

T.C.
KÜTAHYA DUMLUPINAR ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

**YAPAY SİNİR AĞLARI İLE OTEL MÜŞTERİ YORUMLARININ
METİNSEL YOLLA SINIFLANDIRILARAK DUYGU
ANALİZLERİNİN TESPİTİ**

Danışman:
Dr. Öğr. Üyesi Hasan TEMURTAŞ

Hazırlayan:
Ümmügülsüm MENGUTAYCI

Kütahya – 2021

KABUL VE ONAY

Lisansüstü Eğitim Enstitüsü Müdürlüğüne,

Bu çalışma, jürimiz tarafından

Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Tez Jürisi	İmza	
	Kabul	Red
Dr. Öğr. Üyesi Hasan TEMURTAŞ (Danışman)		
Prof. Dr. Ayhan İSTANBULLU		
Dr. Öğr. Üyesi Durmuş ÖZDEMİR		

Onay

İmza

Prof. Dr. Şahmurat ARIK

Enstitü Müdürü

BİLİMSEL ETİK BİLDİRİMİ

Yüksek Lisans tezi olarak hazırladığım “Yapay Sinir Ağları İle Otel Müşteri Yorumlarının Metinsel Yolla Sınıflandırılarak Duygu Analizlerinin Tespiti” adlı çalışmanın öneri aşamasından sonuçlandığı aşamaya kadar geçen süreçte bilimsel etiğe ve akademik kurallara özenle uyduğumu, tez içindeki tüm bilgileri bilimsel ahlak ve gelenek çerçevesinde elde ettiğimi, tez yazım kurallarına uygun olarak hazırladığımı, bu çalışmamda doğrudan veya dolaylı olarak yaptığım her alıntıya kaynak gösterdiğimi beyan ederim.

30/03/2021

Ümmügölsüm MENGUTAYCI

ÖZET

YAPAY SINIR AĞLARI İLE OTEL MÜŞTERİ YORUMLARININ METİNSEL YOLLA SINIFLANDIRILARAK DUYGU ANALİZLERİNİN TESPİTİ

MENGUTAYCI, Ümmügülsüm
Yüksek Lisans Tezi, Bilgisayar Mühendisliği Anabilim Dalı
Tez Danışmanı: Dr. Öğr. Üyesi Hasan TEMURTAŞ
Mart, 2021, 98 sayfa

İnternet kullanımının yaygınlaşmasıyla birlikte insanlar farklı platformlarda duygu ve düşüncelerini de ifade etme imkânı bulmuşlardır. Günümüzde insanlar, aldıkları bir hizmetle ilgili internet ortamında o hizmete yönelik olumlu, olumsuz düşüncelerini paylaşmaktadır. Bu çalışma kapsamında, popüler bir otel sitesinden Türkiye’deki otellere yönelik yorumlar üzerinde duygu analizi çalışması yapılmıştır. Verilerin elde edilmesinde PHP programlama dili kullanılmıştır. Duygu analizi çalışmasında, metin sınıflandırma algoritmalarından YSA algoritması kullanılarak, farklı normalizasyon teknikleri ile sınıflandırılma işlemi yapılarak başarıları karşılaştırılmıştır. Farklı normalizasyon yöntemlerinin sınıflandırmadaki başarıya etkisi gösterilmek istenmiştir.

Anahtar Kelimeler: Metin Sınıflandırma, Yapay Sinir Ağları, Tf-idf Kelime Vektörü, Normalizasyon, Doğruluk.

ABSTRACT**DETERMINATION OF EMOTION ANALYSIS BY CLASSIFICATION OF
HOTEL CUSTOMER COMMENTS WITH ARTIFICIAL NEURAL
NETWORKS BY TEXT CLASSIFICATION****MENGUTAYCI, Ümmügülsüm****Master Thesis, Department of Computer Engineering****Supervisor: Asist. Prof. Hasan TEMURTAŞ****March, 2021, 98 pages**

With the increasing use of the Internet, people have had the opportunity to express their feelings and thoughts on different platforms. Today, people share their positive and negative thoughts about a service they receive on the internet. In this study, the feeling was made popular hotel site analysis study for comments on hotels in Turkey. PHP programming language was used to obtain the data. In the sentiment analysis study, using the ANN algorithm, one of the text classification algorithms, the classification process was performed with different normalization techniques and their success was compared. It was aimed to show the effect of different normalization methods on the success in classification.

Keywords: Text Classification, Artificial Neural Network, Tf-idf vector, Normalization, Accuracy.

ÖNSÖZ

Bu Yüksek Lisans tez çalışmamda, bana her türlü desteği sağlayan kıymetli aileme sonsuz teşekkürlerimi sunarım.

Çalışma boyunca her zaman yanımda olan manevi ablam Gülay SERTTAŞ'a ve değerli arkadaşım Elif AVŞAR'a teşekkürlerimi sunarım.

Çalışma boyunca bana sağladıkları destek ve katkılarından dolayı danışmanım sayın Dr. Öğr. Üyesi Hasan TEMURTAŞ hocama ve Tarsus Üniversitesi öğretim üyesi sayın Dr. Öğr. Üyesi Deniz ÜSTÜN hocama teşekkürlerimi sunarım.



İÇİNDEKİLER

Sayfa

ÖZET.....	v
ABSTRACT	vi
TABLolar LİSTESİ.....	x
ŞEKİLLER LİSTESİ.....	xi
KISALTMALAR	xvi
GİRİŞ	1

BİRİNCİ BÖLÜM

METİN SINIFLANDIRMA

1.1. METİN SINIFLANDIRMA KAVRAMI	4
1.1.1. Veri Kümesinin Toplanması.....	4
1.1.2. Ön işleme.....	5
1.1.3. Özellik Seçimi	5
1.1.3.1. Fisher Skor (F-Skor)	6
1.1.3.2. Ki-Kare testi	6
1.1.3.3. Bilgi Kazancı.....	7
1.1.3.4. İleriye Doğru Öznitelik Seçimi	7
1.1.3.5. ReliefF.....	7
1.1.3.6. Ardışık Geri Yönde Seçim	7
1.1.4. Vektör Uzay Modeli Belirleme	7
1.1.4.1. Bag of Words	8
1.1.4.2. Fasttext	8
1.1.4.3. Word2Vec	8
1.1.4.4. TF-IDF	9
1.1.4.5. Glove	9
1.1.4.6. LSA	10
1.1.4.7. Doc2Vec.....	10
1.1.5. Sınıflandırma Algoritmaları	11
1.1.5.1. Naive Bayes Sınıflandırma Algoritması	11
1.1.5.2. KNN Algoritması	11
1.1.5.3. Lojistik Regresyon Algoritması	12
1.1.5.4. Destek Vektör Makinesi.....	12
1.1.5.5. Karar Ağaçları	13
1.1.5.5.1. C4.5 Algoritması.....	13
1.1.5.5.2. Rastgele Orman Algoritması.....	13
1.1.5.6. Yapay Sinir Ağları	13

İKİNCİ BÖLÜM

LİTERATÜR ARAŞTIRMASI

2.1. METİN SINIFLANDIRMA ALANINDA YAPILAN YAYINLAR.....17

ÜÇÜNCÜ BÖLÜM

YÖNTEM

3.1. MATERYAL, YÖNTEM VE UYGULAMA.....23

3.1.1. PHP Programlama Dili	23
3.1.2. Python Programlama Dili	23
3.1.3. Keras.....	23
3.1.4. Verilerin Toplanması.....	24
3.1.5. Veri Seti Üzerinde İşlemler	27
3.1.6. Eğitim ve Test Veri Setini Oluşturma	29
3.1.7. Veri Setinin Sayısallaştırılma İşlemi	29
3.1.7.1 TF-IDF	29
3.1.7.2. Sınıfları Nümerik Yapma	30
3.1.8. Normalizasyon.....	30
3.1.8.1. Min-Max	31
3.1.8.2. Z-score.....	31
3.1.8.3. Karekök Normalizasyonu.....	31
3.1.8.4. StandartScaler().....	31
3.1.8.5. RobustScaler().....	31
3.1.9. Sınıflandırma İşlemi	31

3.2. UYGULAMA ÇIKTILARI.....32

3.2.1. Manuel Olarak Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması	32
3.2.1.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması.....	32
3.2.1.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması	35
3.2.1.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması.....	37
3.2.1.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması	39
3.2.1.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması	42
3.2.2. Train_test_split Komutu Kullanılarak Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması.....	44
3.2.2.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması.....	44
3.2.2.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması	47
3.2.2.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması.....	49
3.2.2.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması.....	52

3.2.2.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması	54
3.2.3. K Değeri 5 Alınarak Yapılan K-fold Çarpaz Doğrulama İle Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması	56
3.2.3.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması	56
3.2.3.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması	58
3.2.3.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması	59
3.2.3.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması	61
3.2.3.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması	62
3.2.4. K Değeri 10 Alınarak Yapılan K-fold Çarpaz Doğrulama İle Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması	64
3.2.4.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması	64
3.2.4.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması	66
3.2.4.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması	67
3.2.4.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması	69
3.2.4.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması	70
3.3. BULGULAR	72
SONUÇ	73
EKLER	74
KAYNAKÇA	94
DİZİN	98

TABLÖLAR LİSTESİ

Sayfa

Tablo 2.1: Metin Sınıflandırma İle İlgili Çalışmalar	20
Tablo 3.1: Ağın farklı durumlarda başarı oranı.....	72



ŞEKİLLER LİSTESİ

Sayfa

Şekil 1.1: Özellik Seçimi Akış Şeması	6
Şekil 1.2: CBOW ve Skip-gram Çalışması.....	9
Şekil 1.3: PV-DM Yapısı.....	10
Şekil 1.4: PV-DBOW Yapısı.....	11
Şekil 1.5: KNN Algoritması Yapısı.....	12
Şekil 1.6: YSA Yapısı	14
Şekil 1.7: İleri Beslemeli Nöron Yapısı.....	14
Şekil 1.8: Geri beslemeli Nöron Yapısı.....	15
Şekil 3.1: Yarışmada İlk Beş Takım Tarafından Kullanılan Modeller.....	24
Şekil 3.2: Müşteri Yorum ve Durumlarının PHP İle İlgili Siteden Çekilmesi.....	25
Şekil 3.3: Müşteri Yorum ve Durumlarının Mysql Veritabanında Gösterimi.....	25
Şekil 3.4: Olumsuz Müşteri Yorumu.....	26
Şekil 3.5: Olumlu Müşteri Yorumu.....	26
Şekil 3.6: MYSQL Veritabanında Olumlu ve Olumsuz Yorumların Sayısının Gösterimi	27
Şekil 3.7: Veri Setindeki Olumlu ve Olumsuz Yorumların Dağılım Grafiği	27
Şekil 3.8: PHP preg_replace Kullanılarak Bağlaç, Edat vs. Veri Setinden Çıkarılması	28
Şekil 3.9: PHP str_replace Kullanılarak Noktalama İşareti, Sayılar, Özel Karakterlerin Veri Setinden Temizlenmesi	28
Şekil 3.10: PHP Veri Seti İçindeki Büyük Harflerin Küçük Harfe Dönüştürülmesi.....	28
Şekil 3.11: PHP Dili İle Kelimeler Arasındaki Birden Fazla Boşluğun Silinmesi.....	28
Şekil 3.12: TfidfVectorizer İşleminden Sonra Veri Setinin Görünümü	30
Şekil 3.13: Kategorik Olan Hedef Değişkenin Nümerik Gösterimi	30
Şekil 3.14: Modelin Doğruluk-Hata Oranları Grafiği	33

Şekil 3.15: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	33
Şekil 3.16: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	33
Şekil 3.17: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	34
Şekil 3.18: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	34
Şekil 3.19: Modelin Doğruluk-Hata Oranları Grafiği	35
Şekil 3.20: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	35
Şekil 3.21: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	36
Şekil 3.22: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	36
Şekil 3.23: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	37
Şekil 3.24: Modelin Doğruluk-Hata Oranları Grafiği	37
Şekil 3.25: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	38
Şekil 3.26: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	38
Şekil 3.27: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	38
Şekil 3.28: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	39
Şekil 3.29: Modelin Doğruluk-Hata Oranları Grafiği	40
Şekil 3.30: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	40
Şekil 3.32: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	41
Şekil 3.33: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	41
Şekil 3.34: Modelin Doğruluk-Hata Oranları Grafiği	42
Şekil 3.35: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	42
Şekil 3.36: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	43
Şekil 3.37: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	43
Şekil 3.38: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	44
Şekil 3.39: Modelin Doğruluk-Hata Oranları Grafiği	45

Şekil 3.40: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	45
Şekil 3.41: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	45
Şekil 3.42: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	46
Şekil 3.43: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	46
Şekil 3.44: Modelin Doğruluk-Hata Oranları Grafiği	47
Şekil 3.45: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	47
Şekil 3.46: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	48
Şekil 3.47: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	48
Şekil 3.48: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	49
Şekil 3.49: Modelin Doğruluk-Hata Oranları Grafiği	50
Şekil 3.50: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	50
Şekil 3.51: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	50
Şekil 3.52: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	51
Şekil 3.53: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	51
Şekil 3.54: Modelin Doğruluk-Hata Oranları Grafiği	52
Şekil 3.55: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	52
Şekil 3.56: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	53
Şekil 3.57: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	53
Şekil 3.58: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	54
Şekil 3.59: Modelin Doğruluk-Hata Oranları Grafiği	54
Şekil 3.61: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	55
Şekil 3.62: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	55
Şekil 3.63: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı.....	56
Şekil 3.64: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	57

Şekil 3.65: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	57
Şekil 3.66: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	57
Şekil 3.67: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	57
Şekil 3.68: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	58
Şekil 3.69: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	58
Şekil 3.70: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	58
Şekil 3.71: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	59
Şekil 3.72: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	59
Şekil 3.73: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	60
Şekil 3.74: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	60
Şekil 3.75: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	60
Şekil 3.76: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	61
Şekil 3.77: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	61
Şekil 3.78: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	61
Şekil 3.79: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	62
Şekil 3.80: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	62
Şekil 3.81: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	63
Şekil 3.82: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	63
Şekil 3.83: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	63
Şekil 3.84: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	64
Şekil 3.85: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	64
Şekil 3.86: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	65
Şekil 3.87: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	65
Şekil 3.88: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	66
Şekil 3.89: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	66

Şekil 3.90: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	66
Şekil 3.91: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	67
Şekil 3.92: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	67
Şekil 3.93: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	68
Şekil 3.94: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	68
Şekil 3.95: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	68
Şekil 3.96: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	69
Şekil 3.97: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	69
Şekil 3.98: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	69
Şekil 3.99: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	70
Şekil 3.100: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları.....	70
Şekil 3.101: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı	71
Şekil 3.102: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım	71
Şekil 3.103: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik	71

KISALTMALAR

IBM	International Business Machines
CBOW	Continuous Bag of Word
CERN	Avrupa Nükleer Araştırma Merkezi
DVM	Destek Vektör Makinesi
GRU	Kapılı Tekrarlayan Hücre
GLOVE	Global Vektör
HTML	Hypertext Markup Language
KNN	K En Yakın Komşu Algoritması
LSA	Gizli Anlamsal Analiz
LSTM	Uzun Ömürlü Kısa Dönem Belleği
MLP	Çok Katmanlı Yapay Sinir Ağı
MS	Metin Sınıflandırma
MYSQL	Yapısal Sorgu Dili
NASA	Ulusal Havacılık ve Uzay Dairesi
PHP	Üstünyazı Önışlemcisi
PV-DM	Paragraf Vektörü Dağıtılmış Bellek
PV-DBOW	Paragraf Vektörü Dağıtılmış Kelime Torbası
RBF	Radyal Tabanlı Fonksiyon
SMO	Ardışık Minimal Optimal Algoritması
TF-IDF	Terim Frekansı - Ters Doküman Frekansı
XML	Extensible Markup Language
YSA	Yapay Sinir Ağları

GİRİŞ

İnternetin yaygınlaşmasıyla insanların dijital ortamda vakit geçirme süreleri de artış göstermiştir. TÜİK verilerine göre evden internete erişim imkanı %90,7 değerine ulaşmıştır (Türkiye İstatistik Kurumu [TÜİK], 2020). İnsanlar internet üzerinden ürün siparişi, yemek siparişi, rezervasyon işlemi gibi pek çok şeyi kolayca yapabilmektedir. Aynı zamanda internet üzerinden aldıkları hizmete yönelik duygu ve düşüncelerini belirli platformlarda ifade etmektedirler. Online alışveriş, yemek, otel rezervasyonu gibi e-ticaret siteleri buna imkan tanımaktadır. Bu doğrultuda internet üzerinden alışveriş yapmak isteyen müşteri, almak istediği ürüne yönelik diğer müşteri yorumlarına bakarak bu ürün ile ilgili bilgi sahibi olur ve hizmeti alıp almayacağına karar verebilir. Bu nedenle, tez çalışması kapsamında popüler bir web sitesindeki otel yorumları incelenmiştir. Pek çok insan, otel rezervasyonu yaptırmadan önce rezervasyon yaptırmak istediği otelin yorumlarını inceler. Otelin ne kadar temiz olduğu, yemeklerinin nasıl olduğu, konumu, hastaneye ya da bir eczaneye olan uzaklığına bakarak otel hakkında fikir sahibi olmak ister. Önceki müşterilerin otelden memnun olup olmadığını görmek için yorumları okur. Ayrıca, otele gelen olumlu ve olumsuz yorumlar işletme için de önem arz eder. Otel işletmesi, gelen olumsuz yorumları dikkate alarak, eksik yönlerini telafi ederek müşteri kaybetmek istemez. Bu nedenle, çalışma kapsamında otellere gelen yorumlar incelenerek müşterilerin duygu durumlarının tespiti yapılmaya çalışılmıştır.

Tez kapsamında, popüler bir web sitesinden Türkiye'deki otellere yönelik yapılan yorumlar ele alınarak müşterilerin duygu analizi çalışması yapılmıştır. Yorumlar, bu tez kapsamında PHP dilinde yazılan bir web servis tarafından toplanmıştır. Ayrıca, PHP yazılımıyla alınan yorumlar, müşterilerin memnuniyetini belirttiği yorumlar olumlu yorum ve memnuniyetsizliklerini belirttiği yorumlar olumsuz yorum olmak üzere iki kategoriye ayrılmıştır. Olumlu ve olumsuz yorum kategorisi belirlenirken ilgili yoruma ait müşteri puanına bakılarak karar verilmiştir. Yorumlar tek tek okunduğunda 4 ve 5 puanın verildiği yorumlarda müşterilerin otelden memnun olduğu hatta otele tekrar geleceğini belirttikleri yorumlar olduğu görülmüştür. 1 ve 2 puanlı yorumlar incelendiğinde, müşterilerin otelin hizmetinden memnun kalmadığından, yemeklerinin kötü olduğundan ve genelde otel personelinin ilgisizliğinden şikayet ettikleri görülmüştür. 3 puanlı yorumlarda ise otelle ilgili tam

düşüncelerini yansıtmadığı ikilemde kaldıkları ve genellikle de müşterilerin otelle ilgili genel fikirlerinin ne iyi ne de kötü olduğu belirlenemeyecek şekilde genel yorumlarda bulunulduğu görülmüştür. Bu kapsamda 4 ve 5 puanlı yorumlar “olumlu yorum”, 1 ve 2 puanlı yorumlar “olumsuz yorum” olacak şekilde yine aynı yazılımla belirtilmiştir. Veriler MYSQL veritabanına kaydedilmiştir. Yorumlar içerisinde bulunan, sınıflandırmada anlamsal bir veri ifade etmeyecek kelimeler (bağlaç, edat vs.), özel karakterler ve noktalama işaretleri PHP dili kullanılarak metin içerisinden temizlenmiştir. Veriler gerekli önışlemden geçirildikten sonra TF-IDF vektör uzay modeli kullanılarak, her bir terim ağırlıklandırıldı. Bu sayede metin içerisindeki her bir terimin önem derecesi belirlenerek sınıflandırılma kalitesi artırılmış oldu. Bu işlem Python dili ile yapıldı. Veriler üzerinde gerekli işlemlerin bitirilmesinden sonra veriler üç farklı şekilde eğitim ve test verilerine ayrıldı. Python dilinde yazılmış YSA algoritması ile farklı normalizasyon yöntemleri kullanılarak verilerin sınıflandırılması işlemi yapılmış ve oluşturulan modellerin sınıflandırmadaki başarı oranları karşılaştırılmıştır.

Tez çalışması kapsamında, birinci bölümde metin sınıflandırmanın tanımı ve metin sınıflandırma süreci detaylı olarak açıklanmıştır. Tezin ikinci bölümünde, MS alanında yapılan çalışmalardan bahsedilmiş ve bu çalışmalar bir tablo halinde gösterilmiştir. Son bölümde ise tez kapsamında kullanılan materyal ve metot belirtilip, tezin uygulama kısmına yönelik çıktı ve bulgular verilmiştir.



BİRİNCİ BÖLÜM
METİN SINIFLANDIRMA

1.1. METİN SINIFLANDIRMA KAVRAMI

Metin sınıflandırma problemleri, son yıllarda çok fazla çalışmada incelenmiştir (Jiang, vd. , 2018; Kowsari, vd. 'den 2019). Metin sınıflandırma, girilen bir metnin daha önceden belirlenen hangi sınıfa ait olduğunun belirlenmesi işlemidir. Verilen her metin için bir bilgi çıkarımı yapar. MS ile yazıdan ruh hali tahmini, girilen haberin hangi haber kategorisine ait olduğunu bulma, sosyal medyada yapılan yorumlarda siber zorbalık olup olmadığı, müşterilerin aldığı ürünlerle ilgili yaptıkları yorumlardan memnuniyet durumları vb. bulunabilir.

Metin sınıflandırma üzerine çalışma 1950'lerin sonunda H.P. Luhn tarafından kelime frekansları kullanılarak metinlerin otomatik sınıflandırılması çalışması önerildi. Daha sonra 1960 yılında Maron tarafından metin sınıflandırma üzerine ilk makale yayınlandı. Ayrıca, IBM tarafından Intelligent Miner For Text yazılımı geliştirildi (Young, Youwen ve Shixiong, 2009). Bu yazılım metin analizi aracı olarak bir metnin dilini bulmak için dil tanımlama, metinleri içeriklerine göre gruplara bölme, metinleri önceden belirlenmiş bir dizi kategorisine atama, özellik çıkarma işlemlerini yapmayı sunar. Daha sonra 1981 yılına gelindiğinde Çin'de Professor Hou Han-Qing metin madenciliği üzerine çok fazla araştırma yapmıştır. 1987 yılında Çin'de Zhu Lan-Juan ve Wang Yong-Cheng bilimsel literatür deneysel sistemini geliştirmişlerdir (A.g.e).

Metin sınıflandırma süreci genel olarak şu aşamalardan oluşur:

- Verinin toplanması
- Ön işleme aşaması
- Özellik seçimi
- Vektör uzay modeli seçimi
- Sınıflandırma aşaması (İnan,2019).

1.1.1. Veri Kümesinin Toplanması

Bu aşamada ilgili çalışmada kullanılacak veriler toplanmaya çalışılır. Çalışma bir haber sitesindeki haberlerin kategorilere ayrılması ise, ilgili haber sitesinden ilgili kategorilerdeki veriler alınır. İlgili veri seti sosyal medyadaki yorumlardan oluşması gerekiyorsa o platformlardaki konuyla alakalı yorumlar bir uygulama aracılığıyla ya da

tek tek toplanır, belirli bir e-ticaret sitesindeki müşterilerin yorumların incelenmesi durumunda ilgili siteden yorumlar yine bir uygulama aracılığıyla ya da elle toplanır. Sonuç olarak çalışma hangi kapsamda ise ilgili veriler anket, yazılan bir web servis ya da elle manuel olarak toplanır. Bu tez kapsamında popüler bir otel sitesindeki yorumlar veri seti olarak kullanılmıştır. Hanife Göker ve Hakan Tekedere (2017)’ye göre “Verinin toplama süreci, veriyi anlama sürecidir”.

1.1.2. Ön işleme

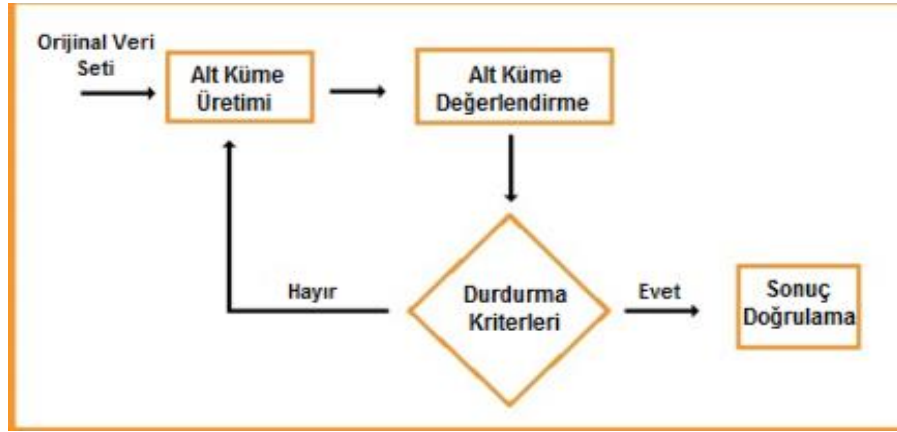
Başlangıç verisini anlaşılabilir formata dönüştürmeyi amaçlar (Işık, N., 2019). Veri ön işleme adımı, eksik verileri silme, edat, zarf, bağlaç, noktalama işaretleri, HTML etiketleri gibi anlamsal bir değeri olmayan, çalışmamızda bize katkı sağlayamayacak olan verileri silme aşamasıdır.

Veri ön işleme aşamasında, metinler XML ve HTML etiketlerinden temizlenir. Metin, kelimelere ayrılır. Metin içinde çok fazla geçen ancak eğitim ve test aşamasında çalışmaya katkı sağlamayacak olan edat, bağlaç ve zamir gibi kelimeler metin içerisinden çıkarılır. Bu işlem durak kelimelerin atılması işlemidir. Kelimelerde kök bulma işlemi yapılır (Göker ve Tekedere, 2017). Bu sayede hem aynı anlama gelen kelimelerin birleştirilmesi hem de özellik matris boyutunun indirgenmesiyle işlem karmaşıklığı azaltılmış olur (Amasyalı, vd., 2012).

1.1.3. Özellik Seçimi

Özellik seçimi, ilgili özelliklerin seçilmesini ve ilgisiz olanların veri setinden atılmasını içerir (Asim, vd., 2017). Özellik seçimi, orijinal veri setini en iyi temsil edebilecek olan alt kümeyi seçebilmedir (Budak, 2018). Bu sebepten ötürü özellik seçimi kullanıldığında veri kümesi içindeki özellik sayısı azalmış olur. Özellik seçimi yöntemiyle birlikte veri kümesi üzerindeki gereksiz ilgili olmayan veri ortadan kaldırılmış olur. Azaltılmış veri kümesiyle yapılan algoritmanın hızı da artmış olur. Daha anlamlı veriler haline dönüştürülen yeni veri setiyle birlikte oluşturulan modelin başarısı da artabilir.

Şekil 1.1: Özellik Seçimi Akış Şeması



Kaynak: Dash ve Liu, 1997; Budak'dan 2018: 1

Özellik seçim yöntemleri filtre, sarmal ve gömülü olmak üzere bu üç yaklaşıma dayanır (Asim, vd., 2017). Filtreleme yöntemlerinde metin madenciliği algoritmaları çalıştırılmadan önce özellik seçimi yapılırken, sarmal yöntemde metin madenciliği algoritmasında en iyi özelliğin bulunabilmesi için bir araç olarak kullanılırken, gömülü yöntemlerde ise sınıflandırma süresinin azaltılması amacıyla metin madenciliği ve özellik seçme yöntemleri eş zamanlı olarak kullanılır (Asim, vd., 2017; Budak, 2018).

1.1.3.1. Fisher Skor (F-Skor)

Filtre tabanlı bir yöntemdir. Her özelliğin diğer özelliklerden farkını hesaplar (Aydilek, Gümüşçü ve Taşaltın, 2016). Özelliklere ait ortalama ve standart sapma değerlerini kullanır (Budak, 2018). Çıkan skor değeri en yüksek olan özellik değerleri tercih edilir (Aydilek, Gümüşçü ve Taşaltın, 2016; Budak, 2018).

1.1.3.2. Ki-Kare testi

Filtre tabanlıdır. Özelliklerin X^2 değerine göre büyükten küçüğe sıralar ve en üst sıradan başlayarak belirlenen boyutta özelliği alır (Budak, 2018).

$$(X)^2 = \sum_{i=1}^n \sum_{j=1}^m \frac{(A_{ij} - (R_{ij}C_{ij}/N))^2}{\frac{R_{ij}C_{ij}}{N}} \quad N: \text{veri setindeki toplam nitelik sayısı}$$

(Yang, 2007; Aydılek, Gümüşçü ve Taşaltın'dan 2016)

1.1.3.3. Bilgi Kazancı

Her özellik için bir IG sayısı hesaplar ve hesaplanan en yüksek IG sayısına sahip özellikler seçilmektedir (Aydilek, Gümüşçü ve Taşaltın, 2016).

1.1.3.4. İleriye Doğru Öznitelik Seçimi

Öznitelik seçme işlemi azdan çoğa yapılır. Öncelikle boş bir veri seti ile işleme başlanır. Hedef ve karşıt hipotez belirlenir. Veri setinden seçilen herhangi bir özniteliğe eğitim yapılır. Eğitim sonucunda hedef hipotez sağlanıyorsa, seçilen öznitelik eğitim setine eklenir. Eğer karşıt hipotez sağlanıyorsa seçilen öznitelik eğitime eklenmez ve diğer özniteliğe geçilerek aynı işlemler yapılır (Kaynar, vd., 2018).

1.1.3.5. ReliefF

Bu yöntem daha çok iki sınıfa ayrılmış olan modellerde daha başarılı sonuç verir. Başlangıçta tüm özelliklerin ağırlıkları 0 olarak belirlenir. Sınıf sayısının bir eksiği alınarak bir k değişkeni belirlenir. Sonraki adımda veri setinden rastgele olarak bir veri seçer. Önce seçilen veri ile aynı sınıftan k tane veriyi alır. Daha sonra farklı sınıftan verileri alır. Seçilen her veriye ait ağırlıklar alınan verilerle güncellenir. Koşulu sağlamayan özellik veri setinden çıkarılır (Kaynar, vd., 2018).

1.1.3.6. Ardışık Geri Yönde Seçim

Tüm özelliklerle işleme başlar ve sınıflandırma başarısında herhangi bir artış olmayana kadar her adımda bir özellik geri alınmamak üzere veri setinden çıkarılır (Budak, 2018).

1.1.4. Vektör Uzay Modeli Belirleme

Metinlerin sayısal ifadelere dönüştürülmesini ifade eder (Aydoğan ve Karcı, 2019). Her nesne vektör yapısında tanımlanır (Göker ve Tekedere, 2017). Frekans ve tahmin bazlı olmak üzere ikiye ayrılmaktadır (Aydoğan ve Karcı, 2019).

1.1.4.1. Bag of Words

Frekans bazlı kelime vektörüdür. Metin içerisindeki her cümle, tekrarı olmayan kelimelere bölünür ve birbirinin aynı olmayan kelime sayısı kadar bir matris oluşturur. Satır sayısı; metin içindeki satır sayısı, sütun sayısı ise kelimelerden oluşur (A.g.e).

1.1.4.2. Fasttext

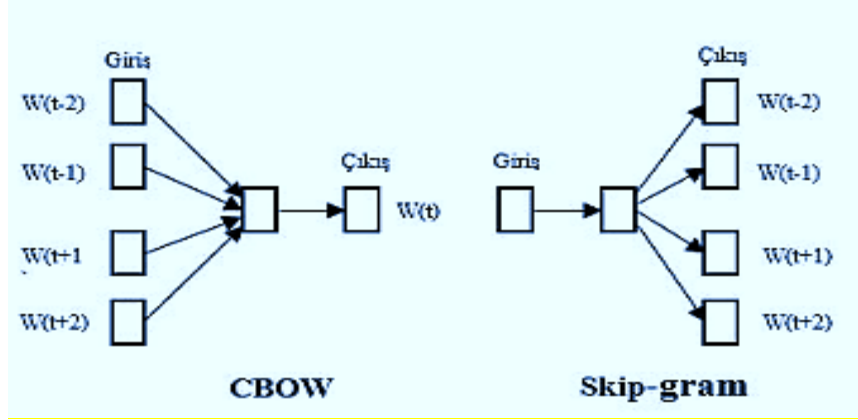
Facebook tarafından geliştirilmiştir. Bir kelimedeki metin içinde kaç tane olduğu yani n-gramları toplamları bulunur. N-gram toplamları vektörel olarak gösterilir (Aksu ve Karaman, 2020). Bütün kelime vektörleri, kelimelerin n-gram ortalaması alınarak hesaplanır. Böylece eğitim setinde olmayan ancak test verisinde olan kelimeler dahil bütün kelimelere vektör ataması yapılmış olur. Basit ve verimli çalışma şekline sahiptir (Balodis ve Dekane, 2019).

1.1.4.3. Word2Vec

Google tarafından geliştirilmiştir (Lilleberg, Zhang ve Zhu, 2015). Bu yöntemde YSA kullanılmaktadır. Denetimsiz öğrenme yöntemiyle kelimeler vektörlerine dönüştürülür (Aksu ve Karaman, 2020). CBOW ve skip-gram olmak üzere iki modeli bulunmaktadır (Aksu ve Karaman, 2020; Avaroğlu, vd., 2019). İki modelde de giriş parametresi olarak pencere boyutu adında bir değişken bulunur. CBOW, işlemi yapılacak olan kelimenin sağından ve solundan önceden belirlenen pencere boyutu kadar kelime, yapay sinir ağına girdi olarak verilir. Sonrasında da çıkan sonuç, kelimenin vektörü olarak kabul edilir (Aksu ve Karaman, 2020).

Skip-gram modelinde ise, kelime YSA algoritmasına girdi olarak alınmaktadır. Daha sonra kelimenin sağından ve solundan verilen pencere boyutu kadar kelime vektörü çıktı olarak alınmaktadır (A.g.e). CBOW modelinde, istenilen kelimenin komşu kelimelerine bakılarak, skip-gram modelinde ise istenilen kelimeye bakılarak komşuları bulunmaya çalışılmaktadır (Avaroğlu, vd.,2019).

Şekil 1.2: CBOW ve Skip-gram Çalışması



Kaynak: Avaroğlu, vd. 2019

1.1.4.4. TF-IDF

Terim frekansı ve ters vektör frekansı olarak anılır. Bag of Words modeline dayanır (Hark, vd., 2019). Kelimelerin terim frekansı yani tf değeri ile ters terim frekansı yani idf değerlerinin çarpımının sonucunda elde edilen değerdir (Doğan ve Uysal, 2018). Tf değeri, kelime köklerinin cümle içerisinde kaç kere tekrar edildiğinin sayısını verir. Idf değeri ise, ilgili kelimenin tüm metin boyunca kaç kere kullanıldığının bilgisini verir (Çalış, Gazdağı ve Yıldız, O., 2013). Terim frekansında bir kelime ne kadar çok kullanılmışsa frekans değeri de o kadar yüksek kabul edilir. Ters terim frekansı hesabında ise, bir kelime ne kadar az kullanılmışsa o kadar değerli kabul edilir (Hark, vd., 2019). Bu nedenle bağlaç, edat, zamir gibi metin içerisinde kullanılma sıklığı fazla olabilecek fakat anlamsal olarak bir şey ifade etmeyen kelimeler göz ardı edilmiş olur. Tf-idf hesabı aşağıda gösterildiği şekildedir.

$$w(t)_i = tf_i * \log\left(\frac{N}{df_i}\right) \quad (\text{Chen, vd., 2016})$$

N: toplam doküman sayısı

1.1.4.5. Glove

Word2Vec modeline benzer çalışmaktadır (Kowsari, 2019). Bu modelde, kelimelerin tüm yapı boyunca kullanım sıklığı hesaplanır (Yıldırım ve Yıldız, T., 2018). Her bir kelime için bir vektör belirlenir ve her bir kelime kendi etrafındaki vektörler kullanılarak eğitilir (Kowsari, 2019).

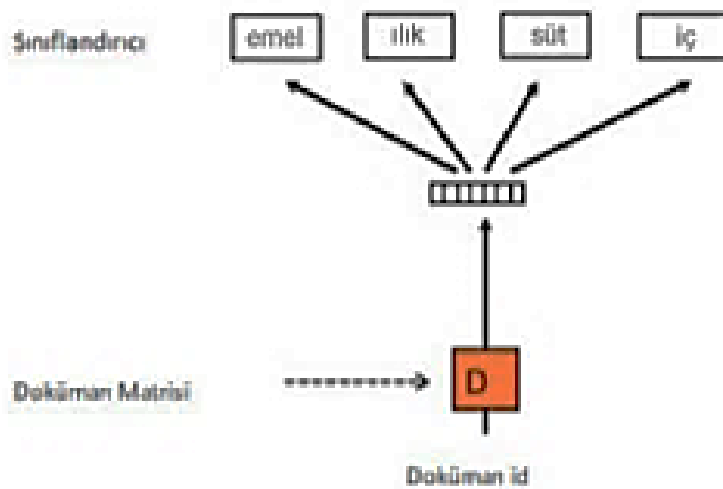
1.1.4.6. LSA

Gizli anlam analizi yöntemidir. Metin matrisi, alt matrislerine parçalanır ve daha düşük boyutta bir matris oluşturulur. Oluşturulan küçük boyutlu matris üzerinde birtakım işlemler yapılarak metin ile aralarındaki benzerliğe bakılır (Uzun ve Yıldıztepe, 2018).

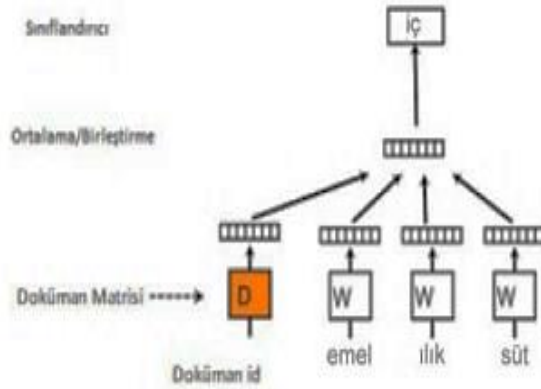
1.1.4.7. Doc2Vec

Word2vec modelinden esinlenilmiştir (Yıldırım ve Yıldız, T., 2018). Quoc Le and Tomas Mikolov tarafından geliştirilmiştir (Le ve Mikolov, 2014; Almatarnah ve Gamallo'dan 2019). Her bir doküman için o dokümana ait bir doküman id oluşturulur. PV-DM ve PV-DBOW olmak üzere iki modeli vardır. PV-DM modeli, Word2Vec kelime temsil yöntemindeki CBOW modeline benzer. PV-DBOW modeli ise, skip-gram yöntemine benzer yapıdadır (Erdoğan ve Güran, 2019).

Şekil 1.3: PV-DM Yapısı



Kaynak: Erdoğan ve Güran, 2019

Şekil 1.4: PV-DBOW Yapısı

Kaynak: Erdiñ ve Güran, 2019

1.1.5. Sınıflandırma Algoritmaları

Sınıflandırma, girilen metnin daha önceden belirlenmiş olan sınıf kategorilerine atanması işlemidir. Veri, eğitim ve test verisi olmak üzere ikiye ayrılır. Eğitim verisinde bulunan verilerin kendi sınıfları kullanılarak öğrenme süreci başlar ve bunun sonucunda da sınıflama modeli oluşur. Test verisinde, bu sınıflama modeli kullanılarak verilerin hangi sınıfta olduğu tahmin edilir (Çınar, 2019).

1.1.5.1. Naive Bayes Sınıflandırma Algoritması

Verilerin kategorilerine ayrılabilmesi için kullanılan denetimli bir sınıflandırma algoritmasıdır. Verinin hangi sınıfa ait olduğunu belirlemek için o sınıfa ait koşullu olasılığını hesaplar (Çalış, Gazdağı ve Yıldız, O., 2013).

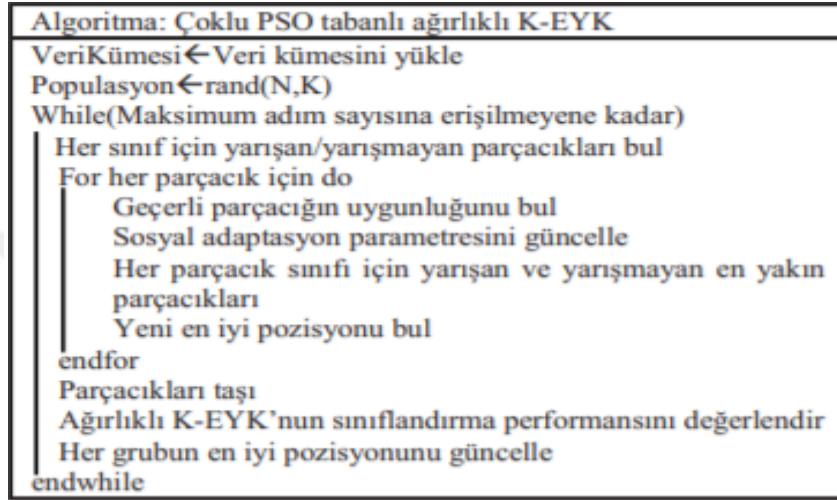
1.1.5.2. KNN Algoritması

Parçacık sürü optimizasyonu tabanlı bir algoritmadır (Aydın, Başkaya ve Salur, 2017). Bir k değeri seçilir. Seçilen değere göre veri kümesindeki her bir veri için mesafesi hesaplanır. En düşük mesafeye sahip k tane veri alınır. Bu verilerin ait olduğu sınıflar arasında en çok hangi sınıf varsa o sınıfa yeni gelen veri atanır. Bu nedenle de doğru bir sınıf seçilebilmesi için k değeri büyük önem taşır (Aşlıyan ve Tekeli, 2016). Sonuçlarda eşitlik olmaması için k değeri tek sayılar arasından bir sayı seçilir.

(Borandağ, vd., 2016). KNN algoritmasında aradaki mesafenin belirlenmesi formülü aşağıdaki gibidir (Dolgun, Özdemir ve Oğuz, 2009: Borandağ, vd. ‘den 2016).

$$d_{(i,j)} = \sqrt{\sum_{k=1}^p (X_{ik} - X_{jk})^2}$$

Şekil 1.5: KNN Algoritması Yapısı



Kaynak: Aydın, Başkaya ve Salur, 2017

1.1.5.3. Lojistik Regresyon Algoritması

Lojistik regresyon modeli, belirli özellikteki verilerin toplam sonuç içerisinde oranını hesaplayan olasılık hesabına, bir durumun olma olasılığının olmama olasılığına dayanan odds hesabına ve odds değerinin logaritmasına dayanır. Sınıflamada iki kategori var ise, ikili lojistik regresyon, sınıflama ikiden fazla sınıfa sahipse çok kategorili lojistik regresyon, eğer sınıflar sıralama ölçeği ile (düşük, orta, yüksek gibi) belirlenmiş ise sıralı lojistik regresyon analizi olarak anılır (Çokluk, 2010).

1.1.5.4. Destek Vektör Makinesi

İki katmanlı ileri beslemeli YSA' ya sahiptir (Haykin,1999: Elmas, Yakut ve Yavuz'dan 2014). Doğrusal olarak ayrıştırılamayan veri setini daha yüksek boyutlu lineer olarak ayrıştırılabilen bir uzaya taşır. Doğrusal olmayan verilerin sınıflandırılmasında başarılı sonuçlar vermektedir (Albayrak, vd. 2016).

1.1.5.5. Karar Ağaçları

Ağaç yapısındadır. Kök düğümü vardır ve kök düğümünden başlanılarak farklı sınıflara ve düğüm boyunca değişen aralıklara göre iki bölüme ayrılır. Her dal değişen sınıflara göre tekrar bölünür. Her bölünme olayında bölünen düğüme ana düğüm, bölünmüş olduğu düğümlere alt düğüm denir (Aytekin, Özfidan ve Sütçü, 2018).

Karar ağaçlarında rastgele orman, C4.5 algoritması gibi daha birçok algoritma kullanılmaktadır. Sınıflandırmada kullanılacak veriler öncelikle bu algoritmalara gönderilir. Algoritmalar aracılığıyla karar ağacı modeli oluşturulmuş olur. Daha sonra oluşturulan model test verilerinin sınıflandırılmasında, hangi sınıfa ait olduğunu tespitinde kullanılır (Aksu ve Karaman, 2017).

1.1.5.5.1. C4.5 Algoritması

Sınıflandırma işlemi, metin içerisindeki verilerin kullanım sıklığına göre yapılır. Her bir sınıfın entropisi hesaplanır. Daha sonra her bir sınıfın entropisi, tüm sınıfın entropisine bölünür. En yüksek çıkan sonuç kök düğüme verilir. Tüm veriler bitene kadar aynı işlemler tekrarlanır (A.g.e).

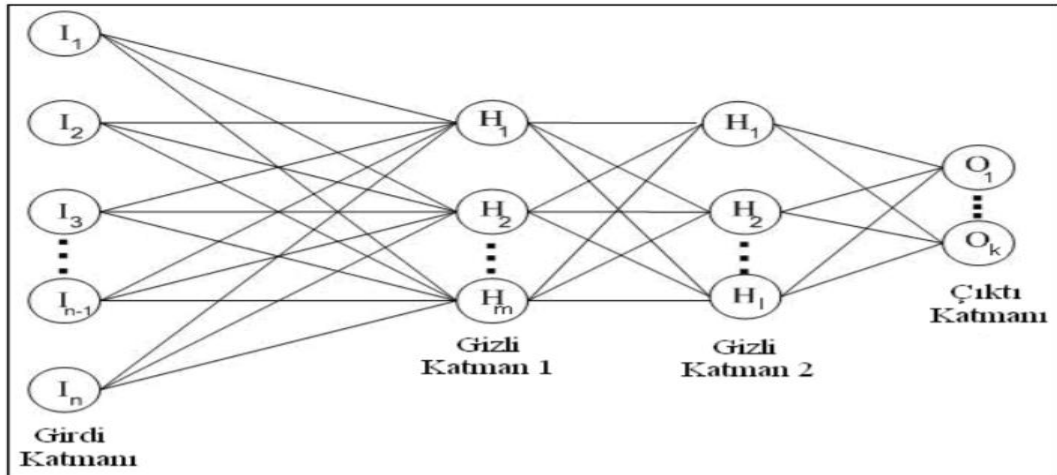
1.1.5.5.2. Rastgele Orman Algoritması

Breiman tarafından ortaya çıkmıştır. Bu algoritma ile birden fazla karar ağacı oluşturulur. Her bir karar ağacı farklı eğitim kümelerinde eğitilir ve daha sonra birleştirilir. Büyük boyuttaki veri kümesinin sınıflandırılmasında büyük başarı gösterdiği ileri sürülmüştür (A.g.e).

1.1.5.6. Yapay Sinir Ağları

İnsan beyninden esinlenilerek geliştirilen bir algoritmadır (Kınacı ve Uğur, 2006). Bir yapay sinir ağında, giriş katman, gizli katman ve çıktı katmanı bulunur. Gizli katman birden fazla olabilir. Her bağlantının ağırlık adı verilen bir sayısal değeri vardır (Wang, 2003). Yapay sinir ağı yapısı genel olarak aşağıdaki şekilde gibidir.

Şekil 1.6: YSA Yapısı

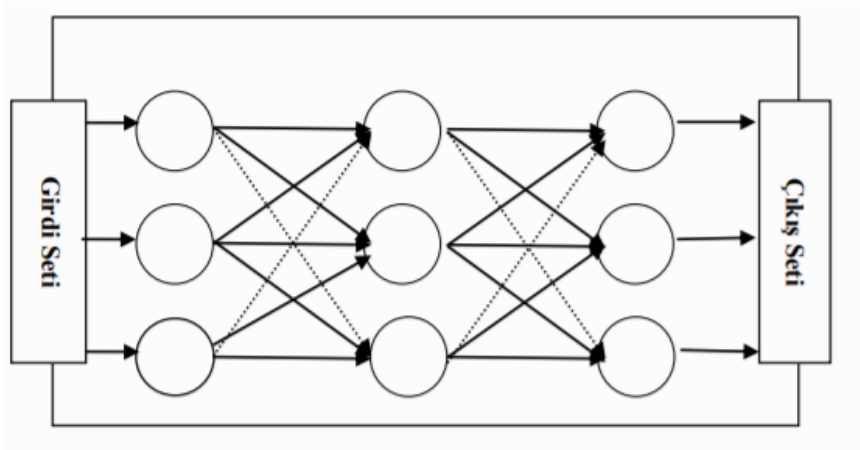


Kaynak: Badur ve Kutlu, 2009.

Giriş katmanından veriler girdi olarak alınır. Giriş katmanı ile çıktı katmanı arasında gizli katman bulunur. Bir katmandaki her nöron, bir sonraki katmanda bulunan nöronlara bir ağırlık değeri ile bağlıdır (Badur ve Kutlu, 2009). Girdi katmanına verilen verilerin girdi bilgileri ile çıktılarına karşılık gelen değerleri, bir eğitim algoritmasına göre ağırlıkları değiştirilmektedir. Eğitim sonunda eğitilmiş ağ, yeni gelen verinin sonucunun ne olabileceğini tahmin edebilmektedir (Kınacı ve Uğur, 2006).

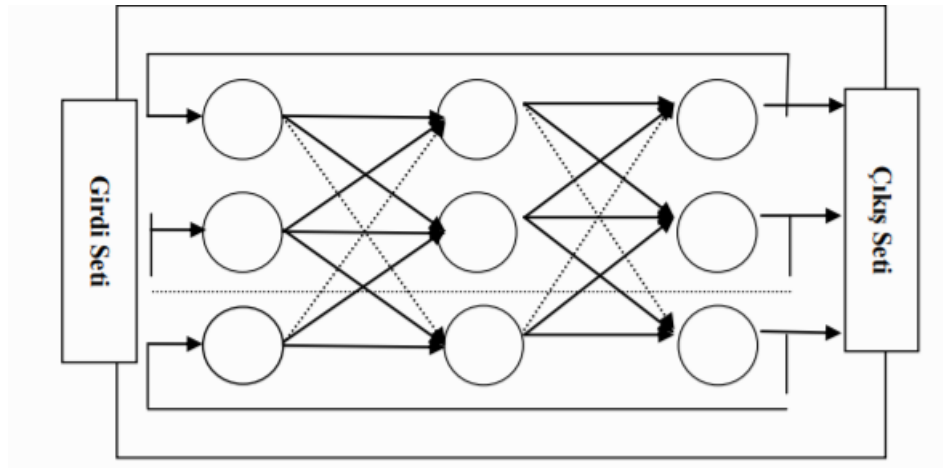
Çalışma yapısına göre YSA ileri beslemeli ve geri beslemeli ağlar olarak ikiye ayrılır. İleri beslemeli ağda, sinirler sadece bir sonraki katmandaki sinirlerle ilişkilidir. Geri beslemeli ağda ise sinirler hem kendisi ile hem kendi katmanındaki diğer sinirler ile hem de sonraki katmandaki sinirler ile bağlantılıdır (Tolon ve Tosunoğlu, 2008).

Şekil 1.7:İleri Beslemeli Nöron Yapısı



Kaynak: Deveci ve Yavuz, 2012

Şekil 1.8:Geri beslemeli Nöron Yapısı



Kaynak: Deveci ve Yavuz, 2012

YSA öğrenme yöntemine göre danışmanlı ve danışmansız öğrenme olmak üzere ikiye ayrılır (Tolon ve Tosunoğlu, 2008). Danışmanlı öğrenmede, ağ ilk önce eğitilir, ağa girdi ve çıktı bilgileri verilir. Danışmansız öğrenmede, ağa sadece girdi bilgileri verilerek ağın modeli oluşturulmaya çalışılır.



İKİNCİ BÖLÜM
LİTERATÜR ARAŞTIRMASI

2.1. METİN SINIFLANDIRMA ALANINDA YAPILAN YAYINLAR

Pilavcılar (2007) tarafından yapılan metin madenciliği ile metin sınıflandırma çalışmasında Anadolu ajansından alınan 2600 haber verisi kullanılmıştır. Ekonomi, politika, sağlık ve spor kategorilerinden oluşan makaleler ele alınmıştır. Kelime ağırlıklandırma yöntemlerinden bit, frekans ve tf-idf ağırlıklandırma kullanılmıştır. Çalışmada sınıflandırma algoritmalarından Naive Bayes ve KNN algoritmaları her bir ağırlıklandırma yöntemiyle test edilerek çıkan sonuçlar karşılaştırılmıştır. En yüksek başarımlı TF-IDF kelime ağırlıklandırma yöntemini kullanan Naive Bayes modelinde olmuştur.

Gürçan (2009) tarafından yapılan web içerik madenciliği ve konu sınıflandırılması çalışmasında ekonomi, sağlık, siyaset, bilişim ve spor sınıflarına ait veriler kullanılmıştır. Bit ve kelime frekansları kullanılarak Naive Bayes, Naive Bayes multinominal, DVM, KNN ve J48 sınıflandırma algoritmaları ile veri setinin sınıflandırılma işlemi yapılmıştır. Yapılan çalışma sonucunda Naive Bayes multinominal modelinin en yüksek sınıflandırma başarısını verdiği belirtilmiştir.

Çalış, Gazdağı ve Yıldız (2013) tarafından yapılan reklam içerikli epostaların metin madenciliği yöntemleri ile otomatik adlı çalışmada, konuyla ilgili 800 eposta toplanmıştır. Toplanan epostaların 400 tanesi normal eposta, geri kalanı ise reklam içerikli eposta olarak belirlenmiştir. Ağırlıklandırma yöntemlerinden binary, frekans ve TF-IDF kullanılmıştır. Çalışmada KNN ve Naive bayes sınıflandırma algoritmaları kullanılmıştır. Yapılan çalışma sonucunda en yüksek başarımlı binary vektör ağırlıklandırma kullanan KNN algoritması göstermiştir.

Al-Momani, Alwada'n ve Mohammad (2016) tarafından yapılan arabic text categorization using support vector machine, Naive Bayes and neural network çalışmasında farklı kategorilere ait 1400 Arapça dokümandan bir veri seti oluşturulmuştur. Metin sınıflandırma modeli olarak Naive Bayes, DVM, MLP yapay sinir ağı kullanılmıştır. Çalışma sonucunda en yüksek başarımlı yapay sinir ağı modeli göstermiştir.

Kaynar ve arkadaşları (2016) tarafından makine öğrenmesi yöntemleri ile duygu analizi adlı çalışmada film yorumlarının pozitif ve negatif sınıf olmak üzere öğrenme algoritmalarından Naive Bayes, merkez tabanlı sınıflayıcı, MLP ve DVM

algoritmalarıyla sınıflandırılma çalışması yapılmıştır. Ön işlem aşamasında TF-IDF vektör uzay modeli kullanılmıştır. Çalışma sonucunda başarımda en yüksek değerler yapay sinir ağları ve destek vektör makinelerinden elde edilmiştir.

Gürcan (2017) tarafından yapılan çalışmada, denetimli öğrenme yordamlarıyla Türkçe metinler üzerinde konu sınıflandırma analizi yapılmıştır. Bu kapsam doğrultusunda çevrimiçi haber sitelerinde, Türkçe haber metinlerinden oluşan bir veri seti hazırlanmıştır. Bu veri seti için ekonomi, sağlık, siyaset, spor olmak üzere dört sınıf belirlenmiştir. Daha sonra metinler üzerinde ön işleme, temizleme ve ayrıştırma işlemleri yapılmıştır. Bu aşamadan sonra dokuman-terim matrisi ile metinler bir matrise dönüştürülmüştür. Sınıflandırma aşamasında, denetimli öğrenme algoritmaları olan Naive Bayes, K-en yakın komşu, destekleyici vektör makinesi algoritmaları kullanılmıştır. Çalışma sonucunda en yüksek başarıma sahip algoritma Naive Bayes algoritması, en düşük başarımları gösteren algoritma ise KNN algoritması olmuştur.

Alghoul ve arkadaşları (2018) tarafından yapılan email classification using artificial neural network çalışmasında, 4601 adet mail veri seti olarak kullanılmıştır. Veri seti spam ve spam olmayan mail olmak üzere iki sınıftan oluşturulmuştur. Çarpaz doğrulama kullanılarak, YSA ile sınıflandırılma işlemi yapılmıştır.

Campos, Silva ve Bernardino (2019) tarafından yapılan text mining in hotel reviews impact of words restriction in text classification çalışmasında, Avrupa'daki oteller incelenmiştir. Çalışma kapsamında 12500 otel yorumu kullanılmıştır. Otellere yapılan yorumlar, pozitif ve negatif olarak kategoriye ayrılmış, durak kelimeler, HTML etiketlerinden temizlenerek TF-IDF matrisine dönüştürülmüştür. Daha sonra Naive Bayes, Karar ağaçları, Rastgele orman, DVM algoritmalarıyla sınıflandırma yapılmış ve algoritmaların performansları karşılaştırılmıştır.

Işık, M. (2019) tarafından yapılan müşteri yorumları üzerinde metin analitiği çalışmaları ve yorumların makine öğrenmesi algoritmaları ile modellenmesi adlı çalışmada, müşterilerin Amazon şirketinden almış oldukları ürünlerle ilgili yorumları incelenmiştir. Çalışmada yorum puanı 4 ve 5 olan yorumlar iyi yorum, 1 ve 2 olan yorumlar kötü yorum kabul edilmiş ve veri seti oluşturulmuştur. Çalışmada kelimelerin sayısal değere dönüştürülmesi işleminde Python içerisinde bulunan CountVectorizer ve TfidfVectorizer kullanılmıştır. Naive Bayes, lojistik regresyon, DVM, RBF algoritmaları sınıflandırmada kullanılmıştır. Çalışma sonucunda İYİ etiketli yorumlarda

algoritmaların başarımları yüksek iken, KOTU etiketli yorumlarda TF-IDF yaklaşımlı Lojistik regresyon modelinin en iyi tahmini yaptığı gözlemlenmiştir.

İnan (2019) tarafından otel yorumlarının sınıflandırılmasında makine öğrenmesi algoritmalarının karşılaştırılması ve Eskişehir otellerine yönelik yorumların duygu analizi çalışması yapılmıştır. Çalışmada öncelikle İstanbul'daki otellere yönelik yorumlar üzerinde ön işleme, TF-IDF vektör uzay modeliyle kelimeler sayısal değerlere dönüştürülmüştür. Daha sonra sınıflandırma algoritmalarından Naive Bayes, lojistik regresyon, KNN, destek vektör makineleriyle sınıflandırma başarımları test edilmiştir. Sonrasında en başarılı yöntemin lojistik regresyon algoritması olduğu belirlendikten sonra Eskişehir'deki altı otele yönelik yorumlar ele alınmış ve onların duygu analizi yapılmıştır. Çıkan sonuçlar neticesinde, müşterilerin genel olarak Eskişehir'deki otellerden memnun olduğu yorumu yapılmıştır.

Ahmetoğlu ve Daş (2020) tarafından Türkçe otel yorumlarıyla eğitilen kelime vektörü modellerinin duygu analizi ile incelenmesi adlı çalışma yapılmıştır. Bu çalışmada Türkçe otel yorumları veri seti olarak kullanılmıştır. Veriler Word2vec ile eğitilerek kelime vektörleri oluşturulmuştur. Eğitilmiş kelime vektörleri ile rastgele değerlere atanarak oluşturulmuş kelime vektörleri, aynı derin öğrenme yöntemi ile sınıflandırma yapılarak başarımları karşılaştırılmıştır. Çalışma sonucunda rastgele kelime vektörleri ile oluşturulmuş olan yöntemde, modelin öğrenme kayıp değerinin daha fazla olduğu görülmüştür.

Khatun ve arkadaşları (2020) tarafından yapılan comparative study on text classification çalışmasında, 5572 mesaj arasından 4825 tanesi ham mesaj, 547 mesaj spam mesaj olacak şekilde iki sınıftan oluşan bir veri seti oluşturulmuştur. Mesajlar TF-IDF matrisi kullanılarak vektör matrisi oluşturulmuştur. Daha sonra metin sınıflandırma algoritmalarından Naive Bayes, DVM, KNN ve karar ağacı algoritmaları kullanılarak sınıflandırılmıştır. Çalışma sonucunda en yüksek doğrulama oranına Naive Bayes algoritmasında ulaşılmıştır.

Tablo 2.1: Metin Sınıflandırma İle İlgili Çalışmalar

Yıl	Yazarlar	Çalışmanın Adı	Özellik Vektörü	Sınıflandırma Yöntemi	En başarılı Sonuç
2006	Türkoğlu	Melez yaklaşımlarla Türkçe dokümanlarda yazar tanıma	-	Naive Bayes, DVM, Rastgele Orman, KNN, Çok Katmanlı Algılayıcı, Öz Düzenleyici Özellik Haritası	Çok Katmanlı Algılayıcı (MLP)
2007	Pilavcılar	Metin madenciliği ile metin sınıflandırma	Bit, Kelime Frekans, TF-IDF	Naive Bayes, KNN	TF-IDF ağırlıklandırma kullanan Naive Bayes
2009	Gürcan	Web içerik madenciliği ve konu sınıflandırılması	Bit, Kelime frekansı	Naive Bayes, Naive Bayes multinominal DVM, KNN, Karar Ağacı(J48)	Naive Bayes multinominal
2010	Sriram, vd.	Short text classification in twitter to improve information filtering	BOW	Naive Bayes	Naive Bayes
2013	Çalış, Gazdağı, Yıldız	Reklam içerikli epostaların metin madenciliği yöntemleri ile otomatik tespiti	Binary, Frekans, TF-IDF	DVM, KNN, Naive Bayes	Binary vektör kullanan KNN
2016	Al-Momani, Alwada'n, Mohammad	Arabic text categorization using support vector machine, Naive Bayes and neural network	TF-IDF	DVM, Naive Bayes, MLP	MLP
2016	Borandağ, vd.	KNN algoritması ve R programlama dili ile metin madenciliği kullanılarak bilimsel makale tasnifi	TF-IDF	KNN, Naive Bayes, J48	KNN
2016	Kaynar, vd.	Makine öğrenmesi yöntemleri ile duygu analizi	TF-IDF	Naive Bayes, MLP, DVM	MLP, DVM
2017	Göker ve Tekedere	Fatih Projesine yönelik metin madenciliği yöntemleriyle değerlendirilmesi	TF-IDF	Naive Bayes, KNN, Karar Ağacı, SMO ve RBF	SMO
2017	Gürcan	Denetimli öğrenme yordamlarıyla Türkçe metinler üzerinde konu sınıflandırma analizi	TF-IDF	Naive Bayes, KNN, DVM	Naive Bayes
2018	Alghoul, vd.	Email classification using artificial neural network	-	YSA	YSA

Tablo 2.1 (Devam): Metin Sınıflandırma İle İlgili Çalışmalar

Yıl	Yazarlar	Çalışmanın Adı	Özellik Vektörü	Sınıflandırma Yöntemi	En başarılı Sonuç
2019	Campos, Silva, Bernardino	Text mining in hotel reviews impact of words restriction in text classification	TF-IDF	Naive Bayes, Karar ağaçları, Rastgele orman, DVM	
2019	Erkaya	Text classification based on organizational data using machine learning	-	Naive Bayes, DVM, rastgele orman, lojistik regresyon, LSTM	LSTM
2019	Işık, M.	Müşteri yorumları üzerinde metin analitiği çalışmaları ve yorumların makine öğrenmesi algoritmaları ile modellenmesi	Count, TF-IDF	Naive Bayes, Lojistik regresyon, RBF, DVM	TF-IDF kullanan Lojistik regresyon
2019	İnan	Otel yorumlarının sınıflandırılmasında makine öğrenmesi algoritmalarının karşılaştırılması ve Eskişehir otellerine yönelik yorumların duygu analizi	TF-IDF	Naive Bayes, lojistik regresyon, DVM, KNN	Lojistik regresyon
2019	Öztürk	Cyberbullying detection using text classification for turkish language	TF-IDF	Naive Bayes, DVM, karar ağacı-J48, rastgele orman ve kendisi tarafından öne sürülen filtre tabanlı sınıflandırıcı	Naive Bayes
2020	Ahmetoğlu ve Daş	Türkçe otel yorumlarıyla eğitilen kelime vektörü modellerinin duygu analizi	Word2vec, rastgele kelime vektörü	Derin öğrenme modeli (GRU)	Rastgele kelime vektörü ile sınıflandırılan GRU modeli
2020	Alıcı, vd.	Metin sınıflandırmada yapay sinir ağları ile bitcoin fiyatları ve sosyal medyadaki beklentilerin analizi	-	Naive bayes, DVM, Lojistik Regresyon, Yapay sinir ağları	Yapay sinir ağları
2020	Emekli ve Selvi	GSM operatörlerine yönelik atılan Türkçe tweetlerin derin öğrenme yöntemleriyle duygu analizi	Python Keras Tokenizer kütüphanesi	Evrişimsel sinir ağı, Tekrarlayan, Uzun süreli sinir ağı (derin öğrenme modelleri)	Uzun kısa süreli bellek modeli



ÜÇÜNCÜ BÖLÜM

YÖNTEM

3.1. MATERYAL, YÖNTEM VE UYGULAMA

3.1.1. PHP Programlama Dili

HTML içine gömülebilen bir programlama dilidir. PHP kodu `<?php ?>` etiketleri arasına yazılır. Sunucu tarafında çalıştırılabilir. Böylece siteye bağlanan kullanıcılar kodu göremez ve müdahale edemezler. Yalnızca uygulama kısmını görebilirler (What is PHP?, 2021).

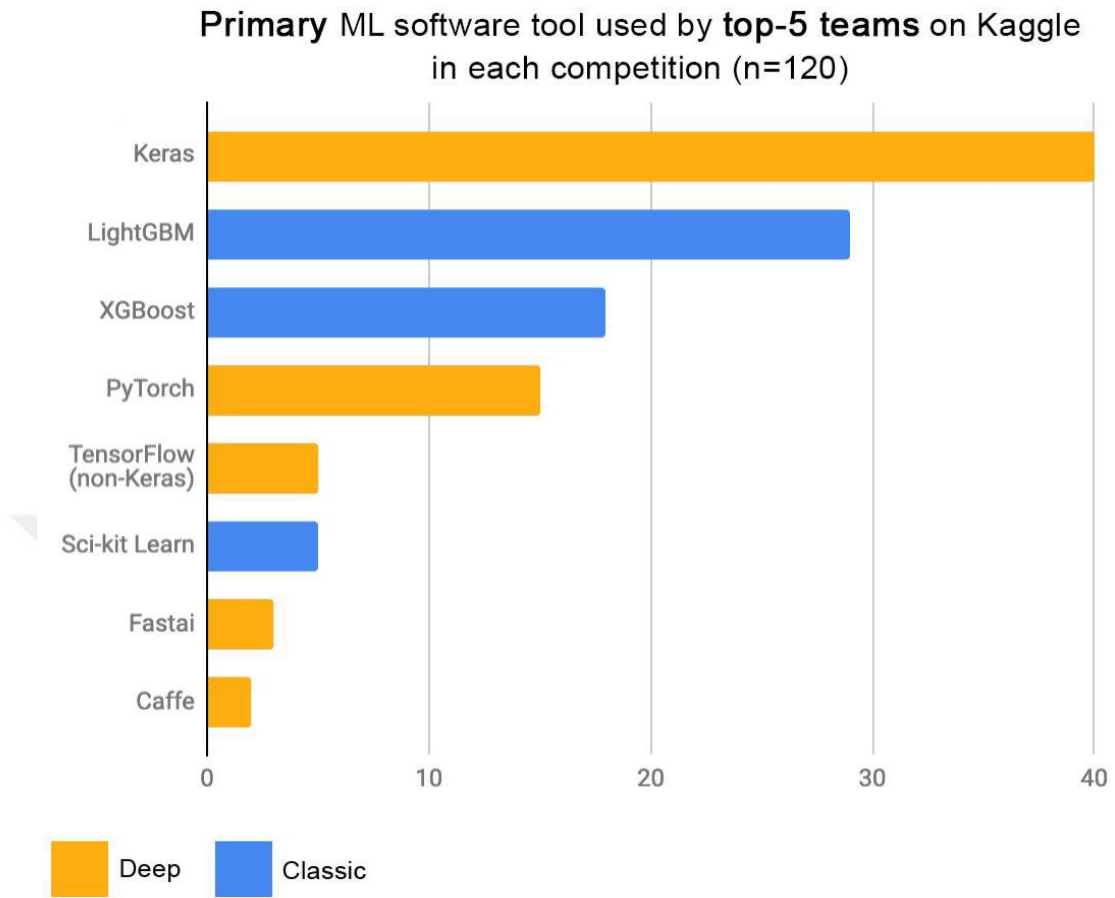
3.1.2. Python Programlama Dili

Python programlama dili, öğrenmesi kolay bir programlama dilidir. Yüksek düzeyli veri yapıları ve nesne tabanlı programlama uygulamalarında etkilidir. Çoğu platformda komut dosyası oluşturma ve hızlı uygulama geliştirme için ideal bir dildir (The Python Tutorial, 2021).

3.1.3. Keras

Bu tez çalışması kapsamında, YSA modeli Python'da yazılmış makine öğrenimi platformu olan Keras kütüphanesi kullanılarak oluşturuldu. 2019 yılında yapılan bir yarışmada en başarılı model olmuştur.

Şekil 3.1: Yarışmada İlk Beş Takım Tarafından Kullanılan Modeller



Kaynak: Why Choose Keras?, 2021

Keras, CERN ve NASA gibi büyük kuruluşlardaki araştırmacılar tarafından da benimsenmiştir. Ayrıca Netflix, Uber gibi uygulamalarda da kullanılmaktadır (Why Choose Keras?, 2021).

3.1.4. Verilerin Toplanması

Popüler bir otel sitesinden Türkiye’deki otellere yönelik yapılan yorumlar ele alındı. Yorumlar ve ilgili yoruma ait müşteri puanları, PHP dilinde yazılmış olan bir web servis sağlayıcısı ile ilgili siteden alınarak MYSQL veritabanına kaydedildi.

Web servis yoluyla müşterilerin ilgili otele yönelik yorum ve puanları alındı. Duygu analizi yapılan çalışmada 4 ve 5 puanlı yorumlar incelendiğinde genel olarak müşterinin verilen hizmetten memnun olduğunu göstermektedir. 1 ve 2 puanlı yorumlar

incelendiğinde ise müşterilerin hizmetten memnun kalmadığı, berbat, kötü gibi terimler kullandığı görülmektedir. 3 puan verilen yorumlarda ise müşterilerin hem olumlu hem olumsuz cümleler kurduğu, bazı yönleriyle otelin eksik ya da pahalı olduğu vs. ama bazı yönleri ile de konumu vs. iyi olduğu şeklinde cümleler kurduğu, müşterilerin arada kaldığı görülmektedir. Çalışmada 3 puanlı yorumlarda müşterilerin memnun kalıp kalmadığı anlaşılmadığından bu tez çalışmasına dahil edilmemiştir. 1 ve 2 puanlı yorumlarda müşterilerin hizmetten memnun kalmadığı kullandıkları kelimelerden net bir şekilde anlaşılmaktadır. Bu nedenle de bu yorumlar olumsuz yorum olarak ele alınmıştır. 4 ve 5 puanlı yorumlarda ise müşterilerin verilen hizmetten memnun olduğu görüldüğünden çalışmada olumlu yorum olarak ele alınmıştır.

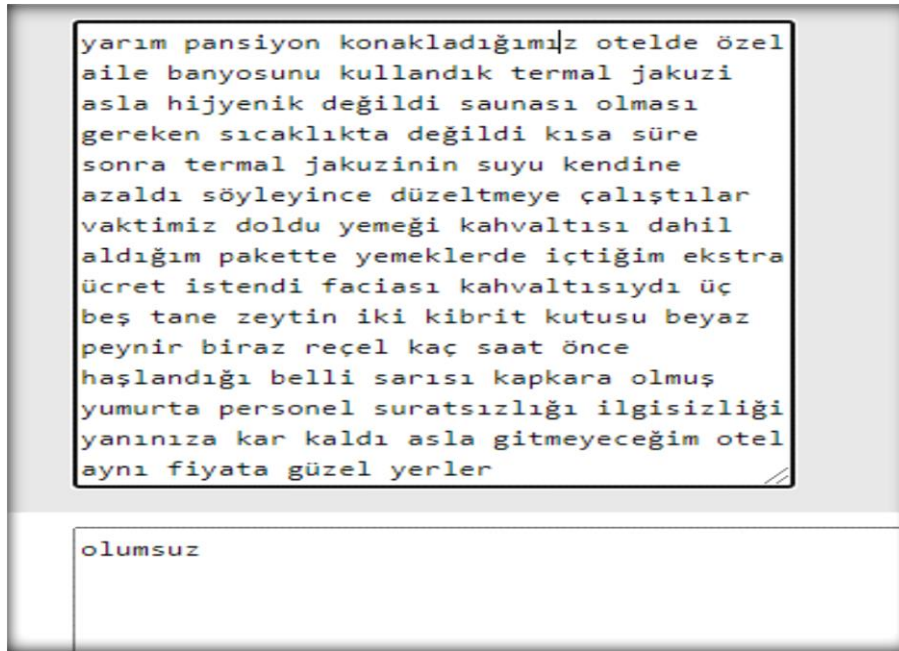
Şekil 3.2: Müşteri Yorum ve Durumlarının PHP İle İlgili Siteden Çekilmesi

```
preg_match_all('#<q class="IRsGHOpm"><span>(.*?)</span></q>#', $siteVerisi,$cikti);
preg_match_all('#<div class="nf9vGX55" data-test-target="review-rating">(.*?)</div>#', $siteVerisi,$cikti2);
/*echo "<pre>";
print_r($cikti[0]);
echo "</pre>"; */
for($j=0;$j<count($cikti[0]);$j++) {
    $yorum=trim($cikti[0][$j], '<q class="IRsGHOpm"><span>');
    $yorum=trim($cikti[0][$j], '</span></q>');
    $yorum=ltrim($cikti[0][$j], 'class="IRsGHOpm">');
    if(strpos($cikti2[0][$j], "bubble_10") || strpos($cikti2[0][$j], "bubble_20")) $durum="olumsuz";
    if(strpos($cikti2[0][$j], "bubble_40") || strpos($cikti2[0][$j], "bubble_50")) $durum="olumlu";
    if(isset($durum) and !empty($durum)){
        echo "<pre>";
        echo $yorum." ----> ". $durum;
        echo "</pre>";
        $conn2->ekle("yorum",array("yorum"=>$yorum,"durum"=>$durum));
    }
    $durum="";
}
```

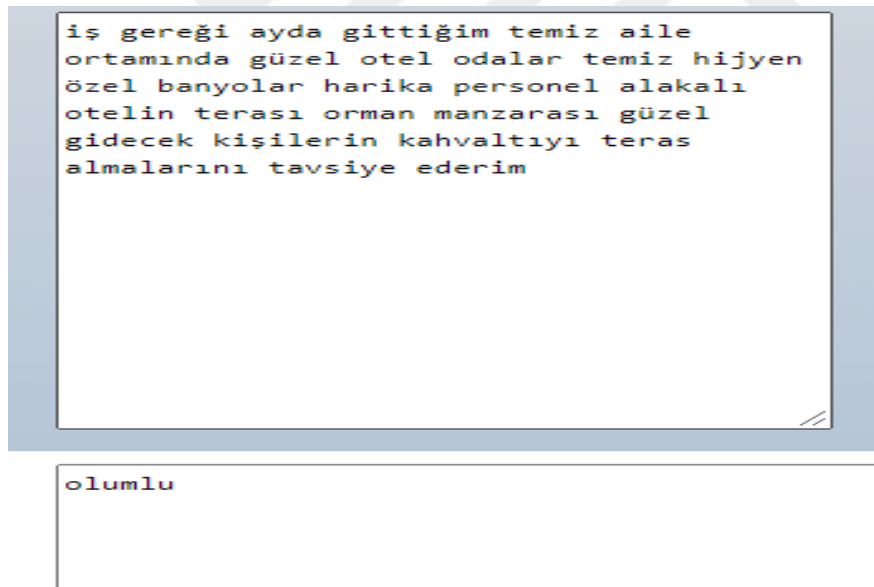
Şekil 3.3: Müşteri Yorum ve Durumlarının Mysql Veritabanında Gösterimi

id	yorum	durum
1	<q class="IRsGHOpm">Kızılcahamam Kaplıca Ter...	olumsuz
2	<q class="IRsGHOpm">Kızılcahamam termal otel...	olumsuz
3	<q class="IRsGHOpm">Heryer pislik içinde rez...	olumsuz
4	<q class="IRsGHOpm">Kızılcahamam a arkadaşla...	olumlu
5	<q class="IRsGHOpm">Çok güzel temiz yeni iyi...	olumlu
6	<q class="IRsGHOpm">Otelin odaları, ortak ku...	olumlu
7	<q class="IRsGHOpm">Biz bayan arkadaşlarla K...	olumlu

Şekil 3.4: Olumsuz Müşteri Yorumu

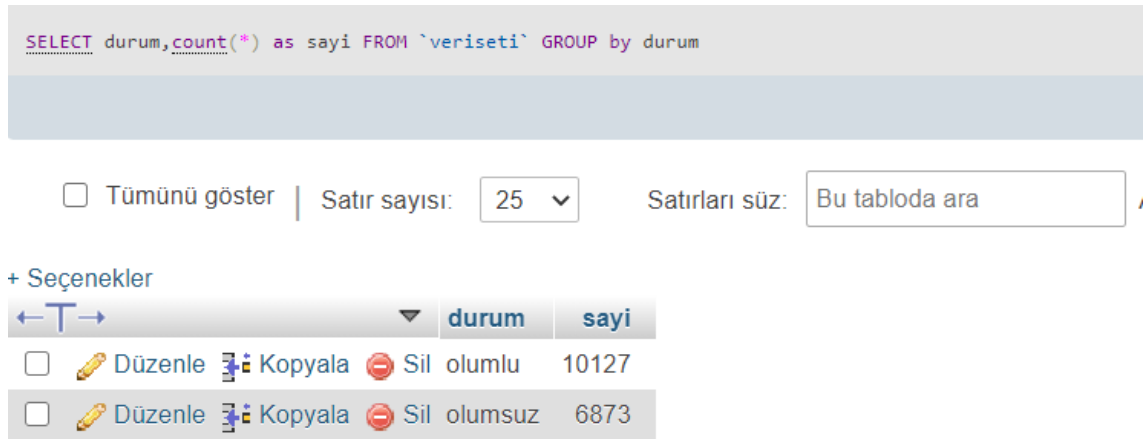


Şekil 3.5: Olumlu Müşteri Yorumu



MYSQL veritabanına, PHP ile dili ile yazılan web servis aracılığıyla ilgili siteden 10127 olumlu yorum, 6873 olumsuz yorum olmak üzere toplam 17000 yorum eklendi.

Şekil 3.6: MYSQL Veritabanında Olumlu ve Olumsuz Yorumların Sayısının Gösterimi



Şekil 3.7: Veri Setindeki Olumlu ve Olumsuz Yorumların Dağılım Grafiği



3.1.5. Veri Seti Üzerinde İşlemler

Veritabanına kaydedilen yorumlar üzerinde temizleme ve düzenleme işlemleri için PHP dilinde `preg_replace`, `str_replace` komutları kullanılmıştır. Müşteri yorumları HTML etiketlerinden ayrıştırılmıştır. Yorumlar içerisinde kullanılan ancak sınıflandırma işleminde anlam ifade etmeyen bağlaç, edat, zarf olan kelimeler ile noktalama işaretleri PHP dili ile yazılan kodla metin içerisinde silinmiştir.

Şekil 3.8: PHP preg_replace Kullanılarak Bağlaç, Edat vs. Veri Setinden Çıkarılması

```
///yorum içindeki bağlaç edat sıl
$yorum=preg_replace("/\bama\b/i","", $yorum);
$yorum=preg_replace("/\bböyle\b/i","", $yorum);
$yorum=preg_replace("/\bdolayısıyla\b/i","", $yorum);
$yorum=preg_replace("/\bher\b/i","", $yorum);
$yorum=preg_replace("/\bki\b/i","", $yorum);
$yorum=preg_replace("/\bolmak\b/i","", $yorum);
$yorum=preg_replace("/\bsadece\b/i","", $yorum);
$yorum=preg_replace("/\byaptığı\b/i","", $yorum);
$yorum=preg_replace("/\bancak\b/i","", $yorum);
$yorum=preg_replace("/\bböylece\b/i","", $yorum);
```

Şekil 3.9: PHP str_replace Kullanılarak Noktalama İşareti, Sayılar, Özel Karakterlerin Veri Setinden Temizlenmesi

```
$yorum=str_replace("/\bay\b/i","", $yorum);
$yorum=str_replace("+","", $yorum); $yorum=str_replace("#","", $yorum);
$yorum=str_replace(".", "", $yorum); $yorum=str_replace("?", "", $yorum);
$yorum=str_replace(")", "", $yorum); $yorum=str_replace(";", "", $yorum);
$yorum=str_replace("=", "", $yorum); $yorum=str_replace("*", "", $yorum);
$yorum=str_replace("/", "", $yorum); $yorum=str_replace("\", "", $yorum);
$yorum=str_replace("1", "", $yorum); $yorum=str_replace("2", "", $yorum);
$yorum=str_replace("5", "", $yorum); $yorum=str_replace("6", "", $yorum);
$yorum=str_replace("9", "", $yorum); $yorum=str_replace("0", "", $yorum);
$yorum=str_replace("□", "", $yorum); $yorum=str_replace("'", "", $yorum);
$yorum=str_replace(">", "", $yorum); $yorum=str_replace("$", "", $yorum);
```

Şekil 3.10: PHP Veri Seti İçindeki Büyük Harflerin Küçük Harfe Dönüştürülmesi

```
$k=array('İ','ı','Ş','Ö','Ğ','Ç','Ü');
$b=array('i','i','ş','ö','ğ','ç','ü');
$yorum=str_replace($k,$b,$yorum);
$yorum = mb_strtolower($yorum,"UTF-8");
```

Şekil 3.11: PHP Dili İle Kelimeler Arasındaki Birden Fazla Boşluğun Silinmesi

```
if(!empty($yorum) and strpos($yorum," ")){
    $parcala=explode(" ", $yorum);
    if(is_array($parcala) and count($parcala)>0){
        $birlestir="";
        for($i=0;$i<count($parcala);$i++){
            $gelen=$parcala[$i];
            if(!empty($gelen) and $gelen!=" " and strlen($gelen)>=3){
                $birlestir.=" ".$gelen;
            }
        }
    }
    $yrm=ltrim($birlestir," ");
}
```

3.1.6. Eğitim ve Test Veri Setini Oluşturma

Bu tez kapsamında, eğitim ve test verileri üç aşamalı olarak oluşturuldu.

Manuel olarak (Elle) oluşturma

Eğitim verisi 8102 olumlu yorum, 5498 olumsuz yorum olmak üzere 13600 yorumdan oluşturuldu. Test verisi 2025 olumlu yorum, 1375 olumsuz yorum olmak üzere toplam 3400 yorumla oluşturuldu. Eğitim ve test verisi içerisindeki olumlu ve olumsuz yorum sayıları belirlenirken olumlu ve olumsuz yorumların yüzde 80'i eğitim, yüzde 20'si test verisi olacak şekilde işleme alınmıştır.

Train_test_split komutu kullanarak oluşturma

Python scikit-learn kütüphanesi içerisinde bir komuttur. Dizileri veya matrisleri rastgele bir noktadan eğitim ve test parçalarına ayırır (Cross-validation: Evaluating Estimator Performance, 2021). Veri seti içindeki verilerin yüzde 20'si test için yüzde 80'i eğitim için ayrıldı.

K-Fold Çarpaz Doğrulama Kullanarak Oluşturma

Train_test_split ile veriler, eğitim ve test verileri olarak ayrıldığında bazı hatalar olabilir. Veri seti ayırımında, her sınıftan veriyi eğitimde görememe ya da test verisinde görememe durumu oluşabilir. Sadece belirli bir sınıftan oluşan verilerle bir model kurulmuş olabilir. Bu durumdan kaçınmak için çarpaz doğrulama kullanılır. Çarpaz doğrulamada, bir k değeri belirlenir. Veri seti k değerine parçalanır. Her iterasyonda k-1 parça eğitim verisi olarak kullanılır. Geriye kalan parça test verisi olarak doğrulama amacı ile kullanılır. K değeri genellikle 5 ya da 10 alınır (Cross-validation: Evaluating Estimator Performance, 2021).

3.1.7. Veri Setinin Sayısallaştırılma İşlemi

3.1.7.1. TF-IDF

Bu vektör modeli, veri setinde geçen kelimelerin sıklıklarını hesaplar. En fazla geçen kelimeler en az TF-IDF değerine sahiptir. Çalışmada, veri temsil yöntemleri arasında kabul edilmiş en uzun tarihe sahip olduğu belirtilen TF-IDF modeli kullanıldı

(Robertson, 2004: Hark, vd., 2019). TF-IDF modelinde, bir kelimenin hem kendi dokümanındaki sıklığına hem de tüm veri seti içindeki kullanım sıklığına bakılır.

Veri setinin TF-IDF vektör modeline dönüştürülmesi işleminde Python dili içerisinde yer alan scikit-learn kütüphanesinden yararlanılmıştır. Bu kütüphane içerisinde yer alan TfidfVectorizer komutu kullanılarak veri seti, sayısal değerlere dönüştürülmüştür.

Şekil 3.12: TfidfVectorizer İşleminde Sonra Veri Setinin Görünümü

```
[ [0. 0. 0. ... 0. 0. 0. ]
  [0. 0. 0. ... 0. 0. 0. ]
  [0. 0. 0. ... 0. 0. 0. ]
  ...
  [0. 0. 0. ... 0. 0. 0. ]
  [0. 0. 0. ... 0.48276788 0. 0. ]
  [0. 0. 0.75254849 ... 0. 0. 0. ]]
```

3.1.7.2. Sınıfları Nümerik Yapma

Bu çalışma kapsamında olumlu ve olumsuz şeklinde iki sınıf kullanılmıştır. Sınıflar, girdi değerlerine karşılık gelen hedef değişkenlerdir. String türünde olan hedef değişkenlerin sayısal değere dönüştürülmesi işleminde Python dili içerisinde bulunan sklearn kütüphanesindeki LabelEncoder komutu kullanılmıştır.

Şekil 3.13: Kategorik Olan Hedef Değişkenin Nümerik Gösterimi

```
[[0. 1.]
 [0. 1.]
 [0. 1.]
 ...
 [0. 1.]
 [0. 1.]
 [0. 1.]
 [0. 1.]
 [0. 1.]
 [0. 1.]]
```

3.1.8. Normalizasyon

Normalizasyon işlemi, eğitimi daha verimli hale getirmek için kullanılır. Veri seti içerisindeki verilerin dağılımının daha düzenli hale gelmesi amaçlanmaktadır. Veri seti içinde çok büyük veya çok küçük değerler olabilir. Hesaplama işlemleri sırasında bu değerler oluşturulan modelin yanlış sonuçlar doğurmasına neden olabilir. Bu nedenle de verilerin değerlerinin belirli bir aralıkta toplanılması yani normalizasyon işlemi yapılır (Deveci ve Yavuz, 2012).

Bu çalışma kapsamında StandartScaler(), RobustScaler(), Min-Max, z-score, karekök normalizasyonları uygulanarak ağın performansı karşılaştırılmıştır.

3.1.8.1. Min-Max

Veri setindeki minimum ve maximum verileri kullanır. Değerleri 0-1 aralığına ölçekleme yapar (Munkhdalai, vd., 2019; Deveci ve Yavuz, 2012).

3.1.8.2. Z-score

Verilerin aritmetik ortalaması ve standart sapması kullanılarak hesaplanır (Jain, Nandakumar ve Ross, 2005).

$$s'_j = \frac{s_j - \min}{\max - \min}, j=1,2,3, \dots, n \text{ (A.g.e).}$$

3.1.8.3. Karekök Normalizasyonu

Verilerin karekökü alınarak hesaplanır.

3.1.8.4. StandartScaler()

Verileri varyans değerine göre ölçeklendirir (scikit-learn.org).

3.1.8.5. RobustScaler()

Verileri nicelik aralığına göre ölçeklendirir. Varsayılan aralık, default=(25.0, 75.0) şeklindedir. (scikit-learn.org).

3.1.9. Sınıflandırma İşlemi

Verilerin sınıflandırması işleminde metin sınıflandırma algoritmalarından biri olan YSA kullanılmıştır. Daha önce metin sınıflandırma alanında yapılan çalışmada da kullanılan farklı metin sınıflandırma algoritmaları içerisinde YSA algoritmasının diğerlerinden daha başarılı bir performans gösterdiği görülmüştür (Alıcı, vd., 2020).

YSA modelinin oluşturulması sırasında Python Keras kütüphanesinde bulunan Sequential() modeli kullanılmıştır. Oluşturulan ağ modelinde, girdi katmanına girdi sayısı kadar nöron eklenilmiştir. YSA modelinde her nöron için ağırlık değerleri atanır ve gerekli hesaplamalar yapıldıktan sonra gizli katmanlara aktivasyon fonksiyonu ile gönderilir (Bati ve Ser, 2019). Katmanlar arasında aktivasyon fonksiyonu relu, çıktı katmanında ise hedef değişken, iki kategoriden oluştuğu için sigmoid fonksiyonu kullanılmıştır. Optimizasyon işleminde ise çalışmalarda en çok tercih edilen adam algoritması kullanılmıştır (Basic Text Classification, 2021).

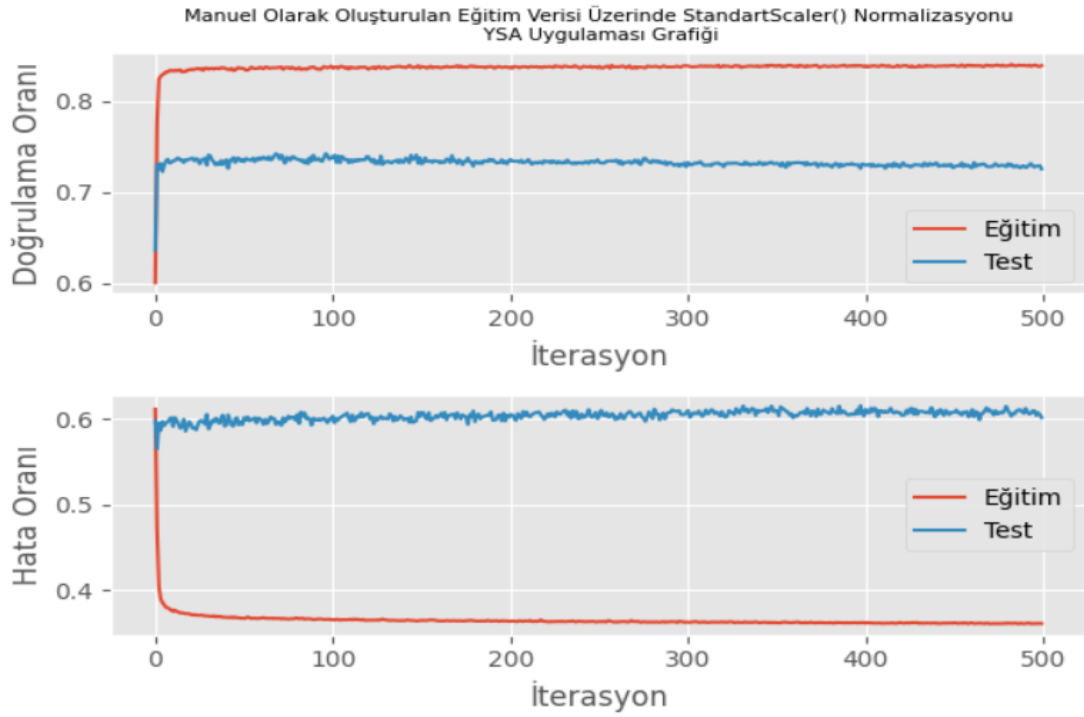
3.2. UYGULAMA ÇIKTILARI

3.2.1. Manuel Olarak Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması

3.2.1.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması

Test doğruluk oranı, modelin eğitilmesi sonucunda ortaya çıkan doğruluk oranından çok daha düşük çıkmıştır. Doğruluk oranları Şekil 3.15'te gösterilmiştir. Sınıflandırmada olumlu ve olumsuz duyguların dağılımına bakıldığında 2125 olumlu yorum ve 1275 olumsuz yorum bulunmaktadır. Ancak test verileri içerisinde gerçekte olan olumlu duygu sayısı 2025, olumsuz duygu sayısı ise 1375 yorumdur.

Şekil 3.14: Modelin Doğruluk-Hata Oranları Grafiği

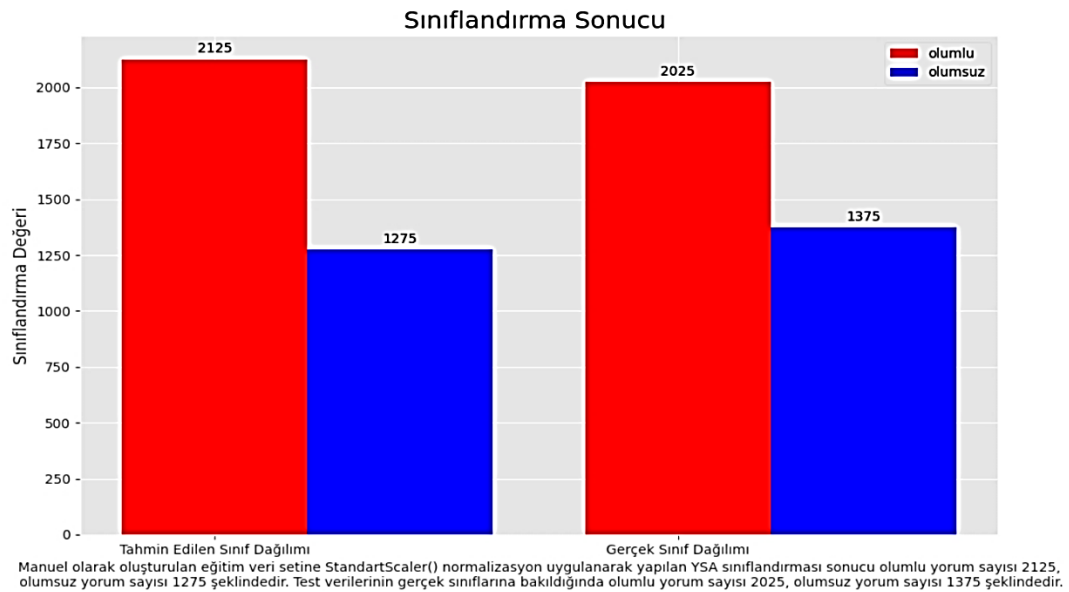


Şekil 3.15: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

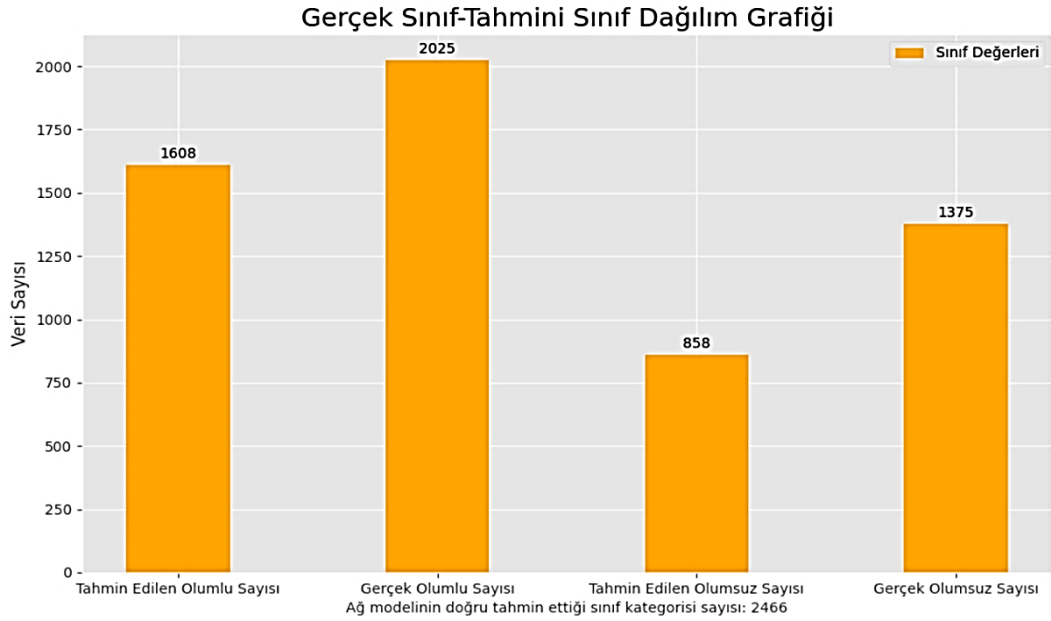
Eğitim Doğruluğu: 0.8399

Test Doğruluğu: 0.7253

Şekil 3.16: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

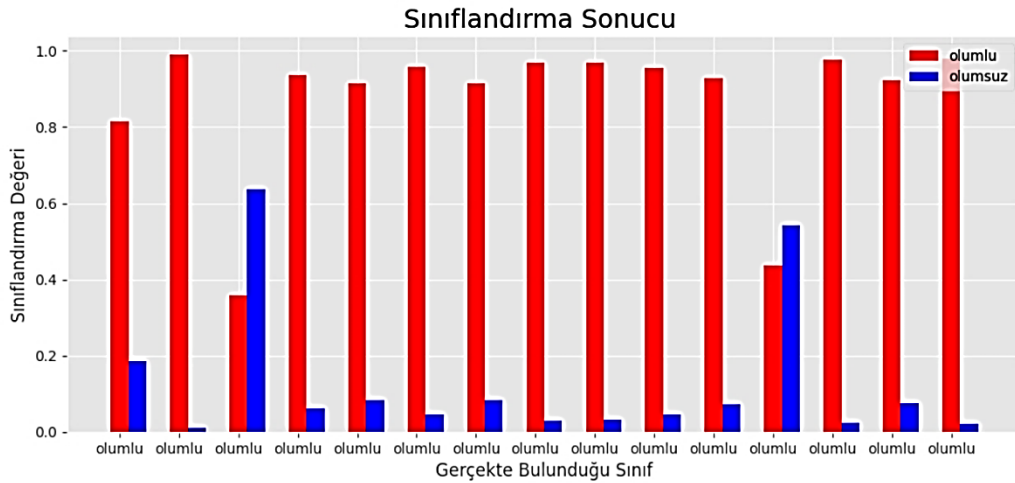


Şekil 3.17: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1608 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan ise 858 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.18’de ise test verisi içerisinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelin tahmin ettiği sınıf değeri gösterilmiştir.

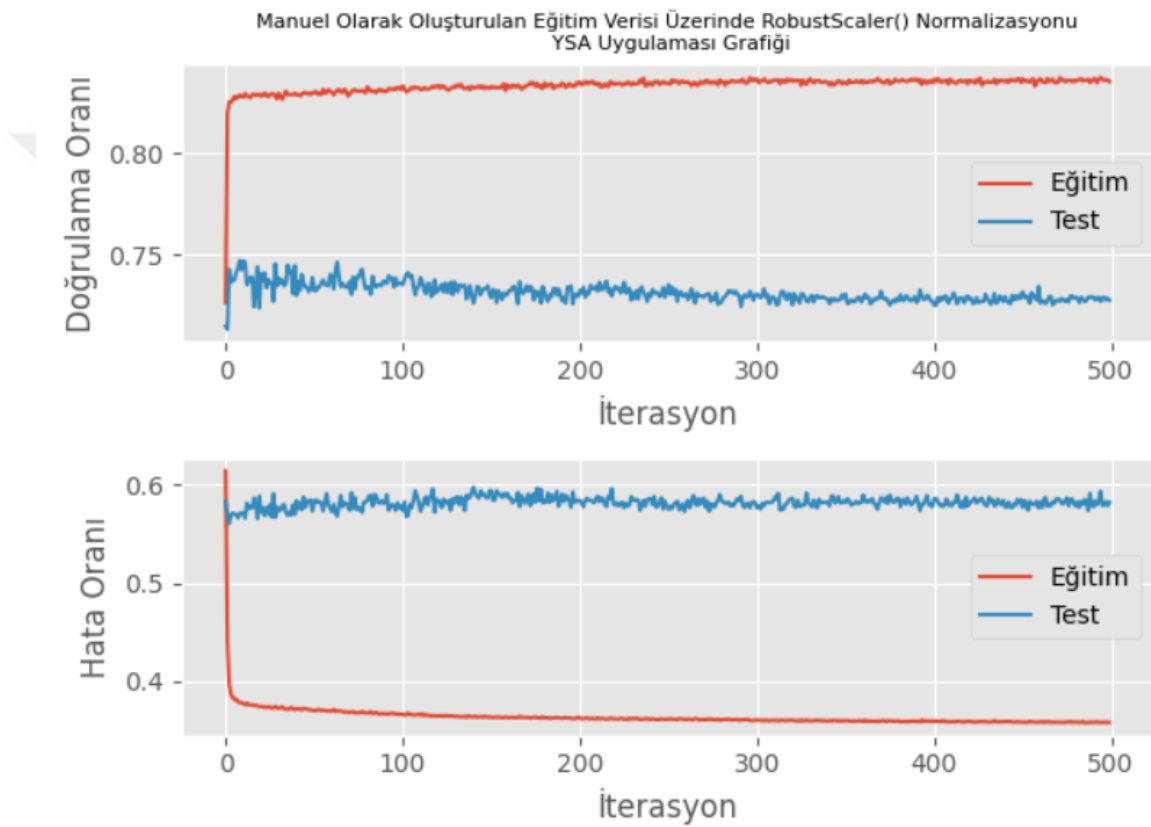
Şekil 3.18: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.1.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması

Test doğruluk oranı, modelin eğitilmesi sonucunda ortaya çıkan doğruluk oranından daha düşük çıkmıştır. Doğruluk oranları Şekil 3.20’de gösterilmiştir. Sınıflandırmada olumlu ve olumsuz duyguların dağılımına bakıldığında 2102 olumlu yorum ve 1298 olumsuz yorum bulunmaktadır. Ancak test verileri içerisinde gerçekte olan olumlu duygu sayısı 2025, olumsuz duygu sayısı ise 1375 yorumdur.

Şekil 3.19: Modelin Doğruluk-Hata Oranları Grafiği

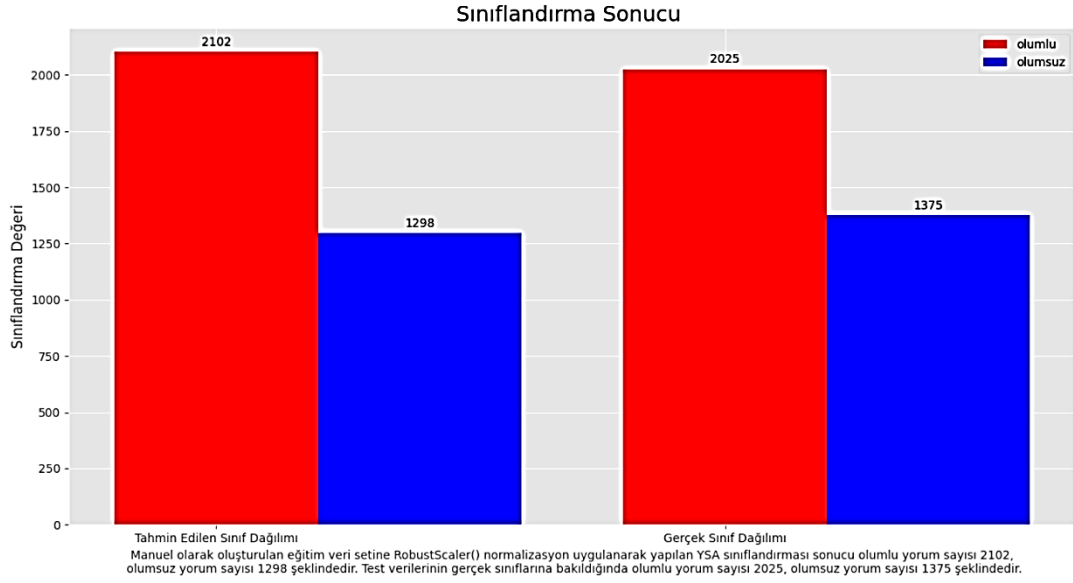


Şekil 3.20: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

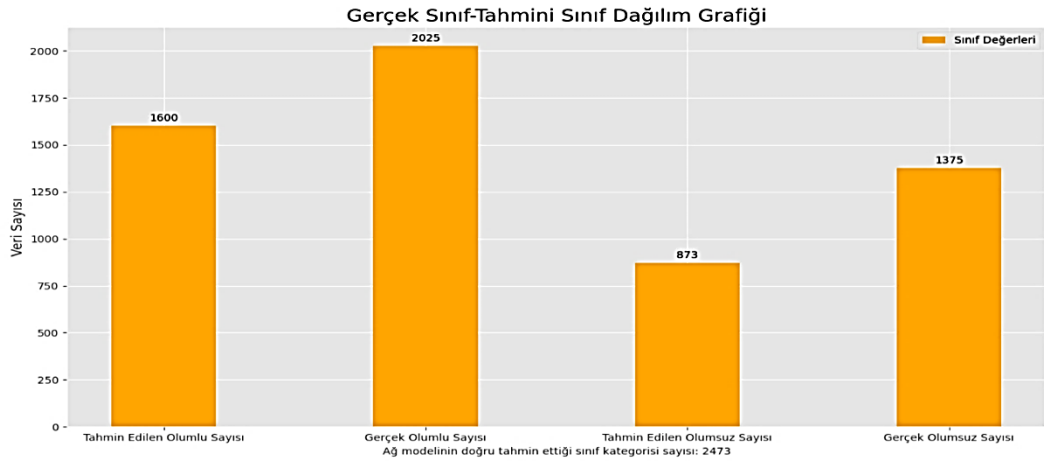
Eğitim Doğruluğu: 0.8375

Test Doğruluğu: 0.7274

Şekil 3.21: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

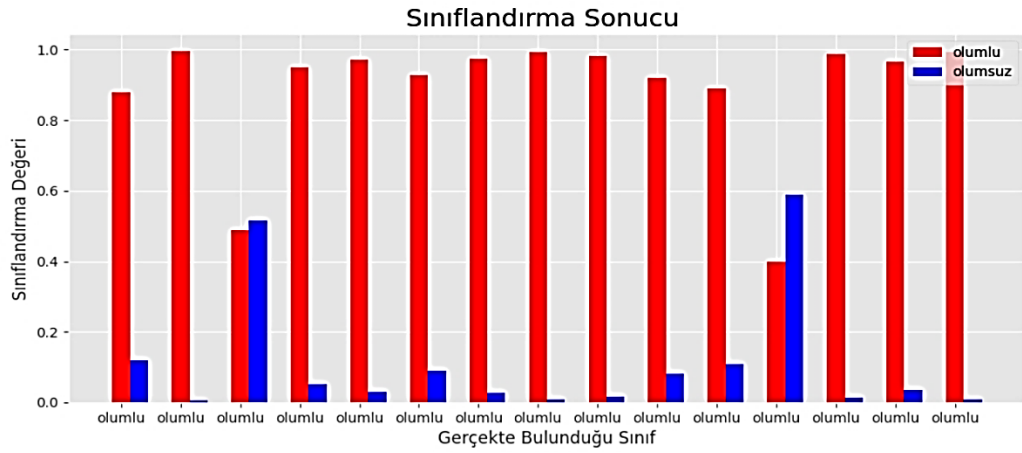


Şekil 3.22: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1600 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan 873 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.23'te ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelin tahmin ettiği sınıf değeri gösterilmiştir.

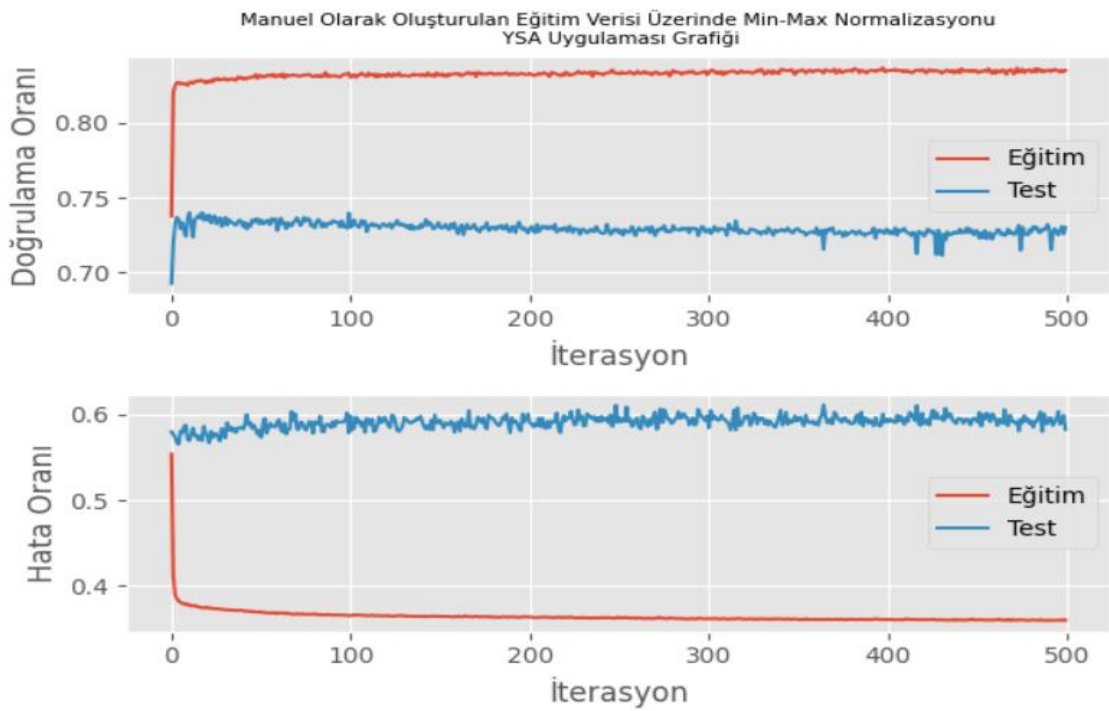
Şekil 3.23: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.1.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması

Test doğruluk oranı, modelin eğitilmesi sonucunda ortaya çıkan doğruluk oranından daha düşük çıkmıştır. Doğruluk oranları Şekil 3.25’de gösterilmiştir. Sınıflandırmada olumlu ve olumsuz duyguların dağılımına bakıldığında 2095 olumlu yorum ve 1305 olumsuz yorum bulunmaktadır. Ancak test verileri içerisinde gerçekte olan olumlu duygu sayısı 2025, olumsuz duygu sayısı ise 1375 yorumdur.

Şekil 3.24: Modelin Doğruluk-Hata Oranları Grafiği

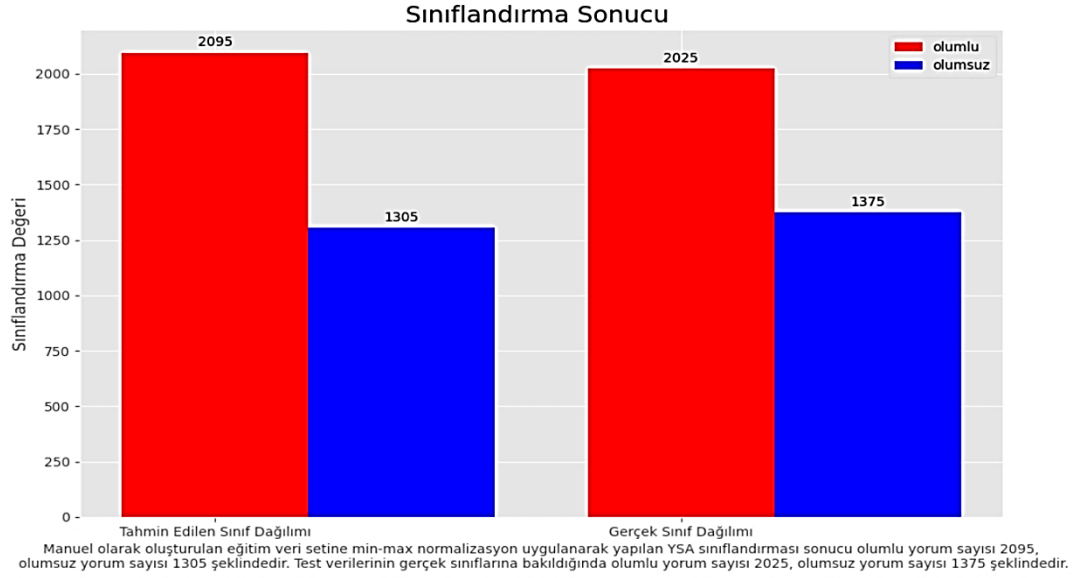


Şekil 3.25: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

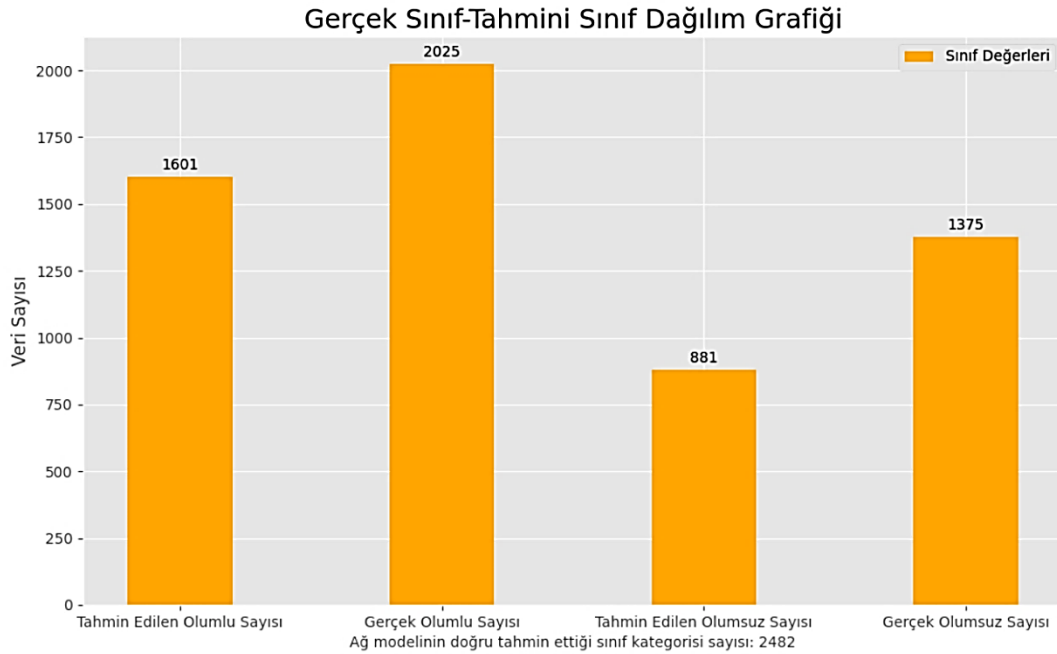
Eğitim Doğruluğu: 0.8354

Test Doğruluğu: 0.7300

Şekil 3.26: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



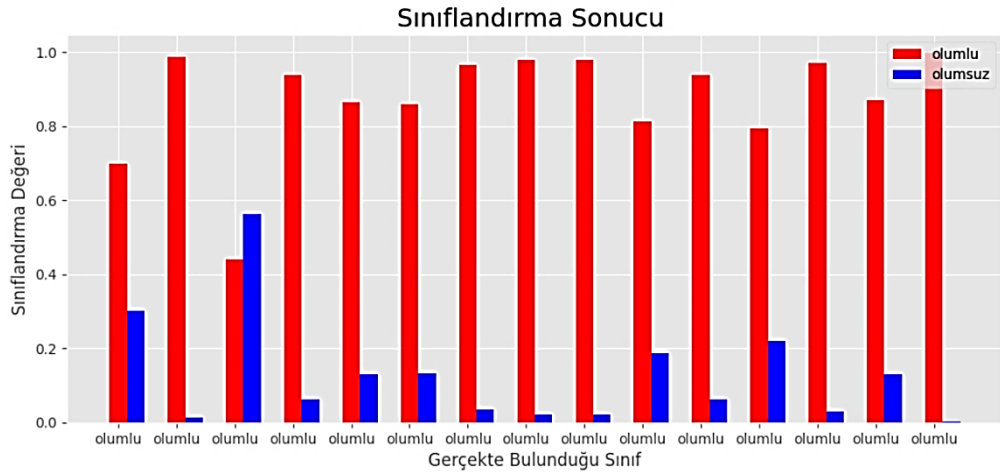
Şekil 3.27: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1601 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan 881 tanesinin sınıfını

doğru tahmin ettiği görülmektedir. Şekil 3.28’de ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelin tahmin ettiği sınıf değeri gösterilmiştir.

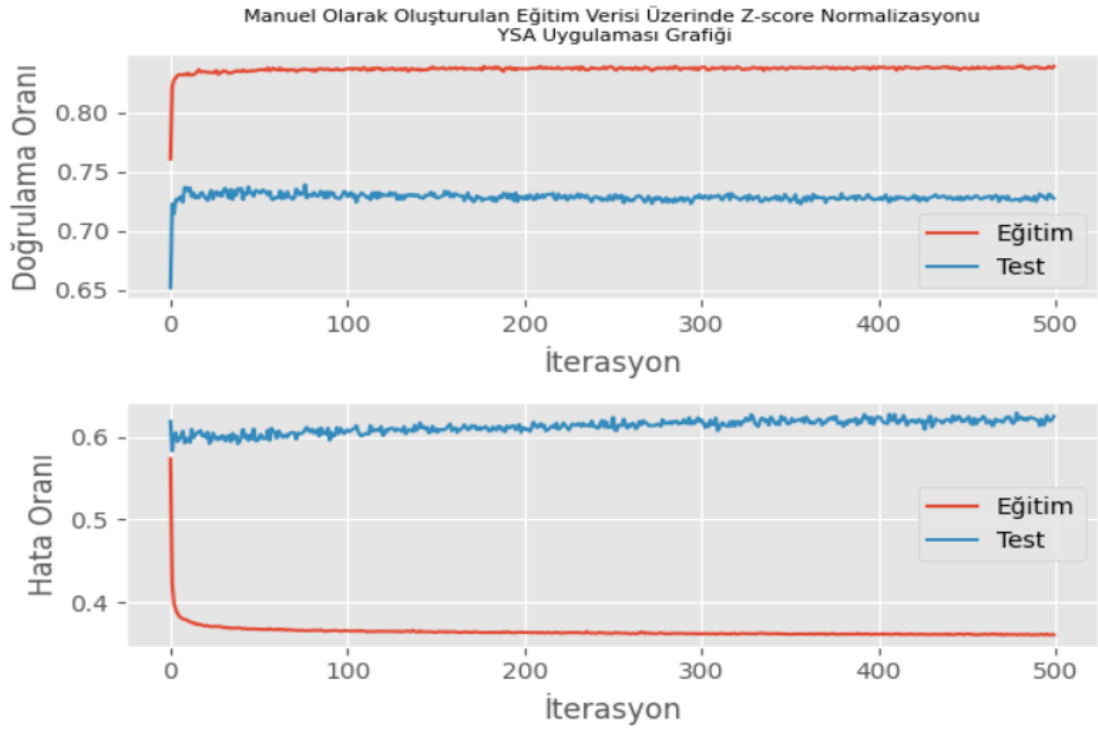
Şekil 3.28: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.1.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması

Test doğruluk oranı, modelin eğitilmesi sonucunda ortaya çıkan doğruluk oranı yaklaşık yüzde 84 iken test doğruluk oranı yüzde 72’ de kalmıştır. Doğruluk oranları Şekil 3.30’da gösterilmiştir. Sınıflandırmada olumlu ve olumsuz duyguların dağılımına bakıldığında 2014 olumlu yorum ve 1386 olumsuz yorum bulunmaktadır. Ancak test verileri içerisinde gerçekte olan olumlu duygu sayısı 2025, olumsuz duygu sayısı ise 1375 yorumdur.

Şekil 3.29: Modelin Doğruluk-Hata Oranları Grafiği

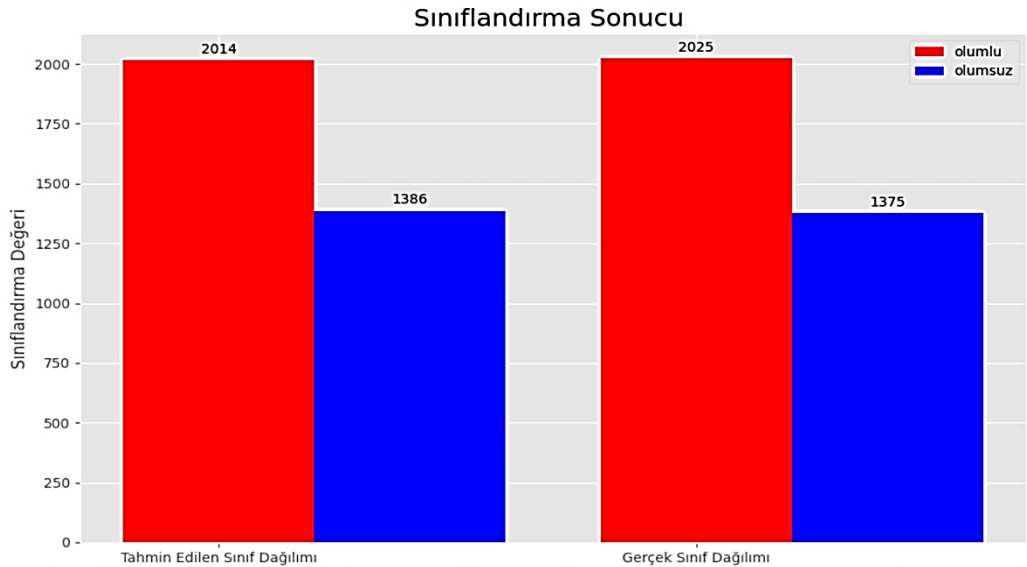


Şekil 3.30: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

Eğitim Doğruluğu: 0.8390

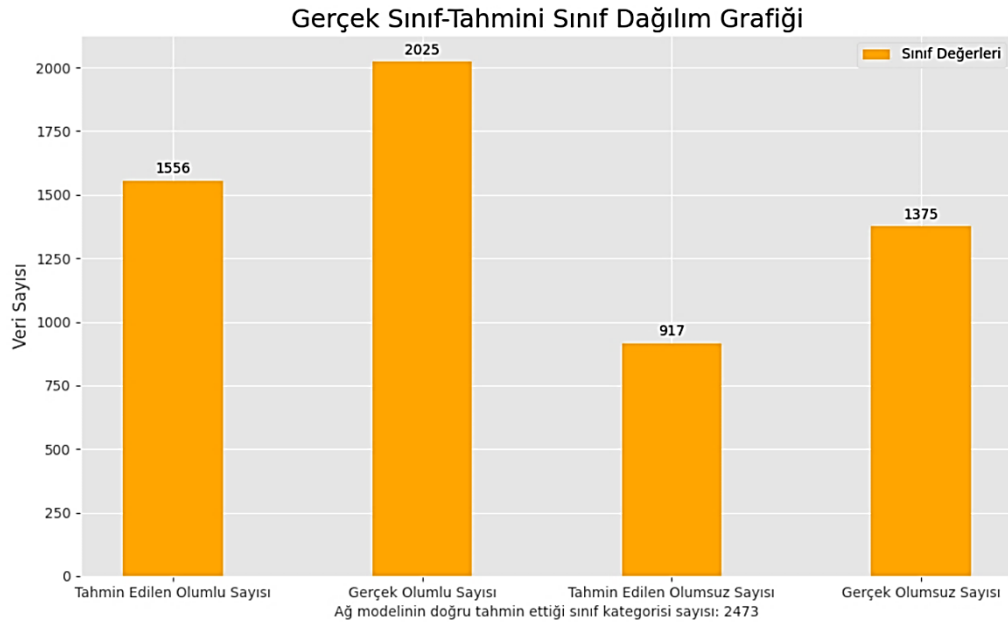
Test Doğruluğu: 0.7274

Şekil 3.31: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



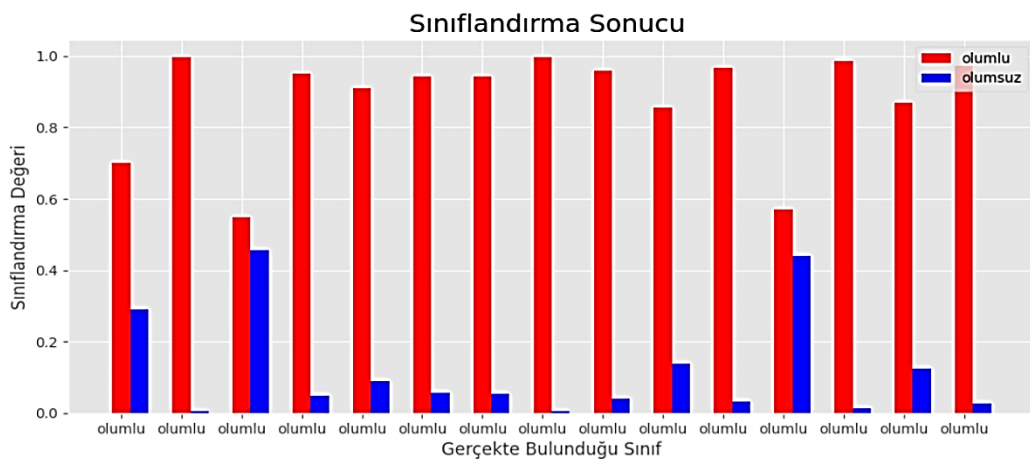
Manuel olarak oluşturulan eğitim veri setine z-score normalizasyon uygulanarak yapılan YSA sınıflandırması sonucu olumlu yorum sayısı 2014, olumsuz yorum sayısı 1386 şeklindedir. Test verilerinin gerçek sınıflarına bakıldığında olumlu yorum sayısı 2025, olumsuz yorum sayısı 1375 şeklindedir.

Şekil 3.32: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1556 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan 917 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.33’de ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelin tahmin ettiği sınıf değeri gösterilmiştir.

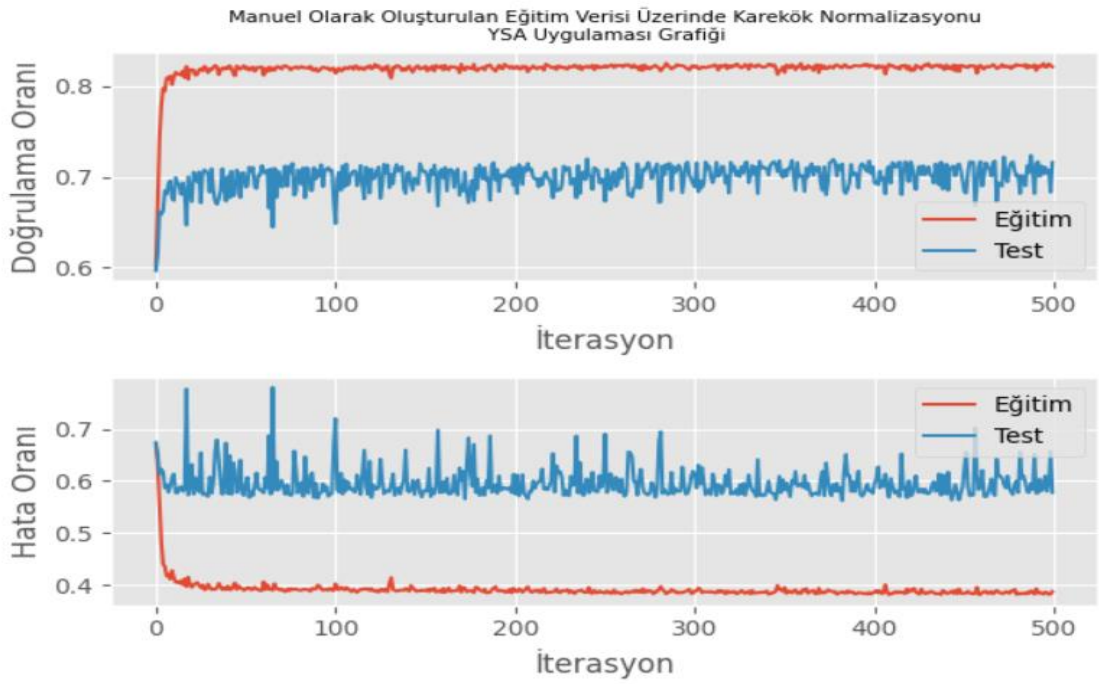
Şekil 3.33: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.1.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması

Test doğruluk oranı, modelin eğitilmesi sonucunda ortaya çıkan doğruluk oranından çok daha düşük çıkmıştır. Doğruluk oranları Şekil 3.35'te gösterilmiştir. Sınıflandırmada olumlu ve olumsuz duyguların dağılımına bakıldığında 2033 olumlu yorum ve 1367 olumsuz yorum bulunmaktadır. Ancak test verileri içerisinde gerçekte olan olumlu duygu sayısı 2025, olumsuz duygu sayısı ise 1375 yorumdur.

Şekil 3.34: Modelin Doğruluk-Hata Oranları Grafiği

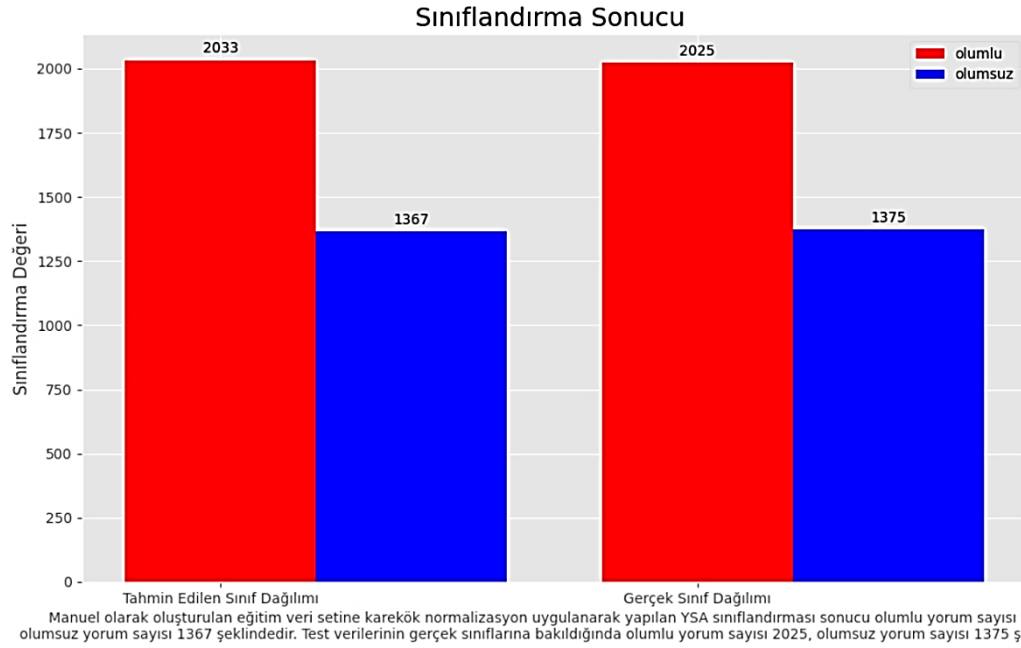


Şekil 3.35: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

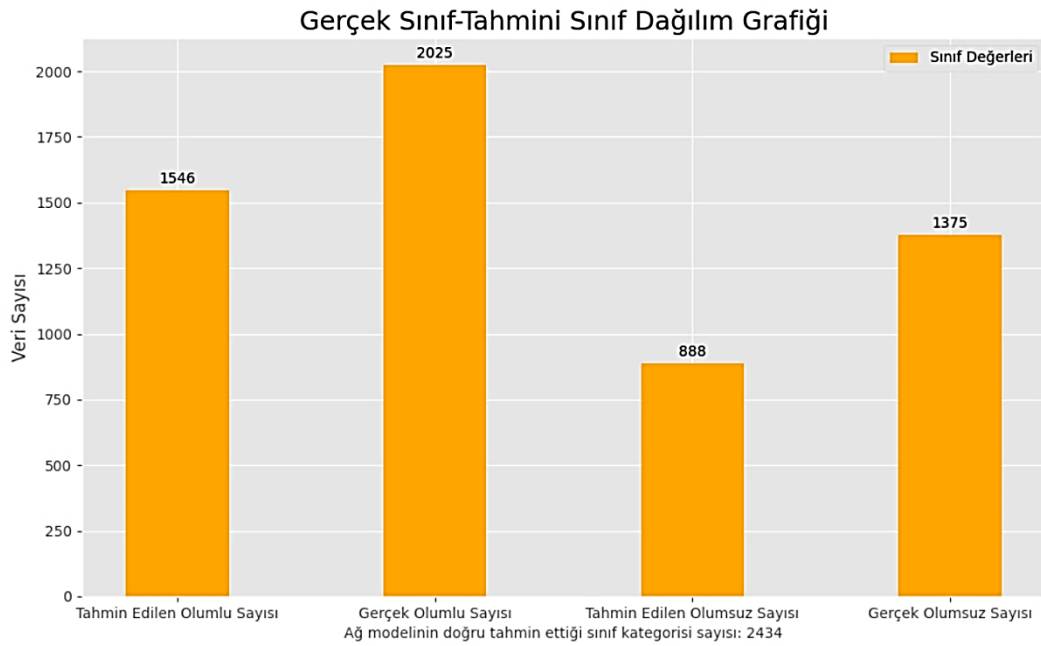
Eğitim Doğruluğu: 0.8210

Test Doğruluğu: 0.7159

Şekil 3.36: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

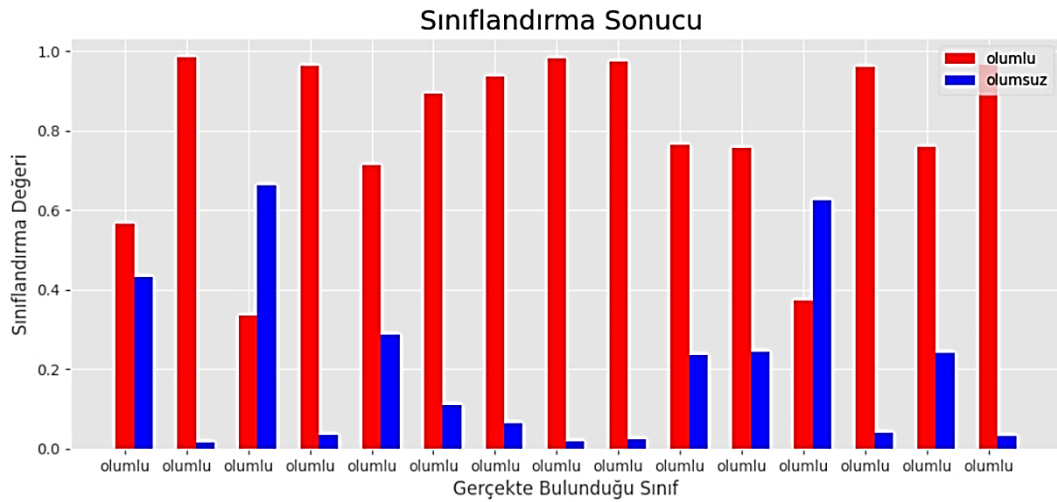


Şekil 3.37: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1546 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan 888 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.38’de ise test verisinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelinin tahmin ettiği sınıf değeri gösterilmiştir.

Şekil 3.38: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı

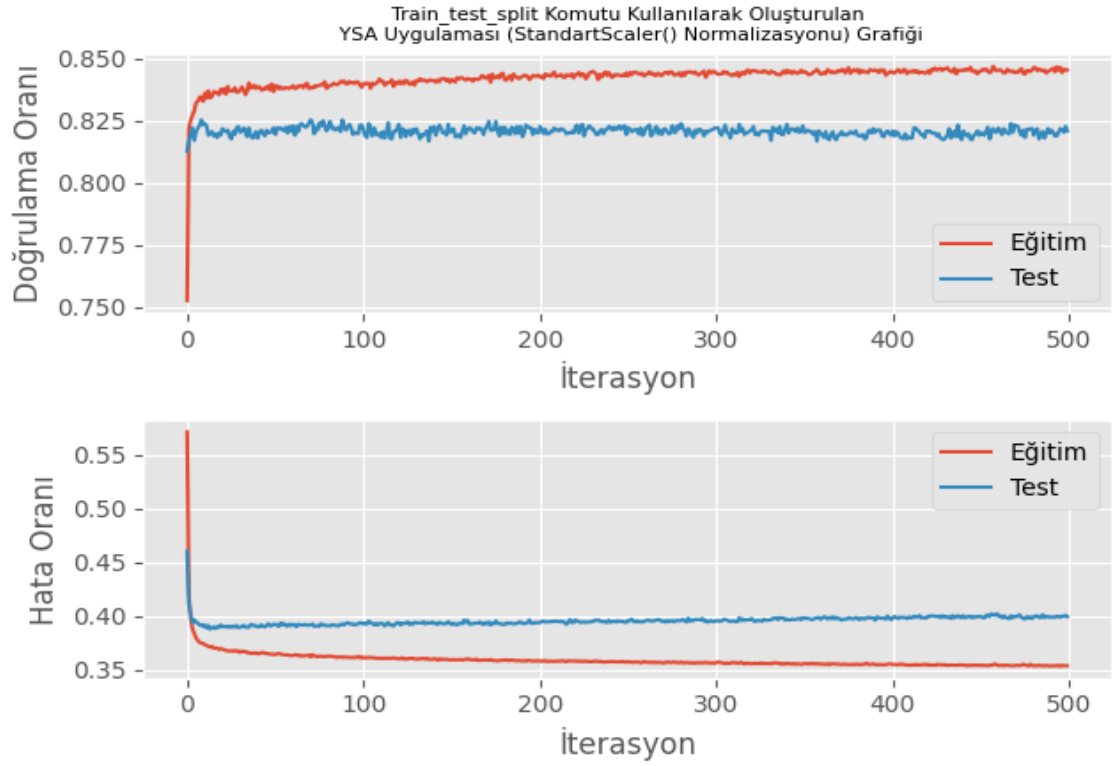


3.2.2. Train_test_split Komutu Kullanılarak Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması

3.2.2.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki StandartScaler() fonksiyonu kullanıldı. Eğitim doğruluğu yüzde 84 iken test doğruluk oranının yüzde 82’de kaldığı görülmüştür. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1961, olumsuz duygu belirten yorum sayısı 1439 olduğu görülmektedir. Ancak gerçekte test verisi incelendiğinde 2042 olumlu yorum, 1358 olumsuz yorum verisi içerdiği görülmektedir.

Şekil 3.39: Modelin Doğruluk-Hata Oranları Grafiği

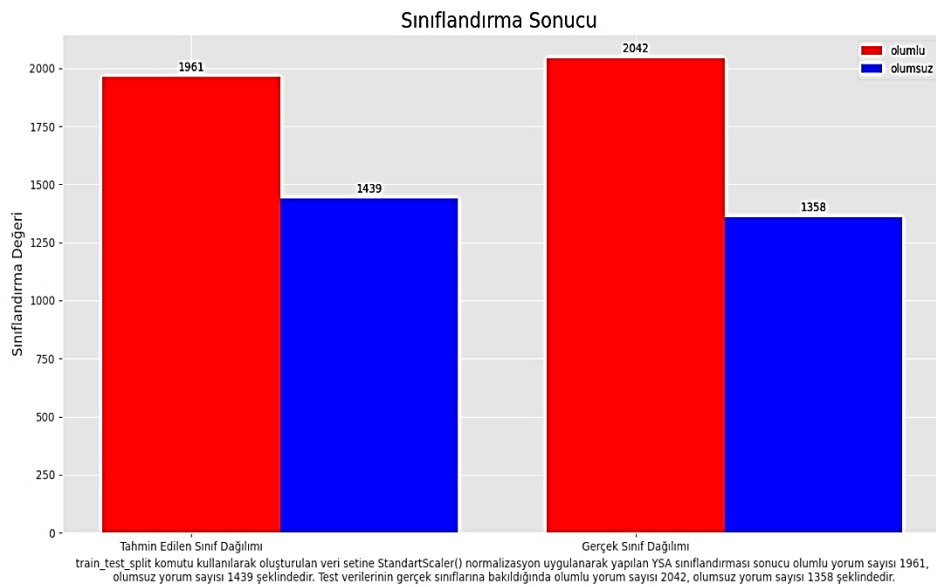


Şekil 3.40: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

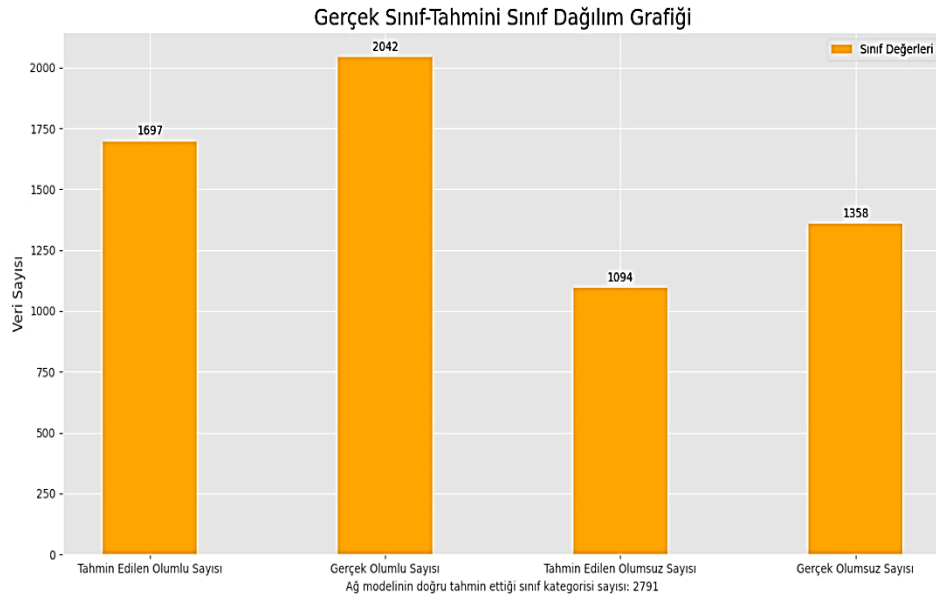
Eğitim Doğruluğu: 0.8465

Test Doğruluğu: 0.8209

Şekil 3.41: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

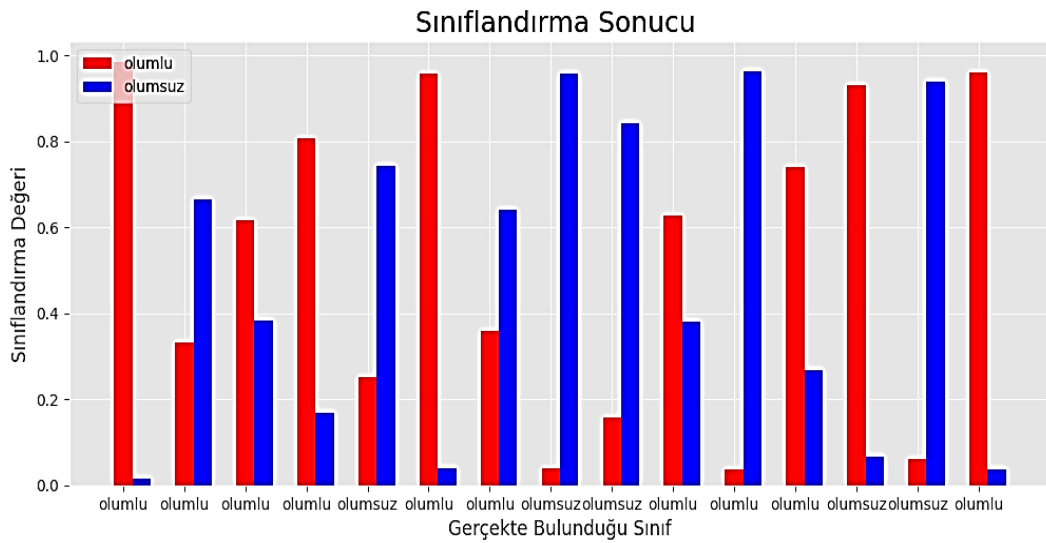


Şekil 3.42: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2042 adet olumlu duygu belirten veriden sadece 1697 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1358 olumsuz yorumdan 1094 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.43'te ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelinin tahmin ettiği sınıf değeri gösterilmiştir.

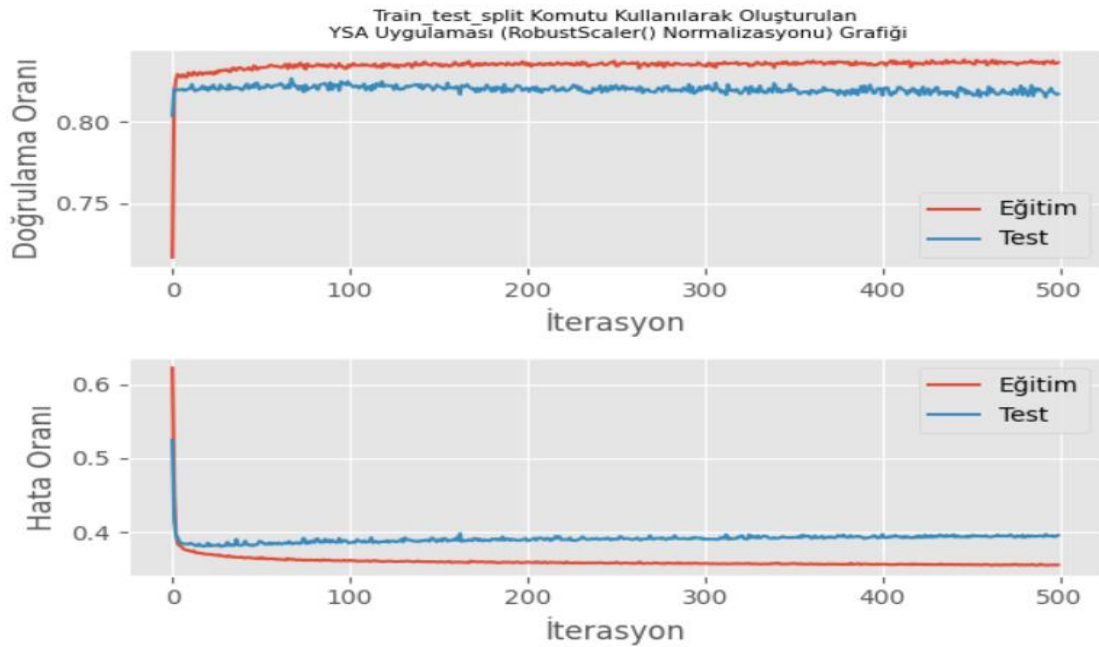
Şekil 3.43: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.2.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki RobustScaler() fonksiyonu kullanıldı. Eğitim doğruluk oranı yaklaşık yüzde 84 iken test doğruluk oranının yüzde 82’de kaldığı görülmüştür. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1979, olumsuz duygu belirten yorum sayısı 1421 olduğu görülmektedir. Ancak gerçekte test verisi incelendiğinde 2042 olumlu yorum, 1358 olumsuz yorum verisi içerdiği görülmektedir.

Şekil 3.44: Modelin Doğruluk-Hata Oranları Grafiği

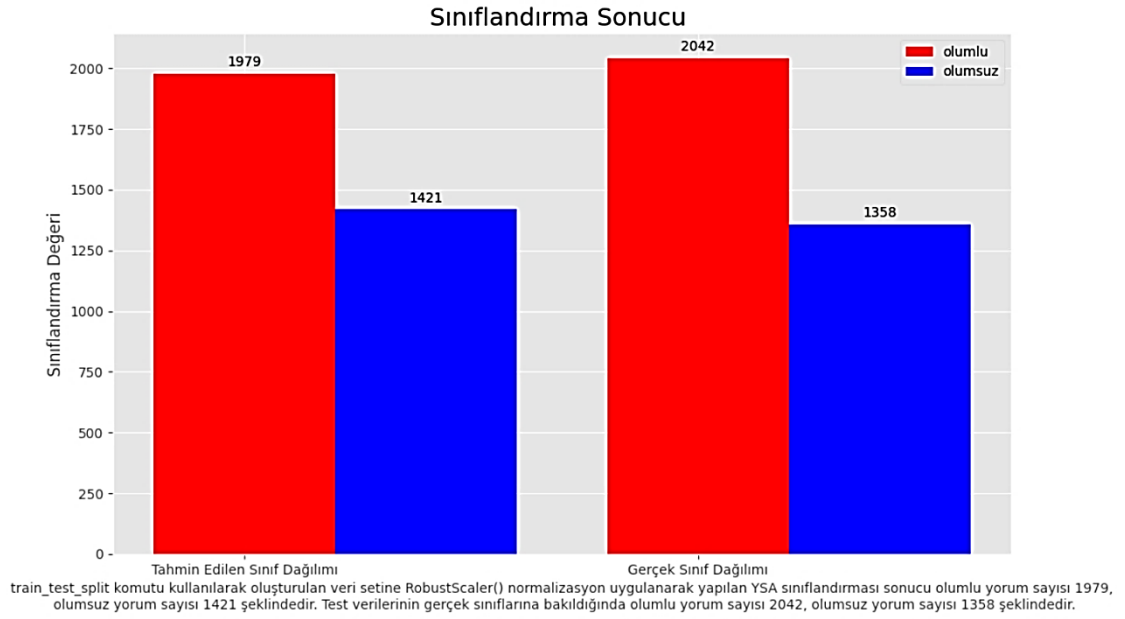


Şekil 3.45: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

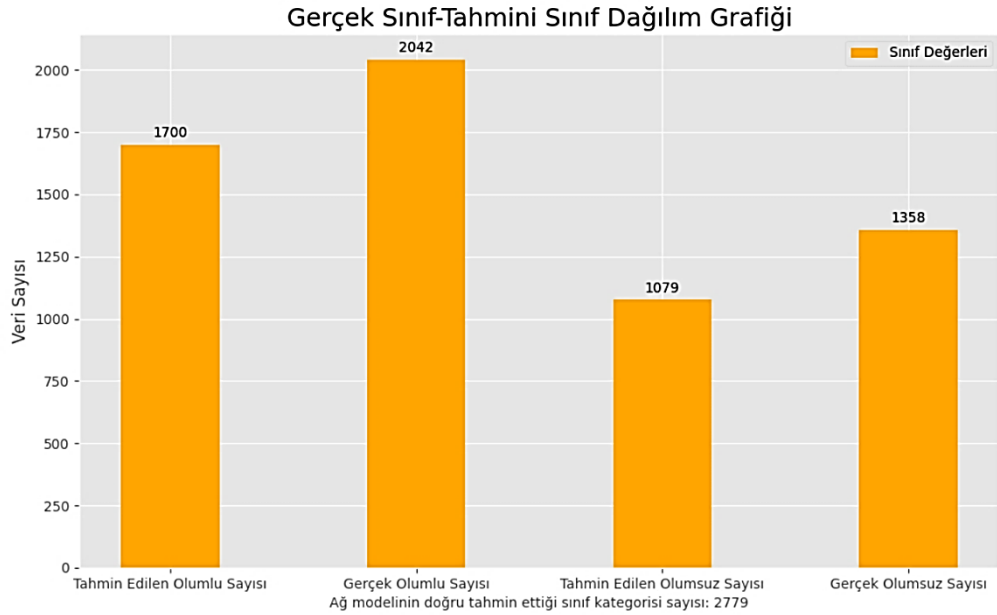
Eğitim Doğruluğu: 0.8385

Test Doğruluğu: 0.8174

Şekil 3.46: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

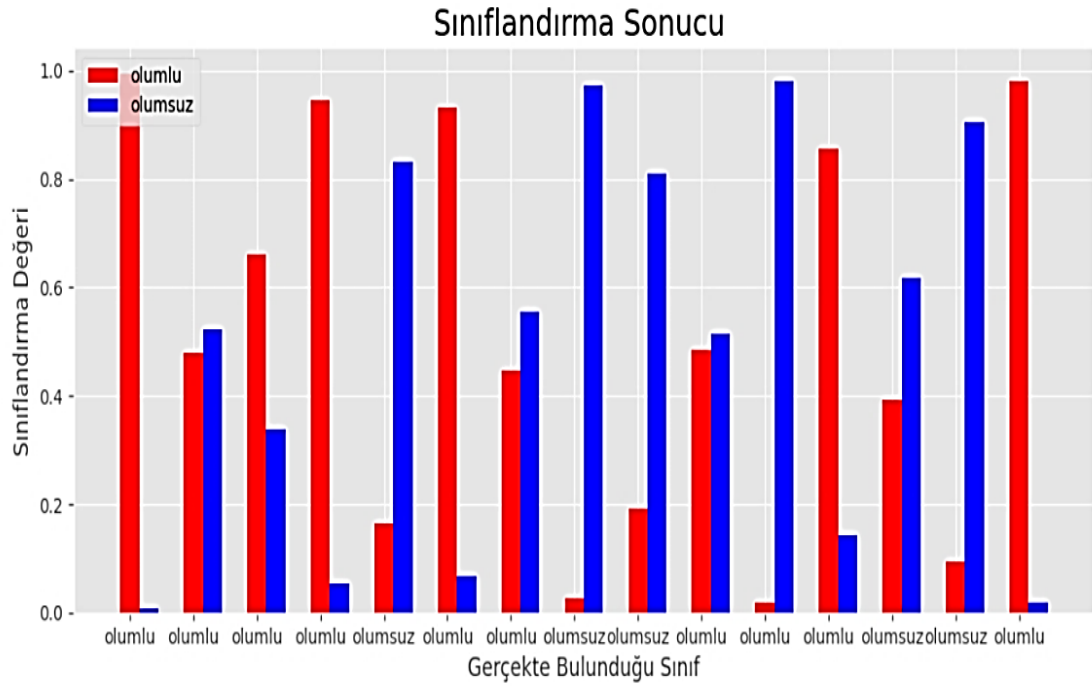


Şekil 3.47: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2042 adet olumlu duygu belirten veriden sadece 1700 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1358 olumsuz yorumdan 1079 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.48’de ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelinin tahmin ettiği sınıf değeri gösterilmiştir.

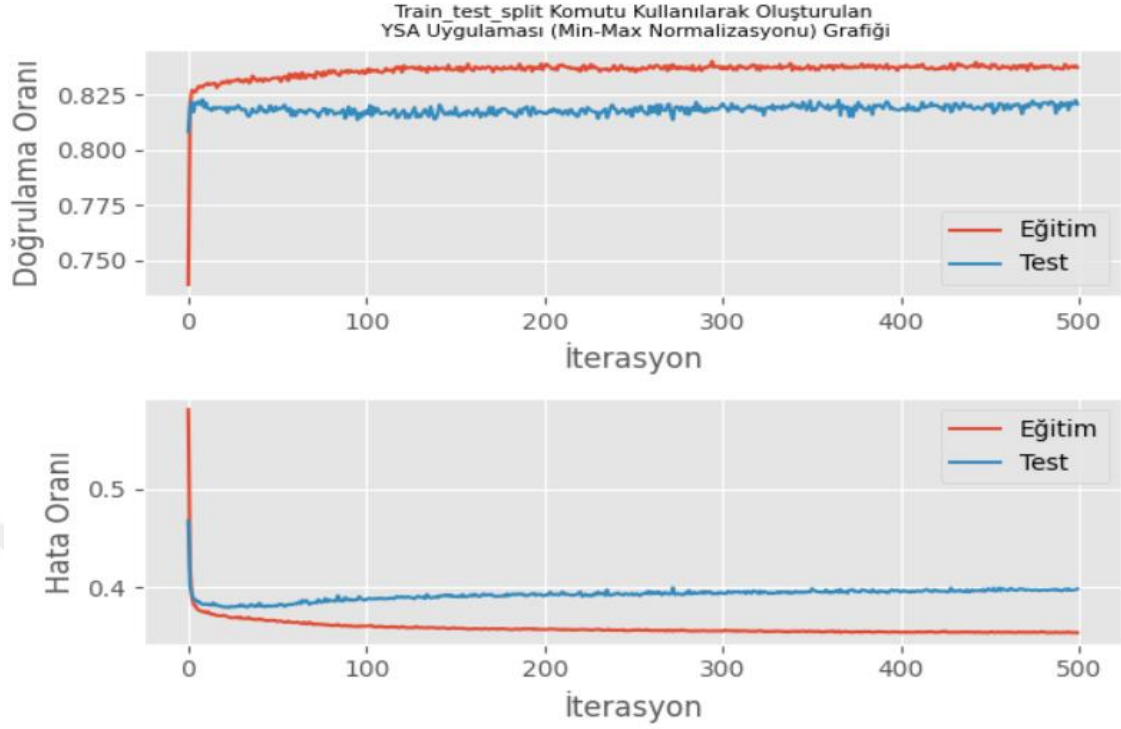
Şekil 3.48: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.2.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki MinMaxScaler() fonksiyonu kullanıldı. Eğitim doğruluğu yüzde 84 iken test doğruluk oranının yüzde 82’de kaldığı görülmüştür. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1946, olumsuz duygu belirten yorum sayısının 1454 olduğu görülmektedir. Ancak gerçekte test verisi incelendiğinde 2042 olumlu yorum, 1358 olumsuz yorum verisi içerdiği görülmektedir.

Şekil 3.49: Modelin Doğruluk-Hata Oranları Grafiği

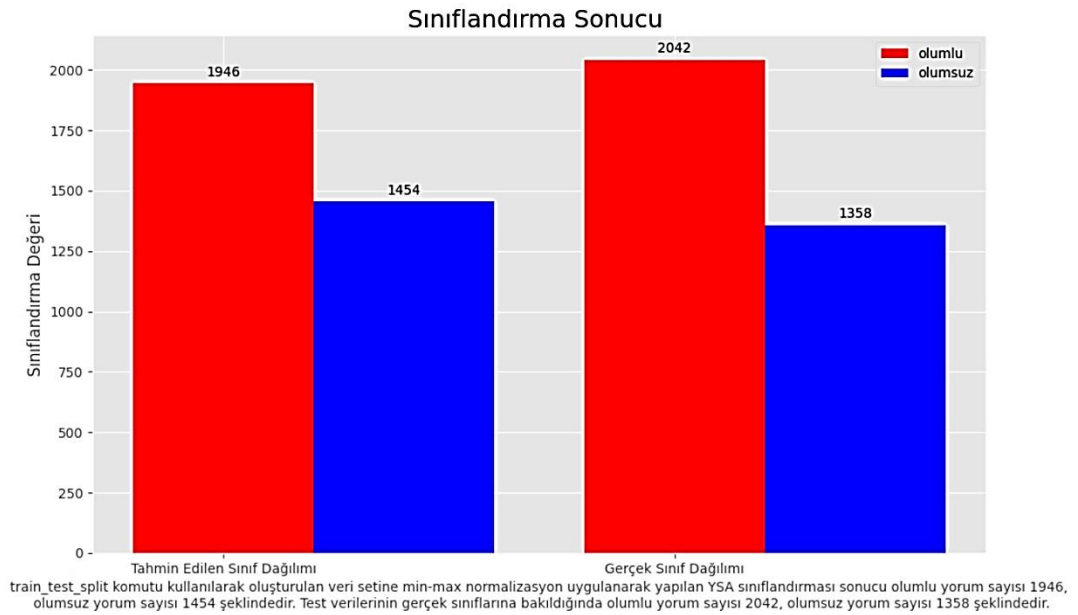


Şekil 3.50: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

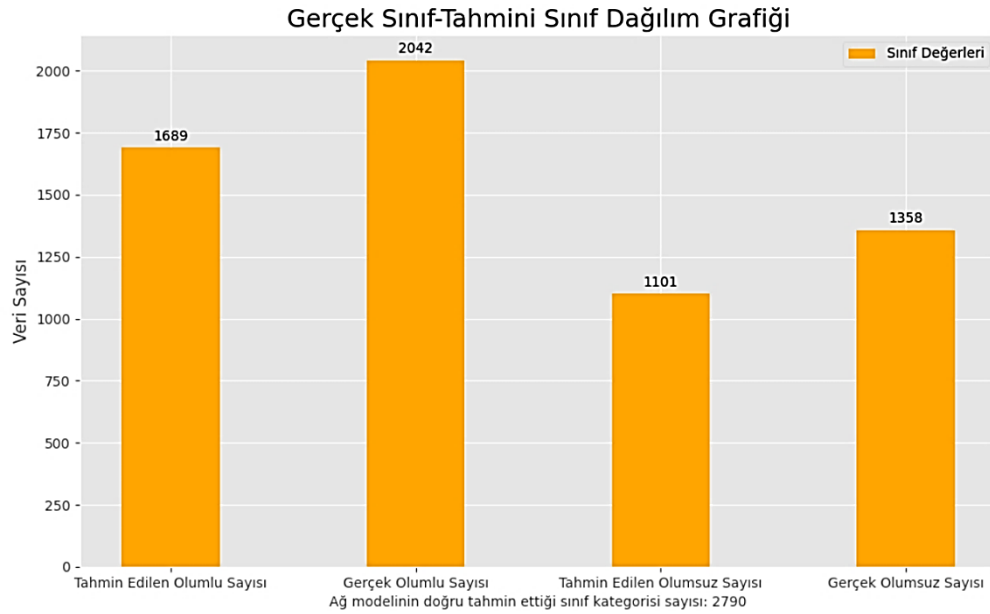
Eğitim Doğruluğu: 0.8390

Test Doğruluğu: 0.8206

Şekil 3.51: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

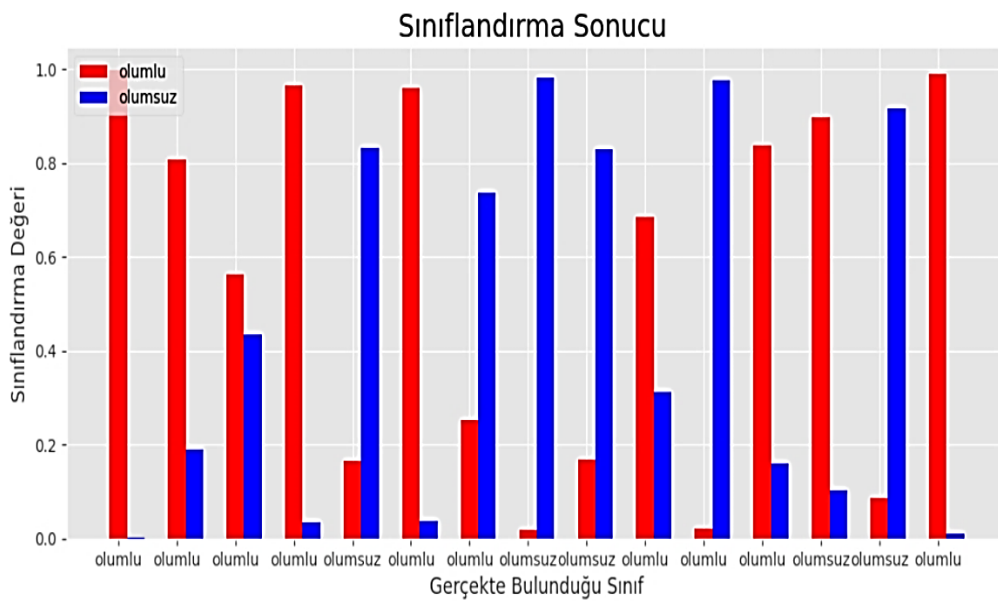


Şekil 3.52: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2042 adet olumlu duygu belirten veriden sadece 1689 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1358 olumsuz yorumdan 1101 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.53'te ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelinin tahmin ettiği sınıf değeri gösterilmiştir.

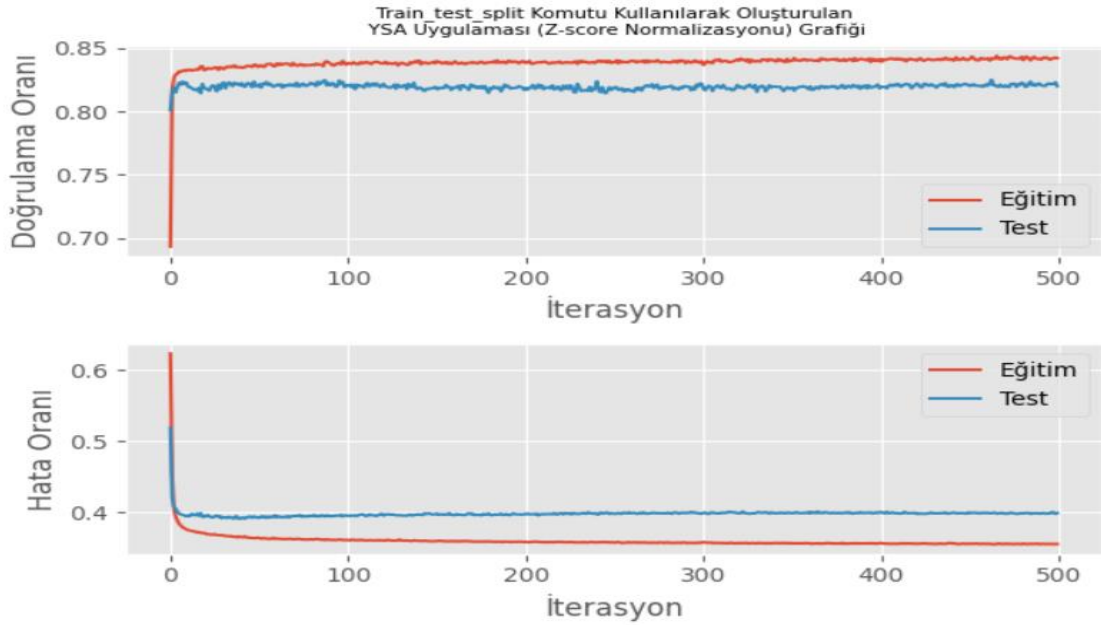
Şekil 3.53: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.2.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması

Z-score normalizasyonu kullanılarak yapılan sınıflandırmada eğitim doğruluğu yüzde 84 iken test doğruluk oranının yüzde 82’de kaldığı görülmüştür. Sınıflandırılma sonucuna bakıldığında ise test verisinde olumlu yorum sayısı 1974, olumsuz yorum sayısı 1426 olduğu görülmektedir. Ancak gerçekte test verisi incelendiğinde 2042 olumlu yorum, 1358 olumsuz yorum verisi içerdiği görülmektedir.

Şekil 3.54: Modelin Doğruluk-Hata Oranları Grafiği

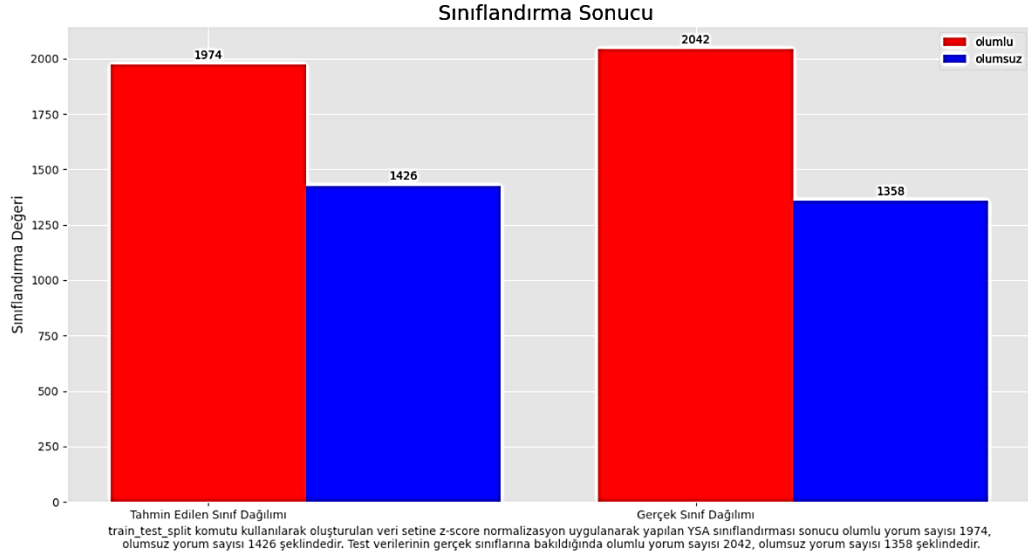


Şekil 3.55: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

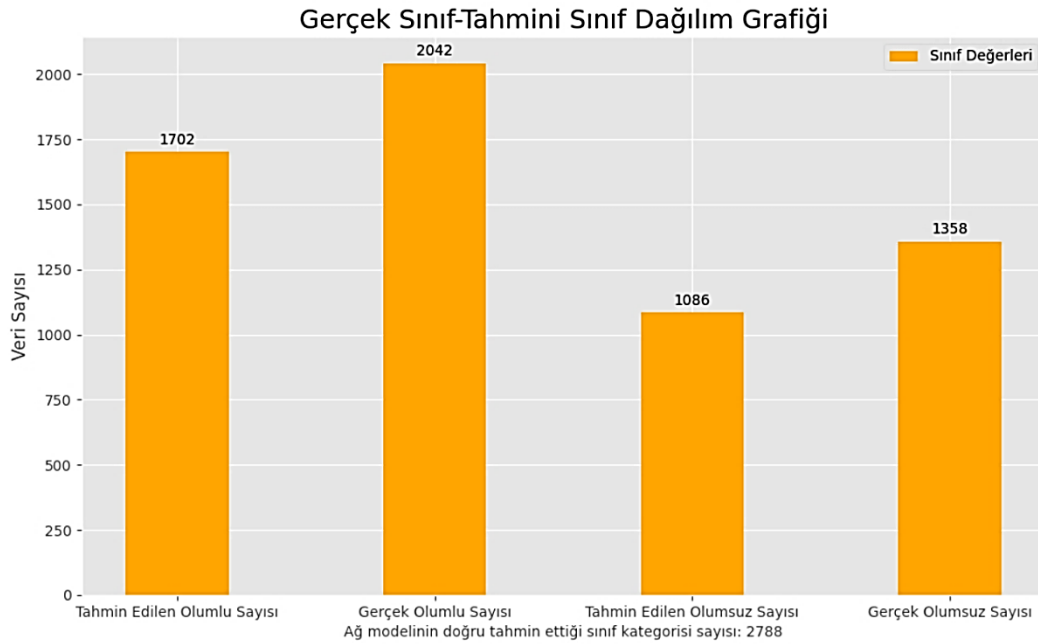
Eğitim Doğruluğu: 0.8438

Test Doğruluğu: 0.8200

Şekil 3.56: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım

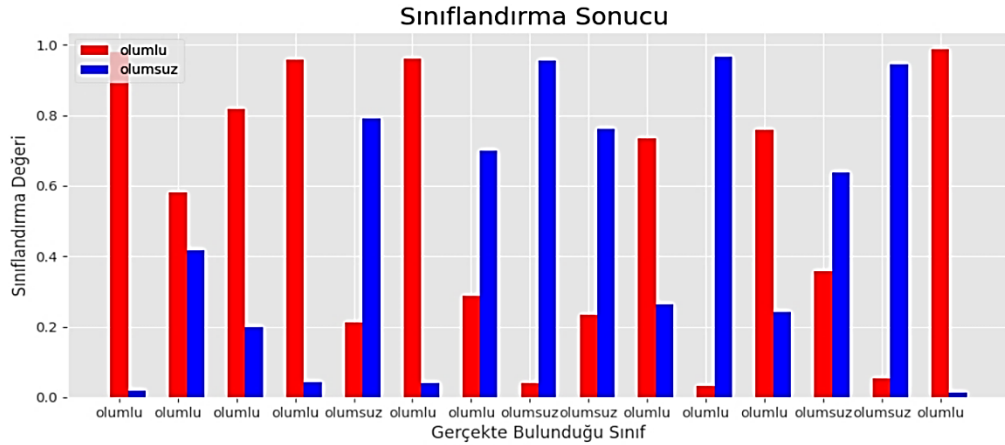


Şekil 3.57: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2042 adet olumlu duygu belirten veriden sadece 1702 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1358 olumsuz yorumdan 1086 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Şekil 3.58’de ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelinin tahmin ettiği sınıf değeri gösterilmiştir.

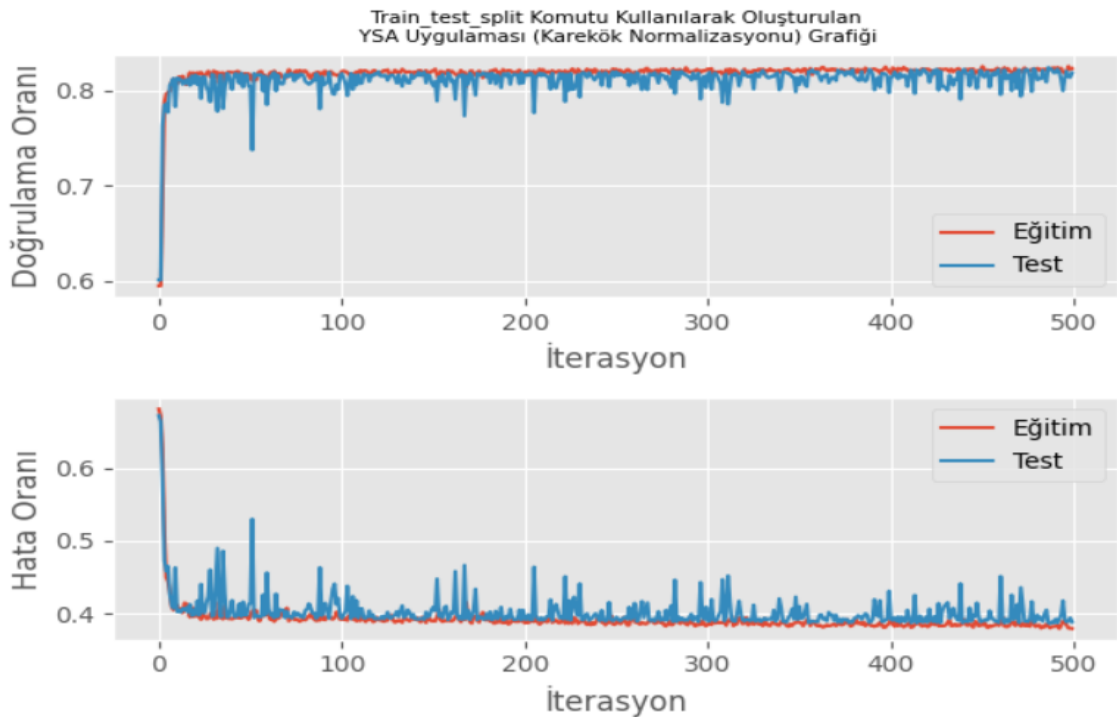
Şekil 3.58: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.2.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması

Eğitim doğruluğu yüzde 82 iken test doğruluk oranının yüzde 81’de kaldığı görülmüştür. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1959, olumsuz duygu belirten yorum sayısı 1441 olduğu görülmektedir. Ancak gerçekte test verisi incelendiğinde 2042 olumlu yorum, 1358 olumsuz yorum verisi içerdiği görülmektedir.

Şekil 3.59: Modelin Doğruluk-Hata Oranları Grafiği

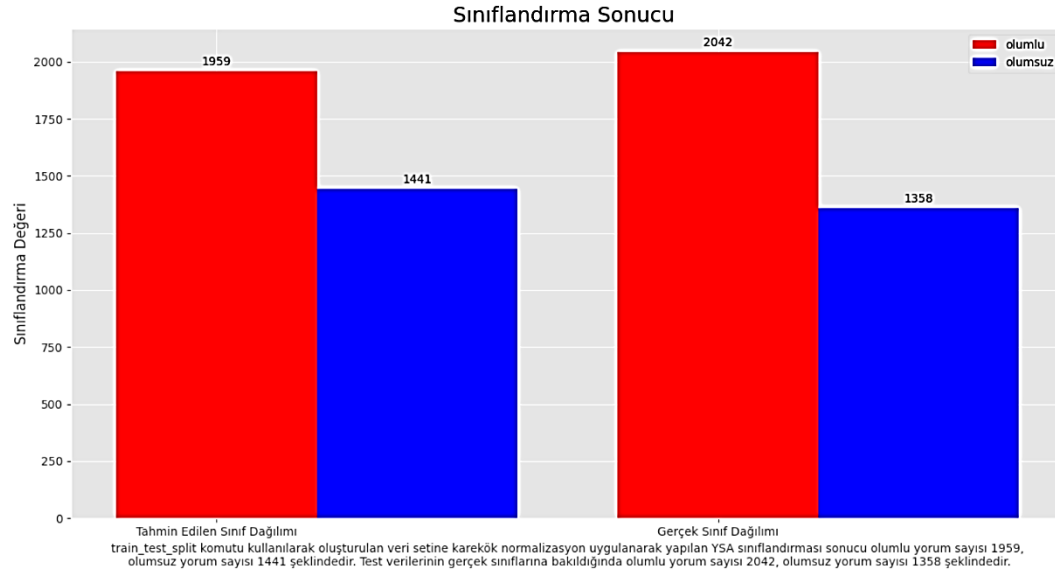


Şekil 3.60: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

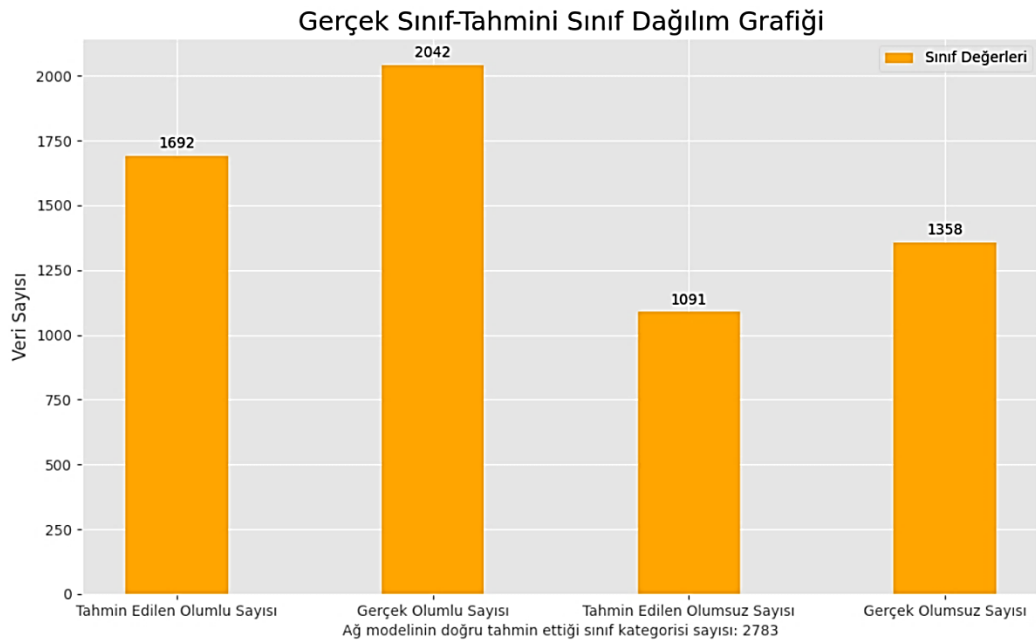
Eğitim Doğruluğu: 0.8241

Test Doğruluğu: 0.8185

Şekil 3.61: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



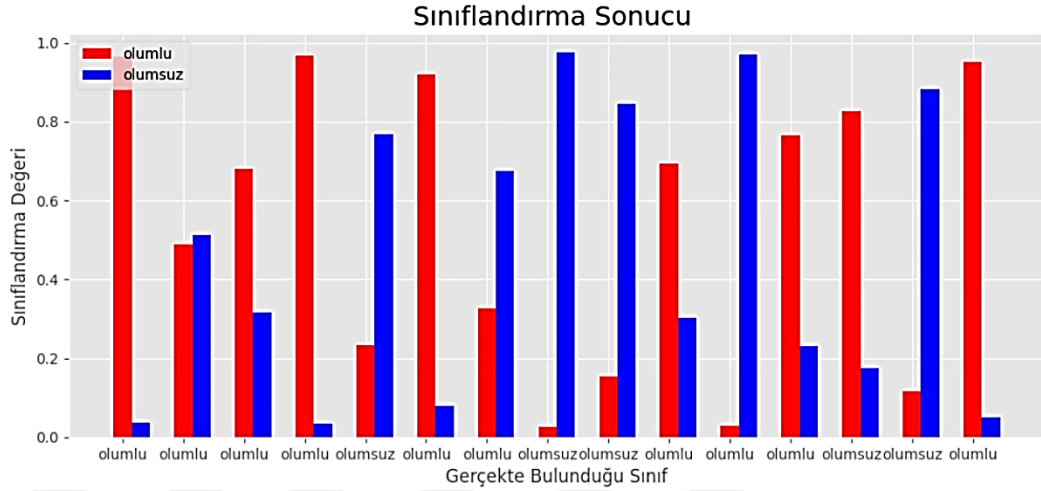
Şekil 3.62: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2042 adet olumlu duygu belirten veriden sadece 1692 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1358 olumsuz yorumdan 1091 tanesinin sınıfını

doğru tahmin ettiği görülmektedir. Şekil 3.63'te ise test verisi içinden alınan 15 yorumun gerçek sınıf değerleri ile ağ modelin tahmin ettiği sınıf değeri gösterilmiştir.

Şekil 3.63: Test Verisi İçinden 15 Veri Seçilerek Her Bir Veriye Ait Tahmin Grafiğinin Dağılımı



3.2.3. K Değeri 5 Alınarak Yapılan K-fold Çarpraz Doğrulama İle Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması

Verileri eğitim ve test verileri olarak parçalara ayırma işleminde kullanılan çarpraz doğrulamada k değeri 5 alındı. Veri seti 5 bölmeye ayrıldı ve her defasında 1 bölüm test verisi, kalan veriler ise eğitim verisi olarak kullanılacak şekilde sınıflandırılma işlemi yapıldı. Her bir döngünün sonunda çıkan doğruluk değerlerinin ortalaması alınarak ağın başarı oranı oluşturuldu.

3.2.3.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki StandartScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri şekil 3.64'te gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.65'te belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1992, olumsuz yorum sayısının 1408 olduğu görülmektedir. Ancak gerçekte test verisinde 2026 olumlu yorum, 1374 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.64: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

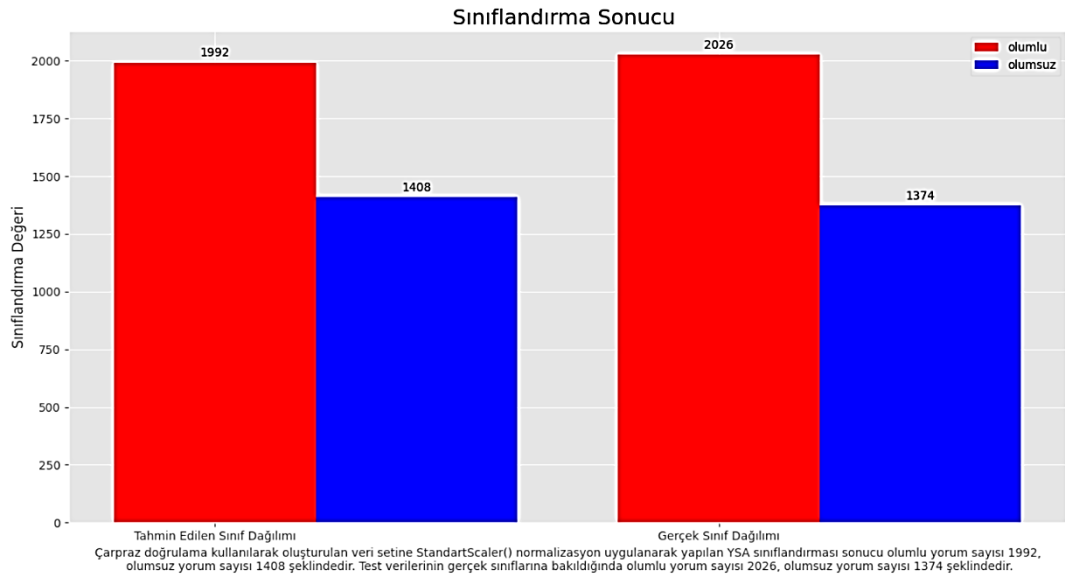
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5
Eğitim Doğruluğu	0.8376	0.844	0.8376	0.845	0.8377
Test Doğruluğu	0.8203	0.82	0.8291	0.8141	0.8341

Şekil 3.65: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

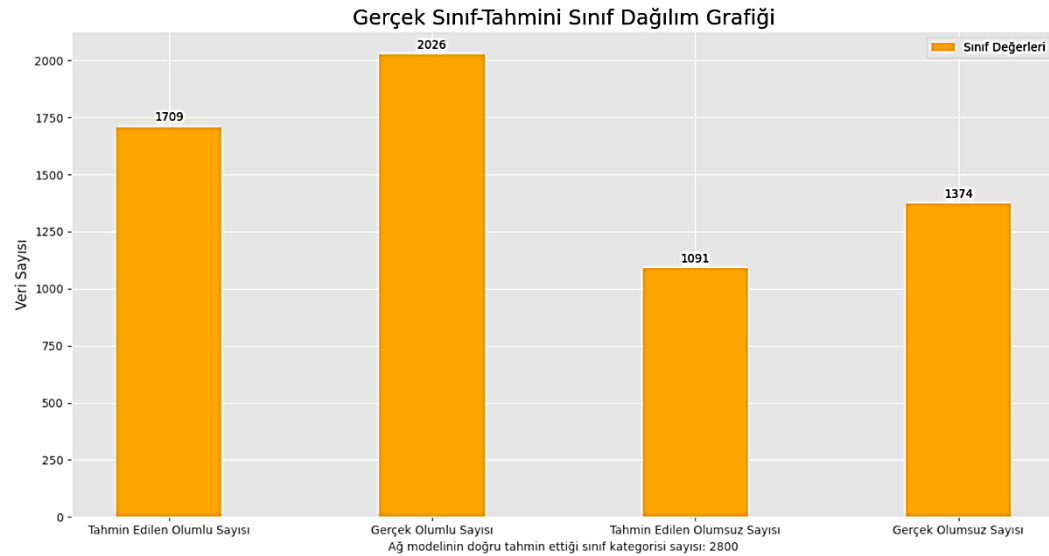
Eğitim Doğruluğu: 0.8404

Test Doğruluğu: 0.8235

Şekil 3.66: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.67: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2026 adet olumlu duygu belirten veriden sadece 1709 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1374 olumsuz yorumdan 1091 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.2.3.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki RobustScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri şekil 3.68’de gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.69’da belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1985, olumsuz yorum sayısının 1415 olduğu görülmektedir. Ancak gerçekte test verisinde 2026 olumlu yorum, 1374 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.68: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

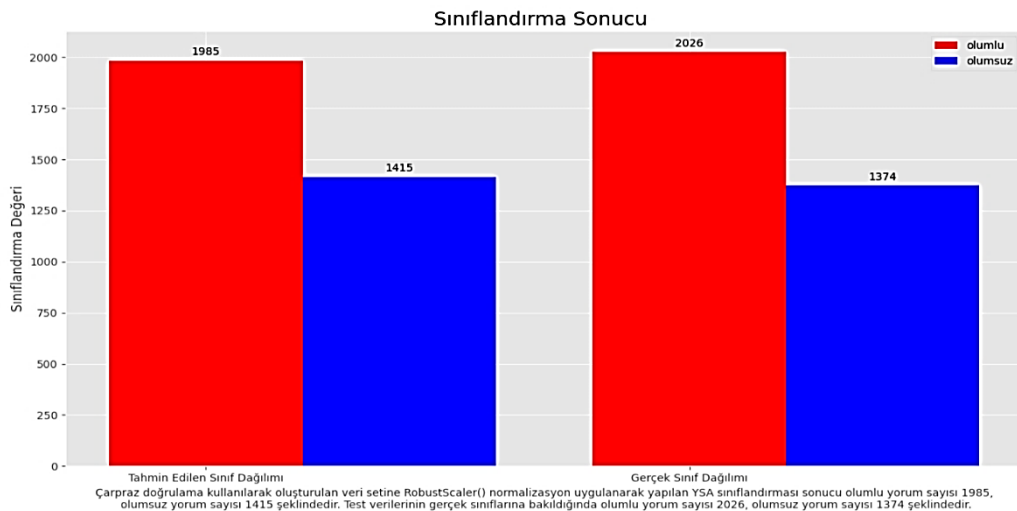
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5
Eğitim Doğruluğu	0.8358	0.839	0.8367	0.841	0.8361
Test Doğruluğu	0.8212	0.8229	0.8294	0.8197	0.8291

Şekil 3.69: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

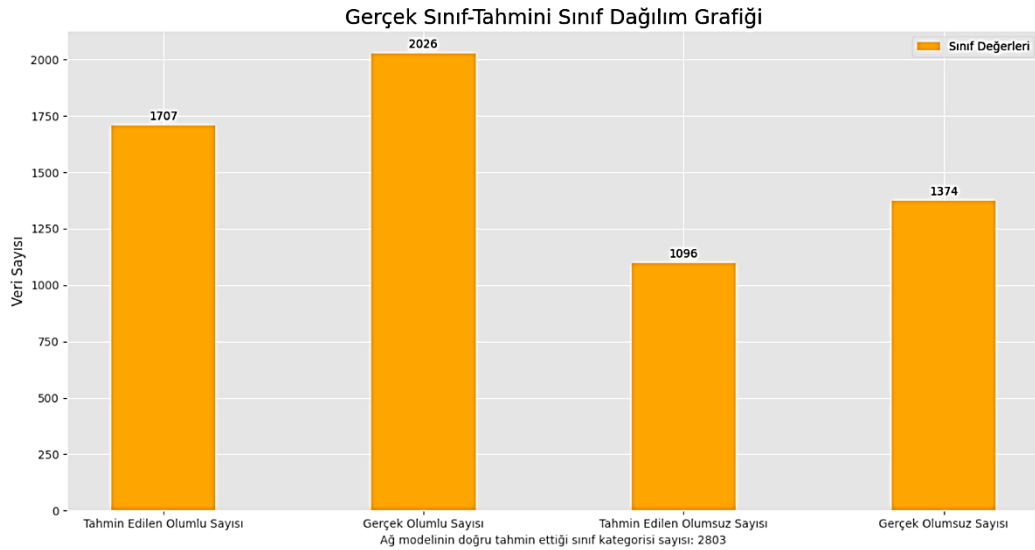
Eğitim Doğruluğu: 0.8377

Test Doğruluğu: 0.8245

Şekil 3.70: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.71: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2026 adet olumlu duygu belirten veriden sadece 1707 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1374 olumsuz yorumdan 1096 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.2.3.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki MinMaxScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri Şekil 3.72’de gösterilmiştir. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.73’te belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 2008, olumsuz yorum sayısının 1391 olduğu görülmektedir. Ancak gerçekte test verisinde 2026 olumlu yorum, 1374 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.72: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

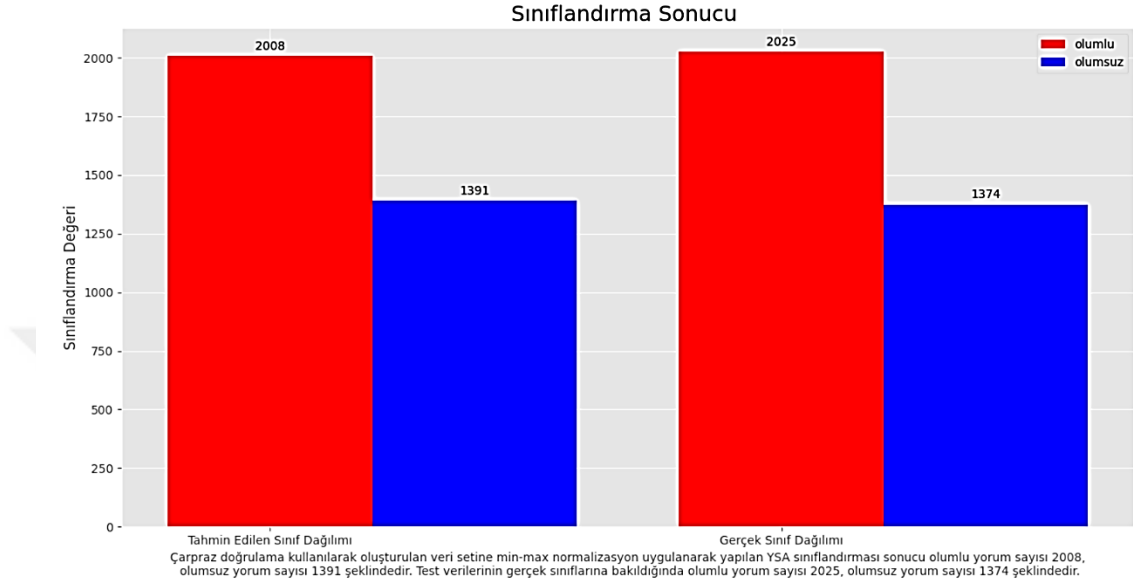
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5
Eğitim Doğruluğu	0.8388	0.8426	0.8351	0.8376	0.8363
Test Doğruluğu	0.8221	0.8262	0.8332	0.8153	0.8391

Şekil 3.73: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

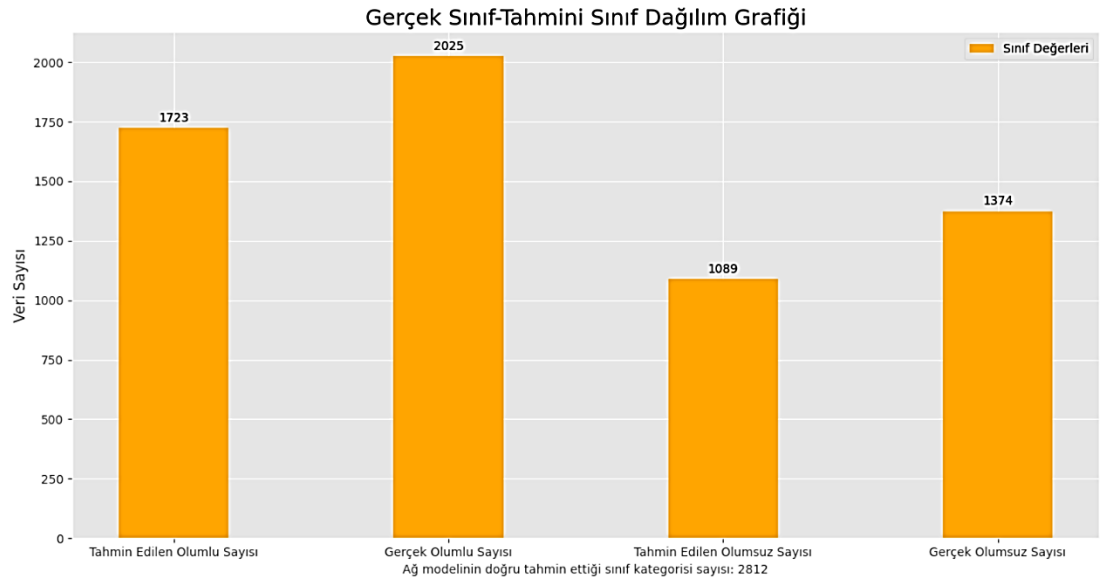
Eğitim Doğruluğu: 0.8381

Test Doğruluğu: 0.8272

Şekil 3.74: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.75: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2026 adet olumlu duygu belirten veriden sadece 1723 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1374 olumsuz yorumdan 1089 tanesinin sınıfını

doğru tahmin ettiği görülmektedir. Ağın toplamda 2812 yorumun gerçek sınıfını doğru tahmin ettiği görülmektedir.

3.2.3.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması

Her bir döngü sonunda alınan doğruluk değerleri Şekil 3.76'da gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.77'de belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1955, olumsuz yorum sayısının 1445 olduğu görülmektedir. Ancak gerçekte test verisinde 2025 olumlu yorum, 1375 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.76: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

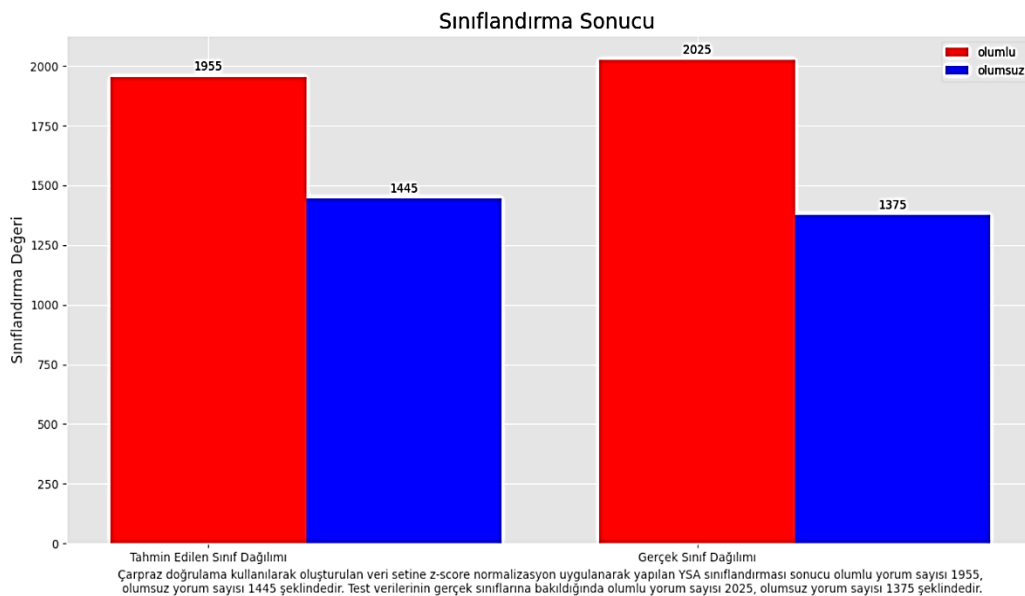
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5
Eğitim Doğruluğu	0.8433	0.843	0.8394	0.8401	0.8428
Test Doğruluğu	0.8203	0.8135	0.8341	0.8182	0.8309

Şekil 3.77: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

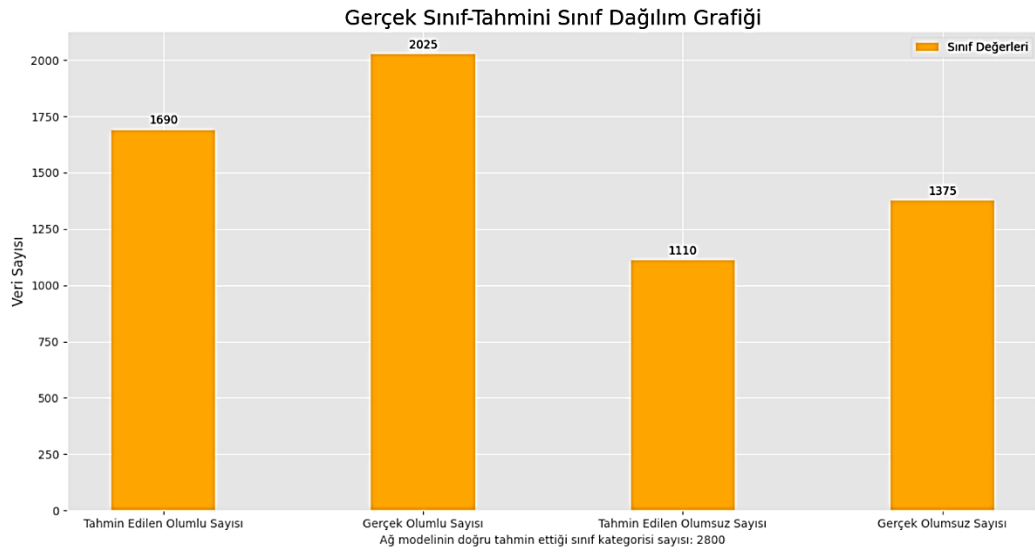
Eğitim Doğruluğu: 0.8417

Test Doğruluğu: 0.8234

Şekil 3.78: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.79: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2025 adet olumlu duygu belirten veriden sadece 1690 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1375 olumsuz yorumdan 1110 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Ağın toplamda 3400 yorum içerisinde 2800 yorumun gerçek sınıfını doğru tahmin ettiği görülmektedir.

3.2.3.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması

Her bir döngü sonunda alınan doğruluk değerleri Şekil 3.80’de gösterilmiştir. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.81’de belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 2827, olumsuz yorum sayısının 573 olduğu görülmektedir. Ancak gerçekte test verisinde 2026 olumlu yorum, 1374 olumsuz yorum verisi olduğu görülmektedir. Ağın verilerin sınıflarını doğru tahmin etmedeki başarı oranının çok düşük olduğu görülmektedir.

Şekil 3.80: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

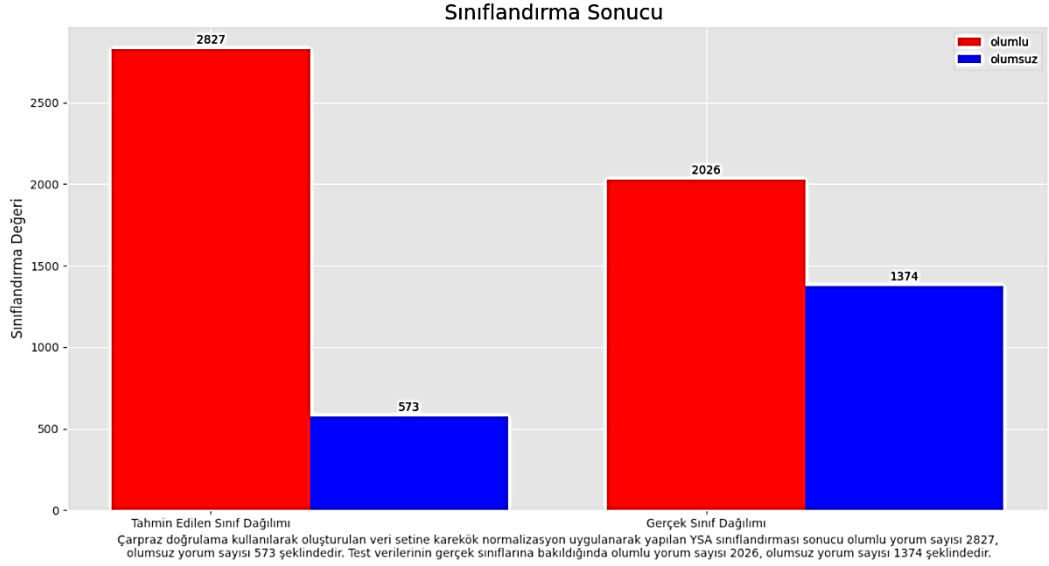
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5
Eğitim Doğruluğu	0.596	0.5965	0.5935	0.8218	0.8207
Test Doğruluğu	0.5944	0.5926	0.6044	0.8179	0.8303

Şekil 3.81: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

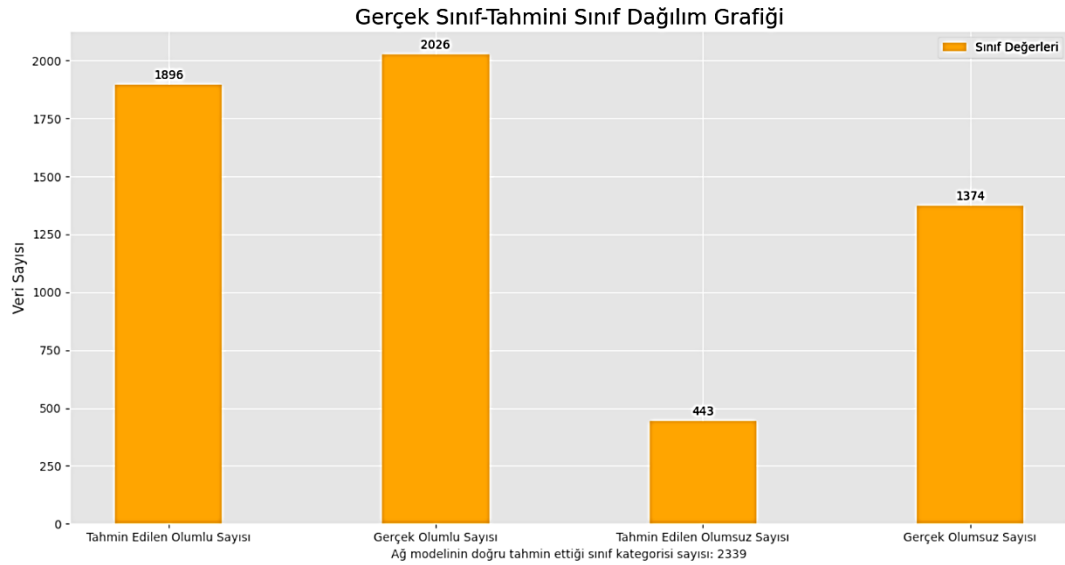
Eğitim Doğruluğu: 0.6857

Test Doğruluğu: 0.6879

Şekil 3.82: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.83: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 2026 adet olumlu duygu belirten veriden sadece 1896 tanesini doğru tahmin ettiği görülmektedir. Gerçek 1376 olumsuz yorumdan 443 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.2.4. K Değeri 10 Alınarak Yapılan K-fold Çarpaz Doğrulama İle Oluşturulan Eğitim ve Test Verilerinde YSA Sınıflandırılması

Verileri eğitim ve test verileri olarak parçalara ayırma işleminde kullanılan çarpaz doğrulamada k değeri 10 alındı. Veri seti 10 bölmeye ayrıldı ve her defasında 1 bölüm test verisi, kalan veriler ise eğitim verisi olarak kullanılacak şekilde sınıflandırılma işlemi yapıldı. Her bir döngünün sonunda çıkan doğruluk değerlerinin ortalaması alınarak ağın başarı oranı oluşturuldu.

3.2.4.1. Standart Normalizasyon Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki StandartScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri şekil 3.84'te gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.85'te belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 974, olumsuz yorum sayısının 726 olduğu görülmektedir. Ancak gerçekte test verisinde 1013 olumlu yorum, 1374 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.84: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

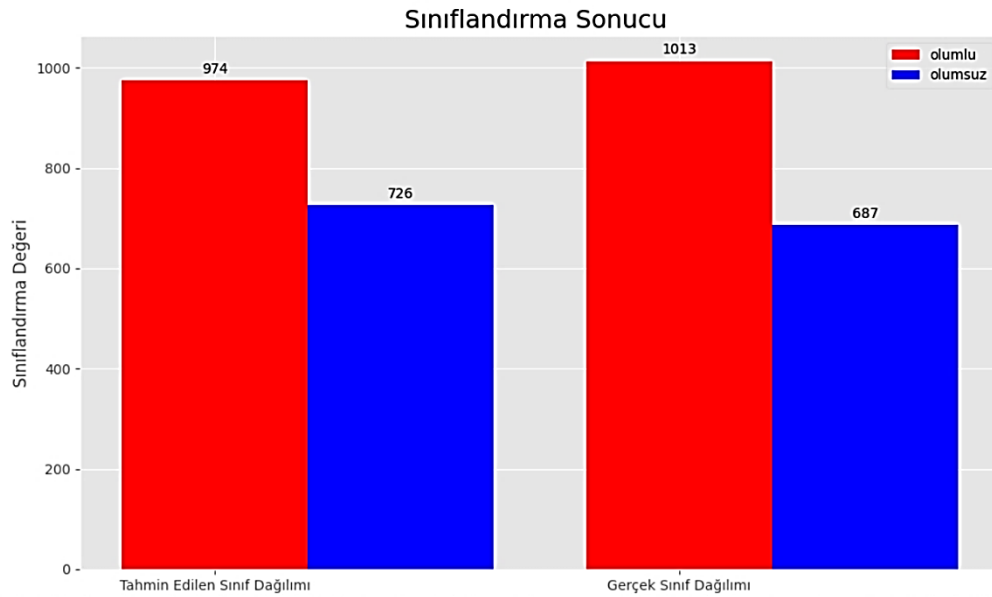
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5	Döngü 6	Döngü 7	Döngü 8	Döngü 9	Döngü 10
Eğitim Doğruluğu	0.838	0.8397	0.8431	0.8383	0.8373	0.8376	0.8377	0.8387	0.837	0.8389
Test Doğruluğu	0.8318	0.8047	0.8171	0.8312	0.8235	0.83	0.8206	0.8288	0.8365	0.8388

Şekil 3.85: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

Eğitim Doğruluğu: 0.8386

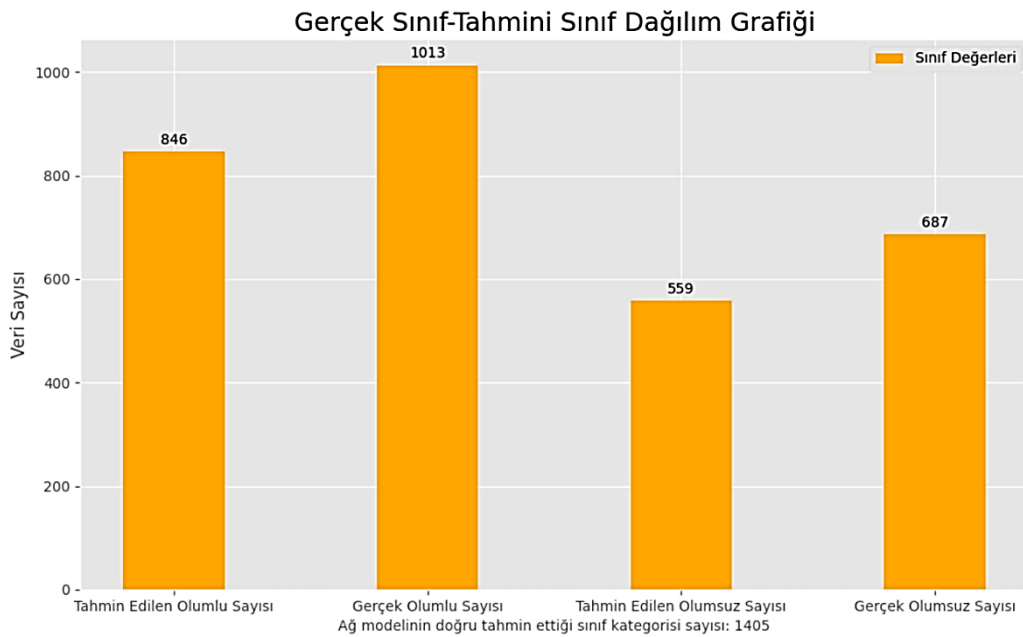
Test Doğruluğu: 0.8263

Şekil 3.86: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Çarpaz doğrulama kullanılarak oluşturulan veri setine StandartScaler() normalizasyon uygulanarak yapılan YSA sınıflandırması sonucu olumlu yorum sayısı 974, olumsuz yorum sayısı 726 şeklindedir. Test verilerinin gerçek sınıflarına bakıldığında olumlu yorum sayısı 1013, olumsuz yorum sayısı 687 şeklindedir.

Şekil 3.87: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 1013 adet olumlu duygu belirten veriden sadece 846 tanesini doğru tahmin ettiği görülmektedir. Gerçek 687 olumsuz yorumdan 559 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.2.4.2. RobustScaler() Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki RobustScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri şekil 3.88'de gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.89'da belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 992, olumsuz yorum sayısının 708 olduğu görülmektedir. Ancak gerçekte test verisinde 1013 olumlu yorum, 687 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.88: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

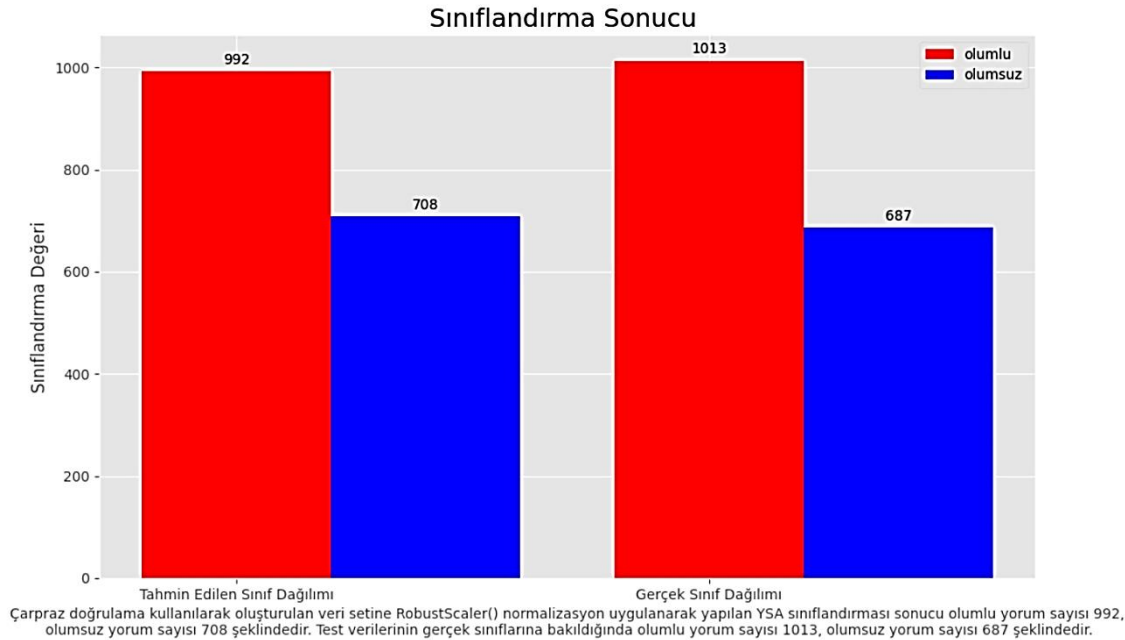
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5	Döngü 6	Döngü 7	Döngü 8	Döngü 9	Döngü 10
Eğitim Doğruluğu	0.8355	0.8386	0.8378	0.8418	0.8373	0.8341	0.8379	0.8357	0.8346	0.8353
Test Doğruluğu	0.8335	0.8153	0.8165	0.8229	0.8382	0.8312	0.8047	0.8259	0.8365	0.8406

Şekil 3.89: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

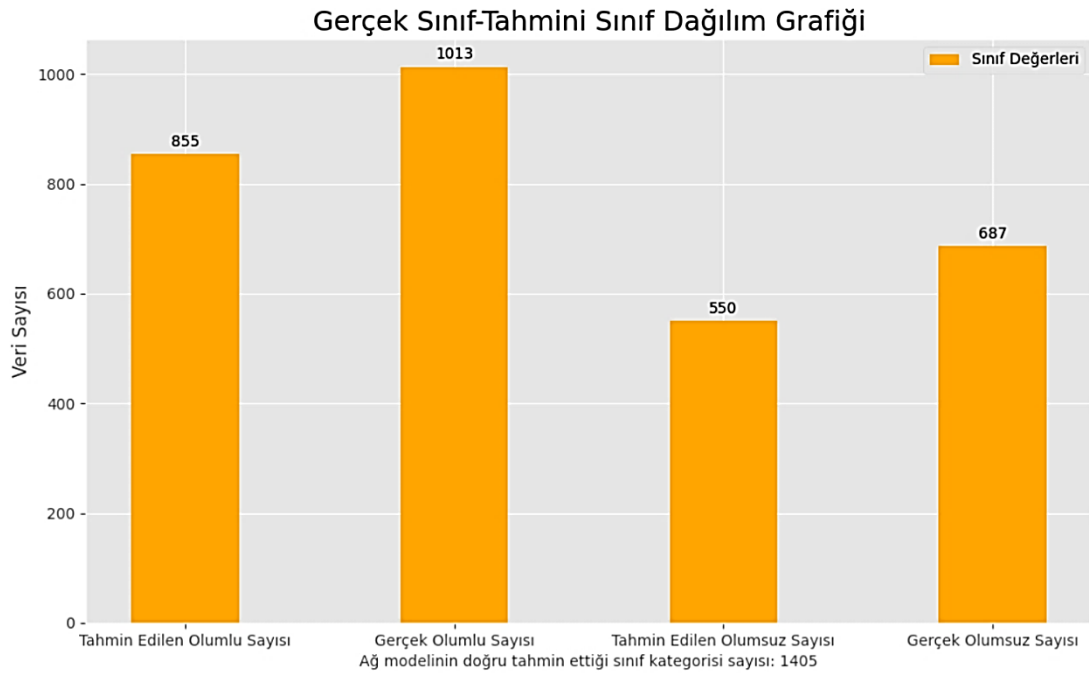
Eğitim Doğruluğu: 0.8369

Test Doğruluğu: 0.8265

Şekil 3.90: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.91: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 1013 adet olumlu duygu belirten veriden sadece 855 tanesini doğru tahmin ettiği görülmektedir. Gerçek 687 olumsuz yorumdan 550 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.2.4.3. Min-Max Normalizasyonu Uygulanan YSA Sınıflandırması

Python scikit-learn kütüphanesindeki MinMaxScaler() fonksiyonu kullanıldı. Her bir döngü sonunda alınan doğruluk değerleri Şekil 3.92’de gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.93’te belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 983, olumsuz yorum sayısının 718 olduğu görülmektedir. Ancak gerçekte test verisinde 1013 olumlu yorum, 688 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.92: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

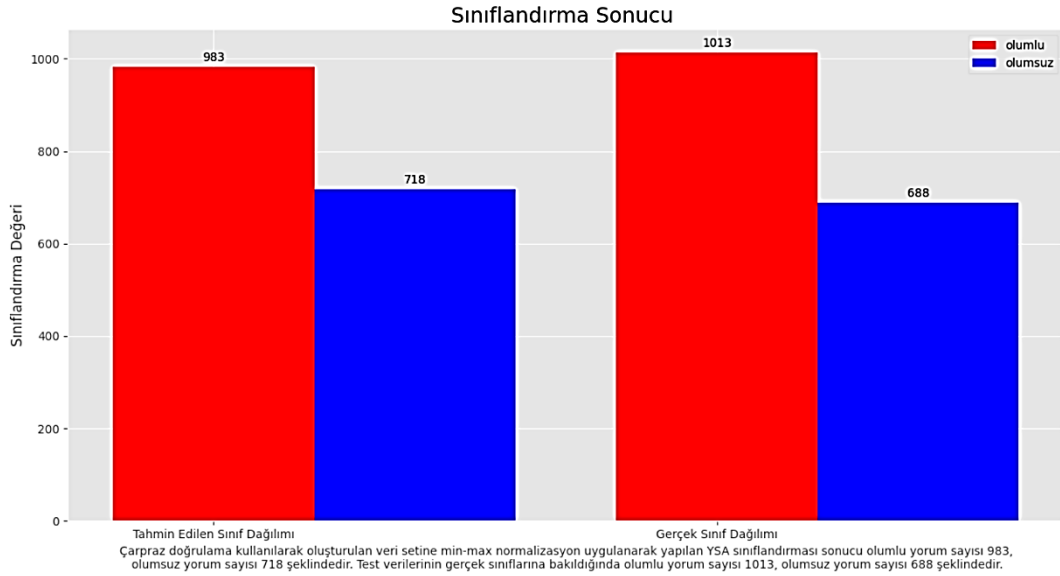
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5	Döngü 6	Döngü 7	Döngü 8	Döngü 9	Döngü 10
Eğitim Doğruluğu	0.8386	0.8404	0.8389	0.8392	0.8384	0.8365	0.8395	0.8365	0.8373	0.8342
Test Doğruluğu	0.8265	0.8153	0.8129	0.8212	0.8306	0.8271	0.8129	0.8235	0.8376	0.8318

Şekil 3.93: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

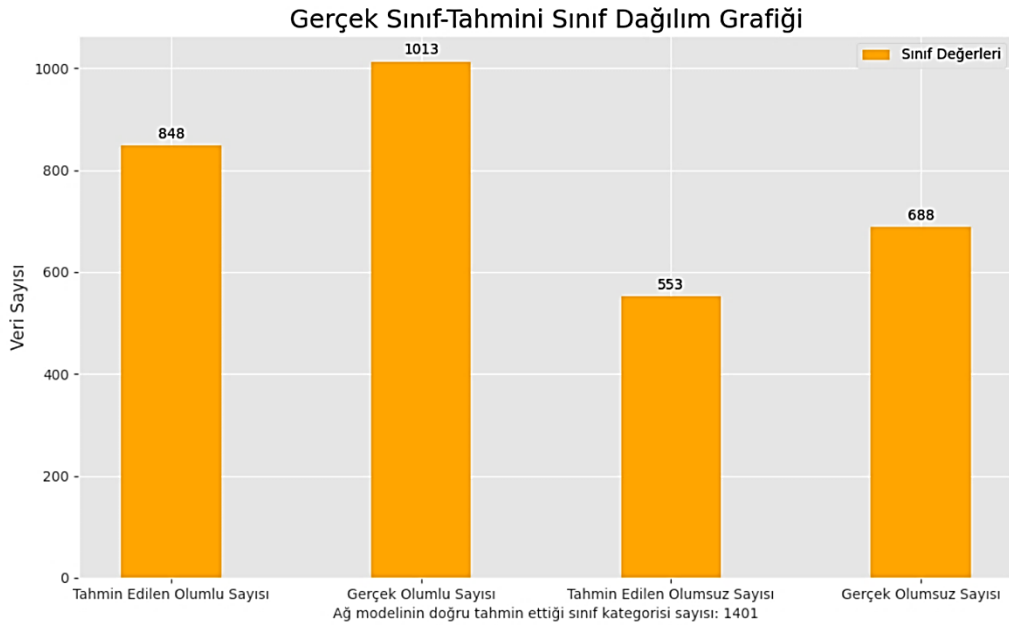
Eğitim Doğruluğu: 0.8379

Test Doğruluğu: 0.8239

Şekil 3.94: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.95: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 1013 adet olumlu duygu belirten veriden sadece 848 tanesini doğru tahmin ettiği görülmektedir. Gerçek 688 olumsuz yorumdan 553 tanesinin sınıfını doğru

tahmin ettiği görülmektedir. Ağın toplamda 1401 yorumun gerçek sınıfını doğru tahmin ettiği görülmektedir.

3.2.4.4. Z-score Normalizasyonu Uygulanan YSA Sınıflandırması

Her bir döngü sonunda alınan doğruluk değerleri şekil 3.96'da gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.97'de belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 977, olumsuz yorum sayısının 722 olduğu görülmektedir. Ancak gerçekte test verisinde 1012 olumlu yorum, 687 olumsuz yorum verisi olduğu görülmektedir.

Şekil 3.96: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

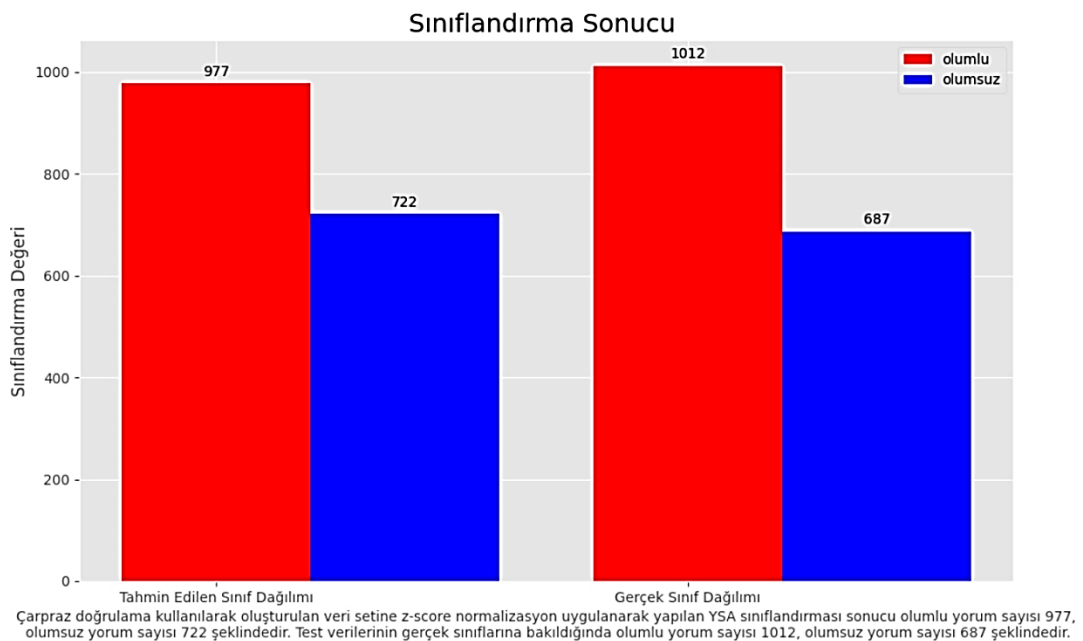
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5	Döngü 6	Döngü 7	Döngü 8	Döngü 9	Döngü 10
Eğitim Doğruluğu	0.8381	0.8392	0.8396	0.8394	0.8373	0.8401	0.8418	0.8421	0.836	0.8404
Test Doğruluğu	0.82	0.8041	0.8106	0.8329	0.8253	0.8276	0.81	0.8176	0.8347	0.8329

Şekil 3.97: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

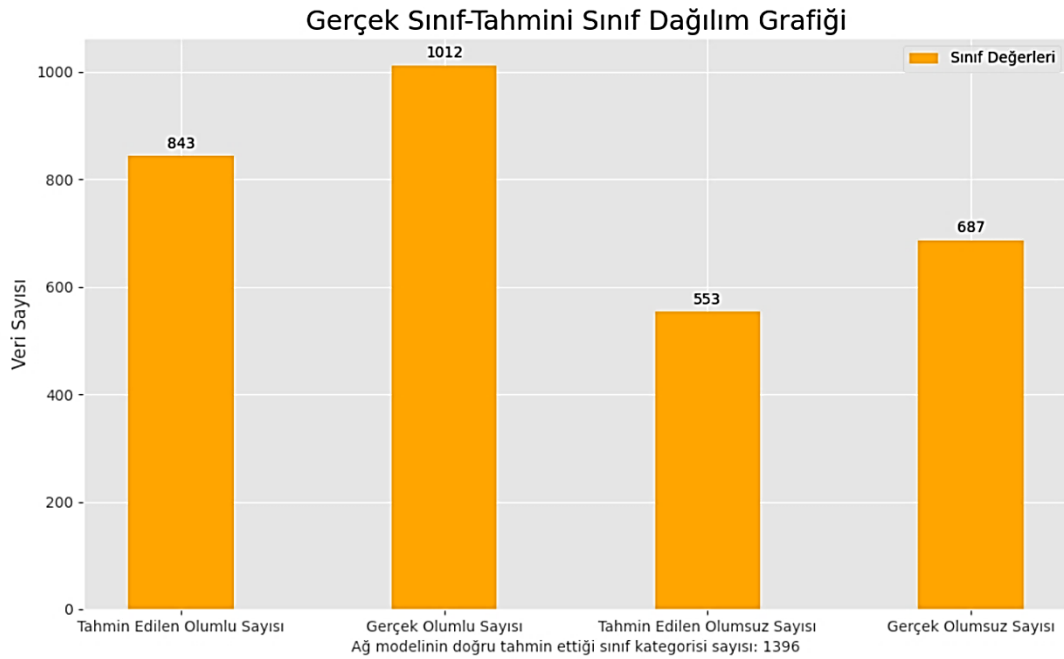
Eğitim Doğruluğu: 0.8394

Test Doğruluğu: 0.8216

Şekil 3.98: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.99: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Yukarıdaki grafik incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 1012 adet olumlu duygu belirten veriden sadece 843 tanesini doğru tahmin ettiği görülmektedir. Gerçek 687 olumsuz yorumdan 553 tanesinin sınıfını doğru tahmin ettiği görülmektedir. Ağın toplamda 1699 yorum içerisinde 1396 yorumun gerçek sınıfını doğru tahmin ettiği görülmektedir.

3.2.4.5. Karekök Normalizasyonu Uygulanan YSA Sınıflandırması

Her bir döngü sonunda alınan doğruluk değerleri Şekil 3.100'de gösterilmiştir. Ağın başarı oranı bu değerlerin ortalaması alınarak hesaplandı. Ağ modelinin eğitim ortalama başarı oranı ve test verilerindeki başarı oranı Şekil 3.101'de belirtilmiştir. Sınıflandırılma sonucuna bakıldığında test verisinde olumlu yorum sayısı 1145, olumsuz yorum sayısının 555 olduğu görülmektedir. Ancak gerçekte test verisinde 1013 olumlu yorum, 687 olumsuz yorum verisi olduğu görülmektedir. Ağın verilerin sınıflarını doğru tahmin etmedeki başarı oranının çok düşük olduğu görülmektedir.

Şekil 3.100: Her Döngü Sonundaki Eğitim ve Test Doğruluk Oranları

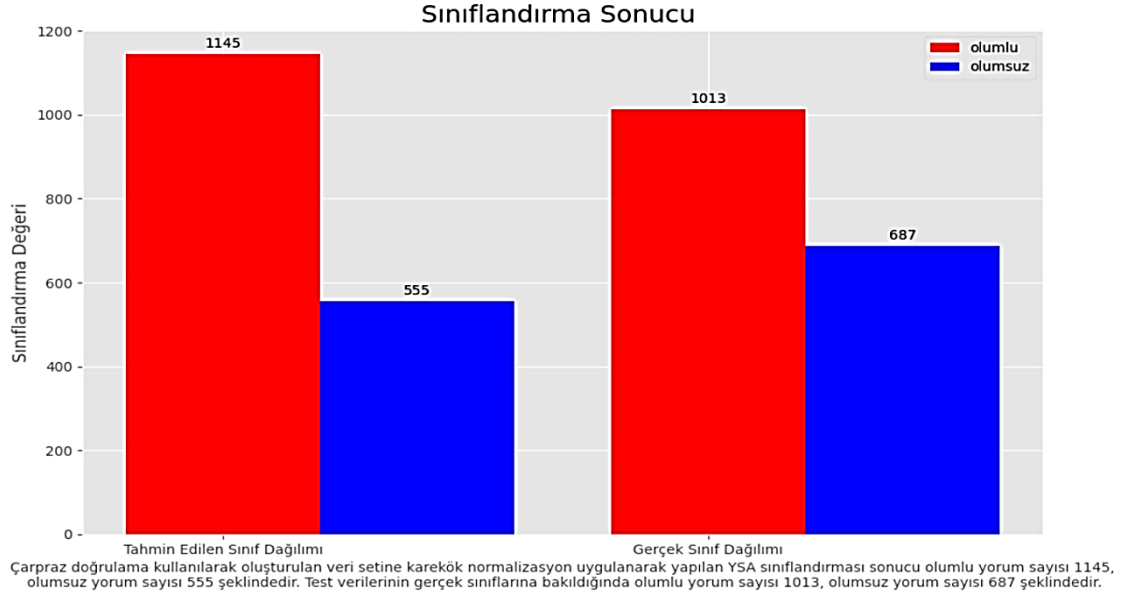
	Döngü 1	Döngü 2	Döngü 3	Döngü 4	Döngü 5	Döngü 6	Döngü 7	Döngü 8	Döngü 9	Döngü 10
Eğitim Doğruluğu	0.8214	0.8291	0.8244	0.8197	0.8207	0.5932	0.5963	0.8217	0.8205	0.8207
Test Doğruluğu	0.8259	0.8129	0.8053	0.8224	0.8182	0.6182	0.5906	0.8271	0.8318	0.8259

Şekil 3.101: Modelin Eğitim ve Test Verisi Üzerindeki Başarı Oranı

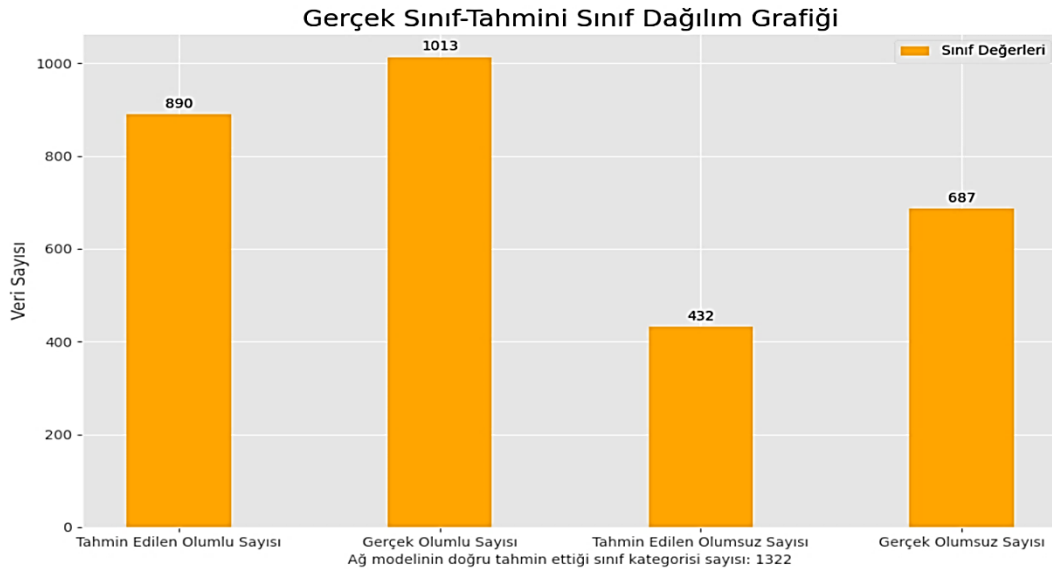
Eğitim Doğruluğu: 0.7768

Test Doğruluğu: 0.7778

Şekil 3.102: Modelin Test Verisi Üzerinde Uygulanması Sonucu Oluşan Dağılım



Şekil 3.103: Oluşturulan Modelin Gerçekte Olumlu ve Olumsuz Yorumların Ne Kadarını Doğru Tahmin Ettiğini Gösteren Grafik



Şekil 3.103 incelendiğinde oluşturulan ağ modelinin gerçekte test verisi içinde bulunan 1013 adet olumlu duygu belirten veriden sadece 890 tanesini doğru tahmin ettiği görülmektedir. Gerçek 687 olumsuz yorumdan 432 tanesinin sınıfını doğru tahmin ettiği görülmektedir.

3.3. BULGULAR

YSA kullanılarak yapılan MS duygu analizi çalışmasında, verilen modeller içerisinde en iyi performansı Min-Max normalizasyon ile k değeri 5 alınarak yapılan çarpaz doğrulama modeli göstermiştir. En az başarı oranı ise yine aynı çarpaz doğrulama kullanılan karekök normalizasyon yönteminde görülmüştür. Tablo 3.1’de eğitim ve test veri setine ayırmada kullanılan farklı yöntemler ile farklı normalizasyon yöntemi uygulanması sonucu ağın modellere göre değişen başarı oranını göstermektedir.

Tablo 3.1: Ağın farklı durumlarda başarı oranı

Eğitim ve Test Veri Setine Ayırmada Kullanılan Yöntemler				
Normalizasyon	Manuel	Train_test_split	K=5	K=10
StandartScaler()	0.7253	0.8209	0.8235	0.8263
RobustScaler()	0.7254	0.8174	0.8245	0.8265
MinMaxScaler()	0.73	0.8206	0.8272	0.8239
Z-score	0.7274	0.82	0.8234	0.8216
Karekök	0.7159	0.8185	0.6879	0.7778

SONUÇ

Metin sınıflandırma duygu analizi çalışmaları ile kullanıcıların aldıkları hizmete yönelik yaptıkları yorumlardan memnuniyet durumları çıkarılabilmektedir. Müşterilerin memnuniyet durumları kullanılarak otellere bilgi verilebilir ve bu doğrultuda kullanıcı memnuniyetini arttırmak amacıyla çalışmalar yapılabilir.

Bu çalışma kapsamında, Python dilinde bulunan Keras ve Tensorflow kütüphaneleri kullanılarak duygu analizi yapılmıştır. Tez çalışması kapsamında, farklı normalizasyon fonksiyonları ile YSA sınıflandırma modeli kullanılmıştır. Duygu analizi çalışmaları doğrultusunda otel müşteri yorumları, olumlu ve olumsuz yorum olmak üzere iki sınıfta ele alınmıştır.

Sınıflandırma performansları incelendiğinde, genel olarak çarpaz doğrulama kullanılarak yapılan ayırma işleminde ağ modelinin daha iyi performans gösterdiği görülmektedir. Ağ modeli en yüksek başarı oranını k değeri 5 alınarak yapılan çarpaz doğrulama modelinde min-max normalizasyon yönteminde göstermiştir. Bu modelde ağ, 3400 test verisinden 2812 tanesinin gerçekte bulunduğu sınıfı doğru tahmin etmiştir. Bütün eğitim ve test verilerine ayırma yöntemlerine bakıldığında en düşük başarı oranları karekök normalizasyonu yönteminde görülmektedir. Bu tez kapsamında en düşük başarı oranı k değeri 5 alınarak yapılan karekök normalizasyon modelinde görülmektedir. 3400 test verisi içerisinde 2783 yorumun gerçek sınıfını doğru tahmin etmiştir.

Gelecek çalışmalarda, daha yüksek bir veri seti kullanılarak ağın performansı yükseltilebilir. Ayrıca, otel yorumları hijyen, ulaşım, yeme-içme gibi kategorilere ayrılarak hangi sınıfta müşterilerin memnun ya da memnuniyetsiz olduğu belirlenebilir. Otel bazlı duygu analizi yapılarak en çok hangi otele daha çok olumsuz yorum geldiği belirlenebilir.



Ek 1

Manuel Eğitim-Test verilerinin sınıflandırılmasında Python kodları

```

from random import random
import matplotlib
import pandas as pd
import numpy as np
import tensorflow_addons
from sklearn import preprocessing
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from keras.utils import np_utils
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LayerNormalization
from sklearn.metrics import classification_report
from tensorflow_addons.layers import WeightNormalization
from sklearn.preprocessing import StandardScaler
from keras import backend as K
#####çizim kütüphanesi çiz ve sonuçları göster
import matplotlib.pyplot as plt
matplotlib.axes.Axes.bar
matplotlib.pyplot.bar
matplotlib.axes.Axes.annotate
matplotlib.pyplot.annotate
from tensorflow.python.layers.normalization import BatchNormalization
def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying its height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}' .format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3), # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')
###normalizasyon
##standart sapma normalize
def standard_norm(arr):
    mean = np.mean(arr)
    stdev = np.std(arr)
    return (arr-mean)/stdev
def log_normalize(arr):
    return np.log(arr)
def sqrt_normalize(arr):

```

```

        return np.sqrt(arr)

plt.style.use('ggplot')
dizi=[]
#uygulama="Min-Max Normalizasyonu"
#normalizasyon="min-max"
#uygulama="Z-score Normalizasyonu"
#normalizasyon="z-score"
#uygulama="StandartScaler() Normalizasyonu"
#normalizasyon="StandartScaler()"
#uygulama="RobustScaler() Normalizasyonu"
#normalizasyon="RobustScaler()"
#uygulama="Min-Max Normalizasyonu"
#normalizasyon="min-max"
#uygulama=""
#normalizasyon=""
uygulama="Karekök Normalizasyonu"
normalizasyon="karekök"
data = pd.read_csv("egitim.csv", engine='python')
data2 = pd.read_csv("test.csv", engine='python')
#print(data.keys()) ##data keyleri getir.
###girdiler ilk beş satır
X1=pd.DataFrame(data=data.yorum,columns=data.yorum)
#print(X.head())
###çıkıtlar ilk beş satır
y1=pd.DataFrame(data=data.durum,columns=["durum"])
#print(y.head())
#print(y.kategori.value_counts()) ###kategorilere göre yorum sayılarını getirir.
#eğitim test verileri ayırma başlama
sentences_training = [doc for doc in data.iloc[:,0]]
#print(sentences_training[0:10])
sentences_testing = [doc2 for doc2 in data2.iloc[:,0]]
#print(sentences_testing)
classification_training = [doc for doc in data.iloc[:,1]]
y = data.iloc[:,1].values
y2 = data2.iloc[:,1].values
#print(y)
##tfidf vektör ile kelimeleri matrise dönüştür
vectorizer = TfidfVectorizer(min_df = 0.1, use_idf = True,analyzer='word', lowercase =
True)
sen_train_vector = vectorizer.fit_transform(sentences_training)
vectorizertest = TfidfVectorizer(min_df = 0.1, use_idf = True,analyzer='word',
lowercase = True)
sen_train_vectortest= vectorizertest.fit_transform(sentences_testing)
#print(classification_training.toarray())
#X=sen_train_vector.toarray()
X_train=sen_train_vector.toarray()
X_test=sen_train_vectortest.toarray()
#print(X_train)
# Kategorik olan hedef değişkeni nümerik yap
encoder = LabelEncoder()
y= encoder.fit_transform(y)

```

```

# Nümerik hedef değişkenden gölge değişkenler yarat
y = np_utils.to_categorical(y)
#print(y)
y2= encoder.fit_transform(y2)
y2 = np_utils.to_categorical(y2)
#print(y2)
y_train=y
y_test=y2
#print(y2)
#print(y_test.shape)
#print(X_test.shape)
#print(y_train.shape)
#print(X_train.shape)
if X_train.shape[1]!=X_test.shape[1]:
    if X_train.shape[1]<X_test.shape[1]:
        X_test = X_test[:, 0:X_train.shape[1]]
    if X_train.shape[1]>X_test.shape[1]:
        X_train = X_test[:, 0:X_test.shape[1]]
#X_test=X_test[:,0:36]
###normalizasyon yap
##karekök normalizasyonu
exp_data = np.exp(X_train)
exp_data2 = np.exp(X_test)
X_train=sqrt_normalize(exp_data)
X_test=sqrt_normalize(exp_data2)
###RobustScaler
from sklearn.preprocessing import RobustScaler
"""scaler = preprocessing.RobustScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)"""
##min max uygula
"""from sklearn.preprocessing import MinMaxScaler
mms = MinMaxScaler()
X_train = mms.fit_transform(X_train) ##Eğitim setine normalizasyon uygulamak
X_test = mms.transform(X_test) ##Test setine normalizasyon uygulamak
"""###z score uygula
"""from scipy.stats import zscore
X_train=zscore(X_train, ddof=1)
X_test=zscore(X_test, ddof=1)"""
### StandardScaler
"""scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test) ##Test setine normalizasyon uygulamak
"""
y_train=y
#print(X_test.shape)
#print(X_test)
##ağı eğitme
# model kur
classifier = Sequential()
input_dim=X_train.shape[1]

```

```

#print(X_train.shape[1])
#classifier.add(Dense(64,activation="relu",input_shape=X_train[1].shape))
classifier.add(Dense(kernel_initializer = 'uniform', input_dim = input_dim, units = 5,
activation = 'relu'))
classifier.add(LayerNormalization(axis=1 , center=True , scale=True))
classifier.add(Dense(input_dim,activation="relu"))
#classifier.add(WeightNormalization(Dense(512, activation='relu')))
classifier.add(Dense(2,activation="sigmoid"))
#classifier.add(Dense(kernel_initializer = 'uniform', units = 3, activation =
'softmax'))
###optimizer: öğrenme oranını kontrol eder. Çoğundan adm kullanırlı. adam, öğrenme
boyunca öğrenme oranını ayarlar.
###çoklu sınıflandırma olduğundan categorical ikili olsaydı binary kullanılacaktı
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
###batch_size: Model eğitilirken kaçar örnek alınacağını belirler.
###validation_split modelin eğitimini iyileştirmek için kullanılır.
#print(y_train)
history=classifier.fit(X_train, y_train, batch_size=256,
epochs=500,validation_data=(X_test,y_test))
loss, accuracy = classifier.evaluate(X_train, y_train, verbose=False)
print("Eğitim Doğruluğu: {:.4f}".format(accuracy))
loss, accuracy = classifier.evaluate(X_test, y_test, verbose=False)
print("Test Doğruluğu: {:.4f}".format(accuracy))
##precision(kesinlik),f1score,recall(duyarlılık) değerleri
y_pred = classifier.predict(X_test)
y_pred=np.argmax(y_pred, axis=1)
y_test2=np.argmax(y_test, axis=1)
#print(y_pred)
#print(y_test2)
print(classification_report(y_test2, y_pred))
#print(accuracy_score(y_test, y_pred.round(), normalize=True))
###grafik eğitim ve doğruluk değerleri dağılım grafiği
plt.subplot(2,1,1)
plt.title("Manuel Olarak Oluşturulan Eğitim Verisi Üzerinde "+uygulama+"\n YSA
Uygulaması Grafiği",fontsize=8)
plt.plot(history.history["accuracy"])
plt.plot(history.history["val_accuracy"])
plt.xlabel("İterasyon")
plt.ylabel("Doğrulama Oranı")
plt.legend(["Eğitim","Test"])
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
plt.xlabel("İterasyon")
plt.ylabel("Hata Oranı")
plt.legend(["Eğitim","Test"])
plt.grid(True)
plt.tight_layout()
plt.show()
### test verisinde seçilen bir verinin tahmin-gerçek sonucu grafikte gösterme

```

```

kategoriler = ['olumlu', 'olumsuz']
test_et=[]
veri1=[]
veri2=[]
labels = []
olumlutahmin=0
olumsuztahmin=0
olumlugercek=0
olumsuzgercek=0
olumsuzbildi=0
olumlubildi=0
for t in range(len(X_test)):
    aralik=t+1
    test_et.append(classifier.predict(X_test[t:aralik]))
#print("test verisi\n",test_et)
for i in range(len(X_test)):
    ata=test_et[i]
    print(ata[0][0])
    print(ata[0][1])
    veri1.append(ata[0][0])
    veri2.append(ata[0][1])
    labels.append(kategoriler[np.argmax(y_test[i:i+1])])
    ##tahminsel sınıfını bul
    if ata[0][0]>=ata[0][1]:
        sinifi="olumlu"
        olumlutahmin=olumlutahmin+1
    if ata[0][1]>ata[0][0]:
        sinifi="olumsuz"
        olumsuztahmin=olumsuztahmin+1
    if kategoriler[np.argmax(y_test[i:i+1])]=="olumlu":
        olumlugercek=olumlugercek+1
    if kategoriler[np.argmax(y_test[i:i+1])]=="olumsuz":
        olumsuzgercek=olumsuzgercek+1
    ##gerçekte olumlu sınıfa ait verinin kaç tanesini doğru bildi
    if sinifi=="olumlu" and kategoriler[np.argmax(y_test[i:i+1])]=="olumlu":
        olumlubildi=olumlubildi+1
    ##gerçekte olumsuz sınıfa ait verinin kaç tanesini doğru bildi
    if sinifi=="olumsuz" and kategoriler[np.argmax(y_test[i:i+1])]=="olumsuz":
        olumsuzbildi=olumsuzbildi+1

    print("Tahmin sonucu:", kategoriler[np.argmax(test_et[i])])
    print("Gerçek sonuç:", kategoriler[np.argmax(y_test[i:i+1])])
###ilk grafik test verisinin tahmini ve gerçek değerleri dağılımı
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tveri=[olumlutahmin,olumlugercek]
tveri2=[olumsuztahmin,olumsuzgercek]
labelgrafik=["Tahmin Edilen Sınıf Dağılımı","Gerçek Sınıf Dağılımı"]
rects1=plt.bar(np.arange(len(tveri)), tveri, width=width,color='red')
rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["olumlu","olumsuz"])

```

```

plt.title("Sınıflandırma Sonucu ",fontsize=18)
plt.ylabel("Sınıflandırma Değeri",fontsize=12)
plt.xlabel("Manuel olarak oluşturulan eğitim veri setine "+str(normalizasyon)+"
normalizasyon uygulanarak yapılan YSA sınıflandırması sonucu olumlu yorum sayısı
"+str(olumlutahmin)+"\n olumsuz yorum sayısı "+str(olumsuztahmin)+" şeklindedir. Test
verilerinin gerçek sınıflarına bakıldığında olumlu yorum sayısı "+str(olumlugercek)+"
olumsuz yorum sayısı "+ str(olumsuzgercek)+" şeklindedir.",fontsize=10)
x = np.arange(len(labelgrafik)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labelgrafik)
autolabel(rects1)
autolabel(rects2)
plt.show()

###ikinci grafik gerçekte olumlu veya olumsuz olan verinin kaç tanesini ağ doğru tahmin
etti dağılımı
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tkiyas=[olumlubildi,olumlugercek,olumsuzbildi,olumsuzgercek]
#tveri2=[olumsuztahmin,olumsuzgercek]
labeldeger=["Tahmin Edilen Olumlu Sayısı","Gerçek Olumlu Sayısı","Tahmin Edilen Olumsuz
Sayısı","Gerçek Olumsuz Sayısı"]
rects1=plt.bar(np.arange(len(tkiyas)), tkiyas, width=width,color='orange')
#rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["Sınıf Değerleri"])
plt.title("Gerçek Sınıf-Tahmini Sınıf Dağılım Grafiği",fontsize=18)
plt.ylabel("Veri Sayısı",fontsize=12)
toplamdeger=olumlubildi+olumsuzbildi
plt.xlabel("Ağ modelinin doğru tahmin ettiği sınıf kategorisi sayısı:
"+str(toplamdeger),fontsize=10)
x = np.arange(len(labeldeger)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labeldeger)
autolabel(rects1)
plt.show()

###üçüncü grafik 15 test verisinin ayrı ayrı olumlu olumsuz dağılım oranları ve gerçek
sınıfsal konumları
width =0.3
f, ax = plt.subplots(figsize=(12,5))
plt.bar(np.arange(15), veri1[:15], width=width,color='red')
plt.bar(np.arange(15)+ width, veri2[:15], width=width,color='blue')
plt.legend(["olumlu","olumsuz"])
plt.title("Sınıflandırma Sonucu ",fontsize=18)
plt.ylabel("Sınıflandırma Değeri",fontsize=12)
plt.xlabel("Gerçekte Bulunduğu Sınıf",fontsize=12)
x = np.arange(15)
ax.set_xticks(x)
ax.set_xticklabels(labels[:15])
plt.show()

```

Train_test_split kullanılarak yapılan sınıflandırmadaki Python kodları

```

from random import random
import matplotlib
import pandas as pd
import numpy as np
import tensorflow_addons
from sklearn import preprocessing
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from keras.utils import np_utils
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LayerNormalization
from sklearn.metrics import classification_report
from tensorflow_addons.layers import WeightNormalization
from sklearn.preprocessing import StandardScaler
from keras import backend as K
#####çizim kütüphanesi çiz ve sonuçları göster
import matplotlib.pyplot as plt
matplotlib.axes.Axes.bar
matplotlib.pyplot.bar
matplotlib.axes.Axes.annotate
matplotlib.pyplot.annotate
from tensorflow.python.layers.normalization import BatchNormalization
def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying its height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{} '.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3), # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')
###normalizasyon
##standart sapma normalize
def standard_norm(arr):
    mean = np.mean(arr)
    stdev = np.std(arr)
    return (arr-mean)/stdev
def log_normalize(arr):
    return np.log(arr)
def sqrt_normalize(arr):
    return np.sqrt(arr)

```

```

plt.style.use('ggplot')
dizi=[]
a=2
#egitimtur="Olumlu yorum sayısı fazla"
#egitimtur2="Olumlu Yorum Sayısı Fazla"
egitimtur="Olumsuz yorum sayısı fazla"
egitimtur2="Olumsuz Yorum Sayısı Fazla"
#egitimtur="Dengeli yorum sayılı"
#egitimtur2="Dengeli Yorum Sayılı"
#uygulama="Min-Max Normalizasyonu"
#normalizasyon="min-max"
#uygulama="Z-score Normalizasyonu"
#normalizasyon="z-score"
#uygulama="StandartScaler() Normalizasyonu"
#normalizasyon="StandartScaler()"
#uygulama="RobustScaler() Normalizasyonu"
#normalizasyon="RobustScaler()"
#uygulama="Min-Max Normalizasyonu"
#normalizasyon="min-max"
#uygulama=""
#normalizasyon=""
uygulama="Karekök Normalizasyonu"
normalizasyon="karekök"
data = pd.read_csv("veriseti.csv", engine='python')
#print(data.keys()) ##data keyleri getir.
###girdiler ilk beş satır
X1=pd.DataFrame(data=data.yorum,columns=data.yorum)
#print(X.head())
###çıkıtlar ilk beş satır
y1=pd.DataFrame(data=data.durum,columns=["durum"])
#print(y.head())
#print(y.kategori.value_counts()) ###kategorilere göre yorum sayılarını getirir.
#eğitim test verileri ayırma başlama
sentences_training = [doc for doc in data.iloc[:,0]]
#print(sentences_training[0:10])
classification_training = [doc for doc in data.iloc[:,1]]
y = data.iloc[:,1].values
#print(y)
###tfidf vektör ile kelimeleri matrise dönüştür
vectorizer = TfidfVectorizer(min_df = 0.1, use_idf = True,analyzer='word', lowercase =
True)
sen_train_vector = vectorizer.fit_transform(sentences_training)
X=sen_train_vector.toarray()
print(X)
#print(X_train)
# Kategorik olan hedef değişkeni nümerik yap
encoder = LabelEncoder()
y= encoder.fit_transform(y)
# Nümerik hedef değişkenden gölge değişkenler yarat
y = np_utils.to_categorical(y)
print(y)

```

```

##veri setindeki olumlu olumsuz yorumların dağılımları
plt.hist(data.durum,bins='auto',color='#008080',label="Duygu Durumu")
plt.xlabel("Müşteri Duygu Durumları")
plt.ylabel("Dağılımları")
plt.legend()
plt.title("Müşterilerin Duygu Dağılım Grafiği")
plt.show()

##eğitim ve test verilerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
random_state=42) ##eğitim için %80 test için yüzde 20 veri ayırdık.
#print(y2)
#print(y_test.shape)
#print(X_test.shape)
#print(y_train.shape)
#print(X_train.shape)
###normalizasyon yap
##karekök normalizasyonu
exp_data = np.exp(X_train)
exp_data2 = np.exp(X_test)
X_train=sqrt_normalize(exp_data)
X_test=sqrt_normalize(exp_data2)
###RobustScaler
from sklearn.preprocessing import RobustScaler
"""scaler = preprocessing.RobustScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)"""
##min max uygula
"""from sklearn.preprocessing import MinMaxScaler
mms = MinMaxScaler()
X_train = mms.fit_transform(X_train) ##Eğitim setine normalizasyon uygulamak
X_test = mms.transform(X_test) ##Test setine normalizasyon uygulamak
"""

###z score uygula
"""from scipy.stats import zscore
X_train=zscore(X_train, ddof=1)
X_test=zscore(X_test, ddof=1)"""
### StandardScaler
"""scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test) ##Test setine normalizasyon uygulamak
"""

#print(X_test.shape)
#print(X_test)
##ağı eğitme
# model kur
classifier = Sequential()
input_dim=X_train.shape[1]
#print(X_train.shape[1])
#classifier.add(Dense(64,activation="relu",input_shape=X_train[1].shape))
classifier.add(Dense(kernel_initializer = 'uniform', input_dim = input_dim, units = 5,

```

```

activation = 'relu'))
classifier.add(LayerNormalization(axis=1 , center=True , scale=True))
classifier.add(Dense(input_dim,activation="relu"))
#classifier.add(WeightNormalization(Dense(512, activation='relu'))))
classifier.add(Dense(2,activation="sigmoid"))
#classifier.add(Dense(kernel_initializer = 'uniform', units = 3, activation =
'softmax'))
###optimizer: öğrenme oranını kontrol eder. Çoğundan adm kullanırlı. adam, öğrenme
boyunca öğrenme oranını ayarlar.
###çoklu sınıflandırma olduğundan categorical ikili olsaydı binary kullanılacaktı
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
###batch_size: MÖdel eğitilirken kaçır örnek alınacağını belirler.
###validation_split modelin eğitimini iyileştirmek için kullanılır.
#print(y_train)
history=classifier.fit(X_train, y_train, batch_size=256,
epochs=500,validation_data=(X_test,y_test))
loss, accuracy = classifier.evaluate(X_train, y_train, verbose=False)
print("Eğitim Doğruluğu: {:.4f}".format(accuracy))
loss, accuracy = classifier.evaluate(X_test, y_test, verbose=False)
print("Test Doğruluğu: {:.4f}".format(accuracy))
##precision(kesinlik),f1score,recall(duyarlılık) değerleri
y_pred = classifier.predict(X_test)
y_pred=np.argmax(y_pred, axis=1)
y_test2=np.argmax(y_test, axis=1)
#print(y_pred)
#print(y_test2)
print(classification_report(y_test2, y_pred))
#print(accuracy_score(y_test, y_pred.round(), normalize=True))
print(history)
###grafik eğitim ve doğruluk değerleri dağılım grafiği
plt.subplot(2,1,1)
plt.title("Train_test_split Komutu Kullanılarak Oluşturulan\n YSA Uygulaması
("+uygulama+") Grafiği",fontsize=8)
plt.plot(history.history["accuracy"])
plt.plot(history.history["val_accuracy"])
plt.xlabel("İterasyon")
plt.ylabel("Doğrulama Oranı")
plt.legend(["Eğitim","Test"])
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
plt.xlabel("İterasyon")
plt.ylabel("Hata Oranı")
plt.legend(["Eğitim","Test"])
plt.grid(True)
plt.tight_layout()
plt.show()
### test verisinde seçilen bir verinin tahmin-gerçek sonucu grafikte gösterme
kategoriler = ['olumlu', 'olumsuz']
test_et=[]

```

```

veri1=[]
veri2=[]
labels = []
olumlutahmin=0
olumsuztahmin=0
olumlugercek=0
olumsuzgercek=0
olumsuzbildi=0
olumlubildi=0
for t in range(len(X_test)):
    aralik=t+1
    test_et.append(classifier.predict(X_test[t:aralik]))
#print("test verisi\n",test_et)
for i in range(len(X_test)):
    ata=test_et[i]
    print(ata[0][0])
    print(ata[0][1])
    veri1.append(ata[0][0])
    veri2.append(ata[0][1])
    labels.append(kategoriler[np.argmax(y_test[i:i+1])])
    ##tahminsel sınıfını bul
    if ata[0][0]>=ata[0][1]:
        sinifi="olumlu"
        olumlutahmin=olumlutahmin+1
    if ata[0][1]>ata[0][0]:
        sinifi="olumsuz"
        olumsuztahmin=olumsuztahmin+1
    if kategoriler[np.argmax(y_test[i:i+1])]=="olumlu":
        olumlugercek=olumlugercek+1
    if kategoriler[np.argmax(y_test[i:i+1])]=="olumsuz":
        olumsuzgercek=olumsuzgercek+1
    ##gerçekte olumlu sınıfa ait verinin kaç tanesini doğru bildi
    if sinifi=="olumlu" and kategoriler[np.argmax(y_test[i:i+1])]=="olumlu":
        olumlubildi=olumlubildi+1
    ##gerçekte olumsuz sınıfa ait verinin kaç tanesini doğru bildi
    if sinifi=="olumsuz" and kategoriler[np.argmax(y_test[i:i+1])]=="olumsuz":
        olumsuzbildi=olumsuzbildi+1

    print("Tahmin sonucu:", kategoriler[np.argmax(test_et[i])])
    print("Gerçek sonuç:", kategoriler[np.argmax(y_test[i:i+1])])

###ilk grafik test verisinin tahmini ve gerçek değerleri dağılımı
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tveri=[olumlutahmin,olumlugercek]
tveri2=[olumsuztahmin,olumsuzgercek]
labelgrafik=["Tahmin Edilen Sınıf Dağılımı","Gerçek Sınıf Dağılımı"]
rects1=plt.bar(np.arange(len(tveri)), tveri, width=width,color='red')
rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["olumlu","olumsuz"])
plt.title("Sınıflandırma Sonucu ",fontsize=18)

```

```

plt.ylabel("Sınıflandırma Değeri",fontsize=12)
plt.xlabel("train_test_split komutu kullanılarak oluşturulan veri setine
"+str(normalizasyon)+" normalizasyon uygulanarak yapılan YSA sınıflandırması sonucu
olumlu yorum sayısı "+str(olumlutahmin)+"\n olumsuz yorum sayısı "+str(olumsuztahmin)+"
şeklindedir. Test verilerinin gerçek sınıflarına bakıldığında olumlu yorum sayısı
"+str(olumlugercek)+" olumsuz yorum sayısı "+ str(olumsuzgercek)+"
şeklindedir.",fontsize=10)
x = np.arange(len(labelgrafik)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labelgrafik)
autolabel(rects1)
autolabel(rects2)
plt.show()
###ikinci grafik gerçekte olumlu veya olumsuz olan verinin kaç tanesini ağ doğru tahmin
etti dağılımı
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tkiyas=[olumlubildi,olumlugercek,olumsuzbildi,olumsuzgercek]
#tveri2=[olumsuztahmin,olumsuzgercek]
labeldeger=["Tahmin Edilen Olumlu Sayısı","Gerçek Olumlu Sayısı","Tahmin Edilen Olumsuz
Sayısı","Gerçek Olumsuz Sayısı"]
rects1=plt.bar(np.arange(len(tkiyas)), tkiyas, width=width,color='orange')
#rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["Sınıf Değerleri"])
plt.title("Gerçek Sınıf-Tahmini Sınıf Dağılım Grafiği",fontsize=18)
plt.ylabel("Veri Sayısı",fontsize=12)
toplamdeger=olumlubildi+olumsuzbildi
plt.xlabel("Ağ modelinin doğru tahmin ettiği sınıf kategorisi sayısı:
"+str(toplamdeger),fontsize=10)
x = np.arange(len(labeldeger)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labeldeger)
autolabel(rects1)
plt.show()
###üçüncü grafik 15 test verisinin ayrı ayrı olumlu olumsuz dağılım oranları ve gerçek
sınıfsal konumları
width =0.3
f, ax = plt.subplots(figsize=(12,5))
plt.bar(np.arange(15), veri1[:15], width=width,color='red')
plt.bar(np.arange(15)+ width, veri2[:15], width=width,color='blue')
plt.legend(["olumlu","olumsuz"])
plt.title("Sınıflandırma Sonucu ",fontsize=18)
plt.ylabel("Sınıflandırma Değeri",fontsize=12)
plt.xlabel("Gerçekte Bulunduğu Sınıf",fontsize=12)
x = np.arange(15)
ax.set_xticks(x)
ax.set_xticklabels(labels[:15])
plt.show()

```

Çarpaz doğrulama kullanılarak yapılan sınıflandırmada Python kodları

```

from random import random
import PIL
import matplotlib
import pandas as pd
import numpy as np
import sklearn
import tensorflow_addons
from PIL import ImageTk
from sklearn import preprocessing, metrics
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from keras.utils import np_utils
from sklearn.model_selection import train_test_split, cross_val_score, KFold
from sklearn.metrics import accuracy_score, plot_confusion_matrix
from sklearn.metrics import confusion_matrix
import seaborn as sn
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LayerNormalization
from sklearn.metrics import classification_report
from tkinter import *
from tensorflow_addons.layers import WeightNormalization
from sklearn.preprocessing import StandardScaler
from keras import backend as K
#####çizim kütüphanesi çiz ve sonuçları göster
import matplotlib.pyplot as plt
matplotlib.axes.Axes.bar
matplotlib.pyplot.bar
matplotlib.axes.Axes.annotate
matplotlib.pyplot.annotate
from tensorflow.python.layers.normalization import BatchNormalization
def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying its height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}' .format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3), # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')
from tensorflow.python.layers.normalization import BatchNormalization

###normalizasyon
##standart sapma normalize
def standard_norm(arr):

```

```

    mean = np.mean(arr)
    stdev = np.std(arr)
    return (arr-mean)/stdev
def log_normalize(arr):
    return np.log(arr)
def sqrt_normalize(arr):
    return np.sqrt(arr)

plt.style.use('ggplot')
dizi=[]
#uygulama="Min-Max Normalizasyonu"
#normalizasyon="min-max"
#uygulama="Z-score Normalizasyonu"
#normalizasyon="z-score"
#uygulama="StandartScaler() Normalizasyonu"
#normalizasyon="StandartScaler()"
#uygulama="RobustScaler() Normalizasyonu"
#normalizasyon="RobustScaler()"
#uygulama=""
#normalizasyon=""
uygulama="Karekök Normalizasyonu"
normalizasyon="karekök"
data = pd.read_csv("veriseti.csv", engine='python')
#print(data.keys()) ##data keyleri getir.
###girdiler ilk beş satır
X1=pd.DataFrame(data=data.yorum,columns=data.yorum)
#print(X.head())
###çıktılar ilk beş satır
y1=pd.DataFrame(data=data.durum,columns=["durum"])
#print(y.head())
#print(y1.kategori.value_counts()) ###kategorilere göre yorum sayılarını getirir.
#eğitim test verileri ayırma başlama
sentences_training = [doc for doc in data.iloc[:,0]]
#print(sentences_training[0:10])
classification_training = [doc for doc in data.iloc[:,1]]
y = data.iloc[:,1].values
class_names = data.durum
#print(list(data.durum))
print(y)
###tfidf vektör ile kelimleri matrise dönüştür
vectorizer = TfidfVectorizer(min_df = 0.1, use_idf = True,analyzer='word', lowercase =
True)
sen_train_vector = vectorizer.fit_transform(sentences_training)
#print(classification_training.toarray())
#X=sen_train_vector.toarray()
X=sen_train_vector.toarray()
#print(X_train)
# Kategorik olan hedef değişkeni nümerik yap
encoder = LabelEncoder()
y= encoder.fit_transform(y)
# Nümerik hedef değişkenden gölge değişkenler yarat

```

```

y = np_utils.to_categorical(y)
###normalizasyon yap
##karekök normalizasyonu
exp_data = np.exp(X)
X=sqrt_normalize(exp_data)
###RobustScaler
"""from sklearn.preprocessing import RobustScaler
scaler = preprocessing.RobustScaler()
X = scaler.fit_transform(X)
"""

##min max uygula
"""from sklearn.preprocessing import MinMaxScaler
mms = MinMaxScaler()
X = mms.fit_transform(X) ##Eğitim setine normalizasyon uygulamak
"""

###z score uygula
"""from scipy.stats import zscore
X=zscore(X, ddof=1)
"""

### StandardScaler
"""scaler = StandardScaler()
X = scaler.fit_transform(X)
"""

#print(X_test.shape)
#print(X_test)
k=10
kfold = KFold(n_splits=k, shuffle=True, random_state=7)
cvscoresTrain = []
cvscoresTesting = []
cvscores = []
cvscorestest = []
ortalamakacbildi = []
ortalamaoran= []
dogruolumlu = []
yanlisolumlu = []
dogruolumsuz = []
yanlisolumsuz = []
input_dim=X.shape[1]
res=1
#print(X_train.shape,y_train.shape)
for train, test in kfold.split(X, y):
    # create model
    model = Sequential()
    model.add(Dense(kernel_initializer = 'uniform', input_dim = input_dim, units = 5,
activation = 'relu'))
    model.add(LayerNormalization(axis=1 , center=True , scale=True))
    model.add(Dense(input_dim, activation="relu"))
    model.add(Dense(2, activation='sigmoid'))
    # Compile model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

```

```

# Fit the model
history = model.fit(X[train], y[train], validation_data=(X[test], y[test]),
batch_size=256, epochs=500)
loss, accuracy = model.evaluate(X[train], y[train], verbose=False)
print("Eğitim Doğruluğu: {:.4f}".format(accuracy))
scores = model.evaluate(X[train], y[train], verbose=0)
#print("%s: %.2f%%" % (model.metrics_names[1], scores[1] * 100))
cvscores.append(round(accuracy,4))
#print("Eğitim dizi: ")
#print(cvscores)
loss, accuracy = model.evaluate(X[test], y[test], verbose=False)
print("Test Doğruluğu: {:.4f}".format(accuracy))
#print("test evaluate")
#print(model.evaluate(X[test], y[test], verbose=0))
scorestest = model.evaluate(X[test], y[test], verbose=0)
#print("%s: %.2f%%" % (model.metrics_names[1], scores[1] * 100))
####ortalamalarını almak için diziye ata
cvscoresTrain.append(scores[1] * 100)
cvscoresTesting.append(scorestest[1] * 100)
cvscorestest.append(round(accuracy,4))
#print("Test dizi: ")
#print(cvscorestest)
#print(X[test])
#print(y[test])
y_pred = model.predict(X[test])
y_pred = np.argmax(y_pred, axis=1)
y_test2 = np.argmax(y[test], axis=1)
#print(y_pred)
#print(y_test2)
#print(list(y_test2))
#print(list(y_test2).count(1))
#print(list(y_test2).count(0))
#target_names=['olumlu','olumsuz']
##f1score recall precision
print(classification_report(y_test2, y_pred))
dolumlu, yolumlu, yolumsuz, dolumsuz = metrics.confusion_matrix(y_test2,
y_pred).ravel()
dogruolumlu.append(dolumlu)
yanlisolumlu.append(yolumlu)
yanlisolumsuz.append(yolumsuz)
dogruolumsuz.append(dolumsuz)
###grafik eğitim ve doğruluk değerleri dağılım grafiği
plt.subplot(2, 1, 1)
plt.title("Çarpaz Doğrulama Kullanılarak Oluşturulan\n YSA Uygulaması (" + uygulama
+ ") Grafiği",
        fontsize=8)
plt.plot(history.history["accuracy"])
plt.plot(history.history["val_accuracy"])
plt.xlabel("İterasyon")
plt.ylabel("Doğrulama Oranı")
plt.legend(["Eğitim", "Test"])

```

```

plt.grid(True)
plt.subplot(2, 1, 2)
plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
plt.xlabel("İterasyon")
plt.ylabel("Hata Oranı")
plt.legend(["Eğitim", "Test"])
plt.grid(True)
plt.tight_layout()
plt.savefig('kaydet2/'+str(normalizasyon)+'_'+str(res)+'.png',dpi=100)
plt.show()
res=res+1

print("accuracy",cvscores)
print("accuracytest",cvscorestest)
ortalamaTrain= sum(cvscors)/k
ortalamaTest= sum(cvscorstest)/k
print("sonuc: ",ortalamaTrain)
print("Eğitim Doğruluğu: {:.4f}".format(ortalamaTrain))
print("Test Doğruluğu: {:.4f}".format(ortalamaTest))
print("Eğitim Doğruluğu: %.2f%% (+/- %.2f%%)" % (np.mean(cvscors), np.std(cvscors)))
print("Test Doğruluğu: %.2f%% (+/- %.2f%%)" % (np.mean(cvscorstest),
np.std(cvscorstest)))
print("Doğru Olumlu: ",dogruolumlu)
print("Yanlış Olumlu: ",yanlisolumlu)
print("Doğru Olumsuz: ",dogruolumsuz)
print("Yanlış Olumsuz: ",yanlisolumsuz)
##ortalamayı bul
ortadolumlu=int(round(sum(dogruolumlu)/k, 0))
ortayolumlu= int(round(sum(yanlisolumlu)/k, 0))
ortadolumsuz= int(round(sum(dogruolumsuz)/k, 0))
ortayolumsuz= int(round(sum(yanlisolumsuz)/k, 0))
print("Doğru Olumlu: ", ortadolumlu)
print("Yanlış Olumlu: ",ortayolumlu)
print("Doğru Olumsuz: ",ortadolumsuz)
print("Yanlış Olumsuz: ",ortayolumsuz)
#####her döngüdeki eğitim dogruluk oranları ekranda göster tablo
sutun=1
ekran=Tk()
yazegitim=Label(ekran,text="Eğitim Doğruluğu",font=("Times-New-Roman", 10))
yazetest = Label(ekran, text="Test Doğruluğu",font=("Times-New-Roman", 10))
yazegitim.grid(row=1,column=0,sticky=W)
yazetest.grid(row=2,column=0,sticky=W)
dongu1=Label(ekran,text="Döngü 1",font=("Times-New-Roman", 10))
dongu2 = Label(ekran, text="Döngü 2",font=("Times-New-Roman", 10))
dongu3=Label(ekran,text="Döngü 3",font=("Times-New-Roman", 10))
dongu4 = Label(ekran, text="Döngü 4",font=("Times-New-Roman", 10))
dongu5 = Label(ekran, text="Döngü 5",font=("Times-New-Roman", 10))
dongu6 = Label(ekran, text="Döngü 6",font=("Times-New-Roman", 10))
dongu7 = Label(ekran, text="Döngü 7",font=("Times-New-Roman", 10))
dongu8 = Label(ekran, text="Döngü 8",font=("Times-New-Roman", 10))
dongu9 = Label(ekran, text="Döngü 9",font=("Times-New-Roman", 10))

```

```

dongu10 = Label(ekran, text="Döngü 10", font=("Times-New-Roman", 10))
dongu1.grid(row=0, column=1)
dongu2.grid(row=0, column=2)
dongu3.grid(row=0, column=3)
dongu4.grid(row=0, column=4)
dongu5.grid(row=0, column=5)
dongu6.grid(row=0, column=6)
dongu7.grid(row=0, column=7)
dongu8.grid(row=0, column=8)
dongu9.grid(row=0, column=9)
dongu10.grid(row=0, column=10)
yazE="yazE"
yazT="yazT"
for y in range(len(cvscores)):
    satir = 1
    degerE=yazE+str(y)
    degerT=yazT+str(y)
    degerE=Label(ekran, text=cvscores[y], font=("Times-New-Roman", 10))
    degerT = Label(ekran, text=cvscorestest[y], font=("Times-New-Roman", 10))
    degerE.grid(row=satir, column=sutun)
    satir=satir+1
    degerT.grid(row=satir, column=sutun)
    sutun=sutun+1
ekran.mainloop()
###ilk grafik test verisinin tahmini ve gerçek değerleri dağılımı
olumlutahmin=ortayolumsuz+ortadolumlu
olumlugercek=ortadolumlu+ortayolumlu
olumsuztahmin=ortayolumlu+ortadolumsuz
olumsuzgercek=ortayolumsuz+ortadolumsuz
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tveri=[olumlutahmin,olumlugercek]
tveri2=[olumsuztahmin,olumsuzgercek]
labelgrafik=["Tahmin Edilen Sınıf Dağılımı","Gerçek Sınıf Dağılımı"]
rects1=plt.bar(np.arange(len(tveri)), tveri, width=width,color='red')
rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["olumlu", "olumsuz"])
plt.title("Sınıflandırma Sonucu ", fontsize=18)
plt.ylabel("Sınıflandırma Değeri", fontsize=12)
plt.xlabel("Çarpaz doğrulama kullanılarak oluşturulan veri setine
"+str(normalizasyon)+" normalizasyon uygulanarak yapılan YSA sınıflandırması sonucu
olumlu yorum sayısı "+str(olumlutahmin)+",\nolumsuz yorum sayısı "+str(olumsuztahmin)+"
şeklinde dir. Test verilerinin gerçek sınıflarına bakıldığında olumlu yorum sayısı
"+str(olumlugercek)+",olumsuz yorum sayısı "+str(olumsuzgercek)+"
şeklinde dir.", fontsize=10)
x = np.arange(len(labelgrafik)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labelgrafik)
autolabel(rects1)
autolabel(rects2)
plt.show()

```

```

###ikinci grafik gerçekte olumlu veya olumsuz olan verinin kaç tanesini ağ doğru tahmin
etti dağılımı
olumlubildi = ortadolumlu
olumsuzbildi= ortadolumsuz
width =0.4
f, ax = plt.subplots(figsize=(12,7))
tkiyas=[olumlubildi,olumlugercek,olumsuzbildi,olumsuzgercek]
labeldeger=["Tahmin Edilen Olumlu Sayısı","Gerçek Olumlu Sayısı","Tahmin Edilen Olumsuz
Sayısı","Gerçek Olumsuz Sayısı"]
rects1=plt.bar(np.arange(len(tkiyas)), tkiyas, width=width,color='orange')
#rects2=plt.bar(np.arange(len(tveri2))+ width, tveri2, width=width,color='blue')
plt.legend(["Sınıf Değerleri"])
plt.title("Gerçek Sınıf-Tahmini Sınıf Dağılım Grafiği",fontsize=18)
plt.ylabel("Veri Sayısı",fontsize=12)
toplamdeger=olumlubildi+olumsuzbildi
plt.xlabel("Ağ modelinin doğru tahmin ettiği sınıf kategorisi sayısı:
"+str(toplamdeger),fontsize=10)
x = np.arange(len(labeldeger)) #label konum
ax.set_xticks(x)
ax.set_xticklabels(labeldeger)
autolabel(rects1)
plt.show()

```

KAYNAKÇA

- Ahmetoğlu, H., Daş, R. (2020). Türkçe otel yorumlarıyla eğitilen kelime vektörü modellerinin duygu analizi ile incelenmesi. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 24 (2), 455-463.
- Aksu, M. Ç., & Karaman, E. (2017). Karar ağaçları ile bir web sitesinde link analizi ve tespiti. *Acta Infologica*, 1(2), 84-91.
- (2020). FastText ve kelime çantası kelime temsil yöntemlerinin turistik mekanlar için yapılan Türkçe incelemeler kullanılarak karşılaştırılması. *Avrupa Bilim ve Teknoloji Dergisi*, (20), 311-320.
- Al-Momani, O., Alwada'n, T., Mohammad, A. H. (2016). Arabic text categorization using support vector machine, Naive Bayes and neural network. *GSTF Journal of Computing (JOC)*, 5 (1), 108-115.
- Albayrak, A., vd. (2016, 17-18 Eylül). *Makine öğrenmesi yöntemleri ile duygu analizi*. International Artificial Intelligence and Data Processing Symposium (IDAP'16), Malatya, Turkey.
- Alghoul, A., vd. (2018). Email classification using artificial neural network. *International Journal of Academic Engineering Research (IJAER)*, 2 (11), 8-14.
- Alıcı, S., vd. (2020). Metin sınıflandırmada yapay sinir ağları ile bitcoin fiyatları ve sosyal medyadaki beklentilerin analizi. *Mehmet Akif Ersoy Üniversitesi Uygulamalı Bilimler Dergisi*, 106-126.
- Ali, M. S., vd. (2017). Comparison of feature selection methods in text classification on highly skewed datasets. *First international conference on latest trends in electrical engineering and computing technologies (INTELLECT)*, 1-8.
- Almatarneh, S., & Gamallo, P. (2019). Comparing supervised machine learning strategies and linguistic features to search for very negative opinions. *Information*, 10 (1), 16.
- Amasyalı, M. F., vd. (2012). Türkçe metinlerin sınıflandırılmasında metin temsil yöntemlerinin performans karşılaştırılması. *EMO Bilimsel Dergi*, 2 (4), 95-104.
- Aşlıyan, R., & Tekeli, K. (2016, 3-5 Şubat). *Çok katmanlı algılayıcı, K-NN ve C4.5 metotlarıyla istenmeyen e-postaların tespiti*. 18. Akademik Bilişim Konferansı, Aydın, Turkey.
- Avaroğlu, E., vd. (2019, 21-22 Eylül). *Classification of Turkish News Content by Deep Learning Based LSTM Using Fasttext Model*. International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey.
- Aydın, İ., Başkaya, F., Salur, M. U. (2017, 5-8 Ekim). *Sentiment classification with PSO based weighted K-NN*. International Conference on Computer Science and Engineering (UBMK), Antalya, Turkey.
- Aydilek, İ. B. , Gümüşçü, A. , Taşaltın, R. (2016). 3 farklı filtre modeli öznetelik seçme algoritmalarının kombine edilerek iyileştirilmesi. *AKÜ FEMÜBİD* 16, Özel Sayı, 31- 35.

- Aydoğan, M., & Karcı, A. (2019). Kelime temsil yöntemleri ile kelime benzerliklerinin incelenmesi. *Çukurova Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 34(2), 181-195.
- Aytekin, Ç., Öztekin, U., Sütçü, C. S. (2018). Karar ağacı algoritması ile metin sınıflandırma: Müşteri yorumları örneği. *Uluslararası Sosyal Araştırmalar Dergisi*, 11 (55), 782-792.
- Badur, B., & Kutlu, B. (2009). Yapay sinir ağları ile borsa endeksi tahmini. *Yönetim Dergisi*, 20 (63), 45-40.
- Balodis, K., & Deksnė, D. (2019). FastText-Based intent detection for inflected languages. *Information* 2019, 10, 161.
- Basic Text Classification. (2021). Retrieved from: https://tensorflow.rstudio.com/tutorials/beginners/basic-ml/tutorial_basic_text_classification.
- Bati, C. T., & Ser, G. (2019). Derin sinir ağları ile en iyi modelin belirlenmesi: Mantar verileri üzerine keras uygulaması. *Yüzyüncü Yıl Üniversitesi Tarım Bilimleri Dergisi*, 29 (3), 406-417.
- Borandağ, E., vd. (2016). KNN algoritması ve R dili ile metin madenciliği kullanılarak bilimsel makale tasnifi. *Marmara Fen Bilimleri Dergisi*, 89-94.
- Budak, H. (2018). Özellik seçim yöntemleri ve yeni bir yaklaşım. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi Cilt 22, Özel Sayı*, 21-31.
- Campos, D., Bernardino, J., Silva, R. R. (2019). Text mining in hotel reviews: Impact of words restriction in text classification. In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*, 442-449.
- Chen, K., vd. (2016). Turning from TF-IDF to TF-IGM for term weighting in text classification. *Expert Systems with Applications* 66 (2016), 245-260.
- Cross-validation: Evaluating Estimator Performance. (2021). Retrieved from: https://scikit-learn.org/stable/modules/cross_validation.html
- Çalış, K., Gazdağı, O., Yıldız, O. (2013). Reklam içerikli epostaların metin madenciliği yöntemleri ile otomatik tespiti. *Bilişim Teknolojileri Dergisi*, Cilt: 6, Sayı: 1, 1-7.
- Çınar, A. (2019). Sınıflandırma algoritmaları ile bir metin madenciliği uygulaması. İçinde Balaban, M. E., Kartal, E. (Ed.), *Veri madenciliği ve makine öğrenmesi temel kavramlar, algoritmalar, uygulamalar* (s. 105-140). İstanbul: Çağlayan Kitabevi.
- Çokluk, Ö. (2010). Lojistik regresyon analizi: Kavram ve uygulama. *Kuram ve Uygulamada Eğitim Bilimleri*, 10 (3), 1357-1407.
- Deveci, M., & Yavuz, S. (2012). İstatiksel normalizasyon tekniklerinin yapay sinir ağı performansına etkisi. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, Sayı:40, 167-187.
- Doğan, T., & Uysal, A. K. (2018, 9-11 Kasım). *Tıbbi metin dokümanlarının sınıflandırılmasında terim ağırlıklandırma yöntemlerinin başarımlarının kıyaslanması*. 6th International Symposium on Innovative Technologies in Engineering and Science, Antalya, Turkey.

- Emekli, B., & Selvi, İ. H. (2020, 19-20 Haziran). *GSM operatörlerine yönelik atılan Türkçe tweetlerin derin öğrenme yöntemleriyle duygu analizi*. 4. Uluslararası Fen Bilimleri Kongresi, Online.
- Elmas, B., Yakut, E., Yavuz, S. (2014). Yapay sinir ağları ve destek vektör makineleri yöntemleriyle borsa endeksi tahmini. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 19 (1), 139-157.
- Erdinç, H. Y ., & Güran, A. (2019, 24-26 Nisan). *Semi-supervised Turkish text categorization with Word2Vec, Doc2Vec and Fasttext algorithms*. 27th Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey.
- Erkaya, A. E. (2019). *Text classification based on organizational data using machine learning* (Yayımlanmamış Yüksek Lisans Tezi). Ankara Yıldırım Beyazıt Üniversitesi Fen Bilimleri Enstitüsü, Ankara.
- Göker, H., & Tekedere, H. (2017). FATİH projesine yönelik görüşlerin metin madenciliği yöntemleri ile otomatik değerlendirilmesi. *Bilişim Teknolojileri Dergisi*, 10 (3), 291-299.
- Gürcan, F. (2009). *Web içerik madenciliği ve konu sınıflandırılması* (Yayımlanmamış Yüksek Lisans Tezi). Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü, Trabzon.
- , (2017, 7-10 Aralık). *Denetimli öğrenme yordamlarıyla Türkçe metinler üzerinde konu sınıflandırma analizi*. 1 st International Turkish World Engineering and Science Congress, Antalya, Türkiye.
- Hark, C., vd. (2019). Ağırlıklandırılmış çizgelerde Tf-Idf ve Eigen ayrışımı kullanarak metin sınıflandırma. *BEÜ Fen Bilimleri Dergisi* 8 (4), 1349-1362.
- Işık, M. (2019). *Müşteri yorumları üzerinde metin analitiği çalışmaları ve yorumların makine öğrenmesi algoritmaları ile modellenmesi* (Yayımlanmamış Yüksek Lisans Tezi). İstanbul Medeniyet Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Işık, N. (2019). *Metin madenciliği yöntemleri ile e-ticaret markalarına yönelik sosyal medya yorumlarının analizi* (Yayımlanmamış Yüksek Lisans Tezi). Marmara Üniversitesi Sosyal Bilimler Enstitüsü, İstanbul.
- İnan, H. E. (2019, 16-19 EKİM). *Otel Yorumlarının Sınıflandırılmasında Makine Öğrenmesi Algoritmalarının Karşılaştırılması ve Eskişehir Otellerine Yönelik Yorumların Duygu Analizi*. 20. Ulusal Turizm Kongresi, Anadolu Üniversitesi, Eskişehir.
- Jain, A., Nandakumar, K., Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern Recognition*, 2270-2285.
- Kaynar, O., vd. (2018). Makine öğrenmesi ve öznetelik seçim yöntemleriyle saldırı tespiti. *Bilişim Teknolojileri Dergisi*, 11 (2), 175-185.
- Khatun, A., vd. (2020). Comparative study on text classification. *International Journal of Engineering Science Invention (IJESI)*, 9 (9), 21-33.
- Kınacı, A. C., & Uğur, A. (2006, 21-23 Aralık). *Yapay zeka teknikleri ve yapay sinir ağları kullanılarak web sayfalarının sınıflandırılması*. 11. Türkiye’de İnternet Konferansı, TOBB Ekonomi ve Teknoloji Üniversitesi, Ankara, Türkiye.
- Kowsari, K. , vd. (2019). Text classification algorithms: a survey. *Information* 2019, 10, 150.

- Lilleberg, J., Zhang, Y., Zhu, Y. (2015, 6-8 Haziran). *Support vector machines and Word2vec for text classification with semantic features*. IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), Beijing, China.
- Munkhdalai, L., vd. (2019). Mixture of activation functions with extended min-max normalization for Forex market prediction. *IEEE Access*, 183680–183691.
- Öztürk, E. (2019). *Cyberbullying detection using text classification for Turkish language* (Yayımlanmamış Yüksek Lisans Tezi). Çukurova Üniversitesi Fen Bilimleri Enstitüsü, Adana.
- Pilavcılar, İ. F. (2007). *Metin madenciliği ile metin sınıflandırma* (Yayımlanmamış Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- PHP Nedir ?. (2021). Erişim adresi: <https://www.php.net/manual/tr/intro-what-is.php>
- The Python Tutorial. (2021). Retrieved from: <https://docs.python.org/3/tutorial/index.html>
- Scikit-learn Machine Learning in Python. (2021). Retrieved from: <https://scikit-learn.org/>
- Shixiong, X., Young, Z., Youwen, L. (2009). An Improved KNN Text Classification Algorithm Based on Clustering. *Journal of Computers*, 4 (3), 230-237.
- Sriram, B., vd. (2010, Temmuz). *Short text classification in twitter to improve information filtering*. In Proceedings of the 33rd international ACM SIGIR Conference on Research and Development in information retrieval (841-842).
- Tolon, M., & Tosunoğlu, N. (2008). Tüketici tatmini verilerinin analizi: Yapay sinir ağları ve regresyon analizi karşılaştırması. *Gazi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 10 (2), 247-259.
- TÜİK (2020, Ağustos). Hanehalkı Bilişim Teknolojileri (BT) Kullanım Araştırması, 2020. Erişim adresi: http://www.tuik.gov.tr/PreTablo.do?alt_id=1028
- Türkoğlu, F. (2006). *Melez yaklaşımlarla Türkçe dokümanlarda yazar tanıma* (Yayımlanmamış Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.
- Uzun, V., & Yıldıztepe, E. (2018). Olasılıksal yöntemler ile Türkçe metinlerin anlamsal benzerliğinin belirlenmesi. *Sinop Üniversitesi Fen Bilimleri Dergisi*, 66-78.
- Yıldırım, S., & Yıldız, T. (2018). Türkçe için karşılaştırmalı metin sınıflandırma analizi. *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, 24(5), 879-886.
- Wang, SC. (2003). *Artificial Neural Network*. In: *Interdisciplinary computing in java programming*. Boston: Springer Science+Business Media, LLC.
- Why Choose Keras?. (2021). Retrieved from: https://keras.io/why_keras

DİZİN

A

Ağırlıklandırma, 17, 20, 95

C

CERN, 24

G

Google, 8

K

Kelime Vektörü, 8, 19, 21, 94

Keras, 21, 23, 24, 73

M

Metin sınıflandırma, 4, 17, 20, 21, 73

MLP, xvi, 17, 20

MYSQL, xvi, 2, 24

N

NASA, xvi, 24

Netflix, 24

Normalizasyon, v, 30, 32, 44, 56, 64, 72

P

PHP, v, vi, xvi, 1, 23, 24, 26, 27, 97

P

PHP, v, vi, xvi, 1, 23, 24, 26, 27, 97

Python, 2, 21, 23, 29, 32, 44, 47, 49, 56, 58, 59, 64, 66, 67, 73, 97

Q

Quoc Le, 10

S

Skip-gram, 8

T

Tf-idf, v, vi, 9, 20, 21, 30

Tomas Mikolov, 10

Train_test_split, 29, 44, 72

U

Uber, 24

X

XML, 5

Y

Yapay Sinir Ağı, 17

YSA, 8

Ümmügülüm
MENGUTAYCI

**YAPAY SINIR AGLARI İLE OTEL MÜŞTERİ YORUMLARININ
METİNSEL YOLLA SINIFLANDIRILARAK DUYGU
ANALİZLERİNİN TESPİTİ**

KÜTAHYA DÜMLÜPİNAR ÜNİVERSİTESİ
Lisansüstü Eğitim Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı

