🎬 ML Tabanlı Film & Dizi Öneri Sistemi

> ***Makine Öğrenmesi Projesi*** | *6 Farklı ML Algoritması | TMDB Veri Seti*

📋 İçindekiler

🎯 Proje Amacı

Bu proje, **TMDB (The Movie Database)** veri setini kullanarak kullanıcılara film ve dizi önerileri sunan bir **Makine Öğrenmesi** uygulamasıdır.

Temel Hedefler

HEDEF	AÇIKLAMA
ML Karşılaştırması	6 farklı ML algoritmasını aynı veri seti üzerinde karşılaştırmak
En İyi Model Seçimi	Performans metriklerine göre en uygun modeli otomatik seçmek
Görsel Analiz	Veri setinin detaylı istatistiksel analizini sunmak
Kullanıcı Arayüzü	Streamlit ile interaktif bir web uygulaması geliştirmek

Problem Tanımı

> ***"Bir kullanıcı X filmini seviyorsa, hangi filmleri de sever?"***

Bu soruyu cevaplamak için farklı ML yaklaşımları kullanılmıştır.

📊 Kullanılan Veri Seti

TMDB Veri Seti

DOSYA	KAYIT SAYISI	SÜTUN SAYISI	AÇIKLAMA
`tmdb_5000_movies.csv`	4,803	20	Film bilgileri
`tmdb_5000_credits.csv`	4,803	4	Oyuncu ve ekip bilgileri
`TMDB_tv_dataset_v3.csv`	168,639	29	Dizi bilgileri

Önemli Özellikler (Features)

ÖZELLIK	VERI TIPI	AÇIKLAMA	ML KULLANIMI
`overview`	Metin	Film/dizi özeti	TF-IDF vektörizasyonu
`genres`	JSON	Türler listesi	One-hot encoding
`keywords`	JSON	Anahtar kelimeler	TF-IDF
`cast`	JSON	Oyuncular	Feature olarak kullanılır
`crew`	JSON	Ekip (yönetmen vb.)	Yönetmen bilgisi çıkarılır
`vote_average`	Float	Ortalama puan (0-10)	Hedef değişken
`vote_count`	Integer	Oy sayısı	Güvenilirlik filtresi
`popularity`	Float	Popülerlik skoru	Numeric feature

🤖 ML Algoritmaları

Bu projede **6 farklı makine öğrenmesi algoritması** kullanılmıştır:

1️⃣ İçerik Tabanlı Filtreleme (TF-IDF + Kosinüs Benzerliği)

Nasıl Çalışır?

```
Film Özeti → TF-IDF Vektörü → Kosinüs Benzerliği → Benzer Filmler
```

Matematiksel Formül

TF-IDF (Term Frequency - Inverse Document Frequency):

```
TF(t,d) = Terimin dökümandaki frekansı / Toplam terim sayısı
IDF(t) = log(Toplam döküman sayısı / Terimi içeren döküman sayısı)
TF-IDF(t,d) = TF(t,d) × IDF(t)
```

Kosinüs Benzerliği:

```
cos(θ) = (A · B) / (||A|| × ||B||)
```

- İki vektör arasındaki açıyı ölçer

- 0 = hiç benzer değil, 1 = tamamen aynı

Avantajları

- ✅ Yeni içerikler için hemen çalışır (Cold-start yok)
- ✅ Yorumlanması kolay
- ✅ Kullanıcı verisi gerektirmez

Dezavantajları

- ❌ Sadece içerik benzerliğine bakar
- ❌ Surprise factor düşük

2️⃣ K-En Yakın Komşu (KNN)

Nasıl Çalışır?

```
Hedef Film → Feature Vektörü → En Yakın K Komşuyu Bul → Öner
```

Çalışma Prensibi

1. Her filmi bir feature vektörüne dönüştür
2. Hedef filmin vektörünü al
3. Tüm filmlerle mesafe hesapla
4. En yakın K filmi döndür

Mesafe Metrikleri

METRIK	FORMÜL	KULLANIM
Öklid	$\sqrt{\Sigma(xi-yi)^2}$	Genel amaçlı
Kosinüs	$1 - \cos(\theta)$	Metin verisi için
Manhattan	$\Sigma\|xi-yi\|$	Yüksek boyutlu veri

Avantajları

- ✅ Basit ve anlaşılır
- ✅ Non-parametrik (varsayım yok)
- ✅ Lazy learning (hızlı eğitim)

3️⃣ Random Forest

Nasıl Çalışır?

```
Veri → Bootstrap Örnekleme → N Karar Ağacı → Ensemble Tahmin
```

Çalışma Prensibi

1. **Bootstrap Aggregating (Bagging):** Veri setinden rastgele örnekler al
2. **Karar Ağaçları:** Her örneklem için bir ağaç eğit
3. **Ensemble:** Tüm ağaçların tahminlerini birleştir

Hiperparametreler

PARAMETRE	DEĞER	AÇIKLAMA
`n_estimators`	100	Ağaç sayısı
`max_depth`	10	Maksimum derinlik
`random_state`	42	Tekrarlanabilirlik

Feature Importance

Random Forest, hangi özelliklerin model için en önemli olduğunu gösterir:

- Türler (%35)
- Anahtar kelimeler (%25)
- Oyuncular (%20)
- Diğer (%20)

4️⃣ Lineer Regresyon (Ridge)

Nasıl Çalışır?

```
Özellikler → Lineer Model → Puan Tahmini → Benzer Tahminli Filmler
```

Matematiksel Formül

Ridge Regresyon (L2 Regularization):

```
β = argmin { Σ(yi - xi'β)² + λΣβj² }
```

- `λ`: Regularization gücü (overfitting önleme)

Avantajları

- ✅ Çok hızlı eğitim
- ✅ Yorumlanabilir katsayılar
- ✅ Regularizasyon ile overfitting önlenir

5️⃣ SVD (Tekillik Ayrışımı)

Nasıl Çalışır?

```
TF-IDF Matrisi → SVD Ayrışımı → Düşük Boyutlu Uzay → Benzerlik
```

Matematiksel Formül

```
A = U × Σ × V'
```

MATRIS	BOYUT	ANLAMI
U	m × k	Sol tekil vektörler (film faktörleri)
Σ	k × k	Tekil değerler (önem dereceleri)
V'	k × n	Sağ tekil vektörler (özellik faktörleri)

Avantajları

- ✅ Boyut indirgeme (3000 → 100)
- ✅ Gürültü azaltma
- ✅ Latent (gizli) özellikleri keşfeder

6️⃣ Sinir Ağı (MLP - Multi-Layer Perceptron)

Nasıl Çalışır?

```
Girdi → Gizli Katmanlar → Aktivasyon → Çıktı
```

Model Mimarisi

```
┌─────────────────────────────────────┐
│         Input Layer (1000)          │
├─────────────────────────────────────┤
│      Hidden Layer 1 (256, ReLU)     │
├─────────────────────────────────────┤
│      Hidden Layer 2 (128, ReLU)     │
├─────────────────────────────────────┤
│      Hidden Layer 3 (64, ReLU)      │
├─────────────────────────────────────┤
│         Output Layer (1)            │
└─────────────────────────────────────┘
```

Hiperparametreler

PARAMETRE	DEĞER	AÇIKLAMA
`hidden_layers`	(256, 128, 64)	Gizli katman boyutları
`activation`	ReLU	Aktivasyon fonksiyonu
`solver`	Adam	Optimizasyon algoritması
`max_iter`	200	Maksimum iterasyon
`early_stopping`	True	Erken durdurma

Avantajları

- ✅ Non-linear ilişkileri yakalar
- ✅ Universal approximator
- ✅ Büyük veri setlerinde etkili

📈 Model Karşılaştırması

Değerlendirme Metrikleri

METRIK	AÇIKLAMA	FORMÜL
Eğitim Süresi	Modelin eğitilme süresi	saniye
Öneri Süresi	Tek bir öneri için geçen süre	milisaniye
Kapsam	Önerilen benzersiz film oranı	(benzersiz öneriler / toplam film) × 100
Çeşitlilik	Önerilerdeki tür çeşitliliği	(benzersiz türler / toplam tür) × 100
Ortalama Puan	Önerilen filmlerin ortalama puanı	0-10

Karşılaştırma Sonuçları (Örnek)

MODEL	EĞITIM	ÖNERI	KAPSAM	ÇEŞITLILIK	SKOR
İçerik Tabanlı	1.2s	0.01s	2.5%	45%	0.72
KNN	0.8s	0.02s	2.0%	40%	0.68
Random Forest	3.5s	0.05s	1.8%	50%	0.65
Lineer	0.5s	0.01s	1.5%	35%	0.60
SVD	1.0s	0.01s	2.2%	42%	0.70
Sinir Ağı	5.0s	0.02s	2.0%	48%	0.66

Skor Hesaplama Formülü (Güncellenmiş)

```
# Çoklu Metrik Skoru (3 bileşen):
rating_score = avg_rating * 10          # Puan (7.0 = 70)
precision_bonus = 15 if precision >= 70 # İyi film oranı bonusu
speed_bonus = 5 if rec_time < 50ms      # Hız bonusu

Toplam Skor = rating_score + precision_bonus + speed_bonus
```

> 🏆 *Skor ⩾ 80 olan modeller "Kabul Edildi" olarak işaretlenir.*

Accuracy Metrikleri

METRIK	AÇIKLAMA
Precision	İyi film önerme oranı (puan ⩾ 5.5)
TP/FP	True Positive / False Positive sayısı
Confusion Matrix	Model uyum matrisi
Performans Isı Haritası	Görsel karşılaştırma

📁 Proje Yapısı

```
dizi_film_oneri_ai/
│
├── 📊 VERİ DOSYALARI
│   ├── tmdb_5000_movies.csv      # Film verisi
│   ├── tmdb_5000_credits.csv     # Oyuncu/Ekip verisi
│   └── TMDB_tv_dataset_v3.csv    # Dizi verisi
│
├── 🐍 PYTHON MODÜLLER DOMENİLİER
│   ├── app.py                    # Ana Streamlit uygulaması
│   ├── data_analysis.py          # Veri analizi modülü
│   ├── ml_models.py              # 6 ML algoritması
│   └── model_comparison.py       # Model karşılaştırma
│
└── 📋 YAPILANDIRMA
    └── requirements.txt          # Python bağımlılıkları
```

Modül Açıklamaları

DOSYA	SATIR	AÇIKLAMA
`app.py`	~450	4 sayfalı Streamlit UI
`data_analysis.py`	~350	Veri yükleme, temizleme, görselleştirme
`ml_models.py`	~500	6 ML sınıfı, eğitim ve öneri fonksiyonları
`model_comparison.py`	~300	Model değerlendirme ve grafikler

🚀 Kurulum ve Çalıştırma

Gereksinimler

- Python 3.8+
- pip

Adım 1: Bağımlılıkları Yükle

```
pip install -r requirements.txt
```

Adım 2: Uygulamayı Başlat

```
streamlit run app.py
```

Adım 3: Tarayıcıda Aç

```
http://localhost:8501
```

⚡ Optimizasyon: Pre-trained Modeller

Neden Pre-training?

Büyük veri setleri (168k dizi) her seferinde eğitilirse:

- Yüksek RAM kullanımı
- Uzun bekleme süreleri

Çözüm:

```
# İlk seferde modelleri eğit ve kaydet
python preprocess_and_train.py

# Sonraki çalıştırmalarda hızlı yükleme
streamlit run app.py
```

Kaydedilen Dosyalar:

```
processed_data/
├── movies_processed.pkl
└── tv_processed.pkl

trained_models/
├── movies_content_based.pkl
├── movies_knn.pkl
├── movies_rf.pkl
├── movies_linear.pkl
├── movies_svd.pkl
└── movies_mlp.pkl
```

📸 Uygulama Sayfaları

📊 Sayfa 1: Veri Analizi

- Veri seti istatistikleri
- Tür dağılımı grafikleri
- Puan dağılımı histogramları
- Korelasyon ısı haritası
- En iyi içerikler listesi

🎬 Sayfa 2: Öneri Sistemi

- Model seçimi (6 seçenek)
- Film arama ve seçme
- Benzerlik skorlu öneriler
- Progress bar ile görsel skor

📈 Sayfa 3: Model Karşılaştırma

- Tüm modelleri değerlendir
- Performans tablosu
- Karşılaştırma grafikleri
- En iyi model seçimi

📋 Sayfa 4: Teknik Dokümantasyon

- Algoritma açıklamaları
- Matematiksel formüller
- Veri seti bilgileri

👷 Geliştirici Notları

Kullanılan Kütüphaneler

KÜTÜPHANE	VERSIYON	KULLANIM AMACI
pandas	2.x	Veri işleme
scikit-learn	1.x	ML algoritmaları
streamlit	1.x	Web arayüzü
matplotlib	3.x	Görselleştirme
seaborn	0.x	İstatistiksel grafikler
numpy	1.x	Sayısal hesaplamalar

Gelecek Geliştirmeler

- [ ] Collaborative Filtering (kullanıcı bazlı)
- [ ] Deep Learning embeddings (Word2Vec)
- [ ] API entegrasyonu
- [ ] Kullanıcı tercihi kaydetme

📄 Lisans

🎬 ML Film Öneri Sistemi

6 Farklı Algoritma ile Akıllı Öneriler