1- Sınıflandırma ve Regresyon Algoritmaları
   a. Karar Ağaçları

Entropiye Dayalı Algoritmalar	**Sınıflandırma ve Regresyon Ağaçları (CART)**	**İstatistiğe Dayalı Algoritmalar**
1-ID3	1-Twoing	1-CHAID(Chi-Squared Automatic Interaction Detector)
2-C4.5, C5	2-Gini	2-QUEST(Quick, Unbiased, Efficent, Statistical Tree)
	3-Random Forest	

   b. Bellek Tabanlı Sınıflandırma Algoritmaları
      K-En Yakın Komşu
   c. Yapay Sinir Ağları
   d. Bayesçi Ağlar (Bayesian Networks) ve Bayes Sınıflandırıcı

      Naif Bayes Sınıflandırıcı

   e. Regresyon Modelleri

      1-Lineer(Doğrusal) Regresyon

      2-Lojistik Regresyon

      3-En Küçük Kareler Yöntemi (OLS)

   f. Destek Vektör Makinesi

2- Kümeleme Algoritmaları

      1-K-Means

      2-Kohonen Ağlar(Kendini Düzenleyen Ağlar)

      3-Ward'ın Minimum Varyans Yöntemi

      4-Hiyerarşik Kümeleme

3- Birliktelik Kuralları

      Apriori Algoritması

4- Modellerin birleştirilmesi (Ensemble Learning)

      Adaboost

5- BOYUT İNGİRGEME

      Projeksiyon

      Temel Bileşenler Analizi

1- SINIFLANDIRMA VE REGRESYON ALGORİTMALARI

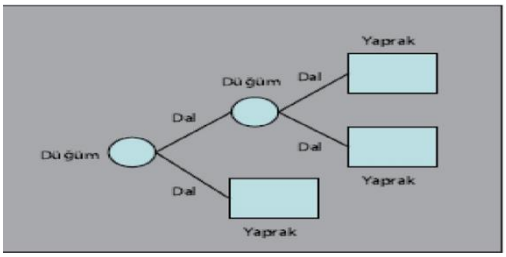
a. Karar Ağaçları

Karar ağaçları, kurgulanmasının, yorumlanmasının ve veri tabanları ile entegrasyonun kolaylığı nedeniyle en yaygın kullanılan öngörü yöntemlerinden/sınıflandırma tekniklerinden biridir.

Karar ağaçlarının hedefi bağımlı değişkendeki farklılıkları maksimize edecek şekilde veriyi sıralı bir biçimde parçalarına (farklı gruplara) ayırmaktır.

Karar Ağacının Yapısı

Karar düğümü: Karar düğümü, gerçekleştirilecek testi belirtir. Her düğüm testi gösterir. Test sonucunda ağacın dalları oluşur. Dalları oluştururken veri kaybı yaşanmaması için verilerin tümünü kapsayacak sayıda farklı dal oluşturulmalıdır.



Dal: Testin sonucunu gösterir. Elde edilen her dal ile tanımlanacak sınıfın belirlenmesi amaçlanır. Ancak dalın sonucunda sınıflandırma tamamlanamıyorsa tekrar bir karar düğümü oluşur. Karar düğümünden elde edilen dalların sonucunda sınıflandırmanın tamamlanıp tamamlanmadığı tekrar kontrol edilerek devam edilir.

Yaprak: Dalın sonucunda bir sınıflandırma elde edilebiliyorsa, yaprak elde edilmiş olur. Yaprak, verileri kullanarak elde edilmek istenen sınıflandırmanın, sınıflarından birini tanımlar.

Karar ağacı işlemi kök düğümünden başlar ve yukarıdan aşağıya doğru yaprağa ulasana dek ardışık düğümleri takip ederek gerçekleşir.

Karar Ağaçlarında Ezberleme, Aşırı Öğrenme(Overfitting)

Öğrenme kümesindeki örneklerin azlığı veya gürültülü olması ezberlemeye neden olabilir. Eğitim verilerini en iyi sınıflandıracak şekle kadar her dalını derinleştirir.Bu mantıklı bir stratejiyken, veriler arasında gürültü varsa zorluklara sebep olabilir.Gürültü varken, her yaprak saf (pure) olana dek bölünmeye izin vermek, veriyi ezberleyen (overfitting) çok büyük bir karar ağacı oluşmasına neden olabilir.

Aşırı öğrenmeyi engellemek için:

Verideki gürültüyü azaltmak için budama yaparız.Düşük öneme sahip özellikleri kullanan dalları kaldırmayı içerir. Böylece ağaçların karmaşıklığı düşer ve aşırı uyum azalarak tahmin gücü artar. Budama kök veya yapraklardan başlayabilir.

Bir ağacın performansı, budama ile daha da arttırılabilir . Düşük öneme sahip özellikleri kullanan dalları kaldırmayı içerir . Bu şekilde ağaçların karmaşıklığını düşürüyoruz ve böylece aşırı uyumu azaltarak tahmini gücünü artırabiliriz. Budama işleminin en basit yöntemi yapraklarda başlar ve bu yaprağın en popüler sınıfa sahip her bir düğümünü kaldırır; bu değişiklik doğruluk bozulmazsa tutulur. Buna hata düzeltme denir.

İki çeşit budama yöntemi vardır:

1. Erken budama(Pre-prunning): Ağacın büyümesini erken durduran bir yaklaşımdır.
2. Geç budama(Post-pruning): Ağacın tamamlanmasının ardından işlem yapılır.Daha doğru bir çözümdür.

En İyi Bölen Nitelik Nasıl Belirlenir? (Dallanma Kriteri)

Uyum iyiliği testleri kullanılarak (Goodness function) hangi algoritmanın veri kümesini daha iyi sınıflara ayırdığına karar verilebilir.

Örneğin ID3 ve C 4.5 algoritmaları için bilgi kazancı(information gain)kullanılabilir. Bilgi kazancı entropiyle alakalı bir yaklaşımdır. Aslında kısaca hangi nitelik hedef değişkeni daha çok açıkladığını araştırır.

Gini index de kullanılabilir. Gini index kısaca dağılımın eşitsizlik ölçüsüdür, dallanma kriteri olarak kullanılırken gini indexin küçük olması istenir.

	C5.0	**CART**	**QUEST**	**CHAID**
Bağımsız Değişkenler: Kategorik	Evet	Evet	Evet	Evet
Bağımsız Değişkenler: Sıralı	Evet	Evet	Evet	Evet
Bağımsız Değişkenler: Sürekli	Evet	Evet	Evet	Evet*
Bağımlı Değişken: Kategorik	Evet	Evet	Evet	Evet
Bağımlı Değişken: Sürekli	Hayır	Evet	Hayır	Evet
Bölünme Tipi	Çoklu	İkili	İkili	Çoklu

* CHAID sürekli bağımsız değişkenleri sıralı değişkenlere dönüştürür.

- **Entropiye Dayalı Algoritmalar: ID3, C 4.5, C5**

Entropi bir sistemdeki belirsizliğin ölçüsüdür.

Eldeki bütün veriler tek bir sınıfa ait olursa, örneğin herkes aynı futbol takımını tutuyorsa, herhangi bir kişiye tuttuğu takımı sorduğumuzda alacağımız yanıt bizi şaşırtmayacaktır. Bu durumda entropi “0” olacaktır.

Karar ağaçları çok boyutlu (özellikli) veriyi belirlenmiş özellik üzerindeki bir şart ile parçalara böler. Her seferinde verinin hangi özelliği üzerinde hangi şarta göre işlem yapacağına karar vermek

çok büyük bir kombinasyonun çözümüyle mümkündür. 5 özellik ve 20 örneğe sahip bir veride 106'dan fazla sayıda farklı karar ağacı oluşturulabilir. Bu sebeple her parçalanmanın metodolojik olması gerekir. Bunun için mutlaka bir bölme niteliği belirlemek gerekir. Bu bölme işlemini de farklı algoritmalar kullanarak belirleyebiliriz.

1- ID3 Algoritması

Nitelikler birbirinden bağımsızdır. Bu algoritmada amaç, ağaç oluşturulurken öğrenme kümesindeki veriler mümkün olduğunca birbirine benzer hale getirilir ve böylece ağaç derinliği minimum seviyede tutulur. Karmaşıklık minimuma indirilir, kazanç maksimum seviyede olur.

Karmaşıklık belirlemek amacıyla entropi kuralları içeren bilgi teorisi kullanılmaktadır. Entropi değer aralığı, sınıf etiket sayısına göre değişiklik göstermektedir.

"n" sınıf etiketine sahip bir veri setindeki entropi değer aralıkları:

$0 < \text{entropi} < \log_2 n$

Bu duruma göre, 2 sınıf etiketine sahip bir veri setinin entropi değer aralıkları 0 ile 1 arasında olacaktır. Entropi değeri 1'e yaklaştıkça belirsizlik(karmaşıklık) artar, 0'a yaklaştıkça belirsizlik azalır. Yani birbirine benzer verilerin bulunduğu bir veri seti 0'a yakın olacaktır.

2- C4.5 ve C5.0 Algoritması

C4.5 ve C5.0 algoritmaları ID3 algoritmasının bazı eksik yönlerini gidermek için geliştirilmiştir. C5.0 algoritması da C4.5'e göre doğruluğu artırmak amacıyla geliştirilmiş bir algoritmadır ve özellikle büyük sayıda verilerden oluşan veri setlerinde kullanılmaktadır. Doğruluğun artırılmasını sağlamak amacıyla boosting algoritmasını da kullanır. C4.5 ve C5.0 algoritmaları da temelde ID3 algoritmasının özellikleri kullanılır. Bu algoritmaların en büyük farkı normalizasyon yapılmasıdır. ID3 ağacı üzerinde entropi hesabı yapılır ve bu değere göre karar noktaları belirlenir. C4.5 ağacında hesaplanan entropi değerleri oran olarak tutulur. Ayrıca ağaç üzerinde erişim sıklıklarına göre alt ağaçlar farklı seviyelere taşınabilir. C4.5 ağacının bir diğer farkı da budama işleminin yapılabilmesidir.

Entropiye Dayalı Algoritmalar
Güçlü Yanları : Yorumlaması kolay, genellikle güçlü tahminler verir. Başarı oranının yüksekliği ve kolay anlaşılır bir grafiksel yorumunun bulunması sebebiyle en çok tercih edilen sınıflandırma tekniğidir. Sürekli ve ayrık nitelik değerleri için kullanılabilir olması da güçlü yanlarındandır.
Zayıf Yanları : Ezberleme (over fitting) eğilimi vardır. Büyük ağaçların yorumlanması güçtür. Sürekli nitelik değerlerini tahmin etmekte çok başarılı değildir. Sınıf sayısı fazla ve öğrenme kümesi örnekleri sayısı az olduğunda model oluşturmada çok başarılı değildir.

- **Sınıflandırma ve Regresyon Ağaçları (CART)**

Bütün aşamalarda her bir grubu kendinden daha homojen grup oluşturacak şekilde alt gruplara ayırmayı hedefleyen bir yapı ile oluşturulmuştur.

CART karar ağacı, her bir karar düğümünden itibaren ağacın iki dala ayrılması ilkesine dayanır. Yani bu tür karar ağaçlarında ikili dallanmalar söz konusudur. CART algoritmasında bir düğümde belirli bir kriter uygulanarak bölünme işlemi gerçekleştirilir. Bunun için önce tüm niteliklerin var olduğu değerler göz önüne alınır ve tüm eşleşmelerden sonra iki bölünme elde edilir. Bu bölünmeler üzerinde seçme işlemi uygulanır.

Güçlü Yanları : Parametreler arasındaki doğrusal olmayan ilişkiler ağaç performansını etkilemez. Yapısı gereği önemli ve önemsiz verileri ayırt edebilir. Kategorik ve numerik değişkenlerle çalışabilir. Gürültülü verilerle çalışabilme avantajına sahiptir. Herhangi bir varsayımı yoktur. Hızlı ve esnektir.
Zayıf Yanları : Sadece ikili duruma göre karar vermesi en belirgin dezavantajıdır. Bunun yanında kullanıcı data setten bazı verileri çıkarması durumunda yanlış bir karar ağacı oluşabilir. Nonparametrik olduğundan test aşamasında verdiği sonucu başka bir test verisinde aynı güçte vermeyebilir.

1. Twoing Algoritması

Twoing algoritmasında eğitim kümesi her adımda iki parçaya ayrılarak bölümleme yapılır. Aday bölünmelerin sağ ve sol kısımlarının her birisi için nitelik değerinin ilgili sütundaki tekrar sayısı alınır. Aday bölünmelerin sağ ve sol kısımlarındaki her bir nitelik değeri için sınıf değerlerinin olma olasılığı hesaplanır.Her bölünme için uygunluk değeri en yüksek olan alınır.

2. Gini Algoritması

Gini algoritmasında nitelik değerleri iki parçaya ayrılarak bölümleme yapılır.

3. Rastgele Orman Algoritması

Bu algoritmada amaç tek bir karar ağacı üretmek yerine her biri farklı eğitim kümelerinde eğitilmiş olan çok sayıda çok değişkenli ağacın kararlarını birleştirmektir. Farklı eğitim kümeleri oluştururken ön yükleme ve rastgele özellik seçimi kullanılır. Çok değişkenli karar ağaçları oluşturulurken CART algoritması kullanılır. Her seviyedeki özniteliği belirlerken önce bütün ağaçlarda hesaplamalar yapılarak nitelik belirlenir, ardından bütün ağaçlardaki nitelikler birleştirilerek en fazla kullanılan öznitelik seçilir. Seçilen nitelik ağaca dahil edilerek diğer seviyelerde aynı işlemler tekrarlanır.

Rasgele orman algoritmasını, daha sonra modellerin birleştirilmesi kısmında da göreceğiz. Buradaki rasgele orman algoritması regresyon modeli gibi düşünülebilir.

- **İstatistiğe Dayalı Algoritmalar**

1- CHAID Algoritması

CHAID algoritması, karar ağaçları algoritmaları içinde sürekli ve kategorik tüm değişken tipleri ile çalışabilmesi nedeni ile yaygın olarak kullanılmaktadır. Bu yöntemde, sürekli tahmin edici değişkenler otomatik olarak analizin amacına uygun olarak kategorize edilmektedir.

CHAID, ki-kare metriği vasıtasıyla ilişki düzeyine göre farklılık rastlanan grupları ayrı ayrı sınıflamakta ve ağacın yaprakları, ikili değil, verideki farklı yapı sayısı kadar dallanmaktadır.

CHAID ve CART arasındaki en büyük fark; CART'da olduğu gibi, CHAID'de bölünmüş değişken ve bölme noktası seçiminin daha az çarpıtılmış olmasıdır. Ağaçlar tahminde kullanıldıklarında bu önemli ölçüde önemsizdir ancak ağaçlar yorumlama için kullanıldığında önemli bir husustur: Algoritmanın bu iki parçasının oldukça karışık olduğu bir ağacın "değişken seçiminde önyargılı" olduğu söylenir. Bu, bölünmüş değişken seçimi birçok olası bölünme ile değişkenleri tercih ettiği anlamına gelir. CART bu anlamda "önyargılı", CHAID o kadar çok değil.

2- QUEST Algoritması

Hızlı, yansız, etkili istatistiksel ağaç (Quick, Unbiased, Efficent, Statistical Tree) olarak bilinir. CART algoritmasında olduğu gibi ikili karar ağaçları oluşturmak üzerine kurulu bir yapısı vardır. Fakat CHAID ve CART' dan farklı olarak değişken seçimi ve ayırma noktası seçimi işlemlerini ayrı ayrı ele almaktadır. En çok bilinen ayrıntılı araştırma metotları CART gibi, ağaç büyüme sürecinde daha fazla bölünmeyi sağlayan fazla kesikli değer ile değişken seçme yoluna gidilmektedir. Bu durum sonuçların genelleştirilmesini azaltarak modelin yanlı olmasına neden olmaktadır. Eğer bağımsız değişken için açıklayıcı değişkenler aynı bilgi vericilikte iseler QUEST bu açıklayıcı değişkenlerden herhangi birini eşit olasılıkla seçecektir. Ayrıca hesaplamalardaki etkinliği açısından da kuvvetli yapıdadır. CART' nın sağladığı bütün avantajlara sahip olmasına rağmen yine de CART gibi ağaç hantal yapıdadır. Bağımlı değişken tek tip nominal türde; bir ya da daha fazla açıklayıcı değişken sınıflayıcı, sıralayıcı ve sürekli türde ölçülmüş olabilir. Kayıp verilerde yerine koyma kuralları uygulanır. QUEST, bağımlı değişkenler kategorik yapıda ise, yansız ağaç önemli ise, büyük ve karmaşık yapıda veri seti varsa ve ağaç tahmininde etkili bir algoritmaya gerek varsa, ağaç ikili bölünme ile sınırlandırılmışsa kullanılır. CART' de olduğu gibi ön olasılıklar kullanılmaktadır. QUEST algoritması şu şekildedir; her ayırma için her açıklayıcı değişken ve bağımlı değişken arasındaki ortaklık Anova F testi ya da Levene testi (sıralayıcı ve sürekli açıklayıcı değişkenler için) ya da Pearson Ki-Kare değeri (sınıflayıcı açıklayıcı değişkenler için) hesaplanarak bulunur. Değişken seçimi için küçük p değerine sahip olan değişken seçilir. Ayırma işleminde ise eğer bağımlı değişken ikiden fazla kategoriye sahip ise iki süper sınıf bulabilmek için iki ortalamalı kümeleme algoritması kullanılır. CART olduğu gibi maksimum-karmaşıklık ölçüsü ile ağaç budanır.

b. Bellek Tabanlı Sınıflandırma Algoritmaları

- **K En Yakın Komşu Algoritması**

Örnekleme yoluyla öğrenmeye dayanır. Bu teknikte tüm örneklemler bir örüntü uzayında saklanır. Algoritma, bilinmeyen bir örneklemin hangi sınıfa dâhil olduğunu belirlemek için örüntü uzayını araştırarak bilinmeyen örnekleme en yakın olan k örneklemi bulur. Yakınlık öklid uzaklığı ile tanımlanır. Daha sonra, bilinmeyen örneklem, k en yakın komşu içinden en çok benzediği sınıfa atanır.

Örneğin k (5 olsun) olarak göstereceğimiz bir sayı seçeriz. Sınıflandırmak istediğimiz yeni bir elemanın hangi sınıfa dahil olacağını belirlemek için bu yeni elemana en çok benzeyen beş elemanın sınıflarına bakarız. En çok eleman hangi sınıfa mensup ise yeni elemanımız da o sınıftandır deriz.

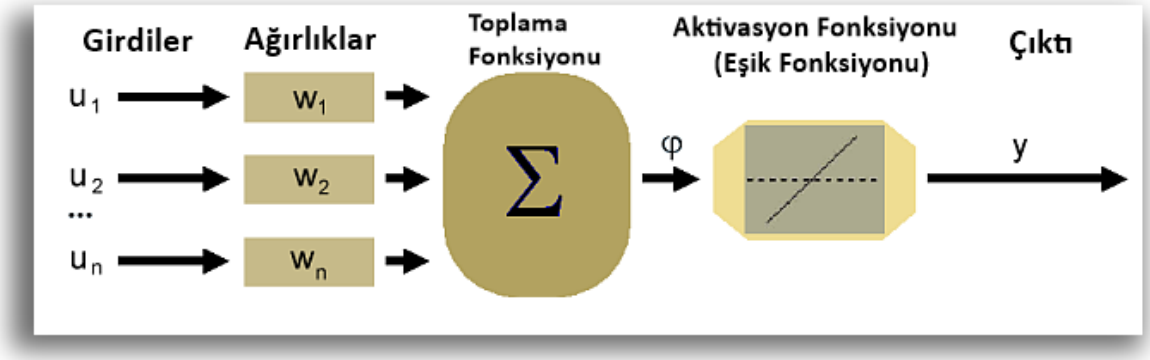
"k" parametresini belirlerken dikkatli olmalıyız."k" için optimum bir belirleme ölçütü yok. Eğer veriniz çok az elemandan oluşuyorsa ve siz çok yüksek bir "k" sayısı belirliyorsanız, her bir parça veriyi temsilden uzaklaşacak ve model testi konusunda sıkıntılar oluşacaktır. "k" için küçük bir değer, düşük önyargıya sahip ancak yüksek varyansa sahip en esnek uyum sağlar. Daha büyük "k" değerleri daha karmaşık karar sınırlarına sahip olacak, bu daha düşük varyans, ancak artan yanlılık anlamına geliyor. Algoritmaları uygularken Cross-validation kullanarak "k" parametresinin oluşturabileceği olası yanlılık vb. durumların önüne geçebiliriz.

Güçlü Yanları : Uygulanması basit Gürültülü verilerine karşı etkili Eğitim kümesindeki veri sayısı fazla ise daha etkindir
Zayıf Yanları : Algoritma başlangıçta K parametresine ihtiyaç duyar En iyi sonucun alınabilmesi için hangi uzaklık ölçümünün uygulanacağı ve hangi özelliklerin alınacağı bilgisi açık değildir

c. Yapay Sinir Ağları (YSA)

YSA, insan beyninin çalışma mekanizmasını taklit ederek beynin öğrenme, hatırlama genelleme yapma yolu ile yeni bilgiler türetebilme gibi temel işlevlerini gerçekleştirmek üzere geliştirilen mantıksal yazılımlardır. YSA biyolojik sinir ağlarını taklit eden sentetik yapılardır.

Yapay sinir ağları yaklaşımının klasik istatistiksel yöntemlere göre avantajı, verilerin dağılımıyla ve değişkenlerle ilgili varsayımlara gereksinim duymamasıdır. Yapay sinir ağları, bazı değişkenlere ait eksik verileri de tolare etme özelliğine sahiptir.



Girdiler: Bir yapay sinir hücresine dış dünyadan gelen bilgilerdir. Bunlar ağın öğrenmesini gereken bilgilerdir. Yapay sinir hücresine dış dünyadan olduğu gibi başka hücrelerden veya kendi kendisinden de bilgiler gelebilir.

Ağırlıklar: Yapay sinir hücresi tarafından alınan girdilerin, sinir hücresi üzerindeki etkisini gösteren uygun katsayılardır. Her bir girdi kendine ait bir ağırlığa sahiptir. Ağırlıkların büyük ya da küçük olması önemli ve ya önemsiz olduğunu göstermez.

Toplama Fonksiyonu: Bu fonksiyon, bir hücreye gelen net girdiyi hesaplar. Burada her gelen girdi değeri, kendi ağırlığı ile çarpılarak toplanır. Literatürde yapılan araştırmalarda toplama fonksiyonu olarak farklı formüllerin kullanıldığı bilinmektedir. Bazı durumlarda toplama işlevi minimum ya da maximum ve ya birkaç normalleştirme algoritması gibi daha karmaşık fonksiyonlar olabilir.

Aktivasyon fonksiyonu (Eşik Fonksiyonu): Bu fonksiyon yapay bir sinir hücresine gelen girdiye uygulandığında, bu girdiye karşılık üretilecek çıktıyı belirler. Aktivasyon fonksiyonunun doğru seçilmesi ağın performansın önemli derecede etkiler. Aktivasyon fonksiyonu genelde tek kutuplu (0 1), çift kutuplu (-1 +1) ve doğrusal olarak seçilebilir. Ağın doğrusal olmayan yapıyı öğrenmesini sağlayan bileşenidir.

- → Lineer Eşik Fonksiyon
- → Rampa Eşik Fonksiyonu
- → Basamak Eşik Fonksiyonu
- → Sigmoid Eşik Fonksiyonu
- → Hiperbolik Tanjant Eşik Fonksiyonu

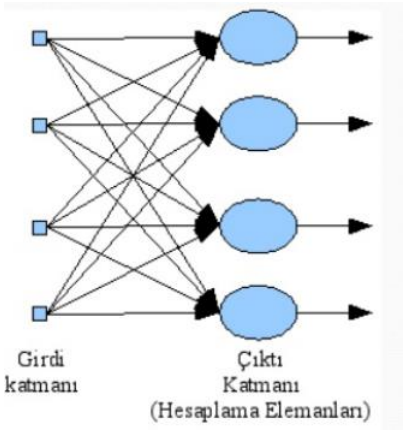
Çıktı: Aktivasyon fonksiyonu tarafından belirlenen çıktı değeridir. Elde edilen çıktı değeri dış dünyaya veya başka sinir hücrelerine gönderilir. Bir sinir hücresinin tek bir çıktısı vardır. Bu çıktı, kendinden sonra gelen birden fazla sinir hücresine girdi olabilir.

Bir bilgi kaynağından öğrenebilme yeteneği YSA'nın en önemli özelliklerinden biridir. Yapay sinir ağlarında bilgi, ağdaki sinirlerin bağlantılarının ağırlıklarında tutulur. Bu nedenle ağırlıkların nasıl belirleneceği önemlidir. Bilgi tüm ağda saklandığı için bir düğümün sahip olduğu ağırlık değeri tek başına bir şey ifade etmez. Tüm ağdaki ağırlıklar optimal değerler almalıdır. Bu ağırlıklara ulaşılabilmesi için yapılan isleme "ağın eğitilmesi" denir. Buna göre bir ağın eğitilebilir olabilmesi için ağırlık değerlerinin belirli bir kural dahilinde dinamik olarak değiştirilebilir olması gerekmektedir.

Kısaca ifade edecek olursak; “Öğrenme işlemi, ağırlıkların en iyi değerinin bulunması” olarak tanımlayabiliriz.

Genel olarak yapay sinir ağları modellerini, ağın yapısına ve öğrenme türüne göre sınıflandırmak mümkündür.

İleri beslemeli (Feed forward): Yapıda bulunan bağlantılar yalnızca daha sonraki katmanlardır. İlk gizli kaman gibi ikinci gizli katman sinirleri de ağırlıklarla önceki katmana tam bağlıdır. Bu sinirlerde girişlerin ve girişlerin ağırlıklarının bir işlevini hesaplayıp sonucu sonraki aşamaya aktarır. Bu işlem çıkış katmanındaki sinirler tarafından da yapıldıktan sonra tamamlanır. Bu ağlar, çok katmanlı ileri beslemeli ağlar olarak da bilinir.

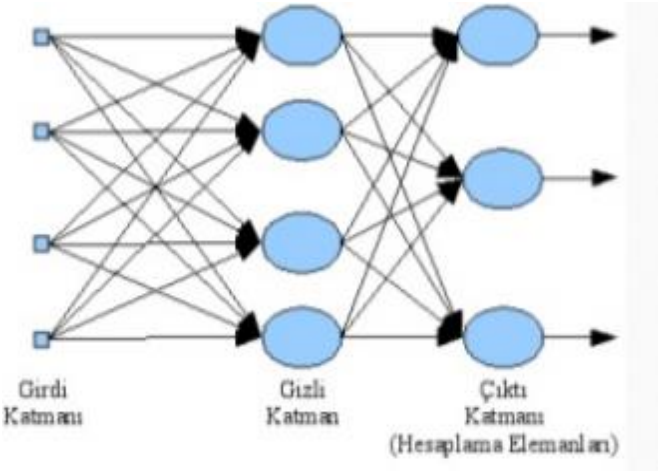


Tek Katmanlı İleri Beslemeli Ağlar

Bu ağ mimarisinde girdiler direk olarak çıkış nöronlarına bağlıdır.

Bağlantılar tek yönlü ve sadece ileri doğrudur.

Sadece çıkış nöronları hesaba katılarak tek katmanlı adını almıştır.

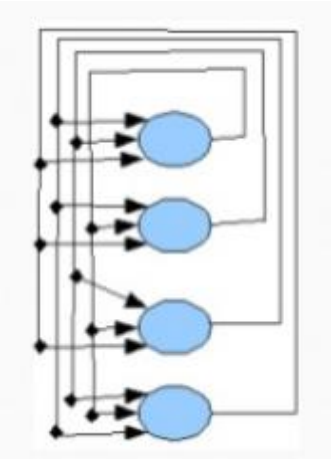


Çok Katmanlı İleri Beslemeli Ağlar

Bu ağ mimarisinde bir yada daha çok gizli katman vardır.

Gizli katmanlardaki nöronlarda gizli nöronlar olarak adlandırılır. Bu yapı özellikle çok girişin olduğu durumlarda daha hassas sonuç elde edilmesini sağlar.

Giriş katmanından gelen veriler gizli katmanlarda çıktı üretir ve bu çıktılar ilerideki katmanlara bağlıdır.



Geri Beslemeli Ağlar

Nöronların çıktıları önceki katmanlardaki nöronlara ya da nöronun kendisine girdi olabilir.

Bir ya da birden çok gizli katman içerebilir.

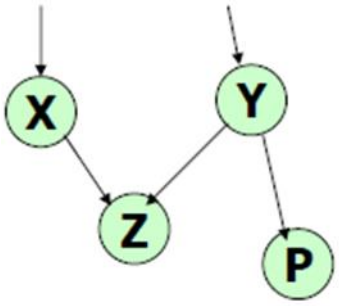
Geri besleme sayesinde hem ağın öğrenme kabiliyeti artar hem de ağ daha etkin bir biçimde işler.

Güçlü Yanları: Gürültülü, hatalı verilerle çalışabilir. Sayısal tahmin, sınıflandırma ve kümeleme problemlerinde kullanılabilir. Karmaşık problemlerin çözümünde iyi sonuç verir. Önceki deneyimlerden öğrenebilir, bir kez eğitildiklerinde yeni bir veri kümesine hemen cevap verebilir. Bir örnekten hareket ederek diğer örnekleri açıklayabilir. Eğitiminde kullanılması için gerekli bir varsayıma yoktur.
Zayıf Yanları: Elde ettikleri çözümlerin gerekçelerini açıklayamazlar. Optimum sonuca ulaşacaklarının garantisi yoktur. Ağın kalitesi ve kapasitesi, uygulamadaki hızı ile orantılıdır. Öyle ki, düğümlerin sayısındaki artış bile zamanın çok daha artmasına sebep olabilir.

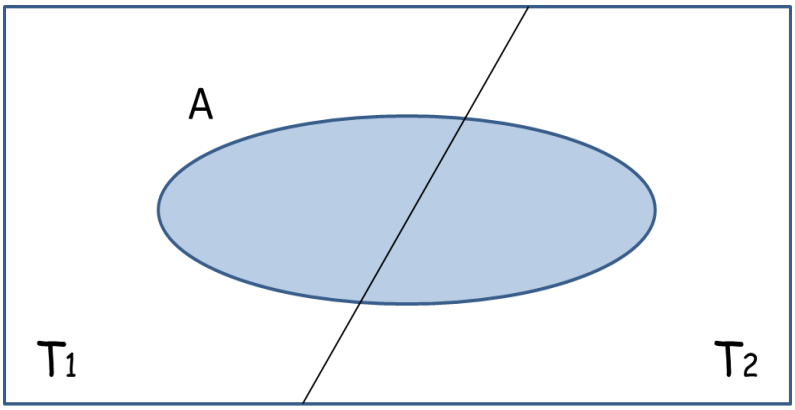
d. Bayesçi Ağlar veya Bayes Sınıflandırıcı

Literatürde bayes network veya blief network (inanç ağı) olarak da geçen ağların özelliği istatistiksel ağlar olmaları ve düğümler (nodes) arası geçiş yapan kolların (edges) istatistiksel kararlara göre seçilmesidir. Bayesçi ağlar değişkenler arası ilişkileri belirlemek için grafiksel modellerden yararlanmaktadır.

Bayesçi Ağlar bir modelde yer alan değişkenler arası bağımlılık ilişkilerini tek bir bağımlı değişkene ve herhangi bir varsayıma bağlı kalmadan, sayısal veriler olarak ortaya koyar.



- düğümler: rasgele değişkenler
- ayrıtlar: olasılıklı bağlılık
- *X* ve *Y*, *Z* değişkeninin ebeveyni
- *Y*, *P* değişkeninin ebeveyni
- *Z* ve *P* arasında bağ yok



Bayes ağlarını anlayabilmek için Bayes teoremini hatırlamakta yarar vardır. Bayes teoremi olasılık hesaplarında önemli bir yere sahiptir. Bayes teoremine dayanarak sınıflandırma işlemi yapmak mümkündür. Bu teoremin esasını ortaya koymak için T1 ve T2 gibi iki olayı göz önüne alalım.

Öncelikle koşullu olasılıkla ilgili şunları bilmeliyiz;

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

$$P(A \cap B) = P(A) * P(B/A)$$

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

$$P(A \cap B) = P(B) * P(A/B)$$

$$\rightarrow \quad P(B/A) = \frac{P(B) * P(A/B)}{P(A)}$$

Bayes teoremini açıklarsak;

$$\boldsymbol{P(T_1/A) = \frac{P(A/T_1)}{P(A)} P(T_1) \quad ise \quad A = (T_1 \cap A) \cup (T_2 \cap A)}$$

$$\boldsymbol{P(A) = P(T_1 \cap A) + P(T_2 \cap A) \qquad ise \qquad P(T_1/A) = \frac{P(A/T_1)P(T_1)}{P(T_1 \cap A) + P(T_2 \cap A)}}$$

$$\boldsymbol{Bayes\ teoremi \qquad P(T_j/A) = \frac{P(A/T_j)P(T_j)}{\sum_{j=1}^{n} P(A \cap T_j)}}$$

Bayes sınıflandırıcısına bir örnek verecek olursak;

Örneğin bir torbada bazı cisimlerin bulunduğunu varsayalım. Cisimlerin yuvarlak ve kırmızı olduğunu bildiğimizi varsayalım. Hipotez "bu cisim bir elmadır." olsun. Bu durumda P(H) bir önsel olasılık olarak karşımıza çıkar. Yani başlangıçta bu olasılığın ne olduğunu biliyoruz. Ancak P(H/X) olasılığı ise H koşullu üzerine kurulduğundan bir "sonsal olasılık" olarak değerlendirilir. Yani bir X'in kırmızı ve yuvarlak olma olasılığı biliniyorsa "bu bir elmadır" sonucunu doğurur.

Bu durumun bayes bağıntısı: $\boldsymbol{P(H/X) = \frac{P(X/H)}{P(X)}}$

Naïve - Bayes Sınıflandırıcı

Naive bayes algoritmasında her kriterin sonuca olan etkilerinin olasılık olarak hesaplanması temeline dayanmaktadır. Niteliklerin hepsi aynı derecede öneme sahiptir. Niteliklerin birbirinden bağımsız olduğu varsayılır. Yani bir niteliğin diğeri hakkında bilgi içermediği kabul edilir.

Örneğin elimizde tenis maçının oynanıp oynanmamasına dair bir bilgi olduğunu düşünelim. Ancak bu bilgiye göre tenis maçının oynanması veya oynanmaması durumu kaydedilirken o anki hava durumu, sıcaklık, nem ve rüzgâr durumu bilgileri de alınmış olsun. Biz bu bilgileri değerlendirdiğimizde varsayılan tahmin yöntemleri ile "hava bugün rüzgârlı, tenis maçı bugün oynanmaz" şeklinde kararları farkında olmasak da veririz. Ancak, VM bu kararların tüm kriterlerin etkisi ile verildiği bir yaklaşımdır. Dolayısıyla biz ileride öğrettiğimiz sisteme "bugün hava güneşli,

sıcak, nemli ve rüzgâr yok" şeklinde bir bilgiyi verdiğimizde sistem eğitildiği daha önce gerçekleşmiş istatistiklerden faydalanarak tenis maçının oynanma ve oynanmama ihtimalini hesaplar ve bize tahminini bildirir.

Güçlü Yanları: İstatistiksel bir teoreme dayandığından gerçeklemesi kolaydır.
Zayıf Yanları : Sınıf bilgisi verildiğinde niteliklerin bağımsız olduğu varsayımı olması. Çünkü gerçek hayatta değişkenler birbirine bağımlıdır. Değişkenler arası ilişki modellenemiyor. Koşullu olasılıklarla yorumlanıyor.

e. Regresyon Modelleri

1- Lineer Regresyon

Regresyon, iki (ya da daha çok) değişken arasındaki doğrusal ilişkinin fonksiyonel şeklini, biri bağımlı diğeri bağımsız değişken olarak bir doğru denklemi olarak, göstermekle kalmaz, değişkenlerden birinin değeri bilindiğinde diğeri hakkında kestirim yapılmasını sağlar. Klasik regresyon tanımıyla, makine öğrenmesinde kullanılan regresyon algoritmalarının temel mantığı aynıdır.

Lineer regresyon sayısal değerlerin tahmini için uygulanır. Nominal yani tamamıyla String verilerin tahminini gerçekleştiremez. Gerçekleştirmesi için bu değerlerin sayısal değere dönüştürülüp daha sonra formül üzerinde uygulanması gerekir.

$$Y=\beta_1 + \beta_2 X_2 + \beta_3 X_3 +...+ \beta_k X_k + u$$

Çoklu regresyon denklemi örneği

Tüm bağımsız değişkenler numerik veya kategorik olmalı, bağımlı değişken ise sürekli olmalı. Bunun yanında uygun bir regresyon analizi yapmak için veri setinin bazı varsayımları sağlaması beklenir.

1- Bağımlı değişken normal dağılım göstermeli.

   Normallik varsayımına uyulmazsa Tip1 hata olasılığı artar. Bu varsayıma uyulmadığı takdirde, veriler üzerinde dönüşüm yaparak normalliğe yaklaştırmak gerekmektedir.

2- Bağımlı değişken ve bağımsız değişkenler arasında doğrusal bir ilişki olmalı.

   Bağımlı değişkenle bağımsız değişken arasındaki ilişkinin doğrusal olması gerekir. Doğrusallık Scatter Plots ile kontrol edilebilir. Dağılımların serpilme grafiğine bakarak değişkenler arasında

doğrusal bir ilişki olup olmadığına bakılır. İlişkinin doğrusal olduğuna karar verildikten sonra "Pearson Korelasyon" katsayısından faydalanarak ilişkinin derecesine bakılabilir.

3- Hataların normal dağılım göstermeli ve ortalamaları sıfır veya sıfıra yakın olmalıdır.
4- Hatalar arasında korelasyon olmamalı.

Hataların arasında otokorelasyon olmaması hataların bağımsız olduğu anlamına gelir. Hatalar arasında ilişki olmaması gerekir. Otokorelasyon plotları ile bu durum kontrol edilebilir. Hatalar veri seti içinde bir kolona kaydedilerek bu kolona ait bir plot oluşturulur. Ayrıca Durbin-Watson testi otokorelasyonun kontrolü için kullanılır. DW sonucunun 2 civarında olması otokorelasyonun olmadığını yani hatalar ilişkili değil anlamına gelir.

5- Bağımsız değişkenler ve hatalar birbiriyle ilişkisiz olmalı.
6- Varyanslar homojen olmalı.

Eş vanyanslılık (Homoscedaticity) varsayımı, hataların her gözlem için eş varyansa sahip olması gerekliliğini ifade eder. Scatter Plot yardımıyla eş varyanslılığı kontrol edebiliriz. Bu varsayımın ihlali analize engel olmamakla birlikte varsayımın sağlanmaması durumunda testin gücü düşmekte ve Tip 2 hata oranı artmaktadır.

7- Bağımsız değişkenler birbirlerinden bağımsız olmalı.

Çoklu doğrusal bağlantı, regresyon modelindeki bağımsız değişkenlerin aralarında bir bağlantı olması durumudur. Bağımsız değişkenler arasında böyle bir ilişki olması testin güvenilirliğini azaltmaktadır. Varyans Enflasyon Faktörü (VIF) 10'un altında, tolerans istatistikleri de 0,2'nin üstünde olduğunda çoklu bağlantı sorunu yoktur deriz.

Güçlü Yanları: Bir veya daha fazla bağımsız değişkenin bağımlı değişkene göreceli etkisini belirleme becerisi iyidir. Belirsizlikleri veya anomalileri belirleme becerisi yüksektir.
Zayıf Yanları: Aykırı değerlere karşı çok hassastır. Regresyon çizgisini ve nihayetinde öngörülen değerleri büyük ölçüde etkiler. Doğrusal olmayan ilişkiler olduğunda doğrusal regresyon kötü performans gösterir. Doğal olarak daha karmaşık desenler yakalayacak kadar esnek değildirler ve doğru etkileşim terimlerini veya polinomlarını eklemek zor ve zaman alıcı olabilir.

2- Lojistik Regresyon

Amaç doğrusal regresyonda olduğu gibi en az değişkeni kullanarak en iyi uyuma sahip olacak şekilde bağımlı değişken ile bağımsız değişkenler arasındaki ilişkiyi tanımlayabilen ve genel olarak kabul edilebilir modeli kurmaktır.

Lojistik regresyon ile doğrusal regresyon arasındaki temel fark, doğrusal regresyonda bağımlı değişken sürekli olabilirken lojistik regresyonda bağımlı değişken kategoriktir. Bağımlı değişken değerleri sıralı ise, sıralı lojistik regresyon denir. Bağımlı değişken çok sınıflıysa, o zaman çoklu lojistik regresyon olarak denir.

Doğrusal regresyonda bağımsız değişkenler regresyon katsayılarıyla çarpılarak bağımlı değişken tahmin edilir. Lojistik regresyonda bağımsız değişkenlerin değerlerini bildiğimizde bağımlı değişkenin meydana gelme olasılığı tahmin edilmeye çalışılır.

$$\hat{p} = \frac{\exp(b_0 + b_1X_1 + b_2X_2 + ... + b_pX_p)}{1 + \exp(b_0 + b_1X_1 + b_2X_2 + ... + b_pX_p)}$$

Çoklu lojistik regresyon denklemi örneği

Doğrusal regresyonun varsayımlarından biri ilişkinin doğrusal olmasıdır. Fakat bağımlı değişkenin kategorik olduğu durumlarda bu varsayım ihlal edilir. Bunu çözmenin bir yolu verilere logaritmik dönüşüm uygulamaktır. Bu işlem doğrusal olmayan bir ilişkiyi koruyarak ilişkinin formunu doğrusal hale getirir. Yani doğrusallık varsayımı ihlal edilmeden çoklu doğrusal regresyon, logaritmik terimlerle açıklanmış olur. Sonuçta bağımlı değişkenin değeri 0 ile 1 arasında çıkar. Değer sıfıra yakınsa bağımlı değişkenin olma olasılığı düşük, 1'e yakınsa yüksektir.

Güçlü Yanları : Lojistik regresyon, bağımlı ve bağımsız değişkenler arasında doğrusal bir ilişki gerektirmez. Farklı türdeki ilişkileri işleyebilir.
Zayıf Yanları : Büyük örneklem boyutları gerektirir, çünkü en düşük olasılık tahminleri düşük örneklem büyüklüklerinde sıradan en küçük kareden daha az güçlüdür.

3- En Küçük Kareler Yöntemi (EKK)(OLS)

EKK regresyon yöntemi hata kareler toplamını en küçük yapmayı amaçlayan istatistiksel bir yöntemdir. (Bütün noktaların regresyon doğrusuna uzaklığının karelerinin toplamını en küçük yapan yöntem.)Bu yöntem, gözlemlenen verilerin normallik, sabit varyanslılık, sapan değer içermeme gibi bazı varsayımların sağlandığı durumlarda güvenilir tahminler elde edilmesini sağlamaktadır. (Varsayımları doğrusal regresyon ile aynıdır.) İstatistiksel çözümlemelerde EKK yöntemi, matematiksel işlemlere en uygun tahmin yöntemi olarak kullanılsa da varsayımların ihlaline karşı olan dayanıksızlığı nedeniyle eleştirilmekte ve alternatif olarak daha güçlü yöntemler önerilmektedir. Regresyon çözümlemesinde varsayımların sağlanmadığı durumlardan biri de veri kümesinin sapan değer içermesidir. Sapan değer, bir veri kümesinde gözlemlerin çoğunun sahip olduğu dağılıma veya modele uymayan gözlemler olarak ifade edilebilir. Sapan değer içeren veri kümesinde varsayımların sağlanamamasından dolayı kurulan regresyon modelinden alınan sonuçlarda yanıltıcı olmaktadır. Bu nedenle regresyon çözümlemesinde veri analizi oldukça önemli bir yer tutmaktadır. Sapan değerlerin veri kümesinden çıkartılması regresyon denklemini tamamen veya kısmen değiştirebilmektedir. Bu nedenle büyük artık değerlere sahip olan gözlemler, regresyon çözümlemesinde oldukça etkilidirler. Böyle durumlarda sapan değerlerin tespiti ve sonuçların güvenirliliği için güçlü regresyon yöntemlerini tercih etmek daha uygundur. Bu güçlü yöntemlerden biri de EKK yöntemidir.

Bir örnek verecek olursak; bir sınıfta bulunan öğrencilerin iki farklı dersten aldıkları notları biliyor olalım. Her öğrenci için, yapılan üçüncü bir sınav notunun değerini bu yönle tahmin edebiliriz. Oluşturulacak 3 bilinmeyenli bir denklemin katsayılarını en küçük kareler yöntemi ile belirleyip,

gerekli değerleri (bilinen iki sınav notları) fonksiyonda yerine koyarsak öğrencilerin üçüncü sınavdan beklenen notlarını bulabiliriz.

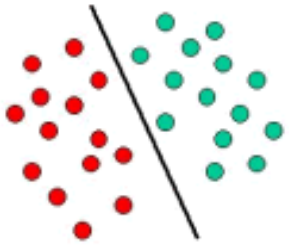
Gauss-Markov Teoremi

Klasik doğrusal regresyon modelinin varsayımları geçerliyken EKK tahminleri en iyi özellikleri taşır. Bu özellikler Gauss-Markov teoremi tarafından tanımlanmıştır. Bu teoreme göre klasik doğrusal regresyon modelinin varsayımları geçerliyken EKK tahmin edicisi doğrusal en iyi sapmasız tahmin edicidir. Doğrusallık, modelin bağımlı değişkeni gibi rassal bir değişkenin doğrusal fonksiyonu olmasını, sapmasızlık, ortalaması veya beklenen değerinin gerçek değerine eşit olmasını, en iyilik (veya etkinlik) ise doğrusal sapmasız tahmin ediciler içinde en düşük varyansa sahip olmayı ifade eder.

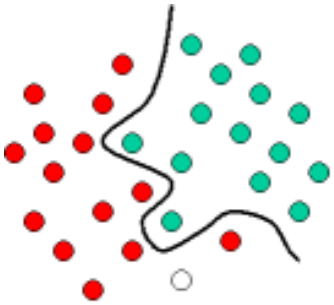
Güçlü Yanları: Lineer cebirden yaygın olarak kullanılan algoritmaları kullanarak bir bilgisayara uygulamak kolaydır. Modern bilgisayarlarda uygulanması etkili, bu nedenle yüzlerce özellik ve on binlerce veri noktası ile ilgili sorunlara bile çok hızlı bir şekilde uygulanabilir. Diğer birçok regresyon tekniğinden matematiksel olarak daha kolay analiz edilir.
Zayıf Yanları: Eğitim setinde aykırı değerler olduğunda iyi öğrenemez. Doğrusal olmayan ilişkiler olduğunda iyi tahmin yapamaz. Bağımsız değişken sayısı çok olduğunda gücü azalır. Değişkenlerin bağımsız olması gerekir fakat bu gerçek hayatta çok da mümkün değildir. Gürültülü veri olduğunda iyi sonuç vermez.

f. Destek Vektör Makinesi (DVM)(SVM)

Bu yöntem sınıflamayı doğrusal ya da doğrusal olmayan bir fonksiyon yardımıyla yerine getirir. Destek vektör makinesi yöntemi, veriyi birbirinden ayırmak için en uygun fonksiyonun tahmin edilmesi esasına dayanır. DVM'ler nonparametrik sınıflayıcılardır. Dağılım hakkında herhangi bir ön bilgi varsayımı yoktur.



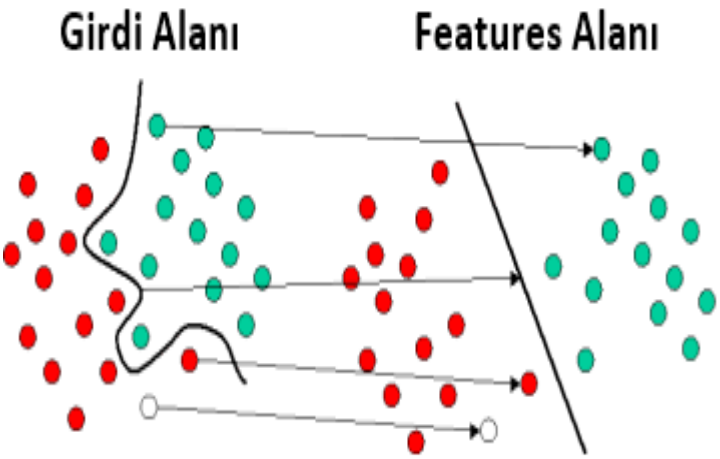
Yandaki örnekte şematik bir örnek gösterilmektedir. Bu örnekte, nesneler ya YEŞİL veya KIRMIZI sınıfına aittir. Ayırma çizgisi, sağ taraftaki tüm nesnelerin YEŞİL olduğu ve solundaki tüm nesnelerin KIRMIZI olduğu bir sınır tanımlıyor. Sağa düşen yeni nesne (beyaz daire) YEŞİL (veya ayırıcı çizginin soluna düşerse KIRMIZI olarak sınıflandırılır) olarak sınıflandırılır.



Yukarıdaki çizgisel bir sınıflandırıcının klasik bir örneğidir, yani bir nesne grubunu kendi gruplarına (bu durumda YEŞİL ve KIRMIZI) bir çizgi ile ayıran bir sınıflandırıcıdır. Bununla birlikte, sınıflandırma görevlerinin çoğu basit değildir ve optimal bir ayrımı yapmak, yani test verisini mevcut örneklere (tren örnekleri) göre doğru olarak sınıflandırmak için genellikle daha karmaşık yapılara ihtiyaç duyulmaktadır. Bu durum yandaki şemada gösterilmektedir. Önceki şemayla karşılaştırıldığında, YEŞİL ve KIRMIZI nesnelerin tam olarak

ayrılmasının eğri (bir çizgiden daha karmaşık) gerektirdiği açıktır. Farklı sınıf üyeliklerinin nesnelerini birbirinden ayırmak için çizgileri ayıran çizgiye dayalı sınıflandırma görevleri hiper plan sınıflandırıcıları olarak bilinir.

Yandaki şema Destek Vektör Makinelerinin arkasındaki temel fikri göstermektedir. Burada, orijinal nesnelerin (sol taraf), çekirdekler olarak bilinen bir takım matematiksel fonksiyonlar kullanılarak eşleştirildiğini, yani yeniden düzenlendiğini görüyoruz. Nesneleri yeniden düzenleme işlemi, haritalama (mapping-dönüşüm) olarak bilinir. Bu yeni ayarda eşlenmiş nesneler (sağ taraf) doğrusal olarak ayrılabilir ve bu nedenle karmaşık eğri (sol taraf) oluşturmak yerine, tek yapmamız gereken, ayırt edebilen en uygun çizgiyi bulmaktır YEŞİL ve KIRMIZI nesneler.



Temel olarak iki sınıfı bir doğru veya düzlem ile birbirinden ayırmaya çalışır. Bu ayırmayı da sınırdaki elemanlara göre yapar.

Bağımsız değişkenler doğrusal olarak ayrılabildiğinde; verileri ayırabilecek sonsuz sayıdaki doğru içerisinden marjini en yüksek yapacak olan doğruyu seçmeyi hedeflemektedir. Doğrusal olarak ayrılamadığında; orijinal çalışma verisini yüksek boyuta dönüştürmek için doğrusal olmayan haritalama (mapping) kullanmaktadır. Verinin taşındığı yeni boyutta “marjini en büyük (optimal)“ ayırıcı düzlemi araştırmaktadır.

DVM modellerinde kullanılabilen çekirdekler vardır. Bunlar; doğrusal, polinom, radyal taban fonksiyonu (RBF) ve sigmoid’dir.

Güçlü Yanları: Yüksek doğruluk, Karmaşık karar sınırları modelleyebilme, Çok sayıda bağımsız değişkenle çalışabilme, Hem doğrusal olarak hem doğrusal olmayan verilere uygulanabilme, Diğer birçok yönteme kıyasla overfitting sorunun az olması.
Zayıf Yanları: Olasılıksal tahminler üretememe / Nokta tahmini (Var-Yok, A sınıfı-B Sınıfı vb..), Çekirdek fonksiyonları pozitif tanımlı sürekli simetrik fonksiyonlar olmalı.

2- KÜMELEME ALGORİTMALARI

Kümeleme, veriyi sınıflara veya kümelere ayırma işlemidir. Aynı kümedeki elemanlar birbirleriyle benzerlik gösterirlerken, başka kümelerin elemanlarından farklıdırlar. Kümeleme veri madenciliği, istatistik, biyoloji ve makine öğrenimi gibi pek çok alanda kullanılır. Kümeleme modelinde, sınıflama modelinde olan veri sınıfları yoktur. Verilerin herhangi bir sınıfı bulunmamaktadır. Sınıflama modelinde, verilerin sınıfları bilinmekte ve yeni bir veri geldiğinde bu verinin hangi sınıftan olabileceği tahmin edilmektedir. Oysa kümeleme modelinde, sınıfları bulunmayan veriler gruplar halinde kümelere ayrılırlar. Bazı uygulamalarda kümeleme modeli, sınıflama modelinin bir önişlemi gibi görev alabilmektedir.

Genellikle kümeleme denetimsiz bir biçimde gerçekleşir. Denetimsiz öğrenme yetisinden dolayı, veriler içinde gizli olan örüntüleri ortaya çıkarmayı sağlar.

Literatürde pek çok kümeleme algoritması bulunmaktadır. Kullanılacak olan kümeleme algoritmasının seçimi, veri tipine ve amaca bağlıdır.

1-K-Means Algoritması

K-means yöntemi, ilk önce n adet nesneden rasgele k adet nesne seçer ve bu nesnelerin her biri, bir kümenin merkezini veya orta noktasını temsil eder. Geriye kalan nesnelerden her biri kendisine en yakın olan küme merkezine göre kümelere dağılırlar. Yani bir nesne hangi kümenin merkezine daha yakın ise o kümeye yerleşir. Ardından her küme için ortalama hesaplanır ve hesaplanan bu değer o kümenin yeni merkezi olur. Bu işlem tüm nesneler kümelere yerleşinceye kadar devam eder. Bağımsız değişkenlerin normal dağılımı varsayımı vardır.

Kısaca algoritma temel olarak 4 aşamadan oluşur:

1-Küme merkezlerinin belirlenmesi

2-Merkez dışındaki örneklerin mesafelerine göre sınıflandırılması

3-Yapılan sınıflandırmaya göre yeni merkezlerin belirlenmesi (veya eski merkezlerin yeni merkeze kaydırılması)

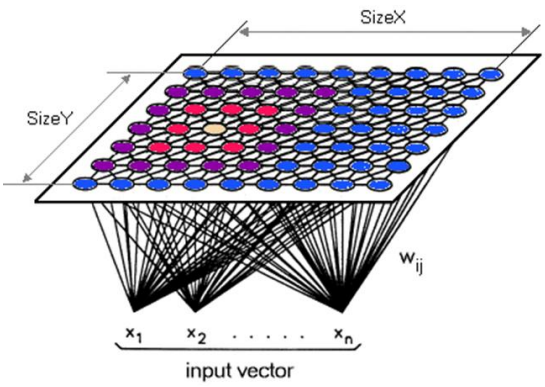
4-Kararlı hale (stable state) gelinene kadar 2. ve 3. adımların tekrarlanması

Güçlü Yanları: Başlangıç noktası ve küme sayısı verildiğinde daha yüksek doğruluk elde eder. Büyük veri setlerinde çalışabilir.
Zayıf Yanları: Küme sayısı belirleyememe. Aynı gruba ait iki merkezin seçilebilmesi.

2- Kohonen Ağlar (Kendini Düzenleyen Haritalar)

Kohonen tarafından önerilen “kendini düzenleyen haritalar” geniş bir uygulama alanına sahiptir. Kendini düzenleyen haritalar, yapay sinir ağları tekniklerinden biri olan standart ileri besleme ağlarından sonra en yaygın kullanılan teknik olup, kümeleme ve boyut indirgeme için kullanılan denetimsiz iki katmanlı bir yapay sinir ağı olarak tanımlanabilir. Bilindiği üzere yapay sinir ağları, insan beyninden esinlenerek geliştirilmiş, ağırlıklı bağlantılar aracılığıyla birbirine bağlanan ve her biri kendi belleğine sahip işlem elemanlarından oluşan paralel ve dağıtılmış bilgi işleme yapılarıdır. Kohonen ağları giriş ve çıkış nöronlarından oluşur. Giriş nöronlarının sayısını veri setindeki değişken sayısı belirler. Çıkış nöronlarının her biri bir kümeyi temsil eder. Diğer yapay sinir ağlarından farklı olarak, çıkış katmanındaki nöronların dizilimi çok önemlidir. Bu dizilim doğrusal, dikdörtgensel, altıgen veya küp şeklinde olabilir.

Kendini düzenleyen haritalar, standart kümeleme algoritmalarından farklı bir yapıya sahiptirler. Temel farklılık, bu yöntemin ayrı kümeler üzerinde işlem yapması yerine, birbiri ile ilişkili veri noktalarını gruplara ataması şeklinde ifade edilebilir. Çok boyutlu ölçekleme tekniğine benzer bir teknik olduğu söylenebilir. Kendini düzenleyen haritalar tekniğinde, her girdi düğümü bir boyuta karşılık gelirken, her çıktı düğümü de iki boyutlu ızgarada bir düğüme karşılık gelir.



Güçlü Yanları: Küme sayısını belirler. Büyük hacimli veriler üzerinde çalışabilme.
Zayıf Yanları: Eğitim uzun zaman alır. Eğitim parametlerinin ayarlanması zor ve her farklı parametre farklı sonuç verir.

3-Ward’ın Minimum Varyans Yöntemi

İki aşamalı kümeleme algoritmaları denildiğinde, ilk akla gelen Punj ve Steward (1983) tarafından önerilen klasik iki aşamalı kümeleme algoritmasıdır. İki aşamalı kümeleme, veri yapısında hem kategorik hem de sürekli değişkenler olduğu durumda kullanılan, Ward’ın minimum varyans yöntemi ile K-means yönteminden oluşan bir hibrid yaklaşımdır. Böyle bir karma yaklaşımın avantajı, Ward’ın minimum varyans yönteminin, K-means yönteminin gerektirdiği küme sayısını hesaplamasından ileri gelmektedir.

Güçlü Yanları: Küme sayısını belirleyebilmesi.
Zayıf Yanları: Büyük hacimli verilerle çalışma zorluğu, Aykırı değerlerden kolayca etkilenmesi, Geri alınamama.

4- Ağaç Yapılı Kümeleme (Hierarchical Clustering) ve DBSCAN

Elimizde N-tane nokta olsun. Ağaç yapılı kümeleme algoritmasını aşağıdaki gibi özetleyebiliriz:

Her bir nokta, ayrı bir öbek olarak işaretlenir. Yani, elimizdeki N-tane nokta için N-tane öbek elde ederiz. En yakın 2 öbek bulunur ve birleştirilir. Elimizdeki öbek sayısı 1 azalmış olur. Yeni elde ettiğimiz öbekle diğer öbekler arasındaki uzaklıklar hesaplanır. Tüm noktalar tek bir öbekte olana kadar 2. ve 3. adımlar tekrar edilir.

DBSCAN (Density-based spatial clustering of applications with noise)

Algoritmayı anlatmadan önce algoritmanın dayandığı iki kavramdan bahsetmemiz gerekiyor:

Tanım 1 (Density Reachability). p noktası, q noktasının \epsilon komşuluğunda ve q noktası, \epsilon komşuluğunda yeteri kadar komşu içeriyorsa, p noktası,q noktasından "yoğun erişebilir (dense reachable)" denir.

Tanım 2 (Density Connectivity). Eğer p ve q noktaları bir \epsilon yarıçaplı çember içerisinde ve bu iki noktaya komşu olan bir r noktası var ise, p ve q noktaları "yoğun bağlıdır(dense connected)" denir.

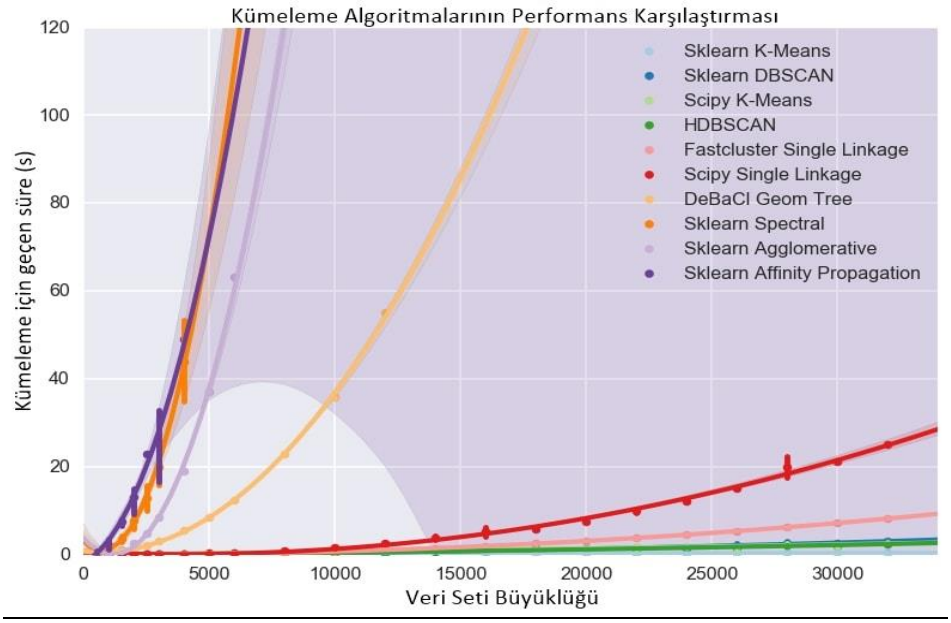
Elimizde yine N-tane nokta olsun. DBSCAN algoritması, iki parametre gerektirir: \epsilon ve bir öbek oluşturmak için gereken minimum nokta sayısı (minPts): Daha önce "ziyaret edilmemiş" keyfi bir başlangıç noktası seçilir. Bu noktanın \epsilon komşuluğundaki noktalar bulunur. Eğer komşu sayısı yeterli ise öbekleme işlemine başlanır ve bu nokta "ziyaret edildi" olarak işaretlenir; komşu sayısı yeterli değil ise de "gürültü (noise)" olarak işaretlenir (Gürültü olarak işaretlenen noktalar daha sonra bir öbeğe ait olabilir). Eğer bir öbeğe ait bir nokta bulunur ise, bu noktanın \epsilon komşuluğu da öbeğe dahil edilir ve 2.adım komşuluktaki tüm noktalar için tekrar edilir. Yeni bir ziyaret edilmemiş nokta bulunur ve işlenir (gürültü ya da ziyaret edildi olarak işaretlenir.). Tüm noktalar ziyaret edildi olarak işaretlenene kadar bu işlemler tekrar edilir. *Hiyerarşik kümeleme kısmının kaynağı için CTRL+Tıklayın*

Kümeleme Algoritmalarının Karşılaştırılması

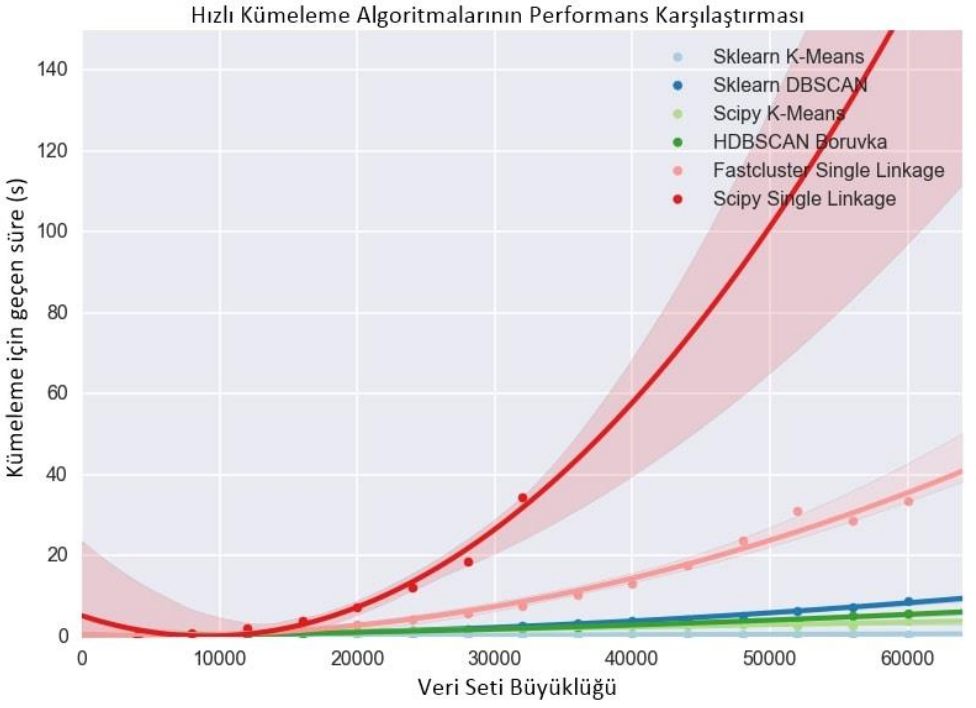
Veri Seti Büyüklüğü				
Algoritmalar	Etkileşimli	Kahve içme süresinde	Öğle yemeği boyunca	Gece boyunca
AffinityPropagation	2000	10000	25000	100000
Spectral	2000	5000	25000	75000
Agglomerative(Ward)	2000	10000	25000	100000
DeBaCl	5000	25000	75000	250000
ScipySingleLinkage	25000	50000	100000	250000
Fastcluster	50000	100000	500000	1000000
HDBSCAN	100000	500000	1000000	5000000
DBSCAN	75000	250000	1000000	2500000
SKLearn KMeans	1000000000	1000000000	1000000000	1000000000

Performans seçilen algoritmaya bağlıdır, ancak belirli algoritmalar (HDBSCAN, DeBaCl'den çok daha iyi hiyerarşik yoğunluk tabanlı kümeleme ve sklearn şimdiye kadar en iyi K-Means uygulaması olmuştur) için önemli derecede değişebilir. HDBSCAN'ın makul bir şekilde ele alabileceği kapsamın ötesinde verileri kümelemeniz gerekiyorsa, tablodaki tek algoritma seçenekleri DBSCAN ve K-Means'tır; DBSCAN, özellikle çok büyük veriler yavaştır, ancak K-Means kümeleme de çok zayıf olabilir - zor bir seçimdir.

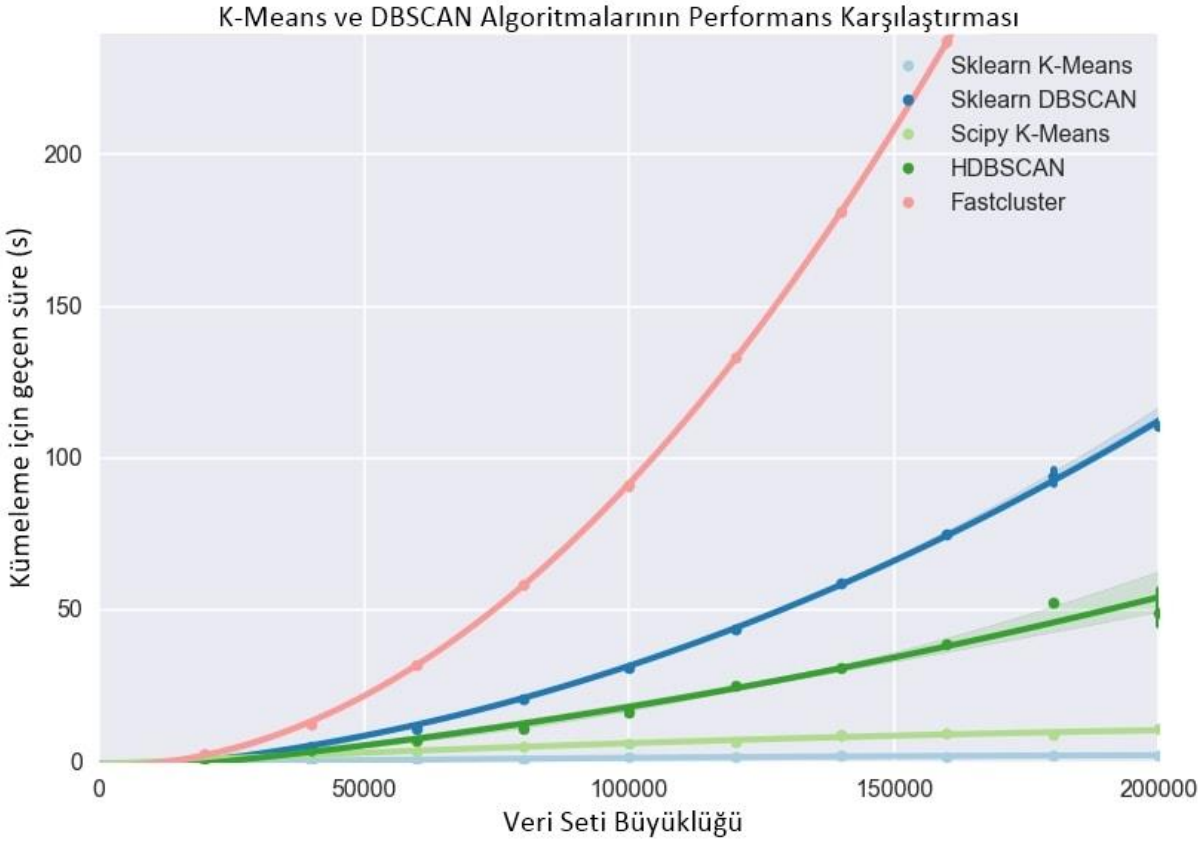
Python’da kullanabileceğimiz 10 algoritmanın karşılaştırılması:



Hızlı algoritmaların karşılaştırılması:



K-Means ve DBSCAN algoritmalarının karşılaştırılması:



3-BİRLİKTELİK KURALLARI

Verilerin birbiriyle olan ilişkilerini inceleyerek, hangi olayların eş zamanlı olarak birlikte gerçekleşebileceklerini ortaya koymaya çalışan veri madenciliği yöntemleridir. Özellikle pazarlama alanında uygulanmaktadır (Pazar sepet analizleri). Bu yöntemler birlikte olma kurallarını belirli olasılıklarla ortaya koyar. Birliktelik çözümlemelerinin en yaygın uygulaması perakende satışlarda müşterilerin satın alma eğilimlerini belirlemek amacıyla yapılmaktadır. Müşterilerin bir anda satın aldığı tüm ürünleri ele alarak satın alma eğilimini ortaya koyan uygulamalara ”Pazar sepet çözümlemesi” denilmektedir.

Örneğin; bir mağazadan parfüm alan müşterilerin %60’ının aynı alışverişte parfüm satın aldıklarını söylemek, bu birlikte gerçekleşen olaylara örnek olarak verilebilir.

Apriori Algoritması

Birliktelik kurallarının üretilmesi için kullanılan en yaygın yöntemdir. Aşamaları:

-Destek ve güven ölçütlerini karşılaştırmak üzere eşik değerler belirlenir. Uygulamadan elde edilen sonuçların bu eşik değere eşit ya da büyük olması beklenir.

-Destek sayıları hesaplanır. Bu destek sayıları eşik destek sayısı ile karşılaştırılır. Eşik destek sayısından küçük değerlere sahip satırlar çözümlemeden çıkarılır ve koşula uygun kayırlar göz önüne alınır.

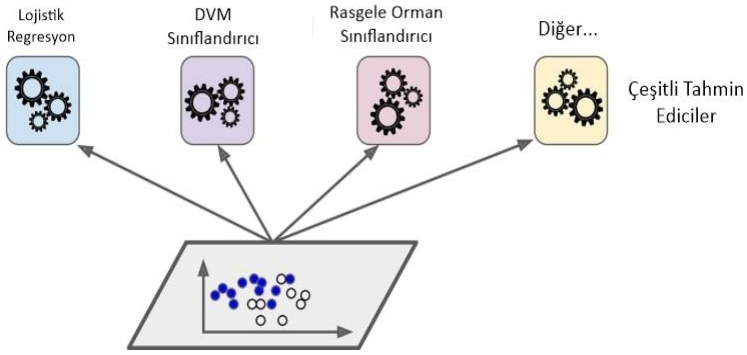
-Bu seçilen ürünler bu kez ikişerli gruplandırılarak bu grupların tekrar sayıları elde edilir. Bu sayılar eşik destek sayıları ile karşılaştırılır. Eşik değerden küçük değerlere sahip satırlar çözümlemeden çıkarılır.

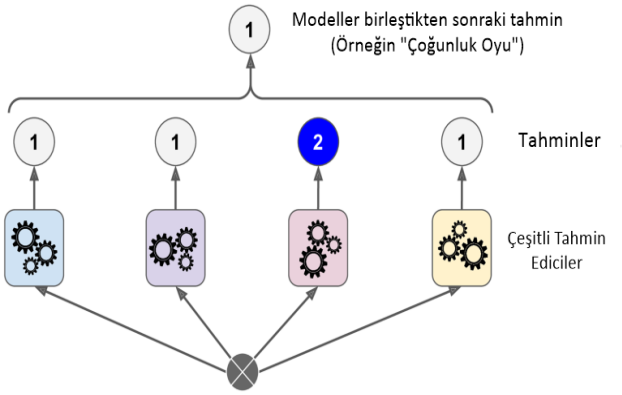
-Bu kez üçerli, dörderli vb. gruplandırmalar yapılarak bu grupların destek sayıları elde edilir ve eşik değeri ile karılaştırılır, eşik değere uygun olduğu sürece işlemlere devam edilir.

-Ürün grubu belirlendikten sonra kural destek ölçütüne bakılarak birliktelik kuralları türetilir ve bu kuralların her birisiyle ilgili olarak güven ölçütleri hesaplanır.

4- MODELLERİN BİRLEŞTİRİLMESİ

"Karmaşık bir soruyu binlerce insana soralım ve verdikleri cevapları birleştirelim. Çoğu durumda, bu birleştirilmiş cevap, tek bir uzmanın verdiği cevaptan çok daha iyi olacaktır. Bu "Wisdom of the crowd" olarak adlandırılır. Benzer olarak, birden fazla tahmin edicinin tahminlerini bir araya getirirsek, tek bir tahmin ediciden daha iyi sonuç elde edebiliriz. Tahmin edicilerin grubu "Ensemble", bu teknikte "Ensemble Learning" olarak adlandırılır."





Daha iyi bir sınıflandırıcı oluşturmanın basit bir yolu, bu sınıflandırıcılara tahminlerini sormak ve en çok "oy alan" sınıfı bulmaktır. Bu tip bir sınıflandırıcı "hard voting classifier" olarak adlandırılır.(Soldaki şemada örneklendirildi.)

Burada 3 algoritma gözlemin 1. Sınıfa ait olduğunu tahmin ediyor, 1 algoritma da 2. Sınıfa ait olduğunu tahmin ediyor. Çoğunluk oyu birleştirme yönteminde o gözlemin 1. Sınıfa ait olduğunu söyleriz.

Eğer sınıflandırıcılarımız bir sınıfa ait olma olasılıklarını da çıktı olarak verebiliyorsa, bu durumda sınıflandırıcıların verdikleri olasılıkların ortalamasını alıp tahminde bulunabiliriz. Bu da "soft voting classifier" olarak adlandırılır.

UYARI: Sınıflandırıcıların birbirlerinden farklı olmaları, ensemble yönteminde çok önemlidir. Sınıflandırıcıların aynı tip hata yapmaları önlenerek, ensemble sınıflandırıcının daha iyi sonuç vermesini sağlar.

Bagging ve Pasting Birleştirme Yöntemleri

Aynı algoritmayı verinin farklı altkümeleri üzerinde çalıştırmaktır. Altkümeleri oluştururken örnekleme işlemi yerine koyma ile yapılırsa bu yöntem bagging(bootstrap aggreating) olarak, yerine koyma ile yapılmıyor ise pasting olarak adlandırılır.

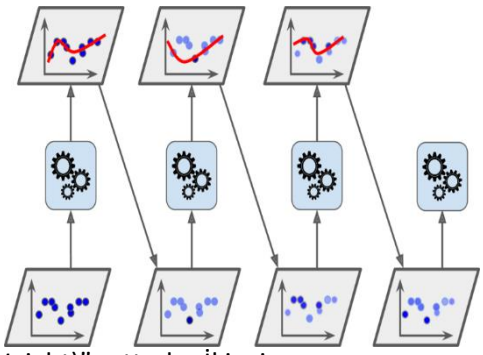
1- Rastgele Ormanlar (Random Forests)

Çok sayıda karar ağacının bagging ya da pasting metodları ile bir araya getirilmesinden oluşur.

2- Adaboost

Boosting

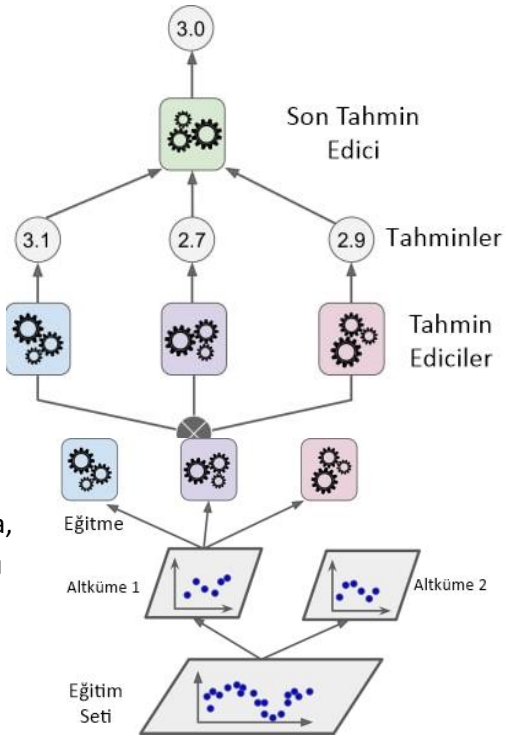
Boosting, bir çok zayıf öğreniciyi (weak learner) bir araya getirerek bir güçlü öğrenici (strong learner) oluşturmak anlamına gelir. Bir çok boosting metodunun ana fikri, tahmin edicileri ardışık olarak eğitmektir. En sık kullanılan boosting metodları, AdaBoost (Adaptive Boosting) ve Gradient Boosting metodlarıdır.



Bir tahmin edicinin, kendinden önce gelen tahmin ediciyi düzeltmesi için bir yol, kendinden önce gelen tahmin edicinin eksik öğrendiği (underfit) eğitim verilerine daha fazla dikkat etmesidir. Bu yöntem Adaboost algoritmasında kullanılır.

Bir Adaboost sınıflandırıcısı inşa etmek için, ilk olarak sınıflandırıcımızı eğitim seti üzerinde eğitiriz ve tahmin yaparız. Daha sonra, yanlış sınıflandırılan eğitim verilerinin "Göreli Ağırlığı (Relative Weight)" arttırılır. İkinci sınıflandırıcı, bu arttırılmış ağırlıklar ile eğitilir ve tekrar tahmin yapılır. Ağırlıklar yine güncellenir ve bu şekilde devam edilir. Tüm tahmin ediciler eğitildiğinde, tahminler, tüm tahmin edicilerin doğruluk oranlarına göre ağırlıkları dikkate alınarak bagging veya pasting metodları ile yapılır.

Stacking



Kısaca stacked generalization. Çok basit bir fikre dayanmaktadır: tahmin edicileri birleştirirken hard voting gibi yöntemler kullanmak yerine, neden birleştirmeyi yapacak bir model eğitmiyoruz?

Sağdaki grafikte, yeni veri noktası üzerinde 3 farklı tahmin edici ile tahmin yapılmış (3.1, 2.7 ve 2.9) ve son tahmin edici (blender veya meta learner olarak adlandırılır) de bu tahminleri alıp, son tahmini yapmaktadır (3.0).

Bir blender eğitmek için, sık kullanılan bir yaklaşım, bir hold-out kümesi kullanmaktır. İlk olarak eğitim seti iki altkümeye ayrılır. Altkümelerden birisi tahmin edicileri eğitmek için kullanılır. Daha sonra, tahmin ediciler ile ikinci altküme (held-out) üzerinde tahmin yapılır. Bu durumda ilk altkümedeki (hold-out) her bir değer için, elimizde 3 adet tahmin bulunur (elimizde 3 tane sınıflandırıcı olduğunu varsayıyoruz). Şimdi elimizdeki tahminleri ve gerçek değerleri kullanarak yeni bir eğitim seti oluşturuyoruz. Blender'ı da bu eğitim seti üzerinde eğitiyoruz. *Modellerin Birleştirilmesi kısmının kaynağı için CTRL+Tıklayın*

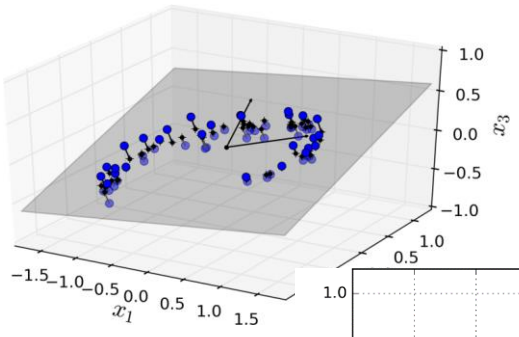
5-BOYUT İNDİRGEME

Birçok makine öğrenimi probleminde, verisetindeki herbir gözlem için yüzler, binler ve belki milyonlarca öznitelik bulunabilir. Bu kadar çok öznitelik olması hem oluşturulan modelin eğitimini yavaşlatmakta hem de iyi bir model oluşturmayı zorlaştırmaktadır. Neyse ki, çoğu zaman, öznitelik sayısını "başedebileceğimiz" bir sayıya indirgeyebiliriz.

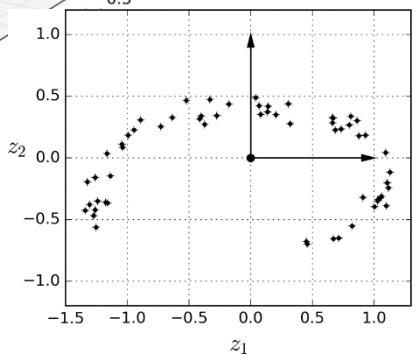
Boyut indirgeme, ayrıca, veri görselleştirme (Data visualization) için de oldukça yararlıdır. Boyutu ikiye (ya da üçe) indirgemek, çok boyutlu bir verisetini görselleştirmeyi mümkün kılar.

Projeksiyon

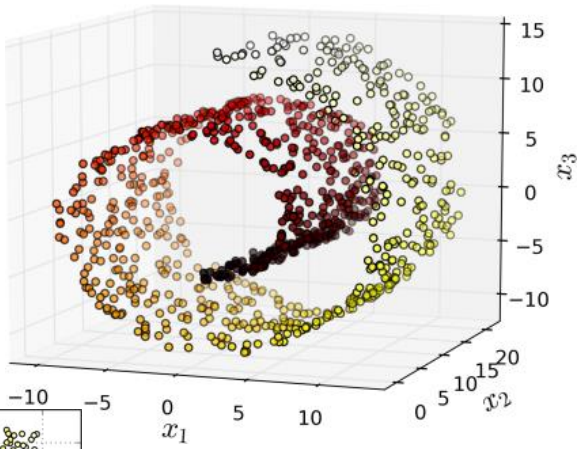
Çoğu problemde, eğitim verileri tüm boyutlara eş olarak dağılmamıştır. Bazı öznitelikler neredeyse sabitken(constant), bazıları da birbiriyle korelasyona sahip olabilir. Bunun bir sonucu olarak, eğitim verileri yüksek boyutlu bir uzayın daha düşük boyutlu bir altuzayında yer almaktadır. Örnek olarak, sağdaki grafikte, eğitim verileri düzlem etrafında uzanmaktadır.

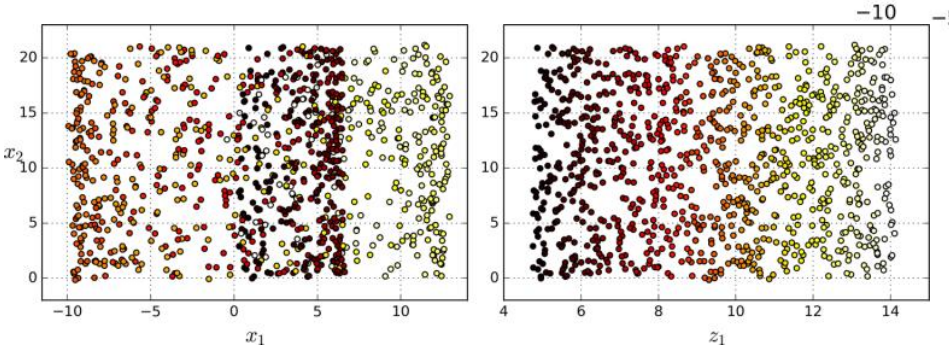


Eğer bu noktaların düzleme olan izdüşümlerini alırsak, 3-boyutlu bir verisetini 2-boyuta indirgemiş oluruz.



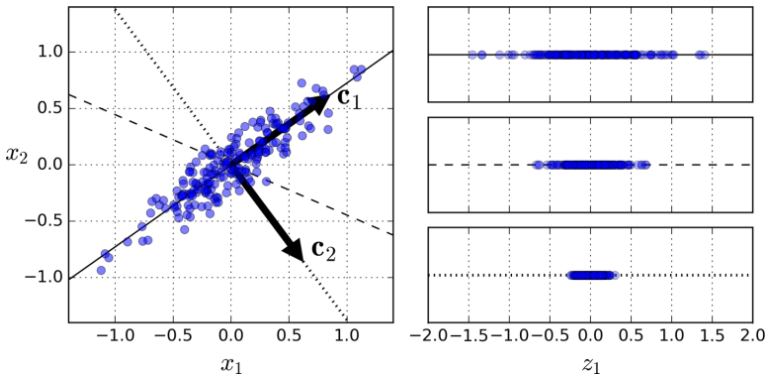
Ancak, projection, boyut indirgemede her zaman iyi bir yaklaşım değildir. Sağdaki verisetini göz önüne aldığımızda, noktaları bir düzleme izdüşürmek, verisetindeki noktaların birbirine karışmasına neden olacaktır. Ancak biz aşağıda, sağdaki gibi bir veriseti elde etmek istiyoruz.





Temel Bileşen Analizi

Sağdaki grafikte, verisetindeki noktaların üç farklı eksen üzerine izdüşümleri alınmıştır. Sağda, en üstteki izdüşümde maksimum varyans, en alttakinde ise minimum varyans vardır. Boyut indirgemede amacımız, maksimum varyans elde edilecek hiperdüzlemi belirlemektir. PCA algoritmasının ana fikri de, orjinal noktalar ile izdüşümleri arasındaki ortalama uzaklığı minimum yapmaktır.



Hiperdüzlemi belirlediğimiz birim vektörler “temel bileşen”(principal component) olarak adlandırılmaktadır. Yukarıdaki grafikte, c_1 ve c_2 temel bileşenlerdir. Peki temel bileşenleri nasıl bulacağız?

Bunun için tekil değer ayrışımı (singular value decomposition) olarak adlandırılan, bir matrisi üç farklı matrisin çarpımı şeklinde yazabildiğimiz bir metod kullanılabilir. Bu üç farklı matrisden biri de bizim temel bileşenlerimizi içermektedir. Temel bileşenlerimizi bulduktan sonra, verisetimizi temel bileşenler ile oluşturduğumuz bir matris ile çarparız. Böylece noktaların izdüşümleri bulunur.