**GÖRÜNTÜ MADENCİLİĞİ VE GÖRÜNTÜ İŞLEMEYE DAYALI MEYVE RESİMLERİNİN SINIFLANDIRILMASI ÜZERİNE BİR UYGULAMA**

**Arash MANZOORİ**

**Yüksek Lisans Tezi**
**Bilgisayar Mühendisliği Anabilim Dalı**
**Yrd. Doç. Dr. Tolga AYDIN**
**2016**

**ATATÜRK ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**

**YÜKSEK LİSANS TEZİ**

# GÖRÜNTÜ MADENCİLİĞİ VE GÖRÜNTÜ İŞLEMEYE DAYALI MEYVE RESİMLERİNİN SINIFLANDIRILMASI ÜZERİNE BİR UYGULAMA

**Arash MANZOORİ**

**BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**

**ERZURUM**
**2016**

T.C.
ATATÜRK ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü Müdürlüğü

TEZ ONAY FORMU

GÖRÜNTÜ MADENCİLİĞİ VE GÖRÜNTÜ İŞLEMEYE DAYALI MEYVE RESİMLERİNİN SINIFLANDIRILMASI ÜZERİNE BİR UYGULAMA

Yrd. Doç. Dr. Tolga AYDIN danışmanlığında, Arash MANZOORİ tarafından hazırlanan bu çalışma, 01/12/2016 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Bilim Dalı'nda Yüksek Lisans tezi olarak **oybirliği / ~~oy çokluğu~~ (.../...)** ile kabul edilmiştir.

Başkan: Yrd. Doç. Dr. Tolga AYDIN İmza :

Üye : Yrd. Doç. Dr. Barış ÖZYER İmza :

Üye : Yrd. Doç. Dr. Mustafa CANIM İmza :

Yukarıdaki sonuç;

Enstitü Yönetim Kurulu'nun 29./12./2016 tarih ve ...50.../...32..... nolu kararı ile onaylanmıştır.

**Prof. Dr. Cavit KAZAZ**
**Enstitü Müdürü**

**Not:** Bu tezde kullanılan özgün ve başka kaynaklardan yapılan bildiriş, çizelge, şekil ve fotoğrafların kaynak olarak kullanımı, 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.

# ÖZET

Yüksek Lisans Tezi

**GÖRÜNTÜ MADENCİLİĞİ VE GÖRÜNTÜ İŞLEMEYE DAYALI MEYVE RESİMLERİNİN SINIFLANDIRILMASI ÜZERİNE BİR UYGULAMA**

Arash MANZOORİ

Atatürk Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd. Doç. Dr. Tolga AYDIN

Görüntüleme ve görüntülerin depolanmasında yapılan gelişmeler, görüntü veri tabanı ve ayrıntılarının büyümesine neden olmuştur. Görüntü madenciliği, görüntüdeki gizli bilgileri ayıklama, görüntü verileri arasındaki ilişkileri bulma ve resimde net olarak depolanmayan modelleri çıkarma konularını tartışır. Görüntü madenciliği, özel yapay görme, görüntü işleme, görüntü alma, veri madenciliği, makine öğrenme, veri tabanları ve yapay zekâ üzerine disiplinler arası inşa edilmiş bir alandır. Ancak her ne kadar bu alanların her birinde birçok araştırma yapılmışsa da görüntü madenciliği araştırmaları yeni ve başlangıç aşamasındadır. Uydu görüntüleri, tıbbi görüntüler ve dijital fotoğraf görüntüleri gibi birçok görüntü günlük olarak yüksek hacimde üretilmektedir ve bu görüntüler incelendiği zaman bizim için çok yararlı olabilirler. Bu çalışmada görüntü işleme ve veri madenciliği yardımı ile bir dizi görüntü (meyve görüntüleri) sınıflandırılacaktır. Görüntü işleme bölümü bulanık mantık, sınıflandırma bölümü yapay sinir ağı algoritmasıyla gerçekleştirilecektir. Uygulamanın son bölümünde, bu çalışma diğer sınıflandırma algoritmaları ile sınanıp, sonuçlar karşılaştırılacaktır.

**2016, 79 sayfa**

**Anahtar Kelimeler:** görüntü madenciliği, görüntü işleme, makine öğrenmesi, bilgi çıkarma, bulanık mantık, yapay sinir ağları

## ABSTRACT

Master Thesis

**AN APPLICATION ON THE CLASSIFICATION OF FRUIT IMAFES BASED ON IMAGE MINING AND IMAGE PROCESSING TECHNIQUES**

Arash MANZOORİ

Atatürk University
Institute of Applied Sciences
Department of Computer Engineering

Supervisor: Asst. Prof. Dr. Tolga AYDIN

Progress made in imaging technology and storage of images leads to enormous growth in very large databases of images. Image mining discusses the context of hidden knowledge extraction from images, finding the relationship between image data or discovering the patterns that are not explicitly stored in the images. Image mining is an interdisciplinary field based on expertise in computer vision, image processing, image retrieval, data mining, machine learning, databases and artificial intelligence. However, research on mining images is a novel issue, although a great deal of research has been done on each of those areas. High volume of images, such as satellite images, medical images, digital photographs, are produced on the daily basis. They may provide us valuable information if analyzed carefully. In this study, a series of images (fruit images) will be classified by the help of image processing and data mining techniques. Image processing task is performed by using fuzzy logic principles and classification part is conducted by using the artificial neural network algorithm. In the last part of the study, other classification techniques are also employed and the results are compared.

**2016, 79 pages**



**Keywords:** image mining, image processing, data mining, knowledge extraction, fuzzy logic, artificial neural networks

## TEŞEKKÜR

Yüksek lisans tezimin belirlenmesi ve tamamlanması aşamalarında, öncelikle tezimi değerlendirerek, değerli görüş ve eleştirileriyle her türlü ilgi ve yardımı esirgemeyen tez danışmanım Sayın Yrd. Doç. Dr. Tolga AYDIN'a en içten teşekkürlerimi sunarım.

Tez çalışmamın her aşamasında bana verdikleri destekle çalışma azmi kazandıran eşim ve aileme en derin sevgi ve saygılarımı sunarım.

**Arash MANZOORİ**
**Aralık, 2016**

# İÇİNDEKİLER

# ŞEKİLLER DİZİNİ

# 1. GİRİŞ

Yaşamış olduğumuz bu zaman diliminde her alışverişte, her bankacılık işleminde, her türlü kamusal alandaki işlemlerde, her görüntü ve ses cihazlarından kayıt edilen veriler hızla depolanmakta ve depolanan bu veriler çok hızlı boyutlarda artmaktadır. Fakat bu veriler istenildiği şekilde değerlendirilememekte ve hızla büyüyen bilgi yığınları şekline dönüşmektedir. Bu veriler belli bir amaç doğrultusunda işlendiği zaman bir anlam ifade etmeye başlar bu yüzden büyük miktardaki verileri işleyebilen teknikleri kullanabilmek büyük önem kazanmaktadır. Bu ham veriyi bilgiye veya anlamlı hale dönüştürme işlemleri veri madenciliği ile yapılabilmektedir.

Günümüzde birçok görüntü (uydu görüntüleri, tıbbi görüntüler, dijital fotoğraf görüntüleri) günlük olarak büyük hacimde üretilmektedir, bu görüntüler incelendiğinde bizim için çok yararlı olabilirler. Bu büyüklükteki verilerin analizi, bu analiz sonucunda daha anlamlı bilgi elde etme ve elde edilen bilgiyi yorumlama işi veri madenciliği alanlarından biri olan görüntü madenciliği alanına girer. Görüntü madenciliği, görüntüdeki gizli bilgileri ayıklama, görüntü verilerinin arasındaki ilişkileri bulma ve resimde net olarak depolanmayan modelleri çıkarma gibi benzer konuları tartışır. Elde edilen bu bilgilerle otomatik yüz tanıma, plaka okuma, güvenlik kameralarından suç tespit etme, tıbbi görüntülerden hastalık teşhis etme, robotların çevrelerini algılamaları, savunma sistemlerinin hedef tanıma ve takibi, uydu görüntülerinden yeryüzü yapılarını tanıma, fotoğraf veri tabanlarında görüntü arama, üretimde hatalarının otomatik tespiti gibi pek çok uygulama yapılabilir. Görüntü madenciliği, görüntü alanında veri madenciliği gelişiminden çok daha fazlasıdır. Görüntü madenciliği, özel yapay görme, görüntü işleme, görüntü alma, veri madenciliği, makine öğrenmesi, veri tabanları ve yapay zekâ üzerine inşa edilmiş bir alandır. Ancak her ne kadar bu alanların her birinde birçok araştırma yapılmışsa da görüntü veri madenciliği araştırmaları yeni ve başlangıç aşamasındadır. Görüntü madenciliğin hızla gelişmesi yolunda önemli engellerden biri konu ve araştırma sonuçlarının yanlış anlaşılmasıdır. Birçok araştırmacı görüntü madenciliğini veri madenciliği uygulamalarının basit bir gelişmesi olarak yanlış algılamışlar, buna ek olarak bazıları, görüntü madenciliğini desen tanımının başka bir

versiyonu olarak bilmektedirler. Oysaki görüntü madenciliği, sadece görüntü için veri madenciliği algoritmalarını kullanmak değildir.

Bu çalışmada ilk olarak görüntü madenciliğinin ne olduğunu tanımlayıp, tarihi gelişimine genel bir bakışımız olacak; ardından görüntü madenciliği çerçevelerini göz önüne alarak görüntü madenciliği süreçlerini tartışacağız. Daha sonra, bulanık mantık ve yapay sinir ağı konuları gözden geçirilecektir. Son olarak uygulama bölümünde görüntü işleme ve veri madenciliği yardımı ile bir dizi görüntü sınıflandırılacaktır. Bu uygulamada görüntü işleme bölümü bulanık mantık; sınıflandırma bölümü yapay sinir ağı algoritmasıyla gerçekleştirilecektir. Uygulamanın son bölümünde bu çalışma diğer sınıflandırma algoritmaları ile sınanıp sonuçlar yapay sinir ağı algoritmasıyla karşılaştırılacaktır.

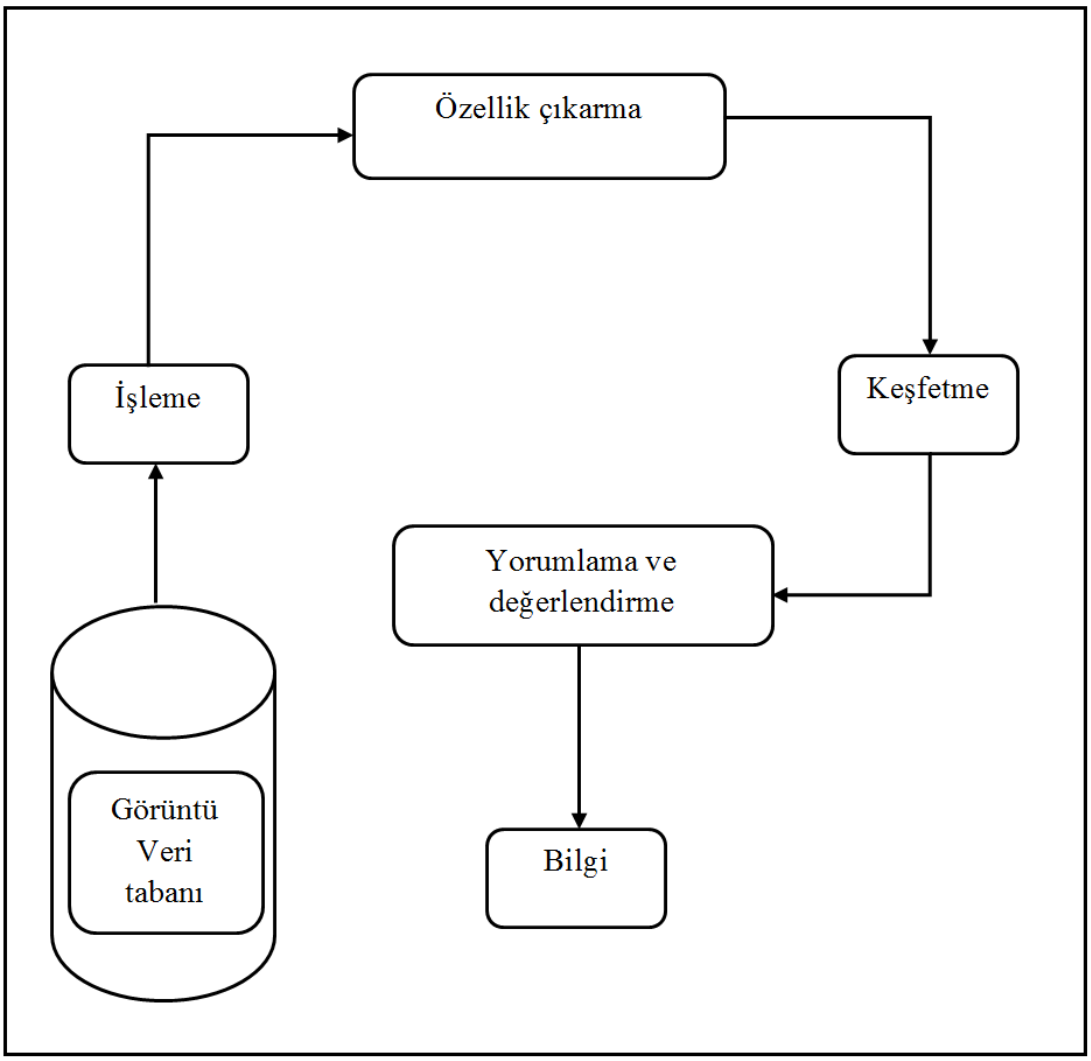
# 2. KURAMSAL TEMELLER

## 2.1. Görüntü Madenciliği Nedir?

Görüntü madenciliği, görüntü verilerinin birbiriyle olan bağımlılığı, bilgi madenciliği ve net bir şekilde görüntülerde depolanmayan desenler üzerinde çalışan bir tekniktir. Görüntü madenciliğin iki temel tekniği vardır, birincisi çok sayıda bireysel görüntü arasında araştırır; ikinci teknikse bir dizi bütünleşmiş ve bağlantılı görüntüler arasında çalışır. Görüntü madenciliğin temel amacı, görüntünün ayrıntılarını bilmeden ondan tüm önemli desenleri ayıklamaktır. Yani girilen bir dizi görüntünün temel bilgileri olmadan, önemli desenleri akıllıca onlardan ayıklamaktır. Ayıklanan desenler, desen sınıflandırma (classification patterns), açıklama kalıpları (description patterns), ilişkili desenler (correlation patterns ), geçici kalıplar (temporal patterns) ve mekânsal desenler (spatial patterns) gibi çeşitli türde olabilir. Görüntü madenciliği sistemi ile aşağıdaki etkinlikleri yapmak mümkün olur.

- Görüntü kaydetme
- Görüntü işleme
- Görüntü özelliklerini çıkartma
- İndeksleme
- Desen ve bilgi keşfi

Şekil 2.1, görüntü madenciliği sistemi için ortak bir yapı modelini göstermektedir. Bu sistem, girdi olarak belirli bir görüntü örneğini göz önüne alarak görüntü işlemlerini gerçekleştirir, daha sonra görüntülerin özelliklerini çıkarmak için onları belirli bir formata döndürür ve oluşan veriler içinde keşfetmeye başlar. Son olarak üretilen sonuçları yorumlar ve değerlendirir. Eğer değerlendirme sonuçları kabul edilirse, istenen bilgi çıkarılır. Genel olarak görüntü madenciliği sisteminde iki tür görüntü girdi olarak kabul edilir, birincisi ikonik (iconic) girdiler ki orjinal görüntülerden (original image) oluşur ve ikinci ise sembolik girdiler (symbolic description of image) ki görüntülerin açıklamaları ve kavramları girdi olarak kabul edilir.




**Şekil 2.1.** Görüntü analiz sistemi

## 2.2. Görüntü Madenciliği Tarihine Genel Bir Bakış

Görüntü madenciliğin alanında çoklu araştırmalar yapılmıştır, bu bölümde bu çalışmaların bazılarına genel bir bakışımız olacak.

2002 yılında, Pinner görüntü arşivleme sistemlerine bir veri madenciliği tekniği sunmuştur. Bu teknik standarttır ve görüntü madenciliğinin diğer sistemleri için

kullanılabilir. O bu yöntemi, tıbbi görüntülerin (MRI, X-ray ve …) analizi için kullanmıştır.

2003 yılında, internet den otomatik bilgi çıkarma modelleriyle bir görüntü sınıflandırma yöntemi Yani tarafından sunuldu. Bu yöntemle, işleme üç aşamada yapılır. İlk aşamada, görüntüler belirtilen anahtar kelimeler ile web'den toplanır. İkinci aşamada, toplanan görüntülerin özellikleri ayıklanır. Üçüncü aşamada, yeni bir resim özelliklerine göre ilgili sınıfa yerleştirilir.

2004 yılında, Srivastava ve Oza görüntü madenciliği için bilgiye dayalı (knowledge driven image mining) bir teknik önerdiler. Bu teknikte görüntülerin otomatik araştırma konusu, Mercer Kernels teorisine dayanarak öne sürülmüştür. Yazar bu teorisini bulut yüzdesi, kar, buz, kuruluk, yangın tehlikesi ve diğer jeofizik süreçlerin görüntülerini otomatik olarak tanımlamak ve etiketlemek için kullanır.

Görüntü içeriğine dayalı doku görüntü madenciliği (tissue image mining) Gholap tarafından 2005 yılında önerilmiştir. Biyolojik verilerin yönetimi ve keşfi, son biyoloji araştırmalarının önemli alanlarıdır. Yüksek verim ve büyük bilgi içeriği her doku mikro dizi analizi (TMA) (tissue microarray analysis) sisteminin iki önemli özelliğidir. Bir patolog bilgisinden istifade ederek dört seviyeli sistem (Görüntü analizi, desen tanımlama ve yapay zekâ) bu yaklaşımda önerilmiştir. Görüntü ve bilgi işleme seviyesinde, renk ve eşitsizlik gibi bilgiler kullanılır. Nesne düzeyinde, hasarlı parçalar ve hücre bileşenleri tespit edilir. Anlam düzeyinde, doku görüntüsündeki tek tek hücrelerin düzeneği ve yapılandırması incelenir. Kavramsal düzeyde (Semantic level), doku görüntüsü üzerinde hücrelerin her birinin yapılandırması kontrol edilir. Ve en üst düzeyde yani bilgi düzeyinde (Knowledge level) patoloğun varsayımları belirlenir.

Sanjay 2007 yılında, dalgacık dönüşüm seviyesini (Wavelet level) kullanarak bir görüntü analiz tekniği sundu. Bu yöntem üç aşamalı (görüntü toplama, öğrenme ve sınıflandırma) bir süreçten oluşur. Mikrodalga dönüşüm seviyesi bir zaman frekansı kullanır. Bu nedenle, görüntü madenciliği sistemlerinde Fourier serilerinin yerine

kullanılabilir. Mikrodalga dönüşümü bir görüntüyü birbirine benzemeyen farklı frekans aralıklara böler ve temel bileşenlerin analizi (PCA) için kullanır.

Görüntü işleme ve görüntü madenciliğe dayalı karar ağacı, Kun-che tarafından 2009 yılında tanıtıldı. Bu yöntemiyle, görüntü verilerinin keşfi ve işlenmesi için karar ağacına dayalı genel bir çerçeve sunuldu. Bu yöntemle, görüntüdeki her pikselin özellikleri çıkarılır ve veri tabanı tablolarına benzer tablolara dönüşür. Oluşan bu veriler, çeşitli veri madenciliği arama ve keşif algoritmalarında kullanılabilir.

2010 yılında görüntü madenciliği için Victor ve Peter kümeleme algoritmalarına dayanarak yeni bir minimum yayılan ağaç (minimum spanning tree) sundular.

Günümüzde bilgilerin çoğu dijital görüntü şeklinde olduğu için görüntü veri tabanları yaygın olarak kullanılmakta bu nedenle araştırmaların önemli alanlarından biri, görüntülerin gerialımı ve keşfidir. Görüntü madenciliği, tıbbi teşhisleri, uzay araştırmaları, biyoloji, uzaktan algılama gibi birçok alanda geniş olarak kullanılmakta ve gelecekteki araştırmalar için çok geniş bir alan sağlamaktadır.

# 3. MATERYAL ve YÖNTEM

## 3.1. Görüntü Madenciliği Çerçeveleri

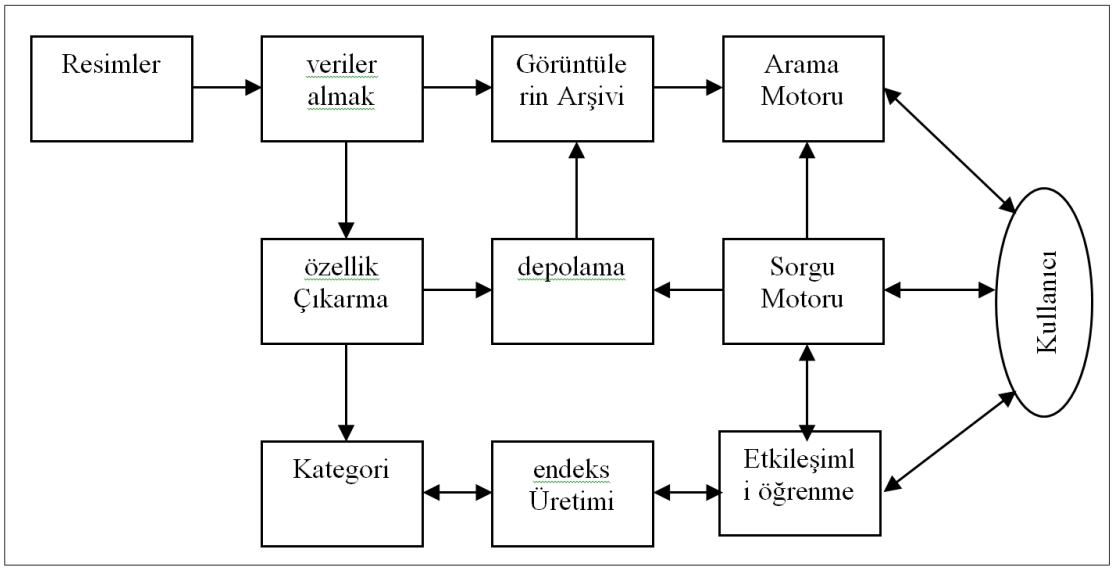
Görüntü madenciliği için uygun bir çerçeve geliştirmek amacıyla, görüntü madenciliği alanında birçok çalışma yapılmıştır. Bir görüntü veri tabanı ham görüntüler içerir ve doğrudan bir görüntü madenciliği sisteminde kullanılamaz. Üst seviyedeki keşif algoritmaları için kullanılabilir bilgiyi üretmek amacıyla, ham görüntülerin işlenmiş olmaları lazımdır. Görüntü madenciliği sistemlerinde, veri madenciliği ve desen tanıma için farklı görüntü alma ve indeksleme teknikleri olduğu için bu sistemler çoğu zaman karmaşıktırlar. İyi bir görüntü madenciliği sisteminden, görüntü deposuna (image repository) rahat erişim sağlaması ve bu depodaki görüntülerden kullanıcıya gerekli bilgi ve deseni keşif etmesi beklenir. Bu amaçları elde etmek için bir görüntü madenciliği sistemi genellikle şu işlemleri içerir: görüntü depolama (image storage), görüntü işleme (image processing), özellik çıkarma (feature extraction), indeksleme ve görüntüleri geri alma (image indexing and retrieval), bilgi keşfi (pattern discovery) ve desen keşif (knowledge discovery).

Görüntü madenciliği sistemlerini tanımlamak için kullanılan çerçeveler iki gruba ayrılır, biri performansa dayalı çerçeveler (function-driven frameworks) ve diğeri bilgiye dayalı (information-driven frameworks) çerçevelerdir. Birincisi bir görüntü madenciliği sistemini düzenlemek için çeşitli bileşen ve fonksiyonlara odaklanır. İkincisi ise bilgi üzerinde özel vurguyla bir hiyerarşik yapı tasarlar.

### 3.1.1. Performansa dayalı çerçeveler

Mevcut görüntü madenciliği sistemlerinin çoğunun mimarileri bu çerçeveye dayanmaktadır. Mihai datku ve Klaus seidel bu çerçeve için iki modülden oluşan bir akıllı uydu madenciliği sistemini önerdiler. (Şekil 3.1)

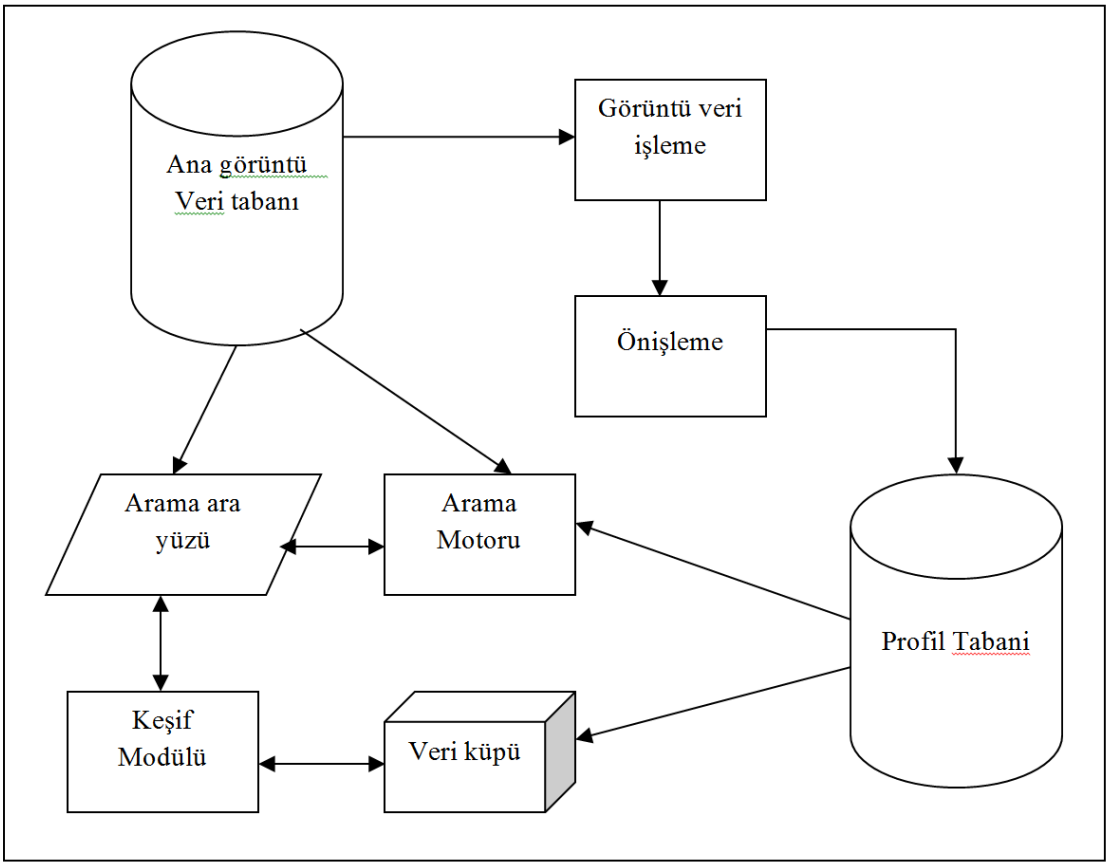
- Görüntülerden bilgi çıkarmadan, ham görüntüleri depolamadan ve görüntülerin geri alımından sorumlu veri alma (data acquisition), ön işleme (preprocessing) ve arşivleme (archiving) modülleri
- Görüntünün anlamını keşfetmek ve ilgili etkinliklerin tespiti için görüntü madenciliği modülleri




**Şekil 3.1.** Uydu keşif sistemi mimarisi

Benzer şekilde, dört ana bileşenden oluşan bir multimedya minör sistem vardır.

- Multimedya veri tabanından görüntülerin ve videoların çıkarılması için görüntü ekskavatörü (image excavator).
- Veri tabanındaki görüntülerin özelliklerini çıkarmak için önişlemce.
- Veri tabanındaki resim ve video özelliklerini sorularla eşleştiren bir arama çekirdeği.
- Tanımlayıcı, sınıflandırıcı ve bağlayıcı modüllerden ulaşan keşif modülleri. Şekil 3.2'de sistem mimarisi gösterilmektedir. .




**Şekil 3.2.** Multimedya minör sistem

### 3.1.2. Bilgiye dayalı çerçeveler

Performansa dayalı bir çerçeve, görüntü verileri için anlamlı madenciliği gerçekleştirebilmek için farklı düzeylerde gerekli bilgiyi elde edemez. Zhang çeşitli seviyelerdeki bilgi rolünü vurgulamak amacıyla bilgiye odaklı bir çerçeve önerdi, bu çerçeve dört seviyeden oluşur:

(a) Piksel seviyesi (pixel level): aynı zamanda en düşük seviyedir. Bu seviye ham görüntü bilgilerinden, örneğin görüntünün pikselleri ve görüntünün ilkel özelliklerinden (renk, doku ve şekil) oluşur.

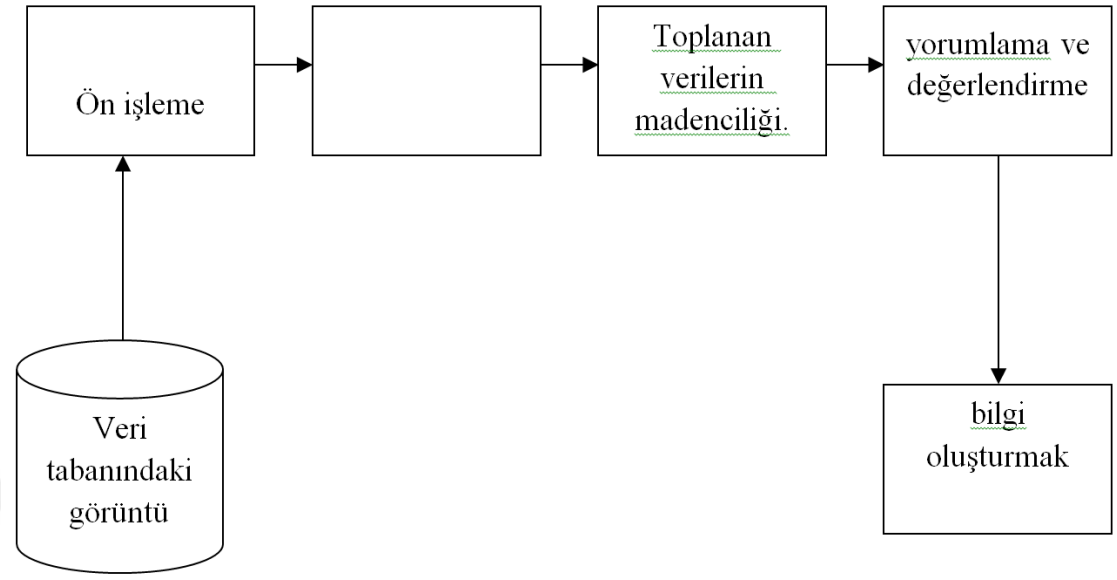
(b) Nesne seviye (object level): piksel seviyesindeki ilkel özelliklere dayalı nesne ya bölge bilgileri ile ilgilenir.

(c) Anlamsal seviye (semantic level): belirlenen nesne ve bölgelerden, alan bilgilerini oluşturmak için üst düzey anlamsal kavramları kullanır.
(d) Desen ve bilgi seviye (pattern & knowledge level): resim verilerinden gizli desen ve bilgiyi keşfetmek için alfa numerik verileri ve anlamsal kavramları kullanır.

### 3.2. Görüntü Madenciliği Süreci

Bir görüntü madenciliği sisteminde istenen görüntüleri elde etmek için farklı aktiviteler yapılabilir. Bu bölümde, görüntü madenciliği sırasında gerçekleşen bazı işlemler ve her işlemde kullanılan bazı teknikler ele alınacaktır. Şu noktaya dikkat edilmelidir ki bu işlemlerin gecikmeleri yâda öncelikleri sadece görüntü analizi için tasarlanmış modele bağlıdır. Şekil 3.3 görüntü madenciliği sürecini göstermektedir. Görüntü madenciliği süreci aşağıdakileri işlemleri içerebilir.

1. Ön işleme ve gürültü azaltma
2. Sınıflandırma
3. Renk işleme
4. Kümeleme
5. Özellik çıkarma
6. Özellik seçme
7. Histogram eşitleme (histogram equalization )
8. Sonuçları değerlendirme




**Şekil 3.3.** Görüntü madenciliği süreci

### 3.2.1. Ön işleme (preprocessing)

Görüntülere ardışık olarak görüntü işleme fonksiyonları uygulanmadan önce, görüntüler bir ön işlemeden geçirilerek elde edilecek sonuçların daha sağlıklı olması sağlanabilir. Bu nedenle özellikleri çıkarma aşamasını daha kolay ve güvenilir hale getirmek için, herhangi bir işleme başlamadan önce görüntünü kalitesini artırmak gerekir. Yaygın olarak kullanılan görüntü önişleme ve analiz operasyonlarının çoğu aşağıda listelenen üç kategori içinde listelenebilir:

- Ön işleme (görüntü restorasyonu): bu operasyonlar ana veri analiz işlemleri öncesinde verideki bozuklukların giderilmesi için uygulanır
- Görüntü iyileştirme: bu işlemler görsel yorumlama ve analiz amacı ile kullanılacak verinin iyileştirilmesi için uygulanır
- Görüntü Dönüşümü: bu operasyonlar çoğunlukla çok bantlı spektral verinin birlikte işlenmesi ile ilgilidir

Filtreleme, bir görüntüyü iyileştirme ve geliştirmek için kullanılan tekniklerden biridir. Görüntüdeki bazı özellikleri vurgulamak ya da kaldırmak için filtreleme kullanılır. Doğrusal veya doğrusal olmayan filtreleme yöntemleriyle, görüntüdeki parazitler silinir. Düşük - geçişli filtreler (low - pass filters ), yüksek - geçişli filtreler (high - pass filters), orta - geçişli filtreler (median - pass filters) filtreleme yöntemlerinin bazılarıdır.

### 3.2.2. Sınıflandırma

Bir görüntüdeki her bir piksel değerinin sahip olduğu özellik grubunun belirlenmesi ve seçilen bantlar için benzer spektral özelliklere sahip piksellerin özellik gruplarına atanması işlemi sınıflandırma olarak tanımlanmaktadır. Sınıflandırma veri madenciliğinin en çok kullanıldığı alandır. Var olan veri tabanının bir kısmı eğitim verisi olarak kullanılarak sınıflandırma kuralları oluşturulur. Bu kurallar yardımıyla yeni bir durum ortaya çıktığında nasıl karar verileceği belirlenir. Veri madenciliğinin sınıflandırma grubu içerisinde en sık kullandığı teknik karar ağaçlarıdır. Aynı zamanda lojistik regresyon, bayes inanç ağları (bayesian belief networks ), discriminant analizi, sinir ağları ve fuzzy setleri de kullanılmaktadır. İnsanlar verileri daima sınıflandırdıkları, kategorize ettikleri ve derecelendirdikleri için sınıflandırma, hem veri madenciliğinin temeli olarak hem de veri hazırlama aracı olarak da kullanılabilmektedir.

### 3.2.3. Renk işleme

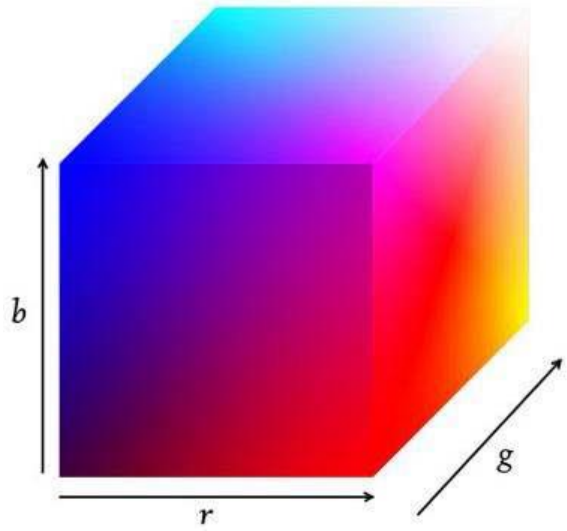
Bir görüntünün bilgisayar ortamında saklanabilmesi için iki boyutlu bir koordinat sistemi içinde piksel adı verilen hücrelere bölünür ve bu hücrelerin yatay-düşey koordinatı ile o hücrenin renk değerini içeren sayısal değerler olarak saklanır. Her pikselin renk değerini (renkli görüntüler için 3 adet, diğerleri için bir adet) içermesi gerekmektedir.

Bir RGB sisteminin yardımıyla üretilen renkli görüntü, M*N*3 Piksellerden oluşan bir dizidir. Her piksel resmin belirli bir koordinatında kırmızı, mavi ve yeşil miktarını temsil eder. Aşağıdaki gibi, görüntüdeki her renk bileşeninin ortalaması hesaplanır.

P= Görüntünün piksellerin toplam sayısı
R(P)= Kırmızı piksellerin sayısı
G(P)= Yeşil piksellerin sayısı
B(P)= Mavi piksellerin sayısı

Kırmızı piksellerin Ortalaması= $\frac{R(P)}{P}$
Kırmızı piksellerin Ortalaması= $\frac{G(P)}{P}$
Kırmızı piksellerin Ortalaması= $\frac{B(P)}{P}$

Görüntüdeki renk bileşenlerinin belirtilmesi, görüntülerin sınıflandırma ve kümelemesinde bize yardımcı olur. **(**Şekil 3.4**)**




**Şekil 3.4.** RGB renk uzayı

### 3.2.4. Kümeleme

Verilerin kendi aralarındaki benzerliklerinin göz önüne alınarak gruplandırılması işlemidir ve kümeleme yöntemlerinin çoğu veri arasındaki uzaklıkları kullanır. Hiyerarşik kümeleme yöntemleri en yakın komşu algoritması ve en uzak komşu algoritmasıdır. Hiyerarşik olmayan kümeleme yöntemleri arasında k-ortalamalar yöntemi sayılabilir. Uygulamalarda çok sayıda kümeleme yöntemi kullanılmaktadır. Bu yöntemler, değişkenler arasındaki benzerliklerden ya da farklılıklardan yararlanarak bir kümeyi alt kümelere ayırmakta kullanılmaktadır. Hangi tekniğin kullanılacağı küme sayısına bağlı olmakla birlikte her iki tekniğin beraber kullanılması çok daha yararlıdır. Böylece hem sonuçları, hem de iki tekniğin hangisinin daha uygun sonuçlar verdiğini karşılaştırmak mümkün olmaktadır. Kümeleme analizinin amacı, gruplanmamış verileri benzerliklerine göre sınıflandırmak ve araştırmacıya özetleyici bilgiler elde etmede yardımcı olmaktır. Kümeleme analizinin uygulanabilmesi için verilerin normal dağılımlı olması varsayımı olmakla birlikte, bu varsayım teoride kalmakta ve uygulamalarda göz ardı edilmektedir. Sadece uzaklık değerlerinin normal dağılıma uygunluğu ile yetinilmektedir. Bu varsayımın sağlanması durumunda kümeleme analizinde kovaryans matrisi için farklı bir varsayım gerekmemektedir.

### 3.2.5. Özellik çıkarma

Görüntüde yer alan farklı her bir bilgi, özellik olarak tanımlanabilir. Bunlar, görüntü piksellerine yansıyan karakteristik özelliklerdir. Piksel gri tonlarındaki değişimler, arka zemin niteliği, doku yapısı ve bölgesel doku farklılıkları bunlara örnek olarak verilebilir. Şekil tanıma acısından ise özellikler, herhangi bir nesneye ait fiziksel biçimi karakterize eden bilgilerden oluşmaktadır. Özellikler, sözü edilen bilgilerin görüntü ya da şekilden ayrıştırılarak seçilmesi sonucu elde edilen sayısal verilerden meydana gelmektedir. Özellik bilgisi çıkartımında, şekil yapısında yer alan köşe noktaları, kenar çizgileri gibi yapısal özellikleri tanımlayan veriler ile birlikte, yine şekilden dönüşüm yöntemleri (moment, fourier v.b.) ile elde edilen veriler de kullanılabilmektedir.

Şekil tanıma işleminde birçok özellik çıkarma yönteminden hangilerinin kullanılacağı, seçilen uygulama doğrultusunda belirlenmektedir. Şekil tanıma amacıyla üretilen özellik bilgilerinden bir özellik seti oluşturma işlemi su bilgileri sağlamayı amaçlar:

- Aynı sınıfa ait örnekler için özellik bilgilerindeki değişim küçük olmalıdır.
- Farklı sınıflar arasındaki değişimler ise büyük olmalıdır.

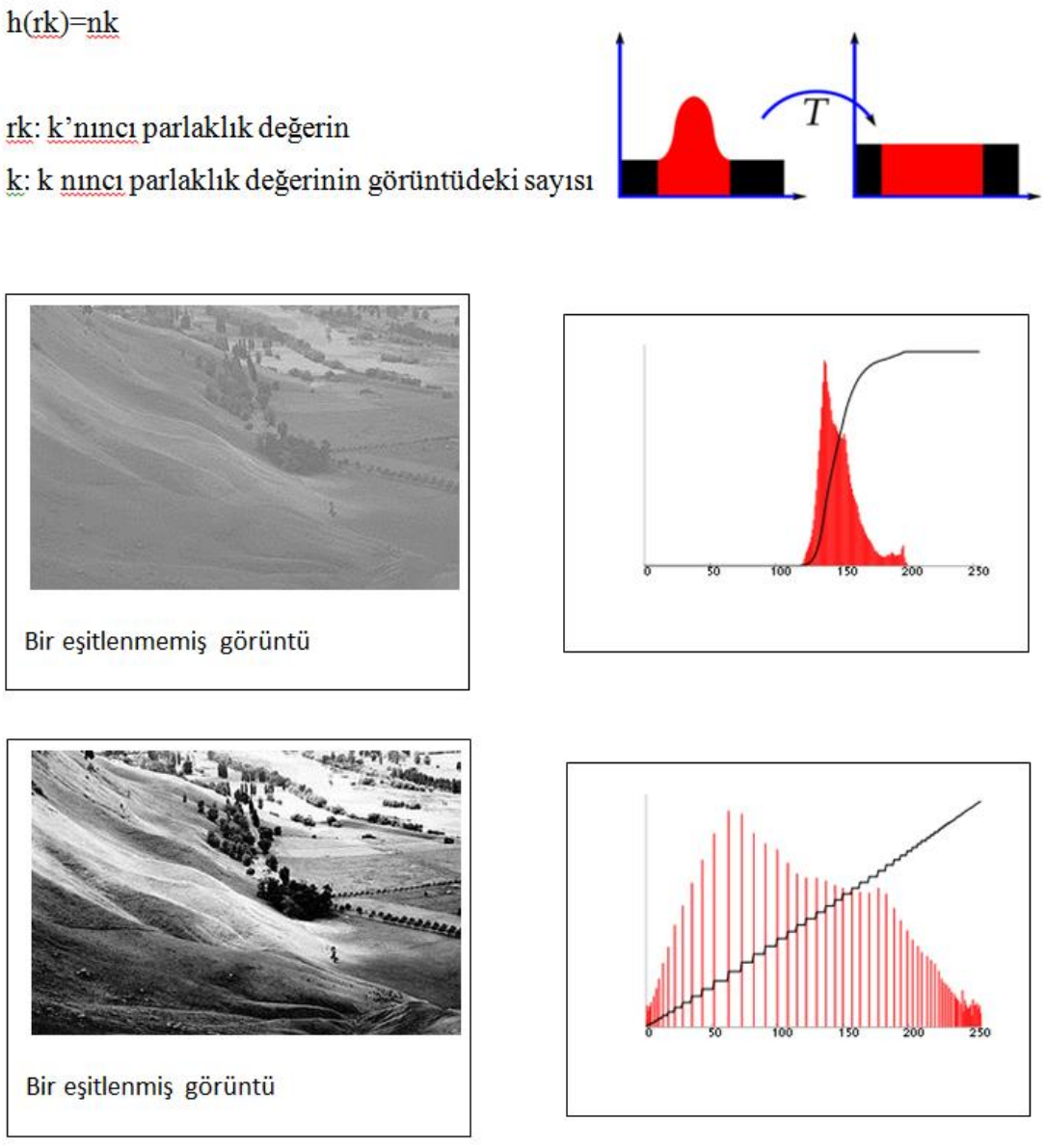
### 3.2.6. Özellik seçme

Literatürde önerilmiş birçok özellik çıkartma yöntemi vardır. Fakat bir probleme ait veride ne kadar çok nitelik varsa makine öğrenmesi yöntemlerinin maliyeti de o kadar artar. Bu istenmeyen bir durumdur. Bu yüzden eldeki problemi en uygun şekilde temsil etmek üzere minimum sayıda özellikten yararlanmak gerekir. Bilinen onlarca özellik çıkartma yönteminden hangilerinin seçilmesi gerektiği üzerine yapılmış yine birçok çalışma vardır. Bu çalışmalar, muhtemel özellik çıkartma yöntemleri içerisinde uygun olanları en kısa sürede bulmayı amaçlar. Özellikleri seçmek için, entropi dayalı ölçüm (entropy based measure), gini endeksi (gini index), gini oranı (gini ratio) ve ki kare (chi square) yöntemleri kullanılır. Özelliklerin seçimi, problemin boyutunun azalmasına neden olur ve böylece tahmin doğruluğu artar ve hesaplama süresi azalır. Bu işlemler ile ilgili olmayan özellikler, gürültü ve atıkları ortadan kaldırmak mümkün olur. Bu nedenle her zaman özelliklerden bir altküme oluşturmaya çalışılır. Genellikle arama prosedürleri ile özellikler seçilir. Bu nedenle farklı arama yöntemleri bu amaç için geliştirilmiştir. Özellik seçimi için kullanılan en popüler algoritmalar arasında ileriye doğru sıralı seçim (sequential forward selection), geriye doğru sıralı seçim (sequential backward selection), genetik algoritmalar ve parçacık sürüsü optimizasyonu (particle swarm optimization ) yer alır.

### 3.2.7. Histogram eşitleme (histogram equalization)

Histogram bir resimdeki renk değerlerinin sayısını gösteren grafiktir. Histogram dengeleme veya eşitleme de bir resimdeki renk değerlerinin belli bir yerde kümelenmiş

olmasından kaynaklanan, renk dağılım bozukluğunu gidermek için kullanılan bir yöntemdir. Histogram matematiksel olarak aşağıdaki şekilde gösterilebilir: (Şekil 3.5)




**Şekil 3.5.** Histogram eşitleme

### 3.2.8. Sonuçları değerlendirme

Bir görüntü madenciliği sisteminde yapılması gereken en önemli işlemlerden biri sonuçların değerlendirmesidir. Değerlendirme sürecinde modelin doğru kurulup kurulmadığı, gelecekte kullanılabilecek farklı verilerin neler olabileceği, modelin genişletilmesi gibi konuları içerir. Farklı yöntemler değerlendirmek için kullanılabilir ki çoğu durumda bu yöntemler, modeli sağlayan sunucunun varsayılan modellerine bağlıdır.

## 3.3. Bulanık Mantık (Fuzzy logic)

Günlük hayatta rastgele kullandığımız birçok terim genellikle bulanık bir yapıya sahiptir. Bir şeyi tanımlarken, bir olayı açıklarken, komut verirken ve daha birçok durumda kullandığımız sözel veya sayısal ifadeler bulanıklık içerir. Bu terimlere örnek olarak; yaşlı, genç, uzun, kısa, sıcak, soğuk, ılık, bulutlu, parçalı bulutlu, güneşli, hızlı, yavaş, çok, az, biraz, fazla, çok az, çok fazla gibi daha bir çok sözel terim gösterilebilir. Biz insanlar bir olayı anlatıp, bir durum karşısında karar verirken bu tür kesinlik ifade etmeyen terimler kullanırız. Kişinin yaş durumuna göre ona yaşlı, orta yaşlı, genç, çok yaşlı veya çok genç deriz. Yolun kayganlık ve rampa durumuna göre arabanın gaz veya fren pedalına biraz daha yavaş veya biraz daha hızlı basarız. Çalıştığımız odanın ışığı yetersiz ise onu biraz artırır, yeterinden fazla ise biraz azaltırız. Bütün bunlar insan beyninin belirsiz ve kesinlik içermeyen durumlarda nasıl davrandığına ve olayları nasıl değerlendirip, tanımlayıp, komut verdiğine dair birer örnektir.

### 3.3.1. Bulanık kümeler

Bulanık kümeler, kullandığımız sözel ifadeleri bilgisayara aktarabilmek için oluşturduğumuz matematiksel modellerdir. Klasik bir küme, kesin sınırlamalarla verilen bir kümedir. Örneğin, klasik bir küme aşağıdaki gibi belirtilebilir:

$$A= \{ x \mid x > 6 \}$$

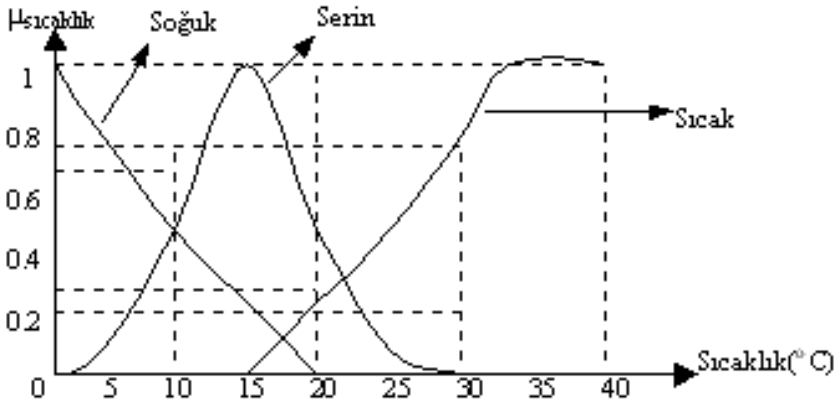
Kapalı sınır noktası burada 6‘dır. Burada olduğu gibi; eğer x bu sayıdan büyükse A kümesine aittir, aksi takdirde bu kümeye ait değildir. Klasik kümelere göre bulanık kümeler, adından da anlaşılacağı gibi kesin limitleri olmayan bir kümedir. Yani, "kümeye ait olan" dan "kümeye ait olmayana" geçiş aşamalı olur ve bulanık kümelere "su sıcak" veya "sıcaklık çok yüksek" gibi tanımlarda olduğu gibi modellemede çoğunlukla dilsel açıklamalara esneklik kazandıran bu düzgün geçiş üyelik fonksiyonları olarak tanımlanmaktadır. Bulanık kümeler üzerindeki işlemler, basit tanımlamalar yardımı ile kesin kümeler üzerinde yapılan işlemlere benzer şekilde yapılır. Bu işlemleri örneklerle açıklayalım:

Örnek: X=[0, +40] aralığında soğuk, serin ve sıcak olarak adlandırılabilecek bulanık sıcaklık kümelerini ele alalım.

Sıcaklık 0°C ile 40°C arasındadır.

Soğuk = {1/0 , 0.8/5 , 0.5/10 , 0.2/15 , 0/20}
Serin = {0/0 , 0.1/5 , 0.5/10 , 1/15 , 0.5/20 , 0.1/25 , 0/30}
Sıcak = {0/15 , 0.2/20 , 0.5/25 , 0.8/30 , 1/35 , 1/40}

Bu kümelerin grafiksel görünümü Şekil 3.6’da gösterilmektedir.




**Şekil 3.6.** Soğuk, Serin ve Sıcak bulanık kümelerinin grafiksel görünümü

### 3.3.2. Bulanık mantığın sağladığı avantajlar

İsminin insanlarda çağrıştırdığı anlamın aksine bulanık mantık, belirsiz ifadelerle yapılan, belirsiz işlemler değildir. Modelleme aşamasında değişkenler ve kuralların esnek bir şekilde belirlenmesidir. Bu esneklik asla rastgelelik ya da belirsizlik içermez. Nasıl bir lastik içinde bulunduğu duruma göre şeklini değiştirirken, bütünlüğünü yapısını koruyabilirse, bir bulanık modelde değişen koşullara değişen cevaplar verirken özündeki yapıyı muhafaza eder. Bulanık Mantığın sağladığı avantajlar:

- İnsan düşünce sistemine ve tarzına yakındır.
- Uygulanmasında mutlaka matematiksel bir modele gereksinim duymaz.
- Yazılımın basit olması nedeniyle, sistem daha ekonomik olarak kurulabilir.
- Bulanık Mantık kavramını anlamak kolaydır.
- Üyelik değerlerinin kullanımı sayesinde, diğer kontrol tekniklerine göre daha esnektir.
- Kesinlik arz etmeyen bilgiler kullanılabilir.
- Doğrusal olmayan fonksiyonların modellenmesine izin verir.
- Sadece uzman kişilerin tecrübeleri ile kolaylıkla bulanık mantığa dayalı bir model yâda sistem tasarlanabilir.
- Gelenekse kontrol teknikleriyle uyum halindedir.

### 3.3.3. Bulanık mantığın uygulandığı alanlarından bazıları

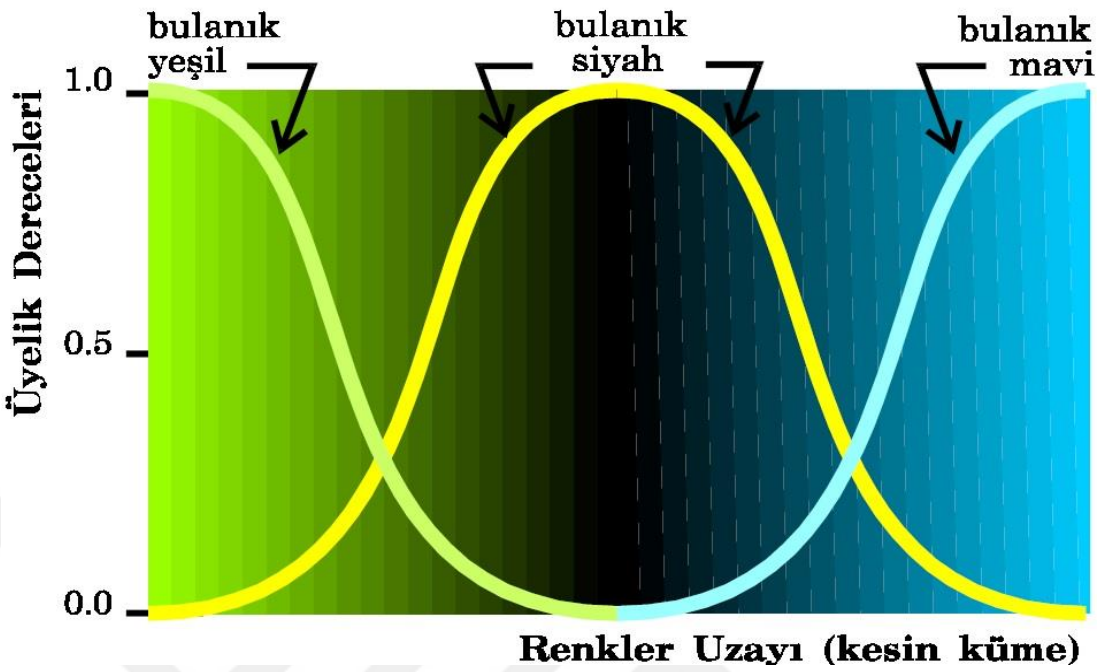
Bulanık mantığın Mamdani ve arkadaşları tarafından denetim sistemlerine ilk uygulanmasından sonra, bu alanda oldukça önemli adımlar atılmaya başlanmıştır. Öyle ki denetim sistemleri bulanık mantığın en fazla uygulandığı alan olarak günümüze kadar gelmiştir.

- Otomatik Kontrol Sistemleri: Robotik, otomasyon, akıllı denetim, izleme sistemleri, ticari elektronik ürünler, vb.

- Bilgi Sistemleri: Bilgi depolama ve yeniden çağırma, Uzman sistemler, bilgi tabanlı sistemler, vb.
- Görüntü Tanımlama: Görüntü işleme, makina görüntülemesi.
- Optimizasyon**:** Fonksiyon optimizasyonu, eğri uydurma, vb.

### 3.3.4. Bulanıklık

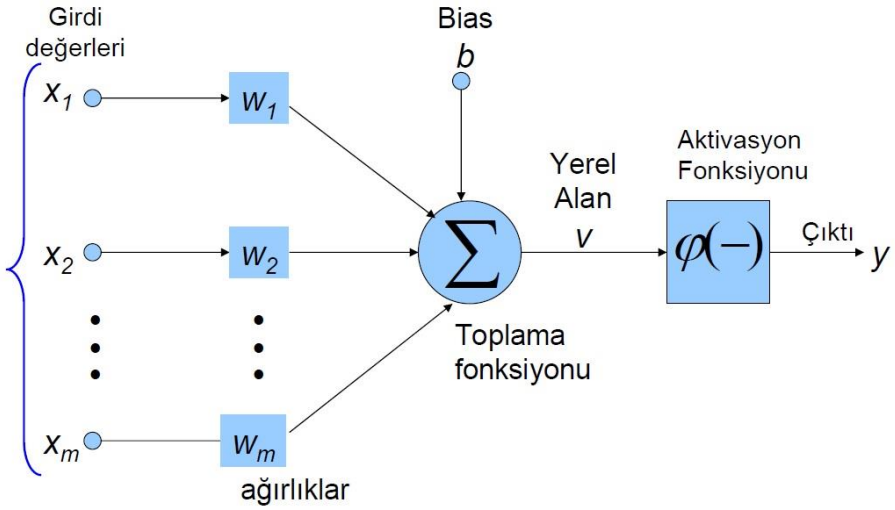
Bulanık mantık konusunun temel elemanı bulanık kümedir. Bulanık kümeler, üyelik fonksiyonları ile karakterize edilirler. Aslında bu üyelik fonksiyonları da birer bulanık sayıdan başka bir şey değildir. Bulanık mantık, üyelik fonksiyonu ve bulanık sayı gibi kavramların iyi anlaşılabilmesi için öncelikle bulanıklık kavramının anlaşılması gerekir. Dikkat edilirse, Şekil 3.7'de, renkler uzayında tanımlı yeşil, siyah ve mavi değişik tonlara sahiptirler. Örneğin soldan sağa doğru ilerledikçe yeşilin renk tonu koyulaşmakta ve siyaha dönüşmektedir. Şeklin tam ortasında renk tam siyahken, sağa doğru ilerleme sürdürülürse, siyahın renk tonu da açılıp mavi olmaktadır. Görüleceği gibi yeşilin bitip siyahın başladığı, siyahın bitip mavinin başladığı noktalar kesin bir şekilde ayrıştırılamamaktadır. Verilen üç renk bölgesi de kesin, sabit bir renk tonuna sahip değildir. Dolayısıyla bu üç renk bölgesini birer bulanık küme ile temsil etmek uygun olacaktır. Verilen şekilde sadece yeşil, siyah ve mavinin tonları bulunduğundan, sadece bu üç rengi temsil eden yeşil, siyah ve mavi bulanık kümelerini tanımlamak yeterli olacaktır.




**Şekil 3.7.** Yeşil, siyah ve mavi bulanık renk kümeleri

### 3.4. Yapay Sinir Ağları

Yapay sinir ağları (YSA), insan beyninin bilgi işleme tekniğinden esinlenerek geliştirilmiş bir bilgiişlem teknolojisidir. YSA ile basit biyolojik sinir sisteminin çalışma şekli simüle edilir. Simüle edilen sinir hücreleri nöronlar içerirler ve bu nöronlar çeşitli şekillerde birbirlerine bağlanarak ağı oluştururlar. Bu ağlar öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. YSA'nın genel blok şeması şöyledir: (Şekil 3.8)



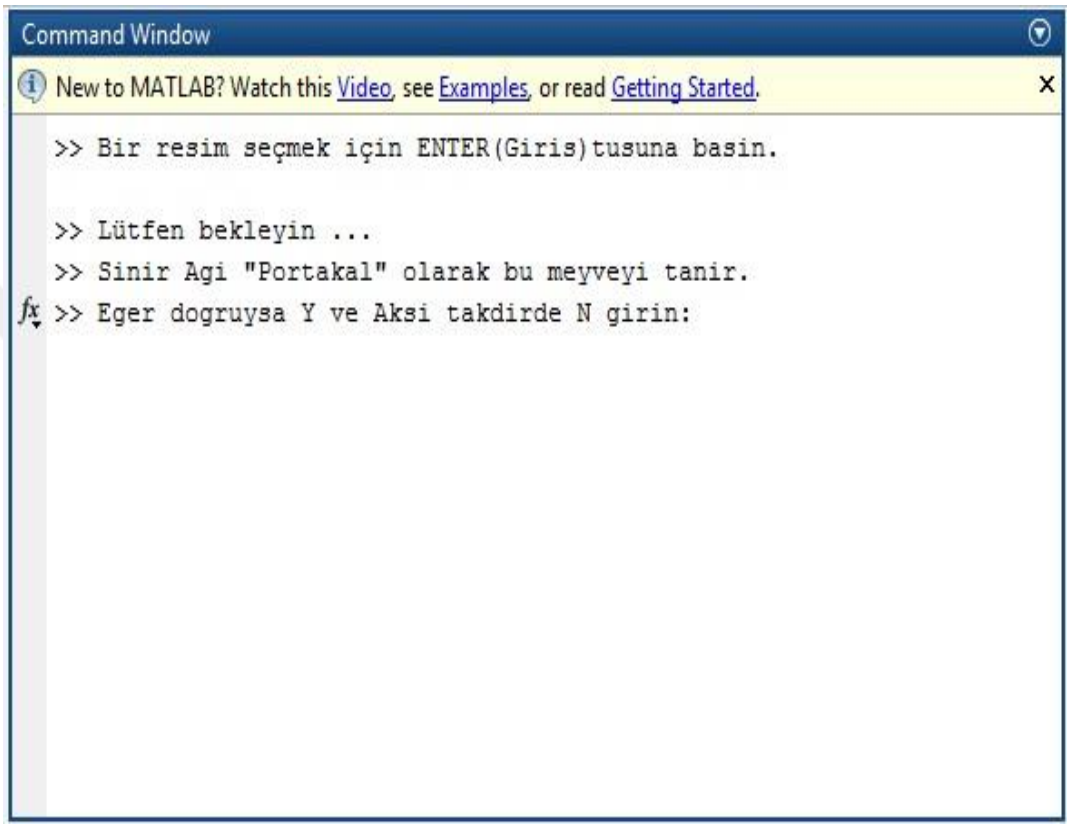

**Şekil 3.8.**Yapay sinir ağları genel yapısı

YSA'lar, ağırlıklandırılmış şekilde birbirlerine bağlanmış birçok işlem biriminden (nöronlar) oluşan matematiksel sistemlerdir. Bir işlem birimi, aslında sık sık transfer fonksiyonu olarak anılan bir denklemdir. Bu işlem birimi, diğer nöronlardan sinyalleri alır; bunları birleştirir, dönüştürür ve sayısal bir sonuç ortaya çıkartır. Genelde, işlem birimleri kabaca gerçek nöronlara karşılık gelirler ve bir ağ içinde birbirlerine bağlanırlar; bu yapı da sinir ağlarını oluşturmaktadır.

### 3.5. Uygulama

Tezin bu aşamasında, görüntü işleme ve veri madenciliği işlemlerini kullanarak görüntüleri sınıflandırmaya çalışacağız. Sınıflandırma, görüntünün içindeki nesneye göre gerçekleşecektir. Genel olarak görüntünün içindeki bir nesneyi tanımlamak için, istenen nesnenin özelliklerini ayıklayıp o özellikleri göz önünde bulundurarak ilgili nesneyi tanımlamak mümkün olur. Bu çalışmada resmin içindeki meyve türünü tanımlamak istediğimiz için seçilen özellikler meyvenin rengi, genişliğin uzunluğa oranı, yuvarlaklık miktarı, alanı ve çevresi olabilir. Bu nedenle bu beş özellik resimden ayıklanır. Ayıklanan bu özelliklere göre sinir ağının eğitimi verilir.

Yazılım şu şekilde çalışır: İlk olarak bir resmi (meyve resmini) alır ve resimdeki meyvenin adı hakkında soru sorarak meyvenin ismini öğrenmeye çalışır, 4 ile 5 resimden sonra meyve türünü belirlemek mümkün olur. (Şekil 3.9)




**Şekil 3.9.** Uygulama

### 3.5.1. Yazılımın ana formu

Yazılım FİNAL adında bir ana dosyadan oluşur, bu dosya içinde tüm fonksiyonlar çağırılır. İlk olarak kullanıcıdan bir resim alınır, mevcut fonksiyonları kullanarak resimden 5 özellik ayıklanır, ayıklanan bu özellikler sinir ağının girişlerini oluşturur. Girdilere göre sinir ağı meyvenin adını tahmin etmeye çalışır, eğer doğru tahmin ederse yeni bir resim almak için beklemeye geçer; tahmini doğru değilse ya da hiç tanımadıysa kullanıcıdan meyvenin ismini sorar, aldığı cevapla sinir ağını eğitir. Test sırasında yazılımın uygulanmasını hızlandırmak için, daha önceden yapılmış olan 8 portakal

resminin işlemleri 2 dosya şeklinde (Fruit and Target) yazılıma verilir Bu nedenle, yazılım başlangıçta portakal meyvesini tanır.

### 3.5.1.a. Fruit

Bu dosya, görüntü işleminin sonucunu saklayan bir matristen oluşur. Oluşan bu matris sinir ağının girdisi olarak kullanılır. Daha önce belirtildiği gibi, 8 görüntünün işlem sonuçlarını yazılıma verdiğimiz için, 8 sütundan oluşan bir matrisimiz olacaktır.

### 3.5.1.b. Target

Target dosyası, sinir ağının çıktısını depolamak için kullanılır.

00 → İlk meyve (portakal)
01 → İkinci meyve
10 → Üçüncü meyve
11 → Dördüncü meyve

```
clear all;
close all;
clc;

load('fruit');
load('target');
load('database');


compare_match=0;
norm_var=[ 1    1    1    1    1    1    1    1;...
```

```
 0.1   0.1   0.1   0.1   0.1   0.1   0.1   0.1];

fruit_codes = [1,0,1,0;...
0,1,1,0];

fruit_learned = {'Portakal','Unknown','Unknown','Unknown'};

%net_train_inputs = get_size(fruit_database);
net_train_input=database;
%-----------------------------------------------
%creat_network;
%feedforwardnet(hiddensizes,trainfcn)
%-----------------------------------------------
net=feedforwardnet(10,'trainlm');
%-----------------------------------------------
% training the neural network by using a train function
%-----------------------------------------------
net=train(net,net_train_input,target);

while(1)
disp('>> Bir resim seçmek için ENTER(Giris)tusuna basin.')
input('');
[file_name,path]=uigetfile();
new_pic = imread([path,file_name]);
disp('>> Lütfen bekleyin ...')
%-------------------------------------------------------------------------
%Noise alimi
%-------------------------------------------------------------------------
h = 3.0; % global smoothing parameter
alpha = 0.8; % structure sensitiviy parameter
new_pic = uint8(GLAS_RGB(new_pic,alpha,h));
%-------------------------------------------------------------------------
```

```
new_pic = imresize(new_pic,[200,200]);
[winner_bin,color] = color_seperation(new_pic);
check=[reshape(winner_bin,40000,1);color];
net_input=get_size(check);
%--------------------------------------------------------------------------
%Convert the image to black and white
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bwImg= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');

for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end

%Determine  Round.
stats = regionprops(LI,'alan','Centroid');

threshold = 0.94;
```

```
boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;

roundness = 4*pi*alan/cevre^2;
%--------------------------------------------------------------------------

net_input(3,1)=cevre;
net_input(4,1)=alan;
net_input(5,1)=roundness;
net_out=round(net(net_input));
compare_match=0;
for i=1:4
 if net_out == fruit_codes(:,i)
f_n=fruit_learned{i};
 compare_match=1;
break;
 end
end
if compare_match==0
for i=1:4
if strcmp(fruit_learned{i},'Unknown')
f_n=fruit_learned{i};
break;
end
end
end
if strcmp(f_n,'Unknown')
```

```
imshow(new_pic)
for qwe=1:5
beep
pause(1)
end
reply=input('>> Sinir Agi Bu meyveyi tanimiyor.\n>> Agi egitmek için meyvenin adini giriniz : ','s');
for j=1:4
if strcmp(fruit_learned{j},reply)
i=j;
break;
end
end

fruit_learned{i}=reply %#ok
fruit_database = [database,net_input];
net_train_inputs = fruit_database;
ntisize=size(net_train_inputs);
a=net_train_inputs(:,ntisize(2));
a=[a a a a a a a];%#ok
a=a+((2*rand(5,8)-1)/1000);
net_train_input=[net_train_input,a];%#ok



target=[target,fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i)];%#ok

clear net
net=feedforwardnet(10,'trainlm');

for j=1:10
```

```
[mynet{1,j},mynet{2,j}]=train(net,net_train_input,target);%#ok
bestperf(j)=mynet{2,j}.best_perf;%#ok
bestvperf(j)=mynet{2,j}.best_vperf;%#ok
besttperf(j)=mynet{2,j}.best_tperf;%#ok
end

 [minimum_perf,index]=min(bestvperf);
net=mynet{1,index};

disp('Öurenme bitti.')

else
imshow(new_pic);
text=['>> Sinir Agi "' ,fruit_learned{i}, '" olarak bu meyveyi tanir. \n>> Eger dogruysa
Y ve Aksi takdirde N girin: '];
for qwe=1:5
beep
pause(1)
end
reply=input(text,'s');
if strcmp(reply,'N')
reply=input('>> Lütfen meyvenin adini girin: ','s');
compare_match=0;
for j=1:4
if strcmp(fruit_learned{j},reply)
i=j;
compare_match=1;
break;
end
end
if compare_match==0
for j=1:4
```

```
if strcmp(fruit_learned{j},'Unknown')
i=j;
break;
end
end
end

fruit_learned{i}=reply %#ok
fruit_database = [database,net_input];%#ok
net_train_inputs = fruit_database;
ntisize=size(net_train_inputs);
a=net_train_inputs(:,ntisize(2));
a=[a a a a a a a];%#ok
a=a+((2*rand(5,8)-1)/1000);
net_train_input=[net_train_input,a];%#ok


target=[target,fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(
:,i),fruit_codes(:,i),fruit_codes(:,i),fruit_codes(:,i)];%#ok

clear net
net=feedforwardnet(4,'trainlm');
for j=1:10
[mynet{1,j},mynet{2,j}]=train(net,net_train_input,target);%#ok
bestperf(j)=mynet{2,j}.best_perf;%#ok
bestvperf(j)=mynet{2,j}.best_vperf;%#ok
besttperf(j)=mynet{2,j}.best_tperf;%#ok
end

 [minimum_perf,index]=min(bestvperf);
net=mynet{1,index};
```

```
disp('>> Ögrenme bitti.')
else
disp('>>Tesekkür ederim.')
end
end
end
```

### 3.5.2. Yazılımın genel işlemi

Bu çalışmada görüntülerin sınıflandırma işlemi iki ana bölümden oluşur.

- Görüntü işlemleri: Bulanık mantık yardımı ile yapılır.
- Veri madeniliği: Yapay sinir ağları yardımı ile yapılır.

### 3.5.3. Yazılımın süreci

Görüntünün tüm tanıma süreci 4 adımda gerçekleşir:

a) Resmin ön işlemi yapılır.
b) Resmin özellikleri çıkarılır.
c) Ayıklanan özelliklere göre sinir ağı oluşturulur.
d) Sinir ağının eğitimi verilir.

### 3.5.3.a. Ön işleme

Başlangıçta sisteme girilen görüntü ilgisiz ve yararsız verileri içerir. Bu nedenle, görüntünün özelliklerini çıkarmadan önce üzerinde ön işlem yapılmalıdır. Ön işleme aşağıdaki aşamaları içerir:

1. Bozukluk (noise) alımı.
2. Yeniden boyutlandırma.
3. RGB (red, green and blue) renk sistemini HSL'ye (hue, saturation and lightness) dönüştürme.

**1. Bozukluk alımı**

Resimler genellikle analog ortamlardan dijital ortamlara geçirildiği için bozukluk (noise) içerir. Görüntü işleme bu hataları düzeltmek için kullanılabilir. Bu yazılımda Xiang Zhu ve Peyman Milanfar tarafından sunulan bir yöntemle girilen görüntülerin gürültüsü alınır.

```
h = 3.0;  % global smoothing parameter
alpha = 0.8; %structure sensitiviy parameter
new_pic = uint8(GLAS_RGB(new_pic,alpha,h));
```

**2. Yeniden boyutlandırma**

İşlem hızını artırmak ve ayrıca tüm resimlerin eşit boyutta olmaları için her resim yazılıma girdikten sonra 200 x 200 piksele dönüştürülür ve toplamda 40000 piksel olur.

```
new_pic = imresize(new_pic,[200,200]);
```

**3. RGB (red, green and blue) renk sistemini HSL'ye (hue, saturation and lightness) dönüştürme**

HSL renk uzayı RGB uzayına göre insan görü düzeneğine daha yakın bir yapı oluşturmaktadır. HSL, RGB renk uzayından doğrusal olmayan bir dönüşüm ile elde edilir. Bu fonksiyon, girdi olarak bir pikselin RGB değerini alır ve çıktı olarak pikselin HSL renk değerlerini verir. Aşağıdaki dönüşüm denklemleri yardımıyla RGB ile HSL'nin silindir biçimi arasında dönüşüm gerçekleştirilebilir:

$$H \in \{0,360\}, S, V, R, G, B \in \{0,1\}$$

RGB'den HSL'ye:

$$MAXO = \max\{R, G, B\}, \qquad MINO = min\{R, G, B\}$$

$$H = \begin{cases} tanimsiz, & eger\ MAXO = MINO \\ 60\dfrac{G-B}{MAXO-MINO} + 0, & eger\ MAXO = R \quad ve \quad G \geq B \\ 60\dfrac{G-B}{MAXO-MINO} + 360, & eger\ MAXO = R \quad ve \quad G < B \\ 60\dfrac{B-R}{MAXO-MINO} + 120, & eger\ MAXO = G \\ 60\dfrac{R-G}{MAXO-MINO} + 240, & eger\ MAXO = B \end{cases}$$

$$S = \begin{cases} 0, & eger\ MAXO = 0 \\ 1 - \dfrac{MINO}{MAXO}, & degilse \end{cases}$$

$$L = \frac{1}{2}(M + m)$$

rgb_t2o_hsl function

```
function hslcolor=rgb_t2o_hsl(R,G,B)
R=R/255;
G=G/255;
B=B/255;

Cmax=max([R,G,B]);
Cmin=min([R,G,B]);
del=Cmax-Cmin;
if del==0
H=0;
else
switch Cmax
case R
H=60*mod(((G-B)/del),6);
case G
H=60*(((B-R)/del)+2);
case B
H=60*(((R-G)/del)+4);
end
end
H=2*(H-(H*(239/358)));
L=(Cmax+Cmin)/2;

if del==0
S=0;
else
S=(del/(1-abs(2*L-1)))*240;
end
L=L*240;
H=round(H);
```

```
S=round(S);
L=round(L);
if S>=245
S=244;
end
if L>=245
L=244;
end
if H>=245
H=244;
end
hslcolor(1)=H;
hslcolor(2)=S;
hslcolor(3)=L;


return
```

### 3.5.3.b. Özellikleri çıkarma

Görüntünün özelliklerini ölçmek, görüntülerin sınıflandırılmasının ve desen tanımının temelidir. Görüntünün özellikleri, karar kurallarının geliştirilmesi için ayıklanır ve bu kuralları kullanarak, görüntünün analiz ve yorumu sağlanır. Bu yazılımda özellik çıkarma aşaması 5 adımda yapılır:

1. Her pikselin rengi belirlenir.
2. Resimde baskın olan renk belirlenir.
3. Resimdeki meyve şeklinin genişliğinin uzunluğuna oranı hesaplanır.
4. Resimdeki meyve şeklinin alanı ve çevresi hesaplanır.
5. Resimdeki meyve şeklinin yuvarlaklık miktarı hesaplanır.

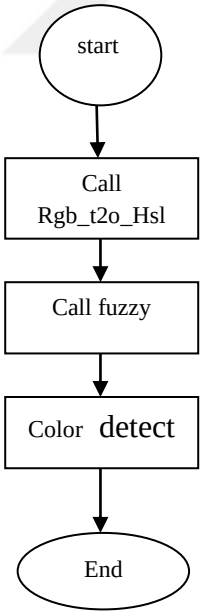
**1. Her pikselin rengini belirlemek**

Her pikselin rengini belirlemek için color _ detect fonksiyonu kullanılır. Bu fonksiyon, girdi olarak görüntüdeki her pikselin RGB değerlerini alır ve rgb_t2o_hsl fonksiyonunu kullanarak HSL renk değerlerini hesaplar. Hesaplanan değerler bulanık mantığa uygulanır. Daha sonra, bulanık mantığın çıktısını analiz edip pikselin rengini belirler.

Color_detect fonksiyonu üç bölümden oluşur:

HSL değerleri, her bir piksel için belirlenir.

1. Her pikselin renk değerleri bulanık mantık kullanılarak belirlenir.
2. Bulanık mantığın çıktısına göre, her pikselin rengi belirlenir.

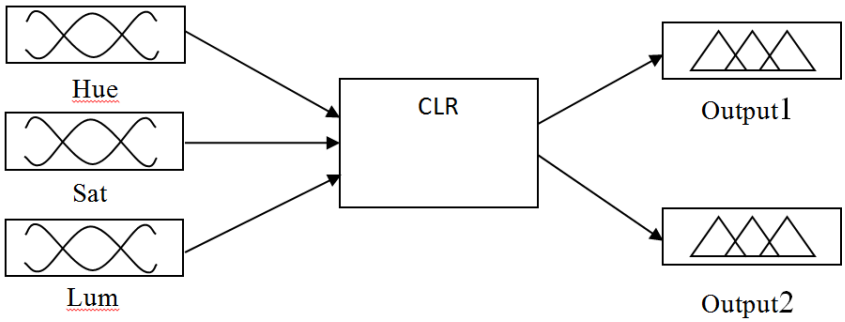



**a. HSL değerlerini belirlemek**

Yukarıda açıklandığı gibi, rgb_t2o_hsl fonksiyonu ile her bir piksel için RGB değerleri, HSL'ye dönüştürülür ve bulanık mantığa uygulanır.

**b. Bulanık mantık**

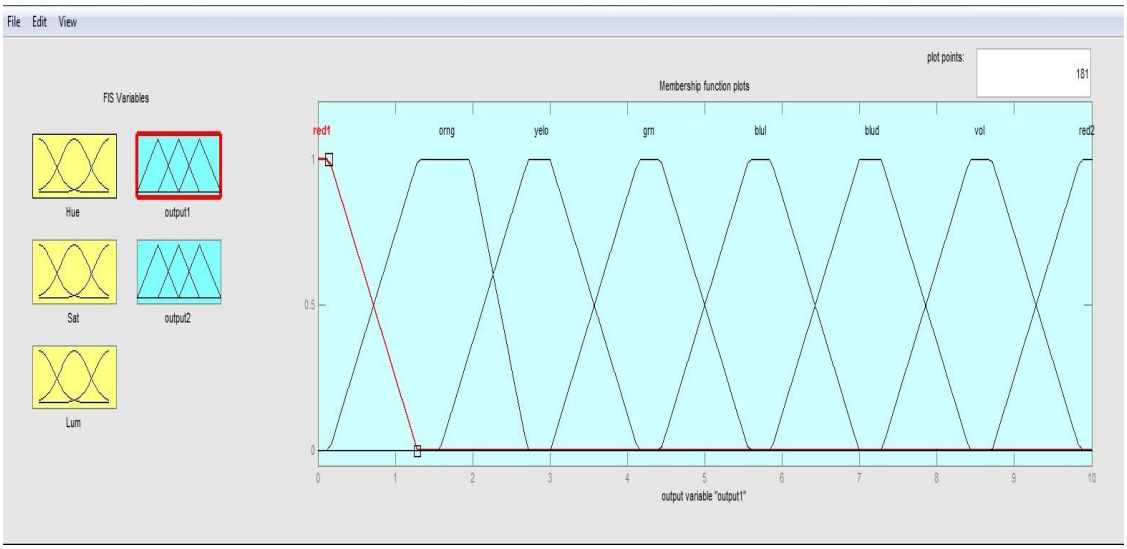
Bulanık mantık kullanarak her pikselin renk değeri tespit edilir. Bulanık mantık, üç girdi ile iki çıktıdan oluşmaktadır. (Şekil 3.10)



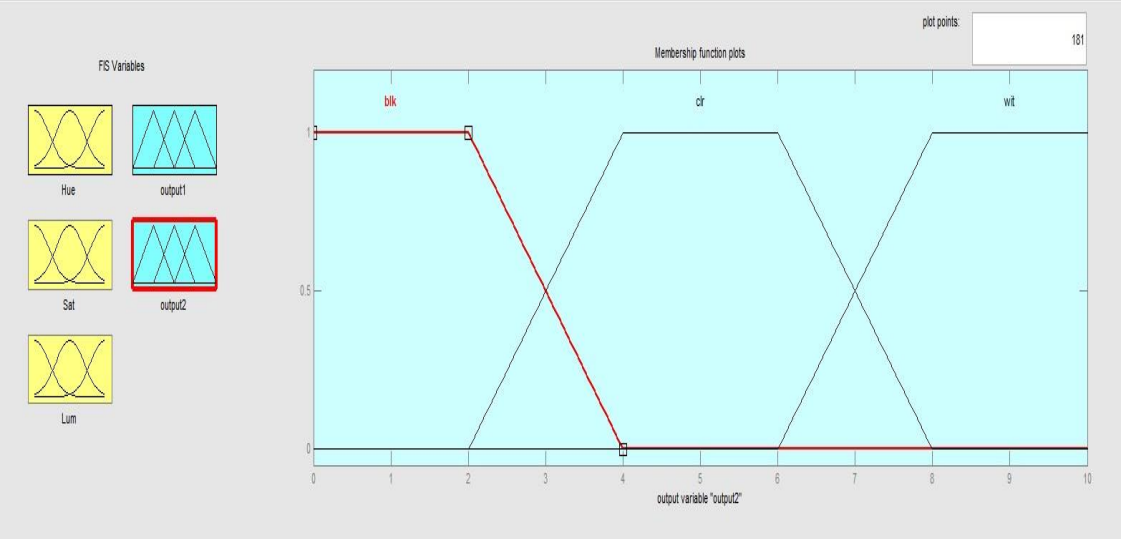

**Şekil 3.10.** Bulanık mantık

Girişler, her pikselin H,S,L değerlerini ve çıkışlar aşağıdakileri içerir.

- 1: Pikselin renk aralığı. (Şekil 3.11)
- 2: Pikselin siyah, beyaz veya renkli olması. (Şekil 3.12)




**Şekil 3.11.** Bulanık mantığın birinci çıkışı




**Şekil 3.12.** Bulanık mantığın ikinci çıkışı

fuzzy funtıon(clr)

[System]
Name='clr'
Type='mamdani'
Version=2.0
NumInputs=3
NumOutputs=2
NumRules=14
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Hue'
Range=[-5 245]

NumMFs=8
MF1='red1':'trapmf',[-10 -5 2 20]
MF2='orng':'trapmf',[5 15 25 35]
MF3='yelo':'trapmf',[18.6772486772487 33.6772486772487 43.6772486772487 58.6772486772487]
MF4='grn':'trapmf',[45 60 100 115]
MF5='blu_l':'trapmf',[100 110 135 145]
MF6='blu_d':'trapmf',[135 150 180 195]
MF7='vlt':'trapmf',[190 200 205 215]
MF8='red2':'trapmf',[210 225 245 260]

[Input2]
Name='Sat'
Range=[-5 245]
NumMFs=2
MF1='gry':'trapmf',[-10 -5 30 50]
MF2='Ngry':'trapmf',[30 50 245 255]

[Input3]
Name='Lum'
Range=[-5 245]
NumMFs=3
MF1='blk':'trapmf',[-10 -5 75 90]
MF2='color':'trapmf',[75 90 200 220]
MF3='wht':'trapmf',[200 220 245 250]

[Output1]
Name='output1'

Range=[0 10]

NumMFs=8

MF1='red1':'trapmf',[-1.286 -0.1429 0.1429 1.286]

MF2='orng':'trapmf',[0.1429 1.286 1.571 2.714]

MF3='yelo':'trapmf',[1.571 2.714 3 4.143]

MF4='grn':'trapmf',[3 4.143 4.429 5.571]

MF5='blul':'trapmf',[4.429 5.571 5.857 7]

MF6='blud':'trapmf',[5.857 7 7.286 8.429]

MF7='vol':'trapmf',[7.286 8.429 8.714 9.857]

MF8='red2':'trapmf',[8.714 9.857 10.14 11.29]

[Output2]

Name='output2'

Range=[0 10]

NumMFs=3

MF1='blk':'trapmf',[-1 0 2 4]

MF2='clr':'trapmf',[2 4 6 8]

MF3='wit':'trapmf',[6 8 10 11]

[Rules]

0 0 1, 0 1 (1) : 1

0 0 2, 0 2 (1) : 1

0 0 3, 0 3 (1) : 1

1 0 0, 1 0 (1) : 1

2 0 0, 2 0 (1) : 1

3 0 0, 3 0 (1) : 1

4 0 0, 4 0 (1) : 1

5 0 0, 5 0 (1) : 1

6 0 0, 6 0 (1) : 1

7 0 0, 7 0 (1) : 1

8 0 0, 8 0 (1) : 1

0 1 1, 0 1 (1) : 1

0 1 3, 0 3 (1) : 1

0 1 2, 0 1 (1) : 1

**c. Her pikselin rengini belirlemek**

Bulanık mantık çıktısına göre her pikselin rengi tespit edilip, color değişkeninde kaydedilir.

| 1: white | 2: black | 3: red | 4: orange | 5: yellow | 6: green | 7: light blue | 8: dark blue | 9: violet |
|---|---|---|---|---|---|---|---|---|

color_detect function

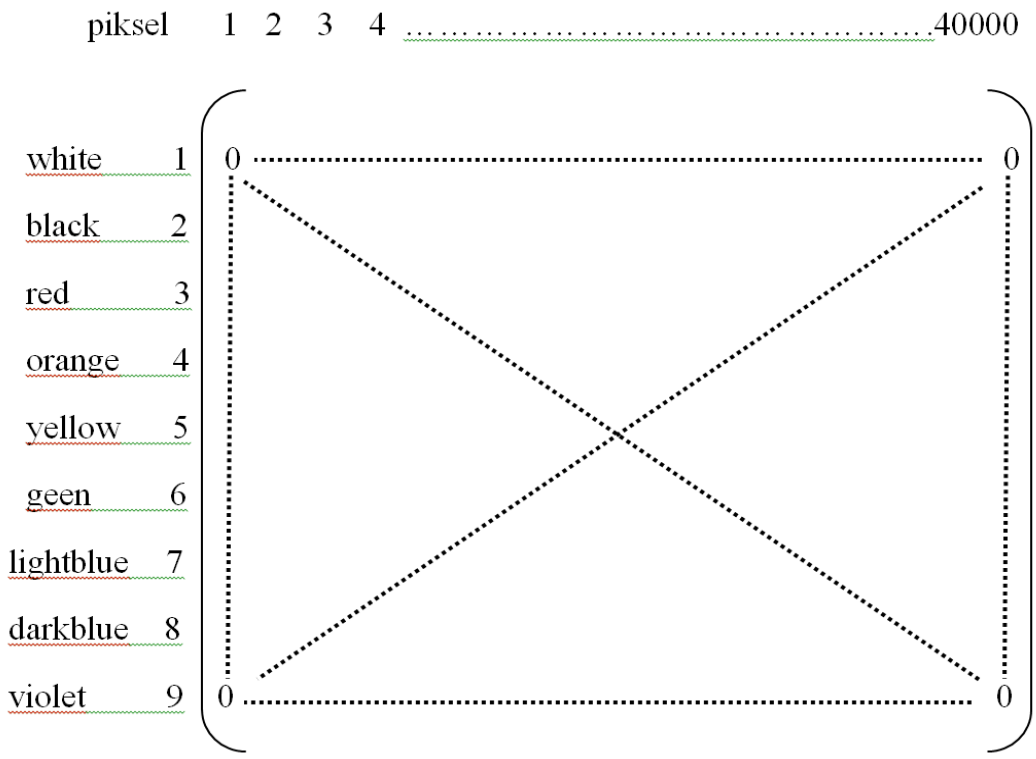
```
function color=color_detect(r,g,b)

hsl_color = rgb_t2o_hsl(r,g,b);

fuzzy_color_detector = readfis('clr');

color_number=evalfis(hsl_color,fuzzy_color_detector);

if color_number(2)<=3.54

color = 'black';

elseif color_number(2)>6.46

color = 'white';

else

if color_number(1)>=9.29 || color_number(1)<=0.65
```

```
color = 'red';
elseif color_number(1) > 0.65 && color_number(1) <= 2.1
color = 'orange';

elseif color_number(1) >2.1&& color_number(1) <= 3.55
color = 'yellow';

elseif color_number(1) > 3.55 && color_number(1) <= 4.965
color = 'green';

elseif color_number(1) > 4.965 && color_number(1) <= 6.425
color = 'light blue';

elseif color_number(1) > 6.425 && color_number(1) <= 7.855
color = 'dark blue';

elseif color_number(1) > 7.855 && color_number(1) < 9.29
color = 'violet';
else
color = 'unknown';
end
```

end

**2. Resimde baskın rengi belirlemek**

Color-seperation fonksiyonu ile görüntüdeki baskın olan renk tespit edilir. Bu fonksiyon ilk önce 9 x 40000 boyutunda sıfır değerlerinden oluşan bir matris oluşturur (her satır bir rengi temsil eder). Ardından girdi olarak 200 x 200 piksellik bir resmi alır ve oluşturduğu matris içinde her pikselin rengini 1 ile işaretler. Son olarak, hangi satırda bir sayısı fazla ise o satırın rengi baskın renk olarak tanımlanır. Bulunan bu renk, yapay sinir ağının girdilerinin birisini oluşturur.




color_seperation function:

function [output,color] = color_seperation(input)
color_pic_matrix=zeros(9,40000);

```
white=1;black=2;red=3;orange=4;yellow=5;green=6;lightblue=7;darkblue=8;violet=9;
input=double(input);
d = com.mathworks.mlwidgets.dialog.ProgressBarDialog.createProgressBar('run
program ...', []);
d.setSpinnerVisible(true);
d.setCircularProgressBar(false);
d.setCancelButtonVisible(true);
d.setVisible(true);
d.setProgressStatusLabel('renk tanima ...');
sq=0;
for i=1:200
d.setValue(sq);
for j=1:200
r1=input(i,j,1);
g1=input(i,j,2);
b1=input(i,j,3);
pixel_color = color_detect(r1,g1,b1);
switch pixel_color
case 'white'
color_pic_matrix(white,(i-1)*200+j)=1;
case  'black'
color_pic_matrix(black,(i-1)*200+j)=1;
case  'red'
color_pic_matrix(red,(i-1)*200+j)=1;
case  'orange'
color_pic_matrix(orange,(i-1)*200+j)=1;
case  'yellow'
color_pic_matrix(yellow,(i-1)*200+j)=1;
case  'green'
color_pic_matrix(green,(i-1)*200+j)=1;
case  'light blue'
color_pic_matrix(lightblue,(i-1)*200+j)=1;
```

```
case  'dark blue'
color_pic_matrix(darkblue,(i-1)*200+j)=1;
case  'violet'
color_pic_matrix(violet,(i-1)*200+j)=1;
end
end
sq=sq+0.0049;
end
for i=1:9
number_of_bits(i)=nnz(color_pic_matrix(i,:));%#ok
end
[value,color]=max(number_of_bits(3:9));%#ok
output = reshape(color_pic_matrix(color+2,:),200,200);
color = color+2;
d.setVisible(false);
return
```

**3. Resimdeki meyve şeklinin genişliğinin uzunluğuna oranını hesaplamak**

Get_size fonksiyonunun yardımıyla resimdeki meyve şeklinin genişliğinin uzunluğuna oranını hesaplanır. Girdi olarak, Color-seperation fonksiyonunun çıktısını alır ve aranan oranı bulur. Bulduğu bu oran yapay sinir ağının girdilerinin birisini oluşturur.

Get_size  function:

```
function out=get_size(pics)
sns=6;
[R,W]=size(pics);
CLR=pics(R,:);
pics(R,:)=[];
for k=1:W
pic=reshape(pics(:,k),[200,200]);
```

```
 [r,w]=size(pic);
for i=1:r
if nnz(pic(i,:)) > sns && nnz(pic(i+1,:)) > sns
up=i;
%up_ind=find(pic(i,:));
break
end
end
for i=1:w
if nnz(pic(:,i)) > sns && nnz(pic(:,i+1)) > sns
left=i;
%left_ind=find(pic(:,i));
break
end
end
for i=r:-1:1
if nnz(pic(i,:)) > sns && nnz(pic(i-1,:)) > sns
down=i;
%down_ind=find(pic(i,:));
break
end
end
for i=w:-1:1
if nnz(pic(:,i)) > sns && nnz(pic(:,i-1)) > sns
right=i;
%right_ind=find(pic(:,i));
break
end
end
out(1:2,k)=[abs(up-down),abs(left-right)];
end
out=[(out(2,:)./out(1,:));CLR];
```

return

**4. Resimdeki meyve şeklinin alan ve çevresini hesaplamak**

Resimdeki meyve şeklinin alan ve çevresini hesaplamak için görüntü siyah ve beyaza dönüştürülür. 30'dan az piksel içeren tüm nesneler kaldırılır, görüntünün bölgeleri ve delikleri doldurulur, nesnenin sınırları bulunur. Bu sınır, nesnenin çevresini; içi de nesnenin alanını oluşturur.

```
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bw Img= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');

for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end

%Determine  Round.
```

```
stats = regionprops(LI,'alan','Centroid');

threshold = 0.94;

boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;
```

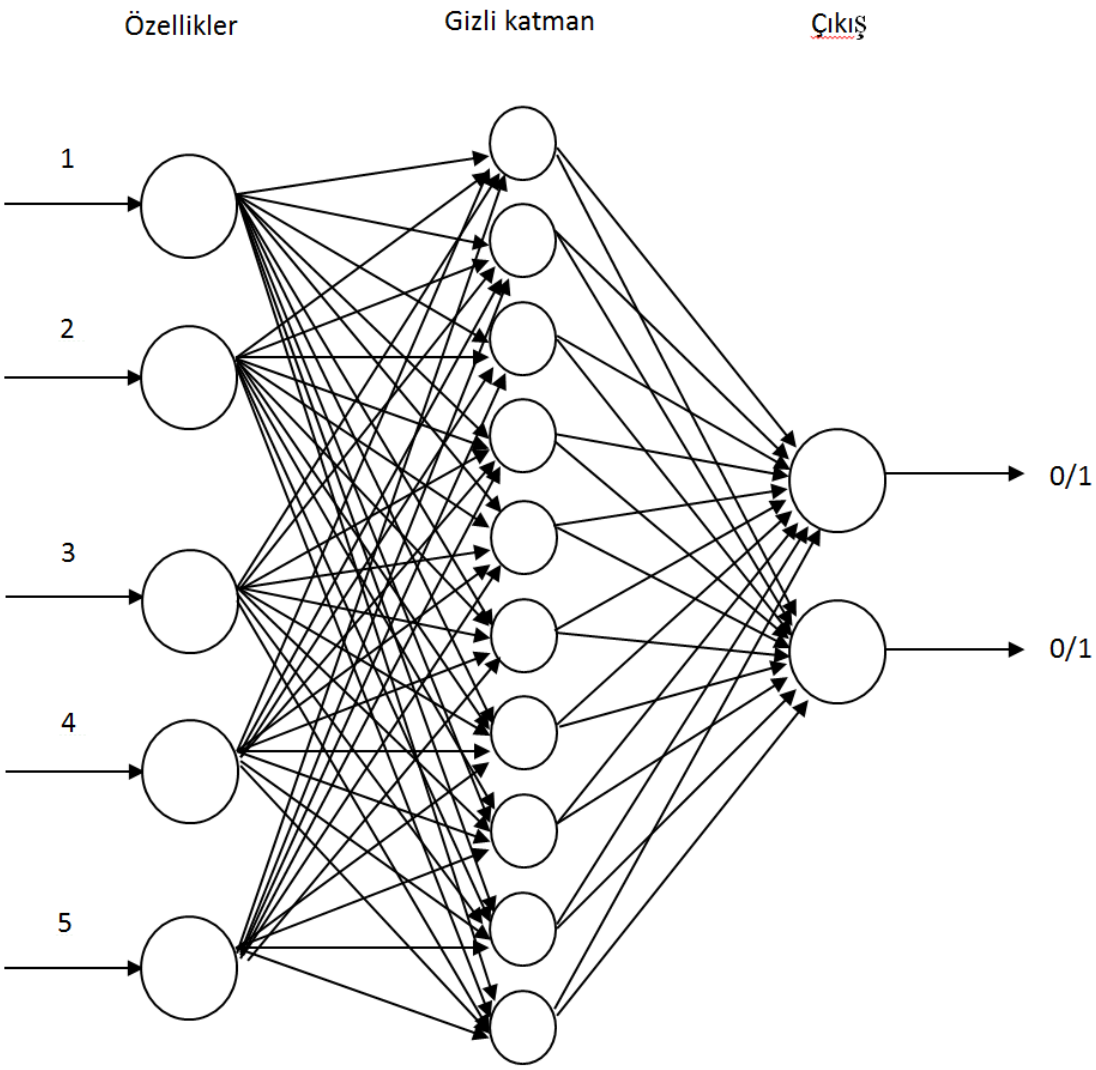
**5. Resimdeki meyve şeklinin yuvarlaklık miktarını hesaplamak**

Meyve şeklinin yuvarlaklığı veya metrik değerini hesaplamak için alan ve çevresi kullanılır ve aşağıdaki denklemle elde edilir:

$$\text{Roundness} = 4\pi\left(\frac{\text{Area}}{Perimeter^2}\right)$$

**3.5.3.c. Yapay sinir ağını oluşturmak**

Sistemin eğitimi için sinir ağı algoritması kullanılmıştır. Bu ağı oluşturmak için bir dizi girişler, çıkışlar ve gizli bir katman lazımdır. Ancak her katmandaki nöron sayısının nasıl seçileceği önemlidir. Seçilen özellik sayısına bağlı olarak sinir ağının girişi belirlenir. Bu uygulamada her nesneyi tanımlamak ve görüntüyü sınıflandırmak için görüntünün içindeki nesnenin 5 özelliği ayıklanmıştır. Bu yüzden giriş katmanında 5 nörona ihtiyaç vardır. Seçilen özelliklere göre gizli katman 10 nöronla oluşturulur ve son olarak çıktıda 2-bitlik bir sayı gerekli olduğundan, çıkış katmanı 2 nöronla oluşur. (Şekil 3.13)




**Şekil 3.13.** Yapay sinir ağı

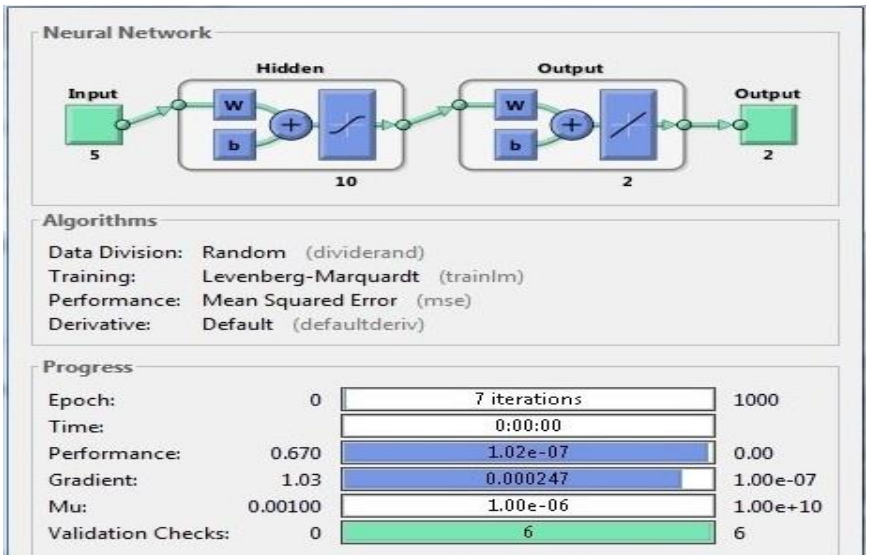
```
%-------------------------------------------------

%creat_network;

%feedforwardnet(hiddensizes,trainfcn)

%-------------------------------------------------

net=feedforwardnet(10,'trainlm');
```

### 3.5.3.d. Sinir ağının eğitimi

Geri yayılma algoritması, basitliği ve uygulamadaki görüş açısı gibi başarılarından dolayı ağ eğitimi için en popüler algoritmalardan biridir. Bu algoritma; hataları geriye doğru çıkıştan girişe doğru azaltmaya çalışmasından dolayı geri yayılım ismini almıştır. Geri yayılmalı öğrenme kuralı ağ çıkışındaki mevcut hata düzeyine göre her bir tabakadaki ağırlıkları yeniden hesaplamak için kullanılmaktadır. Bir geri yayılımlı ağ modelinde giriş, gizli ve çıkış olmak üzere 3 katman bulunmakla birlikte, problemin özelliklerine göre gizli katman sayısını artırabilmek mümkündür. Geri yayılım çok katmanlı ağlarda kullanılan delta kuralı için genelleştirilmiştir bir algoritmadır. Bu algoritma çok katlı ağlarda hesap işlerini öğrenmede kullanılabilmektedir. Geri yayılım ağında hatalar, ileri besleme aktarım işlevinin türevi tarafından, ileri besleme mekanizması içinde kullanılan aynı bağlantılar aracılığıyla, geriye doğru yayılmaktadır. Öğrenme işlemi, bu ağda basit çift yönlü hafıza birleştirmeye dayanmaktadır Bu çalışmada yapay sinir ağı (YSA) eğitimi LM algoritması ile sunulmuştur. (Şekil 3.14)

training the neural network by using a trainlm function

net=trainlm(net,net_train_input,target);




**Şekil 3.14.** Sinir ağının eğitimi

### 3.6. Diğer Algoritmalar İle Yazılımın Tanıtımı

Tezin bu bölümünde aşağıda belirttiğimiz diğer öğrenme algoritmaları ile yazılımı çalıştırıp, yapay sinir ağları algoritmasıyla karşılaştıracağız.

#### 3.6.1. K en yakın komşu algoritması (KNN)

Sınıflandırmada (classification) kullanılan bu algoritmaya göre sınıflandırma sırasında çıkarılan özelliklerden (feature extraction), sınıflandırılmak istenen yeni bireyin daha önceki bireylerden k tanesine yakınlığına bakılmasıdır. Örneğin k=3 için yeni bir eleman sınıflandırılmak istensin. Bu durumda eski sınıflandırılmış elemanlardan en yakın 3 tanesi alınır. Bu elemanlar hangi sınıfa dâhilse, yeni eleman da o sınıfa dâhil edilir. Mesafe hesabında genelde Öklid mesafesi (Euclid distance) kullanılabilir.

Algoritmanın adımları;

1- Yeni gelen birey sınıfa eklenir.
2- k komşusuna bakılır.
3- Çeşitli uzaklık fonksiyonları kullanılarak uzaklık hesaplanır.
4- En yakın neresi ise birey oraya atanır.

Avantajları:

- Uygulanabilirliği basit bir algoritmadır.
- Gürültülü eğitim dokümanlarına karşı dirençlidir.
- Eğitim dokümanları sayısı fazla ise etkilidir.

Dezavantajları:

- K parametresine ihtiyaç duyar.

- En iyi sonuçları elde etmek için, hangi uzaklık tipinin ve hangi niteliğin kullanılacağı konusunda açık değildir.
- Hesaplama maliyeti gerçekten çok yüksektir çünkü her bir sorgu örneğinin tüm eğitim örneklerine olan uzaklığını hesaplamak gerekmektedir. Bazı indeksleme metotları ile (örneğin K-D ağacı), bu maliyet azaltılabilir.
- En yakın komşuluk prensibine dayanır. Tüm dokümanlar vektörel olarak temsil edilir. Sorgu dokümanı ile diğer dokümanlar arasındaki kosinüs benzerliği hesaplanır. Benzerlik oranı 1'e en yakın olan n tane vektörün kategorisinden çok olanı dokümana atanır.

Knn uygulaması:

```
clear all;
close all;
clc;

load('knn_train');

    compare_match=0;
    norm_var=[  1     1     1     1     1     1     1     1;...
    0.1   0.1   0.1   0.1   0.1   0.1   0.1   0.1];

    while(1)
    disp('>> Bir resim seçmek için ENTER(Giris)tusuna basin.')
    input('');
    [file_name,path]=uigetfile();
    new_pic = imread([path,file_name]);
    disp('>> Lütfen bekleyin ...')

    %Noise alimi
    h = 3.0; % global smoothing parameter
```

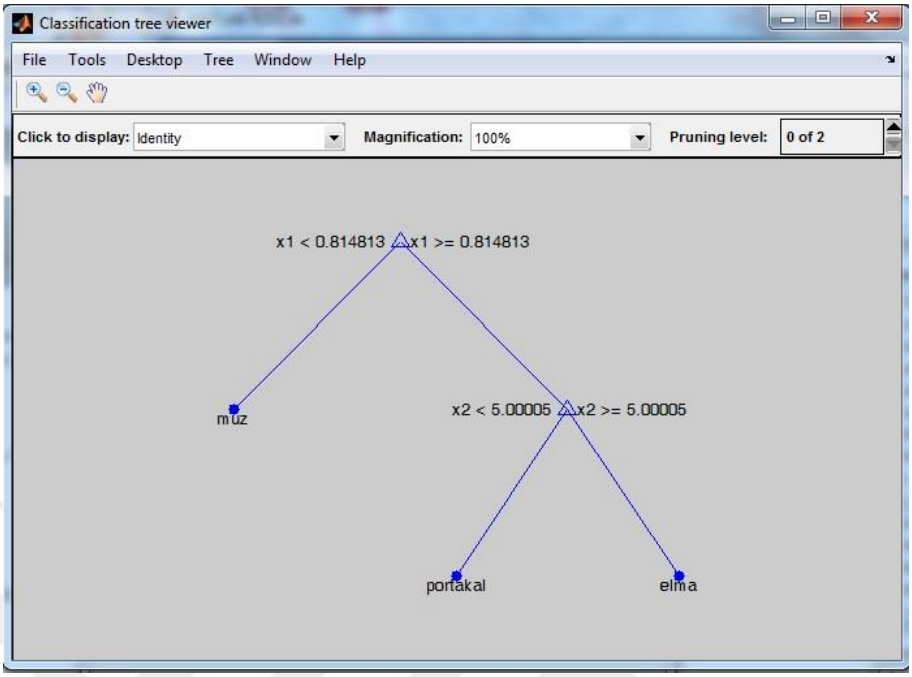
```
alpha = 0.8; % structure sensitiviy parameter
new_pic = uint8(GLAS_RGB(new_pic,alpha,h));

%Calculation of colors and sizes
new_pic = imresize(new_pic,[200,200]);
[winner_bin,color] = color_seperation(new_pic);
check=[reshape(winner_bin,40000,1);color];

%Convert the image to black and white
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bwImg= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');

for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end

%Determine  Round.
stats = regionprops(LI,'alan','Centroid');
```

```
threshold = 0.94;

boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;

roundness = 4*pi*alan/cevre^2;
%------------------------------------------------------------------------

knn_test=get_size(check);
knn_test=knn_test';
knn_test(1,3)=cevre;
knn_test(1,4)=alan;
knn_test(1,5)=roundness;

% X is your nx2 array of training data
% Y is your nx1 array of training labels
X=fruit_database;
Y=fruit_names;
k=5;
model = ClassificationKNN.fit(X,Y,'NumNeighbors',k);

q=0;
for i=1:15
if knn_test(1,2) == X(i,2);
predictedY = predict(model, knn_test);
q=1;
```

```
break;
end
end
if q==0
disp('>> Bu meyve tanimlanamadi')
else
text=['>> Bu meyva',predictedY,'olarak tanildi.'];
disp(text)
end
end
```

### 3.6.2. Karar ağacı öğrenmesi (Decision tree learning)

Karar ağacı öğrenmesi yöntemi, makine öğrenmesi (machine learning) konularından birisidir. Literatürde karar ağacı öğrenmesinin alt yöntemleri olarak kabul edilebilecek sınıflandırma ağacı (classification tree) veya ilkelleştirme ağacı (regression tree, tahmin ağacı) gibi uygulamaları vardır. Karar ağacı öğrenmesinde, bir ağaç yapısı oluşturularak ağacın yaprakları seviyesinde sınıf etiketleri ve bu yapraklara giden ve başlangıçtan çıkan kollar ile de özellikler üzerindeki işlemeler ifade edilmektedir. Karar ağacı öğrenmesi sırasında, öğrenilen bilgi bir ağaç üzerinde modellenir. Bu ağacın bütün iç düğümleri (interior nodes) birer girdiyi ifade eder. Karar ağacı öğrenmesinde, ağacın öğrenilmesi sırasında, üzerinde eğitim yapılan küme, çeşitli özelliklere göre alt kümelere bölünür, bu işlem, özyinele olarak (recursive) tekrarlanır ve tekrarlama işleminin tahmin üzerinde bir etkisi kalmayana kadar sürer. Bu işleme özyinele parçalama (recursive partitioning) ismi verilir. (Şekil 3.15)




**Şekil 3.15.** Karar ağacı

Karar Ağacı uygulaması:

```
clear all;
close all;
clc;

load('dt_train');

compare_match=0;
norm_var=[ 1    1    1    1    1    1    1    1;...
0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1];

while(1)
disp('>> Bir resim seçmek için ENTER(Giris)tusuna basin.')
input('');
[file_name,path]=uigetfile();
new_pic = imread([path,file_name]);
```

```
disp('>> Lütfen bekleyin ...')

%Noise alimi
h = 3.0; % global smoothing parameter
alpha = 0.8; % structure sensitiviy parameter
new_pic = uint8(GLAS_RGB(new_pic,alpha,h));

%Calculation of colors and sizes
new_pic = imresize(new_pic,[200,200]);
[winner_bin,color] = color_seperation(new_pic);
check=[reshape(winner_bin,40000,1);color];

%Convert the image to black and white
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bwImg= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');

for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
```

```
end

%Determine  Round.
stats = regionprops(LI,'alan','Centroid');

threshold = 0.94;

boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;

roundness = 4*pi*alan/cevre^2;
%------------------------------------------------------------------------

%# testing data
dt_test=get_size(check);
dt_test=dt_test';
dt_test(1,3)=cevre;
dt_test(1,4)=alan;
dt_test(1,5)=roundness;

%# construct predicting attributes and target class
vars = {'fruit_database'};
training = fruit_database;  %# mitrainінged continous/discrete data
class_labels = fruit_names;                    %# class labels

%# train classification decision tree
ctree = classregtree(training,class_labels);
```

```
view(ctree)

%# test
class_labelsPredicted = eval(ctree, training);
cm = confusionmat(class_labels,class_labelsPredicted);     %# confusion matritraining
N = sum(cm(:));
err = ( N-sum(diag(cm)) ) / N;            %# testing error

q=0;
for i=1:15
if dt_test(1,2) == training(i,2);
classifierout = eval(ctree, dt_test);
q=1;
break;
end
end
if q==0
disp('>> Bu meyve tanimlanamadi')
else
text=['>> Bu meyva',classifierout,'olarak tanildi.'];
disp(text)
end
end
```

### 3.6.3. Naive Bayes sınıflandırıcı

Naive Bayes sınıflandırma algoritması, adını matematikçi Thomas Bayes'den alan bir sınıflandırma algoritmasıdır. Naïve Bayes sınıflandırması olasılık ilkelerine göre tanımlanmış bir dizi hesaplama ile sisteme sunulan verilerin sınıfını yani kategorisini tespit etmeyi amaçlar.

Naïve Bayes sınıflandırmasında sisteme belirli bir oranda öğretilmiş veri sunulur (Ör: 100 adet). Öğretim için sunulan verilerin mutlaka bir sınıfı/kategorisi bulunmalıdır. Öğretilmiş veriler üzerinde yapılan olasılık işlemleri ile, sisteme sunulan yeni test verileri, daha önce elde edilmiş olasılık değerlerine göre işletilir ve verilen test verisinin hangi kategoride olduğu tespit edilmeye çalışılır. Elbette öğretilmiş veri sayısı ne kadar çok ise, test verisinin gerçek kategorisini tespit etmek o kadar kesin olabilmektedir.

Naïve Bayes sınıflandırma yönteminin birçok kullanım alanı bulunabilir. Fakat burada neyin sınıflandırıldığından çok nasıl sınıflandırıldığı önemlidir. Yani öğretilecek veriler binary veya text veriler olabilir. Burada veri tipinden ve ne olduğundan ziyade, bu veriler arasında nasıl bir oransal ilişki kurduğumuz önem kazanmaktadır.

Naive Bayes uygulaması:

```
clear all;
close all;
clc;

load('nb_train');

compare_match=0;
norm_var=[ 1    1    1    1    1    1    1    1;...
0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1];

while(1)
disp('>> Bir resim seçmek için ENTER(Giris)tusuna basin.')
input('');
[file_name,path]=uigetfile();
new_pic = imread([path,file_name]);
disp('>> Lütfen bekleyin ...')
```

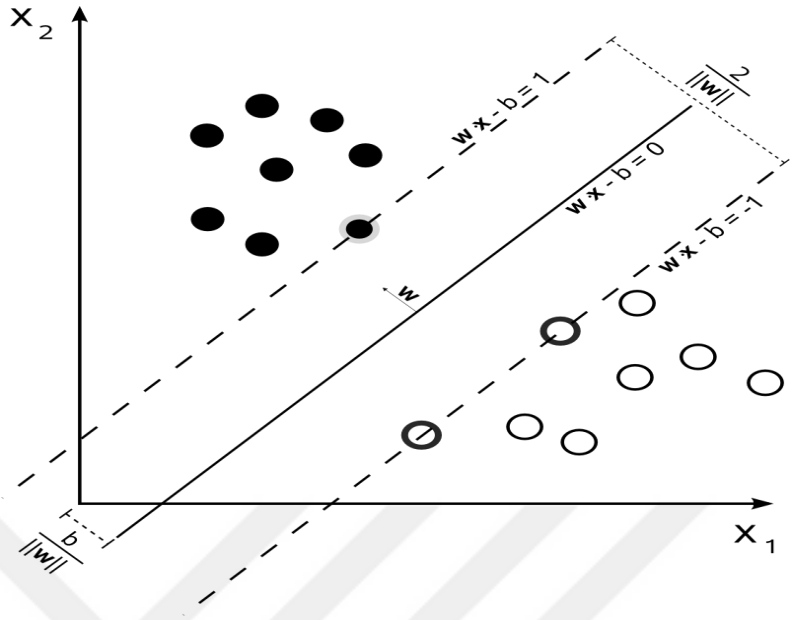
```
%Noise alimi
%h = 3.0; % global smoothing parameter
%alpha = 0.8; % structure sensitiviy parameter
%new_pic = uint8(GLAS_RGB(new_pic,alpha,h));

%Calculation of colors and sizes
new_pic = imresize(new_pic,[200,200]);
[winner_bin,color] = color_seperation(new_pic);
check=[reshape(winner_bin,40000,1);color];

%Convert the image to black and white
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bwImg= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');

for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end
```

```
%Determine  Round.
stats = regionprops(LI,'alan','Centroid');

threshold = 0.94;

boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;

roundness = 4*pi*alan/cevre^2;
%------------------------------------------------------------------------

%# testing data
nb_test=get_size(check);
nb_test=nb_test';
nb_test(1,3)=cevre;
nb_test(1,4)=alan;
nb_test(1,5)=roundness;

%# training data and class
training = fruit_database;
train_class = fruit_names;

%# train model
nb = NaiveBayes.fit(training, train_class);

q=0;
```

```
for i=1:15
if nb_test(1,2) == training(i,2);
classifierout=predict(nb,nb_test);
q=1;
break;
end
end
if q==0
disp('>> Bu meyve tanimlanamadi')
else
text=['>> Bu meyva',classifierout,'olarak tanildi.'];
disp(text)
end
end
```

### 3.6.4. Destek vektör makinesi (Support Vector Machine) (SVM)

Sınıflandırma konusunda kullanılan oldukça etkili ve basit yöntemlerden birisidir. Sınıflandırma için bir düzlemde bulunan iki grup arasında bir sınır çizilerek iki grubu ayırmak mümkündür. Bu sınırın çizileceği yer ise iki grubun da üyelerine en uzak olan yer olmalıdır. İşte SVM bu sınırın nasıl çizileceğini belirler. Bu işlemin yapılması için iki grubada yakın ve birbirine paralel iki sınır çizgisi çizilir ve bu sınır çizgileri birbirine yaklaştırılarak ortak sınır çizgisi üretilir. (Şekil 3.16)




**Şekil 3.16.** Destek vektör makinesi

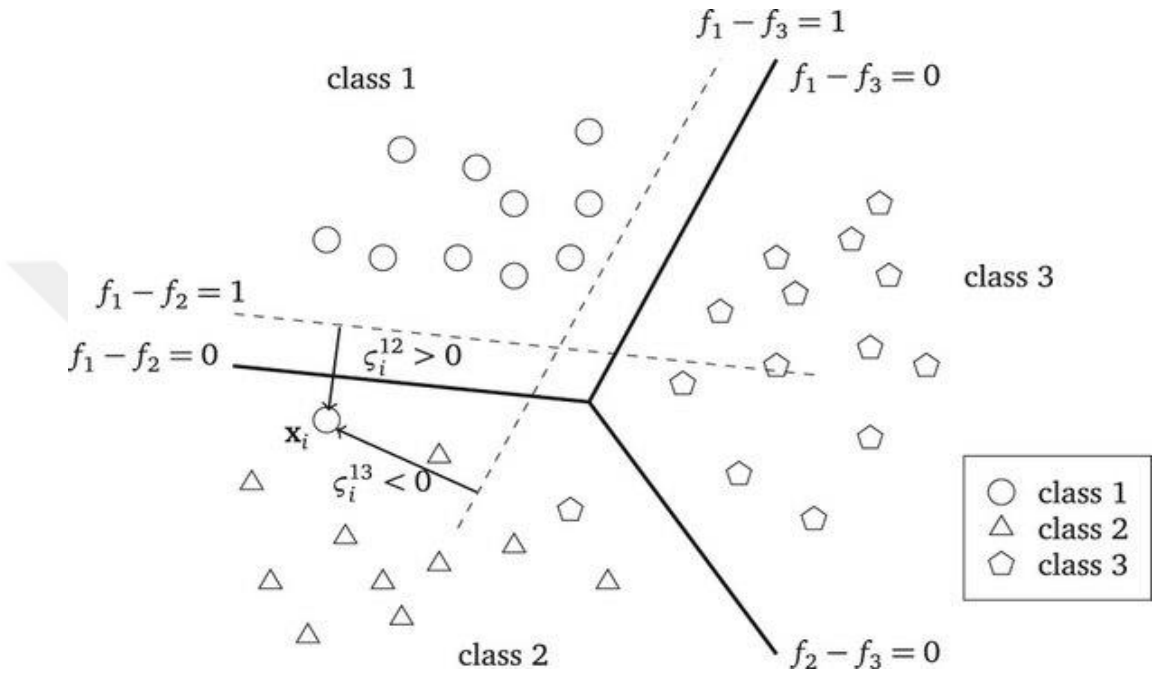
Bu şekilde iki grup iki boyutlu bir düzlem üzerinde gösterilmiştir. Bu düzlemi ve boyutları birer özellik olarak düşünmek mümkündür. Yani basit anlamda sisteme giren her girdinin (input) bir özellik çıkarımı (feature extraction) yapılmış ve sonuçta bu iki boyutlu düzlemde her girdiyi gösteren farklı bir nokta elde edilmiştir. Bu noktaların sınıflandırılması demek, çıkarılmış olan özelliklere göre girdilerin sınıflanması demektir.

Bilindiği üzere destek vektör makinaları iki sınıfın ayrılmasında kullanılmaktadır. Yani bir SVM marifetiyle iki sınıf belirli bir tolerans değerinde birbirinden ayrılmakta ve bu ayrım sonunda çıkarılmış olan öznitelik vektörlerine (feature vectors) göre aşırı düzlem (hyper plane) üzerinde iki düzlem elde edilmektedir.

Çok sınıflı SVM ile kast edilen ise ayrımı yapılacak (sınıflandırılacak) grupların ikiden fazla olması durumudur. Bu durumda aşağıdaki üç yaklaşımdan birisi tercih edilebilir:

- Problemin ikili gruplara indirgenmesi (bire bir yaklaşımı, one-to-one)

- Problemin tek gruptan bütün gruplara modellenmesi (bire çok yaklaşım, one-to-many)
- Çok sınıf sıralama SVM'leri (multiclass ranking SVM) (Şekil 3.17)




**Şekil 3.17.** Multi class SVM

Destek Vektör Makinesi uygulaması:

```
clear all;
close all;
clc;

load('svm_train');

compare_match=0;
norm_var=[ 1    1    1    1    1    1    1    1;...
0.1   0.1   0.1   0.1   0.1   0.1   0.1   0.1];

while(1)
```

```
disp(‘>> Bir resim seçmek için ENTER(Giris)tusuna basin.’)
input(‘’);
[file_name,path]=uigetfile();
new_pic = imread([path,file_name]);
disp(‘>> Lütfen bekleyin ...’)

%Noise alimi
h = 3.0; % global smoothing parameter
alpha = 0.8; % structure eparation parameter
new_pic = uint8(GLAS_RGB(new_pic,alpha,h));

%Calculation of colors and sizes
new_pic = imresize(new_pic,[200,200]);
[winner_bin,color] = color_seperation(new_pic);
check=[reshape(winner_bin,40000,1);color];

%Convert the image to black and white
RGBImg=new_pic;
Img = rgb2gray(RGBImg);
newImg = graythresh(Img);
bwImg = im2bw(Img,newImg);
bwImg=~bwImg;

bwImg = bwareaopen(bwImg,30);

seImg = strel('disk',2);
bwImg= imclose(bwImg,seImg);

bwImg = imfill(bwImg,'holes');

%Find the Boundaries
[BI,LI] = bwboundaries(bwImg,'noholes');
```

```
for ki = 1:length(BI)
boundary = BI{ki};
plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end

%Determine  Round.
stats = regionprops(LI,'alan','Centroid');

threshold = 0.94;

boundaryImg = BI{ki};

deltasq = diff(boundaryImg).^2;
cevre = sum(sqrt(sum(deltasq,2)));
cevre=round(cevre);

alan = stats(k).alan;

roundness = 4*pi*alan/cevre^2;
%------------------------------------------------------------------------

%# testing data
TestSet=get_size(check);
TestSet=TestSet';
TestSet(1,3)=cevre;
TestSet(1,4)=alan;
TestSet(1,5)=roundness;

[~,~,fruit_names] = unique(fruit_names);
u=unique(fruit_names);
numClasses=length(u);
```

```
result = zeros(length(TestSet(:,1)),1);

%build models
for k=1:numClasses
%Vectorized statement that binarizes Group
%where 1 is the current class and 0 is all other classes
G1vAll=(fruit_names==u(k));
models(k) = svmtrain(fruit_database,G1vAll);
end

%classify test cases
for j=1:size(TestSet,1)
for k=1:numClasses
if(svmclassify(models(k),TestSet(j))
break;
end
end
result(j) = k;
end
for i=1:15
if TestSet(1,2) == fruit_database(I,2);
switch result
case 1
classifierout=' elma ';
case 2
classifierout=' muz ';
case 3
classifierout=' portakal ';
end
q=1;
break;
end
```

```
end
if q==0
disp('>> Bu meyve tanimlanamadi')
else
text=['>> Bu meyva',classifierout,'olarak tanildi.'];
disp(text)
end
end
```

### 3.7. Sınıflandırma algoritmalarının karşılaştırılması

Veri madenciliği yöntemleri ile eldeki veriler sınıflandırılarak, gruplandırılarak ya da veriler arasında ilişkiler, bağıntılar, istatistiksel sonuçlar oluşturularak modeller oluşturulur. Oluşturulan model, oluşturulduğu veri kümesinde olmayan yeni bir kayıt geldiğinde, yeni gelen kayıt hakkında tahmin yapma imkânı verir. Yapılan tahminlerin doğruluk derecesi oluşturulmuş olan modelin veri üzerindeki başarımını ortaya koyar. Dolayısı ile bir veri madenciliği uygulamasında hangi algoritma ile daha iyi sonuçlar üretildiği uygulamanın başarımı açısından önemlidir. Ayrıca sürekli geliştirilmekte olan yeni algoritmaların başarım derecesinin var olan algoritma sonuçları ile karşılaştırılması yeni geliştirilen algoritmanın kabul edilebilirliğini ortaya koyması açısından önemlidir.

#### 3.7.1. Öğrenme ve test kümesinin seçimi

Model oluşturulurken kullanılan öğrenme ve test kümelerinin belirlenmesinin de modelin başarımı üzerinde etkisi vardır. Eldeki verinin öğrenme kümesi ve test kümesi olarak ayrılmasında farklı metotlar kullanılabilir. Kullanılan veri madenciliği programında bu işlem için farklı seçenekler bulunabilir. Öğrenme kümesi ve test kümesi farklı dosyalardan programa verilebileceği gibi, programın bir veri dosyasını belirtilen bir oranda test kümesi olarak kullanması ya da n-fold metodu ile programın veri kümesini n sayıdaki parçalara ayırarak sırayla her parçayı test kümesi olarak kullanması

sağlanabilir. Bu çalışma için 3000 meyve resminin özellikleri yukarıda bahsettiğimiz yöntemlerle ayıklanır ve ACCURACY adlı bir dosya halinde programa verilir. Bu dosyadan öğrenme ve test kümeleri seçilir ve programa uygulanır.

### 3.7.2. Model başarım ölçütleri

Model başarımını değerlendirirken kullanılan temel kavram doğrulukdur. (Accuracy) Modelin başarısı, doğru sınıfa atanan örnek sayısı ve yanlış sınıfa atılan örnek sayısı nicelikleriyle alakalıdır.

```
clc
clear all;

load accuracy

ord = randperm(size(fruit_database,1));
fruit_database = fruit_database(ord,:);
fruit_names = fruit_names(ord);
nn_class=target_class(:,ord);

%# lets split into training/testing
training = fruit_database(1:50,:);
testing = fruit_database(51:3000,:);
train_class = fruit_names(1:50);
test_class = fruit_names(51:3000);
nn_train_class=nn_class(:,1:50);
nn_test_class=nn_class(:,51:3000);
%# KNN Accuracy
knn_training=training;
knn_class_labels=train_class;
k=2;
```

```
knn_model = ClassificationKNN.fit(knn_training,knn_class_labels,'NumNeighbors',k);
knn_Accuracy=0;
for i=1:2950
knn_classifierout = predict(knn_model, testing(i,:));
if strcmp(knn_classifierout, test_class(i))==1;
knn_Accuracy=knn_Accuracy+1;
end
end
knn_Accuracy=(knn_Accuracy*100)/2950;
fprintf('KNN Classifier Accuracy: %.3f%%\n',knn_Accuracy )

%# Decision Tree Accuracy
vars = {'fruit_database'};
dt_training = training;
dt_class_labels = train_class;
ctree_model = classregtree(dt_training,dt_class_labels);
dt_Accuracy=0;
for i=1:2950
dt_classifierout = eval(ctree_model, testing(i,:));
if strcmp(dt_classifierout, test_class(i))==1;
dt_Accuracy=dt_Accuracy+1;
end
end
dt_Accuracy=(dt_Accuracy*100)/2950;
fprintf('Decision Tree Accuracy: %.3f%%\n',dt_Accuracy )

%# Naive Bayes Accuracy
nb_training = training;
nb_class_labels = train_class;
nb_model = NaiveBayes.fit(nb_training, nb_class_labels);
nb_Accuracy=0;
for i=1:2950
```

```
nb_classifierout=predict(nb_model,testing(i,:));
if strcmp(nb_classifierout, test_class(i))==1;
nb_Accuracy=nb_Accuracy+1;
end
end
nb_Accuracy=(nb_Accuracy*100)/2950;
fprintf('Naive Bayes Accuracy: %.3f%%\n',nb_Accuracy )
%# support vector machine Accuracy
svm_class_labels=train_class;
svm_testclass_labels=test_class;
[~,~,svm_class_labels] = unique(svm_class_labels);
[~,~,svm_testclass_labels] = unique(svm_testclass_labels);
u=unique(svm_class_labels);
numClasses=length(u);
result = zeros(length(testing(:,1)),1);
for k=1:numClasses
G1vAll=(svm_class_labels==u(k));
models(k) = svmtrain(training,G1vAll);
end
svm_Accuracy=0;
for j=1:2950
for k=1:numClasses
if(svmclassify(models(k),testing(j,:)))
break;
end
end
result(j) = k;
switch result(j)
case 1
svm_classifierout=' elma ';
case 2
svm_classifierout=' muz ';
```

```
case 3
svm_classifierout=' portakal ';
end
if result(j)== svm_testclass_labels(j);
svm_Accuracy=svm_Accuracy+1;
end
svm_classifierout;
end
svm_Accuracy=(svm_Accuracy*100)/2950;
fprintf('support vector machine Accuracy: %.3f%%\n',svm_Accuracy )
%# neural network Accuracy
net_train_input=training';
net_test_input=testing';
net=feedforwardnet(10,'trainscg');
[net,tr]=train(net,net_train_input,nn_train_class);
%plotperf(tr);
nn_Accuracy=0;
for i=1:2950
net_out=round(net(net_test_input(:,i)));
if net_out==nn_test_class(:,i);
nn_Accuracy=nn_Accuracy+1;
end
end
nn_Accuracy=(nn_Accuracy*100)/2950;
fprintf('Neural Network Accuracy: %.3f%%\n',nn_Accuracy )
```

# 4. ARAŞTIRMA BULGULARI ve TARTIŞMA

Görüntü işleme ve makine görme alanında birçok araştırma yapılmıştır. Ama görüntü madenciliği alanında yapılan araştırmalar başlangıç aşamasındadır ve ileride birçok araştırma alanları oluşabilir. Görüntü madenciliği alanında yapılabilecekler şunlardır:

- Görüntü madenciliğinin önemli uygulama alanlarından biri tıbbi teşhislerdir. Bu alanın hassasiyeti, insan sağlığı ve yaşam ile ilgili olduğu göz önüne alındığında çok yüksektir. Bu alanda aşağıdaki özelliklere sahip uygulamalar tasarlanıp uygulandığında yararlı olabilir:

• Çekilen fotoğrafların depolanması
• Tanım desenlerinin belirlenip depolanması
• Uzmanların yorumları ve desen tanımlamasından oluşan teşhislerden bir bilgi tabanı oluşturulması
• En sonunda uzmanın sadece çekilen görüntüleri programa girmesi yeterlidir. Görüntüler mevcut bilgi tabanı ile karşılaştırılır ve benzer sonuçlar görüntülenir. Bu işlem ile uzmanın teşhis doğruluğu artar.

- Kriminoloji, jeoloji araştırmaları, astronomi araştırmaları ve diğer benzeri uygulamalarda kullanılabilir.
- Görüntü madenciliği tekniklerini orman yangınları, sulak alanların yok olması, hayvan türlerinin yok olma tehlikesi ve diğer çevresel uygulamalarda kullanabiliriz.
- Görüntü madenciliği tekniklerini atmosferik koşullar ve hava kirliliği üzerine yapılan araştırmalarda kullanabiliriz.

# 5. SONUÇ

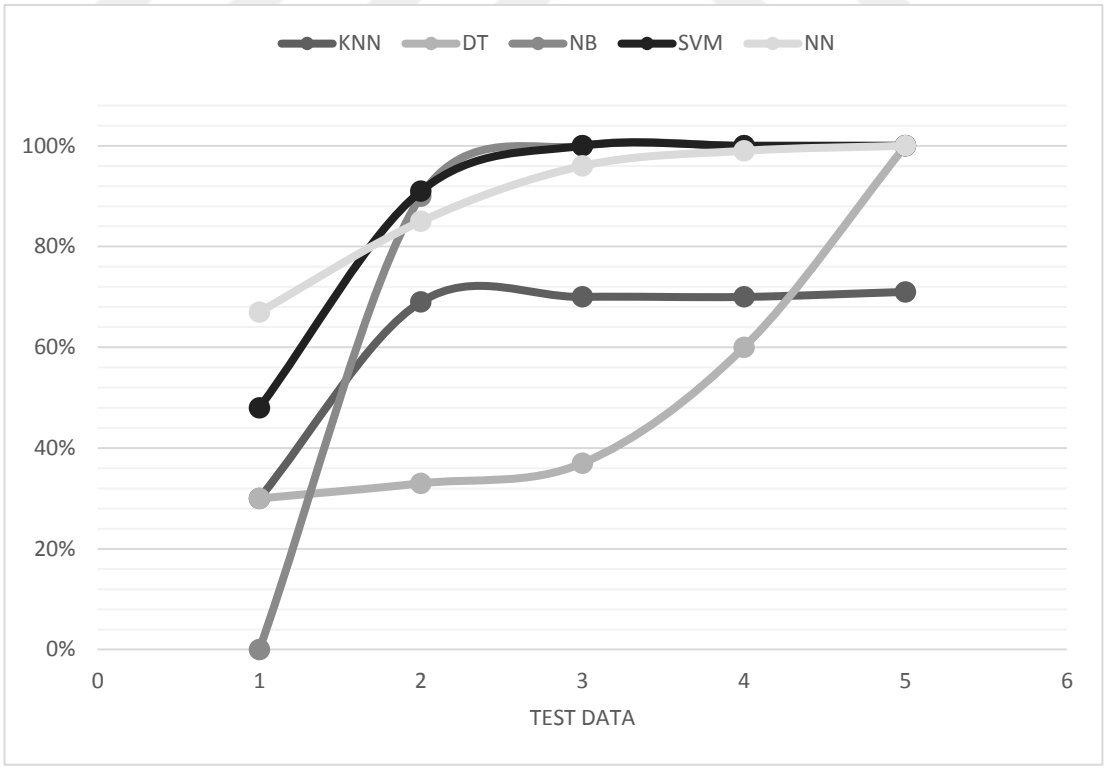
Görüntüleme ve görüntülerin depolanmasında yapılan gelişmeler, görüntü veri tabanı ve ayrıntılarının büyümesine neden olmuştur. Görüntü madenciliği, görüntülerden gizli bilgileri çıkarma, görüntü verilerinin arasındaki ilişkileri bulma ve resimde net olarak depolanmayan modelleri çıkarma konularında kullanılır. Görüntü madenciliği; özel yapay görme, görüntü işleme, görüntü alma, veri madenciliği, makine öğrenmesi, veri tabanları ve yapay zekâ üzerine disiplinler arası inşa edilmiş bir alandır. Görüntü madenciliği, bilinmeyen bilgileri ayıklar ve otomatik olarak etkili karar modelleri oluşturur. Yapay görme ve görüntü işleme belirli bir görüntü için belirli özellikleri ayıklamak üzerinde çalışır. Halbuki, görüntü madenciliği büyük bir resim yığınında desenleri ayıklamaya çalışır.

Uydu görüntüleri, tıbbi görüntüler ve dijital fotoğraf gibi birçok görüntüler günlük olarak yüksek hacimde üretilmektedir; bu görüntüler incelendiğinde bizim için çok yararlı olabilirler. Bir görüntünün içindeki nesneleri algılamak için piksellerin işlenmesi, görüntü madenciliğin en temel tartışmasıdır.

Test sonucunda ulaşılan sonuçların başarım bilgileri karışıklık matrisi ile ifade edilebilir. Karışıklık matrisinde satırlar test kümesindeki örneklere ait gerçek sayıları, kolonlar ise modelin tahminlemesini ifade eder. Model başarımının ölçülmesinde kullanılan en popüler ve basit yöntem, modele ait doğruluk oranıdır. Doğruluk oranı doğru sınıflandırılmış örnek sayısının, toplam örnek sayısına oranıdır. Hata oranı ise bu değerin 1'e tamlayanıdır. Diğer bir ifadeyle yanlış sınıflandırılmış örnek sayısının toplam örnek sayısına oranıdır. Aşağıdaki tabloda, test verilerinin sayısına göre her algoritmanın doğruluk oranı gösterilmektedir. Grafikde görüldüğü gibi bir eğitim verisi için en yüksek doğruluk oranı sinir ağlarına aittir ve eğitim verilerinin artırılmasıyla da artmaktadır.

Diğer algoritmalara göre bu projede uygulanan sinir ağlarının bir diğer avantajı, öğrenme yeteneğidir. Bu çalışmada eğitim verisinin büyüme kabiliyeti vardır.

| Test data | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| KNN | %30 | %69 | %70 | %70 | %71 |
| DT | %30 | %33 | %37 | %60 | %100 |
| NB | %0 | %90 | %100 | %100 | %100 |
| SVM | %48 | %91 | %100 | %100 | %100 |
| NN | %67 | %85 | %96 | %99 | %100 |



**KAYNAKLAR**


Agrawal R, Imielinski T, Swami AN 1993 "Mining association rules between sets of items in large databases". ACM SIGMOD international conference on management of data, pp 207–216

Berlage, T., 2005. "Analyzing and mining image databases", Drug Discovery Today: Basilica, DDT, Volume 10, Number 11.

Bhatt, C. A., Kankanhalli, M. S., 2010. "Multimedia data mining: state of the art and challenges", Multimedia Tools and Applications Volume 51, Issue 1, pp 35–76.

Brown R., Pham B., 2005. "Image mining and retrieval using hierarchical support vector machines", Multimedia Modelling Conference Proceedings of the 11th International.

Changjin, Y., hongxia, X., 2009. "The Invention of Image Mining Framework" Proceedings of IEEE Conference on Aerospace, vol. 3, pp. 253.

Cruz-Roa, A., C.Caicedo, J., A.Gonzalez, F., 2011. "Visual pattern mining in histology image collections using bag of features", Artificial Intelligence in Medicine Volume 52, Issue 2, Pages 91–106.

Fan, J., Yuligo, H., Hacid, Said, M., 2005. "Mining image on semantics via statistical learning", ACM sigkdd international conference on knowledge discovery in data mining pages 22-31.

Hema A., Annasaro E., 2013. "A survey in need of image mining techniques", International journal of advanced research in computer and communication engineering vol. 2

Kannan, A., Mohan, V., Anbazhagan, N., 2010. "Image clustering and retrieval using image mining techniques" IEEE International Conference on Computational Intelligence and Computing Research.

Lee, M. L., Zhang, J., Hsu, w., 2002 "Image Mining Trends and Developments", Journal of Intelligent Information Systems, manufactured in the Netherlands.

Missaoui R., Plenichka R., 2005 "effective image and video mining: an overview of model based approaches" Association for Computing Machinery (ACM)

Mohanty, A. K., Senapati, M. R., Lenka, S. K., 2013. "A Novel Image Mining Technique for Classification of Mammograms Using Hybrid Feature Selection" Neural Computing and Applications Volume 22, Issue 6, pp 1151–1161.

Perner, P., 2002. "Image mining: issues, framework, a generic toll and its application to medical image diagnosis", Engineering Applications of Artificial Intelligence Volume 15, Issue 2, Pages 205–216

Quartulli, A. K. M., Olaizola, I. G., 2013. "A review of EO image information mining", ISPRS Journal of Photogrammetry and Remote Sensing Volume 75, Pages 11–28

Ramadass S., 2011 "A Survey on Image Mining Techniques: Theory and Applications" Computer Engineering and Intelligent Systems. Vol 2 no.6

Sahu, M., Shrivastava, M., Rizvi, M. A., 2012. "Image Mining: a New Approach For Data Mining Base On Texture", Computer and Communication Technology (ICCCT) Third International Conference on - IEEE.

Zhang, J., and Hsu, W., and Lee, M. L. 2001 "Image Mining: Issues, Frameworks And Techniques", Department of Computer Science, School of Computing National University of Singapore

# ÖZGEÇMİŞ

1982 yılında İran'da doğdu. İlk, orta ve lise tahsilini Urmiye'de tamamladı. 2006 yılında Khoy Üniversitesi Bilgisayar Mühendisliği bölümünden mezun oldu. 2010 yılında Atatürk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans öğrenimine başladı. Aralık 2016 tarihinde eğitimini başarıyla bitirdi.