Yapay Zeka

Proje Raporu

Şarkı Öneri Sistemi

Hazırlayan

Alican Selen

11511092842

**Özet**

Bu projede, Spotify'dan alınan bir veri kümesi kullanılarak şarkı öneri sistemi geliştirilmiştir. Projenin amacı, kullanıcıların sevdikleri şarkılara benzer yeni şarkılar keşfetmelerine yardımcı olmaktır. İlk olarak, veri kümesi yüklenmiş, incelenmiş ve veri temizleme adımları uygulanmıştır. Boş değerler ele alınmış ve gereksiz sütunlar kaldırılmıştır. Metin verileri üzerinde ön işleme adımları gerçekleştirilmiş, bu kapsamda tüm metinler küçük harfe çevrilmiş, tokenizasyon ve stemming işlemleri yapılmıştır. Bu adımların ardından, TF-IDF (Term Frequency-Inverse Document Frequency) vektörizasyonu kullanılarak şarkı sözleri vektörleştirilmiştir. Şarkılar arasındaki benzerlikler, kosinüs benzerliği metrikleri kullanılarak hesaplanmıştır. Son olarak, belirli bir şarkının indeksini bulan ve bu şarkıya benzer diğer şarkıları öneren bir fonksiyon geliştirilmiştir. Öneri sistemi, kullanıcının seçtiği şarkıya en benzer şarkıları listelemektedir. Bu projenin sonucu olarak, kullanıcıların mevcut favori şarkılarına benzer yeni şarkılar keşfetmelerini sağlayan etkili bir şarkı öneri sistemi oluşturulmuştur. Bu projede elde edilen bulgular, şarkı öneri sistemlerinin müzik keşfi süreçlerinde ne kadar etkili olabileceğini göstermektedir. Gelecekte, bu sistemin daha fazla iyileştirilmesi ve farklı müzik veri kümeleriyle test edilmesi planlanmaktadır.

**Kullanılan Teknolojiler**

**Programlama Dili**

**Python:**

Python, veri bilimi, makine öğrenmesi ve yapay zeka projelerinde yaygın olarak kullanılan, yüksek seviyeli ve esnek bir programlama dilidir. Bu projede, veri işleme, modelleme ve analiz adımlarının tümünde Python kullanılmıştır. Python’un geniş kütüphane desteği, veri bilimciler ve araştırmacılar için güçlü araçlar sunmaktadır.

**Geliştirme Ortamları**

**Jupyter Lab:**

Jupyter Lab, etkileşimli veri bilimi ve bilimsel bilgi işlem için popüler bir geliştirme ortamıdır. Kullanıcıların kod yazmalarına, çalıştırmalarına ve sonuçları görsel olarak incelemelerine olanak tanır. Bu projede, veri analizi ve model geliştirme aşamalarında Jupyter Lab kullanılmıştır. Not defteri formatı, kod, metin, görseller ve diğer medya türlerinin bir arada kullanılmasını sağlar, bu da analiz sürecini daha anlaşılır ve tekrarlanabilir kılar.

**Visual Studio Code (VS Code):**

Visual Studio Code, çok çeşitli uzantılarla desteklenen, hafif ama güçlü bir metin editörüdür. Python entegrasyonu ve zengin özellik seti, kod yazmayı ve hata ayıklamayı kolaylaştırır. Bu projede, uygulama geliştirme ve kod düzenleme işlemleri için VS Code kullanılmıştır. VS Code’un uzantıları sayesinde, Python kodu yazarken otomatik tamamlama, hata ayıklama ve test etme gibi işlemler oldukça kolay hale gelir.

**Python Kütüphaneleri ve Araçlar**

**Pandas:**

Pandas, veri manipülasyonu ve analiz için kullanılan güçlü bir Python kütüphanesidir. Veri yapıları ve veri analizi araçları sağlar. Pandas, veri temizleme, dönüştürme ve analiz etme işlemlerini kolaylaştırır. Bu projede, veri kümesini yüklemek, incelemek ve temizlemek için Pandas kullanılmıştır. Pandas’ın DataFrame yapısı, verileri tablo formatında tutarak, SQL benzeri işlemler yapmayı mümkün kılar.

**NLTK (Natural Language Toolkit):**

NLTK, doğal dil işleme için kullanılan bir kütüphanedir. Metin verilerini analiz etmek ve işlemek için çeşitli araçlar ve veri kümeleri sunar. Bu projede, metin verilerinin tokenizasyonu ve stemming işlemleri için NLTK kullanılmıştır. NLTK’nin sağladığı araçlar, metinleri kelime düzeyinde işleyerek, makine öğrenmesi modelleri için uygun hale getirir.

**Scikit-learn:**

Scikit-learn, makine öğrenmesi algoritmaları ve veri ön işleme araçları sunan bir Python kütüphanesidir. Çeşitli sınıflandırma, regresyon, kümeleme algoritmaları ve veri dönüşüm araçları içerir. Bu projede, TF-IDF vektörizasyonu ve kosinüs benzerliği hesaplamaları için Scikit-learn kullanılmıştır. Scikit-learn’ün kullanımı kolay arayüzü, karmaşık makine öğrenmesi algoritmalarını basit kodlarla uygulamayı mümkün kılar.

**Pickle:**

Pickle, Python nesnelerinin serileştirilmesi ve saklanması için kullanılan bir modüldür. Python nesnelerini bir dosyaya kaydetmek ve daha sonra bu dosyadan geri yüklemek için kullanılır. Bu projede, model ve veri setinin saklanması için Pickle kullanılmıştır. Pickle, nesneleri bayt dizilerine dönüştürerek, kalıcı depolama ve yeniden yükleme işlemlerini kolaylaştırır.

**Spotipy:**

Spotipy, Spotify Web API ile etkileşime geçmek için kullanılan bir Python kütüphanesidir. Spotify verilerine erişim ve bu veriler üzerinde işlem yapmayı sağlar. Bu proje kapsamında, Spotify’dan şarkı verilerini çekmek ve analiz etmek için Spotipy kullanılmıştır. Spotipy, kullanıcıların Spotify API’sine kolayca erişmelerini ve çeşitli müzik verilerini çekmelerini sağlar.

**Streamlit:**

Streamlit, makine öğrenmesi ve veri bilimi projeleri için hızlı ve kolay web uygulamaları geliştirmeyi sağlayan bir Python kütüphanesidir. Kullanıcı dostu arayüzü ve etkileşimli bileşenleri ile veri bilimi projelerini paylaşmayı kolaylaştırır. Bu projede, kullanıcıların şarkı önerilerini kolayca alabilmesi için Streamlit kullanılmıştır. Streamlit, verilerin ve model sonuçlarının görsel olarak sunulmasını ve kullanıcıların interaktif uygulamalar geliştirmesini sağlar.

**Veri Yükleme ve İnceleme**

İlk adımda, proje için gerekli olan veri seti yüklenmiş ve incelenmiştir. Bu aşamada, veri setinin boyutu ve yapısı hakkında bilgi edinilmiş ve eksik değerler kontrol edilmiştir.

**metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Veri Ön İşleme**

Veri seti yüklendikten sonra, veri ön işleme adımları uygulanmıştır. Bu aşamada, veri setinden örneklem alınmış, gereksiz sütunlar çıkarılmış ve metin verileri temizlenmiştir. Ayrıca, metin verileri küçük harfe dönüştürülmüş ve satırlardaki gereksiz yeni satır karakterleri (\n) kaldırılmıştır.

**metin, yazı tipi, çizgi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Metin Tokenizasyonu ve Kök Bulma**

Bu aşamada, metin verileri kelime kelime ayrılarak tokenleştirilmiş ve köklerine indirgenmiştir. Bu işlemler, metin verilerini makine öğrenmesi modelleri için daha uygun hale getirmektedir. Tokenizasyon ve kök bulma işlemleri için NLTK kütüphanesi kullanılmıştır.

**metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**TF-IDF Vektörizasyonu**

Metin verilerini sayısal değerlere dönüştürmek için TF-IDF (Term Frequency-Inverse Document Frequency) vektörizasyonu kullanılmıştır. Bu yöntem, metin verilerindeki kelimelerin önemini ölçerek, kelimeleri ağırlıklı sayısal vektörlere dönüştürür. Bu işlem için Scikit-learn kütüphanesi kullanılmıştır.

**metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Kosinüs Benzerliği Hesaplama**

TF-IDF vektörleri oluşturulduktan sonra, bu vektörler arasındaki benzerlikleri ölçmek için kosinüs benzerliği kullanılmıştır. Kosinüs benzerliği, iki vektör arasındaki açıyı ölçerek, ne kadar benzer olduklarını belirler. Bu işlem de Scikit-learn kütüphanesi kullanılarak gerçekleştirilmiştir.

**metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Şarkı Öneri Fonksiyonunun Geliştirilmesi**

Belirli bir şarkıya benzer şarkıları önermek için bir fonksiyon geliştirilmiştir. Bu fonksiyon, verilen şarkının indeksini bulur ve benzerlik matrisini kullanarak en benzer şarkıları sıralar. Daha sonra, en benzer birkaç şarkıyı öneri olarak döner.

**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Modelin ve Verilerin Saklanması**

Model ve veri seti, daha sonra tekrar kullanılmak üzere serileştirilmiş ve dosyalara kaydedilmiştir. Bu işlem için pickle modülü kullanılmıştır. Pickle, Python nesnelerinin bayt dizilerine dönüştürülmesini sağlar ve bu bayt dizileri dosyalara kaydedilebilir.

**metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Web Uygulaması Geliştirme**

Projenin son aşamasında, kullanıcıların belirli bir şarkıya benzer şarkılar önerilerini alabilmesi için bir web uygulaması geliştirildi. Bu uygulama, Streamlit kütüphanesi kullanılarak oluşturuldu ve Spotipy kütüphanesi kullanılarak Spotify API ile entegre edildi.

**Spotify API ile Bağlantı:**

Spotify API'ına bağlanmak için gerekli kimlik bilgileri (CLIENT\_ID ve CLIENT\_SECRET) kullanılarak bir Spotipy istemcisi oluşturuldu.

**metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Albüm Kapak URL'lerinin Alınması:**

Belirli bir şarkının albüm kapak resmini almak için bir fonksiyon tanımlandı. Bu fonksiyon, şarkı adı ve sanatçı adı kullanarak Spotify API'ında arama yapar ve ilgili albüm kapağının URL'sini döner.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Şarkı Öneri Fonksiyonu:**

Kullanıcının girdiği şarkıya benzer şarkıları öneren bir fonksiyon tanımlandı. Bu fonksiyon, önceden hesaplanan benzerlik matrisini kullanarak en benzer şarkıları bulur ve bu şarkıların albüm kapaklarını getirir.

**metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Streamlit Arayüzü:**

Streamlit kullanılarak bir web arayüzü oluşturuldu. Kullanıcılar, bu arayüzde şarkı adlarını girip benzer şarkıları görebilirler. Şarkı isimleri bir açılır menüde listelenir ve seçilen şarkıya benzer şarkılar butona tıklandığında gösterilir.

**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Sonuçlar ve Değerlendirme**

Bu proje, Spotify müzik veritabanındaki şarkılar arasındaki benzerlikleri tespit etmek ve kullanıcıların sevdikleri bir şarkıya benzer diğer şarkıları keşfetmelerini sağlamak amacıyla gerçekleştirilmiştir. Proje kapsamında aşağıdaki adımlar izlenmiştir:

**Veri Toplama ve Ön İşleme:**

* Spotify müzik veritabanından çekilen veriler kullanılarak bir veri kümesi oluşturuldu.
* Veri kümesi, şarkıların çeşitli özelliklerini içermekteydi (örneğin, şarkı adı, sanatçı adı, albüm adı, vs.).
* Metin verileri temizlenerek, küçük harfe dönüştürme ve özel karakterlerin kaldırılması gibi işlemler yapıldı.
* NLTK kütüphanesi kullanılarak metin verileri tokenize edilip köklerine indirildi.

**Model Eğitimi:**

* TF-IDF vektörleştirici kullanılarak şarkı metinleri vektörleştirildi.
* Bu vektörler kullanılarak şarkılar arasındaki benzerlikler hesaplandı ve kosinüs benzerliği matrisi oluşturuldu.

**Web Uygulaması Geliştirme:**

* Kullanıcıların belirli bir şarkıya benzer şarkıları görebileceği bir web arayüzü oluşturmak için Streamlit kullanıldı.
* Spotipy kütüphanesi kullanılarak Spotify API ile entegrasyon sağlandı ve şarkıların albüm kapakları elde edildi.
* Kullanıcıların şarkı seçip benzer şarkıları görmelerini sağlayan interaktif bir uygulama geliştirildi.

**Şarkı Öneri Sistemi:**

* Kullanıcının seçtiği şarkıya en benzer diğer şarkılar önerildi.
* Önerilen şarkıların albüm kapakları da gösterildi, böylece kullanıcılar görsel olarak da bilgi sahibi oldular.

**Genel Değerlendirme**

Proje başarılı bir şekilde tamamlanmış olup, kullanıcıların belirli bir şarkıya benzer şarkıları keşfetmelerine olanak tanıyan bir sistem geliştirilmiştir. Bu sistem, müzik zevklerine göre yeni şarkılar keşfetmek isteyen kullanıcılar için kullanışlı bir araç sunmaktadır. Ayrıca, kullanılan teknikler ve araçlar sayesinde, benzer projelerin başka veri kümeleri veya başka alanlarda da uygulanabilirliği gösterilmiştir.

**Gelecek Çalışmalar**

Bu projede yapılan çalışmaların daha da geliştirilmesi ve genişletilmesi için bazı öneriler şunlardır:

* **Kullanıcı Geri Bildirimleri:** Kullanıcıların önerilen şarkılar hakkında geri bildirimde bulunabilmesi sağlanarak, öneri sisteminin performansı artırılabilir.
* **Genişletilmiş Veri Kümesi:** Daha büyük ve daha çeşitli bir veri kümesi kullanılarak, öneri sisteminin kapsama alanı genişletilebilir.
* **Derin Öğrenme Teknikleri:** Derin öğrenme ve doğal dil işleme (NLP) teknikleri kullanılarak daha karmaşık modeller oluşturulabilir ve öneri sisteminin doğruluğu artırılabilir.

Proje, müzik öneri sistemlerinin temel prensiplerini ve uygulanabilirliğini göstermiş olup, bu alandaki diğer çalışmalar için bir temel oluşturmaktadır.

**Kaynakça**

1. **Jupyter.** "Jupyter Lab." Erişim tarihi: Temmuz 2024, <https://jupyter.org/>.
2. **Microsoft.** "Visual Studio Code." Erişim tarihi: Temmuz 2024, <https://code.visualstudio.com/>.
3. **Pandas Development Team.** "Pandas Documentation." Erişim tarihi: Temmuz 2024, <https://pandas.pydata.org/docs/>.
4. **Bird, S., Klein, E., & Loper, E.** (2009). Natural Language Processing with Python. O'Reilly Media. Erişim tarihi: Temmuz 2024, <https://www.nltk.org/book/>.
5. **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J.** (2011). "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research, 12, 2825-2830. Erişim tarihi: Temmuz 2024, <https://scikit-learn.org/>.
6. **Spotify.** "Spotify Web API." Erişim tarihi: Temmuz 2024, <https://developer.spotify.com/documentation/web-api/>.
7. **Pickle.** "Python Object Serialization." Erişim tarihi: Temmuz 2024, <https://docs.python.org/3/library/pickle.html>.
8. **Streamlit.** "Streamlit Documentation." Erişim tarihi: Temmuz 2024, <https://docs.streamlit.io/>.
9. **Pypa.** "Spotipy." Erişim tarihi: Temmuz 2024, <https://spotipy.readthedocs.io/en/2.16.1/>.
10. **Streamlit kullanım örneği** <https://www.youtube.com/watch?v=qQVAgaGTBHU>
11. **JupyterLab Kurulumu** <https://youtu.be/-nx5l2A3hBs?si=JHgrU39WdVUpYR4E>