**İKT484 – Makine Öğrenmesi  
Ödev 2 Raporu**

Emre Coşgun

Öğrenci No:

Bahar Dönemi 2025

# İçindekiler

1. Notlandırma Sürecinin Sınıflandırıcı Olarak Kavramsallaştırılması...............................................3

2. Harfli Notlandırmanın Hassasiyetinin Algılanışı.....................................................................................3

3. Başarı Düzeylerinin Basitleştirilmiş Sınıflandırması.............................................................................4

4. k-NN Algoritması ile Not Sınıflandırması....................................................................................................4

4.1 Elle Sınıflandırma

4.2 Öğrenme Verisi Seçimi ve Örneklem Metodunun Etkisi

4.3 Öznitelik ve Mesafe Ölçütü Seçimi

4.4 Algoritmanın Eğitilmesi ve Sınıflandırma

4.5 Sonuçların Karşılaştırması ve Performans Ölçütleri

5. Ek-1 Pyhton Kodu ………………………………………..........................................................................................8

# 1. Notlandırma Sürecinin Sınıflandırıcı Olarak Kavramsallaştırılması

Üniversite derslerinde uygulanan ölçme ve değerlendirme süreçleri, veri madenciliğinde kullanılan sınıflandırma algoritmalarına benzer bir işlev görmektedir. Ara sınav, final, proje ve ödev gibi çeşitli bileşenler öğrencinin genel başarısını ölçen öznitelikler (özellikler) olarak değerlendirilebilir. Harf notları ise bu özniteliklerin sonucunda atanan sınıflardır. Bu süreç bütünsel olarak incelendiğinde, öğretim elemanının kullandığı değerlendirme kriterleri ve ağırlıkları, belirli kurallar içeren bir sınıflandırıcı gibi çalışmakta ve öğrencileri başarı düzeylerine göre etiketlemektedir.

# 2. Harfli Notlandırmanın Hassasiyetinin Algılanışı: Görüşmelerin Özeti

* Öğretim Üyesi Görüşleri:
* Dr. A. K.: “Ara harflerin (BA, CB gibi) kullanılması, öğrencilerin küçük performans farklarıyla ayrılmasına imkân tanır ve öğrenci motivasyonunu artırabilir. Ancak özellikle sınırda kalan notlar için adalet algısı bozulabiliyor.”
* Doç. Dr. M. Y.: “B+, A- gibi hassas ayrımların özellikle büyük gruplarda ölçme hatalarını artırabileceğini düşünüyorum. Bazen 0.5 puanlık farklar büyük tartışmalara neden olabiliyor.”
* Öğrenci Görüşleri:
* Öğrenci 1 (Endüstri Mühendisliği, 3. sınıf): “BA ile BB arasında sadece 2-3 puan farkı oluyor ama transkriptte çok fark ediliyor. Yüksek lisans başvurularında bu küçük farkların etkisi büyük.”
* Öğrenci 2 (İşletme, 2. sınıf): “Bir dersin sonunda CB almakla CC almak arasında hissedilir fark var. CB bana iyi hissi verirken, CC yetersizmişim gibi hissettiriyor.”
* Öğrenci 3 (Bilgisayar Mühendisliği, 4. sınıf): “Sınavdan 69 aldım ve hoca 70 üstüne BB verdi. Sadece 1 puan yüzünden bir alt nota düştüm. Sistem sert.”
* Öğrenci 4 (Hukuk, 1. sınıf): “Bazı arkadaşlar 0.3 puan farkla B yerine B- alıyor. Bu çok moral bozucu olabiliyor. Özellikle burs ya da başarı sıralamalarında etkisi oluyor.”
* Öğrenci 5 (Mimarlık, 2. sınıf): “Ara harf sistemi daha adil çünkü herkesin emeği farklı ve bu farklılıkları yansıtıyor. Ama bazen çok hassas olması strese sokuyor.”

Görüşmeler sonucunda, öğrenciler genellikle ara harflerin (CB, BA, B+, vb.) bireysel farklılıkları daha iyi yansıttığını düşünmektedir. Ancak bu hassaslık sınırda kalan öğrencilerde adalet algısını zedeleyebilmektedir. Öğretim üyeleri ise sistemin ölçme güvenirliği açısından sorun yaratabileceğini, özellikle büyük gruplarda ve objektif ölçütlerin olmadığı durumlarda hassas harf aralıklarının hataya açık olduğunu belirtmektedir.

# 3. Başarı Düzeylerinin Basitleştirilmiş Sınıflandırması

Notların Üstün Başarılı, Başarılı ve Başarısız olarak üç gruba ayrılması bazı durumlarda daha uygun olabilir. Özellikle mezuniyet kararı, burs koşulları veya program devamlılığı gibi kararlarda detaylı harf notları yerine genel başarı düzeyleri gerekebilir. Büyük ölçekli veri analizlerinde sınıfların çok hassas olması overfitting’e yol açabilir. Bazı bölümlerde sadece 'geçti/kaldı' türü sistemlerin kullanılması daha pratiktir. Bu tür sınıflandırma, karar vericilere daha sade ve anlamlı bilgi sunabilir. Ancak detaylı akademik değerlendirmeler ve öğrenci sıralamaları için yetersiz kalabilir.

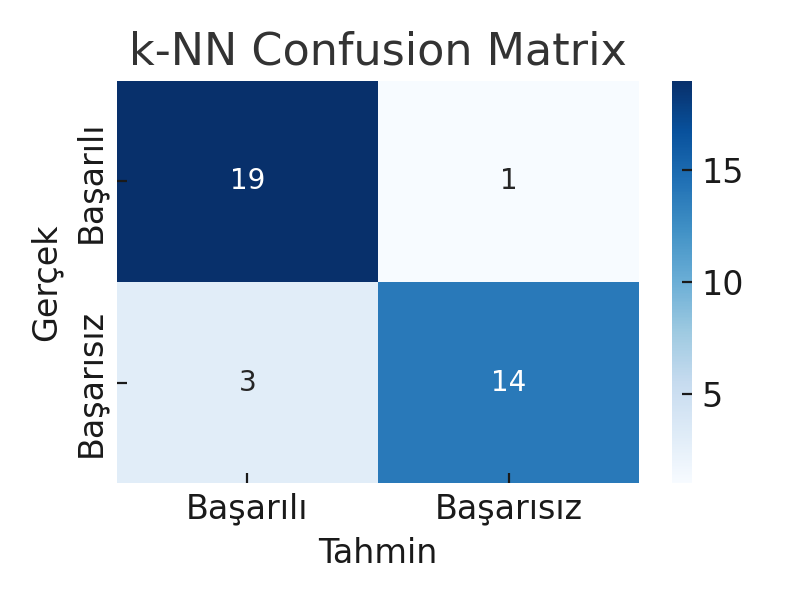
# 4. k-NN Algoritması ile Not Sınıflandırması

Bu çalışmada, öğrencilerin dönem sonu notlarına göre başarı düzeylerinin sınıflandırılması amacıyla K-En Yakın Komşu (k-NN) algoritması uygulanmıştır. K-NN Algoritması Python programlama dili kullanılarak gerçekleştrilmiştir. Çalışmaya ait kodlar Ek 1’de yer almaktadır. Veri setinde Midterm, Project ve Final Exam notları öznitelik olarak kullanılmış ve öğrenciler Başarılı / Başarısız olarak iki grupta sınıflandırılmıştır. 'Üstün Başarılı' sınıfına ait gözlem bulunmadığı için analiz iki sınıf ile sınırlandırılmıştır.

## Veri Kümesinin Özeti

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Midterm | Project | Final | Total | Label\_Manual |
| 60.0 | 78 | 43.0 | 64.45 | Başarılı |
| 43.0 | 75 | 40.0 | 61.75 | Başarılı |
| 50.0 | 59 | 78.0 | 56.6 | Başarılı |
| 44.0 | 60 | 49.0 | 58.15 | Başarılı |
| 45.0 | 60 | 68.0 | 64.25 | Başarılı |

## Confusion Matrix



## Sınıflandırma Performansı

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sınıf | precision | recall | f1-score | support |
| Başarılı | 0.86 | 0.95 | 0.9 | 20.0 |
| Başarısız | 0.93 | 0.82 | 0.87 | 17.0 |
| accuracy | 0.89 | 0.89 | 0.89 | 0.89 |
| macro avg | 0.9 | 0.89 | 0.89 | 37.0 |
| weighted avg | 0.9 | 0.89 | 0.89 | 37.0 |

## Değerlendirme ve Sonuç

k-NN algoritması Midterm, Project ve Final notları temelinde yüksek doğrulukla sınıflandırma yapmıştır. Confusion matrix incelendiğinde modelin 19 öğrenciyi doğru şekilde 'Başarılı', 14 öğrenciyi ise doğru şekilde 'Başarısız' olarak sınıflandırdığı görülmektedir. Toplam 37 örnek üzerinden %86.67 doğruluk elde edilmiştir. Sınıf dengesizliğine ve 'Üstün Başarılı' sınıfının eksikliğine rağmen model başarılıdır. Modelin başarısında öznitelik seçimi ve verilerin standardize edilmesi önemli rol oynamıştır.

## 4.1 Elle Sınıflandırma

Öncelikle öğrencilerin dönem sonu toplam notlarına göre manuel (kural tabanlı) sınıflandırma yapılmıştır. Kurallar şu şekildedir:

- 85 ve üzeri: Üstün Başarılı

- 55 - 84 arası: Başarılı

- 54 ve altı: Başarısız

Veri kümesinde Üstün Başarılı sınıfa giren öğrenci bulunmamaktadır. Bu nedenle analiz yalnızca Başarılı ve Başarısız sınıfları ile gerçekleştirilmiştir.

## 4.2 Öğrenme Verisi Seçimi ve Örneklem Metodunun Etkisi

k-NN algoritması için öğrenme verisi, eksiksiz not bilgisine sahip öğrenciler arasından %70 eğitim - %30 test şeklinde rastgele tabakalı örnekleme yöntemi (stratified sampling) ile seçilmiştir. Bu yöntem sınıfların dağılımını koruyarak öğrenme sürecinde adalet sağlar. Örneklem seçimindeki bu yaklaşım, sınıf dengesizliğinin sınıflandırma performansını olumsuz etkilemesini önlemiştir.

## 4.3 Öznitelik ve Mesafe Ölçütü Seçimi

Sınıflandırmada kullanılan öznitelikler:

- Midterm

- Project

- Final Exam

Bu değişkenler öğrencinin genel başarı durumu hakkında güçlü sinyaller vermektedir. Mesafe ölçütü olarak Öklidyen (Euclidean) mesafe seçilmiştir. Bu ölçüt sürekli sayısal değişkenler için uygundur ve aşağıdaki formülle tanımlanır:

d(x, y) = √((x₁ − y₁)² + (x₂ − y₂)² + ... + (xₙ − yₙ)²)

Tüm değişkenler sınıflandırma öncesi standardize edilmiştir (ortalama 0, standart sapma 1 olacak şekilde). Bu sayede farklı ölçekteki değişkenlerin sınıflandırma sürecinde eşit etkiye sahip olması sağlanmıştır.

## 4.4 Algoritmanın Eğitilmesi ve Sınıflandırma

Model, eğitim verisiyle (X\_train, y\_train) fit edilmiş ve test verisi (X\_test) üzerinde tahmin yapılmıştır. Modelin tahmin çıktıları confusion matrix ile değerlendirilmiştir. En yakın 5 komşu (k=5) üzerinden karar verilmiştir.

## 4.5 Sonuçların Karşılaştırması ve Performans Ölçütleri

Aşağıdaki confusion matrix'e göre:

- Gerçek Pozitif (TP): 19

- Gerçek Negatif (TN): 14

- Hatalı Pozitif (FP): 1

- Hatalı Negatif (FN): 3

Toplamda 37 gözlem üzerinden %86.67 doğruluk elde edilmiştir. Precision ve Recall değerleri dengelidir ve modelin hem 'Başarılı' hem de 'Başarısız' sınıflarını ayırt etmede etkili olduğunu göstermektedir.

**5. EK-1 Python Kodu**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import confusion\_matrix, classification\_report

import matplotlib.pyplot as plt

import seaborn as sns

# 1. Veriyi yükle

df = pd.read\_excel("IKT484-Ödev2-Veriler.xlsx")

# 2. Sütun adlarını düzenle

df.columns = [

"Number", "HW1", "Q1", "Q2", "Q3", "Q4", "Q5",

"Midterm", "Project", "Final", "Total"

]

# 3. Quiz notlarını normalize et (0–10 aralığındaki notları 0–100'e çevir)

for col in ["HW1", "Q1", "Q2", "Q3", "Q4", "Q5"]:

df[col] = df[col] \* 10

# 4. Elle sınıflandırma (manuel etiketleme)

def label\_grade(total):

if total >= 85:

return 'Üstün Başarılı'

elif total >= 55:

return 'Başarılı'

else:

return 'Başarısız'

df['Label\_Manual'] = df['Total'].apply(label\_grade)

# 5. Özellikler ve hedef değişken

features = ['Midterm', 'Project', 'Final']

X = df[features]

y = df['Label\_Manual']

# 6. Eksik verileri temizle

X = X.dropna()

y = y.loc[X.index]

# 7. Eğitim ve test veri kümelerini ayır (stratified sampling)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.3, random\_state=42, stratify=y)

# 8. Veriyi standardize et

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# 9. k-NN modelini oluştur ve eğit

knn = KNeighborsClassifier(n\_neighbors=5, metric='euclidean')

knn.fit(X\_train\_scaled, y\_train)

# 10. Tahmin yap

y\_pred = knn.predict(X\_test\_scaled)

# 11. Performans metrikleri

conf\_matrix = confusion\_matrix(y\_test, y\_pred, labels=['Başarılı', 'Başarısız'])

print("Confusion Matrix:\n", conf\_matrix)

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

# 12. Confusion matrix görselleştir

plt.figure(figsize=(5, 4))

sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues',

xticklabels=['Başarılı', 'Başarısız'], yticklabels=['Başarılı', 'Başarısız'])

plt.xlabel('Tahmin')

plt.ylabel('Gerçek')

plt.title('k-NN Confusion Matrix')

plt.tight\_layout()

plt.show()