**T.C.**

**TRAKYA ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**

# SÜRÜCÜ UYKULULUĞUNUN GERÇEK ZAMANLI GÖRÜNTÜ İŞLEME VE MAKİNE ÖĞRENMESİ TEKNİKLERİ İLE TESPİTİNE YÖNELİK BİR SİSTEM TASARIMI VE UYGULAMASI

**MUHAMMED OZAN AKI**

**DOKTORA TEZİ**

**BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**

**Tez Danışmanı: Yrd. Doç. Dr. İlhan UMUT**

**EDİRNE-2017**

T.Ü. Fen Bilimleri Enstitüsü onayı

Prof. Dr. Murat YURTCAN
Fen Bilimleri Enstitüsü Müdürü

Bu tezin Doktora tezi olarak gerekli şartları sağladığını onaylarım.

Doç. Dr. M. Tolga SAKALLI
Anabilim Dalı Başkanı

Bu tez tarafımca okunmuş, kapsamı ve niteliği açısından bir Doktora tezi olarak kabul edilmiştir.

Yrd. Doç. Dr. İlhan UMUT
Tez Danışmanı

Bu tez, tarafımızca okunmuş, kapsam ve niteliği açısından Bilgisayar Mühendisliği Anabilim Dalında bir Doktora tezi olarak oy birliği ile kabul edilmiştir.

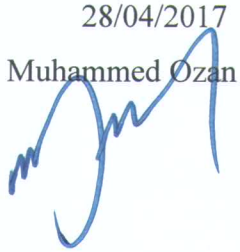
| Jüri Üyeleri | İmza |
|---|---|
| Doç. Dr. Erdem UÇAR | |
| Doç. Dr. Rafet AKDENİZ | |
| Yrd. Doç. Dr. İlhan UMUT | |
| Yrd. Doç. Dr. Cenk MISIRLI | |
| Yrd. Doç. Dr. Edip Serdar GÜNER | |

Tarih: **28 / 04 / 2017**

**T.Ü. FEN BİLİMLERİ ENSTİTÜSÜ**
**BİLGİSAYAR MÜHENDİSLİĞİ DOKTORA PROGRAMI**
**DOĞRULUK BEYANI**

İlgili tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin kaynak gösterilerek ilgili tezde yer aldığını beyan ederim.

28/04/2017
Muhammed Ozan AKI



Doktora Tezi

Sürücü Uykululuğunun Gerçek Zamanlı Görüntü İşleme ve Makine Öğrenmesi Teknikleri ile Tespitine Yönelik Bir Sistem Tasarımı ve Uygulaması

Trakya Üniversitesi Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

# ÖZET

Bu tezin amacı, sürücü uykululuğunun görüntü işleme ve makine öğrenmesi teknikleri ile gerçek zamanlı olarak tespit edilerek sürücünün uyarılmasını sağlayan bir sistemin tasarlanması ve gerçekleştirilmesidir.

Çalışmada veri tabanı oluşturmak için dört gönüllünün gerçek zamanlı kamera görüntüsü verileri kullanılmıştır. Kamera görüntüsü 320x240 uzaysal çözünürlüğe düşürülerek gri ölçeğe dönüştürülmüştür. Sürücünün yüz ve göz görüntüleri Viola-Jones detektörü yöntemi ile tespit edilerek kırpılmıştır. Görüntüdeki ışık değişimlerini normalleştirmek için histogram eşitleme yöntemi kullanılmıştır. Görüntülerin özniteliklerini çıkarmak için sekiz farklı açıda ve beş farklı ölçekte 40 Gabor filtresi kullanılmıştır. Sınıflandırma performansını arttırmak için korelasyon değerleri kullanılarak öznitelik seçimi yapılmıştır. Bu işlemler sonucunda dört gönüllüden elde edilen 1077 resim için 13 özellikten oluşan veri setleri oluşturulmuştur. Oluşturulan veri setleri ile beş farklı makine öğrenme algoritmasının sınıflandırma performansları birbiriyle kıyaslanmıştır. Bu kıyaslama sonucunda sınıflandırma doğruluğu ve sınıflandırma hızı değerlendirilerek J48 karar ağacı algoritması gerçek zamanlı uygulama için önerilmiştir.

Sürücü uykululuğunun tespiti için kişiye özel verilerle çalışan gerçek zamanlı bir yazılım geliştirilmiştir. Eğitim ve sınıflandırma olmak üzere iki modülden oluşan bu yazılımın performansını arttırabilmek için paralel programlama teknikleri kullanılmıştır.

Eğitim modülünde sürücü, açık ve kapalı göz görüntüleri ile veri dosyası oluşturmaktadır. Sınıflandırma modülünde ise, seçilen veri dosyası ile bir öğrenme

modeli oluşturularak canlı video görüntülerinde gözlerin açık ya da kapalı olduğu sınıflandırılmaktadır.

Gerçek zamanlı çalışmada sürücü uykululuğunun tespiti için, birim zaman periyodundaki kapalı göz sayısının, periyot zamanına oranı olan PERCLOS (Percentage of Closure: kapalılık oranı) ve gözlerin kapalı kalma süresi olan CLOSDUR (Closed duration: kapalılık süresi) büyüklükleri hesaplanmıştır. Bir dakikalık zaman periyodu içerisinde, PERCLOS değeri 0,15'ten büyükse birinci seviye, 0,3'ten büyükse ikinci seviye uyarı ile sürücü uyarılmaktadır. Ayrıca gözlerin ardışık olarak iki saniye boyunca kapalı kaldığı (CLOSDUR ≥ 2) tespit edildiğinde, PERCLOS değerine bakılmaksızın ikinci seviye alarm ile sürücü uyarılmaktadır.

Sonuç olarak bu çalışmada, sürücü uykululuğunun tespiti ve sürücünün uyarılması amacıyla görüntü işleme teknikleri ve makine öğrenme algoritmaları kullanılarak geliştirilen yazılımda, görüntü gerçek zamanlı olarak 24 FPS (Frame Per Second: saniyedeki resim sayısı) hızda işlenerek %90 sınıflandırma başarımı elde edilmiştir. Uykululuk düzeyi tanımlanan eşik değerlerine ulaştığında sürücü başarıyla uyarılmıştır.


| | |
|---|---|
| Yıl | : 2017 |
| Sayfa Sayısı | : 94 |
| Anahtar Kelimeler | : sürücü uykululuğu, görüntü işleme, makine öğrenmesi |

Doctoral Thesis
Designing and Implementation of Driver Sleepiness Detection System by Using Real Time Image Processing and Machine Learning Techniques
Trakya University Institute of Natural Sciences
Computer Engineering

# ABSTRACT

The purpose of this thesis is to design and implement a system that enables driver sleepiness to be detected in real time by image processing and machine learning techniques and to warn the driver.

In this study, four volunteers' real time camera images are used for creating data sets. The camera images are reduced to 320x240 spatial resolution and converted to gray scale. The face and eye images of driver are detected using Viola-Jones detector and then cropped. The histogram equalization is used for normalizing light changes in the images. Forty Gabor filters were used at eight different angles and five different scales to extract the features of the images. In order to improve the classification performance, the feature selection was made using correlation values. As a result of these operations, data sets consisting of 13 features were created for 1077 images obtained from four volunteers. These data sets are used for comparing classification performances of five machine learning algorithm. J48 decision tree algorithm is proposed for real time application by evaluating classification accuracy and classification time.

A real time application has been developed that allows working with personal data for detecting driver sleepiness. Parallel programming techniques have been used to improve the performance of this software, which consists of two modules, training and classification.

In the training module, driver creates own data file by its open and closed eye images. In the classification module, a learning model is created with the selected data file to classify whether the eyes are open or closed in live video images.

In order to detect driver sleepiness in real time operation, the number of closed eyes per time period is calculated as PERCLOS (Percentage of Closure) and CLOSDUR (Closed Duration), which is the period of closed eyes. Within one minute time period if the PERCLOS value is greater than 0.15, the driver is warned by first level warning. If the PERCLOS value is greater than 0.3, the driver is warned by the second level warning. Also, when the eyes are closed for two seconds (CLOSDUR ≥ 2) consecutively, the driver is warned by the second level alarm regardless of the PERCLOS value.

As result of this study, 24 FPS (Frame Per Second) real time image processing speed and 90% classification accuracy has been obtained by developed application using image processing and machine learning techniques for detecting driver sleepiness and then warning driver. Driver was successfully warned when sleepiness level is reached the defined threshold values.

Year : 2017
Number of Pages : 94
Keywords : driver sleepiness, image processing, machine learning

# TEŞEKKÜR

Tez çalışmamda gerek yazım gerek uygulama tasarımında yardımlarını esirgemeyen danışmanım Yrd. Doç. Dr. İlhan UMUT'a (Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü), her zaman değerli fikirlerinden faydalandığım hocam Doç. Dr. Erdem UÇAR'a (Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü), uykululuk ve bilimsel makale yazım teknikleri konusunda bilgilerinden faydalandığım Prof. Dr. Levent ÖZTÜRK'e (Trakya Üniversitesi Tıp Fakültesi, Fizyoloji ABD), gerçek zamanlı görüntü işleme sisteminin tasarımı konularındaki yardımlarından dolayı Prof. Dr. Hasan Hüseyin BALIK'a (Yıldız Teknik Üniversitesi, Elektrik Elektronik Fakültesi, Bilgisayar Mühendisliği Bölümü), makine öğrenme algoritmaları konusunda yardımlarından dolayı Prof. Dr. Yılmaz KILIÇASLAN'a (Adnan Menderes Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü), sürüş benzetim ortamında araç kullanarak veri toplanmasında yardımcı olan eşim Mehtap AKI'ya, Öğr. Gör. Dr. Aydın GÜLLÜ'ye, Öğr. Gör. Yusuf AVŞAR'a (Trakya Üniversitesi, İpsala Meslek Yüksekokulu) bu tez çalışmasına bulundukları katkı ve emeklerinden dolayı teşekkür ederim. Tez çalışması boyunca gösterdikleri sabır ve anlayıştan dolayı eşim Mehtap AKI ve kızımız Çimen AKI'ya ayrıca teşekkür ederim.

# İÇİNDEKİLER

**ÖZET** ..... **i**

**ABSTRACT** ..... **iii**

**TEŞEKKÜR** ..... **v**

**İÇİNDEKİLER** ..... **vi**

**SİMGELER ve KISALTMALAR LİSTESİ** ..... **ix**

**ŞEKİLLER LİSTESİ** ..... **xi**

**TABLOLAR LİSTESİ** ..... **xiii**

**BÖLÜM 1 GİRİŞ** ..... **1**

1.1. Konu ..... 1

1.2. Kapsam ..... 2

1.3. Literatür Araştırması ..... 3

**BÖLÜM 2 ARKA PLAN BİLGİSİ** ..... **8**

2.1. Sürücülerde Uykululuk ve Uykululuğun Algılanması ..... 8

2.2. Görüntünün Elde Edilmesi ve Görüntü İşleme ..... 10

2.2.1. Görüntünün Elde Edilmesi ..... 10

2.2.2. Görüntünün İşlenmesi ..... 11

2.2.2.1. Görüntü Ön İşlem Algoritmaları ..... 11

2.2.2.2. Gabor Dalgacık Dönüşümleri ..... 12

2.2.2.3. Yüz Görüntüsünün Bulunması ve Viola Jones Detektörü ..... 14

2.3. Veri Madenciliği ve Makine Öğrenmesi ..... 15

2.3.1. Sınıflandırıcılar ..... 17

2.3.1.1. Bayes Sınıflandırıcılar ..... 17

2.3.1.2. K En Yakın Komşu Algoritması ..... 18
2.3.1.3. Yapay Sinir Ağları ..... 19
2.3.1.4. Karar Ağaçları ..... 20
2.3.1.5. Destek Vektör Makinaları ..... 22
2.3.2. Sınıflandırıcıların Değerlendirme Ölçütleri ..... 23
**BÖLÜM 3 UYGULAMA ..... 27**
3.1. Sürücü Uykululuk Tespit ve Uyarı Sisteminin Genel Tasarımı ..... 27
3.2. Görüntülerin Elde Edilmesinde Kullanılan Sürüş Benzetim Ortamı ..... 28
3.3. Sürücü Görüntüsünün Elde Edilmesi ve Göz Görüntülerinin Kırpılması ..... 31
3.4. Görüntülerin Sınıflandırılması ve Görüntü Veri Tabanı ..... 34
3.5. Görüntü Özelliklerinin Çıkarılması ..... 35
3.6. Makine Öğrenme Algoritmalarının Sınanması ..... 39
3.6.1. Veri Setlerinin Hazırlanması ..... 40
3.6.2. Özniteliklerin İncelenmesi ve Öznitelik Seçimi ..... 40
3.6.3. Makine Öğrenme Algoritmalarının Sınanması ve Kıyaslanması ..... 43
3.6.4. Sınıflandırıcıların Öğrenme Seviyelerinin Değerlendirmesi ..... 48
3.7. Gerçek Zamanlı Çalışan Uygulama ..... 50
3.7.1. Canlı Görüntülerden Veri Dosyası Oluşturma Modülü ..... 50
3.7.2. Veri Dosyasından Model Oluşturma Modülü ..... 51
3.7.3. Canlı Video Görüntüsü Sınıflandırma ve Uykululuk İzleme Modülü ..... 52
3.7.4. Gerçek Zamanlı Çalışan Uygulamanın Kullanıcı Ara Yüzü ..... 56
3.7.5. Sürücü Uykululuğunun Algılanması ve Sürücünün Uyarılması ..... 59
**BÖLÜM 4 SONUÇLAR ve TARTIŞMA ..... 61**
4.1. Sonuçlar ..... 61
4.2. Tartışma ..... 66
4.3. Gelecek Çalışmalar ve Öneriler ..... 67

**EKLER** ..... **68**

5.1. EK-A Özniteliklerin Veri Setlerine Göre Korelasyon Değerleri ..... 68

5.2. EK-B Sınıflandırıcıların Veri Setlerine Göre Başarım Değerleri ..... 69

5.3. EK-C Uygulama Ana Çalışma Döngüsü Akış Diyagramı ..... 71

5.4. EK-D Uygulamada Veri Dosyası Oluşturma Akış Diyagramı ..... 72

5.5. EK-E Görüntü İşleme Döngüsü Akış Diyagramı ..... 73

5.6. EK-F Göz Görüntülerinin Bulunması ve Kırpılması Akış Diyagramı ..... 74

5.7. EK-G Göz Görüntülerini Sınıflandıran Paralel Görevlerin Akış Diyagramı . 75

5.8. EK-H Gözlerin Görüntü İçerisinde Bulunmasını Sağlayan Kaynak Kod ..... 76

5.9. EK-I Görüntü Çerçevelerini İşleyen Ana Fonksiyon Kaynak Kodu ..... 77

**KAYNAKLAR** ..... **80**

**ÖZGEÇMİŞ** ..... **93**

**TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER** ..... **94**

# SİMGELER VE KISALTMALAR LİSTESİ

**AAA** : American Automobile Association

**ANN / YSA** : Artificial Neural Network / Yapay Sinir Ağları

**ARFF** : Attribute Relation File Format (Öznitelik ilişki dosya formatı)

**BLINK** : Göz kırpma

**BLINKDUR** : Blink Duration (Gözün kapalı kalma süresi)

**BLINKFREQ** : Blink Frequency (Göz kırpma sıklığı)

**CLOSDUR** : Closure Duration (Gözün Kapalı Kalma Süresi)

**CLOSURE** : Gözün açık / kapalı olma durumu

**CLOSVEL** : Closure Velocity (Göz kapağının kapanma hızı)

**DA (TP)** : Doğru Artı (True Positive)

**DE (TN)** : Doğru Eksi (True Negative)

**DVM** : Destek Vektör Makineleri

**ECG** : Electrocardiography (Elektrokardiyografi)

**ECU** : Electronic Control Unit (Elektronik Kontrol Birimi)

**EEG** : Electroencephalography (Elektroansefalografi)

**EMG** : Electromyography (Elektromiyografi)

**EOG** : Electrooculography (Elektrokülografi)

**ESP** : Electronic Stability Program (Elektronik Denge Programı)

**ESS** : Epworth Sleepiness Scale (Epworth Uykululuk Ölçeği)

**FFT** : Fast Fourier Transform (Hızlı Fourier Dönüşümü)

**FPGA** : Field Programmable Gate Array

**FSM** : Finite State Machine (Sonlu Durum Makinesi)

**GMM** : Gaussian Mixture Model

**IPM** : Inverse Perspective Matching

**IR** : Infrared (Kızıl Ötesi)

**JDS** : John's Drowsiness Scale (John'un Uyuşukluk Ölçeği)

**KDS** : Karolinska Drowsiness Score (Karolinska Uyuşukluk Ölçeği)

**KNN** **:** K Nearest Neighbors (K En Yakın Komşular)

**KSS** **:** Karolinska Sleepiness Scale (Karolinska Uykululuk Ölçeği)

**LBP** **:** Local Binary Pattern (Yerel İkili Örüntü)

**LDA** **:** Linear Discriminant Analysis

**NHTSA** **:** National Highway Traffic Safety Administration

**OSS** **:** Objective Sleepiness Scale (Objektif Uykululuk Ölçeği)

**PCA** **:** Principles Component Analyses (Temel Bileşen Analizi)

**PERCLOS** **:** Percent of Closure (Gözün Kapalılık Oranı)

**ROC (AİE)** **:** Receiver Operating Curve (Alıcı İşlem Eğrisi)

**SOFM** **:** Self-Organizing Feature Map

**SPA** **:** Salient Point Analyses

**SSS** **:** Stanford Sleepiness Scale (Stanford Uykululuk Ölçeği)

**SVM** **:** Support Vector Machines

**TOKAT** **:** Ulusal Toplu Katalog

**TÜİK** **:** Türkiye İstatistik Kurumu

**UV** **:** Ultra Violet (Mor Ötesi)

**YA (FP)** **:** Yanlış Artı (False Positive)

**YE (FN)** **:** Yanlış Eksi (False Negative)

**YÖK** **:** Yükseköğretim Kurumu

**YSA** **:** Yapay Sinir Ağları

# ŞEKİLLER LİSTESİ

Şekil 2.1 Tek boyutlu Gabor dalgacıkları a) kosinüs (çift) b) sinüs (tek) ..... 13
Şekil 2.2 Haar-benzeri özellikler (Haar-like features) ..... 14
Şekil 2.3 Veri, enformasyon, bilgi ve akıl arasındaki hiyerarşik ilişki ..... 16
Şekil 2.4 Biyolojik sinir hücresinde bulunan gövde, dendrit, akson yapıları ..... 19
Şekil 2.5 Beş giriş ve bir çıkış katmanına sahip örnek bir yapay sinir ağı modeli ..... 20
Şekil 2.6 Hava durumuna bağlı olarak karar veren örnek bir karar ağacı ..... 21
Şekil 2.7 Destek vektör makinesinde sınıf örnekleri arasındaki en geniş aralık ve aralığın orta noktasındaki $H_0$ sınıflandırma fonksiyonu ..... 22
Şekil 2.8 Alıcı işlem karakteristik eğrisi. A eğrisi ideal sınıflandırmayı, B eğrisi olağan sınıflandırmayı, C doğrusu rasgele sınıflandırmayı göstermektedir. ... 26
Şekil 3.1 Tasarlanan sistemin basitleştirilmiş blok diyagramı ve işlem akış şeması ..... 28
Şekil 3.2 City Car Driving isimli sürüş benzetim uygulamasının örnek ekran görüntüsü ..... 29
Şekil 3.3 Sürüş benzetim ortamı ve sürücü görüntüsünün elde edilmesi ..... 30
Şekil 3.4 Sürüş benzetim ortamına yerleştirilen kameradan elde edilen sürücü görüntülerinden iki örnek ..... 31
Şekil 3.5 İki örnek Haar-benzeri özelliğin uygulanması ..... 32
Şekil 3.6 Görüntünün Viola-Jones detektörü ile tespit edilerek kırpılması ..... 32
Şekil 3.7 Göz bölgesinin belirlenmesinde kullanılan düşey yüz oranları ..... 33
Şekil 3.8 Sol ve sağ göz bölgelerinin düşey ve yatay olarak ayrılması ..... 33
Şekil 3.9 Sırasıyla görüntü kırpma işlemleri a) kamera görüntüsü b) kırpılan yüz görüntüsü c) yüz görüntüsünden kırpılan göz görüntüsü ..... 33
Şekil 3.10 Sürücü görüntülerinin elde edilmesini sağlayan uygulama arabirimi ..... 34
Şekil 3.11 Beş farklı ölçek ve sekiz farklı açıda Gabor dalgacıkları ..... 36
Şekil 3.12 Görüntü üzerinde gerçekleştirilen konvolüsyon işlemi ve örnek çıktıları ..... 36
Şekil 3.13 Örnek görüntüye uygulanan sekiz farklı açı ve beş farklı ölçekteki Gabor filtresinin a) gerçek ve b) büyüklük değerlerini veren görüntü çıktıları ..... 37

Şekil 3.14 Örnek bir ARFF veri dosyasının yapısını gösteren ilk 20 satırı ..... 38
Şekil 3.15 Görüntü özelliklerinin otomatik çıkarımını sağlayan uygulama ara yüzü ..... 39
Şekil 3.16 Her bir veri tabanındaki özniteliklerin korelasyon değerleri ..... 41
Şekil 3.17 Öznitelik korelasyonlarının tüm veri setleri için ortalama değerleri ..... 42
Şekil 3.18 Öznitelik seçimi öncesi ve sonrasında doğruluk değerlerindeki değişimler .. 43
Şekil 3.19 Algoritmaların doğruluk değerleri ve saniye cinsinden sınıflandırma süreleri ..... 46
Şekil 3.20 Göz kapağının kapanma ve açılma fazları ..... 47
Şekil 3.21 IBk algoritmasının veri setleri için öğrenme eğrileri ..... 49
Şekil 3.22 J48 algoritmasının veri setleri için öğrenme eğrileri ..... 49
Şekil 3.23 Veri dosyası ve model oluşturma akış diyagramı ..... 51
Şekil 3.24 Otomatik model seçim işleminin akış diyagramı ..... 51
Şekil 3.25 Göz durumlarının kaydedildiği sabit boyutlu göz durum dizisi ..... 53
Şekil 3.26 Gerçek zamanlı çalışan uygulamanın ekran görüntüsü ..... 56
Şekil 3.27 Canlı video görüntülerini sınıflandıran modülün arayüz ekranı öğeleri ..... 57
Şekil 3.28 Canlı video görüntüsünden veri dosyası oluşturan modülün arayüz öğeleri . 57
Şekil 3.29 Özniteliklerin seçilmesini sağlayan pencerenin ekran görüntüsü ..... 58
Şekil 3.30 Veri dosyasından öğrenme modeli oluşturma modülünün arayüz öğeleri ..... 58
Şekil 3.31 İki seviyeli sürücü uyarı sisteminin akış diyagramı ..... 60
Şekil 3.32 Gerçek zamanlı uygulamada sürücünün 1. seviyede uyarılması ..... 60
Şekil 4.1 Gerçek zamanlı video testinde artan PERCLOS değerine karşılık sürücüye verilen uyarı mesajları. Yatay eksen çerçeve sayısını, düşey eksen PERCLOS değerini göstermektedir. ..... 65

# TABLOLAR LİSTESİ

Tablo 1.1 Sürücü uykululuğunu tespit eden çalışmalarda kullanılan yöntemlerin ve ölçülen özelliklerin sınıflandırılması ..... 6
Tablo 2.1 Sınıflandırma işlemi sonucunda elde edilen karmaşıklık matrisi ..... 24
Tablo 3.1 Dört farklı kişiden elde edilen görüntülerle oluşturulan görüntü veri setleri . 35
Tablo 3.2 Veri dosyasında kullanılan öznitelik isimlerinin kodlama tablosu ..... 39
Tablo 3.3 Sınamaya tabi tutulan sınıflandırıcılar ve parametrik varyasyonları ..... 44
Tablo 3.4 Algoritmaların veri setlerine göre doğruluk değerleri ..... 45
Tablo 3.5 Algoritmaların veri setlerine göre ROC eğrisi altında kalan alanları ..... 45
Tablo 3.6 Algoritmaların veri tabanlarına göre saniye olarak sınıflandırma süreleri ..... 46
Tablo 3.7 Gerçek zamanlı uygulamada kullanılan ön tanımlı makine öğrenme algoritmaları ve varsayılan parametreleri ..... 52
Tablo 3.8 Hata ayıklama ve normal çalışmada elde edilen FPS değerleri ..... 55
Tablo 3.9 Farklı öznitelik sayıları ile sınıflandırma yapıldığında görüntü işleme hızları ..... 55
Tablo 3.10 Bu çalışmada kullanılan PERCLOS eşik değerleri ve bu değerlere karşılık gelen uykululuk durumları ile uyarı seviyeleri ..... 59
Tablo 4.1 Gerçek zamanlı çalışma hızının benzer çalışmalarla kıyaslanması ..... 63
Tablo 4.2 Gerçek zamanlı test sınıflandırmasının karmaşıklık matrisi ..... 64
Tablo 4.3 Gerçek zamanlı test sınıflandırmasının başarım ölçütleri ..... 64

# BÖLÜM 1

# GİRİŞ

## 1.1. Konu

Amerikan Otomobil Birliği (AAA) Ulaşım Güvenliği Kurumu'nun yayınlamış olduğu rapora göre, sürücülerin %41'i direksiyon başında uyuklamış ya da uykululuktan dolayı başı öne düşmüştür. 40 yaş üstü sürücülere kıyasla 16 - 24 yaş aralığındaki genç sürücülerin uykululuktan dolayı karıştıkları kaza sayısının iki kat fazla olduğu saptanmıştır. Ayrıca, ölüm ve yaralanmayla sonuçlanan kazaların %16,5'inin yorgun ve uykulu sürücülerden kaynaklandığı belirtilmektedir [1].

Türkiye İstatistik Kurumu raporlarına göre 2002 ile 2015 yılları arasında meydana gelen trafik kazalarında sürücüden kaynaklanan kusurların oranı %92,87 olarak hesaplanmıştır. Sadece 2015 yılı için 210.498 trafik kazasından 187.980 kazanın (%89,3) sürücü kusurlarından kaynaklandığı rapor edilmiştir [2]. Ancak bu oranın ne kadarının uykululuktan kaynaklandığına dair bir çalışmaya rastlanmamıştır.

Edirne ilinde ticari araç sürücüleri için yapılmış çalışmanın sonucuna göre 138 sürücüden 22'si (%15,9) uykululuğa bağlı en az bir trafik kazası geçirmiş ya da kaza tehlikesi atlatmıştır [3]. Edirne ve Hatay illerini kapsayan benzer bir çalışmanın sonuçlarına göre ise, 320 sürücüden 49'u (%15,3) uykululuk nedeniyle en az bir kaza geçirmiş ya da kaza tehlikesi atlatmıştır [4]. Ticari sürücüler üzerinde yapılan bir anket çalışması sonucunda kaza geçiren sürücülerin %17'sinde kazanın uykululuktan kaynaklandığı sonucuna varılmıştır [5].

Uykulu sürücülerin, sürücü kusurlarından kaynaklanan kazalarda önemli bir paya sahip olduğu görülmektedir [1, 3-5]. Bu nedenle, sürücülerin uykulu araç kullanmamaları yönünde yasal düzenlemeler, denetimler ve bilinçlendirme çalışmaları yapılmaktadır.

Uykululuktan kaynaklı kazaların önüne geçebilmek amacıyla, T.C. Karayolları Trafik Yönetmeliği Madde 98, a bendinde, "ticari amaçla yük taşımacılığı yapan ve azami ağırlığı 3,5 tonu geçen araçların şoförleri ile ticari amaçla yolcu taşımacılığı yapan ve taşıma kapasitesi şoförü dâhil 9 kişiyi geçen araçların şoförlerinin 24 saatlik herhangi bir süre içinde; toplam olarak 9 saatten ve devamlı olarak 4,5 saatten fazla araç sürmeleri yasaktır" ifadesi bulunmaktadır [6].

Bununla birlikte, araçlarda sürücülerin uykulu olma durumlarını algılayan ve sürücüyü uyaran bir sistemin, trafik kazalarında uykululuktan dolayı meydana gelebilecek kazaları önemli ölçüde azaltabileceği öngörülmektedir [7-10].

Bu çalışmada, sürücülerin uykululuk durumunu algılayan ve uykululuğun tespit edilmesi halinde bir uyarı ile sürücüyü uyararak mola vermesini ve dinlenmesini tavsiye eden bir sistemin gerçek zamanlı görüntü işleme teknikleri ve makine öğrenme algoritmaları kullanılarak tasarlanması ve gerçekleştirilmesi amaçlanmıştır.

### 1.2. Kapsam

Giriş bölümünde, bu tez çalışmasının konusu, amacı ve önemi hakkında bilgiler verildikten sonra, literatür taraması kapsamında tezin konusu ile ilgili gerçekleştirilmiş daha önceki çalışmalar incelenmiştir. Literatür taraması sonucunda elde edilen yayınlarda kullanılan ya da önerilen yöntemler sınıflandırılarak istatistiksel veriler çıkartılmıştır. Elde edilen sınıflandırmalara göre, yöntemlerin başarımları, maliyetleri, avantajları ve dezavantajları irdelenmiştir.

Arka Plan Bilgisi bölümünde, sistem tasarımında gerekli olan teorik bilgilere yer verilmiştir. Uykululuk ve yorgunluk, bunların fizyolojik ve davranışsal belirtileri ve nasıl tespit edileceği ile ilgili konular araştırılmıştır. Görüntü işleme teknikleri ve algoritmaları hakkında genel bilgiler verilerek, bu tez çalışması kapsamında kullanılan makine öğrenme algoritmaları ve bu algoritmaların değerlendirme ölçütleri incelenmiştir.

Uygulama bölümünde, sürücü uyarı sisteminin tasarım aşamaları detaylı bir şekilde ele alınmıştır. Sürücü görüntüsünün elde edilmesi, sürücü yüz ve göz görüntüsünün tespit edilerek kırpılması, bu görüntülere ait özniteliklerin çıkarılması işlemleri görüntü işleme kapsamında ele alınan konulardır. Göz görüntülerinin özniteliklerinin elde edilmesiyle veri setleri elde edilmiş ve bu veri setleri makine öğrenme algoritmalarının sınanmasında kullanılmıştır. Makine öğrenme algoritmalarının sınıflandırma sonuçları kıyaslanarak bu

uygulama için en uygun makine öğrenme algoritması gerçek zamanlı uygulama için önerilmiştir.

Son bölüm olan Sonuçlar ve Tartışma bölümünde, sürücü uykululuğunu tespit edip sürücüyü uyarmak amacıyla geliştirilen gerçek zamanlı uygulama incelenmiştir. Uygulamanın öğrenme ve sınıflandırma modüllerinin çalışması detaylı bir şekilde açıklanmıştır. Örnek video görüntüsü üzerinden elde edilen başarım ölçütleri değerlendirilmiş ve çalışma hızı benzer çalışmalar ile kıyaslanmıştır. Gerçek zamanlı uygulamadan elde edilen sonuçların karmaşıklık matrisi çıkarılarak sınıflandırma doğruluğu test edilmiştir. Tartışma bölümünde ise, tasarım ve kullanılan yöntemlerin kısıtlamalarından bahsedilmiş ve başarım kıyaslamaları değerlendirilmiştir.

### 1.3. Literatür Araştırması

Literatür taramasında elde edilen yayınlarda kullanılan ya da önerilen yöntemler incelendiğinde, ölçümlerde sürücüye müdahale edilmesi bakımından yöntemlerin iki sınıfa ayrılabileceği görülmüştür. Bunlar kısaca müdahaleli ve müdahalesiz yöntemler olarak isimlendirilmiştir.

Sürücü açısından elektrot takılması, sensör bağlanması, veri girişi istenmesi gibi sürücüye müdahale gerektiren yöntemleri kullanan 30 adet (%13,3) çalışma ve sürücüye müdahale gerektirmeyen, sürücüye herhangi fiziksel bağlantısı olmayan ve sürücüyü rahatsız etmeyen yöntemleri kullanan 195 adet (%86,7) çalışma tespit edilmiştir [11-14].

Sürücüden elde edilen verilerin kaynağı incelendiğinde, fizyolojik ve davranışsal olmak üzere iki tür veri kaynağı kullanıldığı gözlenmiştir. Fizyolojik ölçüm yapan 34 adet (%15,0) çalışma ve davranışsal ölçüm yapan 192 adet (%85,0) çalışma bulunmuştur.

Fizyolojik ölçümler, uykululuğa bağlı olarak sürücünün fizyolojik özelliklerinde meydana gelen değişiklikleri ölçümlemeye dayanır. Buna göre, elektroansefalografi (EEG: electroencephalography), elektrokülografi (EOG: electrooculography), elektrokardiyografi (ECG: electrocardiography), elektromiyografi (EMG: electromyography) sinyallerinin uykululuğun bir göstergesi olarak kullanılabileceği belirtilmiştir [15-24]. Fizyolojik ölçümlerde, 23 adet (%46,0) EEG, 10 adet (%20,0) EOG, altı adet (%12,0) ECG, altı adet (%12,0) kalp, 4 adet (%8,0) EMG ve bir adet (%2,0) göğüs sinyallerini kullanan çalışmalar vardır [25-36] . Bu ölçümlerin dezavantajı, elektrotların sürücüye bağlanma zorunluluğudur [16, 21, 22].

Davranışsal ölçümler, uykuluğa bağlı olarak sürücünün davranış ve tepkilerinde meydana gelen değişimleri tespit etmeye dayalı yöntemleri kapsar [12]. Davranışsal ölçümlerin büyük çoğunluğu görüntü işleme tekniklerini kullanmaktadır. Sürücünün görsel analizinde, 148 adet (%67,9) göz, 36 adet (%16,5) ağız, 34 adet (%15,6) kafanın pozisyonu ile ilgili ölçümler yapılmıştır. Sadece göz ile ilgili ölçüm yapan 49 adet (%29,2) çalışmada göz kapağının kapalılık oranı (PERCLOS: Percentage of Closure) büyüklüğünün ölçüldüğü görülmüştür [37-48]. Toplam 44 (%26,2) çalışmada, gözün açık kapalı durumu (CLOSURE) ölçülmüştür [49-51]. Toplam 40 (%23,8) çalışmada göz kırpma (BLINK), 18 adet (%10,7) çalışmada gözün kapalı kalma süresi (BLINKDUR), 11 adet (%6,5) çalışmada birim zamanda göz kırpma sıklığı (BLINKFREQ: Blink Frequency), altı adet (%3,6) çalışmada göz kapağının kapanma hızı (CLOSVEL: Closure Velocity) parametreleri ölçülmüştür [44, 48, 52-59]. Ağız analizi yapan toplam dokuz çalışmanın yedisinde (%77,8) esneme, ikisinde (%22,2) konuşma durumu ölçülmüştür [60-65]. Kafa ile ilgili ölçüm yapan 31 adet çalışmada, başın uykululuktan dolayı düşmesi ve bakış doğrultusunun sabitlenmesi ölçülmüştür [66-70].

Fizyolojik ölçümlerin hemen hemen tamamı sürücüye müdahale gerektirirken, davranışsal ölçümlerin neredeyse tamamının sürücüye müdahale gerektirmediği gözlenmiştir. İstisnai olarak, direksiyon simidine yerleştirilen basınç ya da nabız sensörleri ile sürücüye müdahale etmeden ölçülebilen fizyolojik büyüklükler olduğu gibi, sürücü başının konumunu ya da gözlerin durumunu algılamak için sürücüye sensör ya da özel gözlük takılması gerektirmesi nedeniyle sürücüye müdahale edilerek yapılan davranışsal ölçümlere de rastlanmıştır [71-73].

Araç üstü sensörler, sürücü davranışlarında meydana gelen değişimlerin araç kullanımı üzerindeki etkisini ölçer. Dolayısıyla sürücü davranışının bir sonucu ölçüldüğünden araç üstü sensörlerden yapılan ölçümler müdahalesiz ve davranışsal ölçüm yöntemi olarak sınıflandırılabilir. Bu tür ölçümler için genellikle hâlihazırdaki araç üstü sensörler kullanılmaktadır [11, 74]. Ancak bazı çalışmalarda sonradan monte edilen sensörler de kullanılmıştır [71-73]. Araç üstü sensörlerle ölçüm yapan 13 adet çalışmada araç hızı, direksiyon açısı, gaz ve fren pedal pozisyonları verileri ölçülmüştür [75-77].

Çevresel ölçümler, sürücü kontrolündeki aracın global ya da yol çizgilerine göre göreceli konumunu ölçer ve bu ölçümler dolaylı olarak sürücü davranışlarına bağlıdır. Bu nedenle, çevresel ölçümler müdahalesiz ve davranışsal ölçüm olarak sınıflandırılabilir.

Bu tür ölçümlerde GPS, şerit çizgileri, yol kenar çizgileri, öndeki ve/veya arkadaki araç ile mesafe gibi araç dışı ölçümler yapılır [78, 79].

Giyilebilir teknolojileri kullanan yöntemler, sürücünün giymek ya da takmak zorunda olduğu bir aksesuar nedeniyle müdahaleli, ancak bir yelek ya da kol saati gibi günlük olarak giyilebilen ve sürüş sırasında herhangi rahatsız edici etkisi olmadığından sürüş sırasında müdahalesiz sayılabilir. Giyilebilir teknolojileri kullanan 6 çalışmaya rastlanmıştır [80-85].

Hibrit yaklaşımlarda ise, açıklanan bu kategorilerdeki en az iki ya da daha çok yöntem aynı anda kullanılmaktadır [67, 68, 86-89].

Literatür taramasında elde edilen yayınlarda kullanılan ya da önerilen yöntemlerin sınıflandırması Tablo 1.1'de özet bir şekilde gösterilmiştir.

Sürücünün uykululuk derecesini ifade etmek amacıyla beş çalışmada Epworth Sleepiness Score (ESS) [4, 5, 90-92], 10 çalışmada Karolinska Sleepiness Scale (KSS) [16, 22, 23, 78, 93-98], iki çalışmada Karolinska Drowsiness Score (KDS) [18, 94], bir çalışmada Objective Sleepiness Scale (OSS) [99] ölçekleri kullanılmıştır. Stanford Sleepiness Scale (SSS) ve John's Drowsiness Scale (JDS) ölçeklerini kullanan çalışmaya rastlanmamıştır. Bu ölçeklerden herhangi birini kullanmayan diğer çalışmalarda ise sürücünün uykululuk durumunu yazarlar kendileri sınıflandırmıştır.

Diğer yandan sürücü uykululuğuna bağlı motorlu taşıt kazalarının azaltılması amacıyla araç üreticilerinin de çeşitli yöntemler geliştirdikleri ve araç üretiminde kullanmaya başladıkları gözlenmiştir [100-103].

Kadın ve erkeklerden oluşan 550 kişilik bir grup sürücü üzerinde yapılan bir denemede, direksiyonu düzeltme hamlelerinin, yorgunluğun ilk belirtilerini işaret ettiği tespit edilmiştir. Buna göre, Attention Assist (dikkat yardımcısı) adı verilen bu sistem, ilk bir kaç dakika boyunca, hızlanma, pedal kullanımı, araç hızı, direksiyon tepkisi (bu yorgunluğu gösteren en belirgin veridir) gibi verileri değerlendirerek bir sürücü profili oluşturmakta ve sürüşün kalanı boyunca elde edilen verileri bu oluşturulan profil ile karşılaştırmaktadır. Sistem, sürüş profilinden sapma tespit ettiğinde, sürücüyü görsel ve akustik olarak uyarmakta ve mola vermesini tavsiye etmektedir. Bu sistem, 2009 yılında Mercedes Benz firması tarafından araçlarında kullanılmaya başlanmıştır [100].

Tablo 1.1 Sürücü uykululuğunu tespit eden çalışmalarda kullanılan yöntemlerin ve ölçülen özelliklerin sınıflandırılması

| Veri Kaynağı | Ölçülen Özellik / Görsel Özellik | | Yayın Sayısı |
|---|---|---|---|
| **Fizyolojik Ölçümler** | | | |
| Sürücü | Sadece EEG | | 12 |
| | EEG, EOG | | 2 |
| | EEG, EOG, EMG | | 2 |
| | EEG, EOG, ECG | | 1 |
| | Kalp | | 2 |
| **Davranışsal Ölçümler** | | | |
| Sürücü | Sadece Göz | PERCLOS | 23 |
| | | BLINK, BLINKDUR, BLINKFREQ | 10 |
| | | CLOSURE, CLOSVEL | 26 |
| | | PERCLOS, BLINK, BLINKDUR, BLINKFREQ, CLOSURE, CLOSVEL | 7 |
| | Sadece Ağız | Esneme, Konuşma | 9 |
| | Sadece Kafa | Kafanın duruşu, Bakış | 9 |
| | Göz ve Kafa | PERCLOS, BLINK, BLINKFREQ, BLINKDUR, Kafanın duruşu, Kafanın öne düşmesi, Bakış | 13 |
| | Göz ve Ağız | PERCLOS, BLINK, BLINKDUR, CLOSURE, Esneme | 15 |
| | Göz, Kafa Ağız | BLINK, BLINKDUR, CLOSURE, Kafanın duruşu, Esneme | 9 |
| Araç üstü ölçümler | Araç hızı, Direksiyon-dönme açısı, Gaz ve Fren pedalı pozisyonları | | 13 |
| Çevresel ölçümler | GPS Araç global pozisyonlama sistemi | | 1 |
| **Hibrit Ölçümler** | | | |
| Sürücü | Göz (PERCLOS, BLINK), Kalp, EEG, EOG, ECG, Nefes | | 5 |
| Sürücü ve araç üstü sensörler | Göz (PERCLOS, BLINKDUR), Kalp, Direksiyon açısı, Direksiyon tutma basıncı, Gaz pedalı pozisyonu | | 15 |
| Araç üstü sensörler ve çevresel ölçümler | Direksiyon açısı, Araç hızı, Gaz pedalı pozisyonu, Şerit çizgileri | | 8 |
| Sürücü, araç üstü sensörler ve çevresel ölçümler | Göz (PERCLOS, BLINKDUR, BLINKFREQ), Esneme, Kafanın öne düşmesi, Direksiyon açısı, gaz pedalı pozisyonu, Fren pedalı pozisyonu, Şerit çizgileri, GPS | | 5 |

İlk ticari uygulamaların hemen sonrasında bu sistem, farklı ticari firmalar tarafından da uygulanmaya başlanmıştır. Diğer bir çalışmada, direksiyon hareketleri için ESP sisteminin zaten bir parçası olan direksiyon açı sensöründen alınan veriler değerlendirilmekte ve benzer şekilde, sürüşün ilk dakikalarında bir sürücü profili oluşturularak sonraki sürüş boyunca bu profilden sapmalar takip edilmektedir. Bu sistem otomotiv sektörününün önemli tedarikçilerinden biri olan Bosch (Almanya) firması tarafından geliştirilmiştir. Bu sistemde sürücünün, sürüş boyunca direksiyonda yaptığı düzeltme hamlelerinin sıklığı, sinyallerin kullanımı, günün saati gibi parametreler kullanılmaktadır. [101, 102]

Başka bir çalışmada ise, yine sürücünün direksiyon düzeltmeleri esas alınmaktadır. Ancak bunu araç üzeri sensörler yerine, dikiz aynasının arkasına yerleştirilen ve araç önündeki yolu görebilen bir kamera ile gerçekleştirir (Ford, ABD). Kamera, yol çizgilerini baz alarak aracın şeridinde gidip gitmediğini kontrol eder. Eğer sık sık şerit ihlalinin ardından düzeltme yapılıyorsa, bu, sürücünün sürüş dikkatinin bozulduğu anlamına gelmektedir. Böylece sürücü mola vermesi için görsel ve işitsel olarak uyarılır. Ön cama yerleştirilen kamera, şerit takip uyarı sistemi ve şerit takip yardımcı sisteminin bir parçasıdır [103].

# BÖLÜM 2

# ARKA PLAN BİLGİSİ

## 2.1. Sürücülerde Uykululuk ve Uykululuğun Algılanması

Araç kullanma işlemi, hızlı ve doğru algılamaya bağlı psikomotor işlevleri içeren ve dikkat gerektiren karmaşık bir faaliyettir [5]. Sürüş sırasında dikkat, hızlı ve doğru algılama ile muhakeme gibi yetileri içeren psikomotor fonksiyonların uykululuk nedeniyle zayıflaması, araçların kaza riskini arttıran başlıca unsurdur. Uykululuk, uyku ihtiyacının tam olarak karşılanmamasının kaçınılmaz bir sonucu olarak, uykuya dalma eğilimi ile karakterize edilir [4, 5].

Literatür taramasından elde edilen yöntemlerin sınıflandırılması sonucunda, uykululuk düzeyinin ya da uykululuk durumunun tespit edilmesinde fizyolojik ölçümler, davranışsal ölçümler ve anket yöntemlerinin kullanıldığı tespit edilmiştir.

Fizyolojik ölçümlerde EEG, EOG, ECG, EMG ve nabız sinyalleri kullanılmıştır. EEG sinyallerinin, uykululuk düzeyinin bir göstergesi olduğu belirtilmiş ve artan uykululuk düzeyine karşın EEG sinyalinin delta bandının hızla düştüğü belirtilmiştir [104]. Ayrıca görsel aktivite azaldığında EEG sinyalinin alfa dalgalarının genliğinin yükseldiği, beta dalgalarının ise, dikkat ve konsantrasyonla bağlantılı olduğu belirtilmektedir [88].

EOG sinyalleri, göz hareketlerinin ölçülmesinde kullanılan sinyallerdir. EOG sinyalleri, görüntü işleme tekniklerine alternatif olarak kullanılmıştır ve uykululuğun tespit edilmesinde kullanılabileceği belirtilmiştir [29].

ECG sinyalleri, temel olarak nabız ölçmek amacıyla kullanılmıştır. Nabızdaki değişkenliğin, otonom sinir sistemi aktivitelerini ölçmenin bir yolu olduğu belirtilmektedir [31, 32].

EMG sinyalleri, sinir ve kas aktivitelerinin ölçülmesiyle elde edilir. EMG sinyallerinin de yüksek bir doğrulukla uykululuğu sınıflandırmada kullanılabileceği belirtilmiştir [33].

Literatür taramasından elde edilen yöntemlerin sınıflandırılması sonucunda davranışsal ölçümlerin büyük çoğunluğunun görüntü işleme tekniklerine dayandığı ortaya çıkmıştır. Sürücü odaklı görüntü işleme yöntemlerinin hemen hemen tamamı, sürücünün kafa ve yüz görüntüsü üzerinde işlem yapmaktadır.

Görüntüyle algılanan kafanın duruş pozisyonu, uyuklama sırasında başın öne düşmesini algılamak amacıyla kullanılmıştır [105].

Sürücünün ağzına yönelik yapılan ölçümlerin önemli bir bölümü esnemeyi algılamak amacını taşır. Bunun yanında ağız analizi, sürücünün konuşmakta olup olmadığının tespiti için de kullanılmıştır.

En çok ve yaygın kullanılan yüz öğesi gözlerdir. Sürücünün göz görüntülerinden birçok parametre elde edilmiştir. Bunlardan en çok kullanılan parametreler, gözlerin kapalılık oranı (PERCLOS), gözlerin kapalılık durumu (CLOSURE), göz kapağının kapanma hızı (CLOSVEL), göz kırpma (BLINK), göz kırma sıklığı (BLINKFREQ), olarak sayılabilir [38, 56, 57, 59, 106, 107]. Bunların yanında, yine gözlerden elde edilen ve üç boyutlu analiz sonucunda bakış doğrultusu (GAZE) ve bakışın sabitlenmesi parametrelerinin de ölçüldüğü gözlenmiştir [68, 108].

Görüntü işleme teknikleri kullanan fakat sürücü görüntüsünü işlemeyen çalışmalarda ise, görüntü işleme tekniklerinin aracın sürüş seyrinin takibi için şerit ve yol çizgilerinin algılanmasında kullanıldığı görülmüştür [109-113]. Sürücü tepkisinin dolaylı bir çıktısı olarak aracın şeridinde düzgün bir şekilde gidip gitmediği izlenmiştir. Şerit ihlallerinin ve ardından yapılan düzeltme hareketlerinin sıklığının, sürücünün uykululuk durumunu işaret ettiği belirtilmiştir [78, 103].

Görüntü işleme tekniklerine dayanmayan ancak dolaylı olarak sürücü tepkilerini ölçen çalışmalarda, araç üstü sensörlerin kullanıldığı görülmektedir [11, 71-77]. Sürücü tepkilerinin bir sonucu olarak sürücünün araç kullanma tarzının öğrenilmesi esasına dayanan yöntemleri kapsamaktadır. Sürüşün ilk dakikalarında sürücünün araç kullanma tarzının öğrenilmesi ve ilerleyen saatlerde, araç kullanma süresi, günün saati gibi ek bilgilerle birlikte daha önce öğrenilen sürüş profilinden sapmaların sürücü uykululuğunu belirlemede kullanılabileceği tespit edilmiştir [100-103].

## 2.2. Görüntünün Elde Edilmesi ve Görüntü İşleme

### 2.2.1. Görüntünün Elde Edilmesi

Görüntü, görme duyusuna sahip biyolojik canlılarda cisimlerden yansıyan ışığın gözün retina tabakasına düşmesiyle meydana gelen algıdır [114]. Görme duyusu değerlidir, zira anlık bir zaman dilimi içinde çevremizle ilgili çok sayıda bilgi ediniriz. Aynı avantajın makinelere uyarlanması fikrinden bilgisayar görüşü ya da diğer bir ifadeyle makine görüşü fikri doğmuştur [115].

Görüntü işleme fikri, sayısal olarak elde edilen görüntü verisinden karar süreçlerinde kullanılabilecek bilgilerin elde edilmesini hedefler. Görüntünün elde edildiği ortam iki kategoriye ayrılabilir;

- Doğal (şartlandırılmamış) ortam
- Şartlandırılmış ortam

Doğal ortam, görüntünün alınacağı ortama hiçbir müdahalemizin olmadığı ya da olamadığı ortamdır. Nesnelerin ya da canlıların doğal ortamı içerisinde müdahalede bulunmadan elde edilen görüntüleri işlenir. Şartlandırılmış ortamlar, görüntü işleme işini kolaylaştırmak amacıyla, yapay aydınlatma, nesnenin konumlandırılması ya da sabitlenmesi gibi müdahaleler ile sistematik olarak düzenlenmiş ortamlardır.

Sayısal görüntüler, iki boyutlu matris formunda elde edilir. Her bir matris elemanı, bir görüntü hücresini temsil eder. Görüntü hücrelerine kısaca piksel adı verilmektedir. Her bir piksel, ait olduğu konumun ışık yoğunluğu bilgisini verir. Görüntünün eni ve boyu, matrisin sütun ve satır sayısı ile ilişkilidir. Birim alandaki satır ve sütun sayıları, uzaysal çözünürlüğü belirler. Uzaysal çözünürlük, görüntü içerisinden elde edilebilecek yüksek frekanslı içeriğin sınırlarını belirler. Genlik çözünürlüğü, görüntüdeki her bir pikselin alabileceği değer sayısını belirler. En düşük *siyah* ya da *beyaz* olmak üzere iki değere sahip olabilen bir piksel değerini, bu iki uç nokta arasında kaç farklı değerde temsil edileceği genlik çözünürlüğü ile belirlenir. Ara değerler, siyahtan beyaza doğru grinin tonları ile temsil edilir. Bu tür görüntülere *gri ölçek* görüntüleri denir. Sayısallaştırma işlemi bir sayısal devre ile gerçekleştirildiğinden, çözünürlük her zaman ikinin üstel katı kadar olur [116].

Renk bilgisinin görüntülerde temsil edilmesi, kırmızı, yeşil ve mavi temel renklere ait verilerin ayrı matrislerde tutulmasıyla gerçekleştirilir. Renk bilgisine gereksinim

duymayan uygulamalar için, tek matristen oluşan ve her bir görüntü hücresinde sadece ışık bilgisinin tutulduğu gri ölçek görüntüleri kullanılır. Gri ölçek görüntüler hesaplamayı basitleştirdiğinden birçok uygulamada tercih edilir [117].

### 2.2.2. Görüntünün İşlenmesi

Görüntü verisinden anlamlı bilgi elde edebilmek için çeşitli algoritmalar geliştirilmiştir. Bu algoritmalar genel olarak, görüntünün işlenmeye hazırlanması için uygulanan ön işlem algoritmaları ve görüntü üzerindeki öznitelikleri belirlemekte kullanılan algoritmalar olarak sınıflandırılabilir.

#### 2.2.2.1. Görüntü Ön İşlem Algoritmaları

Doğadan elde edilen tüm elektriksel sinyallerde olduğu gibi, kameralardan elde edilen görüntülerde de mutlaka bir miktar gürültü bulunmaktadır. Bu nedenle elde edilen görüntülere çeşitli filtreler uygulanarak gürültüler azaltılır. Gürültü azaltmak için kullanılan başlıca yöntemler *Ortalama, Gauss*, *Sınır* ve *Orta Değer* filtreleri olarak sayılabilir.

Ortalama filtresi, her bir piksel için, basitçe komşu piksellerin ışık yoğunluklarının ortalamasını alarak, piksel için yeni bir değer hesaplar [118].

Gauss filtresi, komşu pikseller arasındaki dağılımı, Gauss eğrisine yakınlaştırmayı amaçlar. Bunun için, her bir pikselin değerini, komşu piksellerin ortalamasını alarak belirler [117]. Tipik bir Gauss görüntü maskesi Denklem 2.1'de gösterilmiştir.

$$\frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.1}$$

Denklem 2.2'de gösterilen diğer bir Gauss görüntü maskesi ise, Gauss eğrisine daha yakın bir profile sahiptir.

$$\frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.2}$$

Bu maskeler 3x3 komşuluk sınırları içerisinde işlem yapmaktadır ve bu maskelerin görüntüye uygulanması, komşu piksellerin ağırlık değerlerinden dolayı toplam görüntü yoğunluğunda bir değişme meydana getirmez. Ancak bu filtreler kenar görüntülerinde bulanıklaşmaya neden olur. Bulanıklaşma görüntü işlemede istenmeyen bir durumdur [117].

Sınır filtresi, Gauss filtresine benzer şekilde komşu piksellerin yoğunluklarına göre işlem yapar. Sınır filtresi, bir pikselin ışık yoğunluğunu, komşu piksellerin minimum ve maksimum yoğunluk değerleri ile sınırlandırır. Orta değer filtresi, küçükten büyüğe göre sıralanmış komşu piksel değerlerinden ortadaki değeri alarak yeni bir piksel değeri elde eder. Eğer komşu sayısı çift ise, ortada kalan iki yoğunluk değerinin ortalaması alınır. Orta değer filtresi, ani gürültüleri bastırmada oldukça iyidir. Nesne sınırlarında sebep olduğu bulanıklaşma ise, Gauss filtresine göre çok daha düşüktür [119].

Görüntü elde edilirken kameranın pozlama süresinin ve diyafram açıklığının uygun olmaması, piksel değerlerinin dengesiz dağılım göstermesine neden olur. Görüntü kaydedildikten sonra histogram eşitleme algoritmalarıyla görüntü içerisindeki piksel yoğunluklarının dağılımı, düzgün dağılım olacak şekilde yeniden düzenlenebilir [114, 118].

#### 2.2.2.2. Gabor Dalgacık Dönüşümleri

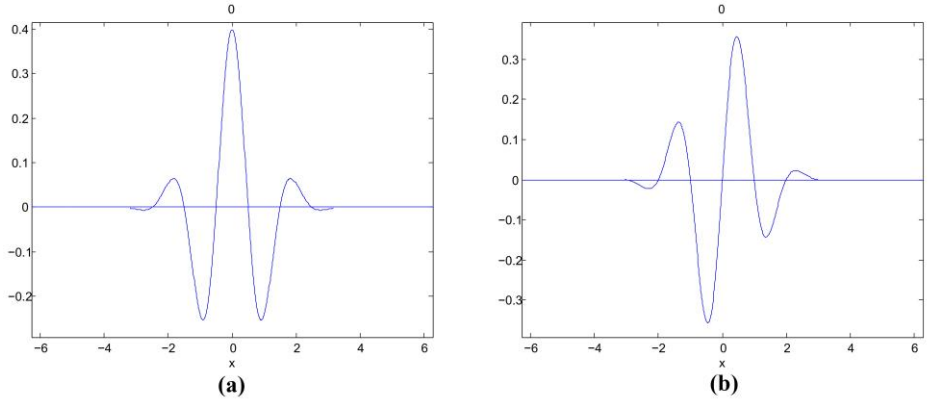
Gabor dalgacık (wavelet) dönüşümleri, ya da diğer adıyla Gabor filtreleri, ilk olarak Dennis Gabor tarafından ortaya konmuştur [120]. Gabor dalgacık dönüşümleri, bir sinyalden zaman ve frekans bilgilerini çıkarmak / elde etmek amacıyla kullanılmaktadır. Ayarlanabilir dalgacık boyutu sayesinde farklı çözünürlüklerde analiz yapabilmektedir. Görüntü işleme uygulamalarında sıkça kullanılan bir yöntemdir [121].

Tek boyutlu bir Gabor dalgacık dönüşümü için Gabor dalgacık elemanı, bir Gauss penceresi içerisindeki sinüs veya kosinüs fonksiyonudur. Özel olarak sinüs fonksiyonu için tek (odd) ve kosinüs fonksiyonu için çift (even) adı verilmektedir. Tek boyutlu Gabor dalgacıkları için çift ve tek fonksiyonları sırasıyla Denklem 2.3 ve Denklem 2.4'te gösterilmiştir [120].

$$g_e(x) = \frac{1}{\sqrt{2 \cdot \pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot \cos(2\pi\omega_0 x) \tag{2.3}$$

$$g_o(x) = \frac{1}{\sqrt{2 \cdot \pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot \sin(2\pi\omega_0 x) \tag{2.4}$$

Bu denklemlerde, $\sigma$ Gauss penceresinin genişliğini, $\omega_0$ ise merkez frekansı belirler.




Şekil 2.1 Tek boyutlu Gabor dalgacıkları a) kosinüs (çift) b) sinüs (tek)

Şekil 2.1'de, tek boyutlu çift ve tek Gabor dalgacıklarının fonksiyon grafikleri gösterilmiştir. Denklem 2.3 ve Denklem 2.4 birleştirilerek, Denklem 2.5'de gösterildiği şekilde Gabor fonksiyonu karmaşık formda yazılabilir.

$$g(x) = g_e(x) + i \cdot g_o(x) \tag{2.5}$$

Bu denklemde $g_e(x)$ ve $g_o(x)$ fonksiyonları yerine yazılırsa,

$$g(x) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot (\cos(2\pi\omega_0 x) + i \cdot \sin(2\pi\omega_0 x)) \tag{2.6}$$

Denklemi elde edilir. Bu denklemin üstel formu Denklem 2.7'de gösterilmiştir:

$$g(x) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot e^{i \cdot (2\pi\omega_o x)} \tag{2.7}$$

Böylece karmaşık formda Gabor denklemi elde edilir. Ancak görüntü işleme alanında, veriler iki boyutlu olduğundan, burada iki boyutlu (2B) Gabor dalgacıkları kullanılır. Gabor dalgacıkları, Daugman tarafından iki boyutlu olacak şekilde geliştirilmiştir [122]. İki boyutlu çift ve tek Gabor dalgacık fonksiyonları sırasıyla Denklem 2.8 ve Denklem 2.9'da görüldüğü şekilde tanımlanmıştır.

$$g_e(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \cdot e^{-\frac{1}{2}\cdot\left(\frac{x^2}{\sigma_x}+\frac{y^2}{\sigma_y}\right)} \cdot \cos(2\pi\omega_{x0}x + 2\pi\omega_{y0}y) \tag{2.8}$$

$$g_o(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \cdot e^{-\frac{1}{2}\cdot\left(\frac{x^2}{\sigma_x}+\frac{y^2}{\sigma_y}\right)} \cdot sin(2\pi\omega_{x0}x + 2\pi\omega_{y0}y) \tag{2.9}$$

Böylece, elde edilen Gabor dalgacık fonksiyonları ile istenen frekans ve açıda Gabor dalgacıkları üretilerek bir görüntüye ait öznitelikler elde edilebilmektedir.

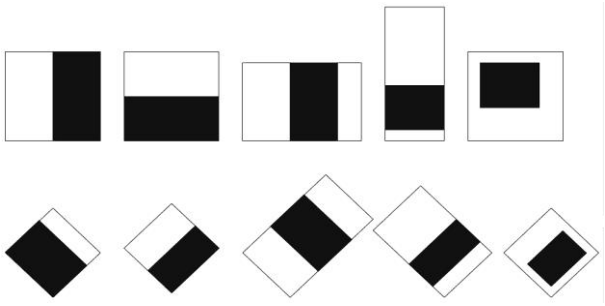
### 2.2.2.3. Yüz Görüntüsünün Bulunması ve Viola Jones Detektörü

Bir video ya da resim içeresindeki yüz görüntüsünü algılamak amacıyla kullanılan yöntemlerden biri de Paul Viola ve Michael Jones tarafından geliştirilen ve AdaBoost sınıflandırıcıyı temel alan yöntemdir [123, 124]. Viola-Jones detektörü ismi de verilen bu sınıflandırıcı, Boosting sınıflandırıcıların en çok kullanılan varyasyonudur. Boosting sınıflandırıcılar, karar ağaçlarını iteratif olarak işleterek karar ağaçlarının sağladığı avantajlardan faydalanırlar [117].

Bu sınıflandırıcılar, yüksek tanıma oranına ve düşük eleme oranına sahiptir. Karar düğümlerinin herhangi aşamasında sınıflandırma dışı sonucu alındığında hesaplama sonlanır ve bu bölgede tanınmak istenen nesnenin olmadığı sonucuna varılır. Gerçek sınıf, sadece hesaplama tüm aşamalarda doğru sonucu verdiğinde bulunur. Buradaki sınıf, nadir bulunan bir sınıf ise (örneğin resimdeki yüz görüntüsü), aşamaların elenmesi işlemi hesaplama maliyetini önemli ölçüde azaltmaktadır [117].

Bu çalışmada kullanılan yüz tanıma tekniği, Viola-Jones detektörü olarak tanınan ve Paul Viola ve Michael Jones tarafından geliştirilen tekniği esas alır. Bu teknik daha sonra Rainer Lienhart ve Jochen Maydt tarafından çapraz (diyagonal) özellikleri de kullanabilecek şekilde geliştirilmiştir [125]. Böylece bu teknik Haar sınıflandırıcı ve kullandığı özelliklerde Haar-benzeri özellikler (Haar-like features) olarak isimlendirilmiştir. Haar sınıflandırıcı, bir denetimli sınıflandırıcıdır (supervised classifier). Bu nedenle, tanınmak istenen nesnenin etiketlendiği görüntüler bu sınıflandırıcıya önceden öğretilmelidir [117]. Sınıflandırıcıda kullanılan örnek Haar benzeri özellikler Şekil 2.2'de gösterilmiştir.

AdaBoost sınıflandırıcıda genellikle birbirinden bağımsız tek seviyeli ikili ağaçlardan oluşan T adet zayıf sınıflandırıcı $h_t$, $t \in (1, \ldots, T)$ eğitilir. Nihai kararı verebilmek için her bir sınıflandırıcıya bir $\alpha_t$ ağırlık değeri verilir.



Şekil 2.2 Haar-benzeri özellikler (Haar-like features)

Giriş veri seti olarak her biri $y_i$ skalar etiketleri ile etiketlenmiş $x_i$ özellik vektörlerini kullanılır. Diğer algoritmalarda herhangi bir reel sayı olabilen etiketler AdaBoost sınıflandırıcı için $y_i \in (-1,+1)$ olacak şekilde ikilidir. İlk olarak, sınıflandırıcının bir veri noktasının yanlış sınıflandırılmasının maliyetini belirtmek için veri noktalarının ağırlık dağılımını gösteren $D_t(i)$ tanımlanır. Sınıflandırıcının anahtar özelliği olan bu maliyet, zayıf sınıflandırıcıların eğitiminde kullanılmaktadır. Algoritmanın kaba kodu şu şekilde yazılabilir;

---

- $D_1(i) = 1/m$, $i=1, \dots, m$
- Her bir $t=1, \dots, T$ için,
  - $D_i(i)$ ağırlıklı hatanın en düşük değerini sağlayan $h_t$ sınıflandırıcısını bul.
  - $h_t = \text{argmin}_{h \in H}$, $\varepsilon_j = \sum_{i=0}^{m} D_t(i)$ , $y_i \neq h_j(x_i)$ , $\varepsilon_j < 0,5$ olduğu sürece.
  - En küçük hata $\varepsilon_j$ için $h_t$ ağırlığı $\alpha_t = \frac{1}{2}\log\left[\frac{1-\varepsilon_t}{\varepsilon_t}\right]$
  - Veri noktaları ağırlığını güncelle $D_{t+1}(i) = \frac{[D_t(i)^{-\alpha_t y_i h_t(x_i)}]}{Z_t}$

---

Öğrenme algoritması tamamlandığında, güçlü olan son sınıflandırıcı yeni giriş vektörü x'i alır ve eğitilmiş zayıf sınıflandırıcılar üzerindeki ağırlıklı toplamı $h_t$'yi kullanarak Denklem 2.10'da gösterildiği şekilde sınıflandırma yapar.

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{2.10}$$

Buradaki *H(x)* fonksiyonu sınıflandırma fonksiyonu, *sign()*, pozitif olan bütün değerleri bir, negatif olan bütün değerleri ise -1 döndüren bir fonksiyondur. Bu fonksiyonda sıfırlar değişmeden kalmaktadır.

### 2.3. Veri Madenciliği ve Makine Öğrenmesi

Veri madenciliği, en basit ifadeyle veriden anlam çıkarılması işidir. Günümüzde verinin var olduğu her alanda analiz, geleceği öngörme, strateji geliştirme ve karar verme işlemlerinde eldeki mevcut verilerden yararlanılır. Ancak eldeki veriler her zaman istenen bilgiyi açıkça göstermeyebilir ve basit algoritmalar ya da grafiklerle veriler arasındaki

ilişkiler açıkça görünmeyebilir. Ayrıca bu veriler oldukça yüksek miktarlarda olabilirler ve bu verilerden çıkarsama yapmak oldukça zordur [126-128].

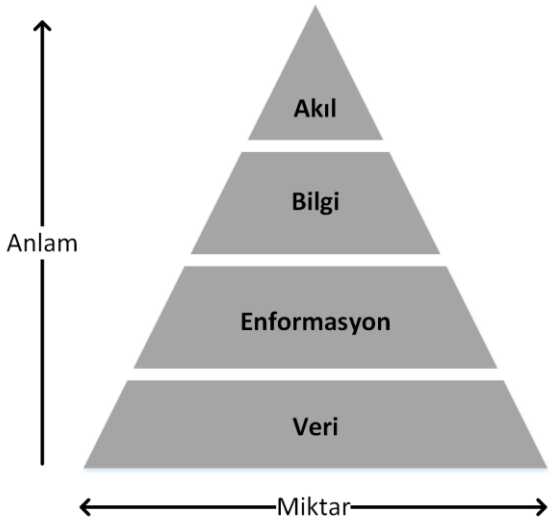
Veri madenciliği, büyük miktardaki veriler içerisinde açıkça görünmeyen ilişkileri ortaya çıkararak, bu verilerden anlamlı bilgileri elde eden istatistiksel ve matematiksel yöntemlerin genel adıdır [128].

Verilere uygulanan belirli algoritmaların kestirilebilir çıktılar üreteceği aşikârdır. Ancak bazı durumlarda, çıktıyı üretecek olan algoritma açıkça tanımlı olmayabilir. Sınırlı sayıdaki örnek girdi verisi için çıktılar önceden tanımlı ise, algoritma çıkarımı yapmak mümkün olabilmektedir. Girdi ve çıktılar arasındaki ilişkiyi tanımlayan algoritmaları bulmak bir öğrenme işidir. Öğrenme işi, bilgisayar adını verdiğimiz mantıksal makineler yoluyla yapıldığından "makine öğrenmesi" ya da "yapay öğrenme" adı verilmiştir [127]. Makine öğrenmesi yoluyla verilerin bilgiye ve akla dönüşümünü gösteren piramit Şekil 2.3'te gösterilmiştir. Piramidin üst kademelerine çıkıldıkça veri miktarı azalmakta ve anlamsal içerik artmaktadır.

Bu çalışmada, girdi verileri ile çıktılar arasında tanımlı bir algoritma bulunmadığından, bu ilişkinin tanımlanması ve daha sonra çıktısı bilinmeyen verilerde kullanılabilmesi amacıyla makine öğrenme algoritmaları kullanılmıştır.

Makine öğrenmesi temel olarak üç kategoride incelenir;

- Denetimli Öğrenme
- Denetimsiz Öğrenme
- Pekiştirmeli Öğrenme




Şekil 2.3 Veri, enformasyon, bilgi ve akıl arasındaki hiyerarşik ilişki

*Denetimli öğrenme,* girdi ve çıktıları belirli olan öğrenme süreçleridir. Bu süreçte, girdiler ve çıktılar arasındaki ilişkiyi tanımlayan bir fonksiyon ya da algoritma öğrenilir. Bu öğrenme türünde, örneklere ait doğru ve yanlış çıktılar önceden var olabilir ya da bir öğretici tarafından verilmiş olabilir. Diğer bir denetimli öğrenme şekli ise, girdi örneklere ait çıktı etkilerinin gözlemlenebilir bir çevre içerisinde doğrudan makine öğrenmesi tarafından algılanabilmesidir [129].

*Denetimsiz öğrenme*, örnek girdiler olmasına rağmen herhangi tanımlı doğru ya da yanlış çıktının olmadığı öğrenme şeklidir. Öğrenme algoritması, tam olarak ne öğrendiğini bilemez, ancak veri içerisindeki açıkça görünmeyen yapıları ortaya çıkarabilir [129].

*Pekiştirmeli öğrenmede,* öğrenme sürecinde sonuca ulaşmak amacıyla gerçekleştirilen eylemlerin ne kadar yararlı olduğunu belirten bir geri besleme sistemi vardır. Öğrenme sürecini sonuca götüren her bir eylem ya da eylemler dizisi için pozitif bir geri besleme alınır. Böylelikle hangi eylemlerin faydalı ya da hangilerinin faydasız olduğu etiketlenerek öğrenme gerçekleştirilir [127].

### 2.3.1. Sınıflandırıcılar

Denetimli bir öğrenme türü olan sınıflandırma işlemi, önceden etiketlenmiş olan örneklerle algoritma eğitilerek, etiketi bilinmeyen verilerin sınıflandırılmasında kullanılır. Sınıflandırıcı eğitiminde, örneklerin etiketleri ile verileri arasında bir örüntü öğrenilmeye çalışılır. Bu öğrenme işlemi sonucunda bir fonksiyon, algoritma ya da kurallar dizisi yoluyla bir model oluşturulur. Bu model daha sonra yeni gelen etiketsiz verilerin sınıflandırılması amacıyla kullanılır [128].

#### 2.3.1.1. Bayes Sınıflandırıcılar

Bayes sınıflandırıcılar istatistiksel teknikleri kullanır. Koşullu olasılık kuramına göre, X örneğinin C sınıfına ait olma olasılığı *P(C | X)* olarak gösterilir. Bu koşullu olasılık Denklem 2.11'de gösterildiği şekilde tanımlanır [127, 128].

$$P(C \mid X) = \frac{P(X \cap C)}{P(X)} \tag{2.11}$$

Bu denklemde $P(X \cap C)$ ifadesi yerine, Denklem 2.12'da gösterilen eşdeğeri yazılabilir.

$$P(X \cap C) = P(C|X) \cdot P(X) = P(X|C) \cdot P(C) \tag{2.12}$$

Bayes kuramını oluşturan denklemi, $P(X \cap C)$ ifadesini Denklem 2.11'de yerine yazarak elde edebiliriz.

$$P(C \mid X) = \frac{P(X|C) \cdot P(C)}{P(X)} \tag{2.13}$$

Naïve Bayes olarak isimlendirilen sınıflandırıcı, hesaplama yükünü azaltmak amacıyla her örneğe ait X değerinin birbirinden bağımsız olduğunu varsayarak Denklem 2.14'te gösterilen ifadeyi tanımlar.

$$P(X|C_i) = \prod_{k=1}^{n} P(X_k|C_i) \tag{2.14}$$

Her bir örnek için yapılan hesaplamada, en yüksek olasılık değerini veren örneklerin, hesaplanan sınıfa ait olduğu kabul edilir [128].

$$C_{max} = \max\left(\prod_{k=1}^{n} P(X_k|C_i)\right) \tag{2.15}$$

Böylelikle, Naïve Bayes sınıflandırıcı için nihai olasılık ifadesi Denklem 2.15'te gösterildiği şekilde yazılabilir [128].

#### 2.3.1.2. K En Yakın Komşu Algoritması

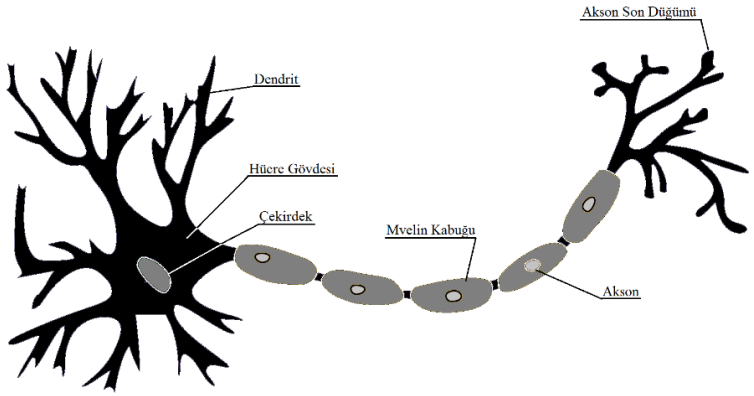
Bu algoritmada sınıflandırılmak istenen örnek, daha önce sınıflandırılmış olan eğitim verisindeki en yakın komşusu ya da komşularının sınıfına atanır. Uzaklıkları değerlendirilen komşu sayısı *k* ile gösterilir. Bu nedenle bu algoritma kısaca k-NN (k Nearest Neighbor) olarak isimlendirilir. Komşuluk mesafeleri Öklid formülü ile hesaplanır [126, 128]. Buna göre, sınıflandırılacak örnek *x* ile mesafesi hesaplanmak istenen komşu *n* arasındaki mesafenin *a* adet özellik için hesaplanması Denklem 2.16'da gösterilmiştir.

$$d(x,n) = \sqrt{\sum_{i=1}^{a}(n_i - x_i)^2} \tag{2.16}$$

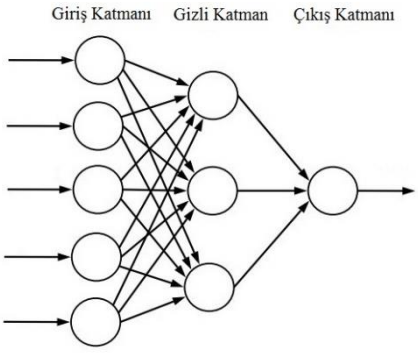
Parametrik olarak seçilen *k* komşu sayısı için mesafeler hesaplandıktan sonra, en yakın *k* komşunun çoğunluğu hangi sınıfa ait ise, *x* örneği bu komşuların sınıfına atanır. Eğer *k* değeri 1 olarak seçilmişse, *x* örneği sadece en yakın komşunun sınıfına atanır. Seçilen *k* değeri büyüdükçe örnek çevresindeki çember genişleyeceğinden, diğer sınıfa ait örneklerin de söz sahibi olması ve dolayısıyla hatalı sınıflandırmaya yatkın olacağı aşikârdır [126, 128]. Bu nedenle, sadece sayısal çokluğa bakmak yerine seçilen *k* komşu için her bir sınıfa ait komşuların mesafeleri de göz önüne alınarak ağırlıklı oylama yöntemi kullanılabilir [128].

### 2.3.1.3. Yapay Sinir Ağları

YSA (Yapay sinir ağları), biyolojik sinir ağlarının temel alındığı bir öğrenme algoritmasıdır. Biyolojik sinir sistemini oluşturan nöron, hücre gövdesi, dendrit ve aksondan meydana gelmektedir. Aksonlar, bilgi aktarımını sağlayan uzantılardır ve diğer nöron hücreleri ile sinaps olarak adlandırılan birleşme noktaları yoluyla haberleşir. Her bir sinir hücresi, çok sayıda sinir hücresi ile bağlantı halindedir. Aynı anda birçok sinir hücresinden uyarı alan bir nöron, bu uyarıların miktarı belirli bir eşik seviyesini geçtiğinde bu uyarıyı diğer nöronlara iletir [126]. Bir sinir hücresinde bulunan gövde, dendrit ve aksonlar Şekil 2.4'te gösterilmiştir.




Şekil 2.4 Biyolojik sinir hücresinde bulunan gövde, dendrit, akson yapıları




Şekil 2.5 Beş giriş ve bir çıkış katmanına sahip örnek bir yapay sinir ağı modeli

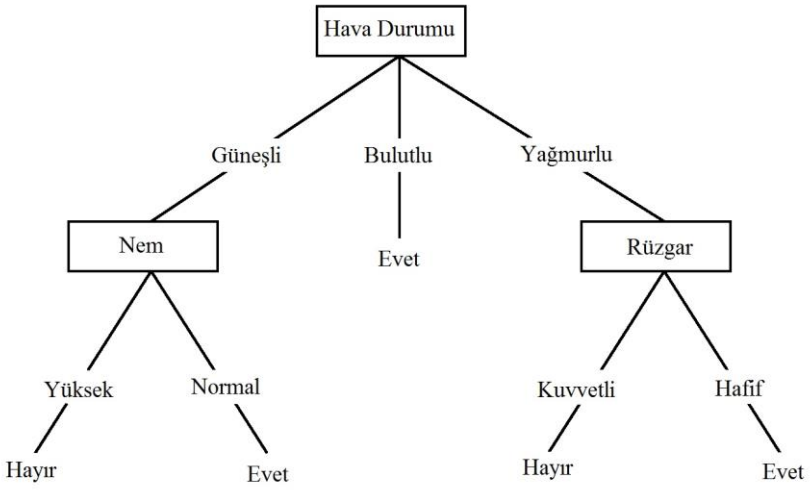
YSA biyolojik hücreye karşılık gelen ve yapay sinir hücresi (artificial neuron), düğüm (node), birim (unit), işlem elemanı (processing element) gibi isimler verilen yapılar ve bunların aralarındaki yoğun bağlantılardan meydana gelmektedir. YSA modeli, her birinde farklı sayılarda işlem elemanı bulunan katmanlardan meydana gelmektedir. Bu katmanlar, girdi, gizli ve çıkış katmanları olarak isimlendirilir. Geriye yayılım ağları ile giriş katmanına verilen bilgi ile çıkış katmanından alınan bilgi, beklenen bilgi ile kıyaslanarak bir hata değeri üretilir ve bu hata değeri gizli katmanda bulunan hesaplama elemanlarına iletilir. Bu şekilde YSA eğitilerek, çıktısı bilinmeyen veriler için çıktı değerleri üretebilir [126]. Örnek bir YSA modeli Şekil 2.5'te gösterilmiştir.

#### 2.3.1.4. Karar Ağaçları

Karar ağaçları, bir kökten başlayarak karar düğümlerinde ilerleyen ve nihayetinde etiketlenmiş yapraklarda sonlanan bir sınıflandırıcıdır ve gözetimli öğrenme kapsamındadır. Kökten başlayarak yapraklara ilerlerken, her karar geçişinde veri dizisinin yarısı elendiğinden sonuca çok hızlı ulaşılabilir. En iyi durumda $n$ sayıdaki düğüm için $\log_2 n$ kararla sonuca ulaşılır [127]. Hava durumuna bağlı olarak dışarıda basketbol oynamaya karar verebilen tipik bir karar ağacı şeması Şekil 2.6'da gösterilmiştir. Bu örnekte, her bir düğümde verilen kararlar ile bir sonraki dal ya da yaprağa ulaşılır. Yapraklara ulaşıldığında nihai karar etiketi elde edilir.

Karar ağaçları yorumlanabilir yapılar olduğundan ve kolayca eğer – ise kurallarına dönüştürülebildiğinden birçok uygulamada öncelikle tercih edilir [127].

Ağaç yapısını oluşturmak için Quinlan tarafından ID3 ve C4.5 algoritmaları geliştirilmiştir. Bu algoritmalar, bir sistemdeki belirsizliğin ölçüsü olan entropi kavramını kullanır.




Şekil 2.6 Hava durumuna bağlı olarak karar veren örnek bir karar ağacı

Entropi, bir *T* kümesindeki olasılıkların toplamı şeklinde Denklem 2.17'de gösterilen *H(T)* fonksiyonu olarak ifade edilir [128].

$$H(T) = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i \tag{2.17}$$

Burada *T*, sınıf özelliğidir ve *T* sınıf özelliği için entropi bulunur. Her bir özelliğin, sınıf bazında ağırlıklı ortalamaları ise Denklem 2.18'de gösterildiği şekilde hesaplanır.

$$H(X,T) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \cdot H(T_i) \tag{2.18}$$

Burada *T* sınıfına ait her bir *X* özelliği için entropi hesaplanır. *T* toplam değer sayısı, $T_i$ ise özelliğin aldığı değerlerin sayısıdır. $H(T_i)$ ise, *T* özelliğinin *i* değeri için hesaplanan entropi değeridir. Her özellik için bilgi kazancı Denklem 2.19'da gösterildiği şekilde hesaplanır.

$$Kazanç(X,T) = H(T) - H(X,T) \tag{2.19}$$

Bu şekilde, en yüksek kazancı sağlayan özellik, ağacın kök düğümü olarak ele alınır. Karar ağacının diğer dal ve düğümlerini oluşturmak için kalan değerler yeniden hesaplanarak her bir alt düğüm, en yüksek kazancı veren özellik olarak eklenir [128].

ID3 algoritması, sadece kategori bazında sınıflandırmalar için geçerliyken, aynı algoritmanın geliştirilmiş hali olan C4.5 algoritması, sayısal değerlere sahip özelliklerin

karar düğümlerinde kullanılmasına izin vermektedir. Sayısal özelliklerin kullanıldığı karar mekanizmalarında *t* eşik değeri kullanılmaktadır. Bölünen değerler için ayrı ayrı kazanç hesaplanarak karar ağacı oluşturulmaktadır [128].

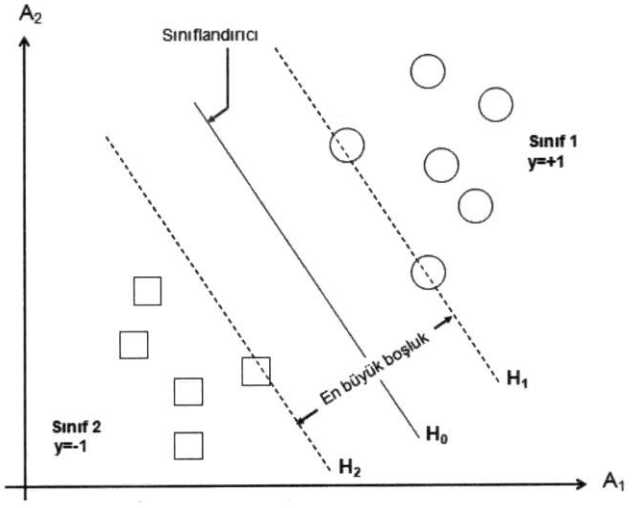
#### 2.3.1.5. Destek Vektör Makinaları

DVM (Destek Vektör Makinesi), iki ayrı sınıfa ait örnekleri birbirinden ayırabilecek en uygun fonksiyonu öğrenme yoluyla hesaplayan bir sınıflandırıcı algoritmadır [128]. Bunun için, sınıflara ait örnekler öncelikle +1 ve -1 değerleri ile etiketlenerek, her iki sınıfa ait örneklerden birbirlerine en yakın olanlar, sınıflar arasındaki sınır bölgesini belirlemek amacıyla kullanılır. Bu sınır içerisinde, sınır genişliğini maksimum yapan fonksiyon aranır. [127]. Hesaplanacak fonksiyon doğrusal olabileceği gibi, doğrusal olmayan fonksiyonlar da iki sınıf arasında ayırıcı olarak hesaplanabilir [128]. Doğrusal olmayan sınıflandırmadaki yüksek başarısı ve güvenilirliği nedeniyle birçok makine öğrenmesi uygulamasında kullanılmaktadır [126].

Şekil 2.7'de gösterilen örnek sınıflandırma grafiğinde, sınıfları ayıran ve aralığı maksimum yapan $H_0$ hiper düzlem fonksiyonu Denklem 2.20'de gösterilmiştir [128].

$$H_0 = W^T \cdot X + b = 0 \tag{2.20}$$

N adet özellik için W ağırlık vektörü $W = \{w_1, w_2, \dots, w_n\}$ olarak yazılırsa, $H_0$ düzlemi Denklem 2.21'de görüldüğü gibi tekrar yazılabilir [128].




Şekil 2.7 Destek vektör makinesinde sınıf örnekleri arasındaki en geniş aralık ve aralığın orta noktasındaki $H_0$ sınıflandırma fonksiyonu

$$H_0 = \sum_{i=1}^{n} w_i \cdot x_i + b = 0 \tag{2.21}$$

Şekil 2.7'de gösterilen iki ayrı sınıfın örnekleri arasında maksimum aralık mesafesini sağlayan $H_1$ ve $H_2$ hiper düzlemleri, sırasıyla Denklem 2.22 ve Denklem 2.23 ile ifade edilir [128].

$$H_1 = W^T \cdot X + b = +1 \tag{2.22}$$

$$H_2 = W^T \cdot X + b = -1 \tag{2.23}$$

$H_0$ hiper düzlem fonksiyonu ile ayrılan örneklerden, düzlemin üzerinde ve altında kalanlar sırasıyla Denklem 2.24 ve Denklem 2.25 eşitsizliği ile ifade edilir.

$$W^T \cdot X + b > 0, \;\; y_i = +1 \tag{2.24}$$

$$W^T \cdot X + b < 0, \;\; y_i = -1 \tag{2.25}$$

Denklem 2.24 ve denklem 2.25 eşitsizlikleri birleştirildiğinde, Denklem 2.26'da gösterilen eşitsizlik elde edilir [128].

$$y_i \cdot ( W^T \cdot X + b) \geq +1 \quad \forall i \tag{2.26}$$

$H_1$ ve $H_2$ hiper düzlemleri üzerindeki örneklere destek vektörleri adı verilir. $x_1$ destek vektörü ile $H_2$ hiper düzlemi arasındaki mesafe *d*, Denklem 2.27'de gösterilmiştir.

$$d = \frac{|W^T \cdot X + b|}{\|w\|} = \frac{1}{\|w\|} \tag{2.27}$$

$H_1$ ve $H_2$ hiper düzlemleri arasındaki mesafeyi maksimum yapabilmek için *w* vektörünü minimum yapmak gerekir. Bunun için, Denklem 2.20 şartını sağlayan minimum *w* değeri Denklem 2.28 ile hesaplanır [128].

$$\min_w \frac{1}{2} \cdot \|w\|^2 \tag{2.28}$$

### 2.3.2. Sınıflandırıcıların Değerlendirme Ölçütleri

Sınıflandırma algoritmaları örnek veriler üzerinde öğrenme işlemini tamamladıktan sonra, öğrenip öğrenmediği ya da ne kadar öğrendiği değerlendirilmelidir.

Tablo 2.1 Sınıflandırma işlemi sonucunda elde edilen karmaşıklık matrisi

| | | **Sınıflandırma Sonucu** | |
|---|---|---|---|
| | | Artı (Positive) | Eksi (Negative) |
| **Gerçek Sınıf** | Artı (Positive) | DA (TP) | YE (FN) |
| | Eksi (Negative) | YA (FP) | DE (TN) |

Bir sınıflandırıcının değerlendirilebilmesi için, sınıflandırılan test örneklerinin sayılarına göre karmaşıklık matrisi (confusion matrix) oluşturulur. Bu matris, sınıflandırıcının artı ve eksi örneklerin kaç tanesinin doğru, kaç tanesinin yanlış sınıflandırdığını açıkça gösterir. Bir sınıflandırma işlemi sonucunda elde edilen değerlerin oluşturduğu karmaşıklık matrisi Tablo 2.1'de gösterilmiştir [127]. Başarım ölçütleri aynı zamanda, sınıflandırıcıların performanslarını kıyaslama imkânı sağlar [128]. Değerlendirme ölçütlerinde kullanılan terimler şu şekilde tanımlanır;

- **Artı (Positive):** Artı sınıfına ait örneklerin sayısıdır.
- **Eksi (Negative):** Eksi sınıfına ait örneklerin sayısıdır.
- **Doğru (True):** Sınıflandırmanın doğru bir şekilde yapıldığını ifade eder.
- **Yanlış (False):** Sınıflandırmanın hatalı yapıldığını ifade eder.
- **DA – Doğru Artı (TP – True Positive):** Gerçekte Artı olan ve Artı olarak Doğru sınıflandırılan örneklerin sayısıdır.
- **YA – Yanlış Artı (FP – False Positive):** Gerçekte Eksi olan ancak Artı olarak Yanlış sınıflandırılan örneklerin sayısıdır.
- **DE – Doğru Eksi (TN – True Negative):** Gerçekte Eksi olan ve Eksi olarak Doğru sınıflandırılan örneklerin sayısıdır.
- **YE – Yanlış Eksi (FN – False Negative):** Gerçekte Artı olan ancak Eksi olarak Yanlış sınıflandırılan örneklerin sayısıdır.

Sadece karmaşıklık matrisi yoluyla sınıflandırıcıları değerlendirmek ve kıyaslamak her zaman pratik olmayabilir. Bu nedenle, karmaşıklık matrisinde ortaya çıkan değerler üzerinden çeşitli ölçütler geliştirilmiştir.

**Doğruluk (Accuracy)**, doğru sınıflandırılan örneklerin, toplam örnek sayısına oranıdır ve Denklem 2.29'da gösterildiği şekilde hesaplanır [127].

$$Doğruluk = \frac{DA + DE}{DA + DE + YA + YE} \tag{2.29}$$

**Hata (Error)**, yanlış sınıflandırılan örneklerin, toplam örnek sayısına oranıdır ve Denklem 2.30'da gösterildiği şekilde hesaplanır [127].

$$Hata = \frac{YA + YE}{DA + DE + YA + YE} = 1 - Doğruluk \tag{2.30}$$

**DA oranı (TP rate)**, doğru sınıflandırılan artı örneklerin, tüm artı örneklerin sayısına oranıdır ve Denklem 2.31'de gösterildiği şekilde hesaplanır [127].

$$DA\ oranı = \frac{DA}{DA + YE} \tag{2.31}$$

**YA oranı (FP rate)**, yanlış sınıflandırılan eksi örneklerin, tüm eksi örneklerin sayısına oranıdır ve Denklem 2.32'de gösterildiği şekilde hesaplanır [127].

$$YA\ oranı = \frac{YA}{YA + DE} \tag{2.32}$$

**Kesinlik (Precision)**, doğru sınıflandırılan artı örneklerin sayısının, tüm artı olarak sınıflandırılmış örneklerin sayısına oranıdır ve Denklem 2.33'te gösterildiği şekilde hesaplanır [127].

$$Kesinlik = \frac{DA}{DA + YA} \tag{2.33}$$

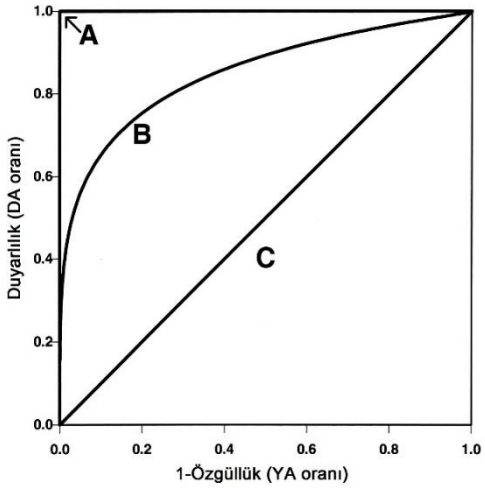
**Anma (Recall)**, doğru sınıflandırılan artı örneklerin sayısının, tüm gerçek artı örneklerin sayısına oranıdır ve Denklem 2.34'te gösterildiği şekilde hesaplanır [127].

$$Anma = \frac{DA}{DA + YE} \tag{2.34}$$

**Duyarlılık (Sensitivity)**, DA oranı ve Anma ile ifade edilen ölçütün aynısıdır ve Denklem 2.31'de gösterildiği şekilde hesaplanır [127].

**Özgüllük (Specificity)**, doğru sınıflandırılan eksi örneklerin sayısının, toplam eksi örnek sayısına oranıdır ve Denklem 2.35'te gösterildiği şekilde hesaplanır. Negatif örneklerin ne kadar iyi sınıflandırıldığının bir ölçütüdür [127].

$$Özgüllük = \frac{DE}{YA + DE} \tag{2.35}$$




Şekil 2.8 Alıcı işlem karakteristik eğrisi. A eğrisi ideal sınıflandırmayı, B eğrisi olağan sınıflandırmayı, C doğrusu rasgele sınıflandırmayı göstermektedir.

**AİK (Alıcı İşlem Karakteristiği) (ROC: Receiver Operating Characteristics)**, dikey ekseni DA oranını, yatay ekseni YA oranını gösteren bir eğridir. DA ve YA sınıflandırma oranlarının birlikte yorumlanmasına olanak verir. Şekil 2.8'de örnek bir AİK eğrisi gösterilmiştir. Bu şekilde A eğrisi, sol üst köşeyle ve dolayısıyla düşey eksenle çakışıktır. İdeal ya da mükemmel sınıflandırıcı olarak tanımlanan A eğrisine ait sınıflandırıcı, tüm gerçek artı örnekleri doğru bir şekilde artı olarak sınıflandırmıştır. C köşegen doğrusu, yanlışların ve doğruların birbirine eşit olduğu en kötü sınıflandırıcıyı temsil etmektedir. Buna göre, çizilen AİK eğrisi ne kadar sol üst köşeye yakın ise, bu sınıflandırıcının daha iyi olduğu söylenebilir ve sınıflandırıcı tercihi için bir seçim kıstası olarak düşünülebilir. Sayısal olarak kıyaslama yapabilmek amacıyla, sınıflandırıcıların AİK eğrisinin altında kalan alanları kıyaslanır. Bu durumda, en kötü sınıflandırıcı alanı 0,5 iken, ideal bir sınıflandırıcının eğri alanı bir olacaktır. Sonuç olarak, eğri alanı daha büyük olan sınıflandırıcının daha iyi olduğu söylenebilir [127].

**F-Ölçütü (F-Measure)**, duyarlılık ve kesinlik ölçütlerinin birlikte değerlendirilebilmesi için, bu iki ölçütün harmonik ortalaması alınır. F-Ölçütü, denklem 2.36'da gösterildiği şekilde hesaplanmaktadır [128].

$$F - \text{Ölçütü} = 2 \cdot \frac{Duyarlılık \cdot Kesinlik}{Duyarlılık + Kesinlik} \tag{2.36}$$

# BÖLÜM 3

# UYGULAMA

## 3.1. Sürücü Uykululuk Tespit ve Uyarı Sisteminin Genel Tasarımı

Tasarlanması hedeflenen sürücü uykululuk tespit ve uyarı sisteminin kısıtlamaları şu şekilde sıralanabilir;

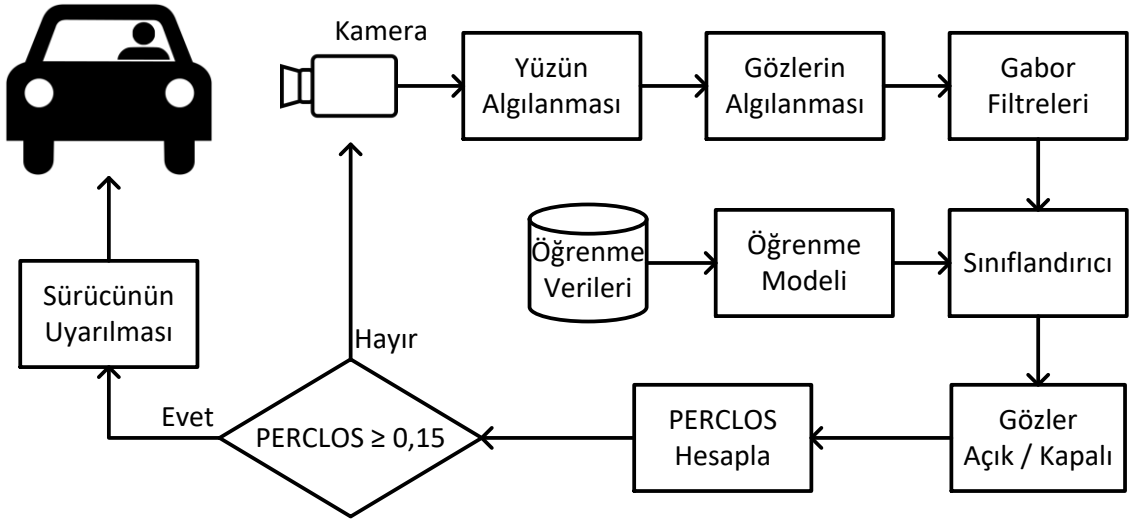
- Sürücüye müdahalesiz bir sistem olacaktır.
- Görüntü işleme teknikleri kullanılacaktır.
- Makine öğrenme algoritmaları kullanılacaktır.
- Gerçek zamanlı çalışacaktır.

Bu kısıtlamalara göre tasarımı gerçekleştirilen gerçek zamanlı sürücü uykululuk tespit ve uyarı sisteminin basitleştirilmiş genel akış diyagramı Şekil 3.1'de gösterilmiştir.

Bu diyagrama göre ilk olarak, hazırlanan sürüş benzetim ortamından sürücünün yüzünü açık bir şekilde içeren görüntü bir kamera yardımıyla elde edilmiştir. Bu görüntü içerisinde önce sürücünün yüz görüntüsü algılanarak belirlenmiş, ardından yüz görüntüsünün içerisinde gözlerin bulunduğu bölge elde edilmiştir. Bu bölgede elde edilen göz görüntülerinin değerlendirilmesi iki aşamada gerçekleştirilmiştir.

Birinci aşamada elde edilen görüntüler kaydedilmiştir. Kaydedilen göz görüntüleri, görüntü içerisindeki gözlerin açık ya da kapalı olma durumlarına göre etiketlenerek sınıflandırılmıştır. Sınıflandırma işlemi sonrasında bu görüntülerin öznitelikleri çıkarılarak veri dosyaları oluşturulmuştur. Bu veri dosyaları makine öğrenme algoritmalarının sınanması amacıyla kullanılmıştır.

İkinci aşamada elde edilen göz görüntüleri kaydedilmemektedir. Canlı kamera görüntüsü, daha önce oluşturulan öğrenme modeli ile sınıflandırılarak açık ya da kapalı göz sonucuna göre sürücünün uykululuk tespitinde kullanılmıştır.




Şekil 3.1 Tasarlanan sistemin basitleştirilmiş blok diyagramı ve işlem akış şeması

Uygulama geliştirme aşamasında şelale (waterfall) yazılım geliştirme modeli esas alınmıştır. Buna göre, problem belirlendikten sonra, sistem analizi ve tasarımı yapılmıştır. Ardından modüllerin kodlanması test edilmesi süreci, modüllerin birleştirilmesi ve son olarak sistem testi gerçekleştirilmiştir. Ortaya çıkan hata ve eksiklikler için önceki adımlara dönülerek geliştirme döngüsü takip edilmiştir.

Yazılımın geliştirilmesinde, Microsoft® Visual Studio 2015 Community geliştirme ortamı ve C# programlama dili [130], OpenCV kütüphanesi [131] ve bu kütüphanenin C# ile kullanılmasını sağlayan Emgu CV sarmalama kütüphanesi [132] , Weka kütüphane dosyaları [133] ile Java fonksiyonlarının C# ile kullanılmasını sağlayan IKVM kütüphanesi [134] kullanılmıştır.

Yazılım geliştirme ve geliştirilen yazılımların test süreçlerinde Intel® i7 4510U 2,6GHz, 8 GB RAM taşınabilir bilgisayar ve Windows 10 Home 1607 işletim sistemi kullanılmıştır.

### 3.2. Görüntülerin Elde Edilmesinde Kullanılan Sürüş Benzetim Ortamı

Sürüş anındaki sürücü görüntülerini elde etmek amacıyla bir sürüş benzetim ortamı kullanılmıştır. Sürüş benzetim ortamı, sürüş benzetim uygulaması ve bu uygulamanın üzerinde çalıştığı bir bilgisayar, direksiyon, vites, pedallardan oluşan sürüş düzeneği ve sürücü görüntülerinin alındığı bir kamera ile kızılötesi aydınlatma sisteminden oluşmaktadır.



Şekil 3.2 City Car Driving isimli sürüş benzetim uygulamasının örnek ekran görüntüsü

Sürüş benzetim uygulaması için City Car Driving isimli araç sürüş benzetim uygulama lisansı alınmıştır. Bu uygulama, sürücünün belirli zamanlarda belirli noktalara gitmesini gerektiren çeşitli görevler verilmesi üzerine kurgulanmıştır. Bu uygulamada trafik ışıkları, trafik tabelaları, yol çizgileri, diğer sürücüler ve yayalar olmak üzere gerçeklik derecesi yüksek bir trafik ortamı benzetimi yapılmıştır. Ayrıca sürüş sırasında trafik kurallarına uyulması gerekmektedir. Yapılan her kural ihlalinde, bununla ilgili bir uyarı gösterilmekte ve puan kaybı olmaktadır. Bu şekilde, sürücülerin, trafik kurallarına uygun bir şekilde hedef noktaya sürüş yapması sağlanmaktadır [135]. Bu uygulamanın örnek bir ekran görüntüsü Şekil 3.2'de gösterilmiştir.

Gerçekçi bir sürüş deneyimi sağlamak amacıyla sürüş benzetim uygulamasının, Logitech Driving Force GT tepki verebilen bir direksiyon, vites ve pedallar ile kullanılması sağlanmıştır. Bu sayede, gerçek bir araç kullanımı sırasında yapılması gereken sinyal verme, vites değiştirme gibi aktiviteler benzetim ortamına uyarlanmıştır.

Ayrıca ekran görüntüsü bir projeksiyon sistemi ile duvara yansıtılarak, daha büyük ve gerçekçi olması sağlanmıştır. Böylece daha geniş görüş açısı ile sürücünün gerçekçi bir sürüş deneyimi içerisinde araç kullanması sağlanmıştır.

Ek olarak, sürüş benzetim ortamına dahil edilen hoparlörler ile trafik ortamında oluşan motor, korna, fren gibi seslerle sürücüye işitsel geri besleme sağlanmıştır.



Şekil 3.3 Sürüş benzetim ortamı ve sürücü görüntüsünün elde edilmesi

Sürüş benzetim ortamını kullanan sürücü, benzetim ortamında istenen görevleri yerine getirmek için trafikte sürüş yaparken, uygun bir açıda yerleştirilen bir kamera ile sürücünün yüz görüntüsü bilgisayar ortamına aktarılmıştır.

Görüntüler, Microsoft LifeCam Studio kamerası kullanılarak elde edilmiştir. Yüksek çözünürlüklü görüntü, geniş açılı cam mercek, otomatik pozlama ve otomatik odaklama bu kameranın öne çıkan özellikleridir.

Ortamdan kaynaklanan yetersiz ve değişken aydınlatmanın etkisini azaltmak ve görüntü kalitesini yükseltmek amacıyla kızılötesi aydınlatma kullanılmıştır. Kızılötesi dalga boyu, insan gözünün algılama sınırlarının dışında olduğundan insan gözü tarafından görülemez. Ancak kamerada kullanılan sensörler kızılötesi dalga boylarına duyarlı olduğundan, insan gözünü rahatsız etmeden gerekli aydınlatma sağlanabilmektedir.

Öğrenme verilerini elde etmek amacıyla, gönüllülerin bu benzetim uygulamasında sürüş yapması sağlanmış ve bu sırada elde edilen görüntüler bir video kaydı şeklinde bilgisayar ortamına kaydedilmiştir. Bu video kayıtları daha sonra açık ve kapalı şeklinde etiketlenmiş göz görüntülerinin elde edilerek görüntü veri tabanlarının oluşturulmasında kullanılmıştır.

Sürüş benzetim ortamı aynı zamanda, bu tez çalışmasında geliştirilen gerçek zamanlı uykululuk tespit ve uyarı uygulamasının test edilmesinde ve performans değerlerinin elde edilmesinde kullanılmıştır.



Şekil 3.4 Sürüş benzetim ortamına yerleştirilen kameradan elde edilen sürücü görüntülerinden iki örnek

### 3.3. Sürücü Görüntüsünün Elde Edilmesi ve Göz Görüntülerinin Kırpılması

Sürüş benzetim ortamından görüntüler, kızılötesi aydınlatma sistemi ile birlikte Microsoft LifeCam Studio 1080p USB kamera kullanılarak elde edilmiştir. Görüntü, sürücünün doğrudan yüz görüntüsünü alacak şekilde ön cepheden fakat görüşünü engellemeyecek şekilde gösterge konsolunun üzerinden alınmıştır. Şekil 3.4'te benzetim ortamından alınan iki farklı sürücünün örnek görüntüleri gösterilmiştir.

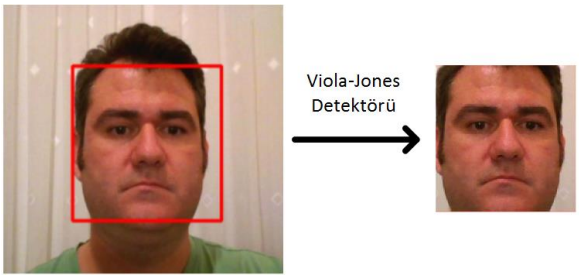
Gündüz koşullarında elde edilen görüntülerde, araç ve sürücü sürekli hareket halinde olduğundan, ışık kaynağının yönü ve açısı sürekli değişmekte ve elde edilen görüntüler üzerinde görüntü işlemeyi zorlaştıran gölgeler ve karanlık bölgeler oluşturmaktadır. Gece sürüş koşullarında ise görüntü şartları daha kötüleşmekte, ışığın yetersiz kalması nedeniyle sürücü yüz ve göz görüntüleri belirsizleşmektedir.

Dış ortam aydınlatmasından kaynaklanan değişkenlikleri azaltmak ve görüntü kalitesini arttırmak amacıyla kızılötesi aydınlatma kullanılmıştır. Böylece, kontrolsüz ve yetersiz olabilen ortam ışığının etkileri en aza indirilerek görüntü üzerindeki değişkenlikler azaltılmıştır.

Elde edilen canlı video görüntüsü ilk olarak 320x240 piksel uzaysal çözünürlüğe indirgenmiştir. Böylece piksel yoğunluğu indirgenerek hesaplama performansının arttırılması hedeflenmiştir. Uzaysal çözünürlüğü düşürülen görüntüler, görüntü işleme algoritmalarında renk bilgisi kullanılmadığı için gri ölçekli görüntülere dönüştürülmüştür. Renkli görüntüler, Denklem 3.1'de gösterilen formül kullanılarak gri ölçeğe dönüştürülmüştür [136].



Şekil 3.5 İki örnek Haar-benzeri özelliğin uygulanması




Şekil 3.6 Görüntünün Viola-Jones detektörü ile tespit edilerek kırpılması

$$\acute{Y} = 0{,}2126 \cdot R + 0{,}7152 \cdot G + 0{,}0722 \cdot B \tag{3.1}$$

Bu denklemde $\acute{Y}$, ışık parlaklığını (luminance), R, kırmızı (red), G, yeşil (green), B, mavi (blue) renk parlaklık değerlerini göstermektedir.

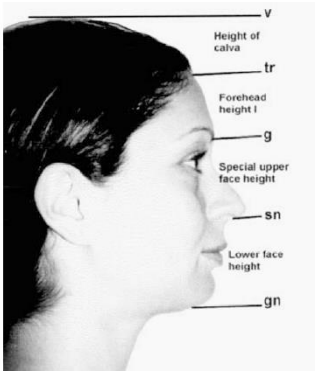
Sürücünün yüz görüntüsünü elde etmek için Viola-Jones detektörü olarak adlandırılan algoritma kullanılmıştır. Bu algoritma, Haar-benzeri özellikleri kullanarak görüntü içerisinde daha önce öğretilen nadir bir nesneyi bulur. Çok sayıda zayıf sınıflandırıcının ardışık olarak işletilmesi prensibiyle çalışır. Herhangi bir sınıflandırıcı negatif yanıt verdiğinde işlem sonlandırılır ve kalan sınıflandırıcılar işletilmez. Böylece işlem yükünden tasarruf edilerek yüksek hızlı bir çalışma sağlanır [123, 124]. Şekil 3.5'te örnek olarak verilen iki Haar-benzeri özelliğin yüz görüntüsüne uygulanması gösteriliyor.

Kamera görüntüsü içerisinde bir yüz görüntüsü tespit edildiğinde, algoritma tarafından döndürülen bu yüz görüntüsü koordinatları kullanılarak görüntü kırpılmıştır. Yüz bulmak için gerekli olan öğrenme veri tabanı için, OpenCV tarafından sağlanan hazır veri dosyaları kullanılmıştır [131]. Şekil 3.6'da kamera görüntüsünden Viola-Jones algoritması kullanılarak kırpılan yüz görüntüsü gösterilmektedir.

Sınıflandırma işleminde sürücünün göz görüntüleri kullanılacağından, yüz görüntüsü üzerinden göz görüntülerinin de ayrıca kırpılması gerekmektedir.

Bunun için, yüz görüntüsünde gözlerin bulunduğu alan, düşey geometrik oranlar yoluyla bulunarak kırpılmıştır. Bu geometrik oranlar Şekil 3.7'de gösterilmiştir [137].

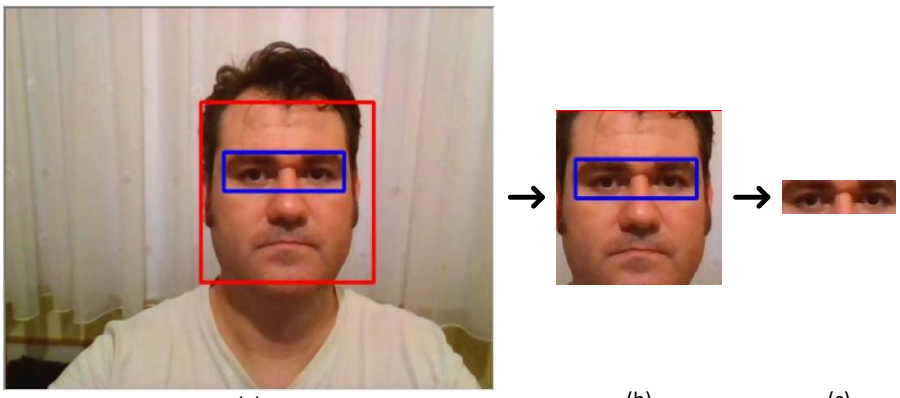
Alın yüksekliği (tr-g) ile üst yüz yüksekliğinin (g-sn) toplamının toplam yüz yüksekliğine (tr-gn) oranı (1,625) kullanılarak gözlerin olduğu bölge kırpılmıştır [137]. Bu oranın, değeri 1,61 olan altın orana çok yakın olması dikkat çekicidir.




Şekil 3.7 Göz bölgesinin belirlenmesinde kullanılan düşey yüz oranları



Şekil 3.8 Sol ve sağ göz bölgelerinin düşey ve yatay olarak ayrılması

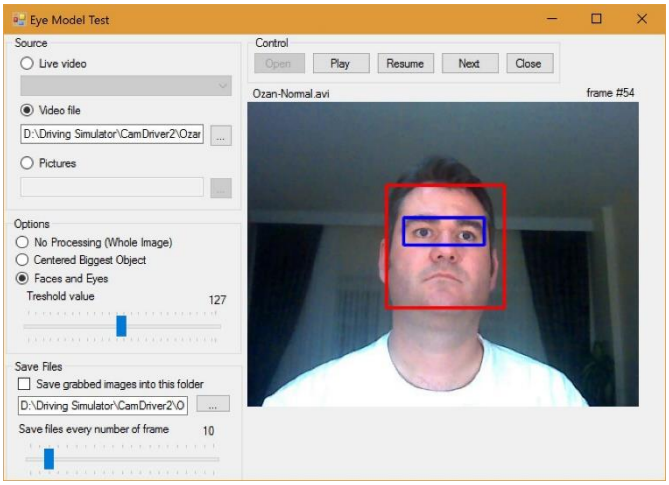


(a) (b) (c)

Şekil 3.9 Sırasıyla görüntü kırpma işlemleri a) kamera görüntüsü b) kırpılan yüz görüntüsü c) yüz görüntüsünden kırpılan göz görüntüsü

Yüz görüntüsü, yatay olarak ikiye bölündükten sonra, üstte kalan alan düşey olarak tam ortadan tekrar ikiye bölünerek sol ve sağ göz için iki ayrı görüntü elde edilmiştir. Şekil 3.8'de sol ve sağ göz bölgeleri ayrılmış örnek sürücü yüz görüntüsü gösterilmiştir.

Sol göz ve sağ göz, kırpılan bu sol ve sağ göz bölgesi görüntülerinin içerisinde Viola-Jones algoritmasıyla paralel işlemle eş zamanlı olarak bulunmuştur. Sol ve sağ göz bulmak için gerekli olan öğrenme veri tabanı için, OpenCV tarafından sağlanan hazır veri dosyaları kullanılmıştır [131]. Kırpılmış iki ayrı görüntü üzerinde paralel arama yapmak, görüntü işleme hızının arttırılmasına önemli katkı sağlamıştır. Sol ve sağ göz görüntüleri paralel olarak bulunduktan sonra bu görüntüler, Denklem 3.2'de gösterilen formül kullanılarak tek bir görüntü olacak şekilde birleştirilmiştir. Böylece, her iki gözü de içeren minimum boyutlara sahip bir görüntü elde edilmiştir. Kırpma işlemlerinin aşamalarını gösteren örnek bir ekran çıktısı Şekil 3.9'da gösterilmiştir.




Şekil 3.10 Sürücü görüntülerinin elde edilmesini sağlayan uygulama arabirimi

$$x = x_l, y = \min(y_l, y_r), w = x_r + w_r - x_l, h = \max(h_l, h_r) \quad (3.2)$$

Bu denklemde *x* ve *y*, koordinatları, *w* genişliği, *h* yüksekliği temsil etmektedir. Sol ve sağ gözler sırasıyla *l* ve *r* indisleri ile belirtilmektedir. Göz görüntülerinin kırpılarak elde edilmesinde kullanılan algoritmanın akış diyagramı EK-F'de verilmiştir.

Makine öğrenme algoritmalarının eğitiminde kullanmak amacıyla göz görüntülerini otomatik olarak kırparak kaydeden bir uygulama geliştirilmiştir. Bu uygulama, canlı olarak kameradan ya da önceden kaydedilmiş bir video görüntüsünden göz görüntülerini kırparak ayrı dosyalar halinde kaydetmektedir. Geliştirilen bu uygulamanın örnek arayüz görüntüsü Şekil 3.10'te gösterilmiştir.

### 3.4. Görüntülerin Sınıflandırılması ve Görüntü Veri Tabanı

Makine öğrenme algoritmalarının eğitimi için, öğrenme görüntülerinin gözlerin açık ya da kapalı olması durumuna göre sınıflandırılmış olmaları gerekmektedir. Sınıflandırma işlemi, hatalı verilere neden olmamak için kontrol edilerek elle yapılmıştır.

Sınıflandırma işlemi tamamlandığında, görüntü dosyalarının her biri, ait olduğu sınıfın adını taşıyan klasörlerin içerisine taşınmış bulunmaktadır. Bu şekilde, dört ayrı kişiye ait görüntüler sınıflandırılarak bir görüntü veri tabanı oluşturulmuştur. Makine öğrenme algoritmalarının sınanması amacıyla oluşturulan görüntü veri tabanlarına ait bilgiler Tablo 3.1'de gösterilmiştir.

Tablo 3.1 Dört farklı kişiden elde edilen görüntülerle oluşturulan görüntü veri setleri

| Veri Seti | Açıklama | Açık Göz Sayısı | Kapalı Göz Sayısı | Toplam Veri Sayısı |
|---|---|---|---|---|
| A | A kişisi (Erkek) | 73 | 102 | 175 |
| B | B kişisi (Erkek) | 189 | 104 | 293 |
| C | C kişisi (Erkek) | 130 | 129 | 259 |
| D | D kişisi (Bayan) | 156 | 194 | 350 |
| **TOPLAM** | | **548** | **529** | **1077** |

### 3.5. Görüntü Özelliklerinin Çıkarılması

Görüntülerin makine öğrenme algoritmalarına veri sağlayabilmesi için, bu görüntülerden özelliklerin çıkarılması gerekmektedir. Bu özellikler, görüntü içerisindeki nesnelerin kenarları, renk bilgisi ya da uygulanan çeşitli filtrelerin dolaylı ya da doğrudan çıktıları olabilir. Böylece görüntü içeriğine bağlı olarak sayısal veriler elde edilebilir.

Bu çalışmada, görüntü özelliklerinin çıkarımı için Gabor dalgacık dönüşümleri, ya da diğer bir ismiyle Gabor filtreleri kullanılmıştır. Bu filtreler, farklı frekans ve genliklerde sinyal analizi yapmak amacıyla kullanılan fonksiyonlardır. Görüntü üzerinde içerik analizi yapabilmek için iki boyutlu Gabor dalgacıkları kullanılmıştır.

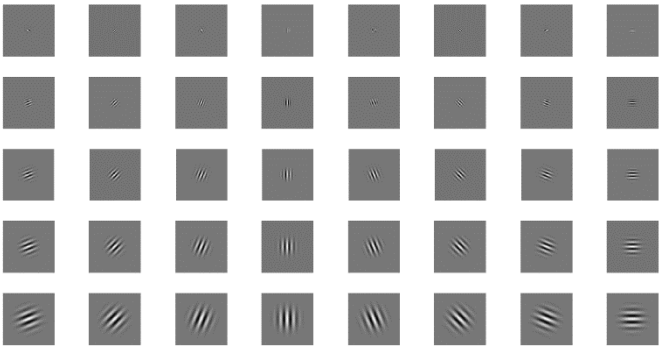
Görüntü özelliklerinin çıkarılması amacıyla kullanılmak üzere beş farklı ölçekte ve 0°, 22,5°, 45°, 67,5°, 90°, 112,5°, 135,0°, 157,5° olmak üzere sekiz farklı açıda toplam 40 Gabor dalgacığı oluşturulmuştur. Gabor dalgacıklarını oluşturmak için Denklem 3.3'te gösterilen eşitlik kullanılmıştır [138].

$$g(x, y; f_0, \theta) = \frac{f_0^2}{\pi \cdot \gamma \cdot \eta} \cdot e^{-\frac{f_0^2}{\gamma^2}\cdot\hat{x}^2+\frac{f_0^2}{\eta^2}\cdot\hat{y}^2} \cdot e^{-i2\pi f_0 \hat{x}} \tag{3.3}$$

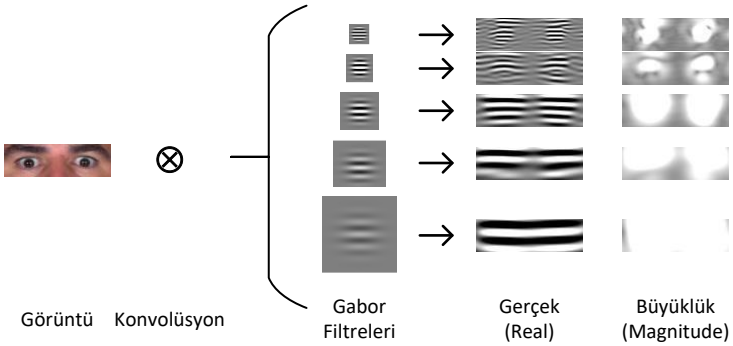
Burada $f_0$ filtre frekansını, θ döndürme açısını, γ ve η keskinliği ifade eden katsayılardır. $\hat{x}$ ve $\hat{y}$ olarak belirtilen değişkenlerin ifadeleri sırasıyla Denklem 3.4 ve Denklem 3.5 ile gösterilmiştir.

$$\hat{x} = x \cdot cos\theta + y \cdot sin\theta \tag{3.4}$$

$$\hat{y} = -x \cdot sin\theta + y \cdot cos\theta \tag{3.5}$$



Şekil 3.11 Beş farklı ölçek ve sekiz farklı açıda Gabor dalgacıkları




Şekil 3.12 Görüntü üzerinde gerçekleştirilen konvolüsyon işlemi ve örnek çıktıları

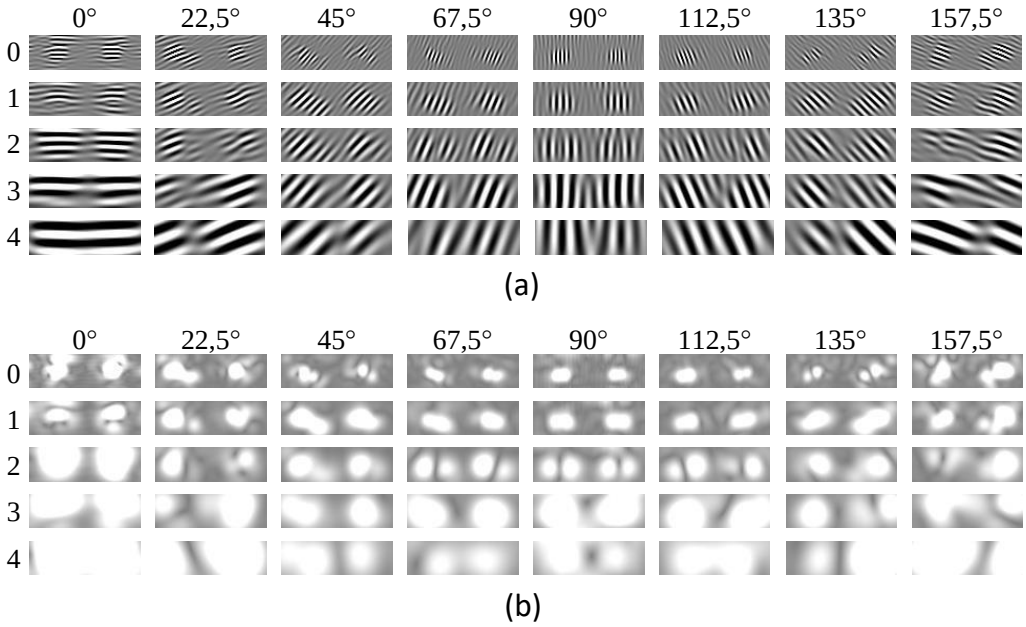
Hesaplamalarda kolaylık olması amacıyla frekans ve açı değişkenleri *i* indis numarasına bağlı olarak sırasıyla Denklem 3.6 ve Denklem 3.7'de gösterildiği şekilde hesaplanmıştır.

$$\theta_i = \frac{2 \cdot \pi \cdot i}{8}, \qquad i \in 0..7 \tag{3.6}$$

$$f_i = k^{-i} \cdot f_{max}, \qquad i \in 0..4 \tag{3.7}$$

Denklem 3.7'de $k$ katsayısı $\sqrt{2}$ olarak alınmıştır. Hesaplama yükünden kaçınmak amacıyla Gabor filtreleri uygulama açılışında bir kez hesaplanarak tüm konvolüsyon işlemlerinde kullanılmıştır. Hesaplanan Gabor filtrelerinin görüntüleri Şekil 3.11'de gösterilmiştir. Sadece 0° için 5 farklı ölçekteki Gabor filtreleri ile gerçekleştirilen konvolüsyon işleminin örnek çıktı görüntüleri Şekil 3.12'de gösterilmiştir.

Özniteliklerin çıkarılmasında görüntülerin Gabor filtre elemanlarıyla konvolüsyonu hesaplanmıştır.




Şekil 3.13 Örnek görüntüye uygulanan sekiz farklı açı ve beş farklı ölçekteki Gabor filtresinin a) gerçek ve b) büyüklük değerlerini veren görüntü çıktıları

Konvolüsyon işlemi, matris formundaki görüntünün her bir elemanının, çarpan matristeki aynı konumdaki elemanlarla çarpılmasını ve çıktının sonuç matrisi üzerinde aynı konuma yazılmasını tanımlayan işlemdir.

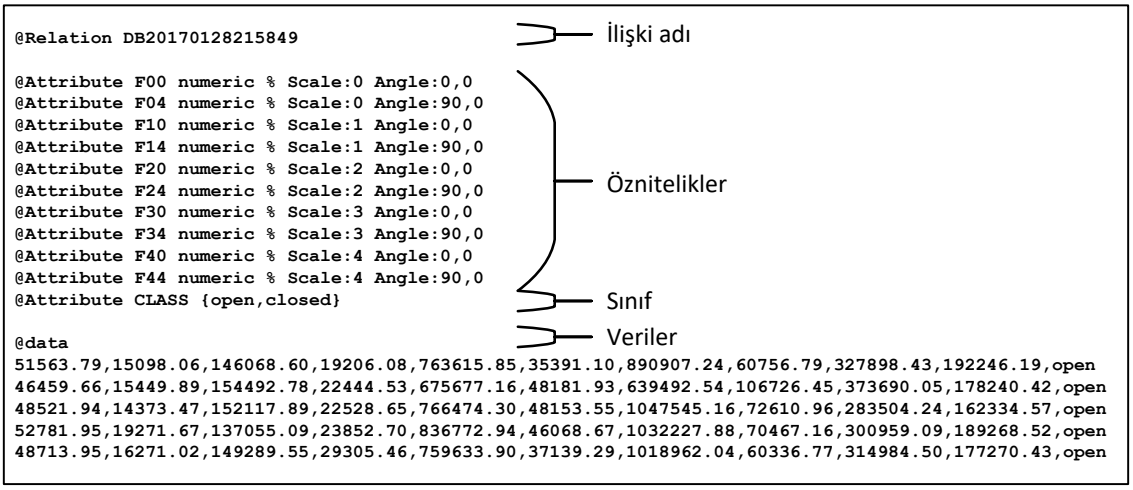
Konvolüsyon sonucunda her bir Gabor filtresi için gerçek (real), sanal (imaginary), büyüklük (magnitude) ve faz açısı (phase) değerleri hesaplanmaktadır. Bu çalışmada, gerçek ve sanal değerlerin bir fonksiyonu olan büyüklük değeri kullanılmıştır. Büyüklük değeri, Denklem 3.8'de gösterildiği şekilde hesaplanmaktadır [139].

$$M(x,y) = \sqrt{R^2(x,y) + I^2(x,y)} \tag{3.8}$$

Burada *M* fonksiyonu büyüklük değerini, *R*, gerçek görüntüyü ve *I* sanal görüntüyü ifade etmektedir. Şekil 3.13'te örnek bir açık göz görüntüsüne ait 40 Gabor filtresinin gerçek ve büyüklük değerlerini gösteren çıktı görüntüleri listelenmiştir.

Hesaplanan her görüntü, bir özniteliği ifade etmektedir. Bunun için, her bir görüntü bir sayısal ifadeye dönüştürülmüştür. Bunu için, her bir çıktı görüntüsünün enerji değeri hesaplanmıştır. Enerji değeri, Denklem 3.9'da gösterildiği şekilde hesaplanmıştır [140].

$$E = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y)^2 \tag{3.9}$$




Şekil 3.14 Örnek bir ARFF veri dosyasının yapısını gösteren ilk 20 satırı

Bu denklemde, *E*, enerji, *x* ve *y* piksel koordinatını, *w* ve *h* sırasıyla görüntü enini ve boyunu, *I* ise *x* ve *y* konumundaki piksel değerini göstermektedir.

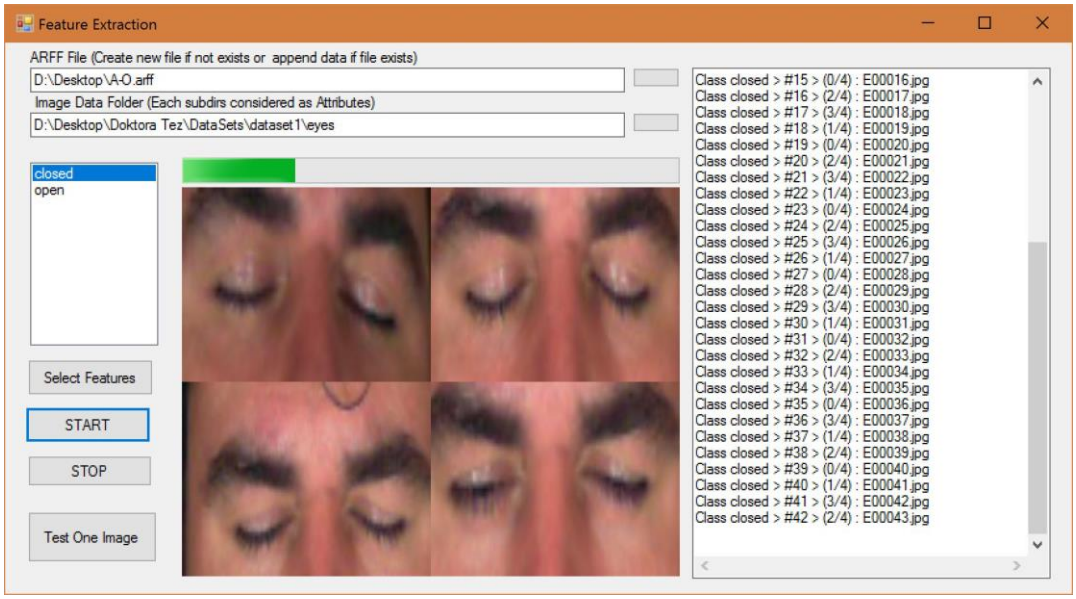
Böylece, görüntü veri tabanındaki sınıflandırılmış tüm görüntüler için toplam 40 Gabor filtresi hesaplanarak 40 adet öznitelik çıkarılmıştır. Bu öznitelikler, her veri seti için ayrı bir ARFF (Attribute Relation File Format – Öznitelik İlişki Dosya Formatı) dosyasına kaydedilmiştir. ARFF, Weka uygulaması tarafından kullanılan metin tabanlı bir veri dosyası formatıdır [133]. Örnek bir ARFF dosyasının içeriği ve yapısı Şekil 3.14'te gösterilmiştir.

Özniteliklerin veri dosyası içerisinde kısa ve anlaşılır bir şekilde temsil edilmesi amacıyla Tablo 3.2'de gösterilen kodlama sistemi kullanılmıştır. Bu kodlama sisteminde, F harfinden sonra gelen iki basamaklı sayının onlar basamağı ölçek indis numarasını, birler basamağı ise açı indis numarasını göstermektedir. Böylelikle veri dosyalarının otomatik oluşturulmasında ya da çözümlenmesinde sistematik bir dönüşüm kolaylığı sağlanmıştır.

Veri tabanlarındaki çok sayıda görüntünün özniteliklerinin kolayca hesaplanabilmesi için bir uygulama geliştirilmiştir. Bu uygulama, veri tabanındaki görüntüleri tarayarak özellikleri hesaplar ve belirtilen ARFF dosyasının içerisine veri satırı olarak ekler. Uygulamada hesaplama performansını arttırmak amacıyla, bilgisayarın çekirdek sayısı kadar görüntü paralel programlama teknikleri ile eş zamanlı olarak hesaplatılmaktadır. Geliştirilen uygulamanın örnek ekran görüntüsü Şekil 3.15'de gösterilmiştir.

Tablo 3.2 Veri dosyasında kullanılan öznitelik isimlerinin kodlama tablosu

| | | **Açı Değerleri (İndis Numaraları)** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | **0° (0)** | **22,5° (1)** | **45° (2)** | **67,5° (3)** | **90° (4)** | **112,5° (5)** | **135° (6)** | **157,5° (7)** |
| **Ölçek İndisleri** | **0** | F00 | F01 | F02 | F03 | F04 | F05 | F06 | F07 |
| | **1** | F10 | F11 | F12 | F13 | F14 | F15 | F16 | F17 |
| | **2** | F20 | F21 | F22 | F23 | F24 | F25 | F26 | F27 |
| | **3** | F30 | F31 | F32 | F33 | F34 | F35 | F36 | F37 |
| | **4** | F40 | F41 | F42 | F43 | F44 | F45 | F46 | F47 |




Şekil 3.15 Görüntü özelliklerinin otomatik çıkarımını sağlayan uygulama ara yüzü

Ek olarak bu uygulamanın, Microsoft® Windows HPC işletim sistemine sahip dağıtık iş istasyonlarında çalışacak şekilde tasarlanmış bir türevi oluşturulmuştur. Bu sayede Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü laboratuvarlarında bulunun Trakya HPC yüksek başarımlı hesaplama makinelerinde hesaplamanın dağıtık olarak hızlı bir şekilde gerçekleştirilmesi sağlanmıştır.

### 3.6. Makine Öğrenme Algoritmalarının Sınanması

Gözlerin gerçek zamanlı olarak sınıflandırılmasında kullanılacak en uygun algoritmayı tespit etmek amacıyla, makine öğrenme algoritmaları daha önce oluşturulmuş olan veri dosyaları ile sınanmıştır.

İlk aşamada, oluşturulan veri dosyalarındaki verilerin uygunluğu ve tutarlılığı irdelenmiştir. İkinci aşamada, öznitelikler incelenerek sınıflandırmadaki başarım ölçütlerine göre öznitelik seçimi yapılmıştır. Son olarak makine öğrenme algoritmalarının sınıflandırma performansları çeşitli başarım ölçütleri temel alınarak değerlendirilmiştir.

Makine öğrenme algoritmalarının sınanmasında ve sınıflandırma başarım değerlerinin elde edilmesinde Weka uygulaması kullanılmıştır [133].

### 3.6.1. Veri Setlerinin Hazırlanması

Veri setlerindeki veriler, görüntülerden sistematik bir şekilde hesaplanarak elde edildiğinden, bu veri setlerinde herhangi kayıp değer bulunmamaktadır. Bu nedenle herhangi kayıp değer tamamlama işlemi yapılmamıştır.

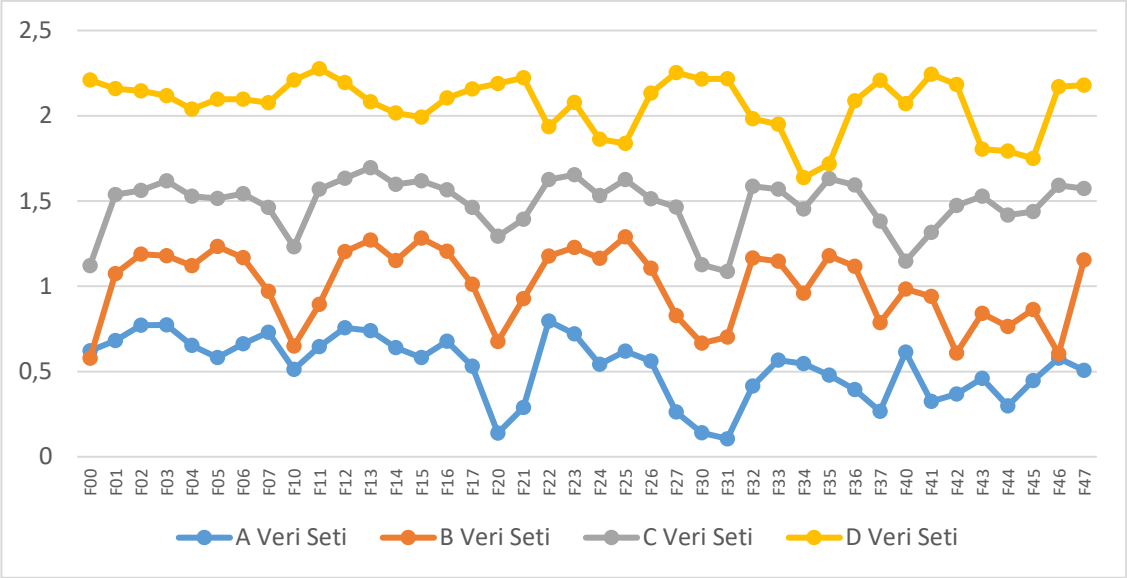
Verilerin hesaplandığı görüntüler, bir video akışındaki ayrık çerçeve görüntüleri olduğundan aynı görüntünün birden fazla veri olarak kaydedilmesi mümkün değildir. Ardışık çerçevelerdeki görüntüler benzer görünseler de, farklı zaman aralıklarında elde edildiğinden aynı görüntüler olarak değerlendirilmemiştir. Bu nedenle veri setlerinde tekrar eden kayıtlar yoktur.

Veri setlerinin oluşturulmasında kullanılan görüntüler elle sınıflandırıldığından, aykırı değerler içeren (çok karanlık, bulanık, hatalı) görüntüler, elle sınıflandırma aşamasında elenmişlerdir. Bu nedenle veri setlerinin sistematik olarak hesaplanmasında uç değerler oluşmamıştır.

### 3.6.2. Özniteliklerin İncelenmesi ve Öznitelik Seçimi

Makine öğrenme algoritmaları veriden bilgi elde ederler. Ancak bu işlemin başarımı verinin niteliği ile sıkı sıkıya bağlıdır. Eğer özniteliklerden bazıları yetersiz bilgi ya da ilgisiz değerlerden oluşuyorsa, bunlar makine öğrenme algoritmalarının doğruluğunu düşürebilir ve hatta bilgi çıkarımını başarısızlığa götürebilir. Bu gibi durumlarda, bahsi geçen ilgisiz ya da yetersiz değerler içeren özniteliklerin elenmesiyle makine öğrenme algoritmalarının doğruluğu iyileştirilebilir. Ayrıca daha az sayıda öznitelik ile hesaplama yükünün azaltılması mümkün olabilir [141].

Kuşkusuz özniteliklerin değerliliği ait olduğu veri setine bağlıdır. Ancak burada, benzer görüntülerden aynı yöntemle hesaplanmış özniteliklere sahip farklı veri setleri için ortak bir öznitelik kümesi elde edilip edilemeyeceği araştırılmıştır.



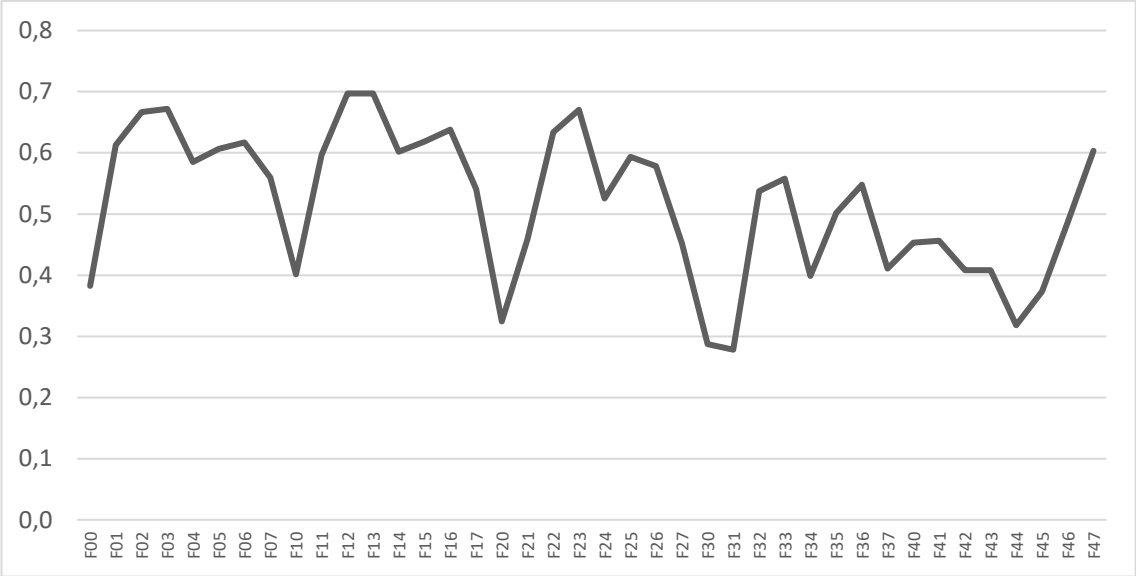

Şekil 3.16 Her bir veri tabanındaki özniteliklerin korelasyon değerleri

Bunun için dört veri setindeki her bir öznitelik için korelasyon (Pearson's) değerleri hesaplanmıştır. Her bir veri setindeki her bir öznitelik için hesaplanan korelasyon değerleri EK-A'da verilmiş ve Şekil 3.16'de grafiksel gösterilmiştir. Bu grafikte, B, C ve D veri setlerine ait korelasyon değerlerine sırasıyla 0,5 puan, 1,0 puan ve 1,5 puan ekleme yapılarak daha anlaşılabilir görünmesi amacıyla grafikler ötelenmiştir.

Şekil 3.16'da gösterilen grafik incelendiğinde, A, B ve C veri setlerinin (erkek) belirli bir düzene sahip olduğu görülmektedir. Özellikle 0° ve 22,5° açılardaki (birler basamağı sıfır ve bir olan) özniteliklerde belirgin düşüş gözleniyor. Ancak D veri seti (bayan) bu düzene uymamaktadır. Bunun, kişilerin göz görüntülerindeki farklılıklardan kaynaklandığı öngörülmüştür.

Veri setlerinden ortak öznitelik çıkarımını gerçekleştirebilmek için, her bir özniteliğin, tüm veri setlerindeki korelasyon değerlerinin aritmetik ortalaması alınmıştır. Korelasyon ortalamaları için elde edilen grafik Şekil 3.17'de gösterilmiştir.

Şekil 3.17'deki grafik incelendiğinde, Şekil 3.16'de ortaya çıkan desen belirgin bir şekilde burada da görülmektedir. Tepe noktalarının biraz altında oluşan basamaklar, burada doğal bir eşik seviyesini göstermektedir. Öznitelik sayısı da göz önüne alınarak eşik seviyesi deneysel olarak 0,6 seçilmiştir. Korelasyon değeri eşik seviyesini geçen öznitelikler F01, F02, F03, F05, F06, F12, F13, F14, F15, F16, F22, F23, F47 olarak tespit edilmiştir. Böylece 40 adet öznitelikten 13 tanesi ortak öznitelik olarak seçilmiştir.




Şekil 3.17 Öznitelik korelasyonlarının tüm veri setleri için ortalama değerleri

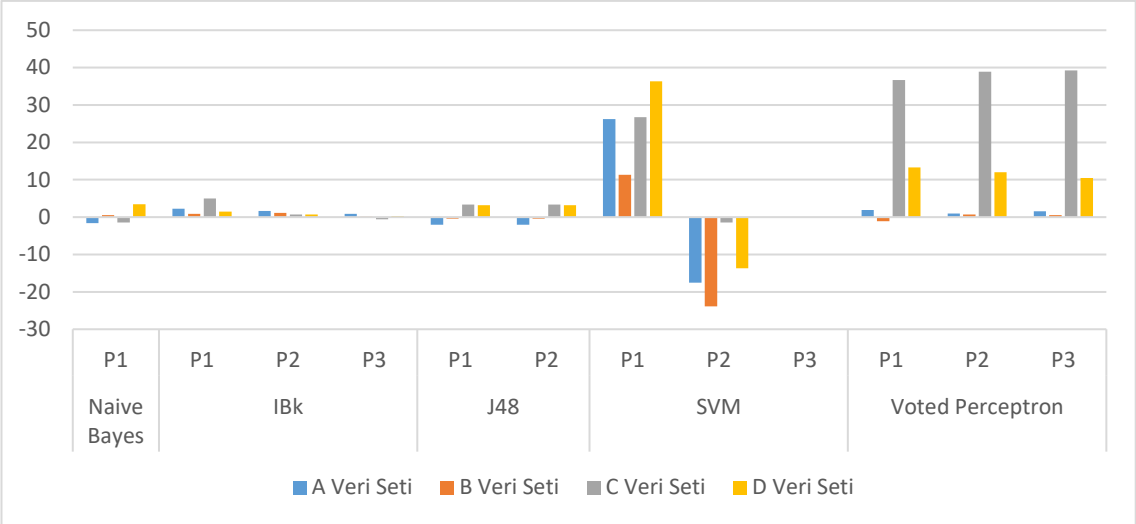
Öznitelik seçimi sonrasında algoritmaların doğruluklarının ne derecede etkilendiğini incelemek amacıyla, tüm algoritmaların tüm veri setleri için öznitelik seçimi öncesi ve sonrasındaki doğruluk değerlerinde meydana gelen değişimleri hesaplanmıştır. Bu doğruluk değişimleri grafiksel olarak Şekil 3.18'de gösterilmiştir.

Şekil 3.18'de gösterilen grafik incelendiğinde, algoritmaların doğruluk değerlerinin genel olarak yükseldiği gözlenmektedir. Göze çarpan en belirgin değişim, SVM (Support Vector Machine – Destek Vektör Makineleri) algoritmasının doğrusal çekirdekli (P1) ve polinom çekirdekli (P2) parametrik varyasyonlarında meydana gelmiştir.

Doğrusal çekirdekli SVM algoritmasında özellik seçimi sonrasında doğruluk ortalama 25 puan yükselmiştir. Düşük korelasyona sahip özniteliklerin elenmesiyle sınıflara ait veriler arasında doğrusal olarak ayrılabilecek bir mesafe açıldığı anlaşılıyor. Voted Perceptron algoritmasının doğruluklarındaki genel yükselişler bunu doğrulamaktadır.

Sonuç olarak, özellik sayısının 40'tan 13'e düşürülerek hesaplamadan tasarruf edilecek olması, doğrulukların olumsuz etkilenmemesi hatta bazı algoritmalarda doğrulun yükselmesi nedeniyle veri setleri üzerinde öznitelik seçimi işlemi gerçekleştirilmiştir.

Bu aşamadan sonra, makine öğrenme algoritmalarının kıyaslanması işleminde elde edilecek olan sonuçlarda, seçilen bu 13 öznitelik kullanılmış ve sadece 13 öznitelik üzerinden başarım ölçütleri hesaplanmıştır.




Şekil 3.18 Öznitelik seçimi öncesi ve sonrasında doğruluk değerlerindeki değişimler

### 3.6.3. Makine Öğrenme Algoritmalarının Sınanması ve Kıyaslanması

Makine öğrenme algoritmalarının kıyaslanmasının amacı, bu çalışmada kullanılan türdeki veriler için sınıflandırma doğruluğu en yüksek ve aynı zamanda gerçek zamanlı işlem için en düşük işlem yoğunluğuna sahip algoritmayı tespit etmektir.

Oluşturulan veri setleri üzerinde sınama yapmak için, birbirinden farklı matematiksel alt yapılara sahip makine öğrenme algoritmalarından parametrik varyasyonları ile birlikte beş farklı makine öğrenmesi algoritması seçilmiştir. Bayes sınıflandırıcı Naïve Bayes, örnek bazında çalışan en yakın komşu sınıflandırıcısı IBk, karar ağaçlarından J48, destek vektör makinesi SVM ve yapay sinir ağlarından Voted Perceptron algoritmaları sınanmak üzere seçilmiştir. Seçilen makine öğrenme algoritmaları ve parametrik değişimleri Tablo 3.3'te listelenmiştir.

Tablo 3.3'te listelenen algoritmalar ve parametrik varyasyonlarının başarım değerleri, her bir veri seti için 10 kat çapraz geçerleme (10-fold cross validation) kullanılarak hesaplanmıştır. Hesaplama sonucunda elde edilen başarım değerlerinin tümünü, veri setleri ve algoritmalar bazında gösteren tablo EK-B'de verilmiştir

Tablo 3.3 Sınamaya tabi tutulan sınıflandırıcılar ve parametrik varyasyonları

| **Sınıflandırıcı** | **Ref.** | **Sınıflandırıcı Parametreleri** | **Parametrik Değişimler** |
|---|---|---|---|
| Naive Bayes | P1 | | Normal Dağılımlı |
| IBk | P1 | -K 1 -W 0 -A LinearNNSearch -A EuclideanDistance -R first-last | 1 Komşu |
| | P2 | -K 5 -W 0 -A LinearNNSearch -A EuclideanDistance -R first-last | 5 Komşu |
| | P3 | -K 10 -W 0 -A LinearNNSearch -A EuclideanDistance -R first-last | 10 Komşu |
| J48 | P1 | -C 0,25 -M 2 | Budanmış ağaç |
| | P2 | -C 0,25 -B -M 2 | Budanmış ikili ağaç |
| SVM | P1 | -S 0 -K 0 -D 3 -G 0,0 -R 0,0 -N 0,5 -M 40,0 -C 1,0 -E 0,001 -P 0,1 -Z -seed 1 | Doğrusal çekirdek |
| | P2 | -S 0 -K 1 -D 3 -G 0,0 -R 0,0 -N 0,5 -M 40,0 -C 1,0 -E 0,001 -P 0,1 -Z -seed 1 | Polinom çekirdek |
| | P3 | -S 0 -K 2 -D 3 -G 0,0 -R 0,0 -N 0,5 -M 40,0 -C 1,0 -E 0,001 -P 0,1 -Z -seed 1 | Radyal tabanlı fonksiyon çekirdek |
| Voted Perceptron | P1 | -I 1 -E 1,0 -S 1 -M 10000 | 1 iterasyon |
| | P2 | -I 3 -E 1,0 -S 1 -M 10000 | 3 iterasyon |
| | P3 | -I 5 -E 1,0 -S 1 -M 10000 | 5 iterasyon |

Veri setlerinde açık ve kapalı göz sınıfları dengeli dağılım gösterdiğinden dolayı kıyaslamalar ilk olarak algoritmaların doğruluklarının kıyaslanması yoluyla gerçekleştirilmiştir. Algoritmaların parametrik varyasyonları ile birlikte veri setlerine göre doğruluk değerleri Tablo 3.4'te gösterilmiştir. Bu tabloda, koyu renkte yazılan değerler, ilgili veri seti sütunundaki en yüksek iki değerdir.

Tablo 3.4 incelendiğinde, en yakın komşu algoritması IBk sınıflandırıcısının performansı göze çarpmaktadır. En yakın komşularının ait oldukları sınıflarına göre işlem yapan bu algoritmanın başarımı, açık ve kapalı gözlere ait örneklerin kendi aralarında kümelenmiş olarak bulunduklarını gösteriyor. Bunun dışında, istatistiksel sınıflandırma yapan Naïve Bayes algoritması ve karar ağaçlarından J48 algoritmasının da oldukça iyi bir sınıflandırma performansı gerçekleştirdiği görülmektedir.

Tablo 3.4 Algoritmaların veri setlerine göre doğruluk değerleri

| Algoritma | Parametreler | A | B | C | D | Ortalama |
|---|---|---|---|---|---|---|
| **Naïve Bayes** | P1 | **96,02** | 96,59 | 88,66 | 91,66 | 93,23 |
| **IBk** | P1 (-K 1) | 93,92 | 96,96 | 90,74 | **94,23** | 93,96 |
| | P2 (-K 5) | 94,88 | **97,03** | **92,52** | **94,60** | **94,76** |
| | P3 (-K 10) | **95,11** | **97,57** | **92,08** | **94,23** | **94,75** |
| **J48** | P1 (-C 0,25) | 92,01 | 95,90 | 87,49 | 92,51 | 91,98 |
| | P2 (-B -C 0,25) | 92,01 | 95,90 | 87,49 | 92,51 | 91,98 |
| **SVM** | P1 (-K 0) | 64,85 | 85,08 | 67,20 | 58,49 | 68,90 |
| | P2 (-K 1) | 83,57 | 96,24 | 69,05 | 88,54 | 84,35 |
| | P3 (-K 2) | 55,27 | 66,90 | 52,89 | 55,43 | 57,62 |
| **Voted Perceptron** | P1 (-I 1 -E 1,0) | 84,53 | 92,87 | 52,89 | 77,20 | 76,87 |
| | P2 (-I 3 -E 2,0) | 87,16 | 93,62 | 52,74 | 80,57 | 78,52 |
| | P3 (-I 5 -E 1,0) | 87,57 | 94,00 | 52,66 | 82,86 | 79,27 |

Tablo 3.5 Algoritmaların veri setlerine göre ROC eğrisi altında kalan alanları

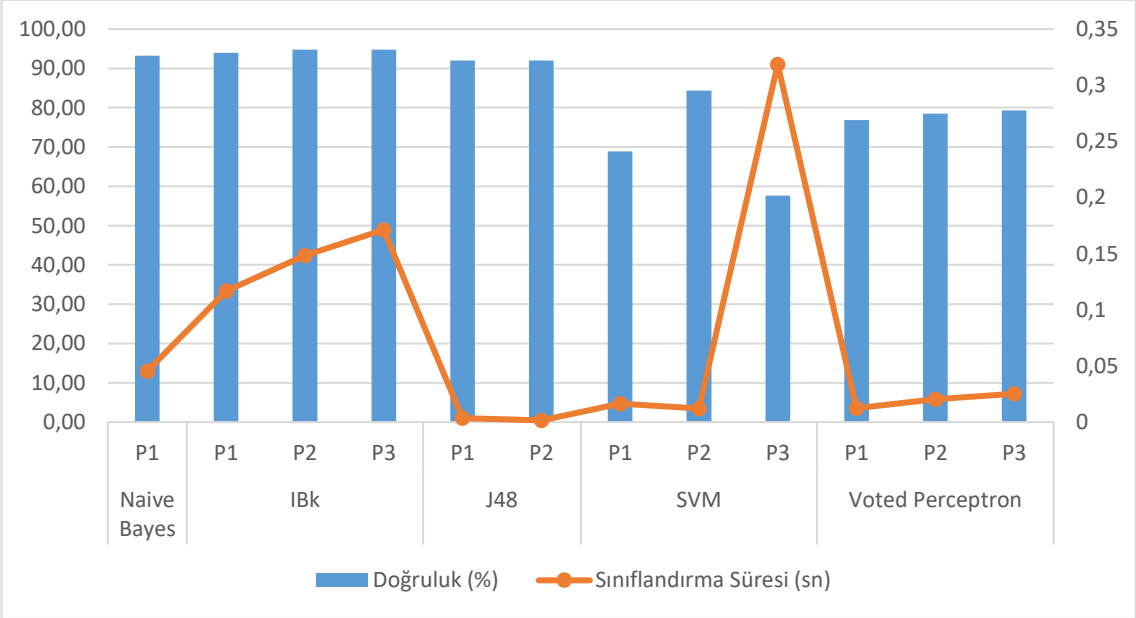
| Algoritma | Parametreler | A | B | C | D | Ortalama |
|---|---|---|---|---|---|---|
| **Naïve Bayes** | P1 | **0,983** | **0,994** | 0,959 | **0,963** | **0,975** |
| **IBk** | P1 (-K 1) | 0,937 | 0,973 | 0,910 | 0,943 | 0,941 |
| | P2 (-K 5) | 0,968 | **0,990** | **0,960** | **0,965** | 0,971 |
| | P3 (-K 10) | **0,976** | **0,990** | **0,965** | 0,962 | **0,973** |
| **J48** | P1 (-C 0,25) | 0,929 | 0,954 | 0,887 | 0,927 | 0,924 |
| | P2 (-B -C 0,25) | 0,929 | 0,954 | 0,887 | 0,927 | 0,924 |
| **SVM** | P1 (-K 0) | 0,646 | 0,839 | 0,673 | 0,582 | 0,685 |
| | P2 (-K 1) | 0,834 | 0,959 | 0,689 | 0,886 | 0,842 |
| | P3 (-K 2) | 0,511 | 0,500 | 0,500 | 0,500 | 0,503 |
| **Voted Perceptron** | P1 (-I 1 -E 1,0) | 0,890 | 0,940 | 0,495 | 0,782 | 0,777 |
| | P2 (-I 3 -E 2,0) | 0,880 | 0,933 | 0,498 | 0,804 | 0,779 |
| | P3 (-I 5 -E 1,0) | 0,879 | 0,935 | 0,499 | 0,823 | 0,784 |

Ek olarak, diğer bir başarım ölçütü olan ROC (Receiver Operating Characteristic – Alıcı İşlem Karakteristiği) eğrisi altında kalan alanlar Tablo 3.5'te gösterilmiştir. Bu tabloda da her bir sütun için en yüksek iki değer koyu olarak işaretlenmiştir.

Tablo 3.5 incelendiğinde, Tablo 3.4'te gösterilen en yüksek doğruluk değerleri ile paralellik gösterdiği görünmektedir. En yüksek ROC eğri alanına sahip algoritmaların IBk ve Naïve Bayes olduğu görülmektedir.

Tablo 3.6 Algoritmaların veri tabanlarına göre saniye olarak sınıflandırma süreleri

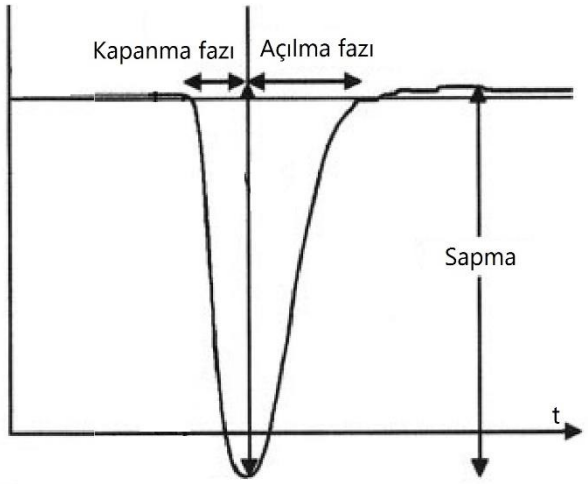
| Algoritma | Parametre | A | B | C | D | Ortalama |
|---|---|---|---|---|---|---|
| **Naïve Bayes** | P1 | 0,036 | 0,057 | 0,05 | 0,039 | 0,046 |
| **IBk** | P1 | 0,103 | 0,124 | 0,081 | 0,159 | 0,117 |
| | P2 | 0,091 | 0,19 | 0,131 | 0,182 | 0,149 |
| | P3 | 0,101 | 0,199 | 0,152 | 0,233 | 0,171 |
| **J48** | P1 | **0,003** | **0,004** | **0,006** | **0,001** | **0,004** |
| | P2 | **0,001** | **0,002** | **0,001** | **0,002** | **0,002** |
| **SVM** | P1 | 0,01 | 0,013 | 0,01 | 0,033 | 0,017 |
| | P2 | 0,01 | 0,008 | 0,018 | 0,012 | 0,012 |
| | P3 | 0,16 | 0,332 | 0,316 | 0,467 | 0,319 |
| **Voted Perceptron** | P1 | 0,007 | 0,012 | 0,01 | 0,021 | 0,013 |
| | P2 | 0,006 | 0,024 | 0,028 | 0,023 | 0,02 |
| | P3 | 0,014 | 0,018 | 0,031 | 0,038 | 0,025 |




Şekil 3.19 Algoritmaların doğruluk değerleri ve saniye cinsinden sınıflandırma süreleri

Doğruluk değerleri hesaplanan algoritmaların gerçek zamanlı çalışma için uygun olup olmadığı ayrıca değerlendirilmiştir. Bu nedenle algoritmaların sınıflandırma için harcadıkları süreler hesaplanarak Tablo 3.6'da gösterilmiştir.

Hem doğruluk hem işlem hızı verilerini birlikte görebilmek için, dört veri tabanının ortalama doğruluk değeri ve sınıflandırma süreleri kullanılarak algoritma bazında doğruluk ve sınıflandırma sürelerini gösteren grafik Şekil 3.19'da verilmiştir. Bu şekilde, çubuk grafikler algoritmaların doğruluklarını, çizgi grafik ise algoritmaların sınıflandırma sürelerini göstermektedir.




Şekil 3.20 Göz kapağının kapanma ve açılma fazları

Sınıflandırma süresi olarak incelendiğinde, J48 karar ağacının tartışmasız olarak en yüksek sınıflandırma hızına sahip algoritma olduğu görülmektedir. Veri setleri için en yüksek doğruluğu ve ROC eğri alanını veren IBk algoritmasının sınıflandırma süresi ile kıyaslandığında, J48 algoritmasının IBk algoritmasından yaklaşık 100 kat hızlı sınıflandırma yaptığı görülmektedir. Doğrulukları kıyaslandığında ise, IBk algoritmasının, J48 algoritmasından %2,78 fark ile daha yüksek doğruluğa sahip olduğu görülüyor. Bu halde algoritma hızlarının, gözlerin açıklık ve kapalılığını gerçek zamanlı olarak sınıflandırabilmesi konusunda bir analiz yapmak gerekmektedir.

Şekil 3.20'de gösterilen grafik, insan göz kapağının kapanma ve açılma fazlarını göstermektedir. İnsan göz kapağının spontane kapanma fazı 92 ±17 milisaniye, açılma fazı 242 ±55 milisaniye olarak rapor edilmiştir. Bilinçli göz kırpmalarında bu süreler kapanma fazı 88 ±13 milisaniyeye, açılma fazı 187 ±34 milisaniyeye düşmektedir [142].

Buna göre insan göz kapağının spontane bir şekilde toplam kapanıp açılma süresi yaklaşık olarak 334 milisaniyedir. Denklem 3.10'da gösterilen formül ile hesaplandığında bu süre yaklaşık olarak 3 Hz frekansa karşılık gelmektedir.

$$F = \frac{1}{T} \tag{3.10}$$

Nyquist kriterine göre, ölçüm yapan cihazın örnekleme frekansı, ölçülmek istenen verinin frekansının iki katından fazla olmalıdır [143]. Bu durumda göz kırpmasını algılayabilmek için en az 6 Hz örnekleme frekansı gereklidir.

Gerçek zamanlı uygulamada, görüntü işleme algoritmalarının çalışma süreleri ihmal edilerek değerlendirildiğinde, IBk algoritması ortalama 146 milisaniye

sınıflandırma süresiyle 6,8 Hz örnekleme frekansına sahip olduğu görülüyor. J48 algoritması 2,5 milisaniye sınıflandırma süresi ile 400 Hz örnekleme frekansına imkan sağlamaktadır.

Görüntü işleme algoritmalarının çalışma süreleri göz önüne alındığında, IBk algoritmasının sınıflandırma süresinin kritik seviyede kaldığı ve görüntü işleme algoritmaları ile birlikte gerekli cevap süresinin yeterli olmayacağı görülmüştür.

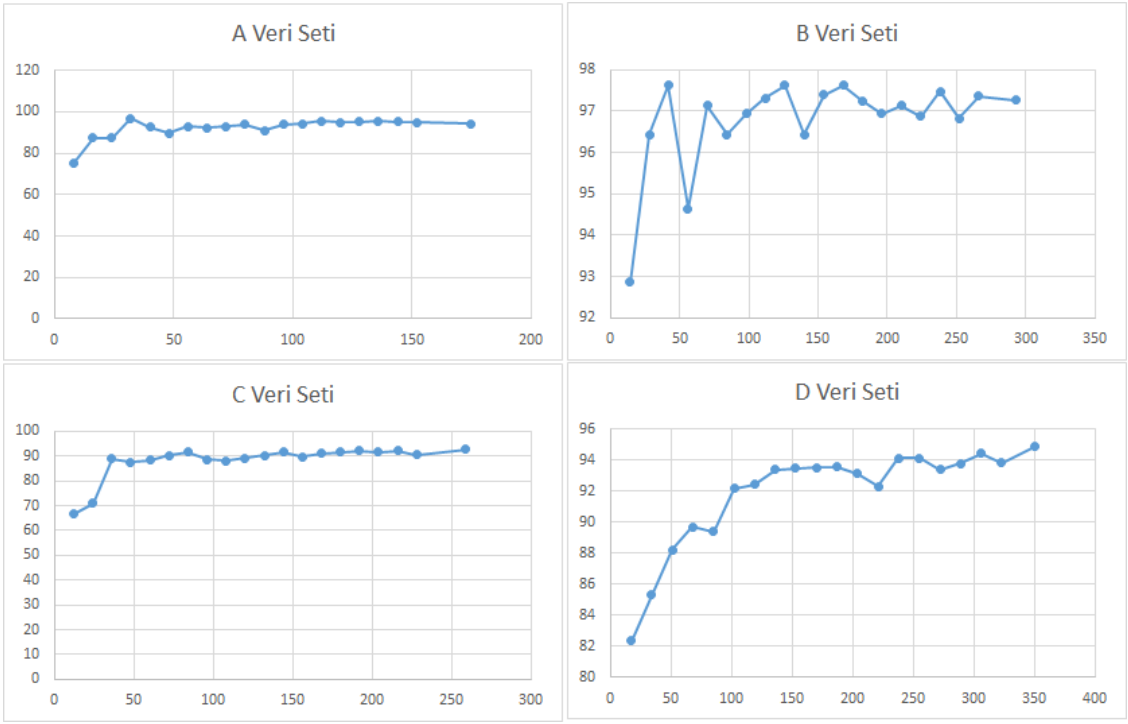
Değerlendirme sonucu olarak, gerçek zamanlı uygulama için ortalama %92 doğruluk ve 2,5 milisaniye sınıflandırma hızı ile J48 algoritmasının kullanılması uygun görülmüştür.

### 3.6.4. Sınıflandırıcıların Öğrenme Seviyelerinin Değerlendirmesi

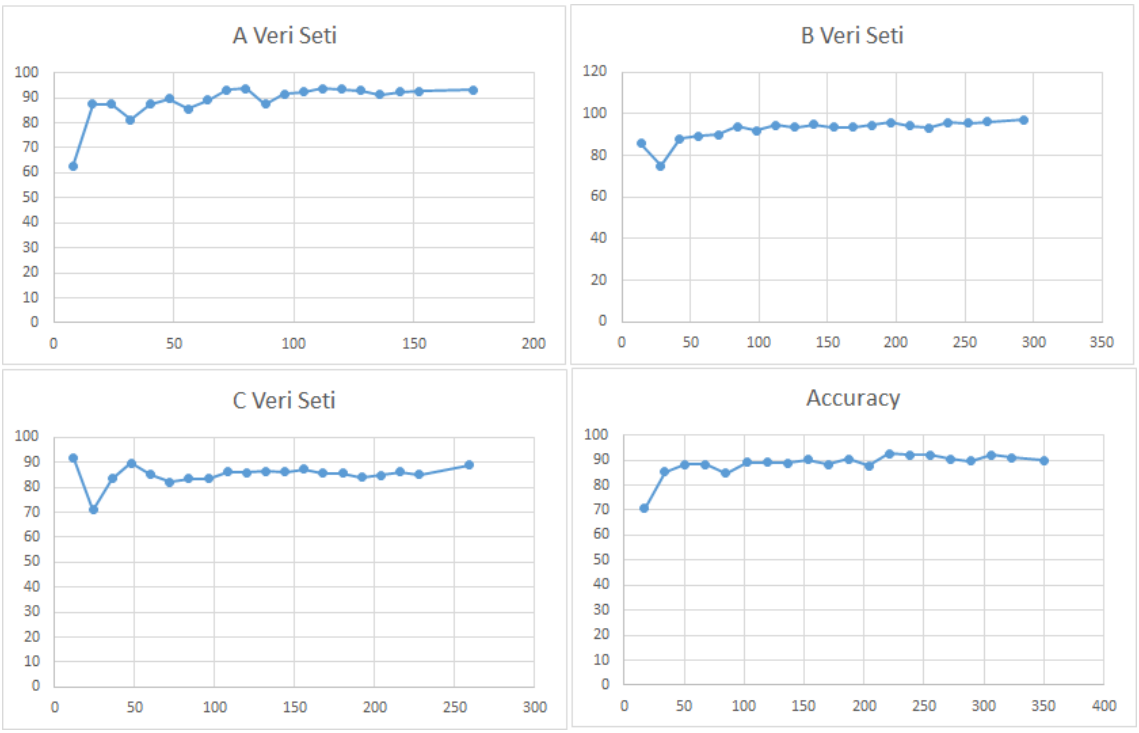
Sınıflandırıcıların eğitimi için kullanılan örnek sayılarının yeterli olup olmadığını yorumlayabilmek için, her bir veri tabanındaki veriler 20 eşit parçaya bölünerek kümülatif olarak artan veri sayısına sahip 20 adet veri dosyası elde edilmiştir. Elde edilen bu veri dosyaları, veri sayısına göre sırasıyla kıyaslamaya konu olan IBk ve J48 algoritmalarında 4-kat çapraz geçerleme (4-fold validation) ile sınanmıştır. Dört kat çapraz geçerleme kullanılmasının nedeni, verilerin 20 eşit parçaya bölünmesinden dolayı geçerleme için kullanılacak veri sayısının 10 kat çapraz geçerleme için yeterli olmamasıdır.

Her bir veri dosyasından elde edilen doğruluk değerleri, veri sayısı ile birlikte grafik olarak çizdirilmiştir. IBk algoritmasının sınama sonuçları Şekil 3.21'de gösterilmiştir. Benzer şekilde, J48 algoritmasının her bir veri dosyası için öğrenme eğrileri ise Şekil 3.22'de gösterilmiştir.

Şekil 3.21 ve Şekil 3.22'de verilen grafikler incelendiğinde, yaklaşık 100 veriden sonra öğrenmenin kararlı hale geldiği görülmektedir. Böylece veri sayılarının, IBk ve J48 algoritmalarının eğitimi için yeterli olduğu sonucuna varılmıştır.




Şekil 3.21 IBk algoritmasının veri setleri için öğrenme eğrileri




Şekil 3.22 J48 algoritmasının veri setleri için öğrenme eğrileri

### 3.7. Gerçek Zamanlı Çalışan Uygulama

Sistem tasarımının son aşaması, gerçek zamanlı çalışan uygulamadır. Bu uygulama, sürücünün canlı video görüntülerini sınıflandırarak uykululuğu tespit eden ve sürücüyü uyaran bir bilgisayar yazılımıdır. Bu yazılımda görüntü işleme teknikleri ve makine öğrenme algoritmaları kullanılmıştır. Uygulama, kişiye özel çalışacak şekilde tasarlanmıştır. Uygulama tasarımı modüler bir yapıda olup üç kısımdan oluşmaktadır:

- Canlı görüntülerden veri dosyası oluşturma modülü
- Veri dosyasından model oluşturma modülü
- Canlı video görüntüsü sınıflandırma ve uykululuk izleme modülü

#### 3.7.1. Canlı Görüntülerden Veri Dosyası Oluşturma Modülü

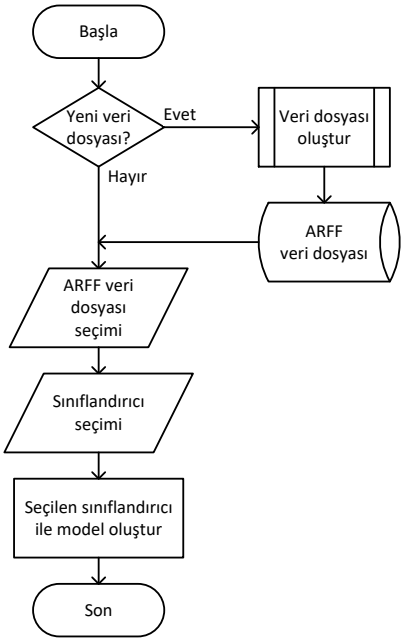
Sistemi ilk kez kullanacak olan bir sürücünün, bir defaya mahsus kendine özel bir veri dosyası hazırlaması gerekmektedir. Bunun için, canlı video görüntülerinden veri dosyası oluşturma modülünü kullanır.

Veri dosyası oluşturma modülünde, önce örnek sayısı seçilmektedir. Örnek sayısı, canlı video görüntüsünden alınan ve göz görüntülerini içeren çerçeve (frame) sayısıdır. Göz görüntülerinin tespit edilemediği çerçeveler ihmal edilmektedir.

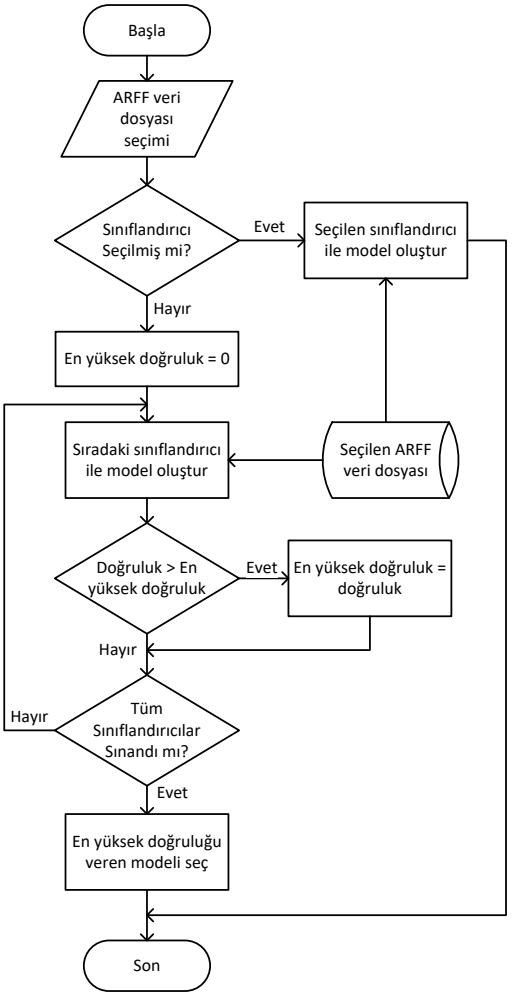
Örnek sayısı belirlendikten sonra sürücü, sırasıyla açık ve kapalı göz örneklerinin toplanmasını sağlar. Bunun için, “Collect Open Eyes” butonuna tıkladıktan sonra, farklı bakış açılarıyla açık gözün değişik formlarındaki örneklerin toplanmasını sağlar. Benzer şekilde “Collect Closed Eyes” butonuna tıkladıktan sonra, kafanın farklı pozisyonları için kapalı gözün değişik açılardaki örneklerinin toplanmasını sağlar.

Bu işlem tamamlandığında veri dosyası, öğrenme modeli oluşturmak için hazır hale gelir. Gerek görülmesi halinde mevcut veri dosyasının üzerine veri eklenmeye devam edilebilir.

Kişiye özel veriler, ARFF (Attribute Relation File Format – Öznitelik İlişki Dosya Formatı) dosyasında saklanmaktadır. Bu dosya daha sonra öğrenme modeli çıkarmak için kullanılmaktadır. Model oluşturmak için gerekli adımlar Şekil 3.23’teki akış diyagramında gösterilmiştir. Akış şemasındaki “Veri dosyası oluştur” alt işlemi, EK-D’de verilmiştir.




Şekil 3.23 Veri dosyası ve model oluşturma akış diyagramı




Şekil 3.24 Otomatik model seçim işleminin akış diyagramı

### 3.7.2. Veri Dosyasından Model Oluşturma Modülü

Sürücü, model oluşturmak amacıyla daha önce kaydettiği veri dosyasını kullanır. Bunun için mevcut bir veri dosyası seçilir. Daha sonra, önceden tanımlanmış makine öğrenme algoritmalarından biri seçilerek öğrenme modeli oluşturulur. Uygulamada ön tanımlı olarak sunulan makine öğrenme algoritmaları Tablo 3.7'de gösterilmiştir.

Öğrenme modeli oluşturmada kullanılan makine öğrenme algoritmalarının parametreleri gerekli görülmesi halinde değiştirilebilmektedir. Eğer hiçbir makine öğrenme algoritması seçilmeden model oluşturulursa, tüm makine öğrenme algoritmaları sınanarak, en yüksek doğruluğu veren algoritma otomatik olarak seçilir. Otomatik model seçimini gerçekleştiren algoritmanın akış diyagramı Şekil 3.24'te gösterilmiştir.

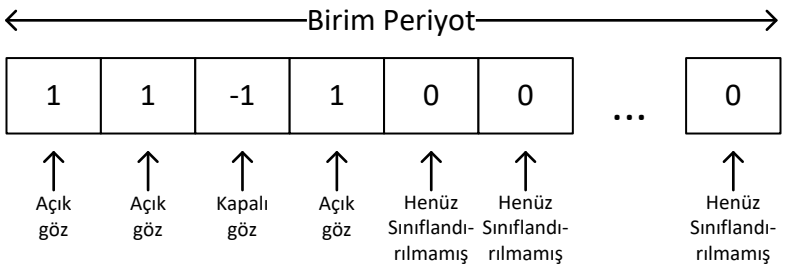
Tablo 3.7 Gerçek zamanlı uygulamada kullanılan ön tanımlı makine öğrenme algoritmaları ve varsayılan parametreleri

| **Algoritma** | **Varsayılan Parametre** |
|---|---|
| Naïve Bayes | |
| | -K |
| | -D |
| IBk | -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A "weka.core.EuclideanDistance -R first-last"" |
| | -K 3 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A "weka.core.EuclideanDistance -R first-last"" |
| | -K 5 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A "weka.core.EuclideanDistance -R first-last"" |
| J48 | -C 0.25 -M 2 |
| | -U -M 2 |
| Voted Perceptron | -I 1 -E 1.0 -S 1 -M 10000 |
| | -I 1 -E 2.0 -S 1 -M 10000 |
| | -I 5 -E 1.0 -S 1 -M 10000 |
| | -I 5 -E 2.0 -S 1 -M 10000 |
| SVM | -S 0 -K 0 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -Z -model "C:\Program Files\Weka-3-8\ -seed 1 |
| | -S 0 -K 1 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -Z -model "C:\Program Files\Weka-3-8\ -seed 1 |
| | -S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -Z -model "C:\Program Files\Weka-3-8\ -seed 1 |
| | -S 0 -K 3 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -Z -model "C:\Program Files\Weka-3-8\ -seed 1 |

### 3.7.3. Canlı Video Görüntüsü Sınıflandırma ve Uykululuk İzleme Modülü

Öğrenme modeli oluşturulduktan sonra sürücü görüntüleri sınıflandırılabilir ve uykululuk durumu izlenebilir. İzlemeyi başlatmak için, bilgisayara bağlı olan kameralardan biri seçilerek “Open” butonuna tıklanır.

Görüntü elde edilmeye başlandığında, eğer öğrenme modeli daha önceden oluşturulmuşsa sınıflandırma işlemi hemen başlar ve sürücünün uykululuk durumu izlenir. Eğer model oluşturulmadan görüntü elde edilmeye başlanırsa, herhangi sınıflandırma işlemi yapılmaz ancak kırpılan gözler ekranda gösterilir.




Şekil 3.25 Göz durumlarının kaydedildiği sabit boyutlu göz durum dizisi

Kameradan alınan görüntüler, görüntü işleme performansını arttırmak için 320x240 piksel uzaysal çözünürlüğe düşürülmüştür. Algoritmalarda renk bilgisi kullanılmadığı için, çözünürlüğü düşürülen görüntüler gri ölçeğe dönüştürülmüştür. Ardından Viola-Jones detektörü ile yüz görüntüsü aranmıştır. Yüz görüntüsü bulunduğunda, yüzün anatomik oranlarına göre gözlerin olduğu bölge ikiye ayrılarak, sağ ve sol göz yine Viola-Jones detektörü ile paralel olarak bulunmuştur. Her iki gözün bulunması durumunda, göz görüntüleri birleştirilerek tek bir görüntü elde edilmiştir. Elde edilen bu görüntü kuyruk belleğine eklenmektedir. Gözlerin bulunarak kırpılmasını sağlayan algoritmanın akış diyagramı EK-F'de, kaynak kodu ise EK-H'de verilmiştir. Görüntü işleme ana döngüsünün akış diyagramı EK-E'de, kaynak kodu EK-I'da verilmiştir.

Görüntü akışı başlatıldığında ilk olarak, sınıflandırma işlemini yapacak olan çekirdek sayısı kadar paralel görev başlatılır. Bu paralel görevler, sürekli olarak kuyruk adı verilen ve ilk giren verinin ilk çıktığı (FIFO – First Input First Output) dinamik belleği kontrol ederler. Kuyrukta herhangi görüntü algılandığında, bunu algılayan ilk paralel görev görüntüyü kuyruktan alır ve daha önce oluşturulan öğrenme modelini kullanarak sınıflandırma yapar. Sınıflandırma sonucunda göz durumlarını saklayan dizide güncelleme yapar. Paralel görevler yarış durumunda (race condition) çalışmaktadır. Uygulama ana döngüsü akış diyagramı EK-C'de, oluşturulan paralel görevlerin akış diyagramları EK-G'de verilmiştir.

Paralel görevlerden elde edilen göz durumu bilgileri, sabit boyutlu bir dizide saklanmaktadır. Açık göz için 1, kapalı göz için -1 değeri bu diziye kaydedilmektedir. Dizinin sıfır değerini içeren elemanları henüz güncellenmemiştir. Her güncelleme sonrasında indis değişkeni bir arttırılmaktadır. Göz durumlarının kaydedildiği sabit boyutlu dizi Şekil 3.25'te gösterilmiştir.

Dizinin boyutu, PERCLOS (Percentage of Closure: gözün kapalılık oranı) değerini hesaplamak için gerekli olan periyot süresini kapsayacak şekilde seçilmiştir ve Denklem 3.11'de gösterildiği şekilde hesaplanmıştır.

$$N = T \cdot FPS \tag{3.11}$$

Bu denklemde *N*, dizideki eleman sayısı, *T*, periyot zamanı (saniye), *FPS (Frame Per Second)* ise saniyede işlenen çerçeve sayısıdır. *FPS* değeri, donanım ve yazılımın performansına bağlıdır ve deneysel olarak hesaplanmaktadır. Periyot süresi *T* ise, PERCLOS hesaplamasında kullanılacak süredir. PERCLOS periyot süresi çalışmalarda 60 saniye olarak seçilmiştir [88, 144].

FPS değeri, uygulamanın hata ayıklama modunda yapılan deneysel çalışmalar sonucunda, 13 FPS olarak ölçülmüştür. Buna göre dizi uzunluğu 60 saniye periyodu için N = 60 x 13, N = 780 olarak hesaplanmıştır.

Her görüntü yakalama çevriminde, göz durumlarını saklayan dizi üzerinde yeni bir PERCLOS değeri hesaplanır. Bu hesaplama Denklem 3.12'de gösterilen formül ile dizide bulunan kapalı göz sayısına göre yapılır.

$$PERCLOS = \frac{1}{N} \cdot \sum_{i=0}^{N} f(i), f(i) = \begin{cases} 1, S_i < 0 \\ 0, S_i \geq 0 \end{cases} \tag{3.12}$$

Burada, göz durumlarının saklandığı dizi *S*, bu dizinin eleman sayısı *N*'dir. Gerçek zamanlı çalışmanın ilk anlarında, eğer dizide halen sıfır değerine sahip güncellenmemiş eleman varsa PERCLOS hesaplaması yapılmamakta ve değeri -1 olarak atanmaktadır.

Gerçek zamanlı uygulamada, paralel görevler ve çerçeve yakalayan metot, ana işlem ile asenkron çalışmaktadır. Uygulamanın arayüzü ana işlem tarafından kontrol edilmektedir. Asenkron görevlerin ara yüz üzerine herhangi bilgi yazdırması, görevlerin birbirini bekletmesinden dolayı performansta önemli düşüşlere neden olmaktadır.

Ancak bu akademik çalışmada, işleyişi gözlemek ve çalışma zamanı değerlerini alabilmek için birçok bilgi ekrana yazdırılmaktadır. Bu ise, saniyede işlenen çerçeve sayısında önemli düşüşlere neden olmaktadır.

Bu sorunu gidermek amacıyla uygulamaya bir seçenek eklenmiştir. Hata ayıklama (debug) seçeneğinde, işleyişle ilgili tüm bilgiler ekrana yazdırılmaktadır ancak performans düşmektedir.

Tablo 3.8 Hata ayıklama ve normal çalışmada elde edilen FPS değerleri

| | **Hata Ayıklama Çalışması** | **Normal Çalışma** |
|---|---|---|
| Yüz görüntüsünün bulunamadığı durumda | 25 FPS | 30 FPS |
| Yüz ve göz görüntülerinin bulunduğu durumda | 13 FPS | 27 FPS |

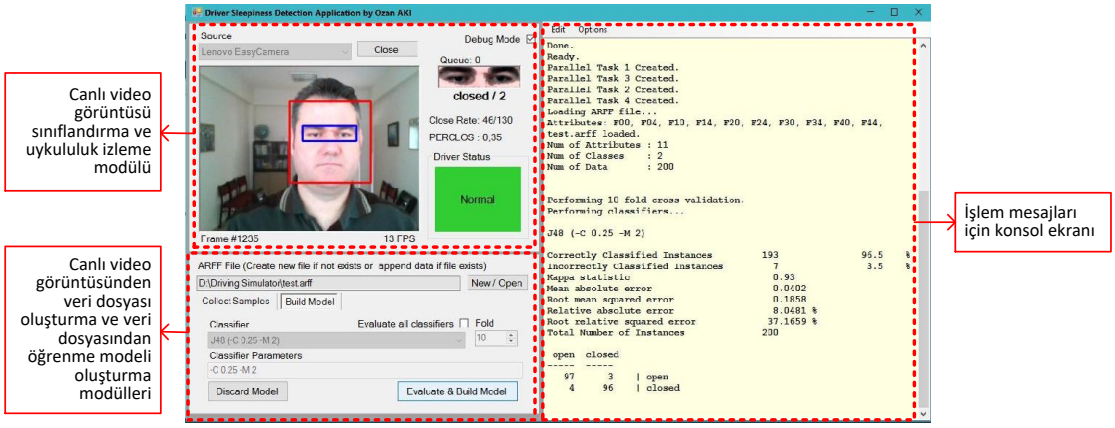
Tablo 3.9 Farklı öznitelik sayıları ile sınıflandırma yapıldığında görüntü işleme hızları

| **Algoritma** | **Öznitelik Sayısına Göre Sınıflandırma ve Görüntü İşleme Hızları (FPS)** | | |
|---|---|---|---|
| | **10 öznitelik** | **20 öznitelik** | **40 öznitelik** |
| Naïve Bayes | 24 | 20 | 14 |
| IBk (3 komşu) | 24 | 21 | 14 |
| J48 (budanmış) | 25 | 21 | 15 |
| Voted Perceptron (1 iterasyon) | 24 | 21 | 14 |
| SVM (doğrusal çekirdekli) | 25 | 22 | 15 |

Normal çalışma seçeneğinde ise, asenkron görevler tarafından kullanıcı ara yüzüne hiçbir şey yazdırılmamaktadır. Böylece, sistemin gerçek çalışma performansı gözlenebilmektedir. Karşılaştırma amacıyla, hata ayıklama ve normal çalışmadaki FPS değerleri, Tablo 3.8'de gösterilmiştir.

Gerçek zamanlı uygulama, Intel® i7 4510U 2,6 GHz işlemci, entegre Intel® HD 4400 grafik işlemci, 8 GB RAM, 1 TB SSD diske sahip taşınabilir bir bilgisayarda test edilmiştir. Yapılan ilk testlerde, herhangi yüz görüntüsü bulunamadığında işlem hızı 27 FPS (Frame Per Second – Saniyedeki Resim Sayısı), yüz ve göz görüntülerinin bulunması ancak sınıflandırma yapılmaması durumunda 13 FPS işlem hızı elde edilmiştir. Göz görüntülerinin elde edilebildiği 13 FPS hız, aynı zamanda 13 Hz örnekleme çözünürlüğü anlamına gelmektedir. 13 Hz > 6 Hz göz kırpmalarını algılayabilme alt sınırından büyük olduğundan dolayı, hata ayıklama modunda görüntü işleme hızının örnekleme için yeterli olduğu sonucuna varılmıştır.

Model oluşturularak sınıflandırma yapılması durumunda performans değerlerinin değerlendirilebilmesi için, 10, 20 ve 40 öznitelik ile farkı sınıflandırıcılarla modeller oluşturularak ölçümler yapılmıştır. Bu ölçümlerin sonuçları Tablo 3.9'da gösterilmiştir.




Şekil 3.26 Gerçek zamanlı çalışan uygulamanın ekran görüntüsü

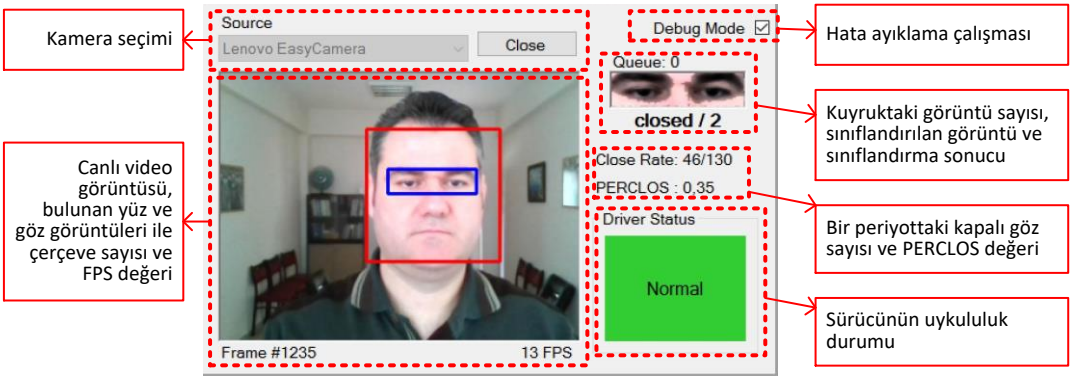
Tablo 3.9'da verilen değerler incelendiğinde, makine öğrenme algoritmalarının sınıflandırma performanslarının, öznitelik sayısına göre değiştiği görülmektedir. Makine öğrenme algoritmalarının türüne bağlı olarak değişkenlik gösteren sınıflandırma sürelerinin, paralel görevler sayesinde ana işlem performansını etkilemeyecek düzeyde standart sürelerle sınırlandırıldığı gözleniyor. Bununla birlikte, öznitelik sayısının sınıflandırma performansında belirleyici olduğu gözlenmiştir. Daha önce analizi yapılarak gerçekleştirilmiş olan öznitelik seçim işlemiyle 13'e düşürülen öznitelik sayısı sayesinde, daha yüksek FPS değerleri elde edilebilmiştir.

### 3.7.4. Gerçek Zamanlı Çalışan Uygulamanın Kullanıcı Ara Yüzü

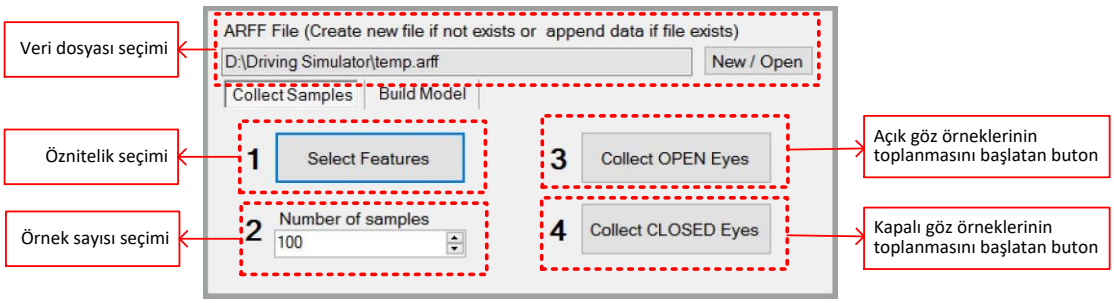
Gerçek zamanlı çalışan uygulamanın örnek arayüz görüntüsü Şekil 3.26'da gösterilmiştir. Geliştirilen modüller, aynı arayüzdeki farklı alanlarda çalışmaktadır.

Canlı video görüntüsü sınıflandırma ve uykululuk izleme modülünde, bilgisayara bağlı kameraların seçilebildiği bir açılır liste ve görüntü almayı başlatan "Open" butonu bulunmaktadır. Görüntü başlatıldığında, eğer hata ayıklama çalışması seçilmişse, elde edilen göz görüntüleri sağdaki küçük pencerede görüntülenmektedir. Eğer model oluşturulmuş ise ayrıca sınıflandırma, kapalı göz sayısı, PERCLOS değeri ve nihai uykululuk durumu da ekranda gösterilmektedir. Canlı video görüntülerini sınıflandıran modülün arayüz öğeleri Şekil 3.27'de gösterilmiştir.

Canlı video görüntülerinden veri dosyalarının elde edildiği modül ekranında, verilerin kaydedileceği dosya seçimi için bir buton, özniteliklerin seçilebileceği bir pencere, örnek sayısının ayarlanabildiği bir giriş kutusu, açık ve kapalı göz örneklerinin toplanmasını sağlayan iki buton bulunmaktadır.




Şekil 3.27 Canlı video görüntülerini sınıflandıran modülün arayüz ekranı öğeleri



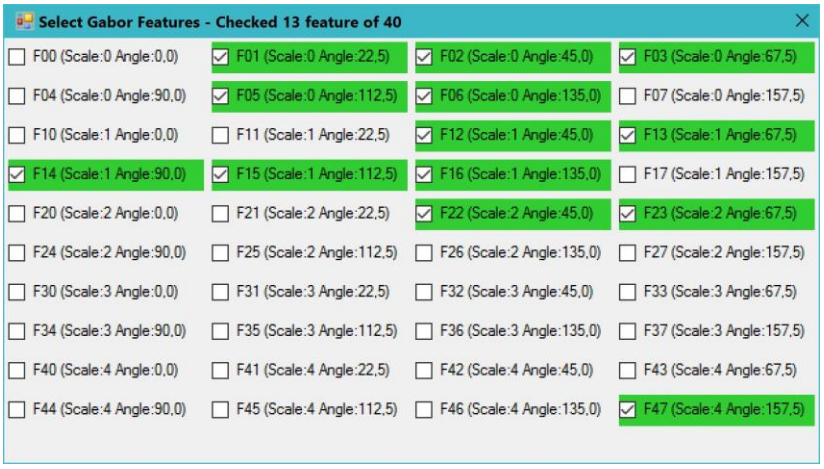

Şekil 3.28 Canlı video görüntüsünden veri dosyası oluşturan modülün arayüz öğeleri

Veri dosyası seçildikten sonra, hesaplanacak öznitelikler ve örnek sayısı belirtilir. Daha sonra sırasıyla açık ve kapalı göz örnekleri toplanır. Canlı video görüntüsünden veri dosyası oluşturan modülün ekran görüntüsü Şekil 3.28'de gösterilmiştir.

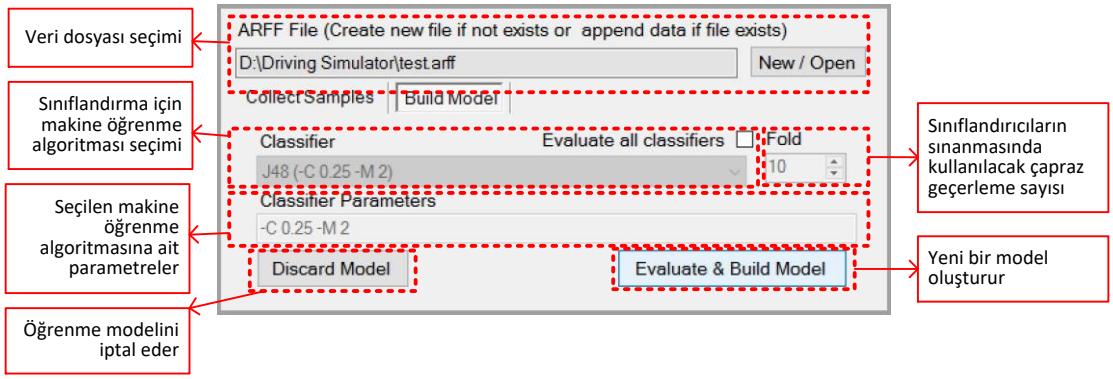
Canlı video görüntülerinden veri dosyası oluşturan modülde ilk olarak veri dosyası seçilir. Dosya belirlendikten sonra öznitelikler seçilir. Öznitelik seçimine tıklandığında, özniteliklerin seçilebileceği bir pencere açılır. Bu pencereden, hesaplanması istenen öznitelikler tıklanarak seçilir ve pencere kapatılır. Bu pencerenin ekran görüntüsü Şekil 3.29'da gösterilmiştir.

Öznitelik seçiminden sonra, toplanacak örnek sayısı belirlenir. Örnek sayısı, her bir sınıf için geçerlidir. Örneğin, 100 örnek sayısı ayarlandıysa 100 adet açık göz ve 100 adet kapalı göz örneği olmak üzere toplam 200 örnek toplanacaktır.

Son aşamada sırasıyla "Collect Open Eyes" ve "Collect Closed Eyes" butonlarına tıklanarak açık ve kapalı göz örneklerinin toplanması sağlanmaktadır. Örnek toplanmasında sürücünün beyanı esas alınmaktadır. Kapalı göz örneklerinin içindeki açık gözler ya da açık göz örneklerindeki kapalı gözler, veri dosyasında gürültü olarak yer alacaktır. Bu nedenle, örnek toplanmasının titizlikle yapılması önem arz etmektedir.




Şekil 3.29 Özniteliklerin seçilmesini sağlayan pencerenin ekran görüntüsü




Şekil 3.30 Veri dosyasından öğrenme modeli oluşturma modülünün arayüz öğeleri

Veri dosyasından öğrenme modeli oluşturma modülünde, ilk olarak önceden oluşturulmuş bir veri dosyası seçilir. Daha sonra ön tanımlı olarak yüklenmiş makine öğrenme algoritmalarından biri seçilerek model oluşturulur. Gerekli görüldüğünde, seçilen makine öğrenme algoritmasının parametrelerinde değişiklik yapılabilir. Eğer hiçbir algoritma seçilmez ise ya da "Evaluate all classifiers" seçeneği seçilmiş ise, yüklenmiş olan sınıflandırıcıların tümü sınanarak, en yüksek doğruluğu veren ilk algoritma otomatik olarak seçilir. Oluşturulan öğrenme modeli, "Discard model" butonu ile iptal edilerek başka bir öğrenme modeli seçilebilir. Veri dosyasından öğrenme modeli oluşturma modülünün ara yüz ekranı öğeleri Şekil 3.30'da gösterilmiştir.

Tablo 3.10 Bu çalışmada kullanılan PERCLOS eşik değerleri ve bu değerlere karşılık gelen uykululuk durumları ile uyarı seviyeleri

| **PERCLOS** | **Açıklama** | **Uyarı Seviyesi** |
|---|---|---|
| 0,0 – 0,15 | Uyanık ve dikkatli | Normal, herhangi uyarı yok |
| 0,15 – 0,30 | Dikkati dağınık | 1. seviyede uyarı, mola tavsiyesi |
| 0,30 ve üzeri | Uykulu | 2. seviyede uyarı, yüksek tonda uyarı |

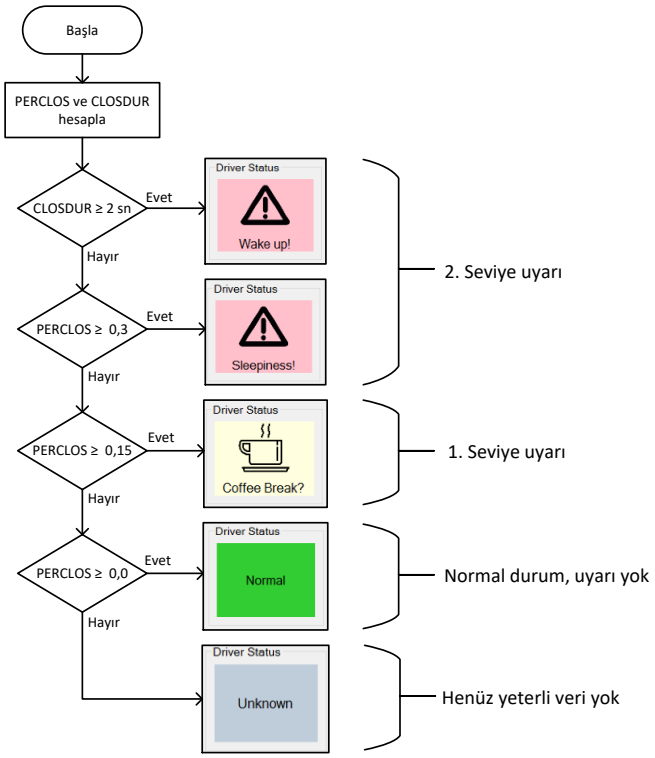
### 3.7.5. Sürücü Uykululuğunun Algılanması ve Sürücünün Uyarılması

Veri dosyası oluşturulup model öğretildikten sonra, uygulama gerçek zamanlı çalışma için hazır hale gelmektedir. Göz durumlarının kaydedildiği dizideki verilere göre hesaplanan PERCLOS değeri, eşik değer ile kıyaslanarak sürücünün uykululuk durumu izlenmektedir.

PERCLOS ölçütü ile sürücü uykululuğunu belirlemede bazı çalışmalar 0,14 değerini eşik olarak seçmişlerdir [88, 145]. Sürücünün dikkatli ya da uykulu olduğunu kademeli olarak belirleyen çalışmada 0,0 - 0,3 aralığı sürücünün dikkatli olduğu, 0,3 - 0,4 aralığında uykulu olduğu belirtilmiştir [34]. Daha detaylı sınıflandırma yapan başka bir çalışmada ise, 0,0 – 0,05 arası uyanık ve dikkatli, 0,05 – 0,125 arası dikkati dağınık ve 0,125 ile 0,3 değerleri uykulu olarak sınıflandırılmıştır [144].

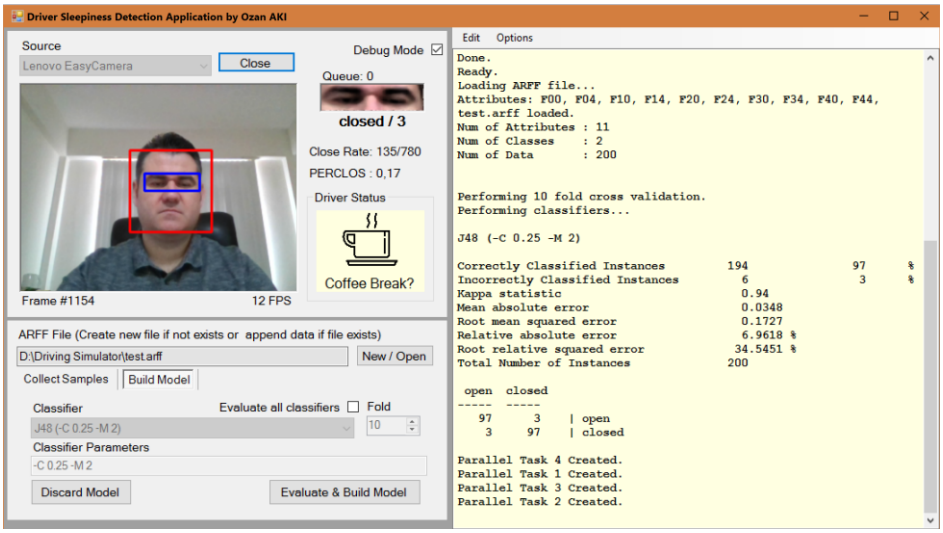
Bu çalışmada kullanılan PERCLOS eşik değerleri, benzer çalışmalardaki değerler de göz önünde bulundurularak deneysel olarak elde edilmiştir. Bu çalışmada kullanılan PERCLOS eşik seviyeleri Tablo 3.10'da gösterildiği gibi seçilmiştir.

Bu çalışmada, iki seviyeli uyarı kullanılmıştır. Birinci seviye için PERCLOS eşik değeri 0,15 olarak seçilmiştir. Bu eşik seviyesinin altında sürücünün dikkatli ve uyanık olduğu değerlendirilir ve herhangi uyarı yapılmaz. PERCLOS değeri 0,15 ile 0,3 arasında ise sürücünün uykululuk belirtileri göstermeye başladığı ve dikkatinin dağıldığına karar verilir ve birinci seviye uyarı yapılarak mola vermesi için sürücüye tavsiye mesajı gösterilir. PERCLOS değeri 0,3 ve üzerinde ise sürücünün uykulu olduğu sonucuna varılarak ikinci seviyede uyarı ile daha güçlü bir ses tonu eşliğinde ünlem işareti ile uyarılır. Ek önlem olarak, birim periyot içerisinde gözlerin ardışık olarak iki saniye boyunca kapalı kaldığı (CLOSDUR $\geq$ 2) tespit edilirse, PERCLOS değerine bakılmaksızın sürücü ikinci seviye uyarı ile uyarılmaktadır. İki seviyeli uyarı sisteminin çalışma akış diyagramı Şekil 3.31'de gösterilmiştir.




Şekil 3.31 İki seviyeli sürücü uyarı sisteminin akış diyagramı

PERCLOS değerlerinin sıfırdan küçük olması, göz durumlarını saklayan dizide henüz yeterli verinin olmadığını ve PERCLOS hesaplamasının henüz yapılmadığını belirtmektedir. Bu durum sadece ilk çalışma anında meydana gelmektedir. Durum dizisi bir defa göz durum verileriyle doldurulduktan sonra bilinmeyen durum artık oluşmamaktadır. Gerçek zamanlı çalışmada birinci seviye uyarı durumunda alınan örnek ekran görüntüsü Şekil 3.32'de gösterilmiştir.




Şekil 3.32 Gerçek zamanlı uygulamada sürücünün 1. seviyede uyarılması

# BÖLÜM 4

# SONUÇLAR VE TARTIŞMA

## 4.1. Sonuçlar

Bu çalışmada, sürücü uykululuğunu görüntü işleme teknikleri ve makine öğrenme algoritmalarıyla gerçek zamanlı olarak tespit ederek sürücünün uyarılmasını sağlayan bir yazılım tasarlanarak geliştirilmiştir.

Görüntü işlemede OpenCV ve Emgu CV görüntü işleme kütüphaneleri kullanılmıştır. Görüntü 320x240 uzaysal çözünürlüğe düşürülerek gri ölçeğe dönüştürülmüştür. Yüz ve göz görüntülerinin tespit edilmesinde AdaBoost sınıflandırıcı tabanlı Viola-Jones detektörü kullanılmıştır. Göz görüntülerinin ışık dağılımı, histogram eşitleme algoritması ile normalleştirilmiştir. Dört ayrı gönüllüden elde edilen video görüntülerinden dört farklı görüntü veri seti elde edilmiştir. Görüntü öznitelikleri, Gabor filtreleri ile çıkarılmıştır. Beş ölçek ve sekiz açıda toplam 40 Gabor filtresi kullanılmış, elde edilen 40 öznitelikten 13 tanesi 0,6 korelasyon eşik değerleri ile seçilmiştir. Bu öznitelikler ile makine öğrenme algoritmaları kıyaslanmış, en yüksek doğruluğu %94,76 ile IBk algoritmasının verdiği, ancak en hızlı algoritmanın ortalama 3 milisaniye sınıflandırma süresi ile J48 olduğu gözlenmiştir. IBk algoritma doğruluğu, J48 algoritmasından %2,78 daha yüksek olmasına rağmen, J48 algoritmasının 100 kat daha hızlı sınıflandırma yaptığı ölçülmüştür. J48 algoritmasının gerçek zamanlı uygulama için en uygun algoritma olduğu sonucuna varılarak, gerçek zamanlı çalışmada bu algoritma önerilmiştir. Elde edilen bulgular neticesinde geliştirilen gerçek zamanlı uygulama ile sürücü görüntüsünün sınıflandırılarak uykululuğun tespit edilmesi ve sürücünün uyarılması sağlanmıştır.

Gerçek zamanlı çalışma için geliştirilen uygulamanın, bazı kısıtlamaları olmasına rağmen üstün yanları fazladır. Kişiye herhangi bir müdahale olmadan çalışması, makine

öğrenmesi algoritmalarının esnek entegrasyonu, yüksek doğruluk oranı, paralel programlama teknikleri sayesinde işlemci çekirdeklerini kullanarak yüksek hızda gerçek zamanlı çalışabilmesi bu uygulamanın üstün yanları olarak sayılabilir.

Geliştirilen yazılım, deneysel sonuçların alınması amacıyla birçok ekran çıktısı göstermesine karşın, kullanımı oldukça kolay ve teknik bilgi gerektirmemektedir. Sürücü, oldukça kısa bir öğrenme süreci içerisinde bir defaya mahsus olarak açık ve kapalı göz durumlarını sisteme öğretmekte, sonraki çalışmalarda oluşturulan bu veri dosyası kullanılmaktadır.

Veri dosyası bir kez oluşturulduktan sonra, bir butona tıklayarak öğrenme modeli oluşturulabilmektedir. Makine öğrenme algoritmalarının tümü sınanarak, en yüksek doğruluğu veren algoritma otomatik olarak seçilmektedir.

Bununla birlikte, model oluşturmada kullanılacak makine öğrenme algoritmalarından herhangi biri, kullanıcının tercihine bağlı olarak seçilebilmektedir. Her bir makine öğrenme algoritmasının parametreleri kullanıcının düzenlemesine açıktır. Böylece uygulamada, gerçek zamanlı çalışma sırasında makine öğrenme algoritmalarının sınanmasında büyük esneklik ve kolaylık sağlanmıştır.

Görüntü işleme tekniklerinde, 30 FPS canlı görüntü akışı içerisinden Viola-Jones detektörü ile yüz görüntüsünün bulunarak kırpılması ve göz görüntülerinin elde edilmesi işlemi 27 FPS gibi yüksek bir hızda gerçekleştirilmiştir. Bu hızın elde edilmesinde, Viola-Jones detektörünün performansının yanında, görüntülerin kırpılarak sağ ve sol gözün paralel bir şekilde bulunmasının önemli katkıları olmuştur.

Görüntü işlemeye ek olarak, model oluşturulup görüntülerin sınıflandırılması işlemi eklendiğinde, öznitelik sayısına göre, 10 öznitelikte 24 FPS, 20 öznitelikte 21 FPS ve 40 öznitelikte 14 FPS hızları elde edilmiştir. Korelasyon değerlerine göre yapılan seçim işlemi sonucunda 13 öznitelik kullanılmıştır. Bu öznitelikler ile sınıflandırma hızı yaklaşık 24 FPS olarak ölçülmüştür.

Gerçek zamanlı olarak nitelendirilen benzer çalışmalar ile kıyaslandığında, elde edilen görüntü işleme hızının, kabul edilebilir sınırların oldukça üzerinde olduğu gözlenmiştir. Gerçek zamanlı çalışan benzer çalışmalardaki görüntü işleme hızları ile bu tez çalışmasındaki ölçümler, saniyede işlenen görüntü sayısı (FPS) cinsinden Tablo 4.1'de gösterilmiştir. Tabloda gösterilen çalışmalar, FPS hızına göre büyükten küçüğe doğru sıralanmıştır.

Tablo 4.1 Gerçek zamanlı çalışma hızının benzer çalışmalarla kıyaslanması

| Kaynak | Görüntü İşleme Yöntemi | Makine Öğrenmesi Algoritmaları | Ölçülen Büyüklük | FPS |
|---|---|---|---|---|
| **(*)** | **Gabor Filtreleri** | **Naïve Bayes, IBk, J48, SVM, Voted Perceptron** | **PERCLOS CLOSDUR** | **24**** |
| [146] | Gabor filtreleri | Temel bileşen analizi (PCA), Gizli Markov modelleri (HMM) | Esneme | **19** |
| [147] | Viola Jones detektörü, Belirgin noktalar analizi (SPA) | - | PERCLOS BLINK Esneme | **16,5** |
| [148] | Yerel ikili desenler (LBP) | Destek vektör makinaları (SVM) | PERCLOS | **12** |
| [111] | Ters perspektif eşleştirme (IPM) | - | PERCLOS BLINKFREQ Yol şeritleri | **6** |
| [149] | Temel bileşen analizi (PCA) | Temel bileşen analizi (PCA) | PERCLOS | **6** |
| [150] | Viola Jones detektörü, Blok yerel ikili desenler (LBP) histogram özellik çıkarımı | İstatistiksel öğrenme yöntemi (PLS), Destek vektör makinaları (SVM) | PERCLOS | **3** |

(*) Bu tez çalışmasında geliştirilen gerçek zamanlı uygulama.

(**) FPS değeri, ekran çıktılarının gösterilmediği, normal çalışma durumunda ölçülmüştür.

Tablo 4.1'de gösterilen çalışmalar ile kıyaslandığında, bu çalışmada elde edilen 24 FPS sınıflandırma hızının, görüntü işleme algoritmaları kullanan benzer çalışmalarda rapor edilen hızlardan daha yüksek olduğu tespit görülmüştür.

Bu çalışmada geliştirilen uygulamanın, gerçek zamanlı kabul edilen çalışma hızı standartlarının üzerinde bir hıza sahip olduğu sonucuna varılmıştır.

Sınıflandırma performansı değerlendirildiğinde, makine öğrenme algoritmalarının kıyaslama sonuçlarını doğrulayan veriler elde edilmiştir. Test için kullanılan video görüntüsünden toplam 3477 görüntü çerçevesi elde edilmiştir. Bu çerçevelerdeki sınıflandırmaların doğruluğu gözle kontrol edilerek Tablo 4.2'de gösterilen karmaşıklık matrisi elde edilmiştir. Bu karmaşıklık matrisine göre, gerçek zamanlı çalışmada elde edilen başarım ölçütleri hesaplanmıştır. Bu hesaplamaların sonuçları Tablo 4.3'te gösterilmiştir.

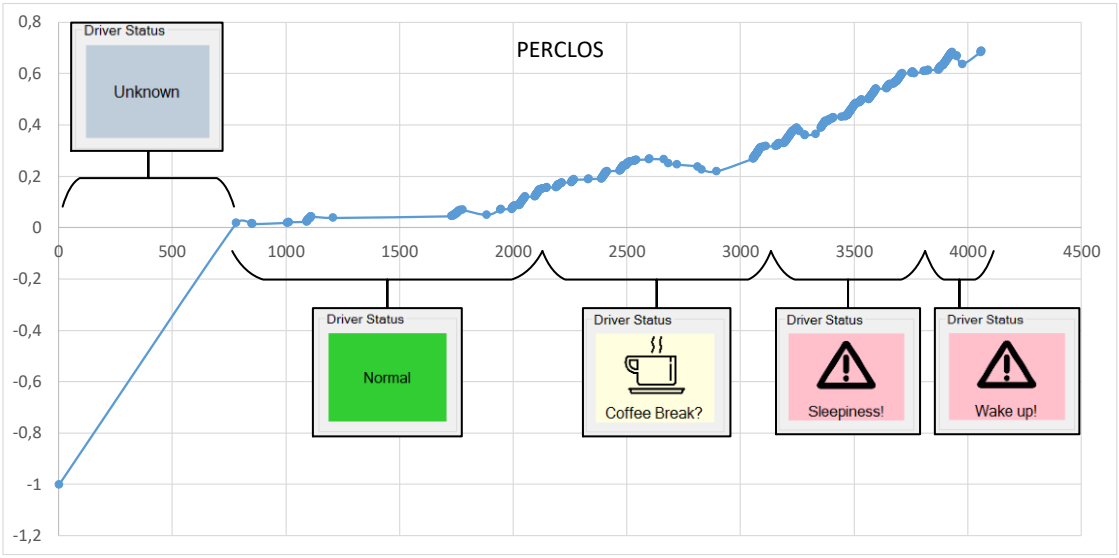
Tablo 4.2 Gerçek zamanlı test sınıflandırmasının karmaşıklık matrisi

| | | **Sınıflandırılan Gözler** | |
|---|---|---|---|
| | | **Açık** | **Kapalı** |
| **Gerçek durum** | **Açık** | 2271 (DA) | 279 (YE) |
| | **Kapalı** | 60 (YA) | 867 (DE) |

Tablo 4.3 Gerçek zamanlı test sınıflandırmasının başarım ölçütleri

| **Ölçüt** | **Formül** | **Hesaplama** | **Sonuç** |
|---|---|---|---|
| Başarı (Accuracy) | $Başarı = \frac{DA + DE}{DA + DE + YA + YE}$ | $\frac{2271 + 867}{2271 + 867 + 60 + 279}$ | 0,90 |
| DA oranı (TP rate) | $DA\ oranı = \frac{DA}{DA + YE}$ | $\frac{2271}{2271 + 279}$ | 0,89 |
| YA oranı (FP rate) | $YA\ oranı = \frac{YA}{YA + DE}$ | $\frac{60}{60 + 867}$ | 0,06 |
| Kesinlik (Precision) | $Kesinlik = \frac{DA}{DA + YA}$ | $\frac{2271}{2271 + 60}$ | 0,97 |
| Anma (Recall) | $Anma = \frac{DA}{DA + YE}$ | $\frac{2271}{2271 + 279}$ | 0,89 |
| Duyarlılık (Sensitivity) | $Duyarlılık = \frac{DA}{DA + YE}$ | $\frac{2271}{2271 + 279}$ | 0,89 |
| Özgüllük (Specificity) | $Özgüllük = \frac{DE}{YA + DE}$ | $\frac{867}{60 + 867}$ | 0,94 |
| F Ölçütü (F-Measure) | $F = 2 \cdot \frac{Duyarlılık \cdot Kesinlik}{Duyarlılık + Kesinlik}$ | $2 \cdot \frac{0{,}89 \cdot 0{,}97}{0{,}89 + 0{,}97}$ | 0,93 |

Gerçek zamanlı sınıflandırma başarım ölçütlerine göre, başarım oranı %90 gibi yüksek bir değer bulunmuştur. Ancak eksi verileri oluşturan kapalı göz sayısı, günlük hayat içerisinde açık gözün yanında oldukça düşük bir oran olduğundan, elde edilen verilerde sınıflar dengesiz dağılımlıdır. Bundan dolayı diğer başarım ölçütleri de hesaplanmıştır. Kesinlik ve anma ölçütlerinin harmonik ortalamasını veren F-Ölçütü incelendiğinde, 0,93 gibi yüksek bir değer verdiği gözlenmiştir.




Şekil 4.1 Gerçek zamanlı video testinde artan PERCLOS değerine karşılık sürücüye verilen uyarı mesajları. Yatay eksen çerçeve sayısını, düşey eksen PERCLOS değerini göstermektedir.

Sonuç olarak, gerçek zamanlı uygulamanın sınıflandırma performansının, sürücü gözlerini ve dolayısıyla uykululuğun tespiti için yeterli olduğu sonucuna varılmıştır.

Gerçek zamanlı uygulamanın uyarı sisteminin test edilmesi için, farklı uykululuk seviyelerinin bulunduğu bir video kullanılmıştır. Video görüntüsünün test sonuçları Şekil 4.1'de gösterilmiştir. Bu videoda, göz bilgilerinin saklandığı dizideki verilerin tamamlanması için geçen ilk bir dakikada süre boyunca sürücü durumu bilinmemektedir.

PERCLOS değerinin 0,15 eşik seviyesini aşmadığı 2113. çerçeveye kadar olan sürede sürücünün uykululuk durumu normal olarak belirtilmiş ve herhangi uyarı yapılmamıştır. 2113 ile 3076. Çerçeveler arasında PERCLOS değeri 0,15'in üzerine çıkmış ve sürücüye birinci seviye uyarı ile mola vermesini tavsiye eden bir mesaj gösterilmiştir. 3076. Çerçeveden sonra uykululuk derecesinin artmasına bağlı olarak PERCLOS değeri 0,3 eşik seviyesi aşmış ve sürücü ikinci seviye uyarı ile uyarılmıştır.

Uykululuk seviyesinin artmasıyla beraber, gözlerin kapalı kalma süresinin de arttığı gözlenmiştir. 3901. Çerçeveden itibaren gözlerin iki saniye boyunca kapalı kaldığı tespit edilmiş ve ikinci seviye uyarı ile tekrar uyarılmıştır.

Gerçekleştirilen bu test sonucunda geliştirilen uygulamanın, tez çalışmasının amacı doğrultusunda sürücü uykululuk durumunu gerçek zamanlı olarak başarıyla izlediği ve tanımlanan eşik seviyelerine ulaşıldığında sürücüyü zamanında uyardığı tespit edilmiştir.

## 4.2. Tartışma

Bu çalışmada geliştirilen uygulamada bazı kısıtlamalar ile karşılaşılmıştır. Bunların en önemlisi, doğal (şartlandırılmamış) ortamdan alınan görüntülerin işlenmesi ile ilgilidir. Doğal ortamdan elde edilen görüntülerde, nesnenin konumlandırması ve aydınlatma kontrolümüz dışında olduğundan bu parametrelerdeki değişkenlikler görüntü kalitesini ve içeriğini etkilemektedir. Görüntülerin kontrolsüz bir şekilde değişkenlik göstermesi, görüntüden elde edilen öznitelikleri etkilemekte ve gürültülü verilerin oluşmasına neden olmaktadır. Bu gürültüler sınıflandırma doğruluğunu olumsuz etkilemektedir. Sonuç olarak, görüntü kalitesi ile sınıflandırma başarımı arasında doğrudan bir ilişki söz konusudur.

Bu nedenle, görüntü işleme ile ilgili çalışmalarda elde edilen başarım ölçütü değeri, sadece bu değerin elde edilmesinde kullanılan görüntüler için geçerlidir. Çalışmalarda kullanılan görüntü işleme tekniklerinin başarım değerlerinin, farklı görüntü veri tabanlarında aynı sonucu vereceği şeklinde genelleme yapmak mümkün değildir.

Bu anlamda, görüntü işleme teknikleri ile benzer işlemleri gerçekleştiren farklı çalışmaların başarım değerlerinin birbiriyle kıyaslanması her ne kadar fikir verse de tutarlı olmayabilir. Farklı tekniklerin başarım değerleri, ancak ve ancak aynı görüntü veri tabanı üzerinde sınanarak kıyaslanabilir.

Sonuç bölümünde gerçekleştirilen başarım kıyaslamaları, bu anlamda sadece fikir verme amacını gütmektedir. Buradan, görüntü işleme tekniklerinin birbirinden daha iyi ya da daha kötü olduğunu sonucu çıkarılamaz.

Görüntü işlemede karşılaşılan diğer bir kısıt işlem hızı ile ilgilidir. Görüntü, büyük miktarda bilgi içeren iki boyutlu veri modelidir. Görüntüler üzerinde işlem yapmak, yüksek işlemci gücü ve bellek gerektirir. Görüntü üzerinde yapılan işlem sayısı arttıkça, saniyede işlenen görüntü sayısı düşmektedir. Günümüzde hemen hemen tüm işlemcilerin çok çekirdekli olduğu düşünüldüğünde, performans kayıplarının çözümü olarak paralel programlama teknikleri kullanılabilir. Eğer çerçeveler arasında bir hesaplama ilişkisi yoksa birden fazla resim çerçevesi farklı çekirdekler üzerinde eş zamanlı olarak işlenebilir. Ek olarak, eğer donanımda mevcut ise grafik işlemcisi GPU (Graphical Processing Unit) üzerinde işlem yaptırılabilir.

### 4.3. Gelecek Çalışmalar ve Öneriler

Bu çalışmada geliştirilen yazılım, deneysel amaçlar doğrultusunda kişisel bir bilgisayar üzerinde çalıştırılmıştır. Ancak yazılım, gömülü sistemlerde çalışacak şekilde değiştirilerek optimize edilmesiyle çok daha küçük boyutlarda ve düşük maliyetle uygulanabilir.

Bununla birlikte, söz konusu olan insan ve uykululuk olduğunda, bu konuyu sadece araç sürücüleri ile sınırlandırmak yanlış olacaktır. İnsan faktörünün olduğu ve dikkat gerektiren çeşitli iş kollarında görüntü işlemeye dayalı uykululuk tespit sistemleri kullanılabilir. Üstelik davranışsal ya da fizyolojik büyüklüklerin doğrudan ölçülmesinin mümkün olmadığı durumlarda uykululuğun tespiti için görüntü işleme tekniklerinin kullanılması zaruri olabilir.

Örneğin, insan hayatı için tehlikeli olabilecek bir makineyi kullanan operatörün, uykulu ya da yorgun olup olmadığı makine üzerine yerleştirilen bir kamera ile tespit edilebilir ve makinenin çalışması buna göre engellenebilir. Bu makine, bir iş makinesi olabileceği gibi, fabrika içerisindeki sabit bir makine de olabilir. Böyle bir uykululuk izleme sistemin makine üzerine eklenmesi, getireceği fayda ile kıyaslandığında makinenin maliyetinin yanında oldukça küçük kalacaktır.

Günümüzde araç üreticilerinin modellerinde kullandığı uykululuk izleme sistemlerinde, maliyet ya da görüntü işlemenin zorluğu gibi nedenlerle her ne kadar görüntü işleme yöntemi yerine araç üstü sensörlerin kullanılması tercih edilmiş olsa da, hızla düşen maliyetler ve araçların teknoloji ile yüksek entegrasyonu gösteriyor ki, gelecekte sürücülerin uykululuk durumunun da takip edileceği, sürücünün baktığı yönlere göre çeşitli görevleri gerçekleştiren, dudak okuyarak komut alan, sürücünün yüz ifadesinden ruh halini algılayarak buna göre otomatik müzik seçen sistemler araçlarda yerini alacaklardır.

# EKLER

## 5.1. EK-A Özniteliklerin Veri Setlerine Göre Korelasyon Değerleri

Her bir özniteliğin veri setlerine göre korelasyonlarını gösteren tablodur.

| Öznitelik | A Veri Seti | B Veri Seti | C Veri Seti | D Veri Seti |
|---|---|---|---|---|
| F00 | 0,622 | 0,0773 | 0,1211 | 0,709 |
| F01 | 0,682 | 0,5737 | 0,5384 | 0,658 |
| F02 | 0,772 | 0,6879 | 0,5623 | 0,645 |
| F03 | 0,773 | 0,6789 | 0,6187 | 0,618 |
| F04 | 0,653 | 0,6199 | 0,5275 | 0,539 |
| F05 | 0,581 | 0,7331 | 0,5142 | 0,597 |
| F06 | 0,662 | 0,6669 | 0,5438 | 0,596 |
| F07 | 0,73 | 0,4697 | 0,4619 | 0,576 |
| F10 | 0,513 | 0,1497 | 0,2323 | 0,71 |
| F11 | 0,646 | 0,394 | 0,5698 | 0,776 |
| F12 | 0,756 | 0,7026 | 0,6337 | 0,695 |
| F13 | 0,74 | 0,7716 | 0,6955 | 0,581 |
| F14 | 0,641 | 0,6513 | 0,5981 | 0,516 |
| F15 | 0,581 | 0,7817 | 0,6185 | 0,492 |
| F16 | 0,677 | 0,7047 | 0,5659 | 0,604 |
| F17 | 0,532 | 0,5121 | 0,4616 | 0,657 |
| F20 | 0,139 | 0,1753 | 0,2941 | 0,689 |
| F21 | 0,289 | 0,4275 | 0,3932 | 0,723 |
| F22 | 0,796 | 0,6765 | 0,6259 | 0,436 |
| F23 | 0,721 | 0,7273 | 0,6547 | 0,578 |
| F24 | 0,543 | 0,663 | 0,5325 | 0,363 |
| F25 | 0,62 | 0,79 | 0,6259 | 0,338 |
| F26 | 0,561 | 0,605 | 0,5139 | 0,633 |
| F27 | 0,262 | 0,3277 | 0,4634 | 0,753 |
| F30 | 0,14 | 0,1673 | 0,1264 | 0,716 |
| F31 | 0,106 | 0,2029 | 0,0869 | 0,717 |
| F32 | 0,415 | 0,6653 | 0,5864 | 0,482 |
| F33 | 0,566 | 0,6467 | 0,5692 | 0,45 |
| F34 | 0,547 | 0,4593 | 0,4523 | 0,137 |
| F35 | 0,479 | 0,6784 | 0,6316 | 0,218 |
| F36 | 0,394 | 0,6163 | 0,5945 | 0,587 |
| F37 | 0,266 | 0,2873 | 0,3825 | 0,708 |
| F40 | 0,613 | 0,4831 | 0,147 | 0,57 |
| F41 | 0,325 | 0,441 | 0,3161 | 0,744 |
| F42 | 0,367 | 0,108 | 0,4737 | 0,684 |
| F43 | 0,459 | 0,3401 | 0,5287 | 0,304 |
| F44 | 0,299 | 0,2642 | 0,4172 | 0,293 |
| F45 | 0,446 | 0,3629 | 0,4373 | 0,249 |
| F46 | 0,579 | 0,1042 | 0,5918 | 0,6719 |
| F47 | 0,507 | 0,6539 | 0,5737 | 0,679 |

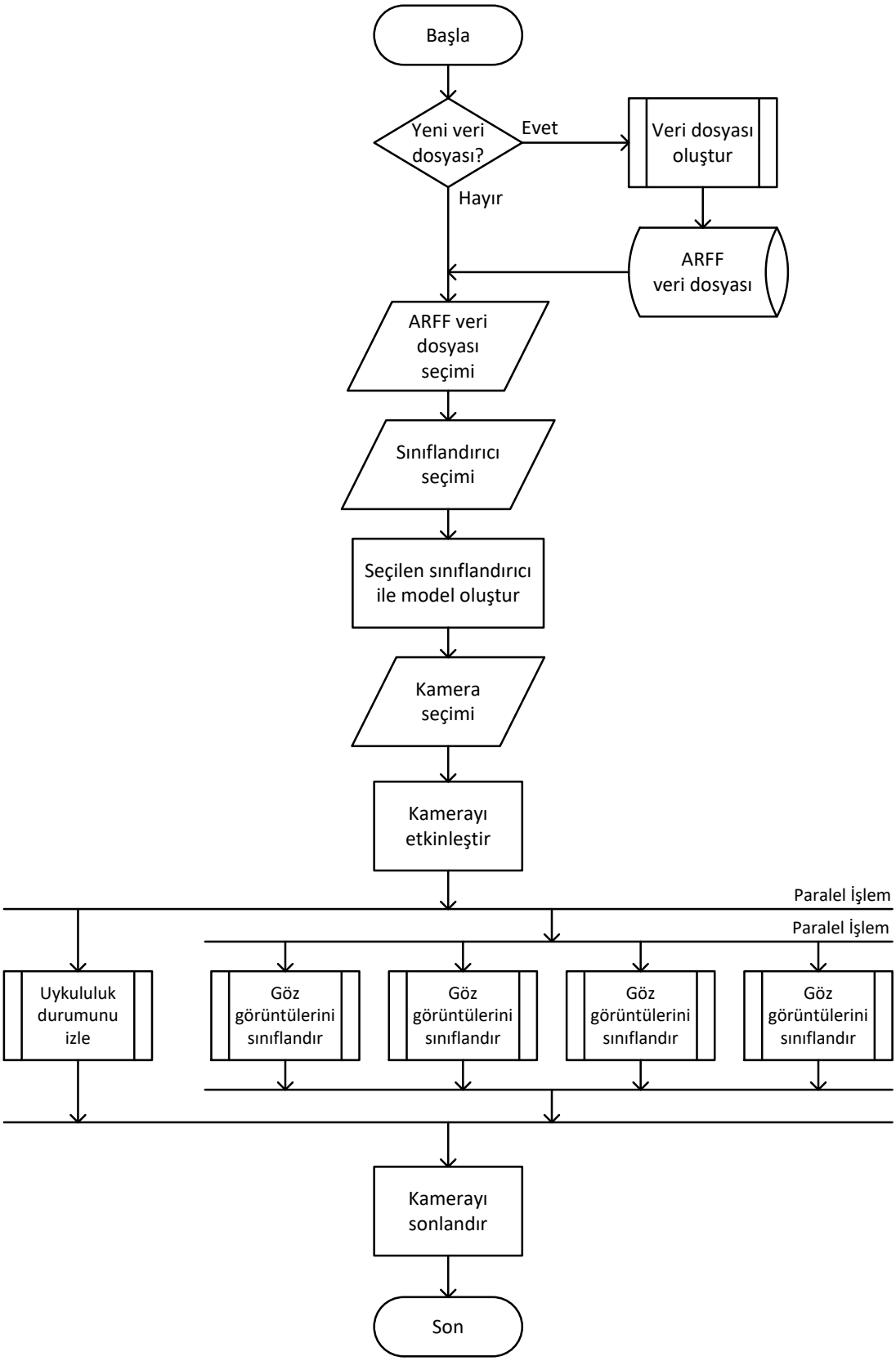
## 5.2. EK-B Sınıflandırıcıların Veri Setlerine Göre Başarım Değerleri

Veri setleri ile sınıflandırıcıların parametrik değişimleri ile birlikte 10 kat çapraz doğrulamalı (10-fold validation) hesaplama ile elde edilen ölçüt sonuçlarını gösteren tablodur. Koyu yazılı değerler, ilgili alt bölümlerin aritmetik ortalama değerleridir.

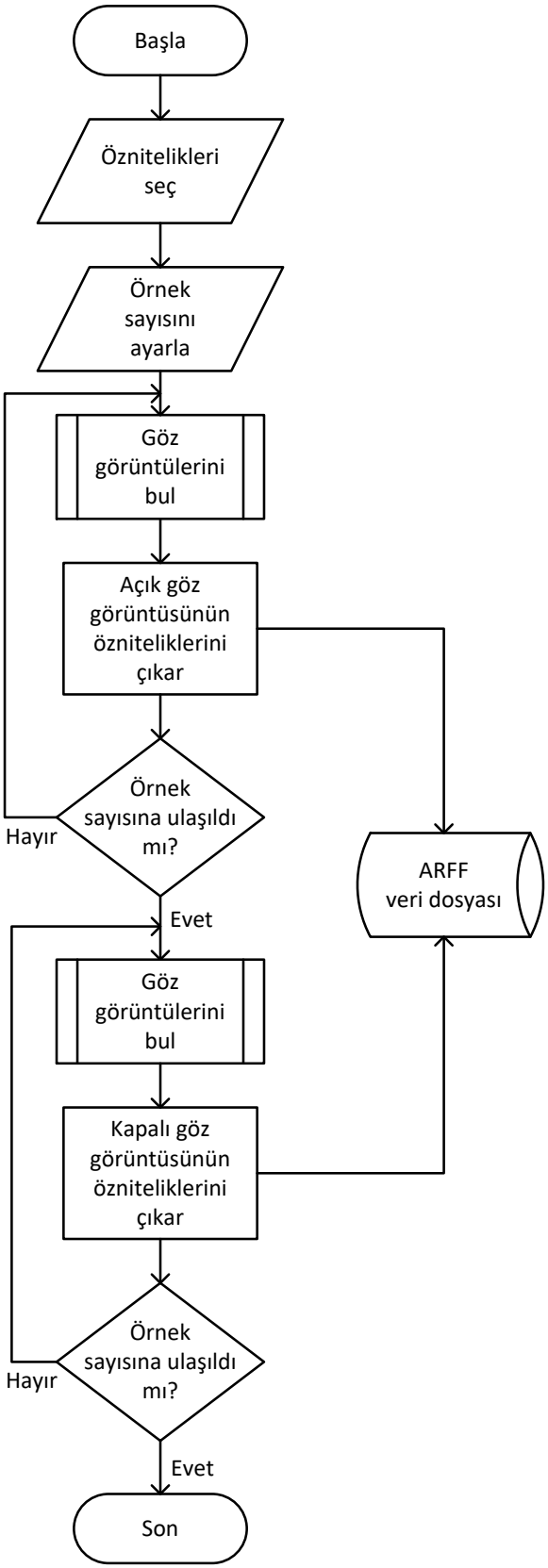
| Veri Setleri ve Makine Öğrenme Algoritmaları | Doğruluk (%) (Accuracy) | Kappa | F Ölçütü (F Measure) | AİK Alanı (ROC Area) | Doğru Artı oranı (TP rate) | Yanlış Artı oranı (FP rate) | Doğru Eksi oranı (TN rate) | Yanlış Eksi oranı (FN rate) | Öğrenme süresi (Train Time) | Sınıflandırma süresi (Test time) |
|---|---|---|---|---|---|---|---|---|---|---|
| **A-FS** | **85,574** | **0,701** | **0,879** | **0,864** | **0,931** | **0,234** | **0,766** | **0,069** | **349,496** | **0,542** |
| **Naïve Bayes** | **96,016** | **0,919** | **0,965** | **0,983** | **0,979** | **0,063** | **0,938** | **0,021** | **0,097** | **0,036** |
| P1 | 96,016 | 0,919 | 0,965 | 0,983 | 0,979 | 0,063 | 0,938 | 0,021 | 0,097 | 0,036 |
| **IBk** | **94,635** | **0,890** | **0,954** | **0,960** | **0,987** | **0,102** | **0,898** | **0,013** | **0,035** | **0,295** |
| P1 | 93,915 | 0,876 | 0,945 | 0,937 | 0,960 | 0,086 | 0,914 | 0,040 | 0,018 | 0,103 |
| P2 | 94,882 | 0,895 | 0,957 | 0,968 | 1,000 | 0,113 | 0,888 | 0,000 | 0,011 | 0,091 |
| P3 | 95,108 | 0,900 | 0,959 | 0,976 | 1,000 | 0,108 | 0,893 | 0,000 | 0,006 | 0,101 |
| **J48** | **92,013** | **0,838** | **0,927** | **0,929** | **0,941** | **0,105** | **0,895** | **0,059** | **0,537** | **0,004** |
| P1 | 92,013 | 0,838 | 0,927 | 0,929 | 0,941 | 0,105 | 0,895 | 0,059 | 0,316 | 0,003 |
| P2 | 92,013 | 0,838 | 0,927 | 0,929 | 0,941 | 0,105 | 0,895 | 0,059 | 0,221 | 0,001 |
| **SVM** | **67,894** | **0,330** | **0,737** | **0,664** | **0,840** | **0,513** | **0,488** | **0,160** | **348,507** | **0,180** |
| P1 | 64,846 | 0,296 | 0,659 | 0,646 | 0,669 | 0,376 | 0,624 | 0,331 | 183,102 | 0,010 |
| P2 | 83,565 | 0,669 | 0,844 | 0,834 | 0,852 | 0,184 | 0,816 | 0,148 | 163,962 | 0,010 |
| P3 | 55,271 | 0,024 | 0,708 | 0,511 | 1,000 | 0,978 | 0,023 | 0,000 | 1,443 | 0,160 |
| **Voted Perceptron** | **86,418** | **0,721** | **0,886** | **0,883** | **0,945** | **0,232** | **0,768** | **0,055** | **0,320** | **0,027** |
| P1 | 84,529 | 0,680 | 0,874 | 0,890 | 0,954 | 0,284 | 0,716 | 0,046 | 0,086 | 0,007 |
| P2 | 87,157 | 0,736 | 0,891 | 0,880 | 0,944 | 0,215 | 0,785 | 0,056 | 0,087 | 0,006 |
| P3 | 87,569 | 0,746 | 0,893 | 0,879 | 0,936 | 0,196 | 0,804 | 0,064 | 0,147 | 0,014 |
| **B-FS** | **92,388** | **0,809** | **0,845** | **0,913** | **0,854** | **0,042** | **0,958** | **0,146** | **304,666** | **0,983** |
| **Naïve Bayes** | **96,585** | **0,924** | **0,949** | **0,994** | **0,959** | **0,031** | **0,969** | **0,041** | **0,069** | **0,057** |
| P1 | 96,585 | 0,924 | 0,949 | 0,994 | 0,959 | 0,031 | 0,969 | 0,041 | 0,069 | 0,057 |
| **IBk** | **97,189** | **0,938** | **0,959** | **0,984** | **0,975** | **0,029** | **0,971** | **0,025** | **0,043** | **0,513** |
| P1 | 96,963 | 0,933 | 0,956 | 0,973 | 0,981 | 0,036 | 0,964 | 0,019 | 0,016 | 0,124 |
| P2 | 97,028 | 0,934 | 0,956 | 0,990 | 0,969 | 0,029 | 0,971 | 0,031 | 0,018 | 0,190 |
| P3 | 97,575 | 0,946 | 0,964 | 0,990 | 0,974 | 0,023 | 0,977 | 0,026 | 0,009 | 0,199 |
| **J48** | **95,901** | **0,908** | **0,938** | **0,954** | **0,937** | **0,030** | **0,970** | **0,063** | **0,624** | **0,006** |
| P1 | 95,901 | 0,908 | 0,938 | 0,954 | 0,937 | 0,030 | 0,970 | 0,063 | 0,359 | 0,004 |
| P2 | 95,901 | 0,908 | 0,938 | 0,954 | 0,937 | 0,030 | 0,970 | 0,063 | 0,265 | 0,002 |
| **SVM** | **82,738** | **0,531** | **0,578** | **0,766** | **0,585** | **0,053** | **0,947** | **0,415** | **303,477** | **0,353** |
| P1 | 85,080 | 0,680 | 0,792 | 0,839 | 0,804 | 0,127 | 0,873 | 0,196 | 251,700 | 0,013 |
| P2 | 96,238 | 0,914 | 0,942 | 0,959 | 0,950 | 0,032 | 0,968 | 0,050 | 48,643 | 0,008 |
| P3 | 66,897 | 0,000 | 0,000 | 0,500 | 0,000 | 0,000 | 1,000 | 1,000 | 3,134 | 0,332 |
| **Voted Perceptron** | **93,495** | **0,854** | **0,903** | **0,936** | **0,914** | **0,055** | **0,945** | **0,086** | **0,453** | **0,054** |
| P1 | 92,867 | 0,842 | 0,896 | 0,940 | 0,918 | 0,066 | 0,934 | 0,082 | 0,059 | 0,012 |

| Veri Setleri ve Makine Öğrenme Algoritmaları | Doğruluk (%) (Accuracy) | Kappa | F Ölçütü (F Measure) | AİK Alanı (ROC Area) | Doğru Artı oranı (TP rate) | Yanlış Artı oranı (FP rate) | Doğru Eksi oranı (TN rate) | Yanlış Eksi oranı (FN rate) | Öğrenme süresi (Train Time) | Sınıflandırma süresi (Test time) |
|---|---|---|---|---|---|---|---|---|---|---|
| P2 | 93,623 | 0,857 | 0,905 | 0,933 | 0,912 | 0,052 | 0,948 | 0,088 | 0,157 | 0,024 |
| P3 | 93,997 | 0,864 | 0,909 | 0,935 | 0,911 | 0,046 | 0,954 | 0,089 | 0,237 | 0,018 |
| **C-FS** | **73,869** | **0,458** | **0,556** | **0,744** | **0,578** | **0,119** | **0,881** | **0,422** | **1707,161** | **0,834** |
| **Naïve Bayes** | **88,657** | **0,772** | **0,879** | **0,959** | **0,881** | **0,109** | **0,891** | **0,119** | **0,051** | **0,050** |
| P1 | 88,657 | 0,772 | 0,879 | 0,959 | 0,881 | 0,109 | 0,891 | 0,119 | 0,051 | 0,050 |
| **IBk** | **91,782** | **0,836** | **0,919** | **0,945** | **0,975** | **0,133** | **0,867** | **0,025** | **0,029** | **0,364** |
| P1 | 90,745 | 0,815 | 0,908 | 0,910 | 0,958 | 0,137 | 0,863 | 0,043 | 0,010 | 0,081 |
| P2 | 92,517 | 0,851 | 0,927 | 0,960 | 0,988 | 0,131 | 0,869 | 0,012 | 0,011 | 0,131 |
| P3 | 92,085 | 0,842 | 0,922 | 0,965 | 0,980 | 0,132 | 0,868 | 0,020 | 0,008 | 0,152 |
| **J48** | **87,494** | **0,750** | **0,869** | **0,887** | **0,883** | **0,132** | **0,868** | **0,117** | **0,605** | **0,007** |
| P1 | 87,494 | 0,750 | 0,869 | 0,887 | 0,883 | 0,132 | 0,868 | 0,117 | 0,387 | 0,006 |
| P2 | 87,494 | 0,750 | 0,869 | 0,887 | 0,883 | 0,132 | 0,868 | 0,117 | 0,218 | 0,001 |
| **SVM** | **63,048** | **0,239** | **0,427** | **0,621** | **0,453** | **0,212** | **0,788** | **0,547** | **1705,543** | **0,344** |
| P1 | 67,200 | 0,343 | 0,657 | 0,673 | 0,710 | 0,363 | 0,637 | 0,290 | 807,927 | 0,010 |
| P2 | 69,052 | 0,374 | 0,625 | 0,689 | 0,650 | 0,272 | 0,728 | 0,350 | 894,616 | 0,018 |
| P3 | 52,892 | 0,000 | 0,000 | 0,500 | 0,000 | 0,000 | 1,000 | 1,000 | 3,000 | 0,316 |
| **Voted Perceptron** | **52,764** | **-0,002** | **0,005** | **0,497** | **0,003** | **0,005** | **0,995** | **0,997** | **0,933** | **0,069** |
| P1 | 52,892 | 0,000 | 0,000 | 0,495 | 0,000 | 0,000 | 1,000 | 1,000 | 0,064 | 0,010 |
| P2 | 52,738 | -0,003 | 0,001 | 0,498 | 0,001 | 0,004 | 0,996 | 0,999 | 0,254 | 0,028 |
| P3 | 52,662 | -0,004 | 0,013 | 0,499 | 0,007 | 0,011 | 0,989 | 0,993 | 0,615 | 0,031 |
| **D-FS** | **83,569** | **0,658** | **0,768** | **0,839** | **0,763** | **0,105** | **0,895** | **0,237** | **1111,109** | **1,210** |
| **Naïve Bayes** | **91,657** | **0,833** | **0,911** | **0,963** | **0,948** | **0,109** | **0,891** | **0,052** | **0,129** | **0,039** |
| P1 | 91,657 | 0,833 | 0,911 | 0,963 | 0,948 | 0,109 | 0,891 | 0,052 | 0,129 | 0,039 |
| **IBk** | **94,352** | **0,887** | **0,939** | **0,956** | **0,971** | **0,079** | **0,921** | **0,029** | **0,029** | **0,574** |
| P1 | 94,229 | 0,883 | 0,936 | 0,943 | 0,944 | 0,059 | 0,941 | 0,056 | 0,012 | 0,159 |
| P2 | 94,600 | 0,892 | 0,943 | 0,965 | 0,983 | 0,084 | 0,916 | 0,017 | 0,009 | 0,182 |
| P3 | 94,229 | 0,885 | 0,939 | 0,962 | 0,986 | 0,093 | 0,907 | 0,014 | 0,008 | 0,233 |
| **J48** | **92,514** | **0,849** | **0,917** | **0,927** | **0,927** | **0,076** | **0,924** | **0,073** | **0,468** | **0,003** |
| P1 | 92,514 | 0,849 | 0,917 | 0,927 | 0,927 | 0,076 | 0,924 | 0,073 | 0,271 | 0,001 |
| P2 | 92,514 | 0,849 | 0,917 | 0,927 | 0,927 | 0,076 | 0,924 | 0,073 | 0,197 | 0,002 |
| **SVM** | **67,486** | **0,312** | **0,475** | **0,656** | **0,485** | **0,173** | **0,827** | **0,515** | **1109,464** | **0,512** |
| P1 | 58,486 | 0,167 | 0,551 | 0,582 | 0,561 | 0,396 | 0,604 | 0,439 | 899,990 | 0,033 |
| P2 | 88,543 | 0,769 | 0,875 | 0,886 | 0,893 | 0,121 | 0,879 | 0,107 | 205,082 | 0,012 |
| P3 | 55,429 | 0,000 | 0,000 | 0,500 | 0,000 | 0,000 | 1,000 | 1,000 | 4,392 | 0,467 |
| **Voted Perceptron** | **80,210** | **0,589** | **0,743** | **0,803** | **0,660** | **0,083** | **0,917** | **0,340** | **1,019** | **0,082** |
| P1 | 77,200 | 0,524 | 0,695 | 0,782 | 0,597 | 0,087 | 0,913 | 0,403 | 0,062 | 0,021 |
| P2 | 80,571 | 0,597 | 0,750 | 0,804 | 0,669 | 0,083 | 0,917 | 0,331 | 0,256 | 0,023 |
| P3 | 82,857 | 0,646 | 0,785 | 0,823 | 0,715 | 0,080 | 0,920 | 0,285 | 0,701 | 0,038 |

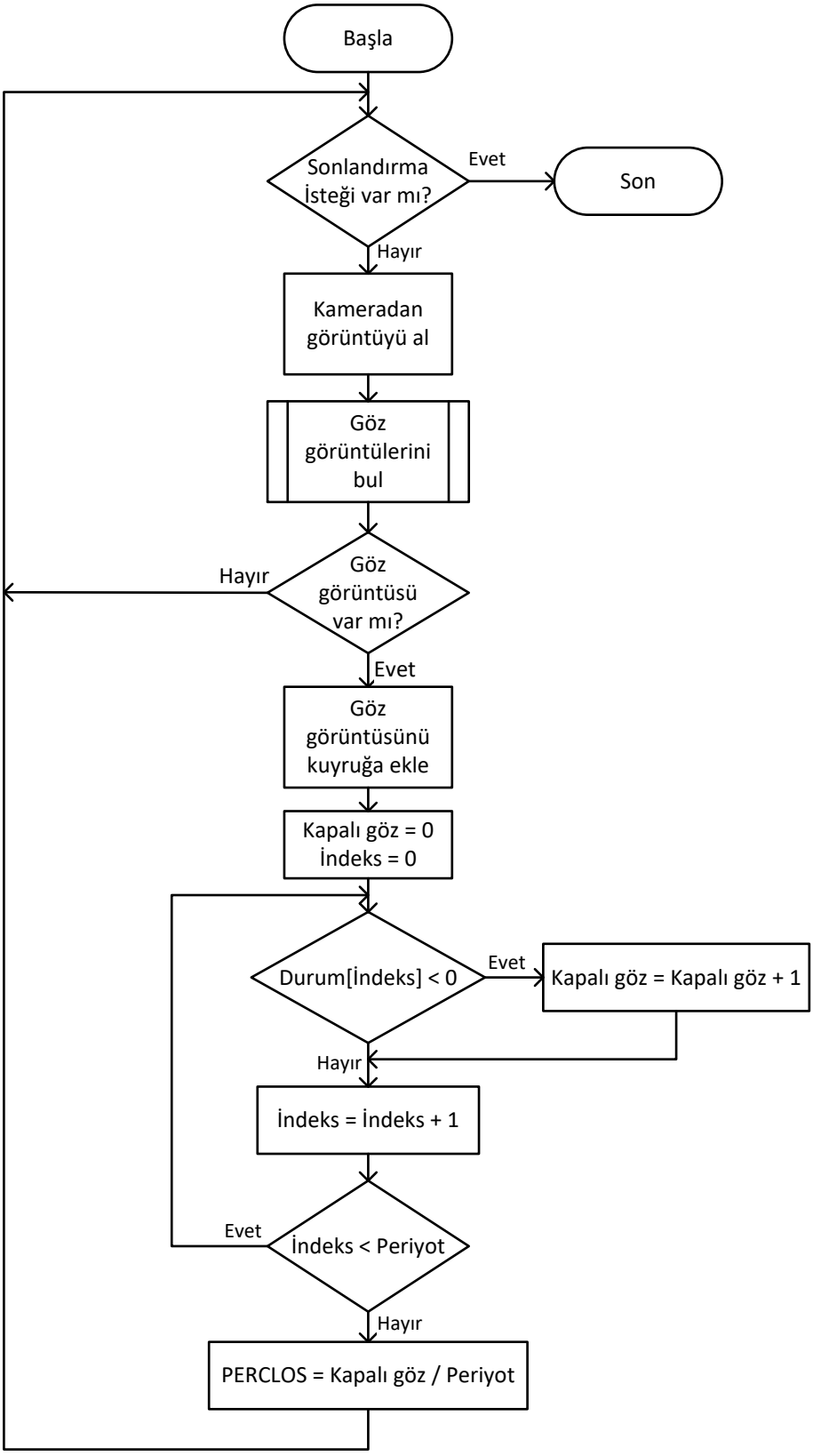
## 5.3. EK-C Uygulama Ana Çalışma Döngüsü Akış Diyagramı



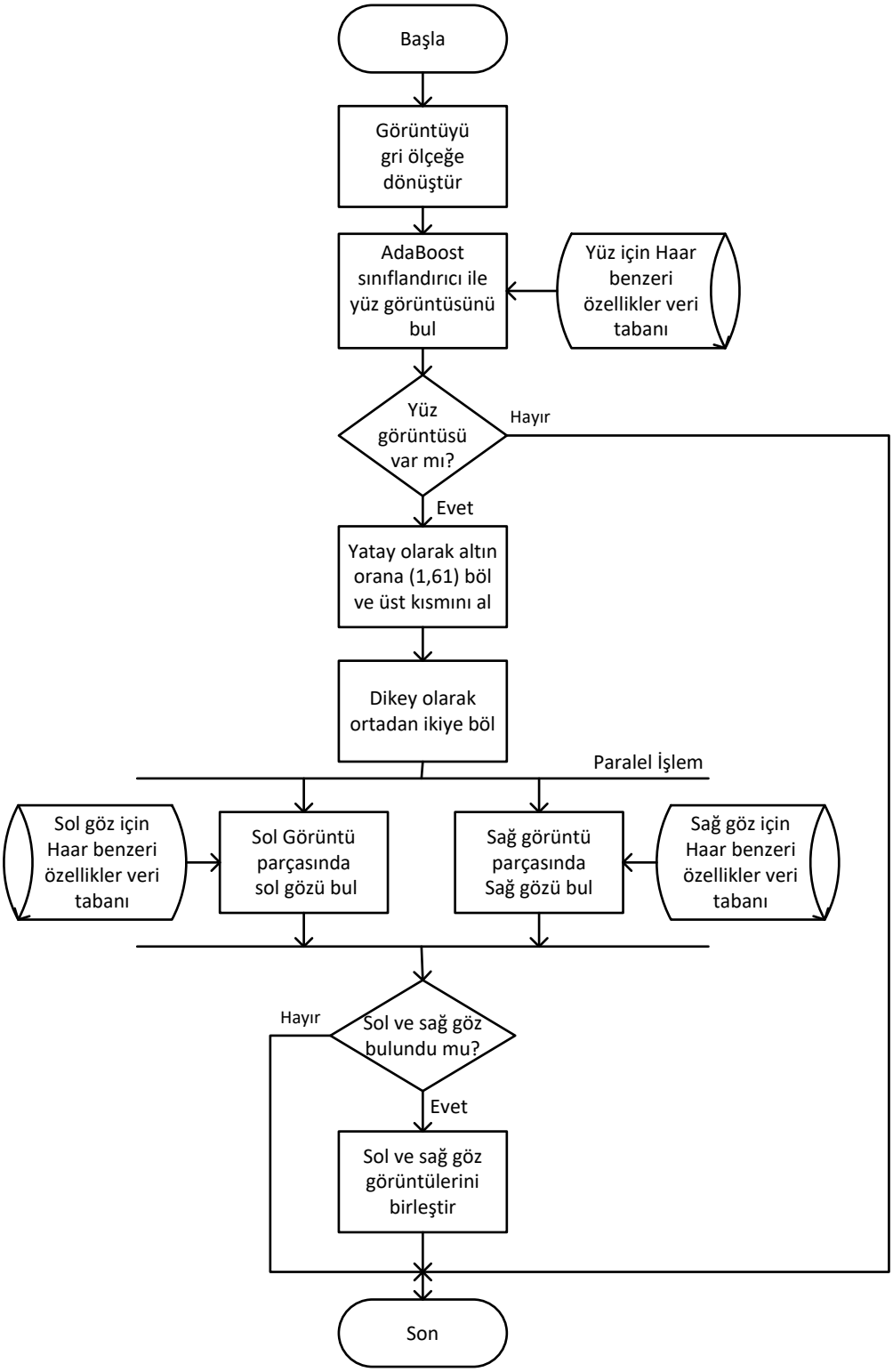
## 5.4. EK-D Uygulamada Veri Dosyası Oluşturma Akış Diyagramı



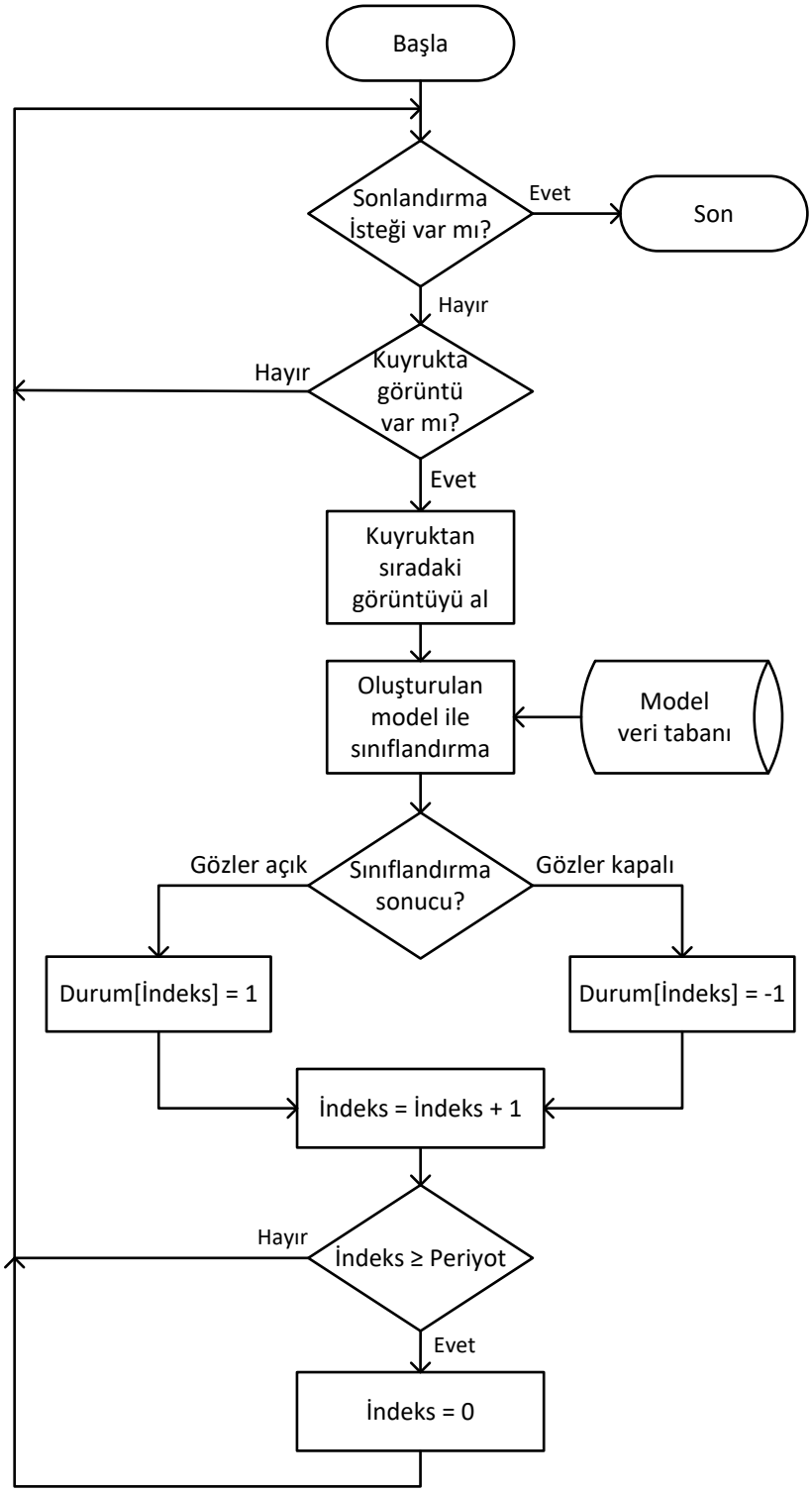
## 5.5. EK-E Görüntü İşleme Döngüsü Akış Diyagramı



## 5.6. EK-F Göz Görüntülerinin Bulunması ve Kırpılması Akış Diyagramı



## 5.7. EK-G Göz Görüntülerini Sınıflandıran Paralel Görevlerin Akış Diyagramı



## 5.8. EK-H Gözlerin Görüntü İçerisinde Bulunmasını Sağlayan Kaynak Kod

```
private void findFaceAndEyes(Image<Bgr, Byte> frame, out Rectangle faceRect, out Rectangle
eyesRect)
{
    faceRect = Rectangle.Empty;
    eyesRect = Rectangle.Empty;

    Image<Gray, Byte> gray = frame.Convert<Gray, byte>();

    Rectangle[] fcs = ccFace.DetectMultiScale(gray, 1.1, -1, new Size(gray.Width / 4, gray.Height
     / 4), new Size(gray.Width, gray.Height));

    if (fcs.Length == 0) return;

    faceRect = fcs[0]; // fcs[0].rect;

    Rectangle rect_eye_left = new Rectangle(faceRect.X, faceRect.Y, faceRect.Width / 2, (int)(
    faceRect.Height / 1.61));
    Rectangle rect_eye_right = new Rectangle(faceRect.X + (faceRect.Width / 2), faceRect.Y,
    faceRect.Width / 2, (int)(faceRect.Height / 1.61));

    gray.ROI = rect_eye_left;
    Image<Gray, Byte> left_eye_area = gray.Copy();

    gray.ROI = rect_eye_right;
    Image<Gray, Byte> right_eye_area = gray.Copy();

    gray.ROI = Rectangle.Empty;

    Task<Rectangle[]> task_left = new Task<Rectangle[]>(() =>
        ccLeftEye.DetectMultiScale(left_eye_area, 1.1, -1, new Size(left_eye_area.Width / 4,
        left_eye_area.Height / 4), left_eye_area.Size));

    Task<Rectangle[]> task_right = new Task<Rectangle[]>(() =>
        ccRightEye.DetectMultiScale(right_eye_area, 1.1, -1, new Size(right_eye_area.Width / 4,
        right_eye_area.Height / 4), right_eye_area.Size));

    // fork
    task_left.Start();
    task_right.Start();

    // join
    Task.WaitAll(task_left, task_right);

    if (task_left.Result==null || task_left.Result.Length==0 || task_right.Result==null ||
    task_right.Result.Length==0) return;

    Rectangle lr = task_left.Result[0];
    Rectangle rr = task_right.Result[0];

    lr.Offset(faceRect.Location);
    rr.Offset(faceRect.X + (faceRect.Width / 2), faceRect.Y);

    int x, y, w, h;
    x = lr.Left;
    y = lr.Top < rr.Top ? lr.Top : rr.Top;
    w = (rr.Left + rr.Width) - lr.Left;
    h = lr.Height > rr.Height ? lr.Height : rr.Height;
    eyesRect = new Rectangle(x, y, w, h);

    // GC.Collect();
}
```

## 5.9. EK-I Görüntü Çerçevelerini İşleyen Ana Fonksiyon Kaynak Kodu

```
void capture_ImageGrabbed(object sender, EventArgs e)
{
    try
    {
        Image<Bgr, Byte> frame = capture.RetrieveBgrFrame();
        if (frame == null) return;
        frameCount++;

        Image<Bgr, Byte> show = frame.Clone();

        Rectangle faceRect, eyesRect;
        findFaceAndEyes(frame, out faceRect, out eyesRect);

        if (!faceRect.IsEmpty)
        {
            show.Draw(faceRect, new Bgr(Color.Red), 2);

            if (!eyesRect.IsEmpty)
            {
                show.Draw(eyesRect, new Bgr(Color.Blue), 2);
                frame.ROI = eyesRect;
                Image<Bgr, Byte> eyes = frame.Clone();
                eyes._EqualizeHist();

                if (modelBuilt)
                {
                    eyeQueue.push(eyes);
                }
                else
                {
                    if (debug)
                    {
                        imageBoxEyes.Invoke(new MethodInvoker(delegate { imageBoxEyes.Image =
                        eyes; }));
                    }

                    if (numSamples > 0)
                    {
                        numSamples--;
                        totalSamples++;
                        string features = gaborFeature.GetFeaturesString(eyes, gaborParameter);
                        features += collectClassname;
                        if (streamWriter != null)
                        {
                            streamWriter.WriteLine(features);
                        }
                        if (debug)
                        {
                            labelClass.Invoke(new MethodInvoker(delegate { labelClass.Text =
                            "Collecting "+collectClassname; }));
                            textBoxConsole.Invoke(new MethodInvoker(delegate
                            {
                                textBoxConsole.AppendText(String.Format("Sample #{0} -> {1}
                                eye\r\n", totalSamples, collectClassname));
                            }));
                        }
                    }
                    else
                    {
                        if (streamWriter != null)
                        {
                            streamWriter.Close();
                            streamWriter = null;
                            soundPlayerNotify.Play();
                            textBoxConsole.Invoke(new MethodInvoker(delegate
                            {
                                textBoxConsole.AppendText(String.Format("Total {0} Number of
                                samples collected.\r\n", totalSamples));
                            }));
                        }
                        if (debug)
                        {
                            labelClass.Invoke(new MethodInvoker(delegate { labelClass.Text = "No
                            Model Built"; }));
                        }
                    }
                }
```

```
            lock (theLock)
            {
                int tempDur = 0;
                continuosClosure = false;
                openDuration = closeDuration = 0;
                for(int i=0; i<NUM_SAMPLES; i++)
                {
                    if (eyeStates[i] > 0)
                    {
                        openDuration++;
                        tempDur = 0;
                    }
                    else if (eyeStates[i] < 0)
                    {
                        openDuration++;
                        tempDur = 0;
                    }
                    else if (eyeStates[i] < 0)
                    {
                        closeDuration++;
                        tempDur++;
                    }
                    if (tempDur > NUM_CLOSDUR)
                    {
                        continuosClosure = true;
                    }
                }
                int total = openDuration + closeDuration;
                if (total >= NUM_SAMPLES)
                {
                    perclos = (double)closeDuration / (double)NUM_SAMPLES;
                }
            }
        }
    }

    frame.ROI = Rectangle.Empty;

    if (debug)
    {
        imageBoxFrame.Invoke(new MethodInvoker(delegate { imageBoxFrame.Image = show; }));

        labelQueue.Invoke(new MethodInvoker(delegate
        {
            labelQueue.Text = String.Format("Queue: {0}", eyeQueue.size());
        }));

        labelCloseRate.Invoke(new MethodInvoker(delegate
        {
            labelCloseRate.Text = String.Format("Close Rate: {0}/{1}", closeDuration,
            openDuration+closeDuration);
        }));
        labelPerclos.Invoke(new MethodInvoker(delegate
        {
            labelPerclos.Text = String.Format("PERCLOS : {0:F2}", perclos);
        }));
    }

    if(continuosClosure)
    {
        labelResult.Invoke(new MethodInvoker(delegate
        {
            labelResult.TextAlign = ContentAlignment.BottomCenter;
            labelResult.Text = "Wake up!";
            labelResult.BackColor = Color.Pink;
            labelResult.Image = DriverSleepinessDetection.Properties.Resources.Warning2;
        }));
        if (!notifiedCC)
        {
            notifiedCC = true;
            soundPlayerAlarm.Play();
        }
    }
```

```
        else if (Math.Abs(perclos - perclos2) > 0.01)
        {
            notifiedCC = false;
            perclos2 = perclos;
            labelResult.Invoke(new MethodInvoker(delegate
            {
                if (perclos > PERCLOS_HIGH)
                {
                    labelResult.TextAlign = ContentAlignment.BottomCenter;
                    labelResult.Text = "Sleepiness!";
                    labelResult.BackColor = Color.Pink;
                    labelResult.Image = DriverSleepinessDetection.Properties.Resources.Warning2;
                    if (!notifiedPH)
                    {
                        notifiedPH = true;
                        soundPlayerAlarm.Play();
                    }
                }
                else if (perclos > PERCLOS_LOW)
                {
                    labelResult.TextAlign = ContentAlignment.BottomCenter;
                    labelResult.Text = "Coffee Break?";
                    labelResult.BackColor = Color.LightYellow;
                    labelResult.Image = DriverSleepinessDetection.Properties.Resources.coffee2;
                    if (!notifiedPL)
                    {
                        notifiedPL = true;
                        notifiedPH = false;
                        soundPlayerWarning.Play();
                    }
                }
                else if (perclos > 0.0)
                {
                    labelResult.TextAlign = ContentAlignment.MiddleCenter;
                    labelResult.Text = "Normal";
                    labelResult.BackColor = Color.LimeGreen;
                    labelResult.Image = null;
                    notifiedPH = false;
                    notifiedPL = false;
                }
                else
                {
                    labelResult.TextAlign = ContentAlignment.MiddleCenter;
                    labelResult.Text = "Unknown";
                    labelResult.BackColor = Color.FromKnownColor(KnownColor.InactiveCaption);
                    labelResult.Image = null;
                    notifiedPH = false;
                    notifiedPL = false;
                }
            }));
        }

        if (stopWatch.ElapsedMilliseconds >= 1000)
        {
            fps = frameCount - frameCount2;
            frameCount2 = frameCount;
            stopWatch.Restart();
        }

        if (debug)
        {
            labelFPS.Invoke(new MethodInvoker(delegate
            {
                labelFPS.Text = String.Format("{0} FPS", fps);
            }));
            labelFrame.Invoke(new MethodInvoker(delegate
            {
                labelFrame.Text = String.Format("Frame #{0}", frameCount);
            }));
        }
        if (frameCount % 50 == 0)
        {
            GC.Collect();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

# KAYNAKLAR


[1] Association, A.A., ***Asleep at the Wheel: The Prevalence and Impact of Drowsy Driving.*** Professional Safety, 2011. 56(1): p. 12-12.

[2] Kurumu, T.İ. ***Türkiye İstatistik Kurumu***, Erişim Tarihi: 31.01.2017, Bağlantı: http://www.tuik.gov.tr.

[3] Öztürk, L., Z. Pelin, ve C. Özer, ***Sürücülerde Epworth Uykuluk Skoru ile Geçirilmiş ya da Atlatılan Trafik Kazası Sayısı Arasındaki İlişki***. 6. Ulusal Uyku ve Bozuklukları Kongresi, 2004.

[4] Özer, C., Ş. Etcibaşı, ve L. Öztürk, ***Daytime sleepiness and sleep habits as risk factors of traffic accidents in a group of Turkish public transport drivers***. International journal of clinical and experimental medicine, 2014. 7(1): p. 268-73.

[5] Öztürk, L., Y. Tufan, ve F. Güler, ***Self-Reported Traffic Accidents and Sleepiness in a Professional Group of Turkish Drivers.*** Sleep and Hypnosis, 2002. 4(3): p. 106-110.

[6] Müdürlüğü, M.G.v.Y.G. ***Karayolları Trafik Yönetmeliği***, Erişim Tarihi: 31.01.2017, Bağlantı: http://www.mevzuat.gov.tr/ Metin.Aspx? MevzuatKod= 7.5.8182& sourceXmlSearch= & MevzuatIliski=0.

[7] Garces Correa, A., L. Orosco, ve E. Laciar, ***Automatic detection of drowsiness in EEG records based on multimodal analysis***. Medical engineering & physics, 2014. 36(2): p. 244-9.

[8] Forsman, P., I. Pyykko, E. Toppila, ve E. Haeggstrom, ***Feasibility of force platform based roadside drowsiness screening - a pilot study***. Accident analysis and prevention, 2014. 62: p. 186-90.

[9] Daza, I.G., L.M. Bergasa, S. Bronte, J.J. Yebes, J. Almazan, ve R. Arroyo, ***Fusion of optimized indicators from Advanced Driver Assistance Systems (ADAS) for driver drowsiness detection***. Sensors, 2014. 14(1): p. 1106-31.

[10] Cyganek, B. ve S. Gruszczyński, ***Hybrid computer vision system for drivers' eye recognition and fatigue monitoring***. Neurocomputing, 2014. 126: p. 78-94.

[11] Kenneth, S., S. Arun, ve M. Murugappan, ***Detecting Driver Drowsiness Based on Sensors: A Review***. Sensors, Vol 12, Iss 12, Pp 16937-16953 (2012), 2012(12): p. 16937.

[12] Dong, Y., Z. Hu, K. Uchimura, ve N. Murayama, ***Driver inattention monitoring system for intelligent vehicles: A review***. 2009 Ieee Intelligent Vehicles Symposium, Vols 1 and 2, 2009: p. 875-880.

[13] Azim, T., M.A. Jaffar, ve A.M. Mirza, ***Fully automated real time fatigue detection of drivers through Fuzzy Expert Systems***. Applied Soft Computing, 2014. 18: p. 25-38.

[14] Liu, C.C., S.G. Hosking, ve M.G. Lenne, ***Predicting driver drowsiness using vehicle measures: recent insights and future challenges***. J Safety Res, 2009. 40(4): p. 239-45.

[15] Vural, E., M. Cetin, A. Ercil, G. Littlewort, M. Bartlett, ve J. Movellan, ***Machine learning systems for detecting driver drowsiness***. In-Vehicle Corpus and Signal Processing for Driver Behavior, 2009: p. 97-110.

[16] Trutschel, U., B. Sirois, D. Sommer, M. Golz, ve D. Edwards, ***Perclos: An alertness measure of the past***. Sixth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design: p. 172-179.

[17] Picot, A., S. Charbonnier, ve A. Caplier, ***On-Line Detection of Drowsiness Using Brain and Visual Information***. IEEE Transactions On Systems Man And Cybernetics Part A-Systems And Humans, 2012. 42(3): p. 764-775.

[18] Ahlstrom, C., K. Kircher, C. Fors, T. Dukic, C. Patten, ve A. Anund, ***Measuring driver impairments: Sleepiness, distraction, and workload***. IEEE Pulse, 2012. 3(2): p. 22-30.

[19] Khushaba, R.N., S. Kodagoda, S. Lal, ve G. Dissanayake, ***Driver Drowsiness Classification Using Fuzzy Wavelet Packet Based Feature Extraction Algorithm***. IEEE, 2011.

[20] Wang, Q., H. Wang, C. Zhao, ve J. Yang, ***Driver fatigue detection technology in active safety systems***. 2011.

[21] Bundele, M.M. ve R. Banerjee, ***ROC analysis of a fatigue classifier for vehicular drivers***. IEEE, 2010.

[22] Friedrichs, F. ve B. Yang, ***Camera-based drowsiness reference for driver state classification under real driving conditions***. 2010 Ieee Intelligent Vehicles Symposium (Iv), 2010: p. 101-106.

[23] Vadeby, A., A. Forsman, G. Kecklund, T. Akerstedt, D. Sandberg, ve A. Anund, ***Sleepiness and prediction of driver impairment in simulator studies using a Cox proportional hazard approach***. Accid Anal Prev, 2010. 42(3): p. 835-41.

[24] Zhang, C., H. Wang, ve R. Fu, ***Automated detection of driver fatigue based on entropy and complexity measures***. IEEE Transactions on Intelligent Transportation Systems, 2014. 15(1): p. 168-177.

[25] Solaz, J., J. Laparra-Hernández, D. Bande, N. Rodríguez, S. Veleff, J. Gerpe, ve E. Medina, ***Drowsiness Detection Based on the Analysis of Breathing Rate Obtained from Real-time Image Recognition***. Transportation Research Procedia, 2016. 14: p. 3867-3876.

[26] Correa, A.G., L. Orosco, ve E. Laciar, ***Automatic detection of drowsiness in EEG records based on multimodal analysis***. Medical Engineering & Physics, 2014. 36(2): p. 244-249.

[27] Golz, M. ve D. Sommer, ***Short-Term EEG Patterns of Driver Drowsiness and their Relation to Crashes***. Biomedizinische Technik / Biomedical Engineering, 2013.

[28] Gurudath, N. ve H.B. Riley, ***Drowsy Driving Detection by EEG Analysis Using Wavelet Transform and K-means Clustering***. Procedia Computer Science, 2014. 34: p. 400-409.

[29] Chieh, T.C., M.M. Mustafa, A. Hussain, S.F. Hendi, ve B.Y. Majlis, ***Development of vehicle driver drowsiness detection system using electrooculogram (EOG)***. IEEE, 2005.

[30] Picot, A., S. Charbonnier, A. Caplier, ve N.S. Vu, ***Using retina modelling to characterize blinking: comparison between EOG and video analysis***. Machine Vision And Applications, 2012. 23(6): p. 1195-1208.

[31] Chui, K.T., K.F. Tsang, H.R. Chi, B.W.K. Ling, ve C.K. Wu, ***An Accurate ECG-Based Transportation Safety Drowsiness Detection Scheme***. IEEE Transactions on Industrial Informatics, 2016. 12(4): p. 1438.

[32] Roy, R. ve K. Venkatasubramanian, ***EKG/ECG based driver alert system for long haul drive***. Indian Journal of Science and Technology, 2015. 8(19).

[33] Sahayadhas, A., K. Sundaraj, ve M. Murugappan, ***Electromyogram signal based hypovigilance detection***. Biomedical Research (0970-938X), 2014. 25(3): p. 281-288.

[34] Gang, L. ve C. Wan-Young, ***Detection of Driver Drowsiness Using Wavelet Analysis of Heart Rate Variability and a Support Vector Machine Classifier***. Sensors, 2013. 13(12): p. 16494-16511.

[35] Vicente, J., P. Laguna, A. Bartra, ve R. Bailon, ***Drowsiness detection using heart rate variability***. Medical & Biological Engineering & Computing, 2016(6): p. 927.

[36] Vicente, J., P. Laguna, A. Bartra, ve R. Bailon, ***Detection of driver's drowsiness by means of HRV analysis***. 2011 Computing in Cardiology (CinC), 2011: p. 89.

[37] Ronald R. Knipling, W.W.W., ***Vehicle-Based Drowsy Driver Detection Current Status and Future Prospects***. 1994.

[38] Yutian, F., H. Dexuan, ve N. Pingqiang, ***A combined eye states identification method for detection of driver fatigue***. IEEE, 2009.

[39] Bhowmick, B. ve K.S.C. Kumar, ***Detection and classification of eye state in IR camera for driver drowsiness identification***. IEEE, 2009.

[40] Liu, A., Z. Li, L. Wang, ve Y. Zhao, ***A practical driver fatigue detection algorithm based on eye state***. IEEE, 2010.

[41] Alioua, N., A. Amine, M. Rziza, ve D. Aboutajdine. ***Eye state analysis using iris detection based on Circular Hough Transform***. in Multimedia Computing and Systems (ICMCS), 2011 International Conference on. 2011. IEEE.

[42] Bo, Z., W. Wenjun, ve C. Bo, ***Driver Eye State Classification Based on Cooccurrence Matrix of Oriented Gradients***. Advances in Mechanical Engineering, Vol 7, Iss 2 (2015), 2015(2).

[43] Dehnavi, M., K. Izadi, ve M. Eshghi, ***Driver drowsiness detection based on open eye detection with visual information***. International Review on Computers and Software, 2012. 7(2): p. 651-656.

[44] Lenskiy, A.A. ve J.S. Lee, ***Driver's Eye Blinking Detection Using Novel Color and Texture Segmentation Algorithms***. International Journal Of Control Automation And Systems, 2012. 10(2): p. 317-327.

[45] Qingzhang, C., W. Wenfu, ve C. Yuqin, ***Research on Eye-state Based Monitoring for Drivers' Dozing***. 2009: p. 373-376.

[46] Wang, H., Y. Chen, Q. Wang, M.W. Ren, C.X. Zhao, ve J.Y. Yang, ***A Practical Eye State Recognition Based Driver fatigue detection method***. Proceedings of the 2009 Chinese Conference on Pattern Recognition and the First Cjk Joint Workshop on Pattern Recognition, Vols 1 and 2, 2009: p. 423-427.

[47] Zhan, T., Z.-m. Li, ve J. Zhang. ***A practical real-time detection visual system for driver's eye closure state tracking***. in Fourth International Conference on Machine Vision (ICMV 11). 2012. International Society for Optics and Photonics.

[48] Zhang, B., W.J. Wang, ve B. Cheng, ***Driver Eye State Classification Based on Cooccurrence Matrix of Oriented Gradients***. Advances In Mechanical Engineering, 2015. 7(2).

[49] Jackson, M.L., S. Raj, R.J. Croft, A.C. Hayley, L.A. Downey, G.A. Kennedy, ve M.E. Howard, ***Slow eyelid closure as a measure of driver drowsiness and its relationship to performance***. Traffic Injury Prevention, 2016(3): p. 251.

[50] Knopp, S.J., P.J. Bones, S.J. Weddell, C.R. Innes, ve R.D. Jones. ***A miniature head-mounted camera for measuring eye closure***. in Proceedings of the 27th Conference on Image and Vision Computing New Zealand. 2012. ACM.

[51] Wilkinson, V.E., M.L. Jackson, J. Westlake, B. Stevens, M. Barnes, P. Swann, S.M.W. Rajaratnam, ve M.E. Howard, ***The Accuracy of Eyelid Movement Parameters for Drowsiness Detection***. Journal Of Clinical Sleep Medicine, 2013. 9(12): p. 1315-1324.

[52] Akrout, B. ve W. Mahdi, ***A blinking measurement method for driver drowsiness detection***. Advances in Intelligent Systems and Computing. Vol. 226. 2013: Springer Verlag. 651-660.

[53] Gheis, M., S. Jamshid, ve S. Abdolhossein, ***A Fast and Adaptive Video-Based Method for Eye Blink Rate Estimation***. International Journal of Advanced Computer Research, Vol 5, Iss 19, Pp 105-114 (2015), 2015(19): p. 105.

[54] Hsieh, C.-S. ve C.-C. Tai, ***An improved and portable eye-blink duration detection system to warn of driver fatigue***. Instrumentation Science & Technology, 2013. 41(5): p. 429-444.

[55] Ito, T., S. Mita, K. Kozuka, T. Nakano, ve S. Yamamoto, ***Driver blink measurement by the motion picture processing and its application to drowsiness detection***. Ieee

5th International Conference on Intelligent Transportation Systems, Proceedings, 2002: p. 168-173.

[56] Lo Castro, F., ***Class I infrared eye blinking detector***. Sensors and Actuators A: Physical, 2008. 148(2): p. 388-394.

[57] Ma'touq, J., J. Al-Nabulsi, A. Al-Kazwini, A. Baniyassien, G. Al-Haj Issa, ve H. Mohammad, ***Eye blinking-based method for detecting driver drowsiness***. Journal of Medical Engineering & Technology, 2014. 38(8): p. 416-419.

[58] Mohammadi, G., J. Shanbehzadeh, ve A. Sarrafzadeh, ***A Fast and Adaptive Video-Based Method for Eye Blink Rate Estimation***. International Journal of Advanced Computer Research, 2015. 5(19): p. 105.

[59] Prasertsak, T. ve R. Choopan, ***A Study of Two Robust Features for Effective Open or Closed Eye Classification***. Applied Mechanics & Materials, 2015. 781: p. 507.

[60] Azim, T., M.A. Jaffar, ve A.M. Mirza, ***Automatic Fatigue Detection of Drivers through Pupil Detection and Yawning Analysis***. IEEE, 2009.

[61] Abtahi, S., B. Hariri, ve S. Shirmohammadi, ***Driver drowsiness monitoring based on yawning detection***. 2011 IEEE Instrumentation & Measurement Technology Conference (I2MTC), 2011: p. 1.

[62] Anitha, C., M.K. Venkatesha, ve B.S. Adiga, ***A Two Fold Expert System for Yawning Detection***. Procedia Computer Science, 2016. 92: p. 63-71.

[63] Fan, X., B.C. Yin, ve Y.F. Sun, ***Yawning detection for monitoring driver fatigue***. Proceedings of 2007 International Conference on Machine Learning and Cybernetics, Vols 1-7, 2007: p. 664-668.

[64] Li, L.L., Y.Z. Chen, ve Z.L. Li, ***Yawning Detection for Monitoring Driver Fatigue Based on Two Cameras***. 2009 12th International Ieee Conference on Intelligent Transportation Systems (Itsc 2009), 2009: p. 12-17.

[65] Omidyeganeh, M., A. Javadtalab, ve S. Shirmohammadi, ***Intelligent driver drowsiness detection through fusion of yawning and eye closure***. 2011 IEEE International Conference on Virtual Environments Human-Computer Interfaces & Measurement Systems (VECIMS), 2011: p. 1.

[66] Liu, K., Y. Luo, G. TEI, ve S. Yang, ***Attention recognition of drivers based on head pose estimation***. IEEE, 2008.

[67] Ayush, J., G. Shruti, ve B. Amit, ***Eye State and Head Position Technique for Driver Drowsiness Detection***. International Journal of Electronics and Computer Science Engineering, Vol 2, Iss 3, Pp 874-879 (2013), 2013(3): p. 874.

[68] Choi, I.H. ve Y.G. Kim, ***Head pose and gaze direction tracking for detecting a drowsy driver***. Applied Mathematics and Information Sciences, 2015. 9(2): p. 505-512.

[69] Popieul, J.C., P. Simon, ve P. Loslever, ***Using driver's head movements evolution as a drowsiness indicator***. Ieee Iv2003: Intelligent Vehicles Symposium, Proceedings, 2003: p. 616-621.

[70] Zhu, Y.D. ve K. Fujimura, ***Head pose estimation for driver monitoring***. 2004 Ieee Intelligent Vehicles Symposium, 2004: p. 501-506.

[71] Chieh, T.C., M.M. Mustafa, A. Hussain, E. Zahedi, ve B.Y. Majlis, ***Driver fatigue detection using steering grip force***. SCOReD 2003: Student Conference on Research and Development, Proceedings, 2003: p. 45-48.

[72] Kyehoon, L.E.E., H. Sung-Ae, ve O.A.H. Shezeen, ***Detecting Driver Fatigue by Steering Wheel Grip Force***. International Journal of Contents, 2016. 12(1): p. 44.

[73] Rogado, E., J.L. Garcia, R. Barea, L.M. Bergasa, ve E. Lopez, ***Driver fatigue detection system***. 2008 Ieee International Conference on Robotics and Biomimetics, Vols 1-4, 2009: p. 1105-1110.

[74] Torkkola, K., N. Massey, ve C. Wood, ***Detecting driver inattention in the absence of driver monitoring sensors***. Proceedings of the 2004 International Conference on Machine Learning and Applications (Icmla'04), 2004: p. 220-226.

[75] Torkkola, K., N. Massey, ve C. Wood, ***Driver inattention detection through intelligent analysis of readily available sensors***. Itsc 2004: 7th International Ieee Conference on Intelligent Transportation Systems, Proceedings, 2004: p. 326-331.

[76] Polychronopoulos, A., A. Amditis, ve E. Bekiaris, ***Information data flow in AWAKE multi-sensor driver monitoring system***. 2004 Ieee Intelligent Vehicles Symposium, 2004: p. 902-906.

[77] Bando, S. ve A. Nozawa, ***Detection of driver inattention from fluctuations in vehicle operating data***. Artificial Life and Robotics, 2015(1): p. 28.

[78] Friedrichs, F., M. Miksch, ve B. Yang, ***Estimation of lane data-based features by odometric vehicle data for driver state monitoring***. IEEE, 2010.

[79] Kim, Y., Y. Kim, ve M. Hahn, ***Detecting Driver Fatigue based on the Driver's Response Pattern and the Front View Environment of an Automobile***. 2008: p. 237-240.

[80] Lee, B.L., W.Y. Chung, ve B.G. Lee, ***Standalone Wearable Driver Drowsiness Detection System in a Smartwatch***. IEEE Sensors Journal, 2016. 16(13): p. 5444-5451.

[81] Leng, L.B., L.B. Giin, ve W.-Y. Chung, ***Wearable driver drowsiness detection system based on biomedical and motion sensors***. IEEE Sensors 2015, 2015: p. 1.

[82] Li, G., B.L. Lee, ve W.Y. Chung, ***Smartwatch-Based Wearable EEG System for Driver Drowsiness Detection***. IEEE Sensors Journal, 2015. 15(12): p. 7169-7180.

[83] Lin, C.T., C.H. Chuang, C.S. Huang, S.F. Tsai, S.W. Lu, Y.H. Chen, ve L.W. Ko, ***Wireless and Wearable EEG System for Evaluating Driver Vigilance***. IEEE Transactions On Biomedical Circuits And Systems, 2014. 8(2): p. 165-176.

[84] Sergio Ríos, A., M. José Luis Miguel, S. Andrés Millán, ve V. Álvaro Sánchez, ***Variation of the Heartbeat and Activity as an Indicator of Drowsiness at the Wheel Using a Smartwatch***. International Journal of Interactive Multimedia and Artificial Intelligence, Vol 3, Iss 3, Pp 96-100 (2015), 2015(3): p. 96.

[85] Warwick, B., N. Symons, X. Chen, ve K. Xiong, ***Detecting Driver Drowsiness Using Wireless Wearables***. 2015 IEEE 12th International Conference on Mobile Ad Hoc & Sensor Systems, 2015: p. 585.

[86] Craye, C., A. Rashwan, M.S. Kamel, ve F. Karray, ***A Multi-Modal Driver Fatigue and Distraction Assessment System***. International Journal of Intelligent Transportation Systems Research, 2016. 14(3): p. 173-194.

[87] Keshava Murthy, G.N. ve Z.A. Khan, ***Smart alert system for driver drowsiness using EEG and eyelid movements***. Middle East Journal of Scientific Research, 2013. 14(5): p. 610-619.

[88] Li, G. ve W.Y. Chung, ***Estimation of Eye Closure Degree Using EEG Sensors and Its Application in Driver Drowsiness Detection***. Sensors, 2014. 14(9): p. 17491-17515.

[89] Mbouna, R.O., S.G. Kong, ve M.G. Chun, ***Visual analysis of eye state and head pose for driver alertness monitoring***. IEEE Transactions on Intelligent Transportation Systems, 2013. 14(3): p. 1462-1469.

[90] Cario, G., A. Casavola, G. Franze, ve M. Lupia, ***A hybrid observer approach for driver drowsiness detection***. 19th Mediterranean Conference on Control and Automation, 2011.

[91] Miyaji, M., M. Danno, ve K. Oguri, ***Analysis of Driver Behavior Based on Traffic Incidents for Driver Monitor Systems***. 2008 Ieee Intelligent Vehicles Symposium, Vols 1-3, 2008: p. 31-36.

[92] Baulk, S.D., S.N. Biggs, K.J. Reid, C.J. van den Heuvel, ve D. Dawson, ***Chasing the silver bullet: measuring driver fatigue using simple and complex tasks***. Accid Anal Prev, 2008. 40(1): p. 396-402.

[93] Ibarra-Orozco, R., M. Gonzalez-Mendoza, N. Hernandez-Gress, F. Diederichs, ve J. Kortelainen, ***Towards a Ready-to-Use Drivers' Vigilance Monitoring System***. 2008: p. 802-807.

[94] Hu, S. ve G. Zheng, ***Driver drowsiness detection with eyelid related parameters by Support Vector Machine***. Expert Systems with Applications, 2009. 36(4): p. 7651-7658.

[95] Sandberg, D., T. Åkerstedt, A. Anund, G. Kecklund, ve M. Wahde, ***Detecting Driver Sleepiness Using Optimized Nonlinear Combinations of Sleepiness Indicators***. IEEE, 2010.

[96] Daza, I.G., N. Hernandez, L.M. Bergasa, I. Parra, J.J. Yebes, M. Gavilan, R. Quintero, D.F. LIorca, ve M.A. Sotelo, ***Drowsiness monitoring based on driver and driving data fusion***. 14th International IEEE Conference on Intelligent Transportation Systems, 2011.

[97] Forsman, P.M., B.J. Vila, R.A. Short, C.G. Mott, ve H.P. Van Dongen, ***Efficient driver drowsiness detection at moderate levels of drowsiness***. Accid Anal Prev, 2013. 50: p. 341-50.

[98] Garcia, I., S. Bronte, L. M. Bergasa, N. Hernandez, B. Delgado, ve M. Sevillano, ***Vision-based drowsiness detector for a realistic driving simulator***. 13th International IEEE Annual Conference on Intelligent Transportation Systems, 2010.

[99] Picot, A., S. Charbonnier, ve A. Caplier, ***On-line automatic detection of driver drowsiness using a single electroencephalographic channel***. Conf Proc IEEE Eng Med Biol Soc, 2008. 2008: p. 3864-7.

[100] ***Car Magazine Web Site***, Erişim Tarihi: 21.12.2013, Bağlantı: http://www.carmagazine.co.uk/News/Search-Results/Industry-News/Mercedes-launches-Attention-Assist/.

[101] ***Bosh Automotive Technology***, Erişim Tarihi: 21.12.2013, Bağlantı: http://www.bosch-automotivetechnology.com/en/de/driving_safety/driving_safety_systems_for_passenger_cars_1/driver_assistance_systems/driver_assistance_systems_2.html

[102] ***Bosh Life Web Site***, Erişim Tarihi: 21.12.2013, Bağlantı: http://life.bosch.com.cn/en/invented-for-life/citizen/bosch-driver-drowsiness-detection.html.

[103] ***Ford Media Web Site***, Erişim Tarihi: 21.12.2013, Bağlantı: http://technology.fordmedia.eu/.

[104] Xie, X.L., J.B. Hu, X.M. Liu, P.S. Li, ve S.Y. Wang, ***The EEG changes during night-time driver fatigue***. 2009 Ieee Intelligent Vehicles Symposium, Vols 1 and 2, 2009. 1-2: p. 935-939.

[105] Sun, G., Y. Jin, Z. Li, F. Zhang, ve L. Jia, ***A vision-based head status judging algorithm for driving fatigue detection system***. Advances in Transportation Studies, 2015(37): p. 51-64.

[106] Mukherjee, K., R. Karmakar, ve S. Das, ***Effective Estimation of Driver Drowsiness Based on Eye Status Detection and Analysis***. 2014 International Conference on Devices, Circuits & Communications (ICDCCom), 2014: p. 1.

[107] Liu, Y.L., H. Zhang, ve J.F. Liu, ***Driver Fatigue Monitoring Method Based on Eyes State Classification***. 2008 Chinese Control and Decision Conference, Vols 1-11, 2008: p. 2257-2260.

[108] Jimenez, P., L.M. Bergasa, J. Nuevo, N. Hernandez, ve I.G. Daza, ***Gaze fixation system for the evaluation of driver distractions induced by IVIS***. IEEE Transactions on Intelligent Transportation Systems, 2012. 13(3): p. 1167-1178.

[109] reza Ashouri, M., A. Nahvi, S. Azadi, M. Niknejad, ve A. Sadeghi, ***Drowsy Driving Analysis Based on Steering & Lane Position Variables Using Passenger Driving Simulator. (English)***. Modares Mechanical Engineering, 2014. 14(9): p. 165.

[110] McDonald, A.D., J.D. Lee, C. Schwarz, ve T.L. Brown, ***Steering in a Random Forest: Ensemble Learning for Detecting Drowsiness-Related Lane Departures***. HUMAN FACTORS, 2014. 56(5): p. 986-998.

[111] Li, X.P., E. Seignez, ve P. Loonis. ***Driver drowsiness estimation by fusion of lane and eye features using a multilevel evidence theory***. in International Conference on Cyber Technology in Automation, Control & Intelligent Systems. 2013.

[112] Huang, S.S., C.F. Chen, P.Y. Hsiao, ve L.C. Fu, ***On-board vision system for lane recognition and front-vehicle detection to enhance driver's awareness***. 2004 Ieee International Conference on Robotics and Automation, Vols 1- 5, Proceedings, 2004: p. 2456-2461.

[113] McDonald, A.D., C. Schwarz, J.D. Lee, ve T.L. Brown. ***Real-time detection of drowsiness related lane departures using steering wheel angle***. in Proceedings of the Human Factors and Ergonomics Society Annual Meeting. 2012. Sage Publications.

[114] Acharya, T. ve A.K. Ray, ***Image processing: principles and applications***. 2005: John Wiley & Sons.

[115] Awcock, G.J. ve R. Thomas, ***Applied image processing***. 1995: McGraw-Hill, Inc.

[116] Young, I.T., J.J. Gerbrands, ve L.J.v. Vliet, ***Fundamentals of Image Processing***. 2007, Delft University of Technology.

[117] Bradski, G. ve A. Kaehler, ***Learning OpenCV: Computer vision with the OpenCV library***. 2008: " O'Reilly Media, Inc.".

[118] Russ, J.C. ve R.P. Woods, ***The image processing handbook***. 1995, LWW.

[119] Davies, E.R., ***Machine vision: theory, algorithms, practicalities***. 2004: Elsevier.

[120] Derpanis, K.G., ***Gabor Filters***, in Gabor Filters. 2007, York University.

[121] Chao, W.L., ***Gabor wavelet transform and its application***. R98942073 (TFA&WT final project), 2010.

[122] Daugman, J.G., ***Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters***. JOSA A, 1985. 2(7): p. 1160-1169.

[123] Viola, P. ve M. Jones. ***Rapid object detection using a boosted cascade of simple features***. in Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. 2001. IEEE.

[124] Viola, P. ve M. Jones, ***Fast and robust classification using asymmetric adaboost and a detector cascade***. Advances in Neural Information Processing System, 2001. 14.

[125] Lienhart, R. ve J. Maydt. ***An extended set of haar-like features for rapid object detection***. in Image Processing. 2002. Proceedings. 2002 International Conference on. 2002. IEEE.

[126] Akpınar, P.D.H., ***Data Veri Madenciliği Veri Analizi***. 2014: Papatya Yayıncılık Eğitim.

[127] Alpaydin, E., ***Introduction to machine learning***. 2014: MIT press.

[128] Özkan, Y., ***Veri madenciliği yöntemleri***. 2008: Papatya Yayıncılık Eğitim.

[129] Russell, S.J., P. Norvig, J.F. Canny, J.M. Malik, ve D.D. Edwards, ***Artificial intelligence: a modern approach***. Vol. 2. 2003: Prentice hall Upper Saddle River.

[130] Microsoft. ***Microsoft Visual Studio 2015 Community***, Erişim, Bağlantı: https://www.visualstudio.com/.

[131] Team, O.D. ***OpenCV - Open Source Computer Vision Library***, Erişim Tarihi: 2017, Bağlantı: http://opencv.org/.

[132] Corporation, E. ***Emgu CV Library***, Erişim, Bağlantı: http://www.emgu.com.

[133] Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, ve I.H. Witten, ***The WEKA data mining software: an update***. ACM SIGKDD explorations newsletter, 2009. 11(1): p. 10-18.

[134] Frijters, J. ***IKVM.NET***, Erişim, Bağlantı: https://www.ikvm.net/.

[135] Development, F. ***City Car Driving***, Erişim, Bağlantı: http://www.citycardriving.com/.

[136] ITU-R, ***Parameter values for the HDTV standards for production and international programme exchange*** Recommendation ITU-R BT.709-6, 2015.

[137] Bozkir, M.G., P. Karakas, ve Ö. Oguz, ***Vertical and horizontal neoclassical facial canons in Turkish young adults***. Surgical and Radiologic Anatomy, 2004. 26(3): p. 212-219.

[138] Ilonen, J., J.-K. Kämäräinen, ve H. Kälviäinen, ***Efficient computation of Gabor features***. 2005: Lappeenranta University of Technology.

[139] Struc, V. ve N. Pavesic, ***From Gabor Magnitude to Gabor Phase Features: Tackling the Problem of Face Recognition under Severe Illumination Changes***. 2010: INTECH Open Access Publisher.

[140] Weiman, C.F. ***Efficient discrete Gabor functions for robot vision***. in SPIE's International Symposium on Optical Engineering and Photonics in Aerospace Sensing. 1994. International Society for Optics and Photonics.

[141] Hall, M.A. ve L.A. Smith, ***Practical feature subset selection for machine learning***. 1998.

[142] VanderWerf, F., P. Brassinga, D. Reits, M. Aramideh, ve B.O. Visser, ***Eyelid movements: behavioral studies of blinking in humans under different stimulus conditions***. Journal of neurophysiology, 2003. 89(5): p. 2784-2796.

[143] Nyquist, H., ***Certain topics in telegraph transmission theory***. Transactions of the American Institute of Electrical Engineers, 1928. 47(2): p. 617-644.

[144] Miguel, T.T. ve J.P. Javier, ***Optical Flow and Driver's Kinematics Analysis for State of Alert Sensing***. Sensors, 2013. 13(4): p. 4225-4257.

[145] Weng, M.C., C.T. Chen, ve H.C. Kao, ***Remote Surveillance System for Driver Drowsiness in Real-time Using Low-cost Embedded Platform***. 2008 Ieee International Conference on Vehicular Electronics and Safety, 2008: p. 60-64.

[146] Fan, X., Y. Sun, B. Yin, ve X. Guo, ***Gabor-based dynamic representation for human fatigue monitoring in facial image sequences***. Pattern Recognition Letters, 2010. 31(3): p. 234-243.

[147] Jimenez-Pinto, J. ve M. Torres-Torriti, ***Face salient points and eyes tracking for robust drowsiness detection***. Robotica, 2012. 30: p. 731-741.

[148] Dasgupta, A., A. George, S.L. Happy, A. Routray, ve T. Shanker, ***An on-board vision based system for drowsiness detection in automotive drivers***. International Journal of Advances in Eng. Sci. and Applied Math., 2013. 5(2-3): p. 94-103.

[149] Dhar, S., T. Pradhan, S. Gupta, ve A. Routray. ***Implementation of real time Visual Attention Monitoring algorithm of human drivers on an embedded platform***. in Proceedings of the 2010 IEEE Students' Technology Symposium. 2010.

[150] Selvakumar, K., J. Jerome, K. Rajamani, ve N. Shankar, ***Real-Time Vision Based Driver Drowsiness Detection Using Partial Least Squares Analysis***. Journal of Signal Processing Systems, 2016. 85(2): p. 263-274.

# ÖZGEÇMİŞ

1977 yılında Almanya'da doğdu. İlkokulu ve Ortaokulu memleketi olan Tekirdağ'ın Malkara ilçesinde bitirdi. Malkara Hüsniye Hanım Teknik Lisesi'ndeki öğrenimini 1996 yılında okul birinciliği derecesi ile bitirdi. Aynı yıl başladığı Marmara Üniversitesi Teknik Eğitim Fakültesi Elektrik Eğitimi Bölümünü 2000 yılında tamamladı. Marmara Üniversitesi Fen Bilimleri Enstitüsü Elektrik Eğitimi Programında Yüksek Lisans öğrenimini 2004 yılında tamamladı. Askerlik görevini tamamladıktan sonra özel sektörde gömülü sistem programlama üzerine çalıştı. 2006 yılında Trakya Üniversitesi İpsala Meslek Yüksekokulu'nda Öğretim Görevlisi olarak görevine başladı. 2008 yılında Trakya Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nde Doktora öğrenimine başladı. Ozan Akı evli ve bir kız çocuğu babasıdır.

# TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER

Bu tez çalışması kapsamında yayınlanan bilimsel çalışmalar aşağıda listelenmiştir.

İlhan UMUT, **Ozan AKI**, Erdem UÇAR, Levent ÖZTÜRK, "*Detection of Driver Sleepiness and Warning the Driver in Real-Time using Image Processing and Machine Learning Techniques*", Advances in Science and Technology Research Journal (ASTRJ), Volume 11, Issue 2, 2017, Lublin, Poland.

İlhan UMUT, Gökhan KOÇYİĞİT, **Ozan AKI**, "*Windows HPC Sunucu ile Dağıtık Görüntü İşleme*", International Conference on Quality in Higher Education ICQH 2016 Bildiriler Kitabı, Kasım 2016, Sakarya.

**Ozan AKI**, Aydın GÜLLÜ, Erdem UÇAR, "*Classification of Rice Grains using Image Processing and Machine Learning Techniques*", International Scientific Conference UNITECH 2015, 20-21 November 2015, Gabrovo, Bulgaria.