

**YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**YENİ MAKİNE ÖĞRENMESİ METOTLARI VE
İLAÇ TASARIMINA UYGULAMALARI**

Bilgisayar Yük. Müh. M.Fatih AMASYALI

**FBE Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Programında
Hazırlanan**

DOKTORA TEZİ

Tez Savunma Tarihi : 14 Ocak 2008
Tez Danışmanı : Prof. Dr. Oya KALIPSIZ (Yıldız Teknik Üniversitesi)
İkinci Tez Danışmanı : Prof. Dr. Okan ERSOY (Purdue Üniversitesi)
Jüri Üyeleri : Prof. Dr. Coşkun SÖNMEZ (Yıldız Teknik Üniversitesi)
: Doç. Dr. Berrin YANIKOĞLU (Sabancı Üniversitesi)
: Doç. Dr. Zehra ÇATALTEPE (İstanbul Teknik Üniversitesi)
: Yrd. Doç. Dr. Baran DİRİ (Yıldız Teknik Üniversitesi)

İSTANBUL, 2008

İÇİNDEKİLER

Sayfa

KISALTMA LİSTESİ.....	viii
ŞEKİL LİSTESİ.....	ix
ÇİZELGE LİSTESİ	xi
ÖNSÖZ	xiv
ÖZET	xv
ABSTRACT	xvii
1. GİRİŞ	1
1.1 Makine Öğrenmesi Nedir?	1
1.2 Tezdeki Bölümlerin Tanıtımı	3
2. İLAÇ TASARIMI.....	5
2.1 İlaç Nedir, Nasıl Çalışır?.....	5
2.2 Yeni İlaç Geliştirmenin Önemi	5
2.3 İlaçlardan Beklenen Özellikler	6
2.4 İlaç Tasarımının Evreleri ve Maliyeti.....	6
2.5 Dünyada İlaç Sanayi	9
2.6 İlaç Tasarımında Makine Öğrenmesi Metotlarının Kullanımı	9
2.7 İlaç Verilerini Elde Etmede Kullanılan Metotlar	10
2.8 İlaç Verilerinin Özellikleri	11
2.9 İlaç Veri Kümelerinin Formatları.....	13
2.9.1 SDF Formatı (Structure Data Format)	13
2.9.2 SMILES Formatı	14
2.9.3 First Order Logic Formatı	14
2.9.4 Sayısal Formatlar	15
2.10 Moleküllerin Sayısal Verilere Dönüştürülmesi.....	15
2.11 İlaç Moleküllerini İfade Etmede Kullanılan Özellikler	16
2.12 İlaç Tasarımında Makine Öğrenmesi Kullanılan Önceki Çalışmalar.....	17
2.13 Sonuç	21
3. SINIFLANDIRMA	22
3.1 Terimler ve Kullanılan Semboller	22
3.2 Önceki Çalışmalar	24
3.2.1 Karar Ağaçları	24
3.2.2 Karar Ağaçlarında Budama	25
3.2.3 Tek ve Çok Değişkenli Karar Ağaçları.....	26
3.2.4 Karar Ağaçlarının Oluşturulmasında Kullanılan Yöntemler	27
3.3 Gerçekleştirilenler	29
3.3.1 Cline Algoritmasının Versiyonları	30

3.3.2	Cline Algoritmasıyla Verileri Sınıflandırma.....	34
3.3.3	Cline Algoritmalarının Görsel Karşılaştırması	35
3.3.4	Cline Hiper Düzlem Parametrelerinin Bulunmasında Kullanılan Analitik Denklemler.....	37
3.3.5	Cline Algoritmasını N Adet Sınıf İçin Genelleştirmek	38
3.4	Deneysel Sonuçlar	39
3.4.1	UCI Veri Kümelerinin Sınıflandırılması	39
3.4.2	İlaç Veri Kümelerinin Sınıflandırılması	43
3.5	Çıkarımlar	45
3.6	Gelecek Çalışmalar.....	46
4.	SINIFLANDIRMA KOMİTELERİ	49
4.1	Tanımlar ve Kullanılan Semboller	49
4.2	Komitelerin Performansları Neden Yüksektir?	52
4.3	Önceki Çalışmalar	53
4.4	Gerçekleştirilenler	54
4.5	Deneysel Sonuçlar	55
4.5.1	UCI Veri Kümeleri Üzerindeki Sonuçlar.....	55
4.5.2	İlaç Verilerindeki Sonuçlar	62
4.6	Çıkarımlar	64
4.7	Gelecek Çalışmalar.....	66
5.	KÜMELEME	67
5.1	Tanımlar ve Kullanılan Semboller	67
5.1.1	Kümeler Arasındaki Benzerliğin Ölçümü.....	67
5.1.2	Kümeleme Kalitesinin Ölçümü	68
5.2	Önceki Çalışmalar	70
5.3	Gerçekleştirilenler	74
5.3.1	Clusline Algoritmasının Versiyonları.....	75
5.4	Deneysel Sonuçlar	78
5.5	Çıkarımlar	80
5.6	Gelecek Çalışmalar.....	81
6.	KÜMELEME KOMİTELERİ	82
6.1	Önceki Çalışmalar	83
6.1.1	Graf Bölümleme Metotları.....	83
6.1.2	Hiyerarşik (agglomerative) Metotlar	84
6.2	Gerçekleştirilenler	84
6.3	Deneysel Sonuçlar	85
6.3.1	Elde Edilen Sonuçlar	86
6.3.2	Kümeleyici Sayısının Etkisi	89
6.3.3	Özellik Sayısı Etkisi	90
6.4	Çıkarımlar	91
6.5	Gelecek Çalışmalar.....	92
7.	ÖZELLİK SEÇİMİ	93
7.1	Tanımlar	93
7.2	Çok Boyutlu Verilerle Çalışmak	93
7.3	Önceki Çalışmalar	95

7.4	Gerçekleştirilenler	95
7.4.1	Özellik Seçim Ormanları	95
7.5	DeneySEL Sonuçlar	97
7.6	Çıkarımlar	99
8.	REGRESYON	100
8.1	Tanımlar ve Kullanılan Semboller	100
8.1.1	Entropi.....	101
8.1.2	Ekstrem ve Aykırı Örnekler	101
8.1.3	Colinerity	101
8.1.4	Regresyonda Hata Ölçüm Metotları	103
8.2	Önceki Çalışmalar	104
8.2.1	Çoklu Lineer Regresyon (Multiple Linear Regression - MLR).....	104
8.2.2	Temel Bileşen Tabanlı Regresyon (Principal Component Regression -PCR)....	105
8.2.3	Parçalı En Küçük Kareler (Partial Least Squares - PLS).....	106
8.2.4	Regresyon Ağaçları	107
8.2.4.1	M5 Model Ağaçları.....	108
8.2.4.2	M5' Model Ağaçları	110
8.2.4.3	SMOTI	110
8.2.5	Örnek Tabanlı Algoritmalar	110
8.3	Gerçekleştirilenler	111
8.3.1	Kmeans ile Regresyon	111
8.3.2	KmeansS ile Regresyon	113
8.3.3	KmeansMOD ile Regresyon	118
8.3.4	KmeansXY ile Regresyon.....	119
8.4	DeneySEL Sonuçlar	120
8.5	Çıkarımlar	122
8.6	Gelecek Çalışmalar	122
9.	META REGRESYON	123
9.1	Tanımlar ve Semboller.....	123
9.2	Farklı Sayıdaki Özellik Sayısına Sahip Veri Kümelerinin (Örneklerin) Bir Arada Kullanılması	124
9.2.1	Histogram Yaklaşımı	125
9.2.2	Kümeleme Yaklaşımı	126
9.3	Önceki Çalışmalar	126
9.4	Gerçekleştirilenler ve DeneySEL Sonuçlar	128
9.4.1	Meta Regresyon Denemelerinde Ortak Kullanılan Meta Özellikler	128
9.4.2	Yapay Veri Kümeleri Koleksiyonunda Meta Regresyon	138
9.4.2.1	Friedman Yapay Veri Kümelerinin Üretilmesi	138
9.4.2.2	Friedman Veri Koleksiyonunda Kullanılan Ek Meta Özellikler.....	139
9.4.2.3	Friedman Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri.....	139
9.4.2.4	Friedman Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri.....	142
9.4.2.5	Friedman Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümelenecekleri	147
9.4.2.6	Friedman Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri	149
9.4.2.7	Friedman Koleksiyonunda Performans Tahminleriyle Kullanılan Meta Özelliklerin	

	Analizleri.....	157
9.4.3	İlaç Veri Kümeleri Koleksiyonunda Meta Regresyon.....	158
9.4.3.1	İlaç Veri Kümeleri	158
9.4.3.2	İlaç Veri Koleksiyonunda Kullanılan Ek Meta Özellikler.....	161
9.4.3.3	İlaç Veri Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri.....	162
9.4.3.4	İlaç Veri Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri.....	163
9.4.3.5	İlaç Veri Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümelenmeleri	167
9.4.3.6	İlaç Veri Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri.....	169
9.4.3.7	İlaç Koleksiyonunda Performans Tahminlerinde Kullanılan Meta Özelliklerin Analizleri.....	175
9.4.4	UCI Veri Kümeleri Koleksiyonunda Meta Regresyon.....	175
9.4.4.1	UCI Veri Kümeleri	176
9.4.4.2	UCI Koleksiyonunda Kullanılan Ek Meta Özellikler.....	177
9.4.4.3	UCI Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri.....	178
9.4.4.4	UCI Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri.....	179
9.4.4.5	UCI Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümelenmeleri	182
9.4.4.6	UCI Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri.....	184
9.4.4.7	UCI Koleksiyonunda Performans Tahminlerinde Kullanılan Meta Özelliklerin Analizleri.....	189
9.4.5	PLS'in Bileşen Sayısının Analizi	190
9.4.5.1	PLS'in Bileşen Sayısının Analizinde Kullanılan Veri Kümeleri	190
9.4.5.2	Meta Özelliklerle PLS'in Optimum Bileşen Sayısının Tahmini.....	191
9.4.5.3	PLS'in Bileşen Sayısının Performansa Etkisi	192
9.4.6	Meta Özelliklerle En Başarılı Algoritmanın Tahmini	192
9.5	Çıkarımlar	193
9.6	Gelecek Çalışmalar.....	196
10.	REGRESYON KOMİTELERİ.....	197
10.1	Önceki Çalışmalar	197
10.1.1	Komite Algoritmaları.....	197
10.1.2	Regresyon Algoritmaları.....	197
10.2	Gerçekleştirilenler	198
10.3	Çıkarımlar ve Gelecek Çalışmalar.....	203
11.	SONUÇ	204
	KAYNAKLAR.....	214
	INTERNET KAYNAKLARI.....	219
	EKLER.....	220
	Ek 1 Cline: A New Decision Tree Family	221

Ek 2 Clusline: A New Clustering Algorithm	222
Ek 3 Cline: New Multivariate Decision Tree Construction Heuristics	223
ÖZGEÇMİŞ.....	224

KISALTMA LİSTESİ

TÜBİTAK Türkiye Bilimsel ve Teknik Araştırma Kurumu

QSAR	Quantitative Structure - Activity Relationship (Kantitatif yapı – etki ilişkileri)
QSPR	Quantitative Structure - Property Relationship (Kantitatif yapı – özellik ilişkileri)
CADD	Computer Aided Drug Design (Bilgisayar Destekli İlaç Tasarımı)
HTS	High Troughput Screening
LTS	Low Troughput Screening
SVM	Support Vector Machines (Karar Destek Makineleri)
LDA	Linear Discriminant Analysis (Lineer Ayırıcı Analizi)
PCA	Principal Component Analysis (Temel Bileşen Analizi)
PCR	Principal Component Regression (Temel Bileşen Tabanlı Regresyon)
MLR	Multiple Linear Regression (Çoklu Lineer Regresyon)
PLS	Partial Least Squares (Parçalı En Küçük Kareler)
SOM	Self Organizing Map
VTYS	Veri Tabanı Yönetim Sistemi
SDF	Structure Data File
SMILES	Simplified Molecular Input Line Entry Specification
ANN	Yapay Sinir Ağları – Artificial Neural Networks
UCI	University of California at Irvine
IND	Investigative New Drug
FDA	Food and Drug Administration
MDL	Molecular Design Limited
MSE	Mean Squared Error (Ortalama Karesel Hata)
RMSE	Root Mean Squared Error (Ortalama Karesel Hatanın Karekökü)
RSE	Relative Squared Error (Rölatif Karesel Hata)
RRSE	Root Relative Squared Error (Rölatif Karesel Hatanın Karekökü)
CC	Corelation Coefficient (Korelasyon Katsayısı)
MAE	Mean Absolute Error (Ortalama Mutlak Hata)
RAE	Relative Absolute Error (Rölatif Mutlak Hata)

ŞEKİL LİSTESİ

Şekil 1.1 Sınıflandırma Problemi (Artı örneği hangi sınıftan?)	2
Şekil 2.1 İlaç - protein etkileşimi.....	5
Şekil 2.2 İlaçta araştırma ve geliştirme süreci.....	8
Şekil 2.3 İlaç geliştirme sürecinde aday moleküllerin ortalama sayıları	8
Şekil 2.4 HTS işleminde kullanılan robot (Hallborn vd., 2002).	10
Şekil 2.5 Örnek bir SDF dosyası	13
Şekil 2.6 Terpenoid molekülünün 3D yapısı sol ve ortada, 2D yapısı sağda [4]	16
Şekil 2.7 Moleküllerden sayısal özelliklerin çıkarılması	16
Şekil 2.8 Sadece eğitim verilerine göre belirlenen ayırıcı(düz çizgi) – test örneklerini de kullanan <i>transductive</i> ayırıcı (kesikli çizgi)	19
Şekil 3.1 Bir sınıflandırıcının parametre sayısının eğitim ve test kümeleri üzerindeki performanslarına etkisi	23
Şekil 3.2 Aynı veri kümesinde tek ve çok değişkenli karar ağaçları.....	27
Şekil 3.3 Cline versiyonları	33
Şekil 3.4 (a) Veri kümesi ve sınıflandırma doğruları (b) Cline karar ağacı.....	34
Şekil 3.5 Bazı veri kümelerinin CL2 ile yapılan sınıflandırmaları.....	35
Şekil 3.6 Metotların oluşturdukları karar sınırları.....	36
Şekil 3.7 Bire karşı bir ve bire karşı hepsi metotlarının karşılaştırılması için veri kümesi	39
Şekil 3.8 Hiper düzlem belirleme metotlarının karşılaştırılması.....	41
Şekil 3.9 Karar ağacı algoritmalarının karşılaştırılması.....	42
Şekil 4.1 En başarılı 8 Cline ve rasgele orman versiyonlarının karşılaştırılması.....	57
Şekil 4.2 Algoritmaların karmaşıklıklarıyla performansları arasındaki ilişki.....	62
Şekil 5.1 Kümeleme için örnek veri kümesi	67
Şekil 5.2 Örneklerden elde edilen hiyerarşik ağaç	74
Şekil 5.3 ClusLine versiyonları	76
Şekil 5.4 Clusline_2 ile adım adım kümeleme işlemi	76
Şekil 5.5 İki farklı veri kümesinde ClusLine2 ile SOM'un karşılaştırılması.....	77
Şekil 5.6 Kümeleme algoritmalarının farklı veri kümeleri üzerindeki sınıflandırma doğrulukları.....	79
Şekil 5.7 Davies-Bouldin indeksi ve siluet genişliği ölçütlerine göre kümeleme algoritmalarının başarıları	80
Şekil 6.1 Üç farklı kümeleme sonucunun birleştirilmesi (Renklerin açıklığı ilişkinin gücüyle doğru orantılıdır)	83
Şekil 6.2 Kümeleyici sayısının performansa ve standart sapmaya etkisi (özellik sayısı $2\log_2 M$).....	89
Şekil 6.3 Kümeleyici sayısının performansa ve standart sapmaya etkisi (özellik sayısı $\log_2 M$).....	89
Şekil 6.4 Kümeleme komitelerinde özellik sayısının performansa ve standart sapmaya etkisi.....	91
Şekil 7.1 1, 2 ve 3 boyutlu uzaylarda merkeze yakın noktalar.....	93
Şekil 7.2 Sınıflandırıcı parametreleriyle özelliklerin ağırlıklarının ilişkisi	96
Şekil 8.1 Aynı doğru üzerinde yer alan örnekleri modelleyebilen düzlemler.....	102
Şekil 8.2 Verilerin PCA ile 2 boyutlu uzaydan 1 boyutlu uzaya dönüşümü.....	106
Şekil 8.3 PLS ve PCA'in bulduğu boyutlar	107
Şekil 8.4 Farklı hata eşikleri için oluşturulmuş regresyon ağaçları ve çıkışları.....	108
Şekil 8.5 Örnek bir M5 model ağacının alt uzay sınırları	109
Şekil 8.6 Örnek bir M5 model ağacı.....	109
Şekil 8.7 Kmeans ile regresyon (K=3 için).....	114
Şekil 8.8 KmeansMOD ile regresyon (K=3 için).....	119
Şekil 8.9 KmeansXY ile örnekleri kümeleme (K=2 için).....	120
Şekil 9.1 Friedman koleksiyonunun meta özelliklerinin korelasyon matrisi.....	142
Şekil 9.2 Yapay veri koleksiyonunda algoritmaların RMSE değerlerine göre (80 boyutlu)	

hiyerarşik kümelenmeleri	147
Şekil 9.3 Yapay veri koleksiyonunda veri kümelerinin RMSE değerlerine göre (29 boyutlu)	
hiyerarşik kümelenmeleri	148
Şekil 9.4 İlaç veri koleksiyonunda 299 meta özelliğin korelasyon matrisi.....	164
Şekil 9.5 İlaç veri koleksiyonunda algoritmaların RMSE değerlerine göre (41 boyutlu)	
hiyerarşik kümelenmeleri	168
Şekil 9.6 İlaç veri koleksiyonunda veri kümelerinin RMSE değerlerine göre hiyerarşik	
kümeleme sonucu.....	169
Şekil 9.7 UCI veri koleksiyonunda meta özelliklerin korelasyon matrisi	180
Şekil 9.8 UCI veri koleksiyonunda algoritmaların RMSE değerlerine göre (60 boyutlu)	
hiyerarşik kümelenmeleri	182
Şekil 9.9 UCI veri koleksiyonunda veri kümelerinin RMSE değerlerine göre hiyerarşik	
kümelenmeleri	183
Şekil 9.10 8 farklı bileşen değeri için, 33 ilaç veri kümesinde PLS algoritmasının hata	
değerleri.....	191
Şekil 9.11 K değerinin hata üzerinde oluşturduğu değişimin standart sapmalarının histogramı	192
Şekil 10.1 Komitelerin 36 ilaç veri kümesi üzerindeki performanslarına göre gruplandırma	
sonuçları	202

ÇİZELGE LİSTESİ

Çizelge 2.1 Örnek bir QSAR veri kümesi.....	12
Çizelge 2.2 Çeşitli moleküllerin SMILES formatında gösterimleri	14
Çizelge 2.3 Tezde kullanılan molekül özellikleri (toplam 1142 adet).....	17
Çizelge 3.1 ID3 algoritması.....	25
Çizelge 3.2 Popüler karar ağacı algoritmaları	29
Çizelge 3.3 Cline algoritması	29
Çizelge 3.4 ECOC kod matrisleri	38
Çizelge 3.5 Veri kümelerinin özellikleri.....	40
Çizelge 3.6 Cline metotlarının ortalama doğruluk oranları ve varyansları.....	40
Çizelge 3.7 Karşılaştırma yapılan karar ağacı algoritmaları ve kısaltmaları	42
Çizelge 3.8 Kullanılan ilaç veri kümeleri	43
Çizelge 3.9 Tek ağaçlı Cline algoritmalarının sonuçları.....	43
Çizelge 3.10 Cline ile karşılaştırılan diğer algoritmaların sonuçları	45
Çizelge 3.11 Cline ile karşılaştırılan diğer algoritmaların sıralama sonuçları	45
Çizelge 4.1 Üç farklı sınıflandırıcının kararlarının birleştirilmesi	52
Çizelge 4.2 Sınıflandırıcıların kararlarının birleştirilmesinde sınıflandırıcı sayılarının ve performanslarının etkisi.....	53
Çizelge 4.3 Cline ve rasgele orman versiyonlarının ortalama başarıları	56
Çizelge 4.4 Bootstrapping'in performansa etkisi	58
Çizelge 4.5 Cline ormanlarının ve Random Forest'ların farklı versiyonlarının ortalama başarıları ve standart sapmaları.....	60
Çizelge 4.6 Algoritmaların her bir veri kümesinde maksimum başarının standart sapma miktarı yakınında kalanları	61
Çizelge 4.7 İlaç veri kümelerinde Cline ile karşılaştırılan diğer algoritmaların sonuçları	63
Çizelge 4.8 Cline algoritmalarının veri koleksiyonlarına göre karşılaştırılmaları	65
Çizelge 5.1 Clusline algoritması.....	75
Çizelge 5.2 Kümelemede kullanılan veri kümeleri ve özellikleri	78
Çizelge 5.3 Kümeleme algoritmalarının ortalama doğrulukları ve sıralama ortalamaları.....	79
Çizelge 6.1 10 adet örneğin 5 farklı kümeleme algoritmasıyla kümeleneşinden elde edilen sonuçlar	82
Çizelge 6.2 Küme benzerliğinin ölçümünde kullanılacak yaklaşımın seçim denemeleri.....	85
Çizelge 6.3 Kümeleme komiteleri için kullanılan veri kümeleri	86
Çizelge 6.4 Küme sayısının sınıf sayısına eşit seçildiğindeki performanslar	87
Çizelge 6.5 Küme sayısının sınıf sayısının iki katı seçildiğindeki performanslar	88
Çizelge 6.6 Kümeleme komitelerinden elde edilen özet sonuçlar.....	88
Çizelge 7.1 Boyut sayısı ile merkeze yakın noktaların oranı arasındaki ilişki.....	94
Çizelge 7.2 Boyut sayısı ile uzayı yeterli bir biçimde ifade etmek için gerekli örnek sayısı arasındaki ilişki	95
Çizelge 7.3 Özellik seçiminde kullanılan veri kümeleri	97
Çizelge 7.4 Özellik seçim metotlarının karşılaştırılması	98
Çizelge 8.1 Örnek regresyon verisi.....	100
Çizelge 8.2 Kmeans, KNN ve ANN'lerin ürettikleri fonksiyonlar, Kmeans ile regresyonun ortalama karesel hatasının eğitim boyunca değişimi, metotların test verileri üzerindeki ortalama karesel hataları.....	112
Çizelge 8.3 KmeansS, KNN ve ANN'lerin ürettikleri fonksiyonlar, Kmeans ile regresyonun ortalama karesel hatasının eğitim boyunca değişimi, metotların test verileri üzerindeki ortalama karesel hataları (<i>alfa</i> ve <i>safla</i> değeri)	115
Çizelge 8.4 Regresyonda kullanılan veri kümeleri [15]	120
Çizelge 8.5 Regresyon sonuçları	121

Çizelge 9.1 Farklı sayıda meta özellik içeren veri kümeleri	125
Çizelge 9.2 Meta özellik grupları	129
Çizelge 9.3 STA meta özellik grubu	130
Çizelge 9.4 ST2 meta özellik grubu	130
Çizelge 9.5 CLUS meta özellik grubu	134
Çizelge 9.6 REGT meta özellik grubu	134
Çizelge 9.7 RMSE meta özellik grubu	136
Çizelge 9.8 PCA meta özellik grubu	138
Çizelge 9.9 Friedman koleksiyonu için ek meta özellikler	139
Çizelge 9.10 80 veri kümesinde minimum RMSE değeri içeren algoritmalar	140
Çizelge 9.11 Meta algoritmalar çıkarıldığında 80 veri kümesinde minimum RMSE değeri içeren algoritmalar	140
Çizelge 9.12 21 algoritmanın 80 veri kümesi üzerindeki performansları	141
Çizelge 9.13 Friedman koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı	143
Çizelge 9.14 Friedman meta özelliklerinin korelasyonu yüksek olanlarından seçmeler	143
Çizelge 9.15 Friedman koleksiyonunda Meta.Bagging tahminleri	150
Çizelge 9.16 Friedman koleksiyonunda M5P tahminleri	151
Çizelge 9.17 Friedman koleksiyonunda meta.AttributeSelectedClassifier tahminleri	152
Çizelge 9.18 Friedman koleksiyonunda meta.RandomSubSpace tahminleri	154
Çizelge 9.19 Friedman koleksiyonunda Reptree tahminleri	155
Çizelge 9.20 Friedman koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları	156
Çizelge 9.21 Friedman koleksiyonunda performans tahmininde kullanılan meta özelliklerden kurallarda en çok yer alanları	157
Çizelge 9.22 Kullanılan ilaç veri kümeleri ve referansları	158
Çizelge 9.23 İlaç veri koleksiyonunda kullanılan ek meta özellikler	161
Çizelge 9.24 41 İlaç veri koleksiyonunda minimum RMSE değeri içeren algoritmalar	162
Çizelge 9.25 İlaç veri koleksiyonunda algoritmaların 41 veri kümesi üzerindeki performansları	162
Çizelge 9.26 İlaç veri koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı	165
Çizelge 9.27 İlaç veri koleksiyonunda yüksek korelasyonlu meta özelliklerden seçmeler ...	165
Çizelge 9.28 İlaç veri koleksiyonunda PLS1 algoritmasının performans tahminleri	170
Çizelge 9.29 İlaç veri koleksiyonunda Kstar algoritmasının performans tahminleri	171
Çizelge 9.30 İlaç veri koleksiyonunda M5P algoritmasının performans tahminleri	172
Çizelge 9.31 İlaç veri koleksiyonunda IBK algoritmasının performans tahminleri	173
Çizelge 9.32 İlaç koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları	174
Çizelge 9.33 İlaç koleksiyonunda performans tahminlerinde en çok kullanılan meta özellikler	175
Çizelge 9.34 Kullanılan UCI veri kümeleri ve özellikleri	176
Çizelge 9.35 UCI veri koleksiyonunda kullanılan ek meta özellikler	178
Çizelge 9.36 60 UCI veri kümesinde minimum RMSE değeri içeren algoritmalar	178
Çizelge 9.37 UCI veri koleksiyonunda 18 algoritmanın 60 veri kümesi üzerindeki performansları	179
Çizelge 9.38 UCI koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı	181
Çizelge 9.39 UCI koleksiyonunda birbiriyle korelasyonu yüksek olan meta özelliklerden seçilmiş örnekler	181
Çizelge 9.40 UCI veri koleksiyonunda M5P algoritmasının performans tahminleri	184
Çizelge 9.41 UCI veri koleksiyonunda PLS2 algoritmasının performans tahminleri	185
Çizelge 9.42 UCI veri koleksiyonunda Kstar algoritmasının performans tahminleri	186

Çizelge 9.43 UCI veri koleksiyonunda Isotonic Reg algoritmasının performans tahminleri	187
Çizelge 9.44 UCI koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları	189
Çizelge 9.45 UCI veri koleksiyonunda performans tahminlerinde en çok kullanılan meta özellikler	189
Çizelge 9.46 PLS algoritmasında bileşen sayısının incelenmesinde kullanılan ilaç veri kümeleri	190
Çizelge 9.47 Veri kümesi koleksiyonlarına göre elde edilen sonuçların özetleri	194
Çizelge 9.48 Üç veri kümesi koleksiyonunda performans tahminlerinde en sık kullanılan meta özelliklerin kesişim kümesi	195
Çizelge 10.1 Kullanılan komite oluşturma algoritmaları	198
Çizelge 10.2 Kullanılan regresyon algoritmaları	198
Çizelge 10.3 Regresyon komiteleri denemelerinde kullanılan veri kümeleri	199
Çizelge 10.4 25 Komite-Algoritma ikililerinin 36 veri kümesi üzerindeki sıralamaları	200
Çizelge 10.5 Algoritmaların ve komite metotlarının ortalama sıraları	200
Çizelge 10.6 Meta algoritma- regresyon algoritması ikililerinin başarı sıralamalarının ortalamaları	201
Çizelge 10.7 Regresyon komitelerinde cevap aranan sorulara deneylerin verdiği cevaplar	203

ÖNSÖZ

İlaçların hayatımızdaki tartışılmaz önemleri ve etkileri, yeni ilaçların tasarlanmasına olan gereksinimi her geçen gün arttırmaktadır. Bununla birlikte ilaç tasarımı maliyeti çok yüksek bir araştırma alanıdır ve ancak ekonomik gücü çok büyük olan firmalar ve devletler tarafından gerçekleştirilebilmektedir. Türkiye ise bu alanda henüz sesini dünyaya duyurabilmiş bir ülke değildir. Ancak Türkiye'nin 2023 yılı vizyonunda “ilaç tasarımı alanında dünyada söz sahibi ülkeler arasında olmak” hedefleri arasında yer almaktadır.

İlaç tasarımı temelde kimyacılar ve eczacıları ilgilendiren bir alan olmakla birlikte geçmiş 50 yıl içinde bu alanda yaşanan gelişmeler bilgisayar tabanlı ilaç tasarım teknolojilerinin vazgeçilmez olmasına sebep olmuştur. Bu tezde, ilaç tasarımı alanındaki çeşitli problemlere çözüm üretebilecek algoritmalar üretilmiş ve araştırmacıların kullanımına sunulmuştur.

Çalışmam boyunca beni destekleyen; tez yöneticilerim Prof. Dr. Oya Kalıpsız ve Prof. Dr. Okan Ersoy'a, desteğini ve ilgisini her zaman hissettiğim hocam Yrd. Doç. Dr. Banu Diri'ye, beni değerli yorumlarıyla her zaman yönlendiren Doç. Dr. Berrin Yanıkoğlu'na, destekleri ve yardımları için çalışma ve oda arkadaşlarıma teşekkür ederim.

Bu tezi beni hayatım boyunca her şartta destekleyen sevgili annem Semra Amasyalı'ya, sevgili halam Aygün Tekneci'ye ve sevgili eşim Seyhan Amasyalı'ya ithaf ediyorum. Sizler olmasanız bunu başaramazdım.

ÖZET

Makine öğrenmesi (yapay öğrenme), eldeki verileri en iyi temsil eden modeli ve parametrelerini bulmak amacıyla geliştirilen algoritmaları içerir. Tezde çeşitli makine öğrenmesi algoritmaları geliştirilmiştir. Günümüzde incelenmesi ve yorumlanması gereken veri miktarı üssel bir biçimde artmaktadır. Bu durum makine öğrenmesinin tüm sektörlerin ihtiyaç duyduğu bir alan haline gelmesine sebep olmuştur. Tezin uygulama alanı olarak, bu sektörlerden biri olan ilaç tasarımı seçilmiştir.

İlaçların insan sağlığına olan olumlu etkisi bilinmektedir. Yeni ilaç tasarımı bu nedenle çok önemli ve vazgeçilmezdir. Buna karşılık çok emek ve uzun zaman isteyen ve buna bağlı olarak çok büyük maliyetler içeren bir sektördür. Yüksek maliyet sebebiyle az sayıda firma tarafından gerçekleştirilebilmektedir. Türkiye’de bu alandaki mevcut çalışmalar sınırlı olmakla birlikte TÜBİTAK tarafından yayınlanan raporda ilaç tasarımı, 2003-2023 yıllarını kapsayan dönemde öncelikli teknolojik faaliyet konuları içinde yer almaktadır.

İlaç tasarımı sürecinin ve maliyetinin önemli bileşenlerinden biri olası ilaç moleküllerinin seçilmesi işlemidir. Bu seçim işlemleri genelde; sınıflandırma, kümeleme, özellik seçimi/çıkarımı, regresyon (eğri uydurma) problemlerinden bir ya da birkaçını içermektedir. Bu tarz problemlere çözüm üretmeyi amaçlayan makine öğrenmesi metodları yardımıyla ilaç tasarımının süresi ve maliyeti azaltılabilmektedir.

Görüldüğü gibi ilaç tasarımı problemlerinde makine öğrenmesinin neredeyse tüm alanlarına ihtiyaç duyulmakta ve kullanılmaktadır. Bu nedenle de tezde makine öğrenmesinin birçok alanını kapsayacak bir çalışma gerçekleştirilmiştir.

Sınıflandırma problemleri için **Cline** adı altında bir algoritma ailesi tasarlanmıştır. Geliştirilen algoritmalar temelde karar ağacı oluşturma algoritmalarıdır. Karar ağaçları, yüksek performansları ve ürettikleri kuralların verinin yapısına ait çıkarımlar yapmayı kolaylaştırması sebebiyle oldukça popüler olmuş makine öğrenmesi algoritmalarındandır. Yapılan denemelerde geliştirilen algoritmaların basitliklerine rağmen UCI ve ilaç veri kümelerinde mevcut algoritmalarla yarışabilecek performansta algoritmalar oldukları görülmüştür.

Sınıflandırıcı komiteleri literatürdeki birçok çalışmada tekil sınıflandırıcılardan daha başarılı sonuçlar üretmiştir. Bu çalışmada da buna paralel sonuçlar alınmış ve Cline algoritma ailesine Cline karar ormanları eklenmiştir. UCI ve ilaç veri kümeleri üzerinde Cline karar ormanları mevcut algoritmalarından çok daha iyi sonuçlar sergilemiştir. Cline karar ağacı ve karar ormanları algoritmaları ClineToolbox adlı bir yazılımla kullanıcıların hizmetine sunulmuştur. Yazılıma tez sahibinin web sayfasından erişilebilir.

Özellik seçimi problemleri için karar ağaçları ve karar ormanlarından yararlanan bir yaklaşım geliştirilmiş ancak tatmin edici sonuçlar elde edilememiştir.

Kümeleme problemleri için **Clusline** adı altında bir algoritma ailesi geliştirilmiş ve mevcut algoritmalarla çeşitli kümeleme performans kriterlerine göre yarışan sonuçlar elde edilmiştir.

Kümeleme komiteleri, sınıflandırıcı komitelerinin üstün performanslarından esinlenilerek geliştirilmiştir. Literatürdeki kümeleme komitelerinin farklı karar birleştirme teknikleri incelenmiş ve geniş bir veri kümesi üzerinde bu teknikler karşılaştırılmıştır. Literatürdeki mevcut karşılaştırmalardan daha kapsamlı olan bu çalışma bu konuda çalışanlara yol gösterici niteliktedir.

Regresyon problemleri için verileri çeşitli alt uzaylarda kümelemeye dayalı bir yaklaşım geliştirilmiş ancak tatmin edici sonuçlar alınamamıştır.

Regresyon komiteleri için, literatürdeki komite oluşturma, karar birleştirme metotlarının ve komitelerde yer alan regresyon algoritmalarının ilaç tasarımı veri kümelerinde performans üzerindeki etkileri incelenmiştir. Bu kapsamlı çalışmada, ilaç veri kümelerinde regresyon komitelerinin kullanımının sınıflandırma da olduğu kadar sonucu iyileştirmedeği görülmüştür.

Bütün veri kümelerinde diğer tüm algoritmalarından daha iyi sonuç veren global bir algoritma bulunmamaktadır. Bu nedenle, bir veri kümesinin hangi algoritma ile en iyi sonucu vereceği genelde deneme yanılma metoduyla bulunmaktadır. Literatürde bu eksikliği gidermek ve son kullanıcılara yardımcı kurallar dizisi oluşturabilmek için algoritmaların performanslarının veri kümesinin çeşitli özelliklerine göre tahmin edilmesi amacını taşıyan yaklaşımlar geliştirilmiştir. Bu yaklaşımların genel adı Meta-Öğrenim'dir. Mevcut Meta-öğrenim çalışmalarında genelde sınıflandırma problemleri üzerine çalışılmıştır. İlaç veri kümelerindeki problemlerin büyük bir kısmı regresyon türünden problemler olduğu için bu çalışmada yeni bir Meta-Regresyon yaklaşımı da geliştirilmiştir. Geliştirilen yaklaşımda Meta-öğrenimde kullanılan standart veri kümesi özelliklerine ek yeni özellikler de kullanılmıştır. Çalışma sonunda bir veri kümesi üzerinde bir algoritmanın performansı veri kümesinin çeşitli özelliklerine bakarak tahmin edilebilen bir model geliştirilmiştir. Bu sayede bir veri kümesinde en iyi performansı gösterecek algoritma da tahmin edilebilmektedir. Ayrıca veri kümelerinin ve algoritmaların birbirlerine benzerliklerine göre kümeleneceği konusunda da çalışılmıştır.

Sonuç olarak bu tezde, makine öğrenmesinin çeşitli konularında birçok yeni yaklaşım geliştirilmiş ve bu konuda çalışan araştırmacılar ve son kullanıcılar için faydalı olacak sonuçlar üretilmiştir. Bu tezin hem ilaç tasarımı hem de makine öğrenmesi konularında Türkiye'de ve Dünya'da yapılan çalışmalara katkıda bulunması dileğimizdir.

Anahtar kelimeler: Makine öğrenmesi, Yapay Öğrenme, Sınıflandırma, Sınıflandırıcı komiteleri, Kümeleme, Kümeleme komiteleri, Özellik seçimi, Özellik çıkarımı, Eğri uydurma, Regresyon, Regresyon komiteleri, Meta özellik, Meta-öğrenim, İlaç tasarımı, Bilgisayar destekli ilaç tasarımı, Rasyonel ilaç tasarımı.

ABSTRACT

Machine learning includes the algorithms which aim to find the best-fit model to the data. In this thesis, several machine learning algorithms are developed. Nowadays, the data need to be investigated is growing exponentially. Therefore, machine learning is needed in all sectors. Drug design is selected as the application area of the thesis.

Drugs are very useful to maintain good health. This is why drug design very important and necessary. Drug design is also very costly, and requires much effort and time to develop. Because of heavy cost, only some large companies have the capability to work in this area. In a report by TUBITAK, drug design has been declared a research direction of high priority for Turkey during 2003 – 2023 planning period.

One of the important components of drug design and cost is the process of choosing potential drug molecules. Those choosing operations usually contain one or more of classifying, clustering, feature selection, regression problems. By using machine learning methods, that process time and cost involved in these operations, can be minimized.

In drug design problems, all subjects in machine learning are almost used. So, in this thesis, a study that contains most such machine learning topics has carried out.

For classification problems, a new algorithm family called **Cline** has been designed. These algorithms are decision tree induction algorithms. Although the algorithms are simple, experiments have shown that they have competitive performance as compared to existing algorithms on UCI and drug datasets.

In previous studies, it has been observed that classifier committees have more successful results than single classifiers'. Also, in this study, similar results have been obtained. So Cline decision forests have been added to Cline algorithms family. Cline decision forests have produce better performance than existing algorithms on UCI and drug design. Those Cline decision tree and decision forest algorithms have been serviced to end users via Cline Toolbox application and web site of the thesis owner.

For feature selection problems, an approach that uses decision tree and decision forest has been developed but needs further improvement.

For clustering problems, an algorithm family called **Clusline** has been developed and some satisfying results have been achieved as compared to other existing algorithms.

Clustering committees have been developed, inspired by the excellent performance of classifying committees. Different decision combination techniques of clustering committees in the literature have been investigated and compared. Our study is one of the most comprehensive studies, and our results can be used as a guideline for researchers.

For regression problems, an approach based on data clustering in subspaces has been developed but needs further improvement.

For regression committees, the effects of algorithms used in committee, different generating and decision combination techniques of committees in the literature have been investigated and compared. The experiments on drug design datasets show that the usage of committees for regression problems gives more or less similar performance with single regressors.

There is no global algorithm that always gives better result than other algorithms for all datasets. So, trial and error is the method to find the best algorithm for a dataset. To eliminate this lack and provide helper rule series to end users, approaches that aim to estimate

algorithms' performance according to the meta features of datasets have been developed. Those approaches are called meta learning. Current meta learning studies are usually for classification problems. Given that most problems in drug data design are regression problems, in this study a new meta regression approach has been developed. By this approach, new meta features have also been extracted in addition to standard features used in meta learning. So, a new model has been developed that can estimate algorithm performance for a dataset by using various dataset features. In this way, the best performable algorithm for a dataset can be estimated in advance. Besides, clustering datasets and algorithms according to similarities have also been investigated.

Consequently, in this thesis, many new approaches about various machine learning subjects have been developed and useful results for researchers and end users have been produced. It is our wish that this thesis contribute to studies about both drug design and machine learning research areas in Turkey and the world.

Keywords: Machine learning, Classification, Classifier ensembles, Clustering, Cluster ensembles, Feature selection, Feature extraction, Function approximation, Regression, Regression ensembles, Meta regression, Meta feature, Dataset descriptors, Drug design, Computer aided drug design, Rational drug design.

1. GİRİŞ

Makine öğrenmesi alanında literatürde birçok algoritma mevcuttur. Ancak tüm problemlere en iyi çözümü üreten tek bir algoritma yoktur. Bu nedenle var olan ve yeni ortaya çıkan problemler için yeni algoritmalar, metotlar geliştirme güncelliğini korumaktadır. Bu tezde makine öğrenmesi alanında yer alan birçok problem için yeni yaklaşımlar önerilmiş ve bir uygulama alanı olarak da ilaç tasarımı problemleri seçilmiştir.

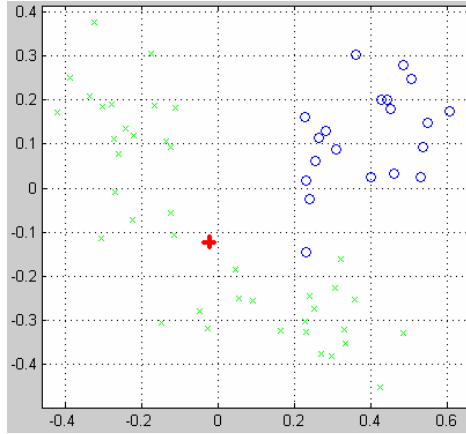
Tezde uygulama alanı olarak ilaç tasarımının seçilme nedenleri olarak Türkiye'nin gelecek vizyonunda bilgisayar destekli ilaç tasarımı konusunun yer alması, çalışmanın bu konudaki araştırmacılara fayda sağlayacak olması ve makine öğrenmesi metotlarının elde ettikleri birçok başarı olmasına rağmen ilaç tasarımı gibi genelde zor modellenebilen veri tabanları için çalışmaların hala yoğun bir şekilde devam ediyor olması verilebilir. Bununla birlikte geliştirilen metotların sadece ilaç tasarımı uygulamaları için değil her türlü sınıflandırma, kümeleme ve regresyon problemlerine uygulanabilecek yapıda olması makine öğrenmesi alanına katkı sağlamıştır.

1.1 Makine Öğrenmesi Nedir?

Bilgi teknolojilerindeki gelişmeler sayesinde artık çok büyük miktarlarda veriyi kaydedebilmekteyiz. Süpermarket kasalarından, para çekme makinelerinden, kredi kartı cihazlarından, e-ticaret uygulamalarından her an milyonlarca veri, verilerin saklandığı merkezlere ulaşmaktadır. Bunlara ek olarak, bir hastanedeki röntgen cihazından, bir güvenlik kamerasından, bir iris tanıma sisteminden, bir kumaş kalite ölçüm kamerasından, bir borsadaki işlemlerden yine birçok veri elde edilmekte ve analiz için beklemektedir. Bu işlemlerin her birinde analizden farklı şeyler beklenmektedir. Bir süpermarket işletmecisi hangi tür ürünlerin bir arada satıldığını, bir borsa analisti hisse senedinin yarınki değerini, iris tanıma sistemi verinin kime ait olduğunu, kredi kartı sistemi kartı kullananın kredi kartının sahibi olup olmadığını, bir güvenlik kamerası olağandışı bir durum olup olmadığını öğrenmek istemektedir. Anlatılan sistemlerin tümünde geçmişteki veriler, o anki verilerin değerlendirilip yorumlanmasında kullanılmaktadır. Ancak çok büyük miktardaki verilerin elle işlenmesi, analizinin yapılması mümkün değildir. Bu problemlere çözüm bulmak amacıyla makine öğrenmesi metotları geliştirilmiş ve geliştirilmeye devam edilmektedir. Makine öğrenmesi metotları geçmişteki verileri kullanarak veriye en uygun modeli bulmaya çalışırlar. Yeni gelen verileri de bu modele göre analiz ederler. Büyük miktarda verinin incelenip onun içinden işe yarayan bilginin (modelin) elde edilmesi işlemine veri madenciliği (data mining)

de denilmektedir. Farklı uygulamaların analizlerden farklı beklentileri olmaktadır. Makine öğrenmesi metodlarını bu beklentilere göre sınıflandırmak mümkündür (Alpaydın, 2004).

1. Sınıflandırma: Geçmiş bilgilerin hangi sınıflara ait olduğu verildiğinde yeni gelen verinin hangi sınıfa dahil olduğunun bulunması işlemidir. Örnek olarak iki tahlil sonucuna göre bir kişinin hasta olup olmadığı belirlenmeye çalışılırsa önceki hasta ve sağlam kişilerin tahlil sonuçları kullanılır. Şekil 1.1’de iki tahlil sonucu X ve Y eksenleriyle, hasta kişiler mavi yuvarlaklarla, sağlam kişiler yeşil çarpılarla, hasta olup olmadığı merak edilen kişinin tahlil sonuçları kırmızı artıyla gösterilmiştir.



Şekil 1.1 Sınıflandırma Problemi (Artı örneği hangi sınıftan?)

2. Kümeleme: Geçmiş bilgilerin sınıflarının / etiketlerinin verilmediği / bilinmediği durumlarda verilerden birbirine benzerlerin yer aldığı kümelerin bulunması işlemidir.
3. Eğri Uydurma (Regresyon): Geçmiş verilerin sınıflarının sürekli sayılar olduğu durumlarda kullanılır. Örneğin bir hisse senedinin değeri sürekli bir sayıdır ve bu senede ait model bu değeri tahmin etmeye yönelik bir eğri uydurma işlemi olacaktır.
4. Özellik Seçimi / Çıkarımı: Veriye ait birçok özellikten, verinin kümesini / sınıfını / değerini belirleyen özelliklerinin hangileri olduğu bilinmeyebilir. Bu durumda tüm özellik kümesinin bir alt kümesi seçilir (özellik seçimi) ya da bu özelliklerin birleşimlerinden yeni özellikler elde edilir (özellik çıkarımı).
5. İlişki Belirleme: Bir süpermarkette X ürününü alan müşterilerden %80’i Y ürününü de alıyorsa, X ürününü alıp Y ürününü almayan müşteriler, Y ürününün potansiyel müşterileridir. Müşterilerin sepet bilgilerinin (bir alışverişte alınan ürün bilgileri) bulunduğu bir veritabanında potansiyel Y müşterilerini bulma işlemi türündeki

problemler iliřki belirleme metotlarıyla çözülmektedir.

Sonuç olarak **makinelere insanlığın işgücüne sağladıkları katkıyı, makine öğrenmesi metotları sayesinde insanlığın beyin gücüne de sağlamaya başlamışlardır.** Her tür uygulama için çok miktarda verinin analiz edilerek gelecekle ilgili varsayımlar geliřtirmemize, kararlar vermemize yardımcı olan makine öğrenmesi metotları önemlerini ve katkılarını her geçen gün arttırmaktadır.

1.2 Tezdeki Bölümlerin Tanıtımı

Tez başlıca 11 bölümden oluşmaktadır. Bu bölümlerde tezin uygulama alanı olan ilaç tasarımı ve makine öğrenmesinin temel problemleri için geliştirilen yaklaşımlar anlatılmıştır.

İlaçların insan sağlığına katkıları ve daha birçok hastalık için etkin tedavi yöntemleri geliştirilemediği düşünöldüğünde yeni ilaç tasarımının hayati önemi ortaya çıkmaktadır. Ancak yeni ilaç tasarımı çok büyük zaman ve maliyet isteyen bir işlemdir. Çok büyük sayıdaki olası moleküllerden hangilerinin ilaç olacağını tespit edebilmek için uzun hesaplamalar ve laboratuvar deneyleri gerekmektedir. Bu uzun işlemi bilgisayarlarla ve özellikle makine öğrenmesi metotlarıyla işbirliği içinde yapmak hem zaman hem de maliyet yönünden önemli tasarruflar sağlamaktadır. Tezin 2. bölümünde bu işbirliğinin temel birimleri tanıtılmıştır.

Tezin 3. bölümünde sınıflandırma problemleri için tasarlanan **Cline** adında bir algoritma ailesi anlatılmıştır.

Sınıflandırıcı komiteleri literatürdeki birçok çalışmada tekil sınıflandırıcılardan daha başarılı sonuçlar üretmiştir. Bu nedenle Cline karar ağaçlarının birleştirilmesiyle, Cline karar ormanları oluşturulmuş ve 4. bölümde tanıtılmıştır.

Özellik seçimi büyük boyutlu veri kümelerinde boyut sayısını azaltmak ya da eldeki özelliklerden hangilerinin çıkışla ilgisi olduğunun anlaşılması için kullanılan metotlardır. 5. bölümde özellik seçimi problemleri için karar ağaçları ve karar ormanlarından yararlanan bir yaklaşım verilmiştir.

Kümeleme, elde örneklere ait bir çıkışın olmadığı ya da veri kümesinin iç yapısının anlaşılmasının istendiği problemlerde kullanılır. 6. bölümde, kümeleme problemleri için geliştirilen **Clusline** adında bir algoritma ailesi anlatılmıştır.

Komitelerin sınıflandırıcılardaki yüksek performansı kümeleyicilerinde birleştirilmesi ifkrini

doğurmuştur. 7. bölümde literatürdeki kümeleme komitelerinin farklı karar birleştirme teknikleri incelenmiş ve geniş bir veri kümesi üzerinde bu teknikler karşılaştırılmıştır.

Veri kümesinin çıkışının bir etiket değil bir sayı olduğu durumlarda regresyon metotları uygulanır. 8. bölümde regresyon problemleri için geliştirilen, verileri çeşitli alt uzaylarda kümelemeye dayalı bir yaklaşım anlatılmıştır.

Bütün veri kümelerinde diğer tüm algoritmalarından daha iyi sonuç veren global bir algoritma bulunmamaktadır. Bu nedenle, bir veri kümesinin hangi algoritma ile en iyi sonucu vereceği genelde deneme yanılma metoduyla bulunmaktadır. Literatürde bu eksikliği gidermek ve son kullanıcılara yardımcı kurallar dizisi oluşturabilmek için algoritmaların performanslarının veri kümesinin çeşitli özelliklerine göre tahmin edilmesi amacını taşıyan yaklaşımlar Meta-Öğrenim adı altında geliştirilmiştir. 9. bölümde farklı yapılara sahip veri koleksiyonları üzerinde yeni ve mevcut veri kümesi tanımlayıcıları kullanılmış ve veri kümelerinde en başarılı algoritmanın tahmini, veri kümelerinin ve algoritmaların birbirlerine benzerlikleri konularında çalışmalar yapılmıştır.

10. bölümde ise regresyon komiteleri konusunda çalışılmış, literatürdeki komite oluşturma, karar birleştirme metotlarının ve komitelerde yer alan regresyon algoritmalarının ilaç tasarımı veri kümelerinde performans üzerindeki etkileri incelenmiştir.

Son bölümde her bölümde elde edilen sonuç ve çıkarımlar özetlenmiştir.

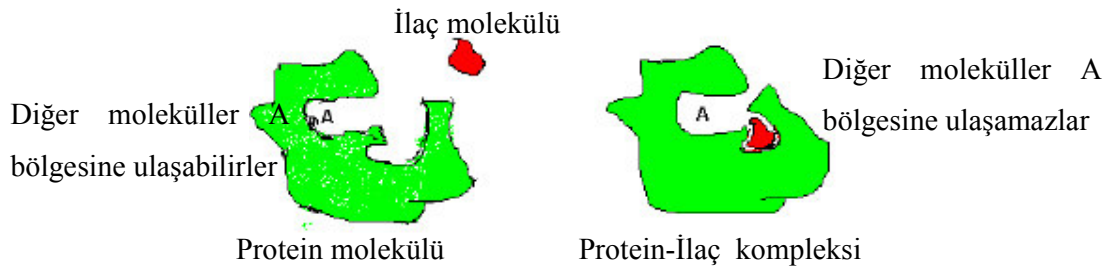
2. İLAÇ TASARIMI

Yeni ilaçların tasarımının önemi, ilaçların geçmiş zaman içerisinde sağladıkları faydalar düşünüldüğünde ortaya çıkmaktadır. Geçmiş zamanda birçok insanın ölümüne sebep olan birçok hastalık, geliştirilmiş olan ilaçlar sayesinde günümüzde ya yok olmuş ya da etkisini son derece azaltmıştır.

İlaç tasarımında günümüzde kullanılan teknolojiler sebebiyle, bilgisayarlı analiz vazgeçilmez bir rol üstlenmektedir. İşte bu bölümde öncelikle ilaçların nasıl çalıştıkları, ilaç tasarımının evreleri ve maliyetleri, dünyadaki ilaç sanayisinden söz edilmiş daha sonra moleküllerden, makine öğrenmesiyle işlenebilen sayısal veri kümelerinin oluşturulmasına kadar giden süreç ele alınmıştır.

2.1 İlaç Nedir, Nasıl Çalışır?

Canlı hücrelerindeki işlemler, moleküllerin birbirine bağlanmalarıyla gerçekleştirilir. Anahtar kilit ilişkisine sahip moleküller birleşerek önceki özelliklerinden farklı özelliklere sahip yeni moleküller oluştururlar. Canlılardaki birçok işlemin ardında bu birleşimler yatmaktadır. Protein-nükleik asit birleşmeleri protein sentezlenmesini sağlar. Kandaki hemoglobinin molekülleri oksijenle birleşerek oksijenin vücuda dağılmasını sağlarlar. Hastalıkların tedavisinde kullanılan ilaçlarda hastalığa sebep olan moleküllere bağlanarak onların özelliklerini değiştiren kimyasal bileşiklerdir. Şekil 2.1’de bir ilacın bir proteine bağlanarak onun özelliğini değiştirip diğer moleküllerle birleşik oluşturmasını önlemesi gösterilmiştir [1].



Şekil 2.1 İlaç - protein etkileşimi

2.2 Yeni İlaç Geliştirmenin Önemi

Günümüze kadar birçok hastalık için ilaçlar geliştirilmiştir. Yeni ilaçlar sayesinde; insanın yaşam süresinin son 100 yılda 40'lı yaşlardan 70'li yaşlara ulaşmış, insanların ameliyat olmadan, hastaneye yatmadan tedavi olabilmeleri sağlanmıştır (Baykara vd., 2003). Ancak

her geçen gün yeni hastalıklar, mevcut ilaçlara karşı bağışıklık kazanmış yeni bakterilerin ya da virüslerin neden olduğu hastalıklar gündeme gelmektedir. Ayrıca AIDS ve kanserin birçok türü için henüz bir ilaç tedavisi geliştirilememiştir. Bunun sonucu olarak yeni ilaçların tasarımı hiçbir zaman önemini kaybetmemiştir.

2.3 İlaçlardan Beklenen Özellikler

İlaç geliştirmede temel amaç, insan yaşamında daha iyiye doğru bir değişiklik yapabilmektir. Geliştirilen her ilaçtan koruyucu, tedavi edici veya hastalığın belirti ve bulgularını azaltıcı bir etkiye sahip olması ve yan etkilerinin kabul edilebilir seviyelerde olması beklenmektedir.

İlaçların hedef bir proteine bağlanıp, bağlandığı proteinin davranışını değiştiren küçük kimyasal moleküller olduklarından daha önce söz edilmişti. İlacın hedefindeki protein insan vücuduna ait bir protein olup onun davranışını değiştirerek kendisine zarar verecek diğer moleküllerle etkileşimini önleyebileceği gibi; bir bakterinin proteini olup onun davranış şeklini değiştirerek bakterinin ölmesine sebep olabilir. Bir molekülün ilaç olabilmesi için sadece hedef proteine tutunması yeterli değildir. Bunun yanında aşağıda listelenmiş kriterlerinde göz önünde tutulması gerekmektedir [2].

- Proteine çok sıkı ya da çok gevşek bağlanmama
- Toksin özellik göstermeme
- Vücutta yan etkileri olmama ya da kabul edilebilir seviyede olma
- Vücuttan gereğinden erken ya da geç atılmama
- Vücutta hedefin haricindeki bölgelere gitmeme
- Kan dolaşımına girebilme
- Hastalığa iyi yönde etki edebilme

Yukarıda listelenmiş özelliklerin tümü literatürde, İngilizce karşılıklarının (Absorption, Distribution, Metabolism, Excretion, Toxicity) baş harflerinden üretilen “ADMET” kelimesi ile ifade edilmektedir. Tezin devamında ADMET kelimesiyle yukarıdaki özellikler ifade edilmiştir.

2.4 İlaç Tasarımının Evreleri ve Maliyeti

Yeni ilaçların tasarımı, çoğu zaman algılandığı gibi, kimya ve ilaç sektörünün bir dalından

çok, esas olarak bir bilgi üretim endüstrisidir. Bu nedenle, ilacı insan sağlığı için değerli ve faydalı kılan, hammaddeden çok geliştirilmesinin ardındaki uzun süreli ve kapsamlı klinik araştırmalardır. Yeni bir ilacın geliştirilmesinin yüksek maliyetinin sebebi de bu araştırmaların çok fazla zaman ve emek istemesidir.

Günümüzde bir ilacın keşfedilmesi ile çeşitli araştırma süreçlerinden geçerek kullanıma sunumu arasında geçen süre 10 - 15 yıl arasında değişmektedir. Bu sürenin çok önemli bir bölümü ise, keşfedilen moleküllerin ve onlardan hazırlanan ilaç şekillerinin etkinliğini ve emniyetini kanıtlamak için yapılan bir seri klinik araştırmalara ayrılmıştır.

Üretilen bir ilacın piyasaya sunulmasından önce onay otoritelerine başvuru aşamasına gelinceye kadar belli bazı aşamaları tamamlamış olması gerekmektedir. Bunlar;

1. Klinik öncesi araştırmalar: Bu aşamada laboratuvar ortamında ve hayvanlar üzerinde deneyler yapılarak hedef endikasyon üzerinde biyolojik etkinliğinin ve hayvanlar üzerinde yüksek emniyet profilinin gösterilmesine çalışılır. Klinik öncesi araştırmaların tamamlanmasından hemen sonra insanlarda ilacın test edilmesi için yeni ilaç izin başvurusu (Investigative New Drug – IND) yapılır.

2. Klinik çalışmalar: IND onayından hemen sonra başlayan bu süreçte çalışmalar fazlar halinde yürütülür.

a. Faz I: Yeni molekülün emniyet profilinin incelenmesi amacıyla 20-100 sağlıklı gönüllü üzerinde metabolizmaya ait bilgiler toplanır.

b. Faz II: Çeşitli dozlarda ilacın etkinlik ve emniyeti, daha büyük hasta gruplarında (100-500 gönüllü hasta) incelenir.

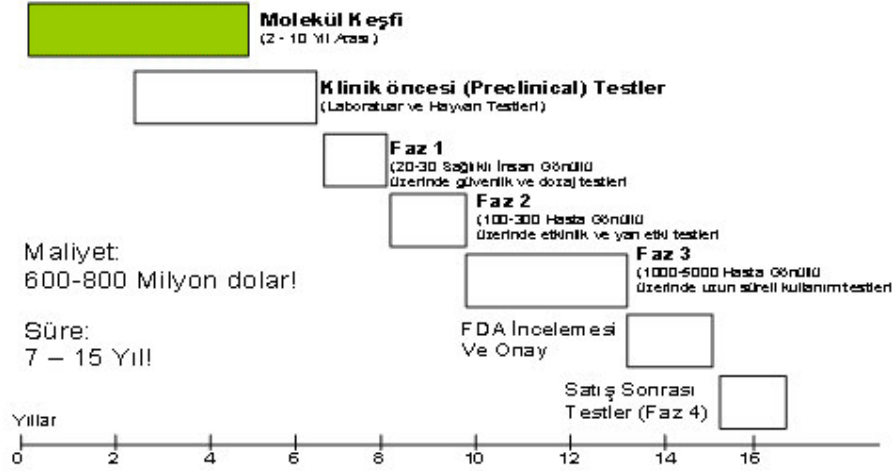
c. Faz III: İlacın etkinlik ve emniyeti ile ilgili incelemeler daha büyük sayıda gönüllü hasta (1000-3000) üzerinde, hastane ve kliniklerde hekimler tarafından devam ettirilir.

3. Klinik çalışmaların tamamlanmasından sonra yeni ilaç başvurusu gerçekleştirilir. Bunun için ilacın tüm bilimsel gelişim aşamaları, etki ve emniyet bilgileri, sayfa sayısı 100.000'i bulabilen dosyalar halinde, ilaç ruhsat otoritesine sunulur.

4. Onay alan ilaçlar kullanıma sunulur ve bu aşamadan sonra da ilacın etkileri izlenerek raporlanır.

ABD ilaç otoritesi olan FDA (Food and Drug Administration), yeni molekülün uzun süreli etkilerini değerlendirmek için pazarlama sonrası (post marketing) bir dizi klinik çalışmayı da

gerekli kılmaktadır ki, buna da Faz IV çalışmaları denilmektedir. Bu aşamalar www.cs.wright.edu/itri/EVENTS/workshop-Spr01-ppts/raymer.ppt adresinden derlenmiş olan Şekil 2.2’de ve Şekil 2.3’te gösterilmektedir.



Şekil 2.2 İlaçta araştırma ve geliştirme süreci

İlaçta araştırma ve geliştirme sürecinin anlatıldığı Şekil 2.2, bu alandaki çalışmaların uzun süreli, yoğun ve yüksek maliyetli olduğunu açıkça göstermektedir. Molekül geliştirme sürecinde, laboratuvarda incelemeye alınan 10.000 molekülden ancak 1 tanesi yeni ilaç etken maddesi olarak kullanıma sunulabilmektedir. Ayrıca geliştirilen her 5 ilaç molekülünden de ancak 1 tanesinin gerekli tüm etkinlik ve emniyet testlerini başarı ile tamamlayarak sağlık hizmetine ilaç olarak sunulabildiği bilinmektedir.

Şekil 2.3’te ilaç tasarımı sürecinde aday molekül sayılarının fazlar arasında geçişi verilmiştir.



Şekil 2.3 İlaç geliştirme sürecinde aday molekül sayılarının ortalama sayıları

Ortalama 10–15 yıllık araştırma ve geliştirme (ARGE) süreci sonucu ortaya çıkan bir ilacın maliyeti, son tahminlere göre ortalama 802 milyon dolardır. Amerika İlaç Araştırmaları ve Üretimi Kurumu (PhRMA)’nın verilerine göre 1999 yılında yeni ilaçların araştırmasına ve geliştirilmesine yalnızca Amerika’da 30 milyar dolar harcanmıştır (Sonka vd., 2002).

2.5 Dünyada İlaç Sanayi

İlaç tasarımındaki yüksek maliyetler sebebi ile ekonomisi güçlü olan ülkelerde ilaç tasarımı yapılabilmektedir. Dünyada ülkeler ilaç sanayisindeki konumlarına göre 4 gruba ayrılmaktadırlar (Baykara vd., 2003):

1.Grup: Yenilikçi ilaç araştırma ve geliştirmeye dayalı çok gelişmiş ilaç endüstrisine sahip ülkeler (ABD, İngiltere, İsviçre, Japonya, Hollanda, Almanya, İsveç, Belçika, Fransa).

2.Grup: Araştırma kapasitesi olan (1961-1990 yılları arasında en az bir yeni molekül keşfetmiş ve piyasaya sunmuş) ülkeler (Arjantin, Avustralya, Avusturya, Çin, Danimarka, Hindistan, İrlanda, İspanya, İsrail, İtalya, Kanada, Kore, Macaristan, Meksika, Portekiz, Slovenya).

3.Grup: Mamul ilaç ve etkin madde üreten ülkeler (Türkiye, 13 ülke ile birlikte bu grupta yer almaktadır).

4.Grup: Sadece mamul ilaç üreten ülkeler (87 ülke bu gruptadır).

Türkiye’nin ilaç sanayisindeki durumunu düzeltmek için, ilaç tasarımı yöntemleri, TÜBİTAK tarafından yayınlanan raporda (bkz. <http://vizyon2023.tubitak.gov.tr/teknolojiongorusu/paneller/raporozet/saglik.pdf>) 2003-2023 yıllarını kapsayan dönemdeki vizyonu içinde öncelikli teknolojik faaliyet konuları içinde yer almaktadır.

2.6 İlaç Tasarımında Makine Öğrenmesi Metotlarının Kullanımı

Bir ilaç firması olan Bristol-Myers Squibb (BMS) aday moleküllerinin %22’sinin toksin özellikler taşıdıklarından dolayı ilaç olamadıklarını ve aday moleküllerin klinik testlerden önce toksin özellikler taşıdıkları belirlenebilirse ilaç başına maliyetlerinin 100 milyon \$ azalacağını bildirmiştir. İlaç geliştiren şirketlere bu toksin özelliklerin geç aşamalarda ortaya çıkışı yıllık 2 milyar \$ zarara sebep olmaktadır [3].

Makine öğrenmesi metotları ile aday moleküllerin, ilaç geliştirme sürecinin çok erken evrelerinde (molekül gerçekten geliştirilmeden önce bile) incelenerek ADMET özelliklerinin

elde edilmesine çalışılmaktadır. İlaç olma olasılığı düşük olan moleküller önceden belirlenerek büyük zaman ve maliyet tasarrufu sağlanmaktadır.

Sonuç olarak ilaç tasarımında kullanılan bilgisayar destekli metotlar yeni moleküller keşfetmemekte, ancak araştırmacılara ellerindeki moleküllerin çeşitli özellikleri (ilaç olma, ADMET vs.) hakkında bilgi vermektedir.

2.7 İlaç Verilerini Elde Etmede Kullanılan Metotlar

Rasyonel ilaç tasarımı yöntemleri (Rational / In silico drug design) tedavi edici özelliği olan yeni moleküllerin bulunmasını hızlandıran metotlardır. Yeni moleküler modeller ve CADD (Computer Aided Drug Design) kullanılarak orijinal bileşiklerin tasarlanması, kombinatorial kimya metotları ve high-throughput screening (HTS) yöntemleri kullanarak yeni kimyasal ilaç adaylarının belirlenmesi, doğal kaynaklı bileşiklerin etkilerinin gene HTS yöntemleri kullanılarak kısa zamanda milyonlarca molekül arasından ayrıştırılabilmesi bugüne kadar kullanılan geleneksel metotlara göre çok daha hızlı ve ucuzdur.

Büyük miktarda molekül yığınları arasından istenilen özelliklerdeki moleküllerin seçilebilmesi için “low-throughput” ya da “high-throughput” adlarındaki iki temel tür metot kullanılır (Hallborn vd., 2002). Low-throughput screening (LTS) metodunda laboratuvar ortamında az sayıda bileşik kimyager ya da biyologlar tarafından el yordamıyla çeşitli test ve tahlillerden geçirilir ve deneylerin sonuçları bilim adamları tarafından yorumlanır.



Şekil 2.4 HTS işleminde kullanılan robot (Hallborn vd., 2002).

Bunun aksine HTS (high-throughput screening) metoduyla çok miktarda bileşiğin tek seferde tamamen otomatik olarak test edilebilmektedir. Bu işlem için kimyagerler her birinin içinde test edilecek bileşiklerin yer aldığı birçok küçük oyuktan oluşan test plakaları hazırlarlar. Test

plakaları Şekil 2.3'teki robot kolunda gözükmektedir. Bu plakalar bir robot kolu tarafından çeşitli test işlemlerine sokulmak için düzenekler arasında gezdirilirler. Test işlemleri sonucunda büyük miktarla veriler elde edildiğinden dolayı bu verilerin işlenebilmesi ve yorumlanabilmesi için bilgisayara ihtiyaç duyulmaktadır.

2.8 İlaç Verilerinin Özellikleri

Bir bileşiğin yapısal formülü bileşiğin fiziksel, kimyasal ve biyolojik özelliklerini yansıtır. Bu varsayım yapısal kimyanın temelini oluşturur ve bu alandaki çalışmalar bir bileşiğin yapısından tüm bu özelliklerin nasıl elde edilebileceği üzerine yoğunlaşmıştır [3]. Nicel yapı-aktivite ilişkileri (Quantitative structure-activity relationships - QSAR's) bileşiklerin kimyasal özellikleriyle biyolojik aktiviteleri arasındaki karmaşık ilişkileri bulmaya çalışan matematiksel modellerdir. Bu modellerin amacı test edilmemiş hatta henüz sentezlenmemiş bileşiklerin biyolojik aktivitelerini tahmin edebilmektir. İlaçların aktivitelerinin yanında başka özelliklerinin de (ADMET) tahmin edilmesi gerekebilmektedir. Bu amaçla yapısı QSAR'a benzeyen nicel yapı-özellik ilişki (Quantitative Structure Property Relationship - QSPR) veri kümeleri de oluşturulmuştur. QSAR ve QSPR problemleri genel olarak Eşitlik 2.1'deki gibi ifade edilmektedirler.

$$y = f(x) \quad (2.1)$$

Eşitlik 2.1'de, X bileşikten elde edilebilen yapısal özellikleri, Y QSAR'de biyolojik aktiviteyi, QSPR'de ise diğer biyolojik etkileri göstermektedir. QSAR/QSPR modelleri F ile gösterilen fonksiyonu bulmayı amaçlamaktadırlar. QSAR/QSPR veri kümeleri bileşiklerin çeşitli özelliklerinden ve laboratuvar çalışmaları sonucu elde edilmiş aktivite/özellik değerlerinden oluşur. Özellikleri bilinen ancak aktivitesi bilinmeyen yeni bir bileşik geldiğinde önceki verilerden yararlanarak aktivite değeri belirlenmeye çalışılır. <http://www.netsci.org/Science/Compchem/feature19.html> adresinden alınan Çizelge 2.1'de çok basit bir QSAR veri kümesi verilmiştir. Çeşitli bileşiklerin hidrofobi katsayıları (hydrophobic substituent constant - π) ile biyolojik aktiviteleri verilmiştir. NHCHO bileşiğinin aktivite değerinin önceki bileşiklerin bilgileri kullanılarak tahmin edilmesi istenmektedir.

Çizelge 2.1 Örnek bir QSAR veri kümesi

Bileşik	π	Log EC ₅₀
H	0.00	1.07
Cl	0.71	0.09
NO ₂	-0.28	0.66
CN	-0.57	1.42
C ₆ H ₅	1.96	-0.62
N(CH ₃) ₂	0.18	0.64
I	1.12	-0.46
NHCHO	-0.98	??

Aktivite değerleri çeşitli sınıflardan oluşabileceği gibi sürekli sayılardan da oluşabilmektedir. Aktivite değerlerinin sınıflardan oluştuğu durumlarda makine öğrenmesi metotlarından sınıflandırma yapanlar, sürekli sayılardan oluştuğunda ise eğri uydurma yapan metotlar kullanılmaktadır. Ayrıca elde edilen özelliklerin hangilerinin aktiviteyi etkilediği ya da etkilemediği sorusuna da özellik seçimi/çıkarımı metotları yanıt vermektedir.

Moleküllerin aktivite değerinin belirlenmesi için çaba ve zaman gerektiren laboratuvar çalışmaları yapılması gerektiğinden ilaç veri tabanlarında genelde az miktarda eğitim verisi yer almaktadır. Moleküllerin yapısal özelliklerinin elde edilmesi ise teknolojinin gelişmesiyle birlikte tamamen otomatik yapılmakta ve bu sayede test verilerinin çok sayıda özellikleri elde edilebilmektedir. Elde edilen moleküllerin özellikleri birbirlerine çok bağımlıdır ve molekülün özellikleriyle biyolojik aktivitesi arasında genelde lineer olmayan bir ilişki vardır. Aşağıda ilaç veri kümelerinin genel özellikleri listelenmiştir.

- Örnek sayısı az
- Özellik sayısı çok
- Özelliklerin birbirleriyle korelasyonu yüksek
- Özelliklerle sonuç arasında lineer olmayan bir ilişkiler var

Bu özellikler sebebiyle ilaç veri tabanları, üzerinde tahmin yapılması zor olan veri tabanları olmuşlardır. Ancak ilaç olamayacak moleküllerin ilaç tasarımının evresinin başlarında tahmin edilmesi çok büyük miktarda zaman ve maliyet tasarrufu sağladığından bu alandaki çalışmalar

hız kazanarak devam etmektedir.

2.9 İlaç Veri Kümelerinin Formatları

Moleküler veri tabanlarının birçok formatı bulunmaktadır. Ancak ilaç uygulamalarında en çok kullanılan 4 tanesi bu bölümde anlatılmıştır. Bugün veri formatları arasında çeşitli dönüşümleri otomatik olarak gerçekleştiren birçok araç mevcuttur. Örnek olarak <http://www.chemaxon.com/marvin/doc/dev/example-sketch3.4.html> adresindeki araç incelenebilir.

2.9.1 SDF Formatı (Structure Data Format)

SDF dosya formatı, Molecular Design Limited (MDL) tarafından moleküllerin özelliklerini ifade etmek için geliştirilmiş ve standartlaşmış bir metin dosya formatıdır. Bir SDF dosyasında, molekülün adı, özellikleri, atomlarının koordinatları, atomlardaki bağların türleri, hangi atomlar arasında yer aldıkları ve bu moleküle ilgili istenilen türdeki bilgiler saklanır. Bir metin dosya formatı olduğundan kelime-işlemci programlarıyla bile açılıp işlenebilmektedir. Bir dosya içinde istenildiği kadar çok molekülün bilgisi saklanabilir. <http://www.mdli.com/downloads/public/ctfile/ctfile.jsp> adresinden bu dosya formatı hakkında geniş bilgiye erişilebilmektedir. Şekil 2.5'te örnek bir SDF dosyası gösterilmiştir.

```

1 benzene
2 ACD/Labs0812062058

3 6 6 0 0 0 0 0 0 0 0 1 V2000
4 1.9050 -0.7932 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
5 1.9050 -2.1232 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
6 0.7531 -0.1282 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
7 0.7531 -2.7882 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
8 -0.3987 -0.7932 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
9 -0.3987 -2.1232 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
10 2 1 1 0 0 0 0
11 3 1 2 0 0 0 0
12 4 2 2 0 0 0 0
13 5 3 1 0 0 0 0
14 6 4 1 0 0 0 0
15 6 5 2 0 0 0 0
M END
$$$$

```

Şekil 2.5 Örnek bir SDF dosyası

Şekil 2.5'te orijinal dosya formatına açıklamalara yardımcı olması için her satırın numarası eklenmiştir. Dosyada 1. satırda başlık, 2. satırda yorum, 3. satırda genel bilgiler (6 atom, 6 bağ vs.), 4. ve 9. satırlar arasında her atomun x, y ve z boyutlarındaki yeri, atomun kendisi ve

ekstra bilgiler bulunmaktadır. 10. ve 15. satırlar arasında bağlantı bilgileri (kaçıncı atomla, kaçıncı atom, hangi tür bağ ile) ve ekstra bilgiler bulunmaktadır.

2.9.2 SMILES Formatı

Simplified Molecule Input Line Entry System (SMILES), moleküllerin 2 boyutlu uzaydaki yapılarını tek bir karakter dizisinde gösterimini sağlayan bir formattır. Genelde QSAR uygulamalarında kimyasal özelliklerin ve biyolojik aktivitelerin tahmininde kullanılır. <http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html> adresinden alınan Çizelge 2.2’de çeşitli moleküllerin Smiles formatında yazılışları gösterilmiştir.

Çizelge 2.2 Çeşitli moleküllerin SMILES formatında gösterimleri

SMILES	Molekül ismi	SMILES	Molekül ismi
CC	Ethane	[OH3+]	hydronium ion
O=C=O	carbon dioxide	[2H]O[2H]	deuterium oxide
C#N	hydrogen cyanide	[235U]	uranium-235
CCN(CC)CC	triethylamine	F/C=C/F	E-difluoroethene
CC(=O)O	acetic acid	F/C=C\F	Z-difluoroethene
C1CCCCC1	cyclohexane	N[C@@H](C)C(=O)O	L-alanine
C1ccccc1	Benzene	N[C@H](C)C(=O)O	D-alanine

2.9.3 First Order Logic Formatı

Moleküllere ait bilgiler First order logic’e (yüklem mantığı) göre tutulmaktadır. Yüklem mantığında sabitler, değişkenler ve yüklem yer alır (Srinivasan vd., 1999). Aşağıda bu türde tutulan bir veri tabanından örnekler ve açıklamaları verilmiştir.

atm(m1,a1,o,2,3.43,-3.12,0.05) ifadesi “m1” molekülünün adına a1 denilen, oksijen olan, sp2 hybridized olan ve 3 boyutlu uzayda [3.43,-3.12,0.05] koordinatlarında yer alan bir atoma sahip olduğunu belirtmektedir.

Bond(m1,a2,a3,2) ifadesi m1 molekülünün a2 ve a3 atomları arasında bir bağ olduğunu ve bu bağın bir çift bağ olduğunu belirtmektedir.

Active(X):- hdonor(X,A), hacc(X,B), zincsite(X,C), dist(X,A,B,3.0,1.0), dist(X,A,C,4.0,1.0), dist(X,B,C,5.0,1.0) ifadesi eğer X molekülünün A isminde bir hidrojen vericisi, B isminde bir hidrojen alıcısı, C isminde bir zincsite’i varsa ve A-B mesafesi 3.0 +/- 1.0 Angström, A-C mesafesi 4.0 +/- 1.0 Angström, B-C mesafesi 5.0 +/- 1.0 Angström ise X molekülünün aktif olduğunu belirtmektedir.

Verilerin bu tarz yapılarda tutulduğu veri tabanlarında verilerin ve sonuçların anlaşılıp yorumlanması daha kolaydır.

2.9.4 Sayısal Formatlar

Klasik veri kümelerinin tutulduğu formattır. Bilgiler satırlarında özellikler, sütunlarında örneklerin yer aldığı bir matriste yer alırlar (Bkz. Çizelge 2.1). Klasik makine öğrenmesi algoritmaları için bilgilerin bu formatta tutulması gerekmektedir. Eğer eldeki veriler diğer formatlarda ise çeşitli metotlar ve yazılımlarla bu formata çevrilmektedirler. Bu tezde de, moleküllerin ifade edilmesinde bu yaygın format kullanılmıştır.

2.10 Moleküllerin Sayısal Verilere Dönüştürülmesi

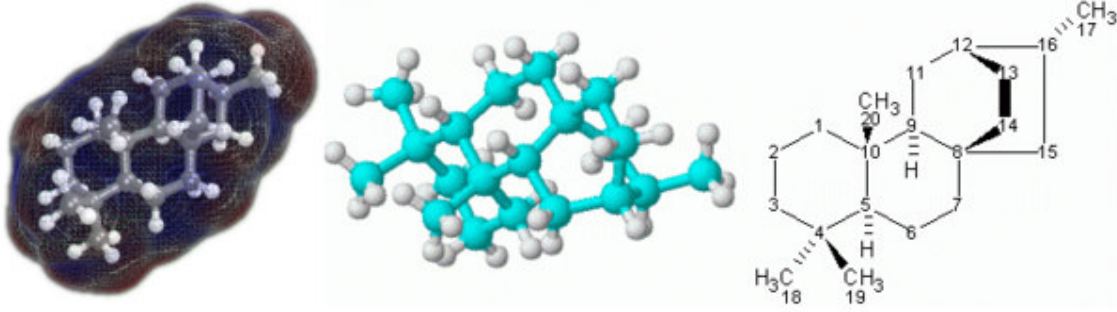
Moleküllerin makine öğrenmesi metotlarıyla sınıflandırılabilmesi için ya da belli özelliklerinin tahmini için genellikle sayılarla ifade edilirler. Moleküllerin sayısallaştırılması için birçok program geliştirilmiştir. Bunlar arasında Molecular Networks (www.mol-net.de) firmasının Adriana Code yazılımı (bu tezde kullanılan yazılım), Chemical Computing Group (www.chemcomp.com) firmasının MOE yazılımı ve Openeye Scientific Software (www.eyesopen.com) firmasının Filter yazılımı sayılabilir. Bu tür programlar SDF formatındaki verileri alıp her molekül için eşit sayıda özellikleri ifade eden sayılara dönüştürmektedirler. Programlar birbirlerinden hesapladıkları molekül özelliklerine göre ayrılmaktadırlar. Moleküllerin SDF formatlarından hesaplanan özellikleri 3 grupta toplanmaktadır.

Genel Özellikler: Molekül Ağırlığı, Erime noktası, Kaynama noktası

2D Özellikler: Molekülü oluşturan atomların birbirlerine bağılılıkları, bağ türleri, belirli fonksiyonel grupların molekülde bulunma sayıları, halka sayıları

3D Özellikler: Yüzey özellikleri

Şekil 2.6'da Bir molekülün 2D ve 3D özelliklerinin hesaplanmasında kullanılan gösterimleri verilmiştir.



Şekil 2.6 Terpenoid molekülünün 3D yapısı sol ve ortada, 2D yapısı sağda [4]

Şekil 2.7’de moleküllerin sayısal özelliklerinin çıkarılmasının özeti verilmiştir.

MOE

Adriana vs.

Molekül	Özellik1	Özellik2	...	ÖzellikN	Sınıf
NH ₃	23	5			1
H ₂ O	34	6.7			1
NCI	45	8.9			2
...					
C ₄ H	4	67			N

Şekil 2.7 Moleküllerden sayısal özelliklerin çıkarılması

2.11 İlaç Moleküllerini İfade Etmede Kullanılan Özellikler

Teknolojinin gelişmesiyle bileşiklere ait elde edilebilen özelliklerin türleri ve sayıları da her geçen gün artmaktadır. Basit bir ilaç veri tabanında bile bir bileşiğe ait yüzlerce özellik yer almaktadır. Bu sayının 100.000'lere ulaştığı veri tabanları da mevcuttur. Çizelge 2.3'te bu tezde “sdf” formatında olan ilaç verilerinin sayısal özelliklere çevrilmesi için kullanılan ADRIANE.CODE yazılımının ürettiği 1142 adet özellik verilmiştir. Özelliklerin ayrıntılarına ve nasıl elde edildiklerine <http://www.molecular-networks.com/software/adrianacode> adresinden erişilebilir. Bu yazılım sayesinde “sdf” formatındaki her molekülün eşit sayıda (1142 adet) özelliği çıkarılmış ve moleküller makine öğrenmesi algoritmalarıyla işlenmeye uygun hale getirilmiştir.

Çizelge 2.3 Tezde kullanılan molekül özellikleri (toplam 1142 adet)

Özellik ismi	Özellik Sayısı	Özellik Grubu	Özellik Tanımı
Weight	1	Global molekül özellikleri	Molekül Ağırlığı
HAcc	1		Toplam nitrojen ve oksijen atomlarının sayısı
Hdon	1		Toplam O-H ve N-H ikililerinin sayısı
MPolariz	1		Molekülün ortalama kutuplanabilirliği
TPSA	1		Topolojik kutupsal yüzey alanı
Dipole	1		Molekülün çift-kutup momenti
ACorr_Ident	11	2D oto-korelasyon	Idendity
ACorr_SigChg	11		σ yük
ACorr_PiChg	11		Π yük
ACorr_TotChg	11		Toplam yük
ACorr_SigEN	11		σ Elektronegatiflik
ACorr_PiEN	11		Π Elektronegatiflik
ACorr_LpEN	11		Lone pair electronegativity
ACorr_APolariz	11		Efektif kutuplanabilirlik
RDF_Ident	128	3D özellikler	Idendity
RDF_SigChg	128		σ yük
RDF_PiChg	128		Π yük
RDF_TotChg	128		Toplam yük
RDF_SigEN	128		σ Elektronegatiflik
RDF_PiEN	128		Π Elektronegatiflik
RDF_LpEN	128		Lone pair electronegativity
RDF_APolariz	128		Efektif kutuplanabilirlik
RDFSurf_HBP	12	Yüzey özellikleri	Hidrojen bağlama potansiyeli
RDFSurf_ESP	12		Molekülün elektrostatik potansiyeli

2.12 İlaç Tasarımında Makine Öğrenmesi Kullanılan Önceki Çalışmalar

Bu bölümde ilaç tasarımı konusunda yapılmış olan çalışmalardan incelenmiş olanlar hakkında kısa bir özet verilecektir. Mümkün olduğunca birbirinden farklı konulara, yöntemlere yönelen çalışmalar seçilmeye çalışılmıştır.

“Molecular Database Mining using Self-Organizing Maps for the Design of Novel Pharmaceuticals,” adlı çalışmada (Arciniegas vd., 2000) SOM kullanılarak QSAR verilerinde özellik seçimi yapılmıştır. Özellikler seçilirken önce tüm özellikler için ayrı ayrı SOM haritaları oluşturulmuştur. Haritalardan birbirine kümeleme, dağılım ve şekil olarak benzeyen özelliklerden molekülün aktivitesiyle korelasyonu en yüksek olan seçilerek diğerleri elenmiştir. Ancak sonuçlar göstermiştir ki en yüksek korelasyonlu özellikler en iyi özellikler değildir. Bu durum QSAR verilerinin genel bir özelliği olan moleküllerin özellikleriyle biyolojik aktiviteleri arasındaki ilişkinin lineer olmamasını desteklemiştir.

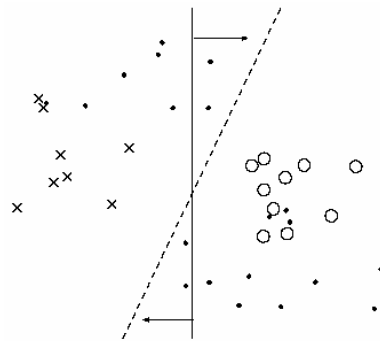
“StripMining for Molecules” adlı çalışmada (Embrechts vd., 2002) duyarlılık analizi ile özellik seçimi (her bir özellik için diğer tüm özellik değerlerini sabit tutup, sadece bir tanesinin değişiminin sınıflandırıcı çıkışı üzerine etkisinin incelenerek, özelliklerden çıkışa etkisinin en büyük olanların seçildiği metot) için yapay sinir ağları, destek vektör makineleri ve parçalı en küçük kareler metotlarının kullanımı anlatılmaktadır. Çalışmanın sonucunda, özellik seçimi metotlarından olan temel bileşen analizinin QSAR problemlerinde örnek sayısının az, buna karşılık özellik sayısının çok olmasından dolayı etkili olamadığı ve duyarlılık analizinin daha iyi sonuç verdiği belirtilmiştir.

“Empirical Evaluation of Ensemble Feature Subset Selection Methods for Learning from a High-Dimensional Database in Drug Design” adlı çalışmada (Mamitsuka, 2003) KDD 2001 yarışmasında kullanılan Thrombin veri seti üzerinde (2543 örnek * 139531 özellik) 4 farklı özellik seçimi metodu denenmiş ve sonuçlar iki sınıflandırıcı (C4.5 ve SVM) tarafından sınıflandırılmıştır. Özellik seçimi metotları doğru özellikleri seçebilme, işlem zamanı ve gürültülü veri ile çalışabilme ölçütlerine göre karşılaştırılmışlardır.

“Feature Selection for In-silico Drug design Using Genetic Algorithms and Neural Networks” adlı çalışmada (Ozdemir vd., 2001), HIV veri kümesi üzerinde duyarlılık analiziyle özellik seçimi yapılmış, yapay sinir ağları ve genetik algoritmaların karşılaştırılması verilmiştir. QSAR çalışmalarında özellik seçiminin önemini, yazarlar özellik sayısının çokluğuna, az miktarda örnek oluşuna ve özelliklerin birbirlerine çok bağımlı oluşuna bağlamışlardır. Seçilen özelliklerin birbiriyle mümkün olduğu kadar az, sonuçla mümkün olduğu kadar fazla ilişkili olması gerektiği belirtilmiştir. Genetik algoritma tabanlı özellik seçiminde kromozomlarda özelliklerin numaraları yer almaktadır. Çeşitli genetik operatörler yardımıyla optimum bir alt özellik altkümesi bulunmaya çalışılmıştır. Kromozomların uygunluk(fitness) fonksiyonu olarak özelliklerin birbiriyle olan korelasyonunu cezalandıran, çıkışla olan korelasyonunu ise ödüllendiren bir fonksiyon seçilmiştir. Ancak genetik algoritmaların

eğitiminin çok uzun zaman alması metodun çok yüksek boyutlu veriler üzerinde uygulanabilirliğini pratik açıdan zorlaştırmaktadır.

“Feature selection and transduction for prediction of molecular bioactivity for drug design” adlı çalışmada (Weston vd., 2003) Knowledge Discovery and Data Mining Cup 2001 yarışmasında kullanılan Thrombin veri seti üzerinde bileşiklerin aktif olup olmadıkları belirlenmeye çalışılmıştır. Veri setinde 1909 eğitim verisi, 634 test verisinin 139,351 adet ikili (0/1) özellikleri bulunmaktadır. Yazarların ifadesine göre veri setinin doğru sınıflandırılabilmesinde 3 temel zorluk bulunmaktadır. Birincisi çok az miktarda pozitif örneğin bulunması(% 2.2) ki bu sınıfların oldukça farklı olasılıklara sahip olması demektir. İkincisi verinin çok büyük boyuta sahip olmasıdır ki bu da eğitim işleminden önce bir ön özellik seçimini zorunlu kılmaktadır. Üçüncüsü ilaç tasarımı veritabanlarının çoğunda yer alan (ilaç tasarımı döngüsünün sebep olduğu) eğitim verisiyle test verilerinin aynı dağılımdan gelmeyişiştir. Yazarlar özellik seçimi problemini çözebilmek için her bir özelliğin dengesiz korelasyon skorunu (unbalanced correlation score) ve entropisini hesaplayıp 1’den 40’a kadar değişen sayılarda özellik alt kümeleri elde etmişler ve sınıflandırma algoritmalarında [en yakın K komşu (kNN), destek vektör makineleri (SVM), karar ağaçları(C4.5)] kullanmışlardır. Eğitim ve test verisinin aynı dağılımdan gelmeme problemine karşı ise uyum sağlayan (transductive) sınıflandırıcılar kullanmışlardır. Transductive sınıflandırıcılar eğitim işleminde hem eğitim verisini hem de test verisini(sınıfları hariç) kullanırlar. Şekil 2.8’de iki sınıf için transductive bir yaklaşım gösterilmektedir. Şekildeki çarpılar ve yuvarlaklar etiketleri belli olan eğitim verilerini; noktalar ise etiketleri belli olmayan test verilerini göstermektedir. Sadece eğitim verilerine göre belirlenen ayırıcı fonksiyon yerine etiketsiz test örneklerine göre de belirlenen ayırıcı fonksiyon transductive sınıflandırıcılarda kullanılmaktadır.



Şekil 2.8 Sadece eğitim verilerine göre belirlenen ayırıcı(düz çizgi) – test örneklerini de kullanan *transductive* ayırıcı (kesikli çizgi)

“A Soft Computing Approach for the Design of Novel Pharmaceuticals” adlı çalışmada (Kewley vd., 1999), eldeki moleküllerin Cholecystokinine molekülüne bağlanma aktiviteleri yapay sinir ağlarıyla tahmin edilmeye çalışılmıştır. Duyarlılık analizi yapılarak tüm özellikler yerine bir alt kümesi üzerinde tahminler gerçekleştirilmiştir. Duyarlılık analizi yapılırken önce eğitim setindeki tüm özelliklerle yapay sinir ağı eğitilmiş, daha sonra her seferinde sadece bir özelliğin değeri değiştirilerek bu değişimin ağı çıkışına etkisi araştırılmıştır. Küçük değişimleri bile ağı çıkışında büyük etkilere sebep olan özellikler (duyarlılığı fazla olan) seçilerek yeni özellikler kümesini oluşturmuştur.

“Particle Swarm Optimization and Neural Network Application for QSAR” adlı çalışmada (Wang vd., 2004) aktivitelerin tahmin edildiği dört farklı QSAR veri seti kullanılmıştır. Veriler yapay sinir ağlarıyla modellenmiş, özellik seçiminde “ikili parçacık sürü optimizasyonu” tekniği kullanılırken, yapay sinir ağlarının eğitiminde geriye yayılım (back propagation) algoritması ve “parçacık sürü optimizasyonu” algoritmaları karşılaştırılmıştır.

“Data to Knowledge in Pharmaceutical Research” adlı çalışmada (DeWitt vd., 2004) LTS verileri eğitim, HTS verileri ise test kümesi olarak kullanılmıştır. LTS verileri laboratuvar ortamında kimyagerler tarafından elle yapılan testler sonucunda elde edilen az sayıda verilerken HTS verileri tamamen otomatik olarak elde edilen çok sayıda verilerdir. Yine incelenen bileşiklerin aktiviteleri tahmin edilmeye çalışılmıştır. Tahmin metodu olarak lojistik regresyon analizi kullanılmıştır.

“Predicting Biochemical Interactions - Human P450 2D6 Enzyme Inhibition” adlı çalışmada (Langdon vd., 2003) insan vücudunun önemli enzimlerinden biri olan P450 2D6 ile ilaç aday moleküllerinin etkileşimi tahmin edilmeye çalışılmıştır. Problem hem regresyon hem de etkileşim aktiviteleri ayrıklaştırılarak sınıflandırma olarak ele alınmıştır. Çalışmada birçok makine öğrenmesi metodu denenmiş ve en başarılı olan genetik algoritmalar olmuştur.

“Pharmacophore Discovery using the Inductive Logic Programming System PROGOL” (Finn vd., 1998), “A new approach to pharmacophore mapping and QSAR analysis using Inductive Logic Programming” (Geneste vd., 2002) adlı çalışmalarda potansiyel ilaç moleküllerinin belirlenmesi amaçlanmıştır. Kullanılan veri tabanında moleküllerin özellikleri yüklem mantığı yapısında tutulmuştur. Bu tarz veri tabanlarının anlaşılabilirliği ve sonuçlarının yorumlanabilmesi ilişkiler sözcüklerle ifade edildiğinden dolayı diğer veri tabanı türlerine göre daha kolaydır.

Ayrıca <http://org.chem.msu.su/people/baskin/neurchem.html> adresinde QSAR çalışmalarında

Yapay Sinir Ağlarının kullanımına ait onlarca çalışma yer almaktadır.

2.13 Sonuç

Görüldüğü gibi ilaç tasarımı problemlerinde makine öğrenmesinin neredeyse tüm alanlarına ihtiyaç duyulmakta ve kullanılmaktadır. Bunun yanında ilaç veri kümeleri mevcut algoritmalarla istenilen ölçülerde modellenememektedir. İlaç verileri üzerinde de başarılı bir şekilde çalışan yeni algoritmalara ihtiyaç vardır. Bu nedenlerle, tezde makine öğrenmesinin birçok alanında yeni algoritmalar öneren bir çalışma gerçekleştirilmiştir. 3. bölümden itibaren bu çalışmalar anlatılmıştır.

3. SINIFLANDIRMA

Bu bölümde öncelikle sınıflandırma alanında kullanılan terimler ve algoritmaların anlatımında kullanılan semboller verilmiştir. Daha sonra popüler sınıflandırma metotlarına değinilmiştir. Bu tezde geliştirilen sınıflandırma algoritmaları ve öncekilerle karşılaştırılmaları bir sonraki bölümü oluşturmaktadır. Son olarak da, elde edilen sonuçların yorumlanması ve gelecek çalışmalar için öneriler verilmiştir.

3.1 Terimler ve Kullanılan Semboller

Sınıflandırma: Verilerden, sınırlı sayıdaki sınıflarına tanımlı bir fonksiyonun kestirilmesi işlemidir. Bu fonksiyon kestirimi için literatürde çeşitli algoritmalar geliştirilmiştir.

Veri kümesi: p boyutlu N adet örnek ve örneklerin sınıflarından oluşur.

Veri kümesindeki örneklerin her biri eşit sayıda özellik içerir. Diğer bir ifadeyle örneklerin her biri p boyutlu uzayda bir noktadır.

Eğitim kümesi: Veri kümesinin, sınıflandırma fonksiyonunun kestirilmesi, diğer bir ifadeyle sınıflandırma modelinin oluşturulması işleminde kullanılan bir alt kümesidir.

Test kümesi: Veri kümesinin sınıflandırma fonksiyonunun kestirilmesinde kullanılmayan kısmıdır. Diğer bir deyişle veri kümesinin tamamından eğitim kümesinin çıkarılmış halidir. Sınıflandırma fonksiyonunun performansının ölçümünde kullanılır. Performans, fonksiyonun daha önce karşılaşmadığı örnekler üzerinde ölçülür.

Sınıflandırma performansı: Bir sınıflandırma algoritmasının performansı, test kümesindeki verilerin kestirilen fonksiyon kullanılarak elde edilen sınıflarının, gerçek sınıflarına uyumluluğu ile ölçülür (Eşitlik 3.1).

$$performans(\%) = (1/N) * \sum_{i=1}^N esit(T_i, C_i),$$

$$esit(A, B) = \begin{cases} 1 & \text{A} = \text{B} \\ 0 & \text{A} \neq \text{B} \end{cases} \quad (3.1)$$

Eşitlik 3.1'deki T_i , i .örneğin tahmin edilen sınıfını, C_i gerçek sınıfını, N toplam örnek sayısını göstermektedir.

Çapraz geçirme: Algoritmaların performans tahminlerindeki rasgeleliği azaltmak amacıyla kullanılan bir metottur. Veri kümesi K eşit elemanlı alt kümeye ayrılır. Her seferinde bir alt küme dışarıda bırakılarak, diğer K-1 adet alt küme ile sınıflandırma fonksiyonu kestirilir, dışarıda bırakılan alt küme üzerindeki performansı bulunur. Bu işlem K kere tekrar edilir. Bu sayede her alt kümenin K-1 kere eğitim kümesinde, 1 kere test kümesinde yer alması sağlanır. Algoritmanın performansı bulunan K adet performansın ortalaması alınarak bulunur. Bu işleme ait sözde kod aşağıda verilmiştir.

```
i=1:K
```

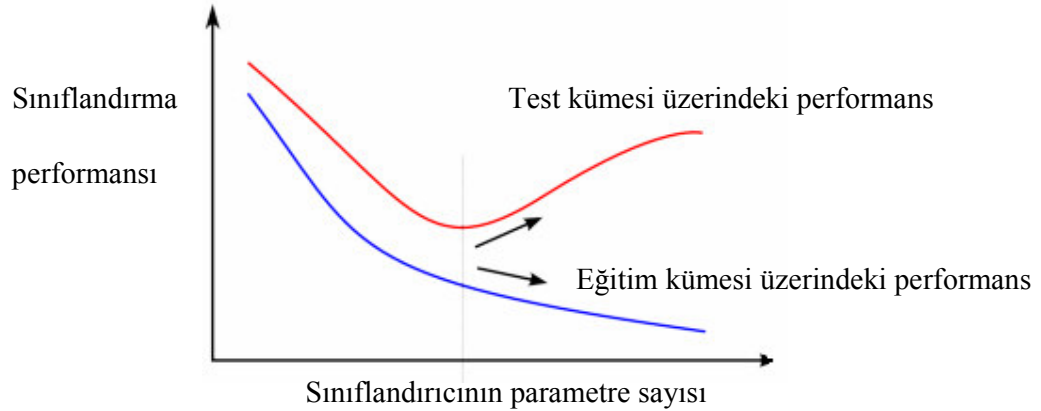
```
    [eğitim_kümesi, test_kümesi]=eğitim_test_olustur(veri_kümesi,i)
```

```
    model=sınıflandırıcı(eğitim_kümesi)
```

```
    performans(i)=performans_hesapla(model, test_kümesi)
```

```
ortalama_performans=1/N toplam(performans)
```

Aşırı eğitim: bir sınıflandırıcının eğitim kümesi üzerindeki performansı yüksek, ancak test kümesi üzerindeki (eğitim sürecinde görmediği örnekler üzerindeki) performansı düşükse *aşırı eğitilmiş* olarak ifade edilir. Böyle sınıflandırıcıların genelleştirme kabiliyetleri zayıftır. Sınıflandırma modelinin parametre sayısının gereğinden çok olması aşırı eğitime sebep olan ana faktördür. Şekil 3.1’de bir sınıflandırıcının parametre sayısına bağlı olarak eğitim ve test kümelerindeki performansı verilmiştir.



Şekil 3.1 Bir sınıflandırıcının parametre sayısının eğitim ve test kümeleri üzerindeki performanslarına etkisi

Şekil 3.1’de görüldüğü gibi parametre sayısının optimum bir değerden sonra artması durumunda sınıflandırıcının eğitim kümesindeki performansı artmakta (örnekleri ezberliyor) iken test kümesindeki performansı düşmektedir. Sistemin genelleştirme kabiliyetinin yüksek

olabilmesi için bu optimum noktanın bulunması gerekmektedir.

Rasgele başarı: Bir veri kümesinin bütün örneklerinin, frekansı en yüksek sınıfla etiketlendiğindeki doğruluk oranıdır. Diğer bir deyişle, bir sınıflandırma algoritmasının başarılı olduğunu söyleyebilmek için göstermesi gereken minimum performanstır.

Semboller:

$$A = \{X_i, C_i\}_{i=1}^N, X_i \in \mathbb{R}^p$$

T_i : i. örneğin tahmin edilen sınıfı

C: Gerçek sınıflar

C_i : i. örneğin gerçek sınıfı

T: Tahmin edilen sınıflar

N : veri kümesindeki örnek sayısı

p : örneklerin boyut sayısı

3.2 Önceki Çalışmalar

Tezin sınıflandırma bölümünde geliştirilen algoritmaların tamamı karar ağacı türünden algoritmalar. Bu seçimin temel sebebi, karar ağacı algoritmalarının yüksek performanslarının yanı sıra eğitim sürecinde oluşturdıkları yapı sayesinde verdikleri kararların son kullanıcılar tarafından anlaşılabilir kurallara dönüştürülebilir olmasıdır.

Geliştirilen algoritmaların karar ağacı algoritmaları olmaları sebebiyle performansları da diğer karar ağacı algoritmalarıyla karşılaştırılmıştır. Bu nedenle bu bölümde önceden geliştirilmiş tüm sınıflandırma algoritmaları yerine sadece popüler karar ağacı algoritmaları anlatılmıştır.

3.2.1 Karar Ağaçları

Karar ağaçları içlerinde verilerin hangi dala yönlendirileceğini belirleyen karar düğümlerinden ve bu dalların uçlarında gelen verinin hangi sınıfta olduğunu söyleyen sınıf etiketlerini içeren yapraklardan oluşan hiyerarşik bir yapıdır. Bir veri karar ağacıyla sınıflandırılmak istediğinde en tepedeki kök karar düğümünden başlanır ve bir yaprağa gelinceye kadar karar düğümlerindeki yönlendirmelere göre dallarda ilerler. Yaprğa gelindiğinde ise verinin sınıfı yaprağın temsil ettiği sınıf olarak belirlenir.

Geliştirilen ilk karar ağaçlarından biri olan ID3'ün algoritması [5] Çizelge 3.1'de verilmiştir.

Çizelge 3.1 ID3 algoritması

Giriş: Eğitim seti	
Çıkış: Karar ağacı	
Başlangıç:	
Veri kümesindeki tüm örnekler aynı sınıftan mı?	
Evet	Hayır
Düğümü Etiketle Çık	Verilerin tüm özellikleri için entropi değerlerini hesapla ve en küçük entropiye sahip özellik için ağaca bir düğüm ekle
	Düğümü oluşturan özelliğin her bir değeri için düğümden bir dal oluştur
	Verileri düğümü oluşturan özelliklerindeki değerlere göre dallara ata
	Rekürsif olarak her bir dal için algoritmanın başına dön

Algoritmada yer alan düğüm etiketleme işlemi, düğümden bulunan örneklerin sınıfıyla o yaprak düğümün etiketlenmesidir. ID3 algoritması, örneklerin özelliklerinin isim (nominal) olduğu durumlarda kullanılmaktadır. Eğer özellikler sayısal ise ve ID3 kullanılmak isteniyorsa bir ön işlem olarak ayrıklaştırma yapılmalıdır. Bununla birlikte üzerinde çalışılan veri kümelerinin büyük bir çoğunluğunda özellikler sayısaldir ve bu nedenle geliştirilen karar ağacı algoritmalarının hemen hemen hepsi sayısal verilerle de çalışabilecek şekilde tasarlanmıştır. Tezin bundan sonraki bölümlerinde bu tarz algoritmalarından bahsedilmektedir.

3.2.2 Karar Ağaçlarında Budama

Karar ağaçlarında oluşturulan modelin karmaşıklığını azaltmak ve genelleştirebilme kabiliyetini yükseltmek için ağaçlar genellikle budanmaktadır. Budama işlemi ağacın aşırı eğitilmiş ya da gürültüleri öğrenmiş olma ihtimaline karşı yapılmaktadır. Bu işlem ağaç oluşturulurken belirli bir bölgede bir eşik değerinden daha az örnek varsa o bölge bir daha bölünmeyerek ya da ağaç oluşturulduktan sonra yapılmaktadır. Sonradan yapılan budama işlemi için eğitim verisinden bir geçerleme kümesi (validation set) ayrılır. Geri kalan eğitim kümesi kullanılarak ağaç oluşturulur ve geçerleme kümesi üzerindeki performansı ölçülür. Daha sonra en yukarıdaki düğümlerden başlanarak ağaçtaki her bir düğüm için bu düğümün olmadığı durumdaki ağacın geçerleme kümesindeki performansı ölçülür. Eğer daha iyi ise bu düğüm altındaki tüm dallarla birlikte budanır ve içindeki örnek dağılımlarına göre bir sınıf etiketi atanarak bir yaprağa dönüştürülür. Karar ağaçları budandıklarında yaprak düğümlerdeki verilerin saflığı (tek sınıftan örnekler içermeye) azalır buna karşın genelleştirme

kabiliyetleri yükselir.

3.2.3 Tek ve Çok Değişkenli Karar Ağaçları

Karar düğümlerinde eğer ağaç tek değişkenli ise tek bir özelliğin adı ve bir eşik değeri yer alır. O düğüme gelen verinin hangi dala gideceğine verinin o düğümdeki özelliğinin eşik değerinden büyük ya da küçük olmasına göre karar verilir. Eşitlik 3.2’de tek değişkenli bir karar düğümüne gelen örneğin eşik değerine göre nasıl alt dallara yönlendirildiği verilmiştir (Alpaydın, 2004).

$$f(x) : x_j \geq w_0 \quad (3.2)$$

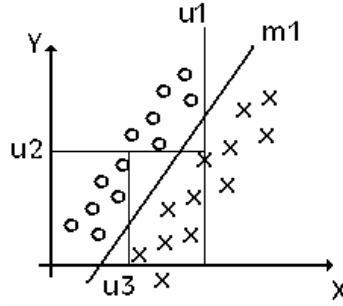
$f(x)$ fonksiyonunun sonucu 1 ya da 0’dır. Örnek, bu fonksiyonda yer alan özellik indisi (j) ve eşik değeri (w_0) kullanılarak alt dallara yönlendirilir. Örneğin j . özelliği, eşik değerinden büyük ise $f(x)$, 1 değerini, küçük ise 0 değerini alacak ve bu sonuçlara göre iki alt daldan birine gönderilecektir.

Çok değişkenli ağaçların karar düğümlerinde ise bir hiper düzlem yer alır. Çok değişkenli bir düğüme gelen bir verinin hangi dala yönlendirileceğine hiper düzlemin hangi tarafında yer aldığına göre karar verilir. Eşitlik 3.3’te (Alpaydın, 2004) bir karar düğümüne gelen p adet özelliği olan bir örneğin (X), hiper düzlem parametreleri ($w_{1..p+1}$) kullanılarak nasıl alt dallara yönlendirildiği verilmiştir.

$$f(x) : w_m^T x + w_{m0} = \sum_{j=1}^p w_{mj} x_j + w_{m0} > 0 \quad (3.3)$$

$f(x)$ fonksiyonunun sonucu 1 ya da 0’dır. Örnek, hiper düzlemin üstündeki bölgede ise $f(x)$, 1 değerini, altındaki bölgede ise 0 değerini alacaktır. Örnek, bu fonksiyonda yer alan hiper düzlemin ($w_{1..p+1}$) hangi tarafında yer aldığına göre 2 daldan birine yönlendirilir.

Şekil 3.2’de aynı veri kümesi için oluşturulmuş, tek değişkenli ve çok değişkenli karar ağaçları verilmiştir.



Şekil 3.2 Aynı veri kümesinde tek ve çok değişkenli karar ağaçları

Şekil 3.2’de görüldüğü gibi veri kümesinin sınıflandırılmasında tek değişkenli karar ağacı kullanıldığında 3 düğüm ($u1$, $u2$, $u3$), çok değişkenli karar. Ağacı kullanıldığında ise tek bir düğüm ($m1$) yeterli olmaktadır. Çok değişkenli karar ağaçlarının bu avantajına karşın, verdikleri kararların anlaşılabilirliği daha azdır.

3.2.4 Karar Ağaçlarının Oluşturulmasında Kullanılan Yöntemler

Tek değişkenli karar ağaçları oluşturulurken mümkün tüm özellikleri ve eşik değerlerini deneyip eğitim setine en uygununu seçmek mümkündür. Ancak çok değişkenli karar ağaçlarında olasılıklar çok fazla olduğundan mümkün her olasılık denenemez. Bu sebeple en optimum parametreleri aramak yerine yeterince iyi parametreleri bulmak için birçok metot geliştirilmiştir.

Yıldız ve Alpaydın (Yıldız vd., 2005) çalışmalarında en çok kullanılan karar ağacı metotlarının kısa bir özetini vermiştir.

(1) Friedman’ın çalışmasında düğümlerdeki parametrelerin belirlenmesinde Lineer Ayırdıcı Analiz (LDA) kullanılmıştır. Ağacın her düğümünde veriler iki dala ayrılırlar. Sınıf sayısı 2’den fazla olduğunda, problem her bir sınıfı diğerlerinden ayıran sınıf sayısı tane alt probleme bölünmektedir ve her biri için ayrı karar ağacı oluşturulmaktadır. Oluşturulan karar ağaçlarının sonuçları oylama metodu ile birleştirilmektedir.

(2) Breiman tarafından önerilen Sınıflandırma ve Regresyon Ağaçları (CART) algoritmasında hiper düzlem parametreleri bulunurken her adımda parametrelerden biri dışında hepsi sabit tutulup, dışarıda kalan parametrenin saflık kriterine göre alacağı en iyi değer bulunmaktadır. Bu işlem hiper düzlemin tüm parametreleri için tekrar edilmektedir. Bu metotta algoritmanın yerel bir minimuma takılma ihtimali vardır.

(3) Hızlı Bir Sınıflandırma Ağacı Algoritması - Fast Algorithm for Classification Trees

(FACT) çalışmasında K farklı sınıftan örnekler içeren bir düğümde her bir sınıfı diğer tüm sınıflardan ayıran K adet LDA ile bulunmuş olan karar sınırı (hiper düzlem) bulunmaktadır.

(4) Yapay Sinir Ağı Ağaçları (Neural Trees), her bir düğümdeki karar sınırını bulmak için yapay sinir ağlarını kullanırlar. Yapay sinir ağları hem lineer hem de lineer olmayan sınırlar ürettiğinden bu algoritma ile üretilen karar ağaçları lineer sınıf sınırlarına sahip olmayabilir. Eğer mutlaka lineer sınır çizgileri istenirse düğümlerde saklı katmanı olmayan yapay sinir ağları (perceptron) kullanılmaktadır.

(5) CART algoritmasının yerel minimumlardan çok etkilenmesi sebebiyle Eğik Sınıflandırıcı (Oblique Classifier - OC1) geliştirilmiştir. CART tarafından bulunan çözümün bütün parametrelerine küçük rasgele eklemeler yapılır ve bu sayede çözüm uzayında bir sıçrama yapılmış olur (yerel minimumdan kurtulmak için) ve CART algoritması bu yeni hiper düzlem parametrelerinden başlanarak yeniden uygulanır. Bu işleme bir yakınsama olana dek devam edilir.

(6) Lineer Karar Ağacı Makinelerinde (Linear Machine Decision Trees- LMDT), K sınıflı bir problem için her düğümden K adet kol çıkar. Her bir kolda bir sınıfa ait lineer ayırt etme fonksiyonu yer alır. Fonksiyonun parametreleri minimum hata yapacak şekilde optimize edilir.

(7) QUEST - Quick Unbiased Efficient Statistical Tree algoritması FACT ağaçlarının bir versiyonudur. Bu algoritmada öncelikle tüm sınıflar iki süper sınıfa gruplanır ve her düğümde verileri bu iki süper sınıfa bölen ikinci dereceden (quadratic) ayırt etme fonksiyonları yer alır

(8) Lineer Ağaçlar (Linear Tree- LTREE) her düğümünden iki kol çıkan ağaçlardır. Düğümlerdeki lineer ayırt etme doğrultusu LDA ile bulunur. Düğümdeki tüm örnekler LDA'ın uzayına dönüştürüldükten sonra en iyi ayrıldıkları noktayı bulmak için tüm olasılıklar denir. Bu algoritmanın ayırt etme fonksiyonlarında ikinci dereceden fonksiyonları kullanan, QTREE, mantıksal ayırt edici fonksiyonları kullanan LGTREE isimli versiyonları mevcuttur.

Yıldız ve Alpaydın'ın çalışmasında yer alan Çizelge 3.2'de, popüler karar ağacı algoritmalarının çeşitli özelliklerine göre karşılaştırılması verilmiştir.

Çizelge 3.2 Popüler karar ağacı algoritmaları

Algoritma	Karar düğümü	Her bir karar düğümünden çıkan dal sayısı	Gruplama	Hata Ölçüsü	İzdüşüm vektörünü arama	Sınıflandırma noktasını arama
C4.5	Uni	2	-	Impurity	-	Exhaustive
Friedman's	Uni/Lin	2	-	Kolm-Smir	Analitik	Analitik
CART	Lin	2	-	Impurity	Backfitting	Exhaustive
FACT	Uni/Lin	K	-	Fisher's	Analitik	Analitik
ID-LP/MLP	Lin/Non	2	Eğiticili	MSE	Gradient	Gradient
OC1	Lin	2	-	Info Gain	Hill climb	Exhaustive
LMDT	Lin	K	-	Misclass	Thermal	Thermal
QUEST	Uni/Lin	2	Eğiticişiz	Fisher's	Analitik	Analitik
Ltree	Lin	2	-	Info Gain	Analitik	Exhaustive
Cruise	Uni/Lin	K	-	Fisher's	Analitik	Analitik

3.3 Gerçekleştirilenler

Geliştirilen Cline (**C**lassification **L**ine) algoritması, karar ağacı tabanlı bir algoritmadır. Veri uzayını hiyerarşik olarak belirlenen kriterlere göre bölgelere böler. Bölme işlemi bölge içinde sadece bir sınıftan örnekler kalana kadar ya da 4'ten az sayıda örnek kalana kadar özyinelemeli bir yapıda devam eder.

Cline karar ağacı çok değişkenli bir karar ağacıdır. Her düğümünde d boyutlu verinin tüm boyutları kullanılarak karar verilir. Her bir düğümünden iki dal ayrıldığından ikili bir ağaçtır. Sayısal örneklerden oluşan tüm sınıflandırma işlemlerinde (d boyut, c sınıf) kullanılabilir.

İki boyutlu bir uzayda sınıfları birbirinden ayıran sınır bir doğru iken, üç boyutlu uzayda bir düzlem, daha yüksek boyutlu uzaylarda ise bir hiper düzlemdir. Her bir düğümde bu hiper düzlemin parametreleri bulunur ve örnekler bu parametrelere göre dallara gönderilir. Cline algoritması Çizelge 3.3'te verilmiştir.

Çizelge 3.3 Cline algoritması

Giriş: Eğitim seti	
Çıkış: Karar ağacı	
Başlangıç: Veri kümesindeki tüm örnekler aynı sınıftan mı? Veya Örnek sayısı 4'ten küçük mü? (Ön budama)	
Evet	Hayır
Düğümü Etiketle Çık	Sınır hiper düzlemini bul ve düğüm olarak ekle
	Verileri sınır hiper düzlemine göre iki bölgeye ayır ve dallara ata
	Her bir daldaki veriler için algoritmanın başına dön

Cline algoritmasında bir yaprak düğümün etiketi, yaprağın bölgesindeki en yüksek frekanslı sınıf olarak atanır.

Cline algoritmasında yer alan hiper düzlemin bulunması işlemi için birçok yaklaşım geliştirilmiş ve çeşitli Cline versiyonları önerilmiştir.

3.3.1 Cline Algoritmasının Versiyonları

Görselleştirilebilmesi için metotlar iki boyutlu uzaydaki veriler üzerinde anlatılmıştır. 2 boyutlu uzayda sınıflar birbirinden bir doğru ile ayrılmaktadır. Ayrıca 2’den daha fazla sınıf içeren problemler iki sınıf içeren alt problemlere dönüştürülerek çözülmektedir (Bkz. Bölüm 3.3.5). Bu nedenle tüm versiyonlardaki problemler 2 sınıflı problemler ele alınarak anlatılmıştır.

CL2: Şekil 3.3.a’da Farklı sınıflardan olan örnekler arasında birbirine en yakın olan iki örnek ‘A’ ve ‘B’ örnekleridir. Bu iki noktanın bulunması Eşitlik 3.4’te verilmiştir. Eşitlik 3.4’teki $dist(x_i, x_j)$ iki nokta x_i ve x_j arasındaki Öklid uzaklığını, C1 ve C2 örneklerin ait oldukları sınıfları göstermektedir.

$$(i, j) = \arg \min_{i,j} \{dist(x_i, x_j) \mid x_i \in C_1, x_j \in C_2\}, A = x_i, B = x_j \quad (3.4)$$

Bu iki örneğin (‘A’ ve ‘B’) orta noktası (m) Eşitlik 3.5 ile bulunmaktadır.

$$m = \frac{1}{2} \sum_{j=1}^p A_j + B_j \quad (3.5)$$

m ’den geçen ve bu iki örneği birleştiren doğruya (w') dik olan tek bir doğru (w) vardır. İşte bu doğru CL2’de sınıflandırma doğrusu olarak kullanılmaktadır. Bu doğru, Eşitlik 3.6’dan faydalanılarak bulunmaktadır.

$$\sum_{j=1}^p w'_j A_j + w'_0 = \sum_{j=1}^p w'_j B_j + w'_0 = 0, w' \times w = -1, \sum_{j=1}^p w_j m_j + w_0 = 0 \quad (3.6)$$

CL4: Şekil 3.3.b’de A örneğine B örneğinden sonra diğer sınıftan en yakın örnek D örneğidir. Aynı şekilde B örneğine A örneğinden sonra diğer sınıftan en yakın örnek C örneğidir. C ve D örnekleri Eşitlik 3.7 ve 3.8 ile bulunmaktadır.

$$k = \arg \min_k \{dist(x_k, B) \mid x_k \in C_1, x_k \neq A\}, C = x_k \quad (3.7)$$

$$t = \arg \min_t \{dist(x_t, A) \mid x_t \in C_2, x_t \neq B\}, D = x_t \quad (3.8)$$

A ve C örneklerinin orta noktasıyla ($m1$), B ve D örneklerinin orta noktasını ($m2$) birleştiren doğrunun (w') parametreleri Eşitlik 3.9 ve 3.10 ile bulunmaktadır.

$$m1 = \frac{1}{2} \sum_{j=1}^p A_j + C_j, m2 = \frac{1}{2} \sum_{j=1}^p B_j + D_j \quad (3.9)$$

$$\sum_{j=1}^p w'_j m1_j + w'_0 = \sum_{j=1}^p w'_j m2_j + w'_0 = 0 \quad (3.10)$$

Eşitlik 3.10'da bulunan w' doğrusuna dik olan ve bu doğrunun tam ortasından geçen tek bir doğru (w) vardır.

$$m' = \frac{1}{2} \sum_{j=1}^p m1_j + m2_j, w' \times w = -1, \sum_{j=1}^p w_j m'_j + w_0 = 0 \quad (3.11)$$

Eşitlik 3.11'de bulunan bu doğru (w) CL4'te sınıflandırma doğrusu olarak kullanılmaktadır.

CL2 ve CL4 algoritmalarının ağaç oluşturma hızları oldukça yüksek olmalarına rağmen, gürültüye aşırı duyarlıdırlar. Bu nedenle CLM, CLLDA, CLLVQ algoritmaları geliştirilmiştir.

CLM(Cline Mean): Şekil 3.3.c'de, A ve B noktaları sınıflarının orta noktalarıdır ve Eşitlik 3.12'deki şekilde bulunmaktadır.

$$A = \{mean(x_i) \mid x_i \in C_1\}, B = \{mean(x_i) \mid x_i \in C_2\} \quad (3.12)$$

A-B doğru parçasının (w') orta noktasından (m) geçen ve ona dik olan tek bir doğru (w) vardır. Bu doğru (w) sınıflandırma doğrusu olarak CLM algoritmasında kullanılır ve CL2 algoritmasındaki Eşitlik 3.6 ile bulunur. Literatürde bu yaklaşım en yakın merkezle sınıflandırma (nearest mean classifier) olarak isimlendirilmektedir.

CLLDA(Cline Linear Discriminant Analysis): Bu metotta önce sınıf merkezlerinin arasındaki mesafeyi maksimum, sınıf içi varyansı minimum tutan Linear Discriminant Analysis (LDA) (Fisher, 1936) kullanılarak çok boyutlu veriyi tek boyuta indirgeyen dönüşüm vektörü bulunur. Şekil 3.3.d'de, A ve B farklı sınıflardan birbirine en yakın olan iki

noktadır. Bu iki nokta CL2 algoritmasındaki Eşitlik 3.4 ile bulunur. A-B doğru parçasının (w') orta noktasından (m) geçen ve LDA'in dönüşüm vektörüne $w''_{1..p}$ dik olan tek bir doğru (w) vardır. Bu doğru CLLDA algoritmasında kullanılan sınıflandırma doğrusudur ve Eşitlik 3.13 ile bulunur.

$$w''_{1..p} = LDA(X, C), \quad w'' \times w = -1, \quad \sum_{j=1}^p w_j m_j + w_0 = 0 \quad (3.13)$$

Eşitlik 3.13'deki C, örneklerin sınıflarını göstermektedir.

CLLVQ(Cline Linear Vector Quantization): Linear Vector Quantization (LVQ) (Kohonen, 2000) kullanılarak iki sınıfı en iyi temsil eden centroidler (A ve B) bulunur (Eşitlik 3.14).

$$[A, B] = LVQ(X, C) \quad (3.14)$$

Eşitlik 3.14'te C, örneklerin sınıflarını göstermektedir. A-B doğru parçasının (w') orta noktasından (m) geçen ve ona dik olan tek bir doğru (w) vardır. Bu doğru Eşitlik 3.6 ile bulunur ve sınıflandırma doğrusu olarak CLLVQ algoritmasında kullanılır.

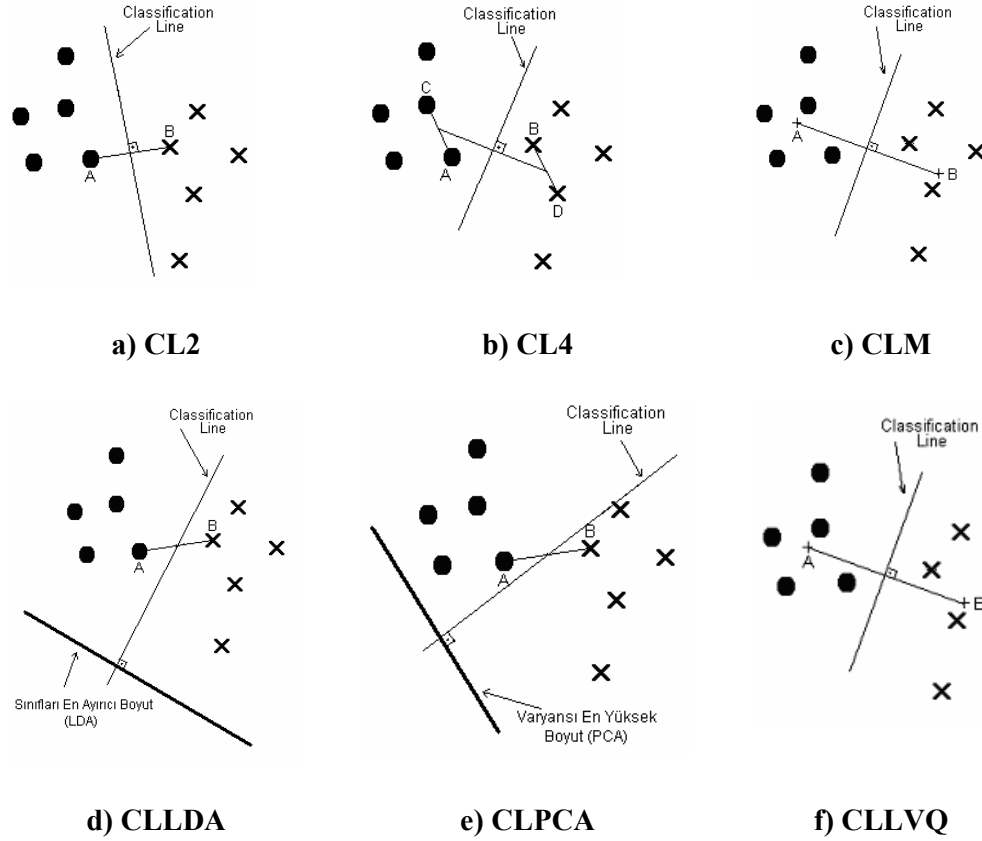
CLPCA(Cline Principal Component Analysis): Önce temel bileşen analizi (PCA) ile verilerin, üzerine izdüşümü yapıldığında varyansının en yüksek olduğu boyut $w''_{1..p}$ bulunur (Eşitlik 3.15).

$$w''_{1..p} = PCA(X) \quad (3.15)$$

Bu boyuta ($w''_{1..p}$) dik olan ve CL2'deki gibi farklı sınıflardan olup birbirine en yakın olan iki noktanın (A ve B) orta noktasından (m) geçen doğru (w) CLPCA'de sınıflandırma doğrusu olarak kullanılmaktadır (Eşitlik 3.16).

$$w'' \times w = -1, \quad \sum_{j=1}^p w_j m_j + w_0 = 0 \quad (3.16)$$

Şekil 3.3'te Cline algoritmasının farklı versiyonlarının aynı veri kümesi üzerinde iki boyutlu uzayda belirledikleri sınıflandırma doğruları görülmektedir.



Şekil 3.3 Cline versiyonları

CLMIX(Cline Mix) : Cline versiyonlarının sınıflandırma doğruluklarını analiz ederken hiçbir versiyonun tüm veri kümelerinde diğer tüm versiyonlardan daha iyi olmadığı ortaya çıkmıştır. En iyi üç versiyon CLM, CLLDA, ve CLLVQ'dir. Bu üç versiyonu tek bir ağaçta birleştirme fikrinden yola çıkılarak CLMIX geliştirilmiştir. CLMIX'de her düğümde veri noktaları 3 metotla ayrı ayrı sınıflandırılır. O anki veri kümesini en yüksek doğrulukla sınıflandıran versiyon ağaca yerleştirilir (Eşitlik 3.17). Böylece o anki veri kümesi için en iyi sınıflandırma metodu kullanılmış olur.

$$w_{CLMIX} = \arg \max_{w \in \{w_{CLM}, w_{CLLDA}, w_{CLLVQ}\}} cs(X, C, w) \quad (3.17)$$

Eşitlik 3.17'deki $cs(X, C, w)$, C etiketlerine sahip X veri kümesinin, w hiper düzlemi ile sınıflandırıldığındaki sınıflandırma başarısını göstermektedir.

CLLVQ4: CLM ve CLLVQ algoritmaları mantıklarından dolayı birbirlerine çok benzer sonuçlar üretmektedir ve her ikisi de genelde ikiden fazla gruba ayrılmış verilerde başarılı olamamaktadırlar. Bu sebeple CLLVQ4 geliştirilmiştir. CLLVQ4'te LVQ ile iki merkez

yerine her sınıfa ait ikişer toplamda 4 merkez bulunur (Eşitlik 3.18).

$$\begin{aligned} [e1, e2, e3, e4] &= \{LVQ(X, T) \mid e1 \in C_1, e2 \in C_1, e3 \in C_2, e4 \in C_2\}, \\ [A, B] &= \{rand(e1, e2, e3, e4) \mid A \in C1, B \in C2, C1 \neq C2\} \end{aligned} \quad (3.18)$$

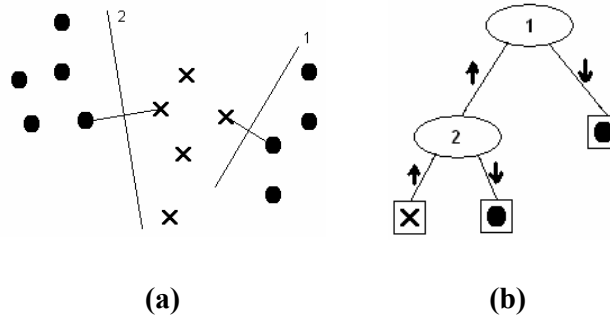
Daha sonra farklı sınıfları temsil eden rasgele iki merkezin (A ve B) orta noktasından (m) geçen ve ikisini birleştiren doğruya (w') dik olan düzlemle w veriler ikiye bölünür (Eşitlik 3.6).

CLdal/CLyaprak: Test işleminde sadece düğümlerdeki sınıf etiketlerinin kullanılmasına alternatif olarak her bir düğümdeki sınıf olasılıklarının da işleme katılması düşünülmüş ve bu yöntemde gerçekleştirilmiştir. Buna göre ağaç oluşturulurken her bir düğümde hiper düzlem parametrelerine ek olarak sınıf dağılımları da kaydedilmiştir. Test örneği ağaç üzerinde ilerlerken geçtiği her bir düğümdeki bu olasılıkları da toplayarak ilerler. Ağacın yaprağına ulaştığında her bir sınıfa ait olma olasılıklarına yaprağın etiketini de ekler ve buna göre örnek, olasılığı yüksek olan sınıfa atanır. Bu yaklaşım CLdal olarak isimlendirilmiştir. Tez raporunda, geliştirilen algoritmaların isimlerinin sonunda yer alan “dal” ifadesi bu metodun kullanıldığını göstermektedir. Sadece yaprak etiketlerinin kullanılması ise isim sonlarına “yaprak” ifadesi eklenerek ifade edilmiştir.

CLbudama: Tez raporunda algoritma isimlerinin sonunda “_p” ifadesi varsa oluşturulan karar ağacının Bölüm 3.2.2’de anlatıldığı gibi budandığını göstermektedir.

3.3.2 Cline Algoritmasıyla Verileri Sınıflandırma

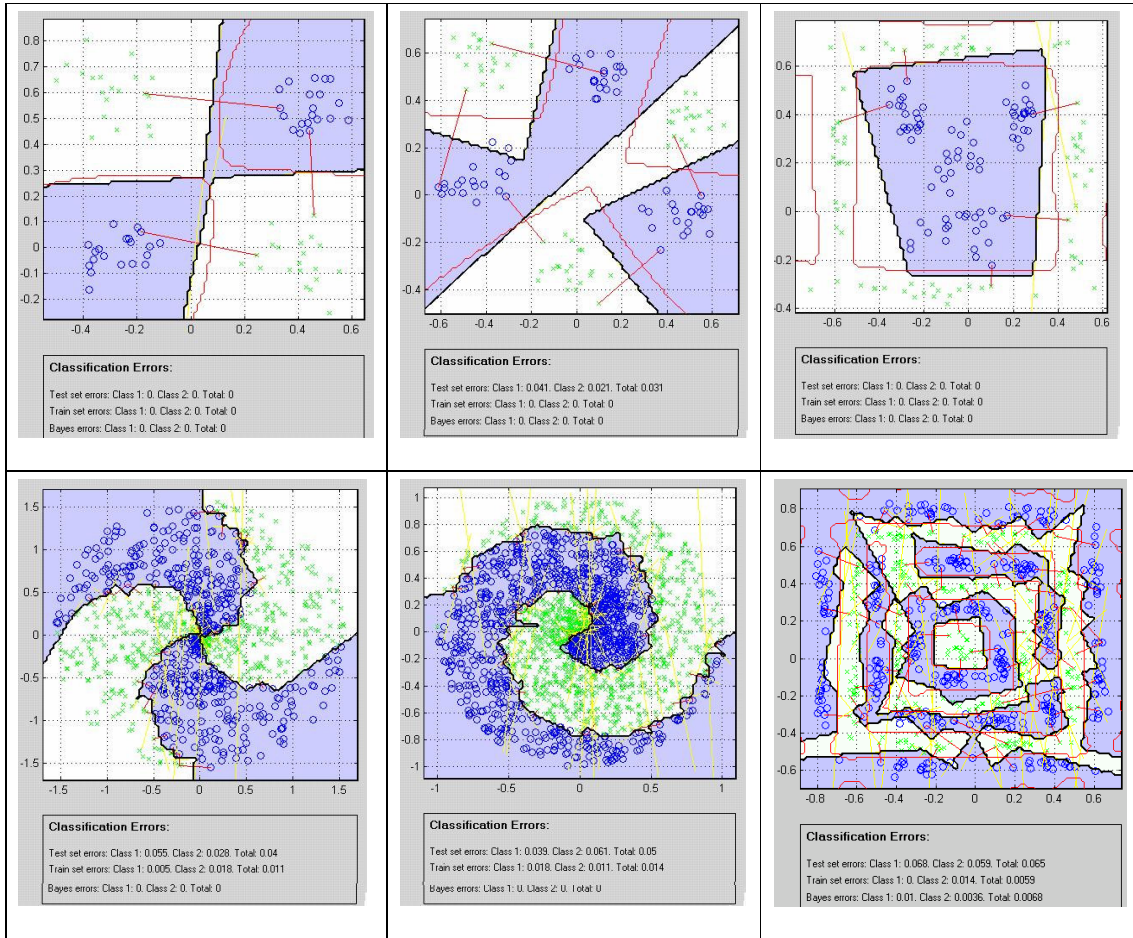
Cline algoritmalarında bir karar ağacı oluşturulduktan sonra ağacın her bir düğümünde bir hiper düzlem, her bir yaprağında ise bir sınıf etiketi yer almaktadır. Eğitim işlemi (karar ağacının oluşturulması) sonunda Şekil 3.4.a’deki veri kümesi için CL2 metoduyla oluşturulan karar ağacı Şekil 3.4.b’de verilmiştir.



Şekil 3.4 (a) Veri kümesi ve sınıflandırma doğruları (b) Cline karar ağacı

Verilerin sınıflandırılmasında işleme ağacının en yukarısındaki sınır hiper düzleminden (sınıflandırma doğrusundan) başlanarak sınıflandırılacak örneğin hiper düzlemin altında ya da üstünde kalmasına göre ağacın aşağıdaki dallarına doğru hareket edilir. Sınıfları gösteren bir seviyeye (yaprak) varılıncaya kadar ilerlenir. Örnek, yaprağın sınıf etiketi ile sınıflandırılır.

Şekil 3.5'te CL2 ile iki boyutlu ve iki sınıf içeren farklı veri kümeleri üzerinde yapılan deneme sonuçları görülmektedir. Eğitim işleminde tüm verinin %20'si, test işleminde %80'i kullanılmıştır.



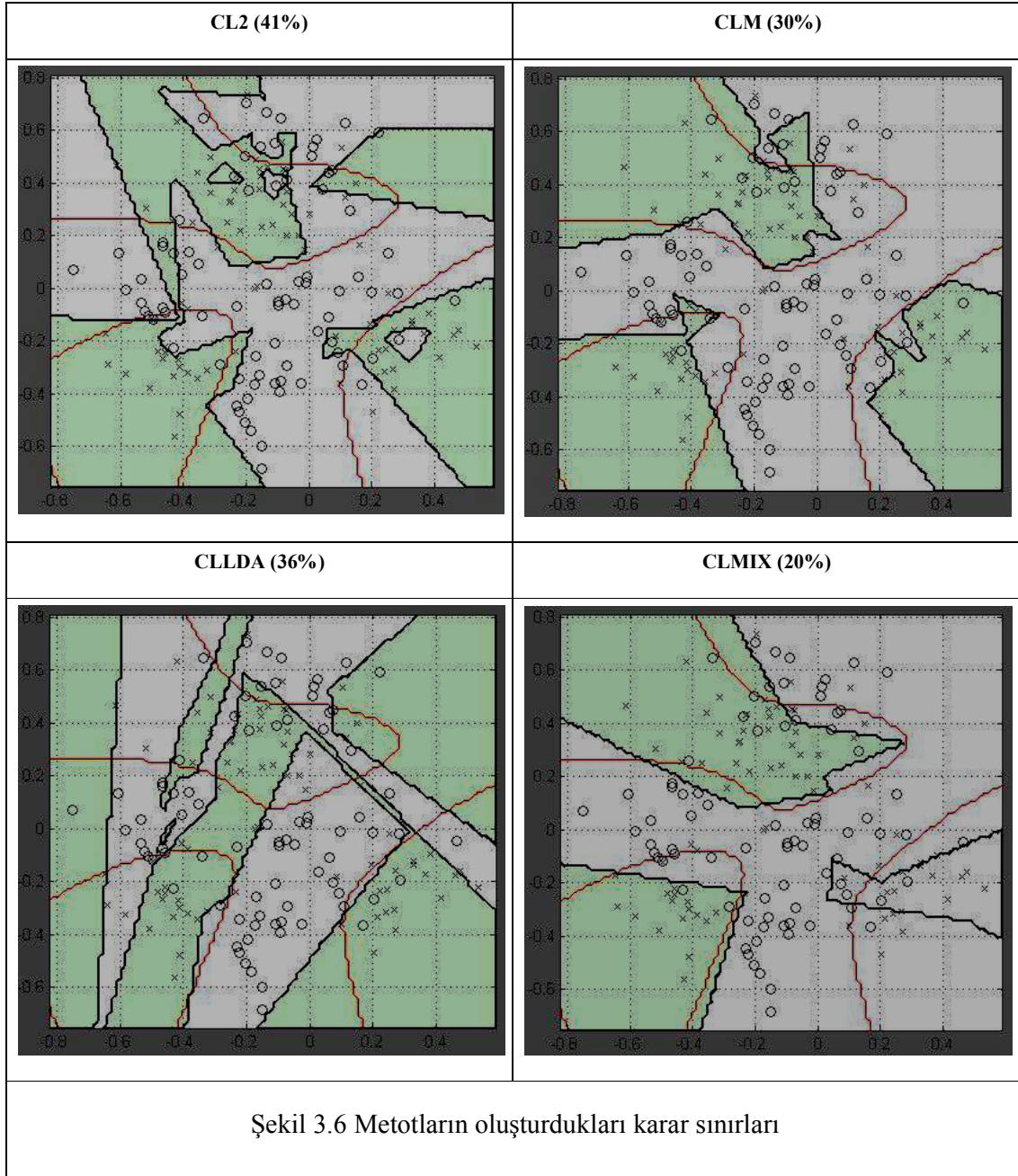
Şekil 3.5 Bazı veri kümelerinin CL2 ile yapılan sınıflandırmaları

Bölüm 3.3.1'de anlatıldığı gibi, Algoritmaların “dal” versiyonlarında ise örneklerin sınıflandırılmasında sadece yapraklardaki sınıf etiketleri kullanılmaz. Ek olarak örneğin kök düğümünden yaprağa kadar geçtiği düğümlerdeki sınıf dağılımları da işleme katılır.

3.3.3 Cline Algoritmalarının Görsel Karşılaştırması

Sınıfların birbirlerinden kolaylıkla ayrılabilirlikleri durumlarda tüm Cline metotları iyi

çalışmaktadır. Ancak veri kümesi birbirinden lineer ayrılamadığı ve/veya gürültülü olduğu durumlarda metotların performansları farklılıklar göstermektedir. Bu farkı gösterebilmek için Şekil 1.4'deki yapay veri kümesi oluşturulmuştur. Gerçek sınıf dağılımları bilinmekte ve optimum Bayes hatası %21'dir. Verilerin %60'ı eğitim, geri kalanı test için kullanılmıştır. Geliştirilen metotların iki boyutlu aynı veri kümesi üzerinde oluşturdukları karar sınırları Şekil 3.6'da verilmiştir. Kalın çizgiler Cline versiyonlarının, ince çizgiler optimum Bayes metodunun karar sınırlarıdır. Her metoda ait hata yüzdesi versiyon isimlerinin yanında verilmiştir.



CL2 algoritması gürültüye karşı oldukça duyarlıdır. Bu Şekil 3.6'daki kompleks karar sınırlarından ve kötü performansından belli olmaktadır. CLM algoritması CL2'den gürültüye karşı daha dayanıklıdır ve karar sınırları CL2'den daha az karmaşıktır. CLMIX algoritması en az karmaşıklıkta karar sınırlarına sahiptir ve en küçük ağaçla en yüksek başarıyı sağlamıştır. Bu sonuçlar CLMIX'in genelleştirebilme kabiliyetini göstermektedir. CLMIX'in performansı optimum Bayes metoduyla yarışır seviyededir.

3.3.4 Cline Hiper Düzlem Parametrelerinin Bulunmasında Kullanılan Analitik Denklemler

Bu bölümde verilen analitik denklemler CL2 algoritması üzerinden anlatılmıştır, ancak diğer tüm versiyonlara kolaylıkla aktarılabilir.

p boyutlu bir uzayda karar sınırını ifade eden hiper düzlem denklemi Eşitlik 3.19'da verilmiştir. Eşitlikte bulunması gereken parametreler p boyutlu bir uzay için $(w_1, w_2, \dots, w_p, w_{p0})$ parametreleridir.

$$w_1x_1 + w_2x_2 + \dots + w_px_p + w_0 = 0 \quad (3.19)$$

$A(x_1, x_2, \dots, x_p)$ ve $B(x_1, x_2, \dots, x_p)$ noktalarından CL2'deki sınır hiper düzlemini elde etmek için analitik geometriden faydalanılmıştır.

3 boyutlu bir uzayda $m(x_1, x_2, x_3)$ noktasından geçen ve $W = [w_1, w_2, w_3]$ vektörüne dik olan düzlem denklemi Eşitlik 3.20'de verilmiştir [6].

$$w_1(x - x_1) + w_2(y - x_2) + w_3(z - x_3) = 0 \quad (3.20)$$

Eşitlik 3.20 daha açık yazılırsa Eşitlik 3.21 elde edilir.

$$\begin{aligned} -w_1x_1 - w_2x_2 - w_3x_3 + w_1x + w_2y + w_3z &= 0 \\ w_1 &= -w_1 \\ w_2 &= -w_2 \\ w_3 &= -w_3 \\ w_0 &= w_1x + w_2y + w_3z \\ w_1x_1 + w_2x_2 + w_3x_3 + w_0 &= 0 \end{aligned} \quad (3.21)$$

Eşitlik 3.21'in sonunda elde edilen denklem, Eşitlik 3.19'un 3 boyutlu uzay için özelleştirilmiş halidir. Bu durumda p boyutlu uzayda hiper düzlemi elde etmek için, düzleme dik bir doğru ve düzlemde yer alan bir nokta yeterlidir. Düzleme dik olan doğru elimizdeki A

ve B noktalarının farkıyla elde edilir.

$$W'(x_1, x_2, \dots, x_p) = A(x_1, x_2, \dots, x_p) - B(x_1, x_2, \dots, x_p) \quad (3.22)$$

Tüm Cline versiyonlarında bir hiper düzleme dik ve belirli bir noktadan geçen hiper düzlemlerin parametreleri bulunmaktadır. Düzlemde yer alan nokta (m) ise A ve B noktalarının orta noktasıdır.

$$m(x_1, x_2, \dots, x_p) = (A(x_1, x_2, \dots, x_p) + B(x_1, x_2, \dots, x_p)) / 2 \quad (3.23)$$

Eşitlik 3.22 ve Eşitlik 3.23'ten faydalanarak hiper düzlemin parametreleri bulunmuştur.

3.3.5 Cline Algoritmasını N Adet Sınıf İçin Genelleştirmek

Cline algoritması temelde iki sınıfa ayıran bir metottur. Sadece iki sınıfa bölen sınıflandırma metotlarının daha fazla sınıfa sahip problemlere de uygulanabilmesi için genelde kullanılan 2 farklı teknik mevcuttur.

1- Bire karşı hepsi : (1 vs All) : N adet sınıf için her seferinde bir sınıfa karşılık, diğer tüm sınıflar aynı sınıftan sayılarak eğitim işlemi yapılmıştır. Bu sayede N adet sınıflandırıcı oluşturulmuştur ve verdikleri kararlar birleştirilmiştir.

2- Bire karşı bir: (1 vs 1): Her sınıf ikilisi için bir sınıflandırıcı oluşturulmuş ($N*(N-1)/2$ adet) ve verdikleri kararlar birleştirilmiştir.

Kararlar birleştirilirken ECOC (Error-correcting output codes) metodu kullanılmıştır (Alpaydın, 2004). ECOC metodunda bire karşı hepsi için Çizelge 3.4 (a)'daki kod matrisi, bire karşı bir için (b)'deki kod matrisi oluşturulmuştur. Çizelge 3.4'te 4 sınıfa sahip bir veri kümesi için kod matrisleri verilmiştir.

Çizelge 3.4 ECOC kod matrisleri

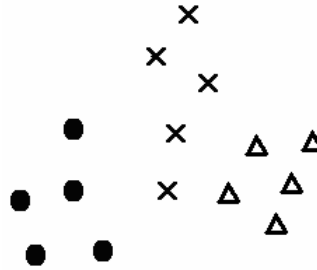
1	-1	-1	-1	1	1	1	0	0	0
-1	1	-1	-1	-1	0	0	1	1	0
-1	-1	1	-1	0	-1	0	-1	0	1
-1	-1	-1	1	0	0	-1	0	-1	-1

(a) Bire karşı hepsi için kod matrisi

(b) Bire karşı bir için kod matrisi

Sınıflandırıcıların çıkışları ya 1 ya da -1'dir. Kod matrislerinin satırları sınıfları, sütunları sınıflandırıcıları göstermektedir. Bire karşı hepsi kod matrisinde örneğin 3. sütun(sınıflandırıcı) 3.sınıfın örneklerini 1,2 ve 4. sınıfların örneklerinden ayırmaya çalışmaktadır. Bire karşı bir kod matrisinde ise 3. sütundaki sınıflandırıcı 1. sınıfın örneklerini 4. sınıfın örneklerinden ayırmaya çalışmaktadır. Bu kod matrisindeki 0'lar "ne olursa olsun" durumunu ifade etmektedir. Bir test örneği geldiğinde tüm sınıflandırıcılara verilmekte ve sınıflandırıcı sonuçlarından 1 ve -1'lerden oluşan bir dizi elde edilmektedir. Bu dizi kod matrisinin en çok hangi satırına benziyorsa o satır numarası ile etiketlenir.

Bu çalışmada her iki metotta denenmiş ve bire karşı bir metodu daha önce yapılan çalışmalara paralel olarak daha yüksek performans göstermiştir. Şekil 4.5'teki veri kümesi için doğrusal sınırlar belirleyen (Ör: Cline, Destek Vektör Makineleri vs.) sınıflandırıcılar birleştirilirken bire karşı bir metodu kullanıldığında sınıflandırıcılar başarılı olurken, bire karşı hepsi metodu kullanıldığında (çarpılar bir sınıf, daireler ve üçgenler diğer sınıf olduğunda) linner sınıflandırıcılar başarılı olamayacaklardır.



Şekil 3.7 Bire karşı bir ve bire karşı hepsi metotlarının karşılaştırılması için veri kümesi

Şekil 3.7'deki 3 sınıfı doğrusal modellerle, bire karşı hepsi ile birbirinden ayırmak mümkün olmamasına rağmen; bire karşı bir ile ayırmak mümkündür.

3.4 DeneySEL Sonuçlar

Geliştirilen Cline algoritmalarının diğer mevcut sınıflandırıcılarla karşılaştırılması 2 ayrı veri kümesi koleksiyonu üzerinde yapılmıştır. İlki makine öğrenmesi problemlerinde yaygın olarak kullanılan UCI (Blake vd., 1998) veri kümesi koleksiyonunun 8 veri kümesi içeren bir alt kümesi, diğeri ise ilaç verileri içeren 6 veri kümesinden oluşan bir koleksiyonudur.

3.4.1 UCI Veri Kümelerinin Sınıflandırılması

Geliştirilen metot bir karar ağacı algoritması olduğundan literatürdeki mevcut karar

ağaçlarıyla karşılaştırılması düşünülmüştür. Lim ve arkadaşları makalelerinde (Lim vd., 2000) 22 adet karar ağacının karşılaştırması yapmış ve kullandığı veri kümelerini yayınlamıştır. Cline Ailesinin Algoritmaları da Lim'in çalışmasında kullanılan veriler üzerinde denenmiş ve böylece tam bir karşılaştırma yapılabilmiştir. Çizelge 3.5'te karşılaştırma için kullanılan veri kümeleri ve özellikleri verilmiştir.

Çizelge 3.5 Veri kümelerinin özellikleri

Veri Kümesi	Kod	Özellik sayısı	Sınıf Sayısı	Örnek Sayısı	Rasgele Başarı(%)
Breast cancer	Bcw	9	2	683	64.71
Boston housing	Bos	12	3	506	33.33
Congressional voting	Vot	16	2	435	61.36
Bupa liver disorders	Bld	6	2	345	57.14
StatLog heart disease	Hea	7	2	270	55.56
Pima Indians diabetes	Pid	7	2	532	66.67
StatLog image	Seg	19	7	2310	14.29
StatLog vehicle	Veh	18	4	3772	25.88

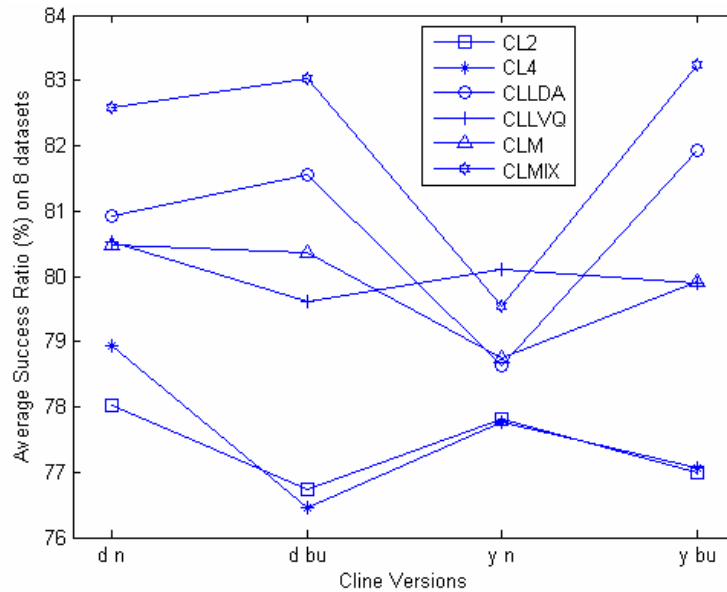
Bu çalışmada hiper düzlemlerin bulunması için 6 metot (CL2, CL4, CLM, CLLDA, CLLVQ ve CLMIX), ağaç oluşturulduktan sonra bir test örneğinin sınıfının belirlenmesi için 2 metot (yaprak ve dal), sınıflandırmada kullanılan ağaç için 2 farklı (budanmış ve budanmamış) ağaç kullanılmıştır. Diğer bir deyişle $24(2*2*6)$ farklı cline versiyonu, 8 veri kümesi üzerinde 10'lu çapraz geçirme kullanılarak çalıştırılmıştır. Cline metotlarının ortalama doğrulukları ve varyansları Çizelge 3.6'da verilmiştir. Çizelge'de Yaprak,dal, budanmış ve budanmamış metotlarının karşısındaki değerler $960(2*6*8*10)$ değer ortalaması ve varyansıdır. CL2, CL4, CLM, CLLDA, CLLVQ ve CLMIX metotlarının karşısındaki değerler $320(2*2*8*10)$ değer ortalaması ve varyansıdır.

Çizelge 3.6 Cline metotlarının ortalama doğruluk oranları ve varyansları.

Metot	Başarı Oranı	Varyans
Yaprak	79.3	1.9
Dal	79.9	2.09
Budanmamış	79.5	1.46
Budanmış	79.7	2.45
CL2	77.4	0.62
CL4	77.6	1.06
CLLDA	80.8	1.49
CLLVQ	80.0	0.37
CLM	79.9	0.79
CLMIX	82.1	1.73

Çizelge 3.6 incelendiğinde budama işleminin ortalama başarı oranı üzerinde küçük bir iyileştirme yaptığı görülmektedir. Aynı şekilde ağaç sınıflandırma türleri arasında dalların kullanımı, yaprakların kullanımına göre biraz daha yüksek bir doğruluğa sahiptir. Hiper düzlem belirleme metotları arasında ise CLMIX belirgin bir farkla en yüksek doğruluğa sahip metottur.

Şekil 3.8’de altı farklı hiper düzlem belirleme metodunun karşılaştırılması yapılmıştır. Bu Şekilde x ekseninde yer alan ‘d n’ dal kullanımını ve budama işleminin olmadığını, ‘d bu’ dal kullanımını ve budama yapıldığını, ‘y n’ yaprak kullanımını ve budama işleminin olmadığını, ‘y bu’ yaprak kullanımını ve budama yapıldığını ifade etmektedir. Bu şekildeki her bir değer 80 (8 veri kümesi * 10’lu çapraz geçişleme) değerinin ortalaması alınarak bulunmuştur.



Şekil 3.8 Hiper düzlem belirleme metotlarının karşılaştırılması.

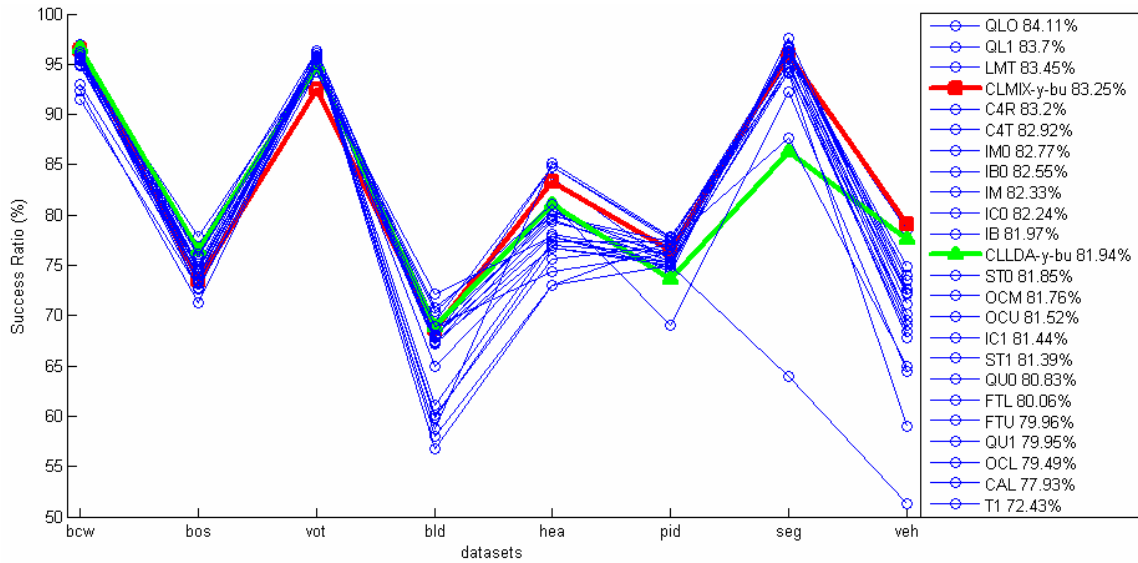
Şekil 3.8’de de budama işleminin CL2, CL4 ve CLLVQ metotları haricinde başarıyı arttırdığı ve budama yapılmadığında dal kullanımının yaprak kullanımına göre daha yüksek başarılarla sahip olduğu görülmektedir. Bu sonuçlara göre budama yapılarak eğitim kümesinin boyutu azaltılmak istenmediğinde sınıflandırmada dalların kullanılmasının daha iyi olduğu ve bu metodun başka karar ağacı algoritmalarında da uygulamanın iyi bir fikir olduğu söylenebilir.

Çizelge 3.7’de Lim’in çalışmasında (Lim vd., 2000) yer alan 22 karar ağacı algoritmasının isimleri ve kısaltmaları verilmiştir.

Çizelge 3.7 Karşılaştırma yapılan karar ağacı algoritmaları ve kısaltmaları

Kısaltma	Algoritma	Kısaltma	Algoritma
QU0	QUEST versiyonları (Loh, Shih 1997)	IB	IND versiyonları (Buntine 1992)
QU1		IB0	
QL0		IM	
QL1		IM0	
FTU	FACT versiyonları (Loh, Vanichsetakul 1998)	IC0	CART versiyonları (Breiman, Freidman, Olshen, Stone 1984)
FTL		IC1	
C4T	C4.5 versiyonları (Quinlan 1993)	ST0	S-PLUS versiyonları (Clark, Pregibon 1993)
C4R		ST1	
OCU	OC1 versiyonları (Murthy, Kasif, Salzberg 1994)	LMT	LMDT (Brodley, Utgoff 1995)
OCL		CAL	CAL5 (Müller, Wysotzki 1997)
OCM		T1	T1 single split

Şekil 3.9’da tek karar ağacı üreten 2 Cline versiyonu ile yine tek ağaç üreten 22 ağaç oluşturma algoritmasının 8 farklı veri kümesi üzerindeki sınıflandırma performansları karşılaştırılmıştır. Bu şekildeki her bir değer 10 (10’lu çapraz geçерleme) değerin ortalamasıdır.



Şekil 3.9 Karar ağacı algoritmalarının karşılaştırılması.

Lim'in çalışmasında en iyi performansa sahip karar ağacı algoritması %84.11'lik doğrulukla

Quick Unbiased Efficient Statistical Tree (QUEST- (Loh vd., 1997))’dir.

Cline metotlarının oluşturduğu en iyi ağaç %83.25’lik doğruluğa sahiptir ve en başarılı dördüncü algoritmadır. Bu sonuca bakılarak Cline versiyonlarının bilinen karar ağacı algoritmalarıyla yarışabilir sonuçları olduğu görülmektedir.

3.4.2 İlaç Veri Kümelerinin Sınıflandırılması

İlaç veri kümeleri SDF formatından sayısal hale MOE yazılımı kullanılarak dönüştürülmüştür. MOE tarafından çıkarılan molekül özellikleri listesine [16] adresinden erişilebilir. Çizelge 3.8’de MOE yazılımıyla elde edilen veri kümelerinin özellikleri verilmiştir.

Çizelge 3.8 Kullanılan ilaç veri kümeleri

Veri kümesi	Özellik sayısı	Sınıf sayısı	Örnek sayısı	Rasgele Başarı(%)
Bbp2	184	2	415	67
Mono	184	2	435	64
Clean1	166	2	476	57
Ca	184	3	513	51
Mutag	184	2	188	66
X232	184	5	232	40

Çizelge 3.9’da tek ağaçlı Cline algoritmalarının performansları verilmiştir. Sonuçlar 5’li çapraz geçişleme yapılarak ortalama olarak verilmiştir.

Çizelge 3.9 Tek ağaçlı Cline algoritmalarının sonuçları

Algoritma	Bbp2	Mono	Clean1	Ca	Mutag	X232	ortalama
CL2_y_n	77	83	83	61	87	43	72.37
CL2_y_bu	73	83	79	58	81	45	69.72
CL2_d_n	74	83	83	64	86	47	72.82
CL2_d_bu	71	81	73	59	83	39	67.67
CLM_y_n	76	89	87	62	90	48	75.42
CLM_y_bu	78	86	83	66	88	47	74.76
CLM_d_n	79	89	88	65	83	51	75.93
CLM_d_bu	75	86	83	66	87	48	74.16
CLLDA_y_n	69	86	75	62	73	38	67.06
CLLDA_y_bu	69	88	72	66	69	43	67.69
CLLDA_d_n	70	85	75	65	80	39	68.84

CLLDA_d_bu	70	90	74	64	69	44	68.52
CLLVQ_y_n	76	91	86	66	91	54	77.29
CLLVQ_y_bu	77	86	80	65	88	47	74.03
CLLVQ_d_n	76	89	85	63	89	50	75.32
CLLVQ_d_bu	72	85	81	61	87	44	71.63
<i>CLLDA CLM CLLVQ</i>							
CLMIX_y_n	70	83	76	57	73	39	66.28
CLMIX_y_bu	70	91	77	66	62	42	67.99
CLMIX_d_n	69	87	77	59	76	41	68.01
CLMIX_d_bu	69	91	76	62	69	40	67.74
<i>CLM CLLVQ</i>							
CLMIX_y_n	78	89	89	66	88	52	77.01
CLMIX_y_bu	75	87	85	60	86	43	72.5
CLMIX_d_n	78	91	86	66	87	50	76.1
CLMIX_d_bu	77	87	82	63	86	44	73.03

Çizelge 3.9 incelendiğinde en başarılı Cline versiyonun CLLVQ_y_n olduğu görülmektedir.

Kullanılan ilaç verilerinde, UCI koleksiyonunun aksine CLLDA'nın başarısı düşüktür. Bunun sebebinin, ilaç veri kümelerindeki özelliklerin birbirleriyle korelasyonunun yüksek olmasının, LDA hesaplanırken sebep olduğu rasgelelik olduğu görülmüştür. Ayrıca yine UCI koleksiyonunun aksine budamanın performansa olumsuz etki ettiği görülmektedir. Tam ağaçlar ilaç verilerinde budanmışlardan daha başarılıdır.

CLMIX versiyonuna CLLDA'nın etkisi olumsuz olmuştur. Bu sebeple 3 algoritma yerine 2 algoritmayı düğümlerinde kullanan CLMIX denenmiş ve daha başarılı sonuçlar elde edilmiştir.

Çizelge 3.10'da Cline ile karşılaştırılan diğer algoritmaların her bir veri kümesindeki ve ortalama başarıları verilmiştir. En son satırda ise en başarılı Cline versiyonunun sonuçları verilmiştir. Cline algoritmalarıyla karşılaştırılan algoritmaların detaylı açıklamalarına WEKA yazılımının dokümantasyonundan ulaşılabilir (Witten vd., 2005).

Çizelge 3.10 Cline ile karşılaştırılan diğer algoritmaların sonuçları

Algoritma	bbp2	Mono	Clean1	Ca	Mutag	X232	Ort
C4,5	77.35	87.36	83.82	64.13	85.11	51.29	74.84
RandomTree	74.22	86.44	76.47	65.11	87.23	46.12	72.60
RBFNetwork	66.51	64.37	56.51	51.66	69.15	41.38	58.26
SVM	76.87	94.71	82.77	72.12	90.96	54.74	78.70
NaiveBayes	74.70	85.06	74.37	47.56	85.64	47.84	69.20
CLLVQ_y	76	91	86	66	91	54	77.29

Çizelge 3.11’de ise algoritmalar en başarılıdan en başarısız göre sıralandıklarındaki sıra değerleri ve ortalama sıraları verilmiştir. Her bir hücrede sütununda yer alan veri kümesinde satırında yer alan algoritmanın kaçınıcı en iyi algoritma olduğu yer almaktadır.

Çizelge 3.11 Cline ile karşılaştırılan diğer algoritmaların sıralama sonuçları

Algoritma	bbp2	Mono	Clean1	Ca	Mutag	X232	Ort
C4,5	1	3	2	4	5	3	3
RandomTree	5	4	4	3	3	5	4
RBFNetwork	6	6	6	5	6	6	5.83
SVM	2	1	3	1	2	1	1.66
NaiveBayes	4	5	5	6	4	4	4.66
CLLVQ_y	3	2	1	2	1	2	1.83

Çizelge 3.10 ve 3.11’de görüldüğü gibi Cline algoritması, Destek Vektör Makineleri’nden sonra en yüksek performansa sahiptir.

3.5 Çıkarımlar

Bu bölümde Cline adında yeni bir karar ağacı algoritma ailesi geliştirilmiş ve hem genel amaçlı UCI veri kümelerinde, hem de ilaç veri kümelerinde mevcut algoritmalarla sınıflandırma doğruluğu açısından karşılaştırılmıştır. Bu denemelerden aşağıdaki çıkarımlara ulaşılmıştır:

- Her iki veri kümesi koleksiyonunda (UCI ve ilaç) da Cline algoritmaları umut vaat eden sonuçlar üretmişlerdir. Ancak en başarılı algoritmalar değillerdir. Bu nedenle tezin bir sonraki bölümünde yer alan tek bir karar ağacı yerine bir karar ormanı oluşturma çalışmaları yapılmıştır.
- UCI veri kümelerinde, ağaç budama işleminin Cline algoritmalarında, ortalama başarı oranı üzerinde küçük bir iyileştirme yaptığı görülmüştür.

- UCI veri kümelerinde, Bir test örneğinin bir karar ağacı kullanılarak sınıflandırılması için Cline içinde önerilen dalların kullanımı, klasik yaprakların kullanımına göre biraz daha yüksek bir doğruluğa sahiptir.
- UCI veri kümelerinde, Cline algoritmaları içinde, hiper düzlem belirleme metotları arasında CLMIX belirgin bir farkla en yüksek doğruluğa sahip metottur. Bunun sebepleri aşağıda verilmiştir.
 - CLMIX, bölgesel kararlar verebilir. Literatürdeki mevcut algoritmaların aksine, ağacın tüm karar düğümlerinde aynı algoritmaya bağımlı değildir.
 - Her Cline algoritmasını daha başarılı olduğu bölgeye uygulamaktadır.
 - Gürültüye daha dayanıklıdır.
- UCI veri kümelerinde, ağaç budama işleminin CL2, CL4 ve CLLVQ metotları haricinde başarıyı arttırdığı ve budama yapılmadığında dal kullanımının yaprak kullanımına göre daha yüksek başarılarla sahip olduğu görülmüştür. Bu sonuçlara göre budama yapılarak eğitim kümesinin boyutu azaltılmak istenmediğinde sınıflandırmada dalların kullanılmasının daha iyi olduğu ve bu metodun başka karar ağacı algoritmalarında da uygulamanın iyi bir fikir olduğu söylenebilir.
- UCI veri kümelerinde Cline metotlarının oluşturduğu en iyi ağaç %83.25'lik doğruluğa sahiptir ve en başarılı dördüncü algoritmadır.
- Kullanılan ilaç verilerinde, UCI koleksiyonunun aksine CLLDA'nın başarısı düşüktür. Bunun sebebinin, ilaç veri kümelerindeki özelliklerin birbirleriyle korelasyonunun yüksek olmasının, LDA hesaplanırken sebep olduğu rasgelelik olduğu görülmüştür. Ayrıca yine UCI koleksiyonunun aksine budamanın performansa olumsuz etki ettiği görülmektedir. Tam ağaçlar ilaç verilerinde budanmışlardan daha başarılıdır.

3.6 Gelecek Çalışmalar

Araştırmaya açık alanlar ve denenebilecek fikirler aşağıda listelenmiştir:

- İkili sınıflandırıcıları N sınıflı sınıflandırıcılara dönüştürürken başka metotların kullanımı:
 - 2'den fazla sınıf içinde Quest'teki gibi önce iki süper sınıf bulup sonra o iki süper sınıfı birbirinden ayıran cline'lar bulunabilir.

- Yönlendirilmiş Çevrimsiz Çizge (Directed Acyclic Graph) metodu kullanımı. Bu metot temel olarak, "Bire Karşı Bir" yöntemiyle benzerlik göstermektedir. Test verisi, ağaç üzerinde, kök düğümdeki ikiliden başlanarak karşılaştırılır ve ağacın en alt seviyesine kadar iteratif olarak ilerlenir. N seviyeli bir ağaç için önce [1,2] sınıf ikilisi kıyaslanır, bu ikiliden galip çıkacak sınıf *kazanan* ise; bir sonraki aşamada [*kazanan*,3] ikilisi kıyaslanır ve bu şekilde ilerlenerek, en sonunda [*kazanan*, N] ikilisi karşılaştırılıp, test verisinin hangi sınıfa daha yakın olduğu bulunur (Pal vd, 2004).
- Her adımda en iyi lineer ayıran özellik altkümelerinin bulunması. (örneğin sadece 2 elemanlı altkümeler olabilir) Olasılık sayısı çok olabileceğinden çıkışla korelasyonu yüksek olan ilk M tanenin X’li altkümeleri içinde arama yapılabilir. Bu yöntemle bir her bir düğümünde farklı özellikleri kullanan tek bir ağaç oluşturulur.
- Düğümlerde, iki boyutlu uzay için, her bir sınıfın varyansı en yüksek doğrultularının (first eigenvector) kesiştiği noktadan geçen ve iki doğrultuyu ortalayan doğrunun kullanılması.
- Birbirinden lineer ayrılabilen en büyük veri altkümelerini bulup bunları birbirinden ayıran doğrunun cline kabul edilmesi. Başlangıçta iki sınıfın birbirine en uzak örnekleri alınıp ve lineer ayrılabilme devam ettiği sürece bu sınıflara yeni örnekler eklenmeye devam edilebilir.
- Hiper düzlemin parametrelerinin bulunmasında ortak ayırt edici vektör (Discriminative Common Vector) kullanılabilir.
- Hiper düzlemin parametrelerinin bulunmasında bağımsız bileşen analizi (Independent component analysis-ICA) kullanılabilir.
- Doğadaki ağaçlarda dallar gövdeden üç farklı şekilde çıkıyor. A) karşılıklı çapraz. İkili sarmal 90 derece dönerek çıkıyor. B) İkili sarmal 90’dan daha küçük bir açıyla dönerek çıkıyor. C) tekli sarmal 90’dan daha küçük bir açıyla dönerek çıkıyor. Doğadaki bu yapının karar ağaçlarına bir uygulaması düşünülebilir. Bir hiper düzlemin parametreleri, altındaki düğümlerin parametrelerinin belirlenmesini direkt sağlayabilir ya da arama için bir başlangıç değeri olabilir.
- Düğümlerde, İki boyutlu uzay için, her bir sınıfın varyansı en yüksek doğrultularının (first eigenvector) kesiştiği noktadan geçen ve iki doğrultuyu ortalayan doğrunun kullanılması. Anca burada iki boyutlu uzayda dairesel dağılmış bir sınıf varsa, her yöne varyansı eşit gibi olacağından iyi sonuç alınamayabilir. Bu nedenle dairesel

olmayan alt özellik kümelerini kullanmak daha iyi sonuçlar üretecektir.

- Küresel / eliptik sınıflandırma ağaçları oluşturulabilir. (ilk denemeler yapıldı ancak iyi sonuçlar elde edilemedi.)
- Dendrogram ağaçları oluşturulabilir. (ilk denemeler yapıldı ancak boyut sayısı fazla olan örneklerde ağaç çok dengesiz oluyor ve sınıflandırma başarısı çok düşük.)
- Tek değişkenli Cline ağaçları denendi ancak çok değişkenliler kadar başarılı değiller. Ağaçlar oluşturulurken klasik tek değişkenli ağaç algoritmalarında olduğu gibi, cline algoritmasıyla verilerin özellikleri teker teker kullanılarak D adet ($d = \text{Boyut sayısı}$) ağaç oluşturulur ve en başarılı olan özellik seçilerek düğüme yerleştirilir.

4. SINIFLANDIRMA KOMİTELERİ

Tüm verilerde en iyi performansı gösteren bir sınıflandırıcı yoktur (No free lunch theorem) (Wolpert vd., 1995). Bir veri kümesinde en iyi performansı gösteren sınıflandırıcı bir başka veri setinde düşük performans gösterebilmektedir. Bu nedenle araştırmacılar bir veri setini sınıflandırırken genelde sadece bir sınıflandırıcı kullanmak yerine birkaç tanesini kullanırlar. Birden fazla sınıflandırıcının kullanımında, seçim ve birleştirme adında iki yaklaşım mevcuttur. Seçim yaklaşımında birkaç sınıflandırıcının eğitim kümesi üzerinde en doğru sınıflandırmayı yapanın kullanılması önerilmiştir. Ancak bu durumda, en iyi sınıflandırıcının doğru sınıflandıramadığı örnekleri doğru sınıflandıran ama genelde daha başarısız olan başka bir sınıflandırıcı olabileceğinden bu yaklaşım pek tercih edilmemektedir. Birleştirme yaklaşımında ise sınıflandırıcıların kararları çeşitli metotlarla tek bir karara dönüştürülmektedir. Diğer bir deyişle veri kümesi tüm sınıflandırıcılara verilmekte ve her biri kendi kararlarını ürettikten sonra bu kararlar birleştirilmektedir.

4.1 Tanımlar ve Kullanılan Semboller

Zayıf sınıflandırıcı: Başarısı rasgele başarıdan biraz daha yüksek olan sınıflandırıcılardır. Sınıflandırıcı komitelerinde yer alan sınıflandırıcılar genelde bu türe girerler. Komitelerin oluşturulmasındaki amaç zayıf sınıflandırıcıların birleşiminden güçlü bir sınıflandırıcı oluşturmaktır.

Güçlü sınıflandırıcı: Başarıları rasgele başarıdan daha yüksek olan sınıflandırıcılardır.

Bagging / Bootstrapping: N örnek içeren eğitim verisinden yine N adet örneğin tekrar izin verilerek seçilmesi işlemidir. Bu şekilde oluşturulan yeni veri kümeleriyle sınıflandırıcılar eğitilir ve kararlarının bir ölçüde birbirlerinden bağımsız olması sağlanır.

Subbagging: Bagging işleminden tek farkı, yeni seçilen örnek sayısının orijinal örnek sayısından daha az olmasıdır. Genelde örnek sayısı çok fazla olan veri kümelerinde hesapsal karmaşıklığın azaltılması amacıyla uygulanır.

Rasgele Alt Uzaylar (Random Subspace): Veri kümesindeki örneklerin tüm özelliklerinin yerine bir alt kümesinin kullanılması işlemidir. Bu şekilde her biri uzayın farklı özellik uzaylarını ifade eden veri kümeleri oluşturulmuş olur. Amaç yine sınıflandırıcı kararların birbirlerinden bağımsızlaştırılmasıdır.

Rasgele Alt Uzaylarda Bagging (RF): Rasgele alt uzaylar ve bagging işlemlerinin aynı anda

uygulanmasıdır. İlk olarak Breinman (Breinman, 1999) tarafından Random Forest algoritmasında kullanılmıştır. Kararların bağımsızlığını arttırmada hem rasgele alt uzaylarda hem de bagging'den daha başarılı sonuçlar vermiştir.

Oylama: Sınıflandırıcıların verdikleri sınıf kararlarından en yüksek frekanslının komitenin kararı olarak belirlendiği metottur. Basitliği sebebiyle en yaygın kullanılan metotlardan biridir.

Ağırlıklandırma: Sınıflandırma kararlarının çeşitli şekillerde ağırlıklandırılarak komitenin kararını oluşturduğu metottur. Literatürde çeşitli versiyonları yer almaktadır.

Adaboost: Bu metotta bir sınıflandırıcının ağırlığı (komite kararına etkisi) kendi hata değeriyle belirlenir. Diğer bir ifadeyle performansı yüksek olan algoritmaların kararlarına, düşük olanlardan daha fazla önem (ağırlık) verilir. Sınıflandırıcıların eğitildiği eğitim kümelerinin oluşturulması işleminde Bagging'den ayrılmaktadır. Bagging'de tüm örneklerin eğitim kümesine seçilme şansı eşittir. Adaboost'ta ise önceki sınıflandırıcılar tarafından yanlış sınıflandırılmış bir örneğin seçilme olasılığı diğerlerinden yüksektir. Bu ilk sınıflandırıcıların kolaylıkla sınıflandırılabilen örneklerle, sonrakilerin ise sınıflandırması zor olan örneklerle yoğunlaşmasını sağlamaktadır (Freund vd., 1997).

Algoritmada kullanılan semboller:

Veri kümesi $A = \{X_j, C_j\}_{j=1}^N, X_j \in \mathbb{R}^p$

$j \rightarrow$ örnek indisi

$L \rightarrow$ komitedeki sınıflandırıcı sayısı

$l \rightarrow$ sınıflandırıcı indisi

$H(x_j) \rightarrow$ j. örneğin H sınıflandırıcısına göre sınıfı

AdaBoost Algoritması:

Örneklerin eğitim kümesine seçilme ağırlıklarına (sw) ilk değerlerini ata (hepsi eşit).

$$sw_1(j) = 1/N, j = 1, \dots, N \quad (4.1)$$

L adet sınıflandırıcı oluştur.

Döngü $l=1, \dots, L$:

1. sw_i ağırlıklarına göre veri kümesinden eğitim kümesi A_l oluştur.
2. A_l kümesi üzerinde H_l sınıflandırıcısını eğit.
3. H_l sınıflandırıcısının tüm veri kümesi (A) üzerindeki ağırlıklı hatasını (e_l) bul.

$$e_l = \frac{1}{N} \sum_j sw_j farkli(H_l(x_j), C_j) , farkli(e1, e2) = \begin{cases} 0 & \leftarrow e1 = e2 \\ 1 & \leftarrow e1 \neq e2 \end{cases} \quad (4.2)$$

Yanlış sınıflandırılmış örneklerin seçilme ağırlıklarını toplamı, sınıflandırıcının hatasını oluşturuyor. Bu nedenle seçilme olasılığı yüksek olan, (daha önceki sınıflandırıcıların doğru sınıflandıramadığı) örneklerde yapılan hatanın etkisi, diğer örneklere göre daha fazla oluyor.

4. Eğer $e_l > 0.5$ ise tüm sw 'leri ilk değerlerine ata ve döngünün başına dön.
5. Sınıflandırıcıların önem derecelerini (α_l) hatalarına bağlı olarak hesapla.

$$\alpha_l = 0.5 \ln \frac{1 - e_l}{e_l} \quad (4.3)$$

6. Örneklerin seçilme ağırlıklarını güncelle:

$$sw_j^{(l+1)} = \frac{sw_j^{(l)}}{Z_l} * \begin{cases} \exp^{-\alpha_l} & \leftarrow H_l(x_j) = C_j \\ \exp^{\alpha_l} & \leftarrow H_l(x_j) \neq C_j \end{cases} \quad (4.4)$$

Z_l , normalizasyon katsayısıdır. $sw_j^{(l+1)}$ 'lerin toplamının 1 kalmasını sağlar.

AdaBoost ile oluşturulan L adet sınıflandırıcının kararlarının birleştirilmesinde Eşitlik 4.5 kullanılır.

$$H^*(x_j) = \arg \max_g \sum_{l=1}^L \alpha_l * esit(H_l(x_j), g) \quad (4.5)$$

Eşitlikteki g olası tüm sınıfları göstermektedir. Bu şekilde komitenin kararı, içindeki sınıflandırıcıların kararlarının, önemleriyle ağırlıklandırılmasıyla elde edilmiş olur.

Adaboost'un Yapısı sebebiyle, her bir sınıflandırıcı kendinden önceki sınıflandırıcıların deneyimlerinden faydalandığından sınıflandırıcıların paralel olarak oluşturulması mümkün değildir.

4.2 Komitelerin Performansları Neden Yüksekler?

Sınıflandırıcıların her birinin veri kümesi üzerinde %50'den büyük bir performansı varsa, ve kararların birbirlerinden bağımsız oldukları kabul edilirse, kararlar oylama tekniği ile birleştirildiğinde sınıflandırma komitesinin performansı içinde sınıflandırıcılardan yüksek olur [15].

Örneğin her birinin sınıflandırma başarısı 0.8 olan 3 sınıflandırıcının kararlarının birleştirildiği düşünülürse:

3 sınıflandırıcının toplam 8 farklı olası kararı vardır. Sınıflandırıcı komitesinin doğru karar verebilmesi için olası kararların içinde doğru kararların yanlış kararlardan fazla olması gerekir. Çizelge 4.1'de sınıflandırıcıların kararlarının ve bunlara karşılık gelen komite kararının olasılıkları verilmiştir.

Çizelge 4.1 Üç farklı sınıflandırıcının kararlarının birleştirilmesi

3 sınıflandırıcının Olası kararları	Komitenin kararı	Kararın olasılığı
1 1 1	1	$0.512 = 0.8*0.8*0.8$
1 1 0	1	$0.128 = 0.8*0.8*0.2$
1 0 1	1	$0.128 = 0.8*0.2*0.8$
0 1 1	1	$0.128 = 0.2*0.8*0.8$
0 0 0	0	$0.008 = 0.2*0.2*0.2$
0 0 1	0	$0.032 = 0.2*0.2*0.8$
0 1 0	0	$0.032 = 0.2*0.8*0.2$
1 0 0	0	$0.032 = 0.8*0.2*0.2$

Çizelge 4.1'deki 1'ler sınıflandırıcı/komite kararının doğruluğunu, 0'lar yanlışlığını göstermektedir. Komitenin doğru karar verme olasılığı 0.896 ($0.512 + 0.128 + 0.128 + 0.128$)'dır ve her bir tekil sınıflandırıcının performansından (0.8) yüksektir. Buradaki olasılıkların toplanma sebebi sınıflandırıcıların kararlarının birbirinden bağımsız olduğu kabulüdür.

Çizelge 4.2'de sınıflandırıcı sayısının ve sınıflandırıcıların performanslarının komite kararının doğruluğu üzerindeki etkisi verilmiştir [6].

Çizelge 4.2 Sınıflandırıcıların kararlarının birleştirilmesinde sınıflandırıcı sayılarının ve performanslarının etkisi

Sınıflandırıcı performansı	Sınıflandırıcı sayısı L=1	L=3	L=5	L=7	L=9
0.4	0.4	0.352	0.317	0.29	0.267
0.51	0.51	0.515	0.519	0.522	0.525
0.6	0.6	0.648	0.683	0.71	0.73
0.7	0.7	0.784	0.837	0.874	0.901
0.8	0.8	0.896	0.942	0.967	0.98

Çizelge 4.2 incelendiğinde, performansı 0.5’den büyük olan sınıflandırıcıların birleştirilmesinde, sınıflandırıcı sayısının artmasının ve tekil sınıflandırıcıların tek başlarına performanslarının komite kararı üzerinde olumlu bir etkisi olduğu görülmektedir. Birleştirilen sınıflandırıcı sayısı arttıkça komite performansı 1’e doğru yaklaşacaktır. Performansı 0.5’den küçük olan sınıflandırıcıların birleştirilmesi ise olumlu bir sonuç vermemektedir.

Sınıflandırıcı komitelerinin iyi sonuçlar üretebilmelerinin bir diğer ön şartı sınıflandırıcı kararlarının birbirlerinden olabildiğince bağımsız olmasıdır. Bu bağımsızlık literatürde çeşitli metotlarla arttırılmıştır.

4.3 Önceki Çalışmalar

Sınıflandırma komiteleri oluşturma metotlarının karşılaştırılması amacıyla literatürde birçok çalışma yapılmıştır. Bu bölümde bu çalışmaların bir kısmından ve elde ettikleri sonuçlardan bahsedilmektedir.

“A comparison of Ensemble Creation Techniques” adlı çalışmada (Banfield vd., 2004), 34 UCI veri kümesi üzerinde komite oluşturma tekniklerini (Bagging, Boosting, Rasgele Alt Uzaylar, Rasgele Alt Uzaylarda Bagging) karşılaştırmıştır. Komitelerde yer alan sınıflandırıcılar C4.5 karar ağaçlarıdır. Diğer komite oluşturma tekniklerinin Bagging’den istatistiki olarak daha iyi olmadığı sonucuna varmışlardır. Bununla birlikte en yüksek performans Rasgele Alt Uzaylarda Bagging tekniği ile elde etmişlerdir.

“An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, Randomization” adlı çalışmada (Dietterich, 1999) 33 UCI veri kümesi üzerinde bagging, boosting, randomization karşılaştırılmıştır. Komitelerde yer alan sınıflandırıcılar C4.5 karar ağaçlarıdır. Yeni bir teknik olarak kullanılan Randomization’da

karar ağaçları oluşturulurken en iyi 20 özellikten biri rasgele seçilerek karar düğümü oluşturulmaktadır. Metotların karşılaştırmaları sonucunda orijinal veri kümelerinde Boosting'in Bagging ve Randomization 'dan daha başarılı olduğu, sınıf gürültüsü eklenmiş verilerde ise Bagging en iyi performansı gösterdiği belirtilmiştir.

“A Comparison of Ensemble Methods for Microarray Data Analysis” adlı çalışmada [7] Adabost, Bagging, BagBoost komite oluşturma teknikleri DNA veri kümesi üzerinde karşılaştırılmıştır. BagBoost, Bagging ve AdaBoost tekniklerinin birlikte uygulanmasına verilen addır. Sonuç olarak Adaboost'un en iyi, Bagging'in en kötü performansa sahip olduğu bulunmuştur.

“Random forests-random features” adlı çalışmada yer alan Random Forest (Breiman, 1999) son zamanlarda üstün başarısıyla öne çıkan bir algoritmadır. Algoritmanın temel fikri tek bir ağaç oluşturmak yerine bir karar ormanı oluşturmaktır. Karar ormanı her biri farklı eğitim kümeleriyle eğitilmiş çok sayıda çok değişkenli karar ağacından oluşur. Farklı eğitim kümeleri orijinal setten rasgele örnek (bootstrap) ve rasgele özellik seçimiyle oluşturulur. Test örneği oluşturulan tüm sınıflandırıcılarla sınıflandırılır. Sınıflandırıcıların en fazla karar verdikleri sınıf test örneğinin sınıfı olarak belirlenir. Ormanın başarısı, ağaçların her birinin başarısına ve ağaçların birbirinden bağımsızlığına bağlıdır.

4.4 Gerçekleştirilenler

Bu çalışmada oylama metodu kullanılarak çeşitli Cline algoritmalarının sonuçları birleştirilmiş ve tekil Cline ağaçlarından daha iyi sonuçlar elde edilmiştir.

CLForest: Breiman'ın karar ormanları fikri kullanılarak, cline metotlarının orman versiyonları geliştirilmiş ve Cline algoritma ailesine eklenmiştir. Cline Ormanları ve Random Forest arasındaki temel fark karar ağacı oluşturma algoritmasıdır. Random Forest, Classification and Regression Tree (CART) (Breiman vd., 1984) kullanırken Cline Ormanları Cline metotlarını kullanırlar.

Ağaç Kararlarını Ağırlıklandırma: Karar ağaçlarının verdikleri kararlara eşit ağırlık vermek yerine ağaçların çeşitli özelliklerine göre kararları ağırlıklandırılmıştır. Ağaçların kararlarını ağırlıklandırmada kullanılan özellikleri aşağıda verilmiştir:

- Doğrulama-validation setindeki en başarılı X tane ağaç kullanılarak: Eğitim verisinden bir kısmı doğrulama verisi olarak ayrılmış ve ormandaki her ağacın bu doğrulama setindeki başarısı ölçülmüştür. Ormanın kararı, en başarılı X tanesinin kararının

ortalamasıdır.

- Ağaçtaki karar düğümü sayısı ile doğru/ters orantılı olarak kararını ağırlıklandırmak: Ormandaki her ağacın karar düğümü sayısı bulunmuş ve ağaçların kararları normalize edilmiş karar düğümü sayısı ile ya da tersiyle çarpılarak ağaçların kararları ağırlıklandırılmıştır.

4.5 Deneysel Sonuçlar

Karar ağaçları yerine karar ormanlarının kullanımının performansa etkisinin ölçülmesi amacıyla Bölüm 3'teki 2 ayrı veri kümesi koleksiyonu (UCI ve İlaç) kullanılmıştır. Bu koleksiyonlarda yer alan veri kümelerinin ayrıntılarına Bölüm 3.4.1 ve 3.4.2'den erişilebilir.

4.5.1 UCI Veri Kümeleri Üzerindeki Sonuçlar

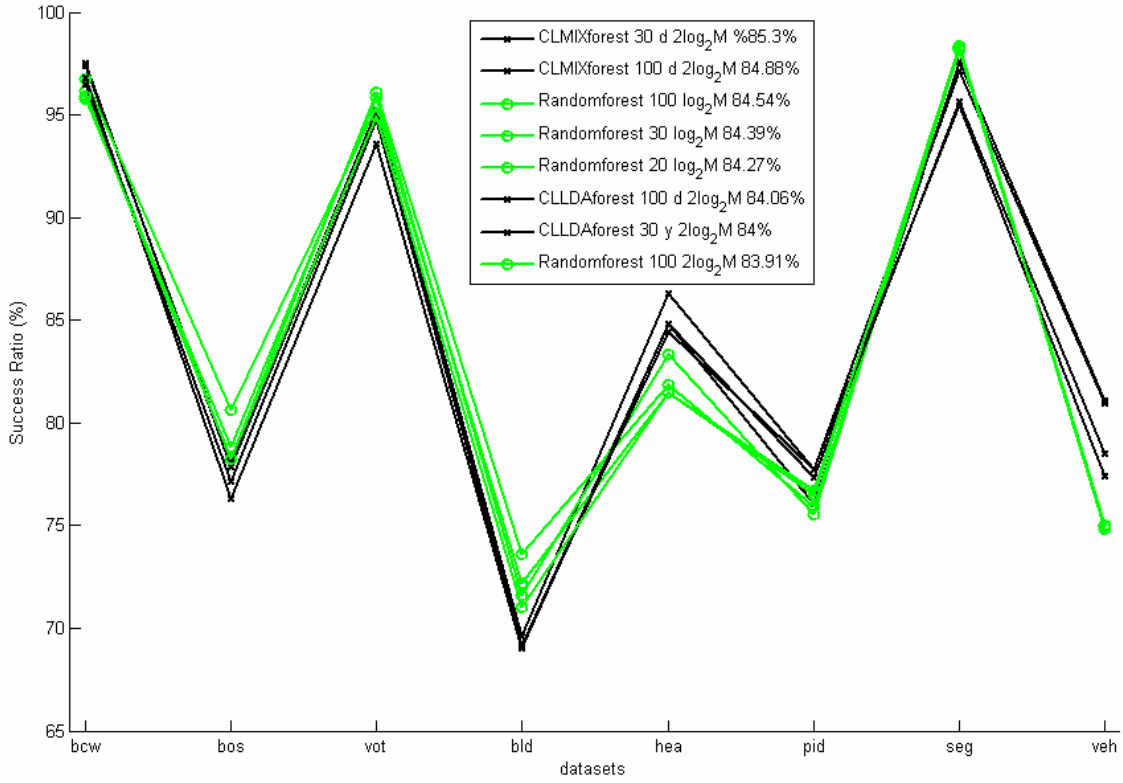
Tek ağaçla en iyi sonuçları veren 4 farklı Cline versiyonunun orman versiyonları gerçekleştirilmiş ve sonuçları Random Forest ile karşılaştırılmıştır. Random Forest'ta budama işlemi olmadığından Cline Forest metodlarında da budama uygulanmamıştır. Sonuçta 64 (4 hiper düzlem belirleme * 4 farklı ormandaki ağaç sayısı * 2 farklı ağaçlardaki özellik sayısı * 2 farklı sınıflandırma türü -yaprak / dal-) adet Cline ormanı, 8 (4 farklı ormandaki ağaç sayısı * 2 farklı ağaçlardaki özellik sayısı) adet Random Forest versiyonu Çizelge 3.4'teki 8 farklı veri kümesi üzerinde 10'lu çapraz geçerleme ile çalıştırılmış ve ortalama başarıları Çizelge 4.3'te verilmiştir. Cline ormanları ve Random Forest'ları oluşturmak için 4 farklı sayıda ağaç sayısı, Cline ve Random ormanlarının her ağacında da 2 farklı özellik sayısı kullanılmıştır. Ayrıca Cline ormanlarında ağaçlar dal ve yaprak kullanarak iki farklı şekilde oluşturulmuştur. Çizelge 4.3'te M verilerin orijinal boyutunu göstermektedir. Burada ortalaması verilen değerlerin hepsi ve standart sapmaları Çizelge 4.4'ten görülebilir.

Çizelge 4.3 Cline ve rasgele orman versiyonlarının ortalama başarıları

		CLM Forest	CLLDA Forest	CLLVQ Forest	CLMIX Forest	Random Forest
Ağaç Sayısı	10	0.8129	0.8289	0.8137	0.8265	0.8280
	20	0.8234	0.8253	0.8218	0.8303	0.8395
	30	0.8227	0.8239	0.8234	0.8356	0.8409
	100	0.8282	0.8252	0.8285	0.8376	0.8423
Ağaçlardaki Özellik Sayısı	$\text{Log}_2(M)$	0.8184	0.8146	0.8175	0.8238	0.8401
	$2\text{log}_2(M)$	0.8252	0.8370	0.8262	0.8412	0.8352
Örneklerin sınıflandırılma Türü	Yaprak	0.8190	0.8249	0.8212	0.8285	---
	Dal	0.8246	0.8267	0.8225	0.8365	

Cline algoritmalarında ve Random Forestlarda da ağaç sayısı arttıkça başarı da artmaktadır. Buna dayanılarak Breiman'ın çalışmasında (Breiman, 1999) belirttiği gibi ağaç sayısını arttırmanın aşırı eğitime (overfitting) sebep olmadığı söylenebilir. Özellik sayısının artması Random Forest'ta başarıyı azaltmışken, Cline versiyonlarında arttırmıştır. Cline Orman versiyonlarında, tek ağaçlı Cline versiyonlarında olduğu gibi dal kullanımı yaprak kullanımına göre daha yüksek bir başarıya sahiptir.

Şekil 4.1'de Cline Forest ve Random Forest'ın 8 farklı versiyonunun her birinin veri kümesindeki başarıları gösterilmektedir.



Şekil 4.1 En başarılı 8 Cline ve rasgele orman versiyonlarının karşılaştırılması

Şekil 4.1’de koyu renkli çizgiler Cline Forest, açık renkli çizgiler ise Random Forest versiyonlarına aittir. UCI veri kümelerini karar ormanlarıyla (Random Forest, Cline Forest) sınıflandırma denemelerinden aşağıdaki sonuçlar elde edilmiştir:

- Şekil 3.9 ve 4.1 karşılaştırıldığında tek ağaç yerine birden fazla ağaçtan oluşan ormanları kullanmanın daha iyi sonuçlar verdiği görülmektedir. Tek ağaçlı algoritmaların en başarılısı olan QUEST %84.11’lik başarıya sahipken, 30 ağaçtan oluşan Cline Ormanı %85.3’lük başarı göstermiştir.
- Şekil 4.1’te görüldüğü gibi en başarılı iki algoritma Cline ailesindendir. Cline ormanları bcw, hea, ve veh kodlu veri kümelerinde daha iyiysen, bld ve seg veri kümelerinde Random Forest’lar daha başarılı sonuçlar vermiştir. Ağaçlarında Cline metotlarını kullanan Cline Forest’ın, ağaçlarında CART algoritmasını kullanan Random Forest’tan daha başarılı versiyonlarının olması normaldir. Çünkü Şekil 3.9’da görüldüğü gibi tek ağaçlı algoritmalarda CLMIX, CART (IC0, IC1)’tan daha yüksek başarı elde etmiştir. Diğer bir ifadeyle, daha başarılı algoritmaların birleştirilmesi daha iyi sonuçlar vermiştir. Bu sonuçtan yola çıkılarak QUEST gibi tek ağaçlı

algoritmaların orman versiyonlarının da orijinallerinden daha başarılı olacağı söylenebilir.

Boostrapping'in Performansa Etkisi:

Cline'in en yakın rakibi olan Random Forest bagging metodunu kullanmaktadır (Breiman, 2001). Bu sebeple bagging metodunun Cline karar ormanlarına etkisi de incelenmiştir. Çizelge 4.4'te bootstrapping eklenmiş ve eklenmemiş Cline karar ormanlarının diğer sınıflandırıcılarla 8 veri kümesi üzerindeki karşılaştırmalı sonuçları verilmiştir. Çizelgedeki algoritma kısaltmalarının uzun halleri, açıklamaları ve gerekli referanslar Çizelge 3.6'da yer almaktadır. Çizelgedeki Cline algoritmalarının isimlerindeki ilk rakam ormandaki ağaç sayısını, sonraki parametre test örneklerinin sınıfının belirlenmesinde yaprak ya da dal kullanımını, daha sonraki parametre ise ormandaki her bir ağaçta kullanılan özellik sayını göstermektedir. Sonunda "BS" ifadesi yer alan algoritmalarda Bootstrap tekniği kullanılmıştır.

Çizelge 4.4 Bootstrapping'in performansa etkisi

Dataset ismi	bcw	bos	Vot	bld	hea	pid	seg	veh	Ortalama
CLM_forest_10_yaprak_log2	0.9693	0.7415	0.9221	0.6052	0.7926	0.7205	0.9515	0.7208	0.8029375
CLM_forest_10_yaprak_log2_BS	0.962	0.7511	0.9264	0.67	0.8	0.7118	0.9485	0.7069	0.8095875
CLM_forest_10_yaprak_2log2	0.9722	0.7787	0.9409	0.61	0.8296	0.7165	0.9688	0.74	0.8195875
CLM_forest_10_yaprak_2log2_BS	0.9708	0.7524	0.952	0.6471	0.8185	0.7505	0.9645	0.7186	0.8218
CLM_forest_10_dal_2log2	0.9707	0.7846	0.9105	0.631	0.8185	0.7551	0.9597	0.7355	0.8207
CLM_forest_10_dal_2log2_BS	0.9679	0.7531	0.927	0.6524	0.8519	0.7671	0.9545	0.7129	0.82335
CLM_forest_20_dal_2log2	0.9708	0.7982	0.95	0.631	0.8519	0.7473	0.9658	0.7494	0.83305
CLM_forest_20_dal_2log2_BS	0.9751	0.7709	0.9471	0.7	0.8333	0.7603	0.9623	0.7266	0.83445
CLM_forest_100_yaprak_2log2	0.9722	0.7825	0.9545	0.61	0.8333	0.7205	0.9762	0.7528	0.82525
CLM_forest_100_yaprak_2log2_BS	0.9708	0.7789	0.9568	0.6933	0.837	0.7591	0.9723	0.7494	0.8397
CLM_forest_100_dal_2log2	0.9737	0.7862	0.9523	0.631	0.8407	0.7477	0.9684	0.7505	0.8313125
CLM_forest_100_dal_2log2_BS	0.9737	0.7787	0.9523	0.691	0.837	0.7613	0.9706	0.7339	0.8373125
LVQ_forest_30_dal_log2	0.9737	0.7709	0.9176	0.6724	0.8333	0.761	0.9511	0.6989	0.8223625
LVQ_forest_30_dal_log2_BS	0.9707	0.763	0.9153	0.6814	0.8222	0.7603	0.9532	0.6995	0.8207
LVQ_forest_100_yaprak_log2	0.9766	0.7689	0.9338	0.6524	0.8222	0.7468	0.9632	0.7363	0.825025
LVQ_forest_100_yaprak_log2_BS	0.9751	0.7572	0.9179	0.7019	0.8333	0.7594	0.9654	0.7387	0.8311125
LVQ_forest_100_yaprak_2log2	0.9737	0.7924	0.9523	0.6429	0.8296	0.7357	0.9723	0.7481	0.830875
LVQ_forest_100_yaprak_2log2_BS	0.9752	0.7867	0.95	0.6705	0.8296	0.7631	0.9732	0.7481	0.83705
LVQ_forest_100_dal_2log2	0.9737	0.7846	0.9455	0.6457	0.8407	0.7567	0.9628	0.727	0.8295875
LVQ_forest_100_dal_2log2_BS	0.9722	0.7804	0.9315	0.6938	0.8407	0.7671	0.9654	0.7233	0.8343
LVQ_forest_100_dal_log2	0.9737	0.7746	0.9179	0.6662	0.837	0.7721	0.9576	0.7305	0.8287
LVQ_forest_100_dal_log2_BS	0.9751	0.763	0.9179	0.701	0.8296	0.761	0.9545	0.728	0.8287625
CLMIX_forest_10_dal_log2	0.9663	0.7513	0.9111	0.7	0.8037	0.7634	0.9481	0.7373	0.82265
CLMIX_forest_10_dal_log2_BS	0.9634	0.7434	0.9059	0.6314	0.8259	0.7332	0.9429	0.7276	0.8092125
CLMIX_forest_10_yaprak_log2	0.9693	0.7668	0.9193	0.6457	0.7815	0.7227	0.9567	0.7449	0.8133625
CLMIX_forest_10_yaprak_log2_BS	0.9677	0.7549	0.9241	0.6157	0.8148	0.7295	0.9463	0.7103	0.8079125

CLMIX_forest_20_dal_2log2	0.9707	0.7923	0.9361	0.6648	0.8222	0.7798	0.9766	0.8011	0.84295
CLMIX_forest_20_dal_2log2_BS	0.9707	0.7491	0.9477	0.7105	0.8556	0.7746	0.9688	0.792	0.846125
CLMIX_forest_30_yaprak_2log2	0.9722	0.7903	0.954	0.6367	0.8481	0.744	0.9766	0.8072	0.8411375
CLMIX_forest_30_yaprak_2log2_BS	0.9678	0.7825	0.95	0.7143	0.837	0.748	0.9771	0.8002	0.8471125
CLMIX_forest_30_dal_log2	0.9692	0.765	0.9179	0.681	0.8296	0.7548	0.9476	0.7268	0.8239875
CLMIX_forest_30_dal_log2_BS	0.9707	0.7632	0.9247	0.7048	0.8222	0.7668	0.9597	0.7505	0.832825
CLMIX_forest_30_dal_2log2	0.9737	0.7807	0.9523	0.6962	0.863	0.7773	0.9714	0.8095	0.8530125
CLMIX_forest_30_dal_2log2_BS	0.9751	0.7885	0.929	0.7429	0.8519	0.782	0.9697	0.7992	0.8547875
CLMIX_forest_100_dal_2log2	0.9751	0.7786	0.95	0.6914	0.8481	0.7606	0.9758	0.8109	0.8488125
CLMIX_forest_100_dal_2log2_BS	0.9722	0.773	0.9474	0.7276	0.8519	0.7705	0.9706	0.8048	0.85225
CLMIX_forest_100_dal_log2	0.9722	0.773	0.9176	0.6962	0.8407	0.7643	0.9623	0.7716	0.8372375
CLMIX_forest_100_dal_log2_BS	0.9663	0.7728	0.9202	0.7414	0.8333	0.7625	0.9576	0.7564	0.8388125
CLMIX_forest_100_yaprak_2log2	0.9737	0.8006	0.9545	0.6386	0.8407	0.7458	0.974	0.7551	0.835375
CLMIX_forest_100_yaprak_2log2_BS	0.9737	0.7924	0.9523	0.731	0.837	0.7838	0.9745	0.807	0.856452
CLMIX_forest_100_yaprak_log2	0.9737	0.7648	0.9224	0.6929	0.8259	0.744	0.9675	0.741	0.829025
CLMIX_forest_100_yaprak_log2_BS	0.9766	0.7791	0.9338	0.6876	0.8481	0.7554	0.9675	0.7552	0.8379125
QU0	0.962	0.733	0.9588	0.611	0.774	0.774	0.9584	0.695	0.808275
QU1	0.9546	0.727	0.9565	0.601	0.756	0.77	0.9528	0.678	0.7994875
QLO	0.9692	0.732	0.9636	0.694	0.848	0.775	0.9541	0.793	0.8411125
QL1	0.9692	0.726	0.9497	0.68	0.848	0.777	0.9524	0.794	0.8370375
POL	0.9576	0.775	0.9477	0.714	0.826	0.766	0.9675	0.838	0.848975
Rfdeki ozellik sayisi (2log2)	8	8	10	6	8	8	10	10	
RF-10	0.9531	0.7786	0.9494	0.6753	0.8037	0.7462	0.9774	0.7375	0.82765
RF-20	0.9575	0.7845	0.9517	0.6927	0.8148	0.7631	0.9809	0.7446	0.836225
RF-30	0.9619	0.7865	0.9494	0.6985	0.8074	0.7631	0.9822	0.7529	0.8377375
RF-100	0.959	0.7826	0.9494	0.7101	0.8148	0.7669	0.9818	0.7482	0.8391
Rfdeki ozellik sayisi (log2)	4	4	5	3	4	4	5	5	
RF-10	0.9619	0.7569	0.9494	0.6782	0.8185	0.7424	0.98	0.73995	0.8284063
RF-20	0.9575	0.7885	0.9586	0.7159	0.8333	0.7556	0.9822	0.7505	0.8427625
RF-30	0.9619	0.7826	0.9609	0.7362	0.8185	0.7593	0.9831	0.7494	0.8439875
RF-100	0.9677	0.8063	0.954	0.7217	0.8148	0.765	0.9835	0.7505	0.8454375

Çizelge 4.5'te ise Çizelge 4.3'e bir sütun (CLMIX_BS) eklenerek Çizelge 4.4'ün özetlenmiş hali verilmiştir.

Çizelge 4.5 Cline ormanlarının ve Random Forest’ların farklı versiyonlarının ortalama başarıları ve standart sapmaları

		CLM Forest	CLLDA Forest	CLLVQ Forest	CLMIX Forest	CLMIX ForestBS	Random Forest
Ormandaki ağaç sayısı	10	81.29 ±0.87	82.89 ±1.26	81.37 ±1.15	82.65 ±1.23	81.93 ±1.77	82.80 ±1.13
	20	82.34 ±0.74	82.53 ±1.35	82.18 ±0.52	83.03 ±1.21	83.16 ±1.6	83.95 ±1.06
	30	82.27 ±0.4	82.39 ±1.74	82.34 ±0.43	83.56 ±1.41	83.64 ±1.35	84.09 ±1.05
	100	82.82 ±0.27	82.52 ±1.43	82.85 ±0.25	83.76 ±0.83	84.64 ±1.03	84.23 ±1.03
Ağaçların karar düğümünde kullanılan özellik sayısı	$\log_2 M$	81.84 ±0.91	81.46 ±0.88	81.75 ±0.93	82.38 ±0.71	82.13 ±1.29	84.01 ±1.05
	$2\log_2 M$	82.52 ±0.51	83.70 ±0.26	82.62 ±0.37	84.12 ±0.79	84.56 ±0.9	83.52 ±1.06
Test örneklerinin sınıflandırılmasında kullanılan yöntem	Yaprak	81.90 ±0.75	82.49 ±1.38	82.12 ±1.01	82.85 ±0.96	83.22 ±1.78	---
	Dal	82.46 ±0.79	82.67 ±1.33	82.25 ±0.63	83.65 ±1.26	83.47 ±1.62	

Çizelge 4.5 incelendiğinde Cline karar ormanlarına bagging eklenmesinin hemen hemen her veri kümesinde ve sonuç olarak, ortalama başarıda performansı arttırdığı görülmektedir. Bu metodun eklenmesiyle Cline karar ormanlarıyla Random Forest’lar arasındaki fark daha da açılmıştır.

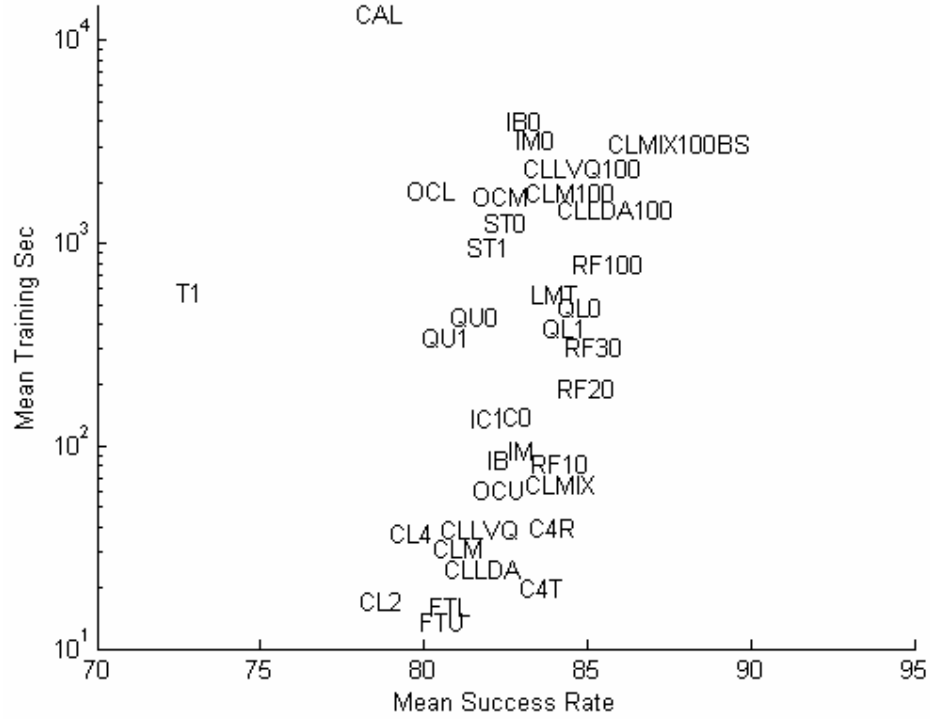
Çizelge 4.6’da, en başarılı 4 Cline ormanı, en başarılı 4 Random Forest ve diğer 22 karar ağacı algoritmaları, her bir veri kümesi için, maksimum başarının standart sapma kadar yakınında yer alanları ‘*’ ile en kötü başarıya sahip olanları ‘?’ ile işaretlenmiştir. 10. ve 11. sütunlarda ise ortalama başarılar ve standart sapmaları verilmiştir.

Çizelge 4.6 Algoritmaların her bir veri kümesinde maksimum başarının standart sapma miktarı yakınında kalanları

Algoritmalar	Veri Kümeleri								Ortalama başarı	Ortalama standart sapma	# of *	# of ?
	bcw	bos	vot	bld	hea	pid	seg	veh				
CLMIX_forestBS_100_y_2log ₂ M	*	*		*	*	*	*	*	85.65	3.50	7	0
CLMIX_forestBS_100_d_2log ₂ M	*			*	*	*	*	*	85.42	3.53	6	0
CLMIX_forestBS_30_d_2log ₂ M	*	*		*	*	*	*	*	84.89	3.93	7	0
CLMIX_forestBS_30_y_2log ₂ M	*	*		*	*	*	*	*	84.69	3.73	7	0
Random Forest_100_log ₂ M	*	*		*	*	*	*	*	84.54	4.43	7	0
Random Forest_30_log ₂ M	*	*		*	*		*	*	84.33	4.41	6	0
QLO	*		*		*	*	*	*	84.11		6	0
Random Forest_20_log ₂ M	*	*	*	*			*	*	84.10	4.30	6	0
Random Forest_100_2log ₂ M		*		*		*	*	*	83.92	4.25	5	0
QL1	*				*	*	*	*	83.70		5	0
LMT	*				*		*	*	83.45		4	0
C4R	*			*		*	*		83.21		4	0
C4T		*					*		82.92		2	0
IM0	*		*				*		82.77		3	0
IB0	*		*				*		82.55		3	0
IM						*	*		82.33		2	0
IC0							*		82.24		1	0
IB							*		81.97		1	0
ST0							*	*	81.85		2	0
OCM			?	*			*		81.76		2	1
OCU							*		81.52		1	0
IC1							*		81.44		1	0
ST1							*		81.39		1	0
QU0	*		*			*	*		80.83		4	0
FTL					*	*	*		80.06		3	0
FTU							*		79.96		1	0
QU1						*	*		79.95		2	0
OCL	?			*		?	*		79.49		2	2
CAL					?	*			77.93		1	1
T1		?		?	?		?	?	72.43		0	5

Çizelge 4.6 incelendiğinde Cline ormanlarının diğer tüm algoritmalarından daha fazla en iyi algoritmalar içinde yer aldığı görülmektedir. Ayrıca standart sapma miktarı da Random Forest'lardan daha düşüktür.

Sınıflandırma performansı kriterine ek olarak Cline algoritmalarının diğer algoritmalarla karmaşıklık kriterine göre de karşılaştırılması için çalışma yapılmış ve Şekil 4.2 elde edilmiştir. Şekildeki x boyutu eğitim ve test toplam süresini saniye cinsinden (logaritmik olarak), y boyutu sınıflandırma performansını göstermektedir.



Şekil 4.2 Algoritmaların karmaşıklıklarıyla performansları arasındaki ilişki

Şekil 4.2 incelendiğinde en yüksek başarıya sahip algoritma olan CLMIX'in en yavaş algoritma olmadığı bununla birlikte algoritmaların karmaşıklıklarıyla performansları arasında pozitif bir ilişki olduğu görülmektedir.

4.5.2 İlaç Verilerindeki Sonuçlar

Çizelge 4.7'de çeşitli Cline ormanı versiyonlarının ilaç veri kümesi üzerindeki performansları verilmiştir. Çizelgedeki Random Forest algoritmalarının isimlerindeki ilk rakam ormandaki ağaç sayısını, sonraki parametre ormandaki her bir ağaçta kullanılan özellik sayını göstermektedir. Cline algoritmalarının isimlerindeki ilk rakam ormandaki ağaç sayısını, sonraki parametre test örneklerinin sınıfının belirlenmesinde yaprak ya da dal kullanımını, daha sonraki parametre ise ormandaki her bir ağaçta kullanılan özellik sayını göstermektedir. Sonunda “ağırlıklı” ifadesi yer alan algoritmalarda ağaçların kararları, ağaçların düğüm sayılarıyla doğru orantılı olarak ağırlıklandırılarak birleştirilmiştir. Diğerlerinde ise geleneksel

oylama tekniği kullanılmıştır.

Çizelge 4.7 İlaç veri kümelerinde Cline ile karşılaştırılan diğer algoritmaların sonuçları

Algoritma	Bbp2	Mono	Clean1	Ca	Mutag	232	Ort	Rank
RandomForest-10-8	82.17	91.95	86.76	68.42	87.76	56.89	78.99	16.33
RandomForest-20-8	82.17	93.1	87.61	71.92	88.82	57.32	80.16	11.83
RandomForest-30-8	81.92	93.79	90.13	70.37	88.82	58.18	80.54	10
RandomForest-100-8	81.78	93.29	91.03	71.97	90.82	59.87	81.46	5
RandomForest-10-16	79.61	92.37	86.89	69.46	90.39	57.85	79.43	15
RandomForest-20-16	81.08	93.42	89.12	70.87	90.49	58.57	80.59	9.33
RandomForest-30-16	80.94	93.65	89.65	71.08	90.82	59.65	80.97	7.5
RandomForest-100-16	81.32	93.86	90.61	71.77	90.45	59.36	81.23	6.17
CLMforest_100_yaprak_16	80.13	93.1	91.2	72.52	89.39	57.64	80.66	9.67
CLMIXforest_30_yaprak_16	79.04	91.95	89.47	70.9	90.47	59.72	80.26	12
CLMIXforest_100_yaprak_16	79.89	93.1	89.66	72.01	89.77	57.12	80.26	11.33
CLMforest_100_dal_8	82.65	91.47	90.15	68.79	89.39	58.23	80.11	11.17
CLMforest_100_dal_16	80.86	91.5	91.82	71.08	89.36	55.75	80.06	12.67
CLMforest_20_dal_16	82.14	90.54	87.38	70.9	88.83	57.68	79.58	14
CLMforest_100_yaprak_8	80.2	92.6	90.74	70.49	87.19	56.01	79.54	15.33
CLMIXforest_20_yaprak_16	80.5	92.89	88.8	67.68	88.86	58.12	79.48	14.67
CLMforest_30_dal_16	80.22	91.74	89.68	72.72	87.78	54.34	79.41	14.33
CLMIXforest_100_yaprak_8	79.52	93.14	88.43	69.99	89.36	56.01	79.41	15.33
CLLVQ4_forest_100_yaprak_16 (ağırlıklı)	81.2	93.35	92.29	72.12	90.5	59.25	81.45	5
CLLVQ_forest_100_yaprak_16 (ağırlıklı)	81.41	93.23	91.36	71.2	88.25	56.09	80.26	10.5
CLM_forest_100_yaprak_16 (ağırlıklı)	83.41	93.81	91.16	73.76	89.25	60.47	81.98	3.83

Algoritmalar karşılaştırılırken 2 yöntem kullanılmıştır. İlki tüm veri kümelerindeki performansların ortalamalarıdır ve çizelgede “ort” sütununda verilmiştir. İkincisi ise algoritmaların her veri kümesinde başarı sıralamalarının ortalamasıdır ve “rank” sütununda verilmiştir. Görüldüğü gibi her iki yöntemde de en başarılı metot “CLM_forest_100_yaprak_ağırlıklı”dır.

İlaç verileriyle yapılan çalışmalar sonucunda aşağıdaki sonuçlara ulaşılmıştır:

- Tek bir ağaç kullanmak yerine bir orman kullanmak, UCI veri kümesindeki sonuçlara paralel olarak, daha başarılı sonuçlar üretmektedir.
- En yüksek başarı, 100 ağaçlı CLM ormanı ile kararlar ağaçtaki düğüm sayısı ile doğru

orantılı bir şekilde ağırlıklandırıldığında elde edilmiştir.

- Ormandaki ağaç sayısı arttıkça başarı artmaktadır.
- Ağaçlardaki özellik sayısı arttıkça başarı artmaktadır.

4.6 Çıkarımlar

Sınıflandırma problemlerine karar ağaçları yerine karar ormanları ile çözüm bulma denemeleri 2 veri koleksiyonu üzerinde yapılmıştır. Bu bölümde bu denemelerden elde edilen çıkarımla verilmiştir:

- Her iki koleksiyonda (UCI ve ilaç) da en başarılı algoritmalar bu tezde geliştirilen algoritmalarlardır.
- UCI ve ilaç koleksiyonlarında, tek ağaç yerine birden fazla ağaçtan oluşan ormanları kullanmanın daha iyi sonuçlar verdiği görülmektedir.
- UCI ve ilaç koleksiyonlarında, ağaç sayısı arttıkça başarı da artmaktadır. Buna dayanılarak Breiman'ın çalışmasında (Breiman, 1999) belirttiği gibi ağaç sayısını arttırmanın aşırı eğitime (overfitting) sebep olmadığı söylenebilir.
- UCI ve ilaç koleksiyonlarında ağaçlarda kullanılan özellik sayısının artması Random Forest'ta başarıyı azaltmışken, Cline versiyonlarında arttırmıştır.
- UCI ve ilaç koleksiyonlarında, Cline Orman versiyonlarında, tek ağaçlı Cline versiyonlarında olduğu gibi dal kullanımı yaprak kullanımına göre daha yüksek bir başarıya sahiptir.
- UCI ve ilaç koleksiyonlarında, daha başarılı algoritmaların birleştirilmesi (ormanlarının oluşturulması) daha iyi sonuçlar vermiştir. Bu sonuçtan yola çıkılarak QUEST gibi tek ağaçlı algoritmaların orman versiyonlarının da orijinallerinden daha başarılı olacağı söylenebilir.
- UCI koleksiyonunda, Cline karar ormanlarına bagging eklenmesi hemen hemen her veri kümesinde ve sonuç olarak ortalama başarı da arttırmıştır.
- UCI koleksiyonunda, ClineMIX ormanlarının performanslarının standart sapma miktarı, Random Forest'lardan daha düşüktür. Bu sebeple daha güvenilir sonuçlar ürettiği söylenebilir.

- UCI koleksiyonunda, algoritmaların performanslarıyla, eğitim ve test toplam sürelerinin arasındaki ilişki incelenmiştir. Buna göre;
 - Algoritmaların karmaşıklıklarıyla performansları arasında pozitif bir ilişki olduğu görülmüştür. Diğer bir ifadeyle daha karmaşık algoritmalar genelde daha iyi performans göstermiştir.
 - En yüksek başarıya sahip algoritma olan CLMIX ormanı, en yavaş algoritma değildir.
- İlaç verileriyle yapılan çalışmalar sonucunda aşağıdaki sonuçlara ulaşılmıştır:
 - Tek bir ağaç kullanmak yerine bir orman kullanmak, UCI veri kümesindeki sonuçlara paralel olarak, daha başarılı sonuçlar üretmektedir.
 - En yüksek başarı, 100 ağaçlı CLM ormanı ile kararlar ağaçtaki düğüm sayısı ile doğru orantılı bir şekilde ağırlıklandırıldığında elde edilmiştir.

Çizelge 4.8’de Cline algoritmalarıyla elde edilen sonuçların veri kümelerine göre farklılıkları verilmiştir.

Çizelge 4.8 Cline algoritmalarının veri koleksiyonlarına göre karşılaştırılmaları

	UCI verileri	İlaç verileri
Ağaç sayısı	Arttıkça başarı da artıyor.	
Ağaçlarda kullanılan özellik sayısı	Arttıkça başarı da artıyor.	
En başarılı algoritma	CLMIX ormanı	CLM ormanı
Bootstrapping	Daha başarılı	Daha başarısız
Dal	Daha başarılı	Daha başarısız
Budama	Daha başarılı	Daha başarısız
Karar Ağırlıklandırma	Denenmedi	Daha başarılı

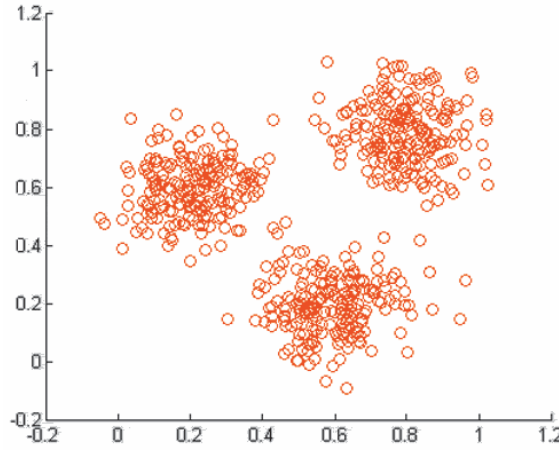
4.7 Gelecek Çalışmalar

Araştırmaya açık alanlar ve denenebilecek fikirler aşağıda listelenmiştir:

- QUEST, LMT, C4.5 gibi metotların orman versiyonlarının geliştirilmesi.
- Genetik algoritmalarla (GA) Cline Orman'ları birleştirilebilir. Literatürde diğer karar ağaçları için örnekleri bulunmaktadır. GA ile optimize edilebilecek parametreler (özellik alt kümeleri, örnek alt kümeleri, ECOC matrisleri)
- Karar Ormanlarında oluşan tüm ağaçları kullanmak yerine bir alt kümesinin kullanımında:
 - Ağaçtaki özelliklerin çıkışla korelasyonuna göre bir puanlandırma ve seçim yapılabilir. Ki bu da alt uzay puanlama olacaktır.
 - Ayrı ya da eğitim setinin yarısından oluşan bir geçerleme kümesi üzerinde en iyi performansı gösterenler seçilebilir.
 - En kısa ağaçlar seçilebilir.
 - En kısa ağaçlarda en sık yer alan özellikleri içeren ağaçların kararlarına daha yüksek ağırlık verilebilir.
 - Ağaçlara kısalıklarıyla ters orantılı olarak ağırlık verilebilir.

5. KÜMELEME

Kümeleme, veri kümesi içindeki benzer örnekleri içeren kümelerin ve bu kümeleri temsil eden merkezlerin bulunması olarak tanımlanabilir. Bir küme içindeki örnek kendisiyle aynı küme içindeki örneğe, bir başka küme içindeki örneğe göre daha çok benzemektedir. Kümelemede, örneklerin önceden belirlenmiş ya da bulunmuş bir etiketleri bulunmaz (Ör: Şekil 5.1). Şekil 5.1'deki örnek veri kümesi için kümelerin sayılarının ve içerdikleri örneklerin büyük çoğunluğunun belirlenmesi kolaydır. Ancak özellik sayısı daha büyük olan veri kümeleri için bu işlemin yapılması zordur.



Şekil 5.1 Kümeleme için örnek veri kümesi

İşte bu gibi durumlar için literatürde birçok yöntem geliştirilmiştir ve bu bölümde bu yöntemler incelenmiş ve yeni geliştirilen ClusLine algoritmalarıyla karşılaştırmaları verilmiştir.

5.1 Tanımlar ve Kullanılan Semboller

5.1.1 Kümeler Arasındaki Benzerliğin Ölçümü

İki kümenin birbirine benzerliğinin ölçümü için literatürde çeşitli metotlar önerilmiştir [8].

n_r : r kümesi içindeki örnek sayısı

$dist(e_1, e_2)$: e_1 ve e_2 örnekleri arasındaki öklid mesafesi

$d(r, s)$: r ve s kümeleri arasındaki uzaklık

x_{ri} : r kümesi içindeki i . örnek olmak üzere

Single: İki kümenin birbirine en yakın örneklerinin uzaklığını kullanır.

$$d(r, s) = \min(\text{dist}(x_{ri}, x_{sj})), i \in (1, \dots, n_r), j \in (1, \dots, n_s) \quad (5.1)$$

Complete: İki kümenin birbirine en uzak örneklerinin uzaklığını kullanır.

$$d(r, s) = \max(\text{dist}(x_{ri}, x_{sj})), i \in (1, \dots, n_r), j \in (1, \dots, n_s) \quad (5.2)$$

Average: İki kümenin örnekleri arasındaki tüm olası uzaklıklar hesaplanır, ortalaması kullanılır.

$$d(r, s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} \text{dist}(x_{ri}, x_{sj}) \quad (5.3)$$

Centroid: Küme merkezlerinin öklid uzaklığını kullanır.

$$d(r, s) = \text{dist}(x_r^-, x_s^-), x_r^- = \frac{1}{n_r} \sum_{i=1}^{n_r} x_{ri} \quad (5.4)$$

Eşitlik 5.4'te x_s^- ve x_r^- , sırasıyla s ve r kümelerinin merkezleridir.

Median: Ağırlıklandırılmış küme merkezlerinin öklid uzaklığını kullanır.

$$d(r, s) = \text{dist}(x_r^+, x_s^+) \quad (5.5)$$

Bir kümenin ağırlıklandırılmış küme merkezi bulunurken, özyinelemeli bir şekilde o kümeyi oluşturan iki kümenin ağırlıklandırılmış küme merkezleri kullanılır. Eşitlik 5.6'da r kümesi p ve q kümelerinin birleşiminden oluşmuştur.

$$x_r^+ = \frac{1}{2}(x_p^+ + x_q^+) \quad (5.6)$$

Ward: Centroid'in bir versiyonudur.

$$d^2(r, s) = n_r n_s \frac{\text{dist}^2(x_r^-, x_s^-)}{n_r + n_s} \quad (5.7)$$

5.1.2 Kümeleme Kalitesinin Ölçümü

Sınıflandırıcılar değerlendirilirken daha önceden görülmemiş veriler hakkında doğru tahmin

yapıp yapmadıklarına göre değerlendirilmektedirler. Bunun yanında kümeleme problemlerinde gerçek sınıf etiketleri elde olmadığından ve/veya sınıflarla kümeler her zaman birbirine denk düşmediklerinden çeşitli kümeleme kalitesi ölçüm metotları geliştirilmiştir.

Literatürde birçok kümeleme kalitesi ölçüm metodu bulunmaktadır. Bu çalışmada 3 adet kalite ölçüm metodu (Davies-Bouldin İndeksi (Davies vd., 1979), Silüet Genişliği (Rousseeuw, 1987) ve Sınıflandırma Başarısı) kullanılmıştır. İlk 2 ölçütte sadece örnekleri küme merkezleri ve örneklerin ait oldukları küme indislerine ihtiyaç varken, 3. ölçüt için örneklerin sınıfları da kullanılmaktadır.

Davies-Bouldin Endeksi: Veri seti N adet kümeye bölündükten sonra $C = \{C_1, C_2, \dots, C_n\}$ her bir kümenin Davies-Bouldin indeksi bulunur. Eşitlik 5.8'de ve Eşitlik 5.9'da i. küme için Davies-Bouldin indeksinin bulunması verilmiştir.

$$DB_i = \max_{j=1..N, i \neq j} (DB_{ij}) \quad (5.8)$$

Eşitlik 4.6'daki DB_{ij} terimi ise Eşitlik 5.9'daki şekilde bulunur.

$$DB_{ij} = \frac{\{sc(C_i) + sc(C_j)\}}{cd(C_i, C_j)} \quad (5.9)$$

Eşitlik 5.9'daki $sc(C_i)$ terimi C_i kümesinin içindeki örneklerin küme merkezine olan ortalama uzaklığını, $cd(x, y)$ iki kümenin merkezlerinin birbirlerine olan uzaklığını ifade etmektedir. Her bir küme için indeksler bulunduktan sonra ortalaması alınarak kümeleme algoritmasının kalitesini ifade etmekte kullanılan tüm veriye ait Davies -Bouldin indeksi bulunur (Eşitlik 5.10).

$$DB = \frac{1}{n} \sum_{i=1}^n DB_i \quad (5.10)$$

Davies-Bouldin indeksinin büyüklüğü kümeleme kalitesiyle doğru orantılıdır.

Silüet Genişliği: Silüet Genişliği bulunurken önce her bir örneğin silüet genişliği bulunur (Eşitlik 5.11). Daha sonra her bir kümenin silüet genişliği içerdiği örneklerin silüet genişliklerinin ortalaması alınarak bulunur. Tüm verinin silüet genişliği ise kümelerin silüet genişliklerinin ortalaması ile bulunur.

$$sw_i = \frac{sc(i) - sd(i)}{\max(sc(i), sd(i))} \quad (5.11)$$

Eşitlik 5.11'deki $sc(i)$, i örneğinin bulunduğu küme içindeki diğer tüm örneklerle olan ortalama mesafesini, $sd(i)$ ise i . örneğin bulunduğu kümeye en yakın olan küme içindeki tüm örneklerle olan ortalama mesafesini göstermektedir.

Bir örneğin siluet genişliğinin 1'e yakın olması örneğin uygun kümede bulunduğunu, 0'a yakın olması bulunduğu küme yerine en yakın kümede de yer alabileceğini, -1'e yakın olması ise yanlış kümede bulunduğunu göstermektedir.

Sınıflandırma Başarısı: Sınıflandırma doğruluğu bulunurken öncelikle her bir küme merkezinin ifade ettiği sınıf bulunur. Veri kümesinde her bir küme merkezine diğer merkezlere göre daha yakın olan örnekler belirlenir. Küme merkezinin temsil ettiği sınıf, ona yakın olan örneklerin sınıflarında frekansı en yüksek olan sınıftır.

Sınıflandırma doğruluğu, her bir örneğin en yakın olduğu küme merkezinin ifade ettiği sınıfla aynı sınıfa sahip olanların yüzdesiyle bulunur.

M adet örnek, N adet küme içeren bir veri setinde sınıflandırma doğruluğu Eşitlik 5.12'deki gibi bulunur.

$$cd = \frac{1}{M} \sum_{i=1}^M [S_i == sm(\arg \min_{j=1, \dots, N} \|X_i, ce_j\|)] \quad (5.12)$$

Eşitlik 5.12'deki $sm(x)$, x . küme merkezinin ifade ettiği sınıfı, $\|x, y\|$ x ile y arasındaki mesafeyi, ce_j j . küme merkezini, S_i ise i . örneğin sınıfını göstermektedir.

Bu bölümdeki algoritmalarda kullanılan genel sembollerin açıklamaları aşağıda verilmiştir.

m_i : i . kümenin merkezi (temsil ettiği örneklerle aynı boyuttadır)

K : küme sayısı

N : örnek sayısı

5.2 Önceki Çalışmalar

Literatürde kümeleme problemi için birçok çözüm önerisi getirilmiştir. Bu bölümde, popüler olarak kullanılan algoritmaların açıklamaları verilmektedir.

Kmeans: Geliştirilen ilk kümeleme algoritmalarından olan Kmeans, örnekleri en iyi temsil edebilecek merkezleri bulmaya çalışırken, örneklerin en yakın oldukları merkezlere uzaklıkları toplamını minimize etmeye çalışır.

Algoritma.Kmeans (Alpaydın, 2004)

Merkezlere $(m_i, i = 1, \dots, K)$ ilk değerlerini ata

Merkezlerin yeri değişmeyene kadar devam et

Her örneğin en yakın olduğu küme merkezini bul.

$$b_i^t = \begin{cases} 1 & \text{Eger } \|x^t - m_i\| = \min_j \|x^t - m_j\| \\ 0 & \text{deg ilse} \end{cases} \quad (5.13)$$

(Eşitlik 5.13'te, i . merkez x^t 'ye diğer merkezlerden daha yakın ise $b_i^t = 1$ olur. Değilse 0 değerini alır.)

Merkezlerin yerlerini kendi kümeleri içindeki örneklerin orta noktasına ata

$$m_i = \frac{\sum_{t=1}^N b_i^t x^t}{\sum_{t=1}^N b_i^t}, i = 1, \dots, K \quad (5.14)$$

Fuzzy Kmeans: Kmean algoritmasında bir örnek sadece bir kümeye dahildir. Bezdek tarafından geliştirilen (Bezdek, 1973) Fuzzy Kmeans algoritmasında ise her örnek her kümeye mesafesiyle bağıntılı bir oranda dahildir.

c : küme sayısı

n : örnek sayısı

u_{ij} : j . örneğin i . kümeye dahil olma olasılığı $[0,1]$

c_i : i . küme merkezi

d_{ij} : i . küme merkeziyle j . örnek arasındaki öklid mesafesi

m : kullanıcı tarafından belirlenen ağırlık katsayısı $[1, \infty]$ olmak üzere:

Algoritma:

Üyelik matrisinin (U) ilk değerlerini Eşitlik 5.15'e uygun ve rasgele olarak ata.

$$\sum_{i=1}^c u_{ij} = 1, \forall j = 1, \dots, n \quad (5.15)$$

Eşitlik 5.16'yı kullanarak küme merkezlerini (c_i) hesapla

$$c_i = \frac{\sum_{j=1}^n u_{ij}^m x_j}{\sum_{j=1}^n u_{ij}^m} \quad (5.16)$$

Küme merkezleriyle örnekler arasındaki toplam mesafe belli bir eşik değerinin altına düşene kadar ya da küme merkezlerinde bir değişiklik olmayıncaya kadar aşağıdaki adımı tekrar et:

Küme merkezleriyle örnekler arasındaki mesafeleri (J) Eşitlik 5.17'yi ve Eşitlik 5.18'i kullanarak hesapla.

$$J(U, c_1, c_2, \dots, c_c) = \sum_{i=1}^c J_i = \sum_{i=1}^c \sum_{j=1}^n u_{ij}^m d_{ij}^2 \quad (5.17)$$

$$u_{ij} = \frac{1}{\sum_{k=1}^c \left(\frac{d_{ij}}{d_{kj}} \right)^{2/(m-1)}} \quad (5.18)$$

En Uzakla Kümeleme(Farthest First-FF): Hochbaum ve Shmoys tarafından önerilen çok hızlı ve basit bir kümeleme algoritmasıdır (Hochbaum vd., 1985).

Algoritma:

K adet merkez bulmak için:

Örneklerden birini 1. merkez (m_1) olarak etiketle

For i=2, ... ,K

Etiketli olmayan örneklerden, etiketlenmiş örnekler kümesine (S) en uzak örneği i. merkez (m_i) olarak etiketle ve S'e ekle.

Bir örneğin bir kümeye olan uzaklığının bulunmasında çeşitli yöntemler mevcuttur (Bkz. Bölüm 5.1.1). Bu algorithmada ise , bu uzaklık, örneğe küme içindeki en yakın örneğin uzaklığıdır (Single-Link) (Eşitlik 5.14).

$$d(x, S) = \min_{y \in S} d(x, y) \quad (5.14)$$

Kendi Kendini Düzenleyen Haritalar (Self Organizing Maps- SOM): Kohonen (Kohonen, 1990) tarafından ortaya atılan bu algorithmada küme merkezleri birbirleriyle 2 boyutlu bir ızgara üzerinde komşudurlar. Bir örneğe en yakın olan merkezin değeri güncellenirken, o merkezin komşuları da güncellenir.

Algoritma:

Merkezlere ilk değerlerini ($m_l, l = 1, \dots, K$) ve komşuluklarını (nb) ata,

Merkezlerin yeri değişmeyene kadar tekrar et

Bir örnek seç (x^t)

Örneğe en yakın merkezi (m_i) bul

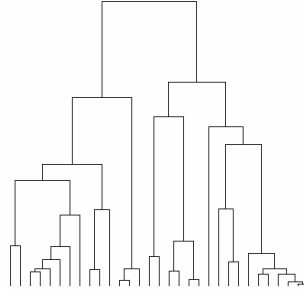
$$i = \arg \min_l \|x^t - m_l\| \quad (5.15)$$

Merkezlerin değerlerini Δm_l kadar arttır.

$$\Delta m_l = \eta * nb(l, i) * (x^t - m_l) \quad (5.16)$$

Eşitlik 5.16 'da $nb(l, i)$, l . merkez i . merkezin komşusu ise 1, değilse 0 değerini almaktadır. η zamanla azalan öğrenme katsayısıdır.

Hiyerarşik Kümeleme: Örneklerin birbirlerine yakınlıklarına göre hiyerarşisini çıkaran algorithmadır. Temelde iki yaklaşıma sahiptir. İlki başlangıçta örneklerin her birini bir küme olarak kabul edip her bir adımında kümelerin birbirlerine uzaklığını hesaplayıp o anda birbirine en yakın olan iki kümeyi birleştiren algorithmadır (özelden genele). İkincisi ise başlangıçta tüm örnekleri tek bir kümeye koyup her seferinde birbirine en uzak iki küme bulmaya çalışan algorithmadır (genelden özele). Her iki yaklaşım sonucunda da hiyerarşik bir ağaç ortaya çıkar (Şekil 5.2).



Şekil 5.2 Örneklerden elde edilen hiyerarşik ağaç

Şekil 5.2’de ağacın yapraklarında birer örnek bulunurken, dallarda örnek kümeleri yer almaktadır. Ağacın kökünde (en üstünde) ise bütün örnekler yer almaktadır. Ağaçta aşağıdan yukarıya doğru ilerlenirse kümelerdeki örnek sayısı artarken, kümelerin birbirlerine benzerliklerinin ortalaması azalır. Aşağıda özelden genele hiyerarşik kümelemenin algoritması verilmiştir.

Algoritma. Hiyerarşik Kümeleme

K: küme sayısı

Her biri sadece 1 örnek içeren kümeler (N adet) oluşturun.

Kümelerin birbirlerine yakınlık matrisini hesapla.

K adet küme kalana kadar tekrar et.

Birbirine en yakın iki kümeyi birleştir.

Yakınlık matrisini güncelle.

Kümelerin benzerliklerinin hesaplanması için literatürde çeşitli yaklaşımlar önerilmiştir (Bkz. Bölüm 5.1.1).

Hiyerarşik kümeleme diğer algoritmalarından farklı olarak örneklerin koordinatlarına ihtiyaç duymaz. Örneklerin arasındaki mesafe matrisi bu algoritma için yeterlidir. Bu sebeple kümeleme kararlarını birleştirilmesinde de kullanılmaktadır (Bkz. Bölüm 6.1.2) (Fern vd., 2004).

5.3 Gerçekleştirilenler

Tezde, sınıflandırma problemlerinde karar ağaçlarının yüksek performans göstermesi,

kümeleme problemlerinde de karar ağaçlarının kullanılabileceğini gündeme getirmiştir ve ClusLine algoritmaları geliştirilmiştir. Geliştirilen Clusline algoritmasının temel fikri; uzayı, her seferinde bir düzlemle iki bölgeye ayırmak ve bu işleme bölgelerdeki verinin standart sapması tüm verinin standart sapmasından X kat daha küçük olana kadar devam etmektir. $X > 1$ durumu için algoritma yakınsamaktadır. Algoritma, uzayı karar ağaçları gibi hiyerarşik olarak bölgelere bölmektedir ve bu sırada bir ağaç yapısı da oluşmaktadır. Algoritmanın oluşturduğu ağacın her bir düğümünde (Cline algoritmasında olduğu gibi) bu hiper düzlemlerin parametreleri bulunur ve örnekler bu parametrelere göre dallara gönderilir. Çizelge 5.1’de Clusline algoritması verilmiştir.

Çizelge 5.1 Clusline algoritması

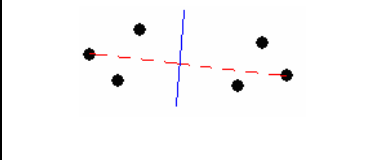
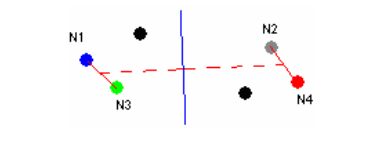
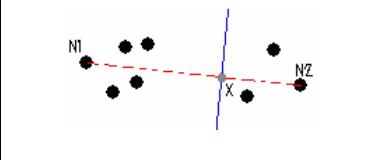
- 1- Tüm verinin standart sapması, şu anki verinin standart sapmasının X katından büyük mü?
 - a. Hayırsa 2. adıma git.
 - b. Evetse verinin ortalamasını alarak küme merkezini belirle.
- 2- Veri kümesini ikiye bölen bir hiper düzlem bul.
- 3- Veri kümesindeki elemanları hiper düzlemin altında ya da üstünde kaldıklarına göre ayır ve öz yinelemeli olarak her bir yeni veri kümesi için 1. adıma geri dön.

Algoritmanın tek parametresi (X) vardır ve 1’den büyük bir değer olarak seçilir. Algoritmanın kaç küme belirleyeceği kullanıcı tarafından belirlenmemektedir. Algoritma, X ’in büyük değerleri için daha çok sayıda, küçük değerleri için az sayıda küme bulmaktadır. Diğer bir ifadeyle X değeri ile algoritmanın ürettiği küme sayısı doğru orantılıdır. Denemelerde X ’e 1.2 ile 4 arasında değerler verilmiştir.

ClusLine algoritmasıyla kümeleme ağacı oluşturulduktan sonra, bir örneğin hangi kümeye ait olduğu bulunurken bu ağaç kullanılmaz. Kümesi belirlenecek örnek, yapraklarda bulunan küme merkezlerinden hangisine en yakınsa o kümeye dahil edilir.

5.3.1 Clusline Algoritmasının Versiyonları

Clusline algoritmasının 2.adımında yer alan veriden iki noktanın seçilip hiper düzlemin elde edilmesi işlemi için 3 farklı yaklaşım geliştirilmiştir. Şekil 5.3’te bu versiyonlar görülmektedir.

		
a) ClusLine2	b) ClusLine4	c) ClusLineW

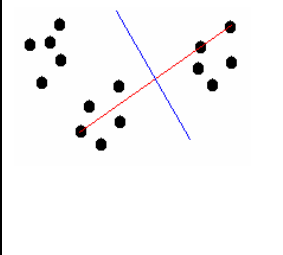
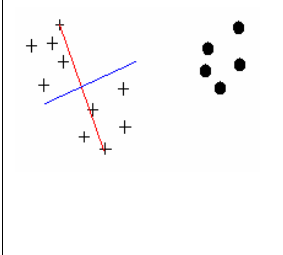
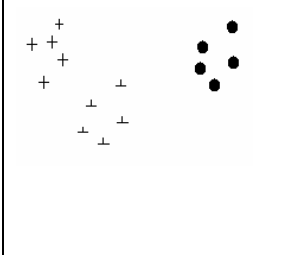
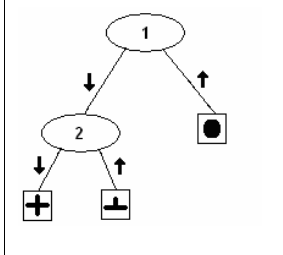
Şekil 5.3 ClusLine versiyonları

ClusLine2: Birbirine en uzak iki noktayı birleştiren (kesik çizgili) doğruya dik ve tam ortasından geçen (düz çizgili) doğru uzayı ikiye bölmektedir.

ClusLine4: Birbirine en uzak iki nokta (N1 ve N4) belirlendikten sonra, noktaların her birine diğerinden sonraki en uzak iki nokta bulunur. N1'e N4'ten sonraki en uzak nokta N2 iken, N4'e N1'den sonraki en uzak nokta N3'tür. N1 ve N3'ün orta noktasıyla, N2 ve N4'ün orta noktasından geçen (kesik çizgili) doğruya dik ve tam ortasından geçen (düz çizgili) doğru uzayı ikiye bölmektedir.

ClusLineW: Birbirine en uzak iki noktaya N1 ve N2 denilirse; veri kümesindeki örneklerden kaçının N1'e N2'den daha yakın olduğu bulunur. Aynı şekilde kaçının N2'ye N1'den daha yakın olduğu bulunur. Şekil 5.3.c'de N1'e yakın 4, N2'ye yakın 2 nokta bulunmaktadır. Bulunan iki sayının oranına göre uzay bölünür. Şekil 5.3.c'de N1-X mesafesi N2-X mesafesinin 2 katıdır.

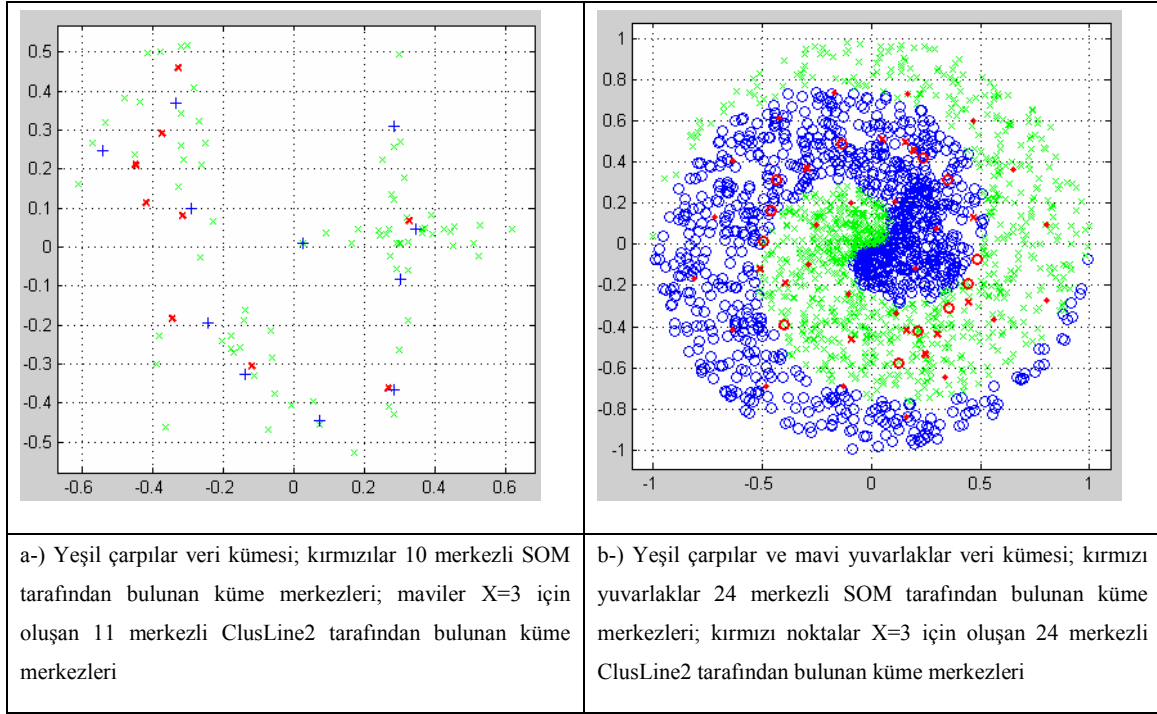
Şekil 5.4'te ClusLine2 algoritması ile iki boyutta kümelemenin nasıl yapıldığı ve algoritma sonucunda oluşan ağaç verilmiştir.

			
1. adım	2.adım	Oluşan kümeler	Oluşan ağaç

Şekil 5.4 Clusline_2 ile adım adım kümeleme işlemi

Clusline algoritmasında da Cline algoritmasında olduğu gibi hiper düzlemlerin parametreleri bulunurken hiper düzleme dik bir doğrultu vektörü ve hiper düzlemin üzerindeki bir nokta kullanılmaktadır. (Bkz. Bölüm 3.3.4)

ClusLine2 algoritmasının yapay veri kümeleri üzerinde uygulanması ve SOM ile karşılaştırılması Şekil 5.5'te verilmiştir.



Şekil 5.5 İki farklı veri kümesinde ClusLine2 ile SOM'un karşılaştırılması

Şekil 5.5 incelendiğinde ClusLine tarafından bulunana küme merkezlerinin veri uzayının her tarafına daha iyi yayıldığı ve bu sayede veri kümesini SOM'a göre daha iyi temsil edebileceği söylenebilir. SOM tarafından bulunan küme merkezlerinin büyük bir kısmı birbirlerine çok yakın yerlerde konumlandıklarından şekilde sayıları daha az gözükmemektedir. Bu merkezlerin birbirlerinden ayrıklaştırılması için 50 olan iterasyon sayısı arttırılmalıdır.

5.4 Deneysel Sonuçlar

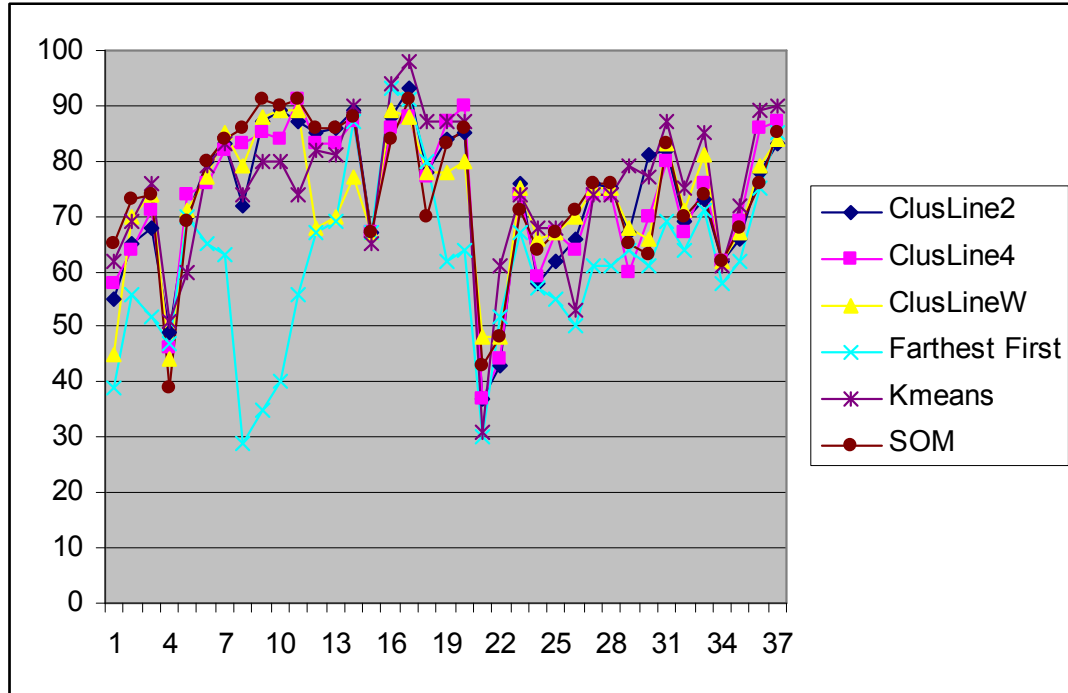
Kümeleme problemleri için kullanılan 14 veri kümesi ve özellikleri Çizelge 5.2’de verilmiştir.

Çizelge 5.2 Kümelemede kullanılan veri kümeleri ve özellikleri

Veri seti No	Veri Seti	Özellik sayısı	Sınıf sayısı	Örnek Sayısı
1	Glass	9	6	170
2	derma	34	6	286
3	ecoli	7	8	266
4	breast-cancer	30	2	456
5	weather	34	2	281
6	iris	4	3	120
7	New-thyroid	5	3	143
8	segmentation	19	7	210
9	bupa	6	2	175
10	wine	13	3	118
11	waveform	21	3	2460
12	Monks1	6	2	124
13	Monks2	6	2	169
14	Monks3	6	2	122

Geliştirilen ClusLine versiyonları ve diğer kümeleme algoritmaları Farthest First (Hochbaum vd., 1985), K-Means, Self Organizing Maps (SOM) (Kohonen, 1990); Bölüm 5.1.2’de anlatılan 3 kalite ölçümü metoduyla test edilmiştir.

Şekil 5.6’da 14 veri kümesi için sınıflandırma doğruluğu ölçütüne göre başarılar yer almaktadır. Her bir veri kümesi için farklı sayıda kümeler oluşturularak (Cline’da farklı X değerleri, diğer algoritmalarda farklı küme sayıları seçilerek) her bir algoritma 37 kez çalıştırılmıştır. Her bir denemede önce Clusline’a X değeri verilmiş ve bulduğu küme sayısı, diğer kümeleme algoritmalarına verilmiştir. Bu sayede kümeleme algoritmalarının hepsi aynı sayıda küme bulduklarında karşılaştırılabilirlerdir.



Şekil 5.6 Kümeleme algoritmalarının farklı veri kümeleri üzerindeki sınıflandırma doğrulukları

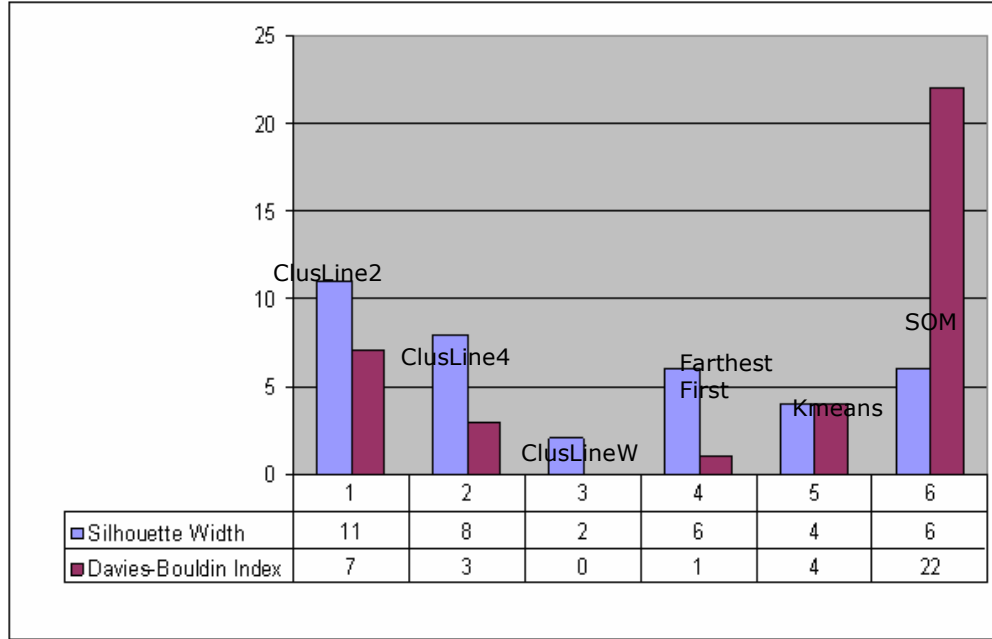
Testlerden elde edilen ortalama sınıflandırma doğrulukları ve başarılarına göre büyükten küçüğe sıralandıklarındaki ortalama sıraları Çizelge 5.3'te verilmiştir. Görüldüğü gibi başarı oranları arasında Farthest First'ün başarısızlığı haricinde çok fazla bir fark yoktur.

Çizelge 5.3 Kümeleme algoritmalarının ortalama doğrulukları ve sıralama ortalamaları

	ClusLine2	ClusLine4	ClusLineW	Farthest First	K-Means	SOM
Ortalama Sınıflandırma Başarıları(%)	73.4	72.7	73.3	61.5	75.1	74.2
Ortalama rankları	3.595	3.297	3.622	5.216	2.622	2.649

Şekil 5.7'de Davies-Bouldin İndeksi ve Siluet Genişliği ölçütlerine göre 14 veri kümesinin farklı küme sayılarıyla yapılan 37 testin sonuçları verilmiştir. Her bir algoritmanın 37 testin kaçında en iyi sonucu verdiği görülmektedir. Algoritmalar Çizelge 5.3'teki numaralarıyla

ifade edilmiştir.



Şekil 5.7 Davies-Bouldin indeksi ve siluet genişliği ölçütlerine göre kümeleme algoritmalarının başarıları

Şekil 5.7’de görüldüğü gibi Siluet Genişliği ölçütüne göre 37 kümeleme denemesinden 11’inde ClusLine2 en yüksek dereceyi elde etmiştir. Davies-Bouldin indeksine göre ise SOM 37 denemenin 22’sinde en yüksek dereceyi elde ederken en iyi ikinci algoritma ClusLine2’dir.

5.5 Çıkarımlar

Karar ağaçlarından esinlenilerek geliştirilen ClusLine kümeleme algoritmaları çeşitli kümeleme performans kriterlerine göre popüler kümeleme algoritmaları ile karşılaştırılmıştır ve aşağıdaki çıkarımla elde edilmiştir.

- Sınıflandırma başarısı ve ranka göre başarı sıralaması: *K-means* > *SOM* > *diğer*
- Siluet Genişliğine göre başarı sıralaması: *Clusline2* > *Clusline4* > *diğer*
- Davies-Bouldin indeksine göre başarı sıralaması: *SOM* > *Clusline2* > *diğer*
- Clusline basitliğine rağmen başarılı sonuçlar elde etmiştir.
- Sadece SOM ve Clusline2, 3 kriterin 2’sinde ilk 2’ye girebilmişlerdir.

- Siluet genişliği ve DB indeks, tanımlarındaki benzerliğe rağmen oldukça farklı sonuçlar üretmiştir.

Sonuç olarak kolay uygulanabilirliği ve başarı seviyesiyle Clusline2 popüler kümeleme algoritmaları arasında yer almaya aday bir algoritmadır.

5.6 Gelecek Çalışmalar

Araştırmaya açık alanlar ve denenebilecek fikirler aşağıda listelenmiştir:

- EM kümeleme algoritması da karşılaştırmalara eklenebilir.
- Kümeleme hiper düzlemlerini bulurken:
 - Birbirine en uzak iki noktadan hiper düzlemin doğrultusu, DBindeks'ine göre en iyi yerden de eşik değeri bulunabilir.
 - En büyük eigenvalue'ya sahip eigenvector hiper düzlemin doğrultusu, orta nokta ya da Dbindex'ine göre en iyi yerden de eşik değeri bulunabilir.
- CluslineMIX geliştirilebilir. (clineMIX gibi -fark sınıflandırma başarısı yerine DBindex'i kullanıp- en iyisi seçilebilir)

6. KÜMELEME KOMİTELERİ

Sınıflandırıcıların kararlarının birleştirilmesinde elde edilen başarılar, kümeleme sonuçlarının birleştirilmesi alanında çalışmalara kaynak oluşturmıştır. Bu yaklaşımda da aynı veri kümesi üzerinde farklı kümeleyici sonuçları elde edilerek birleştirilmesi yoluna gidilmiştir.

Çizelge 6.1’de bir kümeleme komitesinin her bir üyesinin sonuçları görülmektedir. Algoritma sonuçlarında görülen değerler kümelerin etiketlerini göstermektedir.

Çizelge 6.1 10 adet örneğin 5 farklı kümeleme algoritmasıyla kümeleneşinden elde edilen sonuçlar

Örnek No	Alg.1 sonucu	Alg.2 sonucu	Alg.3 sonucu	Alg.4 sonucu	Alg.5 sonucu
1	1	2	2	1	4
2	2	1	2	1	4
3	1	2	2	2	1
4	2	1	3	2	1
5	1	2	1	2	1
6	1	2	1	2	1
7	2	1	3	1	3
8	2	1	3	1	3
9	1	2	1	2	2
10	1	2	1	2	2

Kümeleme sonuçlarını birleştirilmesi sınıflandırma sonuçlarının birleştirilmesinden iki yönden ayrılmaktadır:

- Her bir alt uzay birbirinden farklı sayıda küme içerebilir. Sınıflandırmada ise her alt uzayda sınıf sayısı aynıdır. Çizelge 6.1’de Alg.1, 2 ve 4 2’şer küme, Alg.3 3 küme ve Alg.4 4 küme bulmuştur.
- Her bir alt uzaydaki etiketler birbirinden farklıdır. Bu nedenle sınıflandırmada olduğu gibi en fazla oyu olan sınıfı seçmek gibi bir metot uygulanamaz. Çizelge 2.1’de Alg.1 ve Alg.2 aynı kümelemeyi yapmış olmalarına rağmen kullandıkları etiketler birbirinin zıttıdır.

Sınıflandırıcı komitelerinin başarısı etkileyen faktörlerden biri sınıflandırma algoritmalarının kararlarının birbirinden bağımsız olmalarıdır. Kararların bağımsızlığını etkileyen en önemli faktör ise eğitim kümelerinin birbirlerinden ayrıklığıdır (diversity). Derek ve arkadaşları

(Derek vd., 2004) çalışmalarında 7 farklı komite oluşturma metodu denemiş ve ürettikleri veri kümelerinin ayrıklığı en fazla olanın rasgele alt uzaylar (random subspacing) olarak adlandırılan, örneklerin rasgele seçilen özellik alt kümeleriyle çeşitli veri kümelerinin oluşturulması metodu olduğunu göstermiştir. Bu nedenle, bu bölümdeki çalışmalarda farklı kümeleyici kararları oluşturmak için bu yöntem kullanılmıştır. Öncelikle kümelenecek veri kümesinden rasgele özellik altkümeleri seçilerek çok sayıda veri kümesi oluşturulmuş ve her biri kümeleme algoritmalarınca kümelendikten sonra algoritmaların sonuçları birleştirilmiştir.

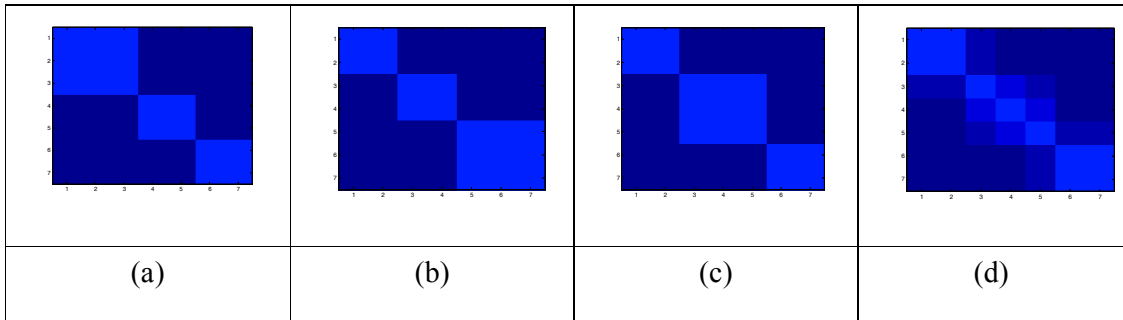
6.1 Önceki Çalışmalar

Her biri orijinal veri kümesinin bir alt kümesine göre karar veren kümeleyicilerin kararlarının birleştirilmesi için literatürde temelde graf bölümlene ve hiyerarşik olmak üzere 2 farklı metot önerilmiştir (Strehl vd., 2002).

6.1.1 Graf Bölümlene Metotları

Graf bölümlene algoritmalarında giriş, köşeler ve ağırlıklandırılmış kenarlardan oluşur. Amaç minimum sayıda ve ağırlıktaki kenarı keserek, grafi K adet eşit büyüklükteki parçaya bölmektir. Kümeleyici kararlarının graf bölümlene algoritmalarıyla çözümlenebilmesi için kümeleme sonuçlarının graf formatına çevrilmesi gerekir. Strehl ve Ghosh, graf bölümleneyi kümeleyici topluluklarında kullanımına dair örnek tabanlı, küme tabanlı ve meta küme tabanlı üç yaklaşım önermişlerdir.

Örnek tabanlı yaklaşım: Grafın köşeleri örnekler, kenar ağırlıkları kenarın bağladığı iki köşedeki örneklerin kaç kümeleme sonucunda aynı küme içinde yer aldıklarıdır. Şekil 6.1. a,b,c 'de 7 örneğin 3 kümeleme sonucunda birlikte yer almaları verilmiştir. Şekil 6.1.d'de ise örneklerin ortalama olarak kaç kümeleme sonucunda aynı kümede kaldıkları verilmiştir. Graf bölümlene algoritmasına son benzerlik matrisi verilir ve Eşitlik 6.1'deki şekilde bulunur.



Şekil 6.1 Üç farklı kümeleme sonucunun birleştirilmesi (Renklerin açıklığı ilişkinin gücüyle doğru orantılıdır)

$$W(i, j) = \frac{1}{K} \sum_{t=1}^K I(g_t(X_i) = g_r(X_j)) \quad (6.1)$$

Eşitlik 6.1’de $I(\cdot)$ içindeki ifade doğru olduğunda 1, olmadığına 0 döndüren bir fonksiyondur. $g_r(\cdot)$ bir örneğin ait olduğu kümeyi döndürür.

Küme tabanlı yaklaşım: Grafın köşeleri kümeler, kenar ağırlıkları kenarın bağladığı iki köşedeki kümenin birbirlerine Jaccard ölçümüne göre benzerliğidir ve Eşitlik 6.2’deki şekilde bulunur. Jaccard ölçümü, ortak eleman sayılarının, birleşim kümesinin eleman sayısına oranı olarak tanımlanmıştır.

$$W(i, j) = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} \quad (6.2)$$

Eşitlik 6.2’deki C_i , i . kümenin içerdiği örnekler kümesidir. Birçok kümeleyici sonucundan, kümeler arası benzerlik matrisi elde edildikten sonra bu matris graf bölümlene algoritmasına verilir ve sonuç kümeleme elde edilir.

Strengl ve Gosh’un makalesinde örnek tabanlı, küme tabanlı ve ayrıntılarına burada girmeyeceğimiz meta küme tabanlı 3 metotla (cspa, hgpa, mcla) yaptıkları denemeler verilmiştir. Örnekler, kümeler ve meta kümeler arası mesafelerin bulunduğu bu metotların sonuçları Graf Bölümlene algoritmalarına gönderilmiş ve örneklere ait sonuç küme etiketleri bulunmuştur. Bu üç metottan en iyi kümeleme sonucu veren “Normalized Mutual information” ile seçen bir uygulama geliştirmişler ve kullanıma sunmuşlardır. Bölüm 6.3’te yapılan karşılaştırmalar bu 3 metottan en iyisini seçen metotla, Bölüm 6.1.2’de anlatılan hiyerarşik metot arasında yapılmıştır.

6.1.2 Hiyerarşik (agglomerative) Metotlar

Örnek tabanlı yaklaşımda sözü edilen, örneklerin ortalama olarak kaç adet kümeleme sonucunda aynı küme içinde yer aldıklarını gösteren benzerlik matrisi bir çeşit örnekler arası mesafenin matrisi olarak da düşünülebilir. Ve elde örnekler arası mesafeler varsa burada her defasında birbirine en benzer iki kümeyi birleştirerek kümeleme yapan hiyerarşik kümeleme metotları kullanılabilir.

6.2 Gerçekleştirilenler

Tezin bu bölümünde, örnek, küme ve meta küme tabanlı gösterimlerden en iyisini seçerek

graf bölümlleme algoritmalarıyla sonuç küme etiketlerini bulan algoritma ile, örnek tabanlı gösterimlerden hiyerarşik metotla sonuç küme etiketlerini bulan algoritma karşılaştırılmıştır. Sonuç çizelgelerinde ilk metot “graf” adıyla, ikinci metot ise “hiyerarşik” adıyla belirtilmiştir.

Hiyerarşik metotla, benzerlik matrisinden dendrogram elde edilirken kümelerin benzerliğinin bulunmasında 6 farklı yaklaşım bulunmaktadır (Bkz. Bölüm 5.1.1). Çizelge 6.2’de bu 6 metodun komite kararlarını birleştirmedeki başarıları 11 veri kümesi üzerinde verilmiştir. Denemeler Fuzzy kmeans (Bezdek, 1973) kümeleme algoritmasının her bir veri kümesi için 10’ar komitesinin kararlarıyla yapılmıştır. Her bir kümeleme algoritmasına, orijinal veri kümesinin özellik sayısına M denirse, $2\log_2 M$ adet rasgele seçilen özellik sayılı veriler verilmiştir.

Çizelge 6.2 Küme benzerliğinin ölçümünde kullanılacak yaklaşımın seçim denemeleri

	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	ortalama
complete	0.8548	0.845	0.8873	0.7154	0.8084	0.5924	0.8364	0.3819	0.814	0.6846	0.6746	0.7359
average	0.8626	0.8327	0.8689	0.6752	0.82273	0.5959	0.8322	0.3781	0.7889	0.6857	0.6805	0.7294
Weighted	0.8591	0.8491	0.8921	0.6966	0.807	0.6206	0.8329	0.3981	0.7029	0.672	0.6771	0.728
Centroid	0.8947	0.839	0.8568	0.703	0.8469	0.5947	0.8329	0.3967	0.8398	0.6766	0.6814	0.742
Median	0.8577	0.8564	0.8761	0.6887	0.8189	0.5612	0.8343	0.3886	0.7819	0.6857	0.6814	0.7301
Ward	0.8352	0.8776	0.8763	0.6898	0.815	0.6047	0.8364	0.3881	0.7871	0.6857	0.6805	0.7342

Çizelge 6.2’de görüldüğü gibi “centroid” en başarılı yaklaşım olmuştur ve bizim çalışmamızda hiyerarşik kümeleme için Matlab’ın *dendrogram* fonksiyonu ‘centroid’ parametresiyle kullanılmıştır.

6.3 Deneyisel Sonuçlar

Kümeleme denemelerinde kullanılan UCI veri kümelerinin özellikleri Çizelge 6.3’te verilmiştir.

Çizelge 6.3 Kümeleme komiteleri için kullanılan veri kümeleri

Veri kümesi	Boyut Sayısı	Sınıf Sayısı	Örnek Sayısı
Glass	9	6	170
Derma	34	6	286
Ecoli	7	8	266
breast-cancer	30	2	456
Weather	34	2	281
İris	4	3	120
New-thyroid	5	3	143
segmentation	19	7	210
ionosphere	34	2	171
Bupa	6	2	175
Wine	13	3	118
Waveform	21	3	2460
Monks1	6	2	124
Monks2	6	2	169
Monks3	6	2	122

Değerlendirme Kriteri: Elimizdeki veri kümelerindeki örneklerin sınıfları mevcut olduğundan kümeleme kararlarını birleştiren algoritmaların başarıları sınıflandırma performanslarıyla ölçülmüştür. Ayrıca rasgeleliği azaltmak için her algoritma her veri kümesine 10 ‘ar kez uygulanmış ve sonuçların standart sapmaları da hesaplanmıştır. Kümeleme algoritmalarının örneklerin sınıflarını vermediği bilinmektedir. Bunun için birleştirilmiş sonuçlardan yeni ortalamalar alınarak yeni küme merkezleri üretilmiş, merkezlere içerdiklerin örneklerin sınıflarına bakılarak sınıf etiketleri atanmıştır. Daha sonra merkezin etiketi içerdiği tüm örneklere aktarılmıştır. Bu sayede kümeleme sonuçlarının gerçek sonuçlarla aynı tür ve sayıda etiketlere sahip olması sağlanmıştır. Küme sayısının artırılması, kullanılan bu metot nedeniyle doğal olarak algoritmaların başarısını arttırmaktadır.

6.3.1 Elde Edilen Sonuçlar

15 veri kümesinden elde edilen sonuçların ortalamaları, kararları birleştirilen kümeleyici türlerine, her bir kümeleyicinin küme sayısına göre ve özellik sayısına göre karşılaştırmalı olarak verilmiştir. Orijinal sütununda tüm veri kümesi tek bir kerede kümelendiğindeki başarı verilmiştir. Küme sayısı veri kümesindeki sınıf sayısına eşit ve 2 katı olacak şekilde 2 farklı şekilde seçilmiştir. Kümeleyicilerde 4 temel kümeleme algoritması kullanılmıştır. Kararların birleştirilmesinde her seferide 10’ar adet kümeleyici oluşturulmuş ve kararları iki farklı yöntemle (graf ve hiyerarşik) birleştirilmiştir. Çizelge 6.4 ve Çizelge 6.5’teki her değer (15 veri kümesi * 10 tekrar sayısı) 150 değerın ortalaması alınarak elde edilmiştir.

Çizelge 6.4'te küme sayısı sınıf sayısına eşit olduğu durumdaki sonuçlar görülmektedir.

Çizelge 6.4 Küme sayısının sınıf sayısına eşit seçildiğindeki performanslar

Küme sayısı = Sınıf sayısı	Orijinal	Hiyerarşik	graf	Hiyerarşik	graf	ortalama
Özellik sayısı	M	Log ₂ M	Log ₂ M	2Log ₂ M	2Log ₂ M	
Kmeans	0.67	0.68	0.69	0.68	0.68	0.68
Fuzzy Kmeans	0.68	0.69	0.68	0.69	0.69	0.69
SOM	0.64	0.68	0.69	0.7	0.69	0.68
Hiyerarşik	0.65	0.62	0.68	0.64	0.69	0.65
Ortalama	0.66	0.67	0.68	0.68	0.69	

Çizelge 6.4 incelendiğinde Kmeans, fuzzykmeans ve SOM algoritmalarının performansları arasında çok az bir fark vardır. Bununla birlikte ortalama olarak en başarılı algoritma Fuzzy Kmeans'dir. Karar birleştirme sonuçları orijinal sonuçlardan neredeyse her zaman daha iyi sonuçlar üretmiştir. Bu da kümeleyici sonuçlarını birleştirmenin sınıflandırıcı sonuçlarını birleştirmedeki gibi performansa olumlu bir katkı yaptığını göstermektedir. Graf algoritmaları, hiyerarşik algoritmalarından daha yüksek bir ortalama başarıya sahiptir. Ancak bununla birlikte 15 veri kümesinde elde edilen en yüksek ortalama başarı **0.7** ile hiyerarşik algoritma sahiptir. Kümeleyicilerde kullanılan özellik sayısının artışının performansa az miktarda da olsa olumlu yansıdığı görülmektedir.

Çizelge 6.5'te küme sayısı veri kümelerindeki sınıf sayısının 2 katı olarak seçildiğindeki alınan sonuçlar verilmektedir.

Çizelge 6.5 Küme sayısının sınıf sayısının iki katı seçildiğindeki performanslar

Küme sayısı = 2*sınıf sayısı	Orijinal	Hiyerarşik	graf	Hiyerarşik	Graf	
Özellik sayısı	M	Log₂M	Log₂M	2Log₂M	2Log₂M	ortalama
Kmeans	0.75	0.75	0.74	0.76	0.74	0.75
Fuzzy Kmeans	0.75	0.75	0.73	0.75	0.75	0.75
SOM	0.74	0.72	0.74	0.74	0.74	0.74
Hiyerarşik	0.7	0.68	0.72	0.7	0.72	0.71
Ortalama	0.74	0.73	0.73	0.74	0.74	

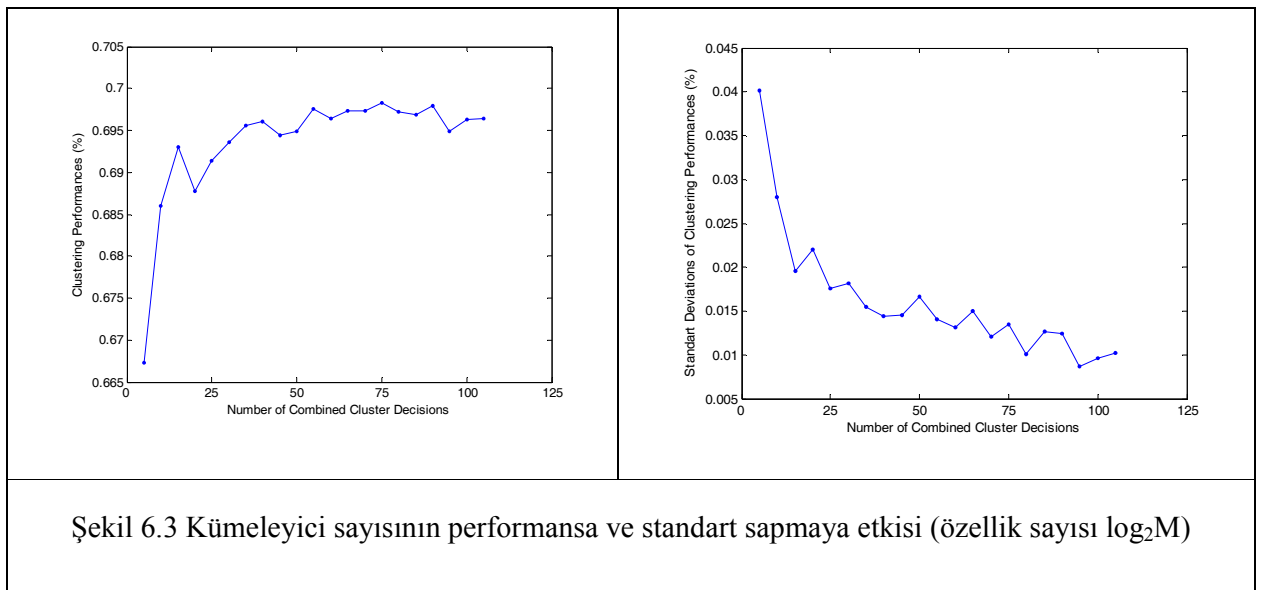
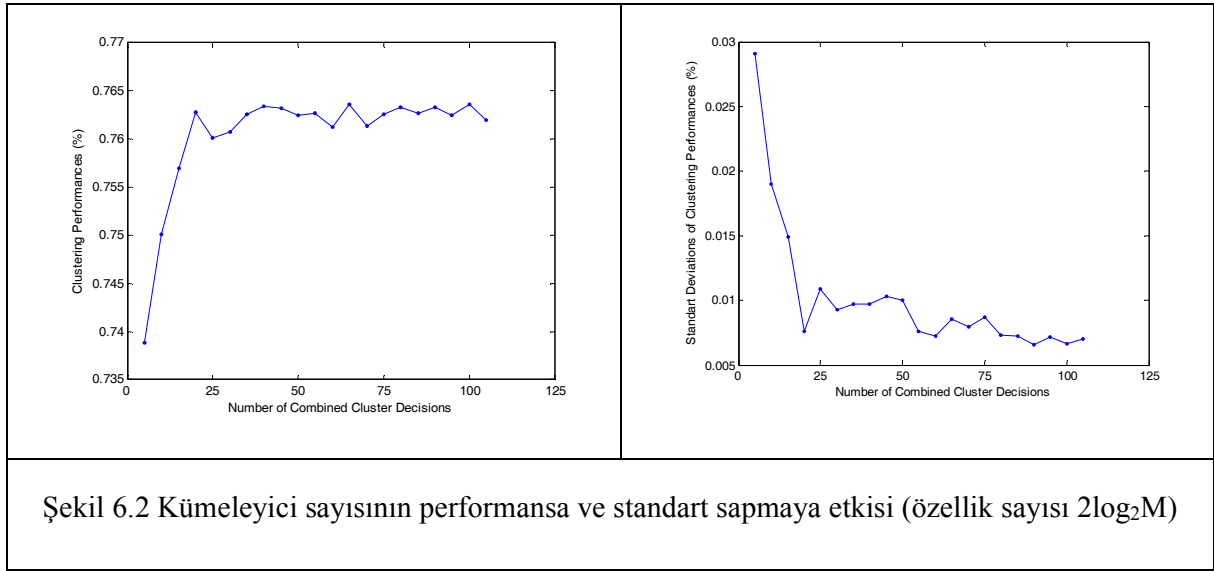
Kmeans ve Fuzzy kmeans türünde kümeleyicilerin performans artışı sadece sonuçlarının hiyerarşi metoduyla birleştirilmesiyle olmuştur. SOM'da ve hiyerarşikte ise performansı sadece graf metodu arttırmıştır. Çizelge 6.6'da yapılan deneylerden çıkarılan sonuçlar karşılaştırmalı olarak verilmiştir.

Çizelge 6.6 Kümeleme komitelerinden elde edilen özet sonuçlar

	Küme sayısı= Sınıf sayısı	Küme sayısı= 2*Sınıf sayısı
En başarılı kümeleme alg.	Fuzzy Kmeans	Kmeans
En başarısız kümeleme alg.	Hiyerarşik	Hiyerarşik
En başarılı sonucu üreten karar birleştirme alg.	Hiyerarşik (2log ₂ M adet özellik ve SOM ile)	Hiyerarşik (2log ₂ M adet özellik ve Kmeans ile)
Ortalama en başarılı karar birleştirme alg.	Graf tabanlı (2log ₂ M adet özellik ile)	Graf tabanlı (2log ₂ M adet özellik ile)
Orijinal / birleştirme sonuçlarının performans sıralaması:	orijinal<hiyerarşik<graf tabanlı	hiyerarşik<orijinal<graf tabanlı
Kümeleyicilerde kullanılan salt uzaylardaki özellik sayısının (boyutun) etkisi	Log ₂ M<2log ₂ M	Log ₂ M<2log ₂ M

6.3.2 Kümeleyici Sayısının Etkisi

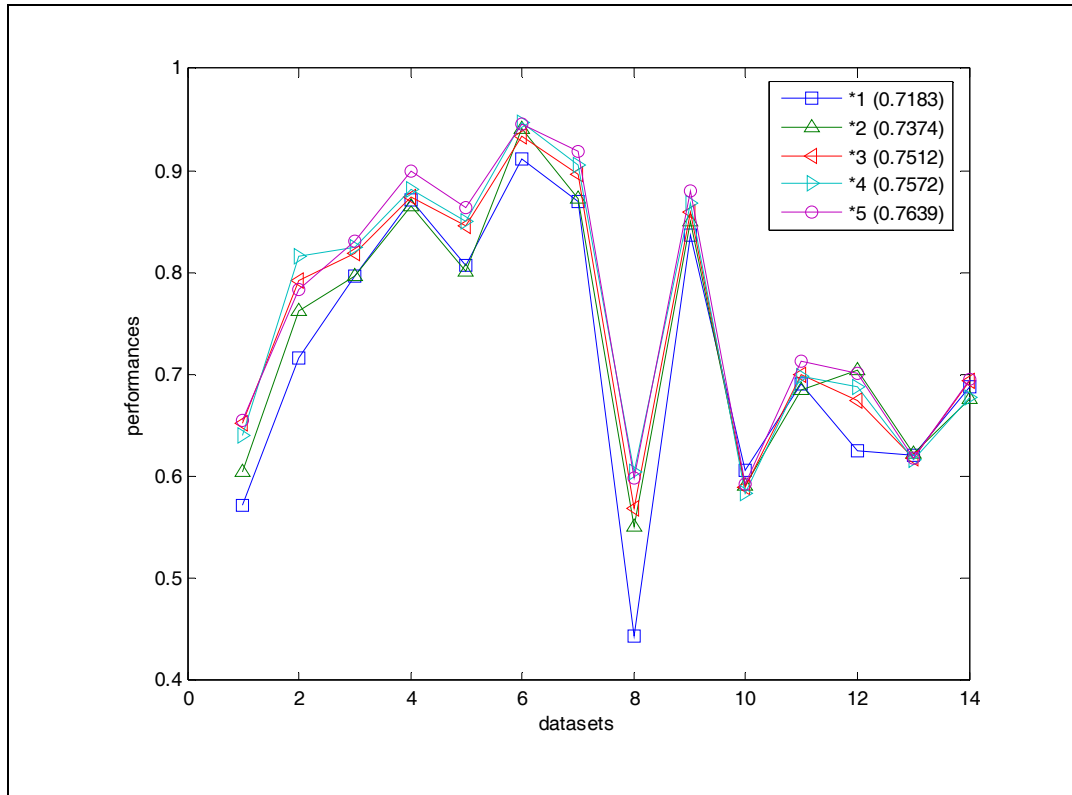
Kararları birleştirilen kümeleyici sayısının etkisinin de incelenmesi için Çizelge 6.3'teki 'waveform' haricindeki 14 veri kümesi üzerinde denemeler yapılmıştır. Şekil 6.2 ve 6.3'te kümeleyicilerin performansları ve standart sapmaları verilmiştir. Verilen değerler 14 veri kümesi üzerinde elde edilen değerlerin ortalamalarıdır. Şekil 6.2'de her kümeleyicinin kullandığı özellik sayısı $2 \cdot \log_2 M$; küme sayısı sınıf sayısının iki katıdır. Şekil 6.3'te özellik sayısı $\log_2 M$, küme sayısı sınıf sayısı ile aynıdır. Denemelerde kullanılan kümeleme algoritması Fuzzy Kmeans'dir. Kümeleyicilerin kararlarının birleştirilmesinde hiyerarşik metot kullanılmıştır.

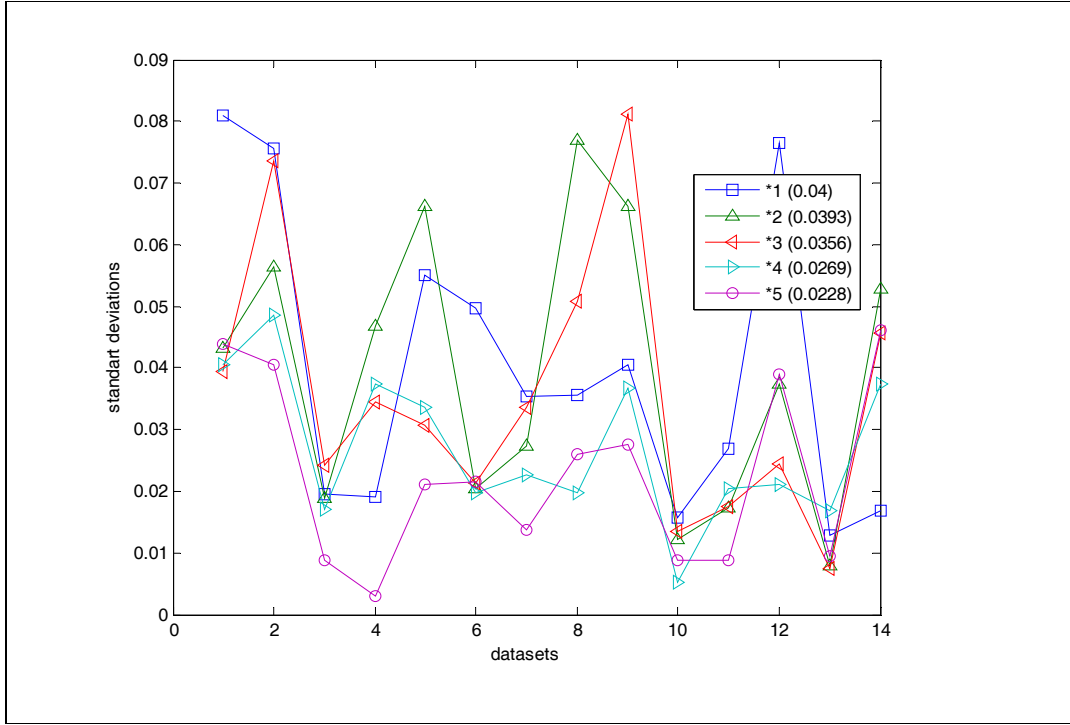


Her iki tür denemede de kararları birleştirilen kümeleyici sayısı arttıkça performans artmakta, standart sapma azalmaktadır.

6.3.3 Özellik Sayısı Etkisi

Kümeleyicilerde kullanılan özellik sayısının etkisinin daha iyi incelenebilmesi için ise aynı veri kümeleriyle denemeler yapılmış ve Şekil 6.4'teki sonuçlar elde edilmiştir. Özellik sayısının $\log_2 M$ 'in 1 katından 5 katına kadar olan değerleri için elde edilen performanslar ve standart sapmaları verilmiştir. Denemeler 14 veri kümesi üzerinde yapılmıştır. Her veri kümesi için yine 10'ar kez kümeleme yapılmış ve ortalama performanslar değerleri gösterilmiştir.





Şekil 6.4 Kümeleme komitelerinde özellik sayısının performansa ve standart sapmaya etkisi

Özellik sayısı arttıkça genel olarak performansın arttığı, standart sapmanın azaldığı görülmektedir.

6.4 Çıkarımlar

Sınıflandırma kararlarının birleştirilmesinin performansa olumlu katkılar yaptığını gören araştırmacılar aynı mantığın kümeleme algoritmaları içinde geçerli olabileceğini düşünmüşlerdir. Ancak kümeleme algoritmalarının sonuçlarının birleştirilmesi, sınıflandırmadaki kadar kolay değildir. Çünkü kümeleyici sonuçlarında küme sayısı sınıf sayısı ile aynı olmak zorunda değildir. Literatürde bu konuda çeşitli yöntemler önerilmiştir. Temel graf bölümlene ve hiyerarşik kümeleme olmak üzere iki tür yaklaşım mevcuttur. Genelde graf algoritmalarının daha başarılı sonuçlar ürettiği söylenmektedir.

Bu çalışmada bu iki türe ait metot 15 veri kümesi üzerinde 4 farklı kümeleme algoritması kullanılarak test edilmiştir. Kümeleme algoritmalarının, karar birleştirme algoritmalarının, kümeleme algoritmalarına verilen aynı veri kümesinden farklı alt kümeler oluştururken seçilen özellik sayısının etkileri incelenmiştir. Vardığımız sonuçlar elimizdeki veri kümeleri üzerinde aşağıdaki şekilde özetlenebilir:

- Kümeleme yaparken, karar birleştirme yapmak genelde daha iyi sonuçlar

 retmektedir. Ancak bu artış sınıflandırıcı algoritmalarındaki kadar belirgin deęildir.

- Ortalama olarak graf algoritmaları daha y ksek başarıya sahiptir ancak bireysel başarılarda hiyerarşik algoritmalar daha başarılıdır.
- Kmeans ve Fuzzy kmeans en başarılı k meleme algoritmalarıdır.
- Karar birleřtirmenin başarısını en fazla arttırdığı k meleme metodu eklemeli k melemedir. Bunun sebebi eklemeli k melemenin tekil başarısının dięerlerine g re daha d ř k olmasıdır. Ancak eklemeli k melemenin kararları birleřtirildiğinde bile dięer algoritmaların performansına ulařamamaktadır.
- K meleyicilerde kullanılan  zellik sayısının ve k meleyici sayısının arttırılması performansı da arttırırken standart sapmayı azaltmaktadır.
-  zellik sayısı fazla iken, k meleyici sayısı artarken performans daha hızlı y kselirken, standart sapma daha hızlı azalmaktadır.

6.5 Gelecek  alışmalar

Arařtırmaya a ık alanlar ve gelecekteki arařtırmacılar tarafından denenebilecek fikirler ařağıda listelenmiřtir:

- K meleme komitesinde sadece tek t r k meleme algoritması yerine farklı k meleyicilerin kullanılabilir. Bu metodun kararların ayrıklığını arttıracağından başarılı olacağı d ř n lmektedir.
- K meleme komitelerinin başarıları sınıflandırma performansına ek olarak dięer kriterlerle (DBİndeks, Siluet Geniřlięi) de  l  lebilir.
- Hiyerarşik k melemede k me tabanlı yaklařım da iřleme dahil edilebilir.  rneklerin birbirlerine uzaklıkları yerine k melerin birbirlerine uzaklıkları kullanılabilir. Her seferinde birbirine en yakın olan k meler birleřtirilir.
-  rneklerin uzaklıklarını ifade eden matriste birlikte ka  adet k me i inde ge tikleri yer alıyor. Bunun yerine farklı bir aęırlıklandırma kullanılabilir.
- T m  rneklerin k melerine bir seferde karar vermek yerine  nce kesinlikle aynı k melerde olanlara karar verilip daha sonra belirsizler ele alınabilir.
- Rasgele alt uzayların etkisi  l  ld ę  gibi bootstrapping etkileri de  l  lebilir.

7. ÖZELLİK SEÇİMİ

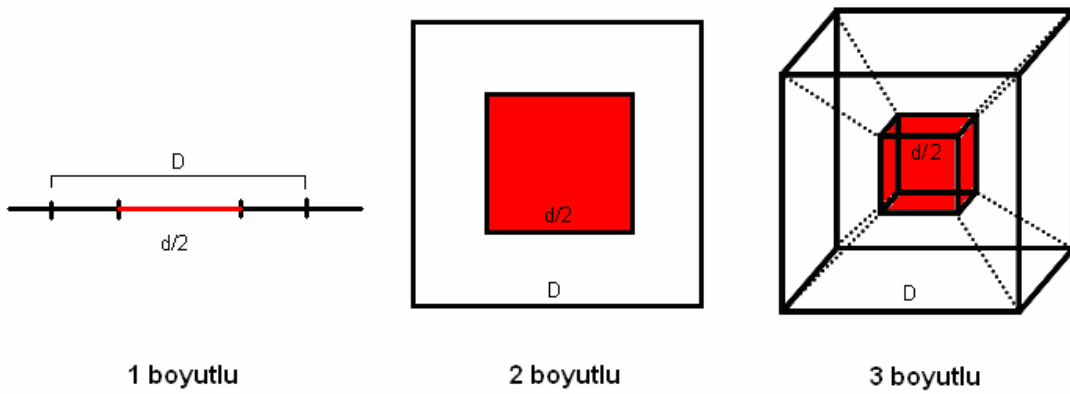
İlaç verilerinde genelde yüksek sayıda özellik bulunduğundan bu özelliklerden sonucu daha fazla etkileyenlerin bulunması (özellik seçimi) problemi de bu tez kapsamına girmektedir. Bunun için öncelikle mevcut özellik seçimi yaklaşımları incelenmiş ve seçilecek özellik sayısına kullanıcının karar verdiği ve WEKA yazılımı (Witten vd., 2005) içinde yer alan 6 tanesi yeni geliştirilen özellik seçimi metoduyla karşılaştırılmıştır.

7.1 Tanımlar

Özellik seçim algoritmalarının çeşitli kategorileri bulunmaktadır. Özelliklerin birer birer ele alındığı metotlar *filter* metotları, alt kümeler halinde ele alındığı metotlar ise *wrapper* metotları olarak isimlendirilir. Filtre metotlarının avantajı hızlı olmalarıdır. Ancak özelliklerin birlikte sınıfları / regresyon değerlerini belirledikleri veri kümelerinde başarıları düşüktür. Bu tür veri kümelerine XOR tipi veriler örnek verilebilir. Wrapper metotları ise büyük bir alt uzayı taradıkları için daha uzun zaman gerektirmektedirler.

7.2 Çok Boyutlu Verilerle Çalışmak

Tezin uygulama alanı olan ilaç verileri genelde çok boyutlu verilerden oluşmaktadır. Bu nedenle bu bölümde verilerin boyutunun artmasının sınıflandırma işlemine etkisi incelenmiştir. Şekil 7.1'de 1, 2 ve 3 boyutlu uzaylarda sınıfın sınırlarından çok sınıfın merkezine daha yakın olan noktalar gösterilmiştir.



Şekil 7.1 1, 2 ve 3 boyutlu uzaylarda merkeze yakın noktalar

Çizelge 7.1'de noktaların yüzde kaçının sınıf merkezine daha yakın olduğu verilmiştir.

Görüldüğü gibi noktaların boyutu arttıkça verilerin büyük bir kısmı sınırlara yakın yerlerde yer almaktadır. Şekil ve tabloda verilerin, uniform olarak dağıldığı kabul edilmiştir. Verilerin uniform değil de Gaussian dağıldığı kabul edilirse Çizelge 7.1'teki yüzdeler değişecektir ancak boyutun artmasıyla noktaların daha çok sınırlarda yer alması karakteristiği bozulmayacaktır.

Çizelge 7.1 Boyut sayısı ile merkeze yakın noktaların oranı arasındaki ilişki

Boyut Sayısı	Merkeze daha yakın noktaların oranı (%)
1	50
2	25
3	12,50
...	...
P	$(\frac{1}{2})^P$

Çizelge 7.1'de görüldüğü gibi çok boyutlu verilerle sınıflandırma yapıldığında verilerin çok büyük bir kısmı sınıfları ayıran sınırlara çok yakın yerlerde bulunacağından sınıflandırma yapmak boyut sayısı arttıkça zorlaşmaktadır. Aynı yargıya çok boyutlu uzaydaki verileri temsil etmek için gerekli örnek sayısının uzayın boyutuyla olan ilişkisi incelendiğinde de ulaşılmaktadır [9, 10].

Tek boyutlu uzayda $[0,1]$ aralığı ele alındığında, bu uzayı yeterli bir şekilde temsil eden nokta sayısı 10 kabul edilsin.

Bu uzaydaki rasgele bir noktanın, uzayı temsil eden noktalardan en yakın olanına ortalama uzaklığı 0.5 olacaktır.

İki boyutlu uzayda rasgele bir noktanın en yakın noktaya olan ortalama uzaklığının düşey ya da dikey 0.5 olması için (bir önceki uzayla aynı kalitede temsil edilebilmesi için) gerekli temsilci nokta sayısı 100 olacaktır. Üç boyutlu bir uzay için aynı kalitede temsil için gerekli nokta sayısı 1000'dir. Çizelge 7.2'de boyut sayısı ile gerekli nokta sayısı arasındaki ilişki gösterilmiştir.

Çizelge 7.2 Boyut sayısı ile uzayı yeterli bir biçimde ifade etmek için gerekli örnek sayısı arasındaki ilişki

Boyut Sayısı	Gerekli temsil eden nokta sayısı
1	10
2	100
3	1000
...	...
p	10^p

Çizelge 7.2’de görüldüğü gibi boyut sayısı, gerekli nokta sayısının üssel katıdır.

Çizelge 7.1 ve 7.2 aynı sonuca işaret etmektedir. Boyut sayısı arttıkça hem veriler sınırlara yaklaşıyor hem de doğru sınıflandırma yapmak için gereken örnek sayısı artıyor. Her iki faktörde yüksek boyutlu uzaylarda sınıflandırma yapmayı zorlaştırmaktadır.

7.3 Önceki Çalışmalar

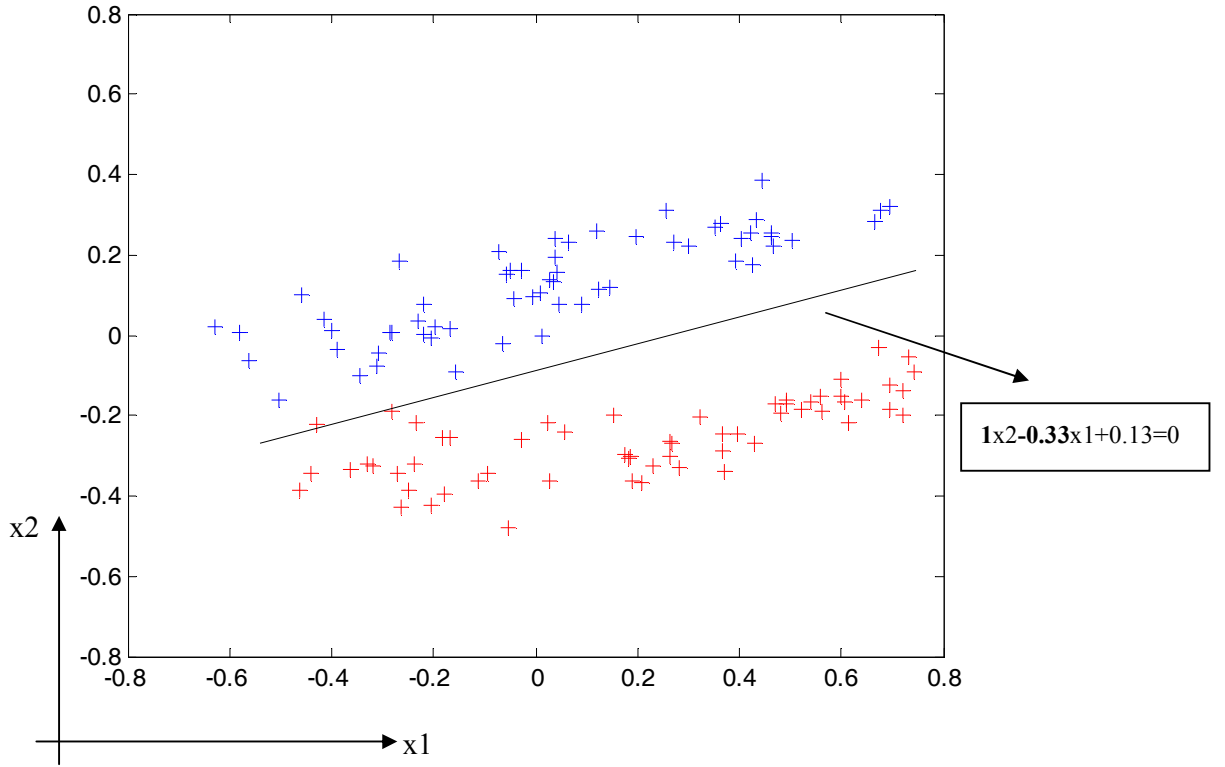
Kullanıcının seçilecek özellik sayısını belirleyebildiği 6 adet özellik seçimi metodu (ReliefAttributeEval, InfoGainAttributeEval, GainRatioAttributeEval, SymmetricalUncertAttributeEval, OneRAttributeEval, ChiSquaredAttributeEval) karşılaştırma yapmak için kullanılmıştır. Tüm metotlar WEKA (Witten vd., 2005) yazılımında yer almaktadır. Metotların ayrıntıları için WEKA’nin dokümantasyonu incelenebilir.

7.4 Gerçekleştirilenler

Tezin önceki bölümlerinde yer alan sınıflandırma ve kümeleme problemlerinde ağaç yapılarının gösterdikleri yüksek performans nedeniyle özellik seçiminde de kullanılabilecekleri düşünülmüştür ve özellik seçim ormanları geliştirilmiştir.

7.4.1 Özellik Seçim Ormanları

Sınıflandırmada kullanılan ormanların özellik seçiminde de kullanılabilirliği araştırılmış ve karar ağaçlarında bulunan hiper düzlem parametrelerinin özelliklerin önemiyle doğru orantılı olduğu görülmüştür. Bu yaklaşım literatürde SVMeval (Guyon vd., 2002) özellik seçimi metodunda da kullanılmaktadır.



Şekil 7.2 Sınıflandırıcı parametreleriyle özelliklerin ağırlıklarının ilişkisi

Şekil 7.2’de sınıfları birbirinden ayıran doğrunun parametreleri incelendiğinde sınıfları birbirinden ayırmada daha fazla etkili olan boyutun daha büyük katsayıya sahip boyut olduğu görülmektedir.

SVMeval’de tek bir hiper düzlem bulunup onun parametreleriyle özelliklerin önemleri belirlenmektedir. Cline karar ağaçlarında ise birçok hiper düzlem bulunmaktadır. Ayrıca Cline karar ormanlarında birçok ağaç bulunmaktadır. Yeni metotla, hiper düzlem parametrelerinden özellik seçimi yapılırken 2 adımlı bir işlem yapılır:

1. Her bir ağaçtaki hiper düzlem parametreleri kullanılarak kullanıcının istediği adet özellik döndürülür.
2. Ormandaki tüm ormanlardan gelen seçilmiş özelliklerden en fazla seçilmiş olanlardan kullanıcının istediği adedi seçilen özellikler olarak döndürülür.

Her bir ağaçtan özellik seçimi için iki metot geliştirilmiştir:

- Ağacın her düğümünde bulunan hiper düzlemlerin parametrelerinin mutlak değerlerinin ortalaması alınarak özellikler, parametrelerin mutlak değerlerine

göre büyükten küçüğe sıralanıp, en büyük katsayıya sahip olanlar seçilir.

- Ağacın kök düğümünde bulunan hiper düzlemin parametrelerinin mutlak değeri büyükten küçüğe sıralanıp, en büyük katsayıya sahip olanlar seçilir.

Ayrıca büyük özellik sayısına sahip veri setleri için ağaçların oluşturulması uzun zaman aldığından önce diğer özellik seçimi metotlarıyla (InfoGain, GainRatio) eleme yapılmış ve geri kalan özelliklerden kullanıcının istediği kadarı seçilmiştir.

En son olarak hiper düzlemin parametre büyüklüklerinin, böldükleri örnek sayısı ile ağırlıklandırılması denendi ancak bu yolla da istenilen başarıya ulaşılamadı.

7.5 DeneySEL Sonuçlar

Özellik seçimi denemelerinde kullanılan veri kümeleri ve özellikleri Çizelge 7.3'te verilmiştir.

Çizelge 7.3 Özellik seçiminde kullanılan veri kümeleri

Veri seti	Sınıf sayısı	Örnek sayısı	Özellik sayısı	Seçilen özellik sayısı
Amlall	2	72	7129	10
Ann	3	3772	21	7
bi75ds3	9	315	470	11
derma	6	286	34	10
gkanser	2	456	30	8
Hava	2	281	34	11
Pima	2	388	8	3
Seg	7	210	19	6
Wine	3	118	13	6
Colon	2	62	2000	10
Mill	3	57	12582	11
Nerv	2	60	7129	10
Spam	2	4601	57	11

Çizelge 7.3'teki veri kümeleri üzerinde mevcut 6 ve yeni geliştirilen 2 özellik seçim metodu uygulanmış ve veri kümelerinin özellik sayıları azaltılmıştır. Her metotla her veri kümesi için eşit sayıda özellik seçilmiştir. Veri kümelerinin orijinal ve seçilen özellik sayıları Çizelge 7.3'teki son iki sütunda verilmiştir. Azaltılmış özellik sayısına sahip veri kümeleri 5 farklı sınıflandırma metoduyla 10'lu çapraz geçerleme uygulanarak sınıflandırılmış ve ortalama başarıları Çizelge 7.4'te verilmiştir. Ayrıca başarı sıralamalarının ortalaması da (rank) verilmiştir. Kullanılan sınıflandırma metotları yine Weka içinde yer alan Naive Bayes, K En

Yakın Komşu, Karar Destek vektörleri, Karar Ağaçları (C4.5) ve Karar Ormanları (Random Forest)’dır. Çizelgedeki her değer 50 (5 algoritma * 10 cv) değerinin ortalamasıdır.

Çizelge 7.4 Özellik seçim metotlarının karşılaştırılması

	Özellik seçimsiz	RAE	InfoGain	GainRatio	SUAE	OneRA	ChiKare	Karar ormanlı özellik seçimi
amlall	88	92.678	91.606	91.892	91.608	91.322	91.32	94.44
ann	96.12	96.354	96.622	96.878	96.634	96.224	96.62	95.456
bi75ds3	75.008	66.16	65.88	61.306	65.628	60.58	66.65	50.154
derma	97.416	77.152	76.942	84.416	75.056	75.056	84.49	90.902
gkanser	94.854	95.162	93.758	93.674	93.892	93.806	93.89	91.488
hava	88.25	90.47	88.668	90.314	89.678	90.106	88.52	88.18
pima	75.26	75.77	76.13	76.13	76.13	75.77	75.77	75.202
seg	86.762	75.714	75.904	85.336	75.524	70.574	75.81	80.4736
wine	96.454	97.64	97.64	97.64	97.64	96.214	97.64	93.896
colon	76.522	81.728	84.738	85.864	83.88	84.962	83.4	84.19
mll	85.268	91.688	90.774	94.36	90.254	92.694	90.57	90.872
nerv	61.666	73.1	79.766	69.968	73.168	73.834	76.63	76.996
spam	89.618	79.538	88.538	88.154	89.826	87.878	87.74	86.856
ortalama	85.477	84.089	85.151	85.841	84.532	83.771	85.312	84.55
rank	4.25	3.62	4.5	3.37	4.25	5.87	4.37	5.75

Çizelge 7.4’te geliştirilmiş olan 2 farklı özellik seçimi versiyonundan en yüksek başarıya sahip olanının (sadece kök düğümlerdeki katsayıların kullanımı) sonuçları verilmiştir. Çizelgede her bir veri kümesinde en yüksek başarıyı gösteren metodun sonucu kalın olarak verilmiştir.

Çizelge incelendiğinde hem ortalamalarda hem ranklarda, özellik seçimi metotları arasında çok büyük performans farklılıkları olmadığı görülmektedir. Ayrıca 4 veri kümesinde özellik seçiminin başarıyı azalttığı görülmektedir. Özellik seçimi metotları arasında ortalama olarak da rank olarak da en başarılısı GainRatio’dur ve özellik seçimi yapılmadığında elde edilen ortalama başarıdan daha yüksek başarı elde eden tek metottur.

Yeni geliştirilen karar ormanlarıyla özellik seçimi metodunda 100 adet karar ağacından oluşan bir orman kullanılmış ve ağaçların sadece kök düğümlerindeki hiper düzlemlerin parametreleri kullanılmıştır. Bu metot, ancak infoGain metodundan daha iyi performans gösterirse başarılı olduğu kabul edilebilir. Çünkü özellik seçilirken zamandan kazanmak için InfoGain ile ön eleme yapılarak kullanıcının istediğinin iki katı özellik seçilmiş ve karar ormanı metoduyla onların arasından kullanıcının istediği kadarı seçilmiştir. Yeni metodun InfoGain’in başarısından daha düşük başarıya sahip olması InfoGain’in seçtiği istenileni iki katı adet özellikten istenilen adedini seçmede daha başarısız olduğunu göstermektedir.

7.6 Çıkarımlar

Bu bölümde “Özellik seçimi karar ağaçlarıyla / ormanlarıyla yapılabilir mi?” sorusuna cevap aranmıştır. Ağaçların karar düğümlerindeki hiper düzlem parametrelerinin mutlak değerce büyüklüğünü kullanan bir metot geliştirilmiştir. Geliştirilen metotla, 6 özellik seçimi algoritması 13 sınıflandırma veri kümesi üzerinde karşılaştırılmıştır. Her bir veri kümesinin önce 7 özellik seçimi metoduyla boyutları indirgenmiş daha sonra 5 farklı sınıflandırma algoritmasının bu veri kümeleri üzerindeki performansları (10 CV) ölçülmüştür. Bu denemelerden elde edilen sonuçlar aşağıda özetlenmiştir:

- Özellik seçimi metotları arasında çok büyük performans farklılıklarının olmadığı görülmüştür.
- Özellik seçimi 13 veri kümesinden 4’ünde başarıyı azaltmıştır.
- Özellik seçimi metotları arasında en başarılısı GainRatio’dur ve özellik seçimi yapılmadığında elde edilen ortalama başarıdan daha yüksek başarı elde eden tek metottur.
- Yeni geliştirilen metot sadece tek bir veri kümesinde başarılı sonuçlar vermiştir ve geliştirilmeye ihtiyacı vardır. Bununla birlikte, geliştirilen metodun işlem zamanının diğer metotlara göre çok fazla olması, metodun dezavantajıdır. Bundan kurtulmak için hızlı özellik seçicilerle (filtre) bir ön eleme yapıp, daha sonra karar ormanlarıyla özellik seçimi uygulanmalıdır.

Gelecek çalışma olarak, tek değişkenli bir Cline ağacı oluşturup düğümlerde en çok yer alan özellikler seçilebilir. Seçilecek özellik sayısı ağaçta yer alanların hepsi/yarısı olabilir ya da kullanıcının istediği kadarı olabilir.

8. REGRESYON

İlaç tasarımı problemlerinin büyük çoğunluğu eğri uydurma (regresyon) türünden problemlerdir. Bu nedenle regresyon konusu da bu tezin kapsamına alınmıştır.

Regresyon problemlerinde veri kümesi Çizelge 8.1’de gösterildiği gibi her bir örneğe ait özellikler ve bu örneğin sonuç değerini içerir.

Çizelge 8.1 Örnek regresyon verisi

	özellik1	özellik2	özellik3	Y
1	23	12	34	2.1
2	34	56	2	5.9
3	3	6	1	2.5
4	8	9	8	3.13
5	5	4	9	4.2
6	3	99	23	3.8

Çizelge 8.1’de yer alan veri kümesinde 6 adet örneğe ait 3 adet özellik değerleri (X) ve sonuç (Y) değerleri gösterilmiştir. Regresyon problemlerinde amaç, mevcut X değerlerini kullanarak daha sonra karşılaşılabilecek X değerlerine karşılık gelen Y değerlerinin tahmin edilmesidir.

Problemin en yaygın ve son kullanıcı tarafından en kolay anlaşılabilir çözümü doğrusal (lineer) modellerle yapılmaktadır. Lineer çözümünün genel yapısı Eşitlik 8.1’de verilmiştir. Lineer modelde her bir özelliğin belirli bir katsayıyla çarpılarak toplanmasıyla Y değerinin elde edilebileceği varsayımına dayanılır. Lineer modellerin sonuçların yorumlanması oldukça kolaydır. Çıkış işaretine giriş işaretlerinden her birinin ne ölçüde ve ne yönde etki ettiği b vektörüne bakılarak kolayca görülebilmektedir.

$$y = x_1.b_1 + x_2.b_2 + x_3.b_3 + \dots + x_n.b_n$$

$$y = X.b \quad (8.1)$$

Eşitlik 8.1’deki b değişkeninin bulunması için birçok metot geliştirilmiştir. Bölüm 8.2’de bu metotlara değinilmiştir.

8.1 Tanımlar ve Kullanılan Semboller

Bu bölümde literatürde ve bu tezde regresyon ile ilgili olarak kullanılan kavramlar

açıklanmıştır.

8.1.1 Entropi

Bir dağılımın entropisi;

$p(s_i)$: s_i 'nin görülme/ oluşma / var olma olasılığı

N : s_i 'nin aldığı farklı değer sayısı

$\sum_{i=1}^N p(s_i) = 1$ olmak üzere Eşitlik 8.2'de verilmiştir (Jaynes, 1957).

$$H(s) = -\sum_{i=1}^N p(s_i) * \log_2 p(s_i) \quad (8.2)$$

8.1.2 Ekstrem ve Aykırı Örnekler

Bir veri kümesinin içindeki, çıkış değerleri (y) ekstrem ya da aykırı olup olmadıklarını belirlemek için kullanılan kurallar sırasıyla Eşitlik 8.3 ve 8.4'te verilmiştir [11].

$Q1$ = Tüm y değerlerinin en düşük %25'lik bölümünün sınır değeri

$Q3$ = Tüm y değerlerinin en düşük %75'lik bölümünün sınır değeri

$IQR = Q3 - Q1$

OF = Aykırılık parametresi (bu çalışmada kullanılan değeri = 3)

EVF = Ekstremlik parametresi (bu çalışmada kullanılan değeri = 6)

olmak üzere:

y değeri aykırıdır eğer:

$$(Q3 + OF * IQR < y \leq Q3 + EVF * IQR) \text{ Ya da } (Q1 - EVF * IQR \leq y < Q1 - OF * IQR) \quad (8.3)$$

y değeri ekstremdir eğer:

$$(y > Q3 + EVF * IQR) \text{ ya da } (y < Q1 - EVF * IQR) \quad (8.4)$$

8.1.3 Colinerity

Bu terim, sözlükte “örneklerin aynı doğru üzerinde bulunması” olarak tanımlanmıştır. İki boyutlu bir uzayda (iki özellikli) örneklerin aynı çizgide bulunması bir problem değildir.

8.1.4 Regresyonda Hata Ölçüm Metotları

Regresyon problemleri için üretilen modelin performansını ölçmek için literatürde birçok kriter mevcuttur. Bu bölümde en popüler olanlarına değinilmiştir [12].

y_i : i. örneğin gerçek çıkış değeri

t_i : i. örneğin tahmin edilen çıkış değeri

N : örnek sayısı

y^* : örneklerin gerçek çıkışlarının ortalaması

t^* : örneklerin tahmin edilen çıkışlarının ortalaması

Ortalama Karesel Hata (Mean Squared Error-MSE): gerçek çıkışların, tahmin edilen çıkışlara karesel uzaklıklarının ortalamasıdır.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - t_i)^2 \quad (8.5)$$

Ortalama Karesel Hatanın Karekökü (Root Mean Squared Error-RMSE): MSE'nin kareköküdür.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - t_i)^2} \quad (8.6)$$

Ortalama Mutlak Hata (Mean Absolute Error-MAE): Gerçek çıkışların, tahmin edilen çıkışlara mutlak uzaklıklarının ortalamasıdır. Büyük hatalı örneklere ortalama karesel hatalardan daha az hassastır.

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - t_i| \quad (8.7)$$

Rölatif Mutlak Hata (Relative Absolute Error-RAE): Mutlak hatanın, gerçek çıkışların ortalamalarına mutlak uzaklıklarının toplamına oranıdır.

$$RAE = \frac{\sum_{i=1}^N |y_i - t_i|}{\sum_{i=1}^N |y^* - y_i|} \quad (8.8)$$

Rölatif Karesel Hata (Relative Squared Error-RSE): Karesel hatanın, gerçek çıkışların ortalamalarına karesel uzaklıklarının toplamına oranıdır.

$$RSE = \frac{\sum_{i=1}^N (y_i - t_i)^2}{\sum_{i=1}^N (y^* - y_i)^2} \quad (8.9)$$

Rölatif Karesel Hatanın Kökü (Root Relative Squared Error-RRSE): RSE'nin kareköküdür.

$$RRSE = \sqrt{\frac{\sum_{i=1}^N (y_i - t_i)^2}{\sum_{i=1}^N (y^* - y_i)^2}} \quad (8.10)$$

Korelasyon Katsayısı (Corelation Coefficient-CC): örneklerin gerçek çıkışlarıyla tahmin edilen çıkışları arasındaki korelasyon katsayısıdır.

$$CC = \frac{S_{yt}}{\sqrt{S_y S_t}},$$

$$S_{yt} = \frac{1}{N-1} \sum_{i=1}^N (t_i - t^*)(y_i - y^*), \quad S_y = \frac{1}{N-1} \sum_{i=1}^N (y_i - y^*)^2, \quad S_t = \frac{1}{N-1} \sum_{i=1}^N (t_i - t^*)^2 \quad (8.11)$$

8.2 Önceki Çalışmalar

Bu bölümde regresyon problemleri için geliştirilmiş temel metotlar anlatılmıştır.

8.2.1 Çoklu Lineer Regresyon (Multiple Linear Regression - MLR)

Lineer (doğrusal) regresyon modelinin en temel çözümü çoklu lineer regresyon metodudur. Bu metotta Eşitlik 8.1'deki b değişkeni yalnız bırakılarak problem çözülür. Eşitlik 8.12'de çözüm verilmiştir.

$$y = X.b + hata$$

$$X'.y = X'.X.b$$

$$(X'.X)^{-1}.X'.y = (X'.X)^{-1}.(X'.X).b \quad (8.12)$$

$$(X'.X)^{-1}.(X'.X) = 1$$

$$b = (X'.X)^{-1}.X'.y$$

Çözüm gayet basit olmakla birlikte her türlü probleme uygulanamamaktadır. X matrisinin rankının örnek sayısından az olduğu durumlarda matrisin tersinin hesaplanması oldukça zordur. Bununla birlikte eğer örnek sayısı boyut sayısından az ise (ki ilaç tasarımı problemlerinin neredeyse tamamında bu durum söz konusudur) matrisin tersi tanımlı bile değildir. Çoklu lineer regresyon gerçek veri kümelerinde bu tür problemler yüzünden uygulanamamaktadır.

8.2.2 Temel Bileşen Tabanlı Regresyon (Principal Component Regression -PCR)

Temel bileşen tabanlı regresyon metodunun temel mantığı verileri matrisin tersinin kolaylıkla bulunabileceği bir uzaya yansıtarak problemi başka bir uzayda çözmektir. Eşitlik 8.13'te PCR'ın regresyon problemini nasıl modellediği görülmektedir.

$$y = X.b = X.P.q \quad (8.13)$$

veriler P matrisiyle çarpılarak yeni bir uzaya izdüşürülmekte ve daha sonra q değişkeni bulunarak regresyon problemi çözülmektedir. Yeni uzayın bulunması için Temel Bileşen Analizi (Principal Component Analysis- PCA) kullanılmakta ve veriler en yüksek özdeğerli(eigenvalue) özvektörlerin(eigenvector) tanımladığı uzaya dönüştürülmektedirler. Bu uzayda özellikler arası korelasyon olmadığından matris tersi alma işlemi kolaylıkla gerçekleştirilebilmektedir. Eşitlik 8.14'te verilerin yeni uzaydaki değerleri T ile ifade edilmektedir.

$$T = X.P$$

$$y = T.q + hata$$

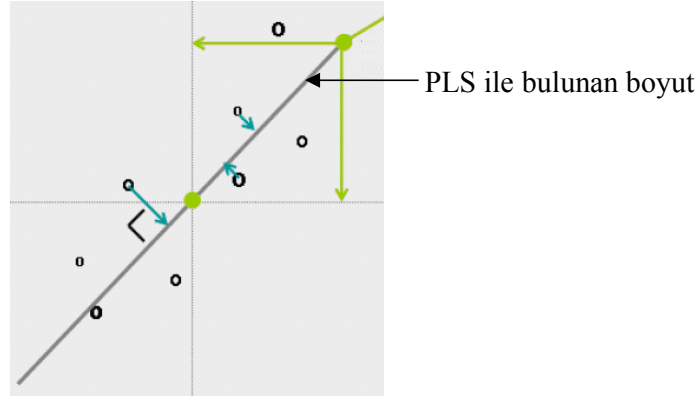
$$T'.y = T'.T.q$$

$$(T'.T)^{-1}.T'.y = (T'.T)^{-1}.(T'.T).q \quad (8.14)$$

$$(T'T)^{-1} \cdot (T'T) = 1$$

$$q = (T'T)^{-1} \cdot T' \cdot y$$

Özvektörler uzayındaki veriler orijinal verilerden daha az sayıda boyuta sahiptirler. Temel bileşen analizi sayesinde veri içindeki varyansın en büyük olduğu yönler (boyutlar) bulunmuş ve verinin boyutu azaltılmıştır. Şekil 8.2’de iki boyutlu örnek bir veri kümesinin en fazla varyansa sahip boyuta (orijinden geçen doğru) yansıtılması/izdüşürülmesi görülmektedir.

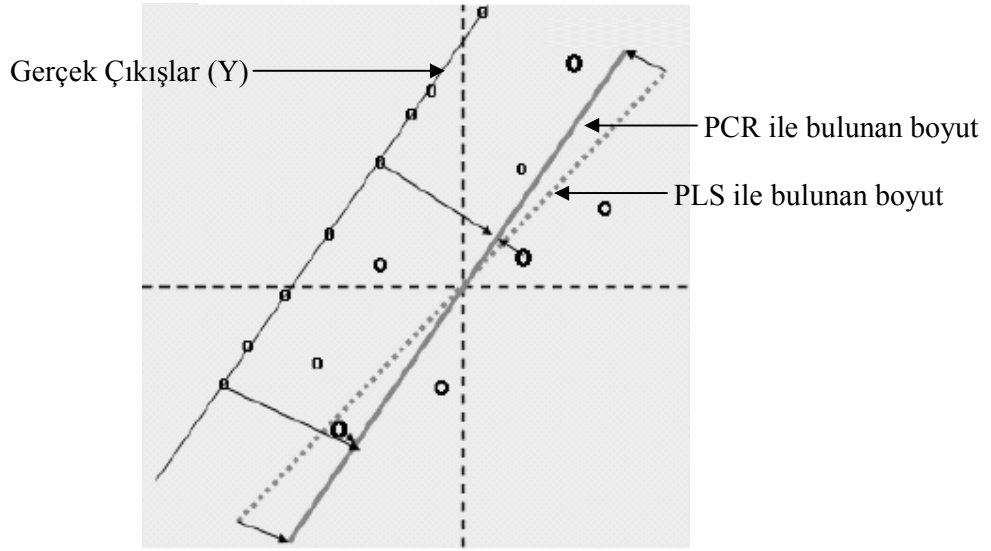


Şekil 8.2 Verilerin PCA ile 2 boyutlu uzaydan 1 boyutlu uzaya dönüşümü

Verinin boyutu azaltıldıktan sonra Eşitlik 8.14’teki q kolaylıkla bulunur ve regresyon için lineer model bulunmuş olur.

8.2.3 Parçalı En Küçük Kareler (Partial Least Squares - PLS)

Verinin boyutunun indirgenmesinde sadece varyansı büyük boyutlara bakılıyor olması matris tersi alma işlemini kolaylaştırmaktadır ancak bulunan yeni boyutların çıkış işaretini (Y) tahmin etmeye uygun olduğunun bir garantisi yoktur. Bu durumda hem öz vektörlerin hem de Y’nin yönünü göz önüne alarak işlem yapılırsa daha doğru tahminlere gidilebilir. Bu görüş “Parçalı En Küçük Kareler” metodunun temel fikridir. Şekil 8.3’te, Şekil 8.2’deki veri kümesi için PCA’nın ve PLS’in bulduğu yönler gösterilmiştir.



Şekil 8.3 PLS ve PCA'in bulduğu boyutlar

Şekil 8.3'te PCA'nın bulduğu boyut orjinden geçen noktalı çizgiyle, PLS'nin bulduğu yön düz çizgiyle gösterilmiştir. PLS, hem PCA'ni bulduğu yönü hem de Y çıkış işaretinin yönünü dikkate alarak iz düşüm yapılacak boyutu belirlemektedir. Eşitlik 4.15'te PLS'in çözümü verilmiştir.

$$\max(\text{cov}(t, y) \mid X \cdot w = t, \|w\| = 1)$$

$$X = t \cdot p' + \text{hata}$$

$$y = t \cdot q' + \text{hata} \quad (8.15)$$

$$y = X \cdot b = X \cdot W \cdot (p' \cdot W)^{-1} \cdot q$$

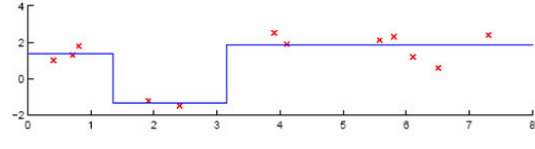
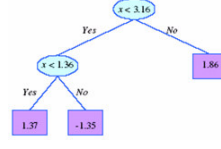
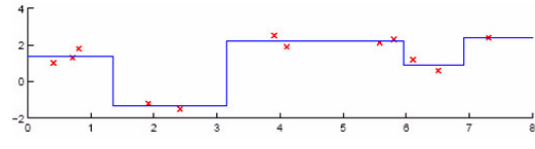
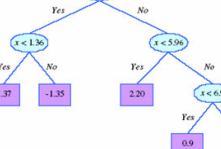
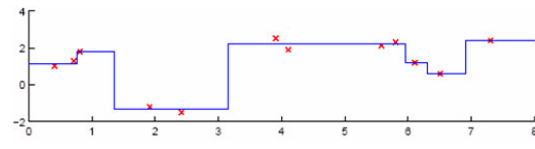
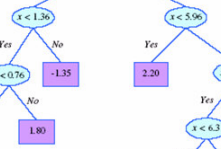
PLS metodu bir taraftan (matris test alma işleminin kolaylıkla yapılabilmesi için) X'teki varyansın büyük olduğu yönleri bulurken, diğer taraftan da Y ile korelasyonu maksimum yapmaya çalışır. www.models.kvl.dk/courses/IntroMatlab/IntroMatlab.pdf adresinden alınan bilgilere göre PLS, PCR'dan daha basit modeller bulabilmektedir ki bu da modelin genelleştirebilme kabiliyetini arttıran bir unsurdur.

8.2.4 Regresyon Ağaçları

Genelde sınıflandırmada kullanılan karar ağaçları regresyon problemlerinde de kullanılabilecek şekilde çeşitli şekillerde düzenlenmişlerdir. Sınıflandırma ağaçlarında genel olarak her boyutta karar sınırının geçebileceği en uygun değer sınıflandırma başarısına göre aranır. Regresyon ağaçlarında ise karar sınırı bölünen iki bölgeden geçirilecek fonksiyonların

MSE'leri toplamalarının minimum olmasına göre aranır (Breiman vd., 1984). Ağacın yapraklarındaki modellerin hata değerleri belirli bir eşik değerinin altına düşene dek, ağacın büyütülmesine (uzayın alt uzaylara bölünmesine) devam edilir. Yapraklarda sabit bir sayı, doğrusal ya da doğrusal olmayan bir model yer alabilir.

Şekil 8.4'te [13] tek özellikli bir veri kümesinde farklı hata eşik değerleri için oluşturulmuş çeşitli regresyon ağaçları ve sonuçları verilmiştir. Yapraklarda sabit sayılar yer almaktadır. Bir yapraktaki hata bulunurken, o yapraktaki örneklerin çıkış değerlerinin ortalamalarına olan karesel uzaklıklarının toplamı kullanılmıştır.

Yapraklardaki hata	Üretilen modelin çıkışı	Regresyon ağacı
Çok		
Orta		
Az		

Şekil 8.4 Farklı hata eşikleri için oluşturulmuş regresyon ağaçları ve çıkışları

Şekil 8.4 incelendiğinde, ağaçların yapraklarındaki hata değerinin azaldıkça ağacın büyüklüğünün arttığı görülmektedir.

8.2.4.1 M5 Model Ağaçları

Quinlan [14] tarafından geliştirilen, yapraklarında lineer modeller bulunan bir regresyon ağacı algoritmasıdır. Veri kümesi yine uzayda alt kümelerle bölünür ve her bir alt uzaya bir model atanır. Alt uzayların sınırları bulunurken örneklerin çıkışlarının standart sapmasını en fazla

azaltan özellik, değer ikilisi kullanılmaktadır. SDR'si (Eşitlik 8.16) en fazla olan özellik, değer ikilisi (i, θ) karar düğümüne yerleştirilir.

$$SDR(i, \theta) = sd(X_i) - \sum_{i \in \{+, -\}} \frac{|X_i^h|}{|X_i|} sd(X_i^h) \quad (8.16)$$

X_i : X veri kümesinin i . özelliklerinin değerleri

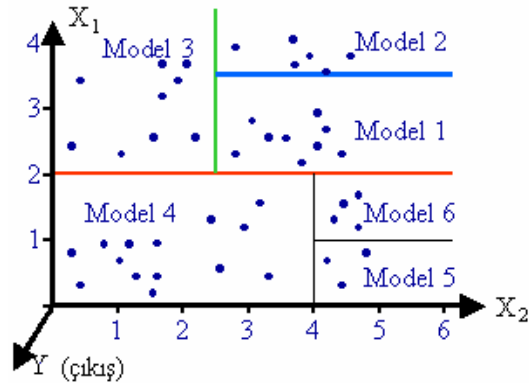
X_i^+ : X veri kümesinin i . özelliği θ 'dan büyük olanlarının değerleri

X_i^- : X veri kümesinin i . özelliği θ 'dan küçük olanlarının değerleri

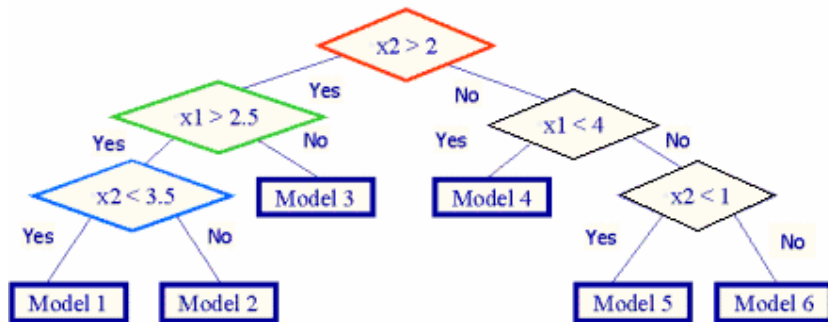
$sd(K)$: K değerlerinin standart sapması

$|K|$: K kümesinin eleman sayısı

Şekil 8.5 ve 8.6'da [14] örnek bir m5 model ağacı ve alt uzayları görülmektedir.



Şekil 8.5 Örnek bir M5 model ağacının alt uzay sınırları



Şekil 8.6 Örnek bir M5 model ağacı

8.2.4.2 M5' Model Ağaçları

M5 model ağaçlarının sınıflandırma da yapabilen bir versiyonudur (Wang vd., 1997). WEKA kütüphanesinde M5P adı ile kullanılmaktadır.

8.2.4.3 SMOTI

Geleneksel regresyon ağaçlarında, lineer modeller sadece ağacın yapraklarında yer almaktadır. SMOTI algoritması ise diğer düğümlerde de lineer modeller bulundurabilmektedir. Çalışmanın ayrıntıları için referansa bakılabilir (Malerba vd., 2004).

8.2.5 Örnek Tabanlı Algoritmalar

Herhangi bir eğitim işlemine ihtiyaç duymayan algoritmalarlardır. En büyük dezavantajları ise yüksek depolama alanı gerektirmeleridir.

Örnek tabanlı algoritmaların temel kabulleri, benzer örneklerin benzer çıkışlara sahip olduğudur. Bir test örneğinin çıkışı, o örneğe en benzeyen eğitim örneklerinin çıkışları kullanılarak bulunur. Bu işlem iki alt seçim içerir. İlki örneklerin birbirlerine benzerliklerinin nasıl bulunacağı diğeri ise test örneğine benzer eğitim örneklerinin çıkışlarının nasıl birleştirileceğidir.

K en yakın komşu algoritması, kullanıcıdan test örneğine en yakın kaç eğitim örneğinin kullanılacağını ifade eden K değerini alır, örneklerin benzerliklerini Öklid ölçütüne göre belirler ve çıkışları en yakın K örneğin çıkışlarının ortalamasını alarak bulur. Sınıflandırma için kullanılıyorsa K adet eğitim örneğinin en çok dahil olduğu sınıfı test örneğinin sınıfı olarak belirler.

En yakın komşu algoritması bu alandaki ilk algoritmadır ve test örneğine sadece en benzeyen eğitim örneğini kullanarak ($K=1$) işlem yapar (Cover vd., 1967).

Literatürde, örnek tabanlı algoritmaların birçok versiyonu mevcuttur. Depolama maliyetini azaltmak ve gürültülü veri kümelerinde daha iyi sonuçlar almak için tüm eğitim örnekleri yerine bir alt kümesinin kullanımı önerilmiştir (Hart, 1968), (Gates, 1972).

“Instance - based Learning Algorithms” adlı çalışmada (Aha vd., 1991) tüm özelliklerin skalasını eşitleyerek her bir özelliğin sonuç üzerindeki etkilerini eşitlenmiştir. Daha sonra tüm örnekler yerine sadece yanlış sınıflandırılan örneklerin depolanması önerilmiştir.

Kstar algoritması ise, örnek benzerliğinde entropiyi kullanır. Her bir özellik için benzerlikleri (birinden diğerine rasgele dönüşüm olasılığı) ayrı ayrı bulur ve özelliklerin sonuçlarının

birleştirirken olasılıklarının çarpımını kullanır. Dönüşüm işlemi için önceden tanımlanmış operatörler mevcuttur (Cleary vd., 1995).

8.3 Gerçekleştirilenler

Bu bölümde regresyon için geliştirilen çeşitli metotlar, özellikleri ve algoritmaları verilmiştir. Görsel anlaşılabilirliğin artırılması için bütün algoritmalar 1 boyutlu veriler üzerinden anlatılmıştır. Metodun gerçek verilerle kullanılmasında her boyut için önerilen algoritma ile bir çıkış değeri üretilmekte ve ortalamaları alınarak sonuç çıkış değeri olarak hesaplanmaktadır.

8.3.1 Kmeans ile Regresyon

Orijinal Kmeans metodunda merkezlerde kendi kümesine ait (kendisine diğer merkezlerden daha yakın) örneklerin ortalaması bulunur. Geliştirilen metotta ise buna ek olarak, kendi kümesine ait verilerin lineer modelini taşımakta ve bu sayede regresyon görevini de yapabilmektedirler. Merkezlerin lineer modelleri Çoklu Lineer Regresyon ile bulunmaktadır.

Algoritma.Kmeans Regresyon. Eğitim:

1. Merkezleri rasgele ata.
2. Karesel hata bir eşik değerinin altına ininceye kadar ya da maksimum tekrar sayısına erişilene kadar 3. ,4. ve 5. adımları tekrar et.
3. Örnekleri giriş verilerine göre en yakın oldukları merkezlere ata.
4. Merkezleri kendi kümelerine dâhil edilen örneklerin ortalamasına çek.
5. Merkezlere model olarak kendi kümelerindeki örneklerden geçen lineer bir fonksiyon ata.

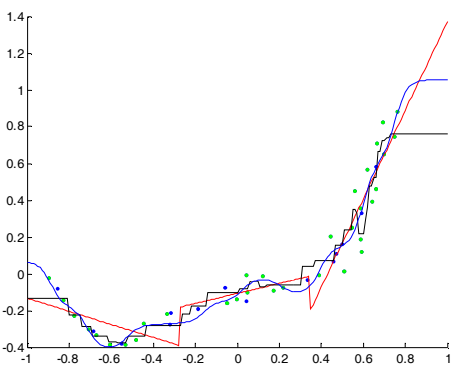
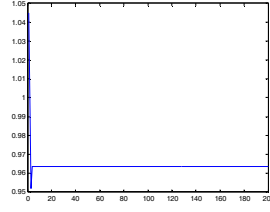
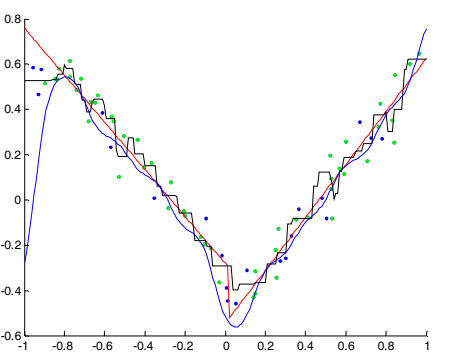
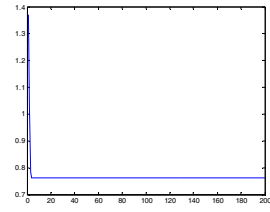
Algoritma.Kmeans Regresyon. Test:

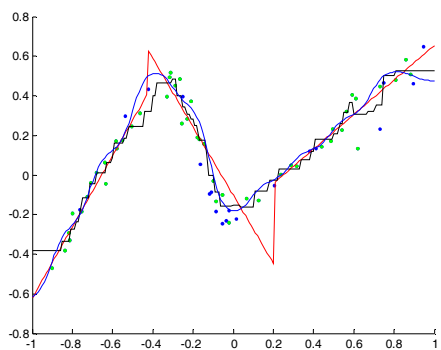
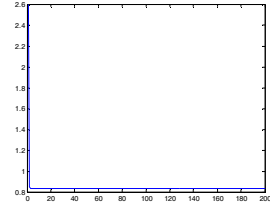
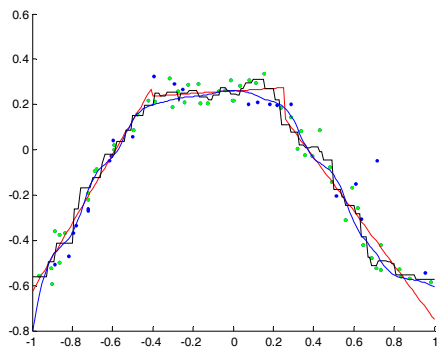
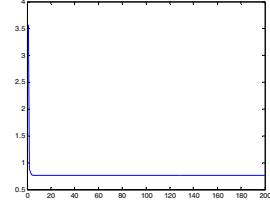
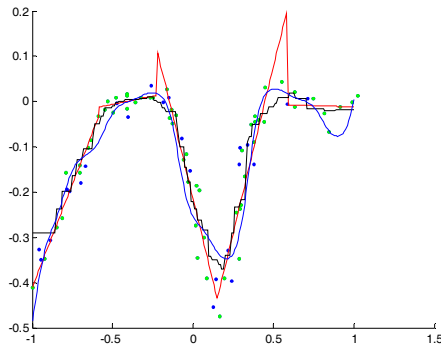
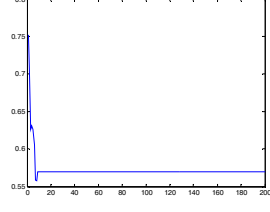
1. Verilen örneğe en yakın merkezi bul.
2. Merkezdeki lineer modele göre örneğin çıkışını hesapla.

Çizelge 8.2’de Kmeans ile regresyonun yapay bir veri kümesi üzerinde bulduğu sonuçlar 2 farklı metotla birlikte görülmektedir. Şekildeki mavi çizgi 10 saklı nöronlu bir yapay sinir ağının, siyah çizgi 3-en yakın komşulu fonksiyonun, kırmızı çizgi ise Kmeans ile regresyonun K=3 için ürettikleri fonksiyonlardır. Yeşil noktalar algoritmaların eğitiminde, mavi noktalar

test edilmesinde kullanılan örnekleri göstermektedir. Eğitim örneklerinin sayısının tüm örneklerin sayısına oranı 0.7'dir.

Çizelge 8.2 Kmeans, KNN ve ANN'lerin ürettikleri fonksiyonlar, Kmeans ile regresyonun ortalama karesel hatasının eğitim boyunca değişimi, metotların test verileri üzerindeki ortalama karesel hataları

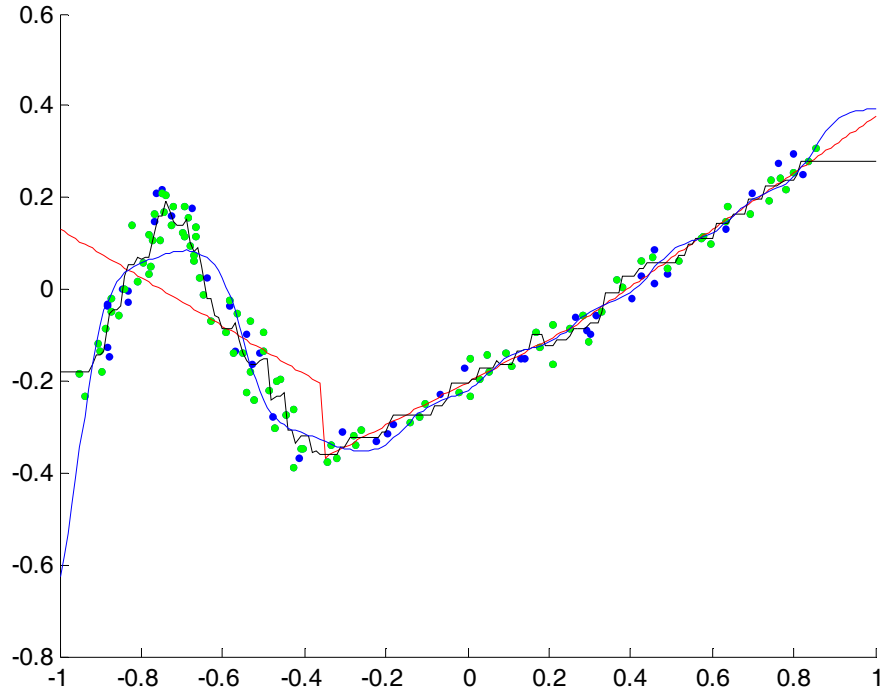
Üretilen fonksiyonlar	Kmean ile regresyonun MSE değişimi	Test verilerindeki MSE'ler
		<p>KMean3= 0.0710</p> <p>KNN3 = 0.0368</p> <p>ANN10= 0.0254</p>
		<p>KMean2= 0.1935</p> <p>KNNT2= 0.2007</p> <p>ANN10= 0.7637</p>

		<p>KMean3= 0.3490</p> <p>KNN3 = 0.1313</p> <p>ANN10 = 0.2285</p>
		<p>KMean3= 0.2487</p> <p>KNN3 = 0.3056</p> <p>ANN10= 0.2825</p>
		<p>KMean5= 0.0582</p> <p>KNN5= 0.0762</p> <p>ANN10= 0.1263</p>

Çizelge 8.2 incelendiğinde karşılaştırılan 3 metodun sonuçlarının birbirleriyle yarışabilir oldukları görülmüştür. Ancak metodların hepsinin varyanslarının çok yüksek olduğu gözlemlenmiştir. Özellikle Kmeans ile regresyon ve ANN'in rasgele başlangıç koşulları içermesinden dolayı böyle bir sonucun normal olduğu düşünülmüştür.

8.3.2 KmeansS ile Regresyon

Şekil 8.3'te Kmean ile regresyonu temeldeki problemi gösterilmiştir.



Şekil 8.7 Kmeans ile regresyon (K=3 için)

Normalde ilk çıkışa bir merkez, ardındaki inişe bir merkez ve son büyük çıkışa bir merkez konması yeterlidir. Ancak Şekil 8.7’de görüldüğü gibi Kmean ile regresyon bunu bulamamaktadır. Bunun yerine baştaki iniş ve çıkışa tek bir merkez koyup, gereksiz yere büyük çıkışa iki merkez koymaktadır. Bunun sebebi ise kümelemedeki eğiticişizliktir. Diğer bir ifadeyle sadece girişlere bakarak, çıkışları dikkate almadan işlem yapmaktır. Kümeleme algoritması hedef değerlerden tamamen habersizdir ve sadece her merkeze yakın sayıda örnek atamakta ve merkeze olan uzaklıkların minimize edilmesiyle ilgilenmektedir.

Bu problemi çözmek için, KmeansS ile regresyon adıyla anılan, her adımda MSE’si en yüksek olan merkeze en yakın merkezin yaklaştırılmasını içeren bir algoritma geliştirilmiş ancak tatmin edici sonuçlar elde edilememiştir. Algoritma 3’te KmeansS ile regresyonun eğitim aşamasının akışı verilmiştir. Test aşaması Kmeans ile regresyonla aynıdır. Algoritma örneklerin hedef değerlerini de içine aldığı için eğitim aşaması eğiticilidir.

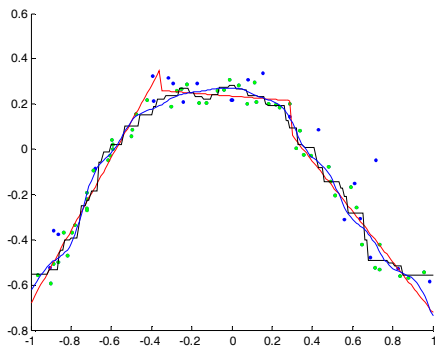
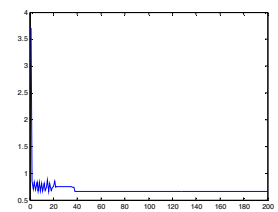
Algoritma. KmeansS Regresyon. Eğitim:

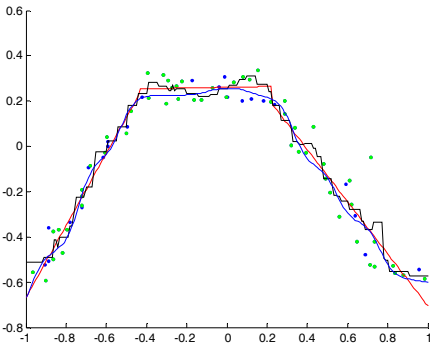
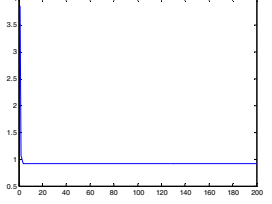
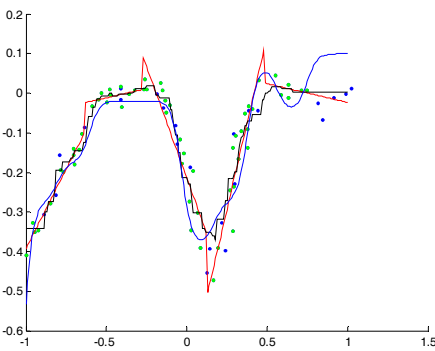
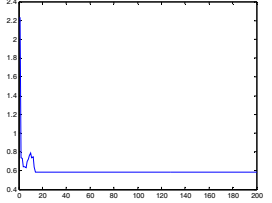
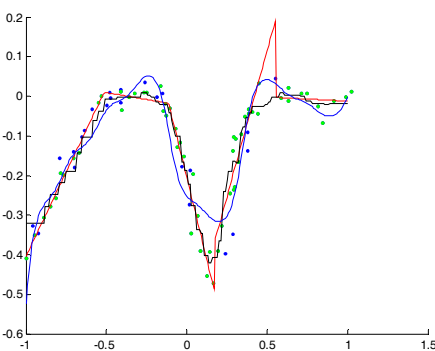
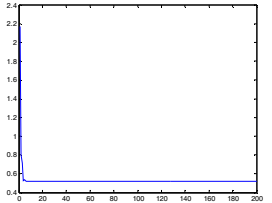
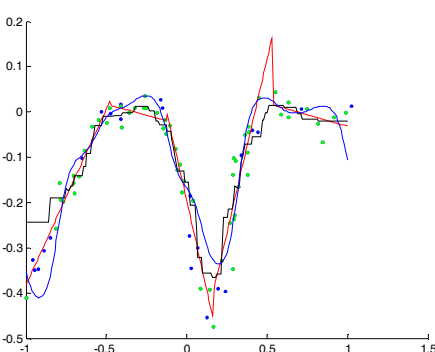
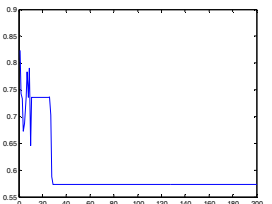
1. Merkezleri rasgele ata.

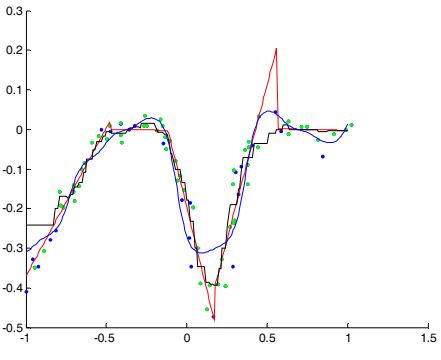
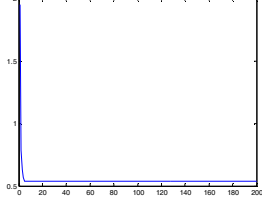
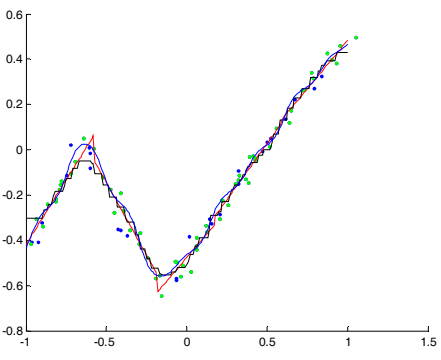
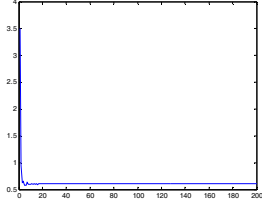
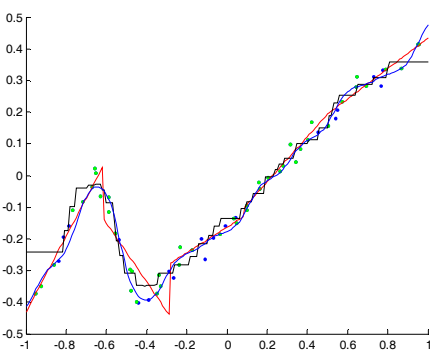
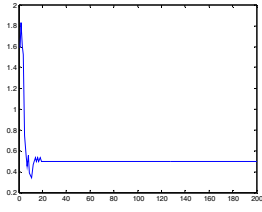
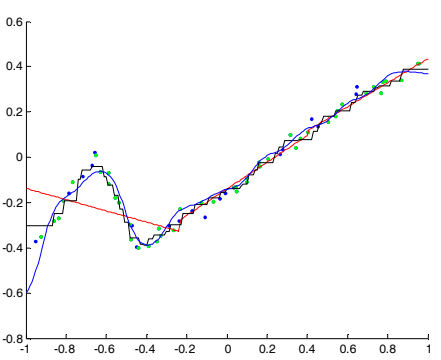
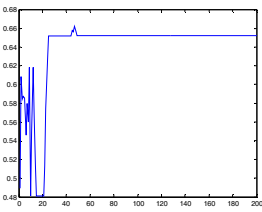
2. Karesel hata bir eşik değerin altına ininceye kadar ya da maksimum tekrar sayısına erişilene kadar aşağıdaki adımları (3-7) tekrar et.
3. Örnekleri en yakın oldukları merkezlere ata.
4. Merkezleri kendi kümelerine dâhil edilen örneklerin ortalamasına çek.
5. Merkezlere model olarak kendi kümelerindeki örneklerden geçen lineer bir fonksiyon ata ve MSE'lerini hesapla
6. MSE'si en yüksek olan merkeze, o merkeze en yakın merkezi *alfa* oranında yaklaştır.
7. $alfa = alfa * safla$

Alfa ve *safla*'nın değerlerini etkisi incelenmiş fakat sonuç performansına çok fazla etkisi gözlemlenmemiştir. Sadece hatanın değişiminin sürdüğü tekrar(epoch) sayısı değişmektedir. Çizelge 8.3'te KmeansS ile regresyon örnekleri verilmiştir.

Çizelge 8.3 KmeansS, KNN ve ANN'lerin ürettikleri fonksiyonlar, Kmeans ile regresyonun ortalama karesel hatasının eğitim boyunca değişimi, metotların test verileri üzerindeki ortalama karesel hataları (*alfa* ve *safla* değeri)

Üretilen fonksiyonlar	KmeanS ile regresyonun MSE değişimi	Test verilerindeki MSE'ler
		<p>KMean3S=0.2790</p> <p>(0.9 , 0.9)</p> <p>KNN3 = 0.3606</p> <p>ANN10= 0.3034</p>

 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=3. The x-axis ranges from -1 to 1, and the y-axis ranges from -0.8 to 0.6. The curves are smooth and follow the general trend of the data points.</p>	 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=3. The x-axis ranges from 0 to 200, and the y-axis ranges from 0.5 to 4. The curves are smooth and follow the general trend of the data points.</p>	<p>KMean3S=0.0813</p> <p>(0.1 , 0.1)</p> <p>KNN3 = 0.1038</p> <p>ANN10= 0.0720</p>
 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from -1 to 1.5, and the y-axis ranges from -0.6 to 0.2. The curves are smooth and follow the general trend of the data points.</p>	 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from 0 to 200, and the y-axis ranges from 0.6 to 2.4. The curves are smooth and follow the general trend of the data points.</p>	<p>KMean5S=0.0591</p> <p>(0.1 , 0.9)</p> <p>KNN5 = 0.0616</p> <p>ANN10= 0.1544</p>
 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from -1 to 1.5, and the y-axis ranges from -0.6 to 0.2. The curves are smooth and follow the general trend of the data points.</p>	 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from 0 to 200, and the y-axis ranges from 0.4 to 2.4. The curves are smooth and follow the general trend of the data points.</p>	<p>KMean5S=0.0844</p> <p>(0.1, 0.1)</p> <p>KNN5= 0.0788</p> <p>ANN5 = 0.0498</p>
 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from -1 to 1.5, and the y-axis ranges from -0.5 to 0.2. The curves are smooth and follow the general trend of the data points.</p>	 <p>A plot showing a function with data points (green dots) and fitted curves (red, blue, and black lines) for K=5. The x-axis ranges from 0 to 200, and the y-axis ranges from 0.5 to 1.0. The curves are smooth and follow the general trend of the data points.</p>	<p>KMean5S=0.0488</p> <p>(0.9, 0.9)</p> <p>KNN5= 0.1144</p> <p>ANN5 = 0.1393</p>

		<p>KMean5S=0.0762</p> <p>(0.9, 0.1)</p> <p>KNN5= 0.1208</p> <p>ANN5 = 0.1021</p>
		<p>KMean5S=0.0871</p> <p>(0.9, 0.9)</p> <p>KNN5 = 0.0947</p> <p>ANN10=0.0964</p>
		<p>KMean5S=0.0435</p> <p>(0.9, 0.9)</p> <p>KNN5= 0.0205</p> <p>ANN10= 0.0150</p>
		<p>KMean3S=0.1895</p> <p>(0.9, 0.9)</p> <p>KNN3= 0.0319</p> <p>ANN10= 0.0454</p>

Çizelge 8.3 incelendiğinde KmeansS'in, Kmeans ile regresyondaki eğiticiisizlik problemine tam bir çözüm getirmediği görülmektedir. Özellikle son örnekte bu durum göze çarpmaktadır. Böylece sadece en hatalı modele sahip merkezin değerinin güncellenmesinin yeterli olmadığı anlaşılmıştır.

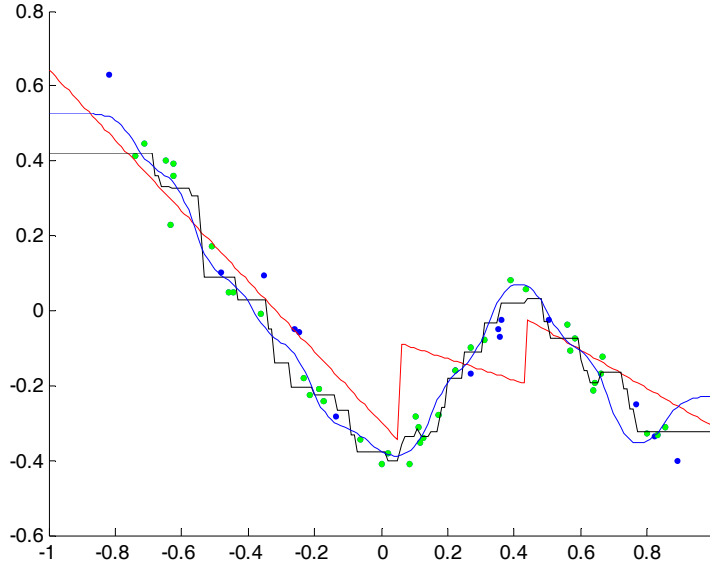
8.3.3 KmeansMOD ile Regresyon

Kmeans ve KmeansS algoritmalarında test aşamasında örnekler en yakın oldukları merkezlere atanmaktadır. KmeansMOD algoritmasında ise örnekler dik uzaklıklarının en az olduğu modele atanmaktadır. Başlangıçta rasgele atanan modellerin eğitimle veri kümesine uygun hale geleceği düşünülmüştür. KmeansMOD'un akışı Algoritma 4'te verilmiştir

Algoritma 4:

1. Modelleri rasgele ata.
2. Karesel hata bir eşik değerinin altına ininceye kadar ya da maksimum tekrar sayısına erişilene kadar 3. ve 4. adımları tekrar et.
3. Örnekleri dik uzaklıkları en az olan modelin kümesine ata.
4. Modelleri, kendi kümeleri içindeki örneklere en uygun lineer model olarak güncelle.

KmeansMOD algoritması merkezleri sadece test aşamasında kullanmaktadır. Bir örneği en yakın merkezin modeline atayarak model çıkışı bulmaktadır. Ancak bu yapının sebep olduğu problem Şekil 8.8'de görülmektedir.

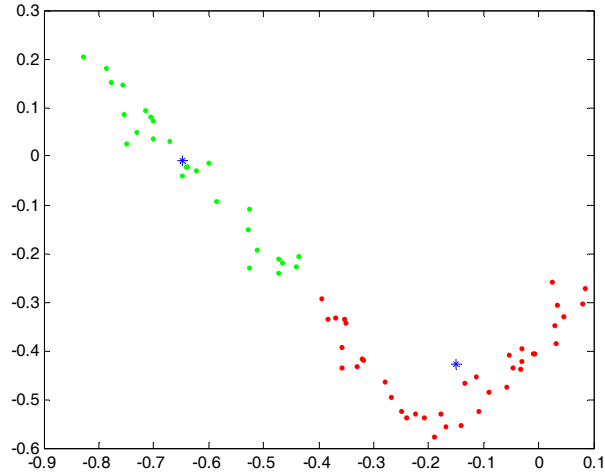


Şekil 8.8 KmeansMOD ile regresyon (K=3 için)

KmeansMOD'daki problemin sebebi, örneklerin modellere atama kısmıdır. Modellerin bir merkezleri olmadığı için bir yerel bölge için parametre oluşturamamakta ve birbirinden uzaktaki noktalar aynı modele atanabilmektedirler. Örneğin Şekil 8.5'te ortada bulunan modelin parametreleri belirlenirken bu durum ortaya çıkmıştır.

8.3.4 KmeansXY ile Regresyon

Kmeans ile regresyondaki eğiticişizliğe bir başka çözümde KmeansXY denemesidir. Bu yapıda veriler Kmeans'de olduğu gibi merkezlere göre kümelenebilir. Ancak verilere bir özellik olarak hedef değerleri de eklenmektedir. Diğer bir ifadeyle verilerin boyutu bir artırılarak yeni bir uzayda kümeleme yapılmaktadır. Test aşamasında ise örneğin hangi merkeze yakın olduğu bulunurken merkezlerin hedef özellikleri haricindeki boyutlarda uzaklık ölçümü yapılmaktadır.



Şekil 8.9 KmeansXY ile örnekleri kümeleme (K=2 için)

Şekil 8.9 incelendiğinde KmeanXY'nin de probleme çözüm olamadığı görülmektedir. Sadece giriş özelliklerine bakılarak yapılan kümelemeden daha iyi bir sonuç elde edilmiştir ancak yine de istenen sonuç değildir.

8.4 Deneyisel Sonuçlar

Geliştirilen algoritmalar esas olarak tek boyutlu veriler için tasarlanmıştır. Ancak gerçek dünya verilerinde boyut sayısı daha fazladır. Bu nedenle geliştirilen algoritmalarda önce her bir boyut için çıkışlar bulunmuş daha sonra ortalamaları alınarak gerçek çıkış bulunmuştur.

Çizelge 8.4 Regresyonda kullanılan veri kümeleri [15]

Veri kümesi	Boyut sayısı	Örnek sayısı
dela	6	7129
dele	7	9512
hous	14	506
kine	9	8192
mach	7	209
stoc	10	950
tria	61	186
wisc	33	194

Çizelge 8.5'te, Çizelge 8.4'te yer alan veri kümeleri üzerinde çeşitli regresyon metotlarıyla 10'lu çapraz geçerleme yapılarak elde edilmiş ortalama karesel hatalar (MSE) verilmiştir.

Geliştirilen algoritmalarından sadece Kmeans ile regresyon en iyi sonuçları verilmiştir. Bu sebeple sadece bu algoritmanın sonuçları verilmiştir.

Çizelgedeki algoritma isimlerindeki KMreg ifadesi Kmeans ile regresyonu, KNNreg ifadesi KNN ile regresyonu, K=X ifadeleri K değerinin X olduğunu belirtmektedir. Sarı kısımlar en iyi, yeşiller ise en iyi ikinci sonuçları göstermektedir.

Çizelge 8.5 Regresyon sonuçları

Veri kümesi	dela	dele	hous	kine	Mach	stoc	tria	wisc
KmregK=6	2.80975E-08	2.0937E-06	15.70591	0.034173263	3267.8441	0.97289557	0.041796712	239554.94
KmregK=5	2.85886E-08	2.09641E-06	22.195441	0.034112788	2165.8216	1.0736404	0.052990556	185378.38
KmregK=4	2.87865E-08	2.09466E-06	15.025692	0.035063155	2443.1484	1.3176651	0.052144635	3227.2873
KmregK=3	2.94481E-08	2.09165E-06	14.890469	0.036746358	6554.6184	1.6576803	0.038095091	4854.8612
KmregK=2	2.96207E-08	2.09933E-06	14.944982	0.038265193	2875.1032	2.031674	0.034350601	1555.5286
KmregK=1	2.96619E-08	2.09967E-06	24.621887	0.040840563	5837.9955	5.5020201	0.023974772	1106.7495
KNNregK=6	3.74256E-08	2.6645E-06	40.150662	0.013775948	3125.9503	0.54574836	0.021212522	1206.0418
KNNregK=5	3.82845E-08	2.70896E-06	41.553513	0.014174928	5381.1499	0.511075	0.022504716	1224.2698
KNNregK=4	3.96111E-08	2.79697E-06	38.478578	0.014799449	4140.7121	0.46739104	0.021173693	1310.1997
KNNregK=3	4.28769E-08	2.95249E-06	40.12841	0.01612356	4143.7017	0.43751096	0.020174057	1399.1165
KNNregK=2	4.63232E-08	3.26563E-06	42.696759	0.018846903	4270.3208	0.41081003	0.020109115	1608.5325
KNNregK=1	9.17237E-08	5.63869E-06	85.107358	0.06944866	24885.115	42.223311	0.024801686	1268.8018
m5p	2.70603E-08	2.03082E-06	13.23486225	0.025679694	2952.072042	0.875605436	0.016987577	1141.098514
Pacereg	2.95565E-08	2.09737E-06	22.99645004	0.040801386	4054.101457	5.476453572	0.022194877	1128.131896
Lineerreg	2.95668E-08	2.09772E-06	23.02706265	0.040807883	3942.480303	5.468463969	0.025619303	1111.65417
m5rules	2.74631E-08	2.04041E-06	14.82917319	0.030351243	4305.769926	0.948216795	0.017594057	1187.22476
Svmreg	3.02169E-08	2.10616E-06	24.44568242	0.04184553	4148.249079	5.745854599	0.021896305	1104.711762
m5' (Malerba vd., 2004)	5.3824E-08	2.26576E-05	12.8164	0.02499561	3059.729312	3.32150625	0.04068289	2643.374717
Smoti (Malerba vd., 2004)	4.00001E-08	2.6569E-06	18.31215173	0.037922499	3289.33916	1.230590862	0.024034301	2061.744793

Görüldüğü gibi sonuçlar çok tatmin edici olmamakla birlikte, umut vaat eden denemelerde içermektedir. Ayrıca geliştirilen metotların karşılaştırıldığı algoritmalar literatürdeki en başarılı sonuçlar üreten algoritmalarlardır.

8.5 Çıkarımlar

Bu bölümde veri kümesinin her bir özelliği için kümeleme tabanlı bir regresyon modeli oluşturan ve bunların sonuçlarını birleştiren çeşitli regresyon algoritmaları geliştirilmiştir. Algoritmalar 8 veri kümesiyle karşılaştırılmış. Her bir özellik için elde edilen sonuçların birleştirilmesinde, ortalama alma gibi basit bir mekanizma kullanılmasına rağmen umut vaat eden sonuçlar elde edilmiştir. Bununla birlikte daha başarılı sonuçlar için geliştirilmeye ihtiyacı vardır.

8.6 Gelecek Çalışmalar

Araştırmaya açık alanlar ve denenebilecek fikirler aşağıda listelenmiştir:

- Her boyut için üretilen sonuçların ortalamasının alınması yerine bazı özelliklerin kullanılması ya da çıkış değerlerinin ağırlıklandırılması yapılabilir.
- Geliştirilen algoritma çeşitli meta algoritmalarla birleştirilebilir.
- K değeri otomatik seçilebilir.

9. META REGRESYON

Makine öğrenmesi alanında tüm veri kümelerinde diğer bütün algoritmalarda daha iyi çalışan tek bir algoritma yoktur. Algoritma (model) seçimi deneme yanılma yoluyla yapılır. Bu deneme yanılmadan kurtulmak isteyen, diğer bir deyişle bu işi otomatikleştirmek isteyen araştırmacılar “Meta Learning” adında bir yaklaşım geliştirmişlerdir. Bu yaklaşıma göre bir veri kümesinin çeşitli özelliklerine bakılarak hangi algoritmalarla ya da hangi tür algoritmalarla daha iyi modellenebileceği bulunabilir. Bu yaklaşım her bir örneğin bir veri kümesi olduğu, her bir örneğin özelliklerinin ise veri kümesine ait çeşitli özellikler (meta özellik) olduğu, hedefin (çıkışın) istenilen bir algoritmanın bu veri kümesindeki başarısı olduğu bir problem uzayı tasarlamaktadır. Böyle bir problem tasarlanırken meta veri kümesi olarak adlandırılabilen, satırlarında örnekler (veri kümeleri), sütunlarında veri kümelerinin özellikleri (meta özellik) olan bir matris oluşturulmaktadır. Buradaki amaç, veri kümelerinin (örneklerin) çeşitli özelliklerinden (istatistiki, basit algoritmaların performansları vs.) karmaşık ve muhtemelen daha yüksek başarımlı bir algoritmanın performansının tahmin edilmesidir. Basit algoritma olarak ifade edilen algoritmalar, hızlı ve performansları muhtemelen düşük algoritmalarlardır.

Meta veri kümesinin hedef kısmı (çıkış) değiştirilerek “bu veri kümesi (örnek) için en başarılı performansı gösterecek algoritma” yapılırsa bu veri kümesi, bu amaçla da (bu veri kümesi için hangi algoritmayı (modeli) kullanmalıyım) kullanılabilir hale getirilebilir.

Yine aynı şekilde hedef kısmı “bu veri kümesi (örnek) için en başarılı performansı gösterecek komite metodu” şeklinde değiştirilirse, bütün komite metotlarını denemeye gerek kalmadan en iyi performansı göstereceğini umduğumuz metot kullanılabilir.

Örnekleri çoğaltmak mümkündür. Diğer bir ifadeyle bu tarz meta veri kümeleriyle bir “makine öğrenmesi uzman sistemi” geliştirilebilir.

9.1 Tanımlar ve Semboller

Meta veri kümesi: Bir meta öğrenme probleminde kullanılmak üzere oluşturulan, veri kümelerinin çeşitli özelliklerini içeren veri kümesidir.

Meta özellik: Meta veri kümelerinin özelliklerine meta özellik denir. Veri kümesinin çeşitli özelliklerini ifade etmekte kullanılırlar. Literatürde çeşitli türleri mevcuttur. Ayrıntılar için tezin “Meta regresyon denemelerinde ortak kullanılan meta özellikler” adlı bölümü incelenebilir.

N. dereceden moment: Bir değişkenin ortalamasından sapmalarının ölçümünde kullanılan bir ölçüttür. N adet örneğin n. dereceden momentini Eşitlik 9.1 ve 9.2’deki gibi bulunur.

$$x^* = \frac{\sum_{i=1}^N x_i}{N} \quad (9.1)$$

$$mom_n = \sum_{i=1}^N (x_i - x^*)^n \quad (9.2)$$

Histogram: Bir değişkenin dağılımının, önceden belirlenmiş kesişmeyen aralıklara ya da kategorilere (*bin*) düşen örnek sayısı ile grafiksel gösterimidir.

Savrukluuk (Kurtosis): Bir değişkenin dağılımının normal dağılıma benzemezliğinin ölçümünde kullanılır. Eşitlik 9.3’teki gibi bulunur (Abramowitz vd., 1972).

$$kurtosis = \frac{N \sum_{i=1}^N (x_i - x^*)^4}{\left(\sum_{i=1}^N (x_i - x^*)^2 \right)^2} \quad (9.3)$$

Yamukluk (Skewness): Bir değişkenin dağılımının asimetrikliğinin ölçümünde kullanılır. Eşitlik 9.4’teki gibi bulunur. Kayıklık olarak ta adlandırılır.

$$skewness = \frac{\sqrt{N} \sum_{i=1}^N (x_i - x^*)^3}{\left(\sum_{i=1}^N (x_i - x^*)^2 \right)^{3/2}} \quad (9.4)$$

9.2 Farklı Sayıdaki Özellik Sayısına Sahip Veri Kümelerinin (Örneklerin) Bir Arada Kullanılması

Meta özellik olarak, özelliklerinin standart sapmalarının kullanılması durumunda her bir veri kümesinin özellik sayısı birbirinden farklı olabileceği için meta uzaydaki her bir örneğin (veri kümesi) farklı sayıda özelliği (boyutu) olacaktır ve sütun sayıları eşit olan düzgün bir yapının aksine Çizelge 9.1’deki gibi bir yapı ortaya çıkacaktır

Çizelge 9.1 Farklı sayıda meta özellik içeren veri kümeleri

	Özellik sayısı	Örnek sayısı	1.özelliğin standart sapması	2.özelliğin standart sapması	3.özelliğin standart sapması	D özelliğin standart sapması	1. basit algoritmanın performansı	2. basit algoritmanın performansı	3. basit algoritmanın performansı	Karmaşık algoritmanın performansı
Veri Kümesi1	2	10	0.75	0.4	?	?				
Veri Kümesi2	3	20	0.1	0.2	0.3	?				
Veri Kümesi3	6	22	0.1	0.2	0.3	0.5				
...										
Veri KümesiN	5	33	0.05	0.9	0.3	0.2				

Bu yapıda iki tür problem mevcuttur. Birincisi, aynı türden olmayabilecek özelliklerin aynı sütunda yer alması (1.veri kümesinin 1.özelliğinin standart sapması ile 2. veri kümesinin 1. özelliğinin standart sapmasını karşılaştırmanın bir anlamı yoktur), ikincisi ise farklı boyut içeren örneklerin birbirleriyle nasıl karşılaştırılabileceği konusudur.

Bu problemlere çözüm olarak standart sapmalar diye üst bir meta özellik tanımlanarak, standart sapmaların ortalamalarının kullanılması olabilir. Ancak bu durumda önemli miktarda veri kaybolmaktadır. Bunun yerine histogram kullanılması veya eklemeli kümeleyici algoritmalarında kullanılan single linkage, avarage linkage tabanlı yaklaşımlar benimsenmiştir.

9.2.1 Histogram Yaklaşımı

Farklı boyutlara sahip özelliklerin kendileri yerine eşit binlere bölünmüş histogram değerleri birebir karşılaştırılabilir. Sırasıyla 10 ve 20 özelliğe sahip iki veri kümesinin (örneğin) standart sapma meta özelliklerinin arasındaki mesafeyi ölçmek için her iki veri kümesinin özelliklerinin 5 bin’li histogramları çıkarılır ve dolayısıyla her iki örnekte eşit sayıda (5’er) özellik ile ifade edilmiş olur. Ya da bu 5’er binli histogram şekillerine birer etiket verilerek

(normal, binominal, poison vs.) yine her iki örnekte eşit sayıda (1'er) özellikle ifade edilmiş olur. Bu tezde benimsenen yaklaşımdır.

9.2.2 Kümeleme Yaklaşımı

Eklmeli kümeleme algoritmalarında (Bkz. Bölüm 5.2) her bir adımda birbirine en çok benzeyen iki küme birleştirilir. Bu kümeler her zaman eşit sayıda örnek içermezler. Bu gibi durumlar için “*single linkage*”, “*average linkage*” tabanlı yaklaşımlar geliştirilmiştir. İki kümenin birbirine olan benzerliği birbirine en benzer örneklerinin benzerliği kadardır ya da en benzemeyen örneklerinin benzerliği kadardır. Bu iki yaklaşımda ekstrem örneklere karşı oldukça duyarlı olduklarından bunların yerine işlem maliyeti daha yüksek ama başarıımı daha yüksek olan ortalama tabanlı bir yaklaşım geliştirilmiştir. Bu yaklaşımda farklı kümelerdeki mümkün tüm örnek ikilileri arasındaki uzaklıklar ölçülür ve ortalaması bu iki kümenin ortalaması olarak kabul edilir. İşte kümeleme için geliştirilmiş bu yaklaşımlar farklı özellik sayısına sahip örneklerin birbirlerine olan uzaklıklarının bulunmasında da kullanılmaktadır.

9.3 Önceki Çalışmalar

Literatürde meta öğrenim konusundaki araştırmaların tamamına yakın kısmı sınıflandırma problemlerine yönelmişlerdir.

“An Evaluation of Landmarking Variants” adlı çalışmada (Fürnkranz vd., 2001), sınıflandırıcıların performanslarının tahmininde basit sınıflandırıcıların performanslarını ve karar ağaçlarından çıkan özellikleri meta özellik olarak kullanılmıştır. Performansların kendilerinin yerine başarı sıralarının, birbirlerine oranlarının, ikili karşılaştırma sonuçlarının kullanımı önerilmiştir.

“A Higher-order Approach to Meta-Learning” adlı çalışmada (Bensusan vd., 2000b), performans tahmini için karar ağaçlarından elde edilen özellikler kullanılmıştır.

“Model Selection via Meta-learning: a Comparative Study” adlı çalışmada (Kalousis vd., 2000), meta özellik olarak istatistik türünden 57 adet özellik kullanılmıştır. Meta veri kümesinin etiketleri olarak 2 sınıflandırıcıdan hangisinin daha başarılı olacağını kullanılmıştır.

“Representational Issues in Meta-Learning” adlı çalışmada (Kalousis vd., 2003) farklı meta özellik sayısına sahip veri kümelerinin birbirlerine uzaklıklarının ölçümü için yeni bir yol (simK) önerilmiştir. Çalışmada meta özellik sayısının birbirinden çok farklı olması

durumlarında veri kümelerinin arasındaki mesafe arttırılmaktadır. Önerilen simK, single linkage ve avarage linkage yaklaşımları çalışmada karşılaştırılmış her üç metotta ortalama almaktan daha iyi sonuçlar üretmiştir. Ayrıca single linkage ve avarage linkage, simK'dan daha iyi sonuçlar üretmiştir. Ancak aradaki fark istatistiki olarak önemli değildir.

“The Data Mining Advisor: Meta-learning at the Service of Practitioners” adlı çalışmada (Carrier, 2005), meta özellik olarak veri kümesinin istatistiki ölçümlerini, basit sınıflandırıcıların performansları ve karar ağaçlarından çıkan özellikler meta özellik olarak kullanılmıştır.

“Ranking with Predictive Clustering Trees” adlı çalışmada (Todorovski vd., 2002a), 65 adet UCI sınıflandırma problemi üzerinde çalışılmıştır. Meta özellik olarak veri kümesinin istatistiki ölçümleri ve basit sınıflandırıcıların performansları kullanılmıştır.

“Experiments in Meta-Level Learning with ILP” adlı çalışmada (Todorovski vd., 1999), 20 adet UCI sınıflandırma veri kümesi üzerinde çalışılmıştır. Meta özellik olarak istatistiki ölçümler kullanılmıştır. Meta özelliklere göre 3 algoritmadan hangisinin kullanılmasının daha başarılı sonuçlar vereceğini gösteren kurallar üretilmiştir.

“Characterization of Classification Algorithms” adlı çalışmada (Gama vd., 1995), 22 algoritmanın 20 sınıflandırma veri kümesindeki performansları ölçülmüştür. Bu performansların tahmininde istatistiki özellikler kullanılmıştır. Performanslar regresyon, kural, model ağaçları ve örnek tabanlı modellerle tahmin edilmiştir. Modellerden hiçbirinin diğerlerinden istatistiki anlamda farklı olmadığı görülmüştür.

“Experiments with Automatic Feature Selection in Meta-Learning” adlı çalışmada (Todorovski vd., 2002b), 65 UCI sınıflandırma veri kümesinin 56 istatistiki meta özelliği kullanılmıştır. Tüm meta özellikleri kullanmak yerine wrapper tarzı bir özellik seçici ile özellik sayısı azaltılmış ve tüm özelliklerin kullanımına göre daha başarılı performans tahminleri yapılmıştır.

“Decision Tree-Based Data Characterization for Meta-Learning” adlı çalışmada (Peng vd., 2002), istatistiki, basit sınıflandırıcı performansları ve karar ağaçlarından elde edilen özellikler meta özellik olarak kullanılmıştır. 10 algoritmanın 47 UCI veri kümesi üzerindeki performansları tahmin edilmiştir. Veri kümelerini karar ağaçlarından elde edilen özelliklerle ifade etmenin diğer meta özellik gruplarına göre daha başarılı olduğu gösterilmiştir. Ayrıca meta özelliklerde özellik seçiminin tahmin performansını değiştirmede de söylenmiştir.

“Ranking Learning Algorithms” adlı çalışmada (Brazdil vd., 2003), istatistiki meta özellikler kullanarak K En Yakın Komşu algoritması ile yeni bir veri kümesine en benzer veri kümelerindeki algoritmaların performansları kullanarak, yeni veri kümesi için algoritmaların performanslarını tahmin eden bir sistem geliştirilmiştir. 10 algoritmanın 53 veri kümesi üzerindeki performansları tahmin edilmiştir.

“Estimating the Predictive Accuracy of a Classifier” adlı çalışmada (Bensusan vd., 2001), meta özellik olarak istatistiki ölçümlerle basit sınıflandırıcıların performansları karşılaştırılmış ve sınıflandırıcı performanslarının, performans tahmininde daha başarılı olduğu gösterilmiştir. Tahmin denemeleri 65 UCI veri kümesi üzerinde yapılmıştır.

“On Learning algorithms selection for classification” adlı çalışmada (Ali vd., 2006), 100 UCI veri kümesi üzerinde istatistiki meta özellikler kullanarak 8 algoritmadan hangisinin hangi durumda daha başarılı olacağının kuralları çıkarılmıştır.

9.4 Gerçekleştirilenler ve Deneysel Sonuçlar

Bu bölümde öncelikle bu çalışmada kullanılan meta özellikler anlatılmıştır. Daha sonra 3 bölüm halinde, 3 veri kümesi koleksiyonu üzerinde yapılan meta regresyon çalışmaları verilmiştir. Her bir bölümde;

- Koleksiyonu oluşturan veri kümeleri
- Ek olarak kullanılan meta özellikler
- Algoritmaların veri kümesindeki performanslarının incelemeleri
- Meta özelliklerin birbirleriyle olan korelasyon analizleri
- Algoritmaların ve veri kümelerinin performanslarına göre hiyerarşik kümelenmeleri
- En başarılı algoritmaların performanslarının meta özelliklerle tahminleri
- Performans tahminlerle kullanılan meta özelliklerin analizleri

yer almaktadır.

9.4.1 Meta Regresyon Denemelerinde Ortak Kullanılan Meta Özellikler

Literatürde birçok meta özellik önerisi bulunmaktadır. Bu çalışmada ise hem mevcut meta özellikler hem de yeni önerilen meta özellikler kullanılmıştır. Bu meta özellikleri 6 gruba

ayırmak mümkündür. Çizelge 9.2’de bu 6 grup verilmiştir.

Çizelge 9.2 Meta özellik grupları

Meta Özellik Grubu	İçerdiği meta özellik sayısı	Açıklaması
STA	15	Veri kümesinin ilk başta göze çarpan istatistiki özellikleridir. (örnek sayısı, özellik sayısı vs.)
ST2	220	Veri kümesinin Kurtosis, skewnes, 3. ve 4. dereceden moment, korelasyon matrisleri gibi istatistiki özellikleridir.
CLUS	5	Yapılan kümeleme işlemleri sonucunda elde edilen özelliklerdir (örneklerin kümelere dağılımları, küme sayıları).
REGT	18	Üretilen Karar ağaçlarının çeşitli özellikleridir (yaprak sayısı, düğüm sayısı vs.).
RMSE	15	Çeşitli algoritmaların yaptıkları hataların RMSE değerleridir.
PCA	22	Temel bileşen analiziyle bulunan özelliklerdir.
Toplam	295	

Bu grupların her birinin içerdikleri meta özelliklerin listeleri ve açıklamaları Çizelge 9.3, 9.4, 9.5, 9.6, 9.7 ve 9.8’de verilmiştir.

Çizelge 9.3 STA meta özellik grubu

Meta Özellik İsmi	Açıklaması
STA.binsayi	Veri kümesindeki sadece 2 değer alan özellik sayısı
STA.cfs_oran	Veri kümesinde CFS ile seçilmiş özellik sayısının toplam özellik sayısına oranı
STA.cfs_sayi	Veri kümesinde CFS ile seçilmiş özellik sayısı
STA.iqr_e_oran	Veri kümesinde çıkış değeri ekstrem olan örnek sayısının toplam örnek sayısına oranı
STA.iqr_extrem	Veri kümesinde çıkış değeri ekstrem olan örnek sayısı
STA.iqr_o_oran	Veri kümesinde çıkış değeri aykırı olan örnek sayısının toplam örnek sayısına oranı
STA.iqr_outlier	Veri kümesinde çıkış değeri aykırı olan örnek sayısı
STA.orneksayi	Veri kümesindeki örnek sayısı
STA.ozelliksayi	Veri kümesinde özellik sayısı
STA.perbinornek	Veri kümesinde sadece 2 değer alan özellik sayısının örnek sayısına oranı
STA.perbinsayi	Veri kümesinde sadece 2 değer alan özellik sayısının toplam özellik sayısına oranı
STA.pertriornek	Veri kümesinde sadece 3 değer alan özellik sayısının örnek sayısına oranı
STA.pertrisayi	Veri kümesinde sadece 2 değer alan özellik sayısının toplam özellik sayısına oranı
STA.pertumsayi	Veri kümesindeki özellik sayısının örnek sayısına oranı
STA.trisayi	Veri kümesindeki sadece 3 değer alan özellik sayısı

Çizelge 9.4 ST2 meta özellik grubu

Meta Özellik İsmi	Açıklaması
ST2.bigcorXpro	X'nin korelasyon matrisinin (diagonal hariç) yüzde kaçının 0.5'den büyük olduğu (Colinearity değeri ile doğru orantılı bir özellik)
ST2.bigcorXYpro	X'nin Y ile korelasyon matrisinin yüzde kaçının 0.5'den büyük olduğu
ST2.corXdeg1..10	X'nin korelasyon matrisinin 10 bin'lik histogram değerleri (küçükten büyüğe

	sıralanmış)
ST2.corXfre1..10	X'nin korelasyon matrisinin 10 bin'lik histogram frekanslarının normalize değerleri
ST2.corXYBolustdXY1..10	X'nin Y ile korelasyon matrisinin $\sqrt{ST2.stdX*ST2.stdY}$ 'e bölümünün histogram değerleri
ST2.corXYdeg1..10	X'nin Y ile korelasyon matrisinin 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.corXYfre1..10	X'nin Y ile korelasyon matrisinin 10 bin'lik frekanslarının normalize değerleri
ST2.freY1..10	Y'nin 10 bin'lik histogramının frekanslarının normalize değerleri. İlk değeri en az rastlanan değerin olasılığını, son değeri en çok rastlanan değerin olasılığını ifade eder.
ST2.kurtcorXfre	X'nin korelasyon matrisinin histogramının frekanslarının kurtosis değeri
ST2.kurtcorXYfre	X'nin Y ile korelasyon matrisinin histogramının frekanslarının kurtosis değeri
ST2.kurtdegX1...10	X'nin kurtosis'lerinin 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.kurtfreX1..10	X'nin kurtosis'lerinin 10 bin'lik histogram frekanslarının normalize değerleri
ST2.kurtfreY	Y'nin histogram frekanslarının kurtosis değeri
ST2.kurtkurtfreX	X'nin kurtosis histogramının frekanslarının şekli için kurtosis değeri
ST2.kurtmom3freX	X'nin 3.dereceden moment histogramının frekanslarının şekli için kurtosis değeri
ST2.kurtmom4freX	X'nin 4.dereceden moment histogramının frekanslarının şekli için kurtosis değeri
ST2.kurtskewfreX	X'nin skewness histogramının frekanslarının şekli için kurtosis değeri
ST2.kurtstdfreX	X'nin standart sapmalarının histogramının şekli için kurtosis değeri
ST2.kurtY	Y'nin kurtosis değeri
ST2.maxcorrXY	X'nin Y ile korelasyonu en yüksek %10'luk dilimin korelasyon değeri
ST2.meancorXdeg	X'nin korelasyon matrisinin histogramının değerlerinin ortalaması
ST2.meancorXfre	X'nin korelasyon matrisinin histogramının frekanslarının ortalaması

ST2.meancorXYdeg	X'nin Y ile korelasyon matrisinin histogramının değerlerinin ortalaması
ST2.meancorXYfre	X'nin Y ile korelasyon matrisinin histogramının frekanslarının ortalaması
ST2.meankurtdegX	X'nin kurtosis histogramının değerlerinin ortalaması
ST2.meankurtfreX	X'nin kurtosis histogramının frekanslarının ortalaması
ST2.meanmom3degX	X'nin 3.dereceden moment histogramının değerlerinin ortalaması
ST2.meanmom3freX	X'nin 3.dereceden moment histogramının frekanslarının ortalaması
ST2.meanmom4degX	X'nin 4.dereceden moment histogramının değerlerinin ortalaması
ST2.meanmom4freX	X'nin 4.dereceden moment histogramının frekanslarının ortalaması
ST2.meanskewdegX	X'nin skewness histogramının değerlerinin ortalaması
ST2.meanskewfreX	X'nin skewness histogramının frekanslarının ortalaması
ST2.mom3degX1..10	X'nin 3.dereceden momentlerinin 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.mom3freX1..10	ST2.mom3degX1..10'in histogram'ının frekanslarının normalize değerleri
ST2.mom3Y	Y'nin 3.dereceden moment değeri
ST2.mom4degX1..10	X'nin 4.dereceden momentlerinin 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.mom4freX1..10	ST2.mom4degX1..10'in histogram'ının frekanslarının normalize değerleri
ST2.mom4Y	Y'nin 4.dereceden moment değeri
ST2.skewcorXfre	X'nin korelasyon matrisinin histogramının frekanslarının skewness değeri
ST2.skewcorXYfre	X'nin Y ile korelasyon matrisinin histogramının frekanslarının skewness değeri
ST2.skewdegX1..10	X'nin skewness'lerinin 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.skewfreX1..10	ST2.skewdegX1..10'in histogram'ının frekanslarının normalize değerleri
ST2.skewfreY	Y'nin histogram frekanslarının skewness değeri
ST2.skewkurtfreX	X'nin kurtosis histogramının frekanslarının şekli için skewness değeri
ST2.skewmom3freX	X'nin 3.dereceden moment histogramının frekanslarının şekli için skewness değeri

ST2.skewmom4freX	X'nin 4.dereceden moment histogramının frekanslarının şekli için skewness değeri
ST2.skewskewfreX	X'nin skewness histogramının frekanslarının şekli için skewness değeri
ST2.skewstdfreX	X'nin standart sapmalarının histogramının şekli için skewness değeri
ST2.skewY	Y'nin skewness değeri
ST2.stdBoluMeanX1..10	X'nin standart sapmalarının ortalamalarına bölümlerinin 10 bin'lik histogram değerleri
ST2.stdcorXdeg	X'nin korelasyon matrisinin histogram değerlerinin standart sapma değeri
ST2.stdcorXfre	X'nin korelasyon matrisinin histogram frekanslarının standart sapma değeri
ST2.stdcorXYdeg	X'nin Y ile korelasyon matrisinin histogram değerlerinin standart sapma değeri
ST2.stdcorXYfre	X'nin Y ile korelasyon matrisinin histogram frekanslarının standart sapma değeri
ST2.stddegX1..10	X'nin standart sapmalarının 10 bin'lik histogram değerleri (küçükten büyüğe sıralanmış)
ST2.stdfreX1..10	ST2.stddegX1..10'in histogram'ının frekanslarının normalize değerleri
ST2.stdfreY	Y'nin histogram frekanslarının standart sapma değeri
ST2.stdkurtdegX	X'nin kurtosis histogramının değerlerinin standart sapması
ST2.stdkurtfreX	X'nin kurtosis histogramının frekanslarının standart sapması
ST2.stdmom3degX	X'nin 3.dereceden moment histogramının değerlerinin standart sapması
ST2.stdmom3freX	X'nin 3.dereceden moment histogramının frekanslarının standart sapması
ST2.stdmom4degX	X'nin 4.dereceden moment histogramının değerlerinin standart sapması
ST2.stdmom4freX	X'nin 4.dereceden moment histogramının frekanslarının standart sapması
ST2.stdskewdegX	X'nin skewness histogramının değerlerinin standart sapması
ST2.stdskewfreX	X'nin skewness histogramının frekanslarının standart sapması
ST2.stdstdfreX	X'nin standart sapma frekanslarının standart sapması
ST2.stdY	Y'nin standart sapma değeri

Çizelge 9.4'teki açıklamalarda X ve Y sembollerinin ifade ettikleri değerler:

$X \rightarrow$ Veri kümesinin giriş özellikleri

$Y \rightarrow$ Veri kümesinin çıkışları

Değerlerin *normalize* edilme işlemi, X ler için toplam özellik sayısına, Y ler için toplam örnek sayısına bölünerek yapılmıştır. Böylece toplamlarının 1 olması sağlanmıştır.

Çizelge 9.5 CLUS meta özellik grubu

Meta Özellik İsmi	Açıklaması
CLUS.EM	Veri kümesi EM algoritması ile kümelendiğinde oluşan kümelerin içerdikleri örnek oranlarının entropisi
CLUS.EM_kume_sayisi	Veri kümesi EM algoritması ile kümelendiğinde oluşan küme sayısı
CLUS.FF	Veri kümesi FF algoritması ile kümelendiğinde oluşan kümelerin içerdikleri örnek oranlarının entropisi
CLUS.kmean	Veri kümesi Kmean algoritması ile kümelendiğinde oluşan kümelerin içerdikleri örnek oranlarının entropisi
CLUS.xmean	Veri kümesi Xmean algoritması ile kümelendiğinde oluşan kümelerin içerdikleri örnek oranlarının entropisi

Çizelge 9.6’da kullanılan karar ağacı algoritmalarında ve Çizelge 9.7’de kullanılan diğer algoritmalarda, kullanıcının belirlediği tüm parametreler (hiper parametreler), WEKA yazılımının varsayılan (default) değerleri olarak kullanılmıştır. Bu sayede tüm veriler eşit şartlarda karşılaştırılabilmiştir.

Çizelge 9.6 REGT meta özellik grubu

Meta Özellik ismi	Açıklaması
REGT.m5p_yaprak_sayisi	Veri kümesi üzerinde oluşturulan M5P karar ağacındaki yaprak sayısı
REGT.m5p_yapraklardaki_tekil_oz_sayisi	Veri kümesi üzerinde oluşturulan M5P karar ağacının yapraklarında (kararlarında) en az 1 kere kullanılmış özellik sayısı
REGT.m5p_ysayi_cfsozsayi_orani	Veri kümesi üzerinde oluşturulan M5P karar ağacındaki yaprak sayısının CFS ile seçilmiş özellik

	sayısına oranı
REGT.m5p_ysayi_orsayi_orani	Veri kümesi üzerinde oluşturulan M5P karar ağacındaki yaprak sayısının örnek sayısına oranı
REGT.m5p_ysayi_ozsayi_orani	Veri kümesi üzerinde oluşturulan M5P karar ağacındaki yaprak sayısının örnek sayısına oranı
REGT.m5p_ytek_ozsayi_cfsozsayi_orani	Veri kümesi üzerinde oluşturulan M5P karar ağacının yapraklarında en az 1 kere kullanılmış özellik sayısının CFS ile seçilmiş özellik sayısına oranı
REGT.m5p_ytek_ozsayi_ozsayi_orani	Veri kümesi üzerinde oluşturulan M5P karar ağacının yapraklarında en az 1 kere kullanılmış özellik sayısının özellik sayısına oranı
REGT.m5r_ksayi_cfsozsayi_orani	Veri kümesi üzerinde M5rules ile bulunan kural sayısının CFS ile seçilmiş özellik sayısına oranı
REGT.m5r_ksayi_orsayi_orani	Veri kümesi üzerinde M5rules ile bulunan kural sayısının örnek sayısına oranı
REGT.m5r_ksayi_ozsayi_orani	Veri kümesi üzerinde M5rules ile bulunan kural sayısının özellik sayısına oranı
REGT.m5r_ksayitekozsayi_cfsozsayi_orani	Veri kümesi üzerinde M5rules ile bulunan kurallarda en az 1 kere geçmiş özellik sayısının CFS ile bulunan özellik sayısına oranı
REGT.m5r_ksayitekozsayi_ozsayi_orani	Veri kümesi üzerinde M5rules ile bulunan kurallarda en az 1 kere geçmiş özellik sayısının özellik sayısına oranı
REGT.m5r_kural_sayisi	Veri kümesi üzerinde M5rules ile bulunan kural sayısı
REGT.m5r_kurallardaki_tekil_oz_sayisi	Veri kümesi üzerinde M5rules ile bulunan kurallarda en az 1 kere geçmiş özellik sayısı
REGT.rep_nsayi_cfsozsayi_orani	Veri kümesi üzerinde oluşturulan RepTree karar ağacındaki düğüm sayısının CFS ile seçilen özellik sayısına oranı
REGT.rep_nsayi_orsayi_orani	Veri kümesi üzerinde oluşturulan RepTree karar ağacındaki düğüm sayısının CFS ile seçilen örnek

	sayısına oranı
REGT.rep_nsayi_ozsayi_orani	Veri kümesi üzerinde oluşturulan RepTree karar ağacındaki düğüm sayısının özellik sayısına oranı
REGT.reptree_node_sayisi	Veri kümesi üzerinde oluşturulan RepTree karar ağacındaki düğüm sayısı

Çizelge 9.7 RMSE meta özellik grubu

Özellik ismi	Özellik Açıklaması	Referans
RMSE.M5R	M5 rules algoritmasının veri kümesi üzerindeki RMSE değeri	(Wand vd., 1997)
RMSE.M5P	M5P algoritmasının veri kümesi üzerindeki RMSE değeri	
RMSE. Decstump	Decision Stump algoritmasının veri kümesi üzerindeki RMSE değeri	Tek karar düğümü, iki yaprağı olan bir karar ağacı üretir. Karar düğümünde tek bir özellik ve sınır değeri yer alır. Özellik ve sınır değerinin seçiminde MSE'yi kullanır.
RMSE.PLS2	Parçalı en küçük kareler (Partial Least Squares) algoritmasının 2 componentle veri kümesi üzerindeki RMSE değeri	(Abdi, 2003)
RMSE.PLS1	Partial Least Squares algoritmasının 1 componentle veri kümesi üzerindeki RMSE değeri	
RMSE.SLR	Simple Linear Regression algoritmasının veri kümesi üzerindeki RMSE değeri	Her bir özellik için lineer bir model oluşturur. En düşük MSE'ye sahip olanını seçip kullanır.
RMSE.SMO	Sequential minimal optimization algoritmasının veri kümesi üzerindeki RMSE değeri	(Shevade vd., 1999)
RMSE.SVM	SVM regresyon algoritmasının veri kümesi üzerindeki RMSE değeri	
RMSE.IBK	En yakın komşu (1 Nearest Neighbour) algoritmasının tek komşulukla veri	(Aha vd., 1991)

	kümesi üzerindeki RMSE değeri	
RMSE.ZeroR	Zero Rule algoritmasının veri kümesi üzerindeki RMSE değeri	Bütün örnekler için aynı değeri çıkış verir. Bu değeri ise eğitim örneklerinin çıkışlarının ortalamasıdır. Zero rule'un RMSE değeri bir regresyon veri kümesinin tahmin edilebilme zorluğu ile doğru orantılıdır.
RMSE.ConjunctiveRule	ConjunctiveRule algoritmasının veri kümesi üzerindeki RMSE değeri	Birbirine “ve”lerle bağlanmış özellik, sınır değeri ikililerinden oluşan tek bir kural üretir. Kuralı oluştururken InfoGain ve Reduced Error Pruning kullanır.
RMSE.Linear Regression	LinearRegression algoritmasının veri kümesi üzerindeki RMSE değeri	Tüm özellikleri içeren lineer bir model oluşturur.
RMSE. RBF	Radial Basis Fonksiyonlar algoritmasının veri kümesi üzerindeki RMSE değeri	Önce Kmeans algoritmasıyla küme merkezleri seçer, daha sonra her kümede bir gaussian radial tabanlı fonksiyon kullanır.
RMSE.Kstar	Kstar algoritmasının veri kümesi üzerindeki RMSE değeri	(Cleary vd., 1995)
RMSE.LWL	Locally weighted learning algoritmasının veri kümesi üzerindeki RMSE değeri	(Frank vd., 2003)

Bir veri kümesinde bir sınıflandırıcının başarısı yüksek ise o veri kümesi, o sınıflandırıcının varsaydığı özellikleri sağlıyor denebilir. Bu açıdan bakıldığında lineer bir sınıflandırıcının başarısı, veri kümesinin lineer ayrılabilirliğinin bir ölçütü olarak, Naive Bayes'in başarısı veri kümesindeki özelliklerin birbirlerinden bağımsızlığının bir ölçütü olarak, K En Yakın Komşu'nun başarısı veri kümesindeki örneklerin kümelenebilirliklerinin bir ölçütü olarak görülebilir (Bensusan vd., 2000a).

Çizelge 9.8 PCA meta özellik grubu

Meta özellik ismi	Açıklaması
PCA.expdeg1..10	Bulunan her bir temel bileşenin verinin varyansının yüzde kaçını açıkladığı içeren bir listenin 10 bin’lik histogram değerleri (küçükten büyüğe sıralanmış)
PCA.expfre1..10	PCA.expdeg1..10’deki histogramın frekanslarının normalize değerleri
PCA.explained_1	İlk temel bileşenin verinin varyansının yüzde kaçını açıkladığı
PCA.x95	Yeni uzayda verinin varyansının %95’ini açıklayan temel bileşen sayısının, tüm özellik sayısına oranı

Değerlerin normalize edilme işlemi toplamlarına bölünerek yapılmıştır. Böylece toplamlarının 1 olması sağlanmıştır.

9.4.2 Yapay Veri Kümeleri Koleksiyonunda Meta Regresyon

Veri kümelerinin bazı özelliklerinin kontrol altında tutulabilmesi ve istenilen özelliklerde istenildiği kadar çok örnek ve özellik üretilebilmesi için literatürde birçok çalışma yapay veri kümeleri üzerinde yapılmaktadır. Meta regresyon denemelerinde de bu bölümde böyle yapay olarak üretilen 80 adet veri kümesi kullanılmıştır. Literatürde veri kümesi üretiminde en çok kullanılan fonksiyonlardan birisi *friedman* fonksiyonudur.

9.4.2.1 Friedman Yapay Veri Kümelerinin Üretilmesi

Friedman fonksiyonunun özelliği, hem lineer, hem de lineer olmayan ilişkileri içermesidir ve normal dağılımlı bir gürültü (ϵ) de çıkışa eklenmiştir. Bu fonksiyon Eşitlik 9.1’de verilmiştir.

$$y = 10 * \sin(\pi * x_1 * x_2) + 20 * (x_3 - 0.5)^2 + 10 * x_4 + 5 * x_5 + \epsilon \quad (9.1)$$

Orijinal friedman veri kümesinde 5 özellik vardır, ancak çıkışla ilgisiz girişlerin sistemin performansı üzerindeki etkilerinin de belirlenebilmesi için yeni özellikler eklenmiştir. Bunlar çıkıştan tamamen bağımsız, ilgisiz özelliklerdir.

Friedmannın bu özelliklerine ek olarak fonksiyona özellikler arası colinearity de eklendi. 5 farklı seviyede colinearity ile veri kümeleri üretildi. Bunun sebebi algoritmaların colinearity’ye olan dayanırlılıklarını, güvenilirliklerini incelemektir.

Friedman veri kümeleri üretilirken kullanılan parametreler ve değerleri:

Özellik sayısı: 5 10 25 50 100

Örnek sayısı: 100 250 500 1000

Colinearity derecesi : 0 1 2 3 4

Colinearity derecesi 4 olan veri kümeleri için özellik sayısı 10, 25, 50 ve 100 alınmıştır.

Colinearity derecesi 0, 1, 2 ve 3 olanlar için ise 5, 10, 25 ve 50 alınmıştır.

Toplamda, 4 farklı sayıda özellik sayısı * 4 farklı sayıda örnek sayısı * 5 farklı seviyede colinearity = 80 adet yapay veri kümesi üretilmiştir.

9.4.2.2 Friedman Veri Koleksiyonunda Kullanılan Ek Meta Özellikler

Tüm veri koleksiyonları için ortak kullanılan meta özelliklere ek olarak Friedman koleksiyonu için 12 adet ek meta özellik daha kullanılmıştır. Çizelge 9.9’da bu meta özellikler verilmiştir. Çizelgedeki meta algoritmaların (başka algoritmaları kullanan algoritmalar) yanlarında içerdikleri algoritmaların isimleri de verilmiştir.

Çizelge 9.9 Friedman koleksiyonu için ek meta özellikler

Meta Özellik İsmi	Açıklaması
Colli	Colinearity derecesi {0,1,2,3,4}
RMSE.GaussianProcesses	Algoritmaların veri kümesi üzerindeki RMSE değerleri
RMSE.meta.AdditiveRegression (Decision Stump)	
RMSE.meta.AttributeSelectedClassifier (M5P)	
RMSE.meta.Bagging (RepTree)	
RMSE.meta.Dagging (IBK)	
RMSE.meta.EnsembleSelc. (SmoReg – m5rules – Zero0)	
RMSE.meta.RandomSubSpace (RepTree)	
RMSE.meta.RegressionByDiscretization (C4.5)	
RMSE.meta.Stacking	
RMSE.meta.Vote (SmoReg – m5rules – Zero0)	
RMSE.REPTree	

9.4.2.3 Friedman Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri

Üretilen 80 yapay veri kümesinde algoritmaların performansları incelenmiş ve en başarılı algoritmalar belirlenmiştir. Çizelge 9.10’da bu algoritmalar verilmiştir.

Çizelge 9.10 80 veri kümesinde minimum RMSE değeri içeren algoritmalar

Algoritma adı	Algoritmanın, 80 veri kümesinde en başarılı algoritma olduğu veri kümesi sayısı
meta.Bagging	33
M5P	18
meta.AttributeSelectedClassifier	9
meta.AdditiveRegression	8
GaussianProcesses	7
Kstar	4
LinearRegression	1

Çizelge 9.11’de ise Meta algoritmalar kullanılmadığındaki başarılı algoritmalar verilmiştir.

Çizelge 9.11 Meta algoritmalar çıkarıldığında 80 veri kümesinde minimum RMSE değeri içeren algoritmalar

Algoritma adı	Algoritmanın, 80 veri kümesinde en başarılı algoritma olduğu veri kümesi sayısı
M5P	58
Kstar	9
GaussianProcesses	7
LinearRegression	2
M5R	2
REPTree	1
ConjunctiveRule	1

Çizelge 9.12’de tüm algoritmaların tüm veri kümeleri üzerindeki RMSE’lerinin minimum, maksimum, ortalama ve standart sapma değerleri verilmiştir.

Çizelge 9.12 21 algoritmanın 80 veri kümesi üzerindeki performansları

Algoritma ismi	Minimum RMSE	Maksimum RMSE	Ortalama RMSE	Standart sapma RMSE
ConjunctiveRule	0.7300	1.0200	0.8735	0.0616
Decstump	0.7300	1.0200	0.8690	0.0604
GaussianProcesses	0.2800	1.0000	0.8056	0.1801
IBK	0.3600	1.3800	1.0091	0.2775
Kstar	0.3200	1.3100	0.8890	0.2700
LinearRegression	0.5200	4.7900	0.8618	0.4669
LWL	0.7100	0.9800	0.8134	0.0531
M5P	0.3100	0.8100	0.5055	0.1212
M5R	0.3500	0.8900	0.5401	0.1200
meta.AdditiveRegression	0.4533	0.8805	0.5514	0.0746
meta.AttributeSelectedClassifier	0.3519	1.0011	0.6001	0.1220
meta.Bagging	0.3136	0.8779	0.5009	0.1241
meta.Dagging	0.3353	1.0018	0.8012	0.1655
meta.EnsembleSelection	0.4509	1.0175	0.7073	0.1241
meta.RandomSubSpace	0.4319	0.9094	0.6168	0.0987
meta.RegressionByDiscretization	0.4008	1.0223	0.6413	0.1409
meta.Stacking	0.9711	0.9995	0.9941	0.0056
meta.Vote	0.5589	1.1743	0.7115	0.0947
PLS1	0.5200	1.0600	0.8361	0.1302
PLS2	0.5200	1.1000	0.8284	0.1438
PLS3	0.5200	1.1900	0.8369	0.1581
PLS4	0.5200	1.2900	0.8405	0.1701
PLS5	0.5200	1.3900	0.8444	0.1823
RBF	0.7800	1.0000	0.9342	0.0625
REPTree	0.4200	1.0000	0.6236	0.1289
SLR	0.7800	0.9700	0.8834	0.0420
SMO	0.5200	2.5300	0.8982	0.2706
SVM	0.5200	2.5300	0.8981	0.2706
ZeroR	0.9700	1.0000	0.9949	0.0069

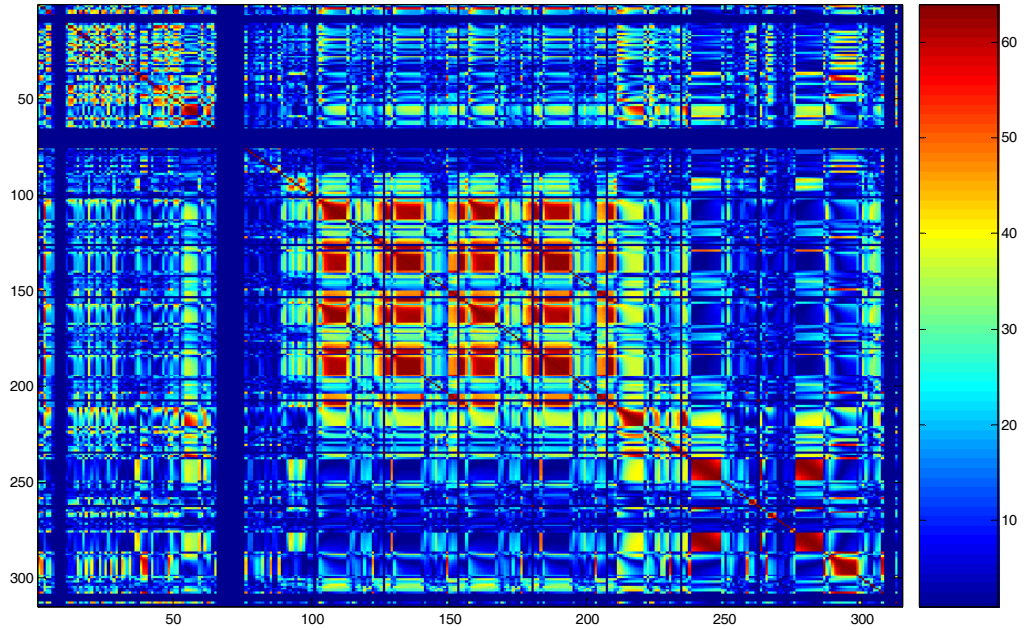
Çizelge 9.12 incelendiğinde aşağıdaki sonuçlara ulaşılmıştır:

- Ortalama olarak en başarılı algoritmalar Meta.Bagging ve M5P'dir.
- En başarılı 6 algoritmanın ortalama başarı sıralaması

- Meta.Bagging > M5P > M5rules > meta.AttributeSelectedClassifier > meta.RandomSubSpace > Reptree şeklindedir.
- PLS algoritmasında indirgenen boyut sayısının sonuca pek fazla bir etkisi olmadığı görülmüştür.
- Standart sapması en büyük olan algoritmalar sırasıyla LinearRegression, IBK, SMOreg, SVMreg ve KStar'dır.

9.4.2.4 Friedman Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri

Şekil 9.1'de 314 meta özelliğin birbirleriyle korelasyonu gösterilmiştir. Negatif korelasyonlarda anlamlı olacağı için korelasyon matrisi görselleştirilirken mutlak değeri alınmıştır.



Şekil 9.1 Friedman koleksiyonunun meta özelliklerinin korelasyon matrisi

Renkler maviden kahverengiye doğru ilerledikçe iki özellik arasındaki korelasyonun yüksekliği artmaktadır.

Şekil incelendiğinde aynı gruplarda yer alan meta özelliklerin birbirleriyle korelasyonlarının

yüksek olduğu görülmektedir. Özellikle ST2 grubundaki özellikler (şeklin ortasında yer alan) birbirleriyle oldukça ilişkilidirler.

Korelasyon katsayısının mutlak değerinin 0.8'den büyük olduğu 1707 özellik ikilisi bulunmuştur. Çizelge 9.13'te bu 1707 ikilinin meta özellik gruplarındaki dağılımları verilmiştir. Çizelgenin ilk satırında meta özellik gruplarında yer alan meta özellik sayıları verilmiştir.

Çizelge 9.13 Friedman koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı

	5	29	18	220	15	22	1
	CLUS	RMSE	REGT	ST2	STA	PCA	colli
CLUS	1	5	2				
RMSE		55	26	46	6	31	
REGT			24	1	2		
ST2				1346	3	54	42
STA					1	3	
PCA						57	

Asıl önemli olan farklı gruplarda yer alan meta özelliklerin birbirleriyle ilişkili olmasıdır. Bunlardan önemli görülenler Çizelge 9.14'te listelenmiştir.

Çizelge 9.14 Friedman meta özelliklerinin korelasyonu yüksek olanlarından seçmeler

Meta özellik 1	Meta özellik 2	Korelasyon katsayısı
CLUS.EM	RMSE.meta.RegressionByDiscretization	-0.89972
CLUS.EM	RMSE.meta.RandomSubSpace	-0.8593
CLUS.EM	RMSE.meta.Bagging	-0.84708
CLUS.EM	RMSE.REPTree	-0.83431
CLUS.EM_kume_sayisi	REGT.rep_nsayi_cfsozsayi_orani	0.84946
CLUS.EM_kume_sayisi	REGT.m5p_ysayi_cfsozsayi_orani	0.83332
CLUS.EM_kume_sayisi	RMSE.meta.RegressionByDiscretization	-0.82157
CLUS.EM_kume_sayisi	REGT.m5r_ksayi_cfsozsayi_orani	0.81829
CLUS.EM_kume_sayisi	REGT.m5p_ysayi_ozsayi_orani	0.80543
Colli	ST2.skewdegX10	0.92522
Colli	ST2.mom3degX10	0.9247
Colli	ST2.kurtdegX10	0.89536

Colli	ST2.mom4degX10	0.89313
REGT.m5p_MAE	STA.orneksayi	-0.82729
REGT.m5r_ksayitekozsayi_ozsayi_orani	ST2.corXdeg2	0.81844
REGT.rep_nsayi_ozsayi_orani	RMSE.meta.Dagging	-0.84004
RMSE.GaussianProcesses	PCA.expdeg2	-0.89283
RMSE.GaussianProcesses	PCA.expdeg3	-0.8481
RMSE.GaussianProcesses	ST2.kurtfreX10	-0.83104
RMSE.GaussianProcesses	ST2.mom4freX10	-0.83104
RMSE.GaussianProcesses	PCA.expdeg1	-0.831
RMSE.GaussianProcesses	ST2.bigcorXYpro	-0.80067
RMSE.IBK	PCA.expfre10	-0.94675
RMSE.IBK	ST2.bigcorXYpro	-0.94441
RMSE.IBK	ST2.kurtfreX10	-0.90454
RMSE.IBK	ST2.mom4freX10	-0.90454
RMSE.IBK	ST2.corXfre10	-0.87163
RMSE.IBK	ST2.corXYfre10	-0.8247
RMSE.Kstar	ST2.bigcorXYpro	-0.9559
RMSE.Kstar	PCA.expfre10	-0.95387
RMSE.Kstar	ST2.kurtfreX10	-0.90507
RMSE.Kstar	ST2.mom4freX10	-0.90507
RMSE.Kstar	ST2.corXYfre10	-0.84744
RMSE.Kstar	PCA.explained_1	-0.84672
RMSE.Kstar	ST2.corXfre10	-0.82479
RMSE.meta.Bagging	ST2.corXdeg1	-0.80859
RMSE.meta.Dagging	PCA.expfre10	-0.88776
RMSE.meta.Dagging	ST2.bigcorXYpro	-0.88471
RMSE.meta.Dagging	ST2.kurtfreX10	-0.8712
RMSE.meta.Dagging	ST2.mom4freX10	-0.8712
RMSE.PLS1	ST2.stdcorXdeg	0.95842
RMSE.PLS1	ST2.corXdeg10	0.94789
RMSE.PLS2	ST2.stdcorXdeg	0.9587
RMSE.PLS2	ST2.corXdeg10	0.92743
RMSE.PLS3	ST2.stdcorXdeg	0.93662
RMSE.PLS3	ST2.corXdeg10	0.87633
RMSE.PLS4	ST2.stdcorXdeg	0.90346
RMSE.PLS4	ST2.corXdeg10	0.82036
RMSE.PLS5	ST2.stdcorXdeg	0.87065
ST2.bigcorXpro	PCA.explained_1	0.89756
ST2.bigcorXYpro	PCA.expfre10	0.99731
ST2.bigcorXYpro	PCA.explained_1	0.87093
ST2.corXfre10	ST2.bigcorXYpro	0.88679

ST2.corXfre10	PCA.explained_1	0.82831
ST2.corXYfre10	PCA.expfre10	0.87165
ST2.kurtcorXfre	ST2.skewcorXfre	0.93702
ST2.kurtcorXYfre	ST2.skewcorXYfre	0.96166
ST2.mom4freX10	ST2.bigcorXYpro	0.94257
ST2.mom4freX10	PCA.expfre10	0.94074
ST2.mom4freX10	ST2.corXfre10	0.81207
ST2.mom4freX10	ST2.corXYfre10	0.80488
ST2.skewfreX10	PCA.expfre10	0.83737
ST2.skewfreX10	ST2.bigcorXYpro	0.83608
STA.cfs_oran	ST2.bigcorXYpro	0.89036
STA.cfs_oran	PCA.expfre10	0.88623
STA.cfs_oran	ST2.kurtfreX10	0.86496
STA.cfs_oran	ST2.mom4freX10	0.86496
STA.cfs_oran	RMSE.Kstar	-0.85206
STA.cfs_oran	PCA.expdeg2	0.83647
STA.cfs_oran	PCA.expdeg3	0.82854
STA.cfs_oran	RMSE.IBK	-0.81778
STA.cfs_oran	RMSE.GaussianProcesses	-0.81534
STA.cfs_sayi	STA.ozelliksayi	0.87097
STA.orneksayi	REGT.reptree_node_sayisi	0.92682
STA.orneksayi	RMSE.M5P	-0.83164
STA.orneksayi	RMSE.REPTree	-0.80881
STA.orneksayi	RMSE.M5R	-0.80189

Çizelge 9.14 ve korelasyon matrisi (Şekil 9.1) incelendiğinde aşağıdaki sonuçlara ulaşılmıştır.

- PLS ailesi birbiriyle ilişkilidir. Ancak PLS'nin bileşen sayısı arttıkça ilişki azalmaktadır.
- Örnek sayısı M5P, Reptree ve M5rules algoritmalarının hataları ile ters ilişkilidir. Diğer bir ifadeyle, örnek sayısı arttıkça algoritmaların performansı da artmaktadır.
- Colinearity derecesi skewness, kurtosis, 3. ve 4. dereceden momentlerle ilişkilidir. Bu ilişki colinearity derecesi bilinmeyen veri kümelerinin colinearity tahmininde kullanılabilir.
- Expectation Minimization algoritmasının ürettiği kümelerdeki örnek sayısının dağılımının entropisi ile RMSE grubundan bazı algoritmaların (meta.RegressionByDisc, meta.randomSubspace, meta.Bagging, RepTree) hataları ters

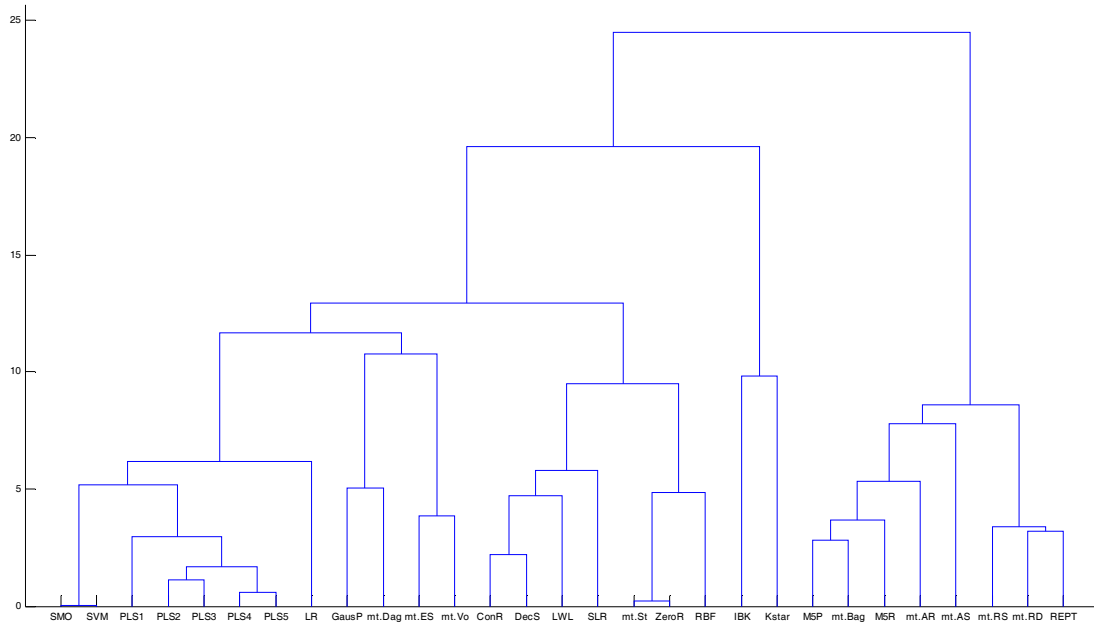
orantılıdır. Entropi ne kadar büyükse (örnekler ne kadar çok kümeye dağılmışsa) hatalar o kadar azalmaktadır.

- Expectation Minimization algoritmasının ürettiği küme sayısı ile karar ağaçlarındaki yaprak / kural sayısının özellik sayısına bölümü doğru orantılıdır. Kurallardaki yaprak sayısı ne kadar fazla, özellik sayısı ne kadar azsa, EM o kadar fazla küme üretmiştir. Diğer bir deyişle, üretilen karar ağaçları ne kadar büyükse, EM'in bulduğu küme sayısı o kadar fazladır.
- Verilerin gaussian dağıldığını varsayan, GaussProcess algoritmasının hatası verilerin dağınıklığı gösteren kurtosis ve moment meta özellikleriyle doğru orantılıdır. Veriler ne kadar dağınıksa (gaussian'lıktan uzaksa) GaussianProcess algoritması o kadar büyük hatalar yapmaktadır.
- Girişlerin çıkışla korelasyonunun büyüklüğünü gösteren ST2.bigcorXYpro ile IBK algoritmasının hatası ters orantılıdır. Korelasyon ne kadar büyükse IBK o kadar az hata yapmaktadır.
- Girişlere PCA uygulandığında ne kadar az PCA bileşen verinin varyansının çoğunu içeriyorsa, IBK algoritmasının hatası o kadar azdır.
- Verinin dağınıklığını ölçen kurtosis ve moment ne kadar büyükse (veri ne kadar dağınıksa) IBK algoritmasının hatası o kadar azdır.
- Girişlerin birbirleriyle korelasyonu ne kadar büyükse IBK algoritmasının hatası o kadar azdır.
- IBK ve Kstar algoritmalarının her ikisi de örnek tabanlı algoritmalar oldukları için performansları arasında doğru orantı vardır ve bu nedenle Kstar'ın performansları da IBK'nın ilişkili olduğu meta özelliklerle ilişkilidir.
- PLS algoritmasının tüm versiyonlarını hataları, girişlerin birbirleriyle korelasyonlarıyla doğru orantılıdır. Girişler birbirleriyle ne kadar korele ise PLS algoritmaları o kadar çok hata yapmaktadır. Bu sonuç şaşırtıcıdır. Çünkü literatürdeki PLS algoritmasının bu özelliğe olan ilişkisizliği ile ters düşmektedir.
- Girişlerle çıkışların korelasyonu ne kadar büyükse, PCA uygulandığında verilerin varyansını açıklamak için o kadar az PCA bileşeni gerekmektedir.

- Girişlerle çıkışların korelasyonu ne kadar büyükse, verilerin dağınıklığı o kadar fazladır.
- İlgili özelliklerin oranı ne kadar büyükse, giriş çıkış korelasyonu ve ilk PCA bileşenlerinin içerdiği varyans o kadar fazladır.
- İlgili özelliklerin oranı ne kadar büyükse, verinin dağınıklığı o kadar fazladır.
- İlgili özelliklerin oranı ne kadar büyükse, IBK, Kstar ve GaussianProcess algoritmaları o kadar az hata yapmaktadır. Diğer bir deyişle, veri kümesi ne kadar az ilgisiz özellik içerirse, bu algoritmalar o kadar başarılı olmaktadır.

9.4.2.5 Friedman Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümelenmeleri

Algoritmaların veri kümeleri üzerlerindeki performanslara göre hiyerarşik kümelemesi yapıldığında Şekil 9.2 elde edilmiştir.

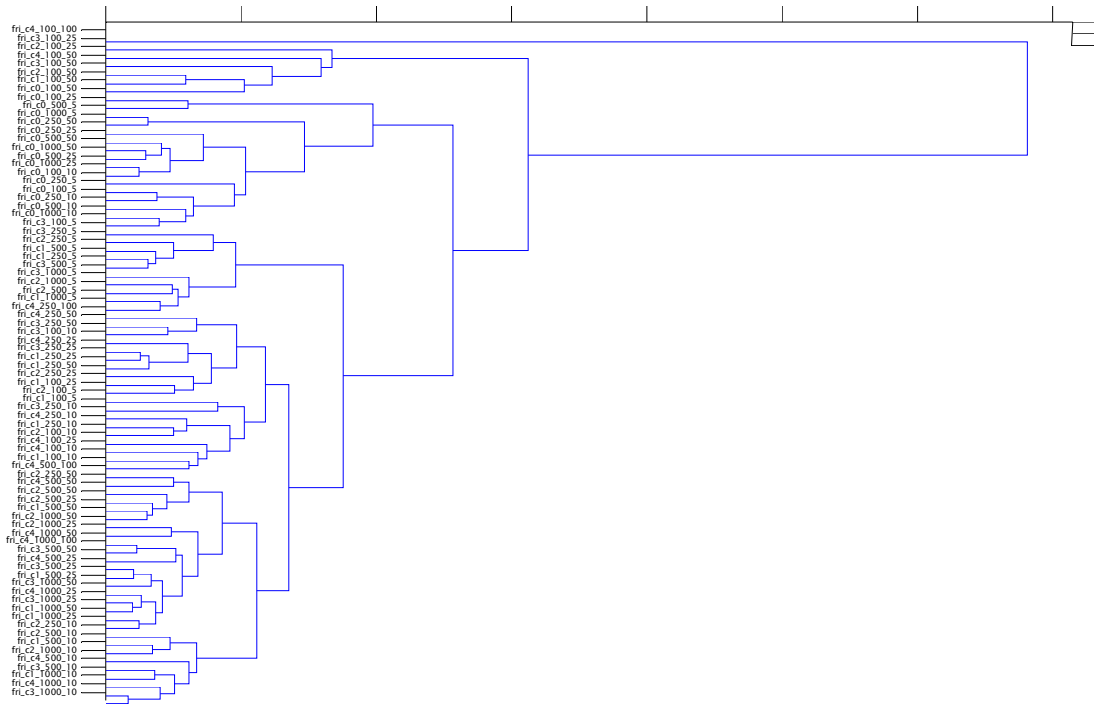


Şekil 9.2 Yapay veri koleksiyonunda algoritmaların RMSE değerlerine göre (80 boyutlu) hiyerarşik kümelenmeleri

Şekil 9.2 incelendiğinde aşağıdaki sonuçlara ulaşılmıştır:

- Lineer karakteristiğe sahip algoritmalar bir kümede toplanmışlardır.
- Tek kural üreten algoritmalar (Conjunctive Rule ve Decision Stump) bir küme de toplanmıştır.
- Örnek tabanlı algoritmalar (Kstar ve IBK) bir küme de toplanmıştır.
- Karar ağaçları ve meta algoritmalar (M5P, M5R, RepTree, meta.Bagging, meta.Attributeselcted, meta.AdditiveRegresyon, meta.RandomSubspace ve meta.Regression byDiscrization) bir küme de toplanmıştır. Buna sebep olarak meta algoritmaların içlerinde genelde karar ağacı algoritmalarının olması verilebilir.

Veri kümelerinin aynı şekilde kümelemesi yapıldığında Şekil 9.3 elde edilmiştir. Veri kümesi isimleri şeklin sol tarafında yer almaktadır ve “fri_ colinearityderecesi_ orneksayisi_ boyutsayısı” şeklinde kodlanmıştır.



Şekil 9.3 Yapay veri koleksiyonunda veri kümelerinin RMSE değerlerine göre (29 boyutlu) hiyerarşik kümelennmeleri

Şekil 9.3 incelendiğinde veri kümelerinin çeşitli alt kümeler oluşturdukları görülmektedir. Bu kümeler kabaca aşağıdaki gibi tanımlanabilir:

- 100 örnek içerenler
- 1000 veya 500 örnek içerenler
- 5 boyutlu olanlar
- 10 boyutlu olanlar
- Colinearity'si 0 olanlar

Bu gruplar incelendiğinde friedman veri kümelerinin genelde boyut, örnek sayısı ve özelliklerin birbiriyle korelasyonunun bir ölçütü olan colinearity'nin var olup olmamasına göre gruplandığı görülmektedir.

9.4.2.6 Friedman Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri

Ortalama olarak en başarılı 6 algoritmadan 5'inin RMSE hataları tahmin edilmeye çalışılmıştır. M5P ve M5rules algoritmalarının korelasyonları çok yüksek olduğundan sadece M5P'nin sonuçları tahmin edilmiştir.

Algoritmaların performanslarının tahmininde 6 grup meta özelliğin hangilerinin daha etkili olduğunun bulunması için:

1. 5 grubun birden
2. 5 gruptan özellik seçimi yapıldıktan sonra

yukarıdaki 2 şekilde M5P ile (10'lu çapraz geçerleme) denemeler yapılmıştır. Bu sayede hangi tür meta özellikleri ve hangi meta özelliklerin performans tahmininde daha başarılı olduğu görülmüştür. Burada M5P kullanılmasının nedeni hem yüksek performansı hem de ürettiği sonuçların kolay yorumlanabilir olmasıdır.

Diğer algoritmaların performanslarının tahmininde kullanılan algoritmalar:

- Decision Stump
- IBK
- Linear Regression
- SLR

- Zero0

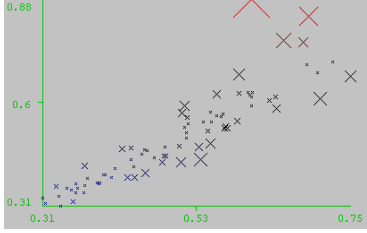
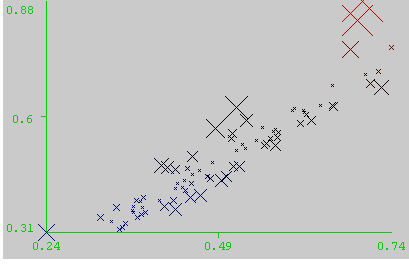
olarak seçilmiştir. Bu algoritmaların seçiliş nedeni, işlem karmaşıklığı düşük ve dolayısıyla hızlı sonuç üretebilen algoritmalar olmalarıdır.

Tahminlerin değerlendirilmesinde çıkışların aralıkları aynı olmadığından RMSE yerine Rölatif mutlak hata (RAE)'larına göre karşılaştırma yapılmıştır.

Çizelge 9.15, 9.16, 9.17, 9.18, ve 9.19'da algoritmaların performanslarının tahmin denemeleri verilmiştir. Bu çizelgelerde yer alan şekillerde X boyutu algoritmanın veri kümesi üzerindeki performansının tahmin edilen değerini Y boyutu ise gerçek değerini göstermektedir.

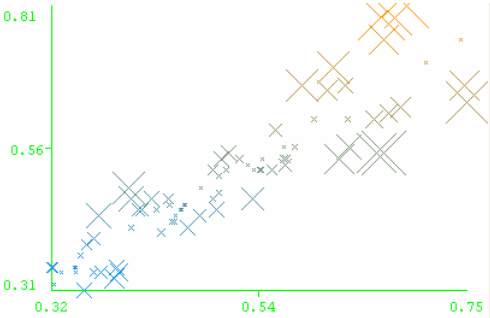
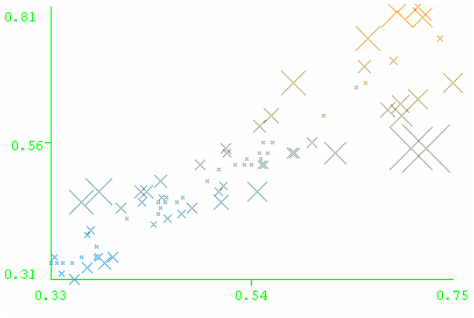
Çizelge 9.15 Friedman koleksiyonunda Meta.Bagging tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (19 özellikle)
RMSE.meta.Bagging = - 0.3694 * CLUS.kmean - 0.0001 * STA.orneksayi + 0.2173 * REGT.rep_nsayi_orsayi_orani - 0.7519 * ST2.freY2 + 0.0598 * ST2.skewdegX5 - 0.2929 * ST2.corXdeg1 - 0.2202 * ST2.corXdeg2 - 0.0062 * ST2.kurtcorXfre + 0.9892 Korelasyon katsayısı 0.9064 Ortalama mutlak hata 0.0345 Ortalama karesel hatanın karekökü 0.0522	RMSE.meta.Bagging = -0.0416 * CLUS.EM - 0.2095 * CLUS.kmean + 1.8724 * REGT.m5r_ksayi_orsayi_orani - 0.0001 * STA.orneksayi + 0.2884 * RMSE.Decstump - 1.0554 * ST2.freY2 + 0.1144 * ST2.skewdegX1 + 0.2049 * ST2.kurtfreX7 - 0.3147 * ST2.corXdeg1 - 0.0378 * ST2.corXfre1 + 0.0778 * STA.pertumsayi + 0.5919

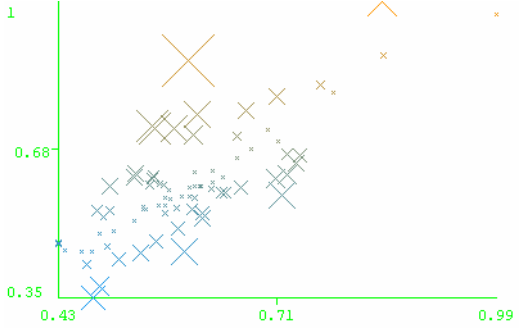
Rölatif mutlak hata	33.8624 %	Korelasyon katsayısı	0.9372
		Ortalama mutlak hata	0.0315
		Ortalama karesel hatanın karekökü	0.043
		Rölatif mutlak hata	30.8655 %
 <p>Görüldüğü gibi büyük hataları olan veri kümelerinde algoritmanın performansının tahmini zor, diğerlerinde daha kolaydır.</p>			

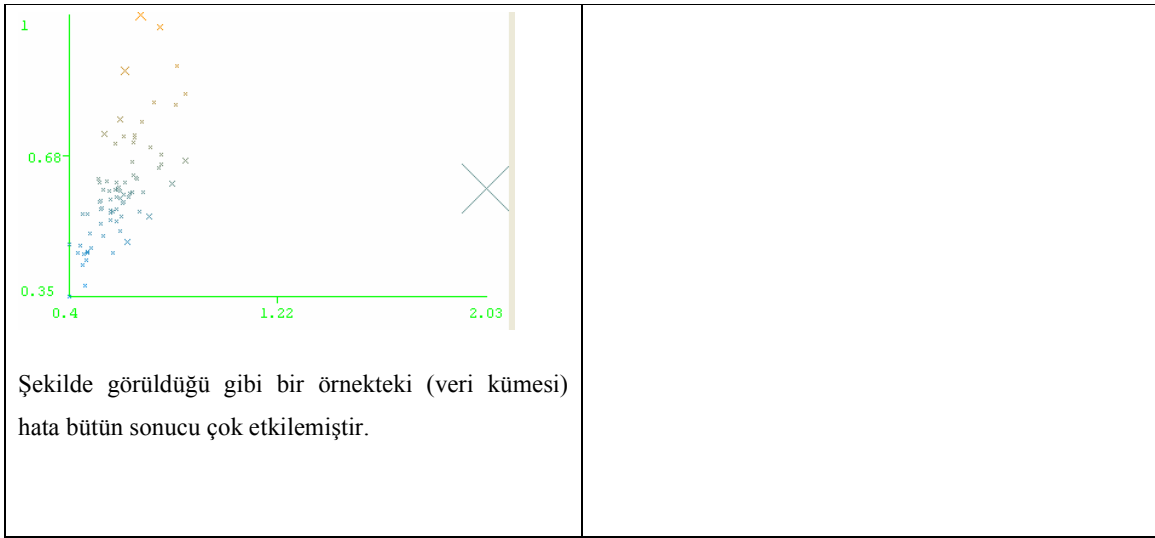
Çizelge 9.16 Friedman koleksiyonunda M5P tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (108 özellikle)
Performans uzayı 9 alt uzaya bölmüş ve her birinde	Performans uzayı 7 alt uzaya bölmüş ve her birinde
STA.cfs_oran	CLUS.FF
STA.cfs_sayi	REGT.m5p_ytek_ozsayi_ozsayi_orani
colli	REGT.m5r_ksayi_ozsayi_orani
CLUS.EM	STA.orneksayi
REGT.m5p_yapraklardaki_tekil_oz_sayisi	ST2.stdfreX5
REGT.m5r_ksayi_ozsayi_orani	ST2.freY4
REGT.m5r_ksayitekozsayi_cfsozsayi_orani	ST2.skewfreY
STA.orneksayi	ST2.skewfreX1
REGT.rep_nsayi_cfsozsayi_orani	ST2.kurtkurtfreX
ST2.stdfreX5	ST2.kurtY
ST2.freY4	ST2.corXdeg2
ST2.freY7	ST2.corXdeg8

<p>ST2.skewfreY</p> <p>ST2.skewfreX1</p> <p>ST2.skewfreX2</p> <p>ST2.skewfreX7</p> <p>ST2.corXdeg2</p> <p>ST2.stdcorXYdeg</p> <p>ST2.stdBoluMeanX5</p> <p>ST2.stdBoluMeanX6</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.9117</p> <p>Ortalama mutlak hata 0.0379</p> <p>Ortalama karesel hatanın karekökü 0.0499</p> <p>Rölatif mutlak hata 39.1983 %</p>  <p>Görüldüğü gibi meta.bagging algoritmasına göre daha kötü tahminler yapılabilmektedir.</p>	<p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.9133</p> <p>Ortalama mutlak hata 0.0367</p> <p>Ortalama karesel hatanın karekökü 0.0498</p> <p>Rölatif mutlak hata 38.0076 %</p> 
--	--

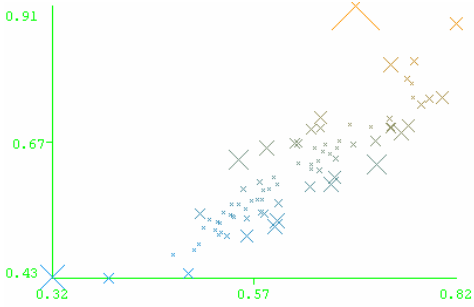
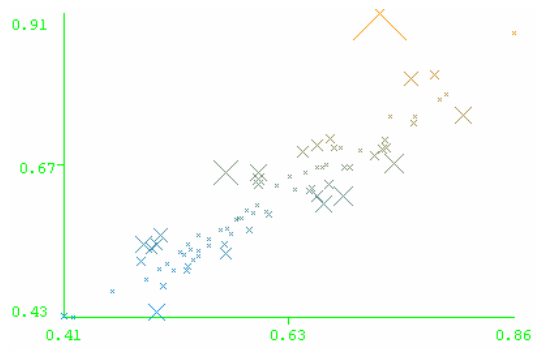
Çizelge 9.17 Friedman koleksiyonunda meta.AttributeSelectedClassifier tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (7 özellikle)
<p>Performans uzayı şeklinde 4 alt uzaya bölmüş ve her birinde</p> <p>colli</p> <p>CLUS.kmean</p> <p>REGT.m5p_yaprak_sayisi</p> <p>REGT.m5p_yapraklardaki_tekil_oz_sayisi</p> <p>REGT.m5p_ysayi_orsayi_orani</p> <p>REGT.rep_nsayi_orsayi_orani</p> <p>RMSE.Decstump</p> <p>ST2.freY3</p> <p>ST2.stdcorXdeg</p> <p>ST2.bigcorXpro</p> <p>ST2.corXYfre6</p> <p>ST2.corXYfre7</p> <p>PCA.expfre1</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.3761</p> <p>Ortalama mutlak hata 0.0829</p> <p>Ortalama karesel hatanın karekökü 0.183</p> <p>Rölatif mutlak hata 93.2214 %</p>	<p>RMSE.meta.AttributeSelectedClassifier =</p> $2.4548 * \text{REGT.m5r_ksayi_orsayi_orani} + 0.7754 * \text{RMSE.Decstump} + 0.171 * \text{ST2.corXfre3} + 0.3386 * \text{ST2.stdcorXdeg} - 0.834 * \text{ST2.stdcorXYfre} - 0.1244$ <p>Korelasyon Katsayısı 0.7853</p> <p>Ortalama mutlak hata 0.0559</p> <p>Ortalama karesel hatanın karekökü 0.0752</p> <p>Rölatif mutlak hata 62.8812 %</p>  <p>Özellik seçiminin performansa önemli katkısı olmuştur.</p>



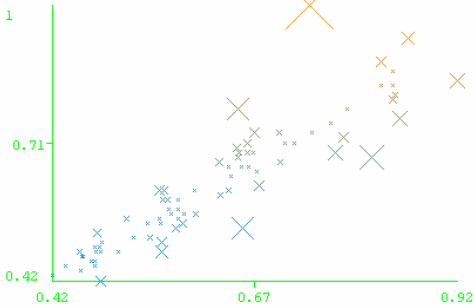
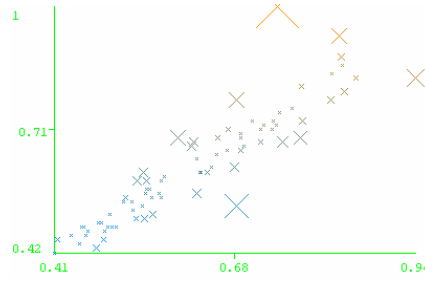
Çizelge 9.18 Friedman koleksiyonunda meta.RandomSubSpace tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (25 özellikle)
Performans uzayı 7 alt uzaya bölmüş ve her birinde	RMSE.meta.RandomSubSpace =
STA.cfs_sayı	-0.0386 * CLUS.EM
colli	- 0.229 * CLUS.kmean
CLUS.EM	- 0.0092 * REGT.m5p_ysayı_ozsayi_orani
CLUS.kmean	+ 0.2023 * RMSE.Decstump
REGT.m5r_ksayı_ozsayi_orani	+ 0.4148 * ST2.freY1
ST2.kurtfreY	- 0.7704 * ST2.freY2
ST2.skewdegX7	- 0.3635 * ST2.corXdeg1
ST2.corXdeg1	- 0.3351 * ST2.stdcorXfre
ST2.corXdeg2	+ 0.8121
özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.	Korelasyon Katsayısı 0.9284
Korelasyon Katsayısı 0.8897	Ortalama mutlak hata 0.0258
Ortalama mutlak hata 0.0328	Ortalama karesel hatanın karekökü 0.0369
	Rölatif mutlak hata 31.9506 %

<p>Ortalama karesel hatanın karekökü 0.0459</p> <p>Rölatif mutlak hata 40.6566 %</p>  <p>Şekilde görüldüğü gibi birkaç örnekteki (veri kümesi) hata bütün sonucu çok etkilemiştir.</p>	 <p>Özellik seçiminin performansa önemli katkısı olmuştur.</p>
---	--

Çizelge 9.19 Friedman koleksiyonunda Reptree tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (20 özellikle)
<p>RMSE.Reptree=</p> <p>0.0166 * colli</p> <p>- 0.0305 * CLUS.EM</p> <p>- 0.6534 * CLUS.kmean</p> <p>- 0.001 * REGT.m5p_yaprak_sayisi</p> <p>+ 0.0339 * REGT.m5p_ytek_ozsayi_ozsayi_orani</p> <p>- 0.0002 * STA.orneksayi</p> <p>+ 0.0007 * REGT.rep_nsayi_cfsozsayi_orani</p> <p>+ 0.5509 * RMSE.SLR</p>	<p>RMSE.Reptree=</p> <p>-0.0375 * CLUS.EM</p> <p>- 0.3848 * CLUS.kmean</p> <p>+ 1.5182 * REGT.m5r_ksayi_orsayi_orani</p> <p>- 0 * STA.orneksayi</p> <p>- 3.6501 * RMSE.ZeroR</p> <p>- 0.3997 * ST2.freY1</p> <p>- 0.3749 * ST2.corXdeg1</p> <p>+ 0.0293 * ST2.corXfre1</p>

$- 0.7517 * ST2.corXdeg3$ $+ 0.0058 * ST2.kurtcorXYfre$ Korelasyon Katsayısı 0.9135 Ortalama mutlak hata 0.0359 Ortalama karesel hatanın karekökü 0.0526 Rölatif mutlak hata 33.4332 % 	$- 0.5897 * ST2.stdcorXfre$ $- 0.0474 * ST2.corXYfre7$ $+ 0 * ST2.stdBoluMeanX1$ $+ 4.7881$ Korelasyon Katsayısı 0.9195 Ortalama mutlak hata 0.0341 Ortalama karesel hatanın karekökü 0.0506 Rölatif mutlak hata 31.7684 % 
---	--

Görüldüğü gibi bazı algoritmaların performansları iyi tahmin edilebilirken bazılarının ki iyi değildir. Algoritmaların tahmin edilebilme performansları çıkışların aralıkları aynı olmadığından RMSE yerine, Rölatif Mutlak Hata (RAE) ve Korelasyon katsayıları kullanılarak karşılaştırılmıştır. Çizelge 9.20’de algoritmaların tüm meta özellikler ve seçilen meta özelliklerle performanslarının tahmin başarıları bu 2 kritere göre özetlenmiştir.

Çizelge 9.20 Friedman koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları

Performansı tahmin edilen algoritma	Meta Özellik sayısı	Rölatif Mutlak Hata	Korelasyon katsayıları
Meta.Bagging	286	33.8624 %	0.9064
Meta.Bagging	19	30.8655 %	0.9372
M5P	286	39.1983 %	0.9117

M5P	108	38.0076 %	0.9133
meta.AttributeSelectedClassifier	286	93.2214 %	0.3761
meta.AttributeSelectedClassifier	7	62.8812 %	0.7853
meta.RandomSubSpace	286	40.6566 %	0.8897
meta.RandomSubSpace	25	31.9506 %	0.9284
Reptree	286	33.4332 %	0.9135
Reptree	20	31.7684 %	0.9195

Meta.AttributeSelectedClassifier ve meta.RandomSubSpace haricindeki tüm algoritmaların performansı 0.9'un üzerinde bir korelasyon değeriyle tahmin edilebilmiştir.

Performansı en iyi tahmin edilebilen algoritma meta.Bagging'tir. Bu algoritma aynı zamanda 80 friedman veri kümesi üzerinde en iyi performansa sahip algoritmadır. Bu özelliği, elde edilen performans tahmin başarısının önemini arttırmaktadır.

Özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

9.4.2.7 Friedman Koleksiyonunda Performans Tahminlerle Kullanılan Meta Özelliklerin Analizleri

Performans tahmininde M5p'nin kurallarında en çok kullanılan meta özellikler ve buna bağlı olarak meta özellik türleri araştırıldığında 286 özellikten 57'sinin en az bir kere kurallarda yer aldığı görülmüştür. Kurallarda 1'den fazla yer alan meta özellik sayısı ise 26'dır. Çizelge 9.21'de bu 26 meta özellik kurallarda yer alma sayılarına göre büyükten küçüğe doğru sıralanmıştır.

Çizelge 9.21 Friedman koleksiyonunda performans tahmininde kullanılan meta özelliklerden kurallarda en çok yer alanları

CLUS.EM	REGT.m5p_ytek_ozsayi_ozsayi_orani
CLUS.kmean	REGT.rep_nsayi_cfsozsayi_orani
STA.orneksayi	REGT.rep_nsayi_orsayi_orani
ST2.corXdeg1	ST2.corXfre1
colli	ST2.corXYfre7
RMSE.Decstump	ST2.freY1
ST2.corXdeg2	ST2.freY4
REGT.m5r_ksayi_orsayi_orani	ST2.skewfreX1
REGT.m5r_ksayi_ozsayi_orani	ST2.skewfreY
ST2.freY2	ST2.stdcorXdeg
REGT.m5p_yaprak_sayisi	ST2.stdcorXfre

REGT.m5p_yapraklardaki_tekil_oz_sayisi	ST2.stdfreX5
REGT.m5p_ysayi_ozsayi_orani	STA.cfs_sayi

RMSE meta özellikleri haricindeki meta özellik gruplarının yoğun bir şekilde seçildiği görülmektedir. RMSE grubundan seçilen tek meta özellik Decstump'tır.

En fazla seçilen meta özellikler Kümeleme meta özellik grubuna aittir.

Özelliklerin birbirleriyle ilişkilerinin derecelerini yansıtan colli ve ST2.corXdeg'de sıklıkla seçilmiştir.

9.4.3 İlaç Veri Kümeleri Koleksiyonunda Meta Regresyon

Bu tezin ana konularından biri olan ilaç tasarımı üretilen veri kümeleri üzerinde de meta regresyon çalışmaları yapılmış ve algoritmaların bu tür veri kümeleri üzerlerindeki performansları tahmin edilmeye çalışılmıştır.

9.4.3.1 İlaç Veri Kümeleri

Literatürde farklı çalışmalarda kullanılan 41 ilaç veri kümesi meta regresyon denemeleri gerçekleştirilmiştir. Veri kümeleri Çizelge 9.22'de verilmiştir. Özellik sayısı 1143 olan veri kümelerinde yer alan moleküllerin özellikleri Adriana.Code yazılımıyla elde edilmiştir. Bu veri kümelerinin haricindekiler ise çalışmalardan hazır olarak alınmıştır.

Çizelge 9.22 Kullanılan ilaç veri kümeleri ve referansları

Veri kümesi ismi	Özellik sayısı	Örnek sayısı	Referans
carbolenes	1143	37	B. D. Silverman and Daniel. E. Platt, J. Med. Chem. 1996, 39, 2129-2140
mtp2	1143	274	Bergstrom, C. A. S.; Norinder, U.; Luthman, K.; Artursson, P. Molecular Descriptors Influencing Melting Point and Their Role in Classification of Solid Drugs. J. Chem. Inf. Comput. Sci.; (Article); 2003; 43(4); 1177-1185
chang	1143	34	David E Patterson, Richard D Cramer, Allan M Ferguson,
cristalli	1143	32	Robert D Clark, Laurence W Weinberger.
depreux	1143	26	Neighbourhood Behaviour:
doherty	1143	6	A Useful Concept for Validation of "Molecular Diversity" Descriptors.
garrat2	1143	14	J. Med. Chem. 1996 (39) 3049 - 3059.

garrat	1143	10	
heyl	1143	11	
krystek	1143	30	
lewis	1143	7	
penning	1143	13	
rosowsky	1143	10	
siddiqi	1143	10	
stevenson	1143	5	
strupcz	1143	34	
svensson	1143	13	
thompson	1143	8	
tsutumi	1143	13	
uejling	1143	9	
yokoyama1	1143	13	
yokoyama2	1143	12	
mtp	203	4450	Karthikeyan, M.; Glen, R.C.; Bender, A. General melting point prediction based on a diverse compound dataset and artificial neural networks. J. Chem. Inf. Model.; 2005; 45(3); 581-590
benzo32	33	195	Harrison,P.W. and Barlin,G.B. and Davies,L.P. and Ireland,S.J. and Matyus,P. and Wong,M.G., Syntheses, pharmacological evaluation and molecular modelling of substituted 6-alkoxyimidazo[1,2-b]pyridazines as new ligands for the benzodiazepine receptor, European Journal of Medicinal Chemistry, (31), 1996, 651-662
PHENETYL1.arff	629	22	H. Kubinyi (Ed.): "QSAR: Hansch Analysis and Related Approaches", VCH, Weinheim (Ger), 1993, pp.57-68
pah	113	80	Todeschini, R.; Gramatica, P.; Marengo, E.; Provenzani, R. Weighted Holistic Invariant Molecular Descriptors.

			Part 2. Theory Development and Applications on Modeling Physico-Chemical Properties of PolyAromatic Hydrocarbons (PAH). Chemom. Intell. Lab. Syst. 1995, 27, 221-229.
pdgfr.arff	321	79	R. Guha and P. Jurs. The Development of Linear, Ensemble and Non-linear Models for the Prediction and Interpretation of the Biological Activity of a Set of PDGFR Inhibitors. J. Chem. Inf. Comput. Sci. 2004, 44 (6), 2179-2189
Phen.arff	111	22	Cammarata, A. Interrelationship of the Regression Models Used for Structure-Activity Analyses. J. Med. Chem. 1972, 15, 573-577
topo_2_1	267	8885	Jun Feng et al, Predictive Toxicology: Benchmarking Molecular Descriptors and Statistical Methods, J. Chem. Inf Comput. Sci., 2003 (43) 1463-1470
yprop_4_1	252	8885	
qsabr1.arff	10	15	Damborsky, J., Schultz, T.W., Comparison of the QSAR models for toxicity and biodegradability of anilines and phenols, Chemosphere 34: 429-446, 1997
qsabr2.arff	10	15	
qsartox.arff	24	16	Blaha, L., Damborsky, J., Nemec, M., QSAR for acute toxicity of saturated and unsaturated halogenated aliphatic compounds, Chemosphere 36: 1345-1365, 1998
qsbr_rw1.arff	51	14	Damborsky, J. et al., Structure-biodegradability relationships for chlorinated dibenzo-p-dioxins and dibenzofurans, In: Wittich, R.-M., Biodegradation of dioxins and furans, R.G. Landes Company, Austin, 1998
qsbr_y2.arff	10	25	Damborsky, J. et al., A mechanistic approach to deriving QSBR- A case study: dehalogenation of haloaliphatic compounds, In: Peijnenburg, W.J.G.M., Damborsky, J., Biodegradability Prediction, Kluwer Academic Publishers
qsbralks.arff	22	13	Damborsky, J. et al., Mechanism-based Quantitative Structure- Biodegradability Relationships for hydrolytic dehalogenation of chloro- and bromo-alkenes, Quantitative Structure-Activity Relationships 17: 450-458, 1998

qsfrdhla.arff	34	16	Damborsky, J., Quantitative structure-function relationships of the single-point mutants of haloalkane dehalogenase: A multivariate approach, Qunatitative Structure-Activity Relationships 16: 126-135, 1997
qsfsr1.arff	10	20	Damborsky, J., Quantitative structure-function and structure-stability relationships of purposely modified proteins, Protein Engineering 11: 21-30, 1998
qsfsr2.arff	10	19	
qsprcmpx.arff	40	22	Cajan, M. et al., Stability of Aromatic Amides with Bromide Anion: Quantitative Structure-Property Relationships, Journal of Chemical Information and Computer Sciences, in press, 2000
selwood.arff	54	31	Selwood, D. L.; Livingstone, D. J.; Comley, J. C.; O'Dowd, A. B.; Hudson, A. T.; Jackson, P.; Jandu, K. S.; Rose, V. S.; Stables, J. N. Structure-Activity Relationships of Antifilarial Antimycin Analogues: A Multivariate Pattern Recognition Study J. Med. Chem., 1990, 33, 136-142

9.4.3.2 İlaç Veri Koleksiyonunda Kullanılan Ek Meta Özellikler

Ortak meta özelliklere ek olarak ilaç koleksiyonunda kullanılan ek meta özellikler Çizelge 9.23'te verilmiştir.

Çizelge 9.23 İlaç veri koleksiyonunda kullanılan ek meta özellikler

Meta özellik	Açıklaması
Tahmin.tur	İlaç veri kümesinde tahmin edilmesi istenen özelliğin türü a=biyolojik aktivite m=kaynama derecesi d=diğer
RMSE.PLS3	Algoritmaların veri kümesi üzerindeki RMSE değerleri
RMSE.PLS4	
RMSE.PLS5	

RMSE.REPTree	
--------------	--

9.4.3.3 İlaç Veri Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri

41 ilaç veri kümesinde algoritmaların performansları incelenmiş ve en başarılı algoritmalar belirlenmiştir. Çizelge 9.24'te bu algoritmalar verilmiştir.

Çizelge 9.24 41 İlaç veri koleksiyonunda minimum RMSE değeri içeren algoritmalar

Algoritma adı	41 veri kümesinin kaçında en başarılı algoritma olduğu
Kstar	12
PLS5	5
SLR	5
M5P	4
PLS2	3
Decstump	2
IBK	2
PLS1	2
PLS4	2
M5R	1
PLS3	1
REPTree	1
SMO	1

Çizelge 9.25'te tüm algoritmaların tüm veri kümeleri üzerindeki RMSE'lerinin minimum, maksimum, ortalama ve standart sapma değerleri verilmiştir.

Çizelge 9.25 İlaç veri koleksiyonunda algoritmaların 41 veri kümesi üzerindeki performansları

Algoritma ismi	Minimum RMSE	Maksimum RMSE	Ortalama RMSE	Medyan RMSE	Standart sapma RMSE
ConjunctiveRule	0.0293	0.429	0.245	0.243	0.084
Decstump	0.0293	0.485	0.256	0.252	0.099
IBK	0.0368	0.788	0.244	0.229	0.128

Kstar	0.0202	0.447	0.222	0.216	0.096
LinearRegression	0.0291	64182.84	1922.91	0.938	1006.39
LWL	0.0292	0.592	0.2611	0.25	0.103
M5P	0.0290	0.64	0.239	0.23	0.11
M5R	0.0289	0.487	0.247	0.251	0.097
PLS1	0.0292	0.545	0.223	0.213	0.091
PLS2	0.0289	0.665	0.226	0.204	0.111
PLS3	0.0291	0.699	0.239	0.206	0.13
PLS4	0.0289	0.679	0.231	0.207	0.128
PLS5	0.0290	0.699	0.245	0.212	0.144
RBF	0.0294	0.503	0.257	0.261	0.088
REPTree	0.0295	0.493	0.256	0.244	0.096
SLR	0.0292	1.841	0.313	0.234	0.305
SMO	0.0298	348.1	24.21	1.346	0.611
SVM	0.0298	347.6	24.17	1.349	0.61

Çizelge 9.25 incelendiğinde Lineer Regresyon, SVM ve SMO algoritmalarının bazı veri kümelerinde yakınsamadığı görülmektedir.

En başarılı 5 algoritma sırasıyla

Medyan RMSE'lere göre :

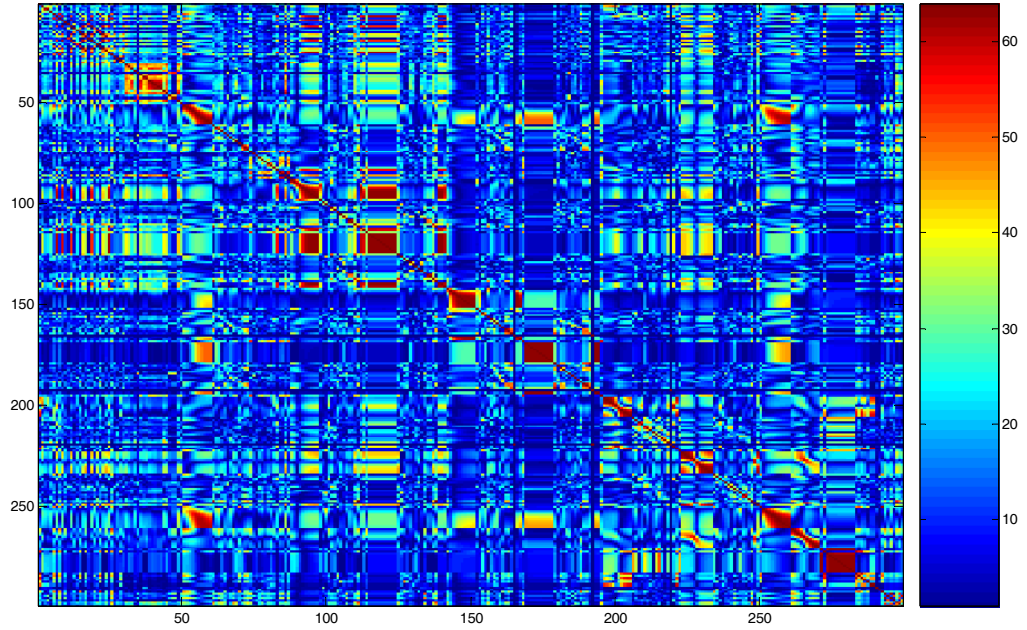
PLS2 > PLS3 > PLS4 > PLS5 > PLS1 > Kstar > IBK > M5P > SLR

Ortalama RMSE'lere göre:

Kstar > PLS1 > PLS2 > PLS4 > PLS3 > M5P > IBK > PLS5 > ConjunctiveRule

9.4.3.4 İlaç Veri Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri

Şekil 9.4'te 299 (300-1 nominal özellik) meta özelliğin birbirleriyle korelasyonu gösterilmiştir. Negatif korelasyonlarda anlamlı olacağı için korelasyon matrisi görselleştirilirken mutlak değeri alınmıştır.



Şekil 9.4 İlaç veri koleksiyonunda 299 meta özelliğin korelasyon matrisi

Renkler maviden kahverengiye doğru ilerledikçe iki özellik arasındaki korelasyonun yüksekliği artmaktadır.

Şekil incelendiğinde, Friedman koleksiyonunun korelasyon matrisine (Şekil 9.1) göre meta özellikler arası korelasyonun daha az olduğu görülmektedir.

Korelasyon katsayısının mutlak değerinin 0.8'den büyük olduğu 857 özellik ikilisi bulunmuştur. Çizelge 9.26'da bu 857 ikilinin meta özellik gruplarındaki dağılımları verilmiştir. Çizelgenin ilk satırında meta özellik gruplarında yer alan meta özellik sayıları verilmiştir.

Çizelge 9.26 İlaç veri koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı

	5	19	18	220	15	22
	CLUS	RMSE	REGT	ST2	STA	PCA
CLUS	1					
RMSE		36		4		
REGT			11	39	8	2
ST2				562	78	35
STA					10	5
PCA						62

Meta özellik ikililerinden önemli görülenler Çizelge 9.27’de listelenmiştir.

Çizelge 9.27 İlaç veri koleksiyonunda yüksek korelasyonlu meta özelliklerden seçmeler

Meta özellik 1	Meta özellik 1	Korelasyon katsayısı
ST2.bigcorXYpro	PCA.expfre10	0.99999
STA.iqr_extrem	STA.orneksayi	0.98979
STA.orneksayi	ST2.meankurtdegX	0.97265
STA.iqr_extrem	ST2.meankurtdegX	0.96827
STA.iqr_extrem	ST2.stdkurtdegX	0.96826
STA.iqr_extrem	ST2.kurtdegX1	0.96811
STA.iqr_outlier	REGT.m5r_kurallardaki_tekil_oz_sayisi	0.96678
ST2.stdfreX1	ST2.meanskewfreX	-0.96516
ST2.stdfreX1	ST2.meankurtfreX	-0.96516
ST2.stdfreX1	ST2.meancorXYfre	-0.96516
STA.iqr_outlier	STA.orneksayi	0.96259
STA.iqr_outlier	REGT.m5p_yapraklardaki_tekil_oz_sayisi	0.96115
STA.cfs_oran	PCA.expfre1	-0.95697
STA.iqr_outlier	ST2.skewdegX10	0.93043
RMSE.LWL	RMSE.PLS1	0.92724
RMSE.REPTree	ST2.stdY	0.9245
STA.iqr_extrem	ST2.kurtY	0.92188
STA.orneksayi	ST2.kurtY	0.9209
STA.iqr_extrem	ST2.stdskewdegX	0.91771
STA.iqr_o_oran	STA.ozelliksayi	0.91664
STA.iqr_outlier	ST2.meankurtdegX	0.91641
STA.cfs_oran	ST2.corXdeg10	-0.91552
REGT.m5r_ksayi_ozsayi_orani	ST2.bigcorXYpro	0.90807

REGT.m5r_ksayi_ozsayi_orani	PCA.expfre10	0.90806
ST2.skewdegX10	ST2.kurtY	0.90596
REGT.m5p_yapraklardaki_tekil_oz_sayisi	ST2.skewdegX10	0.89975
STA.cfs_oran	PCA.x95	0.89876
REGT.m5r_kurallardaki_tekil_oz_sayisi	ST2.skewdegX10	0.89684
RMSE.REPTree	RMSE.ZeroR	0.8936
STA.cfs_oran	ST2.stdcorXdeg	-0.88866
REGT.m5r_kurallardaki_tekil_oz_sayisi	STA.orneksayi	0.88648
REGT.m5p_yapraklardaki_tekil_oz_sayisi	STA.orneksayi	0.88002
STA.iqr_extrem	ST2.meanskewdegX	0.87445
ST2.meankurtfreX	STA.pertumsayi	-0.87344
ST2.meanskewfreX	STA.pertumsayi	-0.87344
RMSE.LinearRegression	RMSE.SVM	0.87095
RMSE.LinearRegression	RMSE.SMO	0.87056
ST2.freY1	ST2.skewY	0.86958
ST2.freY1	ST2.mom3Y	0.85903
STA.iqr_outlier	REGT.m5p_ytek_ozsayi_cfsozsayi_orani	0.85417
STA.iqr_outlier	REGT.m5r_ksayitekozsayi_cfsozsayi_orani	0.85295
RMSE.RBF	ST2.stdY	0.84602
REGT.m5p_yapraklardaki_tekil_oz_sayisi	ST2.meankurtdegX	0.84581
REGT.m5r_kurallardaki_tekil_oz_sayisi	ST2.meankurtdegX	0.84576
RMSE.RBF	RMSE.ZeroR	0.83551
STA.iqr_outlier	ST2.kurtY	0.83474
ST2.freY1	ST2.kurtfreY	0.83011
STA.iqr_extrem	REGT.m5r_kurallardaki_tekil_oz_sayisi	0.82919
STA.iqr_e_oran	ST2.stddegX1	-0.82357
STA.iqr_extrem	REGT.m5p_yapraklardaki_tekil_oz_sayisi	0.82332
ST2.freY1	ST2.mom4Y	0.82326
RMSE.ConjunctiveRule	ST2.stdY	0.80895
ST2.stdfreX1	STA.pertumsayi	0.80768
ST2.bigcorXYpro	PCA.x95	0.80602
RMSE.LinearRegression	RMSE.SLR	0.8035

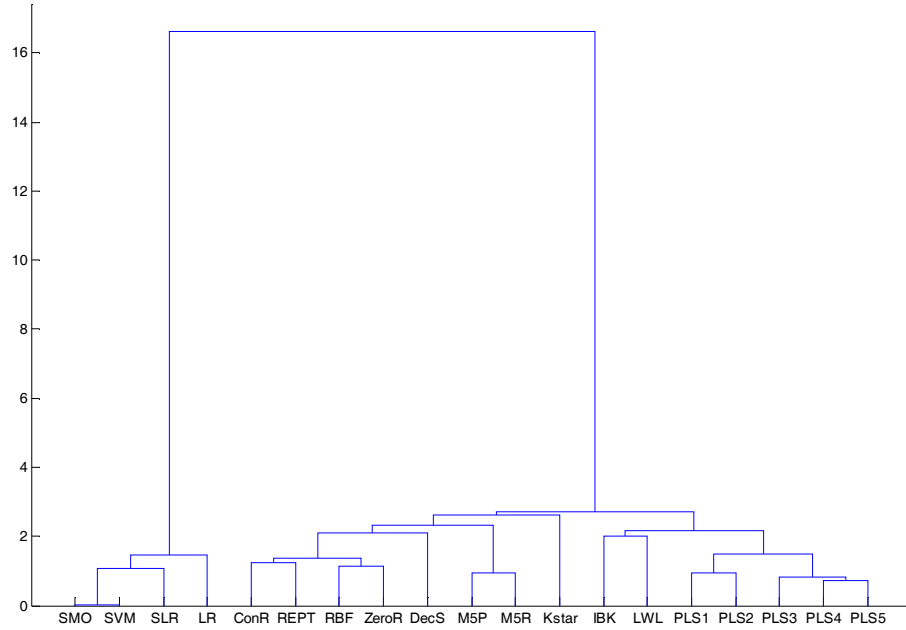
Çizelge 9.27 incelendiğinde aşağıdaki sonuçlara ulaşılmıştır.

- Girişlerin çıkışlarla korelasyonu ne kadar büyükse, PCA'in verinin varyansını açıklamak için ihtiyaç duyduğu bileşen sayısı o kadar azdır.
- Aykırı örnek sayısı ile örnek sayısı ve verinin dağınıklığı arasında doğru orantı vardır.
- Örnek sayısı ne kadar fazla ise verinin dağınıklığı o kadar fazladır.

- Verinin dağınıklığı ile karar ağaçlarının yapraklarında kullanılan özellik sayısı arasında doğru orantı vardır. Diğer bir ifadeyle veri ne kadar dağınıksa karar ağaçlarında o kadar kompleks kurallar kullanılmaktadır.
- İlgili özellik sayısı oranı ile verinin varyansını açıklamak için gereken PCA bileşen sayısı arasında ters orantı vardır.
- RepTree, RBF ve ConjunctiveRule algoritmalarının hataları ile çıkışların standart sapması doğru orantılıdır. Çıkışlar ne kadar dağınıksa bu algoritmalar o kadar çok hata yapmaktadır.
- İlgili özellik sayısı oranı ile giriş özelliklerinin birbirleriyle korelasyonu arasında ters orantı vardır.
- Karar ağaçlarındaki kural başına düşen özellik sayısı oranı ile giriş çıkış korelasyonu ve verinin varyansını açıklamak için gereken PCA bileşen sayısı doğru orantılıdır.
- RBF ve RepTree algoritmalarının hataları ile rasgele hata arasında doğru orantı vardır. Veri kümesinin çıkışlarının tahmini ne kadar zor ise bu algoritmalar o kadar çok hata yapmaktadır.
- Karar ağaçlarındaki kuralların kompleksliği ile örnek sayısı arasında doğru orantı vardır.
- Lineer Regresyon, Basit Lineer Regresyon ve Destek Vektör Makinelerinin performansları doğru orantılıdır. Çünkü üçü de veri için lineer modeller varsaymaktadırlar.
- Karar ağaçlarındaki kuralların kompleksliği ile verinin dağınıklığı doğru orantılıdır. Veri ne kadar dağınıksa, karar ağaçları o kadar kompleks kurallar üretmektedir.
- Giriş ve çıkış korelasyonu ile giriş verilerinin varyanslarının %95'ini içeren PCA bileşen sayısı doğru orantılıdır.

9.4.3.5 İlaç Veri Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümellemeleri

Algoritmaların veri kümeleri üzerlerindeki performanslara göre hiyerarşik kümelemesi yapıldığında Şekil 9.5 elde edilmiştir.

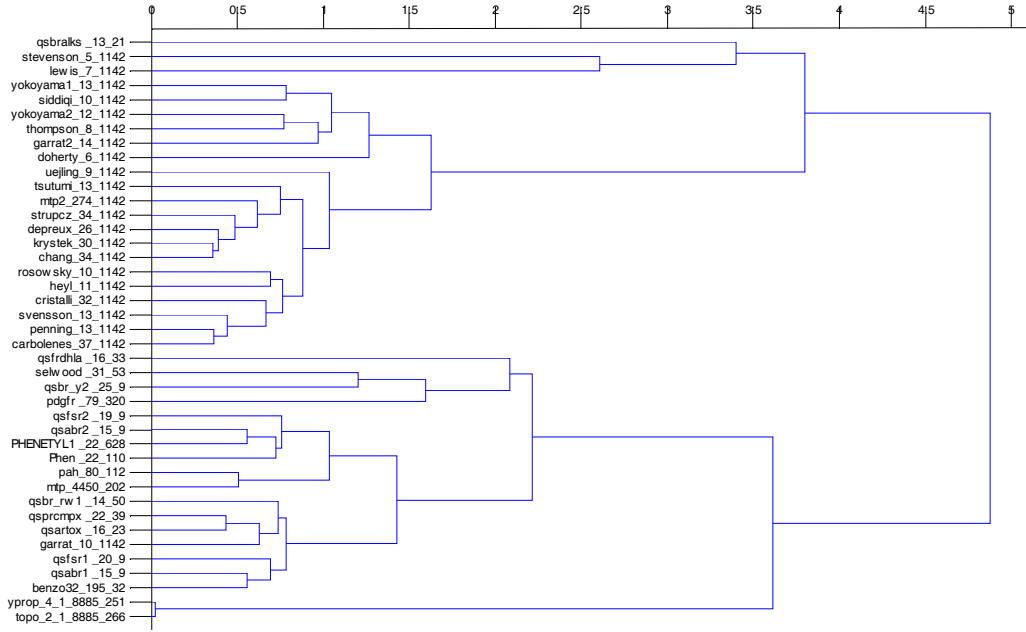


Şekil 9.5 İlaç veri koleksiyonunda algoritmaların RMSE değerlerine göre (41 boyutlu) hiyerarşik kümelermeleri

Şekil 9.5 incelendiğinde:

- Lineer karakteristiğe sahip algoritmaların bir kümede toplandığı
- PLS'lerin bir kümede toplandığı
- Algoritmaların birbirine çok uzak iki kümede toplandıkları görülmektedir.

Veri kümelerinin aynı şekilde kümelemesi yapıldığında Şekil 9.6 elde edilmiştir. Veri kümesi isimleri şeklin sol tarafında yer almaktadır ve “veri kümesi ismi _ örnek sayısı _ özellik sayısı” şeklinde kodlanmıştır.



Şekil 9.6 İlaç veri koleksiyonunda veri kümelerinin RMSE değerlerine göre hiyerarşik kümeleme sonucu

Şekil 9.6 incelendiğinde özellik sayısı 1142 olan veri kümelerinin bir grupta toplandığı görülmektedir.

9.4.3.6 İlaç Veri Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri

En başarılı 9 algortmadan ortalama sıralamasının ve medyan sıralamasının kesişiminde yer alan 4'ünün (PLS1, KStar, M5P, IBK) RMSE hataları tahmin edilmeye çalışılmıştır.

PLS ailesinin birbirleriyle korelasyonları çok yüksek olduğundan sadece birinin tahmini yapılmış ve diğerlerinin sonuçlarının da ona (PLS1) çok benzeyeceği varsayılmıştır.

Diğer algoritmaların performanslarının tahmininde kullanılacak algoritmalar:

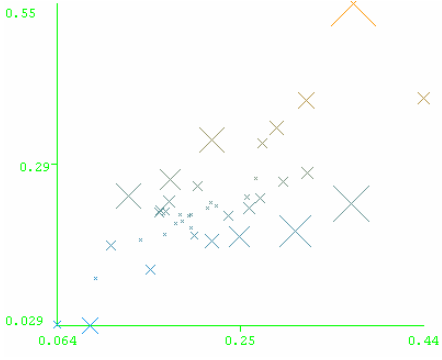
- Decision Stump
- Linear Regression
- Zero0
- RBF

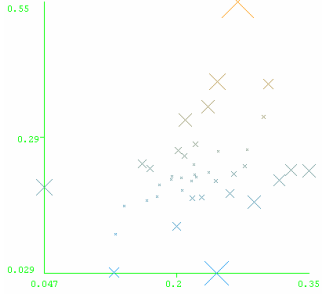
- RepTree

olarak seçilmiştir.

Çizelge 9.28, 9.29, 9.30 ve 9.31’de algoritmaların performanslarının tahmin denemeleri verilmiştir.

Çizelge 9.28 İlaç veri koleksiyonunda PLS1 algoritmasının performans tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (8 özellikle)
Performans uzayı 3 alt uzaya bölmüş ve her birinde	RMSE.PLS1 =
STA.cfs_oran	0.5874 * RMSE.REPTree
STA.cfs_sayi	+ 0.5583 * ST2.kurtfreX5
REGT.m5p_yaprak_sayisi	+ 0.8001 * ST2.mom4Y
RMSE.REPTree	+ 0.0303
RMSE. LinearRegression	
ST2.kurtfreX9	Korelasyon Katsayısı 0.7516
ST2.bigcorXpro	Ortalama mutlak hata 0.0447
özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.	Ortalama karesel hatanın karekökü 0.0598
	Rölatif mutlak hata 73.8913 %
Korelasyon Katsayısı 0.3791	
Ortalama mutlak hata 0.0625	
Ortalama karesel hatanın karekökü 0.0864	
Relative absolute error 103.4757 %	
	
	Özellik seçimi performansı iyileştirmiştir ancak sonuç

 <p>Performans tahmini tatmin edici değildir.</p>	<p>yinede tatmin edici değildir.</p>
--	--------------------------------------

Çizelge 9.29 İlaç veri koleksiyonunda Kstar algoritmasının performans tahminleri

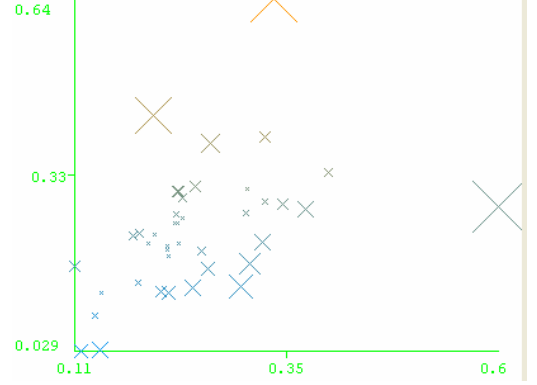
Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (8 özellikle)
<p>Performans uzayı şeklinde 9 alt uzaya bölmüş ve her birinde</p> <p>CLUS.xmean</p> <p>REGT.m5p_ytek_ozsayi_ozsayi_orani</p> <p>RMSE.Decstump</p> <p>RMSE.ZeroR</p> <p>ST2.corXfre4</p> <p>ST2.kurtY</p> <p>ST2.skewfreX10</p> <p>ST2.stddegX7</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.2913</p> <p>Ortalama mutlak hata 0.0841</p> <p>Ortalama karesel hatanın karekökü 0.1896</p>	<p>RMSE.Kstar =</p> <p>$0.7903 * \text{RMSE.ZeroR}$</p> <p>$- 0.339 * \text{ST2.stdfreY}$</p> <p>$+ 0.5348 * \text{ST2.kurtfreX6}$</p> <p>$- 0.0195 * \text{ST2.skewcorXfre}$</p> <p>$+ 0.0765$</p> <p>Korelasyon Katsayısı 0.808</p> <p>Ortalama mutlak hata 0.0438</p> <p>Ortalama karesel hatanın karekökü 0.0563</p> <p>Rölatif mutlak hata 62.0451 %</p>

<p>Rölatif mutlak hata 119.236 %</p> <p>Bir veri kümesi üzerideki tahminin toplam tahmini çok bozduğu görülmektedir.</p>	<p>Özellik seçimi performansı iyileştirmiştir ancak sonuç yine de tatmin edici değildir.</p>
--	--

Çizelge 9.30 İlaç veri koleksiyonunda M5P algoritmasının performans tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (12 özellikle)
<p>RMSE.M5P =</p> $0.066 * STA.iqr_o_oran$ $+ 0.378 * RMSE.Decstump$ $+ 0.5008 * RMSE.REPTree$ $- 0.0227$ <p>Korelasyon Katsayısı -0.0219</p> <p>Ortalama mutlak hata 0.1614</p> <p>Ortalama karesel hatanın karekökü 0.4475</p> <p>Rölatif mutlak hata 202.1148 %</p> <p>2 veri kümesi üzerideki tahminin toplam tahmini çok bozduğu görülmektedir. Performans tahminindeki hata çok</p>	<p>Performans uzayı 2 alt uzaya bölünmüş ve her birinde</p> <p>ST2.corXYBolustdXY7</p> <p>RMSE.Decstump</p> <p>ST2.kurtfreX6</p> <p>REGT.m5r_ksayi_orsayi_orani</p> <p>ST2.mom4freX5</p> <p>RMSE.REPTree</p> <p>ST2.skewfreX4</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.4663</p> <p>Ortalama mutlak hata 0.0765</p> <p>Ortalama karesel hatanın karekökü 0.1039</p> <p>Rölatif mutlak hata 95.8542 %</p>

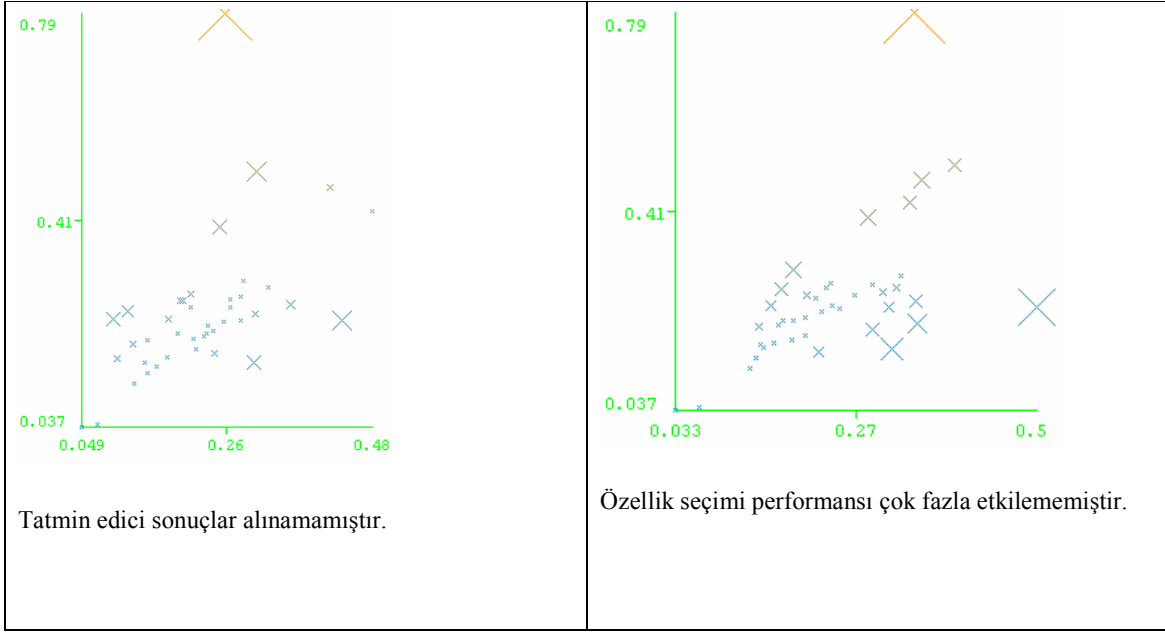
yüksektir.



Özellik seçimi performansı iyileştirmiştir ancak sonuç yine de tatmin edici değildir.

Çizelge 9.31 İlaç veri koleksiyonunda IBK algoritmasının performans tahminleri

Tüm özelliklerle (286 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (6 özellikle)
<p>RMSE.IBK =</p> <p>$0.9763 * \text{RMSE.Decstump}$</p> <p>$+ 0.4941 * \text{ST2.corXYfre3}$</p> <p>$- 0.0332$</p> <p>Korelasyon Katsayısı 0.5392</p> <p>Ortalama mutlak hata 0.0669</p> <p>Ortalama karesel hatanın karekökü 0.1105</p> <p>Rölatif mutlak hata 82.0893 %</p>	<p>RMSE.IBK =</p> <p>$0.8305 * \text{RMSE.Decstump}$</p> <p>$+ 0.9729 * \text{ST2.stdfreX9}$</p> <p>$+ 0.0007$</p> <p>Korelasyon Katsayısı 0.585</p> <p>Ortalama mutlak hata 0.0659</p> <p>Ortalama karesel hatanın karekökü 0.1044</p> <p>Rölatif mutlak hata 80.955 %</p>



Görüldüğü gibi algoritmaların performansları yapay veri kümesi koleksiyonundaki (friedman) gibi iyi bir şekilde tahmin edilememiştir.

Algoritmaların tahmin edilebilme performansları çıkışların range'leri aynı olmadığından RMSE yerine; Rölative Mutlak Hata (RAE) ve Korelasyon katsayıları kullanılarak karşılaştırılmıştır. Çizelge 9.32'de algoritmaların tüm meta özellikler ve seçilen meta özelliklerle performanslarının tahmin başarıları bu 2 kritere göre özetlenmiştir.

Çizelge 9.32 İlaç koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları

Performansı tahmin edilen algoritma	Meta Özellik sayısı	Rölative Mutlak Hata	Korelasyon katsayıları
PLS1	286	103.4757 %	0.3791
PLS1	8	73.8913 %	0.7516
Kstar	286	119.236 %	0.2913
Kstar	8	62.0451 %	0.808
M5P	286	202.1148 %	-0.0219
M5P	11	95.8542 %	0.4663
IBK	286	82.0893 %	0.5392
IBK	6	80.955 %	0.585

Kstar haricindeki tüm hiçbir algoritmanın performansı 0.8'un üzerinde bir korelasyon değeriyle tahmin edilememiştir. Bunun sebebi olarak meta özelliklerle algoritmaların

tahminleri arasında yüksek bir korelasyon olmaması söylenebilir.

Performansı en iyi tahmin edilebilen algoritma Kstar'dır. Bu algoritma aynı zamanda 41 ilaç veri kümesi üzerinde Ortalama RMSE'lere göre en iyi performansa sahip algoritmadır. Bu özelliği, elde edilen performans tahmin başarısının önemini arttırmaktadır.

Özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

9.4.3.7 İlaç Koleksiyonunda Performans Tahminlerinde Kullanılan Meta Özelliklerin Analizleri

Performans tahmininde M5p'nin kurallarında en çok kullanılan meta özellikler ve buna bağlı olarak meta özellik türleri araştırıldığında 300 özellikten 27'sinin en az bir kere kurallarda yer aldığı görülmüştür. Kurallarda 1'den fazla yer alan meta özellik sayısı ise 15'dir. Çizelge 9.33'te, bu meta özellikler ve kurallarda yer alma sayıları verilmiştir.

Çizelge 9.33 İlaç koleksiyonunda performans tahminlerinde en çok kullanılan meta özellikler

Meta Özellik	M5P kurallarında yer alma frekansı
RMSE.Decstump	12
RMSE.ZeroR	10
ST2.kurtY	9
RMSE.REPTree	7
ST2.corXfre4	3
ST2.kurtfreX9	3
REGT.m5p_ytek_ozsayi_ozsayi_orani	2
REGT.m5r_ksayi_orsayi_orani	2
ST2.bigcorXpro	2
ST2.corXYBolustdXY7	2
ST2.kurtfreX6	2
ST2.mom4freX5	2
ST2.skewfreX10	2
ST2.stddegX7	2
STA.cfs_sayi	2

9.4.4 UCI Veri Kümeleri Koleksiyonunda Meta Regresyon

Literatürde en sık kullanılan veri kümelerini içeren UCI koleksiyonundaki regresyon veri kümeleri üzerinde de meta regresyon çalışmaları yapılmış ve algoritmaların bu tür veri

kümeleri üzerlerindeki performansları tahmin edilmeye çalışılmıştır.

9.4.4.1 UCI Veri Kümeleri

Literatürde UCI veri koleksiyonu olarak bilinen koleksiyondan 60 veri kümesi üzerinde meta regresyon denemeleri gerçekleştirilmiştir. Kullanılan regresyon veri kümeleri ve özellikleri Çizelge 9.34’te verilmiştir.

Çizelge 9.34 Kullanılan UCI veri kümeleri ve özellikleri

Veri Kümesi İsmi	Örnek Sayısı	Özellik Sayısı
2dplanes	40768	10
abalone	4177	10
aileron	13750	40
auto93	93	61
auto_price	159	21
autoHorse	203	65
autoMpg	398	25
bank32nh	8192	32
bank8FM	8192	8
basketball	96	4
bodyfat	252	14
bolts	40	7
breastTumor	286	34
cal_housing	20640	8
cholesterol	303	22
cloud	108	9
cpu	209	36
cpu_act	8192	21
cpu_small	8192	12
delta_aileron	7129	5
delta_elevators	9517	6
detroit	13	13
diabetes_numeric	43	2
echoMonths	130	9
elevators	16599	18
elusage	55	13
fishcatch	158	13
fried	40768	10
fruitfly	125	8
gascons	27	4

house_16H	22784	16
house_8L	22784	8
housing	506	13
hungarian	294	22
kin8nm	8192	8
longley	16	6
lowbwt	189	19
machine_cpu	209	6
mbagrade	61	2
meta	528	65
mv	40768	12
pbc	418	23
pharynx	195	213
pol	15000	48
pollution	60	15
puma32H	8192	32
puma8NH	8192	8
pwLinear	200	10
pyrim	74	27
quake	2178	3
schlvote	37	5
sensory	576	32
servo	167	19
sleep	58	7
stock	950	9
strike	625	23
triazines	186	60
veteran	137	10
vineyard	52	3
wisconsin	194	32

9.4.4.2 UCI Koleksiyonunda Kullanılan Ek Meta Özellikler

Ortak meta özelliklere ek olarak UCI koleksiyonunda kullanılan ek meta özellikler Çizelge 9.35’te verilmiştir.

Çizelge 9.35 UCI veri koleksiyonunda kullanılan ek meta özellikler

Meta Özellik İsmi	Açıklaması
RMSE.isotonicreg	Algoritmaların veri kümesi üzerindeki RMSE değerleri
RMSE.leastmedsq	
RMSE.MLP	

9.4.4.3 UCI Koleksiyonunda Algoritmaların Veri Kümesindeki Performanslarının İncelemeleri

60 UCI veri kümesinde algoritmaların performansları incelenmiş ve en başarılı algoritmalar belirlenmiştir. Çizelge 9.36’da bu algoritmalar verilmiştir.

Çizelge 9.36 60 UCI veri kümesinde minimum RMSE değeri içeren algoritmalar

Algoritma adı	60 veri kümesinin kaçında en başarılı algoritma olduğu
Conjunctive rules	1
Decstump	0
IBK	1
Kstar	4
Lineer Regresyon	4
LWL	2
M5P	21
M5R	3
PLS1	4
PLS2	2
RBF	0
SLR	0
SMO	6
SVM	1
Zero Rule	2
isotonic regression	3
Leastmedsq	1
MLP	5

Çizelge 9.37’de tüm algoritmaların tüm veri kümeleri üzerindeki RMSE’lerinin minimum,

maksimum, ortalama ve standart sapma deęerleri verilmiřtir.

Çizelge 9.37 UCI veri koleksiyonunda 18 algoritmanın 60 veri kümesi üzerindeki performansları

Algoritma ismi	Minimum RMSE	Maksimum RMSE	Ortalama RMSE	Standart sapma RMSE
Conjunctive rules	0.0477	0.4016	0.1548	0.0668
Decstump	0.0482	0.3877	0.1547	0.0644
IBK	0.0251	0.4761	0.1477	0.0877
Kstar	0.0222	0.4351	0.1345	0.0797
Lineer Regresyon	0.0248	0.8697	0.1448	0.1275
LWL	0.0476	0.3804	0.1435	0.0606
M5P	0.0031	0.3687	0.1062	0.0692
M5R	0.0007	0.3596	0.1087	0.0691
PLS1	0.0430	0.3549	0.1427	0.0600
PLS2	0.0418	0.3478	0.1308	0.0617
RBF	0.0494	0.4398	0.1730	0.0676
SLR	0.0218	0.3931	0.1440	0.0683
SMO	0.0204	2.7252	0.1656	0.3415
SVM	0.0204	2.7246	0.1655	0.3414
Zero Rule	0.0464	0.4821	0.1948	0.0775
isotonic regression	0.0229	0.3875	0.1383	0.0700
Leastmedsq	0.0211	0.5960	0.1567	0.0987
MLP	0.0043	1.0960	0.1585	0.1679

En başarılı 6 algoritma sırasıyla

Ortalama RMSE lere göre:

M5P > M5R > PLS2 > Kstar > Isotonic Reg > PLS1

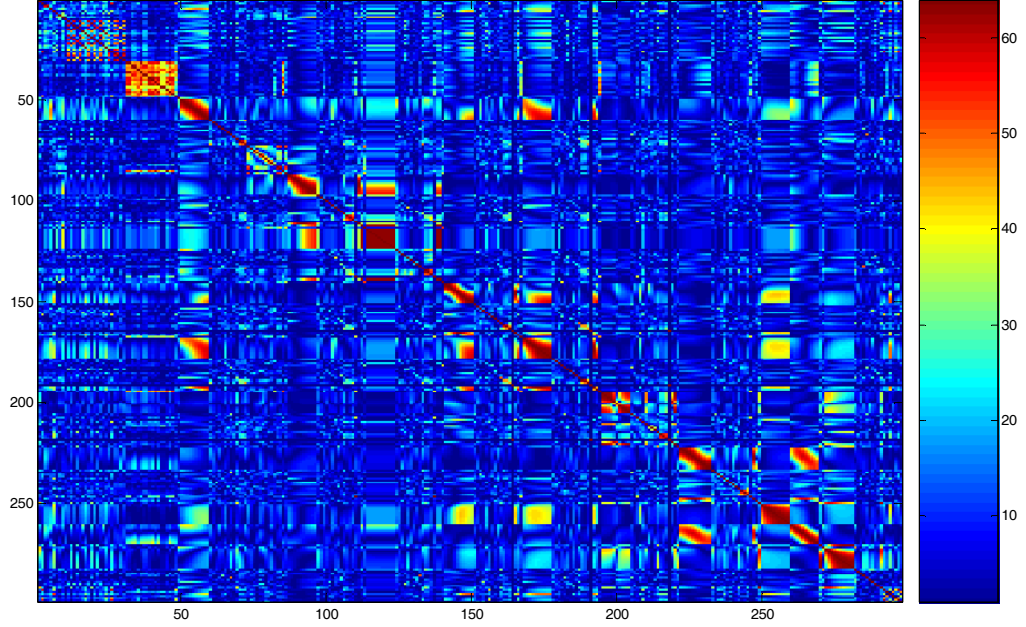
Standart sapması en büyük olan 4 algoritma sırasıyla

SMO > SVM > MLP > Lineer Reg

9.4.4.4 UCI Koleksiyonunda Meta Özelliklerin Birbirleriyle Olan Korelasyon Analizleri

Şekil 9.7’de 298 meta özelliğın birbirleriyle korelasyonu gösterilmiştir. Negatif

korelasyonlarda anlamlı olacağı için korelasyon matrisi görselleştirilirken mutlak değeri alınmıştır.



Şekil 9.7 UCI veri koleksiyonunda meta özelliklerin korelasyon matrisi

Renkler maviden kahverengiye doğru ilerledikçe iki özellik arasındaki korelasyonun yüksekliği artmaktadır.

Şekil incelendiğinde, Friedman koleksiyonunun korelasyon matrisine (Şekil 9.1) göre meta özellikler arası korelasyonun daha az olduğu görülmektedir.

Korelasyon katsayısının mutlak değerinin 0.8'den büyük olduğu 655 özellik ikilisi bulunmuştur. Çizelge 9.38'de bu 655 ikilinin meta özellik gruplarındaki dağılımları verilmiştir. Çizelgenin ilk satırında meta özellik gruplarında yer alan meta özellik sayıları verilmiştir.

Çizelge 9.38 UCI koleksiyonunda yüksek korelasyonlu meta özellik ikililerinin meta özellik gruplarına göre dağılımı

	5	18	18	220	15	22
	CLUS	RMSE	REGT	ST2	STA	PCA
CLUS	1					
RMSE		55		8		
REGT			28		3	
ST2				502	2	6
STA					7	
PCA						43

Meta özellik ikililerinden önemli görülenler Çizelge 9.39’da listelenmiştir.

Çizelge 9.39 UCI koleksiyonunda birbiriyle korelasyonu yüksek olan meta özelliklerden seçilmiş örnekler

Meta özellik 1	Meta özellik 2	Korelasyon katsayısı
RMSE.ZeroR	ST2.stdY	0.99714
ST2.bigcorXYpro	PCA.expfre10	0.96616
STA.binsayi	STA.perbinornek	0.94867
RMSE.RBF	ST2.stdY	0.94194
STA.cfs_sayi	STA.binsayi	0.90104
STA.ozelliksayi	STA.perbinornek	0.88565
STA.orneksayi	REGT.reptree_node_sayisi	0.87564
STA.cfs_sayi	STA.perbinornek	0.87452
ST2.bigcorXYpro	PCA.expdeg1	0.85587
RMSE.ConjunctiveRule	ST2.stdY	0.84636
RMSE.PLS1	ST2.stdY	0.83432
STA.iqr_e_oran	STA.perbinsayi	0.8317
RMSE.Decstump	ST2.stdY	0.83011
RMSE.leastmedsq	ST2.stdY	0.81302
ST2.mom3degX10	STA.perbinsayi	0.81159
ST2.mom3degX9	STA.perbinsayi	0.80584

Çizelge 9.39 incelendiğinde aşağıdaki sonuçlara ulaşılmıştır.

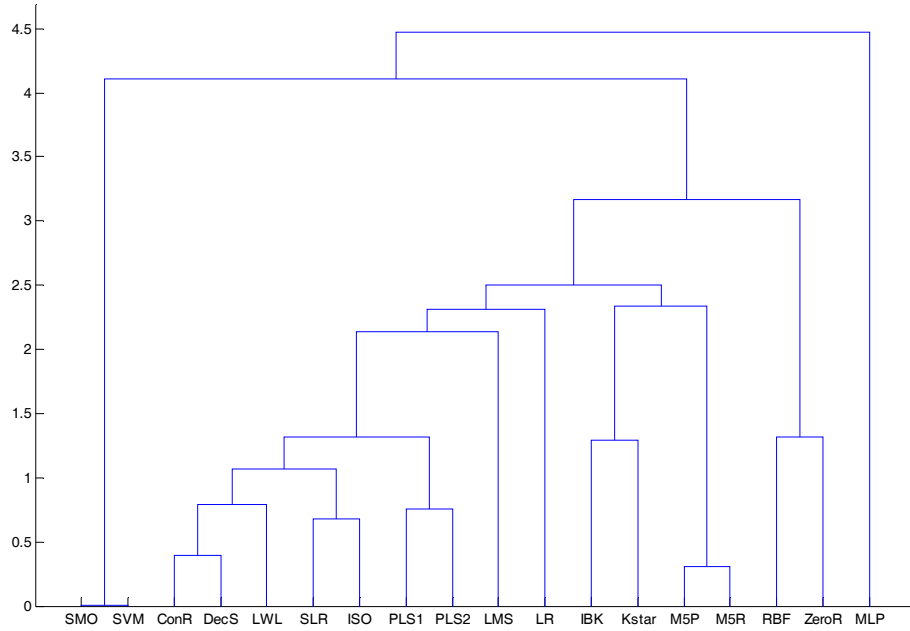
- Rasgele hata ile çıkışların standart sapması doğru orantılıdır. Çıkışlar ne kadar savrukça, rasgele başarı o kadar azdır.
- Girişlerin çıkışla korelasyonu ile ilk PCA bileşenlerinin varyansının ne kadarını

açıkladığı arasında doğru orantı vardır.

- RBF, ConjunctiveRule, PLS, DecisionStump ve LMS algoritmalarının hataları ile çıkışın standart sapması doğru orantılıdır. Çıkış ne kadar savrukça bu algoritmalar o kadar fazla hata yapmaktadır.
- İlgili özellik sayısı oranı ile, sadece iki değer içeren özellik sayısı doğru orantılıdır. Bu durumda ilgili özelliklerin genelde iki değer içeren özellikler olduğu söylenebilir.
- Karar ağaçlarının büyüklüğü ile örnek sayısı doğru orantılıdır.

9.4.4.5 UCI Koleksiyonunda Algoritmaların ve Veri Kümelerinin Performanslarına Göre Hiyerarşik Kümelermeleri

Algoritmaların veri kümeleri üzerlerindeki performanslarına göre hiyerarşik kümelemesi yapıldığında Şekil 9.8 elde edilmiştir.

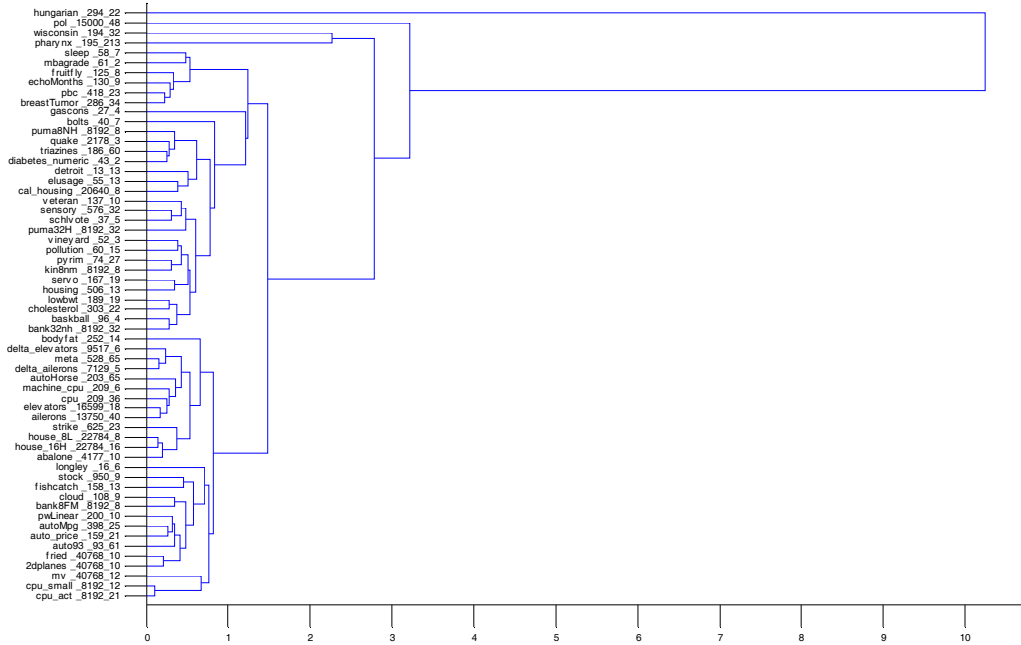


Şekil 9.8 UCI veri koleksiyonunda algoritmaların RMSE değerlerine göre (60 boyutlu) hiyerarşik kümelermeleri

Şekil 9.8 incelendiğinde:

- Tek kural üreten algoritmaların (Conjunctive rule, Decision Stump) bir arada oldukları
- Örnek tabanlı algoritmaların (Kstar, IBK) bir arada oldukları
- Farklı boyut sayısına sahip PLS algoritmalarının bir arada oldukları
- MLP'nin ve SVM'in diğer tüm algoritmalarından farklı bir performans karakteristiği sergilediği görülmektedir.

Veri kümelerinin aynı şekilde kümelemesi yapıldığında Şekil 9.9 elde edilmiştir. Veri kümesi isimleri şeklin sol tarafında yer almaktadır ve “veri kümesi ismi _ örnek sayısı _ özellik sayısı” şeklinde kodlanmıştır.



Şekil 9.9 UCI veri koleksiyonunda veri kümelerinin RMSE değerlerine göre hiyerarşik kümelennmeleri

Şekil 9.9 incelendiğinde performansça birbirine benzeyen veri kümelerinin örnek sayıları ve özellik sayılarına göre herhangi bir şablon içermedikleri görülmüştür.

9.4.4.6 UCI Koleksiyonunda En Başarılı Algoritmaların Performanslarının Meta Özelliklerle Tahminleri

En başarılı 6 algoritmadan 4'ünün (M5P, PLS2, Kstar, Isotonic Reg) RMSE hataları tahmin edilmeye çalışılmıştır. M5P ile M5R'nin, PLS2 ile PLS1'in birbirleriyle korelasyonları çok yüksek olduğundan sadece birer tanelerinin tahmini yapılmış ve diğerlerinin sonuçlarının da diğerine çok benzeyeceği varsayılmıştır.

Diğer algoritmaların performanslarının tahmininde kullanılacak algoritmalar :

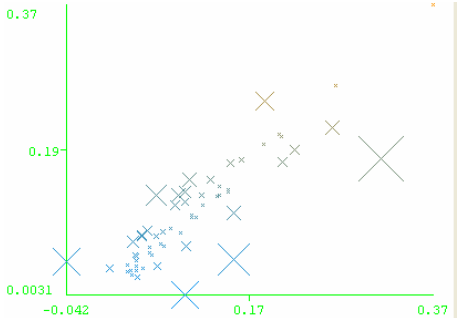
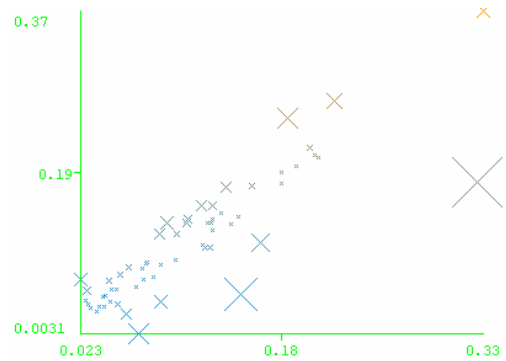
- Decision Stump
- Linear Regression
- Zero0
- RBF
- IBK

olarak seçilmiştir.

Çizelge 9.40, 9.41, 9.42 ve 9.43'te algoritmaların performanslarının tahmin denemeleri verilmiştir.

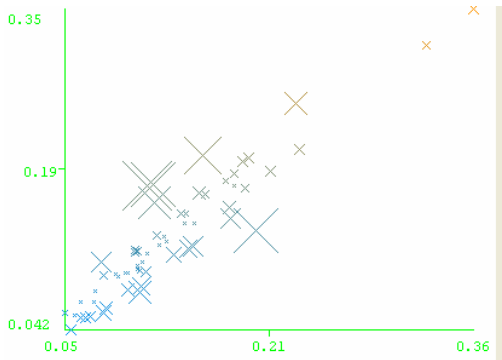
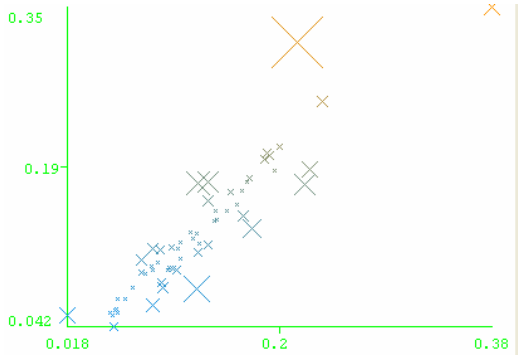
Çizelge 9.40 UCI veri koleksiyonunda M5P algoritmasının performans tahminleri

Tüm özelliklerle (285 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (9 özellikle)
Performans uzayı 7 alt uzaya bölünmüş ve her birinin içinde de REGT.m5p_yapraklardaki_tekil_oz_sayisi RMSE.Decstump RMSE.IBK RMSE.LinearRegression ST2.corXdeg9 ST2.kurtfreX3 ST2.mom4freX4	Performans uzayı 5 alt uzaya bölünmüş ve her birinin içinde de REGT.m5r_ksayitekozsayi_cfsozsayi_orani STA.orneksayi RMSE.IBK RMSE.LinearRegression ST2.stdfreX10 özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.

<p>ST2.skewfreX5</p> <p>ST2.stdfreX3</p> <p>ST2.stdfreX7</p> <p>STA.iqr_o_oran</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.8814</p> <p>Ortalama mutlak hata 0.0237</p> <p>Ortalama karesel hatanın karekökü 0.0364</p> <p>Rölatif mutlak hata 43.0028 %</p> 	<p>Korelasyon Katsayısı 0.8813</p> <p>Ortalama mutlak hata 0.0212</p> <p>Ortalama karesel hatanın karekökü 0.0329</p> <p>Rölatif mutlak hata 38.461 %</p>  <p>Görüldüğü gibi performansın kötü olduğu veri kümeleri üzerindeki performans tahminlerinde daha fazla hata yapılmıştır.</p>
---	--

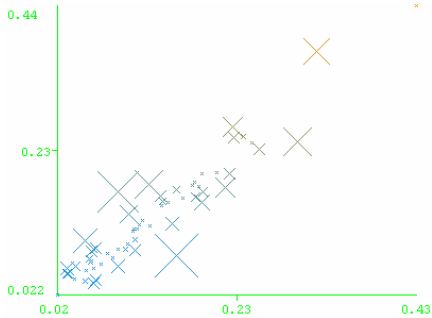
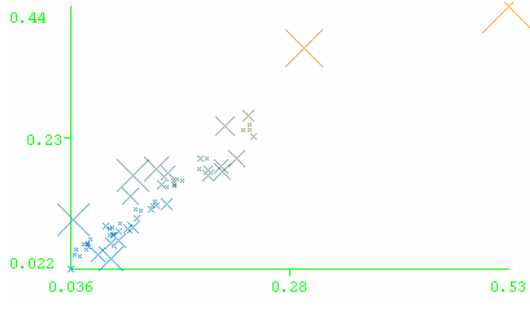
Çizelge 9.41 UCI veri koleksiyonunda PLS2 algoritmasının performans tahminleri

Tüm özelliklerle (285 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (13 özellikle)
Performans uzayı 2 alt uzaya bölünmüş ve her birinde	Performans uzayı 2 alt uzaya bölünmüş ve her birinde
STA.iqr_o_oran	RMSE.Decstump
REGT.m5p_ysayi_orsayi_orani	RMSE.IBK
RMSE.Decstump	RMSE.LinearRegression
RMSE.LinearRegression	RMSE.RBF

<p>RMSE.RBF</p> <p>ST2.freY2</p> <p>ST2.mom4freX10</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.9349</p> <p>Ortalama mutlak hata 0.016</p> <p>Ortalama karesel hatanın karekökü 0.0218</p> <p>Rölatif mutlak hata 32.9325 %</p> 	<p>ST2.corXYfre9</p> <p>ST2.corXYBolustdXY10</p> <p>özelliklerinin çeşitli kombinasyonlarını içeren lineer modeller üretmiştir.</p> <p>Korelasyon Katsayısı 0.9277</p> <p>Ortalama mutlak hata 0.0159</p> <p>Ortalama karesel hatanın karekökü 0.0229</p> <p>Rölatif mutlak hata 32.7879 %</p> 
--	--

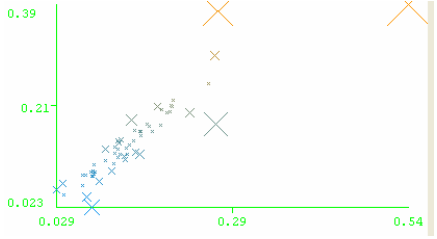
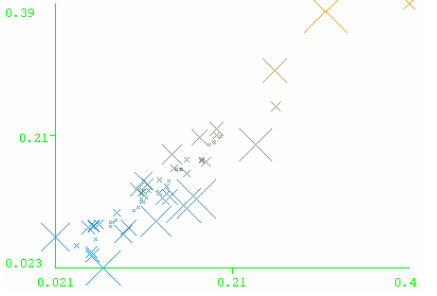
Çizelge 9.42 UCI veri koleksiyonunda Kstar algoritmasının performans tahminleri

Tüm özelliklerle (285 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (9 özellikle)
<p>RMSE.Kstar =</p> <p>0.0725 * STA.iqr_o_oran</p> <p>+ 0.1866 * RMSE.Decstump</p> <p>+ 0.7947 * RMSE.IBK</p> <p>- 0.0282 * ST2.skewfreX1</p> <p>- 0.0105</p>	<p>RMSE.Kstar =</p> <p>0.0966 * RMSE.Decstump</p> <p>+ 0.7739 * RMSE.IBK</p> <p>+ 0.672 * ST2.mom4Y</p> <p>+ 0.0481 * ST2.corXfre7</p> <p>- 0.0045</p>

<p>Korelasyon Katsayısı 0.9471</p> <p>Ortalama mutlak hata 0.019</p> <p>Ortalama karesel hatanın karekökü 0.0259</p> <p>Rölatif mutlak hata 30.047 %</p>  <p>IBK ile Kstar'ın performans korelasyonları çok yüksek olduğundan performans tahmini çok iyi bir şekilde yapılabilmiştir.</p>	<p>Korelasyon Katsayısı 0.9524</p> <p>Ortalama mutlak hata 0.0168</p> <p>Ortalama karesel hatanın karekökü 0.0249</p> <p>Rölatif mutlak hata 26.589 %</p>  <p>En büyük hatalar hata değeri büyük olan veri kümeleri üzerinde yapılmıştır.</p>
--	---

Çizelge 9.43 UCI veri koleksiyonunda Isotonic Reg algoritmasının performans tahminleri

Tüm özelliklerle (285 özellikle)	CFS ile özellik seçimin yapıldıktan sonra (10 özellikle)
<p>RMSE.isotonicreg =</p> <p>0.0006 * REGT.m5p_yapraklardaki_tekil_oz_sayisi</p> <p>- 0.2533 * REGT.m5p_ysayi_orsayi_orani</p> <p>+ 0.8606 * RMSE.Decstump</p> <p>+ 0.079 * RMSE.LinearRegression</p> <p>- 0.0746 * ST2.stdcorXYdeg</p> <p>- 0.0047 * ST2.corXYBolustdXY8</p> <p>+ 0.0195</p>	<p>RMSE.isotonicreg =</p> <p>0.808 * RMSE.Decstump</p> <p>+ 0.0465 * RMSE.IBK</p> <p>+ 0.0534 * RMSE.LinearRegression</p> <p>- 0.0386 * ST2.freY2</p> <p>- 0.0052 * ST2.stdBoluMeanX1</p> <p>- 0.0092 * ST2.corXYBolustdXY10</p> <p>+ 0.0277</p>

<p>Korelasyon Katsayısı 0.9048</p> <p>Ortalama mutlak hata 0.0187</p> <p>Ortalama karesel hatanın karekökü 0.0323</p> <p>Rölatif mutlak hata 35.9926 %</p>  <p>Performanslar başarılı bir şekilde tahmin edilmiştir.</p>	<p>Korelasyon Katsayısı 0.9484</p> <p>Ortalama mutlak hata 0.0159</p> <p>Ortalama karesel hatanın karekökü 0.022</p> <p>Rölatif mutlak hata 30.6167 %</p> 
---	--

Görüldüğü gibi algoritmaların performansları iyi bir şekilde tahmin edilebilmiştir.

Algoritmaların tahmin edilebilme performansları çıkışların aralıkları aynı olmadığından RMSE yerine; Rölatif Mutlak Hata (RAE) ve Korelasyon katsayıları kullanılarak karşılaştırılmıştır. Çizelge 9.44'te algoritmaların tüm meta özellikler ve seçilen meta özelliklerle performanslarının tahmin başarıları bu 2 kritere göre özetlenmiştir.

Çizelge 9.44 UCI koleksiyonunda algoritmaların performanslarını tahmin etme çalışmaları

Performansı tahmin edilen algoritma	Meta Özellik sayısı	Rölatif Mutlak Hata	Korelasyon katsayısı
M5P	286	43.0028 %	0.8814
M5P	9	38.461 %	0.8813
PLS2	286	32.9325 %	0.9349
PLS2	13	32.7879 %	0.9277
Kstar	286	30.047 %	0.9471
Kstar	9	26.589 %	0.9524
Isotonic Reg.	286	35.9926 %	0.9048
Isotonic Reg.	10	30.6167 %	0.9484

M5P haricindeki tüm algoritmaların performansı 0.9'un üzerinde bir korelasyon değeriyle tahmin edilebilmiştir.

Performansı en iyi tahmin edilebilen algoritma Kstar'dır. Buna sebep olarak meta özellikler içinde yer alan IBK algoritmasıyla olan yüksek korelasyonu verilebilir.

Özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

9.4.4.7 UCI Koleksiyonunda Performans Tahminlerinde Kullanılan Meta Özelliklerin Analizleri

Performans tahmininde M5p'nin kurallarında en çok kullanılan meta özellikler ve buna bağlı olarak meta özellik türleri araştırıldığında 298 özellikten 26'sının en az bir kere kurallarda yer aldığı görülmüştür. Kurallarda 1'den fazla yer alan meta özellik sayısı ise 18'dir. Çizelge 9.45'te, bu 18 meta özellik ve kurallarda yer alma sayıları verilmiştir.

Çizelge 9.45 UCI veri koleksiyonunda performans tahminlerinde en çok kullanılan meta özellikler

Meta Özellik İsmi	M5P kurallarında yer alma frekansı
RMSE.LinearRegression	18
RMSE.IBK	17
RMSE.Decstump	9
REGT.m5p_yapraklardaki_tekil_oz_sayisi	8
ST2.skewfreX5	7
ST2.stdfreX10	5
STA.orneksayi	5

REGT.m5r_ksayitekozsayi_cfsozsayi_orani	4
RMSE.RBF	4
ST2.kurtfreX3	4
ST2.stdfreX7	4
STA.iqr_o_oran	4
ST2.corXdeg9	3
ST2.corXYBolustdXY10	3
REGT.m5p_ysayi_orsayi_orani	2
ST2.corXYfre9	2
ST2.freY2	2
ST2.mom4freX10	2

9.4.5 PLS'in Bileşen Sayısının Analizi

Literatürde, PLS algoritmasında kullanıcı tarafından belirlenen (hiper parametre) bileşen sayısının (K) otomatik olarak bulunması üzerine çeşitli çalışmalar yapılmıştır. Bunun sebebi K değerinin değişiminin algoritmanın performansı üzerindeki büyük etkisidir. Bu bölümde ilaç veri kümelerinde, hem K değerinin veri kümesinin çeşitli meta özellikleriyle ilişkisi incelenmiş hem de K değerinin değişiminin PLS algoritmasının performansına etkileri incelenmiştir.

9.4.5.1 PLS'in Bileşen Sayısının Analizinde Kullanılan Veri Kümeleri

PLS'in en yaygın kullanıldığı alan ilaç tasarımı olduğundan bu bölümdeki denemeler de ilaç veri kümeleri üzerinde yapılmıştır. Kullanılan veri kümeleri Çizelge 9.46'da verilmiştir.

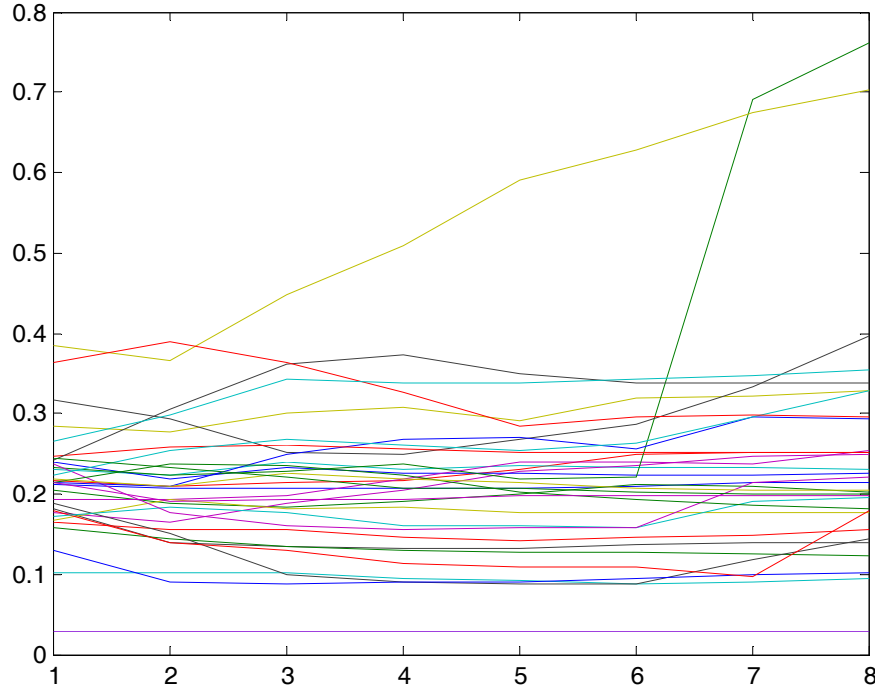
Çizelge 9.46 PLS algoritmasında bileşen sayısının incelenmesinde kullanılan ilaç veri kümeleri

benzo32	heyl	penning	qsbr_rw1	selwood
carbolenes	krystek	Phen	qsbralks	strupcz
chang	mtp	PHENETYL1	qsfrdhla	svensson
cristalli	mtp2	qsabr1	qsfsr1	thompson
depreux	pah	qsabr2	qsprcmpx	topo_2_1
garrat2	pdgfr	qsartox	rosowsky	tsutumi
yokoyama1	yokoyama2	yprop_4_1		

Veri kümelerinin ayrıntıları Bölüm 9.2.3.1'de yer almaktadır.

9.4.5.2 Meta Özelliklerle PLS'in Optimum Bileşen Sayısının Tahmini

İlaç veri kümelerinde, 8 farklı bileşen sayısı (K) için (1,2,3,4,5,6,8 ve 10) PLS algoritması çalıştırılmış ve Şekil 9.10 elde edilmiştir.



Şekil 9.10 8 farklı bileşen değeri için, 33 ilaç veri kümesinde PLS algoritmasının hata değerleri

Şekil 9.10'da X eksenini PLS'deki bileşen sayısının (K) indisini (gerçek bileşen değerleri sırasıyla 1, 2, 3, 4, 5, 6, 8 ve 10'dur), Y eksenini ise veri kümelerindeki performansları göstermektedir. Çizgilerin her biri ise 33 veri kümesinden birini temsil etmektedir.

Minimum hataya sahip denemelerin K değerleri kaydedilmiştir. Daha sonra belirlenen optimum K değerlerinin PCA meta özellikleriyle korelasyonu incelenmiştir.

Optimum K'nın PCA meta özelliklerle yaptığı en yüksek korelasyon değeri 0.35'tir. Bu nedenle optimum K değerinin PCA meta özellikler kümesindeki meta özelliklerle bir ilişkisi bulunamamıştır.

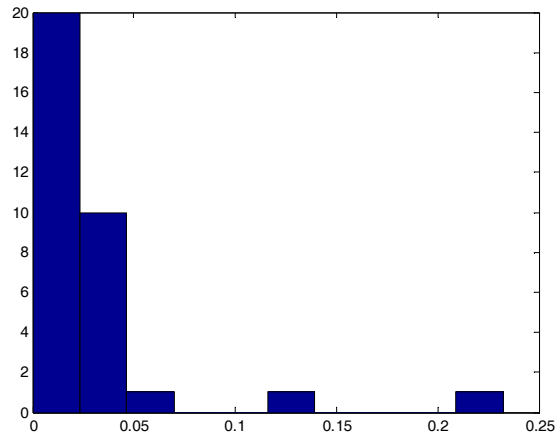
ST2 meta özellikleri ile K ilişkisi de incelenmiş ve bu kümedeki meta özelliklerle yaptığı en yüksek korelasyon 0.41 olarak bulunmuştur.

Bu iki deneme sonucunda ilaç veri kümeleri için, PLS algoritmasında kullanılacak optimum K değerinin meta öğrenme yaklaşımı ile belirlenemeyeceği sonucuna varılmıştır.

9.4.5.3 PLS'in Bileşen Sayısının Performansa Etkisi

Şekil 9.10'da incelendiğinde performansların genelde X eksenine (bileşen sayısına) paralel oldukları görülmektedir. Diğer bir ifadeyle K sayısının çoğu veri kümesi üzerinde PLS'in performansı üzerinde etkili olmadığı söylenebilir.

Şekil 9.11'de ise 33 veri kümesinde K değerinin değişiminin hata üzerindeki standart sapmalarının histogramı verilmiştir. Bu şekilde görülmektedir ki K değeri çok az sayıda veri kümesi üzerinde büyük değişimlere yol açmaktadır.



Şekil 9.11 K değerinin hata üzerinde oluşturduğu değişimin standart sapmalarının histogramı

Bununla birlikte K sayısı bütün veri kümeleri için etkisiz değildir. Bu nedenle minimum hataya sahip olan K değerleri incelenmiş ve en fazla minimum hataya sahip K değerinin 10 veri kümesi ile 2 olduğu; onu 5'er veri kümesiyle 1 ve 5'in takip ettiği görülmüştür.

9.4.6 Meta Özelliklerle En Başarılı Algoritmanın Tahmini

Bir veri kümesinin meta özelliklerine bakarak o veri kümesinde hangi algoritmanın daha başarılı olduğunun tahmin edilmesi için 60 UCI veri kümesi kullanılmıştır. Bu amaçla veri kümesi iki farklı yaklaşımla etiketlenmiş ve problem bir sınıflandırma problemine dönüştürülmüştür.

- 1- Veri kümeleri en başarılı algoritma isimleriyle etiketlenmiştir. 18 algoritmadan 11'i en başarılı algoritmalar kümesine girmiş ve bu 11 algortmada gruplanarak 7 algoritma

ismine dönüştürülmüştür. Bu etiketler Kstar, Lineer, M5, PLS, SVM, MLP ve diğer'dir. Rasgele başarı %40 iken çeşitli sınıflandırma algoritmaları denenmiş ve en başarılı sonuç olarak SVM %53 başarı elde etmiştir.

2- Bu algoritma bu veri kümesinde en yüksek başarıyı gösterir mi sorusuna cevap aranmış ve 2 sınıflı problemler üretilmiştir. En fazla sayıda en başarılı algoritma olan 3 algoritma için 3 ayrı veri kümesi oluşturulmuştur. Alınan sonuçlar aşağıda verilmiştir.

- a. M5 için rasgele başarı %60, en başarılı sonuç %95
- b. SVM için rasgele başarı %88, en başarılı sonuç %95
- c. PLS için rasgele başarı %90, en başarılı sonuç %97

Bu sonuçlar incelendiğinde ilk yaklaşımın çok iyi sonuçlar üretmediği, bunun yanında 2 sınıfla ifade edilen “bu algoritma bu meta özelliklere sahip veri kümesi için en yüksek başarıyı verir mi” problemine tatmin edici cevaplar verebildiği görülmektedir.

Bu iki yaklaşımda en başarılı algoritmayı tahmin ederken kurallar üreten sınıflandırıcılarda en çok kullanılan özellikler REGT, STA grubundandır. Bu sonuç RMSE haricinde, algoritma performans tahminleriyle paralellik göstermektedir.

9.5 Çıkarımlar

Litaratürde “meta learning” alanında yapılan çalışmaların çok büyük bir kısmı sınıflandırma üzerinedir. Regresyon problemleri üzerine kapsamlı bir çalışma yapılmamıştır. Bu bölümde literatürdeki bu eksikliği bir ölçüde kapatmak için regresyon algoritmalarının sonuçlarının tahmin edilmesi ve bu tahminlerde kullanılabilecek meta özellikler üzerinde çalışılmıştır. 3 farklı veri kümesi koleksiyonu ile çalışılmış ve bu sayede oldukça kapsamlı analizler yapılabilmektedir. Bu alanda çalışmak isteyen araştırmacılar içinde oldukça yararlı ve kapsamlı bir kaynak olacak meta regresyon veri kümeleri oluşturulmuştur. Meta regresyon çalışmalarında elde edilen bilgiler Çizelge 9.47’de özetlenmiştir. Çizelgenin ikinci sütununda yer alan “Zero rule RMSE ortalaması” değerinin açıklaması Çizelge 9.7’de yer almaktadır.

Çizelge 9.47 Veri kümesi koleksiyonlarına göre elde edilen sonuçların özetleri

	Zero rule RMSE ortalaması	En başarılı algoritma ve RMSE ortalaması	En başarılı algoritmalar (RMSE ortalamalarına göre sırasıyla)	Algoritma performans tahmininde en çok kullanılan meta özellikler
Yapay veri kümesi koleksiyonu (80 veri kümesi)	0.995	Meta.Bagging 0.501	Meta.Bagging > M5P > M5rules > meta.AttrSelClas > meta.RandomSubSpace > Reptree	CLUS.EM CLUS.kmean STA.orneksayi ST2.corXdeg1 colli RMSE.Decstump ST2.corXdeg2 REGT.m5r_ksayi_orsayi_orani REGT.m5r_ksayi_ozsayi_orani ST2.freY2 REGT.m5p_yaprak_sayisi REGT.m5p_yapraklardaki_tekil_oz_sayisi REGT.m5p_ysayi_ozsayi_orani REGT.m5p_ytek_ozsayi_ozsayi_orani REGT.rep_nsayi_cfsozsayi_orani REGT.rep_nsayi_orsayi_orani ST2.corXfre1 ST2.corXYfre7 ST2.freY1 ST2.freY4 ST2.skewfreX1 ST2.skewfreY ST2.stdcorXdeg ST2.stdcorXfre ST2.stdfreX5 STA.cfs_sayi
İlaç veri kümesi koleksiyonu (41 veri kümesi)	0.25	Kstar 0.222	Kstar > PLS1 > PLS2 > PLS4 > PLS3 > M5P > IBK > PLS5 > ConjunctiveRule	RMSE.Decstump RMSE.ZeroR ST2.kurtY RMSE.REPTree ST2.corXfre4 ST2.kurtfreX9 REGT.m5p_ytek_ozsayi_ozsayi_orani REGT.m5r_ksayi_orsayi_orani ST2.bigcorXpro ST2.corXYBolustdXY7 ST2.kurtfreX6 ST2.mom4freX5 ST2.skewfreX10 ST2.stddegX7 STA.cfs_sayi
UCI veri kümesi koleksiyonu (60 veri kümesi)	0.195	M5P 0.106	M5P > M5R > PLS2 > Kstar > Isotonic Reg > PLS1	RMSE.LinearRegression RMSE.IBK RMSE.Decstump REGT.m5p_yapraklardaki_tekil_oz_sayisi ST2.skewfreX5 ST2.stdfreX10 STA.orneksayi REGT.m5r_ksayitekozsayi_cfsozsayi_orani RMSE.RBF ST2.kurtfreX3 ST2.stdfreX7 STA.iqr_o_oran ST2.corXdeg9 ST2.corXYBolustdXY10 REGT.m5p_ysayi_orsayi_orani ST2.corXYfre9 ST2.freY2 ST2.mom4freX10

Çizelge 9.47 incelendiğinde aşağıdaki sonuçlara ulaşılmıştır:

- Zero rule'a göre ortalama RMSE'si en yüksek veri kümeleri, üretilen yapay veri kümeleridir.
- Her veri koleksiyonunda en başarılı algoritma farklı bir algoritmadır, dolayısıyla hiçbir algoritmanın tüm veriler üzerinde en başarılı olmadığı görülmüştür.
- İlaç veri kümelerinde, algoritmaların rasgele hatayı (zero rule hatası) çok az düşürebildikleri görülmüştür. Bu nedenle en zor modellenen veri kümeleri oldukları söylenebilir.
- M5P algoritması her 3 veri kümesi koleksiyonunda da en iyi performans gösteren algoritmalar arasındadır.
- Bir algoritmanın veri kümesindeki hatası büyükse, onu tahmin etmek zordur. Başarılı sonuçları tahmin etmek daha kolaydır.

Çizelge 9.48'de 3 koleksiyonun performans tahminlerinde en sık kullanılan meta özellikler ve 3 koleksiyondan kaçında kullanıldıkları verilmiştir.

Çizelge 9.48 Üç veri kümesi koleksiyonunda performans tahminlerinde en sık kullanılan meta özelliklerin kesişim kümesi

Meta özellik	3 koleksiyondan kaçında kullanıldığı
REGT.Verit kümesi üzerinde M5rules ile bulunan kural sayısının örnek sayısına oranı	3
RMSE.Decstump algoritmasının veri kümesi üzerindeki RMSE değeri	3
REGT.Verit kümesi üzerinde oluşturulan M5P karar ağacının yapraklarında (kararlarında) en az 1 kere kullanılmış özellik sayısı	2
REGT. Verit kümesi üzerinde oluşturulan M5P karar ağacındaki yaprak sayısının örnek sayısına oranı	2
REGT. Verit kümesi üzerinde oluşturulan M5P karar	2

ağacının yapraklarında en az 1 kere kullanılmış özellik sayısının özellik sayısına oranı	
STA. Veri kümesinde cfs ile seçilen özellik sayısı	2
STA. Veri kümesindeki örnek sayısı	2

Bu denemelere ek olarak, ilaç veri kümelerinde en çok kullanılan algoritmalarından biri olan PLS algoritmasının bileşen sayısının analizi de yapılmış ve aşağıdaki sonuçlara ulaşılmıştır.

- PLS algoritmasında kullanılacak optimum bileşen sayısı meta öğrenme yaklaşımı ile belirlenemez.
- Bileşen sayısı çoğu veri kümesi üzerinde PLS'in performansı üzerinde etkili değildir.
- Minimum hataya sahip olan K değerleri incelendiğinde ve en fazla minimum hataya sahip K değerinin 10 veri kümesi ile 2 değeridir. Onu 5'er veri kümesiyle 1 ve 5 değerleri takip etmektedir.

9.6 Gelecek Çalışmalar

Yeni meta özelliklerin keşfi ve henüz literatüre girmemiş olan *meta kümeleme* gelecek çalışmalar olarak söz edilebilir.

10. REGRESYON KOMİTELERİ

Kümeleme ve özellikle sınıflandırma komitelerinde elde edilen başarılı sonuçlar regresyon problemlerinde de komitelerin başarılı sonuçlar üretebileceğini düşündürmüştür. Literatürde regresyon komiteleri için birçok algoritma önerilmiş ve başarılı sonuçlar alınmıştır.

Bu bölümde, ilaç veri kümelerinde regresyon komitelerinin performansının 2 kriterden nasıl etkilendiği araştırılmıştır. Bu kriterler, komitenin oluşturulmasında, algoritmaların kararlarının birleştirilmesinde kullanılan algoritma (meta algoritma) ve komitede kullanılan algoritmadır. Bu bölümde bu amaçla 4 farklı komite metodu (meta algoritma) ve her birinde 5 farklı algoritma kullanılarak komitelerin performansı ölçülmüştür.

10.1 Önceki Çalışmalar

Literatürde önerilen birçok regresyon algoritması ve meta algoritma vardır. Bu algoritmalarda Bölüm 9’da elde edilen deneyimler sonucunda en başarılı olanlar regresyon komiteleri oluşturmak için kullanılmışlardır.

10.1.1 Komite Algoritmaları

Bagging: Bölüm 4.2’de anlatılmıştır.

Additive Regresyon: AdaBoost algoritmasının regresyon problemlerine uyarlanmasıdır (Friedman, 1999). Her bir iterasyonda bir önceki algoritmanın başarılı olamadığı kısımlara yoğunlaşan bir meta algoritmadır. Her bir iterasyonda üretilen algoritma sonuçları birleştirilerek çıkış verir.

Özellik Seçimli Meta Algoritma: Veri kümesi algoritmaya verilmeden önce özellik sayısı azaltılır. Özellik seçiminde CfsSubsetEval kullanılmıştır. Bu metot (Hall, 1998) özelliklerden çıkışla yüksek korelasyonlu, birbirleriyle düşük korelasyonlu bir özellik alt kümesi seçmektedir.

Rasgele Alt Uzay: Komitedeki her bir algoritmaya veri kümesinin tüm özellikleri yerine rasgele seçilen alt kümeleri verilir (Ho, 1998). Bu yaklaşımla algoritmaların sonuçları arasındaki ayrıklık (diversity) arttırılmış olur. Alt uzaydaki özellik sayısı orijinal özellik sayısının yarısıdır.

10.1.2 Regresyon Algoritmaları

M5 Model Ağaçları: Bölüm 8.2.4.1’de anlatılmıştır.

RepTree: Hızlı bir regresyon ağacı algoritmasıdır. Her bir düğümde varyansı en fazla azaltan özellik, sınır değeri seçilerek oluşturulur. Daha sonra en alt düğümlerden başlanarak rekürsif bir biçimde hata azalması tabanlı budama (reduced-error pruning) gerçekleştirilir.

Partial Least Squares: Bölüm 8.2.3’te anlatılmıştır.

Simple Linear Regresyon: Çizelge 9.7’de anlatılmıştır.

Kstar: Bölüm 8.2.5’te anlatılmıştır.

10.2 Gerçekleştirilenler

Regresyon tipi ilaç veri kümeleri için, farklı algoritmaların farklı komite metodlarındaki performans karşılaştırmaları yapılmıştır. 4 farklı komite metodu ve her birinde 5 farklı algoritma kullanılarak komitelerin performansı ölçülmüştür. Çizelge 10.1 ve 10.2’de kullanılan algoritmalar ve kısaltmaları verilmiştir. Sonuçlara dair tablolarla algoritmalar ve komitelerin isimleri yerine bu tablolardaki kısaltmalar kullanılmıştır.

Çizelge 10.1 Kullanılan komite oluşturma algoritmaları

Komite ismi	Kısaltması
Bagging	BG
Additive Regresyon	ADD
Özellik seçimli	ATT
Rasgele alt uzay	RS

Çizelge 10.2 Kullanılan regresyon algoritmaları

Algoritma ismi	Kısaltması
M5 Model Ağaçları	M5P
RepTree	RT
Partial least Squares	PLS
Simple linear regresyon	SLR
Kstar	KS

Komitelerin ve algoritmaların seçiminde ilaç problemlerinde 9. bölümde elde edilen sonuçlar göze alınmış ve en başarılı olanlar bu bölümde daha detaylı bir incelemeye alınmıştır.

Bu denemelerin yapılmasıyla cevap aranan sorular aşağıda sıralanmıştır:

- Orijinal algoritmalar yerine komiteleri kullanmak daha başarılı sonuçlar üretir mi?
- Hangi komite daha başarılı sonuçlar üretmektedir?
- Hangi algoritma – komite ikilisi en iyi/kötü sonuçları üretmektedir?
- Hangi algoritmalar komitelerle daha iyi çalışmaktadır?
- Hangi algoritmalar daha iyi çalışmaktadır?
- Algoritmaların birbirlerine benzerlikleri (performanslarına göre) nasıldır?

Bu sorulara cevap verebilmek için Çizelge 10.3'te verilen 36 ilaç veri kümesi ile (4 komite +1 orijinal) * (5 algoritma) =25 farklı algoritma 5'li çapraz geçерleme ile çalıştırılmış ve RMSE sonuçları alınmıştır.

Çizelge 10.3 Regresyon komiteleri denemelerinde kullanılan veri kümeleri

Veri kümesi ismi	Özellik sayısı	Örnek sayısı
benzo32	33	195
carbolenes	1143	37
chang	1143	34
cristalli	1143	32
depreux	1143	26
qarrat	1143	10
qarrat2	1143	14
hald12	4	13
heyl	1143	11
krvstek	1143	30
leardi2	11	16
mtp	203	4450
mtp2	1143	274
pah	113	80
pdqfr.arff	321	79
penning	1143	13
Phen.arff	111	22
PHENETYL1.arff	629	22
qsabr1.arff	10	15
qsabr2.arff	10	15
qsartox.arff	24	16
qsbr_rw1.arff	51	14
qsbr_v2.arff	10	25
qsbralks.arff	22	13
qsfrdhl1.arff	34	16
qsfsr1.arff	10	20
qsfsr2.arff	10	19
qsprcmpx.arff	40	22
rosowsky	1143	10
selwood.arff	54	31
siddiqi	1143	10
strupcz	1143	34
svensson	1143	13
tsutumi	1143	13
yokoyama1	1143	13
yokoyama2	1143	12

Çizelge 10.3'de yer alan veri kümelerinin ayrıntıları Bölüm 9.2.3.1'de yer almaktadır.

Denemeler sonucunda algoritmalarından bazılarının (SLR vb.) çok yüksek RMSE'lere sahip oldukları görülmüştür. Bu nedenle algoritmaların başarılarını karşılaştırmak için RMSE'lerin ortalamalarını almak yerine her bir veri kümesi için algoritmaların RMSE'leri küçükten büyüğe sıralanmış ve algoritmaların kaçınıcı en iyi performansa sahip oldukları bulunmuştur. Çizelge 10.4'te bu sıralamalar yer almaktadır. Çizelgenin sütunlarında veri kümelerinin indisleri, satırlarında ise algoritmalar yer almaktadır. Son 2 sütunda ise her bir algoritmanın sıra ortalaması ve standart sapması görülmektedir.

Çizelge 10.4 25 Komite-Algoritma ikililerinin 36 veri kümesi üzerindeki sıralamaları

veriID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	ort	std
RG Rep	14	20	15	25	15	5	13	3	11	19	21	24	16	11	25	12	13	8	8	6	9	10	17	22	8	7	5	9	17	16	7	9	8	17	22	17	13.44	6.14
RG M5P	23	19	13	22	10	9	3	10	10	9	10	21	1	25	23	20	17	2	16	13	13	9	15	7	20	10	16	16	7	25	5	13	2	18	17	22	14.14	6.88
RG PLS2	18	25	12	10	17	17	22	18	21	16	9	7	18	16	14	24	20	18	15	17	20	20	14	17	10	11	19	15	18	22	1	17	20	5	13	13	16.31	5.07
RG SLR	13	10	7	1	2	3	11	11	1	2	20	3	12	14	22	1	12	11	17	12	19	10	7	24	18	17	25	23	2	10	6	2	22	11	10	6	11.28	7.43
RG KS	9	4	25	16	10	21	12	10	17	24	4	16	24	13	8	8	14	1	25	9	12	8	22	21	7	21	9	21	25	11	15	24	10	24	25	14	15.50	7.28
ADD Rep	5	13	10	10	12	4	5	2	10	5	14	11	3	1	5	6	6	6	4	1	8	5	11	5	4	5	3	7	10	8	20	7	14	10	21	16	8.47	5.61
ADD M5P	24	7	11	4	7	11	23	23	13	3	25	10	6	22	4	3	21	23	2	25	2	17	21	10	23	13	15	5	5	1	12	6	16	10	3	21	12.92	8.10
ADD PLS	17	21	5	21	9	23	10	24	25	12	1	22	21	23	7	21	24	20	13	22	25	23	4	1	12	2	10	22	15	17	2	12	10	6	15	7	14.81	7.87
ADD SLR	25	2	1	2	1	2	8	25	3	1	5	18	7	20	12	25	25	16	12	23	16	22	3	2	11	1	18	25	19	5	16	1	1	21	4	3	11.14	9.14
ADD KS	2	22	23	20	25	8	24	8	9	25	2	5	25	7	2	11	3	1	23	3	15	4	24	4	5	10	7	18	6	13	17	20	13	12	10	12.94	8.16	
ATT Rep	6	15	18	11	11	16	1	4	20	23	18	10	8	2	11	14	5	9	21	4	6	7	9	16	2	4	1	2	12	20	9	22	7	9	16	15	10.67	6.40
ATT M5P	22	8	14	3	3	22	4	21	4	4	11	12	14	15	24	4	22	25	19	21	10	15	5	6	25	8	11	10	4	14	22	5	6	1	5	8	11.86	7.56
ATT PLS2	20	17	6	12	14	18	7	20	8	14	7	14	15	19	17	10	23	22	18	24	18	12	16	14	15	9	13	20	8	21	18	8	3	20	2	5	14.33	5.91
ATT SLR	11	6	3	6	4	1	18	7	6	7	23	2	10	10	19	16	8	14	6	15	4	14	2	9	22	16	24	13	21	4	25	11	24	14	12	2	11.36	7.18
ATT KS	3	9	21	15	13	24	14	14	22	18	13	17	2	4	9	22	11	12	9	10	22	6	20	20	16	25	14	24	3	24	10	21	5	25	9	14.81	7.03	
RS Rep	7	14	20	17	18	12	9	5	14	17	15	23	13	8	13	10	1	7	7	5	7	1	19	18	3	6	4	4	0	15	11	10	12	8	20	18	11.36	6.00
RS M5P	16	18	16	9	22	25	6	15	2	8	16	25	20	24	20	2	16	24	14	10	21	24	8	12	14	23	17	8	16	9	21	3	15	4	1	12	14.33	7.22
RS PLS2	15	24	9	24	24	13	10	17	23	15	10	9	19	18	16	18	10	10	20	16	24	16	12	11	10	24	20	14	14	23	4	16	17	2	8	10	15.80	5.77
RS SLR	12	11	4	8	6	7	15	12	7	20	17	6	11	12	21	13	9	17	11	11	11	21	6	10	17	20	21	1	22	6	23	14	25	16	14	4	13.06	6.21
RS KS	8	1	22	14	20	20	13	16	22	8	15	22	5	6	9	10	10	24	8	23	11	25	23	13	22	8	19	24	10	10	25	11	22	24	25	16.03	6.90	
Rep	4	12	17	18	16	15	2	1	18	11	12	13	4	3	10	5	4	5	3	7	5	2	10	15	1	3	2	3	11	7	13	15	4	7	18	14	8.61	5.66
M5P	21	16	8	7	8	10	21	22	12	10	24	20	5	21	3	23	15	15	1	20	1	25	18	25	24	12	12	6	1	2	8	4	21	15	7	20	13.42	7.87
PLS2	19	23	10	23	23	14	16	16	24	13	6	8	17	17	15	17	18	21	10	18	17	18	13	13	9	14	22	12	13	18	3	18	18	3	6	11	14.80	5.54
SLR	10	5	2	5	5	6	17	6	5	6	22	1	9	9	18	15	7	13	5	14	3	13	1	8	21	15	23	11	20	3	24	10	23	13	11	1	10.56	6.95
KS	1	3	24	13	21	19	25	9	15	21	3	4	23	6	1	7	2	3	22	2	14	3	23	3	6	18	6	17	23	12	14	23	9	23	23	23	12.89	8.50

Çizelge 10.5'te ise Çizelge 10.2'in özet sonuçları verilmiştir. Çizelgedeki her bir hücre satırındaki algoritmanın, sütundaki komite ile kullanıldığındaki sıralama ortalamasını göstermektedir. Orj sütununda algoritmaların tek başlarına kullanıldıklarındaki sıralama ortalamaları verilmiştir. Ort sütununda satırındaki algoritmayı içeren algoritmaların, Ort satırında sütundaki algoritmayı içeren algoritmaların sıralama ortalamaları verilmiştir.

Çizelge 10.5 Algoritmaların ve komite metotlarının ortalama sıraları

	BG	ADD	ATT	RS	Orj	Ort
Rep	13.44	8.47	10.67	11.36	8.61	10.51
M5P	14.14	12.92	11.86	14.33	13.42	13.33
PLS2	16.31	14.81	14.33	15.89	14.89	15.24
SLR	11.28	11.14	11.36	13.06	10.56	11.48
KS	15.50	12.94	14.81	16.03	12.89	14.43
Ort	14.13	12.06	12.61	14.13	12.07	

Çizelge 10.4 ve 10.5 incelendiğinde;

- En iyi sonucun (8.47) *Additive regresyon- RepTree* ikilisi ile elde edildiği görülmektedir.
- Reptree sadece Additive regresyon ile tek başına kullanıldığından daha iyi sonuç vermiştir.
- M5P ve PLS, ADD ve ATT regresyon ile tek başına kullanıldığından daha iyi sonuç vermiştir.
- SLR ve Kstar, komitelerin hiçbirinde tek başına kullanıldığından daha iyi sonuç vermemiştir.

Bu sonuçlar doğrultusunda genel olarak komitelerin kullanımının regresyon sonuçlarına iyi yönde katkıda bulunmadığı söylenebilir. Sadece ADD'nin ortalaması orijinal algoritmaların ortalamasından düşüktür.

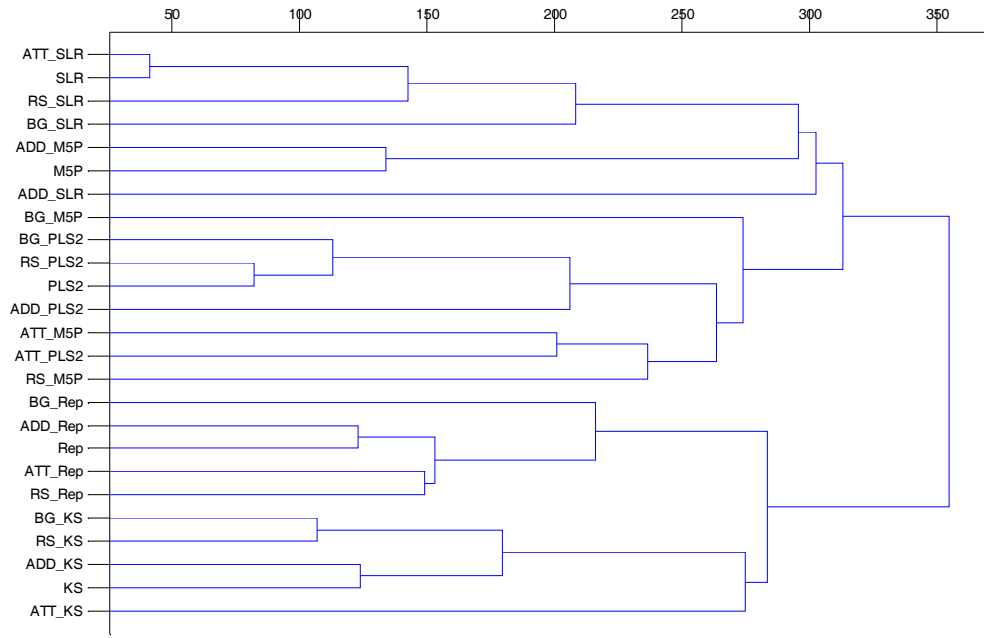
Çizelge 10.6'da sıralama ortalamalarına göre algoritmaların küçükten büyüğe sıralanması verilmiştir.

Çizelge 10.6 Meta algoritma- regresyon algoritması ikililerinin başarı sıralamalarının ortalamaları

Algoritma	Sıra	Algoritma	Sıra ort.
ADD_Rep	8.47	M5P	13.42
Rep	8.61	BG_Rep	13.44
SLR	10.56	BG_M5P	14.14
ATT_Rep	10.67	ATT_PLS2	14.33
ADD_SLR	11.14	RS_M5P	14.33
BG_SLR	11.28	ADD_PLS2	14.81
ATT_SLR	11.36	ATT_KS	14.81
RS_Rep	11.36	PLS2	14.89
ATT_M5P	11.86	BG_KS	15.50
KS	12.89	RS_PLS2	15.89
ADD_M5P	12.92	RS_KS	16.03
ADD_KS	12.94	BG_PLS2	16.31
RS_SLR	13.06		

Çizelge 10.6 incelendiğinde RepTree içeren algoritmaların genelde yukarı sıralarda yer aldıkları. SLR'nin basitliğine rağmen 3. sıraya yerleştiği görülmektedir. Ancak SLR bazı veri kümelerinde (boyut sayısının veri sayısından çok olduğu) çözüm yolu sebebiyle çok yüksek RMSE değerleri vermiştir. Bu nedenle SLR'nin yapısına uygun olmayan veri kümeleri haricinde iyi bir tercih olduğu söylenebilir.

Şekil 10.1'de algoritmaların 36 veri kümesi üzerindeki performanslarına göre gruplanmaları verilmiştir.



Şekil 10.1 Komitelerin 36 ilaç veri kümesi üzerindeki performanslarına göre gruplandırma sonuçları

Şekil 10.1 incelendiğinde;

- Komitelerin içerdikleri algoritmaların türlerine göre gruplandıkları
- Sadece M5P'nin farklı gruplarda (PLS2 ve SLR) yer aldığı
- Kstar ve Rep algoritmaları bir grup, diğer algoritmaların ayrı bir grup oluşturduğu görülmektedir.

10.3 Çıkarımlar ve Gelecek Çalışmalar

Bu bölümün başında yer alan sorulara yaptığımız denemelerle verdiğimiz cevaplar Çizelge 10.7’de verilmiştir. Verilen cevaplar kullandığımız ilaç veri kümelerinde geçerlidir.

Çizelge 10.7 Regresyon komitelerinde cevap aranan sorulara deneylerin verdiği cevaplar

Soru	Cevap
Orijinal algoritmalar yerine komiteleri kullanmak daha başarılı sonuçlar üretir mi?	Genelde hayır.
Hangi komite daha başarılı sonuçlar üretmektedir?	Sıralama: ADD > Orj > ATT > BG =RS
Hangi algoritma – komite ikilisi en iyi/kötü sonuçları üretmektedir?	En iyi: ADD-Rep En kötü: BG- PLS
Hangi algoritmalar komitelerle daha iyi çalışmaktadır?	M5P ve PLS, 2 komitede (ADD, ATT) tek başına kullanıldıklarından daha başarılı sonuçlar vermiştir.
Hangi algoritmalar daha iyi çalışmaktadır?	Sıralama: Rep > SLR > KS > M5P > PLS2
Algoritmaların birbirlerine benzerlikleri (performanslarına göre) nasıldır?	Temelde algoritmalar komite türlerine göre değil komitelerde kullanılan algoritmalara göre gruplanmıştır. Bu nedenle performansı komite türünün değil komitede kullanılan algoritmanın belirlediği söylenebilir.

Gerçekleştirilen çalışmalardan sonra gelecekte yapılması düşünülen çalışmalar aşağıda listelenmiştir:

- Sadece ilaç veri kümelerinde yapılan denemelerin diğer veri kümesi türlerinde de denenip sonuçların genelleştirilmesi ya da özelleştirilmesi
- Elde edilen sonuçların (RepTree’nin yüksek performansı) nedenlerinin araştırılması

11. SONUÇ

Makine öğrenmesi (yapay öğrenme), eldeki verileri en iyi temsil eden modeli ve bu modelin parametrelerini bulmak amacıyla geliştirilen algoritmaları içerir. Tezde çeşitli makine öğrenmesi algoritmaları geliştirilmiştir. Günümüzde incelenmesi ve yorumlanması gereken veri miktarı üssel bir biçimde artmaktadır. Bu durum makine öğrenmesinin tüm sektörlerin ihtiyaç duyduğu bir alan haline gelmesine sebep olmuştur. Tezin uygulama alanı olarak, bu sektörlerden biri olan ilaç tasarımı seçilmiştir.

İlaçların insan sağlığına olan olumlu etkileri geçmiş yıllarda tehlikeli olan birçok hastalığın artık bir tehdit oluşturmadığı gerçeğiyle görülebilir. Ancak yeni ortaya çıkan hastalıklar ve henüz tedavisi bulunmamış hastalıklar yeni ilaçların tasarımı vazgeçilmez kılmaktadır. Bununla birlikte ilaç tasarımı emek ve zaman isteyen ve dolayısıyla çok maliyetli bir sektördür. Yüksek maliyet sebebiyle az sayıda firma tarafından gerçekleştirilebilmektedir. Bu yüksek maliyeti azaltmak için firmalar makine öğrenmesi metotlarına yönelmişlerdir. İlaç tasarımı problemlerinde makine öğrenmesinin neredeyse tüm alanlarına ihtiyaç duyulmakta ve kullanılmaktadır. Bu nedenle de tezde makine öğrenmesinin birçok alanını kapsayacak bir çalışma gerçekleştirilmiştir.

Her bölümde yapılan deneysel çalışmalardan elde edilen sonuçların özetleri aşağıda verilmiştir. Ayrıntılar için her bölümün sonunda yer alan “Çıkarımlar” başlıklarına bakılmalıdır.

Sınıflandırma problemleri için karar ağacı tabanlı **Cline** adı altında bir algoritma ailesi tasarlanmıştır. Elde edilen sonuçlar özetlenirse:

- UCI ve ilaç veri kümesi koleksiyonlarında Cline algoritmaları umut vaat eden sonuçlar üretmişlerdir. Ancak en başarılı algoritmalar değildir.
- UCI veri kümelerinde, Cline algoritmalarında ağaç budama işlemi ortalama başarı üzerinde küçük bir iyileştirme yapmıştır.
- UCI veri kümelerinde, bir test örneğinin bir karar ağacı kullanılarak sınıflandırılması için Cline içinde önerilen dalların kullanımı, klasik yaprakların kullanımına göre biraz daha yüksek bir doğruluğa sahiptir.
- UCI veri kümelerinde, Cline algoritmaları içinde, hiper düzlem belirleme metotları arasında CLMIX belirgin bir farkla en yüksek doğruluğa sahip metottur. Bunun sebebi

olarak, CLMIX'in, literatürdeki mevcut algoritmaların aksine, ağacın tüm karar düğümlerinde aynı algoritmaya bağımlı olmayıp, her Cline algoritmasını daha başarılı olduğu bölgeye uygulaması verilebilir.

- UCI veri kümelerinde, ağaç budama işleminin CL2, CL4 ve CLLVQ metotları haricinde başarıyı arttırdığı ve budama yapılmadığında dal kullanımının yaprak kullanımına göre daha yüksek başarılarla sahip olduğu görülmüştür. Bu sonuçlara göre budama yapılarak eğitim kümesinin boyutu azaltılmak istenmediğinde sınıflandırmada dalların kullanılmasının daha iyi olduğu ve bu metodun başka karar ağacı algoritmalarında da uygulamanın iyi bir fikir olduğu söylenebilir.
- İlaç veri kümelerinde, UCI koleksiyonunun aksine CLLDA'in başarısı düşüktür. Bunun sebebinin, ilaç veri kümelerindeki özelliklerin birbirleriyle korelasyonunun yüksek olmasının, LDA hesaplanırken sebep olduğu rasgelelik olduğu görülmüştür. Ayrıca yine UCI koleksiyonunun aksine budamanın performansa olumsuz etki ettiği görülmektedir. Tam ağaçlar ilaç verilerinde budanmışlardan daha başarılıdır.

Sınıflandırıcı komiteleri literatürdeki birçok çalışmada tekil sınıflandırıcılardan daha başarılı sonuçlar üretmiştir. Bu çalışmada da buna paralel sonuçlar alınmış ve Cline algoritma ailesine Cline karar ormanları eklenmiştir. Cline karar ağacı ve karar ormanları algoritmaları ClineToolbox adlı bir yazılımla kullanıcıların hizmetine sunulmuştur. Yazılıma tez sahibinin web sayfasından erişilebilir. Elde edilen sonuçlar özetlenirse:

- UCI ve ilaç veri kümesi koleksiyonunda da en başarılı algoritmalar Cline algoritmalarıdır.
- Literatürdeki sonuçlara paralel olarak, tek ağaç yerine birden fazla ağaçtan oluşan ormanları kullanmanın daha iyi sonuçlar verdiği görülmüştür.
- Ağaç sayısı arttıkça başarı da artmaktadır. Buna dayanılarak Breiman'ın çalışmasında (Breiman, 1999) belirttiği gibi ağaç sayısını arttırmanın aşırı eğitime (overfitting) sebep olmadığı söylenebilir.
- Ormanlardaki ağaçlarda kullanılan özellik sayısının artması Random Forest'ta başarıyı azaltmışken, Cline versiyonlarında arttırmıştır.
- Cline Ormanlarında tek ağaçlı Cline versiyonlarında olduğu gibi dal kullanımı yaprak kullanımına göre daha yüksek bir başarıya sahiptir.

- CLMIX, CLM'den başarılıdır. CLMIX ormanları da, CLM ormanlarından daha başarılıdır. Bu ifadeyi genelleştirilecek olursak, daha başarılı algoritmaların birleştirilmesi (ormanlarının oluşturulması) daha iyi sonuçlar vermektedir denilebilir. Bu ifade Breiman'ın çalışmasında da yer almaktadır. Bu sonuçtan yola çıkılarak QUEST gibi tek ağaçlı algoritmaların orman versiyonlarının da orijinallerinden daha başarılı olacağı söylenebilir.
- UCI koleksiyonunda, Cline karar ormanlarına bagging eklenmesi hemen hemen her veri kümesinde ve sonuç olarak ortalama başarı da arttırmıştır.
- UCI koleksiyonunda, ClineMIX ormanlarının performanslarının standart sapma miktarı, Random Forest'lardan daha düşüktür. Bu sebeple daha güvenilir sonuçlar ürettiği söylenebilir.
- UCI koleksiyonunda, algoritmaların performanslarıyla, eğitim ve test toplam sürelerinin arasındaki ilişki incelenmiştir. Algoritmaların karmaşıklıklarıyla performansları arasında pozitif bir ilişki olduğu görülmüştür. Diğer bir ifadeyle daha karmaşık algoritmalar genelde daha iyi performans göstermiştir. En yüksek başarıya sahip algoritma olan CLMIX ormanı, en yavaş algoritma değildir.
- İlaç verileriyle yapılan çalışmalar sonucunda en yüksek başarı, 100 ağaçlı CLM ormanı ile kararlar ağaçtaki düğüm sayısı ile doğru orantılı bir şekilde ağırlıklandırıldığında elde edilmiştir.
- İlaç verilerinde Bootstrapping, dal kullanımı ve ağaç budama UCI verilerinin aksine başarıyı azaltmıştır.

Kümeleme problemleri için karar ağaçlarından esinlenilerek **Clusline** adı altında bir algoritma ailesi geliştirilmiş ve çeşitli kümeleme performans kriterlerine göre popüler kümeleme algoritmaları ile karşılaştırılmıştır. Elde edilen sonuçlar özetlenirse:

- Sınıflandırma başarısı ve ranka göre başarı sıralaması: *K-means>SOM>diğer*
- Siluet Genişliğine göre başarı sıralaması: *Clusline2>Clusline4>diğer*
- Davies-Bouldin indeksine göre başarı sıralaması: *SOM>Clusline2>diğer*
- Clusline basitliğine rağmen başarılı sonuçlar elde etmiştir. Sadece SOM ve Clusline2, 3 kriterin 2'sinde ilk 2'ye girebilmişlerdir.

- Siluet genişliği ve DB indeks, tanımlarındaki benzerliğe rağmen oldukça farklı sonuçlar üretmiştir.

Kümeleme komiteleri, sınıflandırıcı komitelerinin üstün performanslarından esinlenilerek geliştirilmiştir. Literatürdeki kümeleme komitelerinin farklı karar birleştirme teknikleri incelenmiş ve geniş bir veri kümesi üzerinde bu teknikler karşılaştırılmıştır. Literatürdeki mevcut karşılaştırmalardan daha kapsamlı olan bu çalışma bu konuda çalışanlara yol gösterici niteliktedir.

- Kümeleme yaparken, karar birleştirme yapmak genelde daha iyi sonuçlar üretmektedir. Ancak bu artış sınıflandırıcı algoritmalarındaki kadar belirgin değildir.
- Ortalama olarak graf algoritmaları daha yüksek başarıya sahiptir ancak bireysel başarılarında hiyerarşik algoritmalar daha başarılıdır.
- Kmeans ve Fuzzy kmeans en başarılı kümeleme algoritmalarıdır.
- Karar birleştirmenin başarısını en fazla arttırdığı kümeleme metodu eklemeli kümelemedir. Bunun sebebi eklemeli kümelemenin tekil başarısının diğerlerine göre daha düşük olmasıdır. Ancak eklemeli kümelemenin kararları birleştirildiğinde bile diğer algoritmaların performansına ulaşamamaktadır.
- Kümeleyicilerde kullanılan özellik sayısının ve kümeleyici sayısının artırılması performansı artırırken standart sapmayı azaltmaktadır.
- Özellik sayısı fazla iken, kümeleyici sayısı artarken performans daha hızlı yükselirken, standart sapma daha hızlı azalmaktadır.

Özellik seçimi problemleri için karar ağaçları ve karar ormanlarından yararlanan bir yaklaşım geliştirilmiş ancak tatmin edici sonuçlar elde edilememiştir. Önerilen yaklaşımın geliştirilmesi gerekmektedir. Denemelerden elde edilen sonuçlar aşağıda özetlenmiştir:

- Özellik seçimi metotları arasında çok büyük performans farklılıklarının olmadığı görülmüştür.
- Özellik seçimi 13 veri kümesinden 4'ünde başarıyı azaltmıştır.
- Özellik seçimi metotları arasında en başarılısı GainRatio'dur ve özellik seçimi yapılmadığında elde edilen ortalama başarıdan daha yüksek başarı elde eden tek metottur.

Regresyon problemleri için veri kümesinin her bir özelliği için kümeleme tabanlı bir regresyon modeli oluşturan ve bunların sonuçlarını birleştiren çeşitli regresyon algoritmaları geliştirilmiştir. Algoritmalar 8 veri kümesi üzerinde popüler regresyon algoritmalarıyla karşılaştırılmıştır. Her bir özellik için elde edilen sonuçların birleştirilmesinde, ortalama alma gibi basit bir mekanizma kullanılmasına rağmen umut vaat eden sonuçlar elde edilmiştir. Bununla birlikte daha başarılı sonuçlar için önerilen yaklaşımın geliştirilmeye ihtiyacı vardır.

Regresyon komiteleri için, literatürdeki komite oluşturma, karar birleştirme metotlarının ve komitelerde yer alan regresyon algoritmalarının ilaç tasarımı veri kümelerinde performans üzerindeki etkileri incelenmiştir. Bu denemelerde elde edilen sonuçlar özetlenirse:

- Orijinal algoritmalar yerine komitelerin kullanılması genelde daha başarısız sonuçlar üretmiştir.
- Komite algoritmalarının başarı sıralaması Additive Regresyon > Orijinal > Özellik Seçimli Regresyon > Bagging = Rasgele Altuzay şeklindedir.
- En başarılı komite - algoritma ikilisi Additive Regresyon - RepTree'dir.
- En başarısız komite - algoritma ikilisi Bagging - PLS'dir.
- Sadece M5P ve PLS , 2 komitede (Additive Regresyon, Özellik Seçimli Regresyon) tek başına kullanıldıklarından daha başarılı sonuçlar vermiştir. Diğer algoritmaların performansları, komitelerle birlikte kullanıldıklarında daha düşüktür.
- Regresyon algoritmalarının başarı sıralaması RepTree > Basit Lineer Regresyon > KStar > M5P > PLS2 şeklindedir.
- Algoritmaların performanslarına göre birbirlerine benzerlikleri araştırıldığında komite türlerine göre değil komitelerde kullanılan algoritmalara göre gruplandıkları görülmüştür. Bu nedenle performansı komite türünün değil komitede kullanılan algoritmanın belirlediği söylenebilir.

Bütün veri kümelerinde diğer tüm algoritmalarından daha iyi sonuç veren global bir algoritma bulunmamaktadır. Bu nedenle, bir veri kümesinin hangi algoritma ile en iyi sonucu vereceği genelde deneme yanılma metoduyla bulunmaktadır. Literatürde bu eksikliği gidermek ve son kullanıcılara yardımcı kurallar dizisi oluşturabilmek için algoritmaların performanslarının veri kümesinin çeşitli özelliklerine göre tahmin edilmesi amacını taşıyan yaklaşımlar geliştirilmiştir. Bu yaklaşımların genel adı Meta-Öğrenim'dir. Mevcut Meta-öğrenim

çalışmalarında genelde sınıflandırma problemleri üzerine çalışılmıştır. İlaç veri kümelerindeki problemlerin büyük bir kısmı regresyon türünden problemler olduğu için bu çalışmada yeni bir **Meta-Regresyon** yaklaşımı da geliştirilmiştir. Geliştirilen yaklaşımda Meta-öğrenimde kullanılan standart veri kümesi özelliklerine ek yeni özellikler de kullanılmıştır. Çalışma sonunda bir veri kümesi üzerinde bir algoritmanın performansı veri kümesinin çeşitli özelliklerine bakarak tahmin edilebilen bir model geliştirilmiştir. Bu sayede bir veri kümesinde en iyi performansı gösterecek algoritma da tahmin edilebilmektedir. Ayrıca veri kümelerinin ve algoritmaların birbirlerine benzerliklerine göre kümelenmesi konusunda da çalışılmıştır. 3 farklı veri kümesi koleksiyonu (UCI, ilaç ve yapay) oluşturulmuştur. Veri koleksiyonlarında sırasıyla 60, 41 ve 80 veri kümesi yer almaktadır. Bu veri kümelerinin her birinin yaklaşık 300 meta özelliği elde edilmiştir. Bu meta özellikler kullanılan denemelerde elde edilen sonuçlar aşağıda özetlenmiştir.

Friedman koleksiyonunda elde edilen sonuçlar:

Yüksek korelasyonlu meta özellik ikilileri incelendiğinde aşağıdaki sonuçlara ulaşılmıştır:

- Örnek sayısı M5P, Reptree ve M5rules algoritmalarının hataları ile ters ilişkilidir. Diğer bir ifadeyle, örnek sayısı arttıkça algoritmaların performansı da artmaktadır.
- Colinearity derecesi skewness, kurtosis, 3. ve 4. dereceden momentlerle ilişkilidir. Bu ilişki colinearity derecesi bilinmeyen veri kümelerinin colinearity tahmininde kullanılabilir.
- Benzer karakteristiğe sahip algoritmaların performansları da benzerdir. Örneğin örnek tabanlı algoritmalar olan IBK ve Kstar, lineer tabanlı algoritmalar olma PLS, SVM ve lineer Regesyon benzer performanslar sergilemiştir.
- Veri kümelerinin genelde boyut, örnek sayısı ve özelliklerin birbiriyle korelasyonunun bir ölçütü olan colinearity'nin var olup olmamasına göre gruplandığı görülmektedir.

Meta özellikler kullanılarak algoritma performansları tahmininde aşağıdaki sonuçlara ulaşılmıştır:

- Performansı en iyi tahmin edilebilen algoritma meta.Bagging'tir. Bu algoritma aynı zamanda 80 friedman veri kümesi üzerinde en iyi performansa sahip algoritmadır. Bu özelliği, elde edilen performans tahmin başarısının önemini arttırmaktadır.
- Performans tahmininde özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

- Performans tahmini yapılırken üretilen kurallarda, RMSE meta özellikleri haricindeki meta özellik gruplarının yoğun bir şekilde seçildiği görülmektedir. RMSE grubundan seçilen tek meta özellik Decstump'tır.
- En fazla seçilen meta özellikler Kümeleme meta özellik grubuna aittir.
- Özelliklerin birbirleriyle ilişkilerinin derecelerini yansıtan colli ve ST2.corXdeg'de sıklıkla seçilmiştir.

İlaç veri koleksiyonunda elde edilen sonuçlar:

Yüksek korelasyonlu meta özellik ikilileri incelendiğinde aşağıdaki sonuçlara ulaşılmıştır:

- RepTree, RBF ve ConjunctiveRule algoritmalarının hataları ile çıkışların standart sapması doğru orantılıdır. Çıkışlar ne kadar dağınıksa bu algoritmalar o kadar çok hata yapmaktadır.
- Karar ağaçlarındaki kuralların kompleksliği ile verinin dağınıklığı doğru orantılıdır. Veri ne kadar dağınıksa, karar ağaçları o kadar kompleks kurallar üretmektedir.
- Algoritmaların benzerlikleri incelendiğinde, lineer karakteristiğe sahip algoritmaların bir kümede toplandığı ve algoritmaların birbirine çok uzak iki kümede toplandıkları görülmektedir.
- Veri kümelerinin performanslarına göre benzerlikleri incelendiğinde özellik sayısı büyük olan veri kümelerinin bir grupta toplandığı görülmektedir.

Meta özellikler kullanılarak algoritma performansları tahmininde aşağıdaki sonuçlara ulaşılmıştır:

- Algoritmaların performansları yapay veri kümesi koleksiyonundaki (friedman) gibi iyi bir şekilde tahmin edilememiştir
- Kstar haricindeki tüm hiçbir algoritmanın performansı 0.8'un üzerinde bir korelasyon değeriyle tahmin edilememiştir. Bunun sebebi olarak meta özelliklerle algoritmaların tahminleri arasında yüksek bir korelasyon olmaması söylenebilir.
- Performansı en iyi tahmin edilebilen algoritma Kstar'tır. Bu algoritma aynı zamanda 41 ilaç veri kümesi üzerinde Ortalama RMSE'lere göre en iyi performansa sahip algoritmadır. Bu özelliği, elde edilen performans tahmin başarısının önemini

arttırmaktadır.

- Özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

UCI veri koleksiyonunda elde edilen sonuçlar:

Yüksek korelasyonlu meta özellik ikilileri incelendiğinde:

- Rasgele hata ile çıkışların standart sapması doğru orantılıdır. Çıkışlar ne kadar savruksa, rasgele başarı o kadar azdır.
- RBF, ConjunctiveRule, PLS, DecisionStump ve LMS algoritmalarının hataları ile çıkışın standart sapması doğru orantılıdır. Çıkış ne kadar savruksa bu algoritmalar o kadar fazla hata yapmaktadır.
- Algoritmaların benzerlikleri incelendiğinde, benzer karakteristiğe sahip algoritmaların benzer performanslar sergilediği, MLP'nin ve SVM'in diğer tüm algoritmalarından farklı bir performans karakteristiği sergilediği görülmüştür.
- Veri kümelerinin performanslarına göre benzerlikleri incelendiğinde, performansça birbirine benzeyen veri kümelerinin örnek sayıları ve özellik sayılarına göre herhangi bir şablon içermedikleri görülmüştür.

Meta özellikler kullanılarak algoritma performansları tahmininde aşağıdaki sonuçlara ulaşılmıştır.

- MSP haricindeki tüm algoritmaların performansı 0.9'un üzerinde bir korelasyon değeriyle tahmin edilebilmiştir.
- Özellik seçiminin etkisi, tüm denemelerde olumlu olmuştur.

3 koleksiyonda yapılan araştırmalardan elde edilen sonuçlar birlikte incelendiğinde aşağıdaki sonuçlara ulaşılmıştır.

- Zero rule'a göre ortalama RMSE'si en yüksek veri kümeleri, üretilen yapay veri kümeleridir.
- Meta özelliklerin birbirleriyle korelasyonları incelendiğinde Friedman yapay veri koleksiyonunda 1707, UCI veri koleksiyonunda 655, ilaç veri koleksiyonunda 857 meta özellik ikilisinin korelasyonu mutlak değerce 0.8'den büyük olduğu görülmüştür. Bu sonuç göstermektedir ki, yapay veri kümeleri, gerçek veri kümelerine göre meta

özellikleri bakımından birbirlerine daha çok benzemektedir.

- Her veri koleksiyonunda en başarılı algoritma farklı bir algoritmadır, dolayısıyla hiçbir algoritmanın tüm veriler üzerinde en başarılı olmadığı görülmüştür.
- İlaç veri kümelerinde, algoritmaların rasgele hatayı (zero rule hatası) çok az düşürebildikleri görülmüştür. Bu nedenle en zor modellenen veri kümeleri oldukları söylenebilir.
- M5P algoritması her 3 veri kümesi koleksiyonunda da en iyi performans gösteren algoritmalar arasındadır.
- Bir algoritmanın veri kümesindeki hatası büyükse, onu tahmin etmek zordur. Başarılı sonuçları tahmin etmek daha kolaydır.
- Algoritma performans tahminlerinde en sık kullanılan meta özelliklerin REGT meta özellik grubundan, M5rules ile bulunan kural sayısının örnek sayısına oranı, M5P karar ağacının yapraklarında (kararlarında) en az 1 kere kullanılmış özellik sayısı, M5P karar ağacındaki yaprak sayısının örnek sayısına oranı, M5P karar ağacının yapraklarında en az 1 kere kullanılmış özellik sayısının özellik sayısına oranı, RMSE grubundan Decstump algoritmasının RMSE değeri, STA grubundan veri kümesinde cfs ile seçilen özellik sayısı ve veri kümesindeki örnek sayısı olduğu görülmüştür.

PLS analizlerinde elde edilen sonuçlar:

- Literatürde ilaç veri kümelerinde en çok kullanılan algoritma PLS'dir. Bu nedenle tezde, PLS'in bileşen sayısının ayrıntılı analizi de yapılmıştır. Bu analiz için 33 ilaç veri kümesinde minimum hataya sahip PLS denemelerinin K değerleri kaydedilmiştir. Daha sonra belirlenen optimum K değerlerinin meta özellikleriyle korelasyonu incelenmiştir. Bu inceleme sonunda, PLS algoritmasında kullanılacak optimum K değerinin incelenen hiçbir meta özellik ile birebir ilişkide olmadığı görülmüştür.
- PLS algoritmasının K değerinin PLS'in performansı üzerinde etkili olup olmadığı da araştırılmıştır. Bu incelemeler sonunda K değerinin çok az sayıda veri kümesi üzerinde büyük değişimlere yol açtığı görülmüştür. Bununla birlikte K sayısı bütün veri kümeleri için etkisiz değildir. Bu nedenle minimum hataya sahip olan K değerleri incelenmiş ve en fazla minimum hataya sahip K değerinin 10 veri kümesi ile 2 olduğu; onu 5'er veri kümesiyle 1 ve 5'in takip ettiği görülmüştür.

En başarılı algoritmanın tahmininde elde edilen sonuçlar:

- Tezin meta regresyon bölümünde ayrıca, bir veri kümesinin meta özelliklerine bakarak o veri kümesinde hangi algoritmanın daha başarılı olduğunun tahmin edilmesi çalışmaları da yapılmıştır. Bunun için 60 UCI veri kümesi kullanılmıştır. Veri kümesi iki farklı yaklaşımla etiketlenmiş ve problem bir sınıflandırma problemine dönüştürülmüştür. İlk yaklaşımda Veri kümeleri en başarılı algoritma isimleriyle etiketlenmiştir. 18 algoritmadan 11'i en başarılı algoritmalar kümesine girmiş ve bu 11 algortmada gruplanarak 7 algoritma ismine dönüştürülmüştür. Rasgele başarı %40 iken çeşitli sınıflandırma algoritmaları denenmiş ve en başarılı sonuç olarak SVM %53 başarı elde etmiştir. İkinci yaklaşımda ise, “bu algoritma bu veri kümesinde en yüksek başarıyı gösterir mi?” sorusuna cevap aranmış ve 2 sınıflı problemler üretilmiştir. En fazla sayıda en başarılı algoritma olan 3 algoritma için 3 ayrı veri kümesi oluşturulmuştur. Alınan sonuçlar aşağıda verilmiştir.
 - M5 için rasgele başarı %60, en başarılı sonuç %95
 - SVM için rasgele başarı %88, en başarılı sonuç %95
 - PLS için rasgele başarı %90, en başarılı sonuç %97
- Bu sonuçlar incelendiğinde ilk yaklaşımın çok iyi sonuçlar üretmediği, bunun yanında 2 sınıfla ifade edilen “bu algoritma bu meta özelliklere sahip veri kümesi için en yüksek başarıyı verir mi” problemine tatmin edici cevaplar verebildiği görülmektedir.
- Bu iki yaklaşımda en başarılı algoritmayı tahmin ederken kurallar üreten sınıflandırıcılarda en çok kullanılan özellikler REGT, STA grubundandır. Bu sonuç RMSE haricinde, algoritma performans tahminleriyle paralellik göstermektedir.

Özetlemek gerekirse bu tezde, makine öğrenmesinin çeşitli konularında birçok yeni yaklaşım geliştirilmiş ve bu konuda çalışan araştırmacılar ve son kullanıcılar için faydalı olacak sonuçlar ve veri kümeleri üretilmiştir. Bu tezin hem ilaç tasarımı hem de makine öğrenmesi konularında Türkiye’de ve Dünya’da yapılan çalışmalara katkıda bulunması dileğimizdir.

KAYNAKLAR

Abdi, H., (2003), "Partial least squares regression (PLS-regression)", In M. Lewis-Beck, A. Bryman, T. Futing (Eds): Encyclopedia for research methods for the social sciences. Thousand Oaks. pp. 792-795.

Abramowitz, M. and Stegun, I. A. (Eds.), (1972) "Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables", 9th printing. New York: Dover, p. 928.

Aha D. ve D. Kibler, (1991), "Instance-based learning algorithms", Machine Learning. 6:37-66.

Ali, S. ve Smith, K. A., (2006), "On Learning Algorithm Selection for Classification" Applied Soft Computing, Elsevier Science, vol.6(2), pp.119-138.

Alpaydın, E., (2004), "Introduction to Machine Learning", The MIT Press, 3-6.

Arciniegas F., Bennett K., Breneman C. M., Embrechts M. J., (2000) "Molecular Database Mining using Self-Organizing Maps for the Design of Novel Pharmaceuticals," Intelligent Engineering Systems through Artificial Neural Networks: Smart Engineering System Design: Vol. 10, ASME Press, 477 – 482.

Banfield R. E., Lawrence O.Hall, Kevin W.Bowyer ve Divya Bhadoria, (2004), "A comparison of Ensemble Creation Techniques", LNCS 3077, pp. 223-232.

Baykara T., Çaylı H., Çelik H., Tokat M., Ünalın T., Ankara, (2003), "Türkiye’de İlaçta Veri Koruması Ve Uygulanmasının Mali Etkileri" adlı rapor, http://www.aifd.org.tr/detay.asp-ID=75&db=sektorun_sorunlari.htm adresinden erişilebilir.

Bensusan H. ve Giraud-Carrier Christophe, (2000a), "Building Automatic Advice Strategies for Model Selection and Method Combination", Eleventh European Conference on Machine Learning, Workshop on Meta-Learning, Barcelona, Spain.

Bensusan H., C. Giraud-Carrier, C. J. Kennedy, (2000b), "A Higher-order Approach to Meta-Learning", Proceedings of the Work-in-Progress Track at the 10th International Conference on Inductive Logic Programming.

Bensusan H. ve A. Kalousis, (2001), "Estimating the Predictive Accuracy of a Classifier", Proceedings of the 12th European Conference on Machine Learning, pp. 25-36.

Bezdek J. C., (1973), "Fuzzy Mathematics in Pattern Classification", PhD Thesis, Applied Math. Center, Cornell University.

Blake, C., ve Merz, C., (1998), "UCI Repository of machine learning databases", <http://www.ics.uci.edu/mllearn/MLRepository.html>

Breiman L., (1999), "Random forests-random features", Technical Report 567, Department of Statistics, University of California, Berkeley.

Breiman L., (2001), "Random Forests", Machine Learning 45 (1), 5-32.

Breiman L., J. Friedman, R. Olshen, ve C. Stone, (1984), "Classification and Regression Trees", Chapman and Hall, New York, NY.

Brazdil P.B., Soares C. ve Costa J.P, (2003), "Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results", Machine Learning, Volume 50, Number 3, pp. 251-277(27).

- Carrier, C. G., (2005), "The Data Mining Advisor: Meta-learning at the Service of Practitioners", In Proceedings of the 4th International Conference on Machine Learning Applications, 113-119.
- Cleary G. ve Leonard E. Trigg, (1995), "K*: An Instance-based Learner Using an Entropic Distance Measure", In: 12th International Conference on Machine Learning, 108-114.
- Cover, T.T. ve Hart, P.E. (1967), "Nearst Neighbour Pattern Classification", IEEE Transactions on Information Theory 11, pp.21-27.
- Davies D.L., Bouldin D.W., (1979) "A cluster separation measure", IEEE Trans. Pattern Anal. Machine Intell., 224-227.
- Derek G., Alexey Tsymbal, Nadia Bolshakova ve Padraig Cunningham, (2004), "Ensemble Clustering in Medical Diagnostics", Proceedings of the 17th IEEE Symposium on Computer-Based Medical Systems (CBMS'04), p. 576.
- DeWitt A., Bayram S., Enciso G., Fernando H., Kao J., Pagnoncelli B., Schmidt D., Hameed J.A.S., (2004), "Data to Knowledge in Pharmaceutical Research", Topics and Descriptions for the 2004 Mathematical Modeling in Industry - A Workshop for Graduate Students –USA.
- Dietterich T. G., (1999), "An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, Randomization", Machine Learning, pp. 1-22.
- Embrechts M. J., Breneman C., Bennett K. P., (2002) "StripMining for Molecules", IEEE International Joint Conference on Neural Networks, IJCNN'02, Honolulu, Hawaii.
- Fern X. Z. ve C. E. Brodley, (2004), "Solving cluster ensemble problems by bipartite graph partitioning", ACM International Conference Proceeding Series; Vol. 69(36).
- Finn P., Muggleton S., Page D., Srinivasan A., (1998) "Pharmacophore Discovery using the Inductive Logic Programming System PROGOL", Machine Learning, 30, 241-273.
- Fisher, R.A., (1936) The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7: 179-188.
- Frank E., Mark Hall ve Bernhard Pfahringer, (2003), "Locally Weighted Naive Bayes", In: 19th Conference in Uncertainty in Artificial Intelligence, 249-256.
- Freund Y. ve R. E. Schapire, (1997), "A decision-theoretic generalization of on-line learning and an application to boosting", *Journal of Computer and System Sciences*, no. 55.
- Friedman J. H., (1999), "Stochastic Gradient Boosting", Erişim: <http://www-stat.stanford.edu/~jhf/ftp/stobst.ps>
- Fürnkranz J. ve Johann Petrak, (2001), "An evaluation of landmarking variants", Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning.
- Gama J. ve P. Brazdil, (1995), "Characterization of Classification Algorithms", 7th Portuguese Conference on Artificial Intelligence-EPIA.
- Gates, G.W., (1972), "The Reduced Nearst Neighbour Rule", IEEE Transactions on Information Theory 18, pp.431-433.
- Geneste N.M., Watson K.A., Alsberg B.K., King R.D., (2002) "New Approach to

Pharmacophore Mapping and QSAR Analysis Using Inductive Logic Programming. Application to Thermolysin Inhibitors and Glycogen Phosphorylase b Inhibitors", J. Med. Chem., 45, 399-409.

Guyon, I., Weston, J., Barnhill, S., ve Vapnik, V., (2002), "Gene selection for cancer classification using support vector machines", Machine Learning, 46, pp. 389-422.

Hall M. A., (1998), "Correlation-based Feature Subset Selection for Machine Learning", Hamilton, New Zealand.

Hallborn J., Carlsson. R., (2002) "Automated screening procedure for high-throughput generation of antibody fragments", BioInvent Therapeutic AB, 33:530-537.

Hart, P.E., (1968), "The Condensed Nearest Neighbour Rule", IEEE Transactions on Information Theory 14, pp.515-516.

Ho T. K., (1998), "The Random Subspace Method for Constructing Decision Forests", IEEE Transactions on Pattern Analysis and Machine Intelligence. 20(8):832-844.

Hochbaum, Shmoys, (1985) "Farthest First Traversal Algorithm: A best possible heuristic for the k-center problem", Mathematics of Operations Research, 10(2) 180-184.

Jaynes, E.T., (1957), "Information Theory and Statistical Mechanics", Physical Review, Vol.106, No.4, pp.620-630.

Kalousis, A. ve Hilario, M., (2000), "Model Selection via Meta-learning: a Comparative Study", Tools with Artificial Intelligence-ICTAI. Proceedings. 12th IEEE International Conference.

Kalousis, A. ve Hilario, M., (2003), "Representational Issues in Meta-Learning", In Proceedings of the 20th International Conference on Machine Learning -ICML.

Kewley R., Embrechts M. J., Breneman C., (1999) "A Soft Computing Approach for the Design of Novel Pharmaceuticals", Transactions of the 1999 IEEE Midwest-Sun Workshop on Soft Computing Methods in Industrial Applications, Kuusamo, Finland.

Kohonen T., (2000), "Self Organizing Maps", Springer Series in Information Sciences, p 245

Kohonen,T., (1990) "The self-organizing map", Proceedings of the IEEE, 78(9), 1464-1480.

Langdon W.B., Barrett S.J., Buxton B. F., (2003) "Predicting Biochemical Interactions - Human P450 2D6 Enzyme Inhibition", CEC 2003, Canberra, 807-814.

Loh W. Y. ve Y. S. Shih, (1997), "Split selection methods for classification trees", Statistica Sinica, 7:815-840.

Malerba D., Floriana Esposito, Michelangelo Ceci ve Annalisa Appice, (2004), "Top-Down Induction of Model Trees with Regression and Splitting Nodes", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.26.5, pp.612 - 625.

Mamitsuka H., (2003) "Empirical Evaluation of Ensemble Feature Subset Selection Methods for Learning from a High-Dimensional Database in Drug Design", Proceedings of the Third IEEE Symposium on BioInformatics and BioEngineering (BIBE'03).

Ozdemir M., Embrechts M. J., Arciniegas F., Breneman C., Lockwood L., Bennett K., (2001), "Feature Selection for In-silico Drug design Using Genetic Algorithms and Neural Networks", Proceedings, 2001 SMCia Mountain Workshop on Soft Computing in Industrial

Applications, IEEE Press, Blacksburg - Virginia, 53-57.

Pal, M., ve Watanachaturaporn, P., (2004), "Support vector machines", In P. K. Varshney & M. K. Arora (Eds.), Advanced image processing techniques for remotely sensed hyperspectral data: Springer-Verlag.

Peng, P. Y., A. Flach, P. Brazdil ve C. Soares, (2002), "Decision tree-based characterization for meta-learning", ECML/PKDD'02 workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-Learning.

Rousseeuw P.J., (1987) "Silhouettes: a graphical aid to the interperation and validation of cluster analysis", Journal of Computational and Applied Mathematics, 20, 53-65.

Shevade S. K., S.S. Keerthi, C. Bhattacharyya, ve K.R.K. Murthy, (1999), "Improvements to smo algorithm for svm regression".

Sonka M., Grunkin M., (2002), "Image Processing and Analysis in Drug Discovery and Clinical Trials", Ieee Transactions On Medical Imaging, 21(10).

Srinivasan A., King. R.D., (1999) "Using Inductive Logic Programming to construct Structure-Activity Relationships", AAAI Press, Menlo Park, CA, 64-73.

Strehl A. ve J. Ghosh, (2002), "Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions", Journal of Machine Learning Research, vol.3 , 583-617.

Tjen-Sien Lim, Wei-Yin Loh ve Yu-Shan Shih, (2000), "A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms", Machine Learning, vol.40, 203-229.

Todorovski L. ve S. Dzeroski, (1999), "Experiments in meta-level learning with ILP", Proceedings of third European Conference on Principles of data mining and knowledge discovery –PKDD.

Todorovski L., H. Blockeel ve S. Dzeroski, (2002a), "Ranking with predictive clustering trees", In Proceedings of the 13th European Conference on Machine Learning, volume 2430 of Lecture Notes in Artificial Intelligence, pages 444-455. SpringerVerlag.

Todorovski L., P. Brazdil ve C. Soares, (2002b), "Experiments with Automatic Feature Selection in Meta-Learning", Technical Report Jozef Stefan Institute.

Wang Z., Durst G.L., Eberhart R.C., Boyd D.B., Miled Z.B., (2004), "Particle Swarm Optimization and Neural Network Application for QSAR", 18th International Parallel and Distributed Processing Symposium (IPDPS'04), Santa Fe, New Mexico.

Wang, Y., Witten, I.H, (1997), "Inducing model trees for continuous classes", Proc of Poster Papers, 9 th European Conference on Machine Learning.

Weston J., Cruz F.P., Bousquet O., Chapelle O., Elisseeff A., Schölkopf B., (2003) "Feature selection and transduction for prediction of molecular bioactivity for drug design", Bioinformatics200319:764-771.

Witten I. H. ve Eibe Frank, (2005), "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco.

Wolpert, D.H. ve Macready, W.G., (1995), "No Free Lunch Theorems for Search", Technical Report SFI-TR-95-02-010, Santa Fe Institute.

Yıldız O. Taner, Alpaydın Ethem, (2005), “Linear Discriminant Trees”, International Journal of Pattern Recognition and Artificial Intelligence, 19, 323:353.

INTERNET KAYNAKLARI

- [1] [http:// staff.science.nus.edu.sg /~scilooe/ srp_2003/ sci_paper/cs/ research_paper/ ong_chin_siang.pdf](http://staff.science.nus.edu.sg/~scilooe/srp_2003/sci_paper/cs/research_paper/ong_chin_siang.pdf)
- [2] <http://www.cs.wisc.edu/~dpage/cs731/cibm3.ppt>
- [3] <http://www2.sas.com/proceedings/sugi30/078-30.pdf>
- [4] <http://antoine.frostburg.edu/chem/senese/101/glossary/m.shtml#molecule>
- [5] <http://www.cs.mdx.ac.uk/staffpages/serengul/The.ID3.algorithm.htm>
- [6] <http://www.deu.edu.tr/userweb/mustafa.ozel/dosyalar/Uzay%20Analitik %20Geometri.ppt>
- [7] Vishakh, “A Comparison of Ensemble Methods for Microarray Data Analysis”, 2006, http://www.vishakh.com/research/bagboost/ensemble_methods_comparison.pdf
- [8] <http://www.mathworks.com/access/helpdesk/help/toolbox/stats/linkage.html>
- [9] <http://www.lans.ece.utexas.edu/~strehl/diss/node28.html>
- [10] http://www.statisticalengineering.com/curse_of_dimensionality.htm
- [11] [http:// www.java2s.com/ Open-Source/ Java-Document/ Science/ weka/ weka/ filters/ unsupervised/attribute/InterquartileRange.java.htm](http://www.java2s.com/Open-Source/Java-Document/Science/weka/weka/filters/unsupervised/attribute/InterquartileRange.java.htm)
- [12] www.liacs.nl/home/joost/DM/mod_11_eval_lift_cost.ppt
- [13] <http://www.cmpe.boun.edu.tr/~ethem/i2ml/slides/v1-1/i2ml-chap9-v1-1.ppt>
- [14] <http://datamining.ihe.nl/research/model-trees.htm>
- [15] <http://ucsub.colorado.edu/~reids/papers/classifier-combination-reid-fall-07.pdf>
- [16] <http://www.chemcomp.com/>

EKLER

- Ek 1 “IEEE Transactions on Neural Networks” dergisinde Şubat 2008’de yayınlanmak üzere kabul edilen “Cline: A New Decision Tree Family” isimli makalenin özeti
- Ek 2 “International Symposium on Innovations in Intelligent Systems and Applications – INISTA” sempozyumunda 2005’te yayınlanan “Clusline: A New Clusternig Algorithm” isimli bildirinin özeti
- Ek 3 “ICSC Congress on Computational Intelligence Methods and Applications - CIMA” sempozyumunda 2005’te yayınlanan “Cline: New Multivariate Decision Tree Construction Heuristics” isimli bildirinin özeti

Ek 1 Cline: A New Decision Tree Family

Abstract. A new family of algorithm called Cline that provides a number of methods to construct and use multivariate decision trees is presented. We report experimental results for two types of data: synthetic data to visualize the behaviour of the algorithms, and publicly available 8 datasets. The new methods have been tested against 23 other decision tree construction algorithms based on benchmark datasets. Empirical results indicate that our approach achieves better classification accuracy compared to other algorithms.

Ek 2 Clusline: A New Clustering Algorithm

Abstract. Unsupervised learning (clustering) algorithms are generally used in discovering data structure and data compression by generating clusters of data. In this work a new such algorithm -ClusLine- is proposed. ClusLine is based on determining cluster centers by constructing a tree which divides data space into subspaces that have smaller standard deviation than all data's standard deviation. The algorithm is running on 14 real world dataset and compared popular clustering algorithms according to three cluster quality measures (Davies-Bouldin Index, Silhouette width and classification accuracy). According to quality measures, The Clusline algorithm can be used in all types of clustering problems because of its simplicity and acceptable performance.

Ek 3 Cline: New Multivariate Decision Tree Construction Heuristics

Abstract. Decision trees are often used in pattern recognition and regression problems. They are attractive due to high performance and easy-to-understand rules. Many different decision tree construction algorithms have been developed because of their popularity. In this work, we describe some new heuristic tree construction algorithms and test with 8 benchmark datasets. We compare the new method with other 21 tree induction algorithms. The results show that cline heuristics can be used in all types of classification problems because of its simplicity and acceptable performance.

ÖZGEÇMİŞ

Doğum tarihi	17.01.1978	
Doğum yeri	İstanbul	
Lise	1991-1994	Fatih Gelenbevi Lisesi
Lisans	1997-2001	Yıldız Teknik Üniversitesi Elektrik Elektronik Fak. Bilgisayar Mühendisliği Bölümü
Yüksek Lisans	2001-2003	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Müh. Anabilim Dalı, Bilgisayar Müh. Programı
Doktora	2003-2008	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Müh. Anabilim Dalı, Bilgisayar Müh. Programı

Çalıştığı kurum(lar)

2001-Devam ediyor YTÜ Elektrik Elektronik Fak. Bilgisayar Müh
Araştırma Görevlisi