıf kodu, regresyonda da nümerik bir değer olarak görünür. Karar ağaçlarında alt kümelere bölmenin amacı her alt kümeyi olabildiğince homojen hale getirmektir. Dezavantajı karar ağacı algoritmalarının greedy yöntemler olmasıdır.



Entropi

2-sınıflı Entropi(S) = -(p_+*log_2p_+ + p_-*log_2p_-)

n-sınıflı Entropi(S) => E(S) = $\sum_{i=1}^{n} -p_i * log_2 p_i$

Örnek 1:

Örnek Sayısı	Örnek Sınıfı
9	+
5	-

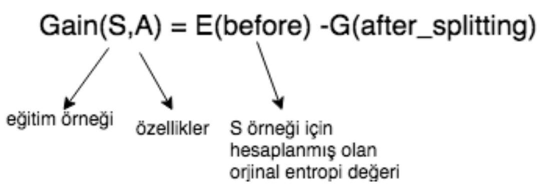
$p_+ = 9/(9+5) = 9/14$ (aynı eğitim sınıfında bulunan bir örneğin + sınıfta bulunma olasılığı)

$p_- = 5/14$ (aynı eğitim sınıfında bulunan bir örneğin - sınıfta bulunma olasılığı)

$E = - 9/14 * \log_2(9/14) - 5/14 * \log_2(5/14) = 0.94$

Kazanım

Karar ağaçlarında kökler, düğümler ve yapraklar kazanım değerine göre oluşturulur.



Örnek 2:

Aşağıdaki tabloyu eğitim seti olarak kabul edip, tabloyu ID3 karar ağacı algoritmasına göre sınıflandırınız:

Haftasonu	Hava Durumu	Ebeveyn Durumu	Para Durumu	Karar(Sınıf)
H1	Güneşli	Evet	Zengin	Sinema
H2	Güneşli	Hayır	Zengin	Tenis
H3	Rüzgarlı	Evet	Zengin	Sinema
H4	Yağmurlu	Evet	Fakir	Sinema
H5	Yağmurlu	Hayır	Zengin	Ev
H6	Yağmurlu	Evet	Fakir	Sinema
H7	Rüzgarlı	Hayır	Fakir	Sinema
H8	Rüzgarlı	Hayır	Zengin	Alışveriş
H9	Rüzgarlı	Evet	Zengin	Sinema
H10	Güneşli	Hayır	Zengin	Tenis

Not: ID3 algoritması WEKA aracında C4.5 olarak gösterilmektedir.

Öncelikle tüm tabloyu en doğru şekilde ikiye bölecek olan özellik seçilmelidir. Bunun için de en yüksek kazanım veren özellik belirlenmelidir.

10 adet eğitim örneği için değerler şu şekilde bölünmektedir.

* 6 adet Sinema
* 2 adet Tenis
* 1 adet Ev
* 1 adet Alışveriş

Başlangıç için bu değerler üzerinden Entropi değeri hesaplanmalıdır.

$E(S) = -((6/10)*\log_2(6/10) + (2/10)*\log_2(2/10) + (1/10)*\log_2(1/10) + (1/10)*\log_2(1/10))$

E(S) = 1.571 (bu değer başlangıç entropi değeri olarak Information Gain'i hesaplamak için kenarda tutulacak.)

Tek tek tüm özelliklerin kazanım değerlerleri hesaplanarak en yüksek kazanım değerine sahip olan özellik kök düğümü olarak seçilir:

Gain(S,Hava Durumu) = ?

Güneşli = 3 (1 Sinema + 2 Tenis)

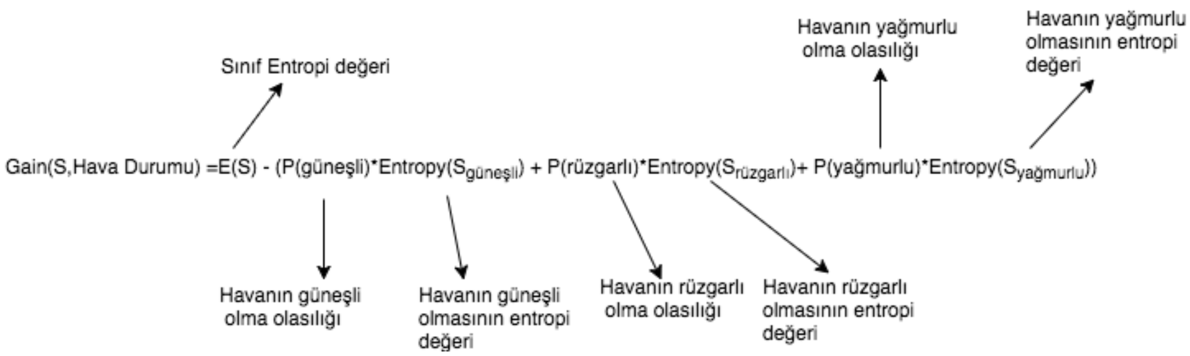
Rüzgarlı = 4 (3 Sinema + 1 Alışveriş)

Yağmurlu = 3 (2 Sinema + 1 Ev)

$Entropy(S_{güneşli}) = - (1/3)*\log_2(1/3) - (2/3)*\log_2(2/3) = 0.918$

$Entropy(S_{rüzgarlı}) = - (3/4)*\log_2(3/4) - (1/4)*\log_2(1/4) = 0.811$

$Entropy(S_{yağmurlu}) = - (2/3)*\log_2(2/3) - (1/3)*\log_2(1/3) = 0.918$



Gain(S, Hava Durumu) = 1.571 - (((1+2)/10)*0.918 + ((3+1)/10)*0.811 + ((2+1)/10)*0.918)
Gain(S, Hava Durumu) = 0.70

Gain(S, Ebeveyn) = ?

Evet = 5 (5 adet Sinema)

Hayır = 5 (2 adet Tenis + 1 adet Sinema + 1 adet Alışveriş + 1 adet Ev)

Entropy(S_{evet}) = - (5/5)*$\log_2$(5/5) = 0

Entropy($S_{hayır}$) = -(2/5)*$\log_2$(2/5) -3*(1/5)*$\log_2$(1/5) = 1.922

Gain(S, Ebeveyn) = Entropy(S) - (P(evet)*Entropy(S_{evet}) + P(hayır)*Entropy($S_{hayır}$))

Gain(S, Ebeveyn) = 1.571 - ((5/10)*Entropy(S_{evet}) + (5/10)*Entropy($S_{hayır}$))

Gain(S, Ebeveyn) = 0.61

Gain(S, Para)=?

Zengin = 7 (3 Sinema + 2 Tenis + 1 Alışveriş + 1 Ev)

Fakir = 3 (3 Sinema)

Entropy(S_{zengin}) = 1.842

Entropy(S_{fakir})=0

Gain(S, Para) = Entropy(S) - (P(zengin)*Entropy(S_{zengin}) + P(fakir)*Entropy(S_{fakir}))

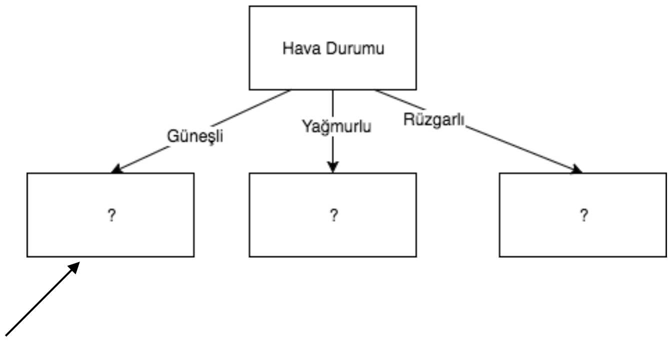
Gain(S, Para) = 0.2816

Son durumda tüm kazanım değerleri alt alta sıralanıp içlerinden en yüksek kazanım değerine sahip olan özellik kök düğüm olarak seçilir:

Gain(S, Hava Durumu) = 0.70

Gain(S, Ebeveyn) = 0.61

Gain(S, Para) = 0.2816



Hafta sonu	Hava	Ebeveyn	Para	Karar (Sınıf)
H1	Güneşli	Evet	Zengin	Sinema
H2	Güneşli	Hayır	Zengin	Tenis
H10	Güneşli	Hayır	Zengin	Tenis

Yukarıdaki örnekte görüldüğü gibi kök seçildikten sonra özelliğe ait tekil değerlerin her biri yaprak olarak belirlenmiş ve bu kez veri seti bu tekil değerler ile özelleştirilmiştir. Yani yukarıdaki tablo hava durumunun "Güneşli" olması durumunda diğer özelliklerin alabileceği değerleri gösteren ayrı bir veri setine dönüştürülmüştür. Böylelikle hava durumu güneşli olduğunda bir alt yaprağın hangi özelliğe ait olacağı bu tabloya göre aşağıdaki gibi bulunacaktır:

* 1 adet Sinema

* 2 adet Tenis

$\text{Entropy}(S_{\text{güneşli}}) = -(1/3)*\log_2(1/3) - (2/3)*\log_2(2/3) = 0.918$

$\text{Gain}(S_{\text{güneşli}}, \text{Ebeveyn}) = ?$

$\text{Gain}(S_{\text{güneşli}}, \text{Ebeveyn}) = \text{Entropy}(S_{\text{güneşli}}) - (P(\text{evet} \mid S_{\text{güneşli}})*\text{Entropy}(S_{\text{evet}}) + P(\text{hayır} \mid S_{\text{güneşli}})*\text{Entropy}(S_{\text{hayır}}))$

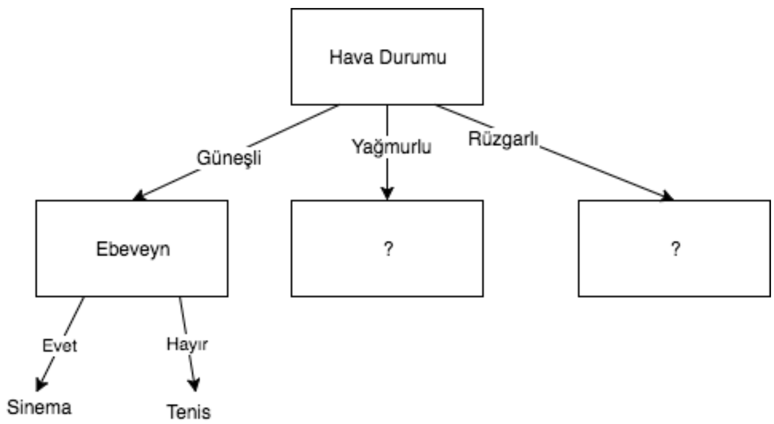
$\text{Entropy}(S_{\text{evet}}) = -(1/3)*\log_2(0) - 0 = 0$

$\text{Entropy}(S_{\text{hayır}}) = -(2/3)*\log_2(0) - 0 = 0$

$\mathbf{Gain}(\mathbf{S}_{\textbf{güneşli}}, \mathbf{Ebeveyn}) = 0.928 - ((1/3)*0 + (2/3)*0) = \mathbf{0.928}$

$\mathbf{Gain}(\mathbf{S}_{\textbf{güneşli}}, \mathbf{Para}) = 0.918 - ((3/3)*0.918 + (0/3)*0) = \mathbf{0}$

Ebeveyn özelliğinin kazanım değeri daha fazla olduğu için Güneşli durumunun yaprağı Ebeveyn olacaktır. Güneşli durumu için oluşturulan veri setine göre de eğer Evet ise Sinema kararı, hayır ise de Tenis kararı çıkacaktır:



Rüzgarlı ve Yağmurlu durumları için de aynı işlemler yapılarak, son durumda karar ağacı tamamlanacaktır.

Overfitting

- Öğrenmenin ezberlemeye dönüşmesidir.
- Başarım grafiğinde belli bir andan sonra başarımın yükselmesinin durup, düşmeye başlamasıdır.
- Eğitim verisi için çok başarılı sonuç verirken, test verisi için başarımın düşük olması overfitting örneğidir.

- Elimizdeki örnekler içerisinde parazitli bir örnek varsa, overfitting'e sebep olabiliyor.
- Ağaç çok büyüdüğünde neredeyse her eğitim verisi için bir sonuç üretilmiş oluyor ve bu da overfitting'e neden oluyor.

Overfitting'den kaçınmak için ağacı ya büyürken ya da ağaç oluştuktan sonra budama yoluna gitmek gerekiyor.

Koşullu Olasılık

Koşullu olasılık A durumu olduğunda B durumunun oluşma olasılığını hesaplamamı sağlar ve bu sayede konuyu spesifik hale getirmemi sağlamış oluyor. Gösterimi P(B|A) şeklindedir.

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

Bayes Kuralı

Bayes kuralı bir rassal değişken için olasılık dağılımı içinde koşullu olasılıklar ile marjinal olasılıklar arasındaki ilişkiyi gösterir. Formulü aşağıdaki gibidir:

likelihood

class attribute

$$P(B|A) = \frac{P(A|B) * P(B)}{P(A)}$$

prior probability

posterior probability

evidence

Not: Terimlerin her yerde ingilizceleri görüleceği için açıklamalarında ingilizceleri kullanılmıştır.

Maximum A Posteriori (MAP)

Bilinmeyen bir kararın tahmini için kullanılır. Tüm verilen veri için tüm hipotezler arasında hangi sınıfın olasılığı daha yüksekse o sınıf seçilir.

$$h_{MAP} = argmax_{h \in H} P(h|D) = \frac{P(D|h) * P(h)}{P(D)} = P(D|h) * P(h)$$

hepsinde sabit olduğu için ihmal ediliyor/hesaplanmıyor

Örnek 3:

P(sınıf=+|kanser)=0.98 (gerçekte kanser olup, test sonucu da kanser(+) çıkan hasta oranı)

P(sınıf=-|~~kanser~~)=0.97 (gerçekte kanser olmayan ve test sonucu kanser değil(-) çıkan hasta oranı)

P(kanser) = 0.008 (kanser hasta oranı)

Yukarıda verilen değerlere göre test sonucu pozitif yani kanser çıkan bir hasta gerçekte kanser sınıftan mıdır değil midir?

P(?|sınıf=+)=?

Çözüm:

Bir olasılığın toplamı her zaman 1 olmalıdır bu yüzden;

P(~~kanser~~) = 1 - P(kanser) => P(~~kanser~~) = 1- 0.008 = 0.992

$$P(kanser|sınıf = +) = \frac{P(sınıf = +|kanser) * P(kanser)}{P(sınıf = +)}$$

P(kanser|sınıf=+) = 0.98 * 0.008 /P(sınıf=+) = **0.008**

$$P(\cancel{kanser}|sınıf = +) = \frac{P(sınıf = +|\cancel{kanser}) * P(\cancel{kanser})}{P(sınıf = +)}$$

P(~~kanser~~|sınıf=+) = 0.03*0.992/P(sınıf=+) = **0.02976**

sonuç: ~~kanser~~

Naive Bayes Teoremi

Naive Bayes Teoremi tüm özellikleri birbirlerinden bağımsız kabul eder. Metin sınıflandırma başarısı oldukça yüksektir.

$$V_{MAP} = argmax_{v_j \in V} P(v_j) * \prod_{i=1}^{n} P(x_i|v_j)$$

Örnek 4:

Aşağıda verilen tablodaki veri setine göre Dergi Promosyonuna Evet, Saat Promosyonuna Evet, Hayat Sigortasına Hayır, Kredi Kartı Sigortasına Hayır cevabını vermiş bir kişinin cinsiyeti nedir?

Dergi Promosyonu	Saat Promosyonu	Hayat Sigortası	Kredi Kartı Sigortası	Cinsiyet
Evet	Hayır	Hayır	Hayır	Erkek
Evet	Evet	Evet	Evet	Kadın
Hayır	Hayır	Hayır	Hayır	Erkek
Evet	Evet	Evet	Evet	Erkek
Evet	Hayır	Evet	Hayır	Kadın
Hayır	Hayır	Hayır	Hayır	Kadın
Evet	Evet	Evet	Evet	Erkek
Hayır	Hayır	Hayır	Hayır	Erkek
Evet	Hayır	Hayır	Hayır	Erkek
Evet	Evet	Evet	Hayır	Kadın

Çözüm:

Soruyu kısaltırsak eğer; (Evet, Evet, Hayır, Hayır) -> Cinsiyet=?

Kontrol edilmesi gereken 2 hipotez var elimizde kadın mı yoksa erkek mi? Bunun için her ikisi için de sonucu hesaplayıp hangisinin değeri daha büyükse o cinsiyettedir cevabını vereceğiz.

Hipotez 1:

$$P(cins = erkek|X) = \frac{P(X|cins = erkek) * P(cins = erkek)}{P(X)}$$

P(X|cins=erkek)=? cevabı için tüm olasılıklar ayrı ayrı bulunarak çarpılacak.

P(dergi_prom=evet|cins=erkek) (cinsiyeti erkek olup dergi promosyonuna evet deme olasılığı) = 4/6 (cinsiyeti erkek olup evet diyenler/cinsiyeti erkek olanların toplam sayısı)

P(saat_prom=evet|cins=erkek) = 2/6

P(hayat_sig=hayır|cins=erkek) = 4/6

P(kredi_kart=hayır|cins=erkek)=4/6

P(X|cins=erkek) = P(dergi_prom=evet|cins=erkek)*P(saat_prom=evet|cins=erkek)*P(kredi_kart=hayır|cins=erkek)*P(hayat_sig=hayır|cins=erkek)

P(X|cins=erkek) = (4/6) *(2/6)*(4/6)*(4/6) =0.0988

P(cins=erkek) = 0.6

P(cins=erkek|X) = 0.0988*0.6 = 0.0593

<u>Hipotez 2:</u>

$$P(cins = kadın|X) = \frac{P(X|cins = kadın) * P(cins = kadın)}{P(X)}$$

P(X|cins=kadın) = (3/4)*(2/4)*(1/4)*(3/4) =0.07

P(cins=kadın)=0.4

P(cins=kadın|X)= 0.0281

0.0593 > 0.0281 olduğu için Bayes sınıflandırıcı cinsiyetin erkek olma olasılığının daha yüksek olduğunu söylüyor.

Not: Örneğin elektronik posta sınıflandırmada her elektronik posta içerisindeki kelimeler üzerinden naive bayes ile bir sınıflandırma yapıp postanın spam mi, gerçek posta mı olduğuna karar veriyorsak, eğitim seti içerisinde bulunmayan bir kelime ile karşılaştığımızda karar aşamasında o değerin eğitim setindeki sayısı 0 olduğu için tüm sonucu çarpımdan ötürü 0'a çekecektir. Bu durumda **Laplacian(1-up) Smoothing** kullanılabilir:

Vj sınıfına ait olan xi kelimelerinin sayısı

$$P(x_i|v_j) = \frac{n_i + 1}{n + |vocab|}$$

postadaki sınıfı olan kelimelerin sayısı

Sözlükteki kelimelerin sayısı

Örnek 5:

Class A: "The cat crabs the crolls off the stairs"

Class B: "It is raining cats and dogs"

Yukarıdaki sınıflara göre X="cats eat mice and dogs bunny bones" metnini sınıflandırın.

Sözlük={cats, crab, croll, stair, rain, dog}

$P(V_j)$ = 1/2(2 adet sınıf olduğu için ya A ya B) |Sözlük|=6 n_A=4 n_B=3

X metninde cat ve dog kelimeleri sadece sözlükte bulunduğu için onlar üzerinden hareket edeceğiz:

$P(x_i|v_j)=(n_{ij} + 1)/(n_j + 6)$

1. Hipotez:

P(A|X) = P(A)*P(cat|A)*P(dog|A) = 1/2 * (1+1)/(4+6) * (0+1)/4+6 = **0.01**

2. Hipotez:

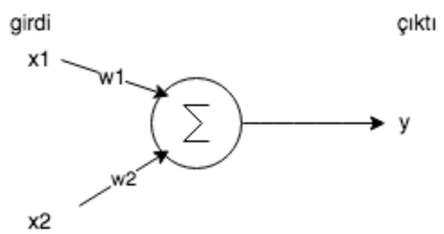
P(B|X) = P(B)*P(cat|B)*P(dog|B) = 1/2 * (1+1)/(3+6) * (1+1)/(3+6) = **0.024**

0.024>0.01 olduğu için X dökümanı B sınıfına aittir.

Artificial Neural Network

Neural Network(NN)	Artificial Neural Network(ANN)
soma, nöron	nöron
dendrit	girdi
akson	çıktı
sinaps	ağırlık

Single Layer Perceptron

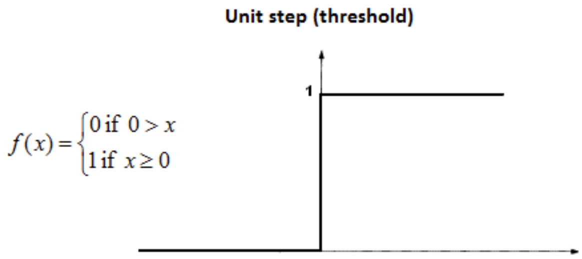


$$y=\sum_{i=1}^{n} x_i \cdot w_i - \theta$$

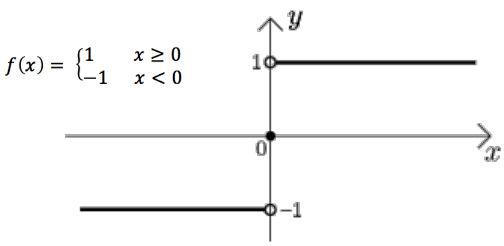
$$y = x1*w1 + x2*w2 - \theta$$

Tek katmanlı perceptronda x'ler girdileri w'ler ağırlıkları ve θ threshold'u ifade eder.

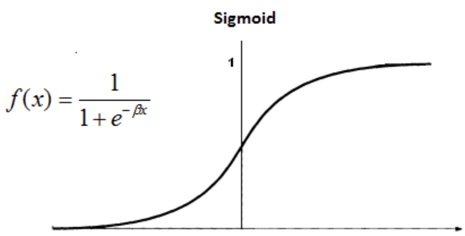
1. Step Function



2. Sign Function



3. Sigmoid Function



Perceptron bu aşamada linear discriminant işlemidir. Sigmoid fonksiyonu değerleri [0,1] aralığına çeker böylece negatif değerler 0'a yaklaşır ve pozitif değerler de 1'e yakın olur ve negatif değerlerle ya da çok büyük sayılarla işlem yapmaya gerek kalmaz.

Training:

1. Girişler ve bu girişler için istenen çıkış değerlerine ait eğitim örnekleri sisteme verilir.

D:<x,y>

2. Bütün aralık değerleri küçük, rastgele (genellikle[-0.5,0.5]) değerlerle ilgilendirilir.
3. Bir eğitim örneği için çıkış değeri hesaplanarak istenilen çıkışa göre hata hesaplanır.

istenilen çıkış

hesaplanan çıkış

hata = $e_t = d_t - y_t$

$Learning\ Rule = \Delta_{w_p} = \mu * (d-y)x_i$ değerlere göre;

öğrenme katsayısı

girdi

$$d = y \rightarrow \Delta_{w_i} = 0$$
$$d > y \rightarrow \Delta_{w_i} > 0$$
$$d < y \rightarrow \Delta_{w_i} < 0$$

$$W_i(t+1) = W_i(t) + \Delta_{w_i}$$

Öğrenme katsayısı; bizim adım büyüklüğümüzü belirtir. Başlangıçta genellikle 1'dir. Gerçekten uzakta olduğunda için büyük adım atmamız gerekir. Genelde 0.6'ya inmek iş görüyor.

$0<\mu<1$

4. Hata ağırlıklar değiştirilerek düzeltilir.
5. Yeni bir örnek ile 3. adımdan devam edilir.

Örnek: OR Operatörünü Single Layer Perceptron ile gösterimi ve step fonksiyonuna göre hesaplanması.

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

$\theta = 0.02$

μ=0.1

w1=0.3

w2=-0.1

1.epoch:

$$y = x1*w1 + x2*w2 - \theta$$

y1 = 0.3*0 + (-0.1)*0 - 0.02 = -0.02   -0.02 <0 olduğu için y1 = 0 diyoruz

y2 = 0.3*0 + (-0.1)*1 - 0.02 = -0.12   -0.12<0 olduğu için y2 = 0 diyoruz

y3 = 0.3*1 + (-0.1)*0 - 0.02 = 0.28   0.28>0 olduğu için y3 = 1 olur diyoruz

y4 = 0.3*1 + (-0.1)*1 - 0.02 = 0.18   0.18>0 olduğu için y4 = 1 olur diyoruz.

Fakat y3'ün 0 çıkması y4'ün de 1 çıkması gerekirdi bu yüzden ağırlıkların değişmesi gerekir. Onun hesabını da aşağıdaki şekilde yapabiliriz. y1 ve y2 doğru olduğu için onlar için aşağıdaki hesabı yapsak da sonuç 0 çıkacak

$$Learning\ Rule = \Delta_{w_p} = \mu * (d - y)x_i$$

$\Delta w1_1 = 0.1 * (0\text{-}0) *0 = 0$

x1 için ağırlık hesapladığım için x1in kendisi

tabloda gördüğüm beklenen sonuç

y1de bulduğum sonuc

$\Delta w2_1 = 0.1 * (0\text{-}0)*0 = 0$

x2 inputu gerisi üstteki ile aynı

$\Delta w1_3 = 0$; $\Delta w2_3 = 0$; $\Delta w1_4 = 0$; $\Delta w2_4 = 0$

$\Delta w1_3 = 0.1 * (1 - 0)*0 = 0$; $\Delta w2_3 = 0.1*(1 - 0)*1 = 0.1$

$w1_3' = w1_3 + \Delta w1_3$ yeni w1 ağırlık değeri $w1_3' = 0.3 + 0 = 0.3$

$w2_3' = w2_3 + \Delta w2_3$ yeni w2 ağırlık değeri $w2_3' = (-0.1) + 0.1 = 0$

2.epoch:

Yeniden çıkış hesaplamaları yapılır;

y1 = 0.3*0 + 0*0 -0.02 = -0.02 <0 olduğu için y1 = 0

y2 = 0.3*0 + 0*1 -0.02 = -0.02 <0 olduğu için y2 = 0

y3 = 0.3*1 + 0*0 -0.02 = 0.28 >0 olduğu için y3 = 1

y4 = 0.3*1 + 0*1 -0.02 = 0.28 > 0 olduğu için y4 = 1

$\Delta w1_1 = 0$; $\Delta w2_1 = 0$; $\Delta w1_3 = 0$; $\Delta w2_3 = 0$; $\Delta w1_4 = 0$; $\Delta w2_4 = 0$ doğru sonuçları veriyor.

$\Delta w1_3 = 0.1 * (1\text{-}0)*0 = 0$ bu yüzden $w1_3' = 0 + 0.3 = 0.3$

$\Delta w2_3 = 0.1 * (1-0) * 1 = 0.1$ bu yüzden $w2_4' = 0.1 + 0 = 0.1$

3.epoch:

Doğru ağırlık değerlerini bulana kadar bu hesaplamalar gerçekleştirileceği için işlemleri yeniden yapıyoruz:

$y1 = 0.3*0 + 0.1*0 - 0.02 = -0.02 < 0$ olduğu için $y1 = 0$

$y2 = 0.3*0 + 0.1*1 - 0.02 = 0.08 > 0$ olduğu için $y2 = 1$

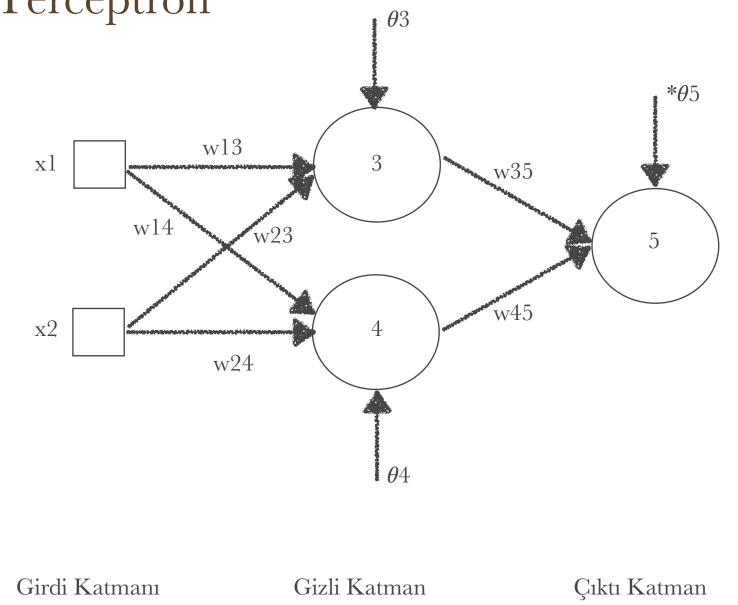
$y3 = 0.3*1 + 0.1*0 - 0.02 = 0.28 > 0$ olduğu için $y3 = 1$

$y4 = 0.3*1 + 0.1*1 - 0.02 = -0.38 > 0$ olduğu için $y4 = 1$

$\Delta w1_1 = 0$; $\Delta w2_1 = 0$; $\Delta w1_2 = 0$; $\Delta w2_2 = 0$;

$\Delta w1_3 = 0$; $\Delta w2_3 = 0$; $\Delta w1_4 = 0$; $\Delta w2_4 = 0$; tüm örnekler için doğru ağırlıklar belirlenmiş oldu. Bundan sonra gelecek test verileri için bu ağırlık değerleri ile doğru sonuçlar hesaplanacaktır.

Multi Layer Perceptron



Back Propagation

Her katmanda hesaplanan sonuçlar bir sonraki katman için giriş olarak kabul edilir. Hata hesaplanırken geriye doğru işlemler ile ağırlıklar yeniden düzenlenir.

Formüller:

t. örnek için hata

y ve d arasındaki Euclidean distance hesabı

$$E = \frac{1}{2} * \sum_{i=1}^{t} (y_t - d_t)^2$$

$$Gradient: \nabla E = \left(\frac{\partial E}{\partial W_1}, \frac{\partial E}{\partial W_2}, \ldots, \frac{\partial E}{\partial W_n}\right)$$

$$Eğitim\ Kuralı: \nabla W = \mu \frac{\nabla E}{\partial W}; \qquad \frac{\partial E}{\partial W_i} = \frac{\partial}{\partial W_i} * \frac{1}{2} * \sum (y_i - x_i)^2$$

$$Çıkış\ Ünitesi\ İçin: \delta_k = y_k(1 - y_k)(d_k - y_k); \qquad \Delta w_{jk} = \mu * y_j * \delta_k$$

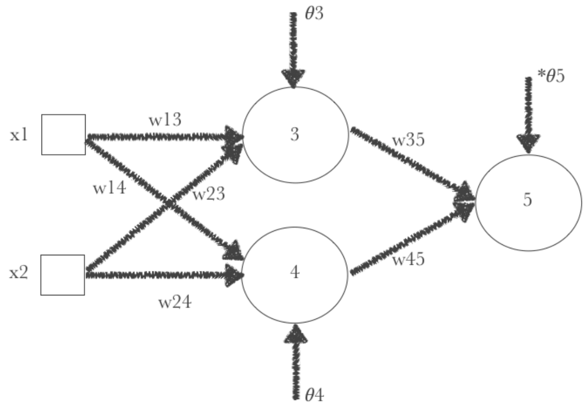
$$Gizli\ Ünite\ İçin: \delta_j = y_j(1 - y_j) \sum_{i \in çıkışlar} W_{ji} * \delta_i; \qquad \Delta w_{jk} = \mu * x_j * \delta_k$$

XOR problemini Single Layer Perceptron ile çözmek mümkün değil.

x1	x2	y	Sonuç
0	0	0	y=0
0	1	1	y=x2*w2>0
1	0	1	y=x1*w1>0
1	1	0	y=x1*w1+x2*w2<0

y = x2*w2 ve y=x1*w1 ‘in 0’dan büyük olması ve aynı zamanda y=x1*w1 + x2*w2 işleminin sonucunun 0’dan küçük olması mümkün değil. Eşitsizlikleri aynı anda sağlayamadığımız için Multi Layer Perceptron ile çözümü önerilmiştir.

Örnek:



w13 = 0.5	$\theta 3$ = 0.8
w23 = 0.4	$\theta 4$ = -0.1
w35 = -1.2	$\theta 5$ = 0.3
w14 = 0.9	μ = 0.1
w24 = 1	
w45 = 1.1	

XOR işlemi için x1=x2=1 için y=0 sonucunu sağlamalı. Değerler sürekli olduğu için sigmoid kullanılıyor.

y3 = sigmoid(x1*w13+x2*w23-θ3)=1/(1+$e^{-(0.5*1\ +\ 1*0.4\ -1*0.8)}$) = 0.525

y4 =sigmoid(x1*w14+x2*w24-θ4)= 0.88

y5 = sigmoid(y3*w35+y4*w45-θ3) = 1/(1+$e^{-(0.525*(-1.2)\ +0.88*1.1-0.3)}$) = 0.5097 ->gerçek çıkış

Hata = Ɛ = (0 - 0.5097) = -0.5097

Output Layer: ς5 = y5*(1-y5)* Ɛ = 0.5097*(1-0.5097)*(-0.5097) ≅ -0.12

Δw35 = ς5*y3*μ = -0.12 * 0.525*0.1 = -0.006

Δw45 = ς5*y4*μ = -0.12 * 0.88*0.1 = -0.01056

Hidden Layer:

ς3 = y3*(1-y3)* ς5*w35 => Δw23=μ*x2*ς3 ; Δw13=μ*x1*ς3

ς4 = y4*(1-y4)* ς5*w45 => Δw24=μ*x2*ς4 ; Δw14=μ*x1*ς4

NOT:

şeklinde tüm ağırlıkları doğru sonuçları elde edilene kadar tekrar tekrar hesaplanıyor. Gizli katman sayısı ve gizli katmanda kaç nöron olacağı tamamiyle tasarlayan kişiye bağlıdır. Klasik yapay nöron için gizli katman sayısı maksimum 3'tür.

Ağırlık Güncellemesi

1. Incremental Mode

Her eğitim aşamasından sonra : $w_{ij} = w_{ij} + \Delta w_{ij}$

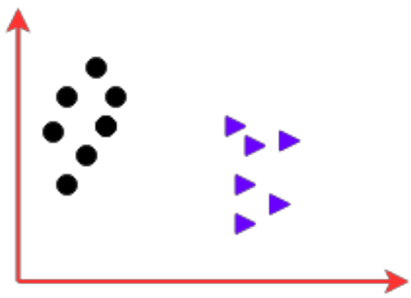
2. Batch Mode

$$sumw_{ij} = \sum_{i=1}^{t} \Delta w_{ij} : her\ instance'\text{tan}\ sonra$$
$$w_{ij} = w_{ij} + sumw_{ij} : her\ epoch'\text{tan}\ sonra$$

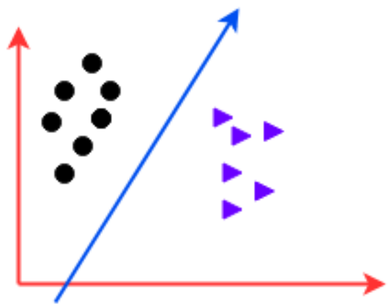
Momentum terimi : t ve (t-1) anındaki değişimleri de birbirine bağlıyor. Daha çok hassasiyet için kullanılıyor.

$$\Delta w_{ij}(t) = \beta * \Delta w_{ij}(t-1) + \mu * y_i(t) * \delta_j(t); \qquad 0 \leq \beta \leq 1$$

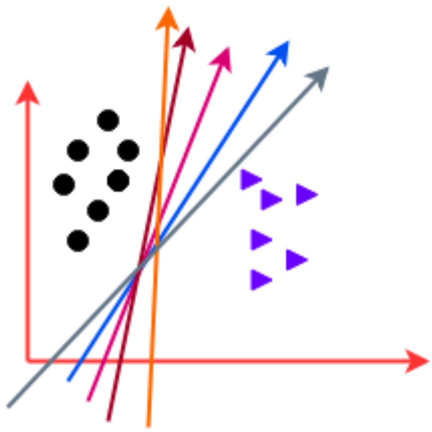
Support Vector Machine (SVM)



- Yukarıdaki örneğe baktığınızda 2 sınıfa ait örnekleri bir doğru çizerek bölseniz nasıl bölersiniz? SVM de tam olarak bunu yapıyor, aşağıdaki örnekteki gibi bir doğruyla 2 sınıfı birbirinden ayırıyor.



- 2 sınıfı birbirinden ayıran doğruya hyperplane (sınıflara bölen çok boyutlu uzay) adı verilir. Buna aynı zamanda decision boundary (karar sınırı) adı da verilmektedir.
- Doğrunun soluna düşen örnekler siyah daire sınıfından sağına düşen örneklerse mavi daire sınıfında kabul edilir.



- Yukarıdaki örnekte görülen doğrulardan herhangi birisi benim hyperplane'im olabilir. Fakat doğrunun nereden geçtiğine göre sınıflandırma sonucum da değişir. Bu noktada önemli olan sadece eğitim hatasının düşük olması değildir, test hatasının da düşük olması gerekir.
-