T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

BÜYÜK VERİ ANALİZİ İÇİN YENİ ALGORİTMALAR

Hazırlayan
Burak GÜLMEZ

Danışman
Prof. Dr. Sinem KULLUK

Doktora Tezi

Ocak 2021
KAYSERİ

T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

BÜYÜK VERİ ANALİZİ İÇİN YENİ ALGORİTMALAR

(Doktora Tezi)

Hazırlayan
Burak GÜLMEZ

Danışman
Prof. Dr. Sinem KULLUK

Ocak 2021
KAYSERİ

BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.

Burak GÜLMEZ

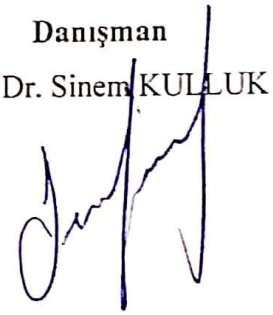
YÖNERGEYE UYGUNLUK

"Büyük Veri Analizi için Yeni Algoritmalar" adlı Doktora tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi'ne uygun olarak hazırlanmıştır.

Hazırlayan
Burak GÜLMEZ

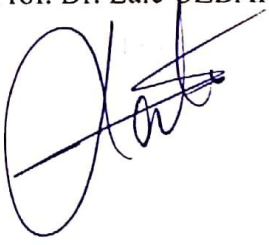


Danışman
Prof. Dr. Sinem KULLUK



Endüstri Mühendisliği ABD Başkanı

Prof. Dr. Lale ÖZBAKIR



KABUL VE ONAY

Prof. Dr. Sinem KULLUK danışmanlığında **Burak GÜLMEZ** tarafından hazırlanan **"Büyük Veri Analizi için Yeni Algoritmalar"** adlı bu çalışma jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalında **Doktora** tezi olarak kabul edilmiştir.

08/01/2021

JÜRİ:

Danışman : Prof. Dr. Sinem KULLUK .....

Üye : Prof. Dr. Lale ÖZBAKIR .....

Üye : Prof. Dr. İbrahim DOĞAN .....

Üye : Prof. Dr. Erkan KÖSE .....

Üye : Dr. Öğr. Üyesi Oğuzhan Ahmet ARIK .....

ONAY:

Bu tezin kabulü Enstitü Yönetim Kurulunun ..... tarih ve .....sayılı kararı ile onaylanmıştır.

...../...../.....

Prof. Dr. Mehmet AKKURT

Enstitü Müdürü

TEŞEKKÜR

Çalışmalarım boyunca benden yardımını esirgemeyen, görüşleri ve katkılarıyla beni yönlendiren, tecrübelerinden faydalandığım başta hocam Prof. Dr. Sinem KULLUK olmak üzere üzerimde emeği bulunan herkese teşekkür ederim.

Burak GÜLMEZ

Ocak 2021, KAYSERİ

BÜYÜK VERİ ANALİZİ İÇİN YENİ ALGORİTMALAR

Burak GÜLMEZ

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Ocak 2021
Danışman: Prof. Dr. Sinem KULLUK

ÖZET

Günümüzde veri toplamak, veri depolamak geçmişe göre daha kolay ve basit bir işlemdir. Veri miktarının büyük olması veri üzerinden elde edilen bilgileri ve analizleri daha sağlıklı bir hale getirmektedir, fakat büyük veri ile baş edebilmek zorlaşmaktadır. Geleneksel veri madenciliği yöntemleriyle büyük veri analizi hem teknik olarak hem de süre olarak imkânsız bir hale gelmektedir. Bu probleme çözüm olarak veriyi parçalamak ve veri parçaları üzerinde paralel işlemler yapmak gerekmektedir. Bunun için genellikle Apache Hadoop ve Apache Spark kullanılmaktadır. Apache Spark, Hadoop'a göre hız ve kapsam bakımından üstünlük sağlamaktadır. Büyük veriler üzerinde analiz için Spark'ın kendi içerisinde makine öğrenme kütüphanesi bulunmaktadır. Bu tez çalışmasında büyük veri analizi için iki yeni algoritma önerilmiştir. Bu algoritmalar Spark Hibrit (SH) ve Yapay Sinir Ağı – Karar Ağacı (YSA-KA) algoritmalarıdır. Bu yeni algoritmaların performansı, hem Pokerhand, Susy, Higgs, Hepmass, Wisdm ve Census literatür sınıflandırma veri kümeleri ile, hem de gerçek hayat verisi olan üretim ve ikinci el araba veri kümeleri üzerinde test edilmiştir. Önerilen algoritmalarla elde edilen sonuçlar Spark içerisindeki makine öğrenme algoritmaları ve literatürde önerilen algoritma sonuçlarıyla karşılaştırılmıştır. İstatistiksel analizler ile algoritmaların sonuçları arasındaki farkların anlamlı olup olmadığı istatistiksel olarak test edilmiş ve yorumlanmıştır. Tez çalışmasında önerilen algoritmalardan YSA-KA algoritması Spark'taki ve literatürde son yıllarda yapılan çalışmalardaki algoritmalara göre genel olarak daha iyi sonuç vermiştir. SH algoritması ise diğer algoritmalarla rekabet edebilecek düzeydedir.

Anahtar Kelimeler: Büyük veri, Makine öğrenme, Spark, Sınıflandırma, Fonksiyon tahminleme

NEW ALGORITHMS FOR BIG DATA ANALYSIS

Burak GÜLMEZ

Erciyes University, Graduate School of Natural and Applied Sciences
PhD Thesis, January 2021
Supervisor: Prof. Dr. Sinem KULLUK

ABSTRACT

Today, collecting and storing data is an easier and simpler process than in the past. The big amount of data makes the knowledge and analysis obtained through the data more accurate, but it becomes difficult to cope with big data. With traditional data mining methods, big data analysis becomes impossible both technically and in terms of time. As a solution to this problem, it is necessary to split the data and perform parallel operations on the data splits. Generally, Apache Hadoop and Apache Spark are used for this. Apache Spark outperforms Hadoop in terms of speed and scope. Spark has its own machine learning library for analysis on big data. In this thesis, two new algorithms are proposed for big data analysis. These algorithms are Spark Hybrid (SH) and Artificial Neural Network-Decision Tree (ANN-DT) algorithms. The performance of these new algorithms has been tested on both Pokerhand, Susy, Higgs, Hepmass, Wisdm, and Census literature classification datasets, as well as production and secondhand car real-life datasets. The results obtained with the proposed algorithms were compared with the machine learning algorithms in Spark and the algorithms suggested in the literature. whether the differences between the results of the compared algorithms were significant was tested and interpreted statistically, using statistical analysis. One of the algorithms suggested in the thesis study, the ANN-DT algorithm has generally yielded better results than the algorithms in Spark and the algorithms in the recent studies in the literature. The SH algorithm is able to compete with other algorithms.

Keywords: Big data, Machine learning, Spark, Classification, Function estimation

İÇİNDEKİLER

BÜYÜK VERİ ANALİZİ İÇİN YENİ ALGORİTMALAR

BİLİMSEL ETİĞE UYGUNLUK ..... **ii**

YÖNERGEYE UYGUNLUK ..... **iii**

KABUL VE ONAY ..... **iv**

TEŞEKKÜR ..... **v**

ÖZET ..... **vi**

ABSTRACT ..... **vii**

İÇİNDEKİLER ..... **viii**

KISALTMALAR ..... **xi**

TABLOLAR LİSTESİ ..... **xii**

ŞEKİLLER LİSTESİ ..... **xiv**

GİRİŞ **1**

1. BÖLÜM **4**

LİTERATÜR ÇALIŞMASI **4**

1.1. Sınıflandırma Literatürü **4**

1.2. Kümeleme Literatürü **9**

1.3. Birliktelik Analizi Literatürü **11**

1.4. Zaman Serileri Literatürü **12**

1.5. Metin Madenciliği Literatürü **13**

1.6. Diğer Literatür **14**

2. BÖLÜM **18**

BÜYÜK VERİ **18**

2.1. Veri Madenciliği **18**

2.1.1. Veri Tabanından Bilgi Keşfi 19

2.2. Büyük Veri **21**

2.3. Büyük Veri Sistemleri **22**

2.3.1. Apache Hadoop....22
2.3.2. Apache Spark....24
2.3.3. Doğrusal Regresyon....28
2.3.4. Lojistik Regresyon....28
2.3.5. İzotonik Regresyon....29
2.3.6. Karar Ağacı....29
2.3.7. Doğrusal Destek Vektör Makinesi....31
2.3.8. Rastgele Orman....32
2.3.9. Yapay Sinir Ağları....33
3. BÖLÜM....40
BÜYÜK VERİ ANALİZİ İÇİN ÖNERİLEN ALGORİTMALAR....40
3.1. Spark Hibrit Algoritma (SH)....40
3.2. Yapay Sinir Ağı ve Karar Ağacı Algoritması (YSA-KA)....45
4. BÖLÜM....51
DENEYSEL ÇALIŞMA VE ANALİZLER....51
4.1. Literatür Verileri Üzerindeki Çalışmalar....51
4.1.1. Literatür Veri Kümeleri....51
4.1.2. Etkinlik Ölçütü ve Veri Kümelerinin Eğitim-Test Kümelerine Parçalanması....54
4.1.3. Deney Tasarımı....55
4.1.4. SH ve YSA-KA Algoritmaları ile Elde Edilen Sonuçlar....90
4.1.5. SH ve YSA-KA Algoritmaları Performans Karşılaştırmaları....91
4.1.6. İstatistiksel Analiz....95
4.2. Gerçek Hayat Veri Kümeleri Üzerindeki Çalışmalar....99
4.2.1. Gerçek Hayat Veri Kümeleri....99
4.2.2. Etkinlik Ölçütleri ve Veri Kümelerinin Eğitim-Test Kümelerine Parçalanması....100
4.2.3. Deney Tasarımı....101

4.2.4. SH ve YSA-KA Algoritmaları ile Elde Edilen Sonuçlar.....114
4.2.5. SH ve YSA-KA Algoritmaları Performans Karşılaştırmaları 115
4.2.6. İstatistiksel Analiz.....116
5. SONUÇLAR.....119
KAYNAKÇA.....121
ÖZGEÇMİŞ.....130

KISALTMALAR

Spark	: Apache Spark
Hadoop	: Apache Hadoop
YSA	: Yapay sinir ağı
KA	: Karar ağacı
DR	: Doğrusal regresyon
LR	: Lojistik regresyon
RO	: Rastgele orman
İR	: İzotonik regresyon
DDVM	: Doğrusal Destek Vektör Makinesi
SH	: Spark hibrit algoritması
YSA-KA	: Yapay sinir ağı karar ağacı algoritması

TABLOLAR LİSTESİ

Tablo 3.1 Doğrusal regresyon algoritması sözde kodu 41
Tablo 3.2 Lojistik regresyon algoritması sözde kodu 41
Tablo 3.3 İzotonik regresyon algoritması sözde kodu 41
Tablo 3.4 Karar ağacı algoritması sözde kodu 42
Tablo 3.5 Doğrusal destek vektör makinesi algoritması sözde kodu 42
Tablo 3.6 Rastgele orman algoritması sözde kodu 42
Tablo 3.7 Yapay sinir ağları algoritması sözde kodu 43
Tablo 3.8 SH algoritması parametreleri 44
Tablo 3.9 SH algoritması sözde kodu 45
Tablo 3.10 Yapay sinir ağları algoritması sözde kodu 46
Tablo 3.11 Karar ağacı algoritması sözde kodu 47
Tablo 3.12 YSA-KA algoritması parametreleri 49
Tablo 3.13 YSA-KA algoritması sözde kodu 50
Tablo 4.1 Pokerhand veri kümesi özellikleri 52
Tablo 4.2 Susy veri kümesi özellikleri 52
Tablo 4.3 Higgs veri kümesi özellikleri 53
Tablo 4.4 Hepmass veri kümesi özellikleri 53
Tablo 4.5 Wisdm veri kümesi versiyonları ve özellikleri 54
Tablo 4.6 Census veri kümesi özellikleri 54
Tablo 4.7 SH algoritması parametreleri ve parametre düzeyleri 57
Tablo 4.8 Pokerhand veri kümesi varyans analizi 57
Tablo 4.9 Susy veri kümesi varyans analizi 60
Tablo 4.10 Higgs veri kümesi varyans analizi 63
Tablo 4.11 Hepmass veri kümesi varyans analizi 66
Tablo 4.12 Wisdm veri kümesi varyans analizi 68
Tablo 4.13 Census veri kümesi varyans analizi 71
Tablo 4.14 YSA-KA algoritması deney tasarımı parametreleri 74
Tablo 4.15 Pokerhand veri kümesi 75
Tablo 4.16 Susy veri kümesi varyans analizi 78
Tablo 4.17 Higgs veri kümesi varyans analizi 81
Tablo 4.18 Hepmass veri kümesi varyans analizi 83
Tablo 4.19 Wisdm veri kümesi varyans analizi 86

Tablo 4.20 Census veri kümesi varyans analizi 88
Tablo 4.21 Literatür verileri üzerinde yeni algoritma sonuçları 91
Tablo 4.22 Literatür verileri üzerinde algoritmaların karşılaştırılması 92
Tablo 4.23 Literatür çalışmaları ile karşılaştırmalar 94
Tablo 4.24 Algoritmaların istatistiksel karşılaştırılması 96
Tablo 4.25 SH algoritması ile literatür verileri istatistiksel analizi 97
Tablo 4.26 YSA-KA algoritması ile literatür verileri istatistiksel analizi 98
Tablo 4.27 Üretim veri kümesi özellikleri 99
Tablo 4.28 İkinci el araba veri kümesi özellikleri 100
Tablo 4.29 Üretim veri kümesi varyans analizi 102
Tablo 4.30 İkinci el araba veri kümesi varyans analizi 105
Tablo 4.31 YSA-KA algoritması parametreleri ve parametre düzeyleri 109
Tablo 4.32 Üretim veri kümesi varyans analizi 109
Tablo 4.33 İkinci el araba veri kümesi varyans analizi 112
Tablo 4.34 Gerçek hayat verileri üzerinde yeni algoritma sonuçları 115
Tablo 4.35 Üretim veri kümesi algoritma sonuçları 115
Tablo 4.36 İkinci el araba veri kümesi algoritmaların RMSE sonuçları 116
Tablo 4.37 Üretim veri kümesi için algoritmaların istatistiksel karşılaştırılması 117
Tablo 4.38 İkinci al araba veri kümesi için algoritmaların istatistiksel karşılaştırılması 117

ŞEKİLLER LİSTESİ

Şekil 2.1 Veri madenciliği süreci [73] 19
Şekil 2.2 Veri tabanından bilgi keşfi süreci [76] 20
Şekil 2.3 Büyük verinin temel üç özelliği [80] 21
Şekil 2.4 HDFS [85] 23
Şekil 2.5 MapReduce örneği [87] 24
Şekil 2.6 Hadoop ve Spark hız karşılaştırması [89] 25
Şekil 2.7 Spark bileşenleri [90] 25
Şekil 2.8 Spark Streaming veri kaydetme [93] 26
Şekil 2.9 Bir graf örneği [95] 27
Şekil 2.10 Basit doğrusal regresyon [96] 28
Şekil 2.11 İzotonik regresyon [97] 29
Şekil 2.12 Karar ağacı örneği [98] 31
Şekil 2.13 Destek vektör makinesi [99] 31
Şekil 2.14 Destek vektör makinesi alternatif vektörler [100] 32
Şekil 2.15 Rastgele orman algoritması 32
Şekil 2.16 Yapay sinir hücresi [103] 33
Şekil 2.17 Sigmoid fonksiyonu [104] 35
Şekil 2.18 Hiperbolik tanjant fonksiyonu [105] 35
Şekil 2.19 Yapay sinir ağlarının yapısı 36
Şekil 2.20 a) İleri besleme b) Geri besleme [106] 37
Şekil 3.1 Dört farklı algoritma ile SH algoritması 43
Şekil 3.2 SH algoritması oylama ve sonuç elde etme aşaması 44
Şekil 3.3 Yapay sinir ağı ve karar ağacı algoritması 49
Şekil 4.1 Pokerhand veri kümesi pareto grafiği 58
Şekil 4.2 Pokerhand veri kümesi ana etkiler grafiği 58
Şekil 4.3 Pokerhand veri kümesi ikili etkileşim grafiği 59
Şekil 4.4 Susy veri kümesi standartlaştırılmış etkiler pareto grafiği 60
Şekil 4.5 Susy veri kümesi ana etkiler grafiği 61
Şekil 4.6 Susy veri kümesi ikili etkileşim grafiği 62
Şekil 4.7 Higgs veri kümesi standartlaştırılmış etkiler pareto grafiği 63
Şekil 4.8 Higgs veri kümesi ana etkiler grafiği 64
Şekil 4.9 Higgs veri kümesi ikili etkileşim grafiği 65

Şekil 4.10 Hepmass veri kümesi standartlaştırılmış etkiler pareto grafiği 66
Şekil 4.11 Hepmass veri kümesi ana etkiler grafiği..... 67
Şekil 4.12 Hepmass veri kümesi ikili etkileşim grafiği 68
Şekil 4.13 Wisdm veri kümesi standartlaştırılmış etki pareto grafiği..... 69
Şekil 4.14 Wisdm veri kümesi ana etkiler grafiği..... 70
Şekil 4.15 Wisdm veri kümesi ikili etkileşim grafiği 70
Şekil 4.16 Census veri kümesi standartlaştırılmış etki pareto grafiği..... 72
Şekil 4.17 Census veri kümesi ana etkiler grafiği..... 72
Şekil 4.18 Census veri kümesi ikili etkileşim grafiği 73
Şekil 4.19 Pokerhand veri kümesi standartlaştırılmış etkiler pareto grafiği..... 75
Şekil 4.20 Pokerhand veri kümesi ana etkiler grafiği 76
Şekil 4.21 Pokerhand veri kümesi ikili etkileşim grafiği..... 77
Şekil 4.22 Susy veri kümesi standartlaştırılmış etkiler pareto grafiği 78
Şekil 4.23 Susy veri kümesi ana etkiler grafiği..... 79
Şekil 4.24 Susy veri kümesi ikili etkileşim grafiği 80
Şekil 4.25 Higgs veri kümesi standartlaştırılmış etkiler pareto grafiği..... 81
Şekil 4.26 Higgs veri kümesi ana etkiler grafiği..... 82
Şekil 4.27 Higgs veri kümesi ikili etkileşim grafiği 83
Şekil 4.28 Hepmass veri kümesi standartlaştırılmış etkiler pareto grafiği 84
Şekil 4.29 Hepmass veri kümesi ana etkiler grafiği..... 84
Şekil 4.30 Hepmass veri kümesi ikili etkileşim grafiği 85
Şekil 4.31 Wisdm veri kümesi standartlaştırılmış etkiler pareto grafiği..... 86
Şekil 4.32 Wisdm veri kümesi ana etkiler grafiği..... 87
Şekil 4.33 Wisdm veri kümesi ikili etkileşim grafiği 87
Şekil 4.34 Census veri kümesi standartlaştırılmış etkiler pareto grafiği..... 89
Şekil 4.35 Census veri kümesi ana etkiler grafiği..... 89
Şekil 4.36 Census veri kümesi ikili etkileşim grafiği 90
Şekil 4.37 Üretim veri kümesi standartlaştırılmış etkiler pareto grafiği..... 103
Şekil 4.38 Üretim veri kümesi ana etkiler grafiği 104
Şekil 4.39 Üretim veri kümesi ikili etkileşim grafiği..... 105
Şekil 4.40 İkinci el araba veri kümesi standartlaştırılmış etkiler pareto grafiği 106
Şekil 4.41 İkinci el araba veri kümesi ana etkiler grafiği 107
Şekil 4.42 İkinci el araba veri kümesi ikili etkileşim grafiği 108

Şekil 4.43 Üretim veri kümesi standartlaştırılmış etkiler pareto grafiği 110
Şekil 4.44 Üretim veri kümesi ana etkiler grafiği 111
Şekil 4.45 Üretim veri kümesi ikili etkileşim grafiği 111
Şekil 4.46 İkinci el araba veri kümesi standartlaştırılmış etkiler pareto grafiği 113
Şekil 4.47 İkinci el araba veri kümesi ana etkiler grafiği 113
Şekil 4.48 İkinci el araba veri kümesi ikili etkileşim grafiği 114

GİRİŞ

Büyük veri, klasik veri tabanı yönetim araçlarıyla ve veri işleme metotlarıyla işlenemeyecek kadar büyük miktardaki ve karmaşıklıktaki veri topluluklarına verilen addır. Günümüzde veri toplama araçları gittikçe gelişmektedir ve veri toplamak kolaylaşmaktadır. Dijitalleşme, internet, sosyal medya, teknolojik gelişmeler, nesnelerin interneti gibi alanların oluşması ve gelişmesi veri miktarını ve akışını oldukça artırmıştır [1]. Veri miktarının fazla oluşu avantajlı bir durum olsa da aynı zamanda büyük miktardaki veriyle baş edebilmek de gerekmektedir. Klasikleşmiş veri madenciliği yöntemleriyle büyük veri üzerinde çalışılması zor olabilmektedir. Bu durumda, performans ve hız bakımından oldukça geride kalınabilmektedir. Bu yüzden büyük veriler için daha farklı yöntem ve yaklaşımlar gerekmektedir [2].

Günümüzde birçok işletme verilerini düzenli şekilde veri tabanına aktarmaktadır ve bu verileri düzenli olarak saklamaktadır. Bu verilerden yola çıkarak bazı anlamlı ve karar almada yol gösterici bilgi çıkarımları elde edilebilmektedir.

Günümüzde veriler kolaylıkla depolanabilir hale gelse de büyük miktardaki veri ile baş edebilmek için bazı araçlar kullanmak gerekebilmektedir. Bunun için ise genellikle Apache Hadoop, Apache Spark gibi araçlar kullanılmaktadır. Bu programlar sayesinde büyük veriler üzerinde kolay işlemler yapılabilmektedir ve veriler kolayca yönetilebilmektedir. Apache Hadoop daha çok sabit disk (hard disk) üzerinde işlemler yaparken, Apache Spark ise daha çok bellek (ram) üzerinde işlemlerini sürdürür. Bundan dolayı daha hızlıdır ve kullanım olarak daha çok Apache Spark tercih edilmiştir [3, 4].

Tez çalışmasının konusu, makine öğrenme algoritmaları ile büyük veriler üzerinde analizler yapmak amacıyla büyük veriler için halihazırda bulunan makine öğrenme algoritmalarına ek olarak, daha başarılı sonuçlar veren yeni algoritmalar geliştirmektir.

Çalışmada önerilen algoritmalar, büyük veri konusunun sınıflandırma ve fonksiyon tahminleme (regresyon) görevlerini yerine getirmek üzere geliştirilmiştir. Bu kapsamda, Spark hibrit (SH) algoritması ve yapay sinir ağı-karar ağacı (YSA-KA) algoritması isimli iki yeni algoritma önerilmiştir. Önerilen algoritmaların performansları Pokerhand, Susy, Higgs, Hepmass, Wisdm ve Census olmak üzere altı literatür sınıflandırma veri kümesi ve üretim ve ikinci el araba olmak üzere iki gerçek hayat veri kümesi üzerinde analiz edilmiştir. Algoritmaların her bir veri kümesi üzerinde etkin parametre düzeylerini belirlemek amacıyla deney tasarımı gerçekleştirilmiştir. Geliştirilen algoritmalar ile elde edilen sonuçlar, Spark'ta bulunan lojistik regresyon, karar ağacı, rastgele orman, doğrusal regresyon, izotonik regresyon, destek vektör makinesi, yapay sinir ağları algoritmaları ve geçmiş literatür çalışmalarında önerilen algoritmalar ile karşılaştırılmıştır. Karşılaştırmalarda elde edilen sonuçların istatistiksel olarak anlamlı olup olmadığını ortaya koymak için istatistiksel analizler gerçekleştirilmiştir.

Tez çalışmasının giriş bölümünde teze bir giriş yapılmış ve tezin konusu, kapsamı, amacı ve tezde kullanılan yöntemden genel olarak bahsedilmiştir.

Birinci bölümde büyük veri ile alakalı literatür taramasına yer verilmiştir. Literatür genel olarak sınıflandırma, kümeleme, birliktelik analizi, gerçek zamanlı veri madenciliği, metin madenciliği gibi alt alanlara ayrılmıştır.

İkinci bölümde veri madenciliği, büyük veri konularından ve büyük veri sistemlerinden bahsedilmiştir. Apache Spark ve Spark makine öğrenme kütüphanesi hakkında detaylı bilgiler verilmiştir.

Üçüncü bölümde, geliştirilen Spark Hibrit (SH) ve Yapay Sinir Ağı – Karar Ağacı (YSA-KA) algoritmaları hakkında bilgi verilmiştir. Tez çalışmasında önerilen bu iki yeni algoritma, Spark'ta bulunan algoritmaların hibritleştirilerek geliştirilmiş halleridir.

Dördüncü bölümde deneysel çalışmalar sunulmuştur. Öncelikle tez kapsamında kullanılan literatür ve gerçek hayat verileri hakkında bilgiler verilmiş; performans değerlendirmede kullanılan etkinlik ölçütünden bahsedilmiştir. Sonrasında geliştirilen algoritmaların etkin parametre düzeylerini belirlemek için gerçekleştirilen deney tasarımlarına yer verilmiştir. Önerilen algoritmalar ile elde edilen sonuçlar,

karşılaştırmalar ve istatistiksel analizlere de yine bu bölümde değinilmiştir. Böylece algoritmaların başarıları değerlendirilmiştir.

Beşinci bölümde sonuçlar yorumlanarak, gelecek çalışmalar için önerilerde bulunulmuştur.

1. BÖLÜM

LİTERATÜR ÇALIŞMASI

Büyük veri ile alakalı literatürde çok sayıda çalışma mevcuttur. Bu çalışmalar sınıflandırma, kümeleme, birliktelik analizi, zaman serileri, metin madenciliği ve diğer çalışmalar olarak kategorilere ayrılabilir.

1.1. Sınıflandırma Literatürü

Sınıflandırma, eldeki bilgilerle veriyi sınıflara ayırmaya yarayan bir yöntemdir. Sınıflar önceden bellidir ve eldeki veriler ne kadar fazla oranda doğru sınıflara ayrılırsa sınıflandırma algoritma o kadar başarılıdır. Büyük veride sınıflandırma alanında yapılan çalışmalar aşağıdaki gibi özetlenebilir:

Sleeman ve Krawczyk yaptıkları çalışmada [5] sınıflandırma alanında dengeli olmayan veri kümeleri için Spark üzerinde çalışmışlardır. Dengeli olmayan veri kümelerinin eğitimindeki en büyük problem, sınıflardan biri veri kümesinde yüksek bir şekilde baskın olduğu için modelin sürekli bu baskın sınıfa yönelmesidir. Bu probleme çözüm üretmek için Sleeman ve Krawczyk geliştirilen algoritmaları Spark üzerinde çalışır hale getirmişlerdir.

Ahmed vd. [6] küçük bir diyabet verisi üzerinde diyabet tahmini için Apache Spark makine öğrenme algoritmalarını kullanılmışlar ve karşılaştırmışlardır. Ele aldıkları algoritmalardan lojistik regresyon algoritması; karar ağacı, rastgele orman, destek vektör makinesi ve Naive Bayes algoritmalarından daha iyi sonuç vermiştir.

Maillo vd. [7] daha önceden üzerinde çalıştıkları bulanık k-en yakın komşu algoritmasını daha da geliştirmişlerdir ve daha fazla veri üzerinde test etmişlerdir. Hem hız bakımından hem de doğruluk bakımından algoritmayı daha etkili bir hale getirmişlerdir.

Lenz vd. yaptıkları çalışmada [8] sıralanmış ve ağırlıklandırılmış bir şekilde bulanık bir en yakın komşu algoritması geliştirmişlerdir. Sınıflara bulanık alt ve üst sınırlara göre ait olmayan örnekleri sınıflandıran bir algoritmadır. Bu algoritma sayesinde herhangi bir sınıfa atanamayan veriler sınıflara atanarak doğruluk oranı artırılmıştır. Kullandıkları algoritma k en yakın komşu algoritmasıdır.

Oo ve Thein'in yaptığı çalışmada [9] Spark yardımıyla büyük veriler için rastgele orman algoritmasını kullanmışlardır. Bu algoritmayı nitelik indirgeme ile birlikte en iyi parametre değerleriyle kullanmışlardır. Algoritmayı beş büyük veri kümesi üzerinde test etmişlerdir ve Spark'ta bulunan rastgele orman algoritması ile kıyaslamışlardır. Süre olarak daha kısa sürede sonuca ulaşmışlardır.

Za’in vd. yaptıkları çalışmada [10] büyük ölçekli akan veri için ölçeklenebilir bir ağ tabanlı bulanık bir sistem geliştirmişlerdir. Bu sistem ile veriyi dağıtık şekilde analiz etmişlerdir. Geliştirdikleri sistem bir yapay sinir ağı yöntemidir. Çok sınıflı sınıflandırma türünde Spark tabanlı algoritmalarla karşılaştırmışlardır ve daha hızlı sonuçlar almışlardır.

Weinberg ve Last yaptıkları çalışmada [11] farklı ağaç tabanlı algoritma modellerini kullanarak bir algoritma geliştirmişlerdir. Rastgele orman algoritmasındaki yaklaşıma benzer olarak farklı ağaç yapılarını ve algoritmalarını da parça parça çalıştırmışlardır. En son bunlar arasında en çok seçilen sınıfta karar kılmışlardır. Çalışmaları rastgele orman algoritmasının gelişmiş bir versiyonu olarak düşünülebilir.

Wang vd. [12] sınıflandırma problemi için kısa zamanda yüksek doğruluk değeri veren bir algoritma geliştirmişlerdir. Geliştirdikleri algoritma karar ağacı temelli bir algoritmadır. Bu yeni geliştirdikleri algoritmayı Susy, Higgs gibi büyük verilerin yanında 20'den fazla küçük veriyle de test etmişlerdir. Paralel işlem sistemi geliştirmedikleri için algoritmaları normalde hızlı olsa da paralel sistemler kadar hızlı olamamıştır.

Donkal ve Verma yaptıkları çalışmada [13] sunuculara yapılan saldırıların tutulduğu bir veri kümesi üzerinde sınıflandırma çalışması yapmışlardır. Genetik algoritma yardımıyla nitelik seçimi yapmışlardır. Yaptıkları nitelik seçimlerini Spark'taki algoritmalar ile test etmişlerdir ve doğruluk oranını artırmışlardır.

Dahiya ve Srivastava yaptıkları çalışmada [14] network saldırılarını Spark kullanarak analiz etmişlerdir. Dokuz farklı tipteki saldırı türü ve saldırı olmayan normal bağlantı türünü bulmaya çalışmışlardır. Öncelikle nitelik seçimi yaptıktan sonra sınıflandırma algoritmaları ile bağlantıları sınıflandırmışlardır. Çalışmalarında rastgele orman algoritması en iyi sonucu vermiştir.

Maillo vd. yaptıkları çalışmada [15] bulanık K-en yakın komşu algoritması kullanmışlardır. Kullandıkları algoritmadaki mantık yeni gelen her örnek için eğitim kümesindeki sınıflara yakınlıkları ölçmektir. Veri miktarı arttıkça bu algoritmanın çalışması zorlaşmaktadır. Bu yüzden Apache Spark yardımıyla dağıtık şekilde çalıştırmışlardır. Bu sayede daha hızlı sonuçlar elde etmişlerdir.

İnoue vd. yaptıkları çalışmada [16] derin ve tekrarlayan yapay sinir ağı geliştirmiş ve mobil insan aktiviteleri üzerinde kullanmışlardır. Cep telefonu yardımıyla ölçülen veriler üzerinden aktivite tahmini yapmışlardır.

İgnatov yaptığı çalışmada [17] evrişimsel yapay sinir ağı kullanarak insan hareketlerini tahmin etmeye yarayan bir sistem geliştirmiştir. Veriyi bir zaman serisi gibi dizayn edip kullanmıştır. Verileri 1 saniye aralıklarla kullanmıştır. Ayrıca nitelik seçimi ile de testler yapmıştır.

Chakrabarthy ve Biswas yaptıkları çalışmada [18] finans verisi için istatistiksel bir yaklaşım geliştirmişlerdir. Bu istatistiksel analizler sonucunda önemli olan niteliklere karar vermişlerdir. Önemli olan nitelikler üzerinden karar ağacını ilerletmişlerdir. Birden fazla karar ağacı kullanmışlardır ve rastgele orman yapısı oluşturmuşlardır.

Maillo vd. yaptıkları çalışmada [19] büyük veriler için bulanık bir k en yakın komşu algoritması geliştirmişlerdir. Bu algoritmayı kümeleme problemleri yerine sınıflandırma problemleri için kullanmışlardır. Test kümesindeki örnekleri, eğitim kümesindeki örnekler üzerinden hangisine en yakın çıkıyorsa o sınıfa atamışlardır. Bu işlemi yapmak

için çok büyük bir hesaplamaya ve zamana ihtiyaç vardır. Bundan dolayı bu algoritmalarını Spark yardımıyla dağıtık bir biçimde çalıştırmışlardır ve süre olarak klasik yöntemlerden çok daha hızlı sonuç almışlardır. Çalışmalarında kullandıkları veri kümeleri Pokerhand, Susy ve Higgs veri kümeleridir.

Weinberg ve Last yaptıkları çalışmada [20] büyük veriyi paralel işlemek için parçalara ayırmışlardır. Ayrılan her bir parça karar ağacıyla eğitilmiştir. Bunun sonucunda elde edilen karar ağaç modelleri birbirleriyle kıyaslanmıştır ve diğer modellere en çok benzeyen model seçilmiştir. Bu seçilen model üzerinden çalışılmıştır ve ortalama sonuçlar elde edilmiştir, çok iyi doğruluk değerleri elde edilememiştir. Fakat hız olarak ve bilgisayar hafıza kullanımı olarak daha verimli sonuçlar elde edilmiştir.

Maillo vd. [21] veri madenciliğinde k en yakın komşu algoritmasının etkili bir algoritma olduğundan bahsetmişlerdir. Bu algoritmanın büyük veri üzerinde klasik şekilde uygulanamayacağını belirtmişlerdir. Buna sebep olarak süre ve hafıza limitleri gösterilmiştir. Bu limiti aşmak için algoritma Apache Spark ve Apache Hadoop üzerinde çalıştırılmıştır. Literatürde bulunan dört veri kümesi üzerinde farklı parametrelerle algoritma test edilmiştir. Apache Spark, Apache Hadoop'a göre 10 kata yakın daha az sürede sonuç vermiştir.

Garcia-Gil vd. tarafından yapılan çalışmada [22] Apache Spark ve Apache Flink ölçeklenebilir büyük veriler üzerinden kıyaslanmıştır. Bu kıyaslama için destek vektör makinesi (SVM) ve doğrusal regresyon algoritmaları kullanılmıştır. Buna ek olarak nitelik seçimi algoritmaları da test edilmiştir. Apache Spark hız olarak oldukça öne çıkmıştır. Fakat Apache Flink'in makine öğrenme kütüphanesi çok az farkla önde çıkmıştır. Bunun sebebi olarak araştırmacılar veri dönüşümünü ifade etmişlerdir.

Bechini vd. [23] CAR (Classification Association Rules) adında bir algoritma üzerinde çalışmışlardır. Bu algoritmayı, büyük veri üzerinde çalışması için uygun hale getirilmiş dağıtılmış FP-Growth algoritması olarak tanımlamışlardır. Bu algoritma yedi adet bilinen literatür veri kümesiyle test edilmiştir. Bu çalışmada Spark'ın yanı sıra Mahout da kullanılmıştır. Aralarında anlamlı bir farklılık bulunmamıştır.

Alsheikh vd. [24] mobil cihazların ve nesnelerin interneti teknolojisinin yaygınlaşmasıyla mobil alanında çok büyük veriler elde edildiğinden bahsetmişlerdir. Veriler sensör

yardımıyla toplanan ve kişinin o an ne yaptığını tahmin eden verilerdir. Toplanan büyük miktardaki mobil verilerden anlamlı bilgiler ve gizli şablonlar keşfedilmiştir. Bunlar derin öğrenme metotlarının Apache Spark üzerinde çalıştırılmasıyla elde edilmiştir. Her bir parça bir derin öğrenme modeli oluşturmuştur ve oluşturulan bu modellerin ortalaması alınmıştır. Apache Spark sayesinde bu işlem yaklaşık dörtte biri kadar kısa sürede gerçekleştirilmiştir.

Ronao ve Cho yaptıkları çalışmada [25] cep telefonları sensörleri sayesinde kaydedilen, insanların hareketlerini tahmin etmeye çalışmışlardır. Çalışmalarında derin öğrenme yapay sinir ağı kullanmışlardır. İlk katmanda evrişimsel yapay sinir ağı kullanmışlardır, devamında ileri beslemeli tam bağlantılı yapay sinir ağı kullanmışlardır. Daha sonra aynı veri üzerinde yaptıkları başka bir çalışmada [26] ise hiyerarşik sürekli gizli Markov modellerini kullanmışlardır.

Silva vd. yaptıkları çalışmada [27] markov geometrik difüzyon nitelik seçimi yöntemini daha da geliştirmişlerdir ve bunu büyük veriler için uygun hale getirmişlerdir. Bu sayede az etkili veya etkili olmayan nitelikleri büyük veriler için daha kolay bir şekilde saptamışlardır. Çalışmalarında kullandıkları algoritma k en yakın komşu algoritmasıdır.

Fish vd. yaptıkları çalışmada [28] finansal veriler için destek vektör makine ve lojistik regresyon yöntemleriyle sınıflandırma çalışması yapmışlardır. Veriler içindeki dengesiz durumları düzeltmek için ise veriler içinde ağırlıklarla da çalışmışlardır.

Triguero vd. [29] büyük veri çağında, büyük ölçekli veri kümelerini analiz etmenin ve bilgiyi ayıklamanın çok ilginç ve zor bir görev olduğunu belirtmişlerdir. Bu veri kümelerinde standart veri madenciliği araçlarının uygulanması zordur. Dolayısıyla, bulut platformlarının muazzam depolama ve işleme kapasitesini benimseyen, yeni bir ölçeklenebilir madencilik yöntemi sınıfı gerekmektedir. En yakın komşu sınıflandırmasında prototip azaltma teknikleri için yeni bir dağıtılmış bölme yöntemi önerilmektedir. Bu yöntemler orijinal eğitim verilerini daha az sayıda örnek olarak temsil etmeyi amaçlamaktadır. Ana amaçları, sınıflandırma sürecini hızlandırmak ve en yakın komşu kuralı için saklama gereksinimlerini ve duyarlılığı azaltmaktır. Bununla birlikte, standart prototip azaltma yöntemleri çok büyük veri kümeleri ile baş edemez. Bu kısıtlamanın üstesinden gelmek için, bu algoritmaların işleyişini, çok parçalı çözümleri

tek bir algoritmaya entegre etmek için çeşitli algoritmik stratejiler önermek üzere bir bilgisayar öğesi kümesi aracılığıyla dağıtmak için MapReduce tabanlı bir çerçeve geliştirilmiştir. Önerilen model prototip azaltma algoritmalarının büyük veri sınıflandırma problemlerine önemli bir doğruluk kaybı olmaksızın uygulanmasını sağlamıştır.

Shashidhara vd. yaptıkları çalışmada [30] banka ve higgs veri kümeleri üzerinde lojistik regresyon ve doğrusal destek vektör makine algoritmalarını Weka, Scikit Learn ve Apache Spark üzerinde test etmişlerdir. Çıkan sonuçlara göre platformları karşılaştırmışlardır. Küçük boyutta veriler için Scikit Learn, büyük boyutta veriler için Spark'ı önermişlerdir. CPU kullanımını en iyi Spark başarmıştır.

Oza yaptığı çalışmada [31] Naive Bayes algoritması ve yapay sinir ağları ile sınıflandırma üzerinde çalışmıştır. Veri üzerinde ağırlıklandırma yöntemini de kullanmıştır. Böylece hatalı sınıflandırılan örneklerin ağırlıklarını artırarak daha doğru bir sonuç almıştır.

Rio vd. tarafından yapılan çalışmada [32] dengesiz büyük miktardaki veri kümeleri için rastgele orman algoritması üzerinde çalışılmıştır. Rastgele orman algoritması bir sınıflandırma algoritmasıdır. Veri büyüklüğüyle baş edebilmek için MapReduce tekniği kullanılmıştır. Farklı parametreler ile literatür verileri üzerinde analizler yapılmıştır. Kısa süreler içerisinde algoritmalar sonuçlar verebilmiştir.

Anguita vd. yaptıkları çalışmada [33] insan hareketlerini tahmin etmek için destek vektör makinesi yöntemini kullanmışlardır. Ayrıca nitelik seçimi yapıp önemsiz nitelikleri çıkartmışlardır.

Kamiran vd. yaptıkları çalışmada [34] karar ağacı ile sınıflandırma yapmışlardır. Veri olarak ABD ve Hollanda finans verisini kullanmışlardır. Karar ağacında önemsiz gözüken niteliklerin belirlenerek veri kümesinden kaldırılması önerilmiştir.

1.2. Kümeleme Literatürü

Kümeleme, sınıflandırmaya benzer bir yapıdadır, fakat sınıflar önceden belli değildir. Kümelemede amaç, verileri birbirlerine benzerlik oranlarına göre ayıklamaktır. Büyük veride kümeleme alanında yapılan çalışmalar şöyle özetlenebilir:

Himabindu vd. [35] kümeleme türündeki problemler için ateşböceği (Firefly) algoritması tabanlı bir k-ortalamalar algoritması geliştirmişlerdir. Ateşböceği algoritmasını kümelerin merkez noktasını bulmak için kullanmışlardır. İterasyonlar ilerledikçe merkez noktalar değişmekte ve kümeleme daha iyi şekilde yapılmaktadır. Küçük veriler için algoritmayı test etmişlerdir ve süre olarak çok iyi olmasa da doğruluk olarak daha iyi sonuç elde etmişlerdir.

Tao vd. [36] yüksek nitelik sayısı olan veriler için akıllı bir kümeleme algoritması geliştirmişlerdir. Normal kümeleme algoritmaları, yüksek nitelik sayısına sahip olan verilerde iyi sonuç vermemektedirler, bunun sebebi bütün niteliklerin aynı ağırlıkta değerlendirilmesidir. Sürü algoritması ile k-ortalamalar algoritmasını hibritleyerek bu problemin üstesinden gelmişlerdir. Sürü algoritması ile kümelerin merkezleri ve verideki niteliklerin ağırlıkları optimize edilmiştir. Çok sayıda nitelik değerine sahip veri kümeleri üzerinde hem tek işlemci ile hem de Spark kullanarak paralel bir şekilde algoritmayı çalıştırmışlardır. Etkili sonuçlar elde etmişlerdir.

Lu yaptığı çalışmada [37] kümeleme performansını ve hızını artırmak için k-ortalamalar algoritması üzerinde geliştirme yapmıştır. Bu yeni algoritmayı Hadoop üzerinden hız performansını artırmak için test etmiştir. İlk olarak noktalar ile küme merkezleri arasında bir uzaklık aralığı belirlemiştir ve bu aralığı aşmayan noktaları o kümeye atamıştır. Dışarıda kalan noktaları ise en yakın kümeye atamıştır. Bu algoritmayı Hadoop ve Apache Mahout üzerinde test etmiş ve iyi sonuçlar elde etmiştir.

Wu vd. yaptıkları çalışmada [38] yeni bir kümeleme algoritması geliştirmişlerdir. Veri arttıkça doğrusal bir biçimde işlem yükünün arttığını ve bundan dolayı büyük veriler için iyi bir yöntem olduğunu söylemişlerdir. Yöntemi k-ortalamalar yöntemiyle kıyaslamışlar ve daha iyi sonuçlar elde etmişlerdir.

Hai vd. yaptıkları bir çalışmada [39] büyük veri platformlarından Spark, Hadoop ve DataMPI'i kümeleme algoritmaları üzerinden kıyaslamışlardır. Paralel şekilde çalışan k-ortalamalar, bulanık k-ortalamalar ve Canopy algoritmalarını kullanmışlardır. Sırasıyla DataMPI, Spark ve Hadoop en etkili sistemler olarak ortaya çıkmıştır. İşlemci sayısı ve hafıza arttıkça hız da artmıştır.

Cui vd. [40] kümeleme analizi üzerinde çalışmışlardır. Kümeleme, en çok kullanılan veri işleme algoritmalarından biridir. K-ortalamalar algoritması basitliği nedeniyle en çok kullanılan kümeleme algoritmasıdır. Son zamanlarda veri miktarı arttığı için araştırmacılar MapReduce'a yönelmeye başlamışlardır. MapReduce tekrarlanan iş, büyük veri okuma ve karıştırma süreleri nedeniyle yinelenen algoritmalar için uygun değildir. Büyük ölçekli verilerin k-ortalamalar kümeleme algoritması kullanılarak işlenmesi ve yinelemeye bağımlılığın ortadan kaldırılması ve yüksek performans elde etmek için MapReduce'da yeni bir işleme modeli önerilmiştir.

Zheng vd. [41] kümeleme tekniği yardımıyla kişisel haber öneri sistemi kurmuşlardır. Öncelikle haber sitesindeki okuyucuları kümelere ayırmışlardır. Sonra haberleri hiyerarşik bir yapıda kategoriye ayırmışlardır. Kümelerdeki okuyucular bu hiyerarşik yapıdan kendilerine uygun düşen haberleri öneri olarak görmüşlerdir.

1.3. Birliktelik Analizi Literatürü

Birliktelik analizi eldeki verilerde birlikte geçen durumları saptamaya çalışır. Market, sepet analizi gibi adlandırmalara da sahiptir. Markette birlikte alınan ürünler gibi durumlarda kullanılır. Büyük veride birliktelik analizi alanında yapılan çalışmalar aşağıdaki gibi özetlenebilir:

Shang vd. [42] bulanık bir birliktelik analizi yöntemiyle şirketlerin finansal risklerini önceden saptamayı amaçlamışlardır. Bunun için ilk olarak korelasyon yardımıyla kullanılacak finansal gösterge sayısını 32'den 12'ye düşürüşümlerdir. Şirketlerin bu 12 göstergesini uzun dönemde inceleyip 4 farklı kümeye bölmüşlerdir ve bu kümeler üzerinden birliktelik analizi uygulamışlardır.

Chan vd. [43] grip aşılarıyla alakalı Twitter üzerinden topladıkları verilerle çalışma yapmışlardır. Tweetlerde geçen kelimeler üzerinde birliktelik analizi uygulamışlardır. Bu işlemleri MapReduce ile yaptıkları için de süreci hızlandırmışlardır. Birliktelik analizindeki kelimeleri ayrıca yıl içerisindeki aylara göre de analiz etmişlerdir.

Chen vd. [44] NGEP algoritması ile birliktelik analizi üzerinde çalışmıştır. NGEP algoritması ile FP-Growth ve en sık kullanılan Apriori algoritmaları karşılaştırılmıştır.

Daha iyi sonuçlar elde edilmiştir. Ayrıca süre olarak da değerlendirilmiştir. Süre olarak da önerdikleri algoritma daha başarılı sonuçlar vermiştir.

Zhang vd. [45] tarafından DFIMA (Distributed frequent itemset mining algorithm) adında birliktelik analizi üzerine bir algoritma önermişlerdir. Bu algoritma Apache Spark üzerinde çalıştırılmıştır. DFIMA ve FP-Growth algoritmaları 4 MB, 15 MB ve 2 GB boyutlarındaki veri kümeleriyle test edilmiştir. Çalışılan algoritma FP-Growth algoritmasını geçmiştir.

Moens vd. yaptıkları çalışmada [46] birliktelik analizi veya market sepet analizi için kendi geliştirdikleri iki algoritmayı test etmişlerdir. MapReduce tekniği sayesinde bu iki algoritma büyük veri üzerinde test edilmiştir. Algoritmalar başarı ve süre bakımından karşılaştırılmıştır.

1.4. Zaman Serileri Literatürü

Zaman serileri belirli periyotlarla gelişen olayların geçmişten elde edilen bilgilerle, gelecek tahminini yapmaya çalışan problem türüdür. Borsada, hava tahmininde ve buna benzer veri türlerinde kullanılır. Büyük veride zaman serileri konusunda yapılan çalışmalar şu şekilde özetlenebilir:

Syed vd. [47] Londra'nın enerji harcama hesaplaması tahmini için bir çalışma yapmışlardır. Kısa, orta ve uzun dönemler için bu tahminleri yapmışlardır. Apache Spark ve Hadoop kullanmışlardır. Spark'ın makine öğrenme kütüphanesi olan Mllib'ten de yararlanmışlardır. Bu makine öğrenme kütüphaneleri içerisindeki regresyon, rastgele orman, karar ağacı gibi algoritmaları denemişlerdir. Çıkan sonuçlarda Spark sayesinde oldukça hızlı ve başarısı yüksek değerler elde etmişlerdir.

Xu vd. yaptıkları çalışmada [48] rüzgar hızını tahmin etmeye çalışmışlardır. Bunun için Çin'de her 3 dakikada bir rüzgâr hızı kaydeden bir veri kümesi kullanmışlardır ve bu veriler üzerinden zaman serisi çalışması yapmışlardır. Yapay sinir ağlarından yararlandıkları bu çalışmalarını Spark üzerinden paralel ve dağıtık bir biçimde yaptıkları için tek bir bilgisayarda çalışmaya göre hızlı sonuçlar almışlardır.

Llames vd. [49] büyük veriler için zaman serileri tahmininde çalışmışlardır. K ağırlıklı en yakın komşu algoritmasını Spark kullanarak İspanya enerji tüketim verileri üzerinde test etmişlerdir. Diğer algoritmalara göre daha iyi sonuç elde etmişlerdir.

Galicia vd. yaptıkları çalışmada [50] yüksek frekanslı ölçümlerle oluşturulan zaman serileri üzerinde Spark ile bir metot geliştirmişlerdir. Spark makine öğrenme kütüphanesi çok değişkenli regresyonu desteklemediği için problemi parçalara bölerek alt problemler oluşturmuşlardır. Alt problemleri Spark'ın kendi içerisinde olan makine öğrenme algoritmalarıyla eğitmişlerdir. Veri olarak ise İspanya'daki elektrik taleplerinin 10 dakikalık arayla tutulan zaman serisi verisini kullanmışlardır. Tahmin hatalarını %2'ye kadar düşürmüşlerdir.

1.5. Metin Madenciliği Literatürü

Metin madenciliğinde amaç metinlerden anlamlı bilgiler elde etmektir. Metinler çeşitli aşamalardan geçirilerek sayısallaştırılır ve bilgisayar tarafından işlenebilir ve sonuç verebilir bir hale dönüştürülür. Genellikle Twitter gibi sitelerden toplanan verilerle işlemler yapılır. Büyük veride metin madenciliği alanında yapılan çalışmalar aşağıdaki gibi özetlenebilir:

Sohangir vd. [51] finansal bir sosyal ağ olan StockTwits'ten veriler almışlardır. Bu veriler finans ve borsa ile alakalı fiyat bilgileri, alım satım tavsiyeleri, hareketlenmeler, trendler gibi birçok metinden oluşmaktadır. Gelecekteki fiyat tahminleri ile bu kullanıcılardan toplanan veriler arasında ilişki bulmaya çalışmışlardır. Bunu duygu analizi (sentiment analysis) ile yapmışlardır. Bazı başarılı tahminlerde bulunmuşlardır.

Alaoui vd. [52] sosyal medya verilerini analiz etmişlerdir. 2016 yılındaki ABD başkanlık seçimleri ile alakalı yazılar üzerinden metin madenciliği yapmışlardır. Bunun için Kafka, Apache Spark ve Apache Hadoop kullanmışlardır. Twitter verileri üzerinden analiz yapmışlardır. Analizlerinde Donald Trump ve Hillary Clinton'u karşılaştırmışlardır. Trump, Clinton'dan daha iyi bir skor elde etmiştir.

Nodarakis vd. [53] Twitter üzerinde büyük ölçekli duygu analizi (sentiment analysis) çalışmışlardır. Klasik yöntemlerle Twitter paylaşımlarının çok az bir kısmı analiz edilebilmektedir. Bu durumu geliştirmek ve daha yüksek sayıda analiz yapabilmek için

Apache Spark üzerinde çalışan bir metot geliştirmişlerdir. kNN (k en yakın komşu) algoritması üzerinden gitmişlerdir. İyi performans gösteren bir çalışma yapmışlardır.

Shoro ve Soomro tarafından yapılan çalışmada [54] Twitter Stream API ve Apache Spark kullanılarak Twitter paylaşımları ile alakalı analizler yapılmıştır. En çok kullanılan kelimeler, en çok kullanılan diller gibi bilgilere ulaşılmıştır ve bu bilgiler grafikler aracılığıyla görselleştirilmiştir.

Aggarwal ve Wang'a [55] göre sosyal ağlar metin ve multimedya bakımından oldukça zengindir. Metinler ilgili algoritmaların kullanımıyla değerli bir sonuca dönüşebilir. Bu metinleri işlemek için çeşitli algoritma kullanım tavsiyeleri sunmaktadırlar.

Ritter vd. [56] başarılı bir şekilde tweet analizleri yapmışlardır. İnsanlar her gün 100 milyondan fazla sayıda tweet atmaktadırlar. Bu tweetler 140 karakterle sınırlıdır ve yazım yanlışlarıyla dolu, gayri resmi, düzensiz şekilde yazılmaktadır. Her insan farklı şekilde yazmaktadır. Standart dil işleme işlemleri tweetler için pek başarılı değildir. Özellikle bu problem üzerinde durmuşlardır.

1.6. Diğer Literatür

Yukarıda sınıflandırılarak verilen büyük veri çalışmalarının yanı sıra bu sınıflara girmeyen aşağıda açıklanan çalışmalar da büyük veri konusunda gerçekleştirilmiştir.

Ruihong ve Zhihua yaptıkları çalışmada [57] Apache Hadoop ile Apache Spark'ı kıyaslamışlardır. Hafıza kullanımı olarak Spark yaklaşık olarak %5 daha fazla hafıza kullanmıştır ve %5 daha hızlı çıkmıştır. Güvenlik olarak bakıldığında ise Hadoop %5 daha güvenli çıkmıştır. Genel olarak değerlendirdiklerinde Spark'ı daha iyi bulmuşlardır.

Yin vd. tarafından yapılan çalışmada [58] Spark kullanarak nitelik azaltma algoritması geliştirmişlerdir. Önceden Hadoop MapReduce kullanılarak yapılan algoritmalara göre Spark daha hızlı olduğu için hız anlamında gelişmiş bir algoritmadır. Yapılan iterasyon sayısını da bir formüle bağlamış ve azaltmışlardır. Bu sayede buradan da hız bakımından bir üstünlük sağlamışlardır.

Exposito vd. tarafından yapılan çalışmada [59] SMusket adında Spark temelli bir DNA hata düzeltici yapmışlardır. MapReduce mantığını kullanarak yaptıkları paralel işlemlerle çok büyük bir hız avantajı yakalamışlardır.

Karim vd. yaptıkları çalışmada [60] trafikte gerçekleşen araç ve yaya kazalarının bulanık mantık kullanarak risk analizini gerçekleştirmişlerdir. Simülasyon üzerinden veri üretmişlerdir ve gerçek zamanlı olarak Spark ve Hadoop üzerinde çalıştırmışlardır. Simülasyon dışında gerçek hayatta nesnelerin interneti kullanılarak kullanılabileceğinden bahsetmişlerdir.

Islam vd. yaptıkları çalışmada [61] bir bilgisayarda veya sunucuda çalışan çok sayıdaki büyük veri uygulamalarının etkili bir biçimde çalışması üzerinde durmuşlardır. Amaçları dinamik bir biçimde kaynak dağıtımlarını düzenleyerek elektrik ve sunucu masraflarını en aza indirgemektir. Çalışmalarını Apache Mesos üzerinde yapmışlardır ve Spark'a göre daha az kaynak kullanmışlardır.

Manogaran ve Lopez [62] yaptıkları çalışmada büyük veri analizini iklim değişikliğini gözlemlemek için kullanmışlardır. Çok büyük miktardaki iklim verileri eski yöntemlerle tutulamamaktadır. Yaptıkları çalışmada Apache Hadoop altyapısıyla MapReduce algoritmasını kullanmışlardır. İklimsel parametrelerin mevsimsel olarak ölçümleri yapılmış ve ortalamalar hesaplanmıştır. Kümülatif ortalamalar üzerinden grafikler takip edilmiştir. Herhangi bir tahmin yapılmamış, sadece veriler üzerinde hesaplamalar yapılmıştır.

Garcia-Gil vd. tarafından yapılan çalışmada [63] özel bir nitelik seçimi metodu olan temel bileşenler analizini Spark üzerinde uygulamışlardır. Higgs, Susy gibi büyük veriler üzerinde yöntemlerini uygulamışlardır ve iyi sonuçlar almışlardır.

Nair vd. yaptıkları çalışmada [64] Twitter'dan anlık veri toplayarak gerçek zamanlı sağlık durumu tahmini sistemi geliştirmişlerdir. Önceden belirlenmiş olan bir formatta kişinin vücut ölçümleri ve fiziksel durumu tweet olarak gönderilir. Bu atılan tweetler Spark streaming ile çekilir ve kişinin sağlık durumu tahmin edilir. Kişinin sağlık durumu mesaj olarak kullanıcıya gönderilir. Çalışmalarında örnek olarak literatürdeki kalp hastalıkları veri kümesini kullanmışlardır. Çalışmaları büyük miktardaki veriden daha çok gerçek zamanlı veri madenciliğine girmektedir.

Gallego vd. tarafından yapılan çalışmada [65] büyük nitelikli büyük veriler üzerinde nitelik seçimi üzerinde durulmuştur. Nitelik seçimi işlemleri büyük veriler üzerinde paralelleştirme sayesinde hızlandırılmıştır. Verilerden sonuç çıkarma işlemleri yerine sadece nitelik seçim kısmıyla ilgilenmişlerdir ve normalde çok uzun süren işlemleri kısaltmışlardır.

Assefi vd. [66] yapay zekâ ve makine öğrenme tekniklerinin verilerden bilgi çıkarmak için önemli olduğunu söylemişlerdir. Fakat büyük miktardaki veriler için bu yöntemleri uygulamanın pahalı ve zor olacağını da eklemişlerdir. Çok fazla işlemci, hafıza kullanacaklarından bahsetmişlerdir. Makine öğrenme algoritmalarını hem Apache Spark üzerinde hem de bir veri madenciliği programı olan Weka üzerinde denemişlerdir. Apache Spark, Weka'ya göre oldukça hızlı sonuç vermiştir.

Lv vd. [67] tarafından yapılan çalışmada büyük veri sayesinde trafik yoğunluk ve akışı tahmini yapılmıştır. Otomatik kodlayıcı (Autoencoder) kullanılarak derin sinir ağları oluşturulmuştur. Veri olarak California şehrindeki trafik akışı kullanılmıştır.

Walker [68] büyük verinin önemiyle birlikte, Google'nin bir günde 24 petabayt boyutundaki veriyi işlediğinden ve bir günde Twitter'da 400 milyona yakın paylaşım yapıldığından bahsetmiştir. Bu kadar verinin bir insan tarafından kontrol edilemeyeceğini ve insan sınırını aştığını söylemiştir. Visa'nın normalde 1 ayda işlediği bir veri kümesini Apache Hadoop kullanarak 13 dakikada işlediğinden bahsetmiştir.

Kitchin çalışmasında [69] büyük veri sayesinde akıllı şehirler oluşturulabileceğini söylemiştir. Gerçek zamanlı olarak şehirdeki hava durumu, rüzgâr şiddeti, trafik durumu, bisiklet kullanımı, tünel ve metro durumu gibi bilgilerin takip edileceğinden bahsetmiştir. Gerçek zamanlı akıllı şehirlerin yaygınlaşacağı görüşünü paylaşmıştır.

Kasun vd. [70] oto kodlayıcı ile kurdukları derin sinir ağları ile görüntü işleme üzerinde çalışmışlardır. El yazılarından oluşan büyük miktardaki veri kümesi üzerinde çalışmışlardır. Başarılı sonuçlar elde etmişlerdir.

Brin ve Page [71] büyük kapsamlı bir internet arama motoru olan Google'ı oluşturmuşlardır. İnterneti etkili bir biçimde indekslemeyi ve aranan sonuçlara kolayca ulaşmayı amaç edinmişlerdir. Geleneksel arama tekniklerini kullandıklarında iyi sonuç

elde edilememektedir. Veri miktarının büyük olduğu düşünüldüğünde geleneksel arama yöntemlerine farklılıklar katarak ve daha etkili arama algoritmaları kullanarak daha iyi sonuçlar bulan bir arama motoru inşa etmişlerdir.

Bu tez çalışmasında büyük veride sınıflandırma ve fonksiyon tahminleme görevlerini yerine getirebilmek ve kısa sürede etkin sonuçlar elde edebilmek amacıyla iki yeni algoritma önerilmiştir. Algoritmaların gerek literatür, gerekse de gerçek hayat büyük verileri üzerindeki performansları çeşitli geleneksel algoritmalar ve literatürde önerilen algoritmalarla karşılaştırılmıştır. Yapılan deneysel çalışmalar, önerilen algoritmaların sınıflandırma ve fonksiyon tahminleme alanlarında oldukça başarılı olduğunu göstermiştir.

2. BÖLÜM

BÜYÜK VERİ

Günümüzde veri toplama araçları gittikçe gelişmektedir ve veri toplamak kolaylaşmaktadır. Dijitalleşme, internet, sosyal medya, teknolojik gelişmeler, nesnelerin interneti gibi alanların oluşması ve gelişmesi veri miktarını ve akışını oldukça artırmıştır. Veri miktarının fazla oluşu avantajlı bir durum olsa da aynı zamanda büyük miktardaki veriyle baş edebilmek de gerekmektedir. Klasikleşmiş veri madenciliği yöntemleriyle büyük veri üzerinde çalışılması zor olabilmektedir. Performans ve hız bakımından oldukça geride kalınabilmektedir. Bu yüzden büyük veriler için daha farklı yöntem ve yaklaşımlar gerekmektedir.

Büyük veri, klasik veri tabanı yönetim araçlarıyla ve veri işleme metotlarıyla işlenemeyecek kadar büyük miktardaki ve karmaşıklıktaki veri topluluklarına verilen addır.

2.1. Veri Madenciliği

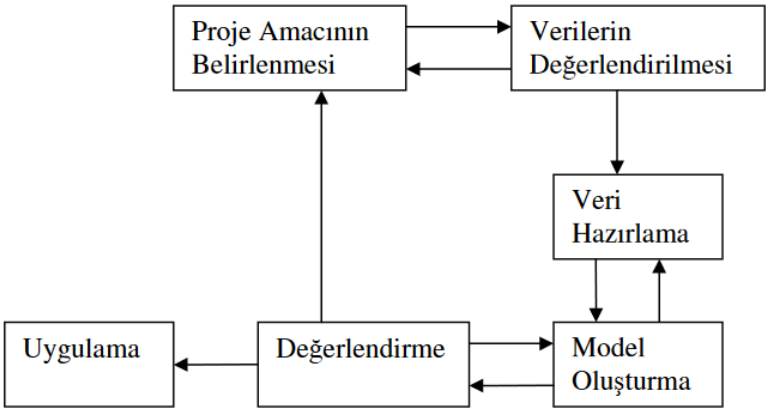
Veri madenciliği veriden anlamlı bilgiler çıkarma işlemidir. Veri madenciliği sayesinde işletmeler, kurumlar veya veri üzerinde çalışmalar yapan kişiler yaptıkları çıkarımlar ile daha etkili ve verimli kararlar alabilirler.

Veri madenciliği süreci döngüsel olarak ilerleyen bir süreçtir. Ve sürekli yenilenmesi, ilerlemesi gerekmektedir [72]. Bu sürecin adımları:

- Proje amacının belirlenmesi
- Verilerin değerlendirilmesi
- Veri hazırlama

- Model oluşturma
- Değerlendirme
- Uygulama

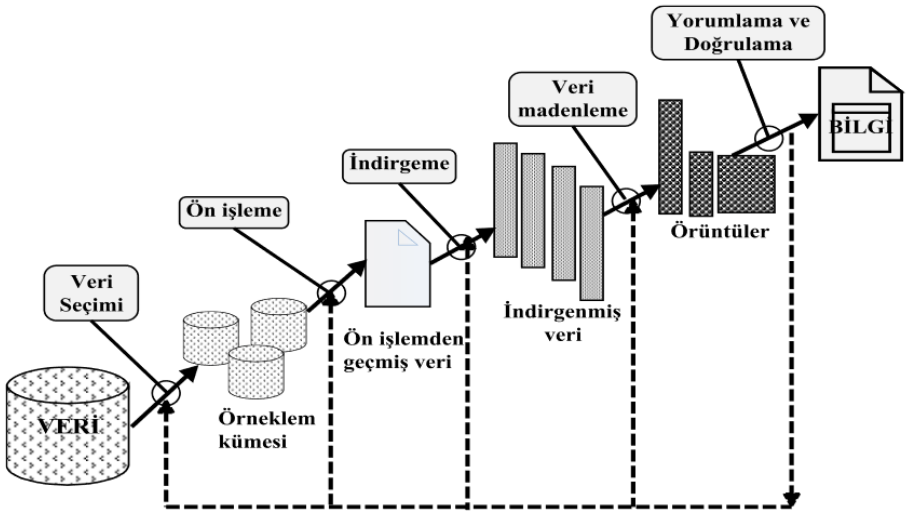
olarak belirlenebilir [73]. Bu süreç Şekil 2.1'de gösterilmiştir.



Şekil 2.1 Veri madenciliği süreci [73]

2.1.1. Veri Tabanından Bilgi Keşfi

Veri tabanından bilgi keşfi, veri tabanından bilgiye ulaşmak için gereken süreçlerin tamamını kapsar. Veri madenciliği ise bu süreçteki en önemli adımdır ve verilerden modeller veya örnekler üretmek için veri analizleri ve keşif algoritmaları kullanmayı gerektirir. Veri tabanından bilgi keşfi süreci birkaç aşamadan oluşan etkileşimli ve iteratif bir süreçtir [74]. Bu aşamalar veri seçimi, veri temizleme ve önişleme, veri indirgeme, veri madenciliği ve değerlendirme aşamalarıdır [75]. Şekil 2.2 veri tabanından bilgi keşfi sürecini göstermektedir.



Şekil 2.2 Veri tabanından bilgi keşfi süreci [76]

Veri seçimi, veri kümesinden uygun örneklem kümesinin elde edildiği adımdır. Veri temizleme ve önişleme, seçilen örneklemdeki hatalı örneklerin çıkarıldığı ve eksik nitelik değerlerinin değiştirildiği aşamadır. Bu aşamada keşfedilen bilginin kalitesi artırılır. Veri indirgeme, seçilen örneklemdeki alakasız niteliklerin atıldığı ve tekrarlı örneklerin saptandığı aşamadır. Bu sayede, veri madenciliğinin çalışma süresi kısaltılır ve daha iyi bir performansta çalışması sağlanır. Veri madenciliği, belirlenen veri madenciliği işleminin yapılmasıdır. Değerlendirme ve sonuç, sonuçların ve keşfedilen bilgilerin değerlendirilme aşamasıdır [77].

Veri madenciliği görevleri genel olarak üçe ayrılabilir. Bunlar:

- Fonksiyon tahminleme
- Sınıflandırma
- Kümeleme

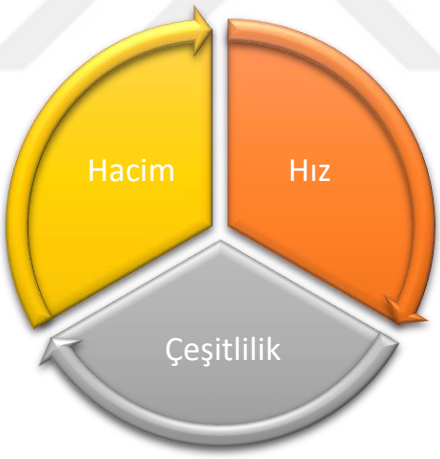
Fonksiyon tahminleme; değer tahmini veya regresyon gibi farklı şekillerde de adlandırılabilir. Değişkenler arasındaki ilişkiyi bulmaya yarayan bir araçtır. Fonksiyon tahminlemede amaç, girdilerden elde edilen bilgilerle bir çıktı değeri elde etmektir. Bu çıktı değeri bir matematiksel fonksiyon sonucu gibidir, bir sayıdır. Bu sayı verinin türüne göre elde edilen bir bilgidir. Veri kümesine göre bir ürünün fiyatı olabilir veya bir verimlilik oranı olabilir.

Sınıflandırmada eldeki veriler belirli sınıflara ayrılmıştır. Çıktı değeri kesiklidir. Eldeki verilerin özelliklerine göre verinin hangi sınıfa ait olduğu tahmin edilmeye çalışılır. Örneğin bir banka verisindeki müşterilerin sınıflandırılması olabilir.

Kümeleme, sınıflandırmaya benzemektedir. Fakat burada, önceden belirlenmiş sınıflar yoktur. Kümeleme verileri benzerliklerine göre kümelere ayırır. Kümelemenin nasıl yapılacağı, kaç kümeye ayrılacağı gibi konular değişiklik gösterebilir. Bir banka verisindeki müşterilerin benzerliklerine göre kümelenmesi örnek gösterilebilir.

2.2. Büyük Veri

Bir verinin büyük veri olarak nitelendirilebilmesi için bazı özelliklere sahip olması gerekmektedir. Bu özellikler genel olarak Şekil 2.3'deki gibi çeşitlilik, hacim ve hız olarak belirtilmektedir [78]. Ayrıca değişkenlik ve karmaşıklık da bu özelliklere eklenebilir [79]. Güvenilirlik, değer, görselleştirme gibi eklemelere de literatürde rastlanmaktadır.



Şekil 2.3 Büyük verinin temel üç özelliği [80]

Çeşitlilik: Veri sadece bir kaynaktan toplanmak zorunda değildir. Çeşitli internet siteleri, kayıt dosyaları, sosyal medya, epostalar, sensörler gibi birden fazla farklı kaynaklardan farklı yapılarda ve farklı miktarlarda veriler toplanabilir [79].

Hacim: Hacim veri miktarındaki büyüklüğü temsil eder. Verinin türüne göre veri tabanlarının büyüklükleri gigabaytları, terabaytları, petabaytları hatta zetabaytları bulabilir. Çok popüler olan sosyal medya siteleri bir günde terabayt büyüklüğünde veri kaydetmektedir. Sadece metin içeriklerinin dışında video, resim gibi çok yer kaplayan dosyalarla çalışmaları bunda büyük etkiye sahiptir. Resim ve video gibi büyük boyuttaki

verilerle çalışan bir veri tabanı için 1 TB çok küçük bir veri hacmiyken, sadece metin verisiyle çalışan bir veri tabanı için 1 GB çok büyük bir hacimdir [78, 79, 81].

Hız: Hız verinin hangi sıklıklarla ne kadar miktarda geldiğiyle alakalıdır. Burada önemli bir nokta verilerin sadece hangi sıklıkla geldiğiyle ilgilenilmemesi, ayrıca hangi miktarda geldiğiyle de ilgilenilmesidir. Örneğin bir sensör verisini düşünürsek, bir sensör ölçtüğü her değeri hemen veri olarak gönderiyor olabilir veya bunun yerine ölçüm değerlerini biriktirip belirli zaman periyotlarında da gönderebilir. Bazı büyük veri çalışmalarında veriler gerçek zamanlı olarak işlenmektedir. Bu yüzden verilerin geliş hızları ve işlenme hızları büyük öneme sahiptir [79, 81].

Değişkenlik: Değişkenlik veri akışındaki tutarsızlıkları dikkate alır. Veriler zaman içerisinde değişkenlik gösterebilir. Verilerin evirilmesiyle bu şekilde değişkenlikler gelişebilir. Bunun dışında verilerin toplanma kaynaklarından ve yöntemlerinden dolayı da çeşitlilikler gözlemlenebilir. Farklı kaynaklardan farklı şekilde bilgiler gelebilir [79].

Karmaşıklık: Karmaşıklık büyük verilerin çok sayıda kaynak aracılığıyla üretildiğini ifade eder. Farklı veri kaynaklarından gelen verileri bağlamak, eşleştirmek, temizlemek ve dönüştürmek gerekmektedir [81].

Büyük verinin tanımlanmasında kullanılabilecek bu özellikler birbirleriyle alakalıdır ve iç içedir. Bir özellikteki değişim diğer özellikleri de etkilemektedir.

2.3. Büyük Veri Sistemleri

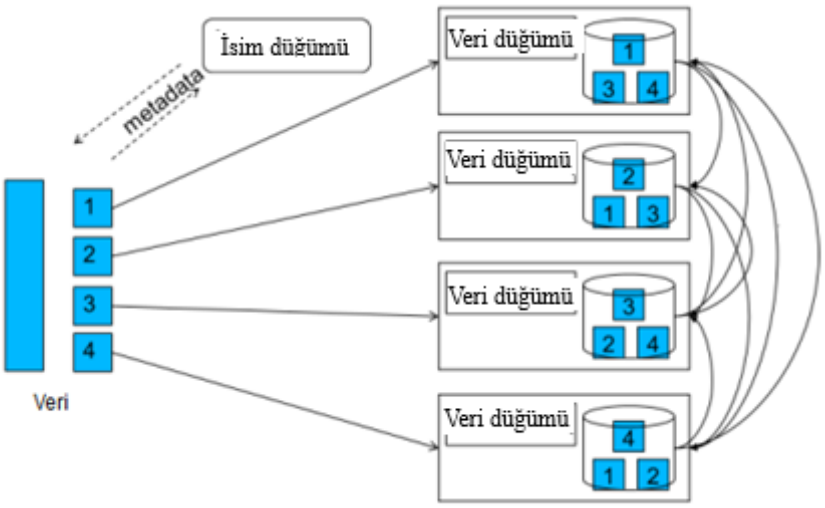
Büyük veri üzerinde çalışmak için bazı sistemler kullanılabilir. Bu sistemler sayesinde veri madenciliği dağıtık ve paralel şekilde yapılabilmektedir. Böylece normal şartlarda imkânsız olan veya çok uzun süreler ve kaynaklar isteyen işlemler daha kolay ve hızlı bir şekilde yapılabilmektedir.

2.3.1. Apache Hadoop

Hadoop büyük veri kümelerini farklı bilgisayarlara veya disklere dağıtmaya yarayan ve veri üzerindeki işlemleri dağıtık olarak çalıştırdıktan sonra birleştirerek sonuç veren bir sistemdir [82]. Büyük oranda Yahoo ve Google tarafından geliştirilmiştir. Yahoo ve Google dışında da katkı sağlayan firmalar vardır [83]. Hadoop çeşitli ana bileşenlerden oluşmaktadır. Bu ana bileşenler:

- Hadoop dağıtık dosya sistemi (Hadoop distributed file system)
- Haritalama ve indirgeme (MapReduce)
- Kaynak yöneticisi (Yet another resource negotiator)

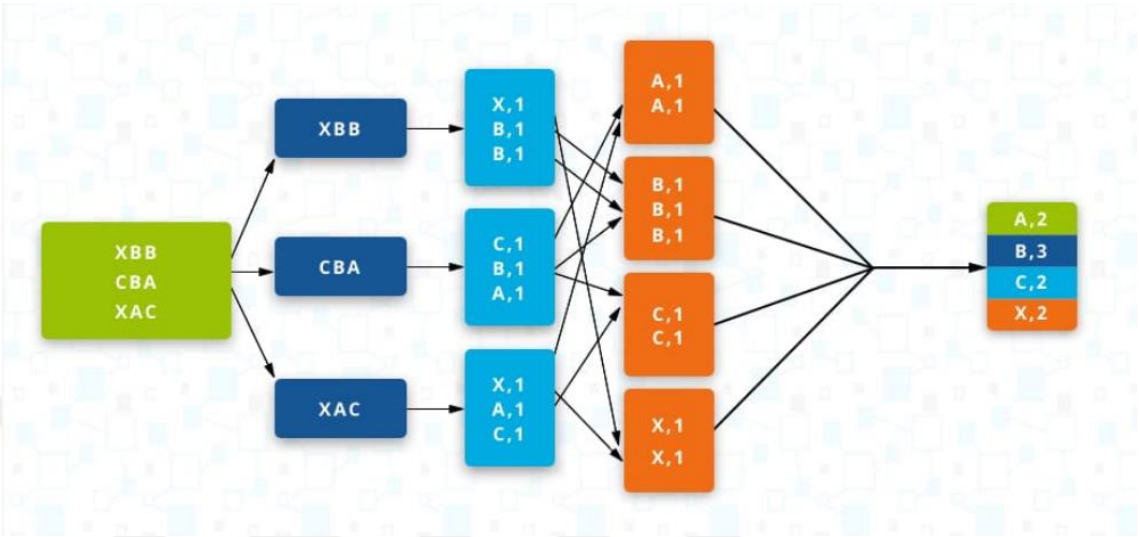
Hadoop Dağıtık Dosya Sistemi (HDFS): Hadoop dağıtık dosya sistemi (Hadoop distributed file system), Hadoop'un ana depolama sistemidir. Çok büyük miktardaki verileri farklı kaynaklara dağıtmaya yarar. Farklı kaynaklardaki veriler üzerinde işlem yaparken veriyi tek bir kaynaktaymış gibi görebilmeye yarar. Verinin türüne göre veya kullanım amaçlarına göre bir veri parçasının kopyasını farklı kaynaklarda tekrar tutabilir. Örneğin Facebook sunucularından birisi arızalandığında aynı veriler farklı bir sunucudan karşılanabilir. HDFS iki temel bileşene sahiptir. Bunlar isim düğümünden (NameNode) ve veri düğümünden (DataNode) oluşmaktadır. İsim düğümü, veri tutmamaktadır, verilerin konumlarından oluşan indeks değerlerini tutmaktadır. Veri düğümü ise veri tutmaktadır [84]. Bir HDFS örneği Şekil 2.4'teki gibidir.



Şekil 2.4 HDFS [85]

Haritalama ve İndirgeme (MapReduce): İşlemlerin farklı kaynaklardaki veriler üzerinde çalışmasını ve elde edilen sonuçların birleştirilip ana sonucun elde edilmesine yarayan bir çerçevedir. Haritalama (Map) ve İndirgeme (Reduce) fonksiyonlarından oluşur. Haritalama (Map), dağıtılmış veya ayrılmış veri parçalarında istenen işlemin gerçekleşmesidir. Bu işlemler paralel olarak yapılır. Böylece işlemler daha hızlı olur. İndirgeme (Reduce), haritalamadan çıkan sonuçların birleştirilmesine ve tek sonuçta

toplanmasına verilen addır [86]. Bir kelime sayma uygulamasında MapReduce kullanımı Şekil 2.5'teki gibidir.



Şekil 2.5 MapReduce örneği [87]

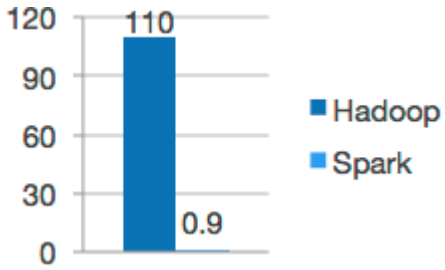
Kaynak Yöneticisi (Yet Another Resource Negotiator): YARN, HDFS ile MapReduce arasındaki koordinasyonu ve yönetimi sağlayan sistemdir. Eğer birden fazla kullanıcı sistem üzerinde işlem yaparsa bunu en optimum şekilde ayarlamaya çalışır [88].

Ayrıca Hadoop bu üç bileşenler dışında farklı bileşenlerle de çalışabilir. Apache Pig, Apache Hive, Apache Mahout, Apache Kafka gibi bileşenlere de sahiptir. Apache Pig ve Hive, SQL benzeri sorgularla verilerden sorgular yapmaya yarar. Apache Mahout ise Hadoop üzerinde makine öğrenme algoritmalarının çalıştırılmasına yarar. Apache Kafka ise gerçek zamanlı anlık veri işleme aracıdır.

2.3.2. Apache Spark

Apache Spark büyük veriler üzerinde analizler gerçekleştirmek, sorgular yapmak ve makine öğrenme kütüphaneleri ile işlemler gerçekleştirmek için kullanılan bir çerçevedir. Apache Spark, Hadoop'a göre daha gelişmiş ve hızlıdır. Hadoop'tan daha geniştir. SQL sorgulamaları, makine öğrenme kütüphaneleri kendi içerisinde mevcuttur. Mimarisi Java programlama dili ile yazılmıştır. Fakat Java dışında da programlama dilleri ile çalıştırılabilir. Bunlar Scala, Python ve R programlama dilleridir. Apache Hadoop'tan en büyük farkı hızlı olmasıdır. Bunun sebebi ise RAM üzerinden çalışmasıdır. Hadoop'ta

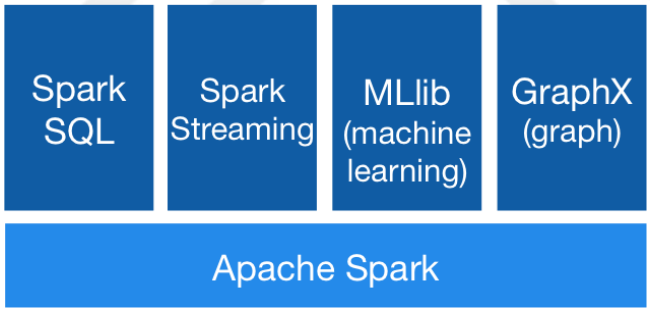
işlemler sürekli diske yazılır ve diskten okunur. Bu durum Hadoop'u yavaş kılar. Şekil 2.6'de Hadoop ile Spark'ın bir doğrusal regresyon örneğindeki hız karşılaştırmasına ait bir örnek verilmiştir. Spark, Hadoop'a göre yüz kat daha hızlı çalışabilir.



Şekil 2.6 Hadoop ve Spark hız karşılaştırması [89]

Spark dört farklı bileşenden oluşmaktadır. Bunlar:

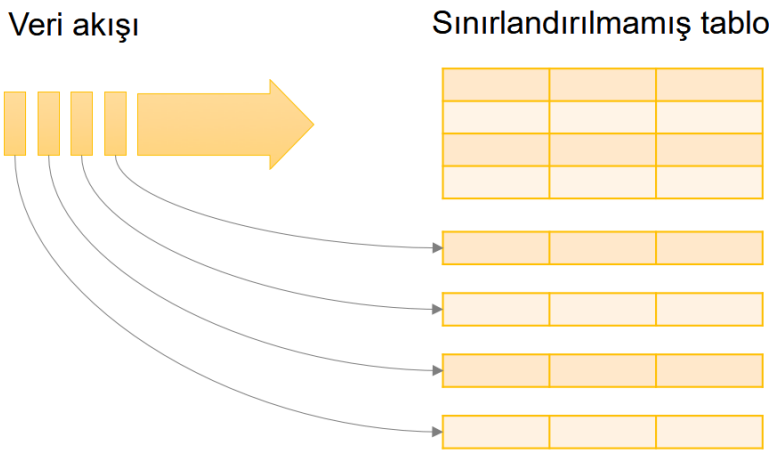
- Spark SQL
- Spark Streaming
- Spark Mllib
- Spark GraphX'den oluşmaktadır.



Şekil 2.7 Spark bileşenleri [90]

Spark SQL: Spark üzerinde SQL sorgular çağırmak için kullanılan bir araçtır[91].

Spark Streaming: Spark Streaming, gerçek zamanlı anlık veriler üzerinde çalışmaya yarar. Gelen anlık veriler belirli bir zaman aralığında toplanır ve işlenir. Toplanan veriler Şekil 2.8'deki gibi kümülatif olarak tutulur ve veri tablosuna eklenir. Zaman aralığı çok kısa olduğu için gerçek zamanlı ve sürekli devam eden bir işlem olmaktadır [92].



Şekil 2.8 Spark Streaming veri kaydetme [93]

Spark Mllib: Spark Mllib, Spark içerisindeki makine öğrenme kütüphanesidir. Bu kütüphane sayesinde nitelik seçimi, veri içerisindeki ön işlemler, makine öğrenme algoritmaları gibi işlemler gerçekleştirilebilir. Bu kütüphanede çok sayıda algoritma ve yöntem mevcuttur. Sınıflandırma, kümeleme, fonksiyon tahmini gibi işlemler bu kütüphane üzerinde kolaylıkla yapılabilir [89].

Spark Mllib kendi içerisinde çeşitli bölümlere ayrılmıştır. Bu bölümler:

- Basit istatistik
- Bağlantı hattı
- Veri dönüşümü, seçimi
- Sınıflandırma ve regresyon algoritmaları
- Kümeleme algoritmaları
- Birliktelik kuralı

Basit istatistik: Korelasyon, ki-kare hipotez testi, p değeri, minimum değer, maksimum değer, ortalama, varyans, standart sapma, toplam sayı gibi değerlerin görülebileceği kısımdır.

Bağlantı hattı (pipeline): Veri ön işleme, dönüştürme, bir algoritma ile eğitme gibi adımları birbirine bağlayan yapıdır.

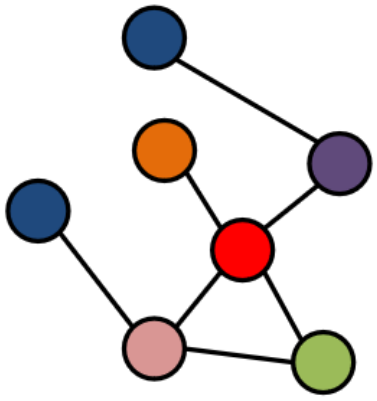
Veri dönüşümü, seçimi: Ham veriden özellik çıkarma, veri dönüşümü, indeksleme, normalleştirme, veri seçimi, bir metinden kelimeleri ortaya çıkarma, vektör haline çevirme, sayma gibi işlemleri gerçekleştiren yapıdır.

Sınıflandırma ve regresyon algoritmaları: Sınıflandırma algoritmaları olarak lojistik regresyon, karar ağacı, rastgele orman, yapay sinir ağları, destek vektör makinesi gibi algoritmalar vardır. Regresyon algoritmaları olarak doğrusal regresyon, karar ağacı, rastgele orman, izotonik regresyon gibi algoritmalara sahiptir.

Kümeleme algoritmaları: Kümeleme için kullanılabilecek k-ortalamalar, LDA, Gauss karışım modeli gibi algoritmalara sahiptir.

Birliktelik kuralı: Birliktelik kuralı için FP-Growth, PrefixSpan gibi algoritmalara sahiptir.

Spark GraphX: Spark GraphX, Spark'ın graf kütüphanesidir. Grafik anlamında kullanılmamaktadır. Düğüm ve bağlantılar ile birbirine bağlanan ağlara karşılık gelen graf kelimesini karşılamaktadır. Şekil 2.9'te bir örneği sunulmuştur. Bu kütüphane sayesinde birbirlerine bağlı ağ yapısında olan veriler üzerinde kolay ve hızlı bir biçimde işlemler gerçekleştirilebilmektedir [94].



Şekil 2.9 Bir graf örneği [95]

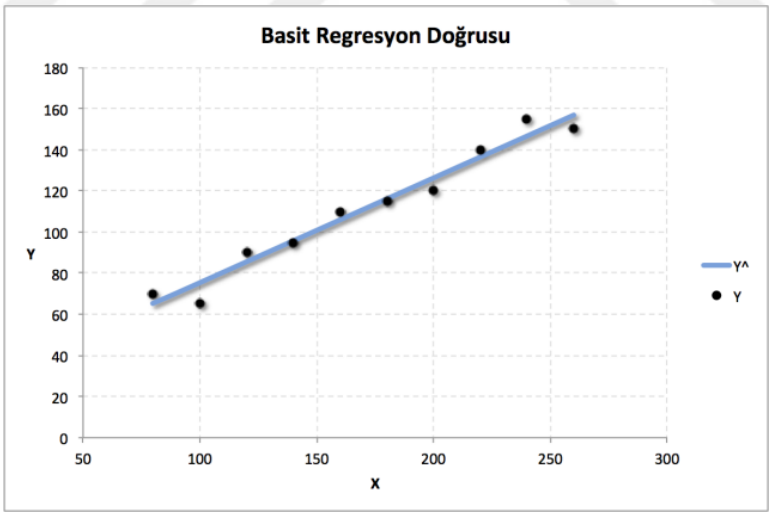
Spark içerisinde çok sayıda makine öğrenme algoritması vardır. Bunlardan tez kapsamında ele alınan sınıflandırma ve regresyon algoritmalarına aşağıda kısaca değinilmiştir.

2.3.3. Doğrusal Regresyon

Regresyon çok sayıdaki değişkenler arasındaki ilişkiyi ölçmek için yapılan bir analiz yöntemidir. Doğrusal regresyonda değişkenlere doğrusal olarak katsayılar verilerek en iyi sonuç elde edilmeye çalışılır. Basit ve çoklu regresyon olarak ikiye ayrılabilir. Basit regresyonda bir değişkene bağlı regresyon analizi yapılır, çoklu regresyonda ise çok sayıdaki değişkenlere bağlı olarak regresyon analizi yapılır. Basit doğrusal regresyon, formül 4.1'de görüldüğü gibidir. Çoklu regresyon ise formül 4.2'den görülebilir. Basit bir doğrusal regresyonun grafiği ise Şekil 2.10'de görüldüğü gibidir. x değerlerine bağlı olarak y değerleri tahmin edilmiştir ve regresyon doğrusu gerçek değerler ile tahmin edilen değerler arasındaki farkı en aza indirmiştir. Apache Spark bünyesindeki doğrusal regresyon algoritması regresyon (fonksiyon tahmini) için mevcuttur, sınıflandırma problemleri için mevcut değildir.

$$y = ax + b \tag{2.1}$$

$$y = a_1x_1 + a_2x_2 + a_3x_3 + \cdots + b \tag{2.2}$$



Şekil 2.10 Basit doğrusal regresyon [96]

2.3.4. Lojistik Regresyon

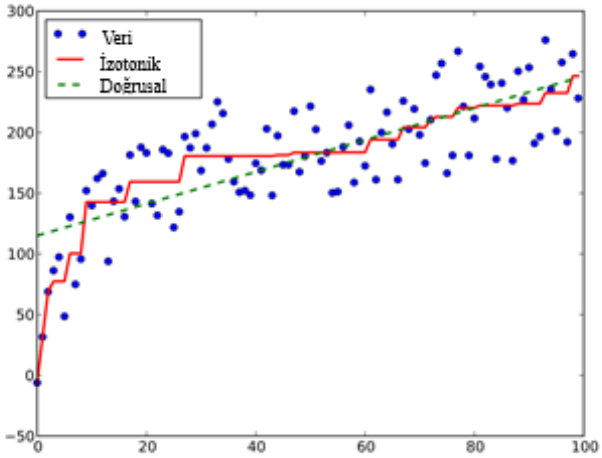
Sınıflandırma problemi için kullanılabilecek bir yöntemdir. Bağımlı ve bağımsız değişkenler arasındaki ilişkiyi modellemeyi amaçlamaktadır. Amaçlanan sınıf sayısına göre ikili veya çoklu olarak ayrılabilir. Genellikle iki sınıflı problemlerde tercih edilir.

Hangi sınıfa ait olunduğuyla alakalı olasılıksal tahmin yapar ve yüksek olasılığı seçer. Lojistik regresyon lojistik fonksiyonundan diğer adıyla sigmoid fonksiyonundan yararlanır. Lojistik fonksiyonu her zaman 0 ile 1 arasında değer alır. Sınıfların olasılıklarının birbirine olan oranı logaritmik fonksiyon içine alınır ve lineer regresyon formülüne benzer bir formül elde edilir. Elde edilen olasılık sonucuna göre tüm sınıflar test edilir ve en yüksek olasılıklı sınıf seçilir. Formülü Eşitlik (2.3)'de olduğu gibidir.

$$P = \frac{e^{a+bx}}{1 + e^{a+bx}} \tag{2.3}$$

2.3.5. İzotonik Regresyon

İzotonik regresyon tahminlemeyi serbest bir şekilde yapan bir algoritmadır. Her yerde sürekli olarak artış veya azalış göstermez. Verilerin durumuna göre en iyi noktadan geçmeyi hedeflemektedir. Bu yüzden doğrusal regresyon gibi bir forma sahip değildir. Ancak küçük aralıklarda benzer formlar gösterebilir. Farklı aralıklarda farklı formlara sahiptir. Grafiği Şekil 2.11'deki gibidir.



Şekil 2.11 İzotonik regresyon [97]

2.3.6. Karar Ağacı

Popüler algoritmalardan birisidir. Değişkenler üzerinden bir ağaç oluşturur ve değişkenlerin aldıkları değerlere göre ağaç dallanır. Ağacın dallanması ile veriler daha

küçük kümelere bölünür ve sonuç alınır. Karar ağaçlarının avantajlı yanları vardır. Anlamlandırılması ve yorumlanması kolaydır. Ağaç yapısı görselleştirilebilir ve görülebilir. Hem regresyon tipinde hem de sınıflandırma türünde uygulanabilir. Dezavantajları olarak karmaşık ağaçlar üretebilirler. Veriyi ezberleme söz konusu olabilir. Ağacın parametrelerini iyi ayarlamak gerekebilir.

Karar ağacında ağacın nasıl bölüneceği sonuca etki etmektedir. Ağaçta alt düğümlere gidildikçe yani derinlik arttıkça homojenliğin artması beklenir. Böylece doğruluğu daha yüksek bir sonuç elde edilir.

Entropi verideki belirsizliği tanımlar. Veri kümesindeki benzerlik ne kadar fazlaysa, veri ne kadar homojen ise entropi o kadar düşük çıkacaktır. Veri ne kadar iyi bölünürse entropi o kadar düşük olur. Entropi formülü Eşitlik (2.4)'deki gibidir.

$$H = -\sum p(x) \log p(x) \tag{2.4}$$

H entropiyi, p(x) ise belirli bir sınıfa ait yüzdeleri belirtir.

Entropiyi en aza indirmek için bilgi kazançları kullanılır. Bilgi kazançları Eşitlik (2.5)'deki formülle hesaplanır.

$$Kazanç\ (S,V) = H(S) - \sum_{V \in S} \frac{|V|}{|S|} H(V) \tag{2.5}$$

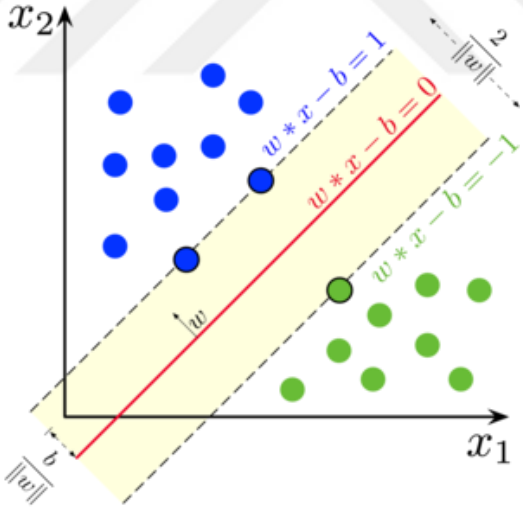
Formüldeki S veri kümesidir. V bölünmüş veri parçasıdır. V'lerin toplamı S'yi oluşturur. Bu formüle göre ağaç dallanmasının nasıl yapılacağına karar verilir ve en verimli şekilde ağaç oluşturulur. Basit bir karar ağacı örneği Şekil 2.12'de görülebilir.



Şekil 2.12 Karar ağacı örneği [98]

2.3.7. Doğrusal Destek Vektör Makinesi

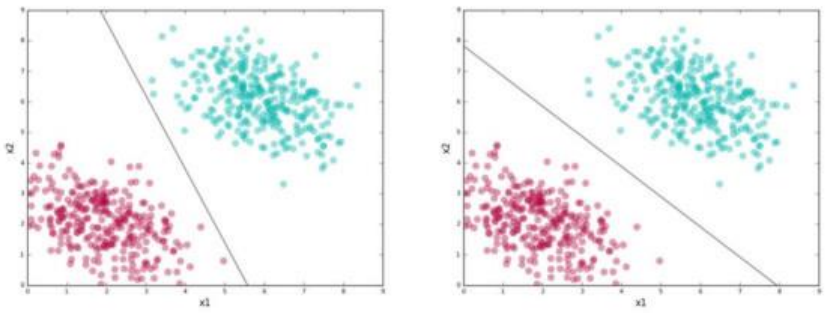
Destek vektör makinesi verileri bulundukları uzayda en iyi şekilde bölmeye ve sınıflandırmaya çalışır. Sınıflar arası ayrımı en iyi şekilde yapmak için bir vektör ile sınıfları ayırır. Bu ayrım Şekil 2.13'deki gibi gösterilebilir.



Şekil 2.13 Destek vektör makinesi [99]

Sınıfları ayırmak için kullanılabilecek vektörler sonsuz sayıda çizilebilir. Fakat bunlar içerisinde en iyisini bulmak gerekir. En iyi vektörün diğerlerine göre sahip olması gereken üstünlük, vektöre en yakın konumda bulunan farklı sınıflara ait noktaların diğer alternatiflere göre uzak olmasıdır. Bu uzaklık değerlerine marj denir. En yüksek marja sahip olan vektör en iyisidir. Örnek olarak Şekil 2.14'da görülebilir. İki alternatifte de

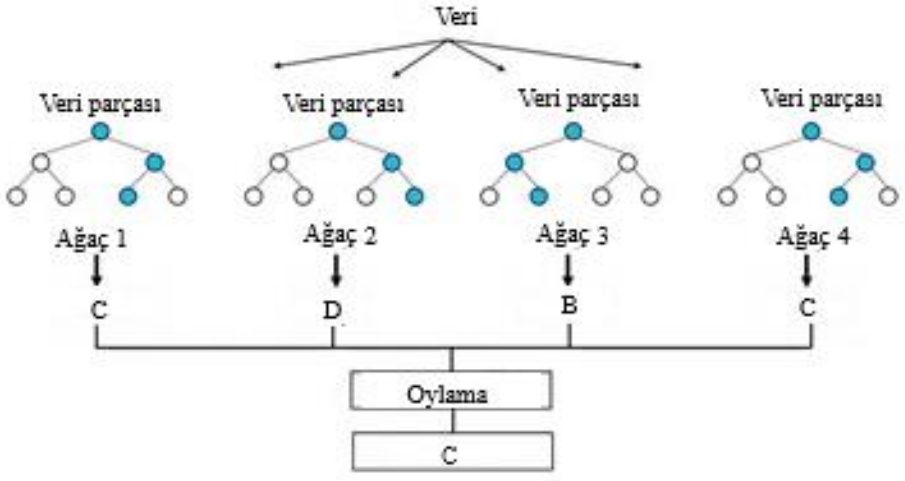
başarılı bir sınıflandırma yapılmış olsa da sağdaki vektörün marjı yüksektir ve veriler arttıkça daha başarılı sonuçlar verir.



Şekil 2.14 Destek vektör makinesi alternatif vektörler [100]

2.3.8. Rastgele Orman

Rastgele orman algoritması bir karar ağacı algoritmasıdır. Eğitim veri kümesini küçük parçalara böler ve her bir küçük parça için bir karar ağacı çıkartır. Bu karar ağaçlarını test kümesi üzerinde test eder ve en çok tercih edilen sınıfı seçer. Çoğunluğun kararına göre karar verir. Çok sayıda karar ağacı barındırdığı için ve küçük veri parçalarını rastgele seçtiği için rastgele orman adını almıştır. Şekil 2.15'de görüldüğü gibidir.

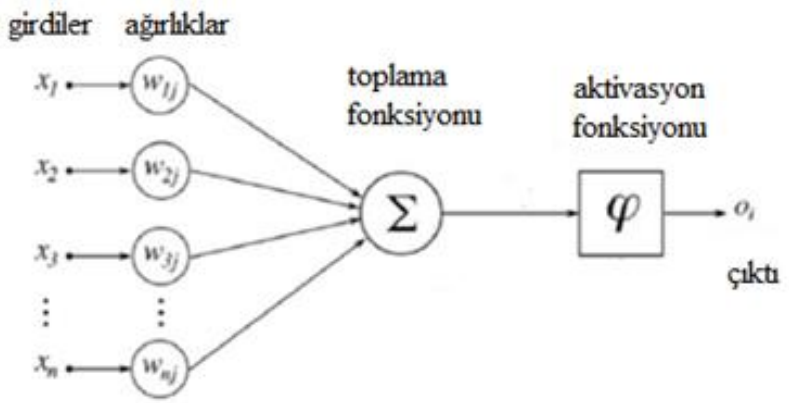


Şekil 2.15 Rastgele orman algoritması

2.3.9. Yapay Sinir Ağları

Yapay sinir ağları insan beynindeki sinirlerin çalışma prensibinden esinlenilerek öğrenme, hatırlama, tecrübe edinme, tahmin yürütme gibi yetenekleri kullanmaya imkân sağlayan bir yöntemdir [101]. Yapay sinir ağları insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, oluşturabilme, keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacıyla geliştirilen bilgisayar sistemleridir [102].

Yapay Sinir Hücresi: Biyolojik sinir ağının sinir hücrelerinden oluşması gibi, yapay sinir ağları da yapay sinir hücrelerinden oluşmaktadır. Bir yapay sinir ağı birbirleriyle bağlantılı çok sayıdaki yapay sinir hücrelerinden meydana gelmektedir. Yapay sinir hücreleri biyolojik sinir hücrelerinin basit bir modelidir. Bir yapay sinir hücresinin beş temel elemanı vardır. Bunlar; girdiler, ağırlıklar, toplama fonksiyonu, aktivasyon fonksiyonu ve çıktıdır. Bir yapay sinir hücresinin yapısı Şekil 2.16'de gösterilmiştir.



Şekil 2.16 Yapay sinir hücresi [103]

Girdiler: Girdiler, yapay sinir hücresine dışarıdan gelen bilgilerdir. Bu bilgiler; veri kümesinin elemanları, başka bir sinir hücresinden gelen çıktılar olabileceği gibi kendinden gelen bir bilgi de olabilir.

Ağırlıklar: Ağırlıklar, sinir hücresi tarafından alınan girdilerin sinir hücre gövdesine iletilmeden önceki etkisini belirleyen değerlerdir. Ağırlıkların küçük veya büyük değerde olması o girdinin önemli veya önemsiz olduğu anlamına gelmez. Bu ağırlıklar negatif değerler de alabilirler. Yapay sinir ağları eğitiminde algoritmalara göre ağırlık değerleri değişkenlik gösterebilir. Ağırlık değerleri değiştikçe ağın performansı değişir.

Toplama fonksiyonu: Toplama fonksiyonu ile hücreye gelen net girdi hesaplanır. Bu hesaplama için farklı fonksiyonlar kullanılabilir. Bu fonksiyonların en yaygın kullanılanı ağırlıklı toplam fonksiyonudur. Bu fonksiyonda her girdi değeri kendi ağırlık değeriyle çarpılarak toplanır. Bu toplam net girdiyi ifade eder. Net girdinin formülü Eşitlik (2.6)'de gösterildiği gibidir.

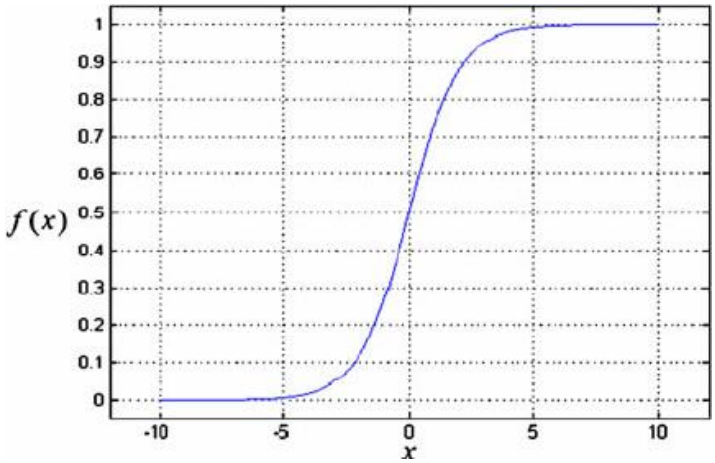
$$Net\ girdi = \sum_{i=1}^{n} Gi\ Ai \tag{2.6}$$

Bu denklemde *Gi i*. girdiyi, *Ai i*. girdinin ağırlığını, *n* ise hücreye gelen toplam girdi sayısını ifade etmektedir.

Aktivasyon fonksiyonu: Aktivasyon fonksiyonu, toplama fonksiyonundan gelen girdiyi işleyerek yapay sinir hücresinin çıkışını belirler. Transfer fonksiyonu olarak da bilinir. Çeşitli tipleri vardır ve genellikle doğrusal olmayan bir fonksiyondur. Doğrusal olarak kullanılmamasının sebebi doğrusal fonksiyonlarda girdi ile çıktının doğru orantılı olmasıdır. Bu durum ilk yapay sinir ağları denemelerinde başarısız sonuçların alınmasına sebep olmuştur. En sık kullanılan aktivasyon fonksiyonu sigmoid ve tanjant hiperbolik aktivasyon fonksiyonlarıdır. Farklı türde aktivasyon fonksiyonları mevcuttur [102]:

Sigmoid fonksiyonu; sürekli, doğrusal olmayan ve türevi alınabilir bir fonksiyondur. En çok kullanılan fonksiyondur. Her bir girdi değeri için 0 ile 1 arasında bir çıktı değeri vermektedir. Formülü Eşitlik (2.7)'daki gibidir. Koordinat düzlemindeki görünümü de Şekil 2.17'de gösterilmiştir.

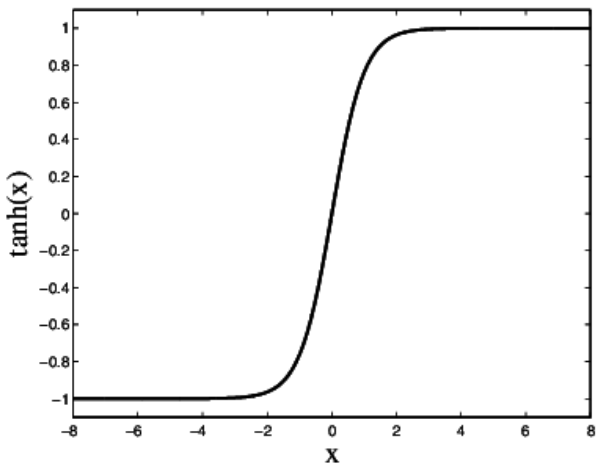
$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.7}$$



Şekil 2.17 Sigmoid fonksiyonu [104]

Hiperbolik tanjant fonksiyonu; sürekli, türevlenebilir ve doğrusal olmayan bir fonksiyondur. Sigmoid fonksiyonuna benzer. Sigmoid fonksiyonu 0 ile 1 arsında değer alırken hiperbolik tanjant fonksiyonu -1 ile 1 arasında değer alır. Bu yönüyle sigmoid fonksiyonundan farklıdır. Formülü Eşitlik (2.8)'daki gibidir. Gösterimi ise Şekil 2.18'de gösterilmiştir. Şekil olarak da sigmoid fonksiyonuna benzemektedir.

$$f(x) = \frac{e^{x} + e^{-x}}{e^{x} - e^{-x}} \tag{2.8}$$

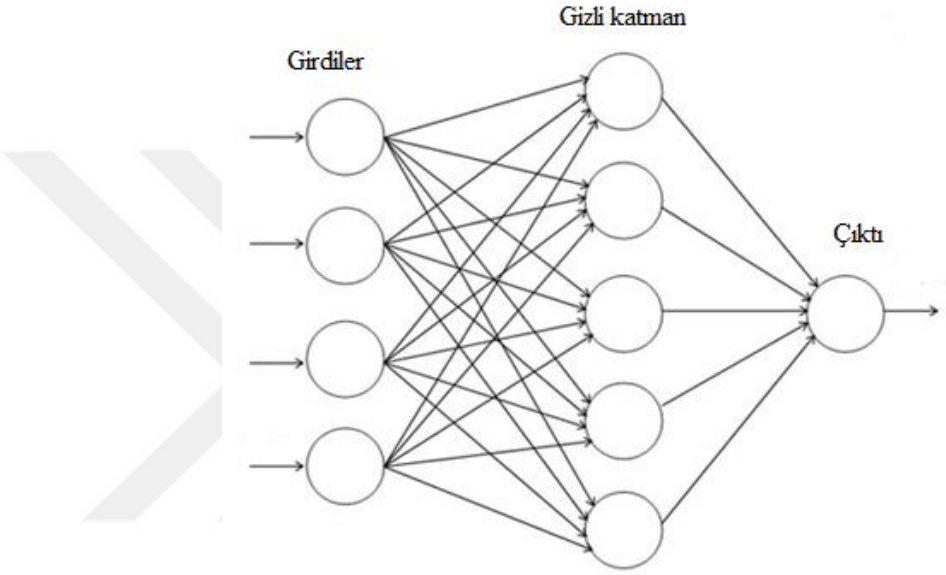


Şekil 2.18 Hiperbolik tanjant fonksiyonu [105]

Çıktı: Çıktı değeri, yapay sinir hücresinin verdiği sonuçtur. Diğer bir hücreye girdi olarak gönderilebileceği gibi kendi kendine de gönderilebilir.

Yapay Sinir Ağlarının Yapısı

Yapay sinir hücreleri çok sayıda bir araya gelip, birbirleriyle bağlantılı şekilde çalıştıkları zaman yapay sinir ağını oluştururlar. Bir ağda çok sayıda hücre bulunur. Bu hücrelerin bir kısmı aynı doğrultu üzerinde bulunurlar. Bu aynı doğrultu üzerindeki hücreler topluluğuna katman denir. Temel olarak YSA'da üç katman vardır. Bunlar girdi katmanı, gizli katman ve çıktı katmanıdır. Bir yapay sinir ağının yapısı ve katmanlar şekilsel olarak Şekil 2.19'de gösterilmiştir.



Şekil 2.19 Yapay sinir ağlarının yapısı

Girdi katmanı: Dış dünyadan gelen bilgileri alırlar ve gizli katmana iletirler. Girdileri hücrelerin dizayn edilişine göre birebir aynı şekilde veya farklılaştırarak gönderebilirler.

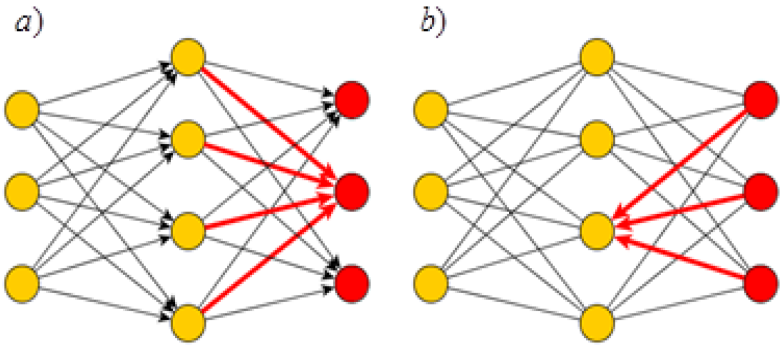
Gizli katman: Tek bir katmandan da oluşabilir, birden fazla katmandan da oluşabilir. Girdi katmanından gelen bilgiyi işler ve çıktı katmanına iletir.

Çıktı katmanı: Çıktı katmanı, gizli katmandan gelen bilgileri işler ve yapay sinir ağının genel çıktısını verir. Çıktı katmanında tek hücre olabileceği gibi birden fazla hücre de olabilir. Problemin türüne göre çıktı değerlendirilir.

Geri Yayılım Algoritması

Yapay sinir ağı eğitiminde en çok kullanılan algoritmadır. Hataları çıktı katmanından girdi katmanına geriye doğru yönlendirdiği için geri yayılım ismini almıştır. Geri yayılım

algoritması ileri besleme ve geri besleme olarak iki kısma ayrılabilir ve bu kısımlar Şekil 2.20'daki gibi ifade edilebilir.



Şekil 2.20 a) İleri besleme b) Geri besleme [106]

Çok katmalı bir ileri beslemeli ağ için algoritma aşağıdaki gibidir [107].

Algoritmada;

q : katman numarası

H_i^q : q'inci katmandaki i biriminin girdisi

y_i^q : q'inci katmandaki i biriminin çıktısı

δ_i^q : q'inci katmandaki i birimindeki hata miktarı

v_i^q : q'inci katmandaki i birimindeki sonuç

n : öğrenme katsayısı

w_{ij}^q : *(q-1)*'inci katmandaki i birimini, q'inci katmandaki j birimine bağlayan ağırlık değerini ifade etmektedir.

Algoritmanın adımları aşağıdaki gibidir.

1. w'ye başlangıç olarak rastgele sayılar atanır.
2. Katmanlardaki her bir eleman için ileri yöndeki çıktı değeri Eşitlik (2.9) ile hesaplanır.

$$y_i^q = f(\sum_i y_i^{q-1} \times w_i^{q-1}) \quad (2.9)$$

3. Çıkış birimleri için hata miktarları hesaplanır.

$$\delta_i^Q = \left(v_i^q - y_i^q\right) \times f'\left(H_i^q\right) \quad (2.10)$$

4. Q=2, 3, ...Q-1, Q katmanlarındaki bütün i birimleri için geriye yayılımla deltalar yani gizli katman birimleri için hata terimleri hesaplanır.

$$\delta_i^{q-1} = f'\left(H_i^{q-1}\right) \times \sum_i (\delta_i^q \times w_i^q) \quad (2.11)$$

5. Bütün ağırlıklar wij'ler kullanılarak güncellenir.

$$w_{ij}^{yeni} = w_{ij}^{eski} + \Delta w_{ij}^q \quad (2.12)$$

$$\Delta w_{ij}^q = n \times \delta_i^q \times y_i^{q-1} \quad (2.13)$$

6. 2. Adıma dönülür ve toplam hata kabul edilebilir bir düzeye gelene kadar işlemler tekrarlanır.

Geri yayılım algoritmasının çalışması için transfer fonksiyonunun türevlenebilir bir fonksiyon olması gerekmektedir. Bunun için genellikle sigmoid fonksiyonu kullanılır. Sigmoid fonksiyonu Eşitlik (2.14)'deki gibidir, türevi ise Eşitlik (2.15)'deki gibi olmalıdır.

$$f(x) = \frac{1}{1 + e^{-\beta x}} \quad (2.14)$$

$$f'(x) = \beta \times f(x) \times (1 - f(x)) \tag{2.15}$$

3. BÖLÜM

BÜYÜK VERİ ANALİZİ İÇİN ÖNERİLEN ALGORİTMALAR

Bu tez çalışmasındaki amaç Spark'ta hâlihazırda bulunan algoritmalara ek olarak bu algoritmalardan daha etkili yeni bir algoritma geliştirmektir. Algoritmalar farklı verilerde ve farklı problemlerde farklı başarılarda sonuçlar verirler. Her algoritmanın kendine özgü avantajları ve dezavantajları vardır. Farklı algoritmaların birleştirilip hibritlenmesi ile avantajlı tarafları öne çıkarılıp, eksikliklerinin kapatılmasıyla daha etkin algoritmalar elde edilebilir ve bu sayede daha yüksek başarı oranı yakalanabilir. Bu nedenle tez çalışması kapsamında iki hibrit algoritma geliştirilmiştir. Bunlar:

- Spark Hibrit (SH) algoritması
- Yapay sinir ağı - karar ağacı (YSA-KA) algoritmasıdır.

3.1. Spark Hibrit Algoritma (SH)

Karar ağacı algoritmasının çok sayıda kullanılıp rastgele orman algoritmasının elde edilmesinden esinlenilmiştir. Apache Spark'ta bulunan algoritmalar kullanılarak geliştirilen bir algoritmadır.

Eğer bir veri sadece bir model ile eğitilirse, yerel bir optimum noktaya takılabilmektedir ve farklı yerlerdeki noktaları kaçırabilmektedir. Veriyi birden fazla modelle eğitmek çeşitlilik sağlar ve bu çeşitlilik sayesinde veri içerisindeki farklılıkların üstesinden gelinebilir. Bu sayede daha doğru sonuçlar elde edilir. Ayrıca, bu modeller sadece bir algoritma yerine farklı algoritmalardan gelirse daha fazla çeşitlilik sağlanır ve sonuçlar daha da gelişir. Bundan dolayı bu algoritmada Spark'ta bulunan farklı algoritmalar uygulanmıştır. Algoritmalar tüm veriye uygulanmamış, veri kümesi algoritmalara

paylaştırılmıştır. Böylece algoritmalar eşdeğer durumda çalışarak, veri kaynaklı birbirlerine üstünlük sağlama durumu söz konusu olmamıştır.

Bu algoritmada kullanılan alt algoritmalar 2. bölümde anlatılan makine öğrenme algoritmalarıdır. Sınıflandırma için lojistik regresyon, karar ağacı, rastgele orman, doğrusal karar destek sistemi, yapay sinir ağları algoritmaları ele alınmıştır. Regresyon için ise doğrusal regresyon, izotonik regresyon, karar ağacı, rastgele orman algoritmaları kullanılmıştır. Bu algoritmalar 2. Bölüm'de daha detaylı anlatılmıştır.

Doğrusal regresyon algoritması sözde kodu Tablo 3.1'deki gibidir.

Tablo 3.1 Doğrusal regresyon algoritması sözde kodu

Veriyi oku Verinin eğim ve kesişim katsayılarını hesapla Elde edilen değerler ile doğrusal regresyon formülünü oluştur Regresyon formülü ile veriyi test et

Lojistik regresyon algoritması sözde kodu Tablo 3.2'deki verilmiştir.

Tablo 3.2 Lojistik regresyon algoritması sözde kodu

Veriyi oku Bağımlı, bağımsız değişken değerlerin ortalamasını bul Elde edilen değerleri ile lojistik regresyon formülünü oluştur Regresyon formülü ile veriyi test et

İzotonik regresyon algortiması sözde kodu Tablo 3.3'deki gibidir.

Tablo 3.3 İzotonik regresyon algoritması sözde kodu

Veriyi oku Bağımlı, bağımsız değişken değerlerin ortalamasını bul Elde edilen değerleri ile izotonik regresyon formülünü oluştur Regresyon formülü ile veriyi test et

Karar ağacı algoritması sözde kodu Tablo 3.4'deki gibidir.

Tablo 3.4 Karar ağacı algoritması sözde kodu

```
Veriyi oku
Karar ağacındaki derinlik değerine kadar:
        Verideki her bir özellik için:
                Özelliğin bilgi kazanımını hesapla
        En yüksek bilgi kazanımı sağlayan özelliği seç
        Bu özellik üzerinden ağaca dal oluştur
Karar ağacı oluştur
Karar ağacı üzerinden veriyi test et
```

Doğrusal destek vektör makinesi algoritması sözde kodu Tablo 3.5'deki gibidir.

Tablo 3.5 Doğrusal destek vektör makinesi algoritması sözde kodu

```
Veriyi oku
Verideki her bir özellik için:
        Sınıfları ayıran noktaları tespit et
        Sınıfları ayıran birbirine en yakın noktaları bul
        Bu noktalar arasına marjı en çok yapacak eğimi bul
        Bulunan eğime göre doğru oluştur ve orta noktadan veriyi ayır
Elde edilen doğrulara göre veriyi test et
```

Rastgele orman algoritması sözde kodu Tablo 3.6'deki gibidir.

Tablo 3.6 Rastgele orman algoritması sözde kodu

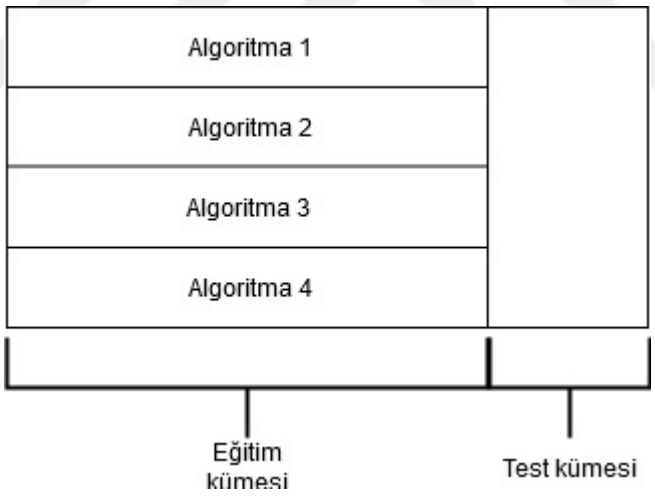
```
Veriyi oku
Veriyi belirli sayıda parçalara böl
Her bir parça için:
        Karar ağacı oluştur ve ağacı eğit
Her bir örnek için:
        Elde edilen her bir karar ağacı için:
                Örneği eğit
        Karar ağacı sonuçlarını oyla
Oylama sonuçlarını gerçek değerlerle kıyasla ve algoritmanın başarısını hesapla
```

Yapay sinir ağları algoritması sözde kodu Tablo 3.7'deki gibidir.

Tablo 3.7 Yapay sinir ağları algoritması sözde kodu

```
Veriyi oku
Uygun topolojiye göre yapay sinir ağı oluştur
İterasyon sayısı kadar:
        Yapay sinir ağını ağırlık değerleriyle ileriye doğru hesapla
        Çıktı katmanındaki ihtimalleri hesapla
        Doğru sınıf ile ihtimal arasındaki farkı hata olarak hesapla
        Geri yayılım algoritması ile bu hatayı geriye doğru dağıt
        Ağırlıkları güncelle
Son ağırlıklarla yapay sinir ağı üzerinden veriyi test et
```

SH algoritmasında ilk olarak eğitim kümesi ve test kümesi belirlenir. Eğitim kümesi, belirli sayıda parçalara ayrılır. Bu parçalara ayırma işleminde veri rastgele bir biçimde dağıtılır. Bundaki amaç, verideki çeşitliliğin algoritmalara dağıtılmasıdır. Bu durum Şekil 3.1'de gösterildiği gibidir. Ayrıca verinin kaç parçaya bölüneceği (n) algoritmanın bir parametresidir.

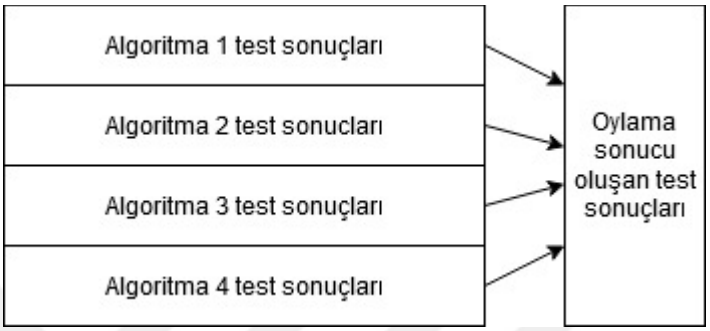


Şekil 3.1 Dört farklı algoritma ile SH algoritması

Parçalanan eğitim kümesi üzerinde veriler sırasıyla eğitilir ve eğitilmiş bir biçimde n adet model elde edilir.

Uygulanan algoritmaların eğitimi sonucu elde edilen eğitilmiş modeller sırasıyla test kümesi üzerinden test edilir. Bütün sonuçlar toplanarak oylamaya sunulur ve değerlendirilir. Şekil 3.2'de gösterildiği gibi en çok oyu alan sınıf tercih edilen sınıftır. Oylama iki alternatif şekilde yapılır. Birinci alternatifte her model aynı ağırlığa sahiptir

ve oy ağırlığı 1'dir. İkinci alternatifte her model aynı ağırlıkta değildir. Her model kendi doğruluk değeri kadar ağırlığa sahiptir ve bu ağırlıklar üzerinden sınıfa karar verilir. Bu da oylama türü olarak algoritmanın bir parametresidir. Bu parametrenin ele alınmasının amacı eğer modeller içerisinde çok kötü sonuç veren algoritmalar çıkarsa onların sonuca olan olumsuz etkisini azaltmaktır.



Şekil 3.2 SH algoritması oylama ve sonuç elde etme aşaması

SH algoritmasının parametreleri Tablo 3.8'deki gibidir.

Tablo 3.8 SH algoritması parametreleri

Parametre	**Açıklama**
Parça sayısı	Eğitim kümesinin kaç parçaya bölüneceği
Oylama türü	Modellerin eşit veya doğruluk değerlerine göre değerlendirilmesi

Spark Hibrit algoritması sözde kodu Tablo 3.9'deki gibidir.

Tablo 3.9 SH algoritması sözde kodu

```
Veriyi oku
Veriyi karıştır
Veriyi eğitim ve test kümeleri olarak ayır
Eğitim kümesini n parçaya böl
Her parçanın hangi algoritma ile çalıştırılacağını belirle
Her bir parça için:
        Eğer ilgili algoritma doğrusal regresyon ise:
                Doğrusal regresyon ile eğit (Tablo 3.1)
        Eğer ilgili algoritma lojistik regresyon ise:
                Lojistik regresyon ile eğit (Tablo 3.2)
        Eğer ilgili algoritma izotonik regresyon ise:
                İzotonik regresyon ile eğit (Tablo 3.3)
        Eğer ilgili algoritma karar ağacı ise:
                Karar ağacı ile eğit (Tablo 3.4)
        Eğer ilgili algoritma doğrusal destek vektör makinesi ise:
                Doğrusal destek vektör makinesi ile eğit (Tablo 3.5)
        Eğer ilgili algoritma rastgele orman ise:
                Rastgele orman algoritması ile eğit (Tablo 3.6)
        Eğer ilgili algoritma yapay sinir ağları ise:
                Yapay sinir ağları ile eğit (Tablo 3.7)
        Eğitilen modelin doğruluk oranını hesapla
        Eğitilen modeli test kümesi ile çalıştır ve seçilen sınıfı hafızada tut
Eğer oylama türü 1 ise:
        Hafızada tutulan sınıfları say ve maksimum olanını seç
Eğer oylama türü doğruluk ise:
        Hafızada tutulan sınıfları doğruluk değerleriyle ağırlıklandır ve seç
Seçilen sonuçlar ile test kümesindeki gerçek sonuçları karşılaştır ve doğruluk değerini
hesapla
```

3.2. Yapay Sinir Ağı ve Karar Ağacı Algoritması (YSA-KA)

Yapay sinir ağları ve karar ağacı veri madenciliğinde sıklıkla kullanılan algoritmalardır. Bu algoritmaların hibitleştirilmiş hali de yeni bir yaklaşımdır.

Yapay sinir ağları insan beyninden esinlenmiş bir yöntemdir. İnsan beynindeki nöronların tepkileri iletmesi ile bilgi akışı sağlanır. Çok sayıda nöron birleştikçe nöronlar bir ağ yapısı oluşturur ve bu yapıya sinir ağı denir.

Yapay sinir ağında bir nörona birden fazla nörondan bilgi gelir ve bir nörondan birden fazla nörona bilgi gönderilir. Bir nöronun girdileri önceki katmandaki nöronlardan gelen değerlerdir. Bu değerler ilgili ağırlık değerleriyle çarpılır ve ilgili toplama fonksiyonuyla

toplanır. Genellikle toplama fonksiyonu olarak matematiksel toplama işlemi kullanılır. Daha sonra bu değer bir aktivasyon fonksiyonuna girdi olarak girer ve bir sonuç verir. Aktivasyon fonksiyonu olarak sigmoid fonksiyonu oldukça yaygın olarak kullanılmaktadır. Bu sonuç sonraki katman için bir girdi değeri oluşturmaktadır. Böylece önceki katmanlardan gelen bilgi sonraki katmana aktarılmış olur.

Yapay sinir ağları katmanlardan oluşur. Bu katmanlar girdi katmanı, gizli katman ve çıktı katmanıdır. Girdi katmanı veri girişini sağlar. Veri girdi katmanından gizli katmanlara, gizli katmanlardan çıktı katmanına aktarılır. Çıktı değeri ise elde edilmek istenen değerdir. Eğer problem türü sınıflandırma ise çıktı katmanında sınıf sayısı kadar düğüm olur. Bu düğümlerin değerlerine göre olasılıklar hesaplanır ve en yüksek olasılığa sahip sınıf seçilir. Eğer regresyon ise tek düğümden oluşur ve o değer soncu ifade eder. Ayrıca eğer gizli katman birden fazla sayıdaysa derin öğrenme denilmektedir. Derin öğrenme bazen sonuçlara çok olumlu etkiler yapabilmektedir.

Yapay sinir ağlarını eğitmek demek ağırlık değerlerini optimize etmek demektir. Bunun için farklı yöntemler kullanılabilir. Matematiksel bir yöntem olan ve en çok kullanılan yöntem geri yayılım yöntemidir. Bu yöntemde hatalar geriye doğru yayılır ve ağırlık değerleri değiştirilir. Türev bazlı ilerleyen bir yöntemdir, bundan dolayı aktivasyon fonksiyonları genellikle türevlenebilir fonksiyonlardan seçilir.

Yapay sinir ağlarının sözde kodu Tablo 3.10'deki gibidir. Yapay sinir ağları ve geri yayılım algoritması 2. Bölüm'de daha detaylı olarak anlatılmıştır.

Tablo 3.10 Yapay sinir ağları algoritması sözde kodu

```
Veriyi oku
Uygun topolojiye göre yapay sinir ağı oluştur
İterasyon sayısı kadar:
        Yapay sinir ağını ağırlık değerleriyle ileriye doğru hesapla
        Çıktı katmanındaki ihtimalleri hesapla
        Doğru sınıf ile ihtimal arasındaki farkı hata olarak hesapla
        Geri yayılım algoritması ile bu hatayı geriye doğru dağıt
        Ağırlıkları güncelle
Son ağırlıklarla yapay sinir ağı üzerinden veriyi test et
```

Karar ağacı algoritması sınıflandırma ve regresyon problemlerinde yaygın olarak kullanılan ağaç tabanlı bir algoritmadır. Karar ağacının en üstteki düğümüne kök denir ve kökten aşağıya doğru yeni düğümler iner. Bu düğümlere dal denir. Dallardan aşağıya doğru inen son düğümlere ise yaprak denir ve bu yapraklardan sonuçlar elde edilir. Bu yapı ağaca benzediği için karar ağacı denmiştir.

Öncelikle kök düğümün ne olduğuna karar verilmesi gerekmektedir. Kök düğümü seçerken hangi nitelik olacağına karar vermek gerekmektedir. Bunun için gini veya entropi formülleri kullanılabilir. Bu formüllere göre en iyi değeri elde eden nitelik kök düğüm olarak seçilir.

Kök düğümün belirlenmesinden sonra veri kümesi alt dallara parçalanmış olur. Her parça için tekrar aynı hesaplamalar yapılır ve düğümler alta doğru bölünür. Bu işlem ağacın derinliğine kadar sürekli tekrar eder.

En son düğümler yaprak olarak geçer. Ağaç dalları yapraklarda son bulur. Bu yapraklardaki değerler sonucu ifade eder.

Karar ağacının bazı avantajları vardır. Hem sınıflandırma hem de regresyon problemleri için kullanılabilir. Genellikle iyi sonuçlar veren bir algoritmadır. Sonuçlar ağaç okunarak neden o sonuca varıldığı görülebilir. Açıktır, kara kutu değildir.

Karar ağacının dezavantajları ise eğer derinlik fazla olursa ağaç ezberlemeye gider, eğer yetersiz olursa çok genel ve basit kalır. Bu parametreleri dengeli ayarlamak gerekebilir. Karar ağacı yöntemi 2. Bölüm'de daha detaylı olarak anlatılmıştır.

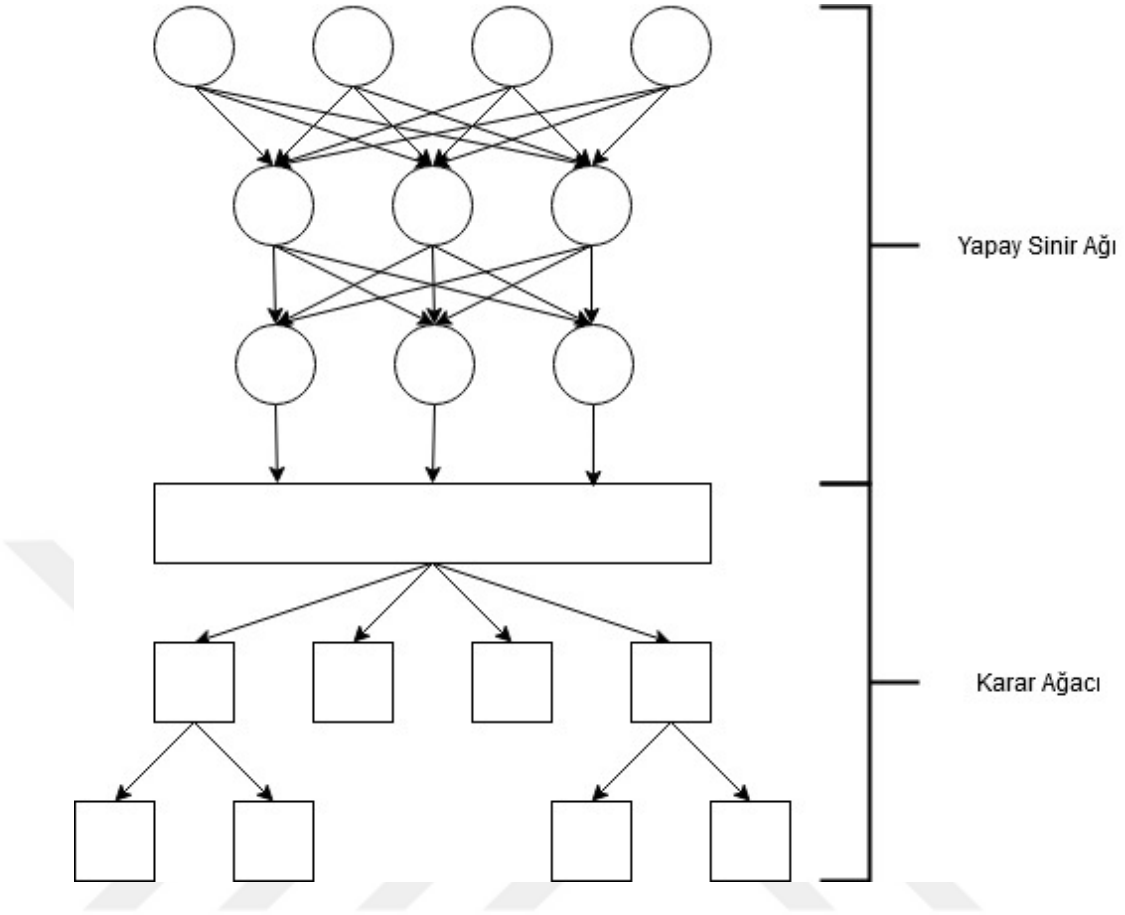
Karar ağacının sözde kodu Tablo 3.11'de görüldüğü gibidir.

Tablo 3.11 Karar ağacı algoritması sözde kodu

```
Veriyi oku
Karar ağacındaki derinlik değerine kadar:
        Verideki her bir özellik için:
                Özelliğin bilgi kazanımını hesapla
        En yüksek bilgi kazanımı sağlayan özelliği seç
        Bu özellik üzerinden ağaca dal oluştur
Karar ağacı oluştur
Karar ağacı üzerinden veriyi test et
```

YSA-KA algoritmasında öncelikle veri, eğitim ve test kümesi olarak ayrılır. Eğitim kümesi üzerinde veri yapay sinir ağına girdi katmanına gelir. Girdi katmanına gelen veriler girdi katmanı ile gizli katmanlar arasındaki ağırlıklar ile çarpılarak ve aktivasyon fonksiyonlarına uygulanarak çıktı katmanına kadar gelir ve çıktı katmanındaki her bir düğümde olasılıklara dağılır. Normal yapay sinir ağı yönteminde çıktı katmanındaki olasılıklardan en yüksek olanı seçilir. Fakat algoritmada bu aşamada herhangi bir seçim yapılmaz. Bu adım algoritmanın ilk aşamasıdır. Bu aşamada yapay sinir ağının hangi yapıda olacağı, kaç gizli katmana sahip olacağı, gizli katmanlarda kaç düğüm olacağı parametre olarak belirlenmiştir. Ayrıca yapay sinir ağının eğitiminde geri yayılım algoritması kullanılmıştır ve bu geri yayılım algoritmasındaki iterasyon sayısı da bir parametre olarak değerlendirilebilir. Buna ilaveten aktivasyon fonksiyonları da parametre olarak eklenebilir.

Yapay sinir ağından elde edilen çıktı olasılık değerleri, karar ağacına girdi olarak kullanılmaktadır. Bu aşamada karar ağacı elde edilen değerlere göre bir tahmin üretir. Önerilen YSA-KA algoritmasının temel yapısı Şekil 3.3’de gösterilmiştir.



Şekil 3.3 Yapay sinir ağı ve karar ağacı algoritması

Bu algoritmadaki parametreler ve açıklamaları Tablo 3.12'deki gibidir.

Tablo 3.12 YSA-KA algoritması parametreleri

Parametre	**Açıklama**
Gizli katman yapısı	Yapay sinir ağındaki gizli katman ve düğüm sayıları
İterasyon sayısı	Geri yayılım algoritmasının kaç kere ağırlık değerlerini değiştireceği
Aktivasyon fonksiyonu	Yapay sinir ağının aktivasyon fonksiyonu

Yapay sinir ağı-Karar ağacı algoritmasının sözde kodu Tablo 3.13'deki gibidir.

Tablo 3.13 YSA-KA algoritması sözde kodu

```
Veriyi oku
Veriyi karıştır
Veriyi eğitim ve test kümeleri olarak ayır
Parametrelere göre yapay sinir ağı oluştur
İterasyon sayısı kadar:
        Eğitim verisini yapay sinir ağındaki ağırlık değerleriyle hesapla
        Çıktı katmanındaki sınıf ihtimallerini hesapla
        Doğru sınıf ile ihtimal arasındaki farkı hata olarak hesapla
        Geri yayılım algoritması ile bu hatayı geriye doğru dağıt
        Ağırlıkları güncelle
En son yapay sinir ağını kaydet
Yapay sinir ağının çıktı katmanındaki sınıf ihtimallerini karar ağacı ile eğit
Tüm modeli kaydet
Kaydedilen son modeli test kümesi üzerinden test et ve doğruluk hesapla
```

4. BÖLÜM

DENEYSEL ÇALIŞMA VE ANALİZLER

Tez çalışması kapsamında geliştirilen Spark hibrit ve yapay sinir ağı-karar ağacı algoritmalarının performansını ortaya koyabilmek için deneysel çalışmalar ve analizler gerçekleştirilmiştir. Bu kapsamda literatürde mevcut 6 sınıflandırma veri kümesi ve 2 gerçek hayat veri kümesi kullanılmıştır. Gerçek hayat veri kümelerinin ilki bir üretim sınıflandırma problemi iken, ikinci gerçek hayat veri kümesi bir fonksiyon tahminleme veri kümesidir.

Çalışma kapsamında her bir veri kümesinde önerilen algoritmaların etkin parametre düzeylerini belirlemek için deney tasarımı gerçekleştirilmiş, belirlenen etkin düzeylerle her bir veri kümesinde algoritmalar Intel(R) Xeon(R) Silver 4110 CPU @2.10 GHz 2095 Mhz, 8 çekirdek, 32 GB RAM, Windows 10 işletim sistemi özelliklerine sahip bilgisayarda 30 kez çalıştırılarak ortalama değerler sunulmuştur.

Elde edilen sonuçların kalitesi klasik yöntemlerle ve son yıllarda literatürde yer alan çalışmalarla karşılaştırılmış, sonuçların istatistiksel anlamlılığı test edilmiştir. İlerleyen bölümlerde bu çalışmalar ile ilgili detaylar verilmiştir.

4.1. Literatür Verileri Üzerindeki Çalışmalar

4.1.1. Literatür Veri Kümeleri

Tez çalışması kapsamında önerilen makine öğrenme algoritmaları, literatürde mevcut 6 farklı sınıflandırma veri kümesi üzerinde test edilmiştir. Bu veriler Pokerhand, Susy, Higgs, Hepmass, Wisdm ve Census veri kümeleridir.

Pokerhand

Poker durumları ile ilgili bir veri kümesidir. 1.025.010 adet örnekten ve 10 adet nitelikten oluşmaktadır. Sınıf olarak ise 10 farklı çıktıya sahiptir. 10 özellik, her biri kartın türünü ve üzerindeki sayıyı tutan 5 farklı kart verisinden oluşmaktadır. Kartın özelliği 1 ile 4 arasındadır ve bu sayılar maça, kupa, karo ve sineği temsil etmektedir. Diğer özellik ise 1 ile 13 arasındadır ve kart üzerindeki sayıyı temsil etmektedir. Bu şekilde her kart için 2 özellik vardır. Toplam 10 özellik vardır. Bu 10 özellik sonucunda 10 farklı durum ortaya çıkmaktadır. Bu 10 farklı durum 10 sınıf olarak ifade edilmektedir [108]. Pokerhand veri kümesi özellikleri Tablo 4.1'de görülebilir.

Tablo 4.1 Pokerhand veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Kesikli nitelik sayısı	Sınıf sayısı
Pokerhand	1.025.010	5	5	10

Susy

Susy verileri Monte Carlo simülasyonları kullanılarak üretilmiştir. İlk 8 özellik, hızlandırıcıdaki parçacık detektörleri tarafından ölçülen kinematik özelliklerdir. Son 10 özellik, ilk 8 özelliğin fonksiyonlarıdır; bunlar, iki sınıf arasında ayrım yapmaya yardımcı olmak için fizikçiler tarafından türetilen üst düzey özelliklerdir. Veri kümesi 5.000.000 adet örneğe ve 18 niteliğe sahiptir. Amaç sinyal ve sinyal değil değerlerini alan 2 sınıfı tahmin etmektir. [109]. Susy veri kümesi özellikleri Tablo 4.2'de görülebilir.

Tablo 4.2 Susy veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Sınıf sayısı
Susy	5.000.000	18	2

Higgs

Veriler, Monte Carlo simülasyonları kullanılarak üretilmiştir. İlk 21 özellik, hızlandırıcıdaki parçacık detektörleri tarafından ölçülen kinematik özelliklerdir. Son yedi özellik, ilk 21 özelliğin fonksiyonlarıdır; bunlar, iki sınıf arasında ayrım yapmaya yardımcı olmak için fizikçiler tarafından türetilen üst düzey özelliklerdir. Veri kümesi

11.000.000 adet örneğe ve 28 niteliğe sahiptir. Higgs bozonu sınıflandırması olarak sinyal ve sinyal değil şekilinde 2 sınıfı mevcuttur. [110]. Higgs veri kümesi özellikleri Tablo 4.3'de görülebilir.

Tablo 4.3 Higgs veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Sınıf sayısı
Higgs	11.000.000	28	2

Hepmass

Yüksek enerjili fizik deneylerinden oluşturulan bir veri kümesidir. Higgs ve Susy veri kümeleri gibi Monte Carlo simülasyonuyla üretilmiştir. Toplam 7.000.000 örnekten oluşmaktadır ve bu örneklerin yarısı sinyal yarısı arka plan olmak üzere dengeli bir veri kümesidir. Her bir örnek toplam 28 niteliğe sahiptir. [111]. Hepmass veri kümesi özellikleri Tablo 4.4'de görülebilir.

Tablo 4.4 Hepmass veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Sınıf sayısı
Hepmass	10.500.000	28	2

Wisdm

Açılımı "wireless sensor data mining" olan Wisdm laboratuvarı tarafından toplanan veri kümesi, kişilerin hareketlerini tahmin etmek için toplanmıştır. Kişilerin hareketleri cep telefonlarıyla bir uygulama üzerinden toplanmıştır. Hareketler x, y ve z düzlemlerindeki değişiklikler üzerinden tahmin edilmeye çalışılmıştır. Tahmin edilmeye çalışılan hareketler:

- Yürüyüş
- Tempolu yürüyüş
- Merdiven çıkmak
- Oturmak
- Ayakta beklemek
- Yatmak

Bu veri kümesi birkaç versiyona sahiptir. İlk versiyon en basit halidir, zaman değerlerine göre tutulmuştur. Bundan dolayı tek başına kullanımı bir anlam ifade etmemektedir. İkinci versiyonu daha anlamlı bir türdür, ilk versiyonundan türetilmiştir. Diğer üçüncü versiyonu ise daha önceki çalışmalarda kullanılmış ve test edilmiştir. Bu versiyonu örnek sayısı olarak daha küçük olsa da nitelik sayısı olarak oldukça çoktur ve eski çalışmalarda kullanılan versiyonu budur. Sırasıyla ham veri, yarı işlenmiş veri ve işlenmiş veri olarak adlandırılabilir. Özellikleri Tablo 4.5'deki gibidir [112, 113].

Tablo 4.5 Wisdm veri kümesi versiyonları ve özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Sınıf sayısı
Ham veri	2.980.765	6	6
Yarı işlenmiş veri	1.369.349	46	6
İşlenmiş veri	5.435	563	6

Census

Census veri kümesi, insanları yıllık gelirlerine göre sınıflandıran bir veri kümesidir. İnsanların yaş, çalıştığı yer, eğitim seviyesi, evli olup olmadığı, cinsiyeti, etnik kökeni, hangi ülkeden geldiği gibi kişisel bilgileri üzerinden yıllık kazancının 50.000 dolar altı veya üstü olmasını bulmayı amaçlamaktadır. [114]. Veri kümesinin özellikleri Tablo 4.6'dan görülebilir.

Tablo 4.6 Census veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Kesikli nitelik sayısı	Sınıf sayısı
Census	48.842	6	8	2

4.1.2. Etkinlik Ölçütü ve Veri Kümelerinin Eğitim-Test Kümelerine Parçalanması

Algoritmalar karşılaştırılırken etkinlik ölçütü olarak sınıflandırma problemlerinde test doğruluğu kullanılmıştır. Doğruluk Eşitlik (4.1)'de gösterildiği gibi hesaplanmaktadır.

$$Doğruluk = \frac{Test\ kümesinde\ doğru\ sınıflandırılmış\ örnek\ sayısı}{Test\ kümesindeki\ toplam\ örnek\ sayısı} \quad (4.1)$$

Karşılaştırmalarda kullanılan literatür çalışmalarının çoğunda veri kümesinin eğitim ve test kümelerine parçalanmasında “basit bölümleme” kullanılmış ve veri kümelerinin %75’lik oranı eğitim kümesi, kalan %25’lik oranı ise test kümesi olarak ele alınmıştır. Tez çalışmasında sunulan ve literatürde mevcut algoritmalar ile karşılaştırmaların daha doğru ve adil olması için çalışmada veri kümeleri literatür ile aynı şekilde eğitim ve test kümelerine parçalanmıştır. Ancak Census ve Hepmass veri kümeleri orijinalinde eğitim ve test kümelerine ayrıldığı için bu veri kümelerine basit bölümleme gerçekleştirilmemiştir.

4.1.3. Deney Tasarımı

Tez kapsamında önerilen algoritmalar farklı parametrelere sahiptir ve bu parametrelerin aldığı değerlere göre çözüm etkinlikleri değişmektedir. Bu parametrelerden etkin olanlarını ve parametrelerin en uygun değerlerini belirlemek amacıyla bu bölümde bir deney tasarımı gerçekleştirilmiştir. Önerilen algoritmalar az sayıda parametreye sahip olduğu için, tam faktöriyel deney tasarımları Minitab 19 istatistiksel analiz paket programı kullanılarak düzenlenmiştir. Deneyler on kez tekrarlı şekilde, her bir veri kümesi için ayrı ayrı yapılmıştır.

Tam faktöriyel tasarımında sınıflandırma doğruluk oranı üzerinde faktörlerin etkileri analiz edilirken, varyans analizi sonuçları, ana etkiler ve ikili etkileşimler dikkate alınmıştır. Parametre sayısının fazla olması durumunda oldukça kullanışlı olan ana etki grafikleri, düzey ortalamalarını karşılaştırmaya ve etkinlik ölçütünü en çok etkileyen parametrelerin belirlenmesine olanak sağlamaktadır. Ana etki grafiğinde elde edilen doğrunun x-eksenine göre konumu parametrenin etkinlik düzeyi hakkında bilgi vermektedir. Eğer doğru, x-eksenine paralel ise bu durum parametrenin ele alınan her düzeyinin ölçütü aynı yönde etkilediği ve ana etkinin olmadığı anlamına gelmektedir. Eğer elde edilen doğru belirli bir eğime sahipse, bu durumda bir ana etki söz konusudur ve ele alınan parametrenin farklı düzeyleri, etkinlik ölçütünü farklı şekilde etkilemektedir. Doğrudaki eğimi, etkinin oranını göstermektedir, eğim arttıkça, parametrenin ölçüt üzerindeki etkisi de artmaktadır. Ana etki grafiklerinde, doğruların eğimleri karşılaştırılarak, parametrelerin göreceli önemi belirlenebilmektedir.

Ana etki grafikleri, parametrelerin etkinlikleri hakkında net bir sonuca varmak için yeterli değildir, etkileşim grafiklerine de bakmak gerekir. Çünkü etkileşim eğrileri, ana etkiyi doğrulayabilecekleri gibi ortadan da kaldırabilirler. Bu nedenle, ana etki grafiklerinin yanı sıra etkileşim grafiklerinin de incelenmesi gerekir. Bir etkileşim grafiği, bir faktörün düzeylerindeki değişimin diğer bir faktör düzeylerindeki etkisini gösterir. Eğer etkileşim grafiklerinde ele alınan parametrelerin doğruları paralele yakınsa, parametreler birbirinden bağımsızdır ve parametreler arasında çok az veya hiç etkileşim yoktur denilebilir. Etkileşim grafiklerinde birbirini kesen doğrular, parametreler arasındaki güçlü etkileşimleri ifade ederler, iyi yorumlanmalıdırlar.

Deney tasarımı sonucu varyans analizi p-değerleri, ana etki ve ikili etkileşim grafiklerinden faydalanılarak sınıflandırma doğruluk oranı üzerinde etkisi olan algoritma parametreleri ve bu parametrelerin etkin düzeyleri belirlenmiştir. Tez çalışması kapsamında önerilen iki algoritma, her bir veri kümesi için algoritma parametrelerinin belirlenen düzeylerinde 10'ar kez çalıştırılmış ve etkinlik ölçütleri hesaplanmıştır. Deneysel tasarımda analizlerin her bir veri kümesi için ayrı ayrı ele alınmasının nedeni, veri kümesindeki örnek ve değişken sayılarının değişmesi ile faktörlerin etkinliğinin değişebilmesidir.

Spark Hibrit Algoritması Deney Tasarımı

Spark Hibrit algoritması, parça sayısı ve oylama türü olmak üzere iki parametreye sahiptir. Parça sayısı parametresi, eğitim kümesinin kaç parçaya bölüneceğini ifade etmektedir. Parça sayısı; yapılan ön testler sonucunda 10, 20 ve 30 olmak üzere üç düzey olarak ele alınmıştır. Oylama türü parametresi ise, eğitim kümesindeki veri parçalarını eğiten algoritmaların oy hakkı değeridir ve deney tasarımında iki düzey olarak ele alınmıştır. İlk düzey, her algoritmanın 1 değerinde oya sahip olması, ikinci düzey ise doğruluk değeri oranında oya sahip olmasını ifade etmektedir. Deney tasarımı parametreleri ve parametre düzeyleri Tablo 4.7'de verilmiştir.

Tablo 4.7 SH algoritması parametreleri ve parametre düzeyleri

Parametre	**Değerler**
Parça sayısı	10 – 20 – 30
Oylama türü	1 – doğruluk

Deney tasarımında 3x2=6 deney bulunmaktadır. Her bir veri kümesi ve tasarımın her bir deneyi için algoritma 10 kez çalıştırılarak bu 10 çalıştırmanın ortalama sınıflandırma doğruluk oranı deney tasarımında etkinlik ölçütü olarak ele alınmıştır.

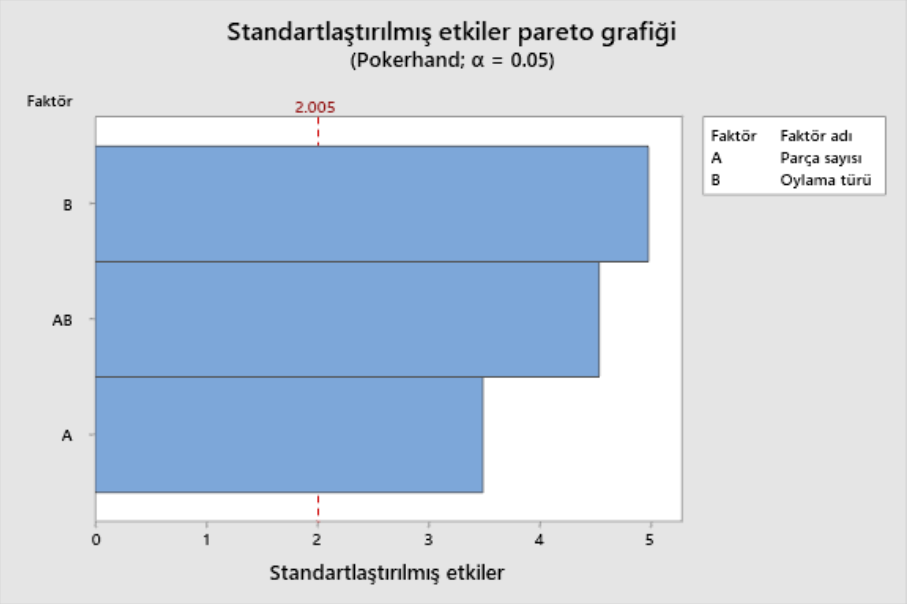
Pokerhand Veri Kümesi Deney Tasarımı

Pokerhand veri kümesi için varyans analizi sonuçları Tablo 4.8'de verilmiştir. Anlamlılık düzeyi α=0.05 olarak dikkate alındığında parça sayısı, oylama türü parametreleri ve parça sayısı*oylama türü etkileşimlerinin etkili olduğu görülmektedir (p değerleri <0.05).

Tablo 4.8 Pokerhand veri kümesi varyans analizi

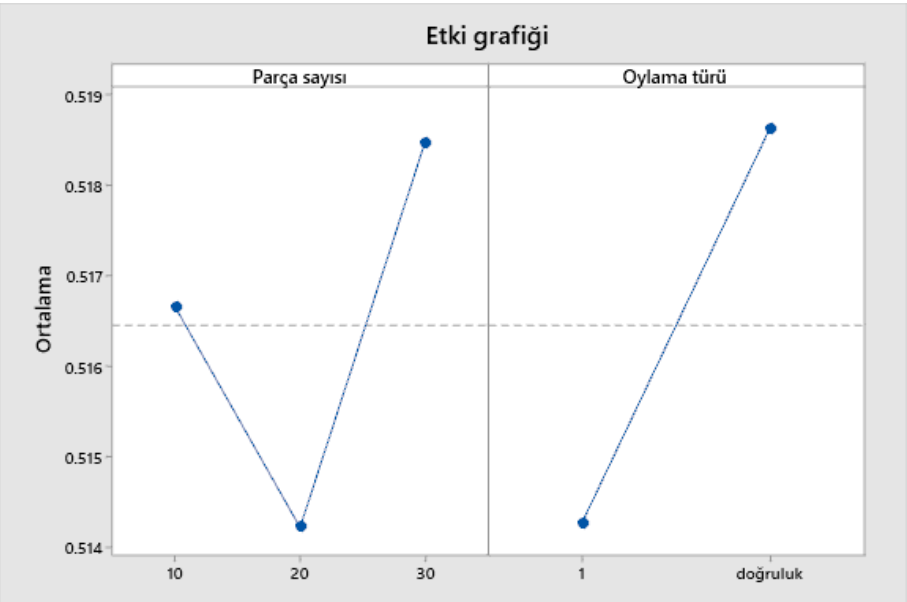
Kaynak	**Serbestlik Derecesi**	**Ayarlanmış SS**	**Ayarlanmış MS**	**F-Değeri**	**P-Değeri**
Model	5	0.000761	0.000152	13.17	0.000
Lineer	3	0.000470	0.000157	13.54	0.000
Parça sayısı	2	0.000183	0.000091	7.91	**0.001**
Oylama türü	1	0.000287	0.000287	24.80	**0.000**
2-li Etkileşimler	2	0.000292	0.000146	12.61	0.000
Parça sayısı*Oylama türü	2	0.000292	0.000146	12.61	**0.000**
Hata	54	0.000625	0.000012		
Toplam	59	0.001386			

Pokerhand veri kümesi için Şekil 4.1'de verilen standartlaştırılmış pareto grafiği incelendiğinde parça sayısı, oylama türü ve parça sayısı-oylama türü ikili etkileşiminin etkili olduğu belirlenmiştir. Bunlar arasında en etkili olanı ise parça sayısıdır.



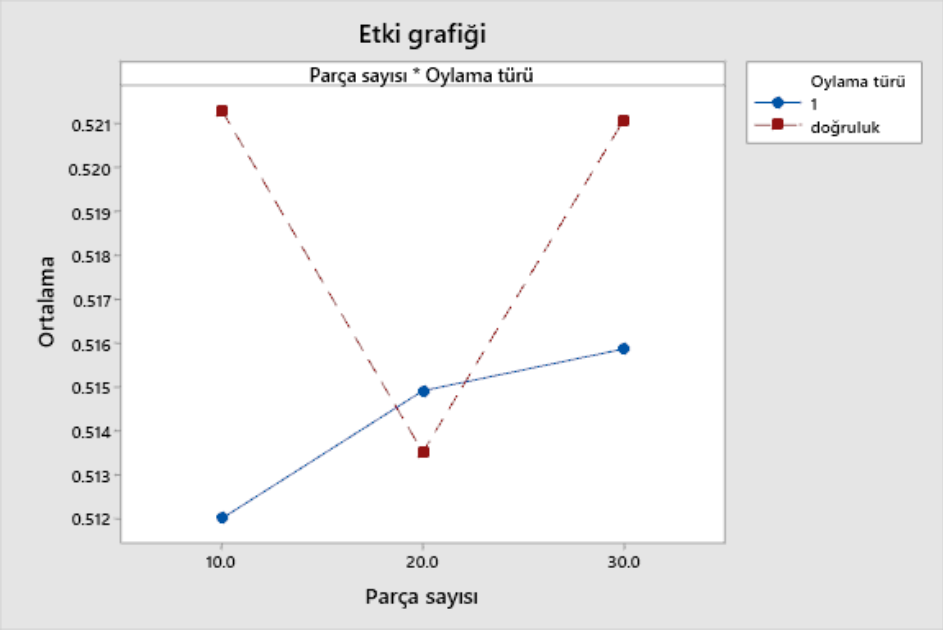
Şekil 4.1 Pokerhand veri kümesi pareto grafiği

Şekil 4.2'te Pokerhand veri kümesi için ana etkiler grafiği verilmiştir. Şekilden de görüldüğü üzere parça sayısı ve oylama türü parametrelerinin her ikisinin de doğruları belirli bir eğime sahiptir dolayısıyla her ikisi de etkili parametrelerdir. Ana etki grafiğine göre sınıflandırma doğruluğu etkinlik ölçütünde parça sayısı parametresinde etkin düzey 30, oylama türü parametresinde ise etkin düzey doğruluk olarak elde edilmiştir.



Şekil 4.2 Pokerhand veri kümesi ana etkiler grafiği

Şekil 4.3'te Pokerhand veri kümesi için ikili etkileşim grafiği verilmiştir. Şekilden de görüldüğü üzere etkileşim doğruları birbirini kesmektedir, dolayısıyla parça sayısı ve oylama türü parametrelerinin farklı düzeylerinin birbirlerini etkilediği söylenebilir. İkili etkileşim grafiğinden sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkin parametre değerlerinin parça sayısı için 10 (ikili etkileşimler, ana etki değerini değiştirmiştir) ve oylama türü parametresi için doğruluk olduğu görülmektedir.



Şekil 4.3 Pokerhand veri kümesi ikili etkileşim grafiği

Sonuç olarak Pokerhand veri kümesi için etkin parametre değerleri:

- Parça sayısı: 10
- Oylama türü: doğruluk

olarak belirlenmiştir.

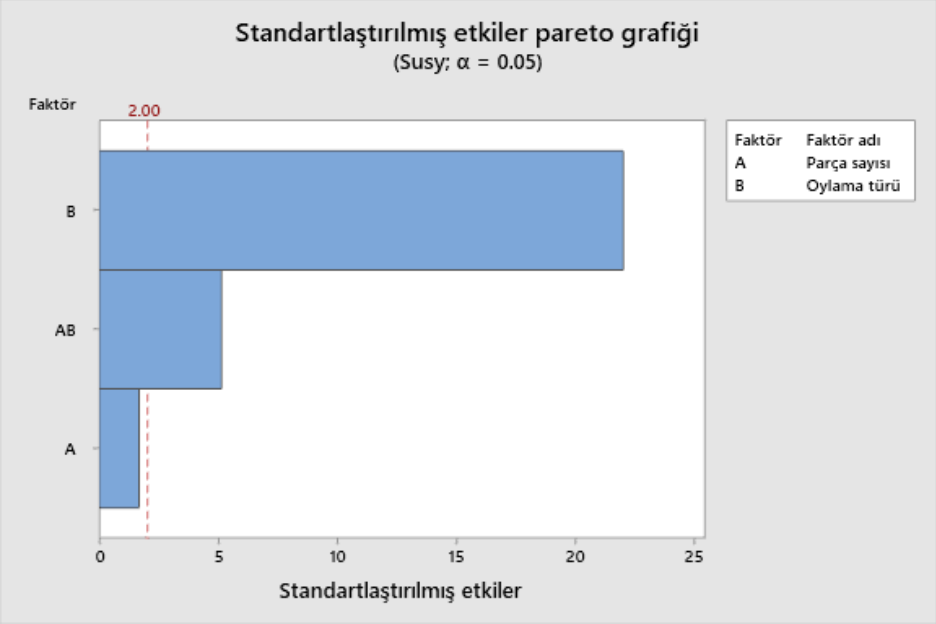
Susy Veri Kümesi Deney Tasarımı

Tablo 4.9'da sunulan Susy veri kümesi için varyans analizi sonuçları incelendiğinde, oylama türü parametresi ve parça sayısı*oylama türü ikili etkileşiminin etkili olduğu görülmektedir (p değeri<0.05). Ancak, parça sayısı parametresinin p değeri 0.10 ile α=0.05 değerinden yüksek çıkmıştır. Dolayısıyla parça sayısı parametresi sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili bir parametre değildir.

Tablo 4.9 Susy veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.000122	0.000024	104.51	0.000
Lineer	3	0.000115	0.000038	163.60	0.000
Parça sayısı	2	0.000001	0.000001	2.40	0.100
Oylama türü	1	0.000114	0.000114	486.01	**0.000**
2-li Etkileşimler	2	0.000007	0.000004	15.86	0.000
Parça sayısı*Oylama türü	2	0.000007	0.000004	15.86	**0.000**
Hata	54	0.000013	0.000000		
Toplam	59	0.000135			

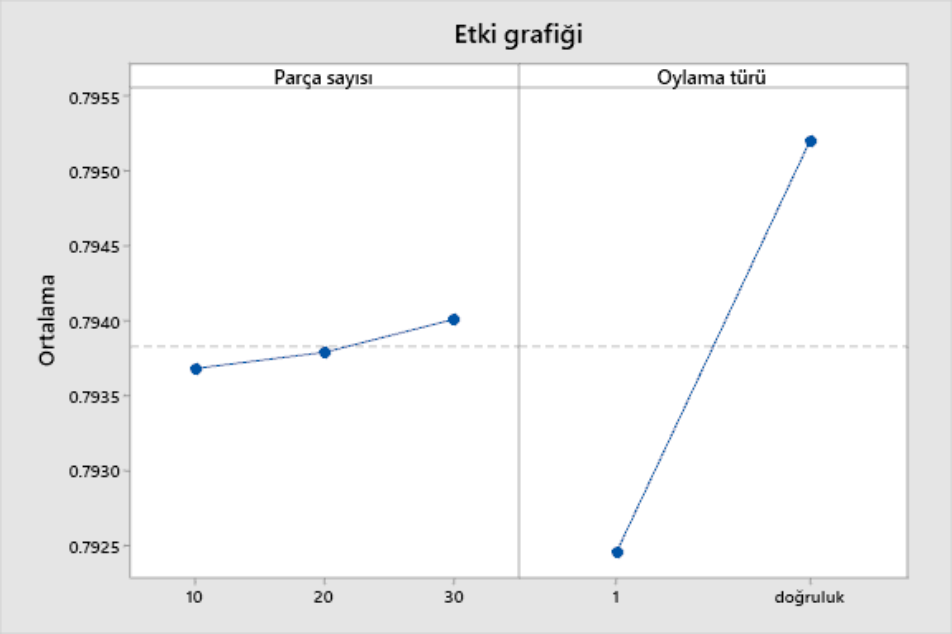
Şekil 4.4'da gösterilen standartlaştırılmış etkiler pareto grafiği varyans analizi sonuçlarını doğrulamaktadır. Oylama türü parametresi ve parça sayısı-oylama türü ikili etkileşimi sınıflandırma doğruluğu üzerinde etkili çıkmıştır. Parça sayısı parametresi ise etkili bir parametre değildir.



Şekil 4.4 Susy veri kümesi standartlaştırılmış etkiler pareto grafiği

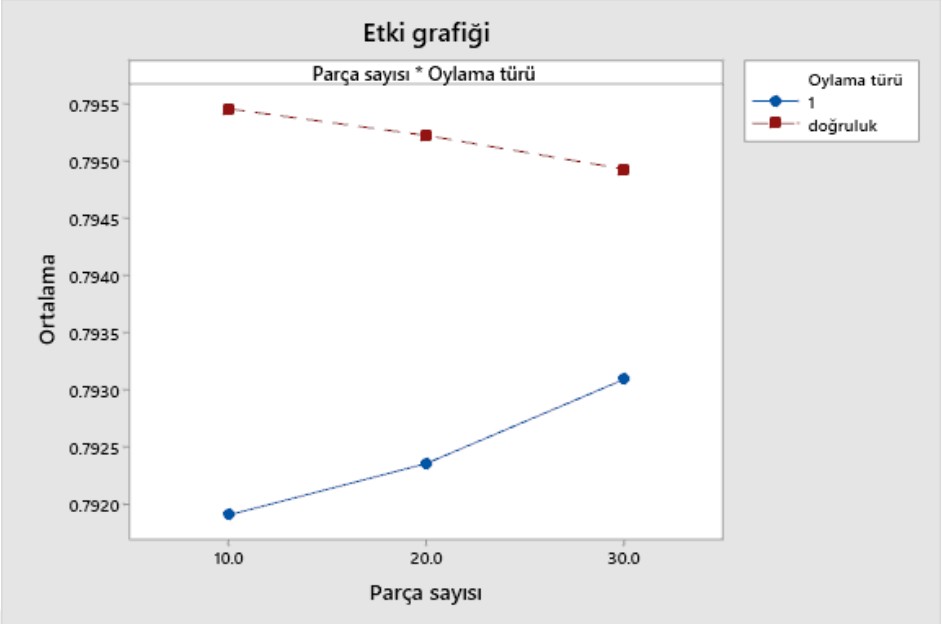
Şekil 4.5'da Susy veri kümesi için ana etki grafikleri verilmiştir. Ana etki grafikleri incelendiğinde oylama türü parametresine ait doğrunun oldukça yüksek bir eğime sahip

olduğu dolayısıyla aldığı farklı parametre değerlerinin sınıflandırma doğruluğu üzerinde oldukça etkili olduğu ve en yüksek sınıflandırma doğruluğunu doğruluk düzeyinin verdiği görülmektedir. Parça sayısı parametresine ait etki doğrusunun ise eğimi oldukça azdır, dolayısıyla sınıflandırma doğruluğu üzerinde parametrenin düzeylerinin etkisinin de az olduğu söylenebilir. Parça sayısı parametresi için ana etki grafiğinden etkili parametre değerinin 30 olduğu görülmektedir.



Şekil 4.5 Susy veri kümesi ana etkiler grafiği

Şekil 4.6'da Susy veri kümesi için ikili etkileşim grafiği sunulmuştur. İkili etkileşim grafiği incelendiğinde doğruların birbirini kesmediği ancak birbirlerine paralel de olmadığı görülmektedir. İkili etkileşim grafiğine bakıldığında parça sayısı parametresinin 10, oylama türü parametresinin doğruluk düzeylerinin etkili düzeyler olduğu görülmektedir.



Şekil 4.6 Susy veri kümesi ikili etkileşim grafiği

Sonuç olarak Susy veri kümesi için etkin parametre değerleri:

- Parça sayısı: 10
- Oylama türü: doğruluk olarak belirlenmiştir.

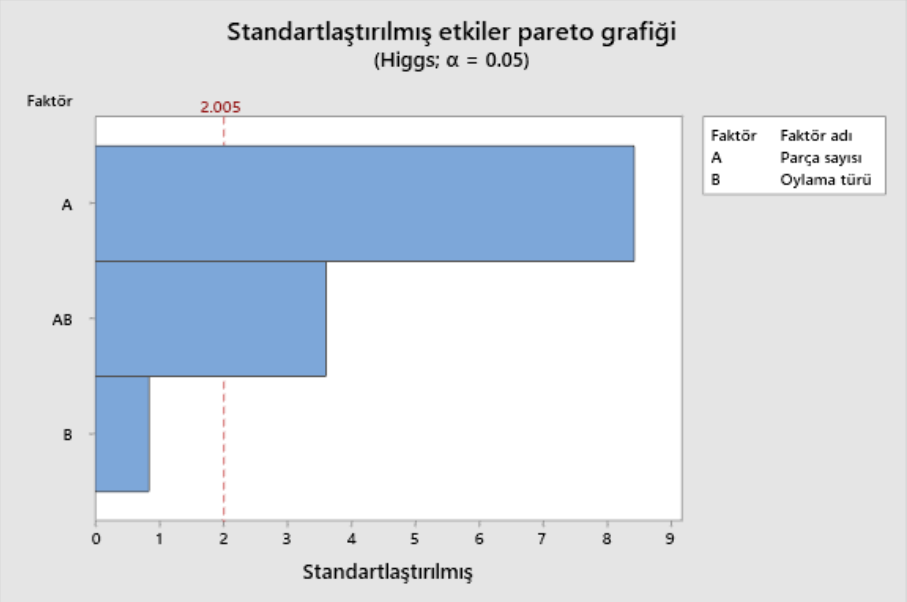
Higgs Veri Kümesi Deney Tasarımı

Tablo 4.10'te Higgs veri kümesi için varyans analizi sonuçları verilmiştir. Varyans analizi sonuçlarında p değerleri değerlendirildiğinde parça sayısı parametresinin ve parça sayısı*oylama türü ikili etkileşiminin sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili oldukları görülmektedir (p değerleri<0.05). Oylama türü parametresinin p değeri α=0.05'den yüksek çıktığı için etkinlik ölçütü üzerinde etkili bir parametre değildir.

Tablo 4.10 Higgs veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.000259	0.000052	75.74	0.000
Lineer	3	0.000247	0.000082	120.65	0.000
Parça sayısı	2	0.000247	0.000123	180.61	**0.000**
Oylama türü	1	0.000000	0.000000	0.71	0.402
2-li Etkileşimler	2	0.000011	0.000006	8.39	0.001
Parça sayısı*Oylama türü	2	0.000011	0.000006	8.39	**0.001**
Hata	54	0.000037	0.000001		
Toplam	59	0.000296			

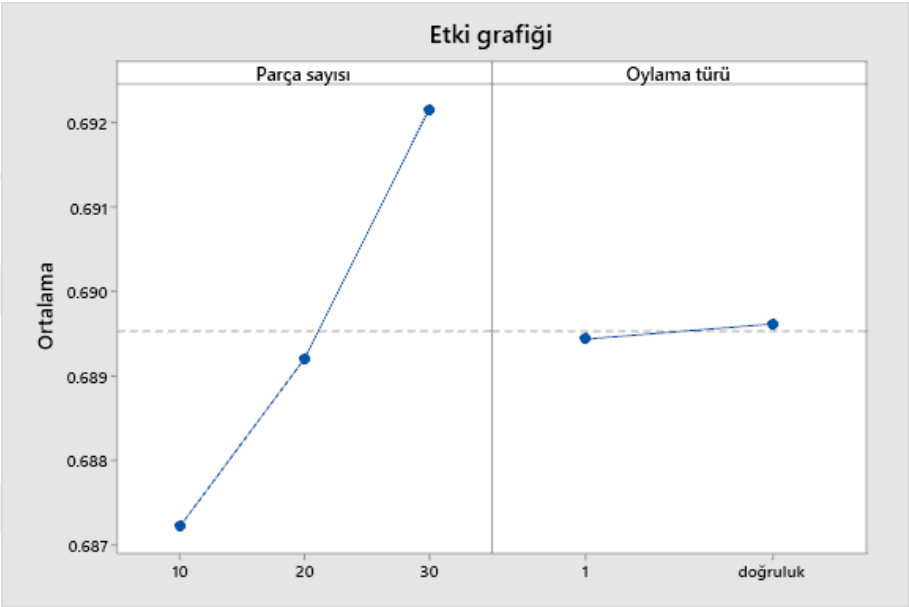
Şekil 4.7'de verilen standartlaştırılmış etkiler pareto grafiği de varyans analizi sonuçlarını doğrulamaktadır. Pareto grafiğine göre de parça sayısı parametresi ve parça sayısı-oylama türü ikili etkileşimi etkili parametrelerdir. Oylama türü parametresi ise etkili bir parametre değildir.



Şekil 4.7 Higgs veri kümesi standartlaştırılmış etkiler pareto grafiği

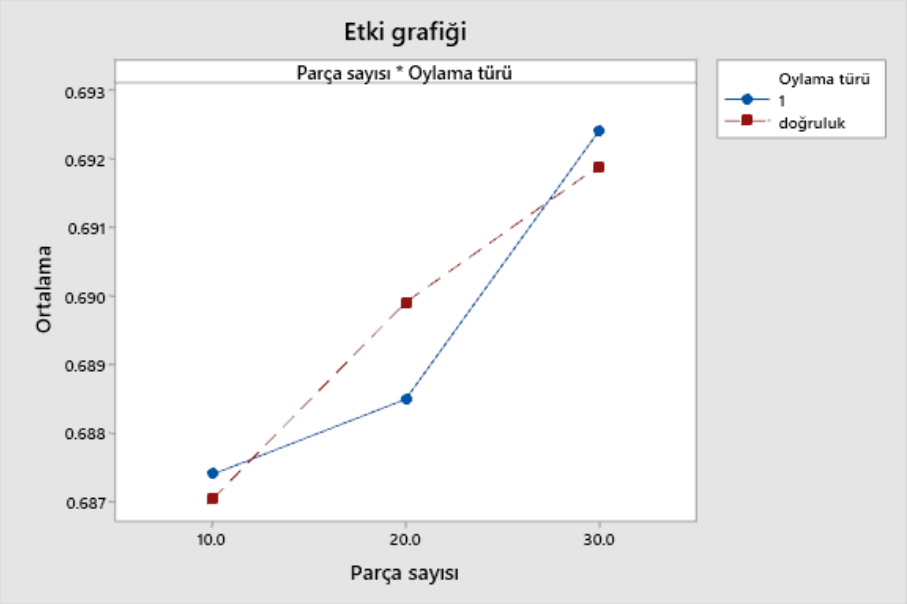
Susy veri kümesinde parça sayısı ve oylama türü parametrelerinin etkin düzeylerini ortaya koymak için Şekil 4.8'te ana etkiler grafiği sunulmuştur. Ana etkiler grafiği

incelendiğinde parça sayısı parametresi oldukça yüksek bir eğime sahiptir ve sınıflandırma doğruluk etkinlik ölçütü üzerinde etkili değeri 30'dur. Oylama türü parametresinin ise ana etki doğrusu neredeyse doğrusaldır, dolayısıyla etkinlik ölçütü üzerinde parametrenin değerlerinin etkisinin çok fazla olduğu söylenemez. Ancak hafif bir eğimle de olsa oylama türü parametresinde doğruluk düzeyinin sınıflandırma doğruluğu üzerinde daha etkin olduğu söylenebilir.



Şekil 4.8 Higgs veri kümesi ana etkiler grafiği

Ana etkiler ile elde edilen sonuçların geçerliliğini test etmek için ikili etkileşimler Şekil 4.9'te gösterilmiştir. İkili etkileşim grafikleri incelendiğinde doğruların birbirini kestiği dolayısıyla parça sayısı ve oylama türü parametrelerinin aldığı düzey değerlerinin birbirini etkilediği söylenebilir. İkili etkileşim grafiğine göre, parça sayısı parametresi için 30 ve oylama türü parametresi için 1 düzeyleri etkin parametre değerleridir.



Şekil 4.9 Higgs veri kümesi ikili etkileşim grafiği

Higgs veri kümesi için etkin parametre değerleri

- Parça sayısı: 30
- Oylama türü: 1 olarak belirlenmiştir.

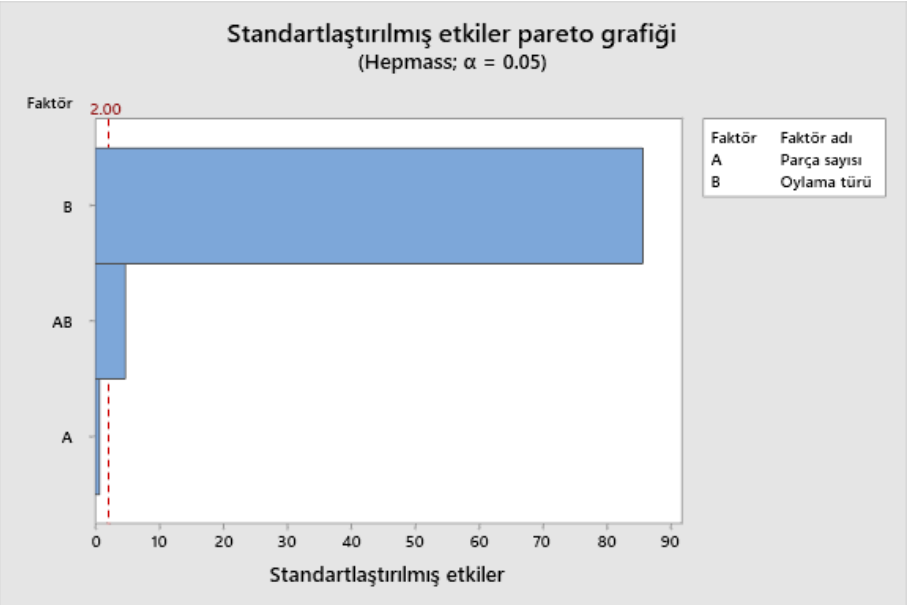
Hepmass Veri Kümesi Deney Tasarımı

Tablo 4.11 Hepmass veri kümesi için varyans analizi sonuçlarını göstermektedir. Varyans analizi sonuçlarından p değerleri incelendiğinde %95 anlamlılık seviyesinde oylama türü parametresi ve parça sayısı*oylama türü ikili etkileşimleri p değerleri α=0.05 değerinden küçük çıktığı için etkili olarak belirlenmiştir. Parça sayısı parametresinin p değeri 0.05 değerinden yüksek çıktığı için sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili bir parametre değildir.

Tablo 4.11 Hepmass veri kümesi varyans analizi

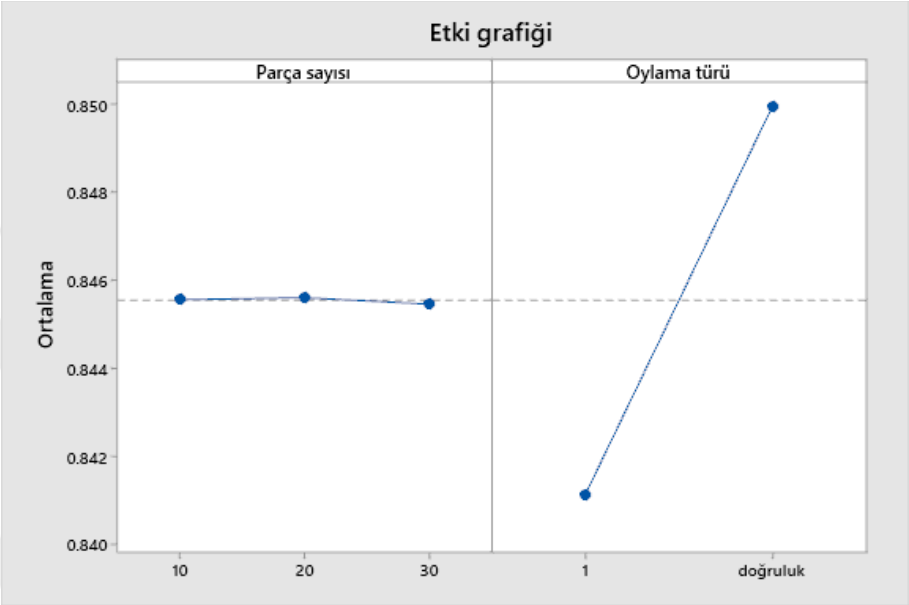
Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.001177	0.000235	1474.23	0.000
Lineer	3	0.001172	0.000391	2447.86	0.000
Parça sayısı	2	0.000000	0.000000	0.73	0.486
Oylama türü	1	0.001172	0.001172	7342.12	**0.000**
2-li Etkileşimler	2	0.000004	0.000002	13.78	0.000
Parça sayısı*Oylama türü	2	0.000004	0.000002	13.78	**0.000**
Hata	54	0.000009	0.000000		
Toplam	59	0.001186			

Varyans analizi sonuçlarını Şekil 4.10'da verilen standartlaştırılmış etkiler pareto grafiği desteklemektedir. Parato grafiğine göre oylama türü parametresi oldukça yüksek etkili bir parametredir. Parça sayısı-oylama türü ikili etkileşimleri de oylama türü parametresi kadar olmasa da etkilidir. Parça sayısı parametresi ise sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili değildir.



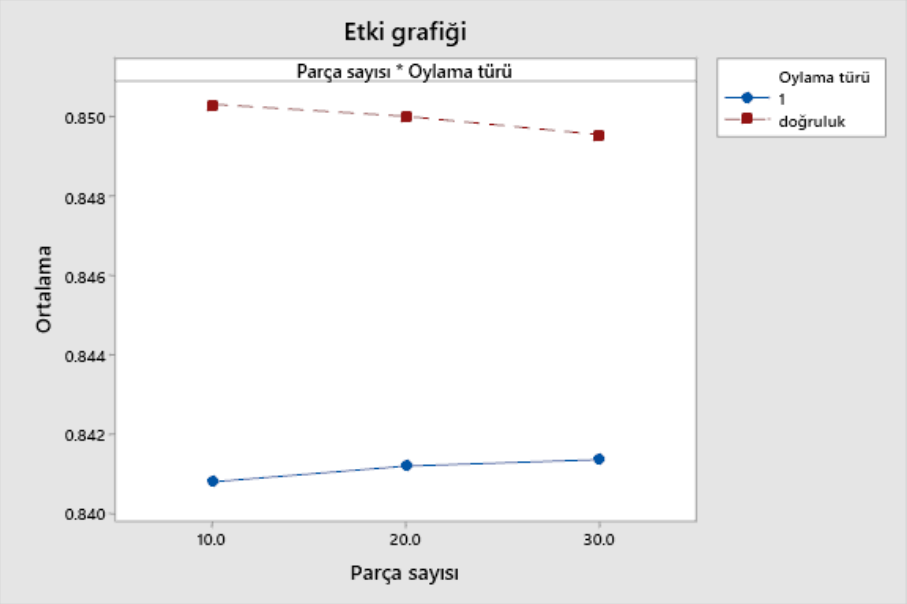
Şekil 4.10 Hepmass veri kümesi standartlaştırılmış etkiler pareto grafiği

Şekil 4.11'da sunulan ana etkiler grafiğine göre oylama türü parametresi oldukça yüksek eğimli bir doğruya sahip olduğu için etkinlik ölçütü üzerinde etkili bir parametredir ve sınıflandırma doğruluk oranı için etkinlik düzeyi doğruluk değeridir. Parça sayısı parametresinin doğrusu ise neredeyse yatay eksene paralel çıkmıştır, dolayısıyla sınıflandırma doğruluk oranı üzerinde etkisi hemen hemen hiç yoktur.



Şekil 4.11 Hepmass veri kümesi ana etkiler grafiği

Parametrelerin düzeylerinin birbirleri üzerindeki etkisini analiz etmek için ikili etkileşimler incelenmiştir. Şekil 4.12'de gösterilen ikili etkileşimler incelendiğinde en etkin parametre düzeylerinin parça sayısı parametresi için 10, oylama türü parametresinin ise doğruluk olduğu görülmektedir.



Şekil 4.12 Hepmass veri kümesi ikili etkileşim grafiği

Sonuç olarak Hepmass veri kümesi için etkin parametre düzeyleri

- Parça sayısı: 10
- Oylama türü: doğruluk olarak belirlenmiştir.

Wisdm Veri Kümesi Deney Tasarımı

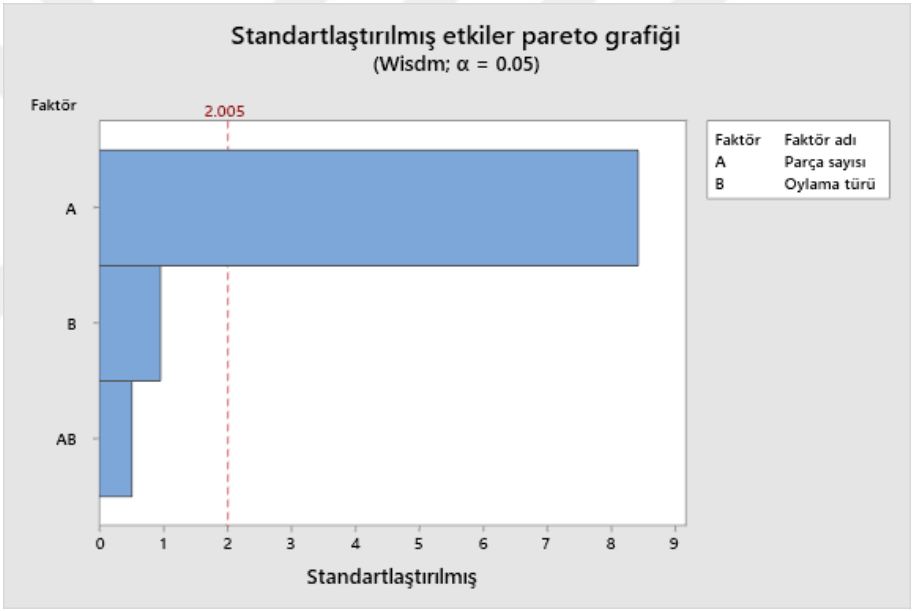
Tablo 4.12'da Wisdm veri kümesi için varyans analizi sonuçları verilmiştir. Tablodaki p değerleri incelendiğinde % 95 anlamlılık düzeyinde sadece parça sayısı parametresinin sınıflandırma doğruluğu etkinlik ölçütü üzerinde etkili olduğu ($p<0.05$), oylama türü parametresinin ve parça sayısı-oylama türü ikili etkilemişinin etkili olmadığı ($p>0.05$) görülmektedir.

Tablo 4.12 Wisdm veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.003194	0.000639	19.60	0.000
Lineer	3	0.003161	0.001054	32.34	0.000
Parça sayısı	2	0.003131	0.001566	48.05	**0.000**

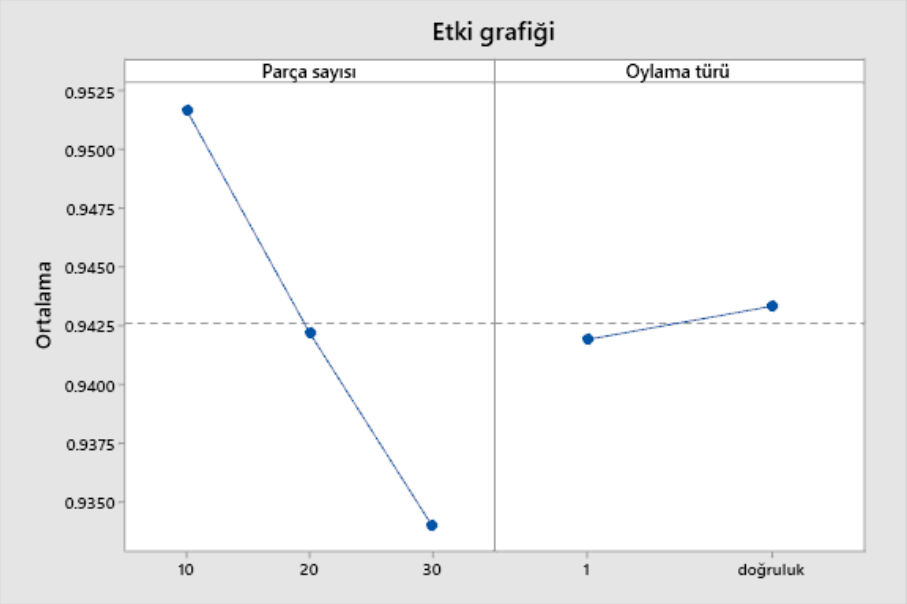
Oylama türü	1	0.000030	0.000030	0.93	0.339
2-li Etkileşimler	2	0.000032	0.000016	0.50	0.612
Parça sayısı*Oylama türü	2	0.000032	0.000016	0.50	0.612
Hata	54	0.001760	0.000033		
Toplam	59	0.004953			

Şekil 4.13'de Wisdm veri kümesi için standartlaştırılmış etkiler pareto grafiği verilmiştir. Grafikten görüldüğü gibi tek etkili parametre parça sayısıdır ve parça sayısı ile oylama türü parametreleri arasında da ikili etkileşim söz konusu değildir.



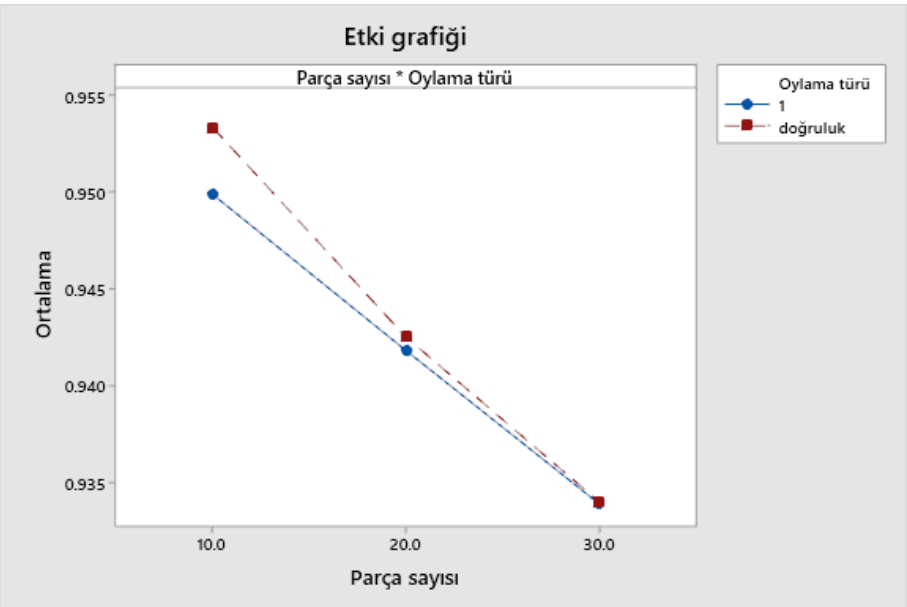
Şekil 4.13 Wisdm veri kümesi standartlaştırılmış etki pareto grafiği

Wisdm veri kümesi için düzenlenen deney tasarımı sonucu elde edilen ana etkiler grafiği Şekil 4.14'te sunulmuştur. Ana etkiler grafiği incelendiğinde parça sayısının sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde oldukça etkili bir parametre ve etkin düzeyinin 10 olduğu, oylama türü parametresinin etkinlik ölçütü üzerinde çok düşük bir etkisinin olduğu ve etkin düzeyinin doğruluk olduğu görülmektedir.



Şekil 4.14 Wisdm veri kümesi ana etkiler grafiği

Şekil 4.15'te gösterilen ikili etkileşim grafiği sonuçları ana etkiler grafiği sonuçlarını desteklemektedir. Bunun sebebi, ikili etkileşim doğrularının birbirini kesmemesi yani ikili etkileşimin söz konusu olmamasıdır. İkili etkileşim grafiğine göre de parça sayısı parametresinin etkin düzeyi 10, oylama türü parametresinin etkin düzeyi doğruluk olarak elde edilmiştir.



Şekil 4.15 Wisdm veri kümesi ikili etkileşim grafiği

Wisdm veri kümesi için etkin parametre değerleri

- Parça sayısı: 10
- Oylama türü: doğruluk olarak belirlenmiştir.

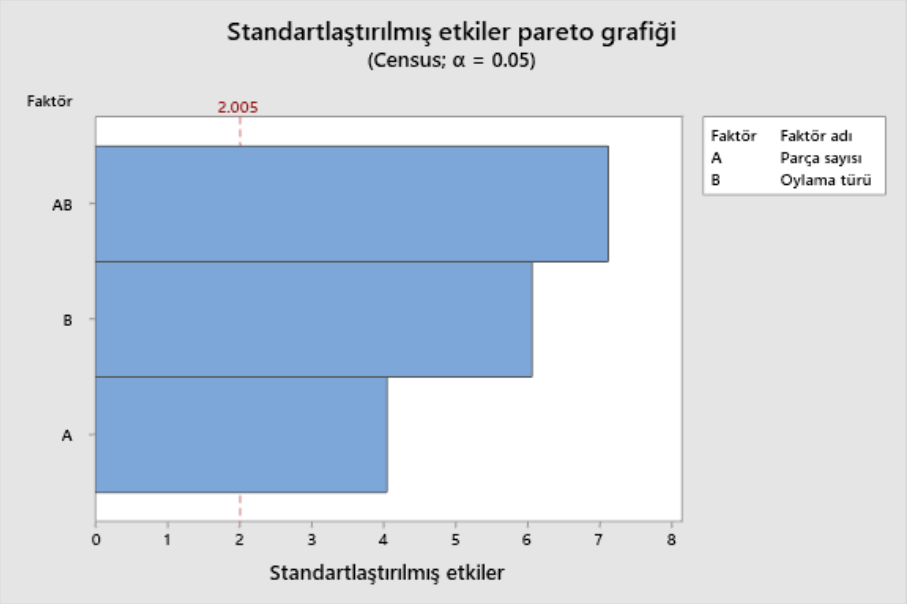
Census Veri Kümesi Deney Tasarımı

Tablo 4.13'de Census veri kümesi için varyans analizi sonuçları gösterilmektedir. Tablodan da görüldüğü üzere varyans analizinde bütün parametreler için p değeri 0.05'in altında çıkmıştır. Bütün parametreler etkilidir.

Tablo 4.13 Census veri kümesi varyans analizi

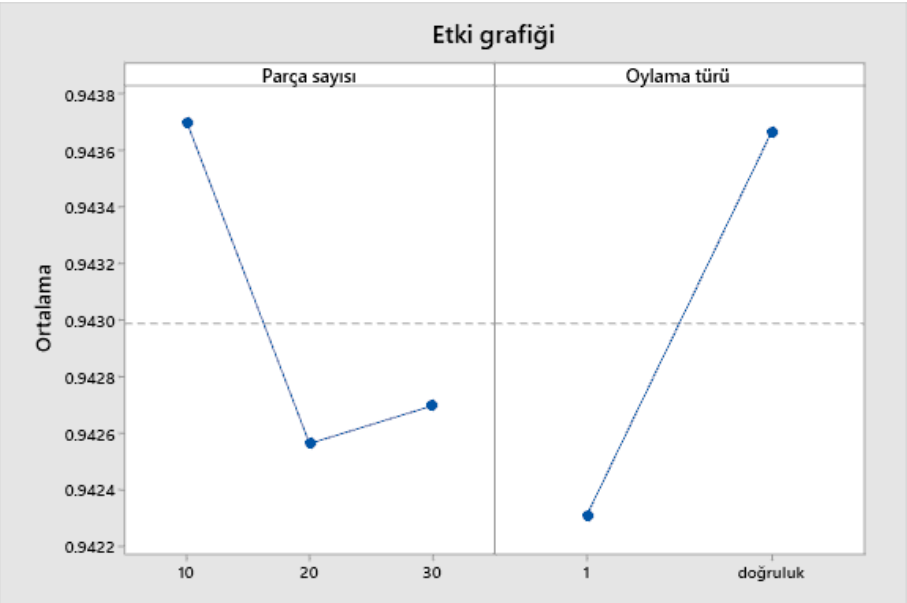
Kaynak	**Serbestlik Derecesi**	**Ayarlanmış SS**	**Ayarlanmış MS**	**F-Değeri**	**P-Değeri**
Model	5	0.000087	0.000017	23.15	0.000
Lineer	3	0.000043	0.000014	19.14	0.000
Parça sayısı	2	0.000016	0.000008	10.30	**0.000**
Oylama türü	1	0.000028	0.000028	36.82	**0.000**
2-li Etkileşimler	2	0.000044	0.000022	29.16	0.000
Parça sayısı*Oylama türü	2	0.000044	0.000022	29.16	**0.000**
Hata	54	0.000041	0.000001		
Toplam	59	0.000128			

Şekil 4.16'da sunulan standartlaştırılmış etkiler pareto grafiğine göre de parça sayısı, oylama türü parametreleri ve parça sayısı-oylama türü ikili etkileşimleri etkilidir.



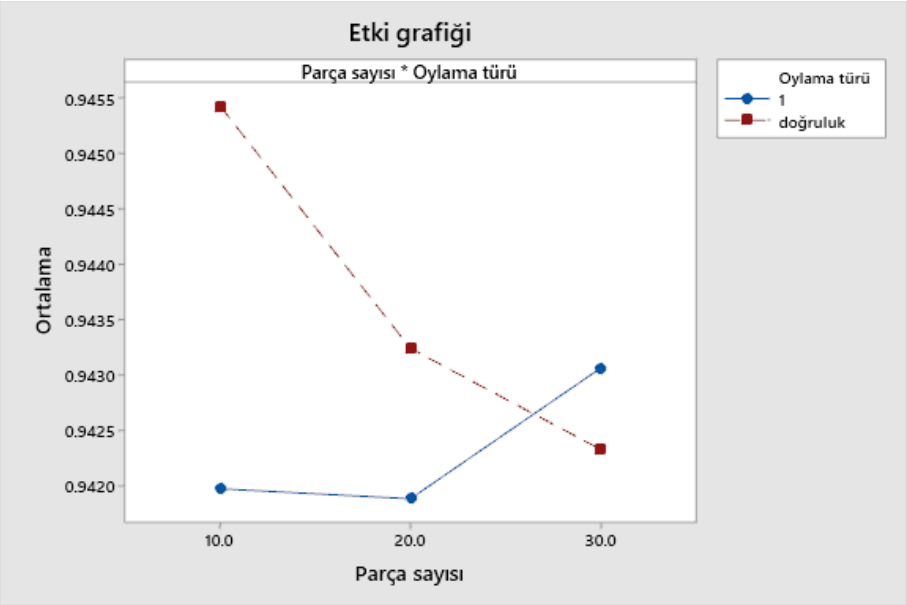
Şekil 4.16 Census veri kümesi standartlaştırılmış etki pareto grafiği

Census veri kümesinde algoritma parametrelerinin etkin düzeylerine karar verebilmek için Şekil 4.17'da sunulan ana etkiler grafiğinden faydalanılmıştır. Ana etkiler grafiğine göre her iki parametrenin etki doğrusu da eğime sahiptir ve etkilidir; sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde parça sayısı parametresinin etkin düzeyi 10, oylama türü parametresinin etkin düzeyi ise doğruluktur.



Şekil 4.17 Census veri kümesi ana etkiler grafiği

Şekil 4.18'da sunulan ikili etkileşim grafiği incelendiğinde, doğrular birbirini kestiği için parça sayısı ve oylama türü parametreleri arasında etkileşim olduğu görülmektedir. İkili etkileşimler dikkate alındığında sınıflandırma doğruluğu etkinlik ölçütü üzerinde parça sayısı parametresinin etkin düzeyi 10 ve oylama türü parametresinin etkin düzeyi doğruluktur.



Şekil 4.18 Census veri kümesi ikili etkileşim grafiği

Census veri kümesi için etkin parametre düzeyleri:

- Parça sayısı: 10
- Oylama türü: doğruluk olarak belirlenmiştir.

Genel olarak bakıldığında Spark Hibrit algoritması için en iyi parametre değerleri parça sayısı 10 ve oylama türü doğruluk çıkmaktadır. Sadece Higgs veri kümesinde bunun dışında bir parametre seçimi çıkmıştır.

YSA-Karar Ağacı Algoritması Deney Tasarımı

YSA-Karar Ağacı algoritması, gizli katman ve iterasyon olmak üzere iki parametreye sahiptir. Gizli katman sayısı parametresi yapay sinir ağında kaç gizli katman ve gizli katmanlarda kaç/kaçar proses elemanı (nöron) olduğunu ifade eden bir parametredir. G, GO, G-G/2 ve GO-GO/2 olmak üzere 4 düzeyi deney tasarımında ele alınmıştır. Bu düzeylerden G değeri, yapay sinir ağının tek gizli katman ve gizli katmanda ele alınan

problemin girdi niteliği kadar gizli katman proses elemanına sahip olmasını; GO değeri, YSA'nın tek gizli katman ve gizli katmanda ele alınan problemin girdi sayısı ile sınıf (çıktı) sayısının geometrik ortalaması ($\sqrt{girdi\ sayısı * sınıf\ sayısı}$) kadar gizli katman proses elemanına sahip olmasını; G-G/2 değeri, yapay sinir ağının iki gizli katman ve ilk gizli katmanda ele alınan problemdeki girdi niteliği kadar, ikinci gizli katmanda girdi niteliğinin yarısı kadar proses elemanı içermesini; GO-GO/2 değeri ise YSA'nın iki gizli katman, ilk gizli katmanda ele alınan problemdeki girdi ve sınıf sayılarının geometrik ortalaması, ikinci gizli katmanda girdi sayısı ve sınıf sayısının geometrik ortalamasının yarısı kadar proses elemanına sahip olmasını ifade etmektedir. İterasyon sayısı ise yapay sinir ağının kaç iterasyon çalışacağını gösteren parametredir ve deney tasarımında 100 ve 250 olmak üzere 2 düzeyi ele alınmıştır. Deney tasarımı parametreleri ve parametre düzeyleri Tablo 4.14'de verilmiştir.

Tablo 4.14 YSA-KA algoritması deney tasarımı parametreleri

Parametre	**Değerler**			
Gizli katman	G	GO	G-G/2	GO-GO/2
İterasyon	100	250		

Deney tasarımında 4x2=8 deney bulunmaktadır. Her bir veri kümesi ve tasarımın her bir deneyi için algoritma 10 kez çalıştırılarak bu 10 çalıştırmanın ortalama sınıflandırma doğruluk oranı deney tasarımında etkinlik ölçütü olarak ele alınmıştır.

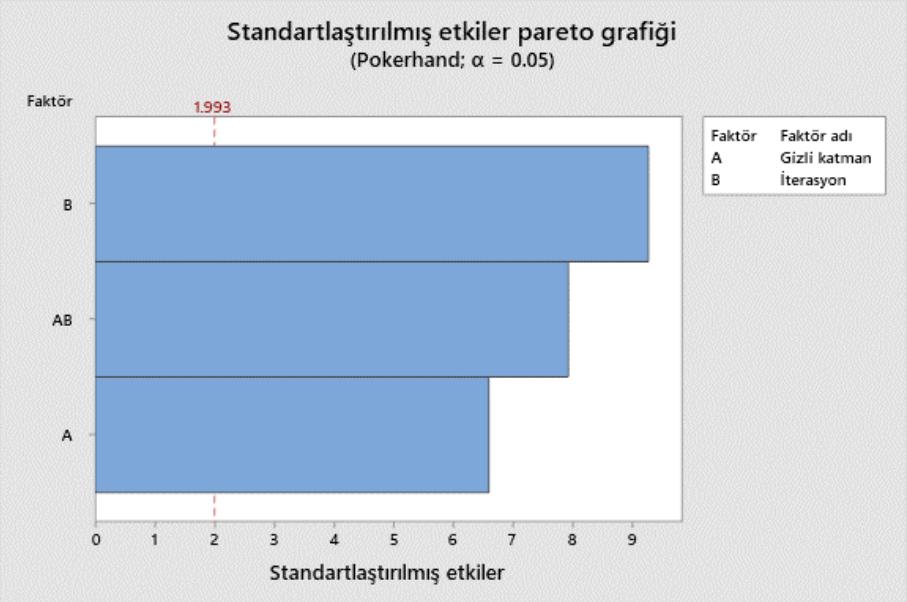
Pokerhand Veri Kümesi Deney Tasarımı

Pokerhand veri kümesi için deney tasarımında varyans analizi sonuçları Tablo 4.15'de gösterilmiştir. %95 anlamlılık düzeyinde p değerleri α=0.05'den küçük çıktığı için gizli katman ve iterasyon parametreleri de gizli katman*iterasyon ikili etkileşimi de sınıflandırma doğruluğu etkinlik ölçütü üzerinde etkili çıkmıştır.

Tablo 4.15 Pokerhand veri kümesi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	7	0.027223	0.003889	33.73	0.000
Lineer	4	0.016273	0.004068	35.28	0.000
Gizli katman	3	0.006352	0.002117	18.36	**0.000**
İterasyon	1	0.009921	0.009921	86.04	**0.000**
İkili etkileşimler	3	0.010950	0.003650	31.65	0.000
Gizli katman*İterasyon	3	0.010950	0.003650	31.65	**0.000**
Hata	72	0.008302	0.000115		
Total	79	0.035525			

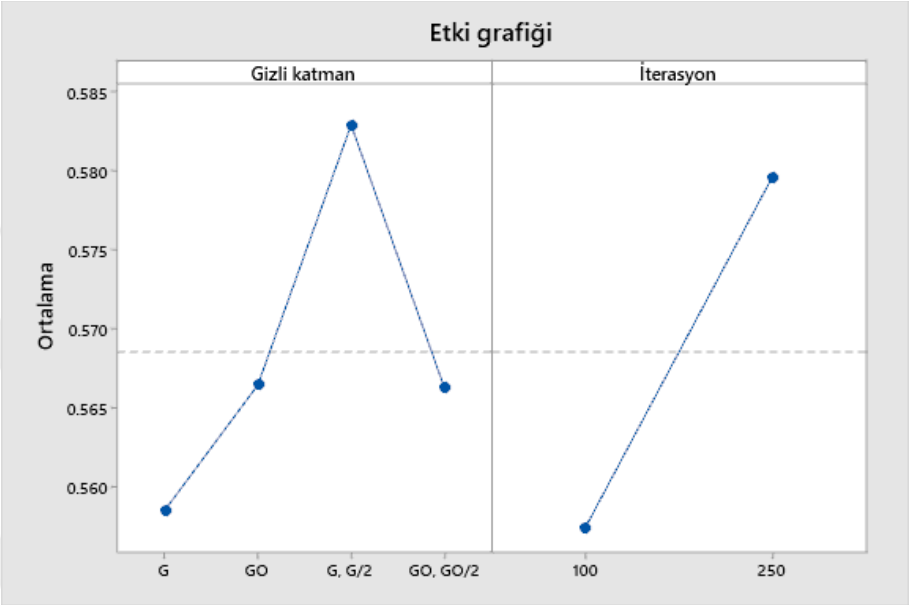
Şekil 4.19'de Pokerhand veri kümesi için standartlaştırılmış etkiler pareto grafiği verilmiştir. Şekilden de görüldüğü gibi varyans analizi ile aynı şekilde iterasyon ve gizli katman parametreleri ve iterasyon-gizli katman ikili etkileşimi etkinlik ölçütü üzerinde etkili çıkmıştır. Bunlar arasında en etkili olan faktör iterasyon parametresidir.



Şekil 4.19 Pokerhand veri kümesi standartlaştırılmış etkiler pareto grafiği

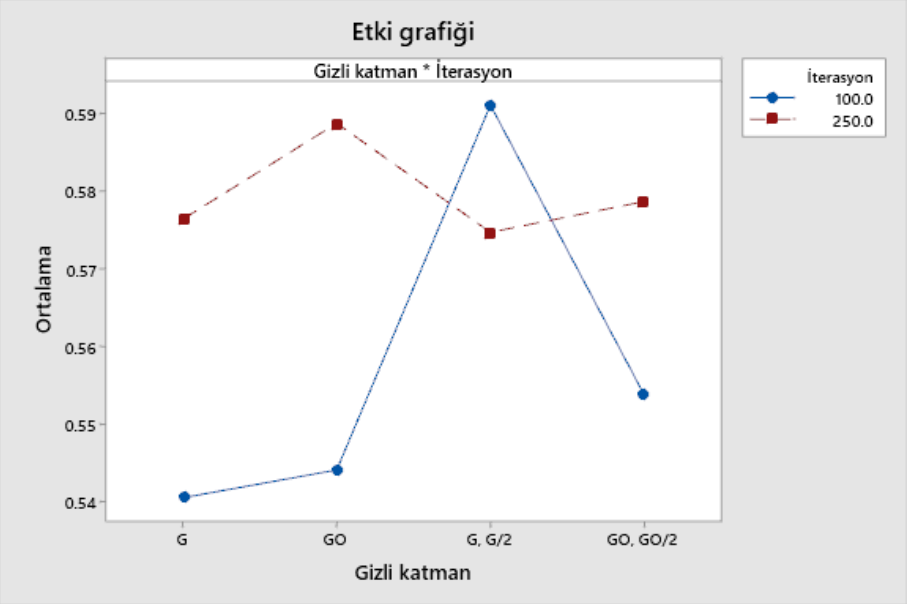
Deney tasarımı kapsamında parametrelerin etkin düzeylerini belirleyebilmek için Şekil 4.20'de verilen ana etkiler grafiği ve Şekil 4'de verilen ikili etkileşim grafiği değerlendirilmiştir. Ana etki grafiği incelendiğinde gizli katman parametresinin de iterasyon parametresinin de etki doğruları eğime sahiptir, dolayısıyla etkili

parametrelerdir. Parametrelerin sınıflandırma doğruluk oranı etkinlik ölçütü üzerindeki etkin değerleri ise gizli katman için G-G/2 yani iki gizli katmanlı ve birinci gizli katmanda veri kümesinin girdi niteliği kadar proses elemanı, ikinci katmanında ise girdi nitelik sayısının yarısı kadar proses elemanı olan YSA yapısı ve iterasyon parametresi için 250'dir.



Şekil 4.20 Pokerhand veri kümesi ana etkiler grafiği

Şekil 4.21'de sunulan ikili etkileşim grafiği incelendiğinde etkileşim doğrularının birbirini kestiğini yani iterasyon parametresi ile gizli katman parametresi arasında ikili etkileşim olduğu ve bir parametredeki düzey değişiminin diğer parametrenin değerini de etkilediği görülmektedir. Etkileşimler dikkate alındığında iterasyon parametresi için ikili etkileşimler ana etkiyi ortadan kaldırmaktadır, dolayısıyla iterasyon parametre değeri 100'dür. Gizli katman parametresi için ise ikili etkileşim ana etkiyi doğrulamakta yani etkin düzey G-G/2 olarak belirlenmektedir.



Şekil 4.21 Pokerhand veri kümesi ikili etkileşim grafiği

Sonuç olarak Pokerhand veri kümesi için etkin parametre değerleri:

- Gizli katman: G-G/2
- İterasyon: 100 olarak belirlenmiştir.

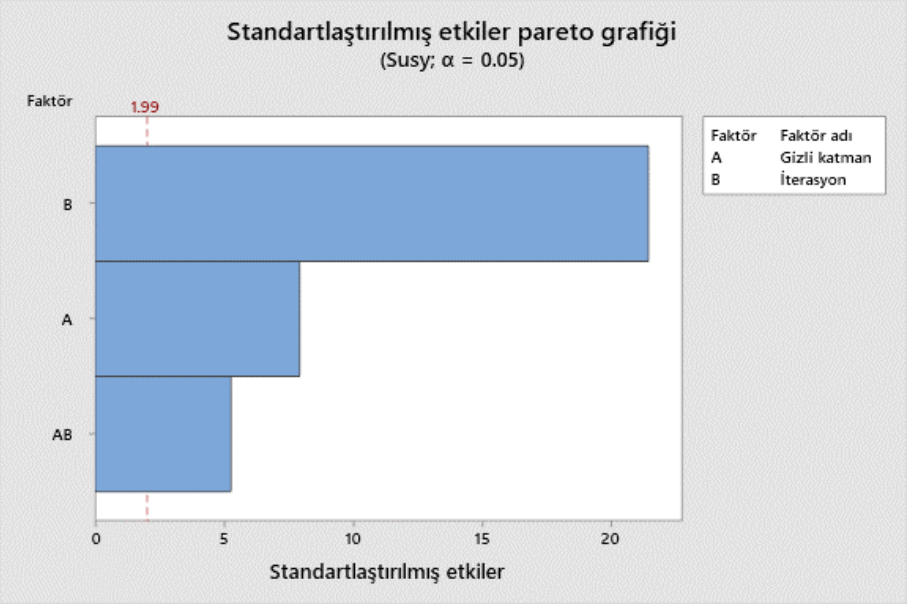
Susy Veri Kümesi Deney Tasarımı

Susy veri kümesi için düzenlenen deney tasarımı neticesinde elde edilen varyans analizi tablosu Tablo 4.16'de gösterilmiştir. Tabloda verilen p değerleri incelendiğinde gizli katman ve iterasyon parametreleri, aynı zamanda gizli katman*iterasyon ikili etkileşimi sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkilidir, parametrelerin ve ikili etkileşimin p değerleri α=0.05 değerinden düşük çıkmıştır.

Tablo 4.16 Susy veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	7	0.000098	0.000014	95.99	0.000
Lineer	4	0.000092	0.000023	158.72	0.000
Gizli katman	3	0.000025	0.000008	58.17	**0.000**
İterasyon	1	0.000067	0.000067	460.38	**0.000**
İkili etkileşimler	3	0.000005	0.000002	12.36	0.000
Gizli katman*İterasyon	3	0.000005	0.000002	12.36	**0.000**
Hata	72	0.000010	0.000000		
Total	79	0.000108			

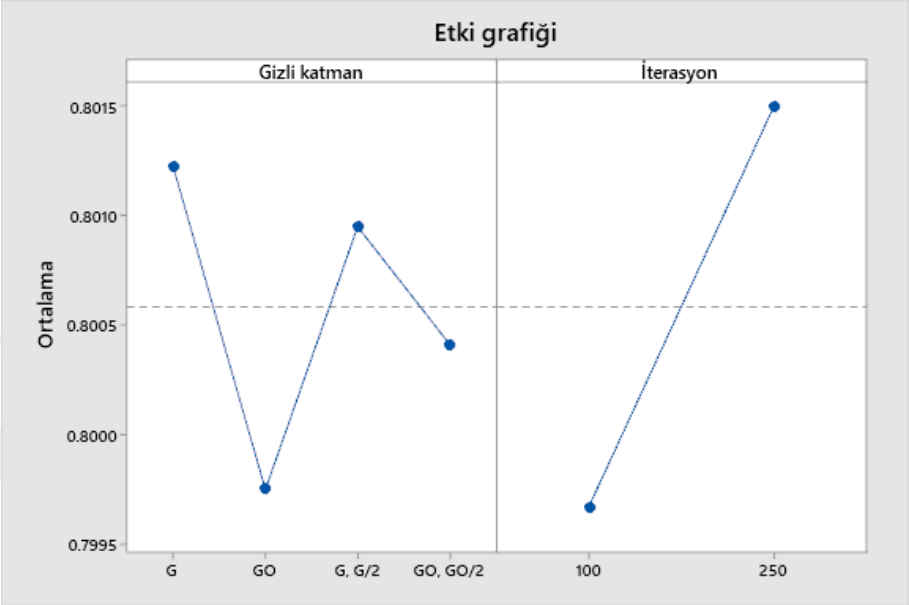
Şekil 4.22'de Susy veri kümesi için standartlaştırılmış etkiler pareto grafiği gösterilmiştir. Pareto grafiğinden de görüldüğü gibi parametreler ve ikili etkileşimleri etkilidir. Etkinlik düzeylerine göre en etkili parametre iterasyon, sonra gizli katman ve en son bu iki parametrenin ikili etkileşimidir.



Şekil 4.22 Susy veri kümesi standartlaştırılmış etkiler pareto grafiği

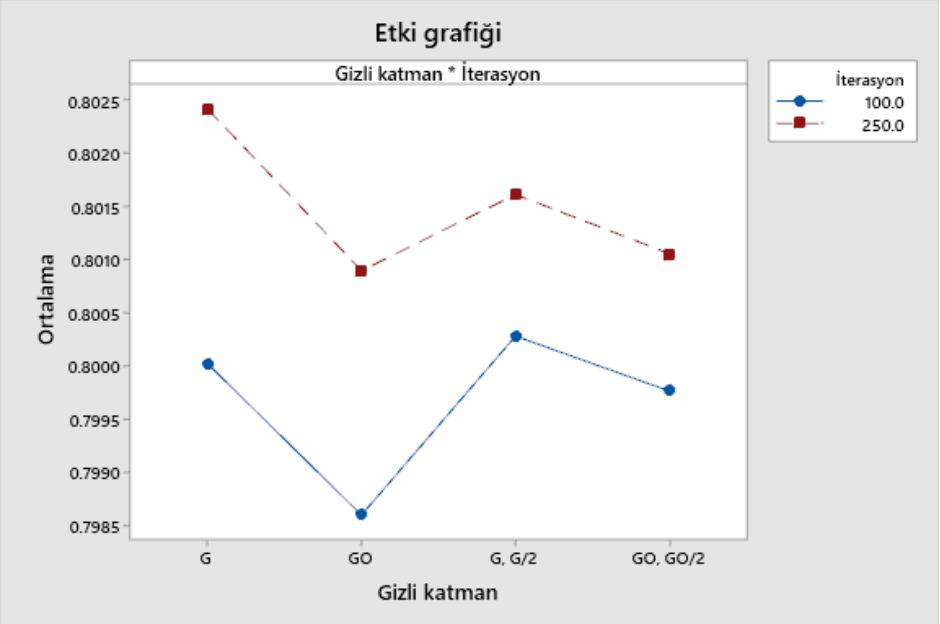
Susy veri kümesinde iterasyon ve gizli katman parametrelerinin etkin düzeylerini belirleyebilmek için Şekil 4.23'de gösterilen ana etki grafikleri incelenmiştir. Grafikten

de görüldüğü gibi gizli katman parametresinin etkin düzeyi G yani tek gizli katman ve gizli katmanda veri kümesinin girdi niteliği kadar proses elemanlı YSA yapısı, iterasyon parametresinin etkin düzeyi ise 250'dir.



Şekil 4.23 Susy veri kümesi ana etkiler grafiği

Şekil 4.24'de Susy veri kümesi için elde edilen ikili etkileşim grafiği gösterilmektedir. İkili etkileşimler incelendiğinde, ikili etkileşimlerin ana etkileri doğruladığı yani gizli katman parametresinin etkin düzeyinin G, iterasyon parametresinin etkin düzeyinin ise 250 olduğu görülmektedir.



Şekil 4.24 Susy veri kümesi ikili etkileşim grafiği

Sonuç olarak Susy veri kümesi için etkin parametre değerleri:

- Gizli katman: G
- İterasyon: 250 olarak belirlenmiştir.

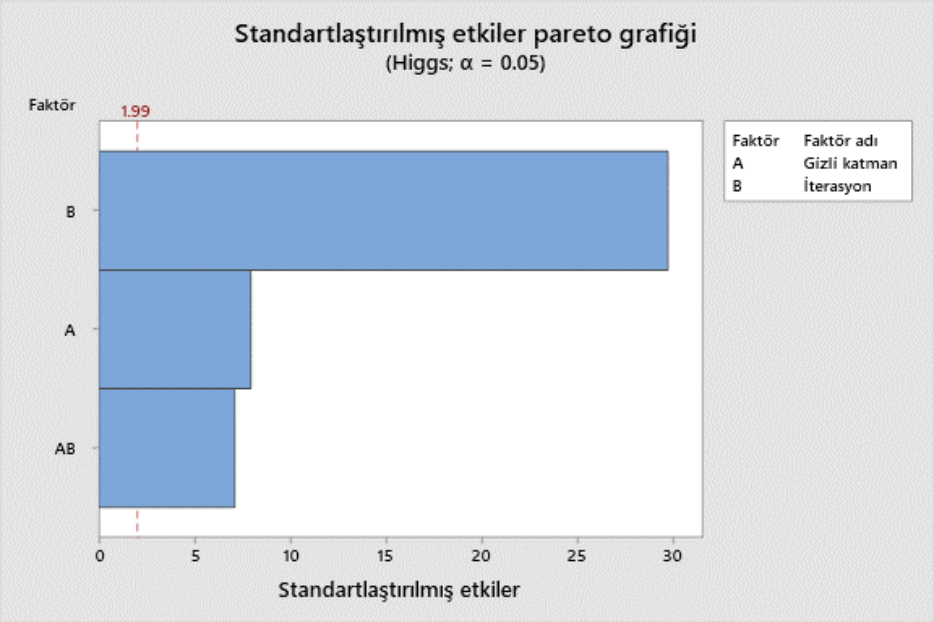
Higgs Veri Kümesi Deney Tasarımı

Tablo 4.17 Higgs veri kümesi için deney tasarımı varyans analizi sonuçlarını göstermektedir. Tabloda p değerlerinden de görülebileceği gibi gizli katman ve iterasyon parametrelerinin her ikisi de ve iki parametre arasındaki gizli katman*iterasyon ikili etkileşimi de sınıflandırma doğruluğu etkinlik ölçütü üzerinde etkili çıkmıştır.

Tablo 4.17 Higgs veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	7	0.005384	0.000769	146.74	0.000
Lineer	4	0.005055	0.001264	241.07	0.000
Gizli katman	3	0.000417	0.000139	26.52	**0.000**
İterasyon	1	0.004638	0.004638	884.72	**0.000**
İkili etkileşimler	3	0.000330	0.000110	20.95	0.000
Gizli katman*İterasyon	3	0.000330	0.000110	20.95	**0.000**
Hata	72	0.000377	0.000005		
Total	79	0.005762			

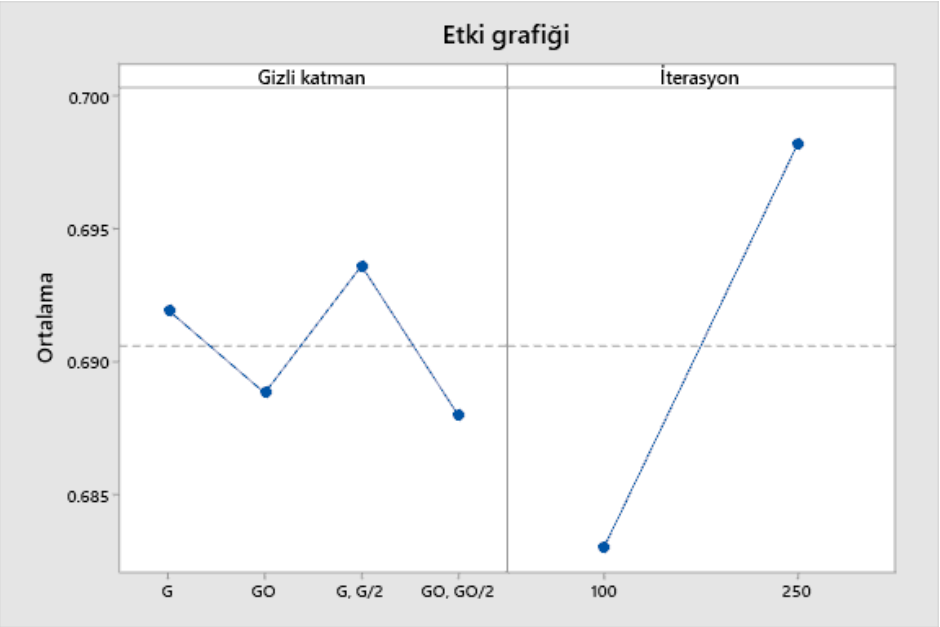
Varyans analizinden elde edilen sonuçları Şekil 4.25'de verilen standartlaştırılmış pareto grafiği de doğrulamaktadır. En etkili parametre iterasyon, ikinci etkili parametre gizli katman ve son olarak etkili olan ise gizli katman ile iterasyon arasındaki ikili etkileşimdir.



Şekil 4.25 Higgs veri kümesi standartlaştırılmış etkiler pareto grafiği

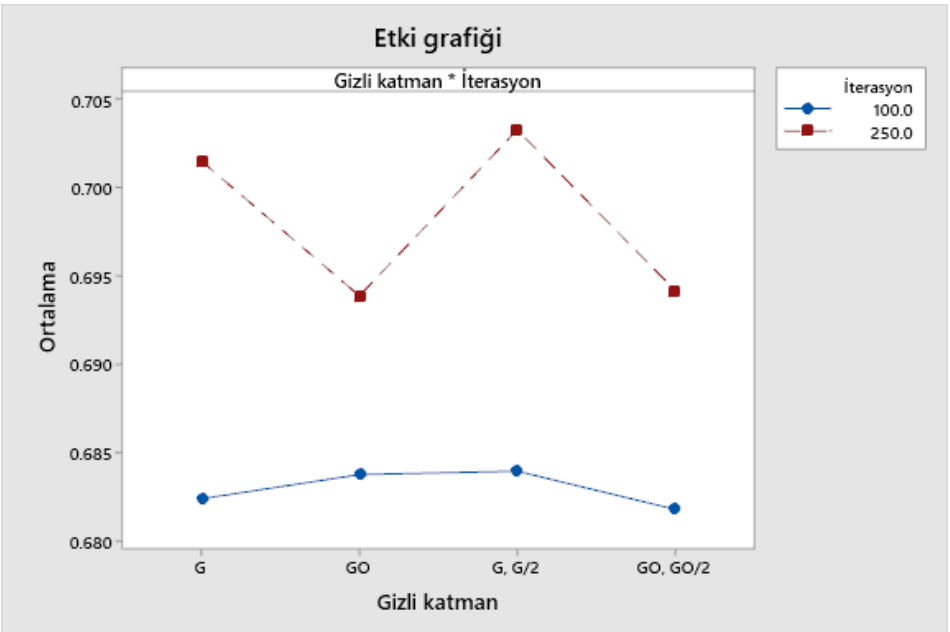
Higgs veri kümesinde iterasyon ve gizli katman parametrelerinin sınıflandırma doğruluğu etkinlik ölçütü üzerinde etkin düzeylerini belirleyebilmek için öncelikle Şekil 4.26'de verilen ana etkiler grafiği incelenmiştir. Parametrelerin etkin değerleri ise gizli katman için G-G/2 yani iki gizli katmanlı ve birinci gizli katmanda veri kümesinin girdi niteliği

kadar proses elemanı, ikinci katmanında ise girdi nitelik sayısının yarısı kadar proses elemanı olan YSA yapısı ve iterasyon parametresi için 250'dir.



Şekil 4.26 Higgs veri kümesi ana etkiler grafiği

Şekil 4.27'de Higgs veri kümesi için elde edilen ikili etkileşim grafiği gösterilmektedir. İkili etkileşimler incelendiğinde, ikili etkileşimlerin ana etkileri doğruladığı yani gizli katman parametresinin etkin düzeyinin G-G/2, iterasyon parametresinin etkin düzeyinin ise 250 olduğu görülmektedir.



Şekil 4.27 Higgs veri kümesi ikili etkileşim grafiği

Sonuç olarak Higgs veri kümesi için etkin parametre değerleri:

- Gizli katman: G-G/2
- İterasyon: 250 olarak belirlenmiştir.

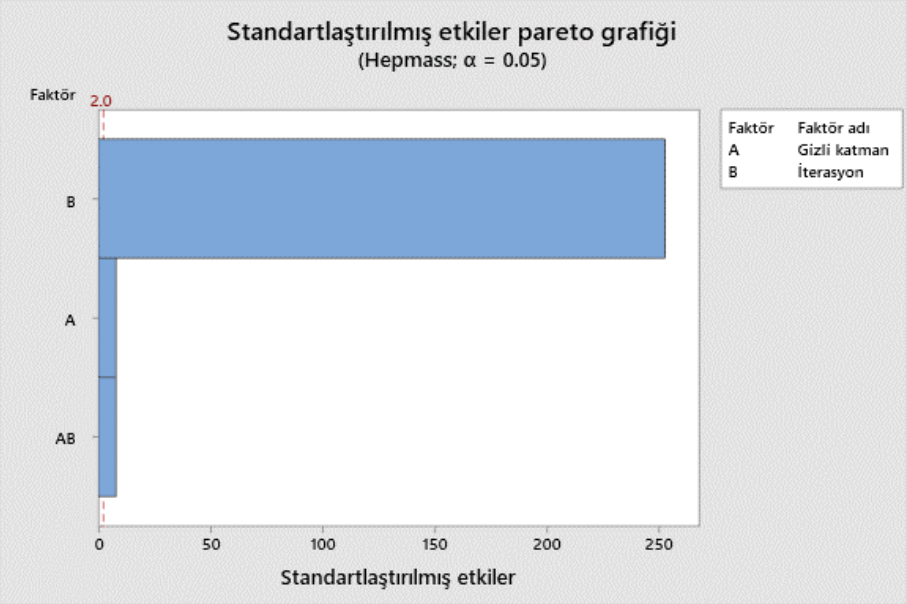
Hepmass Veri Kümesi Deney Tasarımı

YSA-KA algoritmasının Hepmass veri kümesi üzerinde deney tasarımı varyans analizi sonuçları Tablo 4.18'de gösterilmektedir. %95 anlamlılık seviyesinde, tabloda verilen p değerlerinden de görüldüğü gibi gizli katman ve iterasyon parametreleri, aynı zamanda gizli katman-iterasyon ikili etkileşimleri sınıflandırma doğruluğu etkinlik ölçütü üzerinde etkili parametreler (p değerleri<0.05) olarak belirlenmiştir

Tablo 4.18 Hepmass veri kümesi varyans analizi

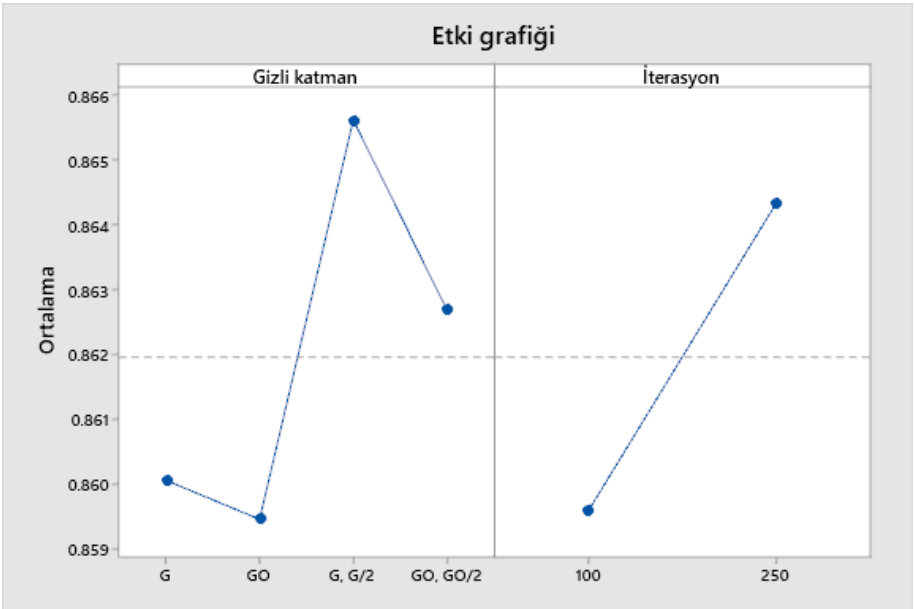
Kaynak	**Serbestlik Derecesi**	**Ayarlanmış SS**	**Ayarlanmış MS**	**F-Değeri**	**P-Değeri**
Model	7	0.001038	0.000148	20950.00	0.000
Lineer	4	0.000928	0.000232	32768.77	0.000
Gizli katman	3	0.000476	0.000159	22406.97	**0.000**
İterasyon	1	0.000452	0.000452	63854.16	**0.000**
İkili etkileşimler	3	0.000110	0.000037	5191.65	0.000
Gizli katman*İterasyon	3	0.000110	0.000037	5191.65	**0.000**
Hata	72	0.000001	0.000000		
Total	79	0.001039			

Şekil 4.28'de verilen standartlaştırılmış etkiler pareto grafiğinden de gizli katman ve iterasyon parametrelerinin ve gizli katman-iterasyon etkileşiminin etkili olduğu görülmektedir. Etkili parametre ve etkileşimden iterasyon parametresi sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde diğerlerine göre çok daha etkilidir.



Şekil 4.28 Hepmass veri kümesi standartlaştırılmış etkiler pareto grafiği

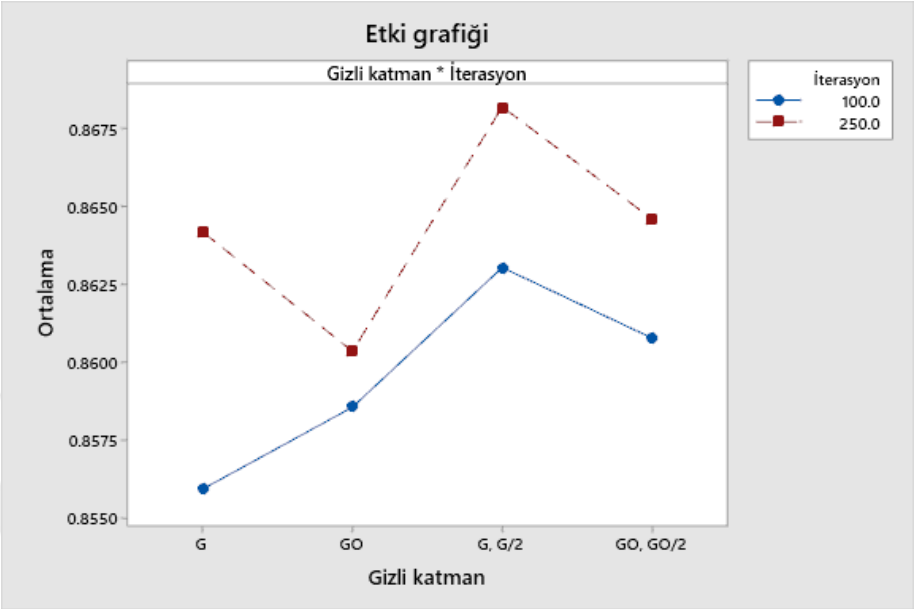
Şekil 4.29 Hepmass veri kümesi için ana etkiler grafiğini göstermektedir. Şekilden görüldüğü gibi gizli katman parametresinin sınıflandırma doğruluğu üzerinde etkin değeri G-G2, iterasyon parametresinin etkinlik ölçütü üzerinde etkin değeri ise 250'dir.



Şekil 4.29 Hepmass veri kümesi ana etkiler grafiği

Şekil 4.30'de Hepmass veri kümesi için elde edilen ikili etkileşim grafiği gösterilmektedir. İkili etkileşimler incelendiğinde, ikili etkileşimlerin ana etkileri

doğruladığı yani gizli katman parametresinin etkin düzeyinin G-G/2, iterasyon parametresinin etkin düzeyinin ise 250 olduğu görülmektedir.



Şekil 4.30 Hepmass veri kümesi ikili etkileşim grafiği

Sonuç olarak Higgs veri kümesi için etkin parametre değerleri:

- Gizli katman: G-G/2
- İterasyon: 250 olarak belirlenmiştir.

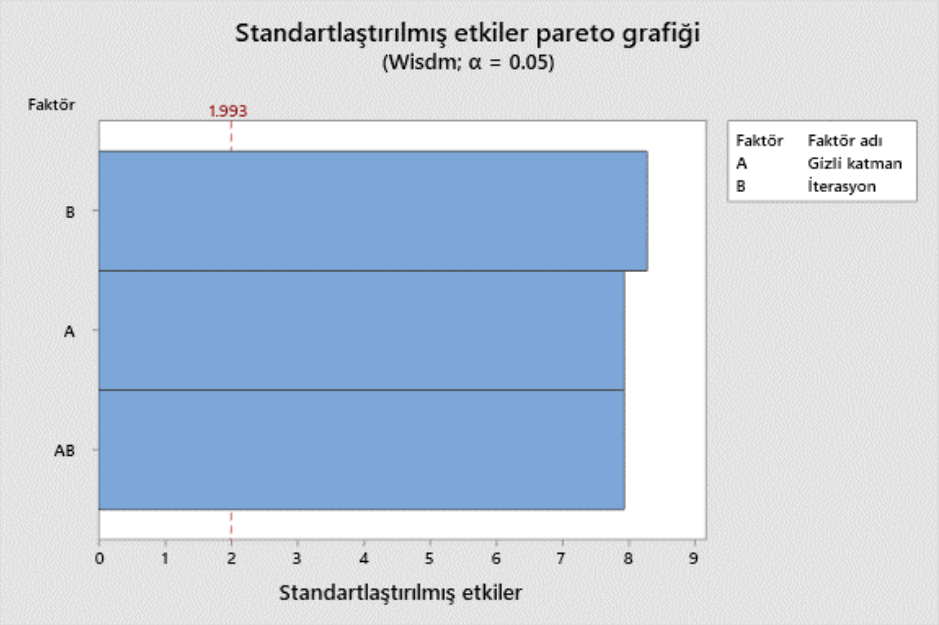
Wisdm Veri Kümesi Deney Tasarımı

Wisdm veri kümesi için deney tasarımı varyans analizi sonuçları Tablo 4.19'de gösterilmiştir. Tablodan da görüldüğü gibi gizli katman ve iterasyon parametreleri ve gizli katman*iterasyon sayısı ikili etkileşimleri etkindir (p değerleri <0.05).

Tablo 4.19 Wisdm veri kümesi varyans analizi

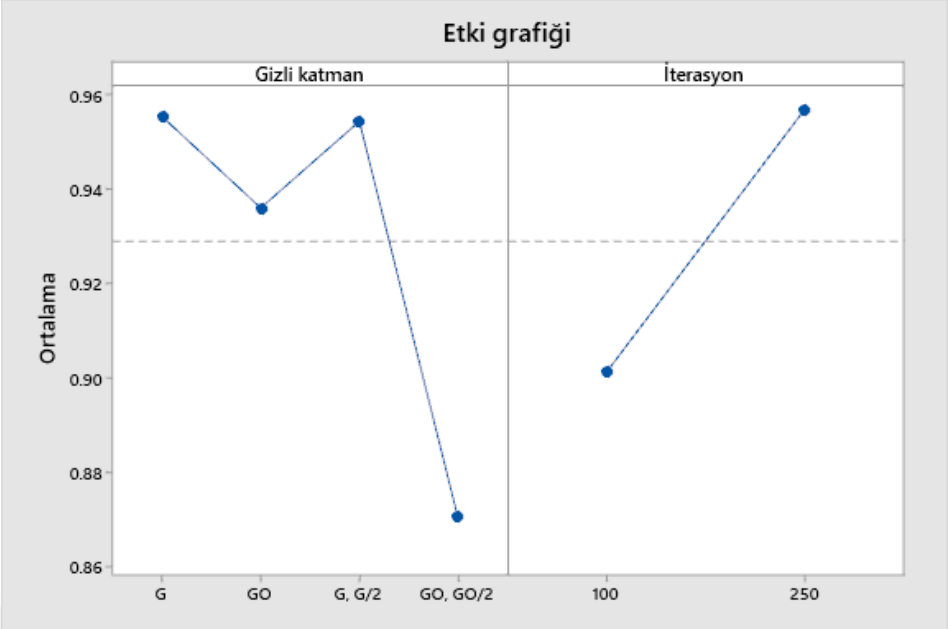
Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	7	0.25503	0.036432	40.52	0.000
Lineer	4	0.15797	0.039491	43.92	0.000
Gizli katman	3	0.09623	0.032077	35.67	**0.000**
İterasyon	1	0.06174	0.061736	68.66	**0.000**
İkili etkileşimler	3	0.09706	0.032353	35.98	0.000
Gizli katman*İterasyon	3	0.09706	0.032353	35.98	**0.000**
Hata	72	0.06474	0.000899		
Total	79	0.31977			

Şekil 4.31'de verilen standartlaştırılmış etkiler pareto grafiğine göre de gizli katman, iterasyon parametreleri ve gizli katman-iterasyon ikili etkileşimi oldukça etkili çıkmıştır.



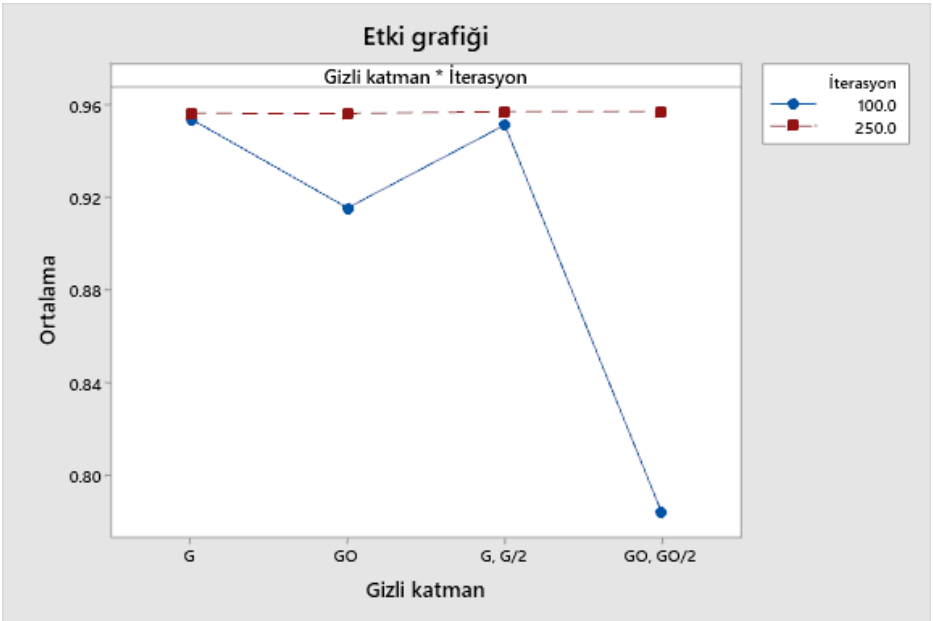
Şekil 4.31 Wisdm veri kümesi standartlaştırılmış etkiler pareto grafiği

Şekil 4.32'de Wisdm veri kümesi için deney tasarımı sonucu oluşturulan ana etkiler grafiği gösterilmektedir. Ana etkiler grafiğine göre iki parametre de etkilidir ve etkin düzeyler gizli katman parametresi için G-G/2, iterasyon parametresi için ise 250'dir.



Şekil 4.32 Wisdm veri kümesi ana etkiler grafiği

Wisdm veri kümesi için Şekil 4.33'de verilen ikili etkileşim grafiği incelendiğinde iterasyon parametresi için etkin düzeyin net bir şekilde 250 olduğu görülmektedir. İterasyon parametresi 250 değerini aldığında, gizli katman parametresi için parametre değerlerinin sınıflandırma doğruluğu üzerindeki etkileri birbirine çok yakındır.



Şekil 4.33 Wisdm veri kümesi ikili etkileşim grafiği

Sonuç olarak Wisdm veri kümesi için etkin parametre değerleri:

- Gizli katman: G-G/2
- İterasyon: 250 olarak belirlenmiştir.

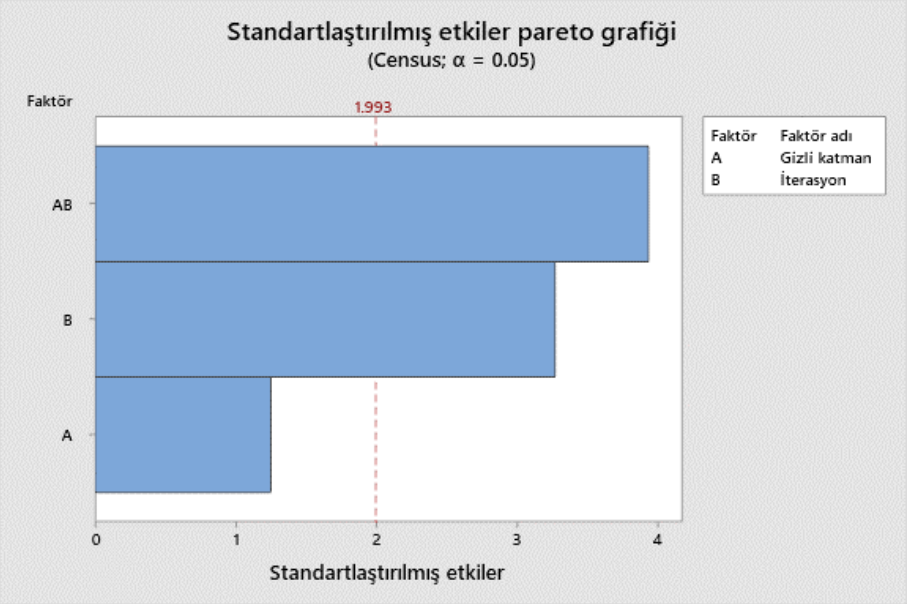
Census Veri Kümesi Deney Tasarımı

Census veri kümesi için deney tasarımı sonucu elde edilen varyans analizi sonuçları Tablo 4.20'de gösterilmiştir. %95 anlamlılık düzeyinde iterasyon parametresi ve gizli katman-iterasyon etkileşimi sınıflandırma doğruluk oranı üzerinde etkilidir. Gizli katman parametresi ise etkisizdir.

Tablo 4.20 Census veri kümesi varyans analizi

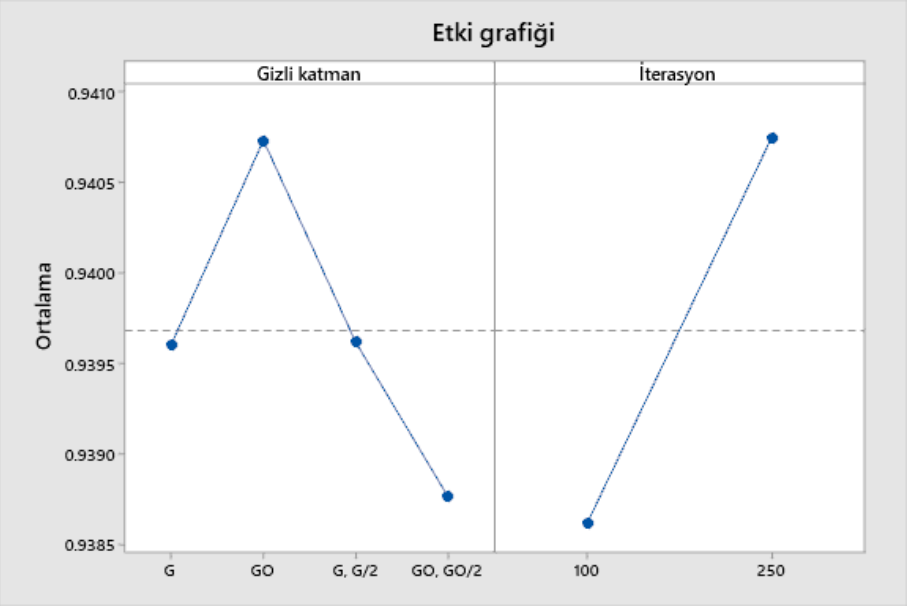
Kaynak	**Serbestlik Derecesi**	**Ayarlanmış SS**	**Ayarlanmış MS**	**F-Değeri**	**P-Değeri**
Model	7	0.000323	0.000046	5.41	0.000
Lineer	4	0.000130	0.000033	3.82	0.007
Gizli katman	3	0.000039	0.000013	1.53	0.215
İterasyon	1	0.000091	0.000091	10.72	**0.002**
İkili etkileşimler	3	0.000193	0.000064	7.53	0.000
Gizli katman*İterasyon	3	0.000193	0.000064	7.53	**0.000**
Hata	72	0.000614	0.000009		
Total	79	0.000937			

Şekil 4.34'de verilen standartlaştırılmış etkiler pareto grafiğinde gizli katman sayısı parametresinin etkili olmadığı, iterasyon sayısı parametresi ve gizli katman-iterasyon etkileşiminin etkili olduğu görülmektedir.



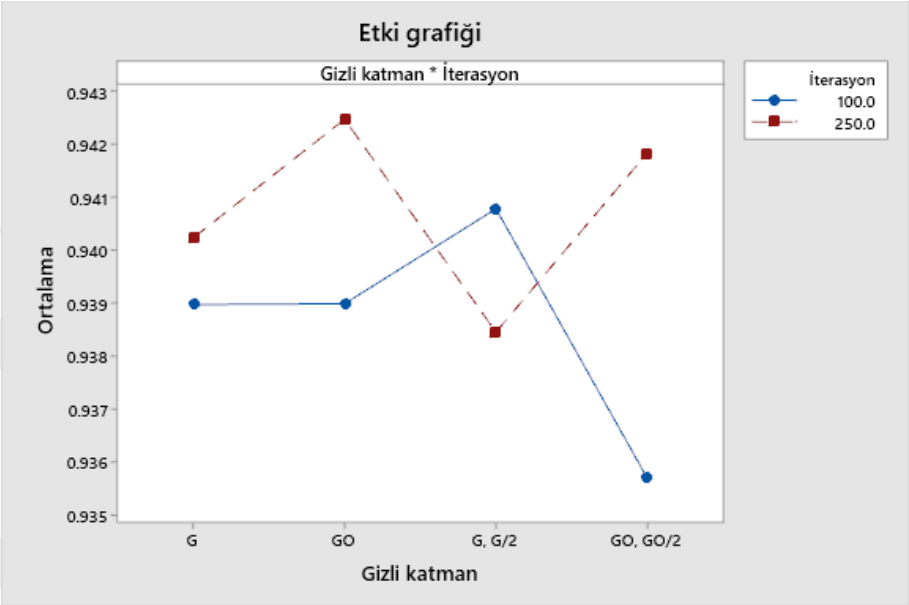
Şekil 4.34 Census veri kümesi standartlaştırılmış etkiler pareto grafiği

Gizli katman ve iterasyon sayısı parametrelerinin etkin düzeylerini belirlemek için Şekil 4.35'de verilen ana etkiler grafiği incelendiğinde gizli katman parametresi için GO (tek gizli katman ve gizli katmanda girdi nitelik ve sınıf sayısının geometrik ortalaması kadar proses elemanı), iterasyon sayısı için 250 düzeylerinin etkin olduğu gözlemlenmektedir.



Şekil 4.35 Census veri kümesi ana etkiler grafiği

Şekil 4.36'de gösterilen ikili etkileşimler de incelendiğinde gizli katman ile iterasyon parametreleri arasında etkileşim olduğu dolayısıyla parametrelerden birinin değerindeki değişimin diğerini de etkilediği söylenebilmektedir. İkili etkileşim grafiğine göre de ana etkiler grafiğinde elde edilen sonuçlar çıkmaktadır. Dolayısıyla gizli katman parametresinin etkin düzeyi GO, iterasyon parametresinin etkin düzeyi 250'dir.



Şekil 4.36 Census veri kümesi ikili etkileşim grafiği

Sonuç olarak Cencus veri kümesi için etkin parametre değerleri:

- Gizli katman: GO
- İterasyon: 250 olarak belirlenmiştir.

4.1.4. SH ve YSA-KA Algoritmaları ile Elde Edilen Sonuçlar

Deney tasarımı sonucunda elde edilen en iyi parametreler ile literatür verileri 30 kez çalıştırılmıştır. Elde edilen bu sonuçlar Tablo 4.21'deki gibidir. Pokerhand veri kümesinde YSA-KA algoritması, 0.586 oranındaki sonucuyla 0.519 oranına sahip olan SH algoritmasına büyük üstünlük sağlamıştır. Susy veri kümesi için YSA-KA algoritması, 0.802 olan doğruluk oranıyla 0.796 oranında doğruluk elde eden SH algoritmasını geçmiştir. Higgs veri kümesi için aynı şekilde YSA-KA algoritması 0.705 doğruluğa sahiptir ve 0.691 doğruluk oranına sahip SH algoritmasını geçmiştir. Benzer bir durum Hepmass veri kümesinde de gerçekleşmiştir. 0.871 doğruluğa sahip YSA-KA,

0.850 doğruluğa sahip SH algoritmasını geçmiştir. Wisdm veri kümesinde ise YSA-KA 0.961 ile 0.951 oranındaki SH'a karşı üstünlük sağlamıştır. Son olarak Census veri kümesine bakıldığında SH aldığı 0.945 oranıyla, 0.942 olan YSA-KA algoritmasına az bir farkla üstünlük sağlamıştır.

Sonuç olarak tez çalışması kapsamında geliştirilen SH ve YSA-KA algoritmalarından Spark hibrit algoritması ele alınan 6 veri kümesinden sadece bir tanesinde YSA-KA hibrit algoritmasından daha yüksek doğruluk değeri verirken, kalan bütün 5 veri kümesinde de yapay sinir ağı-karar ağacı hibrit algoritması SH algoritmasından daha yüksek doğruluklar vermiştir.

Önerilen algoritmalar çalışma süreleri açısından değerlendirildiğinde, YSA-KA algoritması bütün veri kümelerinde SH algoritmasına göre daha kısa sürede sonuç vermiştir, YSA-KA, SH algoritmasına göre daha hızlı bir algoritmadır.

Tablo 4.21 Literatür verileri üzerinde yeni algoritma sonuçları

	SH			**YSA-KA**		
Veri kümesi	**Doğruluk**	**S. sapma**	**Süre(s)**	**Doğruluk**	**S. sapma**	**Süre(s)**
Pokerhand	0.519	**0.003**	135.365	**0.586**	0.013	**78.167**
Susy	0.796	**0.001**	609.310	**0.802**	**0.001**	**483.311**
Higgs	0.691	0.002	1927.906	**0.705**	**0.001**	**1665.320**
Hepmass	0.850	**0.001**	1107.774	**0.871**	0.002	**892.701**
Wisdm	0.951	**0.007**	66.352	**0.961**	**0.007**	**45.413**
Census	**0.945**	**0.001**	77.203	0.942	**0.001**	**60.479**

4.1.5. SH ve YSA-KA Algoritmaları Performans Karşılaştırmaları

Deney tasarımı sonucu elde edilen en iyi parametre değerleriyle çalıştırılan algoritmalar sadece kendi aralarında değil ayrıca Spark içerisinde bulunan makine öğrenme algoritmalarıyla ve literatürde mevcut geçmiş çalışmalardaki sonuçlarla da karşılaştırılmıştır. Karşılaştırmalar sadece doğruluk etkinlik ölçütü açısından gerçekleştirilmiştir, literatürde mevcut çoğu algoritmada çalışma süreleri verilmediği için algoritmaların çalışma süreleri karşılaştırmalarda ele alınmamıştır. Spark'taki

algoritmalar; lojistik regresyon (LR), karar ağacı (KA), rastgele orman (RO), yapay sinir ağı (YSA) ve doğrusal destek vektör makinesi (DDVM) algoritmalarıdır. Doğrusal destek vektör makinesi sadece iki çıktısı olan sınıflandırma problemleri için çalışmaktadır. Bu yüzden Pokerhand ve Wisdm verileri için bu algoritma ile sonuç alınamamaktadır. Karşılaştırmada kullanılan ve Spark içerisinde yer alan algoritmaların literatür verileri üzerindeki doğruluk oranı üzerinden etkinlikleri Tablo 4.22'de görülebilir.

Tablo 4.22 Literatür verileri üzerinde algoritmaların karşılaştırılması

Algoritma	**Pokerhand**	**Susy**	**Higgs**	**Hepmass**	**Wisdm**	**Census**
SH	0.519	0.796	0.691	0.850	0.951	0.945
YSA-KA	**0.586**	**0.802**	**0.705**	**0.871**	0.961	0.942
LR	0.501	0.789	0.638	0.836	0.970	**0.948**
KA	0.551	0.768	0.663	0.834	0.894	0.944
RO	0.549	0.776	0.666	0.820	0.930	0.940
YSA	0.523	0.798	0.652	0.858	**0.971**	0.942
DDVM	-	0.789	0.629	0.837	-	0.940

Tablo 4.22'de görüldüğü gibi Pokerhand veri kümesi için en yüksek doğruluk değerini 0.586 ile YSA-KA algoritması vermiştir. YSA-KA'dan sonra en iyi sonuç 0.551 ile karar ağacı algoritmasından elde edilmiştir. SH algoritması, 0.519 doğruluk değerini vermiştir. Susy veri kümesi için en iyi sonucu YSA-KA 0.802 doğruluk değeri ile vermiştir. YSA 0.798 doğruluk değeri ile ikinci sırada yer almıştır. SH algoritması ise 0.796 doğruluk ile en iyiye çok yakın bir doğruluk değeri vermiştir. Higgs veri kümesi için en iyi doğruluk 0.705 değeridir ve bunu YSA-KA hibrit algoritması sağlamıştır. YSA-KA'ya en yakın değer 0.691 ile SH algoritması ile elde edilmiştir. Hepmass veri kümesi için en iyi doğruluk oranını YSA-KA algoritması 0.871 ile gerçekleştirmiştir. Daha sonra 0.858 doğruluk değeri ile YSA algoritması gelmektedir. SH algoritması 0.850 doğruluk ile üçüncü sırada yer almaktadır. Wisdm veri kümesi için en iyi değeri YSA 0.971 doğruluk ile gerçekleştirmiştir. YSA-KA 0.961, SH ise 0.951 oranında doğruluğa sahiptirler. YSA algoritmasının tek başına YSA-KA'yı geçmesinin sebebi olarak Wisdm veri kümesinin 6 çıktı değerinin çok yakın ihtimaller vermesi sonucu oluşan durum gösterilebilir. Böylece YSA, YSA-KA'yı çok az bir farkla geçmiştir. Census veri kümesi için en yüksek

doğruluğu 0.948 ile LR algoritması vermiştir. SH 0.945, YSA-KA ise 0.942 ile en iyiye çok yakın doğruluklar vermişlerdir. Sonuç olarak, ele alınan 6 veri kümesinin dördünde YSA-KA hibrit algoritması en iyi sonucu verirken diğer 2 veri kümesinde de en iyiye yakın değerler elde edilmiştir. SH algoritması ise ele alınan veri kümelerinin üçünde en iyiye yakın sonuçlar verirken kalan üç veri kümesinde daha düşük doğruluklar üretmiştir.

Geçmişte literatürde yapılan çalışmalar ile sonuçlar karşılaştırılırken geçmiş çalışmalarda veri üzerinde ön işlem, ağırlıklandırma, nitelik seçimi gibi doğruluk değerini algoritmadan bağımsız olarak etkileyebilecek yöntemler dahil edilmemiştir. Karşılaştırmada dikkate alınan algoritmalar ve literatür verileri üzerindeki etkinlikleri Tablo 4.23'de görülebilir. Etkinlik kriteri olarak doğruluk oranları kullanılmıştır. Tabloda (-) işaretleri ile gösterilen değerler çalışmada ilgili veri kümesi ile ilgili sonucun bulunmadığını ifade etmektedir.

Tablo 4.23 Literatür çalışmaları ile karşılaştırmalar

Algoritma	Poker hand	Susy	Higgs	Hepmass	Wisdm	Census
SH	0.519	0.796	0.691	0.850	0.951	**0.945**
YSA-KA	**0.586**	**0.802**	**0.705**	**0.871**	0.961	0.942
Za'in vd. (2019) [10] Yapay sinir ağı	0.511	0.765	0.638	0.838	-	-
Maillo vd. (2018) [15] K en yakın komşu	0.546	0.751	0.616	-	-	-
Maillo vd. (2020) [7] K en yakın komşu	0.550	0.751	0.621	-	-	-
Garcia-Gil vd., (2018) [63] Karar ağacı	0.551	0.785	0.694	-	-	-
Weinberg ve Last (2019) [11] Rastgele orman	0.574	0.781	0.653	-	-	-
Triguero vd. (2015) [29] En yakın komşu	0.518	0.728	-	-	-	-
Maillo vd. (2017) [19] K en yakın komşu	0.534	0.728	-	-	-	-
Maillo vd. (2017) [21] K en yakın komşu	0.539	0.738	-	-	-	-
Weinberg ve Last (2017) [20] Karar ağacı	0.563	0.791	-	-	-	-
Silva vd. (2016) [27] K en yakın komşu	0.548	-	-	-	-	0.943
Lenz vd. (2020) [8] K en yakın komşu	-	0.785	0.679	0.844	-	-
Wu vd. (2019) [38] En yakın komşular	-	0.640	0.530	-	-	-
Oo ve Thein (2019) [9] Rastgele orman	-	0.675	-	-	-	-
Shashidhara vd. (2015) [30] Lojistik regresyon	-	-	0.608	-	-	-
Inoue vd. (2018) [16] Derin tekrarlı YSA	-	-	-	-	0.950	-
Ronao ve Cho (2016) [25] Evrişimsel YSA	-	-	-	-	0.958	-
Ronao ve Cho (2017) [26] Markov modeller	-	-	-	-	0.932	-
Ignatov (2018) [17] Evrişimsel YSA	-	-	-	-	0.948	-
Anguita vd. (2013) [33] Destek vektör makinesi	-	-	-	-	**0.964**	-
Fish vd. (2016) [28] Destek vektör makinesi	-	-	-	-	-	0.853
Chakrabarty ve Biswas (2018) [18] Rastgele orman	-	-	-	-	-	0.882
Kamiran vd. (2010) [34] Karar ağacı	-	-	-	-	-	0.860
Oza (2005) [31] Naive Bayes	-	-	-	-	-	0.940

Tablo 4.23'de görüldüğü üzere karşılaştırmada ele alınan 6 veri kümesinden dördünde YSA-KA, birinde ise SH algoritması en yüksek doğruluk değerlerine sahiptir. Pokerhand veri kümesi için en iyi sonucu 0.586 doğruluk oranı ile YSA-KA algoritması vermiştir. YSA-KA'dan sonra en iyi sonuç 0.574 ile Weinberg ve Last'ın 2019 yılında yaptıkları çalışmadan elde edilmiştir. Susy veri kümesi için en iyi sonucu YSA-KA 0.802 başarı oranı ile vermiştir. SH algoritması 0.796 doğruluk değeri ile ikinci sırada yer almaktadır. Higgs veri kümesi için en iyi doğruluk 0.705 değeridir ve bunu YSA-KA hibrit algoritması sağlamıştır. YSA-KA'ya en yakın değer 0.694 ile Garcia-Gil ve arkadaşlarının 2018 yılında yaptıkları çalışmada elde edilmiştir. Hepmass veri kümesi için en iyi doğruluk oranını YSA-KA algoritması 0.871 ile gerçekleştirmiştir. Daha sonra 0.850 ile SH algoritması gelmektedir. Wisdm veri kümesi için en iyi değeri Anguita ve arkadaşlarının çalışması 0.964 ile gerçekleştirmiştir. YSA-KA 0.961 doğruluk değeri ile ikinci sırada yer almaktadır. Census veri kümesi için en yüksek doğruluğu 0.945 ile SH algoritması vermiştir. YSA-KA ise 0.942 ile en iyiye çok yakın sonuç vermiştir.

4.1.6. İstatistiksel Analiz

Yeni geliştirilen SH ve YSA-KA algoritmaları ile Spark'ta bulunan makine öğrenme algoritmalarının literatür veri kümeleri üzerinde elde ettikleri sonuçlar arasında anlamlı bir farklılık olup olmadığını ortaya koymak için istatistiksel analizler gerçekleştirilmiştir. Her bir algoritma, her bir veri kümesinde 30 kez çalıştırılmış ve sonuçlar alınmıştır. Her bir algoritmada ve veri kümesinde 30 çalıştırma sonucu yer aldığından dolayı merkezi limit teoremi gereğince elde edilen sonuçların normal dağılıma uyduğu varsayılabilir. Böylece istatistiksel analiz olarak eşleştirilmiş-t testi kullanılabilir.

SH ve YSA-KA algoritmaları ile karşılaştırmada dikkate alınan algoritmaların doğruluk değerleri arasındaki farkların istatistiksel olarak anlamlı olup olmadığını anlamak için yapılan eşleştirilmiş-t testi sonuçları Tablo 4.24'de sunulmuştur. % 95 güven aralığında elde edilen p değerleri incelenmiştir. Sadece 0.180 değeriyle Susy veri kümesi için SH/YSA algoritmaları arasında doğruluk etkinlik ölçütü açısından anlamlı bir fark ortaya çıkmamıştır ($p>0.05$). Diğer bütün sonuçlardaki farklar anlamlıdır. Bu sonuçlara göre Pokerhand, Susy, Higgs, Hepmass veri kümelerinde YSA-KA algoritması en iyisidir. Wisdm ve Census veri kümelerinde ise sırasıyla YSA ve LR algoritmaları en iyisidir. Bu farkların hepsi de istatistiksel olarak anlamlıdır.

Tablo 4.24 Algoritmaların istatistiksel karşılaştırılması

Algoritmalar	Pokerhand	Susy	Higgs	Hepmass	Wisdm	Census
YSA-KA/SH	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
YSA-KA/LR	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
YSA-KA/KA	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
YSA-KA/RO	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
YSA-KA/YSA	**0.000**	**0.008**	**0.000**	**0.000**	**0.000**	**0.000**
YSA-KA/DDVM	-	**0.000**	**0.000**	**0.000**	-	**0.000**
SH/LR	**0.000**	**0.001**	**0.000**	**0.000**	**0.000**	**0.000**
SH/KA	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
SH/RO	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
SH/YSA	**0.000**	0.180	**0.000**	**0.000**	**0.000**	**0.000**
SH/DDVM	-	**0.000**	**0.000**	**0.000**	-	**0.000**

İstatistiksel analizler kapsamında, SH/YSA-KA algoritmaları ve literatürde mevcut algoritmalar ile elde edilen doğruluk değerleri aralarındaki farkların istatistiksel olarak anlamlı olup olmadığı da test edilmiştir. Literatürden alınan çalışmalarda sadece tek değer verildiği, algoritmaların 30 çalıştırılması sonucu elde edilen değerler elimizde mevcut olmadığından dolayı istatistiksel analiz aracı olarak tek-örneklem-t testi kullanılmıştır.

SH algoritması ile literatürde mevcut algoritmaların karşılaştırılması Tablo 4.25'de verilmiştir. Sadece Wisdm veri kümesinde SH ile [16] arasında anlamlı bir farklılık yoktur. Diğer bütün değerler istatistiksel olarak anlamlı bir farklılığa sahiptir. Bu sonuçlara göre SH algoritması literatür sonuçları ile karşılaştırıldığında; Pokerhand veri kümesinde alt sıralarda, Susy, Hepmass ve Census veri kümelerinde en iyi sırada, Higgs veri kümesinde ikinci sırada, Wisdm veri kümesinde ise üçüncü sıradadır.

Tablo 4.25 SH algoritması ile literatür verileri istatistiksel analizi

Algoritma	Pokerhand	Susy	Higgs	Hepmass	Wisdm	Census
SH/[10]	**0.000**	**0.000**	**0.000**	**0.000**	-	-
SH/[15]	**0.000**	**0.000**	**0.000**	-	-	-
SH/[7]	**0.000**	**0.000**	**0.000**	-	-	-
SH/[63]	**0.000**	**0.000**	**0.000**	-	-	-
SH/[11]	**0.000**	**0.000**	**0.000**	-	-	-
SH/[29]	**0.000**	**0.000**	-	-	-	-
SH/[19]	**0.000**	**0.000**	-	-	-	-
SH/[21]	**0.000**	**0.000**	-	-	-	-
SH/[20]	**0.000**	**0.000**	-	-	-	-
SH/[27]	**0.000**	-	-	-	-	**0.000**
SH/[8]	-	**0.000**	**0.000**	**0.000**	-	-
SH/[38]	-	**0.000**	**0.000**	-	-	-
SH/[9]	-	**0.000**	-	-	-	-
SH/[30]	-	-	**0.000**	-	-	-
SH/[16]	-	-	-	-	0.264	-
SH/[25]	-	-	-	-	**0.000**	-
SH/[26]	-	-	-	-	**0.000**	-
SH/[17]	-	-	-	-	**0.008**	-
SH/[33]	-	-	-	-	**0.000**	-
SH/[28]	-	-	-	-	-	**0.000**
SH/[18]	-	-	-	-	-	**0.000**
SH/[34]	-	-	-	-	-	**0.000**
SH/[31]	-	-	-	-	-	**0.000**

YSA-KA algoritması ile literatürdeki verilerin karşılaştırılması Tablo 4.26'te verilmiştir. Sadece Wisdm veri kümesinde YSA-KA ile [33] arasındaki p değeri 0.05 çıkmıştır. Diğer bütün değerler 0.05'ten küçüktür, dolayışıysa karşılaştırmalardaki farklar istatistiksel olarak anlamlıdır. Bu sonuçlara göre YSA-KA algoritması literatür sonuçları ile karşılaştırıldığında; Pokerhand, Susy, Higgs, Hepmass veri kümelerinde birinci sırada,

Wisdm veri kümesinde literatürdeki sonuç [33] ile birlikte birinci sırada, Census veri kümesinde ise ikinci sıradadır.

Tablo 4.26 YSA-KA algoritması ile literatür verileri istatistiksel analizi

Algoritma	**Pokerhand**	**Susy**	**Higgs**	**Hepmass**	**Wisdm**	**Census**
YSA-KA/[10]	**0.000**	**0.000**	**0.000**	**0.000**	-	-
YSA-KA/[15]	**0.000**	**0.000**	**0.000**	-	-	-
YSA-KA/[7]	**0.000**	**0.000**	**0.000**	-	-	-
YSA-KA/[63]	**0.000**	**0.000**	**0.000**	-	-	-
YSA-KA/[11]	**0.000**	**0.000**	**0.000**	-	-	-
YSA-KA/[29]	**0.000**	**0.000**	-	-	-	-
YSA-KA/[19]	**0.000**	**0.000**	-	-	-	-
YSA-KA/[21]	**0.000**	**0.000**	-	-	-	-
YSA-KA/[20]	**0.000**	**0.000**	-	-	-	-
YSA-KA/[27]	**0.000**	-	-	-	-	**0.000**
YSA-KA/[8]	-	**0.000**	**0.000**	**0.000**	-	-
YSA-KA/[38]	-	**0.000**	**0.000**	-	-	-
YSA-KA/[9]	-	**0.000**	-	-	-	-
YSA-KA/[30]	-	-	**0.000**	-	-	-
YSA-KA/[16]	-	-	-	-	**0.000**	-
YSA-KA/[25]	-	-	-	-	**0.012**	-
YSA-KA/[26]	-	-	-	-	**0.000**	-
YSA-KA/[17]	-	-	-	-	**0.000**	-
YSA-KA/[33]	-	-	-	-	**0.050**	-
YSA-KA/[28]	-	-	-	-	-	**0.000**
YSA-KA/[18]	-	-	-	-	-	**0.000**
YSA-KA/[34]	-	-	-	-	-	**0.000**
YSA-KA/[31]	-	-	-	-	-	**0.000**

Bütün bu sonuçlara ve istatistiksel analizlere bakıldığında YSA-KA algoritmasının çok iyi bir algoritma olduğu, SH algoritmasının ise en iyi olmasa da yeterli bir seviyede olduğu görülmektedir.

4.2. Gerçek Hayat Veri Kümeleri Üzerindeki Çalışmalar

4.2.1. Gerçek Hayat Veri Kümeleri

Bu çalışmada kullanılan gerçek hayat verileri Kayseri'de üretim yapan bir firmanın yaklaşık olarak 17 yıldır tuttuğu üretim verileri ve Türkiye'de satılan ikinci araç verileridir.

Üretim Verisi

Üretim verisi Kayseri'de üretim yapan büyük bir fabrikanın yaklaşık 17 yıl boyunca tuttuğu verilerden bir kısımdır. Tutulan veriler farklı tablolarda saklanmıştır. Boyutu yüksek olduğu için büyük veri araçları kullanılmadan analizi yapılamamaktadır. Bu veri kümesi toplam 9 tablodan oluşmaktadır ve üretimle alakalı bilgilere sahiptir. Bu tablolar:

- Vardiya Sayaç Bilgileri
- Makine Duruşları
- Vardiya Duruş Bilgileri
- Makine Duruş Aylık
- Makine Duruş Haftalık
- Makine Duruş Günlük
- Makine Randıman Günlük
- Makine Ürün Randıman Günlük
- Otomasyon Üretim Vardiya Değerleri

Bu tablolar üzerinden üretimin verimini ölçen sınıflandırma türünde bir veri kümesi üretilmiştir. Bu üretim verisinin girdileri personel, tezgâh, vardiya, yıl, ay, hafta, üretim merkezi, personel grubu, tezgâh tipi, üretim merkezi tipidir. Bu girdilere göre üretim verimi tahmini yapılmaktadır.

Tablo 4.27 Üretim veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Sınıf sayısı
Üretim	1.048.576	10	5

İkinci El Araba Verisi

İkinci el araba verisi, Türkiye'de ilana koyulan ikinci el arabaların tutulduğu bir veri kümesidir. İnternet ortamındaki ilanlardan toplanarak oluşturulmuştur. Türkiye'de ikinci el araba pazarı oldukça hareketlidir. Araba fiyatına etki eden faktörler marka, model, model yılı, yakıt türü, vites türü, motor hacmi, motor gücü, kilometre, hasar durumu, takas durumu, satıcı durumudur. Bu faktörlere bağlı olarak fiyat belirlenmektedir. İkinci el araba veri kümesi yaklaşık 120.000 araba verisinden oluşmaktadır. Her bir veri için 11 nitelik değeri ve hedef değişkeni ifade eden fiyat değeri mevcuttur. Veri kümesinden fiyatı 1.000.000 TL (bir milyon TL) üzeri ve 1.000 TL (bin TL) altında olan örnekler çıkartılmıştır. Bunun sebebi fiyat giren kişilerin fazla veya az sıfır koymaları neticesinde hatalı veri girişi olması ve doğru bilgi içermemeleridir. Örneğin araba fiyatı 75.000 TL (yetmiş beş bin TL) ise araba sahibi fiyatı girerken kuruş cinsinden düşünerek üç sıfır fazla yazmaktadır ve 75.000.000 TL (yetmiş beş milyon TL) olarak veri girişi yapmaktadır. Buna benzer şekilde 75 TL (yetmiş beş TL) olarak da veri girişi yapanlar mevcuttur. Bu durumların veri kümesini olumsuz etkilememesi için bu düzenleme yapılmıştır. Bu veri kümesinde niteliklerin 8 tanesi kategori şeklinde, diğer 3 tanesi ise tamsayı şeklindedir.

Tablo 4.28 İkinci el araba veri kümesi özellikleri

Veri kümesi	Örnek sayısı	Sürekli nitelik sayısı	Kesikli nitelik sayısı	Hedef değer
İkinci el araba	104.706	3	11	Araba fiyatı

4.2.2. Etkinlik Ölçütleri ve Veri Kümelerinin Eğitim-Test Kümelerine Parçalanması

Algoritmalar karşılaştırılırken etkinlik ölçütü olarak sınıflandırma test doğruluğu kullanılmıştır. Bu oran Eşitlik (4.1)'de verilmiştir.

Fonksiyon tahminlemede ise en küçük hata kareleri toplamı karekökü (RMSE) etkinlik ölçütü olarak kullanılmıştır. RMSE formülü Eşitlik (4.2)'de gösterildiği gibidir.

Fonksiyon tahminlemede ise en küçük hata kareleri toplamı karekökü (RMSE) ve determinasyon katsayısı (R2) etkinlik ölçütü olarak kullanılmıştır. RMSE ve R2 formülleri sırasıyla Eşitlik (4.2) ve (4.3)'de gösterildiği gibidir

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y - y_i)^2} \tag{4.2}$$

$$R^2 = 1 - \frac{\sum(y - y_i)^2}{\sum(y - \bar{y})^2} \tag{4.3}$$

Veri kümelerinin eğitim ve test kümelerine parçalanmasında literatür verilerinde olduğu gibi basit bölümleme kullanılmıştır. Veri kümeleri, %75 oranında eğitim, %25 oranında test veri kümelerine parçalanmıştır.

4.2.3. Deney Tasarımı

Gerçek hayat veri kümeleri için deney tasarımında bir sınıflandırma veri kümesi olan üretim verisinde sınıflandırma doğruluk oranı, bir fonksiyon tahminleme veri kümesi olan ikinci el araba veri kümesinde ise RMSE etkinlik ölçütleri ele alınmıştır. Literatür veri kümelerinde olduğu gibi yine gerçek hayat veri kümeleri için de tam faktöriyel deney tasarımı, bu veri kümelerinde etkili parametreleri ve etkinlik düzeylerini belirlemek için kullanılmıştır. Tam faktöriyel deney tasarımında sınıflandırma doğruluk oranı veya RMSE üzerinde faktörlerin etkileri analiz edilirken, varyans analizi sonuçları, ana etkiler ve ikili etkileşimler dikkate alınmıştır.

Tez çalışması kapsamında önerilen iki algoritma, her bir gerçek hayat veri kümesi için algoritma parametrelerinin belirlenen düzeylerinde 10'ar kez çalıştırılmış ve etkinlik ölçütleri hesaplanmıştır. Deneysel tasarımda analizlerin her bir veri kümesi için ayrı ayrı ele alınmasının nedeni, hem gerçek hayat veri kümelerinin etkinlik ölçütlerinin farklı olması, hem de veri kümesindeki örnek ve değişken sayılarının değişmesi ile faktörlerin etkinliğinin değişebilmesidir.

Spark Hibrit Algoritması Deney Tasarımı

Spark Hibrit algoritması, daha önce de bahsedildiği gibi parça sayısı ve oylama türü olmak üzere iki parametreye sahiptir. Deney tasarımında parametre düzeyleri olarak literatür verileri için de kullanılan düzeyler ele alınmıştır. SH algoritması parametreleri ve deney tasarımında kullanılan düzeyleri Tablo 4.28'de verilmiştir.

Tablo 4.28 SH algoritması parametreleri ve parametre düzeyleri

Parametre	**Değerler**
Parça sayısı	10 – 20 – 30
Oylama türü	1 – doğruluk (R^2)

Deney tasarımında 3x2=6 deney bulunmaktadır. Her bir veri kümesi ve tasarımın her bir deneyi için algoritma 10 kez çalıştırılarak bu 10 çalıştırmanın üretim verisi için ortalama sınıflandırma doğruluk oranı, ikinci el araba gerçek hayat verisi için ise RMSE değeri, deney tasarımında etkinlik ölçütü olarak ele alınmıştır.

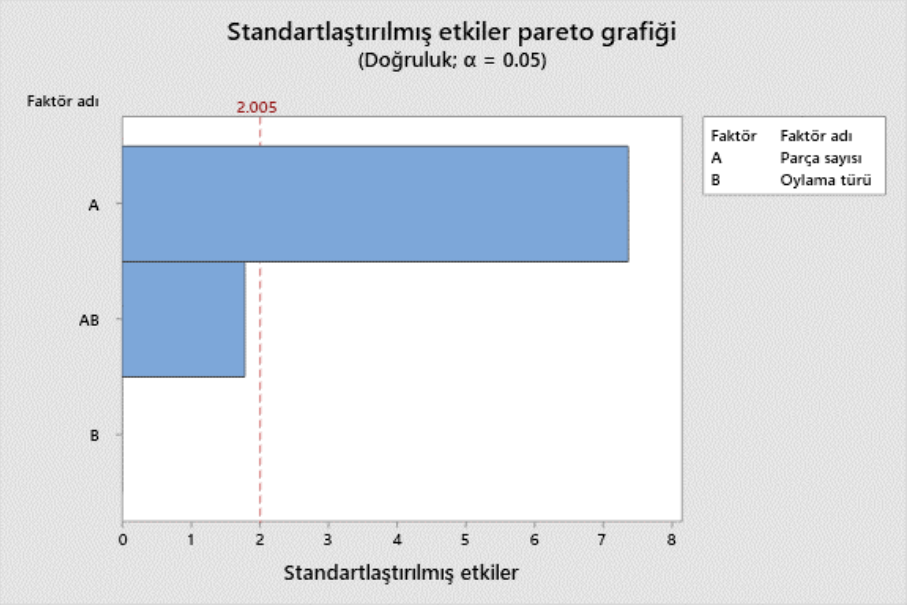
Üretim Veri Kümesi Deney Tasarımı

Üretim gerçek hayat veri kümesi için varyans analizi sonuçları Tablo 4.29'de verilmiştir. Anlamlılık düzeyi $\alpha=0.05$ olarak dikkate alındığında sadece parça sayısı parametresinin sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili olduğu (p değerleri <0.05)., oylama türü parametresi ve parça sayısı*oylama türü etkileşiminin etkili olmadığı görülmektedir (p değerleri >0.05).

Tablo 4.29 Üretim veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.000150	0.000030	13.51	0.000
Lineer	3	0.000138	0.000046	20.75	0.000
Parça sayısı	2	0.000138	0.000069	31.12	**0.000**
Oylama türü	1	0.000000	0.000000	0.00	1.000
2-li Etkileşimler	2	0.000012	0.000006	2.66	0.079
Parça sayısı*Oylama türü	2	0.000012	0.000006	2.66	0.079
Hata	54	0.000120	0.000002		
Toplam	59	0.000269			

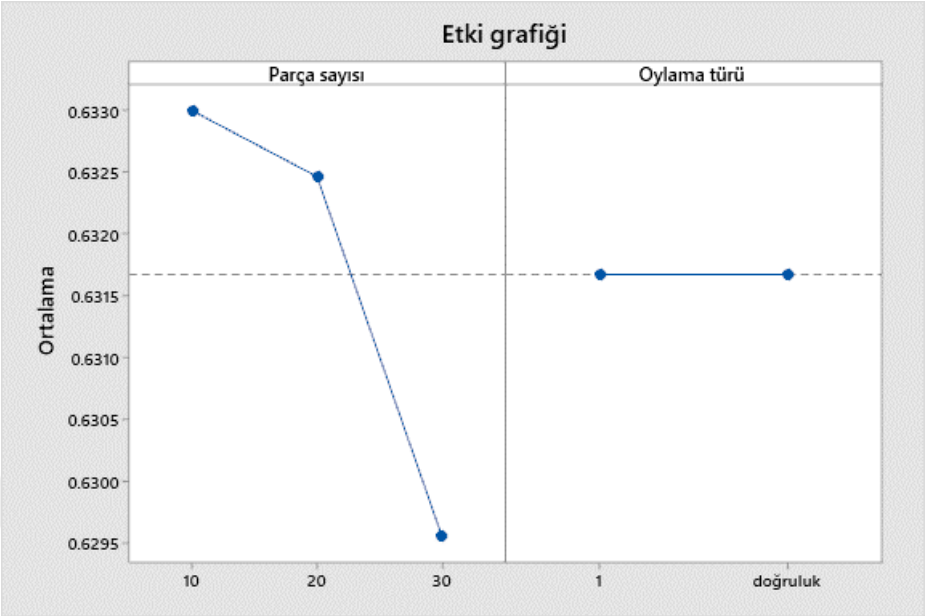
Üretim veri kümesi için Şekil 4.37'de verilen standartlaştırılmış pareto grafiği incelendiğinde parça sayısı parametresinin etkili, oylama türü ve parça sayısı-oylama türü ikili etkileşiminin etkili olmadığı görülmektedir.



Şekil 4.37 Üretim veri kümesi standartlaştırılmış etkiler pareto grafiği

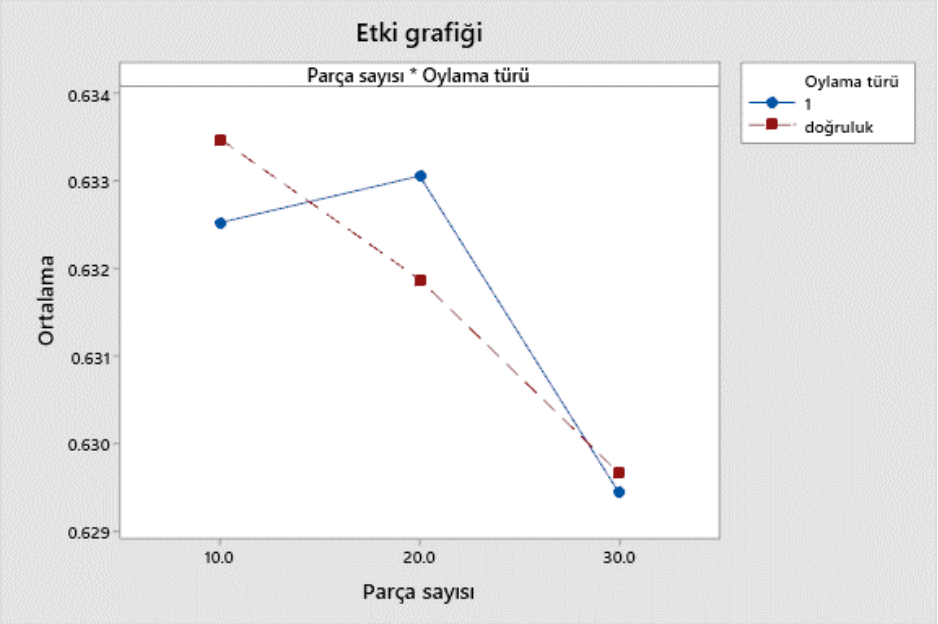
Şekil 4.38'te üretim veri kümesi için ana etkiler grafiği verilmiştir. Şekilden de görüldüğü üzere parça sayısı parametresinin doğrusu belirli bir eğime sahiptir dolayısıyla etkili bir parametredir, ancak oylama türü parametresinin doğrusu neredeyse x-eksenine paraleldir, dolayısıyla etkisiz bir parametredir. Ana etki grafiğine göre sınıflandırma doğruluğu etkinlik ölçütünde parça sayısı parametresinde etkin düzey 10 olarak belirlenmiştir,

oylama türü parametresinin aldığı değerin sınıflandırma değeri üzerinde etkisi olmadığı için her değeri alabilir.



Şekil 4.38 Üretim veri kümesi ana etkiler grafiği

Şekil 4.39'te üretim veri kümesi için ikili etkileşim grafiği verilmiştir. Şekilden de görüldüğü üzere etkileşim doğruları birbirini kesmektedir, dolayısıyla parça sayısı ve oylama türü parametrelerinin farklı düzeylerinin birbirlerini etkilediği söylenebilir. İkili etkileşim grafiğinden sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkin parametre değerlerinin parça sayısı için 10 (ikili etkileşimler, ana etki değerini doğrulamıştır) ve oylama türü parametresi için doğruluk olduğu görülmektedir.



Şekil 4.39 Üretim veri kümesi ikili etkileşim grafiği

Sonuç olarak üretim veri kümesi için etkin parametre değerleri:

- Parça sayısı: 10
- Oylama türü: doğruluk

olarak belirlenmiştir.

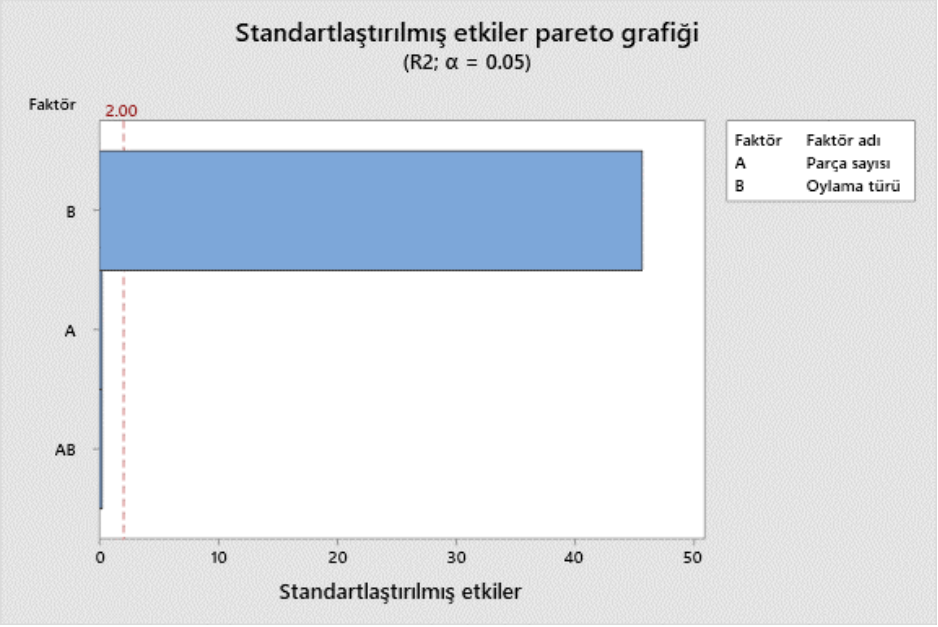
İkinci El Araba Veri Kümesi Deney Tasarımı

Tablo 4.30'de sunulan ikinci el araba veri kümesi için varyans analizi sonuçları incelendiğinde, sadece oylama türü parametresinin etkili olduğu görülmektedir (p değeri<0.05). Ancak, parça sayısı parametresinin ve parça sayısı-oylama türü ikili etkileşiminin p değeri α=0.05'ten yüksek çıkmıştır. Dolayısıyla parça sayısı parametresi ve parça sayısı-oylama türü ikili etkileşimi R-kare etkinlik ölçütü üzerinde etkili değildir.

Tablo 4.30 İkinci el araba veri kümesi varyans analizi

Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	5	0.113343	0.022669	417.47	0.000
Lineer	3	0.113320	0.037773	695.64	0.000
Parça sayısı	2	0.000023	0.000012	0.21	0.807
Oylama türü	1	0.113297	0.113297	2086.48	0.000
2-li Etkileşimler	2	0.000023	0.000012	0.21	0.809
Parça sayısı*Oylama türü	2	0.000023	0.000012	0.21	0.809
Hata	54	0.002932	0.000054		
Toplam	59	0.116275			

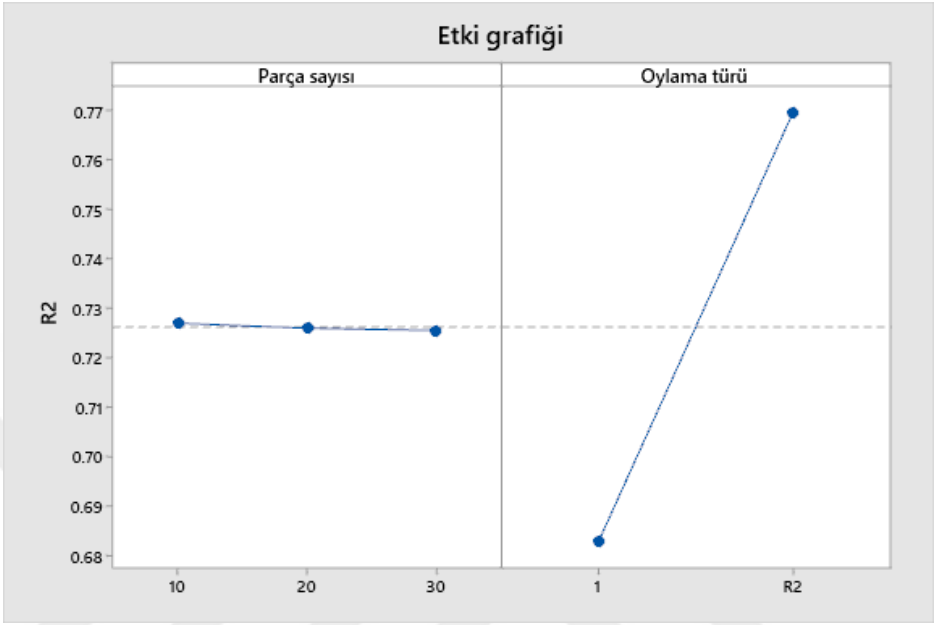
Şekil 4.40'da gösterilen standartlaştırılmış etkiler pareto grafiği varyans analizi sonuçlarını doğrulamaktadır. Oylama türü parametresi R-kare üzerinde etkili çıkmıştır. Parça sayısı parametresi ile parça sayısı-oylama türü ikili etkileşimi etkili bir parametre değildir.



Şekil 4.40 İkinci el araba veri kümesi standartlaştırılmış etkiler pareto grafiği

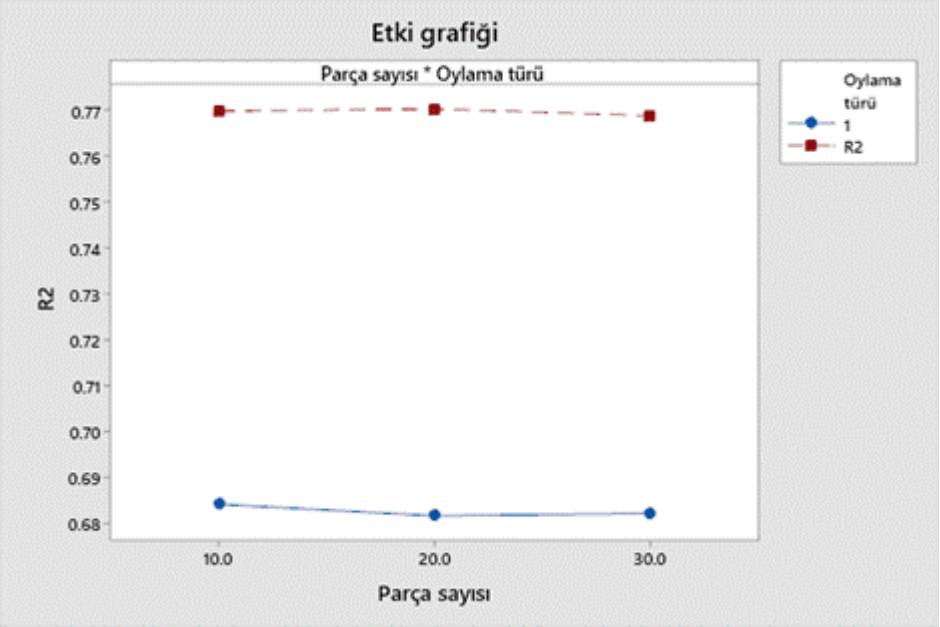
Şekil 4.41'da ikinci el araba veri kümesi için ana etki grafikleri verilmiştir. Ana etki grafikleri incelendiğinde oylama türü parametresine ait doğrunun oldukça yüksek bir eğime sahip olduğu dolayısıyla aldığı farklı parametre değerlerinin R-kare üzerinde oldukça etkili olduğu ve en yüksek R-kare değerini R-kare düzeyinin verdiği

görülmektedir. Parça sayısı parametresine ait etki doğrusunun ise eğimi oldukça azdır, dolayısıyla R-kare üzerinde parametrenin düzeylerinin etkisi de yok denilecek kadar azdır.



Şekil 4.41 İkinci el araba veri kümesi ana etkiler grafiği

Şekil 4.42'da ikinci el araba veri kümesi için ikili etkileşim grafiği sunulmuştur. İkili etkileşim grafiği incelendiğinde doğruların birbirini kesmediği neredeyse birbirlerine paralel olduğu görülmektedir. Dolayısıyla parça sayısı ile oylama türü parametreleri arasında ikili etkileşim yoktur.



Şekil 4.42 İkinci el araba veri kümesi ikili etkileşim grafiği

Sonuç olarak ikinci el araba veri kümesi için etkin parametre değerleri:

- Parça sayısı: 20
- Oylama türü: R-kare olarak belirlenmiştir.

YSA-Karar Ağacı Algoritması Deney Tasarımı

YSA-Karar Ağacı algoritması, gizli katman ve iterasyon olmak üzere iki parametreye sahiptir. Gizli katman sayısı parametresi yapay sinir ağında kaç gizli katman ve gizli katmanlarda kaç/kaçar proses elemanı (nöron) olduğunu ifade eden bir parametredir. G, GO, G-G/2 ve GO-GO/2 olmak üzere 4 düzeyi deney tasarımında ele alınmıştır. Bu düzeylerden G değeri, yapay sinir ağının tek gizli katman ve gizli katmanda ele alınan problemin girdi niteliği kadar gizli katman proses elemanına sahip olmasını; GO değeri, YSA'nın tek gizli katman ve gizli katmanda ele alınan problemin girdi sayısı ile sınıf (çıktı) sayısının geometrik ortalaması ($\sqrt{girdi\ sayısı * sınıf\ sayısı}$) kadar gizli katman proses elemanına sahip olmasını; G-G/2 değeri, yapay sinir ağının iki gizli katman ve ilk gizli katmanda ele alınan problemdeki girdi niteliği kadar, ikinci gizli katmanda girdi niteliğinin yarısı kadar proses elemanı içermesini; GO-GO/2 değeri ise YSA'nın iki gizli katman, ilk gizli katmanda ele alınan problemdeki girdi ve sınıf sayılarının geometrik ortalaması, ikinci gizli katmanda girdi sayısı ve sınıf sayısının geometrik ortalamasının

yarısı kadar proses elemanına sahip olmasını ifade etmektedir. İterasyon sayısı ise yapay sinir ağının kaç iterasyon çalışacağını gösteren parametredir ve deney tasarımında 100 ve 250 olmak üzere 2 düzeyi ele alınmıştır. Deney tasarımı parametreleri ve parametre düzeyleri gerçek hayat veri kümeleri ile aynı Tablo 4.31'de verildiği gibi belirlenmiştir.

Tablo 4.31 YSA-KA algoritması parametreleri ve parametre düzeyleri

Parametre	**Değerler**
Gizli katman	G GO G-G/2 GO-GO/2
İterasyon	100 250

Deney tasarımında 4x2=8 deney bulunmaktadır. Her bir veri kümesi ve tasarımın her bir deneyi için algoritma 10 kez çalıştırılarak bu 10 çalıştırmanın üretim gerçek hayat veri kümesi üzerinde ortalama sınıflandırma doğruluk oranı, ikinci el araba gerçek hayat veri kümesinde ise R-kare değeri deney tasarımında etkinlik ölçütü olarak ele alınmıştır.

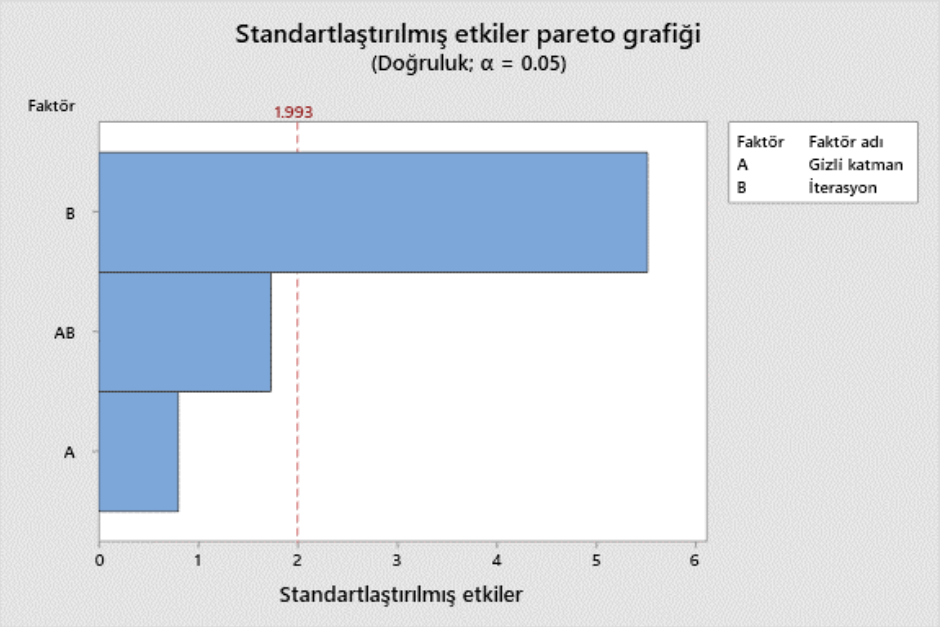
Üretim Veri Kümesi Deney Tasarımı

Üretim gerçek hayat veri kümesi için varyans analizi sonuçları Tablo 4.32'de verilmiştir. Anlamlılık düzeyi $\alpha=0.05$ olarak dikkate alındığında sadece iterasyon parametresinin sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkili olduğu (p değerleri <0.05), gizli katman parametresi ve gizli katman*iterasyon etkileşiminin etkili olmadığı görülmektedir (p değerleri>0.05).

Tablo 4.32 Üretim veri kümesi varyans analizi

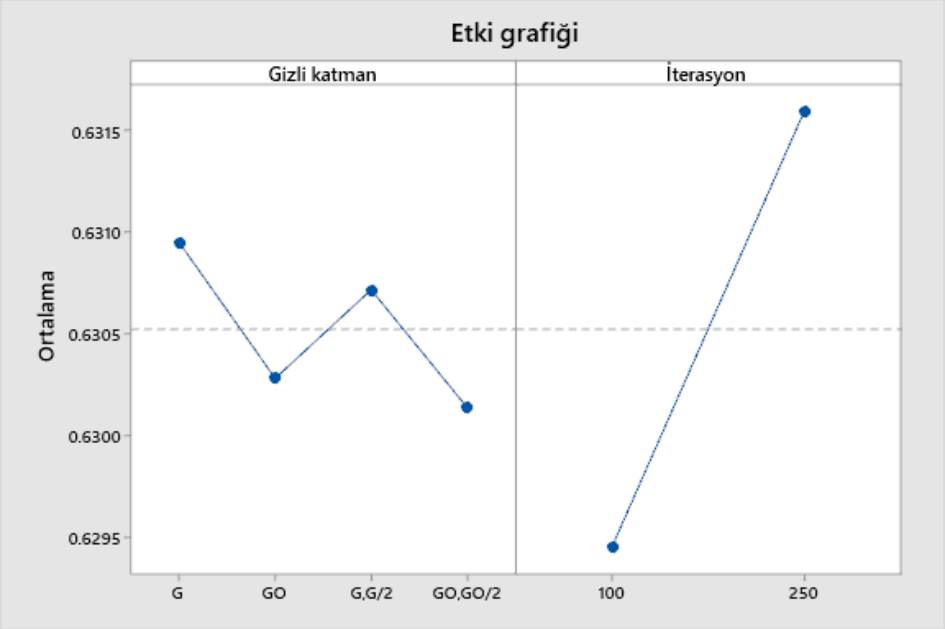
Kaynak	Serbestlik Derecesi	Ayarlanmış SS	Ayarlanmış MS	F-Değeri	P-Değeri
Model	7	0.000121	0.000017	5.73	0.000
Lineer	4	0.000101	0.000025	8.32	0.000
Gizli katman	3	0.000009	0.000003	0.94	0.427
İterasyon	1	0.000092	0.000092	30.47	0.000
2-li Etkileşimler	3	0.000021	0.000007	2.27	0.088
Gizli katman*İterasyon	3	0.000021	0.000007	2.27	0.088
Hata	72	0.000218	0.000003		
Toplam	79	0.000339			

Üretim veri kümesi için Şekil 4.43 Üretim veri kümesi standartlaştırılmış etkiler pareto grafiği'de verilen standartlaştırılmış pareto grafiği incelendiğinde iterasyon parametresinin etkili, gizli katman ve gizli katman-iterasyon ikili etkileşiminin etkili olmadığı görülmektedir.



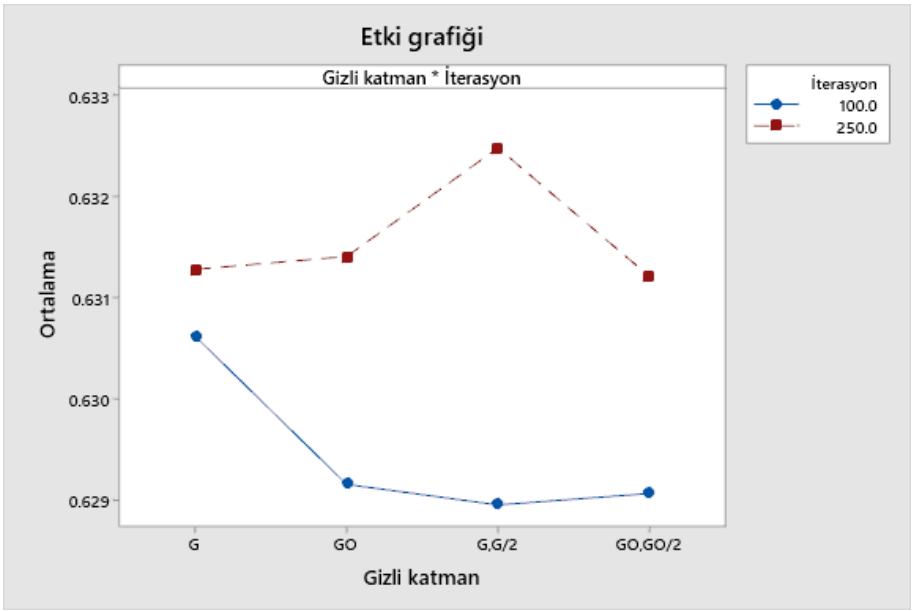
Şekil 4.43 Üretim veri kümesi standartlaştırılmış etkiler pareto grafiği

Şekil 4.44'de üretim veri kümesi için ana etkiler grafiği verilmiştir. Şekilden de görüldüğü gizli katman parametresinin doğrusu da iterasyon parametresinin doğrusu da eğime sahiptir. Ana etki grafiğine göre sınıflandırma doğruluğu etkinlik ölçütünde gizli katman parametresinde etkin düzey G, iterasyon parametresinde etkin düzey olarak 250 belirlenmiştir.



Şekil 4.44 Üretim veri kümesi ana etkiler grafiği

Şekil 4.45'te üretim veri kümesi için ikili etkileşim grafiği verilmiştir İkili etkileşim grafiğinden sınıflandırma doğruluk oranı etkinlik ölçütü üzerinde etkin parametre değerlerinin iterasyon için 250 ve gizli katman parametresi için G-G/2 olduğu görülmektedir.



Şekil 4.45 Üretim veri kümesi ikili etkileşim grafiği

Sonuç olarak üretim veri kümesi için etkin parametre değerleri:

- Gizli katman: G-G/2
- İterasyon: 250 olarak belirlenmiştir.

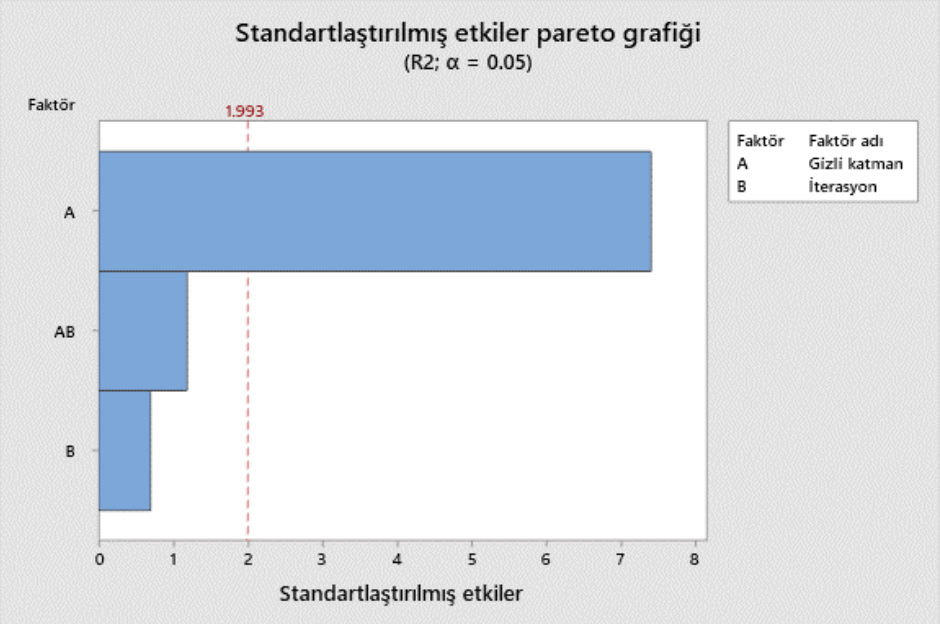
İkinci El Araba Veri Kümesi Deney Tasarımı

Tablo 4.33'de sunulan ikinci el araba veri kümesi için varyans analizi sonuçları incelendiğinde, sadece gizli katman parametresinin etkili olduğu görülmektedir (p değeri<0.05). İterasyon ve gizli katman*iterasyon ikili etkileşiminin p değeri α=0.05'ten yüksek çıkmıştır. Dolayısıyla iterasyon parametresi ve gizli katman*iterasyon ikili etkileşimi R-kare etkinlik ölçütü üzerinde etkili değildir.

Tablo 4.33 İkinci el araba veri kümesi varyans analizi

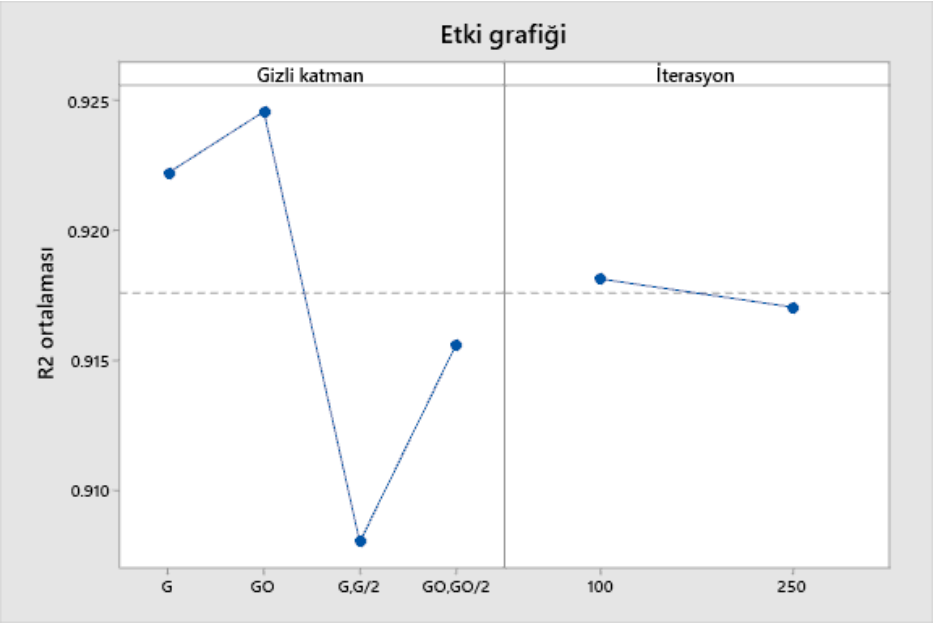
Kaynak	**Serbestlik Derecesi**	**Ayarlanmış SS**	**Ayarlanmış MS**	**F-Değeri**	**P-Değeri**
Model	7	0.003551	0.000507	10.41	0.000
Lineer	4	0.003340	0.000835	17.13	0.000
Gizli katman	3	0.003317	0.001106	22.68	**0.000**
İterasyon	1	0.000024	0.000024	0.48	0.489
2-li Etkileşimler	3	0.000211	0.000070	1.44	0.238
Gizli katman*İterasyon	3	0.000211	0.000070	1.44	0.238
Hata	72	0.003509	0.000049		
Toplam	79	0.007060			

Şekil 4.46'da gösterilen standartlaştırılmış etkiler pareto grafiği varyans analizi sonuçlarını doğrulamaktadır. Gizli katman parametresi R-kare üzerinde etkili çıkmıştır. İterasyon parametresi ile gizli katman-iterasyon ikili etkileşimi etkili parametre değildir.



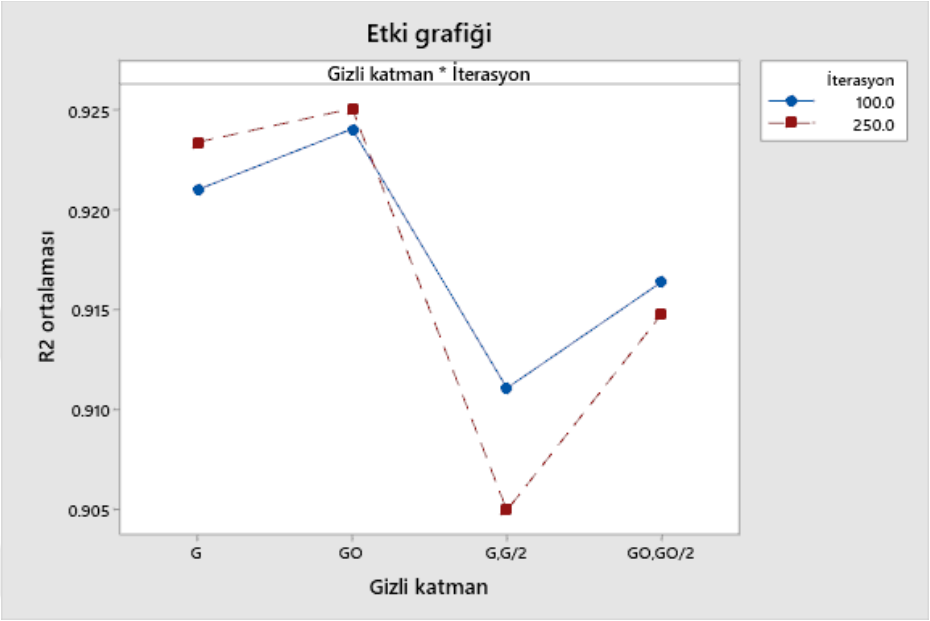
Şekil 4.46 İkinci el araba veri kümesi standartlaştırılmış etkiler pareto grafiği

Şekil 4.47'da ikinci el araba veri kümesi için ana etki grafikleri verilmiştir. Ana etki grafikleri incelendiğinde gizli katman parametresine ait doğrunun oldukça yüksek bir eğime sahip olduğu dolayısıyla aldığı farklı parametre değerlerinin RMSE üzerinde oldukça etkili olduğu ve en düşük RMSE değerini GO düzeyinin verdiği görülmektedir. İterasyon parametresine ait etki doğrusunun ise eğimi daha azdır, en düşük RMSE veren düzeyi ise 100'dür.



Şekil 4.47 İkinci el araba veri kümesi ana etkiler grafiği

Şekil 4.48'da ikinci el araba veri kümesi için ikili etkileşim grafiği sunulmuştur. İkili etkileşim grafiği incelendiğinde doğruların birbirini kestiği, dolayısıyla parametrelerin farklı düzeylerinin birbirlerini etkileyebileceği görülmektedir. İkili etkileşim grafiğine göre gizli katman ve iterasyon parametrelerinin R-kare etkinlik ölçütü üzerindeki etkin değerleri sırasıyla GO ve 250'dir.



Şekil 4.48 İkinci el araba veri kümesi ikili etkileşim grafiği

Sonuç olarak ikinci el araba veri kümesi için etkin parametre değerleri:

- Gizli katman: GO
- İterasyon: 250 olarak belirlenmiştir.

4.2.4. SH ve YSA-KA Algoritmaları ile Elde Edilen Sonuçlar

Deney tasarımı sonucunda elde edilen en iyi parametreler ile gerçek hayat verileri 30 kez çalıştırılmıştır. Sonuçlar Tablo 4.34'deki gibidir. Üretim veri kümesinde SH, YSA-KA algoritmasını çok az bir farkla geçip, daha yüksek doğruluk vermiştir. Araba veri kümesinde ise R^2, RMSE değerlendirme kriterlerine göre YSA-KA, SH algoritmasından daha iyi sonuç vermiştir. Süre olarak değerlendirildiğinde YSA-KA üretim veri kümesi için 72.209 saniye sürmüştür, SH ise 126.721 saniye sürmüştür. YSA-KA ikinci el araba veri kümesi için 12.-90 saniye sürerken, SH ise 18.903 saniye sürmüştür. Bu sonuçlara göre YSA-KA algoritmasının SH'a göre daha hızlı bir algoritma olduğu söylenebilir.

Tablo 4.34 Gerçek hayat verileri üzerinde yeni algoritma sonuçları

	SH			YSA-KA		
Veri kümesi	Doğruluk	S. sapma	Süre(s)	Doğruluk	S. sapma	Süre(s)
Üretim	**0.633**	**0.002**	126.721	0.632	**0.002**	**72.209**
Araba (R^2)	0.770	0.008	18.903	**0.924**	**0.006**	**12.090**
Araba (RMSE)	29076	164.357	18.903	**16630**	**157.708**	**12.090**

İkinci el araba veri kümesinde YSA-KA algoritması tahminleri ile gerçek araba fiyatları arasındaki fark ortalama olarak %1.1 oranında çıkmıştır.

4.2.5. SH ve YSA-KA Algoritmaları Performans Karşılaştırmaları

Deney tasarımı sonucu elde edilen en iyi parametre değerleriyle çalıştırılan algoritmalar sadece kendi aralarında değil ayrıca Spark içerisinde bulunan makine öğrenme algoritmalarıyla da karşılaştırılmıştır. Spark'taki algoritmalar; sınıflandırma için lojistik regresyon (LR), karar ağacı (KA), rastgele orman (RO), yapay sinir ağı (YSA) ve doğrusal destek vektör makinesi (DDVM) algoritmalarıdır. Fonksiyon tahminleme için ise doğrusal regresyon (DR), karar ağacı (KA), rastgele orman (RO) ve izotonik regresyon (IR) algoritmalarıdır. Karşılaştırmalar üretim verisi için Tablo 4.35'da ikinci el araba verisi için Tablo 4.36'da gösterilmiştir. Başarı kriteri olarak sınıflandırma türü olan üretim verisi için doğruluk oranları, regresyon türü olan ikinci el araba verisi için ise R^2, RMSE ve MAE kullanılmıştır. Tablo 4.35'a bakıldığında üretim veri kümesi için en iyi sonucu KA vermiştir. SH üçüncü, YSA-KA ise dördüncü sıradadır.

Tablo 4.35 Üretim veri kümesi algoritma sonuçları

Veri kümesi	**SH**	**YSA-KA**	**LR**	**KA**	**RO**	**YSA**
Üretim	0.633	0.632	0.626	**0.638**	0.635	0.627

Tablo 4.36'a bakıldığında ikinci el araba veri kümesi için en iyi sonucu 16630 RMSE ve 0.924 R2 değeri ile YSA-KA hibrit algoritması vermiştir. SH ise 29076 RMSE ve 0.770 R2 değeri ile dördüncü sıradadır.

Tablo 4.36 İkinci el araba veri kümesi algoritmaların RMSE sonuçları

Veri kümesi	SH	YSA-KA	DR	KA	RO	IR
Araba (RMSE)	29076	**16630**	32960	18763	19531	50242
Araba (R^2)	0.770	**0.924**	0.703	0.904	0.895	0.314

4.2.6. İstatistiksel Analiz

Tez çalışması kapsamında önerilen SH ve YSA-KA algoritmaları ve Spark'ta bulunan algoritmaların gerçek hayat veri kümeleri üzerinde elde ettikleri sonuçlar arasında istatistiksel olarak anlamlı bir farklılık olup olmadığını belirlemek için istatistiksel analizler gerçekleştirilmiştir. Her bir algoritmanın, her bir veri kümesinde 30 kez çalıştırılmasından dolayı merkezi limit teoremi gereğince elde edilen sonuçların normal dağılıma uyduğu varsayılmıştır. Böylece istatistiksel analiz aracı olarak eşleştirilmiş-t testi kullanılmıştır.

Üretim veri kümesi için SH ve YSA-KA algoritmaları ile diğer algoritmalar arasındaki farkların istatistiksel olarak anlamlı olup olmadığını anlamak için yapılan test sonuçları Tablo 4.37'de gösterilmiştir. %95 güven aralığında elde edilen p değerleri incelendiğinde sadece 0.081 değeriyle SH/YSA-KA algoritmaları arasında anlamlı bir fark ortaya çıkmamıştır. Diğer bütün sonuçlardaki farklar anlamlıdır. Bu sonuçlara göre KA birinci, RO ikinci, YSA-KA ve SH algoritmaları birlikte üçüncü sıradadır.

Tablo 4.37 Üretim veri kümesi için algoritmaların istatistiksel karşılaştırılması

Algoritmalar	**Üretim**
YSA-KA/SH	0.081
YSA-KA/LR	**0.000**
YSA-KA/KA	**0.000**
YSA-KA/RO	**0.000**
YSA-KA/YSA	**0.000**
SH/LR	**0.000**
SH/KA	**0.000**
SH/RO	**0.000**
SH/YSA	**0.000**

Tablo 4.38'e ikinci el araba veri kümesi üzerinde RMSE değerleri üzerinde gerçekleştirilen eşleştirilmiş-t testi p değerleri gösterilmiştir. Tablo incelendiğinde, ikinci el araba veri kümesi için bütün p değerleri 0.05'in altında çıkmıştır. Bu sonuca göre bütün farklar anlamlıdır. YSA-KA algoritması birinci sırada, KA ikinci, RO üçüncü, SH ise dördüncü sıradadır.

Tablo 4.38 İkinci al araba veri kümesi için algoritmaların istatistiksel karşılaştırılması

Algoritmalar	**R^2**	**RMSE**
YSA-KA/SH	**0.000**	**0.000**
YSA-KA/DR	**0.000**	**0.000**
YSA-KA/KA	**0.000**	**0.000**
YSA-KA/RO	**0.000**	**0.000**
YSA-KA/IR	**0.000**	**0.000**
SH/DR	**0.000**	**0.000**
SH/KA	**0.000**	**0.000**
SH/RO	**0.000**	**0.000**
SH/IR	**0.000**	**0.000**

Bütün bu sonuçlar değerlendirildiğinde, YSA-KA algoritmasının SH algoritması ile üretim veri kümesi üzerinde sınıflandırma problemi için orta sıralarda olduğu, ikinci el araba veri kümesinde ise YSA-KA algoritmasının ilk, SH algoritmasının ise dördüncü sırada olduğu görüşmüştür.

5. SONUÇLAR

Bu tez çalışmasında büyük verilerin analizinde kullanılmak üzere iki yeni algoritma önerilmiştir. Bu algoritmalar Spark hibrit ve yapay sinir ağı – karar ağacı hibrit algoritmalarıdır. Algoritmaların performanslarını ortaya koymak için tez çalışmasında kapsamlı deneysel çalışmalar gerçekleştirilmiştir. Bu çalışmalar sırasında 6 sınıflandırma literatür veri kümesi ve iki gerçek hayat problemi üzerinde önerilen algoritmaların performansı Spark'ta mevcut algoritmalarla ve özellikle son yıllarda ortaya konan literatür çalışmaları ile karşılaştırılmalı olarak sunulmuştur. Elde edilen sonuçların istatistiksel olarak anlamlı olup olmadığını belirlemek için istatistiksel testler gerçekleştirilmiştir.

SH algoritması literatürdeki çalışmalarla karşılaştırıldığında Pokerhand veri kümesinde alt sıralarda, Susy, Hepmass ve Census veri kümelerinde en iyi sırada, Higgs veri kümesinde ikinci sırada, Wisdm veri kümesinde ise üçüncü sırada çıkmıştır. YSA-KA algoritması literatürdeki çalışmalarla karşılaştırıldığında Pokerhand, Susy, Higgs, Hepmass ve Wisdm veri kümelerinde birinci sırada, Census veri kümesinde ise ikinci sırada çıkmıştır. SH ve YSA-KA algoritmaları birbirleriyle, literatürdeki çalışmalarla ve Spark'taki algoritmalarla karşılaştırıldığında ise genel olarak YSA-KA birinci, SH ise ikinci veya üçüncü sırada çıkmıştır.

Gerçek hayat verileri üzerinde SH ve YSA-KA algoritmaları değerlendirildiğinde YSA-KA algoritmasının iyi bir seviyede olduğu, SH algoritmasının ise karşılaştırılan algoritmalara göre orta sırada yer aldığı görülmüştür.

Bütün deneysel çalışmalar ve sonuçlar analiz edildiğinde YSA-KA algoritmasının büyük veri alanında oldukça iyi bir sınıflandırma/fonksiyon tahminleme algoritması olduğu görülmüştür. SH algoritmasının ise en iyi olmasa da iyi sayılabilecek bir algoritma olduğu

ortaya konmuştur. Ayrıca SH algoritması genel olarak kesikli niteliğe sahip veri kümelerinde daha iyi sonuçlar vermiştir.

Gelecek çalışmalarda, SH algoritması için algoritmanın etkinliğini artırmaya yönelik olarak deney tasarımlarına daha farklı parametreler eklenebilir ve parametrelerin alternatif değerleri çoğaltılabilir. Buna ek olarak algoritma içerisindeki veri parçası üstünde çalışan algoritmalar verimliliklerine göre incelenip ekleme-çıkarmalar yapılabilir. YSA-KA algoritmasında ise daha farklı topolojiler kullanılabilir, yapay sinir ağı daha çok derinleştirilebilir. Görüntü işleme, metin madenciliği, ses analizi gibi yapay zekanın ve derin öğrenmenin konusu olan farklı problem türleri için özelleştirilebilir ve kullanılabilir.

KAYNAKÇA

1. Tole, A.A., 2013. Big data challenges. **Database Systems Journal**, **4** (3): 31-40.
2. García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J.M., vd., 2016. Big data preprocessing: methods and prospects. **Big Data Analytics**, **1** (1): 9.
3. Aziz, K., Zaidouni, D., Bellafkih, M., 2018. Real-time data analysis using spark and hadoop, *2018 4th International Conference on Optimization and Applications (ICOA)*, s. 1-6.
4. Hazarika, A.V., Ram, G.J.S.R., Jain, E., 2017. Performance comparision of hadoop and spark engine, *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, s. 671-674.
5. Sleeman IV, W.C., Krawczyk, B., 2021. Multi-class imbalanced big data classification on spark. **Knowledge-Based Systems**, **212** : 106598.
6. Ahmed, H., Younis, E.M., Ali, A.A., 2020. Predicting diabetes using distributed machine learning based on apache spark, s. 44-49.
7. Maillo, J., García, S., Luengo, J., Herrera, F., vd., 2020. Fast and scalable approaches to accelerate the fuzzy k-nearest neighbors classifier for big data. **IEEE Transactions on Fuzzy Systems**, **28** (5): 874-886.
8. Lenz, O.U., Peralta, D., Cornelis, C., 2020. Scalable approximate frnn-owa classification. **IEEE Transactions on Fuzzy Systems**, **28** (5): 929-938.
9. Oo, M.C.M., Thein, T., 2019. An efficient predictive analytics system for high dimensional big data. **Journal of King Saud University - Computer and Information Sciences**, .
10. Za'in, C., Pratama, M., Pardede, E., 2019. Evolving large-scale data stream analytics based on scalable panfis. **Knowledge-Based Systems**, **166** : 186-197.
11. Weinberg, A.I., Last, M., 2019. Selecting a representative decision tree from an ensemble of decision-tree models for fast big data classification. **Journal of Big Data**, **6** (1): 23.
12. Wang, F., Wang, Q., Nie, F., Yu, W., vd., 2018. Efficient tree classifiers for large scale datasets. **Neurocomputing**, **284** : 70-79.
13. Donkal, G., Verma, G.K., 2018. A multimodal fusion based framework to reinforce ids for securing big data environment using spark. **Journal of Information Security and Applications**, **43** : 1-11.

14. Dahiya, P., Srivastava, D.K., 2018. Network intrusion detection in big dataset using spark. **Procedia Computer Science**, **132** : 253-262.
15. Maillo, J., Luengo, J., García, S., Herrera, F., vd., 2018. A preliminary study on hybrid spill-tree fuzzy k-nearest neighbors for big data classification, *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, s. 1-8.
16. Inoue, M., Inoue, S., Nishida, T., 2018. Deep recurrent neural network for mobile human activity recognition with high throughput. **Artificial Life and Robotics**, **23** (2): 173-185.
17. Ignatov, A., 2018. Real-time human activity recognition from accelerometer data using convolutional neural networks. **Applied Soft Computing**, **62** : 915-922.
18. Chakrabarty, N., Biswas, S., 2018. A statistical approach to adult census ıncome level prediction, *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, s. 207-212.
19. Maillo, J., Luengo, J., García, S., Herrera, F., vd., 2017. Exact fuzzy k-nearest neighbor classification for big datasets, *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, s. 1-6.
20. Weinberg, A.I., Last, M., 2017. Interpretable decision-tree induction in a big data parallel framework. **International Journal of Applied Mathematics and Computer Science**, **27** (4): 737-748.
21. Maillo, J., Ramírez, S., Triguero, I., Herrera, F., 2017. KNN-ıs: an ıterative spark-based design of the k-nearest neighbors classifier for big data. **Knowledge-Based Systems**, **117** : 3-15.
22. García-Gil, D., Ramírez-Gallego, S., García, S., Herrera, F., 2017. A comparison on scalability for batch big data processing on apache spark and apache flink. **Big Data Analytics**, **2** (1): 1-11.
23. Bechini, A., Marcelloni, F., Segatori, A., 2016. A mapreduce solution for associative classification of big data. **Information Sciences**, **332** : 33-55.
24. Alsheikh, M.A., Niyato, D., Lin, S., Tan, H., vd., 2016. Mobile big data analytics using deep learning and apache spark. **IEEE Network**, **30** (3): 22-29.
25. Ronao, C.A., Cho, S.-B., 2016. Human activity recognition with smartphone sensors using deep learning neural networks. **Expert Systems with Applications**, **59** : 235-244.

26. Ronao, C.A., Cho, S.-B., 2017. Recognizing human activities from smartphone sensors using hierarchical continuous hidden markov models. **International Journal of Distributed Sensor Networks**, **13** (1): 1550147716683687.
27. Silva, D.A.N.S., Souza, L.C., Motta, G.H.M.B., 2016. An instance selection method for large datasets based on markov geometric diffusion. **Data & Knowledge Engineering**, **101** : 24-41.
28. Fish, B., Kun, J., Lelkes, Á.D., 2016. A confidence-based approach for balancing fairness and accuracy, s. 144-152.
29. Triguero, I., Peralta, D., Bacardit, J., García, S., vd., 2015. MRPR: a mapreduce solution for prototype reduction in big data classification. **Neurocomputing**, **150** : 331-345.
30. Shashidhara, B.M., Jain, S., Rao, V.D., Patil, N., vd., 2015. Evaluation of machine learning frameworks on bank marketing and higgs datasets, *2015 Second International Conference on Advances in Computing and Communication Engineering*, s. 551-555.
31. Oza, N.C., 2005. Online bagging and boosting, *2005 IEEE International Conference on Systems, Man and Cybernetics*, s. 2340-2345 Vol. 3.
32. del Río, S., López, V., Benítez, J.M., Herrera, F., 2014. On the use of mapreduce for imbalanced big data using random forest. **Information Sciences**, **285** : 112-137.
33. Anguita, D., Ghio, A., Oneto, L., Parra, X., vd., 2013. A public domain dataset for human activity recognition using smartphones, .
34. Kamiran, F., Calders, T., Pechenizkiy, M., 2010. Discrimination aware decision tree learning, *2010 IEEE International Conference on Data Mining*, s. 869-874.
35. HimaBindu, G., Raghu Kumar, Ch., Hemanand, Ch., Rama Krishna, N., 2020. Hybrid clustering algorithm to process big data using firefly optimization mechanism. **Materials Today: Proceedings**, .
36. Tao, Q., Gu, C., Wang, Z., Jiang, D., 2020. An intelligent clustering algorithm for high-dimensional multiview data in big data applications. **Neurocomputing**, **393** : 234-244.
37. Lu, W., 2020. Improved k-means clustering algorithm for big data mining under hadoop parallel framework. **Journal of Grid Computing**, **18** (2): 239-250.

38. Wu, Y., He, Z., Lin, H., Zheng, Y., vd., 2019. A fast projection-based algorithm for clustering big data. **Interdisciplinary Sciences: Computational Life Sciences**, **11** (3): 360-366.
39. Hai, M., Zhang, Y., Li, H., 2018. A performance comparison of big data processing platform based on parallel clustering algorithms. **Procedia Computer Science**, **139** : 127-135.
40. Cui, X., Zhu, P., Yang, X., Li, K., vd., 2014. Optimized big data k-means clustering using mapreduce. **The Journal of Supercomputing**, **70** (3): 1249-1259.
41. Zheng, L., Li, L., Hong, W., Li, T., 2013. PENETRATE: personalized news recommendation using ensemble hierarchical clustering. **Expert Systems with Applications**, **40** (6): 2127-2136.
42. Shang, H., Lu, D., Zhou, Q., 2020. Early warning of enterprise finance risk of big data mining in internet of things based on fuzzy association rules. **Neural Computing and Applications**, .
43. Chan, M.S., Jamieson, K.H., Albarracin, D., 2020. Prospective associations of regional social media messages with attitudes and actual vaccination: a big data and survey study of the influenza vaccine in the united states. **Vaccine**, **38** (40): 6236-6247.
44. Chen, Y., Li, F., Fan, J., 2015. Mining association rules in big data with ngep. **Cluster Computing**, **18** (2): 577-585.
45. Zhang, F., Liu, M., Gui, F., Shen, W., vd., 2015. A distributed frequent itemset mining algorithm using spark for big data analytics. **Cluster Computing**, **18** (4): 1493-1501.
46. Moens, S., Aksehirli, E., Goethals, B., 2013. Frequent ıtemset mining for big data, *2013 IEEE International Conference on Big Data*, s. 111-118.
47. Syed, D., Refaat, S.S., Abu-Rub, H., 2020. Performance evaluation of distributed machine learning for load forecasting in smart grids, s. 1-6.
48. Xu, Y., Liu, H., Long, Z., 2020. A distributed computing framework for wind speed big data forecasting on apache spark. **Sustainable Energy Technologies and Assessments**, **37** : 100582.
49. Talavera-Llames, R., Pérez-Chacón, R., Troncoso, A., Martínez-Álvarez, F., 2018. Big data time series forecasting based on nearest neighbours distributed computing with spark. **Knowledge-Based Systems**, **161** : 12-25.

50. Galicia, A., Torres, J.F., Martínez-Álvarez, F., Troncoso, A., 2018. A novel spark-based multi-step forecasting algorithm for big data time series. **Information Sciences**, **467** : 800-818.
51. Sohangir, S., Wang, D., Pomeranets, A., Khoshgoftaar, T.M., 2018. Big data: deep learning for financial sentiment analysis. **Journal of Big Data**, **5** : 3.
52. El Alaoui, I., Gahi, Y., Messoussi, R., Chaabi, Y., vd., 2018. A novel adaptable approach for sentiment analysis on big social data. **Journal of Big Data**, **5** : 12.
53. Nodarakis, N., Sioutas, S., Tsakalidis, A.K., Tzimas, G., 2016. Large scale sentiment analysis on twitter with spark., s. 1-8.
54. Shoro, A.G., Soomro, T.R., 2015. Big data analysis: apache spark perspective. **Global Journal of Computer Science and Technology**, .
55. Aggarwal, C.C., Wang, H., 2011. Text mining in social networks, s. 353-378.
56. Ritter, A., Clark, S., Mausam, Etzioni, O., 2011. Named entity recognition in tweets: an experimental study, Stroudsburg, PA, USA, s. 1524-1534.
57. Ruihong, Z., Zhihua, H., 2020. Comparative research on active learning of big aata based on mapreduce and spark. **Microprocessors and Microsystems**, : 103425.
58. Yin, L., Qin, L., Jiang, Z., Xu, X., 2020. A fast parallel attribute reduction algorithm using apache spark. **Knowledge-Based Systems**, : 106582.
59. Expósito, R.R., González-Domínguez, J., Touriño, J., 2020. SMusket: spark-based dna error correction on distributed-memory systems. **Future Generation Computer Systems**, **111** : 698-713.
60. Karim, L., Boulmakoul, A., Mandar, M., Lbath, A., vd., 2020. A new pedestrians' intuitionistic fuzzy risk exposure indicator and big data trajectories analytics on spark-hadoop ecosystem. **Procedia Computer Science**, **170** : 137-144.
61. Islam, M.T., Srirama, S.N., Karunasekera, S., Buyya, R., 2020. Cost-efficient dynamic scheduling of big data applications in apache spark on cloud. **Journal of Systems and Software**, **162** : 110515.
62. Manogaran, G., Lopez, D., 2018. Spatial cumulative sum algorithm with big data analytics for climate change detection. **Computers & Electrical Engineering**, **65** : 207-221.
63. García-Gil, D., Ramírez-Gallego, S., García, S., Herrera, F., 2018. Principal components analysis random discretization ensemble for big data. **Knowledge-Based Systems**, **150** : 166-174.

64. Nair, L.R., Shetty, S.D., Shetty, S.D., 2018. Applying spark based machine learning model on streaming big data for health status prediction. **Computers & Electrical Engineering**, **65** : 393-399.

65. Ramírez-Gallego, S., Mouriño-Talín, H., Martínez-Rego, D., Bolón-Canedo, V., vd., 2017. An information theory-based feature selection framework for big data under apache spark. **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, **48** (9): 1441-1453.

66. Assefi, M., Behravesh, E., Liu, G., Tafti, A.P., 2017. Big data machine learning using apache spark mllib, *2017 IEEE International Conference on Big Data (Big Data)*, s. 3492-3498.

67. Lv, Y., Duan, Y., Kang, W., Li, Z., vd., 2015. Traffic flow prediction with big data: a deep learning approach. **IEEE Transactions on Intelligent Transportation Systems**, **16** (2): 865-873.

68. John Walker, S., 2014. Big data: A revolution that will transform how we live, work, and think. Taylor & Francis .

69. Kitchin, R., 2014. The real-time city? big data and smart urbanism. **GeoJournal**, **79** (1): 1-14.

70. Kasun, L.L.C., Zhou, H., Huang, G.-B., Vong, C.M., 2013. Representational learning with elms for big data. .

71. Brin, S., Page, L., 1998. The anatomy of a large-scale hypertextual web search engine, *Seventh International World-Wide Web Conference (WWW 1998)*, Brisbane, Australia, .

72. Larose, D.T., 2004. Discovering Knowledge in Data: An Introduction to Data Mining. Wiley-Interscience Hoboken, N.J, ss. 222.

73. Özçınar, H., 2006. KPSS sonuçlarının veri madenciliği yöntemleriyle tahmin edilmesi. Pamukkale Üniversitesi Fen Bilimleri Enstitüsü, Master's Thesis, .

74. Cios, K.J., Pedrycz, W., Swiniarski, R.W., Kurgan, L., 2007. Data Mining: A Knowledge Discovery Approach. Springer US .

75. Maimon, O., Rokach, L. (Ed.), 2010. Data Mining and Knowledge Discovery Handbook. Springer US .

76. Sever, H., Oğuz, B., 2002. Veri tabanlarında bilgi keşfine formel bir yaklaşım: kısım ı: eşleştirme sorguları ve algoritmalar. **Bilgi Dünyası**, **3** (2): 173-204.

77. Gülmez, B., 2016. Training Artificial Neural Networks with New Metaheuristic Algorithms and Using in Classification. Erciyes University, Master Thesis, Kayseri, .

78. Zaslavsky, A., Perera, C., Georgakopoulos, D., 2013. Sensing as a service and big data. **ArXiv:1301.0159 cs.**, .

79. Katal, A., Wazid, M., Goudar, R.H., 2013. Big data: ıssues, challenges, tools and good practices, *2013 Sixth International Conference on Contemporary Computing (IC3)*, s. 404-409.

80. Zikopoulos, P., Eaton, C., Deroos, D., Deutsch, T., vd., 2017. Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. McGraw Hill Education ss. 176.

81. Gandomi, A., Haider, M., 2015. Beyond the hype: big data concepts, methods, and analytics. **International Journal of Information Management**, **35** (2): 137-144.

82. Welcome to apache™ hadoop®!. (Web sayfası: **http://hadoop.apache.org/**), (Erişim tarihi: Ocak 2018).

83. Bhathal, G.S., Singh, A., 2019. Big data: hadoop framework vulnerabilities, security issues and attacks. **Array**, **1-2** : 100002.

84. Apache hadoop 3.3.0 – hdfs architecture. (Web sayfası: **https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html**), (Erişim tarihi: Aralık 2020).

85. Cornell virtual workshop: distributed file systems. (Web sayfası: **https://cvw.cac.cornell.edu/mapreduce/dfs**), (Erişim tarihi: Aralık 2020).

86. Apache hadoop 3.3.0 – mapreduce tutorial. (Web sayfası: **https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html**), (Erişim tarihi: Aralık 2020).

87. MapReduce 101: what it is & how to get started - talend. (Web sayfası: **https://www.talend.com/resources/what-is-mapreduce/**), (Erişim tarihi: Aralık 2020).

88. Apache hadoop 3.3.0 – apache hadoop yarn. (Web sayfası: **https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html**), (Erişim tarihi: Aralık 2020).

89. MLlib | apache spark. (Web sayfası: **https://spark.apache.org/mllib/**), (Erişim tarihi: Aralık 2020).
90. Apache spark™ - lightning-fast cluster computing. (Web sayfası: **https://spark.apache.org/**), (Erişim tarihi: Ocak 2018).
91. Spark sql & dataframes | apache spark. (Web sayfası: **https://spark.apache.org/sql/**), (Erişim tarihi: Aralık 2020).
92. Spark streaming | apache spark. (Web sayfası: **https://spark.apache.org/streaming/**), (Erişim tarihi: Aralık 2020).
93. Structured streaming programming guide - spark 3.0.1 documentation. (Web sayfası: **https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html**), (Erişim tarihi: Aralık 2020).
94. GraphX | apache spark. (Web sayfası: **https://spark.apache.org/graphx/**), (Erişim tarihi: Aralık 2020).
95. Gonzalez, J.E., Xin, R.S., Dave, A., Crankshaw, D., vd., 2014. GraphX: graph processing in a distributed dataflow framework, *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, s. 599-613.
96. Lineer regresyon uygulaması(makine öğrenmesi) (linear regresion example machine learning). (Web sayfası: **https://medium.com/@mustafaserdarkonca/lineer-regresyon-uygulamas%C4%B1-makine-%C3%B6%C4%9Frenmesi-linear-regresion-example-machine-learning-aae7c0fec274**), (Erişim tarihi: Aralık 2020).
97. Isotonic regression. *Wikipedia* 2020.
98. Makine öğrenmesi algoritmaları- id3 algoritması. (Web sayfası: **https://medium.com/deep-learning-turkiye/makine-%C3%B6%C4%9Frenmesi-algoritmalar%C4%B1-id3-algoritmas%C4%B1-71983b3e3b77**), (Erişim tarihi: Aralık 2020).
99. Sumalatha, R., Rao, R.V., Rajasekhar, D., Thermal face recognition using bag of words. .
100. A to z of svm — machine learning for everyone. (Web sayfası: **https://towardsdatascience.com/a-to-z-of-svm-machine-learning-for-everyone-902fdd8fe9a1**), (Erişim tarihi: Aralık 2020).

101. Dikici, B., Tuntas, R., 2020. An artificial neural network (ann) solution to the prediction of age-hardening and corrosion behavior of an al/tic functional gradient material (fgm): **Journal of Composite Materials**, .
102. Öztemel, E., 2003. Yapay Sinir Ağları. Papatya Yayıncılık .
103. Cintra, R.S., Velho, H.F. de C., 2014. Data assimilation by artificial neural networks for an atmospheric general circulation model: conventional observation. **ResearchGate**, **2** .
104. Qadir, A., Semke, W., Neubert, J., 2014. Vision based neuro-fuzzy controller for a two axes gimbal system with small uav. **Journal of Intelligent & Robotic Systems**, **74** (3-4): 1029-1047.
105. Namin, A.H., Leboeuf, K., Muscedere, R., Wu, H., vd., 2009. Efficient hardware implementation of the hyperbolic tangent sigmoid function. **Proceedings - IEEE International Symposium on Circuits and Systems**, : 2117-2120.
106. Negrov, D.V., Karandashev, I.M., Shakirov, V.V., Matveyev, Y.A., vd., 2015. A plausible memristor ımplementation of deep learning neural networks. .
107. Elmas, Ç., 2003. Yapay Sinir Ağları. Seçkin Yayınları Ankara, .
108. UCI machine learning repository: poker hand data set. (Web sayfası: **https://archive.ics.uci.edu/ml/datasets/Poker+Hand**), (Erişim tarihi: Aralık 2020).
109. UCI machine learning repository: susy data set. (Web sayfası: **https://archive.ics.uci.edu/ml/datasets/SUSY**), (Erişim tarihi: Aralık 2020).
110. UCI machine learning repository: hıggs data set. (Web sayfası: **https://archive.ics.uci.edu/ml/datasets/HIGGS**), (Erişim tarihi: Aralık 2020).
111. UCI machine learning repository: hepmass data set. (Web sayfası: **http://archive.ics.uci.edu/ml/datasets/hepmass**), (Erişim tarihi: Aralık 2020).
112. WISDM lab: dataset. (Web sayfası: **https://www.cis.fordham.edu/wisdm/dataset.php**), (Erişim tarihi: Kasım 2020).
113. Lockhart, J., Weiss, G., Xue, J., Gallagher, S., vd., 2011. Design considerations for the wısdm smart phone-based sensor mining architecture. .
114. UCI machine learning repository: census ıncome data set. (Web sayfası: **https://archive.ics.uci.edu/ml/datasets/census+income**), (Erişim tarihi: Aralık 2020).

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı ve Soyadı :	Burak Gülmez
Uyruğu:	Türkiye Cumhuriyeti
Doğum Tarihi ve Yeri:	21.10.1991 / Sivas
Medeni Durumu:	Bekar
E-posta:	burakgulmez@erciyes.edu.tr
Adres:	Erciyes Üniversitesi Mühendislik Fakültesi Endüstri Mühendisliği Melikgazi / Kayseri

EĞİTİM

Derece	Okul	Mezuniyet Tarihi
Doktora	Erciyes Üniversitesi	2021
Yüksek Lisans	Erciyes Üniversitesi	2016
Yüksek Lisans	Hacettepe Üniversitesi	2015 (Yatay Geçiş)
Lisans	Hacettepe Üniversitesi	2014

İŞ DENEYİMLERİ

Yıl	Kurum	Görev
2014 – Halen	Erciyes Üniversitesi	Araştırma Görevlisi

YABANCI DİL

İngilizce, İspanyolca

YAYINLAR

Tapkan, P., Bahar, F., Kulluk, S., Özbakir, L., Gülmez, B., 2019. A Constraint Programing Based Column Generation Approach for Crew Scheduling in Railway Transportation. (Antalya): 217

Gülmez, B., Kulluk, S., 2019. Social Spider Algorithm for Training Artificial Neural Networks. International Journal of Business Analytics (IJBAN), 6 (4): 32–49

Gülmez, B., Kulluk, S., 2018. Simultaneous Feature Selection And Ann Training For Data Mining Applications. (Kiev): 220

Gülmez, B., Kulluk, S., 2018. SSIM: A Hybrıd Of Socıal Spıder And Ions Motıon Algorıthms And Its Applıcatıon To Ann Traınıng. (Kiev): 221

Gülmez, B., Kulluk, S., 2017. Training Artificial Neural Networks with Ions Motion Algorithm on Real Estate Dataset. (Budapeşte): 505

Gülmez, B., Kulluk, S., Özbakir, L., Tapkan, P., 2017. Stochastic Fractal Search Algorithm for Training ANNs on Production Dataset. (Viyana): 6

Gülmez, B., Kulluk, S., Torun, Hülya, 2017. Training Fuzzy Neural Networks with Social Spider Algorithm. (Ankara)

Gülmez, B., Kulluk, S., Özbakir, L., Tapkan, P., 2017. Üretim Veri Kümesi Üzerinde YSA'nın Eğitiminde Stokastik Fraktal Arama Algoritması. (Adana)

Gülmez, B., Kulluk, S., 2016. İyon Hareketi Algoritması ile Yapay Sinir Ağlarının Eğitimi. (İzmir)

Gülmez, B., Kulluk, S., 2016. Training Artificial Neural Networks with Social Spider Algorithm. (İstanbul): 576–581

Gülmez, B., Kulluk, S., Özbakir, L., 2015. Karınca Koloni Algoritması Kullanılarak Su Dağıtım Sisteminin Optimizasyonu. (Ankara): 169

PROJELER

Özbakır, L., Kulluk, S., Tapkan, P.Z., Gülmez, B., Bahar, F., 2020. Hafif Raylı Sistemlerde Entegre Tarife Belirleme ve Makinist Çizelgeleme Problemi için Matsezgisel Yaklaşım (Tübitak 1001)

Kulluk, S., Gülmez, B., 2017. Yapay Sinir Ağlarının Yeni Metasezgisel Algoritmalar ile Eğitimi ve Veri Madenciliğinde Sınıflandırma Alanında Kullanımı (BAP)