Çokyüzlü Konik Sınıflandırıcılar ve Uygulamaları

Halil Sağlamlar

DOKTORA TEZİ

Elektrik Elektronik Mühendisliği Anabilim Dalı

Aralık 2020

Polyhedral Conic Classifiers and Their Applications

Halil Sağlamlar

DOCTORAL DISSERTATION

Department of Electrical and Electronics Engineering

December 2020

Çokyüzlü Konik Sınıflandırıcılar ve Uygulamaları

Halil Sağlamlar

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Elektrik Elektronik Mühendisliği Anabilim Dalı
Telekomünikasyon-Sinyal İşleme Bilim Dalında
DOKTORA TEZİ
Olarak Hazırlanmıştır

Danışman: Prof. Dr. Hakan Çevikalp

Aralık 2020

ETİK BEYAN

Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kılavuzuna göre, Prof. Dr. Hakan Çevikalp danışmanlığında hazırlamış olduğum "Çokyüzlü Konik Sınıflandırıcılar ve Uygulamaları" başlıklı DOKTORA tezimin özgün bir çalışma olduğunu; tez çalışmamın tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; tezimde verdiğim bilgileri, verileri akademik ve bilimsel etik ilke ve kurallara uygun olarak elde ettiğimi; tez çalışmamda yararlandığım eserlerin tümüne atıf yaptığımı ve kaynak gösterdiğimi ve bilgi, belge ve sonuçları bilimsel etik ilke ve kurallara göre sunduğumu beyan ederim. 05/11/2020

Halil Sağlamlar

İmza

ÖZET

Bu tezde görsel nesne bulma ve açık küme tanıma görevlerinde diğer geniş marjlı yöntemlerden daha iyi bir performans gösteren doğrusal sınıflandırma benzeri bir sınıflandırıcı ailesi önerilmektedir. Bahsedilen görevlerde sınıflandırma problemi pozitif ve negatif örnekler açısından dengeli değildir. Pozitif veriler, negatiflerden sayıca daha az olmakla birlikte, geometrik olarak kompakt ve görsel olarak tutarlı dağılım göstermektedir. Buna karşın negatif veriler dağınık ve çok çeşitli bir dağılıma sahiptir. Bu tarz problemlerde pozitif örnekleri sıkıca çevreleyen, bu esnada iki sınıfın örtüştüğü bölgelerdeki negatifleri de dikkate alarak karar sınırlarını oluşturan bir sınıflandırıcıya ihtiyaç duyulur. Bu amaçla çalışmada kompakt karar sınırları bulan "çokyüzlü konik" sınıflandırıcılar önerilmiştir. Önerilen bu sınıflandırıcılar pozitif verileri bir koninin alt seviye kümelerine yerleştirerek sıkı ve kompakt karar sınırları sağlamaktadır. Bu yöntemlere ilave olarak, pozitif verilerin birbirinden uzak ve farklı bölgelerde kümelendiği durumlarda sınıflandırmayı amaçlayan farklı bir çokyüzlü konik sınıflandırıcı da geliştirilmiştir. Önerilen yöntemler, doğrusal Destek Vektör Makineleri (SVM) ile benzer özelliklere ve çalışma süresine sahiptir. Ayrıca önerilen yöntemler SVM'in kullandığı kuvadratik programlama kullanılarak iki sınıflı veya yalnızca pozitif sınıfla da eğitilebilir. Yapılan deneylerde nesne bulma, yüz doğrulama, açık küme tanıma ve geleneksel kapalı küme sınıflandırma problemlerinde önerilen yöntemlerin doğrusal SVM ve mevcut tek sınıf sınıflandırıcılardan daha iyi sonuçlar verdiği görülmüştür.

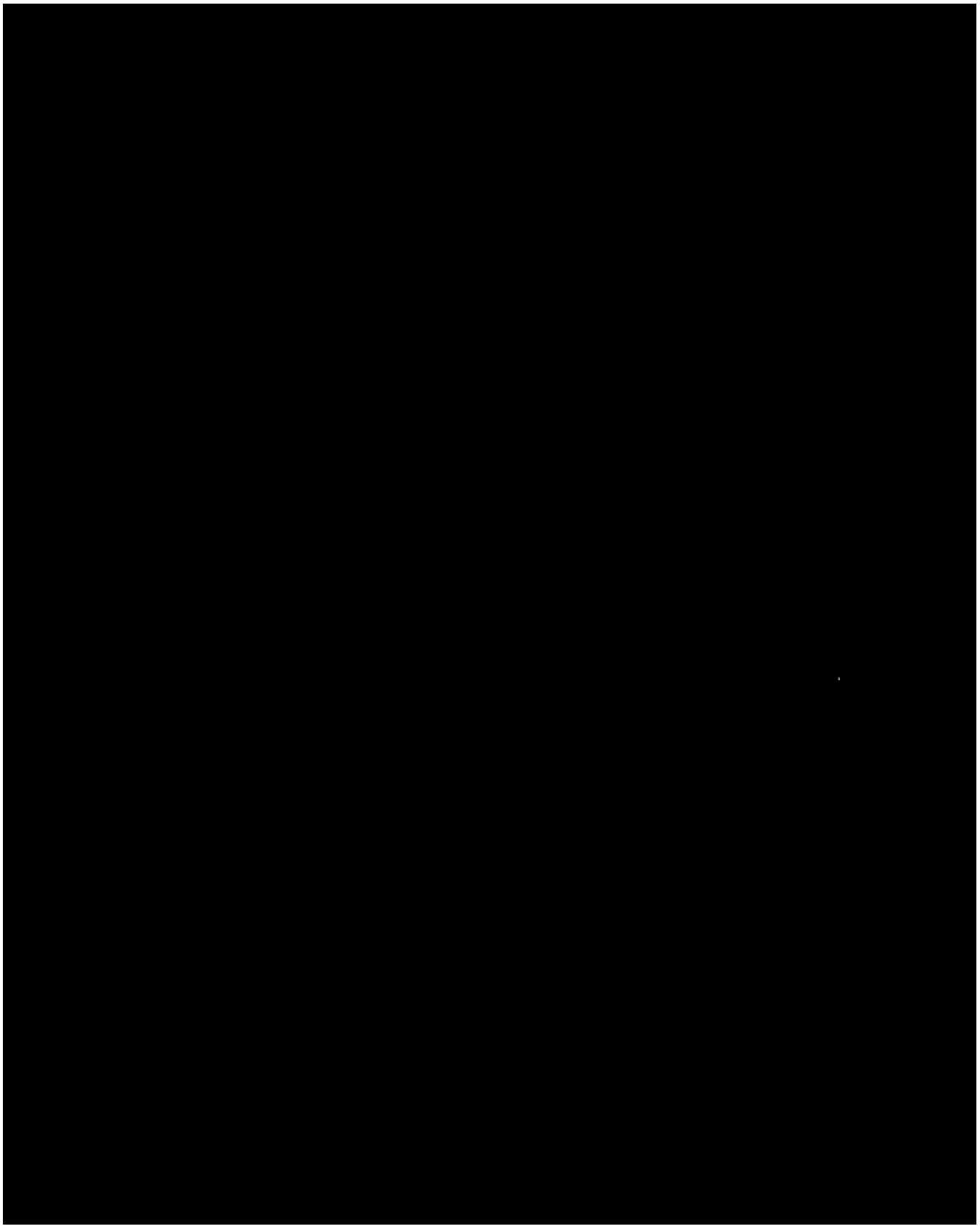
Anahtar kelimeler: Çokyüzlü konik sınıflandırıcılar, sınıflandırma, geniş marjlı sınıflandırıcı, nesne bulma, açık küme tanıma, SVM.

SUMMARY

This dissertation is based on a family of quasi-linear discriminants that outperform current large-margin methods in visual object detection and open set recognition tasks. In these applications, the classification problems are both numerically imbalanced. The positive samples are much rarer than negatives and typically form compact, visually coherent groups. On the other hand, negative data have scattered and diverse distribution. For such tasks, there is a need for discriminants whose decision regions focus on tightly circumscribing the positive class, while still taking account of negatives in zones where the two classes overlap. For this purpose, a family of quasi-linear "polyhedral conic" discriminants are proposed in this study. These proposed classifiers provide tight and compact decision boundaries by placing positive data in sub level sets of a cone. A variant of the proposed methods is offered when the positive class samples lie in different regions of the input space far from each other. The proposed methods have properties and run-time complexities comparable to linear Support Vector Machines (SVM), and they can be trained from either binary or positive-only samples using constrained quadratic programs related to SVM. Experiments show that they significantly outperform linear SVMs and existing one-class discriminants on a wide range of object detection, face verification, open set recognition and conventional closed-set classification tasks.

Keywords: Polyhedral conic classifiers, classification, large margin classifiers, object detection, open set recognition, SVM.

TEŞEKKÜR



İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

Şekil **Sayfa**

ŞEKİLLER DİZİNİ (devam)

Şekil **Sayfa**

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
1S-BFHC	1-Sided Best Fitting Hyperplane Classifier, Tek Taraflı En Uygun Hiperdüzlem Sınıflandırıcı
AP	Average Precision, Ortalama Hassasiyet
CNN	Convolutional Neural Netwoks, Evrişimli Sinir Ağları
CPM	Convex Polytope Machine, Dışbükey Polytope Makinesi
DN	Doğru Negatif
DP	Doğru Pozitif
DPO	Doğru Pozitif Oranı
EPCF	Extended Polyhedral Conic Function, Genişletilmiş Çokyüzlü Konik Fonksiyon
EPCC	Extended Polyhedral Conic Classifier, Genişletilmiş Çokyüzlü Konik Sınıflandırıcı
FDDB	Face Detection Dataset and Benchmark
FV	Fisher Vektör
GEPSVM	Generalized Eigenvalue Proximal Support Vector Machine, Genelleştirilmiş Özdeğer Proksimal Destek Vektör Makinesi
HPCAMS	Hybrid Polyhedral Conic and Max–min Separability
HOG	Histogram of Oriented Gradients, Yönlendirilmiş Gradyanların Histogramı
IPCS	Incremental Polyhedral Conic Separation, Artımlı Çokyüzlü Konik Ayırma
ICF	Incremental Conic Functions, Artımlı Konik Fonksiyon
KSVM	Kernel Support Vector Machines, Kernel Destek Vektör Makineleri
LBP	Local Binary Patterns, Yerel İkili Örüntüler
LR	Letter Recognition, Mektup Tanıma
mAP	Ortalama AP
MCPCC	Multi Center Polyhedral Conic Classifier, Çok Merkezli Çokyüzlü Konik Sınıflandırıcı
MCPCF	Multi Center Polyhedral Conic Function, Çok Merkezli Çokyüzlü Konik Fonksiyon

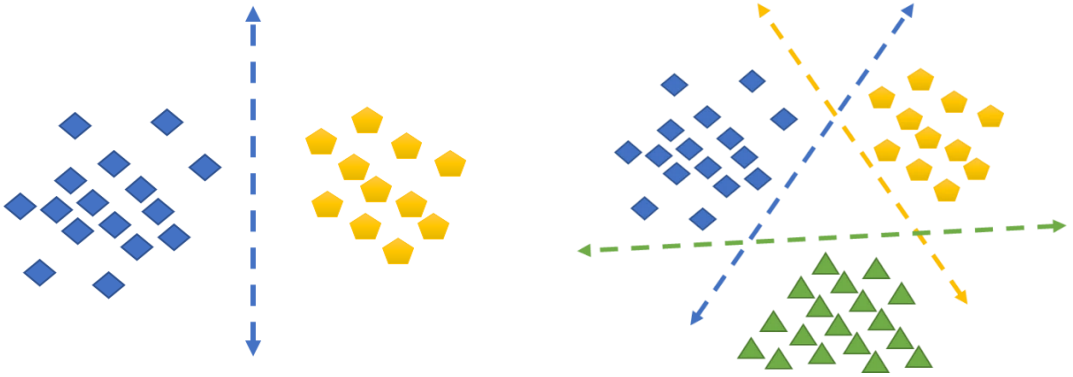
SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
MF	Multiple Features, Çoklu Öznitelikler
ML	Machine Learning, Makine Öğrenmesi
OAO	One Against One, Bire-Bir
OAR	One Against Rest, Bire-Tümü
OC-EPCC	One Class Extended Polyhedral Conic Classifier, Tek Sınıf Genişletilmiş Çokyüzlü Konik Sınıflandırıcı
O-PCF	One-Class Polyhedral Conic Function, Tek Sınıf Çokyüzlü Konik Fonksiyon
PaSC	Point-and-Shoot Face Recognition Challenge
PCC	Polyhedral Conic Classifier, Çokyüzlü Konik Sınıflandırıcı
PCF	Polyhedral Conic Function, Çokyüzlü Konik Fonksiyon
R-CNN	Region-based Convolutional Neural Networks, Bölge Tabanlı Evrişimli Sinir Ağları
SVDD	Support Vector Data Description, Destek Vektör Veri Açıklaması
SVM	Support Vector Machine, Destek Vektör Makineleri
WDBC	Wisconsin Diagnostic Breast Cancer
YN	Yanlış Negatif
YP	Yanlış Pozitif

1. GİRİŞ

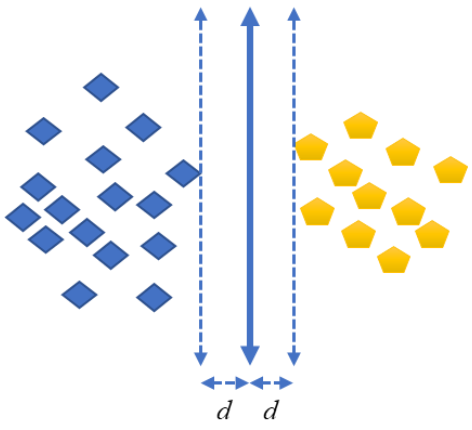
Yapay zekanın bir alanı olan makine öğrenmesi (Machine Learning, ML) son zamanlarda popülerliği oldukça artan bir konudur. Adından da anlaşılacağı üzere, makine öğrenmesinde bilgisayara sunulan veriler ile bilgisayarın istenilen bir bilgiyi öğrenmesi amaçlanmaktadır. Makine öğrenmesi, performans optimizasyonu için örnek veri ya da geçmiş deneyimleri kullanarak bilgisayar programlamaktır (Alpaydın, 2014). Makine öğrenmesi, veriden kural öğrenebilen, değişikliklere uyum sağlayabilen ve deneyim ile performansını geliştirebilen yazılımları tasarlamakla ilgilenmektedir (Blum, 2020). Günümüzde makine öğrenmesi çok farklı alanlarda kullanılmaktadır. Yüz tanıma problemlerinde, görsel olarak nesnelerin tanımlanmasında, el yazısının metne dönüştürülmesinde, metin analizinde, biyometri gibi kişilerin ölçülebilir biyolojik izlerini öğrenmede, bankalarla etkileşim kurulduğunda, sosyal medyada verimli, sorunsuz ve güvenli iletişimde makine öğrenmesinin uygulamaları görülmektedir.

Bilgisayara öğrenmesi için sunulan bilgi, veri kümesi veya veri tabanı (data set) olarak adlandırılmaktadır. Bu veri kümesinde bulunan her bir örnek ortak özniteliklere (features) sahiptir. Veri kümesindeki her örneğin bu özniteliklere karşılık gelen değerleri mevcuttur. Makine öğrenmesinde, öncelikle bilgisayar kendisine sunulan verilerin öznitelikleri arasındaki ilişkiyi ortaya çıkarıp model oluşturmakta (eğitim) ve bu modeli kullanarak benzer durumlar için tahminler (test) yürütmektedir. Bilgisayar eğitiminde veri bir kategoriye, bir sınıfa ayrılacaksa bu probleme sınıflandırma (classification) problemi, bilgisayar nümerik bir değer tahmin etmeyi öğrenecekse regresyon (regression) problemi olarak tanımlanmaktadır. Sınıflandırma probleminde eğer veriler pozitif ve negatif olarak iki sınıfa ayrılacaksa (evet/hayır, sıcak/soğuk, hasta/sağlıklı vb.) bu problem ikili sınıflandırma (binary classification) problemi olarak adlandırılmaktadır (Şekil 1.1). Eğer veriler çok daha fazla sınıfa aitse bu sınıflandırma işlemine çok sınıflı sınıflandırma (multi-class classification) problemi denir.



Şekil 1.1 (a) İkili sınıflandırma (binary classification), (b) çok sınıflı sınıflandırma (multi-class classification)

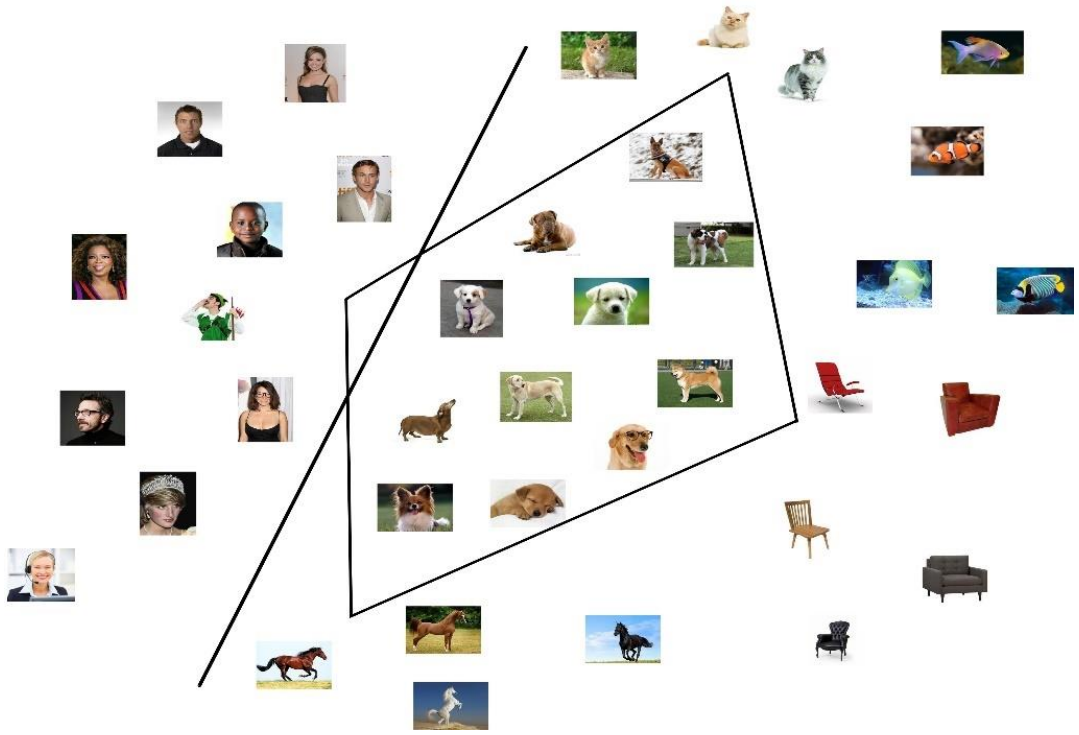
Bir sınıflandırıcı ile pozitif/negatif örnekler arasındaki mesafe marj (margin) olarak adlandırılmaktadır ve Şekil 1.2'de *d* mesafesi olarak gösterilmiştir. Bu iki sınıfı birbirinden ayırabilecek sonsuz adet doğru bulunabilir. Şekil 1.2'deki en iyi genellemeyi yapabilecek olan sınıflandırıcı, en büyük marjı sağlayan doğrudur. Bu sayede sınıflandırma hatalarının da azaltılması sağlanabilir. En büyük marjı bulmayı hedefleyen sınıflandırıcılar, geniş marjlı sınıflandırıcılar (large margin classifiers) olarak adlandırılmaktadır.



Şekil 1.2 En büyük marjı sağlayarak sınıflandırmayı hedefleyen sınıflandırıcılar geniş marjlı sınıflandırıcılar (large margin classifiers) olarak adlandırılmaktadır.

Geniş marjlı sınıflandırıcılar makine öğrenmesinde yaygın olarak kullanılmıştır. (Cortes ve Vapnik, 1995; Ertekin vd., 2011; Cevikalp ve Triggs, 2009, 2013; Cevikalp vd., 2008;). Son yıllarda derin öğrenme yöntemiyle daha başarılı sonuçlar elde edilse de bazı

derin öğrenme algoritmaları geniş marjlı sınıflandırıcılar ile birlikte kullanılmaktadır. Geniş marjlı sınıflandırıcıların en bilinen ve temel olanı Destek Vektör Makineleridir (Support Vector Machines, SVM) (Cortes ve Vapnik, 1995). SVM yöntemi, öznitelik uzayında bir hiperdüzlem ile iki sınıfı birbirinden ayırmaktadır. Bulunan hiperdüzlem maksimum marjı sağlayacak şekilde konumlanır. SVM hakkında daha geniş bilgi Bölüm 2.3'te sunulmuştur.

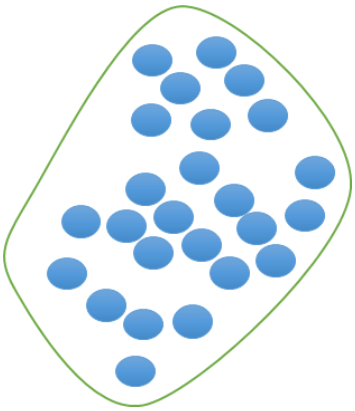


Şekil 1.3 SVM tarafından belirlenmiş hiperdüzlem pozitif sınıfı (köpek) negatif sınıftan (insan) ayırmaktadır. Bununla birlikte başka sınıflardan örnekler dışlanmamakta ve iki sınıftan birine atanmaktadır. Cevikalp ve Triggs'den (2017 a) alınmıştır.

Geniş marjlı sınıflandırıcılar dahil çoğu makine öğrenmesi sınıflandırıcısı kapalı küme (closed set) yaklaşımına göre çalışmaktadır. Kapalı küme yapısında, test sırasında sınıflandırıcının eğitimde öğrendiği sınıflar haricinde sınıf kullanılmamaktadır. Test sürecinde sınıflandırıcı sadece öğrendiği sınıflardan sorgulanmaktadır. Eğitimde hiç görmediği sınıftan bir örneği, test sırasında dışlamamakta ve eğitim örneklerinden çok uzakta olsa dahi, öğrendiği sınıflardan birine atamaktadır. Şekil 1.3'te bu problem belirtilmiştir. Şekil 1.3'te SVM tarafından belirlenmiş hiperdüzlem, pozitif sınıfı (köpek) negatif sınıftan (insan) ayırmaktadır. Bununla birlikte başka sınıflardan örnekler dışlanmamakta ve yakınlığına göre iki sınıftan birine atanmaktadır. Gerçekte ise eğitim sırasında olabilecek tüm sınıfları hesaba katabilmek olanaksızdır. Bununla beraber bazen

sadece belirli bir sınıfa odaklanılması da gerekebilmektedir. Örneğin görsel nesne sınıflandırma (visual object classification) problemlerinde belirli bir sınıfa ait örnekler ön plana çıkmakta diğer her şey arka plan olarak tanımlanmaktadır. Benzer olarak, yüz doğrulama problemlerinde bir kişiye ait yüz örnekleri pozitif, diğer tüm yüzler negatif olarak tanımlanmaktadır. Her iki durumda da eğitim sırasında negatif olabilecek bütün sınıfların önceden bilinmesi olanaksızdır.

Scheirer vd.nin (2013) tanımladığı açık küme tanıma (open set recognition) yaklaşımında, eğitim sırasında bazı sınıflar öğrenilir, fakat test sırasında öğrenilmemiş sınıflardan örnekler de kullanılmaktadır. Bu yaklaşımda sınıflandırıcının öğrenilmemiş sınıflara ait örnekleri dışlaması beklenmektedir. Klasik marj tabanlı sınıflandırıcılarda bilinmeyen sınıfa ait örnekler, pozitif örneklerden çok uzakta olsa bile, başka bir karar sınırı (decision boundary) olmadığından bir sınıfa dahil edilebilmektedir. Bundan dolayı marj tabanlı sınıflandırıcılar açık küme yaklaşımında hatalar yapabilmektedir (Şekil 1.3). Bu tarz sorunların üstesinden gelmek amacıyla tek sınıf sınıflandırıcılar (one class classifiers) ön plana çıkmıştır. Tek sınıf sınıflandırıcılar, negatif örnekler olmasa dahi, karar sınırlarıyla pozitif örnekleri sıkıca çevrelemeyi amaçlamaktadır (Şekil 1.4). Bu sayede bilinmeyen sınıfa ait örnekleri daha iyi dışlayabilmektedir.



Şekil 1.4 Tek sınıf sınıflandırıcılar, negatif örnekler olmasa dahi, karar sınırlarıyla pozitif örnekleri sıkıca çevrelemeyi amaçlamaktadır.

Tek sınıf sınıflandırıcı olarak, Tax ve Duin (2004) Destek Vektör Veri Açıklaması (Support Vector Data Description, SVDD) yöntemini önermiş olup burada pozitif örnekleri çevreleyen bir hiperküre bulmayı amaçlamıştır. Cevikalp ve Triggs (2017 b) yüz ve insan

tespiti için, kademeli bir şekilde pozitif bölgeleri negatif bölgelerden ayıran dışbükey modeller önermektedir. Mangasarian ve Wild (2006) tarafından önerilen Genelleştirilmiş Özdeğer Proksimal Destek Vektör Makinesi (Generalized Eigenvalue Proximal Support Vector Machine, GEPSVM) yöntemi, pozitif örnekleri en iyi şekilde ayırırken, negatif örneklerden de olabildiğince uzak olan bir hiperdüzlem bulmaktadır. Benzer şekilde, Jayadeva vd.de (2007) İkiz Destek Vektör Makinesi (Twin Support Vector Machine, TSVM) yöntemini önermiştir. Bu sınıflandırıcı da en iyi yerleşen bir hiperdüzlemi, kuvadratik programlama probleminin çözümüyle bulmaktadır. Cevikalp (2016) de en iyi yerleşen hiperdüzlemi bulmaya çalışan bir yöntem önermiştir. Vedaldi ve Zisserman (2012) tarafından önerilen Eklemeli Çekirdek (Additive Kernel) ile Rahimi ve Recht'in (2007) Rastgele Öznitelik (Random Features) yöntemleri, örnekleri daha büyük boyutlu uzaya dönüştürerek doğrusal olmayan bir ayrıma imkân sağlamakta ve pozitif örnekleri çevrelemektedir. Manwani ve Sastry (2010) de çokyüzlü karar sınırlarını lojistik fonksiyon kullanarak öğrenen bir yöntem önermiştir. Burada oluşturulan çokyüzlü karar sınırı birden fazla hiperdüzlemin kesiştirilmesiyle oluşturulmaktadır. Yöntemde hiperdüzlem sayısının eğitimden önce belirlenmesi gerekmektedir. Bu ise bu yöntemin en büyük kısıtıdır. Bununla birlikte çalışmadaki deneylerde küçük ölçekli veri tabanları kullanılmıştır. Kantchelian vd. (2014) ise Dışbükey Polytope Makinesi (Convex Polytope Machine, CPM) yöntemiyle büyük ölçekli veri tabanlarında çokyüzlü karar sınırı bulmaktadır. Bu yöntem Manwani ve Sastry'den (2010) bir adım öteye geçerek sabit sayıda hiperdüzlem ile pozitifleri ve negatifleri ayırabilmesine karşın her zaman çokyüzlü karar sınırının oluşturulmasını garanti edememektedir. Benzer şekilde pozitif örnekleri çevreleyen çokyüzlülerin önerildiği yöntemler de mevcuttur (Dundar vd., 2008; Bagirov vd., 2013). Bahsedilen bu yöntemlerde, eğitim kümesindeki örnek sayısının artmasıyla hesaplama yükü artmakta, lokal optimum ve aşırı öğrenme oluşabilmekte, ilave olarak kümeleme veya etiketleme gerekebilmektedir. Bu durum ise büyük ölçekli uygulamalar için uygun olmamaktadır.

Tek sınıf sınıflandırıcılar için önerilen en dikkat çekici çalışma ise Gasimov ve Ozturk (2006) tarafından yapılmıştır. Bu çalışmada Çokyüzlü Konik Fonksiyonlar (Polyhedral Conic Functions, PCF) kullanılarak karar sınırları oluşturulmuştur. Pozitif örnekler, PCF ile verilen koninin seviye kümelerine yerleştirilmeye çalışılmaktadır. Bu sayede kapalı, pozitif örnekleri çevreleyen sıkı karar sınırı elde edilebilmektedir. Sıkı ve

kapalı karar sınırı sayesinde hem açık küme hem kapalı küme problemlerinde SVM gibi yarı uzaylı karar sınırına sahip sınıflandırıcılardan daha başarılı olmaktadır.

Gasimov ve Ozturk'ün (2006) çalışmasında kullanılan doğrusal programlama yöntemi, PCF sınıflandırıcısının büyük öznitelik vektörü kullanan bazı büyük veri tabanlarında çalışmasını engellemektedir. Ayrıca hata-ceza yaklaşımı ile genellemeye gitmek yerine çok sınıflandırıcılı bir yapıya sahip olması aşırı öğrenmeye neden olabilmektedir. Bu tez çalışmasında bahsedilen sorunların çözümü amacıyla Çokyüzlü Konik Sınıflandırıcı (Polyhedral Conic Classifier, PCC) oluşturulmuştur. PCF'te yapılan değişiklikler ile de Genişletilmiş Çokyüzlü Konik Fonksiyon (Extended Polyhedral Conic Function, EPCF) geliştirilmiş ve EPCF'i kullanan Genişletilmiş Çokyüzlü Konik Sınıflandırıcı (Extended Polyhedral Conic Classifier, EPCC) önerilmiştir. PCC ve EPCC yöntemleri ile kapalı ve sıkı karar sınırları elde edilebilmektedir. Bu sayede pozitifler negatiflerden etkili bir şekilde ayrılabilmekte ve bütün negatif sınıfların önceden bilinmesine gerek kalmadan negatif örnekler dışlanabilmektedir. Çalışmada PCC ve EPCC yöntemleri doğrusal SVM denklemine dönüştürülerek SVM için kullanılan tüm matematiksel yaklaşımların ve yazılımların da kullanılabilmesi sağlanmaktadır. Bu sayede önerilen yöntemler için çok hızlı ve etkili eğitim ve test süreçleri kullanılabilmektedir. Herhangi bir çekirdek hilesi (kernel trick) kullanmadan, önerilen yöntemler SVM'in çekirdek hilesi kullanan kompleks yöntemleri kadar ve daha iyi sonuçlar göstermektedir. Bunu gerçekleştirirken doğrusal SVM hızına yakın eğitim ve test sürelerini de korumaktadır.

Önerilen yöntemlerle PCF ve EPCF fonksiyonlarının bazı durumlarda karar sınırını kapalı olarak oluşturamadığı gözlemlenmiştir. Bu sorunu gidermek amacıyla, her zaman kapalı bir karar sınırı sağlayan Tek Sınıf Genişletilmiş Çokyüzlü Konik Sınıflandırıcı (One Class Extended Polyhedral Conic Classifier, OC-EPCC) anlatılmıştır (Cevikalp ve Triggs, 2017 a). OC-EPCC ile negatif eğitim örnekleri olmasa dahi kapalı karar sınırları elde edilebilmektedir. Bu durum ise öngörülemeyen negatif örneklere karşı çok iyi bir yapıya sahip olmasını sağlamaktadır.

Diğer karşılaşılan bir problem ise pozitif örneklerin aynı uzayda farklı bölgelerde kümelenmesidir. Örneğin bir arabaya ait yandan, üstten, arkadan ve önden görünümler öznitelik uzayında farklı bölgelerde kümelenebilmektedir. Bu durumda her bir küme için

ayrı bir sınıflandırıcı kullanılması gerekir. Birden fazla sınıflandırıcı, eğitim ve test sürecini hem yavaşlatabilmekte hem de karmaşık hale getirebilmektedir. Bu kapsamdaki problemleri çözmek amacıyla hem PCF basitliğini koruyan hem de tek bir sınıflandırıcı kullanan Çok Merkezli Çokyüzlü Konik Sınıflandırıcı (Multi Center Polyhedral Conic Classifier, MCPCC) geliştirilmiştir. Bu sınıflandırıcı ile yapılan sentetik ve gerçek veri tabanı deneylerinde etkili bir karar sınırının elde edilebildiği görülmüştür. MCPCC'nin, birkaç sınıflandırıcı ile elde edilebilecek karar sınırlarını tek bir sınıflandırıcı kullanarak elde ettiği ve başarılı sonuçlar verdiği görülmüştür.

Çalışmada önerilen yöntemlerin başarısını incelemek amacıyla büyük ve küçük ölçekteki veri tabanlarında testler yapılmıştır. Ayrıca nesnelerin farklı yöntemlerle elde edilen öznitelikleri ile yöntemlerin başarısı incelenmiştir. Önerilen sınıflandırıcılar; nesne bulma (object detection), görsel nesne sınıflandırma, yüz doğrulama (face verification), açık küme tanıma problemlerinde incelenmiş ve popüler başka sınıflandırıcılar ile karşılaştırılmıştır.

Tezin 2. bölümünde önerilen yöntemlerin daha iyi anlaşılabilmesi için bazı teorik bilgiler hatırlatılmıştır. 3. bölümde konik fonksiyonlar ile ilgili şimdiye kadar yapılan çalışmalara ait literatür araştırmasına yer verilmiştir. 4. bölümde önerilen yöntemlerin matematiksel altyapısı anlatılmıştır. 5. bölümde ise önerilen yöntemlerin başarısı farklı problemlerde incelenmiştir. Bu kapsamda farklı veri tabanlarında deneyler yapılmış ve benzer yöntemlerle karşılaştırılmıştır. Elde edilen sonuçlar ışığında 6. bölümde tez sonuçlandırılmıştır.

Tezin metnindeki terimler Beşer'in (2020) hazırladığı yapay zekâ sözlüğü ve Türkiye Bilimler Akademisi (TÜBA, 2013) tarafından hazırlanan Türkçe Bilim Terimleri Sözlüğü ile uyumlu olarak kullanılmıştır. Matematiksel ifadelerde nümerik değişkenler eğik harflerle ("n" gibi), vektör değişkenler küçük kalın harflerle ("$\mathbf{x}$" gibi), matrisler büyük kalın harfle ("$\mathbf{A}$" gibi), kümeler büyük harfle ("S" gibi) gösterilmiştir.

2. TEORİK BİLGİ

Bu bölümde önerilen yöntemlerin daha iyi anlaşılabilmesi için genel kavramlar ve yöntemler anlatılmıştır. Bu kapsamda makine öğrenimi konusunda açıklayıcı bilgi sunulmuştur (Bölüm 2.1). Ayrıca Bölüm 2.2'de destek vektör makineleri detaylı olarak açıklanmıştır. Tezin temelini oluşturan çokyüzlü konik fonksiyonların matematiksel altyapısı da Bölüm 2.3'te anlatılmıştır. MCPCC sınıflandırıcısında kullanılan k-merkezli kümeleme (k-means clustering) yöntemi Bölüm 2.4'te açıklanmıştır. Deneylerde kullanılan performans kriterlerinin detayları da bölüm 2.5'te sunulmuştur. Bu bölümde k-çapraz katman doğrulama yöntemi ve performans ölçütleri anlatılmıştır.

2.1 Makine Öğrenmesi

2.1.1 Makine öğrenmesi nedir?

Yapay zekanın bir alt kümesi olan makine öğrenmesinde bilgisayarın öğrenmesine yardımcı olmak için matematiksel modeller kullanılmaktadır. Makine öğrenmesi, performans optimizasyonu için örnek veri ya da geçmiş deneyimleri kullanarak bilgisayar programlamaktır (Alpaydın, 2014). Makine öğrenmesi, veriden kural öğrenebilen, değişikliklere uyum sağlayabilen ve deneyim ile performansını geliştirebilen yazılımları tasarlamakla ilgilenmektedir (Blum, 2020). Makine öğrenmesinde bilgisayar kendisine sunulan verilerdeki ilişkiyi ortaya çıkarıp model oluşturmakta ve bu modeli kullanarak benzer durumlar için tahminler yürütmektedir. Çok veri ile daha iyi öğrenebilmekte ve daha doğru tahminler yapabilmektedir. Verilerin, ihtiyaçların, görevlerin sürekli değiştiği durumlarda (gereksiz e-postaların tespiti, müşteri davranışlarının tahmini, hasta izleme ve salgın tahmini, trafik anormalliği belirleme vb.) veya klasik kodlamanın imkânsız olduğu durumlarda makine öğrenmesinin kolay şekilde uyum sağlaması, makine öğrenmesinin çok tercih edilmesini sağlamaktadır.

Makine öğrenmesi ile eğitilen bir bilgisayar eğer iki sınıf için eğitildiyse gelen yeni örneği bu iki sınıftan birine atamaktadır. Örneğin, gereksiz cep telefonu mesajlarını (SMS) tespit eden bir sınıflandırıcıyı yeterince veri ile eğitmiş olalım. Eğitilmiş model ile yeni gelen

bir mesajın gereksiz mesaj mı yoksa önemli mesaj mı olduğuna karar verilebilmektedir. İki sınıf sınıflandırma yerine çok sınıflı sınıflandırma ile de bilgisayar eğitilebilmektedir. Örneğin motorlu araçların resimleri kullanılarak eğitilen bir bilgisayar ile yeni gelen bir resmin araba, kamyon, motosiklet, tren sınıflarından birine sınıflandırılabilmesi sağlanır. Makine öğrenimi ile yeni gelen örneklerin kategorilere ayrılabilmesinin yanında süreklilik gösteren değerlerin tahmin edilebilmesi de mümkündür. Örneğin bir semtteki ev özellikleri ile fiyat ilişkisini kuran bir bilgisayar modelini eğitmek için bu semtteki ev satış verilerini kullanalım. Eğitilen model ile "site içinde 2. kattaki garajlı, 100 m^2, 2+1" bir evin fiyatı tahmin edilebilmektedir. Burada beklediğimiz model çıktısı herhangi bir sınıflandırmadan ziyade bir nümerik değer olmaktadır. Bu tarz makine öğrenimi teknikleri regresyon problemlerine örnektir.

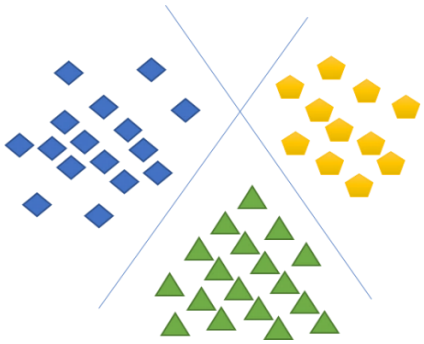
Bilgisayar teknolojisinin gelişmesiyle çok büyük çaplı veriler kaydedilip saklanabilmektedir. Tüm bu veriler analiz edilip işlenirse anlamlı bilgiye dönüşmektedir. Makine öğrenimi ile büyük çaptaki veriler işlenmekte ve kullanılabilir basit modeller oluşturularak veriler anlamlı bilgiye dönüştürülebilmektedir (Alpaydın, 2014). Örneğin bir müşterinin alışveriş tercihleri vardır, markete gittiği zaman rastgele alışveriş yapmaz. Müşteriler ile aldıkları arasında bir ilişki mevcuttur. Makine öğreniminde kümeleme yöntemleri kullanılarak, müşteriler kategorize edilebilmekte ve aynı kategorideki müşterilerin tercihleri öngörülebilmektedir. Bu sayede pazarlama stratejileri daha etkin ve etkili şekilde geliştirilebilmektedir. Bununla birlikte makine öğrenmesi kullanıcıların özelliğine göre özel hedefli içeriklerin sunulmasında, otomatik sesli sanal yardımcılarda, müşteri deneyimini iyileştirmede de kullanılmaktadır.

Sürekli değişiklik gösteren dolandırıcılık taktiklerine karşı makine öğrenmesi kolaylıkla uyum sağlayabilmektedir. Makine öğrenmesi sayesinde olağan dışı durumlar tespit edilebilmekte (anomaly detection) ve dolandırıcılık denemeleri başarılı olmadan engellenebilmektedir. Makine öğrenmesinin olağan dışı durumları tespit edebilmesi ile malzeme hatalarının tespiti, yapısal bozuklukların belirlenmesi, yazım hatalarının bulunması da sağlanabilmektedir. Genel olarak makine öğrenmesi; yüz tanıma problemlerinde, görsel olarak nesnelerin tanımlanmasında, el yazısının metne dönüştürülmesinde, metin analizinde, arama motorlarında, tıbbi teşhis koymada, doğal dil işlemede, müşteri davranışlarının belirlenmesinde ve buna benzer birçok alanda kullanılmaktadır.

2.1.2 Makine öğrenmesi teknikleri

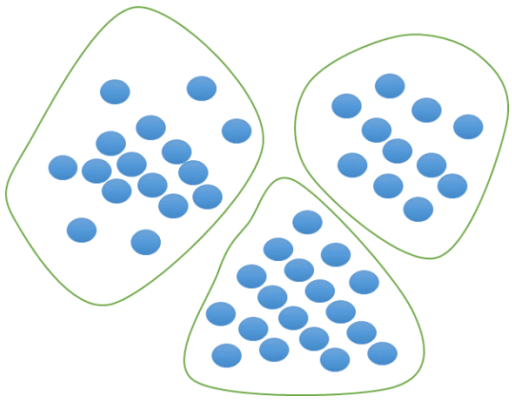
Gözetimli öğrenme (supervised learning): Gözetimli öğrenmede eğitim için sonuçları bilinen (etiket bilgisi veya çıktı değeri olan) veriler kullanılmaktadır. Bu verilerle bilgisayar eğitilir ve bir model oluşturulur. Bu model sayesinde yeni gelen örnekler için bir değer bulunur veya sınıflandırılır (Şekil 2.1). Örneğin şehirlerin nüfusu ile ilgili 50 yıllık veri elimizde olsun. 5 yıl sonra belli bir şehrin nüfusunu tahmin etmek istiyoruz. Bunun için elimizdeki veriler ile (girdi: şehir, yıl, ölüm oranı, doğum oranı, göç miktarı, kişi başı gelir; çıktı: nüfus) eğitim yapılarak bir model oluşturulmaktadır. Sistem girdileri daha da fazla olabilir. Makine öğrenmesi ile bu girdilerle nüfus arasındaki ilişki bilgisayar tarafından öğrenilmekte ve uygun bir model oluşturulmaktadır. Bu model kullanılarak girdilerdeki değişikliklere göre 5 yıl sonra o şehrin nüfusu tahmin edilebilmektedir. En çok bilinen gözetimli öğrenme algoritmaları aşağıdaki gibidir:

- Destek vektör makinesi
- Doğrusal regresyon
- Lojistik regresyon
- Naif Bayes sınıflandırıcı
- Doğrusal ayırma analizi
- Karar ağaçları
- *k*-en yakın komşu algoritması
- Yapay sinir ağları
- Derin öğrenme (Evrişimli sinir ağları, CNN)



Şekil 2.1 Gözetimli öğrenmede sınıflara ait etiket bilgisi bulunmaktadır. Gözetimli öğrenme ile sınıfları birbirinden ayıran karar sınırları elde edilir.

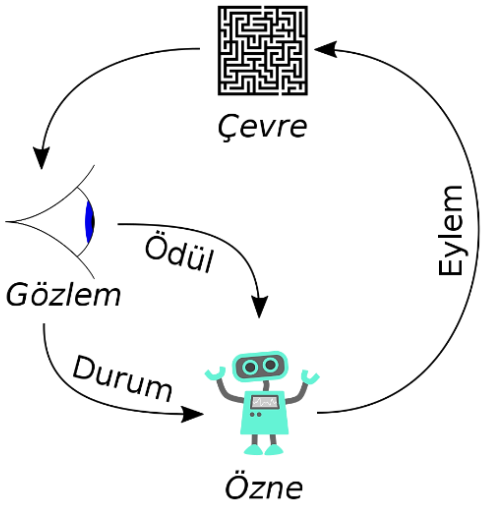
Gözetimsiz öğrenme (unsupervised learning): Gözetimsiz öğrenmede denetimli öğrenmede olduğu gibi etiket veya sınıf bilgisi yoktur, sadece veriler mevcuttur. Buradaki amaç veri içerisindeki düzeni bulmaktır. Sınıflar önceden bilinmediğinden ve sınıflandırmaya yönelik bir eğitim yapılamayacağından; verilerin birbirine olan mesafesi, komşuluk ilişkileri ve yoğunlukları kullanılarak verilerle ilgili çıkarımlar yapılır. Gözetimsiz öğrenmede iki yaklaşım kullanılır: boyut indirgeme ve kümeleme. Boyut indirgemede verilerin boyutlarının veriler üzerindeki etkisi incelenmekte ve verileri birbirinden ayırabilecek en yararlı boyutlar öğrenilmektedir. Kümeleme analizinde ise sınıflandırılmamış, kategorize edilmemiş, etiketlenmemiş verilerin gruplandırılması amaçlanır (Şekil 2.2). Örneğin, gazete haberlerinin kategorize edilmesinde, sözlükteki kelimelerin haberlerdeki kullanımın sıklığına göre, benzer kelimeleri kullanan haberler aynı kategoriye ayrılmaktadır.



Şekil 2.2 Gözetimsiz öğrenmede etiket veya sınıf bilgisi yoktur, sadece veriler mevcuttur. Buradaki amaç veri içerisindeki düzeni bulmaktır.

Pekiştirmeye dayalı öğrenme (reinforcement learning): Bazı uygulamalarda sistemin çıktısı bir dizi eylemdir. Böyle durumlarda tek bir eylemden ziyade belirli eylemleri doğru şekilde tercih ederek hedefe ulaştıran politikalar önemlidir. Bilgisayar, oluşturulan politikaların başarısını değerlendirebilmeli ve geçmişte uyguladığı doğru eylem dizilerinden öğrenerek yeni politikalar üretebilmelidir. Bu tarzdaki öğrenme yöntemi pekiştirmeye dayalı öğrenme olarak adlandırılmaktadır (Alpaydın, 2014). Pekiştirmeye dayalı öğrenmede bilgisayar uyguladığı eylemler karşısında aldığı geri bildirimlerle (doğru, yanlış, nötr) en doğru eylemler dizisini öğrenmeye ve en yüksek ödülü almaya çalışır. Temel pekiştirmeli öğrenme modeli aşağıdakilerden oluşur:

1. Öznenin ve ortamın durumlarını içeren bir S kümesi,
2. Öznenin yapabileceği eylemleri içeren bir A kümesi,
3. Her durumda hangi eyleme geçileceğini belirleyen politika,
4. Bir durum geçişinin kazandıracağı skaler anlık ödülü hesaplamak için kurallar
5. Öznenin gözlemlerini betimlemek için kurallar (Anonim, 2020 b).



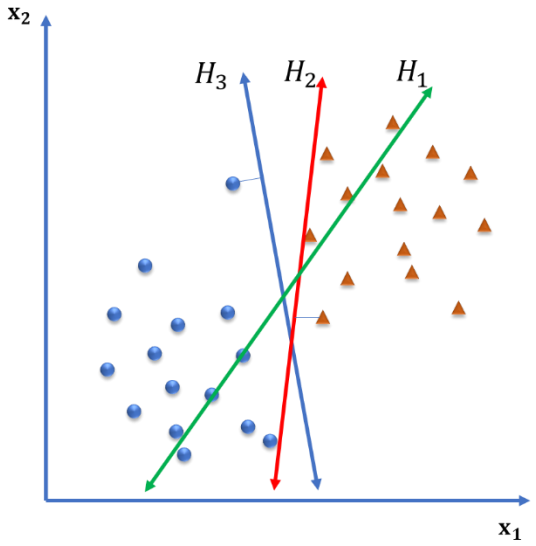
Şekil 2.3 Pekiştirmeli öğrenmedeki öğrenme süreci. (Anonim, 2020 b)

Pekiştirmeli öğrenmede özne (bilgisayar) çevrede bir eylem gerçekleştirir. Bu eylem karşısında aldığı ödül ile durum değişiklikleri gözlemlenir ve özne tarafından değerlendirme yapılır. Bilgisayar bu döngüler ile en iyi eylemler dizisini öğrenir.

Tez çalışmasındaki çokyüzlü konik sınıflandırıcılar, yukarıda bahsedilen yöntemlerden gözetimli öğrenme yöntemine bir örnektir. Etiket bilgisi olan veri tabanları ile model eğitilmekte ve test örnekleri bu etiketler ile sınıflandırılmaktadır.

2.2 Destek Vektör Makinesi

En bilinen sınıflandırma yöntemlerinden biri Destek Vektör Makinesidir (Cortes ve Vapnik, 1995). Gözetimli öğrenme yöntemlerinden olan SVM, daha çok sınıflandırma uygulamalarında kullanılsa da regresyon uygulamalarında da kullanılabilmektedir. SVM ile iki sınıflı eğitim kümesi kullanılarak doğrusal bir sınıflandırıcı oluşturulmaktadır.



Şekil 2.4 İki sınıfın sınıflandırılması

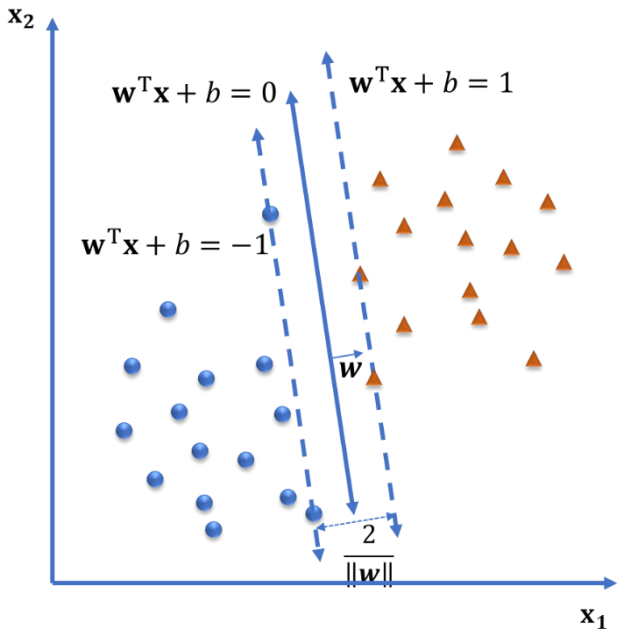
Şekil 2.4'te verilen iki boyutlu düzlemde, iki ayrı sınıfa ait örnekler $(\mathbf{x}_i, y_i)$, $\mathbf{x}_i \in \mathbb{R}^2$ için, pozitif örnekler $y_i = +1$, negatif örnekler $y_i = -1$ olarak tanımlanmıştır. Bu iki sınıfı birbirinden ayıran sonsuz sayıda doğru bulunabilir. Şekil 2.4'te örneklerin sınıflandırılmasında H_1 doğrusu iki sınıfı ayıramazken H_2 ve H_3 doğruları ayrımı yapabilmektedir. H_3'ün, sınıfları diğerlerinden daha etkili bir şekilde ayırdığı gözlemlenmektedir. H_3'ün başarısı, iki sınıfın örnekleriyle arasındaki mesafeyi, marjı, büyütmesinden ileri gelmektedir. Böylelikle daha iyi bir genelleme yapabilmektedir.

Şekil 2.4'te verilen iki boyutlu sınıflandırma probleminden farklı olarak, sınıflandırma problemlerinde öznitelik uzayı çok daha büyük boyutlardan oluşabilmektedir. Çok boyutlu bir uzayda ise SVM, denklem (2.1) ile verilen d boyutlu bir hiperdüzlem bularak iki sınıfı birbirinden ayırmaktadır. Hiperdüzlem denkleminde $\mathbf{x}_{test}$ örneği $\mathbf{w}^{\mathrm{T}}\mathbf{x}_{test} + b \geq 0$ ise pozitif bölgede, $\mathbf{w}^{\mathrm{T}}\mathbf{x}_{test} + b < 0$ ise negatif bölgede yer almaktadır. Doğrusal SVM, en uygun hiperdüzlemi bulmak için en büyük ayırma marjını sağlayan $\mathbf{w}$ ve b parametrelerinin bulunmasını hedefler.

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0\,,\ \mathbf{x}, \mathbf{w} \in \mathbb{R}^d,\ b \in \mathbb{R} \qquad (2.1)$$

Şekil 2.5'teki gibi iki sınıfa ait örnekler doğrusal olarak ayrılabiliyorsa, bu ayırma düzlemine paralel ve yine iki sınıfı ayırabilen iki düzlem bulabiliriz. Bu iki düzlem denklemi

$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 1$ ve $\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = -1$ olarak gösterilebilir. Marj sınırlarını oluşturan bu iki hiperdüzlem, Şekil 2.5'te görüldüğü gibi, ayırdığı sınıflara ait bazı örneklerden geçmektedir. Bu örnekler destek vektörleri (support vectors) olarak adlandırılmaktadır. Destek vektörleri kullanılarak hiperdüzlem denklemi elde edilebilmektedir. Destek vektörleri dışında kalan örnekler marj hiperdüzlem denklemini oluşturmak için gerekli değildir.



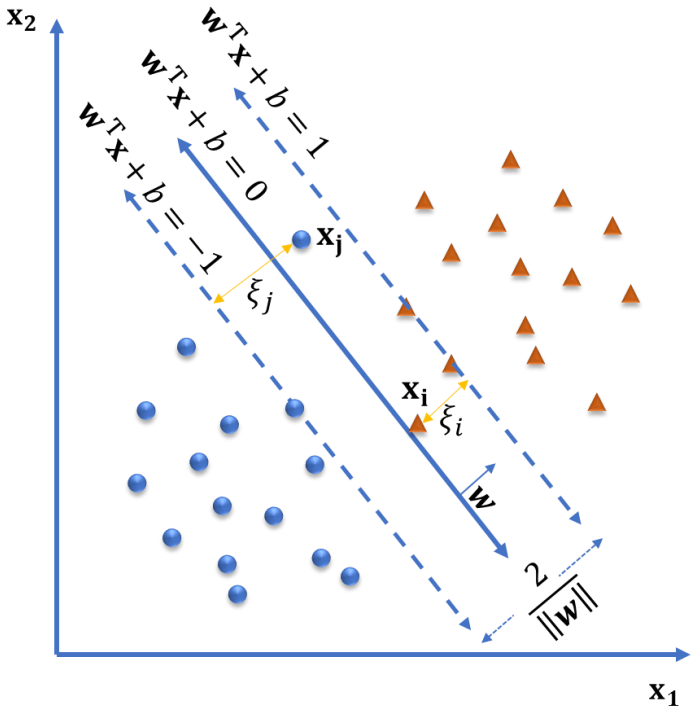
Şekil 2.5 Katı marj SVM sınıflandırma

Marj sınırlarını oluşturan bu iki hiperdüzlem arasındaki uzaklık $2/\|\mathbf{w}\|$ olarak hesaplanır. Bu marjın maksimuma çıkarılması aynı zamanda $\|\mathbf{w}\|$'nun minimize edilmesi demektir. Bu optimizasyon problemini çözerken her bir pozitif ve negatif $\mathbf{x}_i$ örneğinin marjın uygun tarafına yerleştirilmesi, denklem (2.2) ve (2.3)'teki kısıtlar ile zorlanır:

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b} \geq 1 \text{ , eğer } \mathrm{y}_{\mathrm{i}} = +1 \tag{2.2}$$

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b} \leq -1 \text{ , eğer } \mathrm{y}_{\mathrm{i}} = -1 \tag{2.3}$$

Yukarıdaki denklem (2.2) ve (2.3) kısıtları $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1$ olarak da birleştirilebilmektedir.



Şekil 2.6 Yumuşak marj SVM sınıflandırma

Şekil 2.5'te katı marj (hard margin) yaklaşımı kullanılmaktadır. Bu yaklaşımda her bir örneğin hiperdüzlemle bölünen uzayın ilgili bölümünde olması zorlanır. Hiperdüzleme yakın bazı örneklerin, belirli bir hata payı ile, bu kuralı bozması müsaade edilirse daha iyi bir genelleme sağlayan daha geniş bir marj elde edebiliriz. Bu sayede doğrusal olarak ayrılamayan örnekleri de bir hiperdüzlemle ayırma olanağı elde etmiş oluruz. Belli bir hataya müsaade edilmesiyle, Şekil 2.6'daki gibi daha iyi bir genelleme yapabilen doğrusal bir SVM hiperdüzlemi elde edilebilir. Bu yaklaşım yumuşak marjlı (soft margin) sınıflandırma olarak adlandırılmaktadır. Yumuşak marj yaklaşımında artıran değişken (slack variable), $\xi_i = \max(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))$, devreye girmektedir. Artıran değişken, ξ_i, denklem (2.2) ve (2.3)'t eki kısıtların sağlanması durumunda 0 olmakta, diğer durumlarda hata ölçüsünde değer almaktadır. Diğer bir ifadeyle artıran değişken, $\mathbf{x}_i$ örneği marjın doğru bölgesindeyse 0, yanlış bölgesindeyse marja olan uzaklığı kadar hata değeri almaktadır.

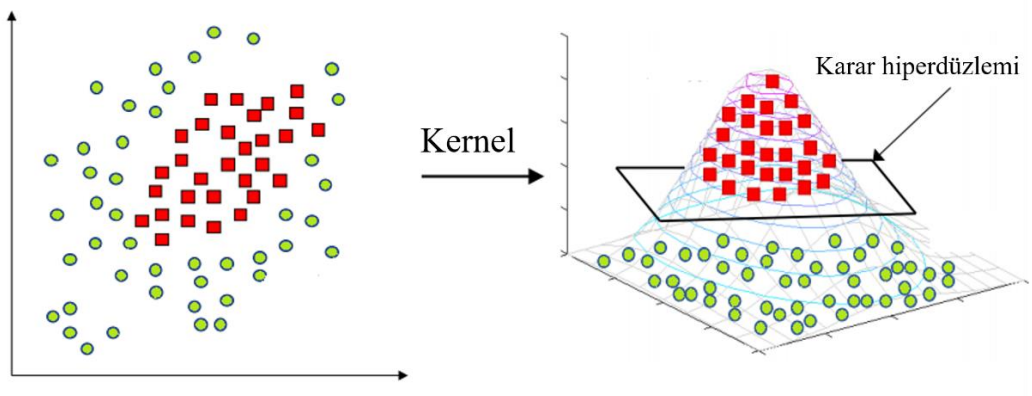
Katı marj SVM'de sadece $\|\mathbf{w}\|$ minimize edilirken, yumuşak marjlı SVM çözümü için $\|\mathbf{w}\|$ ve toplam ξ'nin minimize edilmesi gerekir. $\|\mathbf{w}\|$'yi minimize etmek, aynı zamanda $\mathbf{w}^T\mathbf{w}$ 'yi de minimize etmektir. Bu sayede mutlak değerli terimi kaldırmış oluruz. Yumuşak

marjlı SVM çözümü için kuvadratik programlama ile elde edilen minimum problemi denklem (2.4)'teki gibi olur:

$$\begin{aligned} &\min_{\mathbf{w},b} \ \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C_{+}\sum_i \xi_i + C_{-}\sum_j \xi_j \\ &s.t.\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b + \xi_i \geq +1,\ i \in \mathrm{I}_{+}, \\ &\mathbf{w}^{\mathrm{T}}\mathbf{x}_j + b - \xi_j \leq -1,\ j \in \mathrm{I}_{-}, \\ &\xi_i, \xi_j \geq 0 \end{aligned} \tag{2.4}$$

Denklem (2.4)'te I_{+} pozitif örnek indisleri, I_{-} negatif örnek indisleri, ξ_i pozitif örneklerin artıran değişkeni, ξ_j negatif örneklerin artıran değişkeni, C_{+} ve C_{-} sırasıyla pozitif ve negatif artıran değişkenin $\mathbf{w}^{\mathrm{T}}\mathbf{w}$ terimine göre baskınlığını dengeleyen parametreyi ifade eder. C parametresinin artması ile artıran değişkenin küçültülmesi ön plana çıkmaktadır. Bu ise katı marjlı probleme yaklaşılması demektir. C parametresinin düşürülmesi ise artıran değişkenin daha kabul edilebilir olduğu anlamına gelmekte ve daha genel bir çözüme olanak sağlamaktadır. C_{+} ve C_{-} parametreleri pozitif ve negatif örnekler için aynı alınabileceği gibi problemin özelliğine göre farklı da seçilebilmektedir.

Doğrusal olarak ayrılamayan durumlarda yumuşak marjlı yaklaşım işe yarasa da daha karmaşık sınıflandırmanın yapılabilmesi amacıyla çekirdek hilesi (kernel trick) kullanılmaktadır. Çekirdek hilesinde, örneklerin bulunduğu uzaydan daha yüksek boyutlu bir öznitelik uzayına dönüşümle veya doğrusal olmayan bir dönüşümle sınıflandırma amaçlanmaktadır. Dönüşüm sağlanan uzayda örneklerin ayrımı yine o uzaydaki bir hiperdüzlem ile sağlanmaktadır (Şekil 2.7).



Şekil 2.7 Çekirdek hilesiyle örneklerin başka bir uzaya dönüştürülmesi (Anonim, 2020 a).

Denklem (2.4)'teki SVM kuvadratik çözümü, asal probleme (primal problem) ait bir çözümdür. Çekirdek hilesini kullanabilmek için denklem (2.5)'teki Lagrange eşiz problemi (dual problem) kullanılmaktadır.

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \, \alpha_j y_i y_j \mathbf{x}_i . \mathbf{x}_j \\ \text{s.t.} \quad & 0 < \alpha_i \leq C \\ & \sum_i \alpha_i \, y_i = 0 \\ & i \in \mathrm{I_{DV}} \text{ , } (\mathrm{I_{DV}}\text{, destek vektörler indisleri}) \end{aligned} \tag{2.5}$$

α_i her bir destek vektörüne karşılık gelen Lagrange katsayısıdır ve denklem (2.4)'te tanımlanan C'den küçük ve eşit pozitif bir sayıdır. Eşiz problem, asal problemin aksine bir maksimizasyon problemi olup $\mathbf{w}$,b parametreleri yerine α_i değerlerini bulmaktadır.

Çekirdek hilesini kullanmak için denklem (2.6)'daki φ dönüşümünü uygulanırsa, eşiz problemdeki $\mathbf{x}_i . \mathbf{x}_j$ çarpımı $\varphi(\mathbf{x}_i) . \varphi(\mathbf{x}_j)$ ifadesine dönüşür. Eşiz problemin çözümü için öncelikle her bir örneğin dönüştürülmesi ve ardından iç çarpımının alınması gerekmektedir. Denklem (2.7)'den görüleceği üzere eşiz problemin çözümü için her bir örneğin dönüştürülmesinden ziyade, dönüştürülmüş örneklerin iç çarpımlarına ihtiyaç duyulmaktadır. Bu kapsamda $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$ için kullanacak çekirdek fonksiyonları sayesinde doğrudan bir dönüşüm yapmak yerine, dönüşüm ve iç çarpımı içeren tek bir fonksiyon kullanılmaktadır.

$$\mathbf{x}^{\boldsymbol{\varphi}} = \varphi(\mathbf{x}) \tag{2.6}$$

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) . \varphi(\mathbf{x}_j) \tag{2.7}$$

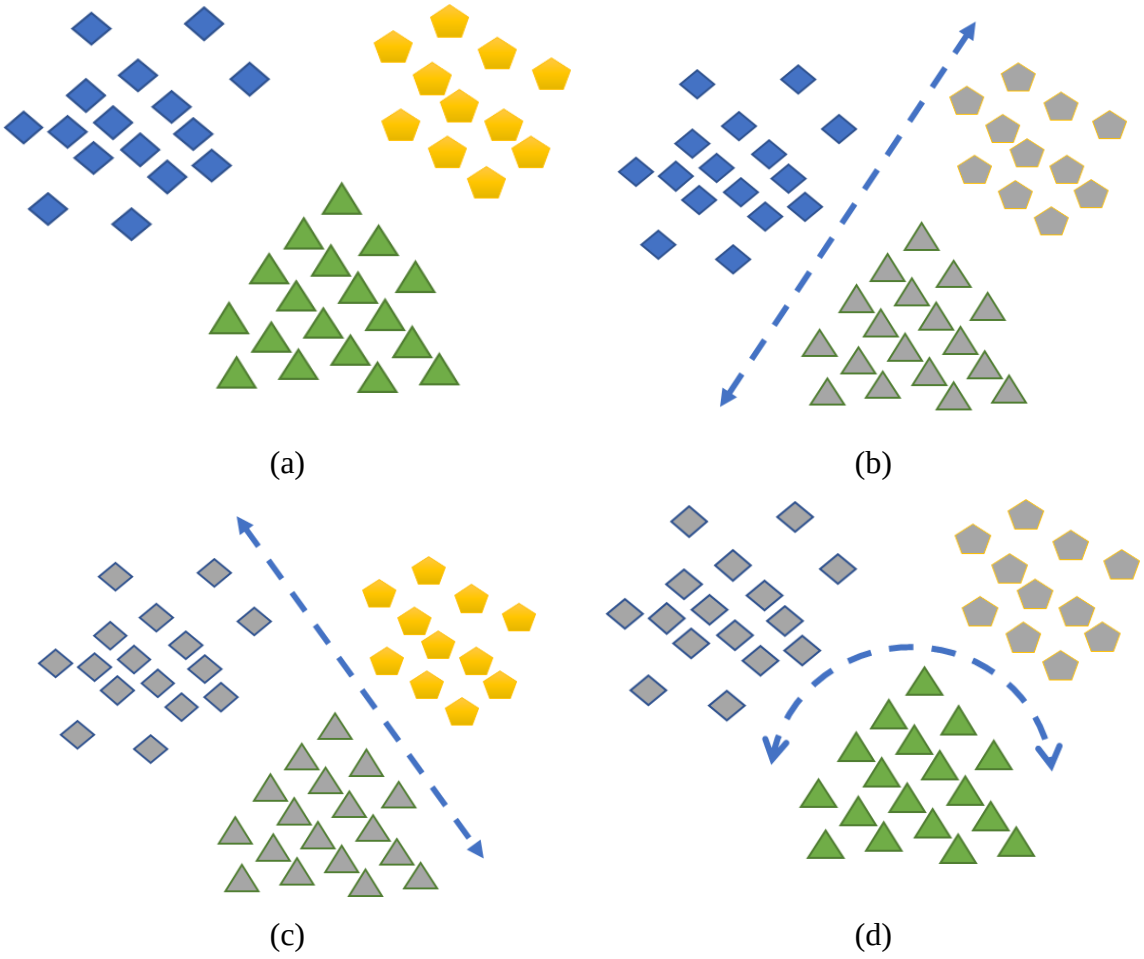
$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$ ifadesinde iç çarpım yerine farklı çekirdekler kullanılarak çekirdek hilesi elde edilir. Bazı çekirdek fonksiyonları aşağıdaki gibi tanımlanmıştır:

- $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i . \mathbf{x}_j + 1)^d$, polinom çekirdek
- $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{(-\gamma(\mathbf{x}_i - \mathbf{x}_j)^2)}$, RBF (Radial Basis Function) çekirdek
- $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \mathbf{x}_i . \mathbf{x}_j + c)$, hiperbolik tanjant çekirdek

2.2.1 Çok sınıflı yaklaşım, bire-tümü

SVM yöntemi iki sınıflı bir ayrım yöntemidir. Çok sınıflı bir sınıflandırma probleminde doğrudan kullanılamamaktadır. Bu problemlerde, bazı yaklaşımlarla SVM yöntemi kullanılarak eğitim yapılmakta ve istenilen çok sınıflı sınıflandırma modelleri oluşturulabilmektedir.

Bu yöntemlerden ilki bire-tümü (one-against-rest, OAR) yaklaşımıdır. Bu yöntemde bir sınıf pozitif olarak belirlenir ve diğer tüm sınıflar negatif sınıf olarak atanır. Elde edilen bu yeni veri yapısıyla bilgisayar eğitilir ve ilk sınıfımız için sınıflandırma modeli oluşturulur. Bu prosedür tüm sınıflara uygulanarak sınıf sayısı adedince SVM sınıflandırıcısı elde edilir (Şekil 2.8).



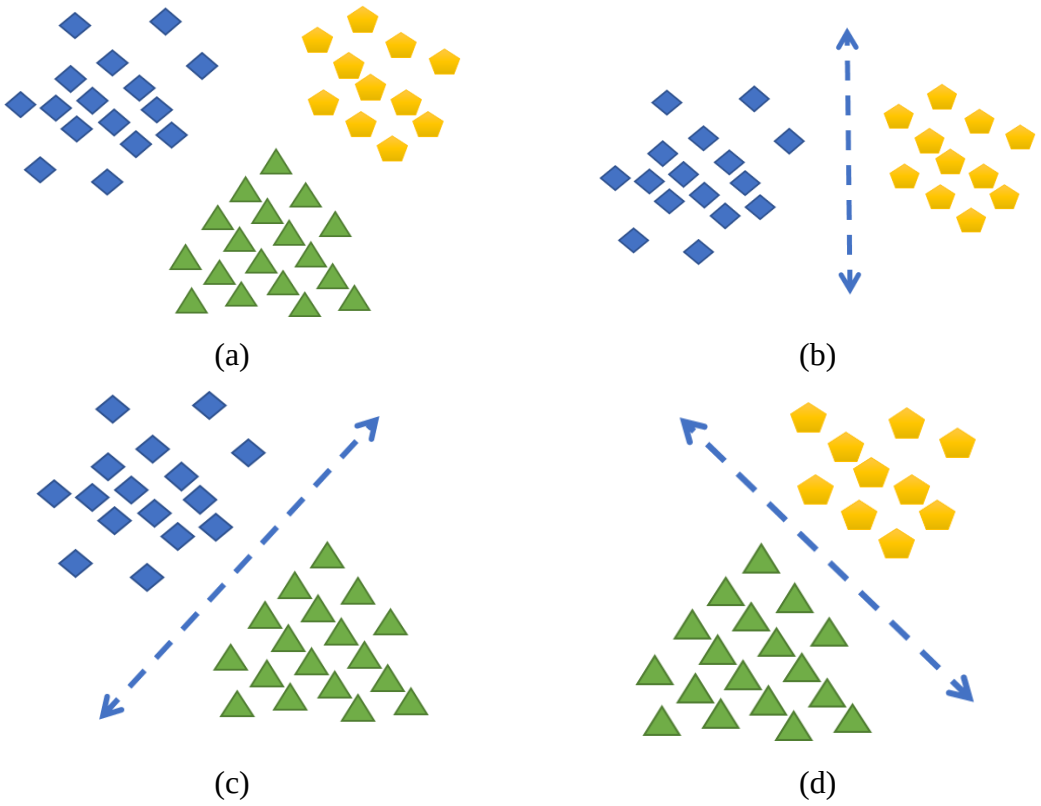
Şekil 2.8 Bire-tümü (one-against-rest, OAR) yaklaşımında bir sınıf pozitif olarak belirlenir ve diğer tüm sınıflar negatif sınıf olarak atanır. (a) ile gösterilen veriler için bire-tümü yaklaşımıyla sınıflandırıcılar (b,c,d) bulunmaktadır.

$$\hat{y} = \arg\max_i f_i(\mathbf{x}_{\text{test}}) \tag{2.8}$$

Bire tümü yöntemiyle n adet sınıflı bir sınıflandırma probleminde herhangi bir test örneğinin, $\mathbf{x}_{test}$, sınıfını belirlemek için, $\mathbf{x}_{test}$ denklem (2.8)'deki gibi tüm SVM modellerinde $(f_i, i = 1,..n)$ sınanır ve en yüksek değeri elde ettiği modelin sınıfına $(\hat{y})$ atanır.

2.2.2 Çok sınıflı yaklaşım, bire-bir

Diğer bir yöntemde bire-bir (one-against-one) yaklaşımıdır. Bu yöntemde ise aynı şekilde bir sınıf pozitif olarak seçilir. Negatif olarak kalan sınıflardan sadece biri seçilir ve SVM modeli oluşturulur. Pozitif sınıf sabit kalmak üzere diğer bir sınıf bu sefer negatif seçilir ve ayrı bir SVM modeli daha oluşturulur. Bu şekilde sınıfların tüm ikili kombinasyonları ile sınıflandırıcılar eğitilir (Şekil 2.9).



Şekil 2.9 Bire-bir (one-against-one, OAO) yaklaşımında (a)'daki veri kümesinde bir sınıf pozitif olarak belirlenir ve başka bir sınıf da negatif sınıf olarak atanır. Bu şekilde tüm sınıfların ikili kombinasyonları şeklinde sınıflandırıcılar (b,c,d) eğitilir. Test aşamasında test örneği en çok hangi sınıfa atanmışsa o sınıfa dahil edilir.

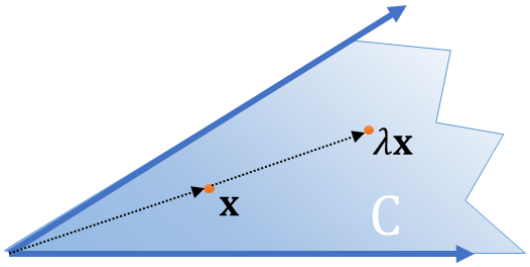
Bu yöntemde hesaplama yükü bire-tümünden daha fazla olabilmektedir. Bunun sebebi n adet sınıf için $\binom{n}{2} = n(n-1)/2$ adet SVM modelinin eğitilmesinin gerekmesidir. Fakat bire-tümü yöntemine göre eğitilmesi gereken sınıflar daha küçük olacağından, eğitilen modeller daha kolay hesaplanabilmektedir. Sınıflandırma probleminin özelliğine göre, bire-tümü veya bire-bir yöntemi avantajlı olabilmektedir. Bire-bir yönteminde $\mathbf{x}_{test}$ örneğinin hangi sınıfa ait olduğuna karar vermek için $\mathbf{x}_{test}$ tüm modellerde sınanır. Sonuçta en çok oy alan sınıf, $\mathbf{x}_{test}$ örneğinin sınıfı olarak belirlenir.

2.3 Çokyüzlü Konik Fonksiyon

2.3.1 Çokyüzlü koni

Matematikte denklem (2.9)'daki şartları sağlayan C kümesi, koni olarak tanımlanmaktadır. Şekil 2.10'da görüldüğü gibi C kümesindeki her **x** örneğinin pozitif skaler çarpımları yine C kümesi elemanı oluyorsa C kümesi bir koniyi oluşturmaktadır.

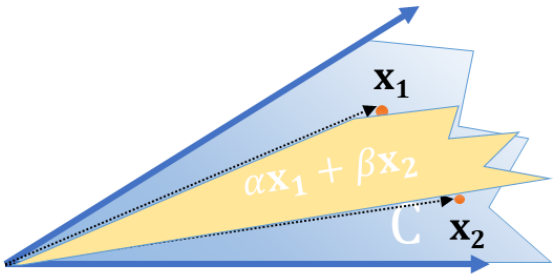
$$\mathbf{x} \in \mathrm{C} \Longrightarrow \lambda\mathbf{x} \in \mathrm{C}, \ \forall\lambda \geq 0 \tag{2.9}$$



Şekil 2.10 C kümesindeki x örneğinin pozitif skaler çarpımları yine C kümesi elemanı oluyorsa C kümesi bir konidir.

Bir koninin dışbükey olabilmesi için $\mathbf{x_1}, \mathbf{x_2} \in \mathrm{C} \Longrightarrow \alpha\mathbf{x_1} + \beta\mathbf{x_2} \in \mathrm{C}, \forall\alpha, \beta \geq 0$ şartının sağlanması gerekmektedir (Şekil 2.11). C kümesindeki $\mathbf{x_1}, \mathbf{x_2}$ elemanlarının doğrusal kombinasyonları yine C kümesinin içinde kalıyorsa bu koni bir dışbükey konidir. Norm konisi, denklem (2.10)'daki gibi tanımlanmaktadır ve dışbükey bir konidir.

$$\mathrm{C} = \{(\mathbf{x}, b) \in \mathbb{R}^{d+1} | \ \|\mathbf{x}\| \leq b\} \tag{2.10}$$



Şekil 2.11 C kümesindeki $\mathbf{x_1}$, $\mathbf{x_2}$ elemanlarının doğrusal kombinasyonları yine C kümesinin içinde kalıyorsa bu koni bir dışbükey konidir.

Çok boyutlu geometride hiperdüzlem $\{\mathbf{x} \in \mathbb{R}^d | \, \mathbf{w}^T\mathbf{x} + b = 0\}$, yarı uzay $\{\mathbf{x} \in \mathbb{R}^d | \, \mathbf{w}^T\mathbf{x} + b \leq 0\}$ veya $\{\mathbf{x} \in \mathbb{R}^d | \, \mathbf{w}^T\mathbf{x} + b \geq 0\}$, $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$ olarak tanımlanmaktadır. Hiperdüzlemler ve yarı uzaylar dışbükey kümelerdir.

$\mathbb{R}^d$ uzayındaki çokyüzlü (polyhedron) ise sınırlı sayıdaki yarı uzayın kesişimi ile oluşan kümedir. Dışbükey uzayların kesişimi dışbükey olduğundan, çokyüzlü de dışbükey bir kümedir. Çokyüzlü küme, denklem (2.11)'deki gibi tanımlanmaktadır.

$$\{\mathbf{x} \in \mathbb{R}^d | \, \mathbf{Ax} + \mathbf{b} \preccurlyeq \mathbf{0}\}, \; \mathbf{A} \in \mathbb{R}^{m \times d}, \mathbf{b} \in \mathbb{R}^m \qquad (2.11)$$

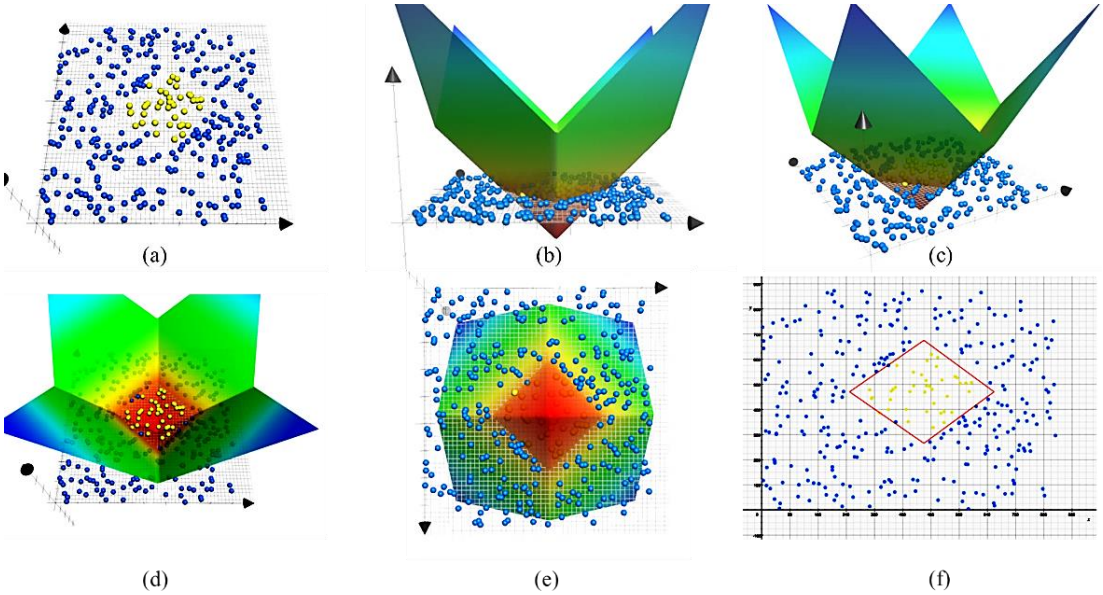
Denklem (2.11)'de **A** matrisinin her bir satırı bir eşitsizliği temsil eder. " $\preccurlyeq$ " sembolü matrisin her bir satırındaki eşitsizliği gösterir. Çokyüzlü koni ise $C = \{\mathbf{x} \in \mathbb{R}^d | \, \mathbf{Ax} \preccurlyeq 0\}$ şeklinde tanımlanır. Geometrik olarak her bir satır orijinden geçen yarı uzaydır. Denklem (2.10) ile verilen norm konisinde, eğer L_1 normu kullanıldıysa oluşan yapı çokyüzlü bir konidir. L_2 normunda ise oluşan yapı koni olsa da çokyüzlü değildir.

2.3.2 Çokyüzlü konik fonksiyon

Denklem (2.12)'te verilen PCF fonksiyonu iki ayrık kümeyi birbirinden ayıran karar sınırlarını oluşturmak için kullanılmaktadır ve $\mathbb{R}^d \rightarrow \mathbb{R}$'ye tanımlıdır. Denklem (2.12) ile tanımlanan PCF fonksiyonunun grafiği tepe noktası $(\mathbf{c}, b) \in \mathbb{R}^d \times \mathbb{R}$ olan çokyüzlü bir konidir (Gasimov ve Ozturk, 2006).

$$f_{\mathbf{w},\gamma,\mathbf{c},b}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) + \gamma\|\mathbf{x} - \mathbf{c}\|_1 - b \qquad \text{(PCF) (2.12)}$$

Denklem (2.12)'de $\mathbf{x} \in \mathbb{R}^d$ test noktası, $\mathbf{c} \in \mathbb{R}^d$ koninin tepe noktası bileşenini, $\mathbf{w} \in \mathbb{R}^d$ ağırlık vektörü, $\gamma \in \mathbb{R}^+$ ağırlık katsayısı, $b \in \mathbb{R}^+$ ofset değer ve koninin tepe noktası bileşeni, $\|\mathbf{u}\|_1 = |u_1| + \ldots + |u_d|$ L_1 normu ifade etmektedir.



Şekil 2.12 İki sınıfın ayrılması amacıyla PCF fonksiyonu ile koni oluşturulmaktadır. Bu koninin farklı açılardan görünümü (b,c,d,e) ile gösterilmiştir. (f)'te, pozitif (sarı) örnekler, negatif (mavi) örneklerden PCF seviye kümesinden elde edilmiş karar sınırı ile ayrılmıştır (Cevikalp ve Saglamlar'dan, 2019).

PCF fonksiyonunun alt seviye kümeleri, S, dışbükey bir çokyüzlü oluşturur (Gasimov ve Ozturk, 2006). Alt seviye kümeleri denklem (2.13) ve (2.14) ile ifade edilir. Şekil 2.12 (f)'teki iki boyutlu düzlemde PCF'in alt seviye kümesi bir dörtgendir.

$$S(\alpha) = \left\{\mathbf{x} \in \mathbb{R}^d \middle| \, f(\mathbf{x}) \leq \alpha\right\}, \alpha \in \mathbb{R} \tag{2.13}$$

$$S(\alpha) = \left\{\mathbf{x} \in \mathbb{R}^d \middle| \, \mathbf{w}^T(\mathbf{x} - \mathbf{c}) + \gamma\|\mathbf{x} - \mathbf{c}\|_1 - b \leq \alpha\right\}, \alpha \in \mathbb{R} \tag{2.14}$$

$$S(\alpha) = \{\mathbf{x} \in \mathbb{R}^d | \, \tilde{\mathbf{w}}^T(\mathbf{x} - \mathbf{c}) - b \leq \alpha\}, \alpha \in \mathbb{R}, \quad \tilde{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ \gamma \, sgn(\mathbf{x} - \mathbf{c}) \end{bmatrix} \tag{2.15}$$

Denklem (2.15)'te görüleceği üzere PCF fonksiyonunun seviye kümeleri denklem (2.11)'deki çokyüzlü şartını sağlamaktadır. Buradan $S(\alpha)$ alt seviye kümelerinin çokyüzlü olduğu görülür. Seviye kümeleri çokyüzlü olan ve fonksiyon grafiği koni olan PCF, çokyüzlü koni olarak tanımlanır (Gasimov ve Ozturk, 2006).

Şekil 2.12'de PCF fonksiyonu ile nasıl sınıflandırma yapıldığı gösterilmektedir. Pozitif örnekler yoğun bir şekilde negatif örneklerle sarılmıştır. Bu sınıfları doğrusal olarak ayırmak mümkün değildir. PCF konisi ile pozitif örnekler koni içine alınmakta, negatif örnekler de koni dışında tutularak ayırım gerçekleştirilmektedir.

İki ayrı sınıfa ait örnekler $(\mathbf{x}_i, y_i)$ için pozitif örnekler $y_i = +1$, negatif örnekler $y_i = -1$ olarak etiketlenirse, PCF kullanıldığında pozitif ve negatif örnekler denklem (2.16) ve (2.17)'deki şartları sağlamaktadır.

$$f(\mathbf{x}_i) < 0, y_i = +1 \tag{2.16}$$

$$f(\mathbf{x}_i) \geq 0, y_i = -1 \tag{2.17}$$

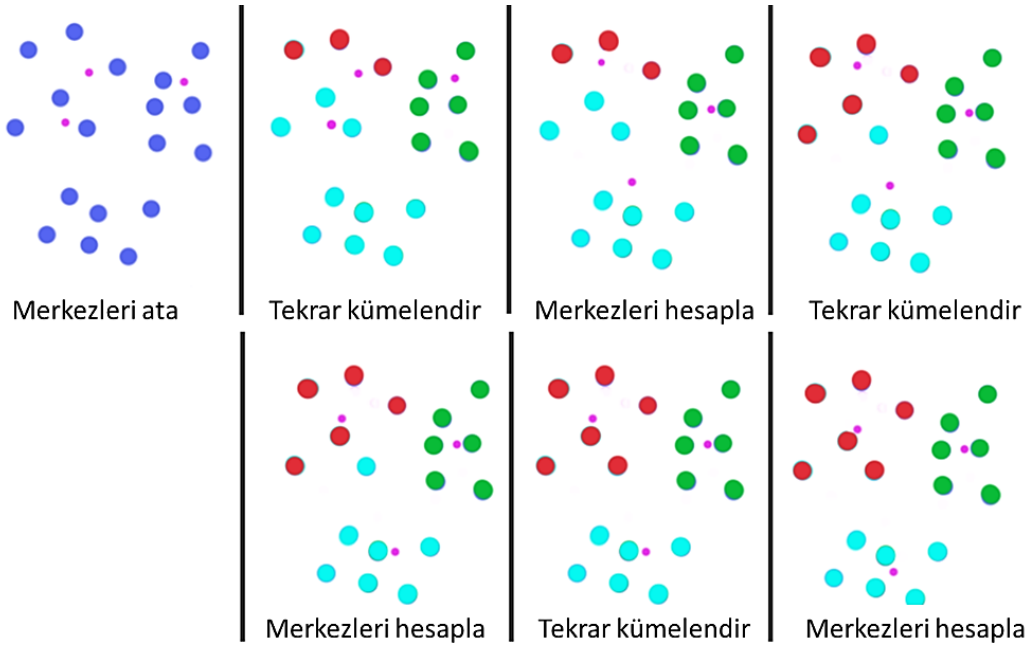
Örneklerin denklem (2.16) ve (2.17)'deki gibi ayrımı, SVM'den farklılık göstermektedir. SVM'de pozitifler için $\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \geq 0$, negatifler için $\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \leq 0$ şartları sağlanmaktadır. Halbuki PCF'te pozitif örnekler için negatif, negatifler için pozitif PCF değeri elde edilmektedir. PCF'in sınıflandırması SVM'den bu yönüyle farklılık göstermektedir.

Denklem (2.12) ile belirtilen en uygun koninin bulunması amacıyla $\mathbf{w}, \gamma, b$ parametrelerinin eğitim sürecinde bulunması gerekmektedir. Gasimov ve Ozturk (2006), bu parametrelerin bulunması amacıyla doğrusal programlama yöntemini kullanılmıştır. Ayrıca PCF'te bulunan koninin tepe noktası bileşeni, $\mathbf{c}$, eğitim sürecinde hesaplanmamakta ve eğitim sürecinde kullanılan verilenden rastgele olarak seçilmektedir.

2.4 *k*-Merkezli Kümele Yöntemi

İlk olarak MacQueen (1967) tarafından kullanılan *k*-merkezli kümeleme yöntemi, ağırlık merkezi kullanan algoritmalardan en sık kullanılan ve uygulaması kolay olan bir kümeleme algoritmasıdır. Küme sayısı k olmak üzere, kümeler $\mathrm{S} = \{\mathrm{S}_1, \ldots, \mathrm{S}_k\}$ şeklinde ifade edilebilir. Yöntemin başlangıcında her kümenin ağırlık merkezi, $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)$, rastgele bir şekilde seçilir. Daha sonra her bir örnek, $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, Öklid uzaklığına göre kendine en yakın ağırlık merkezinin kümesine atanır. Bütün örneklerin ataması bittikten sonra, k adet kümenin yeni üyeleriyle ağırlık merkezi tekrar hesaplanır ve bir önceki adım tekrarlanır

(Şekil 2.13). Yöntem, denklem (2.18)'de verilen küme içi kareler toplamının azaltılmasını ve bu şekilde kümelemeyi amaçlar. Denklem (2.18)'de verilen küme içi kareler toplamındaki değişim yeterince azaldığında veya artık hiçbir örneğin kümesinin değişmediği durumda yöntem sonlandırılır. Yöntem en uygun küme sayısını belirlememektedir, bunun için k algoritmaya başlamadan önce belirlenmelidir. Yönteme ilk başlangıç rastgele ağırlık merkezleriyle başladığı için k-merkezli kümeleme yöntemi, her başlatıldığında farklı yerel optimumlara ulaşabilmektedir. Özellikle büyük veri kümelerinde mutlak optimumlara yaklaşmak oldukça güçtür. Buna rağmen k-merkezli kümeleme yöntemi tercih edilen oldukça etkili bir yöntemdir.



Şekil 2.13 k-merkezli kümeleme yöntemi ile verilerin üç kümeye ayrılmasının aşamalı gösterimi (mavi, kırmızı, yeşil noktalar kümelere ayrılan örnekleri; mor küçük noktalar küme merkezlerini temsil eder). Hui (2020)'den uyarlanmıştır.

$$\operatorname{argmin}_S \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \tag{2.18}$$

Üç adımlı k-merkezli kümeleme algoritması aşağıdaki gibidir:

Adım-1: Rastgele *k* adet örnek seç ve bunları kümelerin ağırlık merkezi, ($\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$), olarak ata.

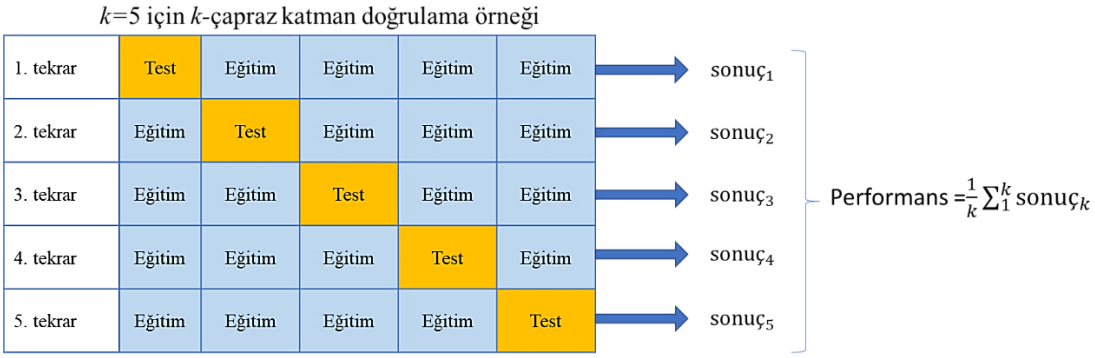
Adım-2: Her bir örneği kendine en yakın ağırlık merkezinin kümesine ata ve *k* adet kümeyi, $S = \{S_1, \ldots, S_k\}$, yeniden oluştur.

Adım-3: Eğer denklem (2.18)'deki değerin değişimi yeterince küçükse veya kümesi değişen örnek yoksa işlemi sonlandır, yoksa *k* kümenin merkezlerini tekrar hesapla ve Adım 2'yi tekrar uygula.

2.5 Performans Değerlendirme Yöntemleri

2.5.1 *k*-çapraz katman doğrulama

Sınıflandırıcılar bir veriyi ayırırken ne kadar iyi genelleme yapabilirse veya yeni örnekler karşısında iyi sonuçlar elde edebilirse o kadar başarılıdır. Bunun için yeni örnekler ile performansın denenmesi gerekir. Bu yeni verileri elde etmek her zaman pratik ve etkili olmayabilir. Bunun yerine daha basit ve uygulanabilir k-çapraz katman doğrulama (k-fold cross validation) yöntemi kullanılabilir. Bu yöntemde mevcut veri kümesi k adet eşit parçaya ayrılır. Bu parçaların homojen bir dağılıma sahip olması, her sınıftan aynı miktarda örnek içermesi beklenmektedir. Sonra bir parça test için seçilir. Kalan $(k-1)$ parça ise eğitim amacıyla kullanılır. Bu işlem her bir parça test için kullanılana kadar tekrarlanır ve süreç sonunda k adet sonuç elde edilir (Şekil 2.14). Elde edilen sonuçların ortalaması ve standart sapması sınıflandırma performansı açısından yöntemin gücünü gösterir.



Şekil 2.14 k-çapraz katman doğrulama (k-fold cross validation) yöntemi ile veri kümelerinin bölümlere ayrılması ve sonuçların elde edilmesi

2.5.2 Performans değerlendirme ölçütleri

Performans değerlendirme ölçütü sınıflandırıcının başarımını gösteren bir orandır. Verilen sınıflandırma probleminin özelliğine göre farklı performans kriterleri tercih edilebilir. Örneğin pozitif örneklerin az ve negatif örneklerin çok fazla olduğu bir sınıflandırma probleminde, pozitiflerin doğru bulunması daha önemli olabilir. Fakat seçtiğimiz performans ölçeğinde negatiflerin doğru bulunması pozitiflerin yanlış bulunmasının önüne geçerek sınıflandırıcının başarımı hakkında yanlış bilgi verebilir. Bu açıdan performans ölçeğinin seçimi önemlidir.

Performans değerlendirmesi Çizelge 2.1'de verilen hata matrisi (confusion matrix) ile yapılmaktadır. Pozitif ve negatif sınıfa ait örneklerin doğru veya yanlış tahmin edilmesine göre bu tablo oluşturulmaktadır. Bu matriste veri kümesinde tahmin edilen sınıfla eşleşen veya eşlemeyen gerçek sınıfların sayısı gösterilmektedir.

Çizelge 2.1 Hata matrisi (confusion matrix)

		Gerçek Sınıf	
		Pozitif	**Negatif**
Tahmin	**Pozitif**	*DP*	*YP*
	Negatif	*YN*	*DN*

Sınıflandırma işleminde doğruluk değeri (accuracy) denklem (2.18) ile ifade edilmekte olup doğru tahmin edilen örneklerin tüm örneklere oranı olarak tanımlanır.

$$\text{Doğruluk değeri } = \frac{DP + DN}{DP + DN + YP + YN} = \frac{DP + DN}{\text{Toplam örnek sayısı}} \quad (2.18)$$

Denklem (2.19)'da tanımlanan yanlış pozitif oranı (false positive rate), tüm negatif örneklerdeki yanlış pozitif oranıdır. Doğru pozitif oranı (true positive rate) ise doğru pozitiflerin tüm pozitif örneklere göre oranı olup denklem (2.20)'de verilmiştir. Doğru pozitif oranı aynı zamanda Duyarlılık (recall) ölçeği olarak da adlandırılmaktadır.

$$\text{Yanlış pozitif oranı} = \frac{YP}{YP + DN} = \frac{YP}{\text{Toplam negatif sayısı}} \quad (2.19)$$

$$\text{Doğru pozitif oranı} = \text{Duyarlılık} = \frac{DP}{DP + YN} = \frac{DP}{\text{Toplam pozitif sayısı}} \quad (2.20)$$

Kesinlik (precision) ölçeği pozitif tahmin edilen örneklerden kaçının doğru olarak belirlendiğini göstermekte olup denklem (2.21)'da belirtilmiştir.

$$\text{Kesinlik } = \frac{DP}{DP + YP} \quad (2.21)$$

F-ölçüsü (F-score) Kesinlik ve Duyarlılık ölçeklerinin harmonik ortalaması alınarak denklem (2.22)'deki gibi bulunmaktadır.

$$F - \text{ölçüsü } = \frac{2 \times Kesinlik \times Duyarlılık}{Kesinlik + Duyarlılık} \quad (2.22)$$

Everingham vd.nin (2010) PASCAL VOC'ta belirttiği Ortalama Hassasiyet (Average Precision, AP) ölçeği Kesinlik ve Duyarlılık ölçekleri kullanılarak elde edilmektedir. Bu ölçeğin hesaplanabilmesi için öncelikle Kesinlik ve Duyarlılık grafiğinin çizilmesi gerekir. PASCAL VOC'ta bir resimde tahmin edilen nesnenin *DP* olabilmesi için nesnenin gerçek kuşatan kutusu (bounding box) ile tahmin edilen kuşatan kutu 0,5'ten daha fazla oranda örtüşmelidir. Tahmin edilen nesneler, güven puanına (confidence score) göre büyükten küçüğe doğru sıralanır ve her nesne tahmini sonrası Kesinlik ve Duyarlılık değerleri hesaplanır. Tüm nesne tahminleri bittikten sonra Kesinlik ve Duyarlılık değerleri ile duyarlılık doğruluk grafiği (precision-recall graph) çizilir. Duyarlılık doğruluk eğrisi

altında kalan alanın değeri Ortalama Hassasiyet (Average Precision, AP) olarak tanımlanan ölçeği vermektedir.

Yukarıda verilen ölçekler iki sınıflı sınıflandırmada performans değerlendirilmesi için kullanılmaktadır. Çok sınıflı bir veri kümesinde performans değerlendirmesi için Bölüm 2.2.1'de belirtilen çok sınıflı yaklaşımlar kullanılarak her bir örneğin sınıfı belirlenir ve denklem (2.23)'teki sınıflandırma oranı (classification rate) hesaplanır.

$$Sınıflandırma\ oranı\ = \frac{Doğru\ sınıflanan\ örnek\ sayısı}{Toplam\ örnek\ sayısı} \tag{2.23}$$

Çok sınıflı veri kümelerinde kullanılan diğer bir performans ölçüsü ise yukarıda bahsedilen AP kullanılarak yapılmaktadır. Bölüm 2.2.1'de belirtilen çok sınıflı yaklaşımlar ile her sınıf için ayrı AP değeri elde edilir. Tüm sınıfların AP değerlerinin ortalaması alınarak ortalama AP (mAP) denklem (2.24)'teki gibi hesaplanır. Denklem (2.24)'teki n sınıf sayısını ifade etmektedir. Hesaplanan mAP çok sınıflı veri tabanlarının karşılaştırılmasında kullanılır.

$$Ortalama\ \text{AP (mAP)} = \frac{\sum_{i=1}^{n} AP_i}{n} \tag{2.24}$$

3. LİTERATÜR ARAŞTIRMASI

Gasimov ve Öztürk'ün (2006) önerdiği sınıflandırma yöntemi çokyüzlü konik sınıflandırıcıların ilki olup bu yöntemde rastgele bir koni tepe noktası, **c**, seçilmekte ve doğrusal programlama ile PCF parametreleri bulunmaktadır. Bulunan parametrelerle oluşan koni, bütün pozitif örnekleri karar sınırı içine almadıysa, dışarıda kalan pozitif örneklerle yeni bir PCF için yukarıdaki adımlar tekrarlanmaktadır. Dışarıda herhangi bir pozitif kalmadıysa yöntem durmaktadır. Diğer bir ifadeyle sınıfların tam olarak ayrıştırılamadığı durumlarda bu örnekler için hata-ceza yaklaşımı kullanmak yerine, ayrı PCF sınıflandırıcısı eğitilmektedir. Ayrıca bir sınıflandırıcı yerine aynı amaçla çok sayıda PCF sınıflandırıcı kullanılması test hızını etkilemektedir. Diğer karşılaşılan bir problemse doğrusal programlama kullanan bu sınıflandırıcıyla veri sayısının 20 000 üzerinde olduğu ve yüksek öznitelik boyutuna sahip (620 boyutlu LBP+HOG öznitelikleri vb.) bazı veri tabanlarında hafıza problemiyle karşılaşılabilmektedir.

Bagirov vd. (2013) ise PCF'in büyük veri tabanlarında daha iyi çalışması için farklı bir yaklaşım (Hybrid Polyhedral Conic and Max–min Separability, HPCAMS) önermiştir. Yöntem iki aşamadan oluşmaktadır. İlk aşamada her bir sınıf için bir PCF seti bulunur. Bulunan PCF'leri içeren örnekler çıkarılır ve geri kalan örneklerle parçalı doğrusal sınırlar hesaplanır.

Öztürk vd. (2015) ise yine PCF kullanan aşamalı bir yöntem (Incremental Polyhedral Conic Separation, IPCS) önermiştir. Yöntemde öncelikle en uygun başlangıç noktalarının tespiti için bir adım eklenmiştir. Sonrasında bir hata fonksiyonunun azaltılmasını amaçlayan ve bu doğrultuda PCF fonksiyonlarını aşamalı olarak üreten bir algoritma uygulanmıştır.

Öztürk ve Çiftçi (2015) önerdiği yöntemde, veri tabanındaki tüm sınıflar bire-tümü yöntemiyle ayrıştırılmıştır. Her bir sınıfa ait örnekler *k*-merkezli kümeleme yöntemiyle k adet kümeye ayrılmış ve her bir küme için de bir PCF elde edilmiştir. Toplamda $k \times sınıf\ sayısı$ kadar PCF bulunmaktadır. Doğrusal programlamada her bir örnek kullanıldığından, $k \times sınıf\ sayısı$ kadar PCF eğitimi için her bir noktanın tekrar tekrar kullanılması gerekmektedir. Bu ise eğitim sürecini uzatmaktadır. Bundan dolayı büyük veri

tabanları için doğrusal programlama yöntemin uygulanması kolay olmamaktadır. Bu sorunu aşmak için Çimen vd. (2017) yeni bir yöntem (Incremental Conic Functions, ICF) önermiştir. Buradaki yöntemde Öztürk ve Çiftçi'nin (2015) yöntemindeki sıkıntılara çözüm aranmıştır. PCF oluştururken kullanılmasına gerek olmayan örnekler çıkarılmış ve doğrusal programlamanın yükü azaltılmaya çalışılmıştır. Ayrıca küme sayısı k sabit alınmamış ve algoritmanın içinde en uygun değeri verecek şekilde hesaplanmıştır. Bu yeni yöntemle Öztürk ve Çiftçi'nin (2015) yöntemindeki sonuçlara yakın sonuçlar elde edilmekle birlikte %64 oranında bir hızlanma sağlamış ayrıca k değerinin önceden belirlenmesine de gerek kalmamıştır.

Çimen ve Öztürk (2019) ise Öztürk ve Çiftçi'nin (2015) yöntemine benzeyen bir yöntem (One-Class Polyhedral Conic Functions, O-PCF) önermiştir. Tek sınıf bir sınıflandırıcıda eğitim sırasında sadece pozitif sınıfa ait örnekler kullanılmaktadır. Pozitif örnekler dışında başka bir veri bulunmamaktadır. Tek sınıflı sınıflandırıcı kullanan Çimen ve Öztürk'ün (2019) yönteminde, PCF'in karar sınırlarının büyüklüğü sınıflandırma hatasına göre belirlenerek en uygun PCF'in bulunması amaçlanmıştır.

Öztürk ve Çimen'in (2019) PCF uygulamasında ise iki aşamalı bir yaklaşım izlenmiştir. İlk aşamada veri tabanındaki örneklere öznitelik eklenmiş, ikinci aşamadaysa genişletilmiş öznitelikler ile veri tabanı SVM kullanılarak eğitilmiştir. İlk aşamada öznitelik bulunmadan önce Öztürk ve Çiftçi'nin (2015) yöntemindeki gibi her bir sınıf için *k*-merkezli kümeleme yöntemiyle k adet küme bulunmaktadır. Her bir örnek için, her sınıfın k adet PCF'inden aldığı en düşük değer, o örneğe öznitelik olarak eklenir. Bu işlem sonundan tüm örneklerin öznitelik boyutu, sınıf sayısı kadar artmış olur. Sonrasında ise ikinci aşamaya geçilir ve genişletilmiş öznitelikler ile veri tabanı SVM yöntemiyle eğitilir.

Uylaş (2016) ise yine PCF kullanan ve büyük marj yaklaşımıyla PCF karar sınırlarını oluşturan bir yöntem önermiştir. Bu yöntemde de PCF parametreleri doğrusal programlama ile hesaplanmaktadır.

Yukarıda, PCF fonksiyonu kullanılarak geliştirilen yöntemlerden bahsedilmiştir. Bu yöntemlerin çoğu doğrusal programlama yöntemiyle eğitilen modeller üzerinden hazırlanmıştır. Doğrusal programlamada tüm örnekler aynı anda eğitim için

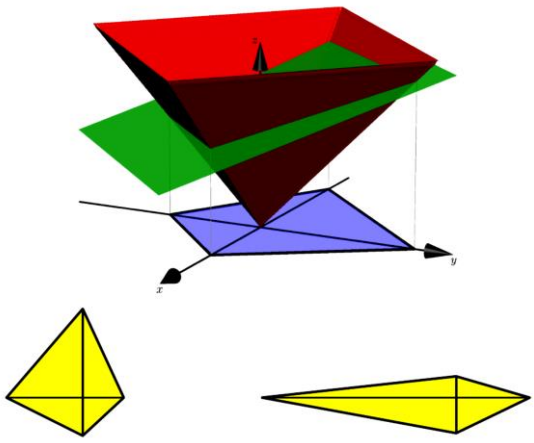
kullanılmaktadır. Bundan dolayı veri tabanının boyutlarının artmasıyla doğrusal programlamanın hesap yükü artmaktadır. Ayrıca birden fazla PCF kullanımı ile de hesaplama yükü katlanmaktadır. Birden fazla PCF kullanımının eğitim sürecine olumsuz katkısı olduğu gibi test süresini de uzatmaktadır. Ayrıca birden fazla PCF kullanımı eğitim sürecinin karmaşıklaşmasına da neden olabilmektedir. Bu çalışmada basit PCF yapısı kullanılarak tek bir PCF sınıflandırıcı ile hızlı ve başarılı PCF yöntemleri önerilmiştir.

4. YÖNTEM

4.1 Çokyüzlü Konik Sınıflandırıcı

Çokyüzlü konik sınıflandırıcı (Polyhedral Conic Classifier), denklem (2.12)'deki PCF fonksiyonu temel alarak önerilmiştir. PCF ile elde edilen karar sınırı, bir arada toplanmış pozitif örnekleri diğer tüm negatif örneklerden ayıran kompakt ve dışbükey alandır (Bkz. Şekil 2.13). Ayrıca bu alan marj tabanlı öğrenmeye de uygundur. Önerilen PCC'de makul sayıdaki serbest değişken ile de aşırı öğrenme ve hız kontrolü kolaylıkla yapılabilmektedir.

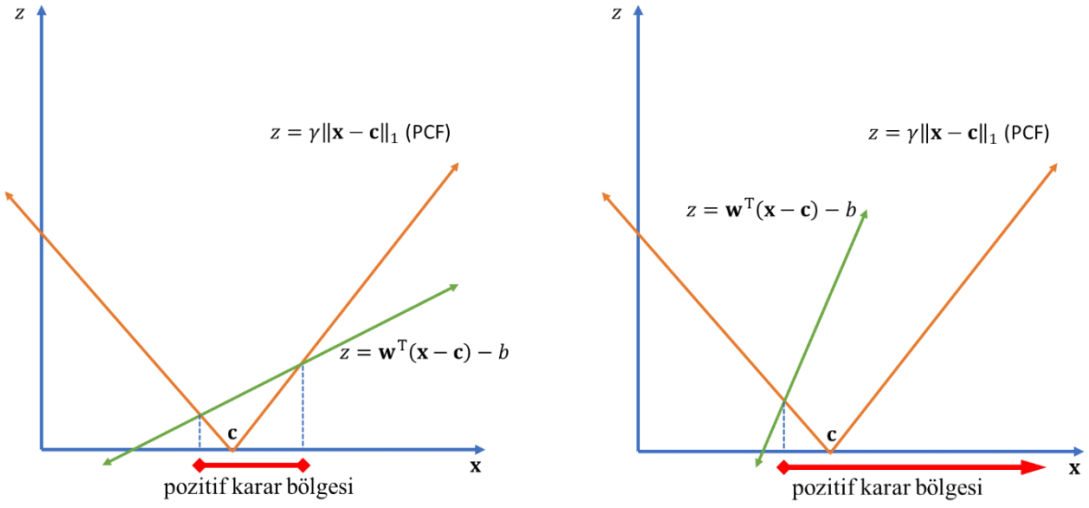
Denklem (2.12)'deki PCF fonksiyonu ile elde edilen pozitif karar bölgeleri, $\mathbf{c}$ merkezli bir koniden geçen hiperdüzlemin kesit alanını ile ifade edilmektedir (Şekil 4.1). Burada $z = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) - b$ hiperdüzleminin, $z = \gamma\|\mathbf{x} - \mathbf{c}\|_1$ konisi içinde kalan kesitinin öznitelik uzayındaki izdüşümü pozitif karar bölgesini oluşturmaktadır. PCF fonksiyonu ile pozitif örnekler için $f(\mathbf{x}) \leq 0$, negatif örnekler için $f(\mathbf{x}) > 0$ şartı sağlanmaktadır. Marj tabanlı eğitim sürecinde ise pozitifler için $f(\mathbf{x}) < -1$ ve negatifler için $f(\mathbf{x}) > 1$ şartlarının sağlaması zorlanmaktadır.



Şekil 4.1 PCF, bir koniden geçen hiperdüzlemin öznitelik uzayındaki izdüşümüdür (Cevikalp ve Triggs'den, 2017 a).

Denklem (2.12)'deki PCF fonksiyonunda $b > 0, \gamma > 0$ şartları tanımlanmıştı. Bu şartlarla beraber $\|\mathbf{w}\|_\infty < \gamma$ ($\|\mathbf{u}\|_\infty = \max_{i=1}^{d}|u_i|$) şartı da sağlanırsa, herhangi bir $\tau \in \mathbb{R}$

için $f(\mathbf{x}) < \tau$ ile belirtilen bölge, $\mathbb{R}^d$'de dışbükey ve kompakt bir alan oluşturur (Şekil 4.2). Gasimov ve Öztürk'ten (2006) farklı olarak PCC'de $b > 0, \gamma > 0$ şartları zorlanmadan ve $\|\mathbf{w}\|_\infty < \gamma$ şartı sağlanmadan eğitim süreci gerçekleştirilmiştir. Eğitim sürecinde kullandığımız veri tabanındaki pozitif örnekler kompakt ise pozitif karar bölgesinin de kompakt bir yapıya yakınsaması beklenmektedir.



(a) (b)

Şekil 4.2 a) $b > 0, \gamma > 0$, $\|\mathbf{w}\|_\infty < \gamma$ şartları ile $z = \mathbf{w}^T(\mathbf{x} - \mathbf{c}) - b$ hiperdüzlem (yeşil) eğimi koninin (turuncu) yanal yüzeyinin eğiminden daha küçük olur ve pozitif karar bölgesi kompakt sınırlı bir alan oluşturur, b) diğer durumda karar bölgesi sınırlı olmaz, sonsuza uzar.

PCC'de marj tabanlı sınıflandırıcıyı tanımlarken, basit bir dönüşüm kullanarak sınıflandırıcımızı doğrusal SVM sınıflandırıcısına benzetebiliriz. Bu dönüşüm; $(\mathbf{x} - \mathbf{c})$ vektörüne $\|\mathbf{x} - \mathbf{c}\|_1$ ekleyerek denklem (4.1)'deki gibi, $-\mathbf{w}$ vektörüne $-\gamma$ ekleyerek denklem (4.2)'deki gibi, b ise denklem (4.3)'te olduğu gibi kullanılarak yapılmaktadır. Dönüşüm sonucunda SVM yapısındaki denklem (4.4) elde edilmektedir.

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ \|\mathbf{x} - \mathbf{c}\|_1 \end{pmatrix} \in \mathbb{R}^{d+1} \tag{4.1}$$

$$\tilde{\mathbf{w}} = \begin{pmatrix} -\mathbf{w} \\ -\gamma \end{pmatrix} \in \mathbb{R}^{d+1} \tag{4.2}$$

$$\tilde{b} = b \tag{4.3}$$

$$f_{\tilde{\mathbf{w}},\tilde{b}}(\tilde{\mathbf{x}}) = \tilde{\mathbf{w}}^T\tilde{\mathbf{x}} + \tilde{b} \tag{4.4}$$

Yapılan bu dönüşümle denklem (4.4)'deki PCC karar fonksiyonu, SVM yöntemindeki gibi pozitifler için $\widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}} + \tilde{b} \geq 0$, negatifler için $\widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}} + \tilde{b} < 0$ şartlarını sağlamış olur. Ayrıca PCF'in ∓ 1 marjı da SVM'de kullanılan ± 1 marjına dönüşür. Bu dönüşüm sayesinde denklem (2.12)'deki PCF fonksiyonu, SVM'deki marj tabanlı karar fonksiyonuna benzetilmiş olur. Bu da bize SVM için kullanılan matematiksel çözümlerin ve yaygın yazılımların PCC sınıflandırıcısı için de kullanılabilmesine olanak sağlar. Çok bilinen denklem (2.4)'teki SVM kuvadratik programlama PCC için uygulanırsa:

$$\arg\min_{\widetilde{\mathbf{w}},\tilde{b}} \frac{1}{2}\widetilde{\mathbf{w}}^{\mathrm{T}}\widetilde{\mathbf{w}} + C_{+}\sum_{i}\xi_i + C_{-}\sum_{j}\xi_j \tag{4.5}$$

$$s.t.\ \widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_i + \tilde{b} + \xi_i \geq 1, i \in \mathrm{I}_{+}, \tag{4.6}$$

$$\widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_j + \tilde{b} - \xi_j \leq -1, j \in \mathrm{I}_{-}, \tag{4.7}$$

$$\xi_i, \xi_j \geq 0, \tag{4.8}$$

Denklem (2.4)'deki gibi denklem (4.5)'te de $\mathrm{I}_{+(-)}$ pozitif (negatif) sınıfa ait örneklerin indislerini, $C_{+(-)}$ ceza katsayısını, $\xi_{i(j)}$ ise kısıtları sağlamayan örneklere karşılık gelen artıran değişkeni (slack variable) ifade etmektedir. Bu problemi çözmek için SVM eğitiminde kullanılan büyük veri ile uyumlu herhangi bir algoritma (Shalev-Shwartz vd.nin (2011) rassal gradyanı veya Franc ve Sonnenburg'un (2008) kesen hiperdüzlemler -cutting planes- gibi) kullanılabilir. Eğer pozitif sınıfa ait veriler öznitelik uzayında kompakt bir bölgede toplanmışsa (Bkz. Şekil 2.13), sınıflandırıcı pozitif örnekleri etrafındaki negatif örneklerden başarılı şekilde ayırabilmektedir. Oluşan karar bölgesi de kompakt ve dışbükeydir. Yukarıdaki eğitim sonucunda karar bölgelerini oluşturan $\widetilde{\mathbf{w}}, \tilde{b}$ parametreleri öğrenilmektedir. $\widetilde{\mathbf{w}}, \tilde{b}$ parametreleri ve $\mathbf{c}$ koni merkezi ile model oluşturulmaktadır.

Eğitim sürecinden sonra elde edilen model ile herhangi bir $\mathbf{x}_{test}$ örneğinin sınıflandırılması sağlanabilir. Öncelikle $\mathbf{x}_{test}$ örneğine eğitimde uygulanan dönüşüm uygulanmalıdır. Bunun için modeldeki $\mathbf{c}$ merkezi kullanılarak denklem (4.1)'deki $\tilde{\mathbf{x}}_{\boldsymbol{test}} = \left(\dfrac{\mathbf{x}_{\boldsymbol{test}} - \mathbf{c}}{\|\mathbf{x}_{\boldsymbol{test}} - \mathbf{c}\|_1}\right)$ dönüşüm uygulanır. Sonrasında $\tilde{\mathbf{x}}_{\boldsymbol{test}}$'in denklem (4.4)'ten aldığı değer ile sınıflandırma yapılmaktadır. Burada $\widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} \geq 0$ ise $\mathbf{x}_{test}$ pozitif sınıfa, $\widetilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} < 0$ ise $\mathbf{x}_{test}$ negatif sınıfa atanmaktadır.

PCC'de kullanılan L_1 normu yerine L_2 norm da kullanılabilir. Bu sayede bazı veri tabanlarında daha iyi sonuçlar elde edilebilmektedir. Eklenmiş öznitelik vektörü denklem (4.1) yerine denklem (4.9)'daki gibi ifade edilir. $\|.\|$ işlemi denklem (4.10)'de tanımlanan L_2 (Öklid) normun karesidir. Bu eşitlik kullanılarak elde edilen PCF ile elipsoit karar bölgeleri elde edilmektedir.

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ \|\mathbf{x} - \mathbf{c}\| \end{pmatrix} \in \mathbb{R}^{\mathrm{d}+1} \tag{4.9}$$

$$\|\mathbf{u}\| = \mathrm{u}_1{}^2 + \cdots + \mathrm{u}_\mathrm{d}{}^2 \tag{4.10}$$

Denklem (4.2)'deki kuvadratik programlama $\tilde{\mathbf{w}}$ ve $\tilde{b}$ parametrelerinin bulunmasını amaçlar. Koninin tepe noktası olan $\mathbf{c}$ parametresi ise bu işlemde hesaplanmamaktadır. $\mathbf{c}$ parametresi yerel olarak optimizasyonla hesaplanabilir. Bunu gerçekleştiren Dordinejad ve Çevikalp'in (2017) çalışması mevcuttur. Bölüm 4.5'te bu çalışmadan bahsedilmiş ve bazı sonuçlar Bölüm 5.8'de sunulmuştur. Dordinejad ve Çevikalp'in (2017) önerdiği yöntem, sonuca katkı sağlasa da işlem yükünü oldukça artırmaktadır. Bu yöntem dışında, $\mathbf{c}$ koni merkezi pozitiflerin ortalaması veya orta değeri olarak seçebilir. Bu çalışmadaki deneylerde $\mathbf{c}$ parametresi için eğitim kümesindeki pozitif örneklerinin ortalaması tercih edilmiştir.

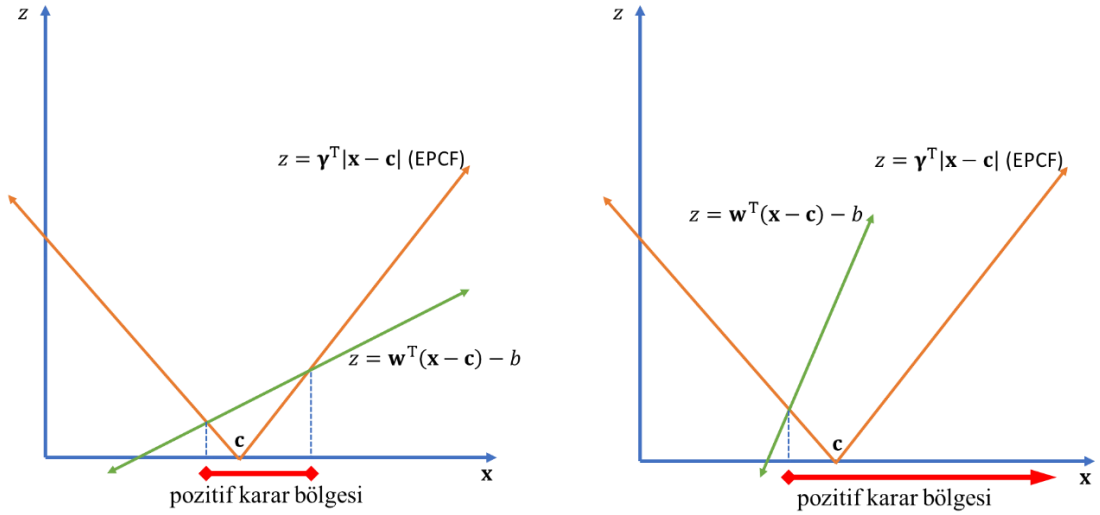
4.2 Genişletilmiş Çokyüzlü Konik Sınıflandırıcı

Genişletilmiş çokyüzlü konik sınıflandırıcı (Extended Polyhedral Conic Classifier) PCF fonksiyonu temel alınarak önerilmiş bir sınıflandırıcıdır. EPCC'de PCC'den farklı olarak denklem 2.12)'deki PCF fonksiyonunun $\gamma\|\mathbf{x} - \mathbf{c}\|_1$ teriminde, L_1 norm yerine tüm bileşenlerin mutlak değeri, $\boldsymbol{\gamma}^{\mathrm{T}}|\mathbf{x} - \mathbf{c}|$, kullanılmaktadır. $|\mathbf{u}|$ bileşen bazında mutlak değer, $|\mathbf{u}| = (|u_1|, \ldots, |u_d|)^{\mathrm{T}}$, ve $\boldsymbol{\gamma} \in \mathbb{R}^d$ bu terime ait ağırlık vektörü olarak tanımlanmaktadır. Bu farklılıklar dışındaki kalan şartlar PCC ile aynıdır. Yeni tanımlanan Genişletilmiş Çokyüzlü Konik Fonksiyon (Extended Polyhedral Conic Function) denklem (4.11)'de sunulmuştur.

$$f_{\mathbf{w},\boldsymbol{\gamma},\mathbf{c},b}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) + \boldsymbol{\gamma}^{\mathrm{T}}|\mathbf{x} - \mathbf{c}| - b \qquad \text{(EPCF)} \tag{4.11}$$

EPCF fonksiyonu da aynı PCF gibi $z = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) - b$ hiperdüzleminin $z = \boldsymbol{\gamma}^{\mathrm{T}}|\mathbf{x} - \mathbf{c}| = \|\mathrm{diag}(\boldsymbol{\gamma})(\mathbf{x} - \mathbf{c})\|_1$ konisinin içinde kalan kesitidir. Buradaki koninin PCF konisinden farkı, diyagonal olarak ölçeklenmiş bir koni olmasıdır.

Denklem (2.12)'deki PCF fonksiyonunda $b > 0, \gamma > 0$ şartları tanımlanmıştı. Bu şartlarla beraber denklem (4.11)'de $|w_i| < \gamma_i, i = 1, \dots, d$ şartı da sağlanırsa, herhangi bir $\tau \in \mathbb{R}$ için denklem (4.11)'deki EPCF fonksiyonunda $f(\mathbf{x}) < \tau$ ile belirtilen bölge, $\mathbb{R}^d$'de dışbükey ve kompakt bir alan oluşturur (Şekil 4.3). EPCC'de de PCC sınıflandırıcısında yapıldığı gibi bu şartlar serbest bırakılıp, eğitim sürecinde sınıflandırıcının en uygun karar alanını bulması beklenmektedir.



(a) (b)

Şekil 4.3 $b > 0, \gamma > 0$, $|w_i| < \gamma_i, i = 1, \dots, d$ şartları ile $z = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) - b$ hiperdüzlem (yeşil) eğimi koninin (turuncu) yanal yüzeyinin eğiminden daha küçük olur ve pozitif karar bölgesi kompakt sınırlı bir alan oluşturur, b) diğer durumda karar bölgesi sınırlı olmaz, sonsuza uzar.

PCC sınıflandırıcısında uygulanan dönüşüm, benzer şekilde EPCC'de de uygulanmaktadır. Bu kapsamda denklem (4.12), (4.13) ve PCC'deki denklem (4.3) ($\tilde{b} = b$) dönüşümleri EPCC için uygulanır ve SVM yapısındaki denklem (4.4) elde edilir. Bu dönüşümle, denklem (4.5)'deki SVM kuvadratik programlama kullanabilir. Eğitim sonucunda öğrenilen $\tilde{\mathbf{w}}, \tilde{b}$ parametreleri ile $\mathbf{c}$ koni merkezi de kullanarak model oluşturulur. Burada dikkat çeken husus PCC'de $(d + 1)$ olan öznitelik boyut büyüklüğü EPCC'de $2d$'ye

çıkmıştır. Boyutun artması bir dezavantaj gibi görülse de bazı problemlerin çözümünde daha iyi başarım alınmaktadır.

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ |\mathbf{x} - \mathbf{c}| \end{pmatrix} \in \mathbb{R}^{2d} \tag{4.12}$$

$$\tilde{\mathbf{w}} = \begin{pmatrix} -\mathbf{w} \\ -\boldsymbol{\gamma} \end{pmatrix} \in \mathbb{R}^{2d} \tag{4.13}$$

Eğitim sürecinden sonra elde edilen model ile herhangi bir $\mathbf{x}_{test}$ örneğinin sınıflandırılması sağlanabilir. Öncelikle PCC'nin testinde uygulandığı gibi $\mathbf{x}_{test}$ örneğine de dönüşüm uygulanmalıdır. Bunun için modeldeki $\mathbf{c}$ merkezi kullanılarak denklem (4.12)'deki $\tilde{\mathbf{x}}_{\boldsymbol{test}} = \begin{pmatrix} \mathbf{x}_{\boldsymbol{test}} - \mathbf{c} \\ |\mathbf{x}_{\boldsymbol{test}} - \mathbf{c}| \end{pmatrix}$ dönüşüm uygulanır. Sonrasında $\tilde{\mathbf{x}}_{\boldsymbol{test}}$ kullanılarak denklem (4.4)'ten alınan değer ile sınıflandırma yapılır. Burada $\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} \geq 0$ ise $\mathbf{x}_{test}$ pozitif sınıfa, $\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} < 0$ ise $\mathbf{x}_{test}$ negatif sınıfa atanmaktadır.

PCC'de kullanılan L_2 normun karesi yöntemi EPCC içinde uyarlanabilir. Her ne kadar PCC'deki gibi EPCC'de bir L_2 norm elde edilemese de PCC ile analojiyi korumak adına bu yöntem normun karesi olarak kullanılmadan sadece L_2 olarak adlandırılmıştır. EPCC yönteminde de öznitelik vektörü, $\tilde{\mathbf{x}}$, denklem (4.14)'deki gibi oluşturularak, L_2 yöntemi elde edilir. Diğer süreçler EPCC'nin L_1 normunda olduğu gibi uygulanır.

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ (x_1 - c_1)^2 \\ \vdots \\ (x_d - c_d)^2 \end{pmatrix} \in \mathbb{R}^{2\mathrm{d}},\ \mathbf{x} = (x_1, \dots, x_d)^{\mathrm{T}} \in \mathbb{R}^d,\ \mathbf{c} = (c_1, \dots, c_d)^{\mathrm{T}} \in \mathbb{R}^d \tag{4.14}$$

4.3 Tek Sınıf Genişletilmiş Çokyüzlü Konik Sınıflandırıcı

EPCC esnekliğinden dolayı genelde PCC ve SVM'den daha iyi sonuçlar göstermektedir. Bunula birlikte EPCF'in sağladığı karar bölgeleri her zaman kapalı olmamaktadır. EPCF fonksiyonunun her bir bileşenin $|w_i| < \gamma_i \ , i = 1, \dots, d$ şartını sağlandığında pozitif karar bölgesi dışbükey ve sınırlı olmaktadır (Bkz. Şekil 4.3). Eğer pozitifler her yönden negatiflerle çevrelenmezse, pozitif karar bölgesinin negatifle çevrelenmeyen tarafından sonsuza uzama ihtimali bulunmaktadır. Bu durumda koniyi kesen hiperdüzlemin o eksenindeki eğimi, koninin aynı eksendeki yanal yüzeyinin eğiminden daha

büyük olmakta ve pozitif karar bölgesi kompakt, sınırlı ve dış bükey olamamaktadır. Sonuç olarak pozitifler sıkı bir şekilde çevrelenememektedir. Her ne kadar SVM'den daha küçük karar bölgeleri oluşsa da karşılaşılan durum tercih edilmeyen bir çözümdür. Eğer her bir eksende $|w_i| < \gamma_i, i = 1, \dots, d$ şartı zorlanırsa daha sıkı karar bölgeleri elde edilebilir. Bu sayede açık küme yaklaşımında ve sadece pozitif verilerin kullanıldığı tek sınıf sınıflandırmada daha iyi sonuçlar elde edilebilir.

EPCC'de ± 1 marj sistemi ağırlıkları sınırlayan ve de kötü bir çözümü önleyen tek şeydir ve bu marj sisteminin iyi şekilde uygulanabilmesi için negatif veriler gereklidir. EPCC'nin açık küme yaklaşımında ve sadece pozitif veriler ile daha iyi çalışabilmesi için pozitif karar bölgelerinin sıkı ve kompakt olması şarttır. Karar bölgesi i ekseni boyunca $O(b/\gamma_i)$ genişliğindedir, bu nedenle genişliği sınırlayabilmek için γ_i'nin sıfıra yaklaşması engellenmelidir. Bunu başarmanın en kolay yolu, ± 1 marj ölçeklemesini $b = 1$ ofset ile değiştirmek, γ_i için negatif maliyet cezası eklemek ve yeni pozitif-negatif $[0,1]$ marjının geometrik genişliğine maliyet cezası getirmektir. Böylece karar bölgesinin genişliğinin küçük olması ve kümelerin iyi ayrılması sağlanabilir. Denklem (4.15) ile verilen tek sınıf genişletilmiş çokyüzlü konik sınıflandırıcısı (One Class Extended Polyhedral Conic Classifier), bunları sağlamaktadır (Cevikalp ve Triggs, 2017 a).

$$\arg\min_{\mathbf{w},\boldsymbol{\gamma}} \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \frac{1}{n_+}\sum_{\mathrm{i}} \xi_i + \frac{1}{n_-}\sum_{\mathrm{j}} \xi_{\mathrm{j}} - \mathbf{s}^{\mathrm{T}}\boldsymbol{\gamma} \tag{4.15}$$

$$\text{s.t. } \mathbf{w}^{\mathrm{T}}(\mathbf{x}_i - \mathbf{c}) + \boldsymbol{\gamma}^{\mathrm{T}}|\mathbf{x}_i - \mathbf{c}| - 1 \leq \xi_i, i \in \mathrm{I}_+, \tag{4.16}$$

$$\mathbf{w}^{\mathrm{T}}\left(\mathbf{x}_j - \mathbf{c}\right) + \boldsymbol{\gamma}^{\mathrm{T}}\left|\mathbf{x}_j - \mathbf{c}\right| - 1 \geq 1 - \xi_{\mathrm{j}}, \mathrm{j} \in \mathrm{I}_-, \tag{4.17}$$

$$\xi_{\mathrm{i}}, \xi_{\mathrm{j}} \geq 0, \tag{4.18}$$

OC-EPCC çözümü için önerilen denklem (4.15)'te, denklem (4.5)'teki kuvadratik programlamadan farklı olarak; $\lambda, \mathbf{w}$ için bir düzenlileştirme parametresi, n_+ pozitif örnek sayısı, n_- negatif örnek sayısı, $\mathbf{s} > 0$ kullanıcı tarafından sağlanan ve azalan $\boldsymbol{\gamma}$ için bir maliyet ceza vektörü olarak kullanılmaktadır. Pozitif ve negatif örnekler $[0,1]$ marjının dışında kalmaya zorlanmaktadır, fakat $\xi_{i(j)}$ artıran değişkeni ile pozitif (negatif) örnekler için bu kural denklem (4.16) ve (4.17)'de esnetilmektedir. Denklem (4.15)'teki optimizasyon problemini çözmek için eğitim sürecinde Stokastik Gradyan (SG) yöntemi

kullanılmaktadır. Bu minimum probleminin çözümü ile $\mathbf{w}, \boldsymbol{\gamma}$ parametreleri bulunmakta ve sınıflandırma modeli oluşturulmaktadır. Denklem (4.15)'te, denklem (4.5)'teki kuvadratik programlamadaki gibi herhangi bir dönüşüm uygulanmamıştır.

SG yöntemini kullanan OC-EPCC algoritması; $\mathbf{w}_1, \boldsymbol{\gamma}_1, T > 0, \alpha_0 > 0, \epsilon_{\mathbf{w}} > 0,$ $\epsilon_{\boldsymbol{\gamma}} > 0$, $\mathbf{x}_i$ eğitim setindeki örnekler, n_+ pozitif örnek sayısı, n_- negatif örnek sayısı, $n = n_+ + n_-$, parametrelerinin başlangıç değerleri sağlanarak aşağıdaki gibi tanımlanmıştır:

$t \in 1, \dots, T$ kez *Adım-1,-2 ve -3*'ü tekrarla,

Adım-1: Bir önceki döngü parametrelerini sakla, öğrenme katsayısını güncelle,

$$\mathbf{w}_{t-1} = \mathbf{w}_t;$$

$$\boldsymbol{\gamma}_{t-1} = \boldsymbol{\gamma}_t;$$

$$\alpha_t \leftarrow \alpha_0/t;$$

Adım-2: Tüm örneklerin sıralamasını karıştır ve $i = 1{:}\, n$'e kadar *Adım-2.1* ve *2.2'*yi tekrarla,

Adım-2.1: Seçili örnek için gradyanları hesapla,

$$\mathbf{g}_{\mathbf{w}}^t = \begin{cases} \frac{\lambda \mathbf{w}}{n} + \frac{\mathbf{x_i}}{n_+}, eğer\ y_i = 1\ \&\ y_i(\mathbf{w}_t^{\mathrm{T}}(\mathbf{x}_i - \mathbf{c}) + \boldsymbol{\gamma}_t^{\mathrm{T}}|\mathbf{x}_i - \mathbf{c}| - 1) \geq 0 \\ \frac{\lambda \mathbf{w}}{n} - \frac{\mathbf{x_i}}{n_-}, eğer\ y_i = -1\ \&\ y_i(\mathbf{w}_t^{\mathrm{T}}(\mathbf{x}_i - \mathbf{c}) + \boldsymbol{\gamma}_t^{\mathrm{T}}|\mathbf{x}_i - \mathbf{c}| - \rho_t) \geq 0 \\ \frac{\lambda \mathbf{w}}{n}, \text{diğer durumlarda} \end{cases};$$

$$\mathbf{g}_{\boldsymbol{\gamma}}^t = \begin{cases} -\frac{s}{n} + \frac{\mathbf{x_i}}{n_+}, eğer\ y_i = 1\ \&\ y_i(\mathbf{w}_t^{\mathrm{T}}(\mathbf{x}_i - \mathbf{c}) + \boldsymbol{\gamma}_t^{\mathrm{T}}|\mathbf{x}_i - \mathbf{c}| - 1) \geq 0 \\ -\frac{s}{n} - \frac{\mathbf{x_i}}{n_-}, eğer\ y_i = -1\ \&\ y_i(\mathbf{w}_t^{\mathrm{T}}(\mathbf{x}_i - \mathbf{c}) + \boldsymbol{\gamma}_t^{\mathrm{T}}|\mathbf{x}_i - \mathbf{c}| - \rho_t) \geq 0 \\ -\frac{s}{n}, \text{diğer durumlarda} \end{cases};$$

Adım-2.2: Hesaplanan gradyanlarla parametreleri güncelle,

$$\mathbf{w}_t \leftarrow \mathbf{w}_t - \alpha_t \mathbf{g}_{\mathbf{w}}^t;$$

$$\boldsymbol{\gamma}_t \leftarrow \boldsymbol{\gamma}_t - \alpha_t \mathbf{g}_{\boldsymbol{\gamma}}^t;$$

Adım-3: Aşağıdaki şartlar oluştuysa döngüye devam etme,

$$\|\mathbf{w}_t - \mathbf{w}_{t-1}\| < \epsilon_{\mathbf{w}};$$

$$\|\boldsymbol{\gamma}_t - \boldsymbol{\gamma}_{t-1}\| < \epsilon_{\boldsymbol{\gamma}};$$

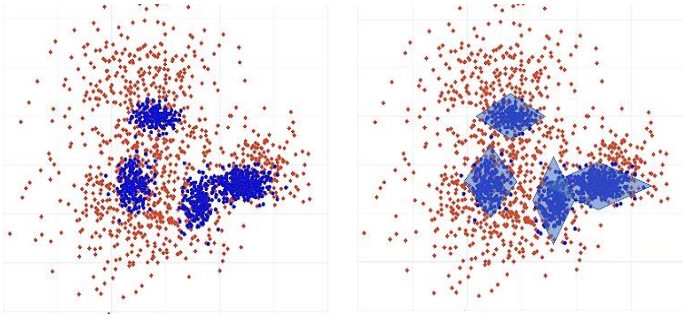
Yukarıdaki OC-EPCC algoritması ile $\mathbf{w}, \boldsymbol{\gamma}$ parametreleri bulunur. $b = 1$ olarak sabitlenmişti. Bundan dolayı oluşan model $\mathbf{w}, \boldsymbol{\gamma}, \mathbf{c}$ parametrelerini içerir. Bir $\mathbf{x}_{test}$ örneğinin sınıfını belirlemek için modeldeki parametreler ile denklem (4.11)'deki EPCF fonksiyonunun $f(\mathbf{x}_{test})$ değeri hesaplanır. $\mathbf{x}_{test}$ örneği, eğer $f(\mathbf{x}_{test}) \leq 0$ ise pozitif, $f(\mathbf{x}_{test}) > 0$ ise negatif sınıfa atanır. OC-EPCC'de SVM dönüşümü yapılmadığından PCF'in karar prosedürü uygulanır. Bu yönüyle PCC ve EPCC'nin test aşamalarından farklılık göstermektedir.

EPCC'de kullanılan L_2 yöntemini OC-EPCC'de de kullanmak mümkündür. Denklem (4.11), (4.16) ve (4.17)'de $|\mathbf{x} - \mathbf{c}|$ terimi yerine denklem (4.19)'daki terim kullanıldığında OC-EPCC-L_2 yöntemi elde edilir. Diğer tüm eğitim ve test süreci OC-EPCC-L_1 normunda olduğu gibi uygulanmaktadır.

$$\begin{pmatrix} (x_1 - c_1)^2 \\ \vdots \\ (x_d - c_d)^2 \end{pmatrix}, \quad \mathbf{x} = (x_1, \dots, x_d)^{\mathrm{T}} \in \mathbb{R}^d, \ \mathbf{c} = (c_1, \dots, c_d)^{\mathrm{T}} \in \mathbb{R}^d \tag{4.19}$$

4.4 Çok Merkezli Çokyüzlü Konik Sınıflandırıcı

Sınıflandırma probleminde karşılaşılan bir durum Şekil 4.4'teki gibi pozitif örneklerin aynı uzayda farklı bölgelerde kümelenmesidir. Örneğin bir insana ait yandan, arkadan ve önden görünümler veya farklı pozisyonlardaki vücut şekilleri öznitelik uzayında farklı bölgelerde kümelenebilmektedir. Bu durumda her bir küme için ayrı bir sınıflandırıcı kullanılması gerekir. Birden fazla sınıflandırıcı ise eğitim ve test sürecini hem yavaşlatabilmekte hem de karmaşık hale getirebilmektedir.



Şekil 4.4 Pozitif sınıf verilerinin farklı bölgelerde kümelendiği karmaşık yapılı bir veri (mavi: pozitif örnekler, kırmızı: negatif örnekler)

PCF'te bir adet koni tepe noktası mevcuttur. Pozitif verilerin bir merkez etrafında toplanması durumunda, diğer PCC yöntemleri (PCC, EPCC, OC-EPCC) çok iyi bir sınıflandırma sağlamaktadır. Bu kapsamda bahsedilen problemin çözümü için PCF'te bir tepe noktası yerine birden fazla tepe noktasının kullanılması önerilmiş ve denklem (4.20)'deki çok merkezli çokyüzlü konik fonksiyon (Multi Center Polyhedral Conic Function) ve bu fonksiyonu kullanan çok merkezli çokyüzlü konik sınıflandırıcı (Multi Center Polyhedral Conic Classifier) oluşturulmuştur.

$$f_{\mathbf{w},\boldsymbol{\gamma},\mathbf{c},b,n}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x}-\mathbf{c}_1) + \boldsymbol{\gamma}^{\mathrm{T}}\begin{bmatrix}\|\mathbf{x}-\mathbf{c}_1\|_1\\ \vdots \\ \|\mathbf{x}-\mathbf{c}_n\|_1\end{bmatrix} - b \qquad \text{(MCPCF) (4.20)}$$

MCPCC sınıflandırıcısında, denklem (2.15)'teki PCF'ten farklı olarak bir merkez yerine n adet merkez, $\mathbf{c}_i \in \mathbb{R}^d, i = 1, \dots, n$, kullanılmaktadır. $\mathbf{x} \in \mathbb{R}^d$ örneği için, bu merkezlerden uzaklıklar L_1 normuna göre hesaplanmakta ve $\boldsymbol{\gamma} \in \mathbb{R}^n$ ağırlık katsayısıyla f fonksiyonuna eklenmektedir. Diğer çokyüzlü konik sınıflandırıcılardan farklı olarak, merkezler pozitif örneklerin dağılım gösterdiği bölgelerden seçilirse yöntem doğrusal olmayan karar sınırları sağlayabilmektedir. PCF'ten diğer bir fark ise ilk terimde yer almaktadır. PCF'te tek merkez olduğundan, ilk terimde bu tek merkez, **x**'ten çıkarılmaktadır ($\mathbf{w}^{\mathrm{T}}(\mathbf{x}-\mathbf{c})$)**.** MCPCF'te *n* adet merkez olduğundan, pozitif eğitim örneklerinin ortalaması olan ve ayrı bir merkez olarak da seçilen $\mathbf{c}_1$, **x** test örneğinden ($\mathbf{w}^{\mathrm{T}}(\mathbf{x}-\mathbf{c}_1)$) çıkarılmıştır. Buradaki amaç diğer çokyüzlü konik sınıflandırıcıların kullandığı $\mathbf{c}_1$ merkezinin avantajlarını MCPCF'e dahil etmektir.

MCPCC'in eğitimi de aynı PCC eğitimi gibidir. PCC eğitimindeki yaklaşım kullanılarak doğrusal SVM karar fonksiyonuna dönüşüm yapılmaktadır. Böylece SVM için kullanılan matematiksel çözümlerin ve yaygın yazılımların MCPCC sınıflandırıcımız için de kullanılabilmesini sağlanmaktadır. Bu kapsamda denklem (4.21) ve (4.22)'deki dönüşüm ile PCC'deki denklem (4.3) dönüşümü ($\tilde{b} = b$) uygulanmaktadır. Dönüşüm sonrası elde edilen denklem (4.5) kullanılarak eğitim süreci gerçekleştirilmektedir. Eğitim sürecinde $\widetilde{\mathbf{w}}, \tilde{b}$ parametreleri öğrenilir. Bu parametreler ve n adet merkezle, $\mathbf{c}_i$, birlikte model oluşturulur.

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ \|\mathbf{x} - \mathbf{c}_1\|_1 \\ \vdots \\ \|\mathbf{x} - \mathbf{c}_n\|_1 \end{pmatrix} \in \mathbb{R}^{\mathrm{d+n}} \tag{4.21}$$

$$\tilde{\mathbf{w}} = \begin{pmatrix} -\mathbf{w} \\ -\boldsymbol{\gamma} \end{pmatrix} \in \mathbb{R}^{d+n} \tag{4.22}$$

MCPCC eğitim algoritması; $\mathbf{x}_i, i = (1, \dots, m)$, m adet eğitim verisi, y_i etiket, n koni merkez sayısı olmak üzere aşağıdaki gibidir:

Adım-1: Pozitif eğitim örneklerinin ortalamasını, $\mathbf{c}_1$, bul.

Adım-2: *k*-merkezli kümeleme ile $n-1$ adet küme merkezini, $\mathbf{c}_i$, $i = 2, \dots, n$, hesapla.

Adım-3: $\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ \|\mathbf{x} - \mathbf{c_1}\|_1 \\ \vdots \\ \|\mathbf{x} - \mathbf{c_n}\|_1 \end{pmatrix}$, $\tilde{\mathbf{w}} = \begin{pmatrix} -\mathbf{w} \\ -\boldsymbol{\gamma} \end{pmatrix}$, $\tilde{b} = b$ dönüşümlerini gerçekleştir

Adım-4: SVM sınıflandırıcısı kullanarak $\tilde{\mathbf{w}}, \tilde{b}$ parametrelerini hesapla.

Burada dikkat çeken husus da örneklerin öznitelik vektörünün $(d + n)$ boyutlu yeni öznitelik vektörüne transfer edilmesidir. Bu da MCPCC yönteminde merkez sayısının performans/doğruluk arasında bir denge parametresi gibi kullanılabilmesini sağlamaktadır. Eğer $n = 1$ seçilirse bir adet koni tepe noktası kullanılmış olur. Bu ise PCC sınıflandırıcımızı vermektedir. Bu açıdan MCPCC sınıflandırıcısı PCC sınıflandırıcısının genelleştirilmiş halidir denilebilir. Burada dikkat edilmesi gereken bir husus da artan n değeri ile hesaplama yükünün de artmasıdır.

Eğitim sürecinden sonra elde edilen model ile herhangi bir $\mathbf{x}_{test}$ örneği sınıflandırılabilir. Öncelikle PCC'nin testinde uygulandığı gibi $\mathbf{x}_{test}$ örneğine dönüşüm uygulanmalıdır. Bunun için modeldeki n adet merkezle denklem (4.21)'deki $\tilde{\mathbf{x}}_{\boldsymbol{test}}$ dönüşümü

uygulanır. Sonrasında denklem (4.4)'ten aldığımız değer ile sınıflandırma yapılır. Denklem (4.4)'te eğer $\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} \geq 0$ ise $\mathbf{x}_{test}$ pozitif sınıfa, $\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}_{test} + \tilde{b} < 0$ ise $\mathbf{x}_{test}$ negatif sınıfa atanmaktadır.

Merkez sayısı olan n, MCPCC algoritması dışında belirlenmesi gereken bir parametredir. Deneylerde, n değeri kademeli olarak artırılarak performans sonuçları gözlemlenmiş ve sonuçların değişiminin azaldığı ve doyum noktasına ulaştığı andaki n değeri seçilmiştir. n parametresi belirlenirken daha küçük ayrı bir veri kümesi kullanılarak daha hızlı sonuçlar elde edilmiştir.

Diğer bir MCPCC sınıflandırıcısı ise L_1 norm yerine L_2 (Öklid) norm kullanılmasıyla denklem (4.23)'teki gibi elde edilmektedir. Denklem (4.23)'teki $\|.\|$ işlemi, $\|\mathbf{u}\| = \sqrt{{u_1}^2 + \ldots + {u_d}^2}$, L_2 normunu temsil etmektedir. MCPCC-L_2 yöntemindeki dönüşüm ise denklem (4.24), (4.22) ve PCC'deki denklem (4.3) ($\tilde{b} = b$) kullanılarak gerçekleştirilir.

$$f_{\mathbf{w},\boldsymbol{\gamma},\mathbf{c},b,n}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{c}_1) + \boldsymbol{\gamma}^{\mathrm{T}} \begin{bmatrix} \|\mathbf{x} - c_1\| \\ \vdots \\ \|\mathbf{x} - c_n\| \end{bmatrix} - b \tag{4.23}$$

$$\tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} - \mathbf{c} \\ \|\mathbf{x} - \mathbf{c}_1\| \\ \vdots \\ \|\mathbf{x} - \mathbf{c}_n\| \end{pmatrix} \in \mathbb{R}^{d+n} \tag{4.24}$$

4.5 Koni Tepe Noktası Tahmini

Çokyüzlü konik sınıflandırıcılarda (PCC, EPCC ve OC_EPCC) koni tepe noktası olarak eğitim sınıfındaki pozitif verilerin ortalaması kullanılmaktadır. Tepe noktasının pozitif örneklerin ortalaması olarak seçilmesi, örneklerin bir arada ve kompakt bir yapıya sahip olduğu varsayılarak yapılmıştır. Ayrıca bu seçim sınıflandırıcıların hızlı çalışması açısından da tercih edilmektedir. Bu seçim zorunlu değildir, herhangi bir tepe noktası (medoit, vb.) seçilebilmektedir. Çokyüzlü konik sınıflandırıcılarda daha başarılı sonuç elde etmek için koni tepe noktasını optimal olarak bulan bir yöntem Dordinejad ve Çevikalp (2017) tarafından önerilmiştir. Bu yöntemde optimum tepe noktası, $\mathbf{c}$, pozitif örneklerin

doğrusal kombinasyonu olarak, $\mathbf{c} = \mathbf{X}_{pos}\boldsymbol{\alpha}$ şeklinde ifade edilmiştir. $\mathbf{X}_{pos}$ her sütununda pozitif örnekleri içeren bir matristir ve $\boldsymbol{\alpha}$ ise doğrusal kombinasyon katsayılarını göstermektedir. Bu şartlarda $\boldsymbol{\alpha}$ parametresi bulunarak $\mathbf{c}$ tepe noktası bulunabilir. Bu haliyle denklem (4.11)'deki EPCF denklemi (4.25)'teki gibi tekrar yazılabilir.

$$f_{\mathbf{w},\gamma,\boldsymbol{\alpha},b}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}) + \boldsymbol{\gamma}^{\mathrm{T}}\left|\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}\right| - b \tag{4.25}$$

En iyi koni tepe noktasıyla beraber optimum EPCF parametrelerini de bulabilmek için denklem 4.26'daki minimum probleminin çözülmesi gerekmektedir. Aynı çözümün PCF'te kullanılabilmesi için γ'nın skaler bir değer alması ve mutlak değer yerine L_1 norm kullanılması yeterlidir.

$$\arg\min_{\mathbf{w},\boldsymbol{\gamma},\boldsymbol{\alpha},b} \ \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\alpha} + C_{+}\sum_{i}\xi_i + C_{-}\sum_{j}\xi_j \tag{4.26}$$

$$s.t. \quad \mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}) + \boldsymbol{\gamma}^{\mathrm{T}}\left|\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}\right| - b - \xi_j \leq -1, \ j \in \mathrm{I}_{+}, \tag{4.27}$$

$$\mathbf{w}^{\mathrm{T}}(\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}) + \boldsymbol{\gamma}^{\mathrm{T}}\left|\mathbf{x} - \mathbf{X}_{pos}\boldsymbol{\alpha}\right| - b + \xi_i \geq +1, \ i \in \mathrm{I}_{-}, \tag{4.28}$$

$$\xi_i, \xi_j \geq 0 \tag{4.29}$$

Buradaki optimizasyon problemi tüm parametrelere göre dışbükey değildir. Fakat bazı parametreler sabitlenirse dışbükey olabilir. Bu amaçla problemin çözümü için almaşık optimizasyon (alternating optimization) yöntemi kullanılmaktadır. Burada önce $\mathbf{c}$ tepe noktasına başlangıç değeri atanır, sonra denklem (4.5)'teki gibi EPCF parametreleri $(\mathbf{w}, \boldsymbol{\gamma}, b)$ bulunur. Sonra sınıflandırıcı parametreleri sabitlenerek denklem (4.26)'daki minimizasyon probleminin $\boldsymbol{\alpha}$ parametresine göre türevi alınır ve SG yöntemiyle en iyi $\boldsymbol{\alpha}$ bulunur. Bulunan bu $\boldsymbol{\alpha}$ ile yeni merkez hesaplanır ve merkez sabit tutularak EPCF parametreleri tekrar hesaplanır. Bu prosedür parametrelerdeki değişim azalıncaya kadar iteratif şekilde devam eder. Görüldüğü üzere bu yaklaşımla iki tane minimizasyon problemi peş peşe çözülmektedir. Her ne kadar optimum çözüme ulaşılsa da hesaplama yükü artmaktadır. Bölüm 5.8'de koni tepe noktasının tahmin edilerek elde edilen deney sonuçları ile görsel deneyler sunulmuştur.

5. BULGULAR VE TARTIŞMA

Çalışmada önerilen yöntemler hem sentetik hem de gerçek veri tabanlarında nesne bulma, yüz doğrulama, görsel nesne sınıflandırma, açık küme tanıma ve çok sınıflı sınıflandırma problemlerinde test edilmiştir. Kullanılan veri tabanlarında diğer benzer yöntemlerle de sonuçlar elde edilerek önerilen yöntemlerle karşılaştırma yapılmıştır. Bu yöntemler aşağıda listelenmiştir:

- SVM
- 2. derece polinom ve Gauss Kernel SVM (KSVM)
- 1S-BFHC (Cevikalp vd., 2013 b)
- GEPSVM (Mangasarian ve Wild, 2006)
- SVDD (Tax ve Duin, 2004)
- Eklemeli Çekirdek (Additive Kernels) (Rahimi ve Recht, 2007)
- Hızlı R-CNN (Fast R-CNN) (Girshick, 2015)
- CPM (Kantchelian vd., 2014)
- Bire Karşı Küme Makinesi (1-vs-Set Machine) (Scheirer vd., 2013)
- PCF+Doğrusal SVM (Öztürk ve Çimen, 2019)

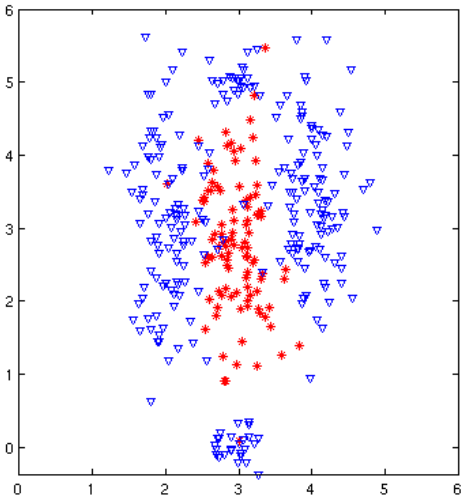
Dundar vd.nde (2008) bahsedilen çokyüzlü sınıflandırıcı için gerekli yazılım olmadığından bu yöntemle bir karşılaştırma yapılamamıştır. Sonuçlar Everingham vd.nin (2010) PASCAL VOC'ta belirttiği AP olarak hesaplanmıştır. Çok sınıflı deneylerde ise puanlama Sınıflandırma Oranı olarak veya her sınıfın AP değerinin ortalaması, mAP, olarak hesaplanmıştır. Çok sınıflı deneylerde bire-tümü yöntemiyle sınıflandırma yapılmıştır. Bu yöntemde bir sınıf pozitif olarak seçilmekte diğer tüm sınıflar da negatif olarak tanımlanmaktadır. Bu sayede sınıf adedi kadar sınıflandırıcı üretilmekte ve bir test örneği hangi sınıflandırıcıdan en yüksek puanı alırsa o sınıfa atanmaktadır.

MCPCC sınıflandırıcısı, pozitif örneklerin öznitelik uzayında farklı bölgelerde kümelendiği durumlarda PCC'nin daha iyi bir performans göstermesi amacıyla geliştirilmiştir. Bu kapsamda bahsedilen yapıya sahip olduğu değerlendirilen veri tabanları kullanılarak ayrı bir alt bölümde deneyler gerçekleştirilmiştir.

Koni tepe noktasının, **c**, optimum olarak bulunması için Bölüm 4.5'te önerilen yöntem, son bölümde görsel deneylerle ve gerçek veri tabanı deneyleri ile incelenmiştir.

5.1 Sentetik Veri Kümesi ile Deneyler

İlk deney sentetik veri kümesi ile yapılmıştır. Şekil 5.1'de oluşturulan iki boyutlu sentetik veriler; 250 pozitif (mavi), 750 negatif (kırmızı) örnekten oluşmaktadır. Eğitim kümesinde pozitif örnekler $\binom{3}{3}$ merkezli ve $\binom{0,1}{0,9}$ standart sapmalı, negatif örnekler ise birkaç merkezli ve aynı standart sapmalı gauss dağılımına sahiptir. Test verileri de aynı yöntemle oluşturulmuştur. Test sonuçları ise tüm yöntemler için AP skoru hesaplanarak Çizelge 5.1'de sunulmuştur. Ayrıca önerilen yöntemlerle oluşan karar bölgeleri Şekil 5.2'de gösterilmiştir.



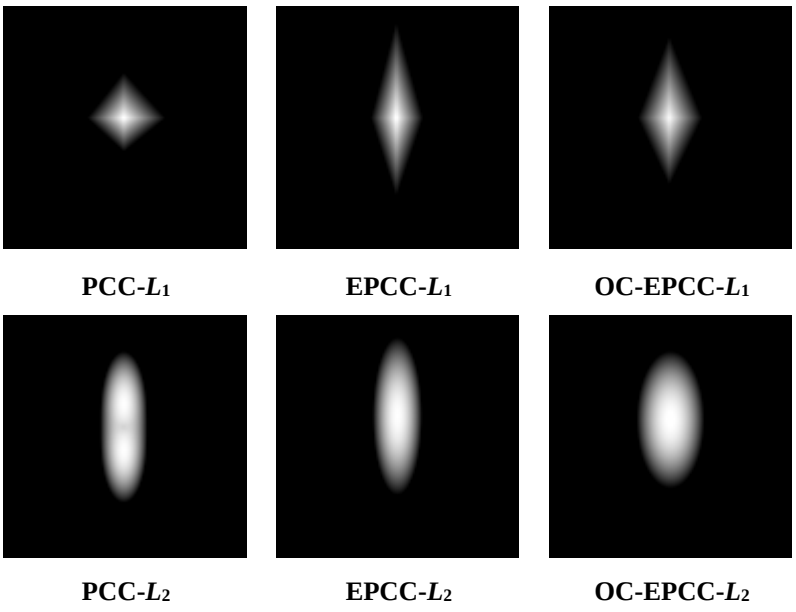
Şekil 5.1 İki Boyutlu sentetik veri kümesi, pozitif (kırmızı) veriler negatif (mavi) verilerle çevrelenmiştir (Cevikalp ve Saglamlar'dan, 2019).

Deneyde en iyi sonucu istatiksel Bayes sınıflandırıcı vermiştir. Hemen ardından PCC-L_2 yöntemi çok az bir farkla en iyi başarımı sağlamıştır. OC-EPCC sınıflandırıcıları ise sadece pozitif örnekler kullanılarak eğitilmiş olmasına rağmen oldukça iyi sonuçlar sergilemiştir. Buradaki problemde pozitif ve negatif örnekler doğrusal olarak ayrıştırılamadığından, SVM yönteminin çok kötü bir sonuç verdiği görülmüştür. Additive Kernel yöntemi, örnekleri 18 boyutlu bir uzaya taşıyarak iyi bir başarım gösterse de 3 ve 4 boyutlu uzaya taşıyan PCC ve EPCC yöntemleri kadar bir başarım sağlayamamıştır.

Şekil 5.2'de gösterilen karar bölgeleri incelendiğinde karar bölgelerinin pozitif verileri sıkıca çevrelediği ve L_1 yöntemlerle iki boyutta uçurtma şeklini andıran karar bölgeleri, L_2 yöntemlerle aynı boyutlara yakın eliptik karar bölgeleri elde edildiği görülmektedir.

Çizelge 5.1 İki boyutlu sentetik veri kümesi deneyindeki sonuçlar (AP, %)

Yöntem	**Skor (AP, %)**
Bayes Optimal	**90,89**
EPCC-L_1	86,62
EPCC-L_2	88,85
OC-EPCC-L_1	84,87
OC-EPCC-L_2	84,26
PCC-L_1	79,90
PCC-L_2	**90,64**
Additive Kernel	76,80
SVDD	71,14
GEPSVM	44,25
SVM	22,85

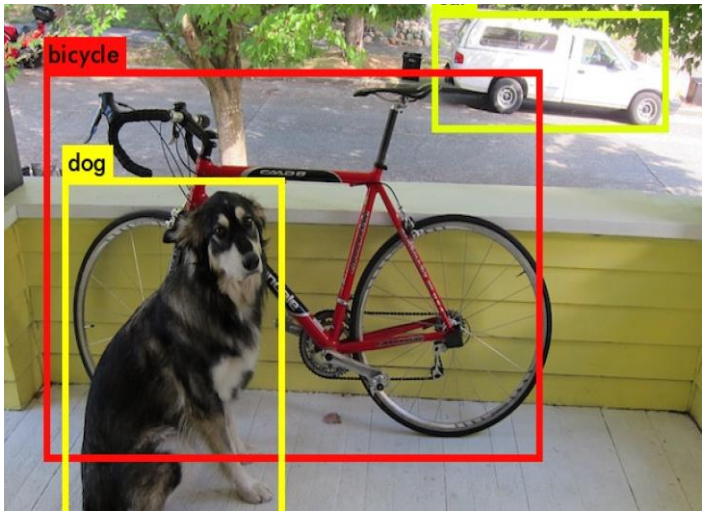


Şekil 5.2 Sentetik veri kümesinde karar bölgeleri, daha çok skor olan noktalar daha parlak gösterilmiştir.

5.2 Nesne Bulma Deneyleri

Nesne bulma (object detection), görüntü veya videolardaki belirli sınıflara ait nesneleri bulmaya yönelik bir bilgisayarla görme tekniğidir. İnsanlar resimlere veya

videolara baktığında, nesneleri birkaç saniye içinde tanıyabilir ve konumlandırabilir. Nesne bulmanın amacı, bilgisayara bu yeteneği öğretmektir. Bu kapsamda resim ve videodaki bulunan nesnelerin yerleri kuşatan kutu ile tespit edilir ve sınıfı belirtilir (Şekil 5.3). Nesne bulma deneyleri kapsamında yüz ve yaya bulma deneyleri ile MS COCO veri tabanında deneyler gerçekleştirilmiştir.



Şekil 5.3 Nesne bulma yönteminde görüntü veya videolardaki belirli sınıflara ait nesneler bulanarak kuşatan kutu içinde gösterilir. Redmon vd.den (2015) alınmıştır.

5.2.1 Yüz bulma deneyleri

Yüz bulma deneyleri için Jain ve Learned-Miller'in (2010) FDDB (Face Detection Dataset and Benchmark) veri tabanı ve ESOGU veri tabanı (http://mlcvdb.ogu.edu.tr/facedetection.html) kullanılmıştır. ESOGU veri tabanında yüksek çözünürlükte 667 renkli resim bulunmakta olup toplamda 2042 ön yüz çehresi etiketlenmiştir. Her iki veri tabanında çok farklı yüzler, değişik konumlarda ve ölçeklerde, karmaşık zeminlerde, farklı ışıklandırmada ve bazen de kısmen örtülmüş şekildedir.

Mevcut durumda açık olarak kullanılabilen yüz algılama veri tabanlarının azlığından dolayı, web ortamından 20 000 adet kesilmiş yüz resmi eğitim amacıyla kullanılmıştır. Her bir resim kırpılmış, ölçeklendirilmiş ve 35 x 28 çözünürlüğe dönüştürülmüştür. Negatif veriler için yüz resminin olmadığı 10 000 adet rastgele kompleks resim bölgesi seçilmiştir. Sonrasındaysa tüm hazırlanmış örnekler 620 boyutlu LBP (Local Binary Patterns) + HOG (Histogram of Oriented Gradients) öznitelik vektörü ile temsil edilmiştir.

Pozitif veriler, Bolla'nın (2013) spektral kümeleme (Spectral Clustring) yöntemiyle üç kümeye ayrılıp her bölüm ayrı ayrı eğitilmiştir. İlk eğitim sonrası, tüm resimlerin içerisinden zor negatifler tespit edilmiş ve negatif kümesine eklenmiştir. Son eklenen negatiflerle birlikte eğitim kümesinde yaklaşık 250 000 adet veriye ulaşılmıştır. Tüm bu verilerle son bir kez daha model eğitilmiştir. Test için ise Felzenszwalb vd.nin (2010) kullandığı standart kayan pencere (3 piksel yatay, 4 piksel dikey, 1.15 oranlı) yöntemi kullanılmıştır.

PASCAL VOC metrikleri ile bulunan sonuçlar Çizelge 5.2'de sunulmuştur. Sonuçlara Cevikalp ve Triggs (2012), Kalal vd. (2008), Viola ve Jones (2004) yöntemleri de eklenmiştir. Eklenen bu yöntemler diğerleriyle tam anlamıyla karşılaştırılamamaktadır. Çünkü bu yöntemler, herkese açık olmayan farklı eğitim setleri ve doğrusal olmayan son aşamalarıyla çok kademeli yapılar kullanmışlardır. Önerilen yöntemlerde ise sadece tek bir doğrusal aşama bulunmaktadır.

Çizelge 5.2 Yüz bulma deneyindeki sonuçlar (AP, %)

Yöntem	**FDDB**	**ESOGU**
EPCC-L_1	71,86	**89,11**
EPCC-L_2	65,33	72,40
PCC-L_1	67,17	78,79
PCC-L_2	54,25	65,30
SVM	37,60	47,66
Additive Kernel	55,70	78,70
1S-BFHC	70,5	80,0
Cevikalp ve Triggs (2012)	**74,1**	87,4
Kalal vd. (2008)	66,3	79,7
Viola ve Jones (2004)	67,6	76,2

FDDB veri tabanında EPCC-L_1 yöntemi Cevikalp-Triggs yönteminden sonra en iyi ikinci sonucu elde etmesiyle birlikte ESOGU veri tabanında en iyi sonucu vermiştir. Bu sonuçlar oldukça önemlidir, çünkü önerilen yöntemlerde herhangi bir kernel kullanılmamıştır ve yöntemler doğrusal bir biçime sahiptir. Diğer yöntemler ise doğrusal olmayan yöntemlerdir. L_2 yöntemiyle ile PCC ve EPCC'nin doğruluğunun azaldığı görülmüştür. 1S-BFHC yöntemiyse EPCC-L_1 yönteminden sonra en iyi sonucu vermiştir. Bu deneyde SVM en kötü sonucu sergilemiştir. Additive Kernel yöntemiyle, SVM

desteklenerek daha iyi sonuçlar elde edilse de PCC ve EPCC yöntemlerinden daha düşük bir performans göstermiştir.

5.2.2 Yaya bulma deneyleri

Yaya algılama deneyi Dalal ve Triggs (2005) tarafından sunulan INRIA yaya veri tabanında (http://pascal.inrialpes.fr/data/human/) gerçekleştirmiştir. Felzenszwalb vd.nin (2010) gizli eğitim (latent training) yöntemi, parça modelleri (part models) kullanılmadan sadece kök çifti (pair-root) kullanılarak uygulanmıştır. İlk olarak her bir kök çifti *k*-merkezli kümeleme yöntemiyle tespit edilmiştir. HOG öznitelikleri Felzenszwalb vd.nin (2010) yöntemindeki gibi (8x8 piksel hücreler, 8 adımlı pencereler, 1,07 oranlı piramit yapı) çıkarılmıştır. Karşılaştırma amacıyla önceden yayımlanmış Felzenszwalb vd.nin (2010) (HOG ile 8 parçalı 1 simetrik kök çifti), Hussain ve Triggs'in (2010) (iki aşamalı; doğrusal sonra HOG+LBP+LTP kullanan kuvadratik kaskat gizli SVM), Dalal ve Triggs'in (2005) yöntemlerinin sonuçları da Çizelge 5.3'e eklenmiştir.

Çizelge 5.3 INRIA yaya veri tabanı deneyindeki sonuçlar (AP, %)

Yöntem	Skor (AP, %)	Test Süresi (s)
EPCC-L_1	**85,6**	1,8
EPCC-L_2	85,0	1,6
PCC-L_1	83,6	1,8
PCC-L_2	84,3	1,5
SVM	80,4	1,6
Additive Kernel	80,9	19,1
1S-BFHC	78,5	1,6
Felzenszwalb vd. (2010)	**86,9**	3,5
Hussain ve Triggs (2010)	84,1	-
Dalal ve Triggs (2005)	75,0	-

Elde edilen sonuçlar ile her bir resim için harcanan ortalama test süreleri Çizelge 5.3'te sunulmuştur. Eğitilen yöntemlerin içinde EPCC-L_1 yöntemi en iyi sonucu vermiştir. Çok parçalı ve çok kök çiftini kullanan Felzenszwalb vd.nin (2010) yönteminden sonra en iyi sonuç EPCC-L_1 yöntemine aittir. Hussain ve Triggs'in (2010) yöntemi hem iki aşamalı hem de daha iyi öznitelik vektörlerine sahip olmasına rağmen, EPCC-L_1 yöntemi bu yöntemden daha iyi sonuç göstermiştir. EPCC-L_2, PCC-L_1 ve PCC-L_2 yöntemlerinin başarımı ise oldukça iyidir. Yüz bulma deneyinde görülen L_1 ve L_2 yöntemlerinin sonuçları

arasındaki farkın burada çok olmadığı görülmüştür. Önerilen yöntemlerin iyi sonuçlar almasının yanında dikkat edilmesi gereken diğer bir hususta test süreleri açısından neredeyse SVM kadar hızlı olmalarıdır. Additive Kernel yöntemiyse SVM sonucuna az bir katkı sağlasa da süre açısından değerlendirildiğinde oldukça yavaş kalmaktadır.

5.2.3 MS COCO veri tabanında deneyler

2014 yılında yayınlanan Microsoft COCO veri tabanıyla nesne algılama deneyi yapılmıştır. Lin vd. (2014) sunduğu MS COCO veri tabanı, 80 sınıftan oluşan, 330 000'den fazla resim içeren, 1,5 milyon tanımlı nesneye sahip büyük ölçekli bir veri tabanıdır. Derin öğrenmenin de giderek yaygınlaşması neticesinde, önerilen yöntemlerin derin öğrenmeyle iş birliği içinde daha iyi sonuçlar alınabileceği değerlendirilmiştir. Bazı derin öğrenme yöntemlerinin son katmanında SVM sınıflandırma amacıyla kullanılmaktadır. Bu kapsamda son katmanında SVM kullanan Girshick'in (2015) önerdiği Hızlı R-CNN (Fast R-CNN) yöntemi kullanılarak deney gerçekleştirilmiştir. Sonuçlar MS COCO hesaplama standardı olan, gerçek ve hesaplanan kuşatan kutunun kesişiminin birleşim üzerindeki oranı (IoU - Intersection over Union) [0,5:0,05:0,95] olduğundaki AP değerlerinin ortalaması, mAP, olarak hesaplanmıştır. Çizelge 5.4'te ise mAP@[.5, .95] ve PASCAL VOC standardı mAP@0.5 olarak belirtilmiştir.

Çizelge 5.4 MS COCO veri tabanı ile nesne algılama deneyindeki sonuçlar (mAP, %)

Yöntem	**mAP@0.5**	**mAP@[0.5,0.95]**
EPCC-L_1	37,2	18,4
Hızlı R-CNN (SVM)	35,1	17,8

Deneyde Hızlı R-CNN yöntemi Ren vd.nin (2017) yazılımı ile kullanılmış olup, Hosang vd.nin (2015) kuşatan kutu için çerçeve önerileri (selective search region proposals) kullanılmıştır. Kullanılan çerçeve önerileri her resim için yaklaşık 2000 adetten oluşmaktadır. Russakovsky vd.nin (2015) ImageNet veri tabanında önceden eğitilmiş VGG-16 derin öğrenme modeli, MS COCO veri tabanı kullanılarak hassas olarak yeniden eğitilmiştir. İlk 240 000 iterasyonda eğitim katsayısı 0,001, sonraki 80 000 iterasyonda ise 0,0001 olarak seçilmiştir. Model oluşturulduktan sonra ikinci aşama olarak, FC7 katmanındaki çıktılar öznitelik olarak kullanılmıştır. Bu aşamada SVM ve EPCC-L_1 sınıflandırıcısı ile sınıflandırma eğitimi gerçekleştirilmiştir. Negatif örnek sayısı hafıza

kısıtlaması nedeniyle 90 000 ile sınırlandırılmıştır. Elde edilen sonuç Çizelge 5.4'te sunulmuştur. Önerilen EPCC-L_1 sınıflandırıcısı, SVM yöntemini kullanılan Hızlı R-CNN yöntemine göre %2 daha başarılı olmuştur. Bulunan SVM sonuçları Ren vd.nin (2017) SVM sonuçlarından daha düşük çıkmıştır. Bu farkın Ren vd.nin (2017) 8 GPU kullanılması ile farklı eğitim parametrelerinden kaynaklanmış olabileceği değerlendirilmiştir. Bu deneyde ise yayınlanan 1 GPU kullanan yöntem ve parametreler kullanılmıştır.

5.3 Yüz Doğrulama Deneyleri

Yüz doğrulama (face verificaiton) deneyi için Beveridge vd.nin (2013) oluşturduğu Point-and-Shoot Face Recognition Challenge (PaSC) veri tabanı kullanılmıştır. PaSC veri tabanı 265 insana ait 2802 video içermektedir. Videoların yarısı sabit kamera (control) ile çekilmiş olup, diğer yarısı ise aynı sahnelerinin el kamerası (handheld) ile çekilmiş halidir. Deneydeki amaç hedef (target) ve sorgu (query) videolarının aynı kişiye ait olup olmadığını bulmaktır. Deneyler iki çeşittir: control-control ve handheld-handheld deneyleri. Hedef ve sorgu videoları aynı video setinden seçilir. Hedef video, control videolarından ise sorgu da control videolarından seçilmektedir.

Yöntemleri test etmek için Huang vd.nin (2015) kullandığı prosedür ve CNN öznitelik vektörleri kullanılmıştır. Öncelikle tüm videoların diğer videolar ile birebir benzerliği hesaplanmış ve benzerlik matrisi bulunmuştur. Sonrasında bu matris kullanılarak ROC eğrisi (receiver operating characteristic curve) oluşturulmuştur. ROC eğrisi denklem (2.19) ve (2.20)'de tanımlanan yanlış pozitif oranına (false positive rate) ve doğru pozitif oranına (DPO) (true positive rate) göre oluşturulmaktadır. Yanlış pozitif oranının 0,01 olduğundaki DPO değeri deneylerde hesaplanmış ve karşılaştırma değeri olarak kullanılmıştır. Karşılaştırmalar hem bu DPO değerine göre hem de ortalama mAP skoruna göre yapılmıştır.

Benzerlik hesaplanırken iki farklı yaklaşım kullanılmıştır. İlk yaklaşımda bir hedef ve bir sorgu videosundan gelen verilerle ikili sınıflandırıcı eğitilmiştir. Bulunan marjın $(1/\|\mathbf{w}\|)$ tersi $(\|\mathbf{w}\|)$ benzerlik skoru olarak kullanmıştır. Marj ne kadar küçükse benzerlik skoru da o kadar büyük olmaktadır. Burada dikkati çeken diğer bir husus da sadece iki video verisi kullanıldığından, video verileri arasında iyi bir ayrım olmakta ve iki sınıf bir

hiperdüzlemle kolaylıkla ayrılabilmektedir. Bu yaklaşım, çok sınıflı sınıflandırma problemlerinde ikili sınıflandırıcıların kullanıldığı bire-bir (one against one-OAO) yöntemine benzediği için bu yaklaşım "bire-bir" olarak tanımlanmıştır.

İkinci yaklaşımda ise sorgu kümesindeki tüm veriler negatif olarak belirlenip, hedef videodaki veriler pozitif alınarak ikili sınıflandırıcı eğitilmiştir. Benzerlik matrisi oluşturulurken, ayırım yapan hiperdüzleme en yakın 10 sorgu videosunun ortalaması alınmıştır. Bu yaklaşım ise bire-tümü (one against rest-OAR) olarak tanımlanmıştır. Bire-bir yaklaşımından farklı olarak bire-tümü yaklaşımında, pozitif ve negatif sınıfları doğrusal bir hiperdüzlemle ayırmak daha zordur. Bu durumda ise daha iyi bir başarım için PCC ve EPCC gibi sıkı karar bölgeleri sağlayan yöntemleri kullanılmak avantaj sağlamaktadır.

Çizelge 5.5 PaSC veri tabanı ile yüz doğrulama deneyindeki sonuçlar

	Bire-Bir				**Bire-Tümü**			
	Doğru Pozitif Oranı (DPO, %)		**Ortalama Hassasiyet (mAP, %)**		**Doğru Pozitif Oranı (DPO, %)**		**Ortalama Hassasiyet (mAP, %)**	
Yöntem	**Control**	**Handheld**	**Control**	**Handheld**	**Control**	**Handheld**	**Control**	**Handheld**
EPCC-L_1	88,94	75,20	**71,00**	57,15	91,47	81,99	68,07	64,47
EPCC-L_2	88,54	74,46	70,22	56,59	**91,63**	**82,13**	**73,82**	**64,49**
PCC-L_1	**89,33**	74,97	70,53	**57,30**	90,59	79,20	72,58	61,97
PCC-L_2	86,17	75,58	67,14	56,42	91,25	80,87	72,97	63,15
SVM	86,68	74,57	69,73	56,98	78,06	71,48	58,26	51,04
Additive Kernel	86,07	70,80	67,96	53,59	88,76	76,34	64,71	59,75
CERML-EG (Huang vd., 2015)	80,11	77,37	-	-	-	-	-	-
DAN (Rao vd., 2017)	**92,06**	80,33	-	-	-	-	-	-

Control ve handheld deneylerinde elde edilen sonuçlar Çizelge 5.5'te sunulmuştur. Çokyüzlü konik karar bölgelerinin bire-tümü yaklaşımında daha iyi sonuçlar verdiği görülmektedir. En yüksek başarım EPCC-L_2 yöntemiyle bire-tümü yaklaşımında elde edilmiştir. Öyle ki elde edilen sonuç, Huang vd.nin (2015) bulduğu sonuçlardan bile yüksektir (control %80,11, handheld %77,37). Şu an PaSC handheld'de en yüksek skoru elde eden Rao vd.nin (2017) %80,33 sonucunu geçmiş, control'deki %92,06'ya ise oldukça yaklaşılmıştır. Bire-bir yönteminden ziyade bire-tümü yönteminde daha iyi sonuçlar elde edilmektedir. Bunun nedeni ise çok fazla negatif örnek ile karar bölgelerinin daha sıkı olması ve pozitifleri sarmasıdır. Bire-bir yaklaşımında ise karar bölgeleri daha esnektir. Bu ise bize

daha düşük skorlar vermektedir. SVM ise bire-bir yaklaşımında bire-tümünden daha iyi sonuç vermektedir. Bunun nedeni ise bire-bir yaklaşımında SVM'in doğrusal bir hiperdüzlem ile ayrımı başarmasıdır.

5.4 Görsel Nesne Sınıflandırma Deneyleri

Görsel nesne sınıflandırma (visual object classification) deneylerinde bir resimde belirli sınıfa ait nesnenin olup olmadığı bulunur. Nesnenin resim içindeki yeri önemli değildir. Bu kapsamında çok sınıflı PASCAL VOC 2007, Caltech-256 ve CIFAR-100 veri tabanları kullanılarak deneyler yapılmıştır.

5.4.1 PASCAL VOC 2007

PASCAL VOC 2007 görsel nesne sınıflandırma veri tabanında evrişimli sinir ağlarından elde edilen öznitelik vektörü ile deney yapılmıştır. Krizhevsky vd.nce (2012) tanımlanan AlexNet yapısının, Jia vd.nin (2014) AlexNet CNN Caffe uygulaması kullanılarak, ILSVRC2012'de önceden eğitilmiş modeli ile öznitelik vektörleri çıkarılmıştır. Her bir resim 256×256 olacak şekilde yeniden boyutlandırılmış ve sonuçta 4096 boyutlu öznitelik vektörü elde edilmiştir. Literatürde bulunan ve bu deneyde test edilen diğer yöntemlerle daha iyi bir karşılaştırma yapabilmek için, mevcut ILSVRC öznitelikleri PASCAL veri tabanında herhangi bir ince ayar eğitiminden geçmeden kullanılmıştır.

Deney sonuçları her sınıf için PASCAL VOC ölçeği AP skoru olarak Çizelge 5.6'da sunulmuştur. Ayrıca tüm sınıf ortalamaları mAP olarak son sütunda belirtilmiştir. Additive Kernel ve 2. derece polinom Kernel SVM (KSVM) yöntemleriyle kıyaslandığında, önerilen yöntemler çoğu sınıfta oldukça başarılı sonuçlar vermiştir. En iyi sonucu hem pozitif hem negatif örneklerle eğitilen OC-EPCC-L_1 yöntemi göstermiştir. OC-EPCC-L_1 yöntemi, doğrusal SVM'e göre ortalama %4 daha iyi sonuç vermiş, bazı sınıflarda ise (bottle, bus, chair, dining table, dog, potted-plant, sofa, tv monitor) bu oran %5'e kadar çıkmıştır. Doğrusal SVM'in 3 katı vektör boyutu kullanan Additive Kernel yöntemi, doğrusal SVM'in sonuçlarını bir miktar iyileştirmiştir.

Çizelge 5.6 PASCAL VOC 2007 veri tabanı ile görsel nesne sınıflandırma deneyindeki sonuçlar (AP, %)

Yöntem	Aeroplane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dining Table	Dog	Horse	Motorbike	Person	Potted Plant	Sheep	Sofa	Train	TV Monitor	Ortalama (mAP)
OC-EPCC-L_1	85,1	79,7	82,9	81,3	36,4	**69,5**	83,2	80,7	**57,7**	61,6	**70**	**79,9**	**83,2**	74	**90,4**	51	73,4	58,6	84,5	66,7	**72,5**
EPCC-L_1	**87,2**	**80**	**83,3**	80,9	35,9	66,5	83,4	80,9	56,5	59,4	68,7	78,5	82,6	73,8	90,1	49,7	71,3	57,1	86,5	66,6	72
PCC-L_1	86,3	79	83	80,5	35,3	65,8	83,4	80,2	56,1	60,3	68	77,2	81,8	73,3	89,8	47,9	70,8	55,6	85,9	66,4	71,3
SVM	87	75,7	81,7	80,4	31,2	63,6	80,4	79,1	47,1	58,1	64,2	74	81	73	87,4	41,3	68,5	50,6	86,3	61,4	68,6
KSVM	83,9	77,3	82,2	**81,8**	**38,7**	**69,5**	81,9	79,6	57,5	60,2	69,8	79,2	79,1	71,2	89	**52,6**	**73,8**	**59,3**	84,8	**69,7**	72,1
Additive Kernels	86,6	78,5	83	81,2	35,6	68	82	**81,5**	51	**63,1**	65,5	76,2	82,7	**74,9**	88,7	47,3	72,7	54	**86,7**	64,2	71,2
1S-BFHC	85,9	74	79,9	77,4	30,3	63	78,5	78	46,2	56,6	62	72	79,7	71,9	83,2	39,2	63,1	51	84,4	59,5	66,8
GEPSVM	36,2	21,9	45,1	26,4	10,3	27	34,1	21,9	29	39,9	32	22,2	32	19,6	53,9	15,4	27,2	14,3	39	25,8	28,7
SVDD	65,5	32,4	25	26	21,5	31,2	37,1	48,7	28,3	23,1	17,7	25,5	39,3	31,8	58,8	12,3	21,2	18,5	59,2	25,5	32,4

5.4.2 Caltech-256 ve CIFAR-100

Bu bölümde Caltech-256 (http://www.vision.caltech.edu/Image_Datasets/Caltech256) ve CIFAR-100 (https://www.cs.toronto.edu/~kriz/cifar.html) çok sınıflı veri tabanlarında sınıflandırma deneyleri yapılmıştır. Caltech-256 veri tabanında sınıflandırma deneyi standart protokol ile resimlerin Fisher Vektör (FV) tanımlayıcısı kullanılarak yapılmıştır. Standart protokolde, her sınıftan rastgele 60 resim alınmış ve 30 eğitim, 30 test olarak bölünmüş; sonrasında ise test ve eğitim verileri yer değiştirerek tekrar eğitim ve test yapılmıştır. Her resim başına yaklaşık 164 000 boyutlu FV tanımlayıcı çıkarılmıştır.

CIFAR-100 veri tabanı ise 100 adet sınıftan oluşmakta olup; 50 000 eğitim, 10 000 test örneğine sahiptir. CIFAR-100 veri tabanı deneyinde, 4096 boyutlu olan ince ayarlanmış CNN öznitelikleri kullanılmıştır.

Deneyler çok sınıflı bir problem olduğundan performans metriği olarak AP yerine denklem (2.23)'te belirtilen sınıflandırma oranı kullanılmış olup bulunan sonuçlar Çizelge 5.7'de özetlenmiştir. Additive Kernel yöntemi, Caltech-256 veri tabanında en iyi sonucu göstermiştir. EPCC-L_1, EPCC-L_2 ve KSVM (ikinci dereceden polinom kernel) yöntemleri ise CIFAR-100 veri tabanında en iyi performansı göstermiştir. Additive Kernel yönteminde vektör boyutu Caltech-256'da 3 katına, CIFAR-100'de 5 katına çıkmıştır. Aynı

şekilde KSVM'de yeni vektör boyutu CIFAR-100'de $\binom{d+2-1}{2} = \binom{4096+2-1}{2} = 8\,390\,656$ olmaktadır. Kantchelian vd.nin (2014) CPM yöntemi Caltech-256'da en iyi ikinci sonucu vermiştir. Her ne kadar önerdiğimiz yöntemler Caltech-256'da Additive Kernel ve CPM metotlarını geçemese de diğer yöntemlerden oldukça iyi bir sonuç sergilemiştir. Özellikle 164 000 öznitelik boyutlu Caltech-256'da PCC yöntemi SVM'den sadece bir boyut fazla olmasına rağmen çok daha iyi bir sonuç göstermiştir.

Çizelge 5.7 Caltech-256 ve CIFAR-100 veri tabanı ile görsel nesne sınıflandırma deneyindeki sonuçlar (sınıflandırma oranı, %)

Yöntem	**Caltech-256**	**CIFAR100**
EPCC-L_1	40,1±0,6	**86,4**
EPCC-L_2	40,0±0,7	**86,4**
PCC-L_1	40,4±0,7	86,2
PCC-L_2	38,4±0,4	85,4
SVM	37,6±0,7	85,8
KSVM (Polinom)	38,8±0,7	**86,4**
Additive Kernel	**42,6±0,7**	73,3
1S-BFHC	38,3±1,0	85,6
GEPSVM	13,3±0,6	74,5
SVDD	9,9±0,2	48,8
CPM	41,7±3,7	65,9

5.5 UCI Veri Havuzu Verileri ile Çok Sınıflı Deneyler

UCI veri havuzundan (https://archive.ics.uci.edu/ml/datasets.php) ikili ve çok sınıflı veri tabanlarından 7 adeti (Ionosphere, Iris, Letter Recognition-LR, Multiple Features-MF, Pima Indian Diabetes-PIMA, Wine ve Wisconsin Diagnostic Breast Cancer-WDBC) kullanılarak deneyler yapılmıştır. Kullanılan veri tabanlarının özellikleri Çizelge 5.8'de belirtilmiştir. Deneyler Bölüm 2.5.1 açıklanan 10 çapraz katman doğrulama (10 fold-cross-validation) yöntemiyle tekrarlanmıştır. Ayrıca AP metriği yerine çok sınıflı problemler için kullanılan denklem (2.23)'teki sınıflandırma oranı kullanılmıştır.

Çizelge 5.8 UCI veri tabanlarının özellikleri

Veri Taban	Sınıf Sayısı	Örnek Sayısı	Öznitelik Boyutu
Ionosphere	2	351	34
Iris	3	150	4
LR	26	20 000	16
MF	10	2000	240
PIMA	2	768	8
Wine	3	178	13
WDBC	2	569	30

Çizelge 5.9 UCI veri tabanı ile görsel nesne sınıflandırma deneyindeki sonuçlar (sınıflandırma oranı, %)

Yöntem	Ionosphere	Iris	WDBC	PIMA	Wine	MF	LR
EPCC-L_1	92,0 ±5,5	**98,0** ±3,2	98,0 ±1,6	76,5 ±3,2	97,5 ±3,2	97,2 ±1,3	79,6 ±1,4
EPCC-L_2	92,0 ±5,5	**98,0** ±3,2	97,9 ±1,6	76,9 ±4,2	97,5 ±3,2	97,2 ±1,3	79,3 ±1,4
PCC-L_1	**92,3,** ±4,4	96,7 ±4,7	97,4 ±2,1	**77,3** ±2,5	98,1 ±3,0	96,8 ±1,3	68,6 ±0,8
PCC-L_2	89,9 ±6,9	96,7 ±4,7	97,2 ±1,9	70,1 ±4,9	95,0 ±5,8	95,2 ±1,4	63,6 ±1,3
SVM	87,3 ±7,0	94,7 ±4,9	97,7 ±2,0	76,8 ±3,1	96,9 ±4,4	93,9 ±1,1	59,8 ±1,7
KSVM (polinom)	91,7 ±5,8	**98,0** ±4,5	95,1 ±2,0	72,5 ±5,0	96,3 ±5,3	**98,2** ±0,7	**95,3** ±0,7
GEPSVM	74,8 ±5,8	97,3 ±4,7	89,1 ±5,5	74,7 ±4,4	80,7 ±13,7	53,8 ±4,0	30,5 ±1,1
1S-BFHC	86,8 ±7,2	94,0 ±6,6	97,4 ±2,1	76,0 ±5,0	**98,8** ±2,6	93,8 ±1,5	25,3 ±0,8
SVDD	79,4 ±10,5	91,3 ±7,0	89,8 ±4,8	59,0 ±8,4	90,0 ±8,9	80,1 ±3,5	37,5 ±1,6
Additive Kernels	86,8 ±7,3	96,0 ±4,7	96,3 ±2,3	77,1 ±3,3	96,3 ±4,4	95,1 ±1,0	78,3 ±1,2
CPM	85,2 ±4,9	83,3 ±12,7	91,86 ±2,5	60,7 ±11,3	96,9 ±4,4	96,1 ±1,3	52,9 ±6,0
SPLA1	88,0 ±6,0		93,8 ±2,9	76,9 ±0,7			
SPLA2	90,6 ±1,2		95,8 ±0,4	76,8 ±0,6			
Batch Polyceptron	89,7 ±1,3		**98,5** ±0,1				

Alınan sonuçlar Çizelge 5.9'da sunulmuştur. Manwani ve Sastry'nin (2010) SPLA1 ile SPLA2, Manwani ve Sastry'nin (2011) Batch Polyceptron yöntemi için yazılım olmadığından sonuçlar ilgili makalelerden alınmıştır. Belirtilen makalelerde ise kullandığımız tüm veri tabanları için sonuçlar olmadığından sadece eşleşen veri tabanlarındaki sonuçlar tabloya eklenmiştir. EPCC, PCC yöntemleri ile KSVM (ikinci dereceden polinom kernel), 7 veri tabanının üçünde en iyi sonucu göstermiştir. Önerdiğimiz yöntemler en iyi sonuçları veya ikinci en iyi sonuçları vermiştir. SVM ile karşılaştırdığımızda ise yöntemlerimiz oldukça başarılıdır. Özellikle LR veri tabanında EPCC yöntemi SVM'den %20 daha iyi sonuç vermiştir. Manwani ve Sastry'nin (2011) Batch Polyceptron yöntemi ise WDBC veri tabanındaki en iyi sonuca sahiptir. GEPSVM ve SVDD ise en düşük sonuçları göstermiştir.

5.6 Açık Küme Tanıma Sistemi ile Deneyler

Kapalı küme sınıflandırma uygulamalarında, test esnasında eğitim sürecinde sınıflandırıcının öğrendiği sınıflar haricinde sınıf kullanılmamaktadır. Test sürecinde sınıflandırıcı sadece öğrendiği sınıflardan sorgulanmaktadır. Eğitimde hiç görmediği sınıftan bir örneği, test sırasında dışlamamakta ve eğitim örneklerinden çok uzakta olsa dahi, öğrendiği sınıflardan birine atamaktadır. Scheirer vd.nin (2013) önerdiği açık küme tanıma yaklaşımında, eğitim sürecinde kullanılmayan negatif sınıf örnekleri test sırasında kullanılmaktadır. Bu sayede eğitim sürecinde öğrenilmemiş örnekler ile yöntemin başarısı daha gerçekçi bir yaklaşımla değerlendirilmektedir. Bu bölümde Scheirer vd.nin (2013) kullandığı veri tabanında, açık küme yaklaşımına göre tekrardan düzenlediğimiz CIFAR-10 ve USPS veri tabanlarında deneyler gerçekleştirilmiştir. Açık küme deneylerinde kernel SVM (KSVM) olarak ikinci dereceden polinom kernel kullanılmıştır.

5.6.1 Açık Küme Deneyi

Açık küme deneyinde Scheirer vd.nin (2013) kullandığı 212 sınıflı veri tabanı kullanılmış ve aynı protokol uygulanmıştır. Scheirer vd.de (2013) Caltech-256 ve ImageNet (http://www.image-net.org) veri tabanları kullanılarak HOG ve LBP öznitelik vektörlerine benzer bir veri tabanı oluşturulmuştur. Her pozitif sınıf için Caltech-256 veri tabanından mevcut sınıfa ait rastgele 30 pozitif örnek ile diğer sınıflara ait 30 negatif örnek (rastgele seçilmiş 6 sınıftan beşer örnek) seçilmiştir. Test için mevcut sınıfa ait 30 farklı pozitif örnek ve 6330 negatif örnek (6 tane önceden seçilmiş sınıf ile ImageNet veri tabanından rastgele seçilmiş 206 sınıftan). Bu prosedür 5 defa tekrarlanmış ve hesaplanan AP değerlerinin ortalaması alınarak Çizelge 5.10'da sunulmuştur. Deneyde bu veri tabanından sadece 4 sınıf seçilmiştir (Leopard, Face, Airplane ve Chandelier). OC-EPCC ve diğer yöntemlerde hem pozitif hem negatif örnekler eğitim sırasında kullanılmıştır. Çizelge 5.10'daki sonuçlar incelendiğinde PCC ve EPCC yöntemleri oldukça iyi sonuçlar göstermiştir. OC-EPCC ile de (Airplane sınıfı dışında) sonuçlar geliştirilmiştir. Leopard sınıfında OC-EPCC-L_1, EPCC-L_1'dan %7'e yakın daha iyi sonuç vermiştir. Additive Kernel ve 1-vs-Set Machine yöntemleri de birbirine yakın ve iyi sonuç sergilemiştir, fakat önerilen yöntemler kadar bir başarım sağlayamamıştır. GEPSVM ve SVDD ise en kötü sonucu sergilemiştir.

Çizelge 5.10 Açık küme deneyindeki sonuçlar (AP, %)

Yöntem	Leopard	Face	Airplane	Chandelier
OC-EPCC-L_1	**76,6±2,9**	**70,0±2,8**	13,6±1,5	6,0±1,2
OC-EPCC-L_2	68,8±1,9	**70,0±6,7**	9,0±1,8	**7,9±2,1**
EPCC-L_1	69,8±7,9	69,7±2,8	15,5±2,8	5,6±0,6
EPCC-L_2	64,2±16,4	65,5±6,7	14,1±4,7	4,1±0,8
PCC-L_1	65,3±7,6	68,1±3,3	**15,7±3,2**	5,4±0,8
PCC-L_2	63,8±9,4	50,9±8,9	8,4±2,8	2,8±0,7
1-vs-Set Machine	76,5±6,8	60,2±3,8	12,0±0,9	4,9±1,1
1S-BFHC	62,6±12,7	59,1±3,1	12,4±1,8	4,8±0,7
SVM	63,2±13,1	61,5±4,3	12,0±1,6	4,8±0,5
KSVM (polinom)	68,7±5,8	63,0±2,2	9,3±1,1	7,2±0,9
GEPSVM	2,0±1,0	8,4±7,8	6,8±1,1	2,6±0,5
SVDD	3,6±0,8	2,0±0,4	3,6±0,5	3,3±0,7
Additive Kernels	76,5±6,8	60,2±3,8	12,0±0,9	4,9±1,1

5.6.2 Açık Küme CIFAR-10

Bu deneyde CIFAR-10 (https://www.cs.toronto.edu/~kriz/cifar.html) veri tabanındaki resimler kullanılmıştır. CIFAR-10 veri tabanı her bir sınıfta 6000 resim bulunan 10 sınıflı bir veri tabanıdır. Örnekler 32×32 boyutlu renkli resimlerden oluşmaktadır. Toplamda 60 000 resmin 50 000 tanesi eğitim, 10 000 tanesi test amacıyla kullanılmaktadır.

Açık küme deneyi kapsamında CIFAR-10 veri tabanındaki resimlerin CNN öznitelik vektörleri çıkarılmıştır. Eğitim sürecinde 10 sınıftan rastgele 3 sınıf seçilmiş ve sınıflandırıcı sadece bu 3 sınıfa göre bire-tümü yaklaşımıyla eğitilmiştir. Test aşamasında ise 10 sınıfın hepsi kullanılmıştır. Üç sınıfa ait AP değerlerinin ortalaması (mAP) hesaplanmış ve tüm bu prosedür 10 defa tekrarlanmıştır. 10 tekrarda elde edilen sonuçların ortalaması Çizelge 5.11'de sunulmuştur. Deneyde PCC-L_1 en iyi sonucu vermiştir. CIFAR-10 açık küme deneyinde önerilen yöntemler diğer yöntemlerden belirgin bir biçimde iyi sonuçlar vermektedir. Bu ise beklenen bir sonuçtur. Çünkü önerilen yöntemler yarı uzaylı bir karar bölgesinden ziyade kapalı ve kompakt bir karar bölgesi sağlamaktadır. Bu sayede bilinmeyen örnekleri dışlamayı daha iyi yapabilmektedir. Bu deneyde GEPSVM ve SVDD ise en kötü sonuçları sergilemiştir.

Çizelge 5.11 CIFAR-10 veri tabanı ile açık küme deneyindeki sonuçlar (mAP, %)

Yöntem	**Skor (mAP, %)**
EPCC-L_1	88,85
EPCC-L_2	88,85
PCC-L_1	**90,63**
PCC-L_2	87,90
OC-EPCC-L_1	88,64
OC-EPCC-L_2	88,64
SVM	79,07
KSVM (polinom)	84,27
Additive Kernel	84,02
CPM	81,05
1-vs-Set Machine	79,11
GEPSVM	36,18
SVDD	35,66

5.6.3 Açık Küme USPS

Bu bölümde açık küme yaklaşımıyla USPS rakam veri tabanında deneyler gerçekleştirilmiştir. USPS rakam veri tabanı, 9298 adet gri tonlu 16×16 boyutlu resmi içermektedir. Bu resimlerde el yazısıyla yazılmış rakamlar mevcuttur. 7291 adet resim eğitim, 2007 adet resim test için ayrılmıştır. Problemi zorlaştırmak amacıyla bu resimlere herhangi bir ön işlem ya da öznitelik çıkarma uygulanmadan, resimlerin piksel değerleri öznitelik olarak kullanılmıştır. Önceki açık küme deneyinde olduğu gibi 10 adet sınıftan 3 adet rastgele sınıf seçilmiş ve bu 3 sınıfla eğitim yapılmıştır. Test aşamasında ise bu 10 sınıfın hepsi kullanılmış ve AP değerleri hesaplanmıştır. Bu prosedür 10 defa tekrarlanarak elde edilen ait AP değerlerinin ortalaması (mAP) ve test süreleri Çizelge 5.12'de özetlenmiştir. Sonuçlara bakıldığında, OC-EPCC-L_1 yöntemi en iyi sonucu göstermekle birlikte önerilen yöntemler de en üst sıraları doldurmaktadır. Test süresi açısından değerlendirildiğinde ise önerilen yöntemler GEPSVM, SVDD ve SVM kadar hızlı olmasa da diğer yöntemlerden oldukça daha hızlıdır. Önerilen yöntemlerin test süreleri SVM'e de oldukça yakındır. Çizelge 5.12'de incelendiğinde, SVM PCC'den 1,4 kat daha hızlı; EPCC'den de 2,5 kat daha hızlıdır. KSVM, CPM ve Additive Kernel yöntemleri ise en yavaş yöntemlerdir.

Çizelge 5.12 USPS veri tabanı ile açık küme deneyindeki sonuçlar (mAP, %)

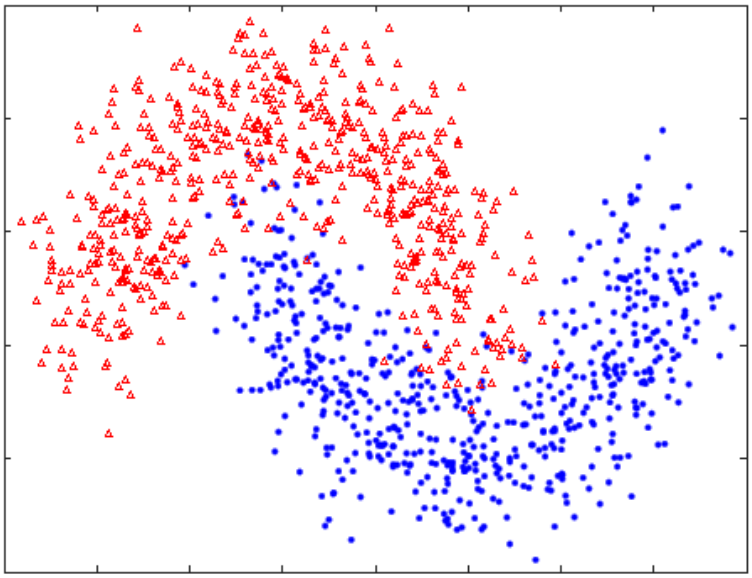
Yöntem	Skor (mAP, %)	Test Zamanı (s)
OC-EPCC-L_1	**81,98±13,74**	0,44
OC-EPCC-L_2	81,97±13,70	0,41
EPCC-L_1	79,63±13,76	0,45
EPCC-L_2	79,63±13,76	0,48
PCC-L_1	78,26±13,56	0,23
PCC-L_2	80,21±11,34	0,24
1-vs-Set Machine	64,28±24,09	0,42
1S-BFHC	68,12±23,15	0,18
SVM	69,7±24,24	0,13
KSVM (polinom)	76,39±21,47	6,61
GEPSVM	45,51±25,93	0,01
SVDD	12,54±8,30	0,01
Additive Kernel	72,08±22,45	1,78
CPM	69,25±14,24	48,1

5.7 MCPCC Yöntemi ile Deneyler

MCPCC sınıflandırıcısı, pozitif örneklerin öznitelik uzayında farklı bölgelerde kümelendiği durumlarda, PCC sınıflandırıcısının daha iyi bir performans göstermesi için geliştirilmiştir. Bu kapsamda bahsedilen veri yapısına sahip olduğu değerlendirilen veri tabanları kullanılarak deneyler gerçekleştirilmiştir. Bu bölümde sentetik veri kümesinde görsel deneyler ile iki sınıflı veri tabanında, MS COCO, ESOGU-285 ve CIFAR-10 veri tabanlarında deneyler gerçekleştirilmiştir.

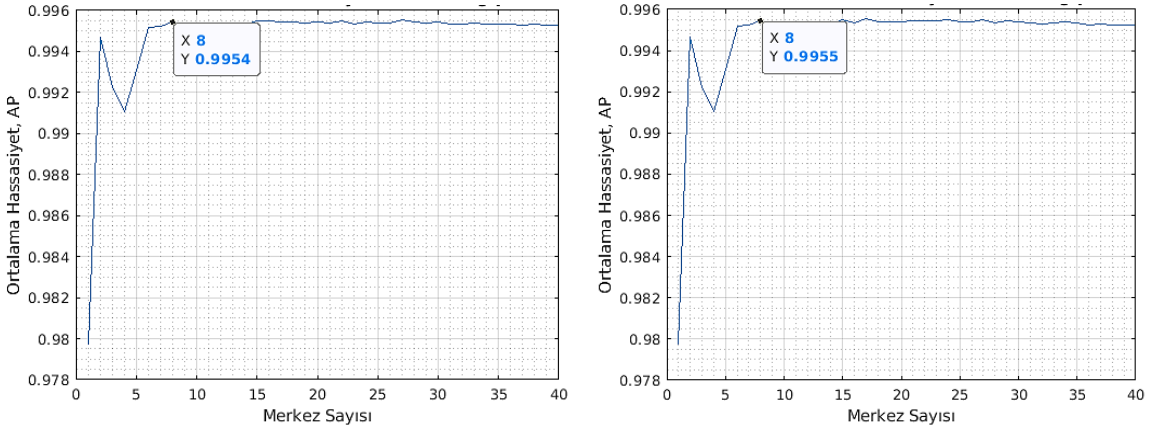
5.7.1 İç içe geçmiş veri kümesi

MCPCC sınıflandırıcısı iki adet sentetik veri kümesinde incelenmiştir. İlk veri kümesi, iç içe geçmiş iki yarım çember şeklinde bir yapıdır (Şekil 5.4). Her biri 600 adet pozitif (mavi) ve 600 adet negatif (kırmızı) örnekten oluşan eğitim ve test verisini içermektedir.



Şekil 5.4 İç içe geçmiş sentetik veri kümesi (Mavi noktalar: pozitif örnekler, Kırmızı üçgenler: negatif örnekler)

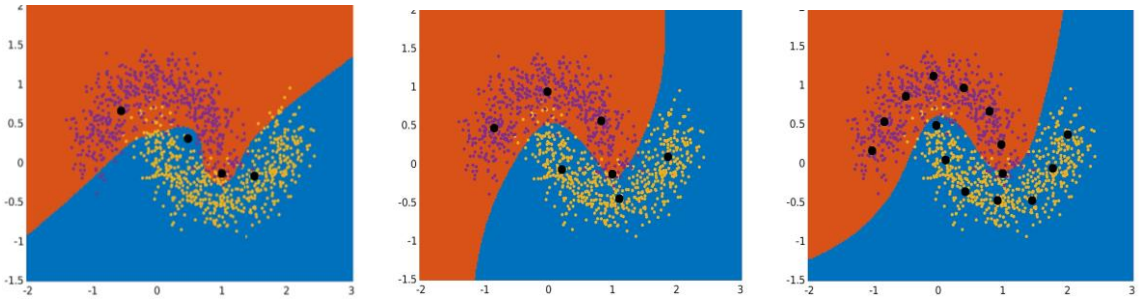
Öncelikle merkez sayısının sınıflandırma performansı üzerindeki etkisini incelemek ve en uygun merkez sayısını belirlemek amacıyla farklı merkez sayılarıyla elde edilen AP değerleri bulunmuştur. Elde edilen sonuçlarla oluşturulan grafik Şekil 5.5'te sunulmuştur. Hem MCPCC-L_1 hem MCPCC-L_2 yönteminde 8 adet merkez ile %99,5'lik AP skoru elde edilmiş ve merkez sayısı arttıkça AP değerinde fazla bir artış gözlenmemiştir. Bundan dolayı merkez sayısının 8 olarak belirlenmesi uygun olacaktır.



(a) (b)

Şekil 5.5 İç içe geçmiş veri kümesinde MCPCC ile farklı merkez sayıları ile AP değişimi, a) MCPCC-L_1 yöntemiyle, b) MCPCC-L_2 yöntemiyle

Merkez sayısının değişiminin pozitif karar bölgesinde nasıl bir etki yaptığı Şekil 5.6'da sunulmuştur. Merkez sayının artmasıyla pozitif karar bölgesinin daha belirleyici olduğu ve daha iyi ayrım yapıldığı görülmektedir.



(a) (b) (c)

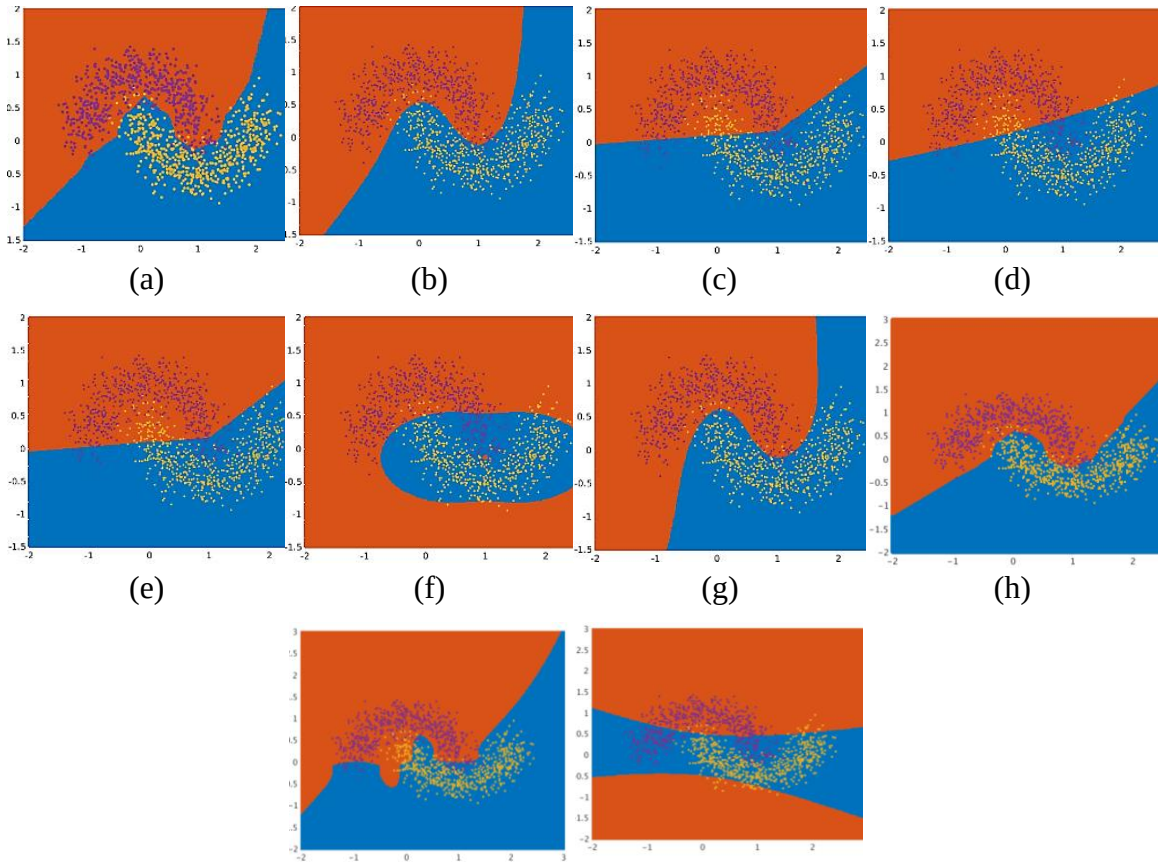
Şekil 5.6 Farklı merkez sayıları ile MCPCC-L_2 karar bölgesi değişimi; merkez sayıları a) 4, b) 7, c) 15; siyah noktalar merkezleri, kırmızı bölge negatif alanı, mavi bölge pozitif alanı temsil etmektedir.

Çizelge 5.13 İç içe geçmiş veri kümesi deneyindeki sonuçlar

Yöntem	**Skor (AP, %)**	**Eğitim Süresi (ms)**	**Test Süresi (ms)**
MCPCC-L_1 (11)	99,55	15,2	1,03
MCPCC-L_2 (11)	**99,56**	16,1	1,04
EPCC-L_1 (4)	95,85	7,3	1,11
EPCC-L_2 (4)	95,17	9,1	1,04
PCC-L_1 (3)	96,17	7,5	1,02
PCC-L_2 (3)	95,85	8,0	1,11
KSVM (Gauss) (92)	99,54	113,2	3,80
PCF+Doğrusal SVM (22)	99.42	-	-
Additive Kernel (5)	96,17	238	1,71
KSVM (Polinom) (668)	82,25	22,0	29

MCPCC yönteminin başarımını karşılaştırmak için PCC, EPCC yöntemleri ile KSVM (Kernel SVM), Additive Kernel ve PCF+Doğrusal SVM yöntemiyle deneyler gerçekleştirilmiştir. Her bir yöntem için farklı parametreler ile deneyler tekrarlanmış ve elde edilen en iyi sonuçlar Çizelge 5.13'te sunulmuştur. Çizelge 5.13 incelendiğinde, MCPCC-L_2 yönteminin en iyi sonucu aldığı görülmektedir. Hemen ardından MCPCC-L_1 ve KSVM (Gauss) en iyi sonuca sahiptir. MCPCC yöntemine benzeyen PCF+Doğrusal SVM ise MCPCC yöntemlerine yakın sonuç göstermiştir. MCPCC sınıflandırıcısı PCC ve EPCC'den daha iyi bir sonuç gösterdiği görülmektedir. Her bir yöntemin kullandığı öznitelik

vektörünün boyutu, Çizelge 5.13'te yöntem isminin yanında parantez içinde gösterilmiştir. KSVM yöntemi için bu değer, kullandığı destek vektör adedi olarak seçilmiştir. Gauss Kernel SVM 92 adet destek vektörü kullanırken, MCPCC-L_1 ve MCPCC-L_2 sadece 11 boyut (2 orijinal boyut + 8 merkez kaynaklı + 1 pozitif örneklerin ortalaması) kullanmıştır.

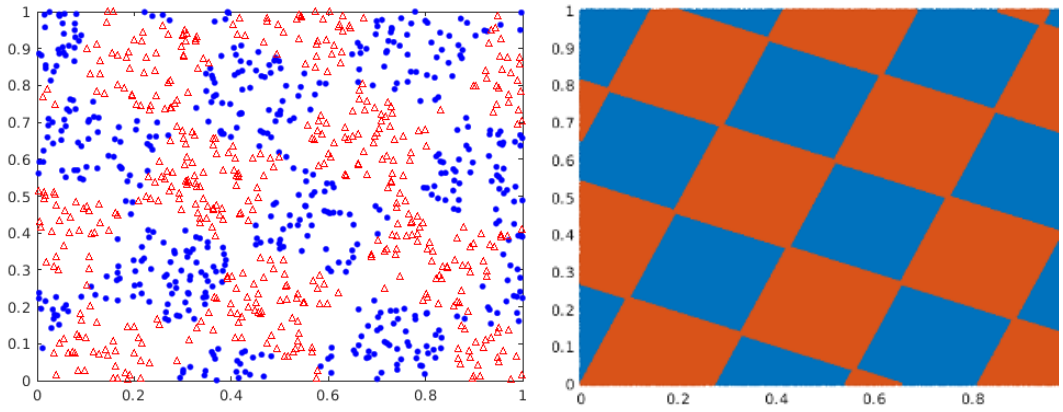


Şekil 5.7 İç içe geçmiş veri kümesi için pozitif karar bölgeleri: a) MCPCC-L_1, b) MCPCC-L_2, c) EPCC-L_1, d) EPCC-L_2, e) PCC-L_1, f) PCC-L_2, g) KSVM (Gauss), h) PCF+Doğrusal SVM, i) Additive Kernel, j) KSVM (Polinom)

MCPCC yöntemlerinin eğitim süresi PCC ve EPCC yöntemlerinden daha uzundur, fakat KSVM'den daha hızlıdır. Test süresi açısından, PCC-L_1 en hızlı yöntem olurken MCPCC yöntemleri ise PCC ve EPCC ile benzer sonuçları vermektedir. Test esnasında MCPCC-L_1 yönteminin KSVM (Gauss)'den 3,5 kat daha hızlı çalıştığı dikkat çekmektedir. Test süresi açısından en yavaş yöntem ise KSVM (Polinom) olmuştur. Deneydeki sonuçlar göz önüne alındığında, MCPCC yöntemi hem hızı hem de basitliği ile ön plana çıkmaktadır. PCF+Doğrusal SVM yönteminin iki aşamalı algoritmasından dolayı sürelerinin karşılaştırmaya uygun olmadığı değerlendirilmiştir. Şekil 5.7'deki karar bölgeleri

incelendiğinde, MCPCC, KSVM (Gauss), PCF+Doğrusal SVM ve Additive Kernel yöntemlerinin pozitif ve negatif verileri başarılı bir şekilde sınıflandırabildiği görülmektedir. Diğer yöntemler bu veri tabanındaki pozitif ve negatifleri ayırmada güçlük çekmektedir.

5.7.2 Döndürülmüş satranç tahtası

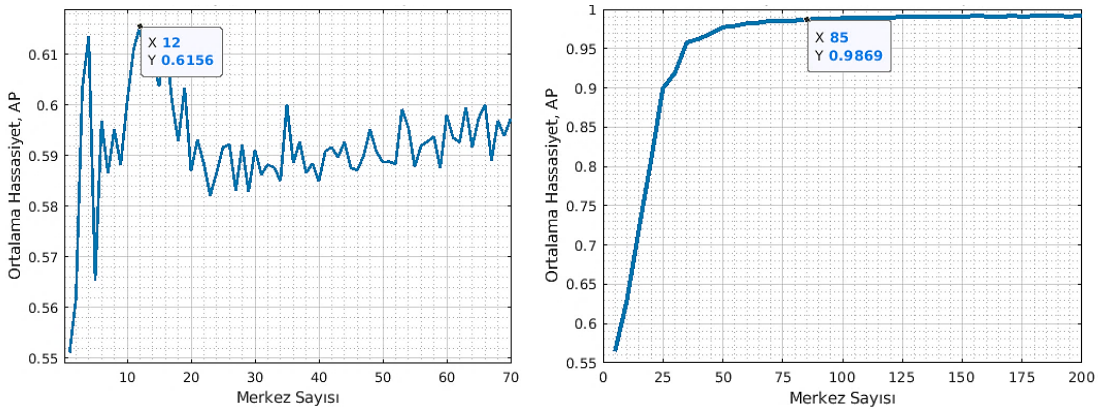


(a) (b)

Şekil 5.8 Döndürülmüş satranç tahtası yerleşimine sahip sentetik veri kümesi: a) gerçek veriler, b) verilerin temsili yerleşimi

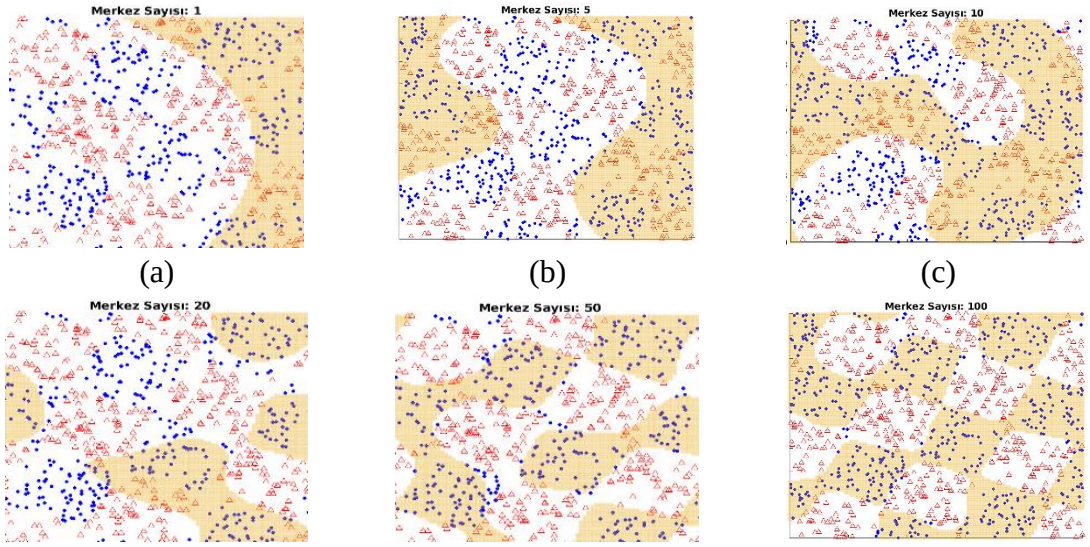
İkinci deneyde π/8 derece döndürülmüş satranç tahtası şeklinde yerleşime sahip sentetik veriler seçilmiştir (Şekil 5.8). Eğitim ve test kümeleri rastgele dağılım gösteren 1000 (pozitif + negatif) örnek içermektedir. En uygun merkez sayısını belirlemek amacıyla farklı merkez sayılarıyla elde edilen sonuçlar hesaplanmıştır (Şekil 5.9). MCPCC-L_2 yöntemiyle 85 adet merkez ile %98,69'luk bir AP değeri elde edilmiştir. MCPCC-L_1 yönteminde 12 merkez ile en iyi sonuç elde edilmiştir.

MCPCC-L_2 kullanılarak merkez sayısının değişimiyle pozitif karar bölgesindeki değişim Şekil 5.10'da gösterilmiştir. Merkez sayısının artmasıyla her bir kare içindeki pozitif verinin daha iyi ayrıştırılabildiği görülmektedir.



(a) (b)

Şekil 5.9 Merkez sayısı değişimi ile ortalama hassasiyet değişimi: a) MCPCC-L_1, b) MCPCC-L_2



(a) (b) (c)

(d) (e) (f)

Şekil 5.10 Farklı merkez sayıları ile MCPCC-L_2 yönteminde pozitif karar bölgeleri (Mavi noktalar pozitif verileri, kırmızı üçgenler negatif verileri, turuncu bölgeler pozitif karar bölgeleri temsil eder.) Merkez sayısı: a) 1, b) 5, c) 10, d) 20, e) 50, f) 100.

Deneyde her bir yöntem için farklı parametrelerle elde edilen sonuçların en iyileri seçilerek Çizelge 5.14'te sunulmuştur. %99,50 AP değeri ile KSVM (Gauss) en yüksek skoru elde etmiştir. Sonra %99,07 ile MCPCC-L_2 yöntemi gelmiştir. Diğer yöntemler ise iyi bir başarım sağlayamamıştır. Bunun en önemli nedeni pozitif örneklerin dağınık yerleşmesi ve karar bölgelerinin oldukça kompleks yapıya sahip olmasıdır. Kümeleme özniteliklerini kullanan PCF+Doğrusal SVM sınıflandırıcısı MCPCC'ye oranla kötü sonuçlar vermekle birlikte PCC, EPCC ve KSVM (Polinom) yöntemlerinden daha iyi sonuç vermiştir.

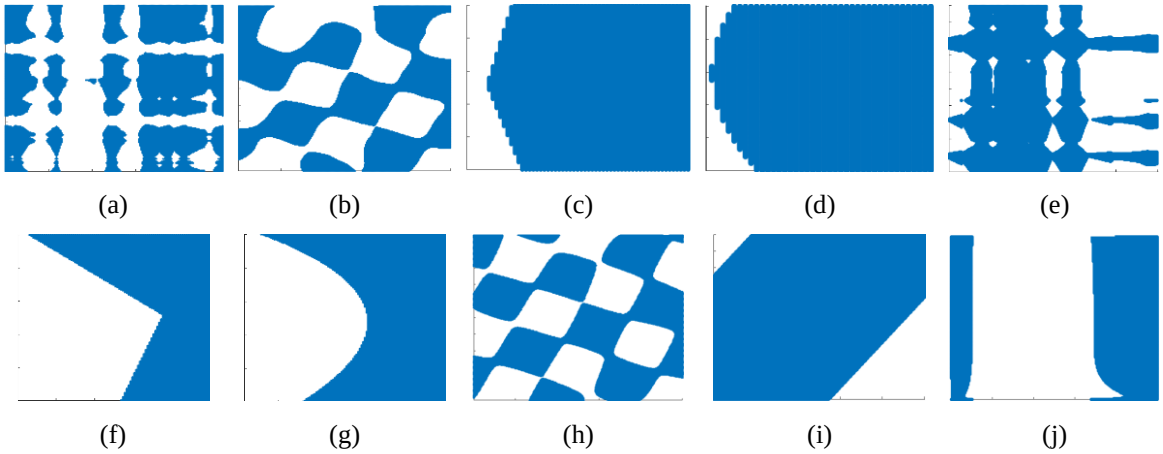
Çizelge 5.14 Satranç tahtası deneyindeki sonuçlar

Yöntem	Ortalama Hassasiyet (AP, %)	Eğitim Süresi (ms)	Test Süresi (ms)
MCPCC-L_1 (15)	62,61	49	1,341
MCPCC-L_2 (88)	99,07	4384	2,326
EPCC-L_1 (4)	54,93	6,71	0,737
EPCC-L_2 (4)	55,33	6,57	0,803
PCC-L_1 (3)	55,08	8,02	**0,717**
PCC-L_2 (3)	56,30	**5,49**	0,822
KSVM (Gauss) (190)	**99,50**	226	2,233
PCF+Doğrusal SVM (6)	57,30	-	-
Additive Kernel (5)	57,41	24,73	1,8
KSVM (Polinom) (3)	52,83	48,38	29,56

Her bir yöntemin kullandığı öznitelik vektörünün boyutu, Çizelge 5.14'te yöntem isminin yanında parantez içinde gösterilmiştir. KSVM (Gauss) yöntemi için bu değer, kullandığı destek vektör adedi olarak seçilmiştir. MCPCC-L_2 yöntemi KSVM'e (Gauss) çok yakın bir başarım sağlamakla birlikte KSVM (Gauss) 190 destek vektörü kullanırken MPCC-L_2 yöntemi 88 boyut kullanmıştır.

Burada dikkat çeken diğer bir husus da MCPCC-L_2 yöntemi için eğitim süresinin diğer yöntemlerden çok daha fazla olmasıdır. Bunun nedeni ise denklem (4.23)'te görüleceği üzere merkez sayısı arttıkça her noktanın merkezlerden uzaklık hesaplamasının da artmasıdır. İlk deneyde MCPCC'de 8 merkez kullanılırken eğitim süresi açısından diğer yöntemlere çok yakındır. Bu deneyde ise merkez sayısının 85 seçilmesi hesaplama yükünü ve eğitim/test sürelerini artırmaktadır.

Şekil 5.11'e sınıflandırıcıların nasıl bir pozitif karar bölgesi meydana getirdiği sunulmuştur. MCPCC-L_2 ve KSVM (Gauss) yöntemlerinde oluşan pozitif karar bölgelerinin satranç tahtasına oldukça benzer olduğu görülmektedir. Diğer yöntemler ise karar bölgelerini doğru bir şekilde tahmin edememektedir.



Şekil 5.11 Döndürülmüş satranç tahtasında pozitif karar bölgeleri, a) MCPCC-L_1, b) MCPCC-L_2, c) EPCC-L_1, d) EPCC-L_2, e) PCF+Doğrusal SVM, f) PCC-L_1, g) PCC-L_2, h) KSVM (Gauss), i) KSVM (Polinom), j) Additive Kernel

5.7.3 İki sınıflı veri tabanları

Bu deneyde MCPCC'ye benzer Öztürk ve Çimen (2019) tarafından önerilen PCF+Doğrusal SVM deneylerinde kullanılan aynı prosedür (10 çapraz katman doğrulama) ve veri tabanları (Liver, Splice ve Svmguide1) (Chang ve Lin, 2011) kullanılmıştır. Bu veri tabanları genellikle küçük ölçekli olup Çizelge 5.15'te özellikleri belirtilmiştir. Öztürk ve Çimen'den (2019) farklı olarak skorlar hesaplanırken sınıflandırma oranı yerine AP ölçeği kullanılmıştır. PCF+Doğrusal SVM yazılımı yazarlarından temin edilmiş olup, yöntemin ikinci aşamasında diğer deneylerde kullanılan SVM yazılımı kullanılmıştır.

Çizelge 5.15 İkili veri tabanlarının özellikleri

Veri tabanı	**Liver**	**Splice**	**Svmguide**
Eğitim örnek sayısı	145	1000	3098
Test örnek sayısı	200	2175	4000
Öznitelik sayısı	5	60	4
Sınıf sayısı	2	2	2

Çizelge 5.16'da elde edilen sonuçlar sunulmuştur. MCPCC-L_1 yöntemi her üç veri tabanında en iyi veya en iyi ikinci sonucu elde etmiştir. MCPCC-L_2 yöntemi de MCPCC-L_1 yöntemine yakın sonuçlar vermiştir. Tek bir koni merkezi içeren EPCC-L_1 Liver veri tabanında, Additive Kernel yöntemi ise Splice veri tabanında en iyi sonucu almıştır.

Çizelge 5.16 İkili veri tabanları ile deney sonuçları (AP, %)

Yöntem	Liver	Splice	Svmguide1
MCPCC-L_1	75,34	95,97	**99,77**
MCPCC-L_2	73,03	92,55	99,73
EPCC-L_1	**77,67**	92,80	99,67
EPCC-L_2	74,13	95,05	99,51
PCC-L_1	69,85	90,87	99,61
PCC-L_2	68,41	90,53	99,59
PCF+Doğrusal SVM	71,69	91,08	99,38
Additive Kernel	75,12	**96,24**	99,68
Polinom Kernel SVM	68,88	89,73	99,61

5.7.4 MS COCO veri tabanı kullanılarak oluşturulan araba sınıfı

Bu deneyde Lin vd. (2014) tarafında sunulan MS COCO veri tabanı kullanılarak, araba sınıfına ait özel bir veri tabanı oluşturulmuştur. MS COCO veri tabanındaki bütün araba sınıfına ait örnekler kullanılarak pozitif sınıf oluşturulmuştur. Sonra rastgele 10 000 resimden araba olmayan örnekler seçilerek negatif örnekler belirlenmiştir. Son olarak tüm veri örnekleri 80×80 piksel boyutuna dönüştürülerek HOG öznitelikleri çıkarılmıştır. Bu işlem sonucunda her bir örneğin öznitelik boyutu 2916 olmuştur. Oluşturulan veri tabanının özellikleri Çizelge 5.17'de belirtilmiştir.

Çizelge 5.17 COCO araba veri tabanı özellikleri

COCO Araba Veri Tabanı	Pozitif Örnek Sayısı	Negatif Örnek Sayısı	Toplam
Eğitim	30 785	69 055	99 840
Test	15 014	68 540	83 554

Deneydeki sonuçlar Çizelge 5.18'de sunulmuştur. KSVM (2. derece polinom kernel) en iyi sonucu göstermiştir. Hemen arkasından MCPCC-L_2 ve MCPCC-L_1 yöntemleri en iyi sonucu göstermiştir. Diğer PCC sınıflandırıcılar ile karşılaştırıldığında MCPCC yöntemleri daha iyi sonuçlar vermiştir. KSVM yönteminde örneklerin öznitelik boyutu $\binom{d+2-1}{2} = 4\,250\,070$ iken MCPCC-L_2 yönteminde öznitelik boyutunun 3517 olması dikkat çekmektedir. Additive Kernel yöntemi 4. sırada olmasına rağmen, örnekler orijinal öznitelik boyutunun 3 katına ($2916 \times (2 \times N + 1) = 8748,\ N = 1$ kernel derecesi)

çıkmaktadır. Bu deneyde satranç tahtası deneyinde olduğu gibi MCPCC-L_2, MCPCC-L_1'den daha iyi bir sonuç göstermiştir.

Çizelge 5.18 COCO araba veri tabanındaki deney sonuçları (AP, %)

Yöntem	Skor (AP, %)
MCPCC-L_1	64,69
MCPCC-L_2	70,16
EPCC-L_1	61,43
EPCC-L_2	61,69
PCC-L_1	60,97
PCC-L_2	61,06
PCF+Doğrusal SVM	61,18
Additive Kernel	64,61
KSVM (Polinom)	**75,16**

5.7.5 ESOGU-285 yüz veri tabanı

Yalcin vd. (2015) tarafından kullanıma sunulan ESOGU-285 yüz veri tabanı, 285 farklı insana ait yüz videosunu içermektedir. Her bir insanın 4 farklı senaryoda 2 farklı zamanda çekilmiş videosu bulunmaktadır. En kısa video 100, en uzunu 1360 resimden oluşmaktadır. Bu deney için seçilen senaryoda, kişilerden Şekil 5.12'deki gibi serbest doğal baş hareketleri yapması istenmiştir. Veri tabanından rastgele 10 adet insan seçilerek resimler kırpılmış ve 120×90 boyutuna sabitlenmiştir. Problemi zorlaştırmak için herhangi bir ön işlem veya öznitelik çıkarma uygulamadan resmin salt piksel değerleri öznitelik olarak kullanılmıştır. Böylece her bir resim için 120×90=10 800 boyutlu öznitelik vektörü elde edilmiştir. Çok sınıflı bir veri tabanı olduğu için bire-tümü yaklaşımıyla eğitim gerçekleştirilmiştir.

Şekil 5.12 ESOGU-285 yüz veri tabanında bulunan bir videodan alınmış resim örnekleri

Yöntemlerin sınıflandırma sonuçları Çizelge 5.19'da sunulmuştur. EPCC-L_1 ve EPCC-L_2 en iyi sonucu göstermiştir. Hemen arkasından Additive Kernel ve MCPCC-L_1 yöntemleri gelmektedir. EPCC orjinal öznitelik boyutunun 2 katı (2×10800), Additive Kernel 3 katı öznitelik boyutuna sahip olmakla birlikte, MCPCC-L_1 orijinal öznitelik boyutu + merkez sayısı (10800 + 401 = 11 201) kadar öznitelik boyutuna sahiptir. Görüleceği üzere MCPCC ile en iyi sonuç veren EPCC sınıflandırıcısı arasında %4 fark bulunmasına karşın MCPCC'nin öznitelik boyutu EPCC'den oldukça azdır.

Çizelge 5.19 ESOGU Yüz veri tabanındaki deney sonuçları (sınıflandırma oranı, %)

Yöntem	Skor (Sınıflandırma Oranı, %)
MCPCC-L_1	68,47
MCPCC-L_2	67,42
EPCC-L_1	**72,44**
EPCC-L_2	**72,44**
PCC-L_1	59,84
PCC-L_2	59,87
PCF+Doğrusal SVM	52,68
Additive Kernel	69,05
KSVM (Polinom)	64,84

5.7.6 CIFAR-10 veri tabanı

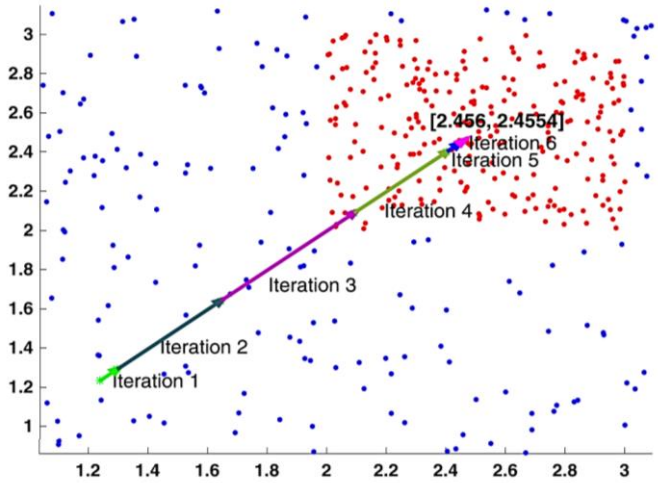
Son olarak CIFAR-10 veri tabanı kullanılarak deney yapılmıştır. Öncelikle AlexNet kullanılarak 4096 boyutlu CNN öznitelikleri çıkarılmıştır. Farklı parametrelerle deneyler tekrar edilmiş ve elde edilen en iyi sonuçlar Çizelge 5.20'de sunulmuştur. MCPCC-L_2 yöntemi en iyi sonucu göstermiştir. Sonra ise KSVM (2. derece polinom kernel) ve EPCC-L_1 yöntemleri gelmektedir. Burada dikkati çeken bir husus da MCPCC-L_2 yöntemi 4897 öznitelik boyutunu kullanırken; Kernel SVM 8 386 560, EPCC-L_1 8192 öznitelik boyutu kullanmaktadır. PCF+Doğrusal SVM yönteminde ise doğrusal programlama kullanan ve öznitelik vektörünün oluşturulduğu ilk aşama, veri tabanının boyutundan dolayı çalıştırılamamıştır.

Çizelge 5.20 CIFAR-10 veri tabanı deneyindeki sonuçlar (Sınıflandırma Oranı, %)

Yöntem	Skor (Sınıflandırma Oranı, %)
MCPCC-L_1	75,25
MCPCC-L_2	**78,81**
EPCC-L_1	75,97
EPCC-L_2	75,22
PCC-L_1	75,66
PCC-L_2	75,69
PCF+Doğrusal SVM	-
Additive Kernel	75,45
KSVM (Polinom)	77,16

5.8 Koni Tepe Noktası Tahmini Deneyi

Bölüm 4.5'te açıklanan koni tepe noktasının, **c**, optimum olarak bulunması için önerilen prosedür, bu bölümde görsel deneylerle ve gerçek veri tabanı deneyleri ile incelenmiştir. İlk deney, **c** noktasının tahminini görsel olarak gösterilmesi amacıyla Şekil 5.13'teki sentetik veri kümesinde yapılmıştır. Bu veri kümesinde 250 pozitif örnek $(2,3)$ koordinatları arasında kare şeklinde ve rastgele dağılmıştır. Pozitif verilerin merkezi $\begin{pmatrix}2,5\\2,5\end{pmatrix}$ noktasıdır. 750 negatif örnekse pozitifleri çevreleyecek şekilde rastgele yerleştirilmiştir. **c** tepe noktasının başlangıç değeri $\begin{pmatrix}0\\0\end{pmatrix}$ alınmış ve 6 adım sonra $\begin{pmatrix}2,456\\2,455\end{pmatrix}$ noktasına yaklaştığı görülmüştür.



Şekil 5.13 Sentetik veri kümesinde optimizasyon adımlarıyla tepe noktasının değişimi (Cevikalp ve Saglamlar'dan, 2019)

UCI veri havuzundan ikili ve çok sınıflı veri tabanlarından 7 adeti (Iris, Letter Recognition-LR, Multiple Features-MF, Wine, Glass, Wisconsin Diagnostic Breast Cancer-WDBC ve Pima Indian Diabetes-PIMA) kullanılarak deneyler yapılmıştır. EPCC-L_1-C ve EPCC-L_2-C yöntemlerinde koni tepe noktası optimize edilerek bulunmuştur. EPCC-L_1 ve EPCC-L_2 yöntemlerinde ise koni tepe noktası pozitif örneklerin ortalaması olarak seçilmiştir. Elde edilen sonuçlar karşılaştırma yapılması için Çizelge 5.21'de sunulmuştur. Sonuçlardan görüldüğü üzere optimizasyonla koni tepe noktası bulan deneylerin daha başarılı olduğu görülmektedir. Bunula birlikte tepe noktası pozitif örneklerin ortalaması olan deneylerdeki sonuçlar bunlara oldukça yakındır. En fazla fark ise PIMA'da ve %0,77'dir. Ayrıca MF, Wine, Glass veri tabanlarında ise aynı sonuçlar elde edilmiştir. Tüm bunlar değerlendirildiğinde, tepe noktasının optimizasyonla bulunması sonuçlarda çok az bir ilerleme sağlamakla birlikte işlem yükünü oldukça arttırmaktadır.

Çizelge 5.21 UCI veri havuzunda koni tepe noktası optimizasyonla bulunarak elde edilen sonuçlarla koni tepe noktası pozitif örneklerin ortalaması olarak alındığında elde edilen sonuçların karşılaştırılması (Sınıflandırma Oranı, %)

Yöntem	**Iris**	**LR**	**MF**	**Wine**	**Glass**	**WDBC**	**PIMA**
EPCC-L_1-C	**96,67**	79,60	95,30	**96,25**	**97,62**	**97,91**	76,41
EPCC-L_2-C	96,00	**79,69**	**95,60**	95,63	94,36	97,73	**77,30**
EPCC-L_1	95,99	75,41	95,40	95,63	**97,62**	97,86	76,53
EPCC-L_2	95,99	79,34	**95,60**	**96,25**	96,19	97,73	76,89

6. SONUÇ VE ÖNERİLER

Sınıflandırma problemlerinde geniş marj tabanlı sınıflandırıcılar iki sınıfı birbirinden ayıran en uygun karar sınırını bulmayı amaçlar. SVM'de geniş marj sınıflandırıcılardan olup en geniş marjı sağlayan hiperdüzlemi bulmayı amaçlar. Bu hiperdüzlem öznitelik uzayını ikiye bölerek pozitif ve negatif verileri birbirinden ayırır. Günlük hayatta karşılaşılan sınıflandırma örnekleri her zaman iki sınıfı bu şekilde ayıracak kadar simetrik değildir. Bazı problemlerde pozitif örnekler çok daha az olmakla birlikte görsel olarak kompakt ve belirli bir dağılım göstermektedir. Negatif örneklerse çok daha dağınık ve farklı yoğunlukta yerleşmektedir. Örneğin görsel nesne sınıflandırma problemlerinde belirli bir sınıfa ait örnekler ön plana çıkmakta diğer her şey arka plan olarak tanımlanmaktadır. Benzer olarak, yüz doğrulama problemlerinde bir kişiye ait yüz örnekleri pozitif, farklı kişiler negatif olarak tanımlanmaktadır. Bu tarz problemlerin çözümlerinde pozitif örneklerin sıkı ve kompakt karar sınırlarıyla çevrelenmesi önemlidir. Bu sayede pozitifler negatif örneklerden çok daha iyi ayrılabilmektedir. Tek sınıf sınıflandırıcılar olarak da adlandırılan bu yöntemler negatif eğitim verilerinin olmadığı durumlarda bile etkili bir karar sınırı bulabilmektedir. Bu tezde de kompakt karar sınırları döndüren PCC, EPCC çokyüzlü konik sınıflandırıcıları önerilmiştir. Bu sınıflandırıcılar güçlü ve ölçeklenebilir geniş marj tabanlı yöntemler olup, pozitif karar bölgeleri koninin içinde kalan hiperdüzlem kesit alanından elde edilmektedir. Uygun parametrelerle pozitif karar bölgeleri kompakt, sıkı, dışbükey bir yapıya sahip olmaktadır. Bu yöntemler, yapılan öznitelik artırma işlemiyle SVM biçimine çevrilebilmektedir. Bu sayede SVM'in kullandığı geniş matematiksel çözümler ve yazılımlar önerilen yöntemler içinde uygulanabilmektedir.

Bazı şartlar altında PCC ve EPCC sınıflandırıcısından kapalı karar bölgeleri elde edilememektedir. Her ne kadar SVM'in yarı uzaylı karar sınırından çok daha küçük olsa da arzu edilen bir durum değildir. EPCC'nin farklı bir çeşidi olan OC-EPCC yöntemi, kapalı karar sınırlarını zorlayan bir sınıflandırıcıdır. OC-EPCC'de Stokastik Gradyan yöntemiyle parametreler bulunmakta ve kapalı, sıkı karar sınırları oluşturulmaktadır.

Diğer karşılaşılan bir problem ise pozitif örneklerin aynı uzayda farklı bölgelerde kümelenmesidir. Örneğin bir arabaya ait yandan, üstten, arkadan ve önden görünümler

öznitelik uzayında farklı bölgelerde kümelenebilmektedir. Bu durumda her bir küme için ayrı bir sınıflandırıcı kullanılması gerekir. Birden fazla sınıflandırıcı, eğitim ve test sürecini hem yavaşlatabilmekte hem de karmaşık hale getirebilmektedir. Bu kapsamdaki problemleri çözmek amacıyla hem PCF basitliğini koruyan hem de tek bir sınıflandırıcı kullanan MCPCC geliştirilmiştir. MCPCC ile tek bir sınıflandırıcı kullanılarak farklı merkezleri içeren PCF fonksiyonu oluşturulmuştur. Bu merkezler *k*-merkezli kümeleme yöntemi kullanılarak bulunmuştur. Eğitimden önce küçültülmüş veri tabanı ile merkez sayıları kademeli olarak artırılmış ve en uygun merkez sayısı seçilmiştir. Ayrıca merkez sayısı performans/hız dengesini sağlayabilen bir denge parametresi olarak da kullanılabilmektedir. Bir merkezli MCPCC ise PCC sınıflandırıcısını vermektedir. MCPCC, PCC'nin genel hali olarak kabul edilebilir. MCPCC ile farklı bölgelerde öbeklenen pozitiflerin etkili bir şekilde sınıflandırılmasının sağlandığı görülmüştür.

Çalışmada ayrıca PCF fonksiyonunda kullanılan tepe noktasını ve parametreleri optimizasyonla bulan bir algoritmaya da yer verilmiştir. Bu algoritmada birbiri ardınca çözülen iki optimizasyon problemi mevcuttur. En iyi tepe noktası ve parametreler bulunana kadar bu problemler ardışık olarak çözülür. Deneylerle elde edilen sonuçlara göre optimizasyonla bulunan tepe noktası az bir oranda sonuçları geliştirmiştir. Bunula birlikte getirdiği hesaplama yükü oldukça fazladır. Bu açıdan eğitim sırasında kullanılması eğitim sürecini yavaşlatmaktadır. Hız gerektiren uygulamalarda pozitif örneklerin ortalamasının koni tepe noktası olarak tercih edilmesi uygun bir tercih olacaktır.

Yapılan deneylerde nesne bulma, açık küme tanıma, yüz doğrulama ve klasik kapalı küme problemlerinde çok iyi sonuçlar elde edilmiştir. Özellikle görsel nesne bulma ve açık küme sınıflandırma problemlerinde, SVM ve bazı tek sınıf sınıflandırıcılar karşısında aldığı sonuçlar dikkat çekicidir. Bu kapsamda görsel nesne bulma ve sınıflandırma problemlerinde SVM gibi doğrusal sınıflandırıcıların yerine kullanılabilecek bir sınıflandırıcı olduğu değerlendirilmektedir. MCPCC sınıflandırıcısı ise pozitif örneklerin aynı uzayda farklı bölgelerde kümelendiği veri tabanlarında başarılı sonuçlar almıştır. Aldığı sonuçlar ya en iyi ya da en iyi ikinci sonuçlardır. Bunun yanında sağladığı basit yapısı ve hızı MCPCC'yi ön plana çıkarmaktadır.

Son yıllarda derin öğrenme konusu elde edilen başarılı sonuçlardan dolayı popüler olmaya başlamıştır. R-CNN gibi son katmanında sınıflandırma amacıyla SVM kullanan yöntemlerde SVM yerine yöntemlerimizin kullanılmasıyla daha iyi sonuçlar elde edildiği görülmüştür (Cevikalp ve Saglamlar, 2019). Her ne kadar bu tezde derin öğrenme ile ilgili bir çalışmaya yer verilmese de Cevikalp ve Saglamlar (2019) tarafından derin öğrenmede de kullanılabileceği gösterilmiştir. Bunun yanında son katmanda sınıflandırma amacıyla kullanılmak yerine derin öğrenme yapısının içine doğrudan entegre edilerek daha hızlı ve iyi sonuçların elde edileceği değerlendirilmektedir.

KAYNAKLAR DİZİNİ

Alpaydın, 2014, E. Introduction to Machine Learning, Third Edition, The MIT Press, p.13.

Anonim, 2020, Kernel Trick in SVM, https://medium.com/analytics-vidhya/how-to-classify-non-linear-data-to-linear-data-bb2df1a6b781, erişim tarihi: 26.010.2020.

Anonim, 2020, Reinforcement learning https://en.wikipedia.org/wiki/Reinforcement _learning, erişim tarihi: 30.09.2020.

Bagirov, A., Ugon, J., Webb D., 2011, An efficient algorithm for the incremental construction of piece-wise linear classifier, Information systems, 36, p.782-790.

Bagirov, A. M., Ugon, J., Webb, D., Ozturk, G., Kasimbeyli, R., 2013, A novel piecewise linear classifier based on polyhedral conic and max–min separabilities, TOP: An Official Journal of the Spanish Society of Statistics and Operations Research, 21,1, p.3-24.

Beşer F., 2020, Türkçe Yapay Zekâ Terimleri, https://deeplearningturkiye. gitbooks.io/turkce-yapay-zeka-terimleri/content/ingilizce-turkce.html, erişim tarihi: 20.03.2020.

Beveridge, R., Phillips, J., Bolme, D., Draper, B., Givens, G., vd., 2013, The challenge of face recognition from digital point-and-shoot cameras, Conference on biometric theory, applications and systems.

Blum, A., 2020, Machine Learning Theory, http://www.cs.cmu.edu/afs/cs/user/avrim/ www/Talks/mlt.pdf, erişim tarihi: 01.10.2020.

Bolla, M., 2013, Spectral clustering and biclustering: learning large graphs and contingency tables, Wiley, p.199.

Cevikalp, H., Triggs, B., Jurie, F., Polikar, R., 2008, Margin-based discriminant dimensionality reduction for visual recognition. IEEE conference on computer vision and pattern recognition, p.1-8.

Cevikalp, H., Triggs, B., 2009, Large margin classifiers based on convex class models, IEEE 12th international conference on computer vision workshops, p.101-108.

Cevikalp, H., Triggs, B., 2012, Efficient Object Detection Using Cascades of Nearest Convex Model Classifiers, IEEE Computer society conference on computer vision and pattern recognition, p.3138-3145.

Cevikalp, H., Triggs, B., 2013, Hyperdisk based large margin classifier, Pattern recognition, 46, 6 p.1523-1531.

KAYNAKLAR DİZİNİ (devam)

Cevikalp, H., Triggs, B., Franc, V., 2013, Face and landmark detection by using cascade of classifiers, 10th IEEE International Conference on Automatic Face and Gesture Recognition, p.1-7.

Cevikalp, H., 2016, Best fitting hyperplanes for classification, IEEE Transactions on Pattern Analysis and Machine Intelligence. 39, 6, p.1076-1088.

Cevikalp, H., Triggs, B., 2017, Polyhedral conic classifiers for visual object detection and classification, 2017 IEEE Conference on Computer Vision and Pattern Recognition, 123, 3, p. 4114-4122.

Cevikalp, H., Triggs, B., 2017, Visual object detection using cascades of binary and one-class classifiers. International journal of computer vision, 12, p.334-349.

Cevikalp, H., Saglamlar, H., 2019, Polyhedral conic classifiers for computer vision applications and open set recognition, IEEE transactions on pattern analysis and machine intelligence, doi: 10.1109/tpami.2019.2934455 (in press).

Chang, C., Lin, C., 2011, Libsvm: A library for support vector machines, ACM Trans Intell Syst Technol, 2, p.1-39.

Cimen, E., Ozturk, G., Gerek, O., 2017, Incremental conic functions algorithm for large scale classification problems, Digital Signal Processing, 77, p.187-194.

Cimen, E., Ozturk, G., 2019, O-PCF algorithm for one-class classification, Optimization Methods and Software, doi:10.1080/10556788.2019.1581191.

Cortes, C., Vapnik, V., 1995, Support vector networks, Machine learning, 20, p.273-297.

Dalal, N., Triggs, B., 2005, Histograms of oriented gradients for human detection, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1, p.886-893.

Dordinejad, G. G., Çevikalp, H., 2017, Cone vertex estimation in polyhedral conic classifiers, 2017 25th Signal Processing and Communications Applications Conference, p.1-4.

Dundar, M., Wolf, M., Lakare, S., Salganicoff, M., Raykar, VC, 2008, Polyhedral classifier for target detection: A case study: Colorectal cancer, International Conference on Machine Learning, p.288-295.

Ertekin, S., Bottou, L., Giles, C. L., 2011, Nonconvex online support vector machines IEEE transactions on pattern analysis and machine intelligence, 2033, p.368-381.

Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A., 2010, The PASCAL visual object classes challenge, International journal of computer vision, 88, 2, p.303-338.

KAYNAKLAR DİZİNİ (devam)

Felzenszwalb, P., Girshick, R. B., McAllester, D., Ramanan, D.,2010, Object detection with discriminatively trained part based models, IEEE transactions on pattern analysis and machine intelligence, 32,9, p.1627-1645.

Franc, V., Sonnenburg, S., 2008, Optimized cutting plane algorithm for support vector machines, International conference on machine learning, p.320-327.

Gasimov, R., Ozturk, G., 2006, Separation via polyhedral conic functions, Optimization methods & software, 21, 4, p.527-540.

Girshick, R., 2015, Fast R-CNN, Proceedings of the 2015 IEEE international conference on computer vision p.1440-1448.

Hosang, J., Benenson, R., Doll´ar, P., Schiele, B., 2015, What makes for effective detection proposals? IEEE transactions on pattern analysis and machine intelligence, 38, p.814-830.

Huang, Z., Wang, R., Shan, S., Chen, X., 2015, Projection metric learning on grassmann manifold with application to video based face recognition, Computer vision and pattern recognition, p.140-149.

Hui, J., 2020, Machine learning - Clustering, Density based clustering and SOM, https://jhui.github.io/2017/01/15/Machine-learning-clustering/, erişim tarihi: 12.10.2020.

Hussain, S., Triggs, B., 2010, Feature sets and dimensionality reduction for visual object detection, Proceedings of the British Machine Vision Conference, p. 112.1-112.10.

Jain, V., Learned-Miller, E., 2010, Fddb: A benchmark for face detection in unconstrained settings, Technical Report UM-CS-2010-009, University of Massachusetts, 11 p. (unpublished).

Jayadeva, Khemchandani, R., Chandra, S. 2007, Twin support vector machines for pattern classification, IEEE transactions on pattern analysis and machine intelligence, 29, p.905-910.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., vd., 2014, Caffe: Convolutional architecture for fast feature embedding, Computing research repository, http://arxiv.org/abs/1408.5093, erişim tarihi: 04.06.2020.

Kalal, Z., Matas, J., Mikolajczyk, K., 2008, Weighted sampling for large-scale boosting, Proceedings of the British Machine Vision Conference 2008, p.42.1-42.10.

Kantchelian, A., Tschantz, M.C., Huang, L., Barlett, P.L., Joseph, A.D., vd., 2014, Large margin convex polytope machine, advances in neural information processing systems, 2, p.3248-3256.

KAYNAKLAR DİZİNİ (devam)

Krizhevsky, A, Sutskever, I., Hinton, G. E., 2012, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems, p. 1097-1105.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R.B., vd., 2014, Microsoft COCO: common objects in context, Computing research repository, http://arxiv.org/abs/1405.0312, erişim tarihi: 04.06.2020.

MacQueen, J., 1967 Some methods for classification and analysis of multivariate observations, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1, p.281-297.

Mangasarian O. L., Wild, E. W., 2006, Multisurface proximal support vector machine classification via generalized eigenvalues, IEEE Transactions on pattern analysis and machine intelliegence, 28, p.69-74.

Manwani, N., Sastry, P.S., 2010, Learning polyhedral classifiers using logistic function. Asian Conference on Machine Learning, 13, p.17-30.

Manwani, N., Sastry, P. S., 2011, Polyceptron: A polyhedral learning algorithm, Computing research repository, http://arxiv.org/abs/1107.1564, erişim tarihi: 04.06.2020.

Ozturk, G., Bagirov, A. M., Kasimbeyli, R., 2015, An incremental piecewise linear classifier based on polyhedral conic separation, Machine Learning, 101, p.397-413.

Ozturk, G., Ciftci, T.,2015, Clustering based polyhedral conic functions algorithm in classification, Journal of Industrial & Management Optimization, 11, 3, p.921-932.

Ozturk, G., Cimen, E., 2019, Polyhedral conic kernel-like functions for SVMs, Turkish journal of electrical engineering & computer sciences, 27, p.1172-1180.

Rahimi, A., Recht, B., 2007, Random features for large-scale kernel machines, advances in neural information processing systems, p. 1177-1184.

Rao, Y., Lin, J., Lu, J., Zhou, J., 2017, Learning discriminative aggregations network for video-based face recognition, 2017 IEEE international conference on computer vision, p.3801-3810.

Redmon, J., Divvala, S. K., Girshick, R. B., Farhadi, A., 2015, You Only Look Once: Unified, Real-Time Object Detection, abs/1506.02640.

Ren, S., He, K., Girschick, R., Sun, J., 2017, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Transactions on image processing, 39, p.1137–1149.

KAYNAKLAR DİZİNİ (devam)

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., vd., 2015, ImageNet large scale visual recognition challenge, International journal of computer vision, 115, 3, p.211-252.

Sanchez, J., Perronnin, F., Mensink, T., Verbeek, J., 2013, Image classification with the fisher vector: Theory and practice. International journal of computer vision, 34, p.1704–1716.

Scheirer, W. J., Rocha, A., Sapkota, A., Boult T. E., 2013, Towards open set recognition, IEEE transactions on pattern analysis and machine intelligence, 35, p.1757-1772.

Shalev-Shwartz, S., Singer, Y., Srebro, N., 2011, Pegasos: Primal estimated sub-gradient solver for svm, Mathematical Programming, 127, 1, p.3-30.

Tax D. M. J., Duin, R. P. W., 2004, Support vector data description, Machine Learning, 54, p.45-66.

Türkiye Bilimler Akademisi (TÜBA), 2013, Türkçe Bilim Terimleri Sözlüğü, http://www.tubaterim.gov.tr/, erişim tarihi: 06.10.2020.

Uylaş Satı, N., 2016, A Binary Classification Approach Based on Support Vector Machines Via Polyhedral Conic Functions, Celal Bayar Üniversitesi Fen Bilimleri Dergisi, 12, 2, p.135-149.

Vedaldi, A., Zisserman, A., 2012, Efficient additive kernels via explicit feature maps, ıeee transactions on pattern analysis and machine intelligence, 34, p.480-492.

Viola, P., Jones, M.J., 2004, Robust real-time face detection, International journal of computer vision, 57, 2, p.137-154.

Yalcin, M., Cevikalp, H., Yavuz, H.S.,2015, Towards large-scale face recognition based on videos, International Conference on Computer Vision Workshops 2015, p.1078-1085.

ÖZGEÇMİŞ