KÜTAHYA SAĞLIK BİLİMLERİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BÖLÜMLERİ FAKÜLTESİ

KOMUT MÜHENDİSLİĞİ

Komut Mühendisliği İle Kodlama

Mustafa AKER

2118121039

1 Giriş

Film öneri sistemleri, kullanıcılara daha önce izledikleri veya ilgilerini çekebilecek filmleri önerme amacı güden yazılım sistemleridir. Bu sistemler, genellikle büyük veri kümelerindeki örüntüleri kullanarak, kullanıcının tercihlerine göre uygun içerikler sunmayı hedefler. Bu tür uygulamalar, kullanıcıların film keşfetme deneyimini zenginleştirirken, doğru önerilerle kullanıcı memnuniyetini artırabilir.

Bu rapor, bir Film Öneri Uygulaması'nın analizini sunmaktadır. Flask tabanlı bir web uygulaması olarak geliştirilen bu sistem, Full MovieLens Dataset verilerini kullanarak film önerileri sunar. Uygulama, kullanıcılara belirli bir türde ve yılda çıkmış filmleri önermekte ve her film için detaylı bilgiler sunmaktadır. Uygulama, veritabanından alınan verileri filtreler ve önerileri kullanıcıya JSON formatında döndürür.

Bu rapor, uygulamanın temel yapısının, kullanılan teknolojilerin, veri setinin analizinin ve uygulamanın çalışma yöntemlerinin detaylı bir şekilde incelenmesini içermektedir.Ayrıca uygulama farklı tür ve yıl bilgileri ile denenmiş , oluşan Farklı varyasyonların çıktıları EKLER kısmında sunulmuştur

2 Uygulama İçeriği

Film öneri uygulaması, kullanıcıların istedikleri film türü ve yılına göre önerilen filmleri görmek için geliştirilmiştir. Uygulama, Flask web framework'ü, pandas veri işleme kütüphanesi ve JavaScript kullanarak geliştirilmiştir.

2.1 Uygulamanın Temel Özellikleri

- **Film Önerisi:**
  Kullanıcılar, belirli bir film türü ve yılı seçerek öneriler alabilirler.
- **Film Detayları:**
  Her film hakkında başlık, tür, özet, oyuncular, anahtar kelimeler ve daha fazlasını içerir.
- **Frontend ve Backend İletişimi:**
  HTML ve JavaScript ile kullanıcıdan alınan bilgiler, Flask API aracılığıyla backend'e iletilir. Backend, bu verileri kullanarak veritabanında filtreleme yapar ve sonuçları JSON formatında frontend'e döndürür.

- **CORS Desteği:**
  Uygulama, farklı domainlerden gelen talepler için CORS (Cross-Origin Resource Sharing) desteği sağlar.

2.2 Kullanıcı Arayüzü

- **HTML Formu:**
  Kullanıcılar film türünü ve yılını girerek film önerisi talep ederler. Bu formda kullanıcıdan alınan veriler, backend API'sine gönderilir.

- **JavaScript İşlemleri:**
  Kullanıcıdan alınan bilgilerle backend API'ye POST isteği yapılır ve sonuçlar sayfada görüntülenir.

3 Kod Yapısının Analizi

Film öneri uygulaması, Flask (Python) ile geliştirilen backend ve HTML, JavaScript ile oluşturulmuş frontend kısmından oluşmaktadır.

3.1 Backend (Flask Uygulaması)

Backend kısmında Flask framework'ü kullanılmaktadır. Flask, Python tabanlı bir web framework'ü olup, RESTful API'ler oluşturmak için oldukça yaygın bir tercihtir. Flask, minimal bir web framework'üdür ve bu uygulama da API'lerin hızlıca oluşturulmasını sağlar. Flask backend, veri setini yükler, kullanıcıdan gelen film türü ve yılına göre filtreleme yapar ve sonuçları JSON formatında döndürür.

Önemli Kod Parçaları:

- **Flask ve CORS Kullanımı**
  Uygulama, Flask ile geliştirilmiştir ve CORS (Cross-Origin Resource Sharing) kullanılarak frontend ile API arasında veri alışverişi sağlanmaktadır.

```
4   from flask_cors import CORS  # CORS modülünü ekleyin
5
6   app = Flask(__name__)
7
8   # CORS'u etkinleştirin, tüm origin'lere izin verir.
9   CORS(app, resources={r"/api/*": {"origins": "*"}})
10
```

Bu satır, API'nin farklı origin'lerden gelen talepleri kabul etmesini sağlar.

- **Veri Setinin Yüklenmesi ve İşlenmesi**
  Veri setleri (movies.metadata.csv, credits.csv, keywords.csv) pandas kullanılarak yüklenir ve işlenir. Özellikle, CSV dosyaları pandas read.csv() fonksiyonu ile yüklenir ve astype(str) fonksiyonu ile 'id' sütunları string tipine dönüştürülür.

```
11  # NaN ve None değerleri kontrol eden fonksiyon
12  def safe_get_value(value):
13      if value is None or isinstance(value, float) and math.isnan(value):
14          return ""  # Eğer None veya NaN ise, boş bir string döndürelim
15      return value
16
17  # CSV dosyalarını yükleyelim
18  movies_df = pd.read_csv('movies_metadata.csv', low_memory=False)
19  credits_df = pd.read_csv('credits.csv', low_memory=False)
20  keywords_df = pd.read_csv('keywords.csv', low_memory=False)
21
22  # 'id' sütunlarını str (string) türüne dönüştürelim
23  movies_df['id'] = movies_df['id'].astype(str)
24  credits_df['id'] = credits_df['id'].astype(str)
25  keywords_df['id'] = keywords_df['id'].astype(str)
26
```

- **Veri Çerçevelerinin Birleştirilmesi**
  Film verileri, pandas.merge() fonksiyonu kullanılarak, üç farklı veri çerçevesi birleştirilir. Bu, her filme ait detaylı bilgilerin (oyuncular, anahtar kelimeler, vb.) aynı veri çerçevesinde birleştirilmesini sağlar.

```
# Veri çerçevelerini birleştirelim
movies_with_credits_df = pd.merge(movies_df, credits_df, on='id', how='left')
movies_with_all_info_df = pd.merge(movies_with_credits_df, keywords_df, on='id', how='left')

# Cast sütununu işleyerek oyuncu isimlerini alalım
def get_cast_names(cast_column):
    try:
        cast_list = eval(cast_column)  # string'i listeye dönüştürür
        cast_names = [actor['character'] for actor in cast_list[:5]]
        return ', '.join(cast_names)
    except:
        return ''

# Keywords sütununu işleyerek anahtar kelimeleri alalım
def get_keywords(keywords_column):
    try:
        keywords_list = eval(keywords_column)  # string'i listeye dönüştürür
        keywords = [keyword['name'] for keyword in keywords_list[:5]]
        return ', '.join(keywords)
    except:
        return ''

# Cast ve Keywords sütunlarını düzenleyelim
movies_with_all_info_df['cast'] = movies_with_all_info_df['cast'].apply(get_cast_names)
movies_with_all_info_df['keywords'] = movies_with_all_info_df['keywords'].apply(get_keywords)

```

- **API Endpoint'inin Tanımlanması**
  Kullanıcıdan gelen film türü ve film yılı verilerini alır ve bu verilere göre filtreleme yaparak sonuçları döndüren API endpoint'i tanımlanmıştır. Bu endpoint, POST metodunu kullanarak veri alır ve JSON formatında yanıt döndürür.

```python
@app.route('/api/film-onerisi', methods=['POST'])
def film_onerisi():
    # Kullanıcıdan gelen veriyi alalım
    user_data = request.json
    tur = user_data.get("tur")
    yil = user_data.get("yil")

    # Tür ve Yıl bilgisi kontrolü
    if not tur or not yil:
        return jsonify({"error": "Tür ve Yıl bilgisi eksik!"}), 400

    # Tür ve Yıla göre filtreleme yapalım
    filtered_movies = movies_with_all_info_df[
        (movies_with_all_info_df['genres'].str.contains(tur, case=False, na=False)) &
        (movies_with_all_info_df['release_date'].str.contains(str(yil), na=False))
    ]

    # Eğer film bulunmazsa
    if filtered_movies.empty:
        return jsonify({"error": "Film bulunamadı!"}), 404

```

3.2 Frontend: HTML ve JavaScript

Frontend kısmı, kullanıcı etkileşimini sağlayan form ve film verilerini dinamik olarak görüntüleyen bir yapıya sahiptir. Kullanıcıdan film türü ve yılı alınarak, JavaScript Fetch API kullanılarak backend API'ye istek gönderilir ve sonuçlar dinamik olarak ekranda gösterilir.

- **HTML Formu ve Dinamik İçerik Gösterimi**
  Form, kullanıcıdan film türü ve yılı alır ve submit butonuna tıklandığında backend API'ye istek gönderir.

```html
<body>
    <h1>Film Öneri Uygulaması</h1>
    <form id="filmForm">
        <label for="tur">Film Türü:</label>
        <input type="text" id="tur" name="tur" required>

        <label for="yil">Film Yılı:</label>
        <input type="text" id="yil" name="yil" required>

        <button type="submit">Öneri Al</button>
    </form>
    <div id="filmResults"></div>
```

- **JavaScript ile API'ye Bağlantı**
  JavaScript, form verilerini alır ve fetch fonksiyonu ile Flask API'sine gönderir. Gelen sonuçlar, ekranda dinamik bir şekilde görüntülenir.

```
<script>
    document.getElementById('filmForm').addEventListener('submit', function(event) {
        event.preventDefault();

        var tur = document.getElementById('tur').value;
        var yil = document.getElementById('yil').value;

        // Flask API URL'yi doğru şekilde verin
        fetch('http://127.0.0.1:5000/api/film-onerisi', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({
                tur: tur,
                yil: yil
            })
        })
        .then(response => response.json())
        .then(data => {
            var resultDiv = document.getElementById('filmResults');
            resultDiv.innerHTML = ''; // Sonuçları temizle

            if (data.error) {
                resultDiv.innerHTML = '<p>' + data.error + '</p>';
            } else {
                var films = data.films;
                films.forEach(function(film) {
                    var filmDiv = document.createElement('div');
                    filmDiv.innerHTML = `
                        <h3>${film.title} (${film.release_date || 'Bilinmiyor'})</h3>
                        <p><strong>Tür:</strong> ${film.genres || 'Bilinmiyor'}</p>
                        <p><strong>Özet:</strong> ${film.overview || 'Özet mevcut değil'}</p>
                        <p><strong>Oyuncular:</strong> ${film.cast || 'Oyuncular bilinmiyor'}</p>
                        <p><strong>Anahtar Kelimeler:</strong> ${film.keywords || 'Anahtar kelimeler mevcut değil'}</p>
                    `;
                    resultDiv.appendChild(filmDiv);
                });
            }
        })
        .catch(error => {
            console.error('Hata:', error);
        });
    });
</script>
```

4 Veri Setinin Analizi

Veri seti, **https://www.kaggle.com** platformunda bulunan "Full MovieLens Dataset'inden" alınmış olan, yaklaşık 45,000 film hakkında detaylı bilgi sağlayan bir dizi CSV dosyasından oluşmaktadır. Bu dosyalar, çeşitli film verilerini içerir ve film öneri sistemleri için güçlü bir temel oluşturur. Veri seti, hem film metadatası hem de kullanıcı oylamaları içerdiğinden, geniş bir kullanım alanına sahiptir.

4.1 Veri Setinin İçeriği

Veri seti 7 ana dosyadan oluşmaktadır ve her dosya, farklı film bilgilerini içermektedir. Bu dosyaların detaylı açıklamaları şu şekildedir:

- **movies_metadata.csv:**
  Bu dosya, 45,000'den fazla film hakkında temel metadataları içermektedir. Filmlerle ilgili bilgiler şunları içerir:

1. Film Başlığı: Filmin adı.
2. Bütçe ve Gelir: Filmin yapım bütçesi ve elde ettiği gelir.
3. Çıkış Tarihi: Filmin vizyon tarihi.
4. Diller: Filmin hangi dillerde mevcut olduğu.
5. Yapım Ülkeleri ve Şirketleri: Filmin yapıldığı ülkeler ve prodüksiyon şirketleri.
6. Özet (Overview): Filmin kısa açıklaması.
7. Popülerlik, Oy Sayıları ve Oy Ortalamaları: Filmin halk arasında nasıl bir popülerlik kazandığı, izleyicilerden aldığı oy sayısı ve oy ortalaması.

- **keywords.csv:**
  Bu dosya, film konuları ile ilgili anahtar kelimeleri içerir. Filmler için tanımlanmış olan anahtar kelimeler, film türlerini ve içeriğini anlamada önemli rol oynar. Anahtar kelimeler bir JSON formatında saklanmaktadır.

- **credits.csv:**
  Filmin kadrosu ve ekibi hakkında bilgi içerir. Cast (oyuncu) ve crew (yönetmen, yapımcı vb.) verileri JSON formatında mevcuttur. Filmin hangi oyunculara sahip olduğu ve hangi ekip üyelerinin görev aldığı bilgileri burada yer almaktadır.

- **links.csv:**
  Bu dosya, film bilgilerini dış platformlarla eşleştirir. Her filme ait IMDB ve TMDB (The Movie Database) ID'lerini içerir. Bu bağlantılar, film hakkında daha fazla bilgi edinmek için dış veri kaynaklarını kullanmanıza olanak sağlar.

- **ratings_small.csv:**
  700'den fazla kullanıcı tarafından yapılmış 100,000'e yakın oy kaydını içerir. Her oylama, bir kullanıcının belirli bir filme verdiği puanı temsil eder. Bu veri, kullanıcı tercihlerini ve film başarılarını anlamada kullanılır.

- **links_small.csv:**
  Yine, sadece 9,000 filme ait TMDB ve IMDB ID'lerini içerir, ancak daha küçük bir alt kümedir.

4.2 Veri Seti ile Çalışmanın Zorlukları ve Çözüm Yolları

- **Eksik Veriler ve Veri Temizliği:** CSV dosyalarında eksik, bozuk veya hatalı veriler bulunabilir. Örneğin, bazı filmler için bütçe ya da gelir bilgisi mevcut olmayabilir. Bu gibi durumlarda, eksik veriler boş string'ler veya NULL değerlerle doldurulmuştur.
  - Çözüm: ChatGPT, safe_get_value fonksiyonu ile eksik ya da hatalı verileri boş string'lere dönüştürerek veri temizliği işlemini gerçekleştirmiştir. Bu, veri analizinin daha sağlıklı yapılmasını sağlar.
- **JSON Formatındaki Veriler:** Anahtar kelimeler ve oyuncu listeleri gibi bazı bilgiler JSON formatında saklanmaktadır. Bu verilerin doğru şekilde işlenmesi, listeleme ve sıralama gibi işlemler için gereklidir.
  - Çözüm: eval() fonksiyonu kullanılarak JSON verileri, Python listelerine dönüştürülüp üzerinde işlem yapılabilir hale getirilmiştir.

4.3 Veri Seti Kullanım Alanları

Bu veri seti, özellikle film öneri sistemleri için çok uygundur. Kullanıcıların daha önce izlediği filmler ve verdikleri puanlar ışığında, benzer türdeki ve aynı dönemde çıkmış filmler önerilebilir. Ayrıca, bu veri seti ile film kategorilendirmesi, kullanıcı profillemesi ve kapsamlı veri analizi yapılabilir.

5 ChatGPT'nin Uygulama Sürecine Yardımı

ChatGPT, bu film öneri uygulamasının geliştirilmesinde birçok farklı aşamada yardımcı olmuştur. Genel olarak, yazılım geliştirme sürecinde karşılaşılan bazı zorlukları aşmak, doğru teknolojileri seçmek, hataları gidermek ve verimli kodlar yazmak için rehberlik sağlanmıştır. Bu kısımdaki yardım, büyük ölçüde şu alanlarda yoğunlaşmıştır:

5.1 Veri İşleme ve Filtreleme Soruları

Veri işleme ve filtreleme, film öneri uygulamasının temel işlevlerinden biridir. Kullanıcıların belirttiği film türü ve yılına göre film verilerini filtrelemek, doğru sonuçları döndürmek için kritik bir adımdır. Bu noktada, ChatGPT'nin sağladığı birkaç yardımcı öneri ve rehberlik öne çıkmaktadır.

1. **Veri Setini Yükleme ve İşleme**

   - İlk adım olarak, uygulamanın veri setlerini yüklerken ve işleme aşamalarında karşılaşılan bazı problemler için ChatGPT'den yardım alınmıştır.
   - Özellikle, pandas kullanarak CSV dosyalarını nasıl doğru şekilde yükleyeceğimiz ve veriyi nasıl işleyip temizleyeceğimiz konusunda rehberlik alınmıştır. Örneğin, CSV dosyasındaki NaN (eksik veri) ve None (boş değerler) gibi problemleri nasıl ele alacağımız konusunda yardımcı olmuştur.

   ChatGPT Yardımı:

```python
# NaN ve None değerleri kontrol eden fonksiyon
def safe_get_value(value):
    if value is None or isinstance(value, float) and math.isnan(value):
        return ""  # Eğer None veya NaN ise, boş bir string döndürelim
    return value
```

Bu kod parçası, eksik veya geçersiz verilerin yerine boş bir değer döndürülmesini sağlayarak veri setindeki sorunları ortadan kaldırmıştır.

2. **Filtreleme ve Sorgulama**
   Kullanıcılar, film türü ve yılına göre filtreleme yapabilmelidir. Pandas DataFrame'inde bu filtrelemenin nasıl yapılacağı sorusu için ChatGPT, doğru sorgulamayı oluşturma konusunda yardımcı olmuştur.
   Örneğin, aşağıdaki kodu yazarken ChatGPT, tür ve yıl filtreleme işleminin doğru bir şekilde nasıl yapılacağına dair rehberlik sağlamıştır:

```python
# Tür ve Yıl bilgisi kontrolü
if not tur or not yil:
    return jsonify({"error": "Tür ve Yıl bilgisi eksik!"}), 400

# Tür ve Yıla göre filtreleme yapalım
filtered_movies = movies_with_all_info_df[
    (movies_with_all_info_df['genres'].str.contains(tur, case=False, na=False)) &
    (movies_with_all_info_df['release_date'].str.contains(str(yil), na=False))
]
```

Bu kod, genres ve release_date sütunlarına göre filtreleme yaparak, kullanıcının istediği türde ve yılda çıkmış filmleri veritabanından seçmeyi mümkün kılmaktadır.

3. **JSON Formatında Veri Döndürme ve API Tasarımı**
   Filtreleme işlemi başarılı bir şekilde yapıldıktan sonra, sonuçları JSON formatında frontend'e döndürmek gereklidir. Flask uygulamasında bir API endpoint'i yaratmak ve doğru formatta veri döndürmek için ChatGPT'den alınan önerilerle aşağıdaki işlevsellik sağlanmıştır.

   - **API Endpoint'i Tasarımı**
     Filtrelenen film verilerini frontend'e göndermek için bir API endpoint'i oluşturulmuştur. Kullanıcıdan gelen film türü ve yılına göre film verileri, Flask backend'inde işlendikten sonra JSON formatında frontend'e döndürülmektedir.

     ChatGPT Yardımı:

```python
@app.route('/api/film-onerisi', methods=['POST'])
def film_onerisi():
    # Kullanıcıdan gelen veriyi alalım
    user_data = request.json
    tur = user_data.get("tur")
    yil = user_data.get("yil")

    # Tür ve Yıl bilgisi kontrolü
    if not tur or not yil:
        return jsonify({"error": "Tür ve Yıl bilgisi eksik!"}), 400

    # Tür ve Yıla göre filtreleme yapalım
    filtered_movies = movies_with_all_info_df[
        (movies_with_all_info_df['genres'].str.contains(tur, case=False, na=False)) &
        (movies_with_all_info_df['release_date'].str.contains(str(yil), na=False))
    ]

    # Eğer film bulunmazsa
    if filtered_movies.empty:
        return jsonify({"error": "Film bulunamadı!"}), 404

    # Bulunan filmleri formatlayalım
    film_list = []
    for _, movie in filtered_movies.iterrows():
        film_data = {
            "title": movie["title"],
            "release_date": safe_get_value(movie["release_date"]),
            "genres": safe_get_value(movie["genres"]),
            "overview": safe_get_value(movie["overview"]),
            "cast": safe_get_value(movie["cast"]),
            "keywords": safe_get_value(movie["keywords"])
        }
        film_list.append(film_data)

    # JSON yanıtını döndürelim
    return jsonify({
        "message": f"{tur} türünde, {yil} yılına ait filmler:",
        "films": film_list
    })
```

Bu endpoint, film türü ve yılına göre filtrelenen filmleri JSON formatında döndürmektedir.

- **CORS ve API Erişimi**
  Bir diğer önemli konu, frontend ve backend arasında veri iletimi sırasında CORS (Cross-Origin Resource Sharing) politikalarının doğru şekilde yapılandırılmasıdır. Flask üzerinde CORS'u etkinleştirerek, farklı domainlerden gelen API isteklerinin doğru şekilde kabul edilmesi sağlanmıştır. Bu konuda da ChatGPT'nin önerileri, doğru yapılandırmayı yapmamıza yardımcı olmuştur.
  ChatGPT Yardımı:

```
from flask_cors import CORS

CORS(app, resources={r"/api/*": {"origins": "*"}})
```

Bu sayede, uygulama herhangi bir kaynaktan gelen API isteklerini kabul edebilir hale gelmiştir.

4. **Frontend ve Kullanıcı Arayüzü**
   Frontend tarafında, HTML ve JavaScript kullanarak basit bir form ve dinamik sonuç görüntüleme alanı oluşturulmuştur. Kullanıcı, formu doldurduktan sonra JavaScript kullanılarak, API'ye POST isteği gönderilir ve sonuçlar sayfada görüntülenir.

   - **JavaScript API İletişimi**
     JavaScript, kullanıcının girdiği bilgileri alır ve bunları backend API'ye gönderir. API'den gelen sonuçlar ise sayfada dinamik olarak gösterilir.

ChatGPT Yardımı:

```
            // Flask API URL'yi doğru şekilde verin
            fetch('http://127.0.0.1:5000/api/film-onerisi', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    tur: tur,
                    yil: yil
                })
            })
            .then(response => response.json())
            .then(data => {
                var resultDiv = document.getElementById('filmResults');
                resultDiv.innerHTML = ''; // Sonuçları temizle

                if (data.error) {
                    resultDiv.innerHTML = '<p>' + data.error + '</p>';
                } else {
                    var films = data.films;
                    films.forEach(function(film) {
                        var filmDiv = document.createElement('div');
                        filmDiv.innerHTML = `
                            <h3>${film.title} (${film.release_date || 'Bilinmiyor'})</h3>
                            <p><strong>Tür:</strong> ${film.genres || 'Bilinmiyor'}</p>
                            <p><strong>Özet:</strong> ${film.overview || 'Özet mevcut değil'}</p>
                            <p><strong>Oyuncular:</strong> ${film.cast || 'Oyuncular bilinmiyor'}</p>
                            <p><strong>Anahtar Kelimeler:</strong> ${film.keywords || 'Anahtar kelimeler mevcut değil'}</p>
                        `;
                        resultDiv.appendChild(filmDiv);
                    });
                }
            })
            .catch(error => {
                console.error('Hata:', error);
            });
        });
    </script>
</body>
</html>

```

Bu kod, backend'den gelen film verilerini alarak dinamik bir şekilde frontend'de görüntülenmesini sağlar.

5. **Kod Optimizasyonu ve Hata Giderme**
   Geliştirme sürecinde karşılaşılan bazı hatalar ve iyileştirme ihtiyaçları da ChatGPT'nin rehberliğiyle giderilmiştir. Özellikle, API'yi kullanırken bazı hata mesajları ve boş veri durumlarıyla karşılaşılmıştır.

   ChatGPT, bu sorunları çözmek için doğru hata yönetimi tekniklerini önererek, daha sağlam ve kullanıcı dostu bir uygulama geliştirilmesini sağlamıştır.

ChatGPT Yardımı:
Örneğin, eksik veri durumları ve boş sonuçlarla ilgili hata mesajları için aşağıdaki öneriyi sunmuştur:

```
65    # Tür ve Yıl bilgisi kontrolü
66    if not tur or not yil:
67        return jsonify({"error": "Tür ve Yıl bilgisi eksik!"}), 400
68
69    # Tür ve Yıla göre filtreleme yapalım
70    filtered_movies = movies_with_all_info_df[
71        (movies_with_all_info_df['genres'].str.contains(tur, case=False, na=False)) &
72        (movies_with_all_info_df['release_date'].str.contains(str(yil), na=False))
73    ]
74
75    # Eğer film bulunmazsa
76    if filtered_movies.empty:
77        return jsonify({"error": "Film bulunamadı!"}), 404
78
```

Bu tür hata yönetimi kodları, kullanıcıya daha anlamlı ve açıklayıcı hata mesajları sunarak, kullanıcı deneyimini geliştirmiştir.

6 Adım Adım Uygulamanın Çalıştırılması

Bu uygulama, iki ana bileşenden oluşuyor:

1. **Backend (Flask Uygulaması):**
   Sunucu tarafında çalışan Python kodu
2. **Frontend (HTML ve JavaScript):**
   Kullanıcı arayüzü

6.1 Gerekli Programları Yüklemek

1. Python Yükleme
2. Gerekli Python Kütüphanelerini Yükleme
   Python, bazı kütüphanelere ihtiyaç duyacak. Bu kütüphaneleri yüklemek için terminal ya da komut satırını kullanabilirsiniz.
   - pip install flask
   - pip install pandas
   - pip install flask-cors

6.2 Proje Klasörüne Git

Terminal ya da komut satırında Kodun bulunduğu klasöre gitmelisin.

```
C:\Users\90546>cd C:\Users\90546\Desktop\FilmOneriUygulama

C:\Users\90546\Desktop\FilmOneriUygulama>
```

6.3 Sanal ortamını aktif edin:

(a) Sanal ortamını (veya virtual environment), Python projelerinde bağımlılıkların ve kütüphanelerin izole bir şekilde yönetilmesini sağlamak amacıyla kullanılan bir yapıdır. Sanal ortam, projenizin dışında yüklü olan Python kütüphanelerinin projeyi etkilemesini engeller ve her projeye özgü bağımlılıkların yönetilmesine imkan tanır.

i. Sanal Ortamı Aktif Etme Adımları:

A. Sanal ortamı oluşturun:

- python -m venv aker

Burada "aker" sanal ortamın adıdır. İstediğiniz başka bir isim verebilirsiniz.

B. Sanal ortamı aktive edin:

- Windows:
  - aker-Scripts-activate
- macOS/Linux:
  - source aker/bin/activate

C. Sanal ortamı devre dışı bırakma (deactivate):

- deactivate

```
C:\Users\90546\Desktop\FilmOneriUygulama>python -m venv aker

C:\Users\90546\Desktop\FilmOneriUygulama>aker\Scripts\activate

(aker) C:\Users\90546\Desktop\FilmOneriUygulama>
```

6.4 Flask Uygulamasını Çalıştırma (Backend)

(a) Sanal ortamda Flask kütüphanesini yüklemen gerekiyor.

- pip install flask flask-cors pandas

```
(aker) C:\Users\90546\Desktop\FilmOneriUygulama>pip install flask flask-cors pandas
Collecting flask
```

(b) Backend Kodu Yükleme
Python ile yazılmış backend kodunu bir dosyaya kaydedin. Var sayılan olarak dosya adı app1.py olarak verilmiştir.

(c) CSV dosyalarınız (örneğin, movies_metadata.csv, credits.csv, keywords.csv) aynı klasöre yerleştirin. Bu dosyalar uygulamanızın çalışabilmesi için gereklidir. Bu dosyaların içeriği, uygulamanın doğru şekilde çalışabilmesi için çok önemlidir.

3. Uygulamayı Başlatma

(a) Komut satırını (ya da terminali) açın.

(b) Backend Kodu(app1.py) dosyasının bulunduğu klasöre gidin.

4. Uygulamayı başlatmak için şu komutu yazın:

(a) python app.1.py

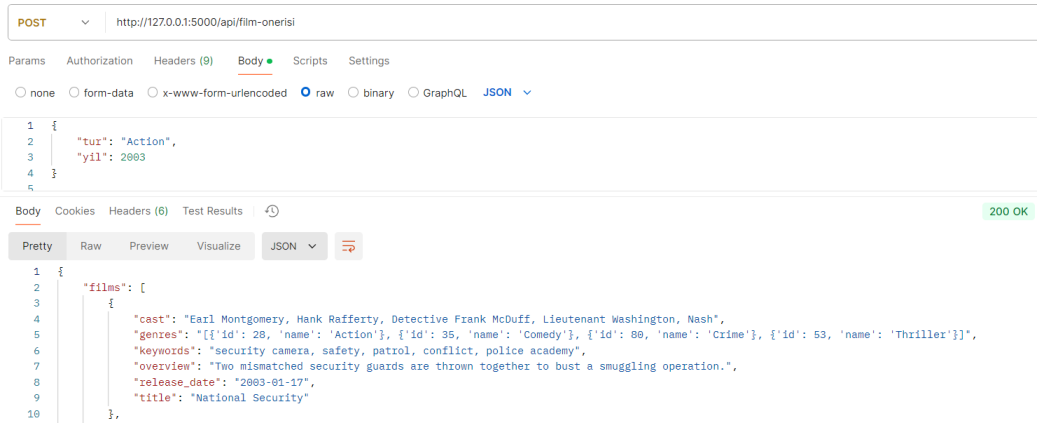
Eğer her şey doğru bir şekilde yüklendiyse, terminalde şu çıktıyı görmelisiniz:

```
(aker) C:\Users\90546\Desktop\FilmOneriUygulama>python app.1.py
 * Serving Flask app 'app.1'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 992-966-262
```

Bu, Flask uygulamanızın başarıyla başlatıldığı ve yerel sunucunuzun **http://127.0.0.1:5000/** adresinde çalıştığı anlamına gelir.

6.5 API'yi Test Et

1. Postman'ı indir ve kur (Eğer yüklü değilse).
2. Postman'ı aç, New butonuna tıkla ve Request seç.
3. İstek için bir isim ver ve Save butonuna tıkla.
4. Yöntemi POST olarak seç.
5. URL kısmına şu adresi yaz:
   http://127.0.0.1:5000/api/film-onerisi
6. Body kısmına geç, raw seçeneğini seç ve JSON formatını seç.
7. JSON verisini gir:
8. Send butonuna tıklayarak isteği gönder.
9. API'nin yanıtını ekranda kontrol et.



6.6 Frontend (HTML ve JavaScript) Dosyasını Çalıştırma

1. HTML Dosyasını Açma

   (a) **FilmOneriUygulama/templates** klasörüne git ve index.html dosyasının orada olduğundan emin ol. index.html dosyasını herhangi bir web tarayıcısında açabilirsiniz. Bunun için index.html dosyasına çift tıklayarak dosyayı açın. Eğer doğru kaydedildiyse, tarayıcıda basit bir form görmelisiniz.

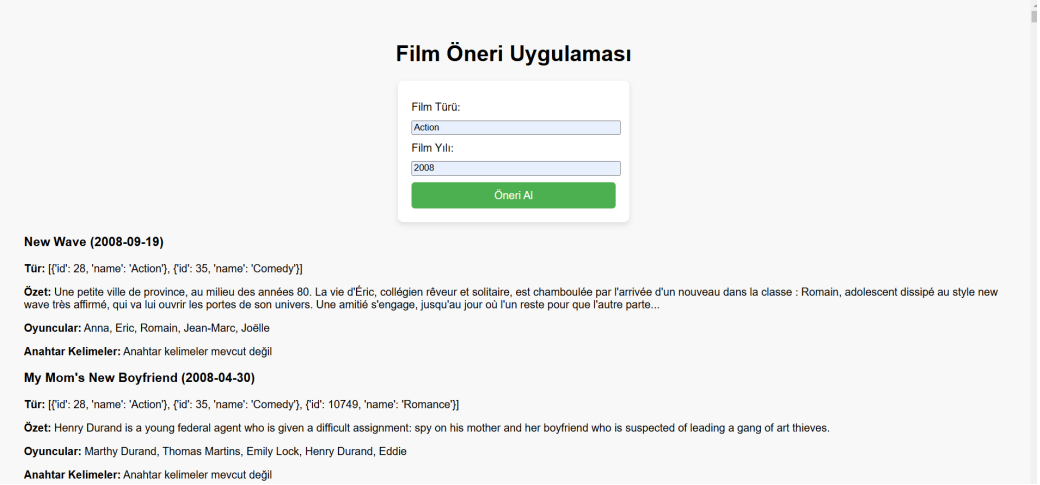
2. Uygulama ile Etkileşim

   (a) Bu formda, "Film Türü" ve "Film Yılı" girmenizi isteyen iki kutucuk olacak.

   (b) Film türünü ve yılını girin. Örneğin

   - Film Türü: Action
   - Film Yılı: 2008

   (c) "Öneri Al" butonuna bastığınızda, frontend kısmındaki JavaScript, Flask API'nize (Backend kısmı) istek gönderir ve yanıt olarak önerilen filmleri alır.



6.7 Film Önerisi Almak

1. Film türünü ve yılını girdikten sonra "Öneri Al" butonuna tıklayın.

2. Web tarayıcınız, Flask sunucusuna bir istek gönderir.

3. Flask, CSV dosyalarını inceleyerek girilen tür ve yıl bilgilerine uygun filmleri arar.

4. Bulunan filmler, tarayıcıda kullanıcıya gösterilecektir. Bu filmlerin başlıkları, tarihleri, türleri, oyuncu listeleri ve anahtar kelimeleri gibi bilgileri görmelisiniz.

7 Sonuç ve Öneriler

Film öneri uygulaması, kullanıcıların film keşfetme deneyimini geliştirirken, doğru veri seti ile güçlendirilmiş etkili bir sistem sunmaktadır. Uygulama, doğru şekilde yapılandırılmış ve kullanıcı dostu bir arayüzle kullanıcıların ihtiyaçlarına hızlıca cevap verebilmektedir.

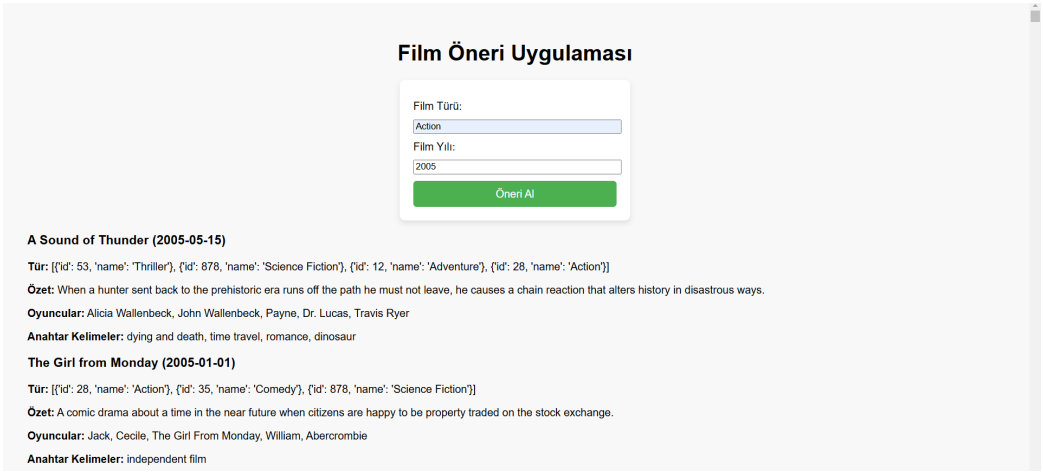
Öneriler:

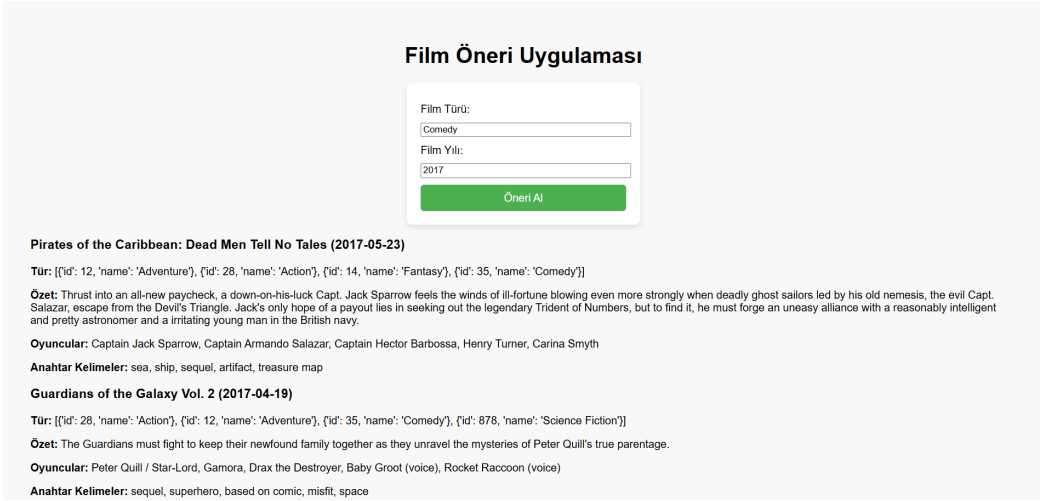
- Daha Fazla Kriter Eklenmesi: Uygulama sadece tür ve yıl bazında filtreleme yapmaktadır. Kullanıcılar için ek filtreleme seçenekleri (örneğin, oyuncular, bütçe, gelir) eklenebilir.
- Veri Setinin Güncellenmesi: Veri seti 2017 yılına kadar olan verileri içermektedir. Daha güncel veriler eklenebilir.
- Kişiselleştirilmiş Öneriler: Kullanıcıların daha önceki tercihlerine göre kişiselleştirilmiş öneriler sunulabilir.

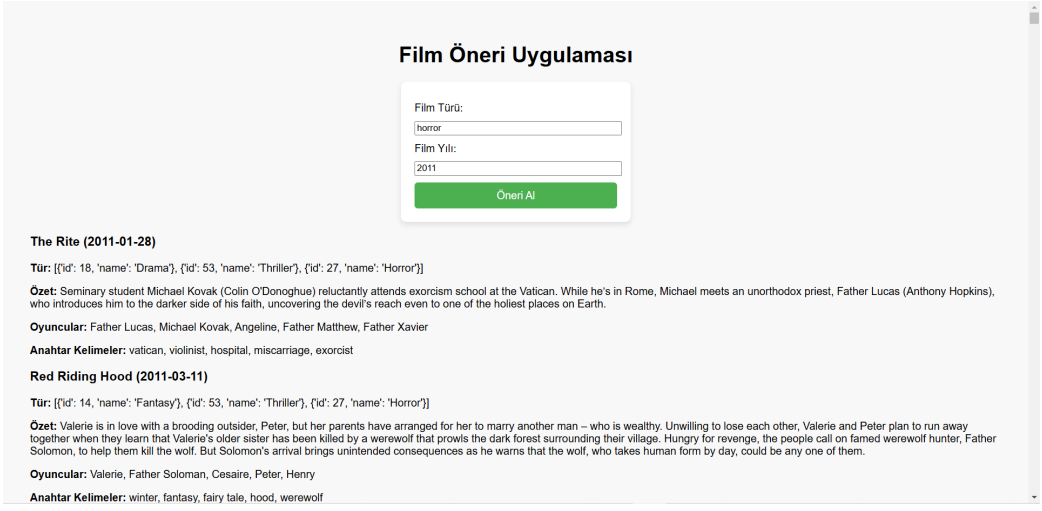
8 KAYNAKÇA

- https://chatgpt.com/
- https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset
- https://student.oys.ksbu.edu.tr/
- Learn to Code using AI - ChatGPT Programming Tutorial https://www.youtube.com/watch?v=dJhlMn2otxA
- Use ChatGPT to Code a Full Stack App https://www.youtube.com/watch?v=GizsSo-EevA
- https://www.youtube.com/@erhanmeydan

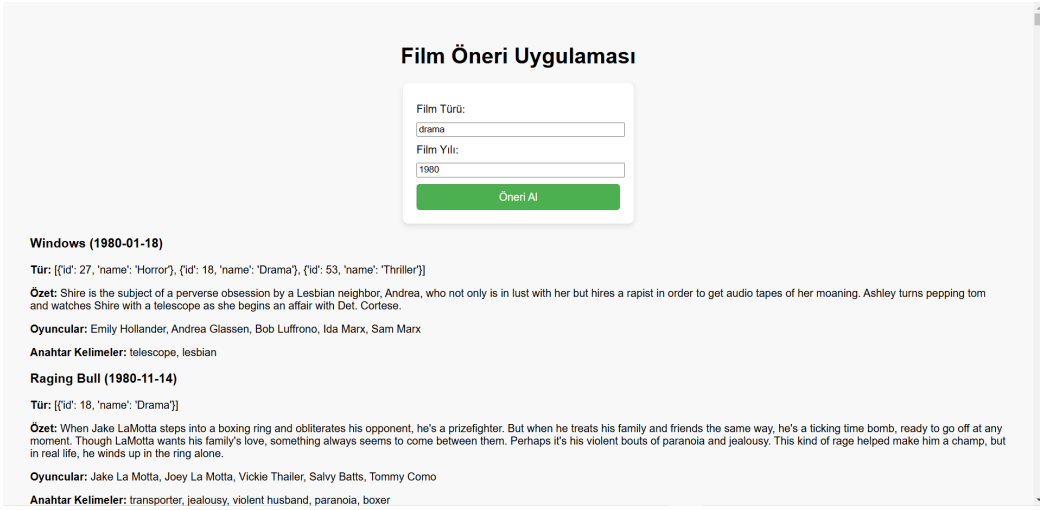
9 EKLER

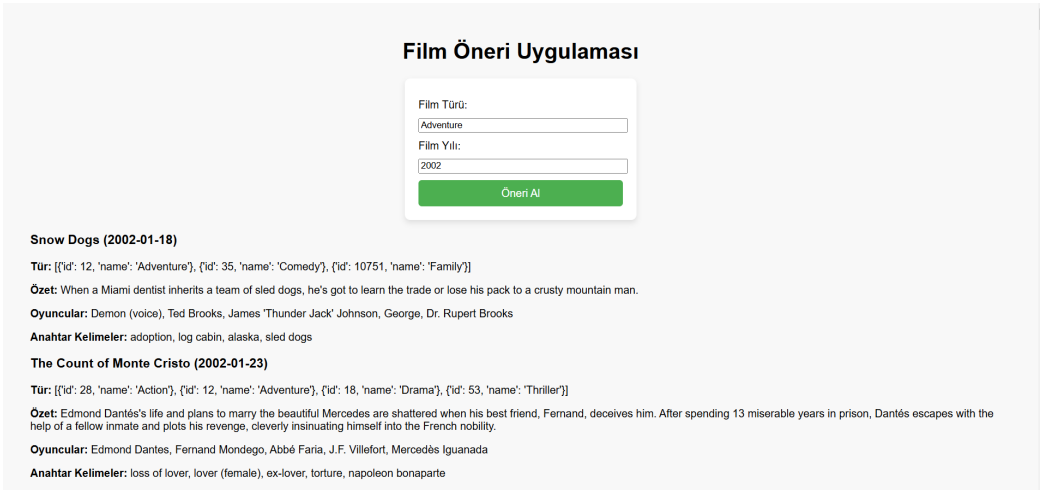

Film Öneri Uygulaması
Film Türü:
Action
Film Yılı:
2005
Öneri Al
A Sound of Thunder (2005-05-15)
Tür: [{'id': 53, 'name': 'Thriller'}, {'id': 878, 'name': 'Science Fiction'}, {'id': 12, 'name': 'Adventure'}, {'id': 28, 'name': 'Action'}]
Özet: When a hunter sent back to the prehistoric era runs off the path he must not leave, he causes a chain reaction that alters history in disastrous ways.
Oyuncular: Alicia Wallenbeck, John Wallenbeck, Payne, Dr. Lucas, Travis Ryer
Anahtar Kelimeler: dying and death, time travel, romance, dinosaur
The Girl from Monday (2005-01-01)
Tür: [{'id': 28, 'name': 'Action'}, {'id': 35, 'name': 'Comedy'}, {'id': 878, 'name': 'Science Fiction'}]
Özet: A comic drama about a time in the near future when citizens are happy to be property traded on the stock exchange.
Oyuncular: Jack, Cecile, The Girl From Monday, William, Abercrombie
Anahtar Kelimeler: independent film


Film Öneri Uygulaması
Film Türü:
Comedy
Film Yılı:
2017
Öneri Al
Pirates of the Caribbean: Dead Men Tell No Tales (2017-05-23)
Tür: [{'id': 12, 'name': 'Adventure'}, {'id': 28, 'name': 'Action'}, {'id': 14, 'name': 'Fantasy'}, {'id': 35, 'name': 'Comedy'}]
Özet: Thrust into an all-new paycheck, a down-on-his-luck Capt. Jack Sparrow feels the winds of ill-fortune blowing even more strongly when deadly ghost sailors led by his old nemesis, the evil Capt. Salazar, escape from the Devil's Triangle. Jack's only hope of a payout lies in seeking out the legendary Trident of Numbers, but to find it, he must forge an uneasy alliance with a reasonably intelligent and pretty astronomer and a irritating young man in the British navy.
Oyuncular: Captain Jack Sparrow, Captain Armando Salazar, Captain Hector Barbossa, Henry Turner, Carina Smyth
Anahtar Kelimeler: sea, ship, sequel, artifact, treasure map
Guardians of the Galaxy Vol. 2 (2017-04-19)
Tür: [{'id': 28, 'name': 'Action'}, {'id': 12, 'name': 'Adventure'}, {'id': 35, 'name': 'Comedy'}, {'id': 878, 'name': 'Science Fiction'}]
Özet: The Guardians must fight to keep their newfound family together as they unravel the mysteries of Peter Quill's true parentage.
Oyuncular: Peter Quill / Star-Lord, Gamora, Drax the Destroyer, Baby Groot (voice), Rocket Raccoon (voice)
Anahtar Kelimeler: sequel, superhero, based on comic, misfit, space


Film Öneri Uygulaması
Film Türü:
horror
Film Yılı:
2011
Öneri Al
The Rite (2011-01-28)
Tür: [{'id': 18, 'name': 'Drama'}, {'id': 53, 'name': 'Thriller'}, {'id': 27, 'name': 'Horror'}]
Özet: Seminary student Michael Kovak (Colin O'Donoghue) reluctantly attends exorcism school at the Vatican. While he's in Rome, Michael meets an unorthodox priest, Father Lucas (Anthony Hopkins), who introduces him to the darker side of his faith, uncovering the devil's reach even to one of the holiest places on Earth.
Oyuncular: Father Lucas, Michael Kovak, Angeline, Father Matthew, Father Xavier
Anahtar Kelimeler: vatican, violinist, hospital, miscarriage, exorcist
Red Riding Hood (2011-03-11)
Tür: [{'id': 14, 'name': 'Fantasy'}, {'id': 53, 'name': 'Thriller'}, {'id': 27, 'name': 'Horror'}]
Özet: Valerie is in love with a brooding outsider, Peter, but her parents have arranged for her to marry another man – who is wealthy. Unwilling to lose each other, Valerie and Peter plan to run away together when they learn that Valerie's older sister has been killed by a werewolf that prowls the dark forest surrounding their village. Hungry for revenge, the people call on famed werewolf hunter, Father Solomon, to help them kill the wolf. But Solomon's arrival brings unintended consequences as he warns that the wolf, who takes human form by day, could be any one of them.
Oyuncular: Valerie, Father Soloman, Cesaire, Peter, Henry
Anahtar Kelimeler: winter, fantasy, fairy tale, hood, werewolf


Film Öneri Uygulaması
Film Türü:
science fiction
Film Yılı:
2020
Öneri Al
Avatar 2 (2020-12-16)
Tür: [{'id': 28, 'name': 'Action'}, {'id': 12, 'name': 'Adventure'}, {'id': 14, 'name': 'Fantasy'}, {'id': 878, 'name': 'Science Fiction'}]
Özet: A sequel to Avatar (2009).
Oyuncular: Ronal, Neytiri, Jake Sully, Grace Augustine, Varang
Anahtar Kelimeler: sequel, alien planet, avatar


Film Öneri Uygulaması
Film Türü:
drama
Film Yılı:
1980
Öneri Al
Windows (1980-01-18)
Tür: [{'id': 27, 'name': 'Horror'}, {'id': 18, 'name': 'Drama'}, {'id': 53, 'name': 'Thriller'}]
Özet: Shire is the subject of a perverse obsession by a Lesbian neighbor, Andrea, who not only is in lust with her but hires a rapist in order to get audio tapes of her moaning. Ashley turns pepping tom and watches Shire with a telescope as she begins an affair with Det. Cortese.
Oyuncular: Emily Hollander, Andrea Glassen, Bob Luffrono, Ida Marx, Sam Marx
Anahtar Kelimeler: telescope, lesbian
Raging Bull (1980-11-14)
Tür: [{'id': 18, 'name': 'Drama'}]
Özet: When Jake LaMotta steps into a boxing ring and obliterates his opponent, he's a prizefighter. But when he treats his family and friends the same way, he's a ticking time bomb, ready to go off at any moment. Though LaMotta wants his family's love, something always seems to come between them. Perhaps it's his violent bouts of paranoia and jealousy. This kind of rage helped make him a champ, but in real life, he winds up in the ring alone.
Oyuncular: Jake La Motta, Joey La Motta, Vickie Thailer, Salvy Batts, Tommy Como
Anahtar Kelimeler: transporter, jealousy, violent husband, paranoia, boxer


Film Öneri Uygulaması
Film Türü:
Adventure
Film Yılı:
2002
Öneri Al
Snow Dogs (2002-01-18)
Tür: [{'id': 12, 'name': 'Adventure'}, {'id': 35, 'name': 'Comedy'}, {'id': 10751, 'name': 'Family'}]
Özet: When a Miami dentist inherits a team of sled dogs, he's got to learn the trade or lose his pack to a crusty mountain man.
Oyuncular: Demon (voice), Ted Brooks, James 'Thunder Jack' Johnson, George, Dr. Rupert Brooks
Anahtar Kelimeler: adoption, log cabin, alaska, sled dogs
The Count of Monte Cristo (2002-01-23)
Tür: [{'id': 28, 'name': 'Action'}, {'id': 12, 'name': 'Adventure'}, {'id': 18, 'name': 'Drama'}, {'id': 53, 'name': 'Thriller'}]
Özet: Edmond Dantés's life and plans to marry the beautiful Mercedes are shattered when his best friend, Fernand, deceives him. After spending 13 miserable years in prison, Dantés escapes with the help of a fellow inmate and plots his revenge, cleverly insinuating himself into the French nobility.
Oyuncular: Edmond Dantes, Fernand Mondego, Abbé Faria, J.F. Villefort, Mercedès Iguanada
Anahtar Kelimeler: loss of lover, lover (female), ex-lover, torture, napoleon bonaparte