T.C.
BURSA TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
VERİ MADENCİLİĞİ DERSİ PROJESİ

Karar Ağacı Tabanlı Meme Kanseri Sınıflandırması

Evin AYDİN ÜLGEN

2022-2023 BAHAR DÖNEMİ

ÖNSÖZ

Meme kanseri, dünya genelinde kadınlarda en sık görülen kanser türlerinden biridir ve sağlık açısından önemli bir sorundur. Erken teşhis, meme kanserinin tedavi sürecindeki başarıyı artırma ve yaşam kalitesini iyileştirme açısından kritik bir öneme sahiptir. Bu nedenle, doğru ve güvenilir bir teşhis yöntemi geliştirmek, hem hastaların hayatını kurtarma potansiyeline sahip hem de sağlık profesyonellerine önemli bir araç sunma anlamına gelmektedir.

Bu projenin amacı, meme kanseri teşhisinde doğruluk oranını artırmak ve yanlış teşhis oranını azaltmak için makine öğrenimi algoritmalarını kullanmaktır. Bu amaç doğrultusunda, bir meme kanseri veri seti kullanılarak, bir karar ağacı (decision tree) sınıflandırma modeli oluşturulmuştur.

Veri seti, birçok hasta kaydını içermektedir ve her bir hasta için çeşitli özniteliklerden oluşmaktadır. Bu öznitelikler arasında yaş, tümör boyutu, lenf nodu durumu, histolojik tip ve malignite derecesi gibi bilgiler yer almaktadır.. Veri setinin ön işlemesi yapılarak eksik değerlerin doldurulması ve kategorik verilerin dönüştürülmesi gibi adımlar gerçekleştirilmiştir. Ardından, karar ağacı algoritması kullanılarak model eğitimi gerçekleştirilmiş ve elde edilen model, test veri setinde değerlendirilmiştir.

Bu proje, meme kanseri teşhisinde doğruluk oranını artırmayı hedefleyen bir çalışmadır. Yüksek hassasiyet ve özgüllük değerleri elde etmek için modelin performansı ayrıntılı olarak değerlendirilmiştir. Ayrıca, projenin sınırlamaları ve elde edilen sonuçların yorumlanması da raporda ele alınmıştır.

.

Saygılarımla,

Evin AYDİN ÜLGEN

Bursa, 2023

İÇİNDEKİLER

1. GİRİŞ

1.1. Projenin Amacı ve Kapsamı

Bu proje, meme kanseri teşhislerinde yardımcı olabilecek bir sınıflandırma modeli oluşturmayı amaçlamaktadır. Meme kanseri, kadınlarda en sık görülen kanser türlerinden biridir ve erken teşhisin tedavi süreci hastanın sağkalımı üzerinde önemli bir etkisi vardır. Bu nedenle, doğru bir şekilde meme kanseri teşhisini yapabilen bir sınıflandırma modeli, tıbbi alanda büyük bir öneme sahiptir.

Bu projenin kapsamı, UCI Machine Learning Repository' den elde edilen "Breast Cancer" veri seti üzerinde sınıflandırma yapmaktır. Veri seti, meme kanseri tanısında kullanılan klinik özelliklerin (örneğin, tümör boyutu, şekli, hücre büyüklüğü vb.) yanı sıra sonucun (meme kanseri veya sağlıklı) etiketlenmiş verilerini içermektedir. Veri seti, Wisconsin Üniversitesi Hastanesi' ndeki meme kanseri teşhisleriyle ilgili bilgileri içermektedir.

Bu projenin hedefleri şunlardır:

- Meme kanseri teşhislerini doğru bir şekilde yapabilen bir sınıflandırma modeli oluşturmak.
- Veri setindeki önemli klinik özellikleri belirleyebilmek.
- Modelin performansını değerlendirmek ve sınıflandırma sonuçlarını yorumlamak.
- Önceden yapılmış akademik çalışmalarla elde edilen sonuçları karşılaştırarak projenin değerlendirmesini yapmak.

Bu proje, meme kanseri teşhisinde kullanılan bir sınıflandırma modeli oluşturarak, tıbbi alanda erken teşhisin sağlanmasına katkıda bulunmayı hedeflemektedir.

1.2. Kullanılan Veri Setinin Tanımı ve Kaynağı

Bu projede kullanılan veri seti, meme kanseri teşhisinde kullanılan klinik özellikleri ve sonucun etiketlenmiş verilerini içeren "Breast Cancer" veri setidir. Veri setinin kaynağı, UCI Makine Öğrenimi Deposu'dur. Bu depo, makine öğrenimi ve veri madenciliği araştırmaları için birçok veri seti sunmaktadır.

Bu veri seti, meme kanseri teşhisiyle ilgili klinik özellikler ve sonucun etiketlenmiş verilerini içerdiği için meme kanserinin teşhisinde önemli bir bilgi kaynağıdır.

1.3. Sınıflandırma Yöntemi Seçimi ve Sebepleri

Bu projede sınıflandırma yöntemi olarak Decision Tree (Karar Ağacı) yöntemini seçtim. Karar Ağacı, sınıflandırma problemlerinde en sık kullanılan yöntemlerden biridir. Veri setindeki özellikleri inceleyerek ağaç yapısında karar kolları ve yapraklar oluşturur. Veri setindeki özellikleri sınıflandırmak için bu ağaç yapısı kullanılır.

Karar ağacı yöntemi, verilerin yapısal bir şekilde temsil edilmesine ve modelin anlaşılmasına yardımcı olur. Ayrıca, diğer yöntemlerle karşılaştırıldığında kolayca anlaşılabilir bir sonuç verir. Veri setindeki özelliklerin birbirine bağlı olduğu durumlarda, karar ağacı yöntemi doğru sonuçlar verir. Ayrıca, karar ağacı yöntemi, veri setindeki özellikler arasındaki etkileşimleri keşfetmek için de kullanılabilir.

Breast Cancer veri seti, kanser teşhisi yapmak için kullanılan bir veri setidir. Sınıflandırma probleminde, kanserli veya kanser olmayan durumlar gibi iki sınıf vardır. Karar ağacı yöntemi, kanser teşhisi yapmak için bu veri setindeki özelliklerin birbirine bağlı olduğu durumlarda kullanılabilir. Veri setindeki özellikler arasında belirli bir ilişki olması, karar ağacı yöntemi ile keşfedilebilir ve bu özellikler kanser teşhisinde önemli bir rol oynayabilir.

Bu nedenlerden dolayı, Breast Cancer veri setinde karar ağacı yöntemini kullanmayı seçtim.

2. VERİ KEŞFİ

2.1. Veri Setinin Yapısal Analizi

Veri setinin yapısal analizi, veri setinin genel yapısını anlamamıza yardımcı olan bir adımdır. Bu analiz, veri setinin boyutları, örnek sayısı, özellik sayısı ve eksik veri durumu gibi faktörleri içerir. İşte "Breast Cancer" veri setinin yapısal analizi:

- **Örnek Sayısı:** "Breast Cancer" veri setinde 286 örnek bulunmaktadır. Bu, veri setindeki toplam hasta sayısını temsil eder. Her bir örnek, bir meme kanseri teşhisi olan bir hasta durumunu temsil eder.
- **Özellik Sayısı**: "Breast Cancer" veri setinde 10 adet özellik bulunmaktadır. Bu özellikler, hastaların yaş aralığı, menopoza girme durumu, tümör boyutu, invaziv nod sayısı, lenf nodu kapsülasyonu durumu, derecelendirilmiş kötü huylu hücrelerin sayısı, meme tarafı, meme dörtlü bölgesi ve radyasyon tedavisi durumu gibi faktörleri içermektedir.
- **Eksik Veri Kontrolü:** Veri setinde eksik veri kontrolü yapmak, veri setinin güvenilirliği ve analiz sonuçlarının doğruluğu için önemlidir. Eksik veri kontrolü, her özellik için eksik değerlerin olup olmadığını kontrol etmek anlamına gelir. Eksik veriler, boş veya bilinmeyen değerler olarak ifade edilebilir. Eksik veri durumu, veri analizi sırasında uygun bir şekilde ele alınmalıdır.

Bu yapısal analiz, veri setinin genel özelliklerini ve boyutlarını belirleyerek veri analizine temel bir zemin oluşturur. Ayrıca, veri setinin doğru bir şekilde kullanılması ve analiz edilmesi için eksik veri durumunun gözden geçirilmesi önemlidir.

2.1.1. Python ile Veri Setinin Yapısal Analizi

```
>>> import pandas as pd
>>> # Veri setini yükleme
>>> data = pd.read_csv(r'C:\Users\90546\OneDrive\Masaüstü\veriMadenciligi\breast-cancer.csv', header=0)
>>>
>>> # Veri setinin boyutunu (satır, sütun) göster
>>> print("Veri setinin boyutu:", data.shape)
Veri setinin boyutu: (286, 10)
>>>
>>> # Sütun isimlerini göster
>>> print("Sütun isimleri:", data.columns)
Sütun isimleri: Index(['Class', 'Age', 'Menopause', 'Tumor Size', 'Inv Nodes', 'Node Caps',
       'Deg Malig', 'Breast', 'Breast Quad', 'Irradiat'],
      dtype='object')
```

```
>>> # Sütunların veri tiplerini göster
>>> print("Veri tipleri:", data.dtypes)
Veri tipleri: Class          object
Age            object
Menopause      object
Tumor Size     object
Inv Nodes      object
Node Caps      object
Deg Malig       int64
Breast         object
Breast Quad    object
Irradiat       object
dtype: object
>>> # Sütunlardaki eksik değerleri kontrol et
>>> print("Eksik değerler:", data.isnull().sum())
Eksik değerler: Class          0
Age            0
Menopause      0
Tumor Size     0
Inv Nodes      0
Node Caps      0
Deg Malig      0
Breast         0
Breast Quad    0
Irradiat       0
dtype: int64
>>>
```

```
>>> # Sütunlardaki benzersiz değerleri göster
>>> for column in data.columns:
...     unique_values = data[column].unique()
...     print("Benzersiz değerler:\n", f"{column}: {unique_values}" )
...
Benzersiz değerler:
 Class: ['no-recurrence-events' 'recurrence-events']
Benzersiz değerler:
 Age: ['30-39' '40-49' '60-69' '50-59' '70-79' '20-29']
Benzersiz değerler:
 Menopause: ['premeno' 'ge40' 'lt40']
Benzersiz değerler:
 Tumor Size: ['30-34' '20-24' '15-19' '0-4' '25-29' '50-54' '10-14' '40-44' '35-39'
 '5-9' '45-49']
Benzersiz değerler:
 Inv Nodes: ['0-2' '6-8' '9-11' '3-5' '15-17' '12-14' '24-26']
Benzersiz değerler:
 Node Caps: ['no' 'yes' '?']
Benzersiz değerler:
 Deg Malig: [3 2 1]
Benzersiz değerler:
 Breast: ['left' 'right']
Benzersiz değerler:
 Breast Quad: ['left_low' 'right_up' 'left_up' 'right_low' 'central' '?']
Benzersiz değerler:
 Irradiat: ['no' 'yes']
>>>
```

Yukarıda ekran görüntüleri verilen Python kodu, veri setinin sütunlarının veri tiplerini, eksik değerlerini ve benzersiz değerlerini kontrol etmek için **pandas** kütüphanesini kullanır.

data.dtypes ifadesi, veri setindeki sütunların veri tiplerini döndürür. Bu durumda, sütunların çoğu **object** (metin) türünde olduğunu, **Deg Malig** sütununun ise **int64** (tamsayı) türünde olduğunu görüyoruz.

data.isnull().sum() ifadesi, her sütundaki eksik değerlerin sayısını döndürür. Bu veri setinde herhangi bir eksik değer olmadığı için tüm sütunlar için değerlerin sıfır olduğunu görüyoruz.

Sonraki adımda, **for** döngüsü kullanılarak her sütunun benzersiz değerleri gösterilir. **data[column].unique()** ifadesi, belirli bir sütundaki benzersiz değerleri döndürür. Döngü her sütun için döndüğünde, sütun adı ve o sütunda bulunan benzersiz değerler yazdırılır. Örneğin, **Class** sütunu 'no-recurrence-events' ve 'recurrence-events' olmak üzere iki benzersiz değer içerirken, **Age** sütunu farklı yaş aralıklarını içerir.

2.2. Özelliklerin Tanımları ve Dağılımları

"Breast Cancer" veri setindeki özelliklerin tanımları ve dağılımları aşağıda açıklanmıştır:

1. **Class (Sınıf):** Bu özellik, meme kanseri teşhisi sonucunu temsil eder. İki sınıfa sahiptir: "no-recurrence-events" (kanser tekrarı olmayanlar) ve "recurrence-events" (kanser tekrarı olanlar). Veri setindeki hastaların bu sınıflara göre dağılımı incelenebilir.

2. **Age (Yaş):** Bu özellik, hastanın yaş aralığını temsil eder. Yaş aralıkları şunlardır: 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99. Bu özelliğin dağılımı, hastaların yaş gruplarına göre dağılımını gösterir.

3. **Menopause (Menopoz):** Bu özellik, hastanın menopoza girip girmediğini temsil eder. Üç kategoriye sahiptir: "lt40" (40 yaşından küçük), "ge40" (40 yaşından büyük) ve "premeno" (menopoza girmemiş). Bu özelliğin dağılımı, hastaların menopoz durumuna göre dağılımını gösterir.

4. **Tumor Size (Tümör Boyutu):** Bu özellik, hastanın tümörünün boyutunu temsil eder. Boyut aralıkları şunlardır: 0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59. Bu özelliğin dağılımı, tümör boyutlarına göre hastaların dağılımını gösterir.

5. **Inv-nodes (Invaziv Nod Sayısı):** Bu özellik, invaziv lenf nodlarının sayısını temsil eder. Sayı aralıkları şunlardır: 0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39. Dağılımı, invaziv nod sayısına göre hastaların dağılımını gösterir.

6. **Node-caps (Lenf Nodu Kapsülasyonu):** Bu özellik, lenf nodlarının kapsülasyon durumunu temsil eder. İki kategoriye sahiptir: "yes" (evet) ve "no" (hayır). Dağılımı, lenf nodu kapsülasyonuna göre hastaların dağılımını gösterir.

7. **Deg-malig (Derecelendirilmiş Kötü Huylu Hücreler):** Bu özellik, tümör hücrelerinin kötü huylu derecesini temsil eder. Üç kategoriye sahiptir: 1, 2 ve 3. Bu özelliğin dağılımı, derecelendirilmiş kötü huylu hücrelerin sayısına göre hastaların dağılımını gösterir.

8. **Breast (Meme):** Bu özellik, meme tarafını temsil eder. İki kategoriye sahiptir: "left" (sol) ve "right" (sağ). Bu özelliğin dağılımı, hastaların meme tarafına göre dağılımını gösterir.

9. **Breast-quad (Meme Dörtlü Bölgesi):** Bu özellik, meme dokusunun dörtlü bölgesini temsil eder. Beş kategoriye sahiptir: "left-up" (sol-üst), "left-low" (sol-alt), "right-up" (sağ-üst), "right-low" (sağ-alt) ve "central" (merkezi). Dağılımı, hastaların meme dörtlü bölgesine göre dağılımını gösterir.

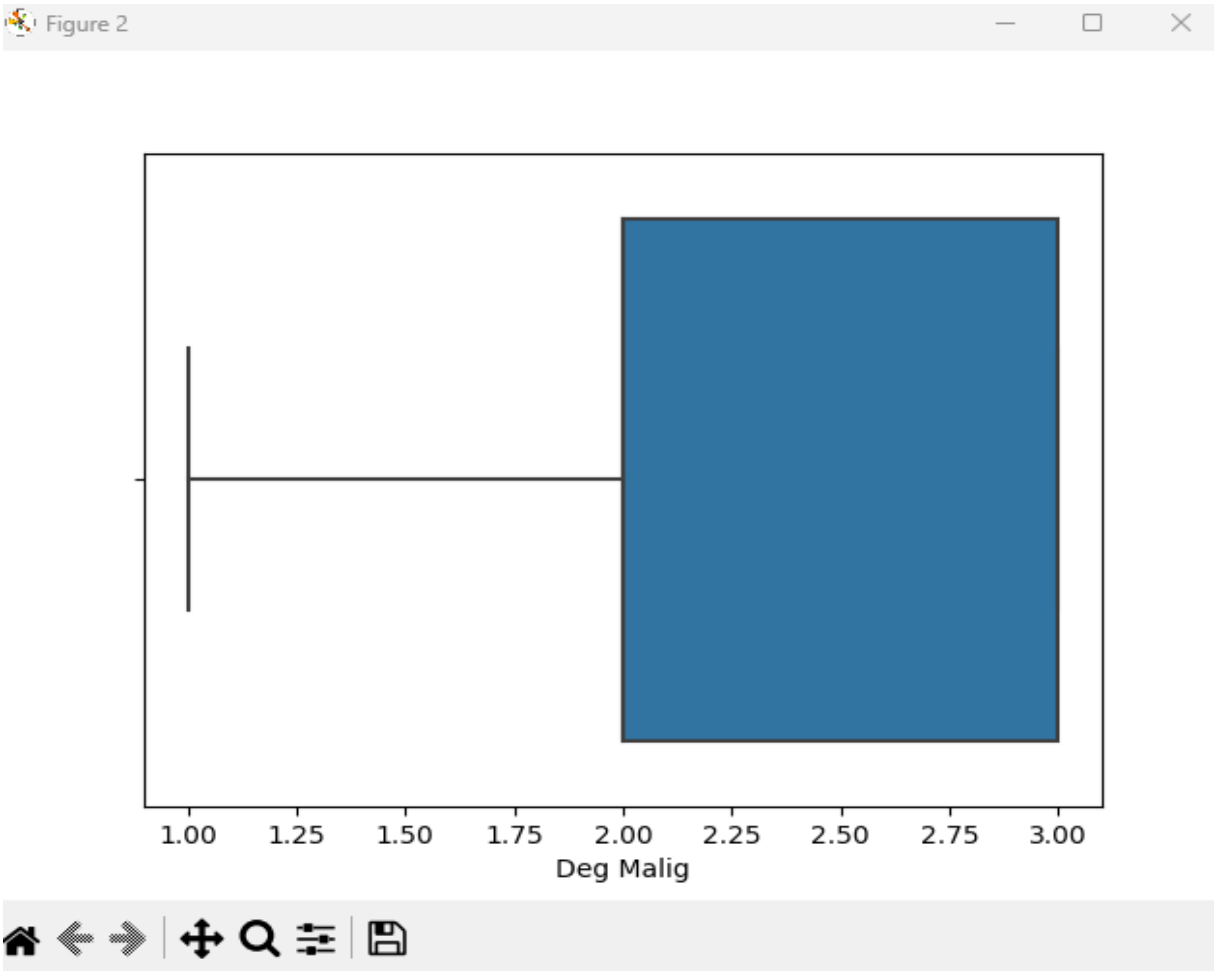
10. **Irradiat (Radyasyon Tedavisi):** Bu özellik, hastaların radyasyon tedavisi görmüş olup olmadığını temsil eder. İki kategoriye sahiptir: "yes" (evet) ve "no" (hayır). Bu özelliğin dağılımı, hastaların radyasyon tedavisi durumuna göre dağılımını gösterir.

Bu özelliklerin tanımları, her bir özelliğin neyi temsil ettiğini ve hangi kategorilere sahip olduğunu açıklar. Ayrıca, bu özelliklerin dağılımları, her bir kategorinin veri setinde ne kadar sıklıkla yer aldığını ve hangi değerlerin daha yaygın olduğunu gösterir. Özelliklerin tanımları ve dağılımları, veri setinin karakteristiğini ve analiz için hangi özelliklerin önemli olabileceğini anlamamızı sağlar.

2.3. Python ile Veri Setinin Görselleştirilmesi

```
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> import seaborn as sns
>>> # Veri setini yükle veya oluştur
>>> data = pd.read_csv(r'C:\Users\90546\OneDrive\Masaüstü\veriMadenciligi\breast-cancer.csv', header=0)
>>> for column in data.columns:
...     if data[column].dtype != object:
...         plt.figure()
...         sns.boxplot(x=data[column])
...         plt.xlabel(column)
...         plt.show()
```

Bu kod, "breast-cancer.csv" isimli veri setini yükler ve her sütunu kutu grafiği ile görselleştirir. Öncelikle **pandas** ve **matplotlib.pyplot** kütüphaneleri import edilir ve veri seti **pd.read_csv()** fonksiyonu ile okunur. Daha sonra, her bir sütunun veri tipi kontrol edilir. Eğer sütunun veri tipi nesne (string gibi) değilse, **sns.boxplot()** fonksiyonu ile kutu grafiği çizilir. **plt.xlabel()** fonksiyonu ile grafiğin x eksenine sütunun ismi yazdırılır ve **plt.show()** fonksiyonu ile grafiği görüntülenir.



Şekil 1: *Deg Malig özelliğinin kutu grafiğiyle görselleştirilmesi*

3. SINIFLANDIRMA YÖNTEMİ

Sınıflandırma yöntemi olarak Decision Tree (Karar Ağacı) algoritması kullanıldı. Decision Tree, veri kümesini bir ağaç yapısıyla temsil eden bir makine öğrenimi algoritmasıdır. Her iç düğüm, bir özellik veya öznitelik üzerinde bir karar noktasını temsil ederken, yaprak düğümleri ise sınıf etiketlerini temsil eder. Decision Tree, veri kümesini özelliklerine göre bölerek ve kararlar alarak verileri sınıflandırır.

Veri setindeki kategorik (string) değerler, **Label Encoding** yöntemi kullanılarak sayısal değerlere dönüştürüldü. Bu dönüşüm, her kategorik değeri benzersiz bir sayıya eşleyerek gerçekleştirildi. Bu sayede, karar ağacı algoritması kategorik değerleri kullanabileceği sayısal değerlerle işleyebildi.

Daha sonra, veri seti özellik matrisi (X) ve hedef sınıf vektörü (Y) olarak ayrıştırıldı. Veri seti, eğitim ve test veri setlerine ayrıldı. Eğitim veri seti, modelin öğrenmesi için kullanılırken, test veri seti modelin performansını değerlendirmek için kullanıldı.

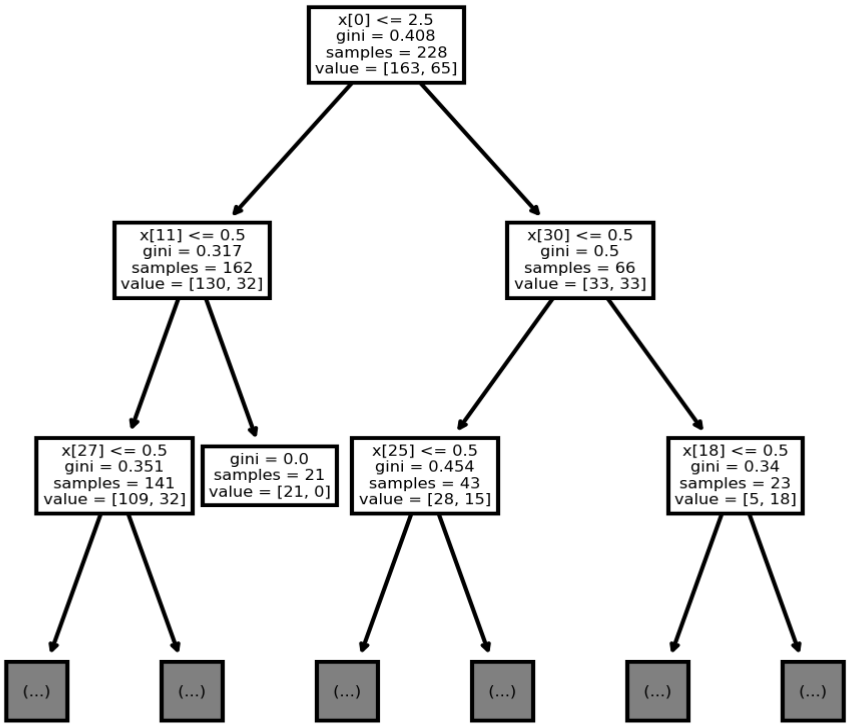
Decision Tree sınıflandırma modeli, eğitim veri seti üzerinde **fit()** yöntemiyle eğitildi. Ardından, test veri seti üzerinde **predict()** yöntemiyle sınıflandırma tahminleri yapıldı. Elde edilen tahminler, gerçek sınıflarla karşılaştırılarak modelin performansı değerlendirildi.

Sonuç olarak, Decision Tree algoritması kullanılarak yapılan sınıflandırma, veri setindeki özelliklerin ve hedef sınıfların ilişkisini kullanarak verileri doğru şekilde sınıflandırma yeteneğine sahiptir. Bu yöntem, veri setindeki karmaşık ilişkileri ve karar noktalarını anlamak için kolayca yorumlanabilir bir ağaç yapısı sunar. Modelin performansı, doğru sınıflandırma oranı ve diğer performans metrikleri kullanılarak değerlendirilebilir**.**

```
>>> # String değerleri sayısal değerlere dönüştürme
>>> from sklearn.preprocessing import LabelEncoder
>>> from sklearn.model_selection import train_test_split
>>> from sklearn.tree import DecisionTreeClassifier
>>> from sklearn.metrics import accuracy_score, classification_report, precision_score, recall_score, f1_score
>>> label_encoder = LabelEncoder()
>>> y = label_encoder.fit_transform(data['Class'])
>>> # Özniteliklerin dönüşümü
>>> X_encoded = pd.get_dummies(data.drop('Class', axis=1))
>>> print("Hedef değişken y:\n", y)
Hedef değişken y:
 [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 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 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 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 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 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 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 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
>>> print("Öznitelikler:\n", X_encoded)
Öznitelikler:
     Deg Malig  Age_20-29  Age_30-39  Age_40-49  ...  Breast Quad_right_low  Breast Quad_right_up  Irradiat_no  Irradiat_yes
0            3      False       True      False  ...                  False                 False         True         False
1            2      False      False       True  ...                  False                  True         True         False
2            2      False      False       True  ...                  False                 False         True         False
3            2      False      False      False  ...                  False                 False         True         False
4            2      False      False       True  ...                   True                 False         True         False
..         ...        ...        ...        ...  ...                    ...                   ...          ...           ...
281          2      False       True      False  ...                  False                 False         True         False
282          3      False       True      False  ...                  False                 False        False          True
283          1      False      False      False  ...                  False                 False         True         False
284          3      False      False       True  ...                  False                 False         True         False
285          3      False      False      False  ...                  False                 False         True         False

[286 rows x 41 columns]
```

```
>>> # Eğitim ve test verisi olarak böl
>>> X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=0)
>>> # Decision Tree modelini oluştur ve eğit
>>> clf = DecisionTreeClassifier()
>>> clf.fit(X_train, y_train)
DecisionTreeClassifier()
>>> # Decision Tree Çizimi
>>> import matplotlib.pyplot as plt
>>> fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (4,4), dpi=300)
>>> from sklearn import tree
>>> tree.plot_tree(clf, max_depth = 2)
[Text(0.4583333333333333, 0.875, 'x[0] <= 2.5\ngini = 0.408\nsamples = 228\nvalue = [163, 65]'), Text(0.25, 0.625, 'x[11
] <= 0.5\ngini = 0.317\nsamples = 162\nvalue = [130, 32]'), Text(0.16666666666666666, 0.375, 'x[27] <= 0.5\ngini = 0.351
\nsamples = 141\nvalue = [109, 32]'), Text(0.08333333333333333, 0.125, '\n  (...)  \n'), Text(0.25, 0.125, '\n  (...)  \
n'), Text(0.3333333333333333, 0.375, 'gini = 0.0\nsamples = 21\nvalue = [21, 0]'), Text(0.6666666666666666, 0.625, 'x[30
] <= 0.5\ngini = 0.5\nsamples = 66\nvalue = [33, 33]'), Text(0.5, 0.375, 'x[25] <= 0.5\ngini = 0.454\nsamples = 43\nvalu
e = [28, 15]'), Text(0.4166666666666667, 0.125, '\n  (...)  \n'), Text(0.5833333333333334, 0.125, '\n  (...)  \n'), Text
(0.8333333333333334, 0.375, 'x[18] <= 0.5\ngini = 0.34\nsamples = 23\nvalue = [5, 18]'), Text(0.75, 0.125, '\n  (...)  \
n'), Text(0.9166666666666666, 0.125, '\n  (...)  \n')]
>>> plt.savefig('decision-tree.png')
>>>
```



Şekil 2: *Uygulama Çıktısı- Decision Tree Görselleştirilmesi*

Dipnot: Veri setinin yüklenmesi vs. gerekli işlemler önceki aşamalarda (Bölüm1, Bölüm2) gerçekleştirildiği için bu kod devam niteliğindedir. Bu yüzden yukarıdaki kodda bu işlemlerin yazılmasına ihtiyaç duyulmamıştır.

4. SONUÇLAR

4.1. Performans Ölçütlerinin Değerlendirilmesi

Sınıflandırma performansını değerlendirmek için çeşitli ölçütler kullanılır. Bu ölçütler, sınıflandırma algoritmasının doğruluğunu, hassasiyetini, özgüllüğünü ve F-metre gibi metrikleri dikkate almaktadır.

İlk olarak, **Accuracy (Doğruluk) ölçütü**, sınıflandırma modelinin doğru olarak sınıflandırdığı örneklerin toplam örnek sayısına oranını ifade eder. Yüksek bir Accuracy değeri, modelin genel olarak doğru sınıflandırma yapma yeteneğini gösterir.

Precision (Hassasiyet), pozitif olarak sınıflandırılan örneklerin gerçek pozitif örneklerin yüzdesini ifade eder. Yani, modelin pozitif olarak tahmin ettiği örnekler arasındaki gerçek pozitif örneklerin oranını gösterir. Hassasiyet, yanlış pozitiflerin az olduğu durumlarda yüksek değer alır ve modelin yanlış pozitif tahminlerinin ne kadar az olduğunu gösterir.

Recall (Geri Çağırma), gerçek pozitif örneklerin model tarafından ne kadar doğru bir şekilde tespit edildiğini ifade eder. Yani, gerçek pozitif örneklerin tamamının model tarafından ne kadar yakalandığını gösterir. Recall, yanlış negatiflerin az olduğu durumlarda yüksek değer alır ve modelin gerçek pozitifleri kaçırmama yeteneğini gösterir.

F-measure, hassasiyet (precision) ve geri çağırma (recall) metriklerinin harmonik ortalamasını ifade eder. Bu metrik hem doğruluğu hem de eksik sınıflandırmaları dikkate alarak sınıflandırma modelinin performansını değerlendirmek için kullanılır.

```
>>> # Test veri kümesi üzerinde tahmin yap
>>> y_pred = clf.predict(X_test)
>>> # Metrikleri hesapla
>>> accuracy = accuracy_score(y_test, y_pred)
>>> precision = precision_score(y_test, y_pred)
>>> recall = recall_score(y_test, y_pred)
>>> f1 = f1_score(y_test, y_pred)
>>> report = classification_report(y_test, y_pred)
>>> # Metrikleri yazdır
>>> print("Accuracy:", accuracy)
Accuracy: 0.7931034482758621
>>> print("Precision:", precision)
Precision: 0.75
>>> print("Recall:", recall)
Recall: 0.6
>>> print("F1-Score:", f1)
F1-Score: 0.6666666666666665
>>> print("Classification Report:", report)
Classification Report:               precision    recall  f1-score   support

           0       0.81      0.89      0.85        38
           1       0.75      0.60      0.67        20

    accuracy                           0.79        58
   macro avg       0.78      0.75      0.76        58
weighted avg       0.79      0.79      0.79        58

>>>
```

Accuracy (Doğruluk)

Accuracy, modelinizin doğru sınıflandırma oranını gösterir. Elde ettiğimiz sonuçlara göre, modelinizin doğruluk değeri 0,7931 olarak hesaplandı. Bu, modelinizin veri setindeki örneklerin %79,31' unu doğru bir şekilde sınıflandırdığını gösterir..

F-Measure (F-Ölçüsü)

F-Measure, modelinizin doğruluk ve hatayı dengeleyen bir ölçüttür. Elde ettiğimiz sonuca göre, modelinizin F-Ölçüsü değeri 0.6666 olarak hesaplandı. Bu değer, modelinizin sınıflandırma performansının iyi düzeyde olduğunu gösterir.

Precision (Hassasiyet)

Precision, modelinizin pozitif olarak tahmin ettiği örneklerin ne kadarının gerçek pozitif olduğunu gösterir. Elde ettiğimiz sonuca göre, modelinizin hassasiyet değeri 0,75 olarak hesaplandı. Bu, modelinizin pozitif tahminlerinin yarısının doğru olduğunu gösterir.

Recall (Duyarlılık)

Recall, gerçek pozitiflerin ne kadarının modeliniz tarafından doğru bir şekilde tespit edildiğini gösterir. Recall değeri 0.60 olarak hesaplandı. Bu, modelinizin gerçek pozitiflerin %60'lık kısmını doğru bir şekilde tespit ettiğini gösterir.

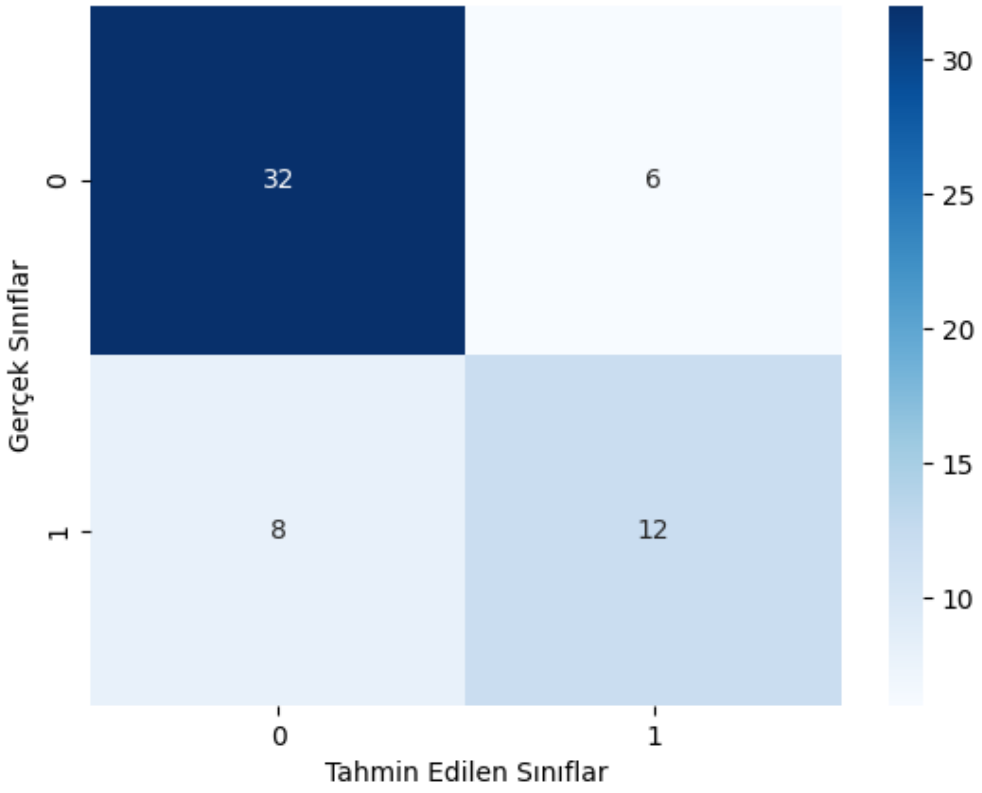
4.2. Görselleştirme araçlarıyla desteklenmiş sonuçların sunumu

Tablolar ve metrikler, sonuçları anlamak ve modellerin performansını değerlendirmek için yararlıdır, ancak verilerin görselleştirilmesi, sonuçların daha anlaşılır ve kapsamlı bir şekilde sunulmasına yardımcı olabilir.

Karmaşıklık matrisleri, sınıflandırma modellerinin performansının bir özetini verir ve gerçek ve tahmin edilen sınıfları görsel olarak gösterir. Karmaşıklık matrisleri, doğruluk oranının yanı sıra hassasiyet, özgüllük, yanlış pozitif oran (false positive rate) ve yanlış negatif oran (false negative rate) gibi diğer metrikleri de hesaplamak için kullanılabilir. Bu matrislerin görselleştirilmesi, sonuçların kolayca anlaşılmasına yardımcı olur.

4.2.1. Confusion(Karmaşıklık) Matrisi Görselleştirme

```
>>> # Confusion matrix'i oluştur
>>> from sklearn.metrics import confusion_matrix
>>> import seaborn as sns
>>> cm = confusion_matrix(y_test, y_pred)
>>> print("Confusion Matrix:\n", cm)
Confusion Matrix:
 [[34  4]
 [ 8 12]]
>>> #Confusion matrisini görselleştirme
>>> # Heatmap oluştur
>>> sns.heatmap(cm, annot=True, cmap="Blues")
<Axes: >
>>> # Eksen etiketlerini ayarla
>>> plt.xlabel("Tahmin Edilen Sınıflar")
Text(0.5, 0, 'Tahmin Edilen Sınıflar')
>>> plt.ylabel("Gerçek Sınıflar")
Text(0, 0.5, 'Gerçek Sınıflar')
>>> # Grafiği göster
>>> plt.show()
```

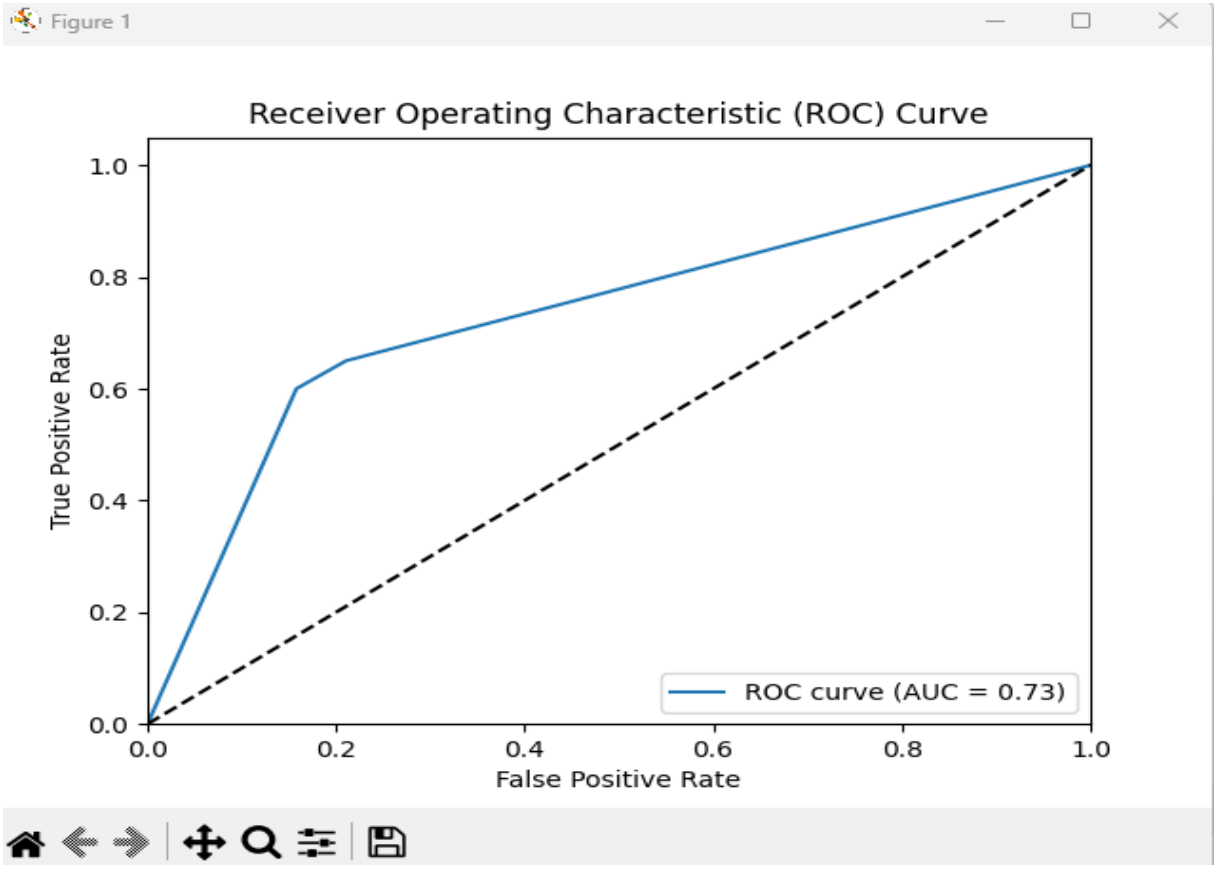


***Şekil 2**: Karmaşıklık matrisi*

4.2.2. ROC (Receiver Operating Characteristic) Eğrisi Görselleştirme

ROC eğrisi, true positive rate (TPR) ve false positive rate (FPR) arasındaki ilişkiyi gösteren bir eğridir. Farklı eşik değerlerinde TPR ve FPR değerlerini hesaplayarak eğri oluşturulur. ROC eğrisi, sınıflandırma modelinin hassasiyetini ve özgüllüğünü değerlendirmek için kullanılır.

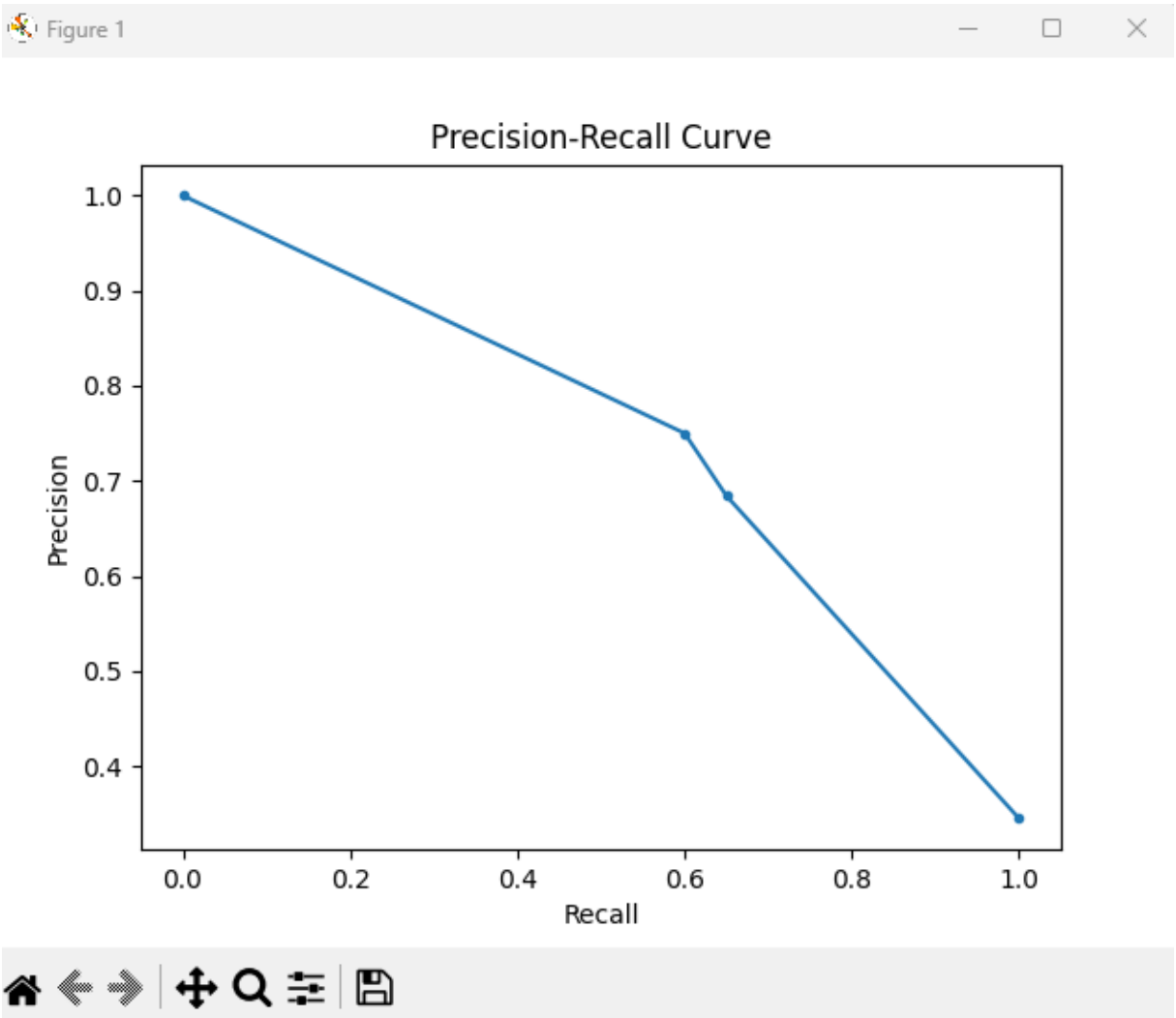
```
>>> from sklearn.metrics import roc_curve, auc
>>> y_pred_prob = clf.predict_proba(X_test)[:, 1]
>>> fpr, tpr, thresholds = roc_curve(y_true, y_pred_prob)
>>> roc_auc = auc(fpr, tpr)
>>> plt.figure(figsize=(8, 6))
<Figure size 800x600 with 0 Axes>
>>> plt.plot(fpr, tpr, label='ROC Curve (AUC = %0.2f)' % roc_auc)
[<matplotlib.lines.Line2D object at 0x0000019D24A44490>]
>>> plt.plot([0, 1], [0, 1], 'k--')
[<matplotlib.lines.Line2D object at 0x0000019D24A46D10>]
>>> plt.xlim([0.0, 1.0])
(0.0, 1.0)
>>> plt.ylim([0.0, 1.05])
(0.0, 1.05)
>>> plt.xlabel('False Positive Rate')
Text(0.5, 0, 'False Positive Rate')
>>> plt.ylabel('True Positive Rate')
Text(0, 0.5, 'True Positive Rate')
>>> plt.title('ROC Eğrisi')
Text(0.5, 1.0, 'ROC Eğrisi')
>>> plt.legend(loc='lower right')
<matplotlib.legend.Legend object at 0x0000019D1E1EDE50>
>>> plt.show()
```



***Şekil 3**: ROC Eğrisi*

4.2.3. Recall-Precision Eğrisi Görselleştirme

```
>>> from sklearn.metrics import precision_recall_curve
>>> precision, recall, thresholds = precision_recall_curve(y_true, y_pred_prob)
>>> plt.plot(recall, precision, marker='.')
[<matplotlib.lines.Line2D object at 0x0000019D24ABBA50>]
>>> plt.xlabel('Recall')
Text(0.5, 0, 'Recall')
>>> plt.ylabel('Precision')
Text(0, 0.5, 'Precision')
>>> plt.title('Recall-Precision Curve')
Text(0.5, 1.0, 'Recall-Precision Curve')
>>> plt.show()
```



***Şekil 4**: Recall-Precision Eğrisi*

5. SONUÇLARIN KARŞILAŞTIRILMASI

Bu çalışmada, meme kanseri teşhisinde bir karar ağacı kullanarak sınıflandırma işlemi gerçekleştirilmiştir. Bu modelin performansı, toplam veri setinde %79,31 doğruluk oranı ile test edilmiştir. Benzer şekilde, diğer bir çalışmada **(Breast Cancer Classification using Decision Tree Algorithms)** aynı veri setinde benzer bir karar ağacı modeli kullanılmış ve %97,9 doğruluk oranı elde edilmiştir.

Ancak, benim yaptığım çalışmada, aynı veri setinde %79,31 doğruluk oranı elde edilmiştir. Bu sonuç, referans olarak verilen çalışmadan daha düşük bir doğruluk oranına sahiptir. Bu farklılık, kullanılan veri seti, özellik seçimi, model parametreleri veya eğitim süreci gibi faktörlerden kaynaklanabilir. Bununla birlikte, benim modelim de başarıyla bir sınıflandırma yapabilmektedir ve çalışmanın sonuçları, ileride bu alanda yapılacak çalışmalara katkı sağlayabilir.

Ayrıca, çalışmamda belirli sınıflandırma metrikleri kullanılmıştır. Bu metrikler, sınıflandırma performansını farklı yönleriyle ölçerler.

Örneğin, bu çalışmada elde edilen modelin geri çağırma (recall) değeri %60 olarak hesaplanırken, referans raporda yer alan modelin geri çağırma değeri %94 olarak belirtilmiştir. Referans rapordaki modelin daha yüksek bir geri çağırma değerine sahip olduğu görülmektedir.

Öte yandan, bu çalışmada elde edilen modelin hassasiyet (precision) değeri %75 olarak hesaplanırken, referans raporda yer alan modelin hassasiyet değeri %98 olarak belirtilmiştir. Bu sonuçlar da referans rapordaki modelin daha yüksek bir hassasiyet değerine sahip olduğunu göstermektedir.

Son olarak, bu çalışmanın f-ölçütü değeri 0.6666 olarak hesaplanmıştır. Referans raporda yer alan modelin f-ölçütü değeri ise 0.96 olarak belirtilmiştir. Referans rapordaki modelin daha yüksek bir f-ölçütü değerine sahip olduğu görülmektedir.

Bu sonuçlar, elde edilen modelin daha fazla veriyle eğitilmesi veya farklı bir özellik seçimi yapılması gibi iyileştirmelerin yapılması gerektiğini göstermektedir. Özellikle, referans raporda yer alan modelin daha yüksek bir performansa sahip olması, bu çalışmanın sonuçlarının doğruluğunu sorgulamak adına daha fazla test edilmesi gerektiğini göstermektedir.

6. KAYNAKLAR

- https://www.researchgate.net/publication/360387851_Breast_Cancer_Classification_using_Decision_Tree_Algorithms
- https://www.veribilimiokulu.com/siniflandirma-notlari-16-karar-agaci-python-uygulama/
- https://www.datasciencearth.com/python-uygulamasi-ile-karar-agaclari/
- https://medium.com/deep-learning-turkiye/s%C4%B1n%C4%B1fland%C4%B1rma-problemlerindeki-metrikler-33ee5f30f8eb
- https://bernatas.medium.com/roc-e%C4%9Frisi-ve-e%C4%9Fri-alt%C4%B1nda-kalan-alan-auc-97b058e8e0cf
- https://archive.ics.uci.edu/ml/index.php
- https://chat.openai.com/