3. Metot - Kullanılan Makine Öğrenme Algoritmaları

Bu bölümde makine öğrenme yöntemlerinden gözetimli öğrenim metotlarının regresyon algoritmalarını inceleyeceğiz. Regresyon, bir değişkenin bir veya daha fazla değişkenle arasındaki bağlantının matematiksel bir fonksiyonla gösterilmesidir[1]. Yani bağımlı(y), bağımsız(x) değişkenleri düşünüldüğünde y'nin, x'in bir fonksiyonu olarak ifade edilen ilişki biçimi regresyon olarak tanımlanır. Bu durum değişkenler arasında bir sebep-sonuç bağlantısını özetlemektedir. Matematiksel fonksiyon durumuna göre doğrusal ve doğrusal olmayan regresyon çeşitleri bulunmaktadır[2]. Bu çalışma da fonların fiyatlarının nasıl tahmin edileceği sorusu bağlamında değerlendirildiğinde, veri setindeki bağımlı değişken olarak fon fiyatlarının olması ve birden fazla bağımsız değişken değerin bulunması kapsamında regresyon algoritmalarının kullanılması daha uygun bulunmuştur. Regresyon metotlarından doğrusal çoklu regresyon algoritmalarından Kısmi En Küçük Kareler ve Ridge Regresyon ile doğrusal olmayan çoklu regresyon algoritmalarından ise Destek Vektör Regresyonu ve Yapay Sinir Ağları kullanılacaktır.

3.1 Doğrusal Çoklu Regresyon

Birden fazla bağımsız değişken ile bir bağımlı değişken arasındaki doğrusal bağlantıyı incelemektedir. Doğrusal çoklu regresyonun matematiksel olarak gösterimi ise;

$$Y_i = (b_0 + b_1X_1 + b_2X_2 + \ldots + b_iX_i) + e_i \qquad \textbf{(3.1)}$$

olarak ifade edilir.

3.1.1 Kısmi En Küçük Kareler Regresyonu (PLSR)

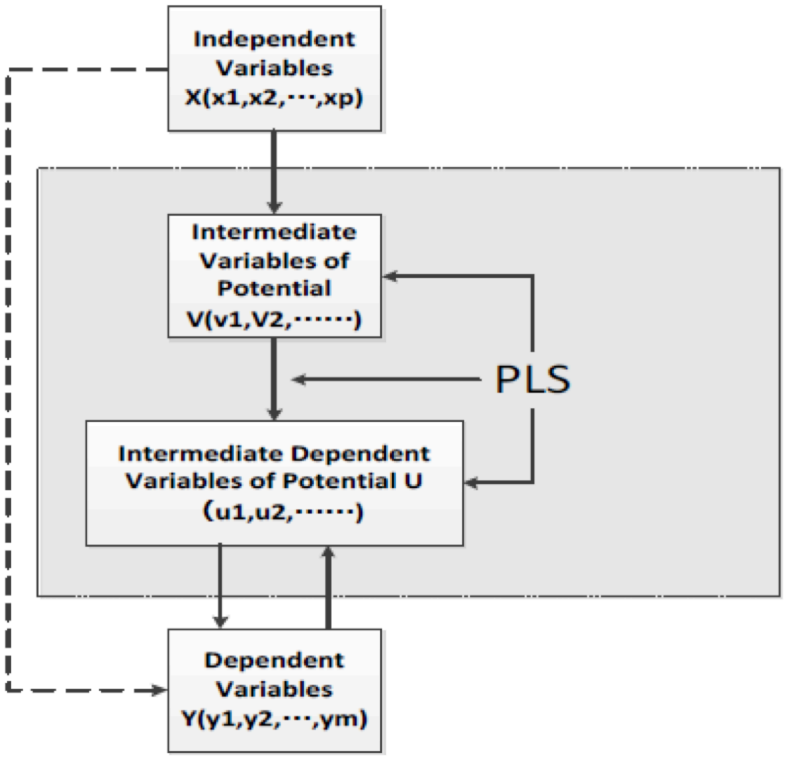
3.1.1.1 Teori

Kısmi En Küçük Kareler Regresyonu, 1960'larda Herman Wold tarafından bir ekonometri tekniği olarak geliştirilmiş ve temel bileşen analizi ile çoklu regresyon özelliklerini genelleştiren ve birleştiren yeni bir teknik olmaktadır. PLS yöntemi, bağımsız(X) değişken grubunun çok fazla olması ve bir dizi bağımlı(Y) değişken tahmin etmemiz gerektiğinde kullanılmaktadır. Teknik olarak ise, X ve Y değişkenleri arasındaki gizli bağlantıdan yararlanarak, veri blokları arasındaki ilişkiyi bulmayı amaçlamaktadır [3]. Ayrıca PLS çalışmaları ilkin, ekonomik ve sosyal durumları modelleme için kullanılmıştır. Fakat yaygın olarak kullanımına oğlu Svante Wold tarafından kemometrik alanındaki çalışmalarla başlanmıştır [4]. Kemometri, istatistik ve matematik

tekniklerinin kullanılarak, kimyasal sistemlerden bilgi alma bilimidir. Kimyasal analizlerde, veriden doğru bilginin ya da gizli bilginin açığa çıkarılmasına imkan tanıyan bir araçtır[5]. İstatistik alanındaki ilk kullanımına ise 1988 yılında Höskulddson [6] ve 1989 yılındaki Naes ve Marten [7] tarafından yapılan çalışmalar örnek verilebilir.

3.1.1.2 Matematiksel Model ve Algoritma

Bu kısımda PLS yöntemin temel matematiksel diyagramından bahsedilecektir. PLS için kullanacağım algoritma ise deneysel çalışma bölümünde fonların fiyat tahmin için geliştirilecek olan model algoritması olarak kullanılacaktır.



Şekil 3.1: PLS Matematiksel Diyagramı[8]

PLS matematiksel diyagram şemasından anlaşılacağı üzere, PLS genel olarak potansiyel bağımsız değişkenler ile potansiyel bağımlı değişkenler arasındaki doğrusal ilişki kurmayı amaçlamaktadır. M tane bağımlı değişkenler Y (y1, y2, y3,…, ym) ile p tane bağımsız değişkenler X (x1, x2, x3,…, xp) arasındaki ilişkiyi ise dolaylı olarak yansıtır. Potansiyel bağımsız değişkenler ve potansiyel bağımlı değişkenler, PLS regresyonundaki değişkenlerin doğrusal kombinasyonunu yansıtır ve iki varsayımı mevcuttur:

1. İki potansiyel değişken grubu, bağımsız değişken veya bağımlı değişkenler mutasyon bilgilerini taşımaktadır.

2. Potansiyel değişkenler arasındaki korelasyon değeri maksimize edilmektedir[8].

Bağımlı Değişken Y ve Bağımsız Değişken X için PLS regresyon matematiksel gösterim şeması şu şekildedir;

$$\mathbf{X} = \sum_{j=1}^{A} \mathbf{t}_j \mathbf{p}_j' + \mathbf{E} \text{ ve } \mathbf{Y} = \sum_{j=1}^{A} \mathbf{u}_j \mathbf{q}_j' + \mathbf{F}$$

Şekil 3.2: Bağımlı ve Bağımsız Denklem[9]

Yukarıdaki matematiksel gösterimde $\mathbf{t_j}$, $\mathbf{u_j}$ ler gizli değişkenler olup, $\mathbf{t_j}$ değişkenleri birbirlerine ve aynı bağlamda sonraki $\mathbf{u_j}$ değişken değerine diktir. PLS model için maksimum sayıda gizli değişken sayısı, bağımsız ve bağımlı değişkenler arasındaki kovaryans değeri maksimum olacak şekilde elde edilir[9].

Matematiksel model anlatımından sonra bu çalışmada kullanılacak PLS algoritması klasik ve standart olan NIPALS (Non-Linear Iterative Partial Least Squares; Doğrusal olmayan yinelemeli kısmi en küçük kareler) algoritması olacaktır. NIPALS bağımsız değişken değerleri birden fazla olan ve bağımlı değişken değeri tekil olan veri setleri için güçlü ve sağlam bir algoritma olduğundan dolayı tercih edilmiştir. Temel de tüm PLS algoritmalarının amacı kovaryans matrislerini en fazla sayıya ulaştıracak bileşenlerin elde edilmesidir[10].

NIPALS algoritmasının adımları;

j= 1,2,…J, bileşen sayısını gösterir.
$\mathbf{X_1} = \mathbf{X}$, $\mathbf{Y_1} = \mathbf{Y}$, orijinal matrisler.

1. Veri setimizde bağımlı değişken sayısı tek olduğu için direkt **Y** değişken sütunu $\mathbf{u_j}$ vektörü olarak tanımlanır.

2. **X** ve **Y** bileşenlerinin $\mathbf{u_j}$ üzerinedeki regresyonu **X** ve **u** arasındaki kovaryans değerini en çoklayan **w** ağırlık vektörü $w_j = X'_j u / (u'_j u)$ ile elde edilir.

3. $\mathbf{w_j} / ||\mathbf{w_j}||$ ve $\mathbf{w_j}$ vektörü normuna bölünüp vektör boyu 1 buluncak şekilde ölçeklendirilir.

4. $t_j = X_j w_j$ eşitliği **X'**in bileşeni $\mathbf{t_j}$, $\mathbf{w_j}$ ise ağırlık vektörü ile **X** 'in değerinin bir kombinasyonu olacak formda hesaplanır.

5. $\mathbf{t_j}$ bileşen değerinin **Y** 'yi modelleme değerini açıklayan $\mathbf{c_j}$ ağırlık vektörü ise $c_j = Y'_j t_j / (t'_j t_j)$ ile **Y** 'nin $\mathbf{t_j}$ üzerindeki regresyon değeri bulunur.

6. $\mathbf{c_j}$ vektörü norm değerine bölünerek boyu 1 bulunacak şekilde ölçeklendirilir. $\mathbf{c_j} / ||\mathbf{c_j}||$ şeklinde hesaplanır.

7. **Y** değerinin ilgili bileşeni $\mathbf{u_{j(yeni)}}$, $\mathbf{c_j}$ ağırlık vektörü ve **Y** 'nin doğrusal kombinasyonu ise $\mathbf{Y_j c_j / (c'_j c_j)}$ şeklinde hesaplanır.

8. İkinci adımda kullanılan $\mathbf{u_j}$ değeri ile yedinci adımda kullanılan $\mathbf{u_{j(yeni)}}$ değerileri arasında bir benzerliğin olup olmadığına bakılır. Bu benzerlik, iki vektörün fark normu $\mathbf{10^{-6}}$ gibi çok düşük bir değer olması sağlanır. Bu benzerlik oranı sağlanır ise bir sonraki adıma geçilerek algoritma sonlandırılır, yoksa yedinci adımda bulunan $\mathbf{u_{j(yeni)}}$ değeri ikinci adımdaki yerine yazılarak algoritmaya devam edilir.

9. **X** bileşeni $\mathbf{t_j}$ üzerine regresyonu, bileşen değerinin açıklayıcı değişken üzerindeki etki faktörünü gösteren yük vektörü $\mathbf{p_j}$, $X'_j t_j / (t'_j t_j)$ ile elde edilir.

10. **Y** bileşeni $\mathbf{u_j}$ üzerine regresyonu, bileşen değerinin bağımlı değişken üzerindeki etki faktörünü gösteren yük vektörü $\mathbf{q_j}$, $Y'_j u_j / (u'_j u_j)$ ile elde edilir.

11. Hem **X** hem de **Y** için bileşen değerleri ayrı ayrı hesaplandığında değerler arasında zayıf bir ilişki bulunduğu görülmektedir. Bu zayıf değer ilişkisinin kaldırılması için her bir bileşen değeri için **Y** 'nin bileşeni $\mathbf{u_j}$ 'nın **X** 'in bileşeni $\mathbf{t_j}$ üzerine regresyon değeri bulunan iç bir $\mathbf{b_a}$ katsayısı $b_a = u'_j t_j / (t'_j t_j)$ şeklinde hesaplanır.

12. Bulunan bileşen değerleri, yüklerin bağımlı değerleri ve açıklayıcı değişkenleri modellemenin yapılmasında kullanılır. Bileşenler açıklayıcı ve bağımlı değişken $X = TP'$ ve $Y = BTC'$ şeklinde modellenir. Bu aşamdan sonra algoritmanın bir sonraki bileşenlerini

elde etmek için kullanılacak olan $\mathbf{X_{j+1}}$ ve $\mathbf{Y_{j+1}}$ artık matrisleri $X_{j+1} \rightarrow X_j - t_j p'_j$ ve $Y_{j+1} \rightarrow Y_j - bt_j c'_j$ şekilde hesaplanmaktadır[11,12].

NIPALS algoritmasının 12 adımda olaşan çalışma akışına göre açıklayıcı ve bağımlı değişkenlerdeki değişimin büyük bir kısmı açıklık elde edilene kadar devam etmektedir. Yani X bileşenindeki değer matrisinin sıfır matrisi oluncaya kadar algoritma çalışır. Ayrıca, algoritma hesaplama sürecinde, ihtiyaç duyulacak en az sayıda bileşen değerini vermektedir[11].

3.1.2 Ridge Regresyonu (RR)

3.1.2.1 Teori

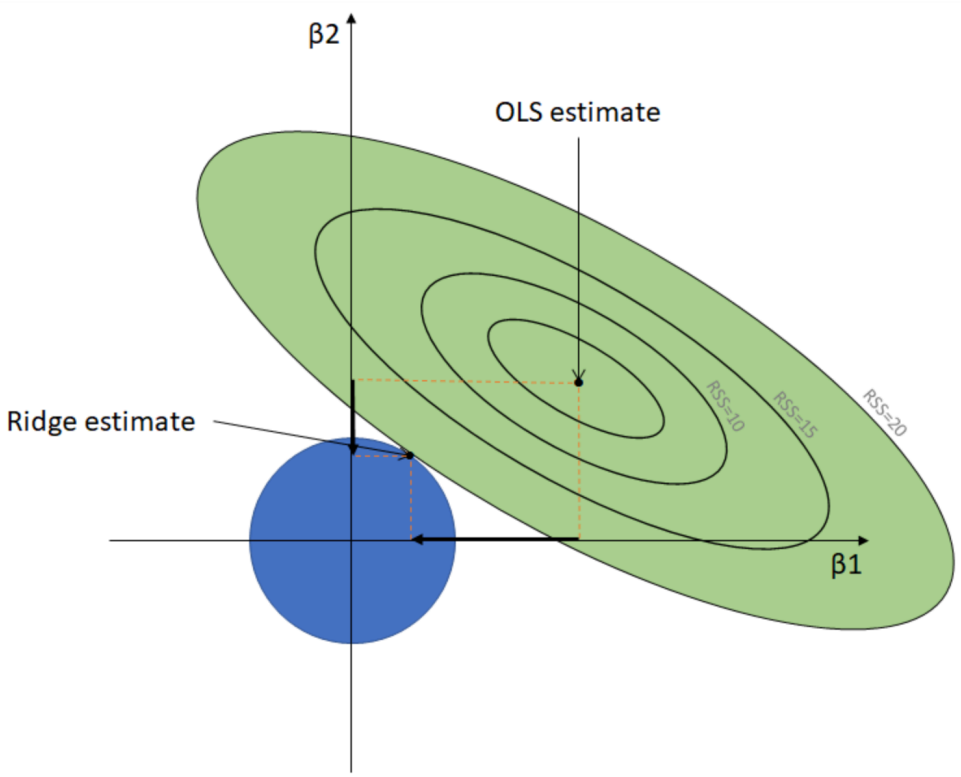
Ridge Regresyonu, 1970 yılında Hoerl ve Kennard tarafından bağımsız değişkenler arasındaki doğrusal yada doğrusala yakın ilişki biçimi olan çoklu doğrusal bağlantı problemini azaltmak için geliştirilen istatistik yöntemidir. Çoklu doğrusal bağlantı problemindeki en önemli olan nokta ise, regresyon katsayılarındaki kovaryans ve varyans sayılarının sonsuza doğru büyümesini sağlayan bir artış göstermesidir. RR yöntemi teknik olarak, X'X gibi bir korelasyon matrisinin köşegen değer elemanlarına, $k > 0$ yanlılık parametre değerlerini ekleyerek ve gerekli olan koşul sayısını azaltarak, çoklu bağlantı sorunundaki kovaryans ve varyans problemini çözmeyi hedeflemektedir[13]. Kısaca RR'nin amacını tanımlamak istersek, en küçük kareler (EKK) yöntemi ile bulunan kareler toplamını minimize etmek için regresyon katsayılarına bir ceza değeri uygulanmasını sağlayarak sonuç bulmaya çalışmaktadır. Ayrıca, EKK yönteminin yetersiz kalması dolayısıyla RR yöntem geliştirilmesi yapılmıştır[14]. Aşağıda maddeler halinde RR yöntem önerileri sıralanmıştır.

1. Modelin aşırı öğrenme yapısına karşı dirençli olmasını sağlar.
2. Yöntem yapısı olarak yanlıdır fakat varyans değeri düşüktür. (Bazen ise yanlı olan modeller daha çok tercih edilir.)
3. Çok fazla parametre olduğunda EKK'ya göre daha iyi sonuçlar verir.
4. Çok boyutluluk problemine karşı çözüm sunar.
5. En önemlisi ise çoklu doğrusal bağlantı sorununa karşı oldukça etkili bir yöntemdir.

6. Tüm değişkenlerle model kurar. İlgisiz değişkenleri modelden çıkarmaz, sadece katsayılarını sıfıra yakınlaştırır[13,14,15].

3.1.2.2 Matematiksel Model ve Algoritma

RR yönteminin teorik anlatımından sonra model ve algoritma anlatımlarını yapacağımız bu kısım da, önce RR ile EKK tahmin yapısının iki boyutlu şekil diyagramından ve matematiksel olarak hesaplanma formülünden bahsedilecektir. En son olarak ise RR algoritmasının EKK yönteminden farkılılaştığı ceza terimi katsayısı olan k (yada λ) adımları yazılacaktır.



Şekil 3.3: İki boyutlu uzayda RR ve OLS(EKK) kestiricisinin gösterimi[16]

İki boyutlu görselden de anlaşılacağı üzere, RR ile En Küçük Kareler Yöntemi (Ordinary Least Squares (OLS)) olan EKK arasındaki tahmin değerlerindeki farkın formunu açıklamaktadır.

RR , $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$ tahmin değerlerini yaklaşık sıfıra doğru çekerek verideki en iyi tahmin değerlerini elde etmiştir. Yani, $\mathrm{RSS(k)} = \sum_{i=1}^{n} (Y_i - X_i\hat{\beta})^2$ hata kareler toplamı minimum

yapılırken, aynı şekilde RR için ceza katsayı terimi olarak hesaplanan $k\sum_{j=1}^{p}\widehat{\beta}_j^2$ minimum olacak değerleri yakalayarak modeldeki veriyi oldukça iyi kuracak tahminler bulmaya çalışmaktadır. Şekildeki RR ile EKK tahminleri için örnek verilirse, $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2)'$ gibi iki boyutlu uzayda RR tahmini, $RSS(k)$ çemberlerinin $\beta_1^2 + \beta_2^2 = c_2^2(k)$ ile tanımlanan değerleri karşıladığı noktadır. Yukarıdaki şekil, çemberlerin eşit $RSS(k)$'lı β'ların değerler bilgisini göstermektedir. Ayrıca, RR ise lacivert çember içerisinde yer alan en optimum değerleri bulmaya çalışmaktadır[17].

Bu bağlamda RR matematiksel modeli aşağıdaki şekilde özetlenebilir;

$$RR = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + k\sum_{j=1}^{p}\beta_j^2$$

Şekil 3.4: RR matematiksel gösterimi

Formüldeki ceza terimi katsayısı olan k ayar parametresi başka çalışmalarda ise λ lambda simgesi olarakta gösterilmektedir.

RR algoritmasının ceza parametresi olan k yada λ adımları;

1. Ceza parametresi için bir faz aralığı tanımlanır.
2. Veri seti , $\{1,\ldots,n\} \setminus i$ ve $\{i\}$ içerecek şekilde örnek eğitim ve test setine sırasıyla bölünür.
3. Eğitim seti cross-validation yöntemi kullanarak her bir λ parametresi için RR ile tahminler yapılır. Aşağıdaki şekilde λ için RR tahmin hesaplama işlemleri şu şekilde yapılır;

$$\widehat{\beta}_i(\lambda) = (X_i^T X_i + \lambda I_{pp})^{-1} X_i^T Y_i \qquad \textbf{(3.2)}$$

   Ayrıca, hata varyansı tahmini ise: $\widehat{\sigma}_i^2(\lambda)$ olarak hesaplanır.

4. Her bir örnek test seti seçilip işlem yapılacak şekilde algoritmanın 1 ile 3.adımları arasında tekrarlanır.
5. Ceza parametresinin cross-validation yöntemi ile bulunan her bir test setinin tahmin performansları ortalaması aşağıdaki yöntemle yapılmaktadır;

$$\frac{1}{n}\sum_{i=1}^{n} log(L[Y_i X_i; \widehat{\beta}_i(\lambda)\widehat{\sigma}_i(\lambda)]) \qquad \textbf{(3.3)}$$

Bu hesaplama ile modelin, ceza parametresinin yeni veriler üzerindeki değerine karşılık gelen tahmin sonuçları bulunur.

6. Son olarak cross-validation yönteminin log olasılığını en üst düzeye çıkaran ceza parametresinin değeri, seçim değeri olarak bulunur[18,19].

RR ceza parametresinin seçilmesini sağlayan yukarıdaki adımlardan anlaşılacağı üzere, 1'den 6.adıma kadar çalışma modeli verilen algoritma adımları çalışılarak en optimum k yada λ değeri bulunmaktadır.

3.2 Doğrusal Olmayan Çoklu Regresyon

Birden fazla bağımsız değişken ile bir bağımlı değişken arasındaki ilişkinin doğrusal olmayan formunu incelemektedir. Doğrusal olmayan regresyon değişkenler arasındaki ilişki biçimini bir fonksiyonla modelleştirir. Aşağıda ki doğrusal olmayan çoklu regresyonun matematiksel gösterim örneklerinden biri ise polinom regresyon fonksiyonu şu şekilde ifade edilir;

$$Y_i = \beta_0 + \beta_1 X + \beta_2 X_i^2 + \ldots + \beta_h X_i^h + e_i \qquad \textbf{(3.4)}$$

3.2.1 Destek Vektör Regresyonu (SVR)

3.2.1.1 Teori

Destek Vektör Makineleri(SVM), 1995 yıllında Vladamir Vapnik tarafından büyük boyutlu ve az sayıda eğitim verisinden öğrenebilen yeni bir yaklaşım metodu olarak, sınıflandırma ve regresyon analizleri için kullanılması önerilen bir gözetimli makine öğrenme algoritmasıdır [20].

SVM matematiksel model algoritması, başlangıçta sınıflandırma problemleri için geliştirilmiştir. Sınıflandırma probleminde varolan iki sınıf verisinin, birbirinden ayırt edilebileceği en uygun tahmin fonksiyonunu bulmaya çalışmaktadır. Bir diğer ifade şekli ile mevcut verideki iki sınıfı en optimum şekilde ayırabileceği hiper-düzlem tanımlanmasına dayanmaktadır. Destek Vektör

Regresyonu (SVR) ise, sınıflandırma problemlerinin genelleştirilmiş formu olarak tanımlanabilir. SVR'nin temel fikri, veri setindeki eğtim hatasının genelleştirme üst sınırını azaltacak şekilde riski minimize eden bir tahmin fonksiyonu oluşturmaktır. Diğer bir deyişle, SVR modeli sürekli-değerler için çok değişkenli bir fonksiyonu tahmin etmektedir [21]. SVM'nin diğer algoritmalara göre üstünlüğü ise, sınıflandırma ve regresyon problemlerindeki lokal minimuma takılma gibi bazı dezavantajlı durumları bertaraf eden bir yöntem olmasıdır.

SVM algoritmasının avantajları aşağıdaki gibi sıralanabilir;

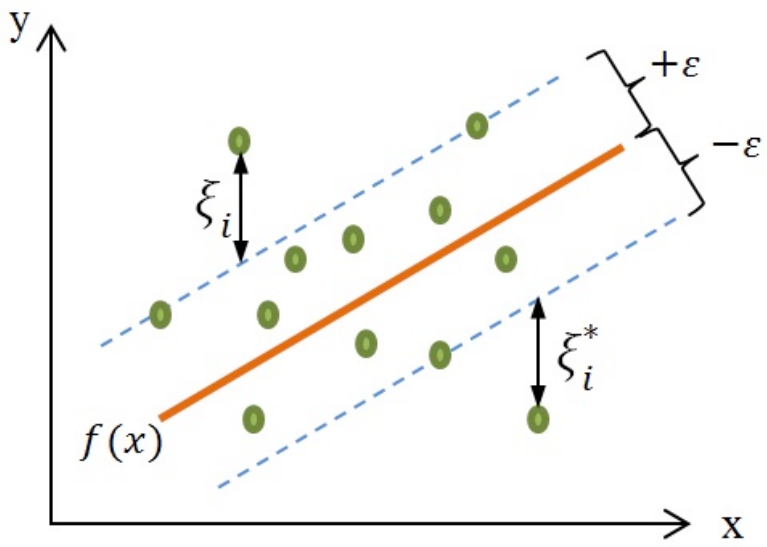
1. Modelin basit bir yapıya sahip olmasına rağmen güvenilir sonuçlar elde etmesi,
2. Güçlü fonksiyon genelleştirme becerileri ve optimum değerlere ulaşması,
3. Model çalışmalarında yüksek boyutlu uzaylarda etkili olması,
4. Model mimari tasarımında çok az kontrol parametresi kullanılması (gerekli olan sınırlı efor için)

Bu sebeplerden dolayı SVM, makine öğrenme tekniklerinden tahmin etme için kullanılan en popüler algoritmalardan birisi olmuştur [22].

Son olarak SVM kullanımına dair, genelleştirebilme özelliğinin çok yüksek olması, model yapısının güçlü ve uygulamalarda yüksek bir performans gösteriyor olmasından dolayı son yıllarda hava durumu, enerji piyasası, hisse fiyatı, ses ve görüntü tanıma gibi bir çok alanda tahmin çalışmaları için kullanılmakta olduğu belirtilmelidir [23]. Ayrıca, SVM modelinin tahmin edilen değerler için güvenilir bir araç olması ve finansal zaman serilerinin tahmin edilmesinde de başarılı sonuçlar elde ettiği belirtilmiştir [24].

3.2.1.2 Matematiksel Model ve Algoritma

Teorik kısımda anlatıldığı üzere SVM algoritmalarının regresyon problemlerinde başarılı sonuçlar elde etmesi ve tahmin problemlerinde sıkça kullanılması, SVR algoritmalarını tercih etmemizi sağlamaktadır. Bu bölümdeki çalışmamızda SVR model yapısı üzerinde durulacaktır. İlk olarak, SVR matematiksel modelin temelini oluşturan ve anlaşılması daha basit olan SVR'nin doğrusal matematiksel fonksiyonundan bahsedilecektir. İkinci olarak SVR ağ yapısı ayrıntılı incelenecek ve son olarak ise model çalışmamız için kullanılacak olan algoritmanın yapısı ve adımlarından bahsedilerek bu bölüm sonlandırılacaktır.



Şekil 3.5: Doğrusal SVR örnek gösterimi[25]

Doğrusal SVR'nin matematiksel olarak gösterimi olan örnek bir $f(x) = wx + b + \epsilon$ fonksiyonu ile doğrusal ya da doğrusal olamayan bir eğri ile fonksiyon modeli bulunur iken, aşağıdaki minimizasyon problemi ile en optimum fonksiyon modeli bulunmaya çalışmaktadır.

Minimizasyon denklemi , $\frac{1}{2}||w||^2 + C\sum_{i=1}^{m}(\epsilon_i + \epsilon_i^*)$ **(3.5)**

olacak sekilde aşağıdaki belli kısıtlar çerçevesinde;

$\epsilon_i, \epsilon_i^* \geq 0 \quad i = 1,...,m \quad \epsilon = Epsilon, \epsilon^* = Kisi$

$$y_i - (w * x_i) - b \leq \epsilon + \epsilon_i \quad \textbf{(3.6)}$$

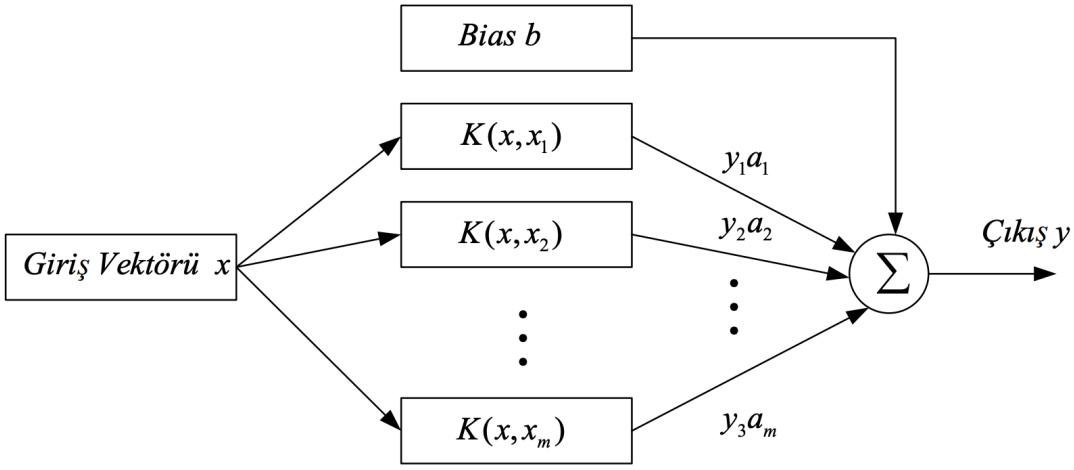
$$(w * x) + b - y_i \leq \epsilon + \epsilon_i^* \quad \textbf{(3.7)}$$

(3.6) ve (3.7) denklem kısıtlarına göre en optimum regresyon denklemi bulmaya çalışırken, gerçek değerler ile tahmin edilen değerler arasındaki farklar, regresyon eğrisinin iki yönünde belirli bir epsilon ve kisi değerlerinden daha uzakta olmayacak şekilde bulunur. Kısıtların denklem üzerindeki etkisini bu şekilde özetleyebiliriz. Ayrıca, (3.5)'te verilen minimizasyon denklemindeki ***C*** parametresi, karmaşıklık ya da ceza değerleri olarak ifade edilir. Bu ceza terimi, epsilon ve kisi artıkları üzerinde bir kontrol mekanizması işlevi görmüş olmaktadır.

Genel olarak ifade edilirse, şu ana kadar anlatılan matematiksel model kısıtları ve yukarıdaki şekil 5' te gösterilmekte olan regresyon eğrisini bulmaya çalışırken ki temel amaç, belirli bir marjın

aralığına maksimum noktayı en küçük hata ile alabilmesini sağlayacak şekilde bir doğru ya da eğri belirlemektir.

SVR modelin bir diğer önemli kısmı olan ağ yapısını inceleyeceğiz,



Şekil 3.6: SVR model ağ yapısı[26]

Model ağ yapısına göre doğrusal olmayan SVR tahmin fonksiyonu,

$$f(x_i) = w^t \,.\, K(x_{x,x_i}) + b \qquad \textbf{(3.8)}$$

olarak gösterilir. Bu aradaki temel amaç, x_i girişine çıktı y_i'nin bağımlılığını ifade eden (3.8) denklem fonksiyonu tanımlamaktır. Fonksiyonun temel bileşenleri olan w ağırlık vektörü , b eğilimi katsayısı ve $K(x, x_i)$ çekirdek fonksiyon ile çıkış değeri olan y tahmini yapılır. Çekirdek fonksiyonu için seçilecek olan metot ise,

1. Model olarak kullanacağımız yöntem de doğrusal olmayan regresyon denklemi oluşturmak,
2. Fonların fiyat tahmini için veri setimizin birden fazla bağımsız ve bir bağımlı değişken yapısından oluşması,

Yukarıdaki sebeplerden dolayı, SVR modeli uygulanırken Radial Basis Function (RBF) çekirdek fonksiyonu kullanılacaktır.

Son olarak, SVR algoritma akış sürecinden kısaca bahsedilirse;

1. **Adım:** Model için eğitim veri seti seçilir.
2. **Adım:** Bağımsız değişken değerleri normalizasyon yöntemi ile verileri [-1, 1] aralığına getirilecek şekilde hazırlanır.
3. **Adım:** Giriş verileri üzerinden çekirdek fonksiyon hesaplama işlemleri yapılır. Modelimiz için seçtiğimiz RBF fonksiyonu ile hesaplamalar gerçekleştirilir.
4. **Adım:** (3.5)'teki denklem ile optimizasyon probleminin çözümü yapılır.
5. **Adım:** Model tahmin fonksiyonu için regresyon denklemi elde edilir.
6. **Adım:** Girdi verileri ile 5.Adım'da bulunan regresyon denklemi kullanılarak bağımlı değişken tahmin sonuçları elde edilir [27].

Yukarıdaki algoritmanın çalışma akışının özetinden anlaşılacağı üzere, 6 adımda SVR model için tahmin işlemi bu şekilde gerçekleştirilmektedir.

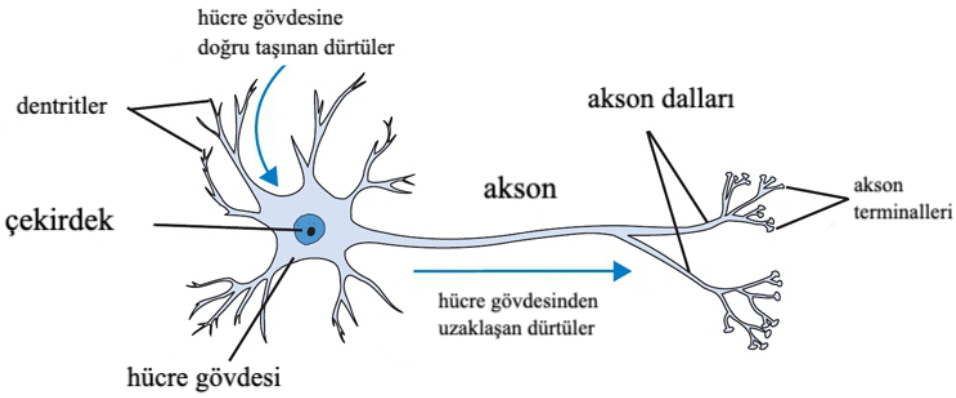
3.2.2 Yapay Sinir Ağları(YSA)

3.2.2.1 Teori

İlk yapay sinir ağ modeli, 1943 yılında Nöropsikiyatrist Warren McCulloch ve bilim adamı Walter Pitts tarafından "A Logical Calculus of The Ideas Immanent In Nervous Activity" adıyla yayınladıkları makale çalışması ile önerilmiştir [28]. Ancak kendi dönemlerinin kısıtlı imkanları nedeniyle, yapay sinir ağ modeli alanında çok fazla ilerleme kaydedilememiştir. Bundan sonraki yıllarda ise çalışmalar artarak devam etmiştir. Fakat sinir ağ modeli için en önemli süreç, 1969'da Marvin Minsky ve Seymore Papert tarafından bu konuda bir kitabın yayınlanmış olmasıdır. Bu kitap, YSA çalışmaları ile ilgili kaygı duyulan etik sorunları ortadan kaldırmış fakat yeni gelişen teknolojilere doğru giden yolu derinleştirerek YSA çalışmalarına olan ilginin azalmasına neden olmuştur. Böylelikle, YSA çalışmalarında 9 yıllık (1960-1969) altın çağı olarak nitelendirilen ilerleme süreci sonlandırılmış, YSA için karanlık çağ olarak belirtilen bir sürecin başlangıcı olmuştur. Daha sonraki süreçte YSA çalışmaları ile ilgili yapılan faaliyetlerde ilk gözle görülür önemli gelişmeler ise 1990'lı yıllara dayanmaktadır [29]. YSA çalışmalarının kısa tarihsel arka

planından sonra YSA'nın teorik bağlamından ve çalışma mekanizmasından bahsetmemiz gerekmektedir.

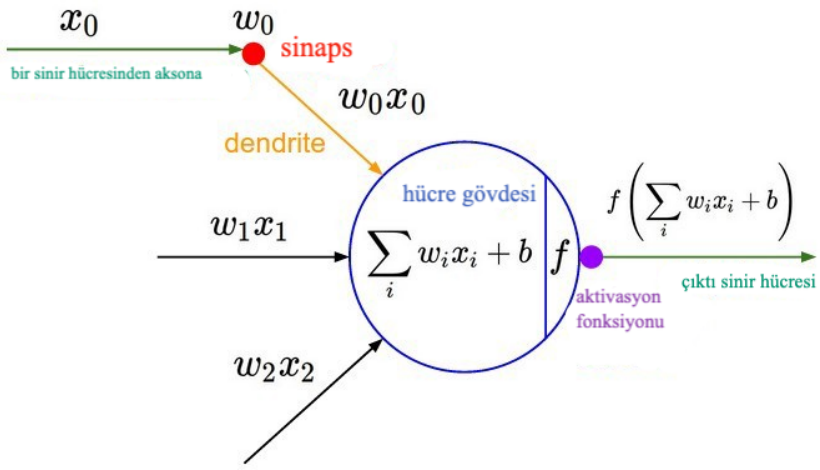
İnsan beyninin çalışma mekanizması üzerine uzun yıllar boyunca bilim adamları tarafından birden fazla araştırma ve çalışma yapılmıştır. Yapılan çalışmaların temel argümanını oluşturan soru ise beynin çalışma mekanizmasının nasıl mevcut gelişen teknoloji altyapısı ile taklit edilebilirliği sağlayabileceği tartışması olmuştur. Beyin sinir hücrelerinin bu kadar mükemmel organize olması ve aynı şekilde çok büyük bir karmaşıklığı kendi içerisinde barındırması, yapay olarak taklit edilebilir mi sorusu bilim insanları için büyük bir merak konusu da olmuştur. Çalışmaların temel felsefesini oluşturan taklit edilebilirlik sorusu, ilk olarak bilim insanlarının sinir hücresinin yapısı üzerinden durmasına ve ilk matematiksel model çalışmalarını bu şekilde yapmasına zemin hazırlamıştır. Bu bağlam bir sinir hücresinin yapısını gösteren şekil 3.7 ve yapay bir sinir hücresinin matematiksel model gösterimi olan şekil 3.8'deki yapıları karşılaştırarak benzerlik ilişkisi üzerinden irdelenebilir.



Şekil 3.7: Sinir hücresi biyolojik gösterimi[30]

Şekil 3.7'de sinir hücresinin temel yapıları olan sinapslar, akson, soma (hücre çekirdeği) ve dentrite'lerden oluşmaktadır. Sinir hücresinin çalışma yapısı; dentrit'lerden alınan sinyalleri somaya (çekirdeğe) iletir. Soma, iletilen sinyalleri çekirdekte toplar, bu toplama işlemi dentrit'lerden alınan sinyallerin belli bir eşik değeri kapsamında toplanmasıyla gerçekleştirilir. Toplanılan sinyalleri akson'lara iletirler, akson ise aldıklarını diğer hücrelere aktarır. Bu aktarım işleminde önce sinaps'lar bir ön işleme tabi tutularak, gönderilen sinyalleri belirli bir eşik değerine getirerek diğer

hücrelere aktarım işlemi yapılır. Bu şekilde bir sinir hücresinde bilgi aktarım süreci tamamlanmış olur [31].



Şekil 3.8: Yapay sinir ağının matematiksel gösterimi [30]

Beyin sinir hücresinin şekil 3.7 deki işleyişinden sonra benzer çalışma akışı olan şekil 3.8'de, yapay sinir hücresinin dışardan gelen *x* girdilerini belli *w* ağırlık değerlerine göre bir toplama fonksiyonu ile işlem yapılarak, bir sonraki adım olan aktivasyon fonksiyonundan geçirip bir çıktı üretilmesini sağlar. Üretilen çıktı ağın diğer bağlantıları üzerinden hücrelere gönderilerek bilgi yani tahmin işlemi gerçekleştirilmiş olur[31].

Aşağıdaki tablo 3.1'deki bir sinir hücresi ile yapay bir sinir hücresinin ortak tanımlarından yola çıkarak iki hücre modelinin ağ yapısını oluşturan terminolojileri özetlenebilir.

Tablo 3.1 : Sinir hücresi ile yapay sinir ağı ortak terminolojisi

Sinir Hücresi Sistemi	Yapay Sinir Ağı
Nöron	İşlem Elemanı
Dentrit	Toplama Fonskiyonu
Hücre Çekirdeği	Aktivasyon Fonksiyonu
Akson	Çıkış Elemanı
Sinaps	Ağırlıklar

Genel olarak YSA tanımlarsak, beynin çalışma mekanizmasının bir işlevini yerine getirmek için yapılan işlemin modellenmiş formu olarak tanımlanabilir. YSA, sinir hücrelerinin kendi aralarında farklı şekillerde bağlanmasından oluşur ve genel olarak katmanlar biçiminde düzenlenmektedir. Teknolojik alt yapısı olarak ise, elektronik devrelerle ya da bilgisayar ortamında yazılımsal olarak gerçekleştirilir. Ayrıca, beynin bilgiyi işleme metodu olarak YSA, modelin öğrenme sürecinde veriyi işleme, depolama ve genelleştirme özleliklerine sahip paralel dağılmış bir işlemcidir. Son olarak, Turing makineleriyle temeli atılan yapay zeka çalışmalarının en çok araştırma yapılmış konularının başında "Yapay Sinir Ağları" gelmektedir. Temel olarak YSA'yı özetlersek, tamamen insan beynini modelleyerek geliştirilmesi yapılmış bir teknolojidir[32].

YSA'lar bir çok alanda uygulama sahasına sahip olmuştur. Başlıca kullanım alanları aşağıdaki gibi kısaca maddeler halinde gruplandırılabilir:

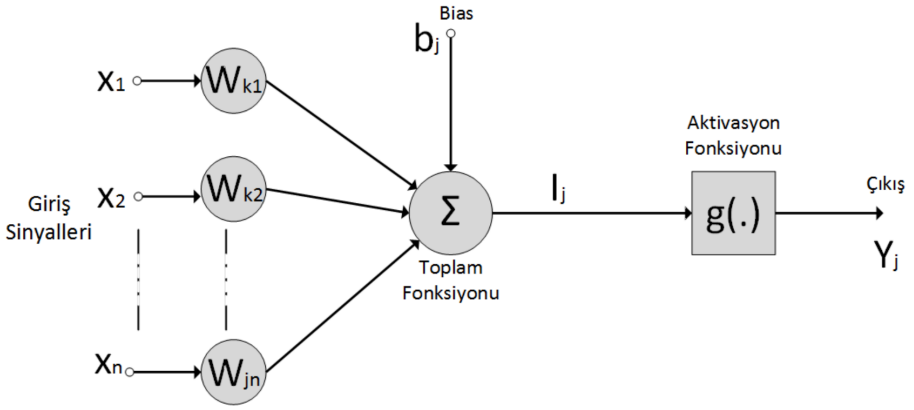
- Bir araya getirme / toplama
- Gruplama / İlişkilendirme
- Sınıflandırma
- Örüntü tanıma
- Regresyon ve genelleme
- Optimizasyon
- Tahmin uygulamaları,

Benzeri alanlar da olmak üzere, pek çok çalışmada kullanılmaktadır. Yapılan çalışmalar incelendiğinde, YSA'ların büyük boyutlu, kompleks, net olmayan, eksik, hatalı olma ihtimali yüksek verilerin olduğu ve probleme dair herhangi bir matematiksel model ya da algoritmanın bulunmadığı durumlarda da oldukça yaygın bir şekilde kullanıldıkları görülmektedir[33].

3.2.2.2 Matematiksel Model ve Algoritma

Bu kısımda ilk olarak YSA'nın, bir yapay sinir hücresinin matematiksel modeli üzerinde durulacak, daha sonra yapay sinir hücrelerinden oluşan ağ yapısı incelenecektir. Son olarak da, bu tez çalışmasında ana sorumuz olan fonlar için tahmin işleminin nasıl yapılacağı konusu üzerine, birden fazla bağımsız değişkenden bir bağımlı değişken tahmini yapılacak olan model çalışması için ileri beslemeli ağ algoritmalarından çok katmanlı algılayıcı (MLP) ağ yapısından ve öğrenme adımlarından bahsedilecektir. MLP'nin tercih edilme sebebi, YSA regresyon problemlerindeki

tahmin çalışmalarında daha çok tercih edilmesi ve araştırmanın model yapısına daha uygun bir algoritma olmasından kaynaklanmaktadır[32,34].



Şekil 3.9: Yapay sinir ağı[29]

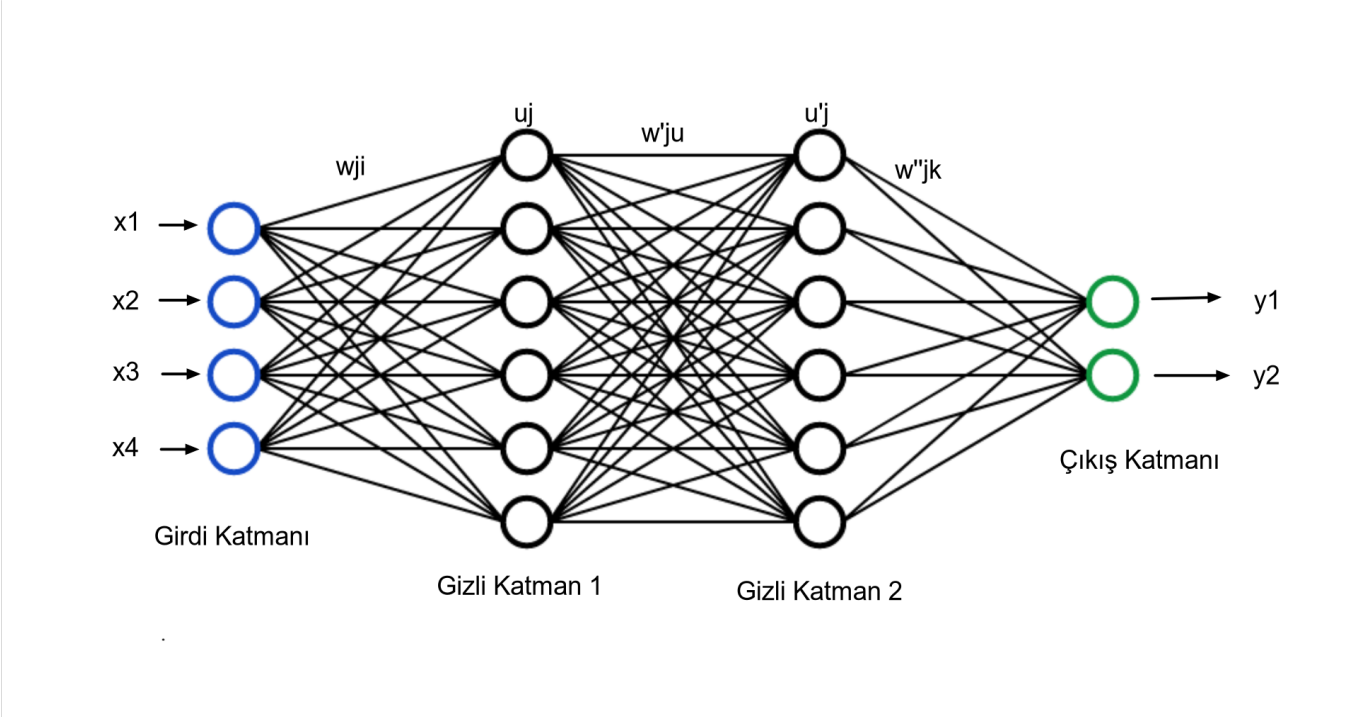
Şekil 3.9'da ki gibi bir yapay sinir hücresindeki tahmin işlemi, giriş sinyalleri $(x_1, x_2 \dots x_n)$ olan bağımsız değişken değerlerinin, her bir girdi değerine karşılık bir w_k ağırlık ile çarpılır. Bu çarpma işleminde her bir değerin çıktıya olan etkisinin kontrölünü sağlanmış olur. Bir sonraki adımda bulunan bu değerler, toplam fonksiyonu ile işlem yapılarak aktivasyon fonksiyonuna iletilir. Aktivasyon fonksiyonu bir dönüştürme işlemine tabi tutularak bir çıktı değeri (y_j) üretmiş olur. Bu şekilde tahmin işlemini gerçekleştirilmiş olmaktadır. Tahmin çalışmalarında çokça kullanılan aktivasyon fonksiyonları ise;

Sigmoid: Sınıflandırma problemlerinde sıklıkla kullanılır. Giriş verilerini 0 ile 1 aralığındaki değerlere dönüştürür.

Tanjant Hiperbolik: Giriş verilerini -1 ile 1 aralığındaki değerlere dönüştürmektedir. En iyileme problemlerinin daha anlaşılır bir forma gelmesine katkı sağlar.

Doğrultulmuş Dorusal Ünite (ReLU): Girdi verilerini [0 , ∞] aralığındaki değerlere dönüştürür. Derin öğrenme problemlerinde diğer fonksiyonlara göre çok daha hızlı eğitilmektedir.

Sızdırılmış Doğrultulmuş Doğrusal Ünite (Leaky ReLU): ReLU fonksiyonun dezanvatajı olan negatif değerleri sıfıra eşitlemesi problemine çözüm olarak, negatif değerler yerine aktivasyon fonksiyonuna bir eğim eklenmesi önerilmiştir[35].



Şekil 3.10: Çok katmanlı yapay sinir ağı

Şekil 3.9 daki yapay sinir hücreleri bir araya gelerek şekil 3.10 daki gibi yapay sinir ağını oluştururlar. Ağ yapısındaki katman bilgileri için tanımları kısaca şu şekildedir;

Girdi Katmanı: YSA modeline giriş değerleri(x_1, x_2, x_3, x_4) olarak verilen bilgiler bu katmanda bulunur. Her bir değerin bir ağırlık (w_{ji}) bilgisi mevcuttur. Gizli katmanlar arasındaki bağlantıyı sağlar. Ayrıca, ağırlık bilgisi çıkış değeri için bir kontrol etkisi de sağlamış olur.

Gizli Katmanı: Girdi değerlerini ağrılık bilgilerini (w'_{ju}) kullanarak çıktıya dönüştürme işleninin yapıldığı katmandır. Problemin zorluk derecesine göre gizli katman sayısı değişmektedir. Şekil 3.10 da ki gizli katman sayı 2 olup, nöron sayısı ise her bir katmanda 6 adet bulunmaktadır. Nöronlar öğrenilen bilgileri kendi yapısında tutan birimler olarak görülmektedir.

Çıktı Katmanı: Verilen girdi değerlerinin, gizli katmandaki işlemden sonra elde edilen çıktı değerileri (y_1, y_2) olarak üretildiği katmandır. Bu katmanda beklenen değer ile gerçek değer arasındaki fark bulunur ve bulunan değer seçilecek olan hata hesaplama fonksiyonu ile işlem

yapılarak ağın performansı değerlendirilir. Bulunan sonucun performansına göre ağrılıkların (w''_{jk}) güncellemesi yapılır. Ağırlık güncellemesi, YSA öğrenme sürecini belirtmektedir. Ağırlık güncelleme için farklı optimizasyon metotları bulunmaktadır. Çalışmamızda kullanacağımız optimizasyon fonksiyonu ise ileri beslemeli ağ algoritmaları olacaktır[34,35].

YSA modelin matematiksel olarak ifade edilmesi ise;

$$h_k(x) = g(\beta_{0k} + \sum_{j=1}^{p} x_j \beta_{jk}) \quad \textbf{(3.9)}$$

$$g(u) = \frac{1}{1 + e^{-u}} \quad \textbf{(3.10)}$$

Şekil 3.9' daki bir sinir ağ hücresinin, 3.9'da verilen matematiksel denklem formülüne göre çıktı üretme işleminin yapılması, 3.10 denklemine (örnek olarak sigmoid fonksiyonu verilmiş) benzer bir dönüştürme işlemi yapılarak bir sinir ağ hücresinin çıktı ile ilişkilendirilerek bilginin üretilmesini sağlar.

$$f(x) = \gamma_0 + \sum_{k=1}^{H} \gamma_k h_k \quad \textbf{(3.11)}$$

Hücrenin bilgi üretiminden sonraki adımda bilginin diğer katmanla ilişkilendirilmesi 3.11 de verilen doğrusal bir denklem ile çıktıya bağlama işlemini gerçekleştirir. Bir hücrenin bilgi üretimi ve çıktıyla ilişkilendirilme denklemlerinden hareketle, nihai olarak optimize edilmiş YSA problem çözme denklemi 3.12'deki gibi formülüze edilebilir.

$$\sum_{i=1}^{n} (y_i - f_i(x))^2 + \lambda \sum_{k=1}^{H} \sum_{j=0}^{P} \beta_{jk}^2 + \lambda \sum_{k=0}^{H} \gamma_k^2 \quad \textbf{(3.12)}$$

3.12'deki matematiksel denkleme dair son olarak belirtilmesi gereken nokta, denklemdeki cezalandırma katsayısı olan λ parametreleri genelde 0 ile 1 arasında değerler almaktadır.

Matematiksel model anlatımdan sonra YSA modelinin öğrenme algoritma adımları ise şu şekildedir;

1. Adım: Test veri seti seçiminin, modelin eğitim setinden öğrenme performansı en iyi olacak şekilde gerçekleştirilmesi, ağın eğitim setinden öğrenme süreci tamamlandıktan sonra test veri seti ile ağın öğrenme performası ölçülür. Bu şekilde yeni veriler karşısındaki model başarısı ağın iyi eğitilip eğitilmediğini gösterir.

2. Adım: Modelin topolojik formunun belirlenmesi, ağın öğrenilmesi gereken çıktı değeri için topolojik yapısını netleştirir. Buna göre girdi verileri, gizli katman sayısı, çıktı katman sayısı gibi ağın öğrenme yapısı bu adımda kararlaştırılır.

3. Adım: Ağın parametre yapılarının belirlenmesi, ceza katsayısı, aktivasyon fonksiyonu gibi metot ve değerler belirlenir.

4. Adım: Girdi verileri için ağırlıkların başlangıç değerlerinin belirlenmesi, verilerin bir diğer katman ile ilişkilendirilmesi sürecinde her bir girdi değerinin ağırlıkları başlangıç değerleri için rastgele atanır. Daha sonra öğrenme sürecine göre uygun ağırlık değerlerini ağın kendisi optimize eder.

5. Adım: Öğrenme sürecinde ağırlıkların işlenmesi, ağın öğrenme aşamasında ağırlık değerleri belirli bir kurala göre ağa giriş değerleri ile işleme girer.

6. Adım: Model öğrenmesi için ileri hesaplama işlemlerin olması, verilen girdi ile beklenen çıktı değerleri için hesaplamalar yapılır.

7. Adım: Tahmin edilen çıktı değerleri ile gerçek değerler arasındaki farkın, hata değerlerinin hesaplanması ile bu adımda gerçekleşir.

8. Adım: Ağırlık değerlerinin güncellemesi, bu adımda hata değerlerinin hesaplanmasına göre bulunan hata değerinin optimize edilmesi için ağırlık değerleri güncellenir.

9. Adım: Model öğrenme sürecinin tamamlama aşaması, tahmin edilen ile gerçek değer arasındaki farkın hata değeri kabul edilir bir aşamaya gelinceye kadar ileri beslemeli sinir ağı metodu ile öğrenme sürecine devam eder[34,36].

YSA modelinin eğitilme/öğrenme süreci yukarıdaki algoritmanın öğrenme akışından anlaşılacağı üzere, 9 adımda tahmin işlemini gerçekleştirmiş olacaktır.

Kaynakça

1. Orhunbilge, N. (2002), Uygulamalı Regresyon ve Korelasyon Analizi, İstanbul, İ.Ü. İşletme Fakültesi.

2. GÖK, M. (2017). MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE AKADEMİK BAŞARININ TAHMİN EDİLMESİ. Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji. 5(3): 148-139.

3. Abdi, H. (2003). Partial Least Squares (PLS) Regression, Lewis-Beck M., Bryman, A., Futing T. (Eds.) Encyclopedia of Social Sciences Research Methods. Thousand Oaks (CA): Sage.

4. Tobias, R.D. (1995). An Introduction to Partial Least Squares Regression. UGI Proceedings, Orlando 2-5 April, pp. 1-8.

5. Polat, E., Günay, S. (2009). Kısmi En Küçük Kareler ve Bir Uygulama. Ondokuz Mayıs Üniversitesi, Fen Edebiyat Fakültesi, İstatistik Bölümü, VI. İSTATİSTİK GÜNLERİ SEMPOZYUMU BİLDİRİLER KİTABI. S,438.

6. Höskuldson, A. (1988). PLS Regression Methods. Journal of Chemometrics, 2: 211-228.

7. Marten, H. ve Naes, T. (1989). Multivariate Calibration. John Wiley & Sons.

8. Wu, J. (2015). Research on Several Problems in Partial Least Squares Regression Analysis. The Open Electrical & Electronic Engineering Journal, 8, 754-758

9. Bulut, E., Alma, G., Ö. (2011). Kısmi En Küçük Kareler Regresyonu Yardımıyla Optimum Bileşen Sayısını Seçmede Model Seçme Kriterlerinin Performans Karşılaştırılması. İstanbul Üniversitesi, İktisat Fakültesi, Ekonometri ve İstatistik Dergisi, Sayı:15 , 38-52.

10. Rosipal, R., Trejo, L. J. (2001). Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space. Journal of Machine Learning Research 2 , 97-123.

11. Bulut, E., Alın, A. (2009). Kısmi En Küçük Kareler Regresyon Yöntemi Algoritmalarından Nipals ve PLS - Kernel Algoritmalarının Karşılaştırılması ve Bir Uygulama. Dokuz Eylül Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi, Cilt:24, Sayı:2, ss.127-138.

12. Ümit, Ö.A., Bulut, E. (2013). TÜRKİYE'DE İŞSİZLİĞİ ETKİLEYEN FAKTÖRLERİN KISMİ EN KÜÇÜK KARELER REGRESYON YÖNTEMİ İLE ANALİZİ: 2005-2010 DÖNEMİ. Dumlupınar Üniversitesi Sosyal Bilimler Dergisi. (37): -.

13. Demirci, A.M. (2014). Ridge Regresyonda Sapma Parametresi Olan k'nın Belirlenmesinde Genetik Algoritma Yaklaşımı, Yüksek lisans Tezi, Ondokuz Mayıs Üniversitesi, Fen Bilimleri Enstitüsü, Samsun.

14. Bağcı, İ. (2017). Kısmi En Küçük Kareler Yönteminin Simülasyon Verileri ile Diğer Yöntemlerle Karşılaştırılması, Yüksek Lisans Tezi, Muğla Sıtkı Koçman Üniversitesi, Fen Bilimleri Enstitüsü, Muğla.

15. Polat, E. (2009). Kısmi En Küçük Kareler Regresyonu, Yüksek Lisans Tezi, Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

16. Kim, K. (2019). Ridge Regression for Better Usage. Erişim Tarihi:…

17. Şahingöz, T.M. (2019). Regresyon Analizinde Yanlı Tahmin Yöntemleri, Yüksek Lisans Tezi, Muğla Sıtkı Koçman Üniversitesi, Fen Bilimleri Enstitüsü, Muğla.

18. Cule, E. &, Lorio, D.M. (2013). Ridge Regression in Prediction Problems: Automatic Choice of the Ridge Parameter. *Genetic Epidemiology, 37*, 704 - 714.

19. Wieringen, W.N. (2015). Lecture notes on ridge regression. *arXiv: Methodology*.

20. Ekinci, E.M. (2017). Destek Vektör Regresyon ile Hava Kirliliği Tahmini, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Eskişehir Osmangazi Üniversitesi, Eskişehir.

21. Arat, M.M. (2014). Destek Vektör Makineleri Üzerine Bir Çalışma, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü , Hacettepe Üniversitesi, Ankara.

22. Tolun, S. (2008). Destek Vektör Makineleri: Banka Başarısızlığının Tahmini Üzerine Bir Uygulama, Doktora Tezi, Sosyal Bilimler Enstitüsü, İstanbul Üniversitesi, İstanbul.

23. Smola, A.J., Scholkopf, B. (2004). A tutorial on support vector regression, Statistics and Computing 14, 199–222.

24. Trafalis, T.B, Ince, H. (2000). Support Vector Machine for Regression and Applications to Financial Forecasting. In: IJCNN 2000: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks: Volume 6 edited by Shun-Ichi Amari, et al., içinde 6348, IEEE Computer Society.

25. Chanklan, R., Kaoungku, N., Suksut, K., Kerdprasop, K., & Kerdprasop, N. (2018). Runoff Prediction with a Combined Artificial Neural Network and Support Vector Regression. *International Journal of Machine Learning and Computing, 8*, 39-43.

26. Uçak, K. (2012). Destek Vektör Regresyonu İle Pıd Kontrolör Tasarımı, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, İstanbul Teknik Üniversitesi, İstanbul.

27. Chen, Y., Tan, H. (2017). Short-term Prediction of Electric Demand in Building Sector Via Hybrid Support Vector Regression, Applied Energy, Volume 204, Pages 1363-1374.

28. McCulloch, W. S., Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4), 115-133.

29. Polat, E. (2019). Türkiye'nin Aylık Elektrik Tüketiminin Yapay Sinir Ağlarıyla Tahmini, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Yıldız Teknik Üniversitesi, İstanbul.

30. Akpınar, B. (2019). Görüntü Sınıflandırma için Derin Öğrenme ile Bayesçi Derin Öğrenme Yöntemlerinin Karşılaştırılması, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Afyon Kocatepe Üniversitesi, Afyonkarahisar.

31. Adıyaman, F. (2007). Talep Tahmininde Yapay Sinir Ağlarının Kullanılması, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, İstanbul Teknik Üniversitesi, İstanbul.

32. Ataseven, B. (2013) Yapay Sinir Ağları ile Öngörü Modellemesi. Dergipark, s.101-115, İstanbul.

33. Akkurt, A. (2005). Yapay Sinir Ağları Ve Türkiye Elektrik Tüketimi Tahmin Modeli, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, İstanbul Teknik Üniversitesi, İstanbul.

34. Akdoğan, E. Ders Notları, Mekatronik Mühendisliği Uygulamalarında Yapay Zekâ, Yapay Sinir Ağları,[Alıntı Tarihi: 22.05.2020]. http://ytubiomechatronics.com/wp-content/uploads/2017/10/YSA.pdf

35. Kuş, Z. (2019). Mikrokanonikal Optimizasyon Algoritması ile Konvolüsyonel Sinir Ağlarında Hiper Parametrelerin Optimize Edilmesi, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Fatih Sultan Mehmet Vakıf Üniversitesi, İstanbul.

36. Çayıroğlu, İ. İleri Algoritma Analizi-5 Yapay Sinir Ağları. Karabük Üniversitesi Mühendislik Fakültesi. [Alıntı Tarihi: 23.05.2020]. http://www.ibrahimcayiroglu.com/dokumanlar/ilerialgoritmaanalizi/ilerialgoritmaanalizi-5.hafta-yapaysiniraglari.pdf