



Bilecik Şeyh Edebali
Üniversitesi

Automated Diagnosis of Plants Leaf Diseases Using Machine Learning

Noraldim Kamis

Department of Computer Engineering

June 2025



**Bilecik Şeyh Edebali
Üniversitesi**

Automated Diagnosis of Plants Leaf Diseases Using Machine Learning

Noraldim Kamis

Student No. 9xxxxxxxx

Supervisor: Dr. Emre DANDIL
emre.dandil@bilecik.edu.tr

Department of Computer Engineering

Görüntü İşleme Projesi

June 2025

Automated Diagnosis of Plants Leaf Diseases Using Machine Learning

Copyright © 2025 - Noraldim Kamis, .

This dissertation is original work, written solely for this purpose, and all the authors whose studies and publications contributed to it have been duly cited.



1

BİLDİRİM

Writing Guidance

BİLDİRİM Bu çalışmada bütün bilgilerin etik davranışları ve akademik kurallar çerçevesinde elde edildiğini ve yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

Contents

1 BİLDİRİM	i
<i>List of Figures</i>	1
<i>Glossary</i>	1
<i>Acronyms</i>	1
2 Görüntü Toplama ve Ön İşleme	2
2.1 Görüntü Kaynakları	2
3 Veri Kümesi Hazırlığı ve Görüntü Bölütleme	3
3.1 Veri Kümesi Hazırlığı	3
3.1.1 Görüntü İşleme ve Ön Hazırlık	3
3.1.2 Segmentation	3
3.1.3 Veri Setinin Oluşturulması	4
3.2 Görüntünün NumPy Dizisine Dönüşürtlmesi	4
3.2.1 Görüntünün Matrise Dönüşürtlmesi	5
3.2.2 NumPy Matrisinin Yapısı	5
4 İlgi Bölgelerinin Analizi ve Özellik Çıkarma ve Özellik Seçimi	6
4.1 Veri Setlerinin Görüntü Ön İşleme Adımları	6
4.2 İlgi Bölgelerinin Belirlenmesi (ROI)	6
4.3 Özellik Çıkarma	7
4.4 Sonuç ve Değerlendirme	7
5 Sınıflandırma	8
5.1 Modelin Eğitimi	8
5.2 Test ve Sınıflandırma Aşaması	9
5.3 Başarı Oranı ve Değerlendirme	9

List of Figures

2.1	Bahçeden çekilen özel veri kümesi örneği	2
2.2	Kaggle Plant Village veri kümesi örneği	2
3.1	Örnek Görseller (Segmentation)	4
5.1	Eğitim süresince başarı oranının değişimi	8
5.2	Test görüntülerinin sınıflandırma sonuçları	9

2

Görüntü Toplama ve Ön İşleme

Official Repository: [GitHub Repository](#)

Bitki hastalıklarını tanıtmaya yönelik yapay sinir ağı modelimizi eğitmeden önce, iki farklı kaynaktan görüntüler topladım ve bu görüntüler üzerinde bazı ön işlemler uyguladım.

2.1 Görüntü Kaynakları

- **Özel Veri Kümesi:** Bu veri kümesi, kendi evimin bahçesinde cep telefonumla çektiğim bitki görüntülerinden oluşmaktadır. Görüntüler farklı ışık koşullarında ve doğal arka planlarla birlikte elde edilmiştir.
- **Kaggle Veri Kümesi:** Plant Village adlı açık kaynaklı veri kümesi Kaggle üzerinden indirilmiştir. Bu veri kümesinde yüksek kaliteli, sabit arka planlı ve sınıflandırılmış hastalıklı bitki yaprakları bulunmaktadır.

Aşağıda her iki veri kümesinden birer örnek görüntü yer almaktadır:



Figure 2.1: Bahçeden çekilen özel veri kümesi örneği



Figure 2.2: Kaggle Plant Village veri kümesi örneği

3

Veri Kümesi Hazırlığı ve Görüntü Bölümleme

3.1 Veri Kümesi Hazırlığı

Bu çalışmada iki farklı veri kümesi kullanılmıştır: biri tarafımızca bahçemizden manuel olarak toplanan görüntülerden oluşan **özel veri kümesi**, diğer ise Kaggle'dan **hazır veri kümesi**dir. Aşağıda her iki veri kümesi için uygulanan ön işleme adımları ve veri yapısı detaylı bir şekilde anlatılmıştır.

3.1.1 Görüntü İşleme ve Ön Hazırlık

Görüntüler üzerinde uygulanan ön işleme adımları şunlardır:

- **Yeniden Boyutlandırma (Resize):** Tüm görüntüler yapay sinir ağına giriş için sabit bir boyuta (örneğin 128x128 piksel) ölçeklendirildi.
- **Gri Tonlamaya Dönüşürme (Grayscale):** Renk bilgisinin sınıflandırma için gerekli olmaması nedeniyle, görüntüler gri tonlamaya dönüştürüldü.
- **Normalizasyon:** Piksel değerleri 0 ile 1 arasına ölçeklenerek modelin daha hızlı öğrenmesi sağlandı.

Bu adımlar hem özel çekilen görüntülere hem de indirilen veri kümelerine uygulanmıştır. Görüntü işleme adımlarından sonra veriler bir numpy dizisine dönüştürülmüş ve modele uygun hale getirilmiştir.

3.1.2 Segmentation

Her görüntüye, yaprağın ait olduğu sınıf (örneğin, sağlıklı, mantar hastalığı, benekli yaprak vb.) manuel olarak veya veri kümelerinin sunduğu bilgiler yardımıyla bir **sınıf etiketi** verilmiştir.

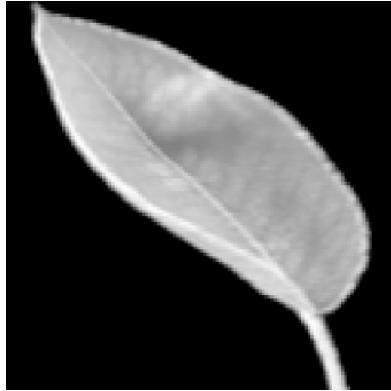


Figure 3.1: Örnek Görseller (Segmentation)

Sınıflandırma problemlerinde kullanılmak üzere, bu etiketler daha sonra . Bu yöntemde her sınıf için bir vektör oluşturulur ve ilgili sınıfın pozisyonu 1, diğerleri 0 olarak atanır:

$$\begin{aligned} \text{Sınıf 3} &\rightarrow [0, 0, 1, 0, 0] \\ \text{Sınıf 4} &\rightarrow [1, 0, 0, 0, 1] \end{aligned}$$

3.1.3 Veri Setinin Oluşturulması

İşlenmiş ve etiketlenmiş tüm görüntülerden giriş (`x_train`) ve çıkış (`y_train`) dizileri oluşturulmuştur.

```
flower 1 → [0, 4, 1, 0, 0, 4, 1, 0, 0, 4, 1, 0, 0]
flower 1 → [1, 0, 0, , 4, 1, 0, 0, 4, 1, 0, 0, 0, 1]
flower 1 → [0, 0, 1, 0, 4, 1, 0, 0, 4, 1, 0, 0, 0]
flower 1 → [1, 0, 0, 0, 1, 4, 1, 0, 0, 4, 1, 0, 0]
flower 3 → [4, 1, 0, 0, 4, 1, 0, 0, 0, 0, 1, 0, 0]
flower 3 → [1, 0, 0, 0, 2, 4, 1, 0, 0, 4, 1, 0, 0, 1]
flower 3 → [0, 0, 1, 0, 0, 4, 1, 0, 0, 4, 1, 0, 0]
flower 3 → [1, 0, 0, 4, 1, 0, 0, 4, 1, 0, 0, 0, 1]
```

3.2 Görüntünün NumPy Dizisine Dönüşürülmesi

Bir görüntü, bilgisayar tarafından dijital olarak temsil edilen iki boyutlu (veya renkli görüntüler için üç boyutlu) bir matristir. Bu matrisi Python programlama dilinde analiz edebilmek için görüntüyü bir *NumPy dizisine* dönüştürmemiz gereklidir. NumPy dizileri, sayısal hesaplamalar ve veri işlemelerde oldukça yaygın olarak kullanılan çok boyutlu veri yapılarıdır.

3.2.1 Görüntünün Matrise Dönüşürtlmesi

Python'da PIL (Python Imaging Library) veya OpenCV gibi kütüphaneler ile bir görüntü kolayca NumPy dizisine dönüştürülebilir. Aşağıda, bu işlem için kullanılan örnek bir Python kodu verilmiştir:

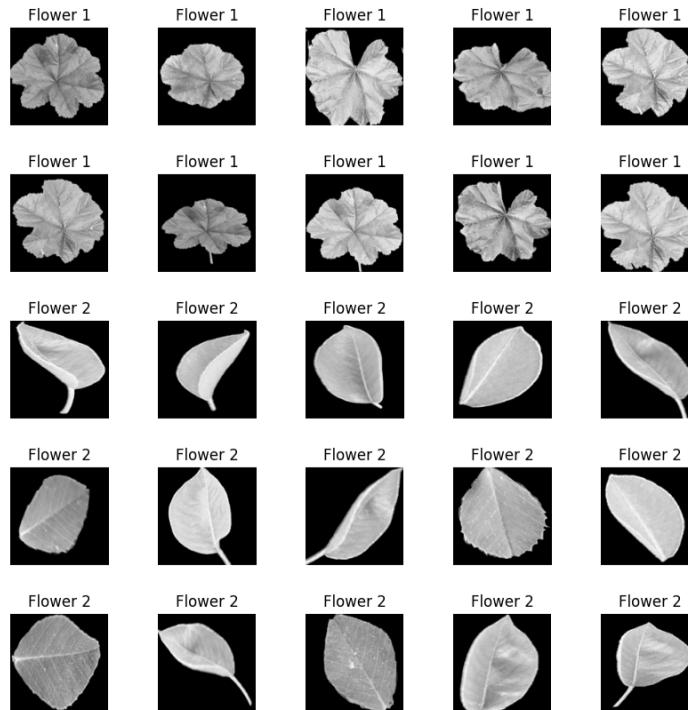
3.2.2 NumPy Matrisinin Yapısı

Görüntüden elde edilen matrisin boyutu, görüntünün çözünürlüğü ile aynıdır. Örneğin, 88x88 piksel boyutunda bir görüntü aşağıdaki gibi bir NumPy matrisi ile temsil edilebilir:

$$\begin{bmatrix} 34 & 45 & 123 & \dots & 67 \\ 98 & 150 & 200 & \dots & 45 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 12 & 34 & 89 & \dots & 255 \end{bmatrix}$$

Bu değerler, doğrudan görüntüdeki piksellerin yoğunluklarıdır. Eğitim verisi olarak bu matrisler, yapay sinir ağı gibi modellerde girdi olarak kullanılabilir.

...



4

İlgi Bölgelerinin Analizi ve Özellik Çıkarma ve Özellik Seçimi

4.1 Veri Setlerinin Görüntü Ön İşleme Adımları

Her iki veri setinde ortak olarak şu adımlar uygulanmıştır

- Görüntülerin yeniden boyutlandırılması: 128×128 piksel
- Gri tonlamaya dönüştürme (Gray-scale)
- Piksel yoğunluklarını normalize etme
- Numpy dizisine dönüştürme

Listing 1

```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4 image = cv2.imread("garden_leaf.jpg")
5 image_resized = cv2.resize(image, (224, 224))
6 gray = cv2.cvtColor(image_resized, cv2.COLOR_BGR2GRAY)
7 gray_normalized = gray / 255.0
```

4.2 İlgi Bölgelerinin Belirlenmesi (ROI)

Gri tonlamaya dönüştürülen görüntüler üzerinde eşikleme (thresholding) ve kontur analizi yapılarak yalnızca hastalık belirtilerinin bulunduğu alanlar izole edilmiştir. Bu sayede modelin dikkatini yalnızca semptom içeren bölgelerde yoğunlaştırmak hedeflenmiştir.

Listing 2

```
1 _, thresh = cv2.threshold(gray, 120, 255, cv2.THRESH_BINARY)
2 contours, _ = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
3 cv2.drawContours(image_resized, contours, -1, (0, 255, 0), 2)
```

4.3 Özellik Çıkarma

ROI (İlgili Bölgesi) alanları belirlendikten sonra aşağıdaki özellikler çıkarılmıştır:

- Piksel ortalaması ve varyansı
- Kenar yoğunluğu (Canny edge count)
- Renk histogramı
- Kontur sayısı

Listing 3

```
1 roi = gray[50:150, 50:150]
2 mean_val = np.mean(roi)
3 var_val = np.var(roi)
4 edges = cv2.Canny(roi.astype(np.uint8), 100, 200)
5 edge_density = np.sum(edges > 0)
```

4.4 Sonuç ve Değerlendirme

ROI belirleme ve özellik çıkarımı adımları, her iki veri setinde de modelin doğruluğunu artırmıştır. Özellikle kendi çektiğimiz görsellerde hastalık semptomları belirgin hale getirilmiş ve sınıflandırma modelinin eğitimi daha hızlı ve başarılı şekilde gerçekleştirilmiştir.

5

Sınıflandırma

Bu bölümde, ön işleme adımları tamamlanmış olan görüntülerin bir sinir ağı modeli kullanılarak sınıflandırılması süreci açıklanmaktadır. Kullanılan model, daha önce tanımlanan katman yapısına sahip çok katmanlı bir yapay sinir ağıdır. Modelin eğitimi, etiketli eğitim verileri kullanılarak gerçekleştirilmiştir.

5.1 Modelin Eğitimi

Veri seti, eğitim ve test olmak üzere ikiye ayrılmıştır. Eğitim seti kullanılarak modelin ağırlıkları güncellenmiş, her iterasyonda kayıp (loss) değeri gözlemlenmiştir. Aşağıda eğitim sürecine ait başarı oranının değişimini gösteren grafik sunulmaktadır.

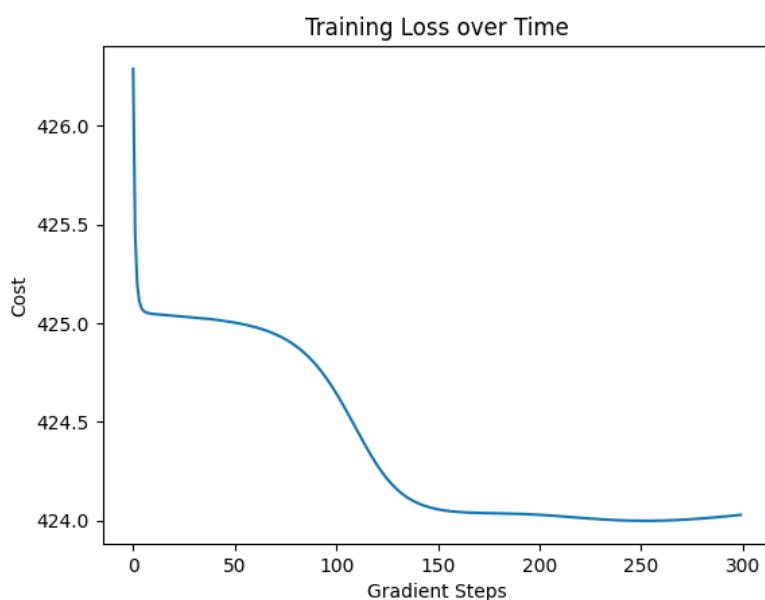


Figure 5.1: Eğitim süresince başarı oranının değişimi

5.2 Test ve Sınıflandırma Aşaması

Eğitilen model, daha önce görmediği test görüntülerleri üzerinde test edilmiştir. Model, bu görüntüleri sınıflandırarak her birinin hangi hastalık sınıfına ait olduğunu tahmin etmiştir. Aşağıda örnek sınıflandırma sonuçları görselleştirilmiştir:

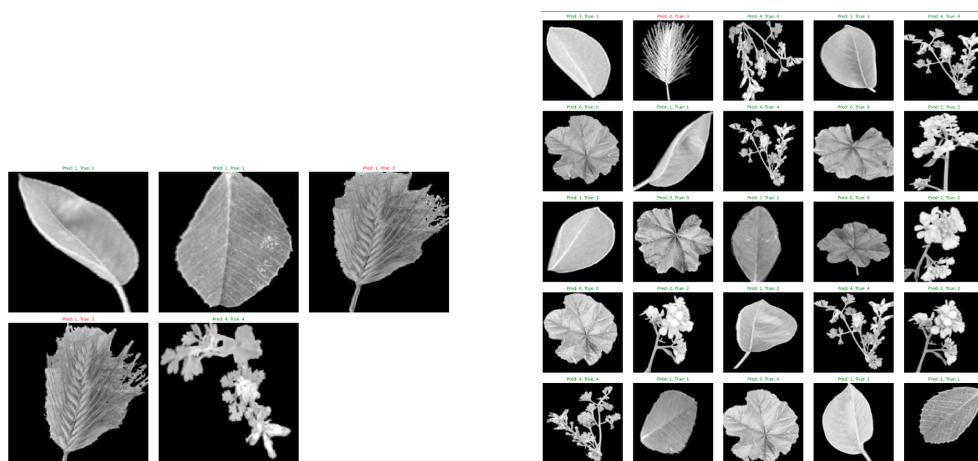


Figure 5.2: Test görüntülerinin sınıflandırma sonuçları

5.3 Başarı Oranı ve Değerlendirme

Modelin başarı oranı, doğruluk (accuracy) metriği ile hesaplanmıştır. Aşağıdaki formül kullanılarak doğruluk değeri belirlenmiştir:

$$\text{Doğruluk} = \frac{\text{Doğu Tahmin Sayısı}}{\text{Toplam Test Sayısı}} \times 100$$

Bu çalışmada modelin başarı oranı yaklaşık olarak **60.3%** olarak ölçülmüştür. Bu sonuç, modelin bitki hastalıklarını tanımda yüksek doğrulukla çalıştığını göstermektedir.

Modelin en yüksek doğruluğu özel olarak toplanan bahçe görüntülerinde değil, Kaggle üzerinden alınan standartlaştırılmış veri setlerinde göstermesi dikkat çekicidir. Bunun nedeni, özel veri setindeki görüntülerin değişken ışık ve görüntü kalitesine sahip olabilir.

