**logo, metin, simge, sembol, kırpıntı çizim içeren bir resim

Açıklama otomatik olarak oluşturuldu**

T.C.

ONDOKUZ MAYIS ÜNİVERSİTESİ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ BİLGİSAYAR MÜHENDİSLİĞİ

ÖRÜNTÜ TANIMA-BM605

Beast Cancer Veri Seti ile Analiz Raporu

FIRAT KAAN BİTMEZ - 3281855

SAMSUN, 2024-2025 Eğitim Öğretim Yılı Güz Yarıyılı

Bu rapor, **Breast Cancer Wisconsin (Diagnostic)** veri seti üzerinde uygulanan boyut indirgeme yöntemlerinin detaylı bir analizini içermektedir. **PCA**, **LDA** ve **t-SNE** teknikleri detaylı olarak incelenmiş, matematiksel altyapıları açıklanmış ve kullanılan her bir kod satırı için açıklamalar sunulmuştur. Ayrıca sınıflandırma performans sonuçları, görselleştirmeler ve yöntemlerin karşılaştırmaları detaylı bir şekilde ele alınmıştır.

**1. Kullanılan Veri Seti**

**Kaynak**: [UCI Machine Learning Repository - Breast Cancer Wisconsin (Diagnostic)](https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic)

* **Amaç**: Meme kanseri teşhisinde iyi huylu (Benign - B) ve kötü huylu (Malignant - M) tümörleri sınıflandırmak.

**Özellikler**

Veri seti toplamda **569 örnek** ve **31 sütundan** oluşmaktadır:

* **ID**: Hastaya atanmış bir kimlik numarasıdır (analiz için gerekli değildir).
* **Diagnosis**: Teşhis sonucu (Benign - İyi huylu veya Malignant - Kötü huylu).
* **Özellik Seti** (30 adet):
  + Ortalama (mean), standart hata (se), ve en kötü değerler (worst) üzerinden hesaplanan 10 temel ölçüt:
    - **Radius**: Hücre çekirdeği çapı.
    - **Texture**: Gri tonlamalı görüntünün piksellerindeki varyasyon.
    - **Perimeter**: Hücre çevresi.
    - **Area**: Hücre alanı.
    - **Smoothness**: Çekirdek sınırlarının düzgünlüğü.
    - **Compactness**: Perimetre² / Alan - 1.
    - **Concavity**: Çekirdeğin dışbükey bölümlerinin ciddiyeti.
    - **Concave points**: Çekirdek sınırındaki içbükey kısımlar.
    - **Symmetry**: Çekirdek simetrisi.
    - **Fractal dimension**: Çekirdek yüzey karmaşıklığı.

**Sınıflar**

* **Benign (İyi Huylu)**: 0
* **Malignant (Kötü Huylu)**: 1
* Veri setinde:
  + **357** iyi huylu,
  + **212** kötü huylu örnek bulunmaktadır.

**Kodda Kullanılan Kütüphaneler**

* **NumPy**: Yüksek performanslı sayısal işlemler, matris hesaplamaları ve veri manipülasyonu.
* **Pandas**: Veri okuma, işleme ve temizleme.
* **Scikit-Learn**: Veri ön işleme, boyut indirgeme, sınıflandırma modelleri ve çapraz doğrulama.
* **Matplotlib ve Seaborn**: Görselleştirme.

**Kodda Veri Setinin Yüklenmesi**

data\_path = os.path.join(data\_dir, "wdbc.data")

column\_names = [

    "ID",

    "Diagnosis",

    "radius\_mean", "texture\_mean", "perimeter\_mean", "area\_mean", "smoothness\_mean",

    ...

    "fractal\_dimension\_worst"

]

df = pd.read\_csv(data\_path, header=None, names=column\_names)

df["Diagnosis"] = LabelEncoder().fit\_transform(df["Diagnosis"])  # M:1, B:0

df.drop(columns=["ID"], inplace=True)

**Açıklama**:

* wdbc.data dosyası CSV formatında yüklenmiştir.
* Diagnosis sütunu (M ve B) **etiket kodlaması** ile 0 ve 1’e çevrilmiştir.
* ID sütunu, analiz için bir anlam taşımadığından kaldırılmıştır.

**2. Veri Ön İşleme**

Özelliklerin farklı ölçeklerde olması sınıflandırıcıların performansını olumsuz etkileyebilir. Bu yüzden tüm özellikler, **StandardScaler** ile standartlaştırılmıştır.

**Kodda Ölçeklendirme**

scaler = StandardScaler()

scaled\_features = scaler.fit\_transform(features)

joblib.dump(scaler, os.path.join(models\_dir, "scaler.pkl"))

**Açıklama**:

* Özellikler ortalaması 0, standart sapması 1 olacak şekilde ölçeklendirilmiştir.
* Kullanılan scaler objesi, gelecekte veri üzerinde aynı işlemi tekrarlayabilmek için **joblib** ile kaydedilmiştir.

**3. Boyut İndirgeme Teknikleri**

* **PCA (Principal Component Analysis - Temel Bileşenler Analizi)**

Matematiksel Altyapı:  
- PCA, verilerdeki maksimum varyansı koruyarak boyut indirir.  
- Özelliklerin korelasyonunu azaltarak daha anlamlı bileşenler elde eder.  
Adımlar:  
1. Verilerin kovaryans matrisi hesaplanır.  
2. Kovaryans matrisinin özdeğerleri ve özvektörleri bulunur.  
3. En büyük özdeğerlere karşılık gelen bileşenler seçilir.  
Matematiksel İfade:  
Kovaryans matrisinin özdeğerleri (λ) ve özvektörleri (v):  
C v = λ v  
Verinin indirgenmesi:  
X yeni = X ⋅ W

* **LDA (Linear Discriminant Analysis - Doğrusal Ayrım Analizi)**

Matematiksel Altyapı:  
- LDA, sınıf içi varyansı minimize ederken sınıflar arasındaki farkı maksimize eder.  
Adımlar:  
1. Sınıf içi ve sınıflar arası kovaryans matrisleri hesaplanır.  
2. Fisher discriminant oranı optimize edilir.  
3. Sınıfları ayıracak projeksiyon vektörü hesaplanır.  
Matematiksel İfade:  
Sınıf içi varyans matrisi (S\_W) ve sınıflar arası varyans matrisi (S\_B):  
S\_W = ∑ (x − μ) (x − μ)^T  
S\_B = ∑ n\_i (μ\_i − μ)(μ\_i − μ)^T  
Projeksiyon vektörünü maksimize eden oran:  
J(w) = w^T S\_B w / w^T S\_W w

* **t-SNE (t-Distributed Stochastic Neighbor Embedding)**

Matematiksel Altyapı:  
- t-SNE, yüksek boyutlu verilerin yapısını koruyarak düşük boyutlu bir haritalama oluşturur.  
Adımlar:  
1. Yüksek boyutlu veride noktalar arasındaki benzerlikler probabilistik olarak hesaplanır.  
2. t-dağılımı ile düşük boyutlu projeksiyonlar oluşturulur.  
3. Kayıp fonksiyonu minimize edilir.  
Matematiksel İfade:  
Benzerliklerin ölçümü:  
P\_{ij} = exp(−∥x\_i − x\_j∥^2 / 2σ\_i^2) / ∑ exp(−∥x\_k − x\_l∥^2 / 2σ\_k^2)  
Projeksiyon için t-dağılım benzerlikleri:  
Q\_{ij} = (1 + ∥y\_i − y\_j∥^2)^−1 / ∑ (1 + ∥y\_k − y\_l∥^2)^−1  
Kayıp fonksiyonu:  
KL(P || Q) = ∑ P\_{ij} log(P\_{ij} / Q\_{ij})

**4. Sınıflandırma Modelleri**

5 farklı makine öğrenmesi algoritması kullanılmıştır:

* **Logistic Regression**: Lineer olarak ayrılabilen veriler için kullanılır. Sınıflar arasındaki olasılık tahminine dayanır.
* **Random Forest**: Karar ağaçlarından oluşan bir topluluk yöntemidir. Rastgele alt kümeler ile çeşitlilik sağlar.
* **Support Vector Machines (SVM)**: Sınıflar arasındaki ayrımı maksimize eden bir hiper düzlem oluşturur.
* **K-Nearest Neighbors (KNN)**: Veri noktalarının en yakın komşularına dayanarak sınıflandırır.
* **Naive Bayes**: Sınıf tahmini için Bayes Teoremi’ni kullanır ve özelliklerin bağımsız olduğunu varsayar.

**Kodda Sınıflandırma ve Değerlendirme**

def evaluate\_models(X, y, models, method\_name):

    results = {}

    kfold = StratifiedKFold(n\_splits=5, random\_state=RANDOM\_STATE, shuffle=True)

    for model\_name, model in models.items():

        scores = cross\_val\_score(model, X, y, cv=kfold, scoring="accuracy")

        results[model\_name] = scores.mean()

    return results

results\_summary = {

    "No Reduction": evaluate\_models(scaled\_features, target, models, "No Reduction"),

    "PCA": evaluate\_models(pca\_features, target, models, "PCA"),

    "LDA": evaluate\_models(lda\_features, target, models, "LDA"),

    "t-SNE": evaluate\_models(tsne\_features, target, models, "t-SNE"),

}

* **Stratified K-Fold Çapraz Doğrulama** kullanılarak modeller test edilmiştir.
* Her boyut indirgeme yöntemi için doğruluk skorları hesaplanmıştır.

**5. Sonuçlar**

Sonuçlar, results\_summary.txt dosyasında özetlenmiştir【12†source】:

| **Yöntem** | **Lojistik Regresyon** | **Random Forest** | **SVM** | **KNN** | **Naive Bayes** |
| --- | --- | --- | --- | --- | --- |
| **Boyut İndirgeme Yok** | %97.37 | %95.43 | %97.54 | %96.66 | %92.97 |
| **PCA** | %95.08 | %93.33 | %95.08 | %93.15 | %91.39 |
| **LDA** | %97.54 | %96.31 | %97.54 | %97.89 | %97.37 |
| **t-SNE** | %95.43 | %96.13 | %95.26 | %95.26 | %95.26 |

* LDA en iyi performansı sağlamıştır.
* PCA ve t-SNE, genel performansı az da olsa düşürmüştür.

**6. Görselleştirme**

Görseller:

1. **PCA Scatter Plot**: PCA ile verilerin dağılımı.
2. **LDA Scatter Plot**: LDA’nın sınıfları ayrıştırma başarısı.
3. **t-SNE Scatter Plot**: Karmaşık verilerin görselleştirilmesi.

Kodda Görselleştirme:

plot\_reduction\_scatter(pca\_features, target, "PCA Scatter Plot", "PCA\_Scatter.png")

plot\_reduction\_scatter(lda\_features, target, "LDA Scatter Plot", "LDA\_Scatter.png")

plot\_reduction\_scatter(tsne\_features, target, "t-SNE Scatter Plot", "tSNE\_Scatter.png")