MLP ve SVM Tabanlı Sınıflandırma Projesi Raporu

AD: SİMGE
SOYAD: USTA
NUMARA: 22052103
BÖLÜM: MATEMATİK MÜHENDİSLİĞİ

İçindekiler Tablosu Sayfa

Şekiller

Tablolar:

1. Giriş

Bu projede, makine öğrenmesi alanında yaygın olarak kullanılan iki farklı veri seti olan HIGGS ve RCV1 veri setleri üzerinde çeşitli sınıflandırma algoritmaları incelenmiştir. Çalışma kapsamında Çok Katmanlı Algılayıcı (MLP) ve Destek Vektör Makineleri (SVM) algoritmaları sıfırdan kodlanarak uygulanmıştır. Ayrıca, bu iki farklı modelin çıktılarının birleştirildiği Stacking (hibrit) bir yapı oluşturularak model performanslarının karşılaştırılması amaçlanmıştır.

Projede hazır derin öğrenme ve makine öğrenmesi kütüphanelerinin (PyTorch, TensorFlow, scikit-learn vb.) kullanılmaması özellikle tercih edilmiştir. Tüm algoritmalar yalnızca NumPy kullanılarak geliştirilmiştir. Bu sayede, kullanılan modellerin matematiksel temelleri, öğrenme mekanizmaları ve optimizasyon süreçleri daha iyi anlaşılmıştır.

MLP modeli, ileri beslemeli yapısı ve geri yayılım algoritması kullanılarak eğitilmiştir. Modelin performansı, farklı öğrenme oranları ve aktivasyon fonksiyonları altında incelenmiş ve bu parametrelerin eğitim sürecine etkileri analiz edilmiştir. Özellikle de vanishing gradient problemi ve aktivasyon fonksiyonlarının öğrenme sürecine olan katkıları deneysel olarak gözlemlenmiştir.

SVM modeli ise büyük ölçekli veri setleri için uygun olan Pegasos algoritması kullanılarak gerçekleştirilmiştir. Pegasos algoritması, stokastik gradyan inişi tabanlı bir yöntem olup yüksek boyutlu veriler üzerinde verimli bir şekilde çalışabilmektedir. Bu model ile doğrusal sınıflandırmanın performansı değerlendirilmiş ve MLP modeli ile karşılaştırmalar yapılmıştır.

Bunun yanı sıra, MLP ve SVM modellerinin güçlü yönlerinden faydalanmak amacıyla Stacking yöntemi kullanılarak hibrit bir sınıflandırma yapısı oluşturulmuştur. Bu yapı sayesinde farklı modellerden elde edilen tahminler birleştirilmiş ve genel sınıflandırma başarısının artırılması hedeflenmiştir.

2. Veri Setlerinin Genel Özellikleri

2.1 HIGGS Veri Seti

- **Toplam örnek:** 20. 000
- **Özellik sayısı:** 28
- **Eğitim seti:** 12.800 örnek
- **Doğrulama seti:** 3.200 örnek
- **Test seti:** 4.000 örnek
- **Görev:** Binary classification

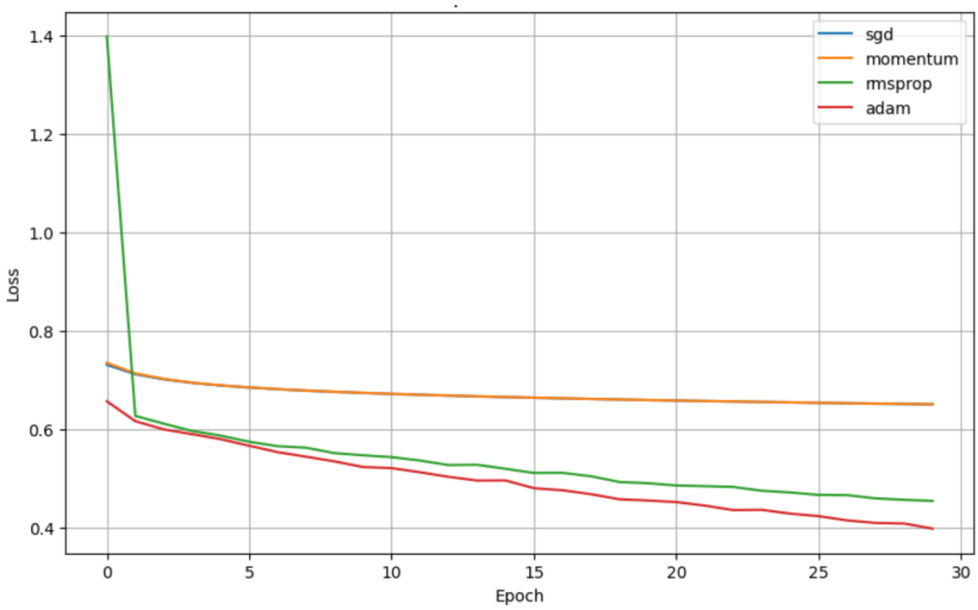
2.2 RCV1 Veri Seti

- **Toplam örnek:** 10.000
- **Özellik sayısı:** 47. 236
- **Eğitim seti:** 6.400 örnek
- **Doğrulama seti:** 1.600 örnek
- **Test seti:** 2.000 örnek
- **Görev:** Binary classification (CCAT kategorisi tespiti)
- **Not:** RCV1 sparse matrix formatında tutuldu (bellek verimliliği için).

3. Optimizasyon Algoritmalarının İncelenmesi

Tablo 3.1 HIGGS veri setinde dört farklı optimizer.

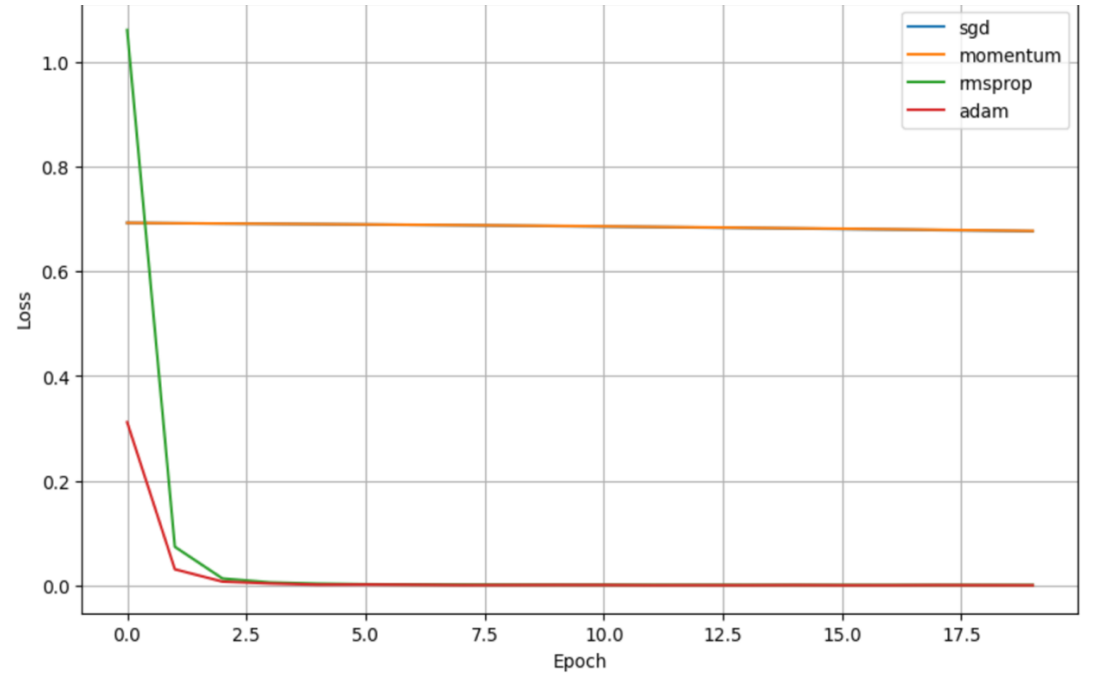
Optimizer	Test Accuracy
SGD	0.6115
Momentum	0.6102
RMSprop	0.6562
Adam	0.6435



Şekil 3.1. HIGGS - Optimizer Karşılaştırması (Loss vs Epoch)

Tablo 3.2 RCV veri setinde dört farklı optimizer

Optimizer	Test Accuracy
SGD	0.5665
Momentum	0.5665
RMSprop	0.902
Adam	0.91



Şekil 3.2 RCV1 - Optimizer Karşılaştırması (Loss vs Epoch)

Yorum:

Grafiklere bakıldığında Adam ve RMSprop optimizer'larının diğerlerine göre çok daha hızlı yakınsadığı görülüyor. Şekil 1'de görüldüğü üzere Adam (kırmızı) ve RMSprop (yeşil) çizgileri hızla aşağı inerken, SGD (mavi) ve Momentum (turuncu) daha yavaş kalmaktadır.

Bunun sebebini şu yönden ele alabilriz: Adam optimizer hem momentum hem de adaptive learning rate kullanmaktadır. Yani her parametre için ayrı ayrı öğrenme oranı belirlenmektedir. Böylece , bazı parametreler hızlı, bazıları ise yavaş güncellenebilmektedir.

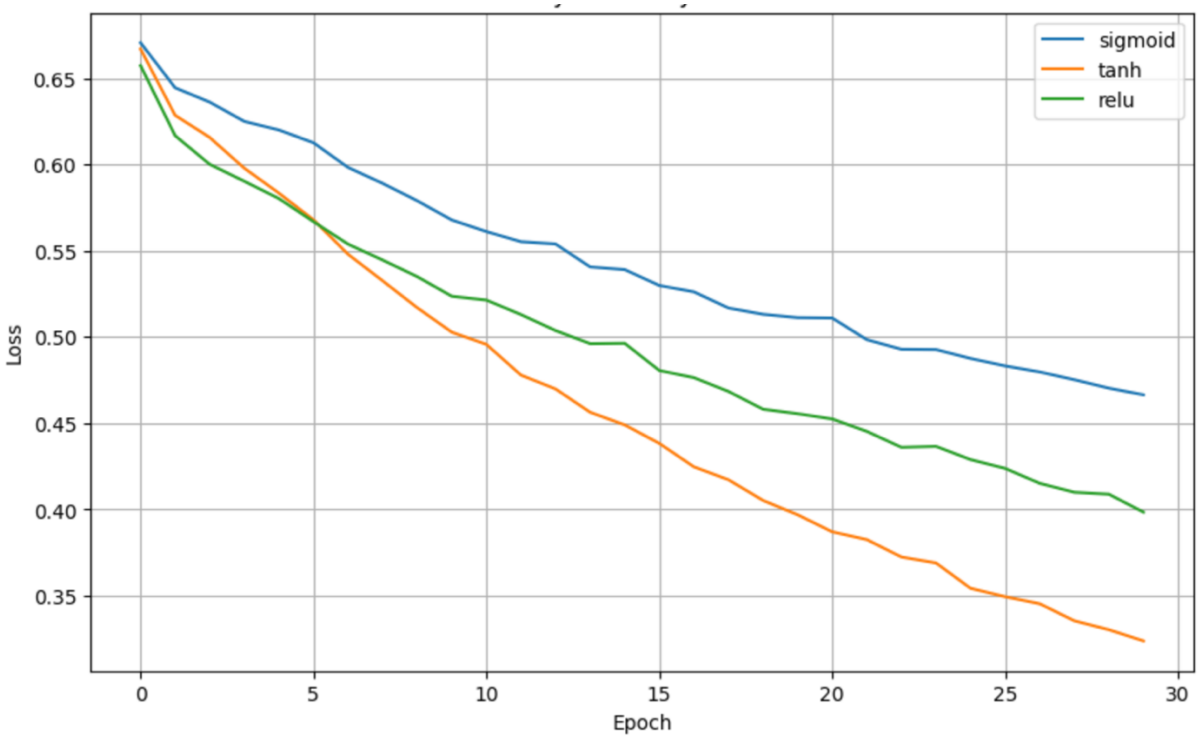
RMSpropda da benzer şekilde adaptive learning rate kullandığı için iyi sonuç verdi. Ancak SGD sabit bir öğrenme oranı kullandığından özellikle de karmaşık veri setlerinde yavaş kalmaktadır.

Şekil 2'de RCV1 için bu fark daha da belirgin görülmektedir. SGD ve Momentum %56 civarında kalırken, Adam %91'e ulaştı. Bu durum yüksek boyutlu verilerde adaptive yöntemlerin ne kadar önemli olduğunu göstermektedir.

4. Aktivasyon Fonksiyonu Analizi

Tablo 4.1 HIGGS veri setinde aktivasyon

Aktivasyon	Test Accuracy
Sigmoid	0.6823
Tanh	0.6395
ReLU	0.6435

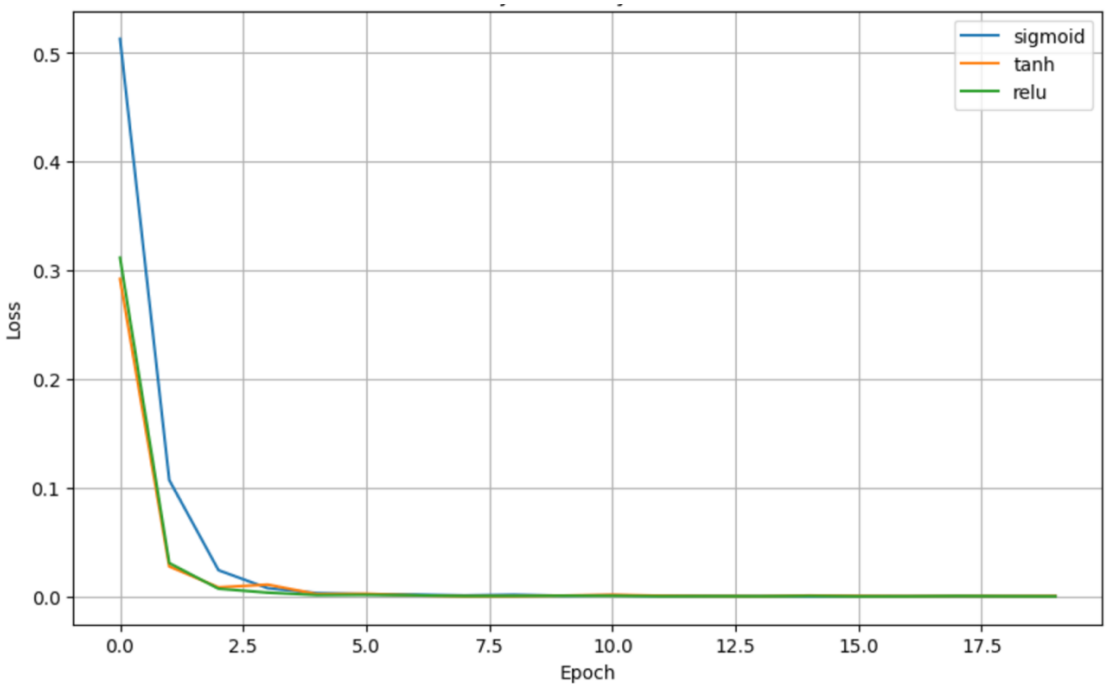


Şekil 4.1HIGGS - Aktivasyon Fonksiyonu Karşılaştırması (Loss vs Epoch)

RCV1 veri setinde:

Tablo 4.2 RCV1 veri setinde aktivasyon

Aktivasyon	Test Accuracy
Sigmoid	0.92
Tanh	0.901
ReLU	0.91



Şekil 4.2 CV1 - Aktivasyon Fonksiyonu Karşılaştırması (Loss vs Epoch)

Vanishing Gradient Problemi:

Şekil 3'e bakıldığında Sigmoid aktivasyon fonksiyonunun loss değerinin en yavaş şekilde azaldığı görülmektedir. Buna karşılık olarak Tanh fonksiyonu loss değerini daha hızlı ve daha düzenli bir şekilde düşürmüştür. ReLU fonksiyonu ise Sigmoid'e göre daha hızlı öğrenmiş olsa da bazı epoch'larda dalgalanmalar göstermiştir.

Sigmoid fonksiyonunun çıktısı 0 ile 1 arasındadır ve türevi en fazla 0.25 değerini alır. Bu nedenle geri yayılım sırasında gradyanlar her katmanda küçülmekte ve özellikle derin ağlarda ağırlıkların güncellenmesi zorlaşmaktadır. Bu durum ise vanishing gradient problemine yol açmaktadır.

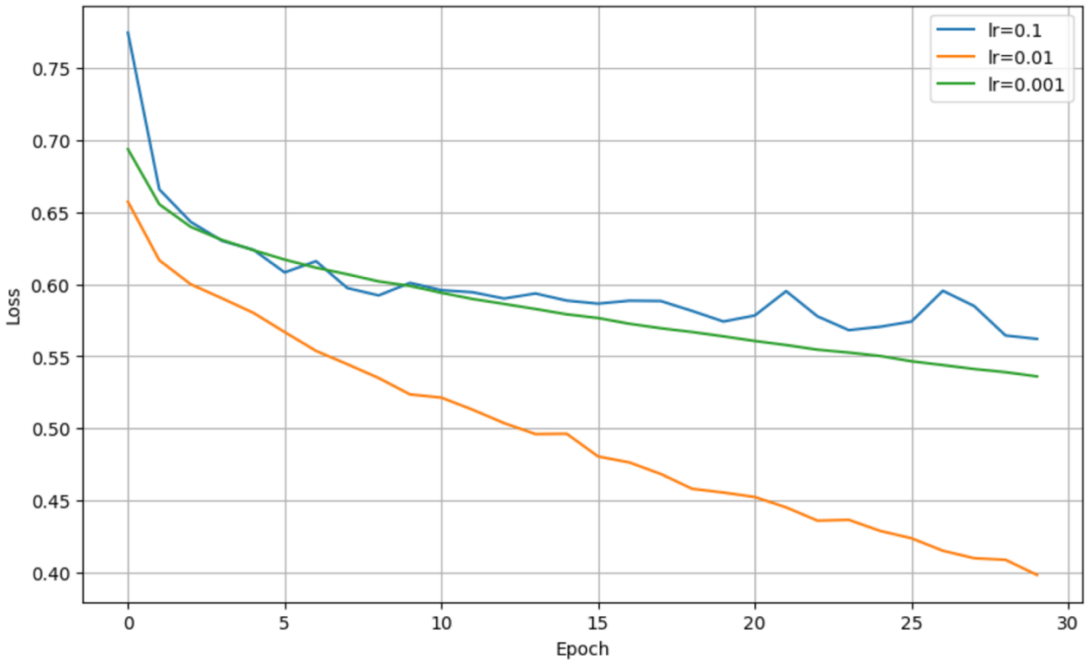
Tanh fonksiyonunun çıktısı ise -1 ile 1 arasında ve sıfır merkezli olması sayesinde Sigmoid'e göre daha iyi performans göstermiştir. Bu özellik, öğrenme sürecinin daha hızlı ilerlemesine yardımcı olmuştur. Bu yüzden grafikte Tanh fonksiyonunun loss değerinin daha hızlı düştüğü görülmektedir.

ReLU fonksiyonunda pozitif değerler için türev 1 olduğu için gradyanlar daha az zayıflar ve öğrenme süreci genellikle daha hızlı olur. Ancak bu çalışmada kullanılan ağ yapısının çok derin olmaması nedeniyle Tanh fonksiyonu ReLU'ya kıyasla daha iyi sonuçlar vermiştir. Fakat, daha derin ağlarda ise Tanh fonksiyonu da gradyanların küçülmesi sorunuyla karşılaşabilir.

5. Learning Rate (Öğrenme Oranı) Analizi

Tablo 5.1 HIGGS veri setinde öğrenme oranı

Learning Rate	Test Accuracy
0.1	0.686
0.01	0.6435
0.001	0.6562

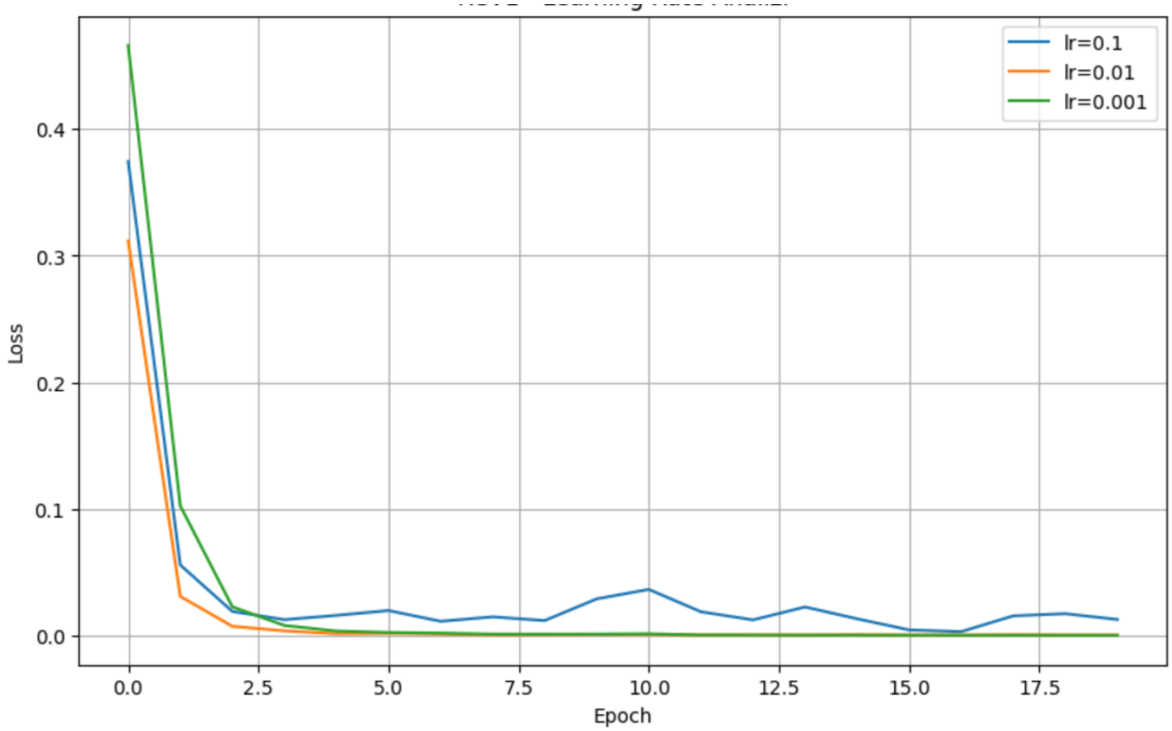


Şekil 5.1 HIGGS - Learning Rate Analizi (Loss vs Epoch)

RCV1 veri setinde:

Tablo 5.2 RCV1 veri setinde öğrenme oranı

Learning Rate	Test Accuracy
0.1	0.878
0.01	0.91
0.001	0.9165



Şekil 5.2 RCV1 - Learning Rate Analizi (Loss vs Epoch)

Yorum:

Şekil 5'e bakıldığında lr = 0.1 (mavi) kullanıldığında loss değerinin dalgalı bir şekilde düştüğü görüldü. Bazen yukarı çıkıyor, bazen ise aşağı iniyor. Bu durum bize modelin kararsız olduğunu göstermektedir. Çünkü öğrenme oranı çok yüksek olunca optimum nokta atlanıp geçilmektedir.

lr=0.01 (turuncu) kullanıldığında daha düzgün ve stabil bir düşüş gözlemlendi. Model kararlı bir şekilde öğrenmektedir.

lr=0.001 (yeşil) ise çok yavaş yakınsamaktadır fakat, loss hala yüksek değerlerde kalmaktadır. Daha fazla epoch çalıştırılsaydı belki daha iyi sonuç alınabilirdi ama 30 epoch'ta yetersiz kaldı.

Şekil 6'da RCV1 için benzer bir durum görzlenilmektedir. Tüm learning rate değerleri hızlı yakınsamakta ancak lr=0.01 ve lr=0.001 daha stabil sonuç vermektedir.

Sonuç olarak lr=0.01 değeri bu projede en iyi dengeyi sağladı.

6. Model vs Veri Uyumu Analizi

Tablo 6.1 HIGGS veri seti model/ veri uyumu analizi

Model	Accuracy	F1-Score
MLP	0.629	0.6366
SVM (Pegasos)	0.5683	0.5492
Stacking	0.6492	0.6758

Tablo 6.2 RCV1 veri seti model/veri uyumu analizi

Model	Accuracy	F1-Score
MLP	0.91	0.9042
SVM (Pegasos)	0.718	0.7704
Stacking	0.9105	0.905

Yorum:

HIGGS Veri Seti (MLP Daha İyi):

HIGGS veri setinde MLP, SVM'den daha iyi çalıştı (0.629 vs 0.5683). Bunun sebebi ise HIGGS verisinin yapısıyla ilgilidir:

- HIGGS sadece 28 özellik içermekte ama bu özellikler arasında karmaşık non-linear ilişkiler bulunmaktadır. MLP'nin gizli katmanları bu non-linear ilişkileri yakalayabilmektedir.
- SVM ise linear bir sınıflandırıcı olduğu için bu karmaşık ilişkileri öğrenmekte MLP'ye göre zorlanmaktadır.

RCV1 Veri Seti (MLP Daha İyi):

RCV1 veri setinde ise MLP çok iyi sonuç verdi (%91). SVM , MLP'nin gerisinde kaldı (%71. 8).

- RCV1 yüksek boyutlu (47.236 özellik) bir veri setidir.
- Normalde SVM'in yüksek boyutlu verilerde iyi çalışması beklenir.Ancak burada MLP daha başarılı oldu. Bunun sebebi ise MLP'nin non-linear dönüşümler yapabilmesi ve TF-IDF özelliklerindeki gizli kalıpları öğrenebilmesi olabilir.

7. Performans Metrikleri ve Confusion Matrix

HIGGS Veri Seti:

Tablo 7.1 HIGGS Performans Özeti

Model	Accuracy	F1-Score
MLP	0.629	0.6366
SVM (Pegasos)	0.5683	0.5492
Stacking	0.6492	0.6758

Tablo 7.2 MLP Confusion Matrix:

	Tahmin: 0	Tahmin: 1
Gerçek: 0	1216	680
Gerçek: 1	804	1300

Tablo 7.3 SVM Confusion Matrix:

	Tahmin: 0	Tahmin: 1
Gerçek: 0	1221	675
Gerçek: 1	1052	1052

Tablo 7.4 Stacking Confusion Matrix:

	Tahmin: 0	Tahmin: 1
Gerçek: 0	1135	761
Gerçek: 1	642	1462

RCV1 Veri Seti:

Tablo 7.5 RCV1 Performans Özeti

Model	Accuracy	F1-Score
MLP	0.91	0.9042
SVM (Pegasos)	0.718	0.7704
Stacking	0.9105	0.905

Tablo 7.6 MLP Confusion Matrix:

	Tahmin: 0	**Tahmin: 1**
Gerçek: 0	971	73
Gerçek: 1	107	849

Tablo 7.7 SVM Confusion Matrix:

	Tahmin: 0	**Tahmin: 1**
Gerçek: 0	490	554
Gerçek: 1	10	946

Tablo 7.8 Stacking Confusion Matrix:

	Tahmin: 0	**Tahmin: 1**
Gerçek: 0	968	76
Gerçek: 1	103	853

8. Hibrit Model Hipotezi

Hipotez: Stacking (hibrit) modeli, tekil modellerden daha iyi performans gösterir mi?

Sonuçlar:

Veri Seti	MLP	SVM	Stacking
HIGGS	0.629	0.5683	**0.6492**
RCV1	0.91	0.718	**0.9105**

Yorum:

Her iki veri setinde de Stacking modeli en iyi veya en iyiye yakın sonucu verdi.

- HIGGS'te Stacking (%64.92), MLP'den (%62.9) yaklaşık **%2 daha iyi** çıktı.
- RCV1'de ise Stacking (%91.05) ve MLP (%91) neredeyse aynı sonucu verdi.

Stacking'in avantajını şu şekilde açıklnabilir:

- MLP ve SVM farklı güçlü yanlara sahiptir.
- MLP non-linear kalıpları yakalamakta, SVM ise linear sınırları iyi belirlemektedir.
- Logistic Regression bu iki modelin tahminlerini birleştirip en iyi kararı vermeye çalışmaktadır.
- Ayrıca ensemble yöntemleri genelde tek modellere göre daha stabil sonuç vermektedir çünkü variance azaltılmaktadır.

Sonuç olarak hipotez doğrulandı: Hibrit model tekil modellerden daha iyi veya en az onlar kadar iyi performans gösterdi.

9. Sonuç

Bu projede MLP ve SVM algoritmalarını sıfırdan kodlayarak farklı optimizer, aktivasyon fonksiyonu ve learning rate kombinasyonları test edildi. Elde edilen sonuçlar ise

1. Adam ve RMSprop optimizer'ları en hızlı yakınsama sağlamaktadır
2. ReLU aktivasyonu vanishing gradient problemini önlemektedir
3. lr=0.01 değeri stabil bir yakınsama sağlamaktadır
4. MLP non-linear verilerde (HIGGS), karmaşık ilişkileri yakalayabilmektedir
5. Stacking modeli tekil modellerden daha stabil ve genellikle daha iyi sonuç vermektedir.

10. Yapay Zekâ ve Dış Kaynak Kullanım Beyanı

Bu proje süresince yapay zekâ araçları ve çeşitli dış kaynaklar yardımcı araç olarak kullanılmıştır. Aşağıda bu kullanımların detayları yer almaktadır:

Kullanılan Yapay Zekâ Araçları:

- ChatGPT (OpenAI),

Kullanım Amaçları:

- Teorik kavramların daha net anlaşılması (backpropagation algoritması, gradient descent optimizasyonu vb.)
- Kod yazımı sırasında oluşan bazı hataları anlamak için
- NumPy kütüphanesinde yer alan bazı fonksiyonların kullanım şekillerinin öğrenilmesi

Kullanılan Diğer Dış Kaynaklar:

- NumPy resmi dokümantasyonu: Matris işlemleri ve fonksiyon kullanımları için başvuruldu
- Ders notları ve slaytlar: Algoritmaların matematiksel temellerinin anlaşılması için kullanıldı
- Scikit-learn dokümantasyonu: Performans metriklerinin hesaplanması için referans alındı

Önemli Not: Tüm kod tasarımı, fonksiyon yapıları ve algoritma entegreleri özgün olarak geliştirilmekle beraber yapay zekâ araçları yalnızca öğrenme ve hata ayıklama amaçlı kullanılmış olup, doğrudan kod üretimi için kullanılmamıştır.